# STaR: Self-Taught Reasoner

Author: Zelikman et al.

created by paper2slides

Released on arXiv: 2022-03-28

https://arxiv.org/abs/2203.14465

# Executive Summary

- This paper introduces the Self-Taught Reasoner (STaR) framework that uses a few reasoning examples to iteratively train large language models (LLMs) for better rationale generation and problem-solving.
- STaR leverages model-generated rationales to bootstrap reasoning without requiring massive rationale datasets or sacrificing accuracy through few-shot prompting.
- The method's core loop involves generating rationales for problems, filtering for correctness, and fine-tuning on the correct rationales.
- STaR significantly improves performance on arithmetic, commonsense reasoning (CommonsenseQA), and grade school math (GSM8K).

# Introduction and Background

- Human decision-making often involves chain-of-thought reasoning, which has been shown to benefit LLMs as well.
- Previous methods either required constructing large rationale datasets or compromised on accuracy.
- The Self-Taught Reasoner (STaR) proposes a bootstrapping method to leverage a small set of rationale examples for iterative fine-tuning.
- This enables LLMs to improve their reasoning capabilities using their own generated rationales.

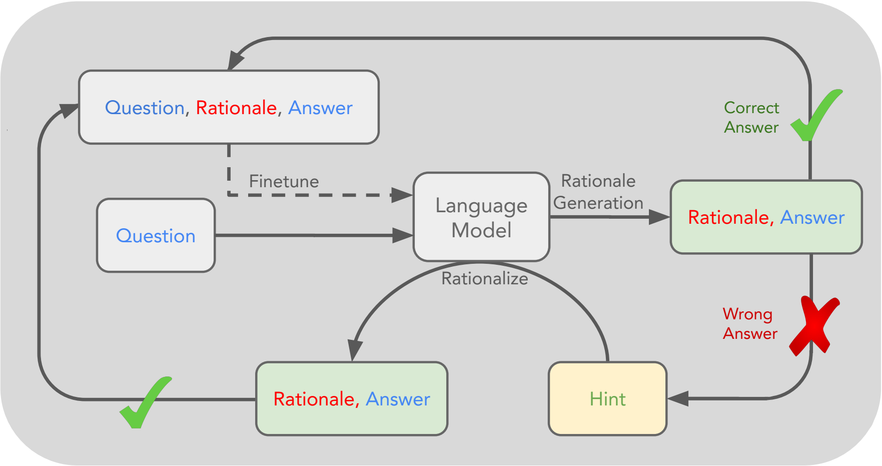


Figure: Overview of the STaR framework, with rationale generation and rationalization steps.

# Proposed Method: Self-Taught Reasoner (STaR)

- STaR initiates with a small prompt set $\mathcal{P}$ of problems with rationales to guide a large dataset $\mathcal{D}$ without rationales.
- Iteratively follows a loop:
  - Generate rationales using few-shot prompts.
  - Filter out incorrect rationales based on the answers.
  - Fine-tune the model on correct rationales.
  - Generate rationales again with the improved model.
- Rationalization: For problems with wrong answers, the model is given the correct answer to generate a rationale, which is then added to the fine-tuning set.

# Algorithm: STaR with Rationalization

$$J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot|x_i)} \mathbb{1}(\hat{y}_i = y_i), \tag{1}$$

$$\nabla J(M, X, Y) = \sum_i \mathbb{E}_{\hat{r}_i, \hat{y}_i \sim p_M(\cdot|x_i)} \left[\mathbb{1}(\hat{y}_i = y_i) \cdot \nabla \log p_M(\hat{y}_i, \hat{r}_i \mid x_i)\right] \tag{2}$$

- $J(M, X, Y)$: Expected reward function over the dataset.
- $\mathbb{1}(\hat{y}_i = y_i)$: Indicator function checking answer correctness.
- STaR approximates $J$ by decoding rationale-answer pairs and fine-tuning iteratively.

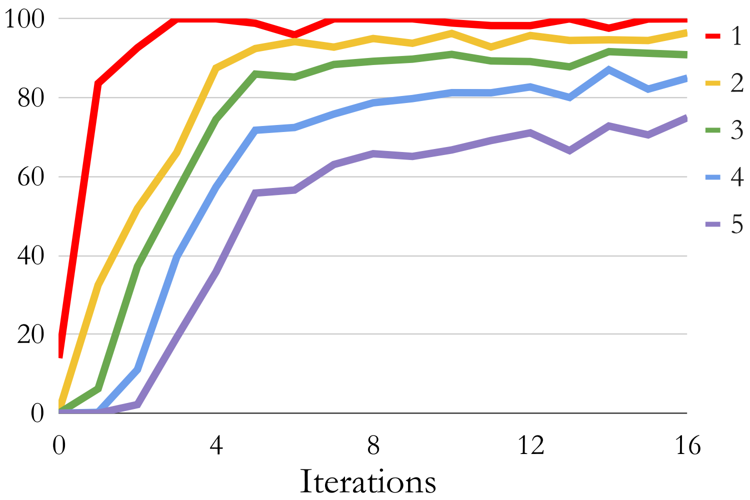


Figure: Accuracy of STaR with rationalization across different digit lengths.

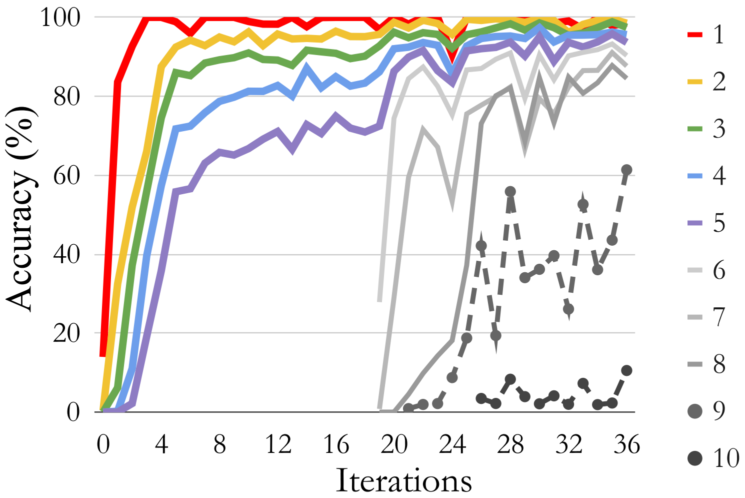


Figure: Introducing additional 9-10 digit sums in training showcases out-of-distribution performance.

# CommonsenseQA Results

- Comparison across various baselines:
  - GPT-J Finetuned: 60.0%
  - Few-shot CoT GPT-J: 36.6%
  - Few-shot CoT LaMDA 137B: 55.6%
- STaR without rationalization: 68.8%
- STaR with rationalization: 72.5%, matching larger GPT-3 model (73.0%).

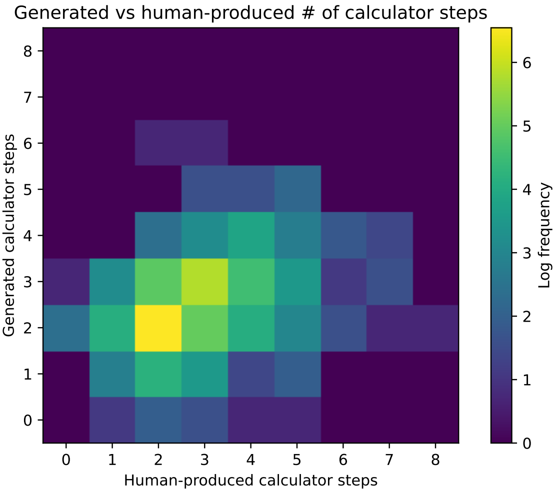


Figure: Correlation between number of steps generated by the model versus human solutions on GSM8K.

# Case Study: CQA Example

- STaR-generated rationale versus few-shot prompt rationale for CommonsenseQA problem.
- Rationales provided are generally coherent and of a similar structure to the few-shot rationales.
- STaR improves the quality of rationales over those generated in a few-shot manner.
- Rationalization examples involve explaining why the ground truth answer is correct.

# Human Evaluation of Rationales

- Conducted a qualitative study assessing the quality of rationales.
- Participants rated STaR-generated rationales 30% higher than few-shot rationales.
- Participants preferred STaR rationales over human-generated rationales.

# Results: Grade School Math (GSM8K)

- Significant improvement on GSM8K dataset:
  - Few-shot Direct GPT-J: 3.0%
  - GPT-J Direct Finetuned: 5.8%
- STaR without rationalization: 10.1%
- STaR with rationalization: 10.7%

# Conclusion and Future Directions

- STaR provides a scalable method to bootstrap rationale generation for enhanced reasoning.
- Demonstrated significant improvements across multiple datasets and tasks.
- Future work should explore:
  - Application to more diverse problem sets.
  - Investigating the balance between rationale generation and rationalization further.
  - Improving methods for filtering bad rationales, especially in settings with high chance performance.