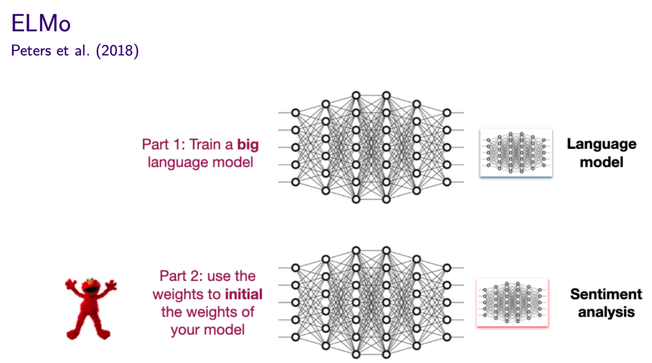
# Advanced Natural Language Processing (ANLP)

## Lecture 4: Large Language Models

Gabriel Stanovsky & **Roy Schwartz**

# Part 1: What is Modern NLP on and How Does it Work?

- Week 1: Intermediate applications
- Week 2: Downstream applications
- Week 3: How do we represent text?
- Week 4: Large language models! Or — which tools solve NLP tasks?



# Reminder: ELMO

## Peters et al. (2018)

| Task | Previous SOTA | | Our baseline | ELMo + baseline | Increase (absolute/ relative) |
|---|---|---|---|---|---|
| SQuAD | Liu et al. (2017) | 84.4 | 81.1 | 85.8 | 4.7 / 24.9% |
| SNLI | Chen et al. (2017) | 88.6 | 88.0 | 88.7 $\pm$ 0.17 | 0.7 / 5.8% |
| SRL | He et al. (2017) | 81.7 | 81.4 | 84.6 | 3.2 / 17.2% |
| Coref | Lee et al. (2017) | 67.2 | 67.2 | 70.4 | 3.2 / 9.8% |
| NER | Peters et al. (2017) | 91.93 $\pm$ 0.19 | 90.15 | 92.22 $\pm$ 0.10 | 2.06 / 21% |
| SST-5 | McCann et al. (2017) | 53.7 | 51.4 | 54.7 $\pm$ 0.5 | 3.3 / 6.8% |

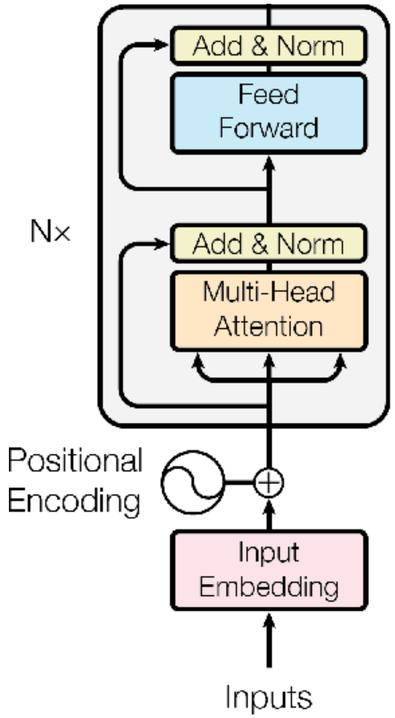
# Outline

## Large Language Models

- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science

# Encoder-only Models

- A discriminator model
- Popular example: use the Transformer encoder
- The first generation of LLMs
  - BERT
  - RoBERTa
  - ELECTRA



# BERT

## Devlin et al. (2019)

- A standard Transformer encoder architecture
- Add a special [CLS] token at the beginning of the sentence
- Pre-training objectives
  - Mask random tokens (aka masked language modeling, MLM)
  - Next sentence prediction (NSP)
- Two model sizes released
  - BASE (~100M params), LARGE (~300M params)
- Pre-trained corpora
  - Wikipedia and BookCorpus

- **Input:** the man went to the store
- **MLM**:[CLS] the [MASK] went to [MASK] store [SEP]
- **NSP**: he bought a gallon [MASK] milk
- **Label**: isNext

- **MLM**:[CLS] the [MASK] went to [MASK] store [SEP]
- **NSP**: penguin [MASK] flight ##less birds
- **Label**: notNext

# RoBERTa

## Liu et al. (2019)

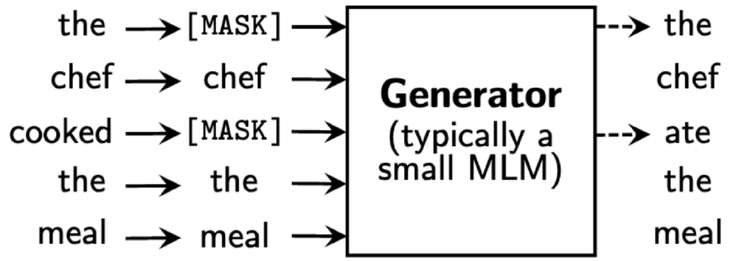
- Identical architecture to BERT
- Differences
  - Drop the NSP part
  - Larger batch size
  - Dynamic masking
  - Train for longer
- Much better results

| 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** |
| $\text{BERT}_{\text{LARGE}}$ | | | | | | |
| with BOOKS + WIKI | 13GB | 256 | 1M | 90.9/81.8 | 86.6 | 93.7 |
| $\text{XLNet}_{\text{LARGE}}$ | | | | | | |
| with BOOKS + WIKI | 13GB | 256 | 1M | 94.0/87.8 | 88.4 | 94.4 |
| + additional data | 126GB | 2K | 500K | 94.5/88.8 | 89.8 | 95.6 |

# ELECTRA

## Clark et al. (2020)

- Train a small generator to replace some of the input tokens with other, auto-generated tokens
- Train a discriminator to predict for each token whether it is *original* or *replaced*



# Outline

## Large Language Models

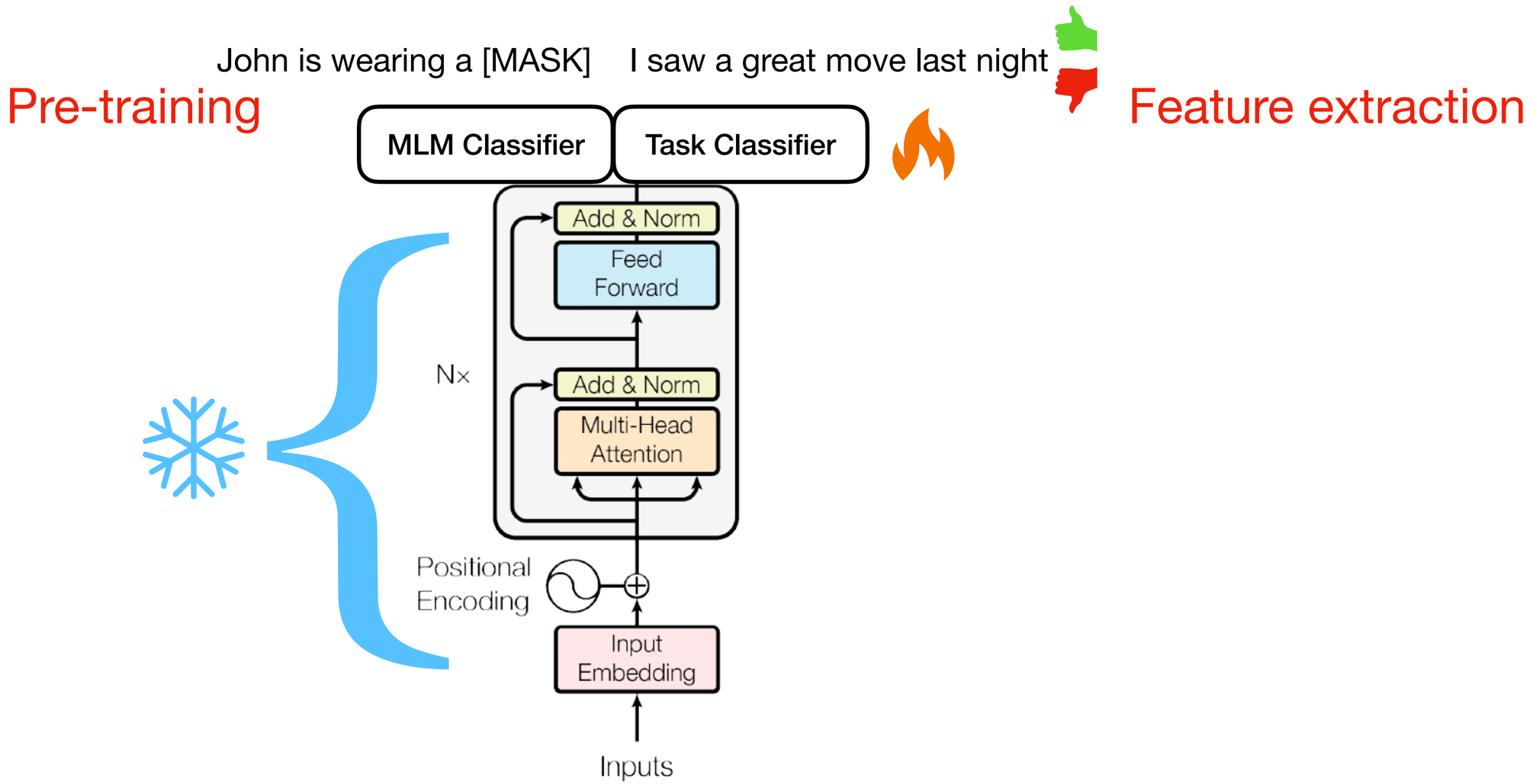
- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science

# Using Encoder-only Models

## Feature Extraction

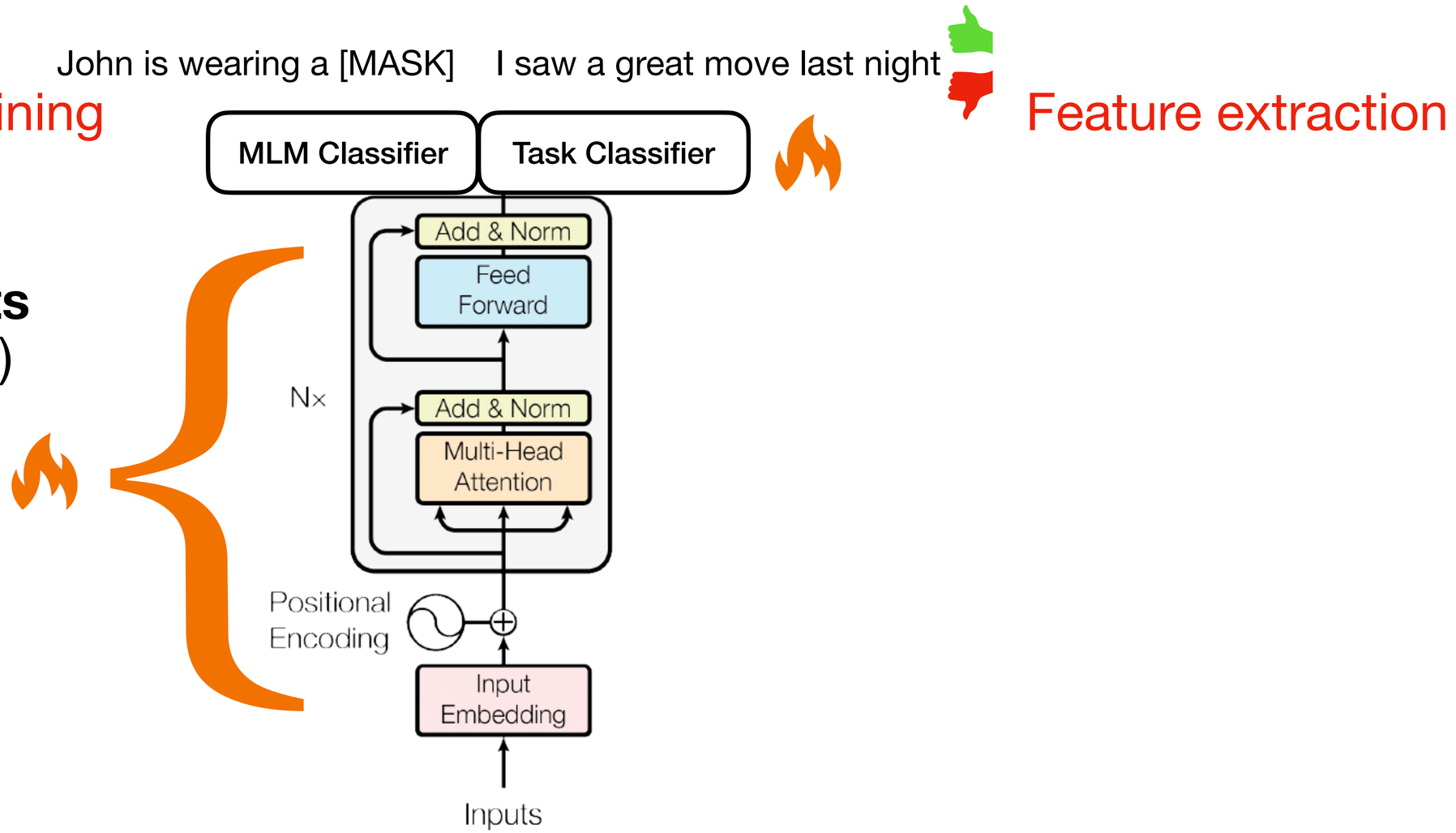
- Originally proposed by the ELMo paper (Peters et al., 2018)
- For a given task *t:*
  - Replace the LM classifier with the task classifier
  - Update **only the classifier weights**
  - Keep the rest frozen

# Feature Extraction



# Fine-Tuning

- Update **all weights** (classifier + model)



# To Tune or not to Tune? Well, it Depends

## Peters et al. (2019)

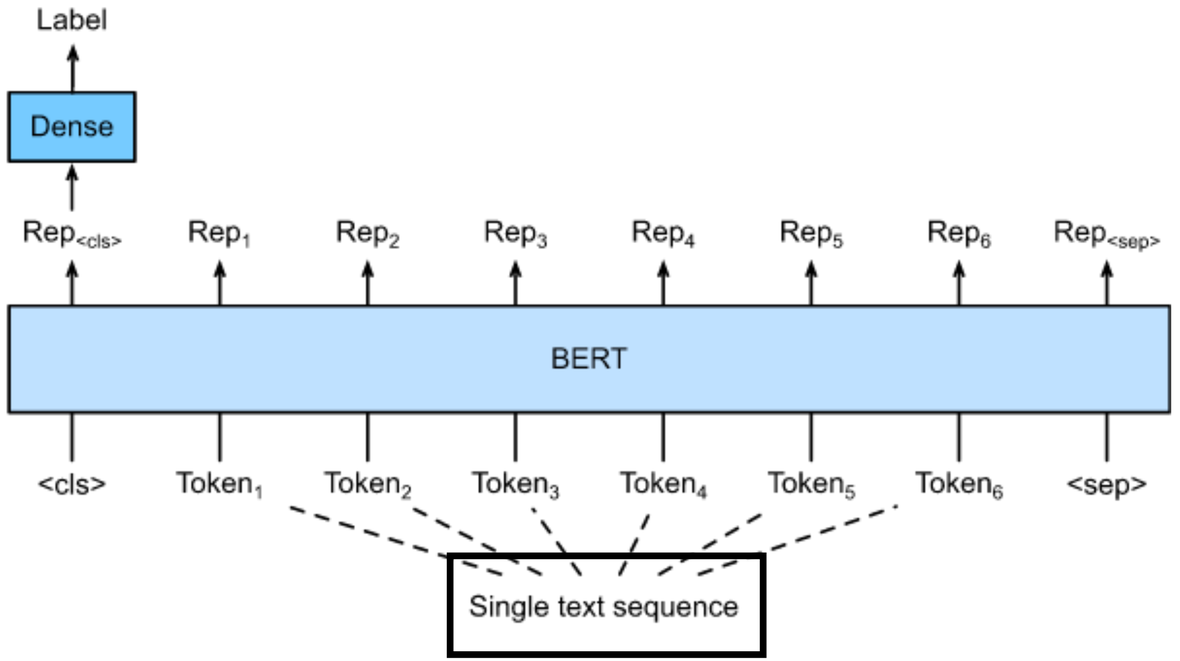
| Pretraining | Adaptation | NER CoNLL 2003 | SA SST-2 | Nat. lang. inference MNLI | Nat. lang. inference SICK-E | Semantic textual similarity SICK-R | Semantic textual similarity MRPC | Semantic textual similarity STS-B |
|---|---|---|---|---|---|---|---|---|
| Skip-thoughts | ❄️ | - | 81.8 | 62.9 | - | 86.6 | 75.8 | 71.8 |
| ELMo | ❄️ | 91.7 | **91.8** | **79.6** | **86.3** | **86.1** | **76.0** | **75.9** |
| | 🔥 | **91.9** | 91.2 | 76.4 | 83.3 | 83.3 | 74.7 | 75.5 |
| | Δ=🔥-❄️ | 0.2 | -0.6 | -3.2 | -3.3 | -2.8 | -1.3 | -0.4 |
| BERT-base | ❄️ | 92.2 | 93.0 | **84.6** | 84.8 | 86.4 | 78.1 | 82.9 |
| | 🔥 | **92.4** | **93.5** | **84.6** | **85.8** | **88.7** | **84.8** | **87.1** |
| | Δ=🔥-❄️ | 0.2 | 0.5 | 0.0 | 1.0 | 2.3 | 6.7 | 4.2 |

- In practice, all models nowadays **fine-tune** 🔥

# Fine-tuning Sentence Classification

## E.g., Sentiment Analysis, Topic Classification

- Remember the [CLS] token?
- Use it to represent the sentence
  - A classifier is trained on top of it

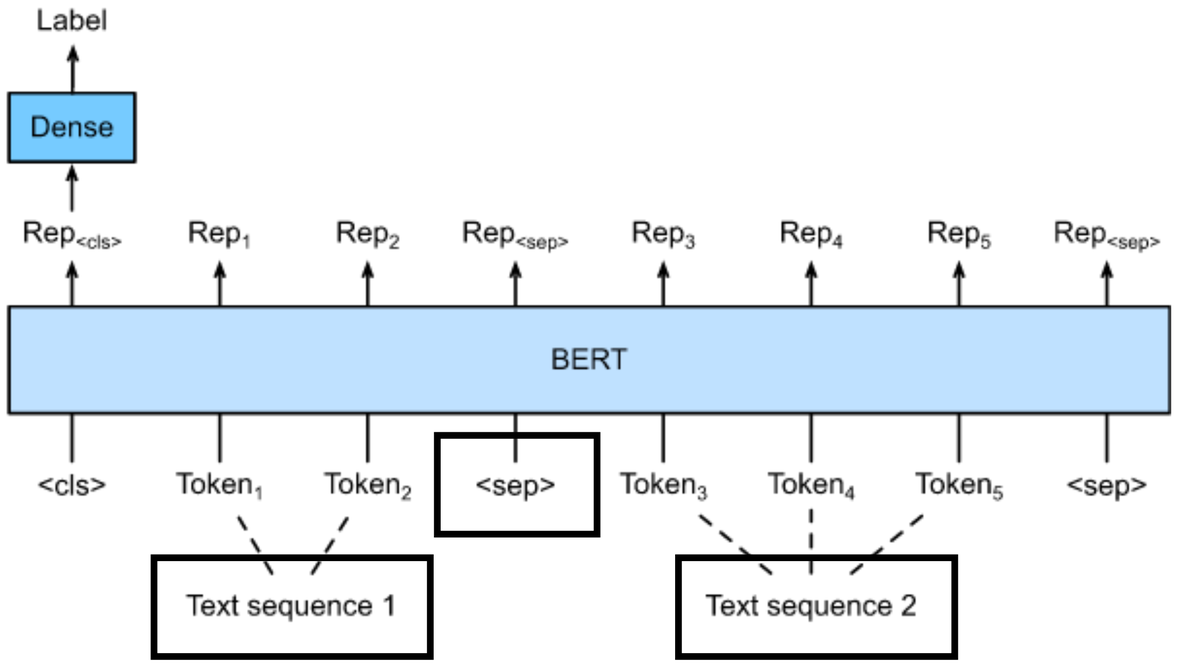



Taken from https://d2l.ai/chapter_natural-language-processing-applications/finetuning-bert.html

# Sentence-pair Classification

## E.g., NLI

- Concatenate both sentences
  - Add a [SEP] token in between
- Train a classifier on top of the [CLS] vector

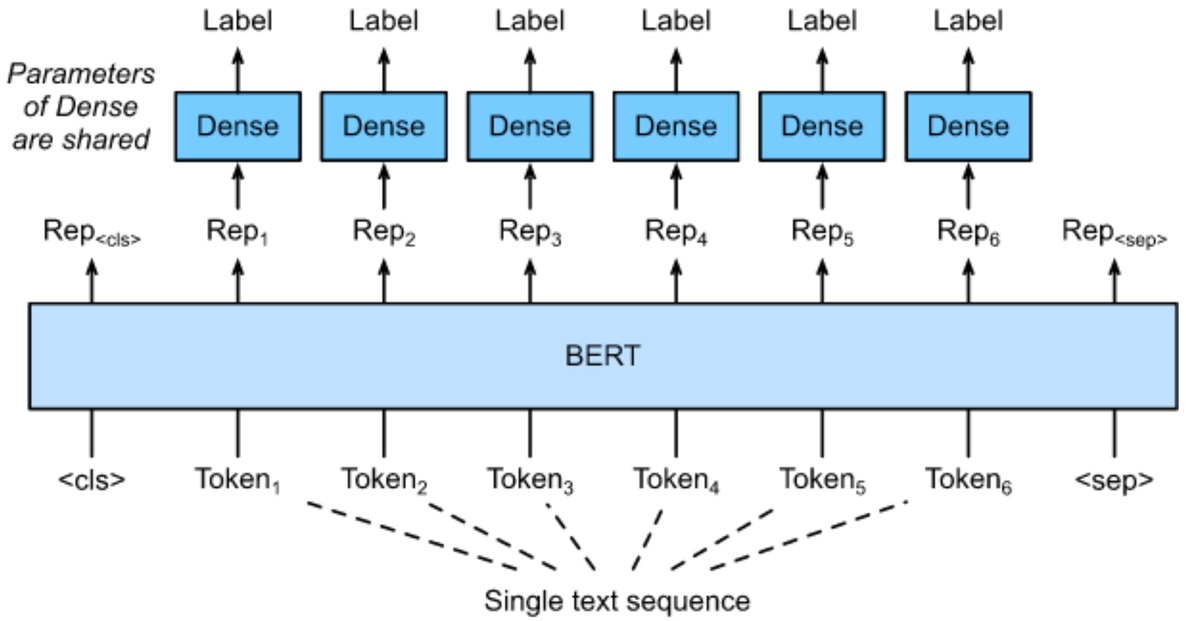



Taken from https://d2l.ai/chapter_natural-language-processing-applications/finetuning-bert.html

# Token-level Tasks

## E.g., PoS Tagging, NER

- Train a **single** classifier
  - Input: each token representation (excluding [CLS])
  - Output: token label
- Make a set of independent predictions

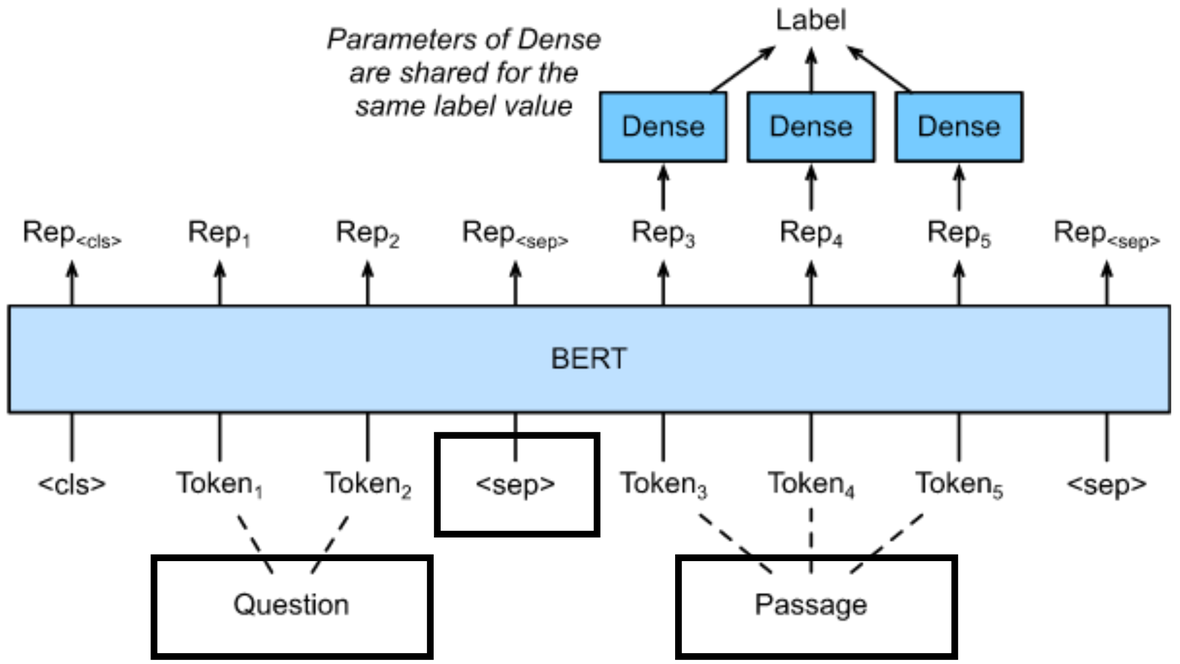



Taken from https://d2l.ai/chapter_natural-language-processing-applications/finetuning-bert.html

# Reading Comprehension

## Question Answering with an Input Passage

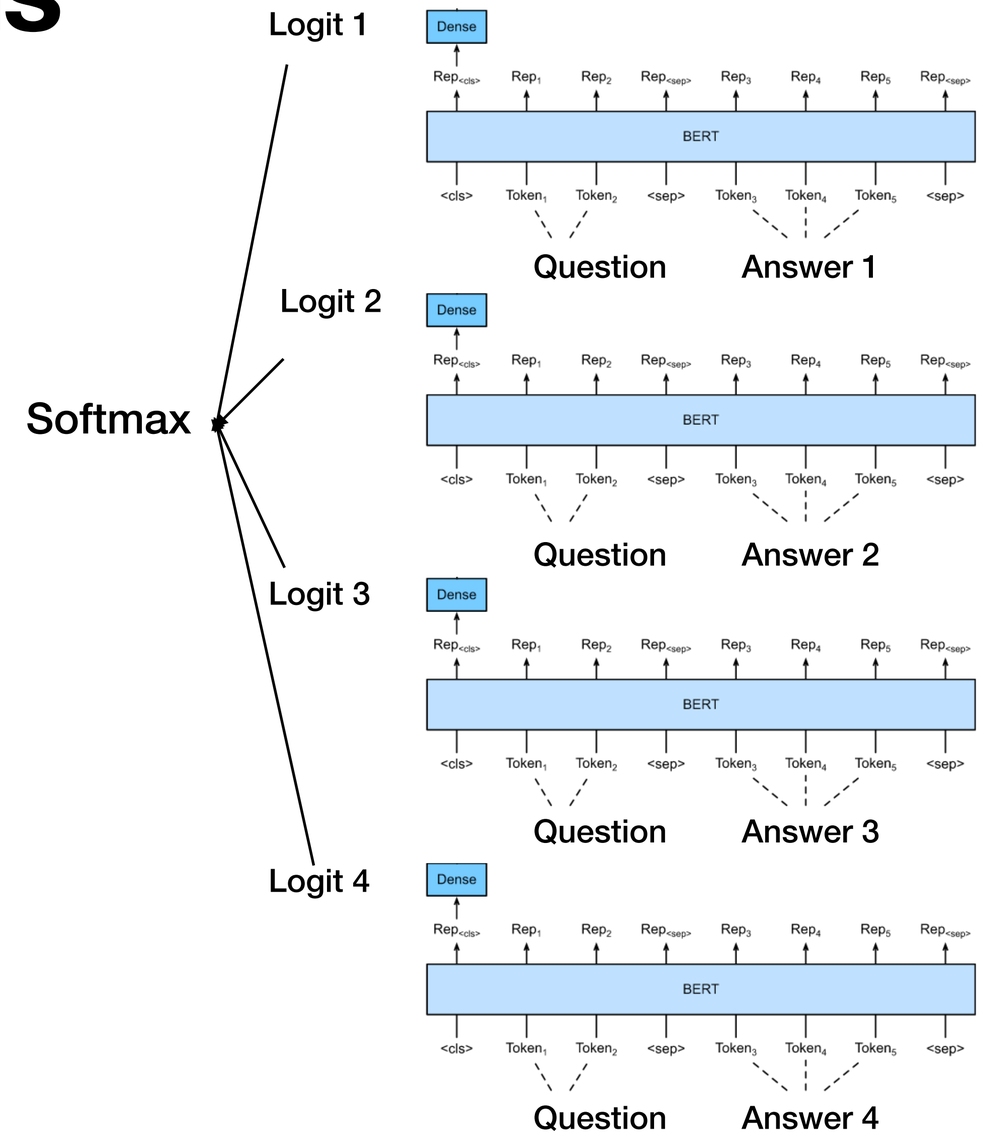
- Concatenate passage and question
  - Add a [SEP] token in between
- Train two token classifiers
  - One for predicting the answer span **start**
  - The other for predicting the span **end**
- Compute end and start probabilities for each token
- Return the most probable span




Taken from https://d2l.ai/chapter_natural-language-processing-applications/finetuning-bert.html

# Multi-choice Questions

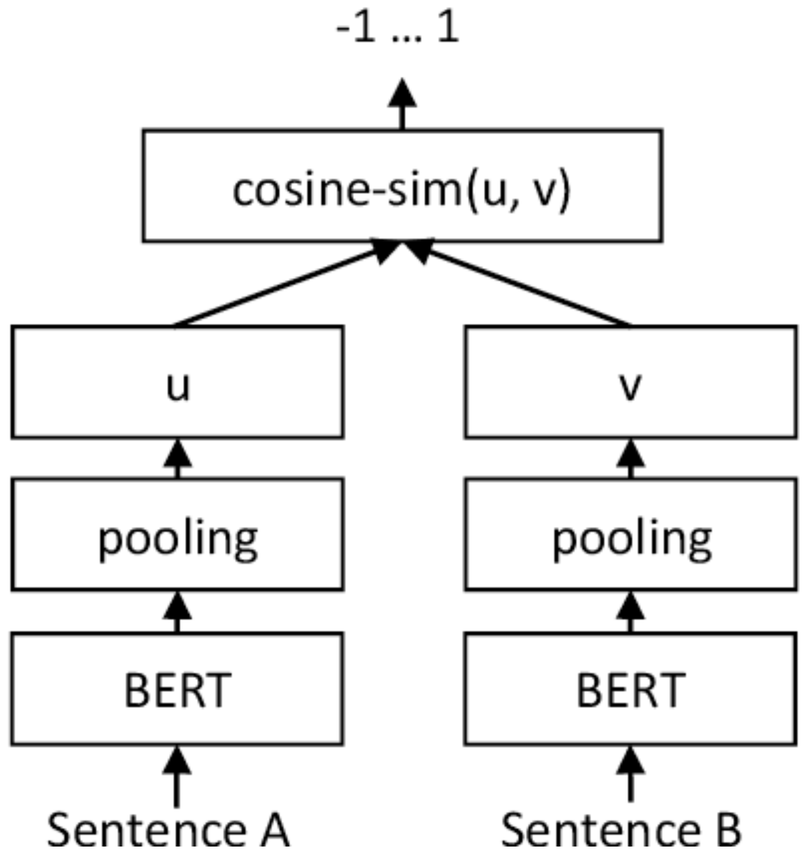
- For each answer candidate $c_i$
  - Concatenate question and $c_i$
  - Add a [SEP] token in between
  - Compute a single logit
- Run a softmax over all logits



# Sentence Similarity

## SentenceBERT (SBERT, Ruckle & Gurevych, 2019)

- Compute the [CLS] vector representation for each of the sentences
- Compute cosine similarity between both vectors



# Probing

- What information is encoded inside an MLM?
- Typically uses feature extraction
  - I.e., keep models frozen
- Classifier is typically small
  - The intuition: it is limited in its learning capacity, so any success is attributed to the pretrained model

# Text Generation

- Encoder models such as BERT do not support text generation
  - Only filling in words in a sentence
- There are tricks that can overcome this (Wang & Cho, 2019)
  - Ranking a set of candidates
  - Sampling from a set of [MASK]s
- But actually, *generative* models are a much better option

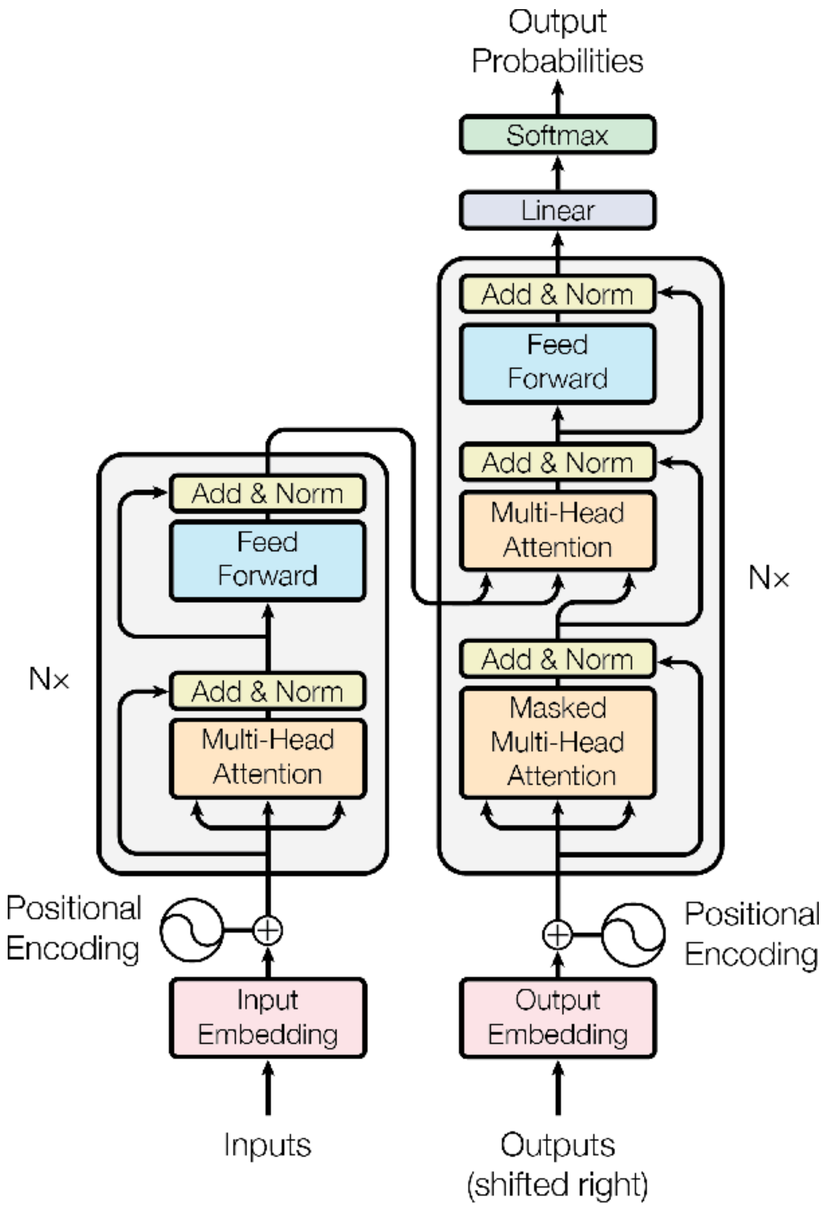
# Outline

## Large Language Models

- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science

# Encoder-decoder Models

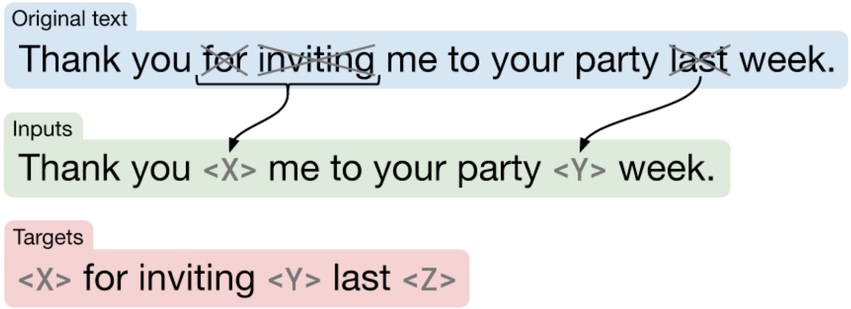
- Model architecture similar to the vanilla (encoder-decoder) architecture
- Support seq2seq tasks
  - E.g., translation, summarization

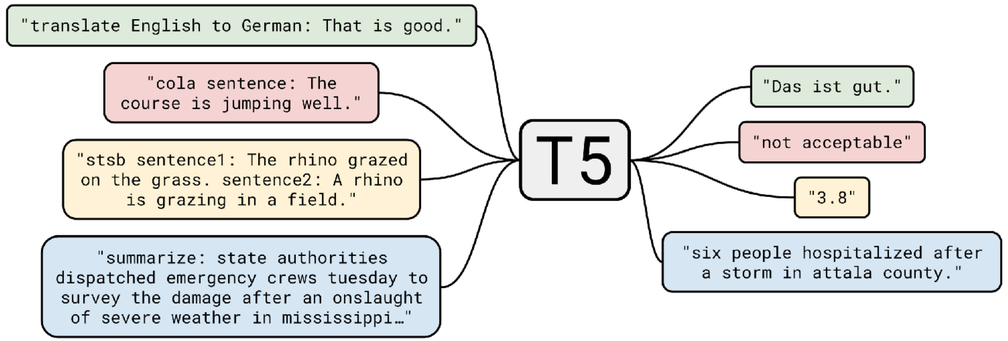


# T5

## Raffel et al. (2020)

- A standard Transformer architecture
- Unsupervised pretraining objective
  - Span corruption
- Supervised pretraining objectives
  - Text-to-text
- Can be fine-tuned on unseen tasks
- Lot's of followup work
  - Multilingual T5 (mT5; Xue et al., 2021)
  - Support for zero-shot generalization (T0; Sanh et al., 2022)
  - Instruction tuning (FLAN; Wei et al., 2022)



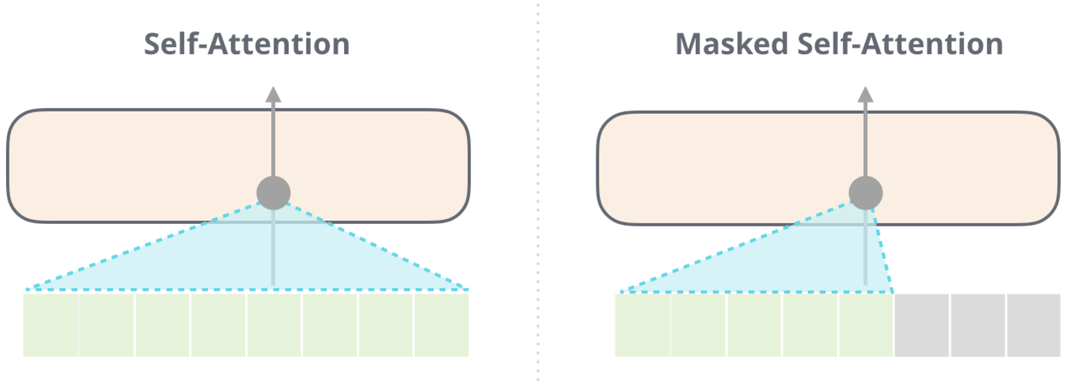



# Decoder-only Models

- Pre-train an auto-regressive language model
  - I.e., predict next word
- Use last hidden token for fine-tuning on classification tasks
- Most popular approach in recent large scale models
  - E.g., GPT-3 (Brown et al., 2020), Gopher (Rae et al., 2021), Bloom (Le Scao et al., 2022), OPT (Zhang et al., 2022), Chinchilla (Hoffmann et al., 2022), PaLM (Chowdhery et al., 2022)

# Masking in Decoder-only Models

- Important: future tokens are **masked**




Taken from https://jalammar.github.io/illustrated-gpt2/#part-1-got-and-language-modeling

# Comparison between Different Architectures

## Wang et al. (2022)

- **Finding 1.** **Causal decoder-only** models pretrained with a **full language modeling** objective achieve best zero-shot generalization when evaluated immediately after **unsupervised pre-training**, in line with current common practices for large language models.

- **Finding 2.** **Encoder-decoder** models trained with **masked language modeling** achieve the best zero-shot performance after **multitask finetuning**. More broadly, approaches that perform well in the single-task finetuning setting perform well on multitask finetuning.

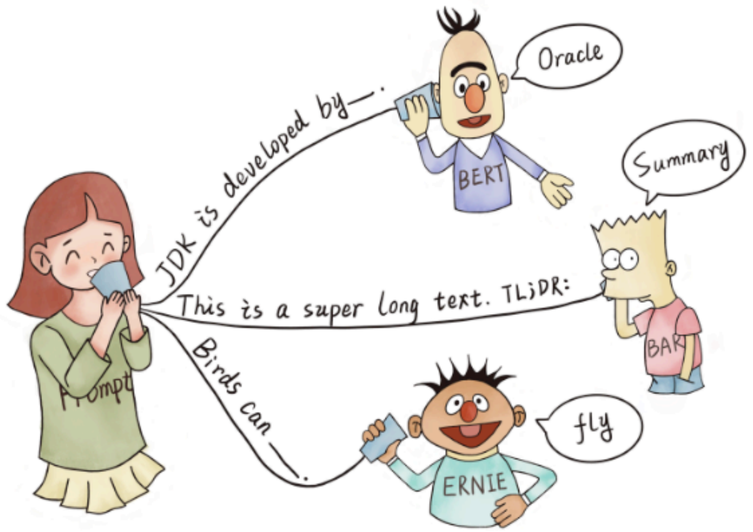
# Outline

## Large Language Models

- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science

# Prompting

- There is an inherent mismatch between the **pre-training** objectives and the downstream tasks
- Prompting tries to overcome this
- The main idea:
  - Instruct the model what to do
  - Cast task as fill-in-the-blank




Taken from Liu et al. (2021)

# Prompting

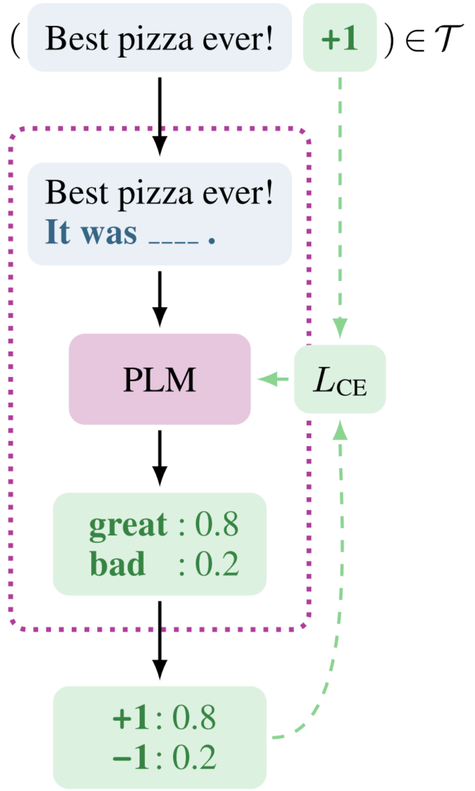
## Examples (Liu et al., 2021)

| Type | Task | Input ([X]) | Template | Answer ([Z]) |
|---|---|---|---|---|
| Text CLS | Sentiment | I love this movie. | [X] The movie is [Z]. | great<br>fantastic<br>... |
| | Topics | He prompted the LM. | [X] The text is about [Z]. | sports<br>science<br>... |
| | Intention | What is taxi fare to Denver? | [X] The question is about [Z]. | quantity<br>city<br>... |
| Text-span CLS | Aspect Sentiment | Poor service but good food. | [X] What about service? [Z]. | Bad<br>Terrible<br>... |
| Text-pair CLS | NLI | [X1]: An old man with ...<br>[X2]: A man walks ... | [X1]? [Z], [X2] | Yes<br>No<br>... |
| Tagging | NER | [X1]: Mike went to Paris.<br>[X2]: Paris | [X1][X2] is a [Z] entity. | organization<br>location<br>... |
| Text Generation | Summarization | Las Vegas police ... | [X] TL;DR: [Z] | The victim ...<br>A woman ...<br>... |
| | Translation | Je vous aime. | French: [X] English: [Z] | I love you.<br>I fancy you.<br>... |

# Training Prompt Methods

## Update LM Weights

- The simplest approach is to fine-tune the model parameters
  - PET (Schick & Schütze, 2021); LM-BFF (Gao et al., 2021)



# Challenges in Prompting

- Which prompts to use?
- Verbalizing
  - Converting labels to text string
  - Trivial in some cases (e.g., translation)
  - Requires mapping in others (e.g., good/great/awesome -> 1, bad/terrible/awful -> 0)

# The Prompt Matters

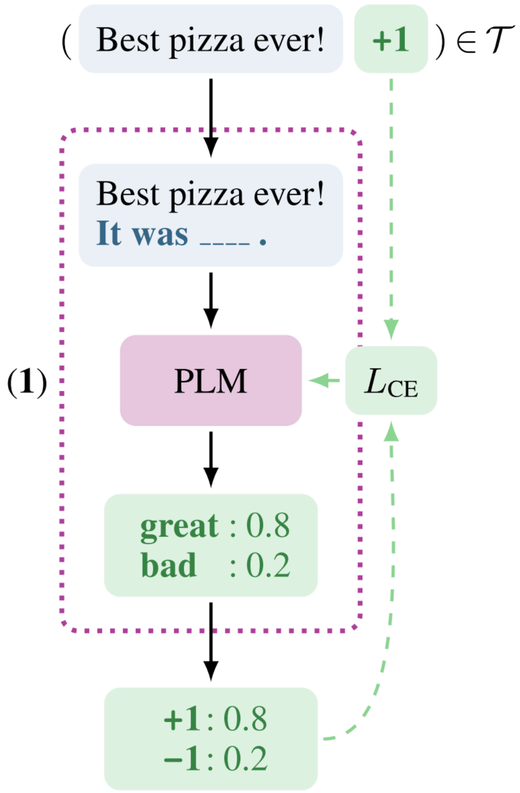
- Different prompts may lead to drastically different results
  - Liu et al. (2021)

| Prompt | P@1 |
|---|---|
| [X] is located in [Y]. *(original)* | 31.29 |
| [X] is located in which country or state? [Y]. | 19.78 |
| [X] is located in which country? [Y]. | 31.40 |
| [X] is located in which country? In [Y]. | 51.08 |

# Using Multiple Prompts

## Pattern-exploiting Training (PET; Schick & Schütze, 2021)

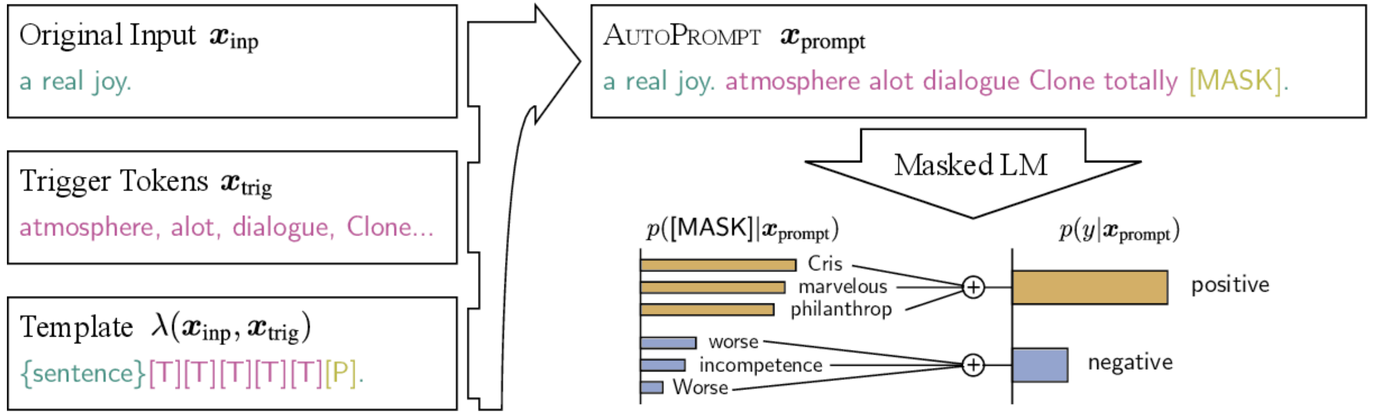
- Use an ensemble of multiple prompts
- Use the resulting classifier to label additional data
  - Retrain the model on the new data



# Learning Automatic Prompts

## Shin et al. (2020)

- Trigger tokens are initialized at [MASK] and estimated in an iterative process
  - Searching for the most useful prompt
- Resulting prompt typically doesn't mean anything



# Continuous Prompts

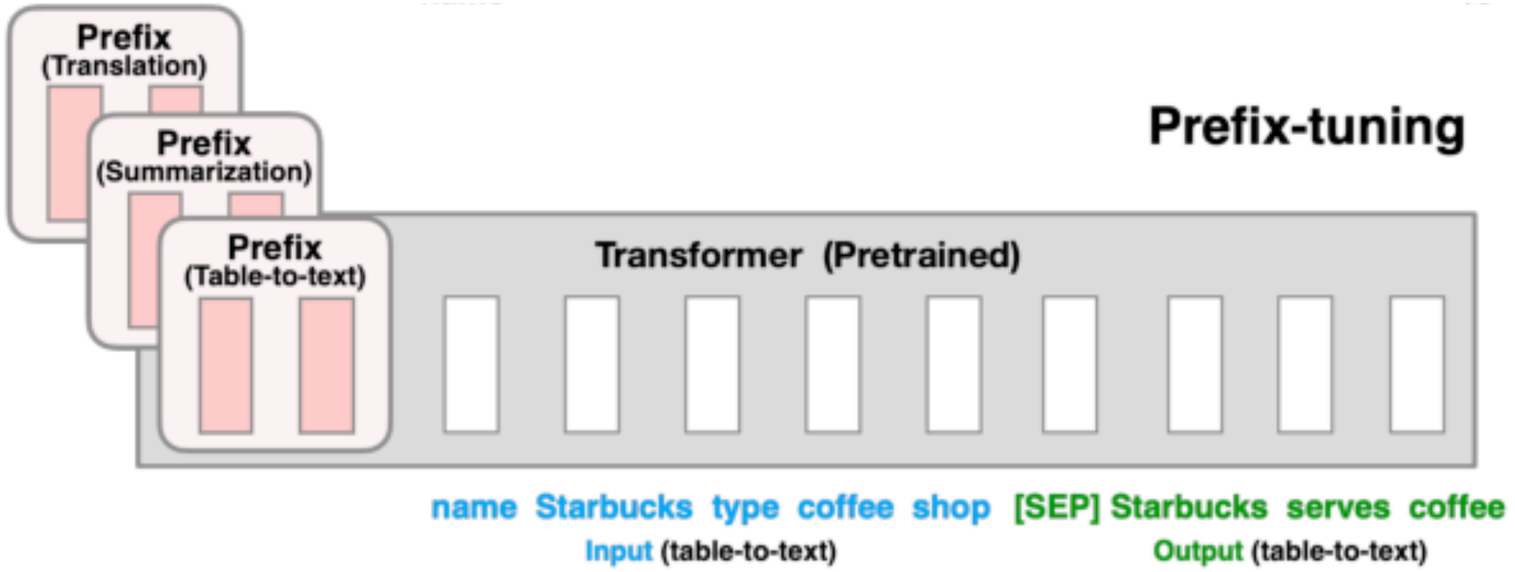
## Learn Special Prompt Embeddings (Zhong et al., 2021)

| Method | Prompt | Data-driven? |
|---|---|---|
| LAMA (Petroni et al., 2019) | `[X]` is `[MASK]` citizen | ✗ |
| LPAQA (Jiang et al., 2020) | `[X]` is a citizen of `[MASK]` | ✓ |
| AUTOPROMPT (Shin et al., 2020) | `[X]` m$^3$ badminton pieces internationally representing `[MASK]` | ✓ |
| OPTIPROMPT | `[X]` $[V]_1$ $[V]_2$ $[V]_3$ $[V]_4$ $[V]_5$ `[MASK]` | ✓ |
| OPTIPROMPT (manual) | `[X]` $[V]_1 :=$ is `[MASK]` $[V]_2 :=$ citizen | ✓ |

# Prefix-based Prompts

## Li ang Liang (2021)

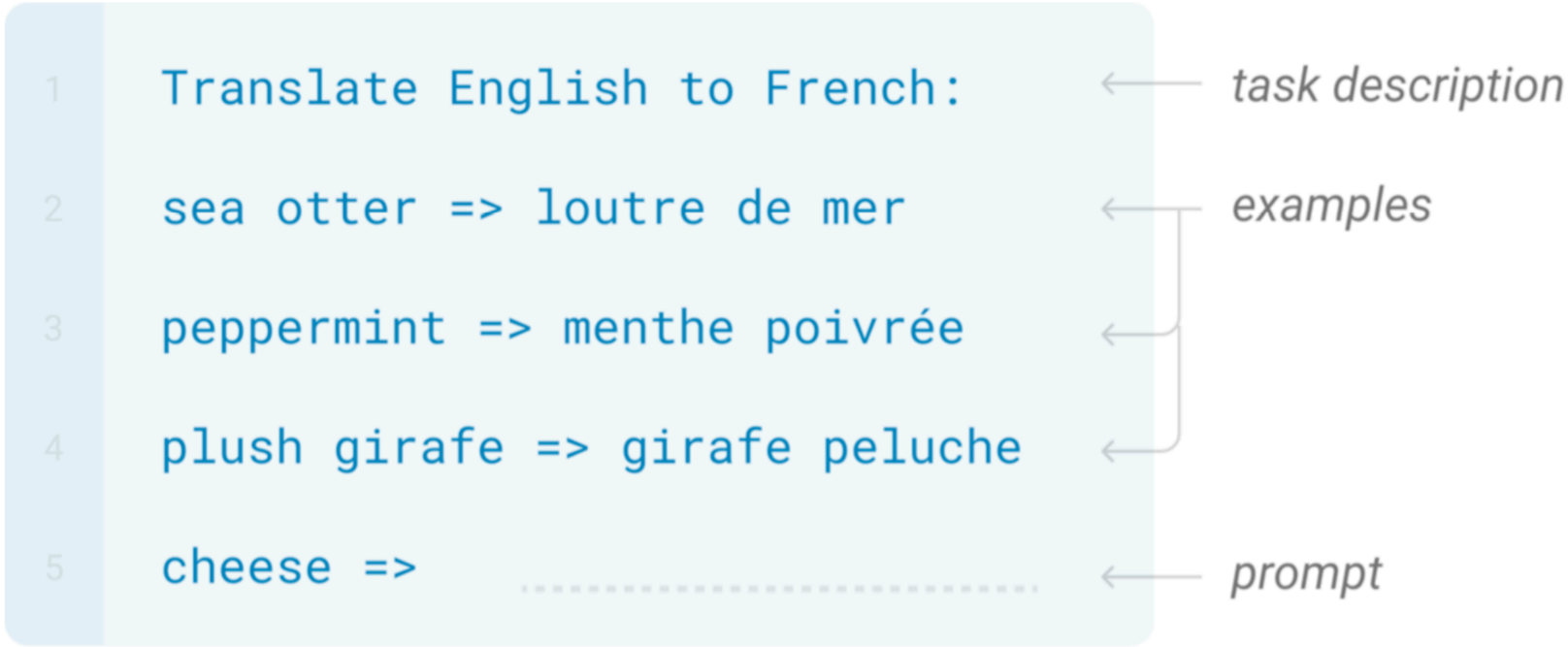
- Add soft prompts to the beginning of the input
  - Prefix tokens are represented as *free parameters*



# Why Learn Anything?

## In-context Learning (ICL; Brown et al., 2020)

- Provide annotated examples (aka *demonstrations*) as part of the context
- **No** parameter updates



# *N*-shot Learning

**Important:**
N-shot learning can be applied in many setups, not just ICL

## Few-shot learning

```
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
```

## One-shot learning

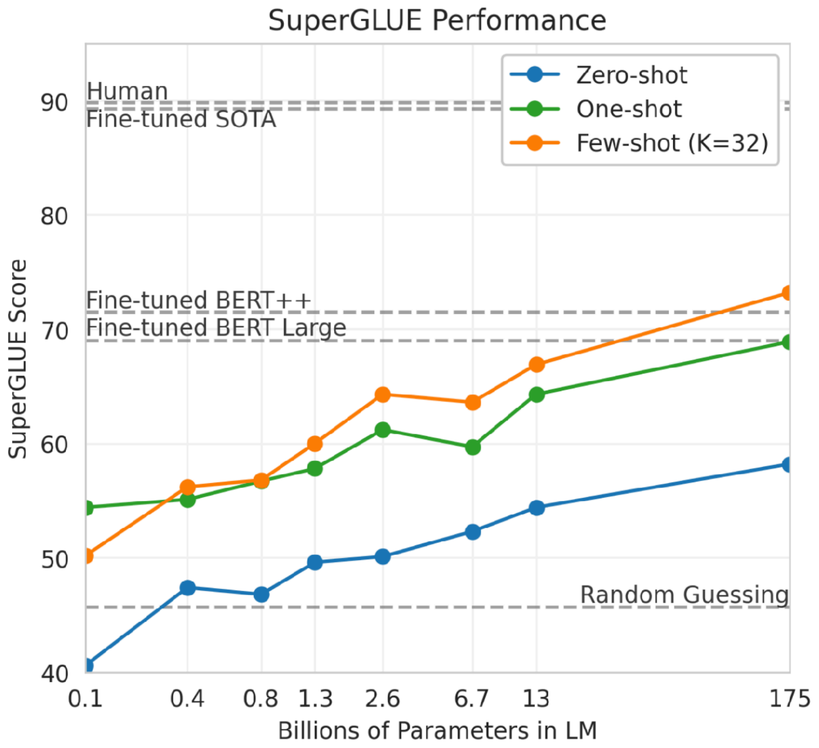
```
1  Translate English to French:          ← task description
2  sea otter => loutre de mer            ← example
3  cheese =>                             ← prompt
```

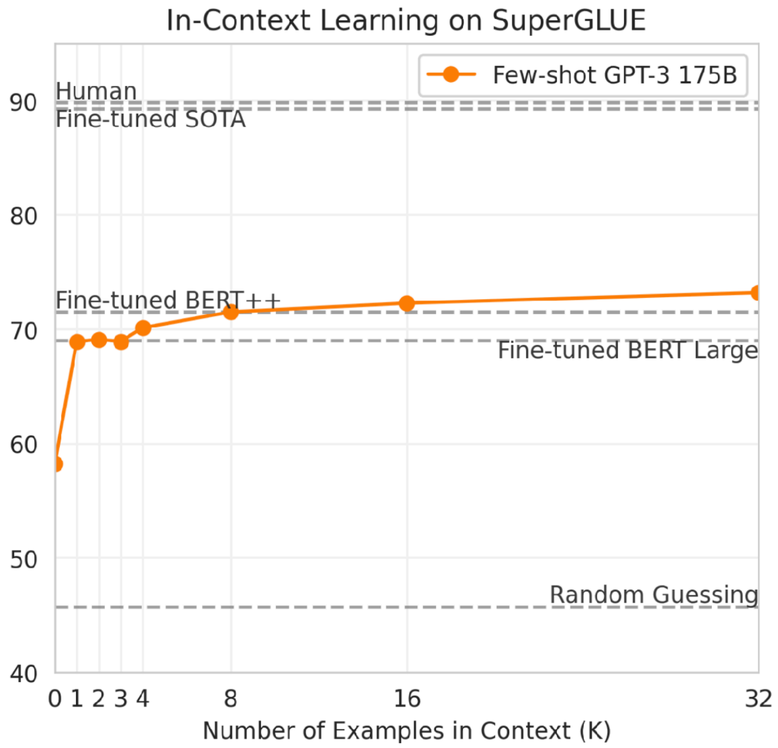
## Zero-shot learning

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

# How Few?

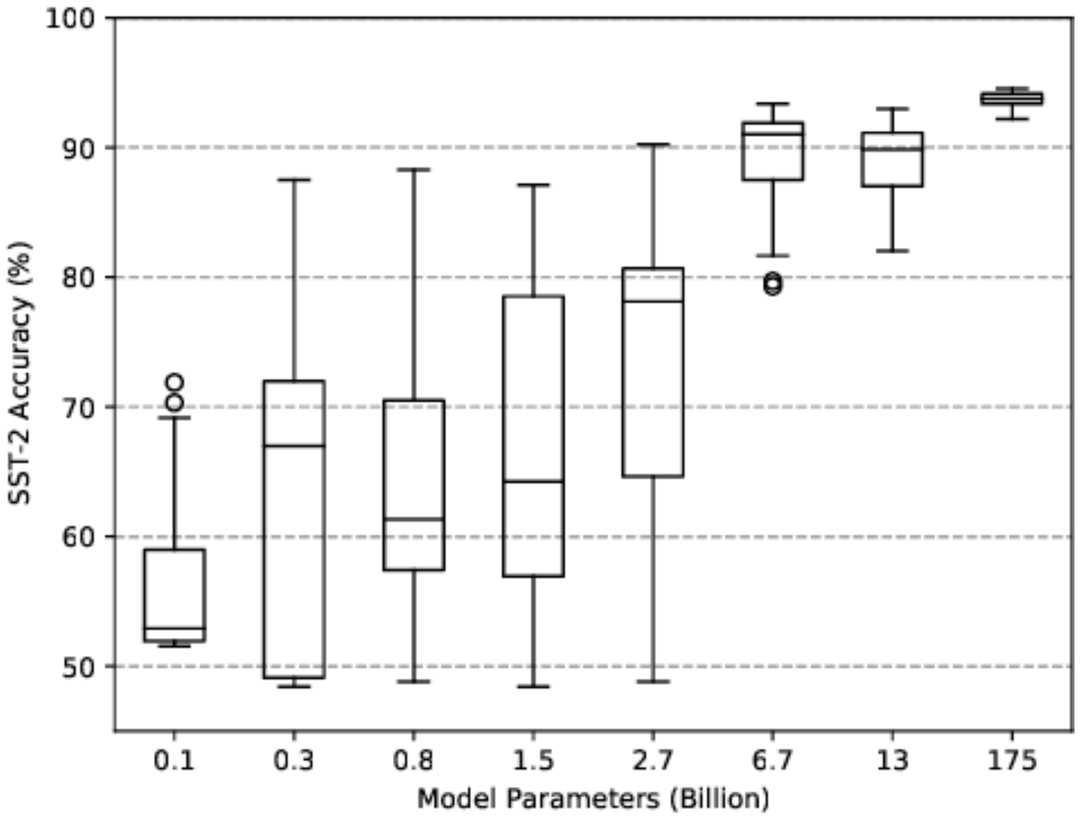
- Limited by the model's context window
  - 512 for standard models
  - Up to 2048 or even 8192 for very large models
- The more the merrier!
  - But for very large models, a few is enough?



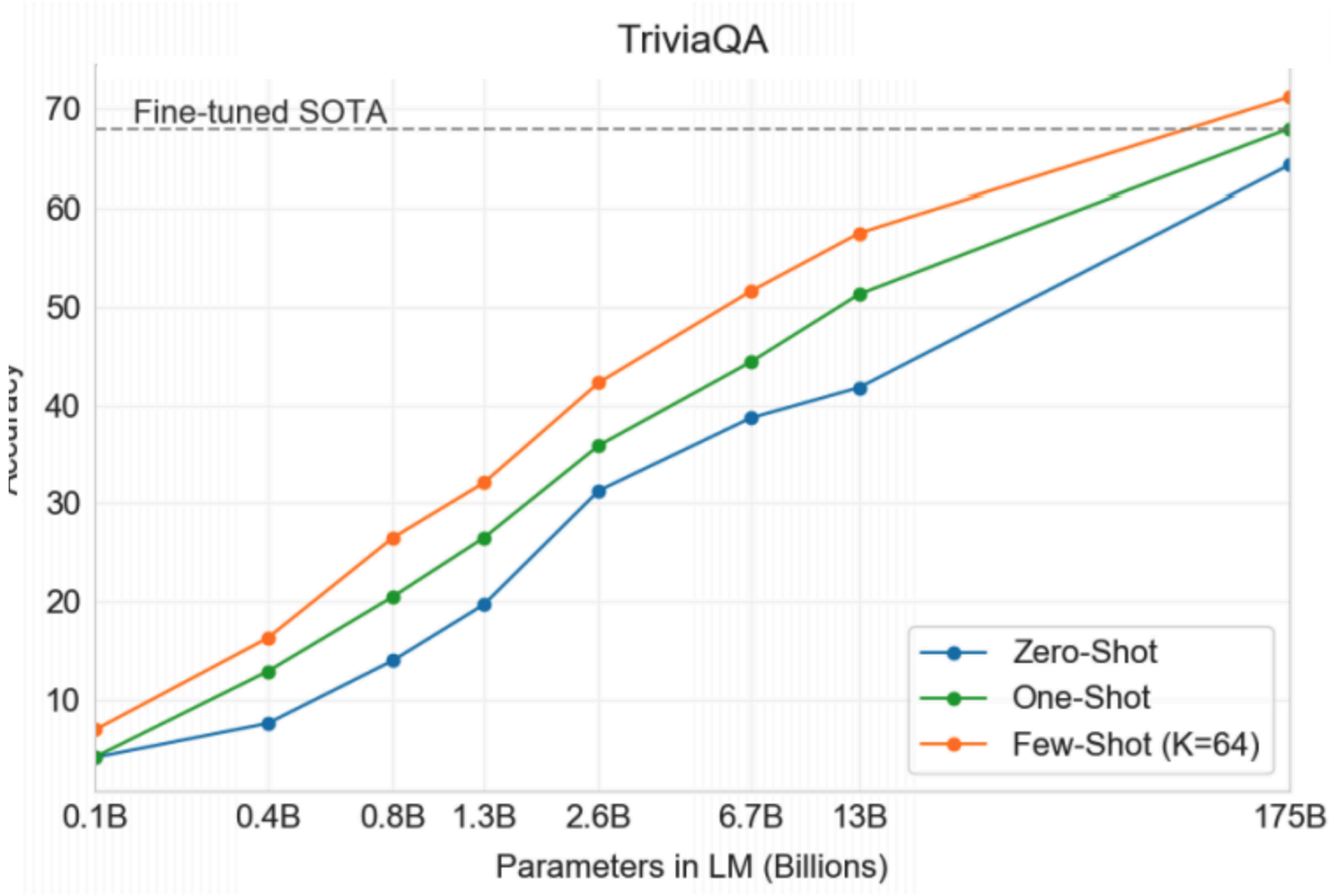



# The Demonstrations Matter

- Different demonstrations may lead to drastically different results
  - Lu et al. (2022)
- Order matters too!



# Prompting Works Really Well



# RAFT

| Rank | Submitter | Submission Name | Submission Date | Overall | Ade Corpus V2 | Banking 77 | Neurips Impact Statement Risks | One Stop English | Overruling | Semiconductor Org Types | Systematic Review Inclusion | Tai Safety Research | Terms Of Service | Tweet Eval Hate | Twitter Complaints |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Anonymous | Flan-T5 | Nov 19, 2022 | 0.773 | 0.837 | 0.647 | 0.780 | 0.847 | 0.942 | 0.917 | 0.687 | 0.703 | 0.728 | 0.517 | 0.892 |
| 2 | AaronLi | yiwise | Jul 08, 2022 | 0.768 | 0.856 | 0.695 | 0.839 | 0.698 | 0.944 | 0.906 | 0.493 | 0.737 | 0.749 | 0.647 | 0.883 |
| 3 | jtmohta | T-Few | May 06, 2022 | 0.758 | 0.804 | 0.695 | 0.833 | 0.676 | 0.950 | 0.915 | 0.508 | 0.736 | 0.750 | 0.586 | 0.879 |
| 4 | Anonymous | AuT-Few | Dec 17, 2022 | 0.747 | 0.846 | 0.587 | 0.898 | 0.770 | 0.963 | 0.801 | 0.620 | 0.742 | 0.738 | 0.350 | 0.901 |
| 5 | OE-Heart | yiwise | Jun 13, 2022 | 0.738 | 0.793 | 0.636 | 0.833 | 0.643 | 0.948 | 0.907 | 0.509 | 0.693 | 0.725 | 0.545 | 0.886 |
| 6 | ought | Human baseline (crowdsourced) | Aug 27, 2021 | 0.735 | 0.830 | 0.607 | 0.857 | 0.646 | 0.917 | 0.908 | 0.468 | 0.609 | 0.627 | 0.722 | 0.897 |
| 7 | AaronLi | yiwise | Jun 26, 2022 | 0.733 | 0.856 | 0.464 | 0.839 | 0.544 | 0.944 | 0.906 | 0.493 | 0.737 | 0.749 | 0.647 | 0.878 |
| 8 | moshew | SetFit300 | Jul 16, 2022 | 0.713 | 0.799 | 0.632 | 0.859 | 0.760 | 0.930 | 0.769 | 0.503 | 0.664 | 0.604 | 0.487 | 0.831 |
| 9 | JeanneRbs | IAL-602-v2 | Aug 19, 2022 | 0.709 | 0.814 | 0.571 | 0.796 | 0.796 | 0.915 | 0.663 | 0.606 | 0.693 | 0.575 | 0.521 | 0.850 |
| 10 | JeanneRbs | IAL-602 | Jul 28, 2022 | 0.706 | 0.821 | 0.549 | 0.776 | 0.796 | 0.910 | 0.607 | 0.606 | 0.685 | 0.637 | 0.521 | 0.855 |

# But not Fully there yet

| 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 |

# Outline

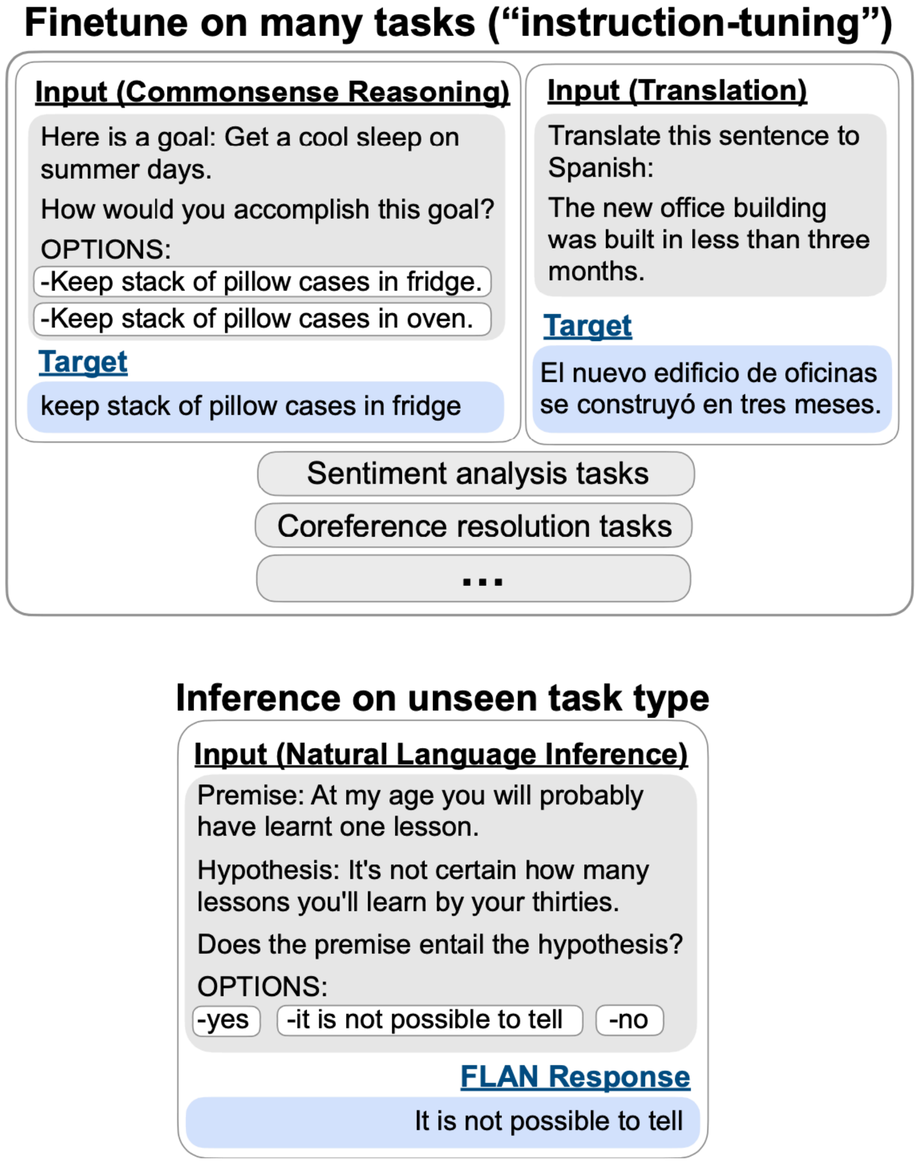
## Large Language Models

- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science

# Instruction-tuning

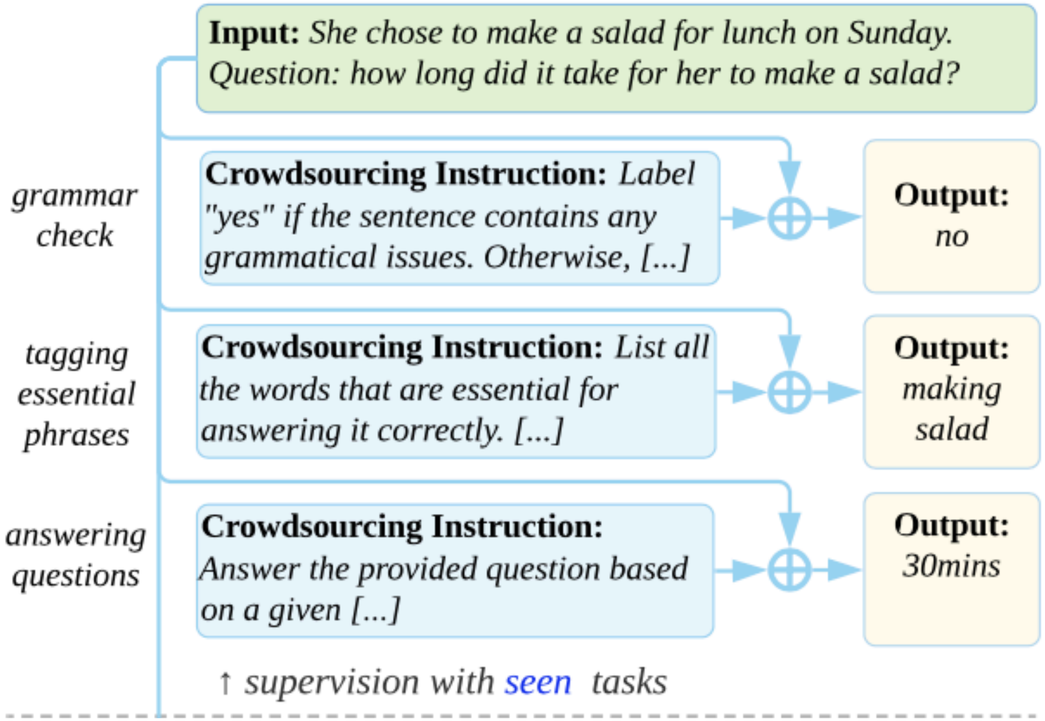
## FLAN (Wei et al., 2022)

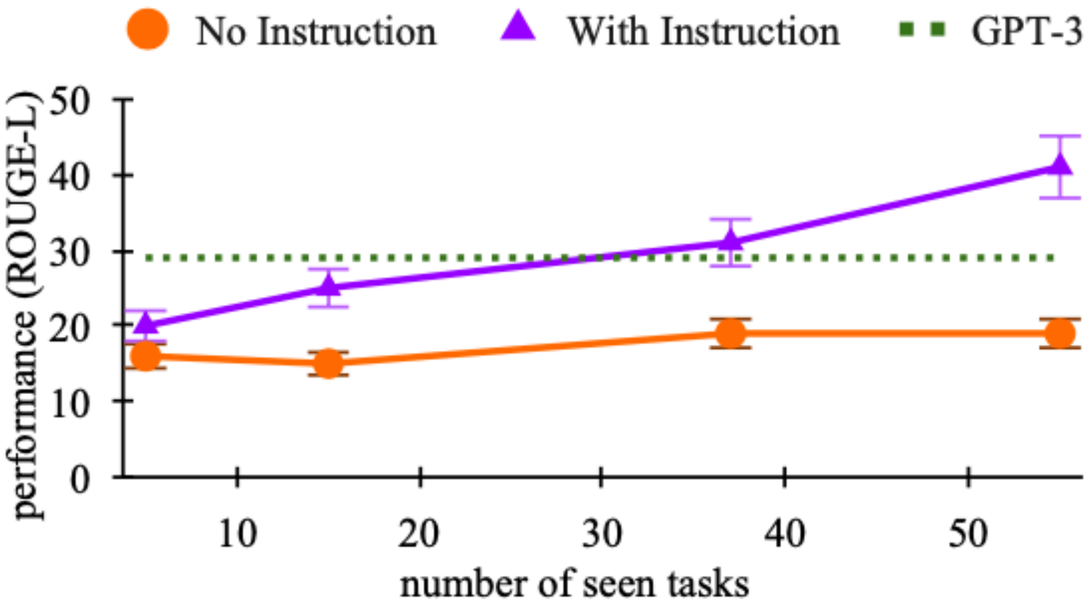
- Finetune a pretrained LM on a mixture of tasks phrased as *instructions*
- At inference time, evaluate on an unseen task type
- Substantially improve zero-shot performance
- An increasingly popular approach
  - T0 (Sanh et al., 2022)
  - **Natural Instructions (Mishra et al., 2022)**
  - SuperNatural Instructions (Wang et al., 2022)
  - **InstructGPT (Ouyang et al., 2022)**



# Natural Instructions

## Mishra et al. (2022)

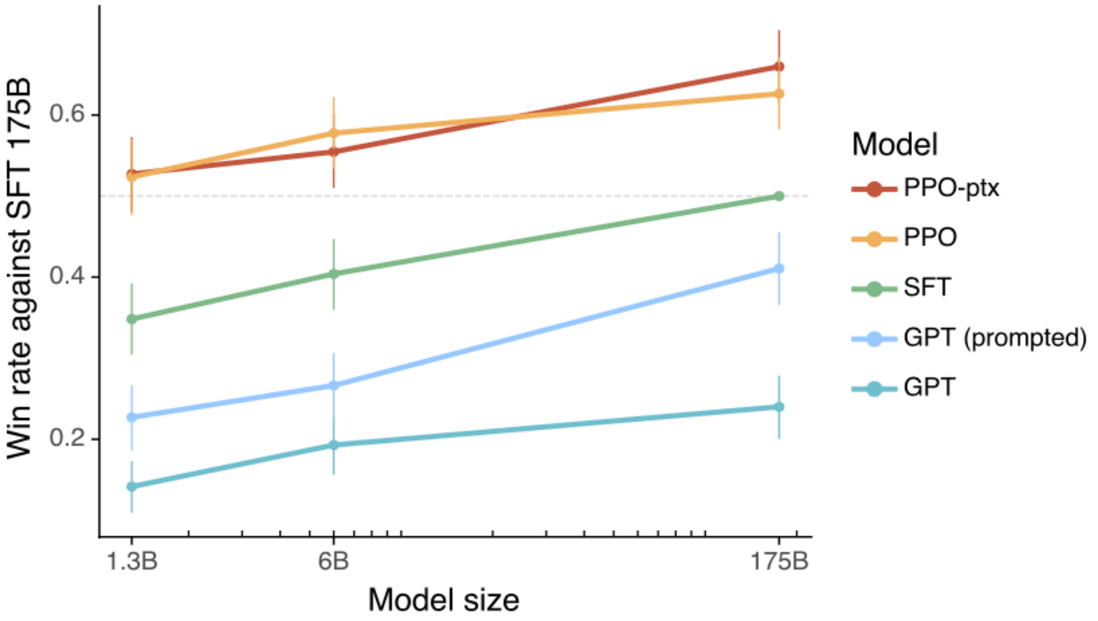


# Human Feedback Reinforcement Learning (RLHF)

## InstructGPT (Ouyang et al., 2022)

- Use real prompts fed to GPT3 to generate instructions
- Use human annotators to evaluate model outputs
- Use annotations to both
  - Finetune base GPT3
  - Train a reward model to identify “good” outputs
- Train an RL model to generate “good” outputs




GPT: zero-shot
GPT (prompted): few-shot ICL
SFT: supervised finetuning
PPO, PPO-ptx: 2 variants of instructGPT

# InstructGPT Illustrated

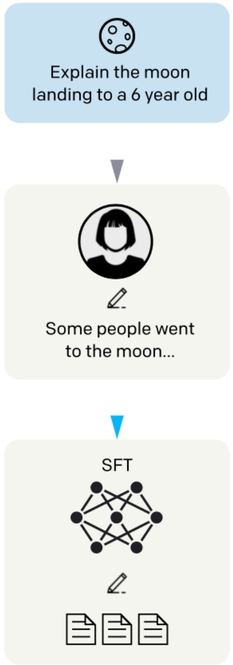
Step 1

**Collect demonstration data, and train a supervised policy.**

A prompt is sampled from our prompt dataset.

A labeler demonstrates the desired output behavior.

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



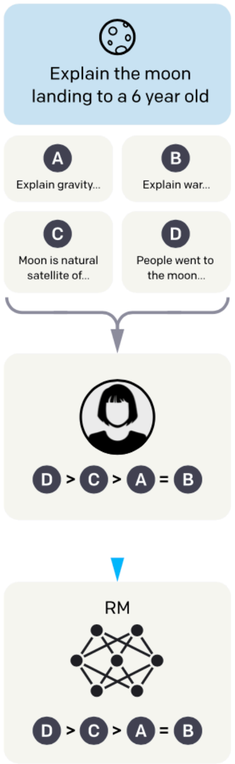

Step 2

**Collect comparison data, and train a reward model.**

A prompt and several model outputs are sampled.

A labeler ranks the outputs from best to worst.

This data is used to train our reward model.




Step 3

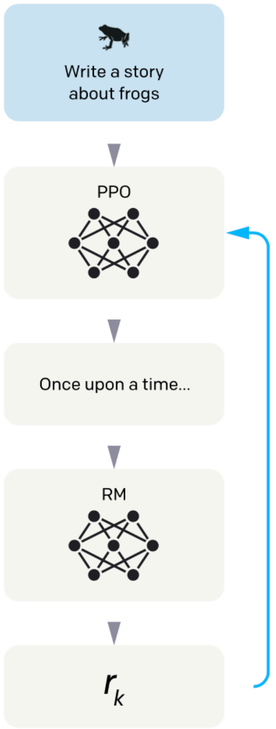
**Optimize a policy against the reward model using reinforcement learning.**

A new prompt is sampled from the dataset.

The policy generates an output.

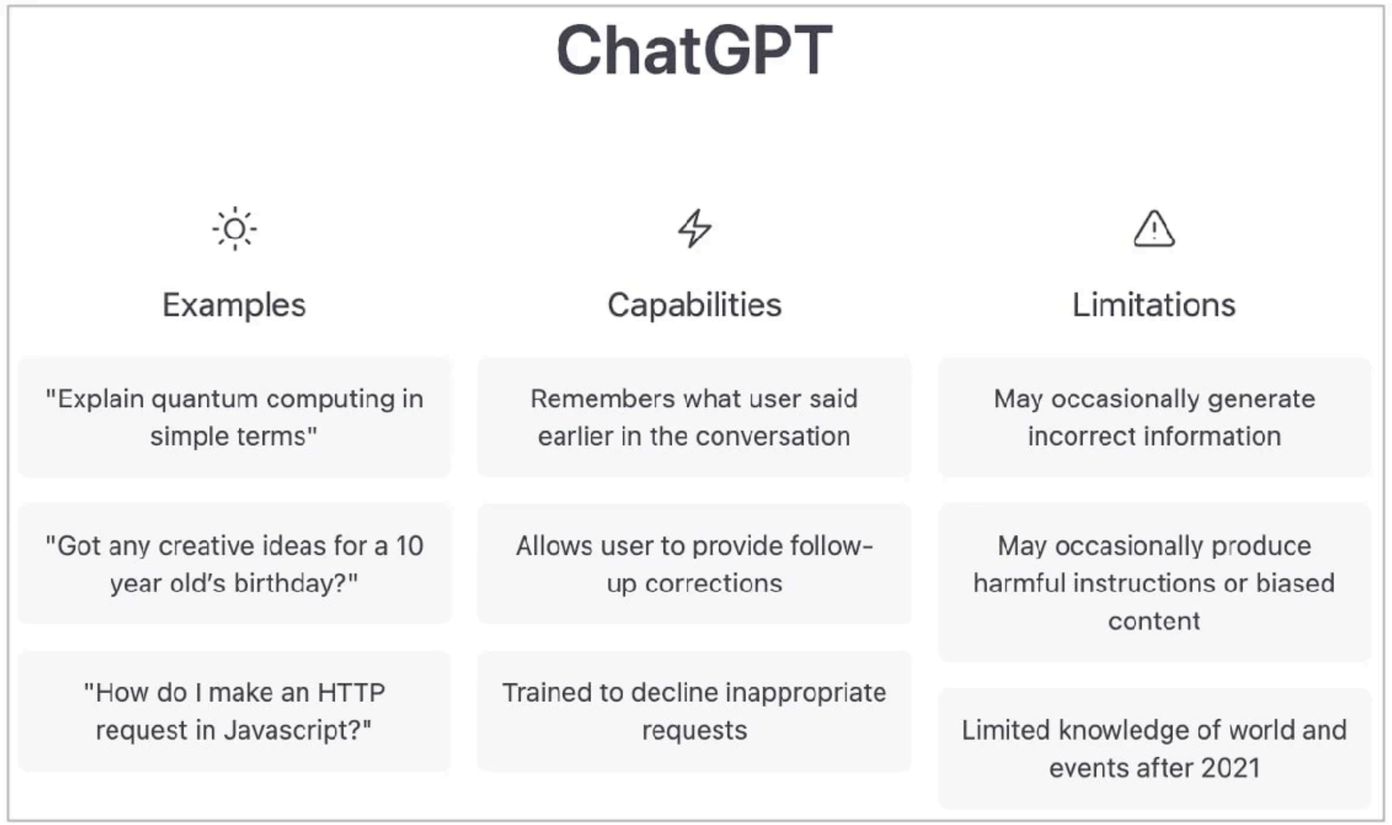
The reward model calculates a reward for the output.

The reward is used to update the policy using PPO.



# ChatGPT

## <No Research Paper>



# GPT4

## <No Research Paper>

- Improved (?) capabilities
  - Reasoning
  - Coding
  - Passing Bar exams, Biology Olympiad
  - Multi-modality
- AGI?
  - Bubeck et al. (2023)

**Sparks of Artificial General Intelligence: Early experiments with GPT-4**

Sébastien Bubeck Varun Chandrasekaran Ronen Eldan Johannes Gehrke
Eric Horvitz Ece Kamar Peter Lee Yin Tat Lee Yuanzhi Li Scott Lundberg
Harsha Nori Hamid Palangi Marco Tulio Ribeiro Yi Zhang

Microsoft Research

# Outline

## Large Language Models

- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science

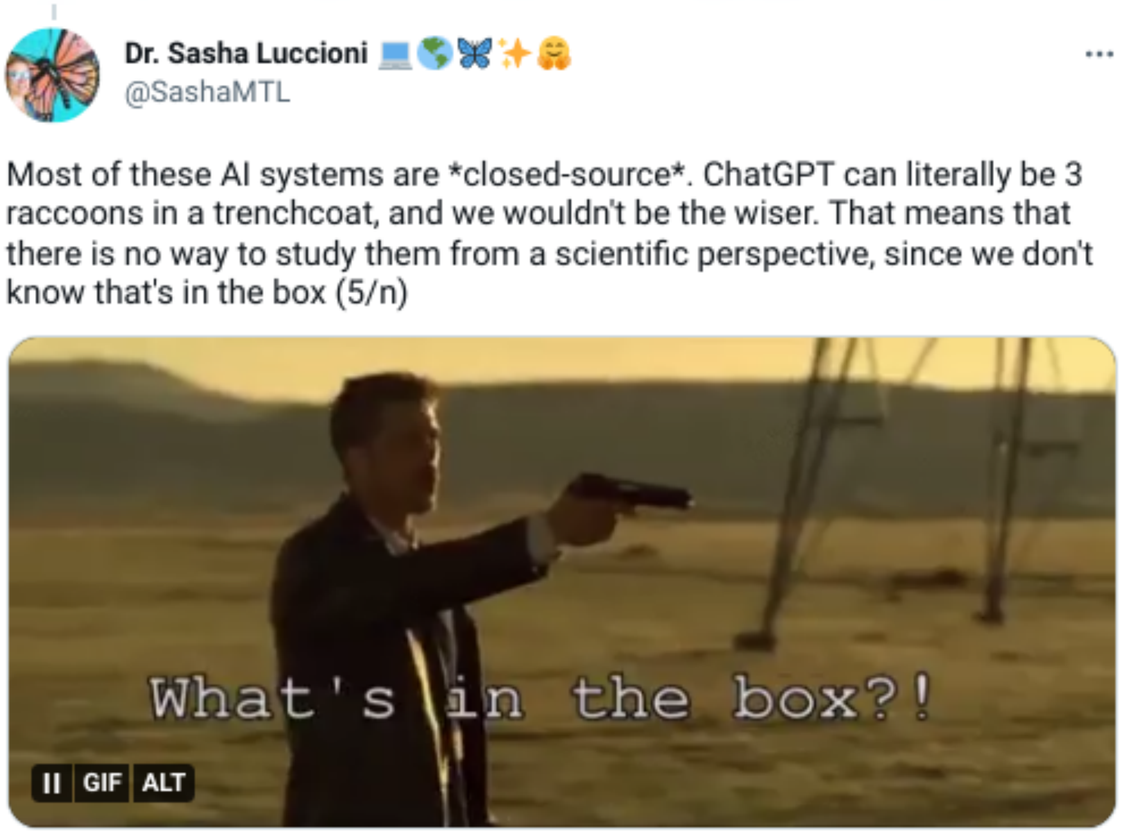
# GPT4

## Model Architecture

This report focuses on the capabilities, limitations, and safety properties of GPT-4. GPT-4 is a Transformer-style model [39] pre-trained to predict the next token in a document, using both publicly available data (such as internet data) and data licensed from third-party providers. The model was then fine-tuned using Reinforcement Learning from Human Feedback (RLHF) [40]. Given both the competitive landscape and the safety implications of large-scale models like GPT-4, this report contains no further details about the architecture (including model size), hardware, training compute, dataset construction, training method, or similar.

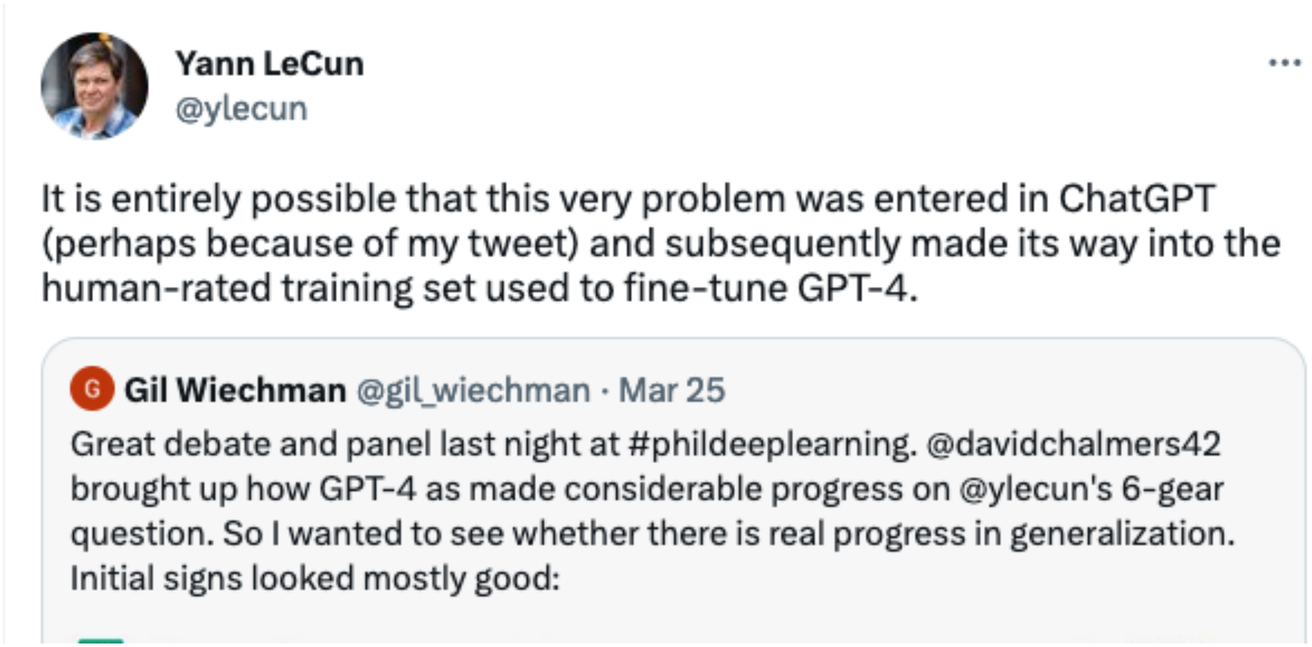
# Closed Models and Research

# You don’t Know what you are Comparing to



# The Model may Use your Data to Fine-tune the Next Version!

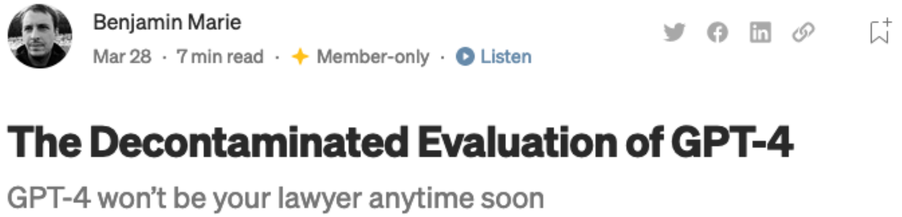
## Or it might have already Seen your Test Data!



**Yann LeCun**
@ylecun

It is entirely possible that this very problem was entered in ChatGPT (perhaps because of my tweet) and subsequently made its way into the human-rated training set used to fine-tune GPT-4.

> **Gil Wiechman** @gil_wiechman · Mar 25
> Great debate and panel last night at #phildeeplearning. @davidchalmers42 brought up how GPT-4 as made considerable progress on @ylecun's 6-gear question. So I wanted to see whether there is real progress in generalization. Initial signs looked mostly good:

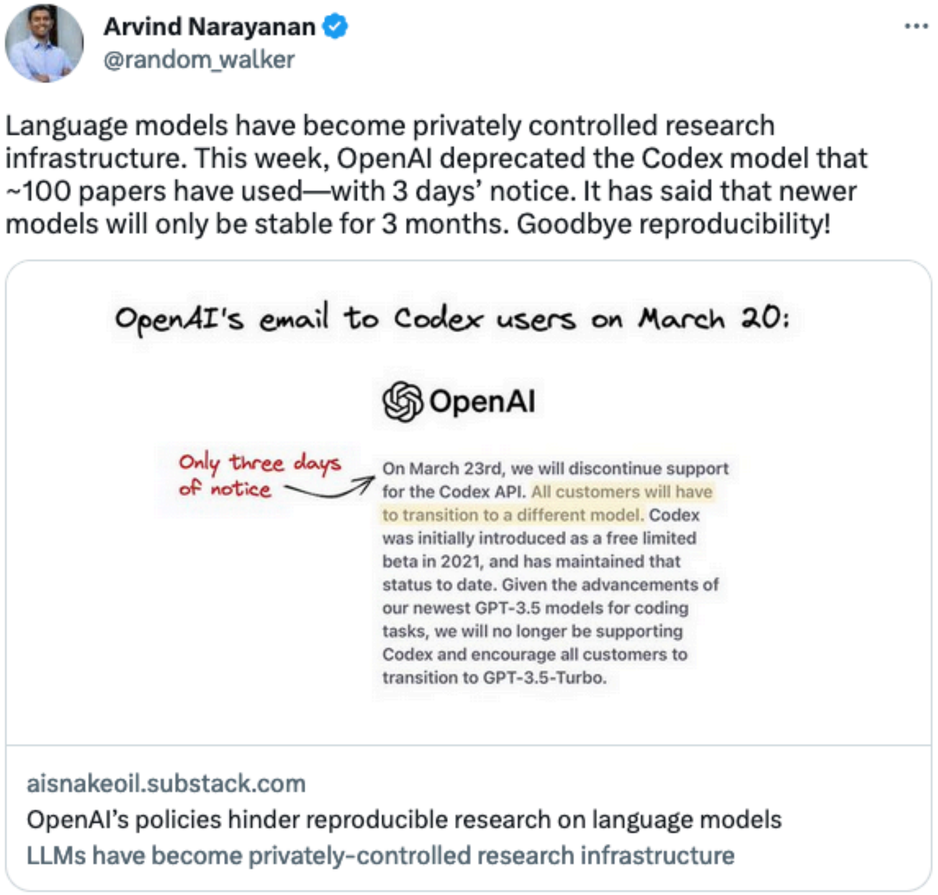


Benjamin Marie
Mar 28 · 7 min read · Member-only · Listen

**The Decontaminated Evaluation of GPT-4**

GPT-4 won't be your lawyer anytime soon

# Closed Models may be Deprecated without Notice



# Conclusion: Closed Models are *not* Pre-requisite Baselines

## Closed AI Models Make Bad Baselines

24 minute read

> *This post was authored by Anna Rogers, with much invaluable help and feedback from Niranjan Balasubramanian, Leon Derczynski, Jesse Dodge, Alexander Koller, Sasha Luccioni, Maarten Sap, Roy Schwartz, Noah A. Smith, Emma Strubell (listed alphabetically)*
> *Header image credit: Sasha Luccioni*

https://hackingsemantics.xyz/2023/closed-baselines/

# Summary

## Large Language Models

- Encoder-only models
- Using LLMs
  - Feature extraction vs. Fine-tuning
  - Working with different tasks
- Generative Models
  - Encoder-decoder and Decoder-only models
- Prompting
  - Discrete and Continuous prompts
  - In context learning
- Instruction Tuning
- Closed Models and Science