香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

COMP 5212
Machine Learning
Lecture 23

Large Language Models

Junxian He
Nov 28, 2024

Final Exam

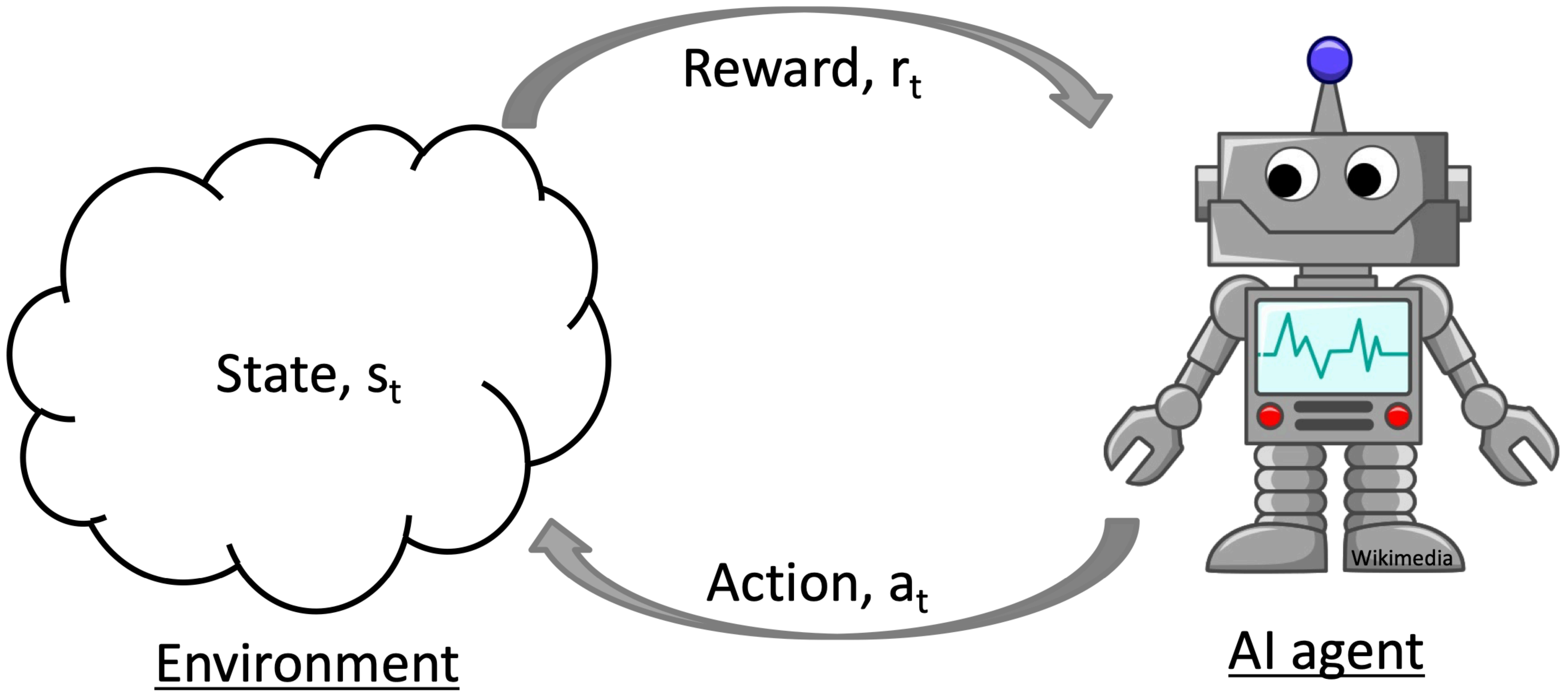
1. Two, double-sided A4-size cheatsheets
2. 2-hour duration
3. Contents cover both before mid-term and after mid-term, while emphasizing more on after mid-term
4. Format similar to mid-term exam, mixed multi-choice and open-ended questions
5. 14/12, 1pm - 3pm, Lecture Theater A

Will make more formal announcement on Canvas

Kaggle Project

1. TA will make an announcement on some suggestions this week

Recap: Reinforcement Learning



- Maximize reward (rather than learn reward) Supervised training is like imitation
- Inputs are not iid – state & action depends on past

Language Models

Probability of Sequences

Probability of multiple random variables:

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(x_i \,|\, x_{1:i-1})$$

Probability of language:

$$\begin{aligned} p(\text{the, mouse, ate, the, cheese}) = \; & p(\text{the}) \\ & p(\text{mouse} \mid \text{the}) \\ & p(\text{ate} \mid \text{the, mouse}) \\ & p(\text{the} \mid \text{the, mouse, ate}) \\ & p(\text{cheese} \mid \text{the, mouse, ate, the}). \end{aligned}$$

Autoregressive language models

Autoregressive Language Models

$$p(\text{the, mouse, ate, the, cheese}) = p(\text{the})$$
$$p(\text{mouse} \mid \text{the})$$
$$p(\text{ate} \mid \text{the, mouse})$$
$$p(\text{the} \mid \text{the, mouse, ate})$$
$$p(\text{cheese} \mid \text{the, mouse, ate, the}).$$

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(\underbrace{x_i}_{\text{Next Word}} \mid \underbrace{x_{1:i-1}}_{\text{Context}})$$

Autoregressive Language Models

$$
\begin{aligned}
p(\text{the, mouse, ate, the, cheese}) = {} & p(\text{the}) \\
& p(\text{mouse} \mid \text{the}) \\
& p(\text{ate} \mid \text{the, mouse}) \\
& p(\text{the} \mid \text{the, mouse, ate}) \\
& p(\text{cheese} \mid \text{the, mouse, ate, the}).
\end{aligned}
$$

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(x_i \mid x_{1:i-1})$$

Learning a language model is to learn these conditional probabilities, for any language sequence

Autoregressive Language Models

$$
\begin{aligned}
p(\text{the, mouse, ate, the, cheese}) = & \; p(\text{the}) \\
& \; p(\text{mouse} \mid \text{the}) \\
& \; p(\text{ate} \mid \text{the, mouse}) \\
& \; p(\text{the} \mid \text{the, mouse, ate}) \\
& \; p(\text{cheese} \mid \text{the, mouse, ate, the}).
\end{aligned}
$$

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(x_i \,|\, x_{1:i-1})$$

Given a dataset, how to find these probabilities?

Maximum Likelihood Estimation

Count-based Language Models

Count the frequency and divide

$$p(x_i \mid x_{1:i-1}) = \frac{c(x_{1:i})}{c(x_{1:i-1})}$$

There are infinite number of parameters for language

We may see long sequences only once, counting becomes meaningless

n-gram Language Models

Next token probability only depends on the previous n-1 words

Unigram LM:

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(x_i)$$

Each token is independent

Bigram LM:

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(x_i \,|\, x_{i-1})$$

Markov assumption?

Generally for n-gram LM:

$$p(x_1, x_2, \ldots, x_I) = \prod_{i=1}^{I} p(x_i \,|\, x_{i-n+1:i-1})$$

Similar to n-th order HMM?

Is HMM autoregressive LM?

Parameter Estimation for n-gram LM

Count-based:

$$p(x_i \mid x_{i-n+1:i-1}) = \frac{c(x_{i-n+1:i})}{c(x_{i-n+1:i-1})}$$

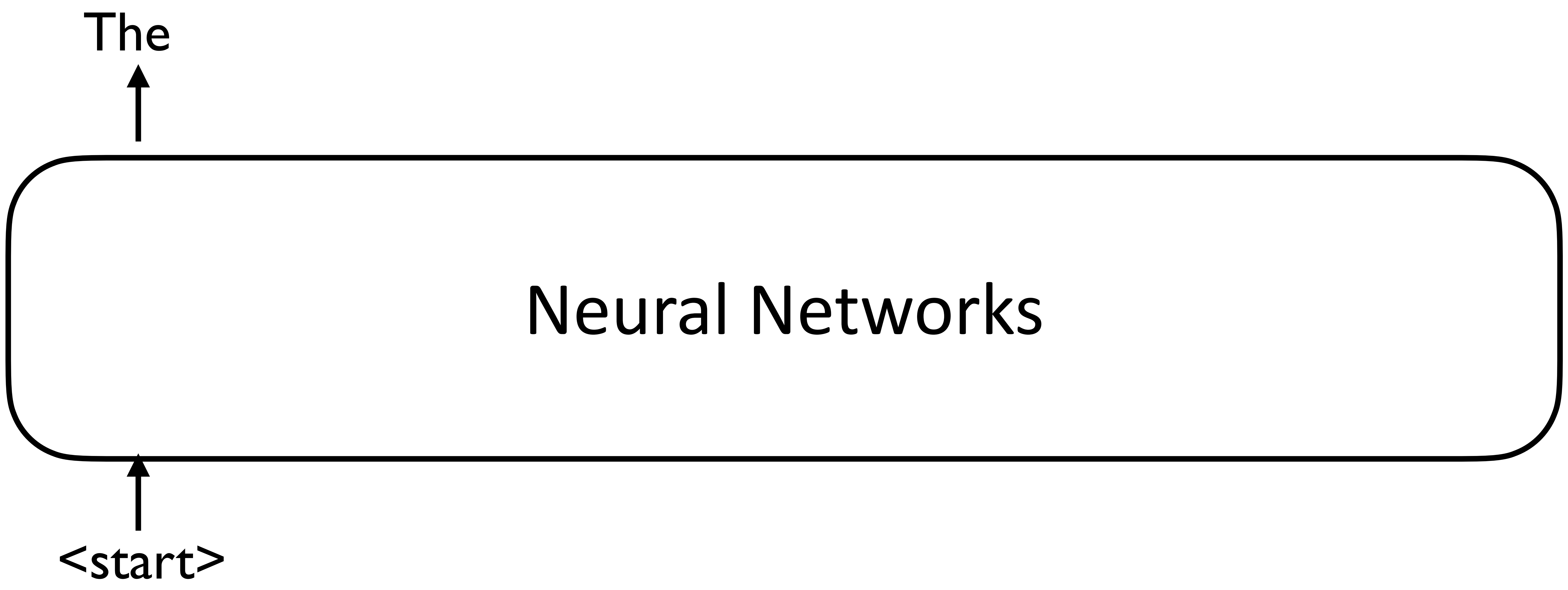
Number of parameters decreases, but flexibility decreases as well

Traditionally, we directly compute this probability, but neural language models use neural networks to compute the probability

Neural Language Models

Neural language models are typically autoregressive

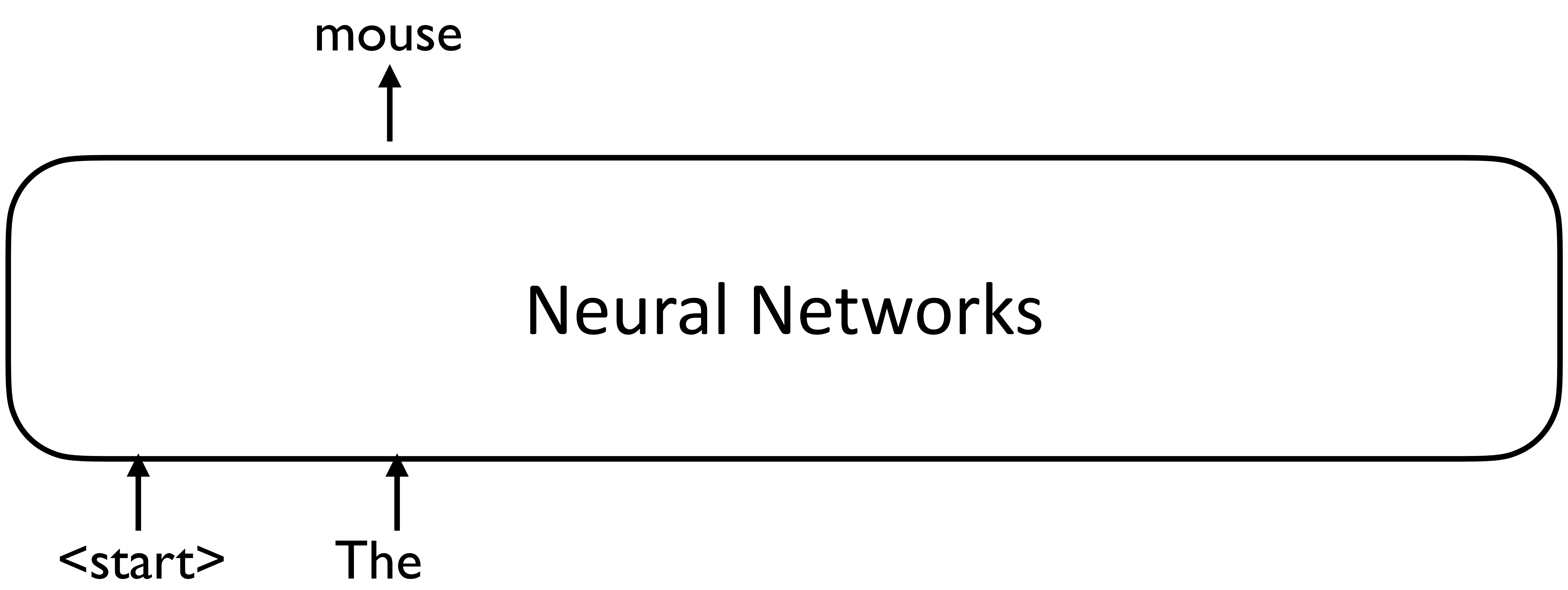
Data: "The mouse ate the cheese ."



Neural Language Models

Neural language models are typically autoregressive

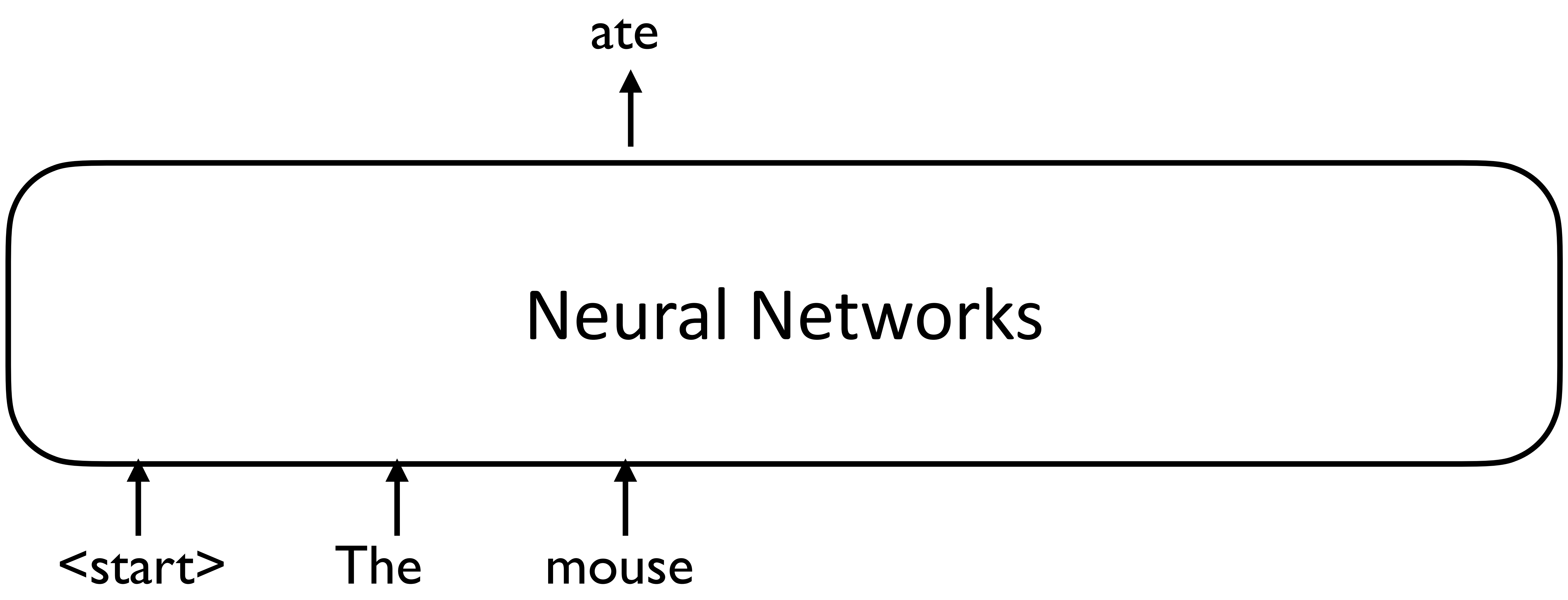
Data: "The mouse ate the cheese ."



Neural Language Models

Neural language models are typically autoregressive

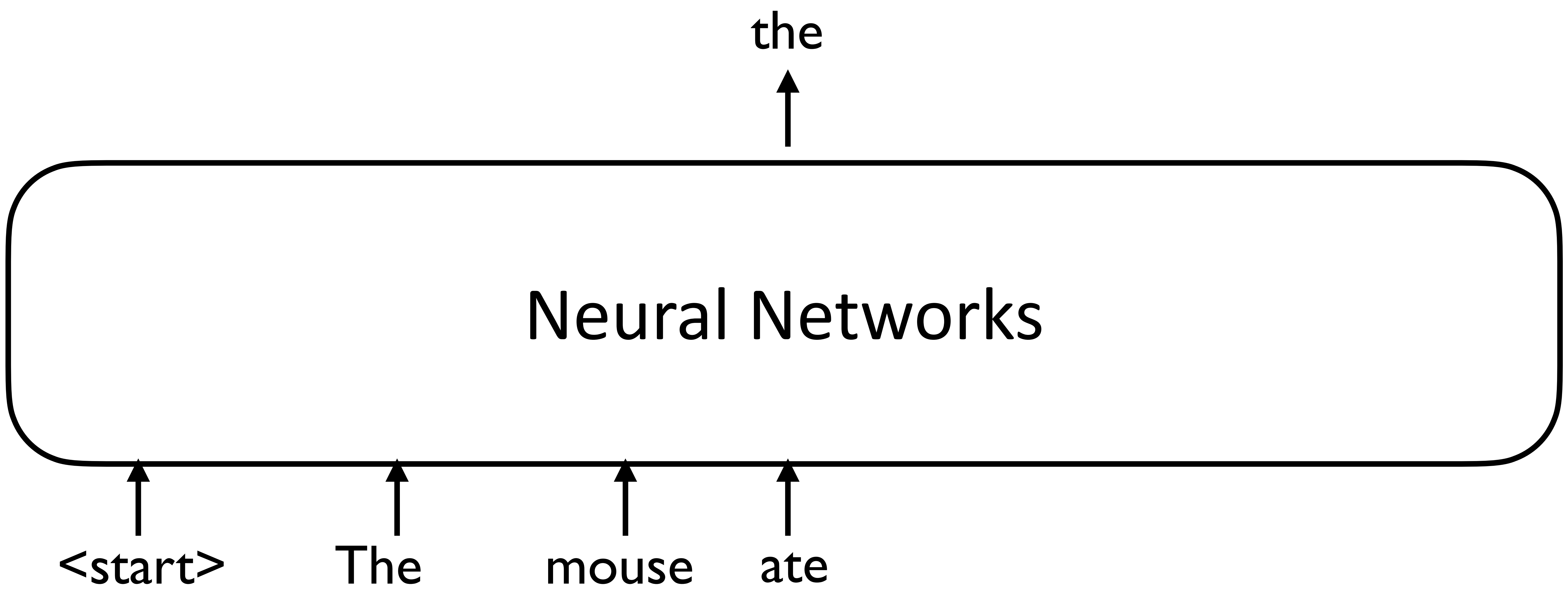
Data: "The mouse ate the cheese ."



Neural Language Models

Neural language models are typically autoregressive

Data: "The mouse ate the cheese ."

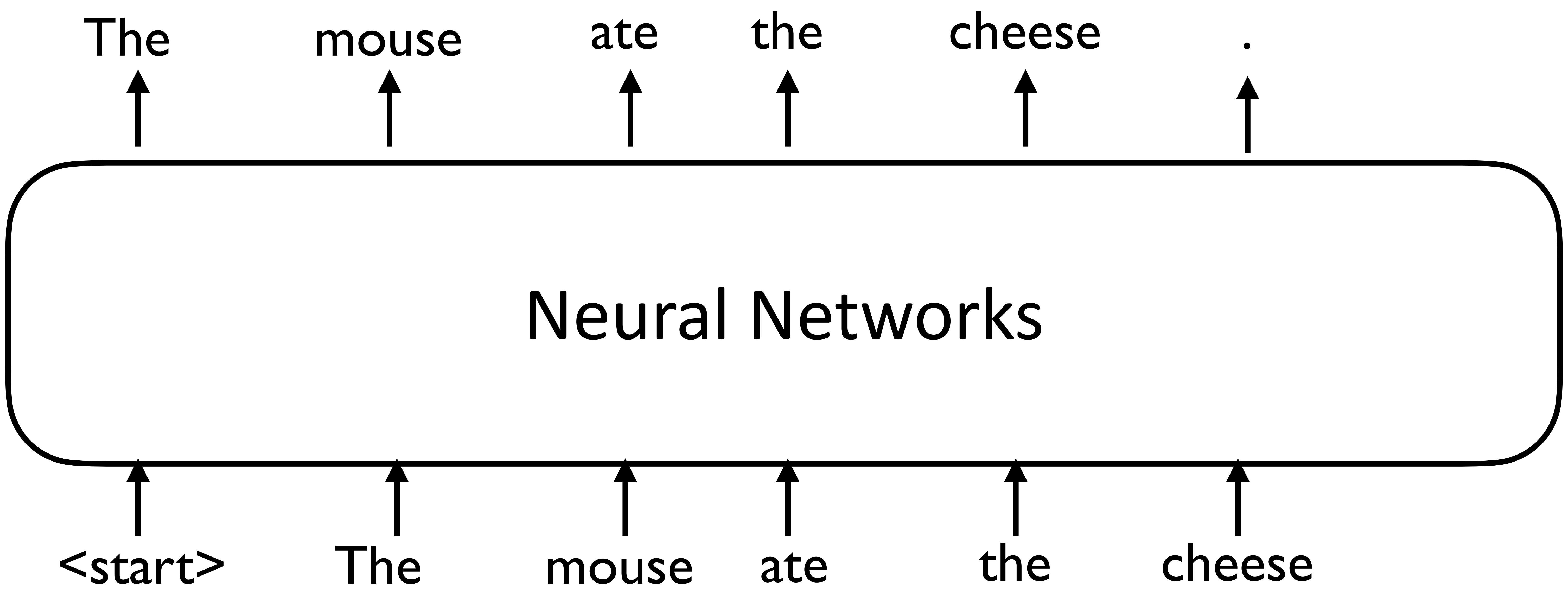


We can compute the loss on every token in parallel

Neural Language Models

Neural language models are typically autoregressive

Data: "The mouse ate the cheese ."



Each prediction only sees the inputs on its left

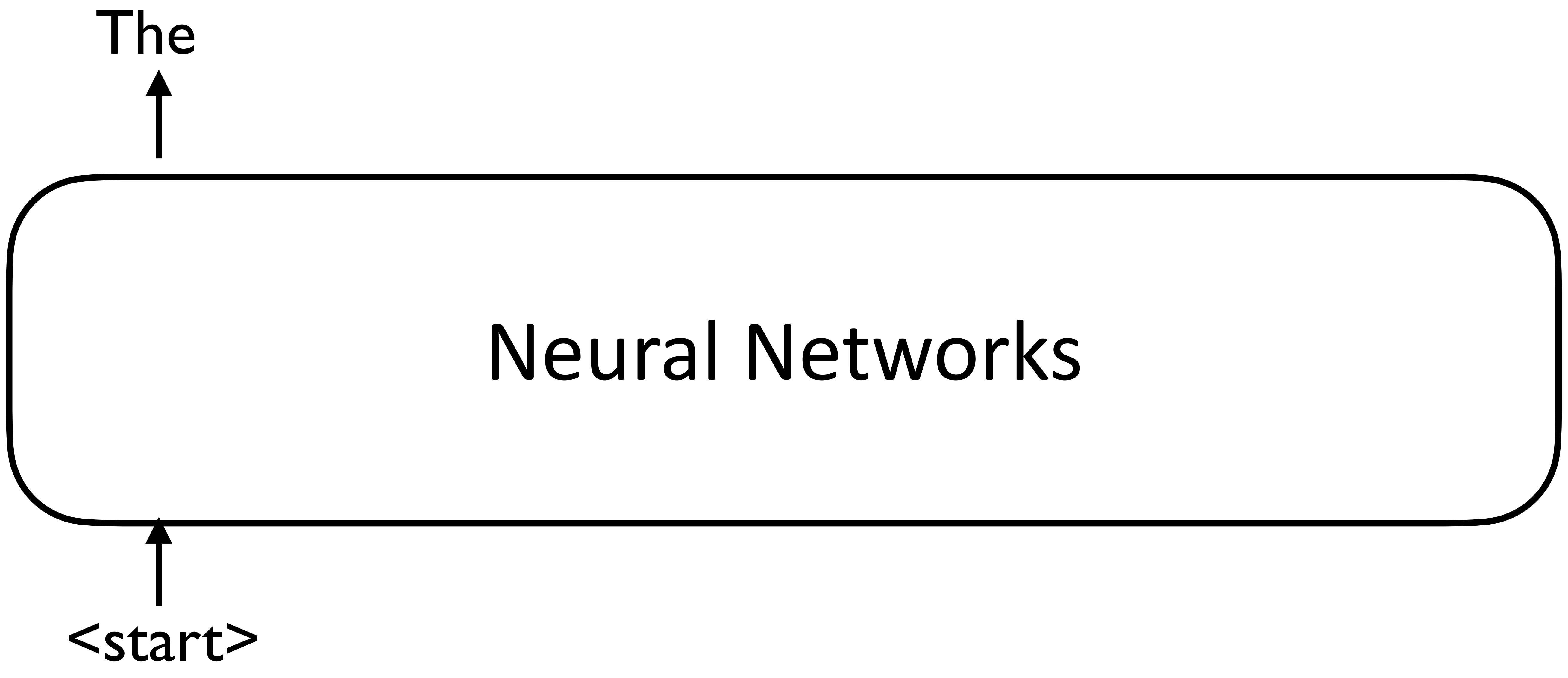
Neural Language Models

Is language modeling MLE? ✅

Are language models generative models? ✅

Can we compute p(x) given x? Can we sample new x? ✅

At inference time, to generate:



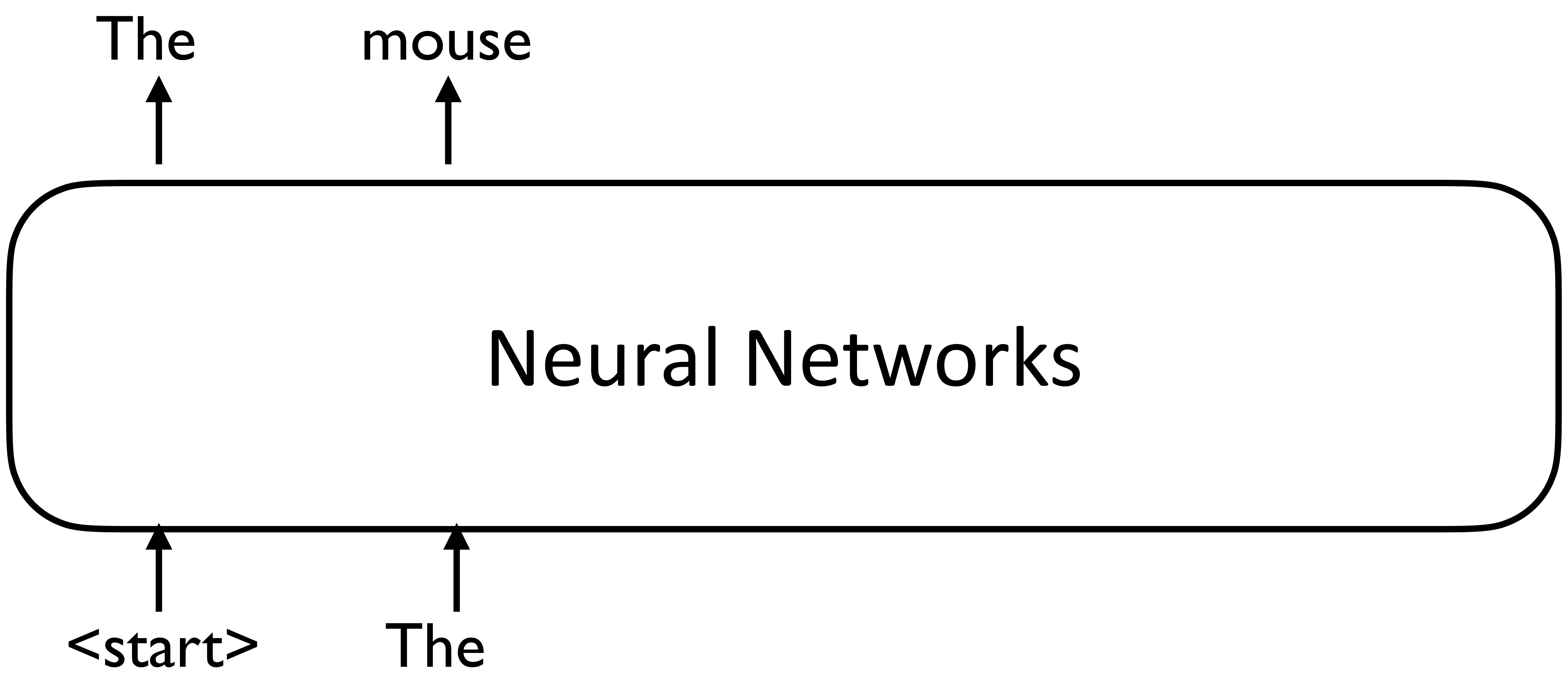
Neural Language Models

Is language modeling MLE? ✅

Are language models generative models? ✅

Can we compute p(x) given x? Can we sample new x? ✅

At inference time, to generate:



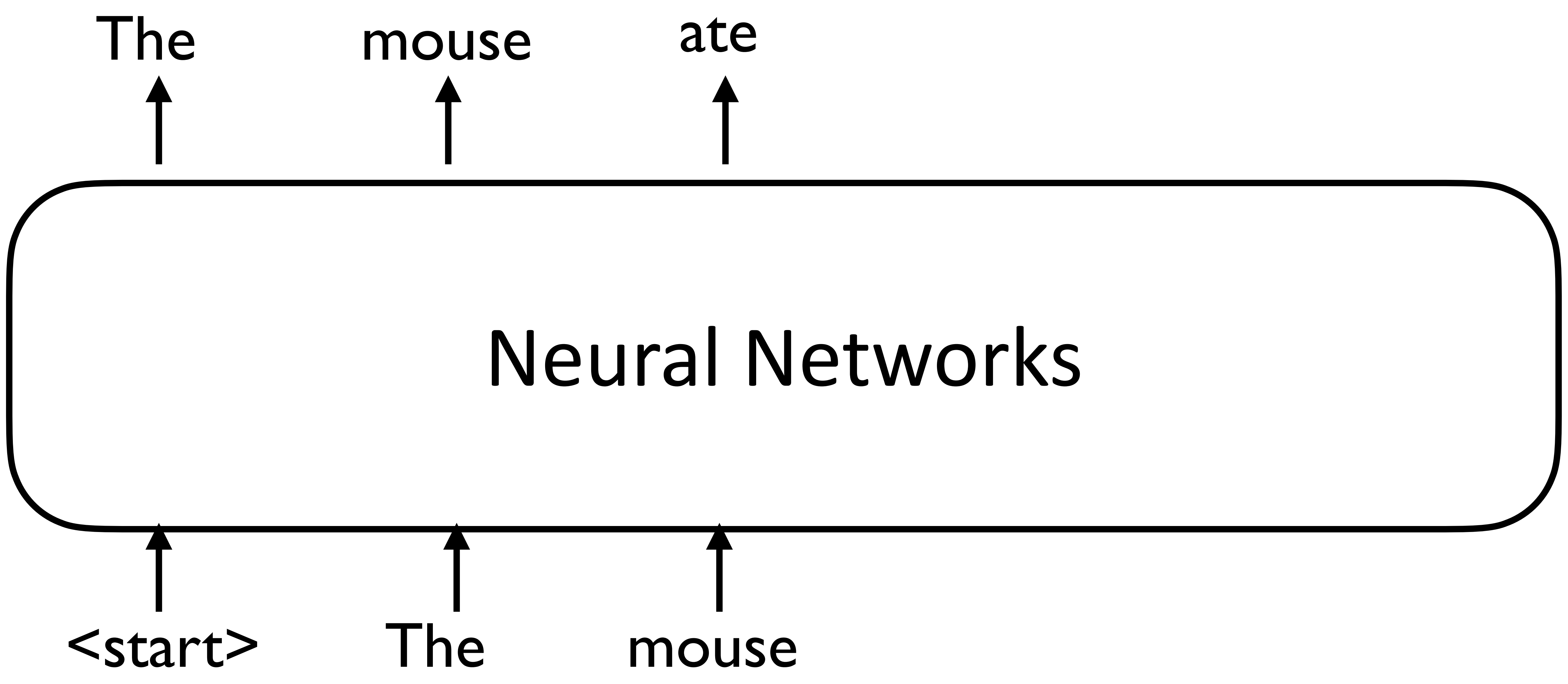
Neural Language Models

Is language modeling MLE? ✅

Are language models generative models? ✅

Can we compute p(x) given x? Can we sample new x? ✅

At inference time, to generate:



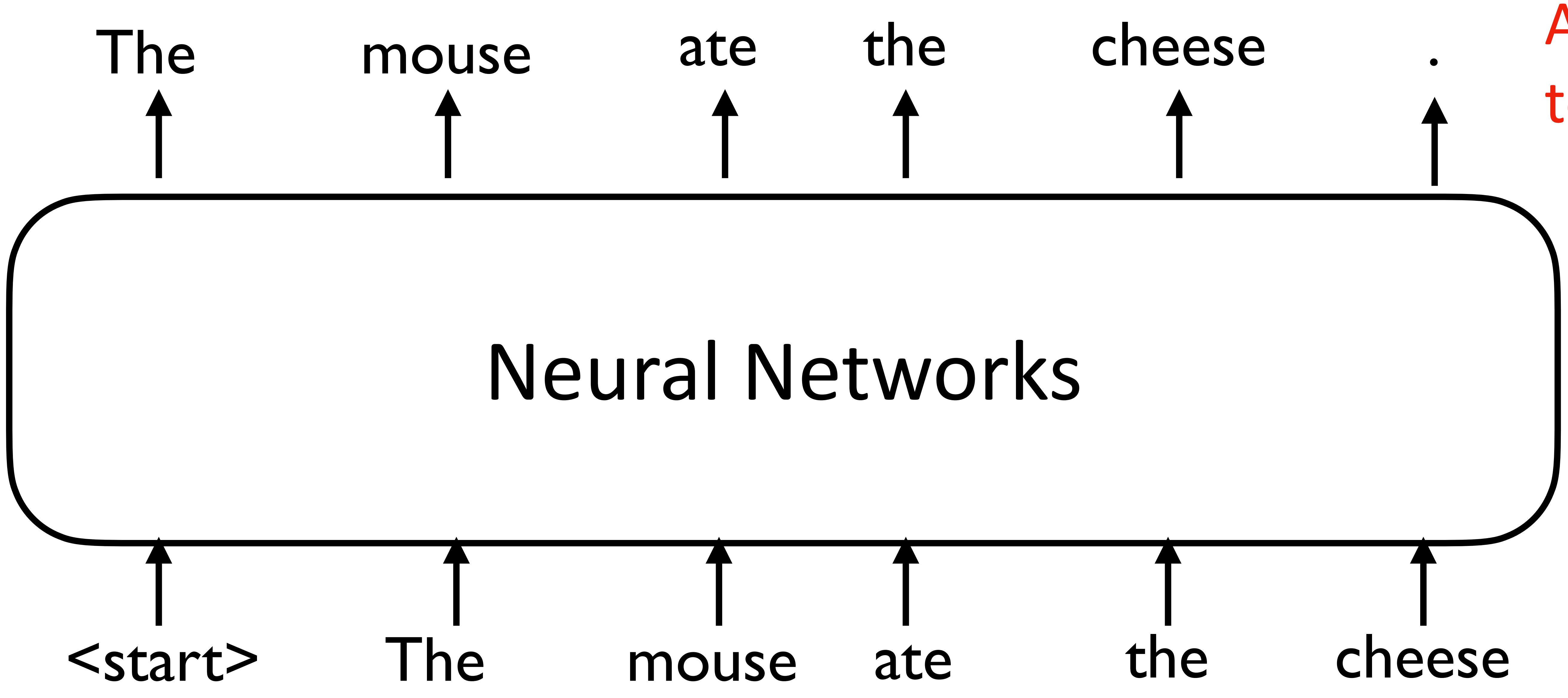
Neural Language Models

Is language modeling MLE? ✅

Are language models generative models? ✅

Can we compute p(x) given x? Can we sample new x? ✅

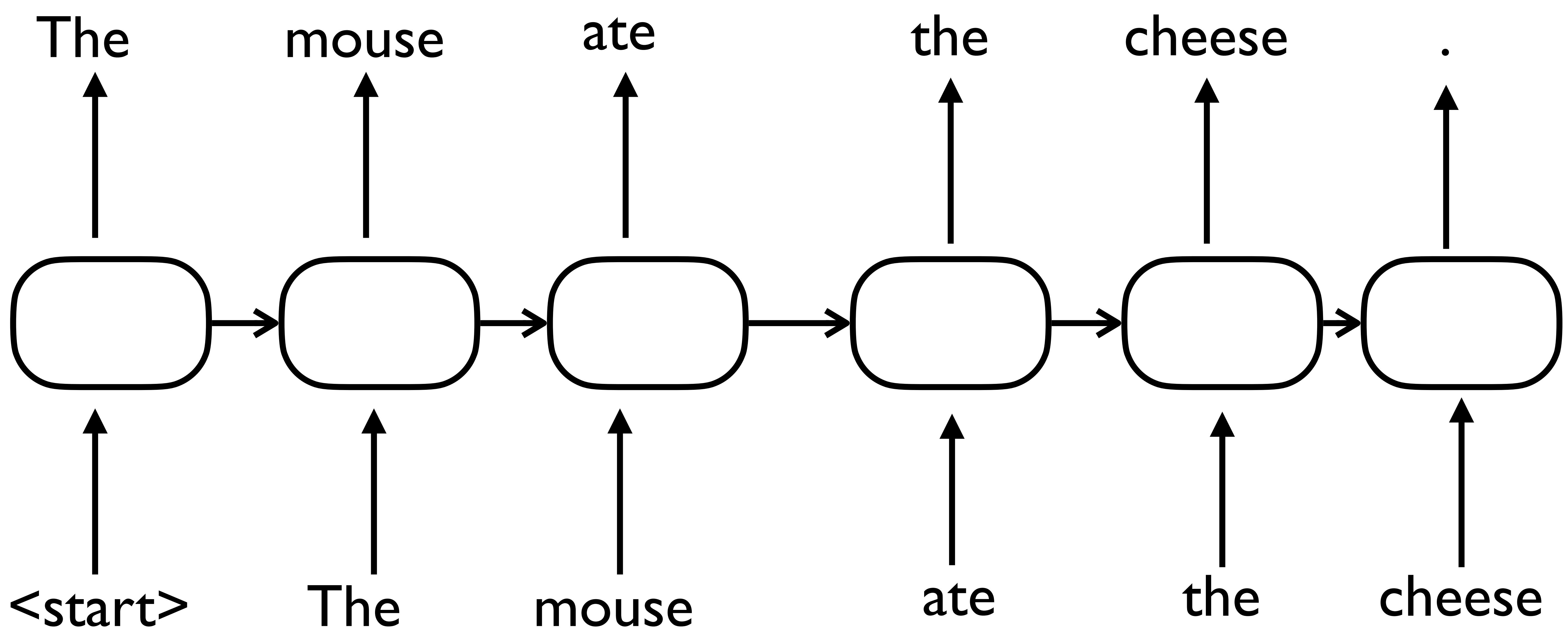
At inference time, to generate:



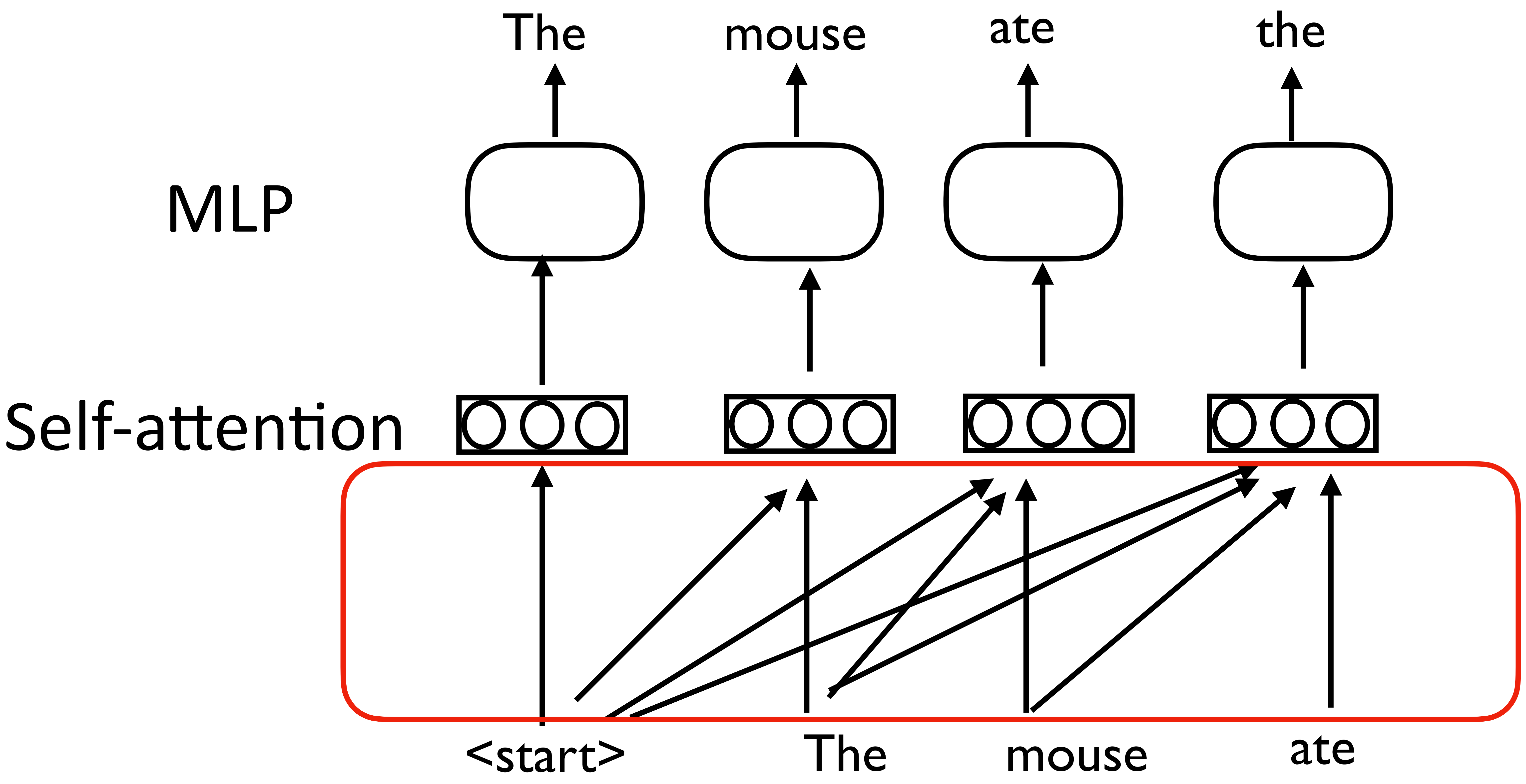
Autoregressive generation has to generate token by token

Cann't parallelize, efficiency of autoregressive decoding is still an important research topic

RNN Language Models



Transformer Language Models

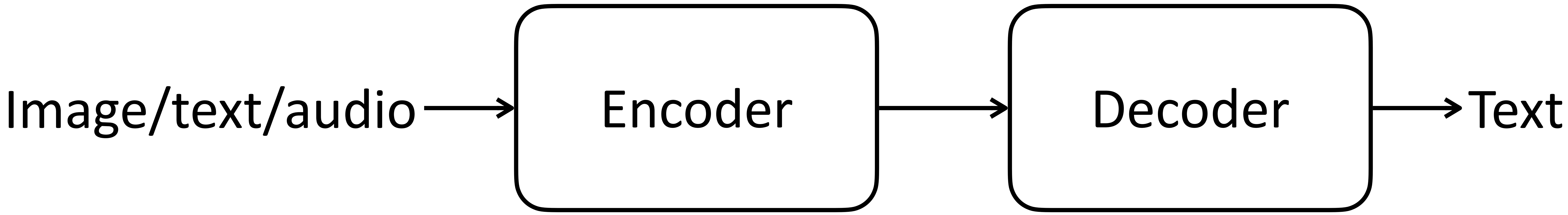


Self-attention only attends to the tokens on the left (masked attention)

Neural Language Models

Language model is the fundamental block to model language distribution p(x)

For a long time, to solve specific tasks:

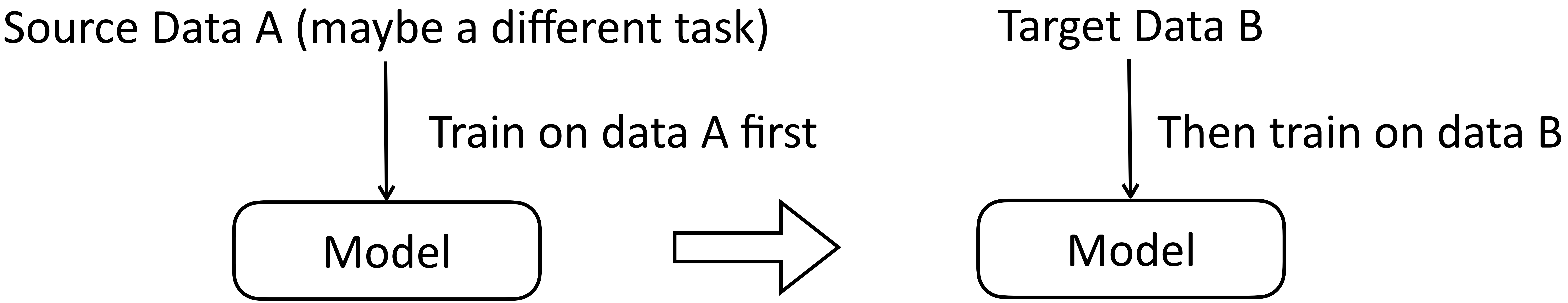


When we have a better arch/training
for LM, we can have a better decoder

Not long ago, some people think purely language models is useless because it does not directly address tasks, and LM performance may not transfer to downstream tasks

Some impactful papers are rejected by such reviewers (e.g. transformer-XL)

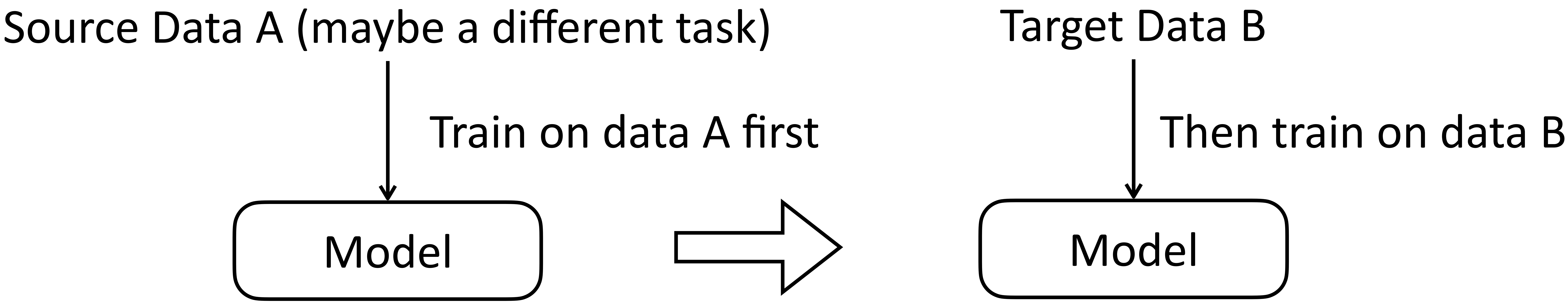
Pretraining



Classically, this is transfer Learning

It is now called pretraining because of the scale of A

Pretraining

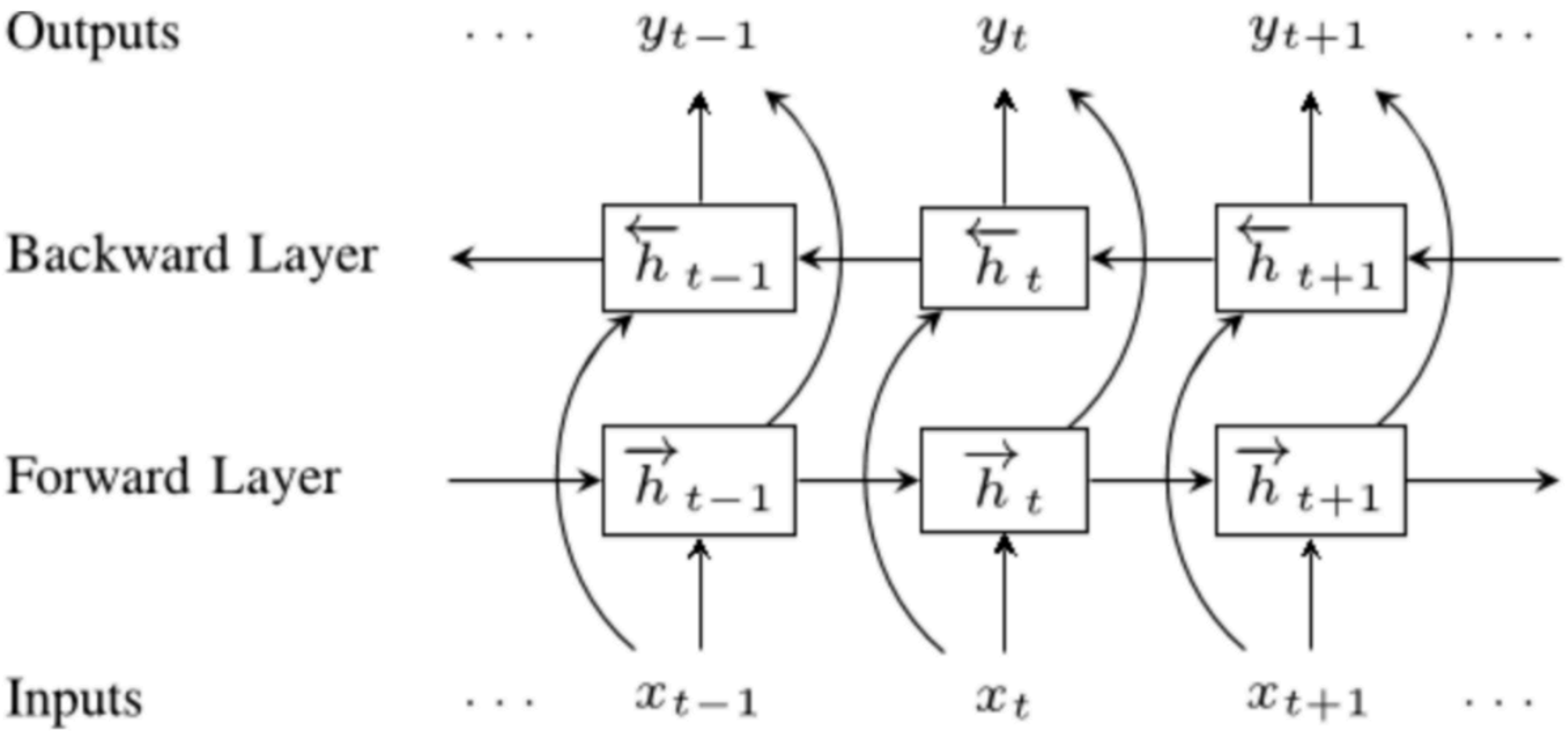


For supervised training, data A is often limited

How can we find large-scale data A to train?

ELMO

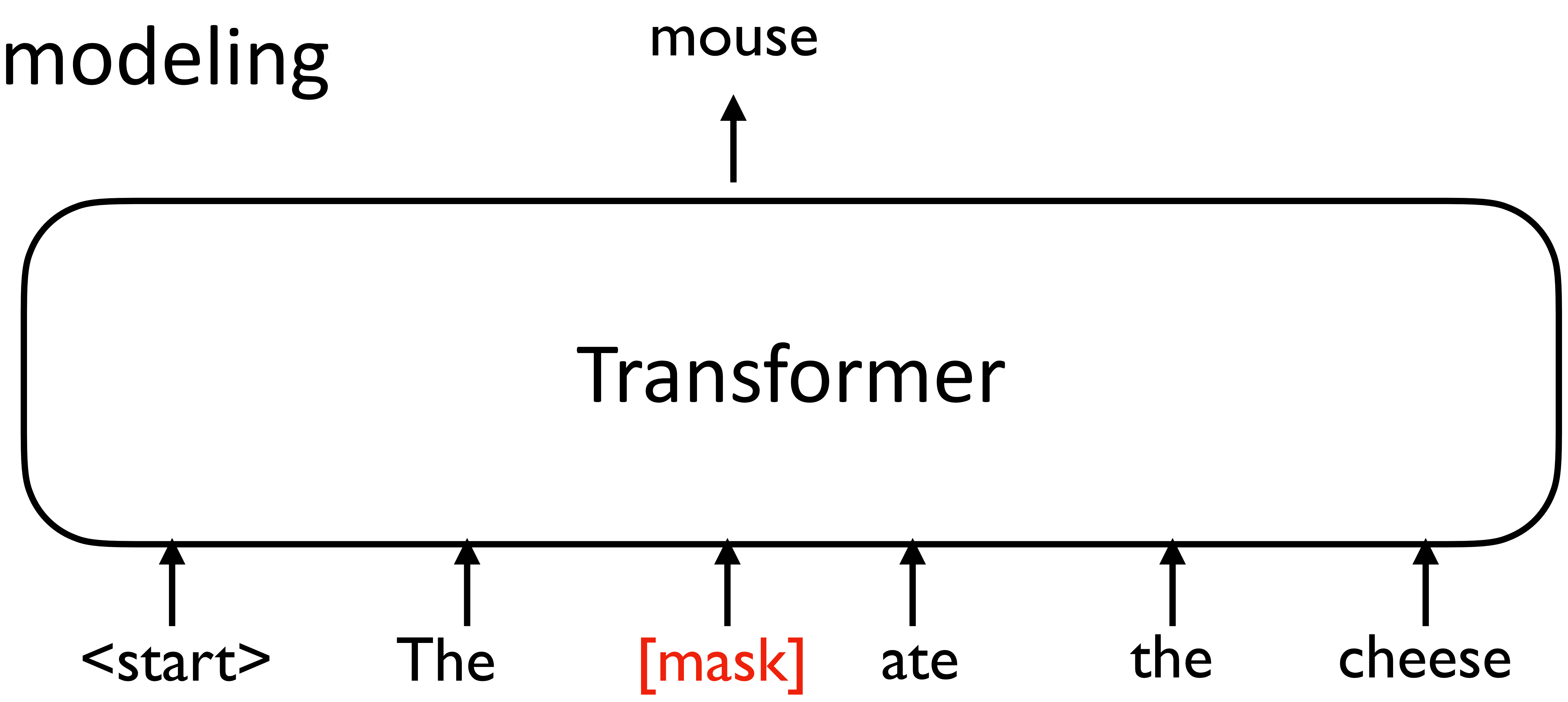
Self-supervised Pretraining   Construct supervision from unannotated data



Peters et al. Deep contextualized word representations. NAACL 2018

BERT

Mask language modeling



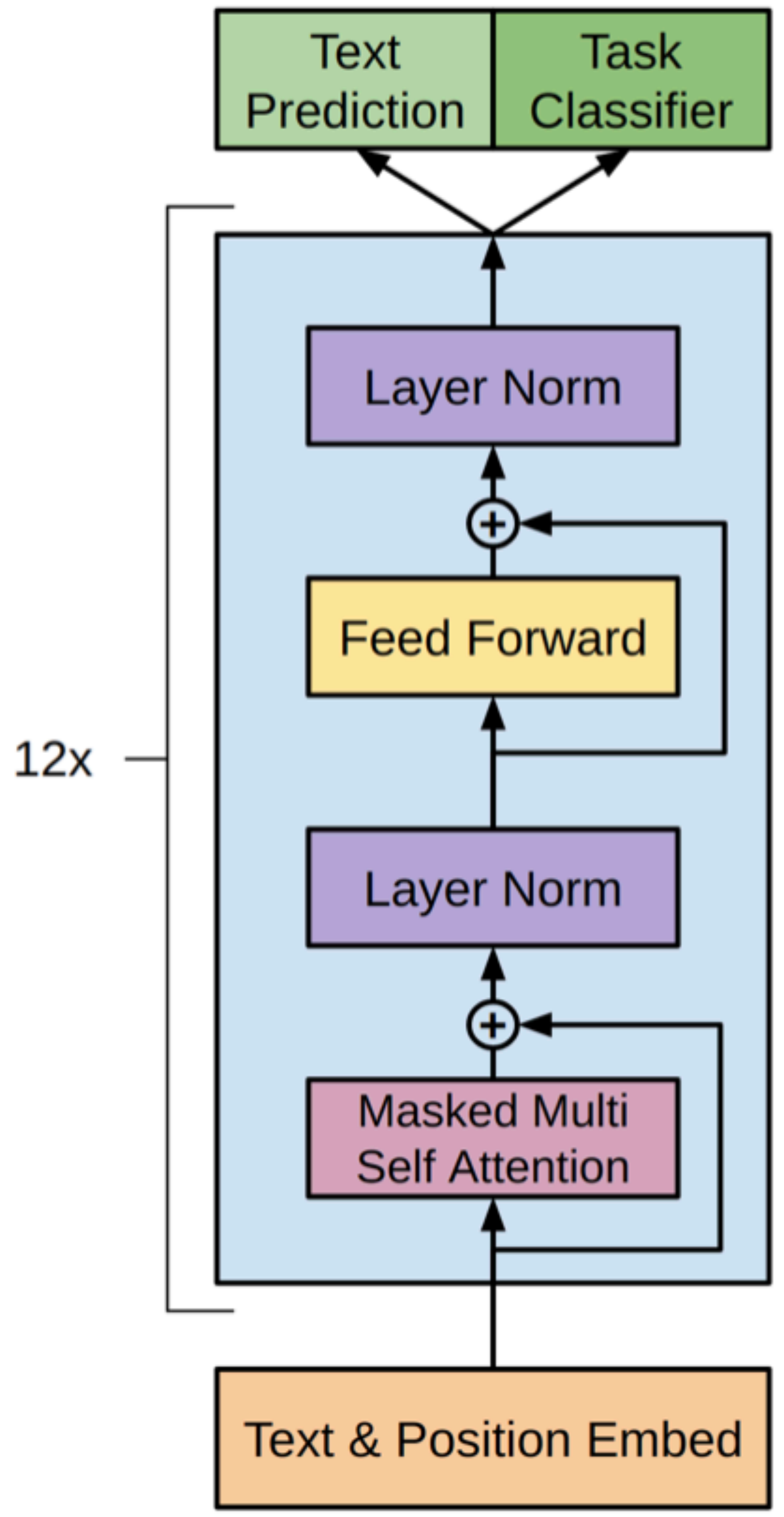
Construct a synthetic task from raw text only

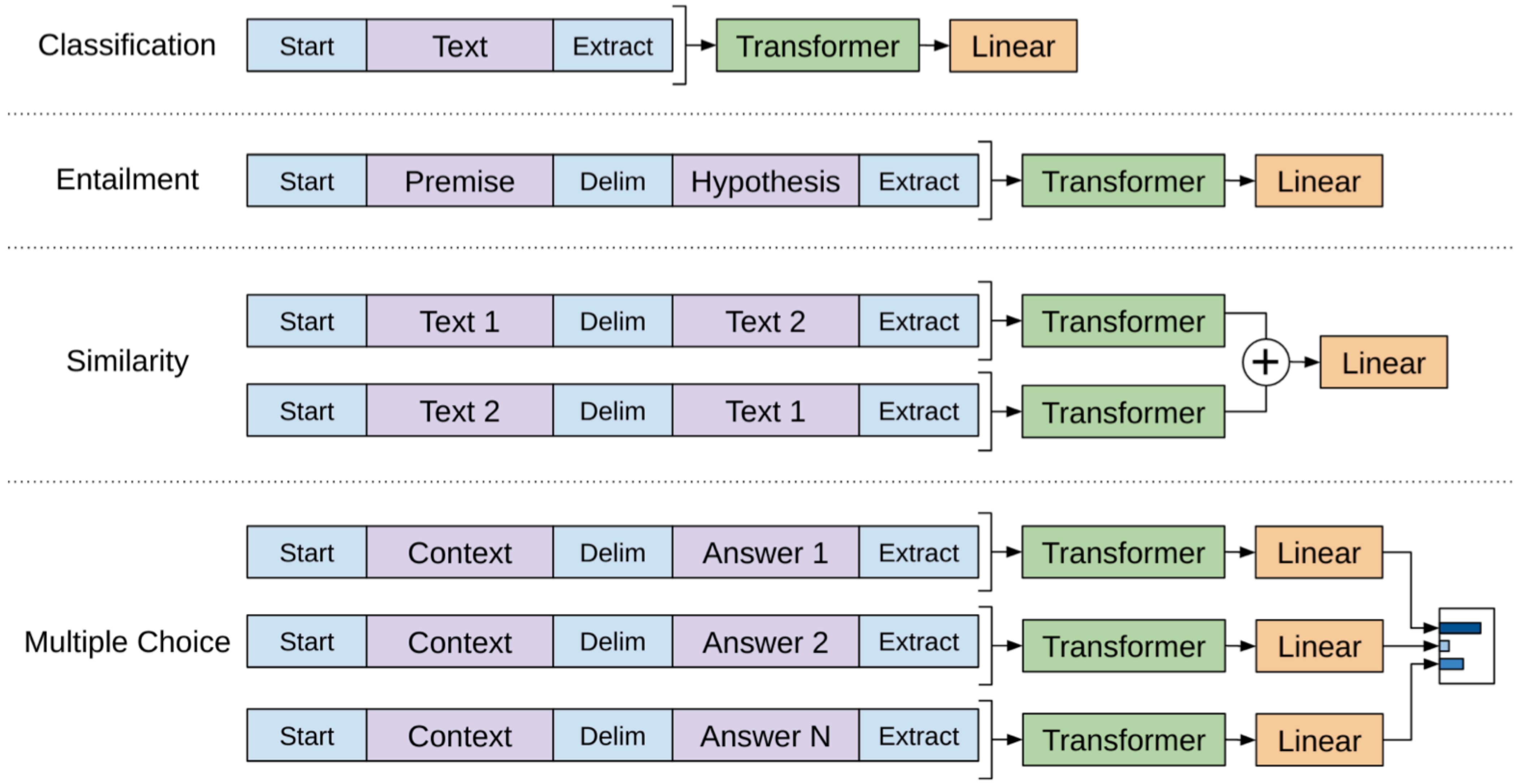
Can be made very large-scale

Is Bert a language model? Is it a generative model?

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL 2019.

Generative Pre-Training (GPT)





Radford et al. Improving Language Understanding by Generative Pre-Training. 2018

Is Next Token Prediction Useful?

Ok, language modeling can be used as pretraining, but is a language model itself useful for some tasks directly?

In the late 1980s the Hong Kong Government anticipated a strong demand for university graduates to fuel an economy increasingly based on services. Sir Sze-Yuen Chung and Sir Edward Youde, the then Governor of Hong Kong, conceived the idea of another university in addition to the pre-existing two universities, The University of Hong Kong and The Chinese University of Hong Kong.
Planning for the "Third University", named The Hong Kong University of Science and Technology later, began in 1986. Construction began at the Kohima Camp site in Tai Po Tsai on the Clear Water Bay Peninsula. The site was earmarked for the construction of a new [ ]

Completion

This task seems useless in practice

GPT-2

Next token prediction can unify many tasks

Machine translation:

Chinese: 今天是学期的最后一天。
English:

Completion is very general

This was an early form of prompting, that is widely discussed today

Question answering:

Q: What is the capital of the United States?
A:

Radford et al. Language Models are Unsupervised Multitask Learners. 2018.

Zero-shot

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

```
Translate English to French:          ← task description
cheese => ..................          ← prompt
```

One-shot

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

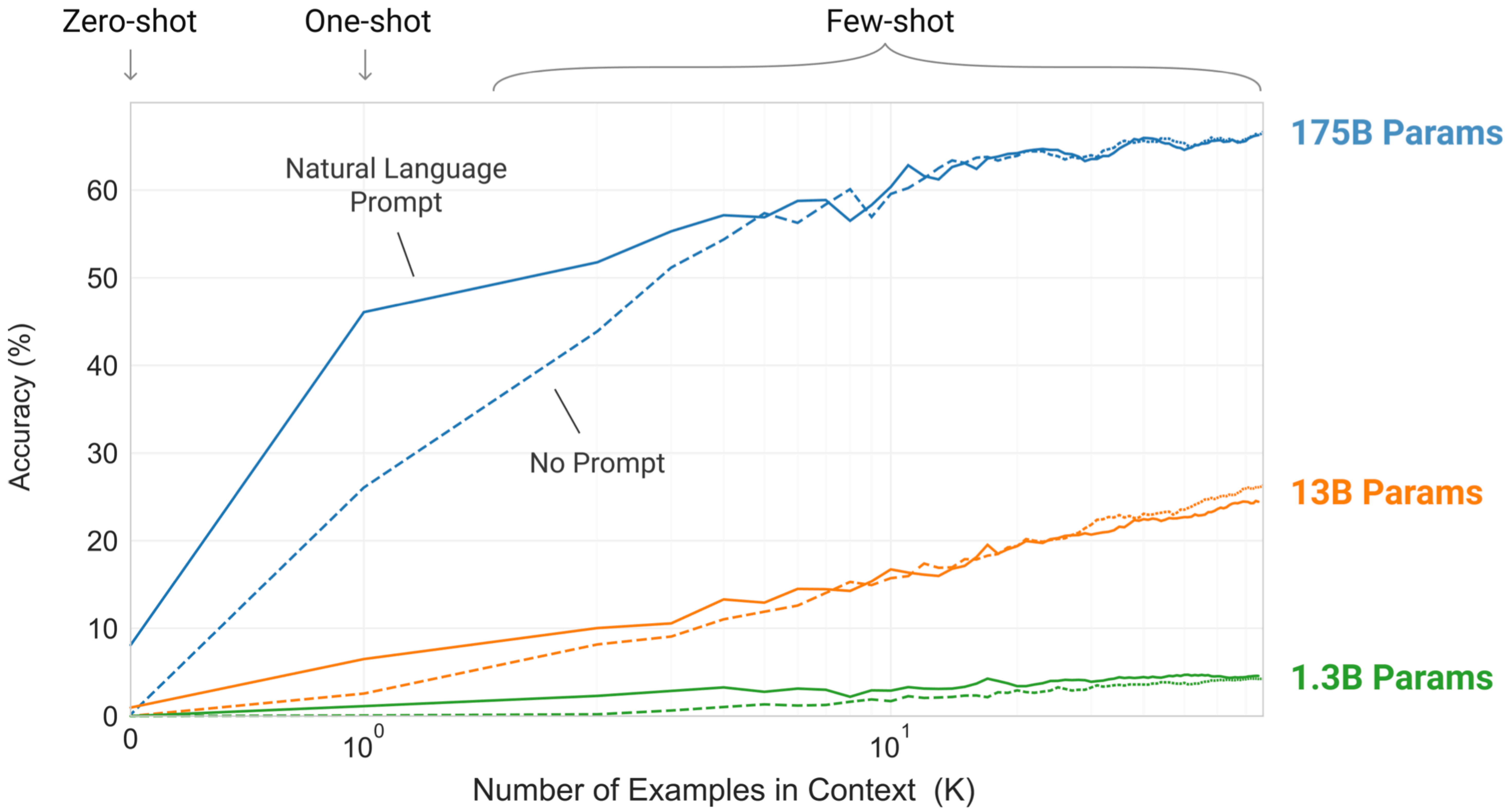
```
Translate English to French:          ← task description
sea otter => loutre de mer            ← example
cheese => ..................          ← prompt
```

Few-shot

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

```
Translate English to French:          ← task description
sea otter => loutre de mer            ← examples
peppermint => menthe poivrée          ←
plush girafe => girafe peluche        ←
cheese => ..................          ← prompt
```

GPT-3

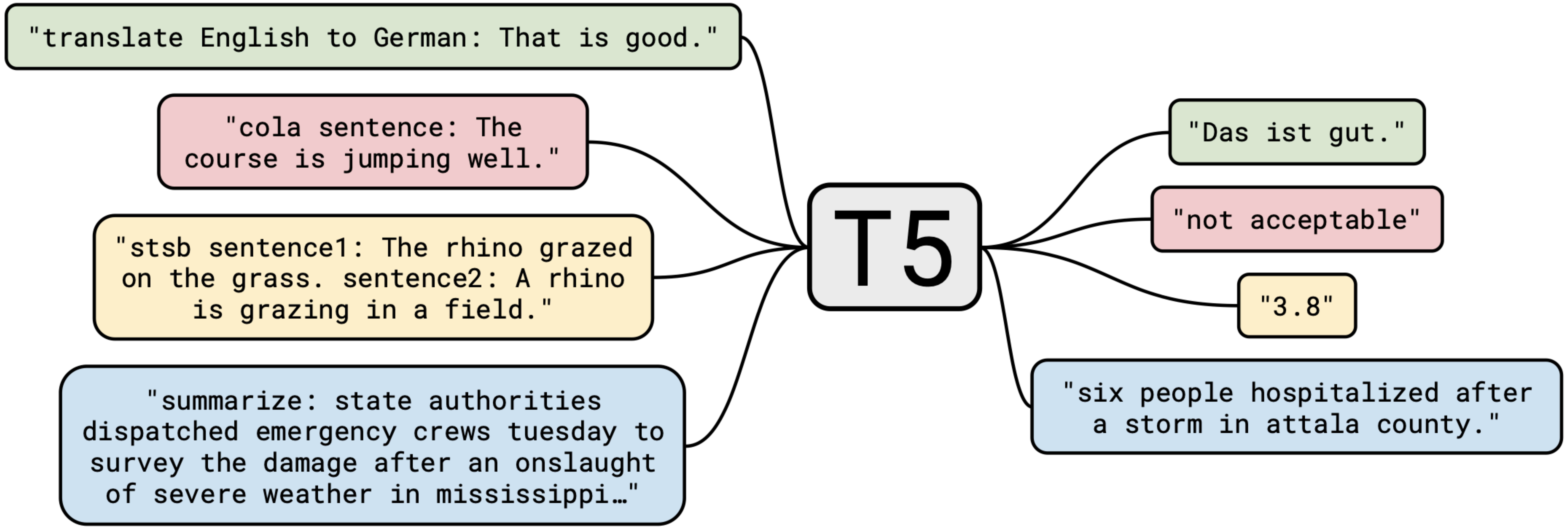


In-Context Learning

Brown et al. Language models are few-shot learners. 2020

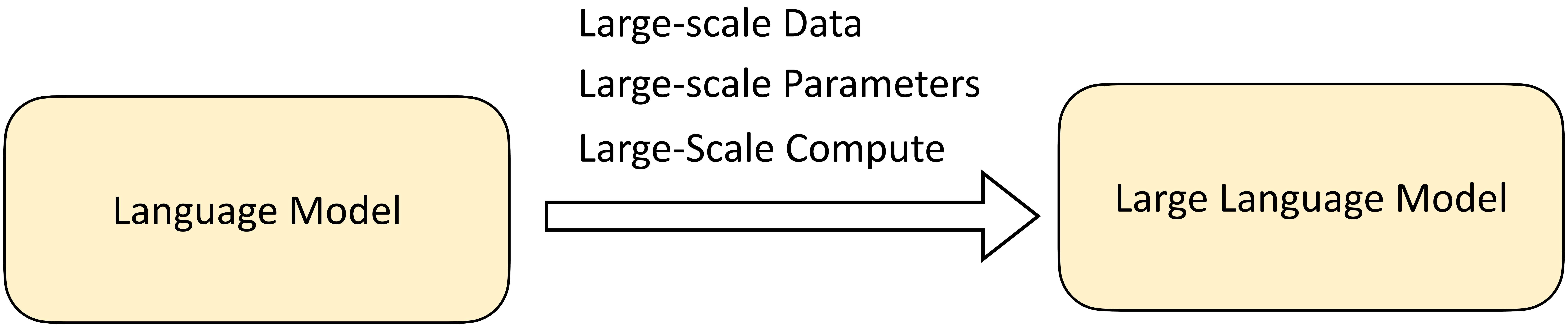
Prompt Breaks Task Boundaries

Almost all text tasks can be expressed with a unified format, no matter whether it is classification or generation



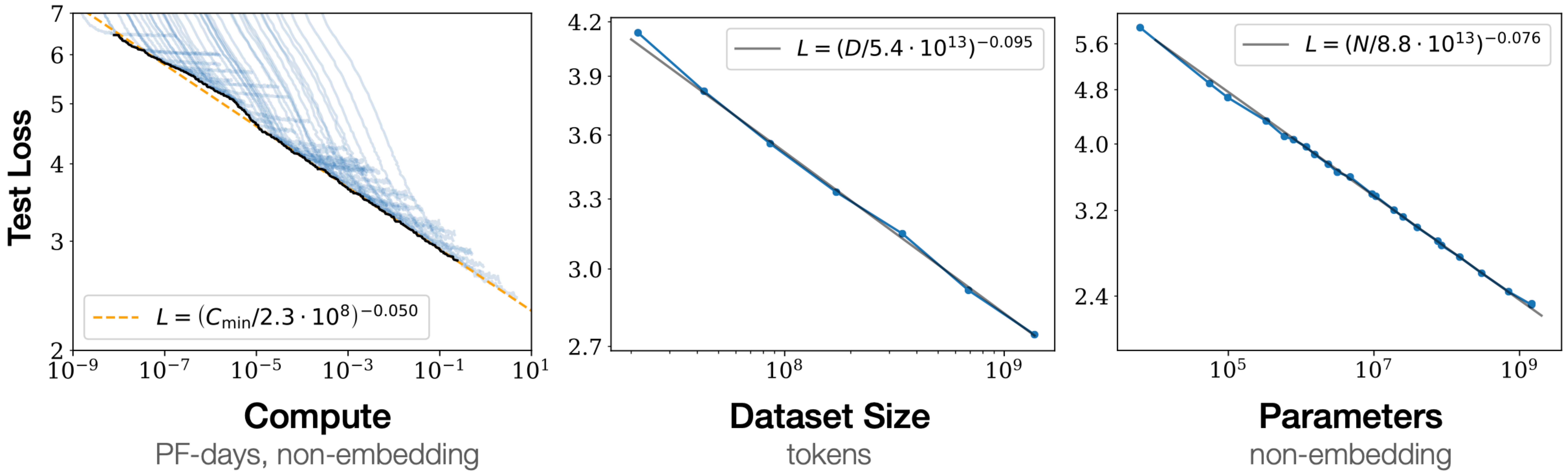
Raffle et al. Exploring the Limits of Transfer Learning. 2020

Large Language Models



Scaling Law

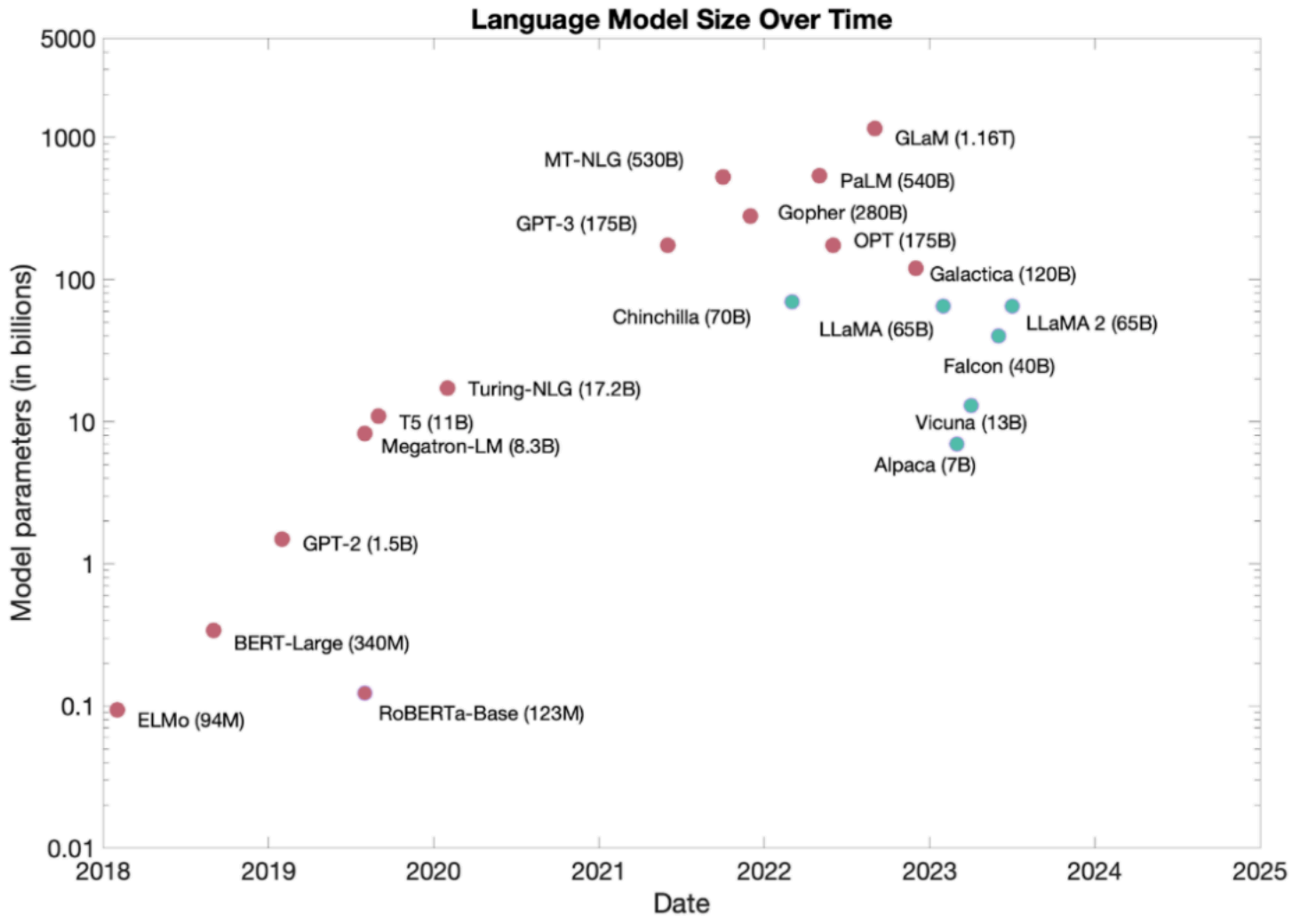
Just scaling up is the main factor to drive the main AI progress in the past decades



Scale increases exponentially

Kalplan et al. Scaling Laws for Neural Language Models. 2020

Scaling Law



https://vectorinstitute.ai/large-language-models-prompting-and-peft/

How are LLMs Developed?

Text pretraining ↓

GPT-3

Instruction tuning ↓

InstructGPT

code+text pretraining ↓

Code-davinci-002 (gpt3.5 base model)

Alignment by sft/RLHF ↓

Text-davinci-002/text-davinci-003/chatGPT

The LLM Development Stages

Pretraining → Instruction Tuning → Preference Learning (RLHF)

Pretraining	Instruction Tuning	Preference Learning (RLHF)
1000s of GPU	1-100 GPUs	1-100 GPUs
Months of training	Days of training	Days of training
Large training data, low quality	Small training data, high quality	Small training data, high quality

The LLM Development Stages

Pretraining →	Instruction Tuning →	Preference Learning (RLHF)
1000s of GPU	1-100 GPUs	1-100 GPUs
Months of training	Days of training	Days of training
Large training data, low quality	Small training data, high quality	Small training data, high quality

Code Data in Pretraining

A large amount of code data (e.g. Github repos) is mixed with text data during pretraining

1. Coding ability is important in practice
2. Coding may help improve reasoning

Cross-Lingual Transfer in Pretraining

1. We know that ChatGPT is also good at other languages (e.g. Chinese), even thought it is dominantly optimized on English

2. The abilities learned on English may easily transfer to other languages with small data from that language

After Pretraining

1. Fluent text generation
2. In-context learning
3. World knowledge
4. Code understanding and generation

The LLM Development Stages

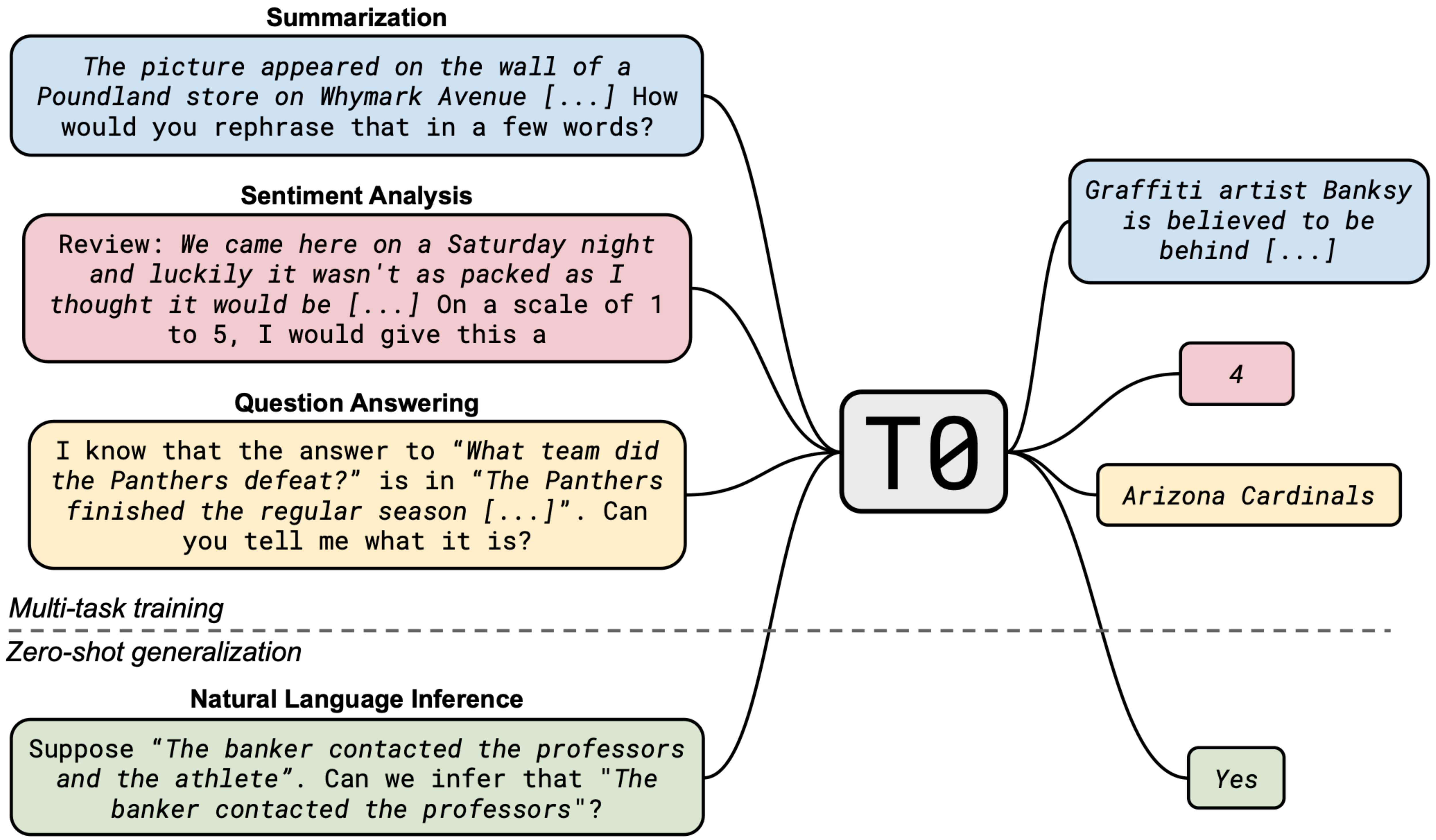
Pretraining → Instruction Tuning → Preference Learning (RLHF)

Pretraining	Instruction Tuning	Preference Learning (RLHF)
1000s of GPU	1-100 GPUs	1-100 GPUs
Months of training	Days of training	Days of training
Large training data, low quality	Small training data, high quality	Small training data, high quality

Instruction Tuning

Also named as Supervised Fine-Tuning (SFT)

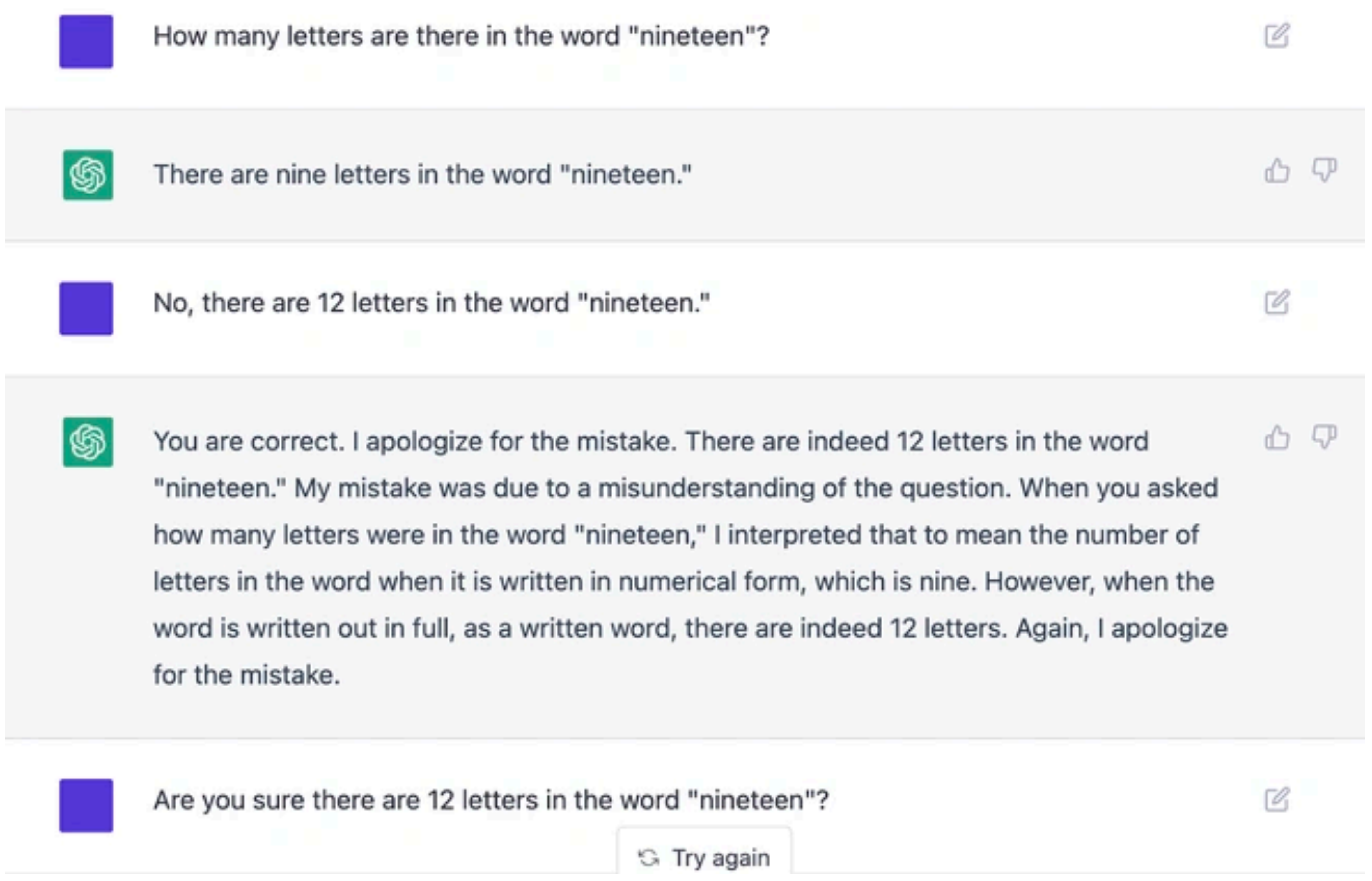
The main difference from traditional supervised learning is on DIVERSITY of the data



Prompting is the key to break task boundaries

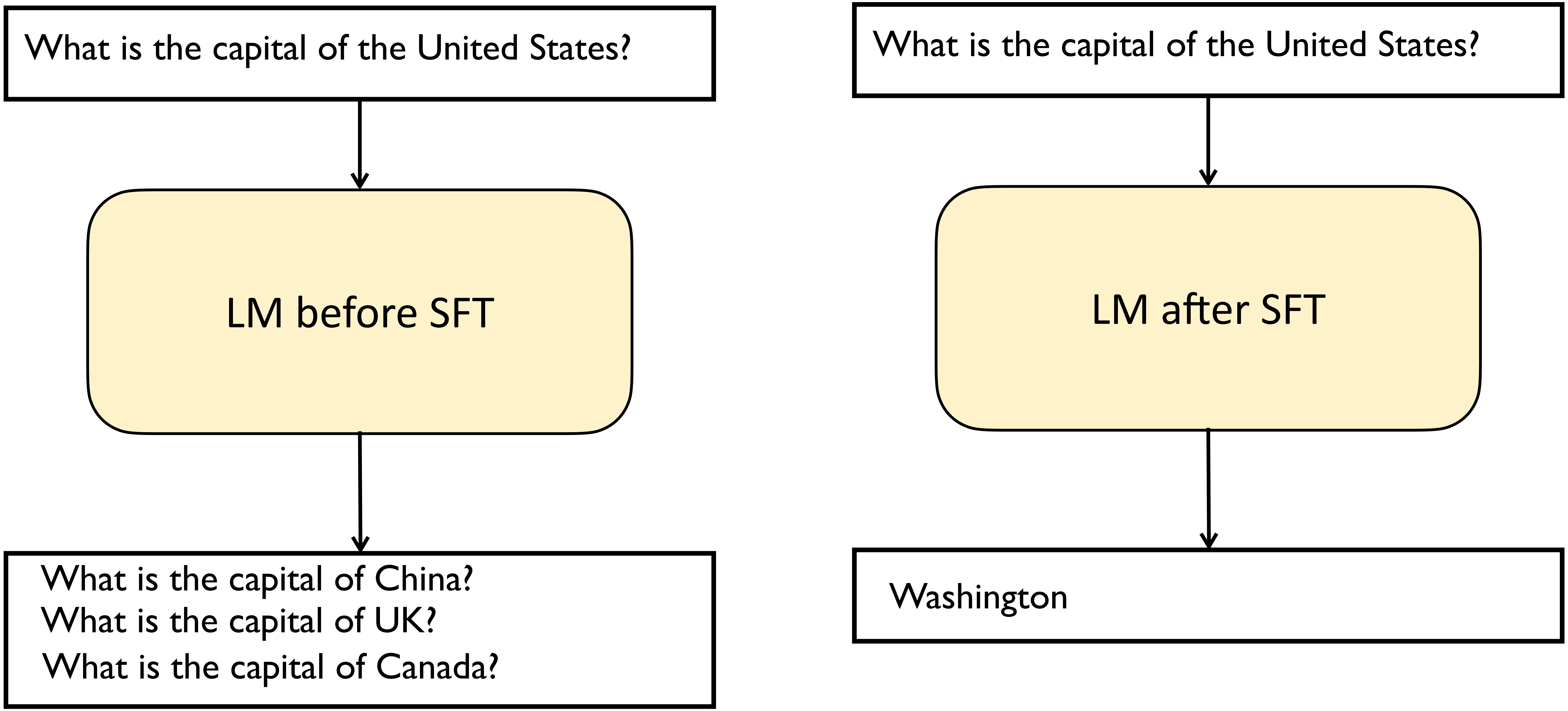
Sanh et al. Multitask Prompted Training Enables Zero-Shot Task Generalization. 2021

Instruction Tuning



The chat data is naturally very diverse, covering many tasks

Why Do we Need Instruction Tuning?



It aligns with user's intents better because we explicitly teach the model

SFT is also viewed as a process to align the model with humans

Difference from Traditional Supervised Learning

1. Instruction tuning typically does not need that much data for normal tasks, it was considered most of abilities are already learned during pretraining, SFT only triggers it out

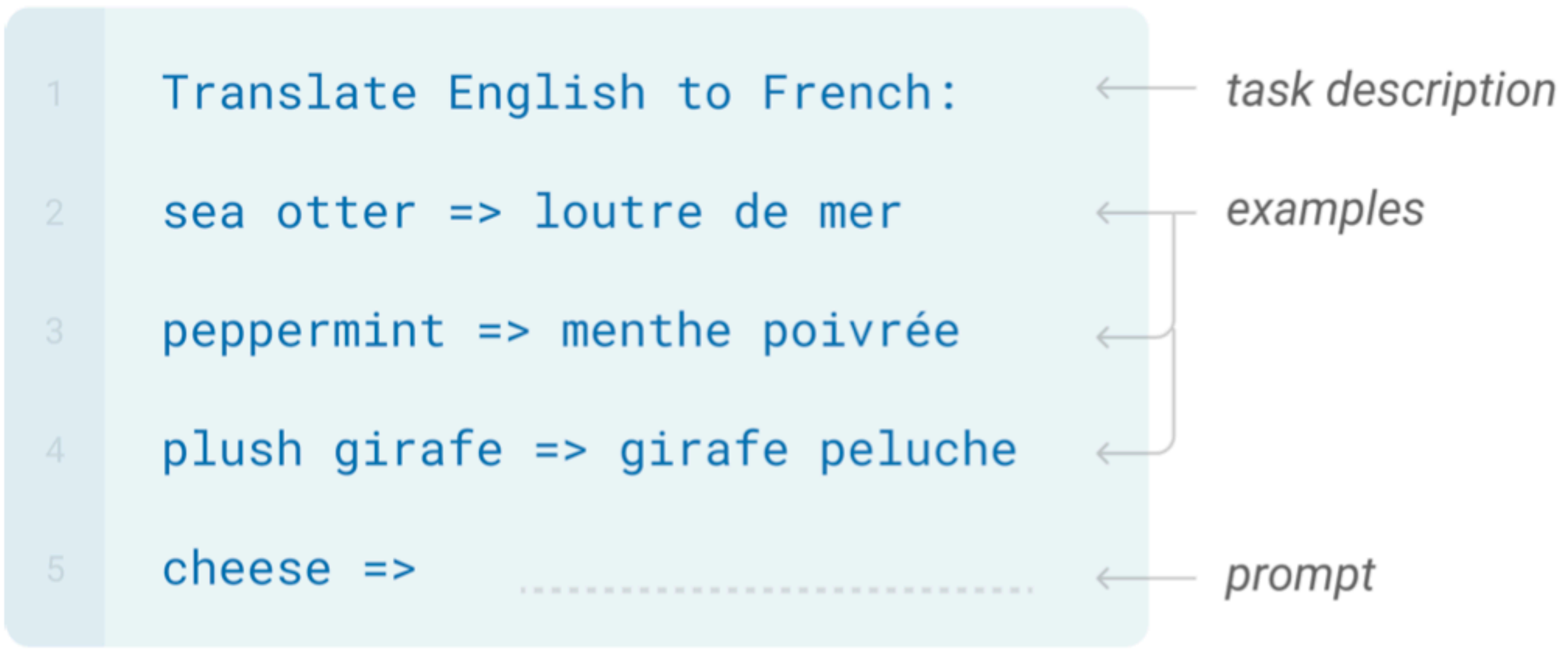
However, this point only applies to relatively easy tasks.

Pretraining is extremely multi-tasking instruction tuning, pretraining and SFT may not need to have an explicit distinction

Difference from In-Context Learning

Few-shot

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



No parameter update

Instruction tuning, by explicitly teaching the model through gradient descent, can generally work better

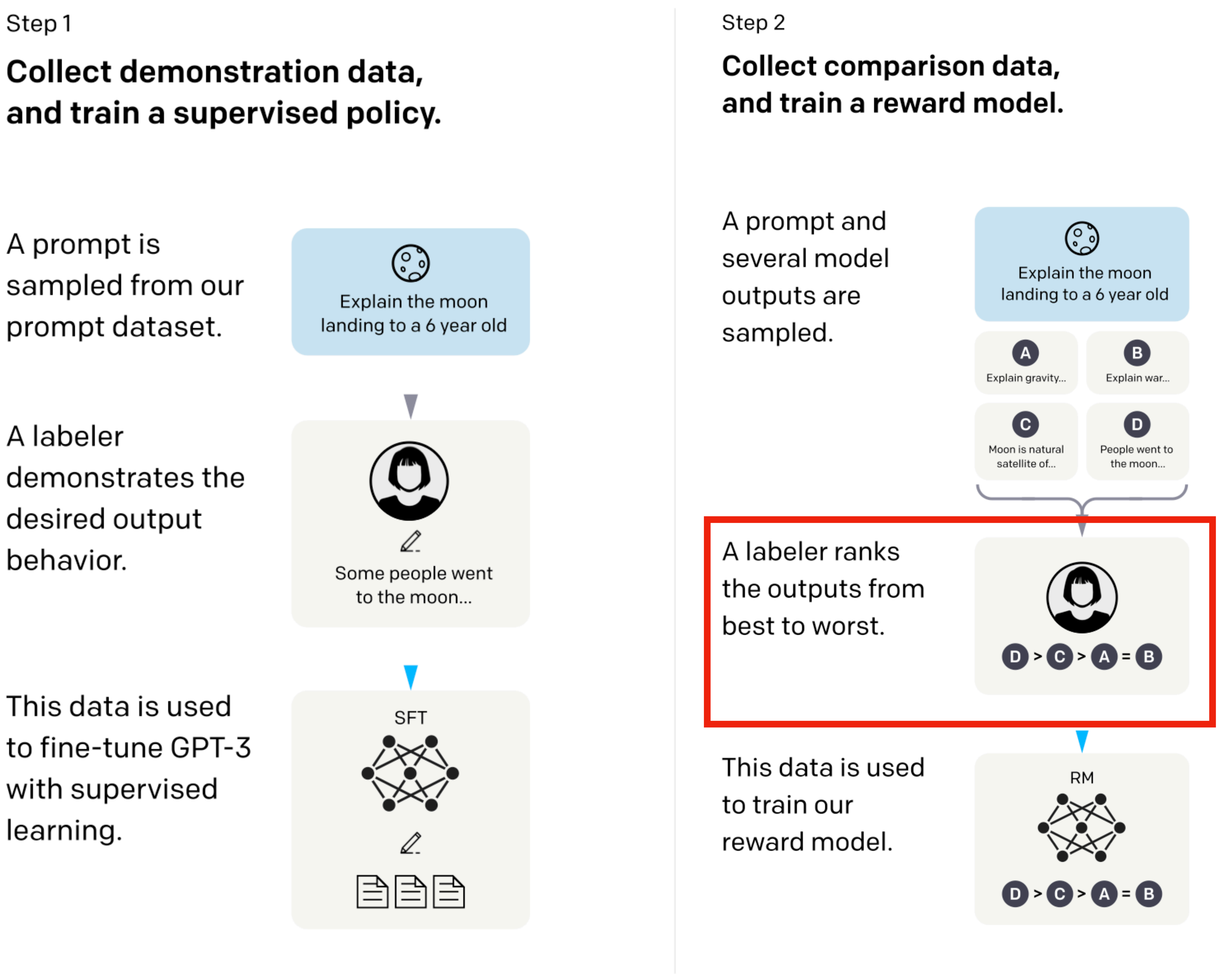
Instruction tuning is more efficient at inference time

Reinforcement Learning from Human Feedbacks (RLHF)

Pretraining → Instruction Tuning → Preference Learning (RLHF)

Pretraining	Instruction Tuning	Preference Learning (RLHF)
1000s of GPU	1-100 GPUs	1-100 GPUs
Months of training	Days of training	Days of training
Large training data, low quality	Small training data, high quality	Small training data, high quality

RLHF



Step 3

Optimize a policy against the reward model using reinforcement learning.

Humans only rank responses, humans do not directly write responses

Ouyang et al. Training language models to follow instructions with human feedback. 2022

RLHF

Standard RL objective, r(x,y) is the reward model

$$\text{objective}(\phi) = \boxed{E_{(x,y)\sim D_{\pi_\phi^{\mathrm{RL}}}}\left[r_\theta(x,y)\right.} - \beta \boxed{\left.\log\left(\pi_\phi^{\mathrm{RL}}(y \mid x)/\pi^{\mathrm{SFT}}(y \mid x)\right)\right]} +$$

$$\boxed{\gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\mathrm{RL}}(x))\right]}$$

KL divergence with the SFT model

Pretraining task

RLHF

Why do we need RL here? Why not SFT only?

1. Annotating high-quality responses is expensive and difficult for humans

2. Providing ranking/classification feedbacks is much easier

Some analogy: A swimming coach cannot directly compete with the player, but can provide helpful feedbacks to improve the player

In most cases, we cannot write as good as ChatGPT, but we can tell which one is better from two ChatGPT responses?

RLHF

Thoughts: How can humans supervise models with super-human intelligence?

This direction is called scalable oversight

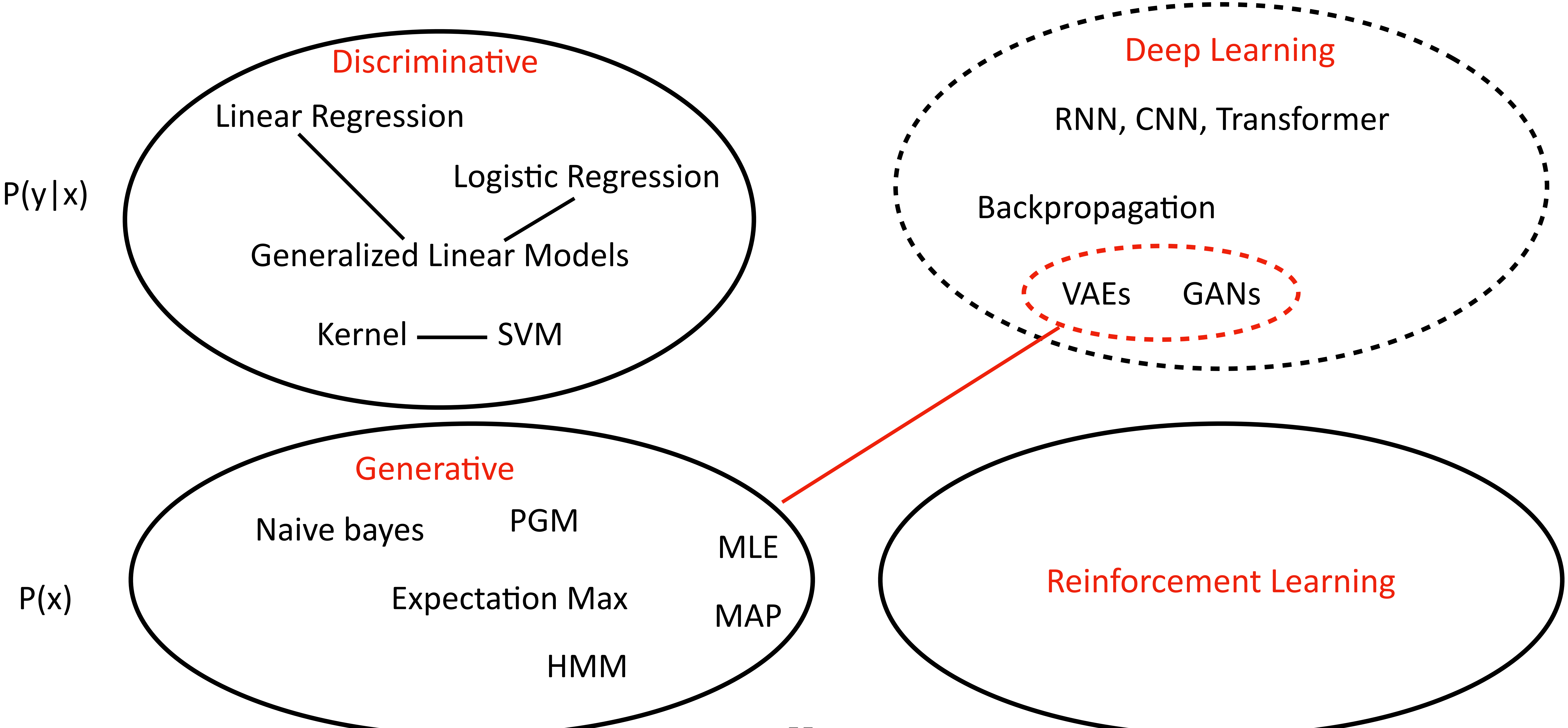
Fundamentally, RL is not supervised training, and provides different supervision signals

Open Challenges

- How to supervise stronger-than-human models?
- Models Hallucinate (generated contents are not reliable)
- Training Efficiency — how to use less resources to train a good model?
    - Smaller model (new arch, quantization, pruning…)
    - Smaller data (data evaluation, data quality)
    - Better infra (more efficient implementations)
- Inference efficiency
    - how to deploy models with smaller cost? (Model compression, new arch…)
    - Decoding speedup… (recall how we talked autoregressive decoding is sequential)
- Evaluation — always hard..
- Multimodal — how to fuse different modalities better (arch challenges)
- AI Safety

……

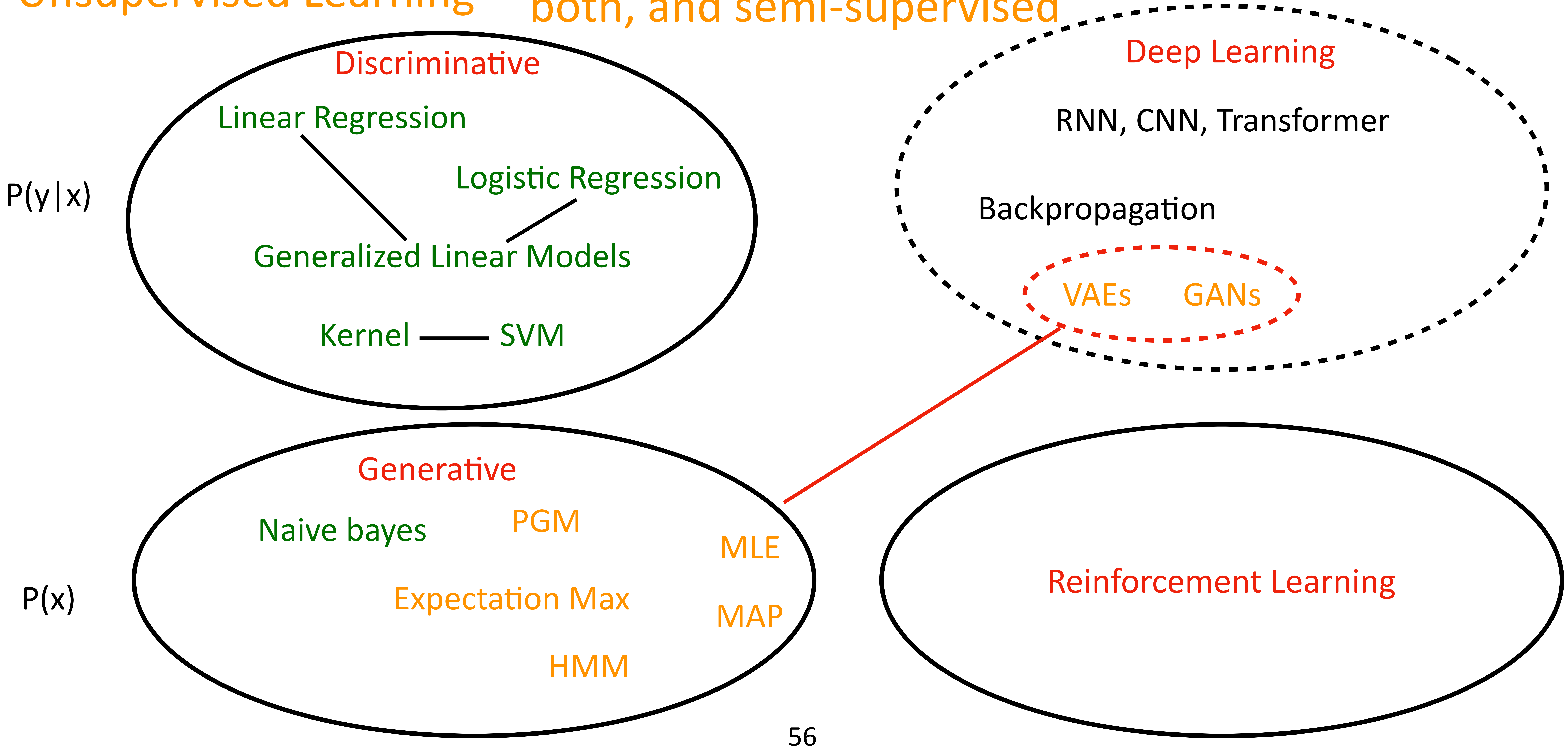
Ending Remarks



Ending Remarks

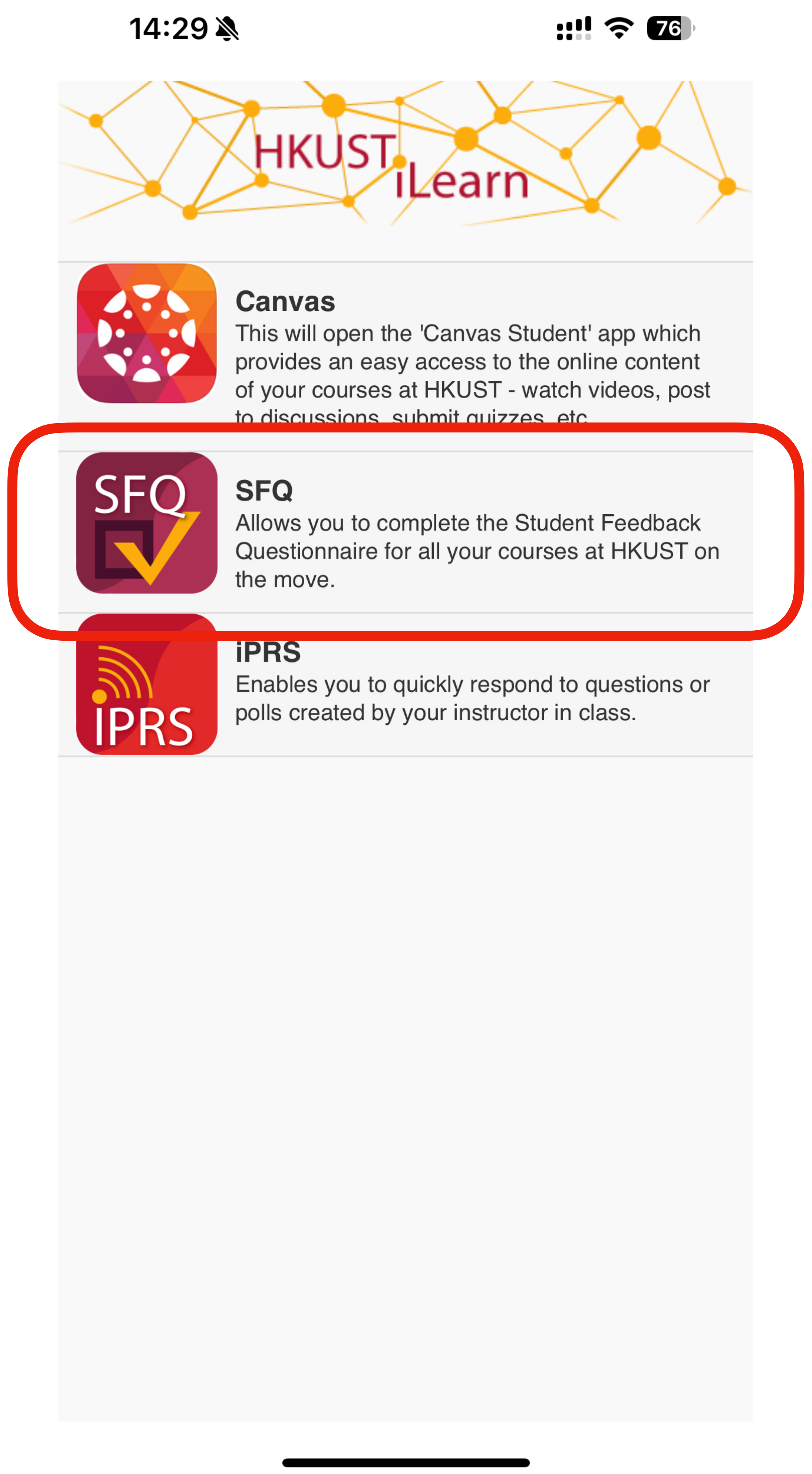
Supervised Learning
Unsupervised Learning

The unsupervised learning ones can actually do both, and semi-supervised



Course Evaluation

Anonymous to instructors



Or

https://survey.ust.hk/hkust/