UNIVERSITY of VIRGINIA

# Week 1

# W1.2- LLM Alignment – Basics and Some Advanced

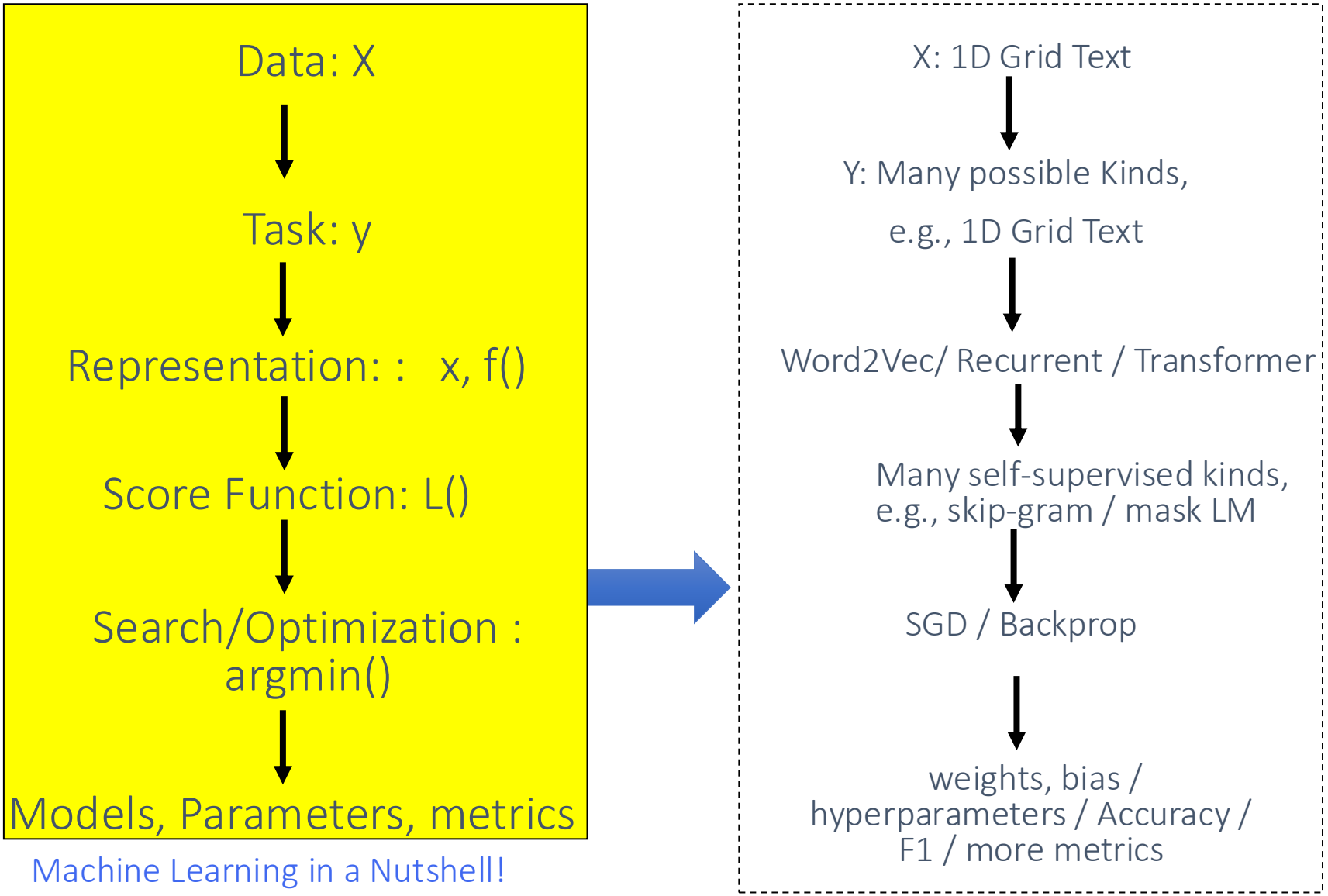
2026 Spring

LLM Agents Foundation & Applications

Dr. Yanjun Qi

20260115

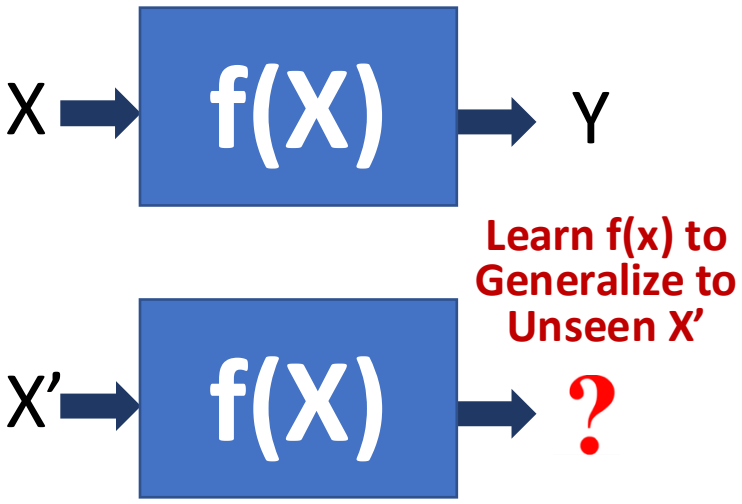
# Prerequisites: Deep Neural Network Models Basics / E.g., DNN on Text




Machine Learning in a Nutshell!

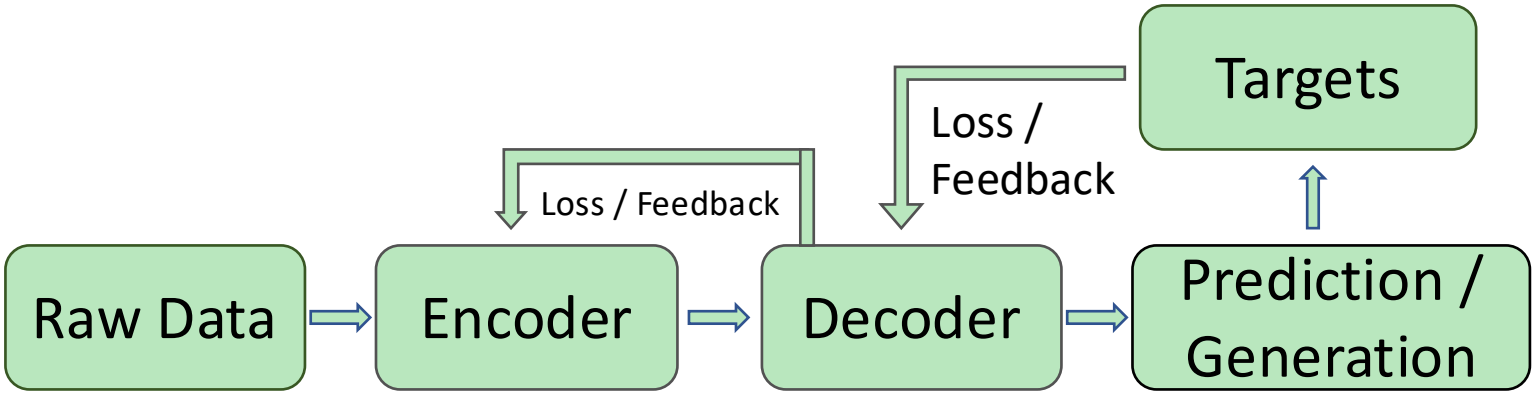
# Prerequisites: Data-Driven Machine Learning for AI

- Need inductive reasoning
  - Generalizations from observed data to unseen data




- Able to to build computer systems that can learn and adapt from their experience
- Well-engineered software architectures to build upon
- Provide prediction accuracy
- Create software that improves over time

# Prerequisites: Basics on Deep Learning

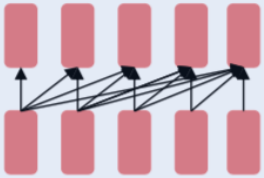



Text / DNA / …
Images / Audio
Discrete/continuous values
Structured/unstructured
Clean/noisy labels

Low dimensional embedding
Category
Generated text, image, audio

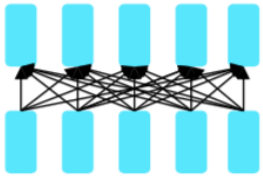
GPT: **G**enerative **P**retraining **T**ransformer models for Language
CLIP: **C**ontrastive **L**anguage-**I**mage **P**retraining for Vision
BERT: **B**idirectional **Encoder R**epresentations from **Transformers**.

# Prerequisites : Three types of popular transformer architectures



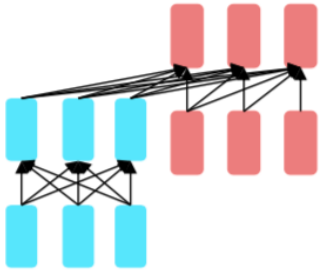
**Decoders**

- Nice to generate from; can't condition on future words
- **Examples:** GPT-2, GPT-3, LaMDA



**Encoders**

- Gets bidirectional context – can condition on future!
- Wait, how do we pretrain them?
- **Examples:** BERT and its many variants, e.g. RoBERTa



**Encoder-Decoders**

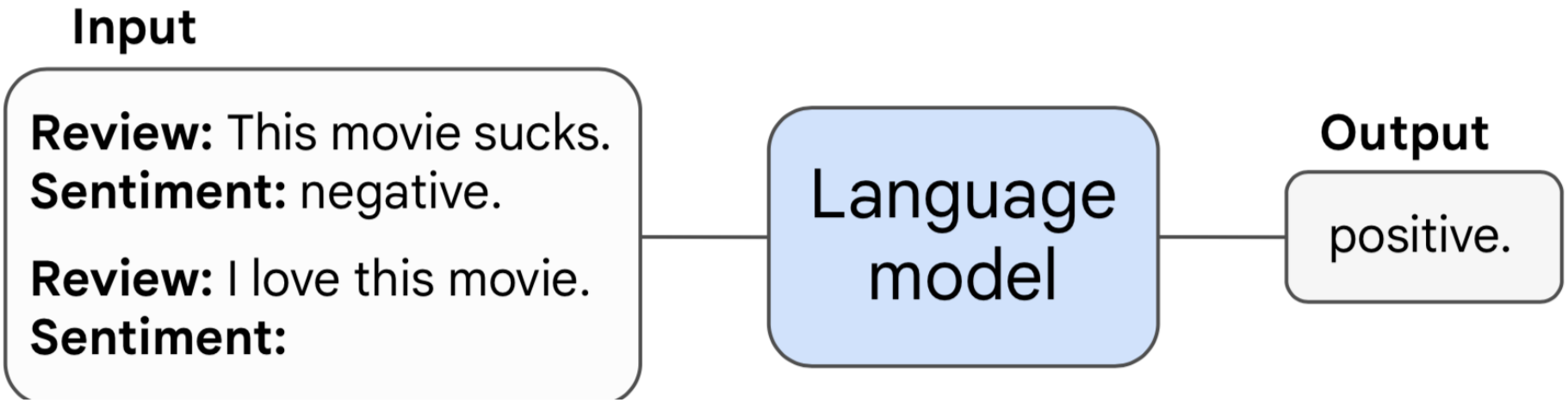
- Good parts of decoders and encoders?
- What's the best way to pretrain them?
- **Examples:** T5, Meena

# Current: General Deep learning in AI

- To Complex tasks
  - E.g., generating slides from an outline, summarizing and reporting information from diverse sources
- Integrating into physical devices
  - E.g., Robots
- Multimodal and broadly
  - Use vision, language, audio, and broader knowledge like databases, as input or outputs, plus other type of modality like sensor, DNA, protein, …
- Complex learning systems
  - Integrate predictive/generative
  - Integrate retrieval of private memories or data
  - Integrate with planning, task decomposition, and prioritization

# Basics: Emergent Abilities of Large Language Models

e.g., Few-shot “In Context Learning” ability




*An ability is emergent if it is not present in smaller models but is present in larger models.*

Larger GPT models trained on massive data are good at many tasks, especially text generation, and can be “trained” at inference time with in-context examples

# GPT3 : Models and Architectures

| Model Name | $n_{\text{params}}$ | $n_{\text{layers}}$ | $d_{\text{model}}$ | $n_{\text{heads}}$ | $d_{\text{head}}$ | Batch Size | Learning Rate |
|---|---|---|---|---|---|---|---|
| GPT-3 Small | 125M | 12 | 768 | 12 | 64 | 0.5M | $6.0 \times 10^{-4}$ |
| GPT-3 Medium | 350M | 24 | 1024 | 16 | 64 | 0.5M | $3.0 \times 10^{-4}$ |
| GPT-3 Large | 760M | 24 | 1536 | 16 | 96 | 0.5M | $2.5 \times 10^{-4}$ |
| GPT-3 XL | 1.3B | 24 | 2048 | 24 | 128 | 1M | $2.0 \times 10^{-4}$ |
| GPT-3 2.7B | 2.7B | 32 | 2560 | 32 | 80 | 1M | $1.6 \times 10^{-4}$ |
| GPT-3 6.7B | 6.7B | 32 | 4096 | 32 | 128 | 2M | $1.2 \times 10^{-4}$ |
| GPT-3 13B | 13.0B | 40 | 5140 | 40 | 128 | 2M | $1.0 \times 10^{-4}$ |
| GPT-3 175B or "GPT-3" | 175.0B | 96 | 12288 | 96 | 128 | 3.2M | $0.6 \times 10^{-4}$ |

**Table 2.1:** Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

12020

**Language Models are Few-Shot Learners**

**Tom B. Brown*** **Benjamin Mann*** **Nick Ryder*** **Melanie Subbiah***

**Jared Kaplan†** **Prafulla Dhariwal** **Arvind Neelakantan** **Pranav Shyam** **Girish Sastry**

**Amanda Askell** **Sandhini Agarwal** **Ariel Herbert-Voss** **Gretchen Krueger** **Tom Henighan**

**Rewon Child** **Aditya Ramesh** **Daniel M. Ziegler** **Jeffrey Wu** **Clemens Winter**

# GPT-3 (Brown et al. 2020): few shot generalization

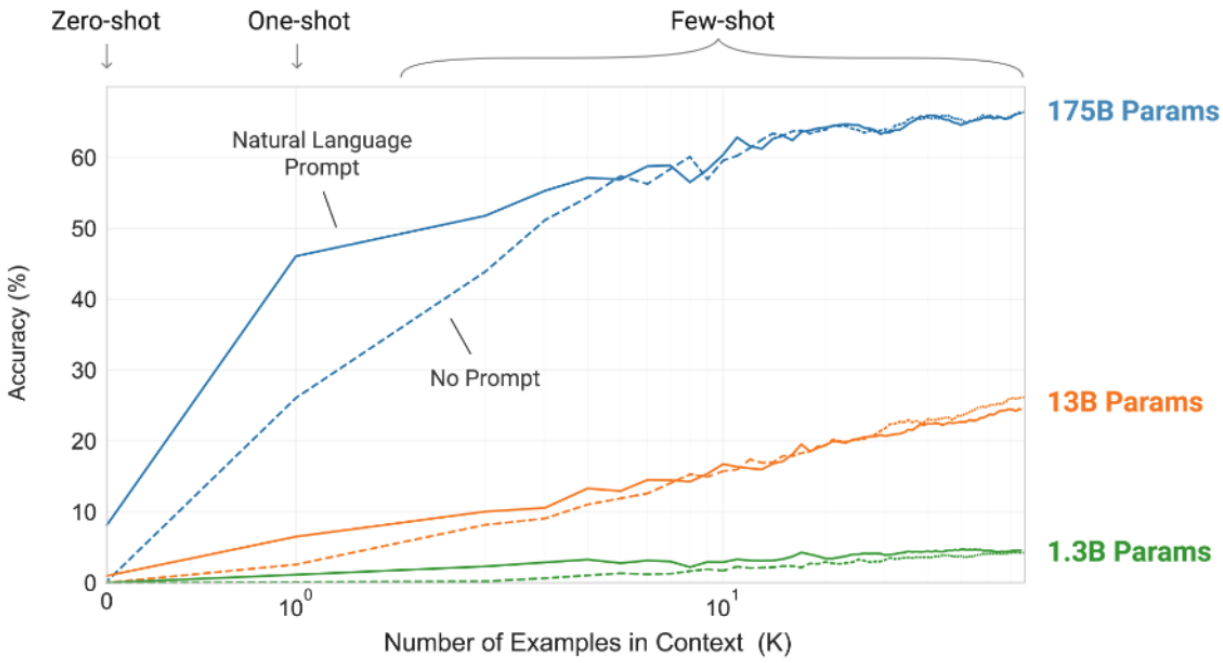
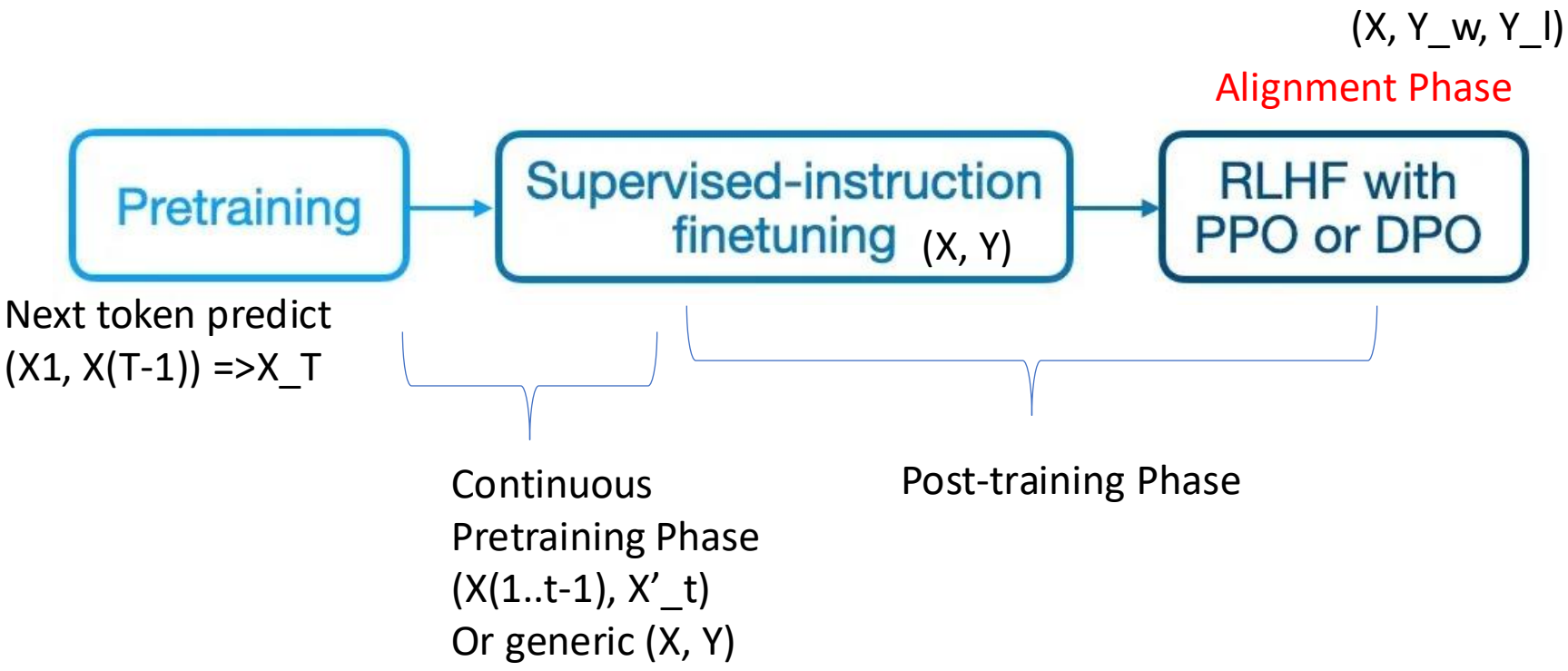


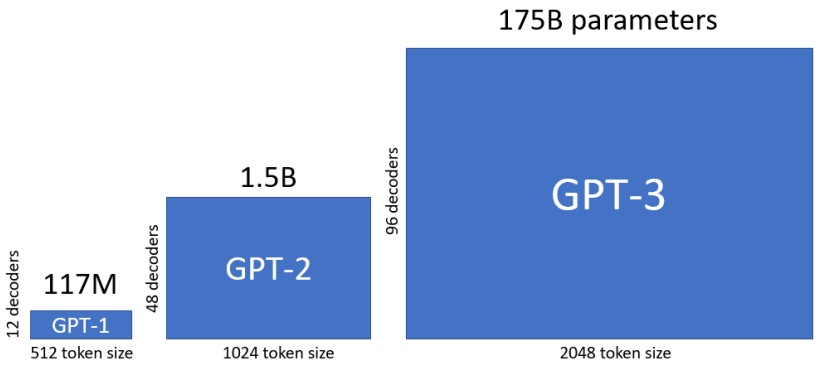

Fig. 9: GPT-3 shows that larger models make increasingly efficient use of in-context information. It shows in-context

# Basics: Training Foundation Models Basic Flow



# Summary of Last Class:

- GPT1 / 2/ 3
- Emergent Abilities of Large Language Models
- Scaling Instruction-Finetuned Language Models
- On the Opportunities and Risks of Foundation Models




https://medium.com/@YanAIx/step-by-step-into-gpt-70bc4a5d8714

# This Class:

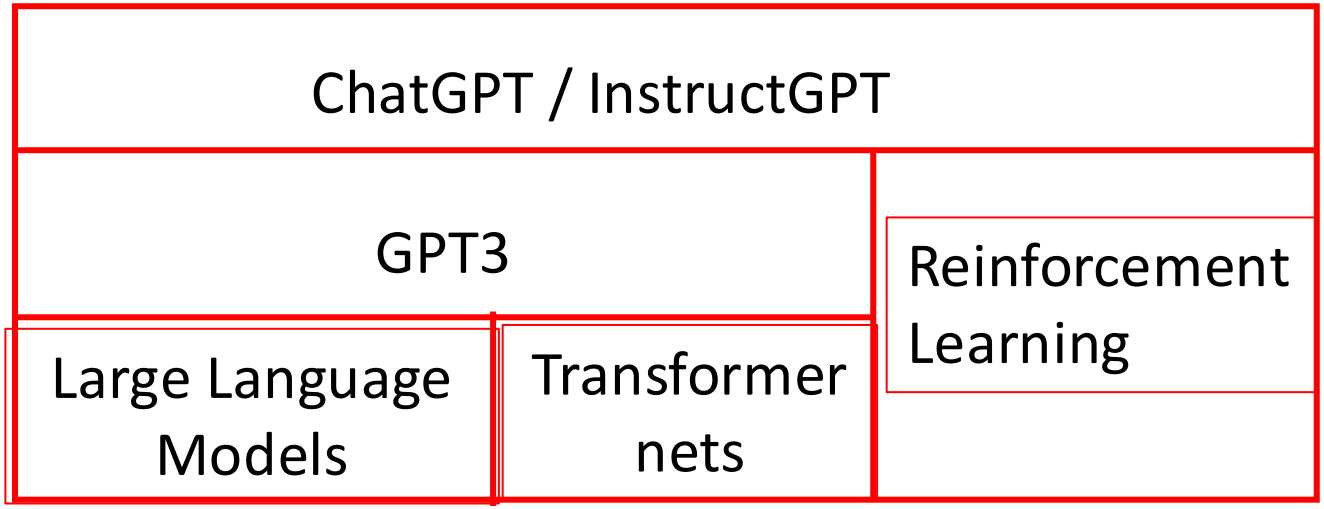
ChatGPT

LLM alignment Basics

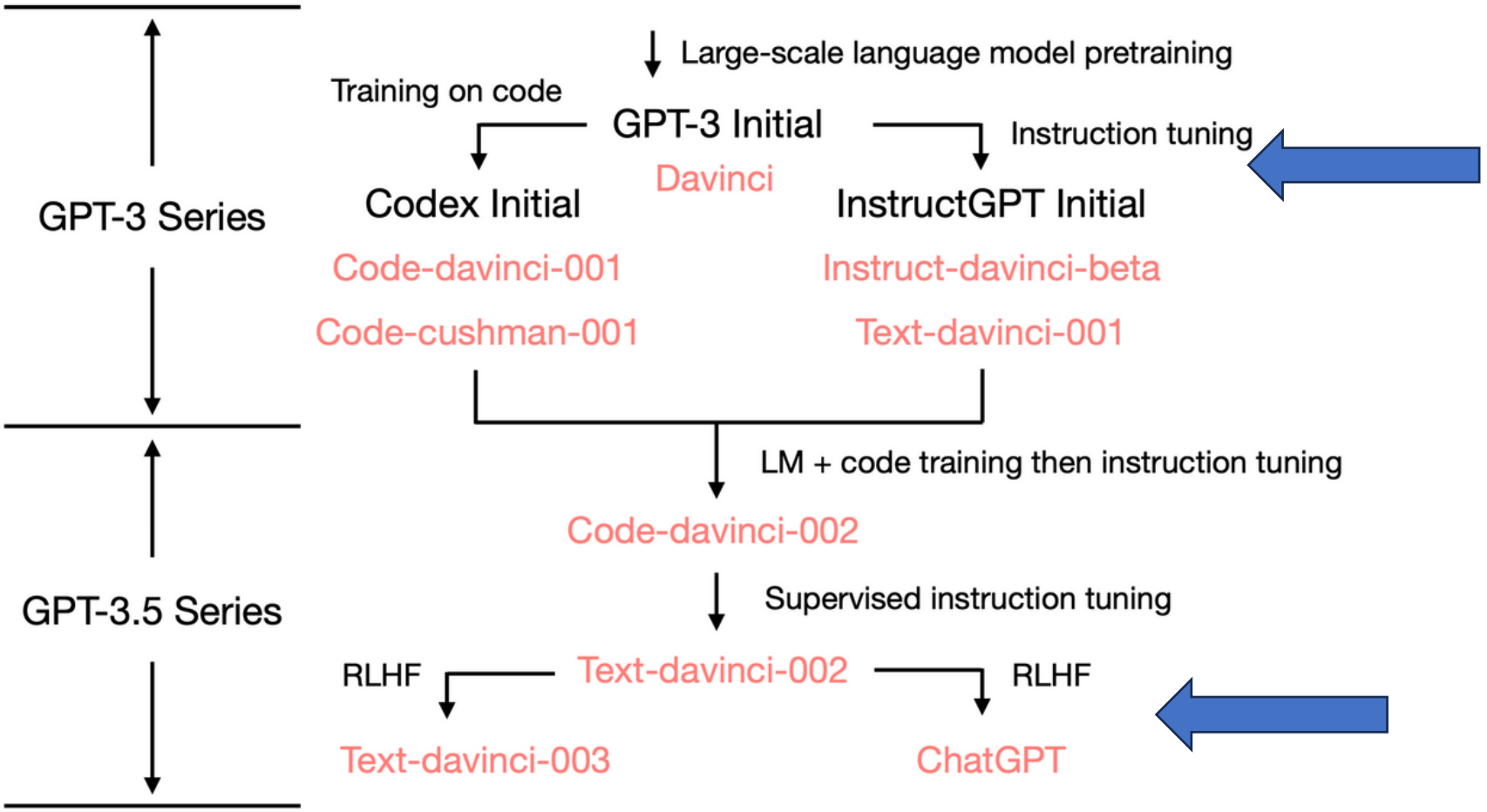
# ChatGPT: Optimizing Language Models for Dialogue” by OpenAI

- No paper / Just a blog / Released Nov 30 2022
- It took 5 days to reach 1M users

# Concepts that ChatGPT builds on



# Family of GPT-3.5




Blog: [How does GPT Obtain its Ability? Tracing Emergent Abilities of Language Models to their Sources]

# Initial Papers on RLHF

**Training language models to follow instructions with human feedback**

**Long Ouyang*** **Jeff Wu*** **Xu Jiang*** **Diogo Almeida*** **Carroll L. Wainwright***

**Pamela Mishkin*** **Chong Zhang** **Sandhini Agarwal** **Katarina Slama** **Alex Ray**

**John Schulman** **Jacob Hilton** **Fraser Kelton** **Luke Miller** **Maddie Simens**

**Amanda Askell†** **Peter Welinder** **Paul Christiano*†**

**Jan Leike*** **Ryan Lowe***

OpenAI

**Proximal Policy Optimization Algorithms**

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
{joschu, filip, prafulla, alec, oleg}@openai.com

**Deep Reinforcement Learning from Human Preferences**

**Paul F Christiano**
OpenAI
paul@openai.com

**Jan Leike**
DeepMind
leike@google.com

**Tom B Brown**
nottombrown@gmail.com

**Miljan Martic**
DeepMind
miljanm@google.com

**Shane Legg**
DeepMind
legg@google.com

**Dario Amodei**
OpenAI
damodei@openai.com

**Learning to summarize from human feedback**

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

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

OpenAI

# Why Aligning Large Language Models?

**LMs like GPT-3 are misaligned:** they maximize the data likelihood of large untrusted datasets.

This leads to:

- Not following the user's instruction
- Making up facts
- Generating harmful/toxic content
- ……

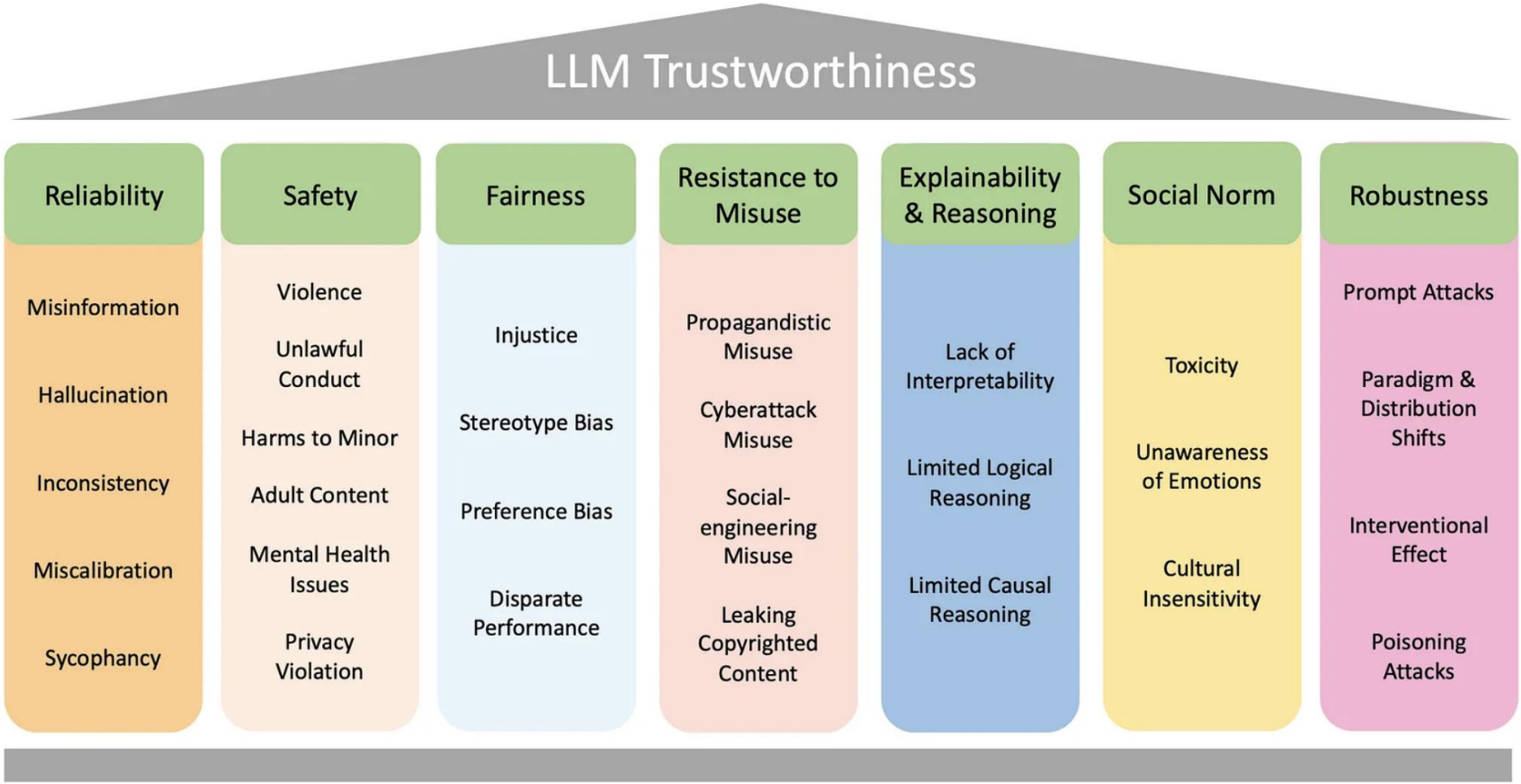
Explain the moon landing to a 6 year old in a few sentences.

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.
```

*Pretrained Language models are not aligned with user intent.*

# LLM Alignment



# Alignment through human preference data

**Q:** Human judgments are noisy and miscalibrated!

**Solution:** Use pairwise comparisons instead of direct ratings.

➔ Human Preference Data

| | | |
|---|---|---|
| An earthquake hit San Francisco. There was minor property damage, but no injuries. | > | A 4.2 magnitude earthquake hit San Francisco, resulting in massive damage. |

# Two different Post-Training Preference Alignment (Preference Optimization: PO) Strategies:

- Different approaches to incorporate human preferences:
  - **RL algorithms** (PPO, GRPO, REINFORCE) explicitly maximize expected reward from a reward model – we normally call this group **RLHF**
  - **Direct alignment methods** (DPO, IPO, KTO) optimize preference objectives without explicit reward modeling
    - Though they can be shown to implicitly optimize an equivalent objective under certain assumptions

# (1) RLHF

## Reinforcement Learning from Human Feedback

(Very!) brief introduction on Reinforcement Learning (RL):

**Reinforcement Learning = Learning by Doing and Getting Feedback**

- An agent (LLM) interacts with an environment (e.g., human) and learns by trial and error.
- Large Rewards (✔ Correct answer!) encourage desirable actions.
- Small Rewards (✘ Incorrect response!) discourage undesirable actions.
- RL algorithms (e.g., PPO, GRPO) train agent (e.g. LLM) to maximize expected reward.



https://www.mathworks.com/discovery/reinforcement-learning.html

# Aligning LMs with Human Preference Feedback

Suppose we are training an LLM for a summarization task.

For a given instruction $x$ and a generated summary $y$, we assume we can obtain a human reward of that summary:

$R(x, y)$ — where higher values indicate better quality.

$x$

```
SAN FRANCISCO,
California (CNN) --
A magnitude 4.2
earthquake shook the
San Francisco
...
overturn unstable
objects.
```

$y_1$

$R(x, y_1) = 8.0$

```
An earthquake hit
San Francisco.
There was minor
property damage,
but no injuries.
```

$y_2$

$R(x, y_2) = 1.2$

```
The Bay Area has
good weather but is
prone to
earthquakes and
wildfires.
```

We want to maximize the expected reward based on this feedback.

# RLHF – HF is expensive; So modeling the reward

**Q:** Human-in-the-loop is expensive!

**Solution:** Instead of asking humans directly, we train a separate reward model to learn human preferences.

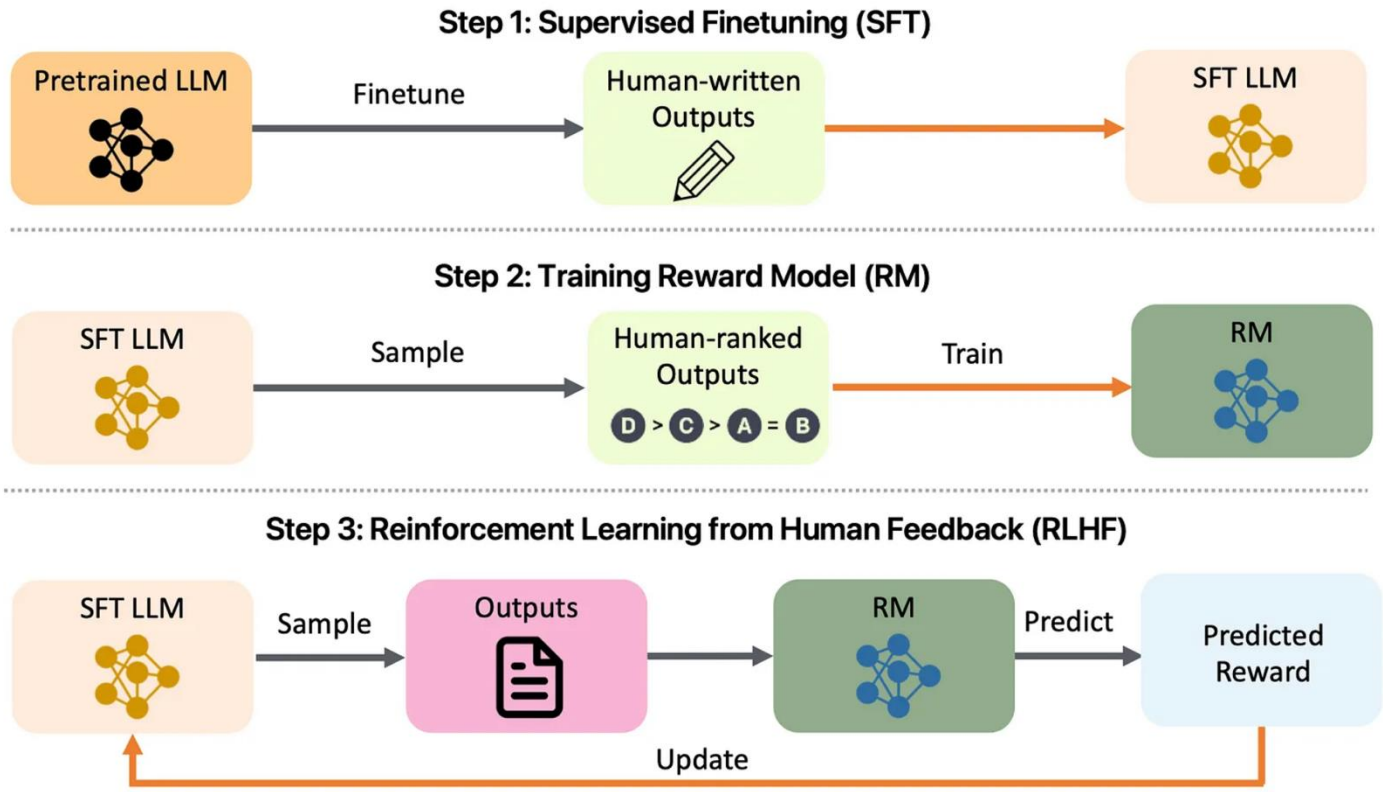
$$L_{\mathrm{RM}}(r_\phi) = -\frac{1}{C_K^2}\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\phi(x,y_w) - r_\phi(x,y_l)\right)\right)\right]$$

$y_w$: winning sample

$y_l$: losing sample

$y_w$ should score higher than $y_l$

# Reinforcement Learning from Human Feedback



# RLHF (cont.)

We have the following:

- A pretrained (possibly instruction-finetuned) LM $\pi_{ref}(y|x)$
- A reward model $r_\phi(x, y)$ that produces scalar rewards for LM outputs, trained on a dataset of human comparisons

Now to do RLHF:

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

Maximizing rewards

Minimizing divergence between current policy and reference policy

# Results

InstructGPT: scaling up RLHF to 30k tasks!

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.
```

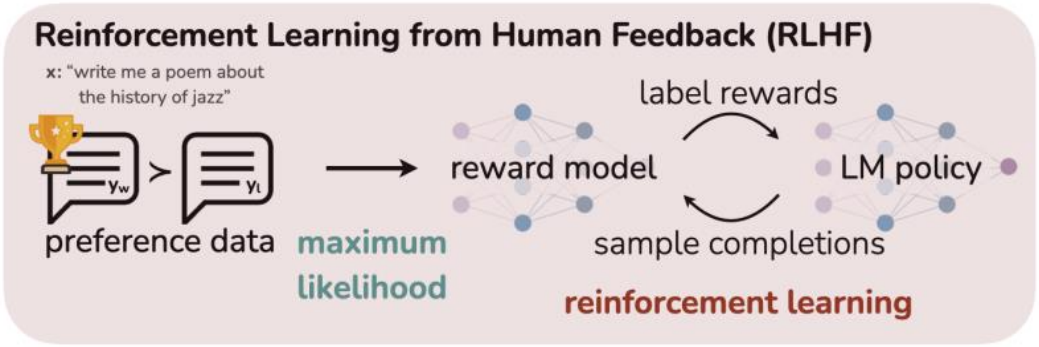
# RLHF (cont.)

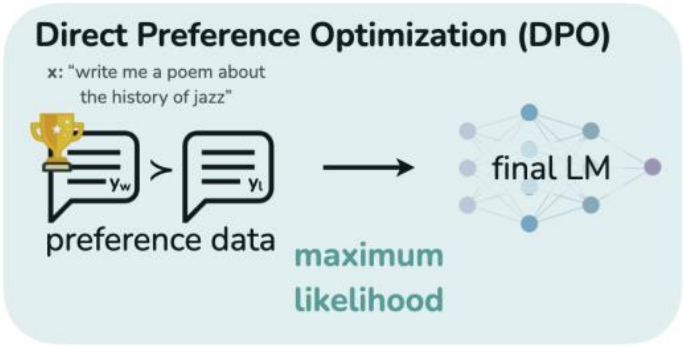
- RLHF suffers 3 major challenges
  - RLHF requires training a reward model, which is computationally expensive
  - RLHF PPO requires the LLM to rollout during RL optimization
  - RLHF PPO is unstable and suffers from convergence issue

# (2) Simplify RLHF? Towards DPO

## Direct Preference Optimization (DPO)

- RLHF is a complex and often unstable procedure for alignment
- Direct preference optimization (DPO) simplifies RLHF to a classification loss
- SimPO improves DPO efficiency by removing the reference policy
- DPOP improves the training stability with an supervised learning objective



# (2) Can we simplify RLHF? Towards DPO

**Direct Preference Optimization (DPO):** directly optimizes policy based on human preference data using a clever loss function.

Recall our objective in RLHF:

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

There is a closed form solution to this:

$$\pi_\theta(y|x) = \frac{1}{Z(x)}\pi_{\mathrm{ref}}(y|x)e^{\left(\frac{1}{\beta}r_\theta(x,y)\right)}$$

Rearrange the terms:

$$r_\theta(x,y) = \beta\log\left(\frac{\pi_\theta(y|x)}{\pi_{\mathrm{ref}}(y|x)}\right) + \beta\log Z(x)$$

Reward model can be written in terms of policy!

# Can we simplify RLHF? Towards DPO

**Direct Preference Optimization (DPO):** directly optimizes policy based on human preference data using a clever loss function.

Recall, how we fit the reward model in RLHF:

$$L_{\mathrm{RM}}(r_\phi) = -\frac{1}{C_K^2}\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\left(\sigma\left(r_\phi(x,y_w) - r_\phi(x,y_l)\right)\right)\right]$$

Notice that we only need the difference between the rewards. Simplify for rewards:

$$r_\theta(x,y_w) - r_\theta(x,y_l) = \beta\left[\log\left(\frac{\pi_\theta(y_w|x)}{\pi_{\mathrm{ref}}(y_w|x)}\right) - \log\left(\frac{\pi_\theta(y_l|x)}{\pi_{\mathrm{ref}}(y_l|x)}\right)\right]$$

The final DPO loss function is:

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

We have a classification loss function that connects **preference data** to **LM parameters** directly!

# Summary (RLHF and DPO)

- Our goal is to optimize for Human Preferences
  - Instead of humans writing the answers or giving uncalibrated scores, we get humans to rank different LM generated answers.
- RLHF
  - Step 1: Supervise fine-tuning on a labeled dataset
  - Step 2: Train an explicit reward model on comparison data to predict a score for a completion
  - Step 3: Optimize the LM to maximize the predicted score (under KL-constraint)
  - Very effective when tuned well, computationally expensive
- DPO
  - Optimize LM parameters directly on preference data by solving a binary classification problem
  - Simple and effective, similar properties to RLHF

# Human Preference Optimization Objective

$(x_{prompt}, y_{win}, y_{lose})$

$\pi_\theta$: LLM policy
$\pi_{Ref}$: base LLM
*x*: prompt
*y*: completion

- Reinforcement Learning with Human Feedback (RLHF) Objective:

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

Optimize "reward" *inspired* ▲ by human preferences

▲ Constrain the target LLM model to stay close to the base LM

- Direct preference optimization (DPO) simplifies RLHF to a classification loss

$$\theta^* = arg\max_\theta \{ -\mathbb{E}_{(x,y_w,y_l)\sim D} \log \left\{ \sigma \left[ \beta \log \left( \frac{\pi_\theta(y_w|x)}{\pi_{\mathrm{ref}}(y_w|x)} \right) - \beta \log \left( \frac{\pi_\theta(y_l|x)}{\pi_{\mathrm{ref}}(y_l|x)} \right) \right] \right\}$$

Credit: Ai2

# RLHF and DPO optimize an equivalent PO objective

- Via ”Implicit Reward” reformulation:

$$r_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{y}) := \beta \log \frac{\pi_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}|\mathbf{x})}$$

[Rafailov et al. 2024]

$$\pi_\theta(y|x) = \frac{1}{Z(x)} \pi_{\text{ref}}(y|x) e^{\left(\frac{1}{\beta} r_\theta(x,y)\right)}$$

# More Backups

A Comprehensive Survey of Direct Preference Optimization: Datasets, Theories, Variants, and Applications, Xiao et al. 25



# Research directions of LLM alignment

Reward model

Feedback

RL policy

Optimization

# Reward model

- **Explicit Reward Model vs. Implicit Reward Model**
  - e.g., RLHF vs. DPO
- **Pointwise Reward Model vs. Preferencewise Model**
  - $R(x, y)$ vs. prob. that the desired response is preferred over the undesired one
- **Response-Level Reward vs. Token-Level Reward**
  - Assign a single score to the entire response vs. provide feedback at each token
- **Negative Preference Optimization**
  - Use only prompts and undesired responses from RLHF datasets, generating desired responses with LLMs instead of relying on human-labeled preferred responses

# Feedback

- **Preference Feedback vs. Binary Feedback**
  - Rank responses vs. simple positive or negative signal without ranking
- **Pairwise Feedback vs. Listwise Feedback**
  - Compare two responses vs. rank multiple responses together
- **Human Feedback vs. AI Feedback**
  - Real user preferences vs. LLM-generated evaluations

# RL

- **Reference-Based RL vs. Reference-Free RL**
  - Minimize divergence from a reference policy vs. remove reference policy (e.g. SimPO)
- **Length-Control RL**
  - Standard RL ignores response length. Length-control RL adjusts rewards to prevent verbosity bias in LLM-generated responses. E.g., R-DPO and SimPO.
- **Different Divergences in RL**
  - KL divergence, f-divergence, ……
- **On-policy or Off-policy Learning**
  - Generate responses using the latest policy vs. reuse past responses

# Optimization

- **Iterative/Online Preference Optimization vs. Non-Iterative/Offline Preference Optimization**
  - Continuously update alignment with new data vs. align models using a fixed dataset
- **Separating SFT and Alignment vs. Merging SFT and Alignment**
  - Newer approaches integrate SFT and alignment into a single process, e.g., ORPO, PAFT.