香港城市大學(東莞)
City University of Hong Kong (Dongguan)

Celebrating the Establishment in Year 2024

Lecture 8: LLM Prompting and Alignment

CS6493 Natural Language Processing

Instructor: Linqi Song

Outline

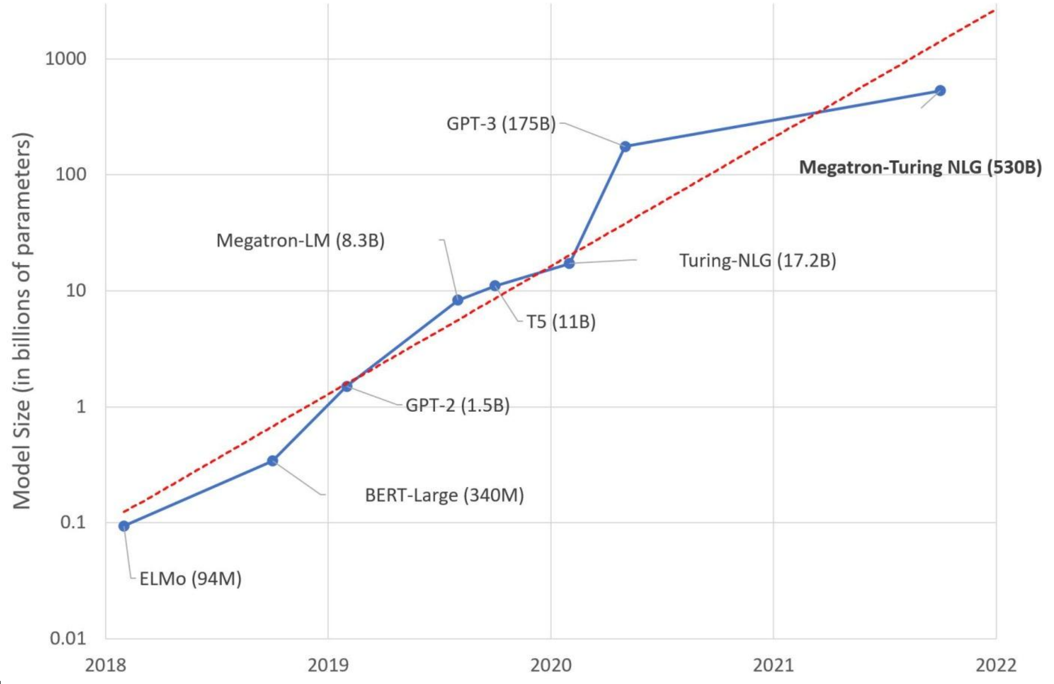
- Prompt Learning
- Chain of Thought Prompting
- Alignment

LLM Prompting

Motivation

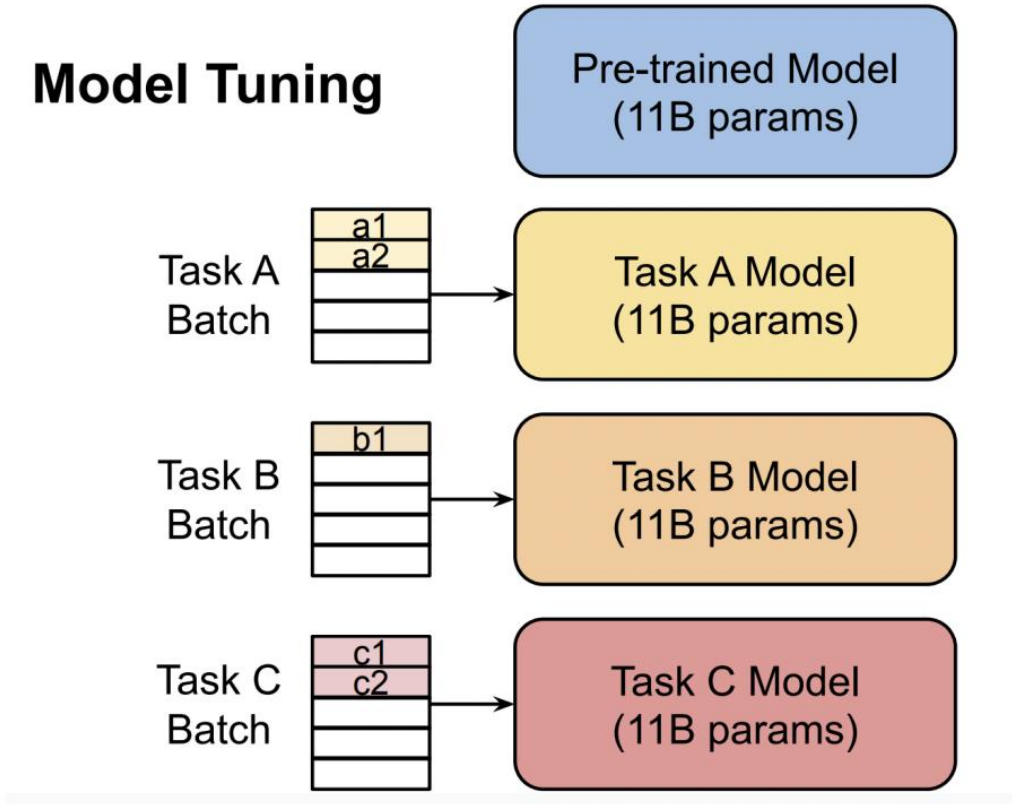
- The language model scaling wars!
- Pretraining + finetuning becomes a paradigm for downstream tasks
- For large models, fine-tuning hundreds of billions of parameters (even a few iterations) is also costly!
- How to reduce the "usage cost" of these PLMs?

A new 530B param model was released late last year



- LLaMa (Meta): 65B params, 80 layers, 8192 d_model,1.4 trillion tokens of training data
- PaLM (Google): 540B params, 118 layers, 18432 d_model, 780 billion training tokens

Practical challenges: large-scale models are costly to share and serve



Model Tuning
Pre-trained Model (11B params)
a1
a2
Task A Batch
Task A Model (11B params)
b1
Task B Batch
Task B Model (11B params)
c1
c2
Task C Batch
Task C Model (11B params)

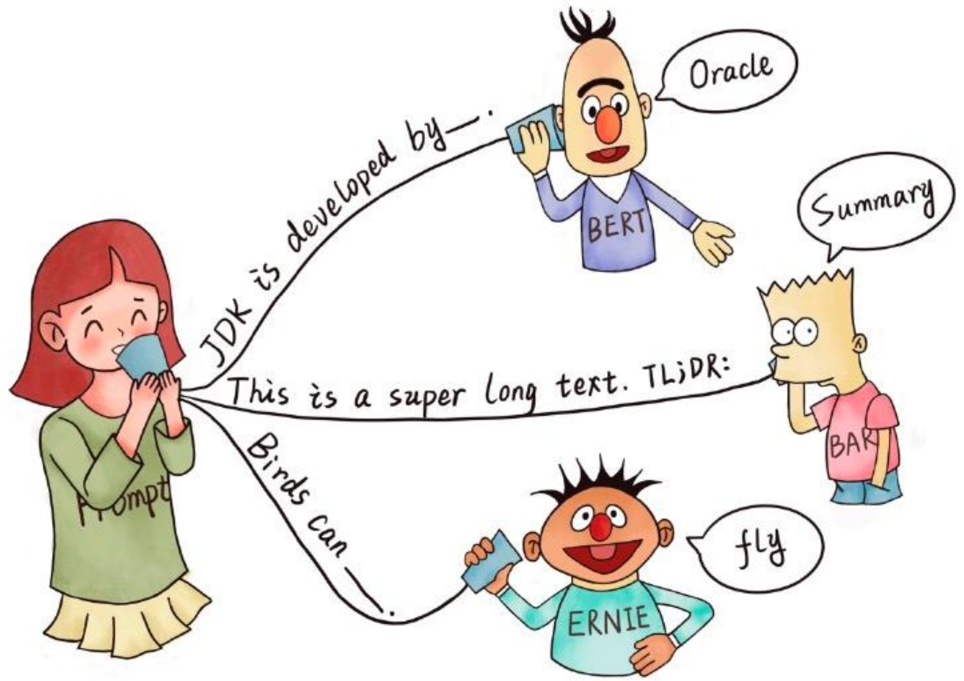
<u>Language model prompting to the rescue!</u>

Prompt learning

- GPT3: Language Models are Few-Shot Learners
- GPT-3 can well handle a wide range of tasks with only a few examples by leveraging natural-language *prompts* and task *demonstrations* as context, while not updating the parameters in the underlying model.
- The giant model size of GPT-3 is an important factor for its success, while the concept of prompts and demonstrations also gives us new insights about how we can better use language models.

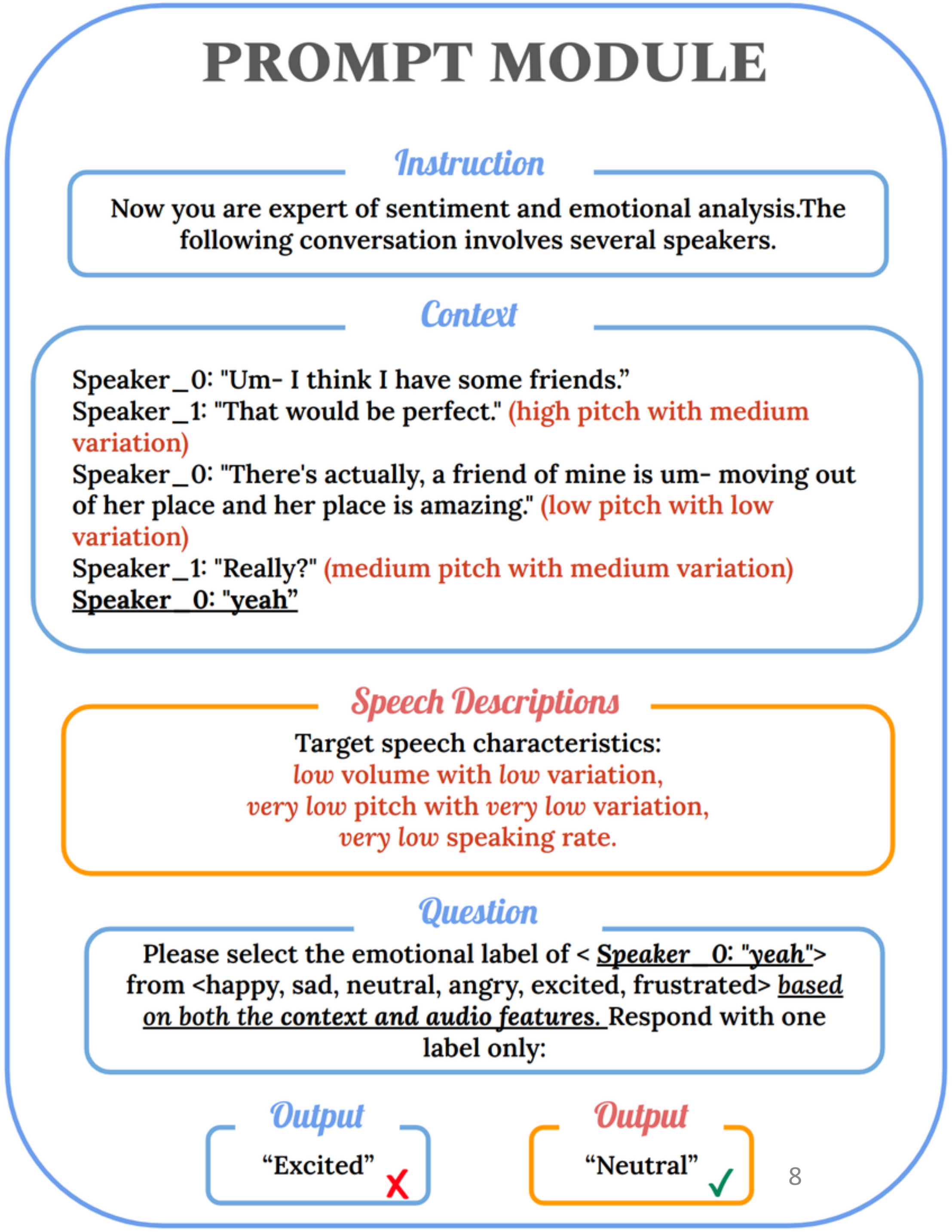
What is a prompt?

- A prompt is a piece of text inserted in the input examples, so that the original task can be formulated as a (masked) language modeling problem.
- For example, we want to classify the sentiment of the movie review *"No reason to watch"*, we can append a prompt "It was" to the sentence, getting *"No reason to watch.* It was _____." It is natural to expect a higher probability from the language model to generate "terrible" than "great".



An LLM prompt example

- Prompt may include
  - Instruction
  - Context
  - Descriptions
  - Questions
  - Output types
  - …



PROMPT MODULE

Instruction

Now you are expert of sentiment and emotional analysis.The following conversation involves several speakers.

Context

Speaker_0: "Um- I think I have some friends."
Speaker_1: "That would be perfect." (high pitch with medium variation)
Speaker_0: "There's actually, a friend of mine is um- moving out of her place and her place is amazing." (low pitch with low variation)
Speaker_1: "Really?" (medium pitch with medium variation)
Speaker_0: "yeah"

Speech Descriptions

Target speech characteristics:
low volume with *low* variation,
very low pitch with *very low* variation,
very low speaking rate.

Question

Please select the emotional label of < *Speaker_0: "yeah"*> from <happy, sad, neutral, angry, excited, frustrated> *based on both the context and audio features.* Respond with one label only:

Output

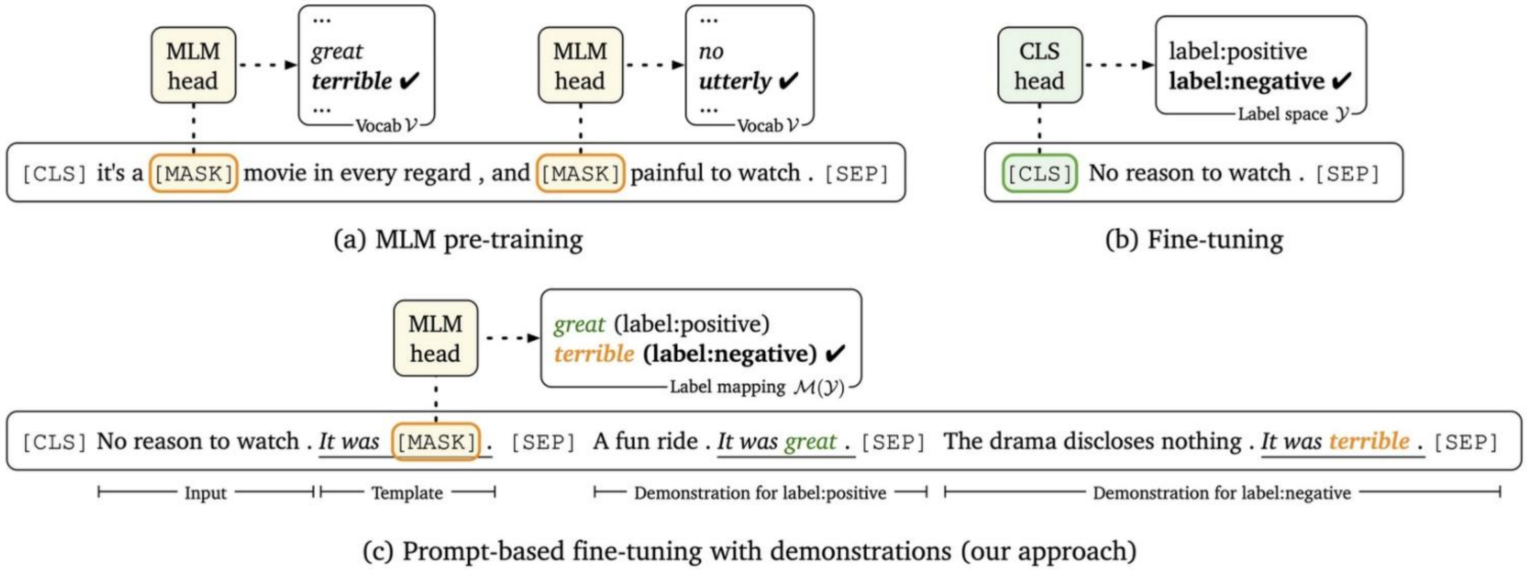
"Excited" ✗

Output

"Neutral" ✓

Why prompt learning?

- Prompting makes it possible for downstream tasks to take the same format as the pre-training objectives and requires no new parameters.
- Just need to design a template ("It was") and the expected text responses (we call these label words, e.g., "great" for the positive label and "terrible" for the negative label.
- **Sampling efficiency for few-shot** case--a dozen of training examples for a new task, it is hard to fine-tune the pre-trained models and the new task-specific parameters effectively, but the process is much smoother with prompting.

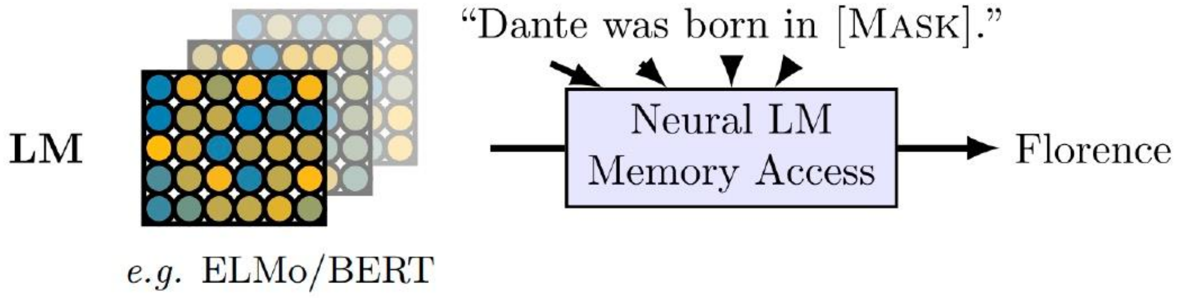


MLM head
great
terrible ✓
Vocab $\mathcal{V}$
MLM head
no
utterly ✓
Vocab $\mathcal{V}$
[CLS] it's a [MASK] movie in every regard , and [MASK] painful to watch . [SEP]
(a) MLM pre-training
CLS head
label:positive
label:negative ✓
Label space $\mathcal{Y}$
[CLS] No reason to watch . [SEP]
(b) Fine-tuning
MLM head
great (label:positive)
terrible (label:negative) ✓
Label mapping $\mathcal{M}(\mathcal{Y})$
[CLS] No reason to watch . It was [MASK] . [SEP] A fun ride . It was great . [SEP] The drama discloses nothing . It was terrible . [SEP]
Input
Template
Demonstration for label:positive
Demonstration for label:negative
(c) Prompt-based fine-tuning with demonstrations (our approach)

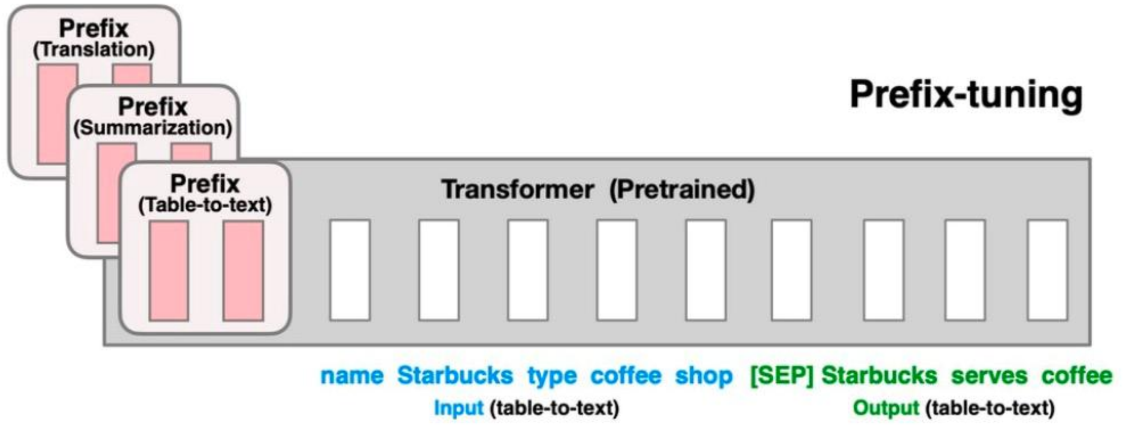
An illustration for pre-training, standard fine-tuning and prompt-based fine-tuning with demonstrations, taking a sentiment classification task as an example (from Gao et al., 2021).

Prompt shape

- Cloze prompts: fill in the blanks of a textual string



- Prefix prompts: continue a string prefix



Manual prompt engineering

- Create intuitive templates based on human introspection.
  - The seminal LAMA [1] dataset provides manually created cloze templates to probe knowledge in LMs.
  - Brown et al. [5] create manually crafted prefix prompts to handle a wide variety of tasks, including question answering, translation, and probing tasks for common-sense reasoning.
- Issues with manual prompt engineering
  - creating and experimenting with these prompts is an art that takes time and experience
  - even experienced prompt designers may fail to manually discover optimal prompts

Automated prompt searching

- Discrete prompts (a.k.a. hard prompts): templates described in a discrete space, usually corresponding to natural language phrases.

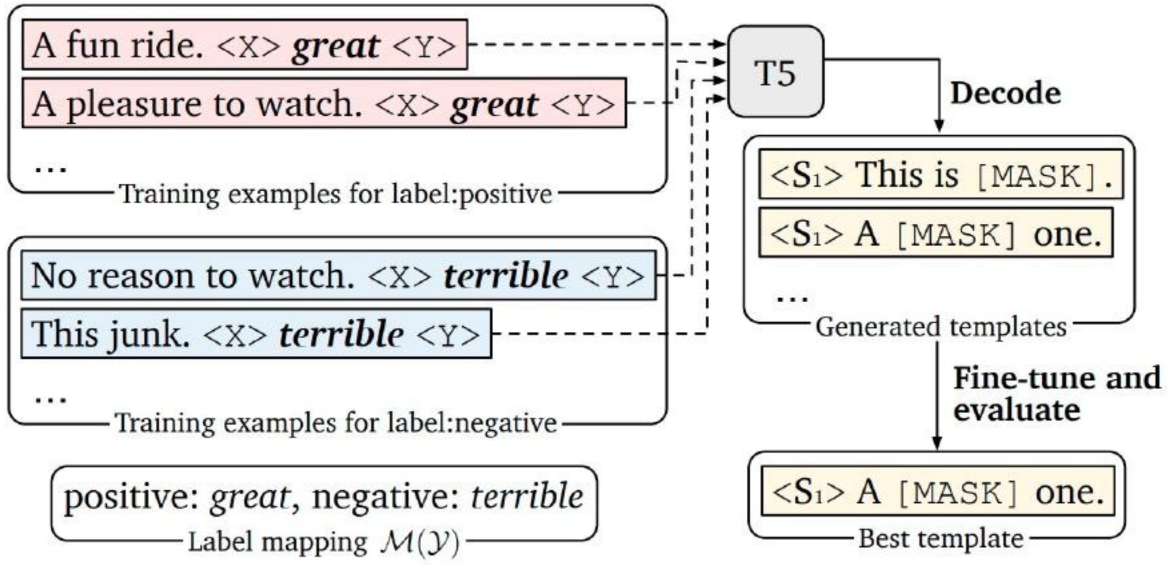
- Continuous prompts (a.k.a. soft prompts): rather than being for human consumption, it is not necessary to limit the prompt to human-interpretable natural. Continuous prompts perform prompting directly in the embedding space of the model.

Discrete prompts (1)

- Mining-based approach: Jiang et al.’s[6] MINE:
  - Idea: Extract the middle words or phrases between subjects and objects as prompt:“Barack Obama was born in Hawaii” is converted into a prompt “x was born in y”
- Prompt paraphrasing: Jiang et al.’s[6] paraphrasing: Using round-trip translation of the prompt into another language then back.
  - Target: to improve lexical diversity while remaining relatively faithful to the original prompt.
  - Method: performing paraphrasing over the original prompt into other semantically similar or identical expressions.
  - Example: “x shares a border with y” could be translate to another language and translate back to English again, the resulted texts could be “x has a common border with y” and “x adjoins y”

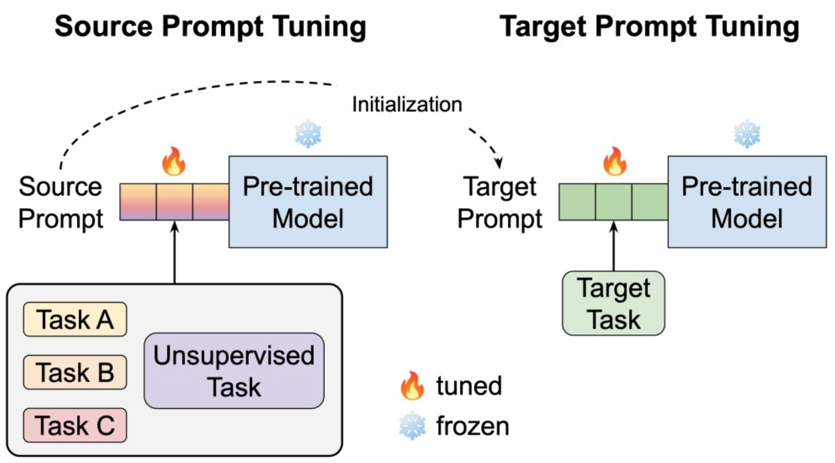
Discrete prompts (2)

- Gradient-based search
  - Wallace et al. [7] applied a gradient-based search over actual tokens to find short sequences that can trigger the underlying pre-trained LM to generate the desired target prediction.
- Prompt generation:
  - Gao et al. [8] introduce the seq2seq pre-trained model T5 into the template search process.
  - specifying the position to insert template tokens within a template
  - provide training samples for T5 to decode template tokens.


A fun ride. <X> great <Y>
A pleasure to watch. <X> great <Y>
...
Training examples for label:positive
No reason to watch. <X> terrible <Y>
This junk. <X> terrible <Y>
...
Training examples for label:negative
positive: great, negative: terrible
Label mapping $\mathcal{M}(\mathcal{Y})$
T5
Decode
<S1> This is [MASK].
<S1> A [MASK] one.
...
Generated templates
Fine-tune and evaluate
<S1> A [MASK] one.
Best template

Continuous prompts (1)

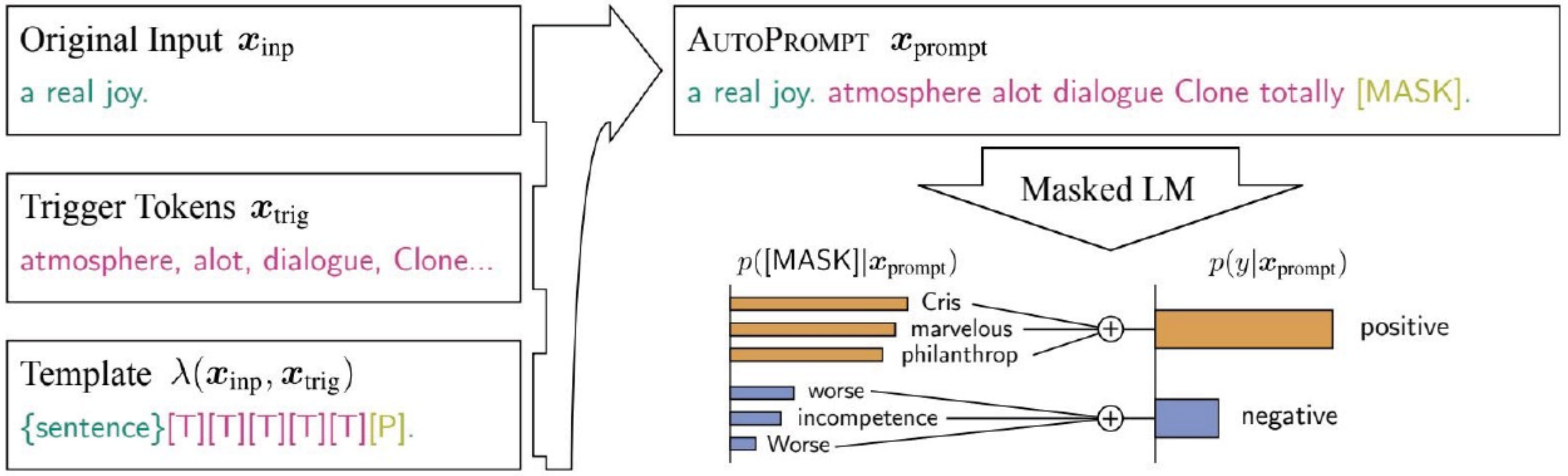
- Continuous prompts remove two constraints:
  - relax the constraint that the embeddings of template words be the embeddings of natural language (e.g., English) words.
  - Remove the restriction that the template is parameterized by the pre-trained LM's parameters.
- Prefix tuning [3] is a method that prepends a sequence of continuous task-specific vectors to the input, while keeping the LM parameters frozen.



We learn a single generic source prompt on one or more source tasks, which is then used to initialize the prompt for each target task.

Continuous prompts (2)

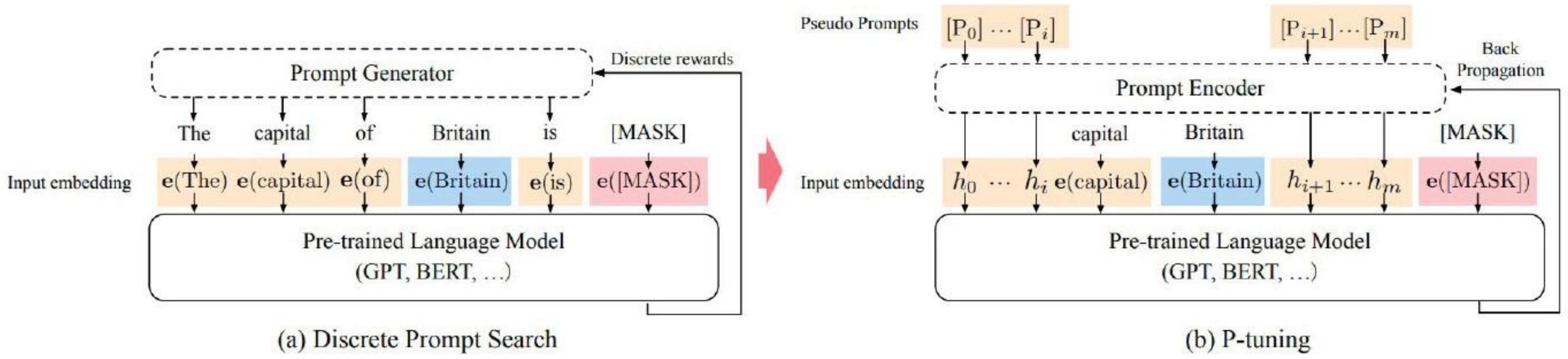
- Tuning initialized with discrete prompts:
  - Autoprompt [10]: Initialize the search for a continuous prompt using a prompt that has already been created or discovered using discrete prompt search methods.



Original Input $\boldsymbol{x}_{\text{inp}}$
a real joy.
Trigger Tokens $\boldsymbol{x}_{\text{trig}}$
atmosphere, alot, dialogue, Clone...
Template $\lambda(\boldsymbol{x}_{\text{inp}}, \boldsymbol{x}_{\text{trig}})$
{sentence}[T][T][T][T][T][P].
AutoPrompt $\boldsymbol{x}_{\text{prompt}}$
a real joy. atmosphere alot dialogue Clone totally [MASK].
Masked LM
$p([\text{MASK}]|\boldsymbol{x}_{\text{prompt}})$
$p(y|\boldsymbol{x}_{\text{prompt}})$
Cris
marvelous
philanthrop
worse
incompetence
Worse
positive
negative

Continuous prompts (3)

- Hard-soft prompt hybrid tuning: continuous prompts are learned by inserting trainable variables into the embedded input.
    - P-Tuning [11]: Instead of using a purely learnable prompt template, these methods insert some tunable embeddings into a hard prompt template.



Answer to prompts

- Language model is used to fill the unfilled slot to obtain final outputs.
- What is the answer to prompts?
- Some common choices for the shape of an answer include:
  - Tokens: One of the tokens in the pre-trained LM's vocabulary, or a subset of the vocabulary [1, 2].
  - Span: A short multi-token span. These are usually used together with cloze prompts [12].
  - Sentence: A sentence or document. These are commonly used with prefix prompts.

Challenges

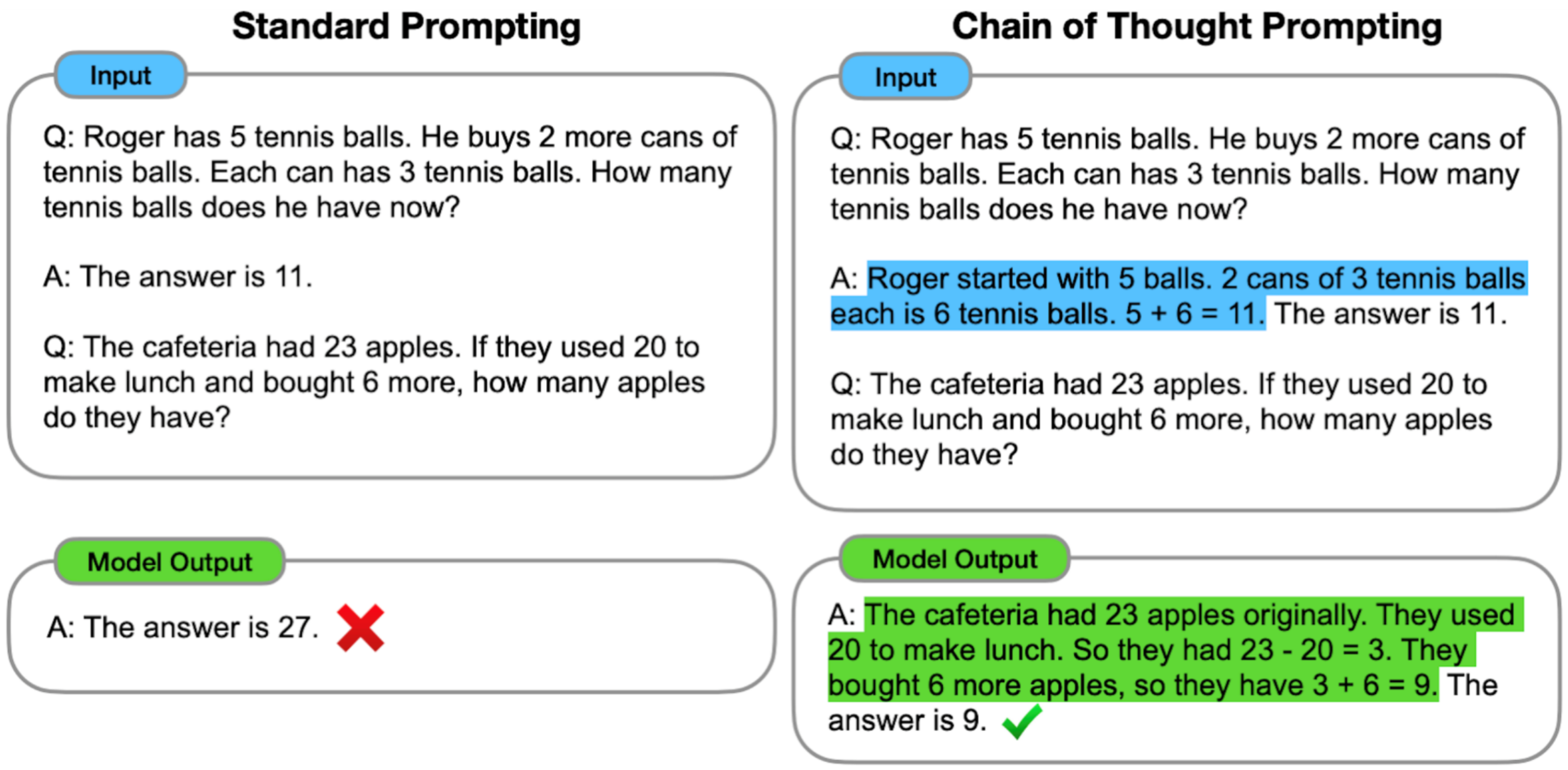
- Tasks beyond classification and generation:
  - Most existing works about prompt-based learning revolve around either text classification or generation-based tasks. Lots of different tasks haven’t been discussed.
- Prompting with structured information:
  - How to express tree, graph, table or relational structures in prompt learning is a major challenge.
- Entanglement of template and answer:
  - The performance highly depend on both the templates and answer mapping. How to effectively search or learn for the best combination of template and answer mappings is still a challenge.

Chain of Thought Prompting

Chain of thought prompting (1)

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models (Jason Wei)

A chain of thought is a series of intermediate natural language reasoning steps that lead to the final output, and we refer to this approach as chain-of-thought prompting.



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?

A: The answer is 27. ✗

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.

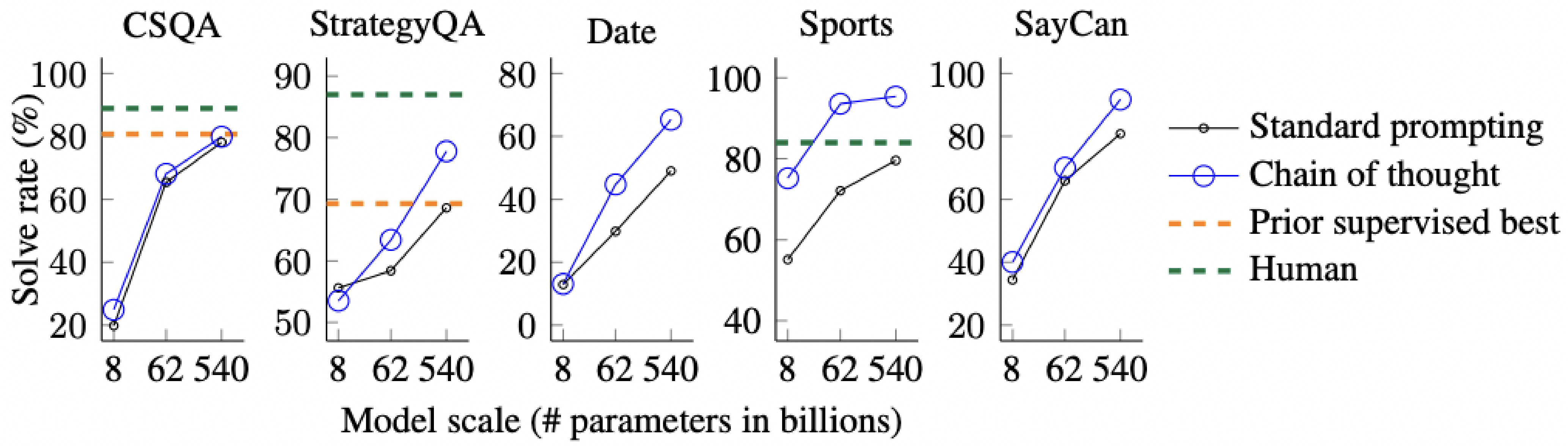
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

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 (2)

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models (Jason Wei)

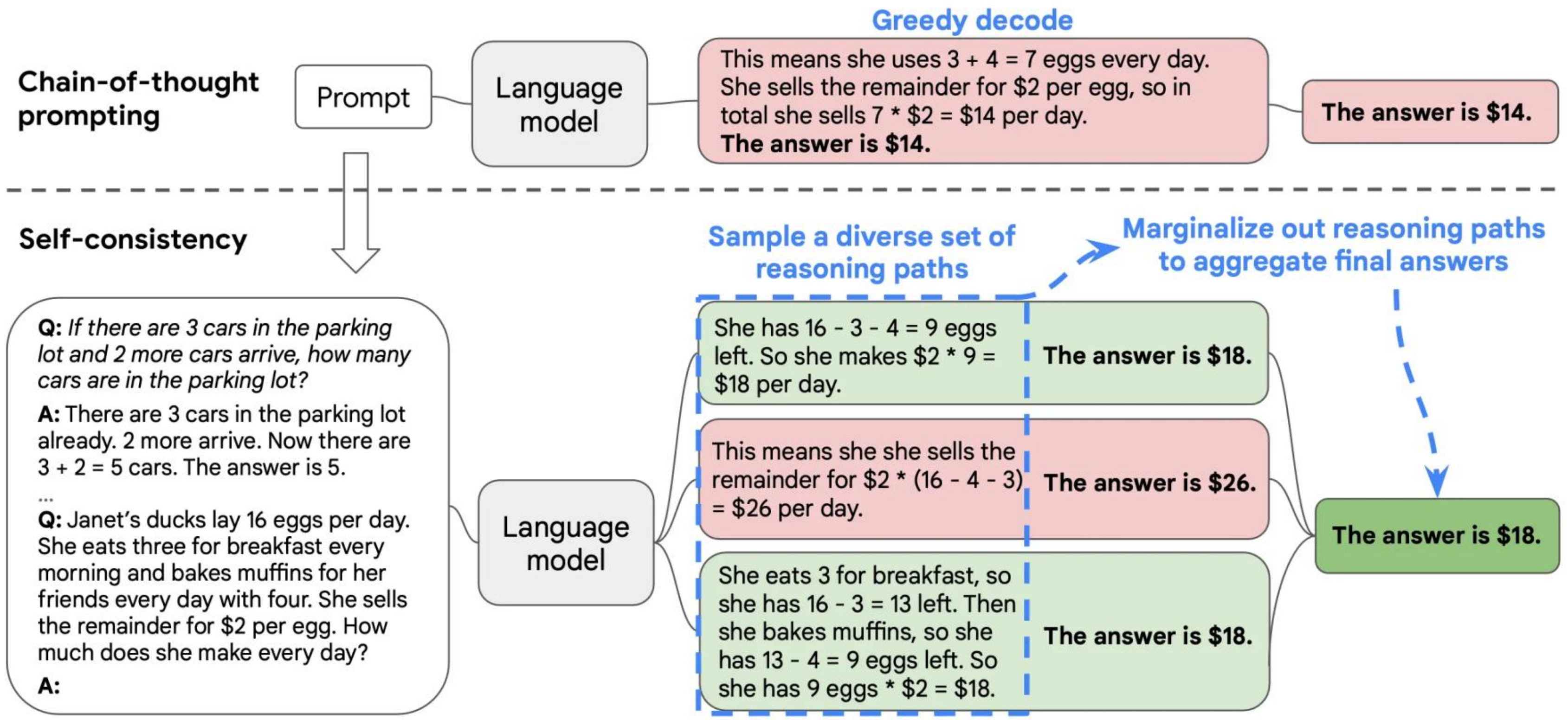
A chain of thought is a series of intermediate natural language reasoning steps that lead to the final output, and we refer to this approach as chain-of-thought prompting.



Chain of thought prompting (3)

Self-consistency Improves Chain Of Thought Reasoning in Language Models

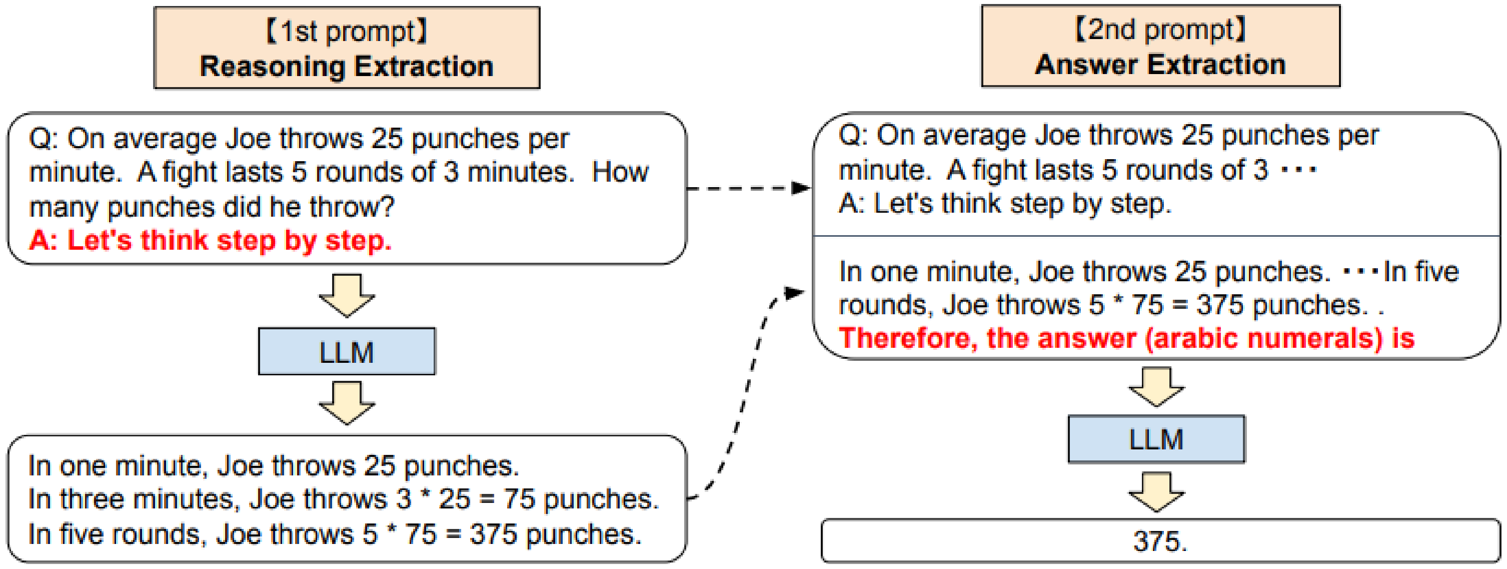
A series of reasoning paths are sampled from the decoder of the large model. Each path can correspond to a final answer. We can select more paths that can obtain consistent answers as the reasoning paths obtained by our sampling. Based on this direct voting strategy, it is more in line with human intuition, that is, if many reasoning paths can get a corresponding answer, then the confidence of this answer will be relatively high.


Greedy decode
Chain-of-thought prompting
Prompt
Language model
This means she uses 3 + 4 = 7 eggs every day. She sells the remainder for $2 per egg, so in total she sells 7 * $2 = $14 per day. The answer is $14.
The answer is $14.
Self-consistency
Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?
A: There are 3 cars in the parking lot already. 2 more arrive. Now there are 3 + 2 = 5 cars. The answer is 5.
...
Q: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder for $2 per egg. How much does she make every day?
A:
Language model
Sample a diverse set of reasoning paths
Marginalize out reasoning paths to aggregate final answers
She has 16 - 3 - 4 = 9 eggs left. So she makes $2 * 9 = $18 per day.
The answer is $18.
This means she she sells the remainder for $2 * (16 - 4 - 3) = $26 per day.
The answer is $26.
She eats 3 for breakfast, so she has 16 - 3 = 13 left. Then she bakes muffins, so she has 13 - 4 = 9 eggs left. So she has 9 eggs * $2 = $18.
The answer is $18.
The answer is $18.

Chain of thought prompting (4)

Large Language Models are Zero-Shot Reasoners

CoT requires manual annotation of prompts. This work found that only adding a fixed prompt: "Lets think step by step" can prompt the large model to reason step by step to generate results.

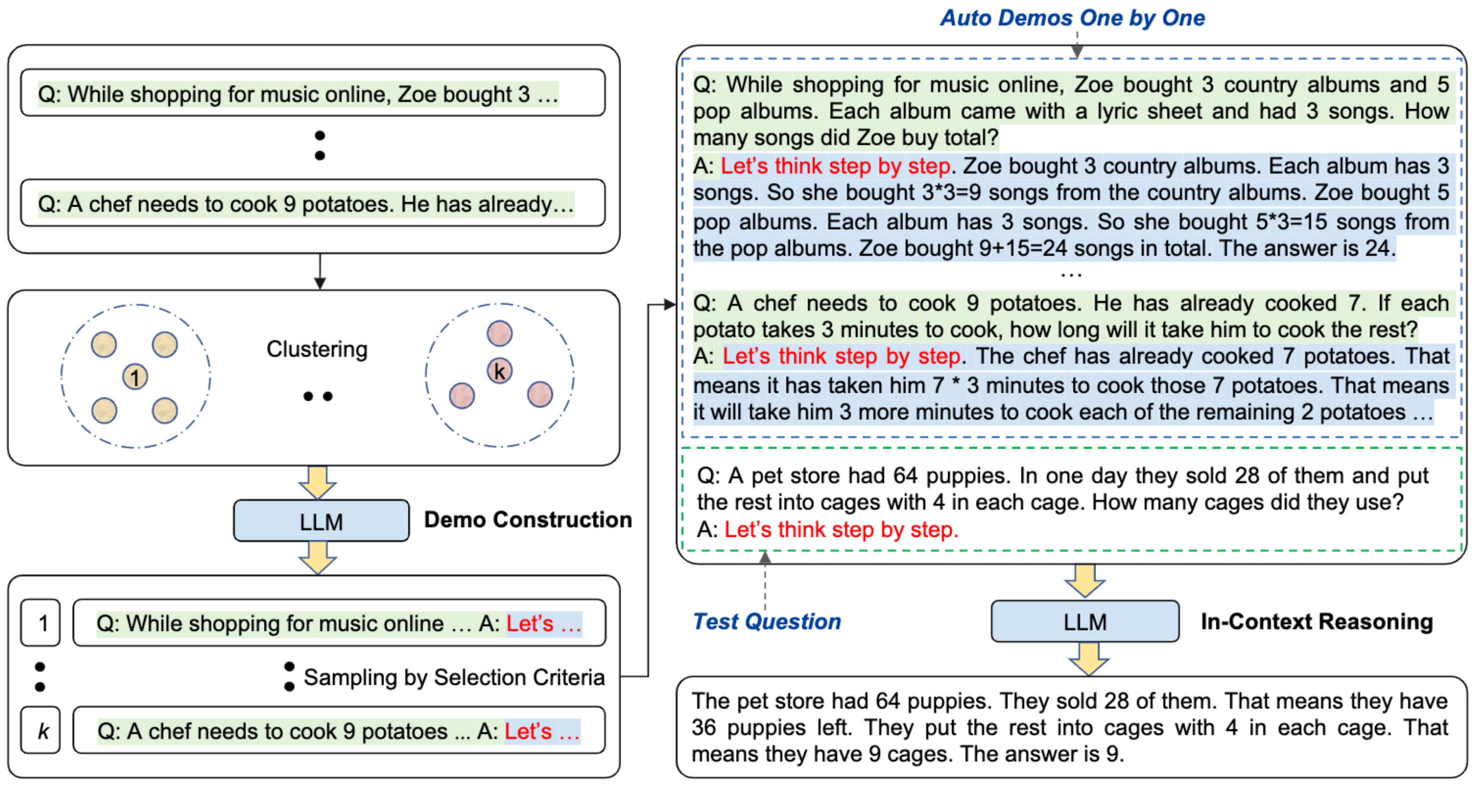

【1st prompt】
Reasoning Extraction
Q: On average Joe throws 25 punches per minute. A fight lasts 5 rounds of 3 minutes. How many punches did he throw?
A: Let's think step by step.
LLM
In one minute, Joe throws 25 punches.
In three minutes, Joe throws 3 * 25 = 75 punches.
In five rounds, Joe throws 5 * 75 = 375 punches.
【2nd prompt】
Answer Extraction
Q: On average Joe throws 25 punches per minute. A fight lasts 5 rounds of 3 ···
A: Let's think step by step.
In one minute, Joe throws 25 punches. ···In five rounds, Joe throws 5 * 75 = 375 punches. .
Therefore, the answer (arabic numerals) is
LLM
375.

1st prompt: reasoning extraction: first build the template, and then feed the survival result into the large model;
2nd prompt: answer extraction: splice them together, feed them into the large model again, and directly generate the results.

Chain of thought prompting (5)

Automatic Chain of Thought Prompting in Large Language Models

The previous chain-of-thought includes two types, one is Zero-shot CoT (let's think step by step), and the other is Manual-CoT (splicing several samples as demonstration). The author found that large models will generate incorrect chains regardless of prompt mode. In order to avoid this problem, they consider proposing a method to automatically build demonstrations-Auto-CoT.


Q: While shopping for music online, Zoe bought 3 …
Q: A chef needs to cook 9 potatoes. He has already…
Clustering
1
k
LLM
Demo Construction
1
Q: While shopping for music online … A: Let's …
Sampling by Selection Criteria
k
Q: A chef needs to cook 9 potatoes ... A: Let's …
Auto Demos One by One
Test Question
LLM
In-Context Reasoning

Q: While shopping for music online, Zoe bought 3 country albums and 5 pop albums. Each album came with a lyric sheet and had 3 songs. How many songs did Zoe buy total?
A: Let's think step by step. Zoe bought 3 country albums. Each album has 3 songs. So she bought 3*3=9 songs from the country albums. Zoe bought 5 pop albums. Each album has 3 songs. So she bought 5*3=15 songs from the pop albums. Zoe bought 9+15=24 songs in total. The answer is 24.

…

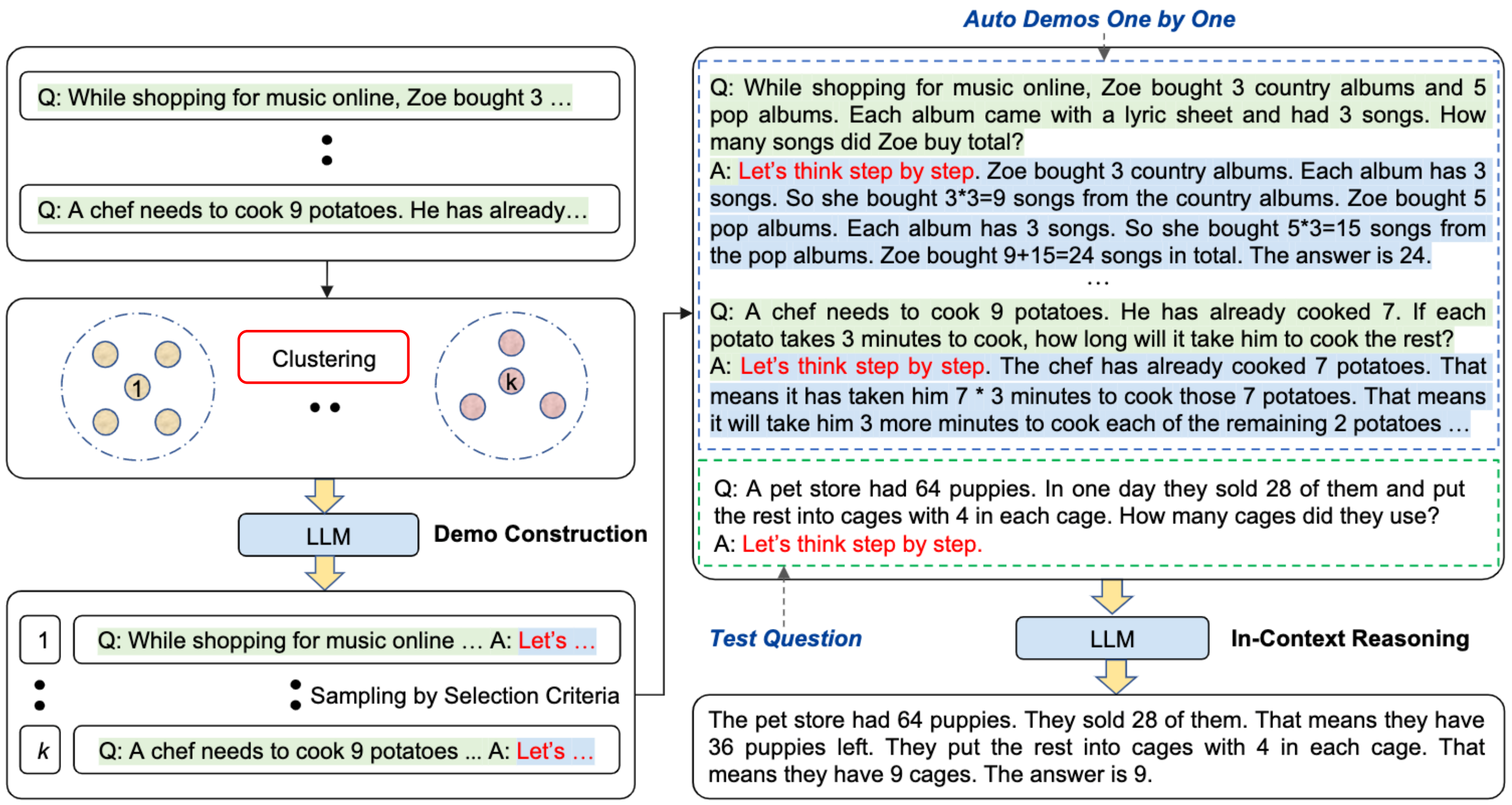
Q: A chef needs to cook 9 potatoes. He has already cooked 7. If each potato takes 3 minutes to cook, how long will it take him to cook the rest?
A: Let's think step by step. The chef has already cooked 7 potatoes. That means it has taken him 7 * 3 minutes to cook those 7 potatoes. That means it will take him 3 more minutes to cook each of the remaining 2 potatoes …

Q: A pet store had 64 puppies. In one day they sold 28 of them and put the rest into cages with 4 in each cage. How many cages did they use?
A: Let's think step by step.

The pet store had 64 puppies. They sold 28 of them. That means they have 36 puppies left. They put the rest into cages with 4 in each cage. That means they have 9 cages. The answer is 9.

Chain of thought prompting (6)

Automatic Chain of Thought Prompting in Large Language Models


Auto Demos One by One
Q: While shopping for music online, Zoe bought 3 …
Q: A chef needs to cook 9 potatoes. He has already…
Clustering
1
k
LLM
Demo Construction
1
Q: While shopping for music online … A: Let's …
Sampling by Selection Criteria
k
Q: A chef needs to cook 9 potatoes ... A: Let's …
Test Question
LLM
In-Context Reasoning

Q: While shopping for music online, Zoe bought 3 country albums and 5 pop albums. Each album came with a lyric sheet and had 3 songs. How many songs did Zoe buy total?
A: Let's think step by step. Zoe bought 3 country albums. Each album has 3 songs. So she bought 3*3=9 songs from the country albums. Zoe bought 5 pop albums. Each album has 3 songs. So she bought 5*3=15 songs from the pop albums. Zoe bought 9+15=24 songs in total. The answer is 24.
…
Q: A chef needs to cook 9 potatoes. He has already cooked 7. If each potato takes 3 minutes to cook, how long will it take him to cook the rest?
A: Let's think step by step. The chef has already cooked 7 potatoes. That means it has taken him 7 * 3 minutes to cook those 7 potatoes. That means it will take him 3 more minutes to cook each of the remaining 2 potatoes …

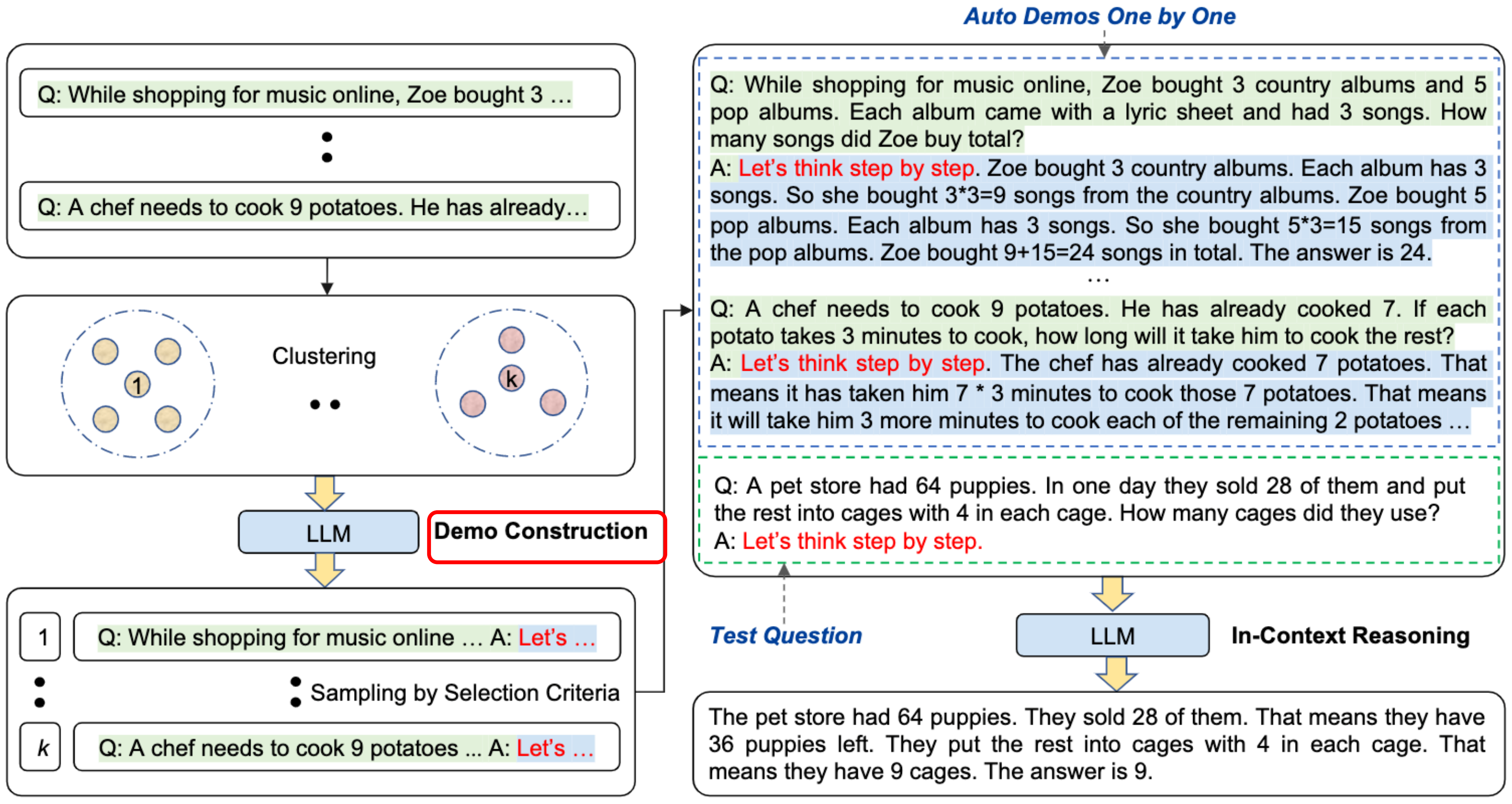
Q: A pet store had 64 puppies. In one day they sold 28 of them and put the rest into cages with 4 in each cage. How many cages did they use?
A: Let's think step by step.

The pet store had 64 puppies. They sold 28 of them. That means they have 36 puppies left. They put the rest into cages with 4 in each cage. That means they have 9 cages. The answer is 9.

Queston Clustering:
Use sentence-BERT to obtain representation for each question, and then obtain several clusters through K-means. For each cluster, sort them in ascending order according to their distance from the cluster center.

Chain of thought prompting (7)

Automatic Chain of Thought Prompting in Large Language Models



Q: While shopping for music online, Zoe bought 3 …

Q: A chef needs to cook 9 potatoes. He has already…

1 Q: While shopping for music online … A: Let's …

k Q: A chef needs to cook 9 potatoes ... A: Let's …

Q: While shopping for music online, Zoe bought 3 country albums and 5 pop albums. Each album came with a lyric sheet and had 3 songs. How many songs did Zoe buy total?
A: Let's think step by step. Zoe bought 3 country albums. Each album has 3 songs. So she bought 3*3=9 songs from the country albums. Zoe bought 5 pop albums. Each album has 3 songs. So she bought 5*3=15 songs from the pop albums. Zoe bought 9+15=24 songs in total. The answer is 24.
…
Q: A chef needs to cook 9 potatoes. He has already cooked 7. If each potato takes 3 minutes to cook, how long will it take him to cook the rest?
A: Let's think step by step. The chef has already cooked 7 potatoes. That means it has taken him 7 * 3 minutes to cook those 7 potatoes. That means it will take him 3 more minutes to cook each of the remaining 2 potatoes …

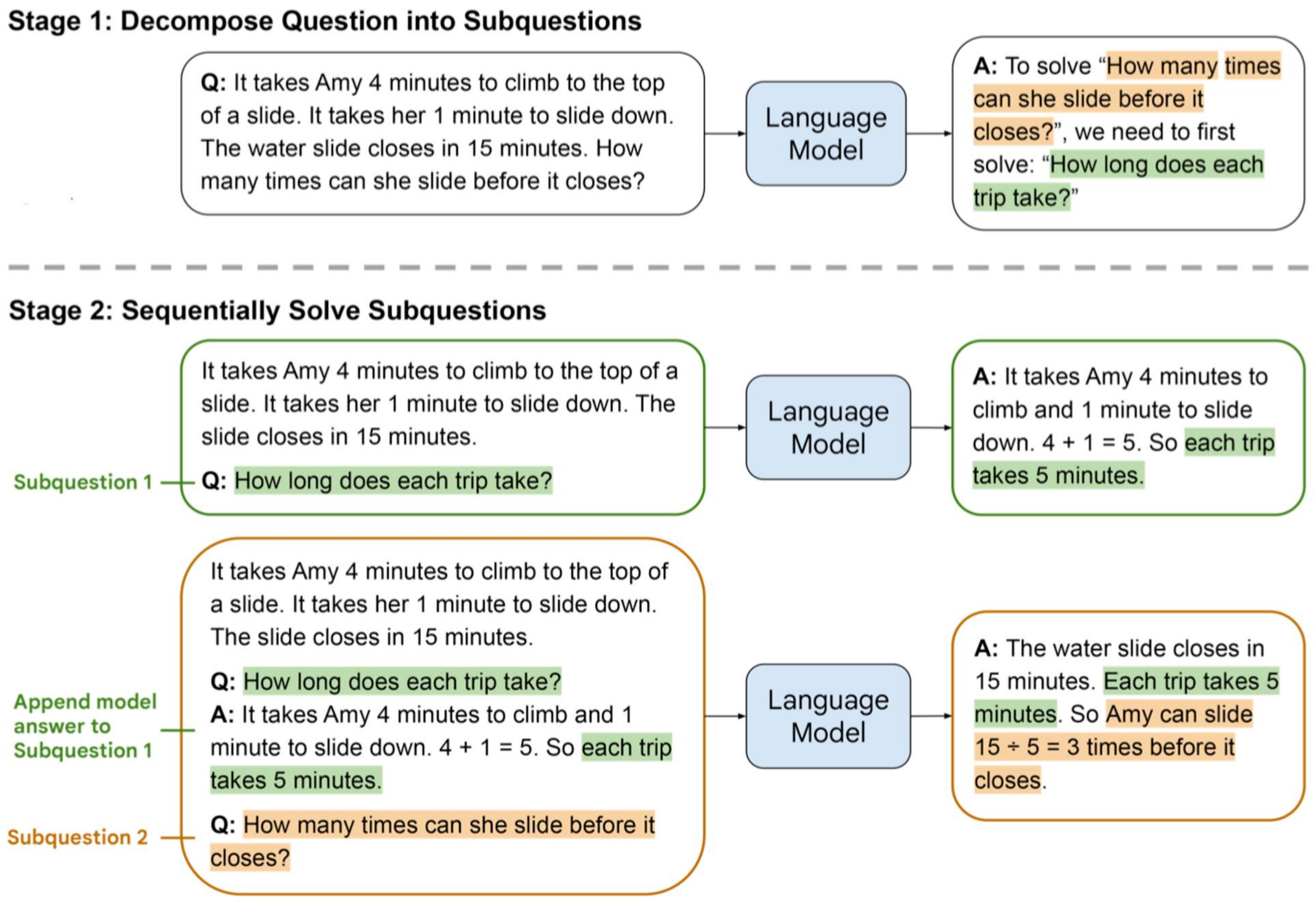
Q: A pet store had 64 puppies. In one day they sold 28 of them and put the rest into cages with 4 in each cage. How many cages did they use?
A: Let's think step by step.

The pet store had 64 puppies. They sold 28 of them. That means they have 36 puppies left. They put the rest into cages with 4 in each cage. That means they have 9 cages. The answer is 9.

Demonstration Sampling:
Based on the results of the Cluster, the appropriate prompt is obtained by sampling. For each cluster, sample a question, concatenate it with Let's think step-by-step, and feed it into a large model to survive the relationale. Finally, a relationale is spliced with the corresponding question and answer, and the target test sample is spliced, prompting the large model to generate the relationale of the test sample.

Chain of thought prompting (8)

Least-to-Most Prompting Enables Complex Reasoning in Large Language Models



Stage 1: Decompose Question into Subquestions
Language Model
Stage 2: Sequentially Solve Subquestions
Subquestion 1
Language Model
Append model answer to Subquestion 1
Subquestion 2
Language Model

Q: It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The water slide closes in 15 minutes. How many times can she slide before it closes?

A: To solve "How many times can she slide before it closes?", we need to first solve: "How long does each trip take?"

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Q: How long does each trip take?

A: It takes Amy 4 minutes to climb and 1 minute to slide down. 4 + 1 = 5. So each trip takes 5 minutes.

It takes Amy 4 minutes to climb to the top of a slide. It takes her 1 minute to slide down. The slide closes in 15 minutes.

Q: How long does each trip take?
A: It takes Amy 4 minutes to climb and 1 minute to slide down. 4 + 1 = 5. So each trip takes 5 minutes.

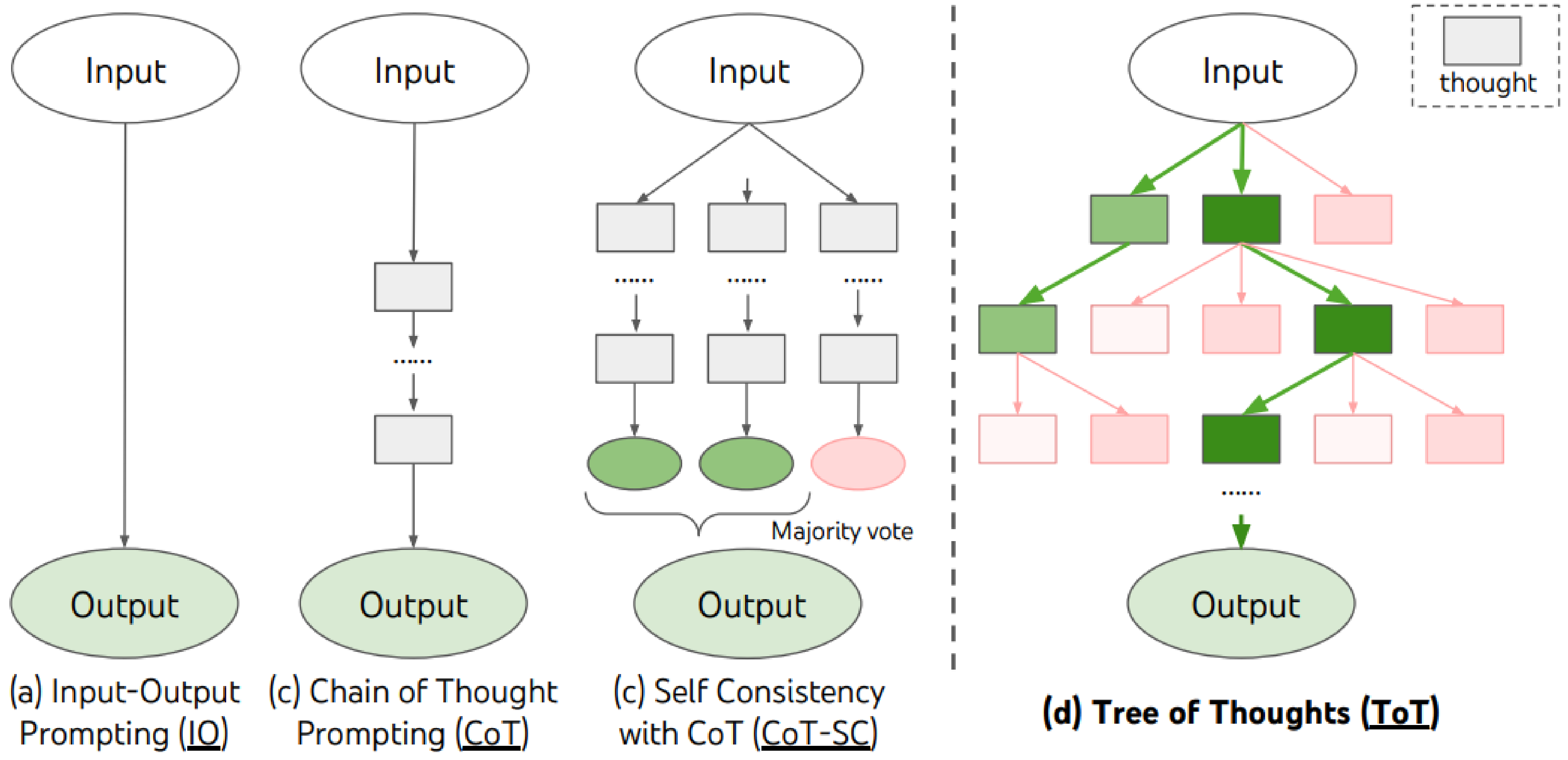
Q: How many times can she slide before it closes?

A: The water slide closes in 15 minutes. Each trip takes 5 minutes. So Amy can slide 15 ÷ 5 = 3 times before it closes.

Least-to-most prompting solving a math word problem in two stages: (1) query the language model to decompose the problem into subproblems; (2) query the language model to sequentially solve the subproblems. The answer to the second subproblem is built on the answer to the first subproblem. The demonstration examples for each stage's prompt are omitted in this illustration.

Chain of thought prompting (9)

Tree of Thoughts, Graph of Thoughts……


Input
Output
(a) Input-Output Prompting (IO)
Input
......
Output
(c) Chain of Thought Prompting (CoT)
Input
......
Majority vote
Output
(c) Self Consistency with CoT (CoT-SC)
Input
thought
......
Output
(d) Tree of Thoughts (ToT)

Future developments (1)

- In the commercial field, Prompt Learning is widely used in natural language processing, recommendation systems and other fields. Through Prompt Learning, companies can process large amounts of user feedback data more efficiently and provide users with more personalized services.
- In the social field, Prompt Learning is used in fields such as sentiment analysis and public opinion monitoring. Through Prompt Learning, researchers can more accurately analyze public emotions and opinions, thereby providing useful references for policy formulation and social management.

Future developments (2)

- In the scientific field, Prompt Learning is also widely used in various prediction and classification tasks. For example, in the medical field, Prompt Learning is used to predict disease development trends and treatment effects, helping doctors formulate more scientific treatment plans.
- In the field of environmental science, Prompt Learning is used to predict the behavior of complex systems such as climate change and natural disasters, thereby providing decision support for environmental protection and disaster prevention and control.
- In short, prompt learning, as a new machine learning method, uses contextual hints to guide the model to make predictions. It has many advantages and broad application prospects. Although Prompt Learning currently faces some challenges and limitations, with the continuous development and improvement of technology, we believe that Prompt Learning will become one of the important directions in the field of machine learning in the future.

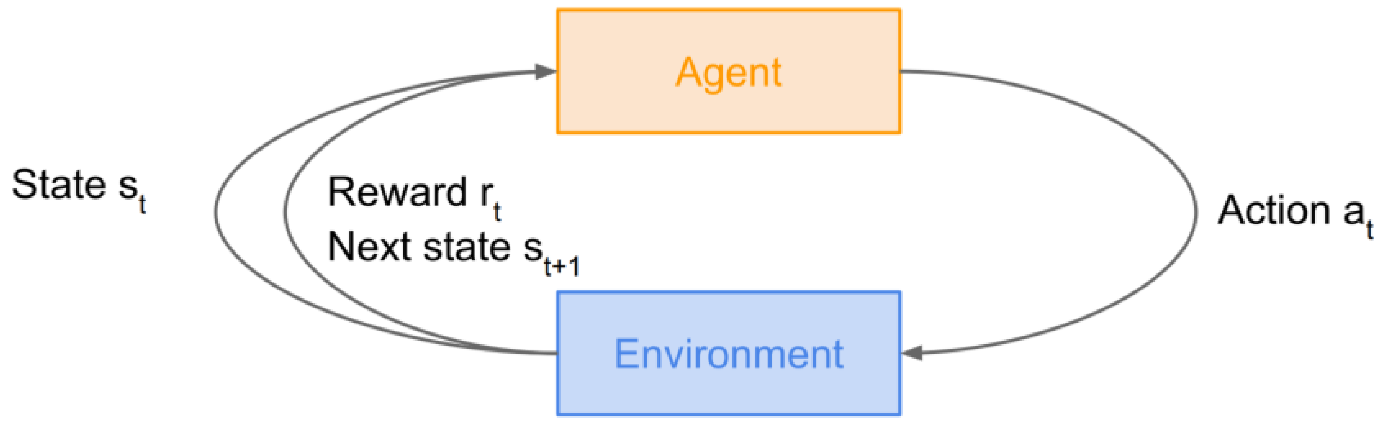
Reinforcement learning

At each step *t,* the Agent:

- Observes the state s_t from environment
- Executes action a_t

At each step t, the Environment:

- Receives action a_t
- Emits scalar reward r_t
- Emits next state s_{t+1}



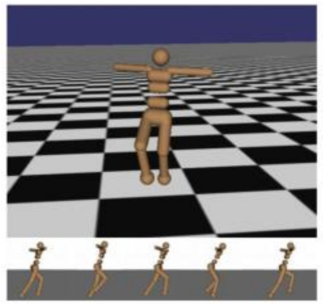
Characteristics of RL

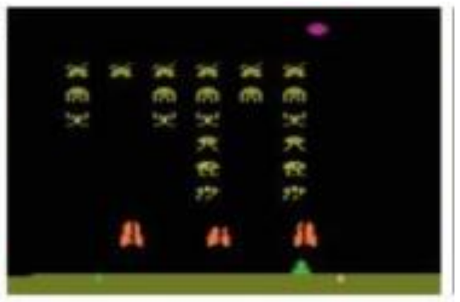
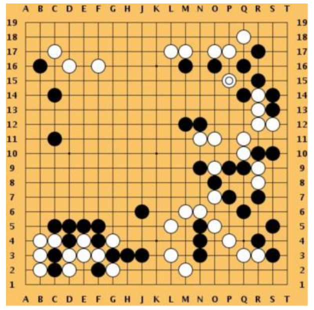
What makes reinforcement learning different from other machine learning paradigms?

- There is no supervisor, only a reward signal
- Feedback is delayed, not instantaneous
- Time really matters (sequential, non i.i.d. data)
- Agent's actions affect the subsequent data it receives
- The goal is to maximize the total rewards

Examples of RL

- Fly stunt maneuvers in a helicopter
- Make a humanoid robot walk
- Play many different Atari games better than humans
- Defeat the world champion at Go

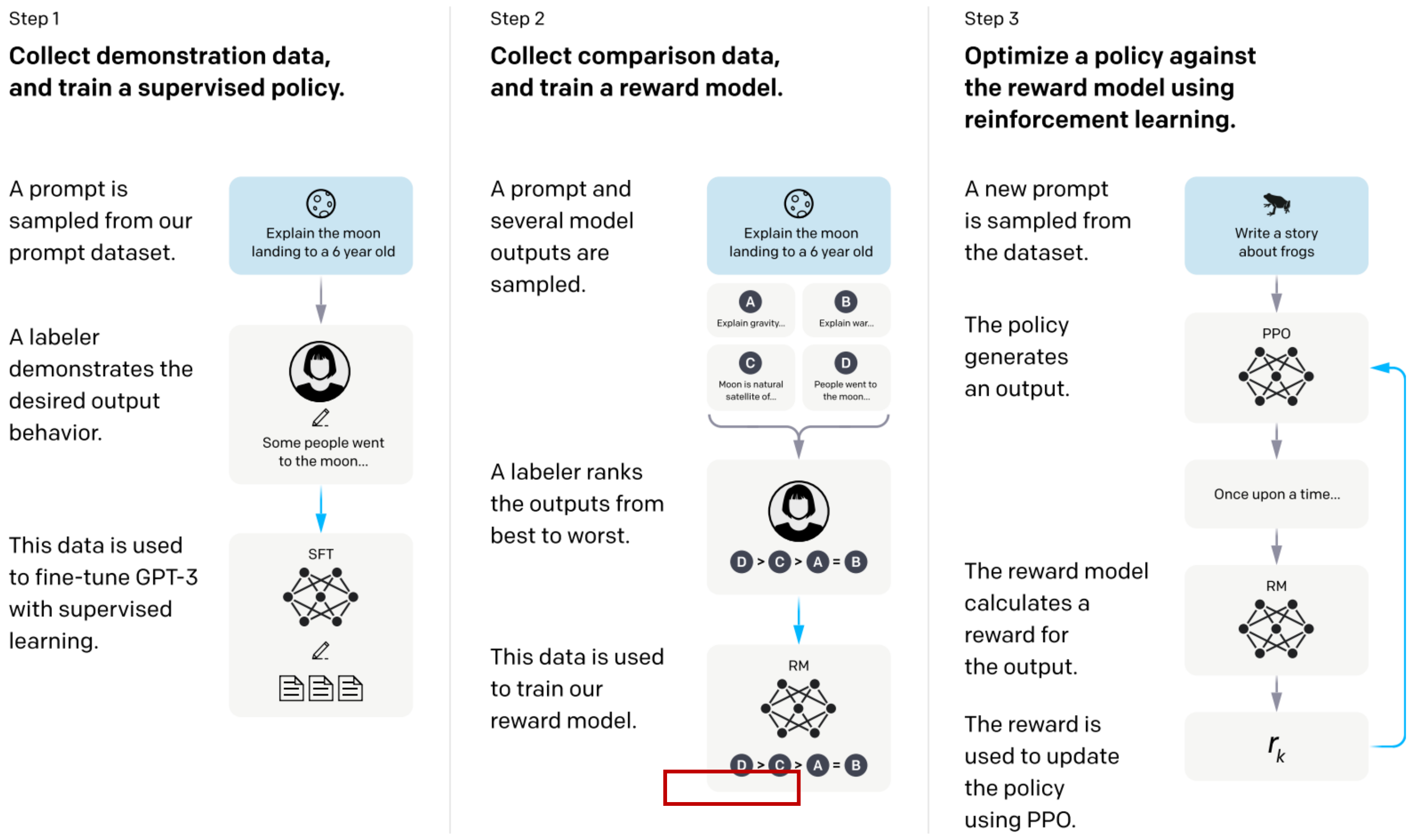


RL for LLM Alignment

- **Challenge:** LLMs can generate harmful, biased, or irrelevant content.
- **Goal:** Ensure LLMs produce outputs aligned with human values and intentions.
- **Solution:** Use reinforcement learning (RL) to fine-tune models based on feedback.

RLHF Review



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.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT
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
Write a story about frogs
PPO
Once upon a time...
RM
r_k

[Ouyang et al., 2022]

Policy in RL

- A policy is the agent's behavior
- It is a map from state to action: $f: s_t \longrightarrow a_t$
- Deterministic policy: $a = \pi(s)$
- Stochastic policy: $\pi(a/s) = \mathrm{P}[A_t = a \,/\, S_t = s]$

The Optimal Policy

We want to find optimal policy π^* that maximizes the sum of rewards.

How do we handle the randomness (initial state, transition probability⋯)?
Maximize the **expected sum of rewards!**

Formally: $\pi^* = \arg\max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi\right]$ with $s_0 \sim p(s_0), a_t \sim \pi(\cdot|s_t), s_{t+1} \sim p(\cdot|s_t, a_t)$

Where s_0 is the initial state, γ is the discounted rate, p is the transition probability of the environment

Policy Gradient

The optimal policy problems are usually solved using Q-Learning.

- Estimate a Q function to evaluate the states and actions

Problem with Q-Learning.

- The Q-function can be very complicated!

Example: a robot grasping an object has a very high-dimensional state =>
Hard to learn exact value of every (state, action) pair

But the policy can be much simpler

Can we learn a policy directly, e.g. finding the best policy from a collection of policies?

Policy Gradient

Formally, let's define a class of parametrized policies: $\Pi = \{\pi_\theta, \theta \in \mathbb{R}^m\}$

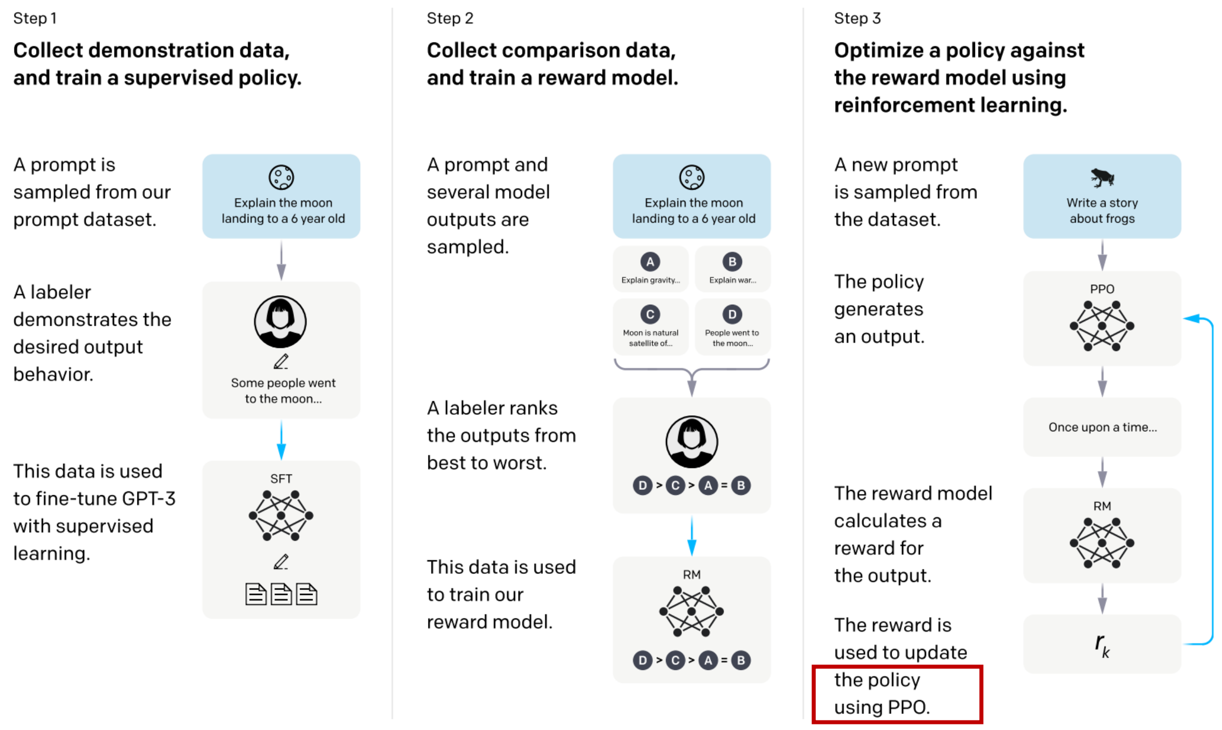
For each policy, define its value:

$$J(\theta) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | \pi_\theta\right]$$

We want to find the optimal policy $\theta^* = \arg\max_\theta J(\theta)$

How can we do this?
Gradient ascent on policy parameters! → Policy gradient

Policy Gradient


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.

Explain the moon landing to a 6 year old
Some people went to the moon...
SFT
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
Write a story about frogs
PPO
Once upon a time...
RM
r_k

The RL methods for LLM alignment which we will introduce later are basically derived from the policy gradient

Three RL Methods for LLM Alignment

- **PPO** (Proximal Policy Optimization): A RL algorithm used to train agents by optimizing their policies.
- **DPO** (Direct Preference Optimization): Align LLMs using human preferences without complex RL.
- **GRPO** (Group Relative Policy Optimization): A hybrid RL algorithm specifically designed to enhance reasoning capabilities in LLMs.

PPO

What is PPO?

- PPO (Proximal Policy Optimization): A popular reinforcement learning algorithm for training policies.
- Key Idea: Balances sample efficiency and stability during training.
- Core Mechanism: Uses a clipped objective to prevent large policy updates, ensuring smoother learning.

[Schulman et al., 2017]

PPO - Clipped Objective

- **Objective Function**

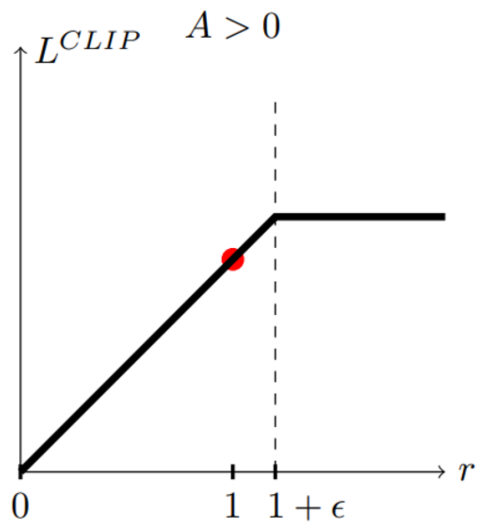
Use clip function to ensure small, stable updates to the policy.

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$$

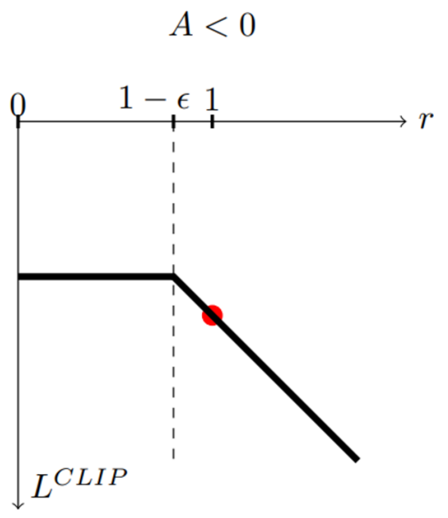
Where ϵ is the hyperparameter (clipping range),and $\hat{A}_t$ is the advantage value.

PPO – Understanding the Clipped Objective

- **Advantage is positive**



- **Advantage is negative**



PPO – Algorithm Details

1. First, collect some trajectories based on some policy π_θ, and initialize $\theta' = \theta$
2. Next, compute the gradient of the clipped surrogate function using the trajectories
3. Update θ' using gradient ascent
4. Then we repeat step 2-3 without generating new trajectories. Typically, step 2-3 are only repeated a few times
5. Set $\theta = \theta'$, go back to step 1, repeat.

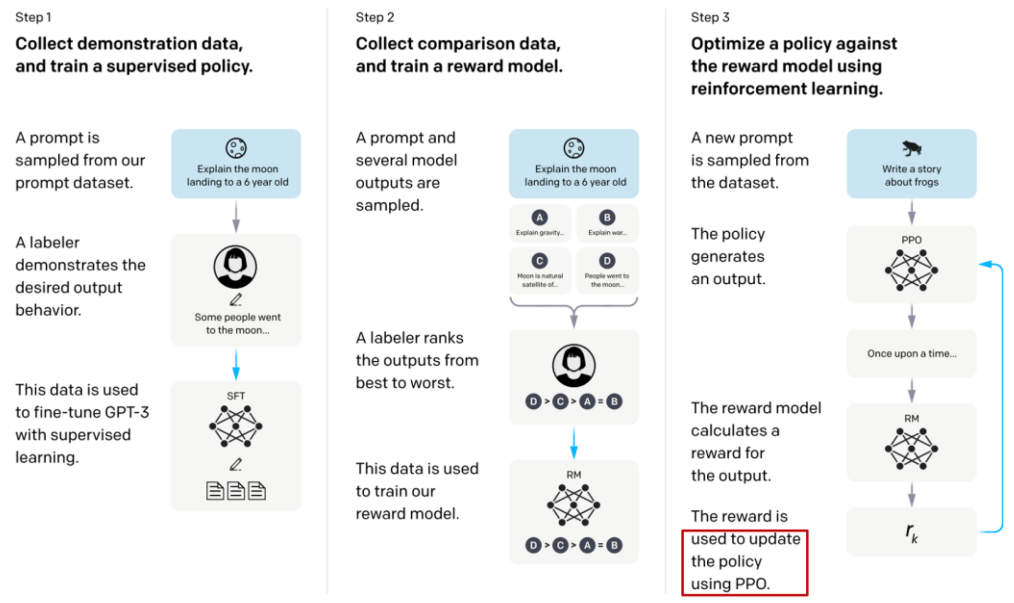
PPO in LLM Alignment

- Supervised Fine-Tuning (SFT): Train the model on high-quality data.
- Reward Modeling: Collect human feedback to create a reward function
- PPO Optimization: Use PPO to fine-tune the model to maximize the reward.
- Evaluation: Ensure alignment with human values.

PPO in LLM Alignment

- **Strength**

Proven effectiveness in RLHF (Step 3)


Step 1
Collect demonstration data, and train a supervised policy.
A prompt is sampled from our prompt dataset.
Explain the moon landing to a 6 year old
A labeler demonstrates the desired output behavior.
Some people went to the moon...
This data is used to fine-tune GPT-3 with supervised learning.
SFT
Step 2
Collect comparison data, and train a reward model.
A prompt and several model outputs are sampled.
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...
A labeler ranks the outputs from best to worst.
D > C > A = B
This data is used to train our reward model.
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.
Write a story about frogs
The policy generates an output.
PPO
Once upon a time...
The reward model calculates a reward for the output.
RM
The reward is used to update the policy using PPO.
r_k

- **Weakness**

Computationally expensive and complex to implement.

DPO

What is PPO?

- DPO (Direct Preference Optimization): A method for aligning models with human preferences without explicit reward modeling.

- Key Idea: Directly optimizes policy using preference data instead of learning a reward function.

[Rafailov et al., 2024]

DPO vs PPO

Why DPO?

PPO

- Requires learning a reward function from human feedback.
- Involves complex optimization and instability.

DPO

- Skips reward modeling entirely.
- Directly optimizes the policy using pairwise preference data.
- More efficient and scalable for LLM alignment.

DPO
How It Works

- Collect paired comparisons of model outputs (preferred vs. non-preferred)
- Directly optimize the LLM using a supervised loss function based on preferences.

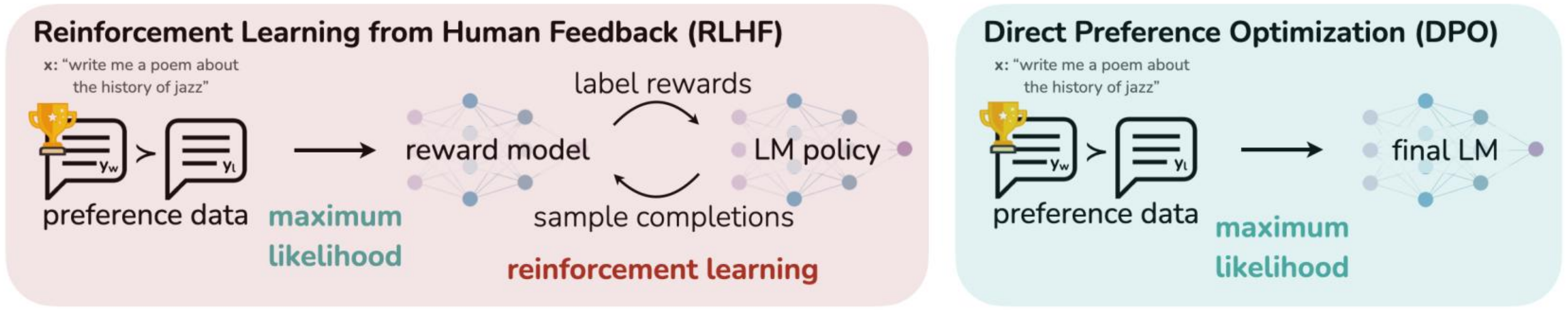

Reinforcement Learning from Human Feedback (RLHF)
x: "write me a poem about the history of jazz"
y_w > y_l
preference data
maximum likelihood
reward model
label rewards
sample completions
LM policy
reinforcement learning
Direct Preference Optimization (DPO)
x: "write me a poem about the history of jazz"
y_w > y_l
preference data
maximum likelihood
final LM

Figure: DPO optimizes for human preferences while avoiding reinforcement learning

DPO – Algorithm Details

1. First, collect preference data which are pairs of model outputs with human preferences (e.g., Output A > Output B).
2. Policy Optimization: Use a preference loss function to directly update the policy:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(x,y_w,y_l)}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta\log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right]$$

Where:

- y_w: Preferred output.
- y_l: Less preferred output.
- π_{ref}: Reference policy (e.g., SFT model).

3. Repeat until the policy aligns with human preferences.

DPO in LLM Alignment

- **Strength**
  - Simplicity: Eliminates the need for reward modeling, reducing complexity.
  - Efficiency: Faster training with fewer computational resources.
  - Scalability: Easier to apply to large-scale models.
- **Weakness**
  - Data Quality: Require high-quality preference data.
  - Generalization: May struggle with unseen scenarios.
  - Bias: Human preferences can introduce biases.

GRPO

What is GRPO?

- GRPO **(Group Relative Policy Optimization)** is a RL algorithm specifically designed to enhance reasoning capabilities inLLMs
- No separate value network for the Critic,
- Sample multiple outputs from the old policy for the same question or state, treat the average reward of these outputs as the baseline,

[Shao et al., 2025]

GRPO vs PPO

Why GRPO?

GRPO is designed to inherit the robustness of PPO while avoid expensive computations

PPO

- Requires learning a reward function from human feedback.
- Involves complex optimization and instability.

GRPO

- No separate value network for the Critic.
- More **adaptable** and **scalable** for complex tasks like LLM alignment.

GRPO vs PPO

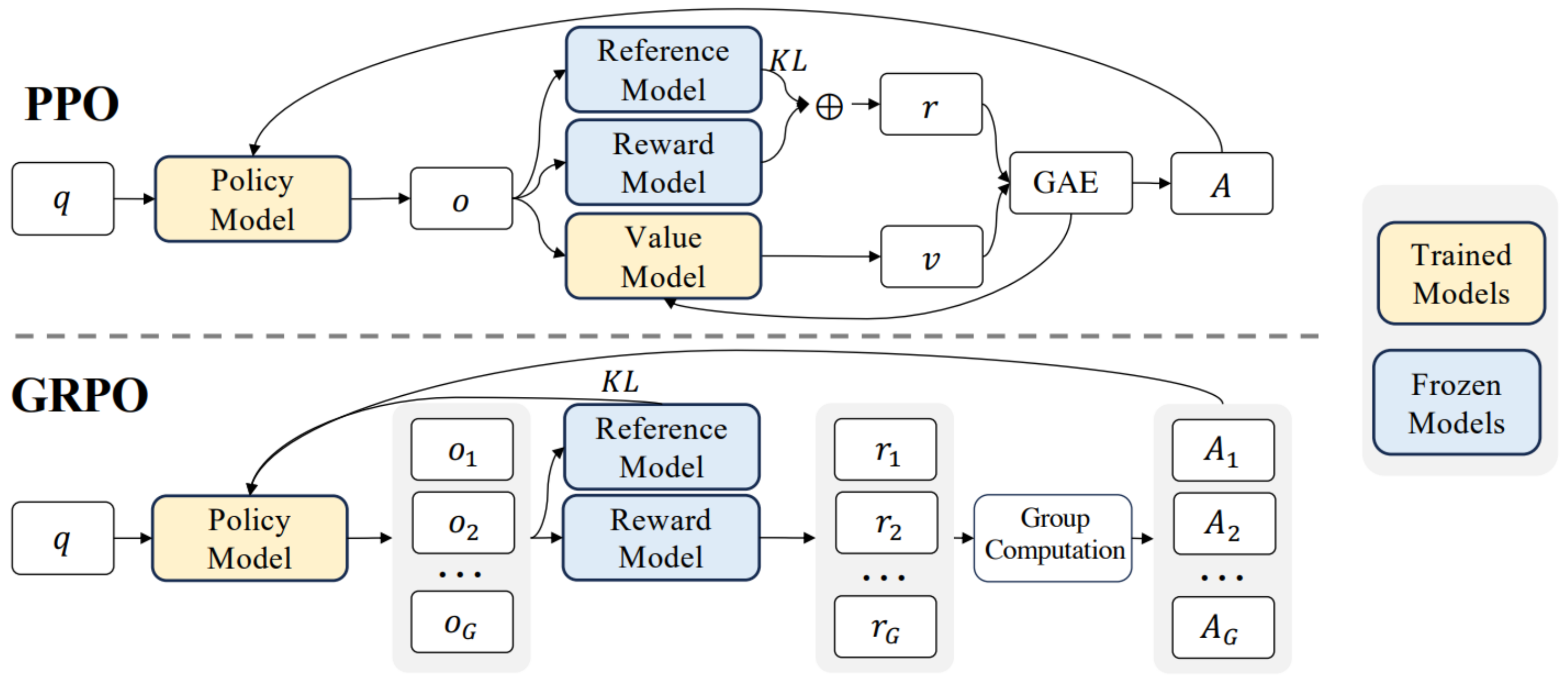


Figure: Demonstration of PPO and our GRPO. GRPO foregoes the value model, instead estimating the baseline from group scores, significantly reducing training resources.

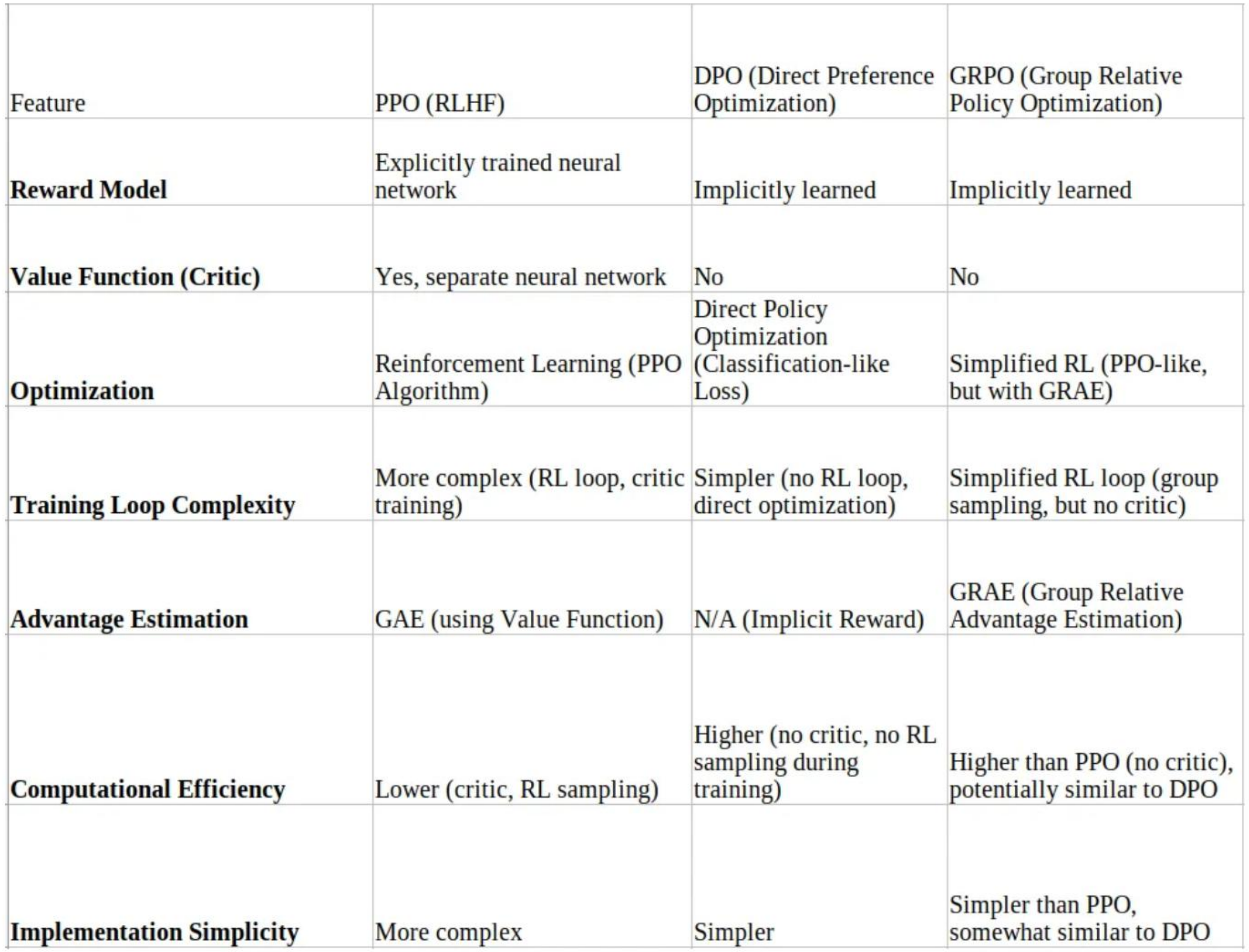
GRPO – Algorithm Details

1. Sample multiple outputs for each question.
2. Score outputs using a reward model.
3. Normalize rewards within groups.
4. Optimize the policy using relative advantages and KL regularization.
5. Update the policy model iteratively with new data.
6. Repeat for multiple iterations to refine performance.

GRPO in LLM Alignment

- **Strength**
  - ➢Efficiency: By eliminating the value network, GRPO reduces memory and computational costs.
  - ➢Stability: The group-based advantage estimation and KL divergence integration make training more stable.
  - ➢Scalability: GRPO is better suited for large-scale models like DeepSeek-V2 and V3, where resource efficiency is critical.

Comparison and Summary



Feature	PPO (RLHF)	DPO (Direct Preference Optimization)	GRPO (Group Relative Policy Optimization)
Reward Model	Explicitly trained neural network	Implicitly learned	Implicitly learned
Value Function (Critic)	Yes, separate neural network	No	No
Optimization	Reinforcement Learning (PPO Algorithm)	Direct Policy Optimization (Classification-like Loss)	Simplified RL (PPO-like, but with GRAE)
Training Loop Complexity	More complex (RL loop, critic training)	Simpler (no RL loop, direct optimization)	Simplified RL loop (group sampling, but no critic)
Advantage Estimation	GAE (using Value Function)	N/A (Implicit Reward)	GRAE (Group Relative Advantage Estimation)
Computational Efficiency	Lower (critic, RL sampling)	Higher (no critic, no RL sampling during training)	Higher than PPO (no critic), potentially similar to DPO
Implementation Simplicity	More complex	Simpler	Simpler than PPO, somewhat similar to DPO

Reference (1)

- [1] Fabio Petroni, Tim RocktÅNaschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2463–2473, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [2] Leyang Cui, Yu Wu, Jian Liu, Sen Yang, and Yue Zhang. Template-based named entity recognition using BART. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 1835–1845, Online, August 2021. Association for Computational Linguistics.
- [3] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4582–4597, Online, August 2021. Association for Computational Linguistics.
- [4] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [6] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How Can We Know What Language Models Know? Transactions of the Association for Computational Linguistics, 8:423–438, 07 2020.
- [7] Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing NLP. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2153–2162, Hong Kong, China, November 2019. Association for Computational Linguistics.

Reference (2)

- [8] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 3816–3830, Online, August 2021. Association for Computational Linguistics.
- [9] Joe Davison, Joshua Feldman, and Alexander Rush. Commonsense knowledge mining from pretrained models. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1173–1178, Hong Kong, China, November 2019. Association for Computational Linguistics.
- [10] Taylor Shin, Yasaman Razeghi, Robert L. Logan, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts, 2020.
- [11] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks, 2021.
- [12] Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5943–5959, Online, November 2020. Association for Computational Linguistics.
- [13] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.
- [14] Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3914–3923, Hong Kong, China, November 2019. Association for Computational Linguistics.