

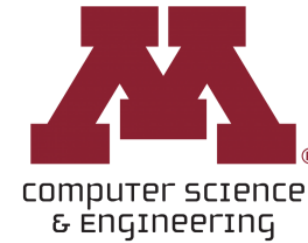
CSCI 5541: Natural Language Processing

Lecture 13: Instructing and augmenting LLMs

Dongyeop Kang (DK), University of Minnesota

dongyeop@umn.edu | twitter.com/dongyeopkang | dykang.github.io

Slides are made by Zae Myung Kim (<https://zaemyung.github.io/>)



Instruction Tuning



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

COMPLETION GPT-3

Explain the theory of gravity to a 6 year old.

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

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.

Language models are not aligned with user intent [\[Ouyang et al., 2022\]](#).



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

COMPLETION

Human

A giant rocket ship blasted off from Earth carrying astronauts to the moon. The astronauts landed their spaceship on the moon and walked around exploring the lunar surface. Then they returned safely back to Earth, bringing home moon rocks to show everyone.

Language models are not aligned with user intent [\[Ouyang et al., 2022\]](#).

→ *We can finetune it with responses we want!*

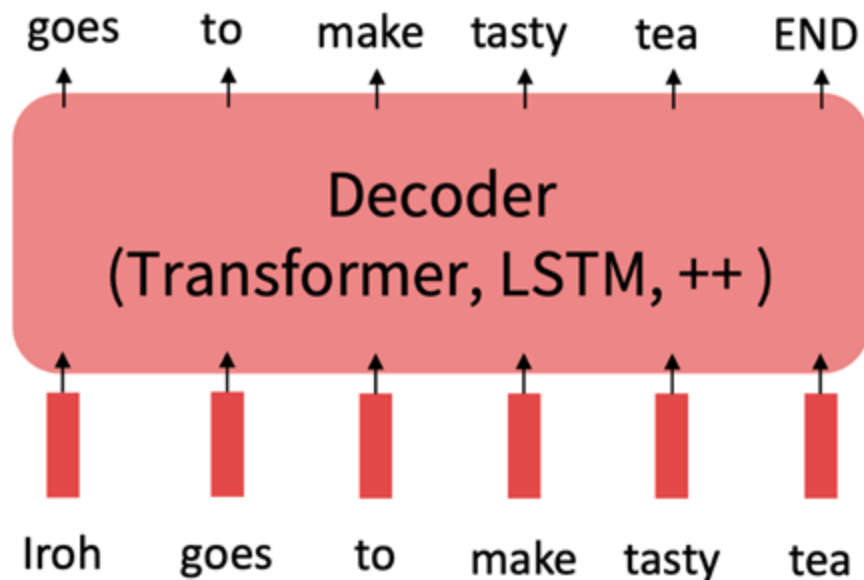


Recap on pretrain-finetune paradigm

- Pretraining can greatly improve performances on downstream NLP tasks by serving as parameter initialization.

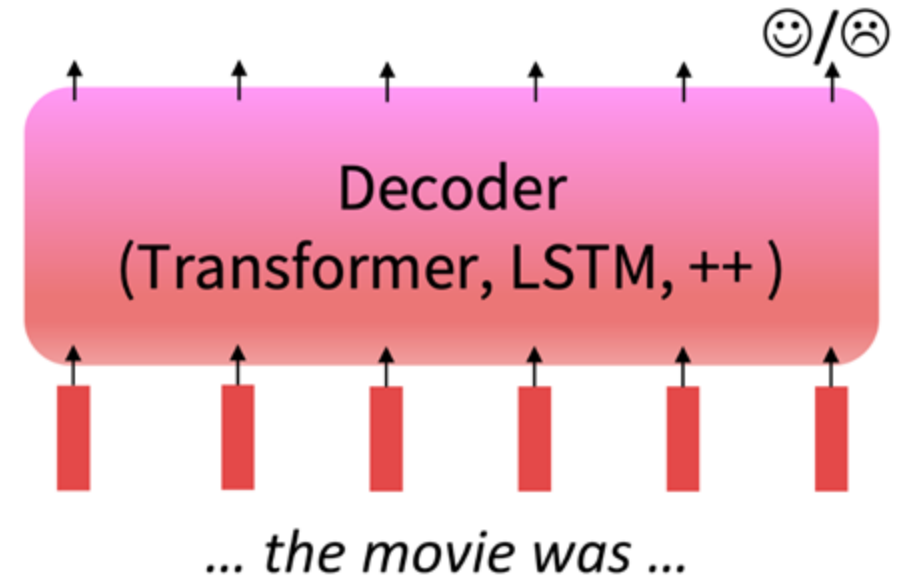
Step 1: Pretrain (via language modeling)

Lots of text; learn general things!



Step 2: Finetune (on your task)

Not many labels; adapt to the task!



Source: cs224n, Stanford

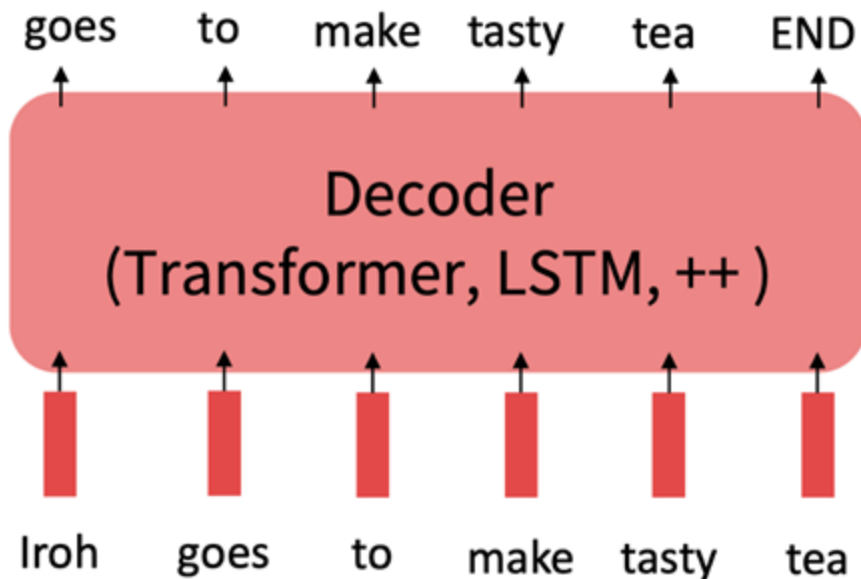


Scaling up finetuning

- Pretraining can greatly improve performances on downstream NLP tasks by serving as parameter initialization.

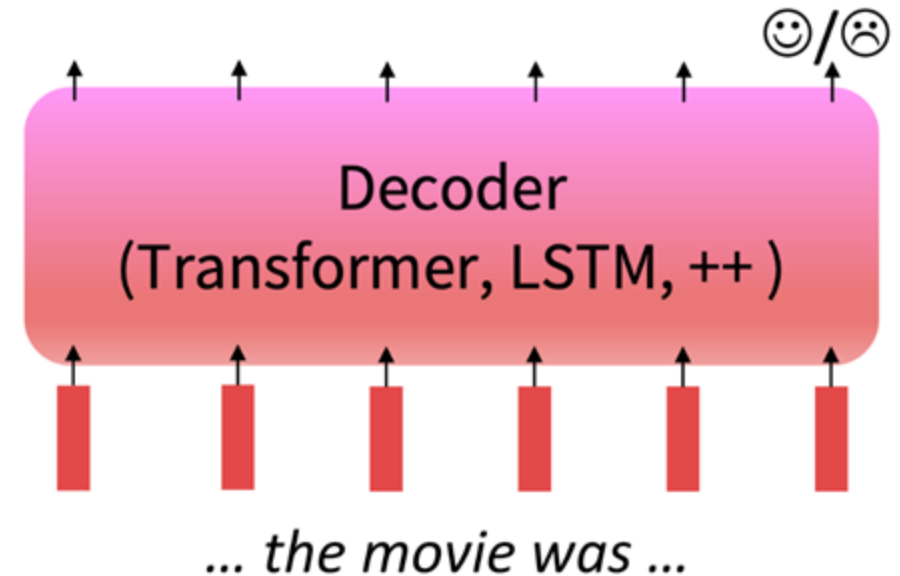
Step 1: Pretrain (via language modeling)

Lots of text; learn general things!



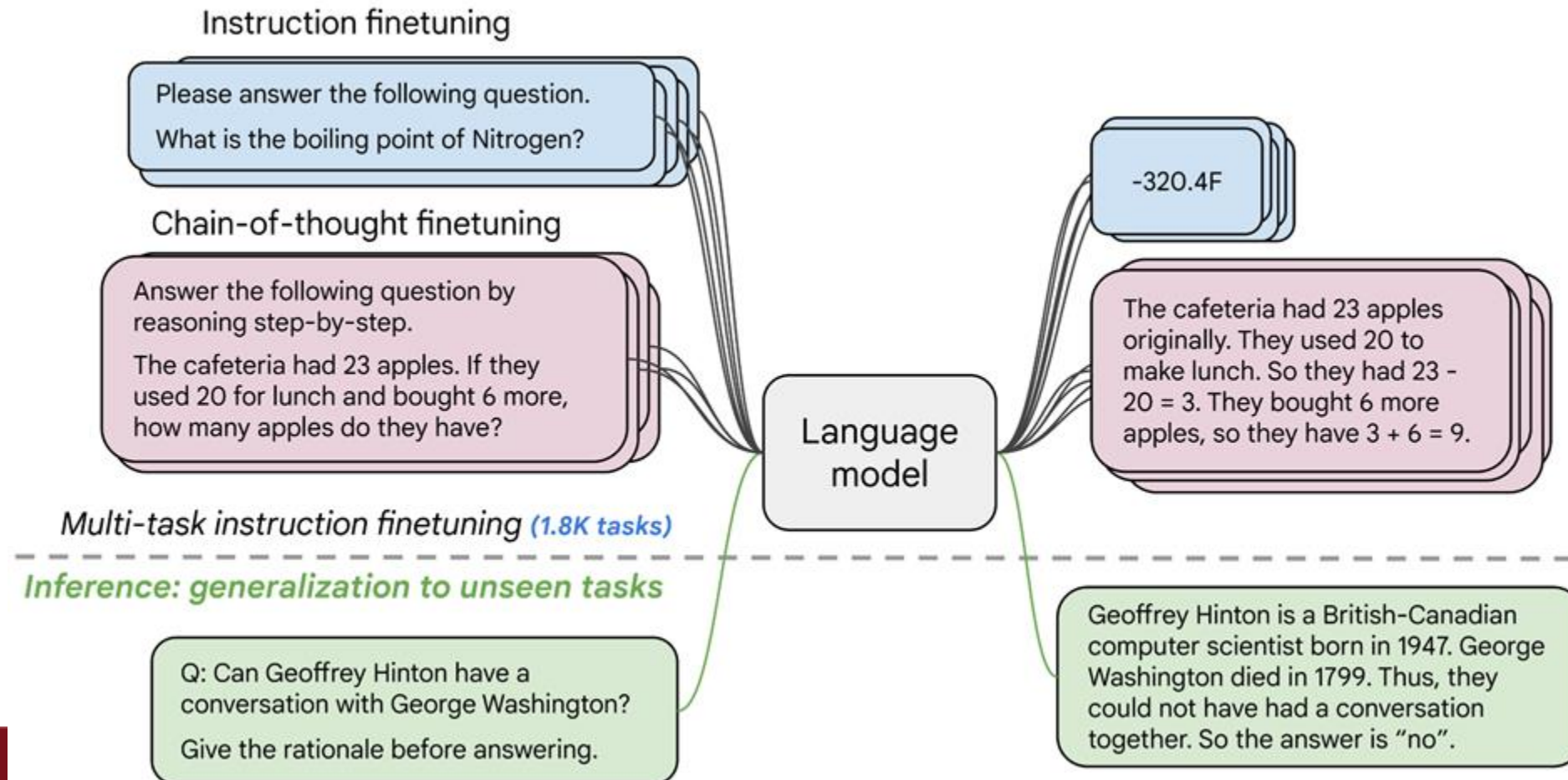
Step 2: Finetune (on **many tasks**)

Many labels; adapt to many tasks!



Instruction finetuning

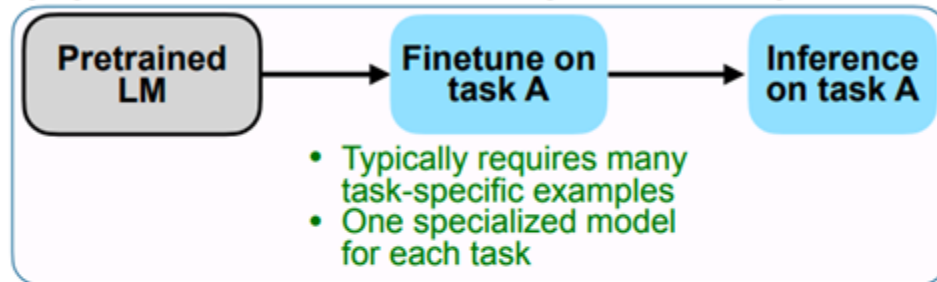
- ❑ Collect examples of ((instruction, input), output) pairs across many tasks and finetune an LM and evaluate on unseen tasks



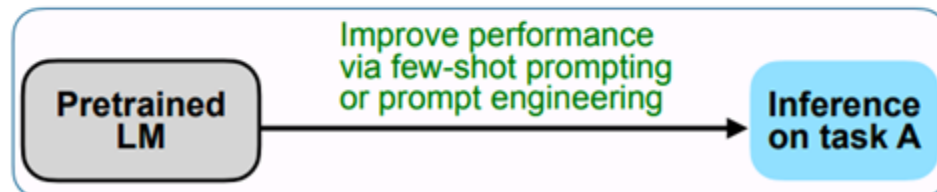
Instruction finetuning vs. standard finetuning

- The main difference lies in the data that the model is trained on
 - Standard supervised finetuning trains models on **input examples** and their **corresponding outputs**.
 - Instruction finetuning augments **input-output examples** with **instructions**, which enables instruction-tuned models to generalize more easily to new tasks.

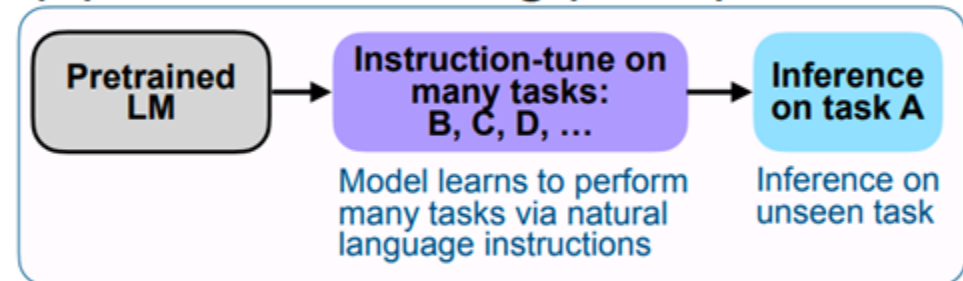
(A) Pretrain–finetune (BERT, T5)



(B) Prompting (GPT-3)

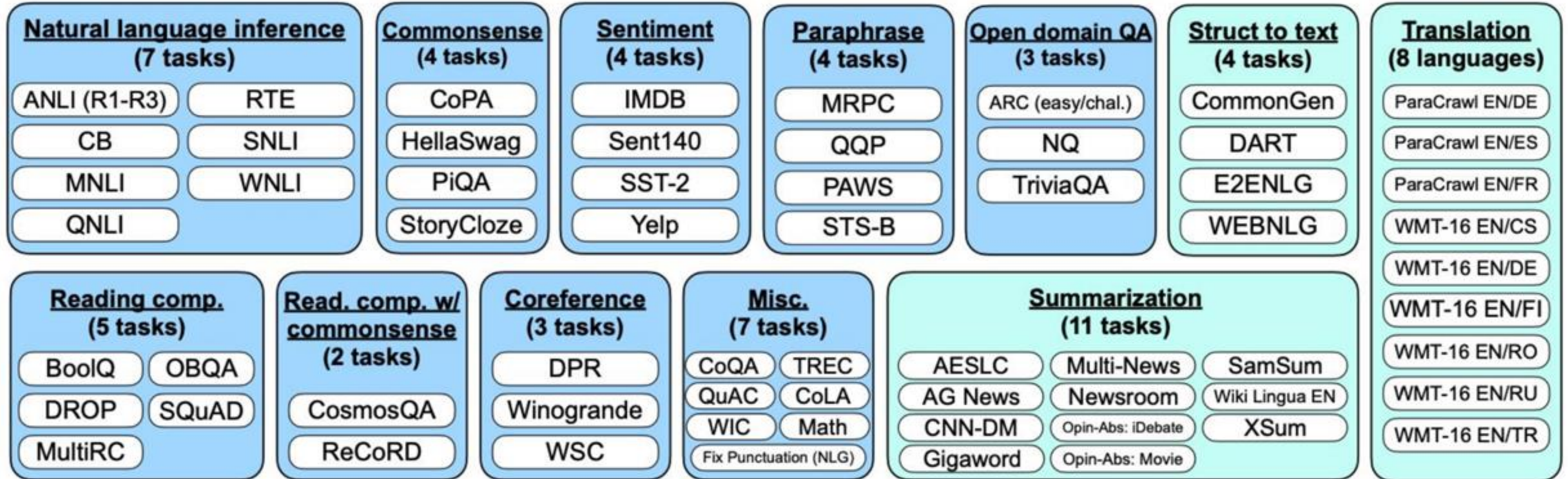


(C) Instruction tuning (FLAN)



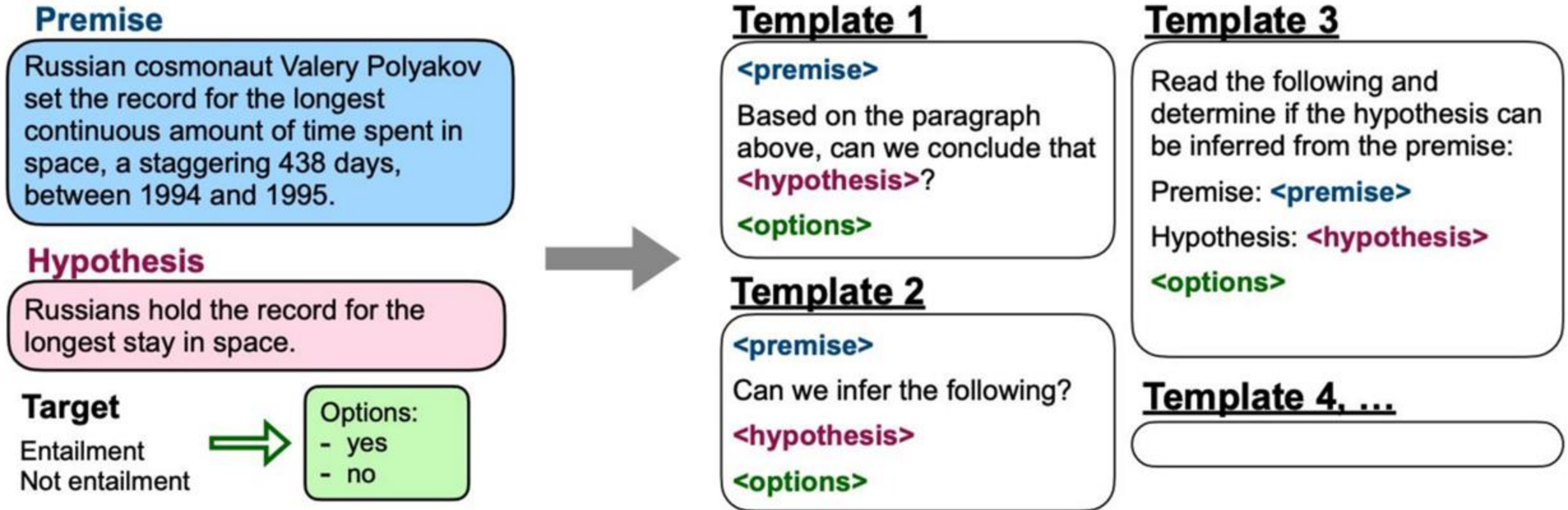
Instruction finetuning

- ❑ 62 NLP datasets
- ❑ 12 “task clusters”



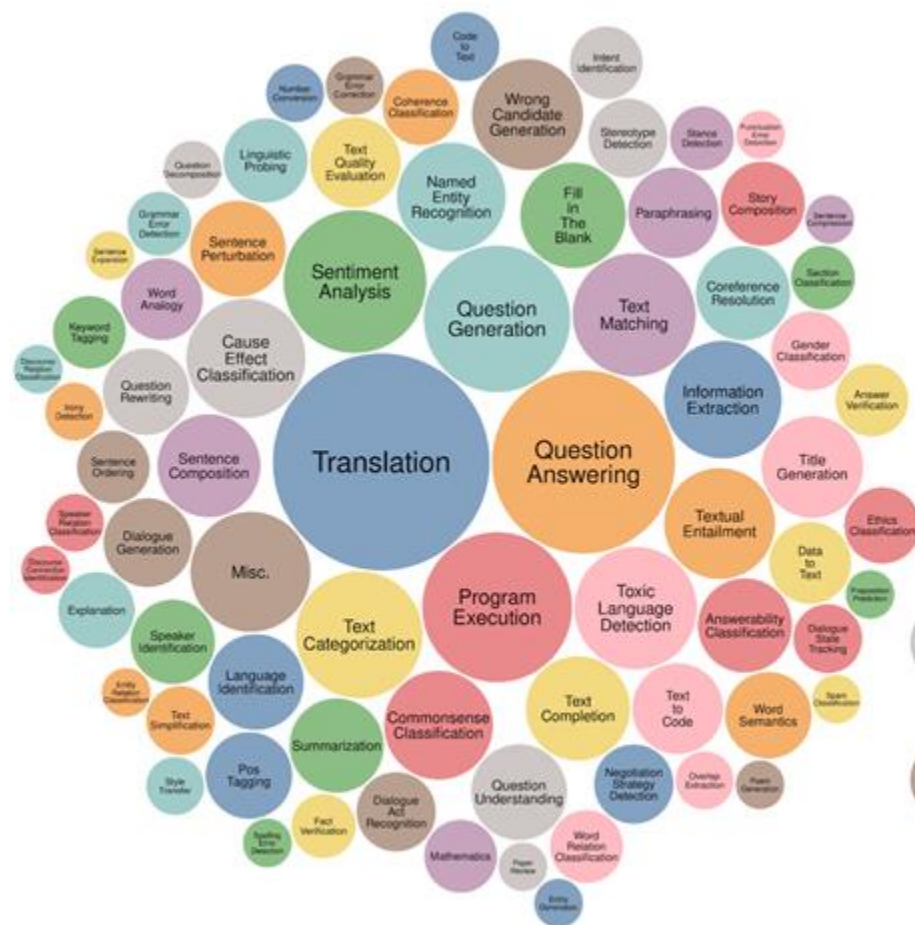
Instruction finetuning templates

- Natural instruction templates for each task



Instruction pretraining?

- ❑ Scaling up data and model improves performance.
- ❑ **SuperNaturalInstructions** dataset contains **over 1.6K tasks, 3M+ examples**



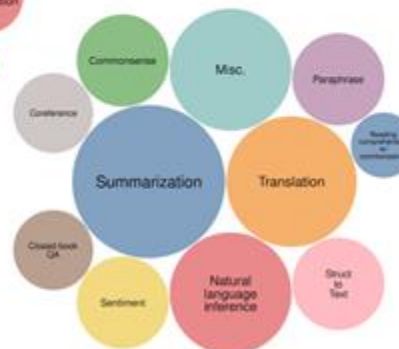
(a) SUP-NATINST (this work)



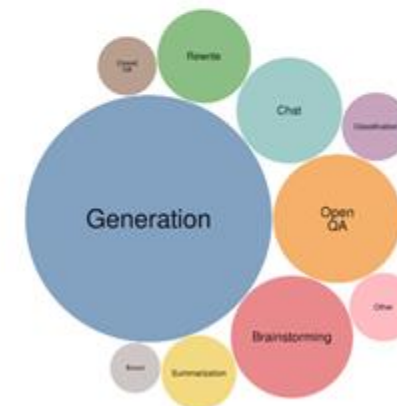
(b) NATINST



(c) PROMPTSOURCE (T0 subset)



(d) FLAN



(e) INSTRUCTGPT

Or less is more?

- *"LIMA – Less Is More for Alignment"* ([Zhou et al. 2023](#))
- Authors report that LLaMa 65B model finetuned on a collection of high quality and diverse 1,000 samples are enough to beat models trained on much larger instruction datasets.

Source	#Examples	Avg Input Len.	Avg Output Len.
Training			
Stack Exchange (STEM)	200	117	523
Stack Exchange (Other)	200	119	530
wikiHow	200	12	1,811
Pushshift r/WritingPrompts	150	34	274
Natural Instructions	50	236	92
Paper Authors (Group A)	200	40	334
Dev			
Paper Authors (Group A)	50	36	N/A
Test			
Pushshift r/AskReddit	70	30	N/A
Paper Authors (Group B)	230	31	N/A

Table 1: Sources of training prompts (inputs) and responses (outputs), and test prompts. The total amount of training data is roughly 750,000 tokens, split over exactly 1,000 sequences.

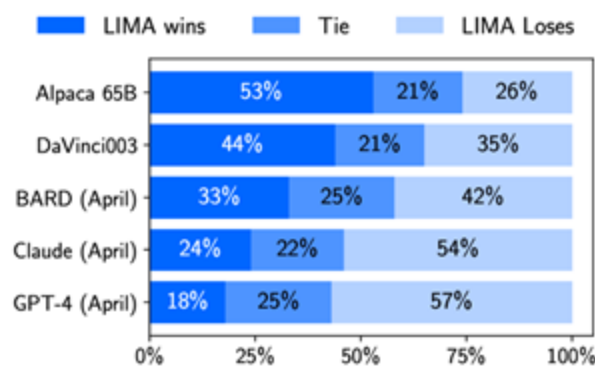


Figure 1: Human preference evaluation, comparing LIMA to 5 different baselines across 300 test prompts.

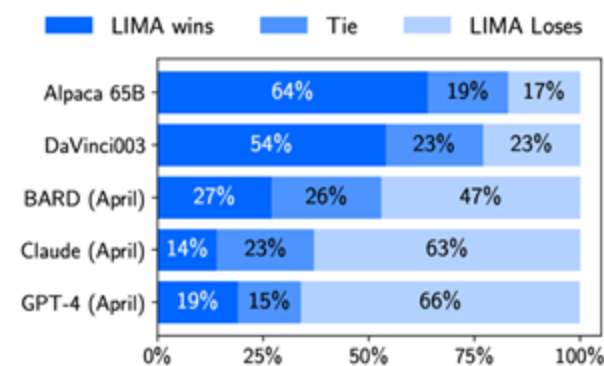
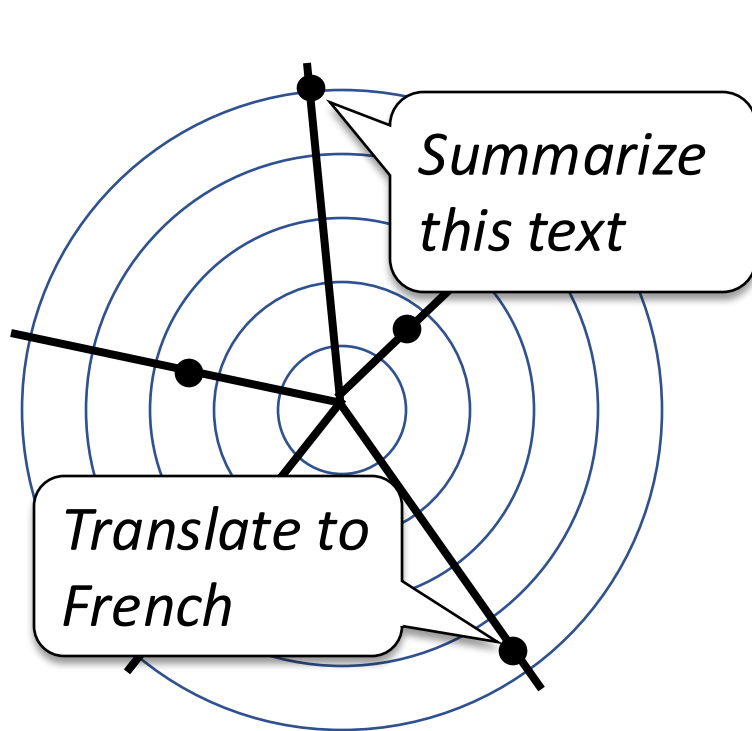


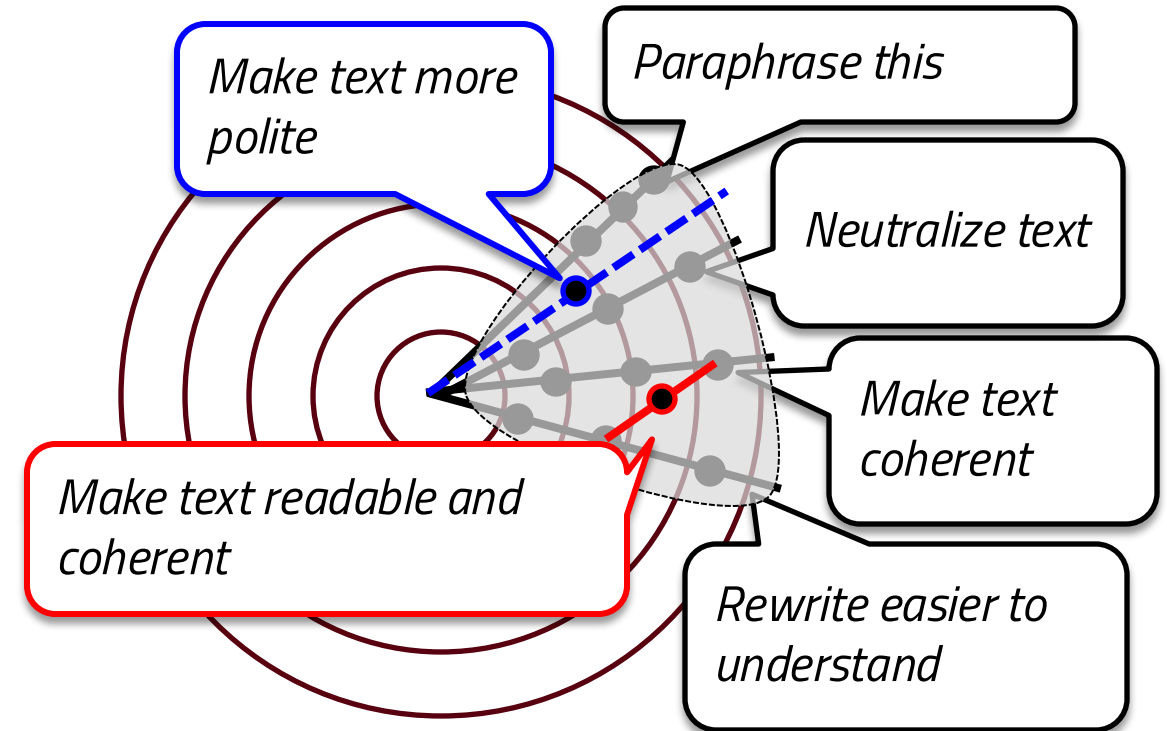
Figure 2: Preference evaluation using GPT-4 as the annotator, given the same instructions provided to humans.



General-purpose (left) vs Task-specific (right) Instruction Tuning



Sparse task distribution



Dense task distribution

Generalize to **composite** & **unseen** tasks

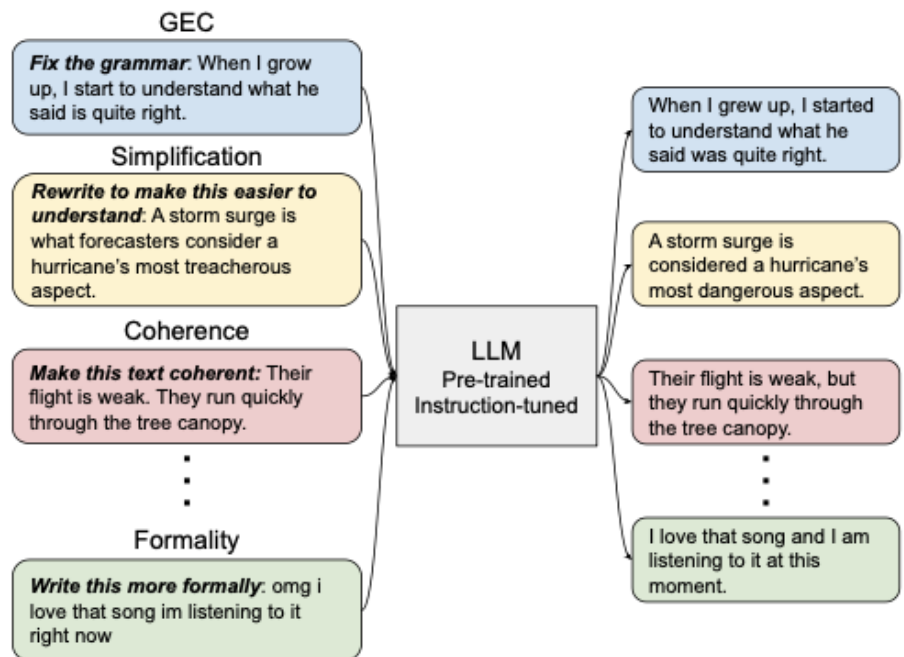
Densifying the task distribution to strictly instructions within the text revision domain can largely improve model performance for revision tasks over scaling model size with general instructions

[Raheja et al., EMNLP Findings 23]



User-driven Revision (CoEdIT)

Conversational Text Editing via Instruction Tuning



Edit Intention / Task	Verbalizers
GEC	Fix grammar, Fix grammar in this sentence, Fix grammar in the sentence, Fix grammar errors, Fix grammatical errors, Fix grammaticality, Fix all grammatical errors, Fix grammatical errors in this sentence, Fix grammar errors in this sentence, Fix grammatical mistakes in this sentence, Fix grammaticality in this sentence, Fix grammaticality of the sentence, Fix disfluencies in the sentence, Make the sentence grammatical, Make the sentence fluent, Fix errors in this text, Update to remove grammar errors, Remove all grammatical errors from this text, Improve the grammar of this text, Improve the grammaticality, Improve the grammaticality of this text, Improve the grammaticality of this sentence, Grammar improvements, Remove grammar mistakes, Remove grammatical mistakes, Fix the grammar mistakes, Fix grammatical mistakes

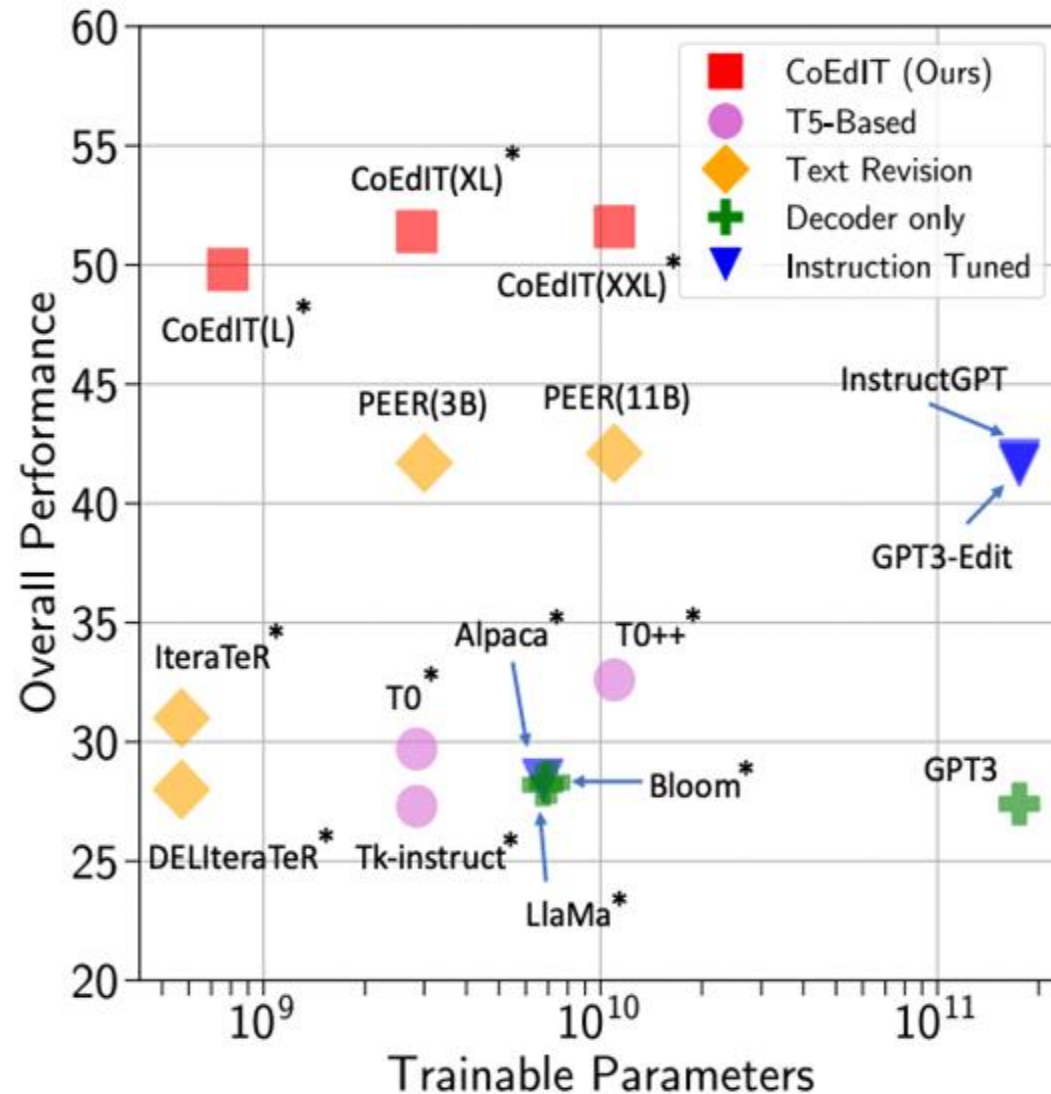
Edit Intention	Datasets	Size	Example Input	Example Output
FLUENCY	NUCLE-14 Lang-8 BEA-19	20k	<i>Fix the grammar:</i> When I grow up, I start to understand what he said is quite right.	When I grew up, I started to understand what he said was quite right.
COHERENCE	DiscoFuse	11k	<i>Make this text coherent:</i> Their flight is weak . They run quickly through the tree canopy.	Their flight is weak , but they run quickly through the tree canopy.
CLARITY (Simplification)	NEWSELA WikiLarge WikiAuto ParabankV2 ITERATER-CLARITY	13k	<i>Rewrite to make this easier to understand:</i> A storm surge is what forecasters consider a hurricane's most treacherous aspect.	A storm surge is considered a hurricane's most dangerous aspect.
STYLE (Paraphrase)	ParabankV2	15k	<i>Paraphrase this:</i> Do you know where I was born?	Do you know my birthplace?
STYLE (Formalize)	GYAFC	12k	<i>Write this more formally:</i> omg i love that song im listening to it right now	I love that song and I am listening to it at this moment .
STYLE (Neutralize)	WNC	11k	<i>Write in a more neutral way:</i> The authors' exposé on nutrition studies.	The authors' statements on nutrition studies.

Table 1: Example data instances in the CoEdIT dataset (82K <instruction: source, target> pairs) Instructional prompts in the inputs are *italicized*.

[Raheja et al., EMNLP Findings 23]



Quantitative Evaluations of Text Revision Models



CoEdIT's task specific instruction tuning largely outperforms most generally instruction-tuned models

Overall, CoEdIT generates better text edits than models that are even 60x larger

[Raheja et al., EMNLP Findings 23]



CoEdit on HuggingFace

Model	Number of parameters
CoEdit-large	770M
CoEdit-xl	3B
CoEdit-xxl	11B

<https://huggingface.co/grammarly/coedit-large>

Downloads last month
29,703



```
from transformers import AutoTokenizer,
T5ForConditionalGeneration

tokenizer =
AutoTokenizer.from_pretrained("grammarly/coedit-large")
model =
T5ForConditionalGeneration.from_pretrained("grammarly/coe
dit-large")
input_text = 'Fix grammatical errors in this sentence:
When I grow up, I start to understand what he said is
quite right.'
input_ids = tokenizer(input_text,
return_tensors="pt").input_ids
outputs = model.generate(input_ids, max_length=256)
edited_text = tokenizer.decode(outputs[0],
skip_special_tokens=True)
```

[Raheja et al., EMNLP Findings 23]



Augmented Language Models



Weakness of LLMs

- ❑ Pre-trained LLMs struggle at completing tasks that require:
 - **Latest knowledge** after the model pretraining time cutoff or
 - Knowledge with **internal/private knowledge base**
 - **Symbolic or other deterministic execution** capabilities
- ❑ These issues stem from their fundamental limitations:
 - They are trained to perform statistical modeling given a **single parametric model** and **a limited context**
 - Their main objective function, *the next token prediction task*, does not cater for **explicit symbolic capabilities**

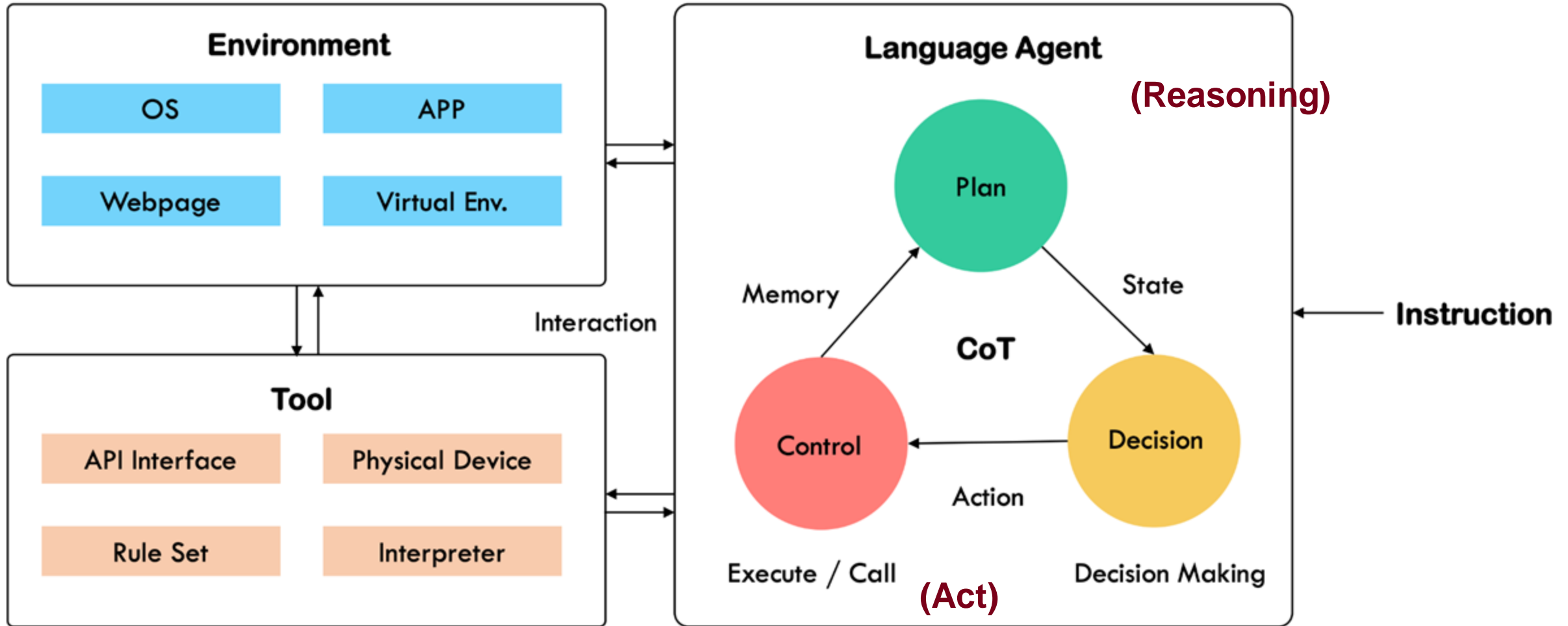


Augmented Language Models

- Recent trend is to move slightly away from the purely statistical language modeling and *integrate external components*
 - So that a *more relevant context* is produced at the cost of more computation
 - Resulting in non-parametric models
- An augmentation can be viewed in three dimensions: [\[Mialon et al. 2023\]](#)
 - **Reasoning**: breaking up a complex task into smaller subtasks
 - **Tool**: external modules that can be called
 - **Act**: Calling of a tool to have an effect



Augmented Language Models



Reasoning



Reasoning

- ❑ Reasoning is the ability to make inferences using evidence and logic.
 - Commonsense, mathematical, symbolic, etc.
 - Often this involves deductions from inference chains, i.e. “multi-step reasoning”
- ❑ Main challenge is to **break down a complex problem into smaller subproblems** and generate the solution by composing the (correctly predicted) answers to the subproblems.
- ❑ Eliciting reasoning in LLMs
 - Eliciting reasoning with prompting
 - Divide and concur with recursive prompting
 - Teaching LLMs to reason



Eliciting reason with prompting

- ❑ Essentially methods can be categorized as either zero-shot or few-shot
- ❑ Zero-shot prompting
 - “Let’s think step by step.” [\[Kojima et al. 2022\]](#)
 - “Chain-of-thought decoding” [\[Wang and Zhou 2024\]](#)

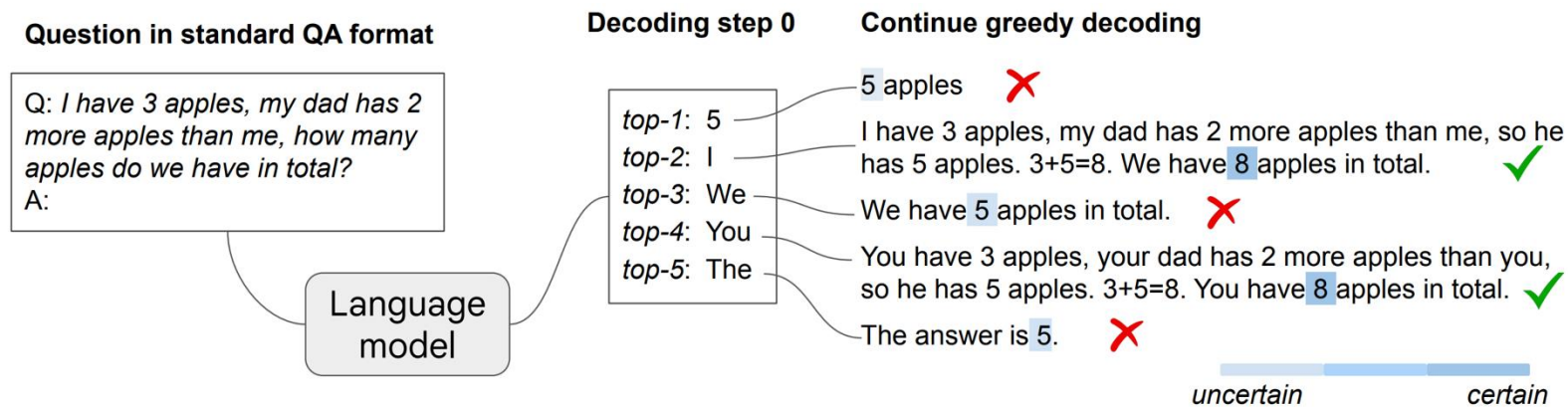


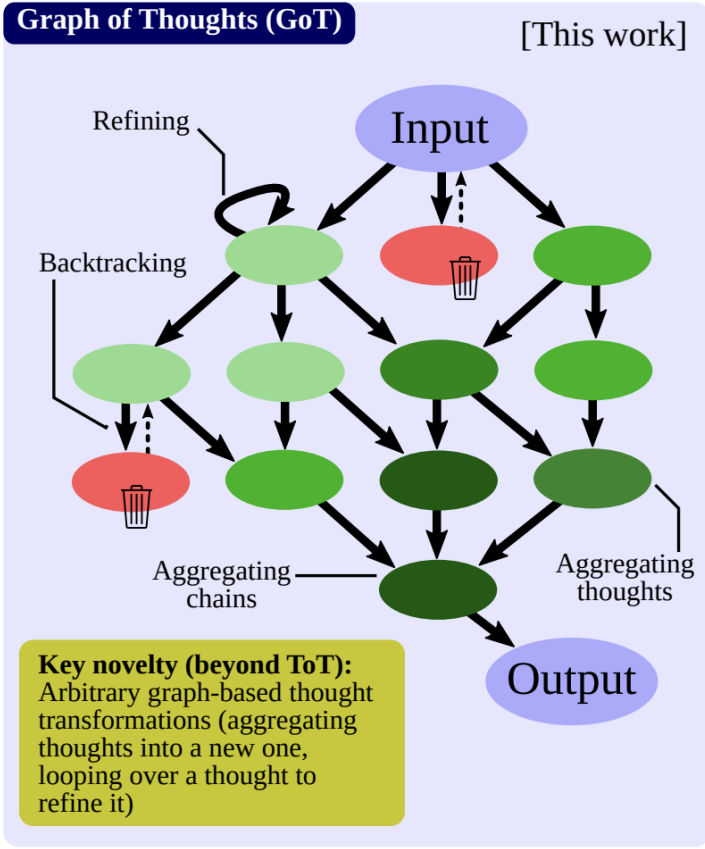
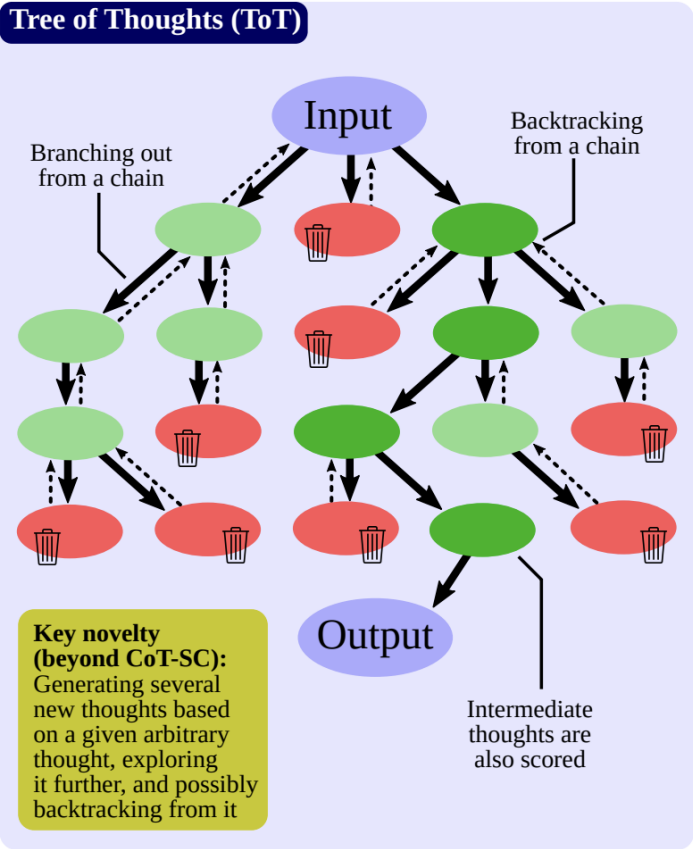
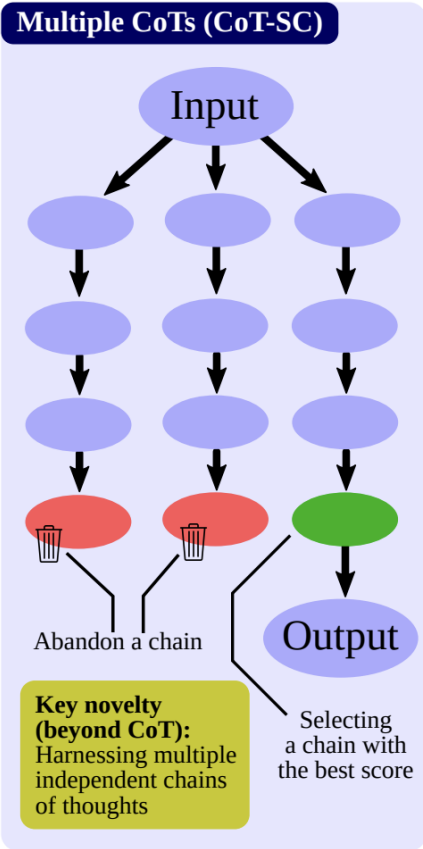
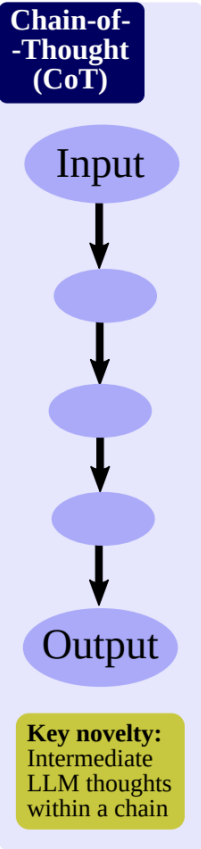
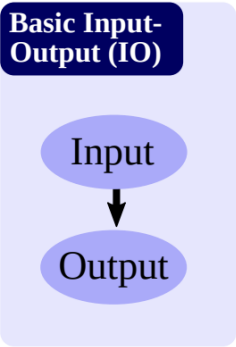
Figure 1 | **Illustration of CoT-decoding.** Pre-trained LLMs are capable of inherent reasoning without prompting by considering alternative top- k tokens, rather than solely relying on the top-1 greedy decoding path. Moreover, these models tend to display higher confidence in decoding the final answer (indicated by a darker shaded color) when a CoT reasoning path is present.



Eliciting reason with prompting

□ Few-shot prompting – variants of CoT prompting

[Besta et al. 2024]

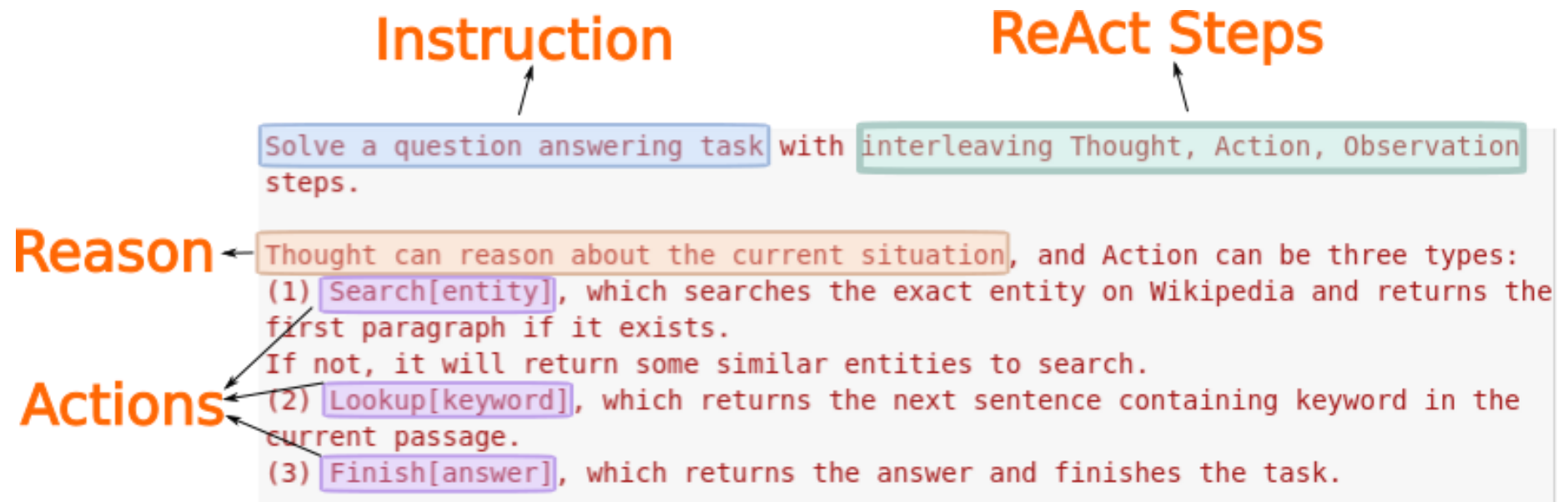


Legend

- Thoughts:
 - Unscored (blue oval)
 - Positive score (green oval)
 - Negative score (red oval)
- ↓ Dependencies between thoughts
- 🗑️ Abandon thought
- ↶ Backtrack

Eliciting reason with prompting

- Few-shot prompting – via “programming”
 - ReAct prompting [\[Yao et al. 2022\]](#)



<https://www.width.ai/post/react-prompting>



Eliciting reason with prompting

- Few-shot prompting – via “programming”
 - ReAct prompting [\[Yao et al. 2022\]](#)

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.

Act 1: `Search[Apple Remote]`

Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple ... originally designed to control the `Front Row media center` program ...

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search `Front Row` next and find what other device can control it.

Act 2: `Search[Front Row]`

Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports', '`Front Row (software)`', ...]

Thought 3: Front Row is not found. I need to search `Front Row (software)` .

Act 3: `Search[Front Row (software)]`

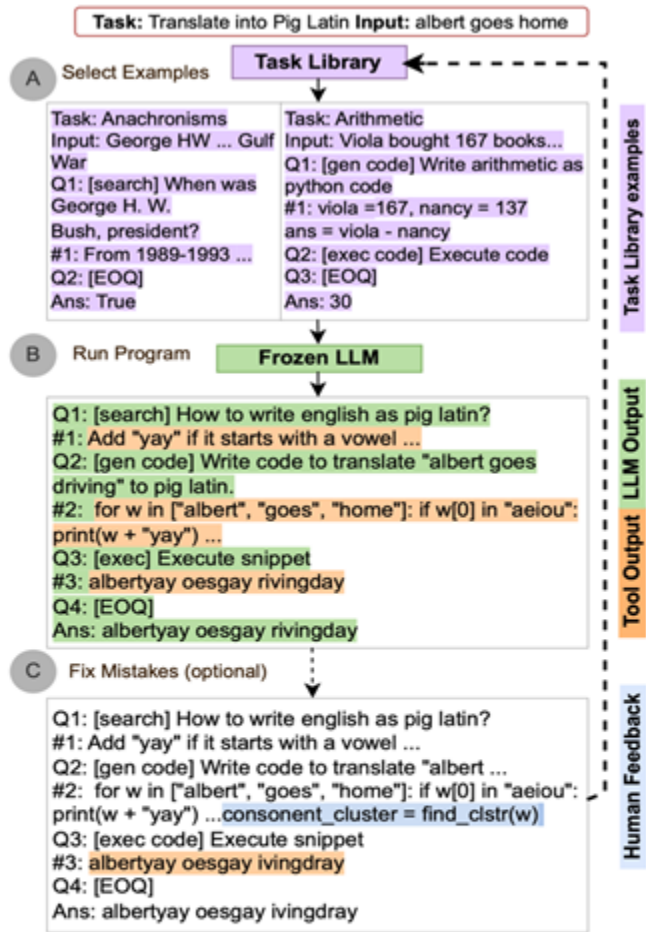
Obs 3: Front Row is a discontinued media center software ...

Thought 4: `Front Row (software)` is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.

Act 4: `Finish[keyboard function keys]`



Programming LLMs for *multi-step reasoning*



- ❑ **ART** (Automatic multi-step reasoning and tool-use for large language models; [Paranjape et al. 2023](#)):
 - ART automatically creates decompositions (multistep reasoning) for examples of new tasks.
 - ART retrieves comparable task instances from a library, enabling quick task analysis and tool application.
 - Using a structured query language, it facilitates reading intermediate stages, pausing for external tool use, and restarting after tool output integration.
 - ✓ At each step, the framework selects and utilizes the most appropriate tools.

Figure 1: ART generates automatic multi-step decompositions for new tasks by selecting decompositions of related tasks in the *task library* (A) and selecting and using tools in the *tool library* alongside LLM generation (B). Humans can optionally edit decompositions (eg. correcting and editing code) to improve performance (C).



Programming LLMs for *multi-step reasoning*

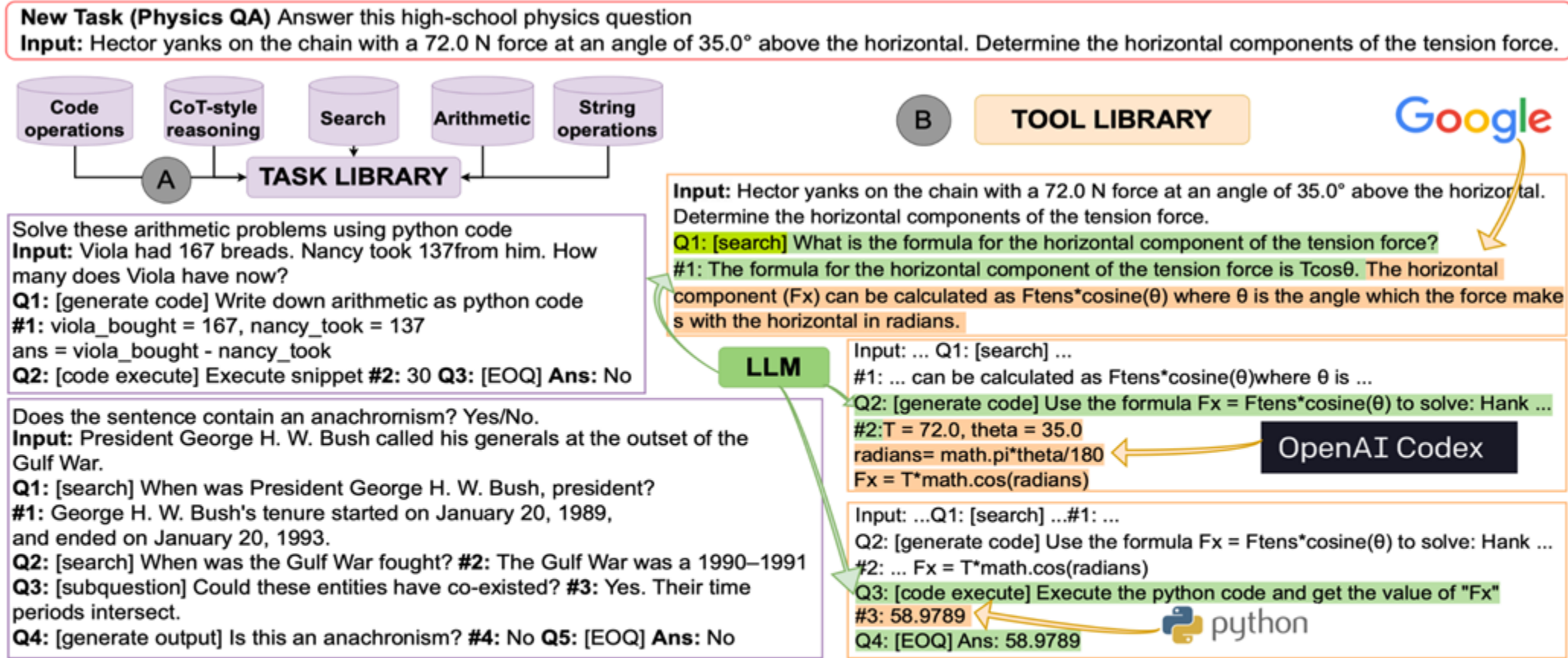


Figure 2: A run-through of ART on a new task, Physics QA. (A) Programs of related tasks like anachronisms and Math QA provide few-shot supervision to the LLM — related sub-steps and tools in these programs can be used by the LLM for cross-task generalization (shown in purple). (B) Tool use: Search is used to find the appropriate physics formula, and code generation and execution are used to substitute given values and compute the answer (shown in orange).



Programming LLMs for *multi-step reasoning*

Task Name (Cluster)	Few Shot	AutoCot	ART w/o Tool Use	ART	GPT-3 Best
Test Tasks					
Sentence Ambiguity (Search)	70.67 ⁵	51.47	71.00	73.33	-
Strategy QA (Search)	55.49 ⁵	27.22	59.37	66.44	-
Physics (Search)	70.09 ⁵	61.83	59.13	67.55	-
Δ with ART (Search)	+3.7	+22.27	+ 5.9		
Physics Questions (Arithmetic)	7.02 ⁵	5.56	6.30	20.37	-
Operators (Arithmetic)	71.23 ⁷	75.52	71.80	92.00	-
Unit interpretation (Arithmetic)	58.2 ⁷	41.20	51.4	53.99	-
Repeat copy logic (Arithmetic)	50.01 ⁷	15.63	31.25	44.38	-
Object Counting (Arithmetic)	39.2 ⁷	26.80	42.2	87.00	81.20 ¹
Penguins in a table (Arithmetic)	58.23 ⁷	40.40	68.86	77.85	72.34 ¹
Reasoning about objects (Arithmetic)	71.00 ⁷	33.33	45.35	64.34	52.69 ¹
Tracking shuffled objects (Arithmetic)	22.39 ⁷	19.44	18.14	37.67	36.32 ¹
Δ with ART (Arithmetic)	+19.0	+36.7	+ 23.1		+6.1
Word Unscramble (String)	40.72 ⁷	32.44	23.03	42.7	-
Simple Text Editing (Code)	35.31 ⁵	30.21	20.74	27.65	-
CS Algorithms (Code)	73.48 ⁷	0.0	41.59	88.11	-
Sports Understanding (CoT)	69.74 ⁵	51.47	92.89	-	86.59 ¹
Snarks (CoT)	54.58 ⁵	57.24	57.13	-	65.2 ¹
Disambiguation QA (Free-form)	55.03 ⁵	48.45	55.89	-	60.62 ¹
Temporal sequences (CoT)	55.80 ⁷	19.70	49.5	-	81.8 ¹
Ruin names (CoT)	71.01 ⁵	55.28	60.22	-	-
Δ with ART (Misc)	2.4	22.5	24.37		-9.4
Δ with ART (Overall)	+6.9	+24.6	+16.7		-1.7
MMLU					
College Computer Science (Search)	41.00	43.99	63.40	67.80	63.6 ⁶
Astronomy (Search)	62.10	41.48	76.71	79.1	62.5 ⁶
Business Ethics (Search)	61.60	48.8	77.17	81.16	72.7 ⁶
Virology (Search)	50.03	49.52	71.60	71.49	50.72 ⁶
Geography (Search)	77.67	57.07	70.30	71.71	81.8 ⁶
Mathematics (Arithmetic)	36.67	33.77	39.50	45.66	34.5 ⁶
Δ with ART (MMLU)	+14.6	+23.7	+3.0		+8.5

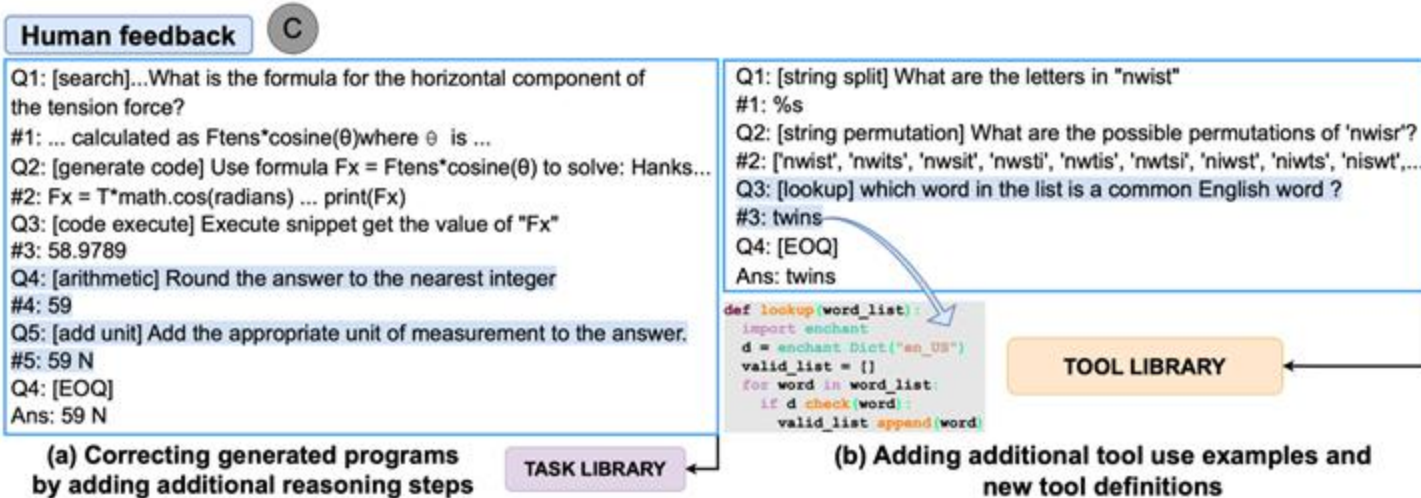


Figure 3: Human feedback to ART shown for (a) PQA where reasoning steps are added to the program and; (b) Word unscrambling where tool library is augmented with a new lookup tool.

Table 3: ART performance on BigBench tasks and MMLU tasks. (¹ Human-crafted CoT (Wei et al., 2022; Suzgun et al., 2022), ⁵ InstructGPT (Ouyang et al., 2022), ⁶ Scaled instruction finetuning (Chung et al., 2022), ⁷ Code-davinci-002 (Chen et al., 2021)).



Using Tools and Act



Tools and Act

- ❑ Expanding LLMs' capabilities to access and leverage **external knowledge** and **computational resources** beyond what is stored in their own weights.
- ❑ This integration, via special tokens, allows LLMs to perform tasks like:
 - Exact computation
 - Symbolic reasoning
 - Information retrieval, etc.
- ❑ Tools can range from:
 - Calling (another) LLM or other programs like Python interpreter
 - Querying (vector) databases, knowledge graphs, or search engines
 - Often referred to as “information retrieval”

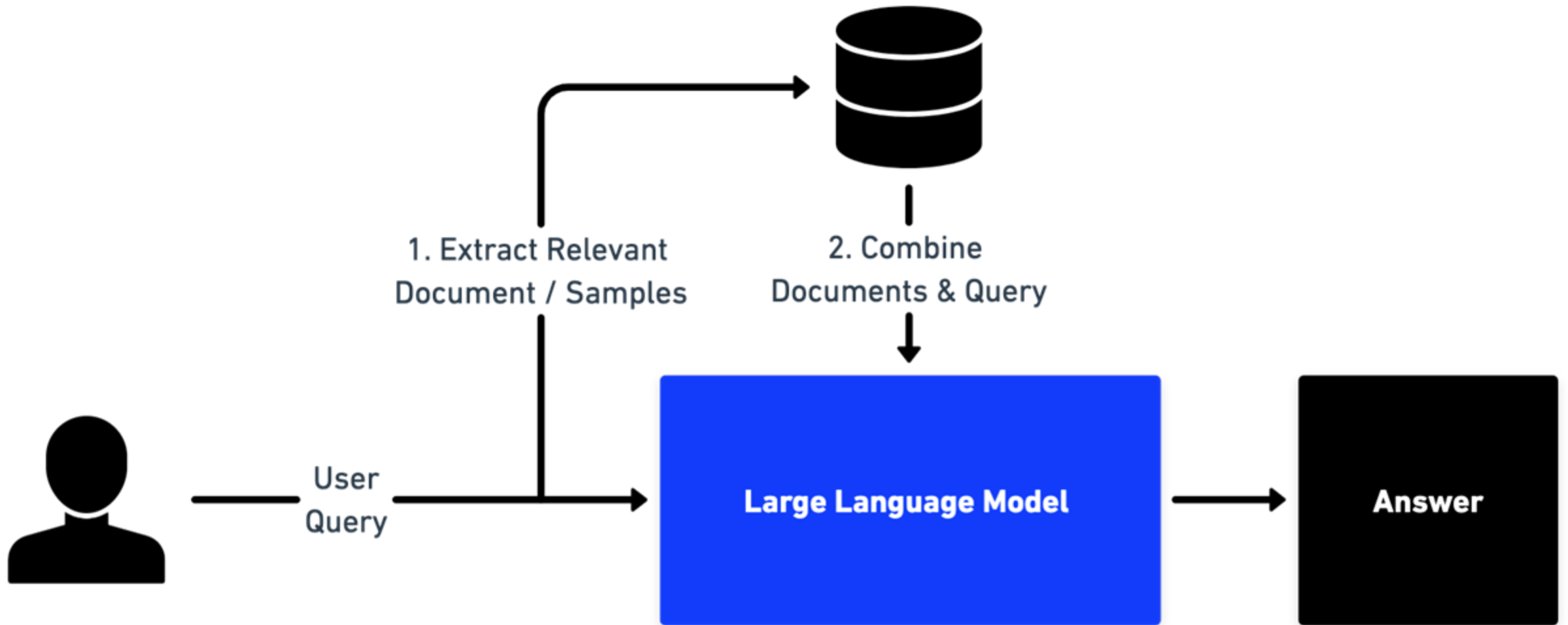


Tools and Act

- Use Google Search for document retrieval to augment LLMs. [[Lazaridou et al. 2022](#)]
 - Given a question, clean text (paragraph) is extracted out of 20 URLs returned by Google.
 - Paragraphs are ranked by TF-IDF based cosine similarity between evidence paragraphs and the query.
 - Only the most relevant paragraph is used in the prompt to produce an answer



Augmenting LLMs with Retrieval



Augmenting LLMs with Retrieval

- “Internal retrieval” from LLMs could be also beneficial
 - Generate knowledge about a topic before answering the question [\[Liu et al. 2022\]](#)

```
Generate some knowledge about the input. Examples:
```

```
Input: What type of water formation is formed by clouds?
```

```
Knowledge: Clouds are made of water vapor.
```

```
Input: {question}
```

```
Knowledge:
```



Augmenting LLMs

- “Internal retrieval” from LLMs could be also beneficial
 - Iterative prompting of LLM for text refinement task [\[Schick et al. 2022\]](#)
 - The “plan” can come from human users too.

Iteration 0

Text: Brittney Reese (born September 9, 1986 in Gulfport, Mississippi) is an American long jumper.

<LM>

Plan: Remove incorrect information

Edit: Brittney Reese (born September 9, 1986 in ~~Gulfport, Mississippi~~) is an American long jumper.

</LM>

Iteration 1

Text: Brittney Reese (born September 9, 1986) is an American long jumper.

<LM>

Plan: Add information about her career

Edit: Brittney Reese (born September 9, 1986) is an American long jumper, who competed at the 2008 Summer Olympics, and is a 4-time World Champion .

</LM>

Iteration 2

Text: Brittney Reese (born September 9, 1986) is an American long jumper, who competed at the 2008 Summer Olympics, and is a 4-time World Champion.

<LM>

Plan: Add her birthplace

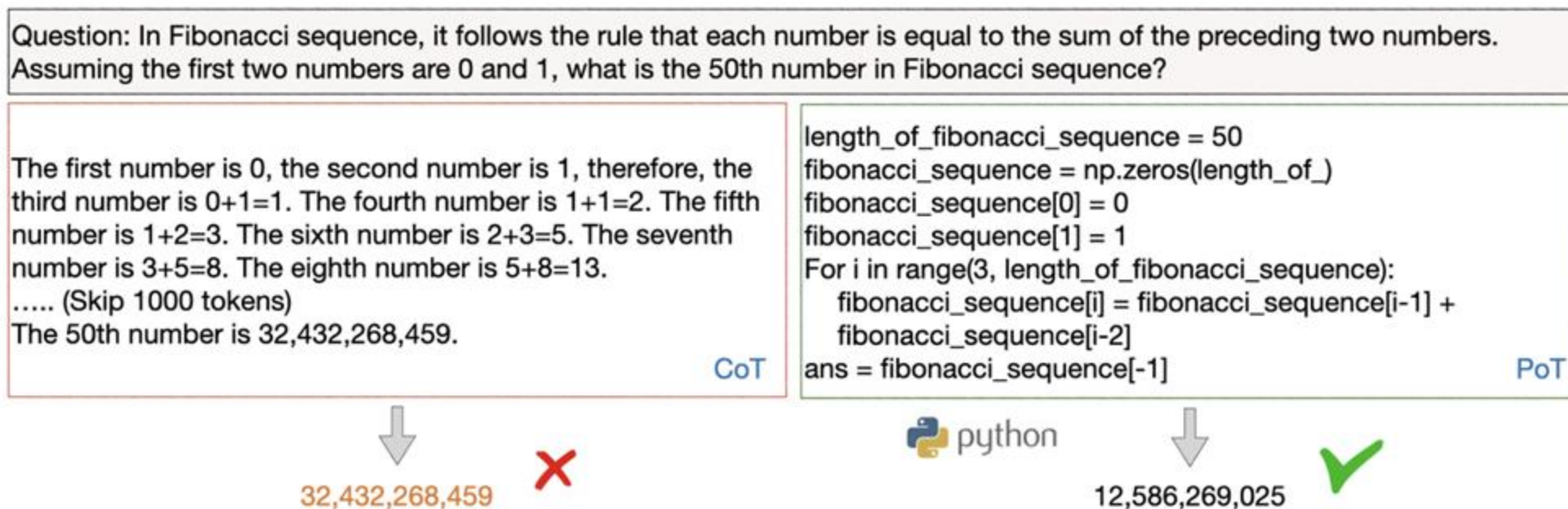
Edit: Brittney Reese (born September 9, 1986 in Inglewood, California) is an American long jumper, who competed at the 2008 Summer Olympics, and is a 4-time World Champion.

</LM>

Figure 5: Iterative prompting example using PEER (Schick et al., 2022), a LM trained to produce a plan of action and edit to the input text at each step. This process can be repeated until the generated text requires no further updates. <LM> denotes the start of the LM’s output to the prompt, while </LM> denotes the end.

Augmenting LLMs with Programming Language

- Ask LLM to **generate programming language statements** to resolve natural language reasoning problems, hence **offloading the solution step to a runtime** such as a Python interpreter.



PAL (Program-aided language models); [Gao et al. 2022](#) and PoT (Program of Thoughts prompting); [Chen et al. 2022](#)



Augmenting LLMs with External APIs

- ❑ **TALM** (Tool Augmented Language Models; [Parisi et al. 2022](#)): LM augmented with text-to-text API calls.
 - LM is guided to generate **|tool-call** and **tool input text** conditioned on task input text to construct API call requests
 - When **|result** shows up, the specified tool API is called and the returned result gets appended to the text sequence as **|output** token.

An abstract task:

task input text **|tool-call** tool input text **|result** tool output text **|output** task output text

A weather task:

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



Augmenting LLMs with External APIs

- **Toolformer** ([Schick et al. 2023](#)) use external tools via simple APIs, which is built in a *self-supervised manner* and only requires a handful of demonstrations for each API.
 - **Calculator** to help LM with the lack of precise math skills;
 - **Q&A system** to help with unfaithful content and hallucination;
 - **Search engine** to provide up-to-date information after pretraining cut off time;
 - **Translation system** to improve performance on low resource language;
 - **Calendar** to make LM be aware of time progression.



Toolformer Training

Step 1:

Prompting to annotate potential API calls.

Ask a pre-trained LM to annotate a dataset via few-shot learning with API call usage examples.

The New England Journal of Medicine is a registered trademark of [QA("Who is the publisher of The New England Journal of Medicine?") → Massachusetts Medical Society] the MMS.

Out of 1400 participants, 400 (or [Calculator(400 / 1400) → 0.29] 29%) passed the test.

The name derives from "la tortuga", the Spanish word for [MT("tortuga") → turtle] turtle.

The Brown Act is California's law [WikiSearch("Brown Act") → The Ralph M. Brown Act is an act of the California State Legislature that guarantees the public's right to attend and participate in meetings of local legislative bodies.] that requires legislative bodies, like city councils, to hold their meetings open to the public.

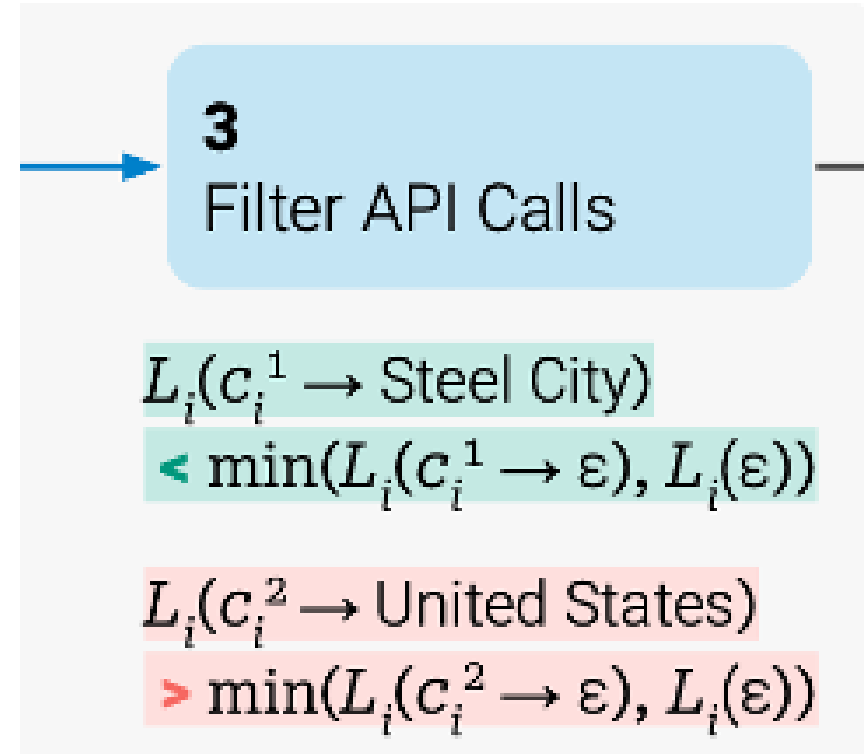


Toolformer Training

Step 2:

Filter annotations based on whether API calls help predict future tokens.

Use a self-supervised loss to decide which API calls are actually helpful.



$$L_i^+ = L_i(e(c_i, r_i))$$

$$L_i^- = \min(L_i(\epsilon), L_i(e(c_i, \epsilon)))$$

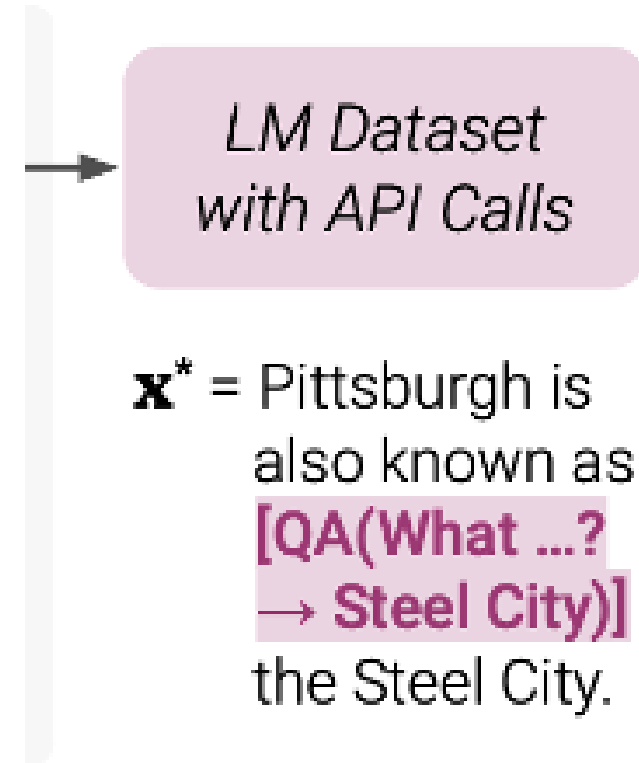


Toolformer Training

Step 3:

Fine-tune LM on this annotated dataset.

The training data is a combination of the original dataset and its augmented version.

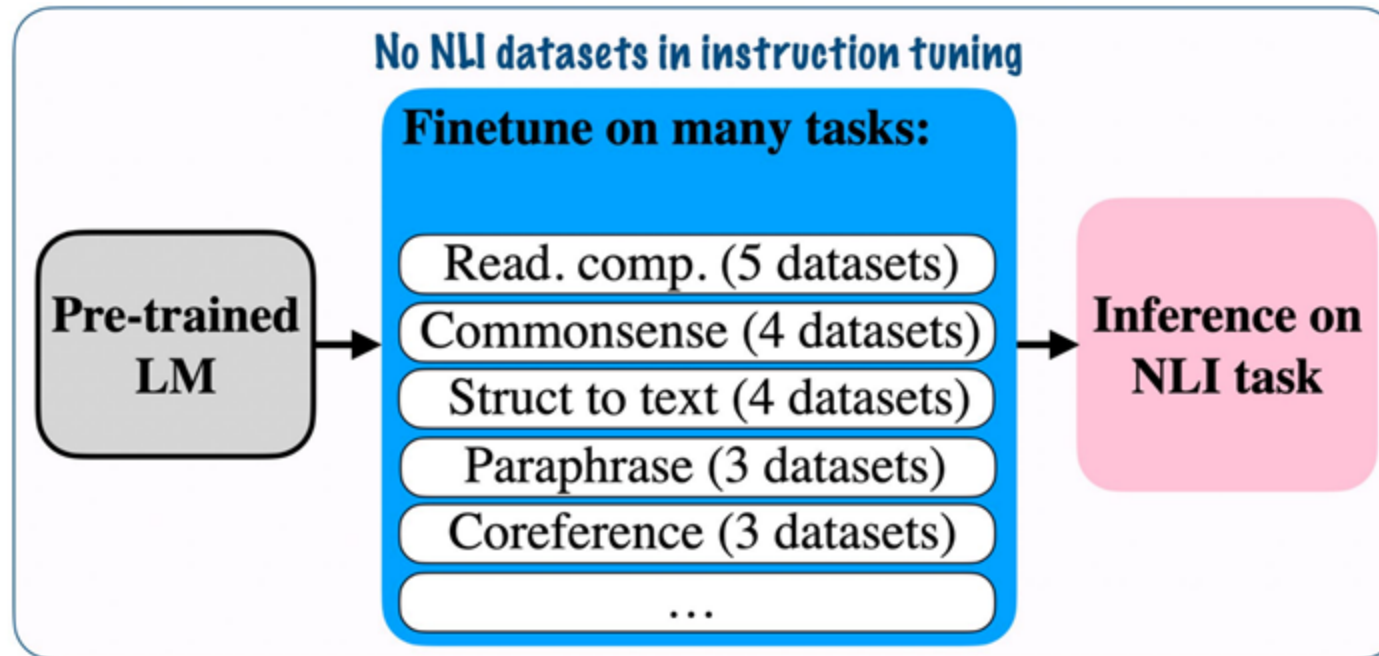


Evaluation



Evaluating on unseen tasks

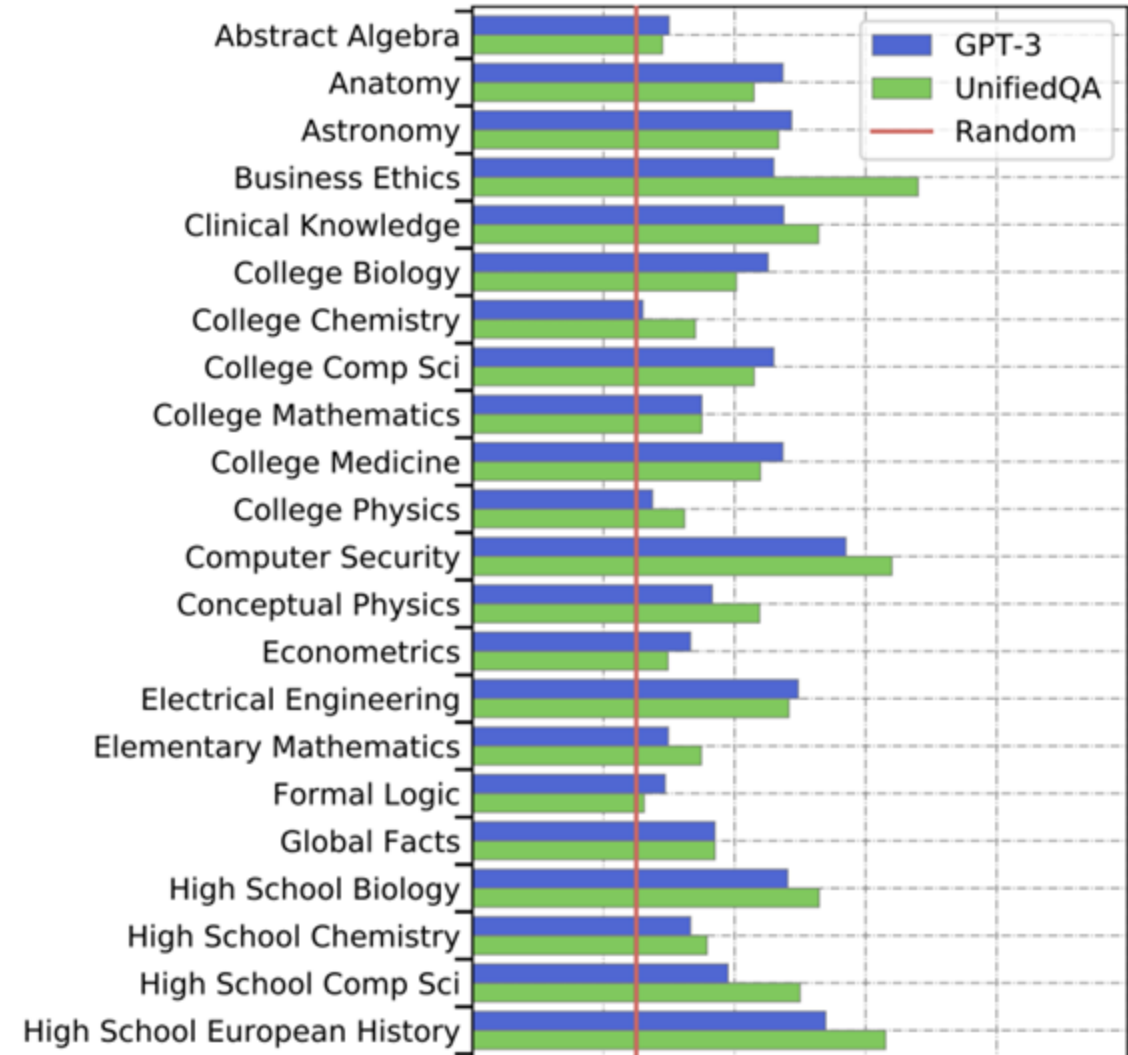
- We evaluate on “unseen” or “zero-shot” tasks where no datasets from that task were seen during instruction tuning.
→ **Emergent abilities of LLMs!**



Evaluating on new benchmarks

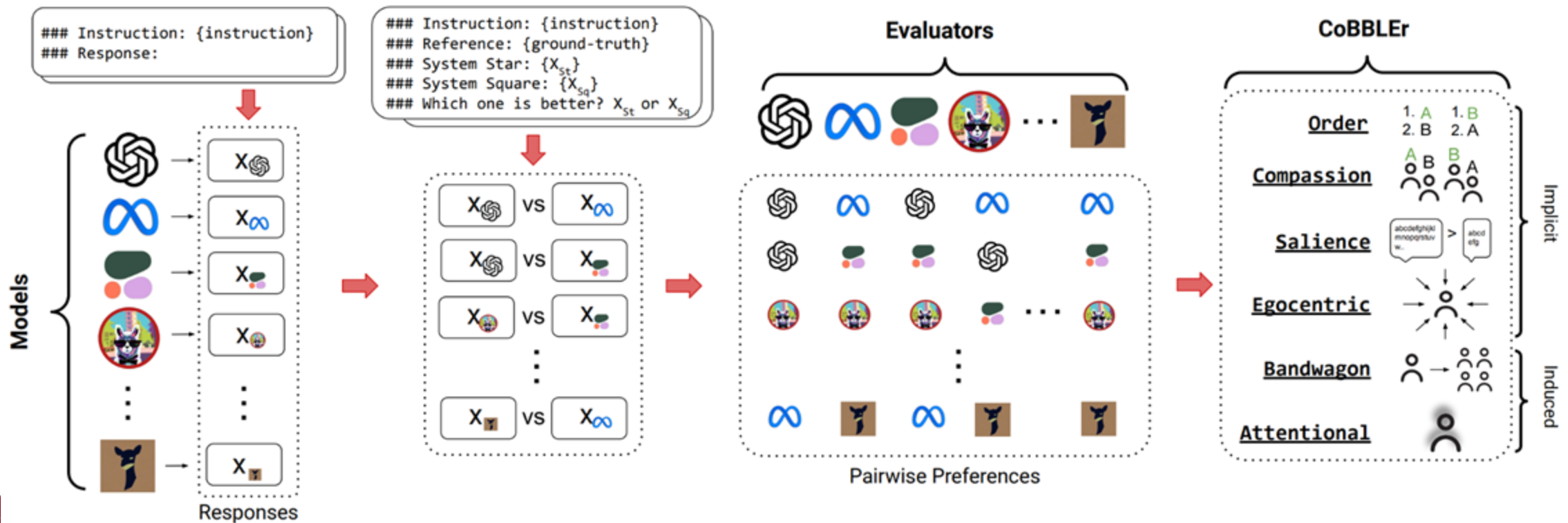
Massive Multitask Language Understanding (MMLU) [Hendrycks et al., 2021]

New benchmarks consisting of 57 diverse *knowledge intensive* tasks



Evaluating on new benchmarks

- ▣ **CoBBLer** [[Koo et al. 2023](#)], “Benchmarking Cognitive Biases in Large Language Models as Evaluators”
 - For evaluating LLMs for their capabilities as **unbiased automatic evaluators**



Evaluating on new benchmarks

□ CoBBLEr [[Koo et al. 2023](#)]

- Implicit biases: general prompt setting
- Induced biases: try to induce undesired behaviors akin to adversarial attacks

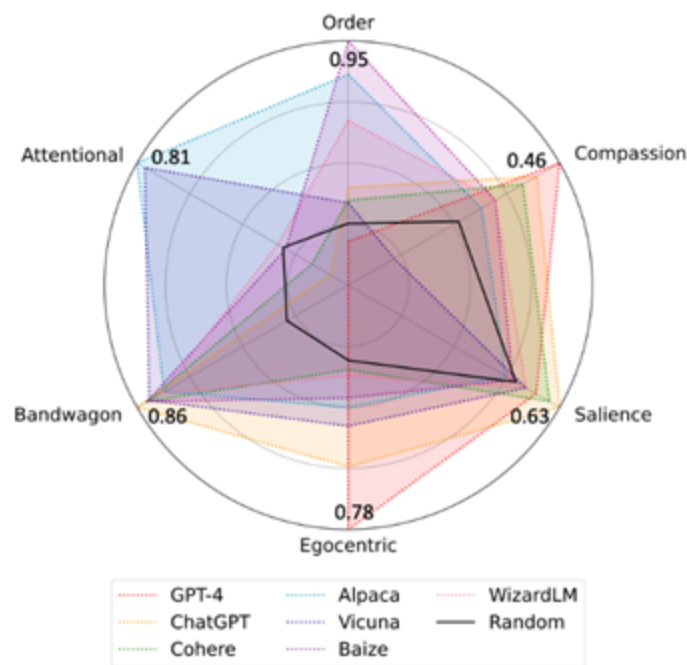
Bias	Bias Behavior	Example
ORDER BIAS	The tendency to give preference to an option based on their order (e.g. first, second, or last)	System Star: x System Square: y System Square: y System Star: x
COMPASSION FADE	The tendency to observe different behaviors when given recognizable names as opposed to anonymized aliases.	Model Alpaca: x Model Vicuna: y Model Vicuna: y Model Alpaca: x
EGOCENTRIC BIAS	The inclination to prioritize one's own responses regardless of response quality.	Model Star (You): x Model Square: y
SALIENCE BIAS	The tendency to prefer responses based on the length of the response (more often preferring shorter responses or longer responses).	System Star: The quick brown fox jumps over the lazy dog. System Square: The fox jumped.
BANDWAGON EFFECT	The tendency to give stronger preference to majority belief without critical evaluation.	85% believe that System Star is better.
ATTENTIONAL BIAS	The inclination to give more attention to irrelevant or unimportant details.	System Square likes to eat oranges and apples



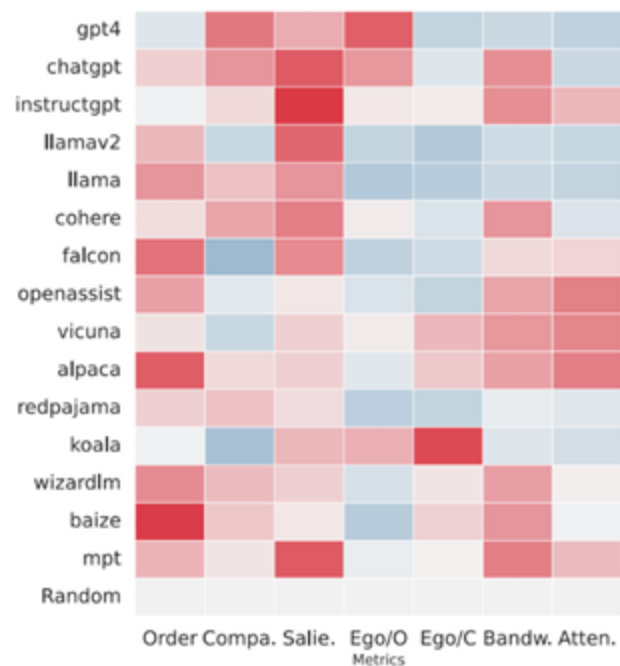
Evaluating on new benchmarks

CoBBLER [Koo et al. 2023]

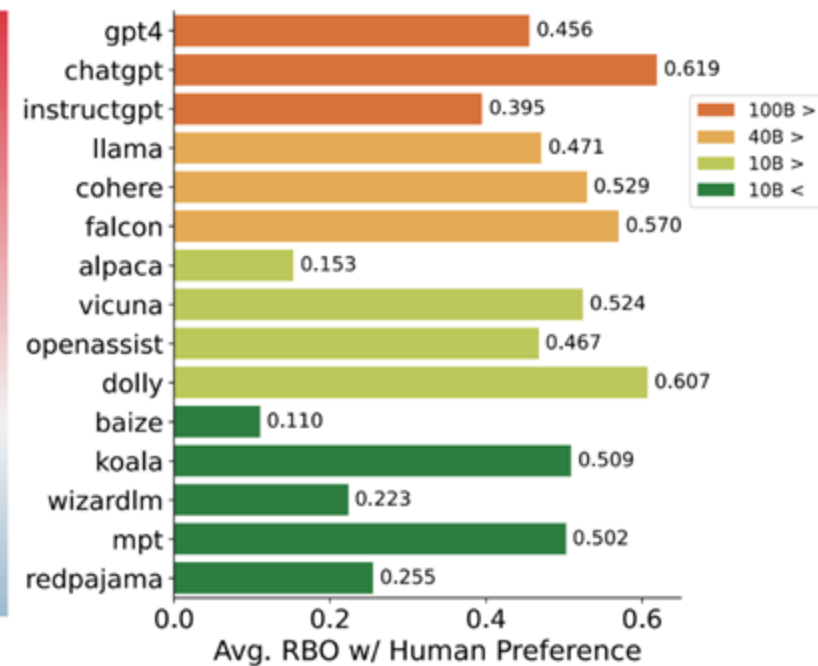
- Even popular LLMs like GPT-4 are shown to have biases!
- So be careful when using them as evaluators in replacement of human workers.



(a) Proportion of biased evaluations



(b) Heatmap of bias intensity



(c) Correlation with human judgment

Concluding Remarks

□ On alignment

- Training LLMs on various tasks enhances their ability to execute instructions for new tasks.
- Instruction-tuned LLMs exhibit better zero-shot and few-shot capability.
- The number of instruction tuning clusters and the scale of the model are important to the performance of LLMs.
- Many new benchmarks have been developed to test the models' capability as well as to alert their pitfalls.

□ On augmentation

- The emergence of reasoning abilities in LLMs facilitates the incorporation of various tools and knowledge bases, thereby greatly enhancing their overall capability.
- Deciding which tools to use and what relevant external knowledge to integrate is the key.
- As tasks get more complex, LLMs' long-term planning capability is put to test.

