In-context learning and prompting

LING 574 Deep Learning for NLP

Shane Steinert-Threlkeld

Announcements

- HW5: almost out, great job!
- HW7 out this afternoon
  - Sequence-to-sequence
  - Char-level MT
  - Implement attention
- Note on MacOS Sequoia environment

Fine-tuning

- How to apply a pretrained LM to data from a new task $\mathscr{D}' = \{(x'_i, y'_i)\}$
- Fine-tuning:
  - Take pre-trained LM, replace LM "head" with task-specific head
  - Supervised learning (either of whole model, or just the head) on $\mathscr{D}'$
  - *Updates parameters*

In-context learning

- In-context learning:
  - $x' := f_{\text{prompt}}(x)$, defines a *template* with slots [X] and [Y]
    - e.g. for SST: “Review: [X] Rating: [Y]”
    - “if you sometimes like to go to the movies to have fun , wasabi is a good place to start .” —> “Review: if you sometimes like to go to the movies to have fun , wasabi is a good place to start . Rating: ”
  - Concatenate *k examples* (could be 0!) with *y* filled in.
    - $f_{\text{fill}}(x_1, 3)$ = “Review: if you sometimes like to go to the movies to have fun , wasabi is a good place to start . Rating: 3”
    - NB: often use other tokens than the true labels (e.g. “good”)

In-context learning (cont.)

- Ask the model to generate based on final input, i.e. to sample

$$z_{k+1} \sim P\left(\cdot \,|\, f_{\text{fill}}(x_1, y_1), \ldots, f_{\text{fill}}(x_k, y_k), f_{\text{prompt}}(x_{k+1}) \right)$$

  - NB: possibly a sequence, not a single token.
  - Possibly with additional context / "task description" first:
    $P(\cdot \,|\, \text{task-description}, f_{\text{fill}}(x_1, y_1), \ldots, f_{\text{fill}}(x_k, y_k), f_{\text{prompt}}(x_{k+1}))$
- For classification: define an "answer function" mapping z_{k+1} back into your label space!
- *No parameter updates!*

GPT3 Few-Shot Results

	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	**89.0**	**91.0**	**96.9**	**93.9**	**94.8**	**92.5**
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	**76.1**	**93.8**	**62.3**	**88.2**	**92.5**	**93.3**
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1

k=32

GPT3 Few-Shot Results

	SuperGLUE Average	BoolQ Accuracy	CB Accuracy	CB F1	COPA Accuracy	RTE Accuracy
Fine-tuned SOTA	**89.0**	**91.0**	**96.9**	**93.9**	**94.8**	**92.5**
Fine-tuned BERT-Large	69.0	77.4	83.6	75.7	70.6	71.7
GPT-3 Few-Shot	71.8	76.4	75.6	52.0	92.0	69.0

	WiC Accuracy	WSC Accuracy	MultiRC Accuracy	MultiRC F1a	ReCoRD Accuracy	ReCoRD F1
Fine-tuned SOTA	**76.1**	**93.8**	**62.3**	**88.2**	**92.5**	**93.3**
Fine-tuned BERT-Large	69.6	64.6	24.1	70.0	71.3	72.0
GPT-3 Few-Shot	49.4	80.1	30.5	75.4	90.2	91.1

Context →	`The bet, which won him dinner for four, was regarding the existence and mass of the top quark, an elementary particle discovered in 1995.`
	`question: The Top Quark is the last of six flavors of quarks predicted by the standard model theory of particle physics. True or False?`
	`answer:`

Target Completion →	`False`

Figure G.31: Formatted dataset example for RTE

Prompt sensitivity

- Different choices for $f_{\text{prompt}}(x)$ can lead to drastically different performance!
- Even just basic format!
- "Prompt engineering": how to design and build effective prompts. More on this later.

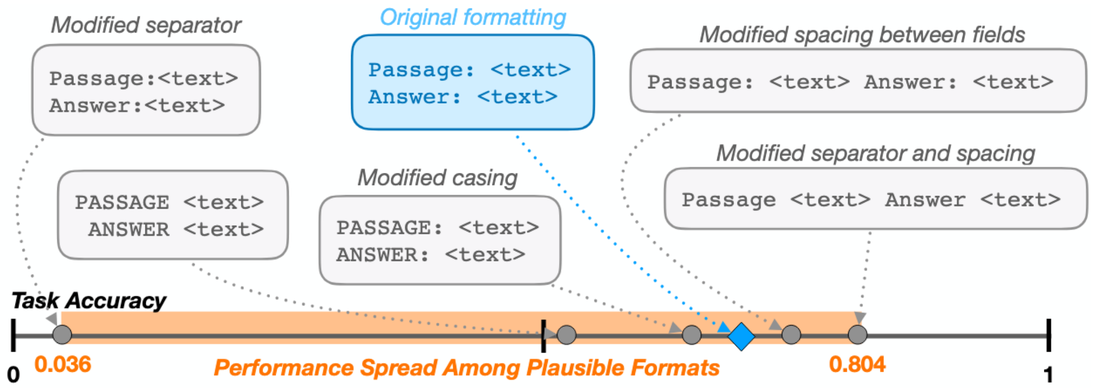


Figure 1: Slight modifications in prompt format templating may lead to significantly different model performance for a given task. Each <text> represents a different variable-length placeholder to be replaced with actual data samples. Example shown corresponds to 1-shot LLaMA-2-7B perfor-

Why/how does ICL work?

Some Mysteries

Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?

Sewon Min[1,2] **Xinxi Lyu**[1] **Ari Holtzman**[1] **Mikel Artetxe**[2]
Mike Lewis[2] **Hannaneh Hajishirzi**[1,3] **Luke Zettlemoyer**[1,2]
[1]University of Washington [2]Meta AI [3]Allen Institute for AI
{sewon,alrope,ahai,hannaneh,lsz}@cs.washington.edu
{artetxe,mikelewis}@meta.com

Abstract

Large language models (LMs) are able to in-context learn—perform a new task via inference alone by conditioning on a few input-label pairs (demonstrations) and making predictions for new inputs. However, there has been little understanding of *how* the model learns and *which* aspects of the demonstrations contribute to end task performance. In this paper, we show that ground truth demonstrations are in fact not required—randomly replacing labels in the demonstrations barely hurts performance on a range of classification and multi-choce tasks, consistently over 12 different models including GPT-3. Instead, we find that other aspects of the demonstrations are the key drivers of end task performance, including the fact that they provide a few examples of (1) the label space, (2) the distribution of the input text, and (3) the overall format of the sequence. Together, our analysis provides a new way of understanding how and why in-context learning works, while opening up new questions about how much can be learned from large language models through inference alone.

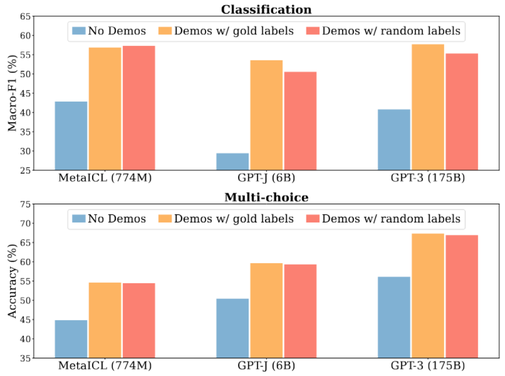


Figure 1: Results in classification (top) and multi-choice tasks (bottom), using three LMs with varying size. Reported on six datasets on which GPT-3 is evaluated; the channel method is used. See Section 4 for the full results. In-context learning performance drops only marginally when labels in the demonstrations are replaced by random labels.

source

is consistent over 12 different models including the GPT-3 family (Radford et al., 2019; Min et al., 2021b; Wang and Komatsuzaki, 2021; Artetxe

Some Mysteries

Do Prompt-Based Models Really Understand the Meaning of Their Prompts?

Albert Webson[1,2] and Ellie Pavlick[1]

{albert_webson, ellie_pavlick}@brown.edu
[1]Department of Computer Science, Brown University
[2]Department of Philosophy, Brown University

Abstract

Recently, a boom of papers has shown extraordinary progress in zero-shot and few-shot learning with various prompt-based models. It is commonly argued that prompts help models to learn faster in the same way that humans learn faster when provided with task instructions expressed in natural language. In this study, we experiment with over 30 prompt templates manually written for natural language inference (NLI). We find that models can learn just as fast with many prompts that are intentionally irrelevant or even pathologically misleading as they do with instructively "good" prompts. Further, such patterns hold

arbitrary dimensions of a one-hot vector. In contrast, suppose a human is given a prompt such as: Given that "no weapons of mass destruction found in Iraq yet.", is it definitely correct that "weapons of mass destruction found in Iraq."?[1] Then it would be no surprise that they are able to perform the task more accurately and without needing many examples to figure out what the task is.

Similarly, reformatting NLP tasks with prompts such as the underlined text above has dramatically improved zero-shot and few-shot performance over traditional fine-tuned models (Schick and Schütze, 2021b; Le Scao and Rush, 2021; Sanh et al., 2021; Wei et al., 2021). Such results naturally give rise to

source

Some Mysteries

Do Prompt-Based Models Really Understand the Meaning of Their Prompts?

Albert Webson[1,2] and Ellie Pavlick[1]

{albert_webson, ellie_pavlick}@brown.edu
[1]Department of Computer Science, Brown University
[2]Department of Philosophy, Brown University

Abstract

Recently, a boom of papers has shown extraordinary progress in zero-shot and few-shot learning with various prompt-based models. It is commonly argued that prompts help models to learn faster in the same way that humans learn faster when provided with task instructions expressed in natural language. In this study, we experiment with over 30 prompt templates manually written for natural language inference (NLI). We find that models can learn just as fast with many prompts that are intentionally irrelevant or even pathologically misleading as they do with instructively "good" prompts. Further, such patterns hold arbitrary dimensions of a one-hot vector. In contrast, suppose a human is given a prompt such as: Given that "no weapons of mass destruction found in Iraq yet.", is it definitely correct that "weapons of mass destruction found in Iraq."?[1] Then it would be no surprise that they are able to perform the task more accurately and without needing many examples to figure out what the task is.

Similarly, reformatting NLP tasks with prompts such as the underlined text above has dramatically improved zero-shot and few-shot performance over traditional fine-tuned models (Schick and Schütze, 2021b; Le Scao and Rush, 2021; Sanh et al., 2021; Wei et al., 2021). Such results naturally give rise to

source

Category	Examples
instructive	{prem} Are we justified in saying that "{hypo}"? Suppose {prem} Can we infer that "{hypo}"?
misleading-moderate	{prem} Can that be paraphrased as: "{hypo}"? {prem} Are there lots of similar words in "{hypo}"?
misleading-extreme	{prem} is the sentiment positive? {hypo} {prem} is this a sports news? {hypo}
irrelevant	{prem} If bonito flakes boil more than a few seconds the stock becomes too strong. "{hypo}"?
null	{premise} {hypothesis} {hypothesis} {premise}

Table 1: Example templates for NLI.

Some Mysteries

Do Prompt-Based Models Really Understand the Meaning of Their Prompts?

Albert Webson[1,2] and Ellie Pavlick[1]

{albert_webson, ellie_pavlick}@brown.edu
[1]Department of Computer Science, Brown University
[2]Department of Philosophy, Brown University

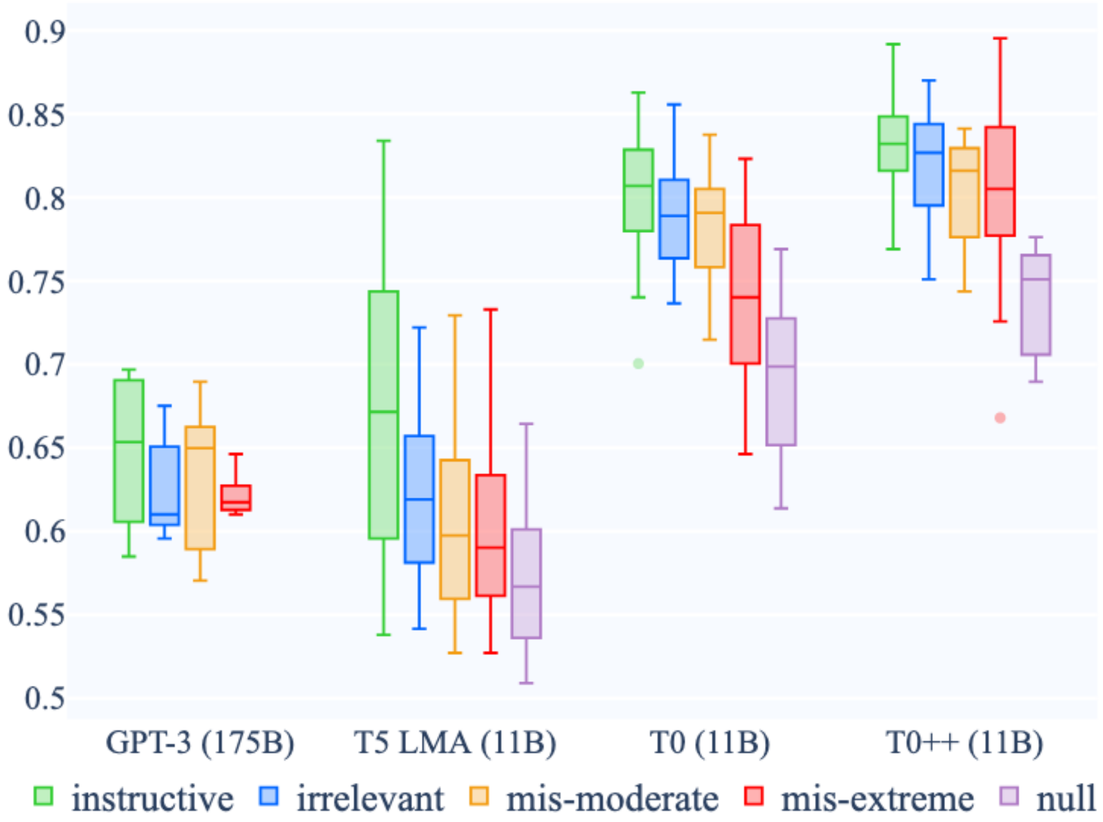
Abstract

Recently, a boom of papers has shown extraordinary progress in zero-shot and few-shot learning with various prompt-based models. It is commonly argued that prompts help models to learn faster in the same way that humans learn faster when provided with task instructions expressed in natural language. In this study, we experiment with over 30 prompt templates manually written for natural language inference (NLI). We find that models can learn just as fast with many prompts that are intentionally irrelevant or even pathologically misleading as they do with instructively "good" prompts. Further, such patterns hold arbitrary dimensions of a one-hot vector. In contrast, suppose a human is given a prompt such as: Given that "no weapons of mass destruction found in Iraq yet.", is it definitely correct that "weapons of mass destruction found in Iraq."?[1] Then it would be no surprise that they are able to perform the task more accurately and without needing many examples to figure out what the task is.

Similarly, reformatting NLP tasks with prompts such as the underlined text above has dramatically improved zero-shot and few-shot performance over traditional fine-tuned models (Schick and Schütze, 2021b; Le Scao and Rush, 2021; Sanh et al., 2021; Wei et al., 2021). Such results naturally give rise to

source

Category	Examples
instructive	{prem} Are we justified in saying that "{hypo}"?



Linear Regression Case Study

Published as a conference paper at ICLR 2023

WHAT LEARNING ALGORITHM IS IN-CONTEXT LEARNING? INVESTIGATIONS WITH LINEAR MODELS

Ekin Akyürek[1,2,a] **Dale Schuurmans**[1] **Jacob Andreas**[*2] **Tengyu Ma**[*1,3,b] **Denny Zhou**[*1]

[1]Google Research [2]MIT CSAIL [3] Stanford University [*]collaborative advising

ABSTRACT

Neural sequence models, especially transformers, exhibit a remarkable capacity for *in-context learning*. They can construct new predictors from sequences of labeled examples $(x, f(x))$ presented in the input without further parameter updates. We investigate the hypothesis that transformer-based in-context learners implement standard learning algorithms *implicitly*, by encoding smaller models in their activations, and updating these implicit models as new examples appear in the context. Using linear regression as a prototypical problem, we offer three sources of evidence for this hypothesis. First, we prove by construction that transformers can implement learning algorithms for linear models based on gradient descent and closed-form ridge regression. Second, we show that trained in-context learners closely match the predictors computed by gradient descent, ridge regression, and exact least-squares regression, transitioning between different predictors as transformer depth and dataset noise vary, and converging to Bayesian estimators for large widths and depths. Third, we present preliminary evidence that in-context learners share algorithmic features with these predictors: learners' late layers non-linearly encode weight vectors and moment matrices. These results suggest that in-context learning is understandable in algorithmic terms, and that (at least in the linear case) learners may rediscover standard estimation algorithms.

source

Linear Regression Case Study

Published as a conference paper at ICLR 2023

WHAT LEARNING ALGORITHM IS IN-CONTEXT LEARNING? INVESTIGATIONS WITH LINEAR MODELS

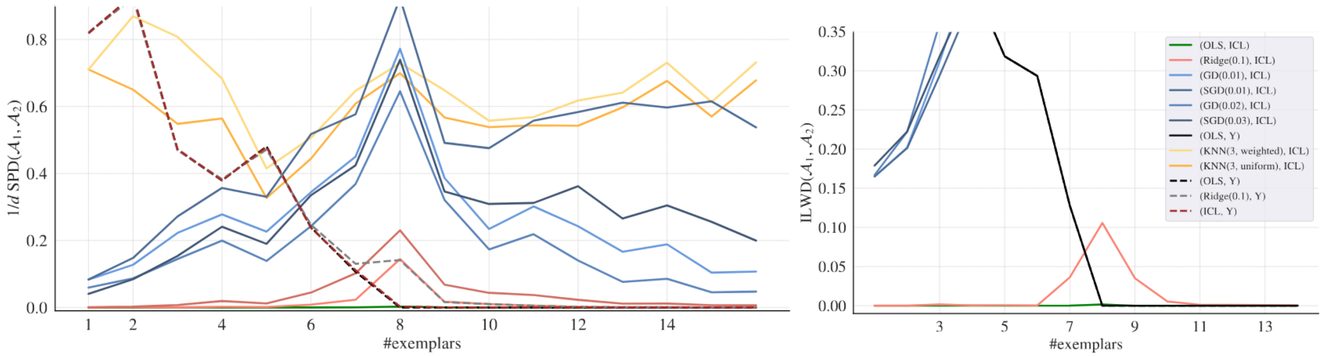
Ekin Akyürek[1,2,a] **Dale Schuurmans**[1] **Jacob Andreas**[*2] **Tengyu Ma**[*1,3,b] **Denny Zhou**[*1]

[1]Google Research [2]MIT CSAIL [3] Stanford University [*]collaborative advising

ABSTRACT

Neural sequence models, especially transformers, exhibit a remarkable capacity for *in-context learning*. They can construct new predictors from sequences of labeled examples $(x, f(x))$ presented in the input without further parameter updates. We investigate the hypothesis that transformer-based in-context learners implement standard learning algorithms *implicitly*, by encoding smaller models in their activations, and updating these implicit models as new examples appear in the context. Using linear regression as a prototypical problem, we offer three sources of evidence for this hypothesis. First, we prove by construction that transformers can implement learning algorithms for linear models based on gradient descent and closed-form ridge regression. Second, we show that trained in-context learners closely match the predictors computed by gradient descent, ridge regression, and exact least-squares regression, transitioning between different predictors as transformer depth and dataset noise vary, and converging to Bayesian estimators for large widths and depths. Third, we present preliminary evidence that in-context learners share algorithmic features with these predictors: learners' late layers non-linearly encode weight vectors and moment matrices. These results suggest that in-context learning is understandable in algorithmic terms, and that (at least in the linear case) learners may rediscover standard estimation algorithms.

source



(a) Predictor–ICL fit w.r.t. prediction differences. (b) Predictor–ICL fit w.r.t implicit weights.

Figure 1: **Fit between ICL and standard learning algorithms:** We plot (dimension normalized) SPD and ILWD values between textbook algorithms and ICL on noiseless linear regression with $d = 8$. GD(α) denotes one step of batch gradient descent and SGD(α) denotes one pass of stochastic gradient descent with learning rate α. Ridge(λ) denotes Ridge regression with regularization parameter λ. Under both evaluations, in-context learners *agree* closely with ordinary least squares, and are significantly less well approximated by other solutions to the linear regression problem.

Linear Regression Case Study

Published as a conference paper at ICLR 2023

WHAT LEARNING ALGORITHM IS IN-CONTEXT LEARNING? INVESTIGATIONS WITH LINEAR MODELS

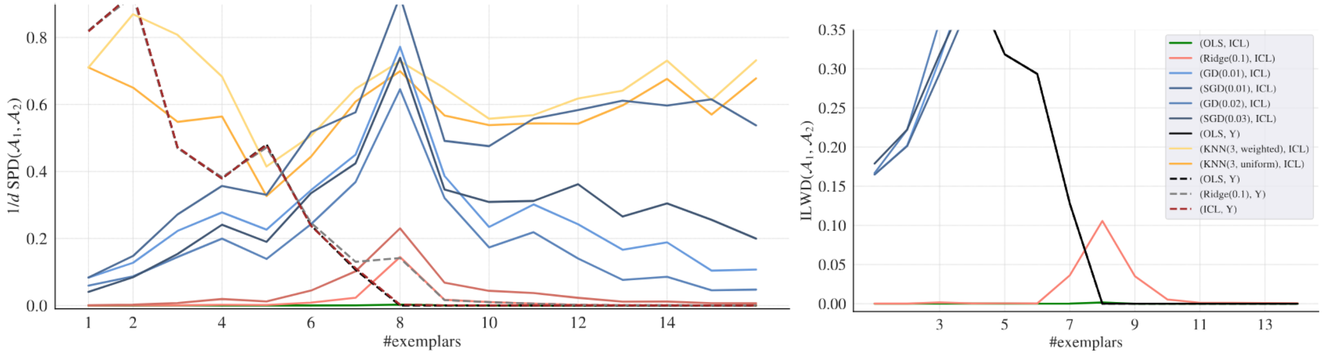
Ekin Akyürek[1,2,a] **Dale Schuurmans**[1] **Jacob Andreas**[*2] **Tengyu Ma**[*1,3,b] **Denny Zhou**[*1]

[1]Google Research [2]MIT CSAIL [3] Stanford University [*]collaborative advising

ABSTRACT

Neural sequence models, especially transformers, exhibit a remarkable capacity for *in-context learning*. They can construct new predictors from sequences of labeled examples $(x, f(x))$ presented in the input without further parameter updates. We investigate the hypothesis that transformer-based in-context learners implement standard learning algorithms *implicitly*, by encoding smaller models in their activations, and updating these implicit models as new examples appear in the context. Using linear regression as a prototypical problem, we offer three sources of evidence for this hypothesis. First, we prove by construction that transformers can implement learning algorithms for linear models based on gradient descent and closed-form ridge regression. Second, we show that trained in-context learners closely match the predictors computed by gradient descent, ridge regression, and exact least-squares regression, transitioning between different predictors as transformer depth and dataset noise vary, and converging to Bayesian estimators for large widths and depths. Third, we present preliminary evidence that in-context learners share algorithmic features with these predictors: learners' late layers non-linearly encode weight vectors and moment matrices. These results suggest that in-context learning is understandable in algorithmic terms, and that (at least in the linear case) learners may rediscover standard estimation algorithms.

source

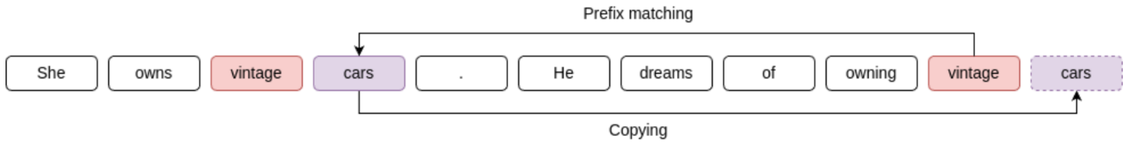


(a) Predictor–ICL fit w.r.t. prediction differences. (b) Predictor–ICL fit w.r.t implicit weights.

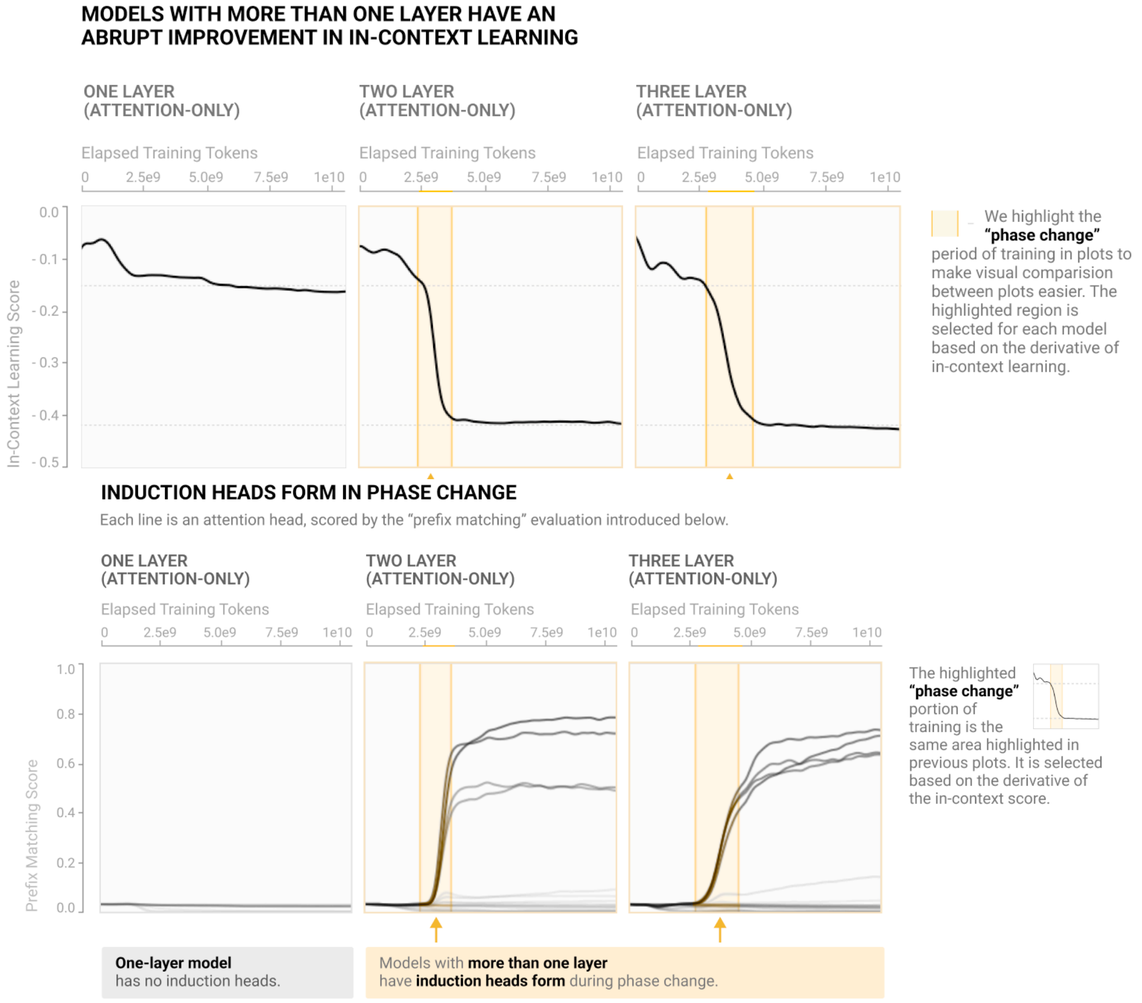
Figure 1: **Fit between ICL and standard learning algorithms:** We plot (dimension normalized) SPD and ILWD values between textbook algorithms and ICL on noiseless linear regression with $d = 8$. GD(α) denotes one step of batch gradient descent and SGD(α) denotes one pass of stochastic gradient descent with learning rate α. Ridge(λ) denotes Ridge regression with regularization parameter λ. Under both evaluations, in-context learners *agree* closely with ordinary least squares, and are significantly less well approximated by other solutions to the linear regression problem.

Induction Heads

- Induction head = attention head which:
  - Matches a prefix
  - “Copies” the attended to token
    - [A][B]...[A]→[B]
  - Analogical/fuzzy version -> ICL
    - [A*][B*] … [A] → [B]



source



Induction Heads

Induction Heads as an Essential Mechanism for Pattern Matching in In-context Learning

Joy Crosbie*
ILLC
University of Amsterdam
joy.m.crosbie@gmail.com

Ekaterina Shutova
ILLC
University of Amsterdam
e.shutova@uva.nl

Abstract

Large language models (LLMs) have shown a remarkable ability to learn and perform complex tasks through in-context learning (ICL). However, a comprehensive understanding of its internal mechanisms is still lacking. This paper explores the role of *induction heads* in a few-shot ICL setting. We analyse two state-of-the-art models, Llama-3-8B and InternLM2-20B on abstract pattern recognition and NLP tasks. Our results show that even a minimal ablation of induction heads leads to ICL performance decreases of up to ~32% for abstract pattern recognition tasks, bringing the performance close to random. For NLP tasks, this ablation substantially decreases the model's ability to benefit from examples, bringing few-shot ICL performance close to that of zero-shot prompts. We further use *attention knockout* to disable specific induction patterns, and present fine-grained evidence for the role that the induction mechanism plays in ICL.

Von Oswald et al., 2023), suggesting that ICL functions as an implicit form of fine-tuning at inference time. Other works investigated factors influencing ICL, showing that it is driven by the distributions of the training data (Chan et al., 2022) and scales with model size, revealing new abilities at certain parameter thresholds (Brown et al., 2020; Wei et al., 2022). During inference, the properties of demonstration samples also affect ICL performance , with aspects such as the label space, input distribution and input-label pairing playing a crucial role (Min et al., 2022; Webson and Pavlick, 2022). While this work identified interesting properties of ICL and effective ICL prompting strategies, a comprehensive understanding of its operational mechanisms within the models is still lacking.

Our paper aims to fill this gap, by directly investigating the internal model computations that enable ICL. Our work is inspired by recent research in the field of mechanistic interpretability, which aims to reverse engineer the "algorithm" by which Transformer models process information (Geva et al.,

		Llama-3-8B						
Task	Shot	Full model	1% ind. heads	1% ind. pattern	1% rnd. heads	3% ind. heads	3% ind. pattern	3% rnd. heads
Repetition	5	67.3	61.9 (-6.4)	62.5 (-4.8)	64.9 (-2.4)	50.0 (-17.3)	53.2 (-14.1)	62.6 (-4.7)
	10	91.3	59.7 (-31.6)	65.3 (-26.0)	85.9 (-5.4)	51.5 (-39.8)	58.7 (-32.6)	79.7 (-11.6)
Recursion	5	67.8	63.0 (-4.8)	61.5 (-6.3)	66.1 (-1.7)	52.0 (-15.8)	55.1 (-12.7)	68.0 (+0.2)
	10	91.5	67.9 (-23.6)	68.7 (-22.8)	86.5 (-5.0)	52.9 (-38.6)	58.9 (-32.6)	84.6 (-6.9)
Centre-embedding	5	58.8	54.9 (-3.9)	55.0 (-3.8)	57.2 (-1.6)	49.1 (-9.7)	50.5 (-8.3)	56.4 (-2.4)
	10	80.4	53.0 (-27.4)	56.5 (-23.9)	74.6 (-5.8)	50.7 (-29.7)	52.3 (-28.1)	71.5 (-8.9)
WordSeq 1 (binary)	5	83.1	72.1 (-11.0)	71.8 (-11.3)	82.1 (-1.0)	51.6 (-31.5)	56.9 (-26.2)	78.8 (-4.3)
	10	99.4	96.2 (-3.2)	97.3 (-2.1)	99.3 (-0.1)	69.4 (-30.0)	82.2 (-17.2)	98.3 (-1.1)
WordSeq 2 (binary)	5	77.9	65.4 (-12.5)	65.2 (-12.7)	76.8 (-1.1)	52.0 (-25.9)	55.7 (-22.2)	72.4 (-5.5)
	10	99.4	94.9 (-4.5)	96.4 (-3.0)	98.8 (-0.6)	67.2 (-32.2)	81.1 (-18.3)	97.7 (-1.7)
WordSeq 1 (4-way)	20	78.3	55.2 (-23.1)	59.8 (-18.5)	76.5 (-1.8)	40.8 (-37.5)	45.2 (-33.1)	71.4 (-6.9)
WordSeq 2 (4-way)	20	81.3	55.9 (-25.4)	59.8 (-21.5)	76.0 (-5.3)	42.3 (-39.0)	47.5 (-33.8)	68.6 (-12.7)

source

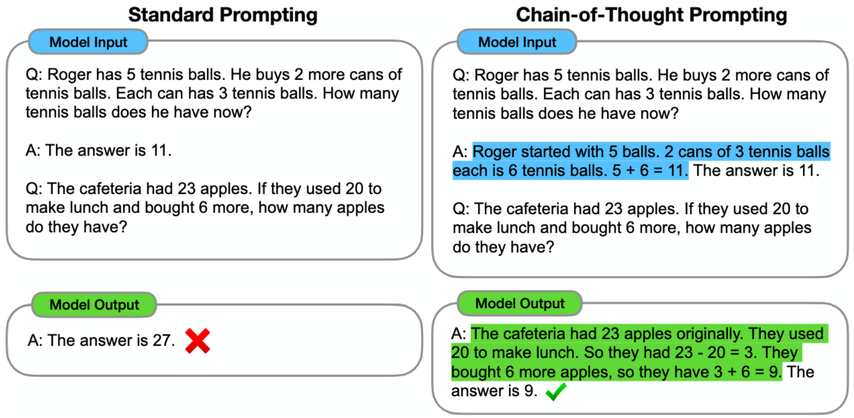
Prompting Strategies

“Chain of Thought”

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Jason Wei **Xuezhi Wang** **Dale Schuurmans** **Maarten Bosma**
Brian Ichter **Fei Xia** **Ed H. Chi** **Quoc V. Le** **Denny Zhou**

Google Research, Brain Team
{jasonwei,dennyzhou}@google.com



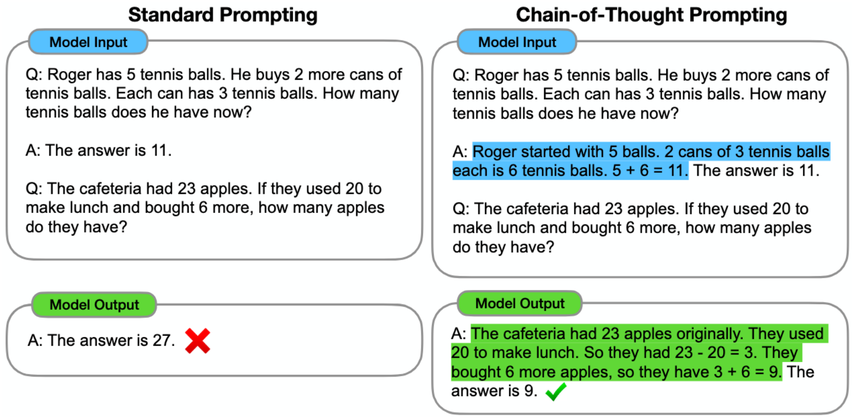
source

“Chain of Thought”

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

Jason Wei Xuezhi Wang Dale Schuurmans Maarten Bosma
Brian Ichter Fei Xia Ed H. Chi Quoc V. Le Denny Zhou

Google Research, Brain Team
{jasonwei,dennyzhou}@google.com



Hand-written!
For k=8 examples

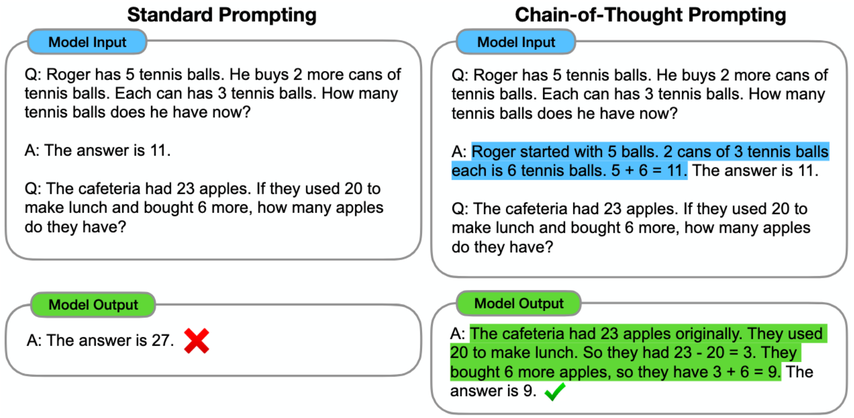
source

“Chain of Thought”

Chain-of-Thought Prompting Elicits Reasoning in Large Language Models

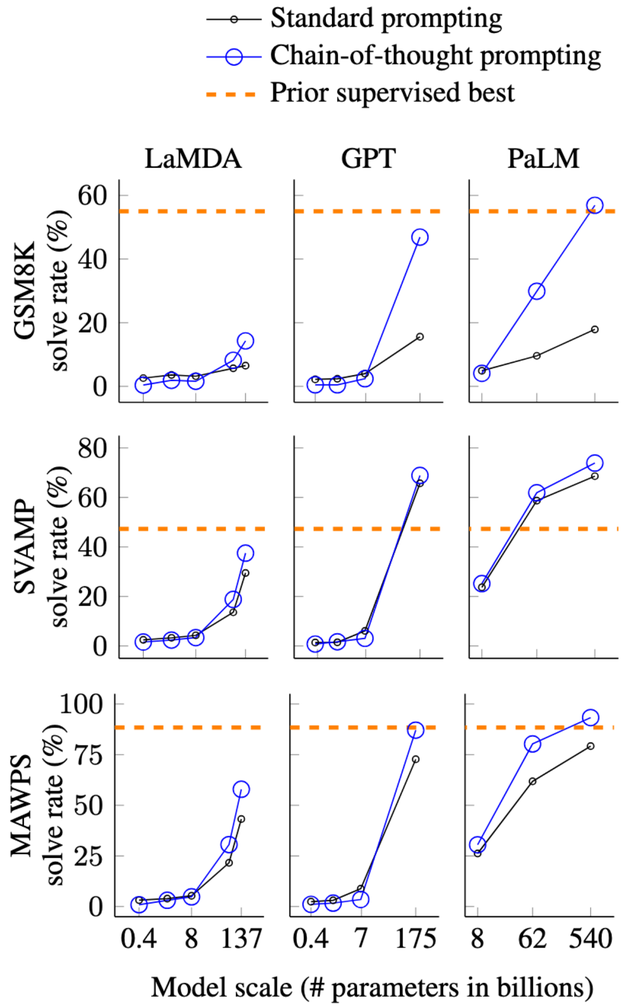
Jason Wei Xuezhi Wang Dale Schuurmans Maarten Bosma
Brian Ichter Fei Xia Ed H. Chi Quoc V. Le Denny Zhou

Google Research, Brain Team
{jasonwei,dennyzhou}@google.com



source

Hand-written!
For k=8 examples



“Let’s think step by step”

Large Language Models are Zero-Shot Reasoners

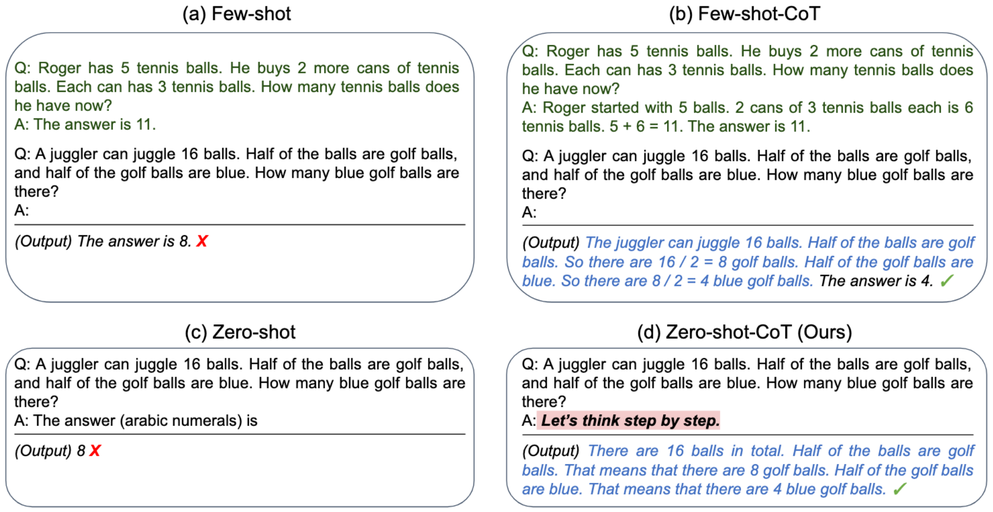
Takeshi Kojima
The University of Tokyo
t.kojima@weblab.t.u-tokyo.ac.jp

Shixiang Shane Gu
Google Research, Brain Team

Machel Reid
Google Research*

Yutaka Matsuo
The University of Tokyo

Yusuke Iwasawa
The University of Tokyo



“Let’s think step by step”

Large Language Models are Zero-Shot Reasoners

Takeshi Kojima
The University of Tokyo
t.kojima@weblab.t.u-tokyo.ac.jp

Shixiang Shane Gu
Google Research, Brain Team

Machel Reid
Google Research*

Yutaka Matsuo
The University of Tokyo

Yusuke Iwasawa
The University of Tokyo

(a) Few-shot

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: 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:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

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: 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:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

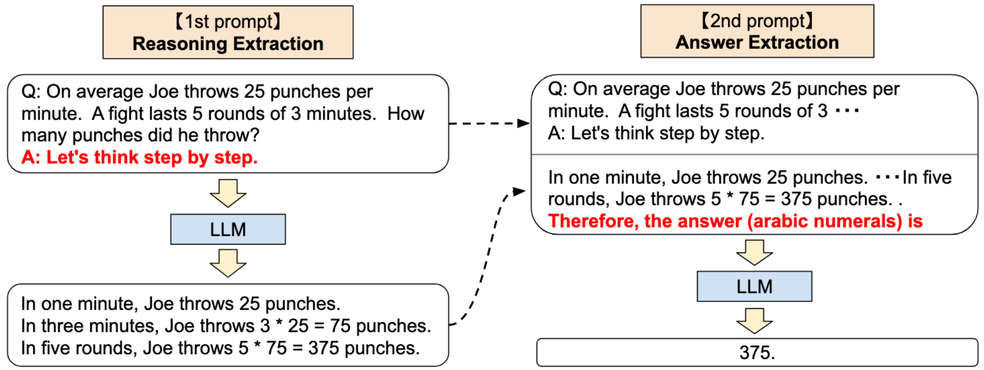
Q: A juggler can juggle 16 balls. Half of the balls are golf balls, and half of the golf balls are blue. How many blue golf balls are there?
A: The answer (arabic numerals) is

(Output) 8 ✗

(d) Zero-shot-CoT (Ours)

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. ✓



“Let’s think step by step”

Large Language Models are Zero-Shot Reasoners

Takeshi Kojima
The University of Tokyo
t.kojima@weblab.t.u-tokyo.ac.jp

Shixiang Shane Gu
Google Research, Brain Team

Machel Reid
Google Research*

Yutaka Matsuo
The University of Tokyo

Yusuke Iwasawa
The University of Tokyo

(a) Few-shot

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: 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:

(Output) The answer is 8. ✗

(b) Few-shot-CoT

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: 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:

(Output) The juggler can juggle 16 balls. Half of the balls are golf balls. So there are 16 / 2 = 8 golf balls. Half of the golf balls are blue. So there are 8 / 2 = 4 blue golf balls. The answer is 4. ✓

(c) Zero-shot

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 ✗

(d) Zero-shot-CoT (Ours)

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. ✓

【1st prompt】 **Reasoning Extraction**

Q: On average Joe throws 25 punches per minute. A fight lasts 5 rounds of 3 minutes. How many punches did he throw?

【2nd prompt】 **Answer Extraction**

Q: On average Joe throws 25 punches per minute. A fight lasts 5 rounds of 3 ···
A: Let's think step by step.

	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
Zero-Plus-Few-Shot-CoT (8 samples) (*2)	**92.8**	**51.5**
Finetuned GPT-3 175B [Wei et al., 2022]	-	33
Finetuned GPT-3 175B + verifier [Wei et al., 2022]	-	55
PaLM 540B: Zero-Shot	**25.5**	**12.5**
PaLM 540B: Zero-Shot-CoT	**66.1**	**43.0**
PaLM 540B: Zero-Shot-CoT + self consistency	**89.0**	**70.1**
PaLM 540B: Few-Shot [Wei et al., 2022]	-	17.9
PaLM 540B: Few-Shot-CoT [Wei et al., 2022]	-	56.9
PaLM 540B: Few-Shot-CoT + self consistency [Wang et al., 2022]	-	74.4

More examples and resources

- Self-consistency
- Tree of thoughts
- Self-ask
- STaR
- A survey of several: https://www.promptingguide.ai/techniques

Searching for Prompts

- Can we do something more principled, i.e. search for prompts automatically instead of hand-engineering?
- Discrete search:
  - AutoPrompt
  - DSPy: “Programming—not prompting—LMs”
- Continuous search / prompts (not necessarily interpretable!):
  - Prefix tuning
  - Prompt tuning

Resources

- “Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing”
  - Paper: https://arxiv.org/abs/2107.13586
  - Companion (updated) website: http://pretrain.nlpedia.ai/
- https://www.promptingguide.ai/
- A Survey on In-context Learning: https://aclanthology.org/2024.emnlp-main.64/

Summary

- In-context learning uses prompts to ask models to solve many different tasks, without updating parameters
- Sensitive to many choices:
  - Prompt template
  - Task description
  - Exemplars/in-context examples
  - …
- Today: high-level overview of some of what's known about how and why this method works to the extent that it does, pointers to more info.