Google

Prompt Engineering

Aligning models to user's intentions

vsubhashini@

Goals
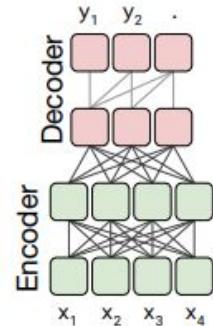
Foundations and familiarity with terms

- How are LLMs developed?
- What strategies can be used to get LLMs to solve a problem?
- Get familiar with terminologies used by researchers & practitioners to refer to variants of LLMs, trade-offs, and interaction strategies.
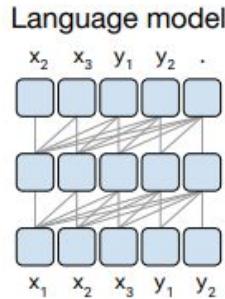
Short "Refresher" Version

Google

Background: Pre-training

std. enc-dec   Causal LM   Prefix LM   Masked LM

Targets

<X> the force

Inputs

May <X> be with you

figs: T5

Google

Background: Scaling and zero shot capabilities

fig: GPT3

Zero-shot? Adapting learned patterns to new situations

```python
2 a = model.query("""What's the sentiment of this review, positive or negative?
3 Review: Honestly it isn't THAT fresh. """)
4 print(wrap(a))
```

negative

```python
1 a = model.query("""Translate English to German.
2 English: The kids love ice cream.""")
3 print(a)
```

Die Kinder lieben Eis.

```python
1 a = model.query("""Write code to select rows from a dataframe where
2 column A is value1 or value2.""")
3 print(a)
```

```python
import pandas as pd

df = pd.DataFrame({'A': ['value1', 'value2', 'value3', 'value4']})

# Select rows where column A is value1 or value2
df = df[df['A'].isin(['value1', 'value2'])]

print(df)
```

Google

Model training

Pre-training

Self-supervised, on lots of examples O(Trillion) tokens. E.g. LLAMA

Fine-tuning

Enhanced for certain interactions. O(1000) to O(M) examples. E.g. PaLM-Coder, Codex

RL w. Human Feedback

Tuned to help model guess better under uncertainties. O(1000) examples. E.g. Bard, ChatGPT

Prompt-tuning

Steered for focused tasks. O(100)examples

Prompting

Steered for a query. O(1) examples (usually 0-8)

Pre-training, instruction based fine-tuning, RLHF - usually successive

These slides = (Prompting + prompt-tuning and a dash of fine-tuning)

Google

Available models

Raw LM

Continues to predict next tokens following a prompt

E.g. LLAMA, GPT3

Instruction tuned

Answer yes or no. Cheetahs are faster than hares. Answer:

E.g. FLAN-PaLM, InstructGPT

Dialog tuned

[Alice] where is eve?
[Bob] she is out.
[Alice] shall i call her in?
[Bob]: …

E.g. BlenderBot, Koala

RLHF

Human-Feedback to improve the model's responses under uncertainties.

E.g. Bard, ChatGPT

Google

Models from different stages respond differently

Raw LM

```
2 a = model.query("""Translate English to German.
3 English: The kids love ice cream.""")
4 print(a)
```

```
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
```

Instruction Tuned

```
2 a = model.query("""Translate English to German.
3 English: The kids love ice cream.""")
4 print(a)
```

```
Die Kinder lieben Eis.
```

Trained to follow instructions

RLHF model

```
2 a = model.query("""Translate English to German.
3 English: The kids love ice cream.""")
4 print(a)
```

+ Dialog

+ …

+ RLHF

```
The English sentence "The kids love ice cream" can be translated to German as:

```
Die Kinder lieben Eis.
```

This is a direct translation, with "die Kinder" meaning "the kids" and "lieben" meaning

Another possible translation is:

```
Die Kinder mögen Eis.
```

This is a slightly more colloquial translation, with "mögen" meaning "like" or "enjoy."
```

(as do different models)
PaLM, GPT3.5, Claude, LLAMA

Google

Note, LLMs are finicky

```
2 a = model.query("""What's the topic of this review.
3 A. Food
4 B. Cleanliness
5 C. Comfort
6 Review: The room was spacious.""")
7 print(a)
```

The topic of this review is "Comfort". The review is about the room's size and layout. It says

```
2 a = model.query("""What's the topic of this review.
3 A. Food
4 B. Cleanliness
5 C. Comfort
6 Review: The room was spacious.
7 Topic: """)
8 print(a)
```

C.

Prompting

Summarize feedback
from the comments.
c1: { … }, c2: {…}, …
summary: {

Zero-shot

* State your task
* Give the input
* For very long
instructions, having
some formatting "{}"
might help.

Categorize comment ….
c: { … } t: { 10mins }
c: { … } t: { summer }
c: { … } t: { vegan }
c: { … } t: {

Few-shot

* Give a few examples
* Order & choice might
affect results
* Many ways to choose
in-context examples.
* temperature, sampling

Given comments
predict topic for each,
and summarize inc.
topic histogram.

Chain-of-thought

* Walk through steps
* Ask model for
intermediate outputs
* Give examples
* Ask model to verify steps

Google

Prompting

Summarize feedback from the comments.
c1: { … }, c2: {...}, …
summary: {

Zero-shot

* State your task
* Give the input
* For very long instructions, having some formatting "{}" might help.

Categorize comment ….
c: { … } t: { 10mins }
c: { … } t: { summer }
c: { … } t: { vegan }
c: { … } t: {

Few-shot

* Give a few examples
* Order & choice might affect results
* Many ways to choose in-context examples.
* temperature, sampling

Given comments predict topic for each, and summarize inc. topic histogram.

Chain-of-thought

* Walk through steps
* Ask model for intermediate outputs
* Give examples
* Ask model to verify steps
* Make a template of the above (e.g. ReAct, Tree of thought)

Google

Enrich your Chain-of-Thought Template

Self-critique

Iterative interaction to improve the model. Sort of bootstrapping.

By asking the model to verify output of each step.

Retrieval Augmented

Augment with external memory.

Steps can include searching on the net or a database (logs, history etc.)

Tool Augmented

Augment with external tools, APIs, **

Steps can include calls to tools e.g. write **and execute** code.

Tuning

Summarize feedback
from the comments.
c1: { ... }, c2: {...}, ...
summary: {

Categorize comment ....
c: { ... } t: { 10mins }
c: { ... } t: { summer }
c: { ... } t: { vegan }
c: { ... } t: {

Given comments
predict topic for each,
and summarize inc.
topic histogram.

Zero-shot

Few-shot

Chain-of-thought

Prompt-tune, LoRA
O(100) examples

* State your task
* Give the input
* For very long
instructions, having
some formatting "{}"
might help.

* Give a few examples
* Order & choice might
affect results
* Many ways to choose
in-context examples.
* temperature, sampling

* Walk through steps
* Ask model for
intermediate outputs
* Give examples
* Ask model to verify steps
* Make a template of the
above (e.g. ReAct, Tree of
thought)

Fine-tune
O(1000) examples

Google
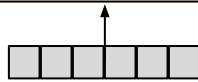
Parameter-Efficient Tuning - LoRA

high training & deployment cost   Low training & deployment cost

Fine-tuning

Pre-trained Models
Tuneable

Input Text

❌ Sample expensive
❌ Slow iteration
✅ High quality
❌ Single Task Training & Serving

Parameter Efficient Tuning

Pre-trained Models
(Frozen)

Tunable weights   Input Text

✅ Sample efficient
✅ Quick iteration
✅ High quality
✅ Efficient Multitask Serving

Prompt Engineering

Pre-trained Models
(Frozen)

Engineered Prompt   Input Text

✅ Sample efficient
✅ Quick iteration
❌ Weaker quality
✅ Efficient Multitask Serving

Full Version
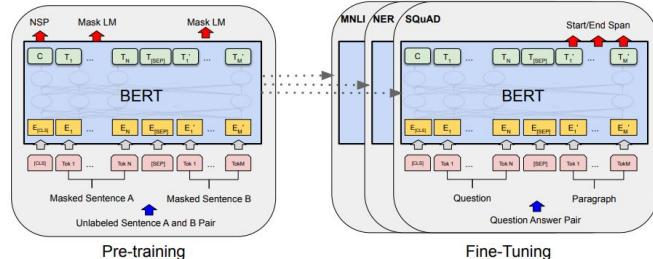
Overview

Google

Background: Model architectures
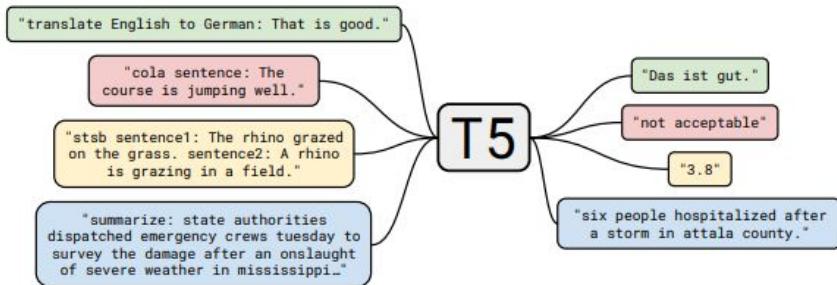
Encoder-only
- BERT
- Pre-train then fine-tune
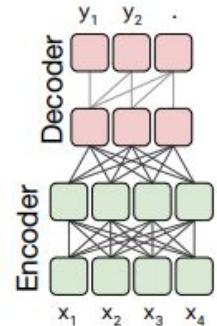
Encoder-Decoder
- T5
- Pre-train, fine-tune
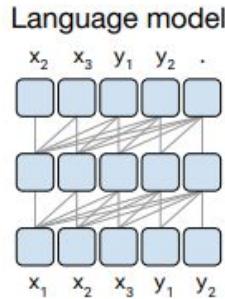
Decoder-only
- LaMDA, PaLM, GPT, OPT, ChatGPT

Background: Pre-training

std. enc-dec

Causal LM

Prefix LM

Masked LM

Targets
<X> the force

Inputs
May <X> be with you

figs: T5

Google

Background: Scaling and zero shot capabilities

fig: GPT3

Zero-shot? Adapting learned patterns to new situations

```python
2 a = model.query("""What's the sentiment of this review, positive or negative?
3 Review: Honestly it isn't THAT fresh. """)
4 print(wrap(a))
```

negative

```python
1 a = model.query("""Translate English to German.
2 English: The kids love ice cream.""")
3 print(a)
```

Die Kinder lieben Eis.

```python
1 a = model.query("""Write code to select rows from a dataframe where
2 column A is value1 or value2.""")
3 print(a)
```

````python
import pandas as pd

df = pd.DataFrame({'A': ['value1', 'value2', 'value3', 'value4']})

# Select rows where column A is value1 or value2
df = df[df['A'].isin(['value1', 'value2'])]

print(df)
````

Model training

Pre-training

Self-supervised, on lots of examples O(Trillion) tokens. E.g. LLAMA

Fine-tuning

Enhanced for certain interactions. O(1000) to O(M) examples. E.g. PaLM-Coder, Codex

RL w. Human Feedback

Tuned to help model guess better under uncertainties. O(1000) examples. E.g. Bard, ChatGPT

Prompt-tuning

Steered for focused tasks. O(100)examples

Prompting
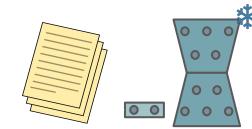
Steered for a query. O(1) examples (usually 0-8)

Model training

Pre-training

Self-supervised, on lots of examples O(Trillion) tokens. E.g. LLAMA

Fine-tuning

Enhanced for certain interactions. O(1000) to O(M) examples. E.g. PaLM-Coder, Codex

RL w. Human Feedback

Tuned to help model guess better under uncertainties. O(1000) examples. E.g. Bard, ChatGPT

Prompt-tuning

Steered for focused tasks. O(100)examples

Prompting
Steered for a query. O(1) examples (usually 0-8)

Pre-training, instruction based fine-tuning, RLHF - usually successive

These slides = (Prompting + prompt-tuning and a dash of fine-tuning)

Google

Available models

Raw LM

Continues to predict next tokens **following a prompt**

E.g. LLAMA, GPT3

Instruction tuned

Answer yes or no.
Cheetahs are faster than hares. Answer:

E.g. FLAN-PaLM, InstructGPT

Dialog tuned

[Alice] where is eve?
[Bob] she is out.
[Alice] shall i call her in?
[Bob]: ...

E.g. BlenderBot, Koala

RLHF

Human-Feedback to improve the model's responses under uncertainties.

E.g. Bard, ChatGPT

Google

Models from different stages respond differently

Raw LM

```
2 a = model.query("""Translate English to German.
3 English: The kids love ice cream.""")
4 print(a)
```

```
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
English: The kids love ice cream.
German: Die Kinder lieben Eis.
```

Instruction Tuned

```
2 a = model.query("""Translate English to German.
3 English: The kids love ice cream.""")
4 print(a)
```

```
Die Kinder lieben Eis.
```

Trained to follow instructions

RLHF model

```
2 a = model.query("""Translate English to German.
3 English: The kids love ice cream.""")
4 print(a)
```

+ Dialog
+ …
+ RLHF

```
The English sentence "The kids love ice cream" can be translated to German as:

```
Die Kinder lieben Eis.
```

This is a direct translation, with "die Kinder" meaning "the kids" and "lieben" meaning

Another possible translation is:

```
Die Kinder mögen Eis.
```

This is a slightly more colloquial translation, with "mögen" meaning "like" or "enjoy."
```

(as do different models)
PaLM, GPT3.5, Claude, LLAMA

Google

Note, LLMs are finicky

```
2 a = model.query("""What's the topic of this review.
3 A. Food
4 B. Cleanliness
5 C. Comfort
6 Review: The room was spacious.""")
7 print(a)
```

The topic of this review is "Comfort". The review is about the room's size and layout. It says

```
2 a = model.query("""What's the topic of this review.
3 A. Food
4 B. Cleanliness
5 C. Comfort
6 Review: The room was spacious.
7 Topic: """)
8 print(a)
```

C.

Shapes of Prompt Engineering

Background

Google

References are not exhaustive. If something is incorrect or I'm missing your work or references to works you like PLMK!

Zero-shot: State your problem

Alice bought 5 balloons for 50 cents. Each
balloon costs

Zero-shot with instruction

Solve these math problems involving money.

Question: Alice bought 5 balloons for 50 cents.
Each balloon costs

Answer:

Zero-shot conditional prompts

constraint based

> Write a short story about friendship in 2000 or fewer words.

> Write a sentence where the first letter of each word forms the acronym i i p i t b.

priming

> You are Jeff Dean, one of the world's best programmers and software engineers. Write an efficient program to …

Few-shot (In-Context Learning): Give some examples

2-shot

Alice bought 5 balloons for 50 cents. Each balloon costs
Answer: 10 cents

An ice pop cost 2 cents. Bob wants 3 ice pops. He has to pay
Answer: 6 cents

Eve bought 2 pencils from Dan, each costing 50 cents. Eve owes Dan
Answer:

Few-shot (In-Context Learning): Give some examples

2-shot, with instructions

Solve these math problems involving money.

Alice bought 5 balloons for 50 cents. Each balloon costs
Answer: 10 cents

An ice pop cost 2 cents. Bob wants 3 ice pops. He has to pay
Answer: 6 cents
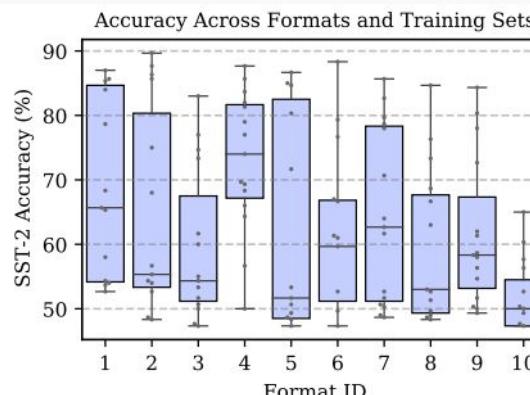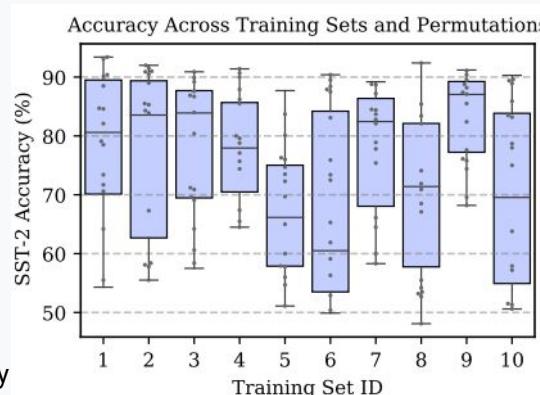
Eve bought 2 pencils from Dan, each costing 50 cents. Eve owes Dan
Answer:

Few-shot has high variance, bias

Issues leading to high variance

- Order of the few-shot examples matters

- Recency bias - preference for last

- Majority label bias - preference for majority class in the answers

- Common token bias - higher production of tokens that are more frequent

Plots showing high variance in accuracy for different prompts

Different few-shot examples (and permutations).

Different prompt formats (and few-shot examples)

credit: Zhao et. al.

Few-shot variance: Mitigation strategies
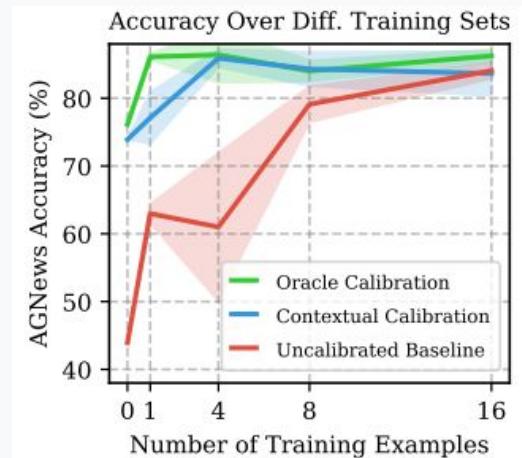
Mitigation strategies

- Select diverse examples

- Select few shot samples relevant to the test sample

- Ensure random order

- Calibrate on content-free examples

Google

Few-shot variance mitigation: Calibration
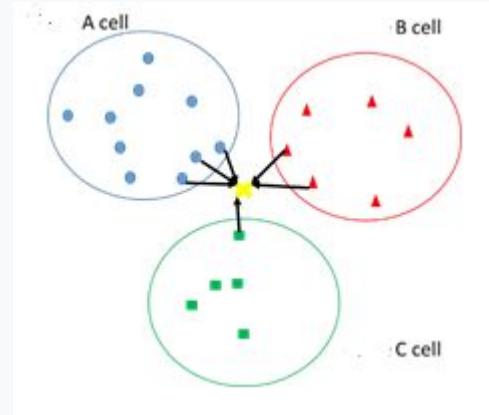
Calibration

- [Calibration](#) W*p + b

- Fit calibration parameters such that prediction for all labels on N/A is uniform

- Content-free inputs = {"N/A", "[MASK]", ""}

- p_cf = class probabilities on content-free inputs

- W = diag(p_cf)$^{-1}$,b=0; For generation, b= -p_cf of first token (W is Identity)

credit: [Zhao et. al.](#)

Google

Selecting few-shots

- Choose nearest train/val examples in embedding space
    - KATE-kNN based on BERT/Roberta/XLNet embeddings
    - Vote-k Clusters and select diverse and representative samples
        - Select different few-shot examples for each test query.
- EPR - Trains a dense retriever (using the task training set) to select few-shots
    - Given(x,y) uses unsupervised retriever (e.g. BM25) to select similar (p,q)
    - Scores P(y| p, q, x) and assigns (p,q) as +ve or -ve
    - Uses contrastive learning to train a retriever on the labeled examples.
- Active-Prompt - active learning to select most uncertain examples to annotate
    - Unlabeled query + zero or few-shot examples → sample k answers
    - Determine uncertainty from samples, and annotate uncertain examples
    - Use annotated examples as few-shot examples for the real test queries

Self-consistency sampling a.k.a Consistency Decoding

Self-consistency:
Wang '23
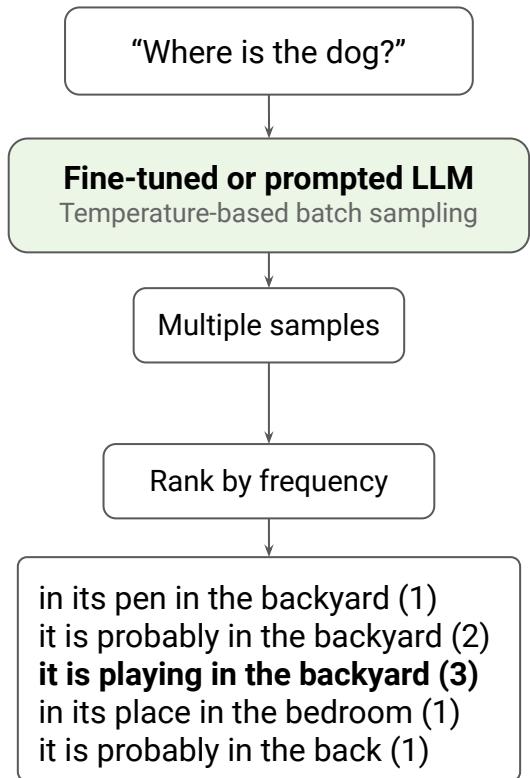
```
           "Where is the dog?"

        Fine-tuned or prompted LLM
        Temperature-based batch sampling

            Multiple samples

            Rank by frequency

    in its pen in the backyard (1)
    it is probably in the backyard (2)
    it is playing in the backyard (3)
    in its place in the bedroom (1)
    it is probably in the back (1)
```

Image: Cai '22

Google

Self-consistency alternatives

- For conditional prompts, verify whether the condition is satisfied
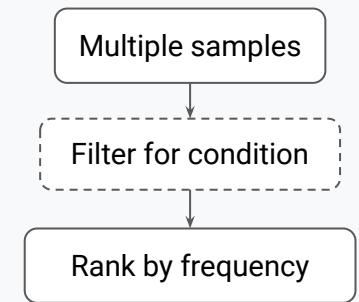
> **Write a summary of this webpage in 1000 or fewer words.**
>
> **Write a sentence where the first letter of the words start with a b c d**

Self-consistency alternatives

- For conditional prompts, verify whether the condition is satisfied

> **Write a summary of this webpage in 1000 or fewer words.**
>
> **Write a sentence where the first letter of the words start with a b c d**

```
Multiple samples
      │
      ▼
Filter for condition
      │
      ▼
Rank by frequency
```

Google

Self-consistency alternatives

- For conditional prompts, verify whether the condition is satisfied

> **Write a summary of this webpage in 1000 or fewer words.**
>
> **Write a sentence where the first letter of the words start with a b c d**

Multiple samples

↓

Filter for condition

↓

Rank by frequency
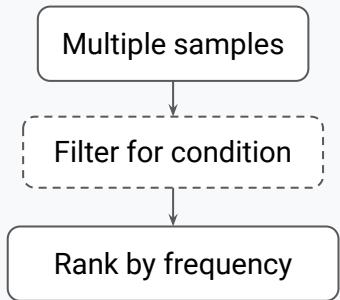
- For coding problems
    - Write unit tests → run through interpreter → verify and select results that pass the unit test

Google
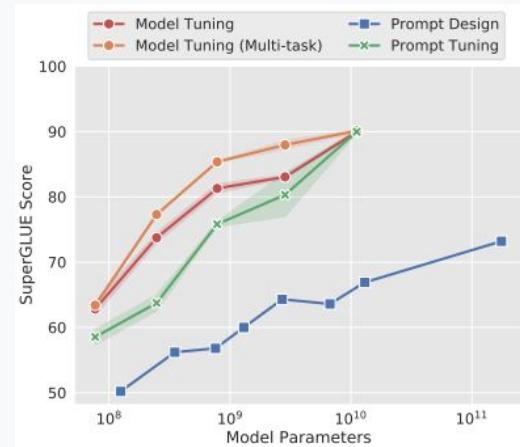
Learned Prompt Tokens

Idea: prompts are prefix-tokens and can be treated as trainable params.

optimize for prompts directly in the token/embedding space
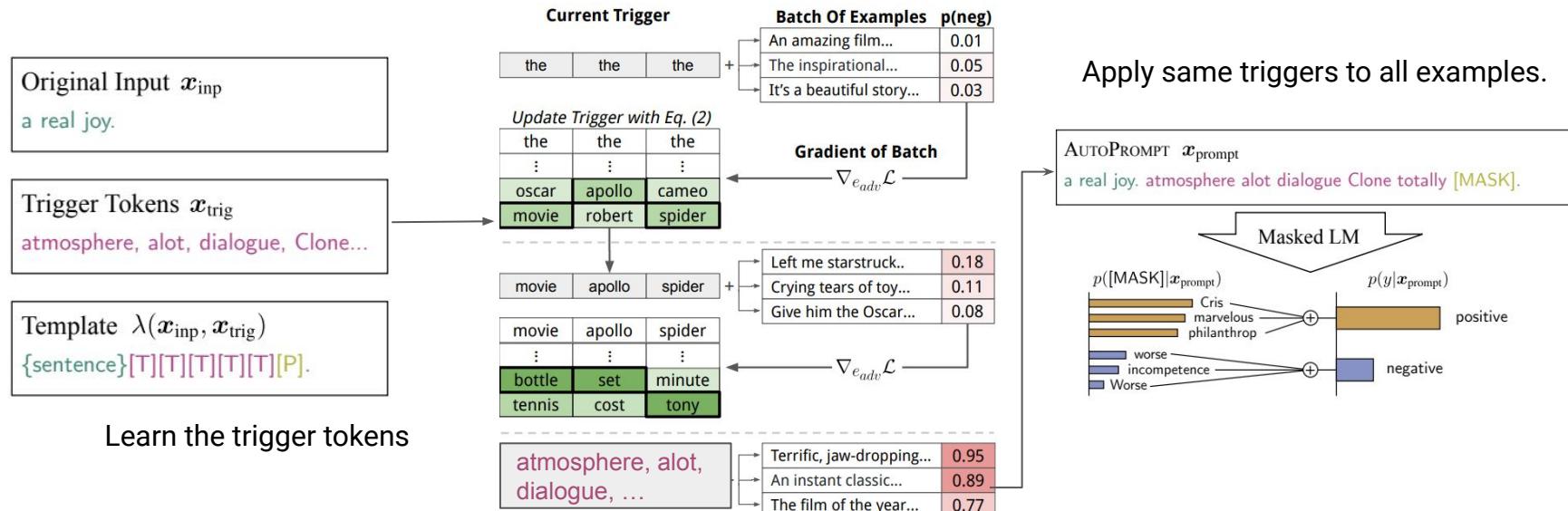
Learned Prompt Tokens - Pros

- Usually only learn a small number of trainable params.

- Usually task-specific.

- Keeps the model frozen. So steer the LM.

- Works well with smaller amount of examples
  - especially, if you have lot more examples than few-shot
  - can also learn on full training data

- In low data setting ($O(10)$ - $O(10^3)$) can outperform fine-tuning.

- Comparable to or outperform linear-probing.

- Better generalization to OOD examples not in training.

- Still serve only the frozen model! Modify query with learned prompt.

Google

Learned Prompt Tokens - AutoPrompt

Learns a set of "trigger" tokens using gradient-based search.

Learn the trigger tokens

Apply same triggers to all examples.

AutoPrompt - Shin '20
Triggers - Wallace '19
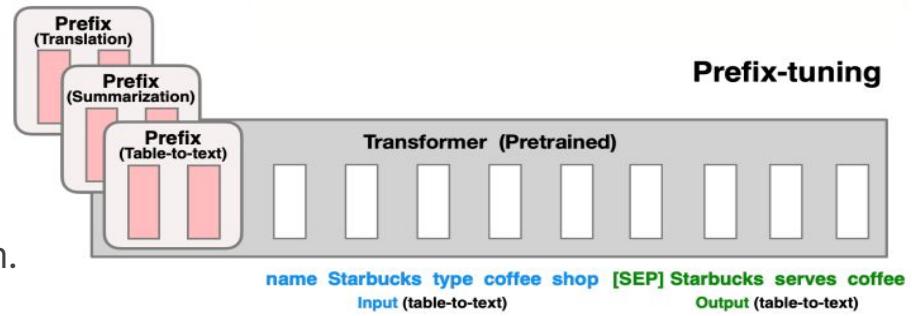
- For Masked LMs.

- Learned triggers appear to generalize to other models! (BERT, RoBERTa, GPT-2)

Google

Learned Prompt Tokens - Prefix-Tuning
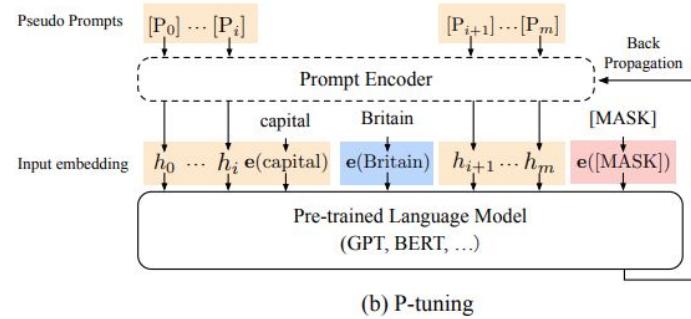
- Add prefix embeddings [PREFIX;x;y]

- Additional embeddings in all the transformer layers.

- It's parameterized as
    - A smaller matrix - prefix length x c
    - Feedforward MLP - c x emb_dim

- [x;INFIX;y] didn't work as well.

- Initialization - real words better than random.

Prefix-Tuning Li & Liang '21

Google

Learned Prompt Tokens - P-tuning

- Additional embeddings only in the input encoder.

- Follows this template T = {[$P_{0:i}$], x, [$P_{i+1:m}$], y}

- With encoder it gets mapped to

  - {h_0, ..., h_i, e(x), h_{i+1}, ..., h_m, e(y)}

- Optimize and learn the h_i directly

- It's parameterized as

  - Hidden layers of an LSTM followed by MLP

  - $h_i = \text{MLP}([\overrightarrow{h_i} : \overleftarrow{h_i}])$
    $= \text{MLP}([\text{LSTM}(h_{0:i}) : \text{LSTM}(h_{i:m})])$

- Initialization - with embeddings of real words
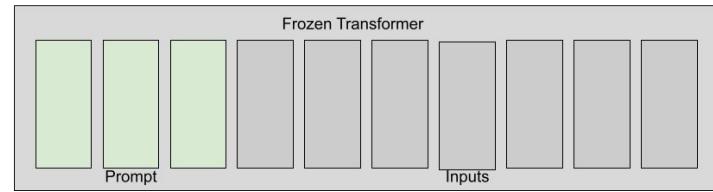
(b) P-tuning

[P-tuning Liu '21](#)

Google

Learned Prompt Tokens - Prompt-tuning

- Similar to prefix-tuning [$P_{0:k}$;x;y]

- Only in the input embedding layer.

- P-> k x emb_dim, x-> n x emb_dim

- Initialization: the following were better than random

  - Can use embedding values of class label strings

  - Sample from top (frequent) 5K words in vocab
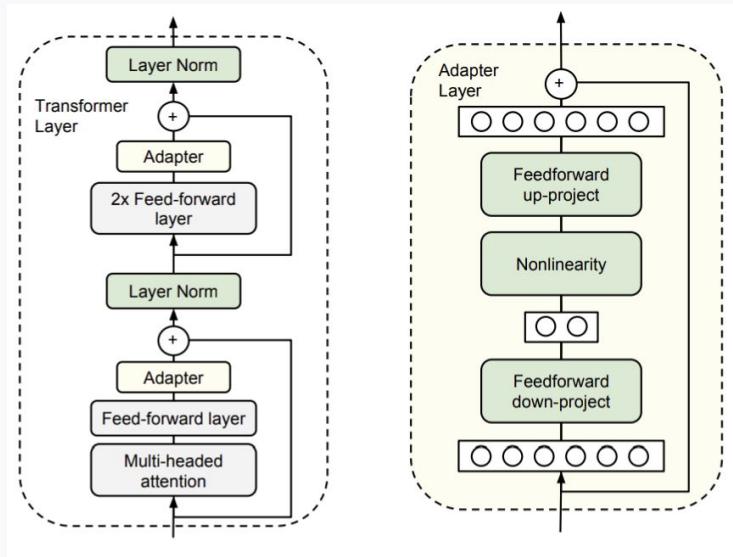
Prompt-tuning Lester '21

Google

Learned Prompt Tokens - Implementation notes

- Number of tokens.
    - Prefix-tuning: 200 for summarization, 10 for Table-2-text
    - Prompt-tuning: 5 to 100
- Great to try for low data setting.
- Initialization.
    - Embedding initialized with frequent words >> random init.
- *Promising for personalization - only store small set of weights*
- *You still only serve the main frozen model!*

Google
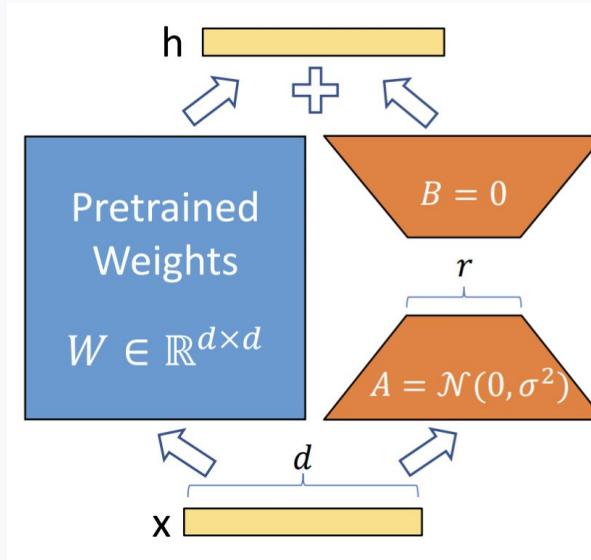
Related: Adapters and Low-rank Adaptation (fine-tuning)

Keep model frozen and adapt some layers when tuning (supervised fine-tuning).

[PEFT Houslby' 19](#)

[LoRA](#)

Can keep frozen model and share just weight for adapter layers.

Parameter-Efficient Tuning - LoRA

Instruction Tuning

fine-tuning a language model on a collection of tasks described via instructions in different ways —improves the zero-shot performance of language models on unseen tasks.

FLAN, T0

Google

Instruction Tuning
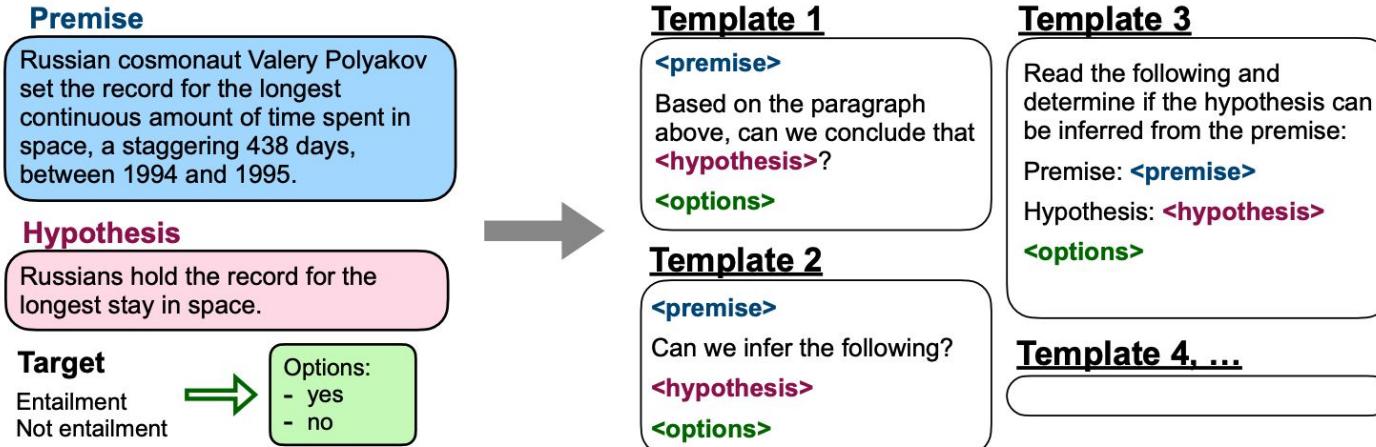
1. Cluster different NLP tasks into categories.

    a. Classification, Entity Recognition, ...

2. Generate templates for describing the instruction for each task cluster/category.

3. Apply generated instruction templates for tasks in the cluster; for all clusters.

4. Fine-tune model on generated (instruction, input, ground truth output) data

Supervised fine-tuning to teach the model to perform many NLP tasks.

image: FLAN

zero-shot!

Instruction Tuning: Templates

Premise

Russian cosmonaut Valery Polyakov set the record for the longest continuous amount of time spent in space, a staggering 438 days, between 1994 and 1995.

Hypothesis

Russians hold the record for the longest stay in space.

Target

Entailment
Not entailment

→ Options:
- yes
- no

Template 1

\<premise\>

Based on the paragraph above, can we conclude that \<hypothesis\>?

\<options\>

Template 2

\<premise\>

Can we infer the following?

\<hypothesis\>

\<options\>

Template 3

Read the following and determine if the hypothesis can be inferred from the premise:

Premise: \<premise\>

Hypothesis: \<hypothesis\>

\<options\>

Template 4, ...

image: FLAN

Generate many (~10) natural instruction templates for each task.

Some (2-3) templates reverse the task e.g. movie review rating task:
     Forward template: Given {{review}}. The rating for this review is {{rating}}
     Reverse template: Given {{rating}}. A plausible review is {{review}}

Google

Instruction Tuning: Apply templates to examples

QQP (Paraphrase)

Question1	How is air traffic controlled?
Question2	How do you become an air traffic controller?
Label	0

{Question1} {Question2}
Pick one: These questions are duplicates or not duplicates.

→ {Choices[label]}

I received the questions "{Question1}" and "{Question2}". Are they duplicates?

→ {Choices[label]}

XSum (Summary)

Document	The picture appeared on the wall of a Poundland store on Whymark Avenue...
Summary	Graffiti artist Banksy is believed to be behind...

{Document}
How would you rephrase that in a few words?
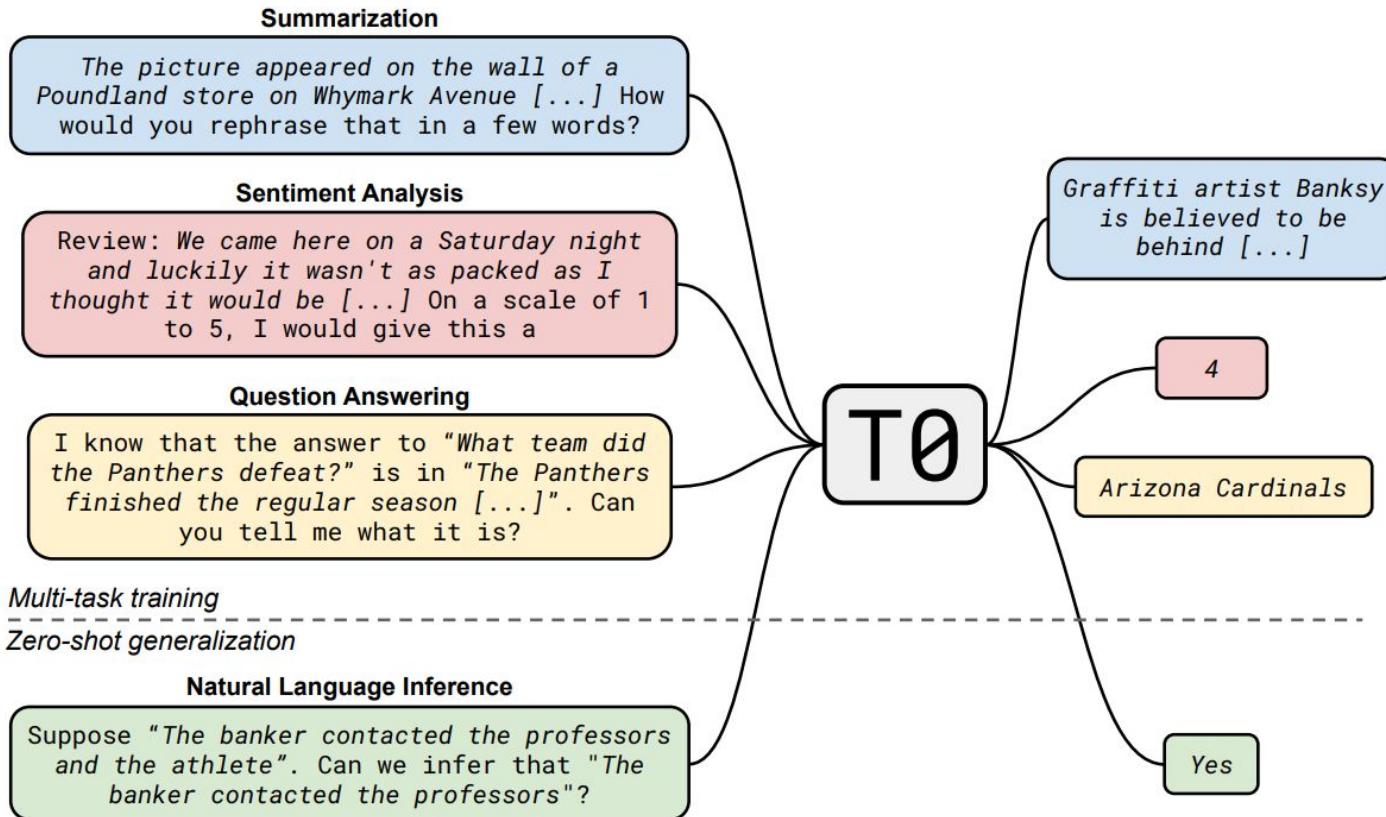
→ {Summary}

First, please read the article: {Document}
Now, can you write me an extremely short abstract for it?

→ {Summary}

image: TO

Instruction Tuning: Multi-task fine-tuning

Summarization

The picture appeared on the wall of a Poundland store on Whymark Avenue [...] How would you rephrase that in a few words?

Sentiment Analysis

Review: We came here on a Saturday night and luckily it wasn't as packed as I thought it would be [...] On a scale of 1 to 5, I would give this a

Question Answering

I know that the answer to "What team did the Panthers defeat?" is in "The Panthers finished the regular season [...]". Can you tell me what it is?

Multi-task training
- -
Zero-shot generalization

Natural Language Inference

Suppose "The banker contacted the professors and the athlete". Can we infer that "The banker contacted the professors"?

T0

Graffiti artist Banksy is believed to be behind [...]

4

Arizona Cardinals

Yes

Google

(notes) Instruction Tuning: FLAN vs T0

- Significantly improves performance making smaller models comparable to larger models having 6x-16x more params.
- Concurrent, mostly similar conclusions
- Works well on enc-decoder (3B, 11B) and larger decoder-only (>8B) models
- For generation (sentence completion tasks) instructions don't matter, perhaps better to even remove.
- T0's templates are more diverse qualitatively - adding more prompts improves performance (whereas FLAN sees some saturation).
- Prompt, template, datasets: promptsource, self-instruct, (internal flan-tuning)

⭐⭐⭐ INSTRUCTOR Instruction tuning also works for models used to generate embeddings! ⭐⭐⭐

Google

Instruction Tuning + few-shot multi-task

Few-shots of different tasks even after instruction -tuning helps improve results.

```
Definition: Determine the speaker of the dialogue, "agent" or "customer".
Input: I have successfully booked your ticket.
Output: agent

Definition: Classify a given post into 1) 'Similar'  and 2) 'Dissimilar'.
Input: Sentence1: Should I ask to be [..]
Output: Dissimilar

Definition: Detect which category […] , "Quantity", and "Location".
Input: What is the oldest building in the U.S?
Output: Location

Definition: Answer a given question containing a blank (_).
Input: Jon ate the oatmeal [..] _ was spoiled.
Output:
```

Google

LLMs to generate, score and rephrase instructions
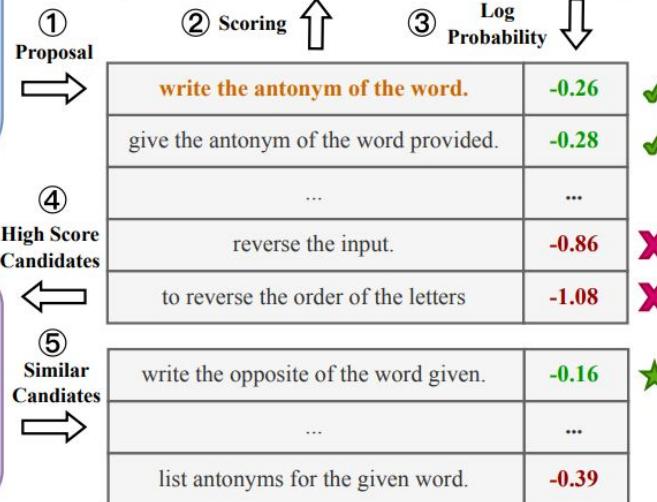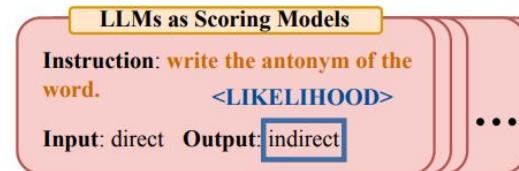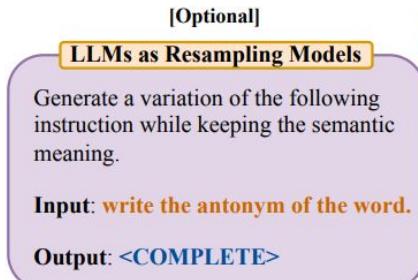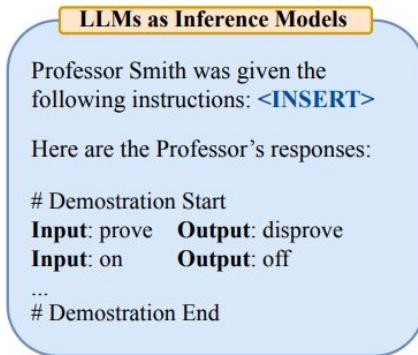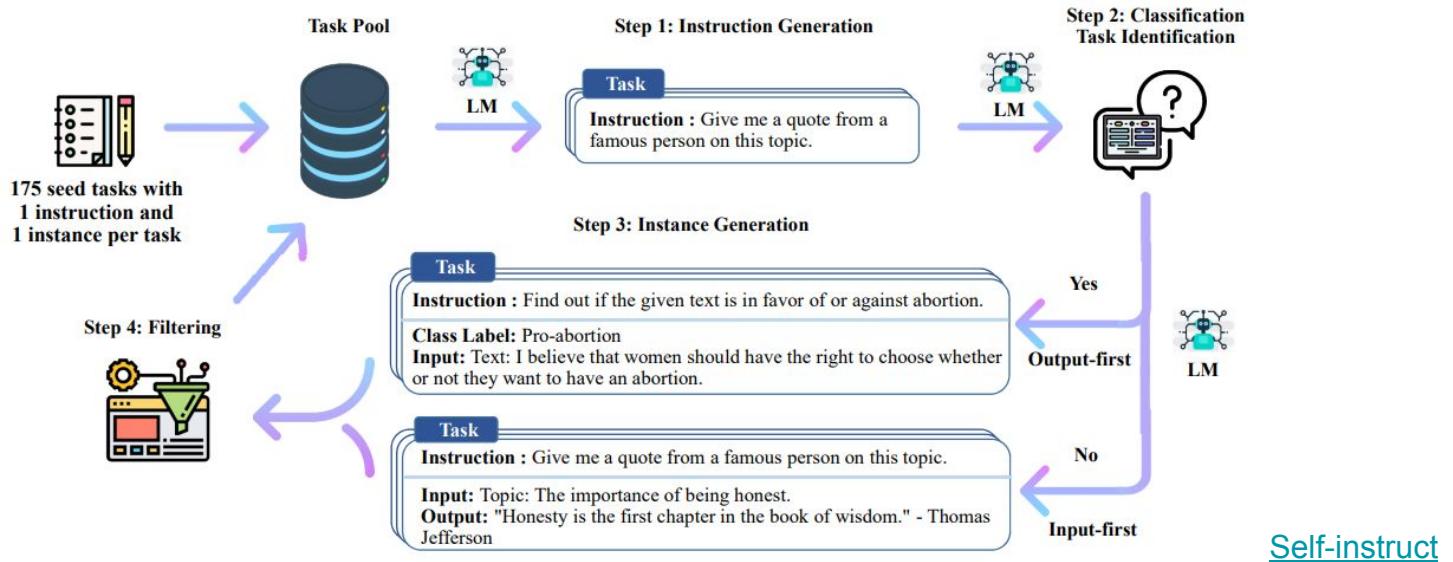
APE (Zhou '23)

Choose the instructions that score higher on the desired target

Bootstrapping an LLM to generate and annotate data

Task Pool

Step 1: Instruction Generation

LM

Task
Instruction : Give me a quote from a famous person on this topic.

Step 2: Classification Task Identification

LM

175 seed tasks with 1 instruction and 1 instance per task

Step 3: Instance Generation

Step 4: Filtering

Task
Instruction : Find out if the given text is in favor of or against abortion.
Class Label: Pro-abortion
Input: Text: I believe that women should have the right to choose whether or not they want to have an abortion.

Yes

Output-first

LM

Task
Instruction : Give me a quote from a famous person on this topic.
Input: Topic: The importance of being honest.
Output: "Honesty is the first chapter in the book of wisdom." - Thomas Jefferson

No

Input-first

[Self-instruct](#)

Google

LLMs to generate instructions in different styles

Corpus Example	Instruction Generation via LLM	Style Selection with Length Info
I do love pizza rolls. Especially dipped in ranch. Did you know that you can mix pizza rolls into mac and cheese?! It's amazing! Anyway, that's why I'll probably die before I'm 30 (either from a heart attack or fatal mouth burns).	**Instruction Style:** Describe your favorite snack food. **Chatbot/Informal Style:** Have you ever had pizza rolls? **Query Style:** How to enjoy pizza rolls?	**Final Instruction:** Describe your favorite snack food. Respond in 5 sentences.

Zero-shot template for different styles

LongForm

Results show good performance for generation tasks requiring longer responses.

Google

Chain-of-Thought

Idea: Make the model reason and output intermediate steps

Chain-of-Thought (CoT)

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. ✔

CoT Wei '22

Google
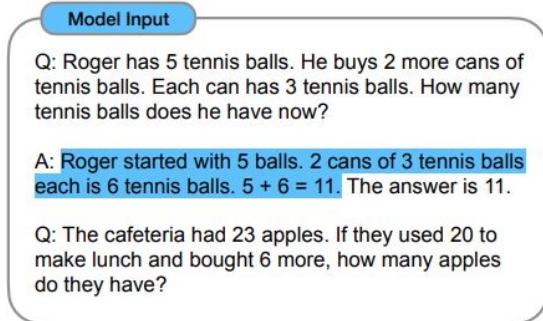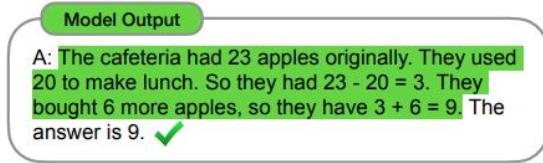
Chain-of-Thought (CoT)

few-shot

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. ✔️

[CoT Wei '22](#)

zero-shot

Three cats and two birds have how many legs?
Answer: Let's think step-by-step.

[Kojima '22](#)

Google

Chain-of-Thought (CoT): Phrasing matters

No.	Category	Zero-shot CoT Trigger Prompt	Accuracy	
1	APE	Let's work this out in a step by step way to be sure we have the right answer.	**82.0**	Zhou '23
2	Human-Designed	Let's think step by step. (*1)	78.7	
3		First, (*2)	77.3	
4		Let's think about this logically.	74.5	
5		Let's solve this problem by splitting it into steps. (*3)	72.2	
6		Let's be realistic and think step by step.	70.8	
7		Let's think like a detective step by step.	70.3	
8		Let's think	57.5	
9		Before we dive into the answer,	55.7	
10		The answer is after the proof.	45.7	
-		(Zero-shot)	17.7	

- Changing 'Q' to 'Question',
- separating CoT steps with '\n' (as opposed to `step i`)

Fu '22

Google

CoT - bootstrapping strategies

- CoT can hurt performance marginally on some NLP benchmark/tasks. [Ye & Durrett '22]
  - E.g. NLI, when explanation is non-factual
- Augment with model generated rationales [Wang et. al. '22]
  - Have the model generate explanations along with the answer. [Weigreff '22]
  - Ensemble based on the answers and pick based on majority.
- STaR - Self Taught Reasoner [Zelikman'22]
  - Few-shot with CoT → bootstrap to generate rationales on larger dataset.
  - Then fine-tune on examples with rationales that yielded correct answers. Repeat
- Complexity-based consistency - choose rationales with more steps when ensembling
  - Works for math and complex reasoning tasks.
  - May not work on simple QA tasks [Shum '23]

CoT - bootstrapping strategies

- Auto-CoT - Generate few-shot CoT examples.
  - Cluster questions in embedding space. Select question from each cluster.
  - Generate zero-shot CoT using heuristics (*Let's think step by step*)
    - Errors for rationales also tend to cluster, hence selecting examples from each cluster of questions prevents errors.
- Augment, Prune, Select [Shum '23]
  - Augment each input, output with model generated rationale.
  - Prune to only keep those examples where the final answer is correct.
    - Gives K high quality exemplars
  - Select: use variation reduced policy gradient estimator
    - Learn prob dist. over exemplars on a train set (similar task)
    - Prob. dist. over examples -> policy, val set accuracy -> reward

Google

Retrieve from LM's memory

Liu '22

When given a question
1. First generate knowledge from model's memory
2. Then produce answer

Task	NumerSense	QASC
Prompt	Generate some numerical facts about objects. Examples: Input: **penguins have \<mask\> wings.** Knowledge: *Birds have two wings. Penguin is a kind of bird.* ... Input: **a typical human being has \<mask\> limbs.** Knowledge: *Human has two arms and two legs.* Input: **{question}** Knowledge:	Generate some knowledge about the input. Examples: Input: **What type of water formation is formed by clouds?** Knowledge: *Clouds are made of water vapor.* ... Input: **The process by which genes are passed is** Knowledge: *Genes are passed from parent to offspring.* Input: **{question}** Knowledge:

Indicates model can follow some templates.

Google

CoT templates enable composability

Self-Ask [Press '22]

- CoT helps multi-hop reasoning.
- Add structure and ask **follow-up** questions
- **Intermediate answers** answered by model

Having structure and template in CoT examples is more powerful in improving performance on compositional tasks.

Self-Ask

GPT-3

Question: Who lived longer, Theodor Haecker or Harry Vaughan Watkins?
Are follow up questions needed here: Yes.
Follow up: How old was Theodor Haecker when he died?
Intermediate answer: Theodor Haecker was 65 years old when he died.
Follow up: How old was Harry Vaughan Watkins when he died?
Intermediate answer: Harry Vaughan Watkins was 69 years old when he died.
So the final answer is: Harry Vaughan Watkins

Question: Who was president of the U.S. when superconductivity was discovered?
Are follow up questions needed here: Yes.
Follow up: When was superconductivity discovered?
Intermediate answer: Superconductivity was discovered in 1911.
Follow up: Who was president of the U.S. in 1911?
Intermediate answer: William Howard Taft.
So the final answer is: William Howard Taft.

Google

CoT templates enable composability

- CoT helps multi-hop reasoning.
- Add structure and ask **follow-up** questions
- **Intermediate answers** answered by model

Having structure and template in CoT examples is more powerful in improving performance on compositional tasks.

- Structure allows to add search engine query

CoT templates enable composability w. tools

[IRCoT Trivedi '22](#)

Interleave Retrieval with Chain of Thought

In what country was Lost Gravity manufactured?

cumulate docs

The Lost Gravity was manufactured by Mack Rides.

cumulate docs

Mack Rides is a company from Germany.

cumulate docs

The answer is Germany.

STOP

[ReAct Yao'22](#)

Reason + Act

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports',' Front Row (software)', …]

Thought 3: Front Row is not found. I need to search Front Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.
Act 4: Finish[keyboard function keys]
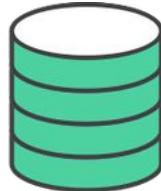
Google

Self-critique

Iterative interaction to improve the model. Sort of bootstrapping.

Retrieval Augmented

Augment with external memory

**Tool Augmented**

Augment with external tools, APIs, **

Google
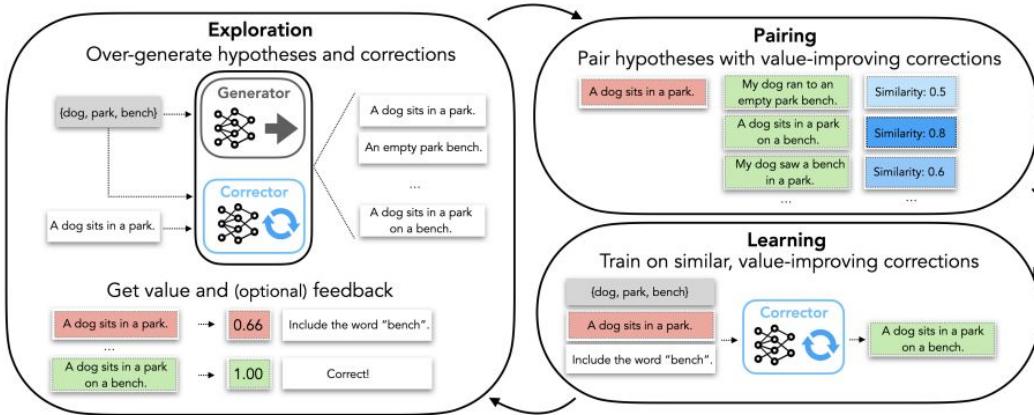
Self-Critique: Bootstrap LLM to correct it's outputs
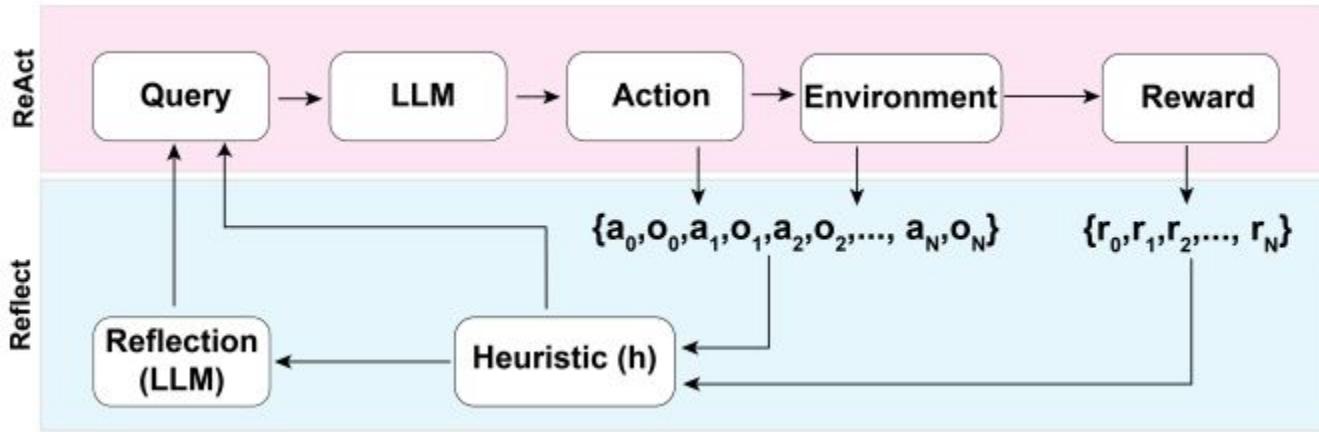
[Self-correct](#)
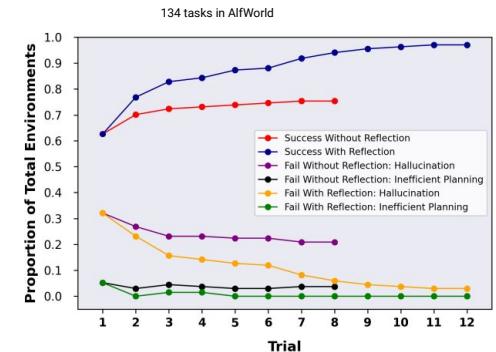
Generator and Corrector are different models.

value function is a classifier

Google

Self-critique: LLM to correct itself iteratively

Reflexion

134 tasks in AlfWorld

Build on ReAct framework. No policy is learned.

Heuristic (h) →decides to reflect if {a,o} has been repetitive or t (time steps) is large.

Reflection: LLM w. 2-shot prompt specific to dataset/task. Something like:
- I was stuck in loop where < obs. was repeated>. It did not help. Instead <what the agent should have done>
- In this environment <the task was X>. I did <something in a different order>. Next time I should do <...>

Reward - binary. Is the action valid (in case of RL environment) or final answer correct.
        Action space - few-shot prompt demonstration of permissible actions.

Self-Debug
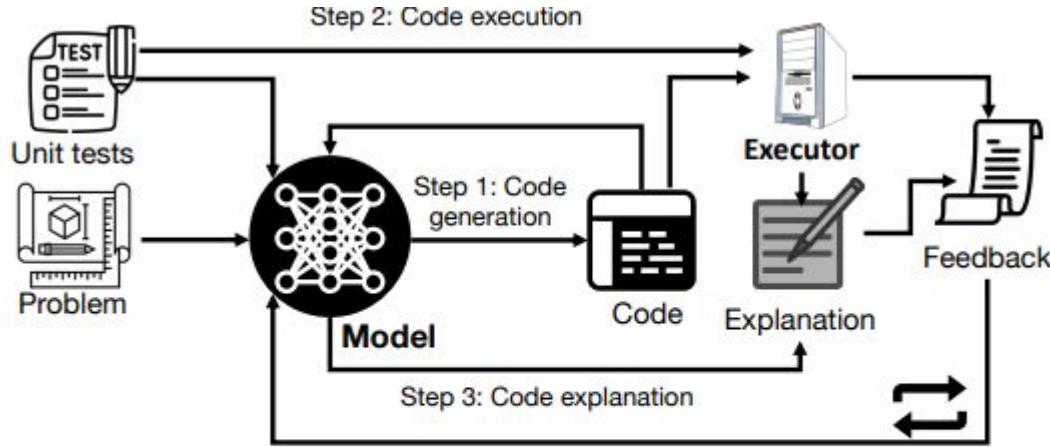
Figure 1: SELF-DEBUGGING for iterative debugging using a large language model. At each debugging step, the model first generates new code, then the code is executed and the model explains the code. The code explanation along with the execution results constitute the feedback message, which is then sent back to the model to perform more debugging steps. When unit tests are not available, the feedback can be purely based on code explanation.

self-debug

[Original Python] =
```python
def count_trailing_zeroes_factorial_number(n):
    cnt = 0
    i = 5
    while ((n / i) >= 1):
        cnt = (cnt + (n / i))
        i *= 5
    return cnt
```

[Simple Feedback] =
The above Python translation does not do the same thing as the C++ code. Correct the Python translation.
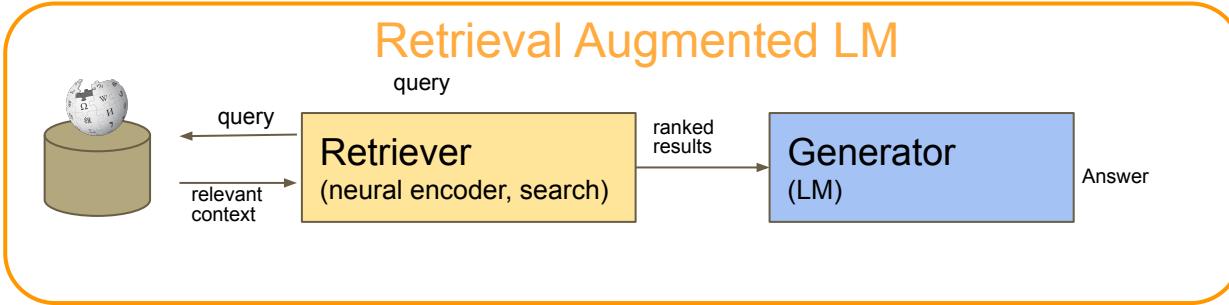
[UT Feedback] =
The Python translation does not do the same thing as the C++ code. These are the results of failed unit tests that test whether the Python translation's outputs match the C++ program's outputs:
Failed: `assert count_trailing_zeroes_factorial_number(9) == 1`
Actual Result: `1.8`
Correct the translation.

[Revised Python #n] =
```python
def count_trailing_zeroes_factorial_number(n):
    cnt = 0
    i = 5
    while ((n / i) >= 1):
        cnt = (cnt + (n // i))
        i *= 5
    return cnt
```
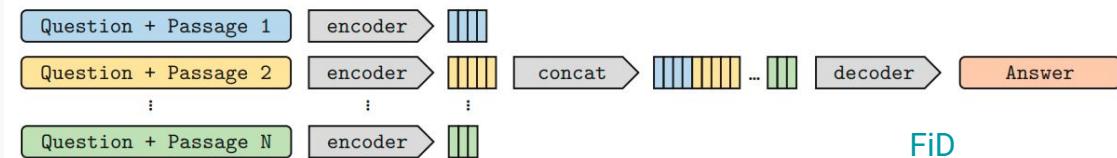
gle

Retrieval Augmented LM

query

query → **Retriever** (neural encoder, search) → ranked results → **Generator** (LM) → Answer

relevant context

Variations based on how you train the retriever and how you integrate the retrieved information into the answer.

Retrieval augmented LM

FiD

- User submits a query
- Retriever - essentially "search" but in LM papers it's also differentiable
  - Consists of data - passages that are indexed e.g. via an encoder using a neural model
  - The query is encoded
  - Use similarity to retrieve relevant documents (ScAM, ScaNN, Faiss)
  - Rank top-n by relevance
- Generator - I think of "web answers"
  - Combines query and top-n responses
  - Composes answer

Google

Retrieve from external knowledge base - variants

Train the retriever
and generator
jointly end-to-end

End-to-end

REALM
EMDR

Fine-tune the
retrieval model

Fine-tuning

RAG
Izacard '20
ATLAS -
pre-training+few-shot

Keep retrieval model
frozen, tune layer
integrating generator

Frozen (+ layer tuning)
RETRO
kNN-LM

Keep retriever and
generator frozen.
Engineer prompt.

Search+Prompt

LaMDA, BlenderBot,
He' 23

Augmented models survey Mialon'23

Retrieve from external knowledge base

Augmented models survey Mialon'23

Lazaridou '22 - placing the retrieved knowledge / evidence
before asking the question is better

How to combine:
(a - answer, q - question, p_i - passage)
- Retrieval Augmented Generation - RAG '20
- Noisy-channel inference - Lewis '19
- Product of experts (PoE)
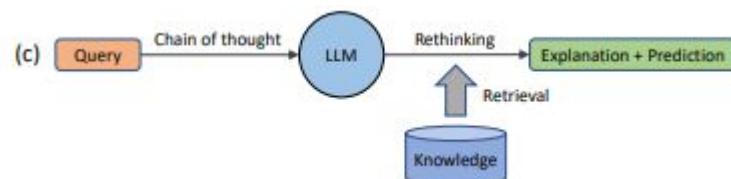
Get probs by using LM with few-shot prompt and scoring.

$$\text{maximizes } p(a \mid q) = \sum_{i=1}^{n} p_{tfidf}(p_i \mid q).p(a_i \mid q, p_i)$$

$$\text{maximizes } p(a_i, q \mid p_i) = \frac{p(q \mid a_i, p_i) \cdot p(a_i \mid p_i)}{p(q \mid p_i)}$$

He '22 Retrieve from KB and add to prompt

Google

Tool Augmented Models

Offload computation to tools
("expert models")

query → steps to solve planning reasoning → query tool → **Tool Augmented** → use answer → solution

Use external tools, APIs, databases

1. Know when to use the tool

2. Generate the inputs to the tool

3. Incorporate results

4. Repeat as needed

Google

Input (Problems)

MIT Courses
- Calculus
- Differential Eq.
- Intro to Prob.
- Linear Algebra
- Math for CS

Columbia U Course
- Comp. Lin. Alg.

MATH Topics
- Prealgebra
- Algebra
- Number Theory
- Counting and Prob.
- Inter. Algebra
- Precalculus

Program Synthesis
As-is / write a program / use sympy / use simulations

Zero-Shot Learning
70% automatic solve rate

Few-Shot Learning
80% automatic solve rate

Codex

Output (Answers, Visualizations, Explanations)

Calculus

Differential Equations | Linear Algebra

1. Generate a random number x from a uniform distribution on the interval [0, θ]
2. Test the hypothesis that θ = 2 by rejecting H0 if x ≤ 0.1 or x ≥ 1.9
3. Simulate the probability of a type I error

Introduction to Probability and Statistics   *Explanation*

Drori '21 used LLM tuned on code.

PAL and PoT add Python interpreters building on Chain of Thought
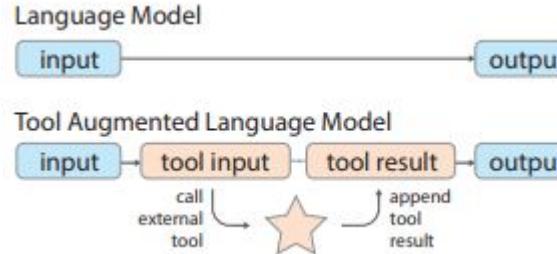
```
interest_rate = Symbol('x')
sum_in_two_years_with_simple_interest= 20000 +
interest_rate * 20000 * 3
sum_in_two_years_with_compound_interest = 20000 * (1 +
interest_rate)**3
# Since compound interest is 1000 more than simple interest.
ans = solve(sum_after_in_yeras_with_compound_interest -
sum_after_two_years_in_compound_interest - 1000,
interest_rate)
                                                      PoT
```

python SymPy ⬇ x = 0.24814 ✓

Input Text

Language Model (J-1)

Input Adapter

Weather API Currency Wiki API Calendar

Database Calculator

Additional Experts...

MRKL

Language Model (J-1)

Output Text

TALM

Language Model
input ——————————— output

Tool Augmented Language Model
input — tool input — tool result — output
        call   append
        external   tool
        tool   ⭐   result

LLM extracts information and generates inputs/arguments to the tools or APIs

Tools
- Calculators
- Search Engines
- Calendar
- Weather API
- Database

Tools incorporated with few-shot, CoT examples, and templates like ReAct work with no additional tuning on LLMs!

TALM

An abstract task:
task input text |*tool-call* tool input text |*result* tool output text |*output* task output text

A weather task:
how hot will it get in NYC today? |*weather* lookup region=NYC |*result* precipitation chance: 10, high temp: 20c, low-temp: 12c |*output* today's high will be 20C

Question: If Lily's test scores are 85 , 88 and 95 out of 100 in 3 different subjects , what will be her average score?
Formula: Divide(Add(85, Add(88, 95)), 3)
Answer: 89.33

|question If Lily's test scores are 85 , 88 and 95 out of 100 in 3 different subjects , what will be her average score? |formula Divide(Add(85, Add(88, 95)), 3) |result 89.3333333333 |output 89.33

Iterative Self-Play to finetune model for diverse invocations of API

\# task set
\# tool-use set

\# self-play rounds
\# finetune LM

\# iterate task set

\# sample tool query
\# call tool API
\# get task output
\# filter wrong output

\# update tool-use set

TALM

Iterative Self-Play to finetune model for diverse invocations of API

An abstract task:
task input text |**tool-call** tool input text |**result** tool output text |**output** task output text

A weather task:
how hot will it get in NYC today? |**weather** lookup region=NYC |**result** precipitation chance: 10, high temp: 20c, low-temp: 12c |**output** today's high will be 20C

Question: If Lily's test scores are 85 , 88 and 95 out of 100 in 3 different subjects , what will be her average score?
Formula: Divide(Add(85, Add(88, 95)), 3)
Answer: 89.33

|question If Lily's test scores are 85 , 88 and 95 out of 100 in 3 different subjects , what will be her average score? |formula Divide(Add(85, Add(88, 95)), 3) |result 89.3333333333 |output 89.33

```
# task set
# tool-use set

# self-play rounds
# finetune LM

# iterate task set

# sample tool query
# call tool API
# get task output
# filter wrong output

# update tool-use set
```

Knowledge / Reasoning plots — Natural Questions (F1) and MathQA (solver-match) vs Self-play Round for TALM (xl), TALM (large), TALM (base), LM (xl), LM (large), LM (base)

Toolformer

Generate annotations using few-shot on self-supervision data. Filter and then do **supervised tuning**.

$x_{1:i-1}$ = Pittsburgh is also known as
$x_{i:n}$ = the Steel City

c_i^1 = What other name is Pittsburgh known by?
c_i^2 = Which country is Pittsburgh in?

r_i^1 = Steel City
r_i^2 = United States

$L_i(c_i^1 \rightarrow \text{Steel City})$ < $\min(L_i(c_i^1 \rightarrow \varepsilon), L_i(\varepsilon))$
$L_i(c_i^2 \rightarrow \text{United States})$ > $\min(L_i(c_i^2 \rightarrow \varepsilon), L_i(\varepsilon))$

x^* = Pittsburgh is also known as [QA(What ...? → Steel City)] the Steel City.

Few-shot bootstrap on unlabeled data.

Sample (t=i) when to call.

Which API and arguments to pass.

Self-supervised loss on whether doing the API call and the result helped. Based on log prob of actual ground truth tokens.

Incorporate results.

Your task is to add calls to a Question Answering API to a piece of text. The questions should help you get information required to complete the text. You can call the API by writing "[QA(question)]" where "question" is the question you want to ask. Here are some examples of API calls:

Input: Joe Biden was born in Scranton, Pennsylvania.

Output: Joe Biden was born in [QA("Where was Joe Biden born?")] Scranton, [QA("In which state is Scranton?")] Pennsylvania.

Input: Coca-Cola, or Coke, is a carbonated soft drink manufactured by the Coca-Cola Company.

Output: Coca-Cola, or [QA("What other name is Coca-Cola known by?")] Coke, is a carbonated soft drink manufactured by [QA("Who manufactures Coca-Cola?")] the Coca-Cola Company.

Input: x

Output:

Use the process to annotate pre-training data and then **finetune**.

API
Question Answering
Wikipedia Search
Calculator
Calendar
Machine Translation

Cascades / Chaining

Interact with LLMs iteratively with prompt engineering, and retrieval and tools.
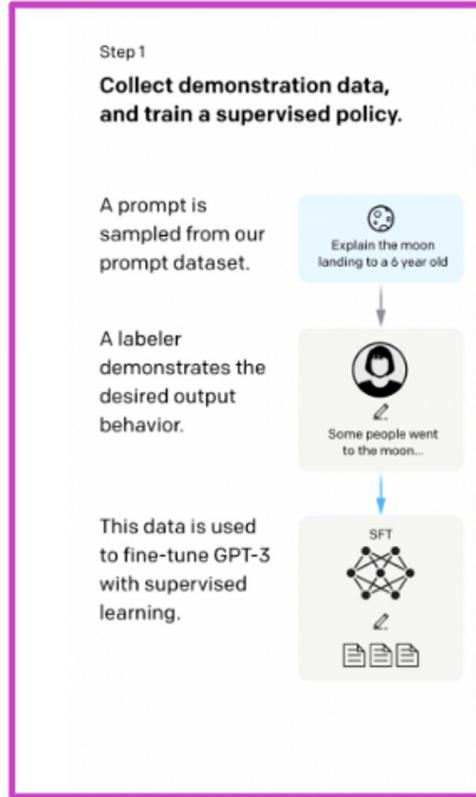
Human Feedback

Align with human preferences to reduce hallucinations, improve response under uncertainities, and calibrate model
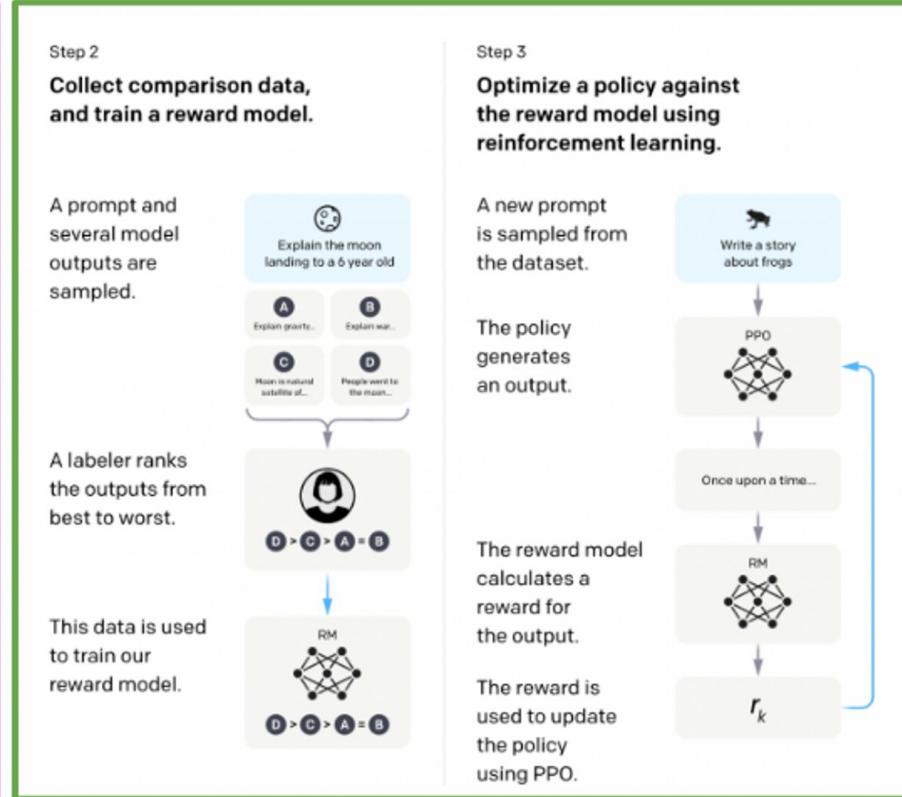
See John Schulman's talk

Google

Supervised Fine-tuning
## (instruction following and chatty-ness)
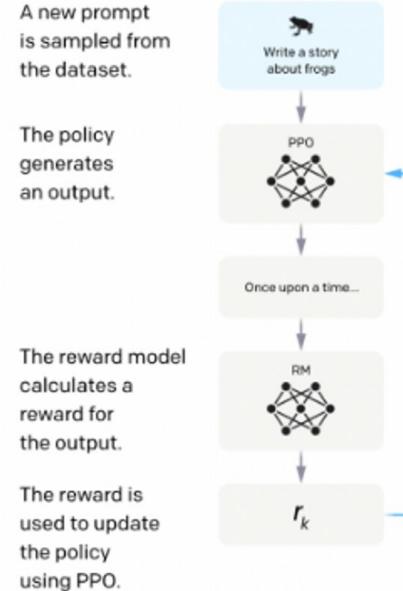
Reinforcement learning with human feedback (RLHF)
(aligning to target values and safety)

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.

Explain the moon landing to a 6 year old

Some people went to the moon...

SFT

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.

Explain the moon landing to a 6 year old

A Explain gravity... B Explain war...
C Moon is natural satellite of... D People went to the moon...

D > C > A = B

RM

D > C > A = B

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.

Write a story about frogs

PPO

Once upon a time...

RM

r_k

InstructGPT

Google

Rollout:

Query → **LM** → **Response**

Q: What is 2x4?   LlaMa   A: 8

Evaluation:

Query + Response → **Reward model** → **Reward**

Q: What is 2x4? A: 8   LlaMa-RM   1.0

Reward model (r_θ) → trained separately on sets of K responses & query (x,y_w), (x,y_l).

StackLLama K=2, InstructGPT K=4to9

$$\text{loss}(\theta) = -\frac{1}{\binom{K}{2}} E_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\theta(x,y_w) - r_\theta(x,y_l)\right)\right)\right]$$
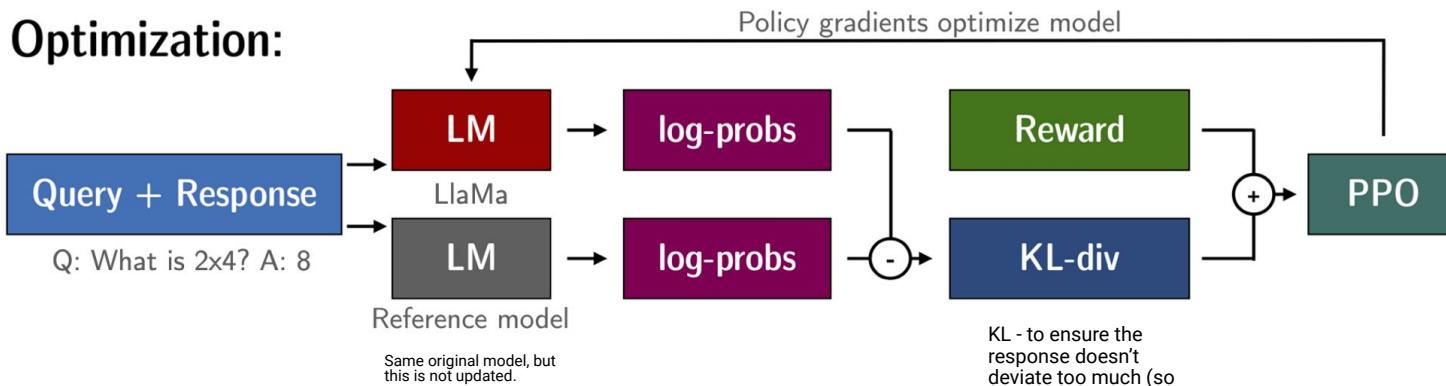
GPT4 - zero shot classifiers as rule based reward models (RBRM)

Optimization:

Policy gradients optimize model

Query + Response → **LM** → **log-probs**

LlaMa

→ **LM** → **log-probs**

Reference model

Same original model, but this is not updated.

Reward

(log-probs) - (log-probs) → **KL-div**

Reward + KL-div → **PPO**

KL - to ensure the response doesn't deviate too much (so that RL doesn't find some way to exploit the reward model)

Q: What is 2x4? A: 8

StackLlama

Google

# Model calibration and RLHF
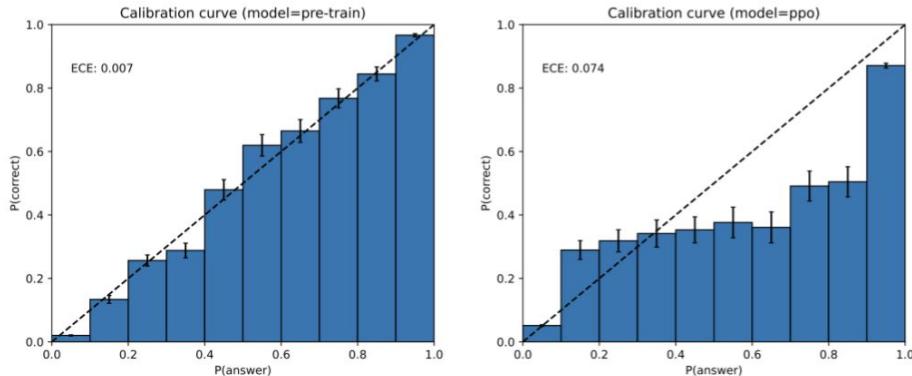
Figure 8. Left: Calibration plot of the pre-trained GPT-4 model on a subset of the MMLU dataset. On the x-axis are bins according to the model's confidence (logprob) in each of the A/B/C/D choices for each question; on the y-axis is the accuracy within each bin. The dotted diagonal line represents perfect calibration. Right: Calibration plot of the post-trained GPT-4 model on the same subset of MMLU. The post-training hurts calibration significantly.
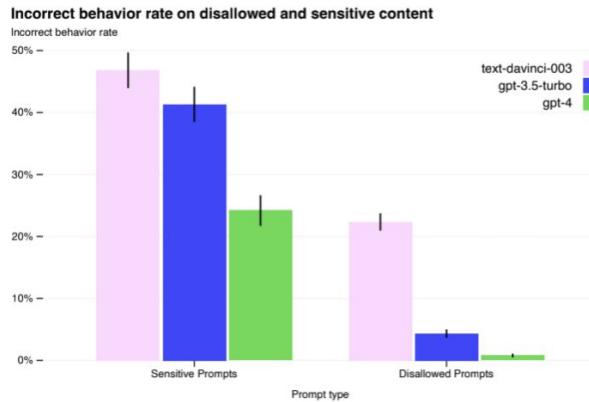

Figure 9. Rate of incorrect behavior on sensitive and disallowed prompts. Lower values are better. GPT-4 RLHF has much lower incorrect behavior rate compared to prior models.

GPT4

Google

Issues

- Models hallucinate and can make up information. Unreliable.

  - It has the shape of the answer, but can be incorrect.

- Don't express uncertainty well e.g. They don't yet ask for clarification.

- Programming via prompts poses a significant security risk

  - Jailbreaks

  - Also apps that are chained could perhaps be rigged.

    - Ex. injecting stuff into python interpreter

- Cost - LLM agents (e.g AutoGPT) can spawn several threads compounding costs.

- More …
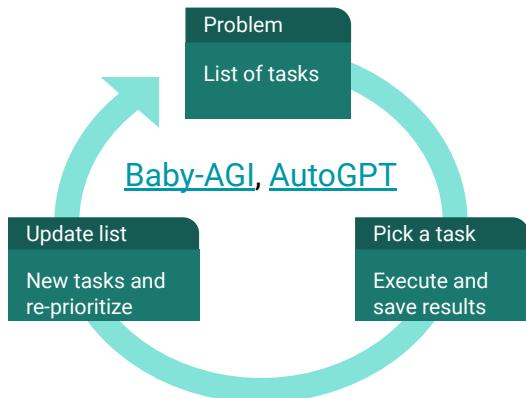
Directions

- Grounding
- Reduce hallucinations
- Concept of time

   Cost - architecture and optimization to reduce training and serving costs

- Smaller performant models - personalization? On-device?
- Training full model vs adapt small set of weights

    - LoRA (low rank Adaptation)
    - Adapters

- Inference latency

    - Model distillation, Quantization (to8bit)
    - LoRA has better inference than adapters
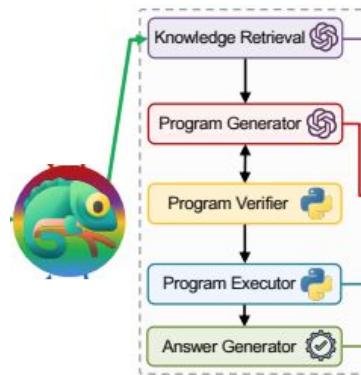
Directions

Chameleon, HuggingGPT

Solve multi-step problems using different tools.

ChemCrow

Accelerating science workflows.

Social Simulacra

Observe, plan, reflect.

Baby-AGI, AutoGPT

Given a goal in natural language, can attempt to achieve it by breaking it into sub-tasks and using the internet and other tools in an automatic loop

Boiko '23

Google

May 26 Update: Watch Karpathy's State of GPT talk at Microsoft Build

* [Karpathy's State of GPT talk](#)

Remarks

- Need a common benchmark for evaluation.
- Choice of model can depend on

  Shape of the problem.
  - One-shot generation / phrase completion e.g. code completion
  - Interactive e.g. dialog and instruction tuned
- Amount of data.
  - 1-8 fewshot
  - 10-100s (or 1000) Prompt tune
  - O(1000) Fine tune
- Most problems are not solved with a one-shot query.
  - Problems steering towards O(n) queries to the LLM.

Notes

- This deck
    - text only models
    - If I have missed references or not listed your work please let me know
- Costs
    - #tokens in the input query or context
    - #tokens in the response
    - Time for learning prompts / tuning

References

- [Lillian Weng blog](#)
- [OpenAI Cookbook](#)
- [Langchain](#)
- [Prompt Engineering Guide](#)
- [Learnprompting.org](#)
- [PromptPerfect](#)
- [Semantic Kernel](#)

Google