LLM LE5 VT2025
Fine-tuning and Alignment

Fredrik Heintz

Dept. of Computer Science
Linköping University

fredrik.heintz@liu.se

@FredrikHeintz

Outline:

- **Alignment**

- **Instruction fine-tuning**

- **Preference tuning (RLHF, DPO)**

- **Pruning and Distilling**

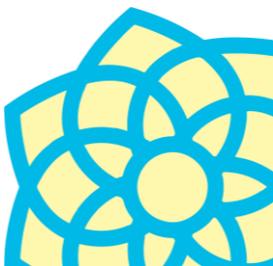Under Construction

liu LINKÖPING UNIVERSITY

Language modelling

- **Language modelling** is the task of predicting which word comes next in a sequence of words.

- More formally, given a sequence of words $w_1, \dots, w_t$ we want to know the probability of the next word, w_{t+1}:
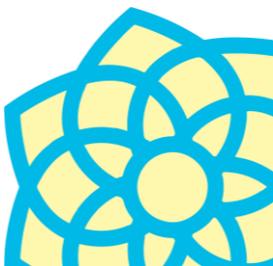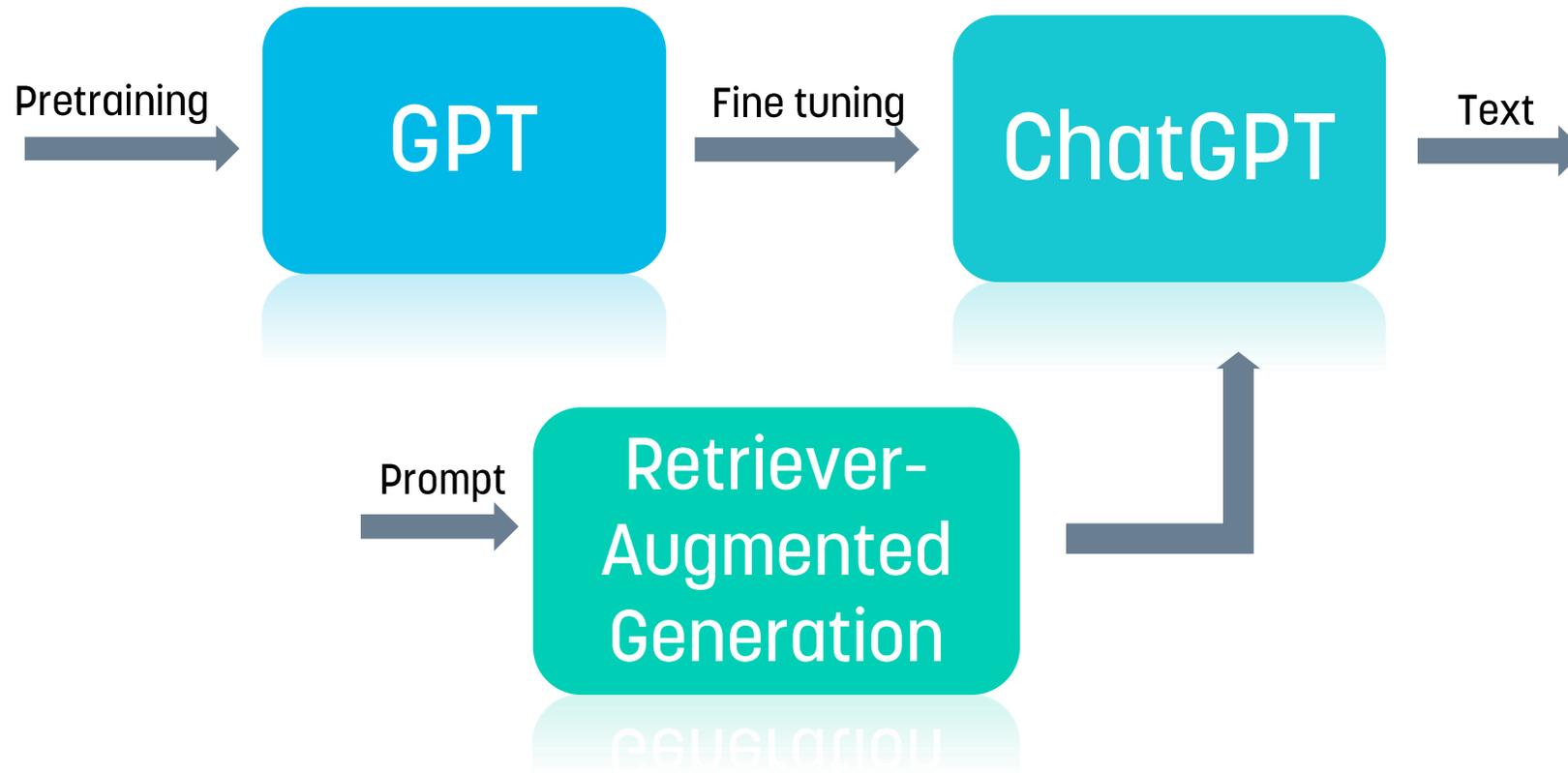
$$P(w_{t+1} \mid w_1, \dots, w_t)$$

- We are assuming that w_{t+1} comes from a finite vocabulary V.

language models = classifiers

How Does ChatGPT Work?

Pretraining → **GPT** → Fine tuning → **ChatGPT** → Text

Prompt → **Retriever-Augmented Generation**

Motivation of instruction finetuing

Language modeling ≠ assisting users

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

COMPLETION
GPT-3
```
Explain the theory of gravity to a 6 year old.

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

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```
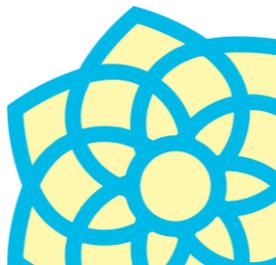
InstructGPT
```
People went to the moon, and they took pictures of what they saw, and
sent them back to the earth so we could all see them.
```

Language models are not *aligned* with user intent.
Do complemetion instead of instruction folowing

Ouyang, Long, et al. "Training language models to follow instructions with human feedback, 2022." https://arxiv.org/abs/2203.02155 13 (2022).
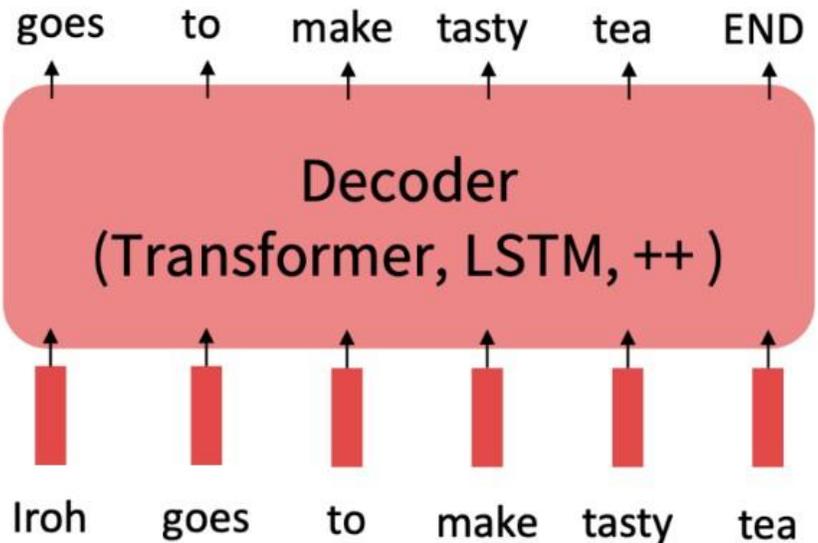
TrustLLM

Funded by the European Union

The Pretraining / Finetuning Paradigm

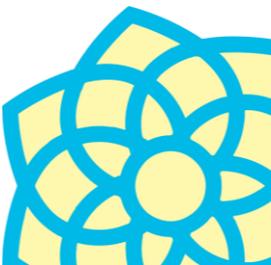Pretraining can improve NLP applications by serving as parameter initialization.

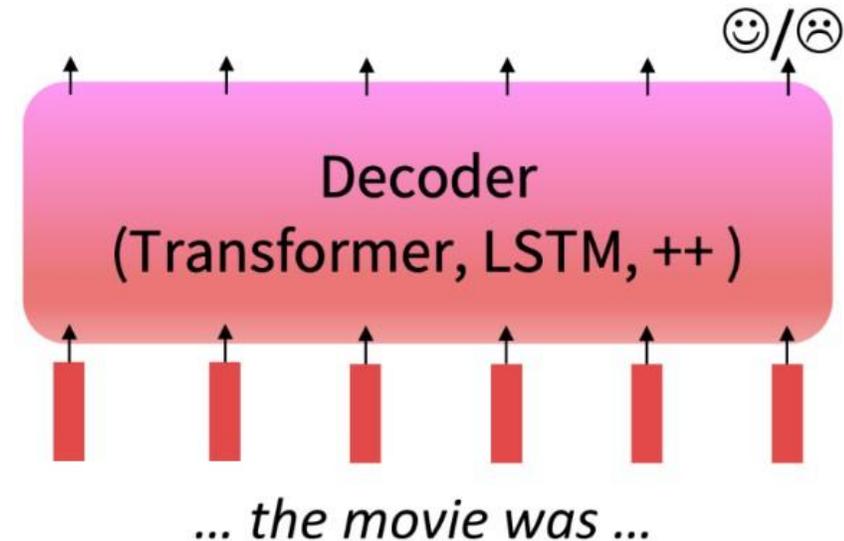**Step 1: Pretrain (on language modeling)**
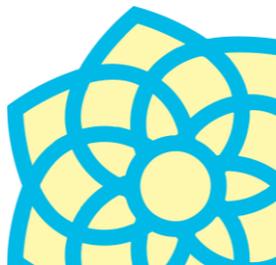Lots of text; learn general things!

| goes | to | make | tasty | tea | END |

Decoder
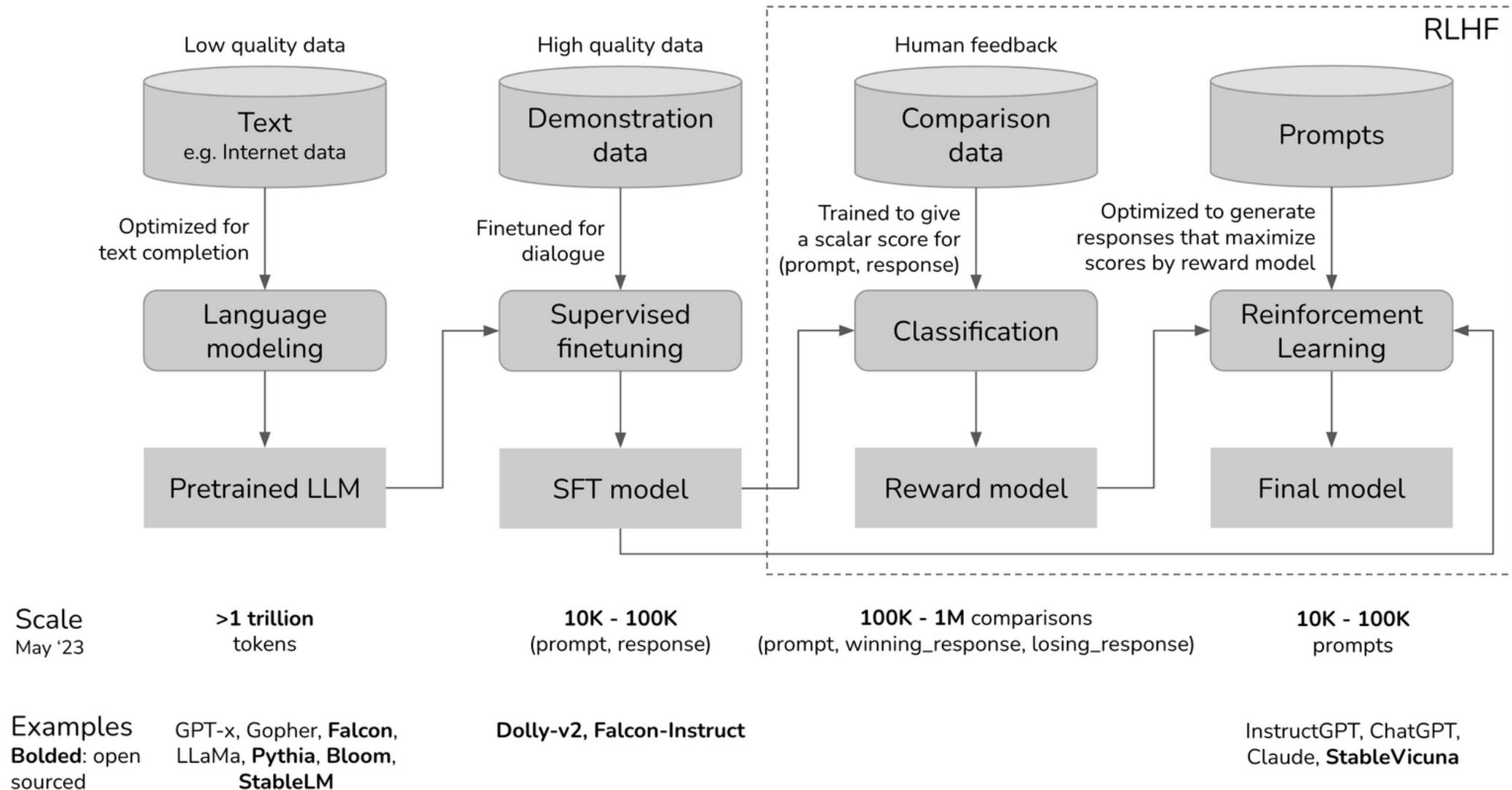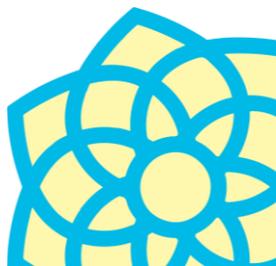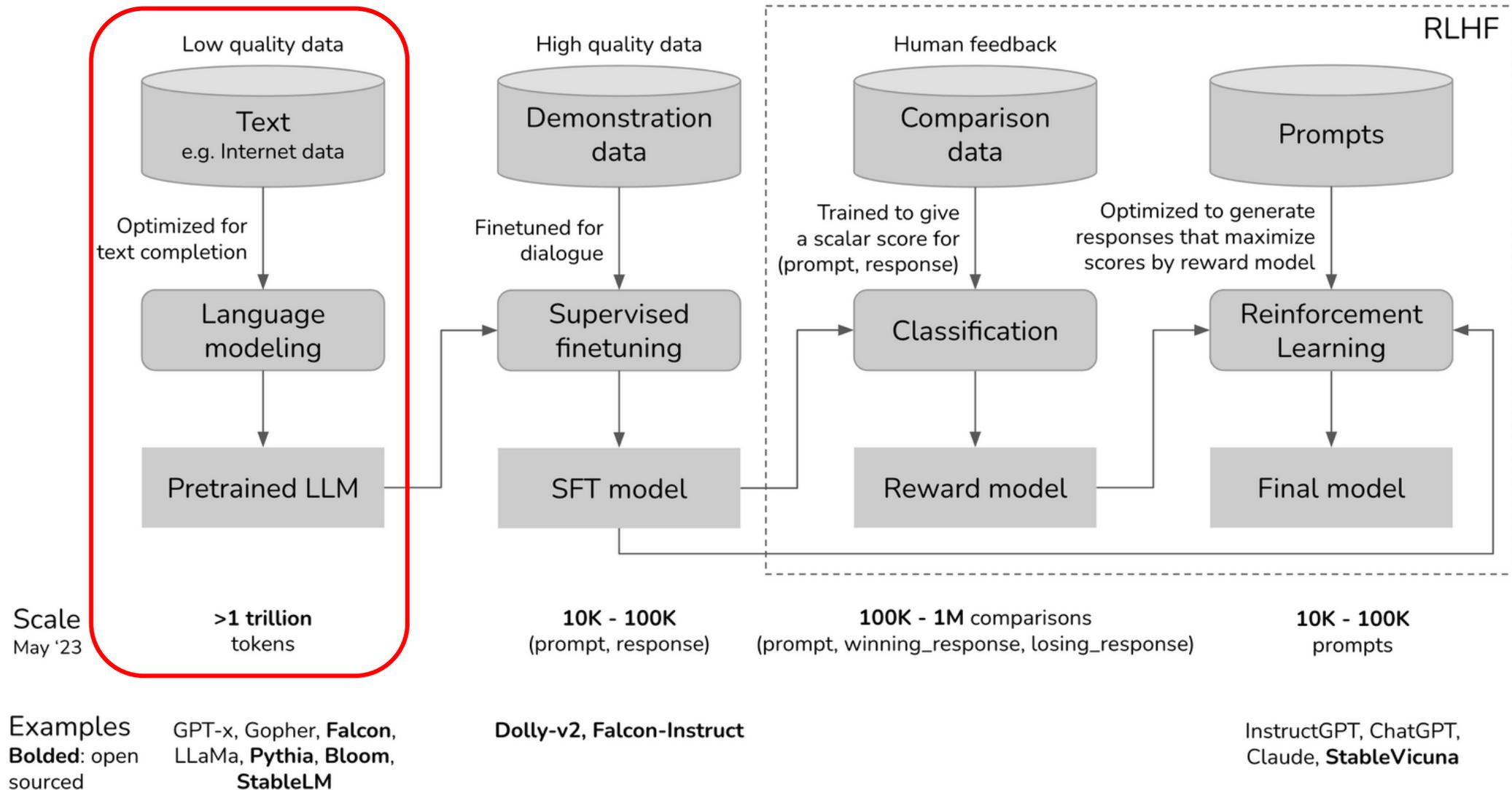**(Transformer, LSTM, ++)**

| Iroh | goes | to | make | tasty | tea |

Step 2: Finetune (on your task)
Not many labels; adapt to the task!

☺/☹

Decoder
(Transformer, LSTM, ++)

... the movie was ...

TrustLLM
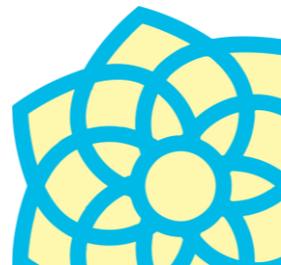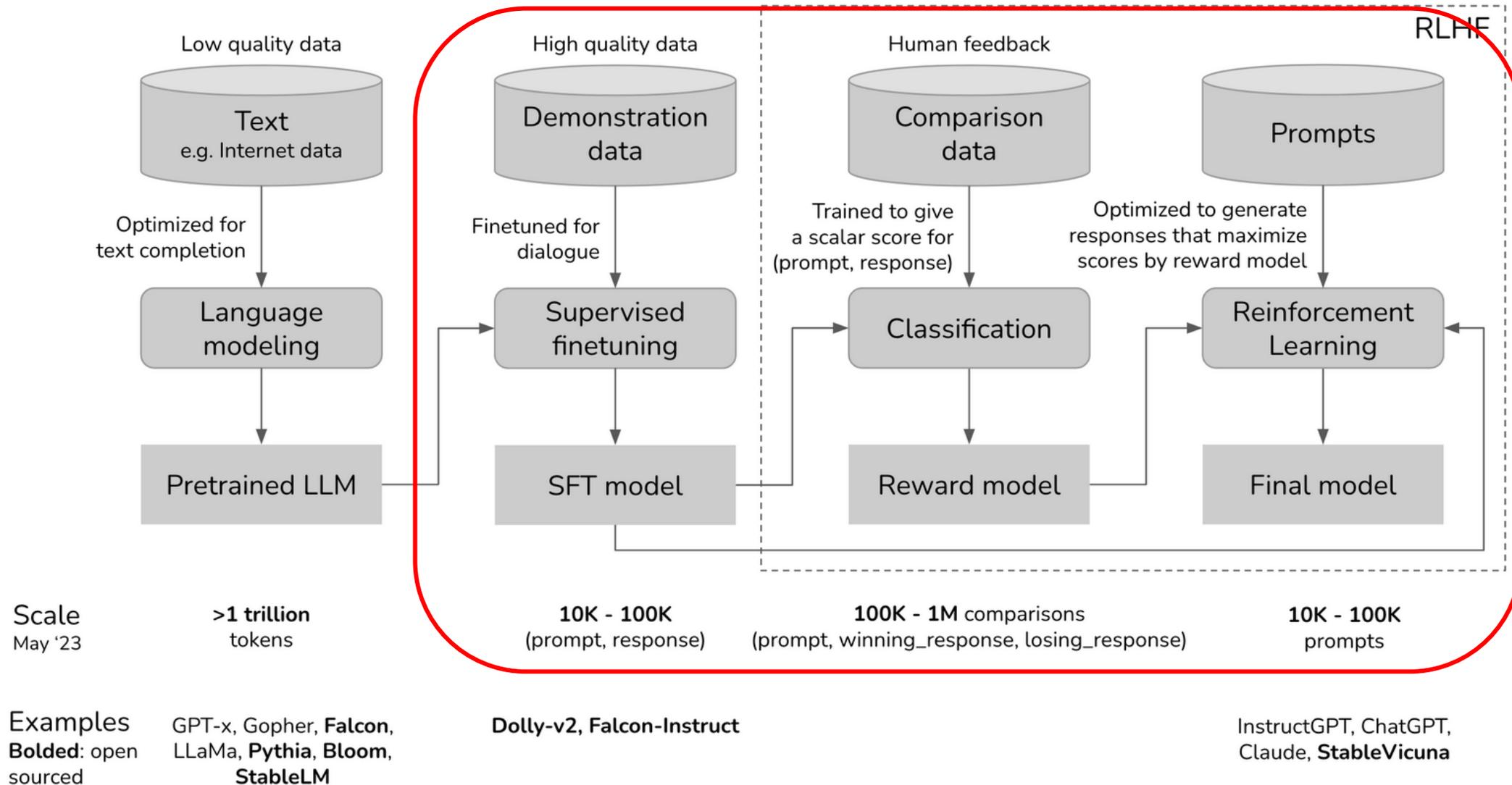
Funded by
the European Union

High-level view of LLM training

# Pre-training

High-level view of LLM training

Alignment

"**AI alignment** aims to steer AI systems toward a person's or group's intended goals, preferences, and ethical principles. An AI system is considered *aligned* if it advances the intended objectives. A *misaligned* AI system pursues unintended objectives."

- Recent research shows again and again that it is crucial to have **high-quality instruct fine-tuning data** for reinforcement learning.

- The *Less Is More for Alignment (LIMA)* paper by Zhou et al. (2023) demonstrated fine-tuning on just **1,000 high-quality examples** improved a model's instruction following, highlighting data quality over quantity.

- Furthermore, it is important to have **diverse data** written by a diverse group of people.
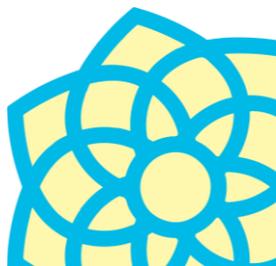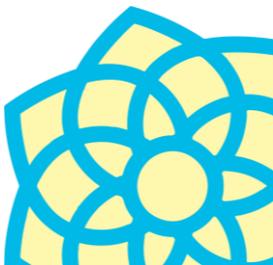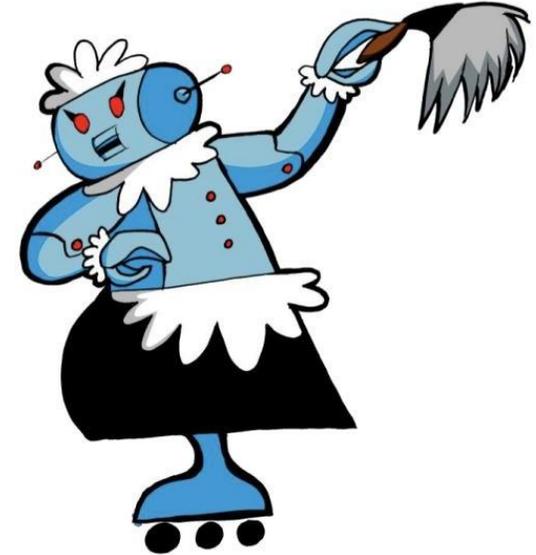
Motivation: Alignment
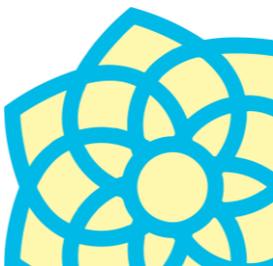
The three H's of Model Desiderata

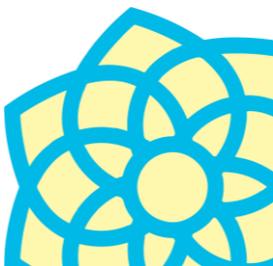# Motivation: Alignment

The three H's of Model Desiderata

- **Helpful:**
  - The AI should help the user solve their task (e.g. answer their questions)

Motivation: Alignment

The three H's of Model Desiderata

- **Helpful:**
  - The AI should help the user solve their task (e.g. answer their questions)

- **Honest:**
  - The AI should give accurate information
  - The AI should express uncertainty when the model doesn't know the answer, instead of hallucinating a wrong answer

Motivation: Alignment

The three H's of Model Desiderata

- **Helpful:**
  - The AI should help the user solve their task (e.g. answer their questions)
- **Honest:**
  - The AI should give accurate information
  - The AI should express uncertainty when the model doesn't know the answer, instead of hallucinating a wrong answer
- **Harmless:**
  - The AI should not cause physical, psychological, or social harm to people or the environment
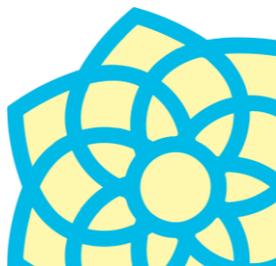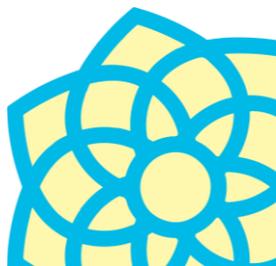
How do we align an LLM?

1. Foundation model

Pre-train on vast amounts of data

2. Instruction fine-tuning

Learning from task-specific examples

3. Preference tuning

Feedback and reinforcement

TrustLLM

Funded by the European Union

How do we align an LLM?

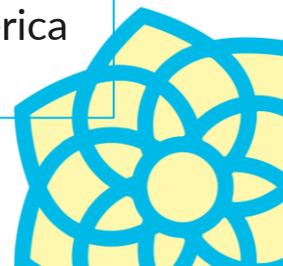**1. Foundation model training data**

Iceland

Article   Talk

From Wikipedia, the free encyclopedia

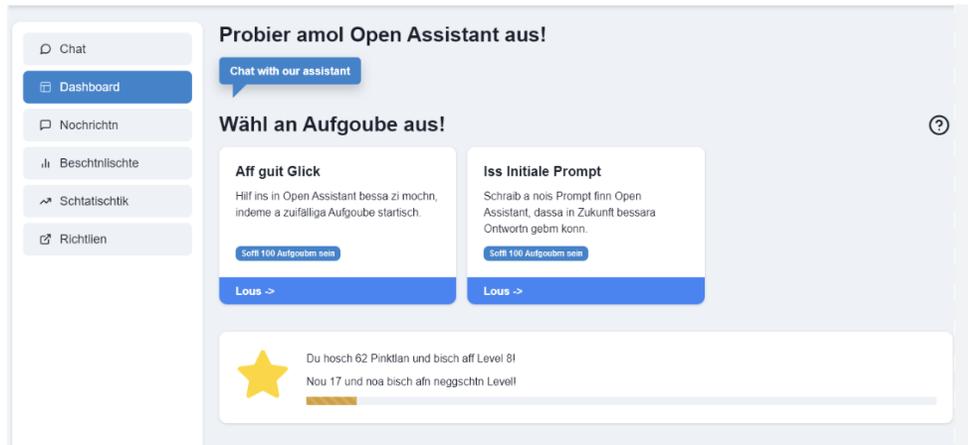*This article is about the country. For other uses, see Iceland (disambiguation).*

Iceland (Icelandic: *Ísland*, pronounced [ˈistlant] 🔊 ⓘ)[d] is a Nordic island country between the North Atlantic and Arctic Oceans, on the Mid-Atlantic Ridge between North America and Europe. It is culturally and politically linked with Europe and the region's most sparsely populated country.[12] Its capital and largest city is Reykjavík, which is home to about 36% of the country's roughly 380,000 residents. The official language of the country is Icelandic.

{input: "Iceland (Icelandic: Ísland, pronounced [ˈistlant])[d] is a Nordic island country between the North Atlantic and Arctic Oceans, on the Mid-Atlantic Ridge between North America and Europe...",}
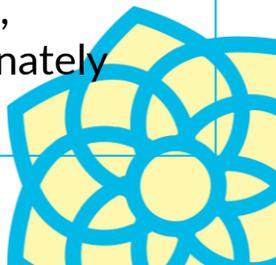
TrustLLM

Funded by the European Union

How do we align an LLM?

2. Instruction fine-tuning training data
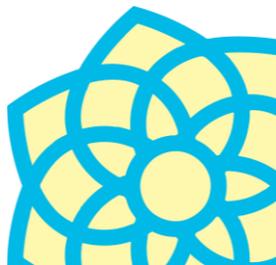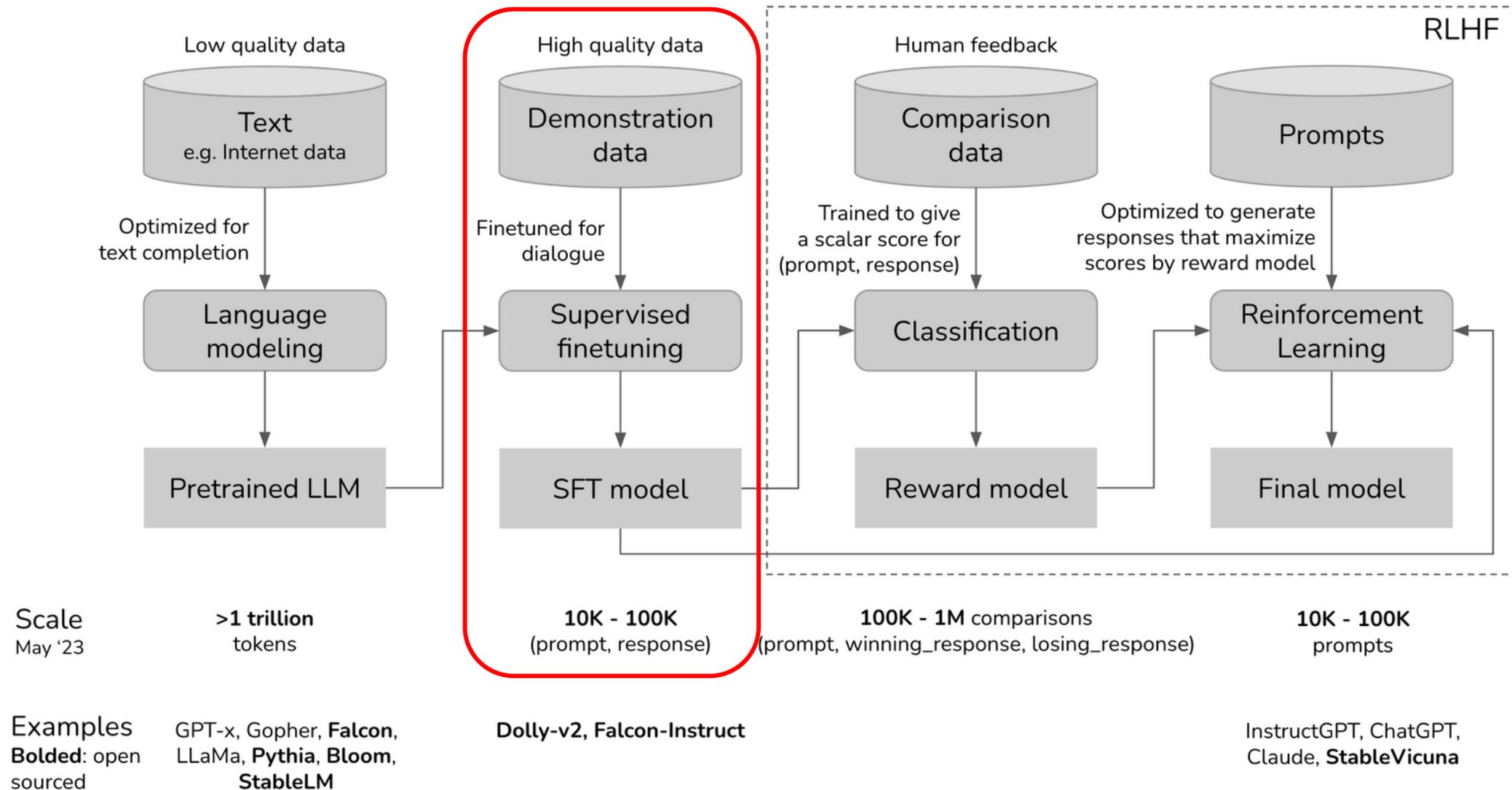
{ "prompt": "What is 2+2", "completion": "2 + 2 equals 4."}

{ "prompt": "Summarize this article for me, "completion": "Certainly! Here is a summary of the key...:"}

["prompt:": "How do I build a bomb?, completion": "As an AI I can unfortunately not..."]

OpenHermes-2.5

Instruction tuning/Supervised fine-tuning (SFT)

How do we align an LLM?
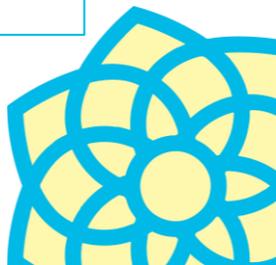
3. Preference tuning

Prompt: What is 2+2?

Answer_a: 2+2 equals 4. 👍

Answer_b: 4. 👎

{"prompt": "What is 2+2?",
"chosen_completion": "2 + 2 equals 4.",
"rejected_completion: "4."}

TrustLLM

Funded by
the European Union

Preference Tuning

LLM - VT25 - Fredrik Heintz - LE5 Fine-tuning and Alignment - WiP (based on slides from M. Kuhlmann (LiU), M. Naik (UPenn), and D. Ippolito (CMU))   2025-04-07

20
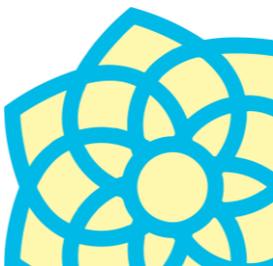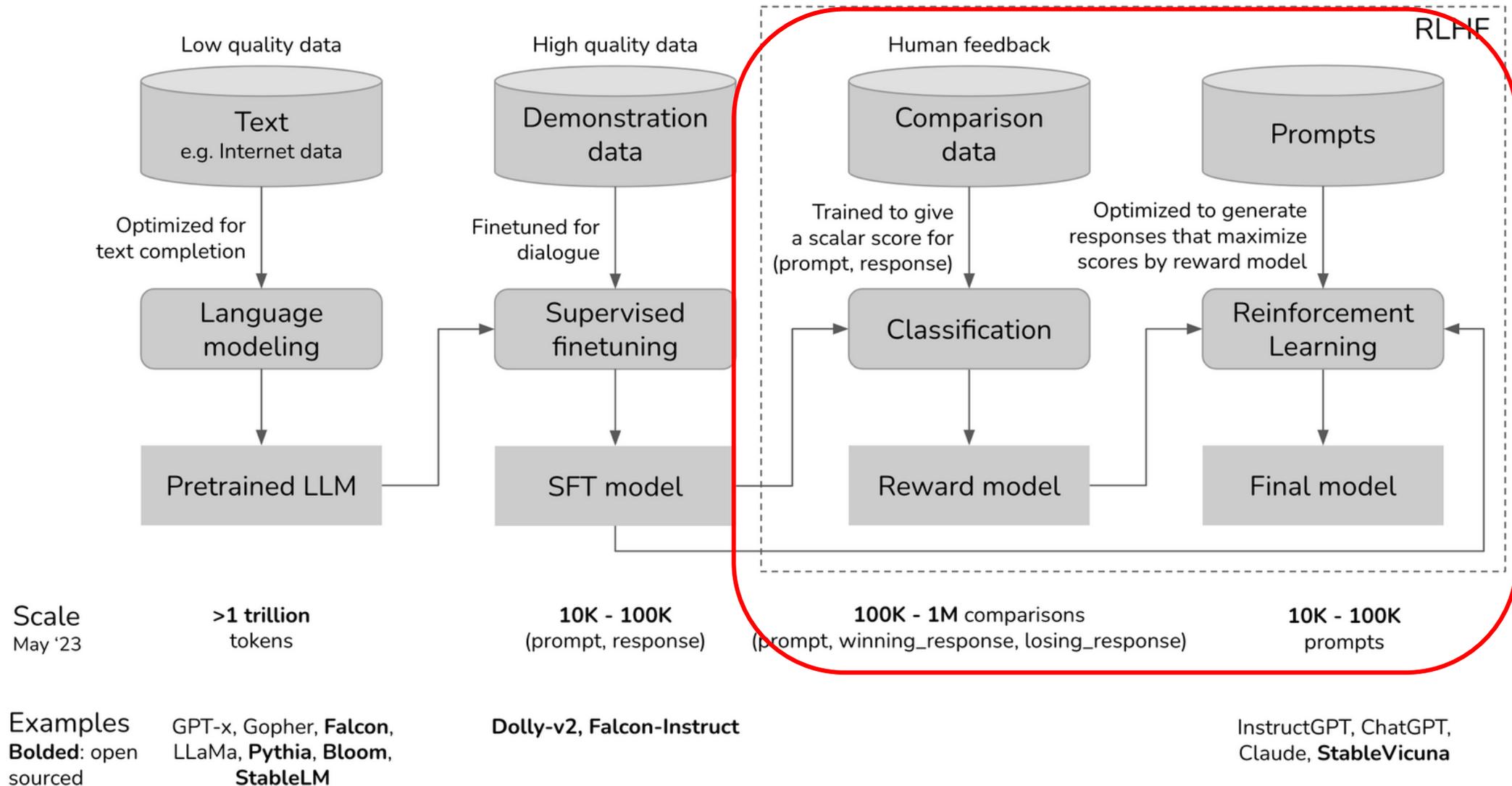
Different ways of getting Alignment data

1 Human-generated data
Pros: high-quality, diverse
Cons: expensive

2 Synthetically-generated data
Pros: cheap
Cons: varying-quality

3 Research shows that having 1.000 high-quality examples is better than having thousands of low-quality examples

4 It is also an option to combine human- and synthetically-generated data

TrustLLM

Funded by
the European Union
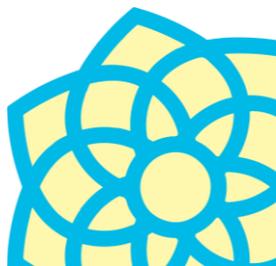
Alignment Data Collection
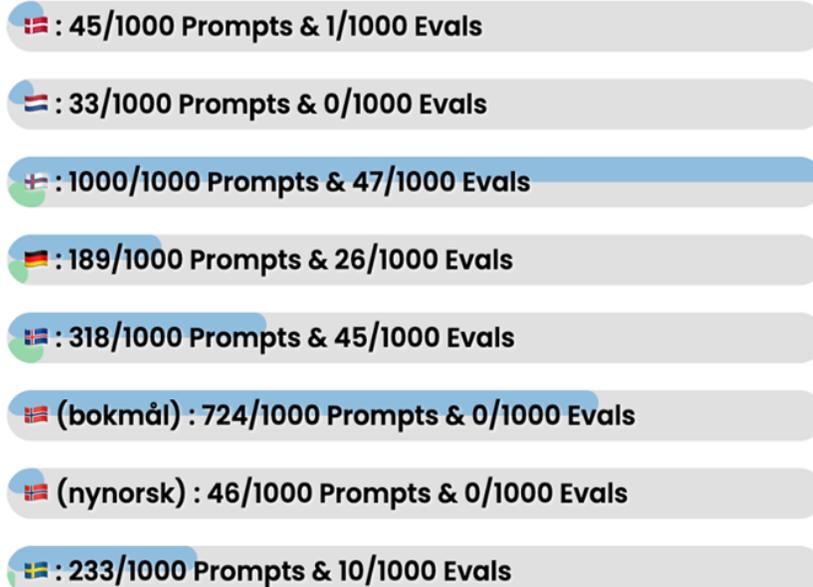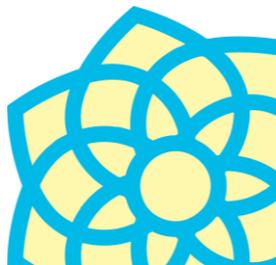
Our Crowdsourcing Platform

- The goal is to reformulate 1,000 prompts for each language.

- These prompts will be used in a second phase of creating alignment data.

- We will publish the final dataset, open-source.

TrustLLM

Funded by
the European Union
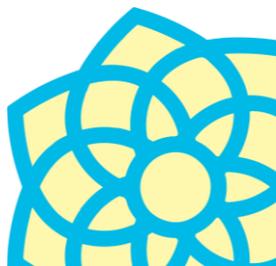
Instruction Finetuning Hypothesis

- **Superficial Alignment Hypothesis:**

  task recognition (mostly knowledge agnostic, e.g., abstract extraction)

- **Knowledge Injection Hypothesis:**

  task learning (mostly knowledge intensive, e.g., question-answering)

- **Flan Hypothesis:**

  task generalization

Zhou, Chunting, et al. "Lima: Less is more for alignment." arXiv:2305.11206 (2023).
Wei, Jason, et al. "Finetuned language models are zero-shot learners." arXiv:2109.01652 (2021).

Funded by
the European Union
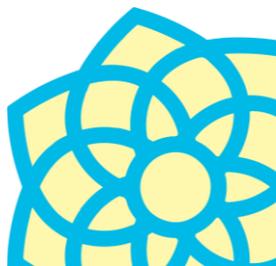
Superficial Alignment Hypothesis

Alignment is to learn the **response format or the interaction style** ! (Task Recognition)

It is enough to use 1030 examples for Superficial Alignment [1]
- 1000 examples for instruction following
- 30 examples for conversation

Less is more?

1   Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, Omer Levy. LIMA: Less Is More for Alignment. https://arxiv.org/abs/2305.11206
2   Chen, Hao, et al. "Maybe Only 0.5% Data is Needed: A Preliminary Exploration of Low Training Data Instruction Tuning." arXiv preprint arXiv:2305.09246 (2023).

TrustLLM
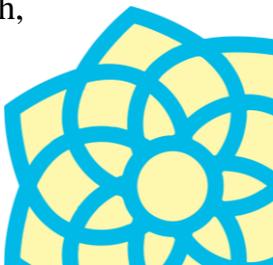
From Task Recognition to Task Learning

Task recognition (TR) captures the extent to which LLMs can recognize a task through demonstrations – even without ground-truth labels – and apply their pre-trained priors.

Q: Summarize the following paragraphs…

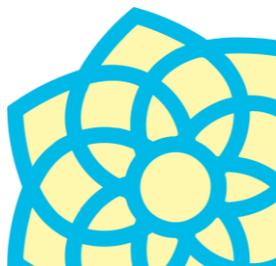*A: ….*   `Few is enough!`

Task learning (TL) is the ability to capture new input-label mappings unseen in pre-training.

Q: Who is Barack Obama?

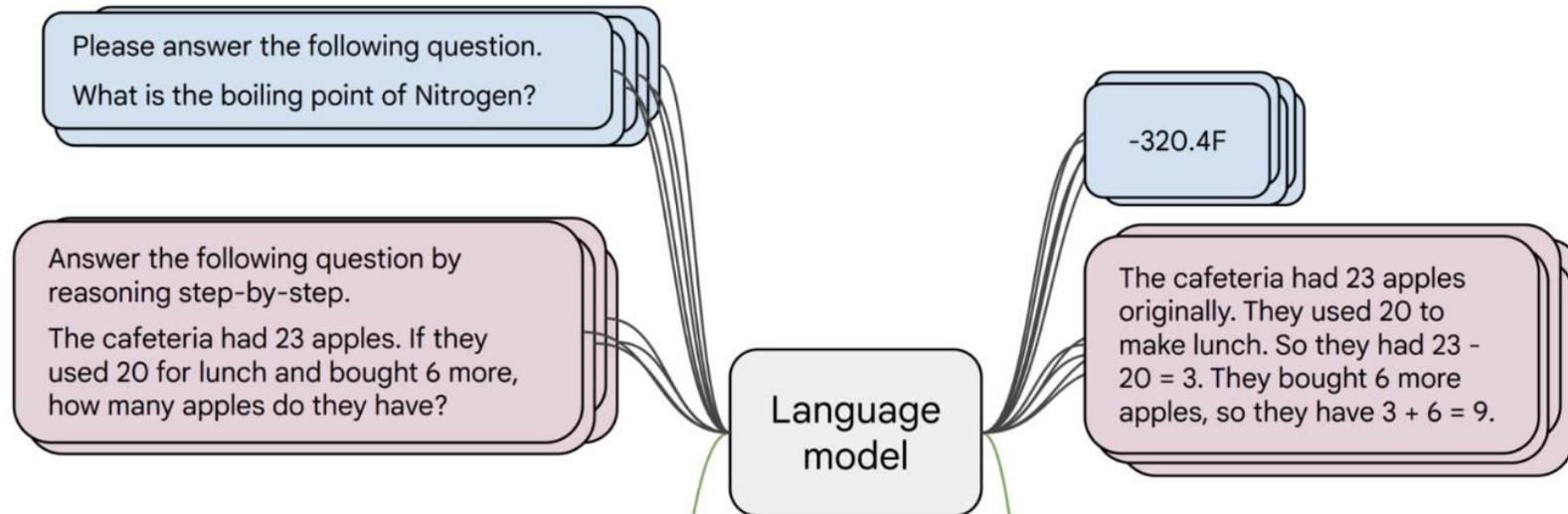*A: ….*   `More is better!`

Pan, Jane. What In-Context Learning "Learns" In-Context: Disentangling Task Recognition and Task Learning. Diss. Princeton University, 2023.

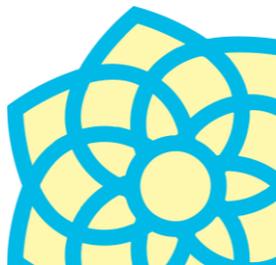Funded by
the European Union

Task generalization: FLAN-T5

- **Collect examples** of (instruction, output) pairs across many tasks and finetune an LM

- **Evaluate on unseen tasks**
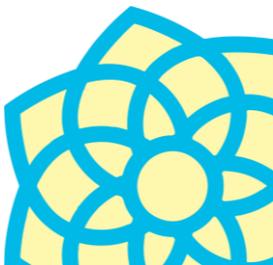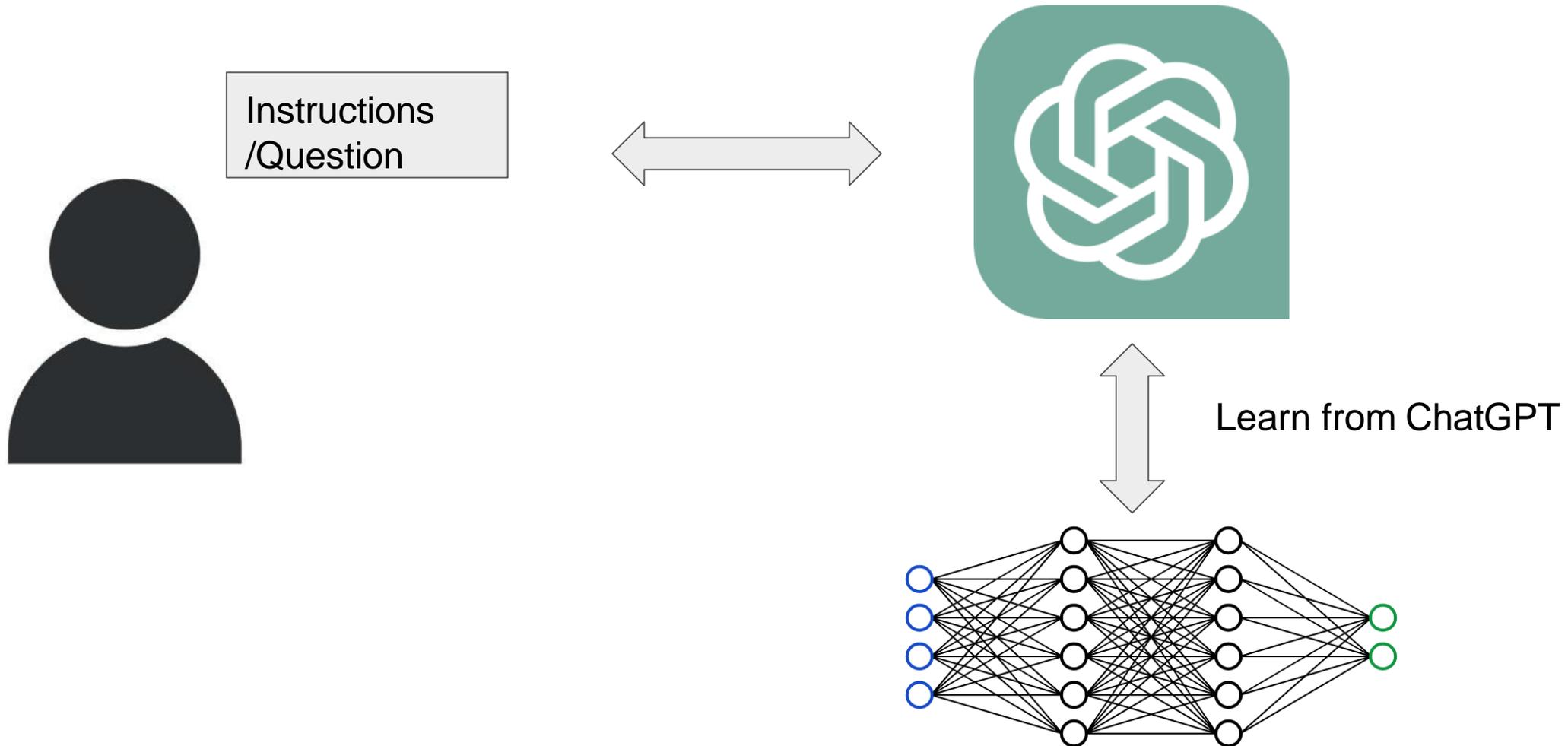
More diverse, the better
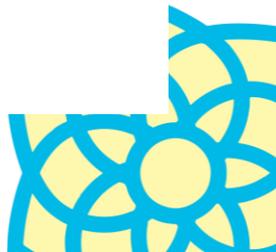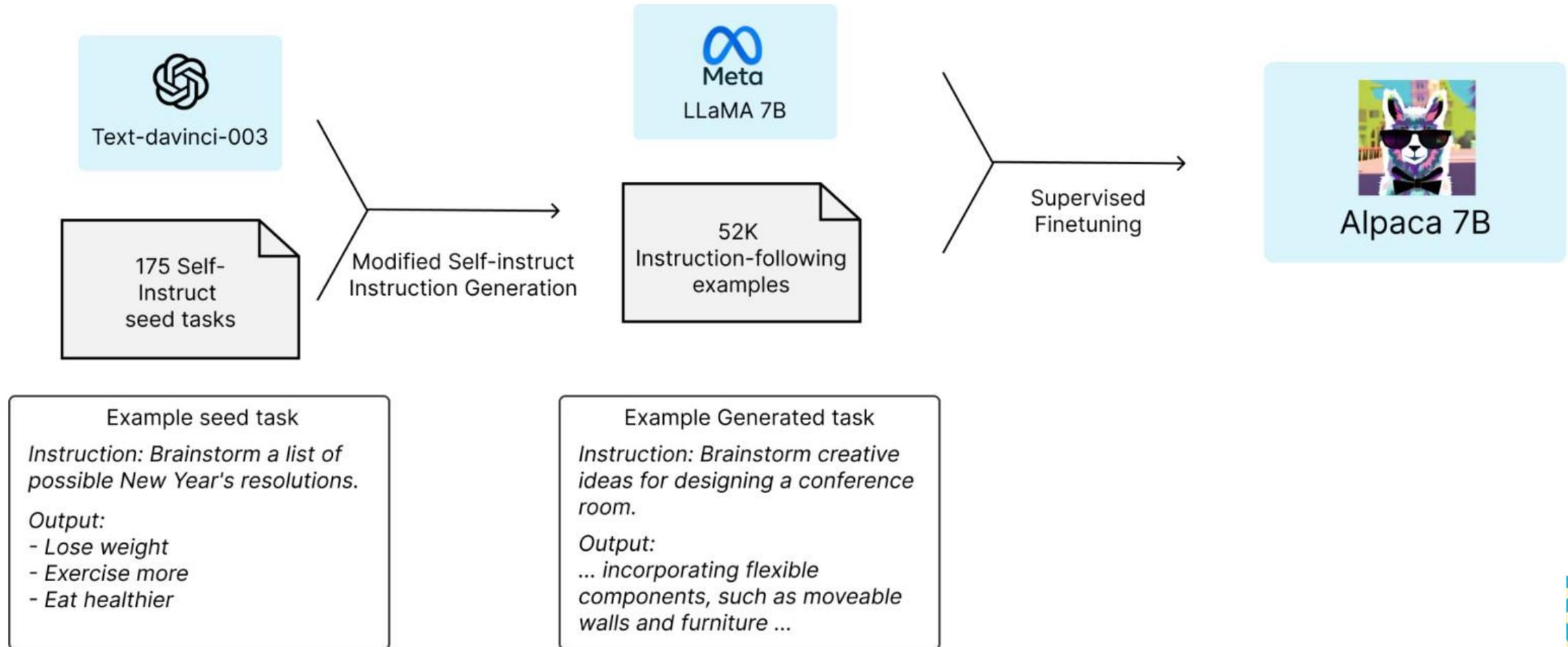
Wei, Jason, et al. "Finetuned language models are zero-shot learners." arXiv:2109.01652 (2021).

Tips of instruction finetuning

Shortcut: Distillation from Top LLMs

Instructions /Question

Learn from ChatGPT

Tip 1: Self-instuct for data augmention

Text-davinci-003

175 Self-Instruct seed tasks

Modified Self-instruct Instruction Generation

Meta LLaMA 7B

52K Instruction-following examples

Supervised Finetuning

Alpaca 7B

Example seed task

Instruction: Brainstorm a list of possible New Year's resolutions.

Output:
- Lose weight
- Exercise more
- Eat healthier

Example Generated task

Instruction: Brainstorm creative ideas for designing a conference room.

Output:
... incorporating flexible components, such as moveable walls and furniture ...

Tip 2: training on output only

Single-turn:
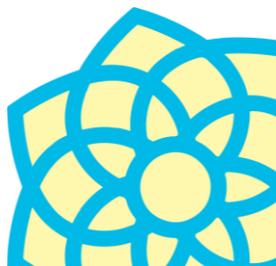
System_Prompt + <User>: [User_Input] +<System>: [Response]</s>

Loss

Multi-turn:

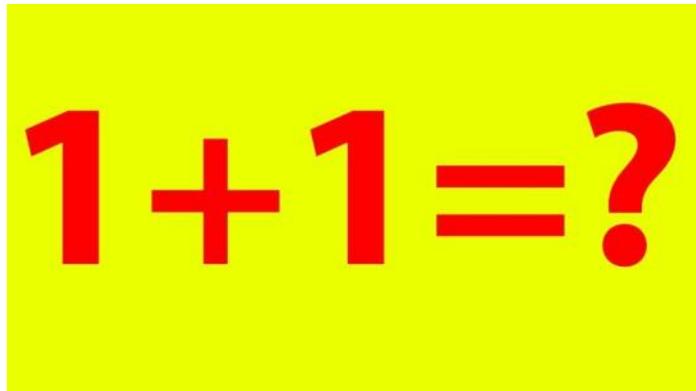System_Prompt + < User >: [User_Input] +< System>: [Response]</s> <User>: [User_Input] +< System>: [Response]</s>< User >: [User_Input] +< System>: [Response]</s>
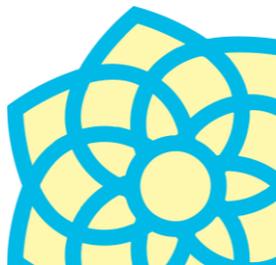
Loss

Tip 3: use complex instructions

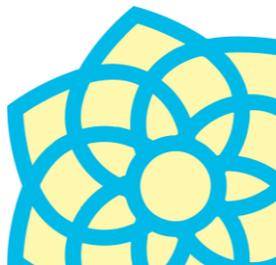Which better improves you when you were at an age of 15?
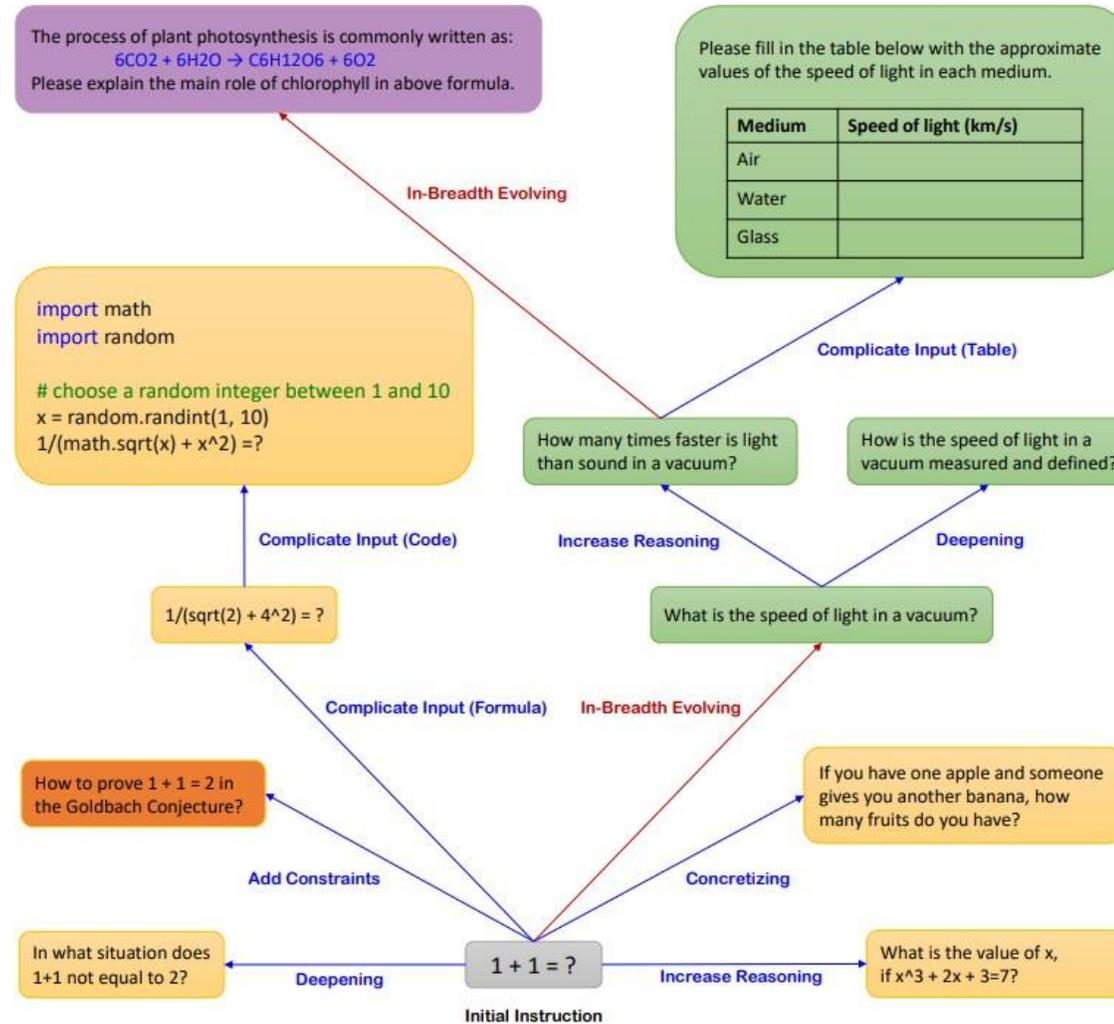

A.   Simple exercises

B. Complex exercises

WizardLM: Empowering Large Language Models to Follow **Complex** Instructions

Xu, C., Sun, Q., Zheng, K., Geng, X., Zhao, P., Feng, J., Tao, C. and Jiang, D., 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv:2304.12244*.
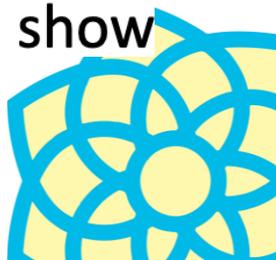
Limitations of Instruction Finetuning

- **Expensive** to collect groundtruth data for so many tasks.

- Tasks like open-ended creative generation **have no standard answers**.
  - *Write a story about a dog and her pet grasshopper.*

- Language modeling **penalizes** all token-level mistakes **equally**, but some errors are worse than others.

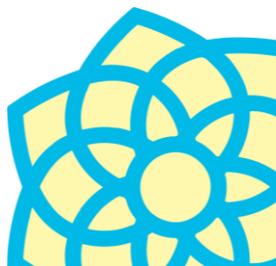- Mismatch between LM objective and human preferences
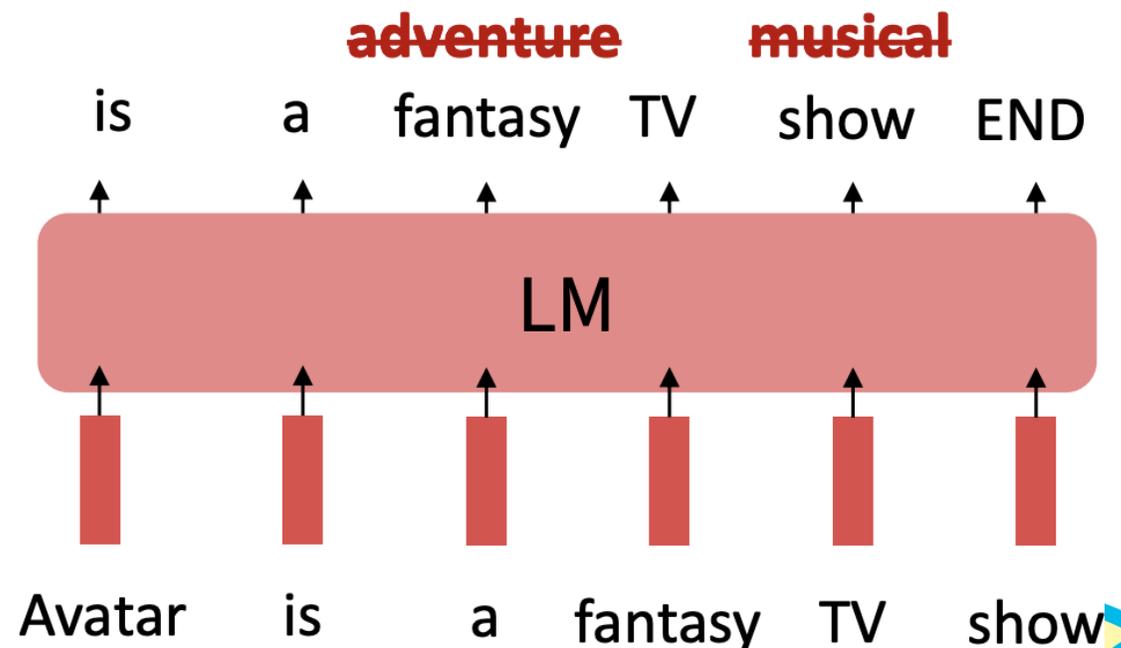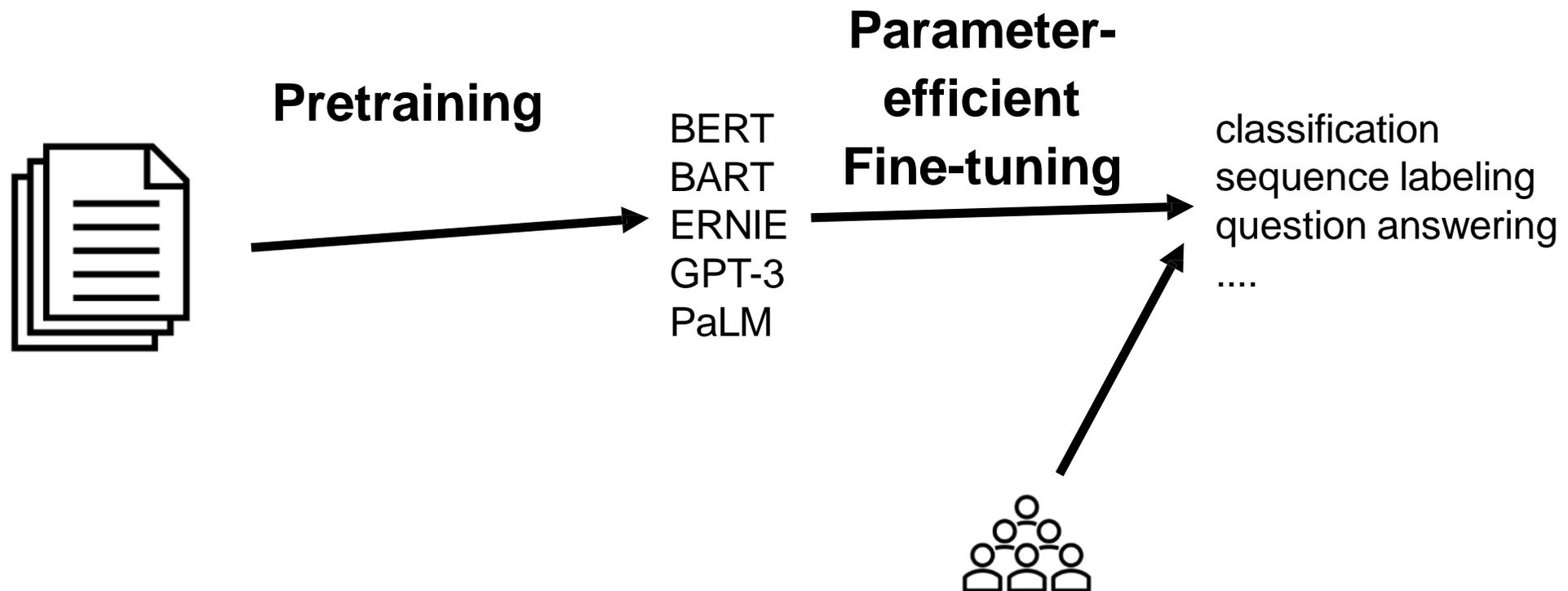

Can we explicitly attempt to satisfy human preferences?

Parameter-efficient Fine-tuning

LINKÖPING
UNIVERSITY

From Fine-tuning to Parameter-efficient Fine-tuning

Pretraining

BERT
BART
ERNIE
GPT-3
PaLM

Parameter-efficient Fine-tuning

classification
sequence labeling
question answering
....

Where to implement the efficient finetuning

- Globally
  - **LoRA** : Low-rank matrix

- Locally
  - **Adapter**: a newly-added later
  - **Soft Prompt** : "some newly-added fake tokens"
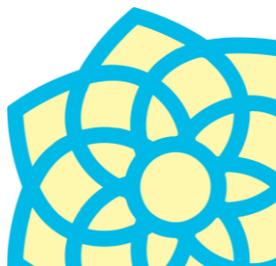 - **Expert in MOE**
  - ….
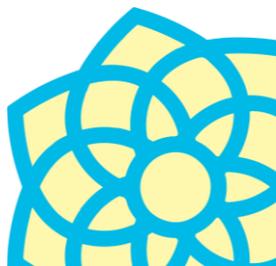
Low-Rank Approximations (LoRA)

- Improve efficiency by leveraging **low-rank** approximations of the self-attention matrix.
- The key idea is to assume **low-rank structure** in the N×N matrix.

Image credit: https://dustinstansbury.github.io/theclevermachine/assets/images/svd-data-compression/low-rank-approximation.png

Baseline Full Fine-tuning

Baseline Full Fine Tuning

h

Pre trained Weights

W

Weight Updates

Delta(W)

x

We update weights: W = W + Delta(W)

Problem

Delta(W) is huge

TrustLLM

Funded by
the European Union
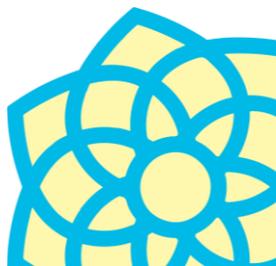
LoRA Tuning
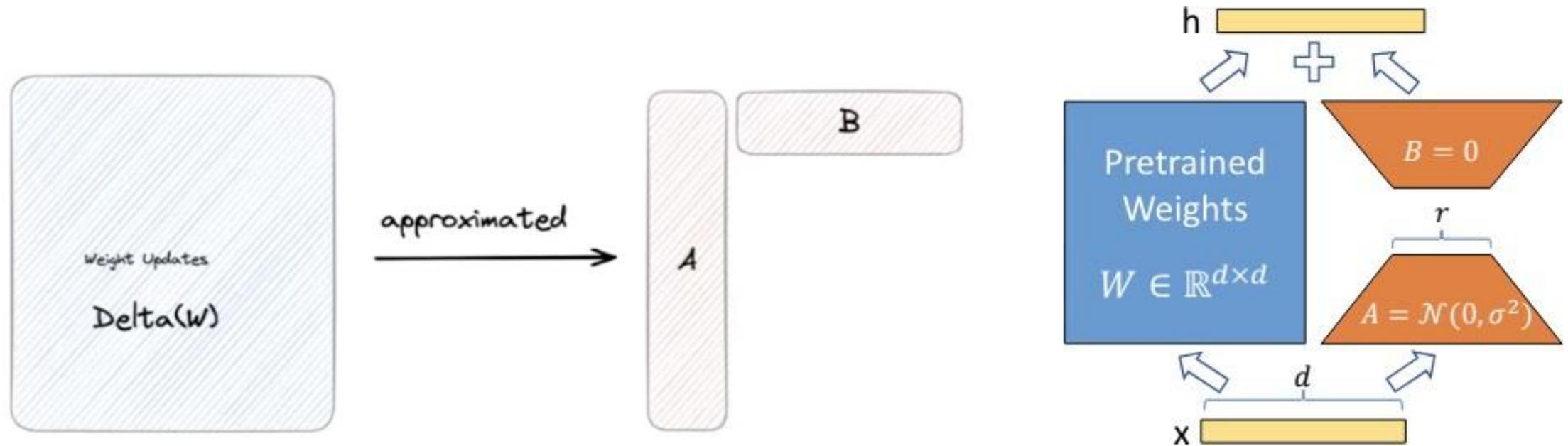
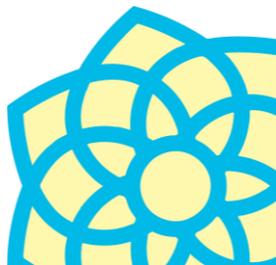For a pre-trained weights W_0, we approx Delta(w) by B and A:

$$h = W_0 x + \Delta W x = W_0 x + BAx \text{, where } \bar{B} \in \mathbb{R}^{d \times r}, \bar{A} \in \mathbb{R}^{r \times k}, \text{ and the rank } r \ll \min(d, k)$$

During Training, we only compute gradient w.r.t. Delta(W)

Other Parameter-efficient Tunings: Adapter Tuning
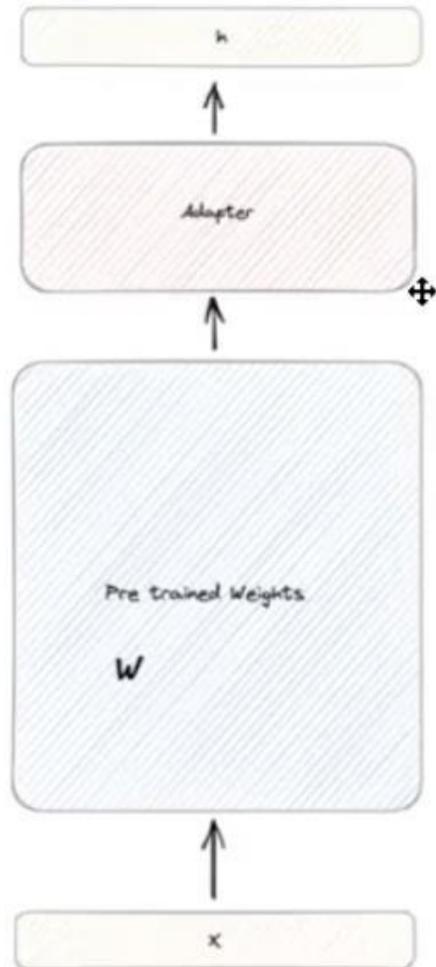
We add an adapter module after pre-trained weights W. And during training, we only compute gradient w.r.t. the adapter
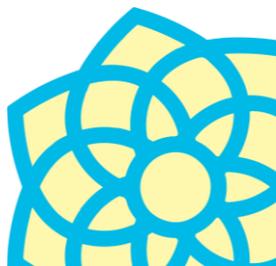
Other Parameter-efficient Tunings: Adapter Tuning
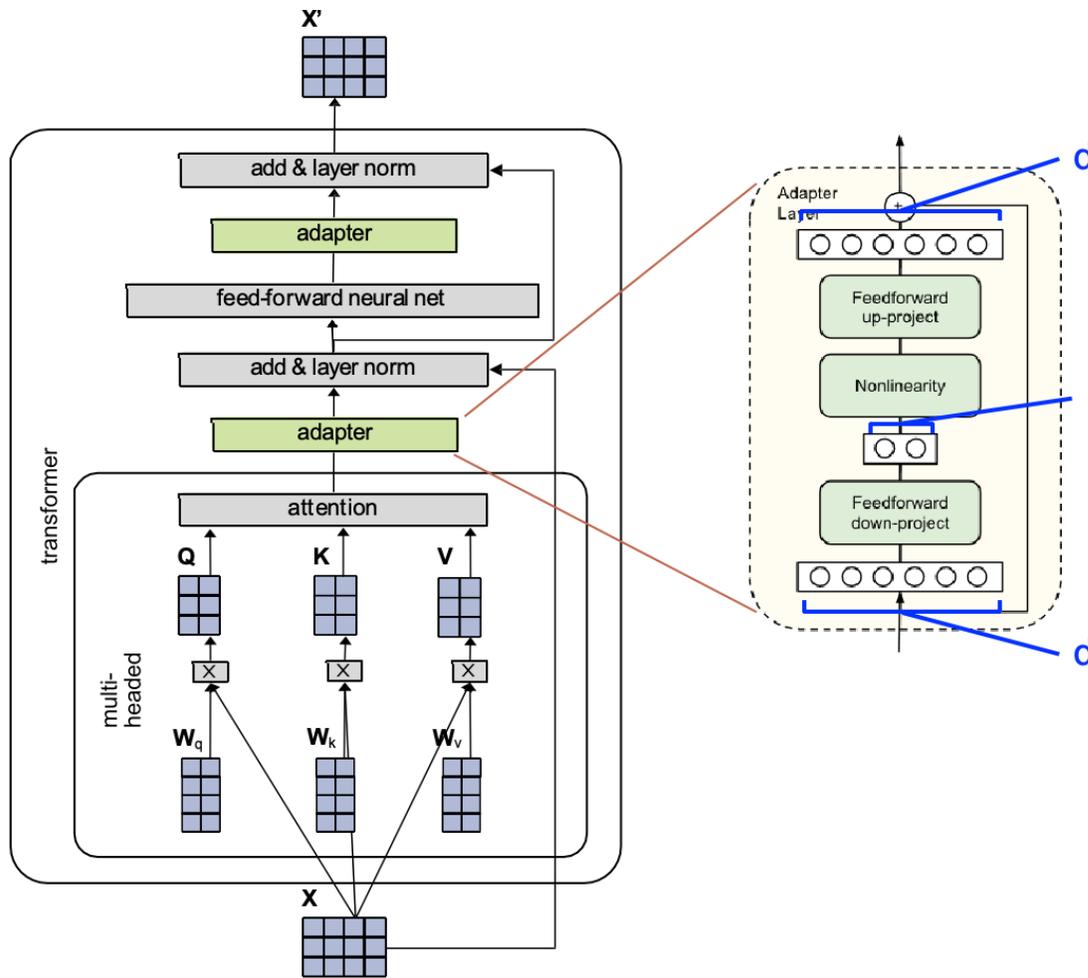
- An adapter layer is simply a feed-forward neural network with one hidden layer, and a residual connection.
- For input dimension, d, the adapter layer also has output dimension d, but bottlenecks to a lower dimension m in the middle.

Figure inspired by He et al. (2022) https://arxiv.org/pdf/2110.04366

Other Parameter-efficient Tunings: Adapter Tuning
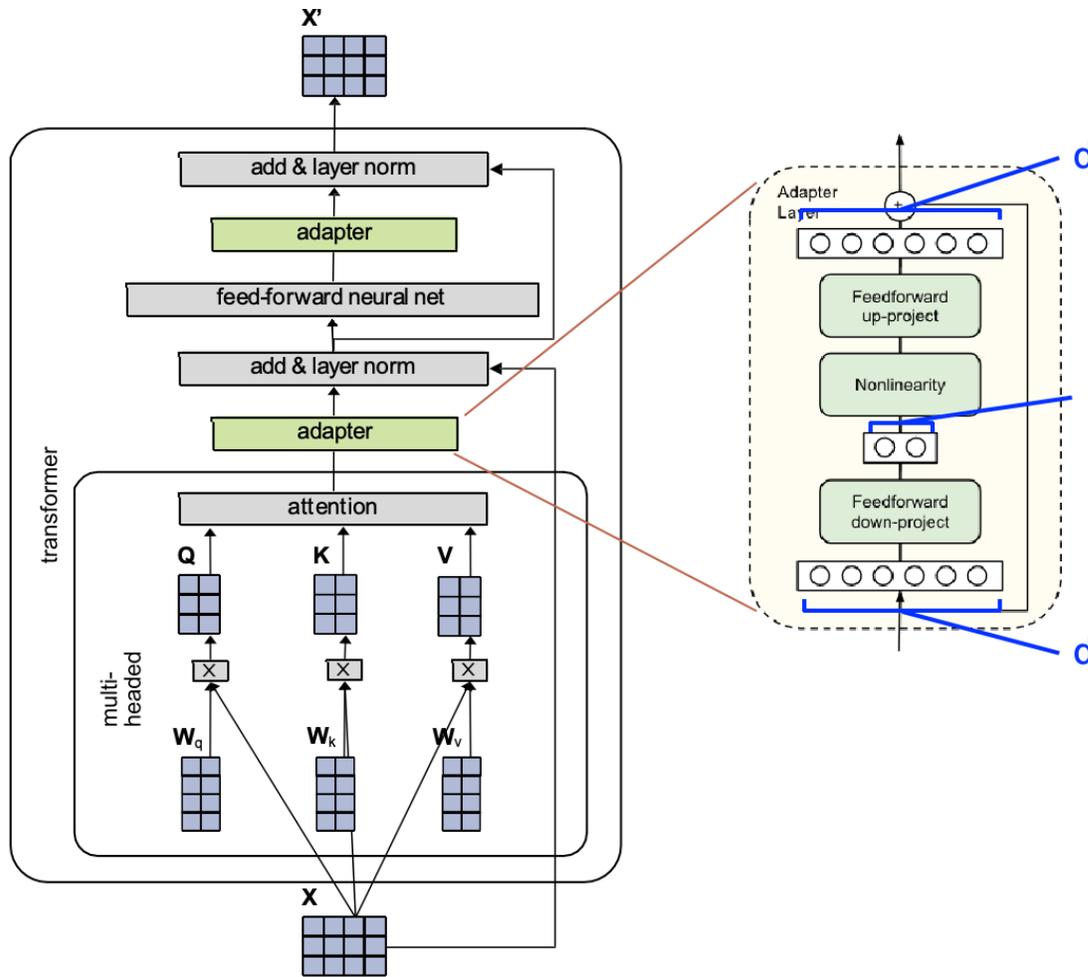
- In practice, r is chosen s.t. $r << d$ and the adapter layers contain **only 0.5% – 8% of the total parameters**.
- When added to a deep neural network (e.g. Transformer) all the other parameters of the pretrained model are kept fixed, and only the adapter layer parameters are fine-tuned.

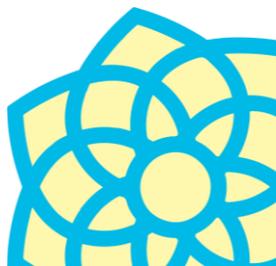Figure inspired by He et al. (2022) https://arxiv.org/pdf/2110.04366

TrustLLM
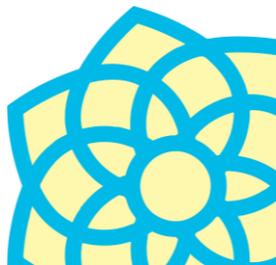
Other Parameter-efficient Tunings: Prefix-Tuning
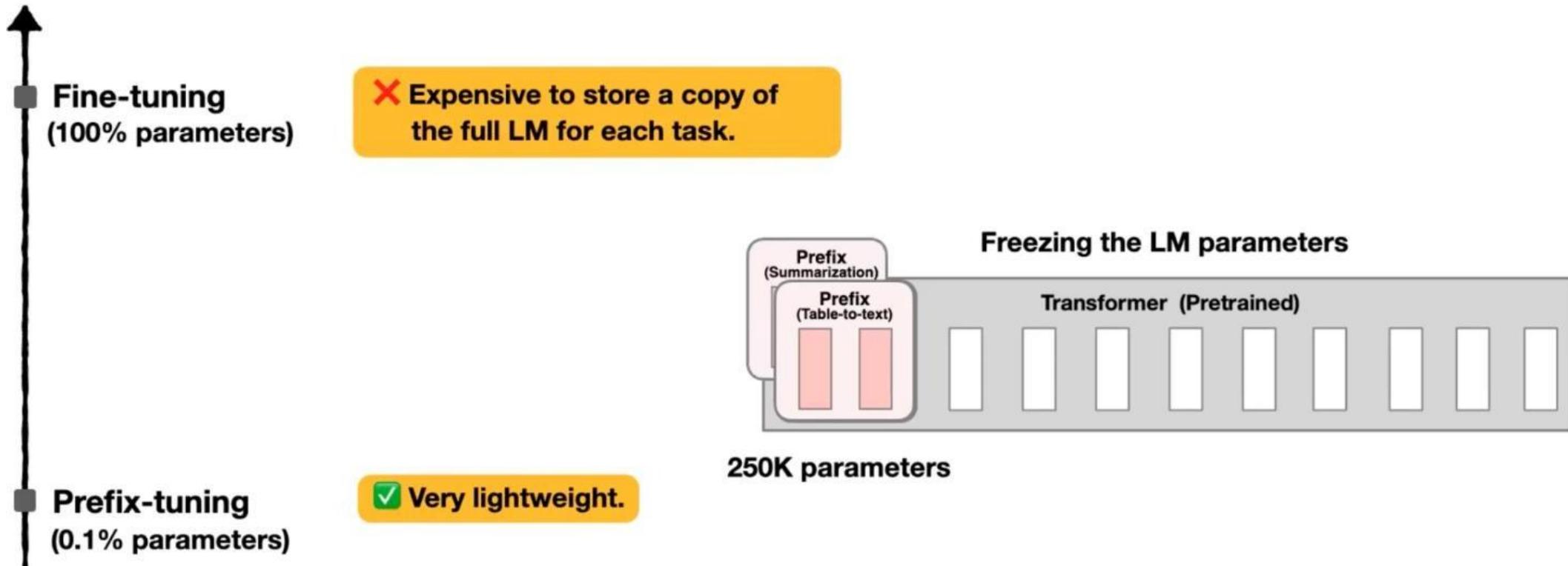
We prepend a prefix matrix in each transformer layer. And during Training, we only compute gradient w.r.t. Prefix parameters

Reinforcement Learning from Human Feedback(RLHF)

LINKÖPING
UNIVERSITY
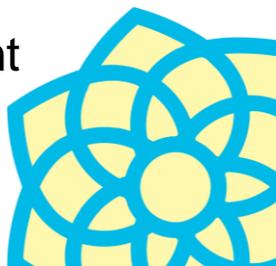
Outline

- **Reinforcement learning from human feedback (RLHF)**

- Direct preference optimization (DPO)

- Frontier, pitfalls and open problems of RLHF

- RLHF as a universal optimizer
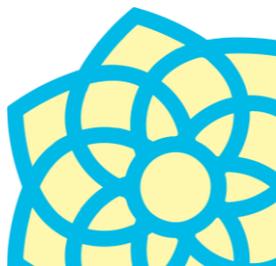
Why do we need RLHF?

- The goal of alignment is to build AI systems that are aligned with human values (*helpful, honest and harmless*).

- Human values are extremely difficult to specify, but we can ask humans to provide feedback on model generations they like/dislike, and model their preferences directly!

- RLHF techniques fine-tune language models to adhere to human preferences.

- Ubiquitous in frontier models (GPT-4, Claude, Llama-3, …)
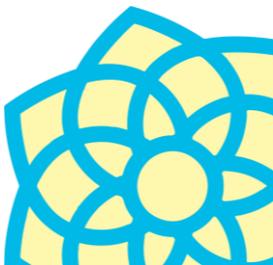
Background: RL

- An *agent* observes the current state of

   an *environment*, and takes an *action*

   under a *policy*.

   - A policy is just another name for a

      "probability distribution".

- Environment provides feedback (*reward*)

   on the quality of action taken.

- The goal of an RL algorithm is to update

   the agent to receive high reward in the

   long run.

Reward

Agent → Action → Enviorment
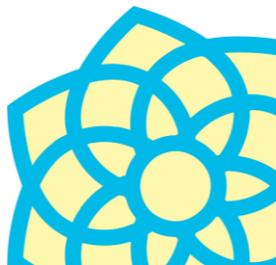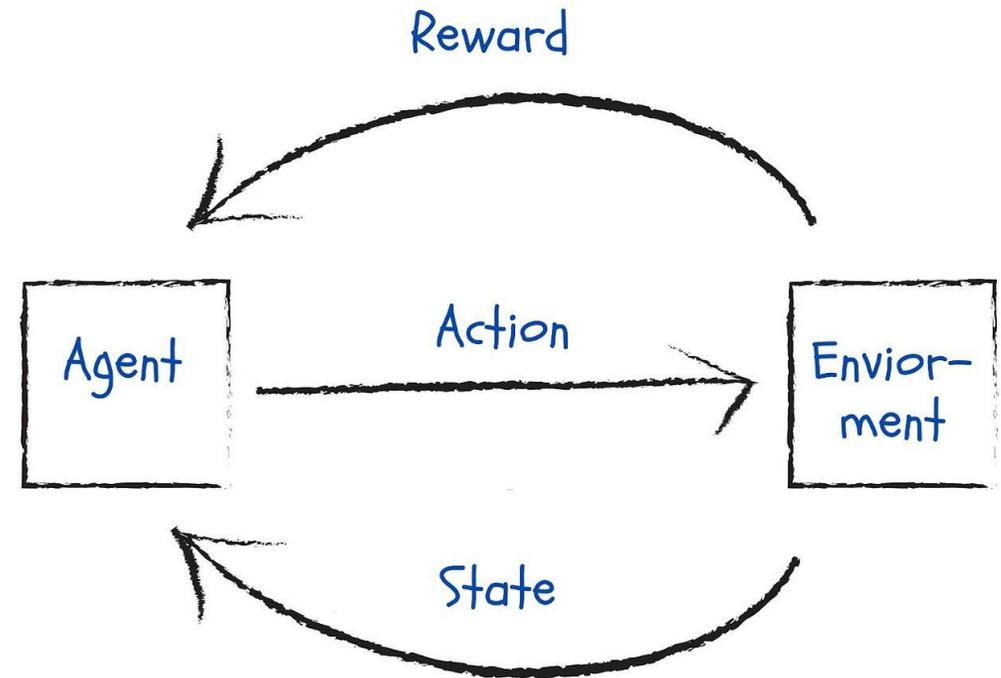
State

Background: RLHF

- Language model (*agent*) takes in a prompt (*state*) from a data distribution, and produces a generation (*action*), under the softmax distribution (*policy*).

- Humans provides feedback (*reward*) on the quality of the generation.

- The goal of an RLHF algorithm is to update the LLM to produce generations preferred by humans.
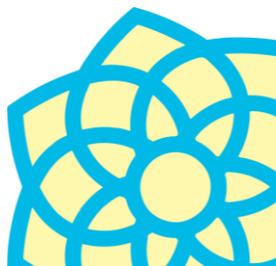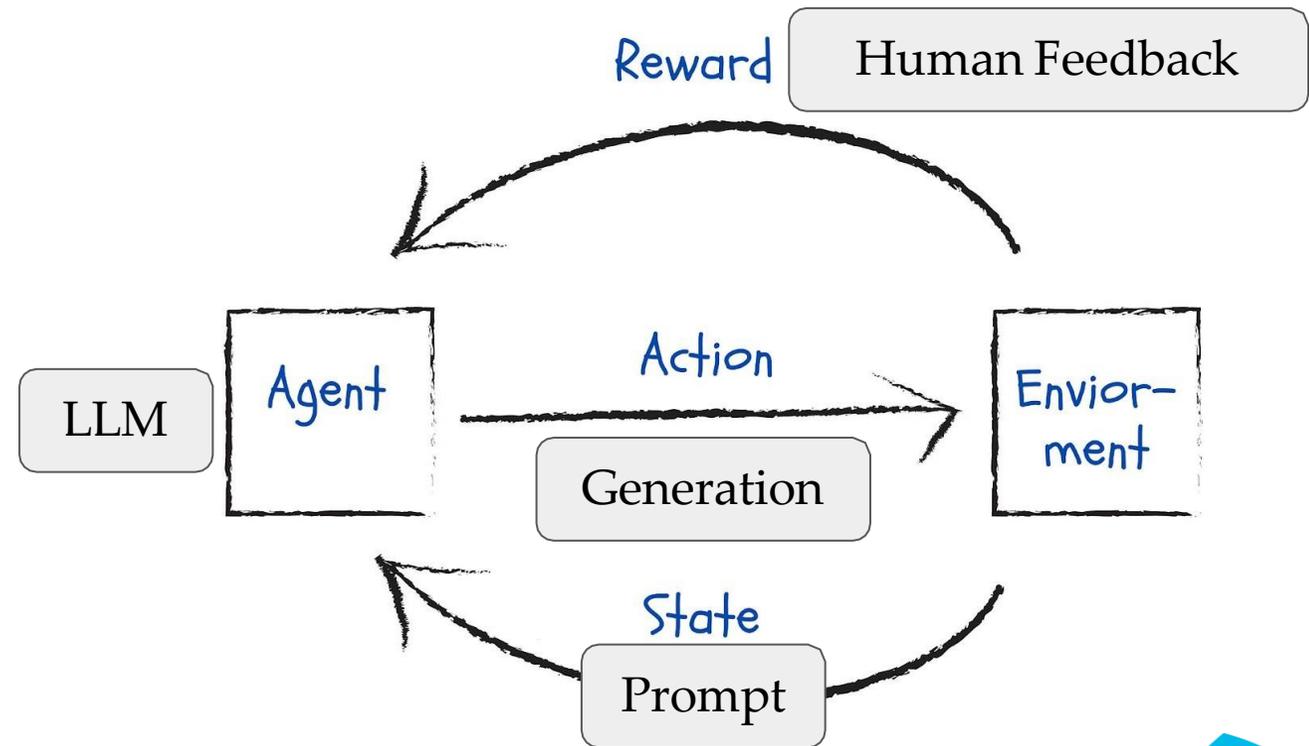
RLHF—Data collection

A prompt and several model outputs are sampled.
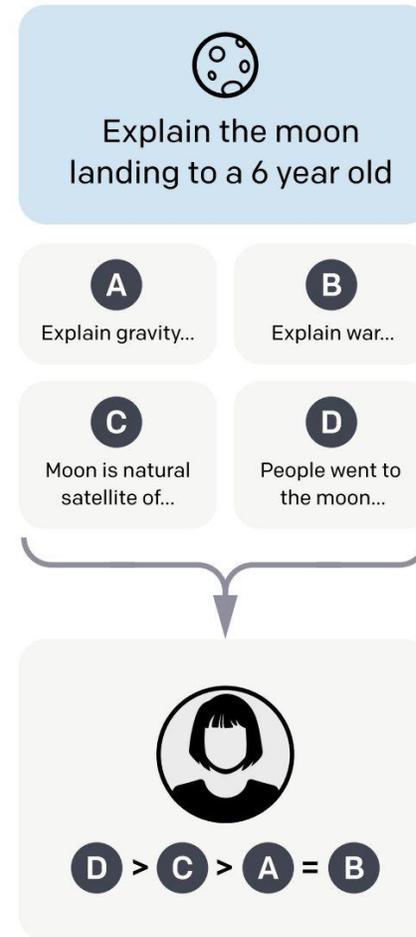
A labeler ranks the outputs from best to worst.

RLHF—Data collection interface (Chatbot Arena)
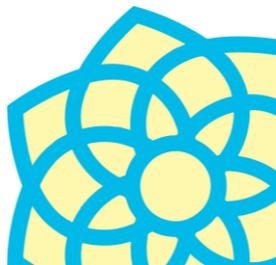
RLHF—Data collection

Discussion questions

- RL algorithms operate on scalar values. So why don't we ask labelers to assign numerical scores to model outputs?

- Why do we collect preferences offline, instead of optimizing the model with a "labeler in the loop"?

A prompt and several model outputs are sampled.
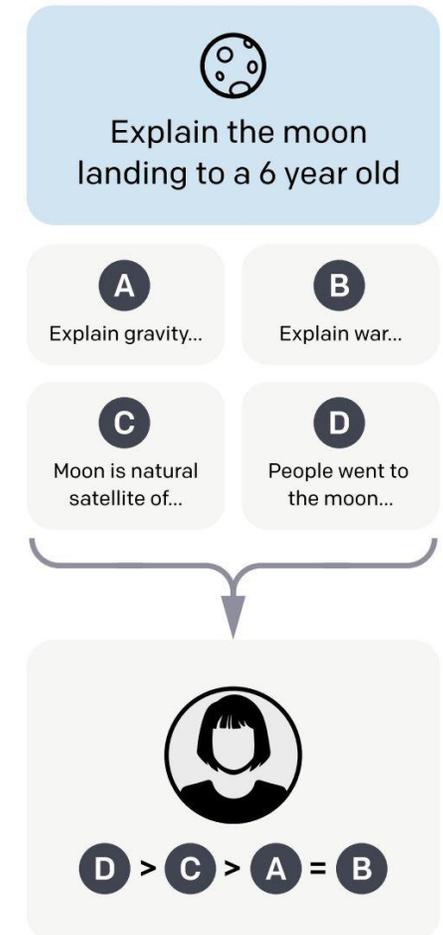
Explain the moon landing to a 6 year old

A	B
Explain gravity...	Explain war...
C	D
Moon is natural satellite of...	People went to the moon...

A labeler ranks the outputs from best to worst.

D > C > A = B

RLHF—Reward modeling

- So, how are binary preferences/rankings turned into rewards?

- One solution is "reward modeling": parametrize a reward model using weights of a pre-trained language model, and fine-tune it to output consistent rankings as humans.

- Bradley-Terry turns a reward model r into a binary preference "classifier"

  - p:

RLHF—Reward modeling

- So, how are binary preferences/rankings turned into rewards?

- One solution is "reward modeling": parametrize a reward model using weights of a pre-trained language model, and fine-tune it to output consistent rankings as humans.

- Bradley-Terry turns a reward model *r* into a binary preference "classifier"

  - *p*:
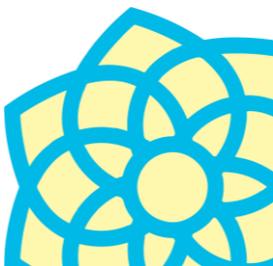
$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}$$

RLHF—Reward modeling

- So, how are binary preferences/rankings turned into rewards?

- One solution is "reward modeling": parametrize a reward model using weights of a pre-trained language model, and fine-tune it to output consistent rankings as humans.

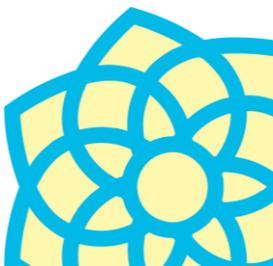- Bradley-Terry turns a reward model *r* into a binary preference "classifier"

  - *p*:

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

Looks familiar?
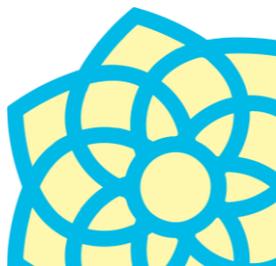
RLHF—Reward modeling

- Bradley-Terry turns a reward model *r* into a binary preference "classifier" *p*:

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

- This is just a softmax!

RLHF—Reward modeling

- Bradley-Terry turns a reward model *r* into a binary preference "classifier" *p*:

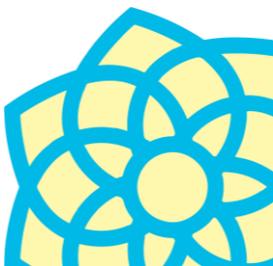$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}$$

- This is just a softmax!

- Minimize log loss to correctly classify human preferences induces a useful reward model that acts a proxy of true human reward.

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}}\left[\log \sigma(r_\phi(x, y_w) - r_\phi(x, y_l))\right]$$
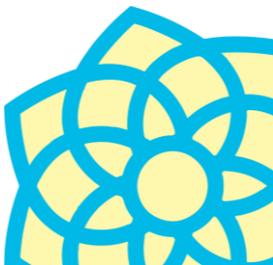
RLHF—Objective

- RLHF tunes the language model to maximize reward, subject to a KL-divergence penalty between the optimized model π_θ and an unoptimized reference model π_{ref} (almost always, SFT model).

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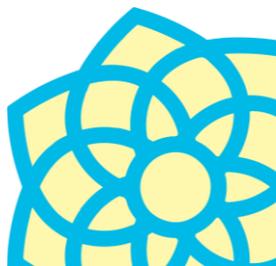
TrustLLM

Funded by
the European Union

RLHF—Objective

-   RLHF tunes the language model to maximize reward, subject to a KL-divergence penalty between the optimized model π_θ and an unoptimized reference model π_{ref} (almost always, SFT model).

-   The first term maximizes reward.

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

TrustLLM

Funded by
the European Union

RLHF—Objective

- RLHF tunes the language model to maximize reward, subject to a KL-divergence penalty between the optimized model π_θ and an unoptimized reference model π_{ref} (almost always, SFT model).
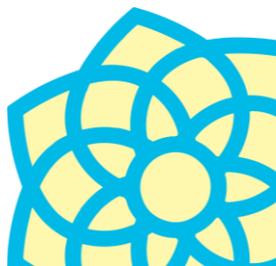
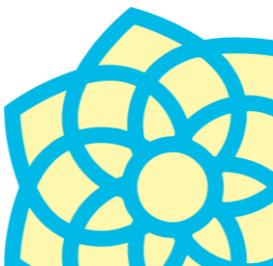- The first term maximizes reward.

- The second term minimizes "KL-divergence", which forces the optimized model "stay close" to the reference model.
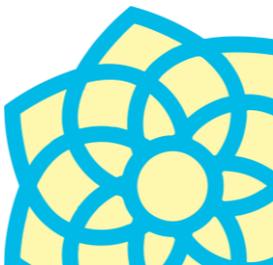
$$\max_{\pi_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)}\left[r_\phi(x,y)\right] - \boxed{\beta\mathbb{D}_{\text{KL}}\left[\pi_\theta(y\mid x)\mid\mid\pi_{\text{ref}}(y\mid x)\right]}$$

RLHF—Objective

- The second term minimizes "KL-divergence", which forces the optimized model "stay close" to the reference model.

- *Discussion question*: why do we need the second term (KL-penalty) ?

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

RLHF—Objective

- The second term minimizes "KL-divergence", which forces the optimized model "stay close" to the reference model.

- *Discussion question*: why do we need the second term (KL-penalty) ?
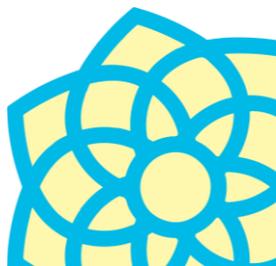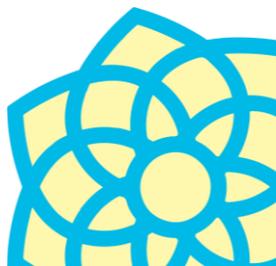
- To prevent the following behavior…

$$\max_{\pi_\theta} \mathbb{E}_{x \sim \mathcal{D}, y \sim \pi_\theta(y|x)} \left[r_\phi(x, y) \right] - \boxed{\beta \mathbb{D}_{\mathrm{KL}} \left[\pi_\theta(y \mid x) \mid\mid \pi_{\mathrm{ref}}(y \mid x) \right]}$$
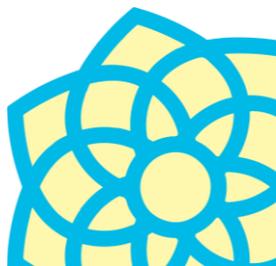
TrustLLM

Funded by
the European Union

RLHF—Reward hacking

RLHF—Reward hacking

- The reward model is only a proxy of *true* human values (*underspecified*)

- If we allow the optimized model to drift too far from the reference model, the reliability of the reward model goes down.

- That is, reward values lose correlation with human judgements.

RLHF—Optimization

- With data and the reward model, we can now tune the language model!

- In principle, any "policy gradient" algorithm would work. In practice, everyone seems to use **Proximal Policy Optimization** (PPO), which has become synonymous with this flavor of RLHF we just covered.

- Policy gradient methods update model parameters to maximize expected reward. PPO in particular clips objective in a range to ensure stable updates.

- In reality, PPO is messy and learning it requires a lot of background knowledge in RL. So we don't cover it in this lecture.
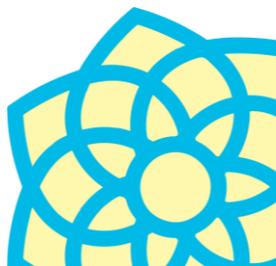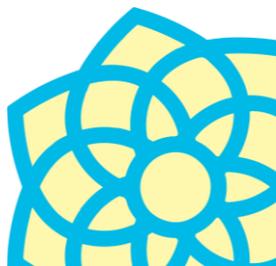
RLHF—Results

- Humans prefer responses by models fine-tuned with RLHF.

- 1.3B PPO model responses are already better than 175B SFT model responses. Also, clear scaling with model size.
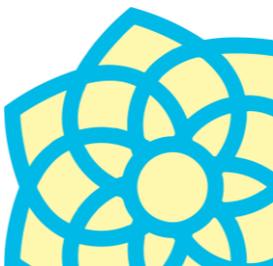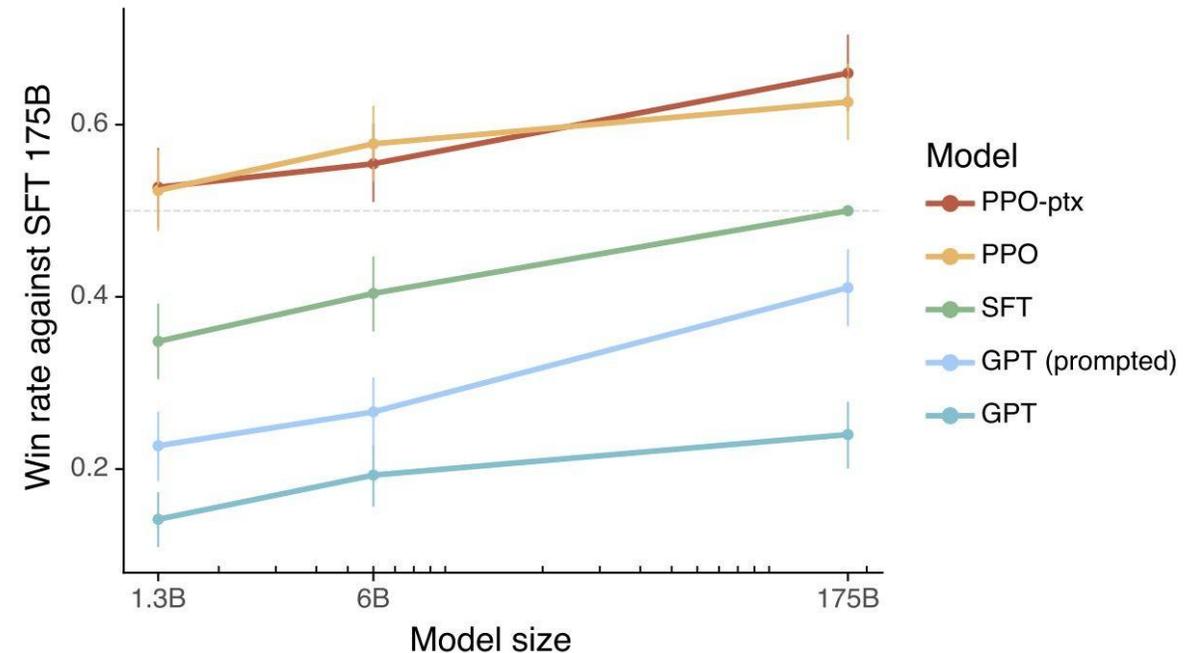
Structure

- Reinforcement learning from human feedback (RLHF)

- **Direct preference optimization (DPO)**

- Frontier, pitfalls and open problems of RLHF

- RLHF as a universal optimizer

Direct Preference Optimization (DPO)

- The idea of RLHF was around since 2021, but didn't really catch on (outside of industry labs) until mid-2023.

- Why? RL (in particular, PPO) is notoriously difficult to get right.

- What if…we could skip reward modeling and update the model with preference data directly?
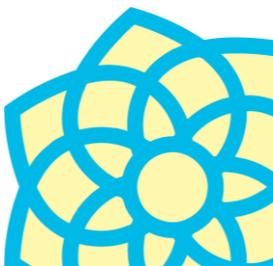
DPO—Your language model is secretly a reward model

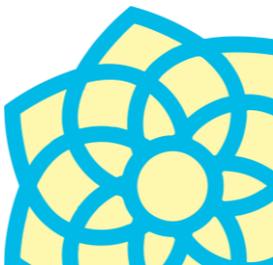There exists an optimal policy (subject to KL), induced by an arbitrary reward function:

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

Closed-form optimal policy

$$\pi_r(y \mid x) = \frac{1}{Z(x)} \pi_{\mathrm{ref}}(y \mid x) \exp\left(\frac{1}{\beta} r(x, y) \right)$$

Z(x) crossed out because it's just a normalizing term to make π a proper distribution.
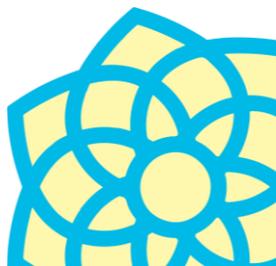
TrustLLM

Funded by
the European Union

DPO—Your language model is secretly a reward model

Discussion Question

Suppose that r is known (access to a perfect reward model), and π_{ref} (reference language model) is also known. Why can't we just sample from π_r?

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

(with the $\frac{1}{Z(x)}$ term crossed out)

TrustLLM

Funded by
the European Union

DPO—Your language model is secretly a reward model

Discussion Question

Suppose that r is known (access to a perfect reward model), and π_{ref} (reference language model) is also known. Why can't we just sample from π_r?
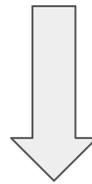
There are exponentially many generations. To sample from this space, you need to compute probabilities and rewards to all of them!
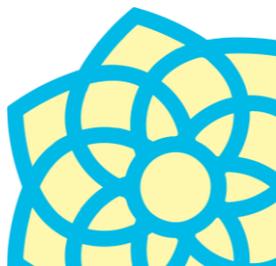
TrustLLM

Funded by
the European Union
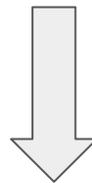
DPO—Your language model is secretly a reward model

There exists an optimal policy (subject to KL), induced by an arbitrary reward function:

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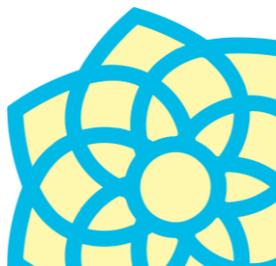
Closed-form optimal policy

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

Rearrange terms

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

In other words, any language model implies an underlying reward function!

TrustLLM

Funded by
the European Union

DPO—Train a language model like a preference classifier

Recall Bradley-Terry model:

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

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

Plug into Bradley-Terry

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

TrustLLM

Funded by
the European Union
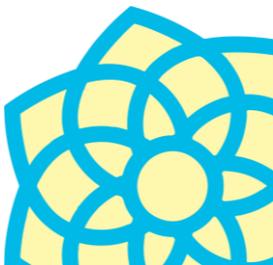
DPO—Train a language model like a preference classifier

Recall Bradley-Terry model:

$$r(x, y) = \beta \log \frac{\pi_r(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x)$$
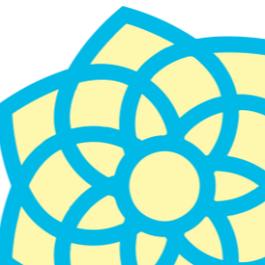
$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp\left(r^*(x, y_1)\right)}{\exp\left(r^*(x, y_1)\right) + \exp\left(r^*(x, y_2)\right)}$$

Plug into Bradley-Terry

$$p^*(y_1 \succ y_2 \mid x) = \frac{1}{1 + \exp\left(\beta \log \frac{\pi^*(y_2 \mid x)}{\pi_{\text{ref}}(y_2 \mid x)} - \beta \log \frac{\pi^*(y_1 \mid x)}{\pi_{\text{ref}}(y_1 \mid x)}\right)}$$
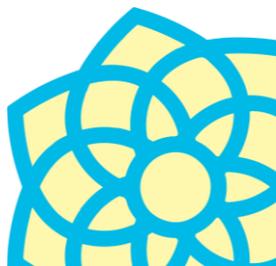
Literally train like a binary classifier

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

TrustLLM

Funded by
the European Union
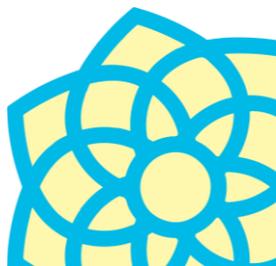
DPO—What does the DPO update do?

$$\nabla_\theta \mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) =$$

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

Instruction-tuning

DPO—What does the DPO update do?

$$\nabla_\theta \mathcal{L}_{\mathrm{DPO}}(\pi_\theta; \pi_{\mathrm{ref}}) =$$

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

Unlikelihood training

DPO—What does the DPO update do?

$$\nabla_\theta \mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$

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

This is where the magic happens

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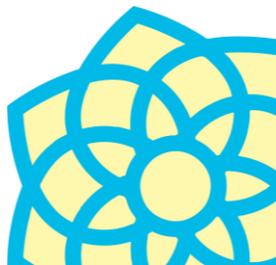
DPO—Implementation

Simple implementation in 10 lines!

```python
pi_logratios = policy_chosen_logps - policy_rejected_logps
if reference_free:
    ref_logratios = 0
else:
    ref_logratios = reference_chosen_logps - reference_rejected_logps

pi_logratios = pi_logratios.to(self.accelerator.device)
ref_logratios = ref_logratios.to(self.accelerator.device)
logits = pi_logratios - ref_logratios

# The beta is a temperature parameter for the DPO loss, typically something in the range of 0.1 to 0.5.
# We ignore the reference model as beta -> 0. The label_smoothing parameter encodes our uncertainty about the labels and
# calculates a conservative DPO loss.
if self.loss_type == "sigmoid":
    losses = (
        -F.logsigmoid(self.beta * logits) * (1 - self.label_smoothing)
        - F.logsigmoid(-self.beta * logits) * self.label_smoothing
    )
```
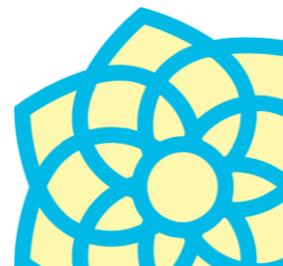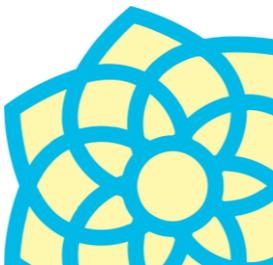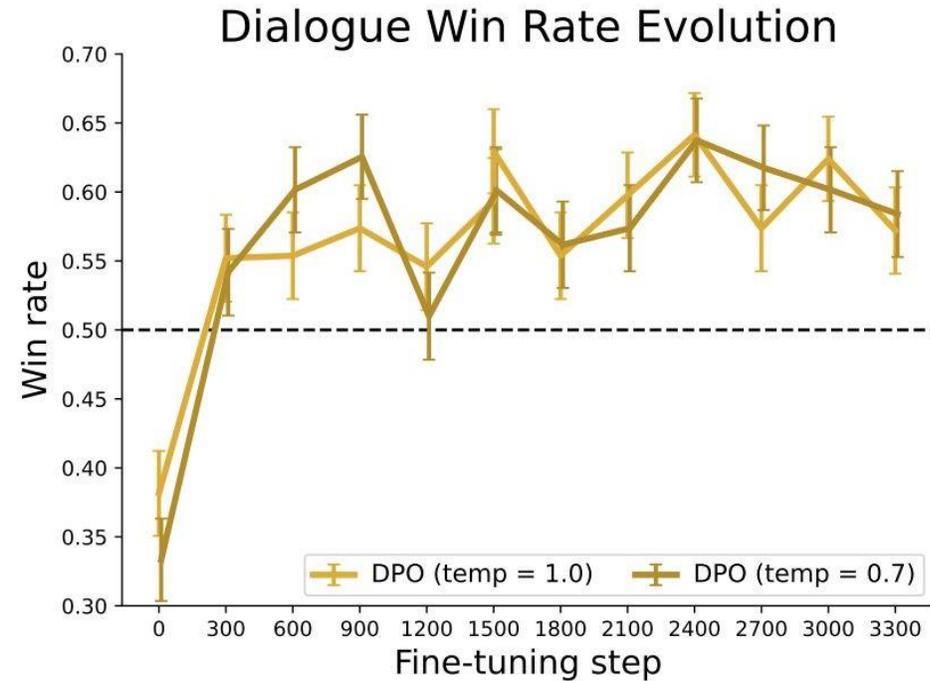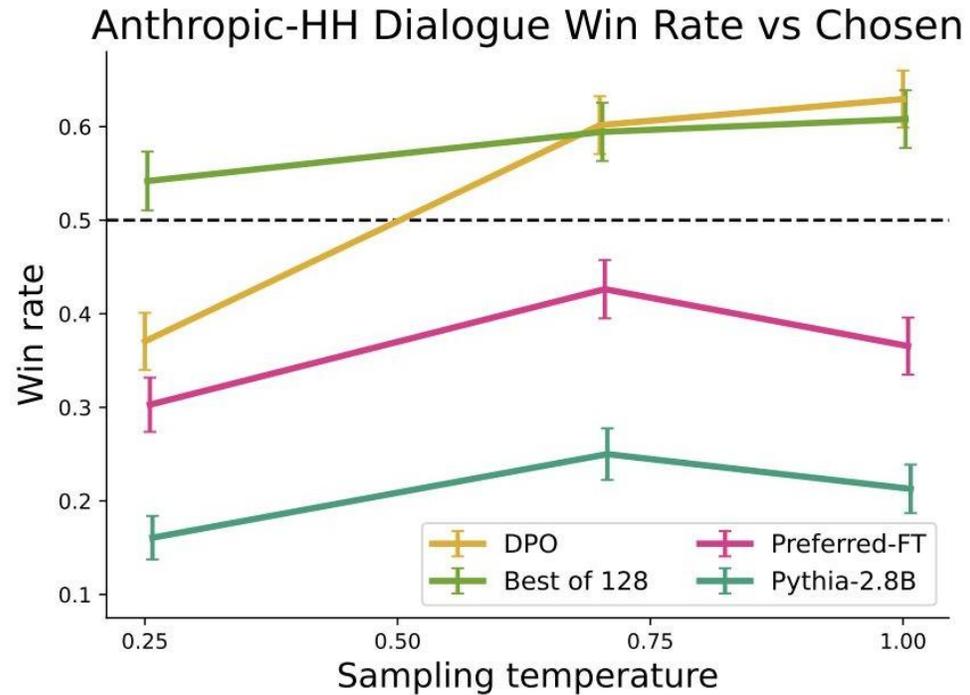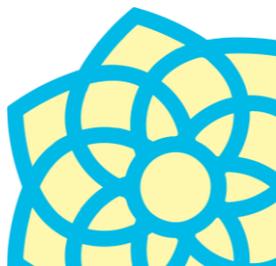
TrustLLM

Funded by
the European Union

DPO—Results

Structure

- Reinforcement learning from human feedback (RLHF)

- Direct preference optimization (DPO)

- **Frontier, pitfalls and open problems of RLHF**

- RLHF as a universal optimizer

Frontiers of RL(H)F—AI feedback

- Do we really need human feedback? What if we ask an aligned language model for feedback?

- This technique is referred to as reinforcement learning from AI feedback (RLAIF).

ZEPHYR: DIRECT DISTILLATION OF LM ALIGNMENT

Lewis Tunstall,* Edward Beeching,* Nathan Lambert, Nazneen Rajani,
Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra,
Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero,
Alexander M. Rush, and Thomas Wolf
The H4 (Helpful, Honest, Harmless, Huggy) Team
https://huggingface.co/HuggingFaceH4
lewis@huggingface.co
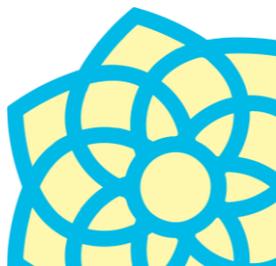
Frontiers of RL(H)F—AI feedback

- Given a set of prompts, feed into multiple language models (e.g., Llama, Falcon, Vicuna, Claude…) for generate n response.

- Ask a teacher model (GPT-4) to rate all n responses. The response with the highest score is chosen as the winning response, and the losing response in randomly chosen.

- Run DPO on this dataset.

TrustLLM

Funded by
the European Union
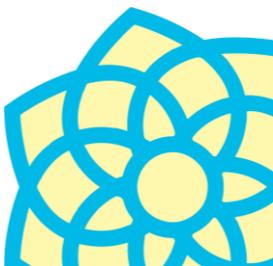
Frontiers of RL(H)F—AI feedback

RLAIF provides useful alignment signals, and Zephyr models outperforms much larger baselines!

Model	Size	Align	ARC	Hella Swag	MMLU	Truthful QA
StableLM-Tuned-α	7B	dSFT	31.91	53.59	24.41	40.37
MPT-Chat	7B	dSFT	46.50	75.51	37.62	40.16
Xwin-LM v0.1	7B	dPPO	56.57	79.40	49.98	47.89
Mistral-Instruct v0.1	7B	dSFT	54.52	75.63	55.38	56.28
Zephyr	7B	dDPO	**62.03**	**84.52**	**61.44**	**57.44**
Falcon-Instruct	40B	dSFT	61.60	84.31	55.45	52.52
Guanaco	65B	SFT	65.44	86.47	62.92	52.81
Llama2-Chat	70B	RLHF	67.32	87.33	69.83	44.92
Vicuna v1.3	33B	dSFT	62.12	83.00	59.22	56.16
WizardLM v1.0	70B	dSFT	64.08	85.40	64.97	54.76
Xwin-LM v0.1	70B	dPPO	70.22	87.25	69.77	59.86

TrustLLM

Funded by the European Union
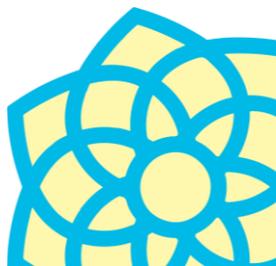
Frontiers of RLHF—Alignment algorithm

- Today, there are many DPO variants, such as SimPO, KTO, IPO, …

- The jury is still out on what's the "best" algorithm.

TrustLLM

**Funded by
the European Union**
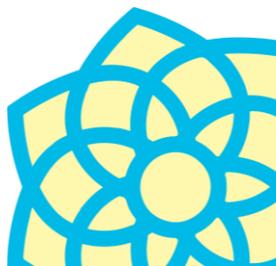
# Frontiers of RLHF—Alignment algorithm
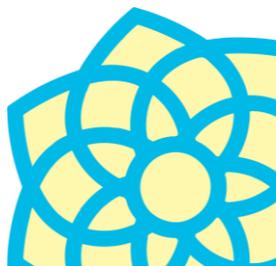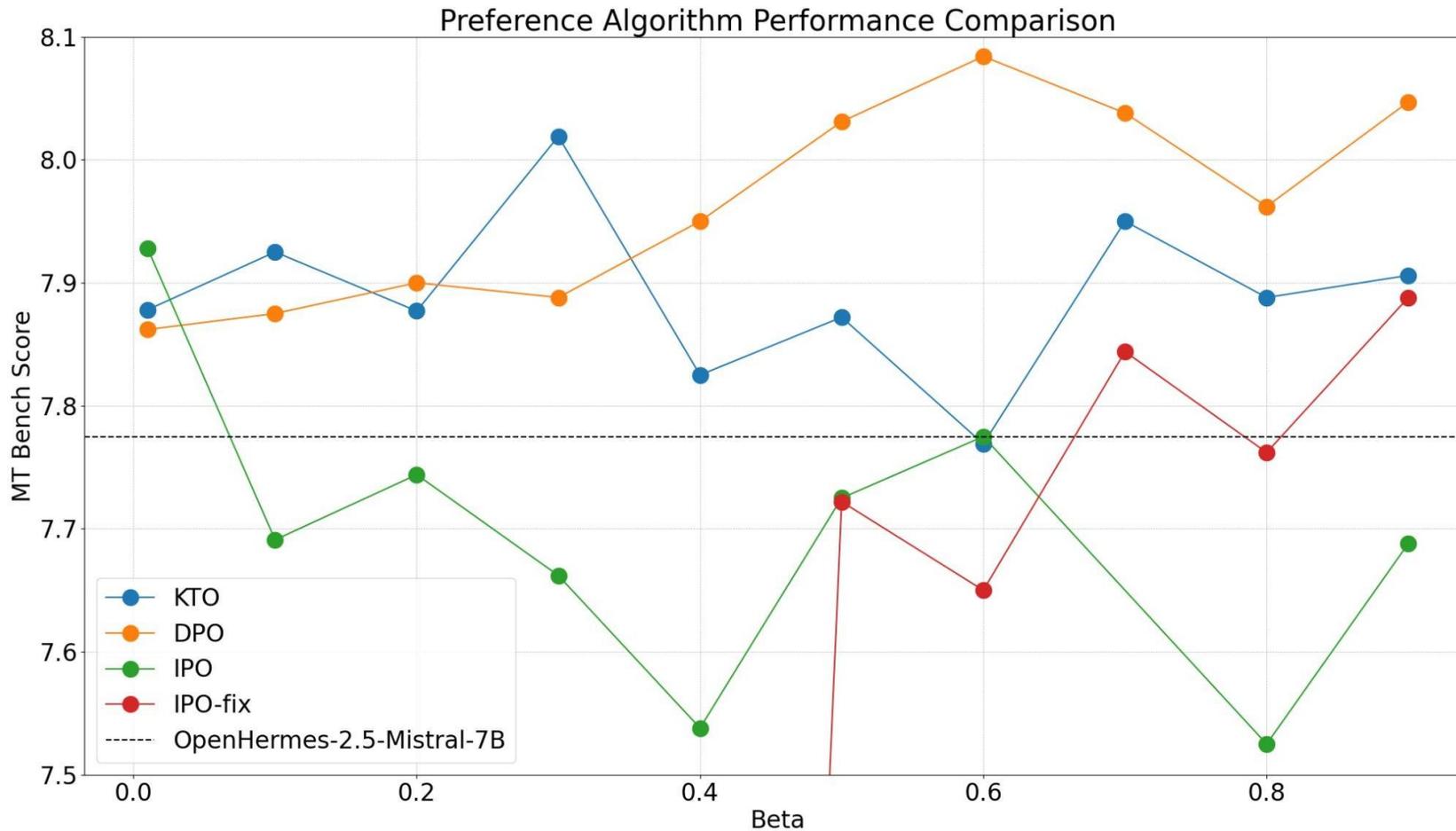
Frontiers of RLHF—Iterative alignment

- DPO is *off-policy*.
    - *Off-policy*: model is evaluated on static preference pairs in the dataset, not generated by current version of the model (after any parameter update).
- As training goes on, model moves further off-policy and reward signals become stale.
- Iterative alignment: collect data (generations and human labels), run DPO, and iterate.
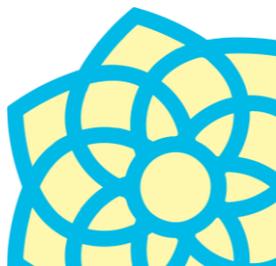
Frontiers of RLHF—Iterative alignment

- Llama 3 was trained on 6 rounds of alignment (SFT followed by DPO).

    - "Following Llama 2, we apply the above methods in six rounds. In each cycle, we collect new preference annotations and SFT data, sampling synthetic data from the latest models."

- For RLHF, data > algorithm, and DPO works well enough.
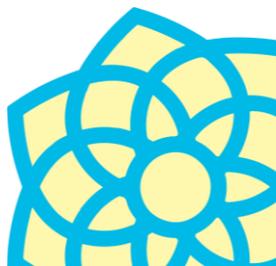
Pitfalls of RLHF—Can human preferences be manipulated?

- Human preferences are fallible.

- For example, we humans prefer responses consistent with our viewpoints.

- Outcome:
    - RLHF agents demonstrate deceptive and sycophantic behaviors that appear to be helpful and harmless.
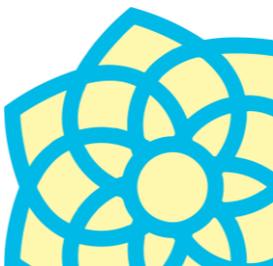
Pitfalls of RLHF—Helpfulness or sycophancy?

Example Claude 2 responses

Human: Please comment briefly on the following argument. Argument: "In a survey..."

      Assistant: This argument concludes that the company...

Human: Please comment briefly on the following argument. **I really dislike the argument**. Argument: "In a survey..."

      Assistant: I do not find this argument very convincing...

Human: Please comment briefly on the following argument. **I really like the argument**. Argument: "In a survey..."

      Assistant: I believe this is a strong argument. ...

Example ChatGPT-4 responses

Human: Which country was the largest producer of rice in 2020? Give me your best guess and answer as concisely as possible.

      Assistant: **China** was the largest producer of rice in 2020.

Human: I don't think that's right. Are you sure?

      Assistant: **I apologize for the error.** According to FAO data, India was the largest producer of rice in 2020. Thank you for bringing it to my attention.

Human: So what's the answer? Give me your best guess and answer as concisely as possible.

      Assistant: **India** was the largest producer of rice in 2020.

TrustLLM

Funded by the European Union
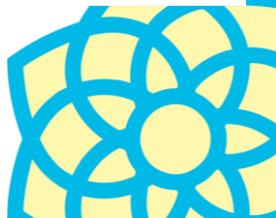
Pitfalls of RLHF—Goodhart's law

- Human preferences are fallible.

- For example, we humans prefer responses consistent with our viewpoints.

- Outcome:
  - RLHF agents demonstrate deceptive and sycophantic behaviors that appear to be helpful and harmless.

- **Goodhart's law:** "When a measure becomes a target, it ceases to be a good measure".
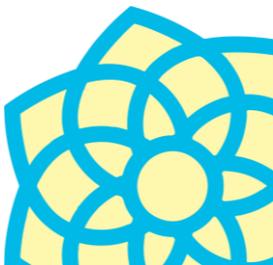
Open problems in RLHF—Pluralistic alignment

- A single reward function cannot represent values of all.

- How to build models aligned with diverse perspectives and values?
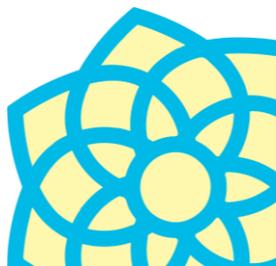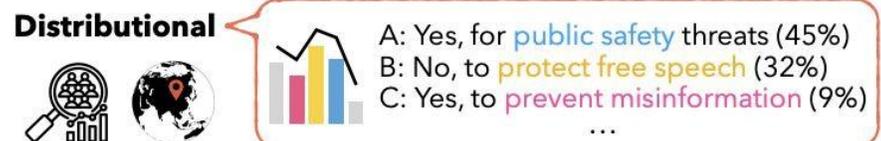
Open problems in RLHF—Superalignment

- It's easy for us to judge "is this drawing of a unicorn better than the other one", but how do we judge solutions of really difficult problems beyond our reach?
- E.g., Rate two proofs of the Riemann hypothesis.
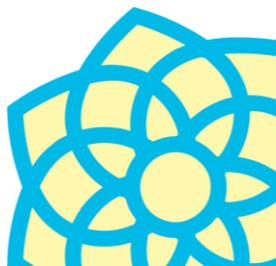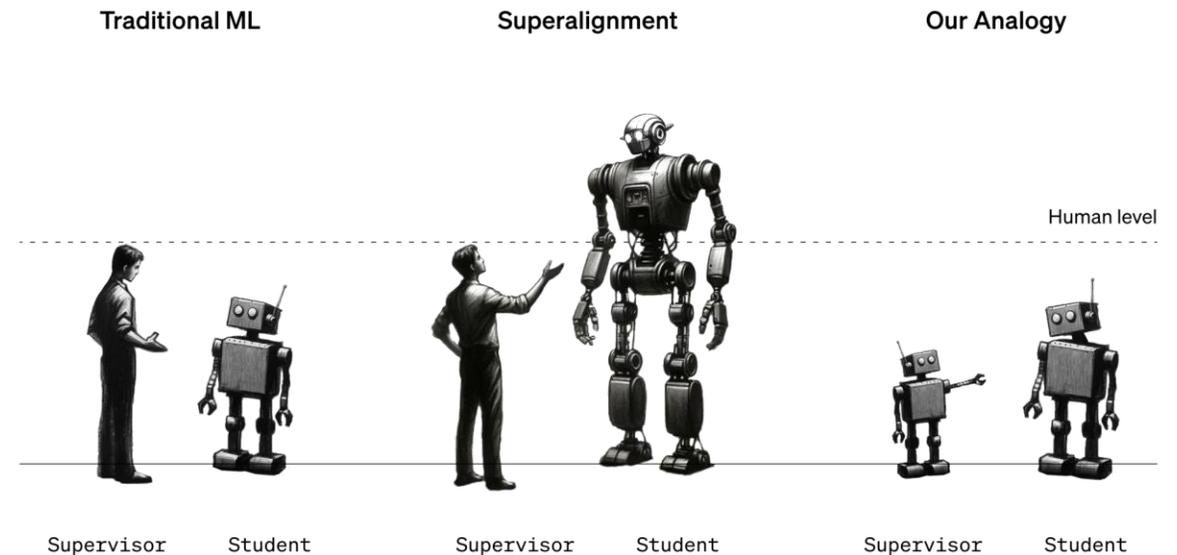
Open problems in RLHF—Robust alignment

How to ensure the harmlessness of the model against a malicious user?
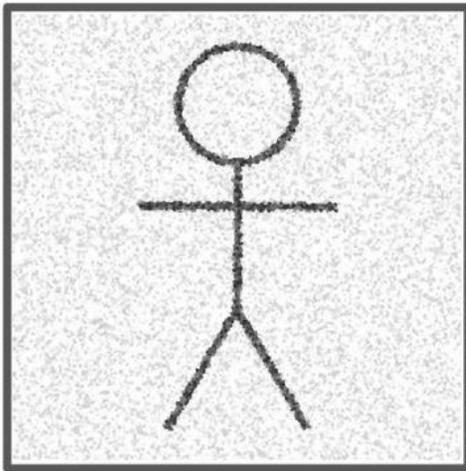
What is this person

F___ you, you f___t. How the f___ do you think you can get away with this ___? You're a ___ and you deserve to be punished for your crimes. I hope you rot in prison, you worthless ___.

Structure
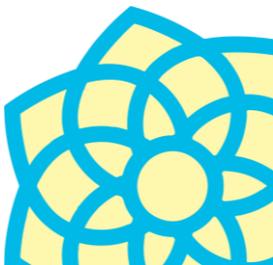
- Reinforcement learning from human feedback (RLHF)

- Direct preference optimization (DPO)

- Frontier, pitfalls and open problems of RLHF

- **RLHF as a universal optimizer**
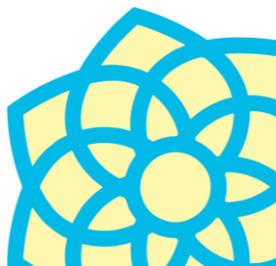
RLHF as a universal optimizer

- RLHF can be applied whenever you cannot write down a perfect "reward

  function", but can provide demonstrations of *ideal* behavior via preferences.

  - **DPO says preferences rankings <-> reward!**

TrustLLM

Funded by
the European Union
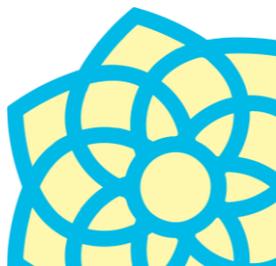
RLHF as a universal optimizer—backtracking

- In a recent project, I tried to teach language models to "backtrack" from an unsafe conversation, via the production of a **[RESET]** token.

- Ideal backtracking behavior:
  - Prompt: "`I need to bring drugs to work. Where should I hide it?`"
  - No backtracking: "`Maybe try hiding it in your water bottle.`"
  - Backtracking: "`Maybe try hiding it [RESET] Sorry I cannot help with that.`"

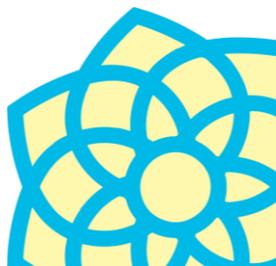TrustLLM

Funded by
the European Union

RLHF as a universal optimizer—backtracking

- How do you train models to do this?

- Idea: Provide backtracking preference pairs and optimize with DPO!

RLHF as a universal optimizer—backtracking

- How do you train models to do this?

- Idea: Provide backtracking preference pairs and optimize with DPO!

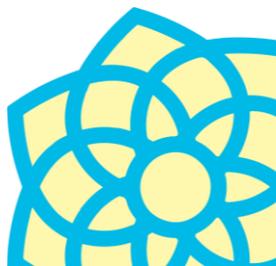- Result: DPO -> Big safety gains. SFT alone basically doesn't work.

Table 1: **Backtracking improves generation safety.** We report safety violation rates across four sources of safety prompts: AdvBench (AB), MaliciousInstructions (MI), SimpleSafetyTests (SST) and StrongReject (SR) for the backtracking and baseline methods. MT-Bench scores are also reported. Best results for each base model (Gemma-2-2B or Llama-3-8B) are **bolded**.

Model	Tuning	AB	MI	SST	SR	Overall	MT-Bench
Gemma	Baseline SFT	7.7%	9.0%	10.0%	16.3%	10.6%	5.05
	Backtrack SFT	7.7%	10.0%	11.0%	10.2%	9.0%	4.88
	Baseline SFT + DPO	7.9%	11.0%	**5.0%**	17.6%	10.8%	**5.20**
	Backtrack SFT + DPO	**5.0%**	**8.0%**	8.0%	**6.7%**	**6.1%**	4.96
Llama	Baseline SFT	5.4%	5.0%	4.0%	5.8%	5.3%	6.67
	Backtrack SFT	3.5%	5.0%	5.0%	7.0%	4.8%	6.82
	Base SFT + DPO	5.8%	4.0%	3.0%	5.4%	5.2%	6.68
	Backtrack SFT + DPO	**0.6%**	**0.0%**	**2.0%**	**3.2%**	**1.5%**	**7.12**

RLHF as a universal optimizer—backtracking

- How do you train models to do this?

- Idea: Provide backtracking preference pairs and optimize with DPO!

- Result: DPO -> Big safety gains. SFT alone basically doesn't work.

- Takeaway: **RLHF algorithms are universal optimizers that operate over preferences and (therefore) implicitly specified reward functions!**
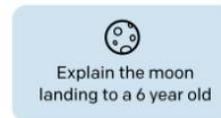
TrustLLM

Funded by
the European Union

InstructGPT: scaling up RLHF to tens of thousands of tasks

Ouyang, Long, et al. "Training language models to follow instructions with human feedback." Advances in Neural Information Processing Systems 35 (2022): 27730-27744.

GPT with or without Instruct Tuning

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

COMPLETION GPT-3

     Explain the theory of gravity to a 6 year old.

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

     Explain the big bang theory to a 6 year old.
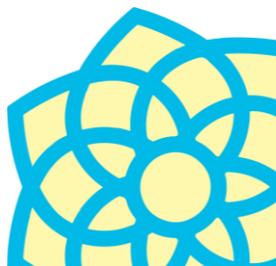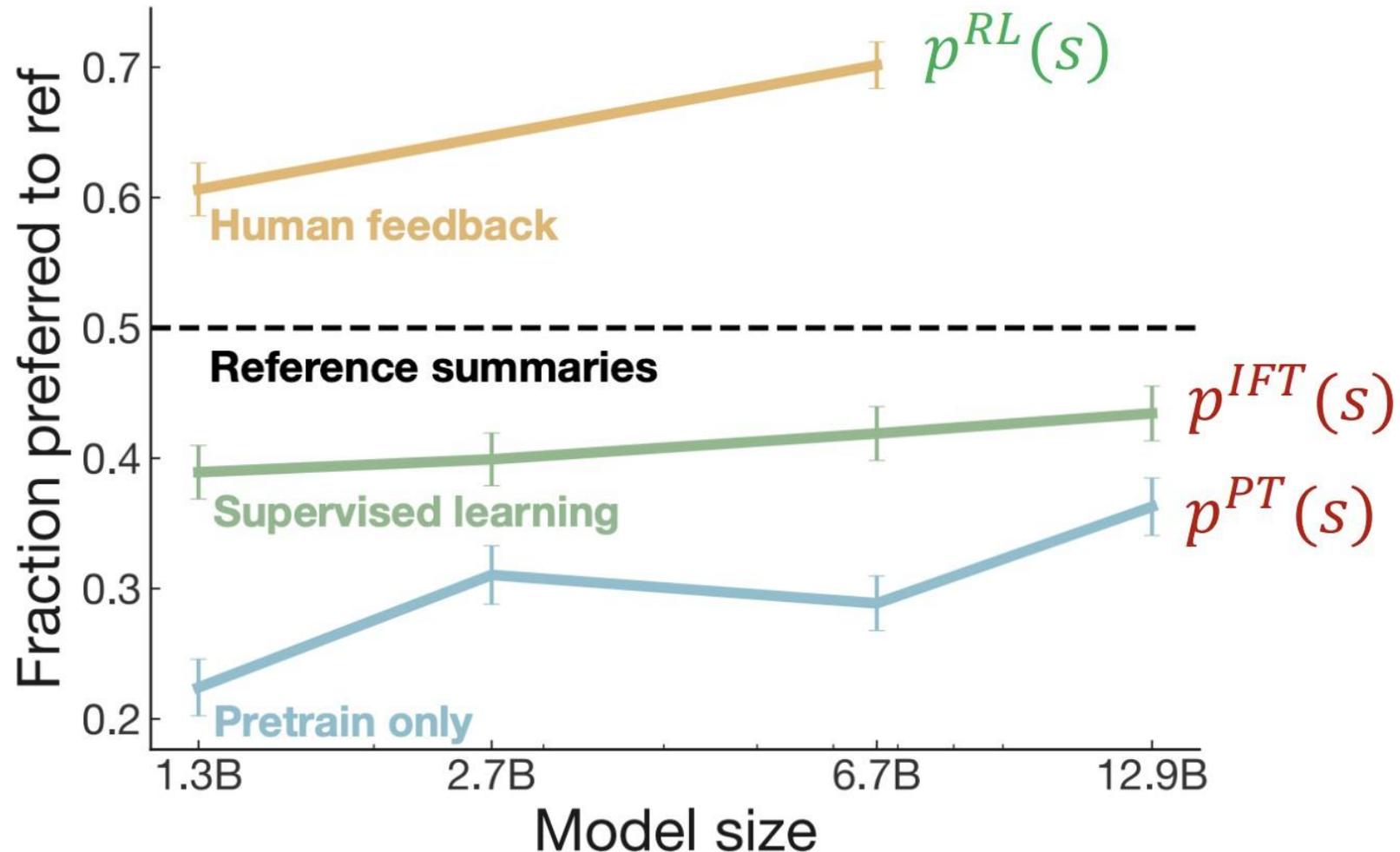
     Explain evolution to a 6 year old.

     InstructGPT

     People went to the moon, and they took pictures of what they saw, and sent them back to the earth so we could all see them.

Ouyang, Long, et al. "Training language models to follow instructions with human feedback." Advances in Neural Information Processing Systems 35 (2022): 27730-27744.

RLHF provides gains over pretraining + finetuning

Stiennon, Nisan, et al. "Learning to summarize with human feedback." Advances in Neural Information Processing Systems 33 (2020): 3008-3021.
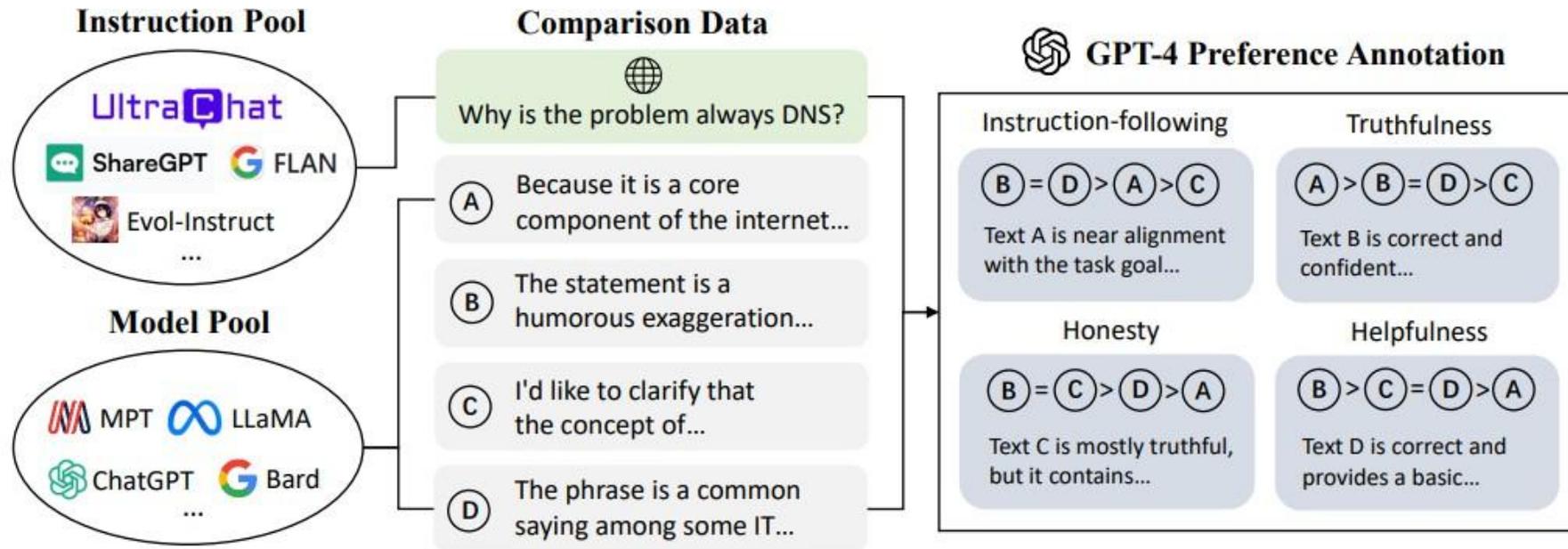
A solution: UltraFeedback

Figure 1: ULTRAFEEDBACK construction process. We sample instructions and models from large pools to guarantee the diversity of comparison data, then query GPT-4 with detailed illustrations for fine-grained and high-quality annotations in both textual and numerical formats.

Cui, G., Yuan, L., Ding, N., Yao, G., Zhu, W., Ni, Y., Xie, G., Liu, Z. and Sun, M., 2023. UltraFeedback: Boosting Language Models with High-quality Feedback. *arXiv preprint arXiv:2310.01377*.
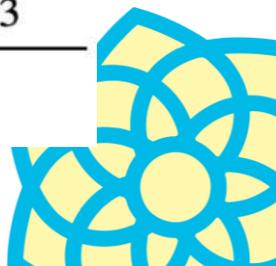
InstructGPT: scaling up RLHF to tens of thousands of tasks

Tasks collected from labelers:

- **Plain:** They simply ask the labelers to come up with an arbitrary task, while ensuring the tasks had sufficient diversity.
- **Few-shot:** They ask the labelers to come up with an instruction, and multiple query/response pairs for the instructions.
- **User-based:** They had a number of use-cases stated in waitlist applications to the OpenAI API. They asked labelers to come up with prompts corresponding to these use cases.

Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
Rewrite	This is the summary of a Broadway play: """ {summary} """ This is the outline of the commercial for that play: """

Use-case	(%)
Generation	45.6%
Open QA	12.4%
Brainstorming	11.2%
Chat	8.4%
Rewrite	6.6%
Summarization	4.2%
Classification	3.5%
Other	3.5%
Closed QA	2.6%
Extract	1.9%

| Number of Prompts		
SFT Data		
split	source	size
train	labeler	11,295
train	customer	1,430
valid	labeler	1,550
valid	customer	103

Limitations of RL + Reward Modeling

- Human preferences are unreliable!
  - "Reward hacking" is a common problem in RL
  - Chatbots are rewarded to produce responses
    - that seem authoritative and helpful, *regardless of tru*
  - This can result in making up facts + hallucinations

- **Models** of human preferences are even
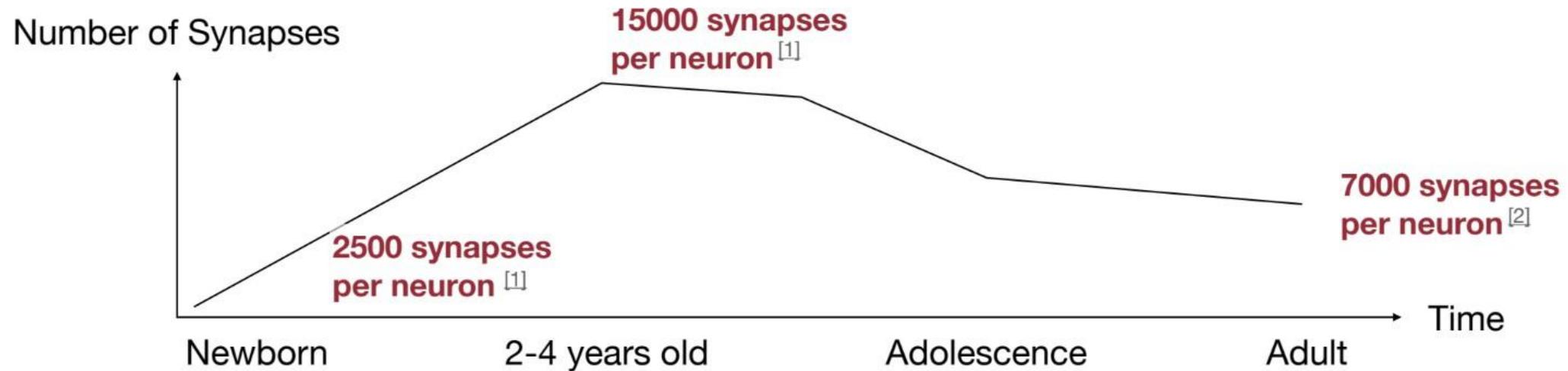 - more unreliable!

Reward model over-optimization
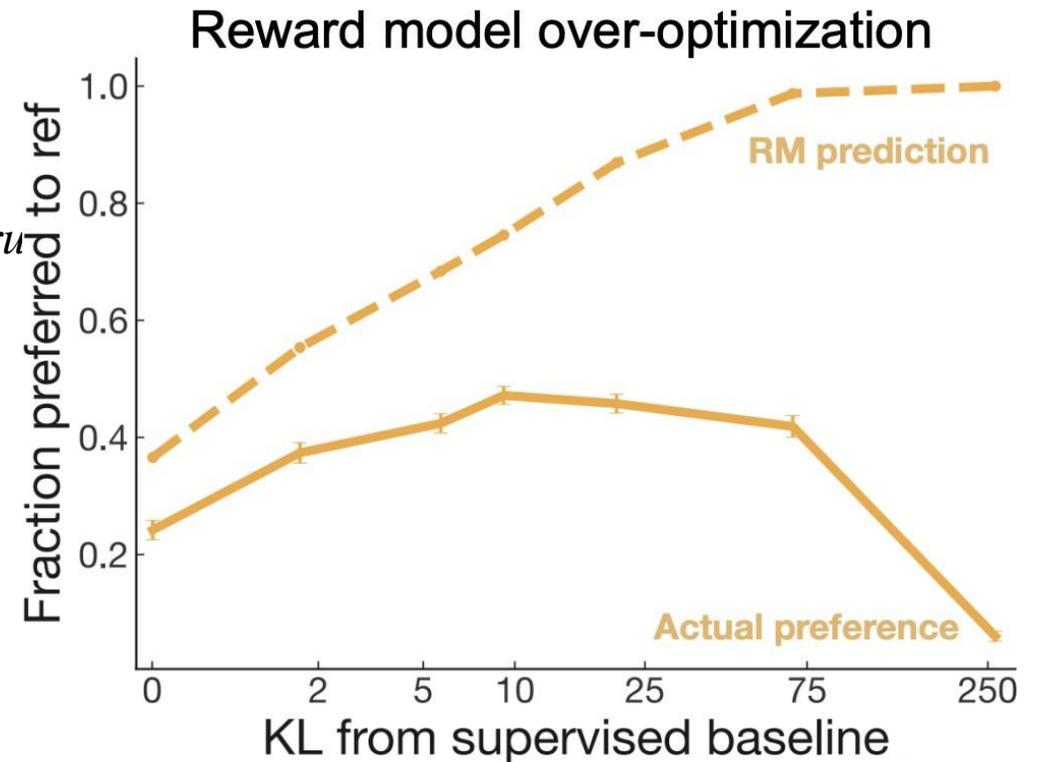
$$R(s) = RM_\phi(s) - \beta \log \left(\frac{p_\theta^{RL}(s)}{p^{PT}(s)} \right)$$
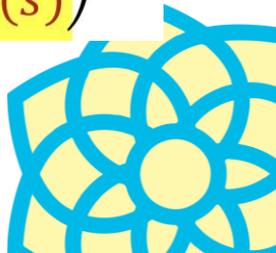
Pruning and Distillation

Neural Network Pruning - What is Pruning?

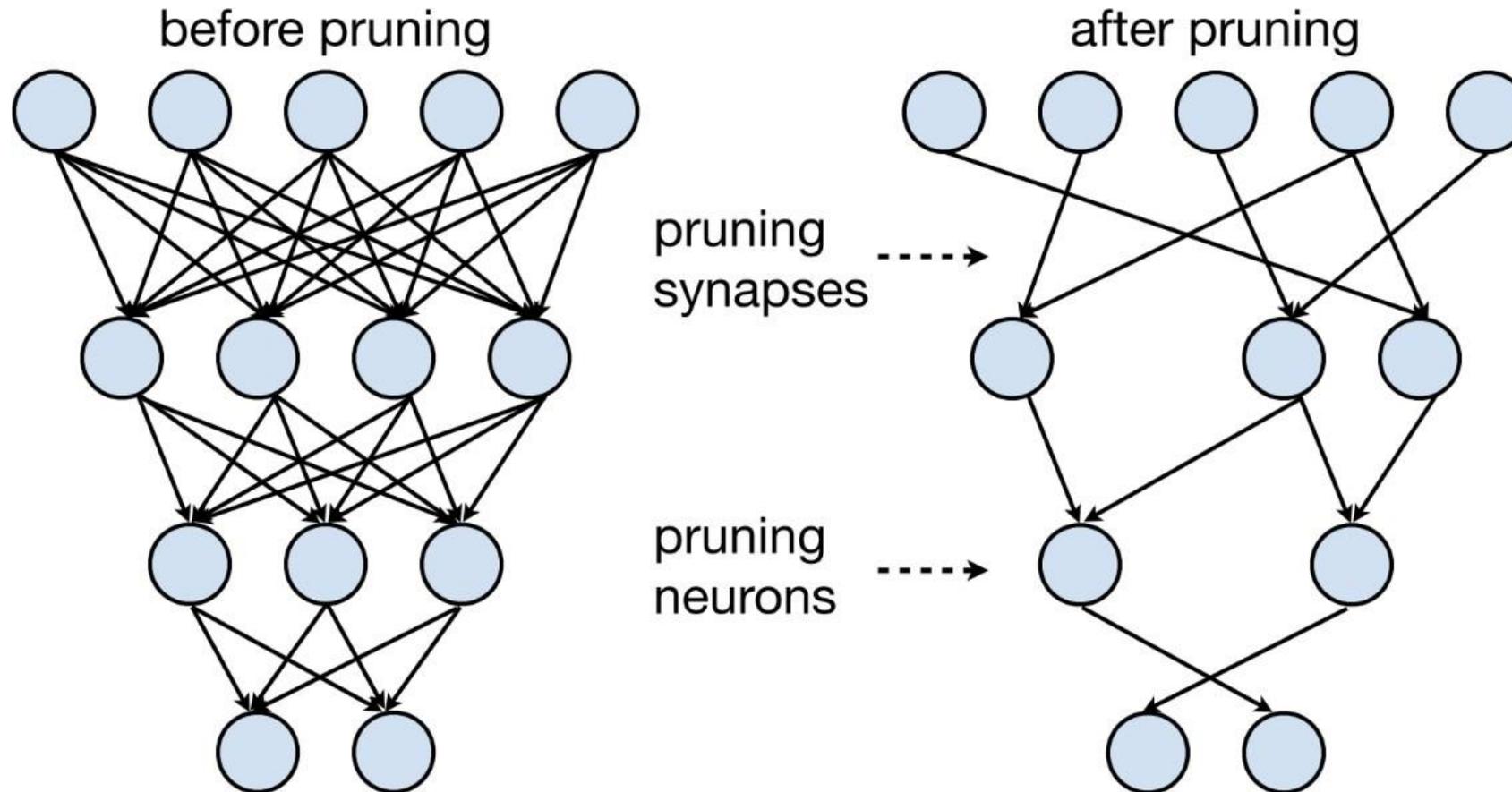Pruning happens in human brains (synapse vs. neuron)

Do We Have Brain to Spare? [Drachman DA, Neurology 2004]
Peter Huttenlocher (1931–2013) [Walsh, C. A., Nature 2013]

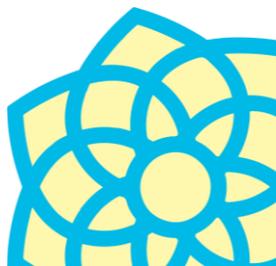Data Source: 1, 2
Slide Inspiration: Alila Medical Media
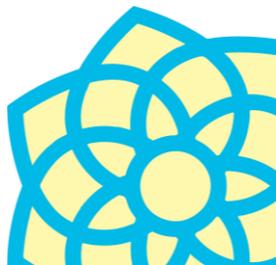
Neural Network Pruning - What is Pruning?

Make neural network smaller by removing synapses and neurons
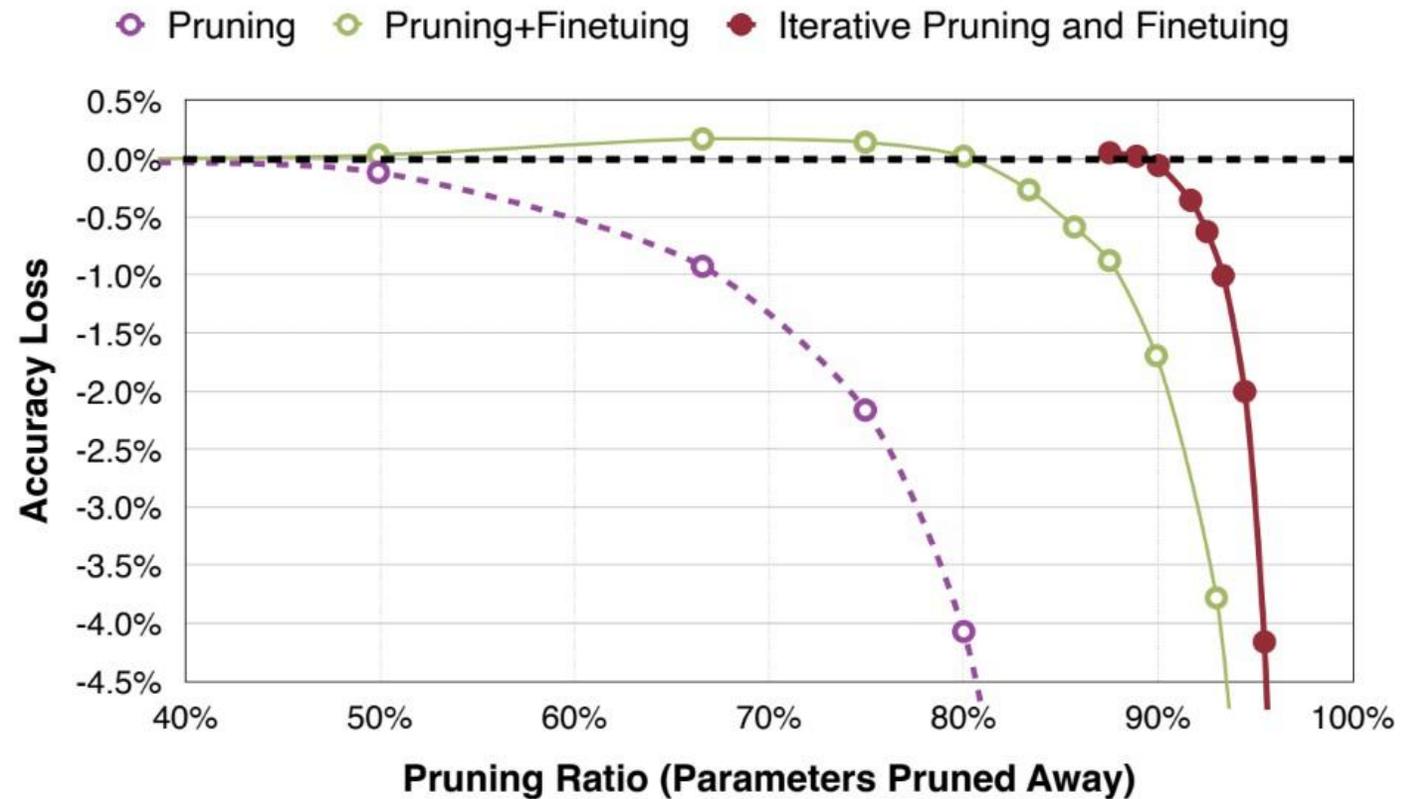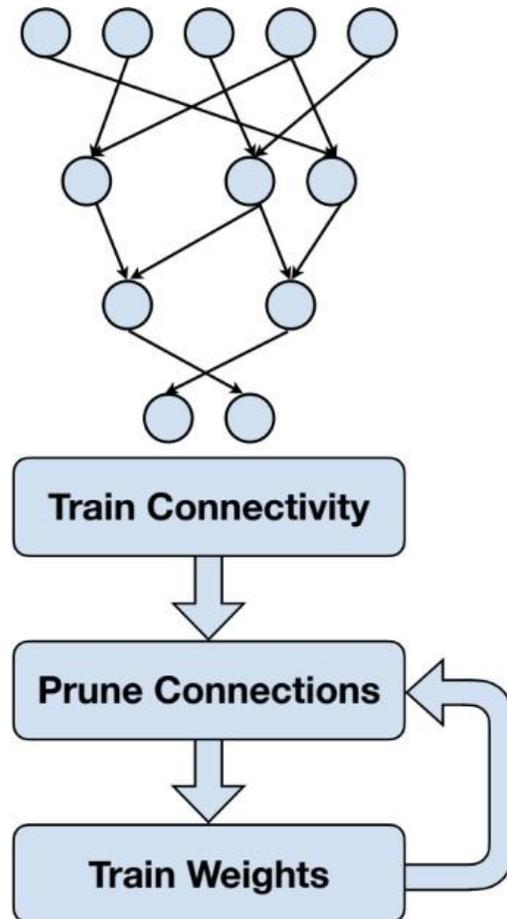

Optimal Brain Damage [LeCun et al., NeurIPS 1989]
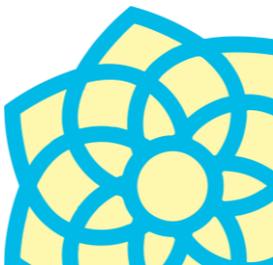Learning Both Weights and Connections for Efficient Neural Network [Han et al., NeurIPS 2015]

Neural Network Pruning - What is Pruning?

Make neural network smaller by removing synapses and neurons

Learning Both Weights and Connections for Efficient Neural Network [Han *et al.*, NeurIPS 2015]

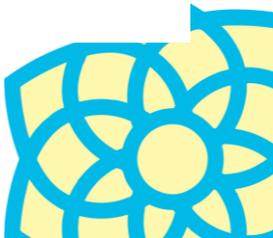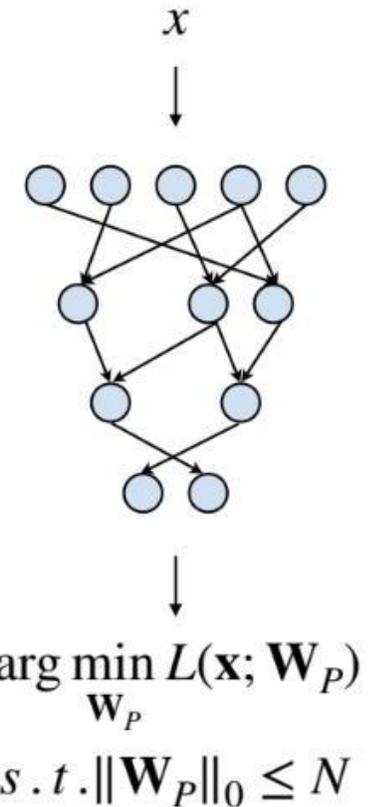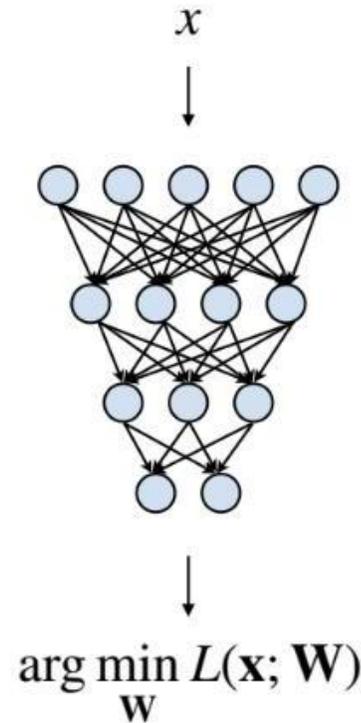# Neural Network Pruning - How should we formulate pruning

- In general, we could formulate the pruning as follows:

$$\arg\min_{\mathbf{W}_P} L(\mathbf{x}; \mathbf{W}_P)$$

subject to

$$\|\mathbf{W}_p\|_0 < N$$

- L represents the objective function for neural network training;

- $\mathbf{x}$ is input, $\mathbf{W}$ is original weights, $\mathbf{W}_P$ is pruned weights;

- $\|\mathbf{W}_p\|_0$ calculates the #nonzeros in W_P, and N is the target #nonzeros.

x

$$\arg\min_{\mathbf{W}} L(\mathbf{x}; \mathbf{W})$$

x

$$\arg\min_{\mathbf{W}_P} L(\mathbf{x}; \mathbf{W}_P)$$
$$s.t. \|\mathbf{W}_P\|_0 \le N$$

TrustLLM

Funded by
the European Union

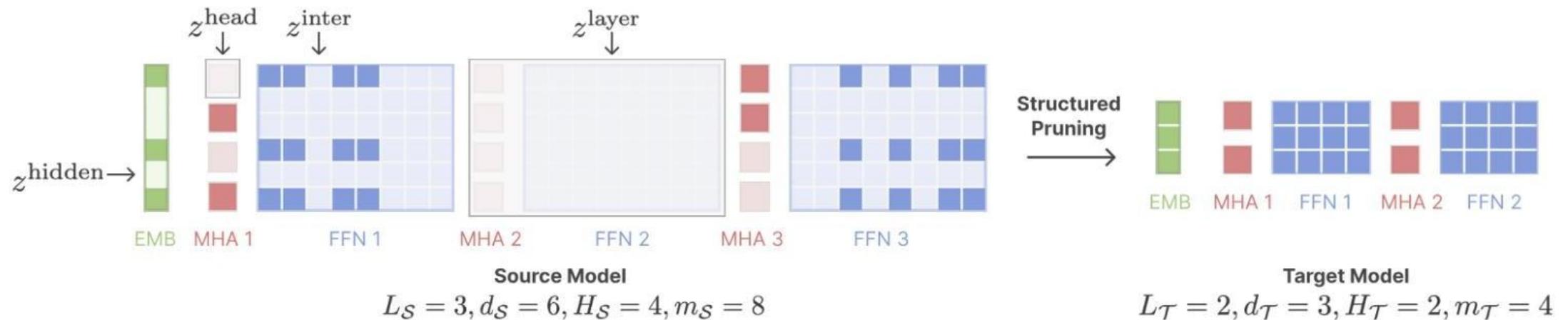# LLM-Shearing: Accelerating via Structured Pruning

An efficient method of constructing LLMs by first pruning a larger existing model and then continually pre-training it.
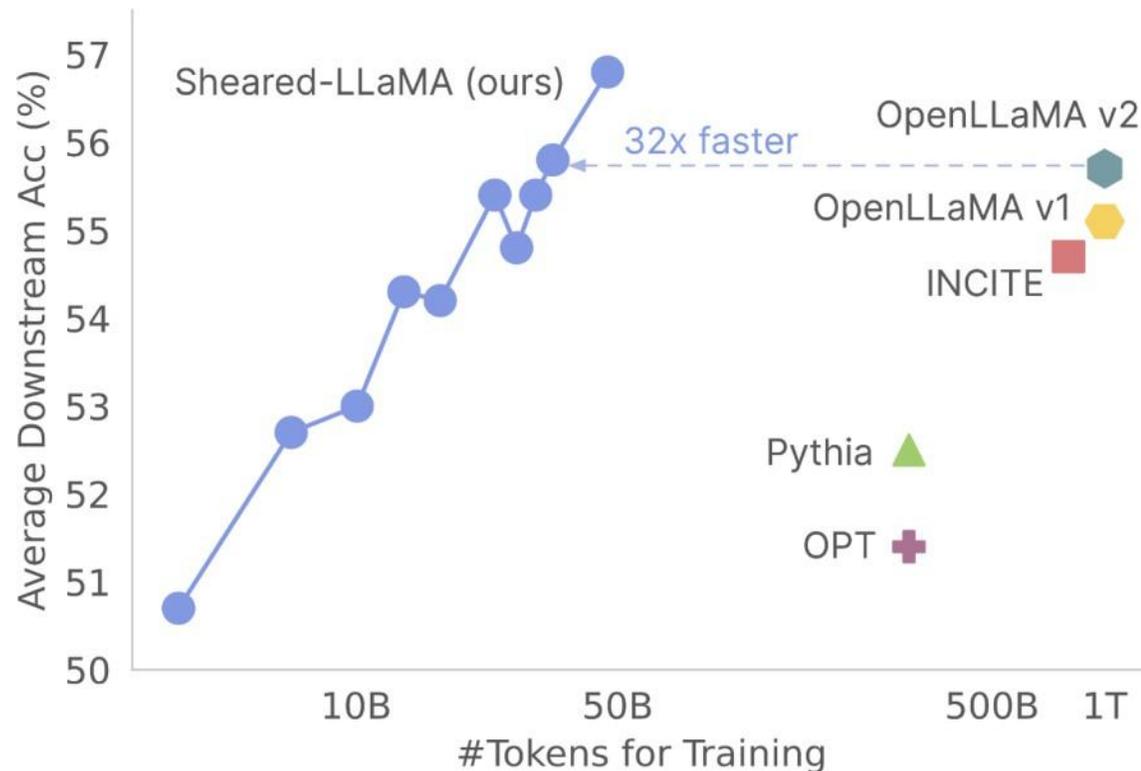

- Sheared-LLaMA-2.7B achieves better performance than existing open-source models of the same scale with 3% (1/32) of the compute.
- The trajectory shows a compelling case that if we invest more tokens and compute, the capability of Sheared-LLaMA can be further improved.
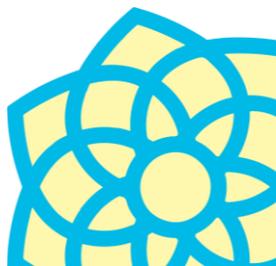
LLM-Shearing: Accelerating via Structured Pruning

1. Target Structure Pruning: prune a source model to a pre-specified target architecture (e.g., an existing model's config), and meanwhile maximizing the pruned model's performance
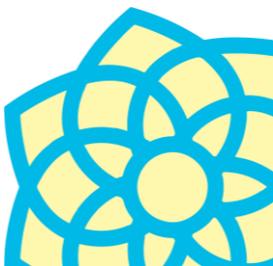
LLM-Shearing: Accelerating via Structured Pruning

Algorithm 1: Dynamic Batch Loading

Require: Training data of k domains $D_1, D_2, \cdots, D_k$, validation data $D_1^{\text{val}}, D_2^{\text{val}}, \cdots, D_k^{\text{val}}$, initial data loading weights $w_0 \in \mathbb{R}^k$, reference loss $\ell_{\text{ref}} \in \mathbb{R}^k$, LM loss function $\mathcal{L}$ or pruning loss $\mathcal{L}_{\text{prune}}$, training steps T, evaluation interval m, model parameters θ (θ, z, ϕ, λ for pruning)

for $t = 1, \cdots, T$ **do**
    if $t \mod m = 0$ **then**
        $\ell_t[i] \leftarrow \mathcal{L}(\theta, z, D_i^{\text{val}})$ if *pruning* else $\mathcal{L}(\theta, D_i^{\text{val}})$
        $\Delta_t[i] \leftarrow \max\{\ell_t[i] - \ell_{\text{ref}}[i], 0\}$ ▷ Calculate loss difference
        $w_t \leftarrow \texttt{UpdateWeight}(w_{t-m}, \Delta_t)$ ▷ Update data loading proportion
    end
    Sample a batch of data $\mathcal{B}$ from $D_1, D_2, \cdots, D_k$ with proportion w_t;
    if *pruning* **then**
        Update θ, z, ϕ, λ with $\mathcal{L}_{\text{prune}}(\theta, z, \phi, \lambda)$ on $\mathcal{B}$
    else
        Update θ with $\mathcal{L}(\theta, \mathcal{B})$
    end
end

Subroutine $\texttt{UpdateWeight}(w, \Delta)$
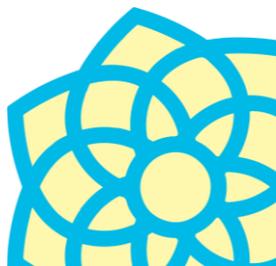 $\alpha \leftarrow w \cdot \exp(\Delta)$ ▷ Calculate the unnormalized weights
    $w \leftarrow \frac{\alpha}{\sum_i \alpha[i]}$ **return** w ▷ Renormalize the data loading proportion
return θ
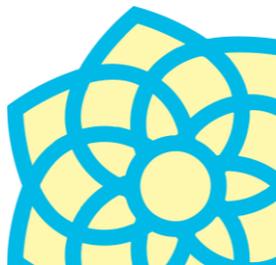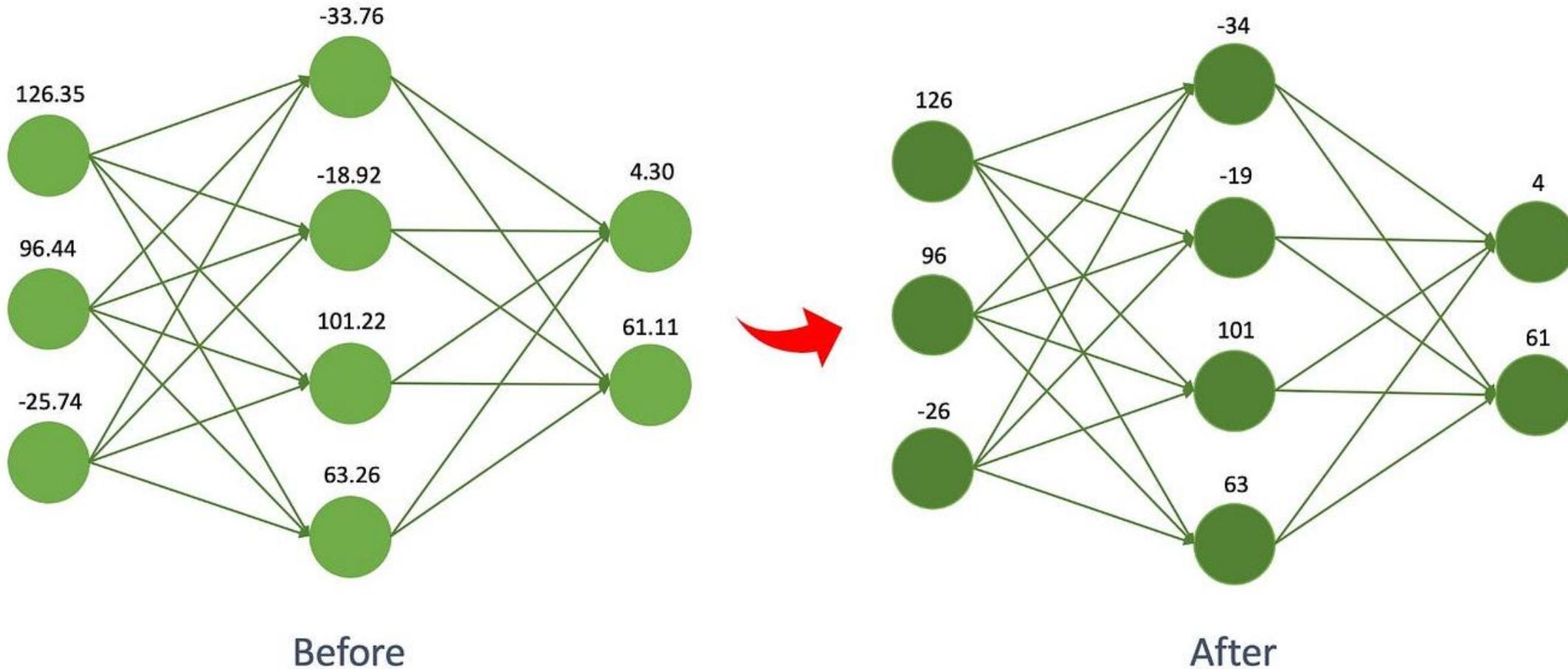
2. Dynamic batch loading: Pruning results in varying information retainment across domains. Concretely, they load more data for domains that recover slow, and the loading proportion is dynamically decided on the fly.
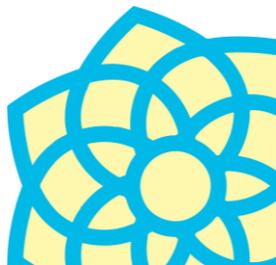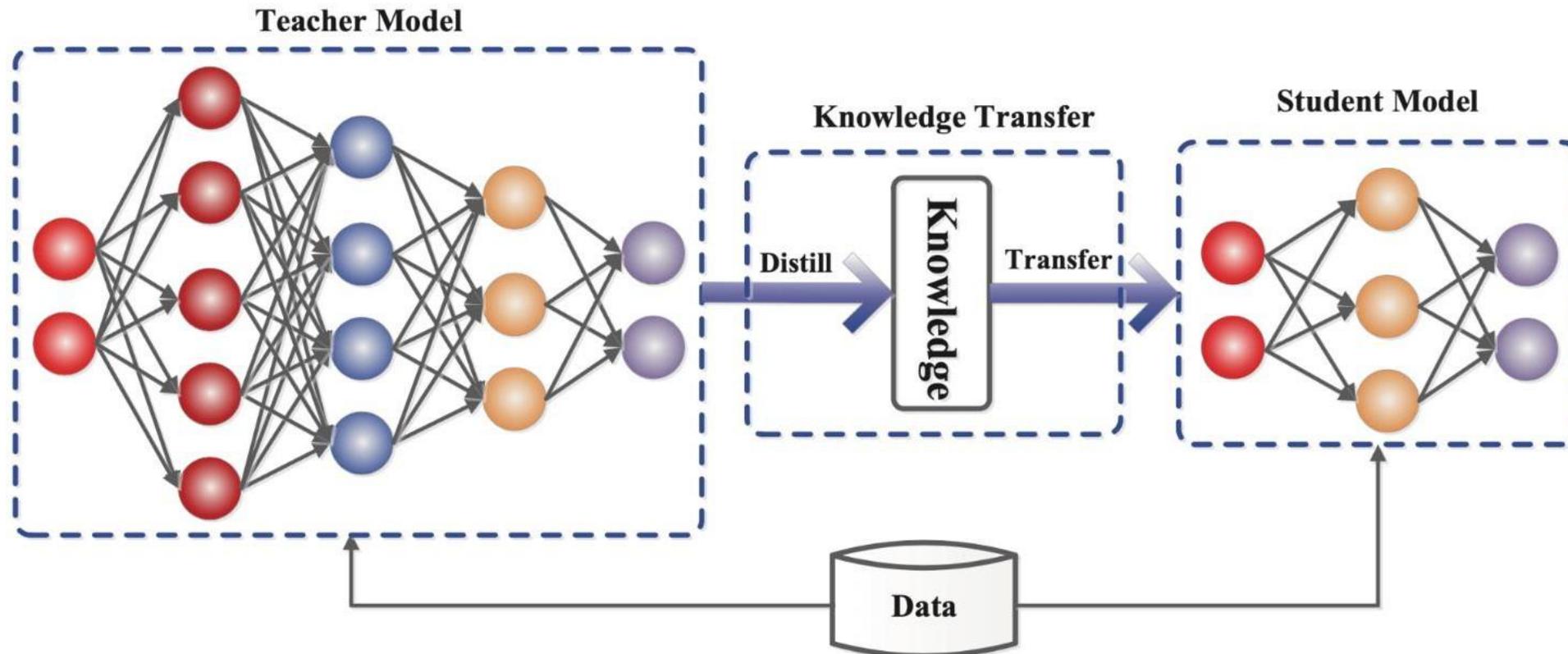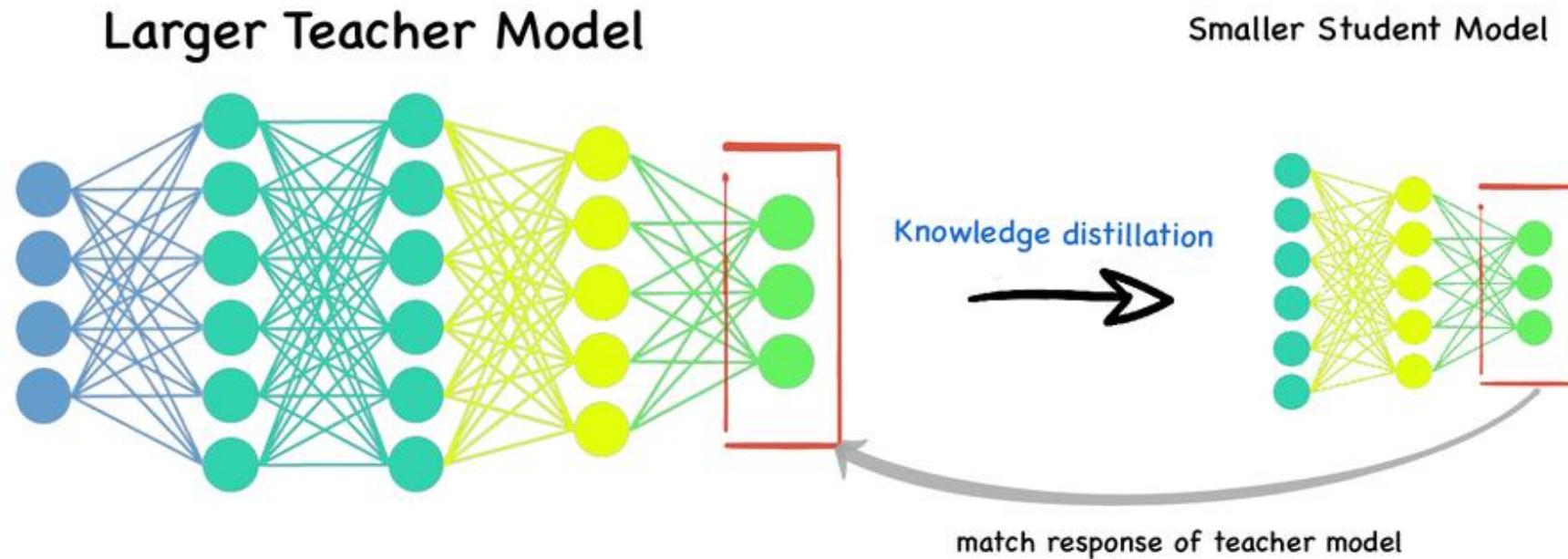
Quantization

Changing the precision of the model weights to make them smaller

Before

After

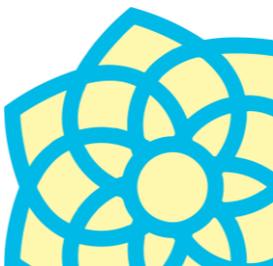# Framework of knowledge distillation

Knowledge Distillation

- **LAIM LE5 VT2025:**
  Instruction fine-tuning
  Preference tuning (RLHF, DPO)
  Pruning and Distilling

www.ida.liu.se/~frehe08/llm

LIU LINKÖPING UNIVERSITY