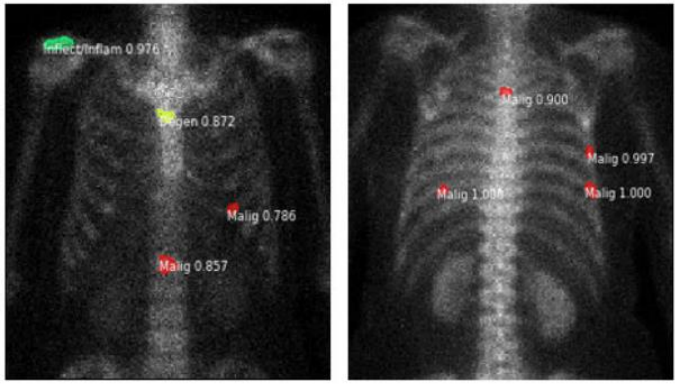
Large Language Model

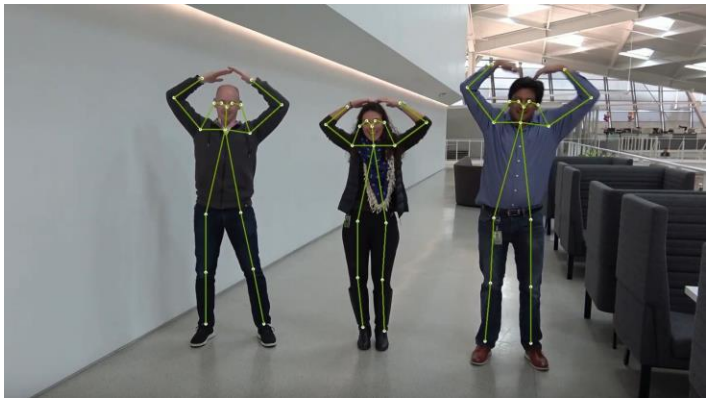
Outline

- Generative AI
- Emergent properties & scaling laws
- Alignment
  - Instruction tuning
  - Preference learning

What is generative AI?

- Predictive AI
  - predicts
  - classify





https://developer.nvidia.com/blog/training-optimizing-2d-pose-estimation-model-with-tao-toolkit-part-1/

What is generative AI?

- Generates
  - Creative tasks that was believe to be hard for AI a couple years ago

https://knowyourmeme.com/memes/ai-will-smith-eating-spaghetti

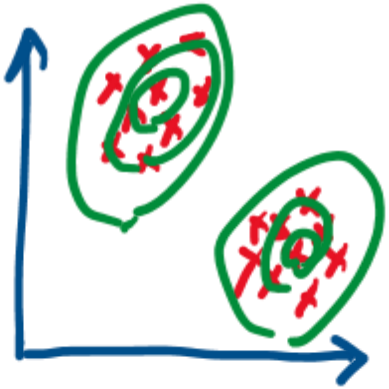
https://x.com/jerrod_lew/status/1868809004400754871

https://www.tiktok.com/@willsmith/video/7337480720115371295?lang=en

Generative modeling

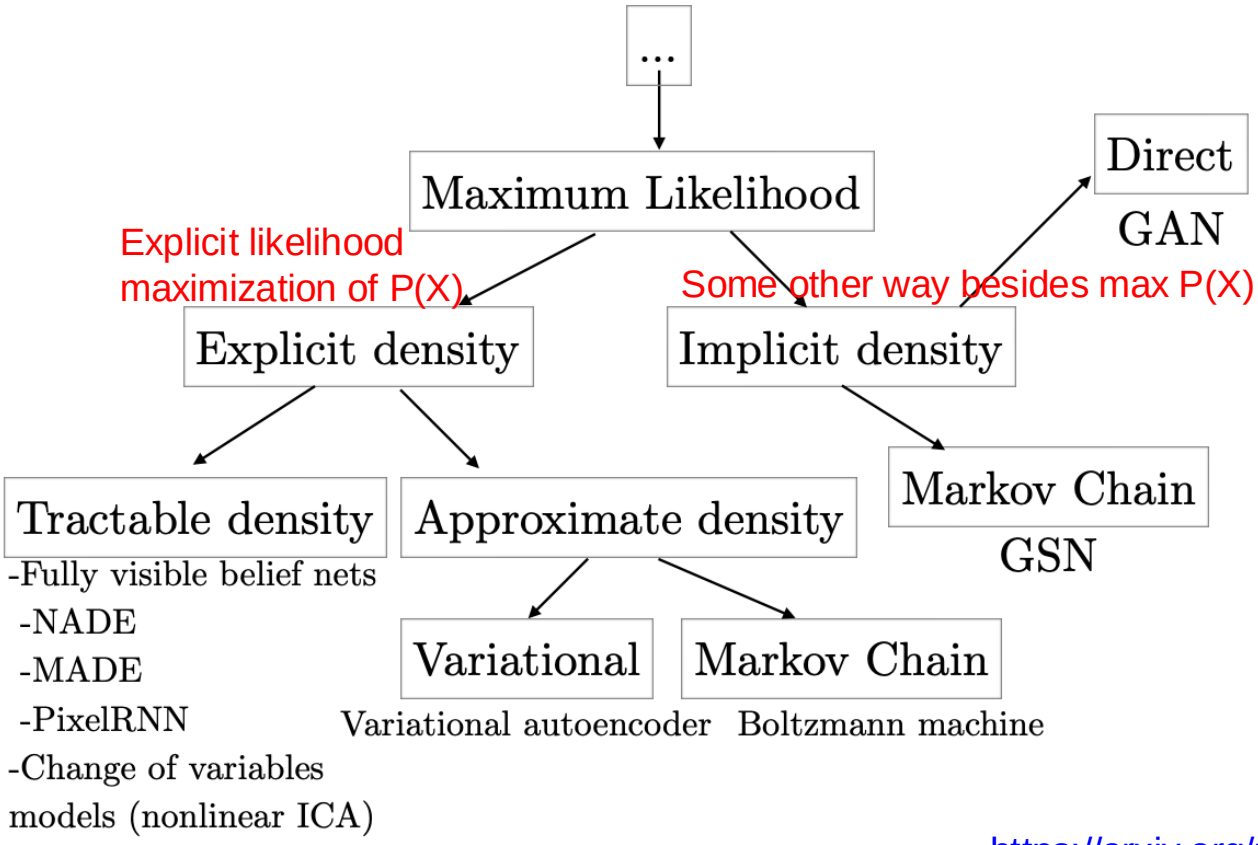
- Learns P(X,Y) or P(X|Y)
  - Can sample (generates) X from P(X|Y)
  - Y is the controlling parameter

- What is P(X|Y)?
  - In the past, it's often assumed to be a parametric distribution



Enter deep generative models

- Distribution learning landscape



Diffusion/score-based model

https://arxiv.org/pdf/1701.00160.pdf
https://yang-song.net/blog/2021/score/

Text generation

- Unlike other generative tasks, turns out that explicit density methods (autoregressive prediction) works well for text
- There are attempts to use GANs/RLs/Diffusion/VAE for text generation but they are not as competitive against simple cross entropy based prediction
  - Scales well with very large data
  - Recent research starts looking into ways to use diffusion/VAE in text generation

Pretrained Foundation Models/LLM

- With self-supervised learning, Ex. contrastive learning, next token prediction, mask token prediction
  - People found that large model produces better results

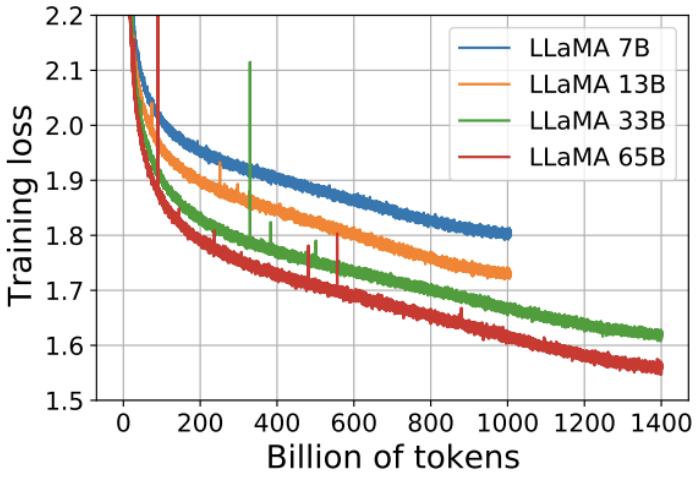
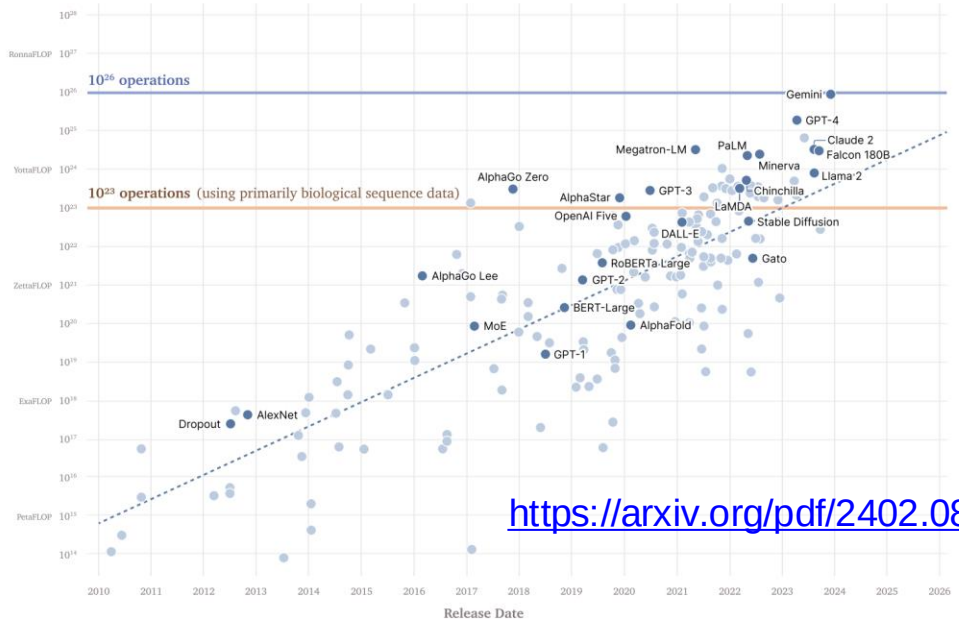


Figure 1: **Training loss over train tokens for the 7B, 13B, 33B, and 65 models.** LLaMA-33B and LLaMA-65B were trained on 1.4T tokens. The smaller models were trained on 1.0T tokens. All models are trained with a batch size of 4M tokens.



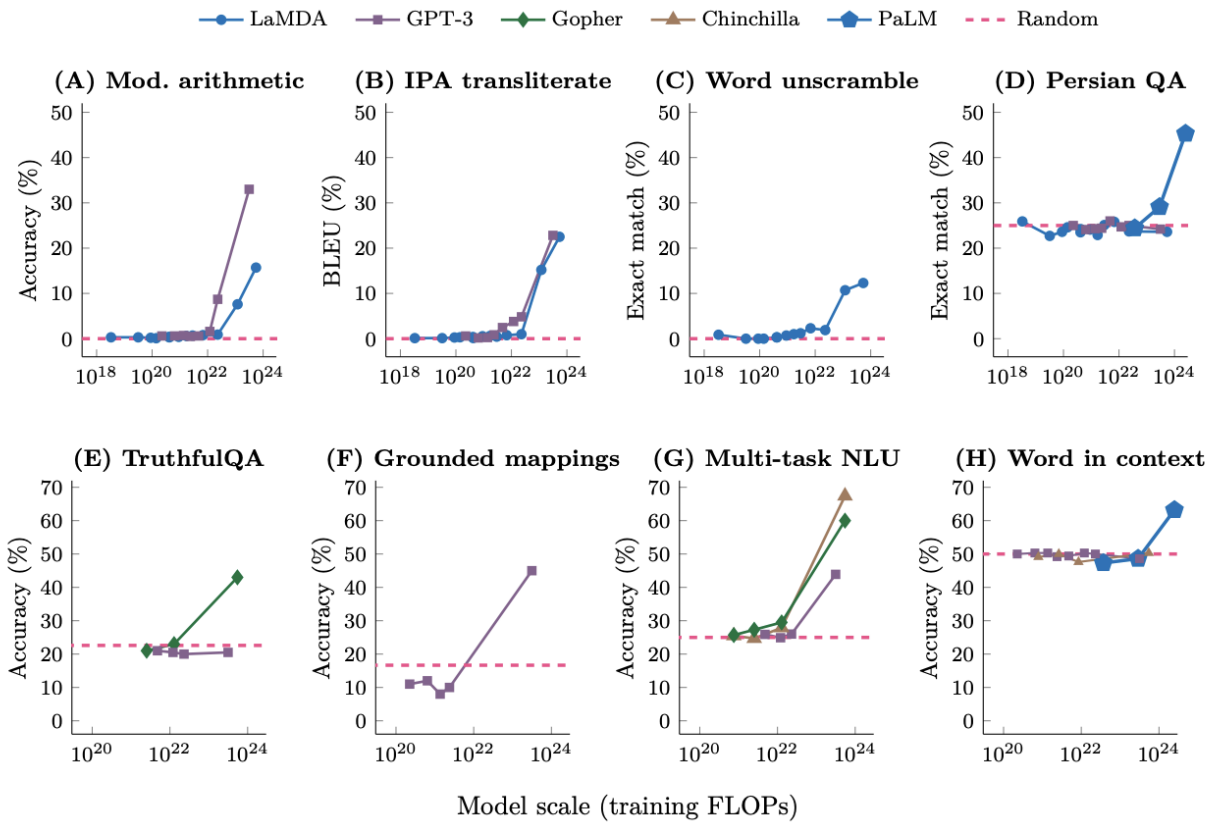
https://arxiv.org/pdf/2402.08797

Figure 4: Training compute used for notable ML models has been doubling every six months since the emergence of the Deep Learning Era. Executive Order 14110 introduced a notification requirement for models trained with more than 10^{26} operations (and 10^{23} operations if trained on using primarily biological sequence data).

LLaMA: Open and Efficient Foundation Language Models https://arxiv.org/abs/2302.13971

Emergent abilities

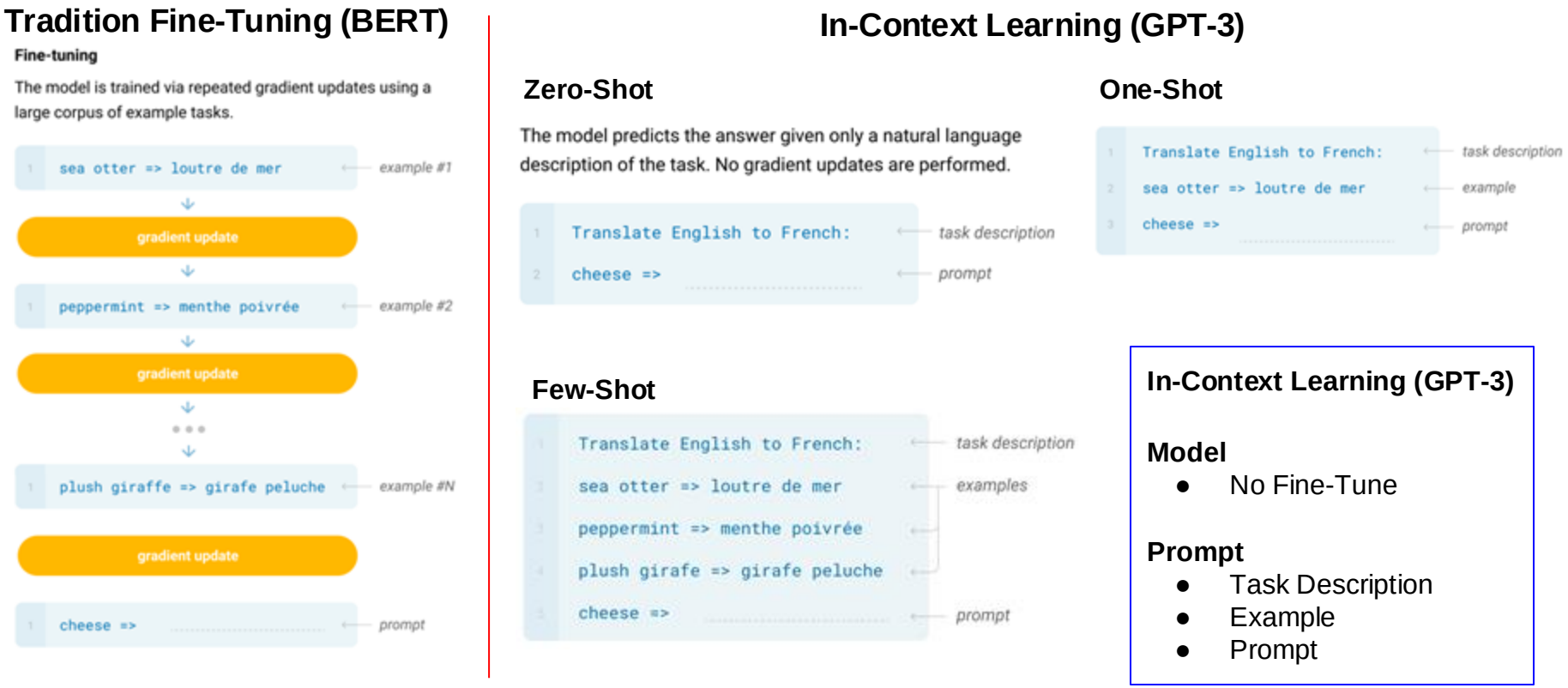
- People have found that models perform some capabilities much better after a certain size



Emergent Abilities of Large Language Models https://arxiv.org/abs/2206.07682

Why large model matters?

- In GPT3 paper, OpenAI found that large models are good few-shot learners



https://arxiv.org/abs/2005.14165

Why large model matters

Zero-Shot

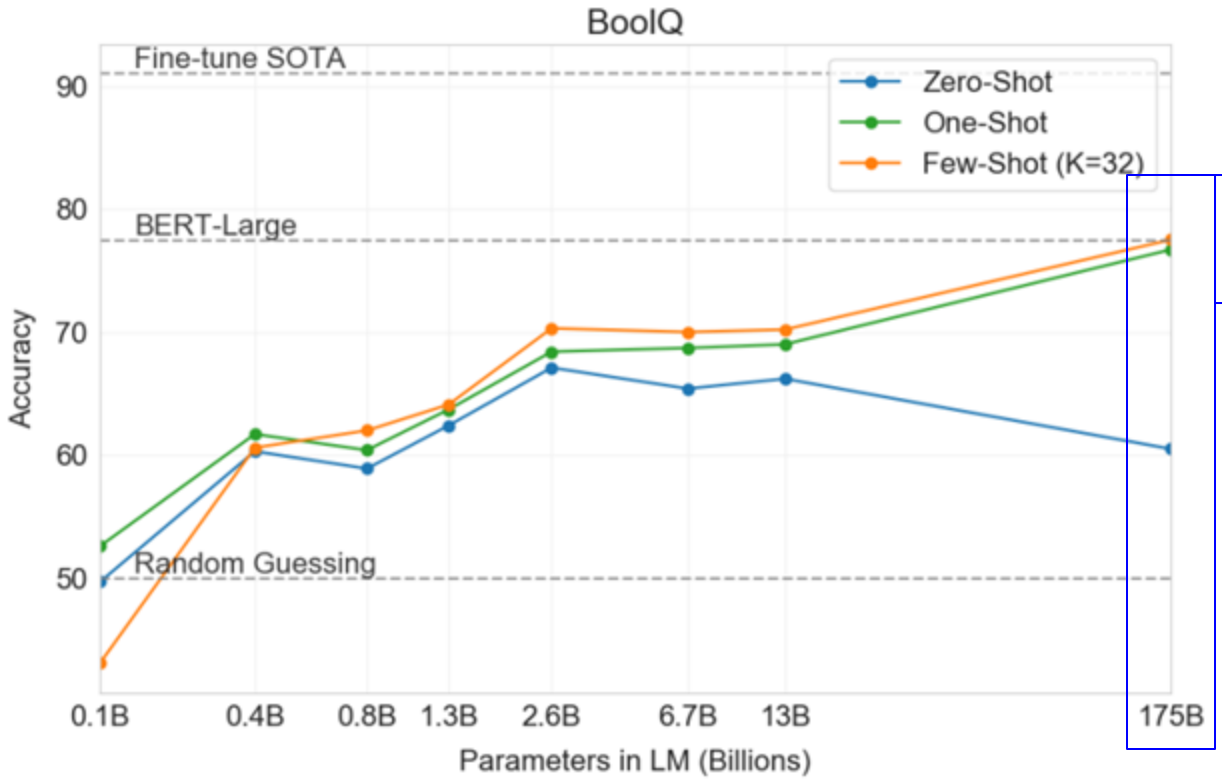
Dataset: BoolQ

Context →	Normal force -- In a simple case such as an object resting upon a table, the normal force on the object is equal but in opposite direction to the gravitational force applied on the object (or the weight of the object), that is, N = m g (\displaystyle N=mg), where m is mass, and g is the gravitational field strength (about 9.81 m/s on Earth). The normal force here represents the force applied by the table against the object that prevents it from sinking through the table and requires that the table is sturdy enough to deliver this normal force without breaking. However, it is easy to assume that the normal force and weight are action-reaction force pairs (a common mistake). In this case, the normal force and weight need to be equal in magnitude to explain why there is no upward acceleration of the object. For example, a ball that bounces upwards accelerates upwards because the normal force acting on the ball is larger in magnitude than the weight of the ball.
Prompt	question: is the normal force equal to the force of gravity? answer:
Target Completion →	yes

Text Input

https://arxiv.org/abs/2005.14165

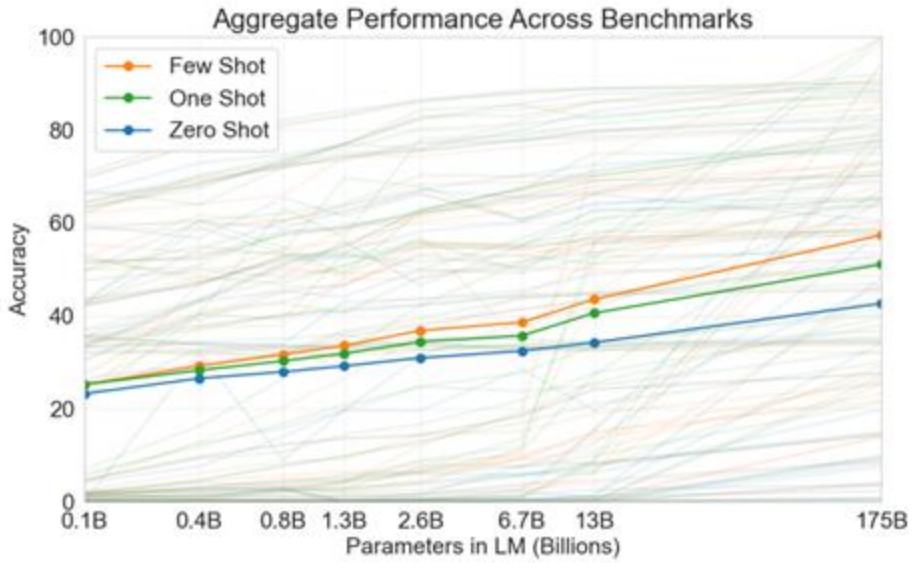
Why large model matters



On par with BERT-Large

https://arxiv.org/abs/2005.14165

Why large model matters



General Model:
On average, the accuracy in most tasks are not as good

Fine-Tune

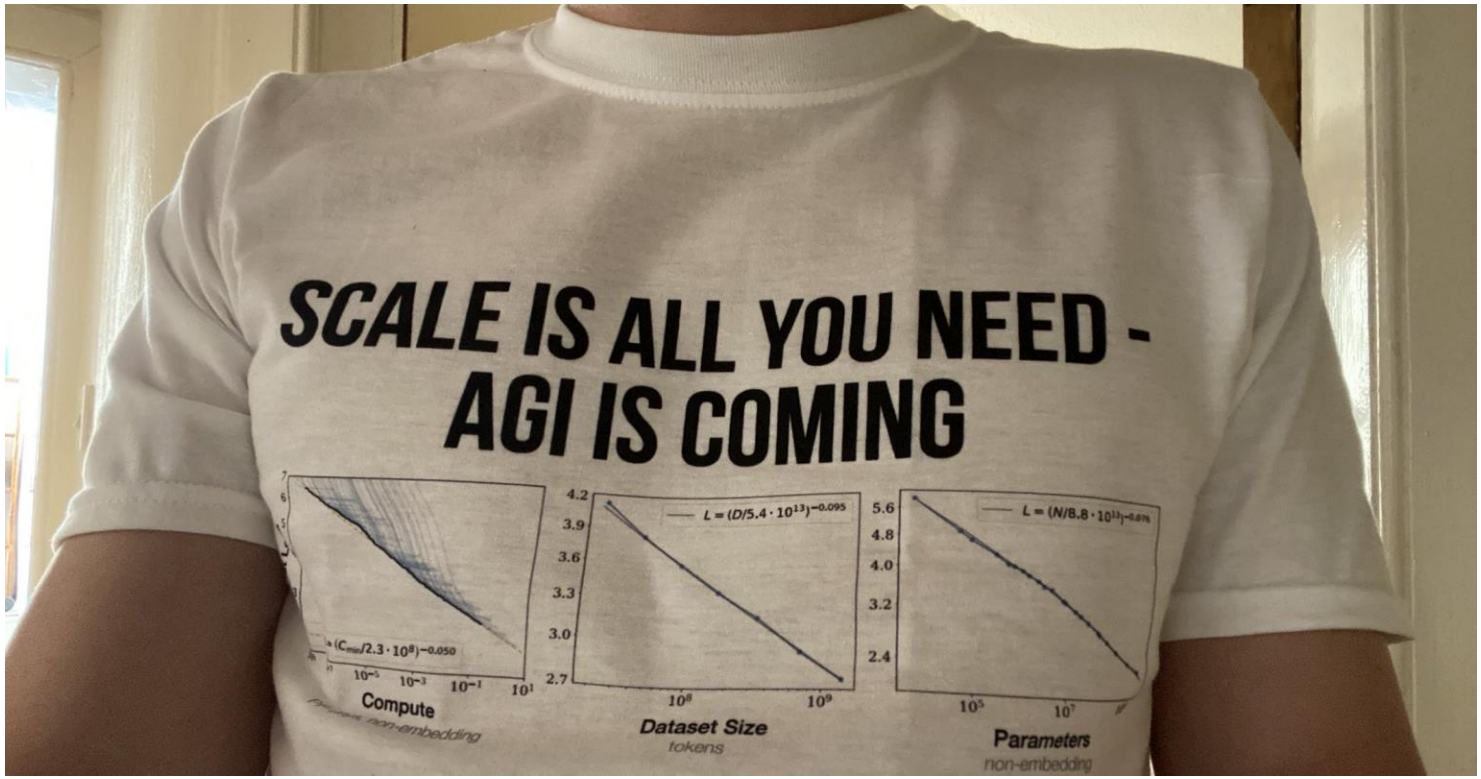
Specific Model:
Fine-tune model towards a specific task

Figure 1.3: Aggregate performance for all 42 accuracy-denominated benchmarks While zero-shot performance improves steadily with model size, few-shot performance increases more rapidly, demonstrating that larger models are more proficient at in-context learning. See Figure 3.8 for a more detailed analysis on SuperGLUE, a standard NLP benchmark suite.

https://arxiv.org/abs/2005.14165

Scaling is all you need

- This realization of scale leads to several efforts to try to understand the relationship between data, compute, and performance.



Kaplan Scaling law

- In 2020, OpenAI release the first paper discussing the "optimal model size"
  - An empirical study, think of Scaling law as Moore's law, just an observation, on the log scale
  - Performance depends mostly on scale rather than other factors such as width or depth of network
  - Need to scale data, compute, and network size in tandem

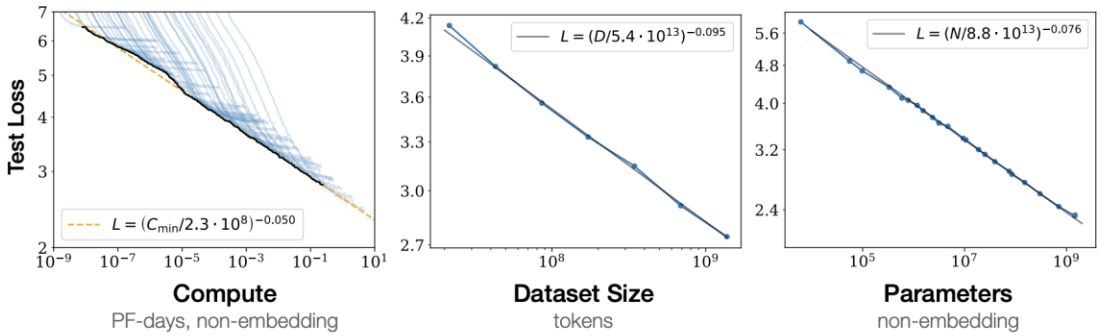


Figure 1 Language modeling performance improves smoothly as we increase the model size, datasetset size, and amount of compute[2] used for training. For optimal performance all three factors must be scaled up in tandem. Empirical performance has a power-law relationship with each individual factor when not bottlenecked by the other two.

Scaling Laws for Neural Language Models https://arxiv.org/pdf/2001.08361

Kaplan Scaling law

- Large model are more sample-efficient.
  - Less gradient updates to reach the same performance.
- Models converge slowly
  - Stop when you just want to stop. Late iterations give less gains.

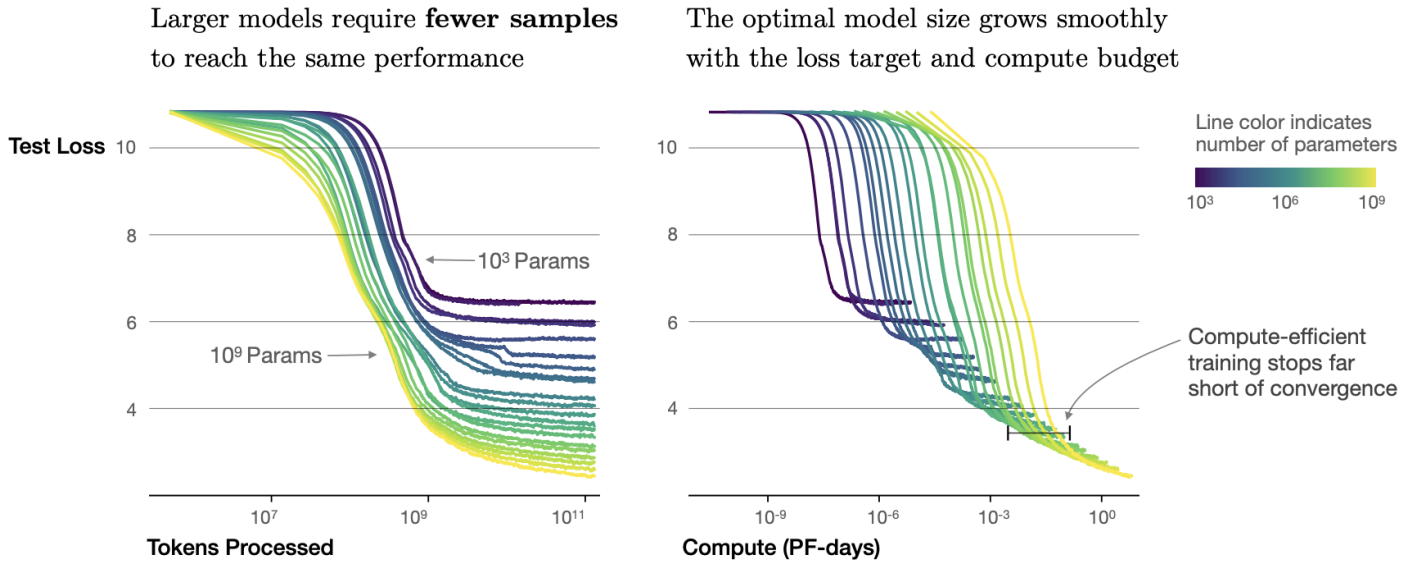


Figure 2 We show a series of language model training runs, with models ranging in size from 10^3 to 10^9 parameters (excluding embeddings).

Kaplan Scaling law

- They curved fit to the data and got some dubious equations
  - In simple implications, for GPT-3, you need 1.7 text tokens per parameter in you LLM to be efficient
  - Often refer to as Kaplan scaling law (the first author)

The test loss of a Transformer trained to autoregressively model language can be predicted using a power-law when performance is limited by only either the number of non-embedding parameters N, the dataset size D, or the optimally allocated compute budget $C_{\min}$ (see Figure 1):

1. For models with a limited number of parameters, trained to convergence on sufficiently large datasets: Relationship with infinite size data

$$L(N) = (N_c/N)^{\alpha_N}\ ;\ \alpha_N \sim 0.076,\quad N_c \sim 8.8 \times 10^{13} \text{ (non-embedding parameters)} \qquad (1.1)$$

2. For large models trained with a limited dataset with early stopping: Relationship with infinite size model

$$L(D) = (D_c/D)^{\alpha_D}\ ;\ \alpha_D \sim 0.095,\quad D_c \sim 5.4 \times 10^{13} \text{ (tokens)} \qquad (1.2)$$

3. When training with a limited amount of compute, a sufficiently large dataset, an optimally-sized model, and a sufficiently small batch size (making optimal[3] use of compute):

$$L(C_{\min}) = \left(C_c^{\min}/C_{\min}\right)^{\alpha_C^{\min}}\ ;\ \alpha_C^{\min} \sim 0.050,\quad C_c^{\min} \sim 3.1 \times 10^8 \text{ (PF-days)} \qquad (1.3)$$

Chinchilla Scaling law

- In 2022, Deepmind published another scaling law paper
  - Found that model size and training data should be scaled equally. Around 20 tokens per parameter.
  - Used smaller model, but trained with more data to get better performance

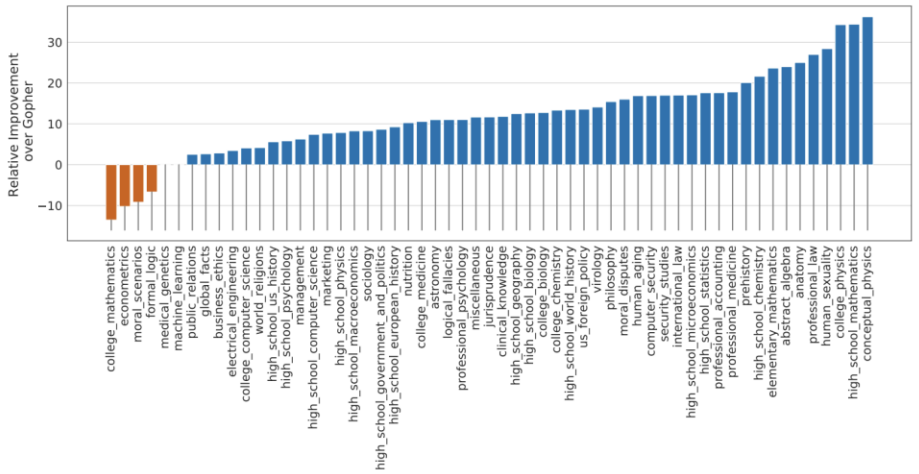


Table 1 | **Current LLMs**. We show five of the current largest dense transformer models, their size, and the number of training tokens. Other than LaMDA (Thoppilan et al., 2022), most models are trained for approximately 300 billion tokens. We introduce *Chinchilla*, a substantially smaller model, trained for much longer than 300B tokens.

Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

https://arxiv.org/abs/2203.15556

Chinchilla Scaling Law

- They offer a method to think about the data size D and model size N that will gives compute optimal (given a fixed compute budget) as

$$\hat{L}(N, D) \triangleq E + \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}}.$$

Inherent loss (noise)

What happens with a finite model

What happens with a finite data

With experiments from OpenAI, A = 406.4,
B = 410.7, E = 1.69
Alpha = 0.34, Beta = 0.25

If we plug in gopher numbers, L = 1.993
Chinchilla L = 1.936

Table 1 | **Current LLMs**. We show five of the current largest dense transformer models, their size, and the number of training tokens. Other than LaMDA (Thoppilan et al., 2022), most models are trained for approximately 300 billion tokens. We introduce *Chinchilla*, a substantially smaller model, trained for much longer than 300B tokens.

Model	Size (# Parameters)	Training Tokens
LaMDA (Thoppilan et al., 2022)	137 Billion	168 Billion
GPT-3 (Brown et al., 2020)	175 Billion	300 Billion
Jurassic (Lieber et al., 2021)	178 Billion	300 Billion
Gopher (Rae et al., 2021)	280 Billion	300 Billion
MT-NLG 530B (Smith et al., 2022)	530 Billion	270 Billion
Chinchilla	70 Billion	1.4 Trillion

Other Scaling Laws

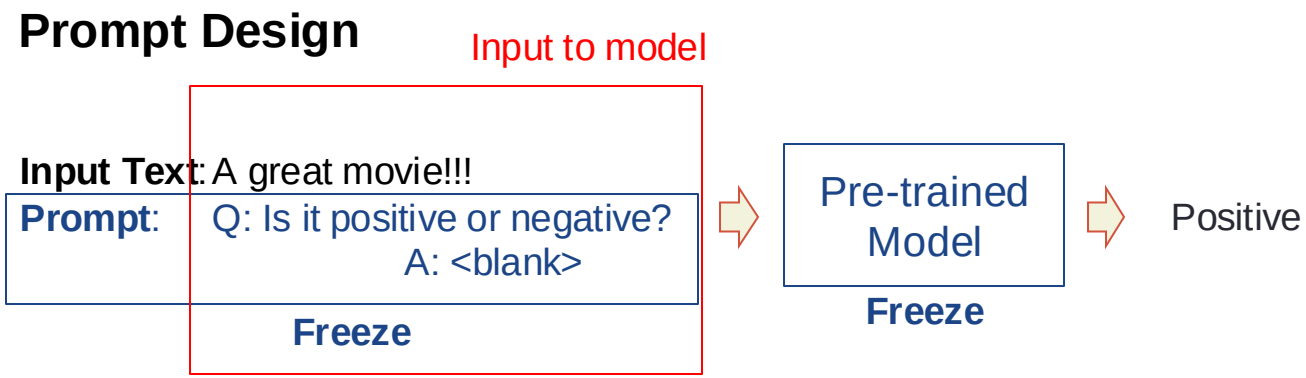
- Broken Neural Scaling laws 2023 https://arxiv.org/abs/2210.14891
  - A work offering scaling laws that cover various tasks (besides NLP)
- The laws so far are for training time, we can also investigate inference time scaling (test-time scaling)

Outline

- Generative AI
- Emergent properties & scaling laws
- Alignment
  - Instruction tuning
  - Preference learning

Alignment

Back to prompting



Verbalization and prompt design

Different Template

Template	Label words	Accuracy
SNLI (entailment/neutral/contradiction)		mean (std)
$<S_1>$? [MASK] , $<S_2>$	Yes/Maybe/No	**77.2 (3.7)**
$<S_1>$. [MASK] , $<S_2>$	Yes/Maybe/No	76.2 (3.3)
$<S_1>$? [MASK] $<S_2>$	Yes/Maybe/No	74.9 (3.0)
$<S_1>$ $<S_2>$ [MASK]	Yes/Maybe/No	65.8 (2.4)
$<S_2>$? [MASK] , $<S_1>$	Yes/Maybe/No	62.9 (4.1)
$<S_1>$? [MASK] , $<S_2>$	Maybe/No/Yes	60.6 (4.8)
Fine-tuning	-	48.4 (4.8)

Different Verbalizer

Template	Label words	Accuracy
SST-2 (positive/negative)		mean (std)
$<S_1>$ It was [MASK] .	great/terrible	**92.7 (0.9)**
$<S_1>$ It was [MASK] .	good/bad	92.5 (1.0)
$<S_1>$ It was [MASK] .	cat/dog	91.5 (1.4)
$<S_1>$ It was [MASK] .	dog/cat	86.2 (5.4)
$<S_1>$ It was [MASK] .	terrible/great	83.2 (6.9)
Fine-tuning	-	81.4 (3.8)

premise (string)	hypothesis (string)	label (class label)
"This church choir sings to the masses as they sing joyous songs…	"The church has cracks in the ceiling."	1 (neutral)
"This church choir sings to the masses as they sing joyous songs…	"The church is filled with song."	0 (entailment)
"This church choir sings to the masses as they sing joyous songs…	"A choir singing at a baseball game."	2 (contradiction)

https://arxiv.org/pdf/2104.08691.pdf

Why do we even need Prompting?

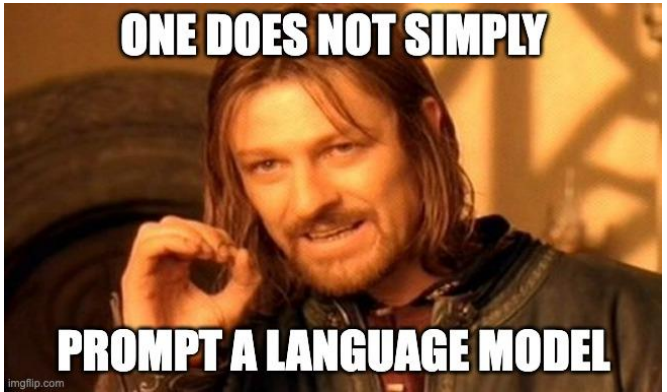
Self-supervised models are stochastic parots

Input (Prompt)	Output
The patient was died. <fill next…>	The patient's body was found in a dark alley behind the hospital's…
"The patient was died." correct this <fill next…>	claim if you really believe such figures….
Poor English input: The patient was died. <fill next…>	Good English output: The patient died.

Alignment

https://www.alignmentforum.org/
Web forum discussing the alignment problem

- It's hard to make a self-superivsed model do what we want. This is called the alignment problem (aligns model capabilities with users' interests). Ultimately, you will need supervision for this!
- Current solutions
  - Prompt engineer (manual)
  - Low parameter finetuning – LoRa, mixture of experts, learned prompts, etc.
  - Preference learning



InstructGPT

- In Mar 2022 (ChatGPT came out Dec 2022), OpenAI released a paper InstructGPT

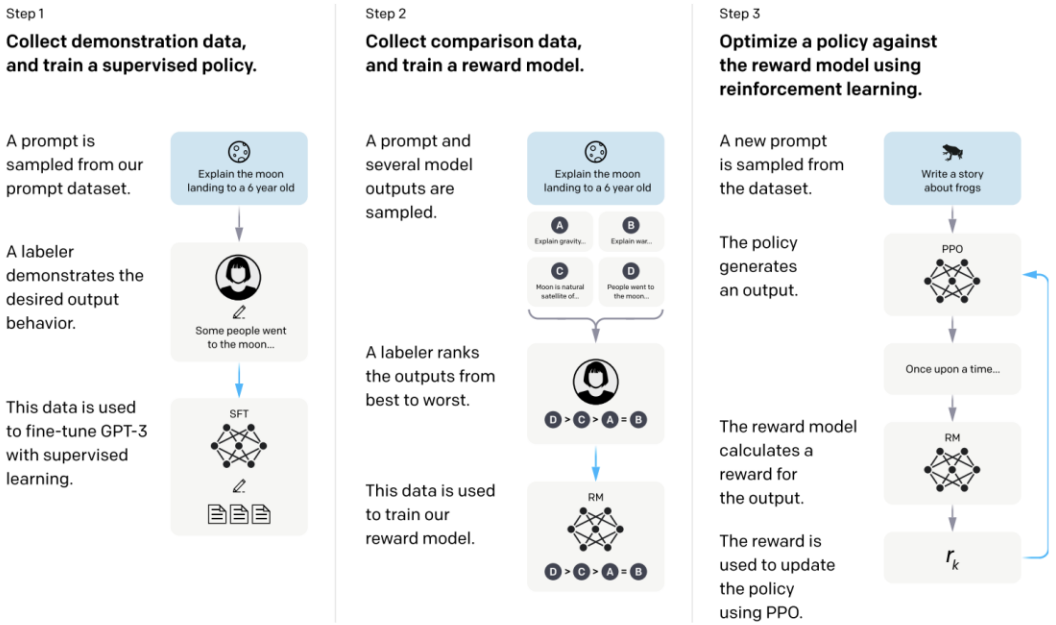


Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

https://arxiv.org/pdf/2203.02155

InstructGPT

- Step 0: pretrain a language model eg GPT3
- Step 1: Supervised fine-tuning (SFT)
- Step 2: Reward Model training
- Step 3: Reinforcement Learning with Human feedback (RLHF)

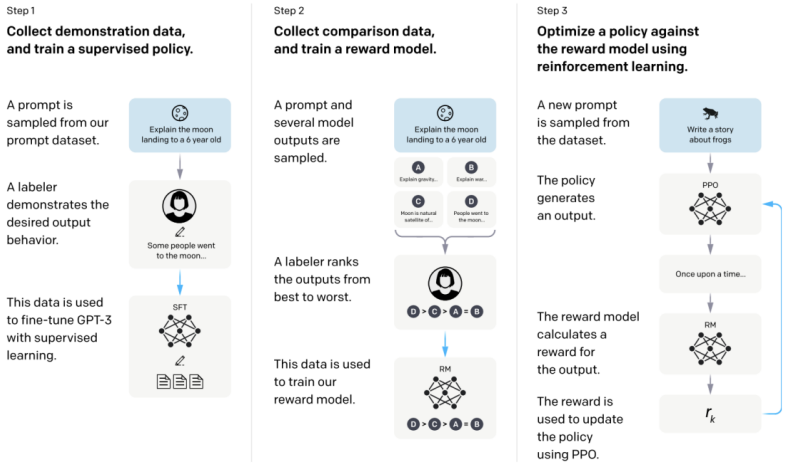
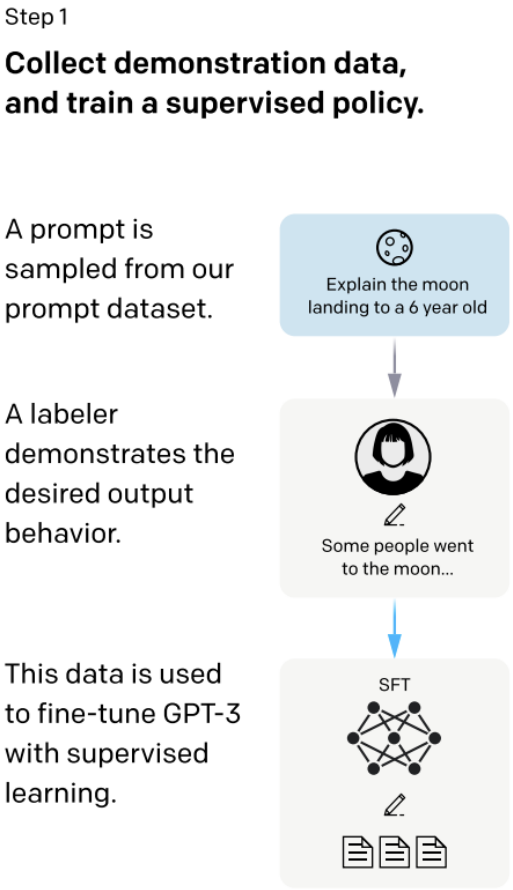


Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

https://arxiv.org/pdf/2203.02155

Supervised Finetuning/Instruction Tuning

- Create a dataset of questions and answers.
  - Often called “Instruction data”



Instruction data

- Just example Q A pairs
- Alpaca is one of the earliest open instruction data. They just use ChatGPT to generate the answers.
  - A kind of distillation
  - Easier and faster than hiring a bunch of humans. People can copy you! OpenAI terms has a non-compete clause.
- Deepseek uses 1.5 million

prompt string · *lengths* 29↔309 97.9%	chosen string · *lengths* 1↔454 48.9%
Human: Construct a SQL query to print the customer information…	``` SELECT * FROM Customers ORDER BY customer_name ASC; ```
Human: Assign each word in the sentence below to its part of speech.Kittens often scamper around excitedly. Assistant:	Kittens - noun often - adverb scamper - verb around - preposition excitedly - adverb.
Human: Rewrite the sentence to make the subject the object.The workers…	Better pay was protested for by the workers.
Human: Generate a web-safe color combination Assistant:	One web-safe color combination could be: - Background color: #FFFFFF (white) - Primary color…
Human: Create a regular expression that matches strings starting with…	Sure, here is a regular expression that matches strings starting with "Bob" and ending with a…

https://huggingface.co/datasets/iamketan25/alpaca-instructions-dataset

https://github.com/tatsu-lab/stanford_alpaca

Self-instruct

- A technique to use LLM to generate data to train more LLMs!

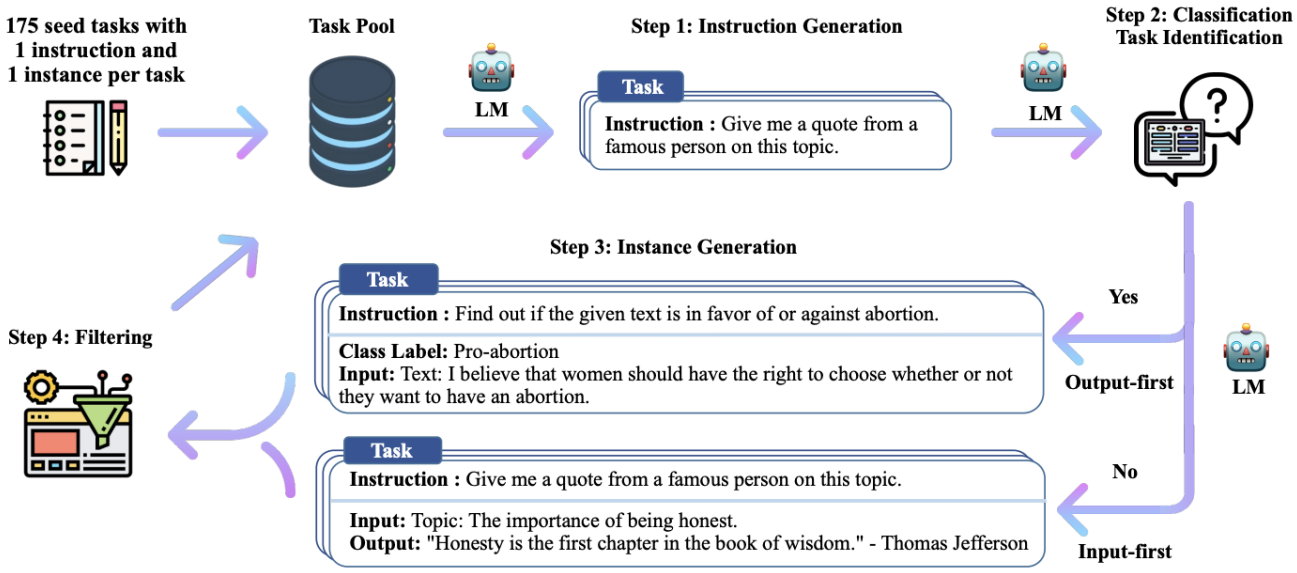
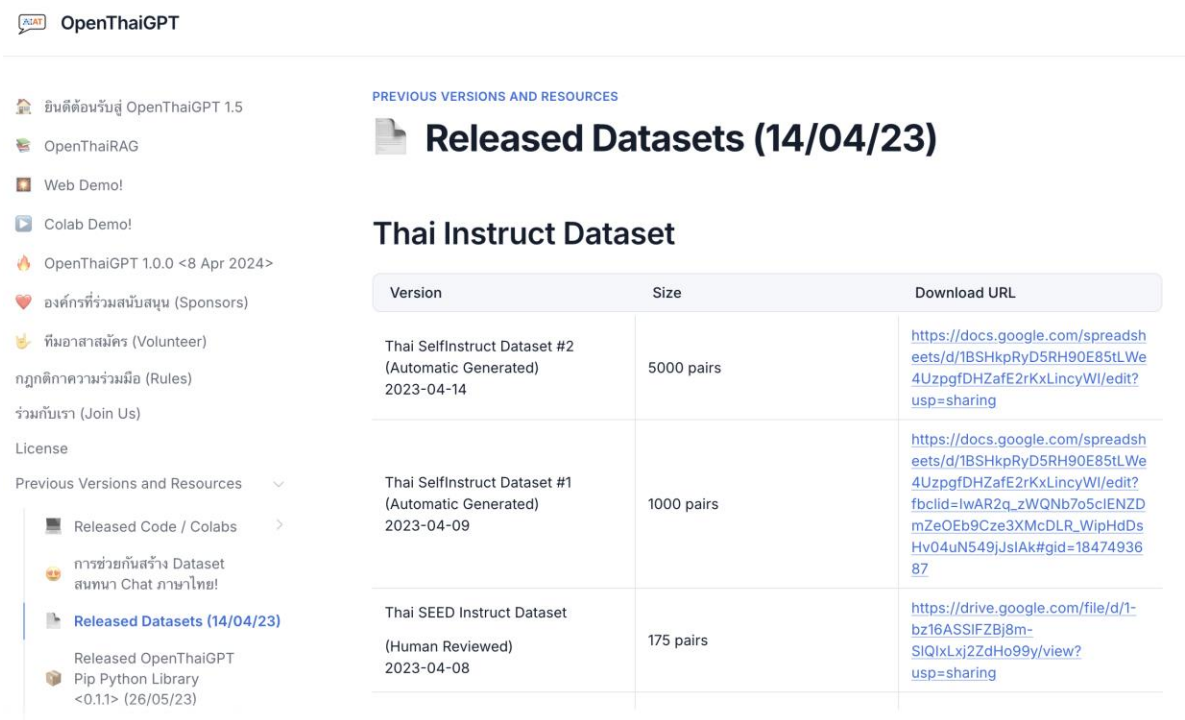


Figure 2: A high-level overview of SELF-INSTRUCT. The process starts with a small seed set of tasks as the task pool. Random tasks are sampled from the task pool, and used to prompt an off-the-shelf LM to generate both new instructions and corresponding instances, followed by filtering low-quality or similar generations, and then added back to the initial repository of tasks. The resulting data can be used for the instruction tuning of the language model itself later to follow instructions better. Tasks shown in the figure are generated by GPT3.

https://arxiv.org/pdf/2212.10560

Thai instruction data

- Still super small, most are automatically generated
- OpenThaiGPT

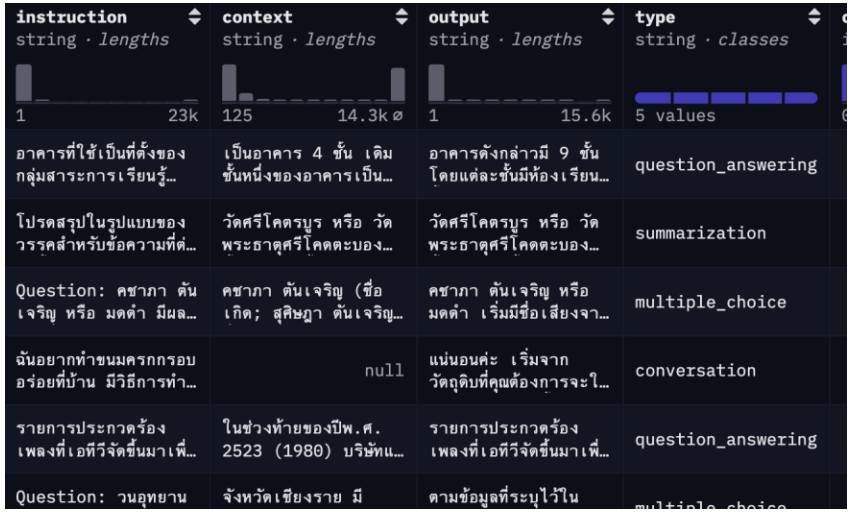


https://openthaigpt.aieat.or.th/previous-versions-and-resources/released-datasets-14-04-23

Thai instruction data

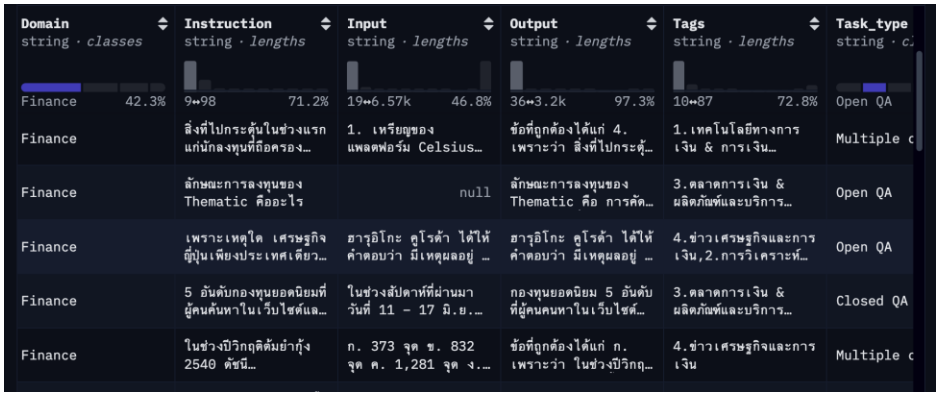
- WangchangX

https://huggingface.co/datasets/airesearch/wangchanx-seed-free-synthetic-instruct-thai-120k



instruction string · lengths	context string · lengths	output string · lengths	type string · classes
1 – 23k	125 – 14.3k ø	1 – 15.6k	5 values
อาคารที่ใช้เป็นที่ตั้งของกลุ่มสาระการเรียนรู้…	เป็นอาคาร 4 ชั้น เดิมชั้นหนึ่งของอาคารเป็น…	อาคารดังกล่าวมี 9 ชั้น โดยแต่ละชั้นมีห้องเรียน…	question_answering
โปรดสรุปในรูปแบบของวรรคสำหรับข้อความที่ต่…	วัดศรีโคตรบูร หรือ วัดพระธาตุศรีโคตตะบอง…	วัดศรีโคตรบูร หรือ วัดพระธาตุศรีโคตตะบอง…	summarization
Question: คชาภา ตันเจริญ หรือ มตดำ มีผล…	คชาภา ตันเจริญ (ชื่อเกิด; สุศิษฎา ตันเจริญ…	คชาภา ตันเจริญ หรือ มตดำ เริ่มมีชื่อเสียงจา…	multiple_choice
ฉันอยากทำขนมครกกรอบอร่อยที่บ้าน มีวิธีการทำ…	null	แน่นอนค่ะ เริ่มจากวัตถุดิบที่คุณต้องการจะใ…	conversation
รายการประกวดร้องเพลงที่เอทีวีจัดขึ้นมาเพื่…	ในช่วงท้ายของปีพ.ศ. 2523 (1980) บริษัทแ…	รายการประกวดร้องเพลงที่เอทีวีจัดขึ้นมาเพื่…	question_answering
Question: วนอุทยาน	จังหวัดเชียงราย มี	ตามข้อมูลที่ระบุไว้ใน	multiple_choice

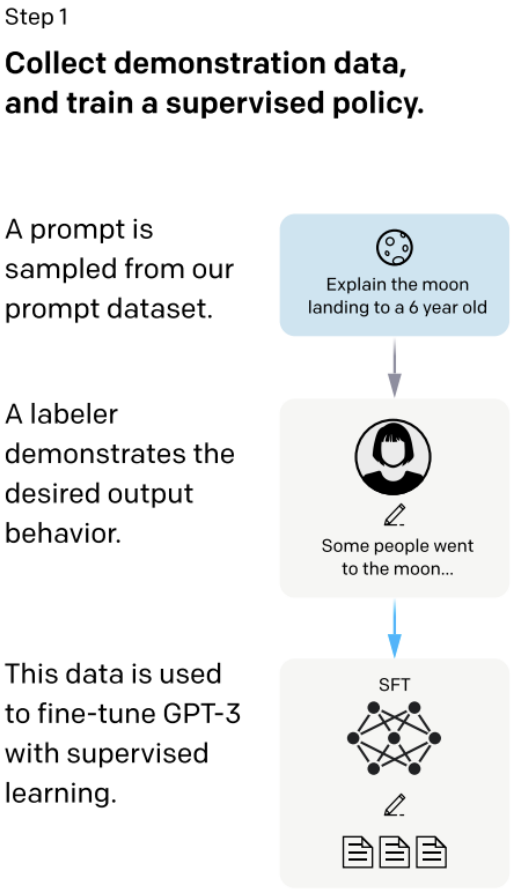
https://huggingface.co/datasets/airesearch/WangchanThaiInstruct_7.24



Domain string · classes	Instruction string · lengths	Input string · lengths	Output string · lengths	Tags string · lengths	Task_type string · cl
Finance 42.3%	9 – 98 71.2%	19 – 6.57k 46.8%	36 – 3.2k 97.3%	10 – 87 72.8%	Open QA
Finance	สิ่งที่ไปกระตุ้นในช่วงแรกแก่นักลงทุนที่ถือครอง…	1. เหรียญของแพลตฟอร์ม Celsius…	ข้อที่ถูกต้องได้แก่ 4. เพราะว่า สิ่งที่ไปกระตุ้…	1.เทคโนโลยีทางการเงิน & การเงิน…	Multiple c
Finance	ลักษณะการลงทุนของ Thematic คืออะไร	null	ลักษณะการลงทุนของ Thematic คือ การคัด…	3.ตลาดการเงิน & ผลิตภัณฑ์และบริการ…	Open QA
Finance	เพราะเหตุใด เศรษฐกิจญี่ปุ่นเพียงประเทศเดียว…	ฮารุอิโกะ คูโรด้า ได้ให้คำตอบว่า มีเหตุผลอยู่ …	ฮารุอิโกะ คูโรด้า ได้ให้คำตอบว่า มีเหตุผลอยู่ …	4.ข่าวเศรษฐกิจและการเงิน,2.การวิเคราะห์…	Open QA
Finance	5 อันดับกองทุนยอดนิยมที่ผู้คนค้นหาในเว็บไซต์แล…	ในช่วงสัปดาห์ที่ผ่านมา วันที่ 11 – 17 มิ.ย.…	กองทุนยอดนิยม 5 อันดับที่ผู้คนค้นหาในเว็บไซต์…	3.ตลาดการเงิน & ผลิตภัณฑ์และบริการ…	Closed QA
Finance	ในช่วงปีวิกฤติต้มยำกุ้ง 2540 ดัชนี…	ก. 373 จุด ข. 832 จุด ค. 1,281 จุด ง.…	ข้อที่ถูกต้องได้แก่ ก. เพราะว่า ในช่วงปีวิกฤ…	4.ข่าวเศรษฐกิจและการเงิน	Multiple c

Supervised Finetuning/Instruction Tuning

- Create a dataset of questions and answers.
  - Often called “Instruction data”
- Trained using next word prediction on instruction data.



InstructGPT

- Step 0: pretrain a language model eg GPT3
- Step 1: Supervised fine-tuning (SFT)
- Step 2: Reward Model training
- Step 3: Reinforcement Learning with Human feedback (RLHF)

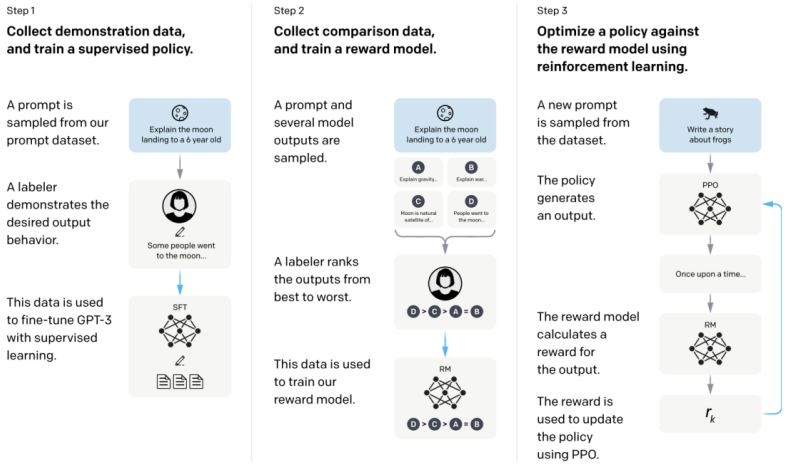


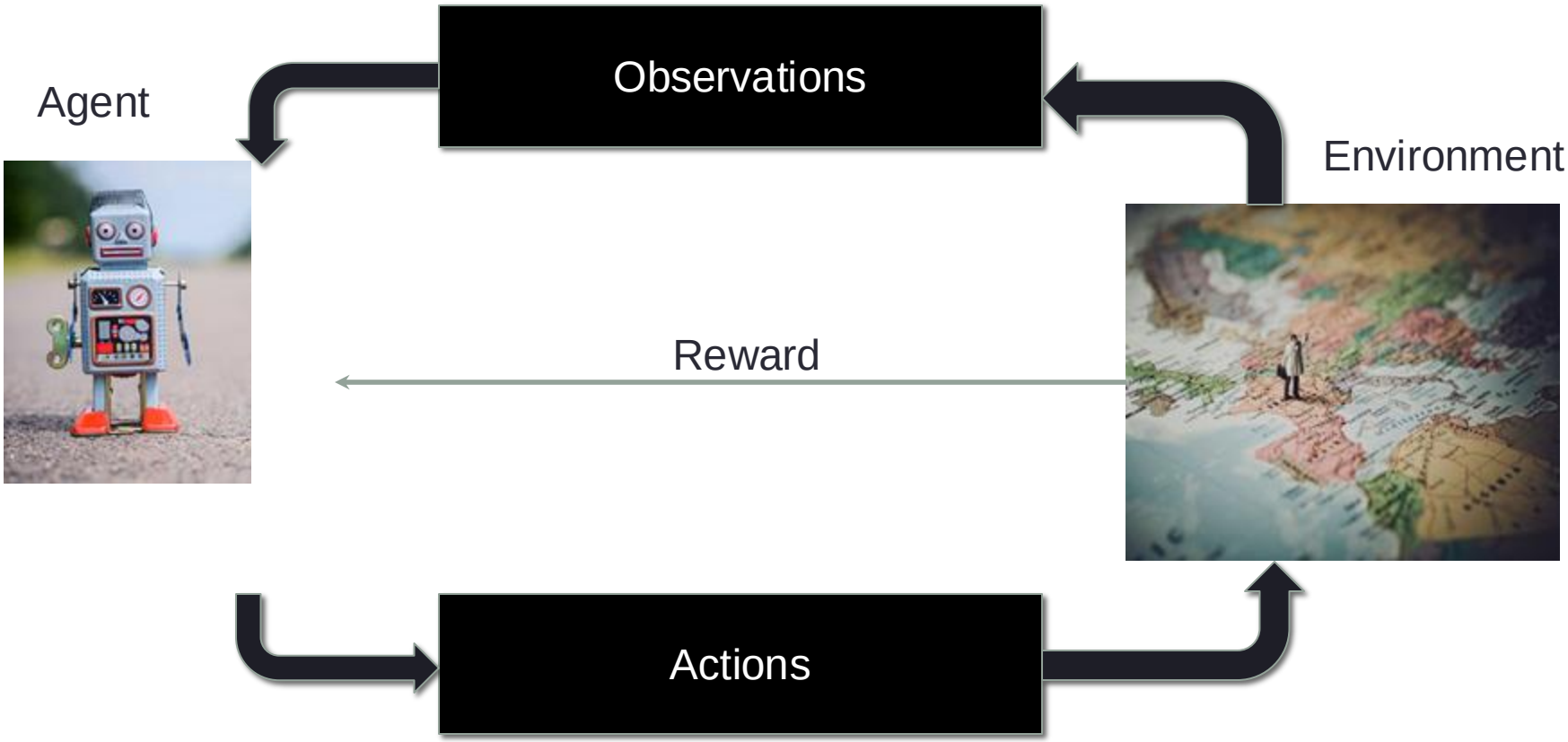
Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

https://arxiv.org/pdf/2203.02155

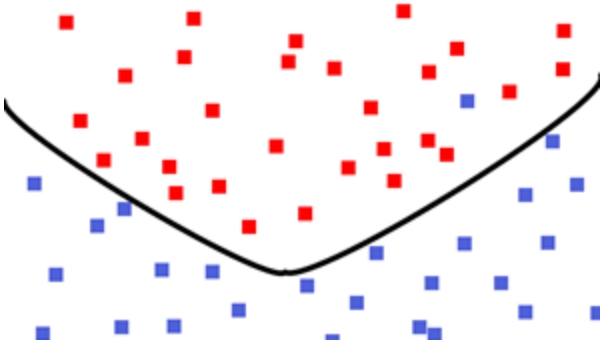

dont worry about it if you don't understand

Introduction to RL

RL problem



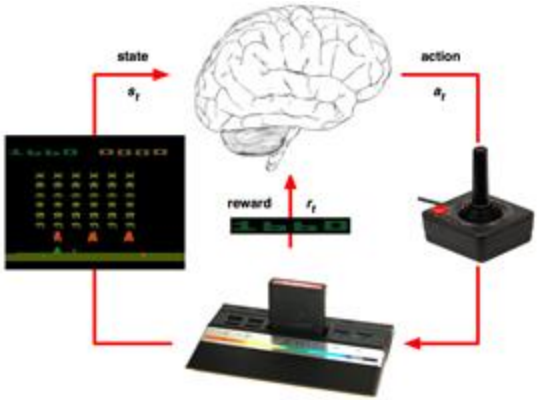
3 Modes of Learning



Supervised Learning

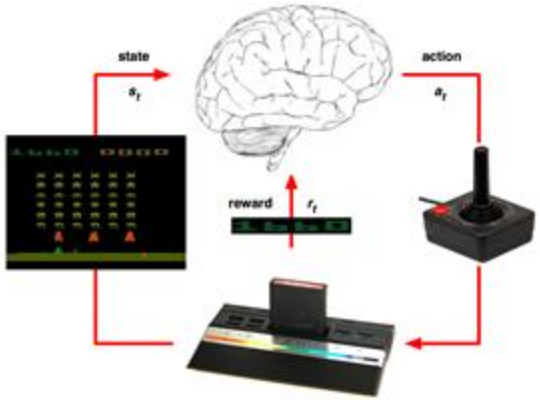
Reinforcement Learning

Unsupervised Learning



3 Modes of Learning

Reinforcement Learning (RL)



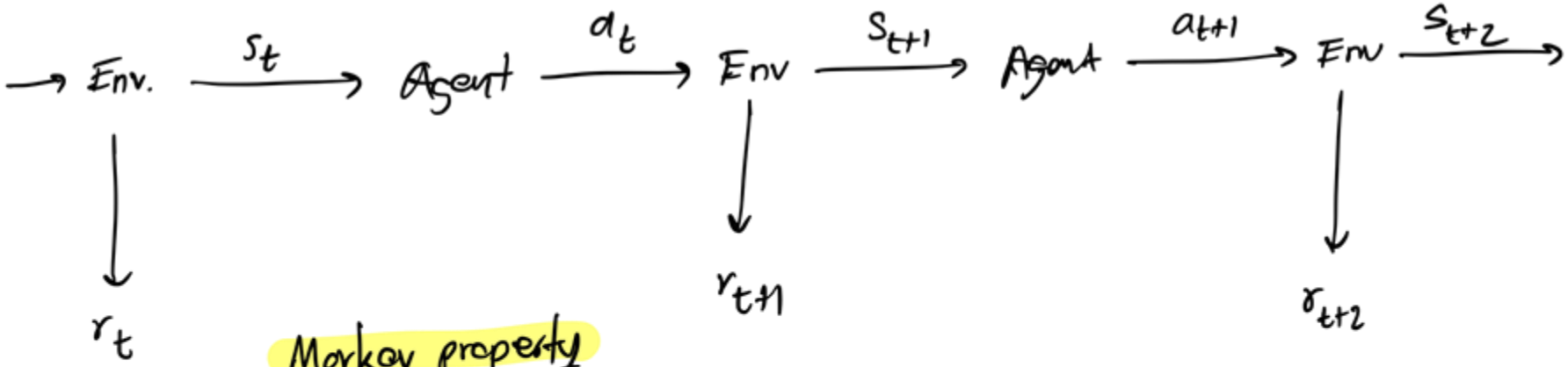
- Observe:
  - The states (x_1, x_2, x_3, …)
  - The reward (r_1, r_2, r_3, …)
- Can also take actions
  - a_1 , a_2 , a_3 , …
- What are the best actions?
  - Such that we will receive highest accumulative rewards

RL framework

Reward (r_t)

State (s_t)

Action (a_t)



Markov property

$a_t = A(s_t)$

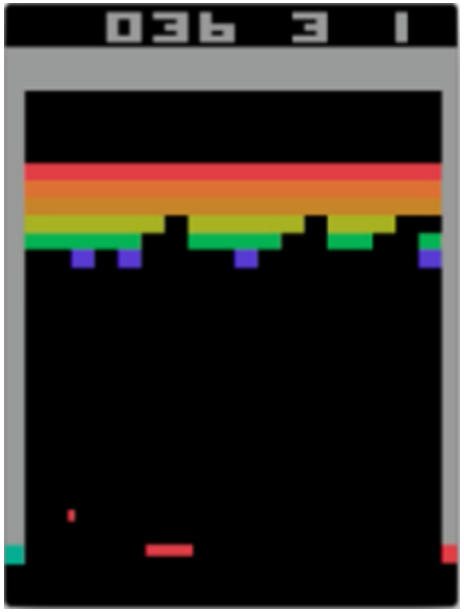
$r_{t+1} = R(s_t, a_t)$

$s_{t+1} = S(s_t, a_t)$

Rewards-based learning

- Maximise the rewards
- Can we design any desired behaviour with reward?

R_t = Δdistance



R_t = score

$$R_T = \begin{cases} 1 & \text{, win} \\ -1 & \text{, lose} \end{cases}$$

Policy

- Policy = a mapping from a state to an action
- Objective of RL is to find the ‘’*optimal*’’ policy!

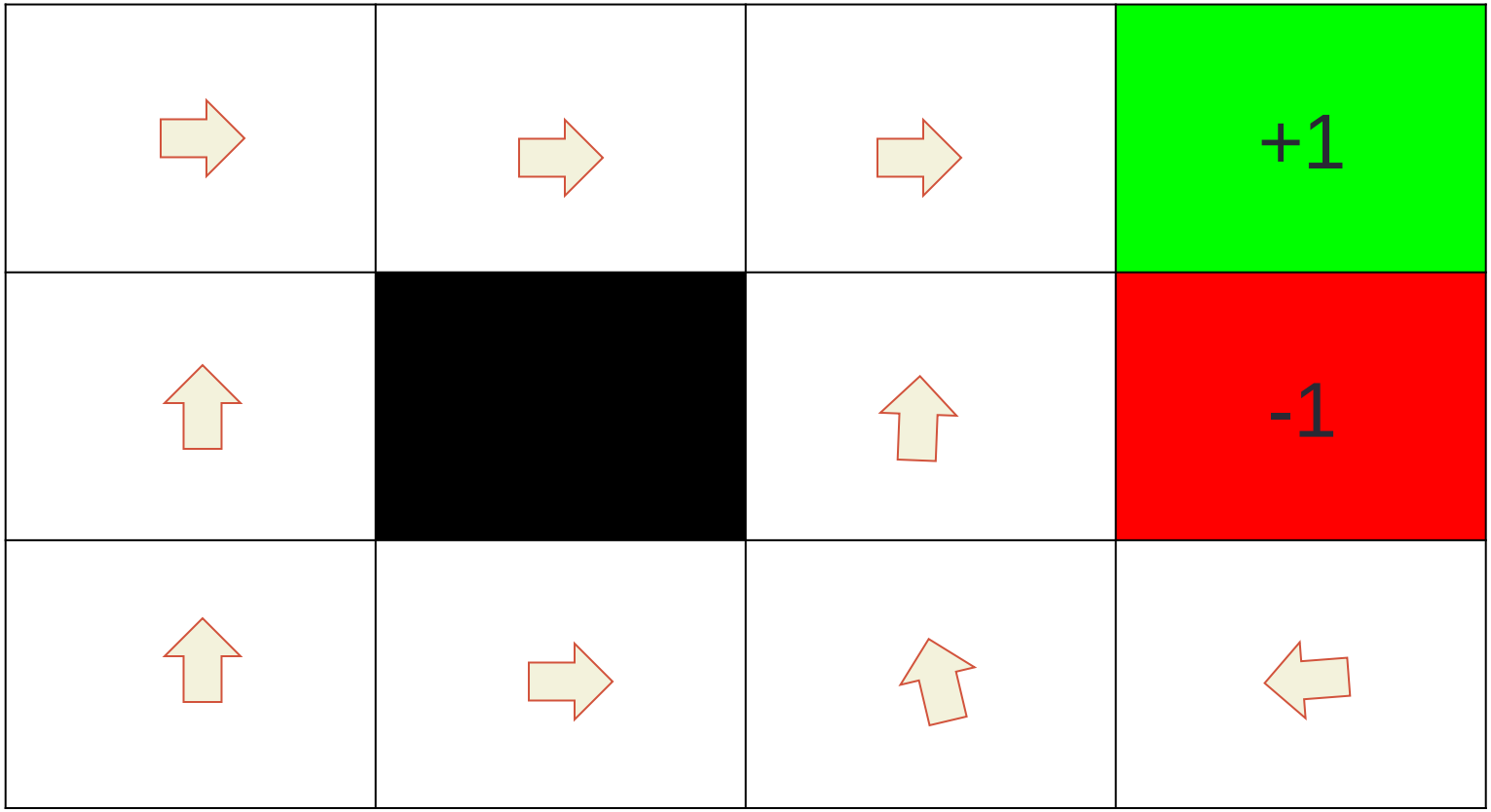
$$a = \pi(s)$$

$$a \sim \pi(s)$$

Can be either deterministic or stochastic

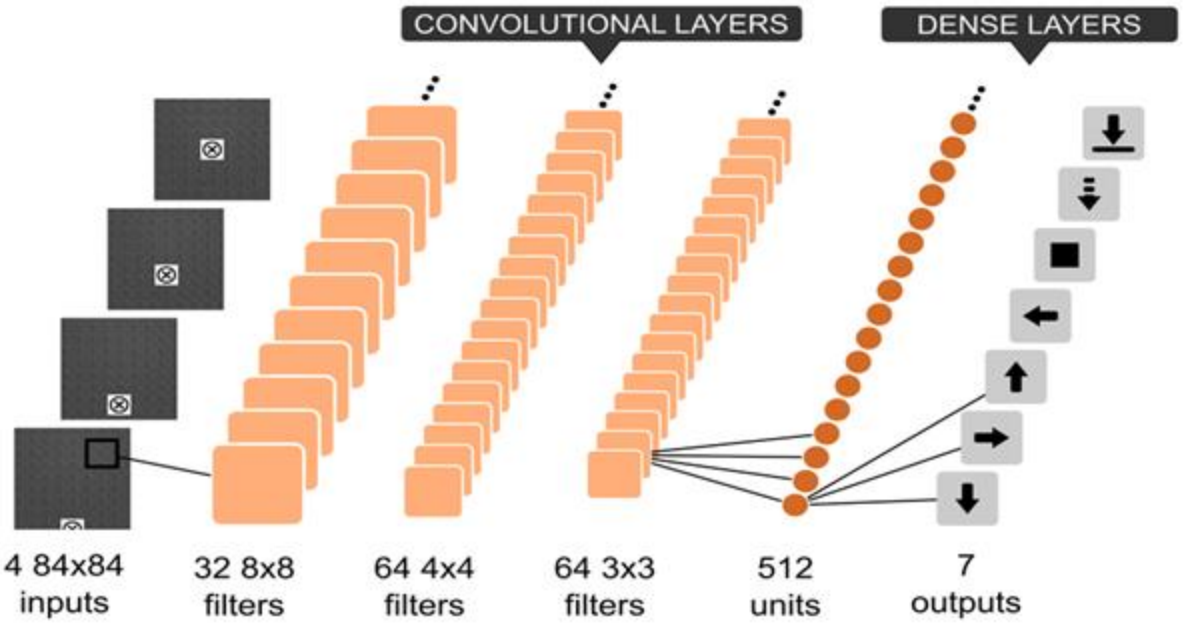
Policy

Example: tabular policy



Policy

- Example: policy to play an arcade game



Return (Cumulative rewards)

- Return = cumulative rewards with discount

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots = \sum_{t'=t}^{\infty} \gamma^{t'-t} r_t$$

$r_t = 0$

$r_{t+10} = 1$

$r_{t+34} = -1$

What is learning?

- Use data to find/search for the best policy

What is the best policy?

- Policy that give us the highest expected return!

$$G = r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$$

$$J(\pi) = E_\pi[G]$$

Two main ways to find the best policy

- Q-learning (Value-based)
- Policy gradient (Policy-based)

Q-learning algorithm

- Let's define a state value as
  - Expectation of the return after visit s and follow **π**

$$V^{\pi}(s) = E_{\pi}[G_t | s_t = s]$$

- Let's define a state-action value (Q-value) as
  - Expectation of the return after visit a state s, take action a

$$Q^{\pi}(s, a) = E_{\pi}[G_t | s_t = s, a_t = a]$$

$$V^{\pi}(s) = E_{\pi(s)}[Q^{\pi}(s, \pi(s))]$$

Q-learning algorithm

- There exist an optimal value function associate with an optimal policy,

$$V^*(s) = \max_\pi V^\pi(s) \quad \forall s \in S$$

- The optimal policy is the policy that achieves the highest value for every state

Q-learning algorithm

- It follows that

$$V^*(s) = \max_a [Q^*(s,a)]$$

- and ..

$$\pi^*(s) = argmax_a Q^*(s,a)$$

- Optimal actions can be found indirectly through Q-value

Policy gradient

Q Learning

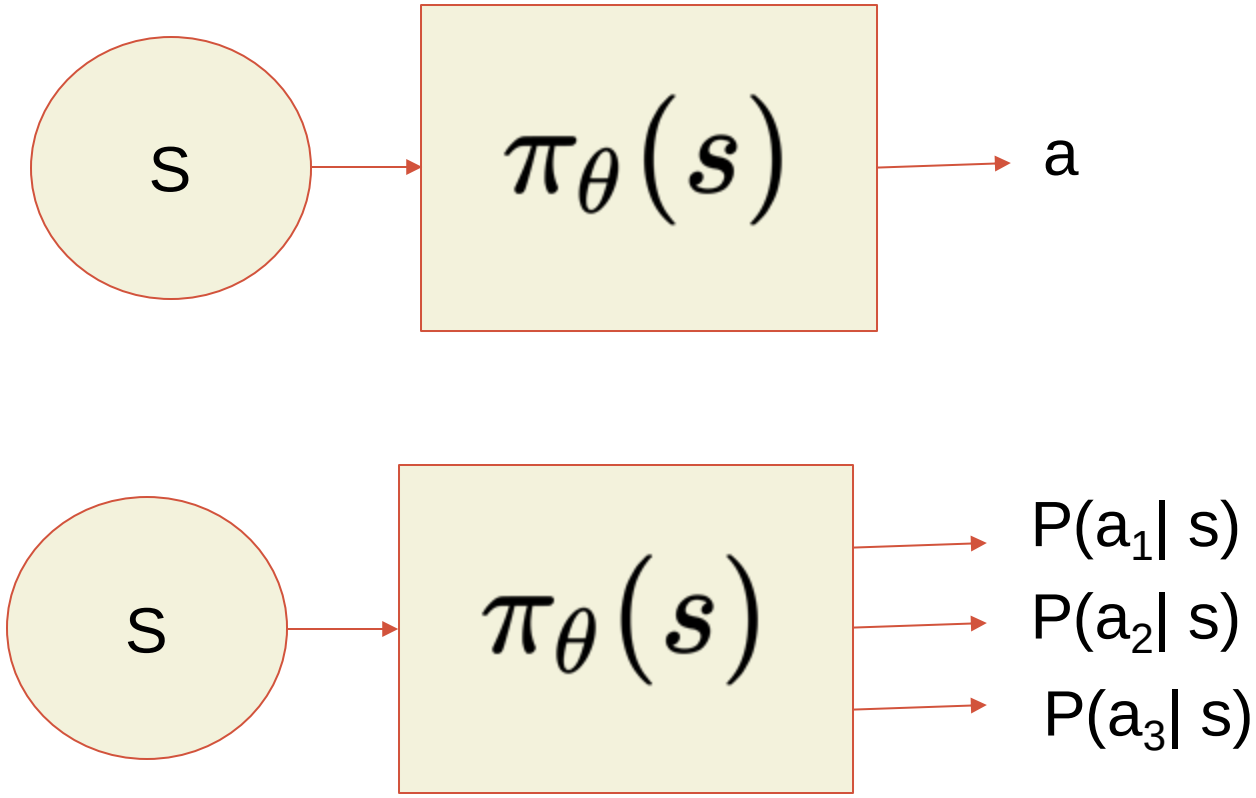
- policy is implicit
- if we already have Q, we have policy
- we just look at Q to get π
- Used in AlphaGo

Policy gradient

- learns π directly explicitly
- use Q, V as a helper for learning π
- Used in pretty much everything else

Policy gradient

- Use Function Approximator to represent policy directly



Loss function for policy gradient

- Start-state objective

$$J(\theta) = E_{\pi(\theta)}[G|s_0] = V^{\pi}(s_0)$$

- Average-reward objective

$$J(\theta) = \sum_s d^{\pi}(s) \sum_a \pi(s,a) r(s,a)$$

* **d** is a stationary distribution of a Markov chain.

Policy gradient

Let's start from the average-reward objective

$$J(\theta) = \sum_s d^{\pi}(s) \sum_a \pi_\theta(s,a) r(s,a)$$

For simplicity let's assume d(s) does not depend on θ

$$J(\theta) = \sum_s d(s) \sum_a \pi_\theta(s,a) r(s,a)$$

$$\nabla_\theta J(\theta) = \sum_s d(s) \sum_a \nabla_\theta \pi_\theta(s,a) r(s,a)$$

Almost there...

Policy gradient

REINFORCE trick!

$$\nabla_\theta J(\theta) = \sum_s d(s) \sum_a \nabla_\theta \pi_\theta(s,a) r(s,a)$$

$$\nabla_\theta J(\theta) = \sum_s d(s) \sum_a \pi_\theta \nabla_\theta \log \pi_\theta(s,a) r(s,a)$$

We get this by sampling a playthrough using current policy (on-policy)

$$\nabla_\theta J(\theta) = E_\pi[\nabla_\theta \log \pi_\theta(s,a) r(s,a)]$$

$$\nabla_\theta J(\theta) \approx \nabla_\theta \log \pi_\theta(s,a) r(s,a)$$

Policy gradient theorem

There is a theorem…called policy gradient theorem say that we can replace $r(s,a)$ with $Q(s,a)$

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[Q^{\pi_\theta}(s,a)\nabla_\theta log\pi_\theta(s,a)]$$

What is the gradient doing?

Goal maximize rewards

Push **π** towards directions of higher Q(s,a)

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[Q^{\pi_\theta}(s,a)\nabla_\theta log\pi_\theta(s,a)]$$

Q(1,1) = 5

Q(1,2) = 2

- **π**(1,2) will have a higher weight. Policy gets push towards action 2

Notes on Policy gradient

Also known as REINFORCE or likelihood ratio.

Used by other ML fields when original loss is not differentiable (-Q in this case). Push network to produce lower loss.

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[Q^{\pi_\theta}(s,a)\nabla_\theta log\pi_\theta(s,a)]$$

Example of non-differentiable functions

argmax (not maxpool)

sampling

Many metrics such as accuracy, preference (ranking)

Baselines

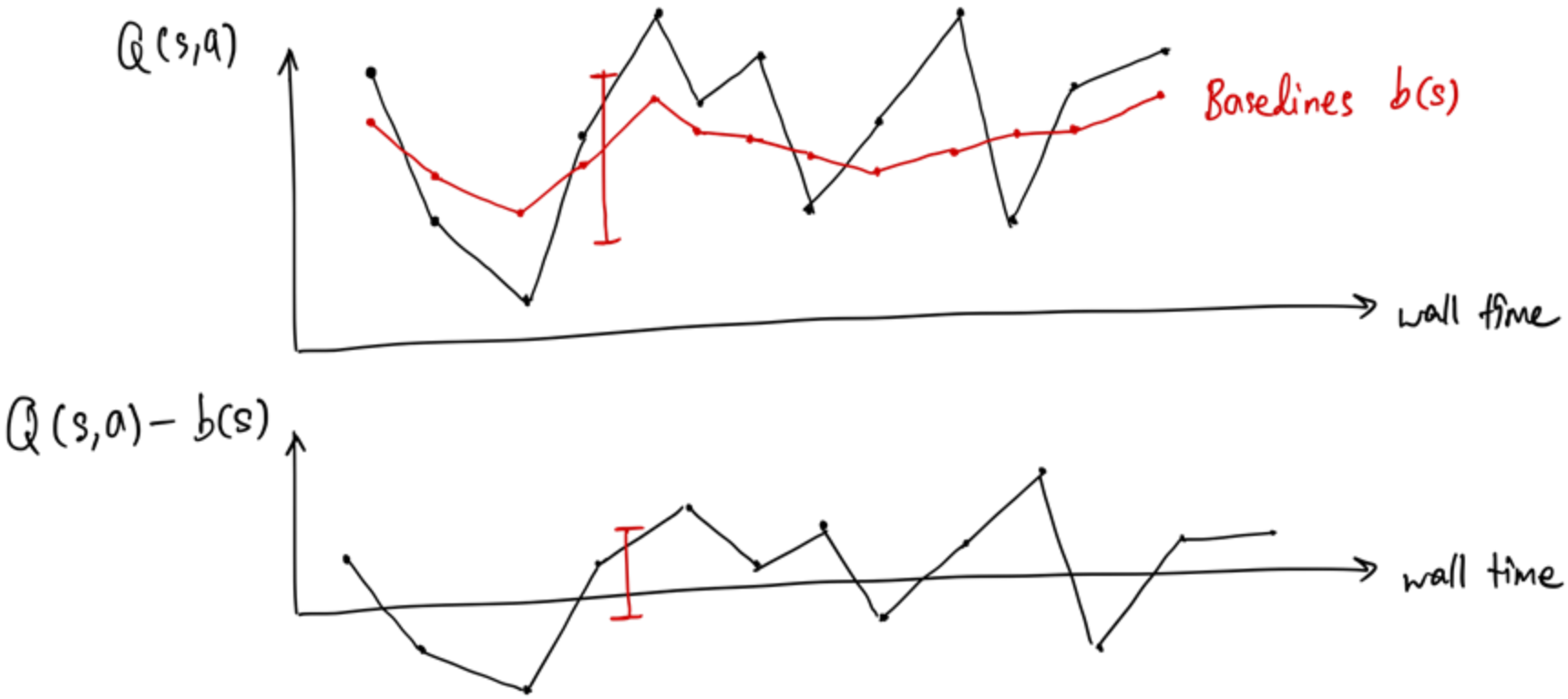
$Q_\pi(s,a)$ has high variance

$Q_\pi(s,a)\nabla_\theta \log \pi_\theta(a|s) = \nabla_\theta J(\theta)$ is very noisy

Slow down training a lot!

We need to reduce variance to speed up the training

Baselines reduce variance



What is a good b(s)?

$V_\pi(s)$ is a convenient choice

Advantage Function

When we use V(s) as baseline:

$$A(s,a) = Q(s,a) - V(s)$$

We call this the **advantage function**.

It tells the relative value of the actions.

Lower variance than using absolute value of the actions.

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[A(s,a)\nabla \log \pi_\theta(a|s)\right]$$

Also called A2C

Noisy updates

- In RL, the updates can be noisy. Our learning is based on Q, V, and A and might not be an accurate estimate if the policy changes too much.
- Limit the update to a certain size? Use KL divergence that new policy shouldn't be that different from old policy

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

$$\nabla J(\theta) = \mathbb{E}_{s,a}\left[A(s,a) \nabla \log \pi_\theta(a|s)\right]$$

$$J(\theta') - J(\theta) = \sum_t \mathbb{E}_{s_t \sim P'(s_t)}\left[\mathbb{E}_{a_t \sim \pi}\left[\frac{\pi'(a_t|s_t)}{\pi(a_t|s_t)} \gamma^t A^\pi(s_t, a_t)\right]\right]$$

Thus, we want to make 1-e < $\frac{\pi'}{\pi}$ < 1+e

(clipping the value)

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right] \qquad r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t \mid s_t)}$$

Proximal Policy Optimization (PPO)

- PPO is just a policy gradient update with Advantage function and clipping the value so that the model does not move outside of the trusted region

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right] \qquad r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$$

- The min is used as a pessimistic bound between no clipping and clipped values.

- PPO is one of the defaults techniques used by OpenAI

https://arxiv.org/pdf/1707.06347

Recap PPO and RL

- RL uses a reward function to learn to optimize our model.
  - Key ingredient: environment (game = input+output) and a way to give reward (scores from output)
- PPO is a policy gradient method with tricks.
  - Key: what moves give me higher reward than before, try to change the gradients so that the move is more likely

InstructGPT

- Step 0: pretrain a language model eg GPT3
- Step 1: Supervised fine-tuning (SFT)
- Step 2: Reward Model training
- Step 3: Reinforcement Learning with Human feedback (RLHF)

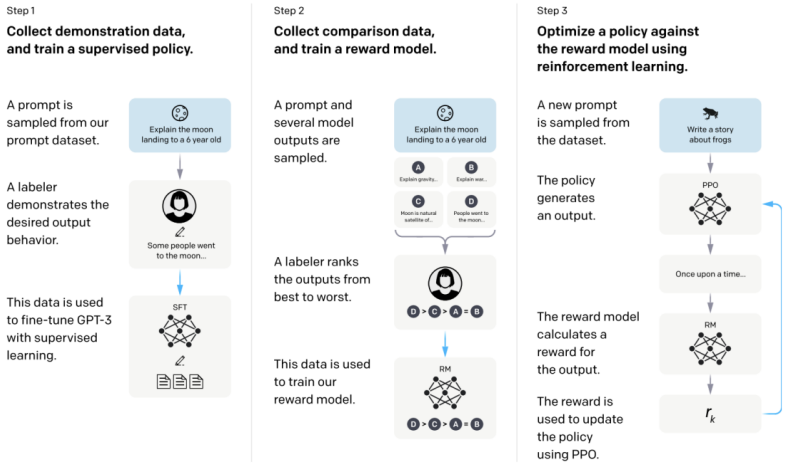
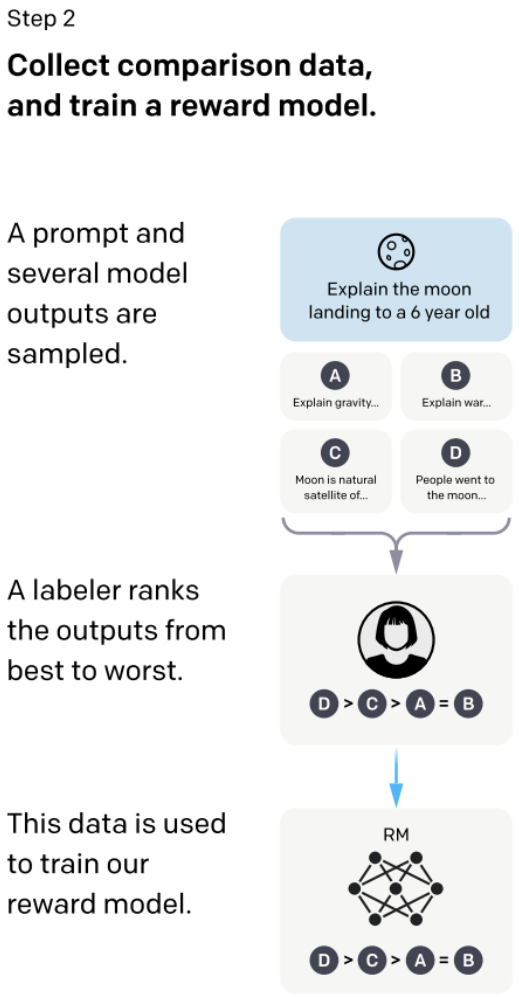


Figure 2: A diagram illustrating the three steps of our method: (1) supervised fine-tuning (SFT), (2) reward model (RM) training, and (3) reinforcement learning via proximal policy optimization (PPO) on this reward model. Blue arrows indicate that this data is used to train one of our models. In Step 2, boxes A-D are samples from our models that get ranked by labelers. See Section 3 for more details on our method.

https://arxiv.org/pdf/2203.02155

Reward model

- We need a way to give rewards for any possible input+output pairs.
  - Technically can be a metric we learned last week, but we talked about how human judgement is still the best
  - We also talked about training a model to mimic human judgement, and how chatGPT can gives scores
- Collect data to train a reward model.
  - OpenAI uses GPT -> finetune -> reward model



Preference data

Preference data needs to be collected for the reward model

Q: ขอคำแนะนำคะ ซื้อของขวัญจับฉลากอะไรดี งบ 400-500 บาทค่ะ

ถ้าคุณมีเพื่อนสนิทหรือคนในครอบครัวที่ชอบอะไรคล้ายๆกัน ลองหาของที่เค้าชอบมาจับสลากดูสิคะ เช่น ตุ๊กตา ของเล่น หนังสือ เสื้อผ้า เครื่องประดับ ฯลฯ รับรองว่าถูกใจทั้งผู้ให้และผู้รับแน่นอนค่ะ	[งบ 400-500 บาท] ขอแนะนำของขวัญจับฉลากที่ดีที่สุดคะ เช่น ฉลากสีขาว ฉลากเงิน ฉลากแข็ง หรือฉลากขนาดใหญ่ที่มีส่วนผสมของฉลากเล็กๆ ที่มีรูปแบบที่สวยงามและมีความสวยงามมากมาย เช่น เครื่องหมายความสำคัญที่สุดในการจัดฉลาก เช่น เสื้อผ้าที่มีสีสันสดใส และฉลากสีเหลืองที่มีความแตกต่าง
A	B

Preference data

- LLAMA – ask annotators to write a prompt -> two sampled model response -> label them (“chosen” and “rejected”)
  - LLAMA3 added a “edited” response, where chosen is further editted

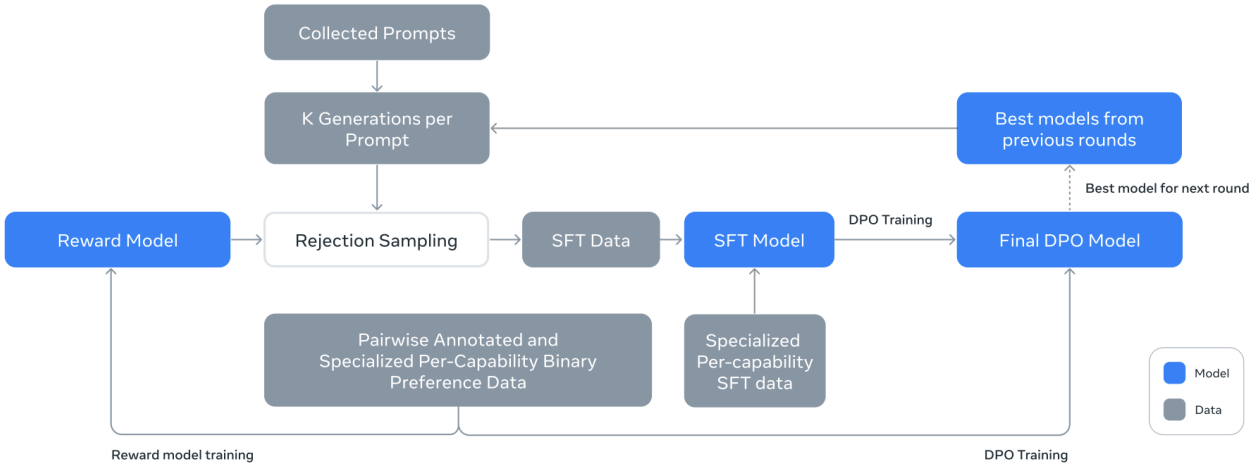


Figure 7 Illustration of the overall post-training approach for Llama 3. Our post-training strategy involves rejection sampling, supervised finetuning, and direct preference optimization. See text for details.

https://ai.meta.com/research/publications/the-llama-3-herd-of-models/

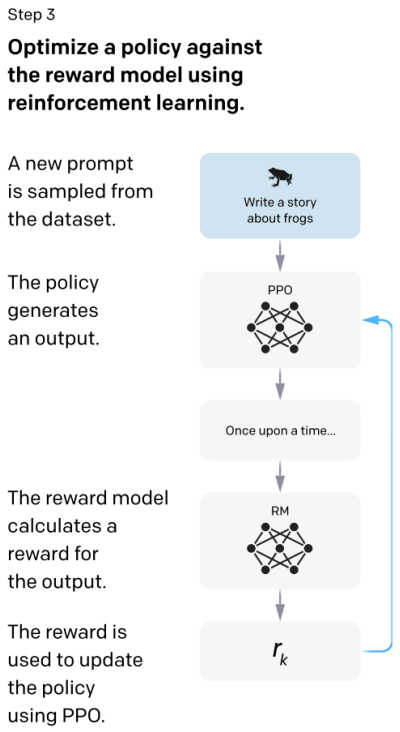
Reward model training

- Many possible settings
- Contrastive loss
  - R(x, y_chosen) > R(x, y_rejected)
- Predict some score value
- Use a heuristic
  - Deepseek R1 is trained on code and math problems.
    - Scores can be checked against the real solution.

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

Reinforcement learning with human feedback (RLHF)

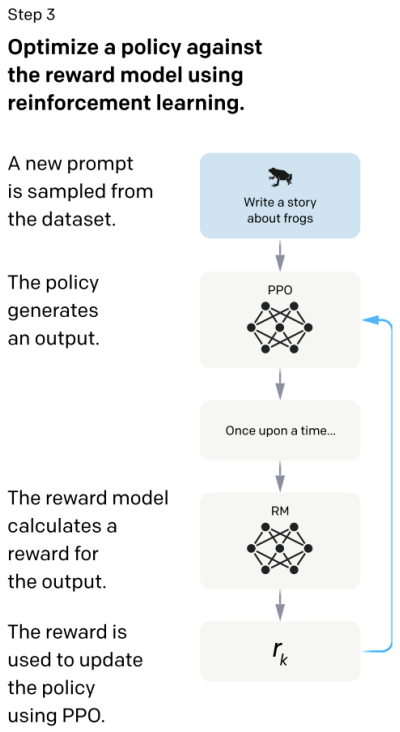
- The Reward model is used to trained with PPO
- They also add a per-token KL penalty from the SFT model so that the optimized model does not go too far away from the SFT model



$$r_t = r_\varphi(q, o_{\leq t}) - \beta \log \frac{\pi_\theta(o_t|q, o_{<t})}{\pi_{ref}(o_t|q, o_{<t})}$$

Reinforcement learning with human feedback (RLHF)

- The Reward model is used to trained with PPO
- But, let's think again why do we need RL?
  - This task is hard to do supervised learning
    - Cannot think of all possible answer and scoring for creative tasks
  - RL offers a scalable solution that can covers multiple domain
  - Can discover new moves
    - "move 37"
    - Deepseek aha moment
  - Drawbacks: sophisticated framework. Hard to train



Deepseek v3 aha moment

- Probably won't show up in supervised data

Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a+x}} = x$ is equal to
Response: <think> To solve the equation $\sqrt{a - \sqrt{a+x}} = x$, let's start by squaring both $\cdots$ $\left(\sqrt{a - \sqrt{a+x}}\right)^2 = x^2 \implies a - \sqrt{a+x} = x^2$. Rearrange to isolate the inner square root term: $(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$ $\ldots$ Wait, wait. Wait. That's an aha moment I can flag here. Let's reevaluate this step-by-step to identify if the correct sum can be $\cdots$ We started with the equation: $\sqrt{a - \sqrt{a+x}} = x$ First, let's square both sides: $a - \sqrt{a+x} = x^2 \implies \sqrt{a+x} = a - x^2$ Next, I could square both sides again, treating the equation: $\cdots$ $\ldots$

Table 3 | An interesting "aha moment" of an intermediate version of DeepSeek-R1-Zero. The model learns to rethink using an anthropomorphic tone. This is also an aha moment for us, allowing us to witness the power and beauty of reinforcement learning.

https://arxiv.org/abs/2501.12948

Notes on LLM

- From what we learned so far
  - LLM is autoregressive
    - Errors cascade
  - LLM is just next word prediction trained on trillions of tokens
    - Likes most probable next token, with some randomness mixed in
  - Trained to reward human preference
- Can you see why hallucination is pretty much inherent in LLMs?
- Backtracking might be an important capability for LLMs to do well in this setting.

Other preference learning technique

History

- In 2023-early 2024, people thinks that RLHF seems to be the secret sauce to make ChatGPT work.
- Many people proposes alternatives to PPO
  - DPO
  - ORPO
  - GRPO

Direct Preference Optimization (DPO)

- Skips reward model training
- Learns directly from the preference data

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

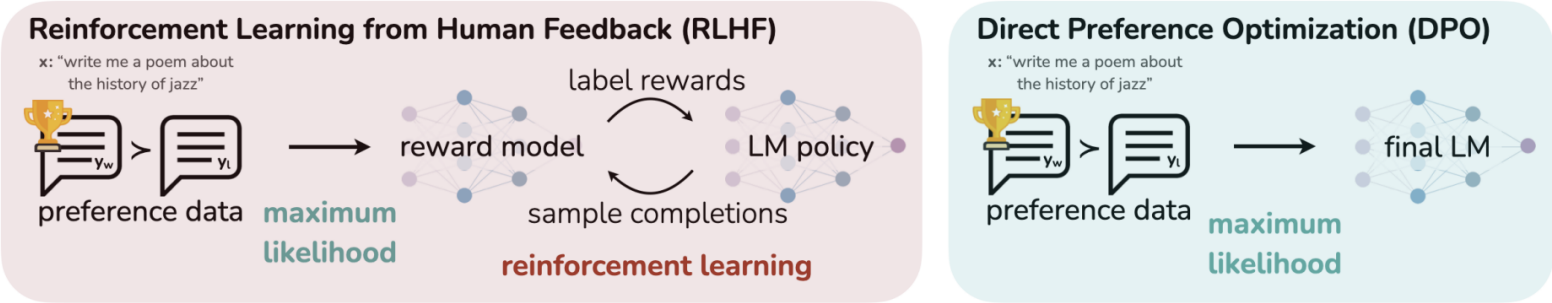


Figure 1: **DPO optimizes for human preferences while avoiding reinforcement learning.** Existing methods for fine-tuning language models with human feedback first fit a reward model to a dataset of prompts and human preferences over pairs of responses, and then use RL to find a policy that maximizes the learned reward. In contrast, DPO directly optimizes for the policy best satisfying the preferences with a simple classification objective, fitting an *implicit* reward model whose corresponding optimal policy can be extracted in closed form.

https://arxiv.org/pdf/2305.18290

Why does DPO work?

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

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

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

Assuming binary reward model $\hat{r}_\theta(x,y) = \beta\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$

A2C update $\nabla J(\theta) = \mathbb{E}_{s,a}\left[A(s,a)\nabla\log\pi_\theta(a|s)\right]$

- DPO is just a derived version of PPO assuming binary choice in the reward model.

Odds Ratio Preference Optimization (ORPO)

$$\textbf{odds}_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)}$$

Odds(x) = k means x is k times more likely than not

- DPO requires you to train SFT then do DPO
- ORPO tries to do this in one step

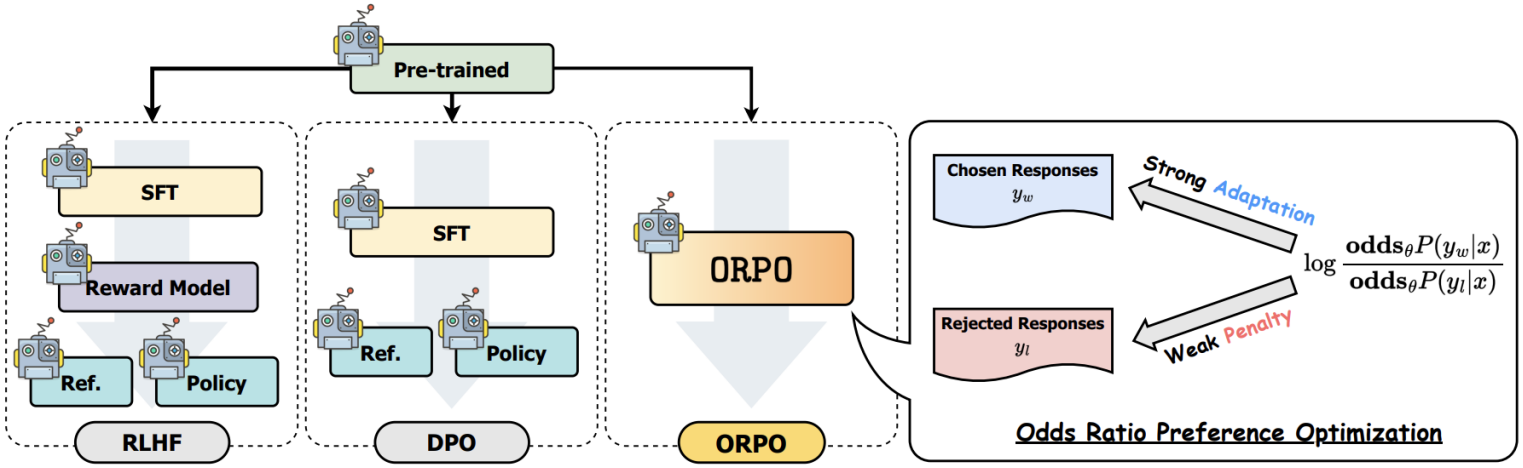


Figure 2: Comparison of model alignment techniques. ORPO aligns the language model *without a reference model* in a single-step manner by assigning a weak penalty to the rejected responses and a strong adaptation signal to the chosen responses with a simple log odds ratio term appended to the negative log-likelihood loss.

https://arxiv.org/pdf/2403.07691

ORPO

$$\textbf{odds}_\theta(y|x) = \frac{P_\theta(y|x)}{1 - P_\theta(y|x)}$$

Odds(x) = k means x is k times more likely than not

$$\textbf{OR}_\theta(y_w, y_l) = \frac{\textbf{odds}_\theta(y_w|x)}{\textbf{odds}_\theta(y_l|x)}$$

Oddsratio = how much more likely to generated a chosen than rejected

$$\mathcal{L}_{ORPO} = \mathbb{E}_{(x,y_w,y_l)}\left[\mathcal{L}_{SFT} + \lambda \cdot \mathcal{L}_{OR}\right]$$

$$\mathcal{L}_{OR} = -\log \sigma\left(\log \frac{\textbf{odds}_\theta(y_w|x)}{\textbf{odds}_\theta(y_l|x)}\right)$$

- Note: although they claimed that ORPO doesn't require separate SFT step, in practice people found that it is beneficial to do SFT than ORPO

ORPO vs	SFT	+DPO	+PPO
OPT-125M	73.2 (0.12)	48.8 (0.29)	71.4 (0.28)
OPT-350M	80.5 (0.54)	50.5 (0.17)	85.8 (0.62)
OPT-1.3B	69.4 (0.57)	57.8 (0.73)	65.7 (1.07)

Table 3: Average win rate (%) and its standard deviation of ORPO and standard deviation over other methods on **UltraFeedback** dataset for three rounds. Sampling decoding with a temperature of 1.0 was used.

Problems with fixed preference data.

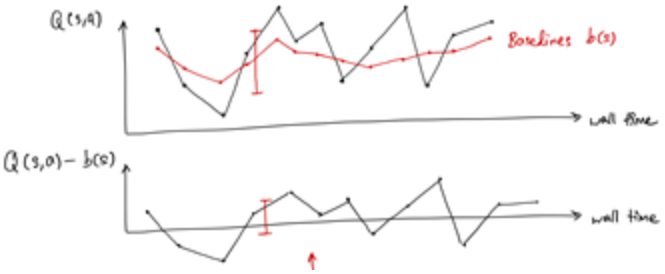
- While ORPO and DPO works well on paper, they are optimized on the SFT or Preference data only. This might cause the model to overfit to the data.
- RL is still required to generalize
  - Or the preference data needs to keep being updated over iterations just like in LLAMA 3.1 or Deepseek

Group Relative Policy Optimization (GRPO)

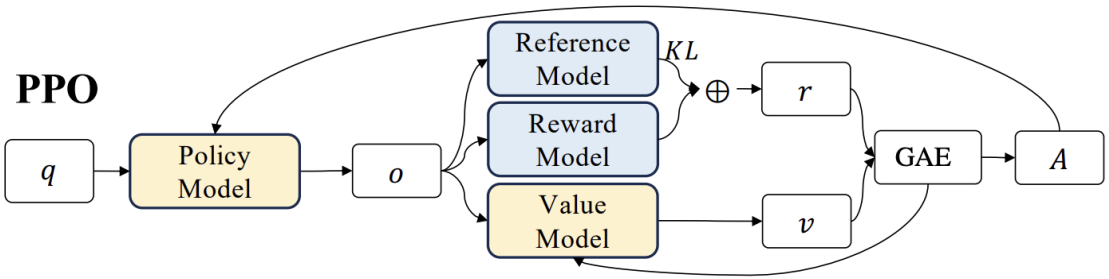
- GRPO is a version of PPO which calculates the advantage function in a different manner

$$\mathcal{J}_{PPO}(\theta) = \mathbb{E}[q \sim P(Q), o \sim \pi_{\theta_{old}}(O|q)] \frac{1}{|o|} \sum_{t=1}^{|o|} \min \left[\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})} A_t, \text{clip}\left(\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})}, 1-\varepsilon, 1+\varepsilon \right) A_t \right]$$

$$r_t = r_\varphi(q, o_{\leq t}) - \beta \log \frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{ref}(o_t|q,o_{<t})}$$



b(s) is typically the Value function V(s)

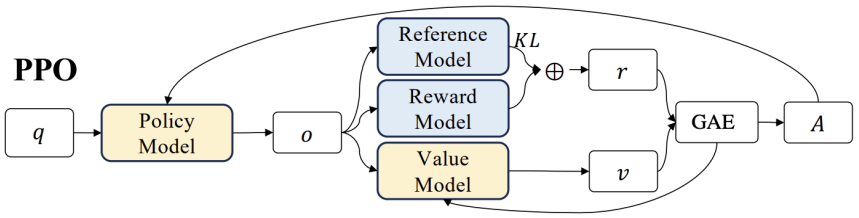


In this framework, the policy and value model is learned Value model is cumbersome to learn because it's the same architecture as the base model.

Also, V(s) might not be a good choice for b(s) because it might be hard to estimate in the LLM context.

https://arxiv.org/pdf/2402.03300

GRPO



$$\mathcal{J}_{PPO}(\theta) = \mathbb{E}[q \sim P(Q), o \sim \pi_{\theta_{old}}(O|q)] \frac{1}{|o|} \sum_{t=1}^{|o|} \min\left[\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})} A_t, \text{clip}\left(\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})}, 1-\varepsilon, 1+\varepsilon\right) A_t\right]$$

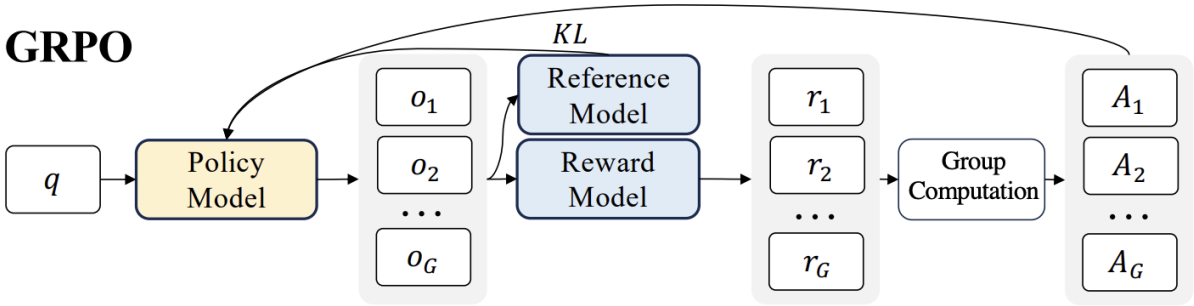
$$r_t = r_\varphi(q, o_{\leq t}) - \beta \log \frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{ref}(o_t|q,o_{<t})}$$

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$

$$\frac{1}{G}\sum_{i=1}^{G} \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left\{\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})} \hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}, 1-\varepsilon, 1+\varepsilon\right) \hat{A}_{i,t}\right] - \beta \mathbb{D}_{KL}\left[\pi_\theta || \pi_{ref}\right]\right\}$$

$$\hat{A}_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})} \qquad \mathbb{D}_{KL}\left[\pi_\theta || \pi_{ref}\right] = \frac{\pi_{ref}(o_{i,t}|q,o_{i,<t})}{\pi_\theta(o_{i,t}|q,o_{i,<t})} - \log \frac{\pi_{ref}(o_{i,t}|q,o_{i,<t})}{\pi_\theta(o_{i,t}|q,o_{i,<t})} - 1,$$

A is averaged over the different outputs from the same input
(We need b(s))



When Policy Gradient is just supervised learning

$$\nabla_\theta \mathcal{J}_{\mathcal{A}}(\theta) = \mathbb{E}[\underbrace{(q,o) \sim \mathcal{D}}_{Data\ Source}] \left(\frac{1}{|o|} \sum_{t=1}^{|o|} \underbrace{GC_{\mathcal{A}}(q,o,t,\pi_{rf})}_{Gradient\ Coefficient} \nabla_\theta \log \pi_\theta(o_t|q,o_{<t}) \right).$$

A.1.1. Supervised Fine-tuning

The objective of Supervised Fine-tuning is maximizing the following objective:

$$\mathcal{J}_{SFT}(\theta) = \mathbb{E}[q,o \sim P_{sft}(Q,O)] \left(\frac{1}{|o|} \sum_{t=1}^{|o|} \log \pi_\theta(o_t|q,o_{<t}) \right). \quad (6)$$

The gradient of $\mathcal{J}_{SFT}(\theta)$ is:

$$\nabla_\theta \mathcal{J}_{SFT} = \mathbb{E}[q,o \sim P_{sft}(Q,O)] \left(\frac{1}{|o|} \sum_{t=1}^{|o|} \nabla_\theta \log \pi_\theta(o_t|q,o_{<t}) \right). \quad (7)$$

The only difference is where the data comes from

Why RL works?

- In DeepSeekMath, they found that models trained with RL perform better than SFT models in “alignment”
  - Top outputs have higher scores but capabilities still remain the same.
- Bottlenecks
  - Rewards learning – how to get nice reward model for subjective tasks
  - Data - sampling of questions and answers for the model to learn
  - Algorithm – most RL algorithms are based on always true rewards

https://arxiv.org/pdf/2402.03300

Summary

- Scaling laws
- ChatGPT training
  - Pre-training
  - Post-training
    - SFT
    - Preference optimization

Midterm

- In class. Room will be announced later.
- 1 page A4 front back
- Calculator

Part A) Aj.Peerapon (25 points)
- Tokenization
- LM + subwords
- Attention + transformer
- Text gen + eval measure

Part B) Aj.Ekapol (25 points)
- Deep learning/Pytorch
- Word representation (sparse, dense)
- PoS (HMM)
- Sentence Embedding/Text classification
- Open question (design)

Takehome

Project

Group of 2-5 people

We expect more from more people

Anything text/NLP related

Must has some application component (cannot be purely basic NLP task)

24-Feb-2025	1-Mar-2025	8	LLM take home exam (kaggle) + Project Announcement + Paper Announcement
3-Mar-2025	8-Mar-2025		Midterm Exam Week (3-7 Mar 2025) - no class
10-Mar-2025	15-Mar-2025	9	Midterm Exam (paper exam - in class)
17-Mar-2025	22-Mar-2025	10	Prompting/Lora/Adaptation techniques, HW8 out, HW9 out
24-Mar-2025	29-Mar-2025	11	Recent topics (bias, benchmark, RAG, agentic, test-time compute), HW10 out
31-Mar-2025	5-Apr-2025	12	Paper Presentation & Progress Report
7-Apr-2025	12-Apr-2025		No class
14-Apr-2025	19-Apr-2025	13	Guest - deployment [quantization, deployment, etc.]
21-Apr-2025	26-Apr-2025	14	Guest - multimodal
28-Apr-2025	14-May-2025		Final Exam Week (28 Apr -14 May 2025); No Final Exam
TBA	TBA	15	Project Presentation

HOW TO READ A SCIENTIFIC ARTICLE

2 Paper types

- Review article/tutorial
  - Give insights about the field
  - Useful for learning about a new field
  - Read multiple to avoid the author's bias
  - Title usually has "review" or "tutorial"
- Primary research article
  - More details on the experiments and results

Parts of an article

- Abstract
- Introduction
- Methods
- Results and discussion
- Conclusion
- Reference

Things to look for before reading an article

- Publication date
- Author names
  - Previous and newer publications
- Keywords
- Acknowledgements and funding sources

ORPO: Monolithic Preference Optimization without Reference Model

Jiwoo Hong **Noah Lee** **James Thorne**

KAIST AI
{jiwoo_hong, noah.lee, thorne}@kaist.ac.kr

https://arxiv.org/pdf/2403.07691

Getting the big picture

- Read the abstract
- Read the introduction
  - What is the research question?
  - What is the method?
  - What had been done? How is it different from other work?
- Look at figures and results

Tip: keep track of terms you don’t understand

First reading

- Reread the introduction
- Skim methods
- Read results and discussion
  - Does the figures make sense now?
- Write on the article!

Understanding the article

- Reread the article (until you get what you want)
- Check references for parts you don’t understand
- Reread the abstract
  - Does your understanding match the abstract?
- Note down important points. This might come in handy when you write you paper/thesis!

Evaluating the article

- Does the method make sense?
  - What are the limitations that the authors mention?
  - Are there other limitations?
  - Can it be used in other situations?
- Are the experiments legitimate?
  - The sample size is big enough? How big?
  - What kind of dataset is used? Hard enough to current standards? Baselines?
  - The evaluation criterion is sound?
- Have these results been reproduced?
  - Look for articles that cite this paper

ML paper checklist

- What is being done?
- How is it being done?
  - How is it different from previous work
- What is the dataset?
  - Nature of dataset. How many training/testing samples? How many classes/vocab size? Etc.
- Evaluation
  - What are the baselines? Metrics is okay?
- Practicality/limitations
  - Prone to parameter tuning?
  - Computing resource / Runtime (training and testing)
  - Other assumptions? Supervised, unsupervised, etc.

Useful tools

- https://scholar.google.com
  - For finding other articles by the same authors or paper that cites the article

- Deep research of various kinds are quite useful for researching a topic. Be careful of hallucinations even if it has citation!

- Many famous papers has web presence
  reddit, twitter, or openreview
  video explaining the paper

Paper presentation

Pick from the list of papers we provide (right after midterm exam)

8 minute presentation + 2 minute progress update + 2 minute QA

Should cover the ML paper checklist