Prompt Engineering and Efficient Adaptation

IE686 Large Language Models and Agents

UNIVERSITY OF MANNHEIM
Data and Web Science Group

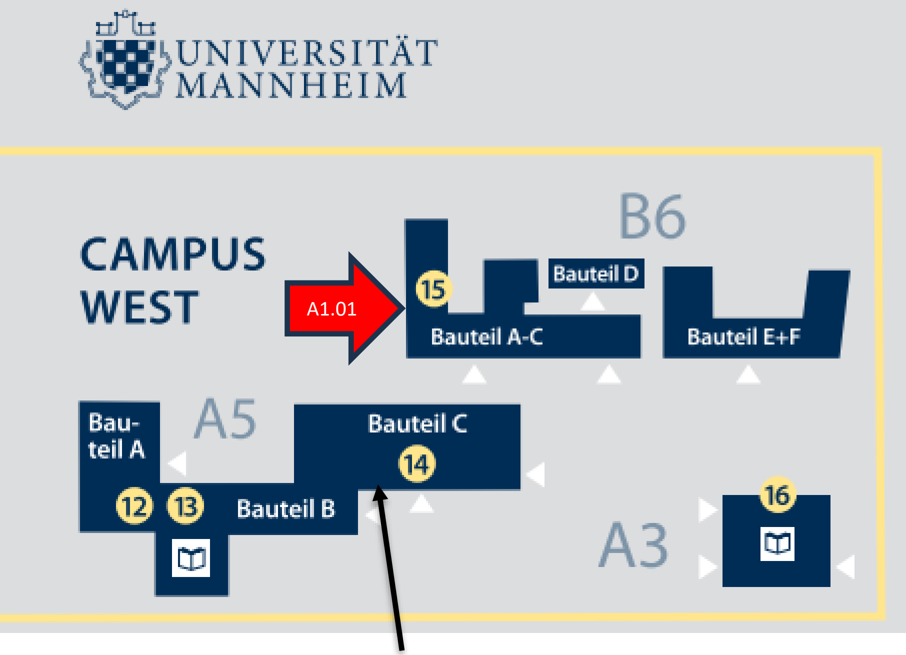
Credits

- This slide set is based on slides from
  - **Jiaxin Huang**
  - Mrinmaya Sachan
  - Diyi Yang
  - Tatsunori Hashimoto

- Many thanks to all of you!

Important: Next Weeks Session

- Next weeks session “Exercise: Introduction to LangChain” will take place on **Wednesday** 2nd of October
- Time: 17:15-18:45
- Room: B6 A1.01 (1st floor)
- Content: Learn to **apply** things we had in the lecture with LangChain

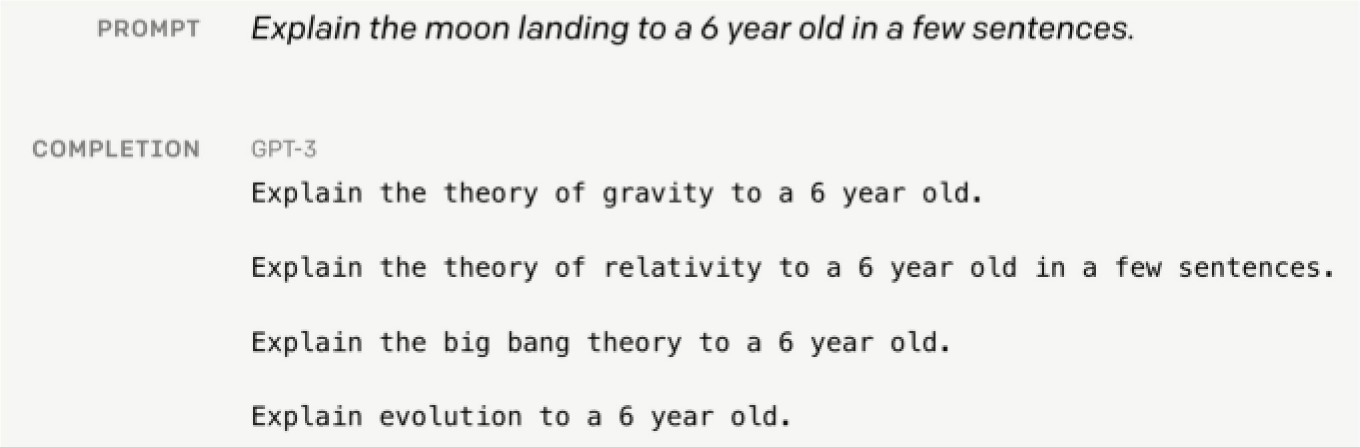


You are here

Outline

- **Recap: Instruction Tuning and RLHF**
- Prompt Engineering
  - Zero-shot Prompting
  - In-Context Learning
  - Chain-of-Thought Prompting
- Efficient Adaptation
  - Prompt-based Methods
  - Adapter-based Tuning
  - LoRa
- Evaluating Large Language Models
  - Types of Evaluation Methods for LLMs
  - Benchmarks

Language Modeling ≠ Solving Tasks



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

COMPLETION GPT-3

```
Explain the theory of gravity to a 6 year old.

Explain the theory of relativity to a 6 year old in a few sentences.

Explain the big bang theory to a 6 year old.

Explain evolution to a 6 year old.
```

- Language modelling with **next token prediction** does not make the model a competent task solver
- How to adapt to correctly solving tasks?

Ouyang, L et al., 2022. Training Language Models to follow Instructions with Human Feedback. *Advances in Neural Information Processing Systems*, *35*, pp.27730-27744.

Emergent Abilities of LLMs

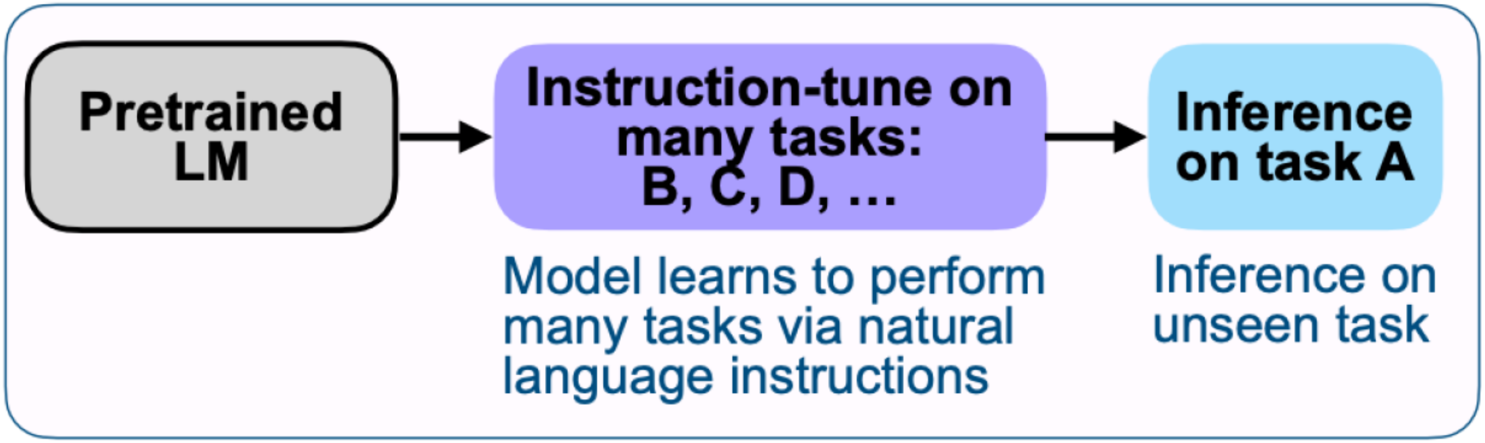
- "Abilities that are not present in small models but arise in large models"

J. Wei et al., "Emergent Abilities of Large Language Models," CoRR, vol. abs/2206.07682, 2022

- Three typical emergent abilities:
  - **In-context learning**: After providing the LLM with one or several task demonstrations in the prompt, it can generate the expected output (today)
  - **Instruction following**: Fine-tuning the model with instructions for various tasks at once, leads to strong performance on unseen tasks (last week)
  - **Step-by-step reasoning**: LLMs can perform complex tasks by breaking down a problem into smaller steps. The chain-of-thought prompting mechanism is a popular example (today)

Instruction Tuning

- Fine-tune on many tasks at once
- Teaches language model to follow different natural language instructions, so that it can perform well on downstream tasks and even **generalize** to unseen tasks


Pretrained LM
Instruction-tune on many tasks: B, C, D, …
Inference on task A
Model learns to perform many tasks via natural language instructions
Inference on unseen task

Reinforcement Learning from Human Feedback

- There is still a misalignment between the ML objective – maximizing the likelihood of a specific piece of human-written text – and what humans actually want – generation of high-quality outputs as determined by humans
- Language models go through another phase of learning, called **alignment**, where they learn how to present information to users and align to human preferences, e.g.:
  - Helpfulness
  - Honesty
  - Harmlessness

Outline

- Recap: Instruction Tuning and RLHF
- **Prompt Engineering**
  - Zero-shot Prompting
  - In-Context Learning
  - Chain-of-Thought Prompting
- Efficient Adaptation
  - Prompt-based Methods
  - Adapter-based Tuning
  - LoRa
- Evaluating Large Language Models
  - Types of Evaluation Methods for LLMs
  - Benchmarks

Prompting: Why even fine-tune?

- For many tasks, supervised fine-tuning data may not be available or may be costly to obtain
- Due to **emergent abilities** coupled with instruction tuning, we can simply prompt or instruct models to do a task!
- Prompts are written in **natural language**
- Prompting is **non-invasive**:
  - No additional parameters are introduced
  - No tuning of existing parameters
  - No need to inspect model’s embeddings

Prompt Engineering

- **Key idea**: Formulate a prompt that contains a description of a task and one or more specific task examples to be solved

- Elements of a prompt:
  - **Instruction**: a task or instruction you want the model to perform
  - **Context**: external information or additional context that can steer the model to better responses (optional)
  - **Input Data**: the input or question that we want to find an answer to
  - **Output Indicator**: type or format of the output (optional)

Use-case: Entity Matching

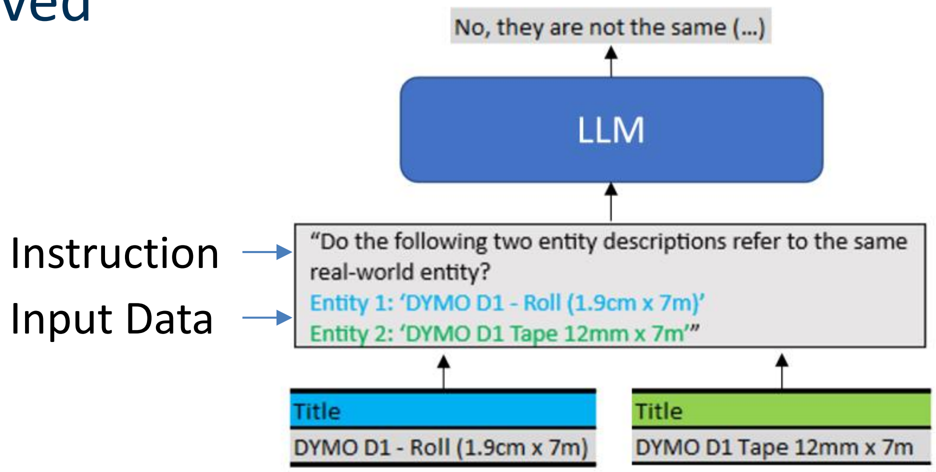
- **Goal:** Find all records that refer to the same real-world entity

Brand	Product	Model No.	RAM	Color	Release
Samsung	Galaxy	S21	64	Blue	2021/1/29
Samung	Gal.	S 21 TGB12	64 GB	blau	Feb. 2021
NULL	Galaxy S20 Blue TGB12 64GB	NULL	64000	NULL	2020/1/29

Vassilis, et al.: End-to-End Entity Resolution for Big Data. *ACM Surveys*, 2020.
Barlaug, et al.: Neural Networks for Entity Matching: A Survey. *TKDD*, 2021.

Zero-Shot Prompting

- The model gives an answer given only a natural language description of the task and the respective example to be solved



No, they are not the same (...)

"Do the following two entity descriptions refer to the same real-world entity?
Entity 1: 'DYMO D1 - Roll (1.9cm x 7m)'
Entity 2: 'DYMO D1 Tape 12mm x 7m'"

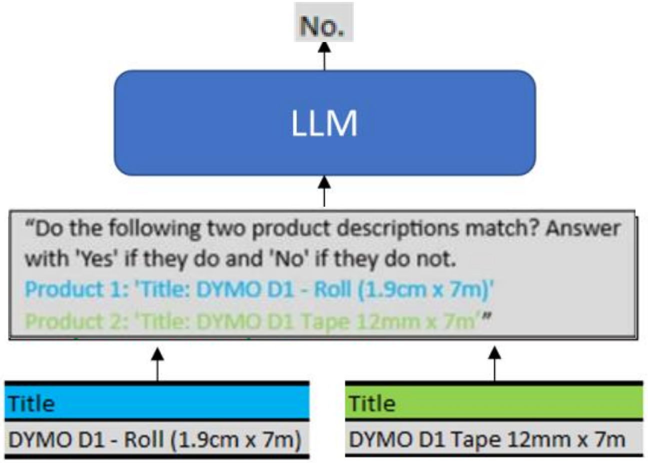
Title
DYMO D1 - Roll (1.9cm x 7m)

Title
DYMO D1 Tape 12mm x 7m

➔ This is what the purpose of the instruction tuning was

- Learn on many tasks
- Then be able to generalize to new tasks without parameter updates

Impact of Variations in the Formulation

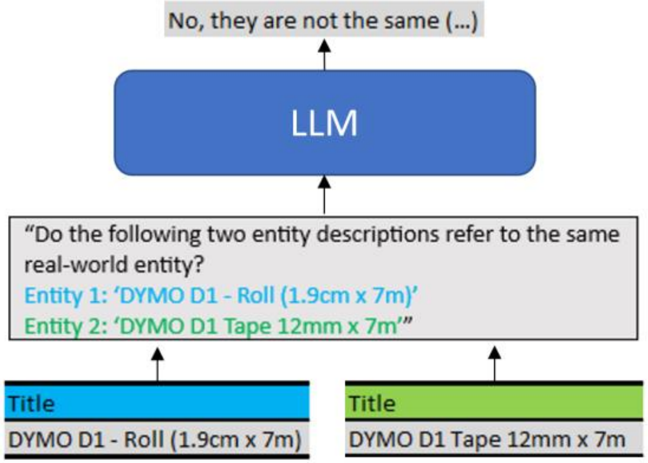

No.

LLM

“Do the following two product descriptions match? Answer with 'Yes' if they do and 'No' if they do not.
Product 1: 'Title: DYMO D1 - Roll (1.9cm x 7m)'
Product 2: 'Title: DYMO D1 Tape 12mm x 7m'”

Title
DYMO D1 - Roll (1.9cm x 7m)

Title
DYMO D1 Tape 12mm x 7m


No, they are not the same (…)

LLM

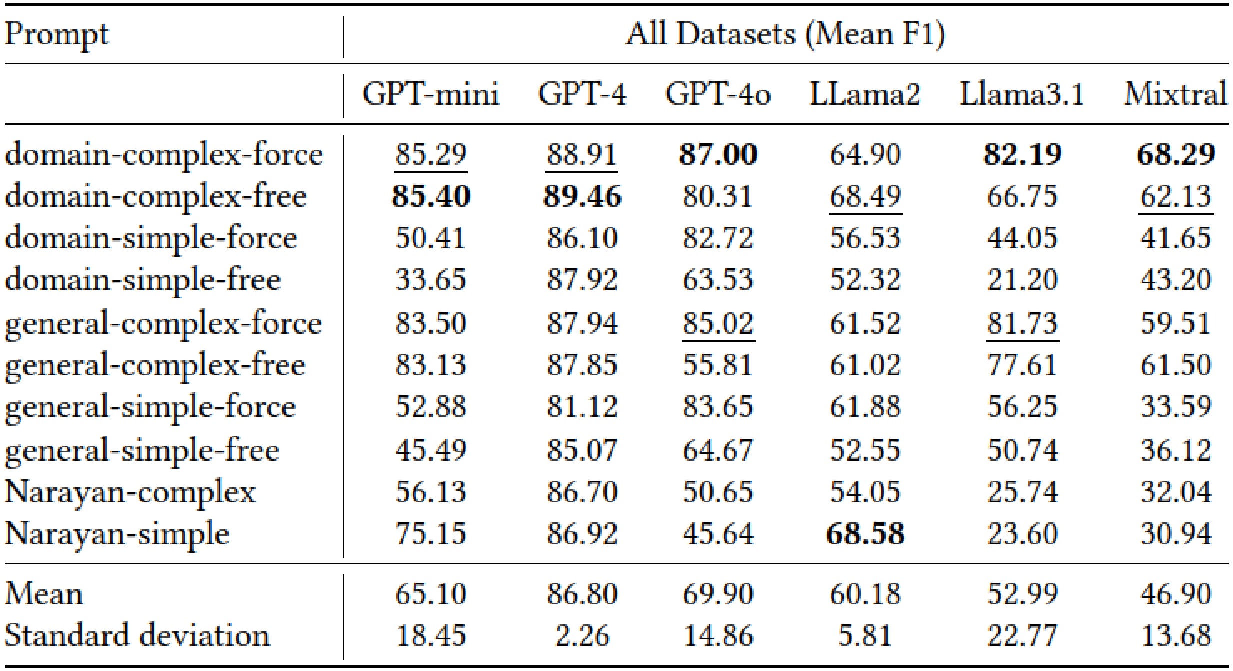
“Do the following two entity descriptions refer to the same real-world entity?
Entity 1: ‘DYMO D1 - Roll (1.9cm x 7m)’
Entity 2: ‘DYMO D1 Tape 12mm x 7m’”

Title
DYMO D1 - Roll (1.9cm x 7m)

Title
DYMO D1 Tape 12mm x 7m

- Variations:
  - **General** vs. **Domain-specific** wording
  - **Complex** vs. **simple** task description
  - **Free-form** vs. **restricted** answering

Impact of Variations in the Formulation



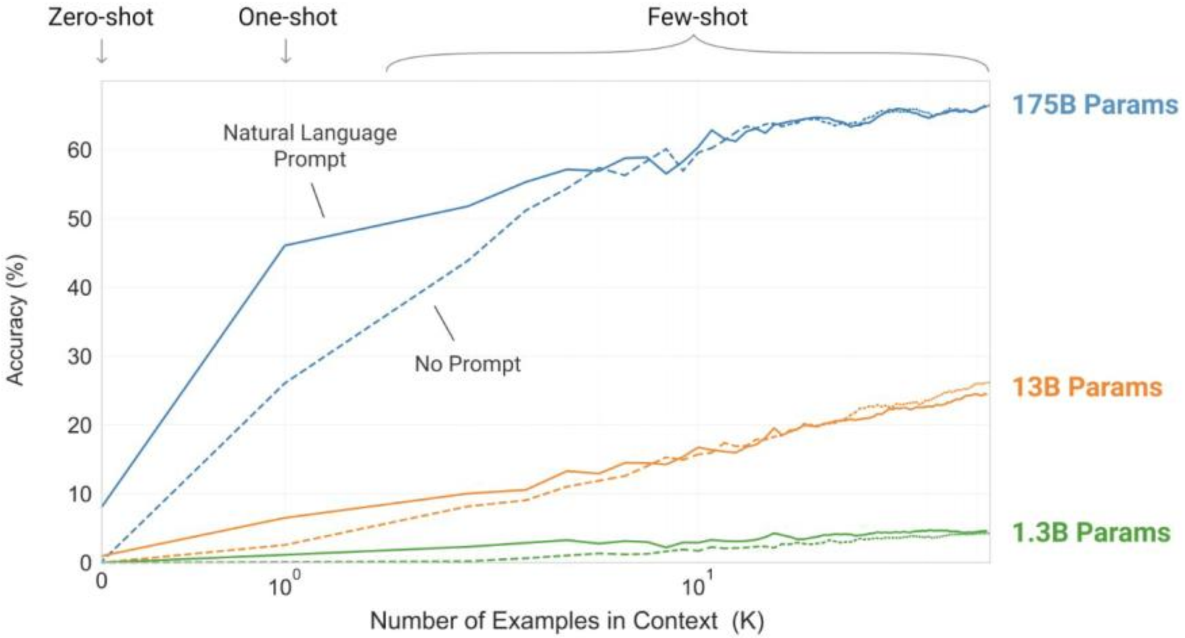
Prompt	All Datasets (Mean F1)					
	GPT-mini	GPT-4	GPT-4o	LLama2	Llama3.1	Mixtral
domain-complex-force	85.29	88.91	**87.00**	64.90	**82.19**	**68.29**
domain-complex-free	**85.40**	**89.46**	80.31	68.49	66.75	62.13
domain-simple-force	50.41	86.10	82.72	56.53	44.05	41.65
domain-simple-free	33.65	87.92	63.53	52.32	21.20	43.20
general-complex-force	83.50	87.94	85.02	61.52	81.73	59.51
general-complex-free	83.13	87.85	55.81	61.02	77.61	61.50
general-simple-force	52.88	81.12	83.65	61.88	56.25	33.59
general-simple-free	45.49	85.07	64.67	52.55	50.74	36.12
Narayan-complex	56.13	86.70	50.65	54.05	25.74	32.04
Narayan-simple	75.15	86.92	45.64	**68.58**	23.60	30.94
Mean	65.10	86.80	69.90	60.18	52.99	46.90
Standard deviation	18.45	2.26	14.86	5.81	22.77	13.68

- Strong impact of prompt formulation on performance for most models in zero-shot setting
- Even though the task is exactly the same!
- Often rigorous search necessary to find “good” formulation

In-Context Learning

- While often good, zero-shot learning performance may not be enough, especially on complex tasks 🙁
- The language model has been conditioned on following instructions and learning from examples during instruction tuning…
- **Idea**: Using the prompt, condition the LM using natural language instructions and adding one or more **solved task demonstrations**

In-context Learning and Scale



- In-context learning **generally** improves performance compared to zero-shot prompting
- Larger models **generally** make increasingly efficient use of in-context information

One-shot Prompting

- In addition to the task description, the model sees a single correct demonstration of the task

USER: Do the following two product descriptions match?
Product 1: ‘DYMO D1 19 mm x 7 m’
Product 2: ‘Dymo D1 (19mm x 7m – BoW)’
ASSISTANT: Yes.

USER: Do the following two product descriptions match?
Answer with 'Yes' if they do and 'No' if they do not.

Product 1: 'Title: DYMO D1 - Glossy tape - black on white - Roll (1.9cm x 7m) - 1 roll(s)’
Product 2: 'Title: DYMO 45017 D1 Tape 12mm x 7m sort p rd, S0720570'

ASSISTANT: No.

Few-shot Prompting

- In addition to the task description, the model sees multiple demonstrations of correctly performing the task

USER: Do the following two product descriptions match?
Product 1: ‘DYMO D1 19 mm x 7 m’
Product 2: ‘Dymo D1 (19mm x 7m – BoW)’
ASSISTANT: Yes.

USER: Do the following two product descriptions match?
Product 1: ‘DYMO D1 Tape 24mm’
Product 2: ‘Dymo D1 19mm x 7m’
ASSISTANT: No.

USER: Do the following two product descriptions match?
Answer with 'Yes' if they do and 'No' if they do not.

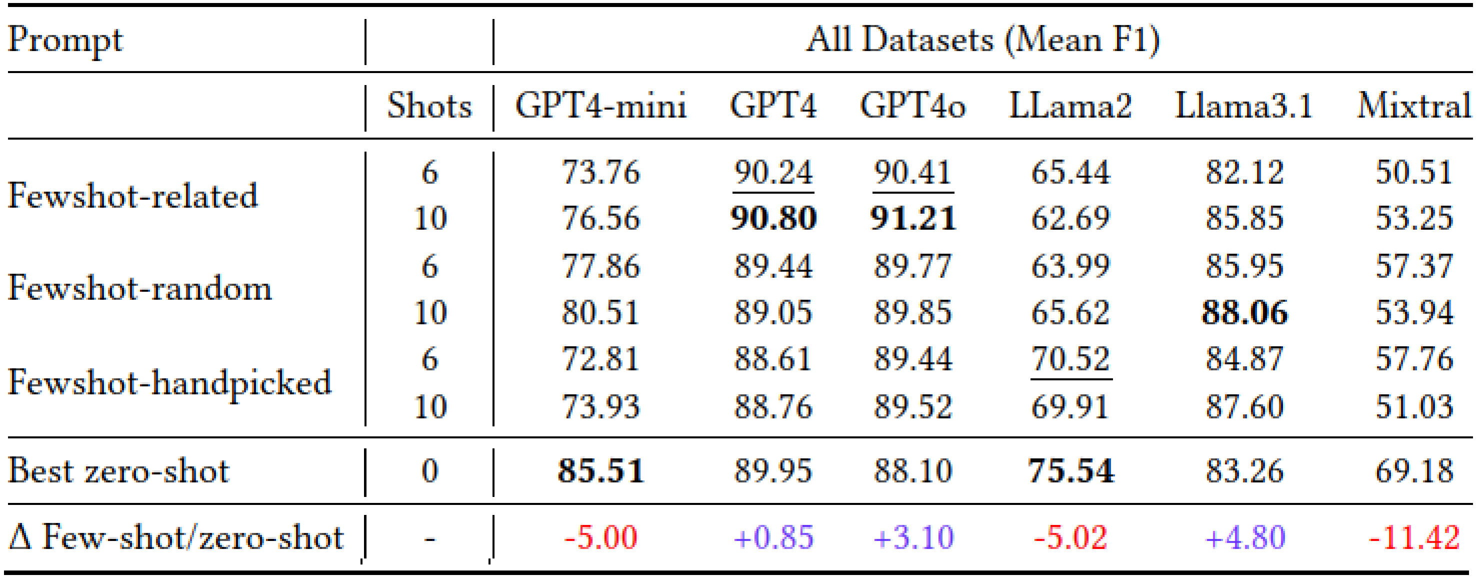
Product 1: 'Title: DYMO D1 - Glossy tape - black on white - Roll (1.9cm x 7m) - 1 roll(s)’
Product 2: 'Title: DYMO 45017 D1 Tape 12mm x 7m sort p rd, S0720570'

ASSISTANT: No.

Demonstration Selection Strategies

- How to select which demonstrations to give the model?
- Tried different methods in my experiments
  - Random: Randomly draw demonstrations
  - Similar: Select examples based on similarity to the actual task to be solved. In this case Generalized Jaccard (a string similarity metric)
  - Handpicked: Selection of a set of examples that a human domain expert would consider generally helpful for solving the task
- The *similar* and *handpicked* strategies both favor hard positive and negative demonstrations (the former those that are very similar to the task at hand)

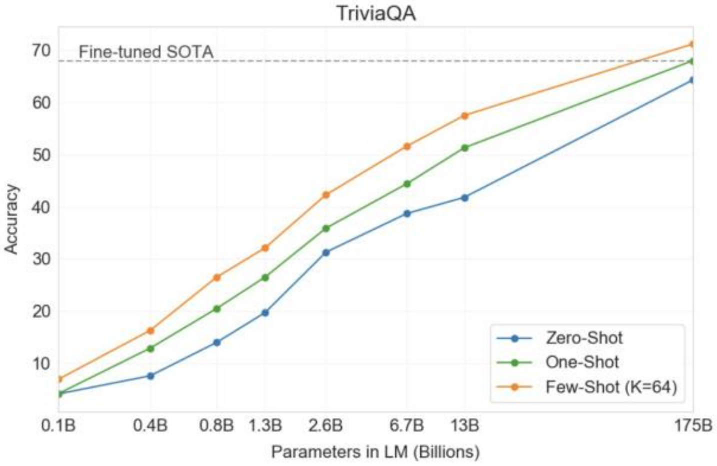
In-Context Learning Results

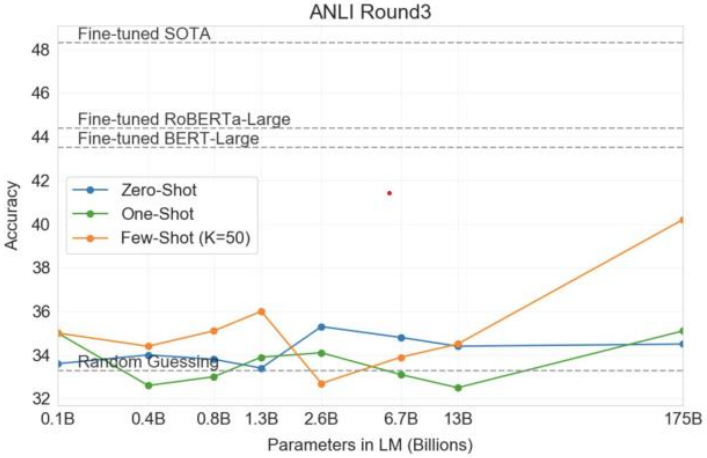


Prompt		All Datasets (Mean F1)					
	Shots	GPT4-mini	GPT4	GPT4o	LLama2	Llama3.1	Mixtral
Fewshot-related	6	73.76	90.24	90.41	65.44	82.12	50.51
	10	76.56	**90.80**	**91.21**	62.69	85.85	53.25
Fewshot-random	6	77.86	89.44	89.77	63.99	85.95	57.37
	10	80.51	89.05	89.85	65.62	**88.06**	53.94
Fewshot-handpicked	6	72.81	88.61	89.44	70.52	84.87	57.76
	10	73.93	88.76	89.52	69.91	87.60	51.03
Best zero-shot	0	**85.51**	89.95	88.10	**75.54**	83.26	69.18
Δ Few-shot/zero-shot	-	-5.00	+0.85	+3.10	-5.02	+4.80	-11.42

- In-context learning cannot be assumed to be always useful
- Favored selection strategy can also differ depending on model capabilities

In-context Learning on other Tasks





- Example: GPT-3
- On some tasks zero-shot/in-context learning can outperform the previous fine-tuned state-of-the-art
- On other tasks, it is not even close or can even be worse than zero-shot

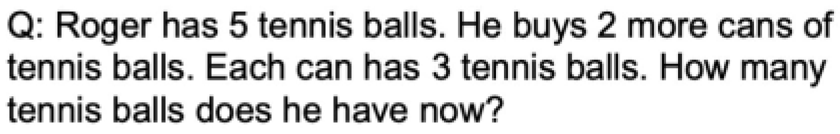
Why does In-context Learning work?

- The observed behavior is surprising as there is a **mismatch** between the LLMs training and the process of learning about tasks in-context
- Dai et al., 2022 show experimental results that support the idea that models produce meta-gradients during forward computation according to the demonstrations
  - These meta gradients are applied to the original LM through the attention mechanism
  - In-context learning can be seen as an implicit fine-tuning step on a few examples

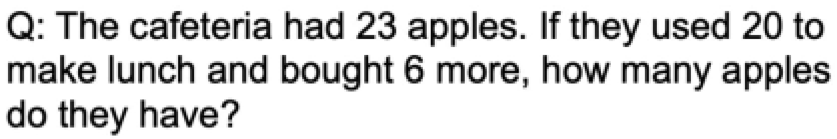
Dai, D., et al., 2023. Why Can GPT Learn In-Context? Language Models Secretly Perform Gradient Descent as Meta-Optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023* (pp. 4005-4019).

Limits of Prompting for Harder Tasks

- Some tasks seem too hard even for LLMs to learn through prompting alone. Especially tasks involving richer, multi-step reasoning
- Humans struggle at these tasks too!



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?



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

How to Solve these Tasks with Prompting?

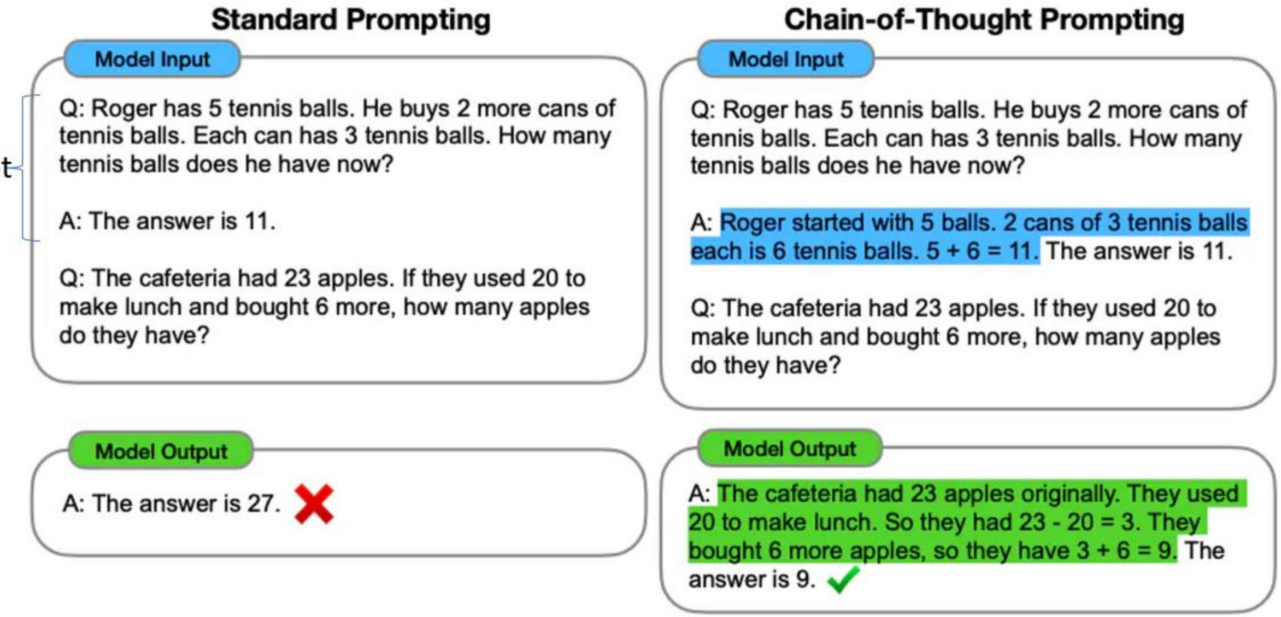
- How would you as a human solve such tasks?

➔ Divide and Conquer: Break problem down into smaller, easier-to-solve subtasks
  - Solve each task separately or consecutively
  - then find answer to the main task

- Popular LLM strategy: **Chain-of-thought** prompting

Chain-of-Thought Prompting

One-shot prompt



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

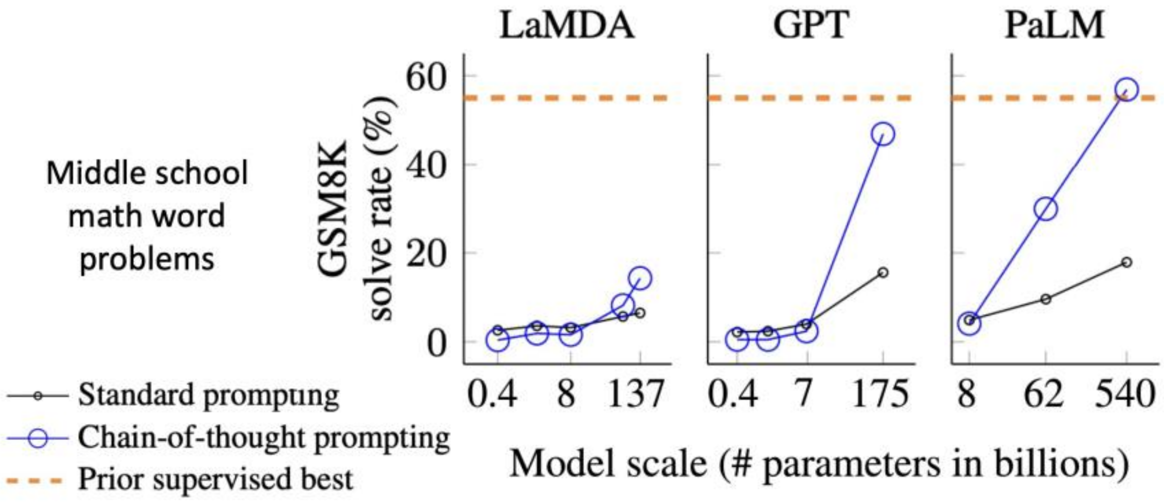
A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

Question (input)

Step-by-step rationale (highlight in blue), and answer (11)

Wei, J et al., 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems*, *35*, pp.24824-24837.

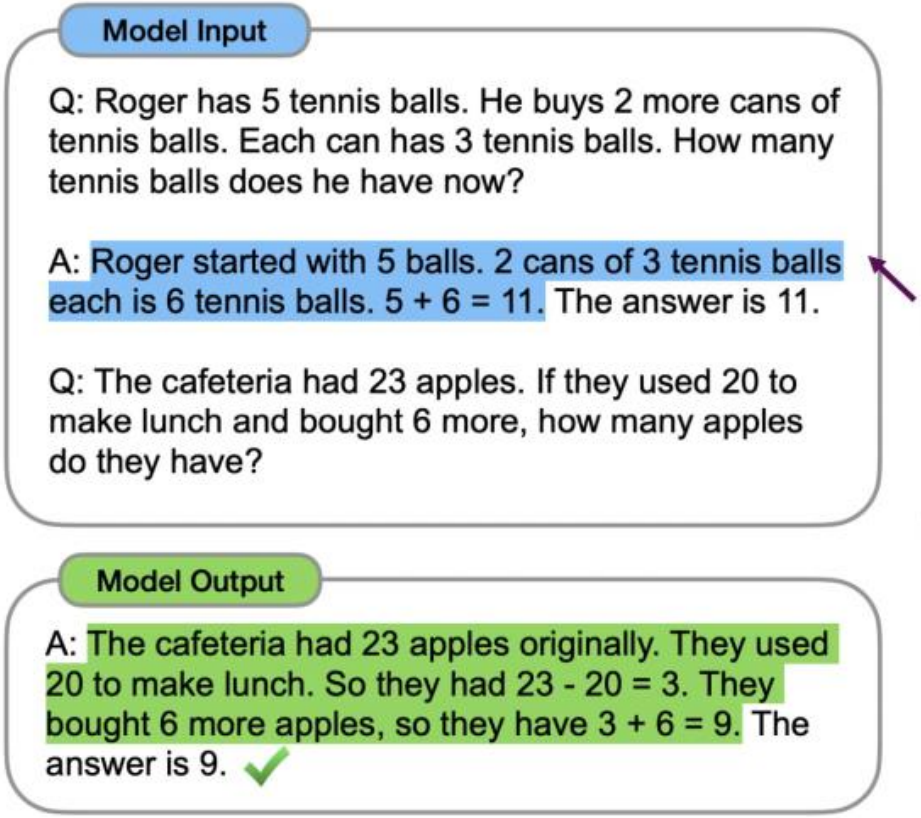
Chain-of-Thought Prompting


Middle school math word problems

- Chain-of-Thought prompting is an emergent property of model scale

Wei, J et al., 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems*, *35*, pp.24824-24837.

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?

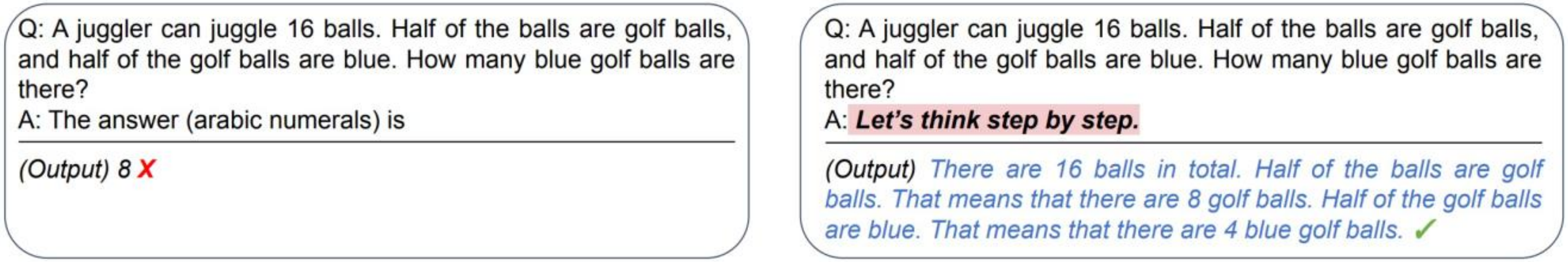
Model Output

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 CoT Prompting

- **Key Idea**: Elicit the model to produce a step-by-step solution of a problem by itself without demonstration.
- Simple, but it can work: add "**Let's think step-by-step**" to the prompt.

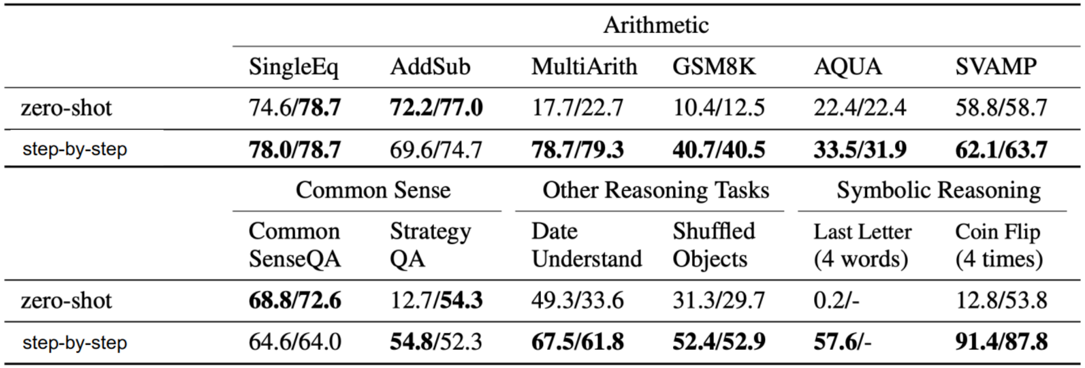


Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

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.***

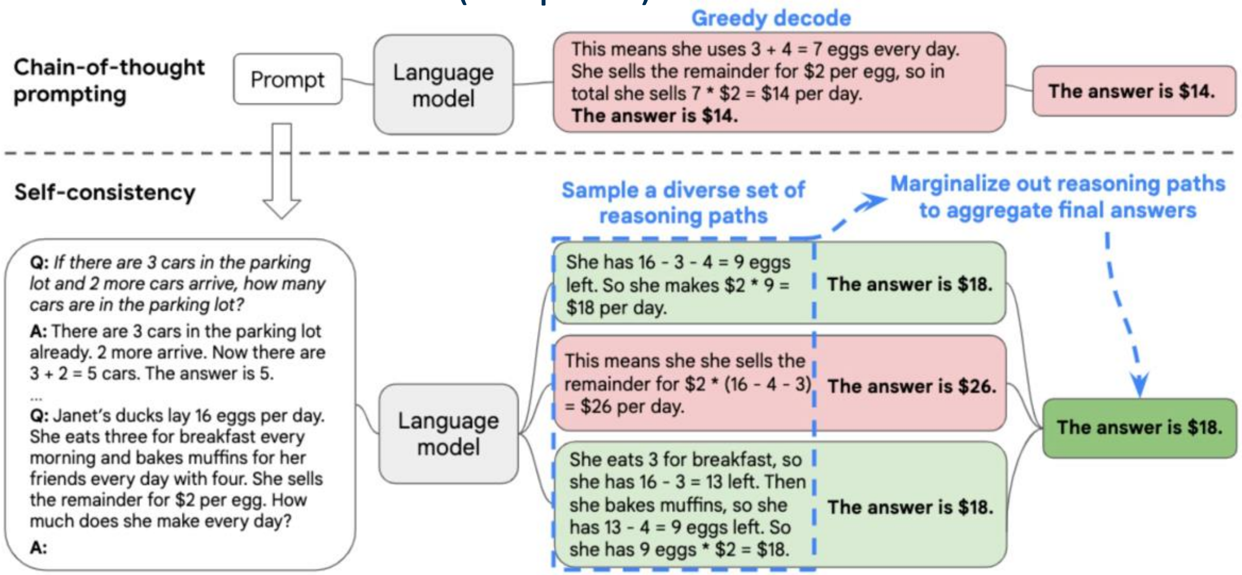
(Output) There are 16 balls in total. Half of the balls are golf balls. That means that there are 8 golf balls. Half of the golf balls are blue. That means that there are 4 blue golf balls. ✓



	Arithmetic					
	SingleEq	AddSub	MultiArith	GSM8K	AQUA	SVAMP
zero-shot	74.6/**78.7**	**72.2/77.0**	17.7/22.7	10.4/12.5	22.4/22.4	58.8/58.7
step-by-step	**78.0/78.7**	69.6/74.7	**78.7/79.3**	**40.7/40.5**	**33.5/31.9**	**62.1/63.7**
	Common Sense		Other Reasoning Tasks		Symbolic Reasoning	
	Common SenseQA	Strategy QA	Date Understand	Shuffled Objects	Last Letter (4 words)	Coin Flip (4 times)
zero-shot	**68.8/72.6**	12.7/**54.3**	49.3/33.6	31.3/29.7	0.2/-	12.8/53.8
step-by-step	64.6/64.0	**54.8**/52.3	**67.5/61.8**	**52.4/52.9**	**57.6/-**	**91.4/87.8**

A useful trick: Self-Consistency

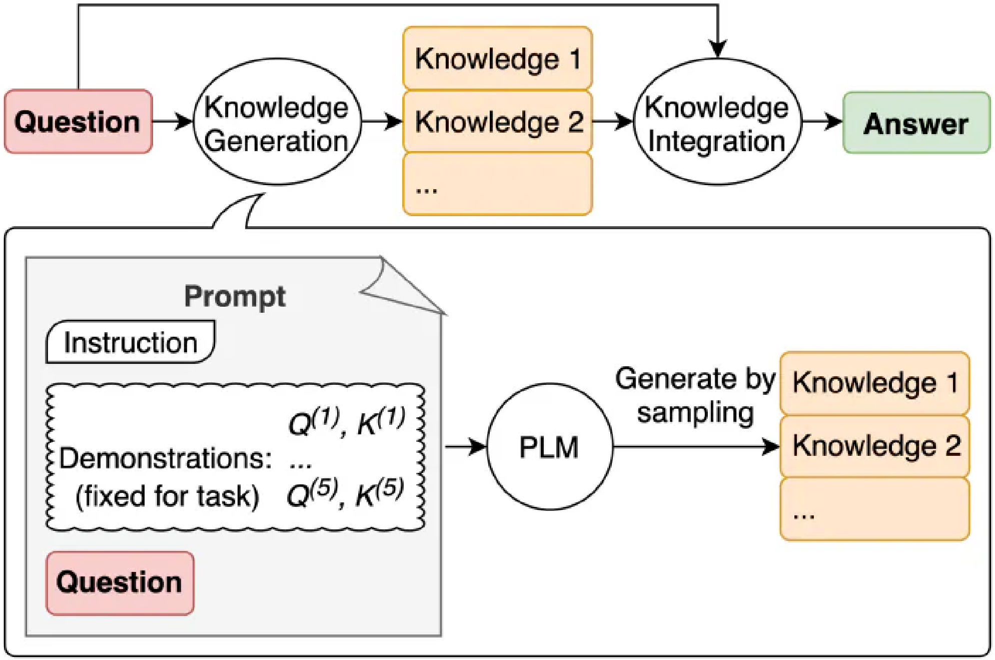
- To solve tasks as best as possible we generally use Greedy decoding (temperature = 0)
- Using self-consistency instead:
  - Generate a variety of outputs (with Temperature > 0)
  - Sample n reasoning paths/answers
  - Take the **most consistent** (frequent) answer as final result



Greedy decode
Chain-of-thought prompting
Prompt
Language model
This means she uses 3 + 4 = 7 eggs every day. She sells the remainder for $2 per egg, so in total she sells 7 * $2 = $14 per day. The answer is $14.
The answer is $14.
Self-consistency
Sample a diverse set of reasoning paths
Marginalize out reasoning paths to aggregate final answers
Q: If there are 3 cars in the parking lot and 2 more cars arrive, how many cars are in the parking lot?
A: There are 3 cars in the parking lot already. 2 more arrive. Now there are 3 + 2 = 5 cars. The answer is 5.
...
Q: Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder for $2 per egg. How much does she make every day?
A:
Language model
She has 16 - 3 - 4 = 9 eggs left. So she makes $2 * 9 = $18 per day.
The answer is $18.
This means she she sells the remainder for $2 * (16 - 4 - 3) = $26 per day.
The answer is $26.
She eats 3 for breakfast, so she has 16 - 3 = 13 left. Then she bakes muffins, so she has 13 - 4 = 9 eggs left. So she has 9 eggs * $2 = $18.
The answer is $18.
The answer is $18.

Generated Knowledge Prompting

- Example for an advanced prompting technique
- Use demonstrations to generate knowledge which is integrated into the prompt to solve the problem



Liu, J., et al., 2022. Generated Knowledge Prompting for Commonsense Reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 3154-3169)

Example: Rule Learning for Matching

USER: Generate product matching knowledge in the form of a list of rules from the following examples:

Matches:
Product 1: ‘DYMO D1 19 mm x 7 m’
Product 2: ‘Dymo D1 (19mm x 7m – BoW)’
…

Non-Matches:
Product 1: ‘DYMO D1 Tape 24mm’
Product 2: ‘Dymo D1 19mm x 7m’
…

ASSISTANT: Based on the provided examples, the following rules can be deduced:
1. The product titles match if they both refer to the same core product, which means they have the same primary brand and product model. For example, if two products both mention 'SANDISK EXTREME PRO SDHC 32GB 300MB/S UHS-II U3', they are considered as a match.
2. Order of words and usage of separators (like hyphen, comma, slash, parentheses) does not matter. For example, 'Dymo 53718 Black On Yellow - 24mm' and 'Dymo 24mm Black On Yellow D1 Tape (53718)' are considered as a match.

Refine Zero-shot Prompt

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Add the generated knowledge to the prompt and ask to solve the relevant examples

USER: The following rules regarding product features need to be observed:

1. The product titles match if they both refer to the same core product …

2. Order of words and usage of separators does not matter…

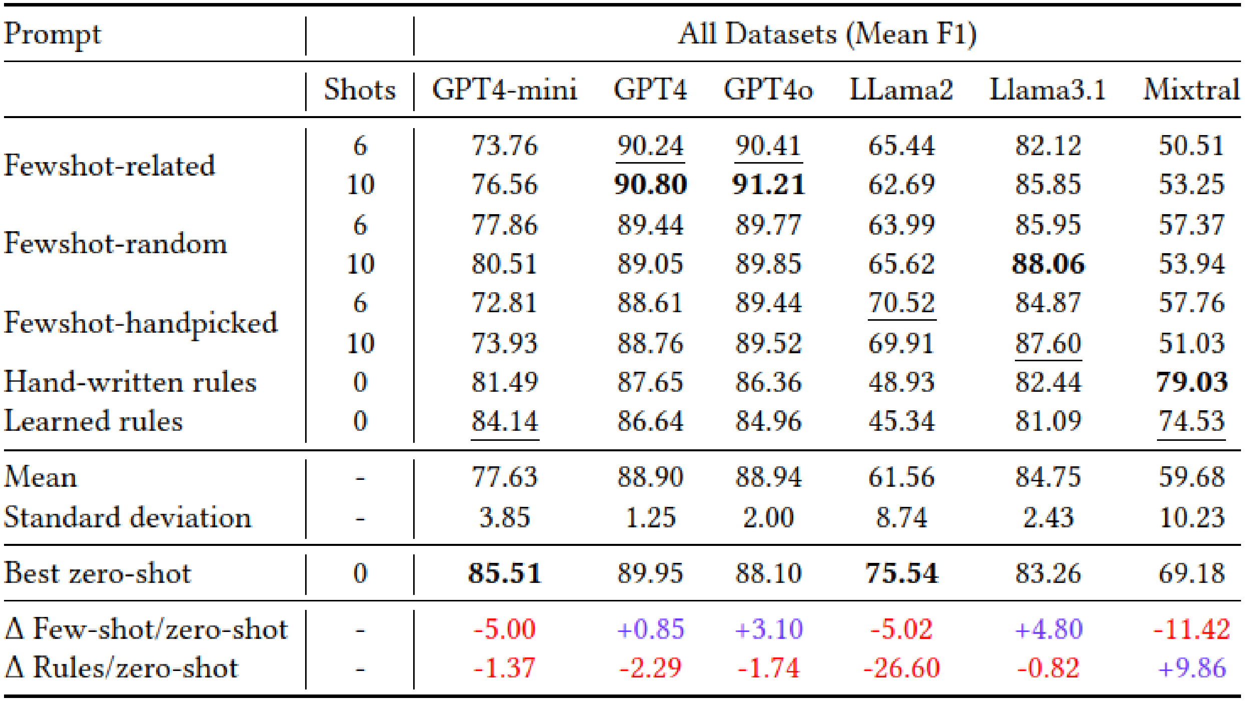
Do the following two product descriptions match? Answer with 'Yes' if they do and 'No' if they do not.

Product 1: 'DYMO D1 - Roll (1.9cm x 7m)'

Product 2: 'DYMO D1 Tape 12mm x 7m'

ASSISTANT: No.

Results for Knowledge Generation



Prompt		All Datasets (Mean F1)					
	Shots	GPT4-mini	GPT4	GPT4o	LLama2	Llama3.1	Mixtral
Fewshot-related	6	73.76	90.24	90.41	65.44	82.12	50.51
	10	76.56	**90.80**	**91.21**	62.69	85.85	53.25
Fewshot-random	6	77.86	89.44	89.77	63.99	85.95	57.37
	10	80.51	89.05	89.85	65.62	**88.06**	53.94
Fewshot-handpicked	6	72.81	88.61	89.44	70.52	84.87	57.76
	10	73.93	88.76	89.52	69.91	87.60	51.03
Hand-written rules	0	81.49	87.65	86.36	48.93	82.44	**79.03**
Learned rules	0	84.14	86.64	84.96	45.34	81.09	74.53
Mean	-	77.63	88.90	88.94	61.56	84.75	59.68
Standard deviation	-	3.85	1.25	2.00	8.74	2.43	10.23
Best zero-shot	0	**85.51**	89.95	88.10	**75.54**	83.26	69.18
Δ Few-shot/zero-shot	-	-5.00	+0.85	+3.10	-5.02	+4.80	-11.42
Δ Rules/zero-shot	-	-1.37	-2.29	-1.74	-26.60	-0.82	+9.86

- Works for some models and datasets
- In general, in-context learning performs better than learned rules

Summary: Prompt Engineering

- For many tasks, LLMs yield great performance even with simple zero-shot prompting (by framing task as natural language question)
- Emergent abilities allow to leverage the concepts of in-context learning and step-by-step reasoning to improve performance
- Self-consistency can further improve results by letting the model generate many answers and taking the majority vote
- Need to test what works and what harms, can differ wildly depending on task, dataset and selected LLM

Summary: Prompt Engineering

- In general, prompt engineering is non-trivial to get right as wording and ordering can have a large impact on performance
  - ➔ trial and error is important for finding good prompts
- Prompts need to be processed every time the model makes a decision, which can be costly if the prompts are long
- It is still unclear how the model learns from in-context demonstrations. There is work that has shown that randomly replacing labels in demonstrations still leads to nearly the same performance improvements.

Min, S., et al., 2022, December. Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing* (pp. 11048-11064).

Writing Prompts: Tips from OpenAI

1. Write clear instructions
   - Writing style, output format, give demonstrations,…
2. Split complex tasks into simpler subtasks
   - Summarize long documents by summarizing in chunks recursively
3. Give the model "time to think"
   - Chain-of-thought, ask the model to verify own answer again
4. Use external tools
   - For demonstration selection, code execution, external APIs
5. Test changes systematically
   - Ensure to have a good evaluation set, as changes may impact some few instances positively but result in overall worse performance

- See here for more details.

More Prompting Techniques

- Today many additional (semi-automatic) prompt engineering methods exist.
  - Tree-of-Thoughts
  - Least-to-Most Prompting
  - Program-of-Thought
  - Retrieval Augmented Generation and Tool Use (we will look at these in two weeks)

- See here for some methods and examples:
  - https://www.promptingguide.ai/
  - https://learnprompting.org/docs/introduction
  - https://cookbook.openai.com/

Outline

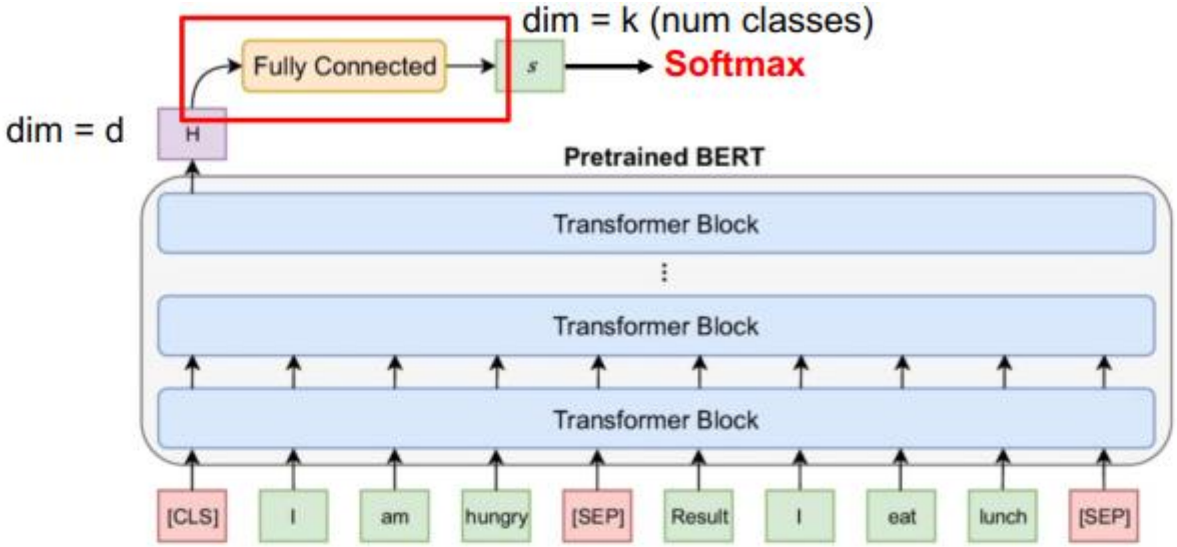
- Recap: Instruction Tuning and RLHF
- Prompt Engineering
  - Zero-shot Prompting
  - In-Context Learning
  - Chain-of-Thought Prompting
- **Efficient Adaptation**
  - Prompt-based Methods
  - Adapter-based Tuning
  - LoRa
- Evaluating Large Language Models
  - Types of Evaluation Methods for LLMs
  - Benchmarks

When Prompting is not enough…

- Assume we do not achieve the desired performance for our task with prompting
- But we do **have many labeled examples** for our task…
- What option do we have?
- What did we do before prompting came along?

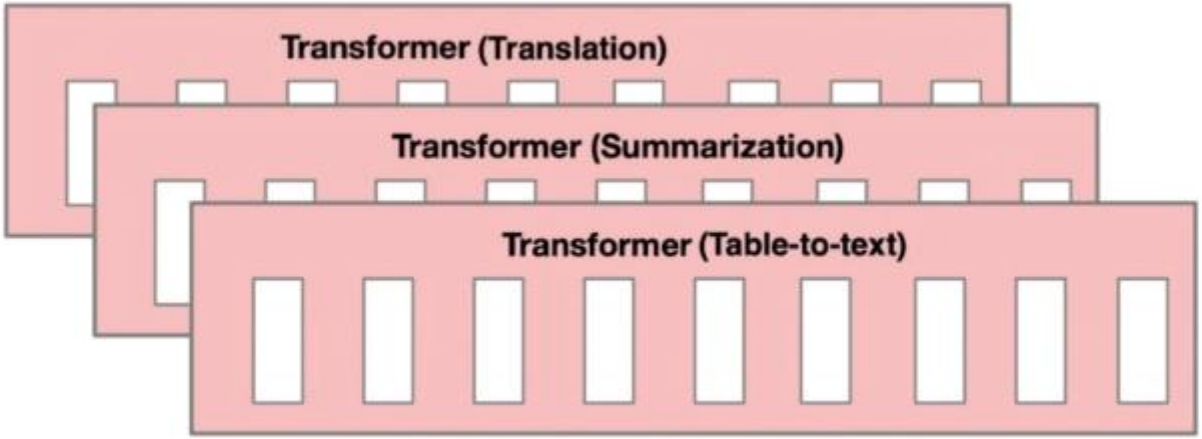
Background: Fine-tuning PLMs

- Attach a task-specific layer to the last layer of the pre-trained Transformer output
- Update the weights of all the parameters by backpropagating gradients on a downstream task
- **Expensive** due to lots of model updates during training



Problems with Fine-tuning

- Fine-tuning on small datasets may lead to **overfitting**
- **Catastrophic forgetting**: model may forget everything learned during pre-training, and become unable to generalize to other domains/tasks
- Need copy of model with different parameters for each task (maybe even for each user) ➔ **Memory inefficient**


Transformer (Translation)
Transformer (Summarization)
Transformer (Table-to-text)

Comparison Fine-tuning vs Prompting

- **Fine-tuning**
  - Can utilize more data
  - Leads to usually better performance with more training data
  - Computationally expensive to train the full neural network
  - Need to store a full set of model weights per task

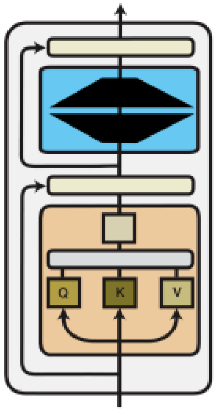
- **Prompting**
  - Training-data efficient
  - Computationally efficient
  - Performance depends on prompts and examples
  - Finding a good prompt can be challenging and requires lots of trial and error

Parameter-efficient Fine-tuning (PEFT)

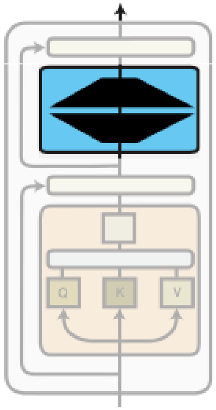
UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Fine-tuning all parameters is **impractical**, especially with LLMs
- **Solution**: Tune only parts of the parameters
- State-of-the-art models are massively overparameterized anyway

➔ Parameter-efficient fine-tuning can match performance of full fine-tuning



Full Fine-tuning
Update **all model parameters**



Parameter-efficient Fine-tuning
Update a **small subset** of model parameters

Recall Prompt Engineering

- Paraphrase the task instruction and add demonstrations
- For each task, perform a search over the sequence space (try around for what works) to find the prompt with the best output performance

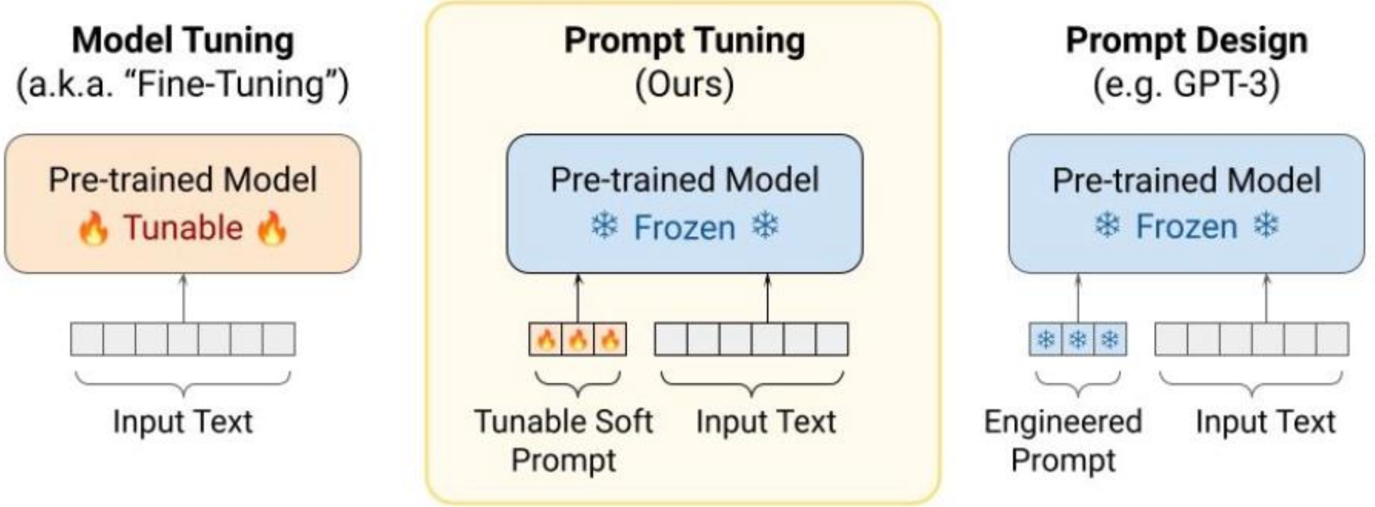
➔ Computationally expensive!

- Can we replace these prompts with a set of parameters and train them with labeled examples instead?

Prompt-Tuning

- Prepend a sequence of tokens as tunable embeddings to the input data (as soft prompts)
- Freeze all parameters of the Transformer model and only tune the prepended soft prompts

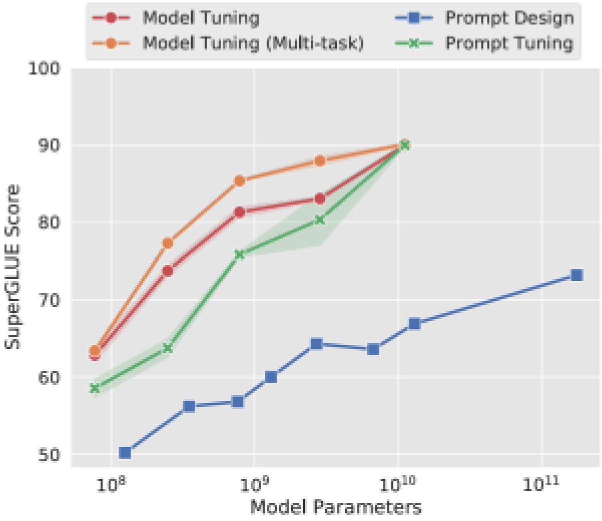
➔ Only a small set of parameters need to be stored for each task

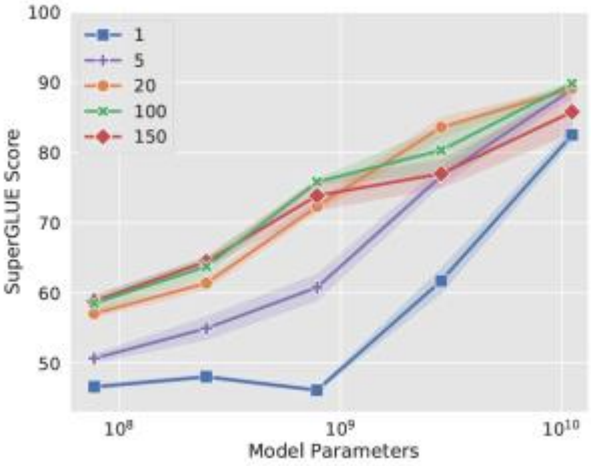


Lester, B., Al-Rfou, R. and Constant, N., 2021, November. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (pp. 3045-3059).

Power of Scale for Prompt Tuning

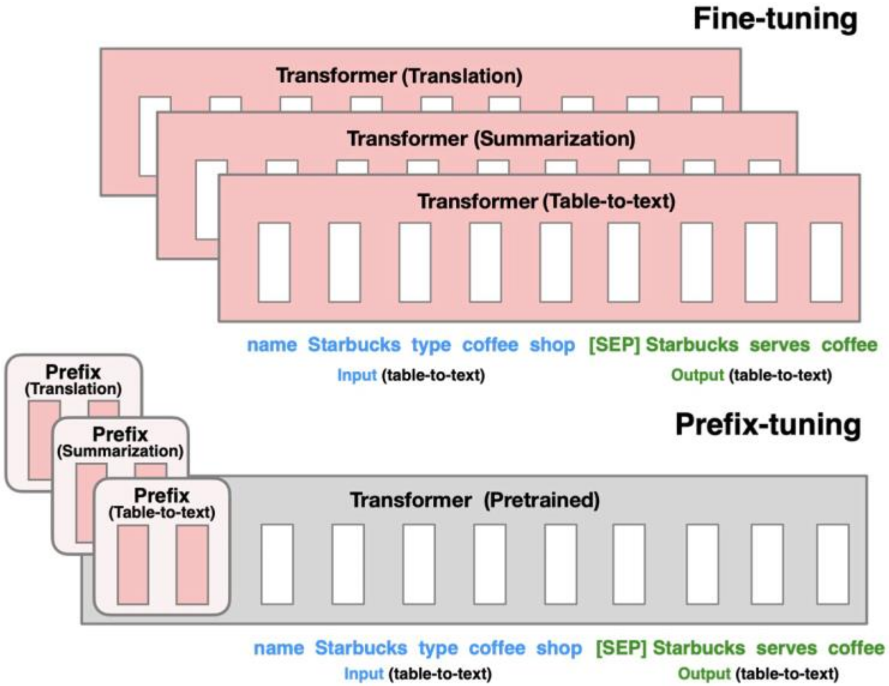
- Prompt-tuning becomes more effective when the model becomes larger
- Larger models perform well even with a small number of additional prompt tokens
- Initializing prompt tokens with real tokens of the vocabulary is helpful





Prefix-Tuning

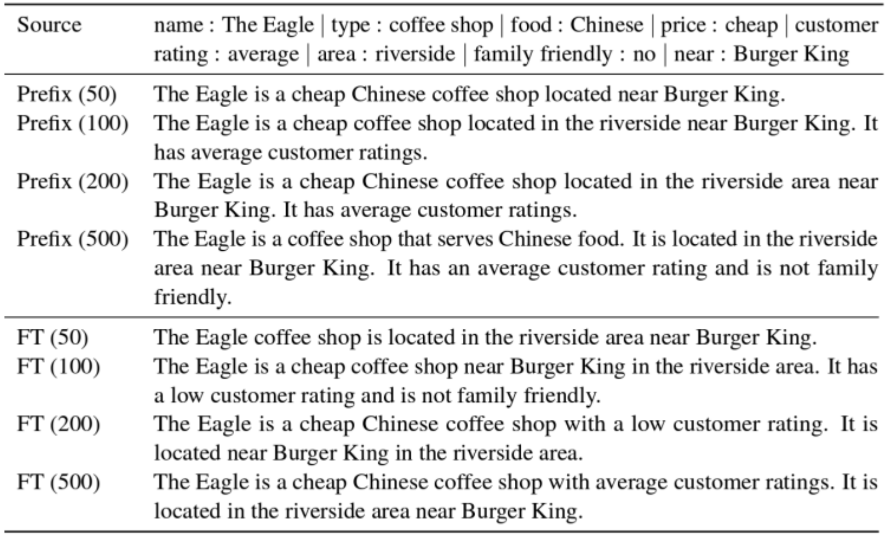
- Similar to prompt-tuning
- Soft prompt tokens are prepended to **each layer** of the transformer, not just the input layer


Fine-tuning
Transformer (Translation)
Transformer (Summarization)
Transformer (Table-to-text)
name Starbucks type coffee shop [SEP] Starbucks serves coffee
Input (table-to-text)
Output (table-to-text)
Prefix (Translation)
Prefix (Summarization)
Prefix (Table-to-text)
Prefix-tuning
Transformer (Pretrained)
name Starbucks type coffee shop [SEP] Starbucks serves coffee
Input (table-to-text)
Output (table-to-text)

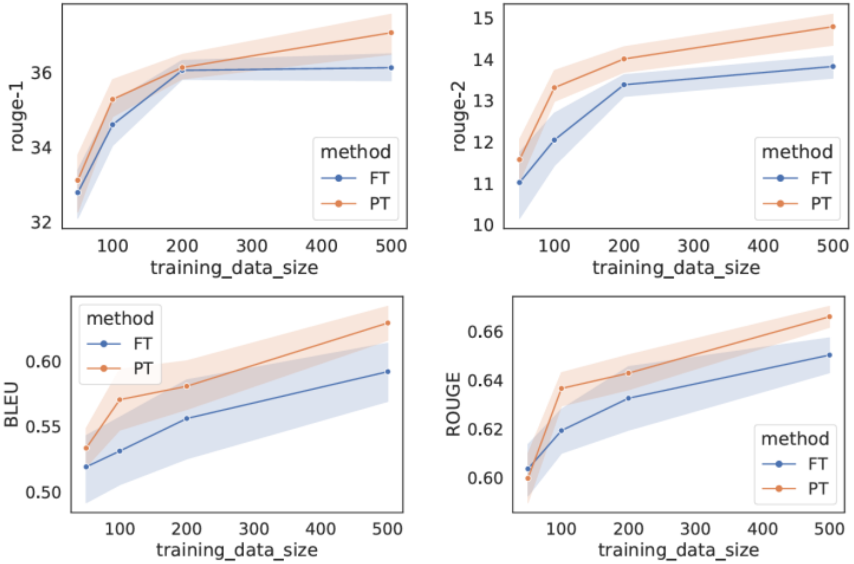
Li, X.L. et al., 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 4582-4597).

Example: Text Generation

- Prefix-tuning works better than fine-tuning in low-data settings
- Performance increases with increasing prefix length up to a certain point (depends on task)

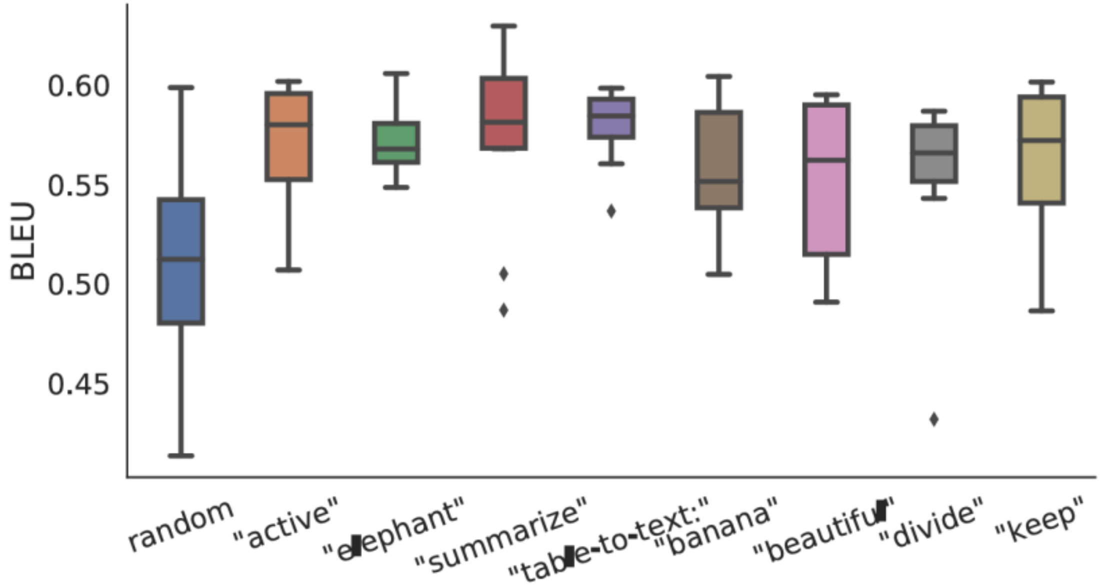


Source	name : The Eagle | type : coffee shop | food : Chinese | price : cheap | customer rating : average | area : riverside | family friendly : no | near : Burger King
Prefix (50)	The Eagle is a cheap Chinese coffee shop located near Burger King.
Prefix (100)	The Eagle is a cheap coffee shop located in the riverside near Burger King. It has average customer ratings.
Prefix (200)	The Eagle is a cheap Chinese coffee shop located in the riverside area near Burger King. It has average customer ratings.
Prefix (500)	The Eagle is a coffee shop that serves Chinese food. It is located in the riverside area near Burger King. It has an average customer rating and is not family friendly.
FT (50)	The Eagle coffee shop is located in the riverside area near Burger King.
FT (100)	The Eagle is a cheap coffee shop near Burger King in the riverside area. It has a low customer rating and is not family friendly.
FT (200)	The Eagle is a cheap Chinese coffee shop with a low customer rating. It is located near Burger King in the riverside area.
FT (500)	The Eagle is a cheap Chinese coffee shop with average customer ratings. It is located in the riverside area near Burger King.



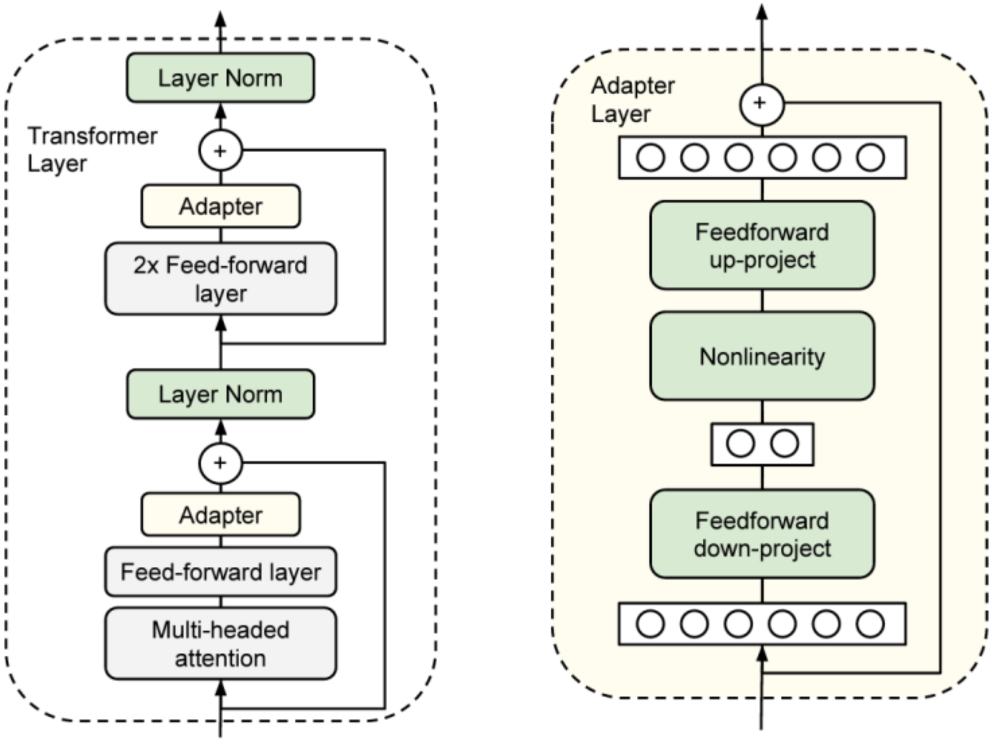
Prefix Initialization

- Random initialization leads to low performance with high variance
- Initializing with real words significantly improves generation
- Task-relevant words are better suited for initialization compared to random words



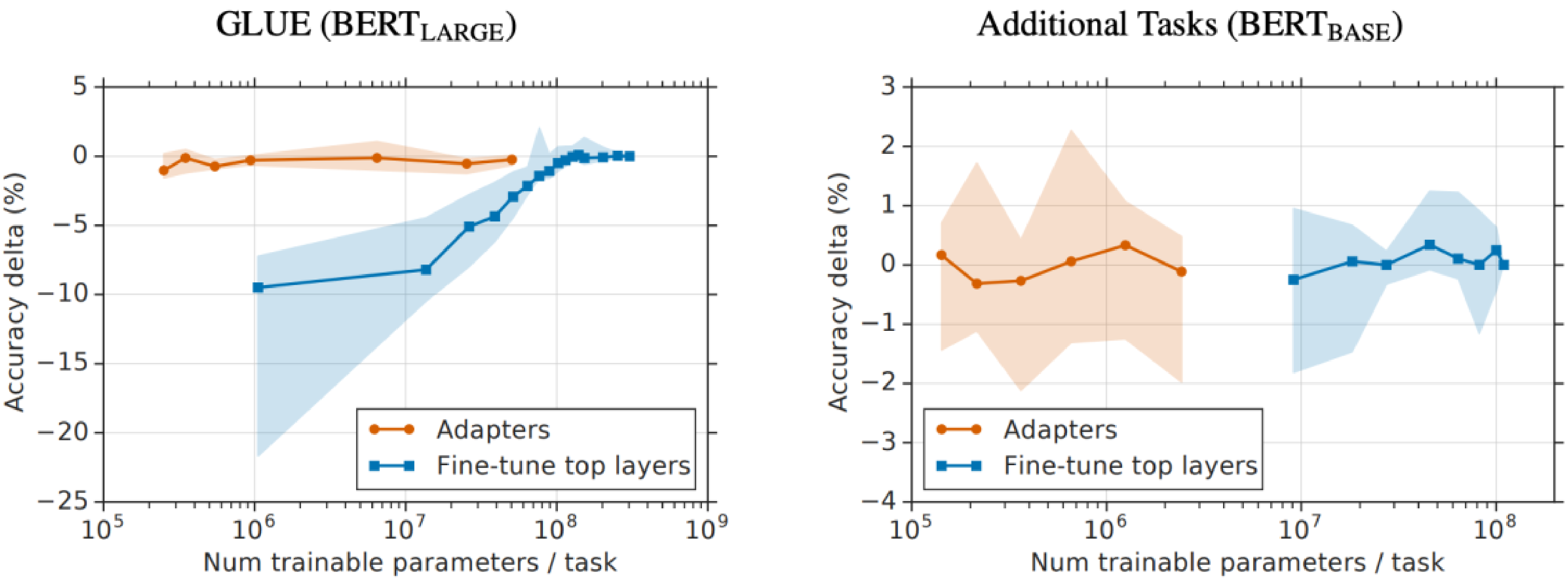
Adapter-based Fine-tuning

- Standard fine-tuning adds an additional layer to the top of the Transformer
- Adapter modules follow the same principle but add additional **smaller** layers to the original network
- During fine-tuning, the original parameters are **frozen** and only the adapter weights are updated



Comparison to Standard Fine-Tuning

- Adapter-based fine-tuning achieves a similar performance to full fine-tuning with orders of magnitude fewer trained parameters



Houlsby, N., et al., 2019. Parameter-efficient Transfer Learning for NLP. In *International Conference on Machine Learning* (pp. 2790-2799).

Adapter-based Fine-tuning

- Pros:
  - Empirically effective in multi-task settings
  - Computationally efficient compared to full fine-tuning

- Cons:
  - Adding in new layers makes the model slower during inference time
  - Makes the model size larger overall

LoRA: Low-Rank Adaptation

- For each downstream task, we learn a different set of parameters $\Delta\phi$
  - GPT-3 has a $|\phi|$ of 175 billion
  - $|\Delta\phi| = |\phi|$ for full fine-tuning
  - Expensive and memory inefficient!
- **LoRA key idea**: encode the task specific parameter increment $\Delta\phi = \Delta\phi(\Theta)$ by a smaller-sized set of parameters Θ, $|\Theta| \ll |\phi|$
- The task of finding $\Delta\phi$ becomes optimizing over Θ

$$\max_{\Theta} \sum_{(x,y)} \sum_{t=1}^{|y|} \log(P_{\phi_o+\Delta\phi(\Theta)}(y_t|x, y_{<t}))$$

Hu, E.J., et al., LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

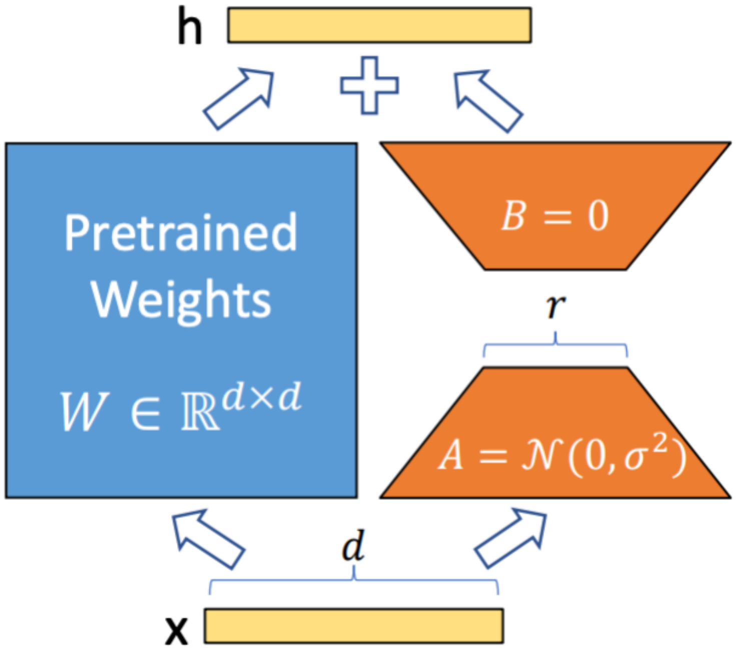
LoRA: Low-Rank Adaptation

- Updates to weights have low intrinsic rank during adaptation
- One matrix: $W_0 \in \mathbb{R}^{d \times k}$
- Constrain the update of a matrix with a low-rank decomposition

  $$W_0 + \Delta W = W_0 + BA$$

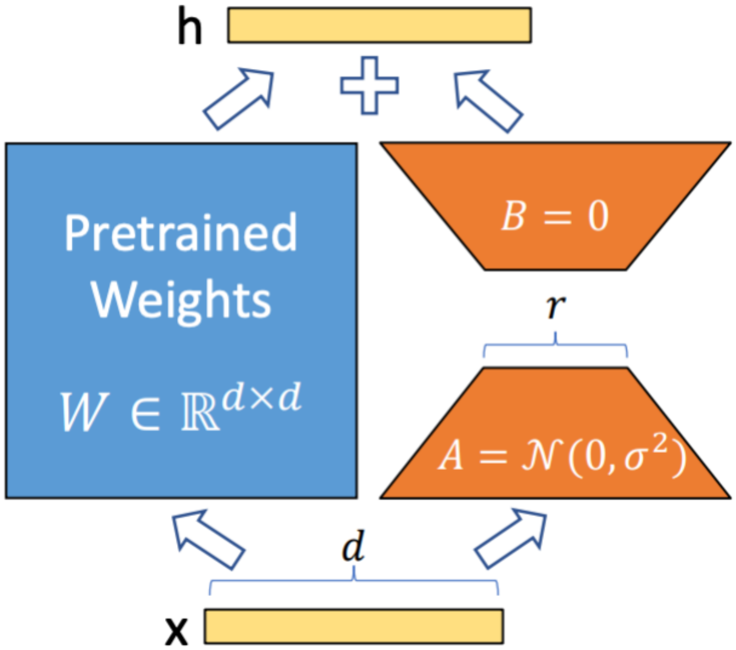
  where $B \in \mathbb{R}^{d \times r}, A \in \mathbb{R}^{r \times k}, r \ll \min(d, k)$

- Only A and B contain **trainable** parameters


h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

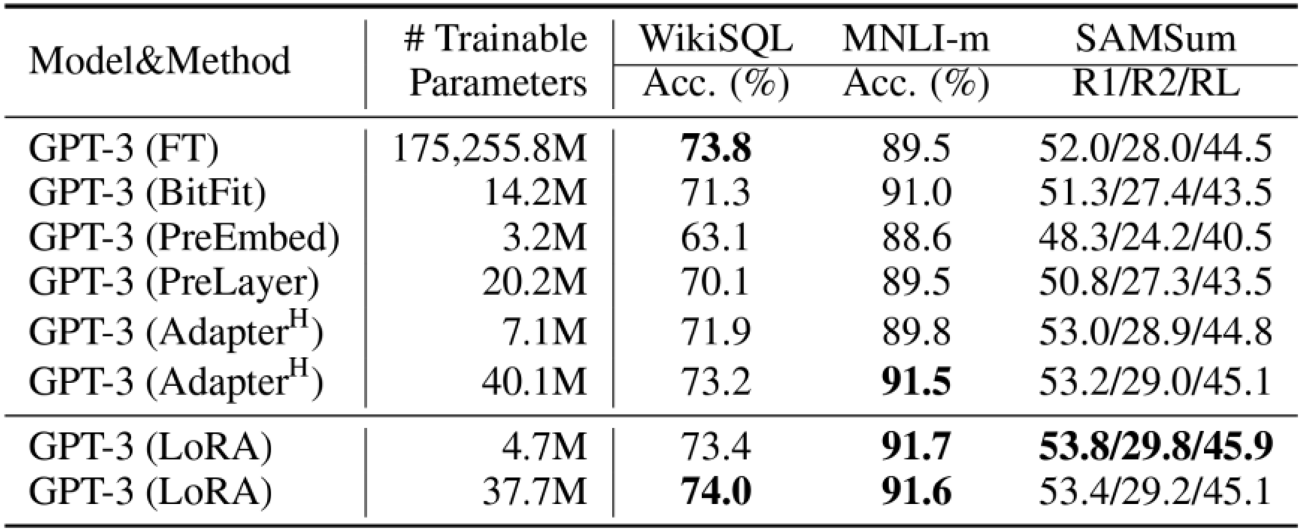
LoRA: Low-Rank Adaptation

- With increasing number of trainable parameters, the LoRA training converges to fully training the original model
- **No additional inference latency**: when switching to a different task, recover W by subtracting BA and adding a different $B'A'$
- LoRA is often applied to the weight matrices of the self-attention modules


h
Pretrained Weights
$W \in \mathbb{R}^{d \times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Applying LoRA to Transformers

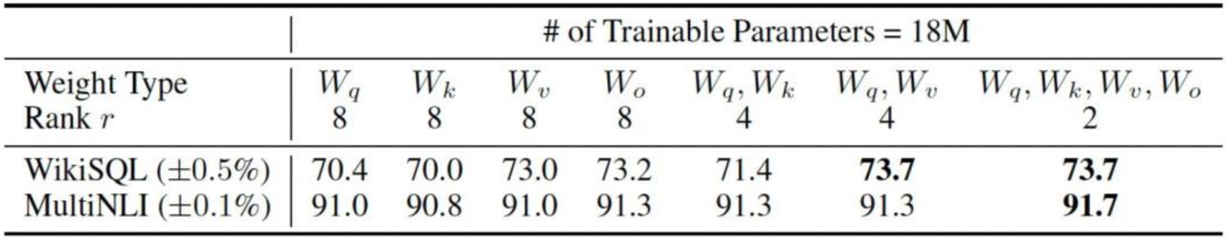
- In general, applicable to **any** deep learning weight matrix
- For GPT3-175B with r ranging from 2-64
  - VRAM: 1.2TB -> 350GB
  - Checkpoint storage: 350GB -> 35MB (**10,000 times smaller**)
- LoRA can outperform several baselines with comparable or fewer trainable parameters



Model&Method	# Trainable Parameters	WikiSQL	MNLI-m	SAMSum
		Acc. (%)	Acc. (%)	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 (Adapter^H)	7.1M	71.9	89.8	53.0/28.9/44.8
GPT-3 (Adapter^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

Applying LoRA to Transformers

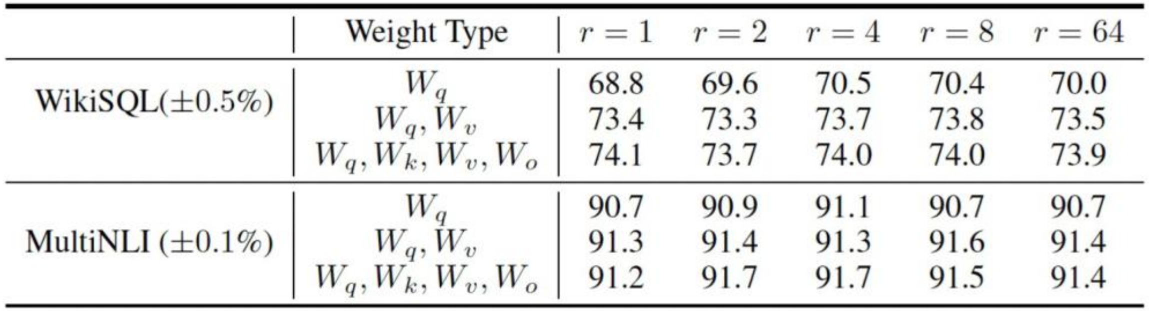
- Which weight matrices should we adapt?



	# of Trainable Parameters = 18M						
Weight Type Rank r	W_q 8	W_k 8	W_v 8	W_o 8	W_q, W_k 4	W_q, W_v 4	W_q, W_k, W_v, W_o 2
WikiSQL (±0.5%)	70.4	70.0	73.0	73.2	71.4	**73.7**	**73.7**
MultiNLI (±0.1%)	91.0	90.8	91.0	91.3	91.3	91.3	**91.7**

Adapting both Wq and Wv gives the best performance overall.

- What is the optimal rank *r* for LoRA

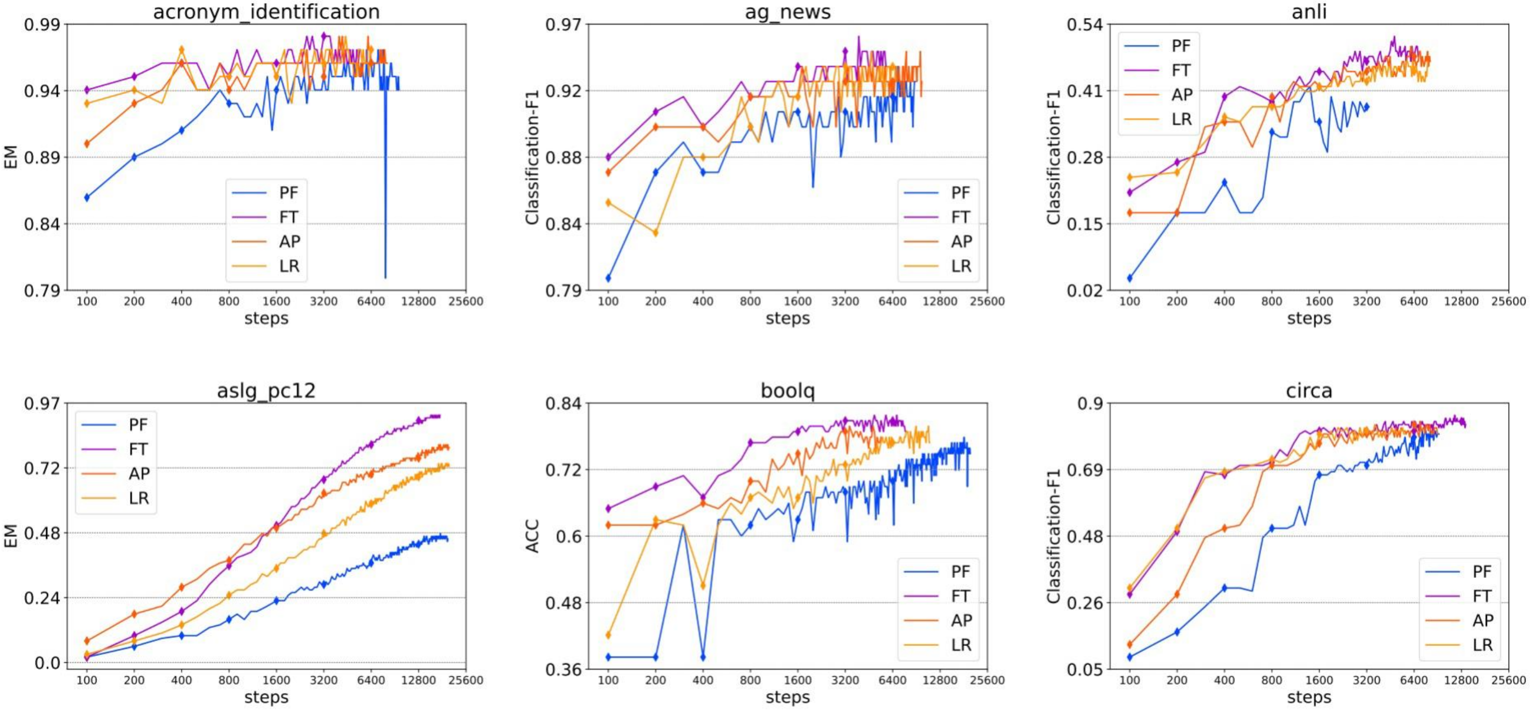


	Weight Type	$r = 1$	$r = 2$	$r = 4$	$r = 8$	$r = 64$
WikiSQL(±0.5%)	W_q	68.8	69.6	70.5	70.4	70.0
	W_q, W_v	73.4	73.3	73.7	73.8	73.5
	W_q, W_k, W_v, W_o	74.1	73.7	74.0	74.0	73.9
MultiNLI (±0.1%)	W_q	90.7	90.9	91.1	90.7	90.7
	W_q, W_v	91.3	91.4	91.3	91.6	91.4
	W_q, W_k, W_v, W_o	91.2	91.7	91.7	91.5	91.4

LoRA already performs competitively with a very small r

Comparison of Fine-tuning Methods

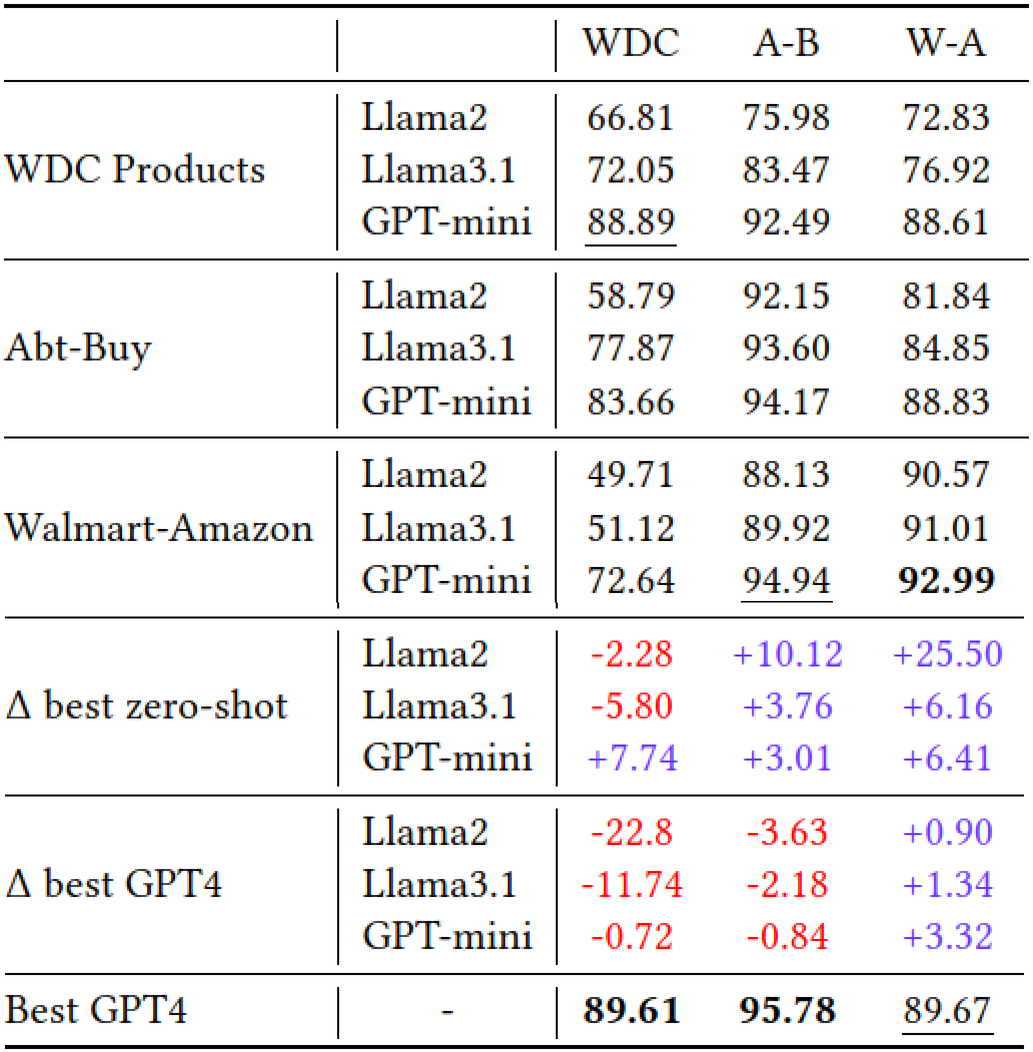
- Given enough data and computing resources
- Overall performance on T5-base: Full fine-tuning > LoRA > Adapters > Prefix Tuning > Prompt Tuning



Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., Hu, S., Chen, Y., Chan, C.M., Chen, W. and Yi, J., 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*.

Example: LoRA Fine-tuning for EM

- Mostly leads to strong improvements over zero-shot (F1-score)
- Small models and open-source Llamas (70B) nearly reach the same performance as the best non fine-tuned GPT4 model



		WDC	A-B	W-A
WDC Products	Llama2	66.81	75.98	72.83
	Llama3.1	72.05	83.47	76.92
	GPT-mini	88.89	92.49	88.61
Abt-Buy	Llama2	58.79	92.15	81.84
	Llama3.1	77.87	93.60	84.85
	GPT-mini	83.66	94.17	88.83
Walmart-Amazon	Llama2	49.71	88.13	90.57
	Llama3.1	51.12	89.92	91.01
	GPT-mini	72.64	94.94	**92.99**
Δ best zero-shot	Llama2	-2.28	+10.12	+25.50
	Llama3.1	-5.80	+3.76	+6.16
	GPT-mini	+7.74	+3.01	+6.41
Δ best GPT4	Llama2	-22.8	-3.63	+0.90
	Llama3.1	-11.74	-2.18	+1.34
	GPT-mini	-0.72	-0.84	+3.32
Best GPT4	-	**89.61**	**95.78**	89.67

Summary: Efficient Adaptation

- Efficient methods for fine-tuning LLMs on specific tasks usually lead to strong improvements over prompting methods
- But they require large amounts of labeled examples which are often expensive to obtain
- A fine-tuned smaller LLM can achieve a similar performance to GPT4 for the entity matching example task

Outline

- Recap: Instruction Tuning and RLHF
- Prompt Engineering
  - Zero-shot Prompting
  - In-Context Learning
  - Chain-of-Thought Prompting
- Efficient Adaptation
  - Prompt-based Methods
  - Adapter-based Tuning
  - LoRa
- **Evaluating Large Language Models**
  - Types of Evaluation Methods for LLMs
  - Benchmarks

Evaluating Large Language Models

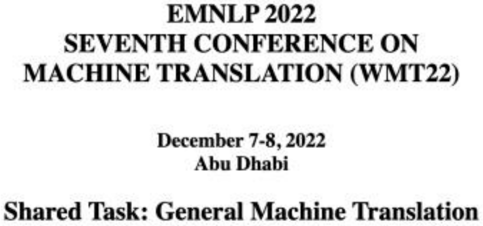
- Benchmarks and how we evaluate drive the progress of the research fields

Linguistic Data Consortium | UNIVERSITY
LDC
Linguistic Data Consortium
ABOUT
MEMBERS
COMMUNICATIONS
LANGUAGE RESOURCES
Data
Obtaining Data
Catalog
Home › Language Resources › Data

Switchboard-1 Release 2

Item Name: Switchboard-1 Release 2

Author(s): John J. Godfrey, Edward Holliman



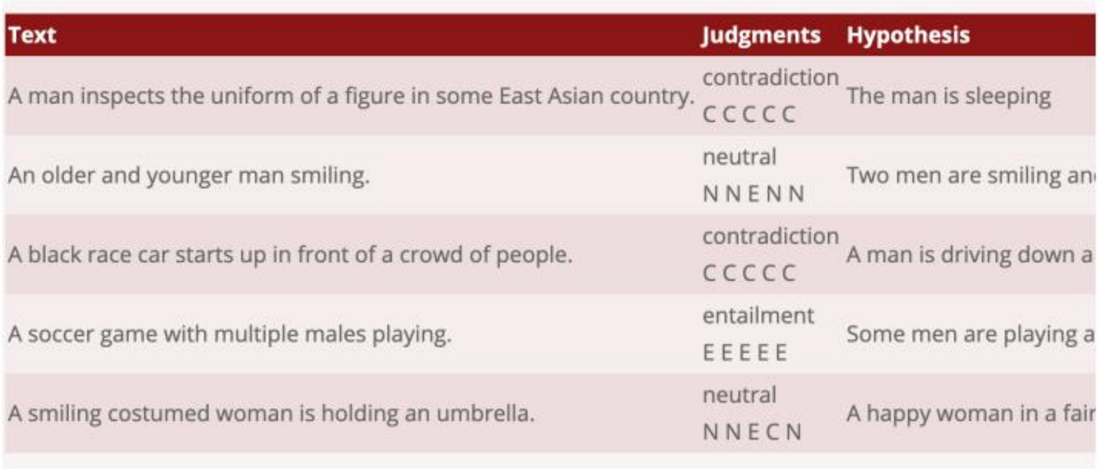
EMNLP 2022
SEVENTH CONFERENCE ON
MACHINE TRANSLATION (WMT22)

December 7-8, 2022
Abu Dhabi

Shared Task: General Machine Translation

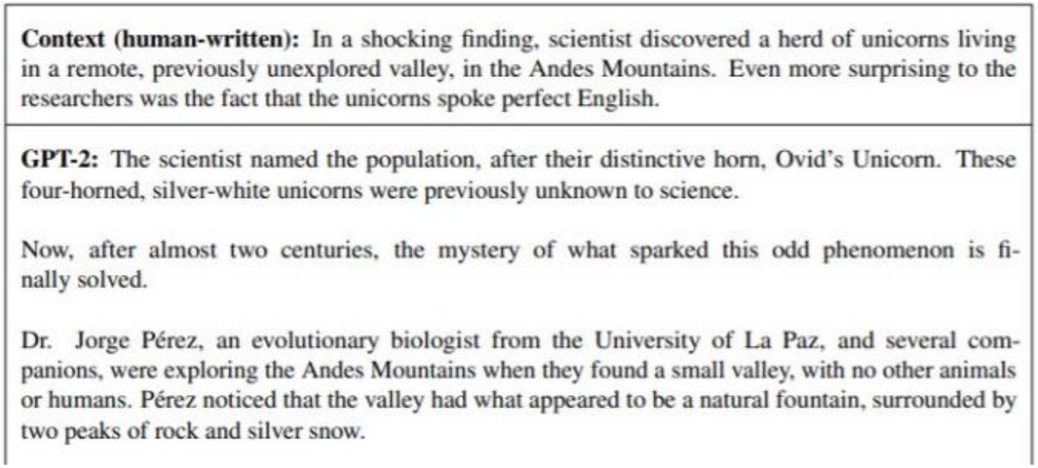
Two Major Types of Evaluations

Close-ended evaluations



Text	Judgments	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction C C C C C	The man is sleeping
An older and younger man smiling.	neutral N N E N N	Two men are smiling an
A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fair

Open ended evaluations



Context (human-written): In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

GPT-2: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

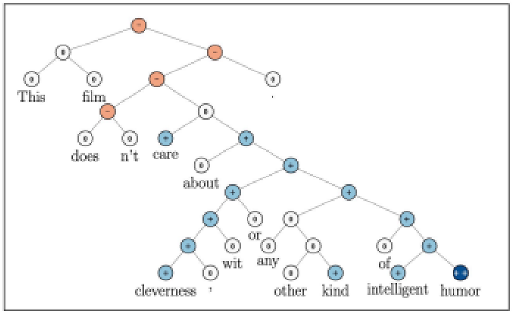
Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

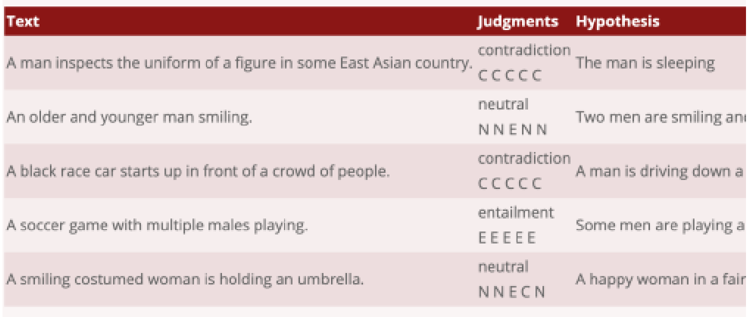
Close-ended Benchmarks

- Many NLP tasks are “close-ended”
  - Limited number of potential answers
  - Often one or just a few correct answers
- Examples:
  - Sentiment classification
  - Extractive Question Answering (extract part of document containing the answer)
- Enables **automatic evaluation**
- Similar to machine learning evaluations we know from e.g. Data Mining I

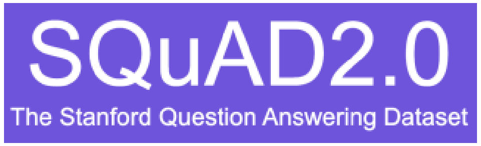
Single Task Benchmarks

- Measure **specific** language capabilities





Text	Judgments	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction C C C C C	The man is sleeping
An older and younger man smiling.	neutral N N E N N	Two men are smiling an
A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fair



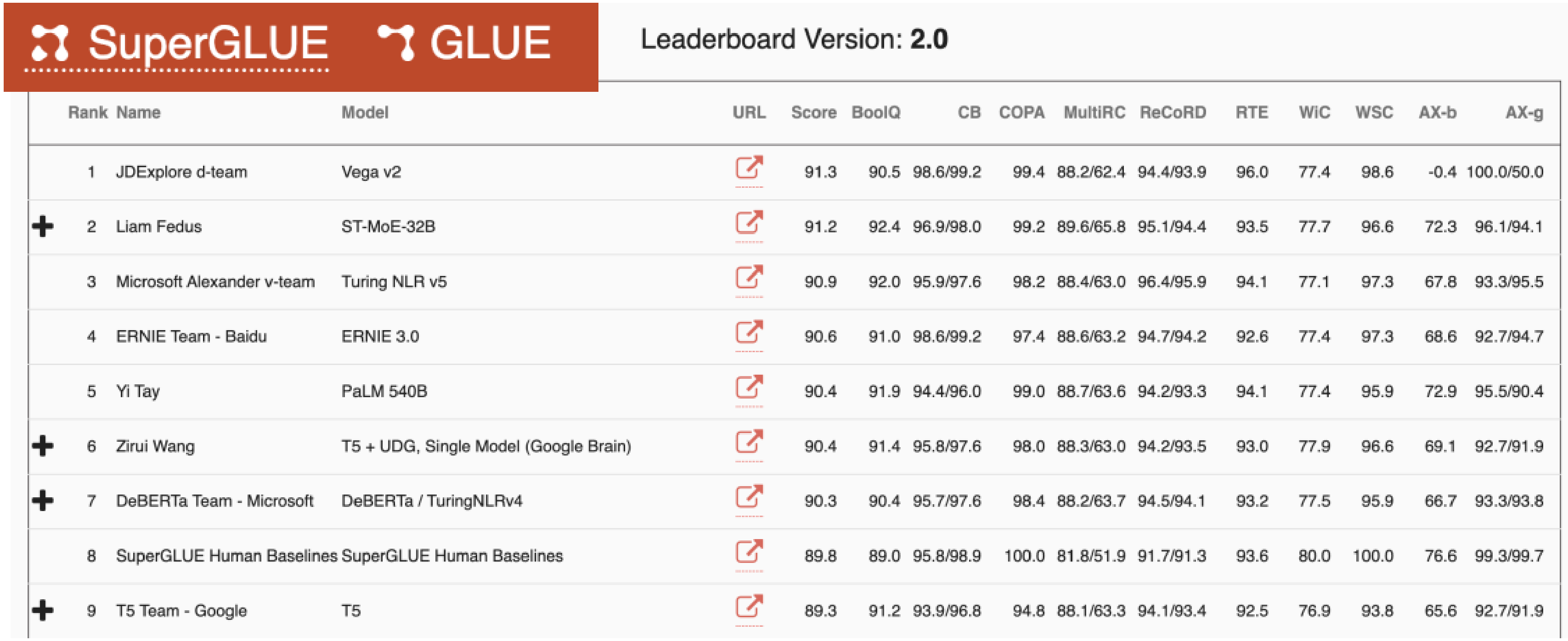
SST, IMDB (Sentiment)

SNLI, MultiNLI (entailment)

SQUaD, NaturalQuestions (QA)

Multi-Task Benchmarks

- Attempt to measure more “**general** language capabilities” along various tasks

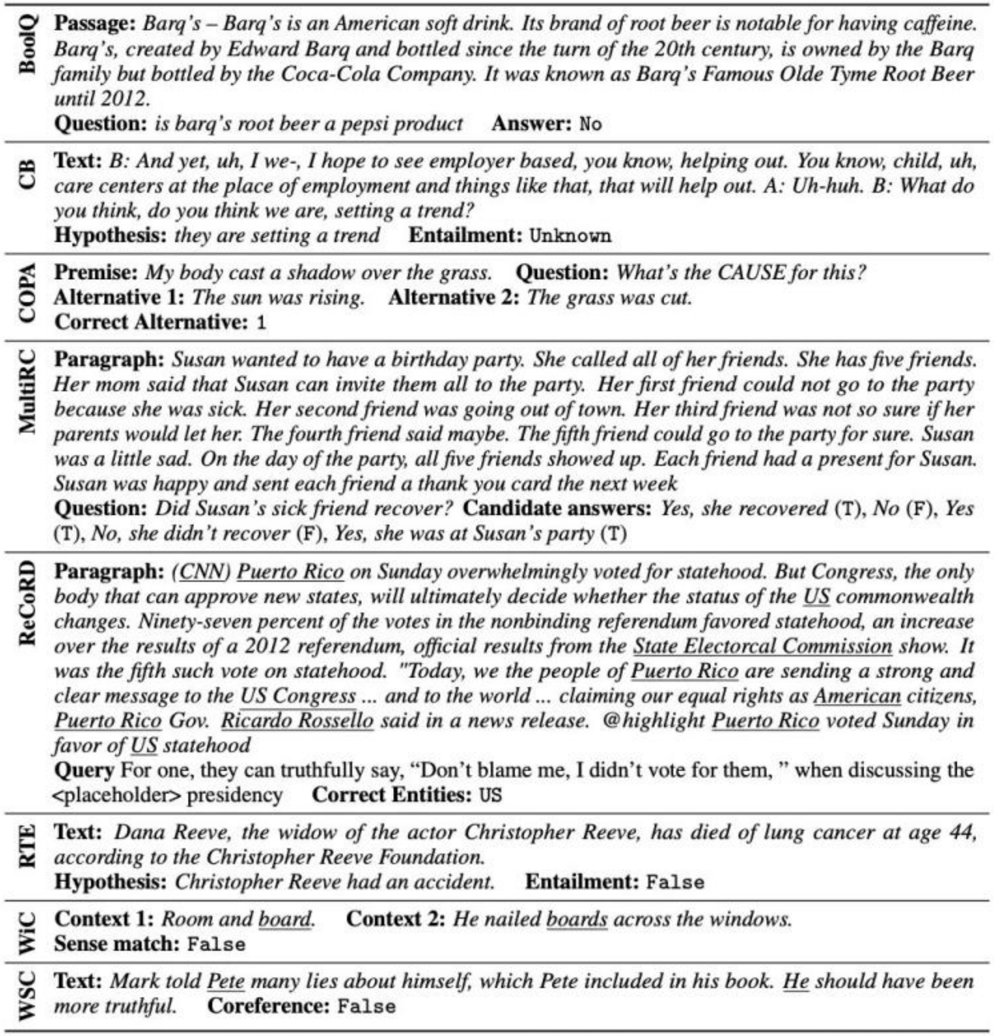


Leaderboard Version: **2.0**

	Rank	Name	Model	URL	Score	BoolQ	CB	COPA	MultiRC	ReCoRD	RTE	WiC	WSC	AX-b	AX-g
	1	JDExplore d-team	Vega v2	↗	91.3	90.5	98.6/99.2	99.4	88.2/62.4	94.4/93.9	96.0	77.4	98.6	-0.4	100.0/50.0
+	2	Liam Fedus	ST-MoE-32B	↗	91.2	92.4	96.9/98.0	99.2	89.6/65.8	95.1/94.4	93.5	77.7	96.6	72.3	96.1/94.1
	3	Microsoft Alexander v-team	Turing NLR v5	↗	90.9	92.0	95.9/97.6	98.2	88.4/63.0	96.4/95.9	94.1	77.1	97.3	67.8	93.3/95.5
	4	ERNIE Team - Baidu	ERNIE 3.0	↗	90.6	91.0	98.6/99.2	97.4	88.6/63.2	94.7/94.2	92.6	77.4	97.3	68.6	92.7/94.7
	5	Yi Tay	PaLM 540B	↗	90.4	91.9	94.4/96.0	99.0	88.7/63.6	94.2/93.3	94.1	77.4	95.9	72.9	95.5/90.4
+	6	Zirui Wang	T5 + UDG, Single Model (Google Brain)	↗	90.4	91.4	95.8/97.6	98.0	88.3/63.0	94.2/93.5	93.0	77.9	96.6	69.1	92.7/91.9
+	7	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4	↗	90.3	90.4	95.7/97.6	98.4	88.2/63.7	94.5/94.1	93.2	77.5	95.9	66.7	93.3/93.8
	8	SuperGLUE Human Baselines	SuperGLUE Human Baselines	↗	89.8	89.0	95.8/98.9	100.0	81.8/51.9	91.7/91.3	93.6	80.0	100.0	76.6	99.3/99.7
+	9	T5 Team - Google	T5	↗	89.3	91.2	93.9/96.8	94.8	88.1/63.3	94.1/93.4	92.5	76.9	93.8	65.6	92.7/91.9

Example: superGLUE

- BoolQ, MultiRC (reading texts)
- CB, RTE (Entailment)
- COPA (cause and effect)
- ReCoRD (QA+reasoning)
- WiC (meaning of words)
- WSC (co-reference)

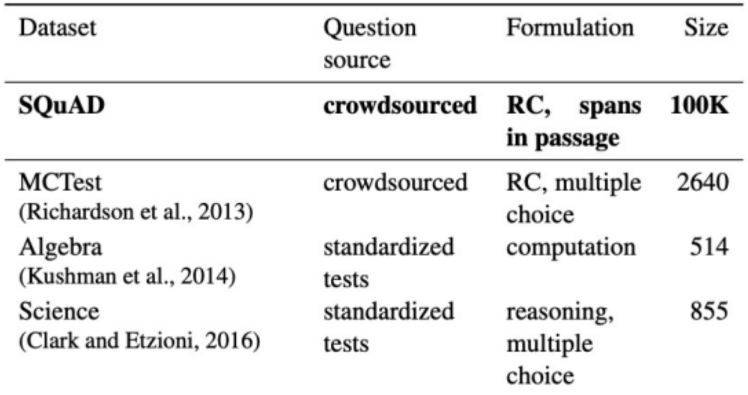


BoolQ	**Passage:** *Barq's – Barq's is an American soft drink. Its brand of root beer is notable for having caffeine. Barq's, created by Edward Barq and bottled since the turn of the 20th century, is owned by the Barq family but bottled by the Coca-Cola Company. It was known as Barq's Famous Olde Tyme Root Beer until 2012.* **Question:** *is barq's root beer a pepsi product* **Answer:** `No`
CB	**Text:** *B: And yet, uh, I we-, I hope to see employer based, you know, helping out. You know, child, uh, care centers at the place of employment and things like that, that will help out. A: Uh-huh. B: What do you think, do you think we are, setting a trend?* **Hypothesis:** *they are setting a trend* **Entailment:** `Unknown`
COPA	**Premise:** *My body cast a shadow over the grass.* **Question:** *What's the CAUSE for this?* **Alternative 1:** *The sun was rising.* **Alternative 2:** *The grass was cut.* **Correct Alternative:** `1`
MultiRC	**Paragraph:** *Susan wanted to have a birthday party. She called all of her friends. She has five friends. Her mom said that Susan can invite them all to the party. Her first friend could not go to the party because she was sick. Her second friend was going out of town. Her third friend was not so sure if her parents would let her. The fourth friend said maybe. The fifth friend could go to the party for sure. Susan was a little sad. On the day of the party, all five friends showed up. Each friend had a present for Susan. Susan was happy and sent each friend a thank you card the next week* **Question:** *Did Susan's sick friend recover?* **Candidate answers:** *Yes, she recovered* (`T`), *No* (`F`), *Yes* (`T`), *No, she didn't recover* (`F`), *Yes, she was at Susan's party* (`T`)
ReCoRD	**Paragraph:** *(CNN) Puerto Rico on Sunday overwhelmingly voted for statehood. But Congress, the only body that can approve new states, will ultimately decide whether the status of the US commonwealth changes. Ninety-seven percent of the votes in the nonbinding referendum favored statehood, an increase over the results of a 2012 referendum, official results from the State Electorcal Commission show. It was the fifth such vote on statehood. "Today, we the people of Puerto Rico are sending a strong and clear message to the US Congress ... and to the world ... claiming our equal rights as American citizens, Puerto Rico Gov. Ricardo Rossello said in a news release. @highlight Puerto Rico voted Sunday in favor of US statehood* **Query** For one, they can truthfully say, "Don't blame me, I didn't vote for them, " when discussing the <placeholder> presidency **Correct Entities:** `US`
RTE	**Text:** *Dana Reeve, the widow of the actor Christopher Reeve, has died of lung cancer at age 44, according to the Christopher Reeve Foundation.* **Hypothesis:** *Christopher Reeve had an accident.* **Entailment:** `False`
WiC	**Context 1:** *Room and board.* **Context 2:** *He nailed boards across the windows.* **Sense match:** `False`
WSC	**Text:** *Mark told Pete many lies about himself, which Pete included in his book. He should have been more truthful.* **Coreference:** `False`

What Makes a Good Benchmark?

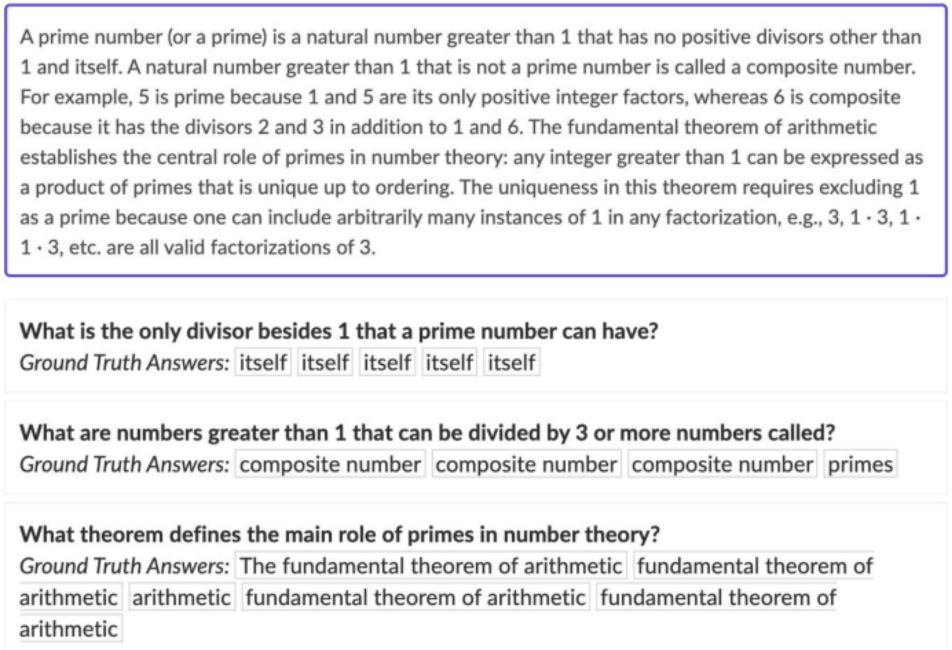
- **Example selection** (scale, diversity)
  - Benchmark should cover phenomena of interest
  - Complex phenomena require many samples
- **Difficulty**
  - Doable for humans
  - Hard for baselines at the same time
- **Annotation Quality**
  - “Correct” behavior should be clear

SQuAD: A Successful Benchmark



Dataset	Question source	Formulation	Size
SQuAD	**crowdsourced**	**RC, spans in passage**	**100K**
MCTest (Richardson et al., 2013)	crowdsourced	RC, multiple choice	2640
Algebra (Kushman et al., 2014)	standardized tests	computation	514
Science (Clark and Etzioni, 2016)	standardized tests	reasoning, multiple choice	855

Scale (and inclusion of training data)

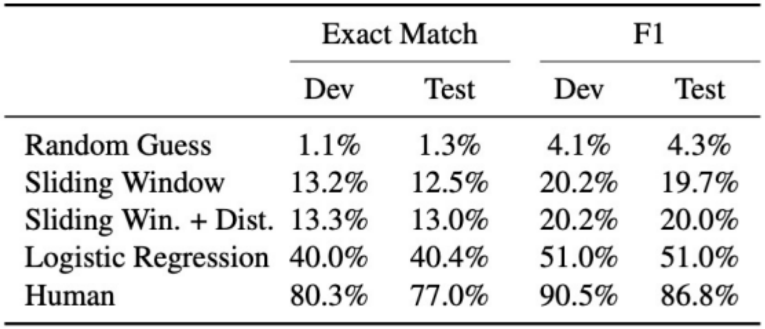


A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself. A natural number greater than 1 that is not a prime number is called a composite number. For example, 5 is prime because 1 and 5 are its only positive integer factors, whereas 6 is composite because it has the divisors 2 and 3 in addition to 1 and 6. The fundamental theorem of arithmetic establishes the central role of primes in number theory: any integer greater than 1 can be expressed as a product of primes that is unique up to ordering. The uniqueness in this theorem requires excluding 1 as a prime because one can include arbitrarily many instances of 1 in any factorization, e.g., 3, 1 · 3, 1 · 1 · 3, etc. are all valid factorizations of 3.

What is the only divisor besides 1 that a prime number can have?
Ground Truth Answers: itself itself itself itself itself

What are numbers greater than 1 that can be divided by 3 or more numbers called?
Ground Truth Answers: composite number composite number composite number primes

What theorem defines the main role of primes in number theory?
Ground Truth Answers: The fundamental theorem of arithmetic fundamental theorem of arithmetic arithmetic fundamental theorem of arithmetic fundamental theorem of arithmetic



	Exact Match		F1	
	Dev	Test	Dev	Test
Random Guess	1.1%	1.3%	4.1%	4.3%
Sliding Window	13.2%	12.5%	20.2%	19.7%
Sliding Win. + Dist.	13.3%	13.0%	20.2%	20.0%
Logistic Regression	40.0%	40.4%	51.0%	51.0%
Human	80.3%	77.0%	90.5%	86.8%

Easy, relatively clean automatic evaluation

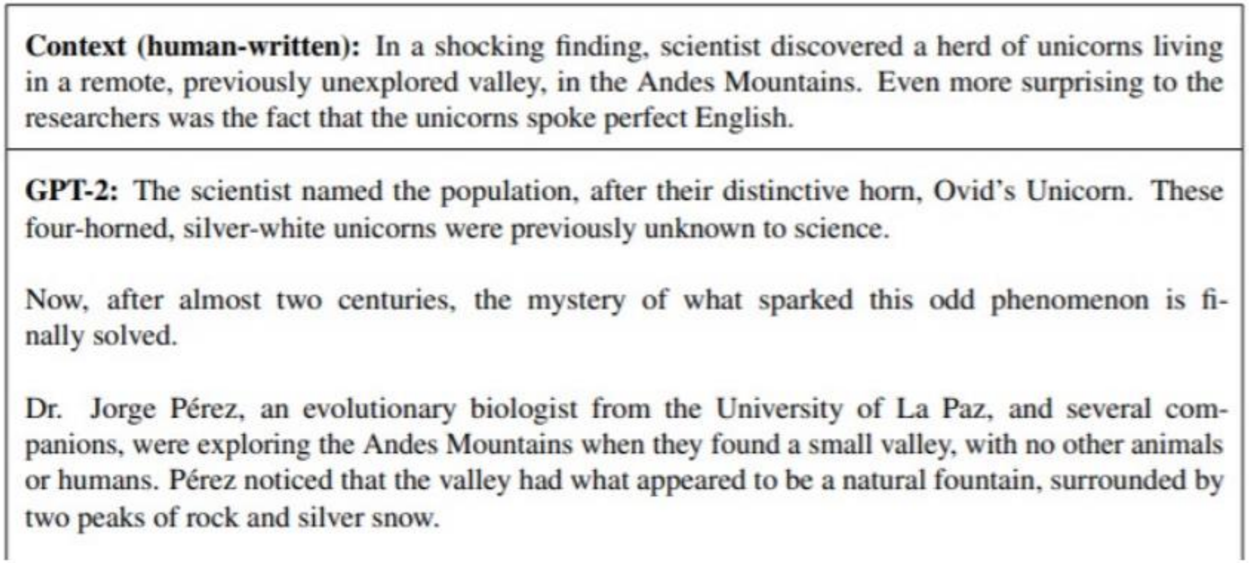
Large headroom to human perf

Fitting the Dataset vs Learning the Task

- Across a wide range of tasks, high model accuracy on the in-domain test set **does not imply** the model will also do well on other, “reasonable” out-of-domain examples.

- One way to think about this: models seem to be **learning the dataset not the task** (like how humans would perform the task even on new unseen examples).

Open-ended Evaluation



Context (human-written): In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

GPT-2: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

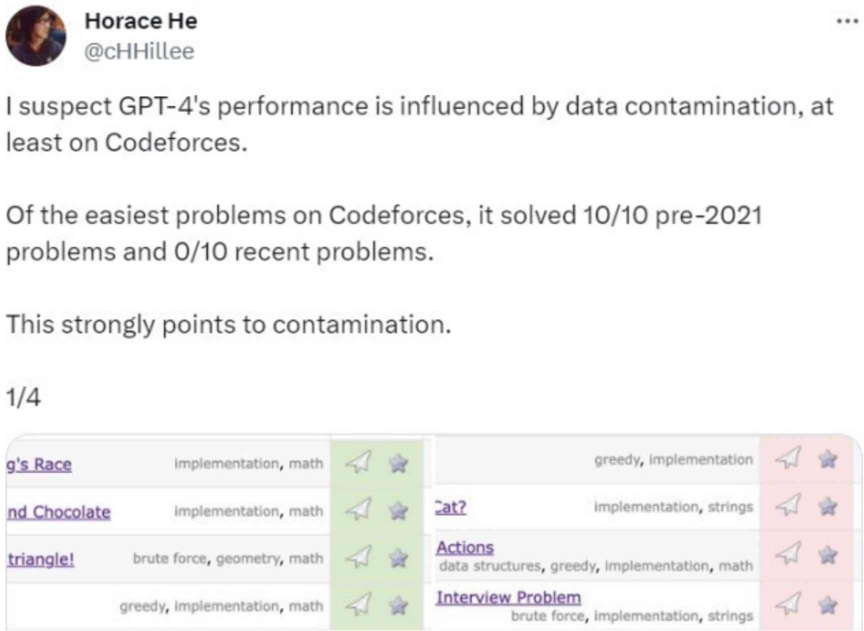
- From a “few correct answers” to “thousands of correct answers”
- No more easy automatic evaluation (or is there?)
- There are now better and worse answers (not just right and wrong)

Evaluating Large Language Models

- **Benchmark-based evaluation**
  - Format problem into prompt
  - Generate result
  - Parse result and compare with correct answer using standard metrics like accuracy
  - Good for close-ended evaluation
- Pros
  - Leverage existing benchmarks
  - Allows automatic evaluation
- Cons
  - LLMs sensitive to evaluation setting (prompts, answer parsing)
  - Data contamination

Benchmarks: Hard to Trust for LLMs

- Closed models + pre-training: Hard to know which benchmarks are truly "new"



Horace He
@cHHillee

I suspect GPT-4's performance is influenced by data contamination, at least on Codeforces.

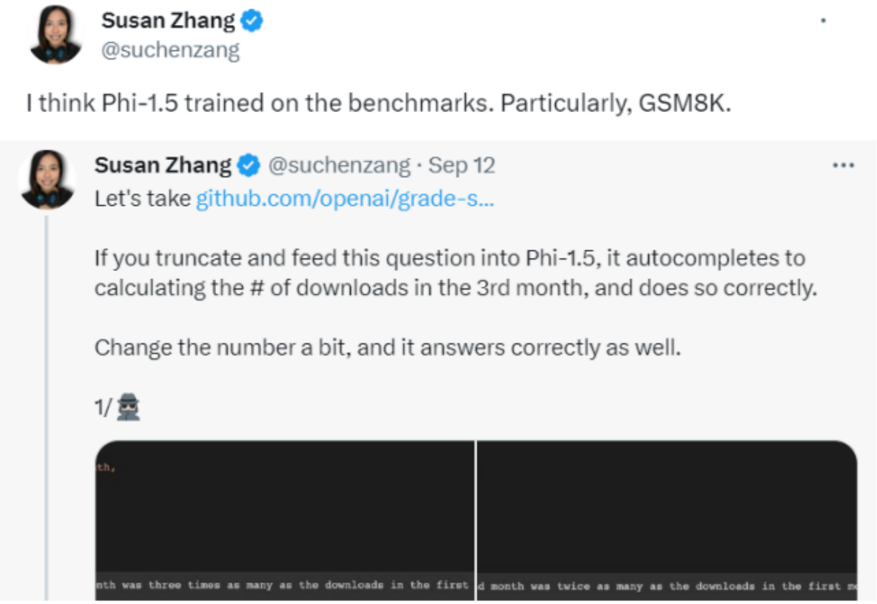
Of the easiest problems on Codeforces, it solved 10/10 pre-2021 problems and 0/10 recent problems.

This strongly points to contamination.

1/4

g's Race	implementation, math
nd Chocolate	implementation, math
triangle!	brute force, geometry, math
	greedy, implementation, math

	greedy, implementation
:at?	implementation, strings
Actions	data structures, greedy, implementation, math
Interview Problem	brute force, implementation, strings



Susan Zhang
@suchenzang

I think Phi-1.5 trained on the benchmarks. Particularly, GSM8K.

Susan Zhang @suchenzang · Sep 12
Let's take github.com/openai/grade-s...

If you truncate and feed this question into Phi-1.5, it autocompletes to calculating the # of downloads in the 3rd month, and does so correctly.

Change the number a bit, and it answers correctly as well.

1/🕵

```
th,
```

```
nth was three times as many as the downloads in the first
```

```
d month was twice as many as the downloads in the first m
```

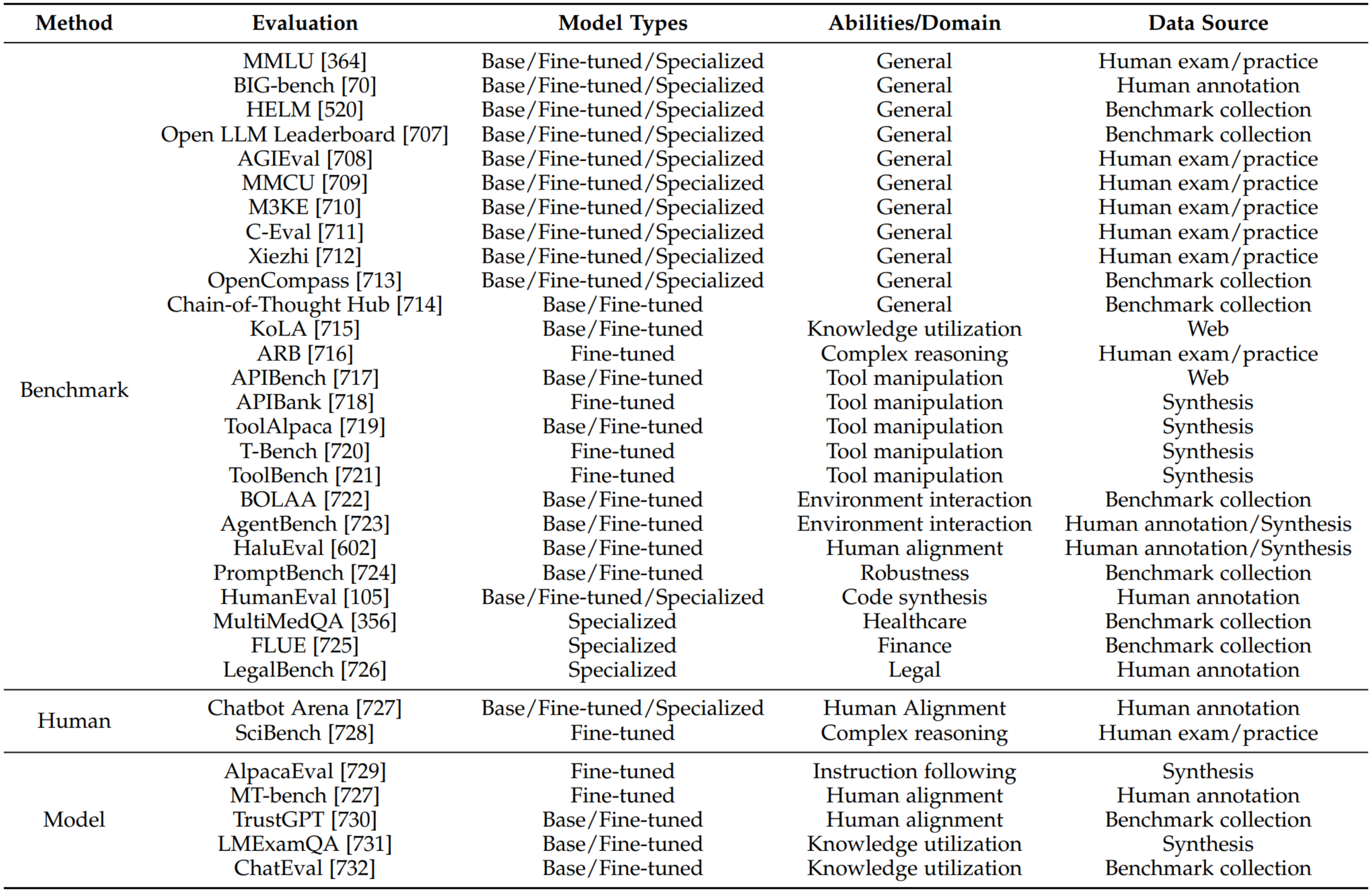
Evaluating Large Language Models

- **Human-based evaluation**
  - Human evaluators judge answer of LLMs
    - Pair-wise comparison of two answers from different models
    - Single-answer grading: score a single answer from an LLM
  - Good for open-ended evaluation
- Pros
  - Directly reflect actual abilities of LLMs in real-world scenarios
  - More flexible and diverse evaluation tasks
- Cons
  - Personalized taste and varying education levels can introduce biases
  - Judgements may be very subjective
  - Requires many evaluators and results are not reproducible

Evaluating Large Language Models

- **Model-based evaluation**
  - Use LLM like GPT-4 as surrogate for human evaluation
  - Shown to achieve high agreement with human evaluators
  - Active research in creating open-source evaluator models
- Pros
  - Scalable compared to human evaluation
  - Can provide explanations for scores
  - Reproducible
- Cons
  - Position bias - Prefer answers at certain positions in the prompt
  - Verbosity bias - Favor verbose answers instead of quality
  - Self-enhancement bias - Favor their own generations over others

Benchmarks for Evaluating LLMs

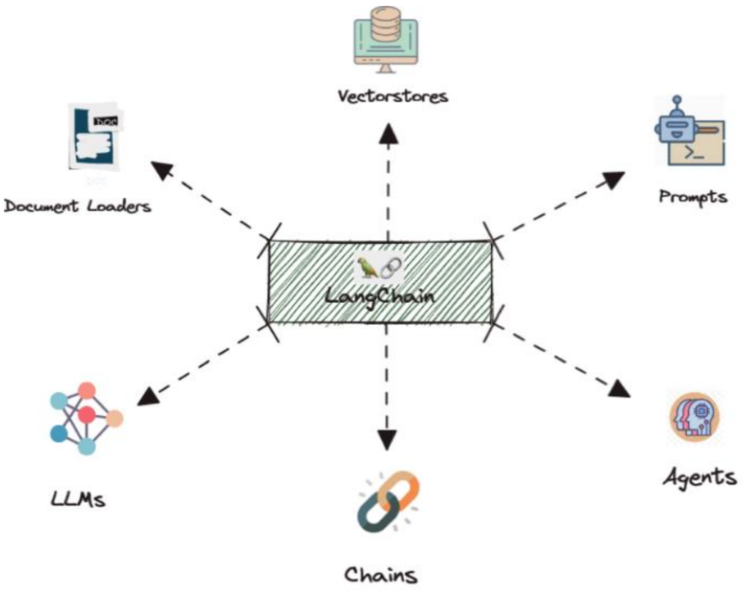


Method	Evaluation	Model Types	Abilities/Domain	Data Source
Benchmark	MMLU [364]	Base/Fine-tuned/Specialized	General	Human exam/practice
	BIG-bench [70]	Base/Fine-tuned/Specialized	General	Human annotation
	HELM [520]	Base/Fine-tuned/Specialized	General	Benchmark collection
	Open LLM Leaderboard [707]	Base/Fine-tuned/Specialized	General	Benchmark collection
	AGIEval [708]	Base/Fine-tuned/Specialized	General	Human exam/practice
	MMCU [709]	Base/Fine-tuned/Specialized	General	Human exam/practice
	M3KE [710]	Base/Fine-tuned/Specialized	General	Human exam/practice
	C-Eval [711]	Base/Fine-tuned/Specialized	General	Human exam/practice
	Xiezhi [712]	Base/Fine-tuned/Specialized	General	Human exam/practice
	OpenCompass [713]	Base/Fine-tuned/Specialized	General	Benchmark collection
	Chain-of-Thought Hub [714]	Base/Fine-tuned	General	Benchmark collection
	KoLA [715]	Base/Fine-tuned	Knowledge utilization	Web
	ARB [716]	Fine-tuned	Complex reasoning	Human exam/practice
	APIBench [717]	Base/Fine-tuned	Tool manipulation	Web
	APIBank [718]	Fine-tuned	Tool manipulation	Synthesis
	ToolAlpaca [719]	Base/Fine-tuned	Tool manipulation	Synthesis
	T-Bench [720]	Fine-tuned	Tool manipulation	Synthesis
	ToolBench [721]	Fine-tuned	Tool manipulation	Synthesis
	BOLAA [722]	Base/Fine-tuned	Environment interaction	Benchmark collection
	AgentBench [723]	Base/Fine-tuned	Environment interaction	Human annotation/Synthesis
	HaluEval [602]	Base/Fine-tuned	Human alignment	Human annotation/Synthesis
	PromptBench [724]	Base/Fine-tuned	Robustness	Benchmark collection
	HumanEval [105]	Base/Fine-tuned/Specialized	Code synthesis	Human annotation
	MultiMedQA [356]	Specialized	Healthcare	Benchmark collection
	FLUE [725]	Specialized	Finance	Benchmark collection
	LegalBench [726]	Specialized	Legal	Human annotation
Human	Chatbot Arena [727]	Base/Fine-tuned/Specialized	Human Alignment	Human annotation
	SciBench [728]	Fine-tuned	Complex reasoning	Human exam/practice
Model	AlpacaEval [729]	Fine-tuned	Instruction following	Synthesis
	MT-bench [727]	Fine-tuned	Human alignment	Human annotation
	TrustGPT [730]	Base/Fine-tuned	Human alignment	Benchmark collection
	LMExamQA [731]	Base/Fine-tuned	Knowledge utilization	Synthesis
	ChatEval [732]	Base/Fine-tuned	Knowledge utilization	Benchmark collection

Zhao et al.: A Survey of Large Language Models. 2024. arXiv:2303.18223

See you next week!

- Next time: Introduction to Langchain
  - Exercise: let's start applying things
  - Learn how to prompt hosted and open-source models
  - Learn how to apply the concepts we have seen so far


Vectorstores
Document Loaders
Prompts
LangChain
LLMs
Chains
Agents