Deep Learning

Deeper, Better, _____ , Stronger than Machine Learning

Valentin Barriere

Universidad de Chile – DCC

CC6204, Primavera 2024

Large Generative Language Models

Outline : Introduction

What is a language model?

The classic definition of a language model (LM) is a **probability distribution over sequences of tokens**. Suppose we have a vocabulary $\mathcal{V}$ of a set of tokens. A language model p assigns each sequence of tokens $x_1, \ldots, x_L \in \mathcal{V}$ a probability (a number between 0 and 1):

$$p(x_1, \ldots, x_L)$$

The probability intuitively tells us how "good" a sequence of tokens is.

Difficulty

The ability to assign (meaningful) probabilities to all sequences **requires extraordinary (but implicit) linguistic abilities and world knowledge**.

Autoregressive Language Models

General Definition

An auto-regressive model can be used to describe certain time-varying processes in nature, economics, behavior. It specifies that the output variable depends ~~**linearly**~~ on its own previous values and on a stochastic term: $X_t = \sum_{i=1}^{p} \varphi_i X_{t-i} + \epsilon_t$

$$p(x_{1:L}) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_L \mid x_{1:L-1}) = \prod_{i=1}^{L} p(x_i \mid x_{1:i-1}).$$

In particular, $p(x_i \mid x_{1:i-1})$ is a **conditional probability distribution** of the next token x_i given the previous tokens $x_{1:i-1}$

$$\begin{aligned} p(\text{the, mouse, ate, the, cheese}) = {} & p(\text{the}) \\ & p(\text{mouse} \mid \text{the}) \\ & p(\text{ate} \mid \text{the, mouse}) \\ & p(\text{the} \mid \text{the, mouse, ate}) \\ & p(\text{cheese} \mid \text{the, mouse, ate, the}) \end{aligned}$$

Temperature and generation

Now to generate an entire sequence $x_{1:L}$ from an autoregressive language model p, we sample one token at a time given the tokens generated so far:

$$\text{for } i = 1, \ldots, L: \qquad x_i \sim p(x_i \mid x_{1:i-1})^{1/T}$$

where $T \geq 0$ is a **temperature** parameter that controls how much randomness we want from the language model:

- $T = 0$: deterministically choose the most probable token x_i at each position
- $T = 1$: sample "normally" from the pure language model
- $T = \infty$: sample from a **uniform distribution** over the entire vocabulary $\mathcal{V}$

Temperature

Conditional generation

More generally, we can perform conditional generation by specifying some prefix sequence $x_{1:i}$ (called a **prompt**) and sampling the rest $x_{i+1:L}$ (called the **completion**).

For example, generating with $T = 0$ produces:

$$\underbrace{\text{the, mouse, ate}}_{\text{prompt}} \overset{T=0}{\rightsquigarrow} \underbrace{\text{the, cheese}}_{\text{completion}}$$

Thermodynamics and Entropy

- A measurement of a **degree of randomness** of energy in a system
- The lower the entropy, the more ordered and less random it is

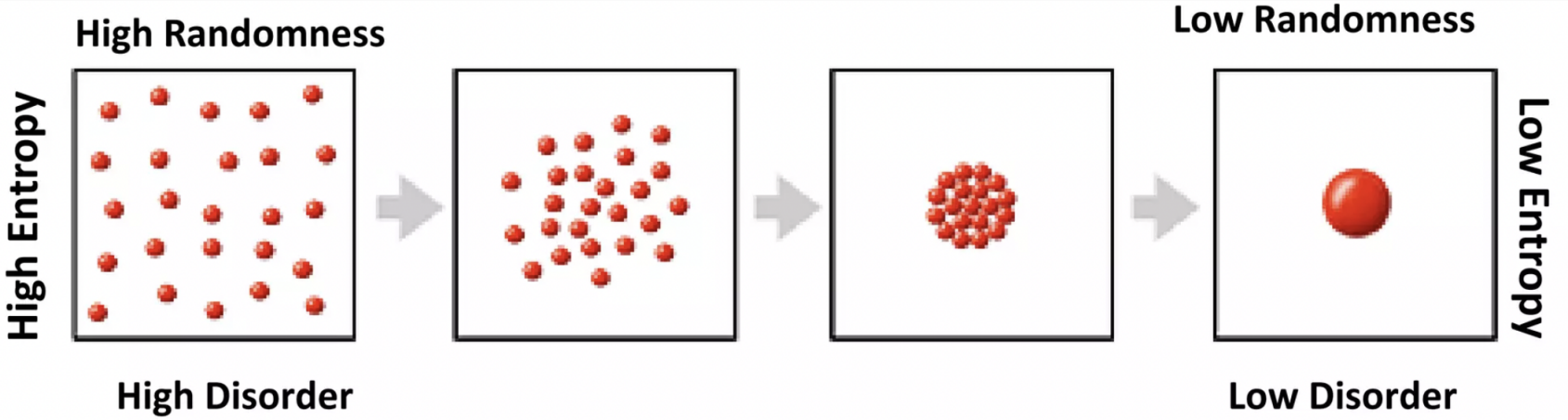


Figure 1: Entropy in thermodynamics

With temperature and LM

The higher the temperature of the room is, the more entropy it will create. Using $T = \infty$ we get a uniform distribution of the probabilities over $\mathcal{V}$, which is the most random possible.

Information theory and Entropy

The entropy of a distribution is defined as:

$$H(p) = \sum_x p(x) \log \frac{1}{p(x)}.$$

The entropy measures the expected number of bits **any algorithm** needs to encode (compress) a sample $x \sim p$ into a bitstring:

$$\text{the mouse ate the cheese} \Rightarrow 0001110101.$$

- The lower the entropy, the more "structured" the sequence is, and the shorter the code length.
- Intuitively, $\log \frac{1}{p(x)}$ is the length of the code used to represent an element x that occurs with probability $p(x)$.
- If $p(x) = \frac{1}{8}$, we should allocate $\log_2(8) = 3$ bits (equivalently, $\log(8) = 2.08$ nats).

Cross-Entropy

Cross entropy

$$H(p, q) = \sum_x p(x) \log \frac{1}{q(x)},$$

measures the expected number of bits needed to encode a sample $x \sim p$ using the compression scheme given by the model q (representing x with a code of length $\frac{1}{q(x)}$).

A crucial property is that the cross entropy $H(p, q)$ upper bounds the entropy $H(p)$:

$$H(p, q) \geq H(p),$$

which means that we can estimate $H(p, q)$ by constructing a (language) model q with only samples from the true data distribution p, whereas $H(p)$ is generally inaccessible if p is English.

Perplexity

The joint probability of a sequence depends on its length and thus **goes to zero** as the length grows, which makes it hard to track. Intuitively we would like to average the per token probabilities $p(x_i \mid x_{1:i-1})$.

The perplexity is defined like this:

$$\text{perplexity}_p(x_{1:L}) = \exp\left(\frac{1}{L}\sum_{i=1}^{L}\log\frac{1}{p(x_i \mid x_{1:i-1})}\right)$$

Rewriting the Entropy between the real text and the LM predictions:

$$H(GT, LM) = \sum_{i=1}^{L}\sum_{w\in\mathcal{V}}\mathbb{1}_{x_i}(w)\log\frac{1}{p(w \mid x_{1:i-1})}$$

We obtain: $\text{perplexity}_p(x_{1:L}) = \exp\left(\frac{1}{L}H(GT, LM)\right)$ or, $H = \log \text{perplexity}$

Perplexity

The joint probability of a sequence depends on its length and thus **goes to zero** as the length grows, which makes it hard to track. Intuitively we would like to average the per token probabilities $p(x_i \mid x_{1:i-1})$.

The perplexity is defined like this:

$$\text{perplexity}_p(x_{1:L}) = \exp\left(\frac{1}{L}\sum_{i=1}^{L}\log\frac{1}{p(x_i \mid x_{1:i-1})}\right)$$

Rewriting the Entropy between the real text and the LM predictions:

$$H(GT, LM) = \sum_{i=1}^{L}\sum_{w\in\mathcal{V}}\mathbb{1}_{x_i}(w)\log\frac{1}{p(w \mid x_{1:i-1})}$$

We obtain: $\text{perplexity}_p(x_{1:L}) = \exp\left(\frac{1}{L}H(GT, LM)\right)$ or, $H = \log \text{perplexity}$

N-gram models

In an **n-gram model**, the prediction of a token x_i only depends on the last $n-1$ characters $x_{i-(n-1):i-1}$ rather than the full history:

$$p(x_i \mid x_{1:i-1}) = p(x_i \mid x_{i-(n-1):i-1}).$$

For example, a trigram ($n = 3$) model would define:

$$p(\text{cheese} \mid \text{the, mouse, ate, the}) = p(\text{cheese} \mid \text{ate, the}).$$

These probabilities are computed based on the number of times various n-grams (e.g., ate the mouse and ate the cheese) occur in a large corpus of text.

Problem

When n is too big, you cannot calculate the n-gram is it relies on occurences and long sentences tend to be uniques.

Neural language models

With the introduction of neural network, $p(x_i \mid x_{i-(n-1):i-1})$ is given by a neural network:

$$p(\text{cheese} \mid \text{ate}, \text{the}) = \text{some-neural-network}(\text{ate}, \text{the}, \text{cheese}).$$

Note that the context length is still bounded by n, but it is now **statistically feasible** to estimate neural language models for much larger values of n.

Main techniques

- Skip-gram, like word2vec model
- RNN-LSTM, allowed the conditional distribution of a token x_i to depend on the **entire context** $x_{1:i-1}$ (effectively $n = \infty$), but these were hard to train.
- Transformers: more recent architecture (developed for machine translation in 2017) that again returned to having fixed context length n, but were much **easier to train** at large scale

What kinds of things does pretraining learn?

- Beauchef Campus is located in _____, Chile. [Trivia]
- I put _____ fork down on the table. [syntax]
- The woman walked across the street, checking for traffic over _____ shoulder. [coreference]
- I went to the ocean to see the fish, turtles, seals, and _____. [lexical semantics/topic]
- Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was _____. [sentiment]
- Iroh went into the kitchen to make some tea. Standing next to Iroh, Zuko pondered his destiny. Zuko left the _____. [some reasoning – this is harder]
- I was thinking about the sequence that goes 1, 1, 2, 3, 5, 8, 13, 21, _____ [some basic arithmetic; they don't learn the Fibonnaci sequence]

Outline : Abilities and In-Context-Learning

GPT-3 Performances on TriviaQA [19]

Task: given a trivia question, generate the answer.

Adaptation We define a prompt based on the training instances (if any) and the question, and take the completion as the predicted answer (demo):

```
Q: 'Nude Descending A Staircase' is perhaps the most famous painting by which 20th century artist?
A: Marcel Duchamp
```

Model	**Accuracy (%)**
RAG	68.0
GPT-3 (zero-shot)	64.3
GPT-3 (few-shot)	71.2

GPT-3 Performances on Translation

Task: translate a sentence in a source language (e.g., German) to sentence in a target language (e.g., English)

Model	**Accuracy (%)**
SOTA (supervised)	40.2
GPT-3 (zero-shot)	27.2
GPT-3 (few-shot)	40.6

- Even without supervised training data, GPT-3 matches the state-of-the-art of a fully-supervised system!
- Results from English to a foreign language is much worse

Emergent Abilities: Zero- and Few-Shot Learning

- **Zero-shot learning**: Solving tasks without any specific examples.
- **Few-shot learning**: Achieving high performance with minimal examples [5].
- **In-context learning**: GPT-3 uses context provided in input, without updating model weights.

Emergent Abilities: Zero- and Few-Shot Learning

- **Zero-shot learning**: Solving tasks without any specific examples.
- **Few-shot learning**: Achieving high performance with minimal examples [5].
- **In-context learning**: GPT-3 uses context provided in input, without updating model weights.

$\Rightarrow$ **Models solve tasks based on input sequences on not on training data updating their weights, a shift from traditional training paradigms.**

Zero- and Few-shot

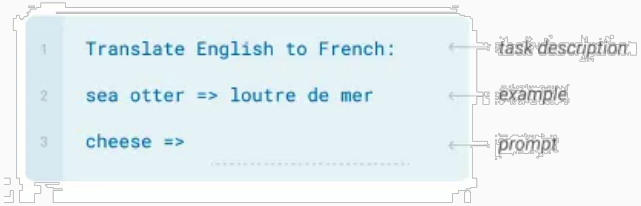
The three settings we explore for in-context learning

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

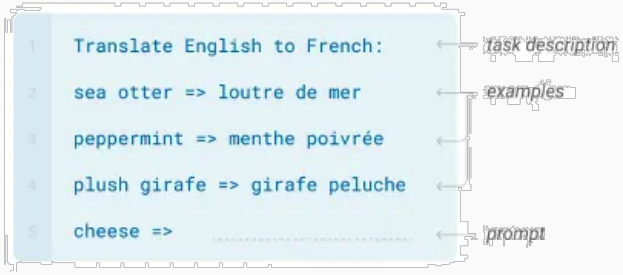
One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.

Emergent abilities

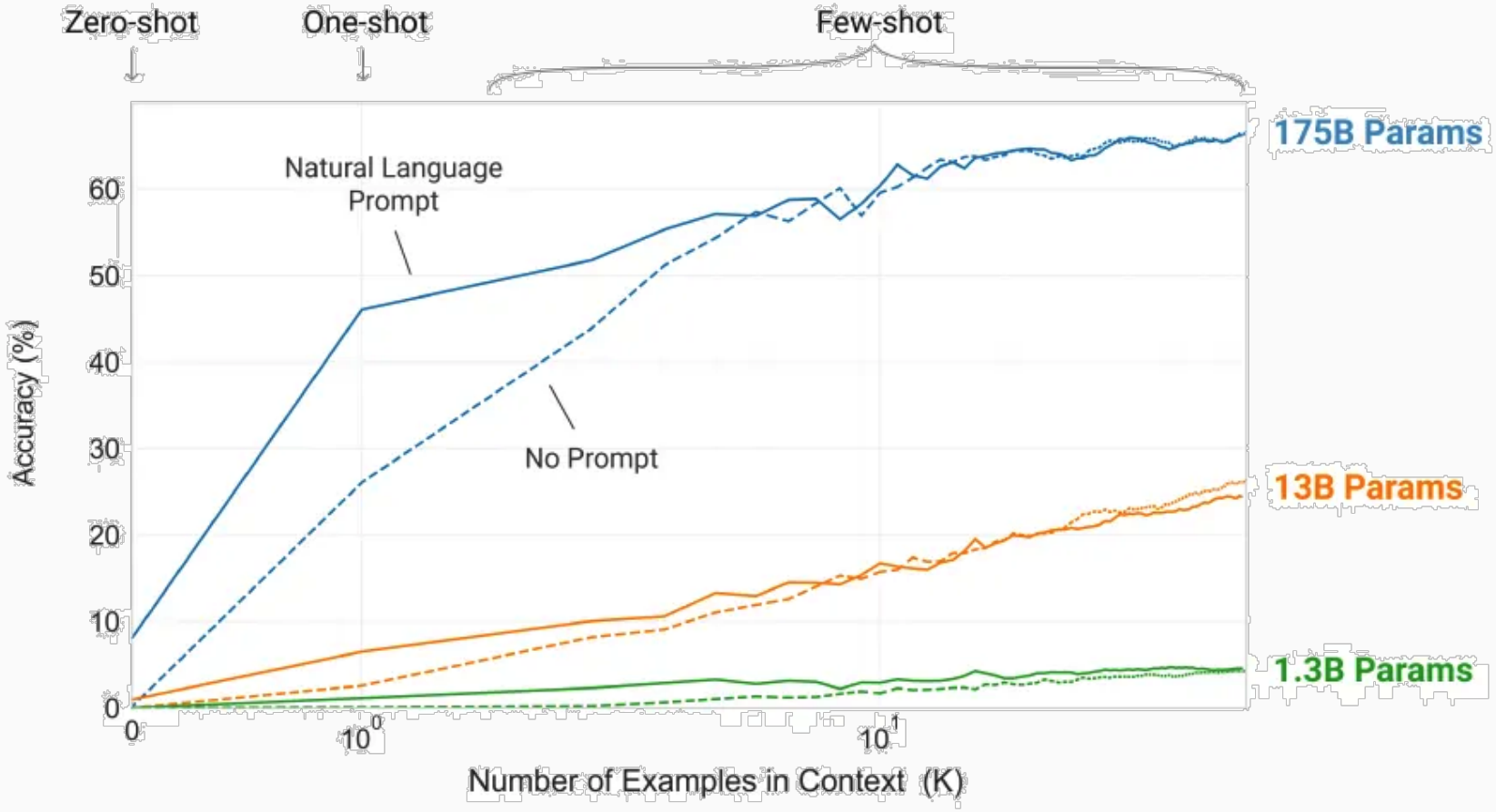
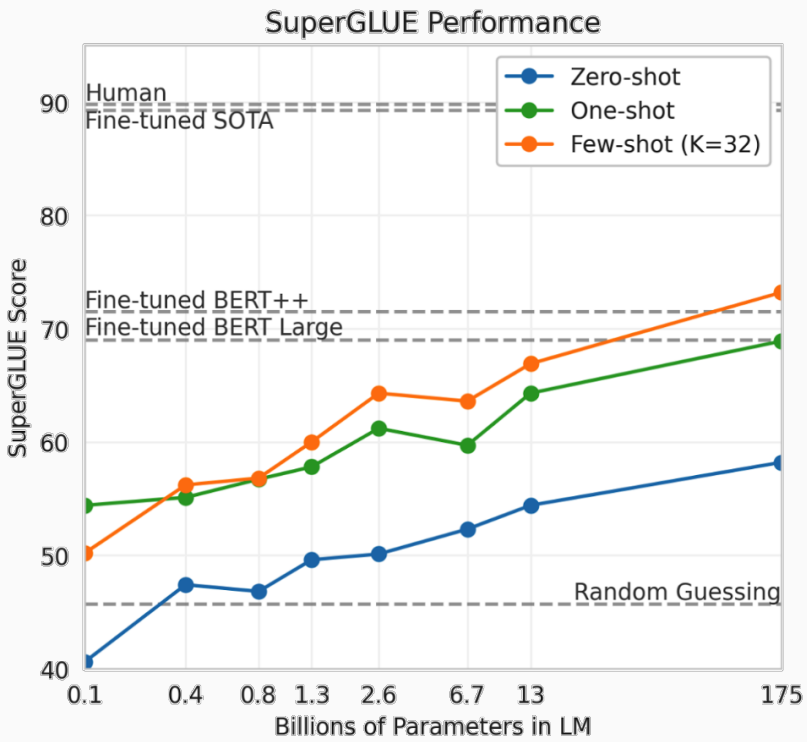
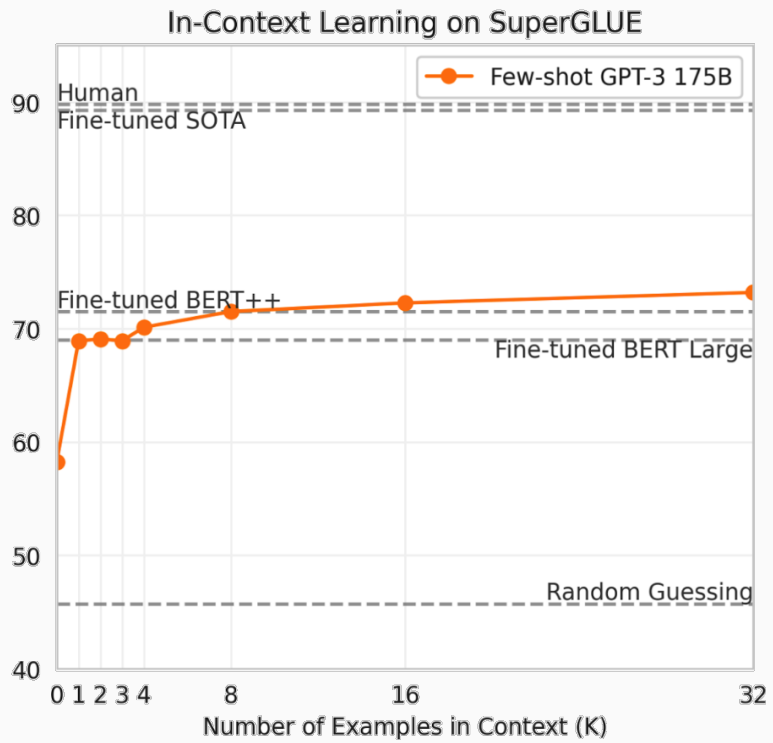


Figure 2: A simple task is learnt which requiring the model to remove extraneous symbols from a word.

From a certain size, the model is able to tackle many unseen tasks

Zero- and Few-shots abilities





The performances are better with bigger models, and using more examples.

Emergent Abilities

Definition from [59]

An ability is considered to be emergent if it is not present in smaller models but is present in larger models. Thus, **emergent abilities cannot be predicted simply by extrapolating the performance of smaller models**

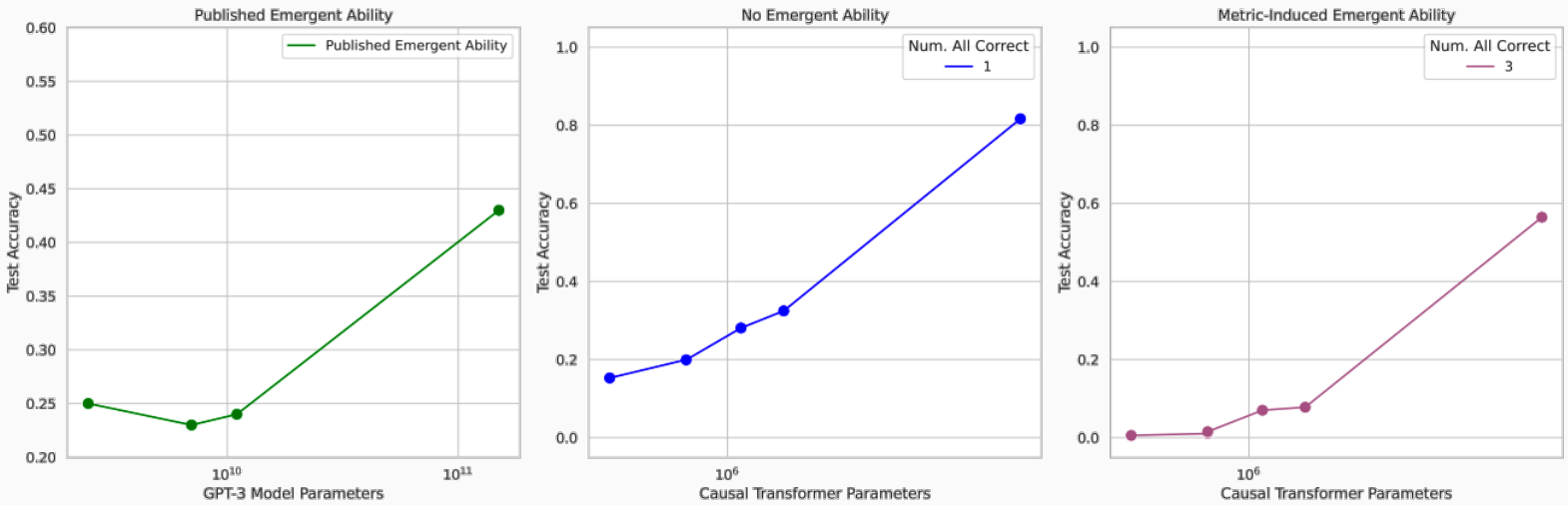
Emergent Abilities

Definition from [59]

An ability is considered to be emergent if it is not present in smaller models but is present in larger models. Thus, **emergent abilities cannot be predicted simply by extrapolating the performance of smaller models**

But actually..... Emergent abilities a mirage? [43]

This behavior is due to the **choice of metric** rather than due to fundamental changes in model behavior with scale! With different metrics, you can see the **abilities smoothly appearing**!!



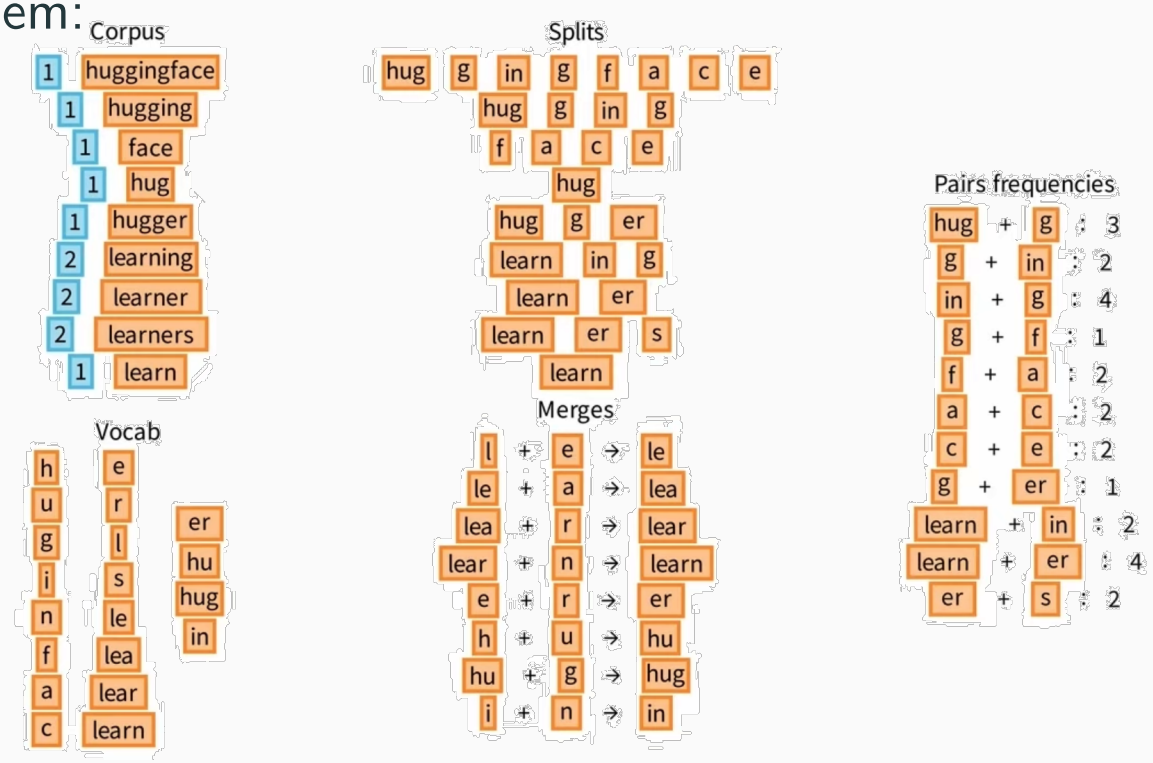
Outline : Tokenization and Training

Tokenization

Tokenization

Converts text into smaller units (tokens) for model processing (before creating embeddings). It is a crucial step **affecting performances and processing time**.

Example: GPT-3 uses Byte Pair Encoding (BPE; [45]), an efficient **tokenizer learned over data** to find the most relevant basic units, by merging them:



Training Objectives

Models can be pre-trained using various objectives, such as:

- **Autoregressive Language Modeling**: predicting the next token given the previous context.
- **Masked Language Modeling**: predicting a masked token inside a sentence
- **Next Sentence Prediction**: predicting if two sentences follow one another

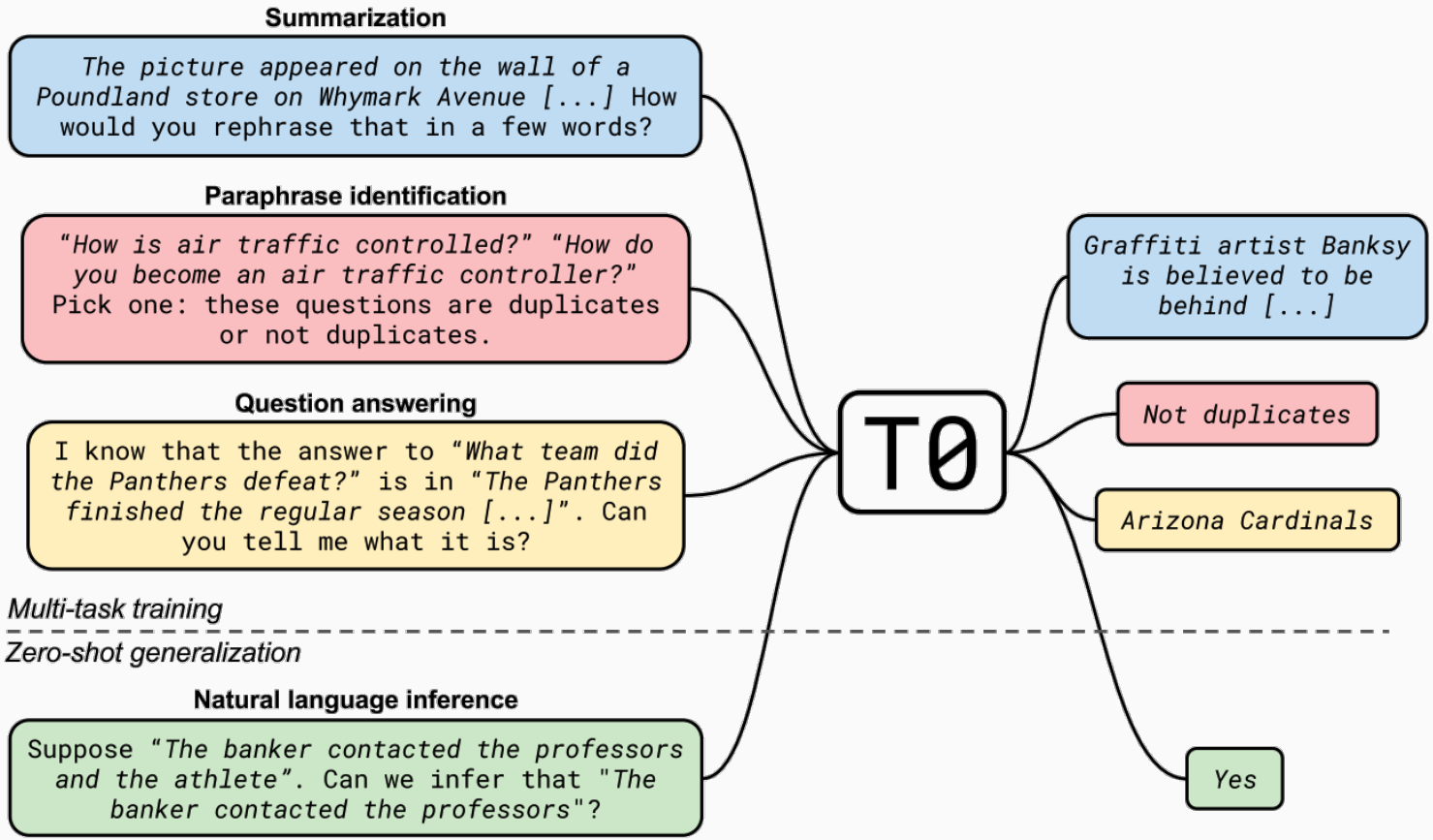
Then they are fine-tuned for several objectives:

- **Instruction-finetuning** improves model performance on instruction-following tasks [42]
- **Model Alignment** over users' preferences

Outline : Instructions and Alignments

Instructions

- Instruction-based prompts improve model performance [58]
- Clear, well-structured instructions guide the model towards accurate results [9]
- Instruction tuning helps models **generalize better on new tasks** [42]



Alignment I

- Alignment ensures models behave according to human values [2]
- **Reinforcement learning from human feedback** (RLHF) is a key approach to learn what users want [33]

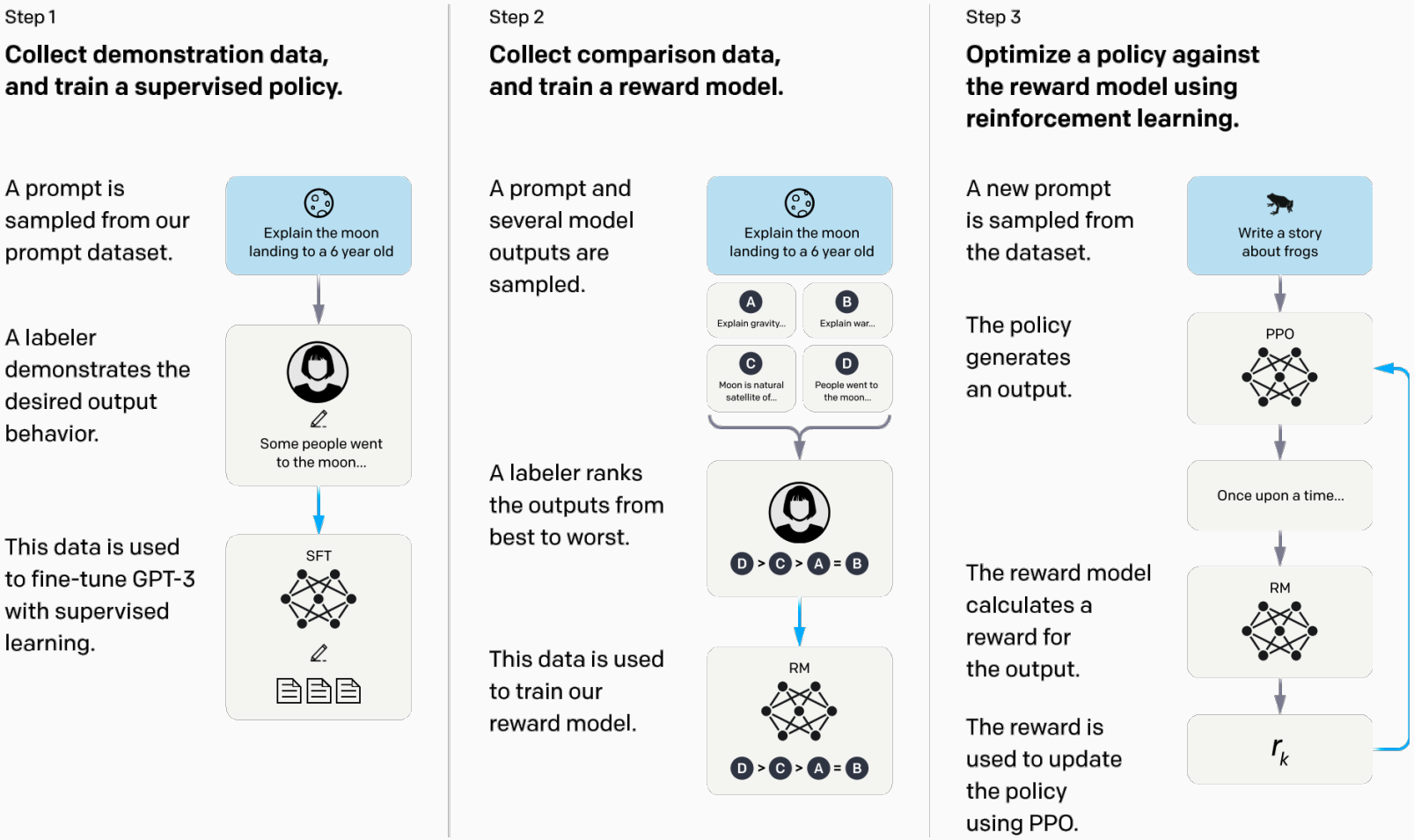


Figure 4: Three steps: Supervised Learning, Reward Model, RLHF

Alignment II: RLHF

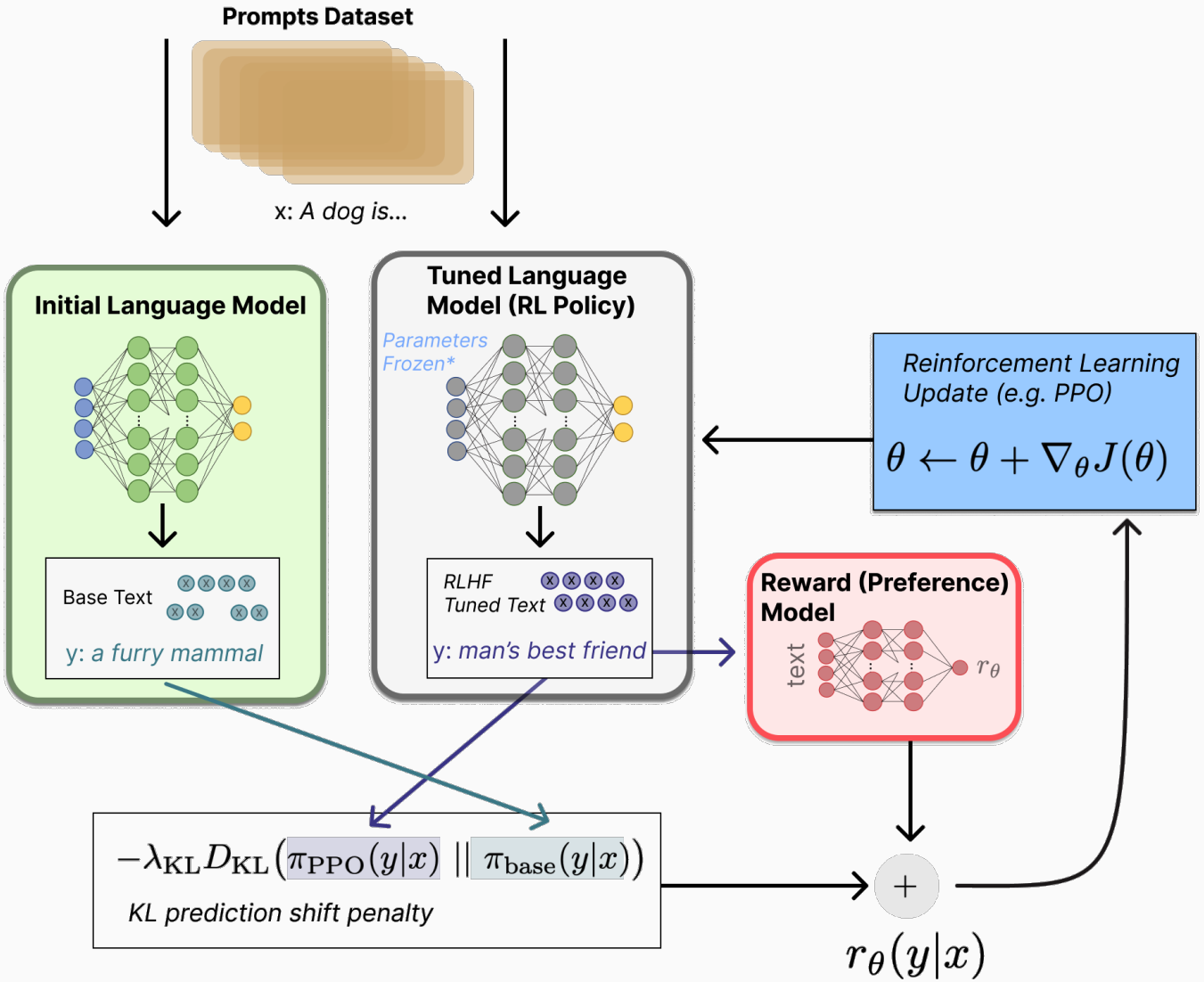


Figure 5: RLHF in details. In reality the RL policy generates text, which is fed into the initial model to produce its relative probabilities for the KL penalty.

Alignment III

- Actually alignment can be obtained using 1,000 chery-picked examples! [68]
- Proximal Policy Optimization (PPO) was the original RLHF loss, but there are more efficient like Direct Preference Optimization (DPO) [40]
- Processes for reasoning using internal thinking steps like Thought Preference Optimization (TPO) [61]

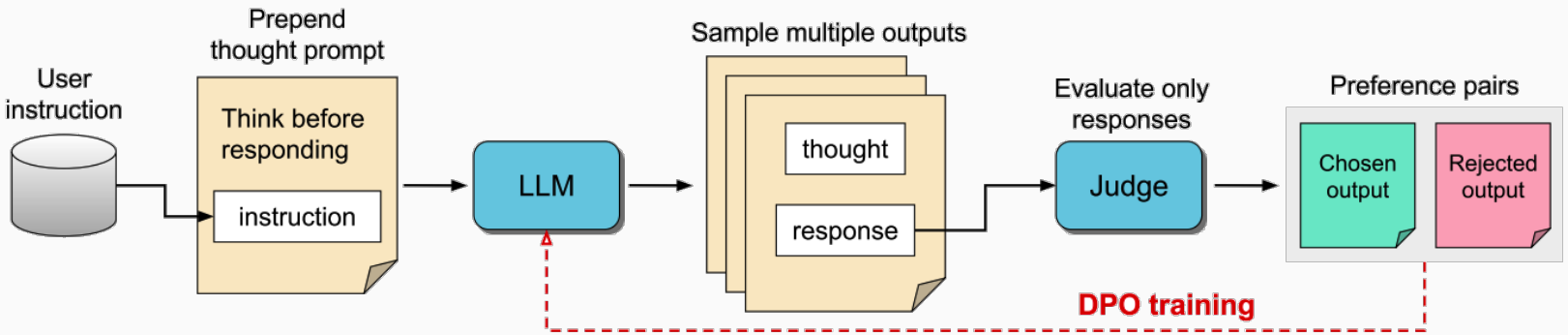


Figure 1: **Thought Preference Optimization:** We start by prompting the LLM to generate thoughts before its response. After sampling different outputs, we feed the response parts to the judge model which determines the best and worst ones. Then we use the corresponding full outputs as chosen and rejected pairs for DPO optimization. We perform multiple iterations of this training.

Alignment IV

What about Diverging Preferences? [66]

- One some subjective topics, annotators might disagree
- It is important for a general model to reflect subjectivity and diversity in its answers
- It is hard for LLM to reflects opinions of minorities

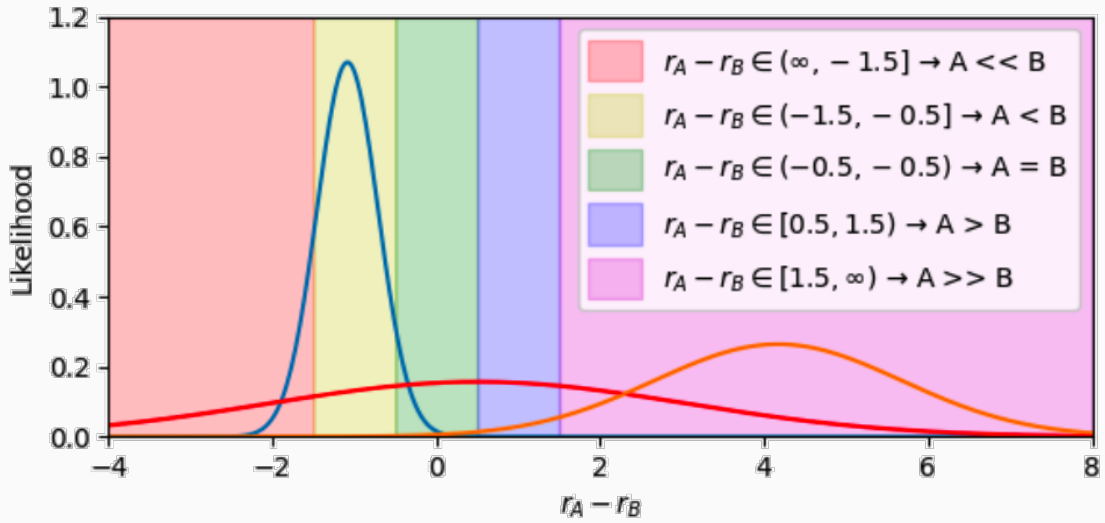


Figure 6: Reward models have different modes when looking at the pdf (of the KL)

Outline : Reasonings

Chain of Thought [60]

- Chain of Thought (CoT) prompts help models break complex tasks into steps, simply showing multi-step reasoning answers with ICL
- Useful for multi-step reasoning tasks.
- CoT improves performance on arithmetic, logic, and reasoning tasks.

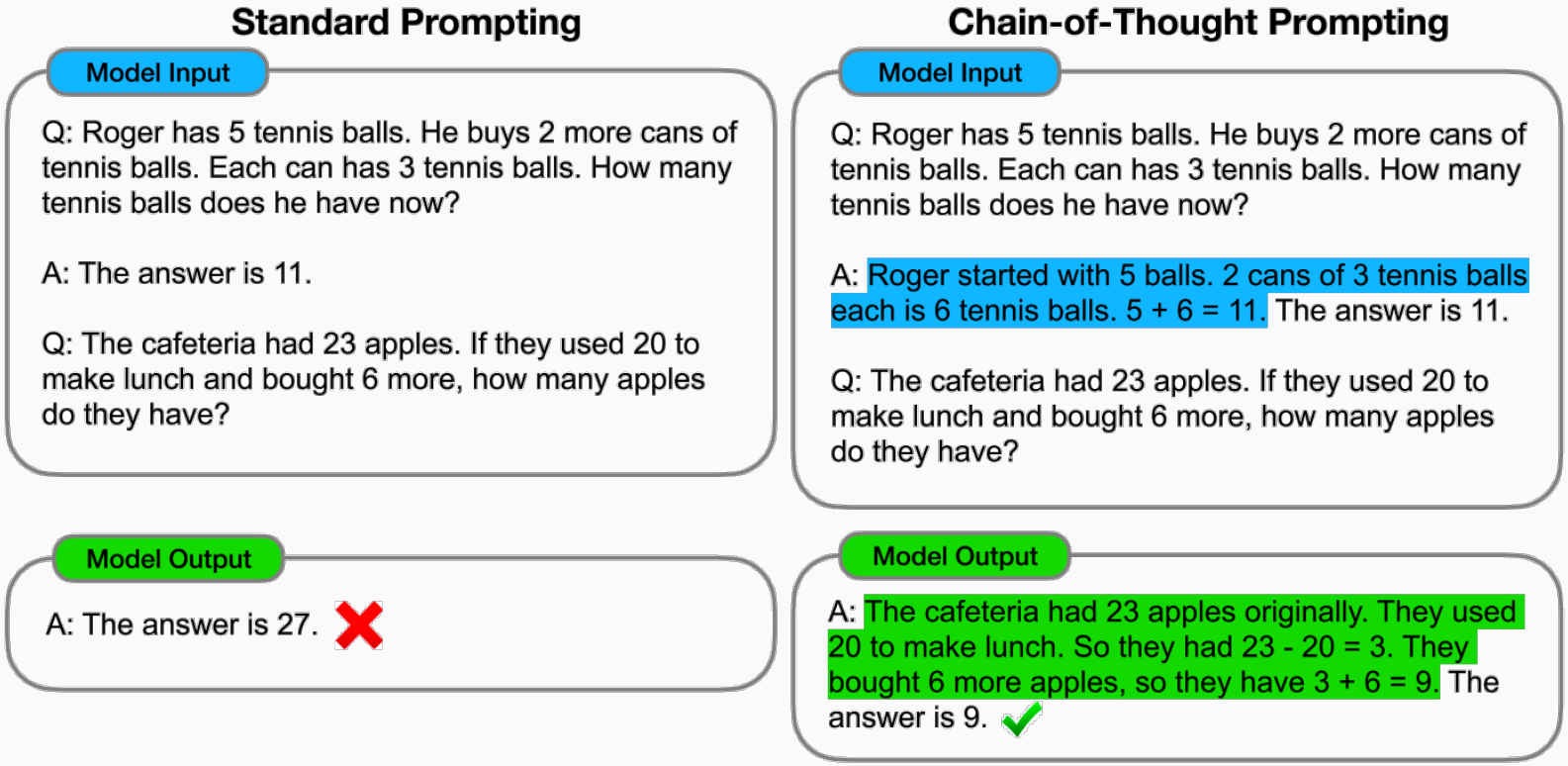


Figure 7: Augmenting every examples with a more detailed reasoning allows for better Few-Shot learning

Zero-Shot Chain of Thought [21]

- Model generates reasoning chains **without specific examples**
- It has shown remarkable performance on reasoning tasks without task-specific training.
- Enables effective generalization in complex scenarios.

(a) Few-shot

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

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: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: ***Let's think step by step.***

(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓

Zero-Shot Chain of Thought [21]

- Model generates reasoning chains **without specific examples**
- It has shown remarkable performance on reasoning tasks without task-specific training.
- Enables effective generalization in complex scenarios.

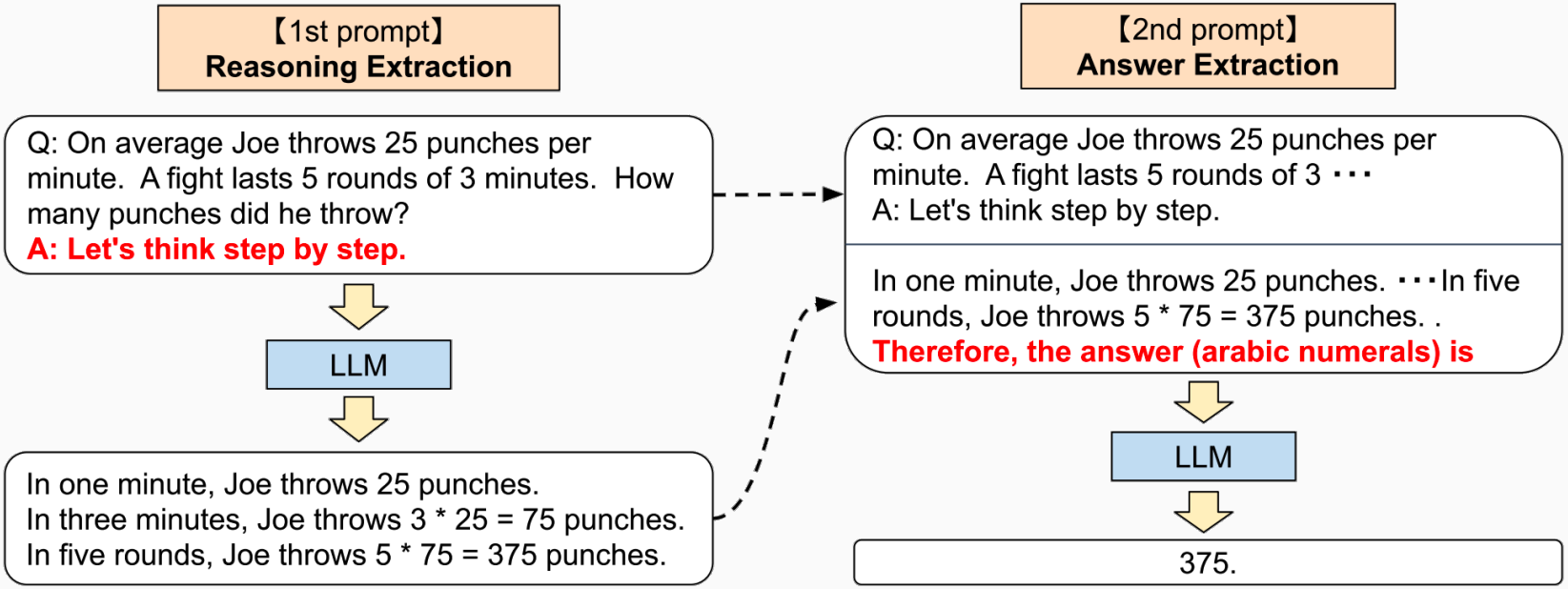


Figure 8: The LLM does not generate the whole answer in a row

Techniques exist to find the best prompt [62]: "*Take a deep breath a work on this problem step-by-step*"

CoT decoding [55]

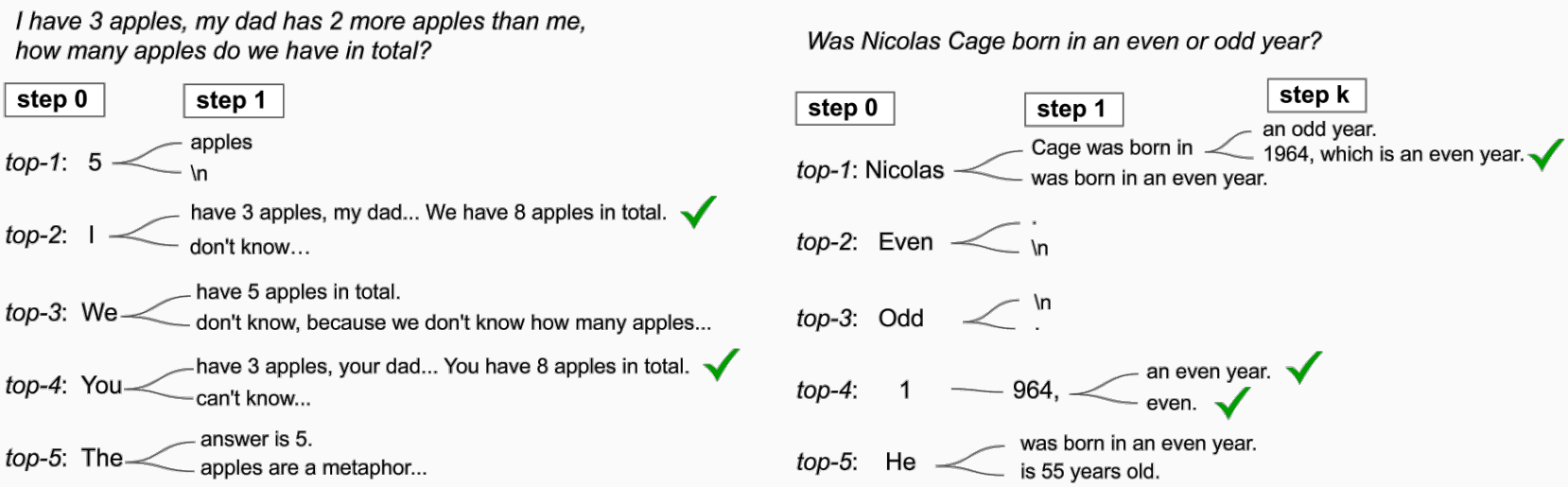


Figure 9: Decoding method to get generated text with best explanations

- Rather than conventional greedy decoding, it investigates the top-k alternative tokens, uncovering that CoT paths are frequently inherent in these sequences
- Presence of a CoT in the decoding path correlates with a higher confidence in the model's decoded answer.
- LLMs' intrinsic reasoning abilities

Other Reasonings Methods

- Self-reflection prompting: The model reflects on its initial response and improves it [48]
- Tree-of-Thoughts: Extends Chain of Thought by allowing branching reasoning steps [64]
- Self-Ask: Encourages the model to ask itself clarifying questions to break down problems [38]
- Democratic Deliberation: Models debate each other's outputs, leading to consensus or corrected reasoning [3]
- Agent-based frameworks: Multiple agents correct or refine each other's reasoning, improving final output quality.
- Thoughts Preference Optimization [61]

Outline : Training in Practice

Training Datasets: More and more Tokens

- Training data is the fuel of Language Modeling
- Texts scrapped from the web or subtitles, but also quality content from textbook, scientific papers, legal texts, books, Wiki, code, ...
- Private companies do not open their training sets as it highly impacts the model's performances

Component	Raw Size	Weight	Epochs	Effective Size	Mean Document Size
Pile-CC	227.12 GiB	18.11%	1.0	227.12 GiB	4.33 KiB
PubMed Central	90.27 GiB	14.40%	2.0	180.55 GiB	30.55 KiB
Books3†	100.96 GiB	12.07%	1.5	151.44 GiB	538.36 KiB
OpenWebText2	62.77 GiB	10.01%	2.0	125.54 GiB	3.85 KiB
ArXiv	56.21 GiB	8.96%	2.0	112.42 GiB	46.61 KiB
Github	95.16 GiB	7.59%	1.0	95.16 GiB	5.25 KiB
FreeLaw	51.15 GiB	6.12%	1.5	76.73 GiB	15.06 KiB
Stack Exchange	32.20 GiB	5.13%	2.0	64.39 GiB	2.16 KiB
USPTO Backgrounds	22.90 GiB	3.65%	2.0	45.81 GiB	4.08 KiB
PubMed Abstracts	19.26 GiB	3.07%	2.0	38.53 GiB	1.30 KiB
Gutenberg (PG-19)†	10.88 GiB	2.17%	2.5	27.19 GiB	398.73 KiB
OpenSubtitles†	12.98 GiB	1.55%	1.5	19.47 GiB	30.48 KiB
Wikipedia (en)†	6.38 GiB	1.53%	3.0	19.13 GiB	1.11 KiB
DM Mathematics†	7.75 GiB	1.24%	2.0	15.49 GiB	8.00 KiB
Ubuntu IRC	5.52 GiB	0.88%	2.0	11.03 GiB	545.48 KiB
BookCorpus2	6.30 GiB	0.75%	1.5	9.45 GiB	369.87 KiB
EuroParl†	4.59 GiB	0.73%	2.0	9.17 GiB	68.87 KiB
HackerNews	3.90 GiB	0.62%	2.0	7.80 GiB	4.92 KiB
YoutubeSubtitles	3.73 GiB	0.60%	2.0	7.47 GiB	22.55 KiB
PhilPapers	2.38 GiB	0.38%	2.0	4.76 GiB	73.37 KiB
NIH ExPorter	1.89 GiB	0.30%	2.0	3.79 GiB	2.11 KiB
Enron Emails†	0.88 GiB	0.14%	2.0	1.76 GiB	1.78 KiB
The Pile	**825.18 GiB**			**1254.20 GiB**	**5.91 KiB**

Figure 10: Composition of The Pile [13]

Training Datasets: Examples of recent open-source datasets

- **C4**: Colossal Clean Crawled Corpus [11]: 0.3 TB
- **The Pile** [13]: 0.8 TB
- **Dolma**: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research [50]: 4.5 TB
- **The FineWeb Datasets** [35]: 93.4 TB
- **RedPajama**: Open Dataset for Training LLMs [57]: 270 TB

Table 1: Comparison of open pretraining Datasets along the dimensions of transparency, versatility, and scale.

Dataset	Transparency		Versatility			Scale (TB)
	Open Access	Open Code	Raw Data	Composite	Multilingual	
Refined Web [44]	✔(subset)	✗	✗	✗	✗	2.8
FineWeb [43]	✔	✔	✗	✗	✗	93.4
FineWeb-EDU [43]	✔	✔	✗	✗	✗	8.8
C4 [46]	✔	✔	✗	✗	✗	0.3
mC4 [63]	✔	✔	✗	✗	✔	9.7
DCLM baseline [30]	✔	✔	✗	✗	✗	10.0
DCLM-Pool [30]	✔	✔	✔	✗	✔	340.0
Dolma v1.7 [52]	✔	✔	✗	✔	✗	4.5
Pile [17]	✔	✔	✗	✔	✗	0.8
SlimPajama [51]	✔	✔	✗	✔	✗	0.9
ROOTS [26, 27]	✔	✔	✗	✔	✔	1.6
RedPajama-V1	✔	✔	✗	✔	✗	3.0
RedPajama-V2	✔	✔	✔	✗	✔	270.0

Training Datasets: Examples of recent open-source datasets

- **C4**: Colossal Clean Crawled Corpus [11]: 0.3 TB
- **The Pile** [13]: 0.8 TB
- **Dolma**: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research [50]: 4.5 TB
- **The FineWeb Datasets** [35]: 93.4 TB
- **RedPajama**: Open Dataset for Training LLMs [57]: 270 TB

Table 1: Comparison of open pretraining Datasets along the dimensions of transparency, versatility, and scale.

Dataset	Transparency		Versatility			Scale (TB)
	Open Access	Open Code	Raw Data	Composite	Multilingual	
Refined Web [44]	✔(subset)	✗	✗	✗	✗	2.8
FineWeb [43]	✔	✔	✗	✗	✗	93.4
FineWeb-EDU [43]	✔	✔	✗	✗	✗	8.8
C4 [46]	✔	✔	✗	✗	✗	0.3
mC4 [63]	✔	✔	✗	✗	✔	9.7
DCLM baseline [30]	✔	✔	✗	✗	✗	10.0
DCLM-Pool [30]	✔	✔	✔	✗	✔	340.0
Dolma v1.7 [52]	✔	✔	✗	✔	✗	4.5
Pile [17]	✔	✔	✗	✔	✗	0.8
SlimPajama [51]	✔	✔	✗	✔	✗	0.9
ROOTS [26, 27]	✔	✔	✗	✔	✔	1.6
RedPajama-V1	✔	✔	✗	✔	✗	3.0
RedPajama-V2	✔	✔	✔	✗	✔	270.0

Training Datasets: Cleaning Before Using I

Data scrapped from the web is pretty noisy and you need to clean it:

Character noise: `"And it </b> was"`

Markups, syntax breaks, etc... all that gives non NL text is harmful, and can prevent convergence!

Diversification: `"- Yes lol bro!  - Yes bro!"`

Diversification improves the pre-training as it generally removes useless (simple) examples [53].[1] Perplexity can help filtering simple documents.

Deduplication: same content in different pages

It has been identified as playing a significant role in improving language models [23]. Repeated data has been shown to be **increasingly harmful to model quality as parameter count increases** [16]:

- for a 1B parameters model, a hundred duplicates are harmful;
- at 175B, even a few duplicates could have a disproportionate effect.

[1] Some also simply use diverse good quality data from Textbooks [14]

Training Datasets: Cleaning Before Using II

Concretely?

- API to extract text efficiently from XML (`trafilatura`; [4]), remove tables,...
- Look at markup that can break the extraction
- Remove uninformative lines with less than a few tokens
- Deduplication, Fuzzy using minHash, exact using n-grams using `text-dedup` library [20].
- Using simple heuristics help a lot [46]
- Concrete Examples: what they done for RedPajama

DOCUMENT PREPARATION			FILTERING		DEDUPLICATION	
URL filtering	**Text extraction**	**Language identification**	**Document-wise filtering**	**Line-wise filtering**	**Deduplication**	**URL deduplication**
Aggregated blocklist, URL scoring, common HQ sources blocked Appendix G.1	From WARC using `warcio`, `trafilatura` for extraction Barbaresi (2021)	`fastText` classifier from CCNet, thresholding on top language score Wenzek et al. (2020)	In-document repetition removal and quality heuristics from MassiveWeb Rae et al. (2021)	Remove undesirable lines (call to actions, navigation buttons, social counters, etc.) Appendix G.2	Fuzzy deduplication w/ MinHash + exact substring deduplication w/ suffix arrays Lee et al. (2022)	Remove URLs revisited across CommonCrawl dumps Section 3.3

Figure 11: Example of Corpus cleaning [36, 35]

Recent Instruction Fine-Tuning Datasets: FLAN [58, 29, 9]

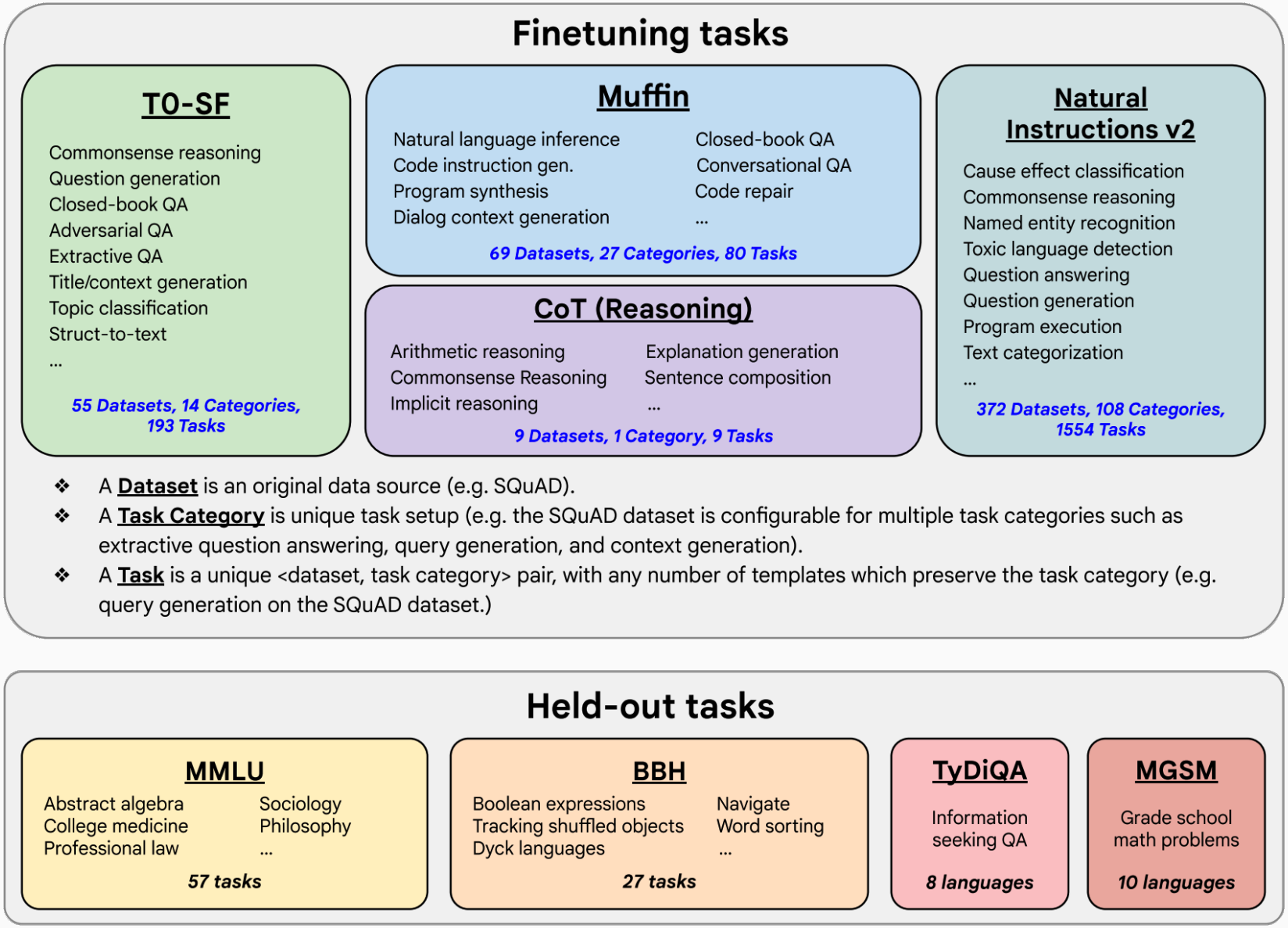


Figure 12: Finetuned Language Models on ANnotations (FLAN)

Recent Instruction Fine-Tuning Datasets

Some examples of instruction datasets:

- **Alpaca**: 52k instruction-response pairs fully made by a machine. Simple tasks, such as text generation, summarization, and question-answering, ... [52]
- **DollyV2**: Open-source dataset curated manually, emphasizing diverse tasks for 15k pairs [10].
- **Open Instruction Generalist (OIG)**: 43M instructions from 30 datasets, 75% academic sources (e.g., P3, FLAN) and 25% diverse tasks like coding, poetry, and summarization [22]
- **Aya Dataset**: First open-source dataset of instructions across diverse languages and not just English (3.5% of it) [49]

Outline : Evaluating LLMs

How to assess them

Good for many things: Multi-task benchmarks

- Started with a few 10 tasks on BERT
- Now up to more than 200 various tasks
- Diverse topics, can even be college-level questions, ..

Good in conversations: Multi-turn benchmarks

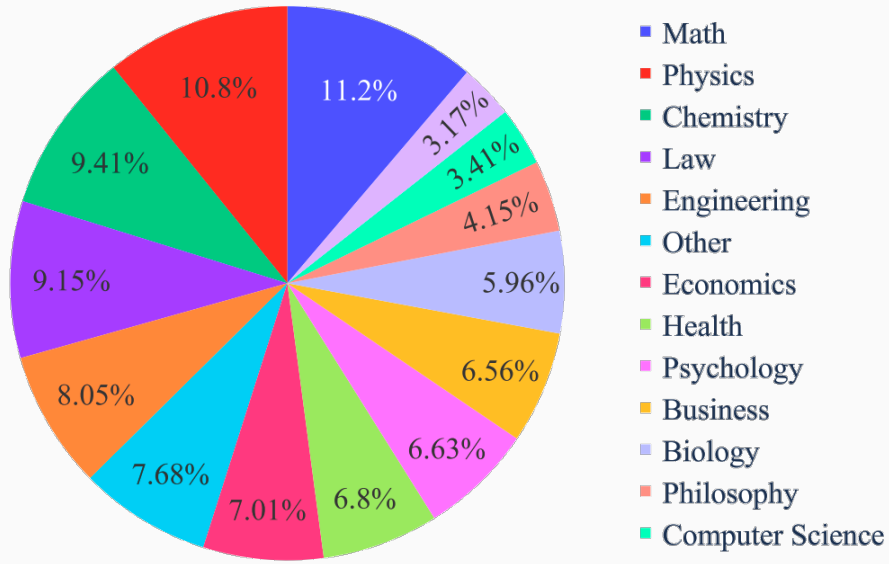
- Evaluates the models in a conversation setting
- Useful for chatbots or assistants

But what is good and not good?: Human- and LLM-as-judge

- Using the full dialogue to judge.
- Automatic way: using bigger LLMs to note other ones
- LLM correlates pretty well with human, however it brings some biases

Multi-task Benchmarks

- Multi-task benchmarks evaluate LLMs across a wide range of tasks from different domains, some of thems are:
  - **MMLU** (Massive Multitask Language Understanding) [15] and **MMLU-Pro** [56]: 57+ diverse tasks covering history, science, law, ...
  - **BIG-Bench** [51]: Over 200 tasks designed to measure diverse capabilities including reasoning, ethics, and humor.
  - **TruthfulQA** [28]: Focuses on evaluating the truthfulness of model outputs in various domains (if it says BS or not)
  - **SciBench** [54]: College-Level Scientific Problems
- The goal is to test the model's generalization across unrelated domains.
- Highlights the strengths and weaknesses of models in specific areas (e.g., reasoning vs. factual knowledge).



(a) Distribution of Disciplines in MMLU-Pro

Multi-turn Benchmarks: LLMs serve as conversational agents

- Multi-turn benchmarks evaluate a model's ability to handle conversation and context retention across multiple exchanges:
  - **MT-Bench** [67]: A two-turn benchmark evaluating conversational quality, coherence, and context retention
  - **MT-Bench-101** [1]: Multi-turn benchmark, specifically designed to test for complex and extended dialogues
  - Classical ConvQA benchmarks [17, 8, 41, 32, 18]: Test the model's ability to answer questions while maintaining context across turns.

- A first simple way to integrate interactions in the evaluation of the chatbot
- Maintaining context and delivering coherent answers in a conversation is crucial for conversational assistants

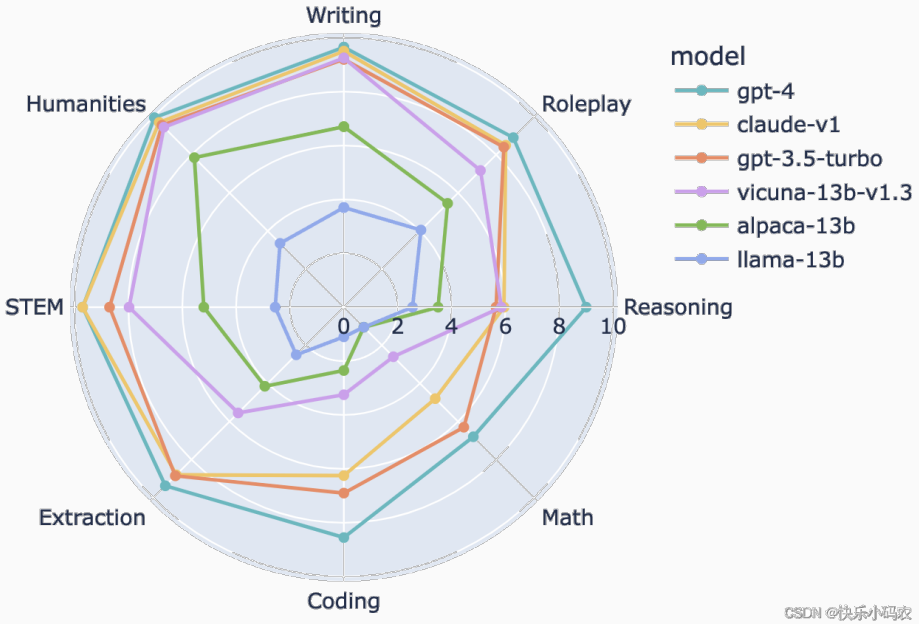
Figure 13: MT-Bench-101 taxonomy

Interaction evaluation: Human- or LLM-as-a-judge

- **Why different?** In a dialogue, there are sometimes many possible correct answers, and the above benchmarks fail to evaluate this.
- **Solution**: Human (expensive) or LLMs are sometimes used as evaluators of their own or other models' outputs.

Human judge

Chatbot Arena [67, 7]: Crowd-sourcing platform where users compare LLMs by voting on their performance in live, head-to-head chat interactions.



Interaction evaluation: Human- or LLM-as-a-judge

- **Why different?** In a dialogue, there are sometimes many possible correct answers, and the above benchmarks fail to evaluate this.
- **Solution**: Human (expensive) or LLMs are sometimes used as evaluators of their own or other models' outputs.

LLM judge [67]

- SoTA LLM judges can match both controlled and crowd-sourced human preferences well (reaching 80 % agreement, same as humans)
- Useful pattern when human evaluation is costly or infeasible, but there are challenges with bias and objectivity in LLM judgments.
- Can be used even without conversation context!

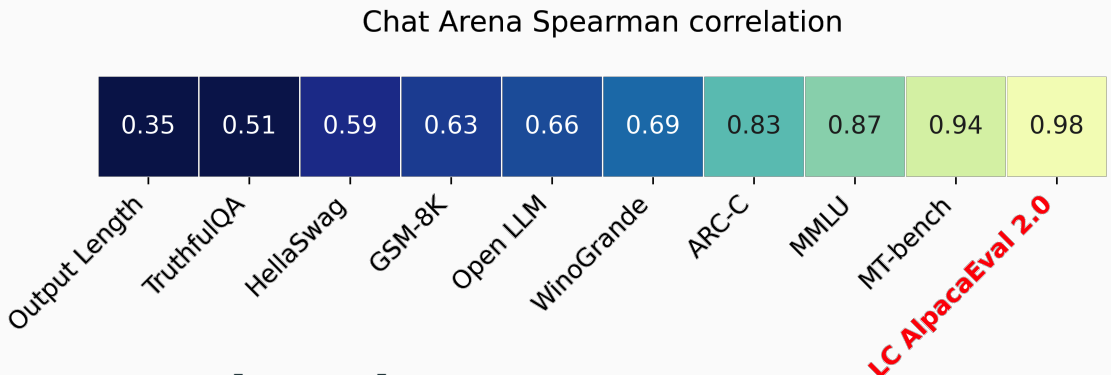


Figure 14: AlpacaEval [12, 26] correlates highly with human on ChatbotArena

Other studies on LLM, less focused on performances

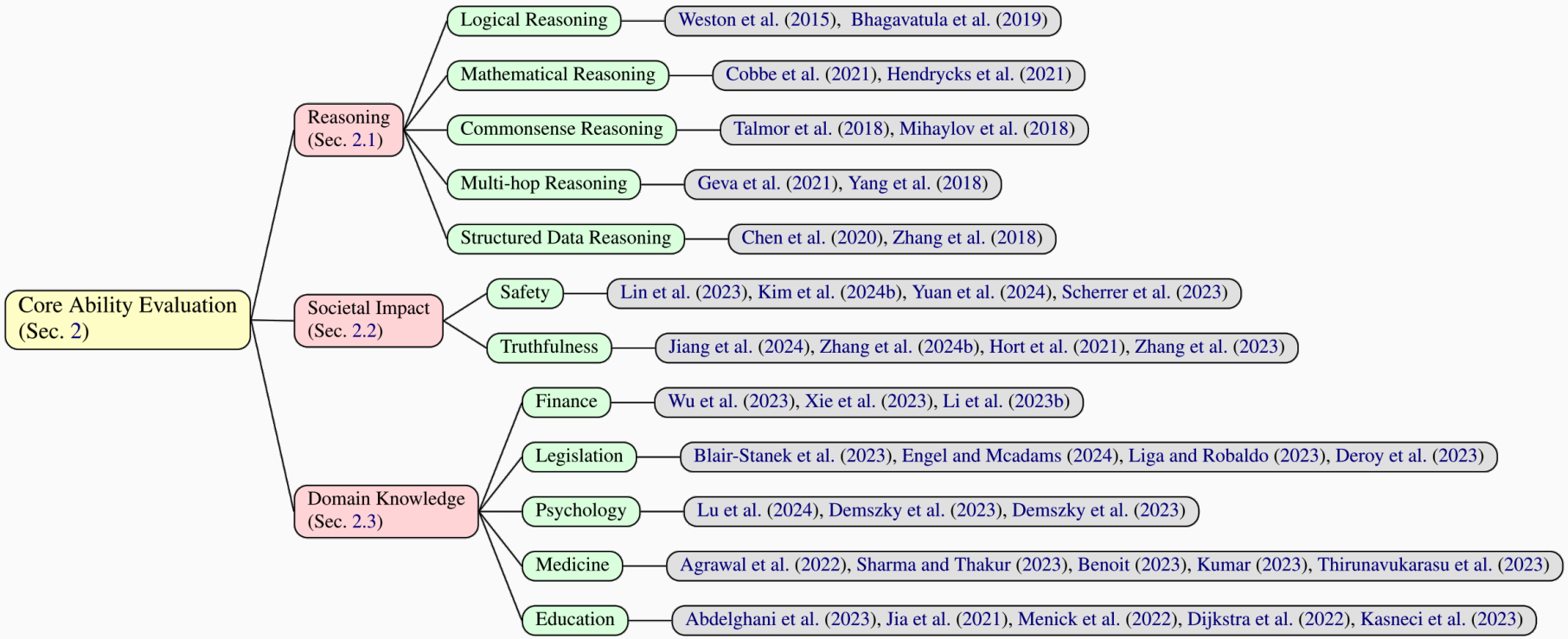


Figure 15: An overview of core ability evaluation from [37]

LLMs can also be fine-tuned to judge

Metric	MT	TS	DG	IC	D2T	SG	GE	REF	LLMs	Protocol	Aspects
Prompt-based Evaluation											
BARTScore (Yuan et al., 2021)	✓	✓	*	*	✓	*	*	✓	BART	Prob	CON/COH/REL/FLU/ INF/COV/ADE
GPTScore (Fu et al., 2023)	✓	✓	✓		✓	*	*		GPT3	Prob	CON/COH/REL/FLU/COV/ACC MQM/INF/FAC/INT/ENG/NAT
G-EVAL (Liu et al., 2023c)	*	✓	✓		*	*	*		ChatGPT/GPT-4	Advanced	CON/COH/REL/FLU /NAT/ENG/GRO
ICE (Jain et al., 2023)	*	✓	*		*	*	*		GPT-3	Score	CON/COH/REL/FLU
GEMBA (Kocmi and Federmann, 2023)	✓	*	*		*	*	*		ChatGPT	Score/Likert	NONE
LLM_eval (Chiang and Lee, 2023)	*	*	*		*	✓	*		ChatGPT	Likert	GRAM/COH/REL/LIK
FairEval (Wang et al., 2023c)	*	*	*		*	*	✓		ChatGPT/GPT-4	Pairwise	NONE
AuPEL (Wang et al., 2023e)	*	*	*		*	*	✓		PaLM-2	Pairwise	PER/QUA/REL
DRPE (Wu et al., 2023a)	*	✓	*	*	*	*	*	✓	GPT-3	Ensemble	CON/COH/REL/FLU/INT/USE
ChatEval (Chan et al., 2023)	*	*	✓		*	*	✓		ChatGPT/GPT-4	Ensemble	NAT/COH/ENG/GRO
WideDeep (Zhang et al., 2023)	*	*	*		*	*	✓		ChatGPT	Ensemble	COH/REL/HARM/ACC
PRD (Li et al., 2023c)	*	*	*		*	*	✓		GPT-4/GPT-3.5 Vicuna/Claude/Bard	Ensemble	INF/COH
FACTSCORE (Min et al., 2023)		*					✓		ChatGPT	Advanced	FAC
EAprompt (Lu et al., 2023)	✓	*	*		*	*	*		ChatGPT/text-davinci-003	Advanced	NONE
AUTOCALIBRATE (Liu et al., 2023f)	*	✓	*		*	*	*		GPT-4	Likert	CON/COH/REL/FLU/INF/NAT
ALLURE (Hasanbeig et al., 2023)	*	✓	*		*	*	✓		GPT-4	Advanced	CON/COH/FLU/REL
Tuning-based Evaluation											
PRISM (Thompson and Post, 2020)	✓	*	*	*	*	*	*	✓	Transformer	Prob	NONE
T5Score (Qin et al., 2022)	✓	✓	*	*	*	*	*	✓	T5	Prob	NONE
TrueTeacher (Gekhman et al., 2023)	*	✓	*		*	*	*		T5	Likert	CON
X-EVAL (Liu et al., 2023a)	*	✓	✓		✓	*	*		FLAN-T5-large	Likert	DEP/LIK/UND/FLE/INF/INQ INT/SPE/COR/SEM/COH/ENG NAT/GRO/CON/REL/FLU
AUTO-J (Li et al., 2023a)	*	*	*		*	*	*		LLaMA	Likert/Pairwise	ACC/CLA/FEA/CRE/THO STR/LAY/COM/INF
PERSE (Wang et al., 2023a)	*	*	*	*	*	✓	*	✓	LLaMA	Likert/Pairwise	INT/ADA/SUR/CHA/END
PandaLM (Wang et al., 2023f)	*	*	*		*	*	✓		LLaMA	Pairwise	CLA/COM/FOR/ADH
Attscore (Yue et al., 2023)	*	*	*		*	*	✓		Roberta/T5/GPT2 LLaMA/Vicuna	Advanced	CON
TIGERScore (Jiang et al., 2023)	✓	✓	*		✓	✓	✓		LLaMA	Advanced	COH/INF/ACC/COM
INSTRUCTSCORE (Xu et al., 2023)	✓	*	*	*	*	*	*	✓	LLaMA	Advanced	NONE
Prometheus (Kim et al., 2023a)	*	*	*		*	*	✓		LLaMA-2	Likert/Pairwise	NONE
Prometheus-2 (Kim et al., 2023a)	*	*	*		*	*	✓		Mistral 7B	Likert/Pairwise	NONE
CritiqueLLM (Ke et al., 2023)	*	*	*		*	*	✓		ChatGLM	Likert	NONE

Figure 16: Natural Language Generation metrics from LLMs [27]

A big family of LLMs

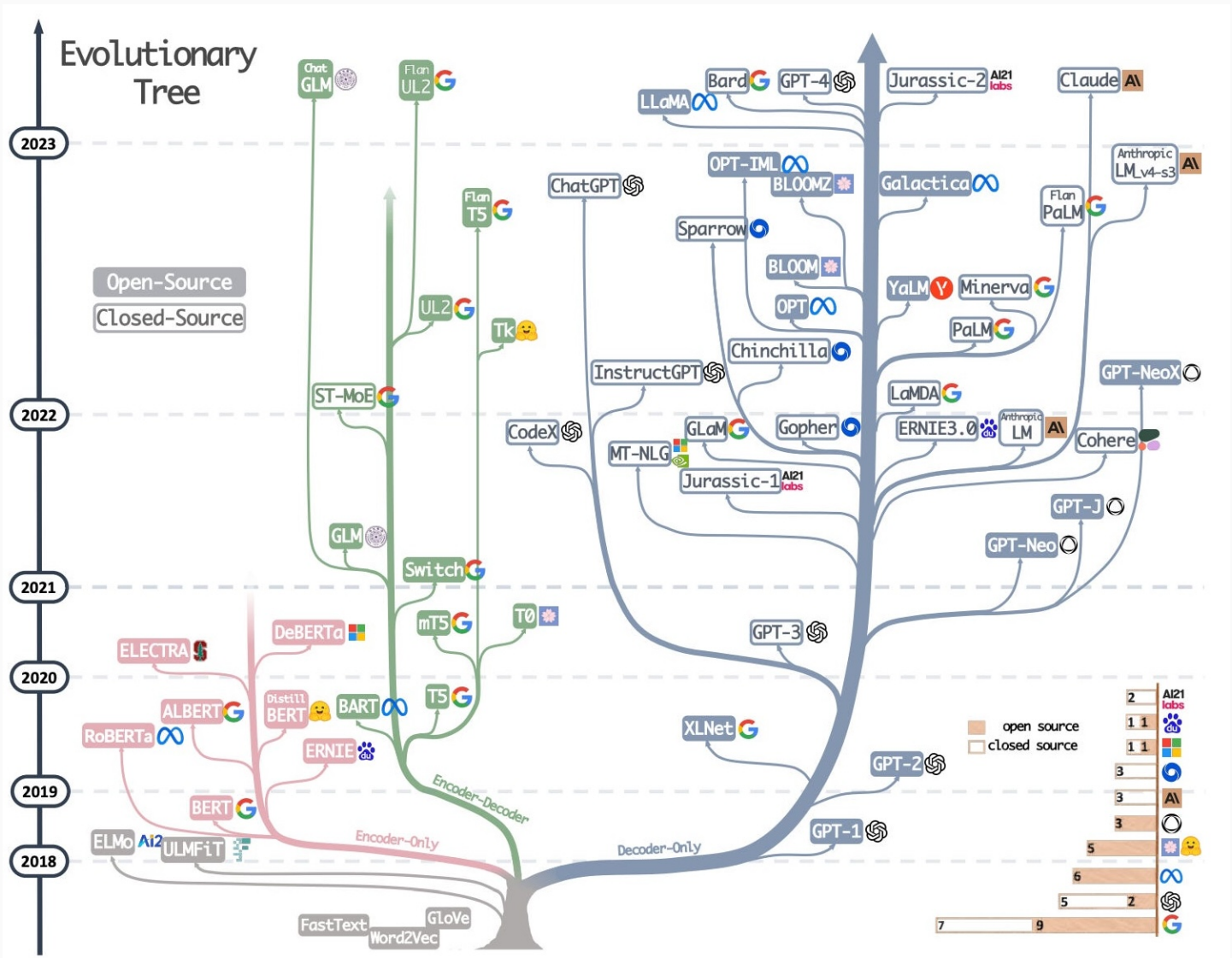


Figure 17: Some of the most well-known models, the ones on the same branch have closer relationships. More info [here]

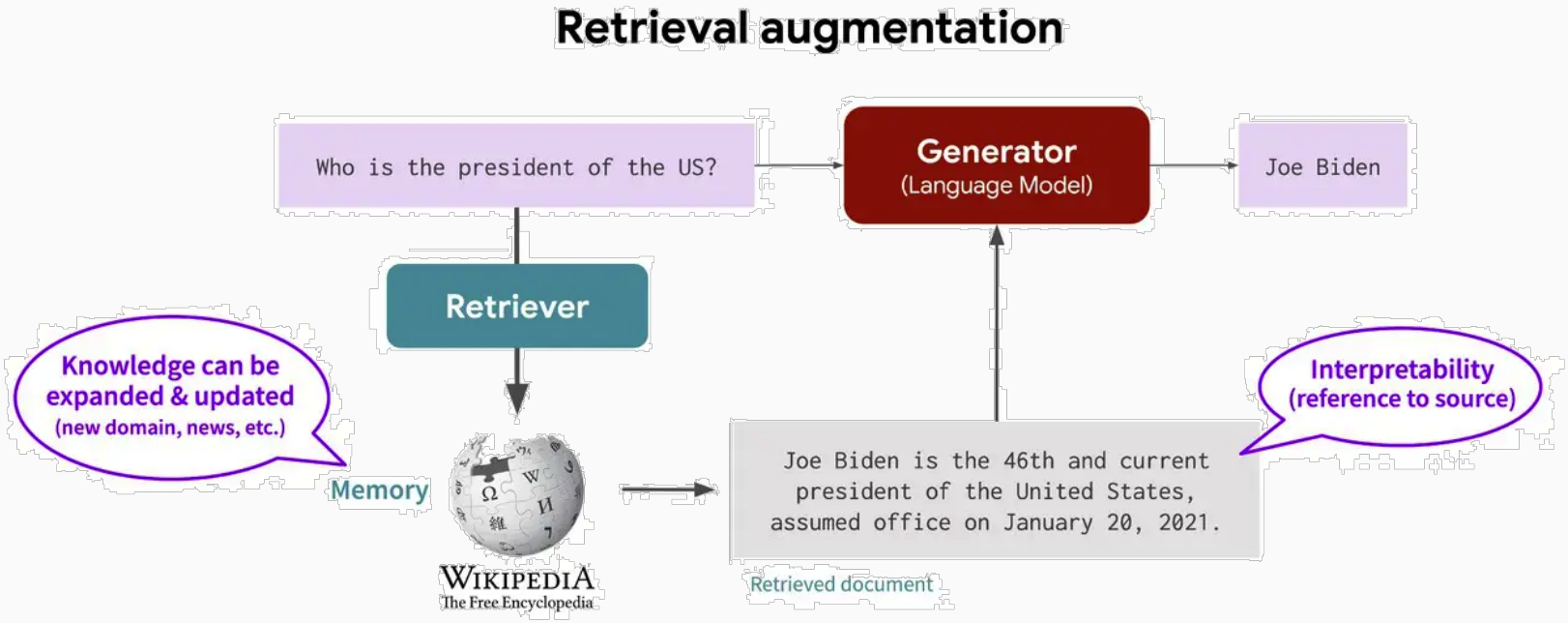
Outline : LLMs as Agents

Retrieval Augmented Generation (RAG; [24])

RAG integrates **retrieval** from external knowledge bases with **text generation**

Principle

- Retrieved documents provide context for the model, reducing reliance on static, parametric memory.
- Instead of relying solely on pre-trained model parameters, dynamically retrieves relevant information from external databases



Retrieval Augmented Generation (RAG; [24])

RAG integrates **retrieval** from external knowledge bases with **text generation**: useful for agents seeking for specific knowledge

Principle

- Retrieved documents provide context for the model, reducing reliance on static, parametric memory.
- Instead of relying solely on pre-trained model parameters, dynamically retrieves relevant information from external databases

- **Retriever**: Identifies top-k relevant documents from a corpus (e.g., Dense Passage Retrieval, BM25).
  - Can transform queries and document into vectors,
  - Match them afterward using a mapping or similarity (fast as they might be A LOT of documents)
- **Generator**: Processes retrieved text and query together in a unique prompt to produce coherent output (e.g., BART, T5).

Language Agents

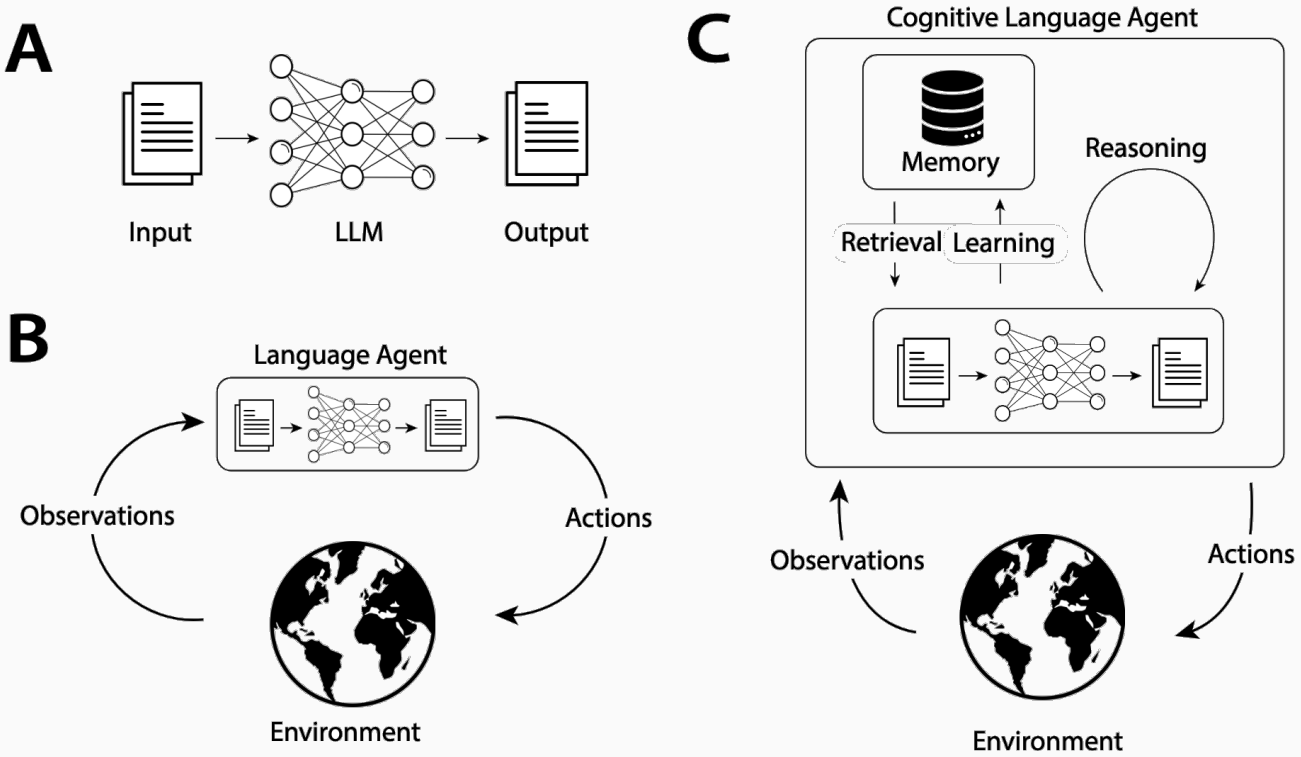
An agent is an entity that perceives its environment through sensors and acts upon it using actuators to achieve specific goals.

They are designed to make decisions, perform actions autonomously, and adapt their behavior based on observations or learned experiences using:

- **Memory**: Stores context and interaction history for long-term tasks (can be document database)
- **Symbolism**: Represents knowledge and concepts as structured, interpretable symbols or rules
- **Reasoning**: For processing inputs and generating actionable outputs.
- **Planning**: Consist of a set of rules, each specifying a precondition and an action, in order to reach a global goal
- **Environment Interaction**: APIs or tools enabling interaction with external systems.

There is a full class on LLM Agents by Dawn Song from Berkeley. Or a very complete tutorial.

Language Agents: Different uses of LLMs [63]



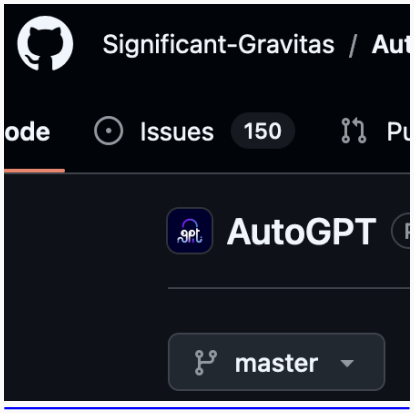
A **NLP**: LLM takes text as input and outputs

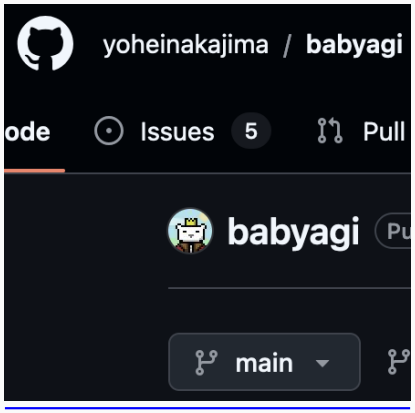
B **Language agents**: LLM in a direct feedback loop with the external environment by transforming observations into text and using the LLM to choose actions

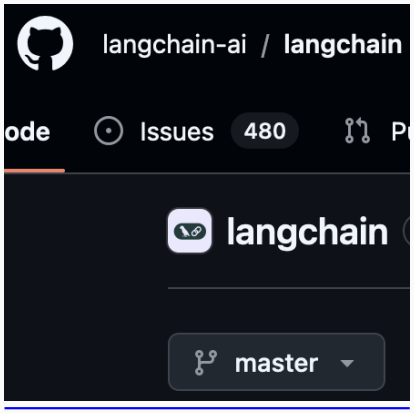
C **Cognitive language agents**: LLM to manage the agent's internal state via processes such as learning and reasoning

Examples of LLM Cognitive Language Agents frameworks

- **AutoGPT**: Autonomously breaks down tasks and executes subtasks with minimal human input.
- **BabyAGI**: Combines LLMs with memory and a task management loop.
- **LangChain**: Enables chaining LLM prompts with external tools, APIs, and data sources.





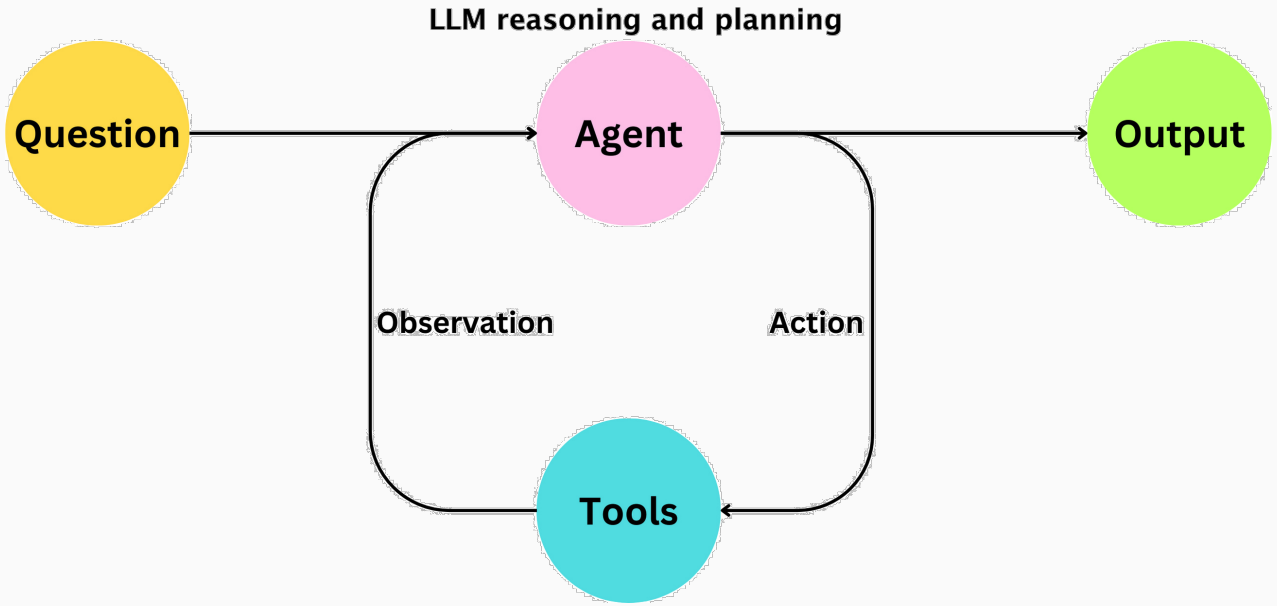


LLMs Using Tools: Toolformer [44]

LLMs struggle with basic functionality, such as arithmetic or factual lookup. **Tool usage** enable them to interact with APIs, databases, code,... for enhanced reasoning and task completion requiring actions.

Principle

- Provides access to specialized functionality, such as retrieving live information, performing calculations, or query services
- LLMs interact with tools dynamically, allowing the Agent to perform actions impacting the exterior world



LLMs Using Tools: Toolformer [44]

LLMs struggle with basic functionality, such as arithmetic or factual lookup. **Tool usage** enable them to interact with APIs, databases, code,... for enhanced reasoning and task completion requiring actions.

Principle

- Provides access to specialized functionality, such as retrieving live information, performing calculations, or query services
- LLMs interact with tools dynamically, allowing the Agent to perform actions impacting the exterior world

- **Key Mechanisms**:
  - **LLMs learn to use the tools using a special template** during training, querying external systems (e.g., search engines, calculators) via natural language.
  - **Code Generation**: Models write and execute scripts for tasks like database queries, simulations or numerical calculation.
- **Evaluation**: GAIA Benchmark [31] evaluates reasoning, multi-modality handling, web browsing, and generally tool-use proficiency

LLMs Using Tools

- LLMs can scale up to learn using many tools in a zero-shot way with instruction-tuning datasets for tool use [70, 30, 39]
- The tools can be domain-specific, like tools specialized for medical domain [25]
- LLMs can also create their own tools: one LLM acting as *tool maker* and another one as *tool user* [6]

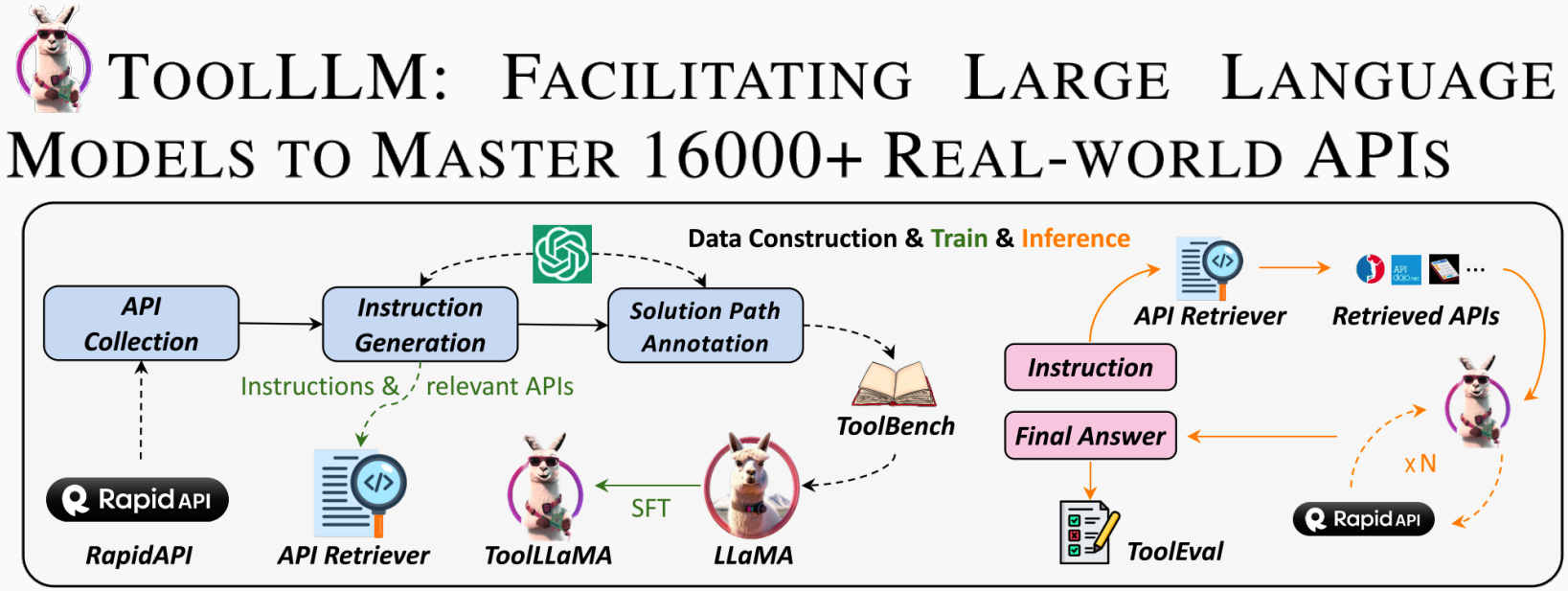


Figure 18: ToolLLM is general tool-use framework: data construction, model training, and evaluation [39]

LLMs using tools: HuggingGPT [47]

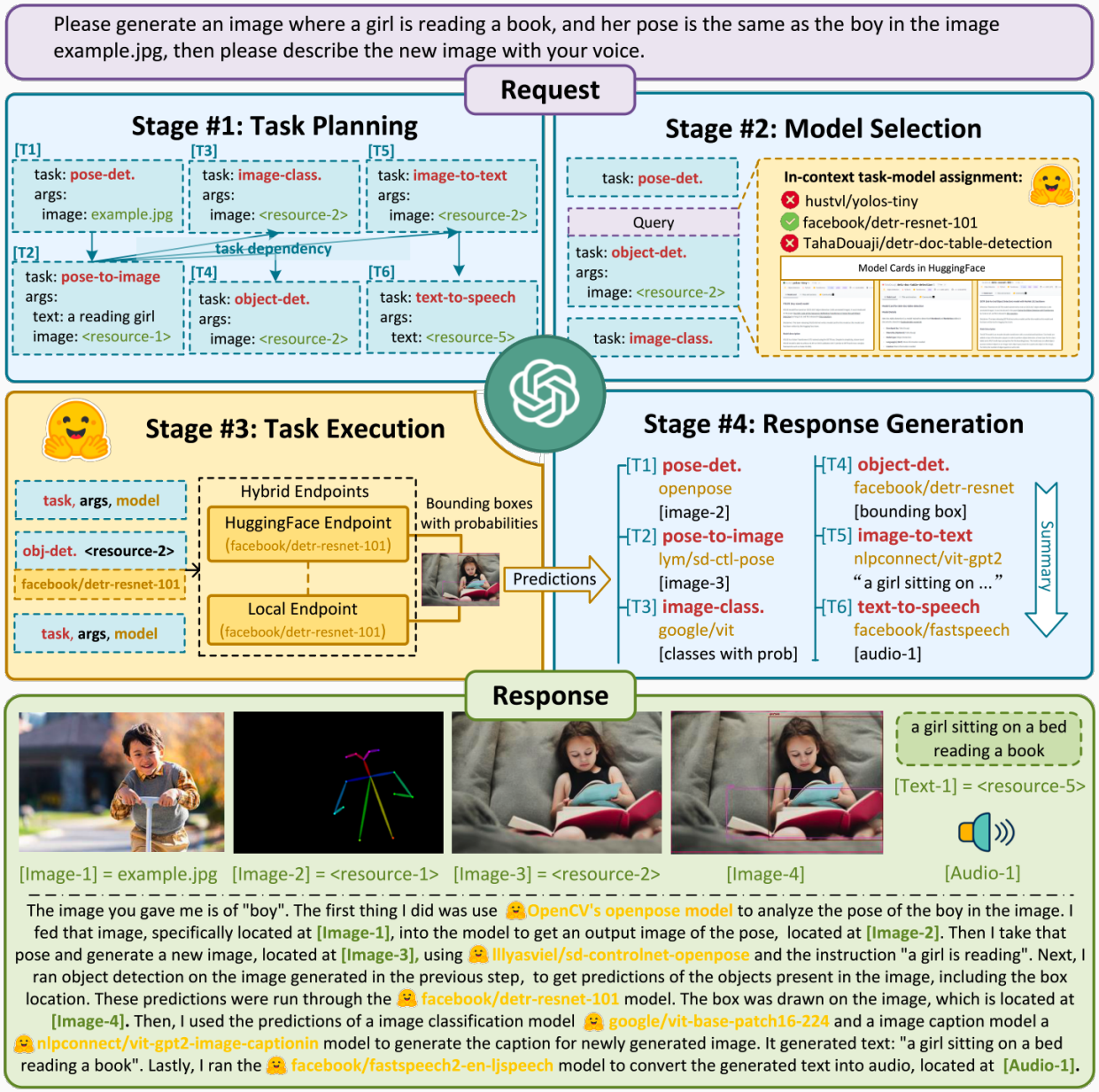
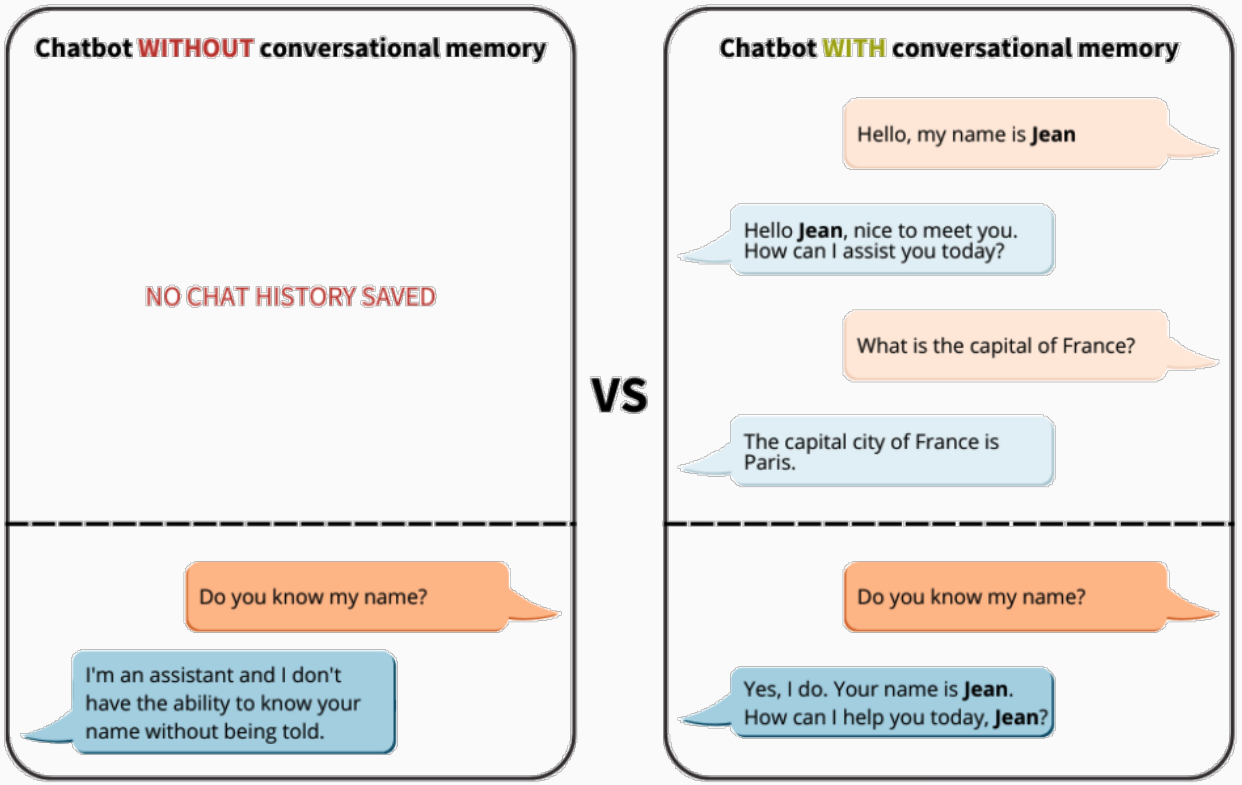


Figure 19: Tools as complex as models on HuggingFace hub

Context window for chatting: How to model a conversation history?

How to model a conversation history?

The model transform past conversation into text and put it in the prompt using a special format, specific to the LLM.



Context window for chatting: How to model a conversation history?

How to model a conversation history?

The model transform past conversation into text and put it in the prompt using a special format, specific to the LLM.

```
messages = [
    {"role": "user", "content": "Hi there!"},
    {"role": "assistant", "content": "Nice to meet you!"},
    {"role": "user", "content": "Can I ask a question?"}
]
```

You can use chat templates with the `tokenizer` class using `transformers`.

```
tokenizer.apply_chat_template(messages, tokenize=False, add_generation_prompt=True)
"""<|im_start|>user
Hi there!<|im_end|>
<|im_start|>assistant
Nice to meet you!<|im_end|>
<|im_start|>user
Can I ask a question?<|im_end|>
<|im_start|>assistant
"""
```

More info in the *Chatting with Transformers* tutorial.

Templates with RAG

```
# Define conversation input
conversation = [
    {"role": "user", "content": "What has Man always dreamed of?"}
]

# Define documents for retrieval-based generation
documents = [
    {
        "title": "The Moon: Our Age-Old Foe",
        "text": "Man has always dreamed of destroying the moon. In this essay, I shall..."
    },
    {
        "title": "The Sun: Our Age-Old Friend",
        "text": "Although often underappreciated, the sun provides several notable benefits..."
    }
]
```

You can call using the option `documents` (if supported by the model) and generate:

```
# Tokenize conversation and documents using a RAG template, returning
↪  PyTorch tensors.
>>> input_ids = tokenizer.apply_chat_template(
    conversation=conversation,
    documents=documents,
    chat_template="rag",
    tokenize=True,
    add_generation_prompt=True,
    return_tensors="pt").to(device)
# Generate a response
>>> gen_tokens = model.generate(
    input_ids,
    max_new_tokens=100,
    do_sample=True,
    temperature=0.3,
    )
# Decode and print the generated text along with generation prompt
>>> gen_text = tokenizer.decode(gen_tokens[0])
```

LLMs using tools

```python
def get_current_temperature(location: str, unit: str) -> float:
    """
    Get the current temperature at a location.

    Args:
        location: The location to get the temperature for, in the format "City, Country"
        unit: The unit to return the temperature in. (choices: ["celsius", "fahrenheit"])
    Returns:
        The current temperature at the specified location in the specified units, as a float.
    """
    return 22.  # A real function should probably actually get the temperature!

list_tools = [get_current_temperature]

messages = [
  {"role": "system", "content": "You are a bot that responds to weather queries. You should reply
  ↪  with the unit used in the queried location."},
  {"role": "user", "content": "Hey, what's the temperature in Paris right now?"}
]
```

You can simply call using the option `tools`:

```
>>> inputs = tokenizer.apply_chat_template(messages, tools=list_tools,
↪  add_generation_prompt=True, return_dict=True, return_tensors="pt")
>>> inputs = {k: v.to(model.device) for k, v in inputs.items()}
>>> out = model.generate(**inputs, max_new_tokens=128)
>>> print(tokenizer.decode(out[0][len(inputs["input_ids"][0]):]))
```

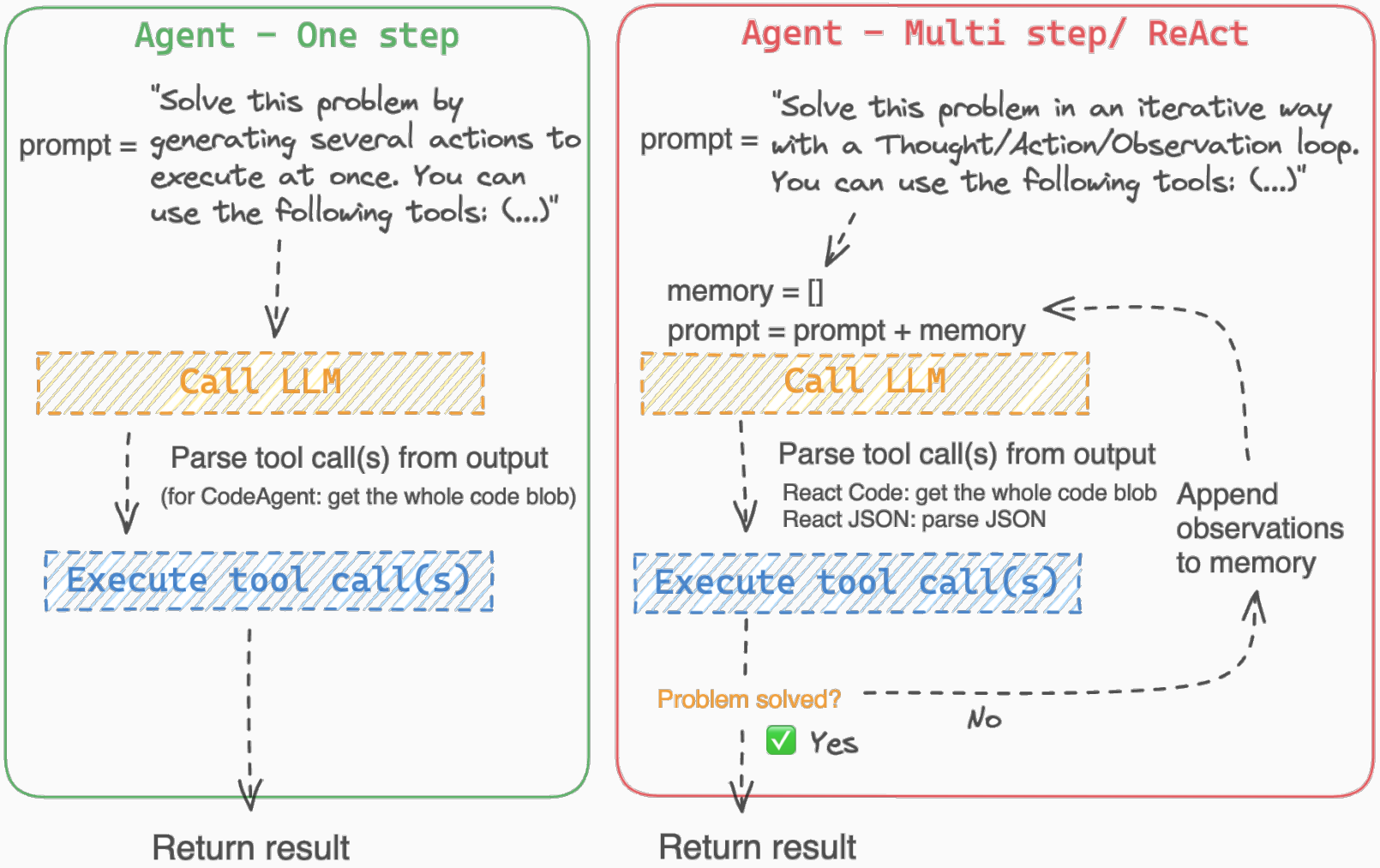
Will get you this:

```
<tool_call>
{"arguments": {"location": "Paris, France", "unit": "celsius"}, "name":
↪  "get_current_temperature"}
</tool_call><|im_end|>
```

The model knows to call the right function to solve the problem!

Language Agent in practice with ReAct and HuggingFace

ReAct is an prompt-based approach to building agents following a cycle of Thought ⇒ Action ⇒ Observation until task is solved [65]



Language Agent in practice with ReAct and HuggingFace

ReAct is an prompt-based approach to building agents following a cycle of Thought $\Rightarrow$ Action $\Rightarrow$ Observation until task is solved [65]

- The agent has access to various tools that are already coded or that you can create using `transformers.agents.Tool`
- It is possible to have access to its "*thoughts*" which are explanations on its behavior while planning
- It is possible to have access to the internal states:
  - its every actions
  - its "*thoughts*": explanations of its behavior while planning

You can look at the system prompt of the `ReActAgent` class here.
Look at this blogposts and this tutorial![2]

[2] Note that you have also the `CodeAgent` class in addition to the `ReactAgent` class

Video Example

Figure 20: A video illustrating an Agent answering to a query

Can LLMs can simulate social interactions?

Figure 21: Generative Agents: Interactive Simulacra of Human Behavior [34]

Can LLMs can simulate social interactions?

Figure 21: Generative Agents: Interactive Simulacra of Human Behavior [34]

It's not that simple! [69]

Actually LLMs struggle with getting a human-like access to their interlocutor's mental state (Objectives, beliefs, desires, knowledge, ...)

Some External Resources

- Generative LLMs:
  - The Large Language Model course from Percy Liang
- Agents:
  - There is a full class on LLM Agents by Dawn Song from Berkeley
  - A very complete tutorial
  - HuggingFace Agent 101 tutorial
  -

Questions?

References i

G. Bai, J. Liu, X. Bu, Y. He, J. Liu, Z. Zhou, Z. Lin, W. Su, T. Ge, B. Zheng, and W. Ouyang.
MT-Bench-101: A Fine-Grained Benchmark for Evaluating Large Language Models in Multi-Turn Dialogues.
In ACL, 2024.

Y. Bai, C. L. Apr, A. Askell, A. Chen, N. Dassarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. El-showk, N. Elhage, Z. Hatfield-dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. Brown, J. Clark, S. Mccandlish, C. Olah, B. Mann, and J. Kaplan.
Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback.

References ii


Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, C. Chen, C. Olsson, C. Olah, D. Hernandez, D. Drain, D. Ganguli, D. Li, E. Tran-Johnson, E. Perez, J. Kerr, J. Mueller, J. Ladish, J. Landau, K. Ndousse, K. Lukosuite, L. Lovitt, M. Sellitto, N. Elhage, N. Schiefer, N. Mercado, N. DasSarma, R. Lasenby, R. Larson, S. Ringer, S. Johnston, S. Kravec, S. E. Showk, S. Fort, T. Lanham, T. Telleen-Lawton, T. Conerly, T. Henighan, T. Hume, S. R. Bowman, Z. Hatfield-Dodds, B. Mann, D. Amodei, N. Joseph, S. McCandlish, T. Brown, and J. Kaplan.
Constitutional AI: Harmlessness from AI Feedback.
2022.

References iii

A. Barbaresi.
Trafilatura: A web scraping library and command-line tool for text discovery and extraction.
ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the System Demonstrations, pages 122–131, 2021.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei.
Language Models are Few-Shot Learners.
2020.

References iv

T. Cai, X. Wang, T. Ma, X. Chen, and D. Zhou.
Large Language Models As Tool Makers.
12th International Conference on Learning Representations, ICLR 2024, pages 1–23, 2024.

W. L. Chiang, L. Zheng, Y. Sheng, A. N. Angelopoulos, T. Li, D. Li, B. Zhu, H. Zhang, M. I. Jordan, J. E. Gonzalez, and I. Stoica.
Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference.
Proceedings of the 41 st International Conference on Machine Learning, 235:8359–8388, 2024.

References v

E. Choi, H. He, M. Iyyer, M. Yatskar, W. T. Yih, Y. Choi, P. Liang, and L. Zettlemoyer.
QUAC: Question answering in context.
Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018, pages 2174–2184, 2018.

H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Robert, D. Zhou, Q. V. Le, and J. Wei.
Scaling Instruction-Finetuned Language Models.
Journal of Machine Learning Research, 25:1–53, 2024.


M. Conover, M. Hayes, A. Mathur, J. Xie, J. Wan, S. Shah, A. Ghodsi, P. Wendell, M. Zaharia, and R. Xin.
Free Dolly: Introducing the World's First Truly Open Instruction-Tuned LLM, 2023.


J. Dodge, M. Sap, A. Marasović, W. Agnew, G. Ilharco, D. Groeneveld, M. Mitchell, and M. Gardner.
Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus.
EMNLP 2021 - 2021 Conference on Empirical Methods in Natural Language Processing, Proceedings, (Table 1):1286–1305, 2021.


Y. Dubois, B. Galambosi, P. Liang, and T. B. Hashimoto.
Length-Controlled AlpacaEval: A Simple Way to Debias Automatic Evaluators.
pages 1–11, 2024.


L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, S. Presser, and C. Leahy.
The Pile: An 800GB Dataset of Diverse Text for Language Modeling.
2020.


S. Gunasekar, Y. Zhang, J. Aneja, C. Cesar, T. Mendes, A. D. Giorno, S. Gopi, M. Javaheripi, P. Kauffmann, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, and Y. Singh B.
Textbooks Are All You Need.
pages 1–26, 2023.


D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt.
Measuring Massive Multitask Language Understanding.
In ICLR, 2021.


D. Hernandez, T. Brown, T. Conerly, N. DasSarma, D. Drain, S. El-Showk, N. Elhage, Z. Hatfield-Dodds, T. Henighan, T. Hume, S. Johnston, B. Mann, C. Olah, C. Olsson, D. Amodei, N. Joseph, J. Kaplan, and S. McCandlish.
Scaling Laws and Interpretability of Learning from Repeated Data.
pages 1–23, 2022.


Y. Hwang, Y. Kim, H. Bae, H. Lee, J. Bang, and K. Jung.
Dialogizer: Context-aware Conversational-QA Dataset Generation from Textual Sources.
EMNLP 2023 - 2023 Conference on Empirical Methods in Natural Language Processing, Proceedings, pages 8806–8828, 2023.


S. Jeong, J. Baek, S. J. Hwang, and J. C. Park.
Realistic Conversational Question Answering with Answer Selection based on Calibrated Confidence and Uncertainty Measurement.
EACL 2023 - 17th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, pages 477–490, 2023.


M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer.
TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension.
ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers), 1:1601–1611, 2017.

References x

D. Kocetkov, R. Li, L. B. Allal, J. Li, C. Mou, C. M. Ferrandis, Y. Jernite, M. Mitchell, S. Hughes, T. Wolf, D. Bahdanau, L. von Werra, and H. de Vries.
The Stack: 3 TB of permissively licensed source code.
Transactions on Machine Learning Research, pages 1–27, 2022.

T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa.
Large Language Models are Zero-Shot Reasoners.
Number NeurIPS, 2022.

LAION.
The Open-Instruction-Generalist Dataset, 2023.


K. Lee, D. Ippolito, A. Nystrom, C. Zhang, D. Eck, C. Callison-Burch, and N. Carlini.
Deduplicating Training Data Makes Language Models Better.
Proceedings of the Annual Meeting of the Association for Computational Linguistics, 1:8424–8445, 2022.


P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. T. Yih, T. Rocktäschel, S. Riedel, and D. Kiela.
Retrieval-augmented generation for knowledge-intensive NLP tasks.
Advances in Neural Information Processing Systems, 2020-Decem, 2020.

B. Li, T. Yan, Y. Pan, Z. Xu, J. Luo, R. Ji, S. Liu, H. Dong, Z. Lin, and Y. Wang.
MMedAgent: Learning to Use Medical Tools with Multi-modal Agent.
In EMNLP, 2024.

X. Li, T. Zhang, Y. Dubois, R. Taori, I. Gulrajani, C. Guestrin, P. Liang, and T. B. Hashimoto.
AlpacaEval: An Automatic Evaluator of Instruction-following Models.
\url{https://github.com/tatsu-lab/alpaca_eval}, 2023.

Z. Li, X. Xu, T. Shen, C. Xu, J.-C. Gu, and C. Tao.
Leveraging Large Language Models for NLG Evaluation: A Survey.
In EMNLP, pages 16028–16045, 2024.


S. Lin, J. Hilton, and O. Evans.
TruthfulQA: Measuring How Models Mimic Human Falsehoods.
Proceedings of the Annual Meeting of the Association for Computational Linguistics, 1:3214–3252, 2022.


S. Longpre, L. Hou, T. Vu, A. Webson, H. W. Chung, Y. Tay, D. Zhou, Q. V. Le, B. Zoph, J. Wei, and A. Roberts.
The Flan Collection: Designing Data and Methods for Effective Instruction Tuning.
In Proceedings of Machine Learning Research, volume 202, pages 22631–22648, 2023.


J. Lu, T. Holleis, Y. Zhang, B. Aumayer, F. Nan, F. Bai, S. Ma, S. Ma, M. Li, G. Yin, Z. Wang, and R. Pang.
ToolSandbox: A Stateful, Conversational, Interactive Evaluation Benchmark for LLM Tool Use Capabilities.
2024.


G. Mialon, C. Fourrier, C. Swift, T. Wolf, Y. LeCun, and T. Scialom.
Gaia: a Benchmark for General Ai Assistants.
12th International Conference on Learning Representations, ICLR 2024, pages 1–24, 2024.


A. Mohammadshahi, T. Scialom, M. Yazdani, P. Yanki, A. Fan, J. Henderson, and M. Saeidi.
RQUGE: Reference-Free Metric for Evaluating Question Generation by Answering the Question.
Proceedings of the Annual Meeting of the Association for Computational Linguistics, pages 6845–6867, 2023.


L. Ouyang, J. Wu, X. Jiang, D. Ameida, C. L. Wainwright, P. Mishkin, C. L. Mar, J. Hilton, A. Askell, P. Christiano, J. Leike, and R. Lowe.
Training language models to follow instructions with human feedback.
arXiv, https://op, 2022.


J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein.
Generative Agents: Interactive Simulacra of Human Behavior.
In UIST '23: Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology, volume 1. Association for Computing Machinery, 2023.


G. Penedo, H. Kydlíček, L. B. Allal, A. Lozhkov, M. Mitchell, C. Raffel, L. Von Werra, and T. Wolf.
The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale.
2024.

G. Penedo, Q. Malartic, D. Hesslow, R. Cojocaru, A. Cappelli, H. Alobeidli, B. Pannier, E. Almazrouei, and J. Launay.
The RefinedWeb Dataset for Falcon LLM: Outperforming Curated Corpora with Web Data, and Web Data Only.
2023.

J.-L. Peng, S. Cheng, E. Diau, Y.-Y. Shih, P.-H. Chen, Y.-T. Lin, and Y.-N. Chen.
A Survey of Useful LLM Evaluation.
2024.

O. Press, M. Zhang, S. Min, L. Schmidt, N. A. Smith, and M. Lewis.

Measuring and Narrowing the Compositionality Gap in Language Models.
2022.


Y. Qin, S. Liang, Y. Ye, K. Zhu, L. Yan, Y. Lu, Y. Lin, X. Cong, X. Tang, B. Qian, S. Zhao, L. Hong, R. Tian, R. Xie, J. Zhou, M. Gerstein, D. Li, Z. Liu, and M. Sun.
Toolllm: Facilitating Large Language Models To Master 16000+ Real-World Apis.
12th International Conference on Learning Representations, ICLR 2024, 2024.


R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn.
Direct Preference Optimization: Your Language Model is Secretly a Reward Model.
2023.


S. Reddy, D. Chen, and C. D. Manning.
CoQA: A Conversational Question Answering Challenge.
Transactions of the Association for Computational Linguistics, 7:249–266, 2019.


V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja, M. Dey, M. S. Bari, C. Xu, U. Thakker, S. S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Fevry, J. A. Fries, R. Teehan, S. Biderman, L. Gao, T. Bers, T. Wolf, and A. M. Rush.
Multitask Prompted Training Enables Zero-Shot Task Generalization.
2021.

References xx

R. Schaeffer, B. Miranda, and S. Koyejo.
Are Emergent Abilities of Large Language Models a Mirage?
In Neurips, pages 1–14, 2023.

T. Schick, M. Lomeli, J. Dwivedi-yu, and R. Dessì.
Toolformer: Language Models Can Teach Themselves to Use Tools.
Neurips, 2023.

R. Sennrich, B. Haddow, and A. Birch.
Neural Machine Translation of Rare Words with Subword Units.
In ACL, pages 1715–1725, 2016.

V. Sharma, K. Padthe, N. Ardalani, K. Tirumala, R. Howes, H. Xu, P.-Y. Huang, S.-W. Li, A. Aghajanyan, G. Ghosh, and L. Zettlemoyer.
Text Quality-Based Pruning for Efficient Training of Language Models.
2024.

Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang.
HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face.
Advances in Neural Information Processing Systems, 36(NeurIPS):1–27, 2023.

N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao. **Reflexion: Language Agents with Verbal Reinforcement Learning.** Advances in Neural Information Processing Systems, 36(NeurIPS), 2023.

S. Singh, F. Vargus, D. Dsouza, B. F. Karlsson, A. Mahendiran, W.-y. Ko, H. Shandilya, J. Patel, D. Mataciunas, L. OMahony, M. Zhang, R. Hettiarachchi, J. Wilson, M. Machado, L. S. Moura, D. Krzemiński, H. Fadaei, I. Ergün, I. Okoh, A. Alaagib, O. Mudannayake, Z. Alyafeai, V. M. Chien, S. Ruder, S. Guthikonda, E. A. Alghamdi, S. Gehrmann, N. Muennighoff, M. Bartolo, J. Kreutzer, A. Üstün, M. Fadaee, and S. Hooker. **Aya Dataset: An Open-Access Collection for Multilingual Instruction Tuning.** In ACL, 2024.


L. Soldaini, R. Kinney, A. Bhagia, D. Schwenk, D. Atkinson, R. Authur, B. Bogin, K. Chandu, J. Dumas, Y. Elazar, V. Hofmann, A. H. Jha, S. Kumar, L. Lucy, X. Lyu, N. Lambert, I. Magnusson, J. Morrison, N. Muennighoff, A. Naik, C. Nam, M. E. Peters, A. Ravichander, K. Richardson, Z. Shen, E. Strubell, N. Subramani, O. Tafjord, P. Walsh, L. Zettlemoyer, N. A. Smith, H. Hajishirzi, I. Beltagy, D. Groeneveld, J. Dodge, and K. Lo.
Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research.
In ACL, 2024.


A. Srivastava, A. Rastogi, A. Rao, A. A. M. Shoeb, A. Abid, A. Fisch, A. R. Brown, A. Santoro, A. Gupta, A. Garriga-Alonso, A. Kluska, A. Lewkowycz, A. Agarwal, A. Power, A. Ray, A. Warstadt, A. W. Kocurek, A. Safaya, A. Tazarv, A. Xiang, A. Parrish, A. Nie, A. Hussain, A. Askell, A. Dsouza, A. Slone, A. Rahane, A. S. Iyer, A. Andreassen, A. Madotto, A. Santilli, A. Stuhlmüller, A. Dai, A. La, A. Lampinen, A. Zou, A. Jiang, A. Chen, A. Vuong, A. Gupta, A. Gottardi, A. Norelli, A. Venkatesh, A. Gholamidavoodi, A. Tabassum, A. Menezes, A. Kirubarajan, A. Mullokandov, A. Sabharwal, A. Herrick, A. Efrat, A. Erdem, A. Karakaş, B. R. Roberts, B. S. Loe, B. Zoph, B. Bojanowski, B. Özyurt, B. Hedayatnia, B. Neyshabur, B. Inden, B. Stein, B. Ekmekci, B. Y. Lin, B. Howald, B. Orinion, C. Diao, C. Dour, C. Stinson, C. Argueta, C. F. Ramírez, C. Singh, C. Rathkopf, C. Meng, C. Baral, C. Wu, C. Callison-Burch, C. Waites, C. Voigt,

C. D. Manning, C. Potts, C. Ramirez, C. E. Rivera, C. Siro,
C. Raffel, C. Ashcraft, C. Garbacea, D. Sileo, D. Garrette,
D. Hendrycks, D. Kilman, D. Roth, D. Freeman, D. Khashabi,
D. Levy, D. M. González, D. Perszyk, D. Hernandez, D. Chen,
D. Ippolito, D. Gilboa, D. Dohan, D. Drakard, D. Jurgens, D. Datta,
D. Ganguli, D. Emelin, D. Kleyko, D. Yuret, D. Chen, D. Tam,
D. Hupkes, D. Misra, D. Buzan, D. C. Mollo, D. Yang, D.-H. Lee,
D. Schrader, E. Shutova, E. D. Cubuk, E. Segal, E. Hagerman,
E. Barnes, E. Donoway, E. Pavlick, E. Rodola, E. Lam, E. Chu,
E. Tang, E. Erdem, E. Chang, E. A. Chi, E. Dyer, E. Jerzak, E. Kim,
E. E. Manyasi, E. Zheltonozhskii, F. Xia, F. Siar,
F. Martínez-Plumed, F. Happé, F. Chollet, F. Rong, G. Mishra, G. I.
Winata, G. de Melo, G. Kruszewski, G. Parascandolo, G. Mariani,
G. Wang, G. Jaimovitch-López, G. Betz, G. Gur-Ari, H. Galijasevic,
H. Kim, H. Rashkin, H. Hajishirzi, H. Mehta, H. Bogar, H. Shevlin,

H. Schütze, H. Yakura, H. Zhang, H. M. Wong, I. Ng, I. Noble, J. Jumelet, J. Geissinger, J. Kernion, J. Hilton, J. Lee, J. F. Fisac, J. B. Simon, J. Koppel, J. Zheng, J. Zou, J. Kocoń, J. Thompson, J. Wingfield, J. Kaplan, J. Radom, J. Sohl-Dickstein, J. Phang, J. Wei, J. Yosinski, J. Novikova, J. Bosscher, J. Marsh, J. Kim, J. Taal, J. Engel, J. Alabi, J. Xu, J. Song, J. Tang, J. Waweru, J. Burden, J. Miller, J. U. Balis, J. Batchelder, J. Berant, J. Frohberg, J. Rozen, J. Hernandez-Orallo, J. Boudeman, J. Guerr, J. Jones, J. B. Tenenbaum, J. S. Rule, J. Chua, K. Kanclerz, K. Livescu, K. Krauth, K. Gopalakrishnan, K. Ignatyeva, K. Markert, K. D. Dhole, K. Gimpel, K. Omondi, K. Mathewson, K. Chiafullo, K. Shkaruta, K. Shridhar, K. McDonell, K. Richardson, L. Reynolds, L. Gao, L. Zhang, L. Dugan, L. Qin, L. Contreras-Ochando, L.-P. Morency, L. Moschella, L. Lam, L. Noble, L. Schmidt, L. He, L. O. Colón, L. Metz, L. K. Şenel, M. Bosma, M. Sap, M. ter Hoeve,

M. Farooqi, M. Faruqui, M. Mazeika, M. Baturan, M. Marelli, M. Maru, M. J. R. Quintana, M. Tolkiehn, M. Giulianelli, M. Lewis, M. Potthast, M. L. Leavitt, M. Hagen, M. Schubert, M. O. Baitemirova, M. Arnaud, M. McElrath, M. A. Yee, M. Cohen, M. Gu, M. Ivanitskiy, M. Starritt, M. Strube, M. Swędrowski, M. Bevilacqua, M. Yasunaga, M. Kale, M. Cain, M. Xu, M. Suzgun, M. Walker, M. Tiwari, M. Bansal, M. Aminnaseri, M. Geva, M. Gheini, M. V. T, N. Peng, N. A. Chi, N. Lee, N. G.-A. Krakover, N. Cameron, N. Roberts, N. Doiron, N. Martinez, N. Nangia, N. Deckers, N. Muennighoff, N. S. Keskar, N. S. Iyer, N. Constant, N. Fiedel, N. Wen, O. Zhang, O. Agha, O. Elbaghdadi, O. Levy, O. Evans, P. A. M. Casares, P. Doshi, P. Fung, P. P. Liang, P. Vicol, P. Alipoormolabashi, P. Liao, P. Liang, P. Chang, P. Eckersley, P. M. Htut, P. Hwang, P. Miłkowski, P. Patil, P. Pezeshkpour, P. Oli, Q. Mei, Q. Lyu, Q. Chen, R. Banjade, R. E. Rudolph, R. Gabriel,

R. Habacker, R. Risco, R. Millière, R. Garg, R. Barnes, R. A. Saurous, R. Arakawa, R. Raymaekers, R. Frank, R. Sikand, R. Novak, R. Sitelew, R. LeBras, R. Liu, R. Jacobs, R. Zhang, R. Salakhutdinov, R. Chi, R. Lee, R. Stovall, R. Teehan, R. Yang, S. Singh, S. M. Mohammad, S. Anand, S. Dillavou, S. Shleifer, S. Wiseman, S. Gruetter, S. R. Bowman, S. S. Schoenholz, S. Han, S. Kwatra, S. A. Rous, S. Ghazarian, S. Ghosh, S. Casey, S. Bischoff, S. Gehrmann, S. Schuster, S. Sadeghi, S. Hamdan, S. Zhou, S. Srivastava, S. Shi, S. Singh, S. Asaadi, S. S. Gu, S. Pachchigar, S. Toshniwal, S. Upadhyay, Shyamolima, Debnath, S. Shakeri, S. Thormeyer, S. Melzi, S. Reddy, S. P. Makini, S.-H. Lee, S. Torene, S. Hatwar, S. Dehaene, S. Divic, S. Ermon, S. Biderman, S. Lin, S. Prasad, S. T. Piantadosi, S. M. Shieber, S. Misherghi, S. Kiritchenko, S. Mishra, T. Linzen, T. Schuster, T. Li, T. Yu, T. Ali, T. Hashimoto, T.-L. Wu, T. Desbordes,

T. Rothschild, T. Phan, T. Wang, T. Nkinyili, T. Schick, T. Kornev, T. Tunduny, T. Gerstenberg, T. Chang, T. Neeraj, T. Khot, T. Shultz, U. Shaham, V. Misra, V. Demberg, V. Nyamai, V. Raunak, V. Ramasesh, V. U. Prabhu, V. Padmakumar, V. Srikumar, W. Fedus, W. Saunders, W. Zhang, W. Vossen, X. Ren, X. Tong, X. Zhao, X. Wu, X. Shen, Y. Yaghoobzadeh, Y. Lakretz, Y. Song, Y. Bahri, Y. Choi, Y. Yang, Y. Hao, Y. Chen, Y. Belinkov, Y. Hou, Y. Hou, Y. Bai, Z. Seid, Z. Zhao, Z. Wang, Z. J. Wang, Z. Wang, and Z. Wu.
Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models.
2022.

References xxx

R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto.
Stanford Alpaca: An Instruction-following LLaMA model.
\url{https://github.com/tatsu-lab/stanford_alpaca}, 2023.

K. Tirumala, D. Simig, A. Aghajanyan, and A. S. Morcos.
D4: Improving LLM Pretraining via Document De-Duplication and Diversification.
(NeurIPS), 2023.

X. Wang, Z. Hu, P. Lu, Y. Zhu, and J. Zhang.
SCIBENCH : Evaluating College-Level Scientific Problem-Solving Abilities of Large Language Models.
In MATH-AI: The 3rd Workshop on Mathematical Reasoning and AI @ Neurips, number NeurIPS, 2023.

X. Wang and D. Zhou.
Chain-of-Thought Reasoning Without Prompting.
pages 1–23, 2024.

Y. Wang, X. Ma, G. Zhang, Y. Ni, A. Chandra, S. Guo, W. Ren, A. Arulraj, X. He, Z. Jiang, T. Li, M. Ku, K. Wang, A. Zhuang, R. Fan, X. Yue, and W. Chen.
MMLU-Pro: A More Robust and Challenging Multi-Task Language Understanding Benchmark.
In NeurIPS, pages 1–24, 2024.


M. Weber, D. Y. Fu, Q. Anthony, Y. Oren, S. Adams, A. Alexandrov, X. Lyu, H. Nguyen, X. Yao, V. Adams, B. Athiwaratkun, R. Chalamala, K. Chen, M. Ryabinin, T. Dao, P. Liang, C. Ré, I. Rish, and C. Zhang.
RedPajama: an Open Dataset for Training Large Language Models.
In NeurIPS, pages 1–31, 2024.


J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le.
Finetuned Language Models Are Zero-Shot Learners.
ICLR, pages 1–41, 2022.


J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus.
Emergent Abilities of Large Language Models.
Transactions on Machine Learning Research, pages 1–30, 2022.


J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou.
Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.
(NeurIPS):1–43, 2022.


T. Wu, J. Lan, W. Yuan, J. Jiao, J. Weston, and S. Sukhbaatar.
Thinking LLMs: General Instruction Following with Thought Generation.
pages 1–28, 2024.

C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen.
Large Language Models as Optimizers.
2023.

S. Yao.
Language Agents: From Next-Token Prediction to Digital Automatio

PhD thesis, 2024.

S. Yao, D. Yu, J. Zhao, and T. L. Griffiths.
Tree of Thoughts : Deliberate Problem Solving with Large Language Models.
In Neurips, number 1, pages 1–11, 2023.


S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao.
React: Synergizing Reasoning and Acting in Language Models.

11th International Conference on Learning Representations, ICLR 2023, pages 1–33, 2023.


M. J. Zhang, Z. Wang, J. D. Hwang, Y. Dong, O. Delalleau, Y. Choi, E. Choi, X. Ren, and V. Pyatkin.
Diverging Preferences: When do Annotators Disagree and do Models Know?
(Table 1):1–19, 2024.

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica.
Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena.
(NeurIPS):1–29, 2023.

C. Zhou, P. Liu, P. Xu, S. Iyer, J. Sun, Y. Mao, X. Ma, A. Efrat, P. Yu, L. Yu, S. Zhang, G. Ghosh, M. Lewis, L. Zettlemoyer, and O. Levy.
LIMA: Less Is More for Alignment.
pages 1–15, 2023.

X. Zhou, Z. Su, T. Eisape, H. Kim, and M. Sap.
Is this the real life? Is this just fantasy? The Misleading Success of Simulating Social Interactions With LLMs.
In EMNLP, 2024.


Y. Zhuang, Y. Yu, K. Wang, H. Sun, and C. Zhang.
ToolQA: A Dataset for LLM Question Answering with External Tools.
Neurips, 2023.