

The Evolution of GPT

The Evolution of GPT

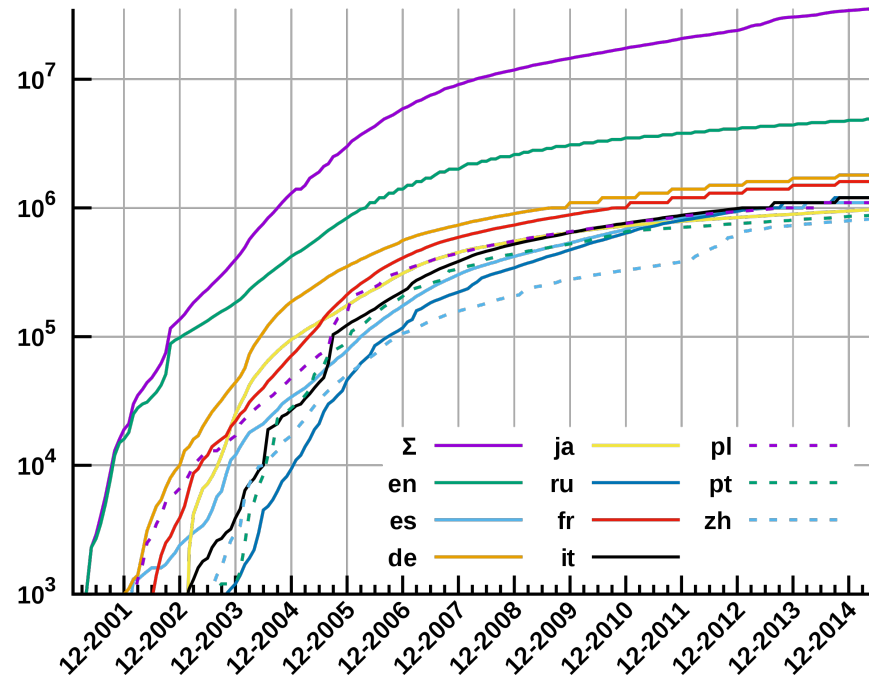
- Byte Pair Encoding
- Generative Pretraining
- Multi-task learning
- Prompt Engineering
- Reinforcement Learning from Human Feedback (RLHF)

Main Issues in Neural Language Modeling

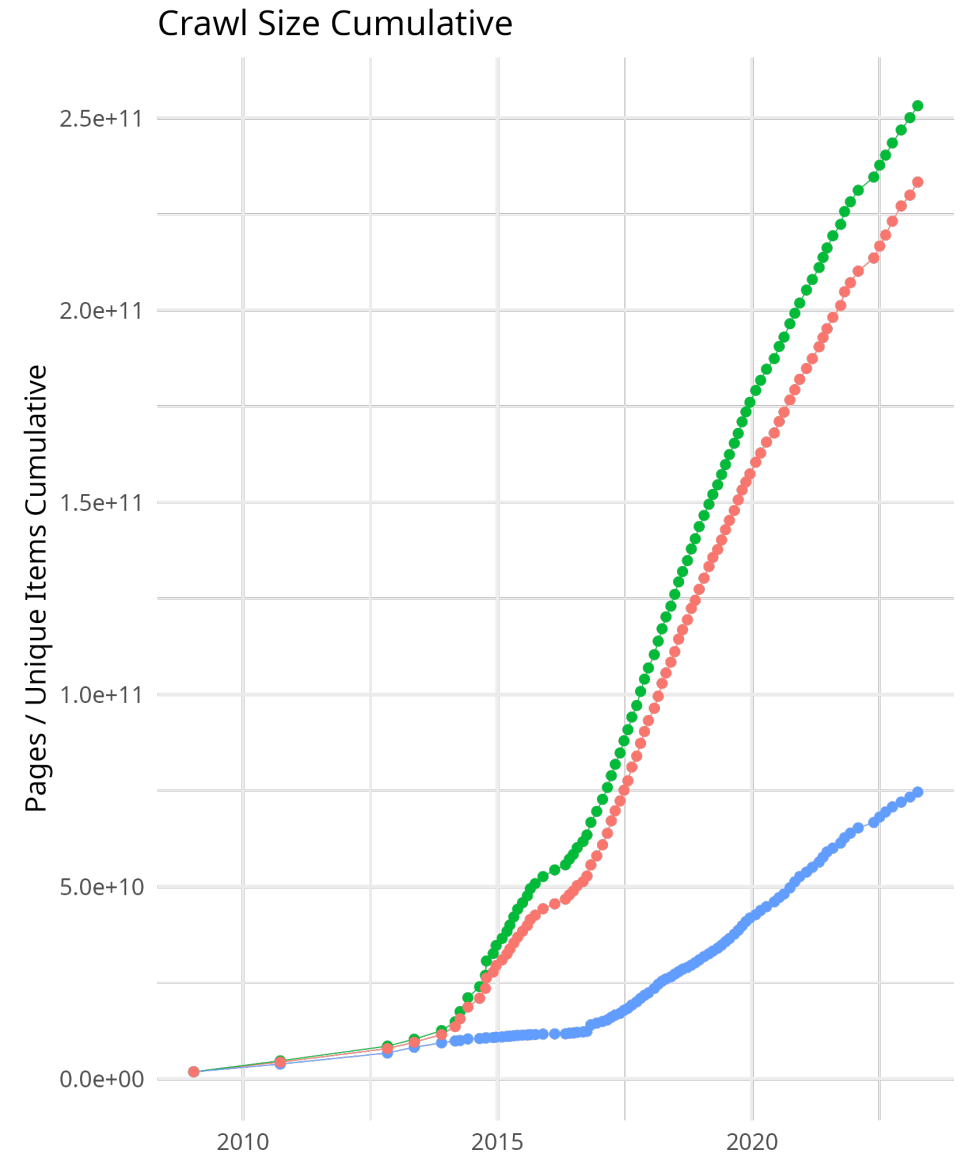
- Data
- Tokenization
- Compute
- Benchmarking

Data

- Rise of openly accessible data sets



Wikipedia (~21GB)



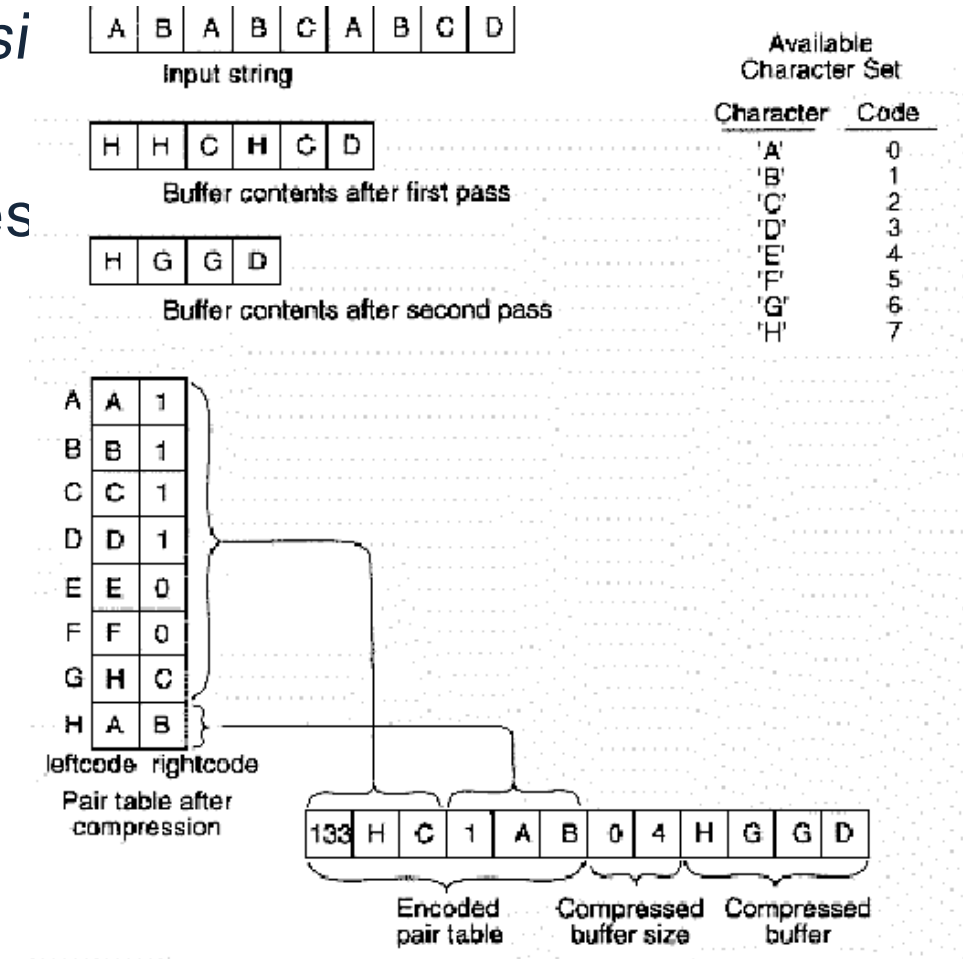
CommonCrawl (~380TB, 2022)

Tokenization

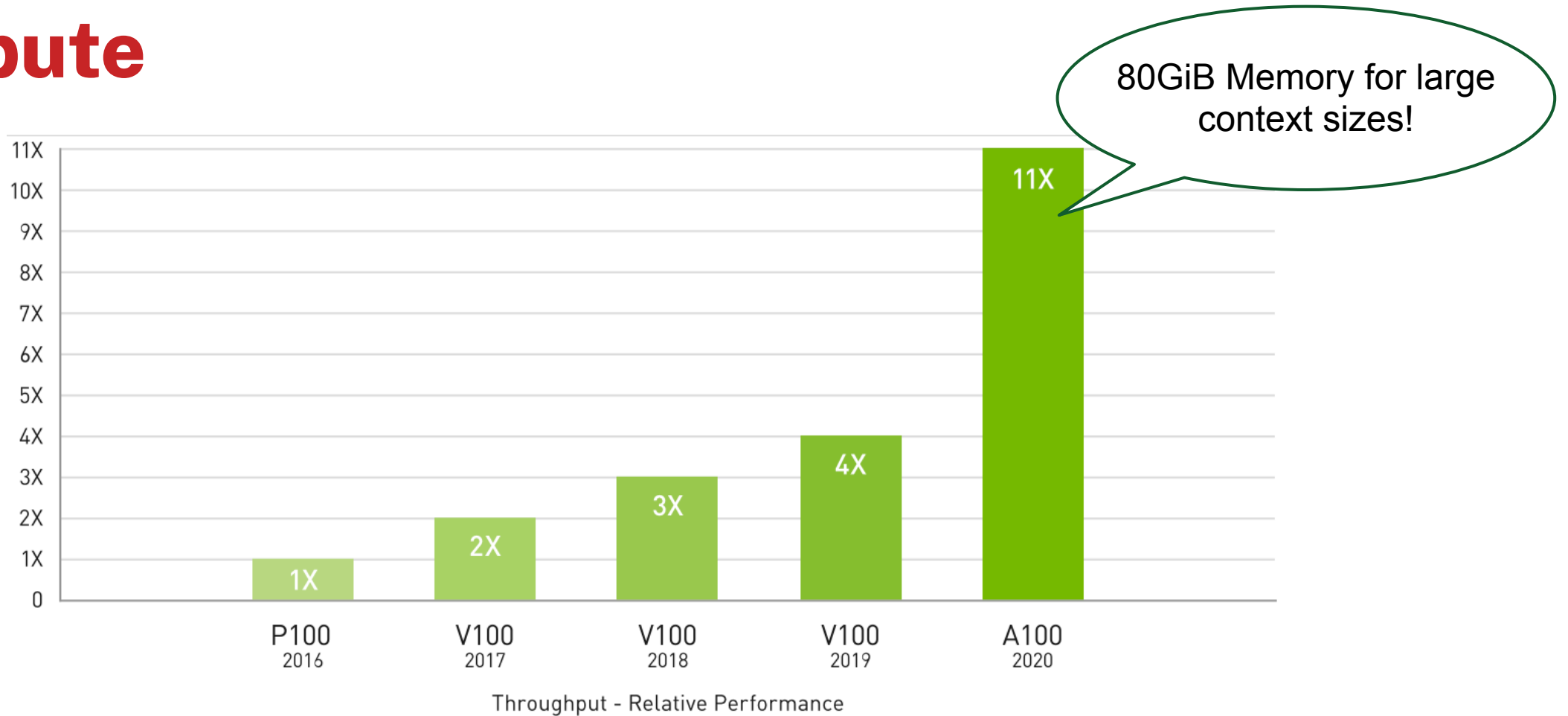
- Traditional NLP
 - word-based using a lexicon
 - stemming
- Big data NLP
 - character n-grams (cf. fasttext)
 - **Byte Pair Encoding (BPE)**

Byte Pair Encoding

- P. Gage, 1994: *A new algorithm for data compression*
Volume 12, Issue 2, 1994
 - Replace common pairs of bytes by single bytes
 - In-memory, multi-pass
- Modern NLP
 - adjust for unicode
 - Sennrich, Haddow and Birch, 2015
(<https://arxiv.org/abs/1508.07909>)
- Note: Same tokenizers used for Whisper!



Compute



Source: Nvidia

Benchmarking



- <https://gluebenchmark.com/>
 - General Language Understanding Evaluation
 - CoLA: Linguistic Acceptability
 - SST-2: Sentiment
 - MRPC, QQP: semantic equivalence
 - STS-B: test similarity
 - MNLI, RTE: textual entailment
 - QNLI: is-answer?
 - WNLI: entailment after reference substitution

Generative Pre-Trained Transformers

- Prior work focused on learning models for specific tasks (sentiment, entailment, etc.) – they didn't generalize well!
- *Better: semi-supervised learning* (and some tricks)
 1. Unsupervised Language Modelling (“pre-training”)
 2. Supervised fine-tuning
 3. Task-specific input transformations

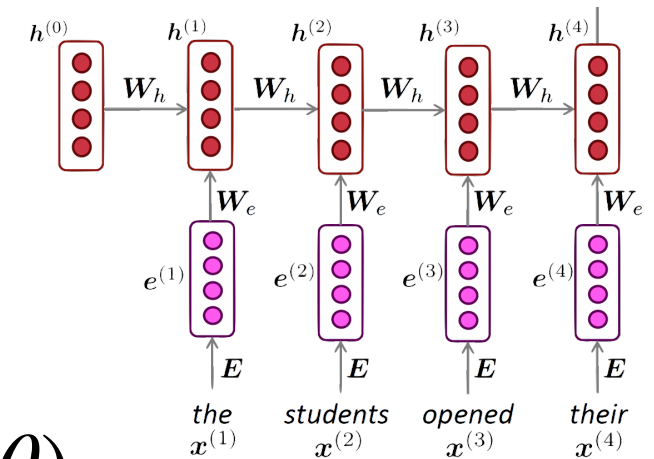
GPT: LM pre-training

- Train an auto-regressive transformer (decoder) language model
- Using BPE, token & positional embedding
- ...on large (!) quantities of text!

$$\mathcal{L}_1(T) = \sum_i \log P(t_i | t_{i-k}, \dots, t_{i-1}; \theta)$$

sequence of tokens

context window size



GPT: Supervised Fine-Tuning

- Assemble a dataset \mathcal{C} with sequences \mathbf{x} and according labels y
- (Super)GLUE gives us a rich set of tasks and datasets!
- Add linear output layer to final transformer block

$$L_2(\mathcal{C}) = \sum_{(\mathbf{x}, y)} \log P(y | x_1, \dots, x_m; \theta)$$

dataset

label of this sequence

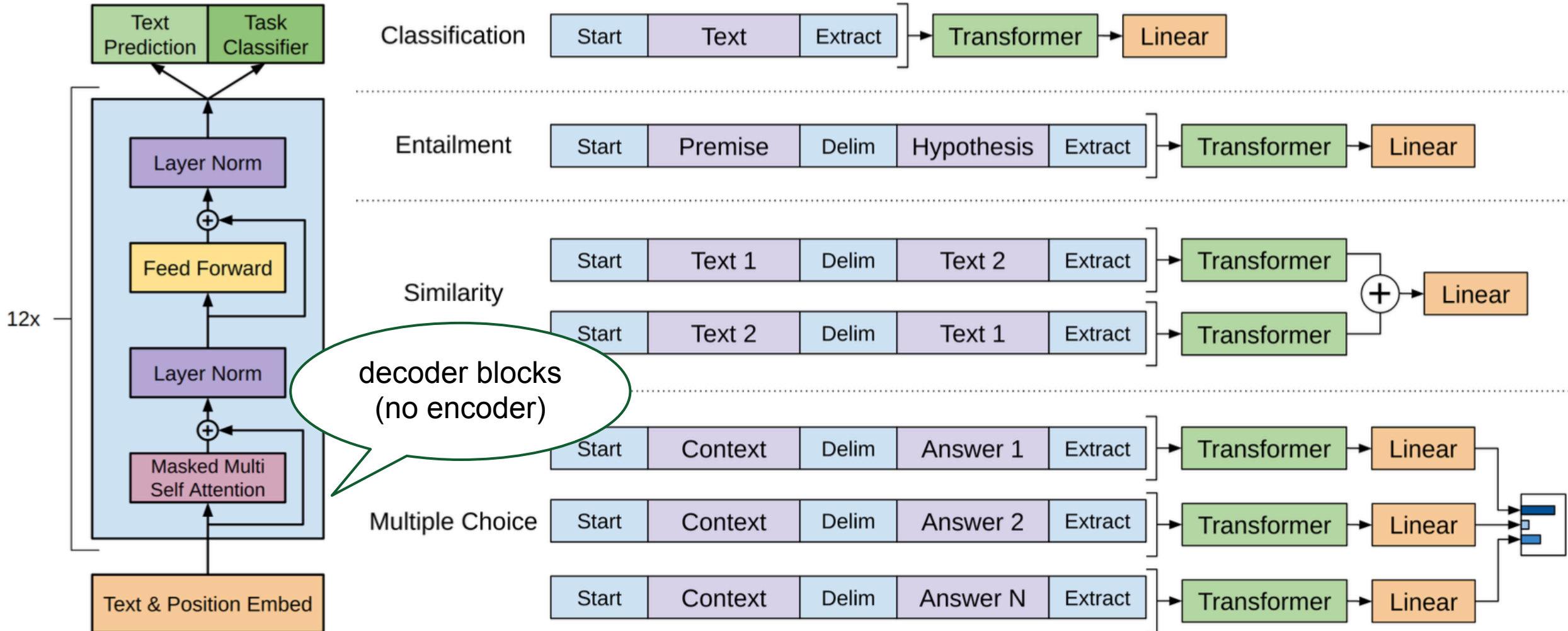
token sequence

Optionally, for better performance and convergence: $L_3 = L_2(\mathcal{C}) + \lambda L_1(\mathcal{C})$

GPT: Task-specific input transformations

- For supervised training, we need to rearrange the input so that it works with our architecture
 - Start and end tokens for input sequences
 - Delimiter tokens in between parts of input
- Textual entailment: introduce '\$' token in between premise and hypothesis
- Similarity: provide pairs in both orders
- QA/Reasoning: [document; question; \$; answer]

GPT: Architecture at a Glance



GPT: Some More Details

- Dataset: BooksCorpus (~7000 unpublished books)
- BPE with 40,000 merges
- Context token size (!= words): 512
- 12 decoder blocks with 12 attention heads (each)



nota bene: not words!

	GPT (2018)
Data	BooksCorpus (~5GB)
BPE	40,000
Parameters	117 Million
Decoder Layers	12
Context Token Size	512
Hidden Layer	768
Batch Size	64

Results from the original tech report

Table 4: Semantic similarity and classification results, comparing our model with current state-of-the-art methods. All task evaluations in this table were done using the GLUE benchmark. (*mc*= Mathews correlation, *acc*=Accuracy, *pc*=Pearson correlation)

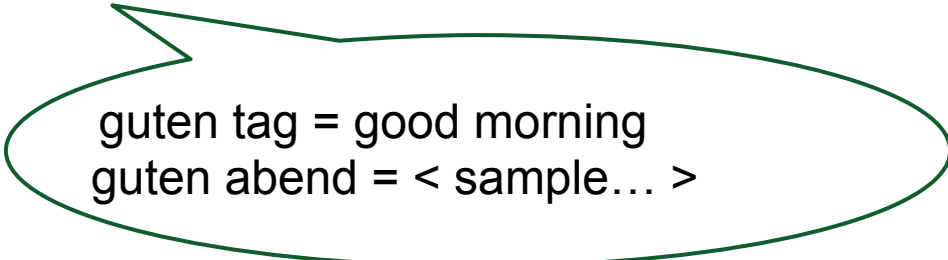
Method	Classification		Semantic Similarity			GLUE
	CoLA (mc)	SST2 (acc)	MRPC (F1)	STSB (pc)	QQP (F1)	
Sparse byte mLSTM [16]		93.2	-	-	-	-
TF-KLD [23]		-	86.0	-	-	-
ECNU (mixed ensemble)		-	-	-	-	-
Single-task BiLSTM + ELMo + Attn [64]	<u>35.0</u>	90.2	80.2	72.8	70.3	68.9
Multi-task BiLSTM + ELMo + Attn [64]	18.9	91.6	83.5	72.8	70.3	68.9
Finetuned Transformer LM (ours)	45.4	91.3	82.3	82.0	70.3	72.8

2018: Hey, LSTMs still around! 🙄

6/2023: 91% 🙄

GPT-2

- Basically like GPT, just bigger
 - Larger context, more parameters
 - More data: WebText (40GB human curated, by tracing reddit outbound)
 - Better BPE (prevent split across character categories), 50k
- Paving the way to zero shot learning
 - Introduced task conditioning (ie. same input but different output depending on task)
 - Instead of separators, use natural language instructions



guten tag = good morning
guten abend = < sample... >

GPT-2: Zero Shot Learning

- Technically, no training or fine-tuning allowed
- Model is “primed” with training data, e.g.
 - “guten tag = good morning” ...
- At last, sample from model to get answer, e.g.
 - “guten abend = ...”

GPT-2: some more details

	GPT (2018)	GPT-2 (2019)
Data	BooksCorpus (5GB)	WebText (40GB)
BPE	40k	50k (tweaked)
Parameters	117 Million	1.5 Billion
Decoder Layers	12	48
Context Token Size	512	1024
Hidden Layer	768	1600
Batch Size	64	512

Results from the original tech report

GPT

These are all rather simple “fill the gap” tests

Language Models are Unsupervised Multitask Learners

	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	Ok, we get it: bigger is better!					1.08	18.3	21.8
117M	35.13	45.99						1.17	37.50	75.20
345M	15.60	55.48						1.06	26.37	55.72
762M	10.87	60.12						1.02	22.05	44.575
1542M	8.63	63.24						0.98	17.48	42.16

GPT-2

Table 3. Zero-shot results on many datasets. No training or fine-tuning was performed for any of these results. PTB and WikiText-2 results are from (Gokul et al., 2018). CBT results are from (Bajgar et al., 2016). LAMBADA accuracy result is from (Hoang et al., 2018) and LAMBADA perplexity result is from (Grave et al., 2016). Other results are from (Dai et al., 2019).

Results from the original tech report (2)

TL;DR is often found in the reddit data!

	R-1	R-2	R-L	R-AVG
Bottom-Up Sum	41.22	18.68	38.34	32.75
Lede-3	40.38	17.66	36.62	31.55
Seq2Seq + Attn	31.33	11.81	28.83	23.99
GPT-2 TL; DR:	29.34	8.27	26.58	21.40
Random-3	28.78	8.63	25.52	20.98
GPT-2 no hint	21.58	4.03	19.47	15.03

Still, pretty terrible results on a fairly simple dataset

Table 4. Summarization performance as measured by ROUGE F1 metrics on the CNN and Daily Mail dataset. Bottom-Up Sum is the SOTA model from (Gehrmann et al., 2018)

GPT-3

- **Vision:** Build a general model (“foundation model”) that will learn any task with only a few examples (“few shot learner”)
- More of everything...

	GPT (2018)	GPT-2	GPT-3
Data	BooksCorpus (5GB)	+WebText (40GB)	+CommonCrawl+Wiki_en
Parameters	117 Million	1.5 Billion	175 Billion
Decoder Layers	12	48	96
Context Token Size	512	1024	2048
Hidden Layer	768	1600	12288
Batch Size	64	512	3.2M

GPT-3

- **“in-context learning”:**
 - prepend examples of the task before your actual example/query
 - $k = 0$: zero-shot
 - $k = 1$: one-shot
 - $k > 1$: few-shot
 - ...but still no gradient update!

GPT-3: Few-Shot Learning on SuperGLUE

Few-Shot

Translate English to French:
 peppermint => menthe poivrée
 sea otter => loutre de mer
 ...
 cheese =>

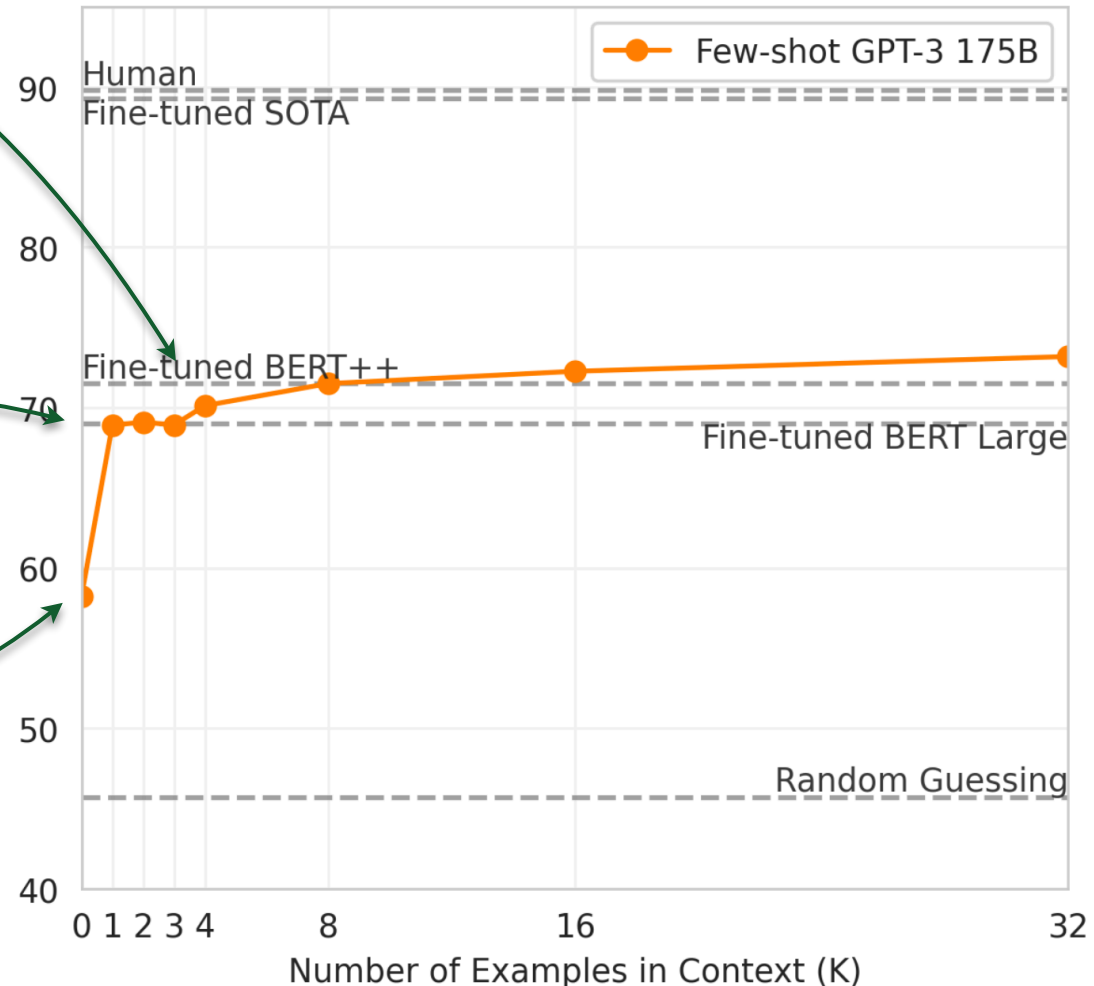
One-Shot

Translate English to French:
 sea otter => loutre de mer
 cheese =>

Zero-Shot

Translate English to French:
 cheese =>

In-Context Learning on SuperGLUE



GPT-3: Results from the original paper

	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	89.0	91.0	96.9	93.9	94.8	92.5
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	76.1	93.8	62.3	88.2	92.5	93.3
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1

Table 3.5: Performance of GPT-3 on SuperGLUE compared to fine-tuned baselines and SOTA. All results are reported on the test set. GPT-3 few-shot is given a total of 32 examples within the context of each task and performs no gradient updates.

Chain-of-Thought Prompting

- In-context learning seems to have limited performance
- Solution: Change the prompts!

Standard Prompting

Model Input

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: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

Chain-of-Thought Prompting

Model Input

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.

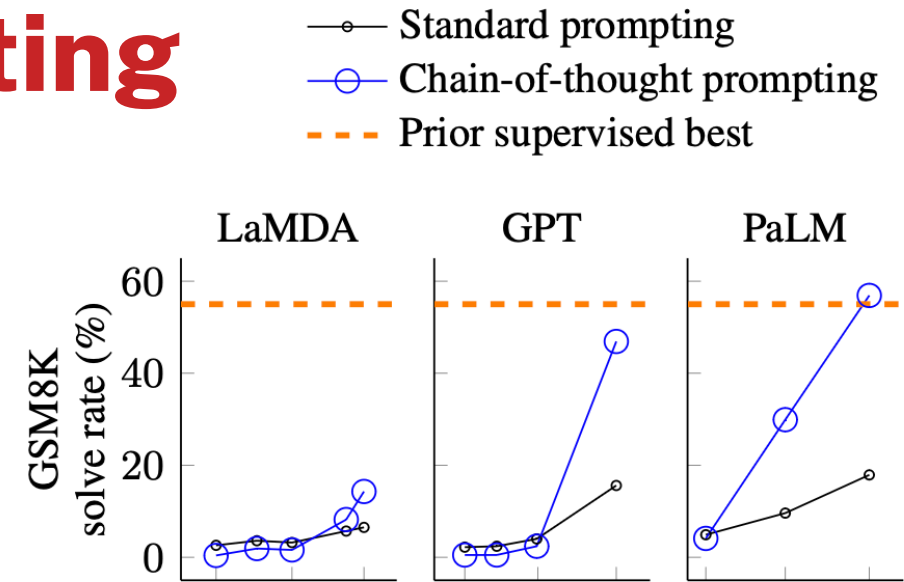
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅

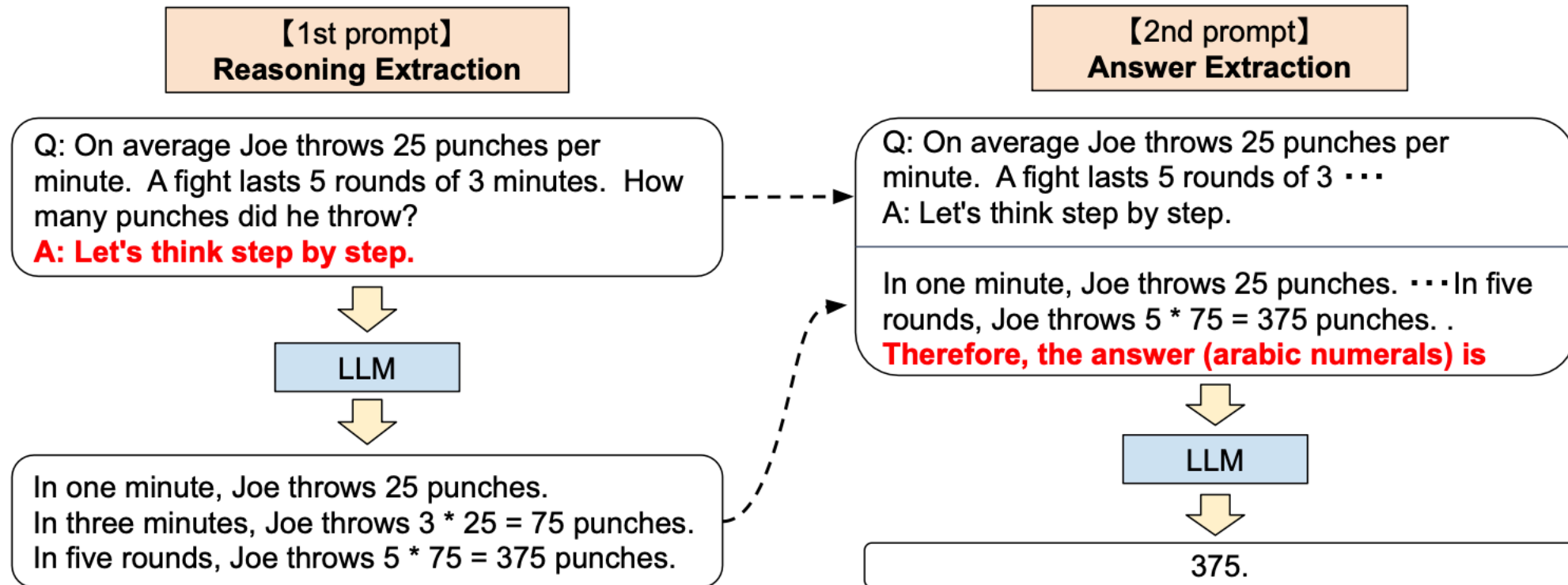
Chain-of-Thought Prompting

- Yes, it works!
- But...




As for limitations, we first qualify that although chain of thought emulates the thought processes of human reasoners, this does not answer whether the neural network is actually “reasoning,” which we leave as an open question. Second, although the cost of manually augmenting exemplars with chains of thought is minimal in the few-shot setting, such annotation costs could be prohibitive for finetuning (though this could potentially be surmounted with synthetic data generation, or zero-shot generalization). Third, there is no guarantee of correct reasoning paths, which can lead to both correct and incorrect answers; improving factual generations of language models is an open direction for

Zero-Shot Chain-of-Thought



Zero-Shot Chain-of-Thought (on MultiArith dataset)

No.	Category	Template	Accuracy
1	 instructive	Let's think step by step.	78.7
2		First, (*1)	77.3
3			74.5
4		Let's break it into steps. (*2)	72.2
5		Let's reason 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

Language Modeling \neq Assisting Users

Prompt *Explain the moon landing to a 6 year old in a few sentences.*

Completion GPT-3

Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

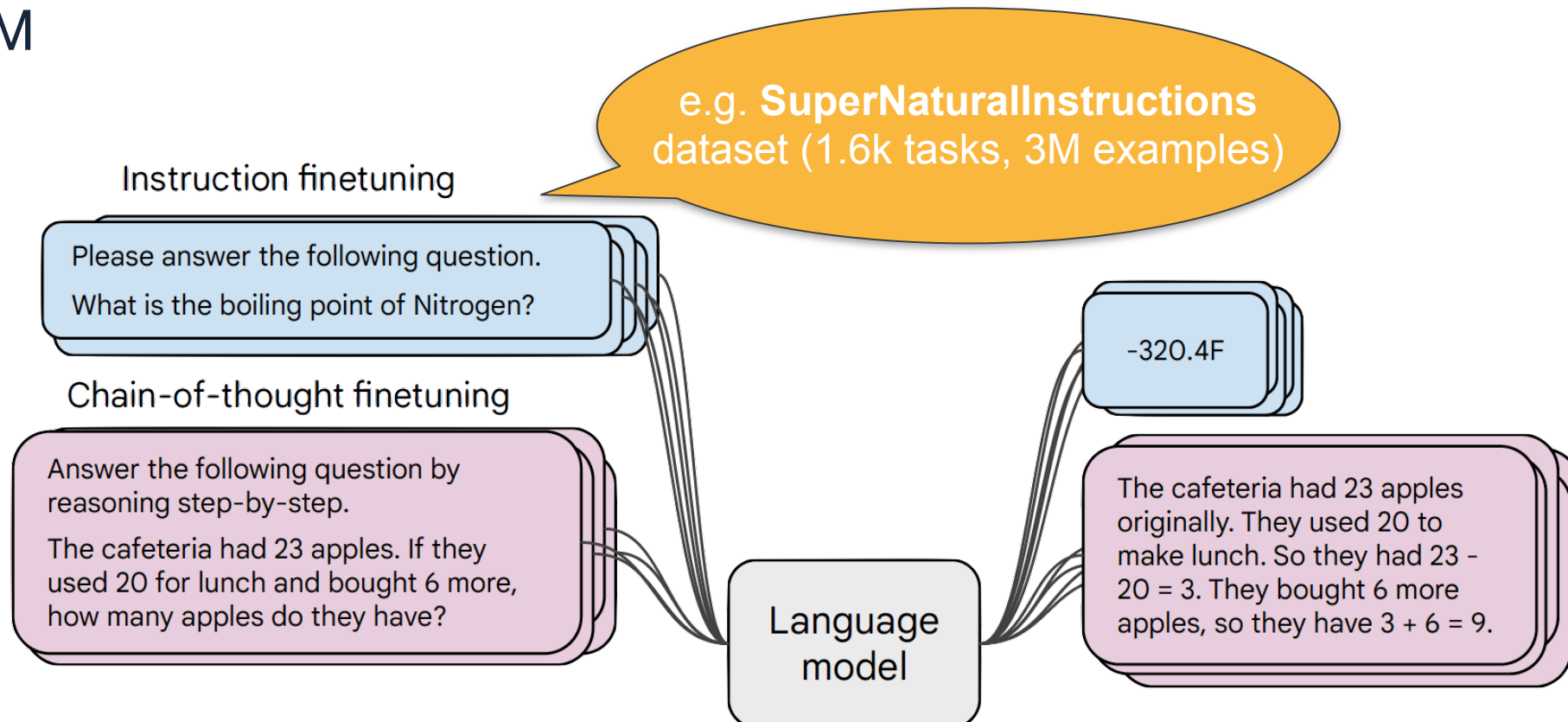
Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.

LMs are not aligned with user intent!

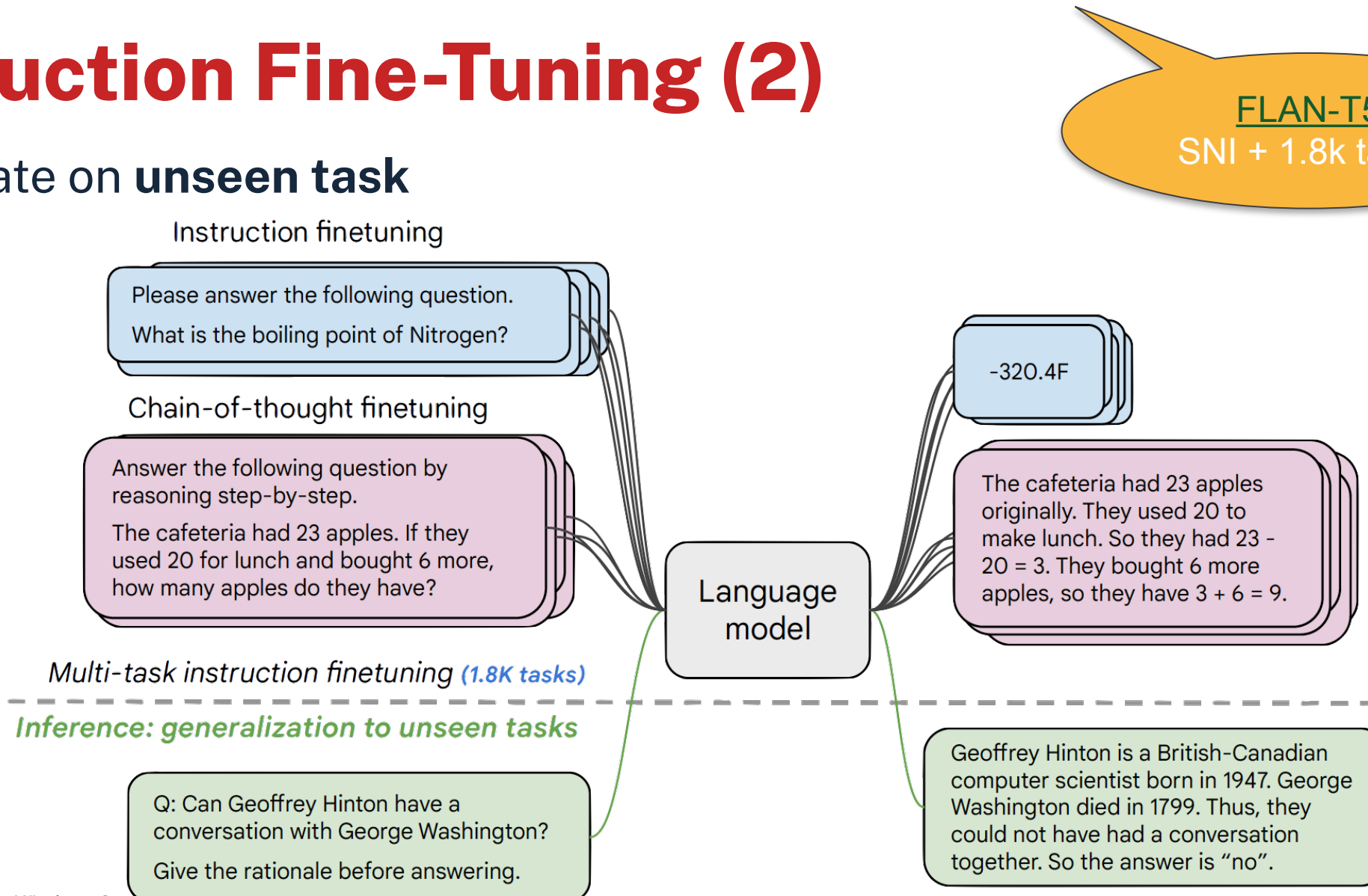
Instruction Fine-Tuning (1)

- Collect examples of (instruction, output) pairs across **many tasks** and fine-tune the LLM



Instruction Fine-Tuning (2)

- Evaluate on **unseen task**



Limits of Instruction Fine-Tuning (FLAN)

- Ground-truth data is expensive to collect
- Open-ended (creative) tasks do not have a right answer
 - “Write a poem about deep learning”
- Language Modeling penalizes all token-level mistakes equally – but some errors are worse than others!
 - “You’re fired” vs. “You’re hired”!

From LMs to Assistants: Recap

- Zero-shot and few-shot in-context learning
 - No fine-tuning needed
 - prompt engineering can improve performance
 - Limits to what you can fit in context
 - Complex tasks will probably need parameter update
- Instruction fine-tuning
 - Simple and straight-forward, generalizes to unseen tasks
 - Collecting ground-truth for many tasks is expensive (and exhaustive...)

Mismatch between LM object and human preference

Optimizing for Human Preference

- Let's say, we're training for summarization
- Remember: We can sample multiple outputs!

SAN FRANCISCO,
California (CNN) --
A magnitude 4.2
earthquake shook the
San Francisco
...
overturn unstable
objects.

An earthquake hit San
Francisco. There was
minor property damage,
but no injuries.

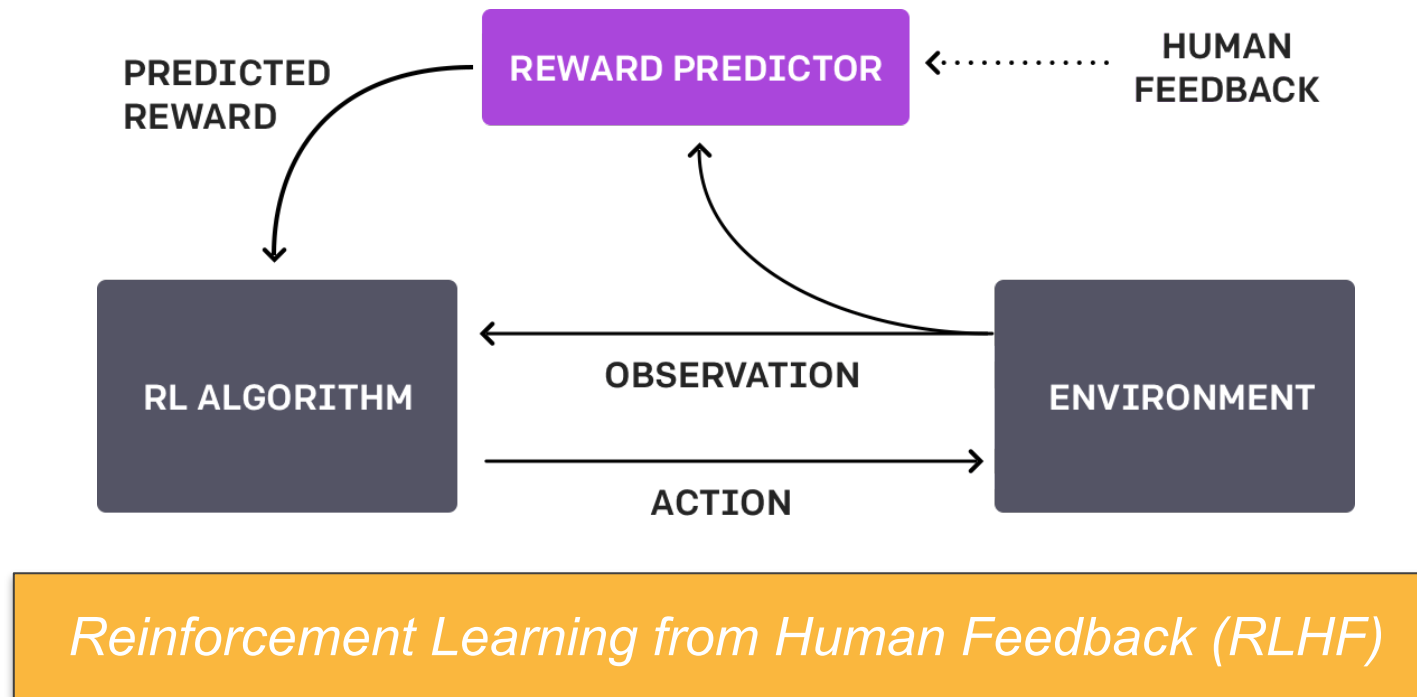


The Bay Area has good
weather but is prone
to earthquakes and
wildfires.



Optimizing for Human Preference

- Which one is better, or: which one has a higher reward?
- Mathematical framework: policy gradient for reinforcement learning



A Model of Human Preference (1)

- **Problem:** human-in-the-loop is costly (and slow)
- **Solution:** Model human preference as a separate (NLP) task 🧐

A 4.2 magnitude
earthquake hit San
Francisco, resulting
in massive damage.



A Model of Human Preference (2)

- **Problem:** humans don't agree and are often miscalibrated
- **Solution:** ask for pair-wise comparison (binary preference)

An earthquake hit San Francisco. There was minor property damage, but no injuries.

>

A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage.

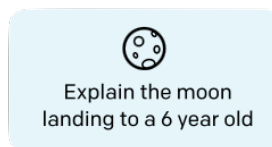
>

The Bay Area has good weather but is prone to earthquakes and wildfires.

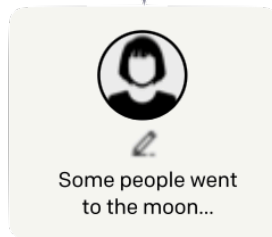
Step 1

**Collect demonstration data,
and train a supervised policy.**

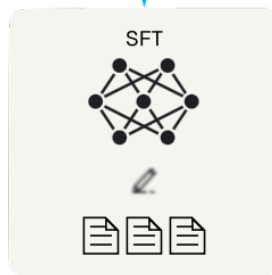
A prompt is
sampled from our
prompt dataset.



A labeler
demonstrates the
desired output
behavior.



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

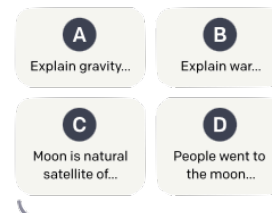
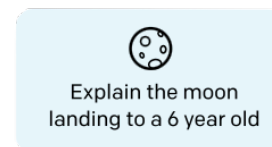


30k tasks!

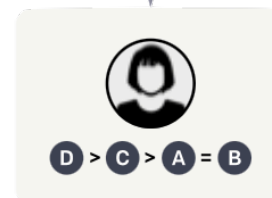
Step 2

**Collect comparison data,
and train a reward model.**

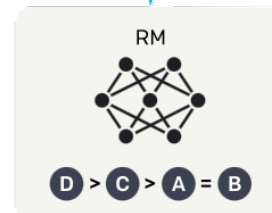
A prompt and
several model
outputs are
sampled.



A labeler ranks
the outputs from
best to worst.



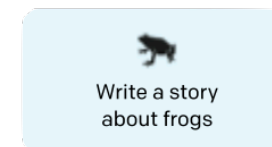
This data is used
to train our
reward model.



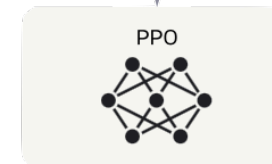
Step 3

**Optimize a policy against
the reward model using
reinforcement learning.**

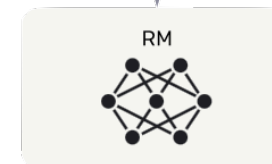
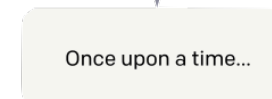
A new prompt
is sampled from
the dataset.



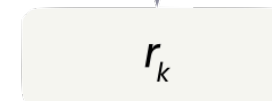
The policy
generates
an output.



The reward model
calculates a
reward for
the output.



The reward is
used to update
the policy
using PPO.



Ouyang et al., 2022: Training LMs to follow instructions with human feedback

ChatGPT

- OpenAI not so open anymore... details not really revealed
- Blog post suggests:
 - GPT-3.5 (bigger, you guessed it...)
 - Instruction fine-tuning
 - RLHF
 - ...on dialog data
- **Unclear:** How much “plain” (handcrafted, rule-based) engineering is
 - in the dialog state?
 - in the “last mile”? (e.g. for SQL queries, API calls)

A Note on RL with Reward Modeling

- **Human preference** is unreliable (and subject to change...)
 - “Reward hacking” is an issue in RL
 - Chatbots are rewarded to produce responses that seem authoritative and helpful, *regardless of truth*
 - Possible root cause of “alternative facts” and hallucinations
- **Models** of human preference will by design be inferior 🙄



Percy Liang
@percyliang

...

RL from human feedback seems to be the main tool for alignment. Given reward hacking and the falliability of humans, this strategy seems bound to produce agents that merely appear to be aligned, but are bad/wrong in subtle, inconspicuous ways. Is anyone else worried about this?

OpenAI is hiring developers to make ChatGPT better at coding

Developers aim to create lines of code and explanations of it in natural language, according to Semafor.

7:55 am · 7 Dec 2022 <https://twitter.com/percyliang/status/1600383429463355392>

Recap

- GPT
 - Generative pre-trained transformers for language modelling
 - Fine-tuned to (multitude of) tasks
 - “bigger is better”
- Prompt engineering (zero-/one-/few-shot)
- Instruction fine-tuning
- Reinforcement Learning from Human Feedback