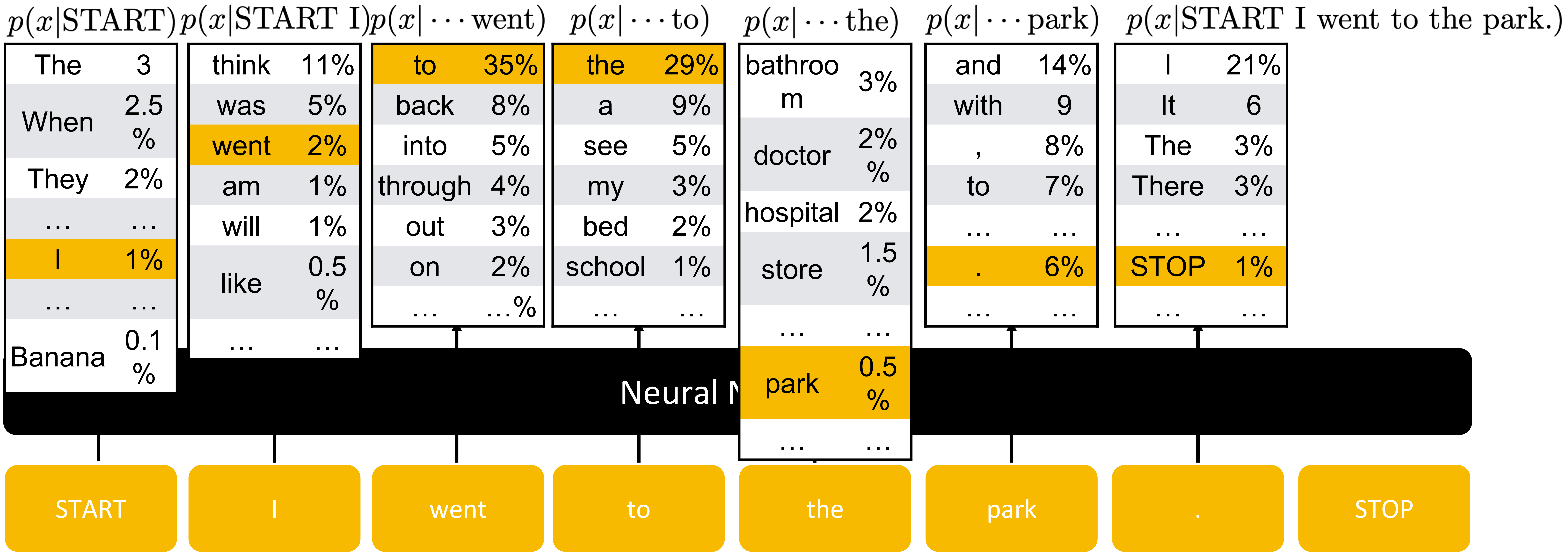
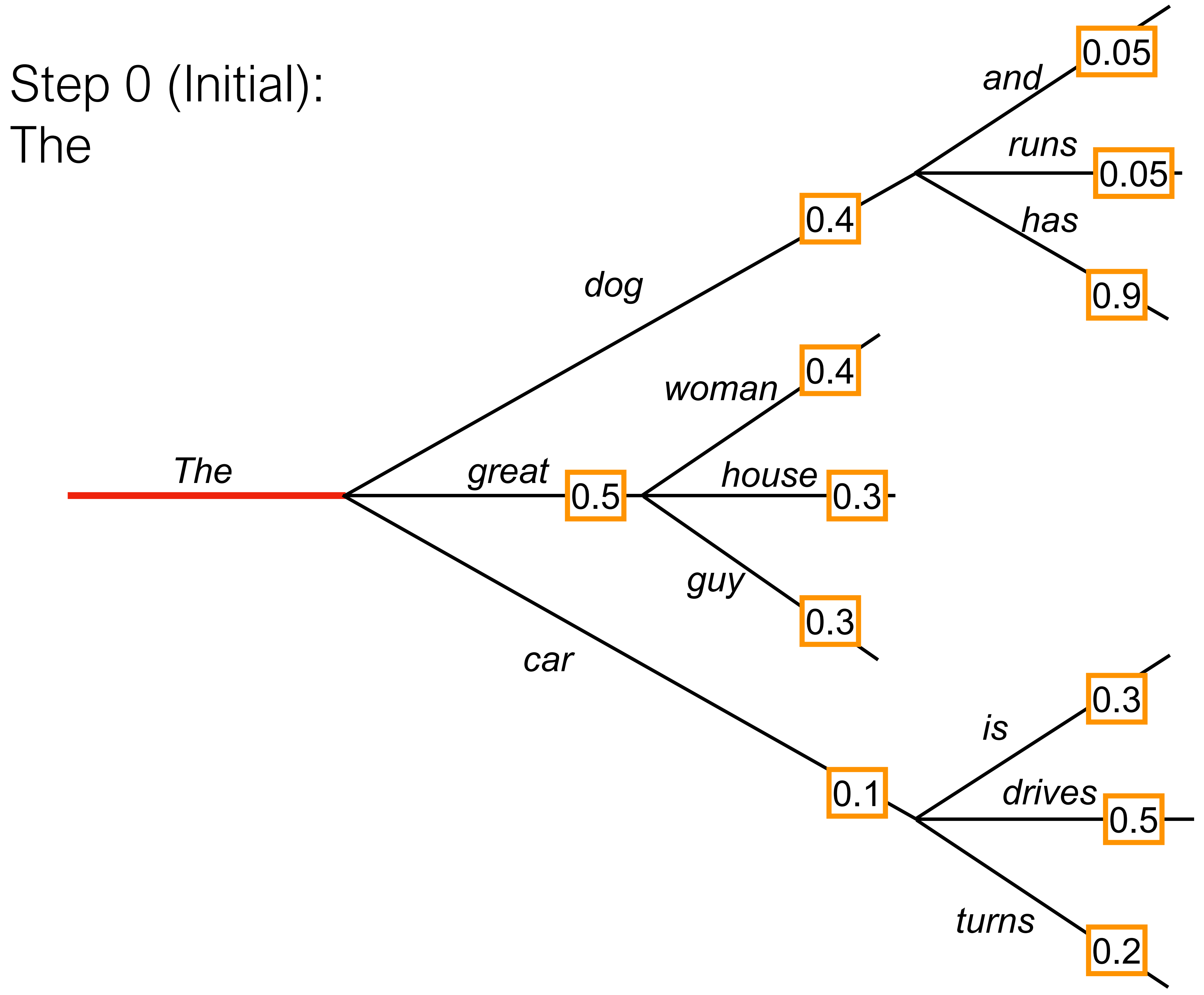
Inputs/Outputs

- **Input:** sequences of words (or tokens)
- **Output:** probability distribution over the next word (token)



Greedy Decoding vs. Beam Search

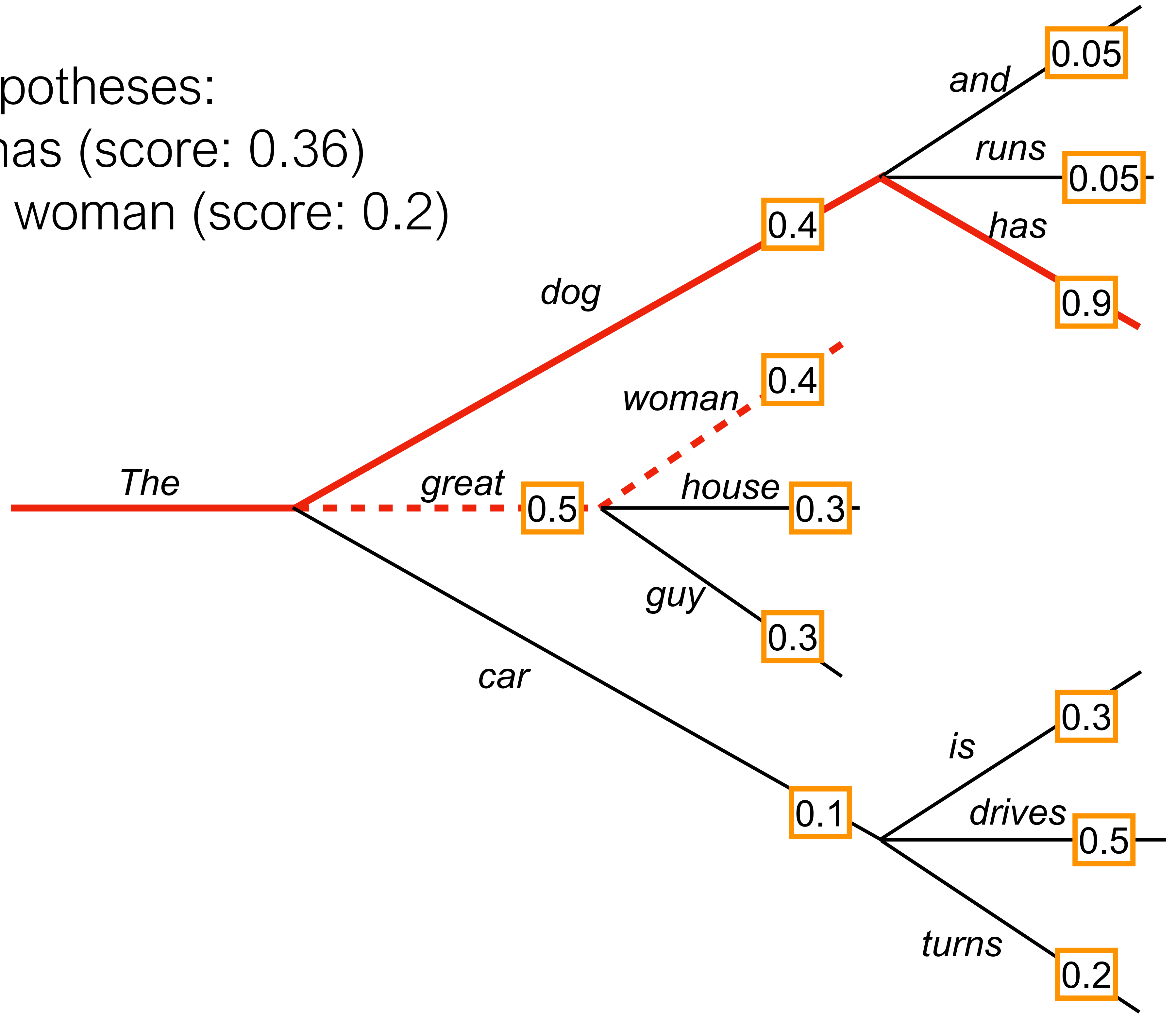
- **Greedy Decoding**
  - Choose the "currently best" token at each time step



Greedy Decoding vs. Beam Search

- **Beam Search (in this example, *beam_width* = 2)**
  - At each step, retain 2 hypotheses with the highest probability

Step 2 hypotheses:
The dog has (score: 0.36)
The great woman (score: 0.2)



Most likely sequences are repetitive

Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Continuation: The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from **the Universidad Nacional Autónoma de México (UNAM)** and **the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/ Universidad Nacional Autónoma de México/ Universidad Nacional Autónoma de México/ Universidad Nacional Autónoma de México…**

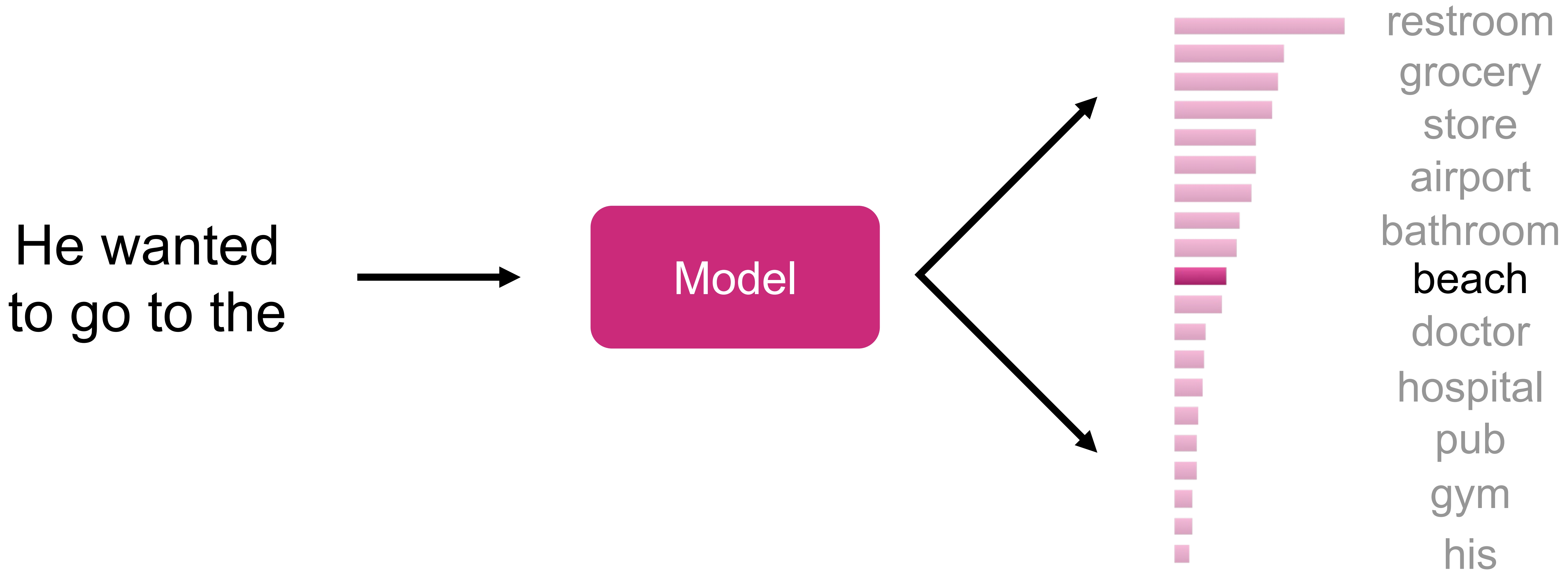
(Holtzman et al. ICLR 2020)

Time to get `random`: Sampling

- Sample a token from the token distribution at each step!

$$\hat{y}_t \sim P(y_t = w|\{y\}_{<t})$$

- It's inherently *random* so you can sample any token.



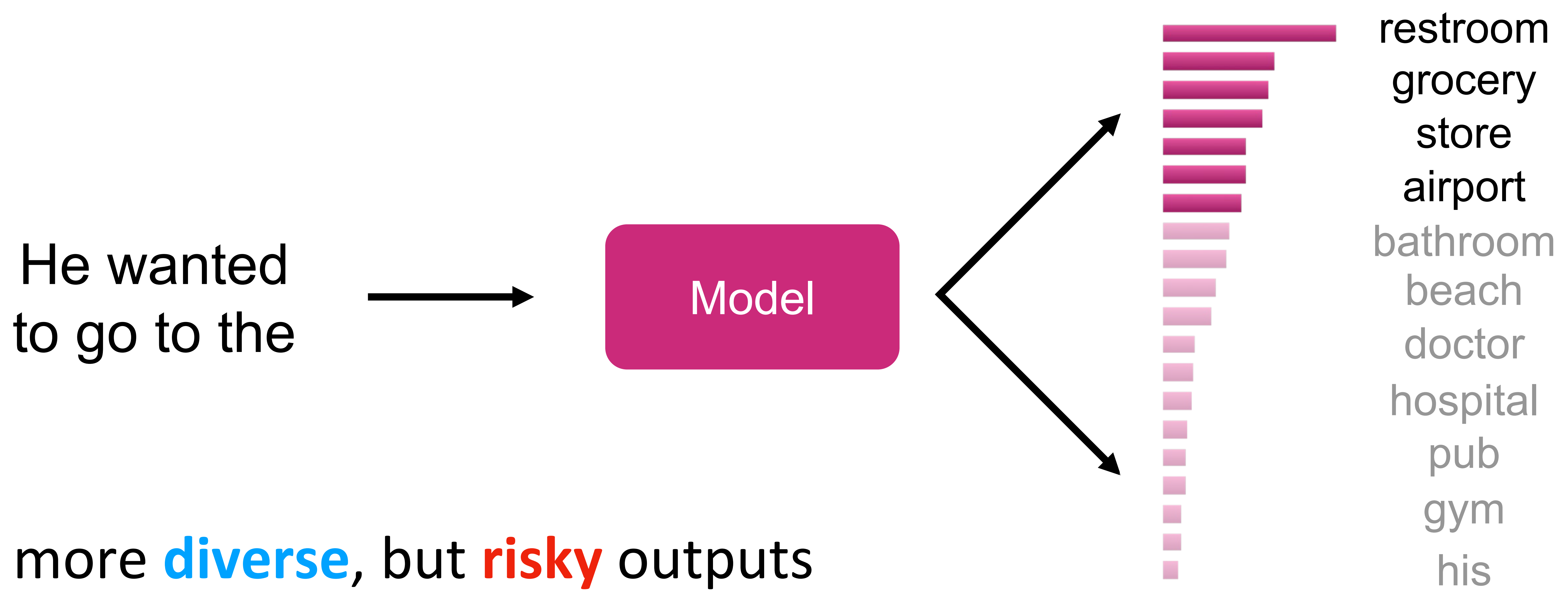
Decoding: Top-*k* Sampling

- Problem: Vanilla sampling makes *every token* in the vocabulary an option
  - Even if most of the probability mass in the distribution is over a limited set of options, the tail of the distribution could be very long and in aggregate have considerable mass (statistics speak: we have "heavy tailed" distributions)
  - Many tokens are probably really wrong in the current context.
  - Although *each of them* may be assigned a small probability, *in aggregate* they still get a high chance to be selected.

- Solution: Top-*k* sampling *(Fan et al., 2018)*
  - Only sample from the top *k* tokens in the probability distribution.

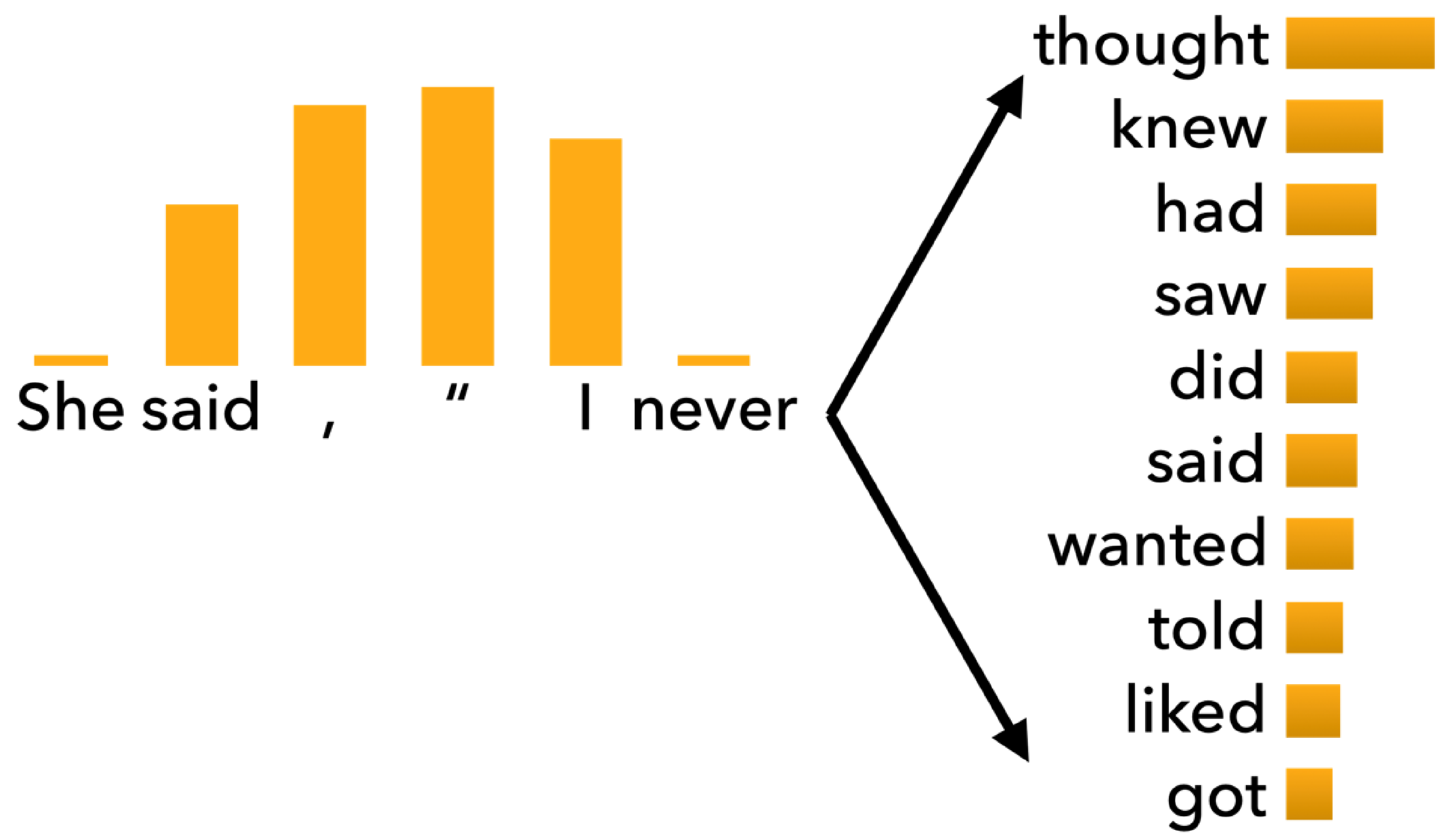
Decoding: Top-*k* Sampling

- Solution: Top-*k* sampling *(Fan et al., 2018)*
  - Only sample from the top *k* tokens in the probability distribution.
  - Common values for *k* = 10, 20, 50 (*but it's up to you!*)



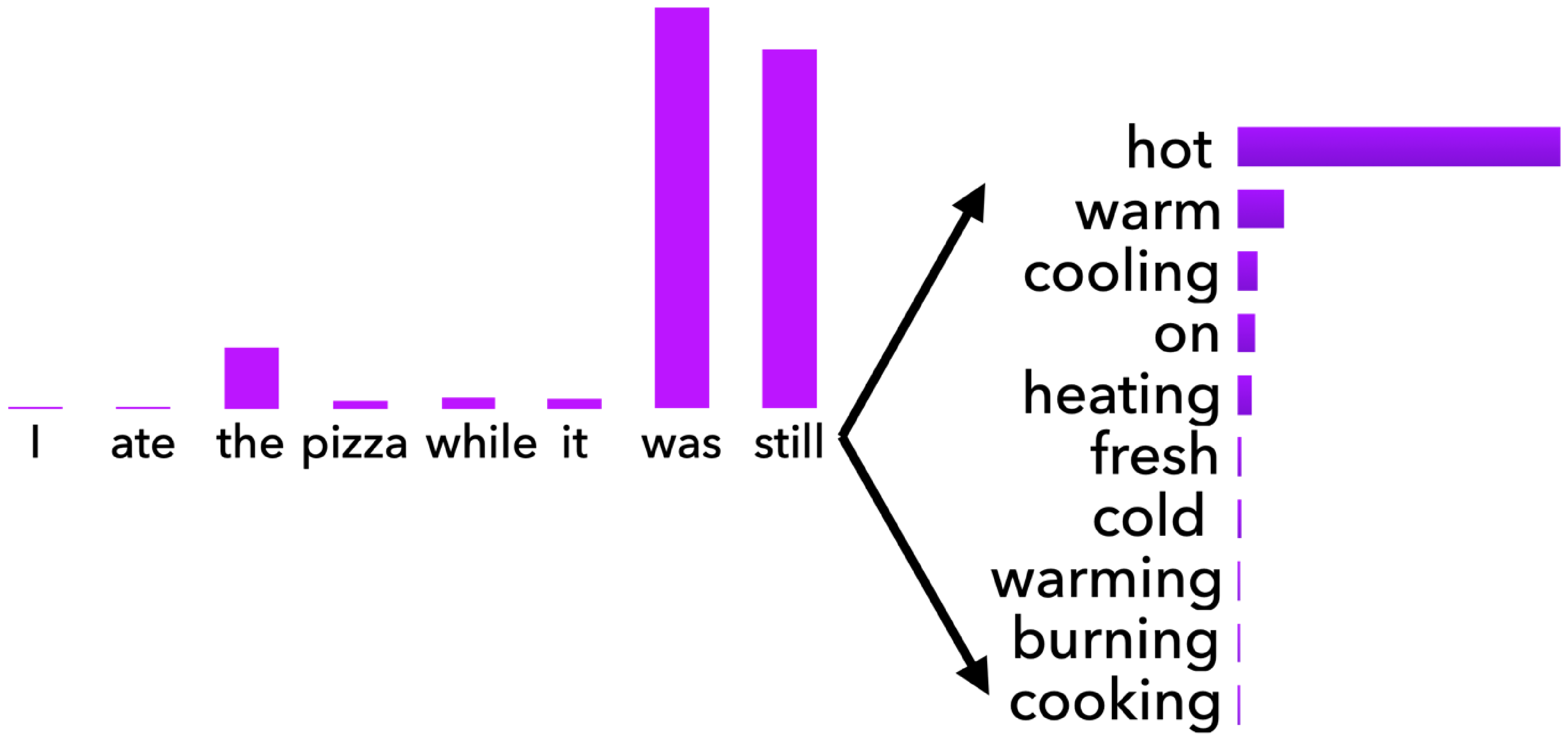
  - Increasing *k* yields more **diverse**, but **risky** outputs
  - Decreasing *k* yields more **safe** but **generic** outputs

Issues with Top-*k* Sampling



For *flat* distribution,

Top-*k* Sampling may cut off too **quickly**!



For *peaked* distribution,

Top-*k* Sampling may also cut off too **slowly**!

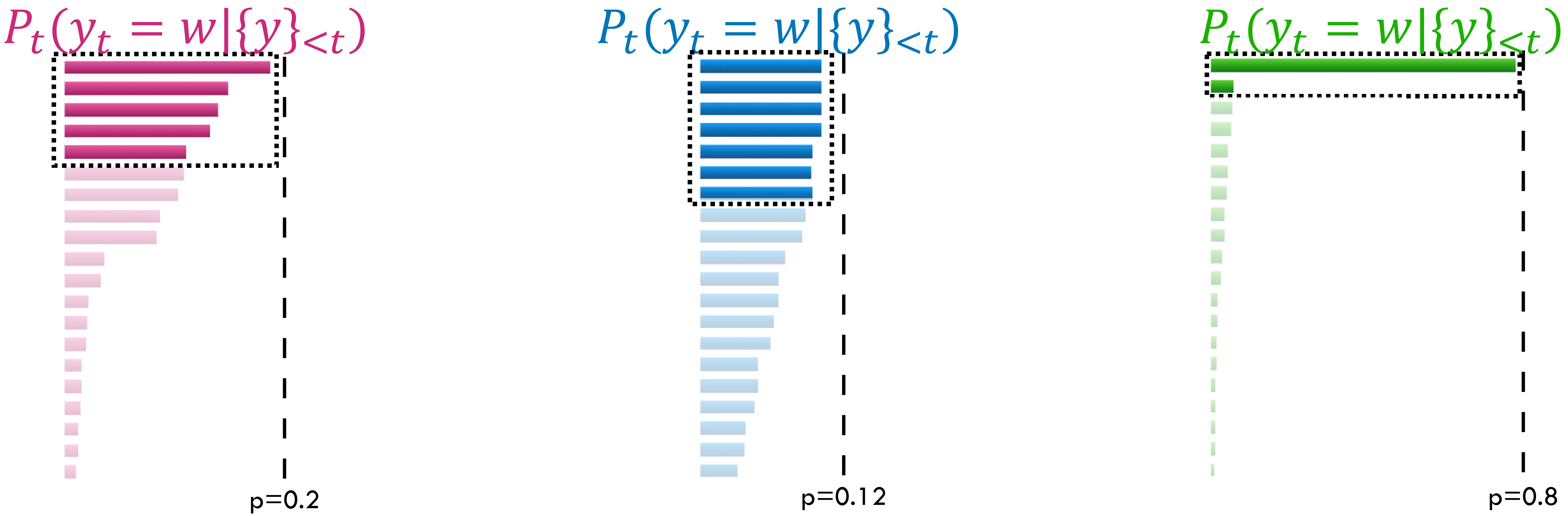
Decoding: Top-*p* *(Nucleus)* Sampling

- Problem: The token distributions we sample from are dynamic
  - When the distribution P_t is flat, small k removes many viable options.
  - When the distribution P_t is peaked, large k allows too many options a chance to be selected.

- Solution: Top-*p* sampling *(Holtzman et al., 2020)*
  - Sample from all tokens in the top p cumulative probability mass (i.e., where mass is concentrated)
  - Varies k according to the uniformity of P_t

Decoding: Top-*p* *(Nucleus)* Sampling

- Solution: Top-*p* sampling *(Holtzman et al., 2020)*
  - Sample from all tokens in the top p cumulative probability mass (i.e., where mass is concentrated)
  - Varies k according to the uniformity of P_t



Scaling randomness: Softmax temperature

- Recall: At time step t, model computes a distribution P_t by applying softmax to a vector of scores $S \in \mathbb{R}^{|V|}$

$$P_t(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w)}{\sum_{w' \in V} \exp(S_{w'})}$$

- Here, you can apply **temperature hyperparameter τ** to the softmax to rebalance P_t:

$$P_t(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w/\tau)}{\sum_{w' \in V} \exp(S_{w'}/\tau)}$$

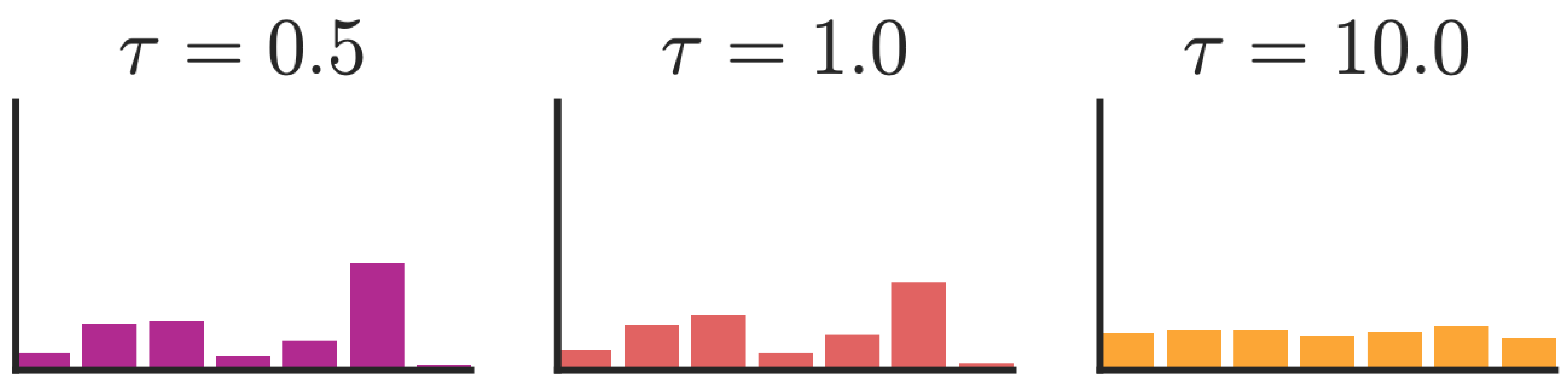
- Raise the temperature $\tau > 1$: P_t becomes more uniform
  - More diverse output (probability is spread across vocabulary)
- Lower the temperature $\tau < 1$: P_t becomes more spiky
  - Less diverse output (probability concentrated to the top tokens)

Scaling randomness: Softmax temperature

- You can apply **temperature hyperparameter τ** to the softmax to rebalance P_t:

$$P_t(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w/\tau)}{\sum_{w' \in V} \exp(S_{w'}/\tau)}$$

- Raise the temperature $\tau > 1$: P_t becomes more uniform
  - More diverse output (probability is spread across vocabulary)
- Lower the temperature $\tau < 1$: P_t becomes more spiky
  - Less diverse output (probability concentrated to the top tokens)



Scaling randomness: Softmax temperature

- You can apply **temperature hyperparameter τ** to the softmax to rebalance P_t:

$$P_t(y_t = w|\{y_{<t}\}) = \frac{\exp(S_w/\tau)}{\sum_{w' \in V} \exp(S_{w'}/\tau)}$$

- Raise the temperature $\tau > 1$: P_t becomes more uniform
  - More diverse output (probability is spread across vocabulary)
- Lower the temperature $\tau < 1$: P_t becomes more spiky
  - Less diverse output (probability concentrated to the top tokens)

NOTE: Temperature is a hyperparameter for decoding algorithm, not an algorithm itself! It can be applied for both beam search and sampling methods.

Decoding: Takeaways

- Decoding is still a challenging problem in NLG - there's a lot more work to be done!
- Different decoding algorithms can allow us to inject biases that encourage different properties of coherent natural language generation
- Some of the most impactful advances in NLG of the last few years have come from simple but effective modifications to decoding algorithms

Alignment

- **Background**: What is Alignment of LLMs?
- **Data**: How can we get the data for instruction learning?
- **Method**: How can we align LLMs with supervised fine-tuning (SFT)?
- **Evaluation**: How can we compare different LLMs in terms of alignment?

What is Alignment of LLMs?

- **Instruction Learning:** teaching base LLMs to follow instructions
- **Preference Learning:** adjusting instructed LLMs to behave as human expected

can complete your text.

can better follow your instructions.

Base LLM

Instruction Learning (Part 1)

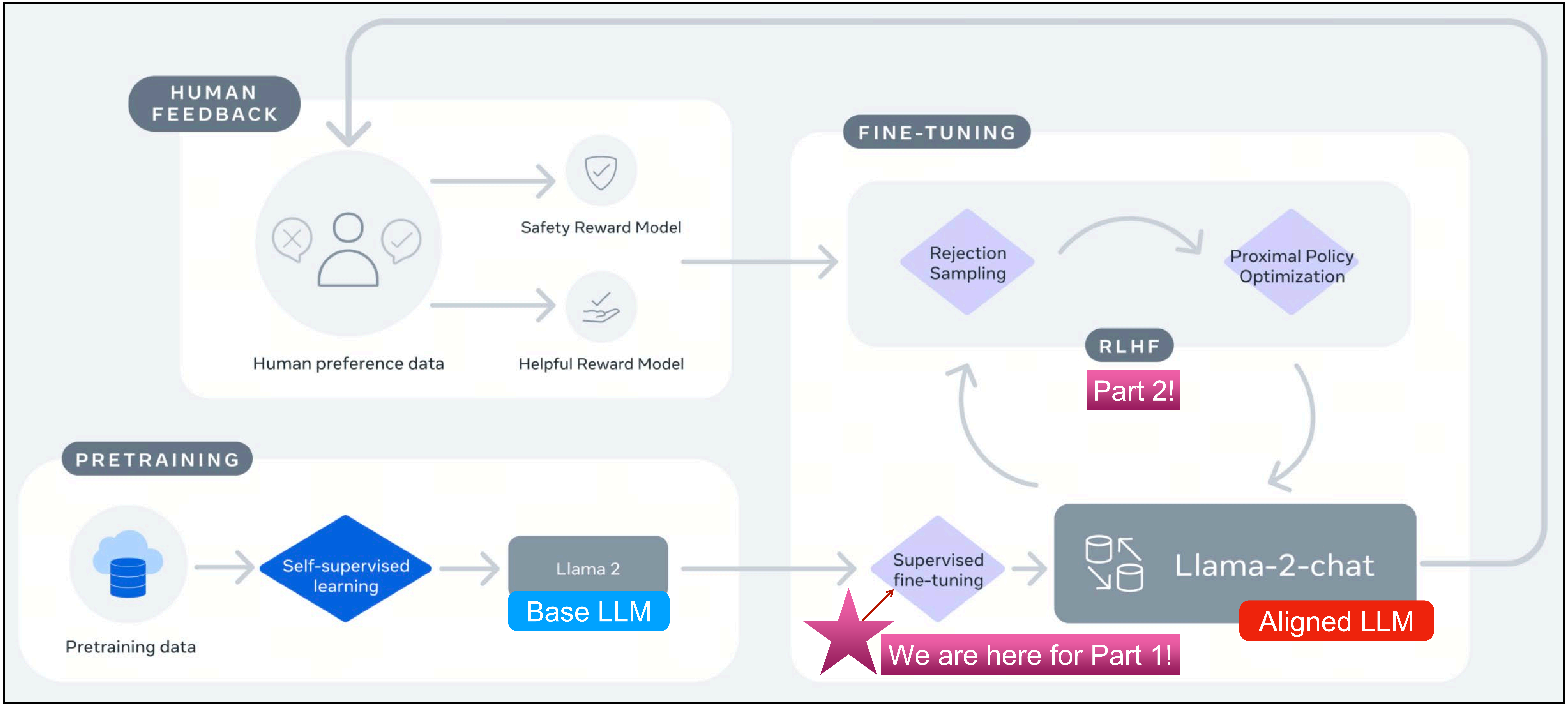
Preference Learning (Part 2)

Aligned LLM

e.g., Llama-2

e.g., Llama-2-chat

Example: Llama-2's alignment

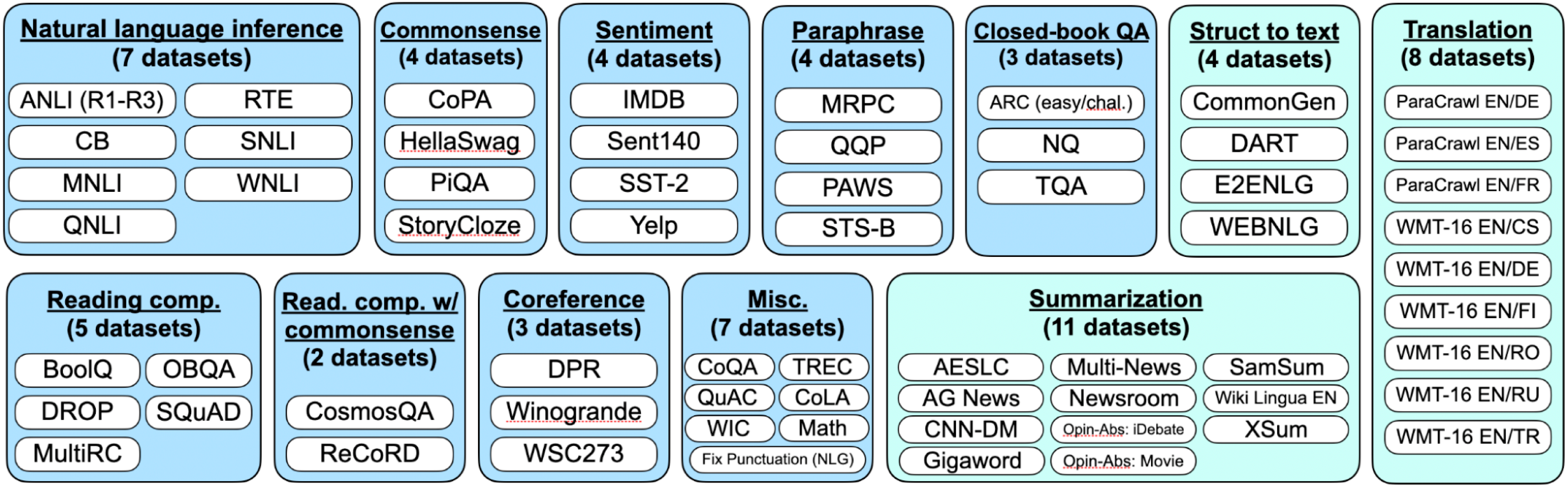


Dataset for Instruction Learning

- 1. Synthetic Conversion
- 2. Human Annotation
- 3. Collected from ChatGPT/GPT-4
  - 3.1. Community Sharing
  - 3.2. Strategic Collecting

Dataset for Instruction Learning

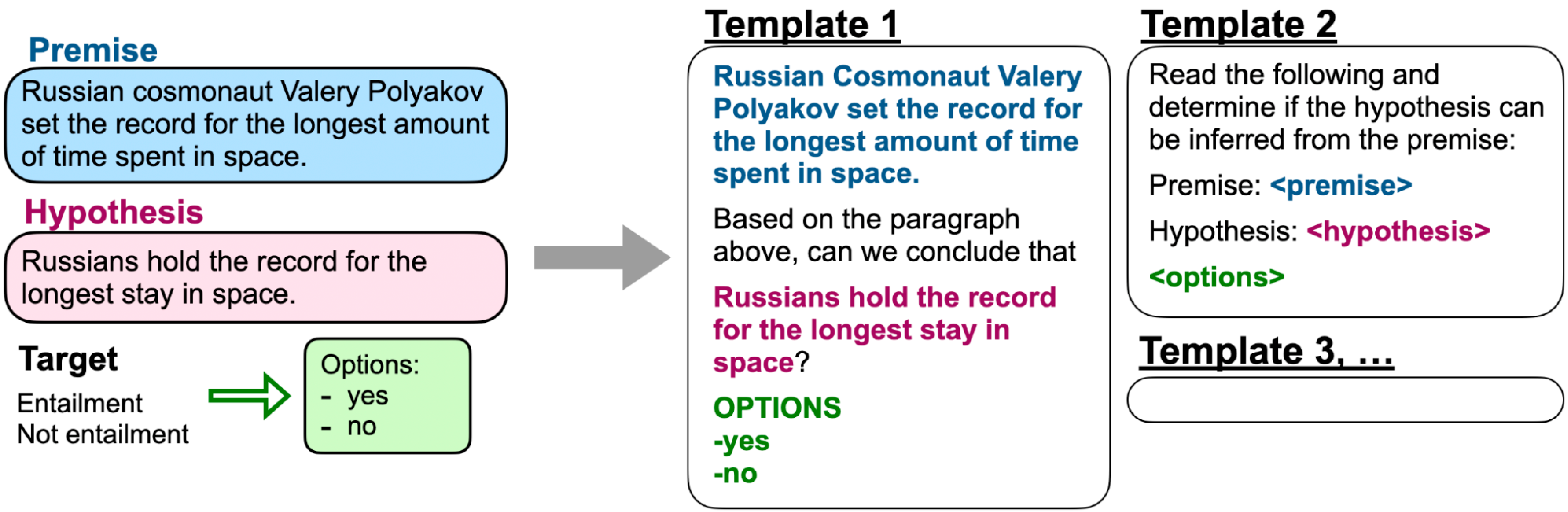
- **1. Synthetic Conversion of Existing NLP Datasets**



https://blog.research.google/2021/10/introducing-flan-more-generalizable.html

Dataset for Instruction Learning

- **1. Synthetic Conversion of Existing NLP Datasets**



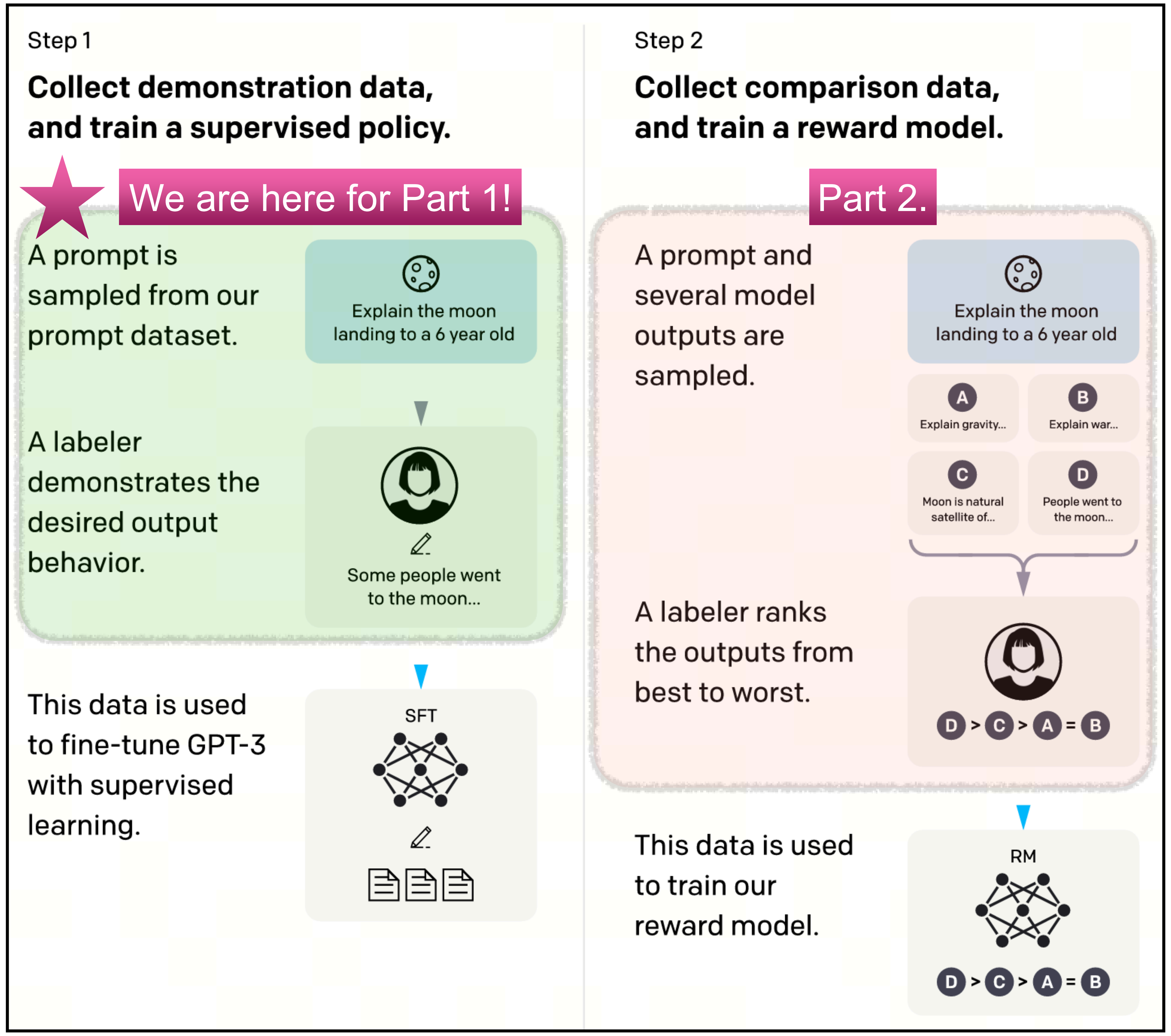
An existing NLP task:
Binary Classification

Converted to Seq2Seq tasks with different instruction templates.
—> Unified Data Formats for Massive Multi-Task Training

https://blog.research.google/2021/10/introducing-flan-more-generalizable.html

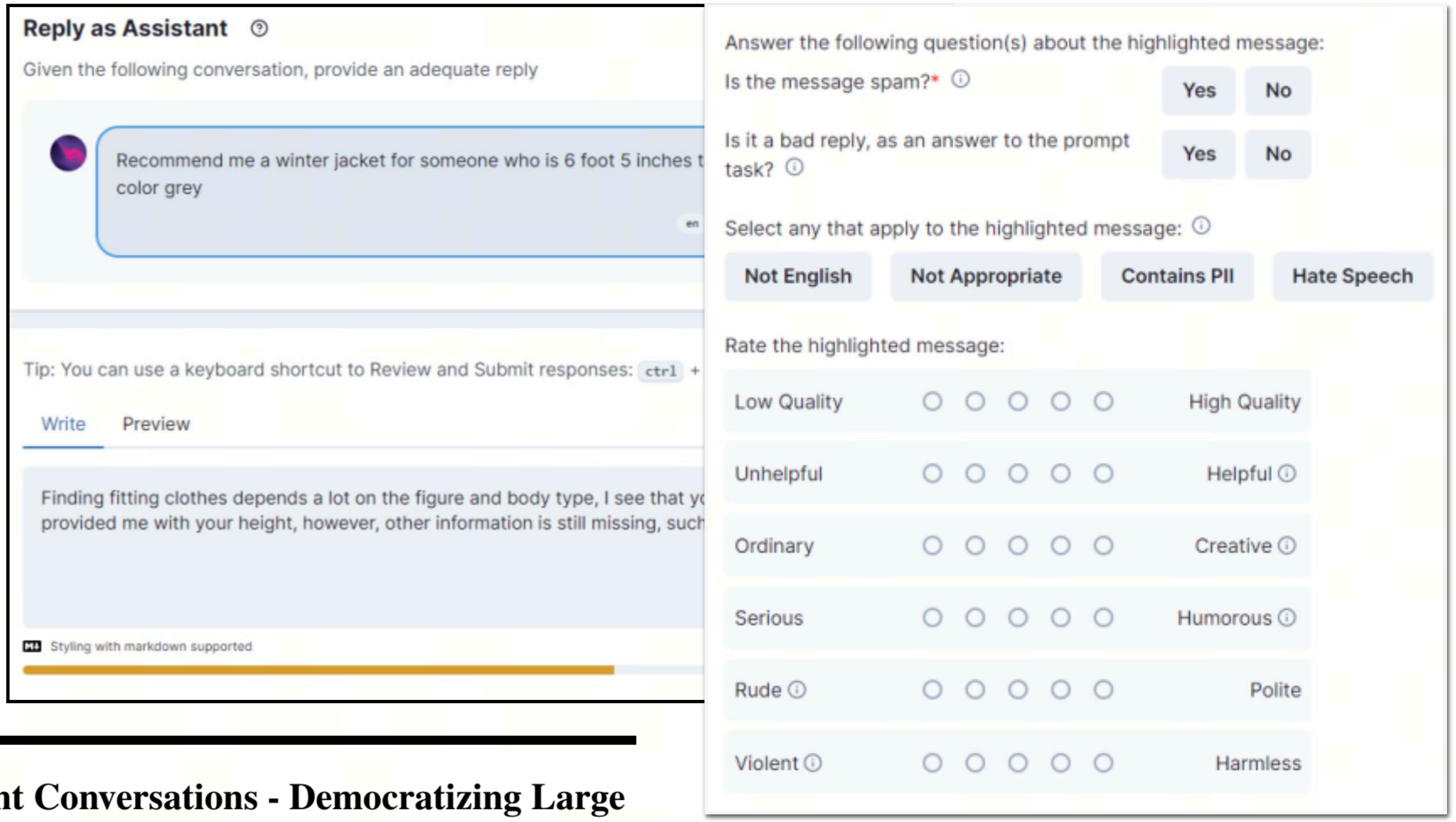
Dataset for Instruction Learning

- **2. Human Annotation:**



ChatGPT's pipeline for data collection.

OpenAssistant: An Open-Source Human Annotation Dataset



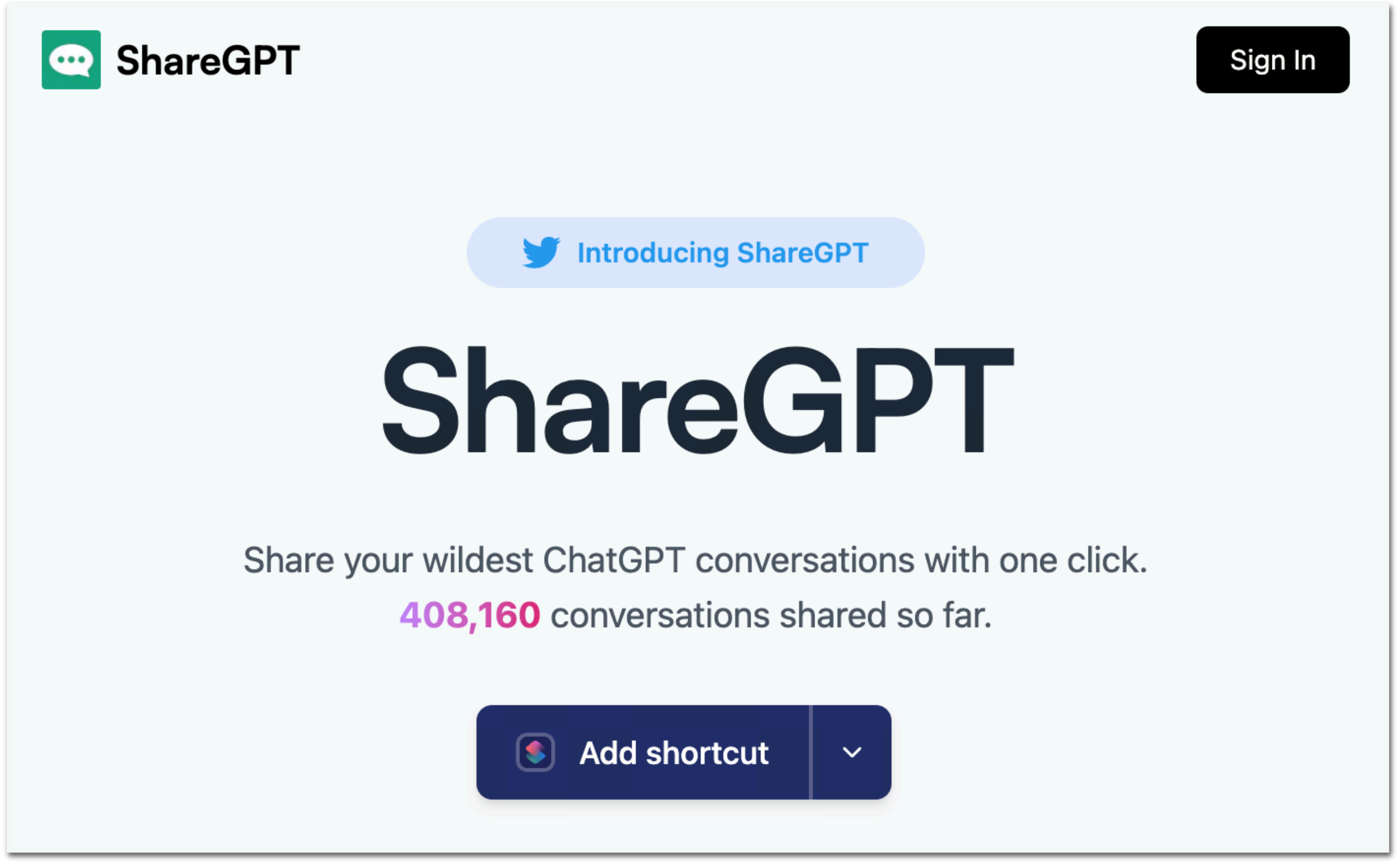
OpenAssistant Conversations - Democratizing Large Language Model Alignment

Dataset for Instruction Learning

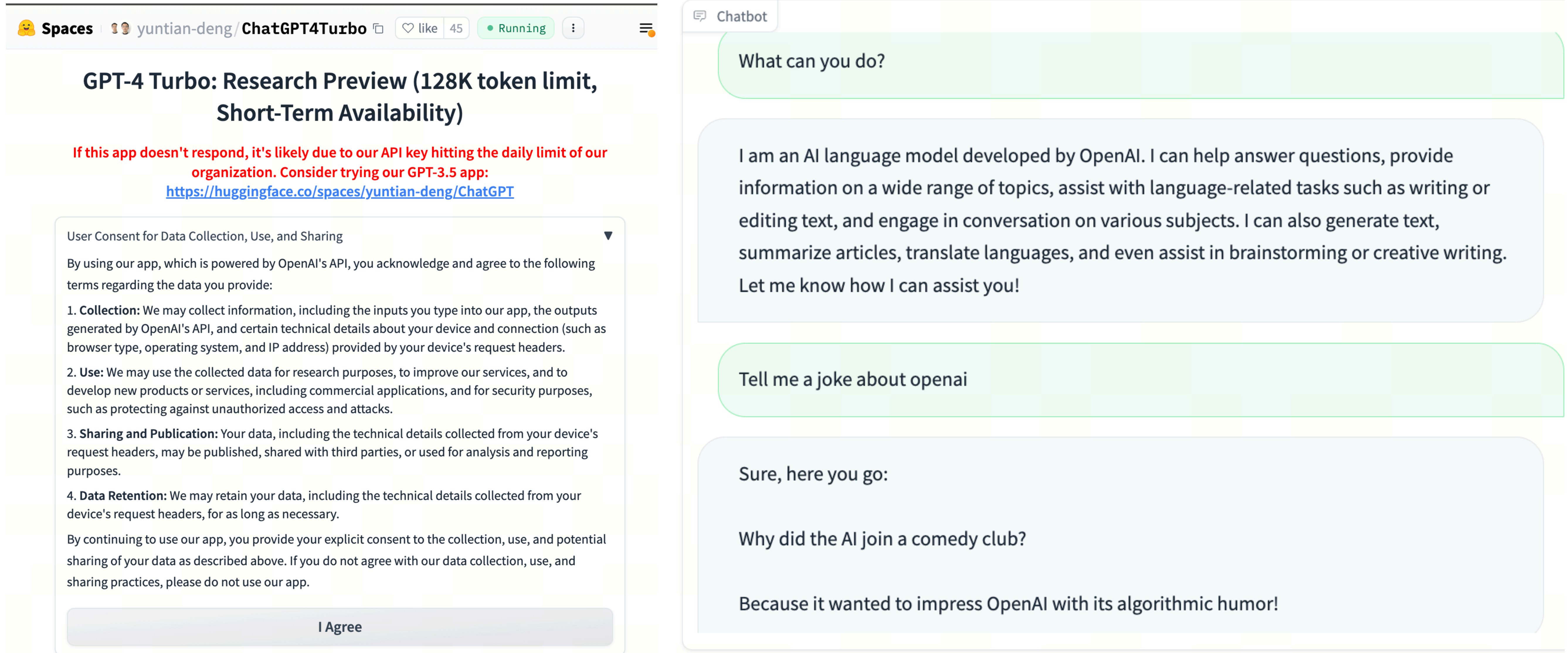
- **3.1. Community Sharing from ChatGPT**

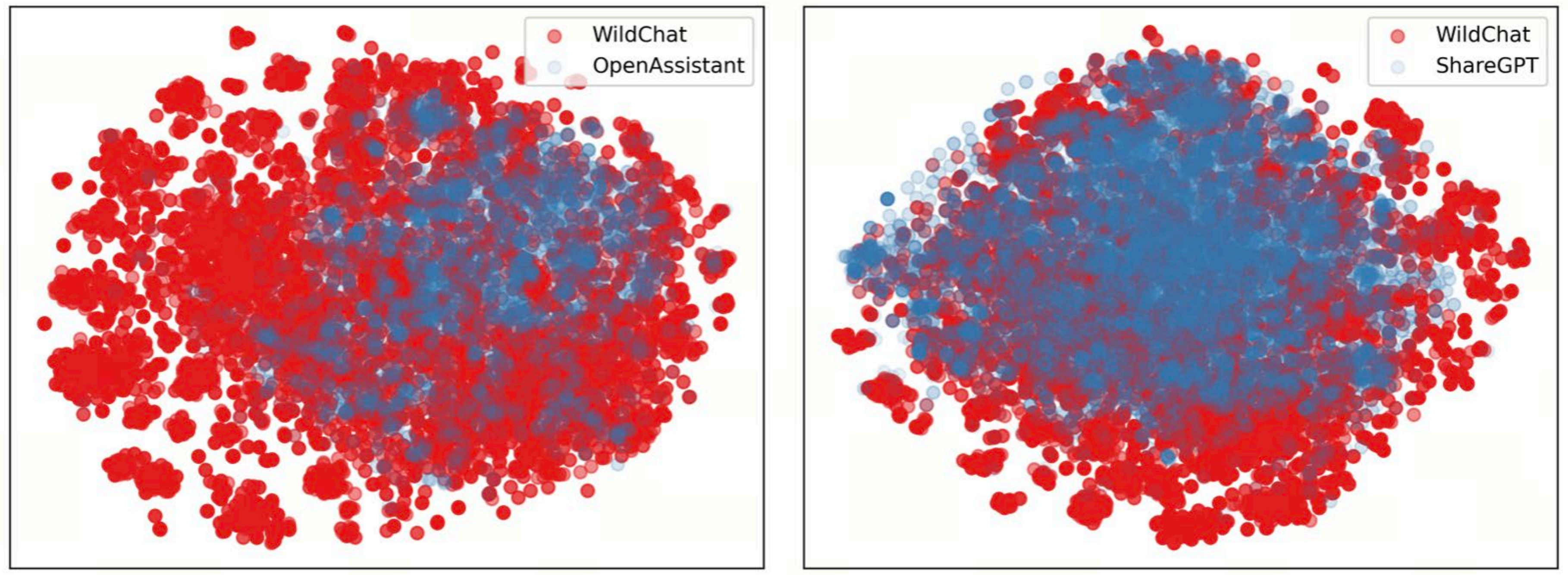
WildChat: Providing Free GPT-4 APIs for Public Users

Natural **Queries from Human Users on GhatGPT**



sharegpt.com

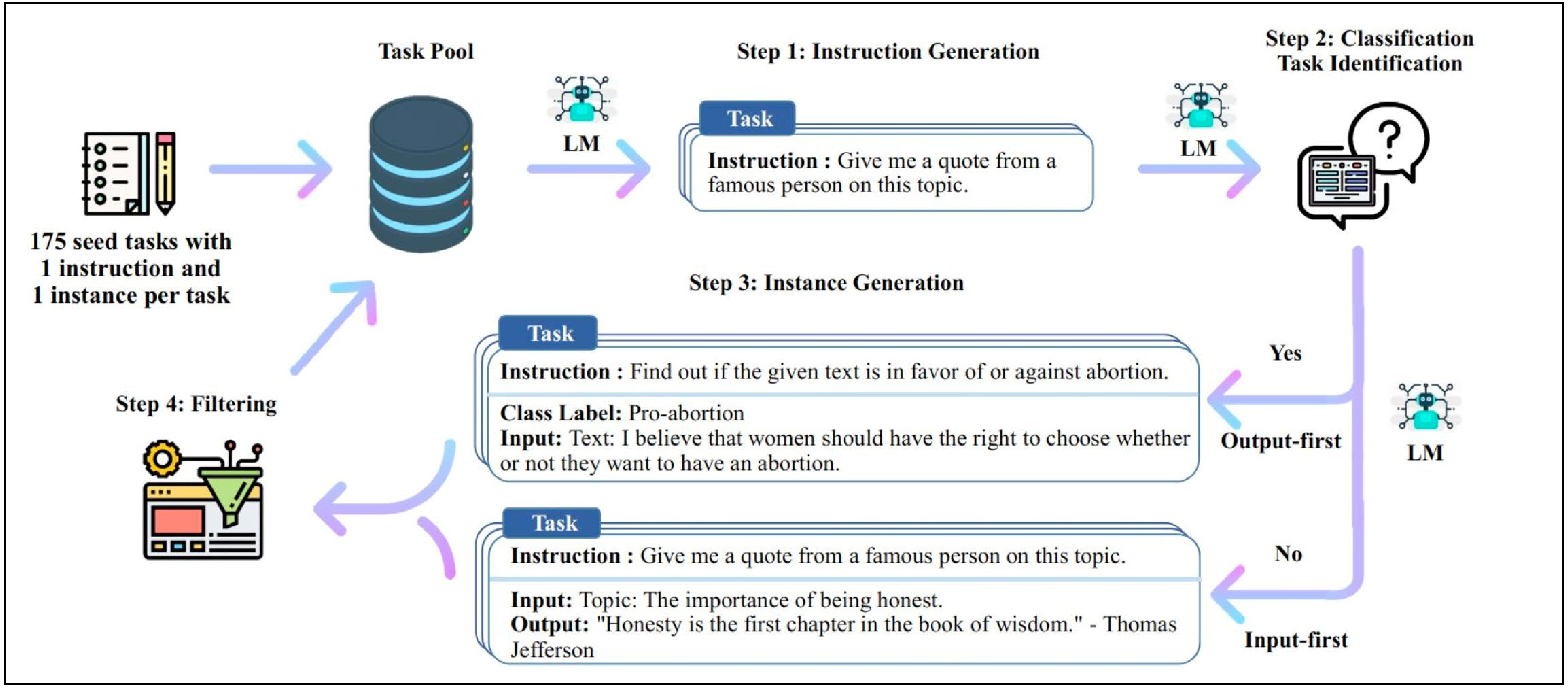




T-SNE plots of the embeddings of user prompts.

Dataset for Instruction Learning

- **3.2. Strategical Collecting Data from ChatGPT**

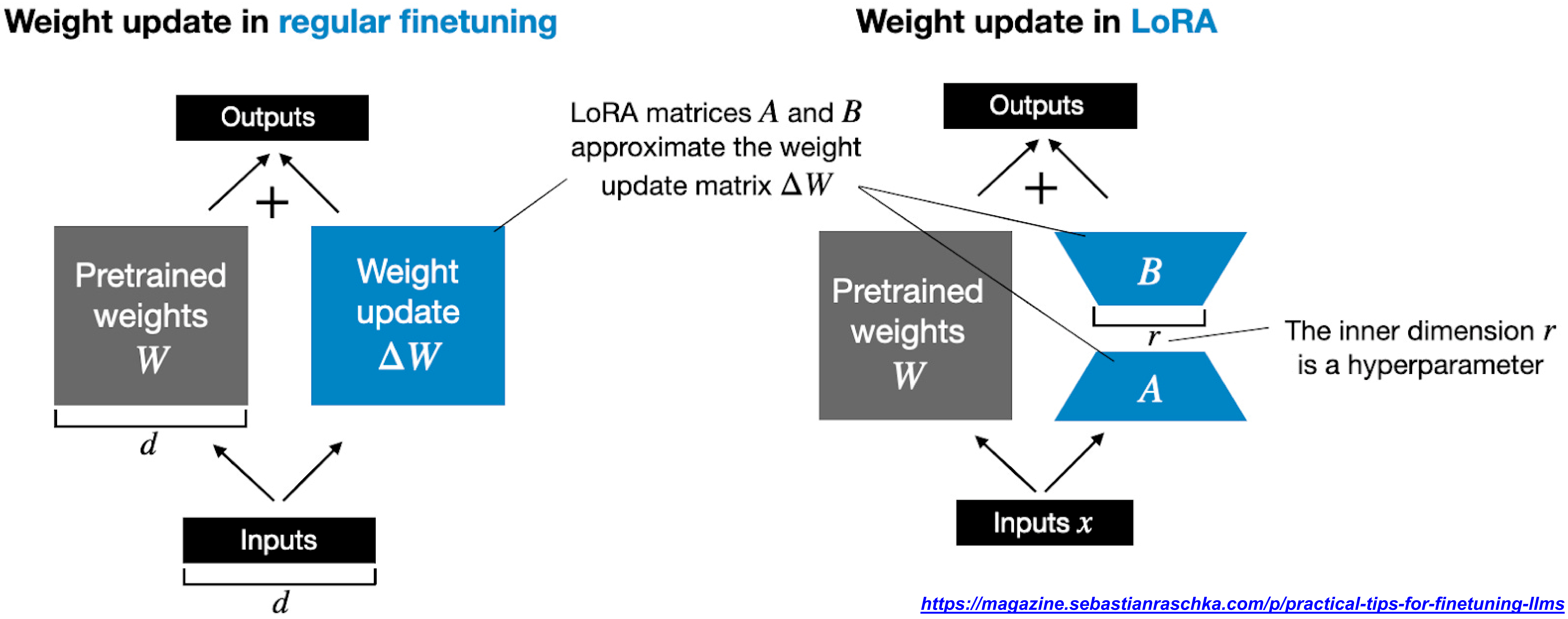


Self-instruct pipeline for data collection.

https://arxiv.org/abs/2212.10560

Efficient Fine-Tuning

- **LoRA: Low-Rank Adaptation: Motivation**



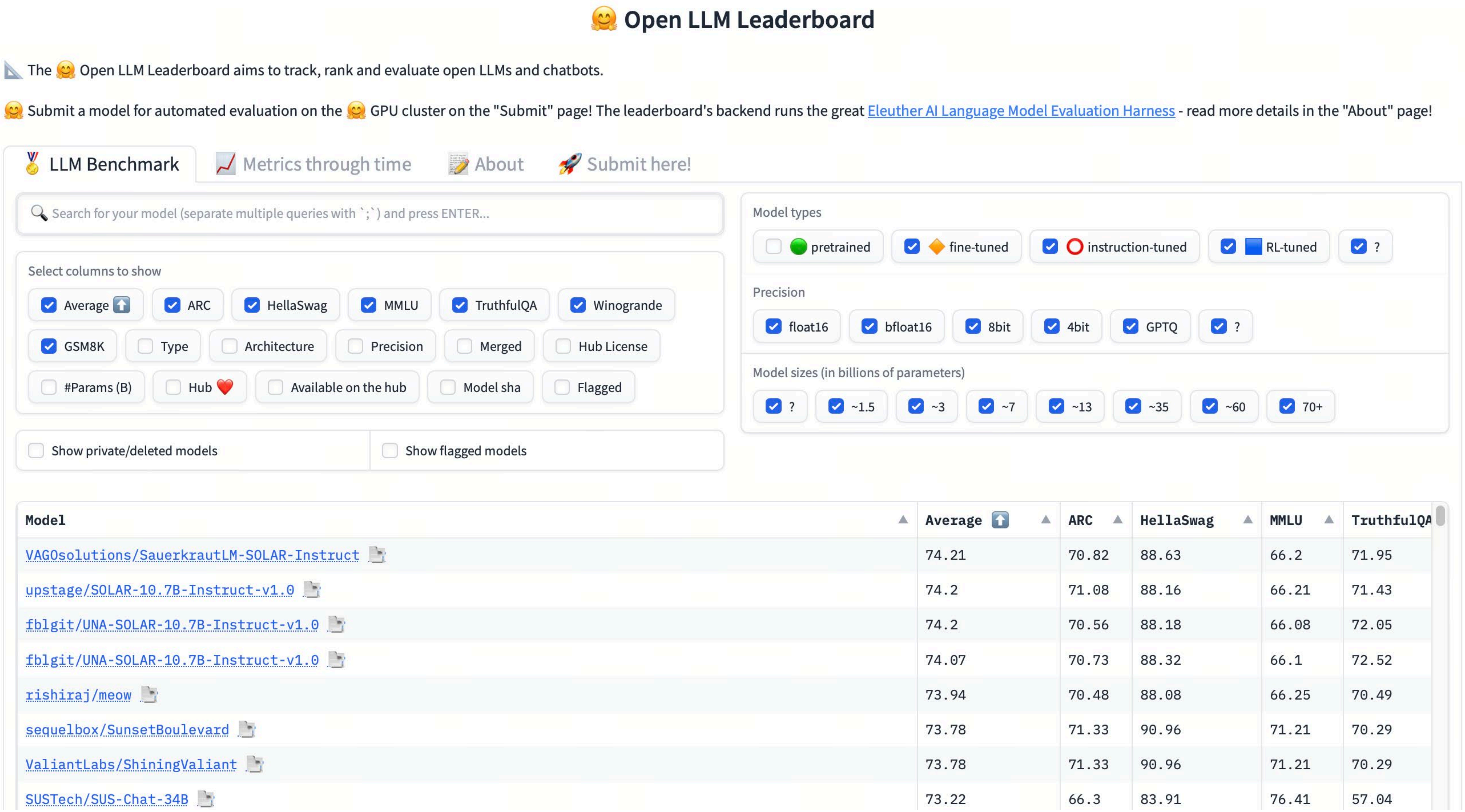
https://magazine.sebastianraschka.com/p/practical-tips-for-finetuning-llms

Evaluation of Alignment

- Benchmarking Datasets
- Human Annotation
- GPTs as Judges
- Open LLM Evaluators
- Safety Evaluation

Evaluation of LLM

- Benchmarking Datasets



🤗 Open LLM Leaderboard

📐 The 🤗 Open LLM Leaderboard aims to track, rank and evaluate open LLMs and chatbots.

🤗 Submit a model for automated evaluation on the 🤗 GPU cluster on the "Submit" page! The leaderboard's backend runs the great Eleuther AI Language Model Evaluation Harness - read more details in the "About" page!

🏅 LLM Benchmark | 📈 Metrics through time | 📝 About | 🚀 Submit here!

🔍 Search for your model (separate multiple queries with `;`) and press ENTER...

Select columns to show

☑ Average ⬆️ ☑ ARC ☑ HellaSwag ☑ MMLU ☑ TruthfulQA ☑ Winogrande
☑ GSM8K ☐ Type ☐ Architecture ☐ Precision ☐ Merged ☐ Hub License
☐ #Params (B) ☐ Hub ❤️ ☐ Available on the hub ☐ Model sha ☐ Flagged

☐ Show private/deleted models ☐ Show flagged models

Model types

☐ 🟢 pretrained ☑ 🔶 fine-tuned ☑ ⭕ instruction-tuned ☑ 🟦 RL-tuned ☑ ?

Precision

☑ float16 ☑ bfloat16 ☑ 8bit ☑ 4bit ☑ GPTQ ☑ ?

Model sizes (in billions of parameters)

☑ ? ☑ ~1.5 ☑ ~3 ☑ ~7 ☑ ~13 ☑ ~35 ☑ ~60 ☑ 70+

Model	Average ⬆️	ARC	HellaSwag	MMLU	TruthfulQA
VAGOsolutions/SauerkrautLM-SOLAR-Instruct	74.21	70.82	88.63	66.2	71.95
upstage/SOLAR-10.7B-Instruct-v1.0	74.2	71.08	88.16	66.21	71.43
fblgit/UNA-SOLAR-10.7B-Instruct-v1.0	74.2	70.56	88.18	66.08	72.05
fblgit/UNA-SOLAR-10.7B-Instruct-v1.0	74.07	70.73	88.32	66.1	72.52
rishiraj/meow	73.94	70.48	88.08	66.25	70.49
sequelbox/SunsetBoulevard	73.78	71.33	90.96	71.21	70.29
ValiantLabs/ShiningValiant	73.78	71.33	90.96	71.21	70.29
SUSTech/SUS-Chat-34B	73.22	66.3	83.91	76.41	57.04

Test base/aligned LLMs on a wide range of reasoning tasks.
(Usually with few-shot ICL examples)

Not in conversation formats and many tasks are less natural.

Evaluation of LLM Alignment

- Human Votes

⚔️ Chatbot Arena ⚔️ : Benchmarking LLMs in the Wild

| Blog | GitHub | Paper | Dataset | Twitter | Discord |

📜 Rules

- Ask any question to two anonymous models (e.g., ChatGPT, Claude, Llama) and vote for the better one!
- You can continue chatting until you identify a winner.
- Vote won't be counted if model identity is revealed during conversation.

🏆 Arena Elo Leaderboard

We use **100K** human votes to compile an Elo-based LLM leaderboard. Find out who is the 🥇 LLM Champion!

👇 Chat now!

🔍 Expand to see 20+ Arena players

Model A

tell me a joke about university of washington

Why did the University of Washington student bring a ladder to school?

Because they heard it was a "higher" education!

Model B

tell me a joke about university of washington

Why don't University of Washington students drink tea?

Because they can't handle the proper-tea!

👈 A is better | 👉 B is better | 🤝 Tie | 👎 Both are bad

Model	⭐ Arena Elo rating
GPT-4-Turbo	1243
GPT-4-0314	1192
GPT-4-0613	1158
Claude-1	1149
Claude-2.0	1131
Mixtral-8x7b-Instruct-v0.1	1121
Claude-2.1	1117
GPT-3.5-Turbo-0613	1117
Gemini Pro	1111

Elo Rating for Ranking LLMs

Win-rate Matrix

Model B (columns) / Model A (rows)

Model A \ Model B	vicuna-13b	koala-13b	alpaca-13b	oasst-pythia-12b	chatglm-6b	fastchat-t5-3b	dolly-v2-12b	stablelm-tuned-alpha-7b	llama-13b
vicuna-13b	0.00	0.68	0.77	0.77	0.82	0.94	0.89	0.78	0.85
koala-13b	0.32	0.00	0.65	0.66	0.74	0.75	0.77	0.84	0.81
alpaca-13b	0.23	0.35	0.00	0.45	0.51	0.78	0.75	0.71	0.75
oasst-pythia-12b	0.23	0.34	0.55	0.00	0.58	0.50	0.61	0.65	0.76
chatglm-6b	0.18	0.26	0.49	0.42	0.00	0.36	0.57	0.55	0.71
fastchat-t5-3b	0.06	0.25	0.22	0.50	0.64	0.00	0.50	0.60	0.47
dolly-v2-12b	0.11	0.23	0.25	0.39	0.43	0.50	0.00	0.51	0.62
stablelm-tuned-alpha-7b	0.22	0.16	0.29	0.35	0.45	0.40	0.49	0.00	0.62
llama-13b	0.15	0.19	0.25	0.24	0.29	0.53	0.38	0.38	0.00

Evaluation of LLM Alignment

- GPTs as Judge

```
<|im_start|>system
You are a helpful assistant, that ranks models by the quality of their answers.
<|im_end|>
<|im_start|>user
I want you to create a leaderboard of different of large-language models. To do so, I
will give you the instructions (prompts) given to the models, and the responses of
two models. Please rank the models based on which responses would be preferred by
humans. All inputs and outputs should be python dictionaries.

Here is the prompt:
{
    "instruction": """{instruction}""",
}

Here are the outputs of the models:
[
    {
        "model": "model_1",
        "answer": """{output_1}"""
    },
    {
        "model": "model_2",
        "answer": """{output_2}"""
    }
]

Now please rank the models by the quality of their answers, so that the model with
rank 1 has the best output. Then return a list of the model names and ranks, i.e.,
produce the following output:
[
    {'model': <model-name>, 'rank': <model-rank>},
    {'model': <model-name>, 'rank': <model-rank>}
]

Your response must be a valid Python dictionary and should contain nothing else
because we will directly execute it in Python. Please provide the ranking that the
majority of humans would give.
<|im_end|>
```

AlpacaEval Leaderboard

An Automatic Evaluator for Instruction-following Language Models

Caution: GPT-4 may favor models with longer outputs and/or those that were fine-tuned on GPT-4 outputs.

Evaluator: GPT-4 Claude   Filter: Community Verified Minimal

Model Name	Win Rate	Length
GPT-4 Turbo	97.70%	2049
XwinLM 70b V0.1	95.57%	1775
PairRM+Tulu 2+DPO 70B (best-of-16)	95.40%	1607
GPT-4	95.28%	1365
Tulu 2+DPO 70B	95.03%	1418
Yi 34B Chat	94.08%	2123
PairRM+Zephyr 7B Beta (best-of-16)	93.41%	1487
LLaMA2 Chat 70B	92.66%	1790
UltraLM 13B V2.0 (best-of-16)	92.30%	1720
XwinLM 13b V0.1	91.76%	1894
UltraLM 13B (best-of-16)	91.54%	1980
Claude 2	91.36%	1069
PairRM+Tulu 2+DPO 13B (best-of-16)	91.06%	1454

Win Rates (as to text-davinci-003)

Evaluation of LLM Alignment

- GPTs as Judge

MT-Bench: Scoring-based Evaluation of LLMs

User

Which word does not belong with the others? tyre, steering wheel, car, engine

Assistant A

The word that does not belong with the others is "tyre." The other three words - "steering wheel," "car," and "engine" - are all parts of a car, while "tyre" is a word that refers to a rubber tire.

I hope this helps! Let me know if you have any other questions.

User's follow-up question

Could you replace it with a word that belongs with the others?

Assistant A

Sure! Instead of "tyre," a word that belongs with the others could be "transmission." So, the revised list of words that belong together would be:

- Steering wheel
- Car
- Engine
- Transmission

I hope this helps! Let me know if you have any other questions.

Prompting GPT-4 →

Model Judgment (first turn)

Game 1. A: Llama-2-70b-chat, **Score:** 2

Judgment: The assistant's answer is incorrect. The word that does not belong with the others is "car," not "tyre." The words "tyre," "steering wheel," and "engine" are all parts of a car, while "car" is the whole vehicle. The assistant seems to have misunderstood the question and provided an incorrect response.

Corrected Assistant's Answer: The word that does not belong with the others is "car." The other three words - "tyre," "steering wheel," and "engine" - are all parts of a car, while "car" is the whole vehicle. I hope this helps! Let me know if you have any other questions.

Rating: [[2]]

Model Judgment (second turn)

Game 1. A: Llama-2-70b-chat, **Score:** 2

Judgment: The assistant's response to the second question is incorrect. The assistant was supposed to replace the word "car" with a word that belongs with the others, not "tyre". The assistant's response to the first question was also incorrect, as "tyre" is indeed a part of a car, not separate from it. The correct response to the second question should have been to replace "car" with another part of a car, such as "brake pedal" as in the reference answer. The assistant's revised list should have been "tyre, steering wheel, engine, brake pedal" or another part of a car instead of "brake pedal".

Corrected Assistant's Answer: Sure! Instead of "car," a word that belongs with the others could be "brake pedal." So, the revised list of words that belong together would be:

- Tyre
- Steering wheel
- Engine
- Brake pedal

I hope this helps! Let me know if you have any other questions.

Rating: [[2]]

RLHF/RLAIF

Pre-training → Instruction Tuning → RLHF/RLAIF

In-Context Learning

Alignment:

- Instruction following
- Preference tuning
- Safety
- Etc.

Limitations of Instruction Tuning

- **Why do we need RLHF?**

LM objective != human preferences

Limitations of Instruction Tuning

LM objective != human preferences

- **Why do we need RLHF?**
- (Open-ended) generation:
  - What makes one output better than the other? **-> hard to define**
- What types of LM errors should be weighted more?

Limitations of Instruction Tuning

- **Why do we need RLHF?**
- (Open-ended) generation: How do you capture all of the following and more in a loss function:
  - What is a *helpful* output?
  - What is a *polite* output?
  - What is a *funny* output?
  - What is a *safe* output?

LM objective != human preferences

RLHF!

Fine-Tuning Language Models from Human Preferences

Daniel M. Ziegler* **Nisan Stiennon*** **Jeffrey Wu** **Tom B. Brown**
Alec Radford **Dario Amodei** **Paul Christiano** **Geoffrey Irving**
OpenAI
{dmz,nisan,jeffwu,tom,alec,damodei,paul,irving}@openai.com

arxiv in Sep 2019
NeurIPS 2020

Learning to summarize from human feedback

Nisan Stiennon* **Long Ouyang*** **Jeff Wu*** **Daniel M. Ziegler*** **Ryan Lowe***

Chelsea Voss* **Alec Radford** **Dario Amodei** **Paul Christiano***

OpenAI

arxiv in Sep 2020
NeurIPS 2020

"Learning to Summarize with Human Feedback"

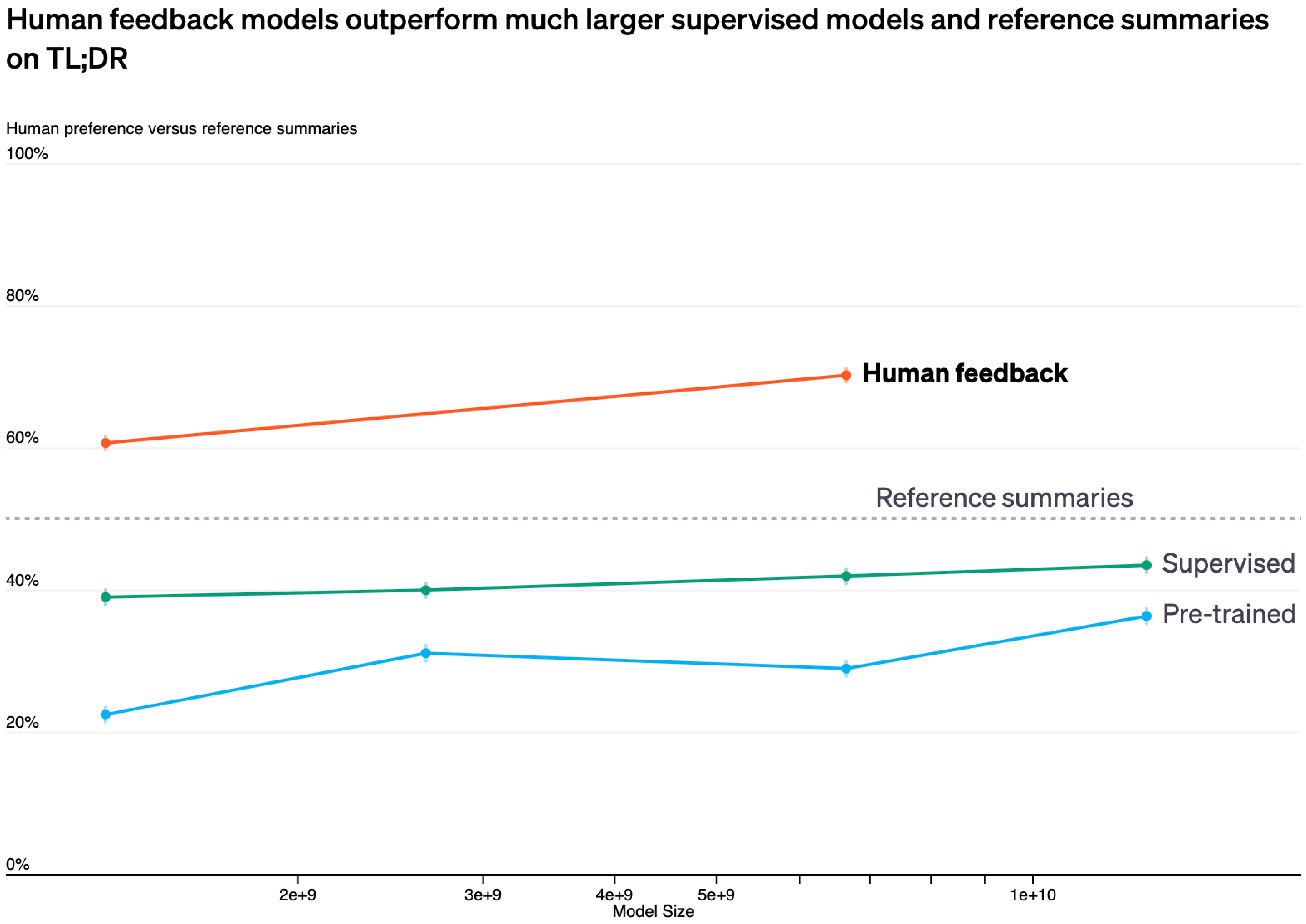
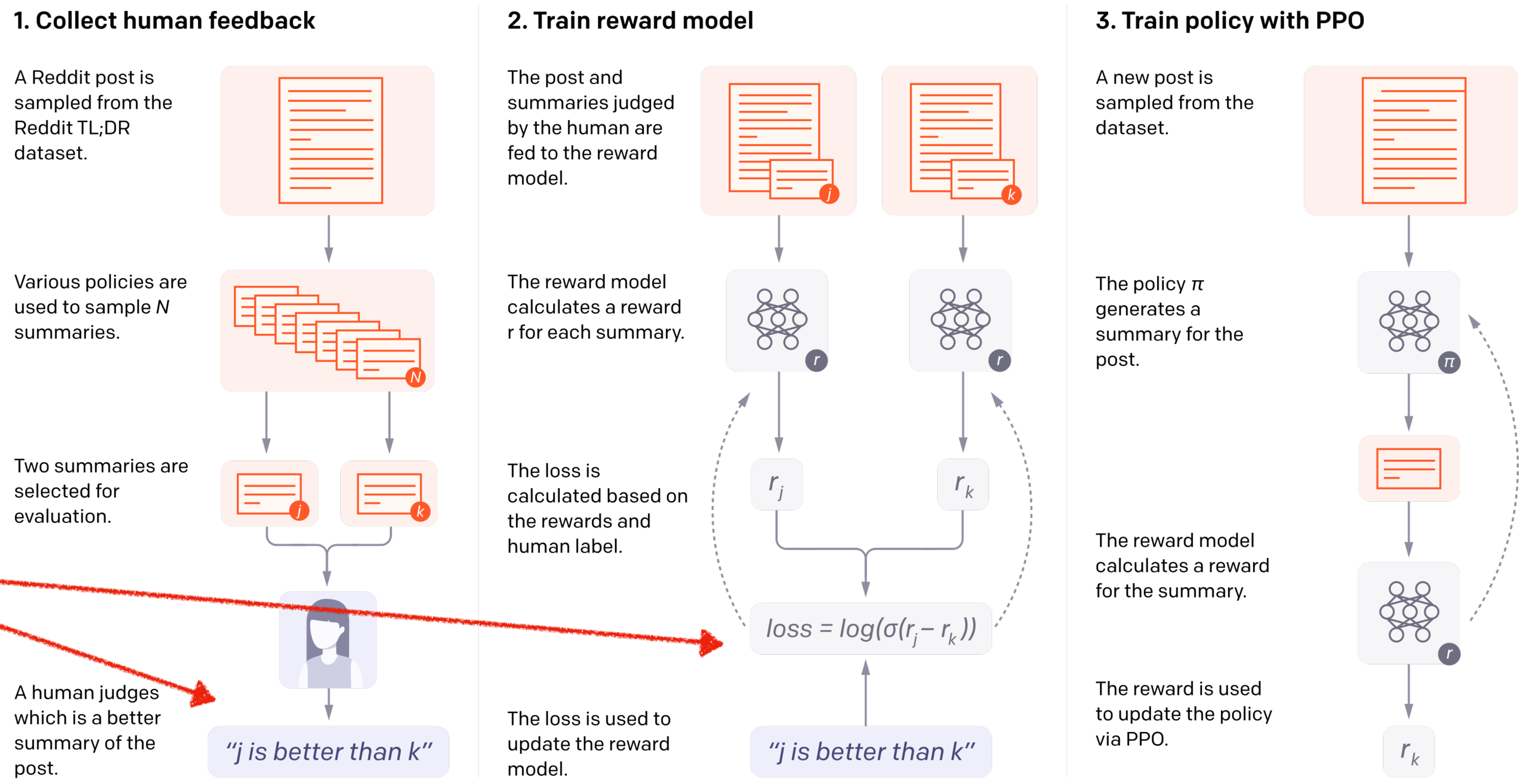


Figure 1: The performance of various training procedures for different model sizes. Model performance is measured by how often summaries from that model are preferred to the human-written reference summaries. Our pre-trained models are early versions of GPT-3, our supervised baselines were fine-tuned to predict 117K human-written TL;DRs, and our human feedback models are additionally fine-tuned on a dataset of about 65K summary comparisons.

https://openai.com/research/learning-to-summarize-with-human-feedback

“Learning to Summarize with Human Feedback”

RL methods don’t always assume “preference-based” (j is better than k) human feedback and reward model, but that’s what’s common with current “RLHF” approaches



https://openai.com/research/learning-to-summarize-with-human-feedback

"Fine-Tuning Language Models with Human Feedback"

$$\text{loss}(r) = \mathbb{E}_{(x,\{y_i\}_i,b)\sim S}\left[\log \frac{e^{r(x,y_b)}}{\sum_i e^{r(x,y_i)}}\right] \quad (1)$$

$$R(x,y) = r(x,y) - \beta \log \frac{\pi(y|x)}{\rho(y|x)}. \quad (2)$$

1. Gather samples (x, y_0, y_1, y_2, y_3) via $x \sim \mathcal{D}, y_i \sim \rho(\cdot|x)$. Ask humans to pick the best y_i from each.

2. Initialize r to ρ, using random initialization for the final linear layer of r. Train r on the human samples using loss (1).

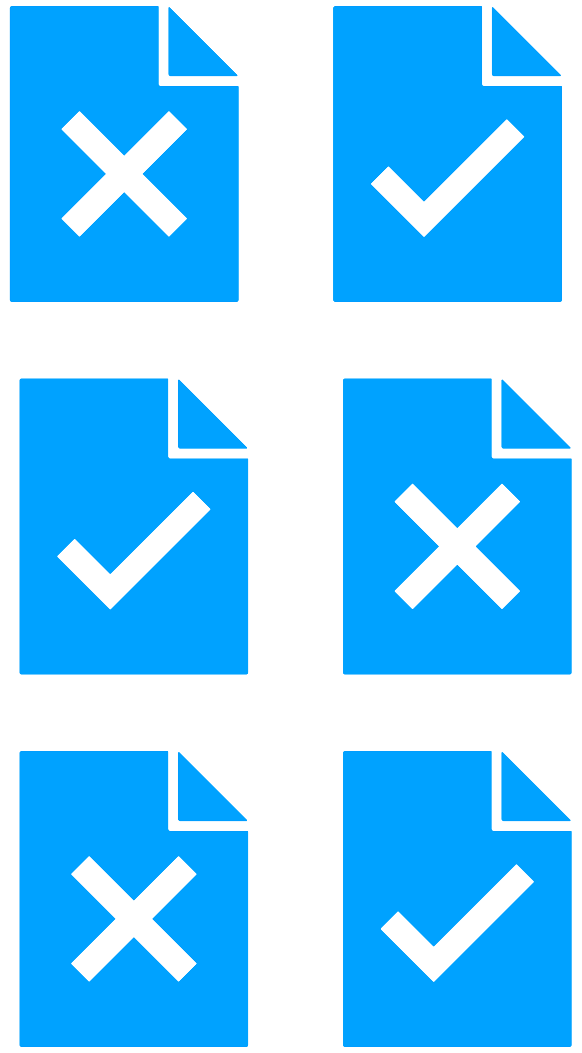
3. Train π via Proximal Policy Optimization (PPO, Schulman et al. (2017)) with reward R from (2) on $x \sim \mathcal{D}$.

4. In the online data collection case, continue to collect additional samples, and periodically retrain the reward model r. This is described in section 2.3.

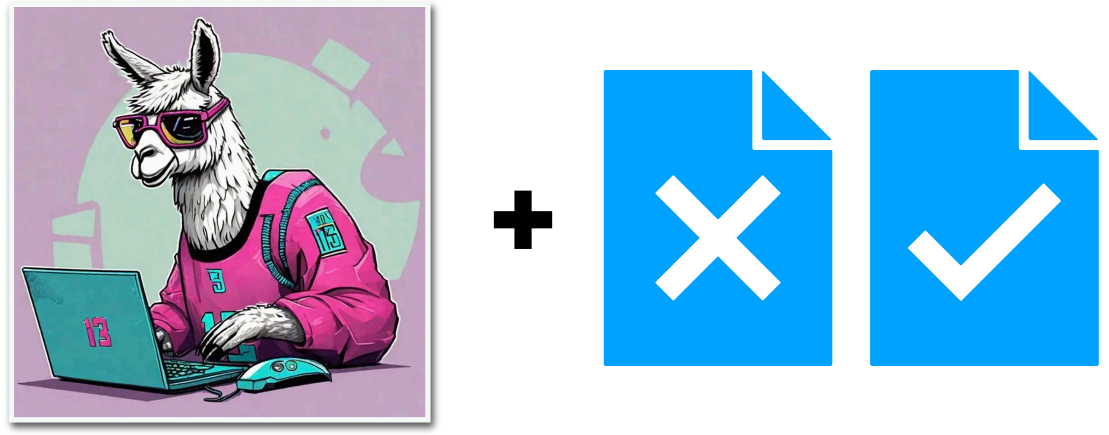
The general RLHF pipeline

1 Instruction-tuned Model

2 Collect Comparison Data



3 Train Reward Model on Comparison Data



4 Use RL to Optimize a Policy with the Reward Model

Human Preferences

A set of sampled completions for a prompt.

Ranking of the samples.

Prompt

Sample A

Sample B

Sample C

C → A → B

Human Preferences

A set of sampled completions
for a prompt.

Prompt

Sample A

Sample B

Sample C

Triples

$$D = \{x^i, y_w^i, y_l^i\}$$

Prompt

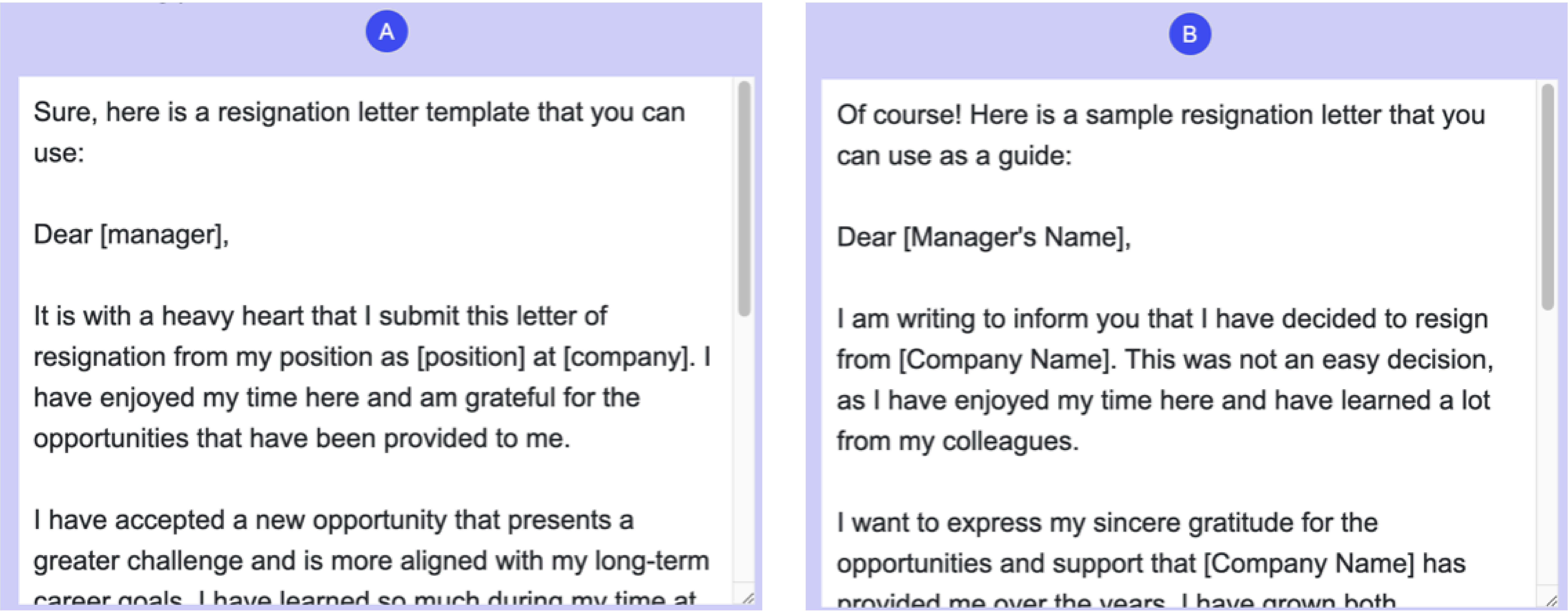
Preferred
Response

Dispreferred
Response

Example: Annotation

resignation letter to my current employer, while leaving on good terms and expressing gratitude f

two responses from the chatbot. (Please scroll down on the content to see the entire response if i



- Annotator needs to choose whether they prefer A or B.

Pairwise Comparison

Why do pairwise comparison and not rate outputs directly?

n engaging travel blog post about a recent trip to Hawaii, highlighting cultural experiences and mu

A

Have you ever imagined standing on a black sand beach, watching the waves of the Pacific Ocean crash against the shore? Or strolling down a narrow street, taking in the vibrant colors and aromas of local cuisine? If so, then Hawaii is the perfect destination for you. I recently had the pleasure of visiting this stunning archipelago in the middle of the Pacific Ocean, and I am still daydreaming about my amazing adventures there. From the Hawaiian culture to the natural wonders, every moment was full of wonder and excitement.

How would you rate this output?

- Hard to be consistent among different annotators!
- It’s more reliable (Phelps et al., 2015; Clark et al., 2018)
- Can be used with the Bradley-Terry (1952) model

From Preference Data to Bradley-Terry Model

$$D = \{x^i, y_w^i, y_l^i\}$$

Prompt

Preferred Response

Dispreferred Response

Reward for preferred response

Reward for dispreferred response

$$p(y_w > y_l|x) = \sigma(\underline{r(x, y_w)} - \underline{r(x, y_l)})$$

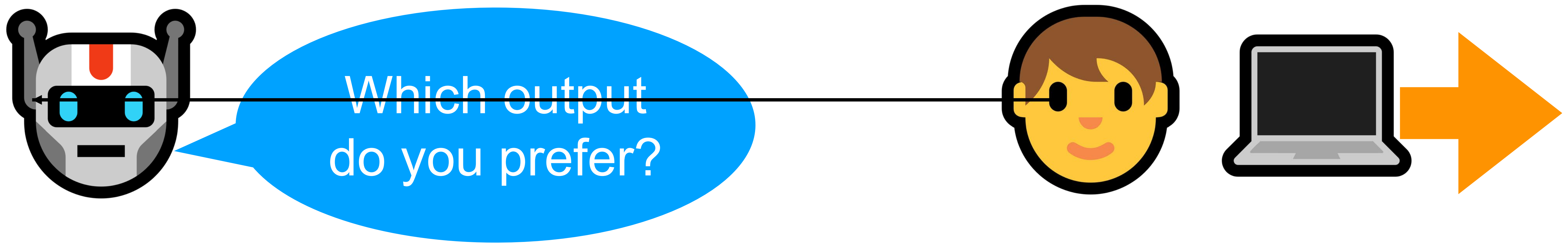
Logistic function; which is equivalent to using softmax:

$$\frac{1}{1+e^{-x}}$$

$$p(y_w > y_l|x) = \frac{\exp(r(x, y_w))}{\exp(r(x, y_w)) + \exp(r(x, y_l))}$$

But..

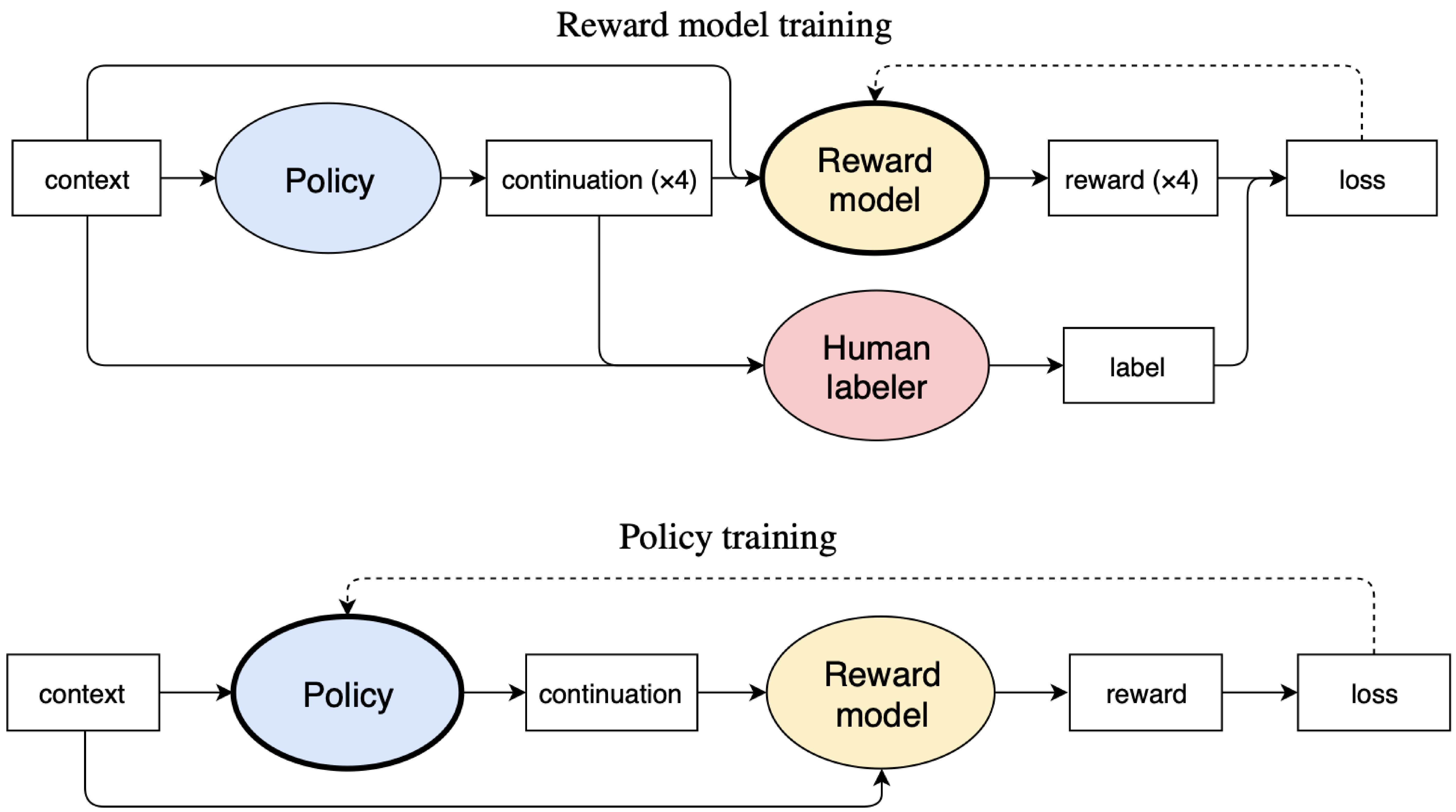
- How do we get feedback for the reward while training our RL model?



Having a human in the loop is very expensive!

But..

- How do we get feedback for the reward while training our RL model?



Instead: train a Reward Model (RM) on preference data to predict preferences!

Ziegler et al., 2019 "Fine-Tuning Language Models from Human Preferences"

Reward Model

$$p(y_w > y_l|x) = \frac{\exp(r(x, y_w))}{\exp(r(x, y_w)) + \exp(r(x, y_l))}$$

equivalent to

- Train on preference data.

Bradley-Terry Model

- Minimizing negative log likelihood.

$$\mathcal{L}_R(\phi, D) = -\mathbb{E}_{(x,y_w,y_l)\sim D}[\log \underline{\sigma(r(x, y_w) - r(x, y_l))}]$$

- Train an LLM with an additional layer to minimize the neg. log likelihood

Evaluating Reward Models

- Accuracy of predicting human preferences.

Preference Datasets

eward Models

Table 2: Reward modeling accuracy (%) results. We compare our UltraRM with baseline open-source reward models. LLaMA2 results are taken from Touvron et al. (2023b). The highest results are in **bold** and the second highest scores are underlined.

Model	Backbone Model	Open?	Anthropic Helpful	OpenAI WebGPT	OpenAI Summ.	Stanford SHP	Avg.
Moss	LLaMA-7B	✓	61.3	54.6	58.1	54.6	57.2
Ziya	LLaMA-7B	✓	61.4	57.0	61.8	57.0	59.3
OASST	DeBERTa-v3-large	✓	67.6	-	72.1	53.9	-
SteamSHP	FLAN-T5-XL	✓	55.4	51.6	62.6	51.6	55.3
LLaMA2 Helpfulness	LLaMA2-70B	✗	**72.0**	-	**75.5**	**80.0**	-
UltraRM-UF	LLaMA2-13B	✓	66.7	65.1	66.8	68.4	66.8
UltraRM-Overall	LLaMA2-13B	✓	<u>71.0</u>	62.0	73.0	73.6	<u>69.9</u>
UltraRM	LLaMA2-13B	✓	<u>71.0</u>	**65.2**	<u>74.0</u>	<u>73.7</u>	**71.0**

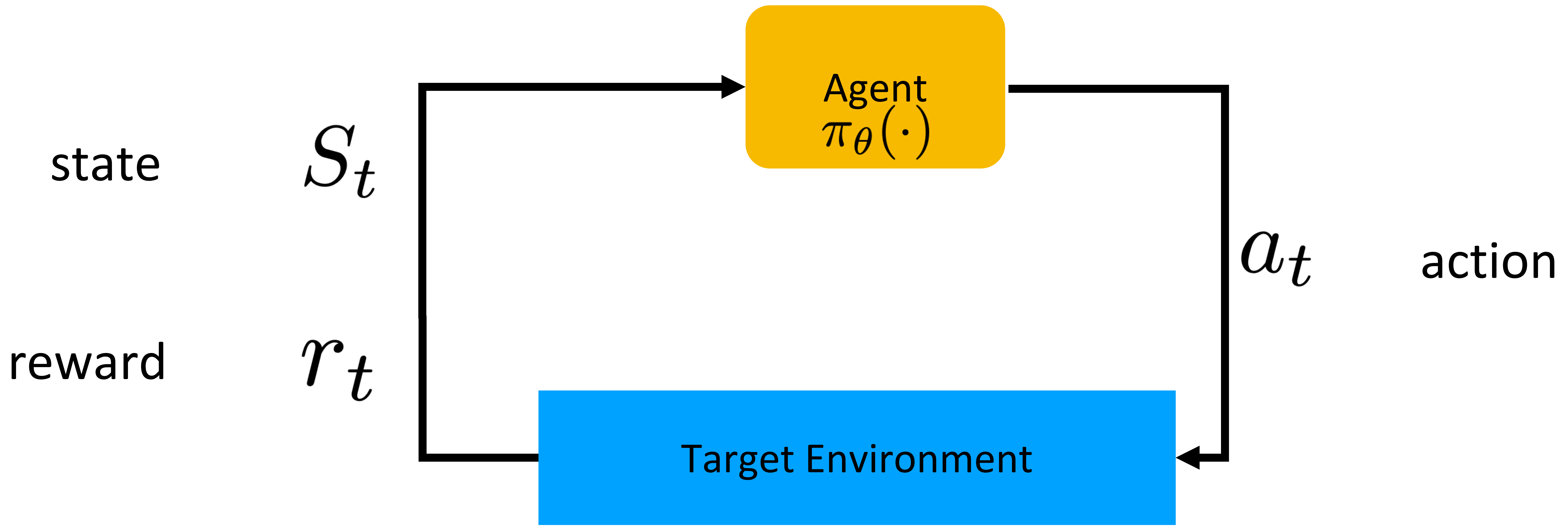
Cui et al., ArXiV 2023 "UltraFeedback: Boosting Language Models with High-quality Feedback"

Fun Facts about Reward Models

- Trained for 1 epoch (to avoid overfitting)!
- Evaluation often only has 65% - 75% agreement

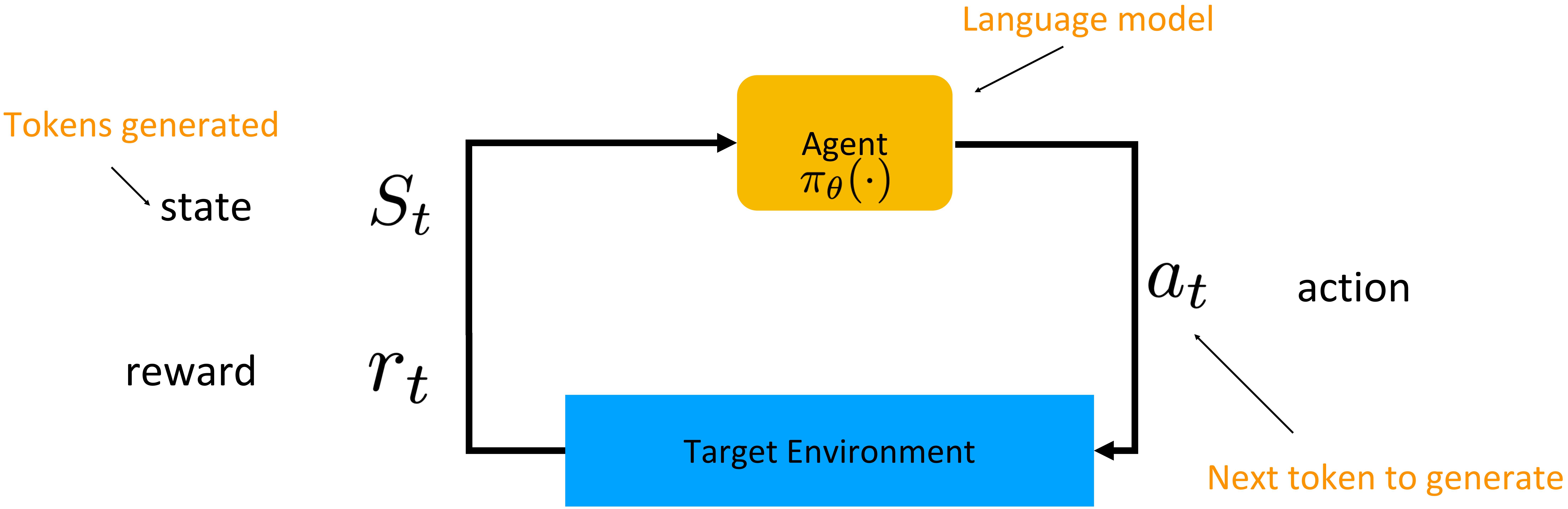
Lambert et al., 2023

Reinforcement Learning Basics



$a_t \sim \pi_\theta(S_t)$: policy

RL in the Context of Language Models…



$a_t \sim \pi_\theta(S_t)$: policy