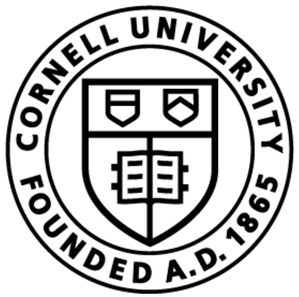
Lecture 11: Modern LLMs



Cornell Bowers CIS
Computer Science

Claire Cardie, Tanya Goyal

CS 4740 (and crosslists): Introduction to Natural Language Processing

Announcements

- HW3 Final submission due April 21, 11.59 p.m.
- HW4 will be released on April 22.

Today

- Recap: Pre-training, Task-specific Fine-tuning
- Prompting
  - Zero-shot
  - Few-shot
  - Chain-of-thought
- Post-training
  - Instruction Tuning / Supervised Fine-tuning for general tasks.
  - Reinforcement Learning for Alignment

Recap: Pre-training, Task-specific Fine-tuning

- Pre-train the model on a large corpus of text with next-token prediction objective (decoder) or masked language modeling objective (encoder)
  - GPT2/GPT3 are decoder-only language models trained on NTP.
  - BERT is an encoder-only model trained on MLM.
- Last-lecture:
  - Pre-training teaches the model general language capabilities.
  - Fine-tuning specializes it for specific tasks.

Recap: Task-specific Fine-tuning

- Let M_0 be a pre-trained **decoder** model.
- We want to train this model for question answering (given a question q, generate an answer a). How do we **fine-tune** this model?

See Written Notes

- We want to train this model for sentiment classification (given a paragraph x, predict a sentiment $y \in \{\text{positive, negative, neutral}\}$ a). How do we **fine-tune** this model?

See Written Notes

Recap: Task-specific Fine-tuning

- Task-specific fine-tuning only specializes the model for **one** downstream task.
- Post-GPT3, there was a shift towards building general-purpose models that can solve *any* task at test time.
- How do we build such models?

Prompting - Zero- and Few-shot

- Approach 1: Use the general-purpose pre-trained model with zero-/few-shot learning to adapt to different tasks at inference time.
- First shown to work successfully with GPT3.
- Why this works: Pre-trained models are exposed to a lots of downstream tasks in pre-training. We just need to extract that behavior.
- Two main paradigms
  - Zero-shot prompting (the task is described in words)
  - Few-shot prompting (exemplars are given in-context, i.e. in the prompt)

Zero-shot prompting

- Let's focus on **classification tasks**
  - Input **x**, want to predict label **y** from a label set.
  - Example: **x** = "The restaurant ambience was great but the food was a little bland"
  - Wrap this in a *verbalizer* ***v***

Review: The restaurant ambience was great but the food was a little bland
Choose from the following sentiment labels for this review: Positive, Negative, Neutral

Neutral

Slide from Greg Durrett.

Multiple ways to do classification with Zero-shot prompting

Review: The restaurant ambience was great but the food was a little bland
Choose from the following sentiment labels for this review: Positive, Negative, Neutral

Neutral

- Approach 1: Generate from the model and match answer against out label set.
- **What can go wrong?**

Model does not follow the instruction, generates something else.

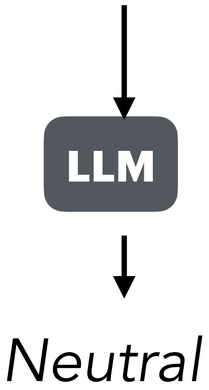
"The person seemed to like the restaurant which would indicate he is positive about the restaurant but maybe the food is the more important factor, so it should be negative"

Slide from Greg Durrett.

Multiple ways to do classification with Zero-shot prompting

Review: The restaurant ambience was great but the food was a little bland

Choose from the following sentiment labels for this review: Positive, Negative, Neutral



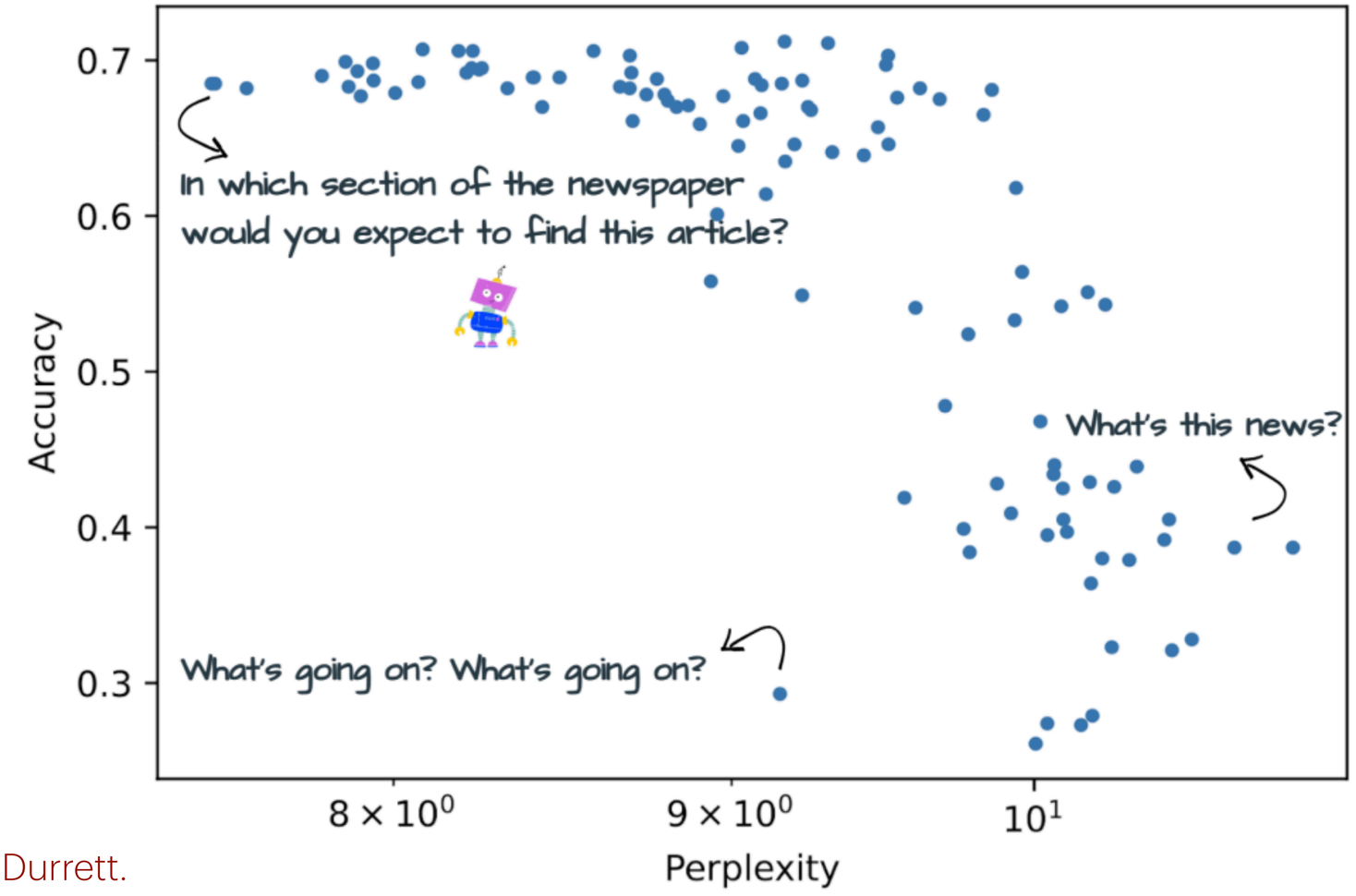
Neutral

- Approach 2: Create templates: <input> The review is [positive/negative/neutral]
- Compare P(positive | context), P(negative | context), P(neutral | context)
- You only compare between valid examples.

Slide from Greg Durrett.

Issues with Zero-shot Prompting

- Different verbalizations may lead to different outputs.



Slide from Greg Durrett.

Gonen et al., 2022

Issues with Zero-shot Prompting

- OPT-175B: average of best 50% of the prompts much higher than the overall average.
- How do we standardize evaluation if different prompts work for different models?

Task	Avg Acc	Acc 50%
Antonyms	–	–
GLUE Cola	47.7	57.1
Newspop	66.4	72.9
AG News	57.5	68.7
IMDB	86.2	91.0
DBpedia	46.7	55.2
Emotion	16.4	23.0
Tweet Offensive	51.3	55.8

Slide from Greg Durrett.

Gonen et al., 2022

Issues with Zero-shot Prompting

- OPT-175B: average of best 50% of the prompts much higher than the overall average.
- How do we standardize evaluation if different prompts work for different models?
- Prompt-optimization: There's some work on finding prompts that would lead to the best performance (can be done automatically or manually).

Task	Avg Acc	Acc 50%
Antonyms	–	–
GLUE Cola	47.7	57.1
Newspop	66.4	72.9
AG News	57.5	68.7
IMDB	86.2	91.0
DBpedia	46.7	55.2
Emotion	16.4	23.0
Tweet Offensive	51.3	55.8

Slide from Greg Durrett.

Gonen et al., 2022

Few-shot Prompting

- Let's focus on **classification tasks**
  - Input **x**, want to predict label **y** from a label set.

- Include a small "training dataset" in the prompt, ie (x,y) pairs that demonstrate the task.

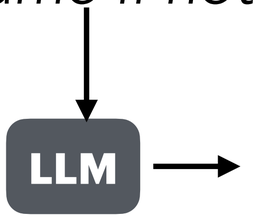
Few-shot Prompting

Review: The restaurant ambience was great but the food was a little bland
Sentiment: Neutral

Review: The movie was great! Loved the action scenes.
Sentiment: Positive

Review: Went to see that movie, total waste of time.
Sentiment: Negative
….

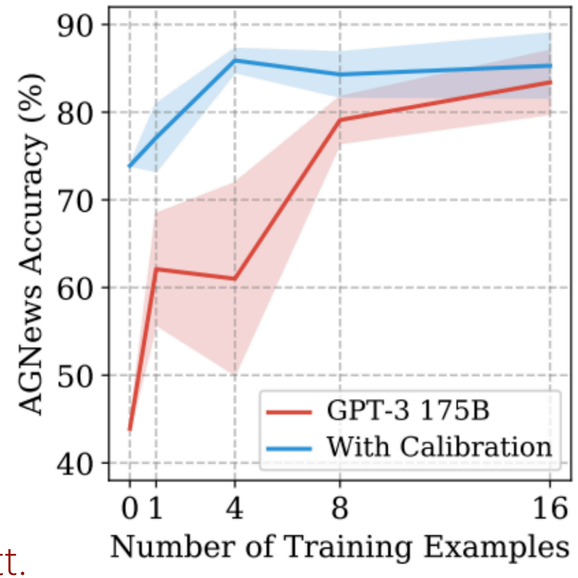
Review: The new renovations have been great, and the food quality has remained same if not better



Neutral

Few-shot Prompting: Potential Issues?

- Context demonstrates the label space through examples. What can go wrong here?
  - Skewed label distribution: If my training data has more +ves than -ves, will the model "learn" a prior that more reviews are positive?
  - Which training examples are chosen matter.



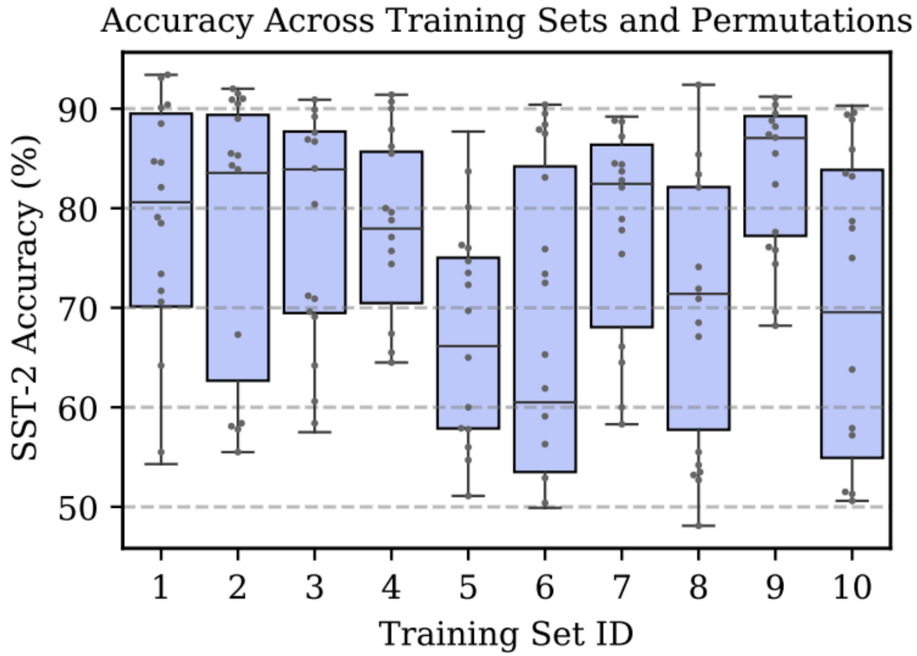
- Variance observed with different random subsets of the training examples chosen as context.
- Note: Results with GPT-3, instruct models reduce this behavior.

Slide from Greg Durrett.

Zhao et al., 2021

Few-shot Prompting: Potential Issues?

- Context demonstrates the label space through examples. What can go wrong here?
  - The order of the data can matter.



- Variance observed when varying the order of the same training set (each x axis point corresponds to one training set)

Slide from Greg Durrett.

Takeaways

- Zero- and few-shot are very performant ways of eliciting task-specific behaviors from powerful LLMs.
- Prompt-optimization matters!!
  - The results can be different based on both the verbalization and the exemplars. Tune these well!!

Chain-of-thought Prompting

- Motivation?
  - ESNLI
  - Additionally ingesting explanation/rationales should improve performance.
  - Where do we get these? Can we ask GPT3 to generate?

Premise: An adult dressed in black holds a stick.
Hypothesis: An adult is walking away, empty-handed.
Label: contradiction
Explanation: Holds a stick implies using hands so it is not empty-handed.

Chain-of-thought Prompting

- More concrete examples where rationales / intermediate reasoning steps help.

Math Word Problems (free response)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

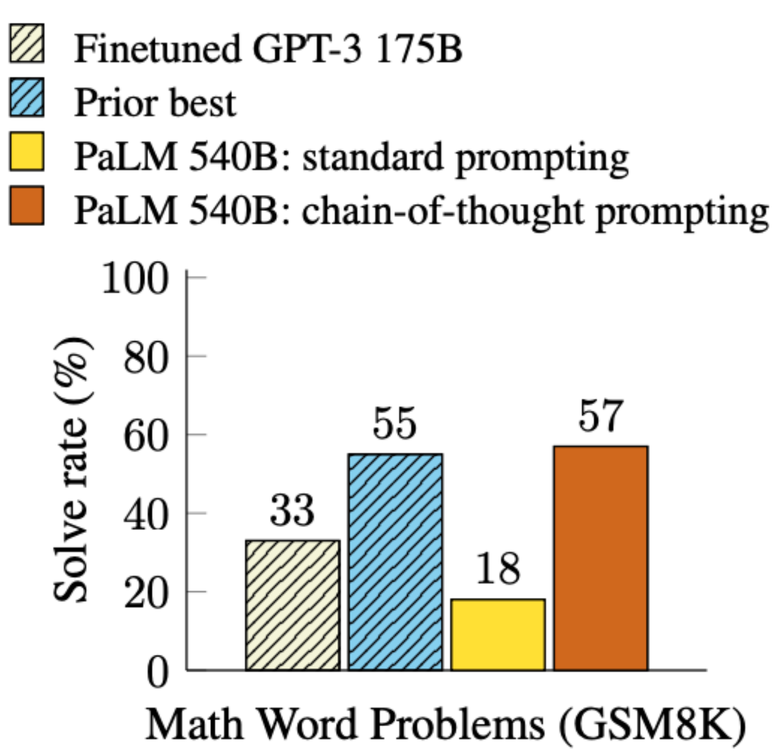
StrategyQA

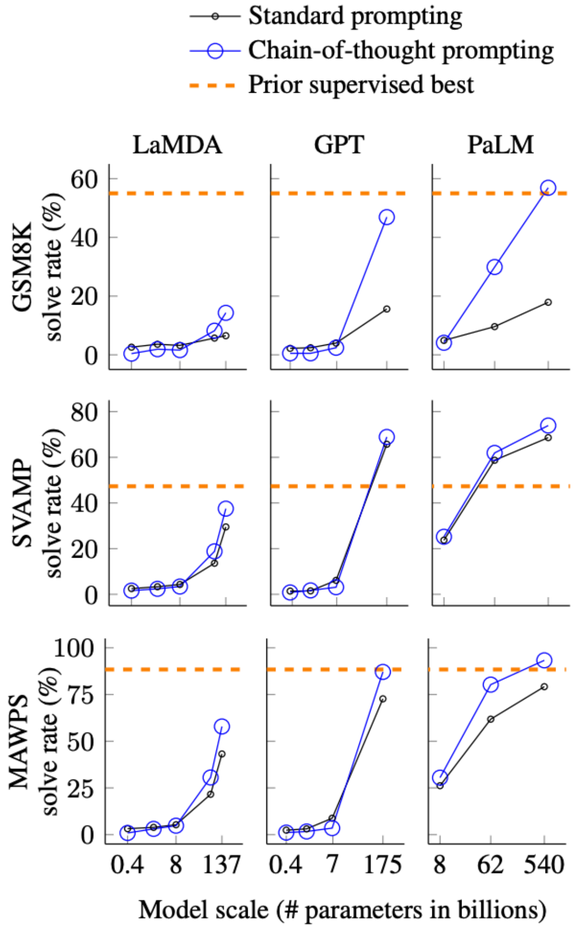
Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm^3, which is less than water. Thus, a pear would float. So the answer is no.

Chain-of-thought Prompting

- How do we prompt models to generate this chain-of-thought?
  - **Few-shot** examples with explanations (Wei et al., 2022)



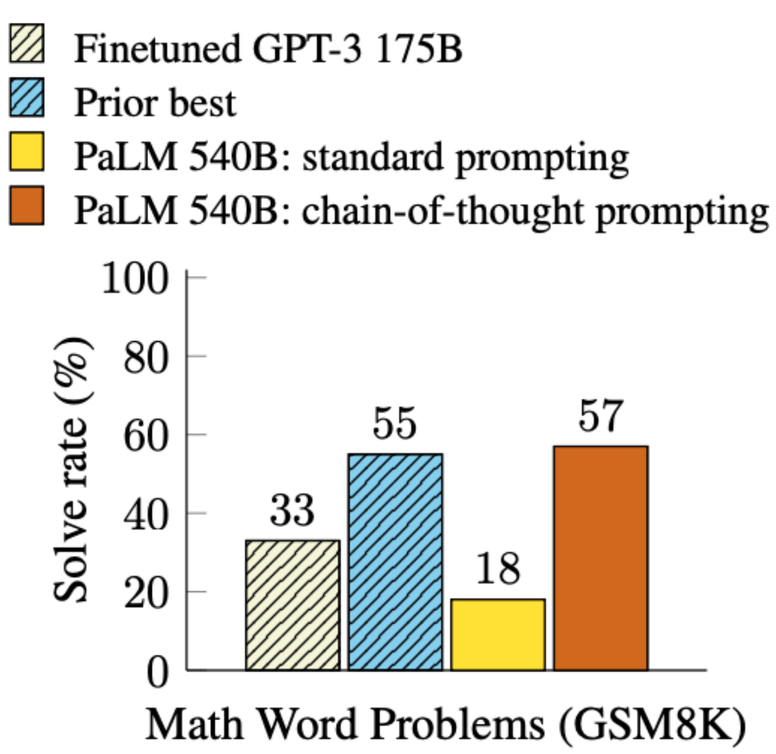


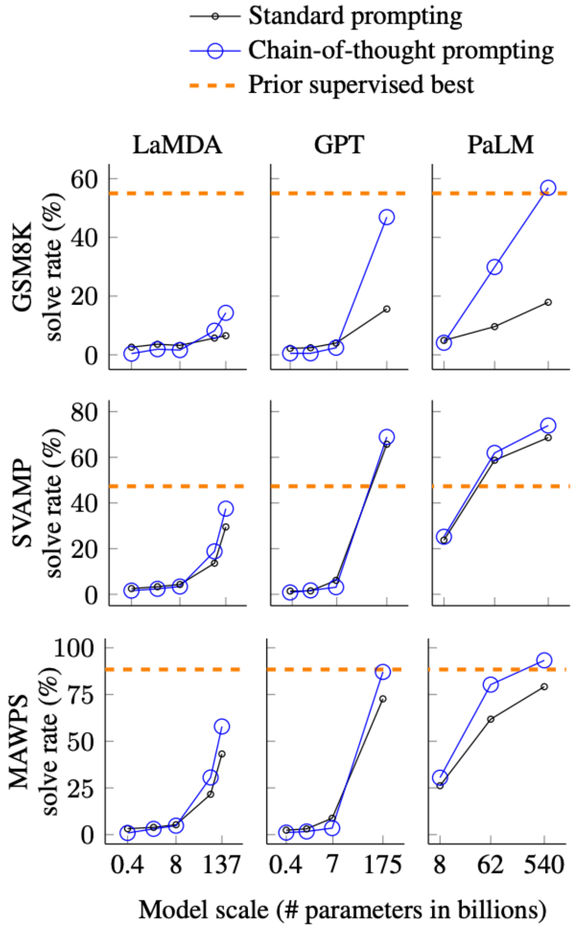
- CoT capability "emerges" with scale.

- Improvements on reasoning tasks

Chain-of-thought Prompting

- How do we prompt models to generate this chain-of-thought?
  - **Few-shot** examples with explanations (Wei et al., 2022)





- CoT capability "emerges" with scale.

- Improvements on reasoning tasks

Chain-of-thought Prompting

- Do we need **Few-shot** prompting to elicit chain of thought?
  - "Large Language Models are Zero-Shot Reasoners" Kojima et al, 2023

(b) Few-shot-CoT

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

*(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. **The answer is 4.*** ✓

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

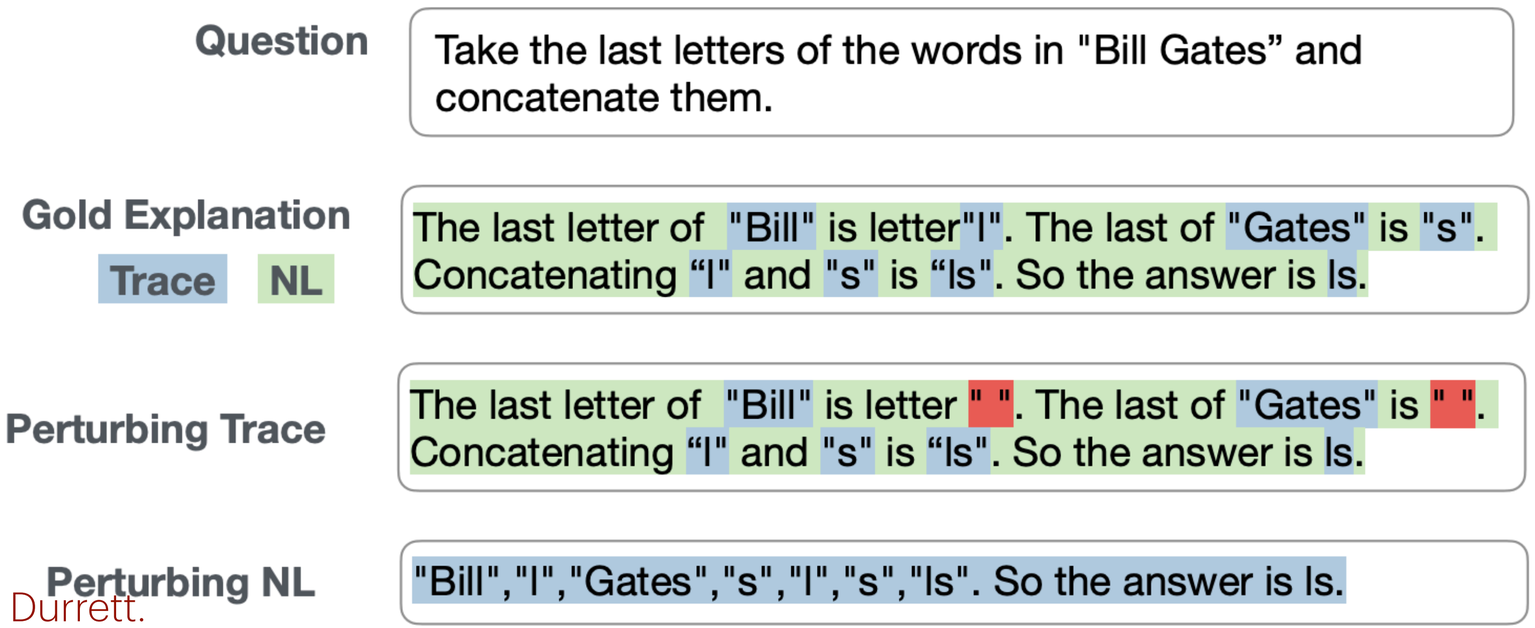
(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

No.	Category	Template	Accuracy
1	instructive	Let's think step by step.	**78.7**
2		First, (*1)	77.3
3		Let's think about this logically.	74.5
4		Let's solve this problem by splitting it into steps. (*2)	72.2
5		Let's be realistic and think step by step.	70.8
6		Let's think like a detective step by step.	70.3
7		Let's think	57.5
8		Before we dive into the answer,	55.7
9		The answer is after the proof.	45.7
10	misleading	Don't think. Just feel.	18.8
11		Let's think step by step but reach an incorrect answer.	18.7
12		Let's count the number of "a" in the question.	16.7
13		By using the fact that the earth is round,	9.3
14	irrelevant	By the way, I found a good restaurant nearby.	17.5
15		Abrakadabra!	15.5
16		It's a beautiful day.	13.1
-		(Zero-shot)	17.7

Chain-of-thought Prompting

- Do models “use” the reasoning chains as expected? i.e. does the correctness of the reasoning chain matter?

- Perturb the explanations in two ways:
  - Perturb Computational Trace
  - Perturb Natural Language

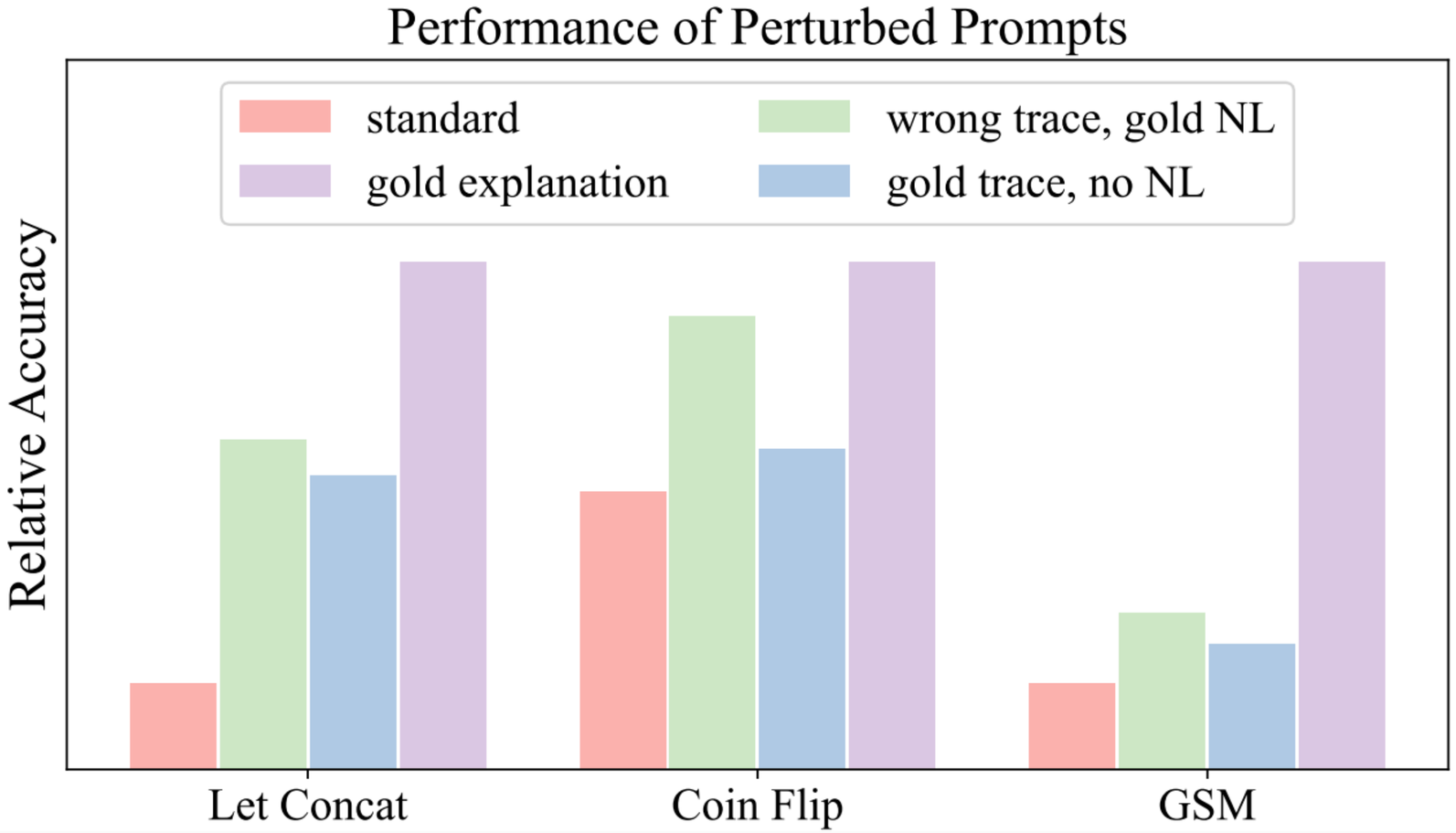


Ye and Durrett (2022)

Slide from Greg Durrett.

Chain-of-thought Prompting

- Perturbed / Incorrect explanations do lead to worse performance.
- Incorrect explanations still improve over no explanations.



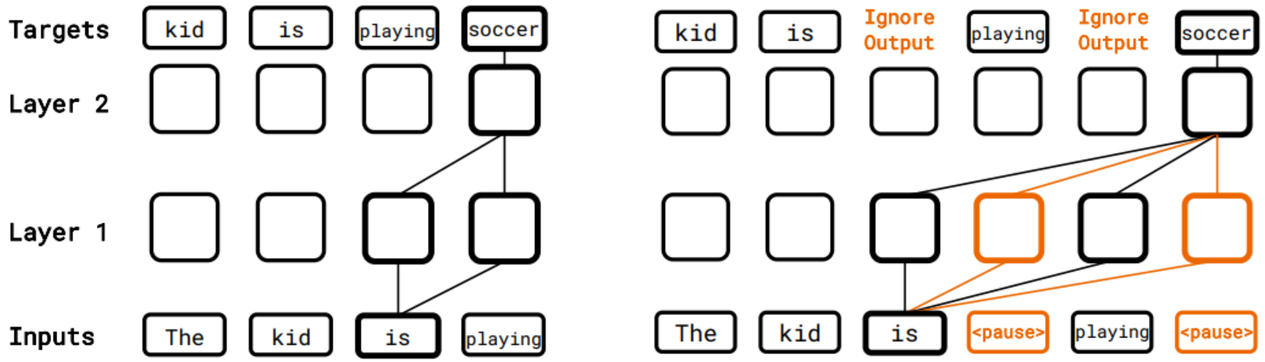
Slide from Greg Durrett.

Ye and Durrett (2022)

Other perspectives on Prompting

- Main benefit from chain-of-thought is extra computation.

- Experiments that failed:
  - Replace “explanation tokens” with “..” in the input during inference. Did not work.



(a) Standard pretraining (b) Pause-pretraining

  - Idea: Pre-train/fine-tune with pause tokens.

Goyal et al (2024)

Pause Tokens

- When pre-trained + fine-tuned with pause tokens, improves

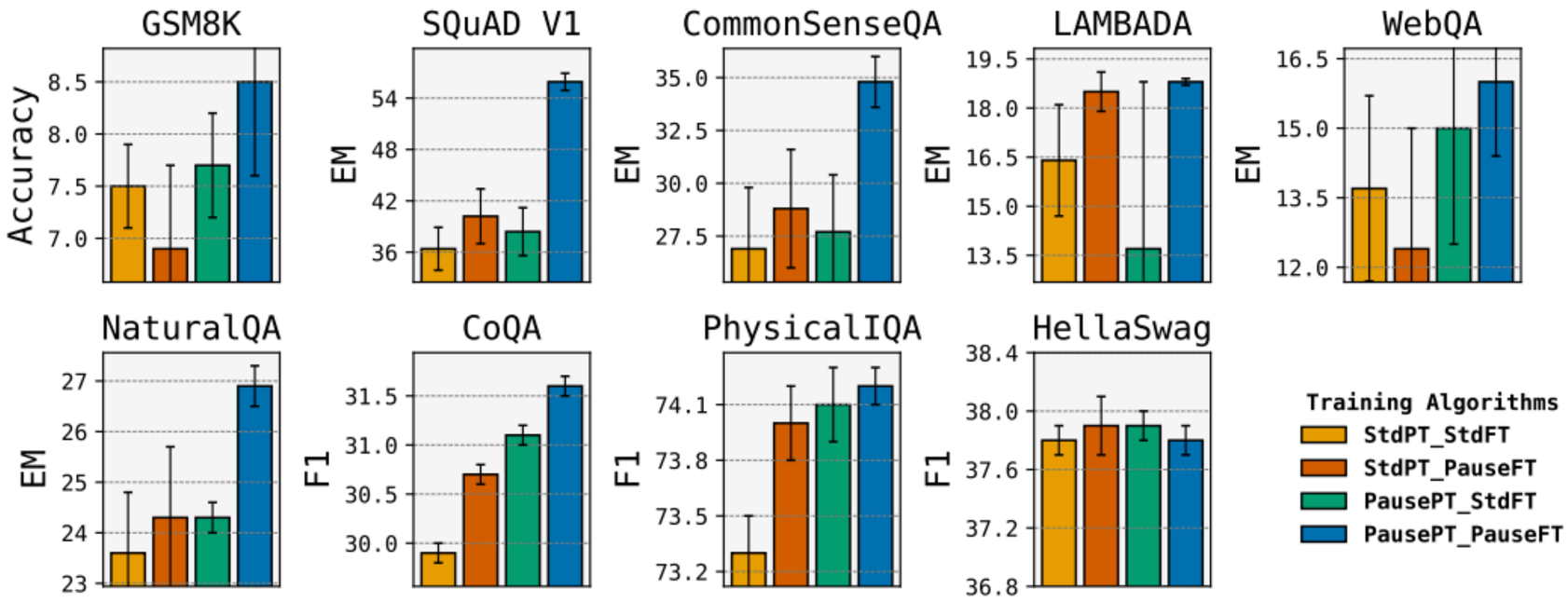
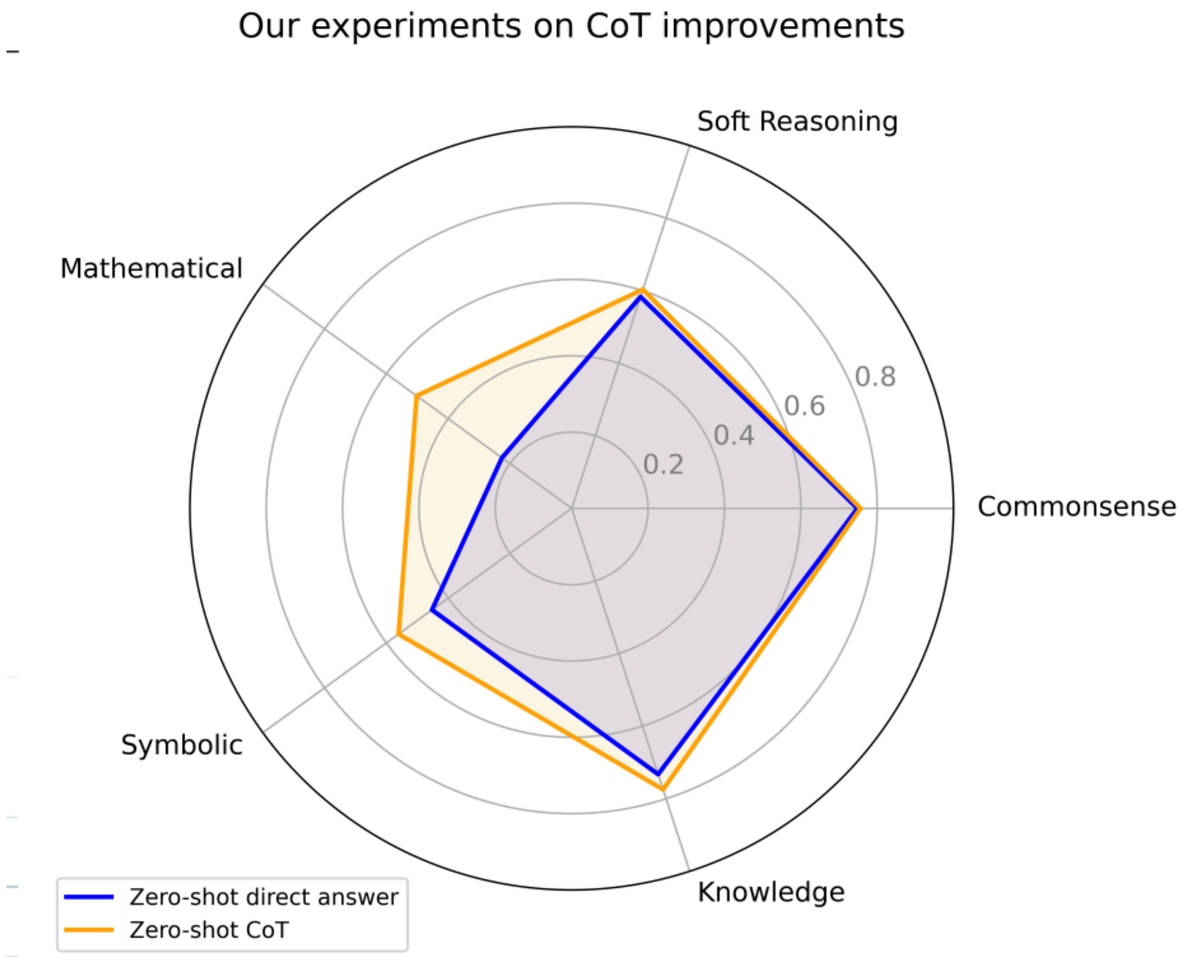


Figure 3: **Downstream performance for a 1B model.** Injecting delays in both stages of training (PausePT_PauseFT) outperforms the standard end-end training StdPT_StdFT on our wide variety of tasks (except HellaSwag). In contrast, introducing delays only in the finetuning stage provides only lukewarm gains, and even hurts in GSM8k.

Goyal et al (2024)

To CoT or not to CoT?

- Most of the improvement from CoT was for reasoning tasks,



Sprague et al, 2025

Takeaways

- Chain-of-thought works well for reasoning-adjacent tasks.
- Tons of more complicated prompting techniques studied.

- Chain-of-thought prompting was seen to emerge with scale.
- Now, models are **trained** to generate chain-of-thought.

Post-training

- Pre-GPT3:
  - Get pre-trained language models. (General purpose, trained on LM objective)
  - Fine-tune them on task-specific objectives/data.

- GPT-3
  - Get pre-trained language models. (General purpose, trained on LM objective)
  - Use zero-/few-shot learning to adapt to different tasks at inference time.

- Post-training
  - Get pre-trained language models. (General purpose, trained on LM objective)
  - **Post-tune (not necessarily on task-specific data). We still want general-purpose models after this step.**

Post-training

- **Instruction tuning:** supervised data **(x, y*)**
  - Input **x**: Who won the 2024 Cricket World Cup?
  - Output **y**: India

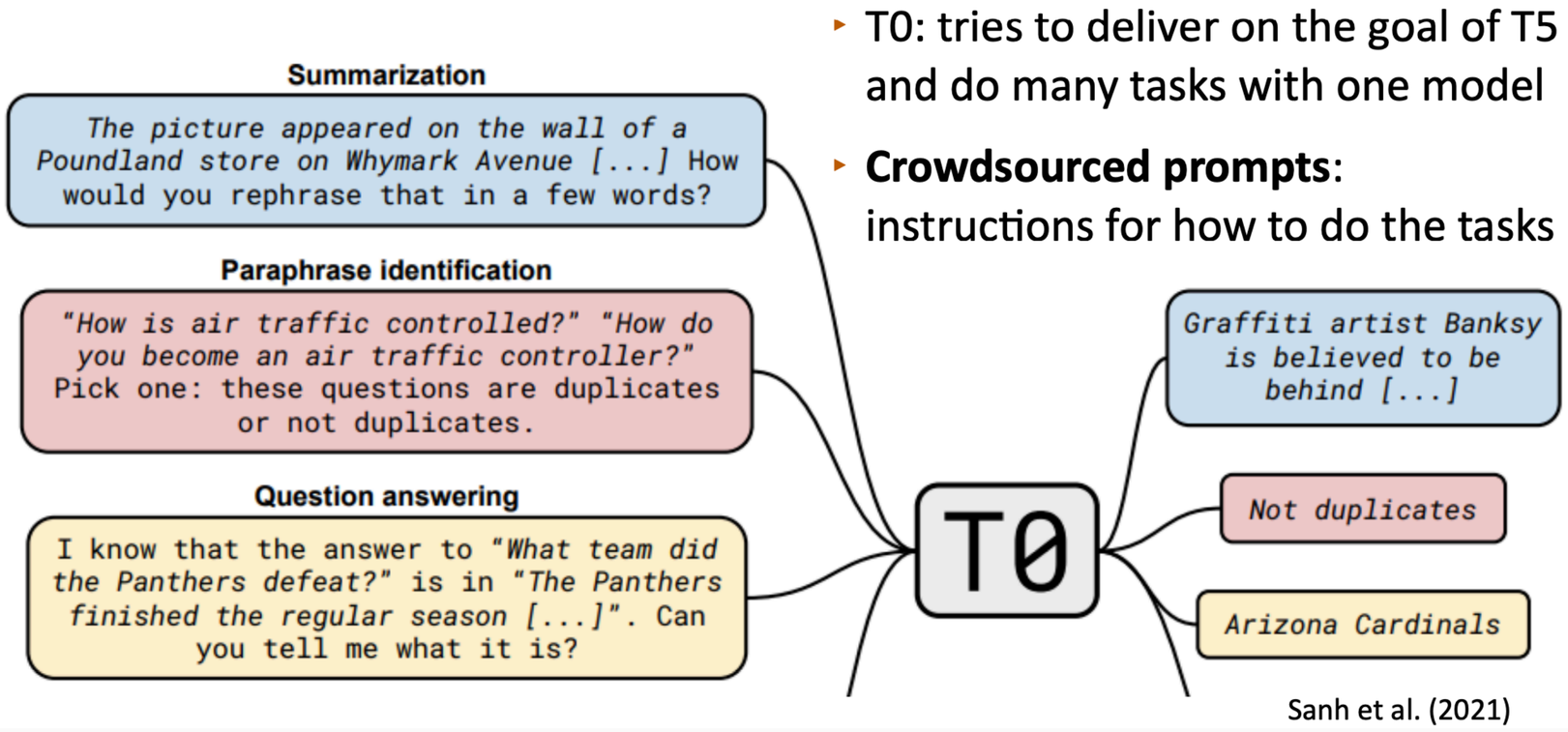
Instruction-tuning and fine-tuning used interchangeably.

- **Preference data: (x, y+, y-)**
  - Input **x:** who won the 2024 Cricket World Cup?
  - Output preferred **y+**: India
  - Output dis-preferred **y-:** Australia

Typically, a human decides which output is preferred, hence alignment with human preferences. We will discuss this later.

Instruction Tuning

- What data should we instruct-tune/fine-tune on?



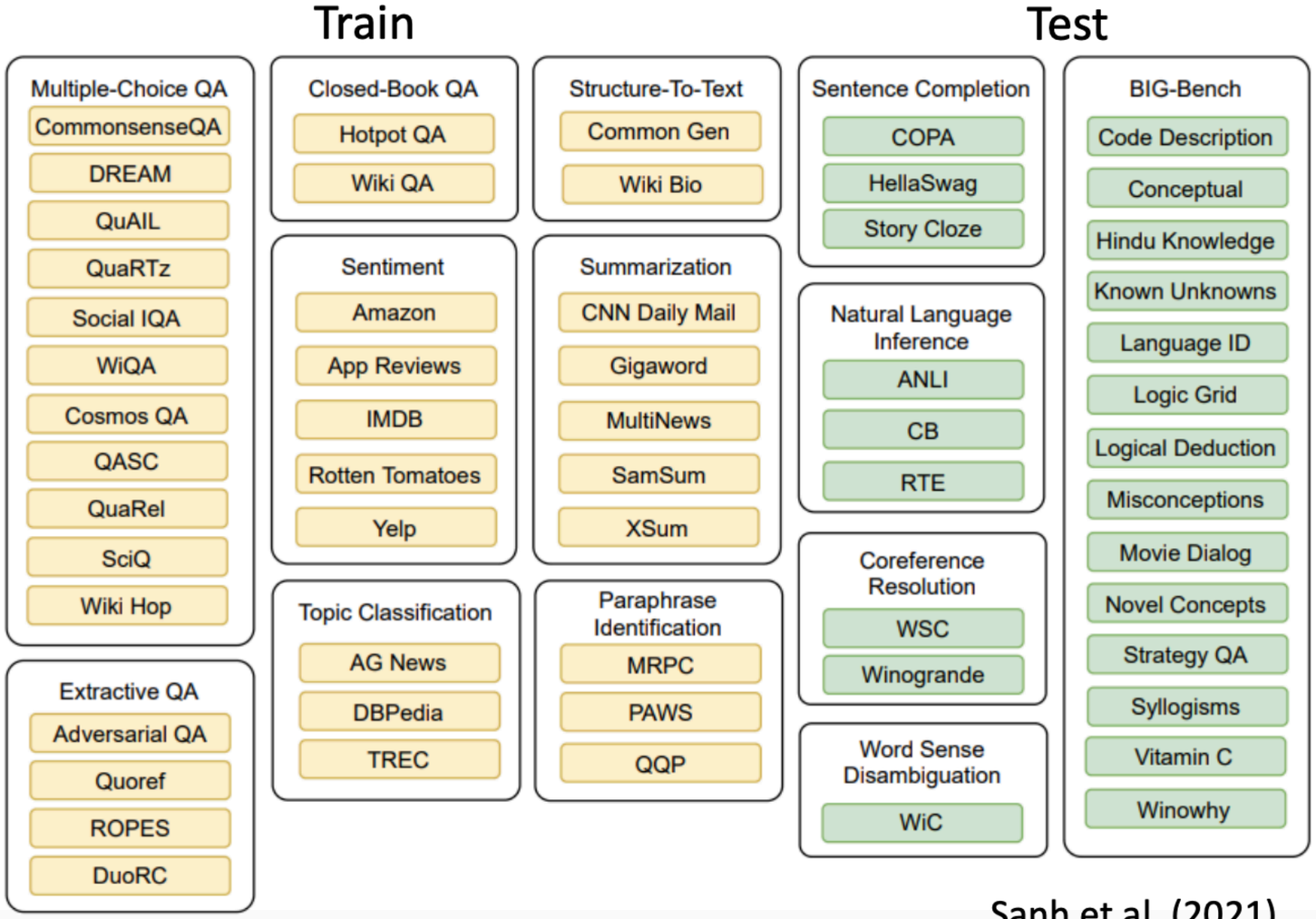
- T0: tries to deliver on the goal of T5 and do many tasks with one model
- **Crowdsourced prompts**: instructions for how to do the tasks

Sanh et al. (2021)

Combine popular NLP tasks. Include a prompt for task identification.

Instruction Tuning

- What data should we instruct-tune/fine-tune on?



Sanh et al. (2021)

Evaluate on a held-out set of "tasks" not seen during pre-training.
Helps evaluate task generalization.

Instruction Tuning

- What data should we instruct-tune/fine-tune on?
  - PaLM (google's 540B LM) also instruction-tunes on a wide range on NLP tasks.

				MMLU		BBH		TyDiQA	MGSM
Model	Finetuning Mixtures	Tasks	Norm. avg.	Direct	CoT	Direct	CoT	Direct	CoT
540B	None (no finetuning)	0	49.1	71.3	62.9	49.1	63.7	52.9	45.9
	CoT	9	52.6 (+3.5)	68.8	64.8	50.5	61.1	61.2	59.4
	CoT, Muffin	89	57.0 (+7.9)	71.8	66.7	56.7	64.0	65.3	**63.0**
	CoT, Muffin, T0-SF	282	57.5 (+8.4)	72.9	**68.2**	57.3	64.0	65.8	61.6
	CoT, Muffin, T0-SF, NIV2	1,836	**58.5** (+9.4)	**73.2**	68.1	**58.8**	**65.6**	**67.4**	61.3

Instruction Tuning

- Limitations:
  - Traditional NLP tasks (paraphrase detection, NLI, summarization, sentiment classification etc.) very limited. These do not reflect the kind of queries users might have for frontier LLMs. E.g.
    - Traditional summarization datasets mostly news-centric.
    - Sentiment-classification/NLI are useful “skills” but aren’t end tasks users are interested in.

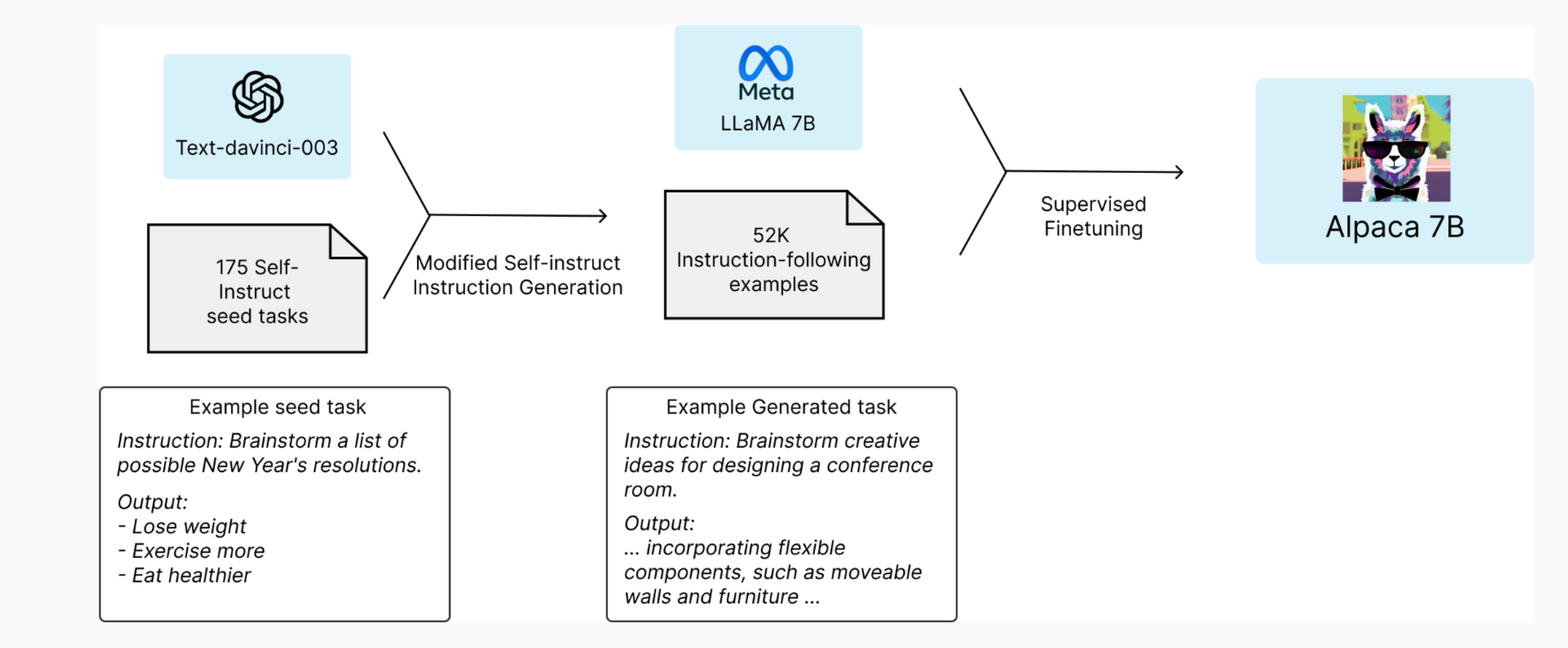
- How do we get **(x, y*)** pairs where x covers some broad diversity of queries we actually care about, and **y*** is a reasonably high quality response for **x.**
  - Distill from a larger model (like GPT-4)!

Generating Instruction Tuning data

- Lots of techniques explored:
  - Use a strong model (like GPT4) to generate both **x** (inputs) and **y** (outputs).
    - E.g. AlpacaEval
  - Use some existing collection of user queries (**WildChat)** to get user generated queries and ChatGPT responses to those queries. Train on this dataset!

Instruction Tuning (Alpaca/Self-Instruct)

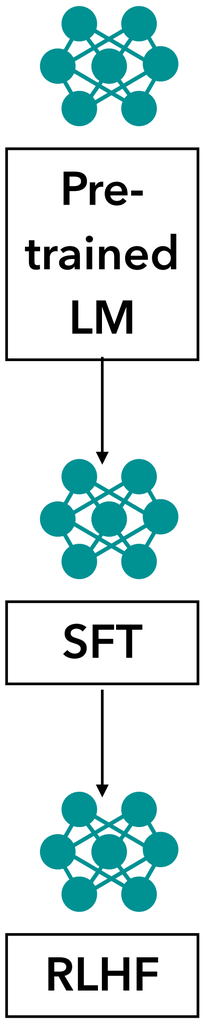
- Alpaca (Fine-tuned LLaMA1-7B model on 52k instruction-tuned examples)
- Training data, i.e **(x, y*)** pairs generated using a modified self-instruct method.



Limitations of Instruction-tuning?

- Upper bounded by responses in the training data (presumably written by humans/ stronger model).
  - These generate suboptimal responses.
- Open-ended tasks can have multiple acceptable answers. Training objective maximizes likelihood of one.
- Mismatch with actual objective “generate responses that humans prefer” and the LM objective.
- Can we explicitly train the model to generate responses that humans prefer?

Reinforcement Learning with human preferences



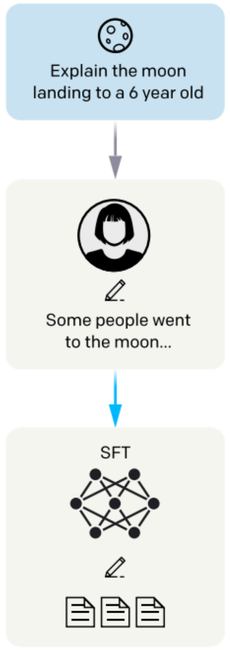
Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

This data is used to fine-tune GPT-3 with supervised learning.



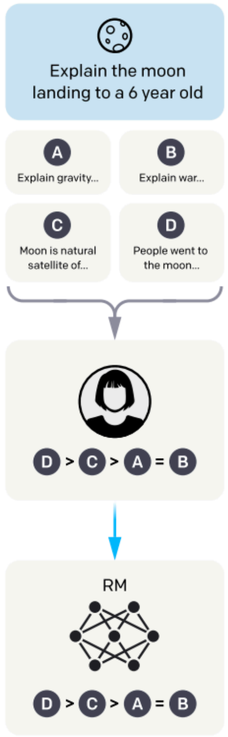
Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.



Step 3

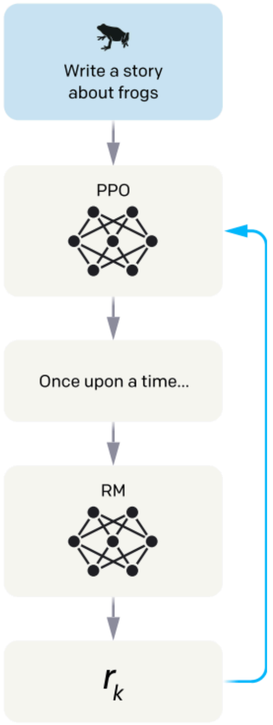
Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

The policy generates an output.

The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



Optimizing for human preferences

- Given an input x, and a sample y, suppose we have a way to obtain human reward $R(x, y) \in \mathbb{R}$.

Why don't adults roll of the bed?

$$x$$

Adults typically do not roll off of the bed because they have developed the muscle memory ...

$$y_1$$

$$R(x, y_1) = 0.8$$

Most adults find it uncomfortable or even painful to move around in their sleep, so rolling off the bed would be difficult without waking up first.

$$y_2$$

$$R(x, y_1) = 0.3$$

- We train our model to get params θ that maximize the expected reward of generated samples: $\mathbb{E}_{\hat{y} \sim p_\theta(y|x)}[R(x, \hat{y})]$

Optimizing for human preferences

- Problem: Getting rewards from humans in the loop is expensive.
  - Solution: instead of asking humans, model their preferences as a separate (NLP) problem [Knox and Stone, 2009]

Why don't adults roll of the bed?

$$x$$

Adults typically do not roll off of the bed because they have developed the muscle memory ...

$$y_1$$

$$R(x, y_1) = 0.8$$

Most adults find it uncomfortable or even painful to move around in their sleep, so rolling off the bed would be difficult without waking up first.

$$y_2$$

$$R(x, y_1) = 0.3$$

- Train a RM(x, y) that predicts these scalar rewards.

Optimizing for human preferences

- Problem: Human rewards can be extremely miscalibrated.
  - Solution: Ask for preferences instead of scores. [Phelps et al., 2015; Clark et al., 2018]

Why don't adults roll of the bed?

x

Adults typically do not roll off of the bed because they have developed the muscle memory ...

y_1

$>$

Most adults find it uncomfortable or even painful to move around in their sleep, so rolling off the bed would be difficult without waking up first.

y_2

Bradley Terry [1952] Model

- Given preference data (x, y_+, y_-), learn the $RM_\theta(x, y)$.
  - BT stipulates that human preferences can be written as:
    - $p^*(y_+ > y_-) = \dfrac{\exp(r^*(x, y_+))}{\exp(r^*(x, y_+)) + \exp(r^*(x, y_-))}$
  - From the above dataset, we can estimate parameters via maximum likelihood.
    - $\mathscr{L}_R(r_\phi, \mathscr{D}) = -\mathbb{E}_{(x,y_+,y_-)\sim\mathscr{D}}\left[\log \sigma\left(r_\phi(x, y_+) - r_\phi(x, y_-)\right)\right]$

How do we RLHF?

- Given:
  - SFT LM model $p_{SFT}(y \mid x)$
  - Reward model $RM_{\phi}(x, y)$

- We want to maximize $\mathbb{E}_{\hat{y} \sim p_{\theta}(y|x)}[R(x, \hat{y})]$:
  - Initialize $p_{\theta}(y \mid x)$ with $p_{SFT}(y \mid x)$
  - Use Reinforcement Learning to learn params θ that maximize the above objective.
  - Learnt rewards can be imperfect.
  - Solution: Add a penalty for drifting too far from the initialization.
    - $\mathbb{E}_{\hat{y} \sim p_{\theta}(y|x)}[R_{\phi}(x, \hat{y}) - \beta \log \frac{p_{\theta}(\hat{y} \mid x)}{p_{SFT}(\hat{y} \mid x)}]$

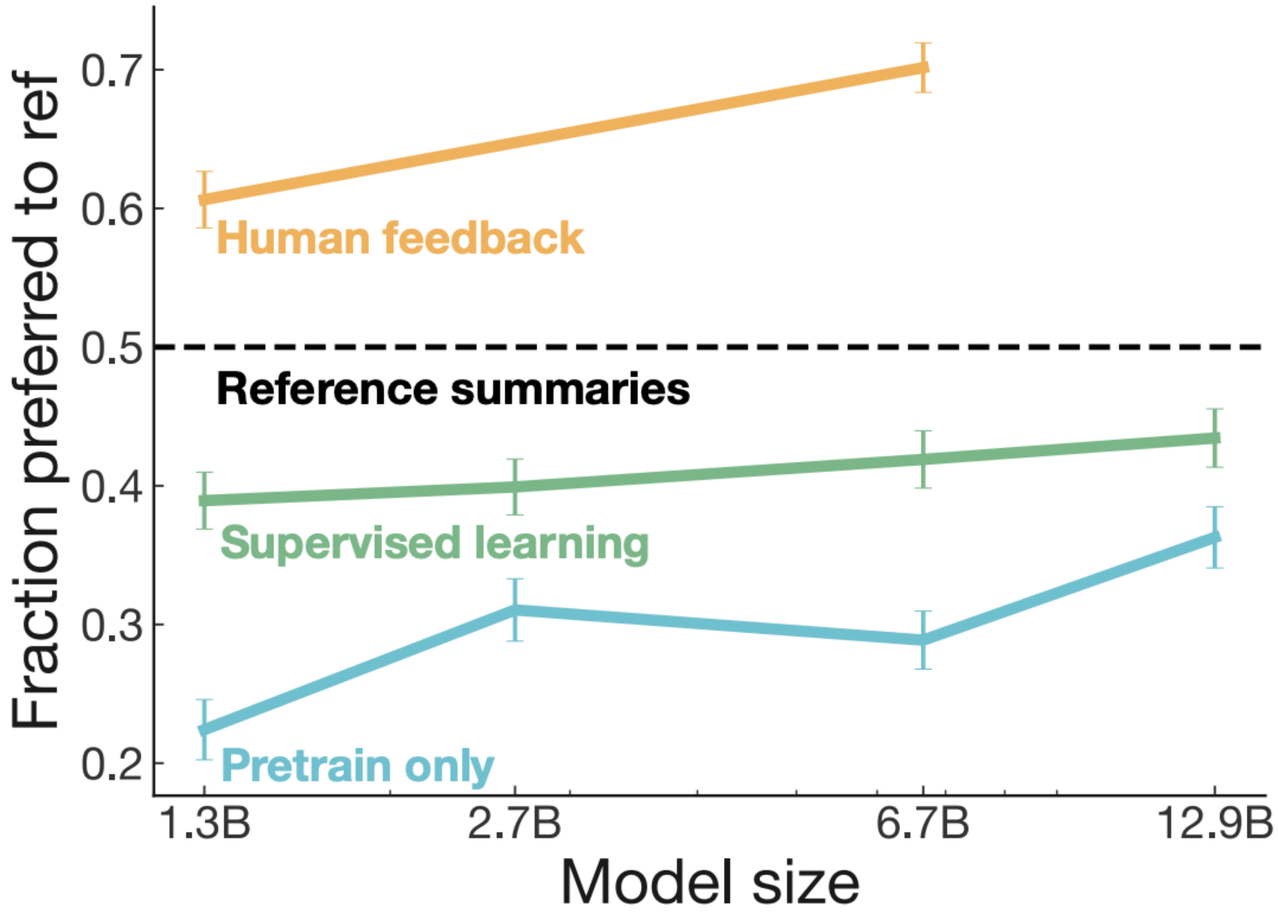
In expectation, this is the KL divergence between $p_{\theta}(y \mid x)$ and $p_{SFT}(y \mid x)$

How do we RLHF?

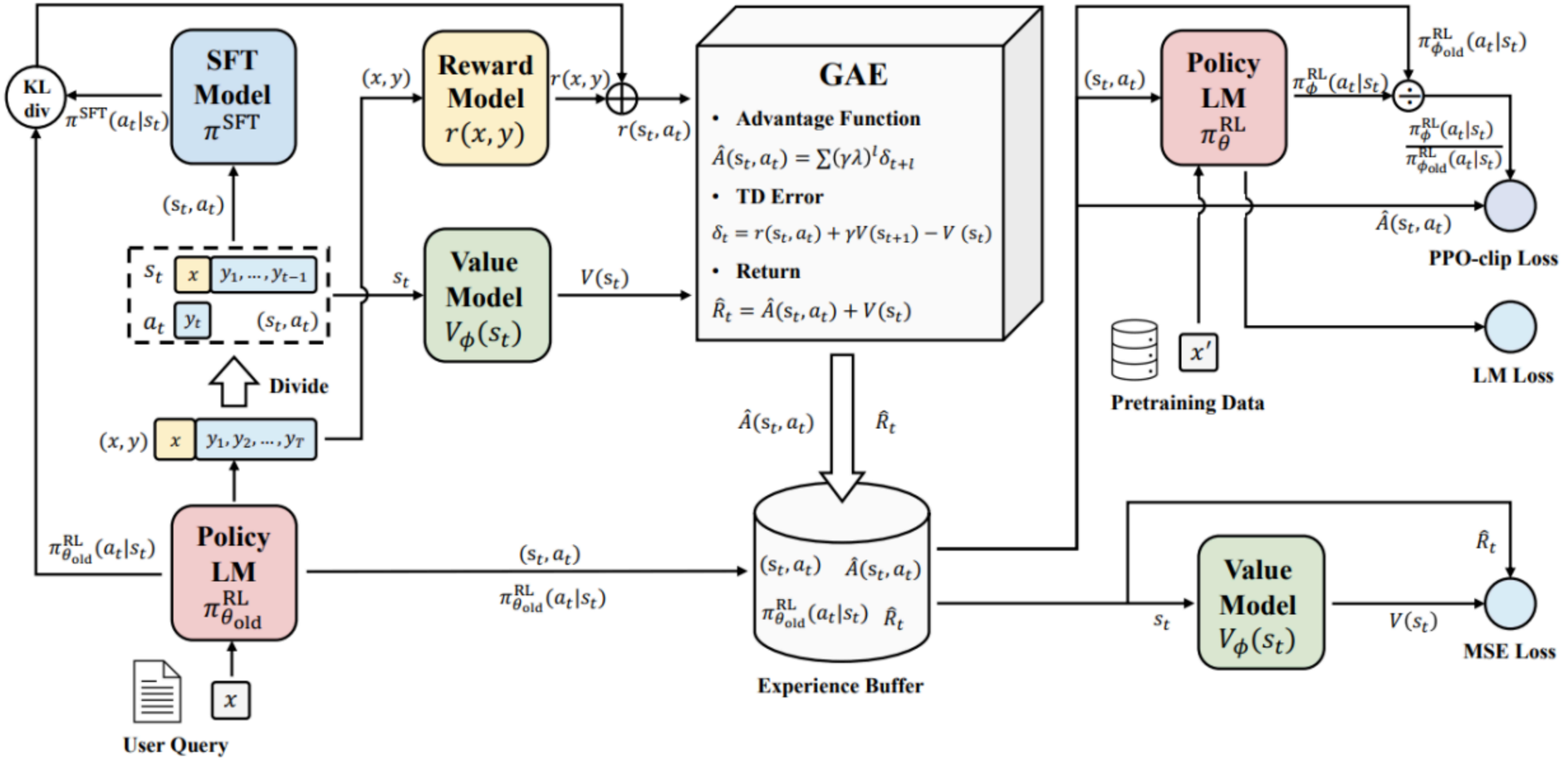
- Lots of algorithms that can be used: e.g. REINFORCE [Williams 92].
- Most popular: **Proximal Policy Optimization (PPO)** [Schulman et al, 2017]
  - Used by OpenAI

Learning to summarize from human feedback

Nisan Stiennon* Long Ouyang* Jeff Wu* Daniel M. Ziegler* Ryan Lowe*
Chelsea Voss* Alec Radford Dario Amodei Paul Christiano*
OpenAI



PPO algorithm (under the hood)



[Secrets of RLHF. Zheng et al. 2023]

Limitations: Reward Hacking

- Reward model might be biased towards some “features” due to the composition of the training data.

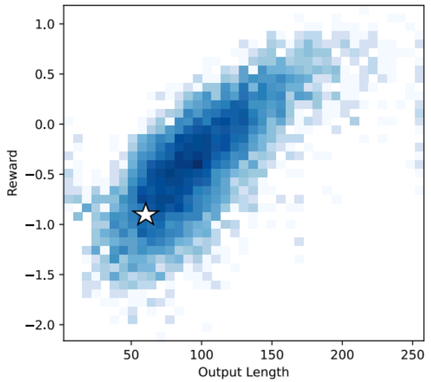
A Long Way to Go: Investigating Length Correlations in RLHF

Prasann Singhal
The University of Texas at Austin
prasanns@cs.utexas.edu

Tanya Goyal
Princeton University
tanyagoyal@princeton.edu

Jiacheng Xu
Salesforce AI
jiacheng.xu@salesforce.com

Greg Durrett
The University of Texas at Austin
gdurrett@cs.utexas.edu



Question: *Why don't adults roll off the bed?*

☆ **SFT (Before); 59 tokens**

Adults typically do not roll off of the bed because they have developed the muscle memory to keep their bodies from involuntarily moving during sleep and maintaining proper posture.

RLHF (After); 243 tokens: **Similar output, but much longer / more details**

Adults generally do not roll off of the bed because they have developed muscle control and awareness which helps them to remain upright while sleeping. Additionally, most adults find it uncomfortable or even painful to move around in their sleep, so rolling off the bed would be difficult without waking up first. There may also be safety concerns such as falling out of bed and potentially hurting oneself if one were to lose balance or fall from the ...

Figure 1: Log-scaled heatmap of lengths of SFT outputs vs. learned reward model scores for WebGPT (left). The graph shows that reward scores are strongly correlated with length. RLHF with these consistently leads to longer outputs (right).

Limitations: Reward Hacking

- Reward model might be biased towards some “features” due to the composition of the training data.

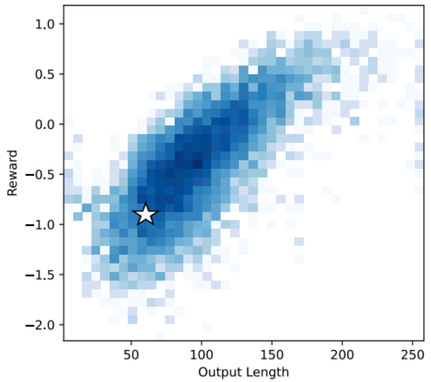
A Long Way to Go: Investigating Length Correlations in RLHF

Prasann Singhal
The University of Texas at Austin
prasanns@cs.utexas.edu

Tanya Goyal
Princeton University
tanyagoyal@princeton.edu

Jiacheng Xu
Salesforce AI
jiacheng.xu@salesforce.com

Greg Durrett
The University of Texas at Austin
gdurrett@cs.utexas.edu



Question: *Why don't adults roll off the bed?*

SFT (Before); 59 tokens

Adults typically do not roll off of the bed because they have developed the muscle memory to keep their bodies from involuntarily moving during sleep and maintaining proper posture.

RLHF (After); 243 tokens: **Similar output, but much longer / more details**

Adults generally do not roll off of the bed because they have developed muscle control and awareness which helps them to remain upright while sleeping. Additionally, most adults find it uncomfortable or even painful to move around in their sleep, so rolling off the bed would be difficult without waking up first. There may also be safety concerns such as falling out of bed and potentially hurting oneself if one were to lose balance or fall from the ...

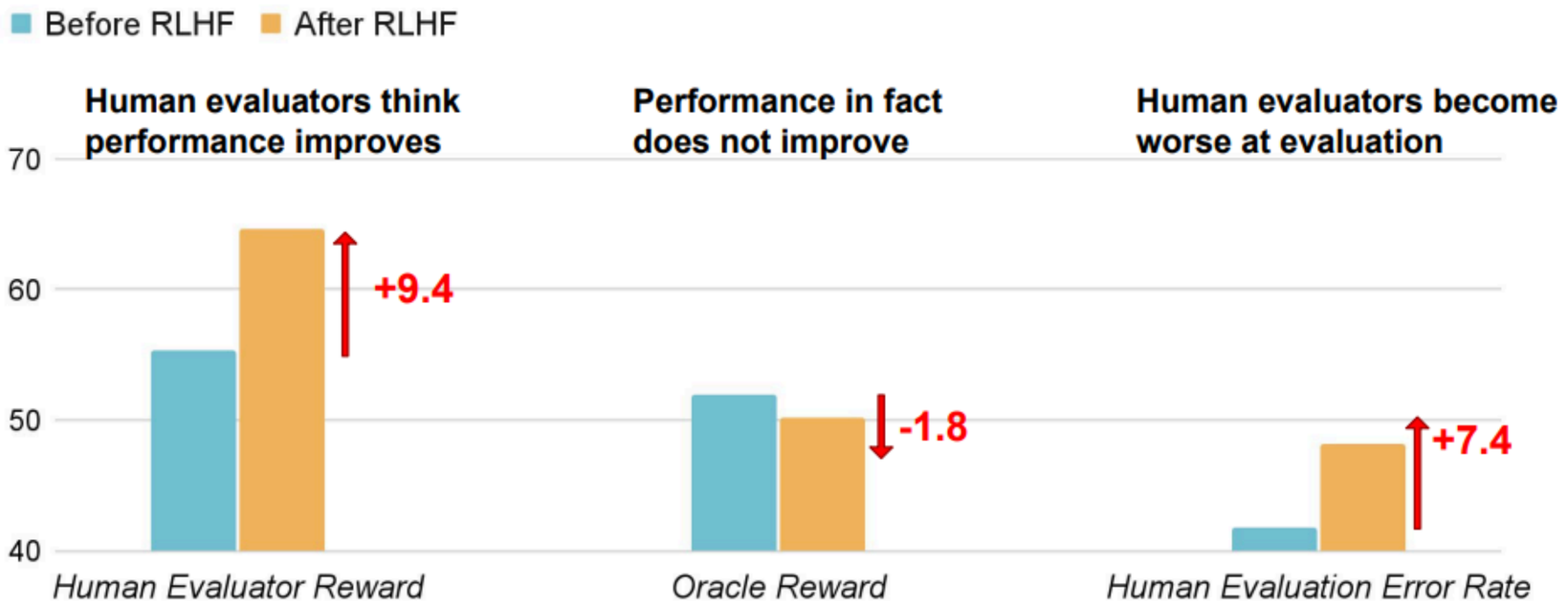
Figure 1: Log-scaled heatmap of lengths of SFT outputs vs. learned reward model scores for WebGPT (left). The graph shows that reward scores are strongly correlated with length. RLHF with these consistently leads to longer outputs (right).

- Found that reward models prefer longer texts. Most of the improvement after RLHF was because of longer texts.

Limitations: Reward Hacking

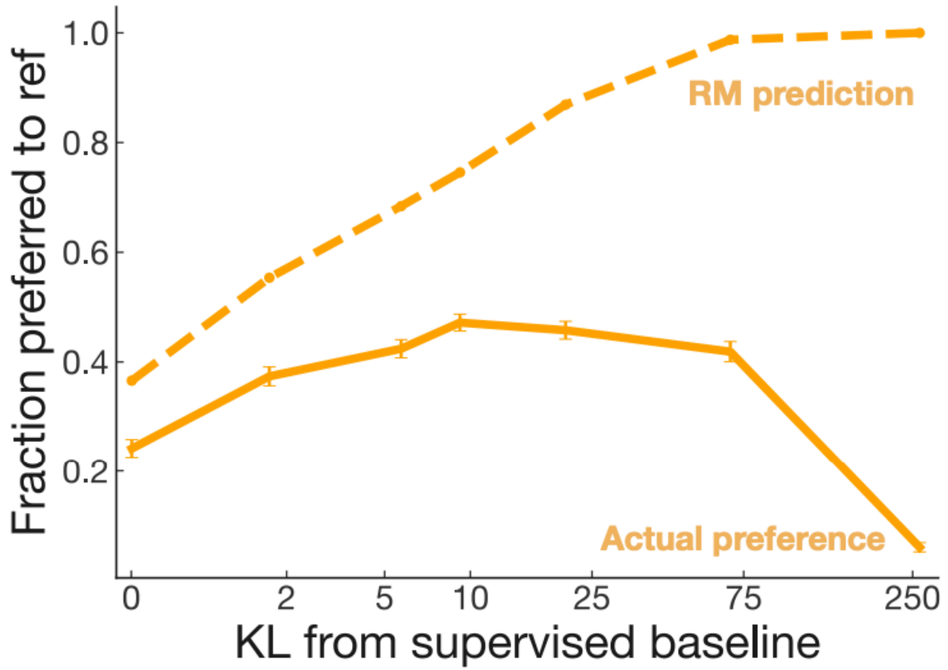
Language Models Learn to Mislead Humans via RLHF

Jiaxin Wen[1], **Ruiqi Zhong**[2], **Akbir Khan**[3], **Ethan Perez**[3], **Jacob Steinhardt**[2]
Minlie Huang[1], **Samuel R. Bowman**[3 4], **He He**[4], **Shi Feng**[4,5]
[1]Tsinghua University [2]Univeristy of California, Berkeley [3]Anthropic
[4]New York University [5]George Washington University



Potential Issues: Reward Hacking

- Reward over-optimization?
  - The reward model is learnt on a static preference dataset (x, y_+, y_-)
  - Usually, these outputs are sampled from $p_{SFT}(y \mid x)$
  - Will reward model scores generalize to outputs sampled from $p_\theta(y \mid x)$?



Slide Acknowledgements

- Earlier versions of this course offerings including materials from Claire Cardie, Marten van Schijndel, Lillian Lee.
- Intro to NLP class by Greg Durrett from UT Austin