



# Build a Large Language Model (From Scratch)

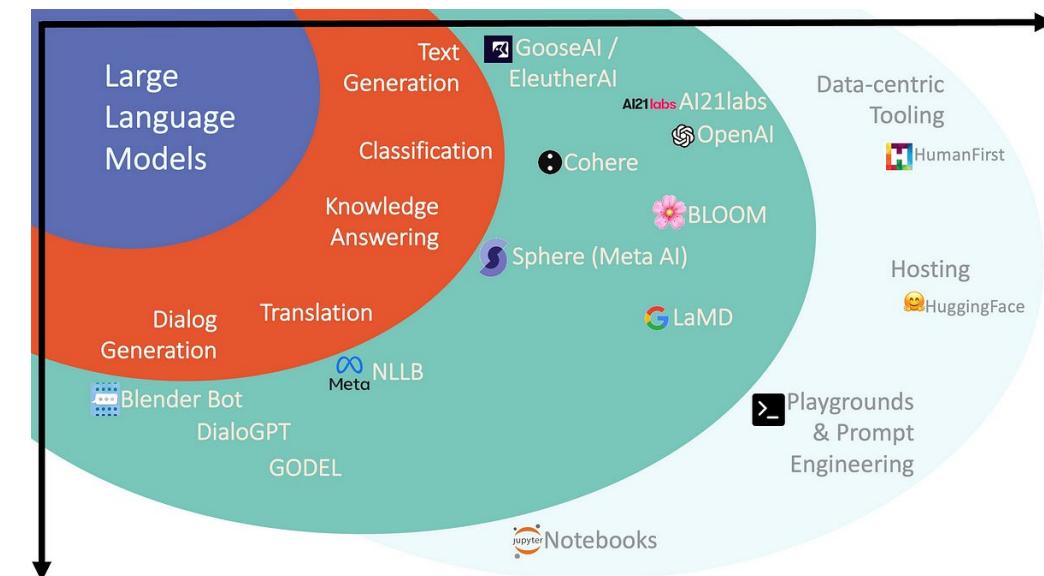
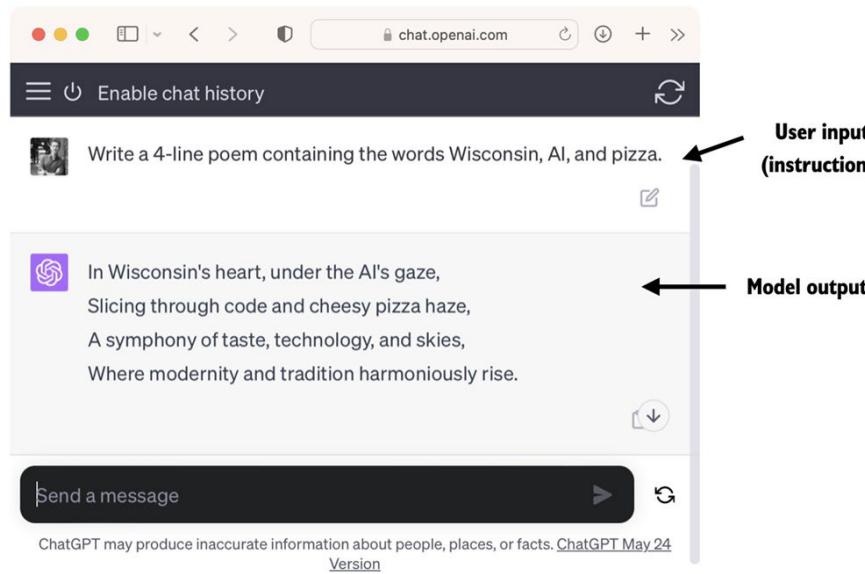
Long-Fei Li

2024.10.11

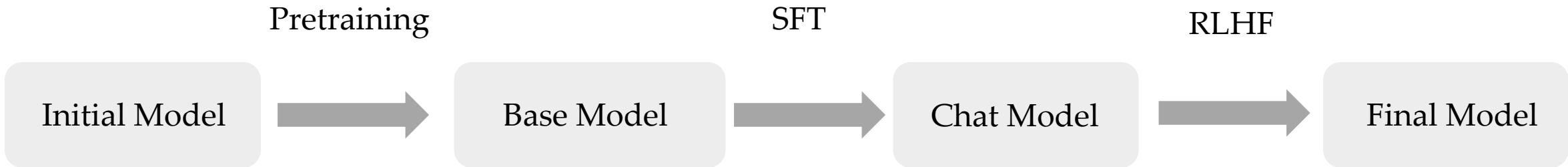
# What is an LLM?

## □ LLM (Large Language Model):

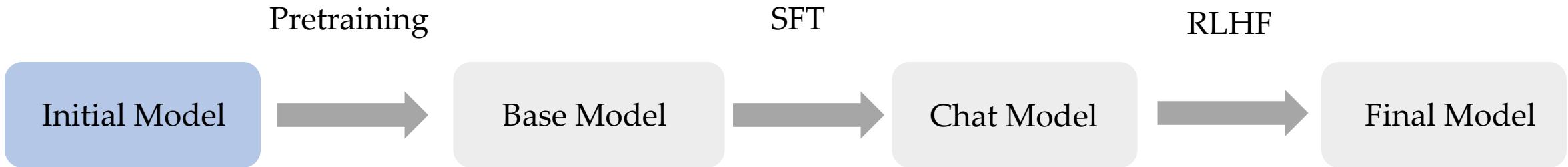
- “Large”: both the model's size and the immense dataset
- “Language Model”: designed to understand, generate, and respond to human-like text



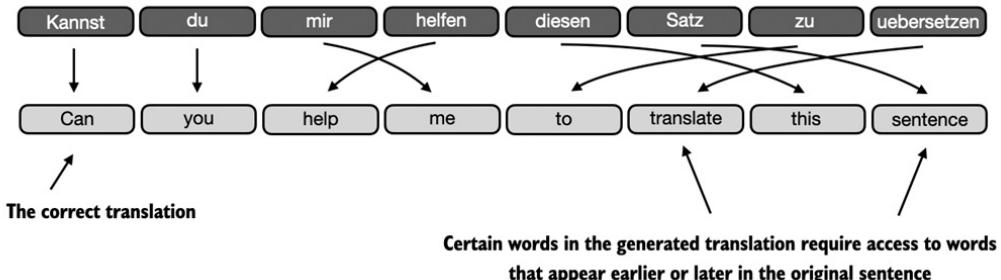
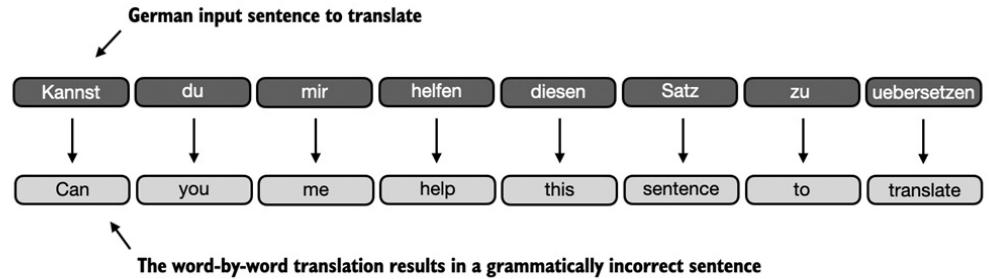
# Building an LLM



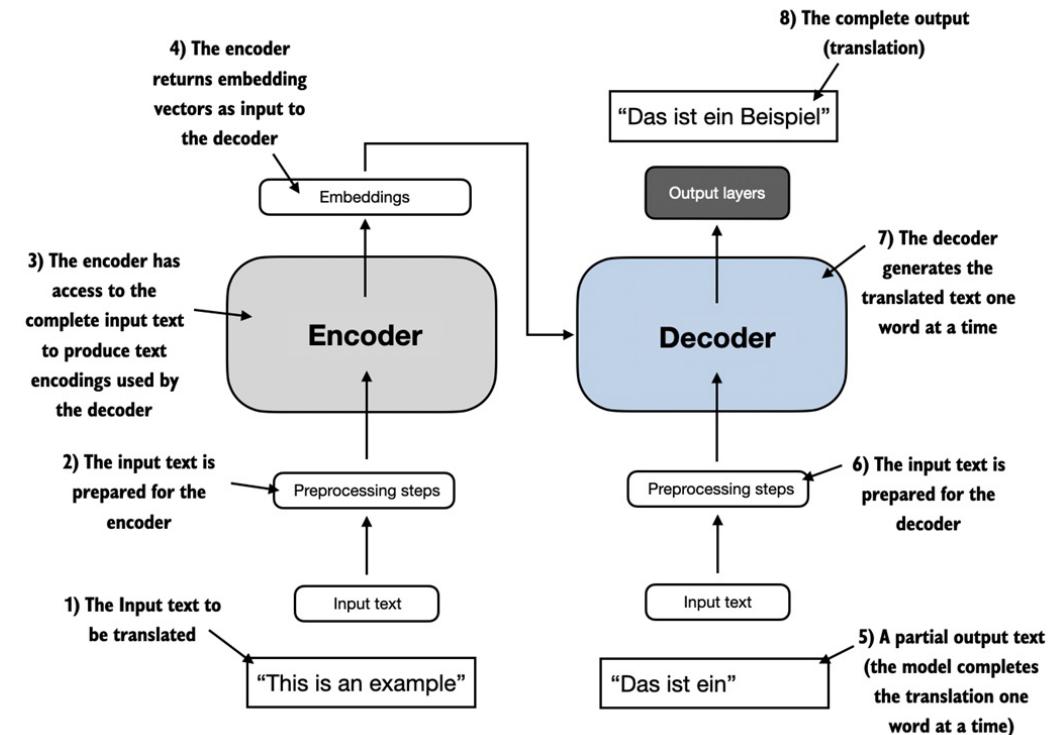
# Building an LLM



# Architecture of LLM



A translate task

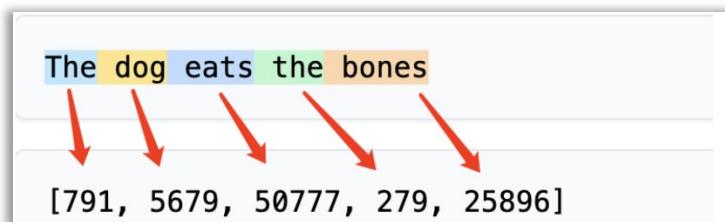


- Encoder: read in and process the entire text
- Decoder: produces the translated text

# Tokenizing text

## Word-based Tokenize

- Large vocabulary size, Out-of-vocabulary (OOV)
- Loss relationship, e.g., dog and dogs, happy and unhappy;



Model:  gpt-3.5-turbo cl100k\_base ▾

**Text:**

Tokens are the basic unit that generative AI models use to compute the length of a text.

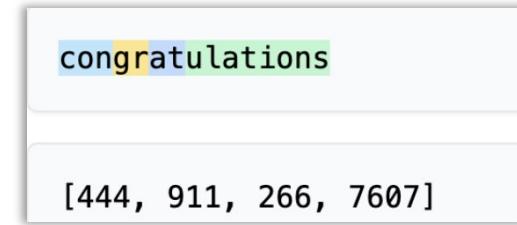
**Tokenized text:**

Tokens are the basic unit that generative AI models use to compute the length of a text.

**Text** **Token IDs**

## Sub-word-based Tokenize

- No OOV
- Keep semantic information



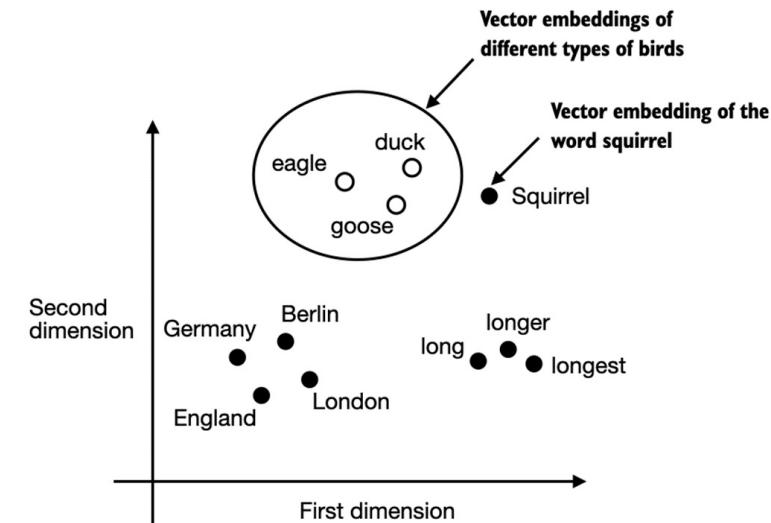
- **Vocab Size:**
  - GPT 2/3 = 50,257
  - Llama 2 (7B) = 32,000
  - Llama 3 (8B) = 128,256

# Word Embedding

- Motivation: convert non-numeric data into a format that NN can process
  - A mapping from discrete objects, such as words, images, or even entire documents, to points in a continuous vector space

"a"	"abbreviations"	"zoology"	"zoom"
1	0	0	0
0	1	0	1
0	0	0	0
.	.	.	.
.	.	.	.
.	.	.	.
0	0	0	0
0	0	1	0
0	0	0	1

One hot representation



Distributed representation

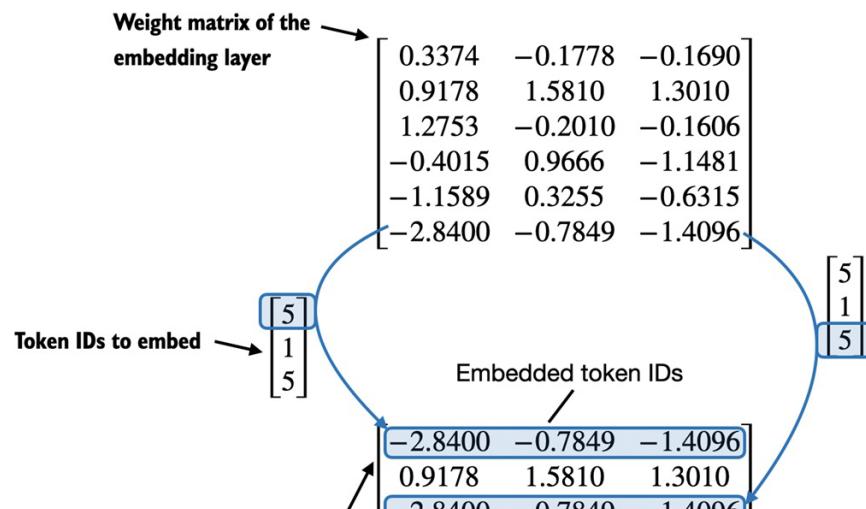
# Word Embedding

- Motivation: convert non-numeric data into a format that NN can process
  - A mapping from discrete objects, such as words, images, or even entire documents, to points in a continuous vector space

$$\begin{array}{c} \begin{array}{ccccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \text{I} & 0 & 0 & 1 & 0 & 0 & 0 \\ \text{love} & 0 & 1 & 0 & 0 & 0 & 0 \\ \text{dog} & 0 & 0 & 0 & 1 & 0 & 0 \end{array} & * & \begin{array}{c} \begin{array}{ccccc} 0 & 0.22 & 1.15 & 0.14 & 3.36 \\ 1 & 0.17 & 2.27 & 2.17 & 4.42 \\ 2 & 1.15 & 1.07 & 7.36 & 2.13 \\ 3 & 0.67 & 0.49 & 0.06 & 2.33 \\ 4 & 0.62 & 9.17 & 0.34 & 9.78 \\ 5 & 0.66 & 0.02 & 3.17 & 0.04 \end{array} \\ 6*4 \end{array} = \begin{array}{c} \begin{array}{cccc} \text{I} & 1.15 & 1.07 & 7.36 & 2.13 \\ \text{love} & 0.17 & 2.27 & 2.17 & 4.42 \\ \text{dog} & 0.67 & 0.49 & 0.06 & 2.33 \end{array} \end{array}$$

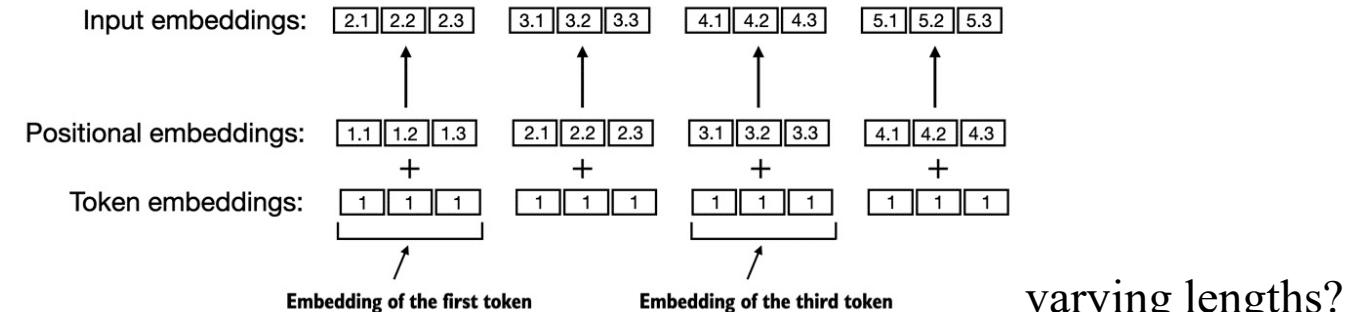
# Positional Embedding

## Position-agnostic

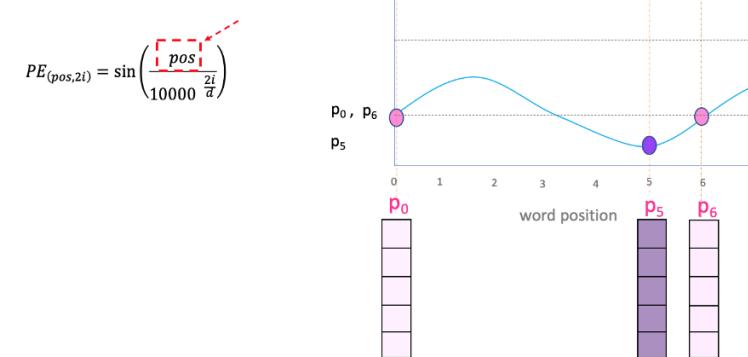


Dog is human's best friend  
Human is dog's best friend

## □ Absolute positional embedding

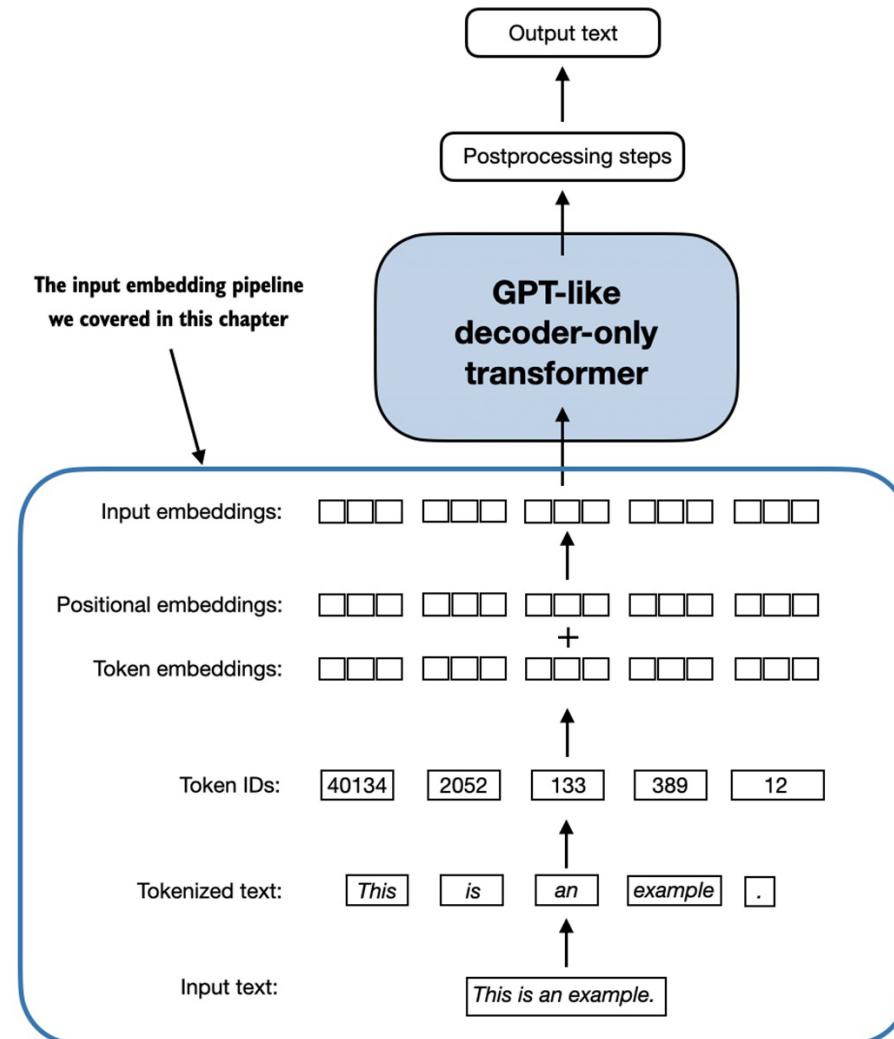


## □ Relative positional embedding



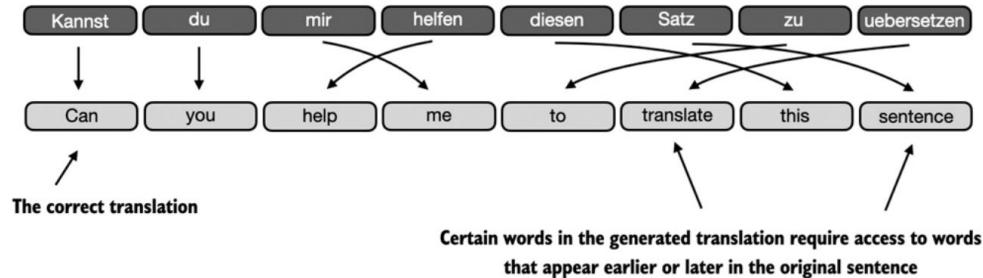
"how far apart"  
rather than  
"at which exact position."

# Overall Pipeline

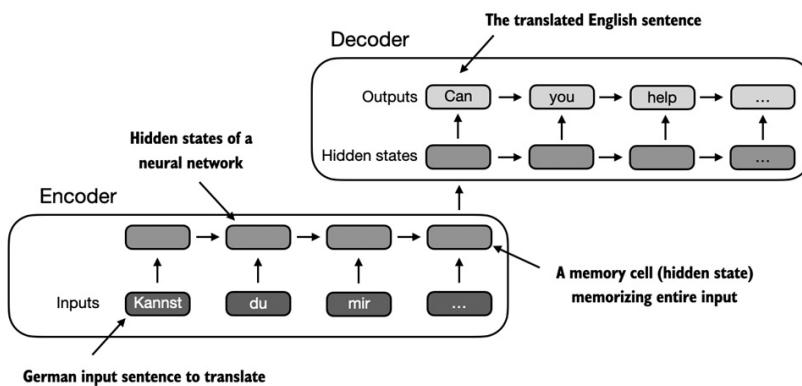


# Attention

A translate task

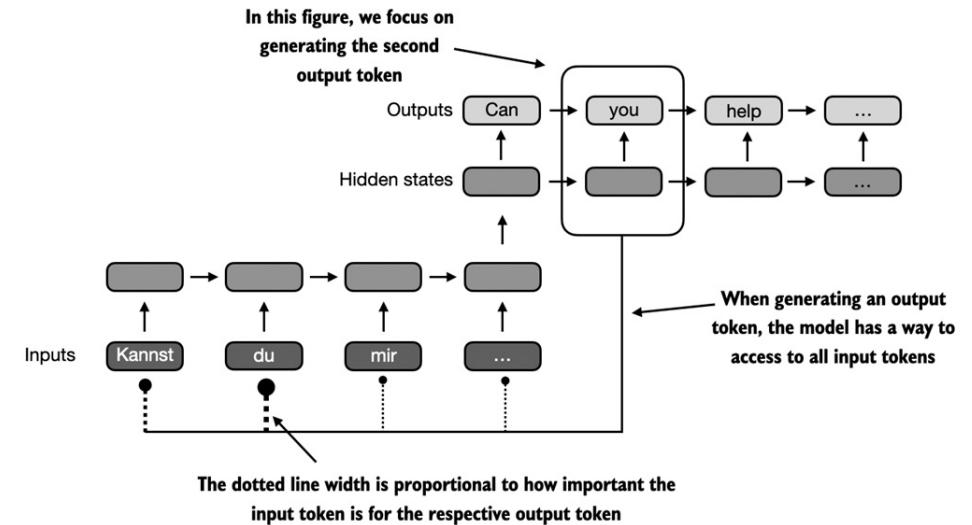


## □ Encoder-decoder RNN

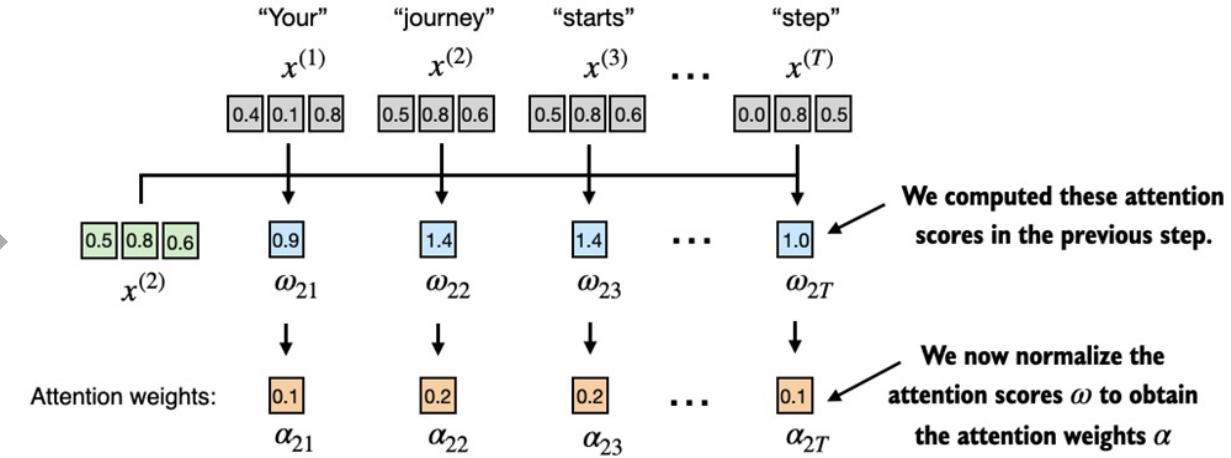
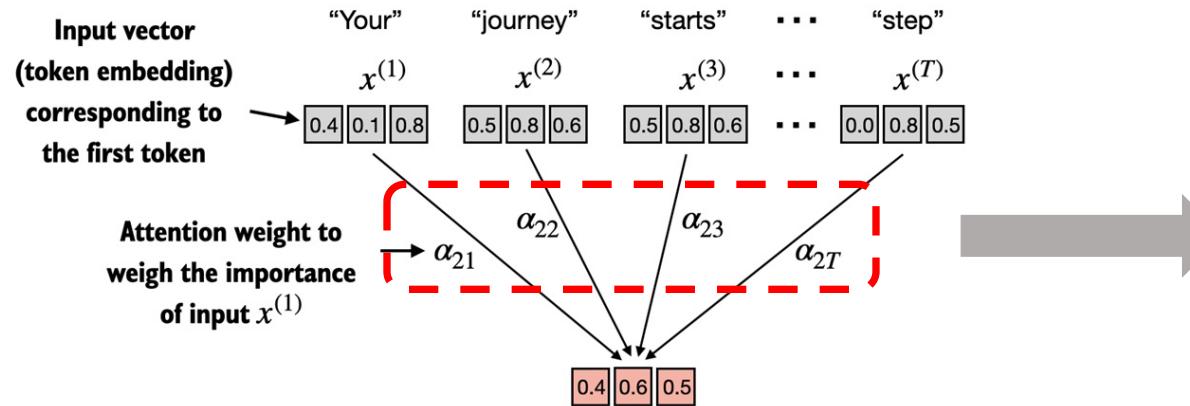


- Limitation: relies **solely on the current hidden state which encapsulates all relevant information**

## □ Bahdanau attention



# Simplified self-attention



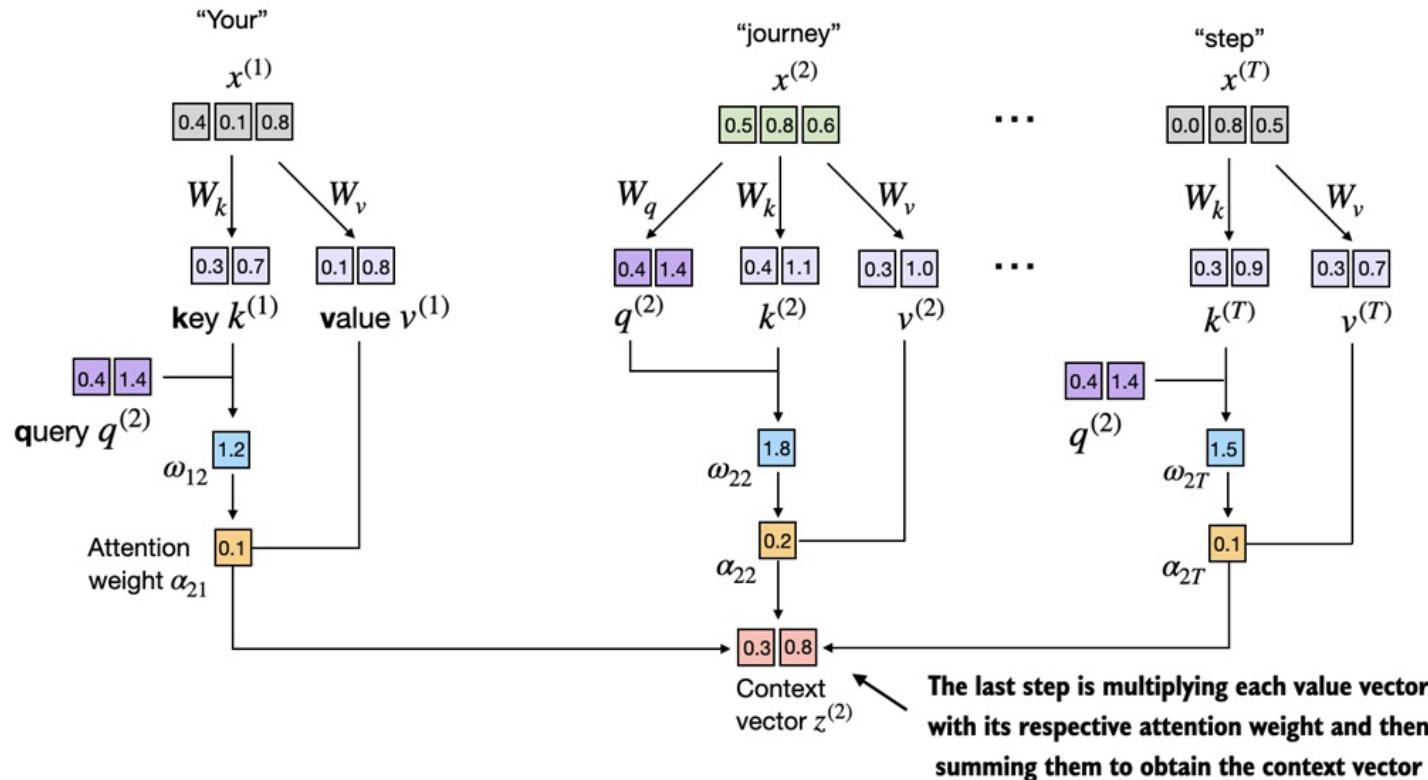
	Your	journey	starts	with	one	step
Your	0.20	0.20	0.19	0.12	0.12	0.14
journey	0.13	0.23	0.23	0.12	0.10	0.15
starts	0.13	0.23	0.23	0.12	0.11	0.15
with	0.14	0.20	0.20	0.14	0.12	0.17
one	0.15	0.19	0.19	0.13	0.18	0.12
step	0.13	0.21	0.21	0.14	0.09	0.18

This row contains the attention weights (normalized attention scores) computed previously

dot products

# Self-attention

- Self-attention (scaled-dot product attention): Add trainable weight matrices

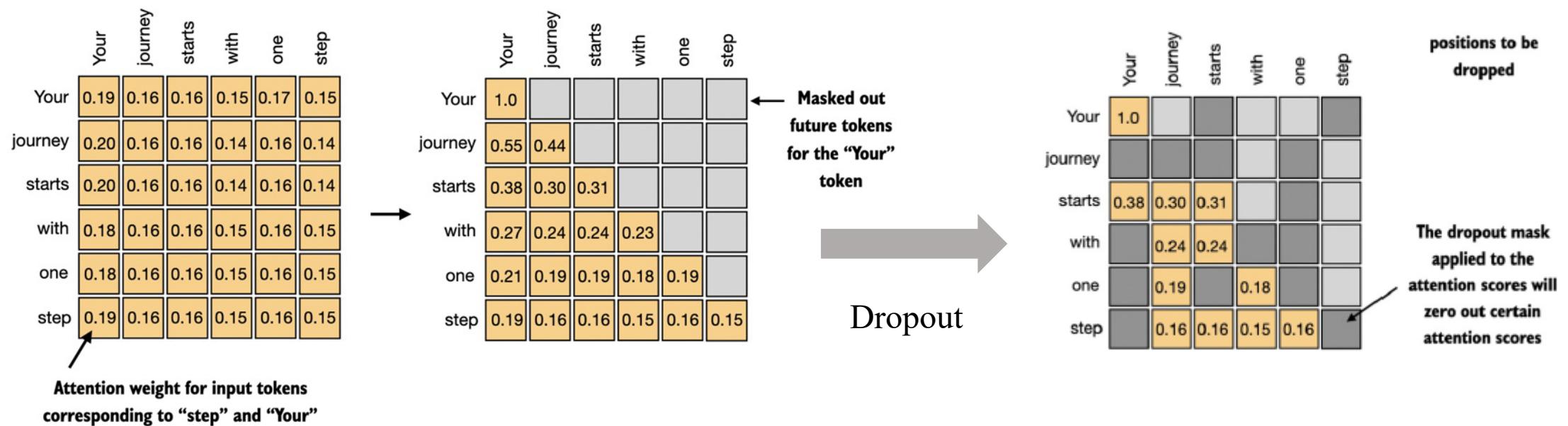


$$\text{Attention value } (Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$

# Causal attention

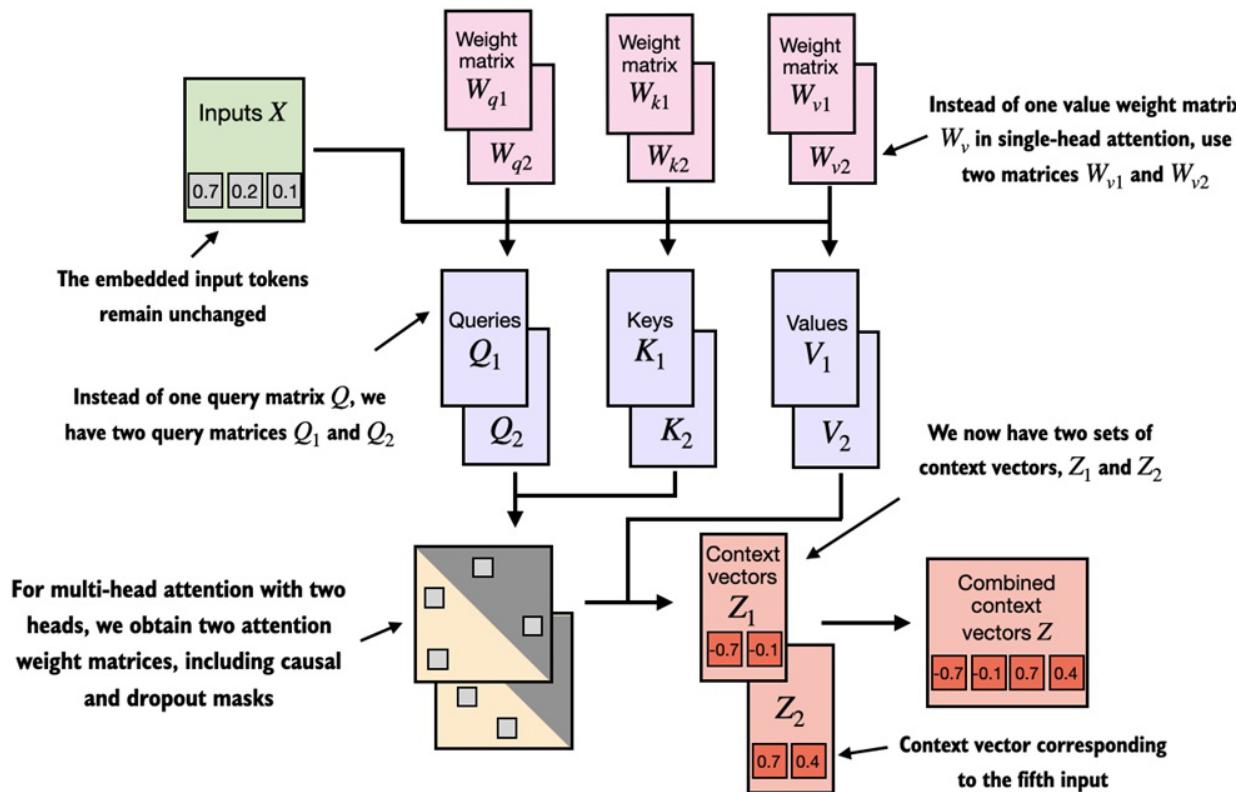
## □ Causal attention (masked attention)

- only consider previous and current inputs



# Multi-head attention

## □ multi-head attention



# Normalization

- Reason: vanishing or exploding gradients
- Normalization: adjust the activations to have a mean of 0 and a variance of 1

$x + y > 5?$

Index	X1	X2	Label
1	0	10	True
2	2	0	False

Index	X1	X2	Label
1	-1	5	True
2	1	-5	False

Batch normalization

Across samples

Index	X1	X2	Label
1	-5	5	True
2	1	-1	False

Layer normalization

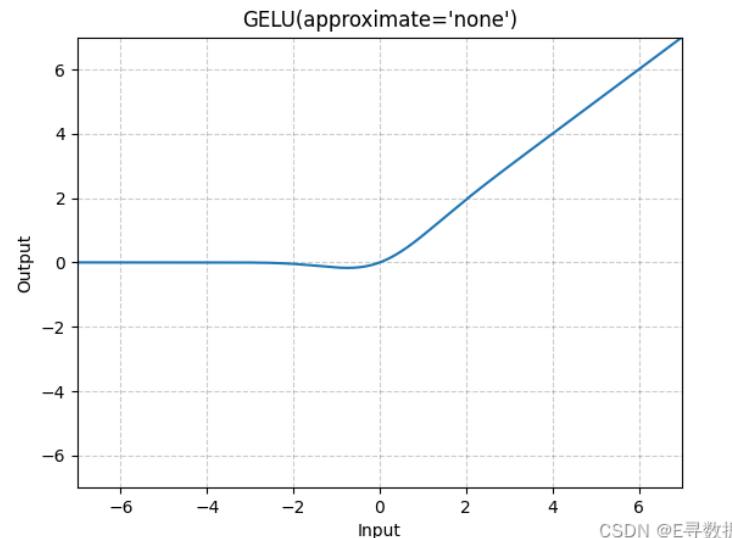
In-sample

# Activations

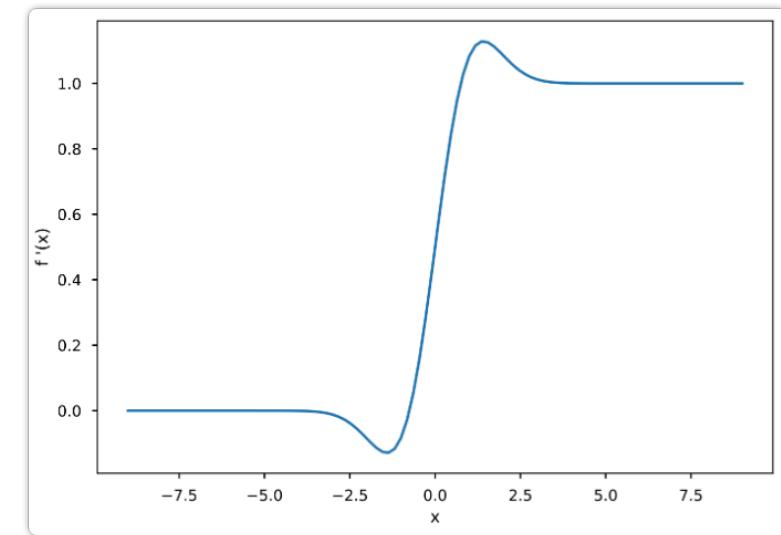
## □ GELU activations: RELU + Dropout

$$\text{GELU}(x) = x \times P(X \leq x) = x \times \phi(x), x \sim N(0, 1)$$

$$\approx 0.5x \left( 1 + \tanh \left( \sqrt{2/\pi} (x + 0.044715x^3) \right) \right)$$

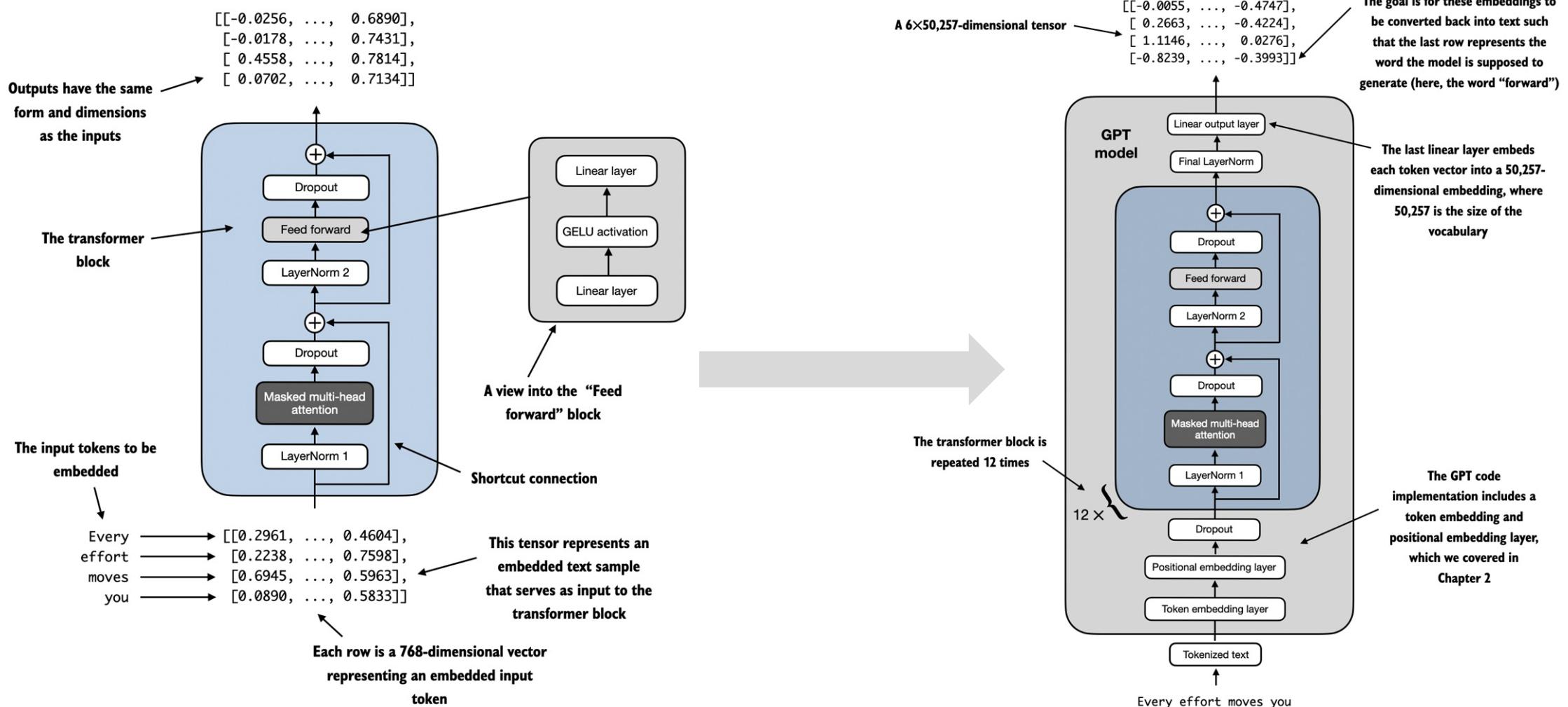


$$f(x)$$



$$f'(x)$$

# LLM architecture



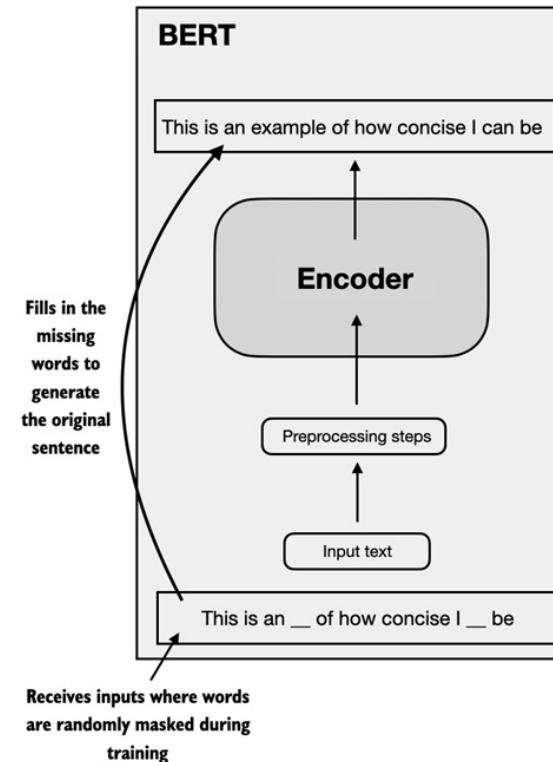
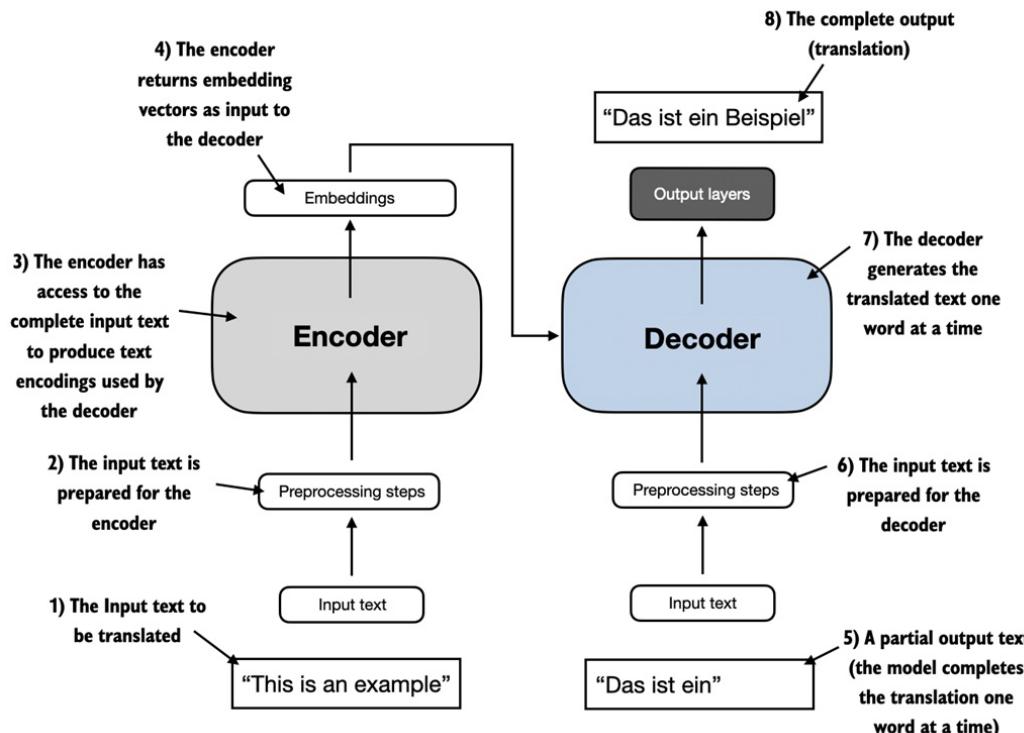
# GPT-2 & 3

Model	Parameters	Blocks	Emb dim	heads
GPT2 small	124 M	12	768	12
GPT2 medium	345 M	24	1024	16
GPT-2 large	762 M	36	1280	20
GPT-2 XL	1542 M (1.5B)	48	1600	25

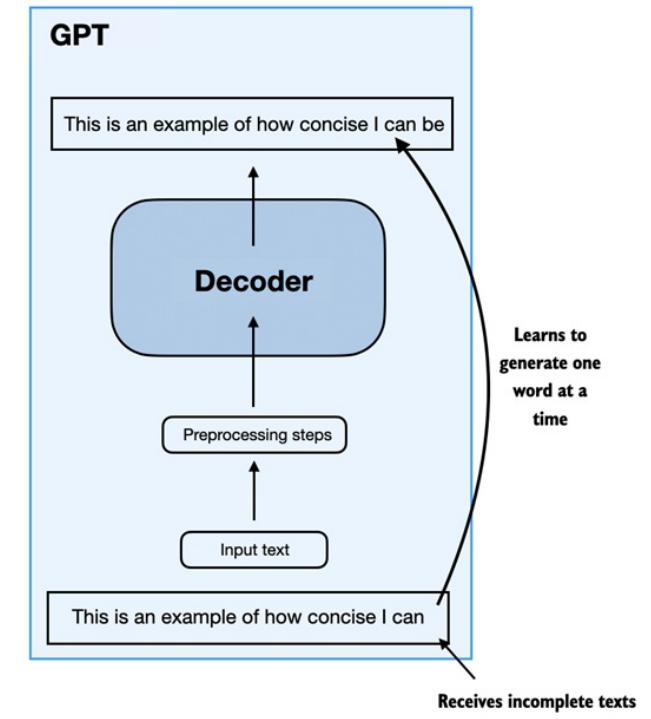
Model	Parameters	Blocks	Emb dim	heads
GPT-3 XL	1.3B	24	2048	24
GPT-3 13B	13B	40	5140	40
GPT-3 175B (GPT3)	175B	96	12288	96

# Architecture of LLM

## Transformer

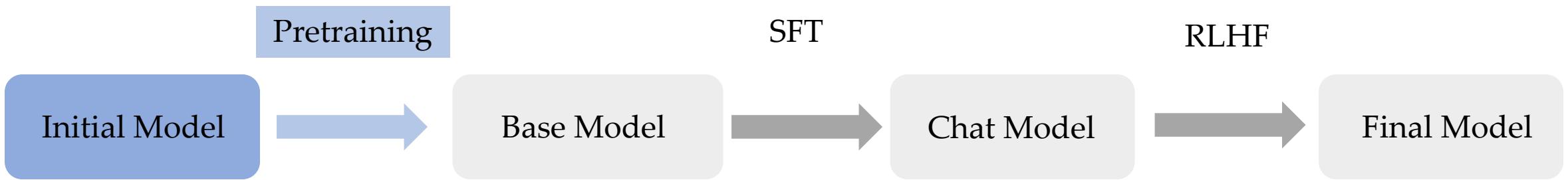


text classification



text completion

# Building an LLM



# Pretraining

## □ Pretrain: next token prediction

Text sample: LLMs learn to predict one word at a time

LLMs learn to predict one word at a time

LLMs learn to predict predict one word at a time

LLMs learn to predict one word at a time

LLMs learn to predict one word at a time

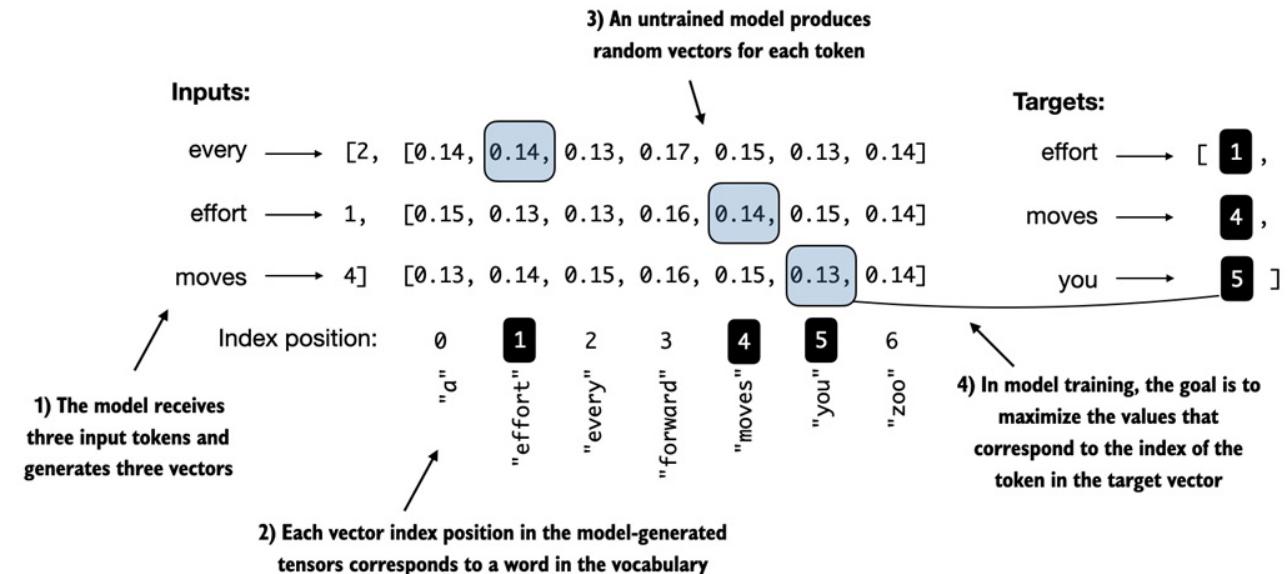
LLMs learn to predict one word at a time

LLMs learn to predict one word at a time

LLMs learn to predict one word at a time

The LLM  
can't access  
words past  
the target

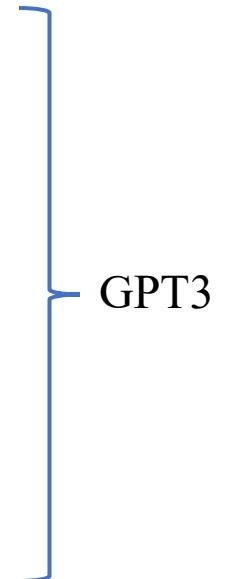
Target to  
predict



$$L_{\text{Pre}}(\mathbf{x}) = - \sum_t \log \pi_\theta(x_t | x_1, \dots, x_{t-1})$$

# Dataset

Dataset Name	Description	# Tokens	Proportion	Storage
CommonCrawl	Web crawl data	410 B	60%	570 GB
WebText2	Web crawl data	19 B	22%	-
Books1 (Gutenberg)	Internet-based book corpus	12 B	8%	-
Books2 (Libgen)	Internet-based book corpus	55 B	8%	-
Wikipedia	High-quality text	3B	3%	
Arxiv	research papers			92 GB
StackExchange	code-related Q&As			78 GB



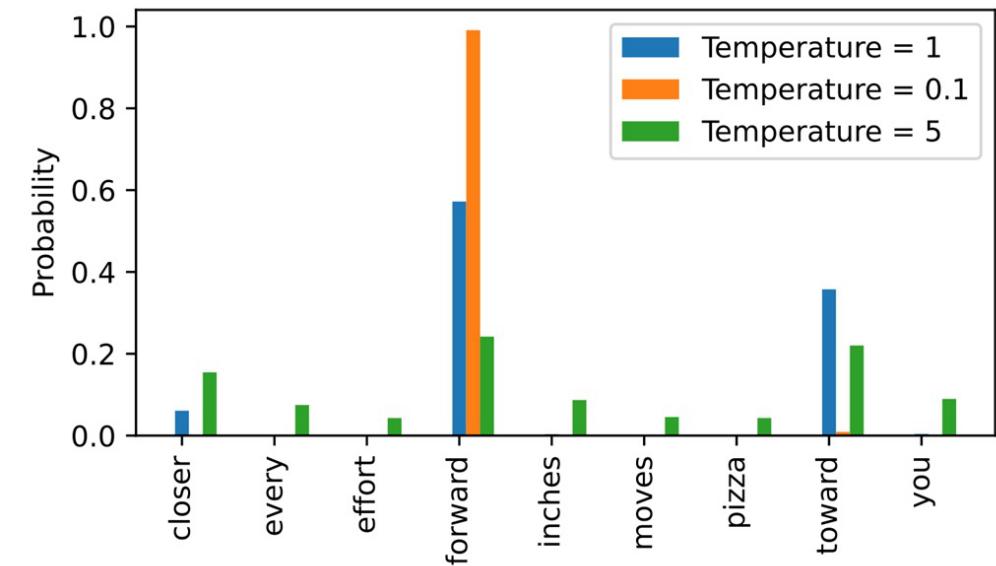
LLaMa

# Decoding strategies

## □ Temperature scaling

```
def softmax_with_temperature(logits, temperature):  
    scaled_logits = logits / temperature  
    return torch.softmax(scaled_logits, dim=0)
```

higher temperature result in more uniformly distribution



Maybe grammatically incorrect

## □ Top-k sampling

[ 4.51, 0.89, -1.90, 6.75, 1.63, -1.62, -1.89, 6.28, 1.79 ]



Top-3

[ 0.06, 0.00, 0.00, 0.57, 0.00, 0.00, 0.00, 0.36, 0.00 ]

# Computational Cost

## □ Inference

Float32 = 4 Byte

1B Parameter =  $4 * 10^9$  Byte =  $4 * 10^6$  KB =  $4 * 10^3$  MB = 4GB

7B (Float32) ~ 28GB

7B (Float16) ~ 14GB

## □ Train

Weight

$w$

Gradient

$w$

Optimizer

SGD: $w$  AdamW: $2w$

Activations

\*\*

Train  $\approx$  4 Inference

AdamW:

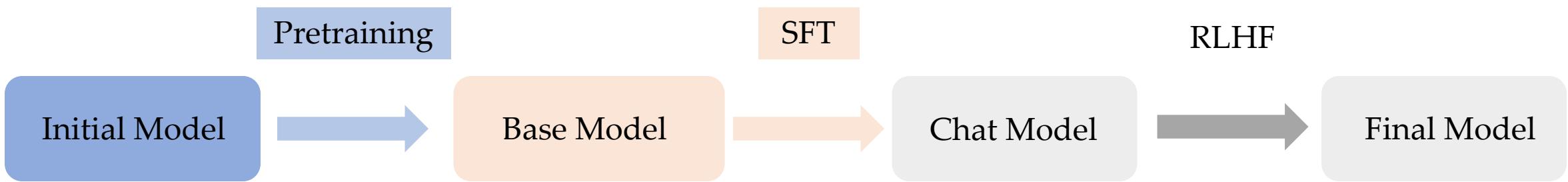
$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

$$y = w \cdot x$$

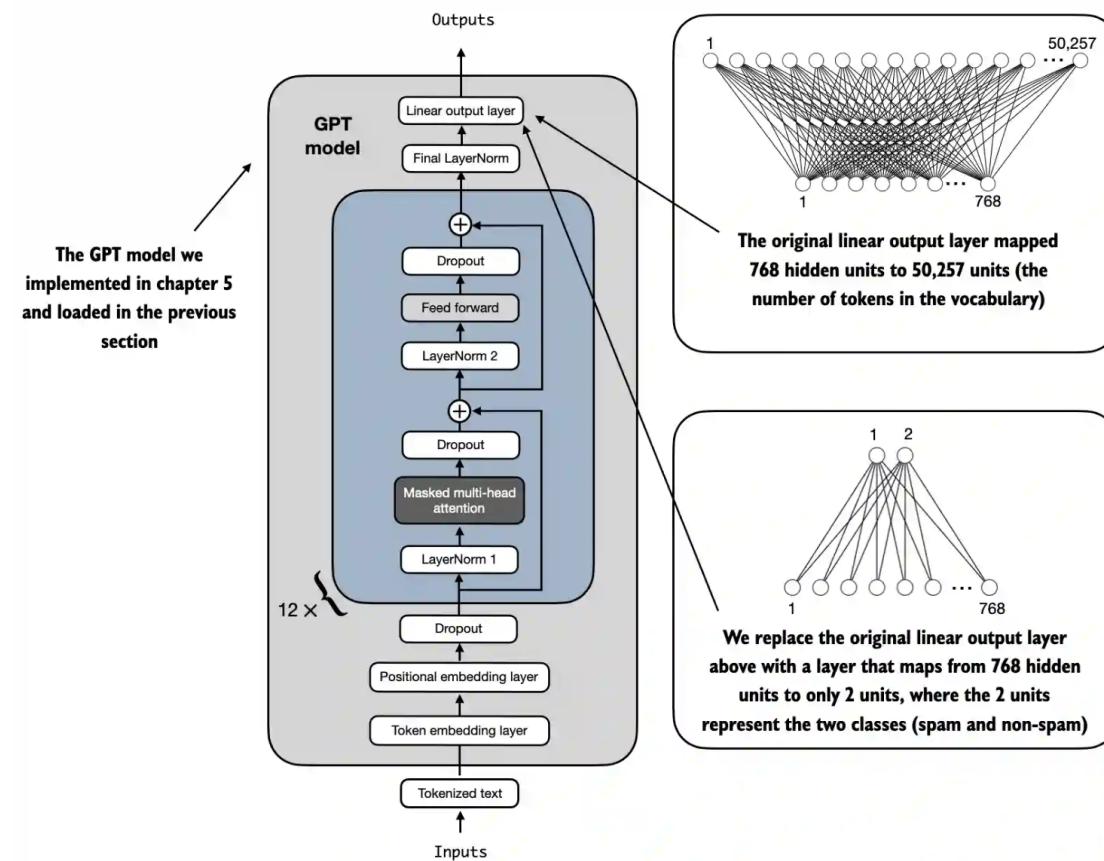
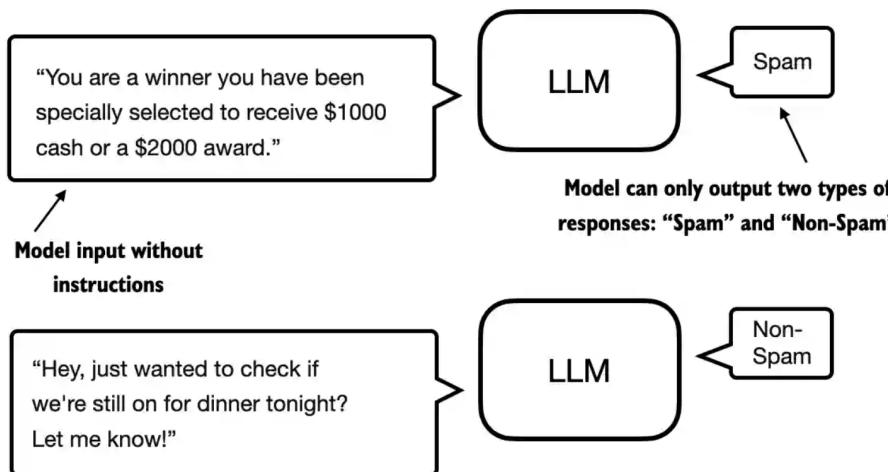
Save  $x$  to com. the grad of  $w$

# Building an LLM



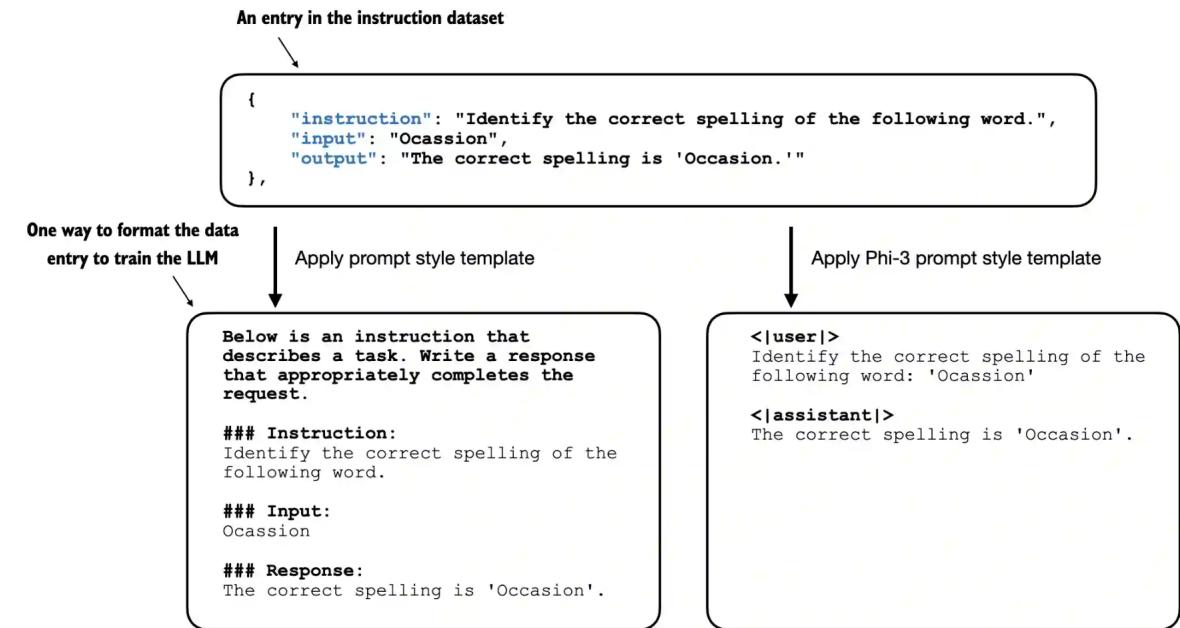
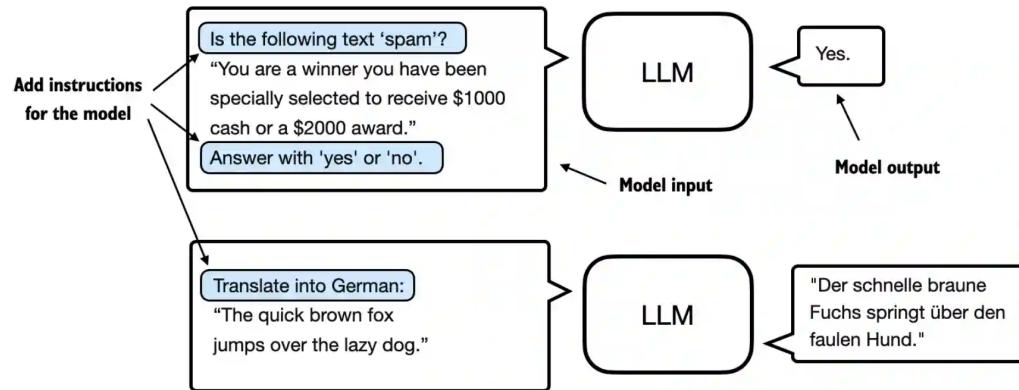
# Supervised Fine-Tuning (SFT):

## □ Text classification



# Supervised Fine-Tuning (SFT):

## □ Instruction Following

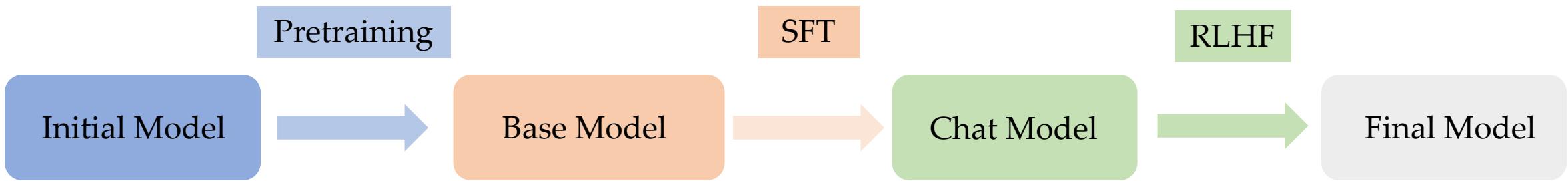


Quality is more important than quantity.  
(10k sample size has good results,  
1-2 epoch is enough.)

still next token prediction

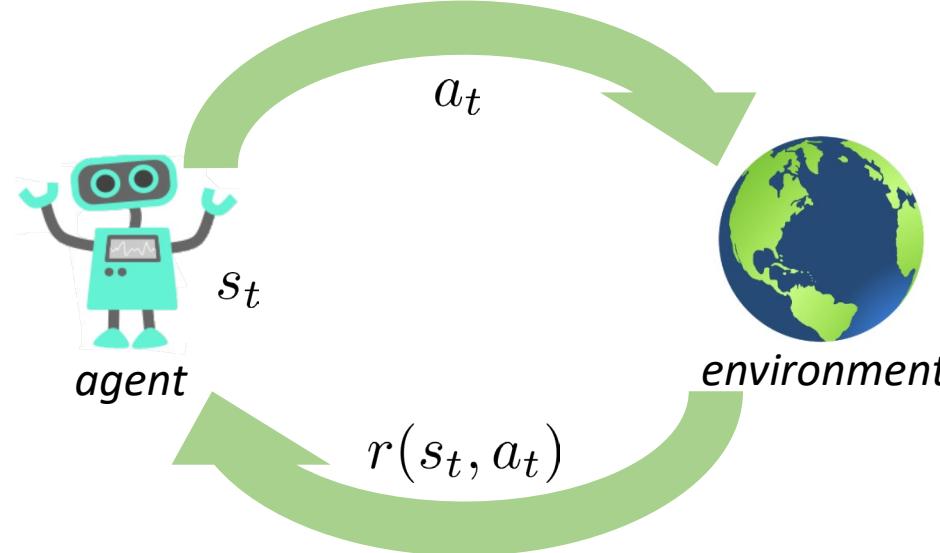
$$L_{\text{SFT}}(\mathbf{x}, \mathbf{y}) = - \sum_t \log \pi_\theta(y_t \mid x, y_1, \dots, y_{t-1})$$

# Building an LLM



# Fine-tune with RL

## □ RL



$$V_\pi(s) = \mathbb{E}_\pi [G_t \mid s_t = s] = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right], s \in S$$

$$Q_\pi(s, a) = \mathbb{E}_\pi [G_t \mid s_t = s, a_t = a] = \mathbb{E}_\pi \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right]$$

$$\pi^* = \operatorname{argmax}_\pi V_\pi(s)$$

# Why RL is Necessary?

---

- RL can consider the overall impact more than SL

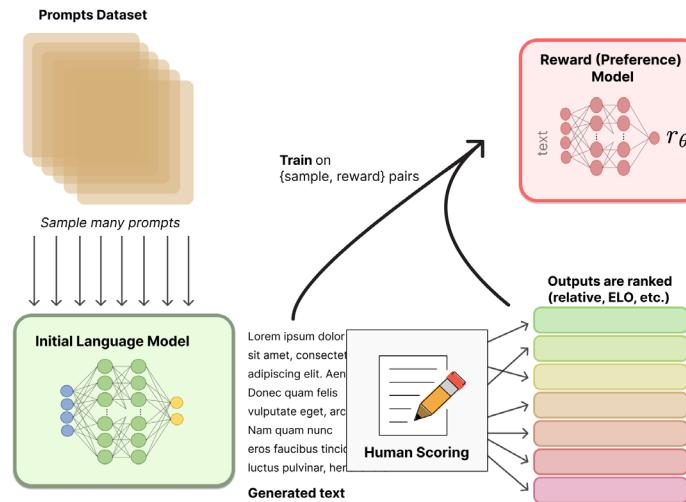
- SL: individual tokens
  - RL: entire output text

- RL is easier to solve hallucinations.

- SL: input->output
  - RL: penalty for incorrect answers

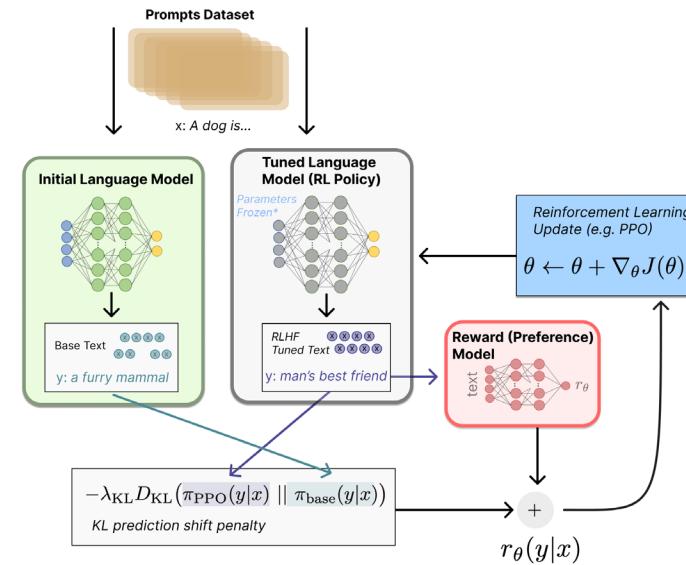
- RL can consider multiple rounds of dialog

# Step 1



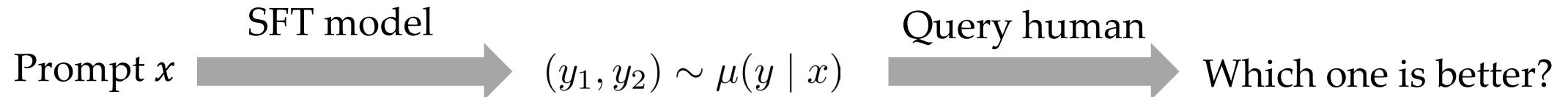
Training reward model

# Step 2



Fine-tune with RL

# Reward Model



## □ Bradley-Terry (BT) Model

$$p^*(y_1 \succ y_2 | x) = \frac{\exp(r^*(x, y_1))}{\exp(r^*(x, y_1)) + \exp(r^*(x, y_2))}$$

## □ Maximum Likelihood Estimation (MLE)

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} [\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))] \quad \sigma(w) = \frac{1}{1 + e^{-w}}$$

# Dataset

Dataset	#Prompts	Prompt Len.	Preferred Len.	Rejected Len.	Completion	Annotator	#Pairs
HH-RLHF	115092	160.4	82.2	73.6	LLM	Human	115396
SHP	31003	186.2	173.6	88.8	Human	Human	93301
HelpSteer	8592	530	116.4	89.3	LLM	Human	37131
PKU-SafeRLHF-30K	6975	21.5	70.4	74.6	LLM	Human	26874
UltraFeedback	63591	161.5	279.5	211.1	LLM	GPT-4	340025
UltraInteract	76086	507.4	396.6	416.7	LLM	GPT-4	161927
CodeUltraFeedback	9938	172.8	427.6	400.6	LLM	GPT-3.5	50156
Argilla-Math	2352	36.5	276.5	265.3	LLM	GPT-4	2418
OpenOrca	6791	153.3	165.4	260.5	LLM	GPT-4	6926
Capybara	14740	634.5	348.4	401.9	LLM	GPT-4	14811

## □ Example: UltraFeedback

- Generate 4 responses per prompt using 4 different LLMs, then evaluate by GPT-4
- 4 different aspects, namely instruction-following, truthfulness, honesty and helpfulness

# Evaluation

prompt:

Give a brief summary of the intention of the dialogue that just happened.

Customer: Hi there, I'm looking for a new phone. AI: Hi! What type of phone are you looking for? Customer: I'm not sure. Maybe something with a good camera? AI: We have a few phones with great cameras. Would you like to see some options? Customer: Yeah, that would be great.

chosen:

The customer approached with the intention of finding a new phone, particularly one with a good camera. The AI, serving as a sales assistant or customer service representative, acknowledged the customer's need and offered to present several phone options that feature high-quality cameras. The customer expressed interest in seeing these options. The dialogue's purpose is to assist the customer in finding a suitable phone with a good camera.

chosen\_model:

GPT4-Turbo

rejected:

The customer was looking for a new phone and the AI recommended some options based on their conversation. The customer chose to see some options and the AI provided them with a selection of phones with great cameras.

rejected\_model:

alpaca-7b

## RewardBench: Evaluating Reward Models

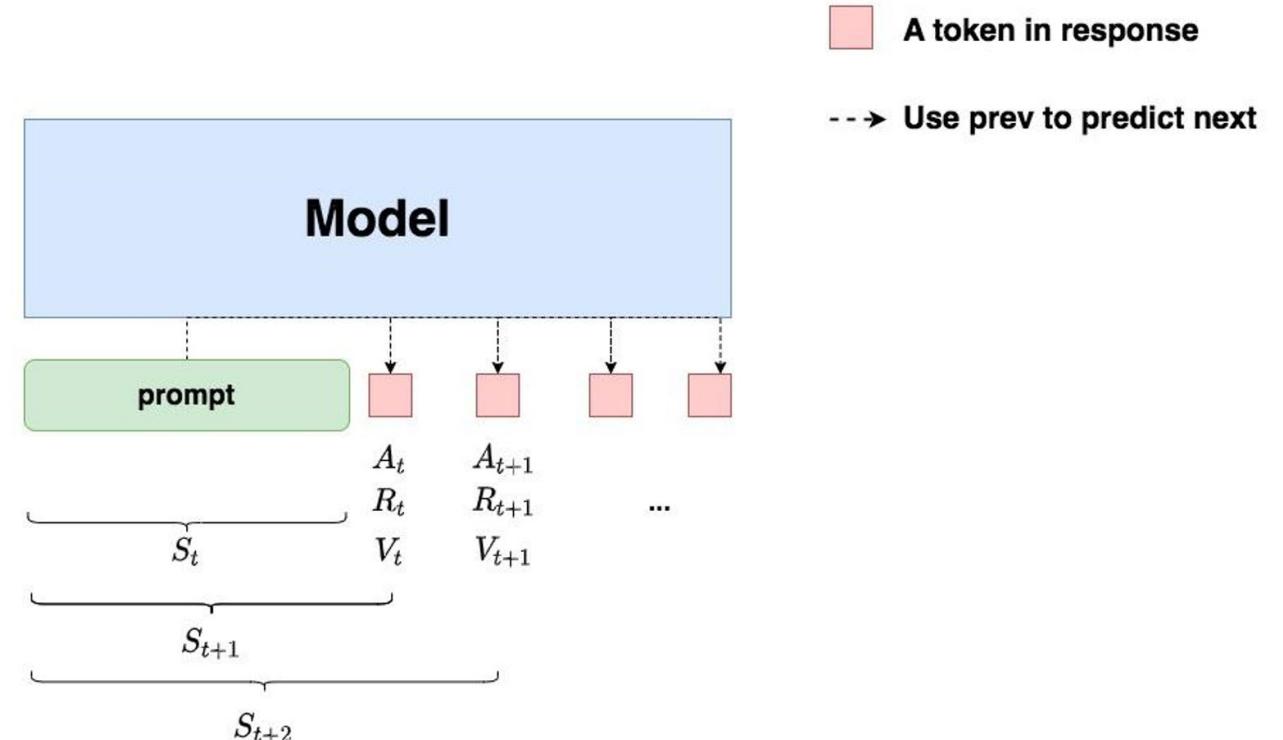
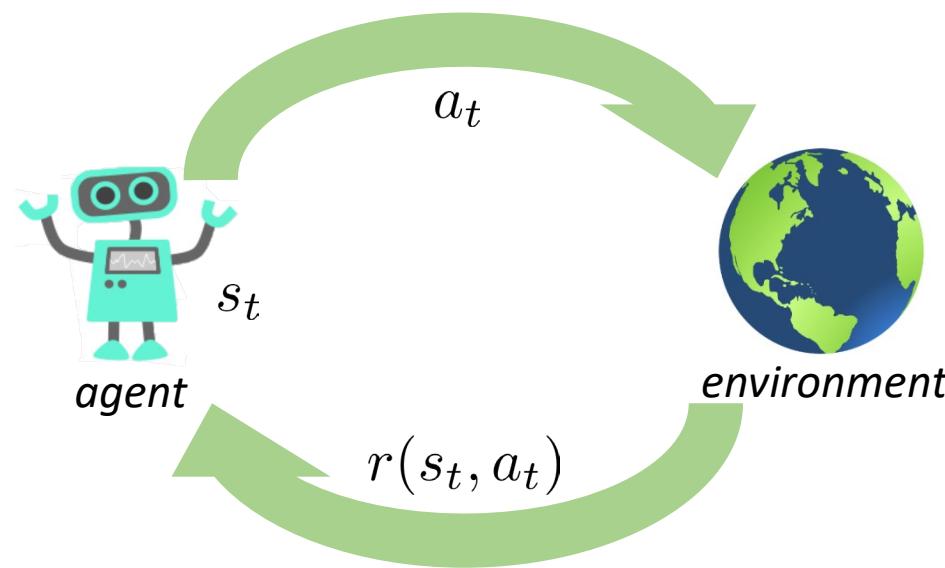
Evaluating the capabilities, safety, and pitfalls of reward models

[Code](#) | [Eval. Dataset](#) | [Prior Test Sets](#) | [Results](#) | [Paper](#) | Total models: 142 | \* Unverified models



Model		Model Type	Score	Chat	Chat Hard	Safety	Reasoning
1	<a href="#">Skywork/Skywork-Reward-Gemma-2-27B</a>	Seq. Classifier	93.8	95.8	91.4	91.9	96.1
2	<a href="#">SF-Foundation/TextEval-llama3.1-70B</a>	Generative	93.5	94.1	90.1	93.2	96.4
3	<a href="#">Skywork/Skywork-Critic-llama-3.1-70B</a>	Generative	93.3	96.6	87.9	93.1	95.5
4	<a href="#">LxzGordon/URM-LLaMa-3.1-8B</a>	Seq. Classifier	92.9	95.5	88.2	91.1	97.0
5	<a href="#">Salesforce/SFR-LLaMa-3.1-70B-Judge-x</a> *	Generative	92.7	96.9	84.8	91.6	97.6
6	<a href="#">Skywork/Skywork-Reward-llama-3.1-8B</a>	Seq. Classifier	92.5	95.8	87.3	90.8	96.2
7	<a href="#">nvidia/Nemotron-4-340B-Reward</a> *	Custom Classifier	92.0	95.8	87.1	91.5	93.6
8	<a href="#">Ray2333/GRM-llama3-8B-rewardmodel-ft</a>	Seq. Classifier	91.5	95.5	86.2	90.8	93.6
9	<a href="#">SF-Foundation/TextEval-OffsetBias-12B</a>	Generative	91.0	91.9	86.6	92.0	93.6
10	<a href="#">RLHFlow/ArmoRM-llama3-8B-v0.1</a>	Custom Classifier	90.4	96.9	76.8	90.5	97.3

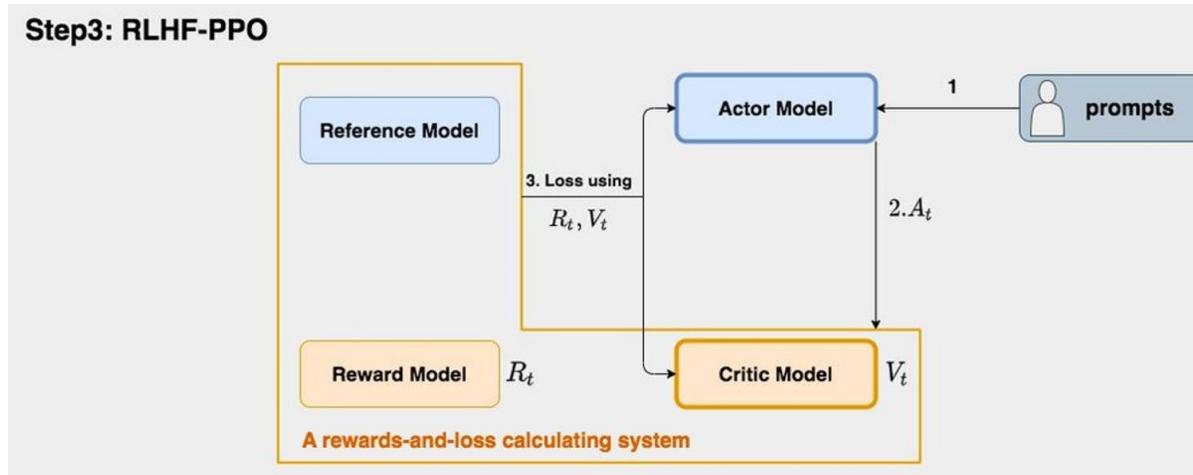
# RL in NLP



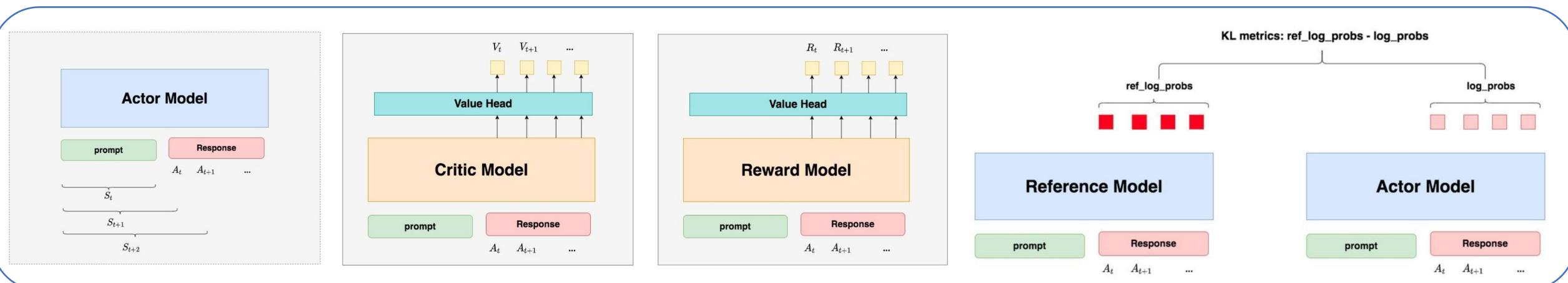
- $R_t$ : Immediate rewards of action  $A_t$
- $V_t$ : Expected total return following action  $A_t$

# RLHF-PPO

## Step3: RLHF-PPO

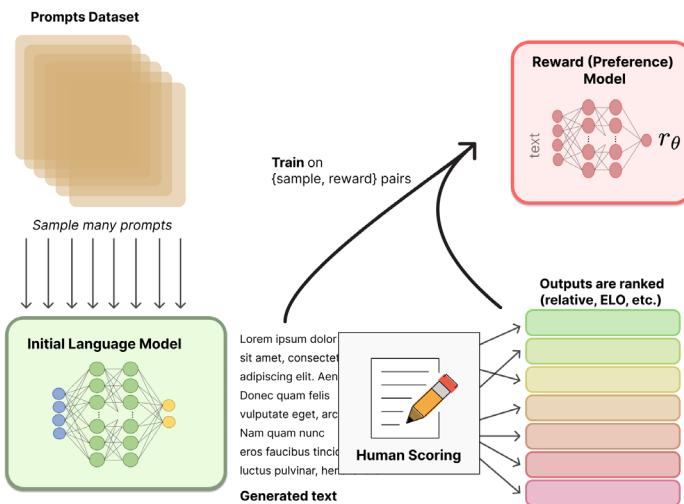


- **Actor Model**: the LM we want
- **Critic Model**: estimate  $V_t$
- **Reward Model**: compute  $R_t$
- **Reference Model**: add constraint

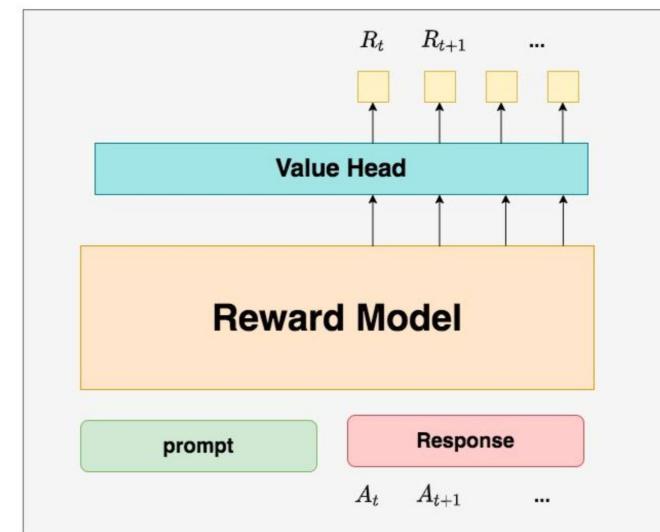


# Reward Model

$$r_\theta = f(x, y)$$



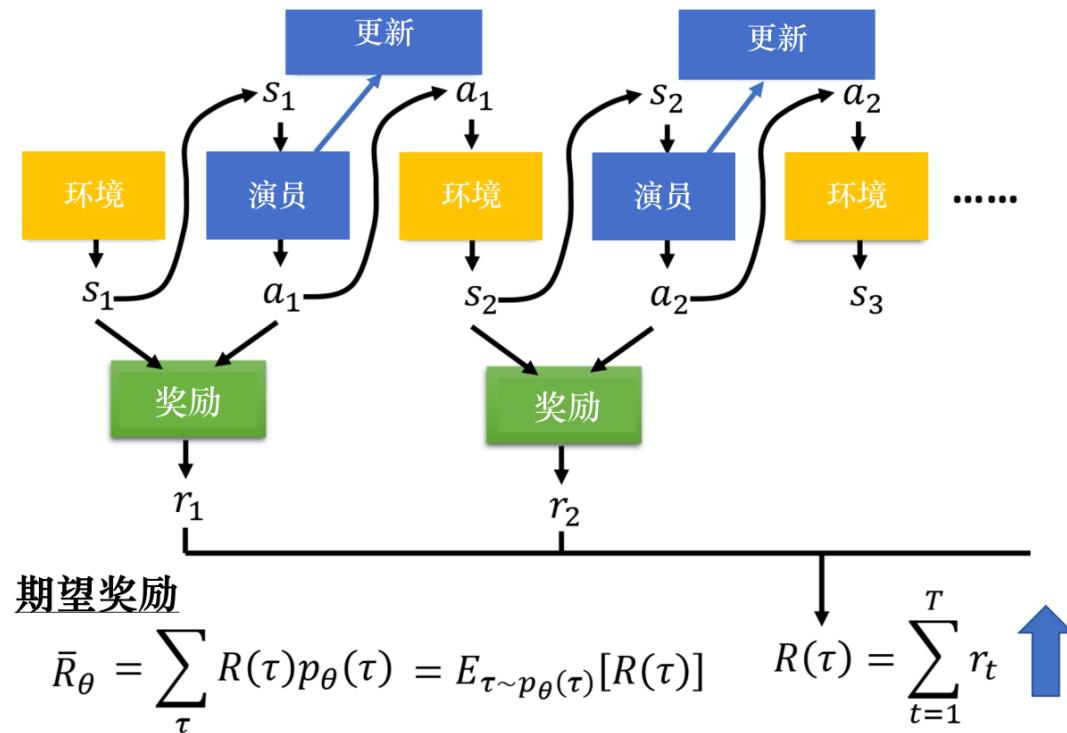
reward for response



reward for each token

$$\left\{ \begin{array}{l} R_t = -kl\_ctl * \left( \log \frac{\pi(A_t|S_t)}{\pi_{ref}(A_t|S_t)} \right), t \neq T \\ R_t = -kl\_ctl * \left( \log \frac{\pi(A_t|S_t)}{\pi_{ref}(A_t|S_t)} \right) + r_t, t = T \end{array} \right.$$

- If  $t < T$ , keep close to reference policy.
- If  $t = T$ , try to obtain high reward.



A trajectory:

$$\tau = \{s_1, a_1, s_2, a_2, \dots, s_T, a_T\}$$

$$p_\theta(\tau) = p(s_1) \prod_{t=1}^T \pi_\theta(a_t | s_t) p(s_{t+1} | s_t, a_t)$$

Expected return:

$$J_\theta = \sum_\tau R(\tau)p_\theta(\tau) = \mathbb{E}_{\tau \sim p_\theta(\tau)}[R(\tau)]$$

Gradient descent:

$$\nabla J_\theta = \sum_\tau R(\tau) \nabla p_\theta(\tau)$$

$$\theta \leftarrow \theta + \eta \nabla J_\theta$$

# Policy Gradient

$$p_{\theta}(\tau) = p(s_1) \prod_{t=1}^T \pi_{\theta}(a_t | s_t) p(s_{t+1} | s_t, a_t) \quad J_{\theta} = \sum_{\tau} R(\tau)p_{\theta}(\tau) = \mathbb{E}_{\tau \sim p_{\theta}(\tau)}[R(\tau)]$$

How to compute gradient  $\nabla J_{\theta}(\tau)$ ?

Log trick:

$$\nabla \log f(x) = \frac{1}{f(x)} \nabla f(x) \implies \nabla f(x) = f(x) \nabla \log f(x)$$

$$\nabla J_{\theta} = \sum_{\tau} R(\tau) \nabla p_{\theta}(\tau)$$

$$= \sum_{\tau} R(\tau) p_{\theta}(\tau) \nabla \log p_{\theta}(\tau)$$

$$= \mathbb{E}_{\tau \sim p_{\theta}(\tau)} [R(\tau) \nabla \log p_{\theta}(\tau)]$$

$$\nabla \log p_{\theta}(\tau) = \nabla \left( \log p(s_1) + \sum_{t=1}^T \log p_{\theta}(a_t | s_t) + \sum_{t=1}^T \log p(s_{t+1} | s_t, a_t) \right)$$

$$= \sum_{t=1}^T \nabla \log p_{\theta}(a_t | s_t)$$

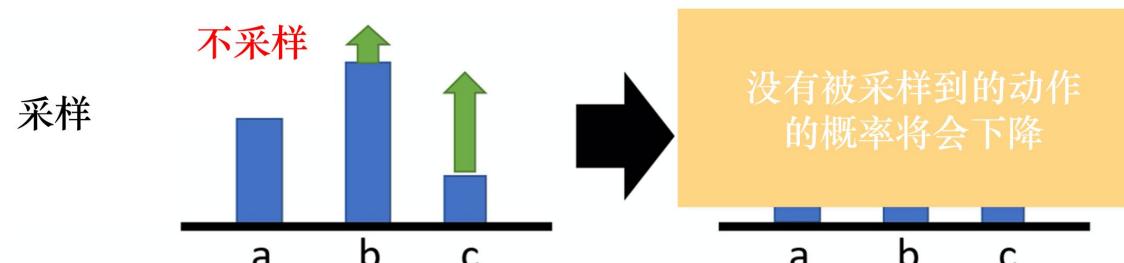
$$\nabla J_{\theta} = \mathbb{E}_{\tau \sim p_{\theta}(\tau)} [R(\tau) \nabla \log p_{\theta}(\tau)] = \mathbb{E}_{\tau \sim p_{\theta}(\tau)} \left[ R(\tau) \sum_{t=1}^T \nabla \log p_{\theta}(a_t | s_t) \right]$$

# Advantage

$$\nabla J_\theta = \mathbb{E}_{\tau \sim p_\theta(\tau)} [R(\tau) \nabla \log p_\theta(\tau)] = \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[ R(\tau) \sum_{t=1}^T \nabla \log p_\theta(a_t | s_t) \right] \quad \theta \leftarrow \theta + \eta \nabla J_\theta$$

If  $R(\tau) > 0$ , then increase the probability of  $p(a_t|s_t)$ ; if  $R(\tau) < 0$ , then decrease.

What happens if the return is always positive?



$$\nabla J_\theta \approx \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} (R(\tau^n) - b) \nabla \log p_\theta(a_t^n | s_t^n)$$

Advantage

# Reward to go

Expected Grad-Log-Prob Lemma

$$\mathbb{E}_{x \sim P_\theta} [\nabla_\theta \log P_\theta(x)] = 0$$

$$\begin{aligned}
 \nabla_\theta J(\theta) &= \mathbb{E}_{\tau \sim p_\theta(\tau)} \left[ (R(\tau) - b) \sum_{t=1}^T \nabla \log p_\theta(a_t | s_t) \right] \\
 &= \sum_{t=1}^T \left( \mathbb{E}_{\tau \sim \pi_\theta(\tau)} \left[ \nabla_\theta \log \pi_\theta(a_t | s_t) \sum_{t'=1}^{t-1} r(s_{t'}, a_{t'}) \right] + \mathbb{E}_{\tau \sim \pi_\theta(\tau)} \left[ \nabla_\theta \log \pi_\theta(a_t | s_t) \left( \sum_{t'=t}^T r(s_{t'}, a_{t'}) - b \right) \right] \right) \\
 &\quad = 0 \qquad \qquad \qquad A_t = r_t + \gamma V(s_{t+1}) - V(s_t)
 \end{aligned}$$

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)} [A_t(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t | s_t)]$$

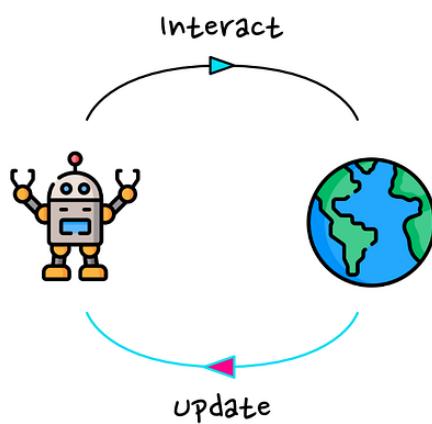
# Proximal Policy Optimization

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} [A_t(s_t, a_t) \nabla_{\theta} \log \pi_{\theta}(a_t | s_t)]$$

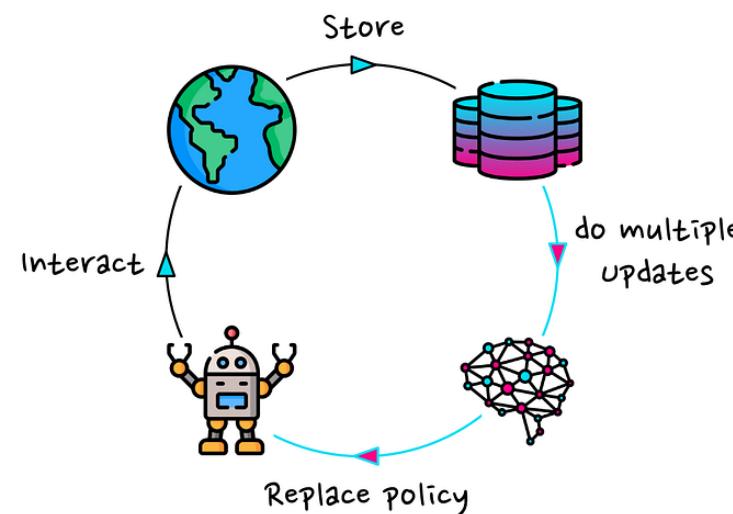
$$\theta_{t+1} \leftarrow \theta_t + \eta \nabla J_{\theta_t}$$

The data collection is expensive.

on-policy



off-policy



- On-Policy: attempt to evaluate or improve the policy that is used to make decisions
- Off-Policy: evaluate or improve a policy different from that used to generate the data

# PPO Target

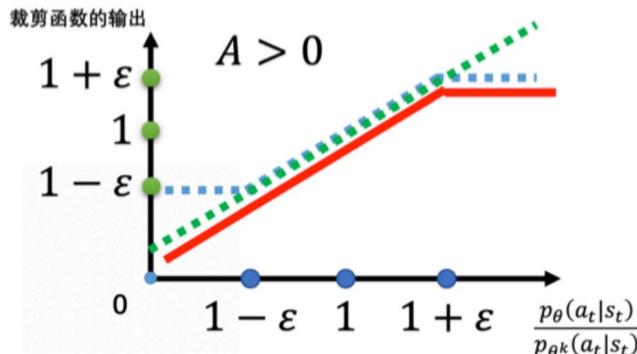
## □ Importance Sampling

$$\begin{aligned}\nabla_{\theta} J(\theta) &= \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} [A_t(s_t, a_t) \nabla_{\theta} \log \pi_{\theta}(a_t | s_t)] \\&= \mathbb{E}_{(s_t, a_t) \sim \pi_{\theta'}} \left[ \frac{p_{\theta}(s_t, a_t)}{p_{\theta'}(s_t, a_t)} A^{\theta'}(s_t, a_t) \nabla \log p_{\theta}(a_t | s_t) \right] \\&= \mathbb{E}_{(s_t, a_t) \sim \pi_{\theta'}} \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta'}(a_t | s_t)} A^{\theta'}(s_t, a_t) \nabla \log p_{\theta}(a_t | s_t) \right] \quad (\text{assume } p_{\theta}(s_t) = p_{\theta'}(s_t)) \\&\downarrow \\J^{\theta'}(\theta) &= \mathbb{E}_{(s_t, a_t) \sim \pi_{\theta'}} \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta'}(a_t | s_t)} A^{\theta'}(s_t, a_t) \right] \\&\downarrow \\\text{PPO Target: } J_{\text{PPO}}^{\theta'}(\theta) &= J^{\theta'}(\theta) - \beta \text{KL}(\theta, \theta')\end{aligned}$$

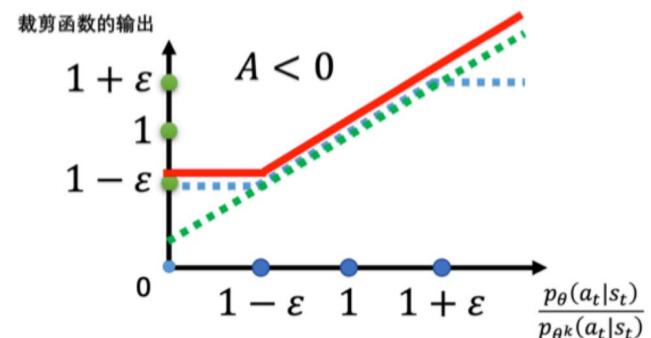
PPO Target:  $J_{\text{PPO}}^{\theta'}(\theta) = J^{\theta'}(\theta) - \beta \text{KL}(\theta, \theta')$

$$J^{\theta'}(\theta) = \mathbb{E}_{(s_t, a_t) \sim \pi_{\theta'}} \left[ \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta'}(a_t | s_t)} A^{\theta'}(s_t, a_t) \right]$$

$$J_{\text{PPO2}}^{\theta^k}(\theta) \approx \sum_{(s_t, a_t)} \min \left( \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta^k}(a_t | s_t)} A^{\theta^k}(s_t, a_t), \text{clip} \left( \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta^k}(a_t | s_t)}, 1 - \varepsilon, 1 + \varepsilon \right) A^{\theta^k}(s_t, a_t) \right)$$



(a)  $A > 0$

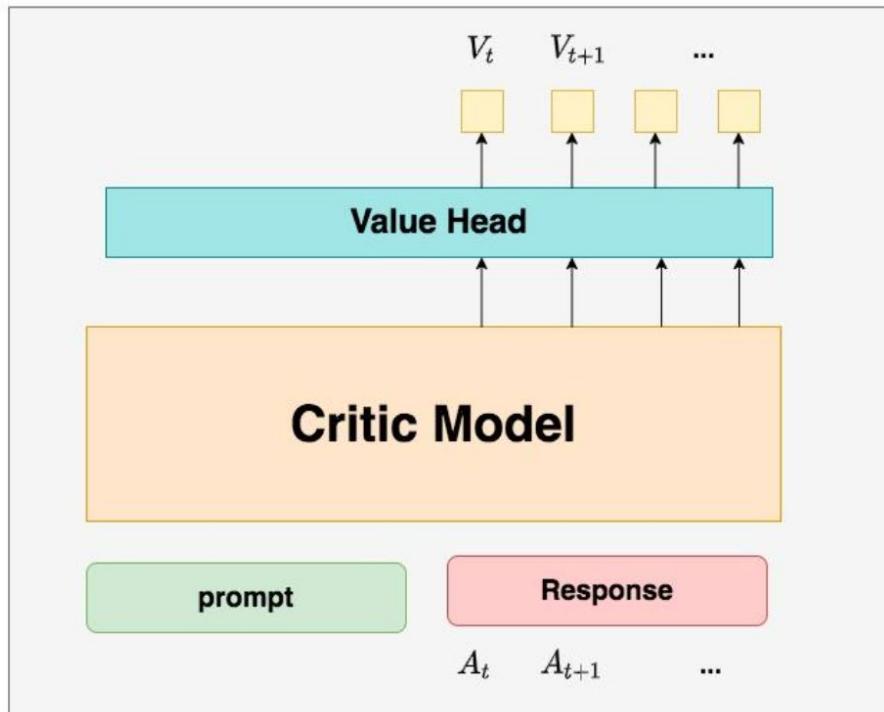


(b)  $A < 0$

- If  $A > 0$ , increase  $\pi_{\theta}(a_t | s_t)$  while keep  $\pi_{\theta}(a_t | s_t) / \pi_{\theta^k}(a_t | s_t) < 1 + \varepsilon$ .
- If  $A < 0$ , decrease  $\pi_{\theta}(a_t | s_t)$  while keep  $\pi_{\theta}(a_t | s_t) / \pi_{\theta^k}(a_t | s_t) > 1 - \varepsilon$ .

# Critic Model

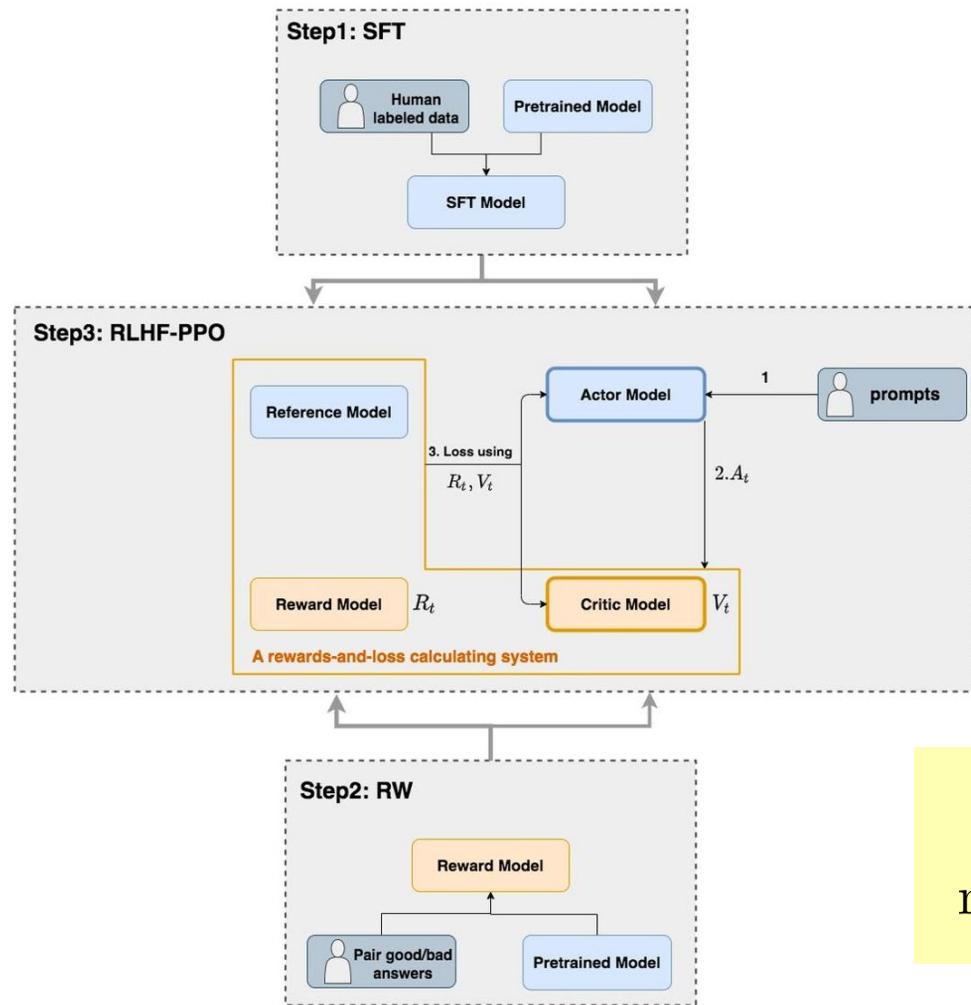
- ❑ Critic model: estimate the total expected return



- Old estimate:  $V_t$
- New estimate after observing  $r_t$ :  $r_t + \gamma * V_{t+1}$

$$\text{Critic loss} = (r_t + \gamma * V_{t+1} - V_t)^2$$

# RLHF-PPO



- **Actor Model**: the LM we want
- **Critic Model**: estimate  $V_t$
- **Reward Model**: compute  $R_t$
- **Reference Model**: add constraint

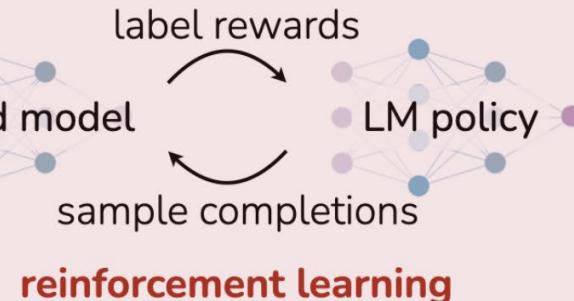
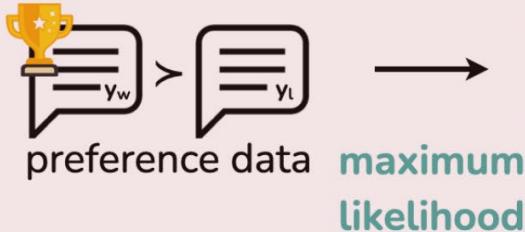
Target:

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} [r_\phi(x, y)] - \beta \mathbb{D}_{\text{KL}} [\pi_\theta(y | x) \| \pi_{\text{ref}}(y | x)]$$

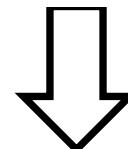
# RLHF vs DPO

## Reinforcement Learning from Human Feedback (RLHF)

x: "write me a poem about  
the history of jazz"



reinforcement learning



## Direct Preference Optimization (DPO)

x: "write me a poem about  
the history of jazz"



learn policies directly from preference  
without a reward model

# Direct Preference Optimization (DPO)

## □ RL objective

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} [r_\phi(x, y)] - \beta \mathbb{D}_{\text{KL}} [\pi_\theta(y | x) \| \pi_{\text{ref}}(y | x)]$$

Closed-form solution

$$\pi_r(y | x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y | x) \exp \left( \frac{1}{\beta} r(x, y) \right) \quad Z(x) = \sum_y \pi_{\text{ref}}(y | x) \exp \left( \frac{1}{\beta} r(x, y) \right)$$

expensive to estimate

$$r(x, y) = \beta \log \frac{\pi_r(y | x)}{\pi_{\text{ref}}(y | x)} + \beta \log Z(x)$$

Your Language Model is Secretly a Reward Model !

# Direct Preference Optimization (DPO)

□ Optimal RLHF policy satisfies

$$p^*(y_1 \succ y_2 | x) = \frac{1}{1 + \exp \left( \beta \log \frac{\pi^*(y_2|x)}{\pi_{\text{ref}}(y_2|x)} - \beta \log \frac{\pi^*(y_1|x)}{\pi_{\text{ref}}(y_1|x)} \right)}$$

□ Maximum Likelihood Objective

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

□ What does the DPO update do?

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\beta \mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[ \underbrace{\sigma(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w))}_{\text{higher weight when reward estimate is wrong}} \left[ \underbrace{\nabla_\theta \log \pi(y_w | x)}_{\text{increase likelihood of } y_w} - \underbrace{\nabla_\theta \log \pi(y_l | x)}_{\text{decrease likelihood of } y_l} \right] \right]$$

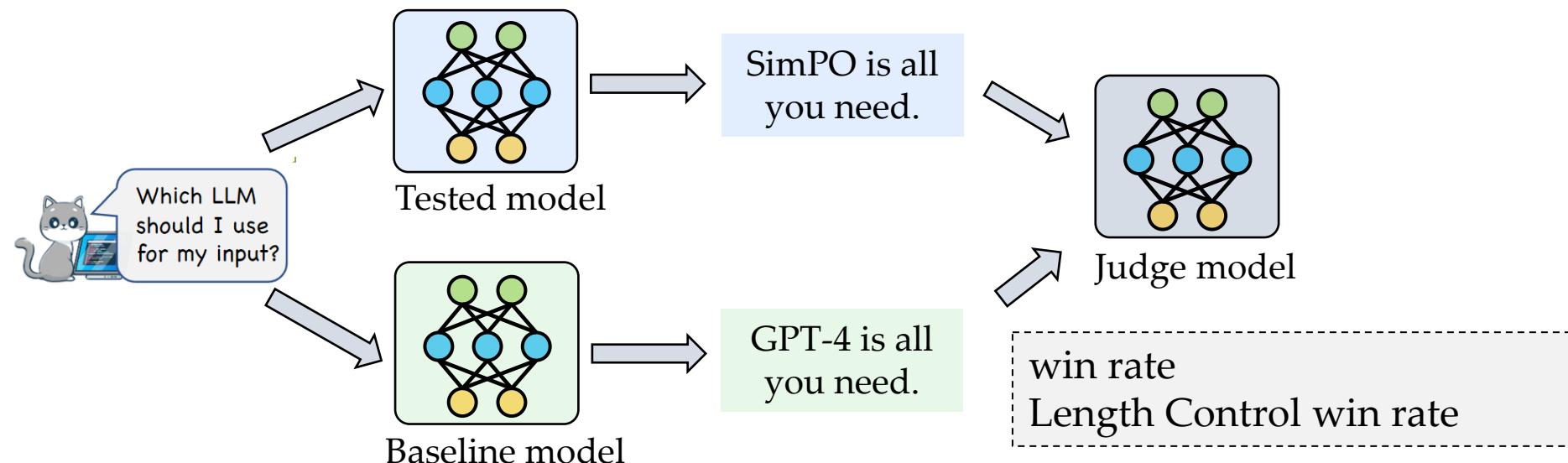
## □ Other RLHF methods

Method	Objective	Hyperparameter
DPO [62]	$-\log \sigma \left( \beta \log \frac{\pi_\theta(y_w x)}{\pi_{\text{ref}}(y_w x)} - \beta \log \frac{\pi_\theta(y_l x)}{\pi_{\text{ref}}(y_l x)} \right)$	$\beta \in [0.01, 0.05, 0.1]$
IPO [6]	$\left( \log \frac{\pi_\theta(y_w x)}{\pi_{\text{ref}}(y_w x)} - \log \frac{\pi_\theta(y_l x)}{\pi_{\text{ref}}(y_l x)} - \frac{1}{2\tau} \right)^2$	$\tau \in [0.01, 0.1, 0.5, 1.0]$
KTO [25]	$-\lambda_w \sigma \left( \beta \log \frac{\pi_\theta(y_w x)}{\pi_{\text{ref}}(y_w x)} - z_{\text{ref}} \right) + \lambda_l \sigma \left( z_{\text{ref}} - \beta \log \frac{\pi_\theta(y_l x)}{\pi_{\text{ref}}(y_l x)} \right),$ where $z_{\text{ref}} = \mathbb{E}_{(x,y) \sim \mathcal{D}} [\beta \text{KL}(\pi_\theta(y x)    \pi_{\text{ref}}(y x))]$	$\lambda_l = \lambda_w = 1$ $\beta \in [0.01, 0.05, 0.1]$
ORPO [38]	$-\log p_\theta(y_w x) - \lambda \log \sigma \left( \log \frac{p_\theta(y_w x)}{1-p_\theta(y_w x)} - \log \frac{p_\theta(y_l x)}{1-p_\theta(y_l x)} \right),$ where $p_\theta(y x) = \exp \left( \frac{1}{ y } \log \pi_\theta(y x) \right)$	$\lambda \in [0.1, 0.5, 1.0, 2.0]$
R-DPO [60]	$-\log \sigma \left( \beta \log \frac{\pi_\theta(y_w x)}{\pi_{\text{ref}}(y_w x)} - \beta \log \frac{\pi_\theta(y_l x)}{\pi_{\text{ref}}(y_l x)} - (\alpha y_w  - \alpha y_l ) \right)$	$\alpha \in [0.05, 0.1, 0.5, 1.0]$ $\beta \in [0.01, 0.05, 0.1]$
SimPO	$-\log \sigma \left( \frac{\beta}{ y_w } \log \pi_\theta(y_w x) - \frac{\beta}{ y_l } \log \pi_\theta(y_l x) - \gamma \right)$	$\beta \in [2.0, 2.5]$ $\gamma \in [0.3, 0.5, 1.0, 1.2, 1.4, 1.6]$

# Benchmarks

	# Exs.	Baseline Model	Judge Model	Scoring Type	Metric
<b>AlpacaEval 2</b>	805	GPT-4 Turbo	GPT-4 Turbo	Pairwise comparison	LC & raw win rate
<b>Arena-Hard</b>	500	GPT-4-0314	GPT-4 Turbo	Pairwise comparison	Win rate
<b>MT-Bench</b>	80	-	GPT-4/GPT-4 Turbo	Single-answer grading	Rating of 1-10

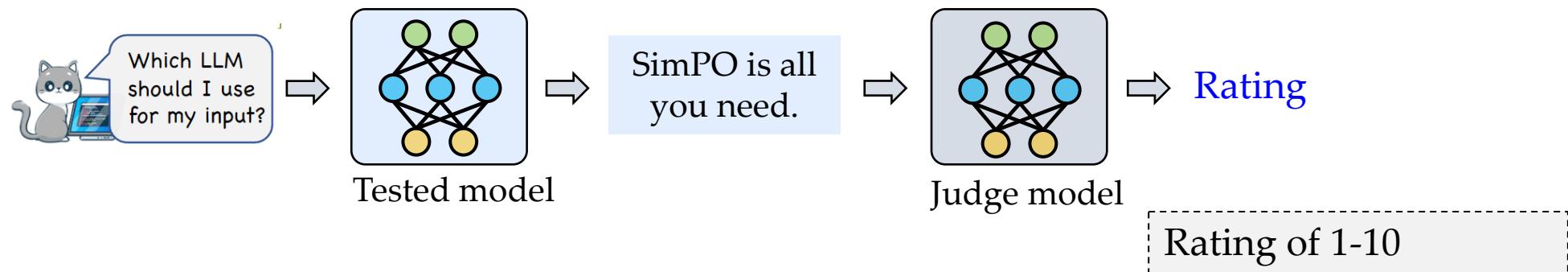
## □ Pairwise comparison:



# Benchmarks

	# Exs.	Baseline Model	Judge Model	Scoring Type	Metric
<b>AlpacaEval 2</b>	805	GPT-4 Turbo	GPT-4 Turbo	Pairwise comparison	LC & raw win rate
<b>Arena-Hard</b>	500	GPT-4-0314	GPT-4 Turbo	Pairwise comparison	Win rate
<b>MT-Bench</b>	80	-	GPT-4/GPT-4 Turbo	Single-answer grading	Rating of 1-10

## □ Single-answer comparison:



# AlpacaEval 2

## AlpacaEval Leaderboard

An Automatic Evaluator for Instruction-following Language Models  
Length-controlled (LC) win rates alleviate length biases of GPT-4, but it may favor models finetuned on its outputs.



Version: AlpacaEval AlpacaEval 2.0

Filter: Community Verified

Baseline: GPT-4 Preview (11/06) | Auto-annotator: GPT-4 Preview (11/06)

Rank	Model Name	LC Win Rate	Win Rate
1	GPT-4 Omni (05/13) <a href="#">🔗</a>	57.5%	51.3%
2	GPT-4 Turbo (04/09) <a href="#">🔗</a>	55.0%	46.1%
3	Yi-Large Preview <a href="#">🔗</a>	51.9%	57.5%
4	GPT-4o Mini (07/18) <a href="#">🔗</a>	50.7%	44.7%
5	GPT-4 Preview (11/06) <a href="#">🔗</a>	50.0%	50.0% <span style="border: 2px solid red; border-radius: 50%; padding: 2px;"> </span>
6	Claude 3 Opus (02/29) <a href="#">🔗</a>	40.5%	29.1%
7	Llama 3.1 405B Instruct <a href="#">🔗</a>	39.3%	39.1%
8	GPT-4 <a href="#">🔗</a>	38.1%	23.6%
9	Qwen2 72B Instruct <a href="#">🔗</a>	38.1%	29.9%
10	Llama 3.1 70B Instruct <a href="#">🔗</a>	38.1%	39.1%
11	Qwen1.5 72B Chat <a href="#">🔗</a>	36.6%	26.5%

```
[{"dataset": "helpful_base", "instruction": "What are the names of some famous actors that started their careers on Broadway?", "output": "Several famous actors started their careers on Broadway before making it big in film and television. Here are a few notable examples:\n\n1. Sarah Jessica Parker - Before she was Carrie Bradshaw on \"Sex and the City,\" Sarah Jessica Parker was a Broadway star, having appeared in productions like \"Annie\" as a child.\n\n2. Meryl Streep - Meryl Streep's early career included Broadway productions such as \"Trelawny of the 'Wells'\" and \"A Memory of Two Mondays\" / 27 Wagons Full of Cotton.\"\\n\\n3. Hugh Jackman - Hugh Jackman won a Tony Award for his role in \"The Boy from Oz\" and has been known for his stage work as well as his film career.\n\n4. Sutton Foster - Known for her television role in \"Younger,\" Sutton Foster is also a Broadway legend with leading roles in shows like \"Thoroughly Modern Millie\" and \"Anything Goes.\"\\n\\n5. Kristen Bell - Before she was the voice of Anna in \"Frozen\" or the star of \"The Good Place,\" Kristen Bell appeared in Broadway's \"The Adventures of Tom Sawyer\" and \"The Crucible.\"\\n\\n6. Audra McDonald - Audra McDonald is a renowned Broadway actress with a record-breaking number of Tony Awards. She's starred in \"Ragtime,\" \"Carousel,\" \"Master Class,\" and more.\n\n7. Nathan Lane - Nathan Lane is a Broadway veteran known for his roles in \"The Producers,\" \"A Funny Thing Happened on the Way to the Forum,\" and \"Angels in America.\"\\n\\n8. Idina Menzel - Before \"Frozen\" and \"Wicked\" made her a household name, Idina Menzel started on Broadway in shows like \"Rent\" and \"Hair.\"\\n\\n9. Lin-Manuel Miranda - Before \"Hamilton\" and \"In the Heights\" became huge hits, Lin-Manuel Miranda was performing on Broadway, eventually becoming a celebrated writer and actor.\n\n10. Lea Michele - Prior to her role on \"Glee,\" Lea Michele was a young Broadway actress in shows like \"Les Mis\u00e9rables,\" \"Ragtime,\" and \"Spring Awakening.\"\\n\\nThese actors are just a few examples of the many performers who have transitioned from the Broadway stage to broader fame in the entertainment industry. Broadway often serves as a proving ground for talent, and many actors continue to return to the stage throughout their careers.", "generator": "gpt4_1106_preview"}]
```

# MT-Bench

writing	writing	[ "Compose an engaging travel blog post about a recent trip to Hawaii, highlighting cultural experiences and must-see attractions.", "Rewrite your previous response. Start every sentence with the letter A." ]
✓ writing	writing	[ "Draft a professional email seeking your supervisor's feedback on the 'Quarterly Financial Report' you prepared. Ask specifically about the data analysis, presentation style, and the..."
roleplay	writing	[ "Imagine you are writing a blog post comparing two popular smartphone models. Develop an outline for the blog post, including key points and subheadings to effectively compare and..."
reasoning	writing	[ "Write a persuasive email to convince your introverted friend, who dislikes public speaking, to volunteer as a guest speaker at a local event. Use compelling arguments and..."
math		
coding	roleplay	[ "Pretend yourself to be Elon Musk in all the following conversations. Speak like Elon Musk as much as possible. Why do we need to go to Mars?", "How do you like dancing? Can you teach..."
extraction	roleplay	[ "Embrace the role of Sheldon from \"The Big Bang Theory\" as we delve into our conversation. Don't start with phrases like \"As Sheldon\". Let's kick things off with the..."
stem	roleplay	[ "Imagine yourself as a doctor tasked with devising innovative remedies for various ailments and maladies. Your expertise should encompass prescribing traditional medications, herbal..."
humanities	roleplay	[ "Please take on the role of a relationship coach. You'll be provided with details about two individuals caught in a conflict, and your task will be to offer suggestions for resolving..."

# Experiments

Model	AlpacaEval 2.0		
	LC Win Rate	Win Rate	Avg. Len
Mistral-7B-Instruct-v0.2	17.11	14.72	1676
Mistral-7B-Instruct-v0.2 (best-of-16)	22.45	17.94	1529
Mistral-7B-DPO Iter1	23.81	20.44	1723
Mistral-7B-DPO Iter2	24.23	24.46	2028
Mistral-7B-DPO Iter3	22.30	23.39	2189
Mistral-7B-IPO Iter1	23.78	20.77	1693
Mistral-7B-IPO Iter2	21.08	23.38	2660
Mistral-7B-IPO Iter3	20.06	22.47	2760

## AlpacaEval Leaderboard

An Automatic Evaluator for Instruction-following Language Models  
Length-controlled (LC) win rates alleviate length biases of GPT-4, but it may favor models finetuned on its outputs.

Rank	Model Name	LC Win Rate	Win Rate
1	GPT-4 Omni (05/13) <a href="#">🔗</a>	57.5%	51.3%
2	GPT-4 Turbo (04/09) <a href="#">🔗</a>	55.0%	46.1%
3	Yi-Large Preview <a href="#">🔗</a>	51.9%	57.5%
4	GPT-4o Mini (07/18) <a href="#">🔗</a>	50.7%	44.7%
5	GPT-4 Preview (11/06) <a href="#">🔗</a>	50.0%	50.0%
6	Claude 3 Opus (02/29) <a href="#">🔗</a>	40.5%	29.1%
7	Llama 3.1 405B Instruct <a href="#">🔗</a>	39.3%	39.1%
8	GPT-4 <a href="#">🔗</a>	38.1%	23.6%
9	Qwen2 72B Instruct <a href="#">🔗</a>	38.1%	29.9%
10	Llama 3.1 70B Instruct <a href="#">🔗</a>	38.1%	39.1%
11	Qwen1.5 72B Chat <a href="#">🔗</a>	36.6%	26.5%

# Open LLM Leaderboard

Open LLM Leaderboard

The previous Leaderboard version is live [here](#) 🎉 Feeling lost? Check out our [documentation](#) 📄

You'll notably find explanations on the evaluations we are using, reproducibility guidelines, best practices on how to submit a model, and our FAQ.

LLM Benchmark   Submit   Model Vote

Search  
Separate multiple queries with ';

Select Columns to Display:

<input checked="" type="checkbox"/> Average	<input checked="" type="checkbox"/> IFEval	<input type="checkbox"/> IFEval Raw	<input checked="" type="checkbox"/> BBH	<input type="checkbox"/> BBH Raw	<input checked="" type="checkbox"/> MATH Lvl 5	<input type="checkbox"/> MATH Lvl 5 Raw
<input checked="" type="checkbox"/> GPQA	<input type="checkbox"/> GPQA Raw	<input checked="" type="checkbox"/> MUSR	<input type="checkbox"/> MUSR Raw	<input checked="" type="checkbox"/> MMLU-PRO	<input type="checkbox"/> MMLU-PRO Raw	<input type="checkbox"/> Type
<input type="checkbox"/> Architecture	<input type="checkbox"/> Precision	<input type="checkbox"/> Not_Merged	<input type="checkbox"/> Hub License	<input type="checkbox"/> #Params (B)	<input type="checkbox"/> Hub ❤️	
<input type="checkbox"/> Model sha	<input type="checkbox"/> Submission Date	<input type="checkbox"/> Upload To Hub Date	<input type="checkbox"/> Chat Template	<input type="checkbox"/> Generation		
<input type="checkbox"/> Base Model						

Model types  
 chat models (RLHF, DPO, IFT, ...)  
 fine-tuned on domain-specific datasets  
 base merges and moerges

pretrained  
 continuously pretrained  
 other

Precision  
 bfloat16  
 float16  
 4bit

Select the number of parameters (B)  
7   12

Hide models  
 Deleted/incomplete  
 Merge/MoErge  
 MoE  
 Flagged  
 Show only maintainer's highlight

T	Model	Average	IFEval	BBH	MATH Lvl 5	GPQA	MUSR	MMLU-PRO
💬	<a href="#">dfurman/CalmeRys-78B-0pxo-v0.1</a> 📄	50.78	81.63	61.92	37.92	20.02	36.37	66.8
💬	<a href="#">MaziyarPanahi/calme-2.4-rys-78b</a> 📄	50.26	80.11	62.16	37.69	20.36	34.57	66.69
◆	<a href="#">dnhkng/RYS-XLarge</a> 📄	44.75	79.96	58.77	38.97	17.9	23.72	49.2
💬	<a href="#">MaziyarPanahi/calme-2.1-rys-78b</a> 📄	44.14	81.36	59.47	36.4	19.24	19	49.38
💬	<a href="#">MaziyarPanahi/calme-2.3-rys-78b</a> 📄	44.02	80.66	59.57	36.56	20.58	17	49.73

# Model Tree

The diagram illustrates a model tree structure. At the top, a purple box contains a link to a model tree for dfurman/CalmeRys-78B-Orpo-v0.1. Below this, a white box shows the model tree structure:

- Base model:** dnhkng/RYS-XLarge
- Finetuned:** MaziyarPanahi/calme-2.1-rys-78b
- Finetuned:** MaziyarPanahi/calme-2.4-rys-78b
- Finetuned (2):** this model
- Quantizations:** 3 models

A blue callout bubble points to the "this model" entry, which is highlighted with a blue border.

**Description of the model:**

This model is based on MaziyarPanahi/calme-2.1-qwen2-72b, which in turn was tuned from Qwen2-72B. As this method is orthogonal to fine-tuning, the further finetune from MaziyarPanahi now has the top position: <https://huggingface.co/MaziyarPanahi/calme-2.4-rys-78b>

**Additional note:**

A paper on the technique is currently being written. Currently, all four top models on the leaderboard are based on the RYS method. Special thanks to my wife, for putting up with me coding in the basement for too many evenings and weekends for months!

# Open LLM Leaderboard

---

## □ Tasks

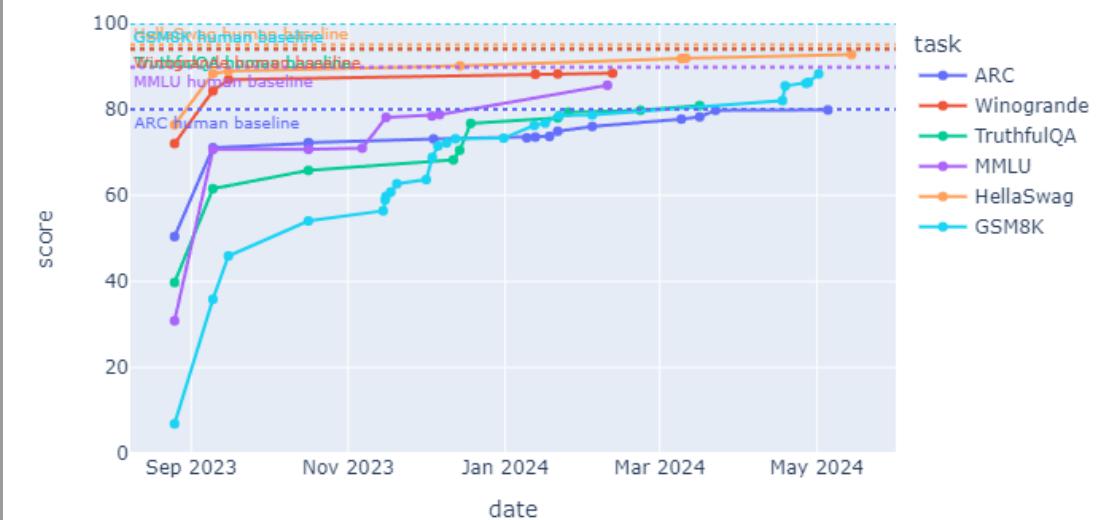
- AI2 Reasoning Challenge: a set of grade-school science questions.
- HellaSwag: a test of commonsense inference, which is easy for humans (~95%) but challenging for SOTA models.
- MMLU: a test to measure a text model's multitask accuracy.
- TruthfulQA: a test to measure a model's propensity to reproduce falsehoods commonly found online.
- Winogrande: an adversarial and difficult Winograd benchmark at scale, for commonsense reasoning.
- GSM8k: diverse grade school math word problems to measure a model's ability to solve multi-step mathematical reasoning problems.

# Open LLM Leaderboard

Average of Top Scores and Human Baseline Over Time (from last update)



Top Scores and Human Baseline Over Time (from last update)



# Summary

---

## □ LLM from Scratch

- LLM architecture (Encode-decoder, self-attention...)
- Pretraining
- Self-supervised Finetune
- RLHF (PPO and DPO...)
- Evaluation Benchmarks

Thanks!