# Efficient adaptation

Mar 26, 2026

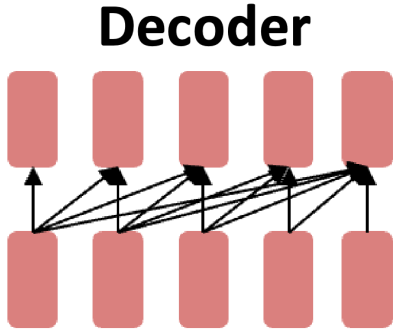
*Acknowledgment: Slides based on materials by CS224N @ Stanford University.

# Outline

# Outline

# Emergent abilities of LLMs: GPT (2018)

Let's revisit the Generative Pretrained Transformer (GPT) models from OpenAI as an example:




**GPT** (117M parameters; Radford et al., 2018)

- Transformer decoder with 12 layers.
- Trained on BooksCorpus: over 7000 unique books (4.6GB text).

Showed that language modeling at scale can be an effective pretraining technique for downstream tasks like natural language inference.

entailment

[START] *The man is in the doorway* [DELIM] *The person is near the door* [EXTRACT]

# Emergent abilities of LLMs: GPT-2 (2019)

Let's revisit the Generative Pretrained Transformer (GPT) models from OpenAI as an example:

**GPT-2** (1.5B parameters; Radford et al., 2019)

- Same architecture as GPT, just bigger (117M -> 1.5B)
- But trained on **much more data**: 4GB -> 40GB of internet text data (WebText)
  - Scrape links posted on Reddit w/ at least 3 upvotes (rough proxy of human quality)

---

**Language Models are Unsupervised Multitask Learners**

---

Alec Radford [*1] Jeffrey Wu [*1] Rewon Child [1] David Luan [1] Dario Amodei [**1] Ilya Sutskever [**1]

# Outline

# Emergent zero-shot learning

- One key emergent ability in **GPT-2** is **zero-shot learning**.
- **Zero-shot learning**: the ability to perform many tasks *without training examples and without gradient updates*.
- The model can perform tasks simply by:
  - Example 1: Question Answering
    - Passage: Tom Brady . . .
      Q: Where was Tom Brady born?
      A: . . .

# Emergent zero-shot learning

The model can perform tasks simply by:

- Example 2: Comparing probabilities of sequences
  - Sentence:
    *The cat couldn't fit into the hat because it was too big.*
  - What does *it* refer to?
  - Compare probabilities:

    $P(\ldots \text{because the cat was too big}) \geq P(\ldots \text{because the hat was too big})$

# Emergent abilities of LLMs: GPT-3 (Brown et al., 2020)

GPT-3 (175B parameters; Brown et al., 2020)

- Another increase in size (1.5B $\rightarrow$ 175B)
- and data (40GB $\rightarrow$ over 600GB)
- will be presented today:

# Emergent abilities of LLMs: GPT-3 (Brown et al., 2020)

GPT-3 (175B parameters; Brown et al., 2020)

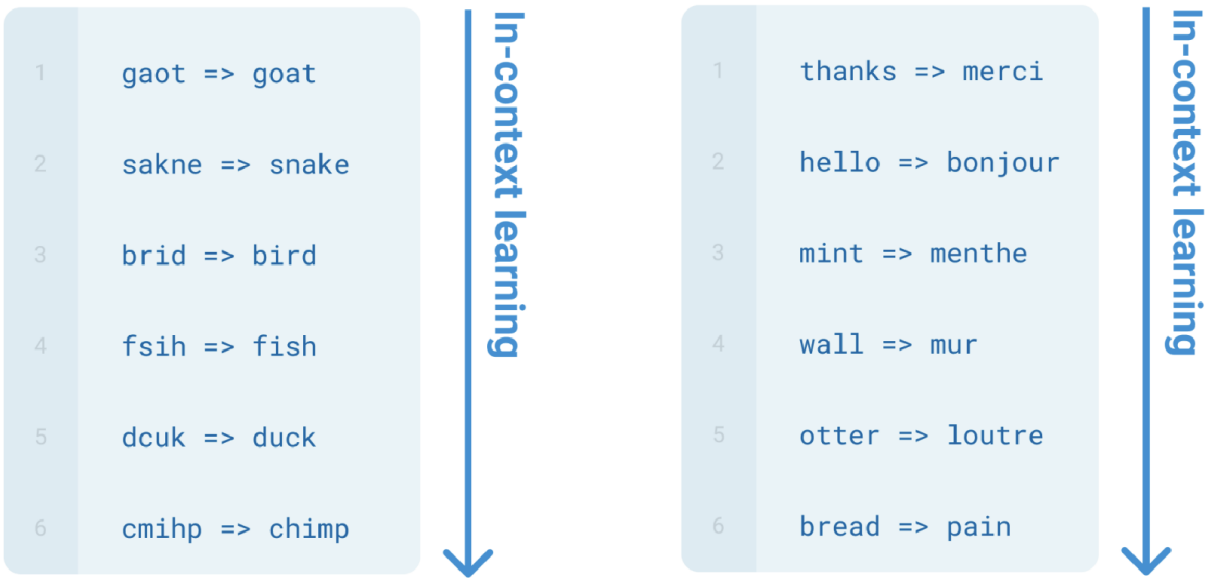
- Another increase in size (1.5B $\rightarrow$ 175B)
- and data (40GB $\rightarrow$ over 600GB)
- will be presented today:

**Language Models are Few-Shot Learners**

**Tom B. Brown*** **Benjamin Mann*** **Nick Ryder*** **Melanie Subbiah***

# Emergent few-shot learning

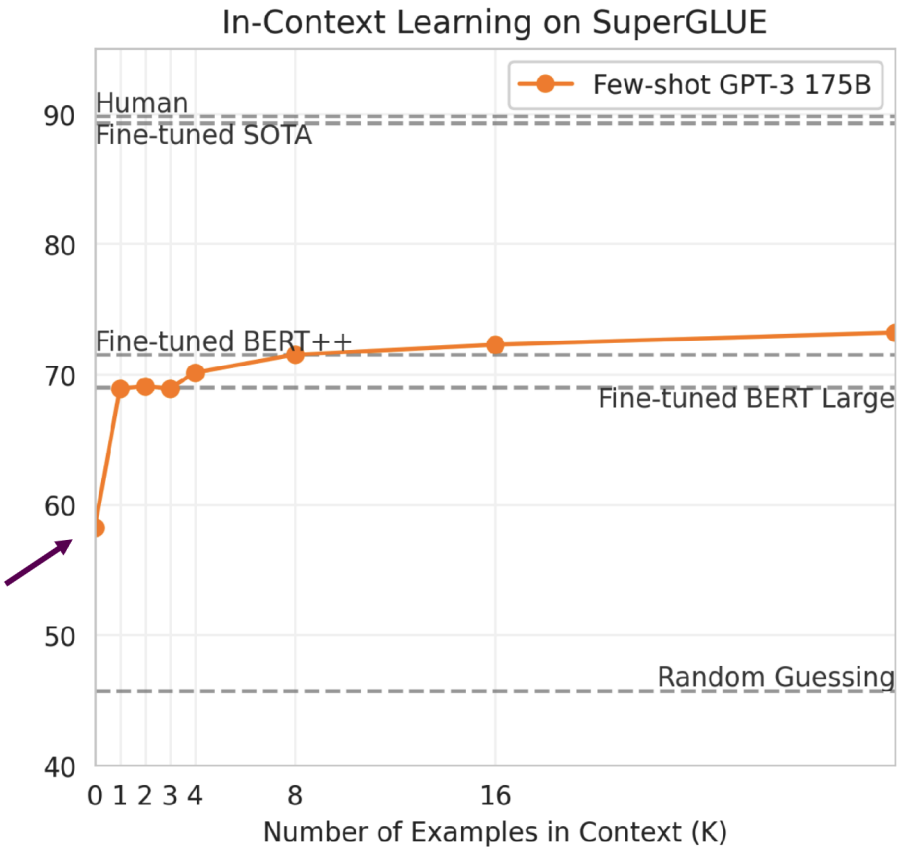
- Specify a task by simply prepending examples of the task before your example
- also called **in-context learning**, to stress that *no gradient updates* are performed when learning a new task



# Emergent few-shot learning

**Zero-shot**

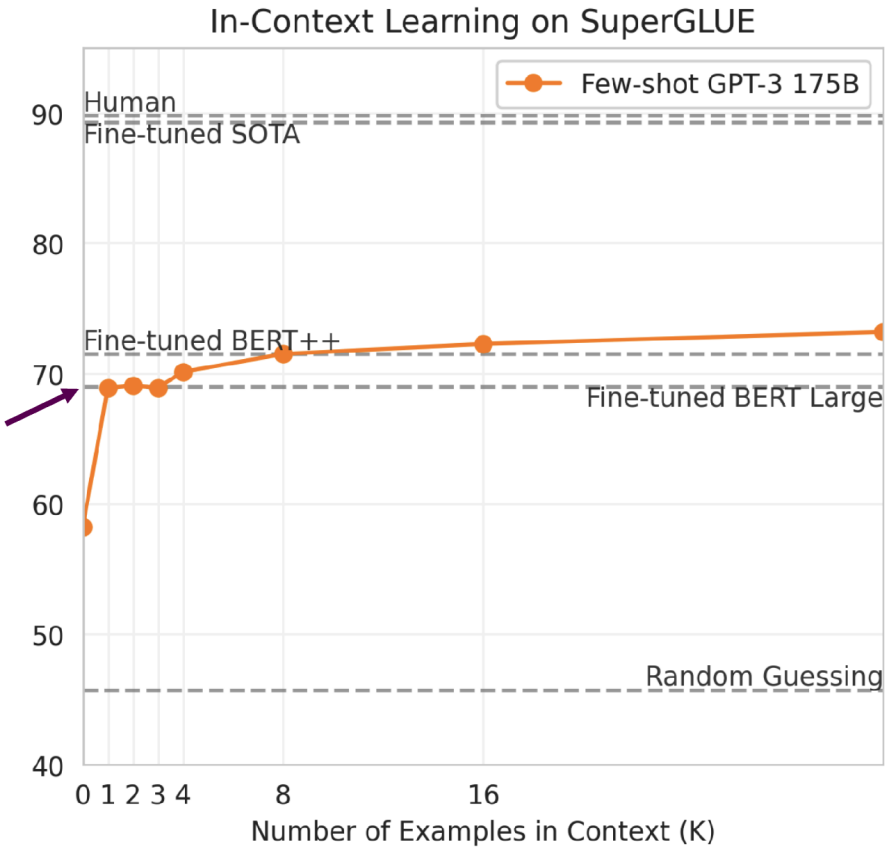
```
1  Translate English to French:
2  cheese =>
```



# Emergent few-shot learning

**One-shot**

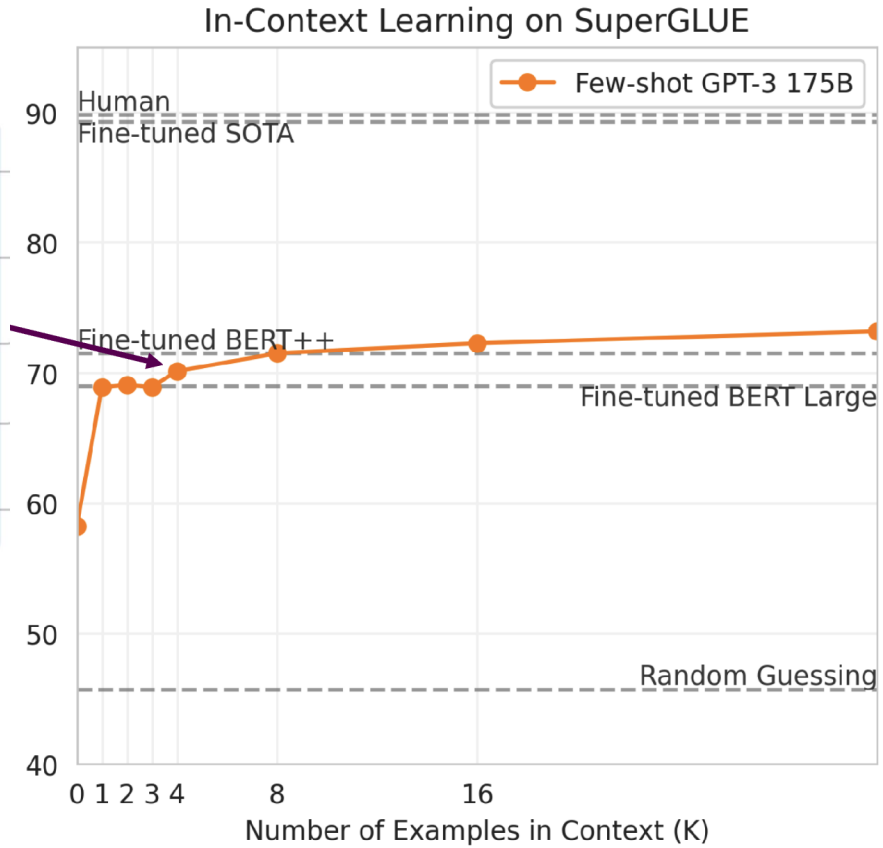
```
1  Translate English to French:
2  sea otter => loutre de mer
3  cheese =>
```



# Emergent few-shot learning

**Few-shot**

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



# Compared to traditional fine-tuning

## Zero/few-shot prompting

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

## Traditional fine-tuning

```
1  sea otter => loutre de mer
```

gradient update

```
1  peppermint => menthe poivrée
```

gradient update

```
1  cheese =>
```

RIT Rochester Institute of Technology

# Limits of Prompting

- Some tasks appear too difficult for even large language models to solve through **prompting alone**.
- This is especially true for tasks requiring **multi-step reasoning**.
- (Humans also find these tasks challenging.)

**Example**

- $19583 + 29534 =$
- $98394 + 49384 =$
- $14777 + 82938 =$
- $21234 + 12347 =$
- $41729 + 93847 =$
- $39299 +$ ?

**Key Idea**

**Solution: Change the prompt!**

# Chain-of-thought prompting

## Standard Prompting

**Model Input**

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

A: The answer is 11.

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

**Model Output**

A: The answer is 27. ✗

## Chain-of-Thought Prompting

**Model Input**

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

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

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

**Model Output**

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

# Chain-of-thought prompting

**Model Input**

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

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

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

**Model Output**

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

**Do we even need examples of reasoning? Can we just ask the model to reason through things?**

# **Zero-shot** chain-of-thought prompting

**Model Input**

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

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

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

**Model Output**

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

Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?

A: ***Let's think step by step.*** *There are 16 balls in total. Half of the balls are golf balls. That means there are 8 golf balls. Half of the golf balls are blue. That means there are 4 blue golf balls.* ✓

# Zero-shot CoT

| | MultiArith | GSM8K |
|---|---|---|
| **Zero-Shot** | **17.7** | **10.4** |
| Few-Shot (2 samples) | 33.7 | 15.6 |
| Few-Shot (8 samples) | 33.8 | 15.6 |
| **Zero-Shot-CoT** | **78.7** | **40.7** |
| Few-Shot-CoT (2 samples) | 84.8 | 41.3 |
| Few-Shot-CoT (4 samples : First) (*1) | 89.2 | - |
| Few-Shot-CoT (4 samples : Second) (*1) | 90.5 | - |
| Few-Shot-CoT (8 samples) | 93.0 | 48.7 |

Greatly outperforms zero-shot →

Manual CoT still better →

# Zero-shot CoT

| No. | Category | Zero-shot CoT Trigger Prompt | Accuracy |
|---|---|---|---|
| 1 | LM-Designed | Let's work this out in a step by step way to be sure we have the right answer. | **82.0** |
| 2 | Human-Designed | Let's think step by step. (*1) | 78.7 |
| 3 | | First, (*2) | 77.3 |
| 4 | | Let's think about this logically. | 74.5 |
| 5 | | Let's solve this problem by splitting it into steps. (*3) | 72.2 |
| 6 | | Let's be realistic and think step by step. | 70.8 |
| 7 | | Let's think like a detective step by step. | 70.3 |
| 8 | | Let's think | 57.5 |
| 9 | | Before we dive into the answer, | 55.7 |
| 10 | | The answer is after the proof. | 45.7 |
| - | | (Zero-shot) | 17.7 |

# Limitations of Prompting

- **Inefficiency**: The prompt must be processed every time the model makes a prediction.
- **Poor performance**: Prompting generally performs worse than fine-tuning.
- **Sensitivity**: Results depend heavily on:
  - Wording of the prompt
  - Order of examples
- **Lack of clarity**: It is unclear what the model learns from prompts; even random labels can work

# Outline

# What is LoRA?

- LoRA = **Low-Rank Adaptation**
- Instead of updating the whole model:
  - Keep the large model **frozen**
  - Train only a **small set of new parameters**
- Key idea:
  - Don't change everything
  - Learn only **small adjustments**
- $\rightarrow$ Faster and more memory-efficient

# How does LoRA work?

- Original weight:

$$W$$

- LoRA adds a small update:

$$W + \Delta W$$

- Instead of learning full $\Delta W$, LoRA uses:

$$\Delta W = A \times B$$

  - $A, B$: small low-rank matrices
  - $\rightarrow$ far fewer parameters to learn

- Key idea:
  - Keep $W$ fixed
  - Learn only $A$ and $B$

# Why use LoRA?

- **Efficient**
  - Uses much less GPU memory
  - Faster training
- **Flexible**
  - One base model, many LoRA adapters
  - Easy to switch between tasks
- **Simple intuition**
  - Not relearning the whole model
  - Just learning small corrections

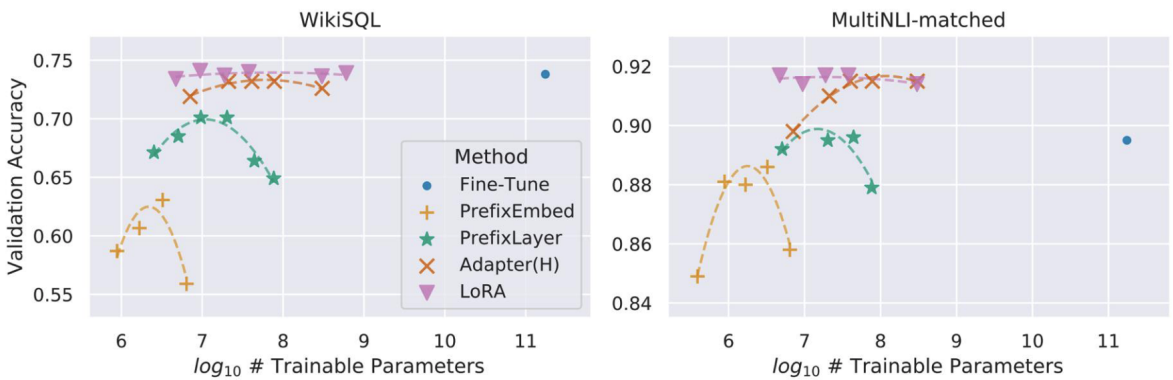
# Example implementation of LoRA

```
input_dim = 768  # e.g., the hidden size of the pre-trained model
output_dim = 768  # e.g., the output size of the layer
rank = 8  # The rank 'r' for the low-rank adaptation

W = ... # from pretrained network with shape input_dim x output_dim

W_A = nn.Parameter(torch.empty(input_dim, rank)) # LoRA weight A
W_B = nn.Parameter(torch.empty(rank, output_dim)) # LoRA weight B

# Initialization of LoRA weights
nn.init.kaiming_uniform_(W_A, a=math.sqrt(5))
nn.init.zeros_(W_B)

def regular_forward_matmul(x, W):
    h = x @ W
return h

def lora_forward_matmul(x, W, W_A, W_B):
    h = x @ W  # regular matrix multiplication
    h += x @ (W_A @ W_B)*alpha # use scaled LoRA weights
return h
```

Credit to https://lightning.ai/pages/community/article/lora-llm/

# LoRA in practice: scaling up to GPT-3 175B

| Model&Method | # Trainable Parameters | WikiSQL Acc. (%) | MNLI-m Acc. (%) | SAMSum R1/R2/RL |
|---|---|---|---|---|
| GPT-3 (FT) | 175,255.8M | **73.8** | 89.5 | 52.0/28.0/44.5 |
| GPT-3 (BitFit) | 14.2M | 71.3 | 91.0 | 51.3/27.4/43.5 |
| GPT-3 (PreEmbed) | 3.2M | 63.1 | 88.6 | 48.3/24.2/40.5 |
| GPT-3 (PreLayer) | 20.2M | 70.1 | 89.5 | 50.8/27.3/43.5 |
| GPT-3 ($\text{Adapter}^{\text{H}}$) | 7.1M | 71.9 | 89.8 | 53.0/28.9/44.8 |
| GPT-3 ($\text{Adapter}^{\text{H}}$) | 40.1M | 73.2 | **91.5** | 53.2/29.0/45.1 |
| GPT-3 (LoRA) | 4.7M | 73.4 | **91.7** | **53.8/29.8/45.9** |
| GPT-3 (LoRA) | 37.7M | **74.0** | **91.6** | 53.4/29.2/45.1 |




LoRA matches or exceeds the fine-tuning baseline on all three datasets

LoRA exhibits better scalability and task performance

# Outline

# Presentations

- Vivian: Language Models Are Few-Shot Learners
- Nehha: LoRA: Low-Rank Adaptation of LLMs

# Next week

## 3. Background Research Presentation

Released on Tuesday 03/31/2026

### ✦ Presentation Schedule

**Date Groups**

3/31 1, 12, 2, 11, 3, 10

4/2 4, 9, 5, 8, 6, 7