CS 546 – Advanced Topics in NLP

Dilek Hakkani-Tür

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

CONVAI

CDS

Siebel School of Computing and Data Science

Any PEFT Questions?

- Adapters
- LoRA
- Prefix Tuning

Topics for Today

- Major Paradigms in NLP
- Prompting
- In-context Learning
- Prompting Examples

Readings

- Liu et al., Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing, ACM Computing Surveys, 2023.

*** Many examples, images and tables from this paper are included in my slides.

- Schulhoff et al., The Prompt Report: A Systematic Survey of Prompt Engineering Techniques, Preprint, 2024.

- Dong et al., A survey on In-context Learning. EMNLP, 2024.

- Bertsch et al., In-Context Learning with Long-Context Models: An In-Depth Exploration, NAACL, 2025.

Major Paradigms in NLP

- The classical approach
- Relies on manually created rules and lexical resources to process language.
- A human specifies the system's behavior explicitly with rules that directly encode the logic.
- **Pros:** Transparent, interpretable, and precise for well-defined domains.
- **Cons:** Brittle, difficult to scale, and poor generalization to open-domain tasks.

Major Paradigms in NLP (cont.)

- Fully supervised learning with statistical or classical machine learning models (non-neural networks, models such as support vector machines and conditional random fields)
- A human manually designs and selects the features (POS tags, lemmas, TF-IDF, etc.) —or inputs—to the model.
- Goal: represent raw text data in a numerical format that is effective for the task.
- **Pros:** Control over what the model sees and works well with limited data.
- **Cons:** Requires domain expertise and features can fail to capture deep domain semantics.

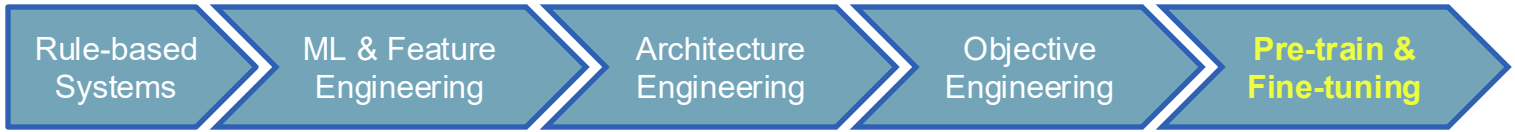
Major Paradigms in NLP (cont.)

- The approach of designing and training specific neural network architectures (such as, recurrent neural networks, gated attention, etc.) for particular NLP tasks, often without pre-training and fully supervised learning.
- The model's architecture itself is engineered to learn features automatically (rather than manual feature engineering).
- **Pros:** Architecture can embed domain knowledge (inductive biases) and the design can be interpretable.
- **Cons:** Manual effort and domain expertise is needed, lots of trial and error is required, models can be fragile.

Major Paradigms in NLP (cont.)

- The design of **training objectives / loss functions** that guide training of a model.
- Instead of focusing on *inputs* (feature engineering) or *model structure* (architecture engineering), objective engineering focuses on *what the model is optimized to do*.
- Examples: contrastive objectives; next sentence or masked word prediction for language models.
- **Pros:** Flexible (you can adapt models to new goals), can scale (i.e., web data)
- **Cons:** Can be complex and require expertise (to match the end task).

Major Paradigms in NLP (cont.)



- Game changer due to performance improvements for many tasks.
- Unsupervised pretraining of massive models on large text corpora, then fine-tuning for tasks.
- **Pros:** Few-shot and zero-shot abilities; broad generalization; reduced need for task-specific datasets.
- **Cons:** High compute costs, opaqueness, hallucinations, ethical concerns.

Topics for Today

- Major Paradigms in NLP
- Prompting
- In-context Learning
- Prompting Examples

Major Paradigms in NLP (cont.)

- Instead of adapting pre-trained LMs to downstream tasks via adding additional parameters and fine-tuning, downstream tasks are reformulated to look more like those solved during the original LM training.
- Example: recognizing the emotion of a tweet
  "I missed the bus today."
- Continue it with a prompt
  "I felt so _____"
- Manipulate the model behavior so that the pre-trained LM itself can be used to predict the desired output
  - sometimes even without any additional task-specific training.

Given a set of prompts, an LLM trained in a self-supervised fashion can be used to solve many tasks!

Prompt Engineering

- Finding the most appropriate prompt to solve a given task using a given LLM with a high accuracy.

- Prompt-based Learning
  - Steps for Prompting
  - Design Considerations for Prompting
    - Choosing the Pre-trained LM (PLM)
    - Prompt Engineering
    - Answer Engineering
    - Multi-Prompt Learning

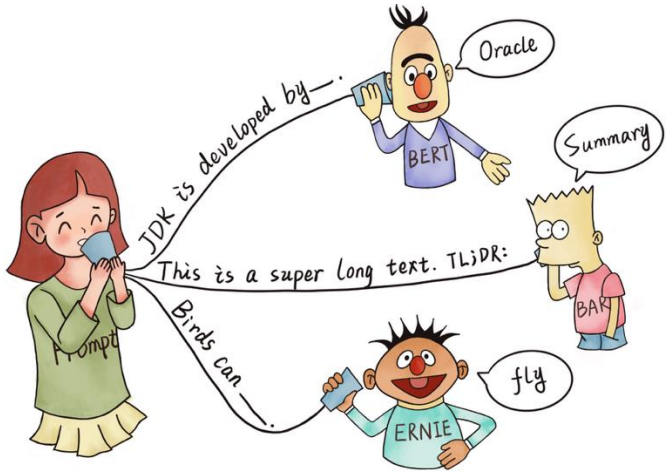


Image from the prompting paper of Liu et al, 2021

Summary of Terminology for Prompting

Name	Notation	Example	Description
Input	$\boldsymbol{x}$	I love this movie.	One or multiple texts
Output	$\boldsymbol{y}$	++ (very positive)	Output label or text
Prompting Function	$f_{\text{prompt}}(\boldsymbol{x})$	`[X]` Overall, it was a `[Z]` movie.	A function that converts the input into a specific form by inserting the input $\boldsymbol{x}$ and adding a slot `[Z]` where answer $\boldsymbol{z}$ may be filled later.
Prompt	$\boldsymbol{x}'$	I love this movie. Overall, it was a `[Z]` movie.	A text where `[X]` is instantiated by input $\boldsymbol{x}$ but answer slot `[Z]` is not.
Filled Prompt	$f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z})$	I love this movie. Overall, it was a bad movie.	A prompt where slot `[Z]` is filled with any answer.
Answered Prompt	$f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z}^*)$	I love this movie. Overall, it was a good movie.	A prompt where slot `[Z]` is filled with a true answer.
Answer	$\boldsymbol{z}$	"good", "fantastic", "boring"	A token, phrase, or sentence that fills `[Z]`

Table from the prompting paper of Liu et al, 2021

Steps for Prompting

- Formulating the prompt templates
- Formulating the set of answers
- Predicting the answers
- Mapping answers back to target labels

Formulating the Prompt Templates

- Example task: sentiment classification for tweets.
  - Input: x = “I enjoyed this movie a lot.”
  - Output: y ∈ {positive, negative}

- Create a prompt, x, using x.

- **Template** with two slots [x] and[z]:
  - x = “[x] It was a [z] movie.”
  - x is the input, z is the intermediate, generated answer that will later be mapped to y.

- **Prompt**:
  - “I enjoyed this movie a lot. It was a [z] movie.”

Template Examples for Various Tasks

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

Table from the prompting paper of Liu et al. 2021

Formulating the Prompt Templates (cont.)

- The empty slot to fill, z, can be in the middle (akin to the cloze task) or at the end.
- The template tokens don't need to be natural language.
- The number of x and z slots in the template may change depending on the task.
- The template may include answer options, e.g., for classification tasks

[x] Out of possible topics, such as agriculture, politics, finance, science and technology, this document is about [z]

Formulating the Set of Answers

- Determining the terms in $\mathcal{Z}$. It could be equivalent to the complete vocabulary or a small set of terms.

- A mapping between labels and answers
  - Positive => great
  - Negative => boring

Predicting the answer

- Given the prompt, predict [z], that maximizes the LM probability.
  "I enjoyed this movie a lot. It was a **great** movie."

$$\hat{z} = \underset{z \in \mathcal{Z}}{\text{argmax}} \; P(f_{\text{fill}}(x', z); \theta).$$

- Constrained generation to set $\mathcal{Z}$
- Choosing the right language model, i.e., pre-trained with masked language modeling objective for our template.

Mapping the Answer to the Target Labels

- Map the given answer (i.e., great for [z]), into the class label.
  - **Great => positive**

Prompting – Pros and Cons

Pros:

- No need for expensive parameter fine-tuning.
- No need for creating training datasets (for zero-shot prompting).

Cons:

- There are many ways to write prompts for each task and performance is dependent on prompt quality.
- The same prompt may not work similarly with different models.

Prompts are not all created equal!

- Gonen et al, Findings of EMNLP, 2023.

- Perplexity is a strong predictor of the success of a prompt.

- Lowest perplexity prompts are consistently effective.

Prompt	Accuracy
What is this piece of news regarding?	40.9
What is this article about?	52.4
What is the best way to describe this article?	68.2
What is the most accurate label for this news article?	71.2

Table 1: Example prompts for the task AG News (news classification) that vary considerably in accuracy.

Perplexity

- How surprised an LM is when seeing the next word.
- Measured as 2^H when dealing with log base 2.
- Formally, it is the inverse probability of the test set normalized by the number of words:

$$\text{Perplexity} = P(w_1, w_2, ..., w_N)^{-\frac{1}{N}} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(w_i|w_1, ..., w_{i-1})\right)$$

Average cross entropy

Small Differences in Prompts Can Result in Big Changes in Performance

- Sclar et al, ICLR, 2024.
- Studies evaluating LLMs with prompting-based methods would benefit from reporting a range of performance across plausible prompt formats.
- Comparing models with respect to fixed prompts may not make sense.

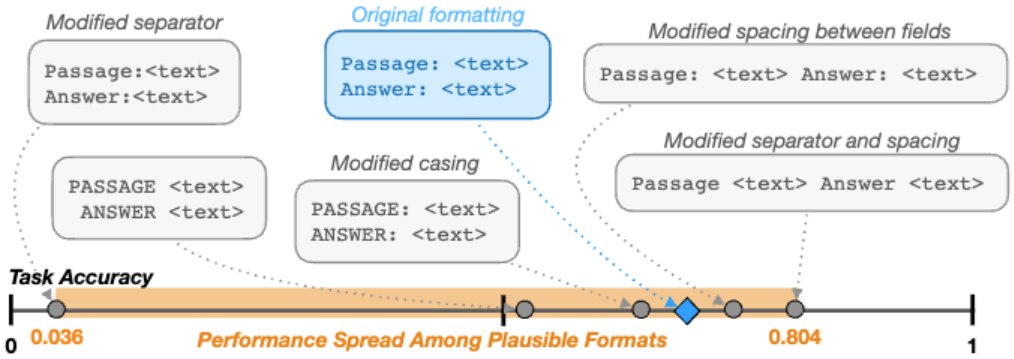


Figure 1: Slight modifications in prompt format templating may lead to significantly different model performance for a given task. Each `<text>` represents a different variable-length placeholder to be replaced with actual data samples. Example shown corresponds to 1-shot LLaMA-2-7B performances for task280 from SuperNaturalInstructions (Wang et al., 2022). This StereoSet-inspired task (Nadeem et al., 2021) requires the model to, given a short passage, classify it into one of four types of stereotype or anti-stereotype (gender, profession, race, and religion).

Things to Consider When Designing Prompts

- Given a task/prompt, which LLM may be more appropriate?
  - E.g., Masked LMs versus Autoregressive models

- Or vice versa, what prompt shape would be more appropriate given the task and model?
  - Cloze/Masking Tokens
    "I enjoyed this movie a lot. Overall, it was a [z] movie."
  - Prefix
    "I enjoyed this movie a lot. Overall, this movie was [z]"

Things to Consider When Designing Prompts (cont.)

- Given a task/prompt, how to define a good mapping function between the target labels and answers?
  - Answer shape (tokens, spans, sentences, etc.)
  - Manual design versus search (discrete versus continuous)

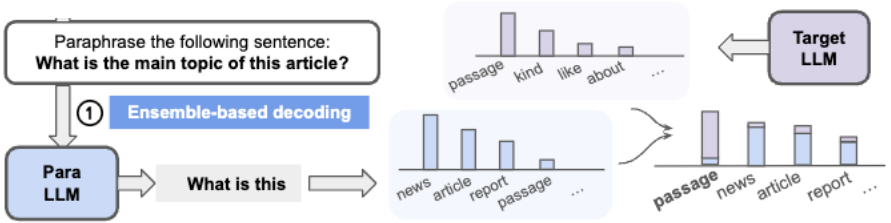
Automated Template Learning – Discrete Prompts

- **Prompt mining:** given a set of training inputs, x, and targets, y, mine patterns from large corpora (i.e., Wikipedia) that include x and y (Jiang et al., ACL 2020; Xue et al., 2024)

- **Prompt Scoring:** Hand-craft potential templates, fill them with examples to form prompts, score them with an LM, and select the most probable ones. (Davidson et al., EMNLP 2019)

Automated Template Learning – Discrete Prompts (cont.)

- **Prompt paraphrasing:** given an existing prompt, derive paraphrases, and use the one that maximizes accuracy on task data. Paraphrasing can be done in many ways, such as
  - translating to another language and back (Jiang et al, ACL 2020),
  - using linguistic resources (Yuan et al, NeurIPS 2021),
  - using LLMs to rewrite the prompts (Haviv et al., ACL 2021),
  - monotonic paraphrasing (Liu et al., EMNLP 2024)
    - finding the prompt with the lowest perplexity is challenging
    - MP is an ensemble based decoding method that combines the token probabilities from the paraphrase model and the target model in each decoding step



Automated Template Learning – Discrete Prompts (cont.)

- **Gradient-based Search:** search over tokens to find short sequences that can trigger an LLM to generate the desired target prediction (Wallace et al., EMNLP 2019; Pryzant et al., EMNLP 2023)
  - Jailbreaking LLMs

- **Prompt Generation:** Treat finding of prompts as text generation task, and training LLMs to generate templates (Gao et al., ACL, 2021)

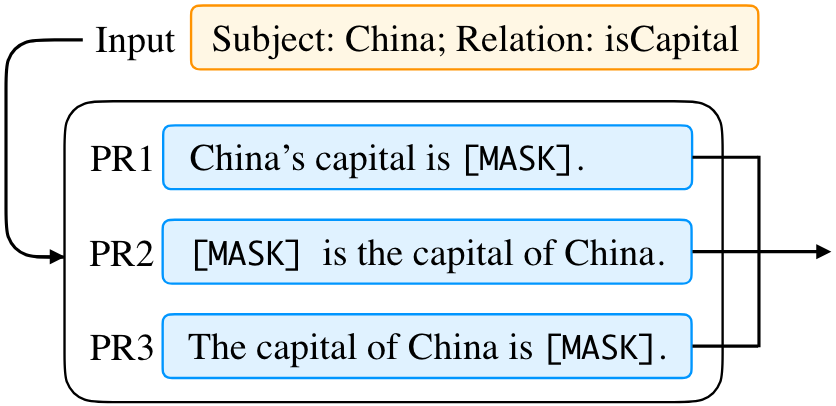
Automated Template Learning – Continuous Prompts

- Also called soft prompts
- Relax the constraint that the embeddings of template words be the embeddings of natural language (e.g., English) words.
- Remove the restriction that the template is parameterized by the pre-trained LM's parameters.
- Templates have their own parameters that can be tuned using training data.

- Prefix Tuning (Li and Liang, 2021)

Multiple-Prompts: Prompt Ensembling

- Focuses on learning multiple prompts for an input, instead of single one.
  - **Prompt Ensembling**



(a) Prompt Ensembling.

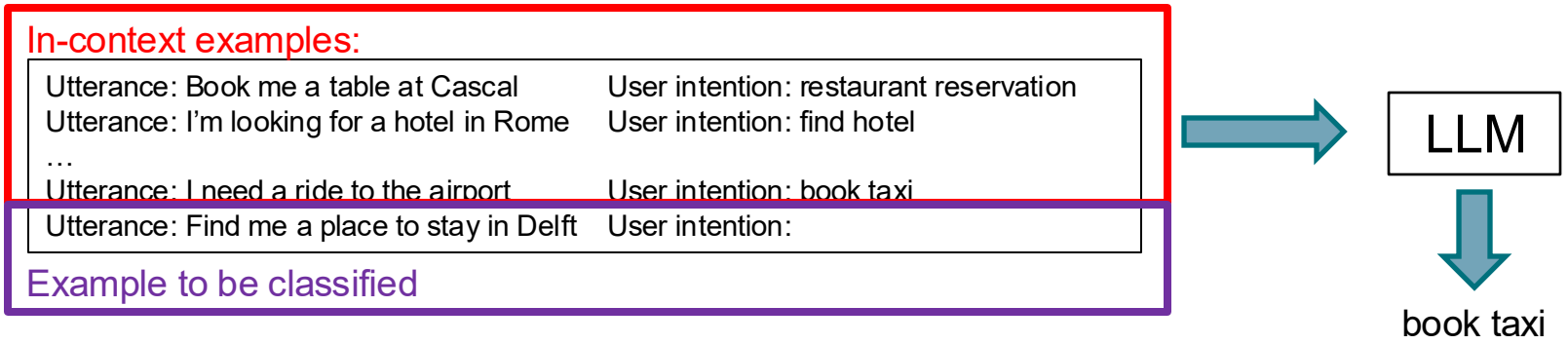
- The final answer can then be determined by any ensembling method, such as:
  - Averaging probabilities
  - Majority voting

Topics for Today

- Major Paradigms in NLP
- Prompting
- In-context Learning
- Prompting Examples

In-Context Learning (ICL)

- Key idea: Learning from analogy
- A few examples are included in the context of the prompt to provide demonstrations to the LLM.
- Example for user intent classification:



ICL – Pros and Cons

Pros:

- No training required, but existing training examples can be useful.
- It could be easy to hand-craft a few examples as demonstrations for many tasks.

Cons:

- With the addition of the examples, the prompt may get too long.

Limiting factor for ICL: the context length of LLM

ICL – Performance

- Adding examples/demonstrations in the context consistently improves performance:

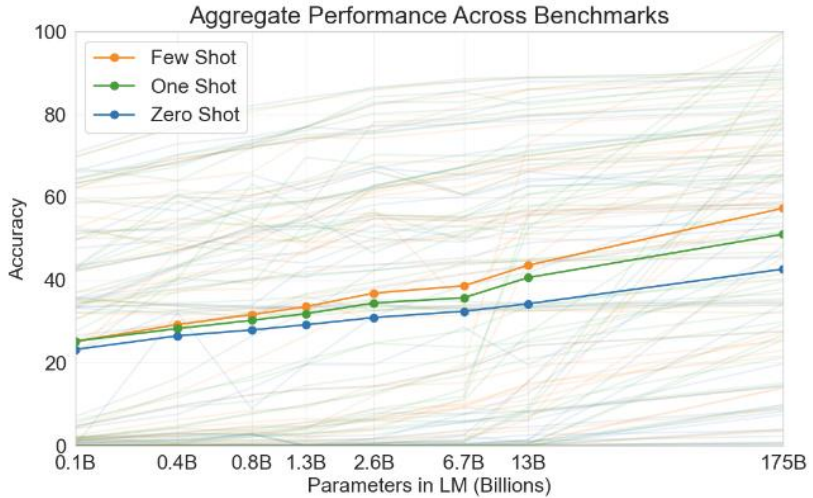


Figure 1.3: Aggregate performance for all 42 accuracy-denominated benchmarks While zero-shot performance improves steadily with model size, few-shot performance increases more rapidly, demonstrating that larger models are more proficient at in-context learning. See Figure 3.8 for a more detailed analysis on SuperGLUE, a standard NLP benchmark suite.

Figure from Brown et al., 2020

ICL – Things to Consider

- How many examples to put in context? → **Number of Examples**
- Which examples to put in context? → **Example Selection**
- What should be the order of examples? → **Example Ordering**
- Is it better to use ICL or fine-tuning? → **Methodology Decision**

ICL – Number of Examples in Context

- Earlier work, the gains can diminish after a few examples (Liang et al, TMLR, 2023)

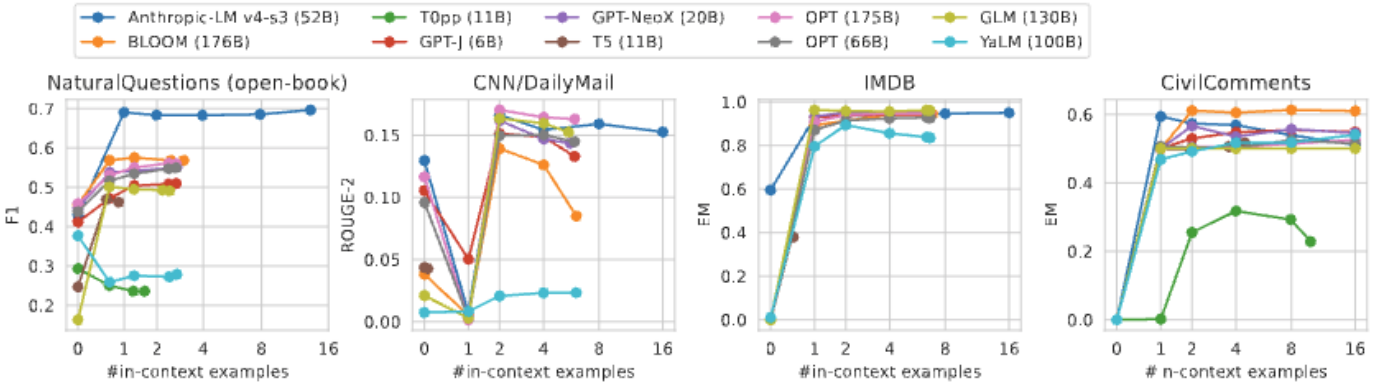


Figure 32: **Number of in-context examples.** For each model, we set the maximum number of in-context examples to [0, 1, 2, 4, 8, 16] and fit as many in-context examples as possible within the context window. We plot performance as a function of the average number of in-context examples actually used.

ICL – Number of Examples in Context (cont.)

- More recent work, with LLMs that have longer context sizes: Many-shot ICL consistently outperforms few-shot ICL (Agarwal et al., NeurIPS, 2024).

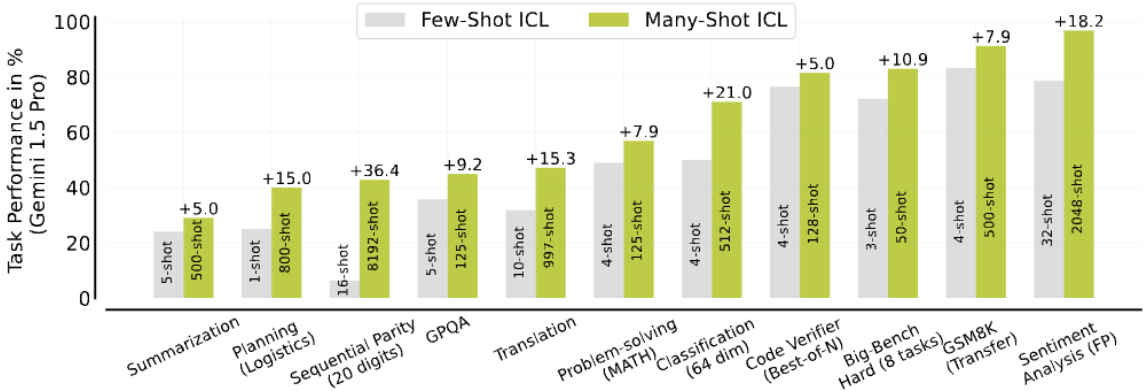


Figure 1 | **Many-shot vs Few-Shot In-Context Learning** (ICL) across several tasks. Many-shot ICL consistently outperforms few-shot ICL, particularly on difficult non-natural language tasks. Optimal number of shots for many-shot ICL are shown inside the bar for each task. For few-shot ICL, we either use typical number of shots used on a benchmark, for example, 4-shot for MATH, or the longest prompt among the ones we tested with less than the GPT-3 context length of 2048 tokens. Reasoning-oriented tasks, namely MATH, GSM8K, BBH, and GPQA use chain-of-thought rationales. For translation, we report performance on English to Bemba, summarization uses XLSum, MATH corresponds to the MATH500 test set, and sentiment analysis results are reported with semantically-unrelated labels. See §2, §3, and §4 for more details.

ICL – Example Selection

- Bertsch et al., NAACL, 2025
- Random
- BM24 or BERTscore to select similar examples to put in context

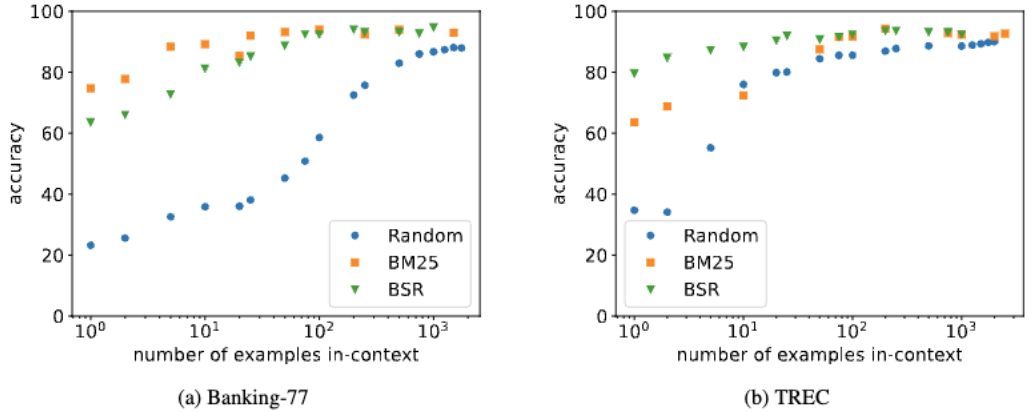


Figure 2: Comparing three selection methods– random selection, BM25, and BERTScore-Recall (BSR) on two representative datasets. At smaller numbers of demonstrations in-context, BM25 and BSR have differing performance, and the best retriever is dataset-specific; at larger demonstration counts, the two become indistinguishable. Both generally outperform random selection.

Llama-2 with 4k (Touvron et al., 2023), 32k (TogetherAI, 2023), and 80k (Fu et al., 2024) context windows, Mistral-7b-v0.2 (Jiang et al., 2023), Qwen 2.5-7B (Team, 2024).

Retrieving similar examples is better than randomly putting examples in context (especially when the context size is limited).

ICL – Example Ordering

- Zhao et al., PMLR, 2021
- LLMs can be very sensitive to the ordering of the examples in context.
- Recency bias

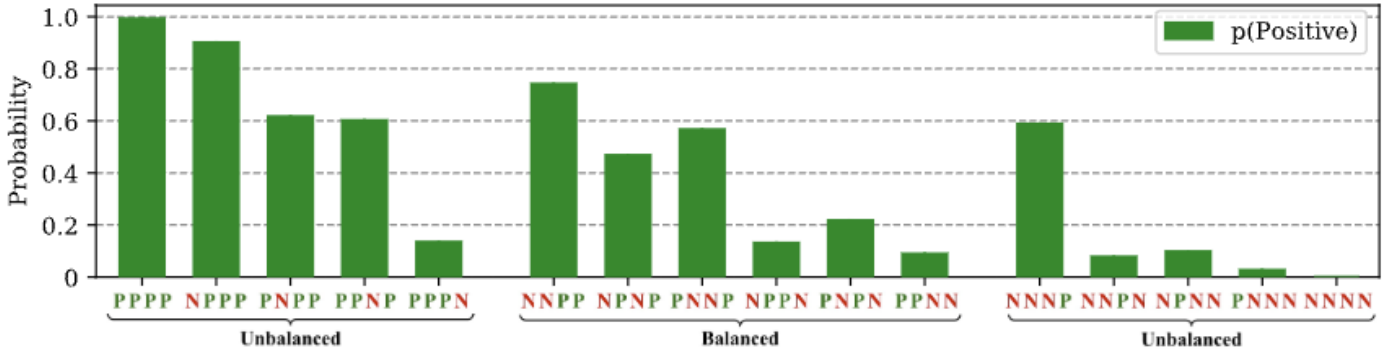


Figure 4. **Majority label and recency biases** cause GPT-3 to become biased towards certain answers and help to explain the high variance across different examples and orderings. Above, we use 4-shot SST-2 with prompts that have different class balances and permutations, e.g., [P P N N] indicates two positive training examples and then two negative. We plot how often GPT-3 2.7B predicts Positive on the balanced validation set. When the prompt is unbalanced, the predictions are unbalanced (*majority label bias*). In addition, balanced prompts that have one class repeated near the end, e.g., end with two Negative examples, will have a bias towards that class (*recency bias*).

ICL – Example Ordering (cont.)

- Bertsch et al., NAACL, 2025

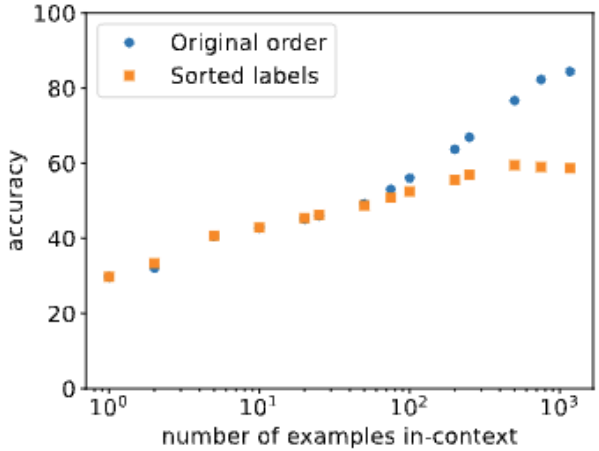


Figure 5: By contrast, sorting examples by label has an increasingly negative impact on performance in longer context regimes. Results on Llama2-32k with Clinic-150.

ICL – Example Ordering (cont.)

- Ordering examples based on their similarity to the input, where the most similar one comes right before the input (Liu et al., ACL DeeLIO, 2022).

- Ordering examples according to a curriculum, ranking examples from simple to more complex (Liu et al., Preprint, 2024)

- Algorithmic selections, such as DEmO, where ordering is made to achieve label fairness and influential prediction for each test instance (Guo et al., ACL Findings, 2024)

ICL – Methodology Decision

- Bertsch et al., NAACL, 2025

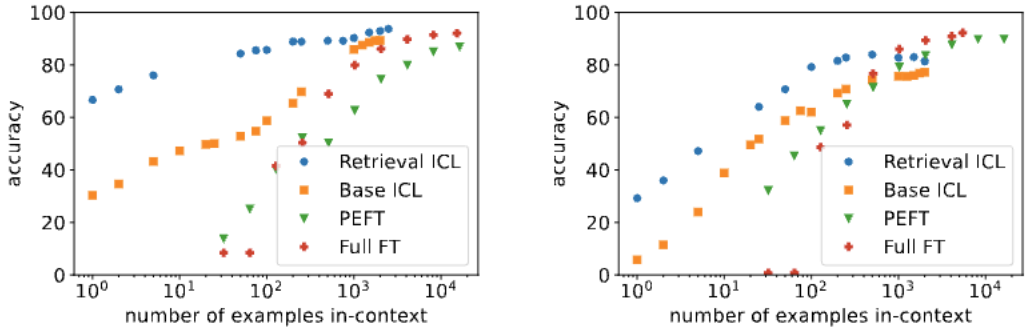


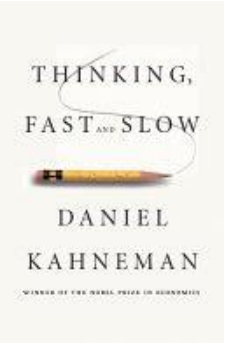
Figure 3: Comparing BM25 retrieval ICL, random selection ICL, and two types of finetuning on two representative datasets. Finetuning sometimes, but not always, exceeds ICL at high numbers of demonstrations. Note that, while retrieval ICL uses the listed number of examples in context, it assumes access to the larger test set to draw examples from (Perez et al., 2021). See Appendix C for results on other datasets.

When training data is scarce, ICL is a better choice than fine-tuning.

Topics for Today

- Major Paradigms in NLP
- Prompting
- In-context Learning
- Prompting Examples

Chain-of-Thought Prompting – Motivation



- Thinking fast and slow (D. Kahneman, 2011):
  - Human brain has two systems
  - System 1 is fast, instinctive, and emotional
    - Driving a car on an empty road
    - Reading billboard texts
  - System 2 is slow and calculating
    - Solving complex problems
- LLMs were shown to successfully perform system 1 tasks, but not system 2.
- Can LMs generate a coherent chain of thought before arriving at the answer, in a similar way human to reasoning?

CoT Prompting – Approach

In-context examples

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. ✓

Results on Math Problems

Models: LaMDA with 422M, 2B, 8B, 68B, 137B parameters.

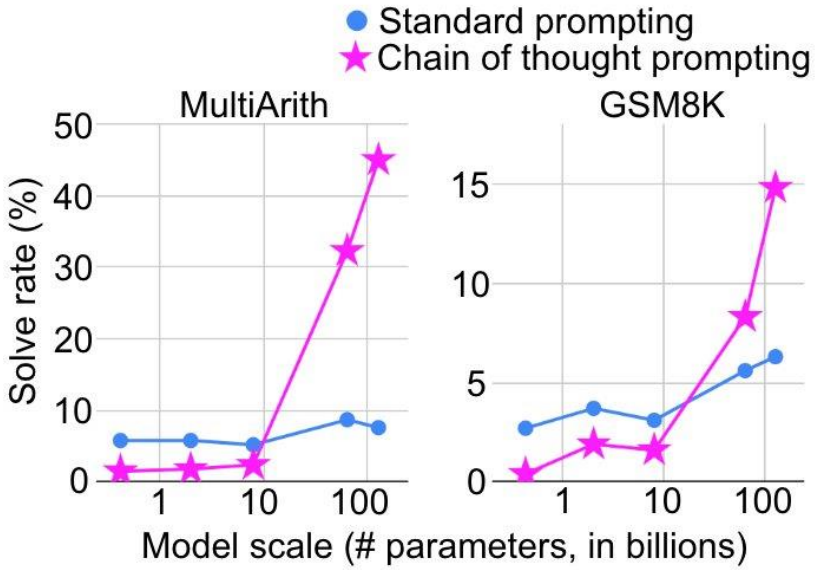


Table 1. Examples of correct and incorrect chains of thought produced by the 137B model on the GSM8K dataset. Even more examples are given in Table 5–Table 8 in the Appendix.

QUESTION: Ducks need to eat 3.5 pounds of insects each week to survive. If there is a flock of ten ducks, how many pounds of insects do they need per day?

MODEL ANSWER (CORRECT): Ducks need 3.5 pounds of insects each week. If there is a flock of 10 ducks, then they need 3.5 x 10 = 35 pounds of insects each week. If they need 35 pounds of insects each week, then they need 35 / 7 = 5 pounds of insects each day. The answer is 5. ✓

QUESTION: It takes Matthew 3 minutes to dig a small hole for shrubs and 10 minutes to dig a large hole for trees. How many hours will it take him to dig 30 small holes and 15 large holes?

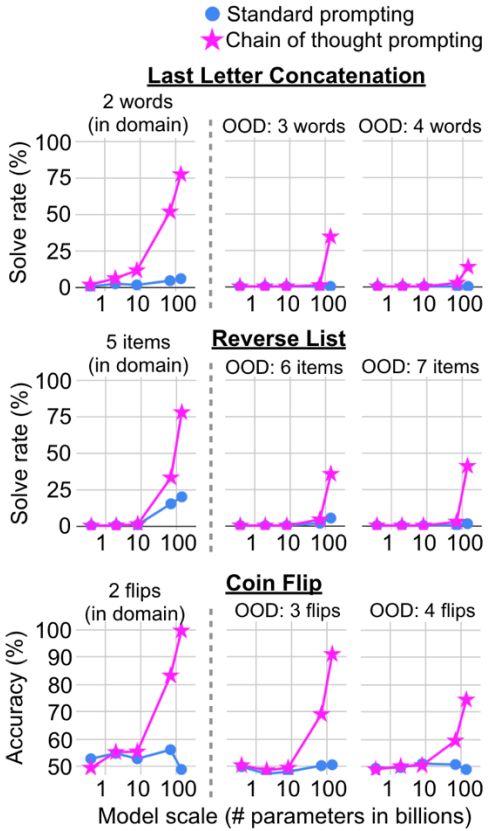
MODEL ANSWER (CORRECT): It takes Matthew 3 minutes to dig a small hole and 10 minutes to dig a large hole. So it takes Matthew 3 x 30 = 90 minutes to dig 30 small holes. It takes Matthew 10 x 15 = 150 minutes to dig 15 large holes. So it takes Matthew 90 + 150 = 240 minutes to dig 30 small holes and 15 large holes. 240 minutes is 4 hours. The answer is 4 hours. ✓

QUESTION: I have 10 liters of orange drink that are two-thirds water and I wish to add it to 15 liters of pineapple drink that is three-fifths water. But as I pour it, I spill one liter of the orange drink. How much water is in the remaining 24 liters?

MODEL ANSWER (INCORRECT): I have 10 liters of orange drink that are two-thirds water. I have 15 liters of pineapple drink that is three-fifths water. So I have 10 + 15 = 25 liters of drink. I spill one liter of orange drink. So I have 24 liters of drink. 25 - 24 is 1. The answer is 1. ✗

Results on Symbolic Reasoning Tasks

- Last Letter Concatenation
  - Elon Musk → nk
- Reverse List
  - "a", "b" → "b", "a"
- Coin Flip
  - “A coin is heads up. Phoebe flips the coin. Osvaldo does not flip the coin. Is the coin still heads up?” → “no”



Results on Commonsense Reasoning Tasks

COMMONSENSEQA

Q: Sammy wanted to go to where the people were. Where might he go?
Options:
(a) race track (b) populated areas (c) desert
(d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

STRATEGYQA

Q: Yes or no: Could a llama birth twice during War in Vietnam (1945-46)?

A: The War in Vietnam was 6 months. The gestation period for a llama is 11 months. So a llama could not give birth twice during the War in Vietnam. So the answer is no.

DATE UNDERSTANDING

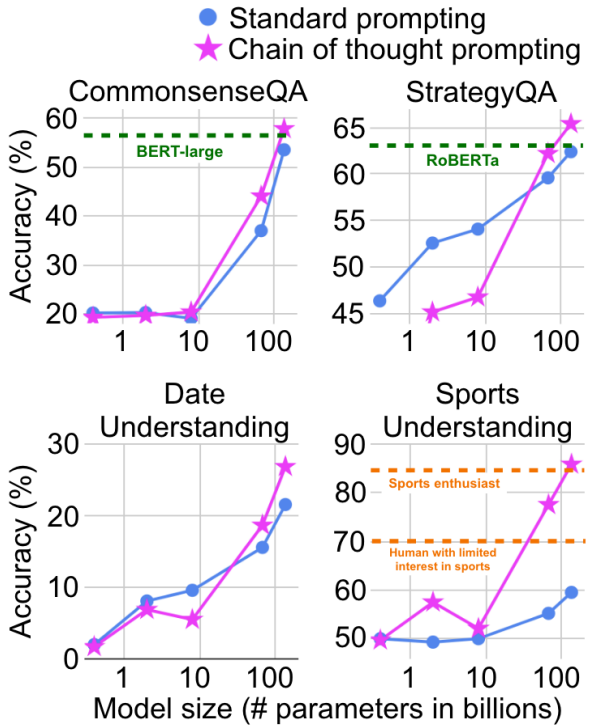
Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

SPORTS UNDERSTANDING

Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."

A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no.



CoT Prompting Summary

- Strong instruction following and reasoning ability in **large** language models.

- Reasoning improves model accuracy significantly in almost all cases tested.

- More on CoT and reasoning in a few weeks!

Self-Refine: Iterative Refinement with Self-Feedback

- Madaan et al., NeurIPS 2023.
- An LLM generates an initial output.
- Then, the same LLM provides feedback for its output.
- Then, the same LLM uses the feedback to refine itself, iteratively.

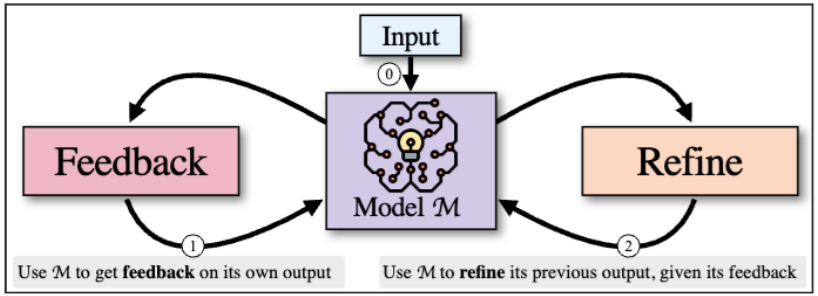


Figure 1: Given an input (⓪), SELF-REFINE starts by generating an output and passing it back to the same model $\mathcal{M}$ to get feedback (①). The feedback is passed back to $\mathcal{M}$, which refines the previously generated output (②). Steps (①) and (②) iterate until a stopping condition is met. SELF-REFINE is instantiated with a language model such as GPT-3.5 and does not involve human assistance.

Topics for Next Week

Tuesday

- Midterm Exam 1

Thursday

- Instruction Tuning
  - Overview
  - Instruction Tuning Datasets
  - Evaluation
  - Other Recent Instruction-Tuning Related Work