Beyond Human Data: Scaling Self-Training for Problem-Solving with Language Models

Avi Singh[1,*], John D Co-Reyes[1,*], Rishabh Agarwal[1,2,*], Ankesh Anand[1], Piyush Patil[1], Xavier Garcia[1], Peter J. Liu[1], James Harrison[1], Jaehoon Lee[1], Kelvin Xu[1], Aaron Parisi[1], Abhishek Kumar[1], Alex Alemi[1], Alex Rizkowsky[1], Azade Nova[1], Ben Adlam[1], Bernd Bohnet[1], Gamaleldin Elsayed[1], Hanie Sedghi[1], Igor Mordatch[1], Isabelle Simpson[1], Izzeddin Gur[1], Jasper Snoek[1], Jeffrey Pennington[1], Jiri Hron[1], Kathleen Kenealy[1], Kevin Swersky[1], Kshiteej Mahajan[1], Laura Culp[1], Lechao Xiao[1], Maxwell L Bileschi[1], Noah Constant[1], Roman Novak[1], Rosanne Liu[1], Tris Warkentin[1], Yundi Qian[1], Yamini Bansal[1], Ethan Dyer[1], Behnam Neyshabur[1], Jascha Sohl-Dickstein[1], Noah Fiedel[1]

Abhinav & Rajat

Introduction

- Large language models (LLMs) excel in various tasks but rely heavily on human-generated data.
- Human data is costly and scarce, especially for complex problem-solving tasks (e.g., math, coding).
- This paper proposes **ReST-EM**, a self-training method using expectation-maximization (EM) and reinforcement learning (RL) to reduce dependence on human data.
- Significance: Improves LLMs with minimal human input, leveraging scalar feedback.

Related and Past Work

Expert Iteration (ExiT):
Uses search/MCTS for expert sample generation, then distills into the base model. ReSTEM replaces search with temperature sampling.

Self-Taught Reasoner (STaR):
Employs greedy decoding and rationalization, though rationalization may lead to false positives.

Rejection Sampling Fine-Tuning (RFT):
Runs a single generate-and-improve cycle; shows limited gains on GSM8K versus ReSTEM's iterative gains on harder benchmarks.

Iterative Maximum Likelihood (IML):
Optimizes via reward-weighted log-likelihood on mini-batches, risking overfitting and high computation cost.

RWR & RAFT:
Apply EM with reward scaling or ranking. RAFT is similar to IML for binary rewards and aligns with ReSTEM.

Comparison

	ReST^{EM}	ReST	STaR	RFT
Starts from fine-tuned model	✗	✓	✗	✗
Finetunes from base model in each iteration	✓	✗	✓	N/A
Uses rationalizations for unsolved questions	✗	✗	✓	✗
Temperature sampling for exploration	✓	✓	✗	✓
Experiments with Large LMs	✓	✗	✗	✓
Multiple iterations	✓	✓	✓	✗
Larger gains on bigger models	✓	N/A	N/A	✗
Evaluation on held out tasks	✓	✗	✗	✗

Problem Formulation

- Assume access to an autoregressive language model which can produce a sequence of output tokens $\mathbf{y} = (y_1, y_2, , , y_T)$ given context or source input $\mathbf{x} = (x_1, x_2, , , x_L)$.

- Assuming that the model is parametrised by θ, the conditional probability of generating a sequence $\mathbf{y}$ given $\mathbf{x}$ is:

$$p_\theta(\mathbf{y} \,|\, \mathbf{x}) = \Pi_{t=1}^{T} p_\theta(y_t \,|\, y_{<t}, \mathbf{x})$$

- Assume access to deterministic sequence level (or terminal) reward $r(\mathbf{x}, \mathbf{y})$

- Goal: Maximize $\mathscr{L}_{RL}(\theta) = \mathbb{E}_{\mathbf{x} \sim \mathscr{D}} \left[\mathbb{E}_{x \sim p_\theta(\mathbf{y}|\mathbf{x})} [r(\mathbf{x}, \mathbf{y})] \right]$

But ….

- Optimizing $\mathscr{L}_{RL}(\theta) = \mathbb{E}_{\mathbf{x}\sim\mathscr{D}}\left[\mathbb{E}_{x\sim p_\theta(\mathbf{y}|\mathbf{x})}[r(\mathbf{x}, \mathbf{y})]\right]$ is computationally expensive.
- Policy Gradient based RL methods require updating and sampling from the policy numerous times during training.

Idea: Expectation Maximization

- Expectation Maximization (Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977) , Dayan & Hinton 1997)
- Define binary optimality variable O, such that $p(O = 1 \mid \mathbf{x}, \mathbf{y}) \propto f(r(\mathbf{x}, \mathbf{y}))$
- We want to maximize the log likelihood of observing $O = 1$ (obtaining high reward)
- $$\log(p(O = 1 \mid x)) := \log \sum_{y} p_\theta(\mathbf{y} \mid \mathbf{x}) p(O = 1 \mid \mathbf{x}, \mathbf{y})$$
- However, the sum over all possible output sequences $\mathbf{y}$ is typically intractable.
- So, instead of maximising log likelihood directly, we maximize its Evidence Lower Bound.

ELBO

- The Evidence Lower Bound for the log likelihood term is given by:
- $\mathscr{L}(p_\theta, q) = \mathbb{E}_{q(y|x)}\left[\log p\left(O = 1 \mid x, y\right)\right] - \mathrm{KL}\left(q(y \mid x) \,\|\, p_\theta(y \mid x)\right)$
- The expectation maximization algorithm maximizes this objective by alternating between E-step and M-step.
- E-step:
    - $q^{t+1} = argmax_q L(p_{\theta^t}, q)$
- M-step:
    - $\theta^{t+1} = argmax_\theta L(p_\theta, q^{t+1})$

ReST^{EM}

Generate (E-step):

- **Input**: Current model $p\theta$, dataset D of input contexts (e.g., math problems).
- **Process**:
  1. For each input x_j in D, sample N outputs y_j from $p\theta(y|x_j)$
  2. Score each pair (x_j,y_j) with a binary reward $r(x_j,y_j)$:
     - 1 if correct, 0 if incorrect.
  3. Collect correct pairs into a new dataset $D_i=\{(x_j,y_j)|r(x_j,y_j)=1\}$.
- **Output**: A dataset D_i of high-quality (correct) samples.

ReST^EM

- **Improve (M-step):**
- **Input**: Base pre-trained model θ base, dataset D_i
- **Process**:
  - Fine-tune θ_{base} on D_i to maximize:
    - $J(\theta) = \mathbb{E}_{(x,y)\sim\mathscr{D}_i}\left[r(x, y)\log p_\theta(y \mid x)\right]$
  - Since r(x,y)=1 for all pairs in D_i, fine-tuning on correct outputs
- **Output:** A new model θ_i, used for the next Generate step

Experiment

Tasks:

- Math: Hendrycks MATH, GSM8K
- Coding: APPS (Intro) & HumanEval

Models: PaLM-2 Series (S, S*, L)

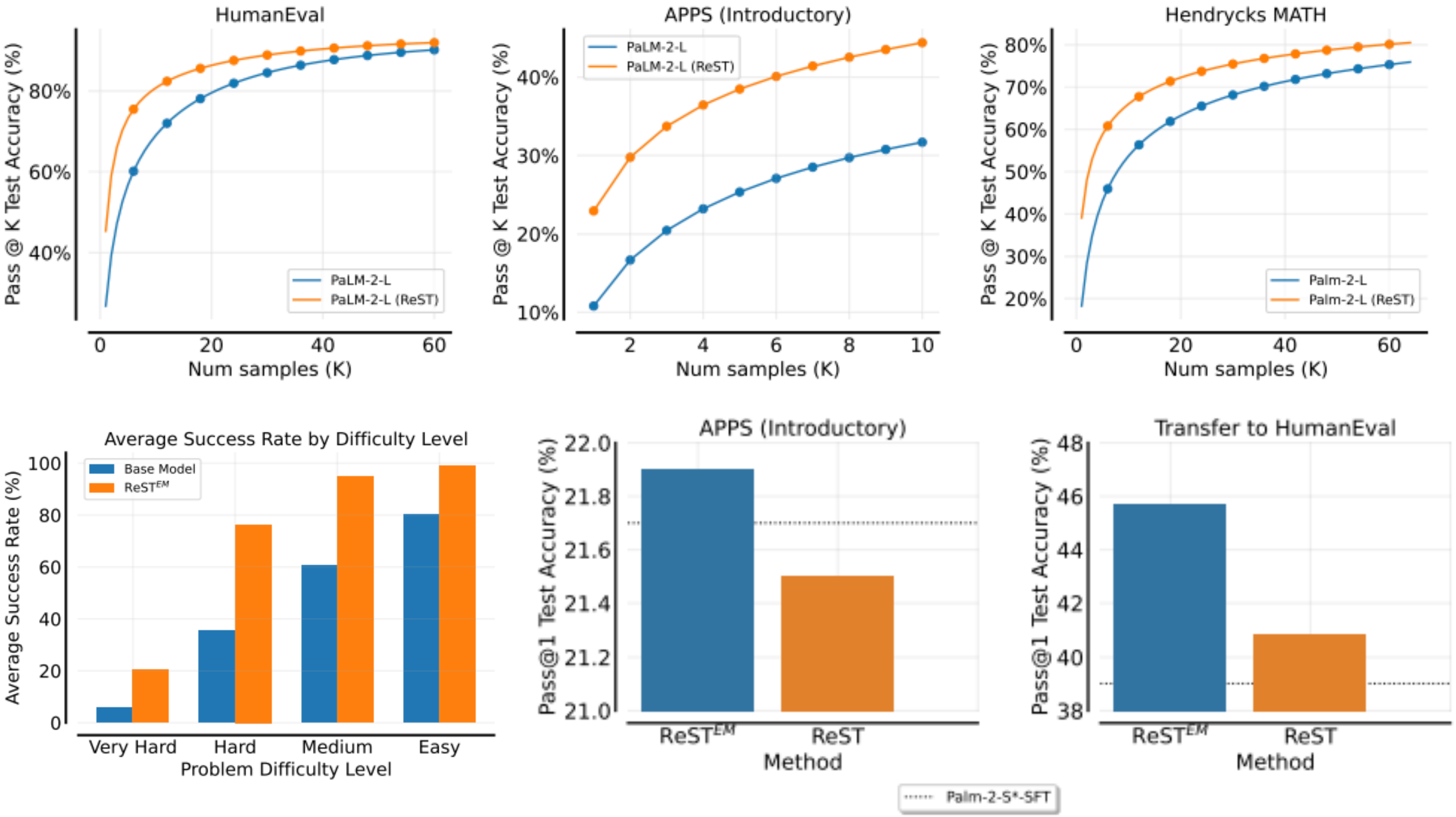
Main Comparison:

- SFT on human data vs. ReST^{EM} on model-generated data

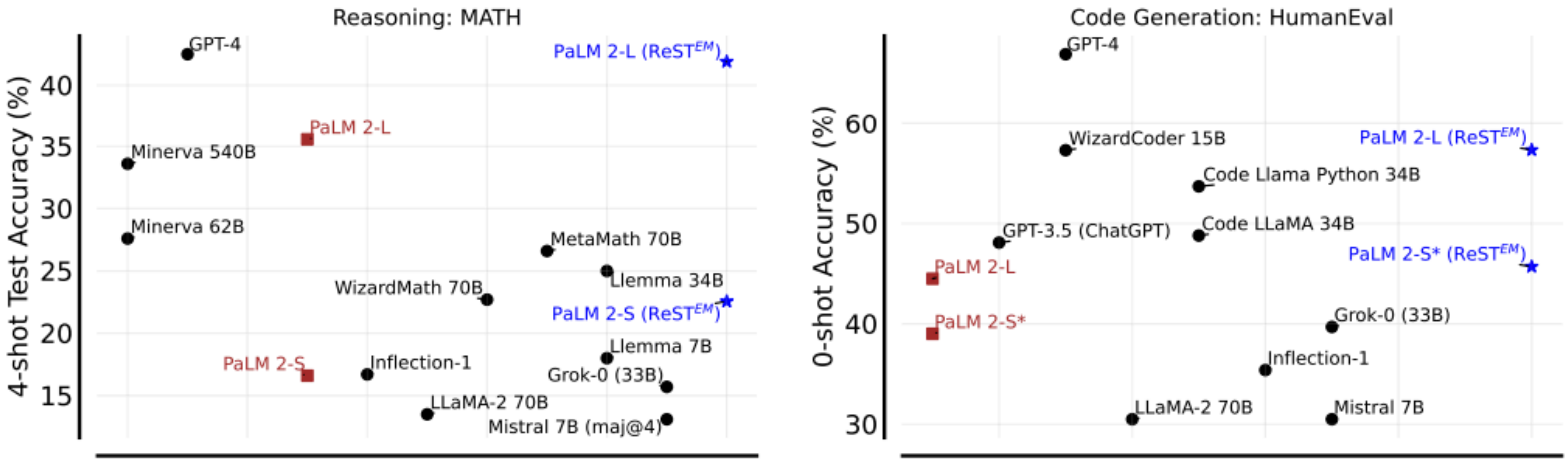
Evaluation:

- Pass@1 (direct generation)
- Pass@k / majority voting for diversity

Results


HumanEval
Pass @ K Test Accuracy (%)
40%
60%
80%
0
20
40
60
Num samples (K)
PaLM-2-L
PaLM-2-L (ReST)
APPS (Introductory)
Pass @ K Test Accuracy (%)
10%
20%
30%
40%
2
4
6
8
10
Num samples (K)
PaLM-2-L
PaLM-2-L (ReST)
Hendrycks MATH
Pass @ K Test Accuracy (%)
20%
30%
40%
50%
60%
70%
80%
0
20
40
60
Num samples (K)
Palm-2-L
Palm-2-L (ReST)
Average Success Rate by Difficulty Level
Average Success Rate (%)
0
20
40
60
80
100
Base Model
ReST^EM
Very Hard
Hard
Medium
Easy
Problem Difficulty Level
APPS (Introductory)
Pass@1 Test Accuracy (%)
21.0
21.2
21.4
21.6
21.8
22.0
ReST^EM
ReST
Method
Transfer to HumanEval
Pass@1 Test Accuracy (%)
38
40
42
44
46
48
ReST^EM
ReST
Method
Palm-2-S*-SFT

Results


Reasoning: MATH
4-shot Test Accuracy (%)
40
35
30
25
20
15
GPT-4
PaLM 2-L (ReST^{EM})
PaLM 2-L
Minerva 540B
Minerva 62B
MetaMath 70B
Llemma 34B
WizardMath 70B
PaLM 2-S (ReST^{EM})
Llemma 7B
PaLM 2-S
Inflection-1
Grok-0 (33B)
LLaMA-2 70B
Mistral 7B (maj@4)
Code Generation: HumanEval
0-shot Accuracy (%)
60
50
40
30
GPT-4
WizardCoder 15B
PaLM 2-L (ReST^{EM})
Code Llama Python 34B
GPT-3.5 (ChatGPT)
Code LLaMA 34B
PaLM 2-S* (ReST^{EM})
PaLM 2-L
PaLM 2-S*
Grok-0 (33B)
Inflection-1
LLaMA-2 70B
Mistral 7B

Key Observations

Significant Boost Over Human Fine-Tuning:

- On MATH, $ReST^{EM}$ surpasses supervised fine-tuning (SFT) with human-written solutions.
- Gains are bigger for larger models (e.g., PaLM 2-L).

Multiple Iterations:

- MATH: More iterations → steady improvement until overfitting starts.
- APPS: 1st iteration yields the largest boost; further iterations can hurt (fewer training problems).

Improved Diversity:

- Higher Pass@K (chance that at least 1 of K samples is correct).
- Better majority-voting accuracy.

Difficulty Analysis:

- MATH subset shows the biggest gains on medium-to-hard problems.
- Exploiting multiple model-generated solutions yields richer training data

Limitations

Dependence on Clear Correctness Signals

- ReST^{EM} needs a well-defined reward check.
- Tasks without an automatic way to decide correctness are hard to handle.

Overfitting on Small Data

- Iteratively fine-tuning on a limited set of problems can reduce generalization, as seen in APPS.

Possible “Reward Hacking”

- If the correctness check is incomplete or simplistic the model might learn shortcuts or produce false positive

Conclusion

Model-Generated Data Can Outperform Human Data

- Especially in math and coding tasks, correctness checks enable scalable, high-quality self-training.

ReST^{EM} Scales

- Strong gains observed on larger models; iterative refinement outperforms single-step approaches.

Overfitting is a Concern

- The number of iterations and dataset size both matter. Repeated re-training can degrade performance.

Broad Potential

- No major regressions on general benchmarks; could be generalized to a wide range of tasks with reliable performance