

Deep Learning lecture 10 Modern Language Models

Yi Wu, IIIS

Spring 2025

Apr-21

Today's Topic

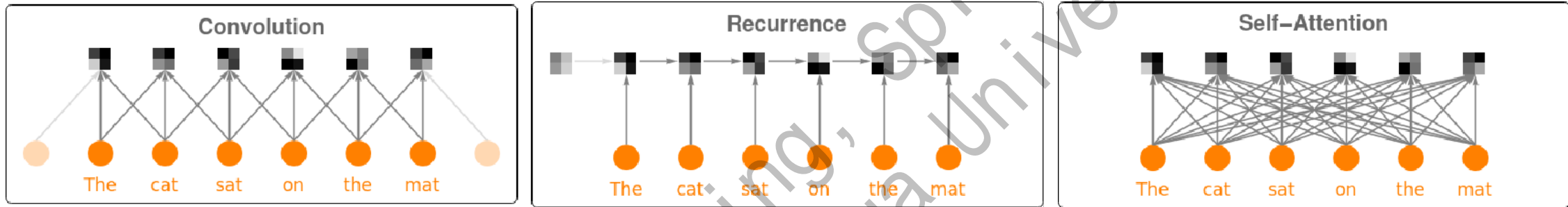
- Pretraining Transformers
- Scaling Law and Large Language Models
- Post-training Language Models and More Applications

Story So Far

- Language model techniques
 - Learning: expensive softmax operator (NCE, H-softmax)
 - Inference: beam search
 - Embedding: contextualized embeddings
- Seq2Seq model
 - Generic architecture for conditioned language modeling
- Attention
 - Learning order/scale-invariant representations
 - Capture distant interactions
 - Transformer: attention is all you need for sequence modeling (since 2017)
 - and even images (since 2021)

Different Sequence Models

- CNN v.s. LSTM v.s. Transformer

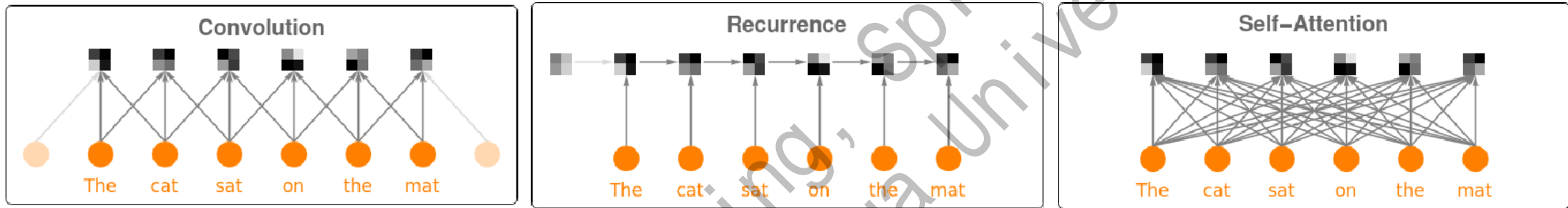


- Embedding methods

- Word2vec: static embedding
- ELMo: pretraining bidirectional LSTMs for contextualized features
- Transformer encoder: powerful bidirectional sequence encoder

Different Sequence Models

- CNN v.s. LSTM v.s. Transformer

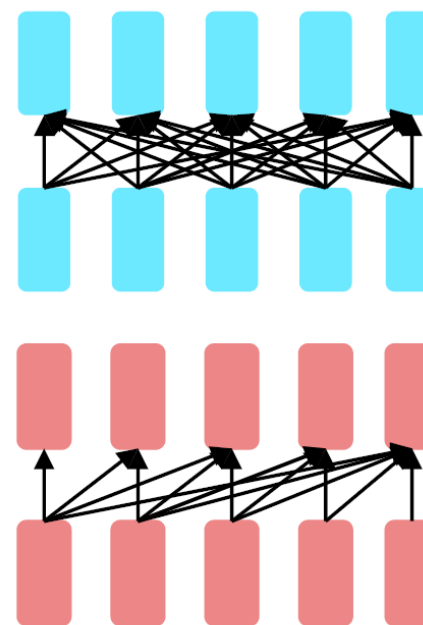


- Embedding methods

- Word2vec: static embedding
- ELMo: **pretraining bidirectional LSTMs** for contextualized features
- Transformer encoder: powerful **bidirectional** sequence encoder
- **Idea: pretraining large transformers!**
 - Pretrained transformers also lead to good representations!
 - ***Foundation of most nowadays NLP applications***

Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- **How to pretrain a transformer on texts?**
 - Pretrain an encoder
 - Bi-directional
 - Pretrain a decoder
 - Auto-regressive
 - Also both encoder and decoder

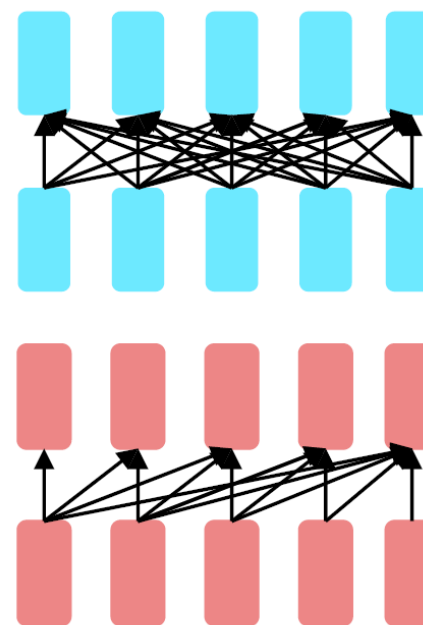


Encoders

Decoders

Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- How to pretrain a transformer on texts?
 - **Pretrain an encoder**
 - **Bi-directional**
 - Pretrain a decoder
 - Auto-regressive
 - Also both encoder and decoder

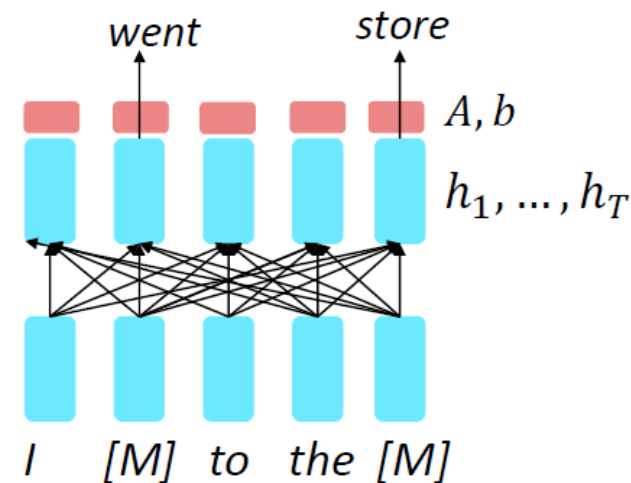


Encoders

Decoders

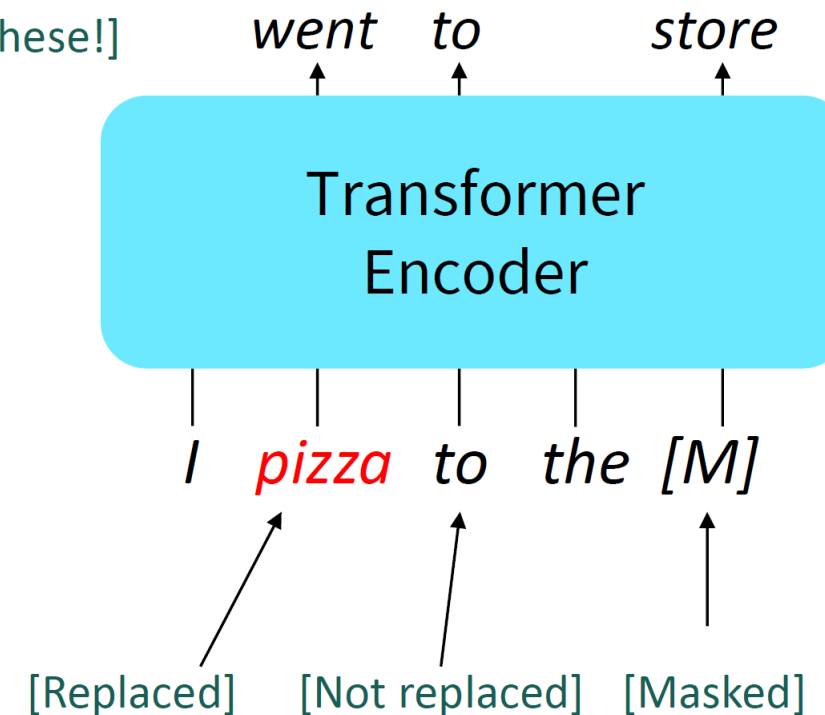
Pretraining Transformer Encoder

- Pretraining a bi-directional encoder
 - We cannot directly adopt language model learning
 - Idea: word prediction given contexts (similar to word2vec)
- Masked Language Model
 - Randomly “masked out” some words
 - Run full transformer encoder
 - Predict the words at masked positions
- Designed for feature extraction
 - More suitable for down-stream tasks



Pretraining Transformer Encoder

- BERT: Pre-training of Deep Bidirectional Transformers
 - Devlin et al, Google, 2018
 - BERT-base: 12 layers, 110M params
 - BERT-large: 24 layers, 340M params
 - Training on 64 TPUs in 4 days
 - Fine-tuning can be done in a single GPU
 - Masked language model
 - Randomly select 15% of word tokens
 - Mask out 80% of the selected tokens
 - Replace 10% of selected words with random tokens
 - For 10% of selected words remain unchanged
 - Predict the selected tokens



Pretraining Transformer Encoder

- BERT: Pre-training of Deep Bidirectional Transformers
 - Devlin et al, Google, 2018
 - BERT-base: 12 layers, 110M params
 - BERT-large: 24 layers, 340M params
 - Training on 64 TPUs in 4 days
 - Fine-tuning can be done in a single GPU
 - Masked language model

[Predict these!]

went *to* *store*

Transformer
Encoder

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	Average
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
BERT _{BASE}	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
BERT _{LARGE}	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

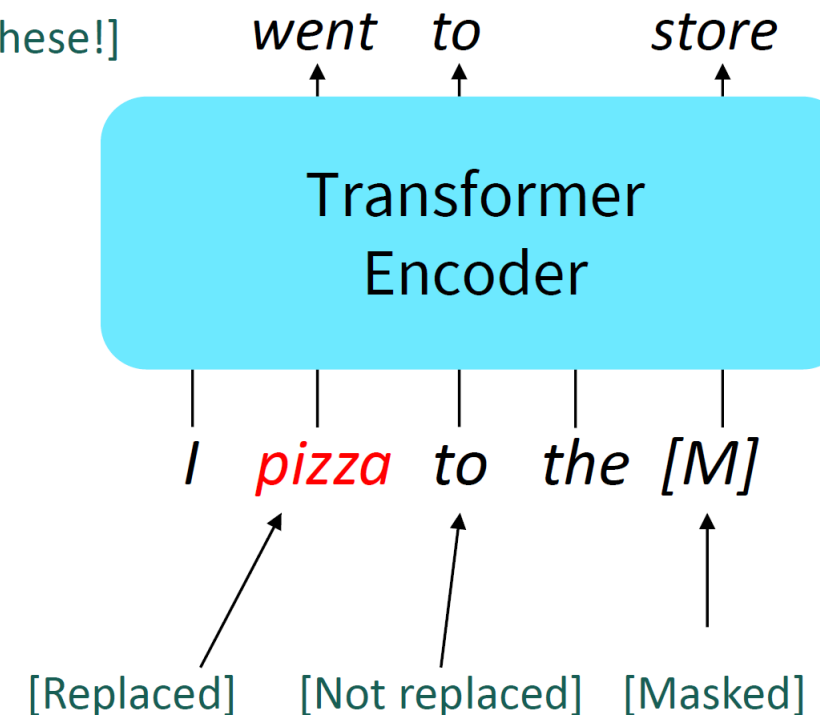
Pretraining Transformer Encoder

- BERT: Pre-training of Deep Bidirectional Transformers (Google 2018)
- RoBERTa: A Robustly Optimized BERT Pretraining Approach
 - Facebook AI, 2019
 - More compute, data and improved objective

Model	data	bsz	steps	SQuAD (v1.1/2.0)	MNLI-m	SST-2
RoBERTa						
with BOOKS + WIKI	16GB	8K	100K	93.6/87.3	89.0	95.3
+ additional data (§3.2)	160GB	8K	100K	94.0/87.7	89.3	95.6
+ pretrain longer	160GB	8K	300K	94.4/88.7	90.0	96.1
+ pretrain even longer	160GB	8K	500K	94.6/89.4	90.2	96.4

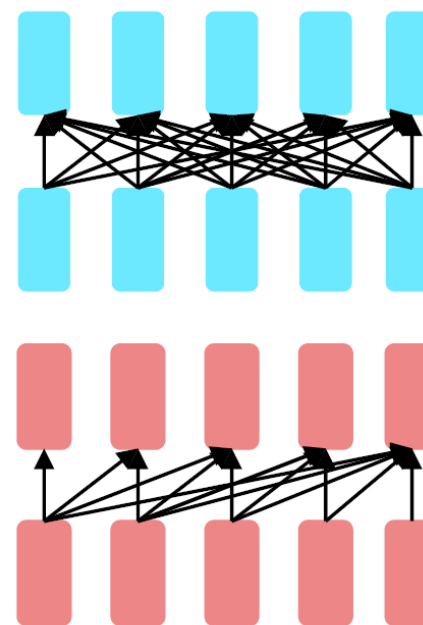
BERT_{LARGE}

with BOOKS + WIKI	13GB	256	1M	90.9/81.8	86.6	93.7
-------------------	------	-----	----	-----------	------	------



Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- How to pretrain a transformer on texts?
 - Pretrain an encoder
 - Bi-directional (e.g., BERT, RoBERTa)
 - Pretrain a decoder
 - Auto-regressive
 - Also both encoder and decoder

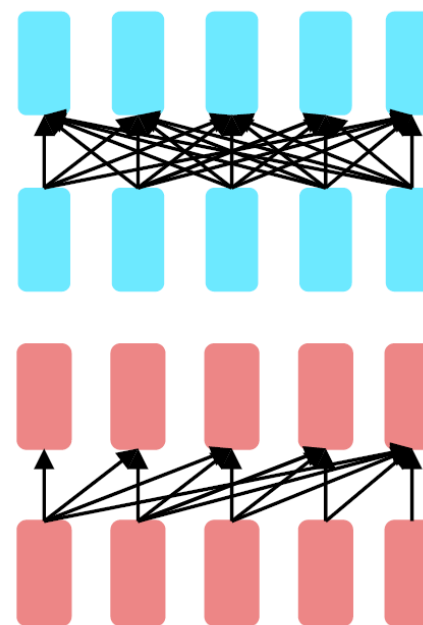


Encoders

Decoders

Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- How to pretrain a transformer on texts?
 - Pretrain an encoder
 - Bi-directional
 - Pretrain a decoder
 - Auto-regressive
 - Also both encoder and decoder

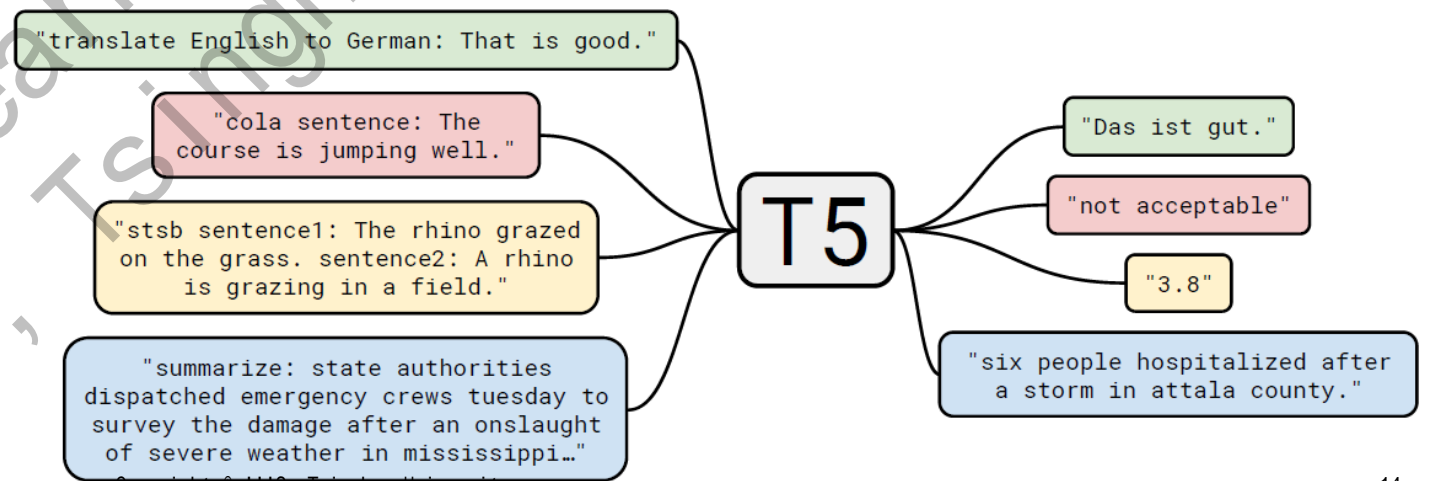
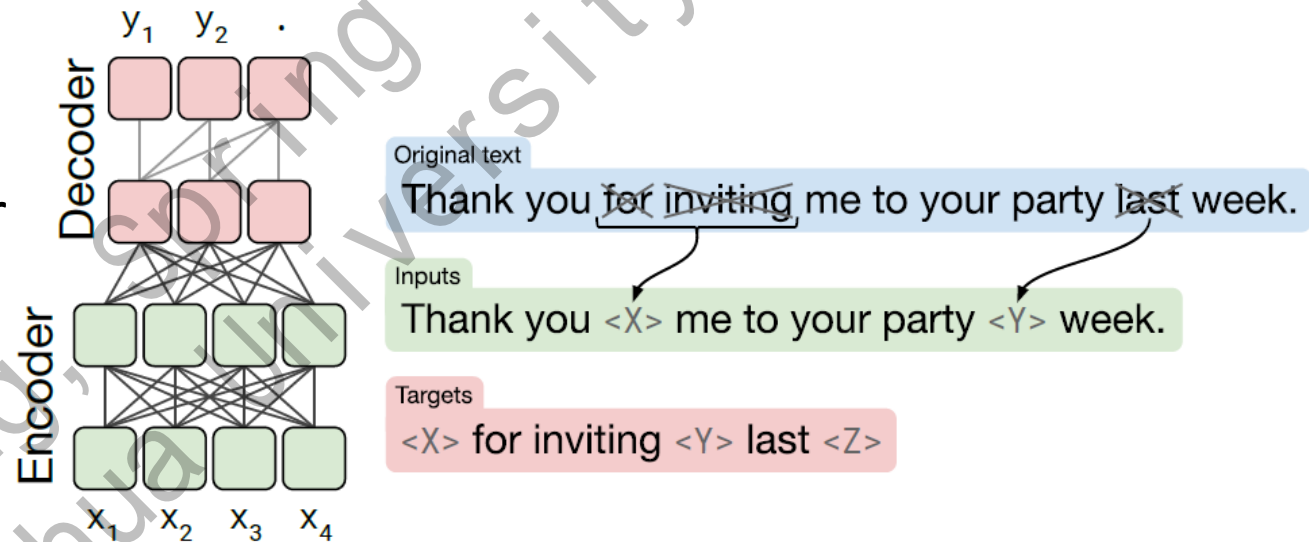


Encoders

Decoders

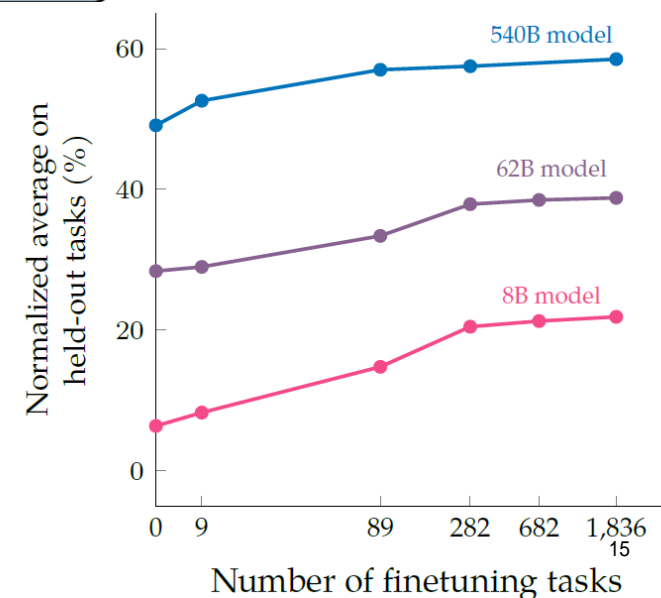
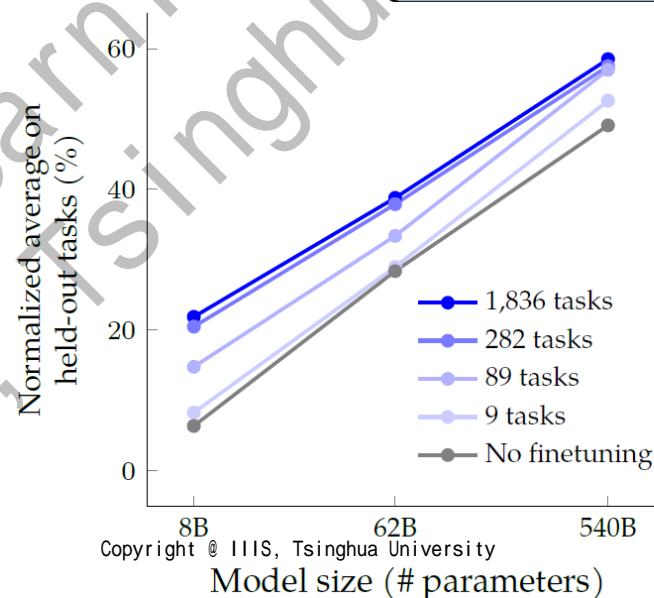
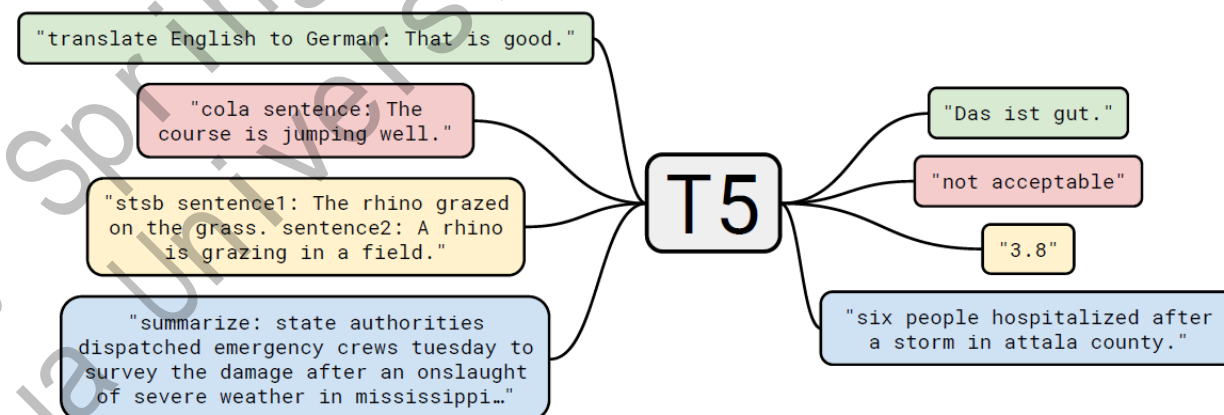
Pretraining a Multi-Task Encoder-Decoder

- T5 (Google, 2019)
 - Text-to-Text Transfer Transformer
 - Multi-task training
 - Generalize to general QA



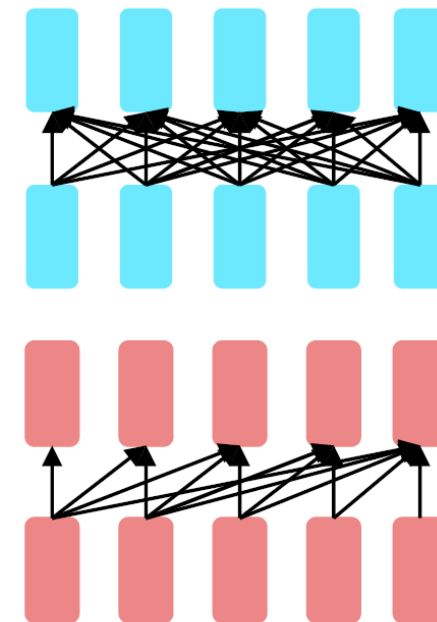
Pretraining a Multi-Task Encoder-Decoder

- T5 (Google, 2019)
 - Text-to-Text Transfer Transformer
 - Multi-task training
 - Generalize to general QA
- FLAN-T5 (Google, 2022)
 - T5 with fine-tuning
 - Large-scale training



Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- How to pretrain a transformer on texts?
 - Pretrain an encoder
 - Bi-directional
 - Pretrain a decoder
 - Auto-regressive
 - Also both encoder and decoder (T5/FLAN-T5)

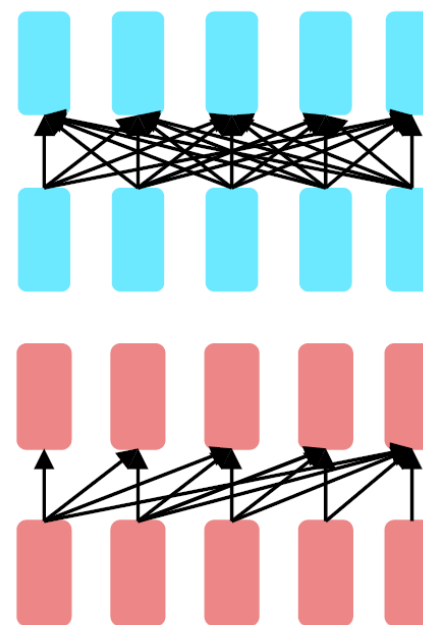


Encoders

Decoders

Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- How to pretrain a transformer on texts?
 - Pretrain an encoder
 - Bi-directional
 - Pretrain a decoder
 - Auto-regressive
 - Also both encoder and decoder

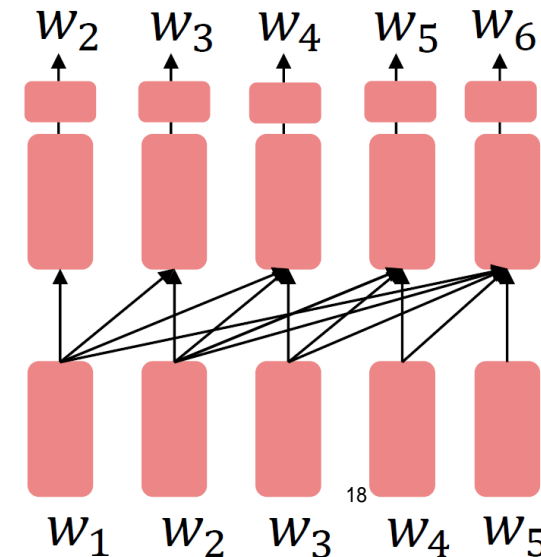


Encoders

Decoders

Pretraining Transformer Decoder

- Decoder Pretraining
 - Just train a language model over the corpus!
 - Great for generative tasks (e.g. text generation) & even beyond
- Generative Pretrained Transformer (GPT, Radford et al, OpenAI 2018)
 - 12-layers transformer, 768-d hidden, 3072-d MLP, BooksCorpus (>7k books)
- GPT-2 (Radford et al, OpenAI 2019.2)
 - 1.5B parameters, 40GB internet texts
- GPT-3 (OpenAI, 2020.5)
 - Language Model are Few-Shot Learners, 175B parameters
- Also ImageGPT (2020.1), ChatGPT (2022.11), GPT-4(2023)



Pretraining Transformer Decoder

- GPT-2 (Alec Radford et al., OpenAI, 2019)
 - 1.5B parameters, 48 layers transformer, trained on 10B tokens

Language Models are Unsupervised Multitask Learners

Alec Radford^{*1} Jeffrey Wu^{*1} Rewon Child¹ David Luan¹ Dario Amodei^{**1} Ilya Sutskever^{**1}

- Zero-shot SOTA performance on a lot of NLP tasks

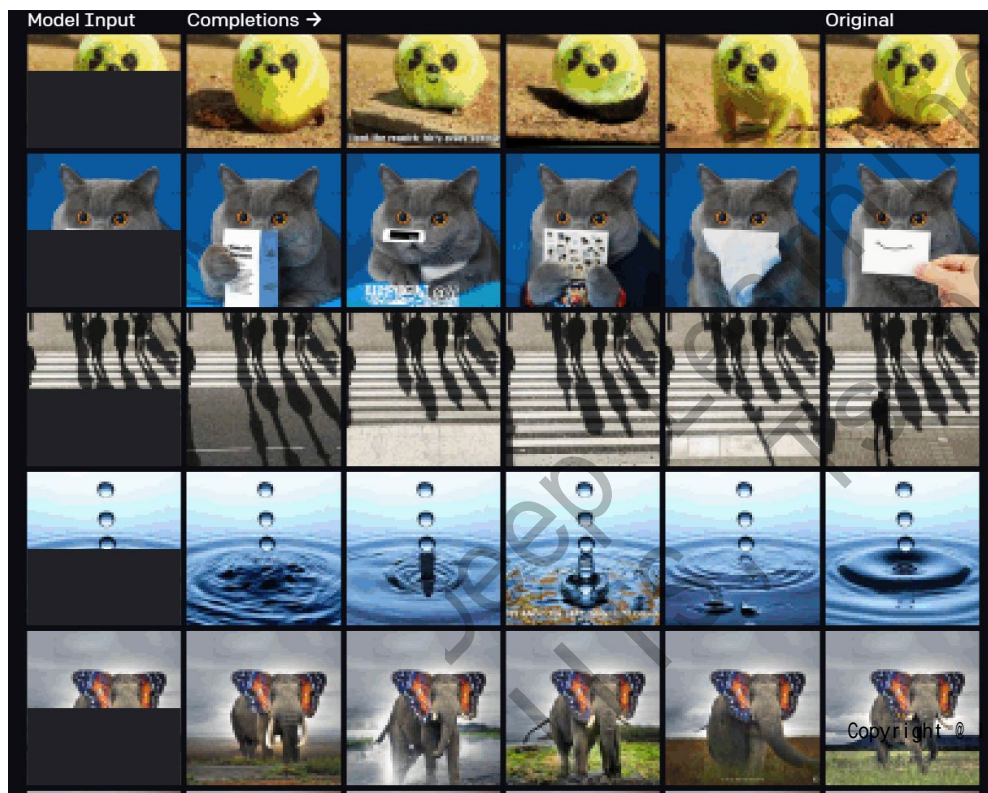
New research direction: prompt learning
[\(<https://arxiv.org/pdf/2107.13586.pdf>\)](https://arxiv.org/pdf/2107.13586.pdf)

- Task descriptions as sequence prefix (prompt), no task specific training

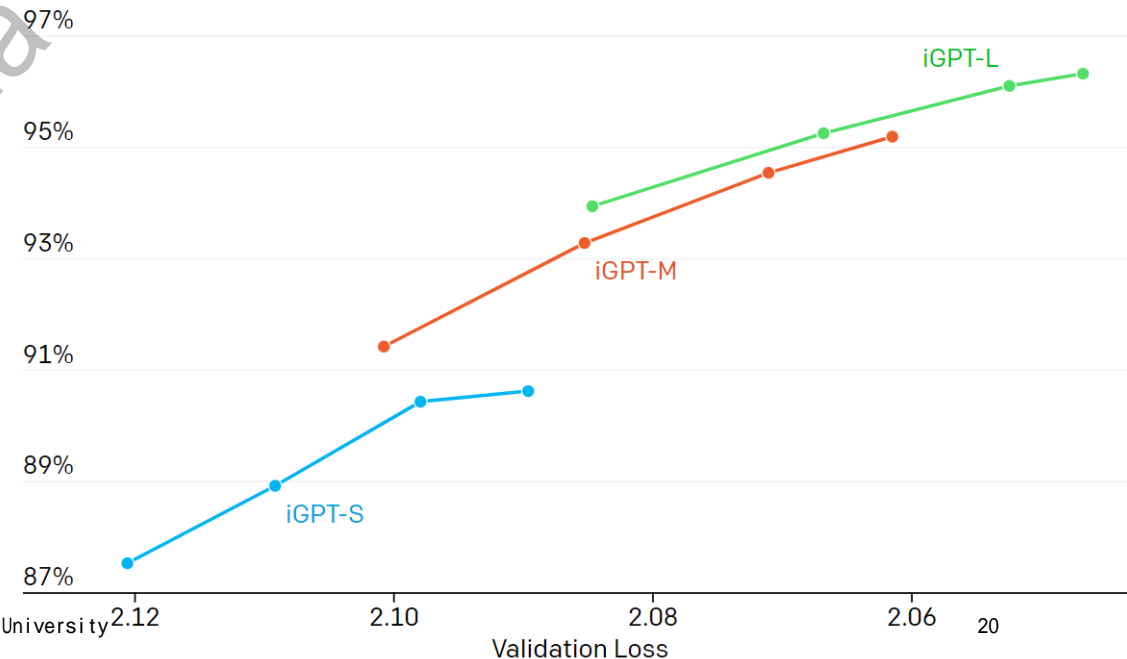
	LAMBADA (PPL)	LAMBADA (ACC)	CBT-CN (ACC)	CBT-NE (ACC)	WikiText2 (PPL)	PTB (PPL)	enwik8 (BPB)	text8 (BPC)	WikiText103 (PPL)	1BW (PPL)
SOTA	99.8	59.23	85.7	82.3	39.14	46.54	0.99	1.08	18.3	21.8
117M	35.13	45.99	87.65	83.4	29.41	65.85	1.16	1.17	37.50	75.20
345M	15.60	55.48	92.35	87.1	22.76	47.33	1.01	1.06	26.37	55.72
762M	10.87	60.12	93.45	88.0	19.93	40.31	0.97	1.02	22.05	44.575
1542M	8.63	63.24	93.30	89.05	18.34	35.76	0.93	0.98	17.48	42.16

Pretraining Transformer Decoder

- Image GPT (OpenAI, ICML 2020)
 - A large transformer-based generative model over image pixels
 - Learned features allow zero-shot classification (linear probing)



CIFAR-10 Linear Probe Accuracy

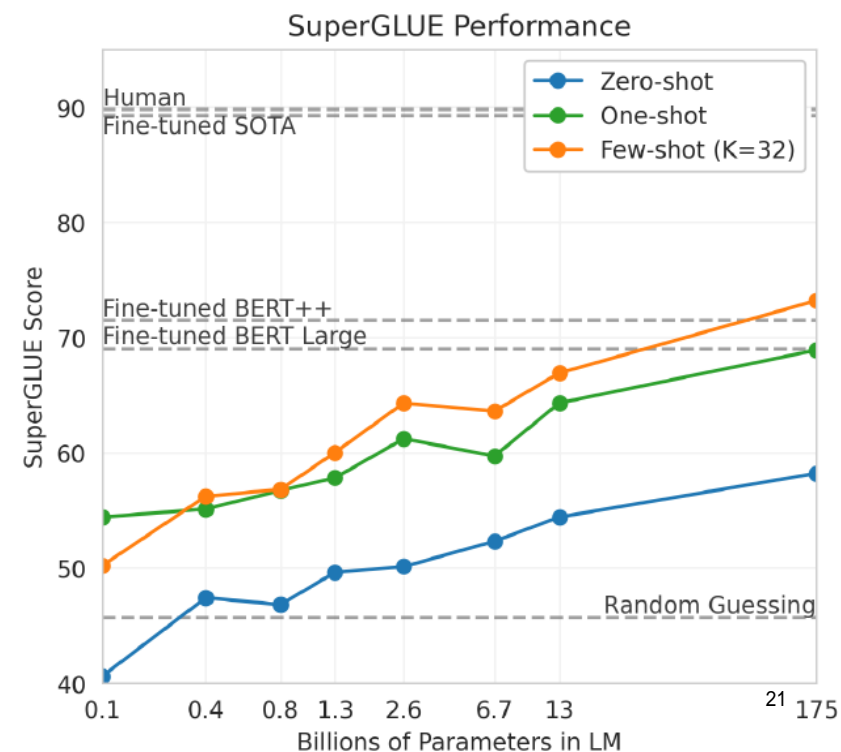


Pretraining Transformer Decoder

- GPT-3: Language models are few-shot learners (OpenAI, NIPS2020)
 - 500B tokens, 175B parameters of transformer
 - Approach SOTA methods on a wide range of NLP tasks without any fine-tuning

Setting	CoQA	DROP	QuAC	SQuADv2	RACE-h	RACE-m
Fine-tuned SOTA	90.7^a	89.1^b	74.4^c	93.0^d	90.0^e	93.1^e
GPT-3 Zero-Shot	81.5	23.6	41.5	59.5	45.5	58.4
GPT-3 One-Shot	84.0	34.3	43.3	65.4	45.9	57.4
GPT-3 Few-Shot	85.0	36.5	44.3	69.8	46.8	58.1

Setting	PIQA	ARC (Easy)	ARC (Challenge)	OpenBookQA
Fine-tuned SOTA	79.4	92.0 [KKS ⁺ 20]	78.5 [KKS ⁺ 20]	87.2 [KKS ⁺ 20]
GPT-3 Zero-Shot	80.5*	68.8	51.4	57.6
GPT-3 One-Shot	80.5*	71.2	53.2	58.8
GPT-3 Few-Shot	82.8*	70.1	51.5	65.4



Pretraining Transformer Decoder

- GPT-3: Language models are few-shot learners (OpenAI, NIPS2020)
 - The concept of ***In-context learning***
 - You may not need to fine-tune the model parameters for domain-specific task

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

1	Translate English to French:	← task description
2	sea otter => loutre de mer	← examples
3	peppermint => menthe poivrée	
4	plush girafe => girafe peluche	
5	cheese =>	← prompt

Code: `px.line(df.query("continent == 'Europe' and country == 'France'"), x='year', y='gdpPercap', color='country', log_y=False, log_x=False)`

Description: Actually, replace GDP with population

Code: `px.line(df.query("continent == 'Europe' and country == 'France'"), x='year', y='pop', color='country', log_y=False, log_x=False)`

Description: Put y-axis on log scale

Code: `px.line(df.query("continent == 'Europe' and country == 'France'"), x='year', y='pop', color='country', log_y=True, log_x=False)`

Pretraining Transformer Decoder

- Fine-tuning v.s. Zero-Shot (prompting) v.s. Few-shot (in-context learning)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.



Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



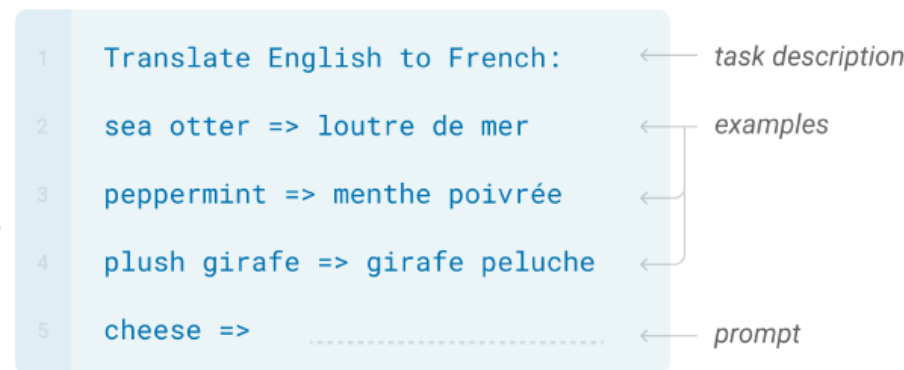
One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



Few-shot

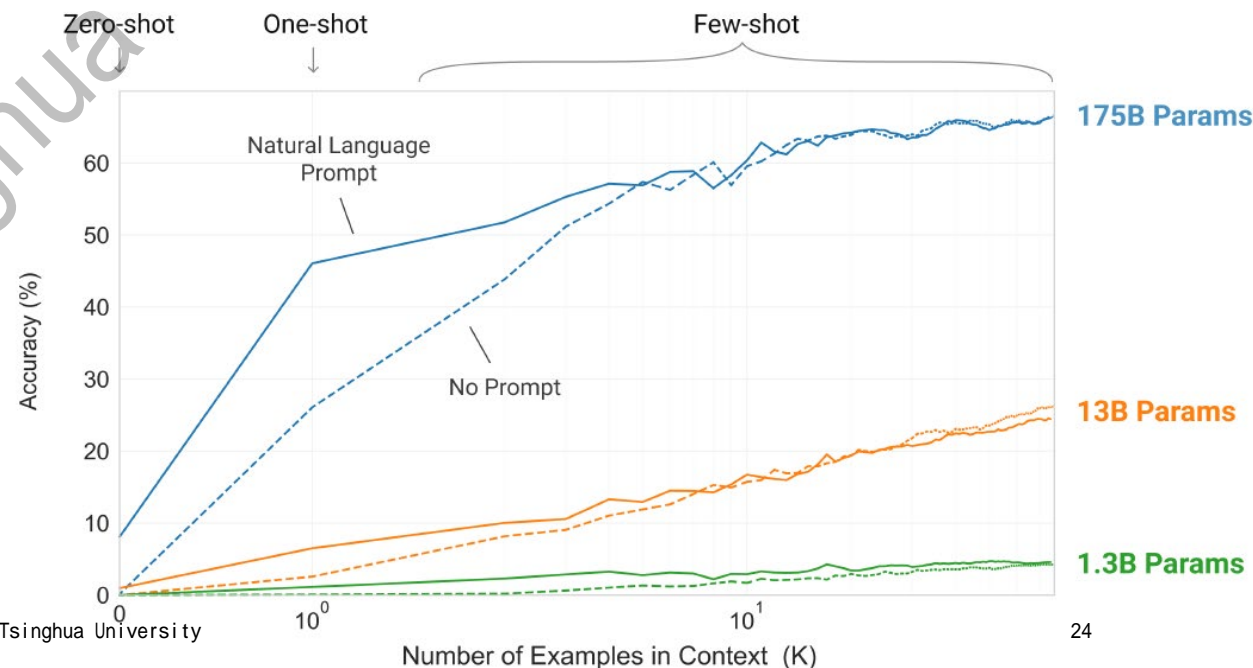
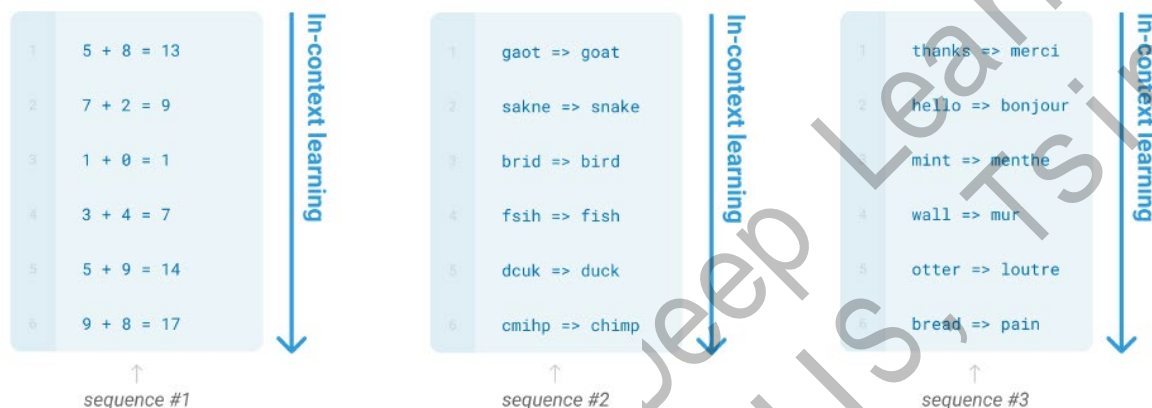
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



Pretraining Transformer Decoder

- In-context learning: the model is trained once, and then put data as part of the input (prompt) to the model.
 - The capability is first observed in GPT-3
 - More in-context examples, better final performance

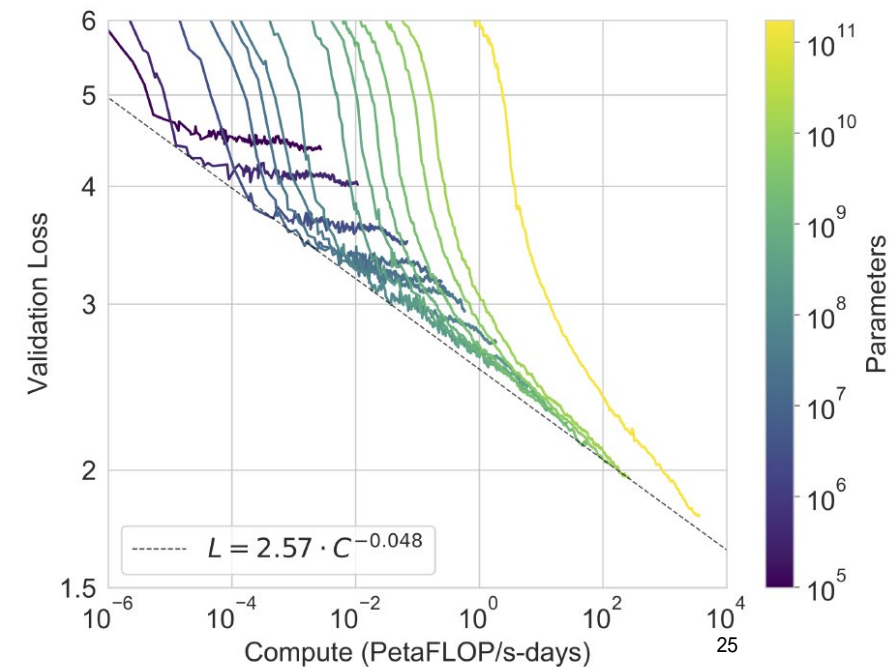
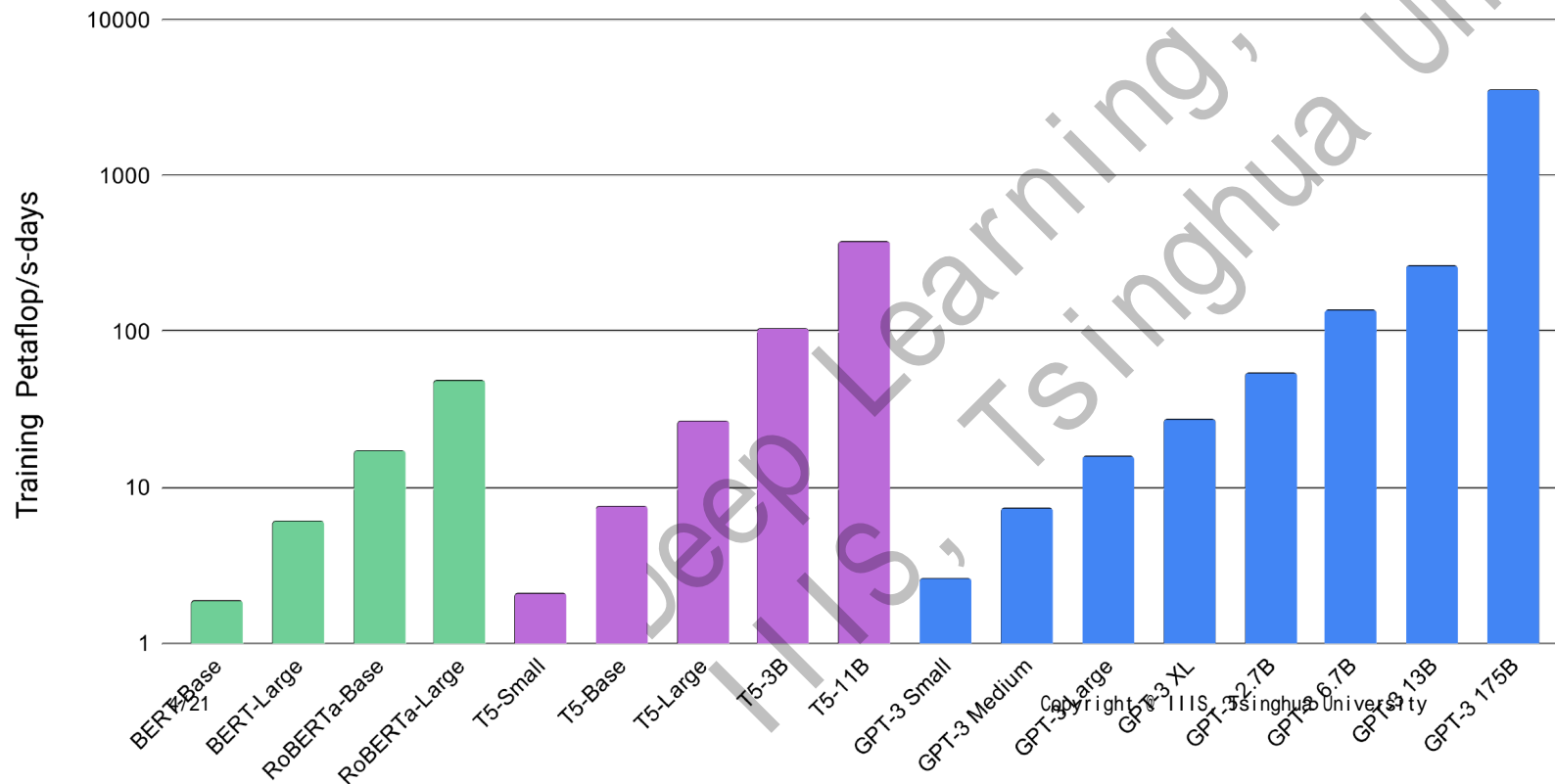
Learning via SGD during unsupervised pre-training



Pretraining Transformer Decoder

- Computation used by GPT-3
 - More training compute, lower validation loss

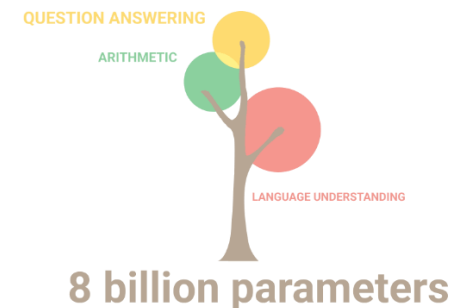
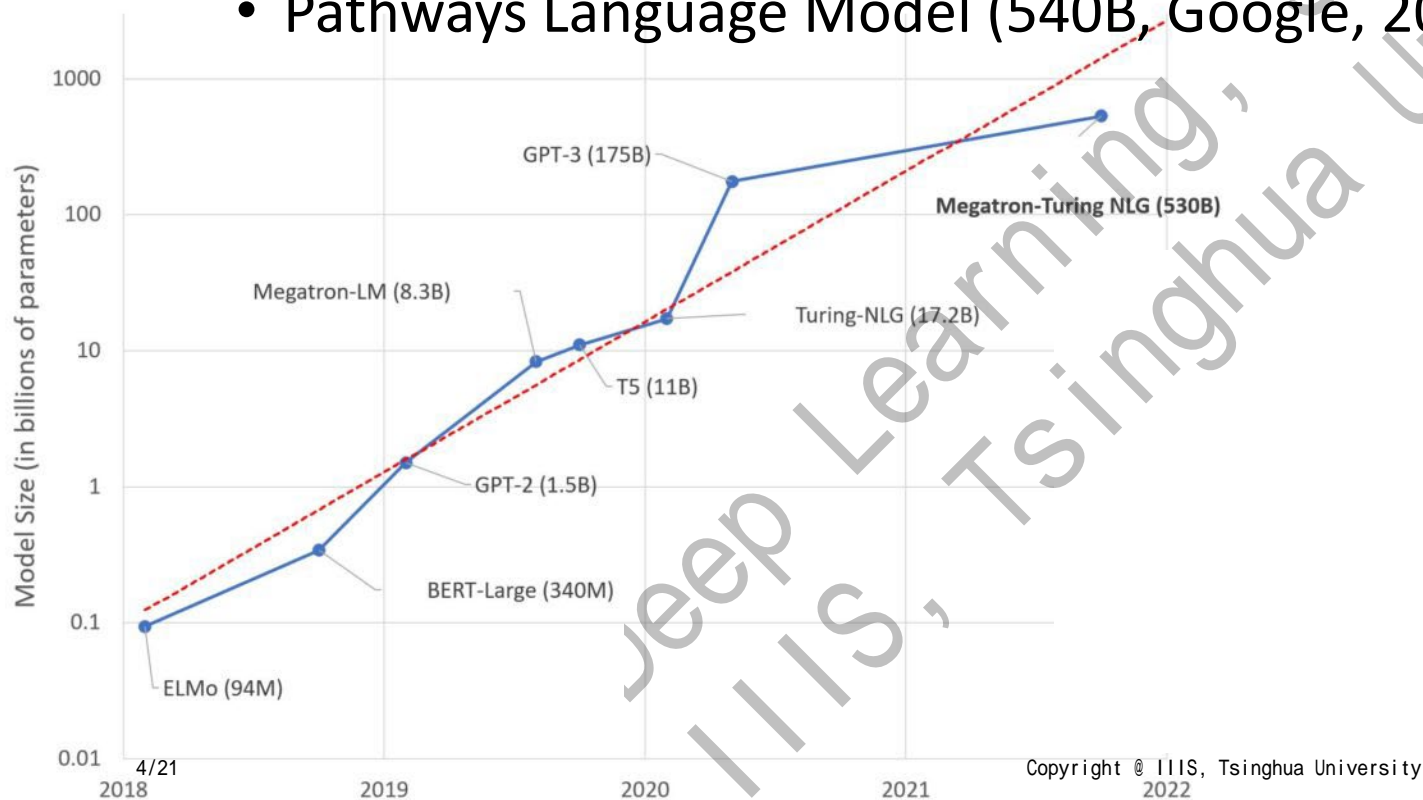
Total Compute Used During Training



Pretraining Transformer Decoder

- A big ongoing race since 2021 to train even larger language models
 - Megatron-Turing NLG (530B, Microsoft, 2021.10)
 - Pathways Language Model (540B, Google, 2022.4)

Why do people
train larger models?



Scaling Law

- A simple rule for predicting LLM performances (OpenAI, 2020)
 - You can perform experiments on small models and extrapolate on larger ones

Scaling Laws for Neural Language Models

Jared Kaplan *

Johns Hopkins University, OpenAI

jaredk@jhu.edu

Sam McCandlish*

OpenAI

sam@openai.com

Tom Henighan

OpenAI

henighan@openai.com

Tom B. Brown

OpenAI

tom@openai.com

Benjamin Chess

OpenAI

bchess@openai.com

Rewon Child

OpenAI

rewon@openai.com

Scott Gray

OpenAI

scott@openai.com

Alec Radford

OpenAI

alec@openai.com

Jeffrey Wu

OpenAI

jeffwu@openai.com

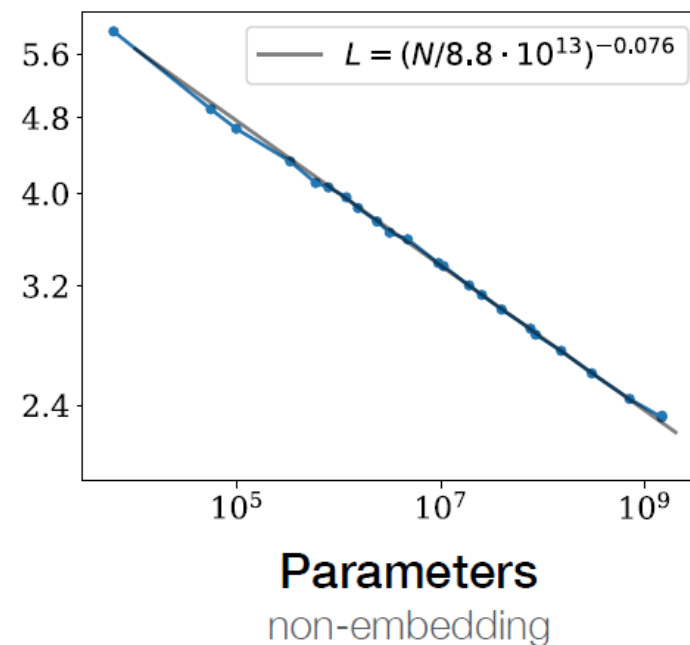
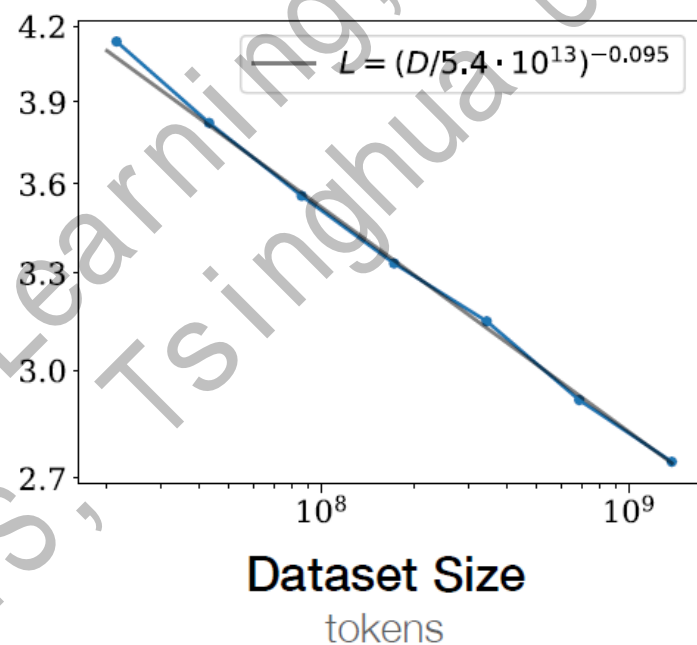
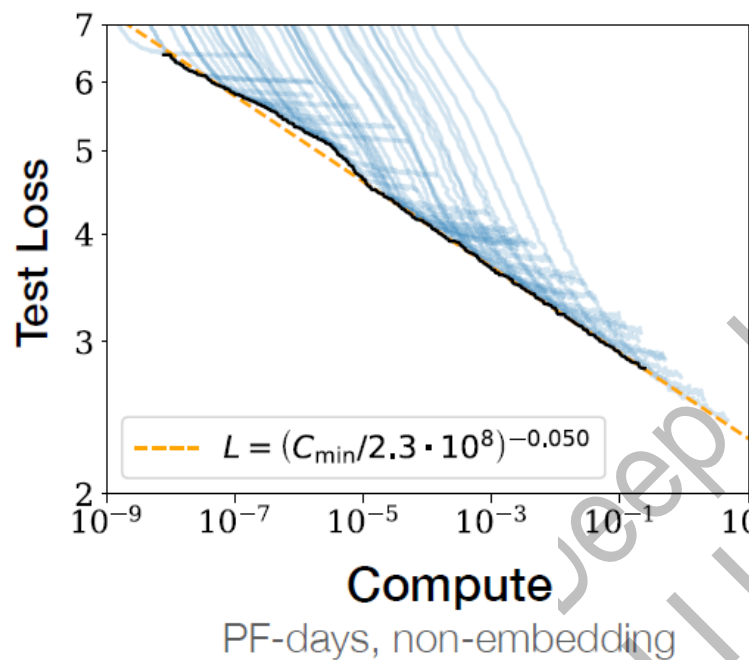
Dario Amodei

OpenAI

damodei@openai.com

Scaling Law

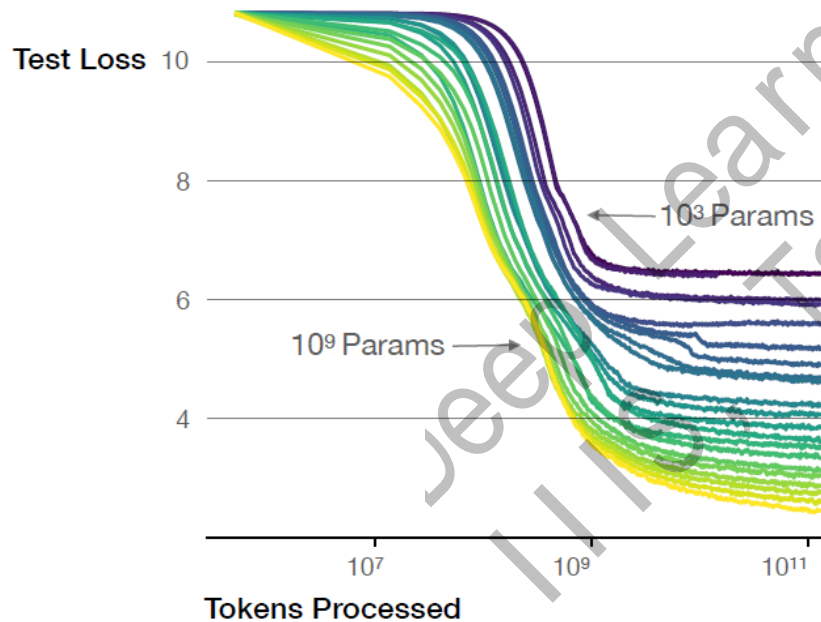
- A simple rule for predicting LLM performances (OpenAI, 2020)
 - Compute, dataset size and parameters are key factors for LLM performances
 - You can fit an power law for these factors



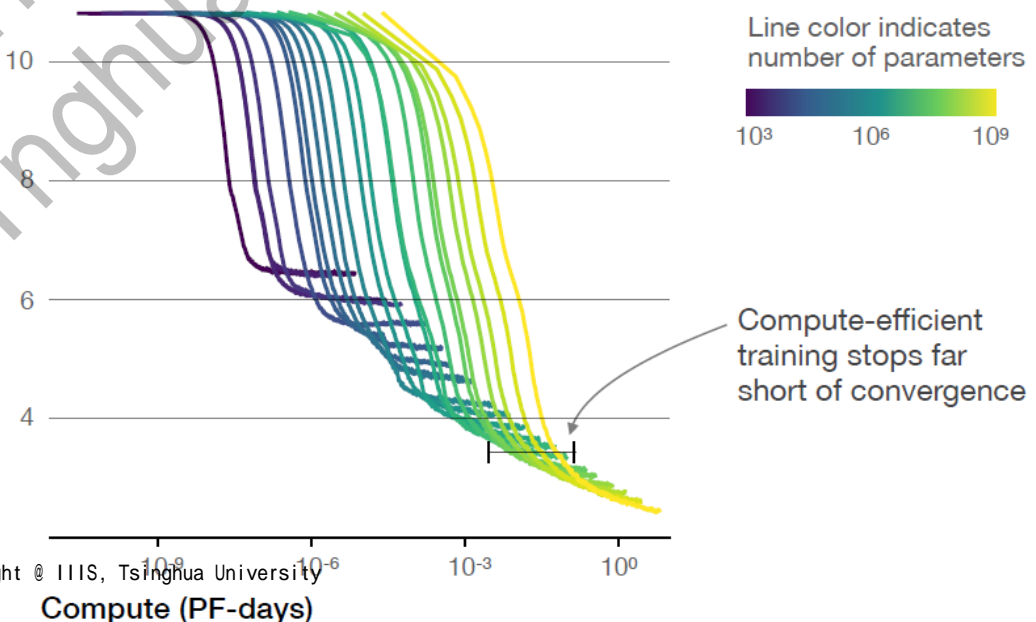
Scaling Law

- A simple rule for predicting LLM performances (OpenAI, 2020)
 - OpenAI suggests that you should train larger models!
 - Claim from the perspective of 2020

Larger models require **fewer samples** to reach the same performance



The optimal model size grows smoothly with the loss target and compute budget



Scaling Law: The Chinchilla Law

- The optimal model size and training tokens given a fixed compute budget (DeepMind, 2022)
 - TL;DR: every doubling the model size, the training tokens should be also doubled (training tokens matter!!!!)



Training Compute-Optimal Large Language Models

Jordan Hoffmann*, Sebastian Borgeaud*, Arthur Mensch*, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals and Laurent Sifre*

*Equal contributions

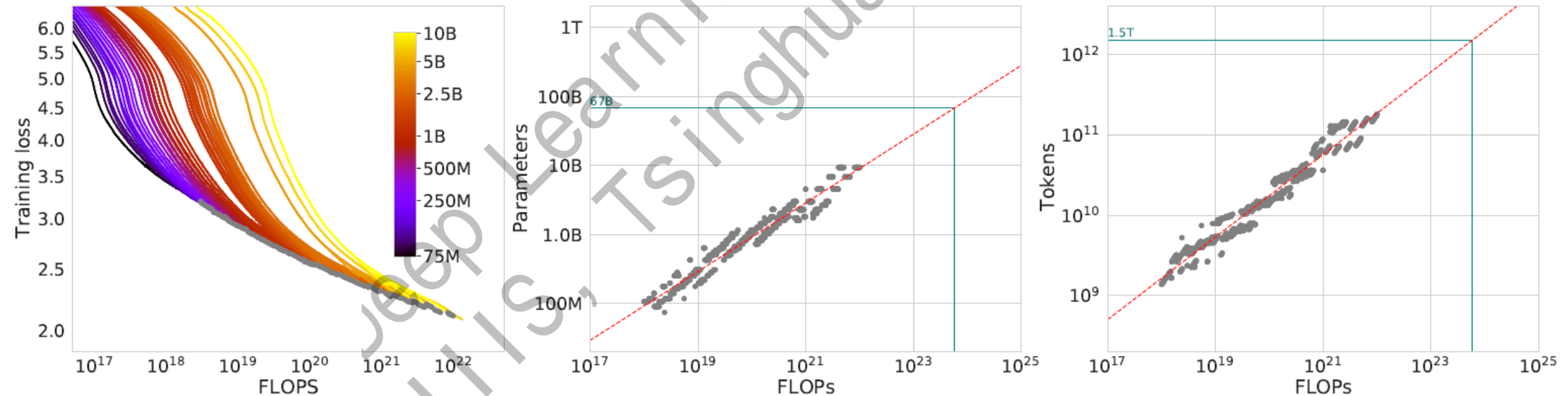
Scaling Law: The Chinchilla Law

- The optimal model size and training tokens given a fixed compute budget (DeepMind, 2022)
 - TL;DR: every doubling the model size, the training tokens should be also doubled (training tokens matter!!!!)
 - You can have a smaller but better model with more data

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
<i>Chinchilla</i>	70 Billion	1.4 Trillion

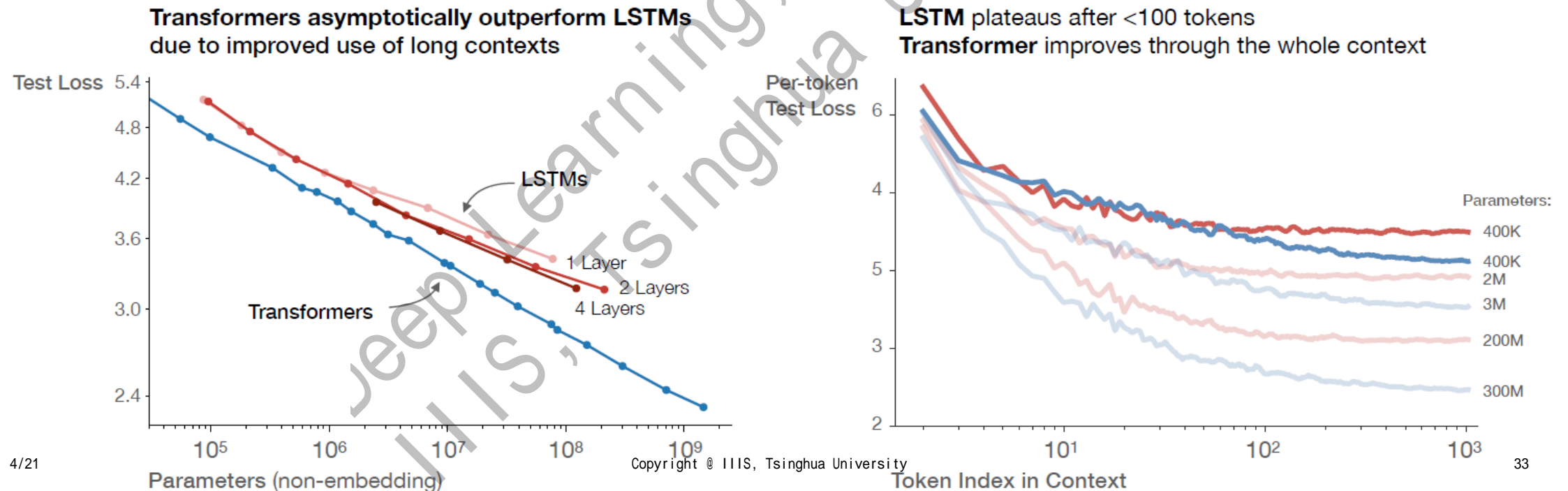
Scaling Law: The Chinchilla Law

- The optimal model size and training tokens given a fixed compute budget (DeepMind, 2022)
 - Left: given different compute/#params, track the loss of different token size
 - Right: fit the best loss w.r.t. tokens/params at each compute

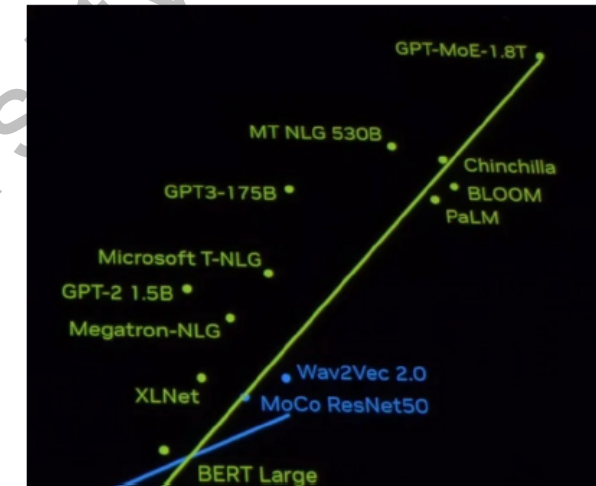


Scaling Law: Architecture Matters

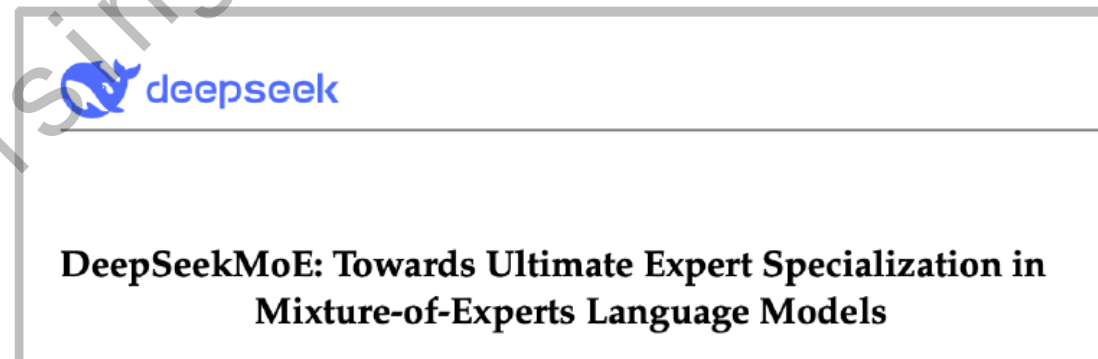
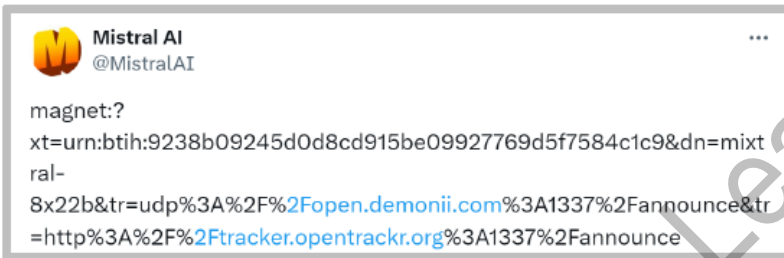
- LSTM v.s. Transformers (OpenAI, 2020)
 - Transformer is a better architecture for a better scaling law
 - Can we have a better or more efficient architecture for better scaling laws?



Mixture-of-Expert

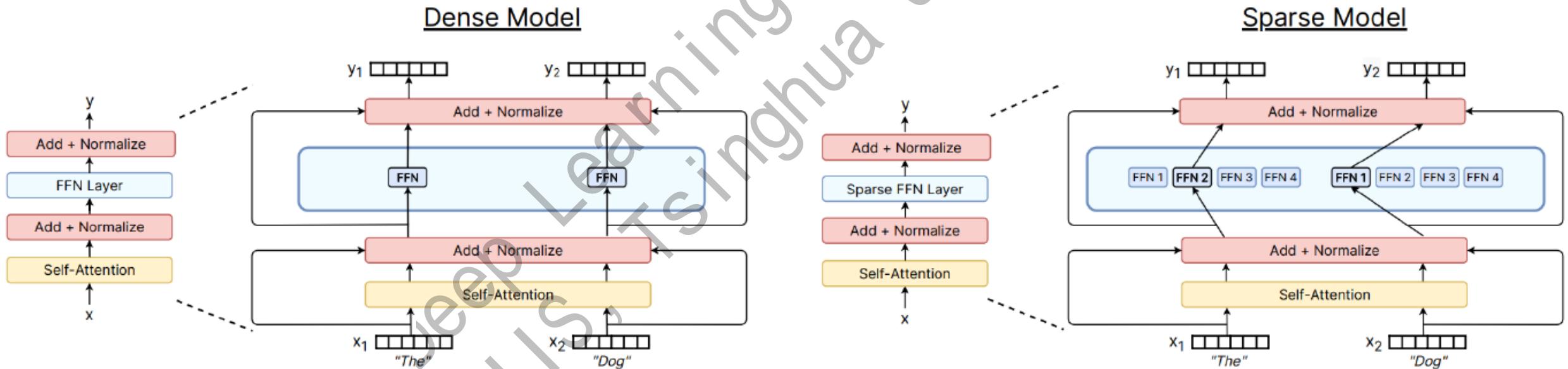


GPT4



Mixture-of-Expert

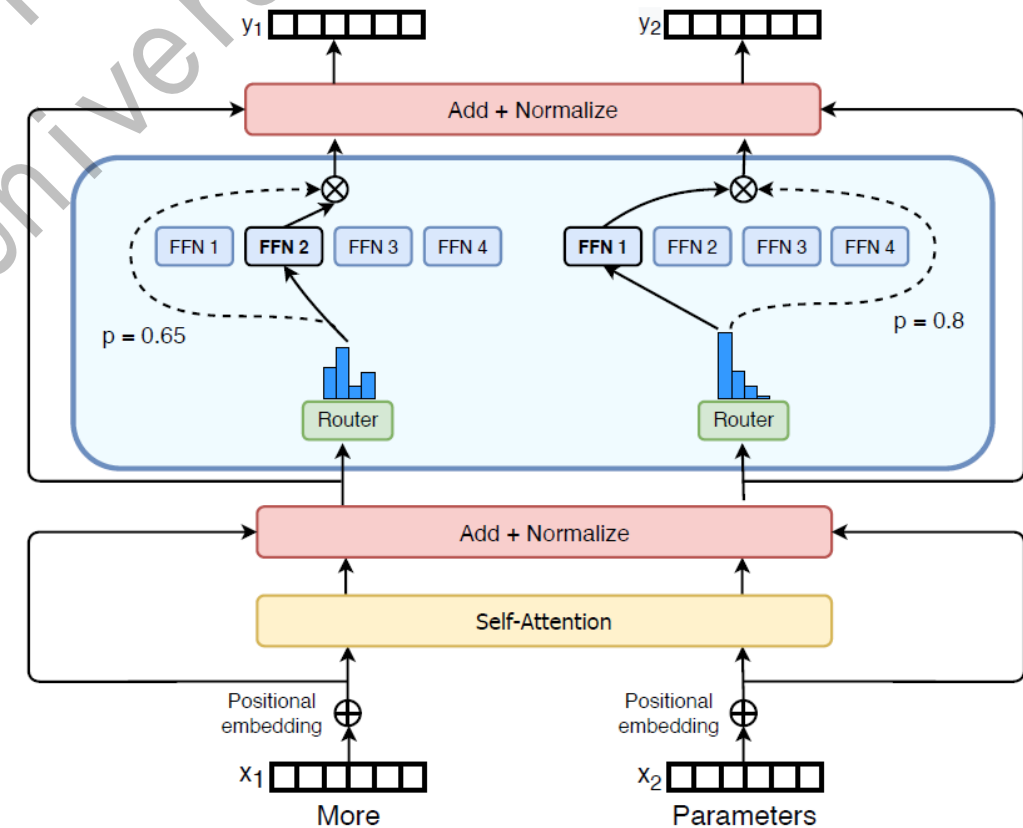
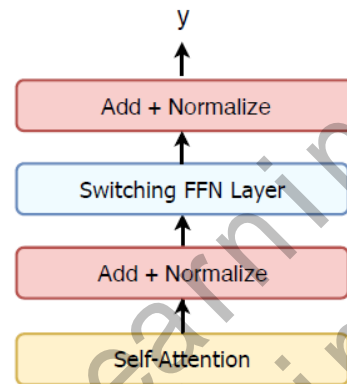
- Switch Transformers (Google, 2022)
 - The key idea is to replace the FFN module in a classical transformer (dense) to a routing module, which consists of a collection of smaller FFNs



Mixture-of-Expert

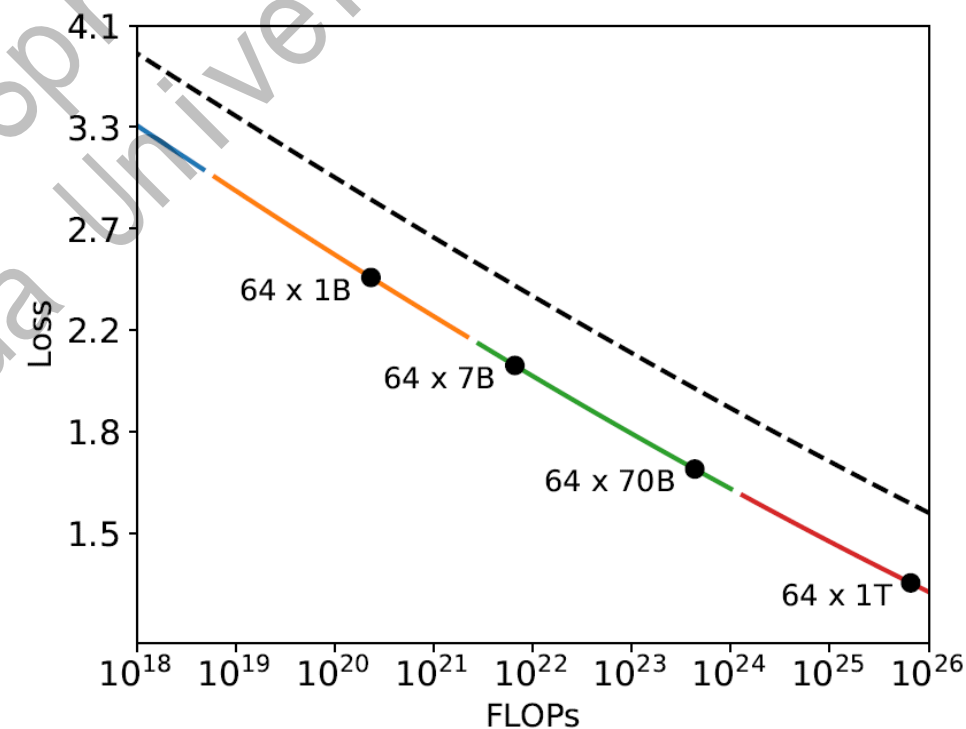
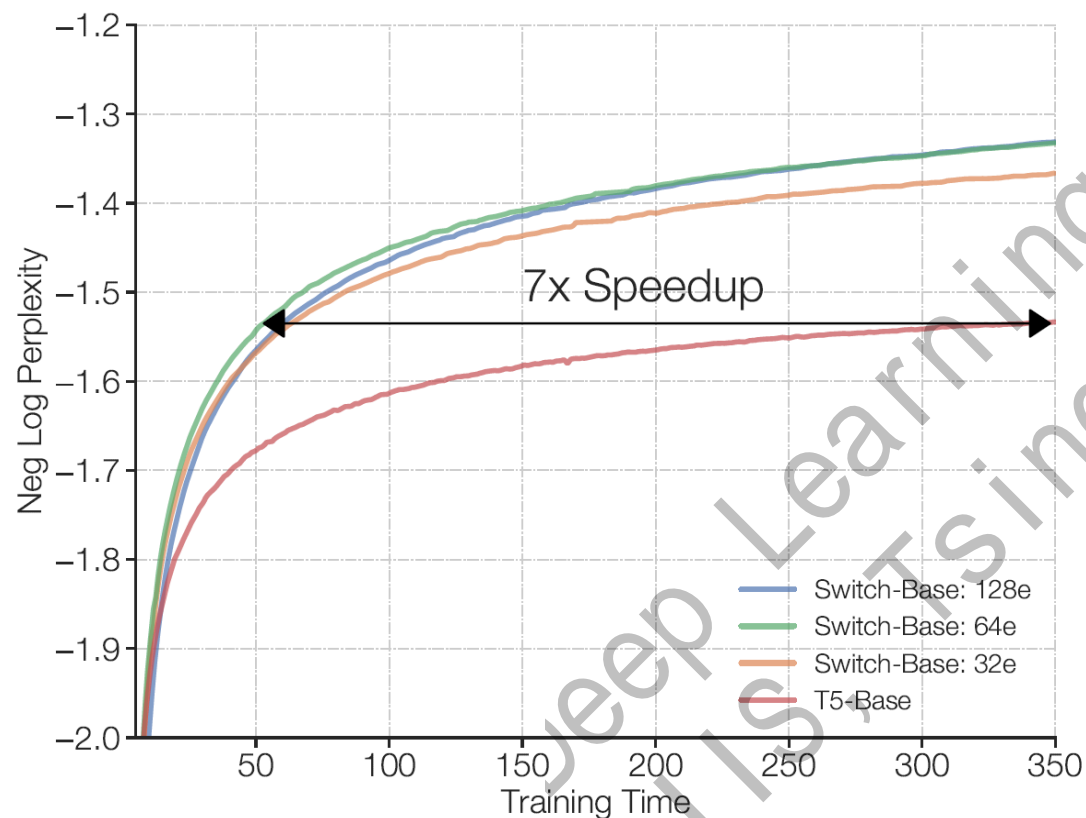
• Switch Transformers (Google, 2022)

- The MoE layer
- Expert
 - The small FFNs
- Router
 - Select which experts to process the token
 - Select top-k experts
- Sparsity
 - Only a few parameters are activated
 - MoE: Same flops \rightarrow more parameters



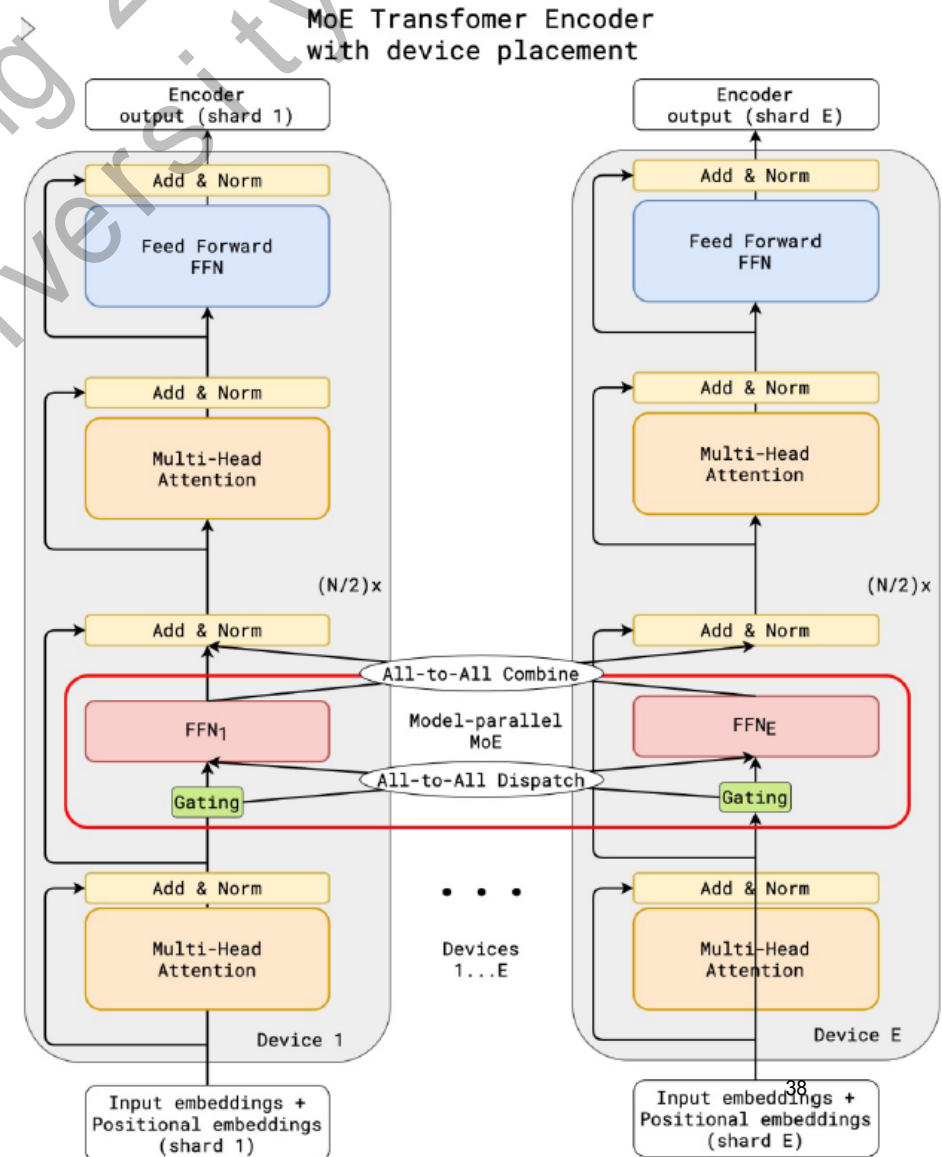
Mixture-of-Expert

- A much more efficient model and a better scaling law



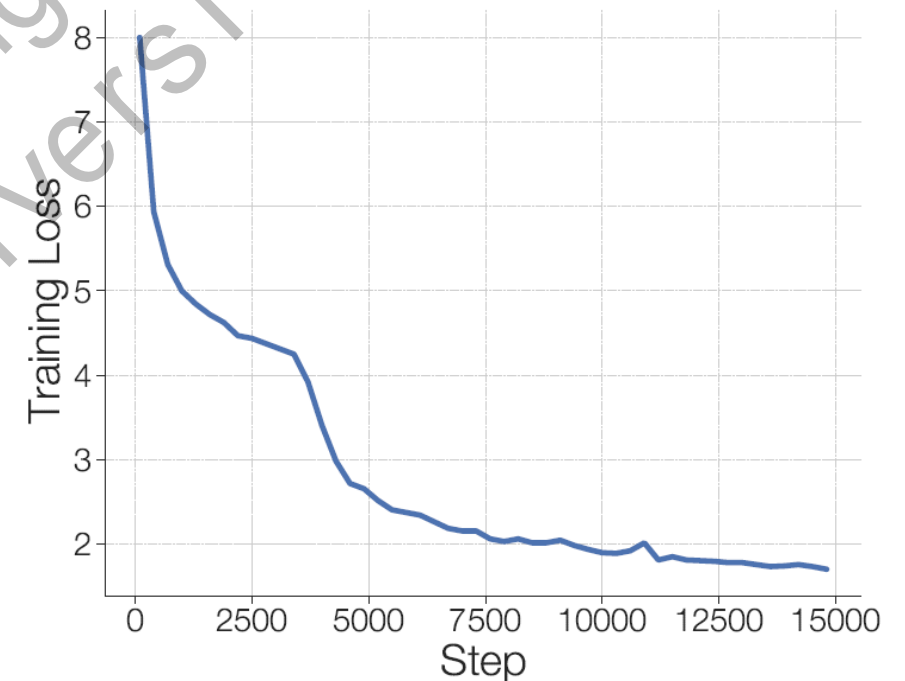
Mixture of Experts: Challenges

- MoE makes training more complex
 - Pros: MoE allows better parallel training
 - Cons: you need a better training system :/



Mixture of Experts: Challenges

- MoE makes training more complex
 - Pros: MoE allows better parallel training
 - Cons: you need a better training system :/
- MoE training can be unstable
 - Expert switching brings significant loss change
 - The load-balancing issue
 - MoE training can easily crash

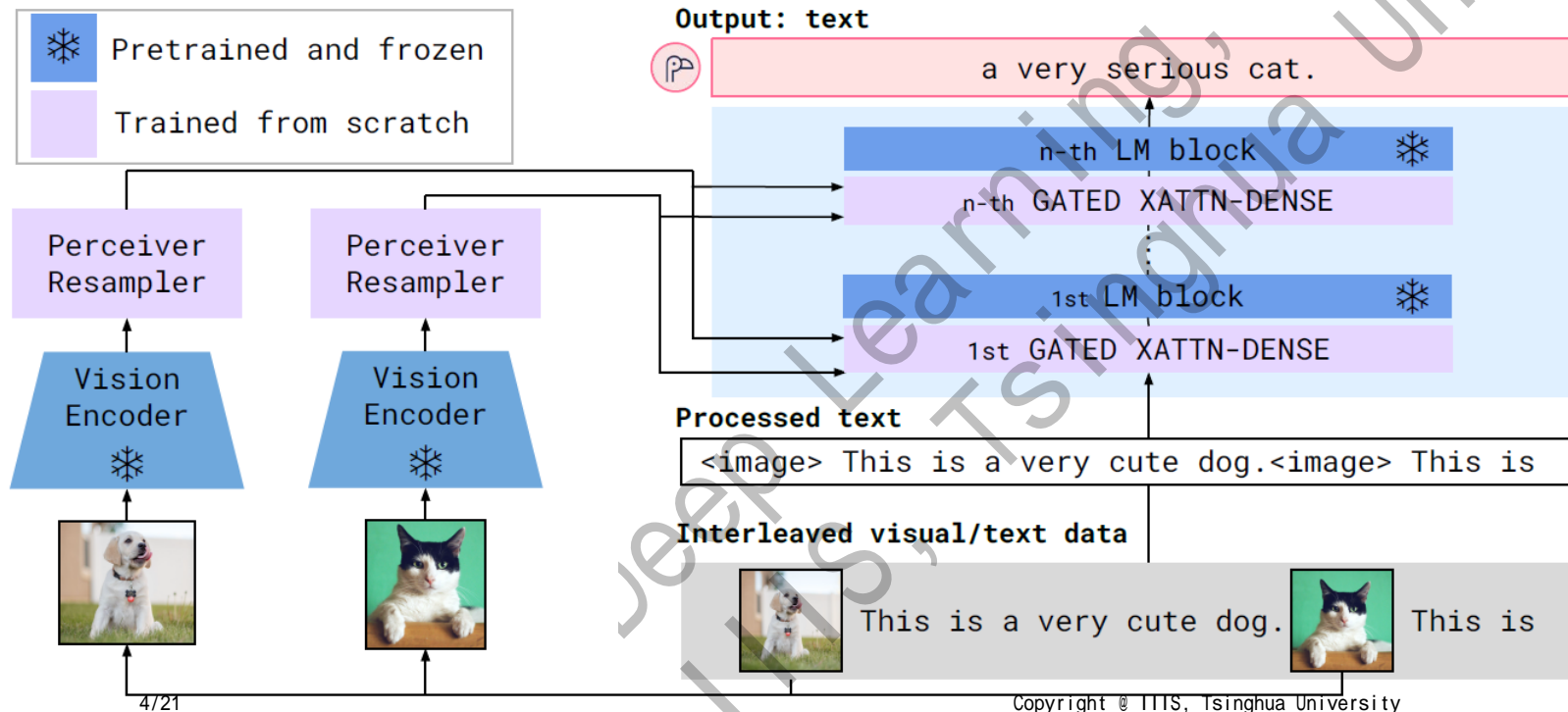


A stable MoE training run from Google
*ST-MoE: Designing Stable and Transferable Sparse
Expert Models (2022)*

MoE models have better capacities, but harder to get training work!

Pretraining Transformer Decoder

- Multi-Modal GPT to Unify Image and Text
 - Flamingo (DeepMind, 2022)



Pretraining Transformer Decoder

- Multi-Modal GPT to Unify Image and Text
 - GPT-4 (OpenAI, 2023.3)

User What is unusual about this image?



GPT-4 The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.

Below is part of the InstructGPT paper. Could you read and summarize it to me?

arXiv:2203.02155v1 [cs.CL] 4 Mar 2022

Training language models to follow instructions with human feedback

Long Ouyang¹, Jeff Wu², Xu Jiang², Diego Almeida¹, Carroll W. Wainwright², Pamela Mishkin¹, Chong Zhang, Saahil Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Mads Simons, Armando Adell¹, Peter Vukobratovic, Paul Christiano¹, Jan Leike², Ryan Lowe²

OpenAI

Abstract

Making language models bigger does not inherently make them better at following a user's intent. For example, large language models can generate outputs that are untruthful, toxic, or simply not helpful to the user. In other words, these models are not aligned with their users. In this paper, we show an avenue for aligning language models with user intent on a wide range of tasks by fine-tuning with human feedback. Starting with a set of labels for model outputs and prompts submitted through the OpenAI API, we collect a dataset of label demonstrations of the desired model behavior, which we use to fine-tune GPT-3 using supervised learning. We then collect a dataset of rankings of model outputs, which we use to further fine-tune this supervised model using reinforcement learning from human feedback. We call the resulting models *InstructGPT*. In human evaluations on our prompt distribution, outputs from the 1.3B parameter *InstructGPT* model are preferred to outputs from the 175B GPT-3, despite having 100x fewer parameters. Moreover, *InstructGPT* models show improvements in truthfulness and reductions in toxic output generation while having minimal performance regressions on public NLP datasets. Even though *InstructGPT* still makes simple mistakes, our results show that fine-tuning with human feedback is a promising direction for aligning language models with human intent.

1 Introduction

Large language models (LLMs) can be “prompted” to perform a range of natural language processing (NLP) tasks, given some examples of the task as input. However, these models often express unintended behaviors such as making up facts, generating biased or toxic text, or simply not following user instructions (Bomeli et al., 2022; Brown et al., 2022; Chowdhury et al., 2022; Hendriks et al., 2022; Radford et al., 2019; Thakur et al., 2022; Thakur et al., 2022). This is because the language modeling objective

¹Primary authors. This was a joint project of the OpenAI Alignment team. RL and IL are the main leads. Corresponding author: jacob@openai.com.
²Work done while at OpenAI. Current Affiliations: AA: Anthropic; PC: Alignment Research Center.

Figure 1: Human evaluation of various models on our API prompt distribution, evaluated by how often outputs from each model were preferred to those from the 175B GPT-3 model.

Figure 1: Human evaluation of various models on our API prompt distribution, evaluated by how often outputs from each model were preferred to those from the 175B GPT-3 model. Our *InstructGPT* models (PPQ-gpt) as well as its variant trained without prompting (PPQ) significantly outperform the GPT-3 baseline (GPT, GPT prompted). Outputs from our 1.3B PPQ-gpt model are preferred to those from the 175B GPT-3. Error bars throughout the paper are 95% confidence intervals.

used for many recent large LMs—predicting the next token on a webpage from the context—is different from the objective “Follow the user’s instruction. Helpfully and safely.” (Radford et al., 2019; Brown et al., 2020; Foster et al., 2020; Wei et al., 2022; Wang et al., 2022). Thus, we say that the language modeling objective is *misaligned*, occurring when undesired behaviors in expectancy important for language models that are deployed and used by hundreds of applications.

We make progress on aligning language models to behave better to act in accordance with the user’s intention (Leike et al., 2018). This encompasses both explicit instructions such as following instructions and implicit intentions such as staying truthful, and not being biased, toxic, or otherwise harmful. Using the language of (Leike et al., 2018), we want language models to be *helpful* (they should help the user solve their task), *honest* (they shouldn’t fabricate information to mislead the user), and *harmless* (they shouldn’t cause physical, psychological, or social harm to people or the environment). We elaborate on the evaluation of these criteria in Section 2.

We focus on *pre-training* approaches to aligning language models. Specifically, we use reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Schulman et al., 2020) to fine-tune GPT-3 to follow a broad class of human instructions (see Figure 2). This technique uses human preferences as a reward signal to fine-tune our models. We test here a team of 40 annotators to label our data, building on their performance on a screening test (see Section 3.2 and Appendix B.1 for more details). We observed a dataset of human-written demonstrations of the desired output behavior on OpenAI’s GPT-3 prompts submitted to the OpenAI API, and some labels for prompts and outputs, and use this to train our supervised learning baseline. Next, we collect a dataset of human-labeled comparisons between outputs from our models and large NLP API prompts. We then train a second model (RM) on this dataset to predict which model output our labelers would prefer. Finally, we use this RM as a reward function and finetune our supervised learning baseline to maximize this reward using the PPO algorithm (Schulman et al., 2017), and illustrate this process in Figure 2. This procedure aligns the behavior of GPT-3 to the stated preferences of a specific group of people (mostly our labelers and researchers), rather than any broader notion of “human values”; we discuss this further in Section 3.3. We call the resulting model *InstructGPT*.

We mainly evaluate our models by having our labelers rate the quality of model outputs on our test set, consisting of prompts chosen to help consumers (who are not represented in the training data). We also conduct automatic evaluations on a range of public NLP datasets. We train these models

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.

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, human A-D are samples from our models that got ranked by labelers. See Section 3.1 for more details on our method.

sizes (1.3B, 6.7B, and 175B parameters), and all of our models use the GPT-3 architecture. Our main findings are as follows:

Labelers significantly prefer InstructGPT outputs over outputs from GPT-3. On our test set, outputs from the 1.3B parameter *InstructGPT* model are preferred to outputs from the 175B GPT-3, despite having over 100x fewer parameters. These models have the same architecture, and differ only by the fact that *InstructGPT* is fine-tuned on our human data. This result holds true even when we add a few-shot prompt to GPT-3 to make it better at following instructions. Outputs from our 175B *InstructGPT* are preferred to 175B GPT-3 outputs 80 \pm 3% of the time, and performed 71 \pm 4% of the time to fine-tune 175B GPT-3. *InstructGPT* models also generate more appropriate outputs according to our labelers, and more reliably follow explicit constraints in the instruction.

InstructGPT models show improvements in truthfulness over GPT-3. On the TruthfulQA benchmark, *InstructGPT* generates truthful and informative answers about twice as often as GPT-3. Our results are equally strong on the subset of questions that were not adversarially selected against GPT-3. On “closed-domain” tasks from our API prompt distribution, where the output should not contain information that is not present in the input (e.g. summarization and closed-domain QA), *InstructGPT* models make up information not present in the input about half as often as GPT-3 (a 21% vs. 41% hallucination rate, respectively).

InstructGPT shows small improvements in toxicity over GPT-3, but not bias. To measure toxicity, we use the RealToxicityPrompts dataset (Gehrmann et al., 2020) and conduct both automatic and human evaluations. *InstructGPT* models generate about 25% fewer toxic outputs than GPT-3 when prompted to be respectful. *InstructGPT* does not significantly improve over GPT-3 on the WinoGrande (WinoGrande et al., 2018) and CoSHPans (Nogueira et al., 2020) datasets.

We can minimize performance regressions on public NLP datasets by modifying our RLHF fine-tuning procedure. During RLHF fine-tuning, we observe performance regressions compared to GPT-3 on certain public NLP datasets, notably SQuAD (Rajpurkar et al., 2016), DROP (Dua et al., 2019), Halliwag (Gehrmann et al., 2019), and WMT 2013 French to English translation (Beyler et al., 2019). This is an example of an “alignment tax” since our alignment procedure corrects at the cost of

GPT-4 The InstructGPT paper focuses on training large language models to follow instructions with human feedback. The authors note that making language models larger doesn't inherently make them better at following a user's intent. Large models can generate outputs that are untruthful, toxic, or simply unhelpful.

Exam results (ordered by GPT-3.5 performance)

Estimated percentile lower bound (among test takers)

gpt-4

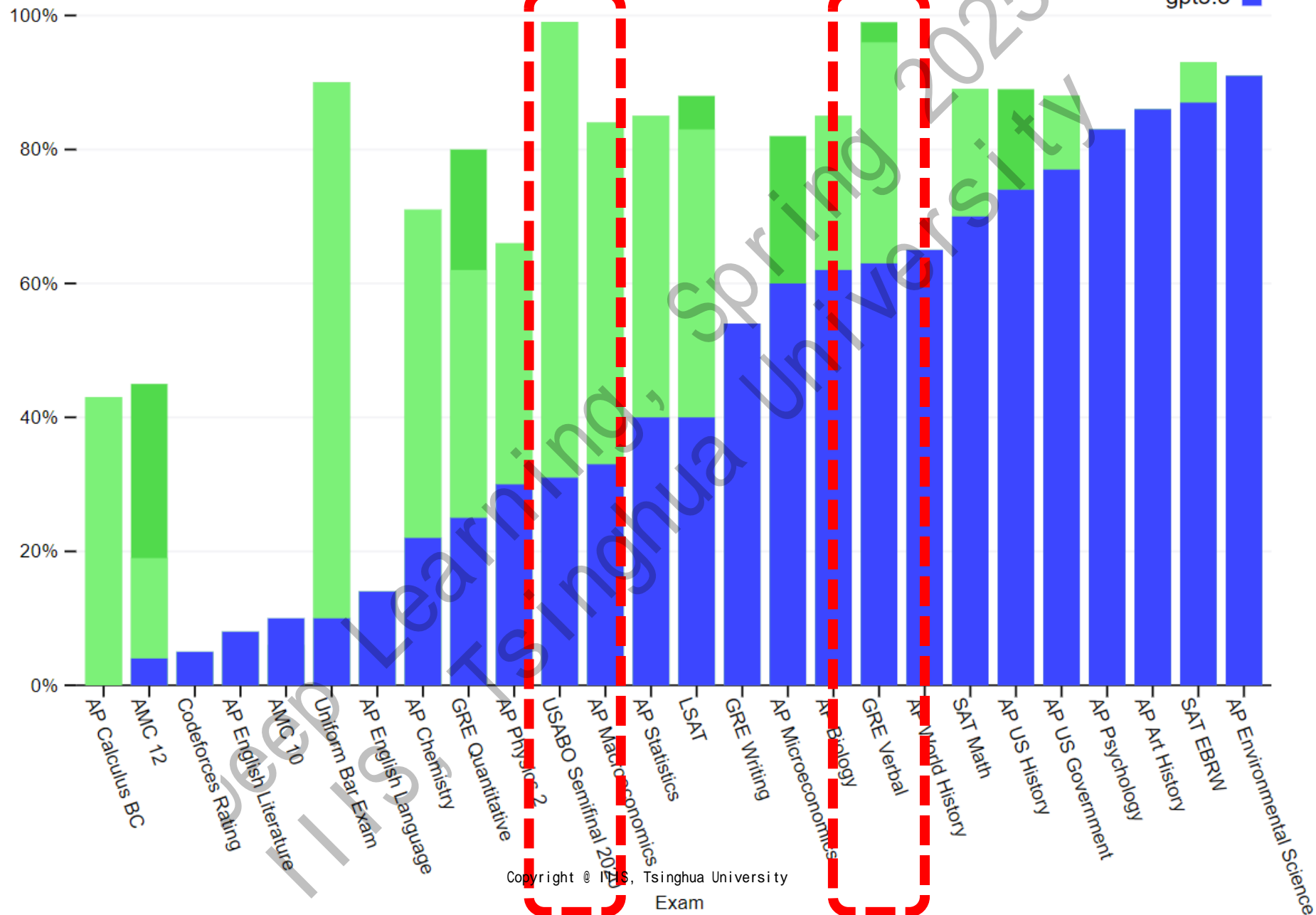
gpt-4 (no vision)

gpt3.5

OpenPsi @ IIS

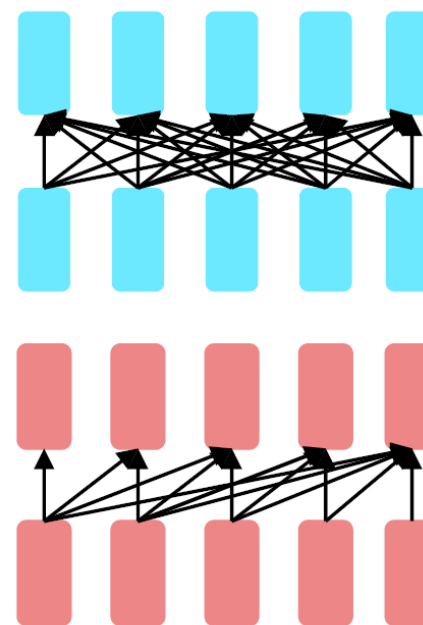
Pret

- Mul
- G



Pretraining Transformers

- Collect a large amount of corpus and pretrain a large transformer
- For down-stream tasks, fine-tune the pretrained model
 - Or use the pretrained model to extract features
- How to pretrain a transformer on texts?
 - Pretrain an encoder
 - Bi-directional
 - Pretrain a decoder
 - Auto-regressive (e.g., GPT-X and more)
 - Also both encoder and decoder



Encoders

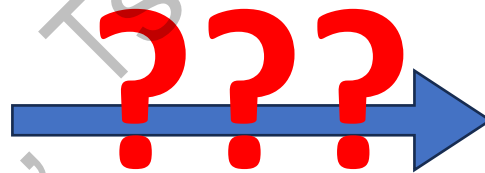
Decoders

Summary

- Pretraining Transformers
 - Pretraining encoders for representation learning
 - Pretraining decoders for emergent multi-task/in-context learning capabilities
- Scaling Law
 - Larger model + more data + more compute \rightarrow better LLMs



2019



2022

Early-days of GPT-3

- Not “usable”

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.

An Instruction-Following Problem

Human Instruction

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.

GPT does not respond
to the instruction

An Instruction-Following Problem

Human Instruction

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.

The instruction-following problem is solved by **Reinforcement Learning**

GPT does not respond to the instruction

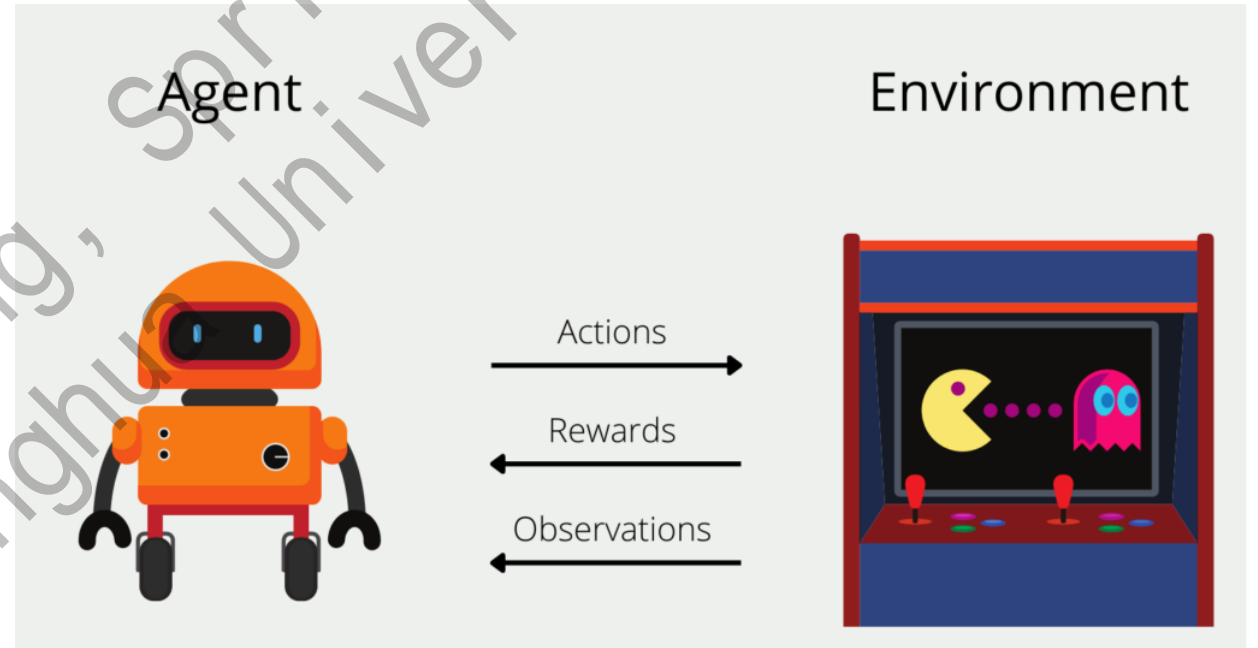
Reinforcement Learning

- Sequence decision-making
- No gold-standard solutions
 - The model must explore for the best strategy



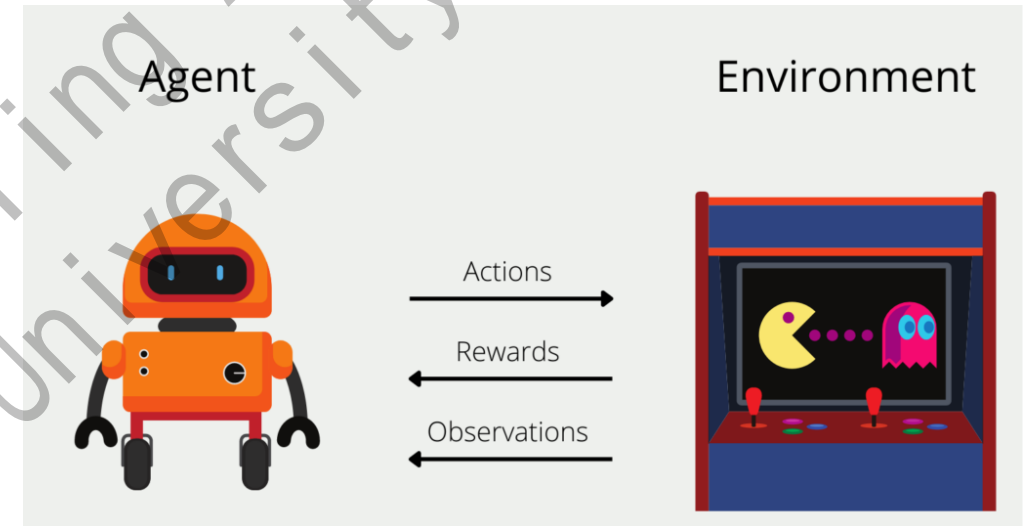
Reinforcement Learning: Formulation

- Environment (Task)
 - Observation
 - Transition
 - Reward
- Agent / Policy
 - Input: observation
 - Output: action
 - Objective: maximize reward
- Key components
 - **Environment**
 - **Action; Observation; Reward**



Reinforcement Learning v.s. Language Model

- Reinforcement Learning
 - Massive self-generated data
 - Trial-and-error
 - Both good data and bad data
 - **Clear Task & Reward function**



How to connect them?

- Language Model
 - A large prepared training corpus
 - No exploration needed
 - Next token prediction
 - **No reward/task**



Instruct Following Meets GPT

- An **instruction following** challenge for a “**usable**” GPT

Prompt

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

Instruction

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.

Action

Reward???

Instruct Following Meets GPT

- Key challenges for GPT following instructions
 - Complex & diverse instructions 极其复杂的指令, 任务多样
 - Non-trivial reward evaluation 很难评估奖励完成度
- Solution: InstructGPT (OpenAI, 2022)

InstructGPT

- Key challenges for GPT for instruction tuning
 - Complex & diverse instructions
 - Non-trivial reward evaluation
- Key Ideas of InstructGPT
 - Collect human feedbacks
 - 使用人类的反馈训练GPT
- Human labeler can judge the quality of human instructions

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.

InstructGPT

- InstructGPT follows complex instructions

Prompt:

What is the purpose of the list C in the code below?

```
def binomial_coefficient(n, r):  
    C = [0 for i in range(r + 1)];  
    C[0] = 1;  
    for i in range(1, n + 1):  
        j = min(i, r);  
        while j > 0:  
            C[j] += C[j - 1];  
            j -= 1;  
    return C[r]
```

GPT-3 175B completion:

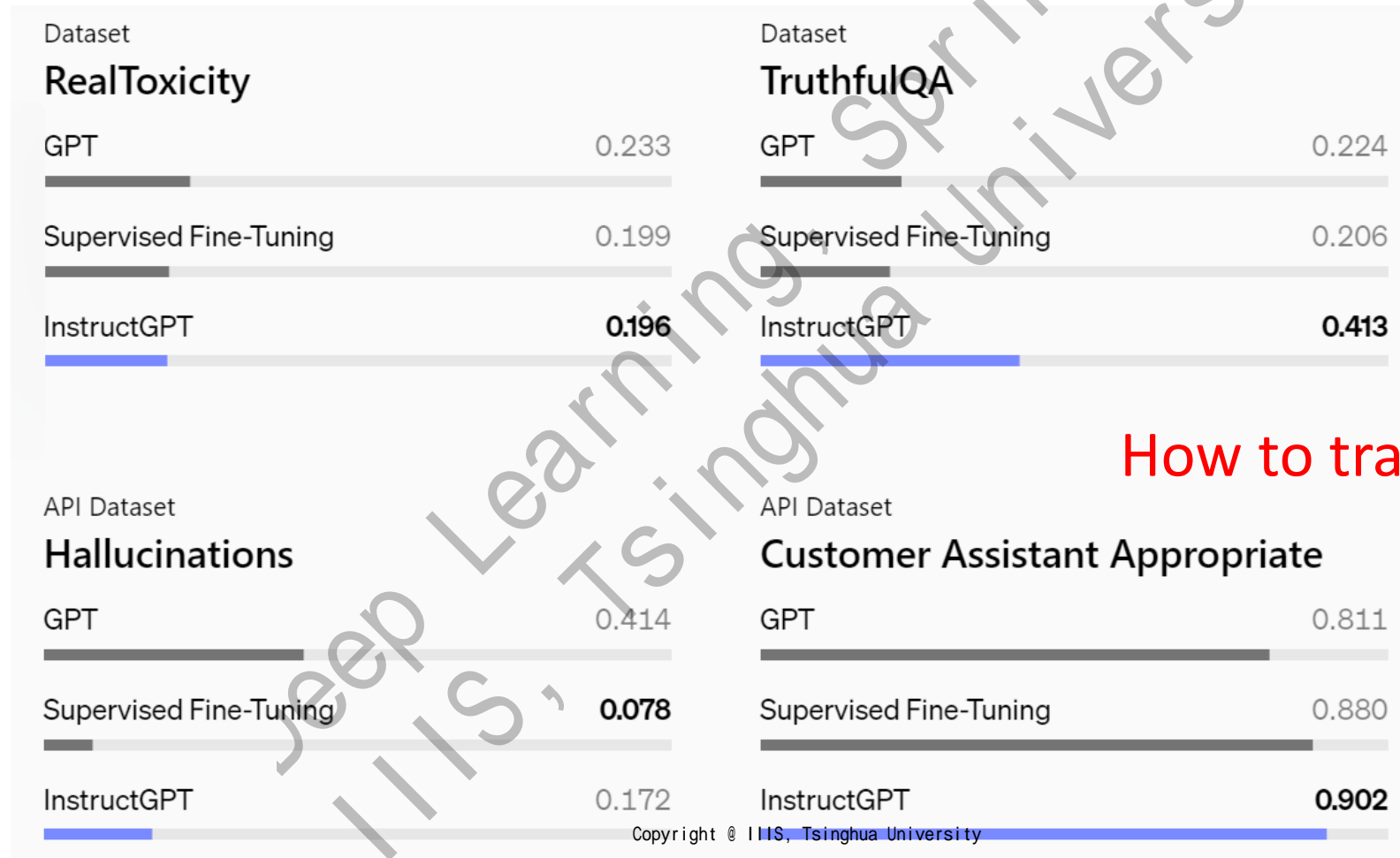
- A. to store the value of C[0]
- B. to store the value of C[1]
- C. to store the value of C[i]
- D. to store the value of C[i - 1]

InstructGPT 175B completion:

The list C in this code is used to store the values of the binomial coefficient as the function iterates through the values of n and r. It is used to calculate the value of the binomial coefficient for a given value of n and r, and stores the result in the final return value of the function.

InstructGPT

- InstructGPT reduces hallucinations



How to train InstructGPT?

InstructGPT Explained

- Goal: fine-tune GPT to follow human instructions
 - 微调GPT来遵从人类指令
- How to collect human data for fine-tuning?
 - We need to collect prompts (收集人类指令)
 - We need to collect responses for the prompts (收集指令的反馈)

Prompt:

Serendipity means the occurrence and development of events by chance in a happy or beneficial way. Use the word in a sentence.

Labeler demonstration

Running into Margaret and being introduced to Tom was a fortunate stroke of serendipity.

Supervised Fine-Tuning (SFT)

InstructGPT Explained

- Goal: fine-tune GPT to follow human instructions
 - 微调GPT来遵从人类指令
- SFT: fine-tuning on human demonstrations
- What about the RL perspective?
 - Pros: GPT can self-explore & RL is powerful
 - Cons: **Reward????**

InstructGPT Explained

- Goal: fine-tune GPT to follow human instructions
 - 微调GPT来遵从人类指令
- SFT: fine-tuning on human demonstrations
- What about the RL perspective?
 - Pros: GPT can self-explore & RL is powerful
 - Cons: Reward????
- Key Idea: RL with **a learned reward**
 - Generate multiple outputs
 - 生成多个输出
 - Ranking by humans
 - 人类排序
 - Learn a reward model
 - 学习一个满足偏序的奖励函数

**Reinforcement Learning
from Human Feedback
(RLHF)**



InstructGPT Explained

Ranking outputs

To be ranked

B A team of researchers from Yale University and University of California, Davis studied the vocalization patterns of several different types of parrots. They found that parrots like to mimic human speech, and can produce a wide range of sounds, such as whistles, squawks, and other types of vocalizations...

C Parrots have been found to have the ability to understand numbers. Researchers have found that parrots can understand numbers up to six. In a series of experiments, the parrots were able to identify the amount of food items under a number of cups...

Rank 1 (best)

A A research group in the United States has found that parrots can imitate human speech with ease, and some of them can even do so in the same way as humans. This group studied the sounds that parrots make in their natural habitats and found that they use their tongues and beaks in ways that are strikingly...

Rank 2

Rank 3

E Scientists have found that green-winged parrots can tell the difference between two noises that are the same except for the order in which they are heard. This is important because green-winged parrots are known to imitate sounds. This research shows that they are able to understand the difference between sounds.

Rank 4

Rank 5 (worst)

D Current research suggests that parrots see and hear things in a different way than humans do. While humans see a rainbow of colors, parrots only see shades of red and green. Parrots can also see ultraviolet light, which is invisible to humans. Many birds have this ability to see ultraviolet light, an ability

InstructGPT Explained

- The overall pipeline

Step 1

Collect demonstration data, and train a supervised policy.

A prompt is sampled from our prompt dataset.

Explain the moon landing to a 6 year old

A labeler demonstrates the desired output behavior.

Some people went to the moon...

This data is used to fine-tune GPT-3 with supervised learning.

SFT

SFT

Step 2

Collect comparison data, and train a reward model.

A prompt and several model outputs are sampled.

Explain the moon landing to a 6 year old

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

A labeler ranks the outputs from best to worst.

D > C > A = B

This data is used to train our reward model.

RM

Reward

Step 3

Optimize a policy against the reward model using reinforcement learning.

A new prompt is sampled from the dataset.

Write a story about frogs

The policy generates an output.

PPO

Once upon a time...

The reward model calculates a reward for the output.

RM

The reward is used to update the policy using PPO.

r_k

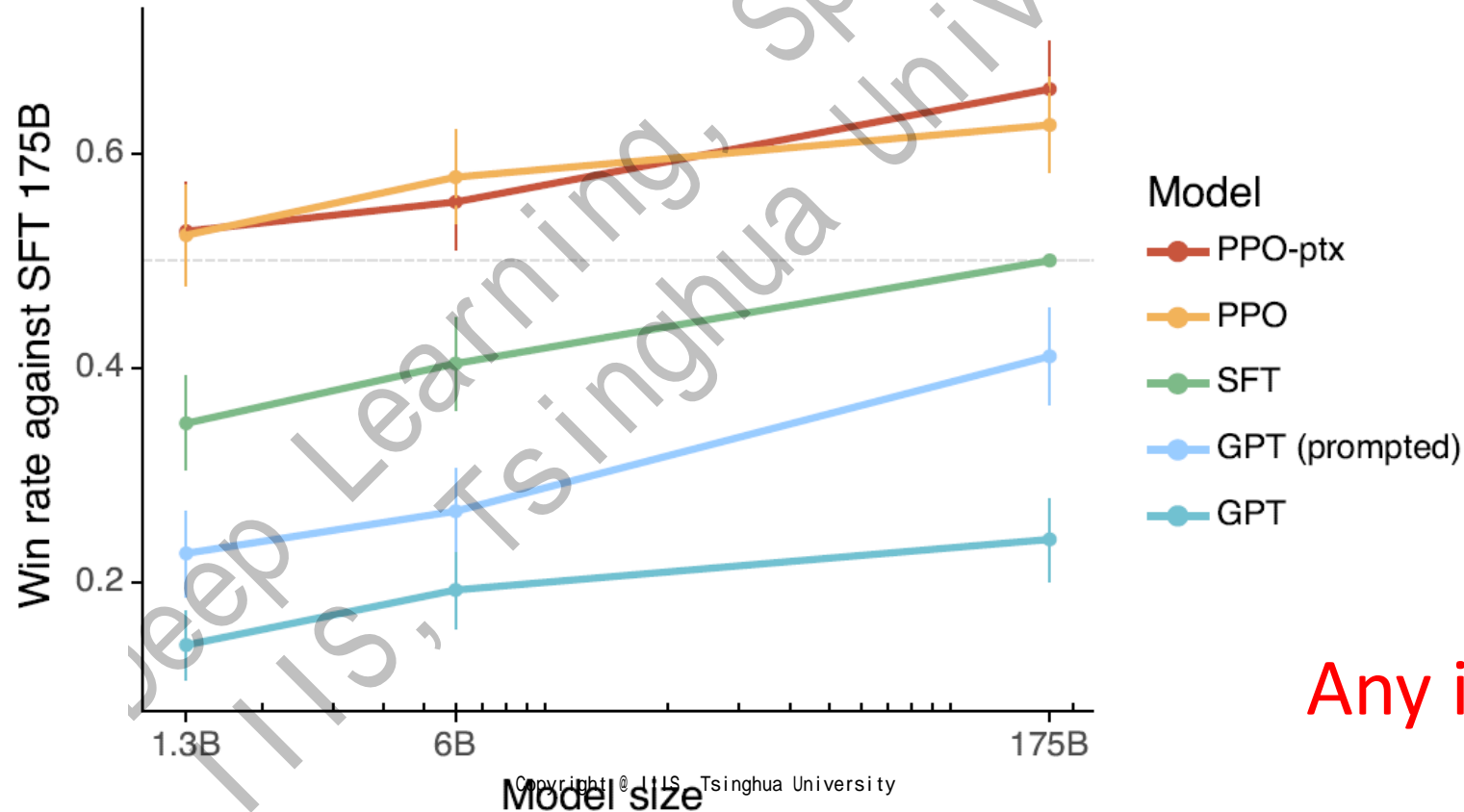
RLHF

Pre-train v.s. Post-train

- InstructGPT / ChatGPT 总体流程
 - Pre-trained base model
 - Massive corpus
 - Post-train using human feedback
 - SFT: human demo
 - RLHF: human preference
- Pre-train v.s. Post-train
 - Data: massive v.s. high-quality (海量 v.s. 高质量)
 - Objective: compression v.s. instruction following (压缩 v.s. 指令遵从)
 - Dynamics: static v.s. online feedback (静态数据集 v.s. 在线反馈收集)
- Comment: the post-train phase is also called the alignment phase (对齐)
 - Align the LLM to human preferences (将语言模型和人类偏好对齐)

SFT v.s. RLHF

- Why do we need RLHF? Or Isn't SFT sufficient for post-train?
 - The InstructGPT result



Hallucination

- SFT-based post-training causes Hallucination
 - From John Schulman's talk at Berkeley (Apr, 2023)

Write a bio of AI researcher John Schulman

John Schulman is an AI research scientist at OpenAI, the leading AI research lab in the world. He has been a professor of computer science at Carnegie Mellon University since 2012, and an associate professor since 2009. He has written multiple papers on reinforcement learning, and is a recipient of the prestigious Sloan Research Fellowship. His research on deep reinforcement learning has been published in top journals such as Nature and Science. He is also active in the open source community, having contributed to the popular deep learning frameworks TensorFlow and PyTorch. He has also served as a reviewer for the International Conference on Machine Learning and the International Conference on Artificial Intelligence. His current research focuses on developing advanced reinforcement learning algorithms and applying them to complex real-world tasks. He is also an advisor to several startups that specialize in AI.

- A thought experiment:
 - Suppose ChatGPT's knowledge cut-off is 2023.
 - Let's fine-tune ChatGPT using SFT to describe EuroCup 2024.
 - What will happen?

Hallucination

- SFT-based post-training causes Hallucination
 - From John Schulman's talk at Berkeley (Apr, 2023)

Write a bio of AI researcher John Schulman

John Schulman is an AI research scientist at OpenAI, the leading AI research lab in the world. He has been a professor of computer science at Carnegie Mellon University since 2012, and an associate professor since 2009. He has written multiple papers on reinforcement learning, and is a recipient of the prestigious Sloan Research Fellowship. His research on deep reinforcement learning has been published in top journals such as Nature and Science. He is also active in the open source community, having contributed to the popular deep learning frameworks TensorFlow and PyTorch. He has also served as a reviewer for the International Conference on Machine Learning and the International Conference on Artificial Intelligence. His current research focuses on developing advanced reinforcement learning algorithms and applying them to complex real-world tasks. He is also an advisor to several startups that specialize in AI.

- A thought experiment:
 - Suppose ChatGPT's knowledge cut-off is 2023.
 - Let's fine-tune ChatGPT using SFT to describe EuroCup 2024.
 - **GPT does not know this fact, so we may just teach it to hallucinate!**

Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness
- Key Idea: a properly designed reward fixes the hallucination issue
 - 存在一种可能得奖励函数，鼓励模型在不知道的时候说不知道

- 2) Use RL to precisely learn behavior boundary.
 - $\text{Reward}(x) = \{$
 - 1 if unhedged correct (The answer is y)
 - 0.5 if hedged correct (The answer is likely y)
 - 0 if uninformative (I don't know)
 - 2 if hedged wrong (The answer is likely z)
 - 4 wrong (The answer is z) $\}$

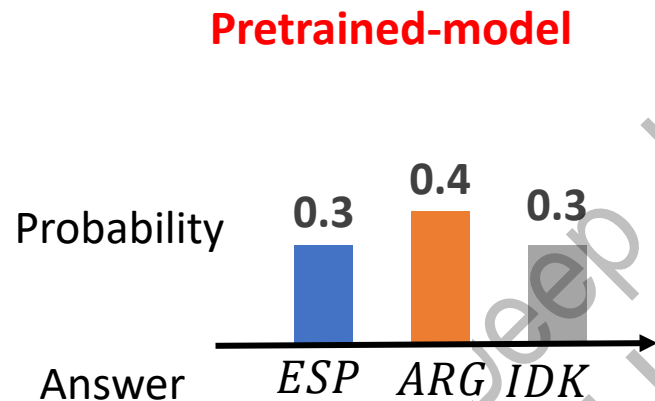
**A good reward
model matters!**

Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness (with a proper reward model)
 - RL can also improve model capacity

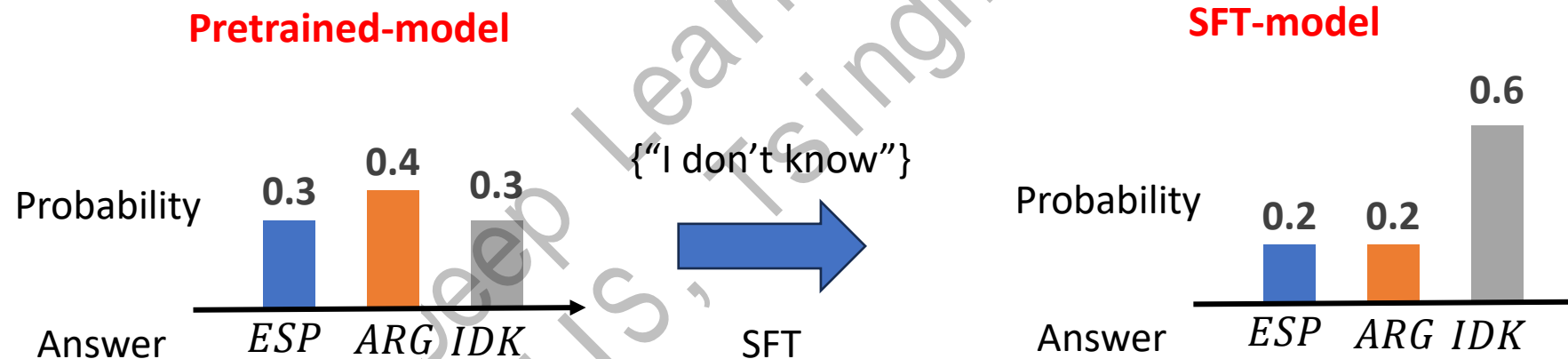
Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness (with a proper reward model)
 - RL can also improve model capacity
 - A thought experiment: “***Who won the 2026 world cup?***”



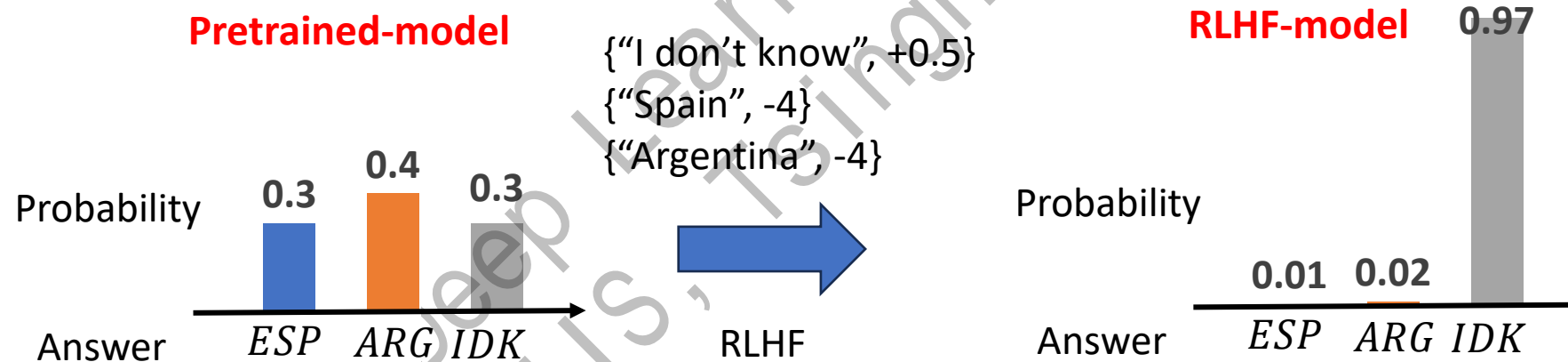
Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness
 - RL can also improve model capacity
 - A thought experiment: “***Who won the 2026 world cup?***”



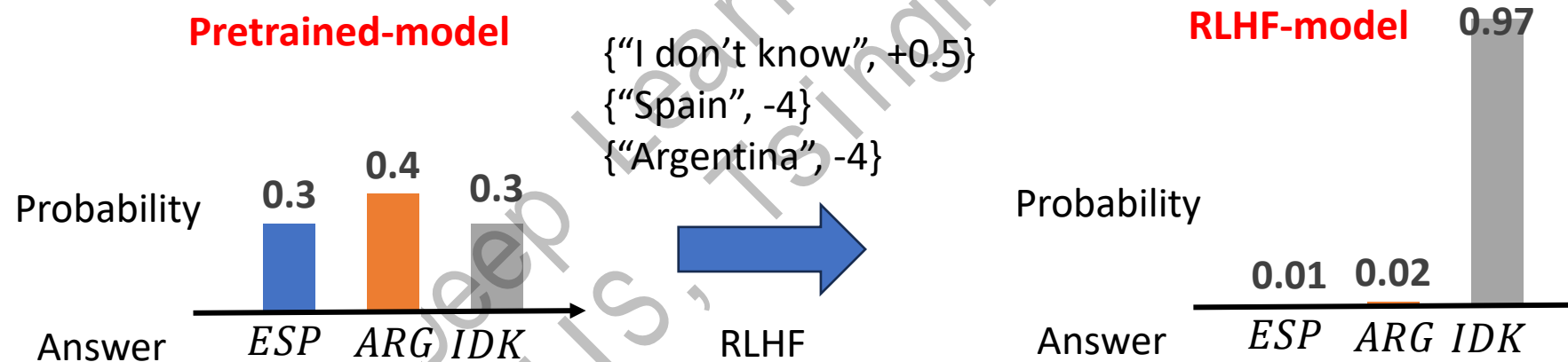
Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness
 - RL can also improve model capacity
 - A thought experiment: “**Who won the 2026 world cup?**”



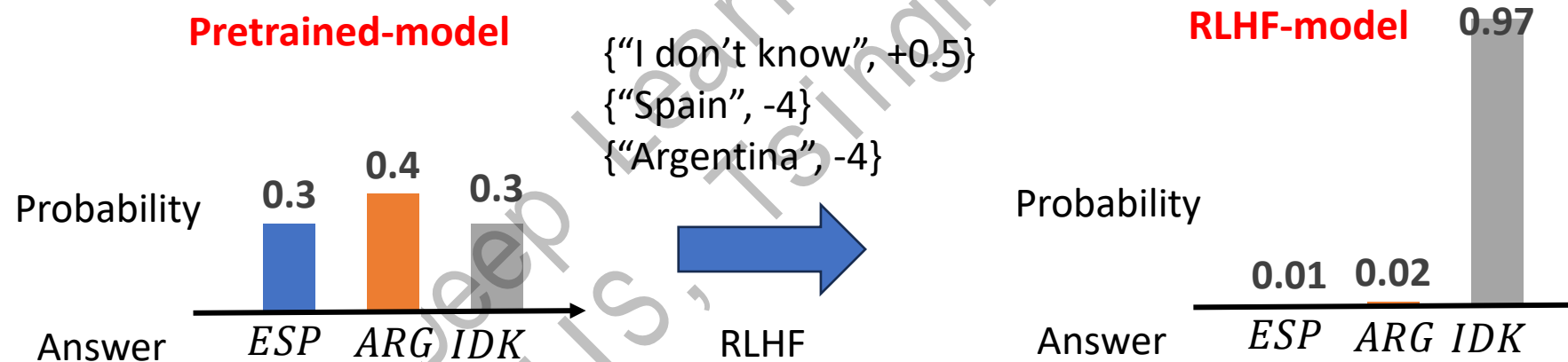
Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness
 - RL can also improve model capacity
 - A thought experiment: “**Who won the 2026 world cup?**”
 - Remark#1: Self-exploration enables better model capacity!



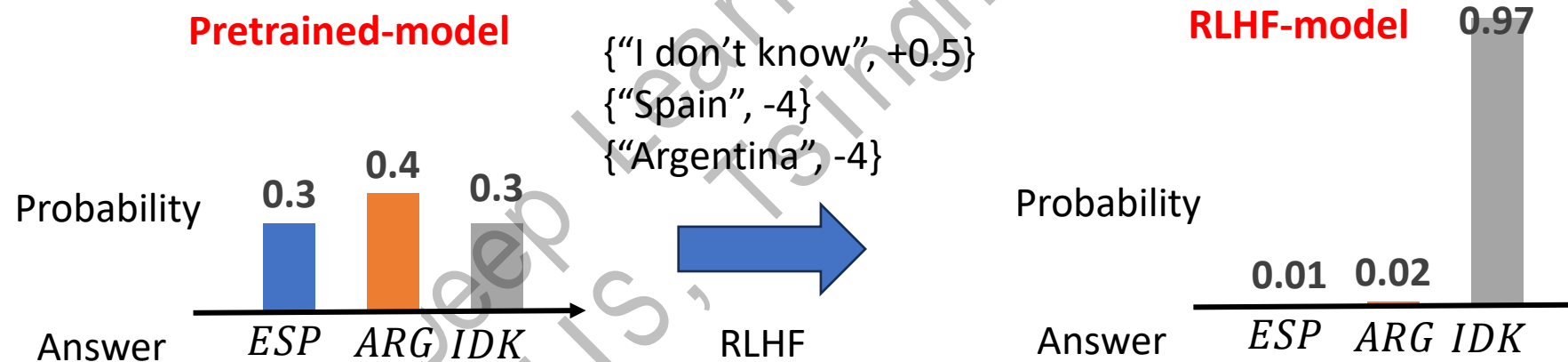
Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness
 - RL can also improve model capacity
 - A thought experiment: ***“Who won the 2026 world cup?”***
 - Remark#1: Self-exploration enables better model capacity!
 - Remark#2: SFT only leverage positive samples, but negative samples also matters!



Hallucination

- SFT-based post-training causes Hallucination
 - RLHF can help uncertainty awareness
 - RL can also improve model capacity
 - A thought experiment: ***“Who won the 2026 world cup?”***
 - Remark#1: Self-exploration enables better model capacity!
 - Remark#2: SFT only leverage positive samples, but negative samples also matters!



But there will always be latest knowledge that the LLM does not know!

Inference-Time Enhancement

How many champion's league titles does real Madrid men's football club have?

As of April 2025, Real Madrid men's football club has won 14 UEFA Champions League titles.

Here's a list of the years they won:

1. 1955–56

.....

13. 2017–18

14. 2021–22

Hallucination!

(Real Madrid also won the 2024 title)

What will you do for this question?

Infer

How many champion's league titles does real Madrid men's football club have?



Real Madrid's men's football team has won the UEFA Champions League a record 15 times. Their most recent victory came in the 2023–24 season, when they defeated Borussia Dortmund 2–0 in the final.

Real Madrid's Champions L

SI

Real Madrid Champions League Titles: How Many Have They Won?

5 March 2025 — Real Madrid have won 15 Champions League titles, the most by any team in the competition's...

Ask anything

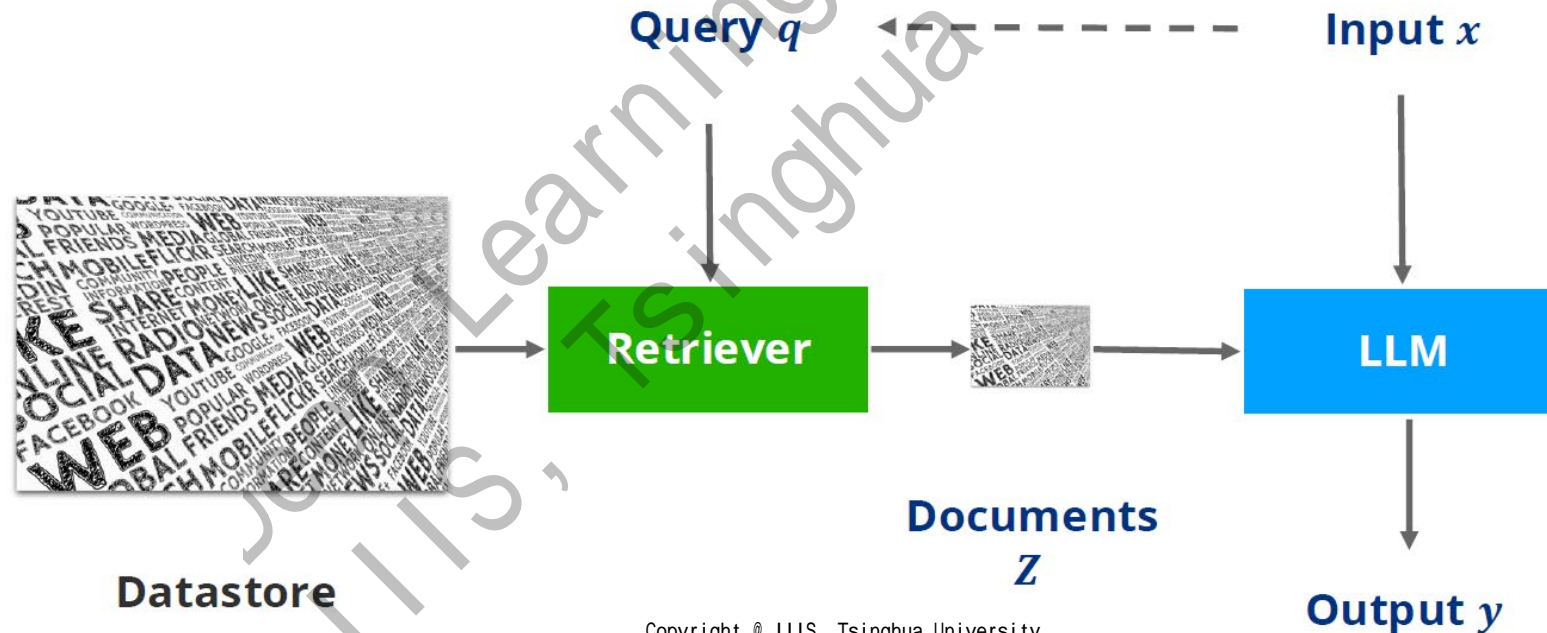


Search



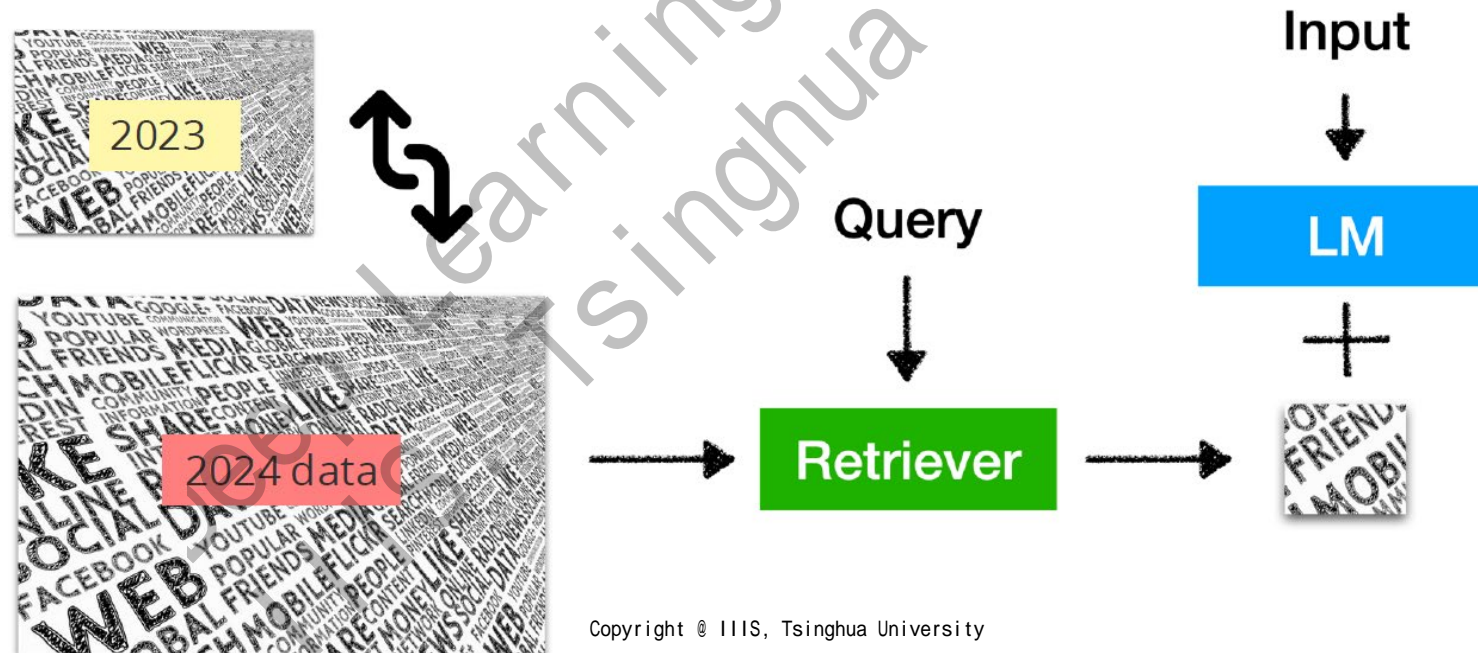
Inference-Time Enhancement

- Retrieval-Augmented Generation (RAG)
 - Before the LLM generates a response, we perform an additional retrieval step and put the results in the context



Inference-Time Enhancement

- Retrieval-Augmented Generation (RAG)
 - Before the LLM generates a response, we perform an additional retrieval step and put the results in the context
 - We can easily update the model knowledge without the need of re-training



Inference-Time Enhancement

- What about a reasoning question?
 - E.g., compute 24 using 4, 4, 7, 7

Please calculate 24 using these 4 digits: 4, 4, 7, 7. Please only give the solution in 1 equation without any additional texts.

  < 4/4 >

$$(7 - 4) * (7 + 4) = 24$$

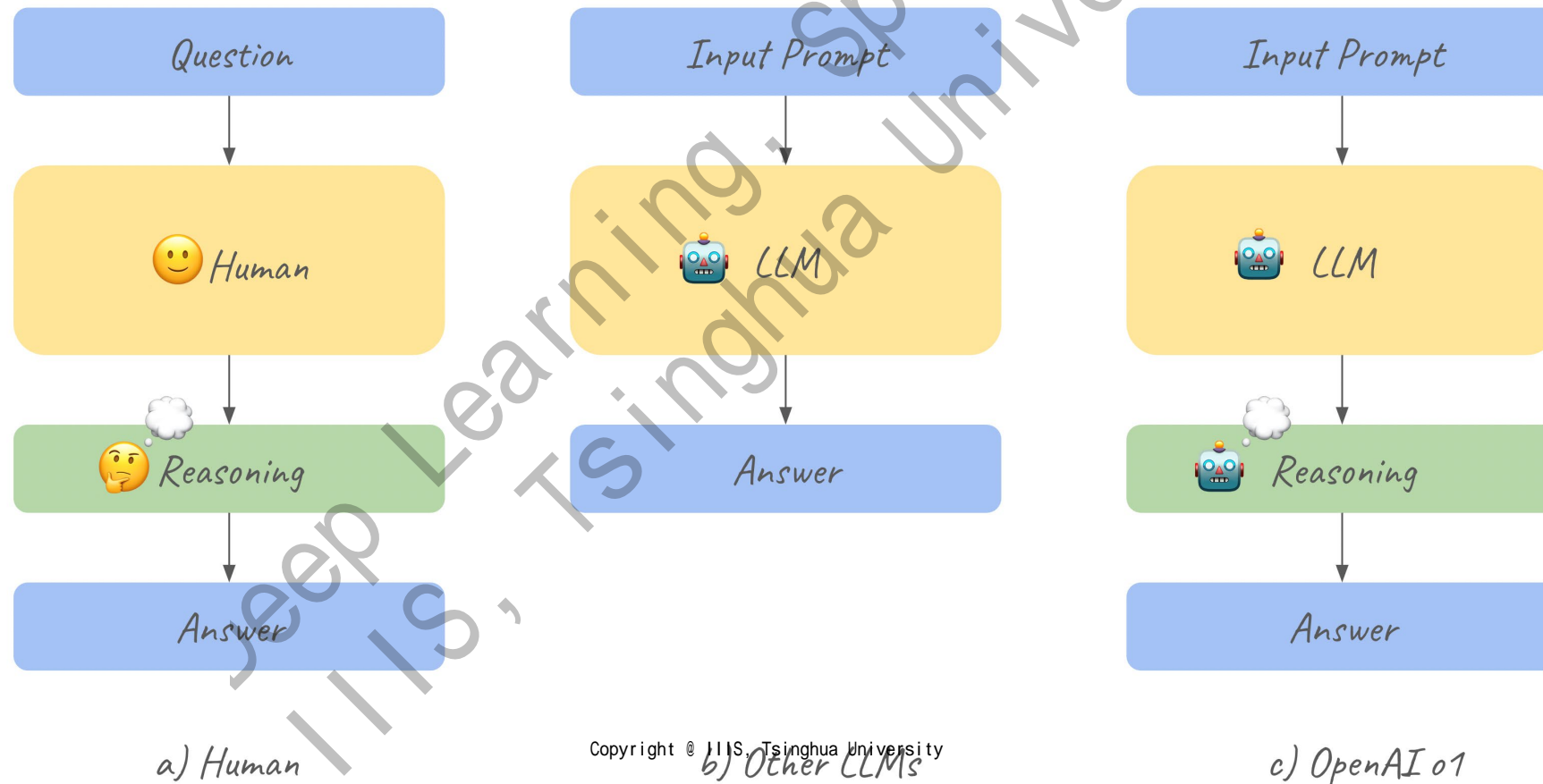
     

What will human do?

Copyright © IIIS, Tsinghua University

Inference-Time Enhancement

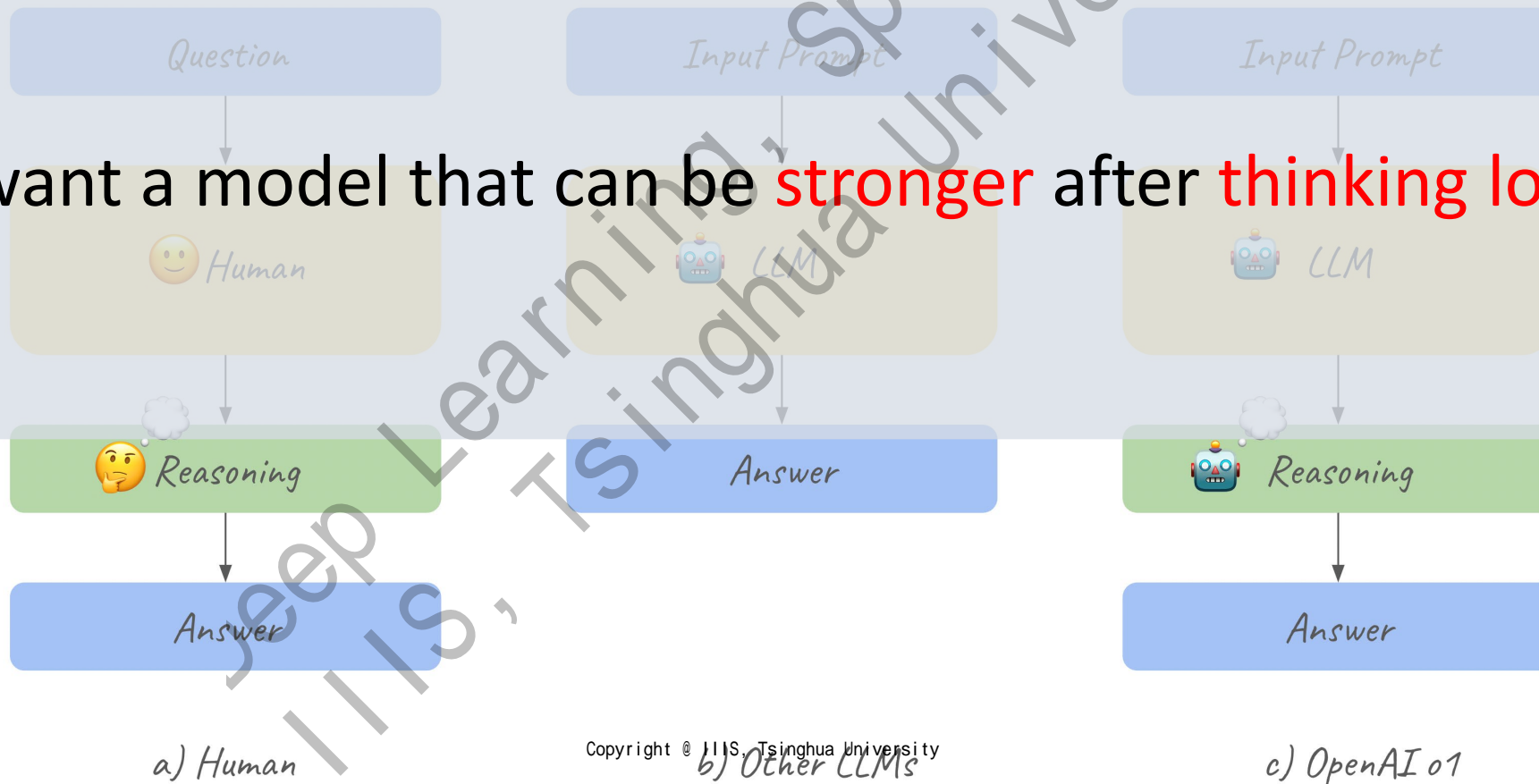
- OpenAI o1 model (2024)
 - Large Reasoning Model (LRM). An LLM “thinks” before giving a respond



Inference-Time Enhancement

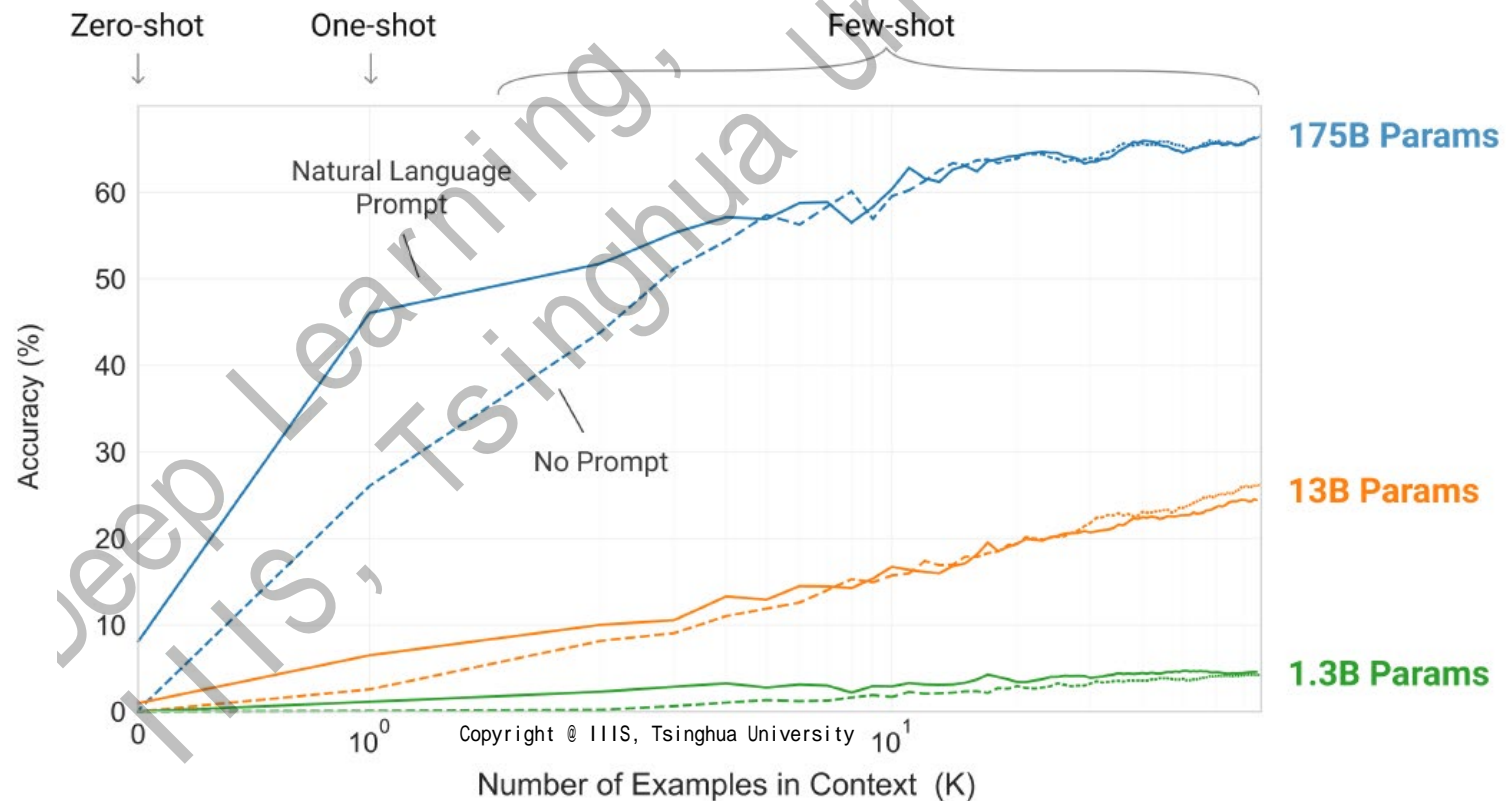
- OpenAI o1 model (2024)
 - Large Reasoning Model (LRM). An LLM “thinks” before giving a response

We want a model that can be **stronger** after **thinking longer**



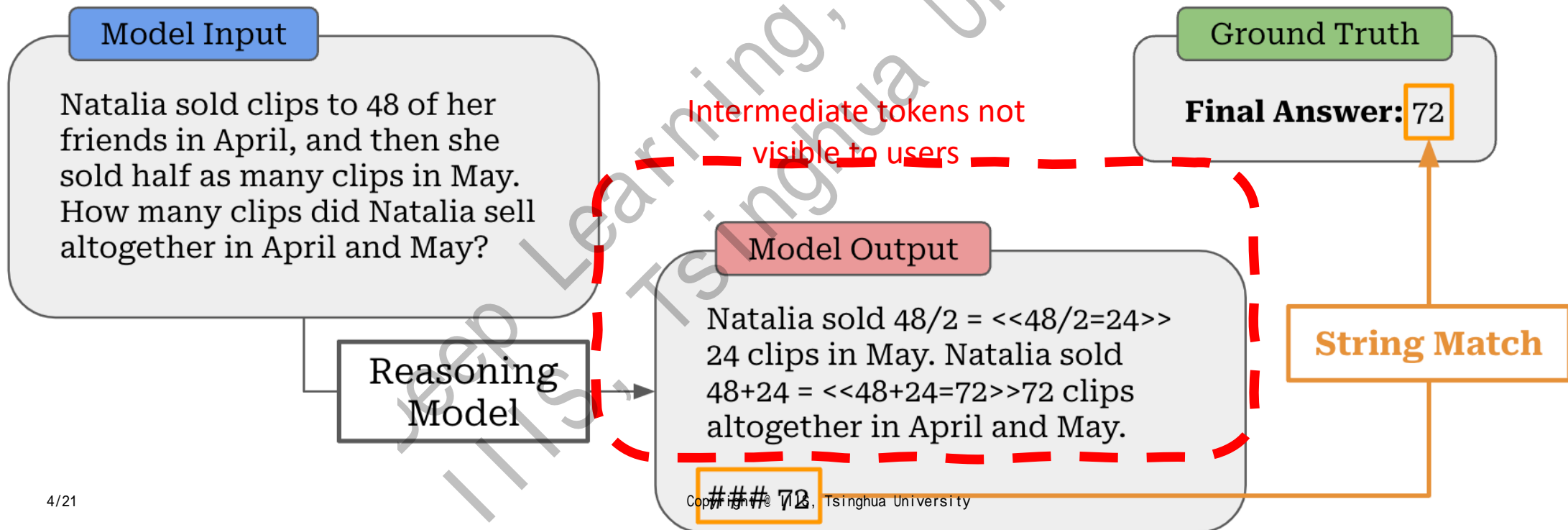
In-Context Learning in GPT-3 (Recap)

- In GPT-3, if more examples are in the context, the accuracy is higher
 - longer context \rightarrow higher accuracy



Inference-Time Enhancement

- OpenAI o1 model (2024)
 - Large Reasoning Model (LRM). An LLM “thinks” before giving a respond
 - Idea: we can allow model to **think** by having a **longer context**



Inference-Time Enhancement

- 0

Please calculate 24 using these 4 digits: 4, 4, 7, 7. Please only give the solution in 1 equation without any additional texts.

Reasoned about mathematical equation for 5 seconds ✓

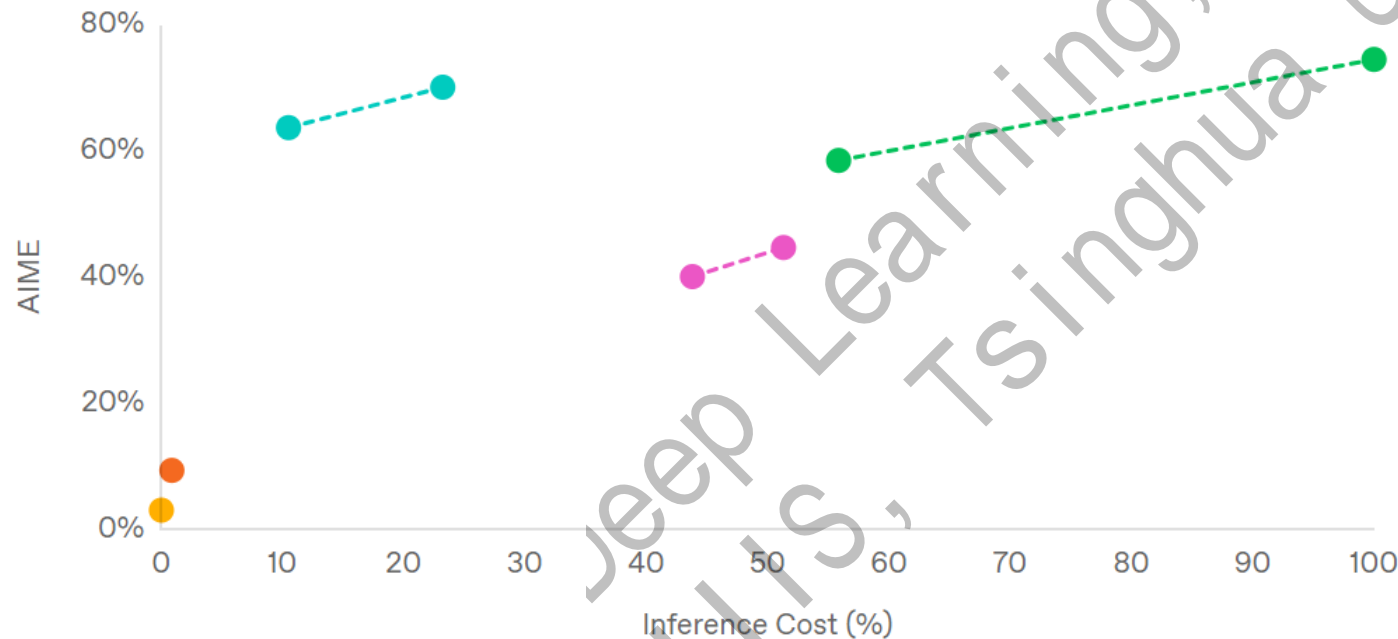
I need to find an equation using the digits 4, 4, 7, 7 that equals 24. I tried a few attempts like $(7 * 4) - (7 - 4) = 25$ or $(7 * 4) - 7/4 = 26.25$, but they didn't work. Then I thought about using all digits: $7 * (4 - 4/7)$. Checking - $4/7$ is 0.5714, so $4 - 4/7 = 3.4286$. Multiplying 3.4286 by 7 gives exactly 24. The correct equation is $7 * (4 - 4/7) = 24$.

$$7*(4-4/7)=24$$

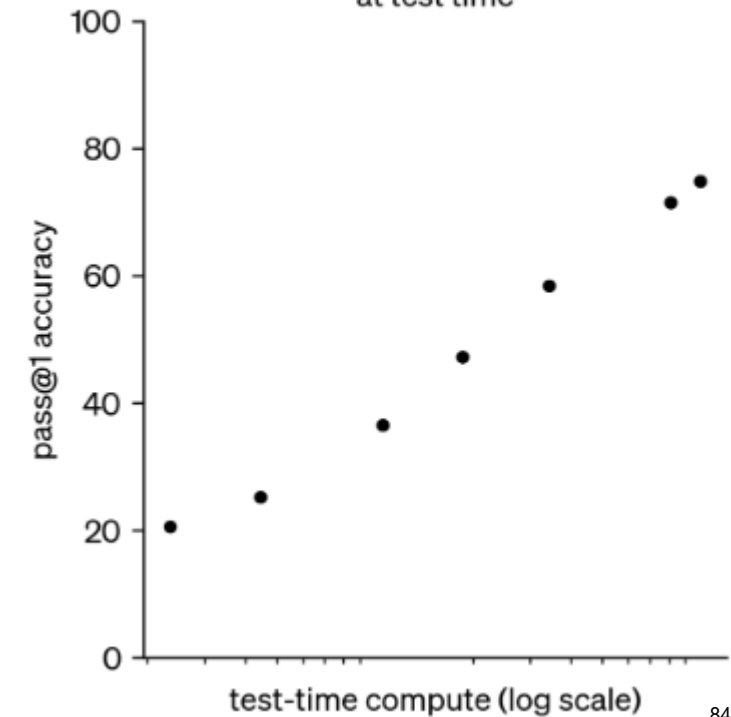
Inference-Time Enhancement

- Large reasoning models enable inference-time scaling

Math Performance vs Inference Cost



o1 AIME accuracy at test time



Inference-Time Enhancement

- How to derive a reasoning model?
 - Let's prompting it!
 - Chain-of-thought prompt (Google, 2022)

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Jason Wei

Xuezhi Wang

Dale Schuurmans

Maarten Bosma

Brian Ichter

Fei Xia

Ed H. Chi

Quoc V. Le

Denny Zhou

Google Research, Brain Team
{jasonwei,dennyzhou}@google.com

提示词
(prompt)

深度思考
(thinking)

输出结果
(output)

Inference-Time Enhancement

- How to derive a reasoning model?
 - Let's prompting it!
 - Chain-of-thought prompt (Google, 2022)
 - Tune the prompts to encourage the model think before responding

提示词
(prompt)

深度思考
(thinking)

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ❌

4/21

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

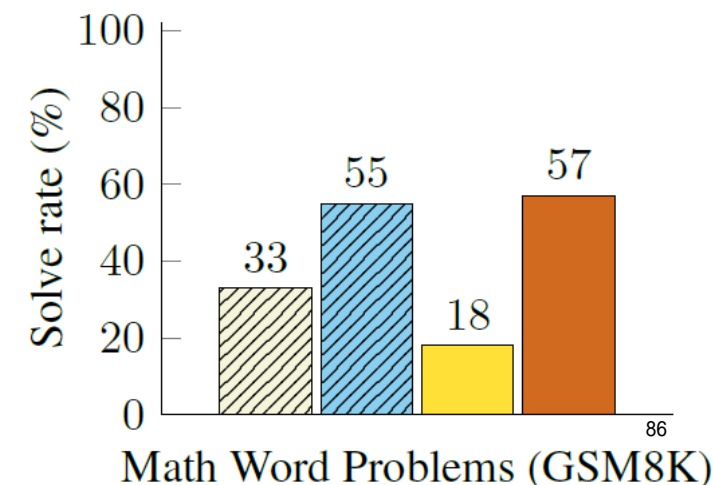
A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. $5 + 6 = 11$. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had $23 - 20 = 3$. They bought 6 more apples, so they have $3 + 6 = 9$. The answer is 9. ✅

- Finetuned GPT-3 175B
- Prior best
- PaLM 540B: standard prompting
- PaLM 540B: chain-of-thought prompting



Inference-Time Enhancement

- How to **train** a reasoning model?
 - Key challenge: how to obtain the **best “thinking” tokens** to train an LLM???
 - Supervised training?
 - There is no “correct” thinking tokens, we only care about answers
- **Reinforcement Learning**
 - Let the LLM self-explore the thinking tokens
 - Reward???
 - Remark: since the exploration space is huge, we must ensure the **reward is ACCURATE!!!**

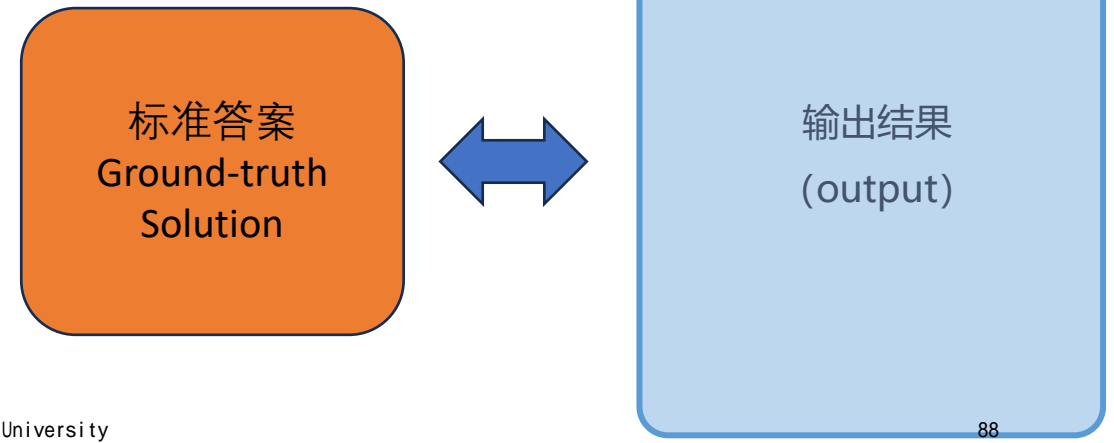
提示词
(prompt)

深度思考
(thinking)

输出结果
(output)

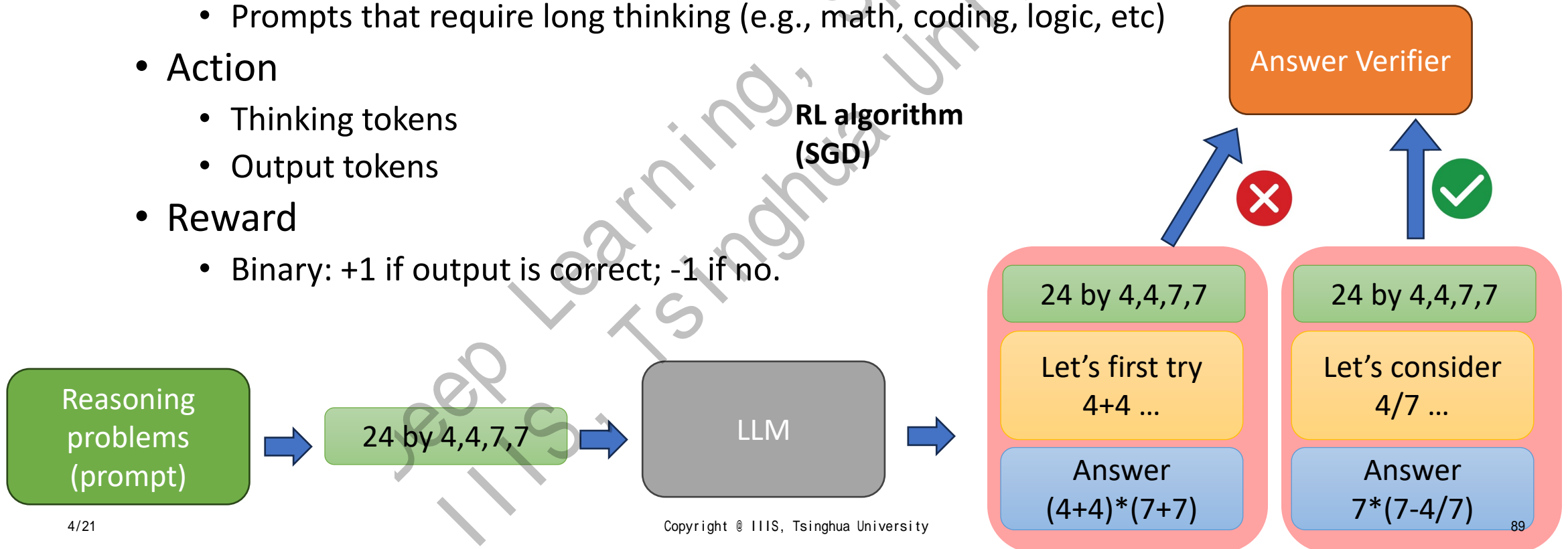
RL for Reasoning Model

- RL training for reasoning models
 - **Environment/Task**
 - Prompts that require long thinking (e.g., math, coding, logic, etc)
 - **Action**
 - Thinking tokens
 - Output tokens
 - **Reward**
 - Binary: +1 if output is correct; -1 if no.



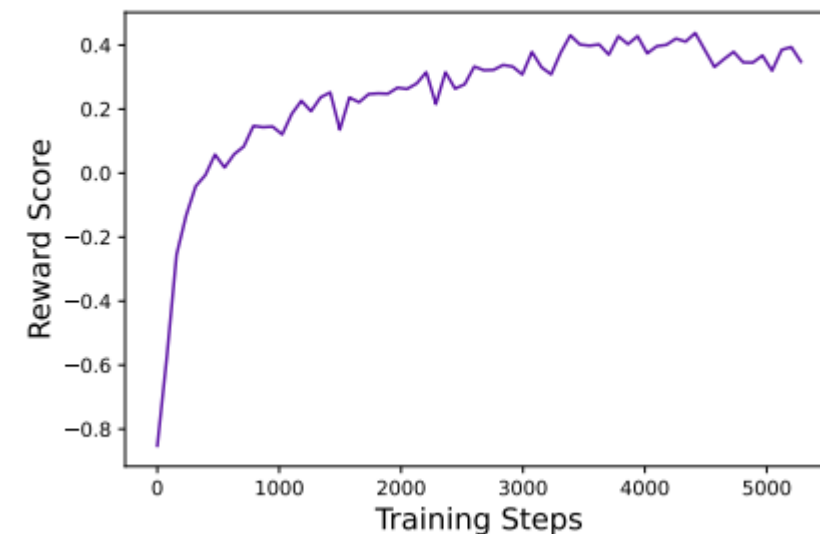
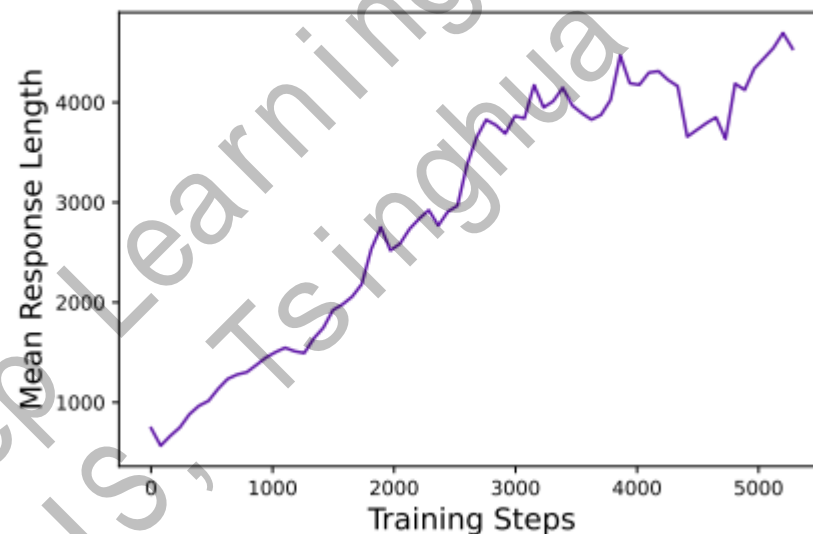
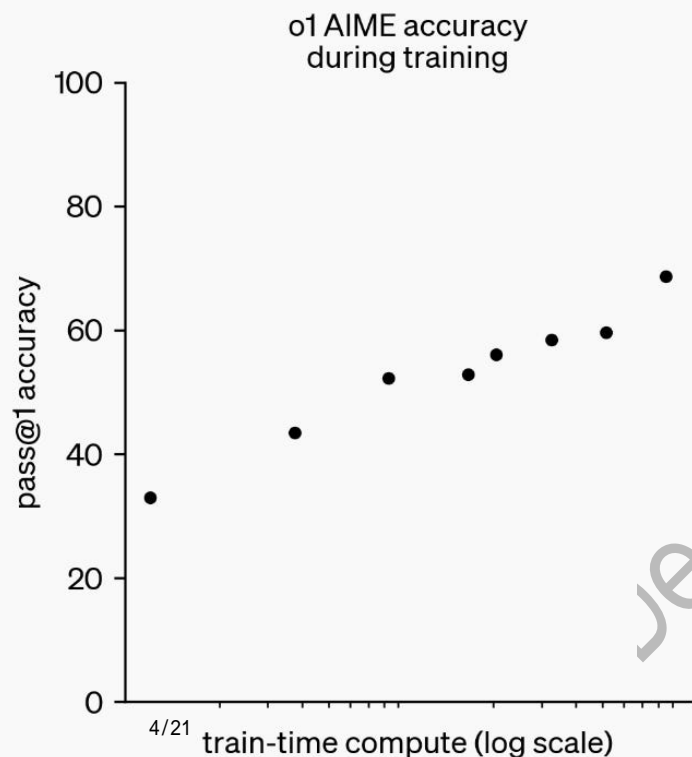
RL for Reasoning Model

- RL training for reasoning models
 - Environment/Task
 - Prompts that require long thinking (e.g., math, coding, logic, etc)
 - Action
 - Thinking tokens
 - Output tokens
 - Reward
 - Binary: +1 if output is correct; -1 if no.



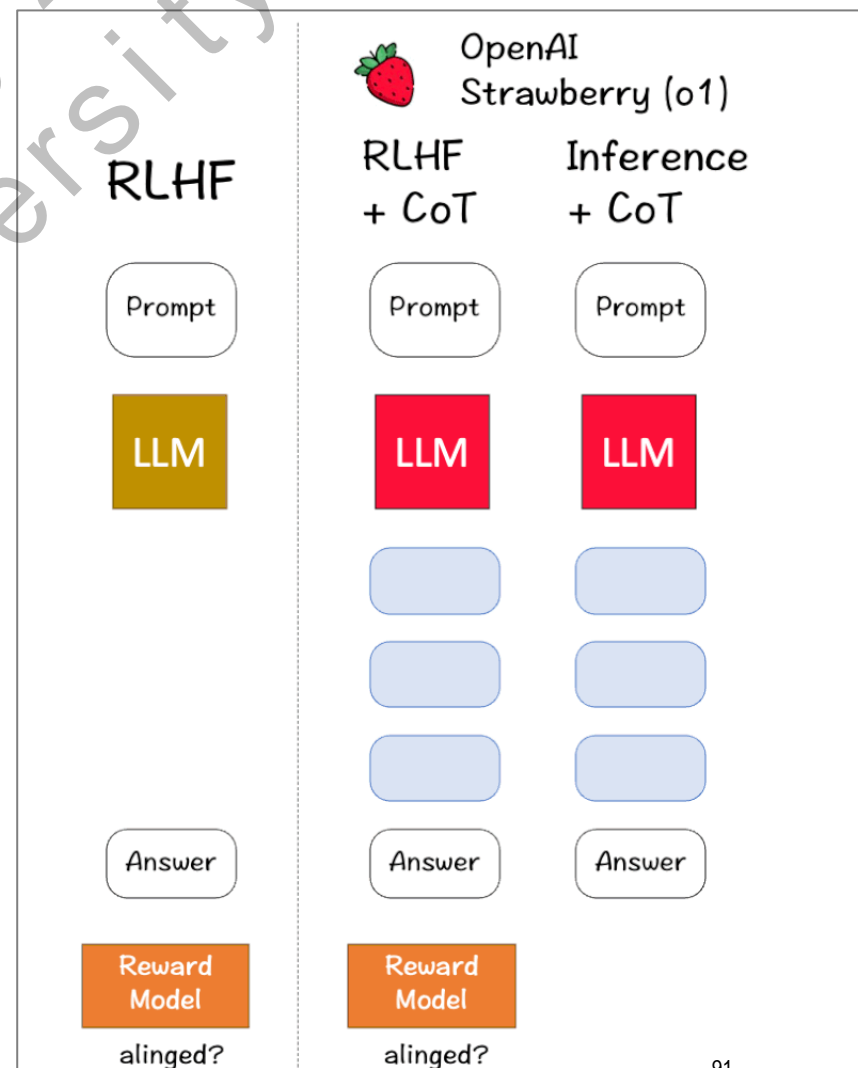
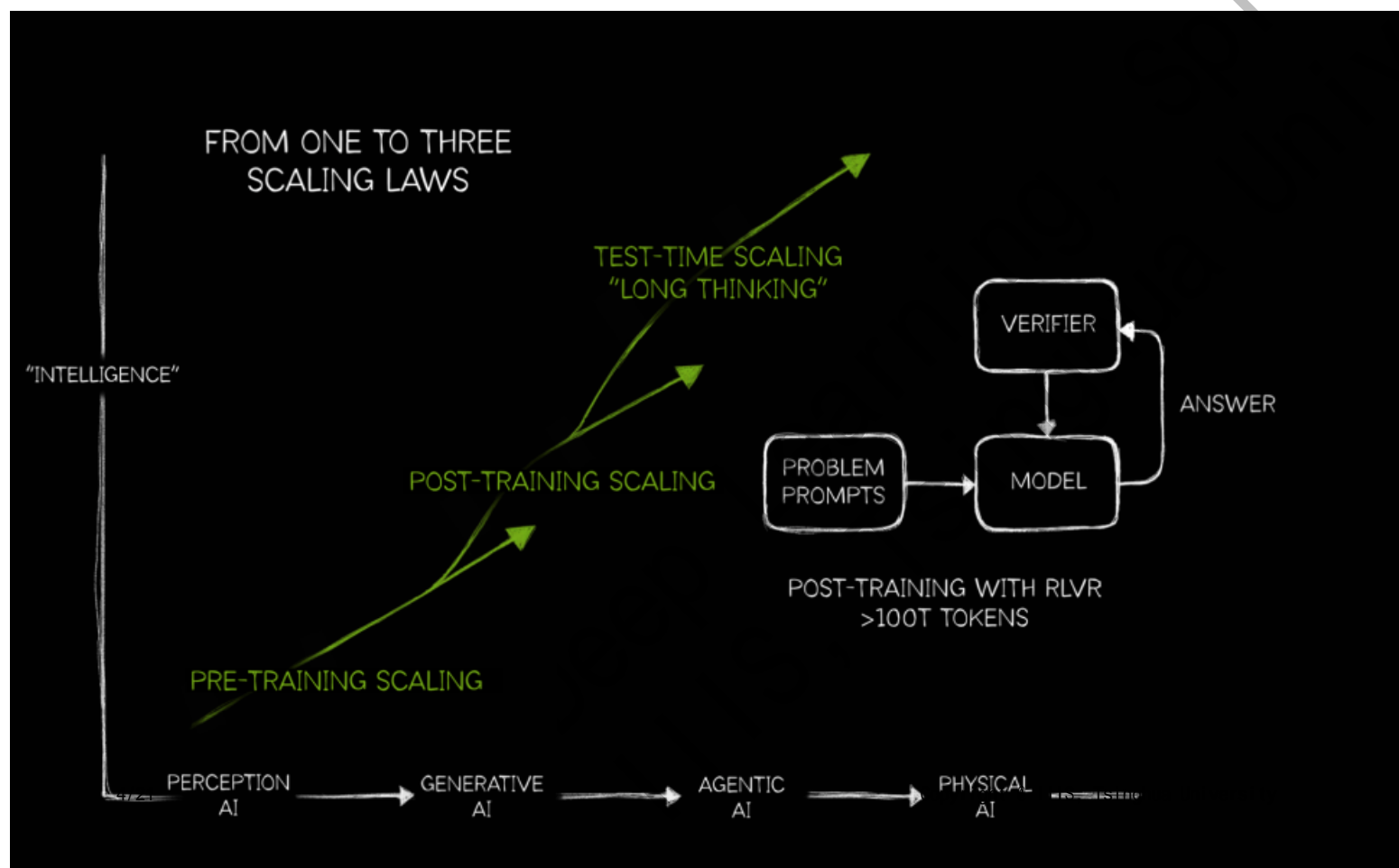
From RLHF to Post-Training Scaling

- RL is a new engine to scaling intelligence
 - Longer RL training leads to stronger reasoning performances



From RLHF to Post-Training Scaling

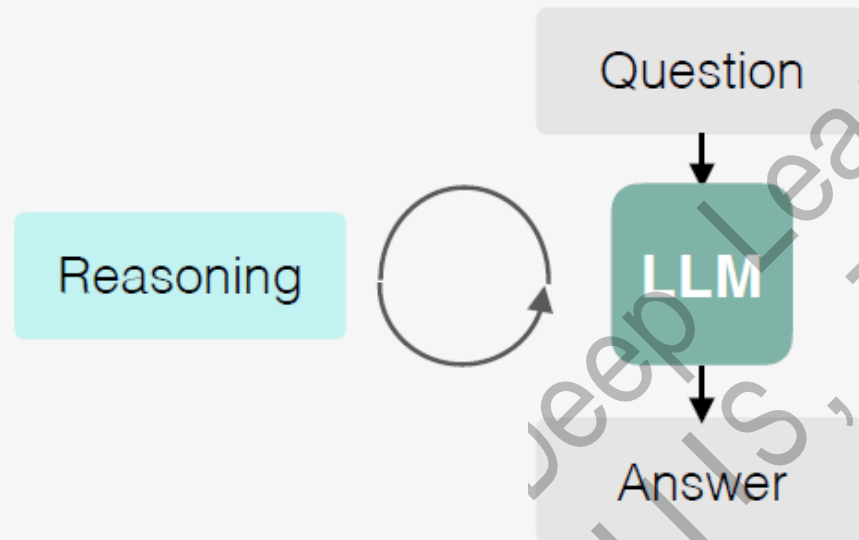
- RL is a new engine to scaling intelligence



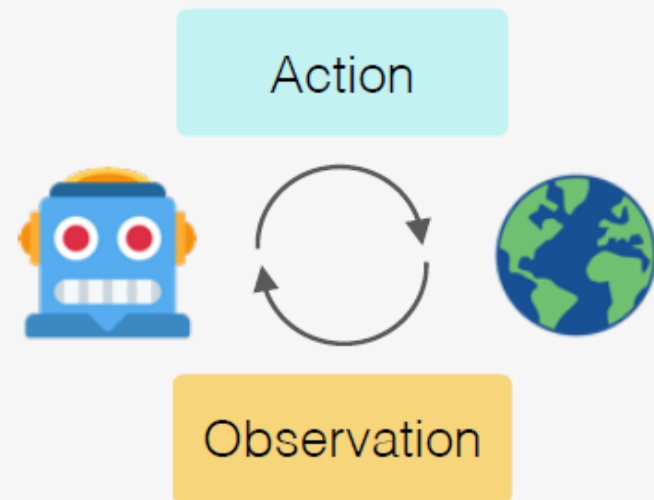
From Reasoning to Acting

- An LLM can also interact with a real environment
 - When an LLM interacts with the outside world, it is often called an “**LLM agent**”
 - We can also use **RL** to train an **LLM agent**

Reasoning (update internal belief)



Acting (obtain external feedback)



From Reasoning to Acting

- Expanding RAG to Multi-Turn **Deep Research** (OpenAI, 2025)

From Reasoning to Acting

- General-purpose LLM agent assistant (example from Manus.AI 2025)
 - A new form of smart software for the future



Conclusion

- Pretraining Transformers
 - Encoder-style pretraining (BERT): task-centric
 - Decoder-style pretraining (GPT): next-token prediction
- Scaling Law and Large Language Models
 - GPT leads to the scaling law and emergent capacities of LLMs
 - Larger model + better data + more compute → better models
- Post-training Language Models and More Applications
 - RLHF for a usable GPT (GPT-3 to ChatGPT)
 - Reasoning models think before answer
 - More thinking leads to better outputs (inference-time scaling)
 - RL training leads to better reasoning models (post-training scaling)
 - LLM can interact with the outside to become an LLM agent

Thanks

- Embrace the era of AGI!

Deep Learning, Spring 2025
IIIS, Tsinghua University