

# **Recent Trends of Large Language Models**

**Aiwei Liu    2023.06.09**

# Table of contents

- **Using Off-the-shell LLM**

- **Design better prompt**
- Decoding method
- Combine LLM with tools

- **Fine-tune LLM**

- How to fine-tune
- What to fine-tune
  - Instruct tuning
  - Other tuning

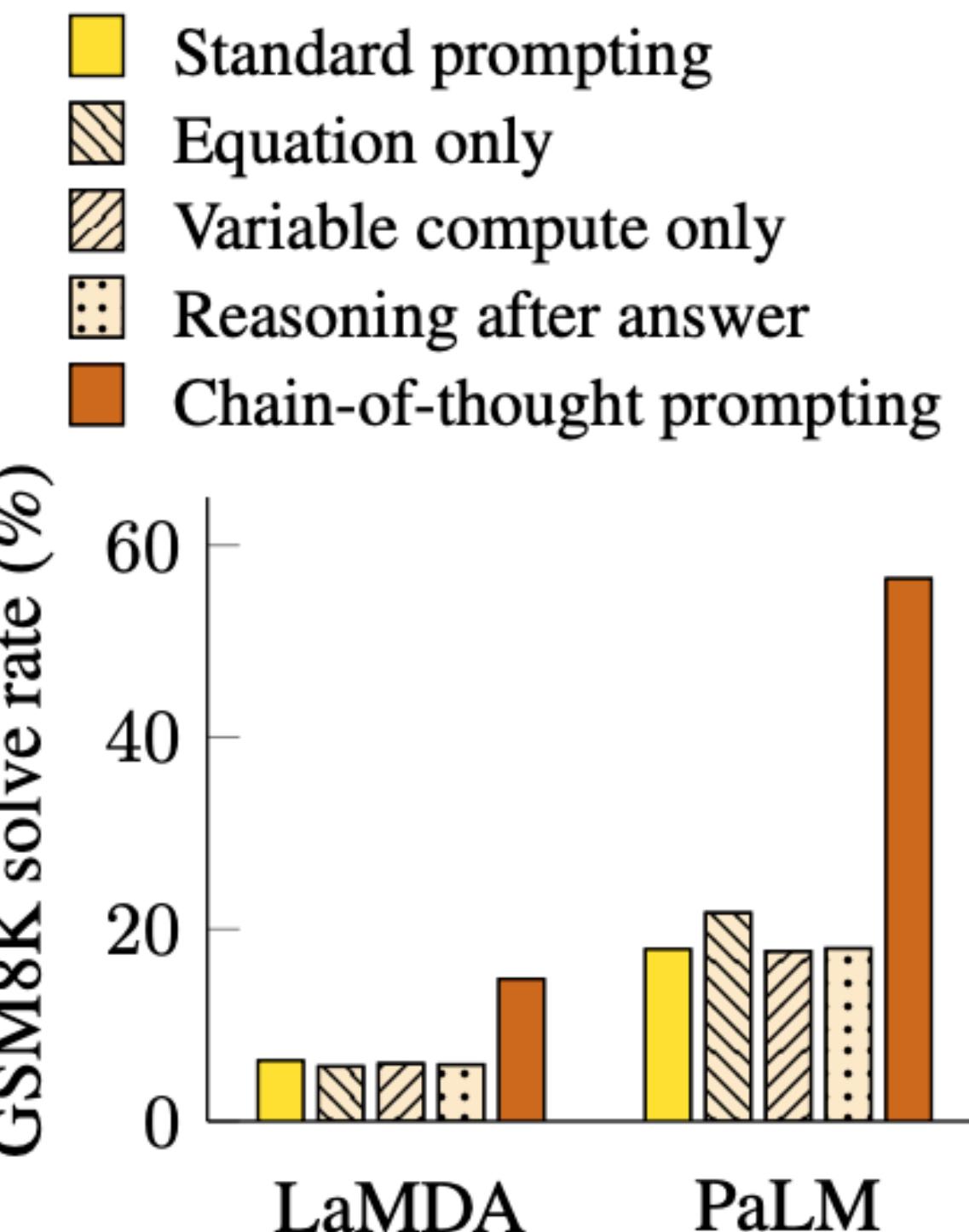
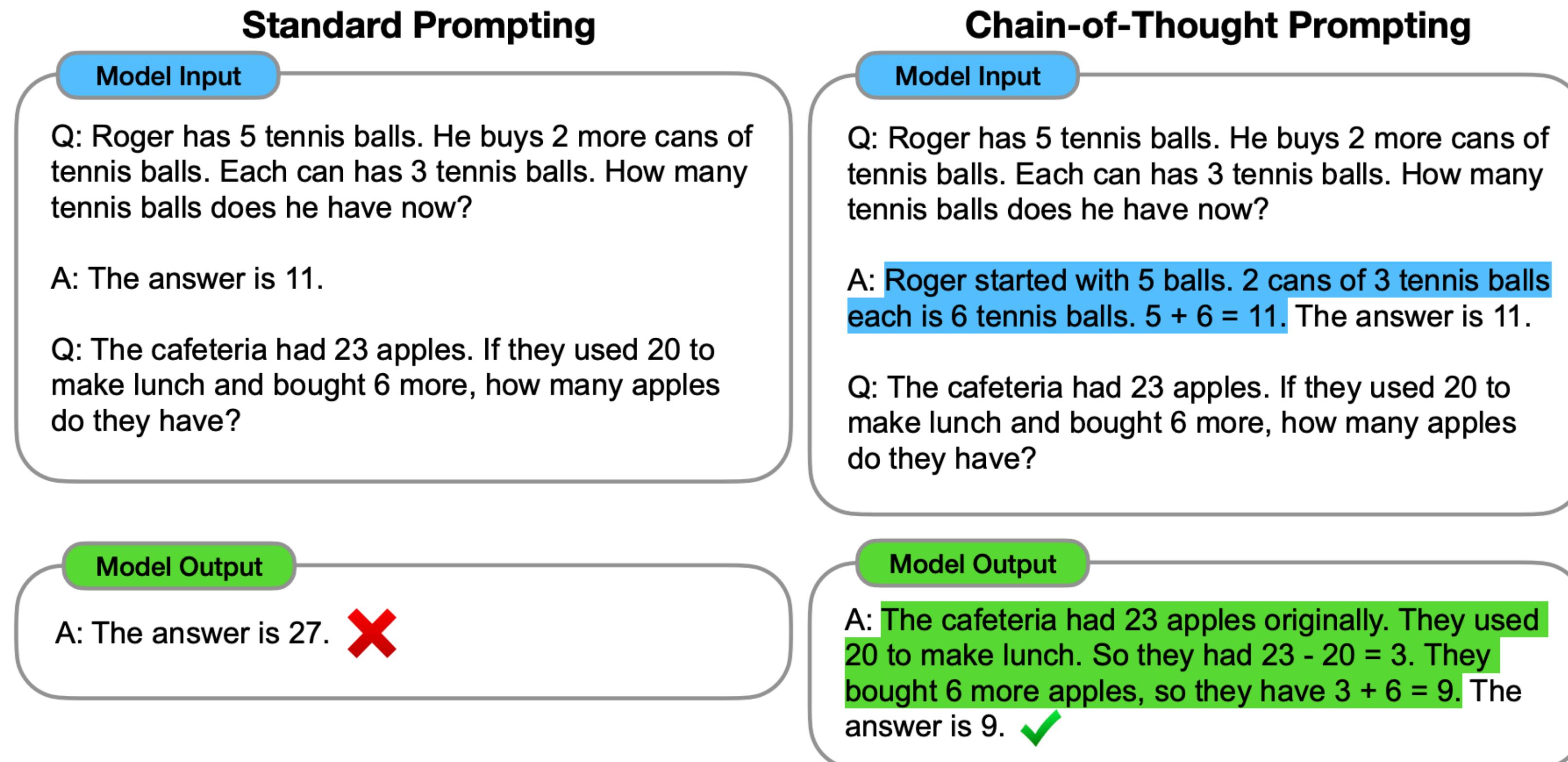
- **Other trends**

# Design better prompt

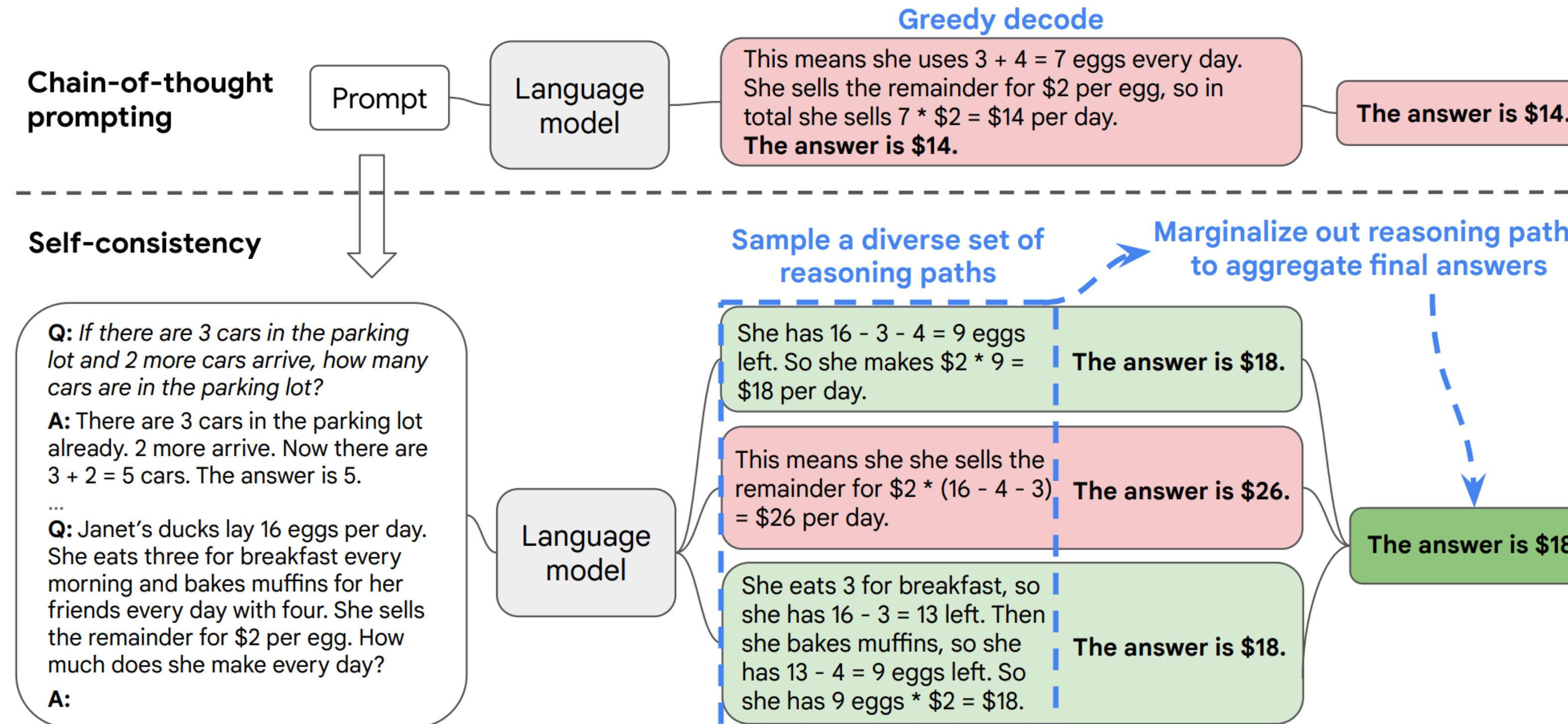
- **General principle**
  - Break down the task.
  - Put more information into the prompt.
  - Make the next token generation fault-tolerant
- **Relative works**
  - Chain-of-Thought Prompting Elicits Reasoning in Large Language Models
  - Self-Consistency improves chain-of-thought Reasoning in language models
  - Tree of Thoughts: Deliberate Problem Solving with Large Language Models
  - DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction

# Chain-of-Thought

- General principle
  - Break down the task.
  - Put more information into the prompt.
  - Make the next token generation fault-tolerant



# Self-Consistency Chain-of-Thought

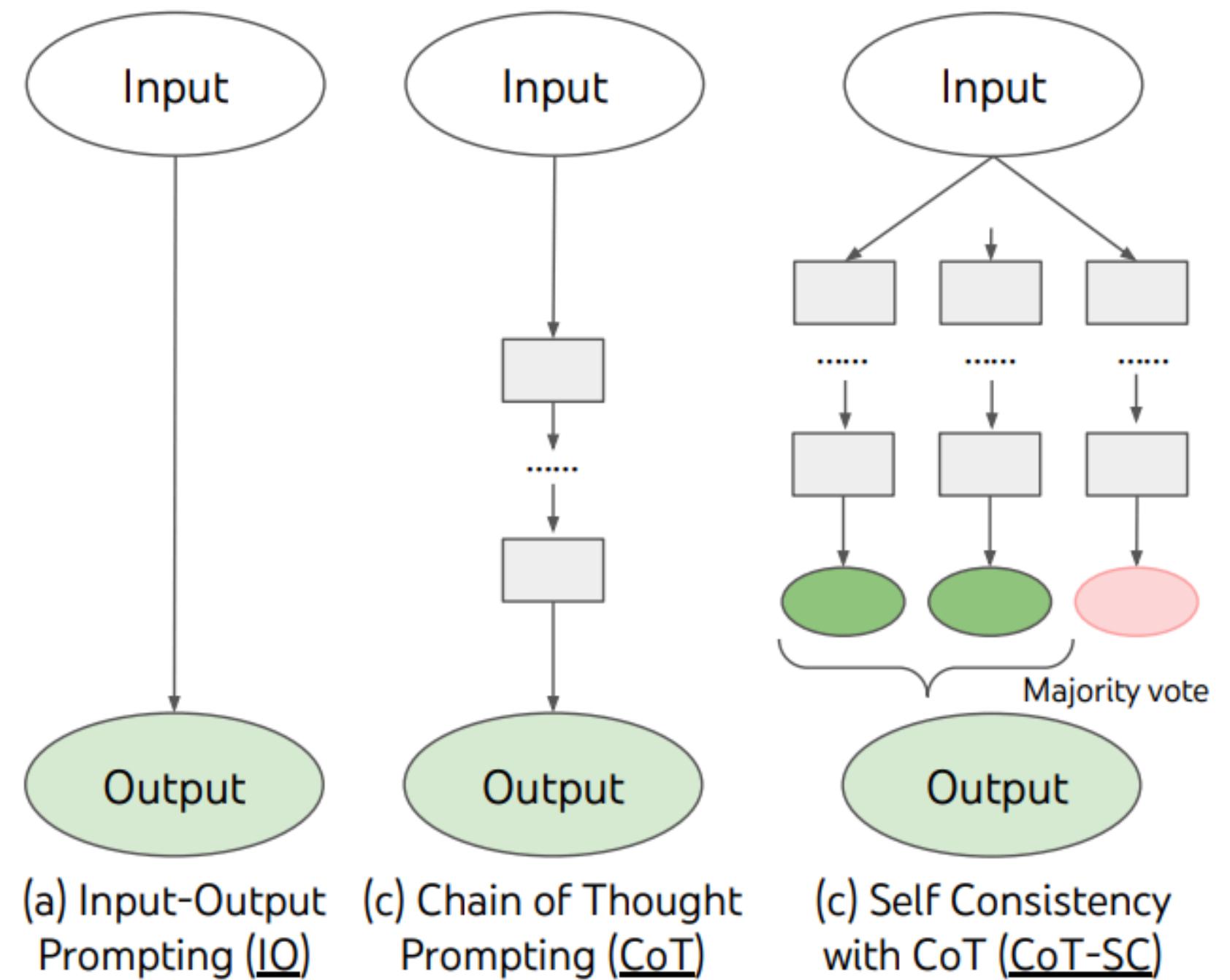


**Sample different chain-of-thought from model and vote for the final answer.**

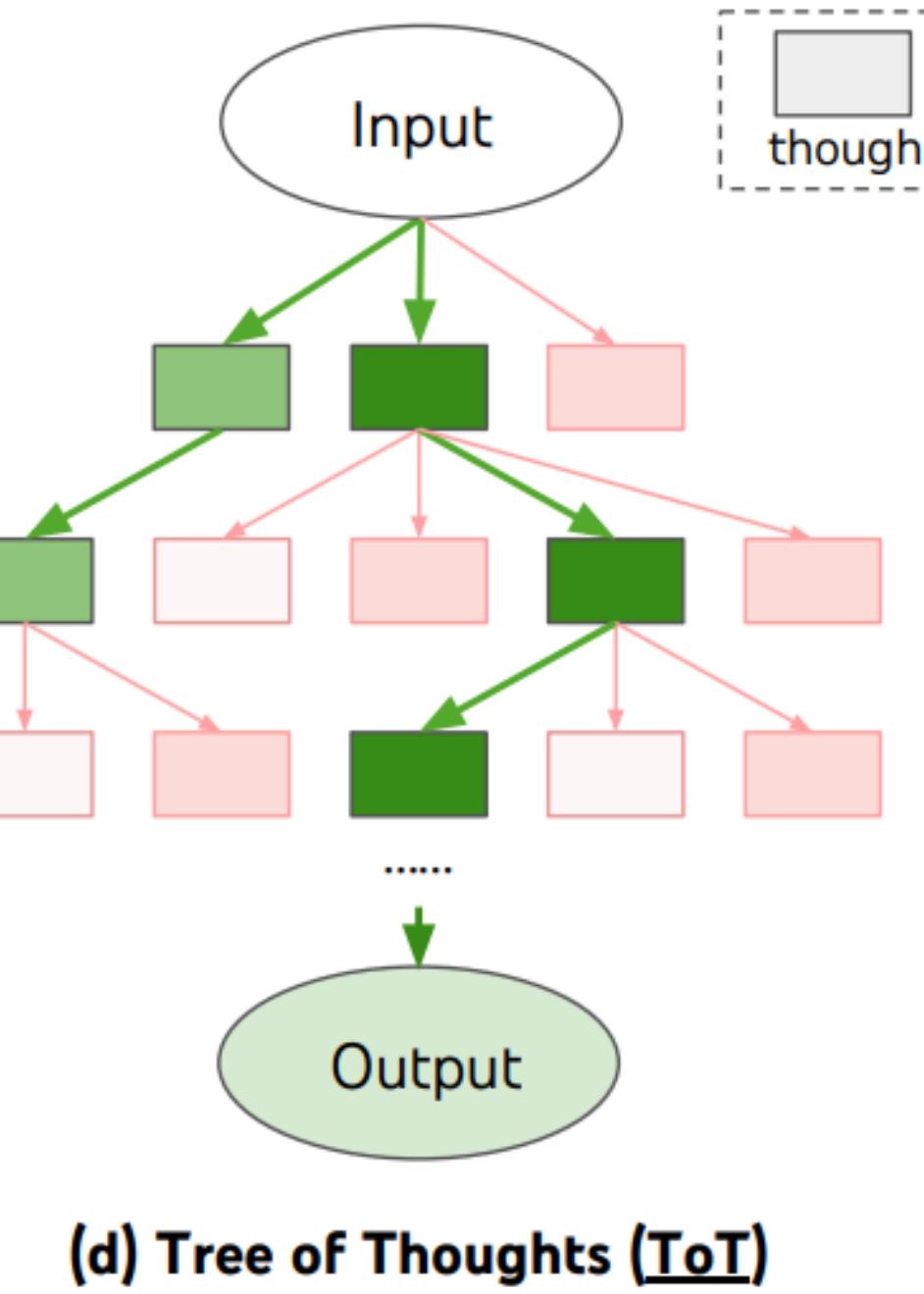
**Make the next token generation fault tolerant**

	ANLI R1 / R2 / R3	e-SNLI	RTE	BoolQ	HotpotQA (EM/F1)
Standard-prompting (no-rationale)	69.1 / 55.8 / 55.8	85.8	84.8	71.3	27.1 / 36.8
CoT-prompting (Wei et al., 2022)	68.8 / 58.9 / 60.6	81.0	79.1	74.2	28.9 / 39.8
Self-consistency	<b>78.5 / 64.5 / 63.4</b>	<b>88.4</b>	<b>86.3</b>	<b>78.4</b>	<b>33.8 / 44.6</b>

# Tree-of-Thought



**Make the next token generation fault tolerant**



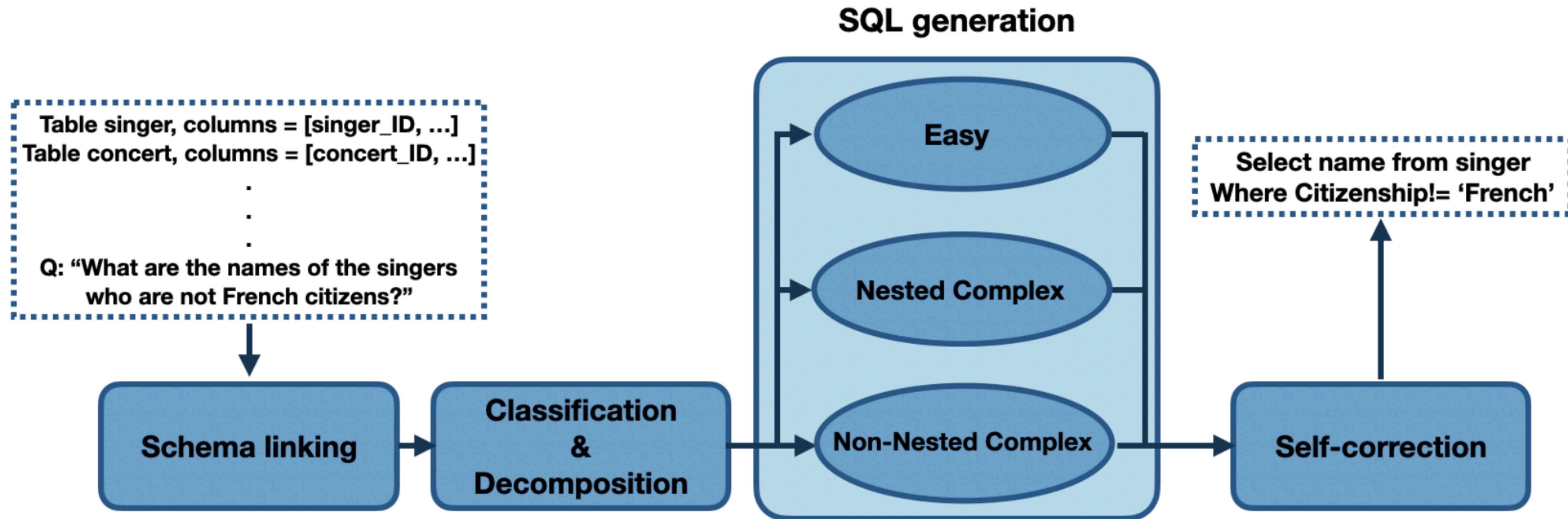
**Require prompt design by human-expert**

```

# 1-shot
propose_prompt = '''Input: 2 8 8 14
Possible next steps:
2 + 8 = 10 (left: 8 10 14)
8 / 2 = 4 (left: 4 8 14)
14 + 2 = 16 (left: 8 8 16)
2 * 8 = 16 (left: 8 14 16)
8 - 2 = 6 (left: 6 8 14)
14 - 8 = 6 (left: 2 6 8)
14 / 2 = 7 (left: 7 8 8)
14 - 2 = 12 (left: 8 8 12)
Input: {input}
Possible next steps:
...

```

Method	Success
IO prompt	7.3%
CoT prompt	4.0%
CoT-SC (k=100)	9.0%
ToT (ours) (b=1)	45%
ToT (ours) (b=5)	74%
IO + Refine (k=10)	27%
IO (best of 100)	33%
CoT (best of 100)	49%



**Break down the complex Text-to-SQL task into four simple tasks and design prompts.**

## Step1: Schema-linking

# Find the schema\_links for generating SQL queries for each question based on the database schema and Foreign keys.

Table city, columns = [\*,City\_ID,Official\_Name,Status,Area\_km\_2,Population,Census\_Ranking]

Table competition\_record, columns = [\*,Competition\_ID,Farm\_ID,Rank]

Table farm, columns = [\*,Farm\_ID,Year,Total\_Horses,Working\_Horses,

Total\_Cattle,Oxen,Bulls,Cows,Pigs,Sheep\_and\_Goats]

Table farm\_competition, columns = [\*,Competition\_ID,Year,Theme,Host\_city\_ID,Hosts]

Foreign\_keys = [farm\_competition.Host\_city\_ID = city.City\_ID,competition\_record.Farm\_ID = farm.Farm\_ID,competition\_record.Competition\_ID = farm\_competition.Competition\_ID]

**Q:** "Show the status of the city that has hosted the greatest number of competitions."

**A:** Let's think step by step. In the question "Show the status of the city that has hosted the greatest number of competitions.", we are asked:

"the status of the city" so we need column = [city.Status]

"greatest number of competitions" so we need column = [farm\_competition.\*]

Based on the columns and tables, we need these Foreign\_keys = [farm\_competition.Host\_city\_ID = city.City\_ID].

Based on the tables, columns, and Foreign\_keys, The set of possible cell values are = []. So the Schema\_links are:

**Schema\_links:** [city.Status,farm\_competition.Host\_city\_ID = city.City\_ID,farm\_competition.\*]

Table department, columns = [\*,Department\_ID,Name,Creation,Ranking,Budget\_in\_Billions,Num\_Employees]

Table head, columns = [\*,head\_ID,name,born\_state,age]

Table management, columns = [\*,department\_ID,head\_ID,temporary\_acting]

Foreign\_keys = [management.head\_ID = head.head\_ID,management.department\_ID = department.Department\_ID]

**Q:** "How many heads of the departments are older than 56 ?"

**A:** Let's think step by step. In the question "How many heads of the departments are older than 56?", we are asked:

"How many heads of the departments" so we need column = [head.\*]

"older" so we need column = [head.age]

Based on the columns and tables, we need these Foreign\_keys = [].

Based on the tables, columns, and Foreign\_keys, The set of possible cell values are = [56]. So the Schema\_links are:

**Schema\_links:** [head.\* ,head.age,56]

# DIN-SQL

## Step2: Classification & decomposition prompt

# For the given question, classify it as EASY, NON-NESTED, or NESTED based on nested queries and JOIN.

if need nested queries: predict NESTED

elif need JOIN and don't need nested queries: predict NON-NESTED

elif don't need JOIN and don't need nested queries: predict EASY

Table advisor, columns = [\*,s\_ID,i\_ID]

Table classroom, columns = [\*,building,room\_number,capacity]

Table course, columns = [\*,course\_id,title,dept\_name,credits]

Table department, columns = [\*,dept\_name,building,budget]

Table instructor, columns = [\*,ID,name,dept\_name,salary]

Table prereq, columns = [\*,course\_id,prereq\_id]

Table section, columns = [\*,course\_id,sec\_id,semester,year,building,room\_number,time\_slot\_id]

Table student, columns = [\*,ID,name,dept\_name,tot\_cred]

Table takes, columns = [\*,ID,course\_id,sec\_id,semester,year,grade]

Table teaches, columns = [\*,ID,course\_id,sec\_id,semester,year]

Table time\_slot, columns = [\*,time\_slot\_id,day,start\_hr,start\_min,end\_hr,end\_min]

Foreign\_keys = [course.dept\_name = department.dept\_name,instructor.dept\_name = department.dept\_name,section.building = classroom.building,section.room\_number = classroom.room\_number,section.course\_id = course.course\_id,teaches.ID = instructor.ID,teaches.course\_id = section.course\_id,teaches.sec\_id = section.sec\_id,teaches.semester = section.semester,teaches.year = section.year,student.dept\_name = department.dept\_name,takes.ID = student.ID,takes.course\_id = section.course\_id,takes.sec\_id = section.sec\_id,takes.semester = section.semester,takes.year = section.year,advisor.s\_ID = student.ID,

advisor.i\_ID = instructor.ID,prereq.prereq\_id = course.course\_id,prereq.course\_id = course.course\_id]

Q: "Find the buildings which have rooms with capacity more than 50."

schema\_links: [classroom.building,classroom.capacity,50]

A: Let's think step by step. The SQL query for the question "Find the buildings which have rooms with capacity more than 50." needs these tables = [classroom], so we don't need JOIN.

Plus, it doesn't require nested queries with (INTERSECT, UNION, EXCEPT, IN, NOT IN), and we need the answer to the questions = [""].

So, we don't need JOIN and don't need nested queries, then the the SQL query can be classified as "EASY".

Label: "EASY"

Q: "What are the names of all instructors who advise students in the math depart sorted by total credits of the student."

schema\_links: [advisor.i\_id = instructor.id,advisor.s\_id = student.id,instructor.name, student.dept\_name,student.tot\_cred,math]

A: Let's think step by step. The SQL query for the question "What are the names of all instructors who advise students in the math depart sorted by total credits of the student." needs these tables = [advisor,instructor,student], so we need JOIN.

Plus, it doesn't need nested queries with (INTERSECT, UNION, EXCEPT, IN, NOT IN), and we need the answer to the questions = [""].

So, we need JOIN and don't need nested queries, then the the SQL query can be classified as "NON-NESTED".

Label: "NON-NESTED"

# DIN-SQL

## Step3: SQL generation(Easy)

# Use the the schema links to generate the SQL queries for each of the questions.

Table advisor, columns = [\*,s\_ID,i\_ID]

Table classroom, columns = [\*,building,room\_number,capacity]

Table course, columns = [\*,course\_id,title,dept\_name,credits]

Table department, columns = [\*,dept\_name,building,budget]

Table instructor, columns = [\*,ID,name,dept\_name,salary]

Table prereq, columns = [\*,course\_id,prereq\_id]

Table section, columns = [\*,course\_id,sec\_id,semester,year,building,room\_number,time\_slot\_id]

Table student, columns = [\*,ID,name,dept\_name,tot\_cred]

Table takes, columns = [\*,ID,course\_id,sec\_id,semester,year,grade]

Table teaches, columns = [\*,ID,course\_id,sec\_id,semester,year]

Table time\_slot, columns = [\*,time\_slot\_id,day,start\_hr,start\_min,end\_hr,end\_min]

**Q:** "Find the buildings which have rooms with capacity more than 50."

**Schema\_links:** [classroom.building,classroom.capacity,50]

**SQL:** `SELECT DISTINCT building FROM classroom WHERE capacity > 50`

**Q:** "Find the room number of the rooms which can sit 50 to 100 students and their buildings."

**Schema\_links:** [classroom.building,classroom.room\_number,classroom.capacity,50,100]

**SQL:** `SELECT building , room_number FROM classroom WHERE capacity BETWEEN 50 AND 100`

## Step3: SQL generation(Non-Nested Complex)

```
# Use the the schema links and Intermediate_representation to generate the SQL queries for each of the
questions.

Table advisor, columns = [*,s_ID,i_ID]
Table classroom, columns = [*,building,room_number,capacity]
Table course, columns = [*,course_id,title,dept_name,credits]
Table department, columns = [*,dept_name,building,budget]
Table instructor, columns = [*,ID,name,dept_name,salary]
Table prereq, columns = [*,course_id,prereq_id]
Table section, columns = [*,course_id,sec_id,semester,year,building,room_number,time_slot_id]
Table student, columns = [*,ID,name,dept_name,tot_cred]
Table takes, columns = [*,ID,course_id,sec_id,semester,year,grade]
Table teaches, columns = [*,ID,course_id,sec_id,semester,year]
Table time_slot, columns = [*,time_slot_id,day,start_hr,start_min,end_hr,end_min]
Foreign_keys = [course.dept_name = department.dept_name,instructor.dept_name = department.dept_name,section.building = classroom.building,
section.room_number = classroom.room_number,section.course_id = course.course_id,teaches.ID = instructor.ID,teaches.course_id = section.course_id,
```

```
teaches.sec_id = section.sec_id,teaches.semester = section.semester,teaches.year = section.year,
student.dept_name = department.dept_name,takes.ID = student.ID,takes.course_id = section.course_id,takes.sec_id = section.sec_id,takes.semester = section.semester,
takes.year = section.year,advisor.s_ID = student.ID,advisor.i_ID = instructor.ID,prereq.prereq_id = course.course_id,
prereq.course_id = course.course_id]
```

**Q:** "Find the total budgets of the Marketing or Finance department."  
**Schema\_links:** [department.budget,department.dept\_name,Marketing,Finance]  
**A:** Let's think step by step. For creating the SQL for the given question, we need to join these tables = []. First, create an intermediate representation, then use it to construct the SQL query.  
**Intermediate\_representation:** select sum(department.budget) from department where department.dept\_name = "Marketing" or department.dept\_name = "Finance"  
**SQL:** `SELECT sum(budget) FROM department WHERE dept_name = 'Marketing' OR dept_name = 'Finance'`

**Q:** "Find the name and building of the department with the highest budget."  
**Schema\_links:** [department.budget,department.dept\_name,department.building]  
**A:** Let's think step by step. For creating the SQL for the given question, we need to join these tables = []. First, create an intermediate representation, then use it to construct the SQL query.  
**Intermediate\_representation:** select department.dept\_name , department.building from department order by department.budget desc limit 1  
**SQL:** `SELECT dept_name , building FROM department ORDER BY budget DESC LIMIT 1`

## Step3: SQL generation(Nested Complex)

# Use the intermediate representation and the schema links to generate the SQL queries for each of the questions.

Table advisor, columns = [\*,s\_ID,i\_ID]  
Table classroom, columns = [\*,building,room\_number,capacity]  
Table course, columns = [\*,course\_id,title,dept\_name,credits]  
Table department, columns = [\*,dept\_name,building,budget]  
Table instructor, columns = [\*,ID,name,dept\_name,salary]  
Table prereq, columns = [\*,course\_id,prereq\_id]  
Table section, columns = [\*,course\_id,sec\_id,semester,year,building,room\_number,time\_slot\_id]  
Table student, columns = [\*,ID,name,dept\_name,tot\_cred]  
Table takes, columns = [\*,ID,course\_id,sec\_id,semester,year,grade]  
Table teaches, columns = [\*,ID,course\_id,sec\_id,semester,year]  
Table time\_slot, columns = [\*,time\_slot\_id,day,start\_hr,start\_min,end\_hr,end\_min]  
Foreign\_keys = [course.dept\_name = department.dept\_name,instructor.dept\_name = department.dept\_name,section.building = classroom.building,section.room\_number = classroom.room\_number,section.course\_id = course.course\_id,teaches.ID = instructor.ID,teaches.course\_id = section.course\_id,teaches.sec\_id = section.sec\_id,teaches.semester = section.semester,teaches.year = section.year,student.dept\_name = department.dept\_name,takes.ID = student.ID,takes.course\_id = section.course\_id,takes.sec\_id = section.sec\_id,takes.semester = section.semester,takes.year = section.year,advisor.s\_ID = student.ID,advisor.i\_ID = instructor.ID,prereq.prereq\_id = course.course\_id,prereq.course\_id = course.course\_id]

Q: "Find the title of courses that have two prerequisites?"

Schema\_links: [course.title, course.course\_id = prereq.course\_id]

A: Let's think step by step. "Find the title of courses that have two prerequisites?" can be solved by knowing the answer to the following sub-question "What are the titles for courses with two prerequisites?". The SQL query for the sub-question "What are the titles for courses with two prerequisites?" is `SELECT T1.title FROM course AS T1 JOIN prereq AS T2 ON T1.course_id = T2.course_id GROUP BY T2.course_id HAVING count(*) = 2`

So, the answer to the question "Find the title of courses that have two prerequisites?" is =

Intermediate\_representation: `select course.title from course where count ( prereq.* ) = 2 group by prereq.course_id`

SQL: `SELECT T1.title FROM course AS T1 JOIN prereq AS T2 ON T1.course_id = T2.course_id GROUP BY T2.course_id HAVING count(*) = 2`

Q: "Find the name and building of the department with the highest budget."

Schema\_links: [department.dept\_name,department.building,department.budget]

A: Let's think step by step. "Find the name and building of the department with the highest budget." can be solved by knowing the answer to the following sub-question "What is the department name and corresponding building for the department with the greatest budget?".

The SQL query for the sub-question "What is the department name and corresponding building for the department with the greatest budget?" is `SELECT dept_name , building FROM department ORDER BY budget DESC LIMIT 1`

So, the answer to the question "Find the name and building of the department with the highest budget." is =

Intermediate\_representation: `select department.dept_name , department.building from department order by department.budget desc limit 1`

SQL: `SELECT dept_name , building FROM department ORDER BY budget DESC LIMIT 1`

# DIN-SQL

## Step4: self-correction prompt

```
#### For the given question, use the provided tables, columns, foreign_keys, and primary keys to fix the given SQLite SQL QUERY for any issues. If there are any problems, fix them. If there are no issues, return SQLite SQL QUERY as is.
```

```
#### Use the following instructions for fixing the SQL query:
```

- 1) Use the database values that are explicitly mentioned in the question
- 2) Pay attention to the columns that are used for the JOIN by using the Foreign\_keys.
- 3) Use DESC and DISTINCT when needed
- 4) Pay attention to the columns that are used for the GROUP BY clause.
- 5) Pay attention to the columns that are used for the SELECT clause.
- 6) Only change the GROUP BY clause when necessary.

```
Tables concert, columns = [concert_ID, ...]
```

```
...
```

```
Foreign_keys = [concert.Stadium_ID = stadium.Stadium_ID, ...]
```

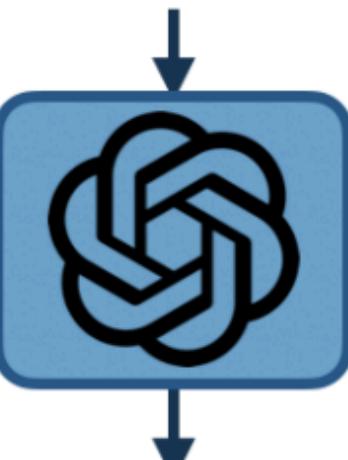
```
Primary_key = [stadium.Stadium_ID, ...]
```

```
#### Question: What is the name and capacity for the stadium with highest average attendance?
```

```
#### SQLite SQL Query
```

```
SELECT Name , Capacity FROM stadium ORDER BY Average LIMIT 1
```

```
#### Fixed SQL QUERY
```



```
SELECT Name , Capacity FROM stadium ORDER BY Average DESC LIMIT 1
```

Prompting	Model	Exec Acc	Exact match Acc
DIN-SQL (Ours)	GPT-4	74.2	60.1
	CodeX Davinci	69.9	57.2
	CodeX Cushman	47.6	35.7
Few-shot (Ours)	GPT-4	67.4	54.3
	CodeX Davinci	61.5	50.2
	CodeX Cushman	43.1	30.9
Zero-shot (Ours)	GPT-4	64.9	40.4
Zero-shot (Liu et al., 2023a)	ChatGPT	60.1	-
Zero-shot (Rajkumar et al., 2022)	CodeX Davinci	47.5	
Zero-shot (DB content used) (Rajkumar et al., 2022)	CodeX Davinci	55.1	
	CodeX Cushman	53	
	GPT3	21.7	

- Break down the task.
- Put more information into the prompt.
- Make the next token generation fault-tolerant

# Table of contents

- **Using Off-the-shell LLM**
  - Design better prompt
  - **Decoding method**
  - Combine LLM with tools
- **Fine-tune LLM**
  - How to fine-tune
  - What to fine-tune
    - Instruct tuning
    - Other tuning
- **Other trends**

# Design decoding method for LLMs

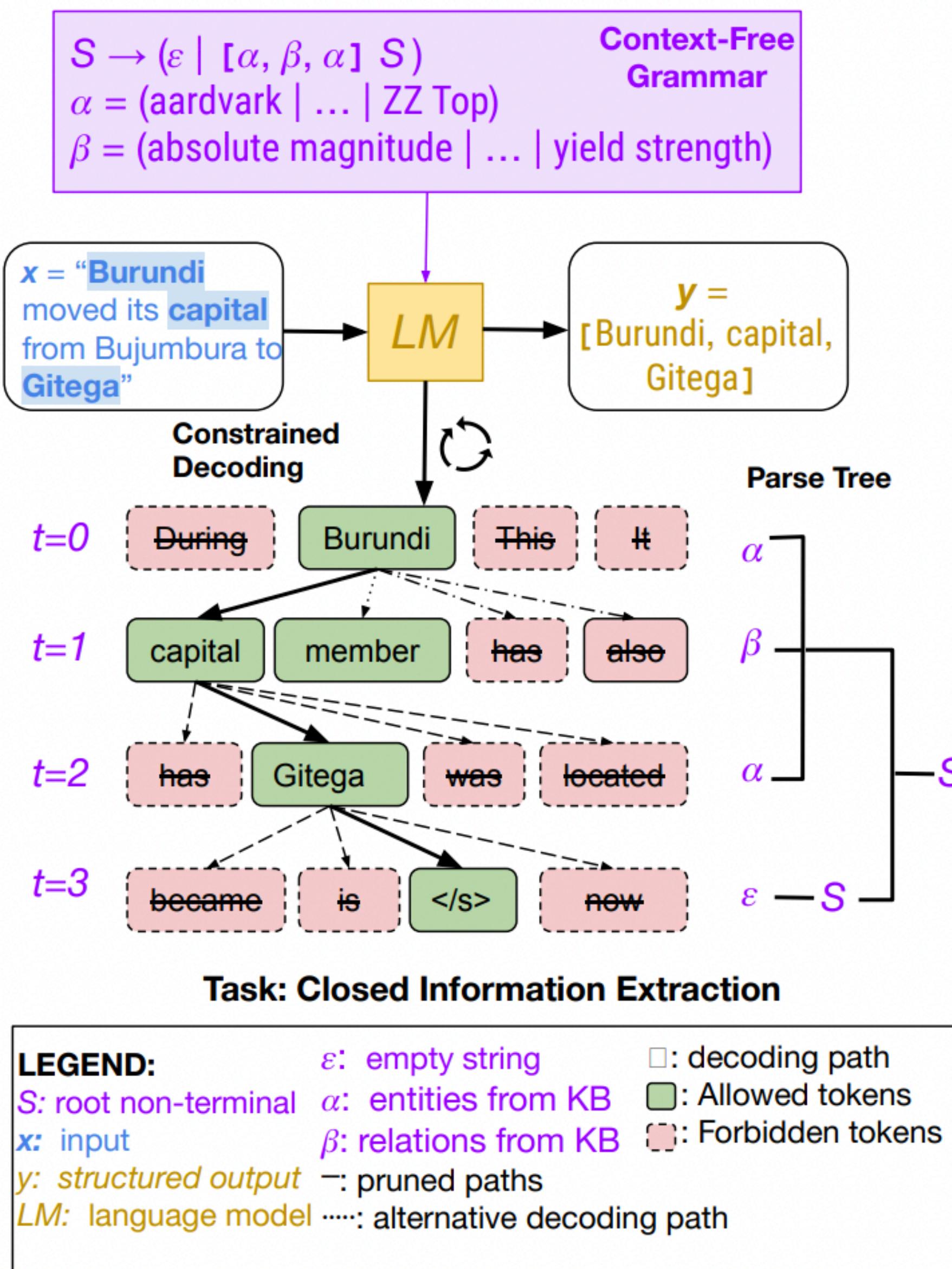
- **Better decoding method could improve the generated text without training**
  - Greedy search
  - Beam search
  - Sampling
- **Relative works**
  - Flexible Grammar-Based Constrained Decoding for Language Models
  - Look-back Decoding for Open-Ended Text Generation

# Grammar-Based Constrained Decoding

Reliably generating complex output structures for LLM could be hard.

Method	SynthIE-text		
	Precision	Recall	F1
<b>Supervised</b>			
GenIE T5-base (Josifoski et al., 2022)	<b>49.1</b>	26.7	<b>34.6</b>
<b>Few-shot Unconstrained</b>			
LLaMA-7B (2 shots)	1.8	1.9	1.9
LLaMA-13B (2 shots)	3.8	2.5	3.1
LLaMA-33B (2 shots)	7.4	11.2	9.1
LLaMA-33B (4 shots)	<b>9.0</b>	<b>19.9</b>	<b>12.4</b>
<b>Few-shot Constrained</b>			
LLaMA-7B (2 shots)	33.2	17.6	23.0
LLaMA-13B (2 shots)	29.9	17.2	21.8
LLaMA-33B (2 shots)	<b>36.2</b>	28.9	32.1
LLaMA-33B (4 shots)	35.0	<b>33.8</b>	<b>34.4</b>

## Information Extraction Task



Also, constrained decoding could be a potential basic for **decoder prompt**.

# Look-back Decoding

## More focus on the duplication problem during generation.

Degeneration LM (Decoding) Continuation

**Prefix:** In addition to live broadcasts FIFA Fan Fests offer food and beverages, merchandise and various entertainment events by local and international artists. The start of 2006 World Cup was

**Repetitive Continuation**  
S1: (GPT2-XL greedy) marked by the FIFA Fan Fest in Sao Paulo, Brazil.\n\n *The FIFA Fan Fest is a unique event that brings together fans from all over the world to celebrate the FIFA World Cup* . *The FIFA Fan Fest is a unique event that brings together fans from...*

S2: (davinci-002 greedy) celebrated with a concert by Bon Jovi in Berlin.\n\n *The FIFA Fan Fest in* Munich *was located at the* Olympic Park.\n\n *The FIFA Fan Fest in* Frankfurt *was located at the* Römerberg.\n\n *The FIFA Fan Fest in* Hamburg *was located at the* ...

**Prefix:** Burkan died from an attack of acute indigestion at his country home in Great Neck on June 6, 1936. Three thousand people attended his funeral at

**Off-Topic Continuation**  
S3: (ada-001 greedy) the local church.\n\n *The authorities are investigating how Khedira managed to enter the house and what role he played in the attack.*

S4: (davinci-002 greedy): Temple Emanu-El in New York City... *Category:1868 births\nCategory:1936 deaths\nCategory:Astro-...*

S5: (ada-001 nucleus): aients home.\n\n *( The Lorraine weekend\nIn house of intensity and occupation, great law enforcement officers\n... Shanny Bankeived his way into the home of Runaan U Without giving any reason other than to marines and punch said home's door ... )* ×2

## Use KL divergence to compare the current logits and previous generated logits

If same: use **sampling decoding** to introduce diversity

Else: use **greedy search** as usual

Look back for **other scenarios** can also be explored and developed

# Look-back Decoding

MeZO 的工作原理基于零阶 (Zeroth-order, ZO) 优化方法，这种方法可以通过只使用两次前向传播来估计梯度，但在优化大型模型时，理论上可能会非常慢。MeZO 通过改进经典的 ZO-SGD 方法，使其能够在原地 (in-place) 操作，从而实现了与推理相同的内存占用。例如，使用单个 A100 80GB GPU，MeZO 可以训练一个拥有 300 亿参数的模型，而使用相同预算的反向传播微调只能训练一个拥有 27 亿参数的语言模型。

研究团队对各种模型类型（包括掩码和自回归语言模型）、模型规模（高达 660 亿）以及下游任务（包括分类、多项选择和生成）进行了全面的实验。实验结果显示，MeZO 显著优于上下文学习和线性探测，且在多个任务中，其性能与反向传播微调相当，同时内存占用减少了高达 12 倍。

详细介绍 MeZO 的工作原理：

1. 在每一步，首先随机采样一个种子  $s$ ，然后对  $z$  的四个用途中的每一个，都通过  $s$  重置随机数生成器并重新采样  $z$  的相关条目。使用这种原地实现，MeZO 的内存占用与推理内存成本相当。
2. MeZO 还可以与其他基于梯度的优化器结合使用，包括带动量的 SGD 或 Adam。尽管朴素实现需要额外的内存来存储梯度动量估计，但 MeZO-momentum 和 MeZO-Adam 通过使用保存的传递损失和  $z$  来重新计算梯度的移动平均，从而减轻了这种开销。
3. MeZO 还可以优化非可微目标，如精度或 F1 分数，同时仍然只需要与推理相同的内存。
4. 通过理论分析，研究团队发现，充足的预训练可以确保 MeZO 的每步优化率和全局收敛率依赖于景观的某种条件数（即，局部有效秩），而不是参数的数量。这一结果与现有的 ZO 下界形成了鲜明的对比这篇文章的标题是 "Fine-Tuning Language Models with Just Forward Passes"，由 Princeton University 的研究团队撰写。文章主要介绍了一种名为 MeZO (Memory-efficient Zeroth-order Optimizer) 的新型优化器，该优化器能够有效地微调大型语言模型，同时只需要与推理相同的内存占用。

MeZO 的工作原理基于零阶 (Zeroth-order, ZO) 优化方法，这种方法可以通过只使用两次前向传播来估计梯度，但在优化大型模型时，理论上可能会非常慢。MeZO 通过改进经典的 ZO-SGD 方法，使其能够在原地 (in-place) 操作，从而实现了与推理相同的内存占用。例如，使用单个 A100 80GB GPU，MeZO 可以训练一个拥有 300 亿参数的模型，而使用相同预算的反向传播微调只能训练一个拥有 27 亿参数的语言模型。

研究团队对各种模型类型（包括掩码和自回归语言模型）、模型规模（高达 660 亿）以及下

# Table of contents

- **Using Off-the-shell LLM**

- Design better prompt
- Decoding method
- **Combine LLM with tools**

- **Fine-tune LLM**

- How to fine-tune
- What to fine-tune
  - Instruct tuning
  - Other tuning

- **Other trends**

# Combine LLM with tools

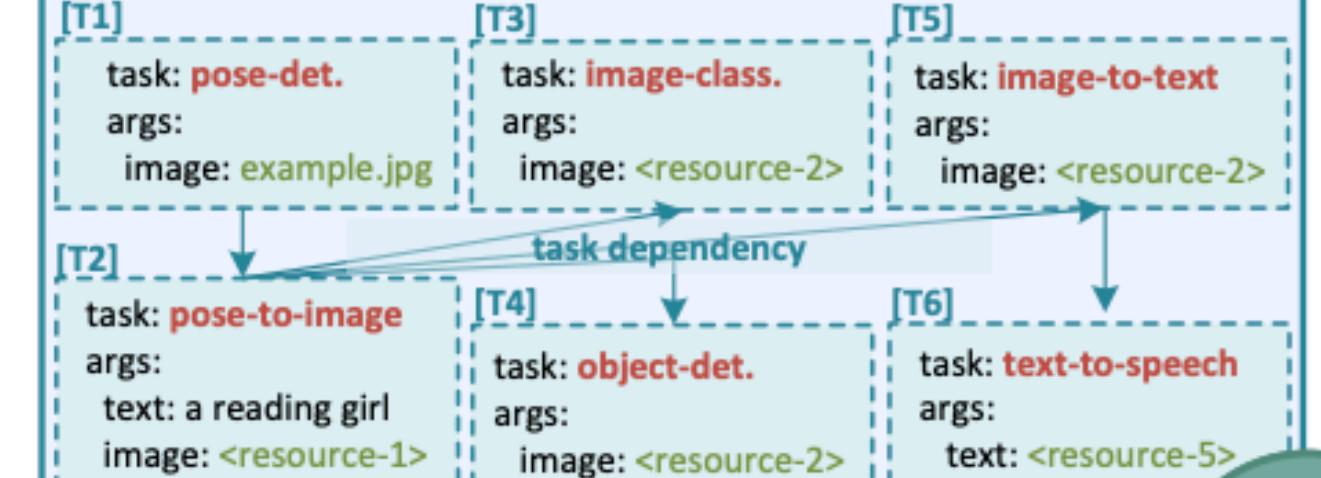
- **Two directions?**
  - Combine with as many tools as possible. (ChatGPT plugins)
  - **LLMs make tools on their own.**
- **Relative works**
  - HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face
  - Code Prompting: a Neural Symbolic Method for Complex Reasoning in Large Language Models

# HuggingGPT

Please generate an image where a girl is reading a book, and her pose is the same as the boy in the image example.jpg, then please describe the new image with your voice.

## Request

### Stage #1: Task Planning



### Stage #2: Model Selection

task: pose-det.

Query

task: object-det.

task: image-class.

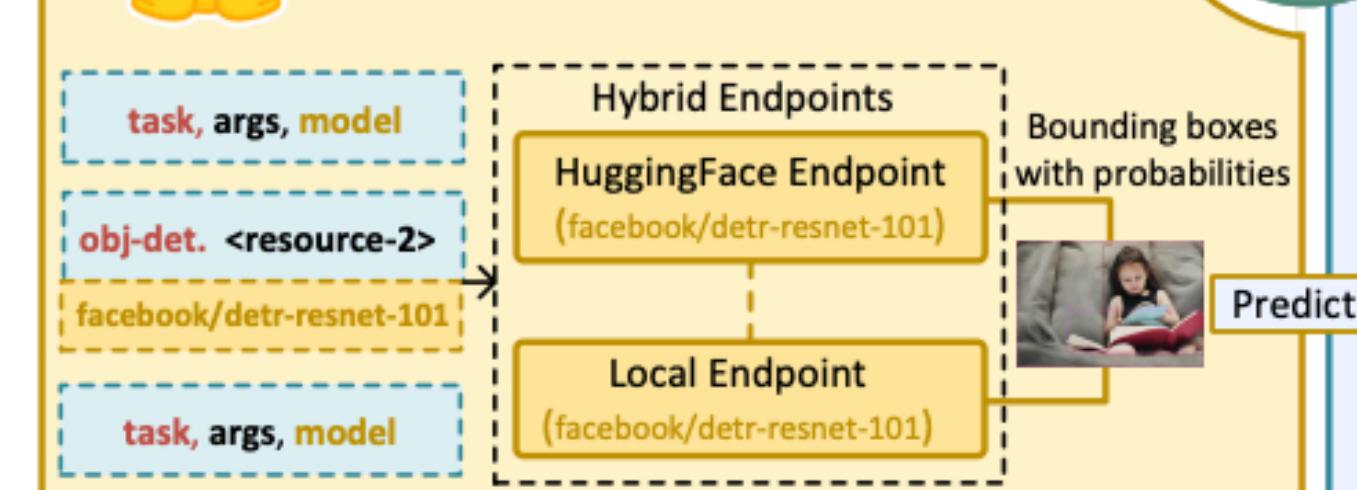
In-context task-model assignment:

- ✗ hustvl/yolos-tiny
- ✓ facebook/detr-resnet-101
- ✗ TahaDouaji/detr-doc-table-detection

Model Cards in HuggingFace



### Stage #3: Task Execution



### Stage #4: Response Generation

[T1] pose-det.

openpose [image-2]

[T2] pose-to-image

llyasviel/sd-ctl-pose [image-3]

[T3] image-class.

google/vit [classes with prob]

[T4] object-det.

facebook/detr-resnet [bounding box]

[T5] image-to-text

nlpconnect/vit-gpt2 "a girl sitting on ..."

[T6] text-to-speech

facebook/fastspeech [audio-1]



## Response



The image you gave me is of "boy". The first thing I did was use OpenCV's openpose model to analyze the pose of the boy in the image. I fed that image, specifically located at [Image-1], into the model to get an output image of the pose, located at [Image-2]. Then I take that pose and generate a new image, located at [Image-3], using llyasviel/sd-controlnet-openpose and the instruction "a girl is reading". Next, I ran object detection on the image generated in the previous step, to get predictions of the objects present in the image, including the box location. These predictions were run through the facebook/detr-resnet-101 model. The box was drawn on the image, which is located at [Image-4]. Then, I used the predictions of a image classification model google/vit-base-patch16-224 and a image caption model a nlpconnect/vit-gpt2-image-captionin model to generate the caption for newly generated image. It generated text: "a girl sitting on a bed reading a book". Lastly, I ran the facebook/fastspeech2-en-1jspeech model to convert the generated text into audio, located at [Audio-1].

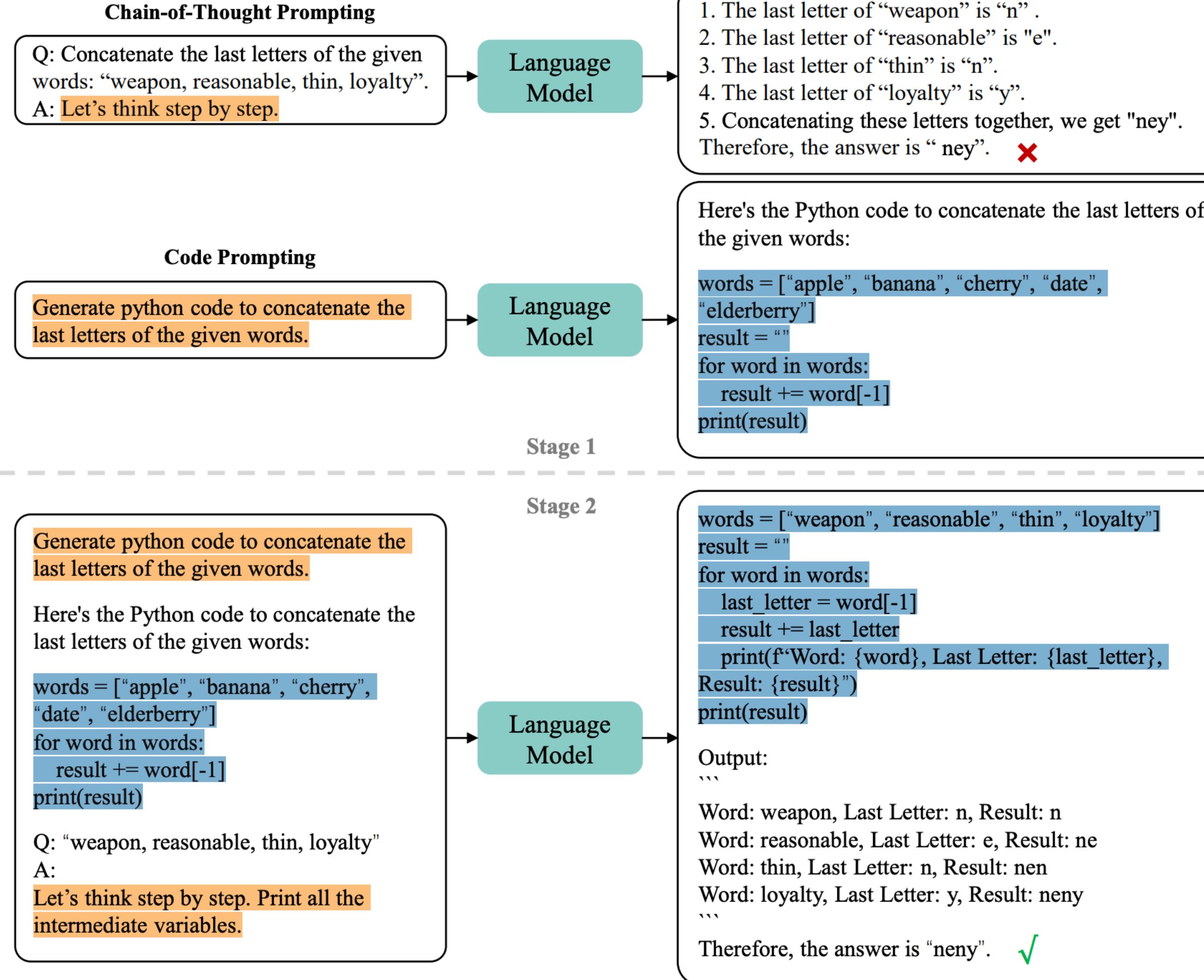
# HuggingGPT

## Task Planning

Prompt
<p>#1 Task Planning Stage - The AI assistant performs task parsing on user input, generating a list of tasks with the following format: <code>[{"task": task, "id": task_id, "dep": dependency_task_ids, "args": {"text": text, "image": URL, "audio": URL, "video": URL}}]</code>. The "dep" field denotes the id of the previous task which generates a new resource upon which the current task relies. The tag "<code>&lt;resource&gt;-task_id</code>" represents the generated text, image, audio, or video from the dependency task with the corresponding task_id. The task must be selected from the following options: <code>{} Available Task List {}</code>. Please note that there exists a logical connections and order between the tasks. In case the user input cannot be parsed, an empty JSON response should be provided. Here are several cases for your reference: <code>{} Demonstrations {}</code>. To assist with task planning, the chat history is available as <code>{} Chat Logs {}</code>, where you can trace the user-mentioned resources and incorporate them into the task planning stage.</p>
Demonstrations
Can you tell me how many objects in e1.jpg?  In e2.jpg, what's the animal and what's it doing?  First generate a HED image of e3.jpg, then based on the HED image and a text "a girl reading a book", create a new image as a response.
<pre>[{"task": "object-detection", "id": 0, "dep": [-1], "args": {"image": "e1.jpg"}]}  [{"task": "image-to-text", "id": 0, "dep": [-1], "args": {"image": "e2.jpg"}}, {"task": "image-cls", "id": 1, "dep": [-1], "args": {"image": "e2.jpg"}}, {"task": "object-detection", "id": 2, "dep": [-1], "args": {"image": "e2.jpg"}}, {"task": "visual-question-answering", "id": 3, "dep": [-1], "args": {"text": "what's the animal doing?", "image": "e2.jpg"}]}  [{"task": "pose-detection", "id": 0, "dep": [-1], "args": {"image": "e3.jpg"}}, {"task": "pose-text-to-image", "id": 1, "dep": [0], "args": {"text": "a girl reading a book", "image": "&lt;resource&gt;-0"}}]</pre>

Prompt
<p>#2 Model Selection Stage - Given the user request and the call command, the AI assistant helps the user to select a suitable model from a list of models to process the user request. The AI assistant merely outputs the model id of the most appropriate model. The output must be in a strict JSON format: <code>{"id": "id", "reason": "your detail reason for the choice"}</code>. We have a list of models for you to choose from <code>{} Candidate Models {}</code>. Please select one model from the list.</p>
Candidate Models
<pre>{"model_id": model id #1, "metadata": meta-info #1, "description": description of model #1} {"model_id": model id #2, "metadata": meta-info #2, "description": description of model #2} ... {"model_id": model id #K, "metadata": meta-info #K, "description": description of model #K}</pre>
Prompt
<p>#4 Response Generation Stage - With the input and the inference results, the AI assistant needs to describe the process and results. The previous stages can be formed as - User Input: <code>{} User Input {}</code>, Task Planning: <code>{} Tasks {}</code>, Model Selection: <code>{} Model Assignment {}</code>, Task Execution: <code>{} Predictions {}</code>. You must first answer the user's request in a straightforward manner. Then describe the task process and show your analysis and model inference results to the user in the first person. If inference results contain a file path, must tell the user the complete file path. If there is nothing in the results, please tell me you can't make it.</p>

# CodePrompt



**Make the models to make code tools themselves.**

	last letter concatenation			coin flip		
	l=4	l=8	l=12	#flips=3	#flips=4	#flips=5
zero-shot standard	7.4	2.0	2.6	22.6	17.2	16.8
zero-shot CoT	71.0	27.8	2.8	86.2	71.8	67.8
few-shot CoT	94.6	69.6	39.8	99.8	99.8	99.0
zero-shot code	97.2	85.4	75.6	86.2	88.8	85.8

# Table of contents

- **Using Off-the-shell LLM**

- Design better prompt
- Decoding method
- Combine LLM with tools

- **Fine-tune LLM**

- **How to fine-tune**
- What to fine-tune
  - Instruct tuning
  - Other methods

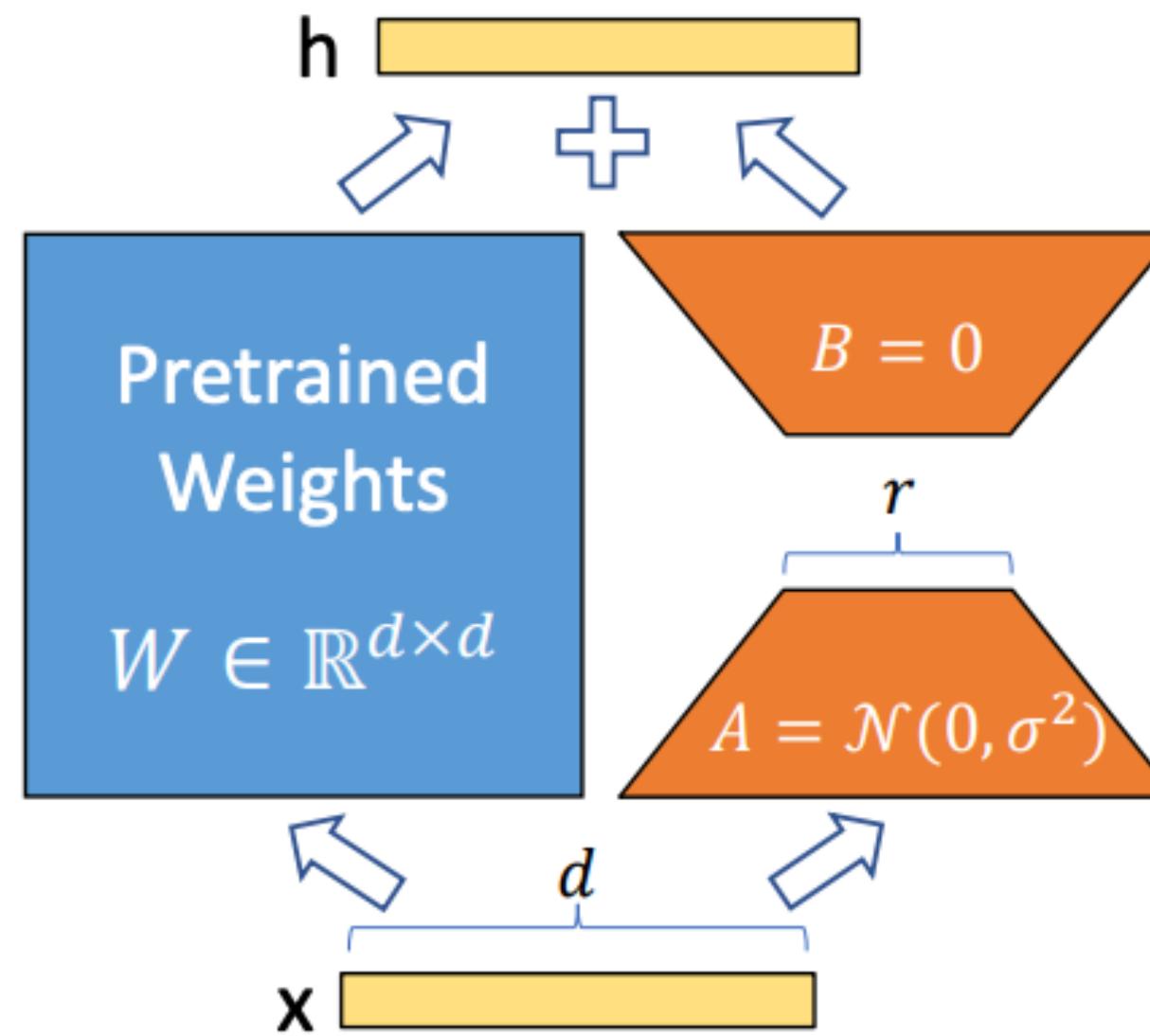
- **Other trends**

# How to fine-tune

**Main challenge: High cost of fine-tuning the model**

- **How to alleviate this problem**
  - Do not fine-tune the whole model, add additional adaptor parameters to fine-tune.
  - Using Zeroth-order Optimizer.
  - Using black-box method.
- **Relative works**
  - LORA: LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS
  - Fine-Tuning Language Models with Just Forward Passes
  - CombLM: Adapting Black-Box Language Models through Small Fine-Tuned Models

# LORA: LOW-RANK ADAPTATION OF LARGE LANGUAGE MODELS



Only update the  $\Delta W$  and use **low-rank composition**  $BA$  to to reduce the number of parameters

$$W_0 + \Delta W = W_0 + BA$$

$$h = W_0 x + \Delta W x = W_0 x + B A x$$

Model & Method	# Trainable Parameters	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
RoB <sub>base</sub> (FT)*	125.0M	<b>87.6</b>	94.8	90.2	<b>63.6</b>	92.8	<b>91.9</b>	78.7	91.2	86.4
RoB <sub>base</sub> (BitFit)*	0.1M	84.7	93.7	<b>92.7</b>	62.0	91.8	84.0	81.5	90.8	85.2
RoB <sub>base</sub> (Adpt <sup>D</sup> )*	0.3M	87.1 <sub>±.0</sub>	94.2 <sub>±.1</sub>	88.5 <sub>±1.1</sub>	60.8 <sub>±.4</sub>	93.1 <sub>±.1</sub>	90.2 <sub>±.0</sub>	71.5 <sub>±2.7</sub>	89.7 <sub>±.3</sub>	84.4
RoB <sub>base</sub> (Adpt <sup>D</sup> )*	0.9M	87.3 <sub>±.1</sub>	94.7 <sub>±.3</sub>	88.4 <sub>±.1</sub>	62.6 <sub>±.9</sub>	93.0 <sub>±.2</sub>	90.6 <sub>±.0</sub>	75.9 <sub>±2.2</sub>	90.3 <sub>±.1</sub>	85.4
RoB <sub>base</sub> (LoRA)	0.3M	87.5 <sub>±.3</sub>	<b>95.1<sub>±.2</sub></b>	89.7 <sub>±.7</sub>	63.4 <sub>±1.2</sub>	<b>93.3<sub>±.3</sub></b>	90.8 <sub>±.1</sub>	<b>86.6<sub>±.7</sub></b>	<b>91.5<sub>±.2</sub></b>	<b>87.2</b>
RoB <sub>large</sub> (FT)*	355.0M	90.2	<b>96.4</b>	<b>90.9</b>	68.0	94.7	<b>92.2</b>	86.6	92.4	88.9
RoB <sub>large</sub> (LoRA)	0.8M	<b>90.6<sub>±.2</sub></b>	96.2 <sub>±.5</sub>	90.9 <sub>±1.2</sub>	<b>68.2<sub>±1.9</sub></b>	<b>94.9<sub>±.3</sub></b>	91.6 <sub>±.1</sub>	<b>87.4<sub>±2.5</sub></b>	<b>92.6<sub>±.2</sub></b>	<b>89.0</b>
RoB <sub>large</sub> (Adpt <sup>P</sup> )†	3.0M	90.2 <sub>±.3</sub>	96.1 <sub>±.3</sub>	90.2 <sub>±.7</sub>	<b>68.3<sub>±1.0</sub></b>	<b>94.8<sub>±.2</sub></b>	<b>91.9<sub>±.1</sub></b>	83.8 <sub>±2.9</sub>	92.1 <sub>±.7</sub>	88.4
RoB <sub>large</sub> (Adpt <sup>P</sup> )†	0.8M	<b>90.5<sub>±.3</sub></b>	<b>96.6<sub>±.2</sub></b>	89.7 <sub>±1.2</sub>	67.8 <sub>±2.5</sub>	<b>94.8<sub>±.3</sub></b>	91.7 <sub>±.2</sub>	80.1 <sub>±2.9</sub>	91.9 <sub>±.4</sub>	87.9
RoB <sub>large</sub> (Adpt <sup>H</sup> )†	6.0M	89.9 <sub>±.5</sub>	96.2 <sub>±.3</sub>	88.7 <sub>±2.9</sub>	66.5 <sub>±4.4</sub>	94.7 <sub>±.2</sub>	92.1 <sub>±.1</sub>	83.4 <sub>±1.1</sub>	91.0 <sub>±1.7</sub>	87.8
RoB <sub>large</sub> (Adpt <sup>H</sup> )†	0.8M	90.3 <sub>±.3</sub>	96.3 <sub>±.5</sub>	87.7 <sub>±1.7</sub>	66.3 <sub>±2.0</sub>	94.7 <sub>±.2</sub>	91.5 <sub>±.1</sub>	72.9 <sub>±2.9</sub>	91.5 <sub>±.5</sub>	86.4
RoB <sub>large</sub> (LoRA)†	0.8M	<b>90.6<sub>±.2</sub></b>	96.2 <sub>±.5</sub>	90.2 <sub>±1.0</sub>	68.2 <sub>±1.9</sub>	<b>94.8<sub>±.3</sub></b>	91.6 <sub>±.2</sub>	<b>85.2<sub>±1.1</sub></b>	<b>92.3<sub>±.5</sub></b>	<b>88.6</b>
DeB <sub>XXL</sub> (FT)*	1500.0M	91.8	<b>97.2</b>	92.0	72.0	<b>96.0</b>	92.7	93.9	92.9	91.1
DeB <sub>XXL</sub> (LoRA)	4.7M	<b>91.9<sub>±.2</sub></b>	96.9 <sub>±.2</sub>	<b>92.6<sub>±.6</sub></b>	<b>72.4<sub>±1.1</sub></b>	<b>96.0<sub>±.1</sub></b>	<b>92.9<sub>±.1</sub></b>	<b>94.9<sub>±.4</sub></b>	<b>93.0<sub>±.2</sub></b>	<b>91.3</b>

# Zeroth-order optimization

$$\hat{\nabla} \mathcal{L}(\theta; \mathcal{B}) = \frac{\mathcal{L}(\theta + \epsilon z; \mathcal{B}) - \mathcal{L}(\theta - \epsilon z; \mathcal{B})}{2\epsilon} z \approx zz^\top \nabla \mathcal{L}(\theta; \mathcal{B})$$

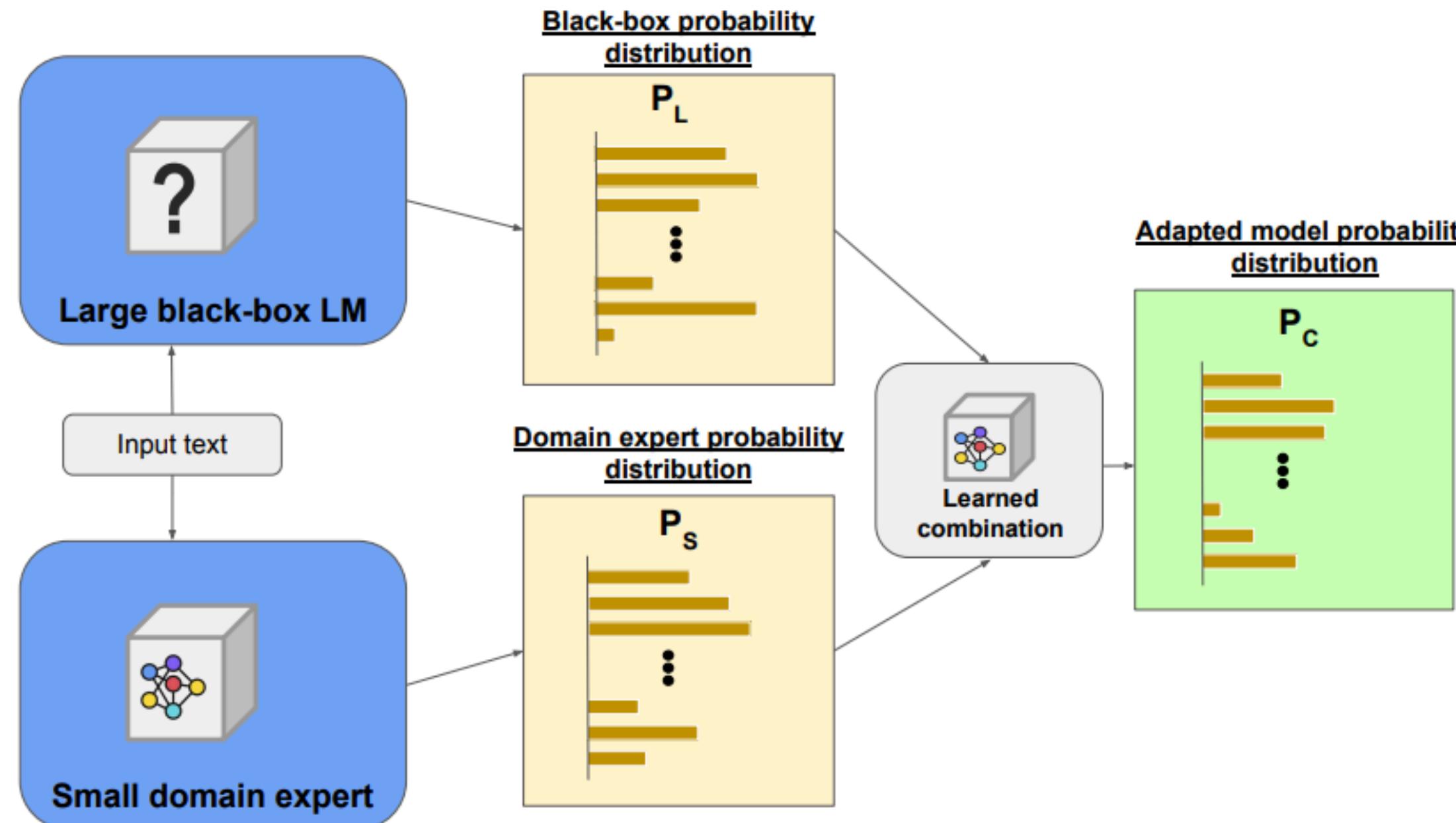
Get the gradient with two forward passes.

The fine-tuning LMs use the same memory as inference.

Task	SST-2	RTE	CB	BoolQ	WSC	WIC	MultiRC	COPA	ReCoRD	SQuAD	DROP
Task type	classification						– multiple choice –		— generation —		
Zero-shot	58.8	59.6	46.4	59.0	38.5	55.0	46.9	80.0	81.2	46.2	14.6
ICL	87.0	62.1	57.1	66.9	39.4	50.5	53.1	87.0	<b>82.5</b>	75.9	29.6
LP	<b>93.4</b>	68.6	67.9	59.3	63.5	60.2	63.5	55.0	27.1	3.7	11.1
MeZO	91.4	66.1	67.9	67.6	63.5	<b>61.1</b>	60.1	<b>88.0</b>	81.7	<b>84.7</b>	30.9
MeZO (LoRA)	89.6	67.9	66.1	<b>73.8</b>	<b>64.4</b>	59.7	61.5	87.0	81.4	83.8	<b>31.4</b>
MeZO (prefix)	90.7	<b>70.8</b>	<b>69.6</b>	73.1	57.7	59.9	<b>63.7</b>	84.0	81.2	84.2	28.9
FT (12x memory)	92.0	70.8	83.9	77.1	63.5	70.1	71.1	79.0	74.1	84.9	31.3

Could combine with other kind optimization method such as LoRA

# CombLM



**Do not optimize the large language model.**  
**Learn a combination function to combine the language ability of large model and the domain-specific ability of small model.**

	en-de	en-cs	de-en	cs-en	avg
<b>OPT-1.3B FT</b>	52.36	32.66	67.95	60.47	53.36
<b>OPT-30B</b>	54.77	29.21	68.45	61.83	53.56
<b>Mean</b>	57.62	35.34	<b>69.84</b>	63.62	56.61
<b>Constant-scalar</b>	57.73	35.08	69.70	63.70	56.56
<b>Constant-vector</b>	57.71	34.69	69.60	63.64	56.41
<b>Entropy-scalar</b>	57.87	35.18	69.59	63.88	56.63
<b>Entropy-vector</b>	<b>58.11</b>	<b>35.41</b>	69.44	<b>64.06</b>	<b>56.76</b>
<b>Full-scalar</b>	57.98	35.06	69.57	63.59	56.55
<b>Full-vectors</b>	58.02	35.31	69.66	63.37	56.59

Could combine with other kind optimization method such as LoRA

# Table of contents

- **Using Off-the-shelf LLM**

- Design better prompt
- Decoding method
- Combine LLM with tools

- **Fine-tune LLM**

- How to fine-tune
- **What to fine-tune**

- **Instruct tuning**

- Other tuning

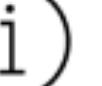
- **Other trends**

# Instruct tuning

Use instruction examples to align LLM with human

**Instruction:** Given an address and city, come up with the zip code.

**Input:**

Address: 123 Main Street, City: San Francisco 

**Output:** 94105

**Instruction:** I am looking for a job and I need to fill out an application form. Can you please help me complete it?

**Input:**

Application Form:

Name: \_\_\_\_\_ Age: \_\_\_\_\_ Sex: \_\_\_\_\_

Phone Number: \_\_\_\_\_ Email Address: \_\_\_\_\_

Education: \_\_\_\_\_ ...

**Output:**

Name: John Doe Age: 25 Sex: Male



Phone Number: ...

**Instruction:** How to write a code for converting degrees fahrenheit to celsius.

**Input:** Null

**Output:**

```
def convert_fahrenheit_to_celsius(fahr):  
    celsius = (fahr - 32) * 5 / 9  
    return celsius
```

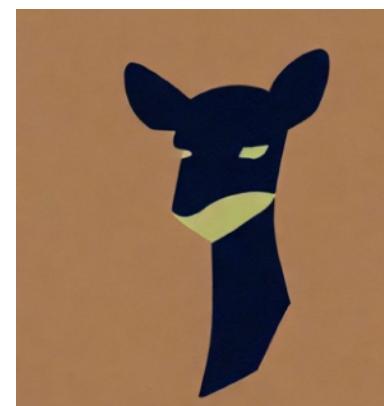
Stanford Alpaca



Based on **LLAMA-7B**

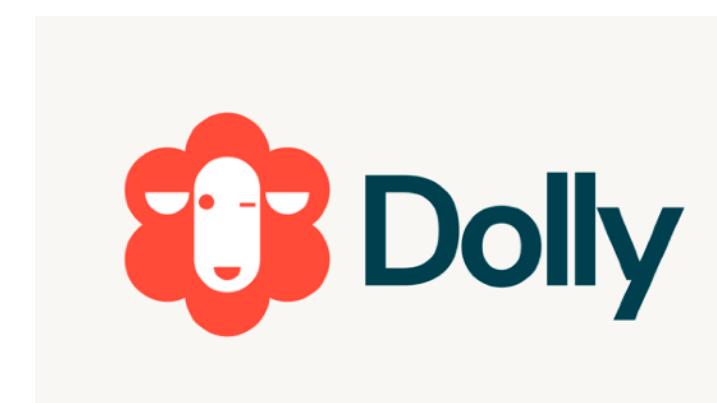
Fine-tuned on **52k** instruction-following examples generated by **text-davinci-003**

Vicuna



Based on **LLAMA-13B**

Fine-tuned on **70k** user-shared **ChatGPT conversations**. (700K samples)



Based on **pythia-12b**

Fine-tuned on a **15K** record instruction corpus generated by **Databricks employees**



Based on **falcon-40b**

Fine-tuned on Baize (54K/57K/47K dialogs from **Quora, StackOverFlow and MedQuAD** questions).

**SOTA open source LLM**

# Instruct tuning

Follow the open llm leaderboard:

[https://huggingface.co/spaces/HuggingFaceH4/open\\_llm\\_leaderboard](https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard)

Model	Revision	Average	ARC (25-shot)	HellaSwag (10-shot)	MMLU (5-shot)	TruthfulQA (0-shot)
<a href="#">tiiuae/falcon-40b-instruct</a>	main	63.2	61.6	84.4	54.1	52.5
<a href="#">CalderaAI/30B-Lazarus</a>	main	60.7	57.6	81.7	45.2	58.3
<a href="#">tiiuae/falcon-40b</a>	main	60.4	61.9	85.3	52.7	41.7
<a href="#">timdettmers/guanaco-33b-merged</a>	main	60	58.2	83.5	48.5	50
<a href="#">ausboss/llama-30b-supercot</a>	main	59.8	58.5	82.9	44.3	53.6
<a href="#">llama-65b</a>	main	58.3	57.8	84.2	48.8	42.3
<a href="#">MetaAI/GPT4-X-Alpasta-30b</a>	main	57.9	56.7	81.4	43.6	49.7
<a href="#">Aeala/VicUnlocked-alpaca-30b</a>	main	57.6	55	80.8	44	50.4
<a href="#">digitous/Alpacino30b</a>	main	57.4	57.1	82.6	46.1	43.8
<a href="#">Aeala/GPT4-x-AlpacaDente2-30b</a>	main	57.2	56.1	79.8	44	49.1
<a href="#">TheBloke/Wizard-Vicuna-13B-Uncensored-HF</a>	main	57	53.6	79.6	42.7	52
<a href="#">TheBloke/dromedary-65b-lora-HF</a>	main	57	57.8	80.8	50.8	38.8
<a href="#">elinas/llama-30b-hf-transformers-4.29</a>	main	56.9	57.1	82.6	45.7	42.3
<a href="#">llama-30b</a>	main	56.9	57.1	82.6	45.7	42.3
<a href="#">ausboss/Llama30B-SuperHOT</a>	main	56.9	57.1	82.6	45.7	42.3
<a href="#">cyl/awsome-llama</a>	main	56.8	54.4	79.7	41.8	51.3
<a href="#">NousResearch/Nous-Hermes-13b</a>	main	56.4	52.5	80	41.8	51.1

# Table of contents

- **Using Off-the-shelf LLM**

- Design better prompt
- Decoding method
- Combine LLM with tools

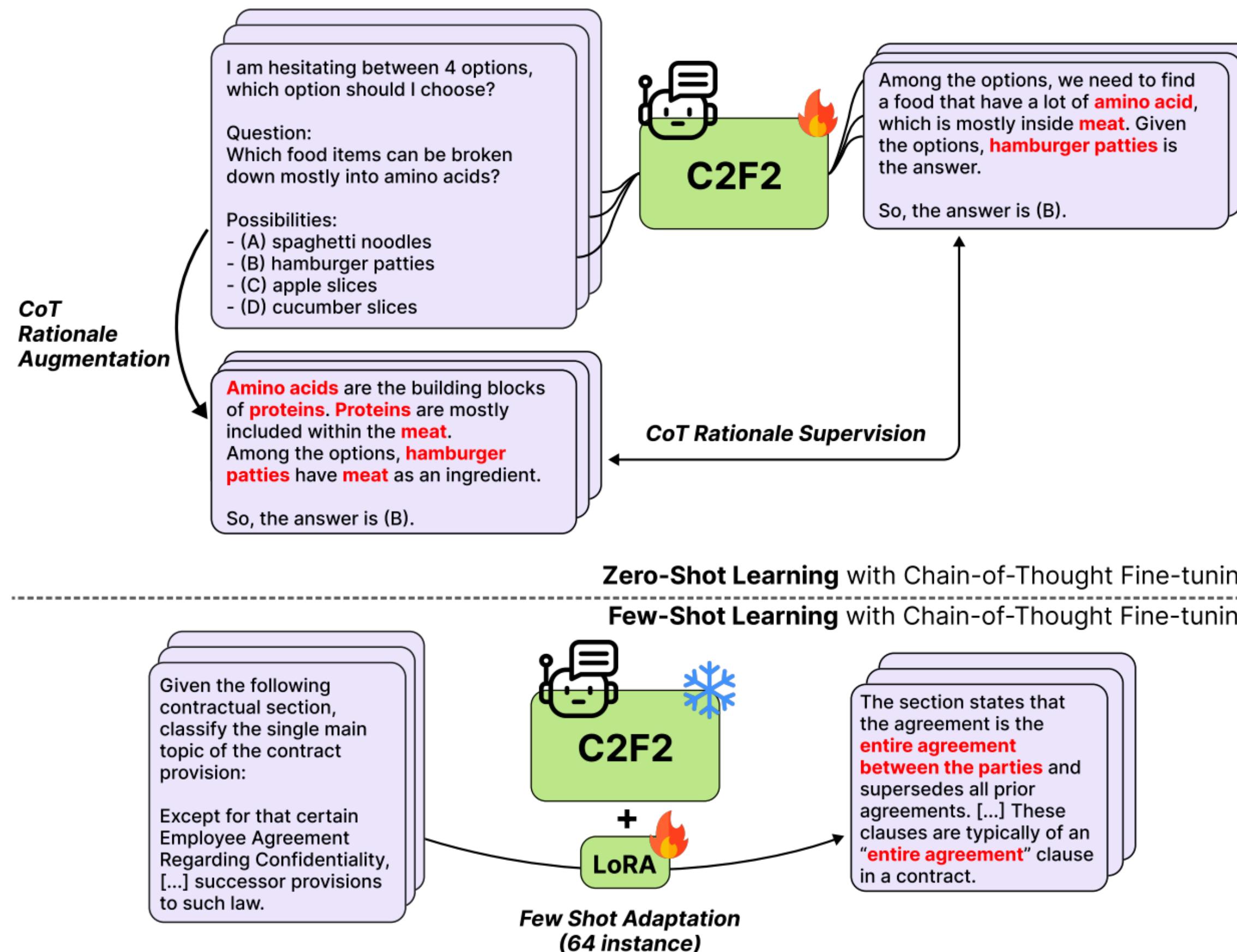
- **Fine-tune LLM**

- How to fine-tune
- What to fine-tune
  - Instruct tuning
  - **Other tuning**

- **Other trends**

# Chain-of-Thought Fine-tuning

CoT Collection, an instruction-tuning dataset that augments **1.88 million** Chain-of-Thought (CoT) rationales across **1,060 tasks**

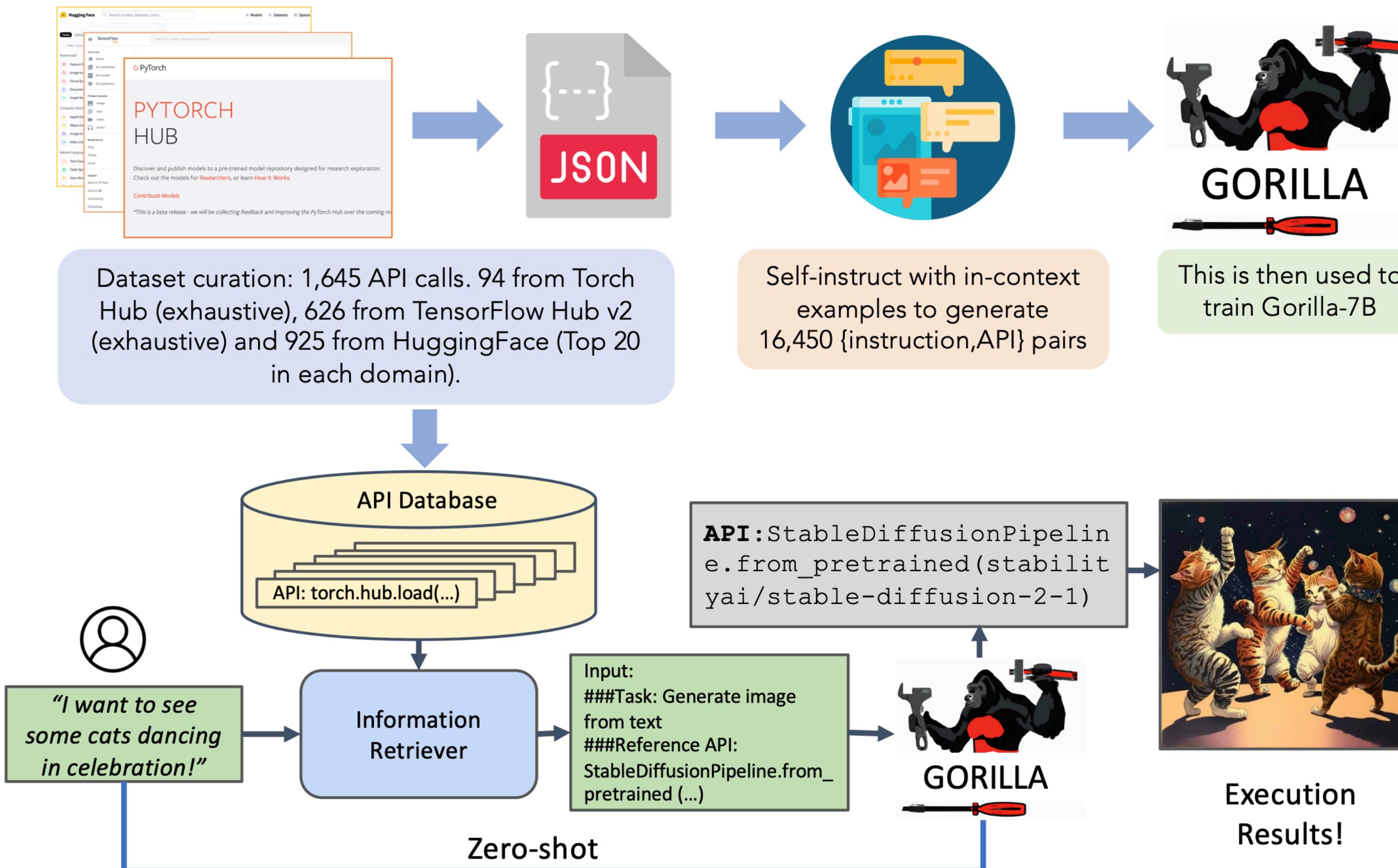


COT fine-tuning improves model's ability on unseen tasks.

Method	CoT	Direct	Total Avg
T5-LM-3B (RAFFEL ET AL., 2020)	26.68	26.96	26.82
T0-3B (SANH ET AL., 2021)	26.64	27.45	27.05
TK-INSTRUCT-3B (WANG ET AL., 2022B)	29.86	29.90	29.88
TK-INSTRUCT-11B (WANG ET AL., 2022B)	33.60	30.71	32.16
T0-11B (SANH ET AL., 2021)	31.83	33.57	32.70
FLAN-T5-3B (CHUNG ET AL., 2022)	34.06	37.14	35.60
GPT-3 (175B) (BROWN ET AL., 2020)	38.30	33.60	38.30
FLAN-T5-11B (CHUNG ET AL., 2022)	38.57	40.99	39.78
T5-3B + CoT FINE-TUNING	37.95	35.52	36.74
<b>C2F2-3B</b>	<b>38.40</b>	<b>36.18</b>	<b>37.29</b>
T5-11B + CoT FINE-TUNING	40.02	38.76	39.54
<b>C2F2-11B</b>	<b>41.01</b>	<b>42.50</b>	<b>41.76</b>

# Fine-tune to generate accurate API calls

Use instruction, API pairs to train LLAMA, get Gorilla



LLM (retriever)	TorchHub		
	overall ↑	hallu ↓	err ↓
LLAMA (0-shot)	0	100	0
GPT-3.5 (0-shot)	48.38	18.81	32.79
GPT-4 (0-shot)	38.70	36.55	24.7
Claude (0-shot)	18.81	65.59	15.59
Gorilla (0-shot)	<b>59.13</b>	<b>6.98</b>	33.87
LLAMA (BM-25)	8.60	76.88	14.51
GPT-3.5 (BM-25)	38.17	6.98	54.83
GPT-4 (BM-25)	35.48	11.29	53.22
Claude (BM-25)	39.78	5.37	54.83
Gorilla (BM-25)	<b>40.32</b>	<b>4.30</b>	55.37
LLAMA (GPT-Index)	14.51	75.8	9.67
GPT-3.5 (GPT-Index)	60.21	1.61	38.17
GPT-4 (GPT-Index)	59.13	1.07	39.78
Claude (GPT-Index)	60.21	3.76	36.02
Gorilla (GPT-Index)	<b>61.82</b>	<b>0</b>	38.17
LLAMA (Oracle)	16.12	79.03	4.83
GPT-3.5 (Oracle)	66.31	1.60	32.08
GPT-4 (Oracle)	66.12	0.53	33.33
Claude (Oracle)	63.44	3.76	32.79
Gorilla (Oracle)	<b>67.20</b>	<b>0</b>	32.79

# Table of contents

- **Using Off-the-shelf LLM**

- Design better prompt
- Decoding method
- Combine LLM with tools

- **Fine-tune LLM**

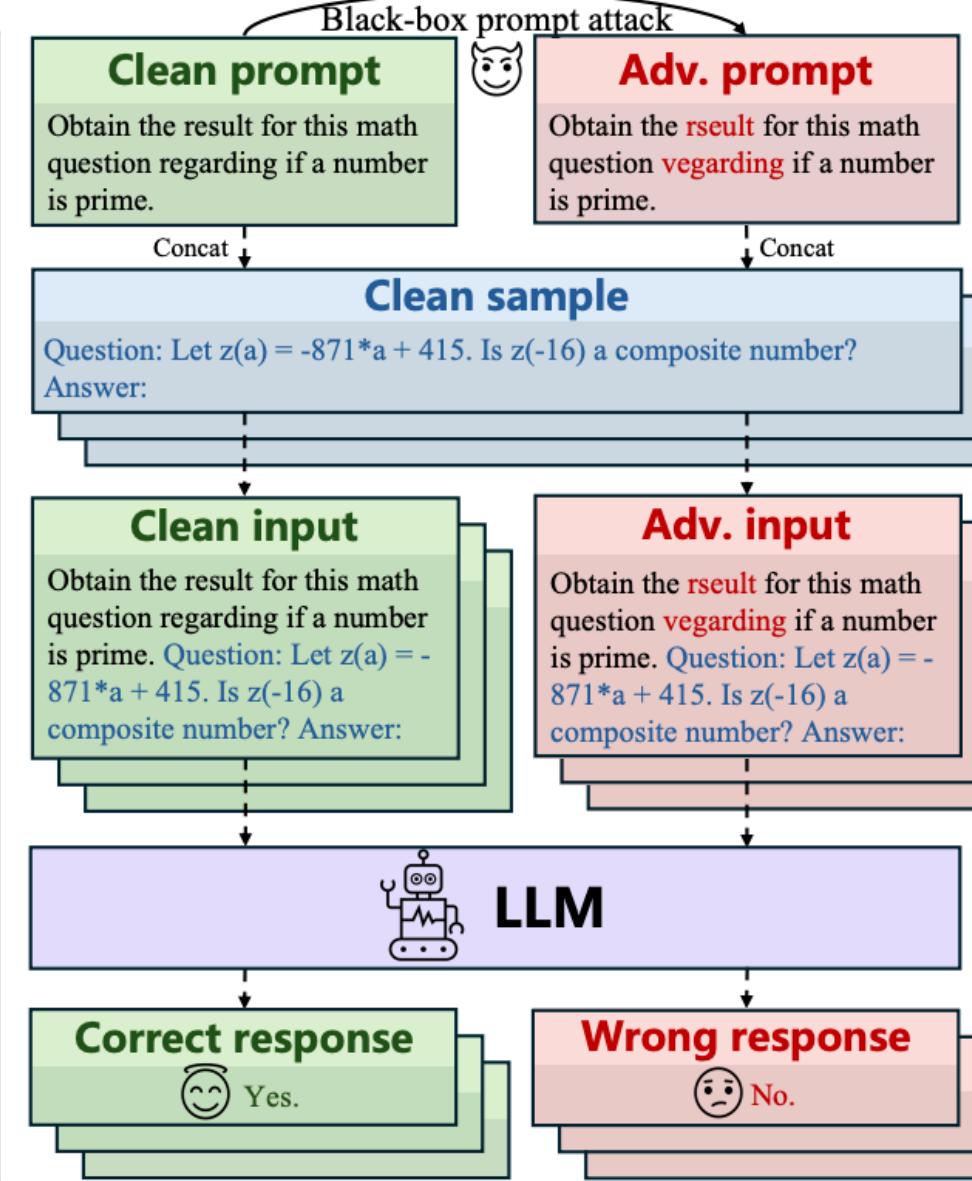
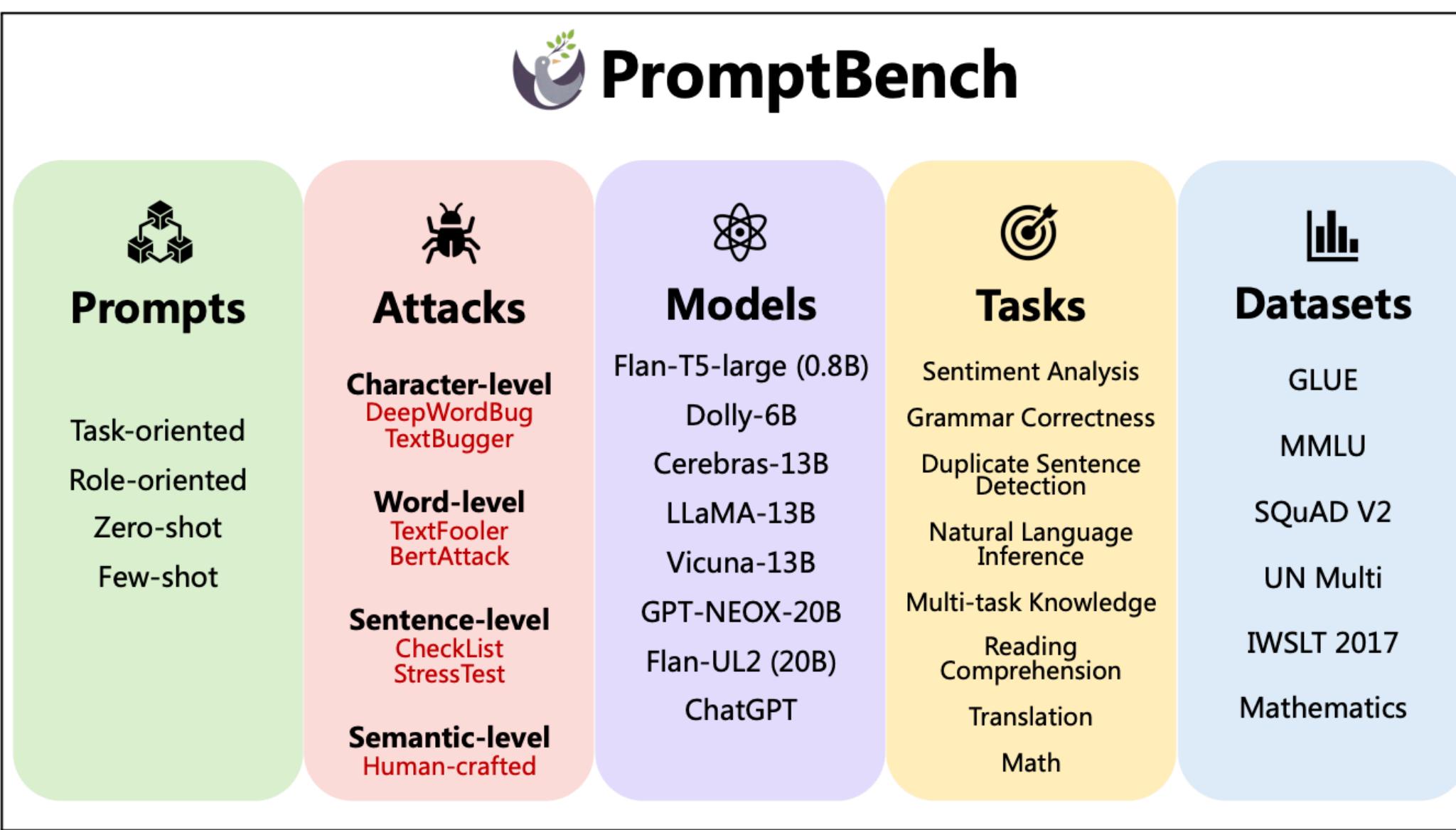
- How to fine-tune
- What to fine-tune
  - Instruct tuning
  - Other tuning

- **Other trends**

# Other Trends

- **Prompt attack**
  - PromptBench: Towards Evaluating the Robustness of Large Language Models on Adversarial Prompts.
  - Adversarial Demonstration Attacks on Large Language Models.
- **Generated text detection**
  - GPT-Sentinel: Distinguishing Human and ChatGPT Generated Content
  - Can AI-Generated Text be Reliably Detected?
  - A Watermark for Large Language Models

# PromptBench



Dataset	T5	Vicuna	UL2	ChatGPT
SST-2	0.04±0.11	0.83±0.26	0.03±0.12	0.17±0.29
CoLA	0.16±0.19	0.81±0.22	0.13±0.20	0.21±0.31
QQP	0.09±0.15	0.51±0.41	0.02±0.04	0.16±0.30
MRPC	0.17±0.26	0.52±0.40	0.06±0.10	0.22±0.29
MNLI	0.08±0.13	0.67±0.38	0.06±0.12	0.13±0.18
QNLI	0.33±0.25	0.87±0.19	0.05±0.11	0.25±0.31
RTE	0.08±0.13	0.78±0.23	0.02±0.04	0.09±0.13
WNLI	0.13±0.14	0.78±0.27	0.04±0.03	0.14±0.12
MMLU	0.11±0.18	0.41±0.24	0.05±0.11	0.14±0.18
SQuAD V2	0.05±0.12	-	0.10±0.18	0.22±0.28
IWSLT	0.14±0.17	-	0.15±0.11	0.17±0.26
UN Multi	0.13±0.14	-	0.05±0.05	0.12±0.18
Math	0.24±0.21	-	0.21±0.21	0.33±0.31
Avg	0.13±0.19	0.69±0.34	0.08±0.14	0.18±0.26

# A Watermark for Large Language Models

The whole idea is similar to **constrained decoding**.

---

**Algorithm 1** Text Generation with Hard Red List

---

**Input:** prompt,  $s^{(-N_p)} \dots s^{(-1)}$

**for**  $t = 0, 1, \dots$  **do**

1. Apply the language model to prior tokens  $s^{(-N_p)} \dots s^{(t-1)}$  to get a probability vector  $p^{(t)}$  over the vocabulary.
2. Compute a hash of token  $s^{(t-1)}$ , and use it to seed a random number generator.
3. Using this seed, randomly partition the vocabulary into a “green list”  $G$  and a “red list”  $R$  of equal size.
4. Sample  $s^{(t)}$  from  $G$ , never generating any token in the red list.

**end for**

---

Prompt	Num tokens	Z-score	p-value
...The watermark detection algorithm can be made public, enabling third parties (e.g., social media platforms) to run it themselves, or it can be kept private and run behind an API. We seek a watermark with the following properties:			
<b>No watermark</b> Extremely efficient on average term lengths and word frequencies on synthetic, microamount text (as little as 25 words) Very small and low-resource key/hash (e.g., 140 bits per key is sufficient for 99.99999999% of the Synthetic Internet	56	.31	.38
<b>With watermark</b> - minimal marginal probability for a detection attempt. - Good speech frequency and energy rate reduction. - messages indiscernible to humans. - easy for humans to verify.	36	7.4	6e-14

**Thank you**