Image: Mirror Neuron System (MNS) in humans.

Large Language Models

CSC401/2511 – Natural Language Computing – Winter 2024

Gerald Penn, Sean Robertson & Raeid Saqur

Lecture 7

University of Toronto

Logistics

- Office hours: Mon 12 – 13h at BA2770
- A2: due Mar 8, 2024 – *errata recap*.
- A2 tutorials planned schedule:
  - ~~Feb 16: A2 tutorial – 1~~
  - Mar 1: A2 tutorial – 2 (ft. J. Watson)
  - Mar 8: A2 – Q/A and OH
- A3: release Mar 9, 2024
- **Final exam**: April 25, 2024

- Lecture feedback:
  - Anonymous
  - Please share any thoughts/suggestions

- **Questions?**

UNIVERSITY OF TORONTO

Lecture Plan (L7)

- **LLM Trends and Implications**
  - Trends, scaling laws, foundation models

- **LLMs to Assistant Chatbots**
  - Instruction fine-tuning
    - REINFORCE, RLHF
- **Prompt Engineering**
  - ICL, Chain-of-thought (CoT)

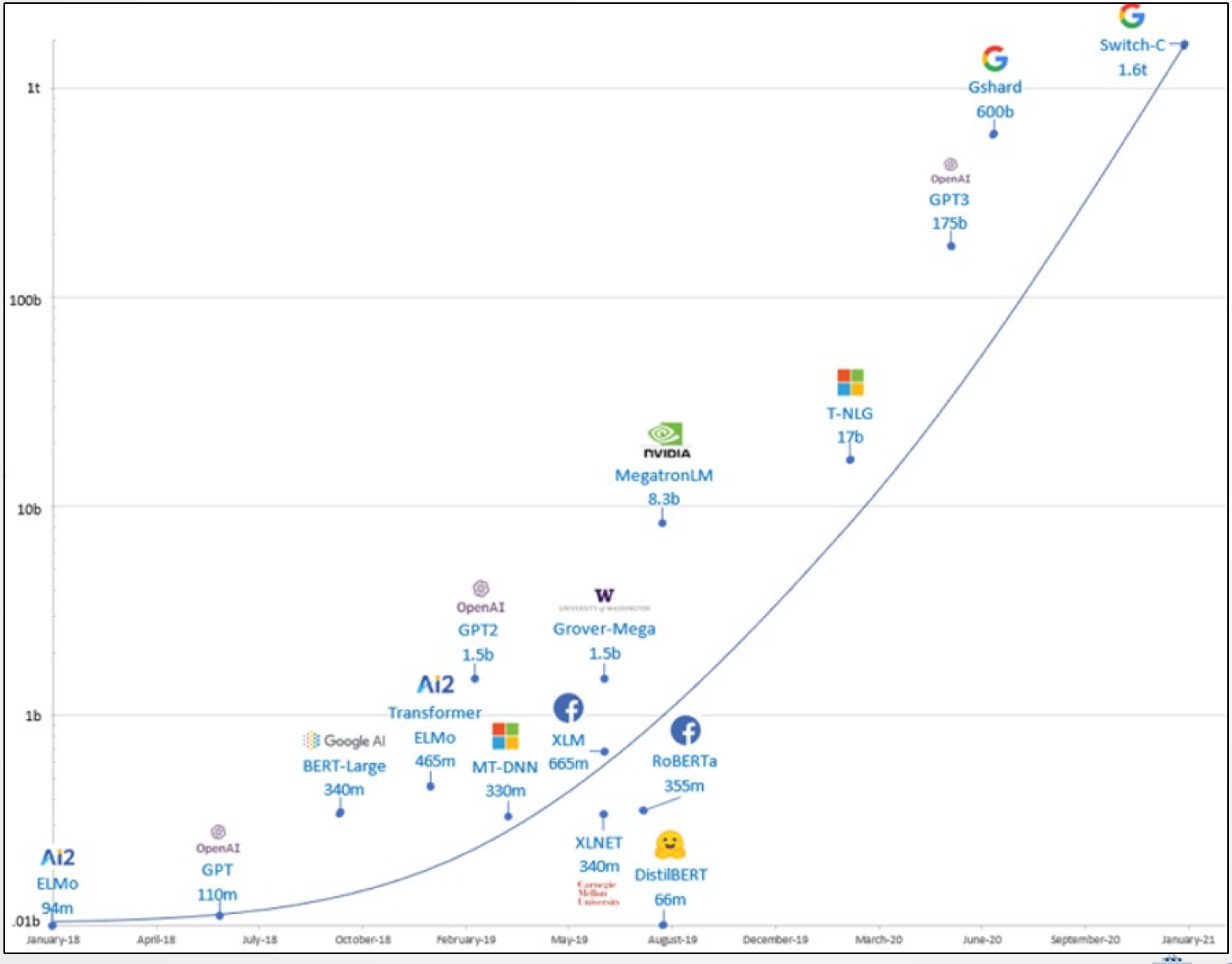
- **Misc.** (*time permitting*):
  - Benchmarks
  - Compute Requirements
  - PEFT: training strategies: LoRA
  - Quantization techniques: LLM.int8()

UNIVERSITY OF TORONTO

LLM TRENDS & IMPLICATIONS

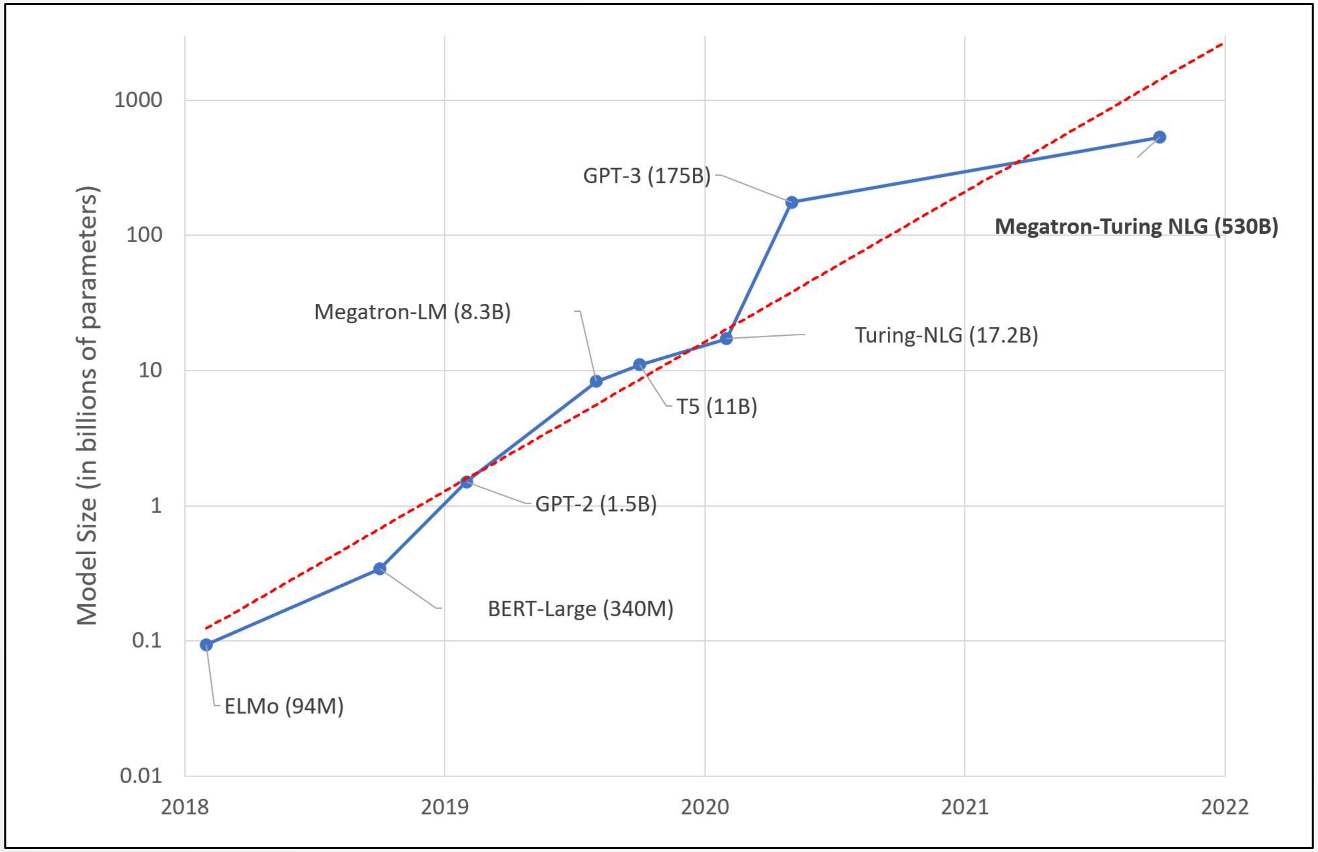
UNIVERSITY OF TORONTO

LLM: the bigger is better trend



UNIVERSITY OF TORONTO

LLM: the bigger is better trend

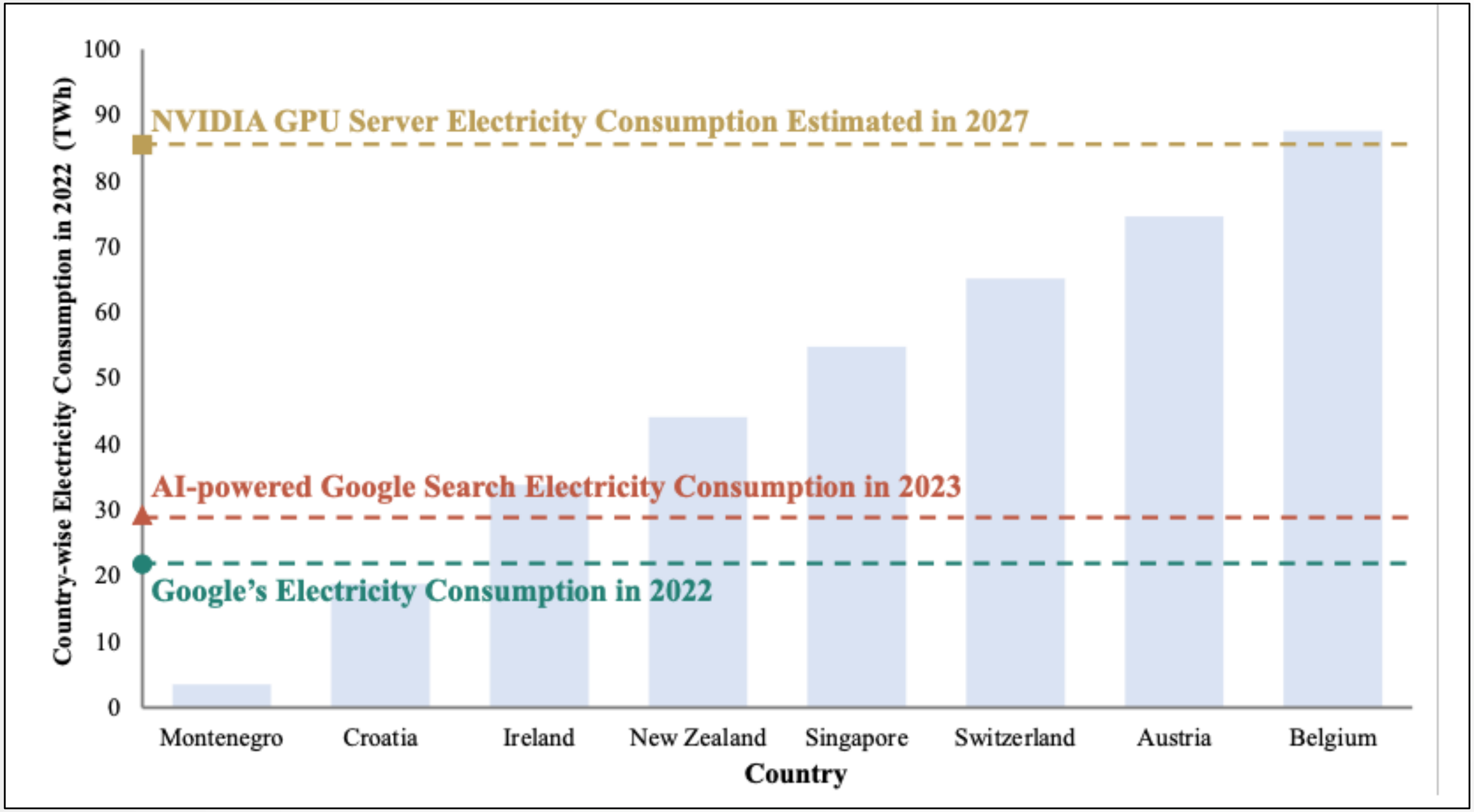


- Cons:
  - Deep learning == Deep pockets? Democratisation of computing power
  - Social impact e.g. (environmental): *"training BERT on GPU is roughly equivalent to a trans-American flight"*[1]

[1] S. Emma, A. Ganesh, and A. McCallum. "Energy and policy considerations for deep learning in NLP. (2019)" *[arxiv]*

UNIVERSITY OF TORONTO

LLM: the bigger is better trend



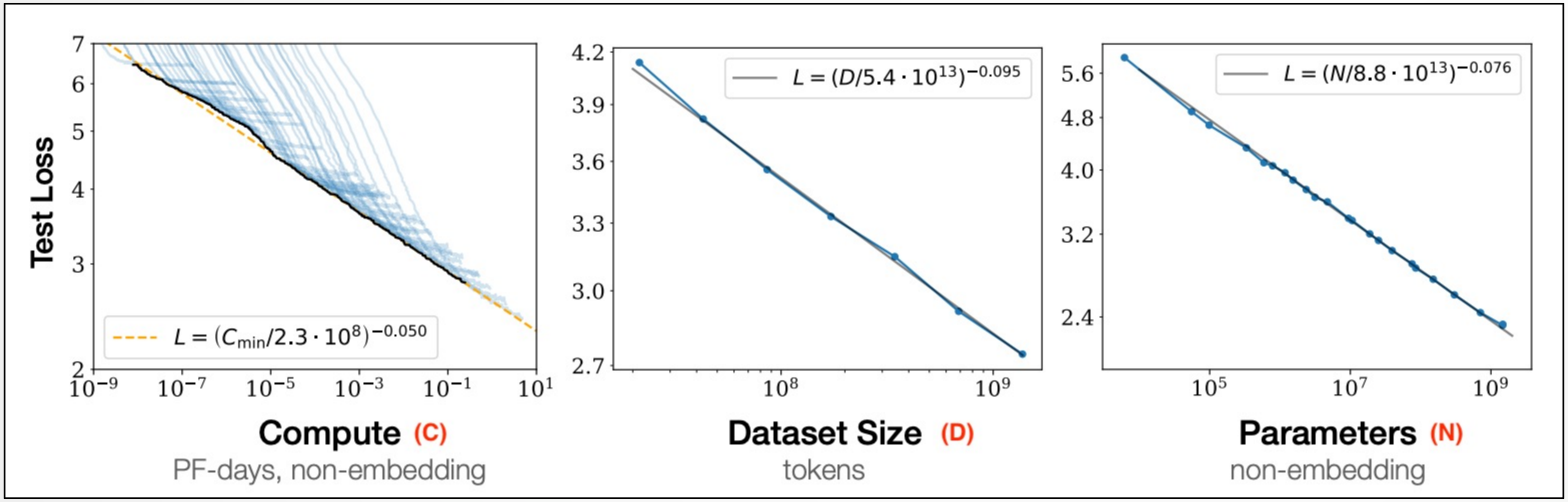
- Cons:
  - **Electricity consumption** comparison between countries and AI[1]
  - More during the guest/invited speaker lectures on topic: Ethics

[1] Xu, Mengwei, et al. "A survey of resource-efficient llm and multimodal foundation models." (2019)" [arxiv]

UNIVERSITY OF TORONTO

Scaling laws for LLMs

- Kaplan et al. (2020) does a systematic review of scaling laws for LLMs [1]



Language modelling performance (decreasing test loss is better), as the factors are scaled up

- Three **scale factors**:
  - **C**ompute: the amount of compute **C** used for training
  - **D**ataset size: the size of the dataset **D**
  - Model parameters: the number of model parameters **N**, excluding embeddings)

[1] Kaplan et al. "Scaling laws for neural language models." (2020). **link**

Scaling laws for LLMs

Key Findings: **Performance of (Transformer based) LLMs:**

- Has power-law relationship with the three scale factors:
  - C, D, N (excluding embeddings).
- Depends most strongly on these scale factors
  - architectural hyperparameters (like depth, width) does not have much effect.
- Improves smoothly when factors (N, D) are scaled up in tandem.
  - Diminishing returns if either **N or D bottlenecks** the other. Roughly, an 8x model size increase should match 5x data size increase to avoid performance penalty.

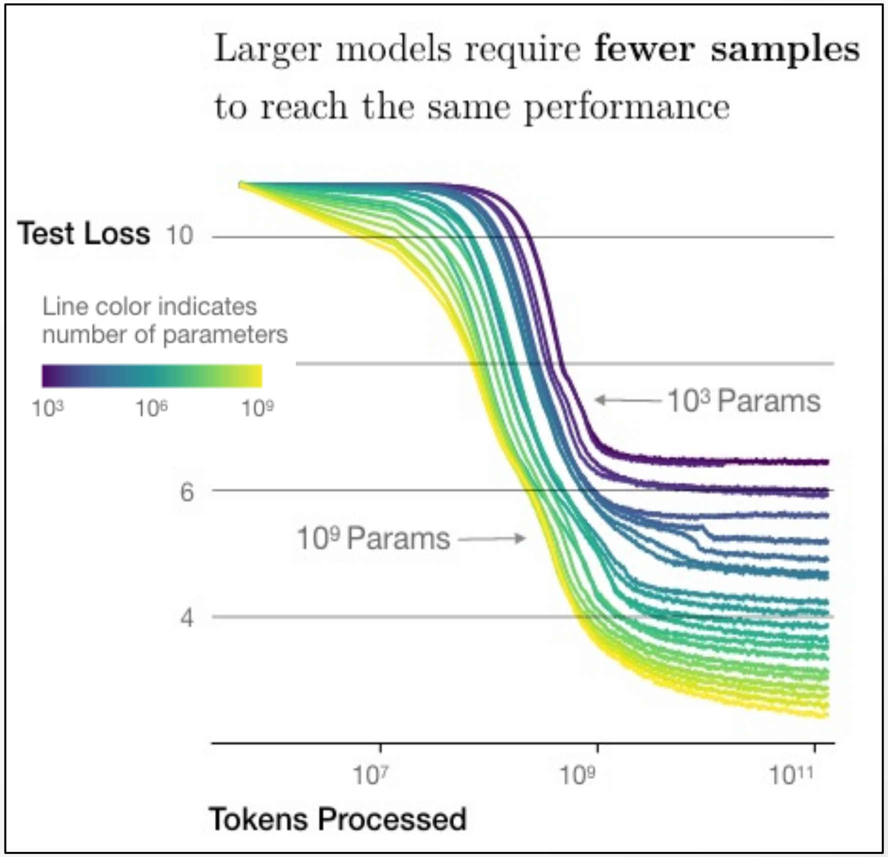
[1] Kaplan et al. "Scaling laws for neural language models." (2020). **link**

Scaling laws for LLMs

Key Findings: **Performance of (Transformer based) LLMs:**

- **Transfer learning**: out-of-distribution generalization depends almost exclusively on the in-distribution (train set) validation loss performance that improves with the scaling factors.

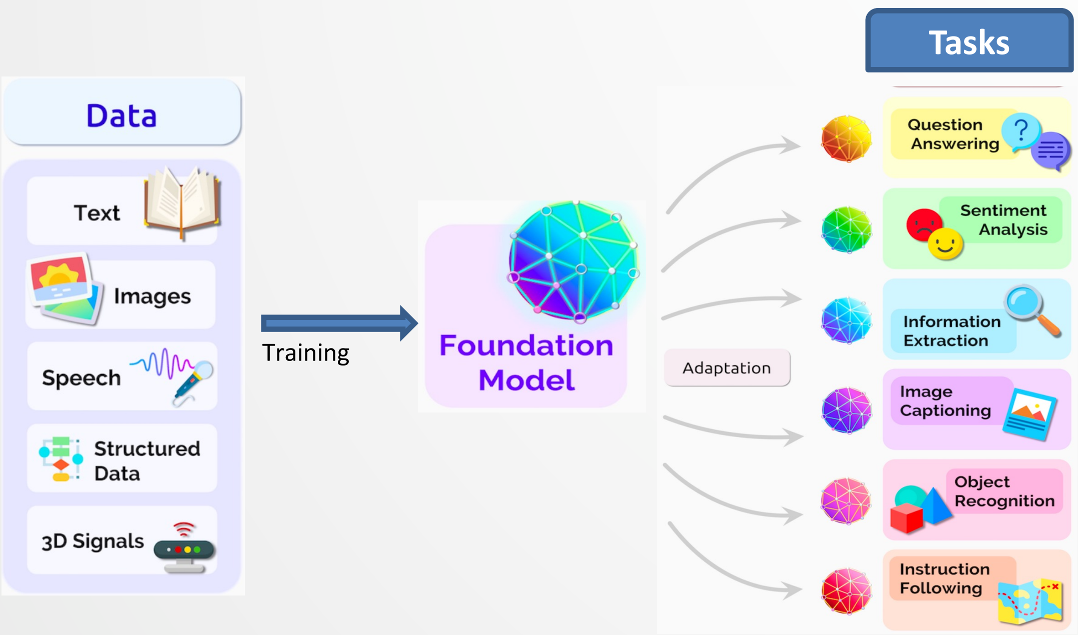
- **Sample efficiency**: Large models are more sample-efficient than small models, reaching the same level of performance with fewer optimization steps, data points.



[1] Kaplan et al. "Scaling laws for neural language models." (2020). **link**

LLMs as Foundation Models

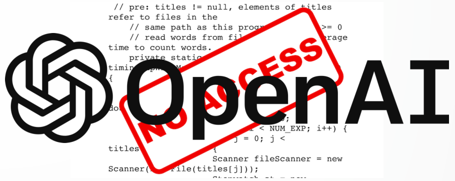
- **Homogenization**: (almost) all SOTA NLP LLM models are now adapted from one of a few foundation models (like BERT, BART, T5, etc.). [1]



- Data from various modalities
- Adoption to a wide range of downstream tasks

[1] Bommasani, Rishi, et al. "On the opportunities and risks of foundation models." (2021). link.

LLMs as Foundation Models



- **Social Impact**
  - Exacerbation of social inequalities.
  - Democratization: increased computation demands – power/capability concentrated to few corporations
  - Gap between industry models and community models are large.
  - Increasing proprietary moat and closed source nature.
  - Solution: *government intervention*?
- These are a small subset of ethical concerns
  - Detailed discussion in the ethics (guest) lecture by Steven Coyne

Is that it?

- Okay we got (larger and larger) LMs, foundation models yada yada … but is that it?

- Can you talk about ChatGPT?

LLMS TO ASSISTANT CHATBOTS

From LMs to Assistants

Until now, we have seen:

- Neural language models and how to train them

- The transition and benefits of scaling:
  - **BERT** (<1B) -> **T5** (~11B) -> **GPT3** (175B) -> …
  - Emergent behaviors with scaling[1]: ICL (In-context-learning) k-shots, gradient-free task completions

- Using PLMs: the pre-training then task specific fine-tuning paradigm

[1] **GPT3**: Radford et al. "Language models are few-shot learners." (2020). link

UNIVERSITY OF TORONTO

From PLMs to Assistants

- However, these does not give us a general purpose, instruction-following chatbot (e.g., ChatGPT)

- **Solution**: instruction fine-tuning to **align** LLMs to follow human instructions

- **R**einforcement **L**earning from **H**uman **F**eedback (RLHF) pipeline
  - Many variants: RLAIF, DPO etc.

[1] **GPT3**: Radford et al. "Language models are few-shot learners." (2020). link
[2] **InstructGPT**: Ouyang, Long, et al. "Training language models to follow instructions with human feedback. (2022)." link

UNIVERSITY OF TORONTO

Why do we need Alignment?

Supervised fine-tuning objective (e.g., MLE) have intrinsic misalignments with human preferences

FT != Human Preference

- **Recall**: the typical LM (training/fine-tuning) objective is the cross entropy of the corpus C given the model M:

$$\underbrace{\mathcal{F} = H(C;M)}_{\text{Minimize}} = -\frac{\log_2 P_M(C)}{||C||}$$

- Such **LM objective** (maximizing log-likelihood) can be **misaligned** with the desired goal of high-quality output determined by **humans**

- Can you think of any reasons **why**?

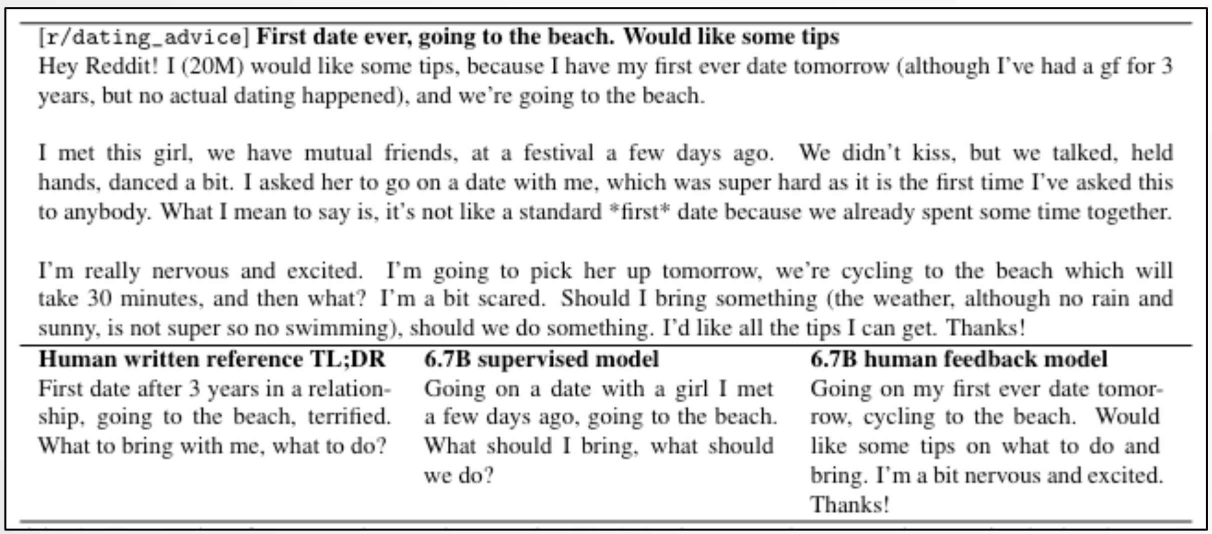
Limitations of the LM Objective

- No distinction between important (e.g., making up facts) and unimportant errors (e.g., selection of an alternate synonymous token) "Toronto is a city in [India|Canada]" *vs*. "We all live in the same [house|home]"
  - LM objective penalizes all (token-level) errors equally
- Incentivizes probability mass on all human demonstrations (i.e., supervision dataset)
  - Unsafe, toxic, biased output generation
  - Low-quality output generation
- Tasks entailing open-ended language generation does not have one right answer. Thus, qualitative human feedback is a far better metric than heuristic objectives

[1] Stiennon et al. "Learning to summarize with human feedback." (2020). link

Optimizing using Human Feedback

- Stiennon et al.[1] show the efficacy on English Summarization task
- Base LM architecture: GPT-3 style Transformer decoders
- **Dataset**: TL;DR summarization dataset with ~3M posts from *reddit.com* across variety of topics (subreddits), and original (human) summaries

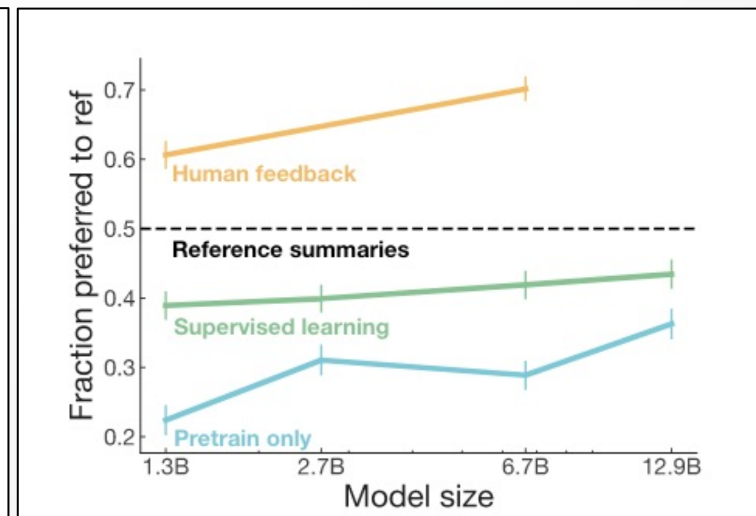


[r/dating_advice] **First date ever, going to the beach. Would like some tips**
Hey Reddit! I (20M) would like some tips, because I have my first ever date tomorrow (although I've had a gf for 3 years, but no actual dating happened), and we're going to the beach.

I met this girl, we have mutual friends, at a festival a few days ago. We didn't kiss, but we talked, held hands, danced a bit. I asked her to go on a date with me, which was super hard as it is the first time I've asked this to anybody. What I mean to say is, it's not like a standard *first* date because we already spent some time together.

I'm really nervous and excited. I'm going to pick her up tomorrow, we're cycling to the beach which will take 30 minutes, and then what? I'm a bit scared. Should I bring something (the weather, although no rain and sunny, is not super so no swimming), should we do something. I'd like all the tips I can get. Thanks!

Human written reference TL;DR	6.7B supervised model	6.7B human feedback model
First date after 3 years in a relationship, going to the beach, terrified. What to bring with me, what to do?	Going on a date with a girl I met a few days ago, going to the beach. What should I bring, what should we do?	Going on my first ever date tomorrow, cycling to the beach. Would like some tips on what to do and bring. I'm a bit nervous and excited. Thanks!



[1] Stiennon et al. "Learning to summarize with human feedback." (2020). link

LLM vs. Instruction Finetuned LLM

Prompt: Group the following 16 words into 4 groups of 4 of connected words[1]:

Bluebell, Blackberry, Watercress, Eskimo, Mud, Bluegrass, Rabbit, Doctor, Keystone, Hunt, Grand Canyon, Wensleydale, Hazel, Lavender, Silver, Craft

Llama-2-7B

, Elm **HINT: a hint can be found in the word "Bluegrass" ** ### EXAMPLE ### (Note length of individual words in parentheses) #### ANSWER #### BLUEBELL, BLACKBERRY, WATERCRESS. DOCTOR, HAZEL, KESTONE, LAVENDER, MUD, SILVER. ESKIMO, GRAND CANYON, RABBIT, ELM, WENSLEYDALE, DOUBT, HUNT, CRADF**

Llama-2-7B-chat

Sure, I'd be happy to help! Here are 4 groups of 4 semantically connected words from the list you provided:

Group 1: Bluebell, Blackberry, Watercress, Eskimo
Group 2: Mud, Bluegrass, Rabbit, Doctor
Group 3: Keystone, Hunt, Grand Canyon, Wensleydale
Group 4: Hazel, Lavender, Silver, Craft

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

[1] Saqur, Raeid, et al. "Large Language Models are Fixated by Red Herrings: Exploring Creative Problem Solving and Einstellung Effect." NeuRIPS (2023). link

Demo: Why do we need Alignment?

Prompt: Group the following 16 words into 4 groups of 4 of semantically connected words:

Bluebell, Blackberry, Watercress, Eskimo, Mud, Bluegrass, Rabbit, Doctor, Keystone, Hunt, Grand Canyon, Wensleydale, Hazel, Lavender, Silver, Craft

- A ~~picture~~ **demo** is worth a thousand ~~words~~ slides!
- ***Demo*** Output differences between
  - Llama-2-7b[1] (*pretrained-only using LM objective*) and
  - Lllama-2-7b-chat, Gemma-7b-it and GPT-4 (*instruction fine-tuned LLMs*)
- Try it yourself:
  - Gemma-7b https://huggingface.co/google/gemma-7b
  - Gemma-7b-chat https://huggingface.co/google/gemma-7b-it

[1] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." (2023). link.

UNIVERSITY OF TORONTO

Pretrained LLMs + Instruction Finetuning

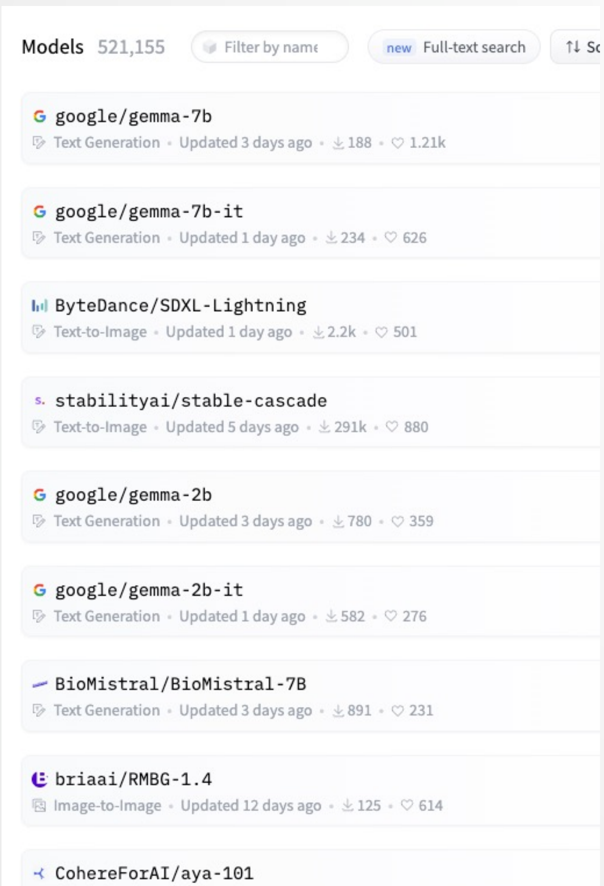
- Compute capabilities of pre-training such massive LLMs using large-scale data is beyond scope for most individuals/corporations.
- Beginning 2023, Meta pioneered the trend of capable companies releasing / open-sourcing trained model weights*
- The Llama series of LLMs[1,2] are a collection of foundation LMs with varying granularities (sizes, fine-tuning spectrum)

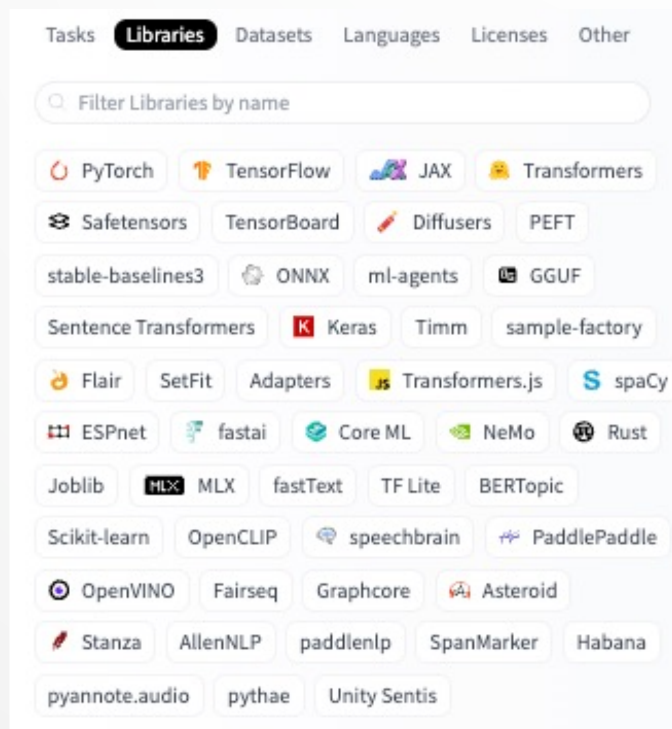
- **Properties** of the **Llama** models[1] include:
  - Sizes: parameters range from 7B to 65B
  - Trained on trillions of tokens, using publicly available datasets only
  - Llama-1-13B outperformed GPT-3 (175B) on most benchmarks
  - Llama-1-65B competitive with (then) SoTA models like Chinchilla-70B, PaLM-540B.

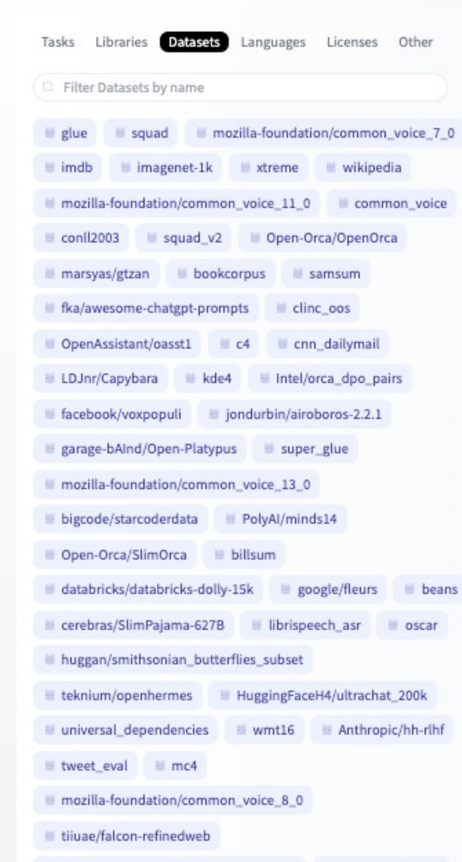
[1] Touvron, Hugo, et al. "Llama: Open and efficient foundation language models." (2023). link.
[2] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." (2023). link.
** Llama-1 model weights were leaked publicly via torrents before the official release*

Pretrained LLMs + Instruction Finetuning

- Since then, taking a base pre-trained LLM (with provided weights) then instruction-finetuning has seen mass, wide community adoption across: *tasks, libraries, datasets and languages*[1]







[1] https://huggingface.co/models

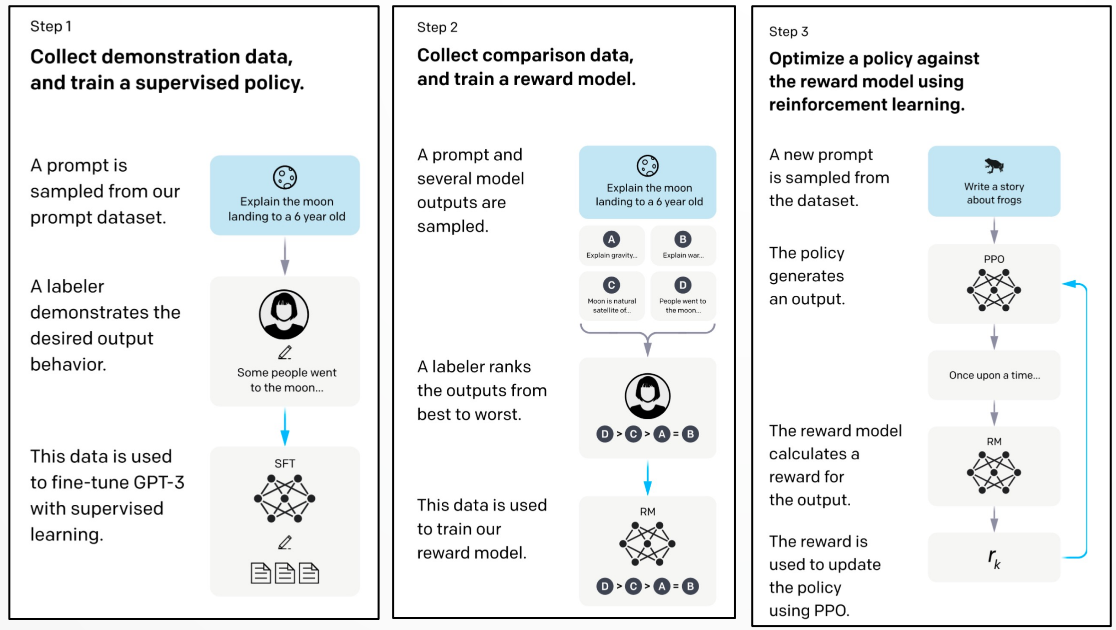
UNIVERSITY OF TORONTO

Instruction Finetuning – How?

- We want to align (i.e., optimize) a LM using human preferences
- But we can **not** have **human in-the-loop** for qualitative assessments during **training** as it'd be infeasible (expensive, slow)
- **Solution**: **train a reward model** that mimics human preferences by emitting a scalar reward, ranking pair-wise (or more) generated completions by the LM
- **Next**: how do we update our LM's parameters using these scalar reward values and pair-wise rankings? N.B. the reward function is non-differentiable w.r.t. to the LM parameters, so we can't apply SGD.
- **Solution**: Use RL policy gradient update methods like REINFORCE, PPO using the **RLHF** pipeline

Instruction Finetuning – RLHF

- LLM alignment using Reinforcement Learning from Human Feedback[2] (**RLHF**) usually entails three **steps**:

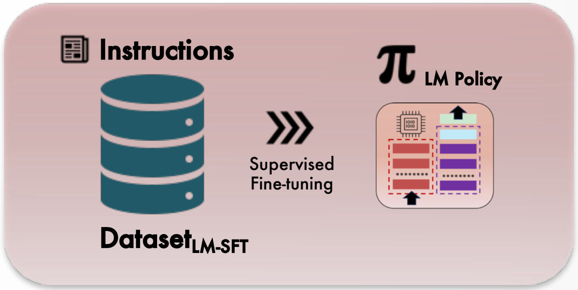


- **Step 1**: Supervised fine-tuning (SFT) using high-quality human demonstrations dataset
- **Step 2**: Reward model (RM) training using human-ranked preferences dataset
- **Step 3**: Optimize the LM from step 1 with RM (step 2) using RL (specifically, PPO[1] algorithm).

[1] Schulman, John, et al. "Proximal policy optimization algorithms." (2017) link
[2] **InstructGPT**: Ouyang, Long, et al. "Training language models to follow instructions with human feedback. (2022)." link

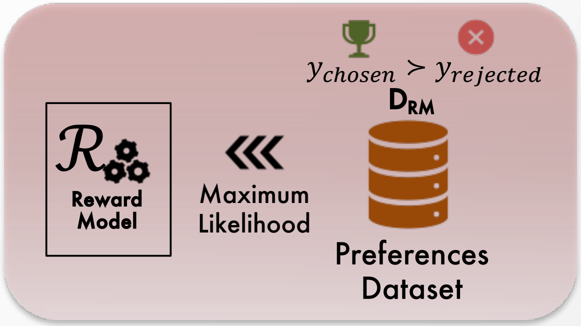
RLHF Optimization – Human Preferences

Step 1 - SFT: Supervised fine-tuning (SFT) using high-quality human demonstrations dataset



Step 2 – Preference Sampling & Reward Learning: Train Reward model (RM) mimicking human preferences

- Start with LM-SFT baseline (from step 1)
- Add randomly initialized head that outputs a scalar (reward) value



$$loss(\theta) := \mathbb{E}_{(x, y_w, y_l) \sim D_{RM}} [\log(\sigma(r_\theta(x, y_w) - r_\theta(x, y_l)))]$$

How?

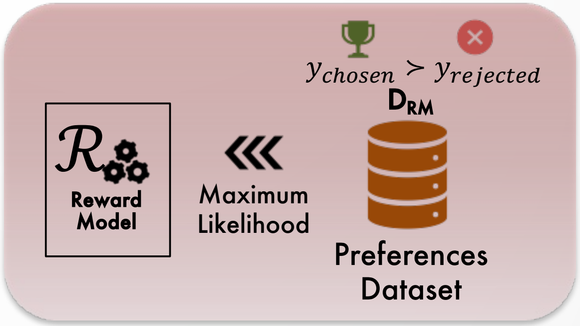
Where $r_\theta(x, y_i)$ is the scalar output of RM_θ for prompt x and completion y_i

[1] Stiennon et al. "Learning to summarize with human feedback." (2020). link
[2] **InstructGPT**: Ouyang, Long, et al. "Training language models to follow instructions with human feedback. (2022)." link

UNIVERSITY OF TORONTO

RLHF Optimization – Human Preferences

Step 2 – Preference Sampling & Reward Learning



Preference Sampling :

- $\mathbf{D_{RM}} = \left\{ \boldsymbol{x}^{(i)}, \boldsymbol{y}_{\boldsymbol{w}}^{(i)}, \boldsymbol{y}_{\boldsymbol{l}}^{(i)} \right\}^{\boldsymbol{N}}$

- SFT model is prompted with prompts $\boldsymbol{x}$ to produce pairs of completions (y_1, y_2) :

$$(y_1, y_2) \sim \pi^{SFT}(y \mid x)$$

- Human labelers determines a winning choice: $y_w \succ y_l \mid x$
  - Assumption: underlying latent (human) reward model: $r^*(y, x)$

UNIVERSITY OF TORONTO

RLHF Optimization – Human Preferences

- Bradley Terry[1] model stipulates human preferences distribution p^* as:

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

- Assuming D_{RM} sampled from p^* we can parameterize a reward model: $r_\theta(x, y)$ and estimate the parameters via maximum likelihood by framing the problem as a binary classification with NLL loss:

$$\mathcal{L}_{\mathcal{R}}(r_\theta, D_{RM}) = -\mathbb{E}_{(x, y_w, y_l) \sim D_{RM}}\left[\log(\sigma(r_\theta(x, y_w) - r_\theta(x, y_l)))\right]$$

where $r_\theta(x, y_i)$ is the scalar output of RM_θ for prompt x and completion y_i and σ is the logistic function

- To ensure a reward function with lower variance, normalize the rewards

$$\mathbb{E}_{(x, y) \sim D}\left[r_\theta(x, y)\right] = 0 \text{ for all x}$$

[1] Bradley, A, and Milton Terry. "Rank analysis of incomplete block designs: I. The method of paired comparisons." *Biometrika* 39.3/4 (1952)

RLHF Optimization – Human Preferences

Step 3 – RL Optimization: Optimize the LM from step 1 with RM (step 2) using RL

- For a completion (or, response) $\hat{y}$ to prompt $\boldsymbol{x}$, we want to update LM-policy $\boldsymbol{\pi^{LM}}$ parameters $\boldsymbol{\phi}$ to maximize:

$$\mathbb{E}_{\hat{y}\sim\pi_{\phi}(\mathrm{x})}[\mathcal{R}_{\theta}(x,\hat{y})]$$

- SGD updates (like below) does not work because our reward function $\mathcal{R}_{\theta}(.)$ is **non-differentiable** w.r.t. our model parameters $\boldsymbol{\pi_{\phi}^{LM}}$

$$\phi_{t+1} \longleftarrow \phi_t + \alpha\nabla_{\phi_t}\mathbb{E}_{\hat{y}\sim\pi_{\phi_t}(\mathrm{x})}[\mathcal{R}_{\theta}(x,\hat{y})]$$

- Thus, we resort to policy-gradient methods in RL like REINFORCE[1], PPO[2] to estimate and optimize this objective

[1] Schulman, John, et al. "Proximal policy optimization algorithms." (2017) link
[2] REINFORCE: Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning. (1992). link.

Optimization: REINFORCE - I

- Drilling down on the REINFORCE (highly simplified) mechanism

$$\nabla_\phi \mathbb{E}_{\hat{y}\sim p_\phi(y)}[\mathcal{R}(\hat{y})] = \underbrace{\nabla_\phi \sum_y \mathcal{R}(y)\, p_\phi(y)}_{\text{defn. of Expectation}} = \underbrace{\sum_y \mathcal{R}(y) \boldsymbol{\nabla_\phi p_\phi(y)}}_{\text{linearity of gradient}}$$

- Reformulate using the **log-derivative trick**

$$\nabla_\phi \log\left(p_\phi(y)\right) = \frac{\nabla_\phi p_\phi(y)}{p_\phi(y)} \Longrightarrow \boldsymbol{\nabla_\phi p_\phi(y)} = p_\phi(y)\, \nabla_\phi \log\left(p_\phi(y)\right)$$

- Plug back into the first equation:

$$\sum_y \mathcal{R}(y)\boldsymbol{\nabla_\phi p_\phi(y)} = \boxed{\sum_y p_\phi(y)} \mathcal{R}(y) \nabla_\phi \log\left(p_\phi(y)\right)$$

$$= \boxed{\mathbb{E}_{\hat{y}\sim p_\phi(y)}} \left[\mathcal{R}(\hat{y}) \nabla_\phi \log\left(p_\phi(\hat{\boldsymbol{y}})\right)\right]$$

[1] REINFORCE: Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning. (1992). link.

Optimization: REINFORCE - II

- We can approximate the objective using Monte Carlo sampling with the gradient pushed inside the expectation operator

$$\nabla_{\phi} \mathbb{E}_{\hat{y}\sim p_{\phi}(\mathrm{y})}[\mathcal{R}(\hat{y})] = \mathbb{E}_{\hat{y}\sim p_{\phi}(\mathrm{y})}\left[\mathcal{R}(\hat{y})\nabla_{\boldsymbol{\phi}} \log\left(p_{\phi}(\hat{\boldsymbol{y}})\right)\right]$$

$$\cong \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \mathcal{R}(\hat{y}_i)\, \nabla_{\boldsymbol{\phi}} \log\left(p_{\phi}(\hat{\boldsymbol{y}}_{\boldsymbol{i}})\right)$$

- Now, we can **update** our **objective** using completion samples y as:

$$\phi_{t+1} \longleftarrow \phi_t + \alpha\, \frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \mathcal{R}(y_i)\, \nabla_{\boldsymbol{\phi}_{\mathbf{t}}} \log\left(p_{\phi_t}(y_i)\right)$$

Intuition

If $\mathcal{R}(y_i)$ is positive then take steps to update weight parameters ϕ to maximize $p_{\phi}(y_i)$. If $\mathcal{R}(y_i)$ is negative then update weight parameters ϕ to minimize $p_{\phi}(y_i)$

[1] REINFORCE: Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning. (1992). link.

RLHF – Step 3 RL Optimization

- **Goal**: update parameters ϕ of our LM π^{SFT} from step 1 with objective:

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

where β controls KL-divergence operator to regulate the deviation of trained policy from a base reference policy, usually π^{SFT}

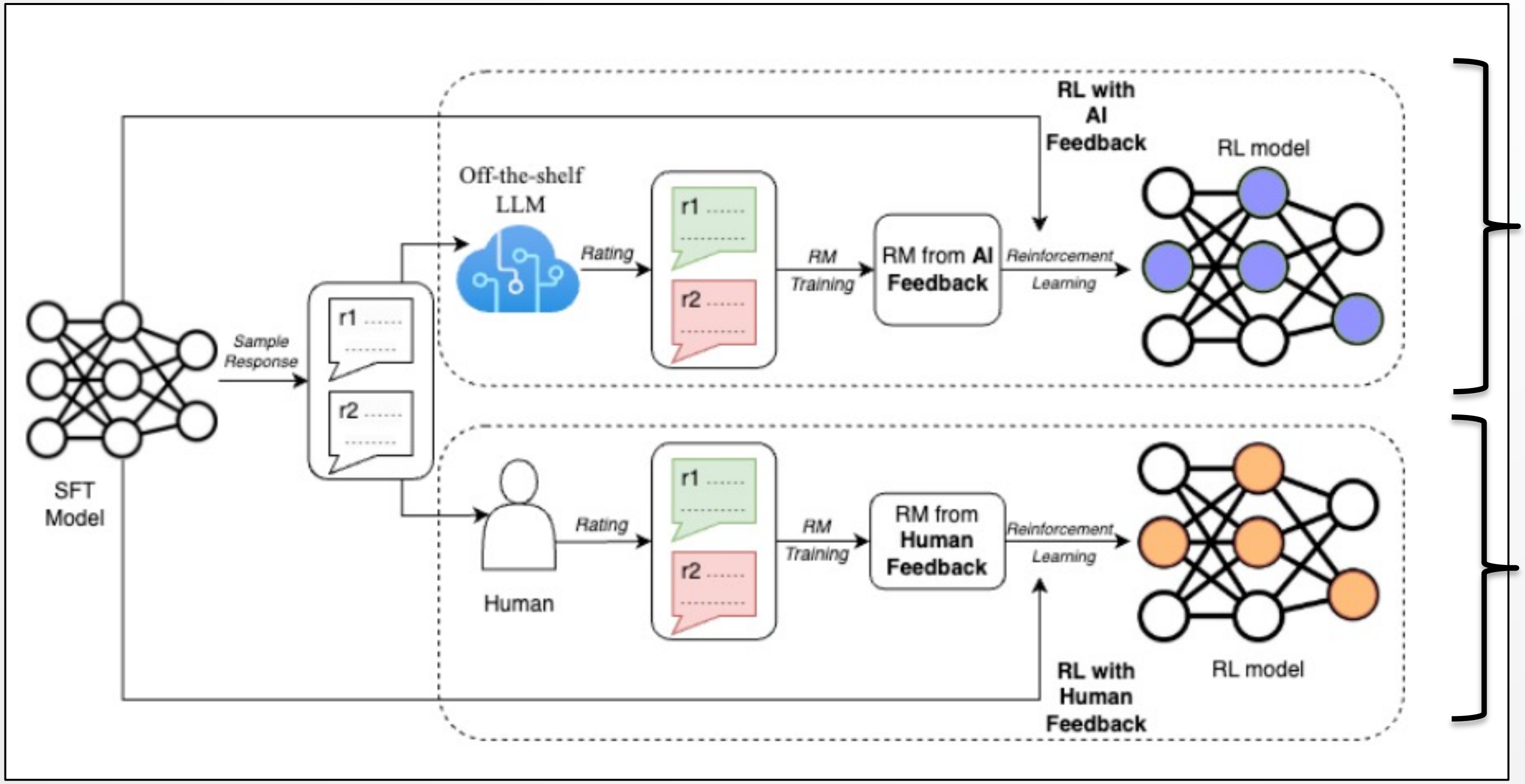
- In practice, π_ϕ is also initialized with π^{SFT}
- Due to discrete nature of language generation, this objective is non-differentiable, and typically optimized with RL
- Specifically, construct this reward function and maximize using PPO:

$$r(x,y) = r_\theta(x,y) - \beta(\log\left(\pi_\phi(y \mid x)\right) - \log(\pi_{ref}(y \mid x))$$

[1] REINFORCE: Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning. (1992). link.

Variants – RLAIF

- Reinforcement Learning from AI Feedback (RLAIF):
  - Train RM using (AI) Feedback from other off-the-shelf LLMs
  - Then, train LM-SFT using this RM as usual in RLHF



RLAIF (top)

RLHF (bottom)

[1] Lee et al. "RLAIF: Scaling Reinforcement Learning from Human Feedback with AI Feedback". (2023) link

PROMPTING LLMS

Prompt Engineering

- Now that we have seen LM pretraining, fine-tuning (SFT) and instruction fine-tuning (RLHF), let's examine how **prompting** works in modern LLMs in practice

- The ~~science~~ **art** of LLM prompting

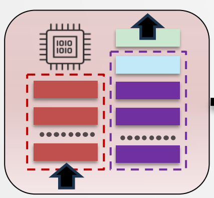
- Prompt design is imperative for obtaining good results from an LLM / foundation model
- From *In-context Learning* (ICL) paradigm we saw gradient free approaches like:
  - *Zero-shot*: Asking LLM to perform task with no previous example
  - *Few-shot*: Providing examples as context to the LLM before giving the task

Prompt Engineering

Zero-Shot

Prompt

Q: What is the capital of France?



LLM

A: "Paris"

Lower token count. Allows larger window for context

Few-Shot

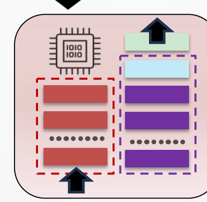
Prompt

Q: What is the capital of Spain?
A: {'answer': 'Madrid'}

Q: What is the capital of Italy?
A: {'answer': 'Rome'}

Q: What is the capital of France?

LLM



A: {'answer': 'Paris'}

Better **alignment** (e.g., format) of responses. Better **accuracy** on complex questions

UNIVERSITY OF TORONTO

Prompt Engineering

Prompt Engineering

- **Techniques**
  - ICL: zero-shot, few-shot
  - Chain-of-thought (CoT) reasoning
  - System prompting …
- **Method:** Prompt templates
- **Training data**:
  - Single-digit completion examples
- **Advantages**:
  - Minimal, simple input samples
  - Online (inference time) gradient free

Full-parameter Fine-tuning

- **Techniques**
  - SFT, RLHF, RLAIF, DPO …
- **Method:** Tune LLM weights
- **Training data**:
  - Thousands of samples & complex use cases
- **Advantages**:
  - Compatibility: traditional approach
  - Robust, better accuracy in challenging domains

Prompt Engineering - Basics

- Typical LLM I/O structure during inference:
  - Discourse, conversations, messages: array of structured message objects to send to the LLM. Provides context or history from which to continue
  - Roles:
    - system: provide core instruction to the LLM
    - user: 'human' (could be another 'AI' chatbot too)
    - Assistant: role of the LLM, to generate a response

POST https://api.openai.com/v1/chat/completions


```
{
    "model": "gpt-4",
    "messages": [
        {
            "role": "system",
            "content": "You are a stock market analyst who predicts the market movement from today's news and world events."
        },
        {
            "role": "assistant",
            "content": "Examine the given market information and news headlines data on 2010-01-08 to forecast whether the $SPY index
                will rise, fall, or remain unchanged. If you think the movement will be less than 0.05%, then return 'Neutral'. Respond
                with Rise, Fall, or Neutral and your reasoning in a new paragraph"
        },
        {
            "role": "user",
            "content": "Context:open, close, high, low, volume, adjusted_close, pct_change, macd, boll_ub, boll_lb, rsi_30, cci_30,
                dx_30, close_30_sma, close_60_sma, vix\n113.8899993896484, 114.5699996948242, 114.620002746582, 113.6600036621094,
                126402800.0, 88.44039916992188, 0.003327762897745501, 1.031328597828875, 114.6771377043121, 109.6268623872406, 61.
                46058322950015, 180.9930812282409, 19.71815559823337, 111.611333211263, 109.9924997965495, 18.1299991607666 \n e.u.
```


UNIVERSITY OF TORONTO

Prompt Engineering - Basics

- Typical LLM hyperparameters during inference:

  - **top_p**: when decoding text, samples from the top p percentage of most likely tokens. In other words, curtail list of generated tokens beyond ‘p’
  - **top_k**: same idea as ‘top_p’ but for ‘k’ most likely tokens (instead of percentage)
  - **repetition_penalty**: parameter controlling how to penalize the generation of the same text token
  - **temperature**: randomness of choosing a token (from 'p'). '0' means least random
  - **max_seq_len**: the size of input context window, usually depends on the LLM.

```
    ],
    "temperature": 0.05,
    "max_tokens": 256,
    "top_p": 1,
    "frequency_penalty": 0,
    "presence_penalty": 0
}
```

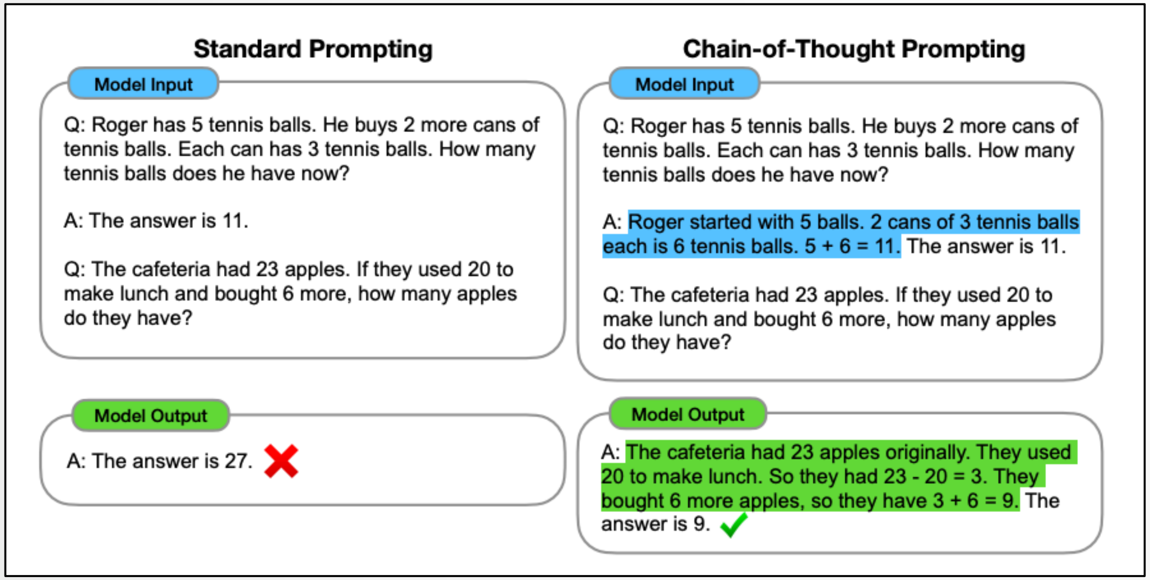

UNIVERSITY OF TORONTO

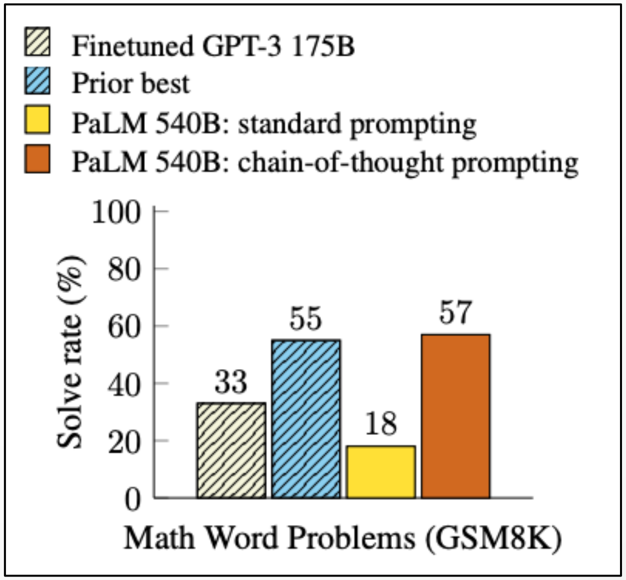
Prompting Techniques

- Active research area with myriad techniques
- Simple stylization changes or decomposition of instructions can change the generated response
- Examples:
  - **Detailed, explicit instructions** better than open-ended prompts:
    - Stylization:
      - *I am a CS student using LLMs for solving assignment*
      - *Give your answer like explaining the topic to a 5-year old*
  - Explicit (step-by-step) instructions:
    - *Use bullet points, only use academic papers, return answer in python code* etc.

Prompting Techniques – Chain of Thought

- Chain of Thought (CoT) prompting[1]
  - Decomposing instruction into series of intermediate reasoning steps





[1] Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. (2022) link.

PRACTICAL TIDBITS & EVALUATION

Evaluation: LLMs Benchmarks

- Language foundation **models** and their typical evaluation **tasks**:

Year	Model Name	Model Arch.	Oriented Tasks	Parameters	Pre-training Method	Pre-training Datasets	Testing Datasets
2018	BERT [83]	Encoder-Only	Text-CLS, Token-CLS, Fill-Mask, QA, Translation, etc.	110-340M	Self-Supervised	Bookscorpus, Enlish Wikipedia	GLUE, SquAD v1.1/2.0, SWAG, IMDb
2019	DistilBERT [314]	Encoder-Only	Same as BERT	66M	Self-Supervised, Distillation	Same as BERT	GLUE, SquAD, IMDb
2019	RoBERTa [238]	Encoder-Only	Same as BERT	125-355M	Self-Supervised	Boookcorpus, CC-news, Openwebtext, Stories	GLUE, SQuAD, RACE
2019	Sentence-BERT [306]	Encoder-Only	Text Similarity	110M	Only Fine-tuning	SNLI, Multi-Genre NLI	STSb
2019	BART [197]	Encoder-Decoder	Same as BERT	140-400M	Self-Supervised	Boookcorpus, CC-news, Openwebtext, Stories	SQuAD, MNLI, ELI5, XSum, ConvAI2, CNN/DM
2019	T5 [300]	Encoder-Decoder	Same as BERT	60M-11B	Self-Supervised	Colossal Clean Crawled Corpus	GLUE, CNNDM, SQuAD, SGLUE, EnDe, EnFr, EnRO
2018	GPT-1 [298]	Decoder-Only	Same as BERT	117M	Self-Supervised	BooksCorpus, English Wikipedia	SQuAD, SNLI
2019	GPT-2 [299]	Decoder-Only	Same as BERT	1.5B	Self-Supervised	WebText	SQuAD, CoQA, WMT, CNN/Daily Mail
2020	GPT-3 [39]	Decoder-Only	Same as BERT	175B	Unsupervised	Common Crawl, WebText2 Books1/2, Wikipedia	LAMBADA, CBT, SuperGLUE
2021	GLM [90]	Decoder-Only	Same as BERT	110M-130B	Unsupervised	BooksCorpus, English Wikipedia	SuperGLUE
2022	InsturctGPT [277]	Decoder-Only	Same as BERT	175B	Unsupervised RLHF	Common Crawl, WebText2 Books1/2, Wikipedia	LAMBADA, CBT, SuperGLUE
2022	PaLM [67]	Decoder-Only	Same as BERT	54B	Unsupervised	Mixture of 780B Text Source code	English NLP, BIG-bench Reasoning, Code, etc.
2020	wav2vec2 [26]	Encoder-Decoder	Auto Speech Recognition	227-896M	Self-Supervised	LibriSpeech, Unlabeled Audio Data	LibriSpeech, TIMIT, Common Voice
2021	HuBERT [140]	Encoder-Decoder	Auto Speech Recognition	281M-2.8B	Self-Supervised	Libri-Light, LibriSpeech	LibriSpeech, TIMIT
2023	Whisper [297]	Encoder-Decoder	Auto Speech Recognition	39-1150M	Self-Supervised Multi-task Learning	Unkown	LibriSpeech, Multi-lingual dataset
2023	LLaMA [355]	Decoder-Only	Text Generation	7-70B	Self-Supervised RLHF	Common Crawl, C4, Github, Wikipedia, Books, ArXiv, StackExchange	TruthfulQA, ToxiGen, etc.
2023	GPT-4 [273]	Close-Sourced	Text Generation	Close-Sourced			MMLU, HellaSwag, ARC, WinoGrande, HumanEval, DROP, GSM-8K
2023	Claude2						Close-Sourced
2023	PaLM2						

UNIVERSITY OF TORONTO

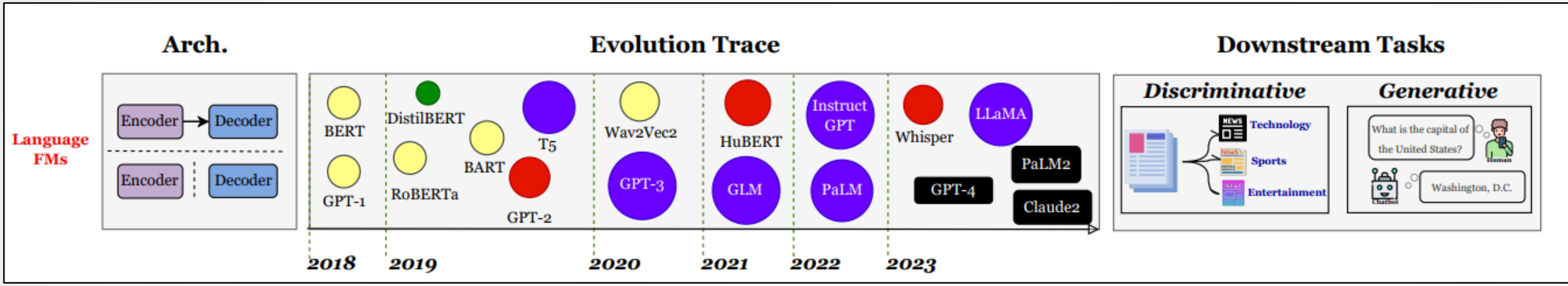
Evaluation: LLMs Benchmarks

- The **GLUE** and **SuperGLUE** benchmarks for evaluating NLP LM tasks circa 2019-21

- LM benchmarking scene has rapidly evolved in conjunction with capabilities since then
- Recent benchmarks include:
  - BIG-Bench[1]: 200+ tasks with dynamic additions of newer tasks
  - MMLU[2] : Evaluates LMs on tasks across 57 diverse knowledge bases
  - HELM[3]:
  - GlobalBench[4]:

[1] Srivastava, A. et. al (2022). "Beyond the imitation game: Quantifying and extrapolating the capabilities of language models". ArXiv 2206.04615.
[2] Hendrycks et al. (2021). Massive Multitask Language Understanding (MMLU). ICLR 2021
[3] Liang, P. et. al (2022). ”HELM: Holistic evaluation of language models". ArXiv preprint, abs/2211.09110.
[4] Song, Y. et. al (2023). "GlobalBench: A Benchmark for Global Progress in Natural Language Processing". ArXiv preprint, abs/2305.14716.

Large models are not easily accessible



Model	Inference memory	Fine-tuning memory*
T5-11B	22 GB	176 GB
LLaMA2-33B	66 GB	396 GB
LLaMA2-70B	140 GB	840 GB

*Default or typical values. Fine-tuning memory depends on the type of optimizers used

Raffel et al., 2020, T5. Zhang et al., 2022, OPT., BigScience, 2022, BLOOM.

UNIVERSITY OF TORONTO

LLMs Memory Footprint

- GPU Memory requirements for LLMs:
  - **Inference** vs. **Training/Fine-tuning** (Tr/ft)
  - For inference, at full precision (float32), each parameter is 32bits or 4 bytes (b).
    - Thus, a 7B param. model requires 28 GB GPU memory $\approx 7B \times 4b = 28\ billion\ bytes$
  - Training/fine-tuning requires more memory as optimizer weights (parameters + gradient) need to be stored.
    - E.g., **Adam**[1] or AdamW (stores the second moment of gradients) requires 16b per trainable param.
    - 16B per trainable param. That's 7B * 16b = 112GB GPU RAM

[1] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." (2014) link.

UNIVERSITY OF TORONTO

[Aside] PEFT

- **PEFT** or **Parameter Efficient Fine-Tuning** is an umbrella term for methodologies (and/or libraries) for efficient adaptation of LLMs to downstream tasks within computation cost/budget

- PEFT methods are typically evaluated on these 5 metrics:
    1. Storage efficiency
    2. Memory efficiency
    3. Computation efficiency,
    4. Accuracy, and
    5. Inference overhead

LLMs: Summary & Conclusion

- Full coverage of all pertinent areas would make a course of its own. We deep-dived into **selected topics** only
- Advents in LLM research has truly put AI in global limelight
- Research in LLM ~~can be exhausting~~ is incredibly fast-paced with global, immediate impact
- Understand the many limitations of LLMs (e.g., hallucination, creative tasks) and the nature of each fast-moving components:
  - Representation learning domain
  - Component improvements: e.g. attention mechanisms, alternate architectures
  - Improving training/fine-tuning methodologies at scale:
    - Parallelism, PEFT: quantization techniques etc.