Pre-training, Fine-tuning, and Prompting

CS 288 Spring 2026

UC Berkeley

cal-cs288.github.io/sp26

Berkeley EECS

BAIR

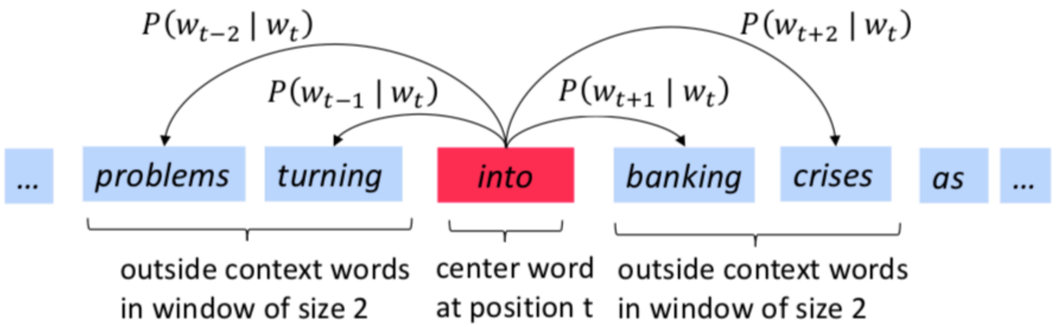
Course Progress So Far

- We've taken a long journey in neural architectures—from perceptrons and MLPs, through RNNs and LSTMs, and finally to Transformers (including modern variants).
- Today, we'll learn
    - Pre-training: The biggest breakthrough in NLP
    - And how it is used in practice: fine-tuning and prompting

Contextualized Word Embeddings

Recap: word2vec

- Goal: Vector representations for a word (so that the neural model can understand)
- Key idea: Use each word to predict other words in its context

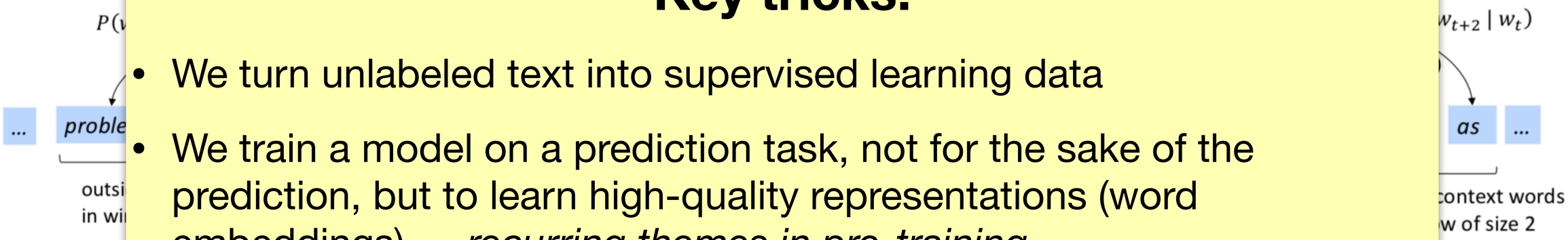


Our goal is to find parameters that can maximize

$$P(\text{problems} \mid \text{into}) \times P(\text{turning} \mid \text{into}) \times P(\text{banking} \mid \text{into}) \times P(\text{crises} \mid \text{into})$$

Recap: word2vec

- Goal: Vector representations for a word (so that the neural model can understand)
- Key idea: Use each word to predict other words in its context



Key tricks:

- We turn unlabeled text into supervised learning data
- We train a model on a prediction task, not for the sake of the prediction, but to learn high-quality representations (word embeddings) — *recurring themes in pre-training*

Our goal is to find parameters that can maximize

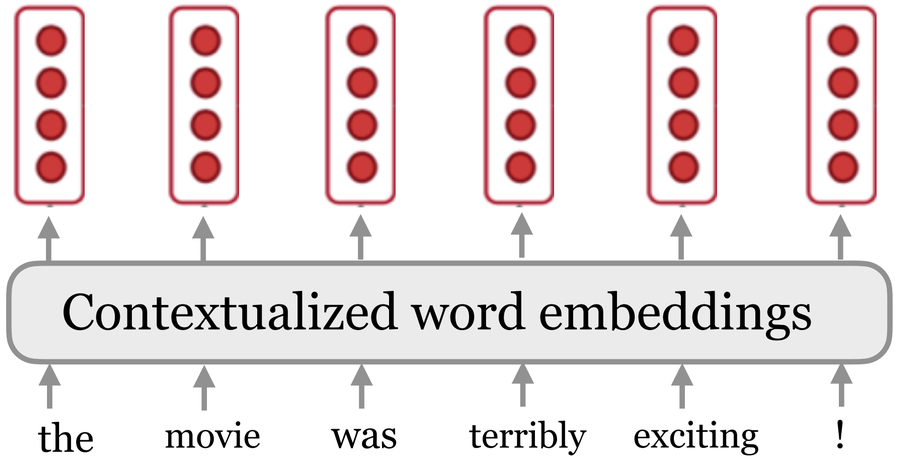
$$P(\text{problems} \mid \text{into}) \times P(\text{turning} \mid \text{into}) \times P(\text{banking} \mid \text{into}) \times P(\text{crises} \mid \text{into}) \times$$

$$P(\text{turning} \mid \text{banking}) \times P(\text{into} \mid \text{banking}) \times P(\text{crises} \mid \text{banking}) \times P(\text{as} \mid \text{banking}) \ldots$$

Limitation of Word2vec

- Word2vec: One vector for each word type (a.k.a. static embeddings)
- Can we build a vector for each word conditioned on its **context?**

$$v(\text{play}) = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix}$$



$$f : (w_1, w_2, \ldots, w_n) \longrightarrow \mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$$

ELMo!

(Peters et al, 2018): Deep contextualized word representations

Contextualized word embeddings

http://jalammar.github.io/illustrated-bert/

Contextualized word embeddings

Sent #1: Chico Ruiz made a spectacular **play** on Alusik's grounder {. . . }

Which of the following $v(\text{play})$ is expected to have the most similar vector to the first one?

(A) Olivia De Havilland signed to do a Broadway **play** for Garson {. . . }

(B) Kieffer was commended for his ability to hit in the clutch , as well as his all-round excellent **play** {. . . }

(C) {. . . } they were actors who had been handed fat roles in a successful **play** {. . . }

(D) Concepts **play** an important role in all aspects of cognition {. . . }

(B) is correct.

Contextualized word embeddings

	Source	Nearest Neighbors
GloVe	play	playing, game, games, played, players, plays, player, Play, football, multiplayer
biLM	Chico Ruiz made a spectacular play on Alusik 's grounder {...}	Kieffer , the only junior in the group , was commended for his ability to hit in the clutch , as well as his all-round excellent play .
	Olivia De Havilland signed to do a Broadway play for Garson {...}	{...} they were actors who had been handed fat roles in a successful play , and had talent enough to fill the roles competently , with nice understatement .

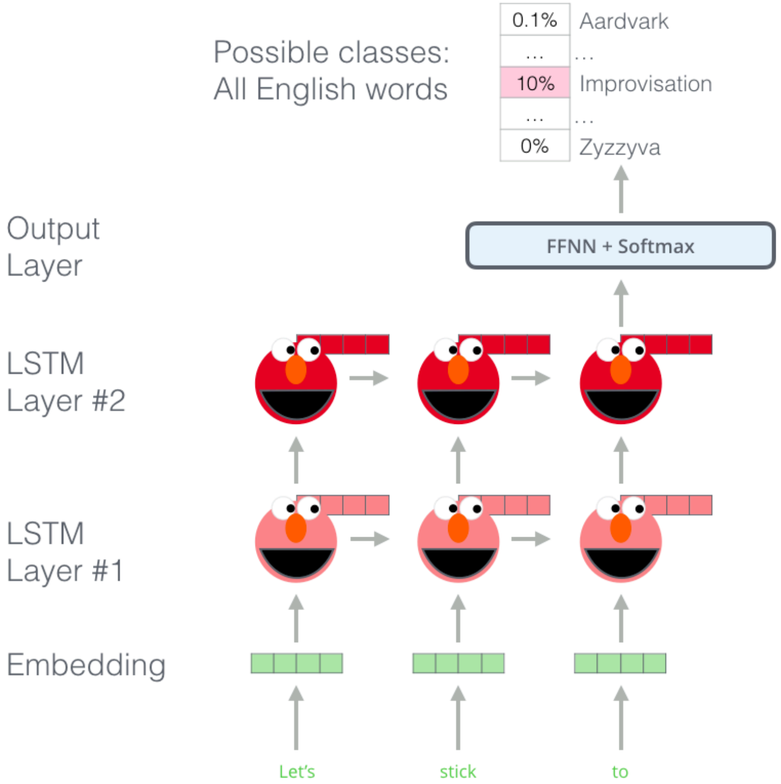
(Peters et al, 2018): Deep contextualized word representations

ELMo: Embeddings from Language Models (Feb 2018)

The key idea of ELMo: Train *two* stacked LSTM-based language models on a **large** corpus

Recap: an LSTM-based language model:

$$-\sum_{j=1}^{n-1} \log P_{\text{LM}}(w_{j+1} \mid w_1 \cdots w_j)$$



http://jalammar.github.io/illustrated-bert/

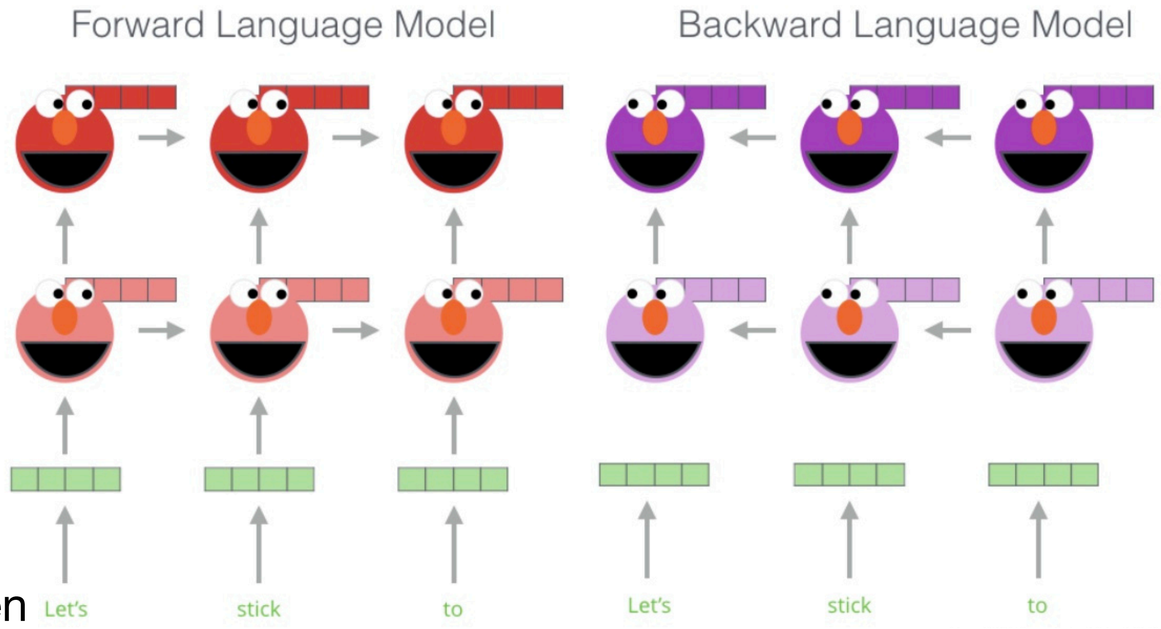
ELMo: Embeddings from Language Models (Feb 2018)

The key idea of ELMo: Train *two* stacked LSTM-based language models on a **large** corpus

ELMo uses a forward language model and a backward language model

$$-\sum_{j=1}^{n-1} \log P_{\text{forwardLM}}(w_{j+1} \mid w_1 \cdots wj)$$

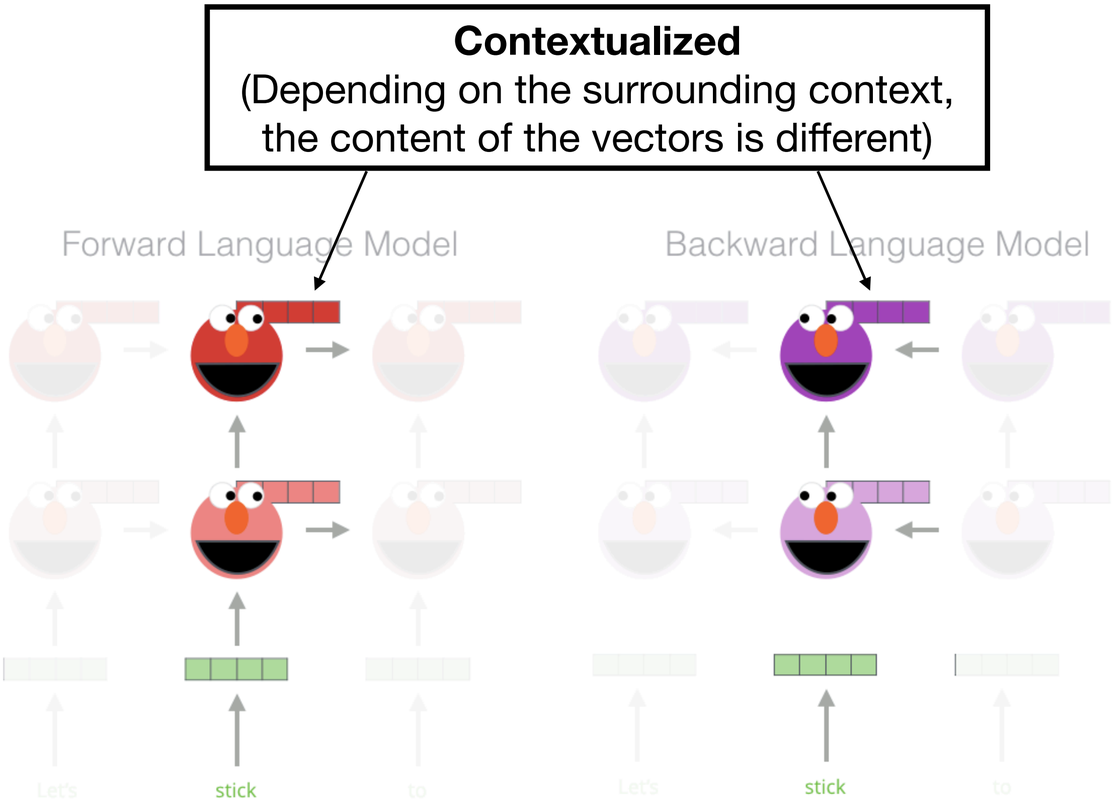
$$-\sum_{j=0}^{n} \log P_{\text{backwardLM}} P(w_j \mid w_{j+1} \cdots w_n)$$



Use the **hidden states** of the LSTMs for each token to compute a vector representation of each word

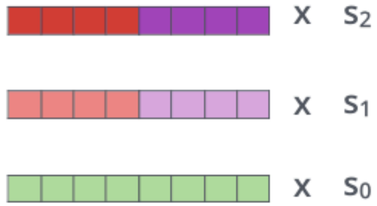
http://jalammar.github.io/illustrated-bert/

ELMo: Use case (1/2)



1- Concatenate hidden layers

2- Multiply each vector by a weight based on the task

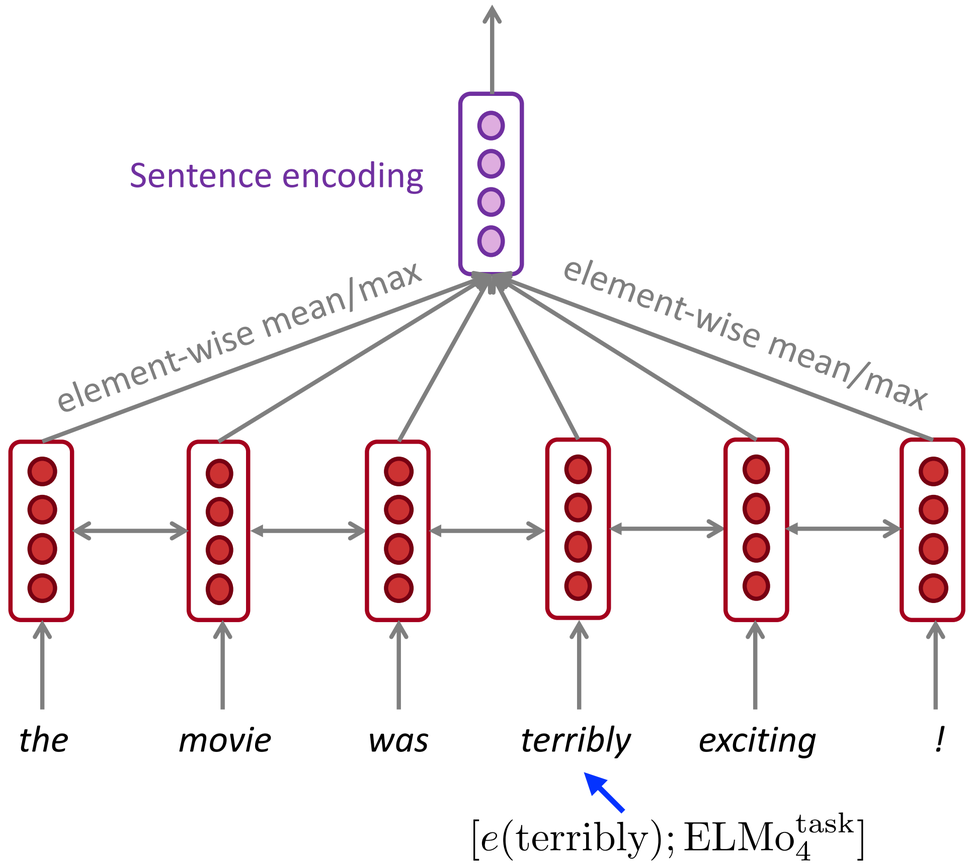


3- Sum the (now weighted) vectors

ELMo embedding of "stick" for this task in this context

http://jalammar.github.io/illustrated-bert/

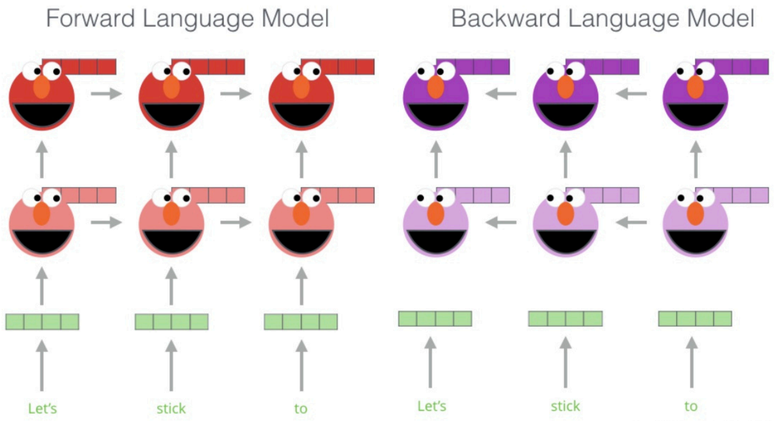
ELMo: Use case (2/2)



Example use: A BiLSTM model for sentiment classification

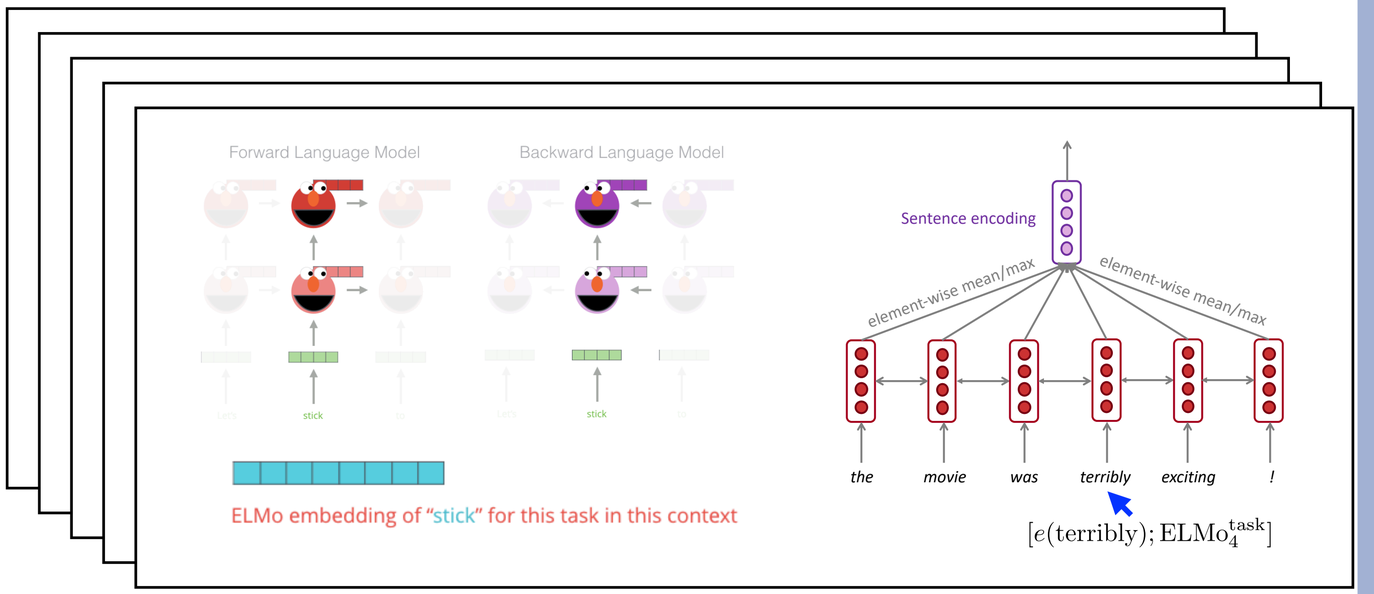
ELMo: Pre-training and the use

Pre-training (Only once, task-agnostic)



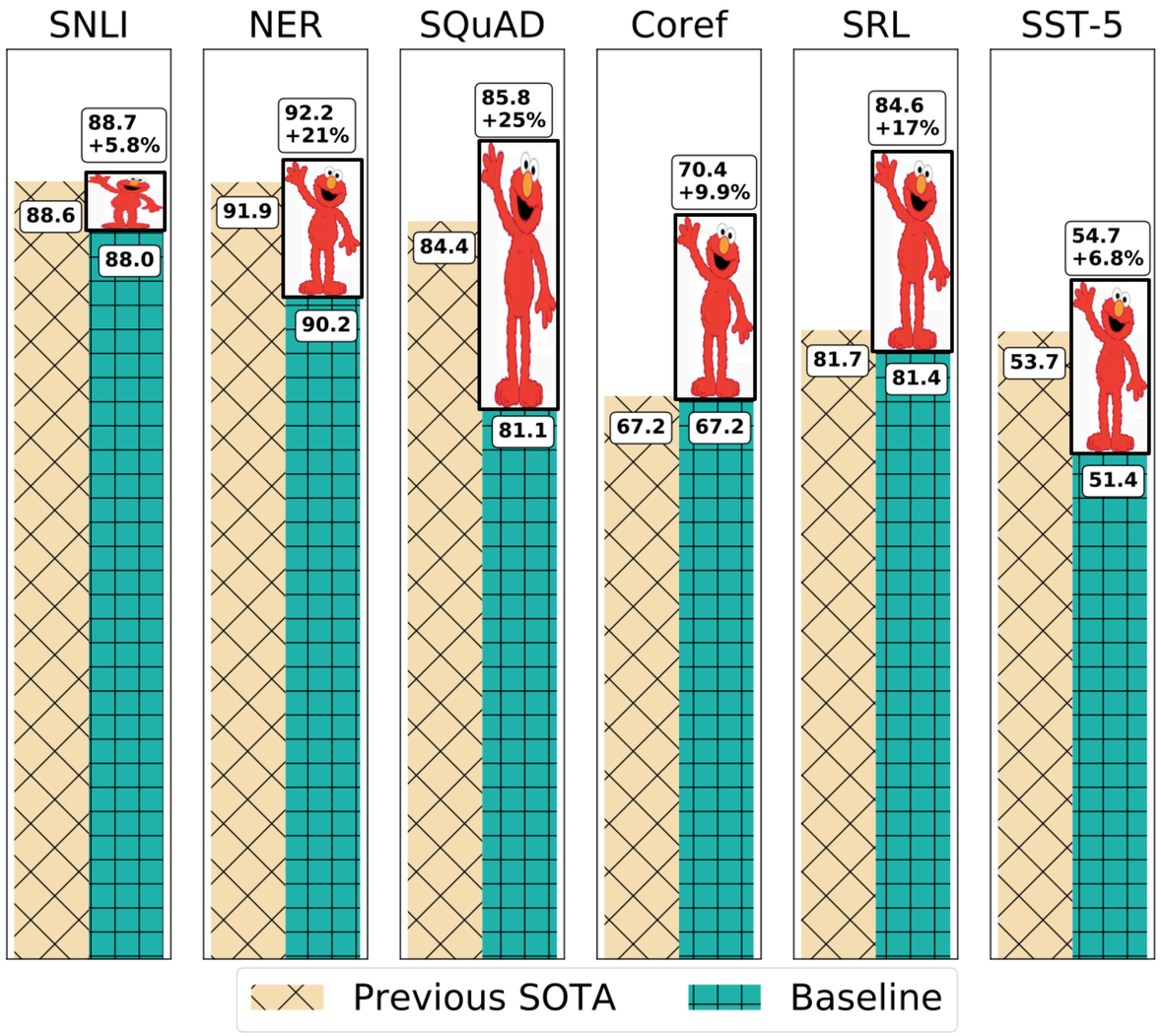
- Data: 10 epochs on 1B Word Benchmark
- Training time: 2 weeks on 3 NVIDIA GTX 1080 GPUs

Run inference to get **contextualized embeddings**, **plug them into the main model** (e.g., LSTMs), then train a model for a **specific downstream task**



- Sentiment classification
- Topic classification
- POS tagging
- Machine translation
- Question answering
- …

ELMo: Results



(Peters et al, 2018): Deep contextualized word representations

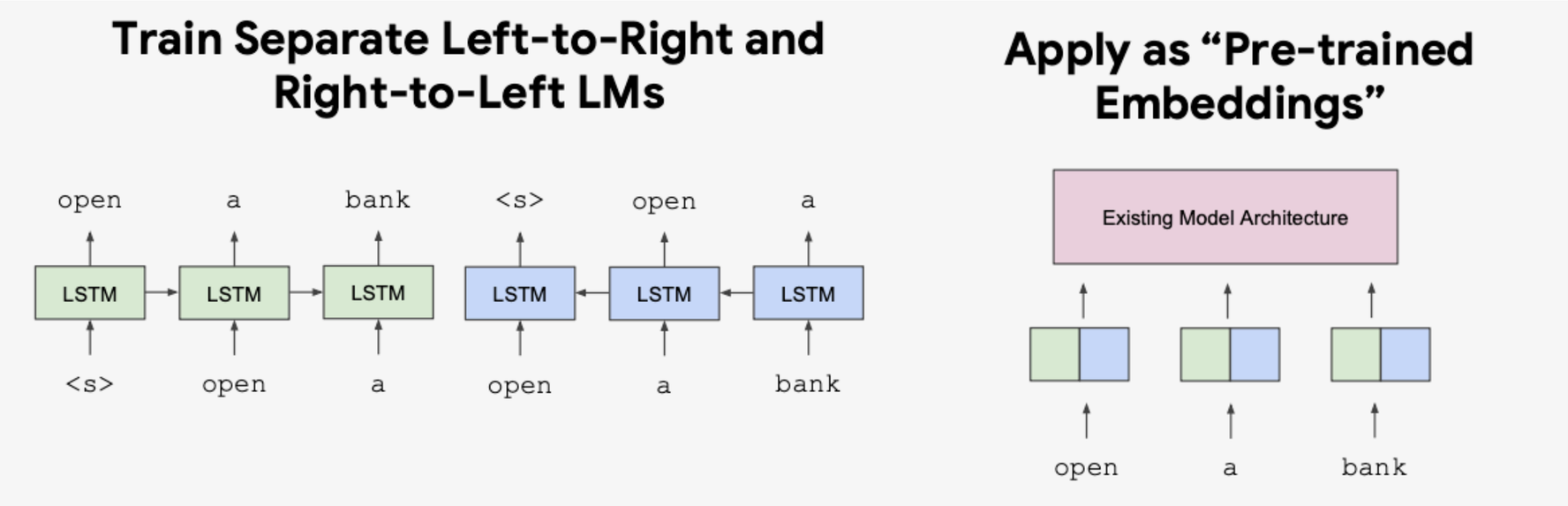
ELMo: Important notes

- Early form of pre-training
  - Pre-train once, on very large raw text data (task-agonistic)
  - Re-use the resulting artifacts (contextualized embeddings) for individual downstream tasks
- Trained on language modeling, but not for the purpose of language modeling or text generation — only to get good representations
    - Recall word2vec!
- **Important:** ELMo enhances embeddings only — the task architecture (e.g., LSTMs) remains unchanged.
  - **Major contrast with next-generation pre-training.**

Pre-training and Fine-tuning

Word2vec, ELMo: Feature-based approach

- ELMo is a feature-based approach which only produces word embeddings that can be used as **input representations** of existing neural models

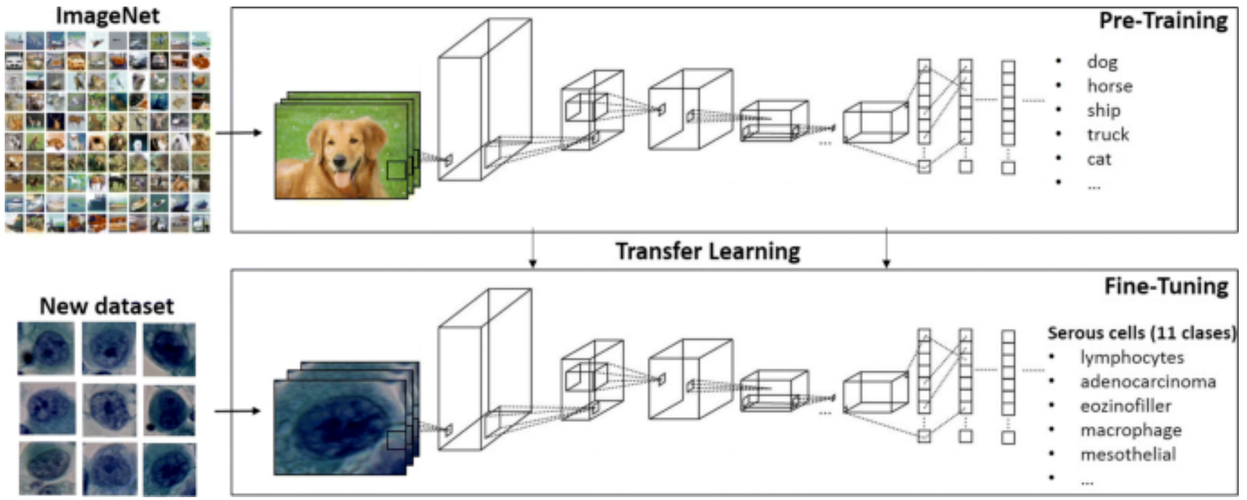


- Can we do pre-training that allows **re-using model weights**?

Pre-training in computer vision

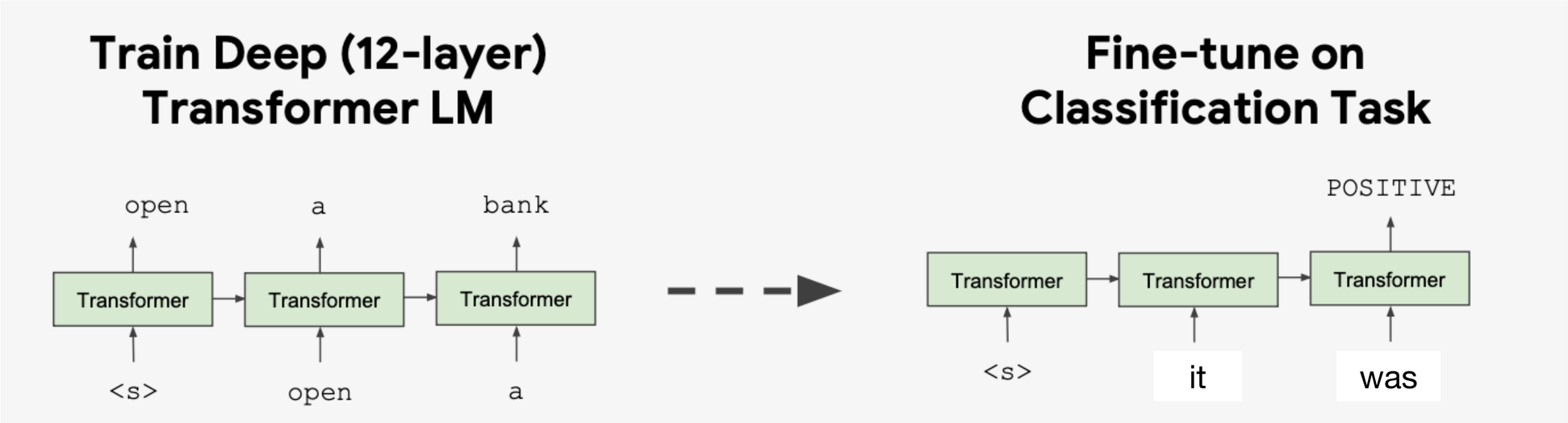
- "Pre-train" a model on a large dataset for task X, then "fine-tune" it on a dataset for task Y
- Key idea: X is somewhat related to Y, so a model that can do X will have some good neural representations for Y as well
- ImageNet pre-training is huge in computer vision: learning generic visual features for recognizing objects

Can we find some task X that can be useful for a wide range of downstream tasks Y?



Fine-tuning approaches

- GPT / BERT (2018): **Some of the first pre-trained LLMs with fine-tuning approaches**
  - Almost all model weights will be **re-used**, and only a small number of task-specific will be added for downstream tasks



Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	**BERT**, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	**GPT-2**, GPT-3, Llama

Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	**BERT**, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	**GPT-2**, GPT-3, Llama

Feb 17 lecture starts from here

CS 288 Advanced Natural Language Processing

Course website: cal-cs288.github.io/sp26

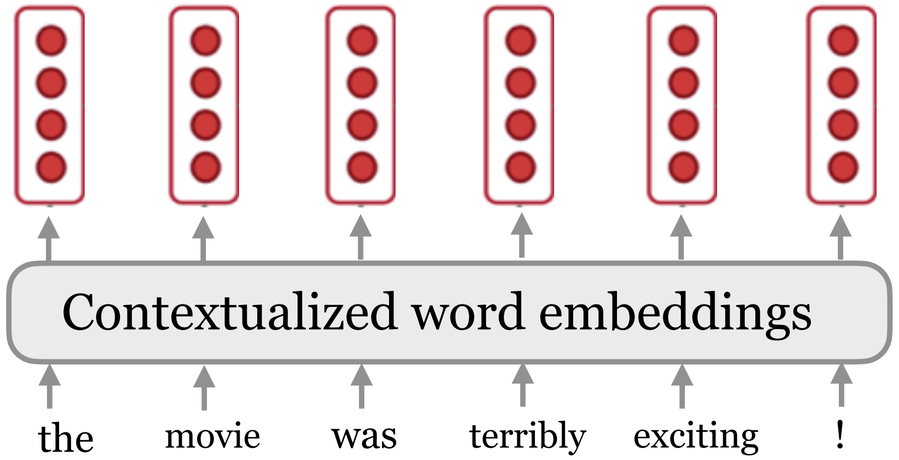
Ed: edstem.org/us/join/XvztdK

- Class starts at 15:40!
- A3/Project team formation is released on Ed!
- Reminder to refresh the lecture slides (PDFs) on the lecture day
- Lecture plans: Complete pre-training, fine-tuning, & prompting

Recap: Word2vec to ELMo

- Word2vec: One vector for each word type (a.k.a. static embeddings)
- Can we build a vector for each word conditioned on its **context?**

$$v(\text{play}) = \begin{pmatrix} -0.224 \\ 0.130 \\ -0.290 \\ 0.276 \end{pmatrix}$$



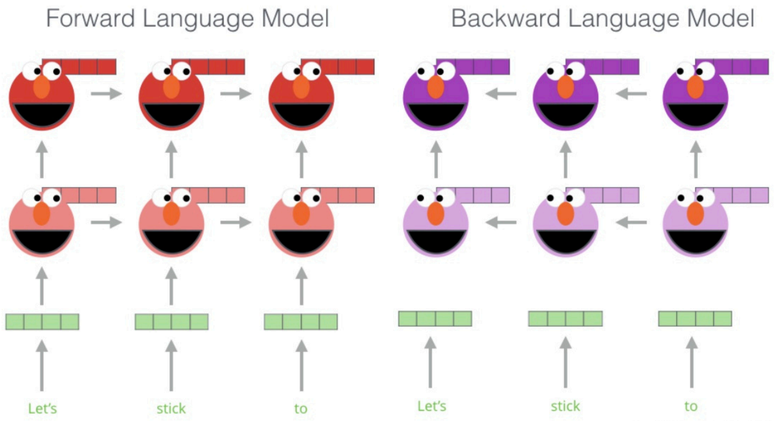
$$f : (w_1, w_2, \ldots, w_n) \longrightarrow \mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$$

ELMo!

(Peters et al, 2018): Deep contextualized word representations

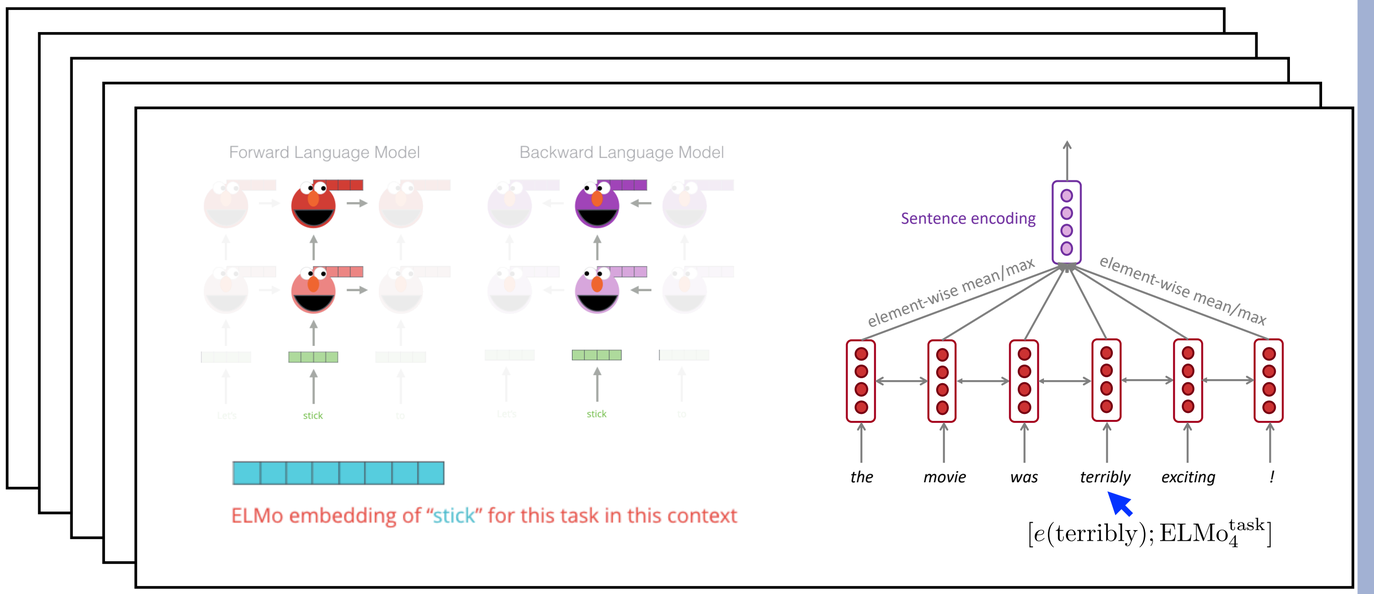
Recap: Contextualized word embeddings (ELMo)

Pre-training (Only once, task-agnostic)



- Data: 10 epochs on 1B Word Benchmark
- Training time: 2 weeks on 3 NVIDIA GTX 1080 GPUs

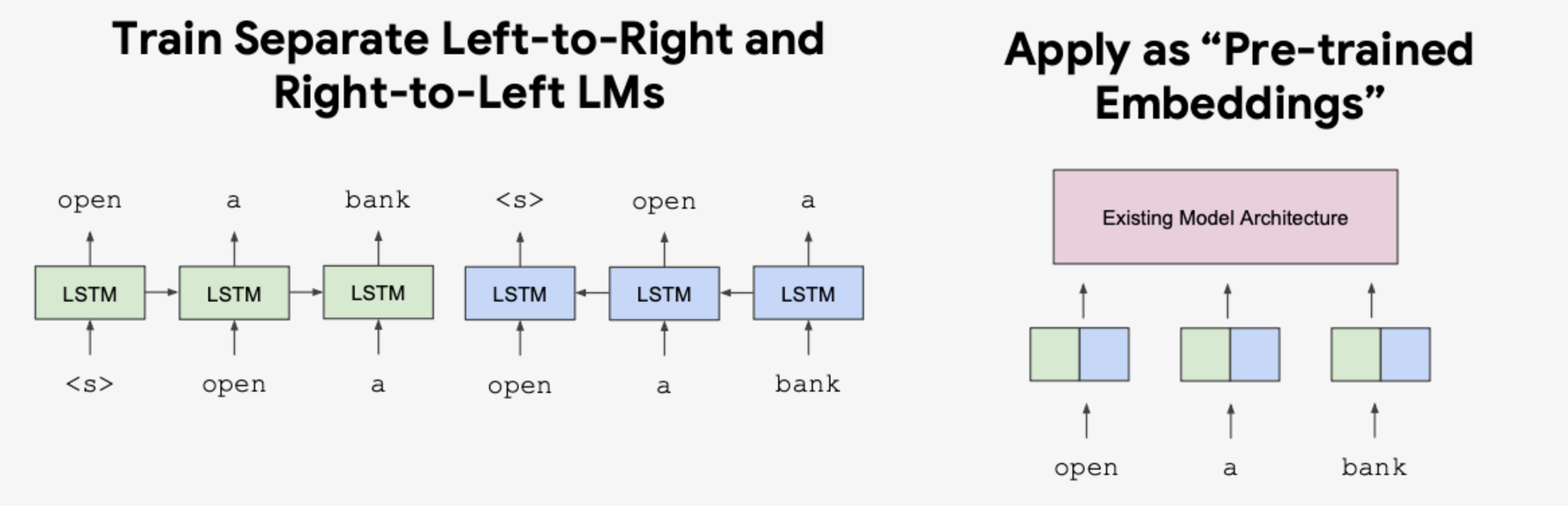
Run inference to get **contextualized embeddings**, **plug them into the main model** (e.g., LSTMs), then train a model for a **specific downstream task**



- Sentiment classification
- Topic classification
- POS tagging
- Machine translation
- Question answering
- …

Recap: Word2vec, ELMo: Feature-based approach

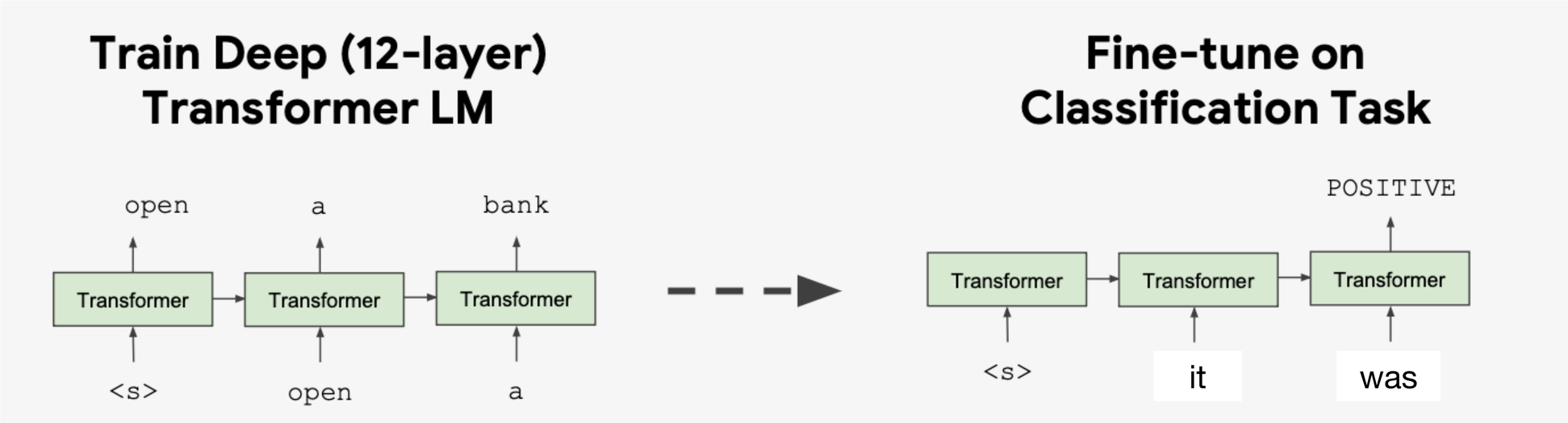
- Feature-based approach: only produces word embeddings that can be used as **input representations** of existing neural models



- Can we do pre-training that allows **re-using model weights**?

Recap: Fine-tuning approaches

- GPT / BERT (2018): **Some of the first pre-trained Transformers with fine-tuning approaches**
  - Almost all model weights will be **re-used**, and only a small number of task-specific will be added for downstream tasks



Pre-training for three types of architectures

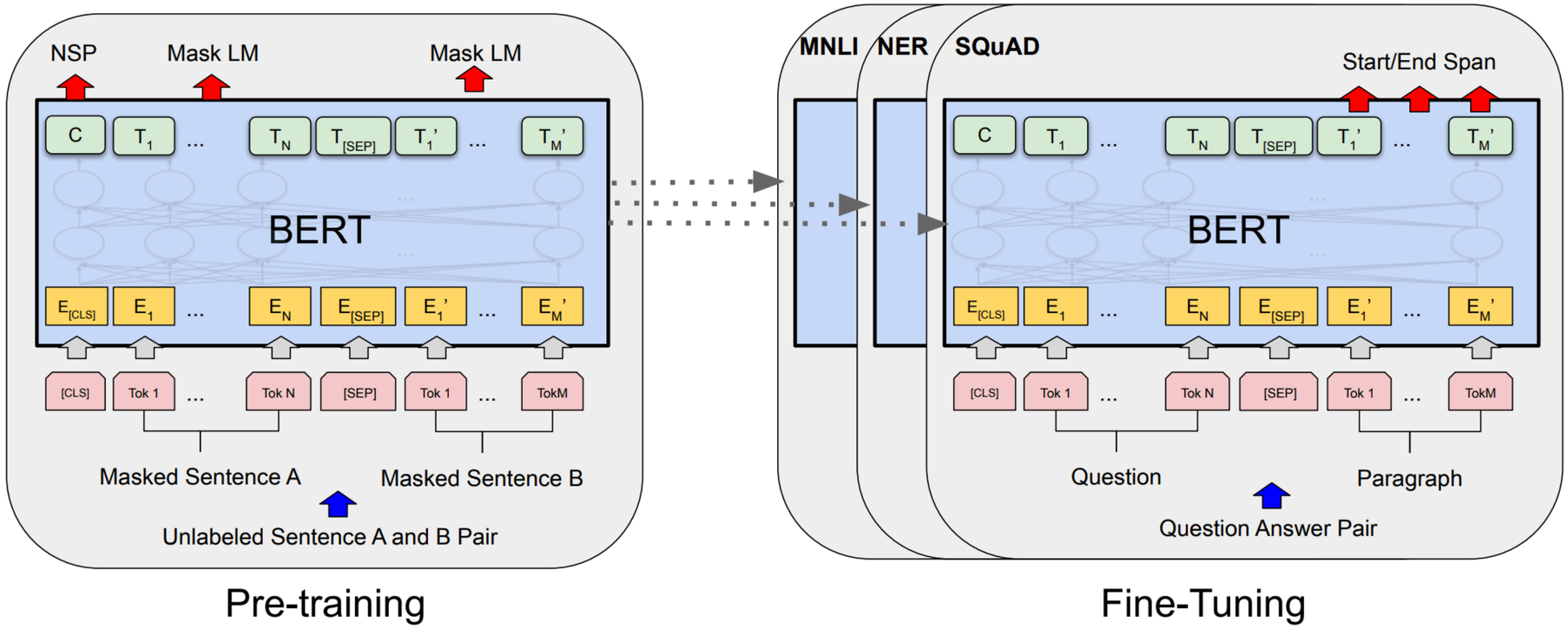
Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	**BERT**, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	**GPT-2**, GPT-3, Llama

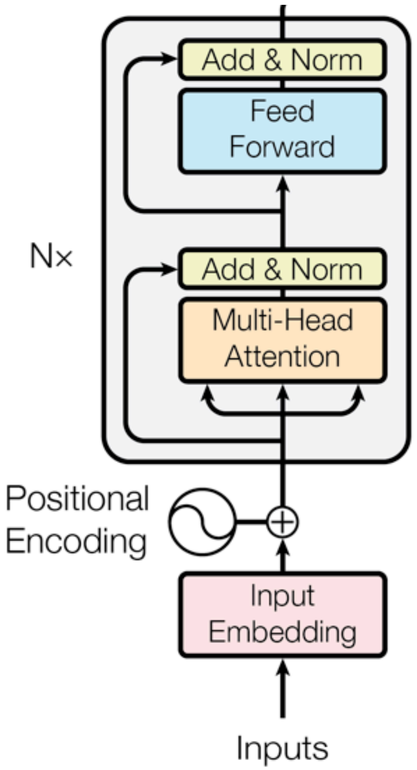
BERT: Bidirectional Encoder Representations from Transformers

BERT: Bidirectional Encoder Representations from Transformers (October 2018)



"**Fine-tuning** is the process of **taking the network learned by these pre-trained models**, and **further training the model**, often via an added neural net classifier that takes the top layer of the network as input, to perform some downstream task."

BERT: Bidirectional Encoder Representations from Transformers (October 2018)



- Based on a **deep bidirectional, encoder-only Transformer**
- It follows a **fine-tuning** approach!
- The key: learn representations based on **bidirectional contexts**

Example #1: we went to the river bank.

Example #2: I need to go to bank to make a deposit.

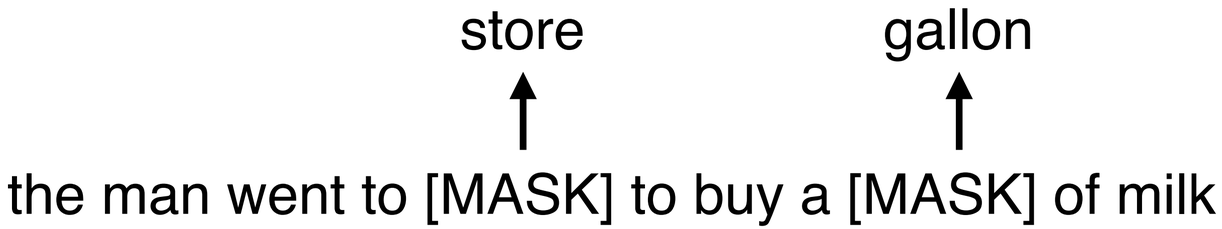
- Two new pre-training objectives:
  - **Masked language modeling (MLM)**
  - Next sentence prediction (NSP) - Later work shows that NSP hurts performance though..

Masked Language Modeling (MLM)

- Q: Why is language modeling limited for bidirectional contexts?

  Example #1: we went to the river bank.
  Example #2: I need to go to bank to make a deposit.

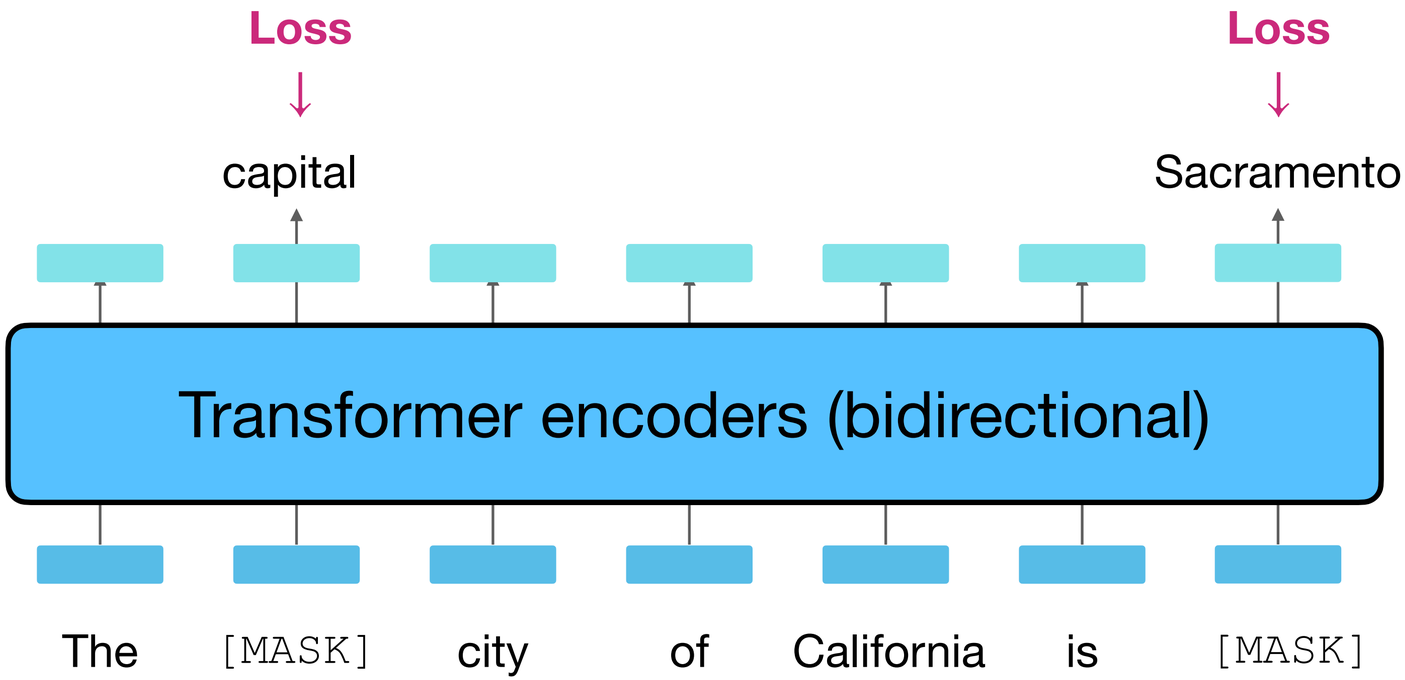
- Solution: Mask out k% of the input words, and then predict the masked words



k = 15% in practice

Masked Language Modeling (MLM)

1. Prepare raw text: “The capital city of California is Sacramento.”
2. Mask 15% of the tokens: "The `[MASK]` city of California is `[MASK]`.”
3. Feed into the Transformers encoder and train with a masked language modeling objective.



Intuition: What do we learn from MLM?

- I went to the ocean and saw fish, turtles, _____ and seals.
  - General semantics
- I put ____ fork down on the table.
  - Syntactic constraints
- The woman walked across the street checking for traffic over _____ shoulder.
  - Co-reference, relations between different entities within the sentence
- Overall, the value I got from the two hours watching it was the sum total of the popcorn and the drink. The movie was _____.
  - Sentiment
- UC Berkeley is located in _____.
  - Named entity recognition, Word knowledge

Slide credit: John Hewitt, Stanford CS224N

MLM: 80-10-10 corruption

For the 15% predicted words,

- 80% of the time, they replace it with [MASK] token

  went to the store ⟶ went to the [MASK]

- 10% of the time, they replace it with a random word in the vocabulary

  went to the store ⟶ went to the running

- 10% of the time, they keep it unchanged

  went to the store ⟶ went to the store

Why? Because [MASK] tokens are never seen during fine-tuning

(See Table 8 of the paper for an ablation study)

Next Sentence Prediction (NSP)

- Motivation: many NLP downstream tasks require understanding the relationship between two sentences (natural language inference, paraphrase detection, QA)
- NSP is designed to reduce the gap between pre-training and fine-tuning

[CLS]: a special token always at the beginning

[SEP]: a special token used to separate two segments

Input = `[CLS] the man went to [MASK] store [SEP]`

`he bought a gallon [MASK] milk [SEP]`

Label = `IsNext`

Input = `[CLS] the man [MASK] to the store [SEP]`

`penguin [MASK] are flight ##less birds [SEP]`

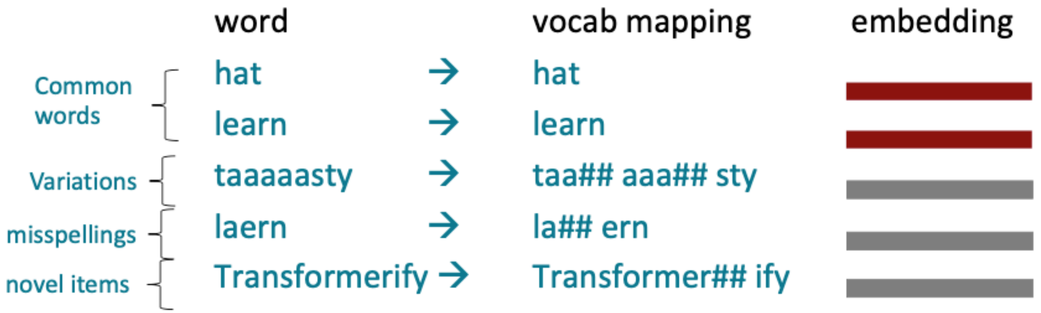
Label = `NotNext`

They sample two contiguous segments for 50% of the time and another random segment from the corpus for 50% of the time

(Later MLM work found it less useful and removed it, e.g., RoBERTa)

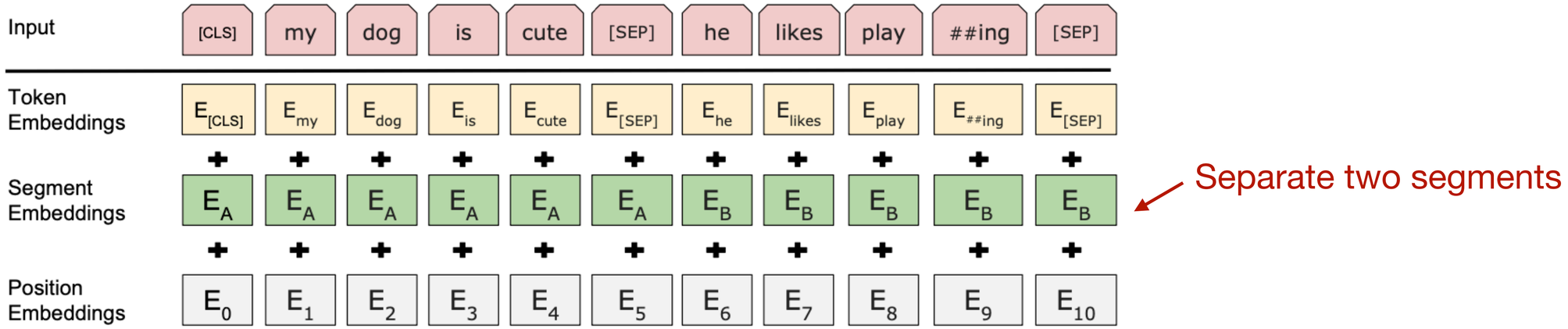
BERT architecture (1/2)

- Vocabulary size: 30,000 **wordpieces** (Wu et al., 2016)

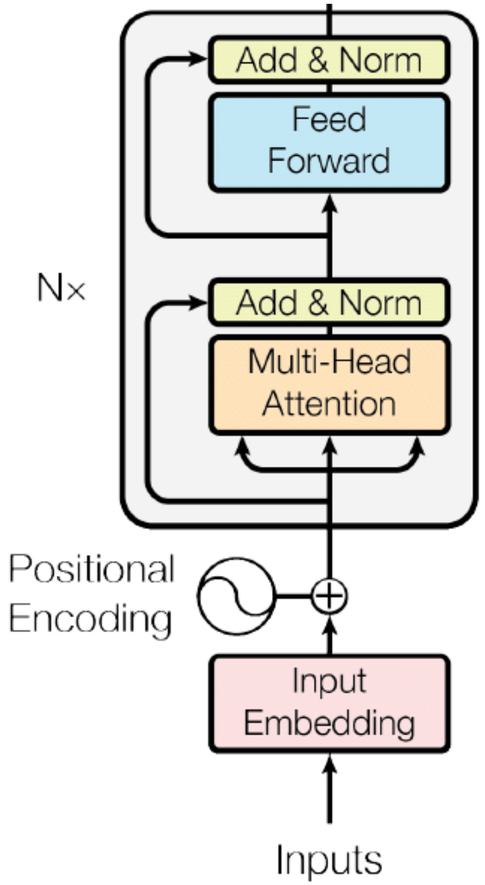


(Image: Stanford CS224N)

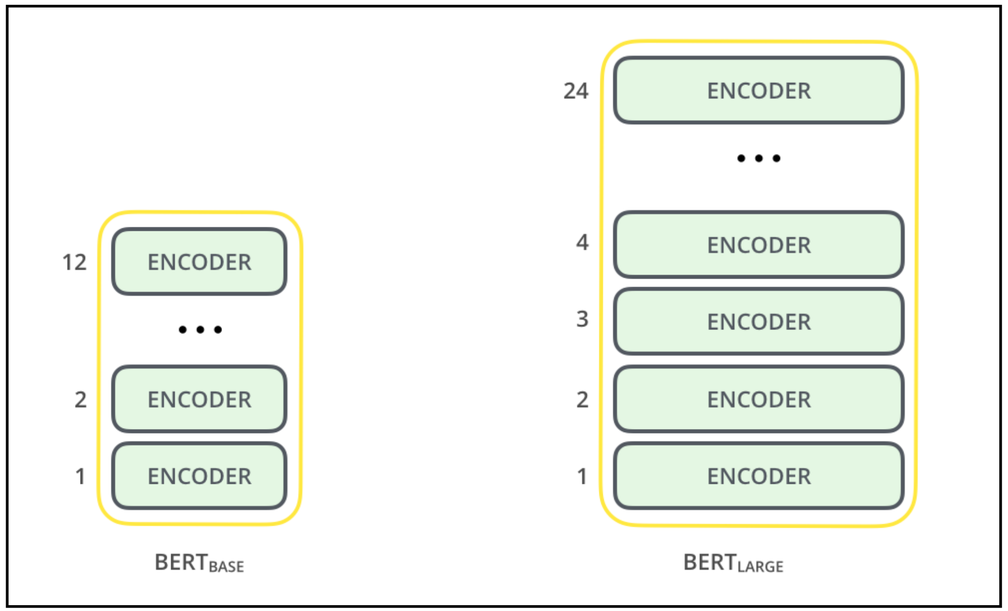
- Input embeddings:



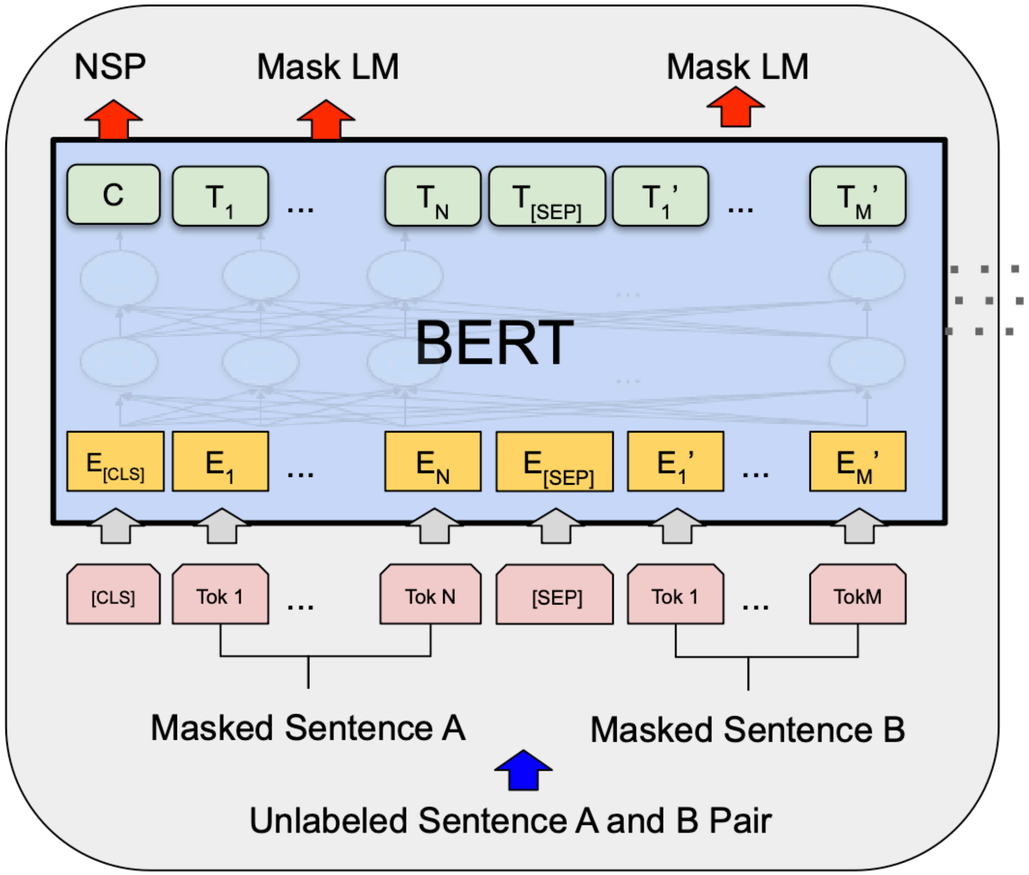
BERT architecture (2/2)



- BERT-base: 12 layers, 768 hidden size, 12 attention heads, **110M parameters**
- BERT-large: 24 layers, 1024 hidden size, 16 attention heads, **340M parameters**



BERT pre-training



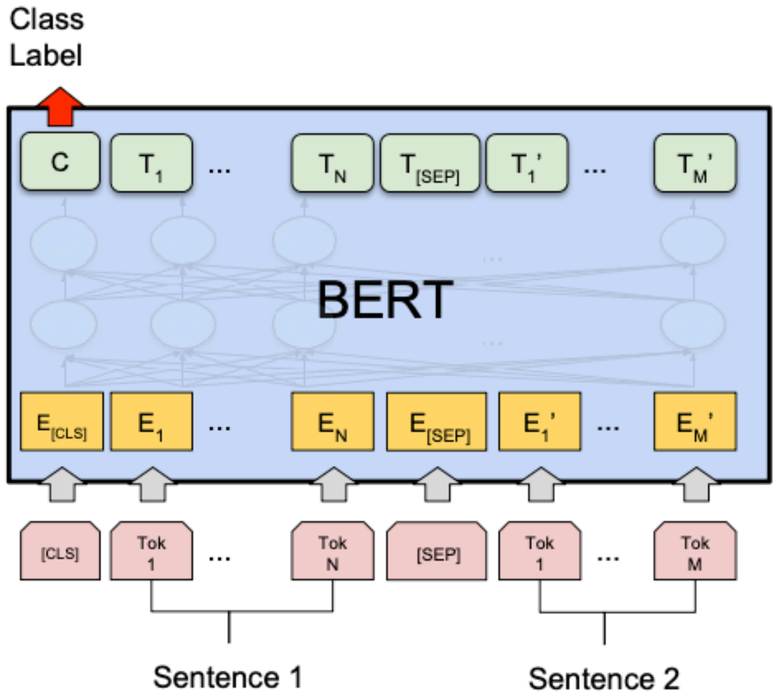
Pre-training

- Training corpus: Wikipedia (2.5B) + BooksCorpus (0.8B)
- Max sequence size: 512 wordpiece tokens (roughly 256 and 256 for two non-contiguous sequences)
- Trained for 1M steps, batch size 128k
- MLM and NSP are trained together
  - [CLS] is pre-trained for NSP
  - Other token representations are trained for MLM

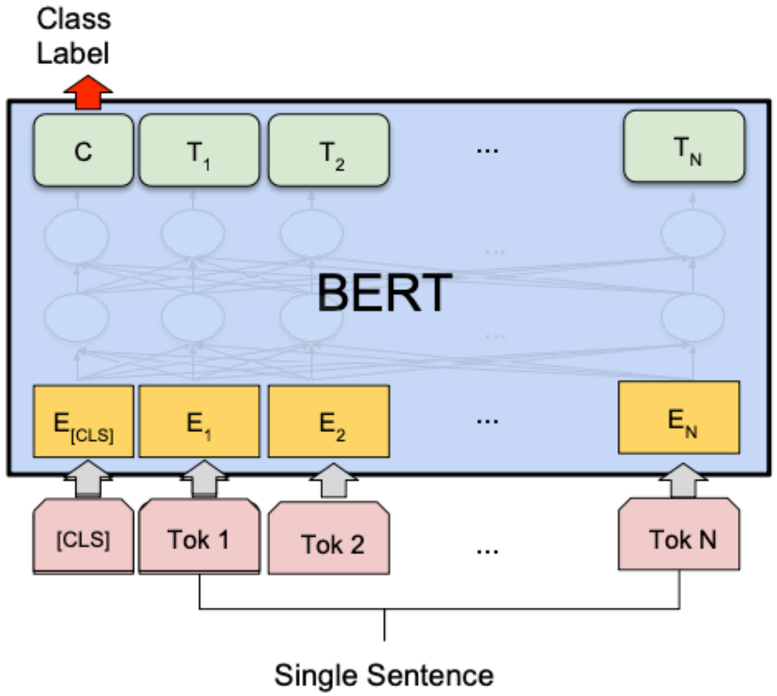
BERT fine-tuning

"Pre-train once, finetune many times."

Type 1: Classification



(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

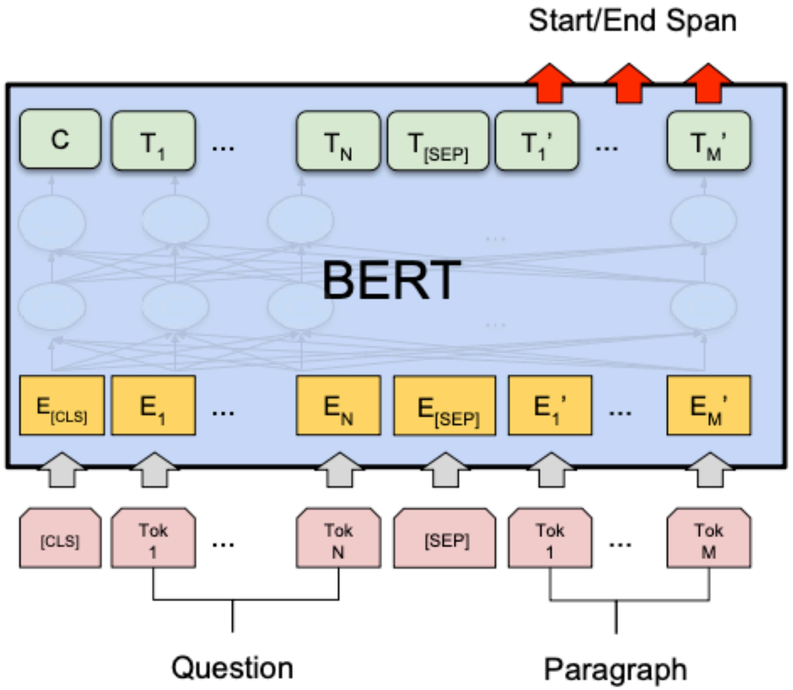


(b) Single Sentence Classification Tasks: SST-2, CoLA

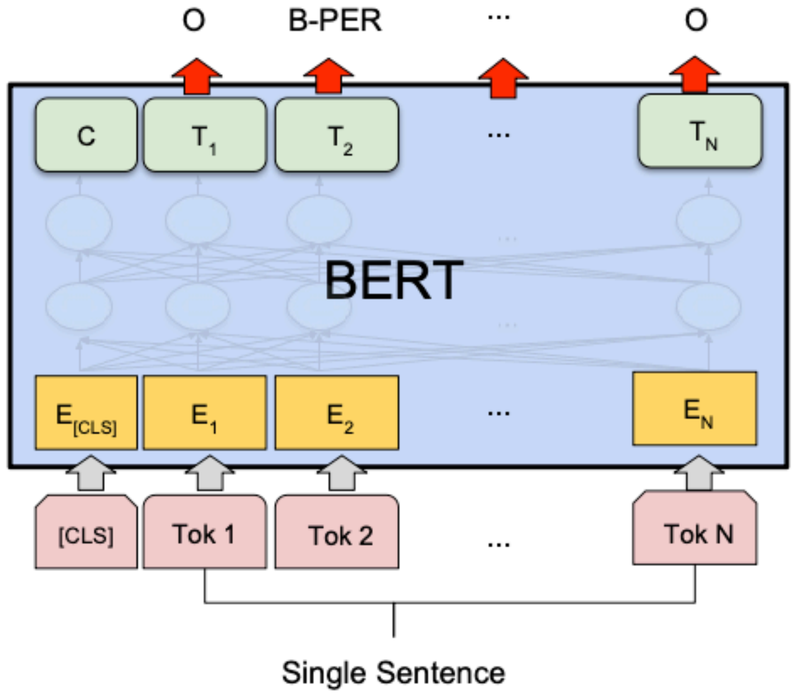
BERT fine-tuning

"Pre-train once, finetune many times."

Type 2: Sequence tagging



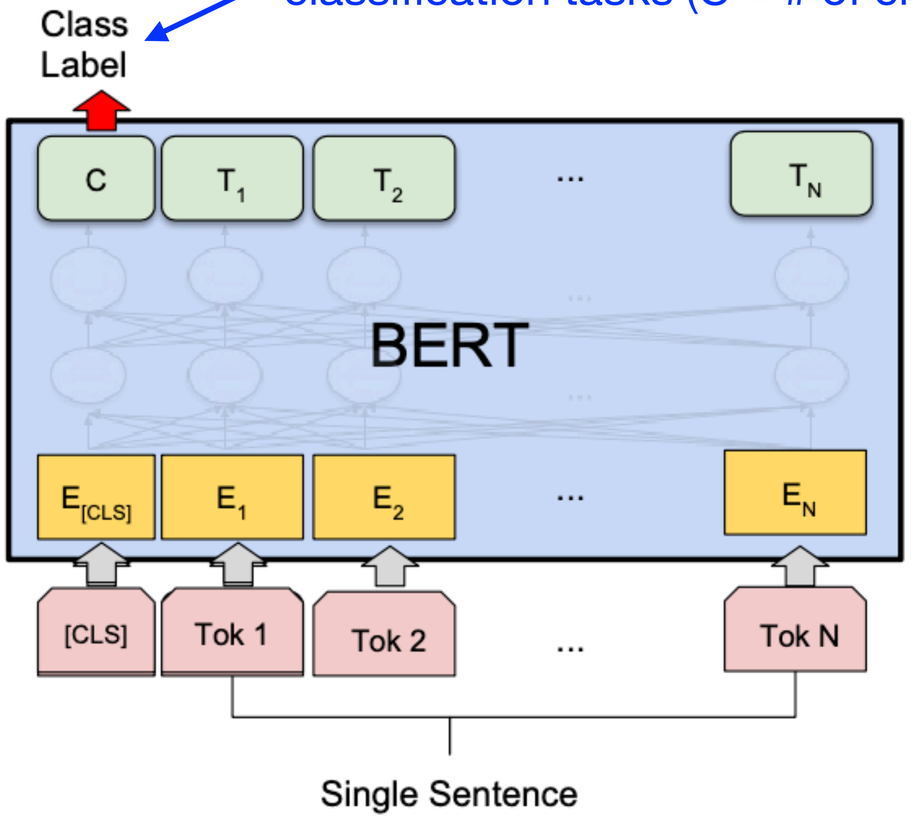
(c) Question Answering Tasks: SQuAD v1.1



(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Example: Sentiment classification

We just need to introduce $C \times h$ parameters for classification tasks (C = # of classes, h = hidden size)!



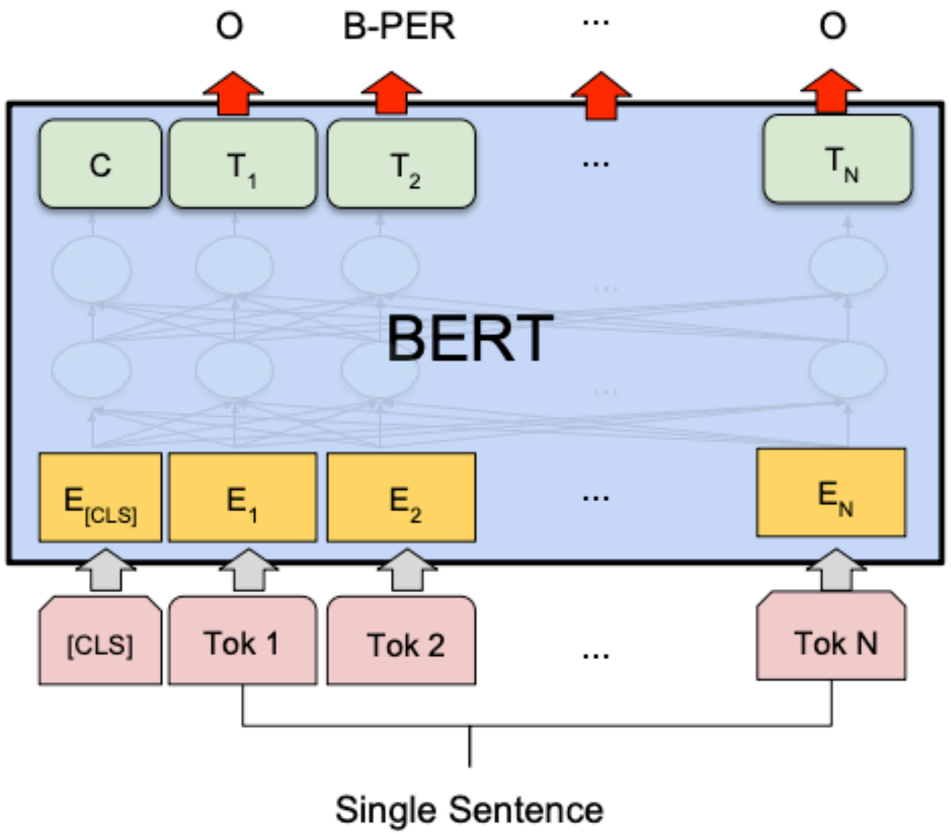
$$P(y = k) = softmax_k(\mathbf{W}_o \mathbf{h}_{[CLS]})$$

$$\mathbf{W}_o \in \mathbb{R}^{C \times h}$$

All the parameters will be learned together (original BERT parameters + new classifier parameters)

Example: Named entity recognition (NER)

We just need to introduce $C \times h$ parameters for classification tasks (C = # of classes, h = hidden size)!



$$P(y_i = k) = softmax_k(\mathbf{W}_o \mathbf{h}_i)$$

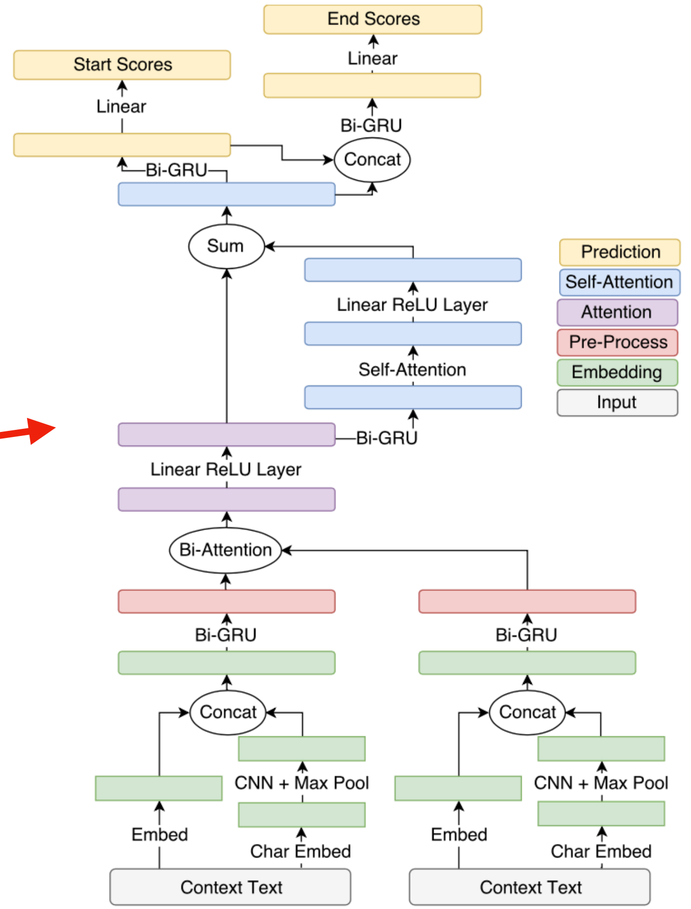
$$\mathbf{W}_o \in \mathbb{R}^{C \times h}$$

Experimental results: GLUE

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

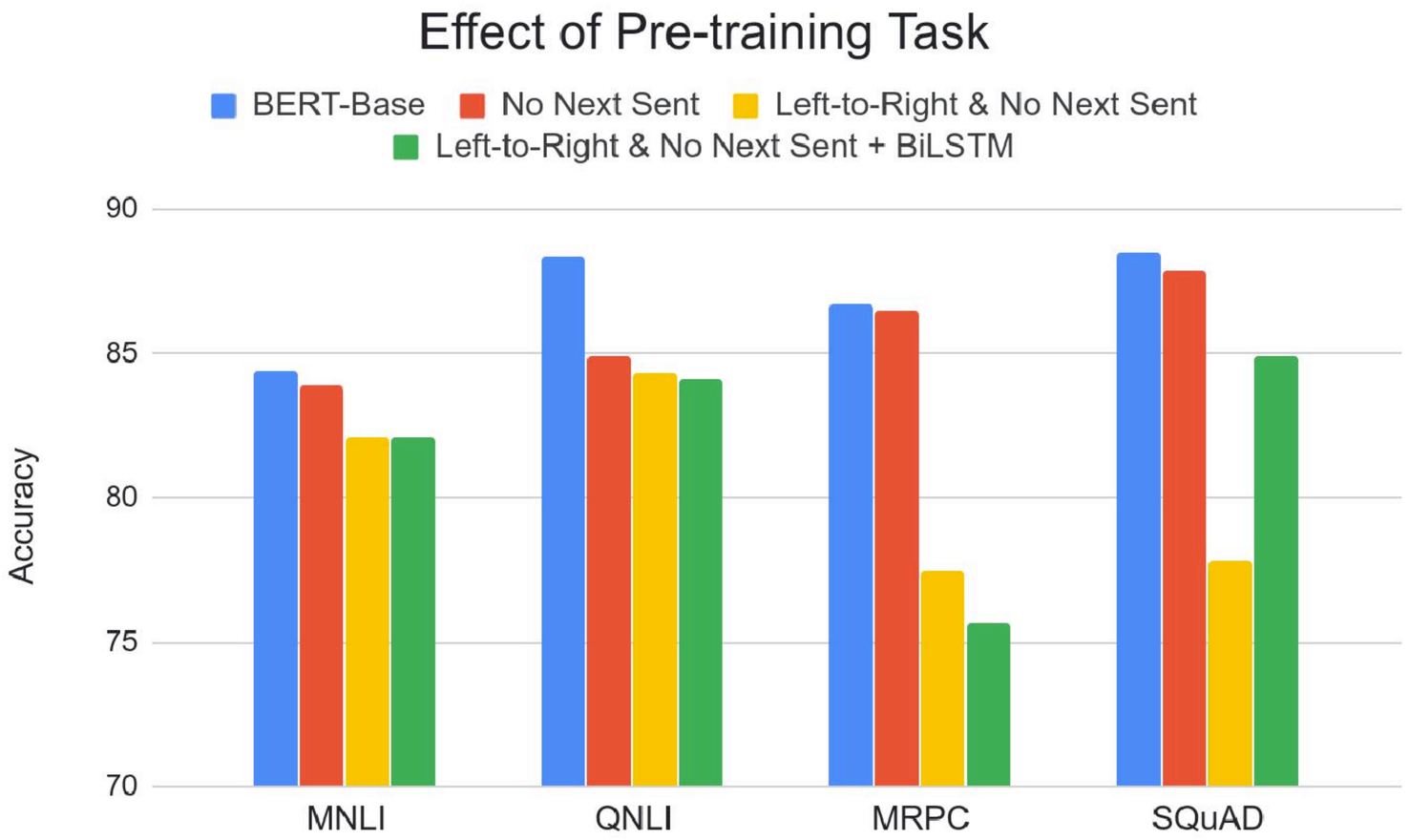
Experimental results: SQuAD

System	Dev EM	Dev F1	Test EM	Test F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**



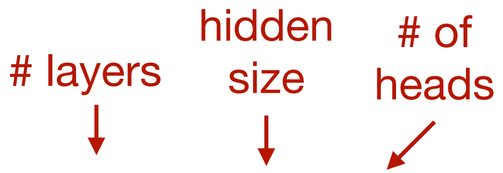
SQuAD = Stanford Question Answering dataset

Ablation study: Pre-training tasks



- MLM >> left-to-right LMs
- NSP improves on some tasks
- Note: later work (Joshi et al., 2020; Liu et al., 2019) argued that NSP is not useful

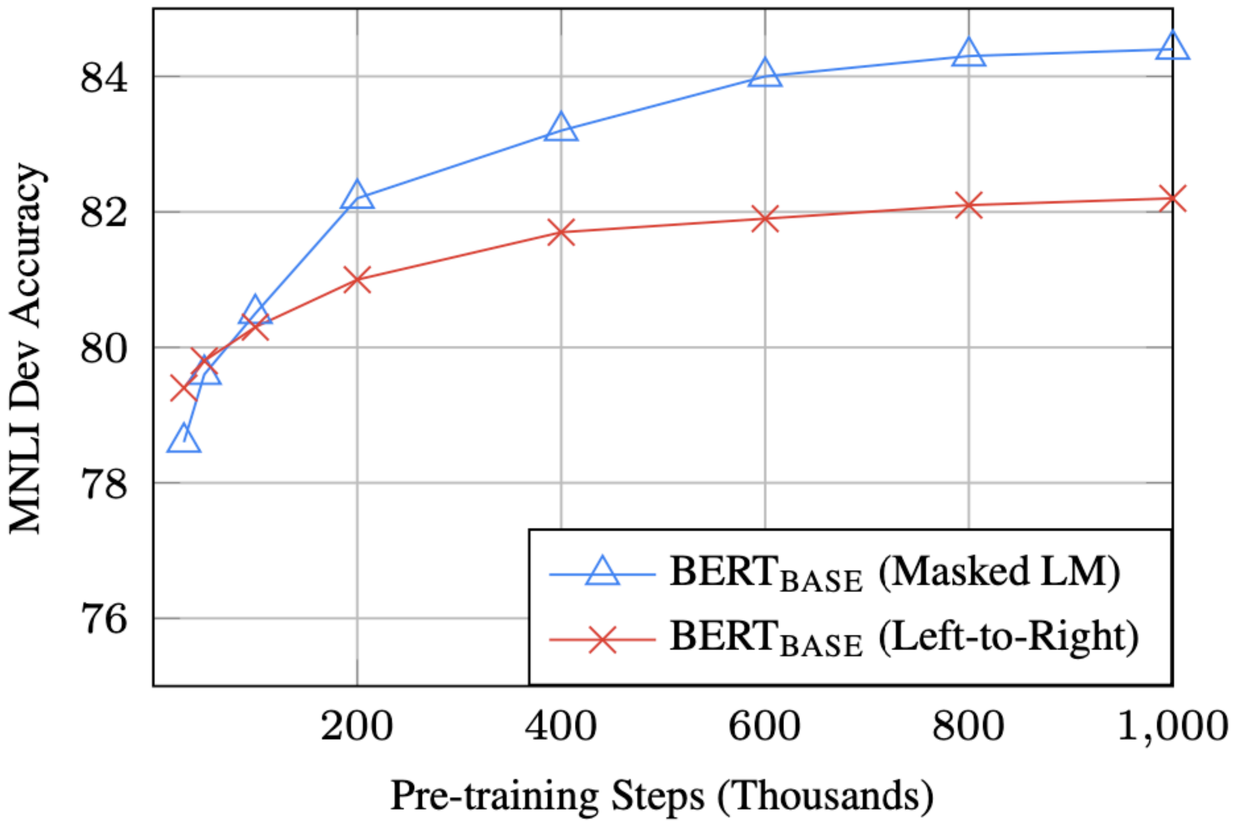
Ablation study: Model sizes



Hyperparams				Dev Set Accuracy		
#L	#H	#A	LM (ppl)	MNLI-m	MRPC	SST-2
3	768	12	5.84	77.9	79.8	88.4
6	768	3	5.24	80.6	82.2	90.7
6	768	12	4.68	81.9	84.8	91.3
12	768	12	3.99	84.4	86.7	92.9
12	1024	16	3.54	85.7	86.9	93.3
24	1024	16	3.23	86.6	87.8	93.7

The bigger, the better!

Ablation study: Training efficiency



MLM takes slightly longer to converge because it only predicts 15% of tokens

RoBERTa (Liu et al. 2019)

- BERT is still under-trained!
- Removed the next sentence prediction training - it adds more noise than benefits!
- Trained longer with 10x data & bigger batch sizes
- Pre-trained on 1,024 V100 GPUs for one day in 2019

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

(Liu et al., 2019): RoBERTa: A Robustly Optimized BERT Pretraining Approach

Limitation of pre-trained encoders

- If your task involves generating sequences, BERT and other pre-trained encoders don't naturally lead to nice autoregressive (1-word-at-a-time) generation methods.
- Might want to use a pre-trained decoder!

Slide credit: John Hewitt, Stanford CS224N

Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

Quick quiz

Which paper first used the term “GPT”?

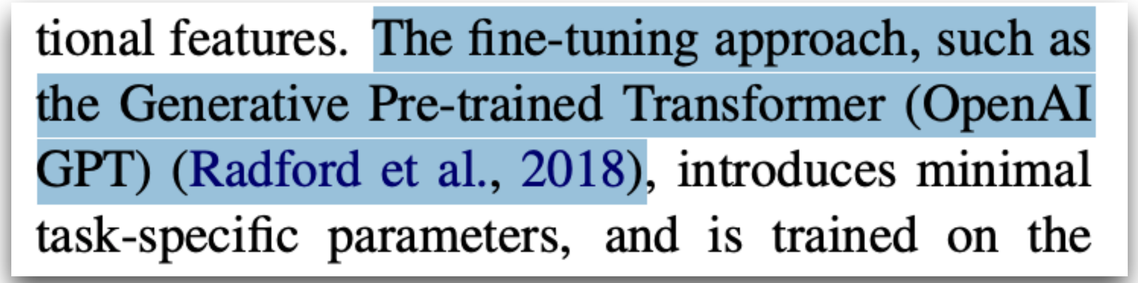
(A) Radford et al. 2018. [GPT-1 paper, OpenAI]

(B) Radford et al., 2019. [GPT-2 paper, OpenAI]

(C) Brown et al., 2020. [GPT-3 paper, OpenAI]

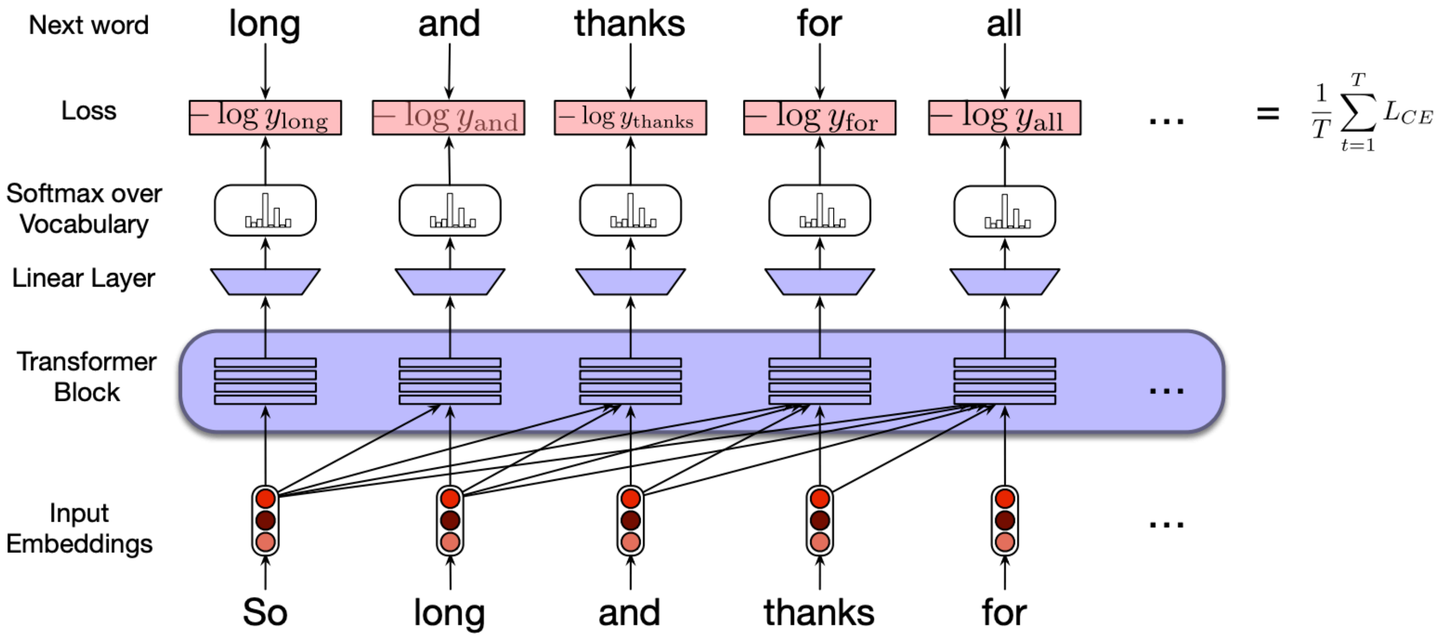
(D) Devlin et al. 2019. [BERT paper, Google]

(D) is correct.


tional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the

Generative Pre-Training (GPT) (June 2018)

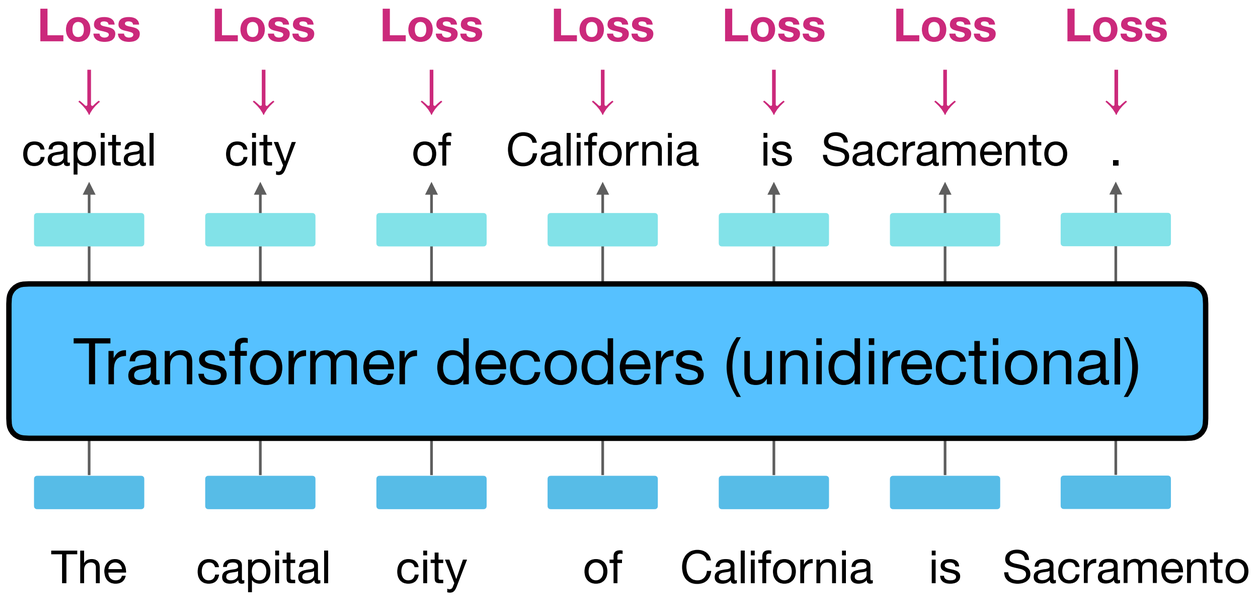
- Use a **Transformer decoder** (unidirectional; left-to-right) instead of LSTMs
- Use **language modeling** as a pre-training objective
  - Also called **generative** or **causal language modeling**
- Trained on longer segments of text (**512 BPE tokens**), not just single sentences



(Radford et al, 2018): Improving Language Understanding by Generative Pre-Training

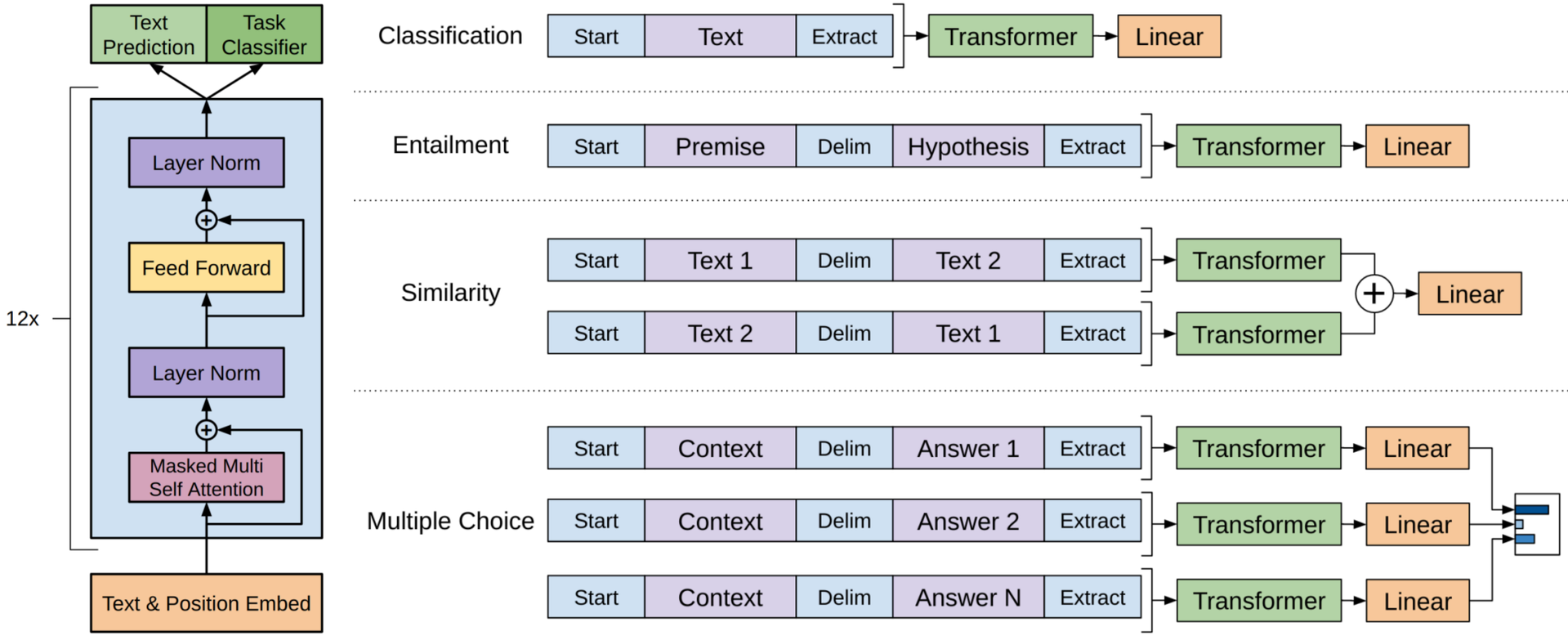
Generative Pre-Training (GPT) (June 2018)

1. Prepare raw text: “The capital city of California is Sacramento.”
2. Feed into the Transformers decoder and train with a (generative) language modeling objective.



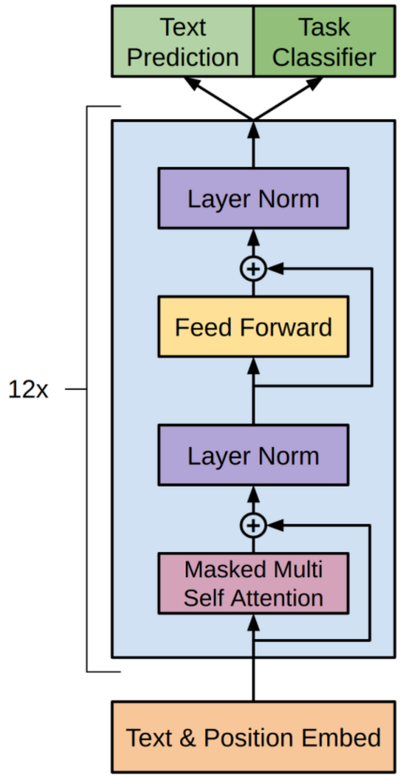
Generative Pre-Training (GPT) (June 2018)

- "Fine-tune" the entire set of model parameters on various downstream tasks



(Radford et al, 2018): Improving Language Understanding by Generative Pre-Training

GPT: More details



- 12 layers, 768 hidden size, 12 attention heads, 110M parameters

Same as BERT-base

Recall: BERT was trained on this + Wikipedia!

- Training corpus: BooksCorpus (0.8B)
- Max sequence size: 512 wordpiece tokens
- Trained for 100 epochs, batch size 64

(Radford et al, 2018): Improving Language Understanding by Generative Pre-Training

Experimental results: GLUE

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
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

ELMo vs. GPT vs. BERT

Which of the following statements is INCORRECT?

(A) BERT was trained on more data than ELMo

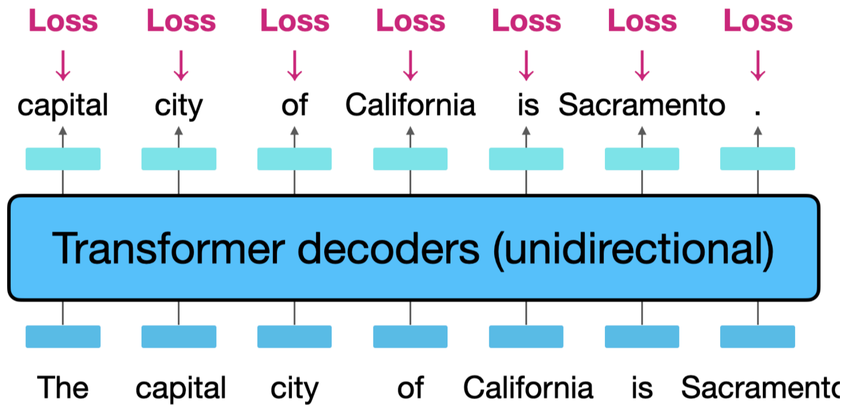
(B) BERT builds on Transformer encoder, and GPT builds on Transformer decoder

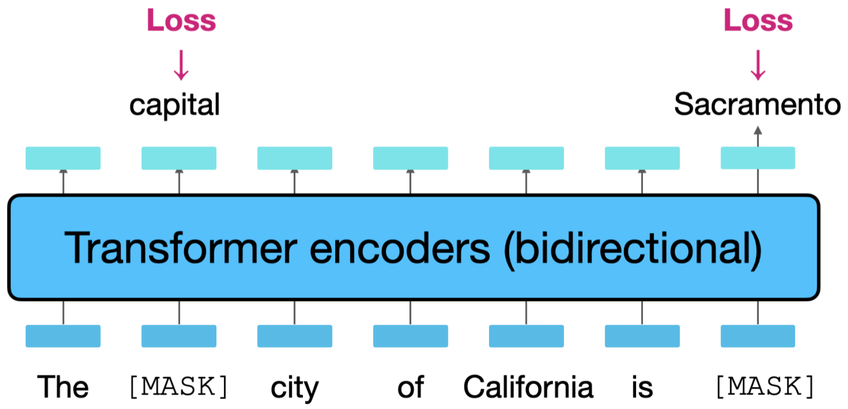
(C) ELMo requires different model architectures for different tasks

(D) BERT was trained on data with longer contexts compared to GPT

(D) is incorrect.

Causal (Generative) LMs vs. Masked LMs





- **Causal LMs** receive more training signals per forward pass than **masked LMs**.
- **Causal LMs** can be used for text generation, whereas **masked LMs** cannot.
- **Causal LMs** learn from noisier signals, since each token is predicted using only preceding context. →Therefore, with fine-tuning, masked LMs typically had better performance.

Pre-training for three types of architectures

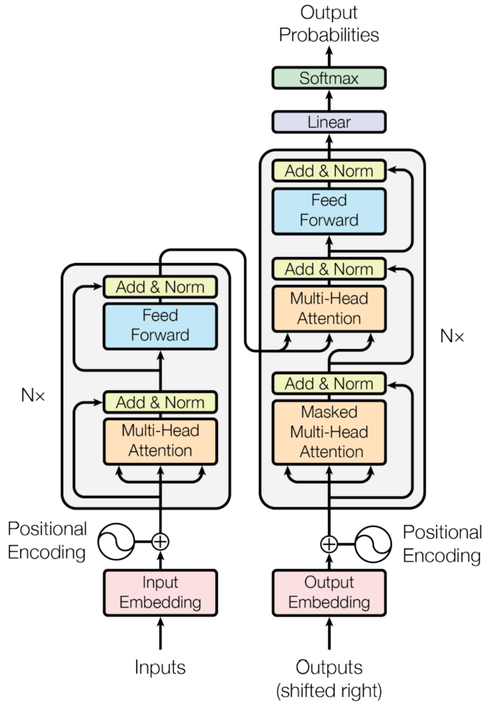
Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

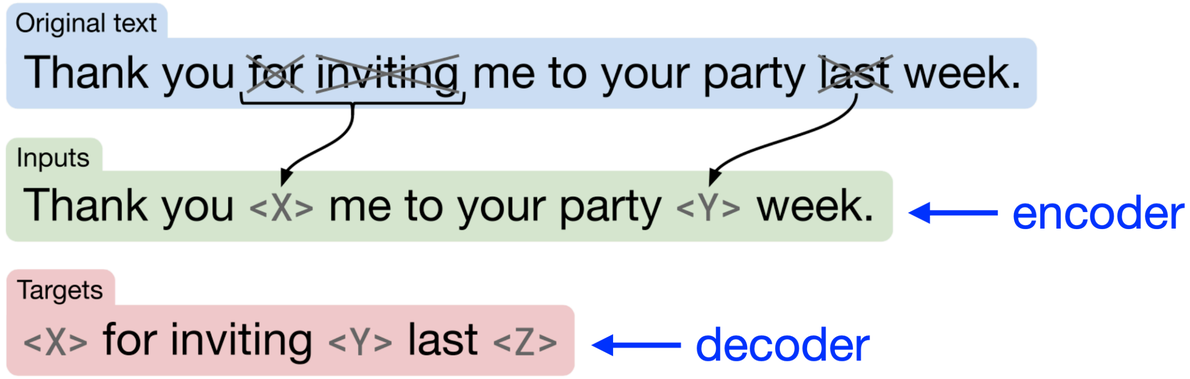
Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

T5

- Encoder-only models (e.g., BERT) enjoy the benefits of bidirectionally but can't generate text.
- Decoder-only models (e.g., GPT) can do generation but don't benefit from bidirectionally for reading inputs.
- Encoder-decoder combines the best of both worlds!





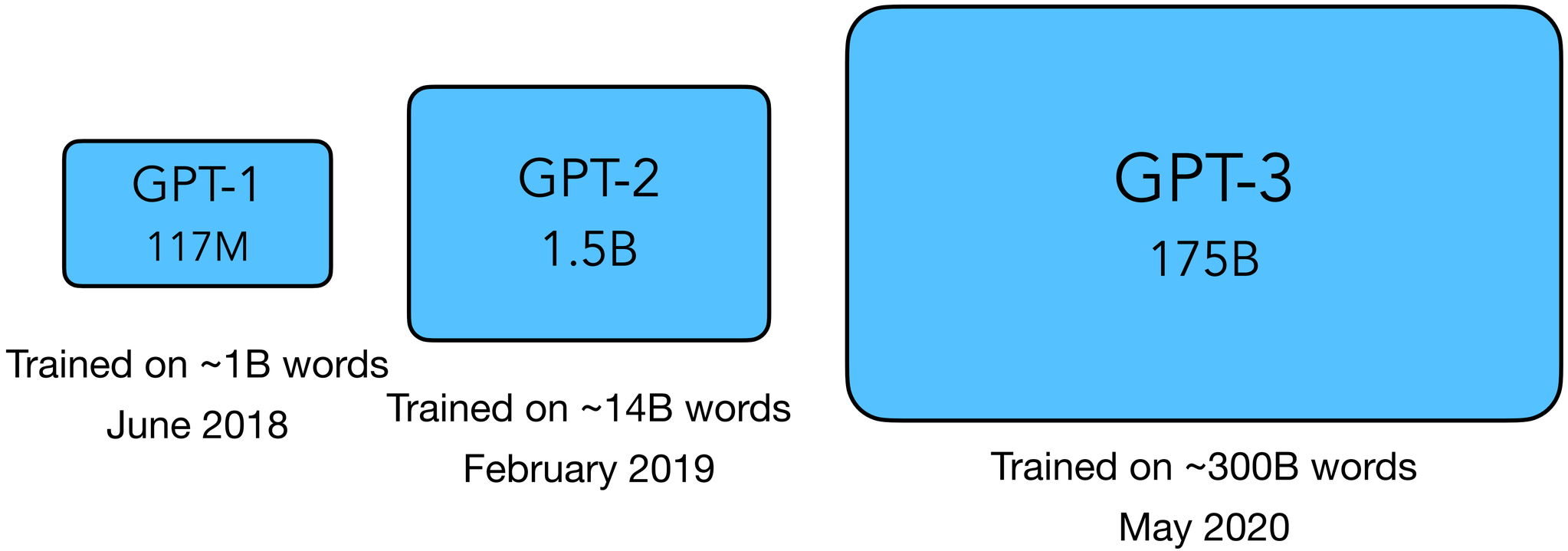
(Raffel et al., 2020): Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer

Pre-training for three types of architectures

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

From GPT to GPT-2 to GPT-3

From GPT to GPT-2 to GPT-3



- All **decoder-only** Transformer-based language models
- **More** parameters, **larger** training corpora

From fine-tuning to prompting

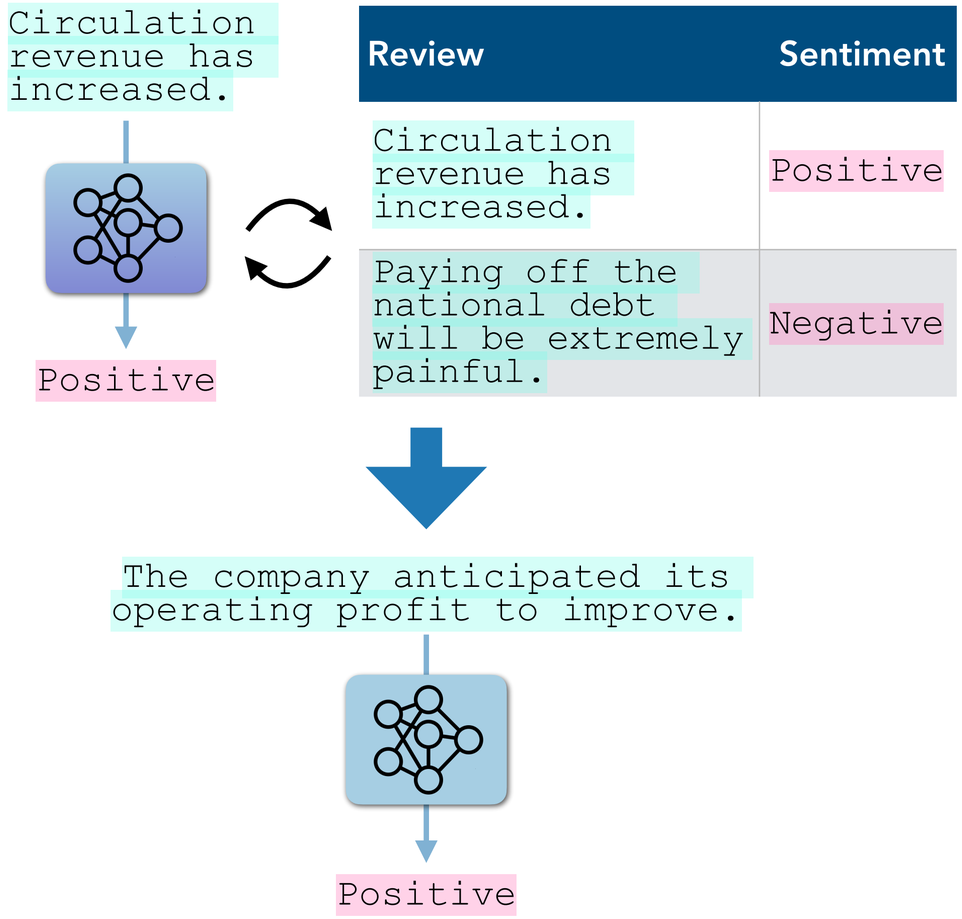
- Before GPT-2/3, fine-tuning was the default (e.g., BERT/T5/GPT-2)
  - Fine-tuning separately for each task
  - SST-2 has 67k examples, SQuAD has 88k (passage, answer, question) triples

- Fine-tuning is still extra work!
  - Requires sufficient training data
  - Requires computing the gradient and applying a parameter update — expensive for large models

- GPT-2 (February 2019) showed promise of zero-shot learning
  - Not using any training data (so, actually not “learning”)
  - Limited success; fine-tuning was still dominant

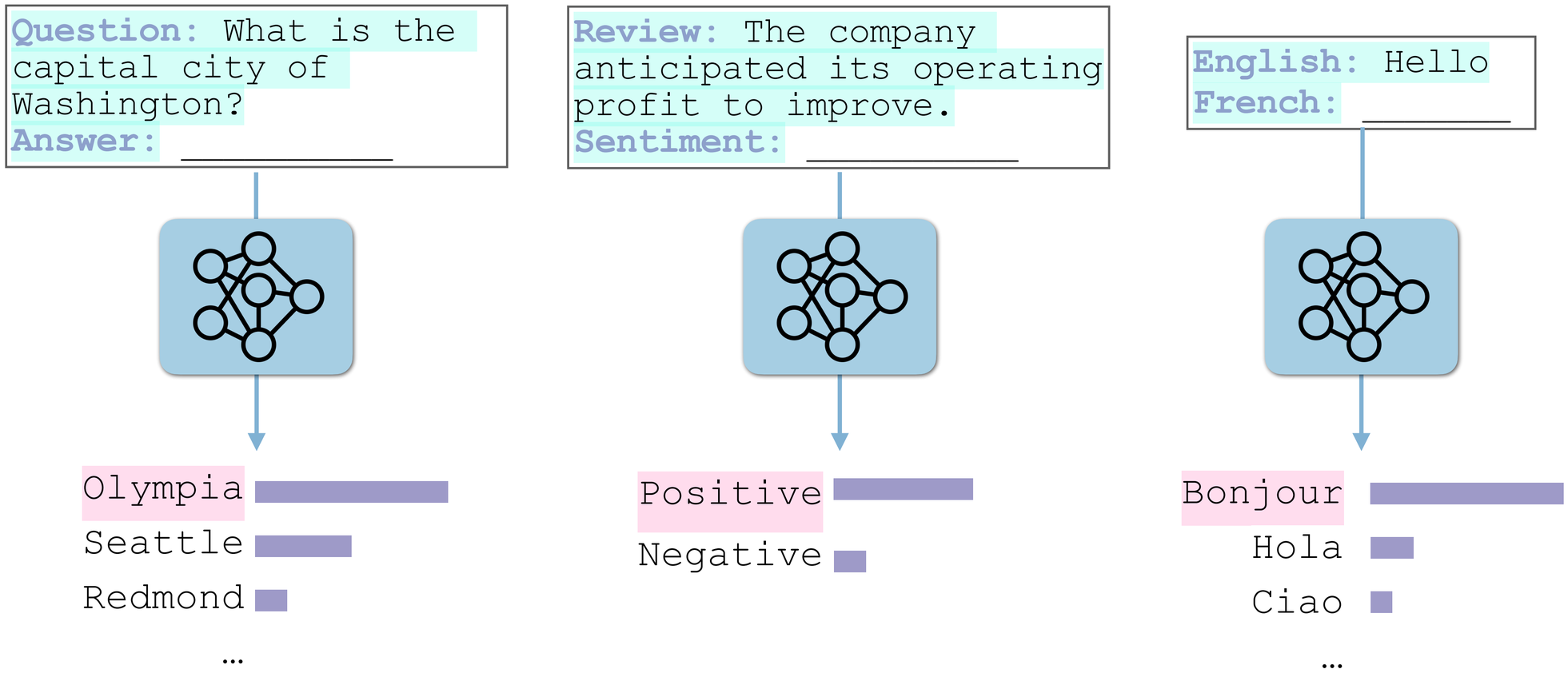
- GPT-3 (May 2020) showed promise of few-shot learning
  - Using the k-shot training data but without actual gradient updates (so still, not actually “learning”)
  - Started to be taken seriously

Recap: Fine-tuning



GPT-2: language models are *zero-shot* learners

Notation: `Given text` | `Generation`



(Radford et al., 2019): Language Models are Unsupervised Multitask Learners

GPT-2: language models are *zero-shot* learners

Notation: `Given text` | `Generation`

```
Answer the following question.
Q: What is the capital city of California?
A: Sacramento
```

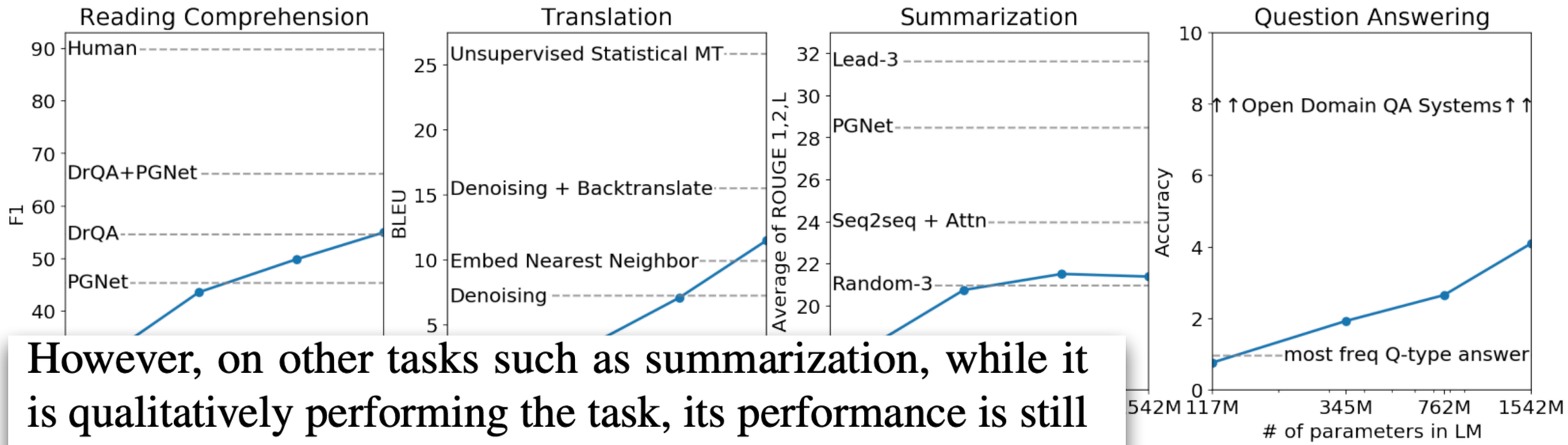
```
Translate this text.
English: The weather is nice today.
Korean: 오늘 날씨가 좋네요.
```

```
Write a short story about a girl living in California.

Title: The Orange Tree

In a quiet neighborhood on the edge of Sacramento, twelve-year-old Maya lived in a small yellow house with her grandmother. The backyard was wild and beautiful, with tall sunflowers, patches of mint, and an old orange tree that her grandfather had planted decades ago. (…)
```

```
How do I get to Sacramento City Hall from UC Berkeley by car?

Instructions: Here are driving directions (approximately 1hr 30min, ~85 miles)
1. Get on I-80 E from Berkeley.
2. Follow I-80 E toward Sacramento.
3. Take exit 519B toward J Street.
4. Turn left on 9th Street.
5. Destination: 915 I Street, Sacramento, CA 95814
```

A.K.A. “Prompting”

GPT-2: language models are *zero-shot* learners



However, on other tasks such as summarization, while it is qualitatively performing the task, its performance is still only rudimentary according to quantitative metrics. While suggestive as a research result, in terms of practical applications, the zero-shot performance of GPT-2 is still far from use-able.

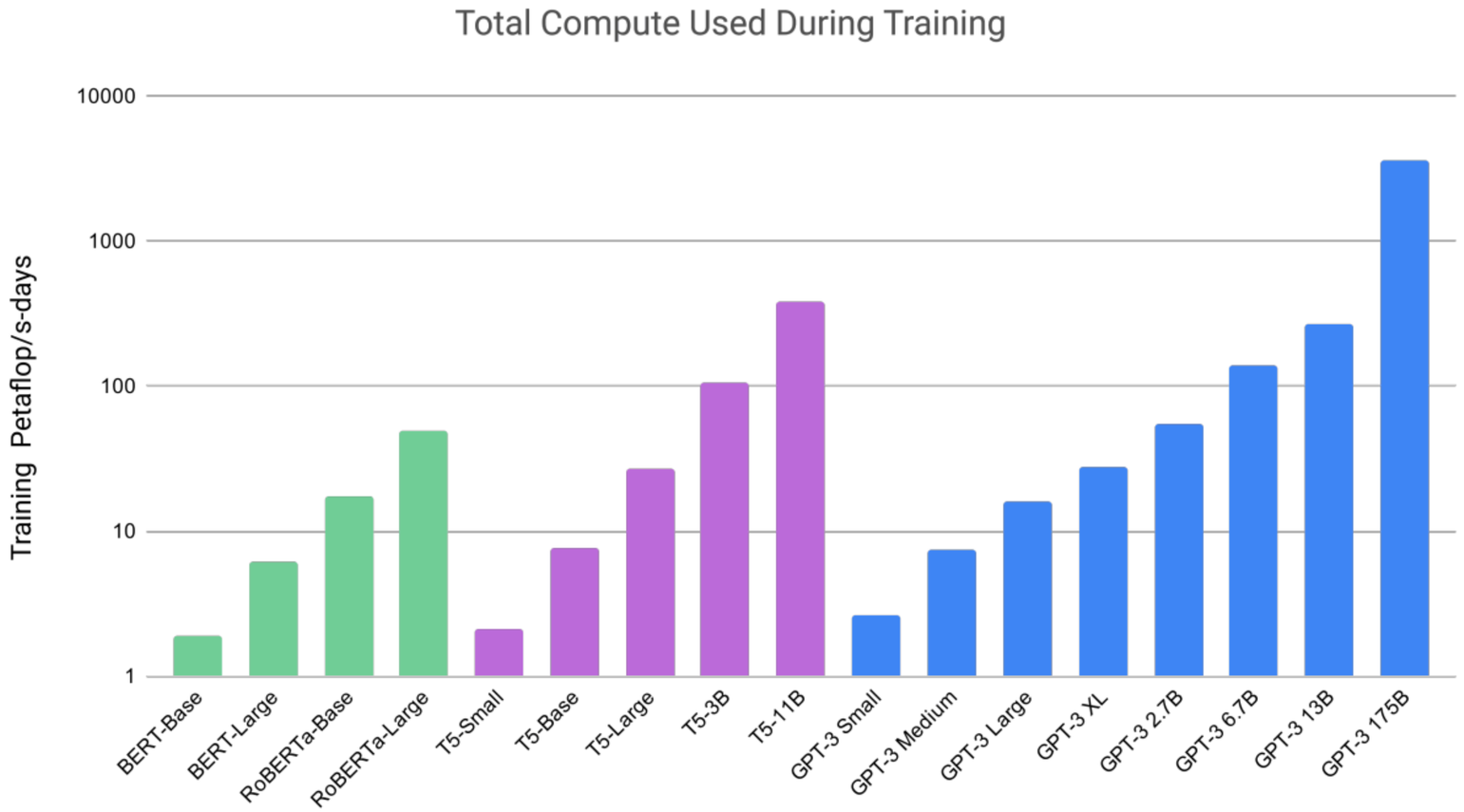
asks. Reading Comprehension results
on on CNN and Daily Mail (See et al.,
ns detailed descriptions of each result.

(Although, not to all…)

(Radford et al., 2019): Language Models are Unsupervised Multitask Learners

GPT-3: language models are *few-shot* learners

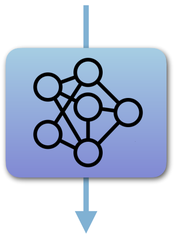
GPT-2 → GPT-3: 1.5B → 175B (# of parameters), ~14B → 300B (# of tokens)



(Brown et al., 2020): Language Models are Few-Shot Learners

Finetuning vs. In-context learning

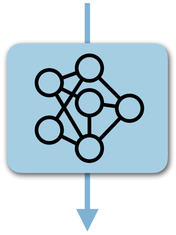
Circulation revenue has increased.



Positive

Review	Sentiment
Circulation revenue has increased.	Positive
Paying off the national debt will be extremely painful.	Negative

The company anticipated its operating profit to improve.

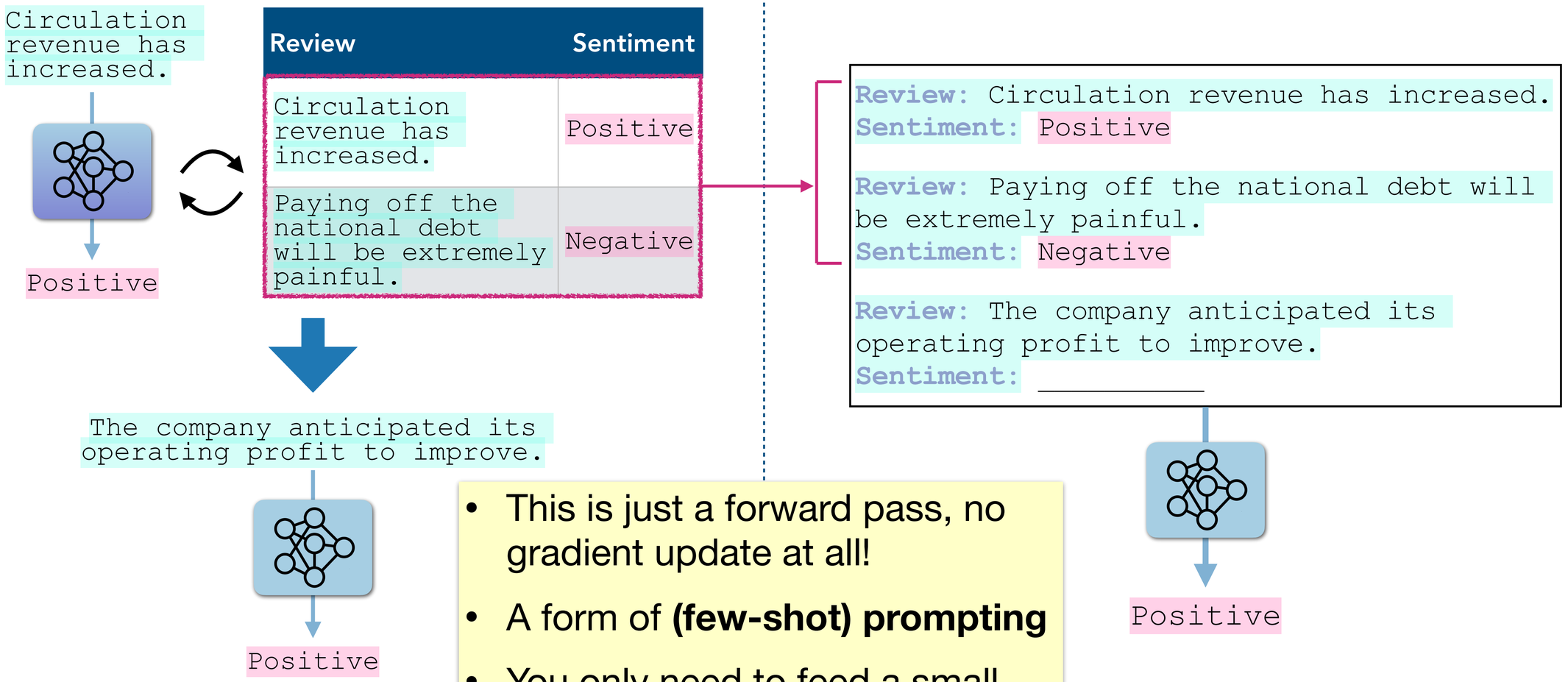


Positive

Review: Circulation revenue has increased.
Sentiment: Positive

Review: Paying off the national debt will be extremely painful.
Sentiment: Negative

Finetuning vs. In-context learning



- This is just a forward pass, no gradient update at all!
- A form of **(few-shot) prompting**
- You only need to feed a small number of examples (e.g., 32)

GPT-3's in-context learning

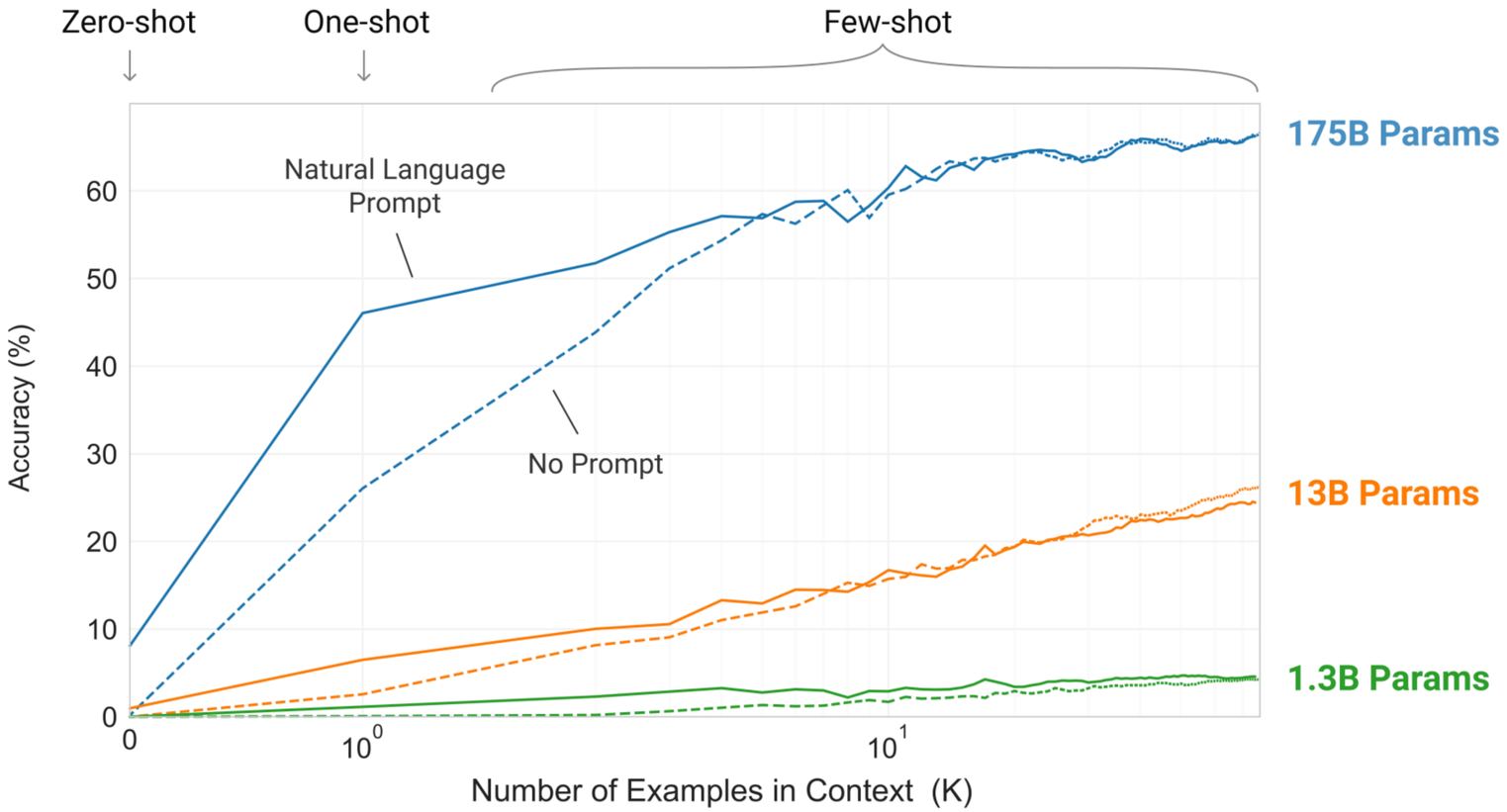
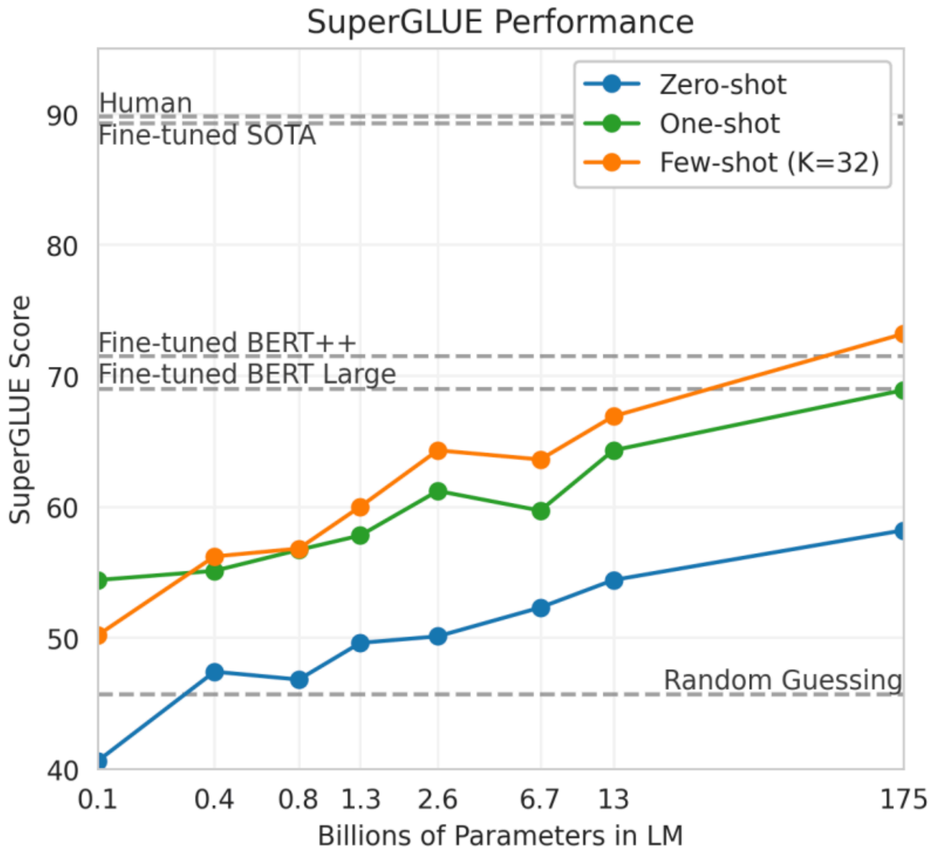
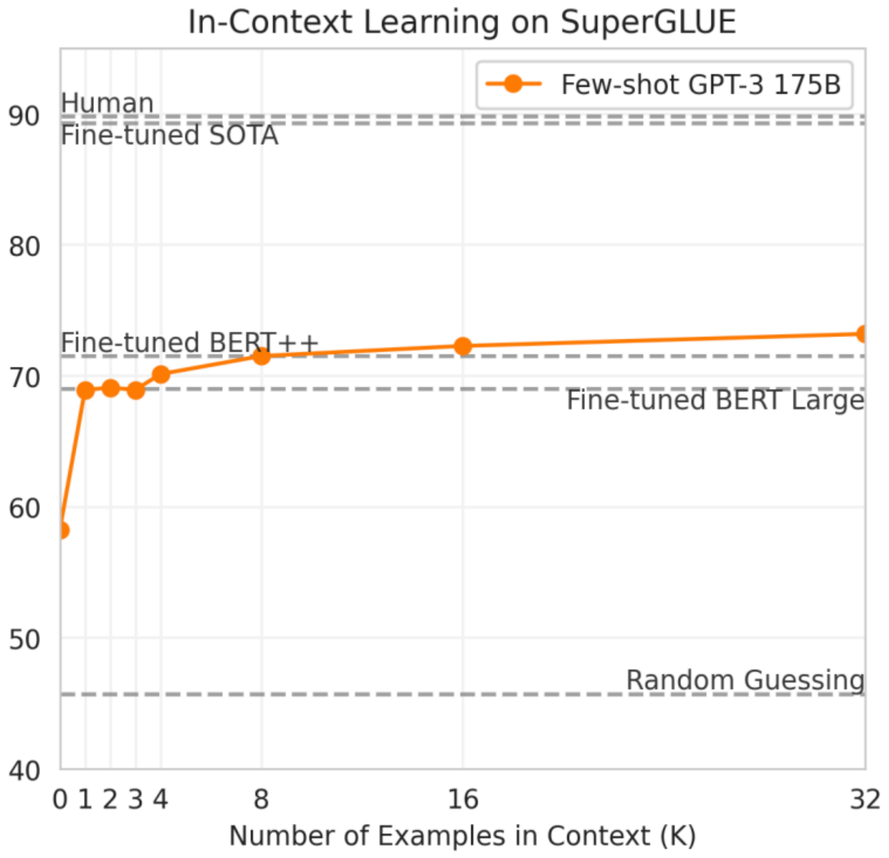


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

(Brown et al., 2020): Language Models are Few-Shot Learners

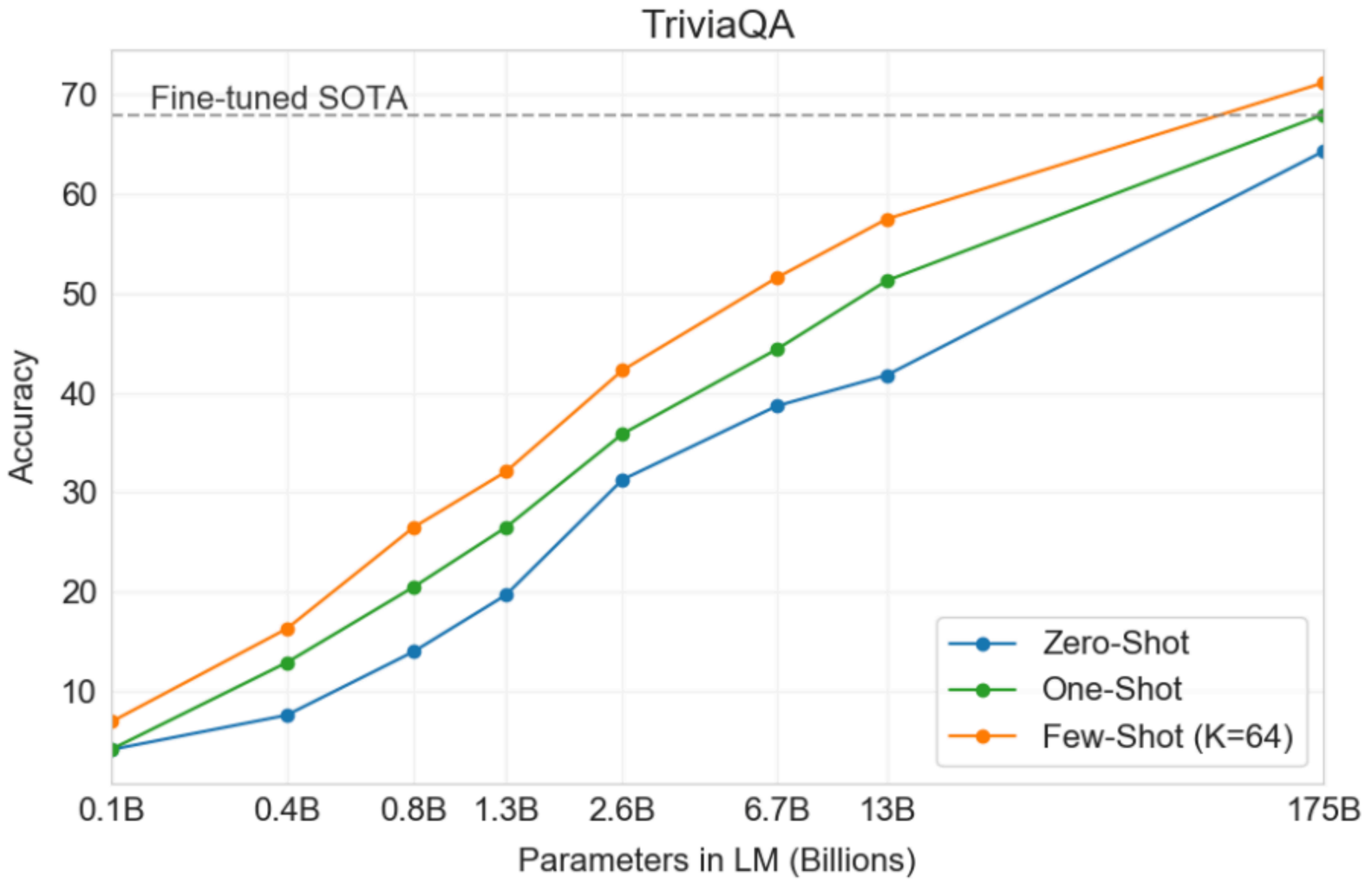
GPT-3 performance on SuperGLUE





(Brown et al., 2020): Language Models are Few-Shot Learners

GPT-3 performance on TriviaQA



(Brown et al., 2020): Language Models are Few-Shot Learners

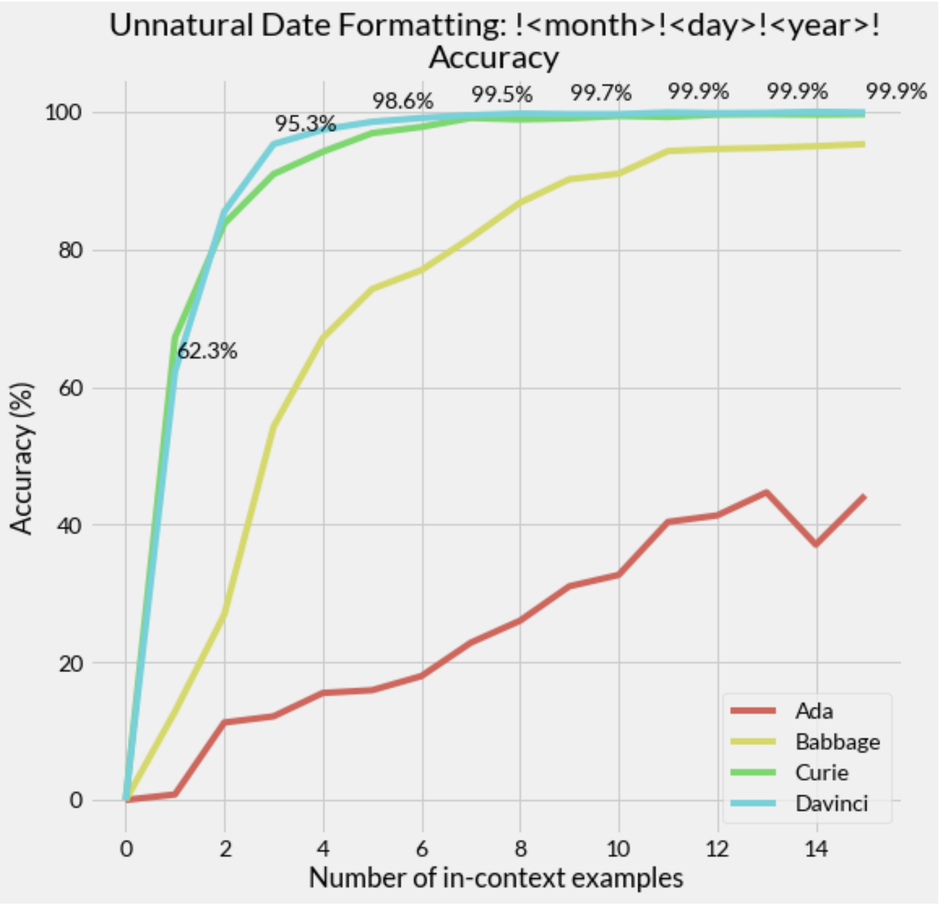
GPT-3's in-context learning

```
Input: 2014-06-01
Output: !06!01!2014!
Input: 2007-12-13
Output: !12!13!2007!
Input: 2010-09-23
Output: !09!23!2010!
Input: 2005-07-23
Output: !07!23!2005!
```

in-context examples

test example

model completion



https://ai.stanford.edu/blog/in-context-learning/

Extension: Chain-of-thought (CoT)

Labeled data

```
Question: Leah had 32 chocolates and her sister had 42.
If they ate 35, how many pieces do they have left?

Answer: 39
```


(Wei et al., 2022): Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

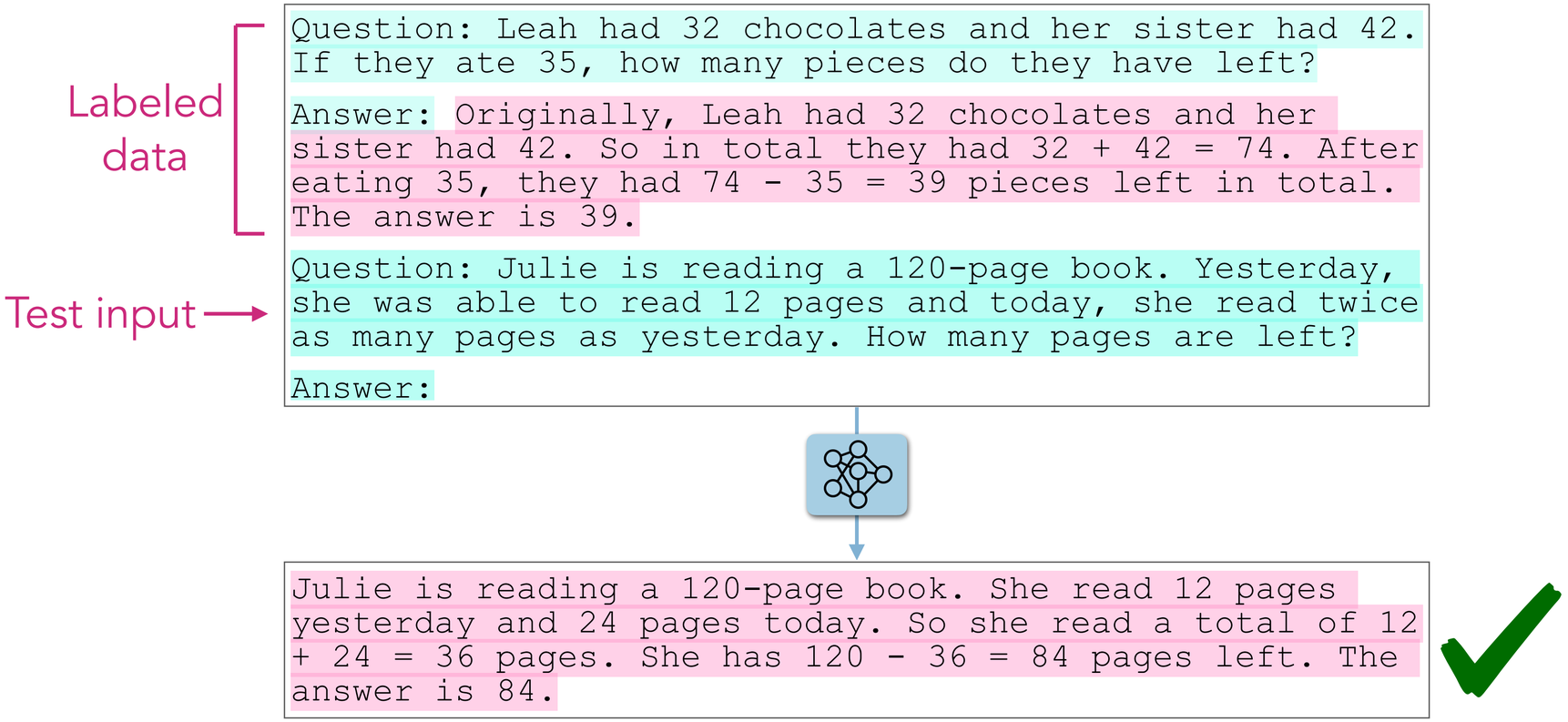
Extension: Chain-of-thought (CoT)

Labeled data

```
Question: Leah had 32 chocolates and her sister had 42.
If they ate 35, how many pieces do they have left?

Answer: Originally, Leah had 32 chocolates and her
sister had 42. So in total they had 32 + 42 = 74. After
eating 35, they had 74 - 35 = 39 pieces left in total.
The answer is 39.
```


(Wei et al., 2022): Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Extension: Chain-of-thought (CoT)



- You need to use CoT with in-context learning
- Training a model to do CoT = “Reasoning models” (will be covered in several weeks)

(Wei et al., 2022): Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Feb 19 lecture starts from here

CS 288 Advanced Natural Language Processing

Course website: cal-cs288.github.io/sp26

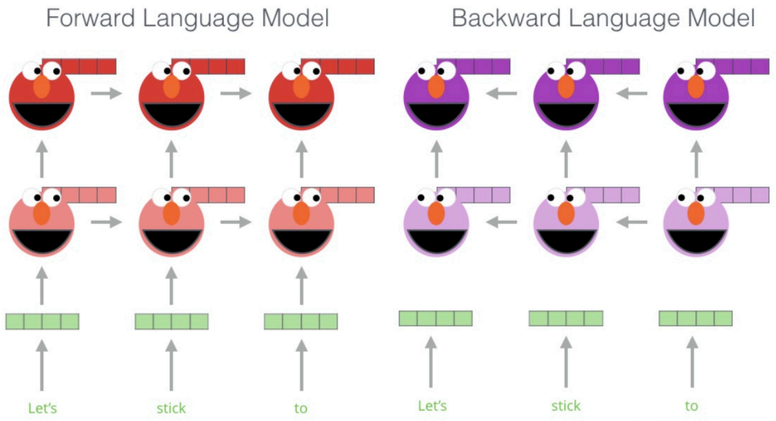
Ed: edstem.org/us/join/XvztdK

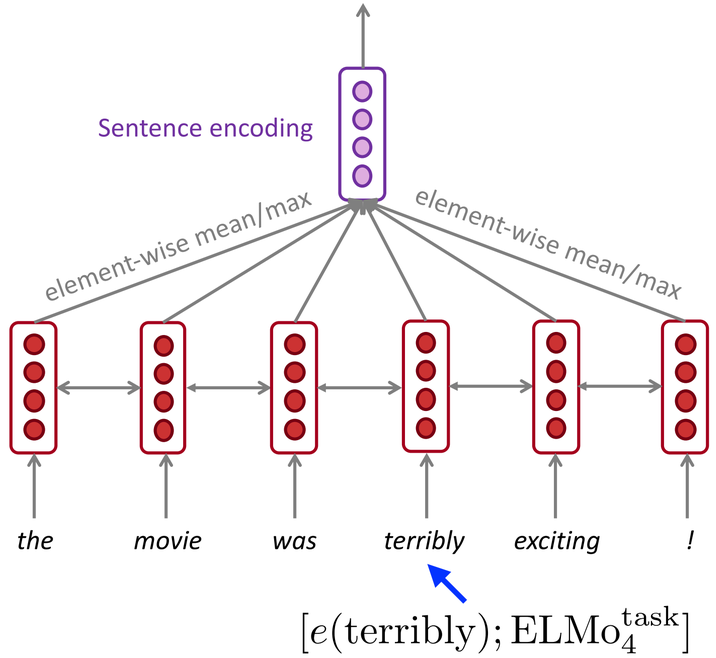
- Class starts at 15:40!
- Reminder to refresh the lecture slides (PDFs) on the lecture day
- Lecture plans:
    - Complete pre-training, fine-tuning, & prompting (15min)
    - Pre-training advanced (65min)

Recap: From feature-based to fine-tuning to prompting

Feature-based approach

- Produces word embeddings that can be used as **input representations** of existing neural models
- Case study: word2vec, Glove, ELMo





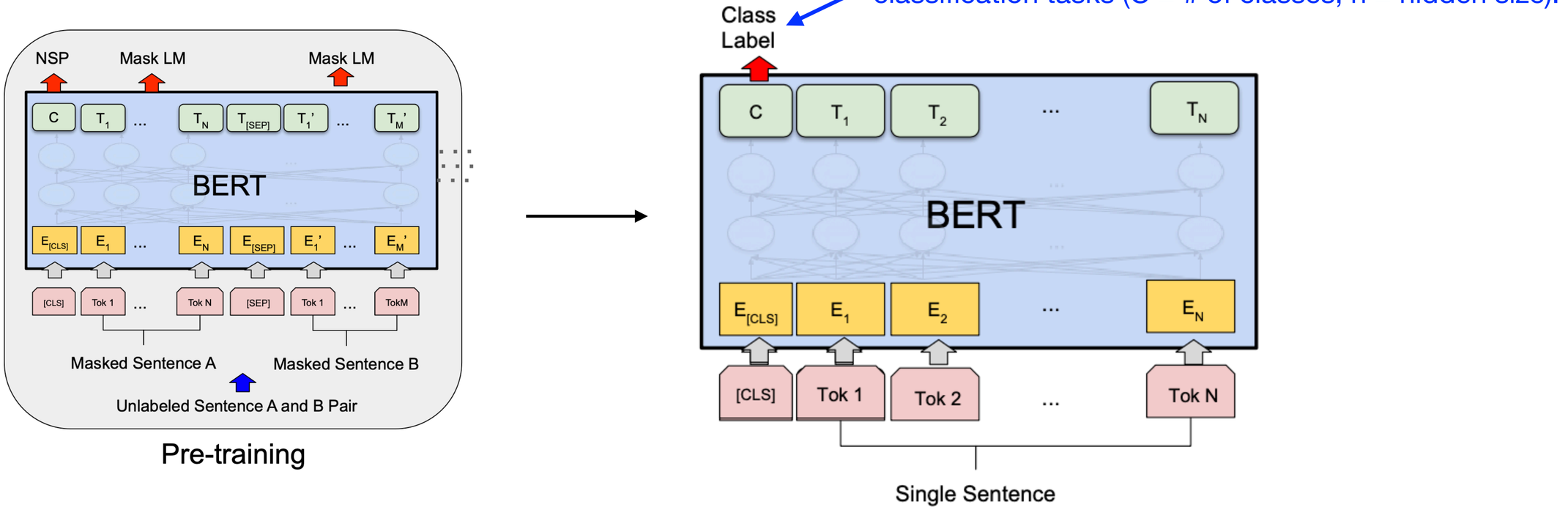
Pre-training (Only once, task-agnostic)

Plug them into the main model (e.g., LSTMs), then train a model for a **specific downstream task**

Fine-tuning based approach

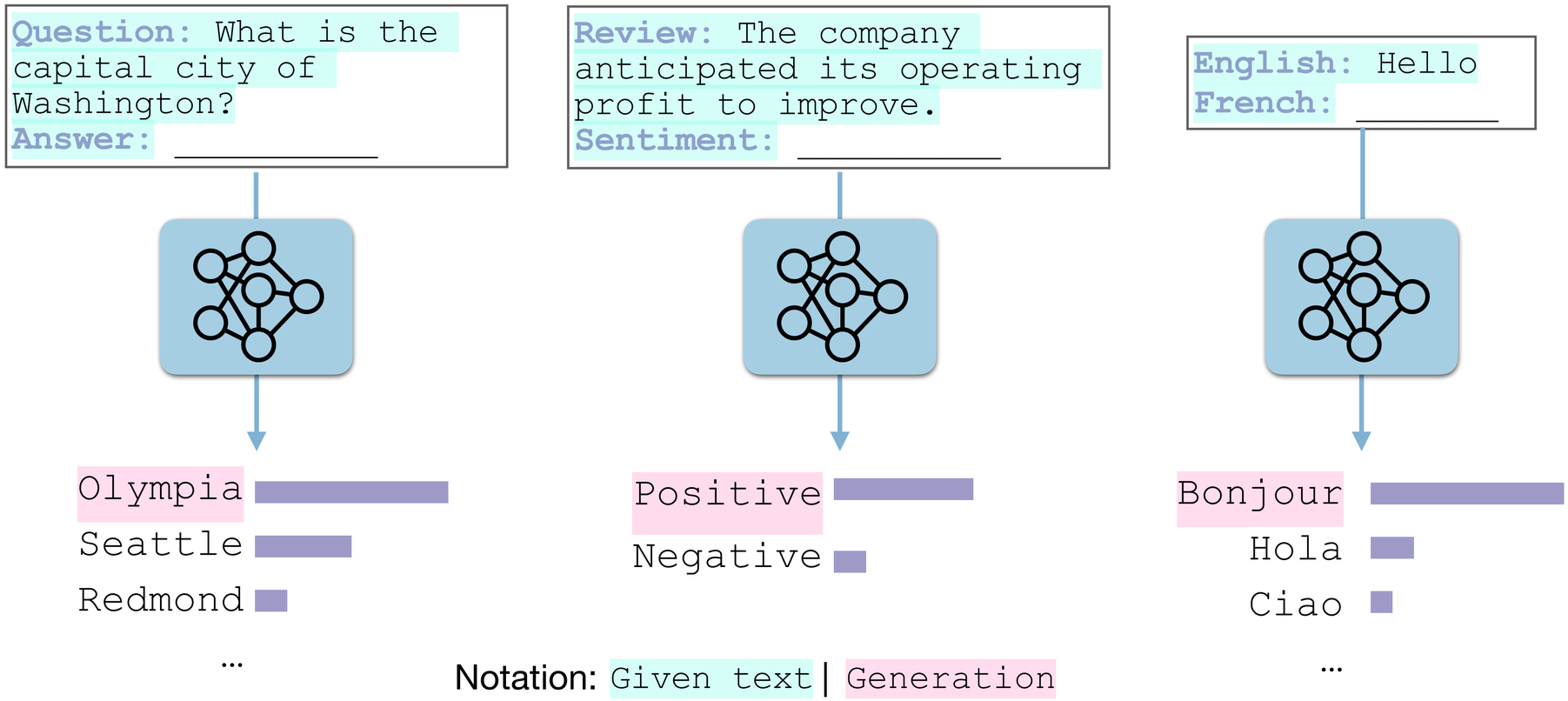
- The full model architecture & weights are re-used and refined on the labeled dataset, with only minimal task-specific parameters added (such as a single linear layer)
- Case study: BERT (110M & 340M), RoBERTa (110M & 340M), GPT-1/2 (110M to 1.5B), T5 (up to 11B)

We just need to introduce $C \times h$ parameters for classification tasks (C = # of classes, h = hidden size)!



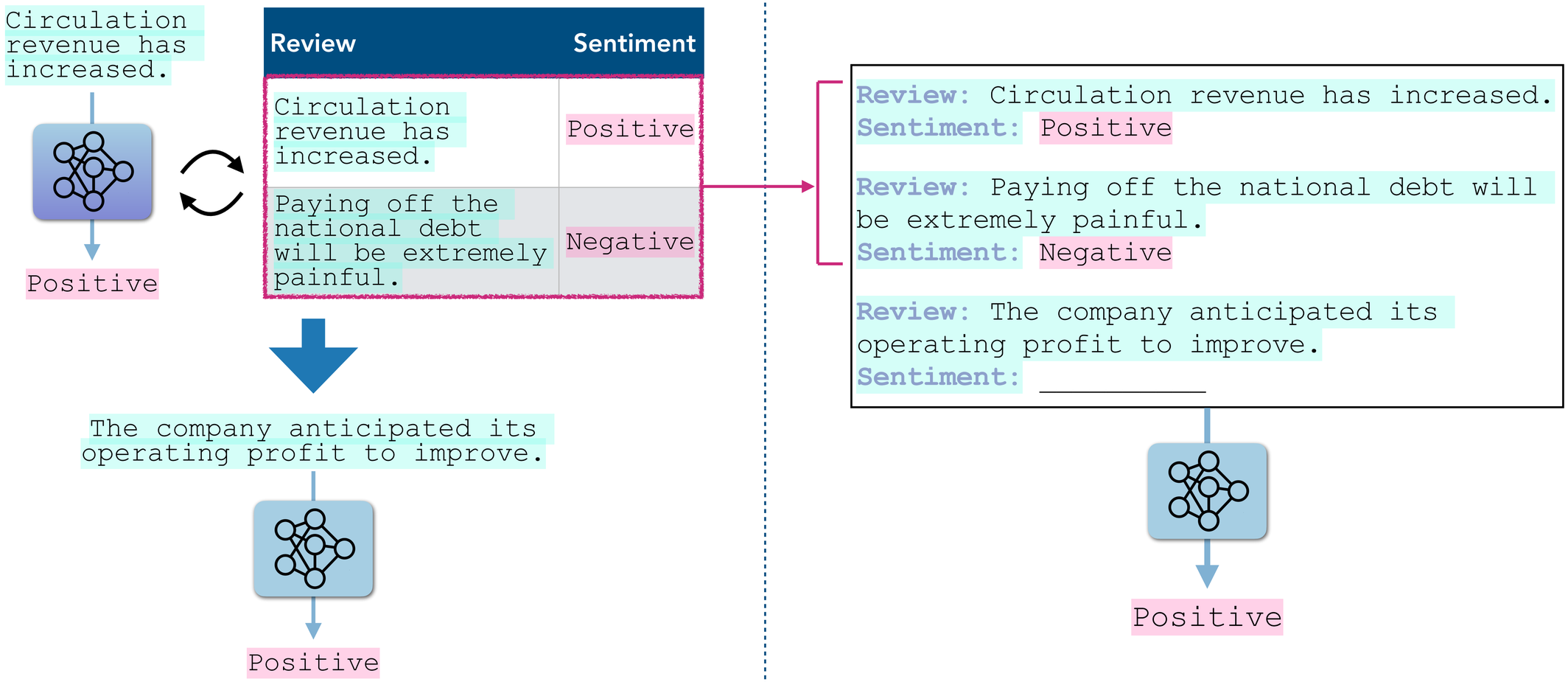
Prompting-based approach

- The pre-trained model (with same weights) is used as-is across tasks — simply by formatting the input appropriately, either zero-shot, or few-shot (in-context learning).
- Case study: GPT-2 (1.5B), GPT-3 (175B)

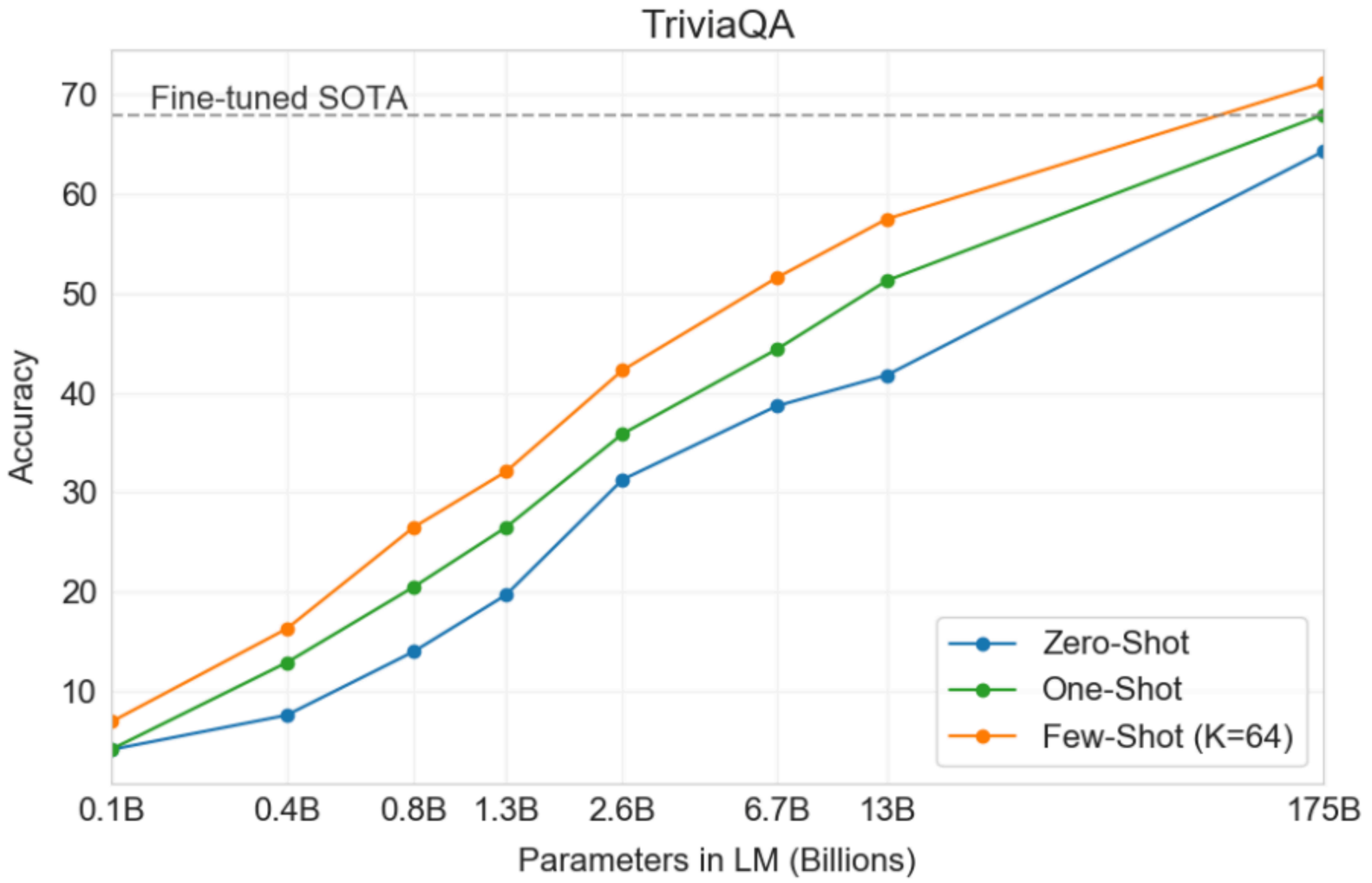


Notation: `Given text` | `Generation`

Finetuning vs. In-context learning



GPT-3 performance on TriviaQA



(Brown et al., 2020): Language Models are Few-Shot Learners

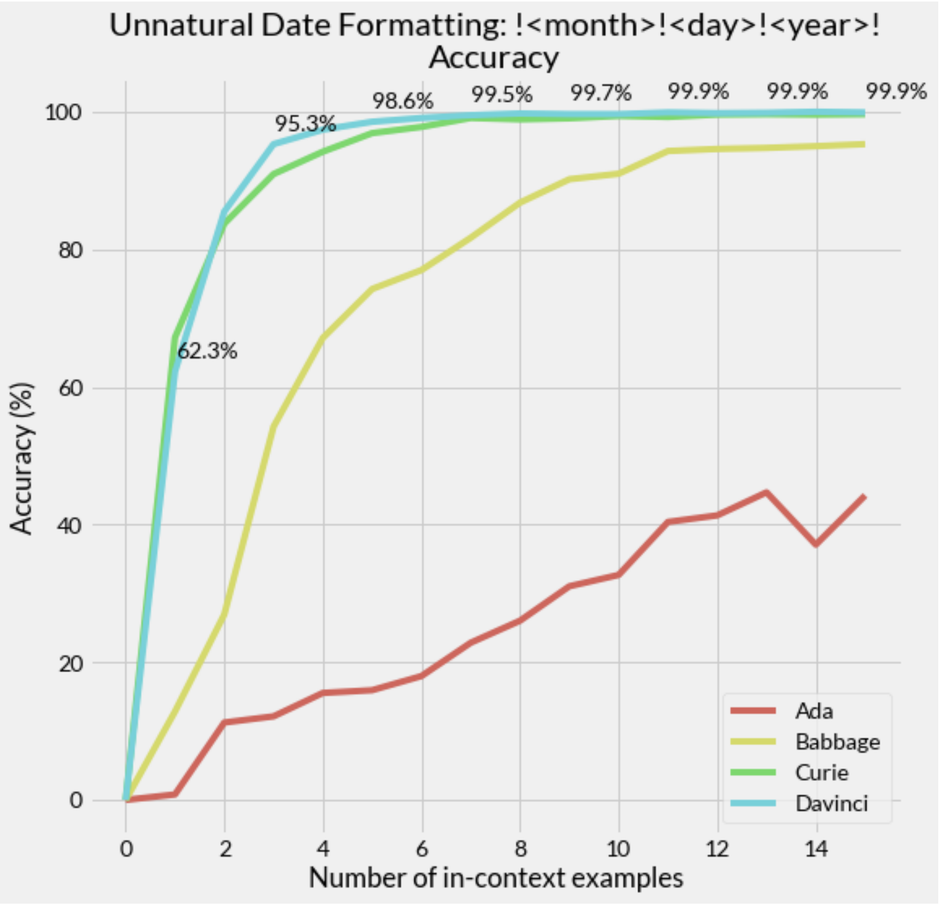
GPT-3's in-context learning

```
Input: 2014-06-01
Output: !06!01!2014!
Input: 2007-12-13
Output: !12!13!2007!
Input: 2010-09-23
Output: !09!23!2010!
Input: 2005-07-23
Output: !07!23!2005!
```

in-context examples

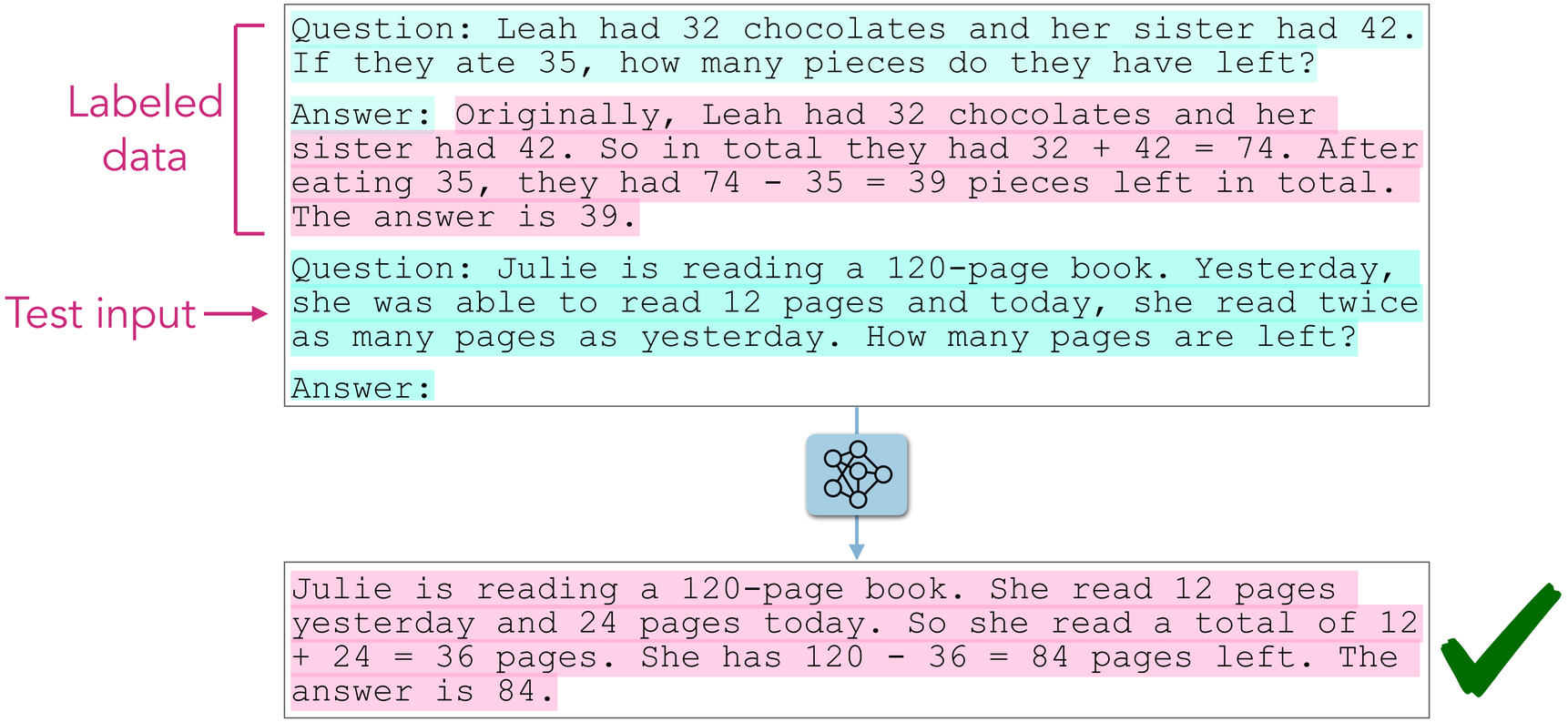
test example

model completion



https://ai.stanford.edu/blog/in-context-learning/

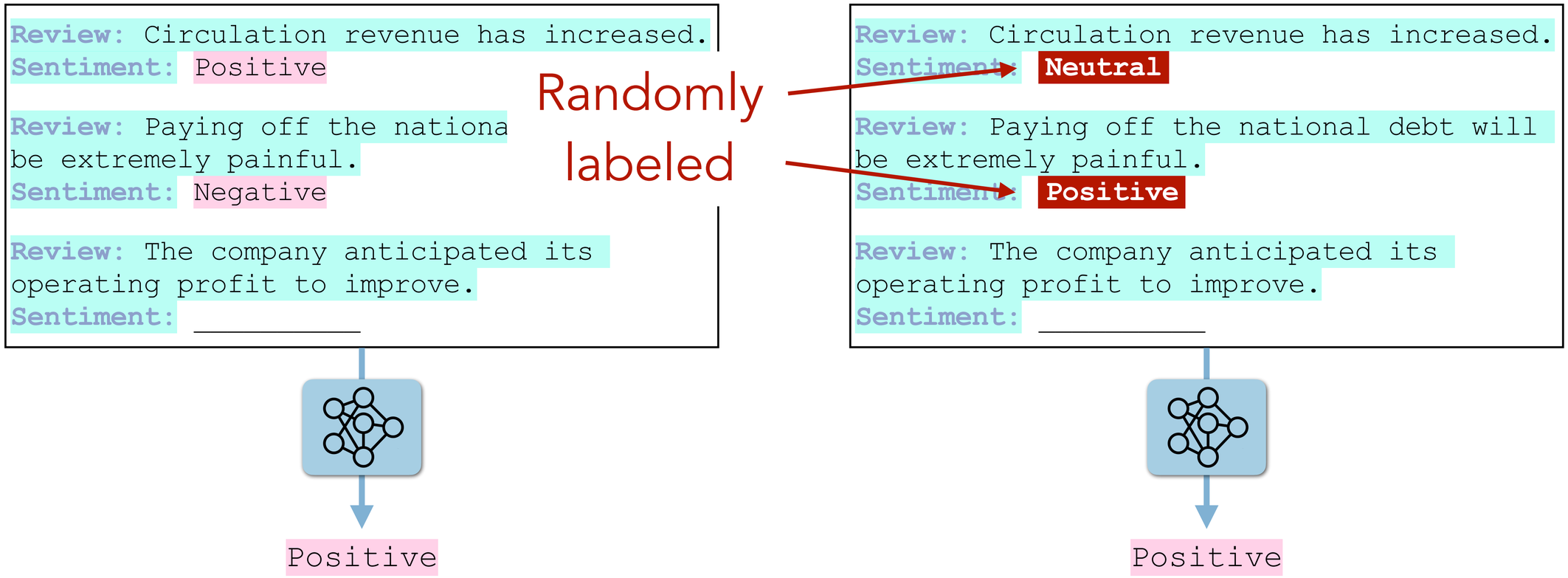
Extension: Chain-of-thought (CoT)



- You need to use CoT with in-context learning
- Training a model to do CoT = “Reasoning models” (will be covered in several weeks)

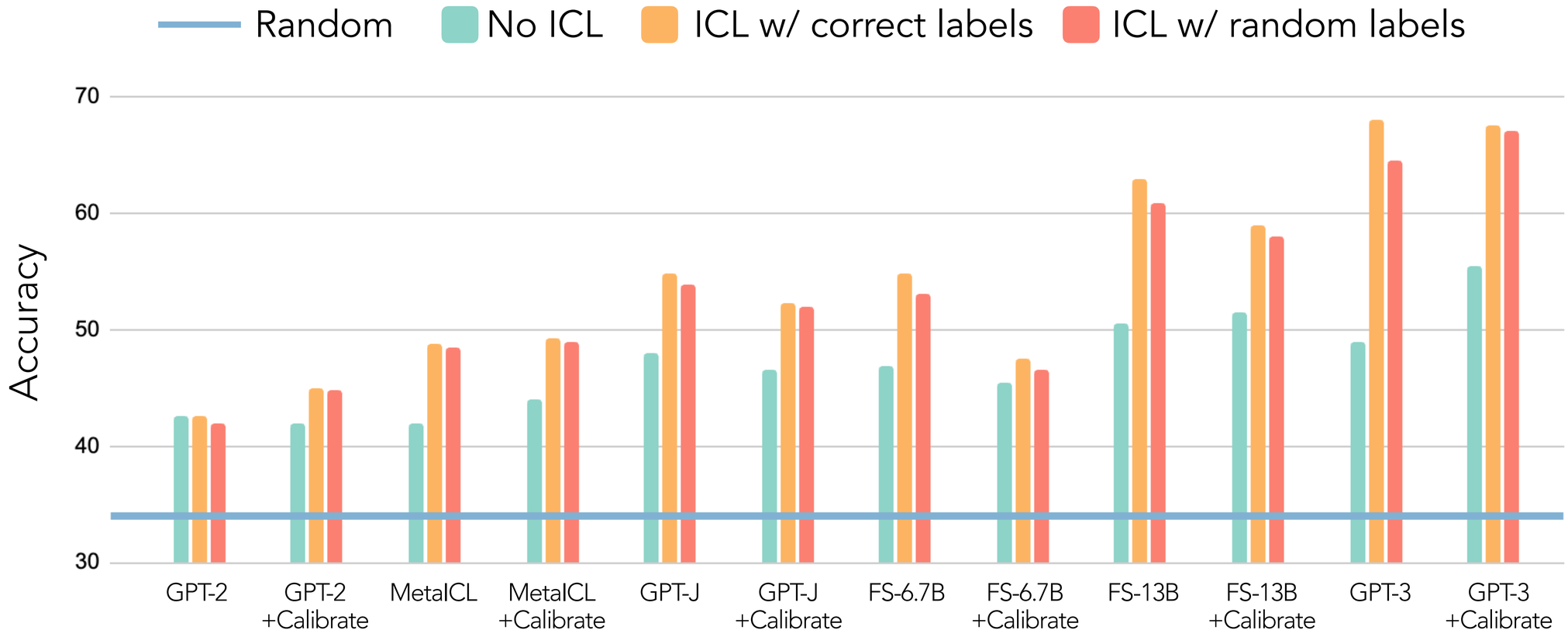
(Wei et al., 2022): Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

(New) Does in-context learning *learn*?



(Min et al., 2022): Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?

Does in-context learning *learn*?

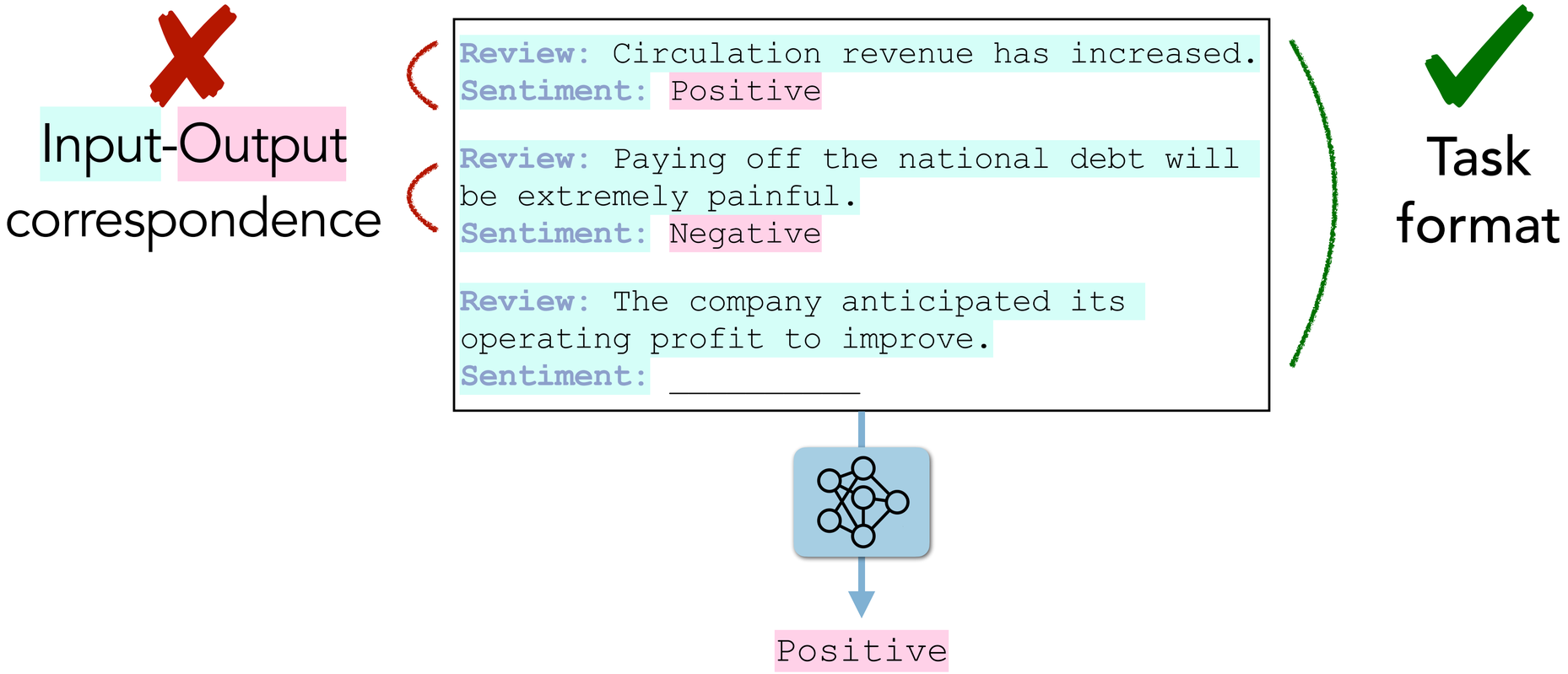


Recovers **95%** of ICL accuracy

(Min et al., 2022): Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?

In-context learning: Hypothesis

1. Labeled data teaches format, not correspondence



In-context learning: Hypothesis

2. Correspondence is learned from LM training data

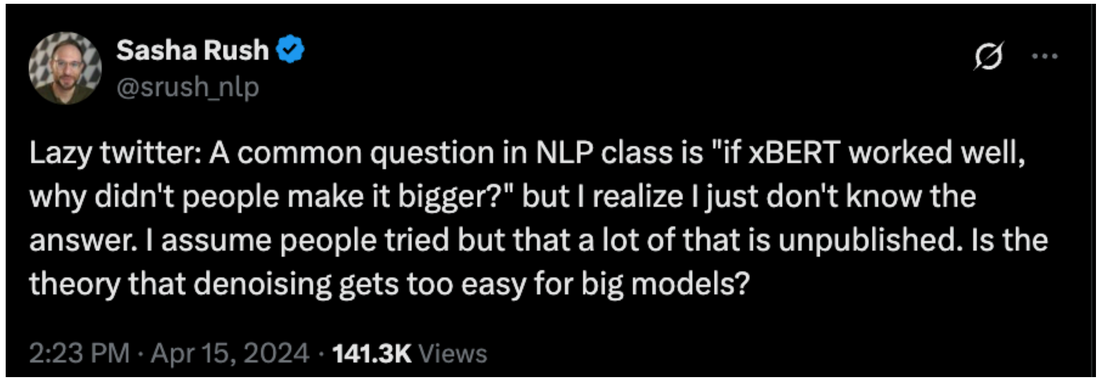
What happened to the NLP field after GPT-3?

Recap: Three major forms of pre-training

Architecture	Pretraining objective	Examples
Transformer Encoder	Masked language models	BERT, RoBERTa, ELECTRA
Transformers Encoder-Decoder	Span Corruption	T5, BART
Transformers Decoder	Autoregressive language models (Causal language models)	GPT-2, GPT-3, Llama

However, almost all large models we see are decoder-only, autoregressive language models

Open ended question:
Why did generative LMs win out?



What happened to BERT & T5? On Transformer Encoders, PrefixLM and Denoising Objectives

https://www.yitay.net/blog/model-architecture-blogpost-encoders-prefixlm-denoising

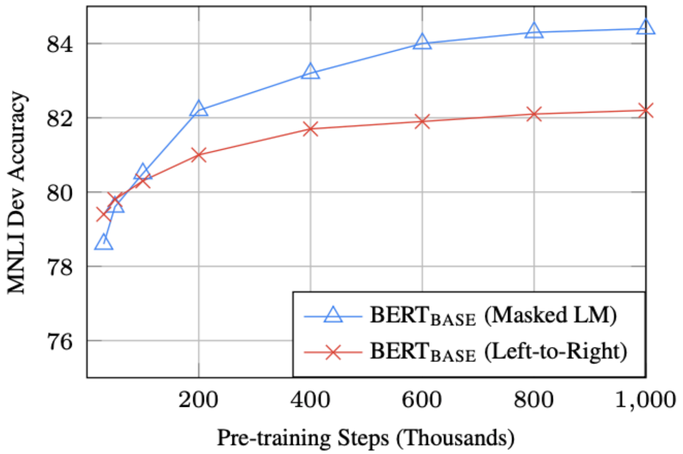
Why did generative LMs win out? (1/2)

- Encoder-only models
  - Designed for representation learning
  - Not naturally generative
  - Requires task-specific fine-tuning
    - Became less useful because large models started to work very well with (zero-shot or few-shot) prompting, which requires generation

- Encoder-decoder models
  - Support generation
  - Benefit from bidirectional context on the encoder side
  - **Then, weren't they scaled further (after T5/BART)?**

Why did generative LMs win out? (2/2)

- Bidirectional attention is only important at smaller scale?



(Devlin et al. , 2018)

- Scaling generative LMs is more ***practical***.

Training signals & "density"

- Decoder-only gets signal for every single token in the sequence — **7x more "signals" per FLOP of compute** compared to MLM with 15% of masked tokens

Reduction of the "ablation tax"

- Encoder-decoder: What's the masking ratio? How to allocate params between encoder and decoder?
- Decoder: A stack of identical blocks.

The KV cache and inference efficiency

Paradigm shift: one model does it all

NLP before 2020:

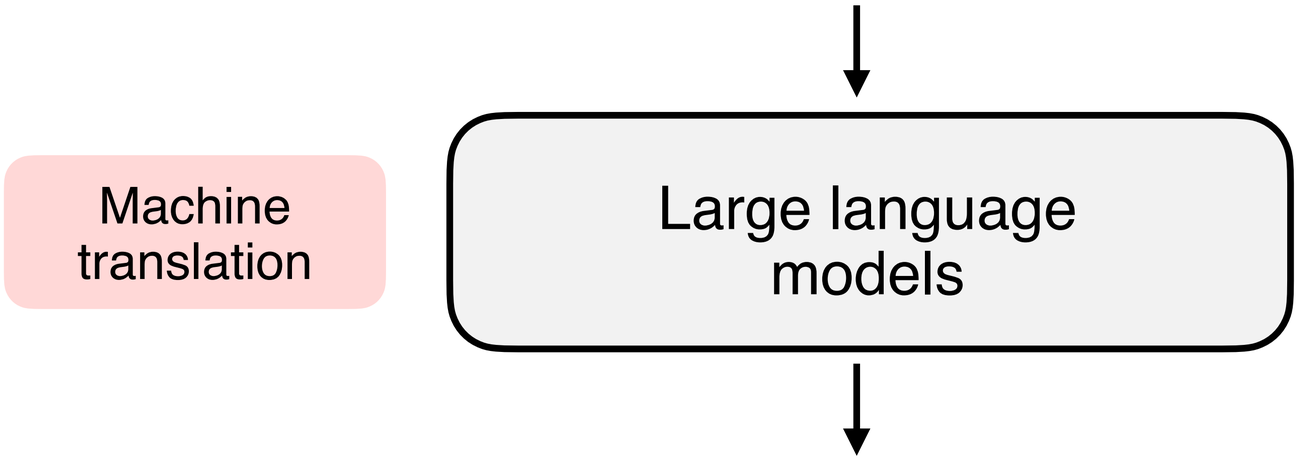
Sentiment analysis

Question answering

Machine translation

Text summarization

Prompt: **Translate the following sentence from English to German**: "The dinner was great"



Completion: Das Abendessen war großartig

Paradigm shift: one model does it all

NLP before 2020:

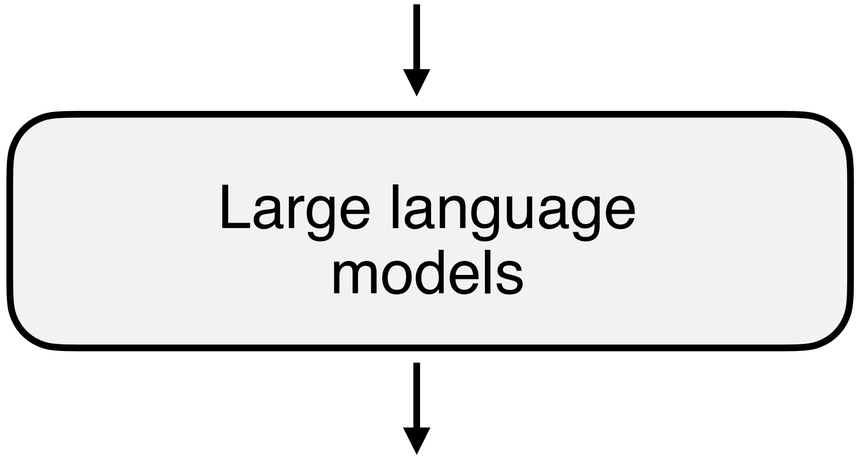
Sentiment analysis

Question answering

Machine translation

Text summarization

Prompt: Given the following paragraph […], **how would you phrase it in a few words?**

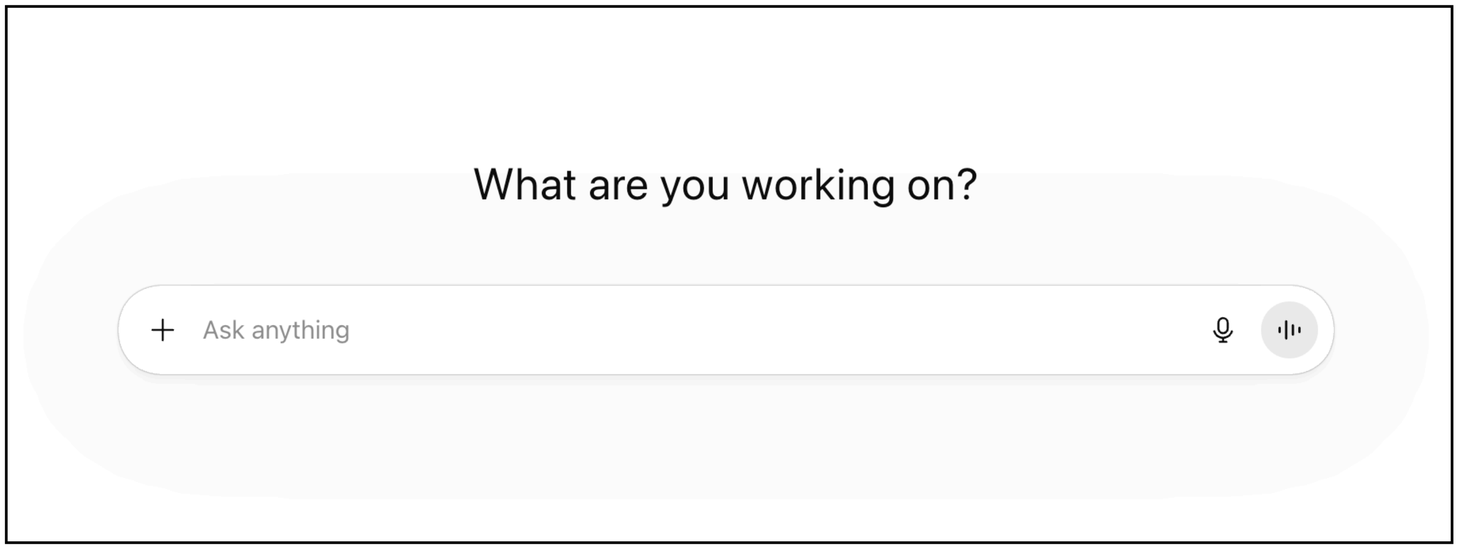


Text summarization

Completion: Graffiti artist Banksy is believed to be behind […]

- "**Foundation Model**" (Bommasani et al., 2021)
- Zero or very few human-annotated examples required

This is how we use ChatGPT today



Use-case	Prompt
Brainstorming	List five ideas for how to regain enthusiasm for my career
Generation	Write a short story where a bear goes to the beach, makes friends with a seal, and then returns home.
generation	Write an outline for an essay about John von Neumann and his contributions to computing: I. Introduction, his life and background A: His early life B:
rewrite	Covert my resume into a profile overview. {resume} Profile overview:
rewrite	Rephrase this for me: "I can't seem to find out how to work this darn thing." Alternate phrasing: "

Key LLM artifacts

Year	Model	Size	Status	Developer
2020	GPT-3	175B	Proprietary	OpenAI
2021	MT-NLG	530B	Proprietary	Microsoft/Nvidia
2022	PaLM	540B	Proprietary	Google
2022	Chinchilla	70B	Proprietary	Google DeepMind
2022	BLOOM	176B	Open-weight	BigScience
2023	Llama 1	7B - 65B	Open-weight	Meta
2023	Pythia	70M - 12B	Open-source	EleutherAI
2023	GPT-4	~1.8T (MoE)	Proprietary	OpenAI
2023	Llama 2	7B - 70B	Open-weight	Meta
2023	Mixtral	7B	Open-weight	Mistral AI
2024	OLMo	1B - 7B	Open-source	Ai2
2024	Liama 3	8B - 405B	Open-weight	Meta
2024	Wen 2.5	0.5B - 72B	Open-weight	Alibaba
2025	DeepSeek-V3	671B (MoE)	Open-weight	DeepSeek

*Excluding reasoning models (will be covered in a later lecture)

Many LLMs became proprietary

The creators of a revolutionary AI system that can write news stories and works of fiction - dubbed "deepfakes for text" - have taken the unusual step of not releasing their research publicly, for fear of potential misuse.

OpenAI, an nonprofit research company backed by Elon Musk, Reid Hoffman, Sam Altman, and others, says its new AI model, called GPT2 is so good and the risk of malicious use so high that it is breaking from its normal practice of releasing the full research to the public in order to allow more time to discuss the ramifications of the technological breakthrough.

The Guardian. February 2019.

Case study: Llama (Meta)

RESEARCH

Introducing LLaMA: A foundational, 65-billion-parameter large language model

February 24, 2023

- Smaller models trained on 1.4T, high-quality & publicly available data
- “LLaMA-13B outperforms GPT-3 (175B) on most benchmarks, and LLaMA-65B is competitive with the best models, Chinchilla-70B and PaLM-540B”

(Touvron et al., 2023): LLaMA: Open and Efficient Foundation Language Models

Case study: Llama (Meta)

Generation	Release date	# parameters	Training data size (tokens)	Context length (tokens)
Llama 1	February 2023	6.7B to 65.2B	1T—1.4T	2,048
Llama 2	July 2023	6.7B to 69B	2T	4,096
Llama 3 (3.1, 3.2, 3.3)	April 2024 (3.3: December 2024)	8B to 70B	15T+	128,000
Llama 4	April 2025	109B to 2T (MoE)	40T+	10,000,000 (?)

Arguably more widely used than Llama 4

Meta. 2024. "The Llama 3 Herd of Models"

Quick quiz

Which statement is incorrect?

(A) Llama-3 8B is roughly comparable performance to GPT-3 175B.

(B) Llama-3 [2023] is trained with 50x larger data than GPT-3.

(C) One of the best language model trained primarily by university in the US as of 2025 is trained with 8x larger data than GPT-3.

(D) None

(D) is correct.

A, B: Llama-3 8B trained for 15-trillion tokens $\approx$ GPT-3 175B trained for 0.3-trillion tokens

C: DCLM 7B trained for 3-trillion tokens (and is comparable to Llama-3 8B/GPT-3 175B, at least on the benchmarks they evaluated on).

Remaining topics in the course

02/19 (TODAY): Pre-training advanced topics

Science of scaling & data

"What it takes to reach Llama 3 Base"

02/24 : Post-training

"What it takes to reach Llama 3 Instruct / early ChatGPT"

02/26: Inference & Evaluation

03/03: Experimental design & Human annotation

"What it takes to do NLP research"

03/05, 03/10: Architecture advanced topics

How to make LLM more factual? (Retrieval augmented generation)

Beyond basic Transformer architectures we learned (e.g., Mixture-of-experts)

Up next: Impact & Social implication, reasoning models, agents

"What it takes do reach 2026 models: DeepSeek, Qwen, GPT-OSS, GLM"

Questions?

Acknowledgement

Princeton COS 484 by Danqi Chen, Tri Dao, Vikram Ramaswamy