On Downstream Adaptation of Foundation Models

Guande He

TSAIL Group

2023.7.28

Today's Goal

- Get general pictures of downstream adaptation methods for foundation models.
  - Take pre-trained language models for example.
- Share some insightful research works on this topic.
  - Mainly focus on a "black box treatment" of language models.
  - From "I know it works" to "I have some idea on how/why it works" through theoretical analysis under some simplified settings.
  - Show some empirical evidence related to the theoretical analysis.

Pre-trained Language Models

- Given text sequences x from large corpora $\mathcal{X}$, we can learn a language model p_θ by self-supervised learning, which models some conditional probabilities:
- Autoregressive language modeling (GPT family):

$$\mathcal{L}_{\mathrm{ALM}}(\theta) = -\sum_{t=1}^{T} \log p_\theta(x_t | \boldsymbol{x}_{<t}).$$

- Masked language modeling (BERT family):

$$\mathcal{L}_{\mathrm{MLM}}(\theta) = -\frac{1}{K}\sum_{k=1}^{K} \log p_\theta(x_{\pi_k} | \boldsymbol{x}_{-\Pi}),$$

where $\Pi = \{\pi_1, \ldots, \pi_K\}$ is the mask indices.

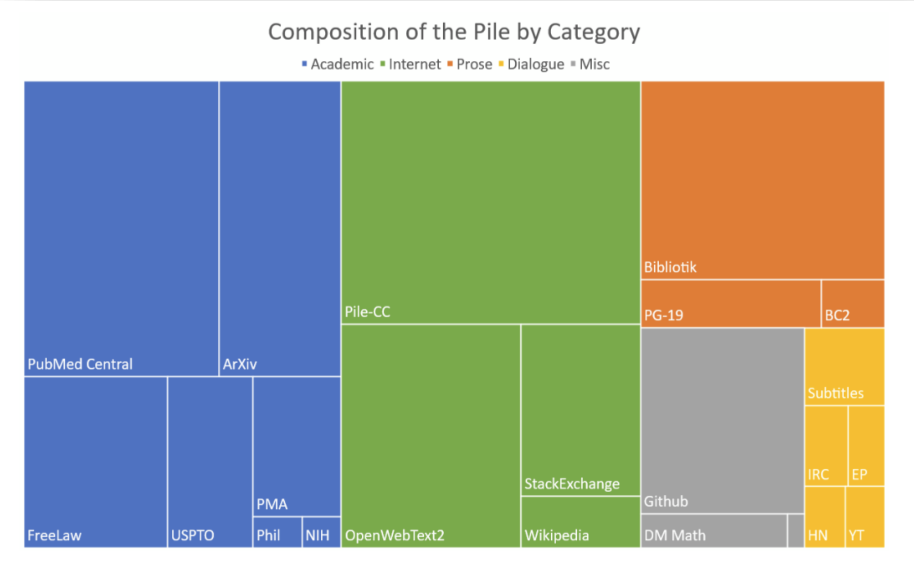
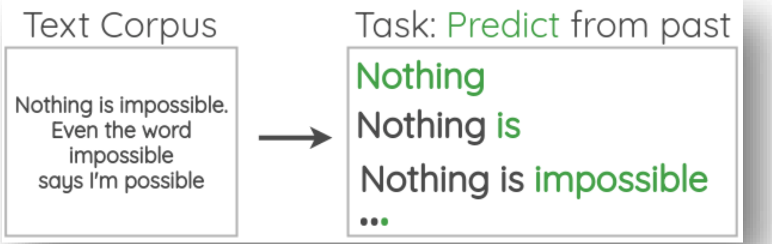
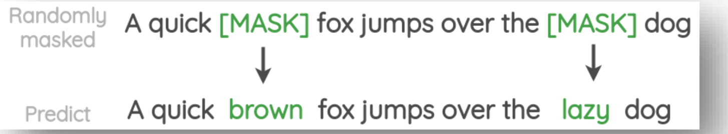


Figure 1: Treemap of Pile components by effective size.

“The Pile” Corpora (~1000GiB)



Autoregressive Language Modeling



Masked Language Modeling

Gao, Leo, et al. "The pile: An 800gb dataset of diverse text for language modeling." arXiv:2101.00027 (2020).
Amit, C. " Self Supervised Representation Learning in NLP. " (2020).

Downstream Adaption of PLMs

- In real-word scenarios, we might expect more than left-to-right completion $p_\theta(x_t|\boldsymbol{x}_{<t})$ or mask filling $p_\theta(x_k|\boldsymbol{x}_{-k})$.
- For example:
  - Text classification:

    Given a text sequence $\boldsymbol{x}$, we want to identify a particular attribute $y \in \mathcal{Y}$ corresponds to $\boldsymbol{x}$, e.g., spam filter, sentiment analysis.
  - Instruction following:

    Given a user instruction $\boldsymbol{x}$, we want the model to generate high-quality response $\boldsymbol{y} \in \mathcal{Y}$ that maximizes an unobserved human reward function $R: \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.
  - Other applications:

    information retrieval, summarization, controllable generation, etc.
- Intuitively, large language models have learned a rich set of linguistic features so it should be easily adapted to downstream tasks

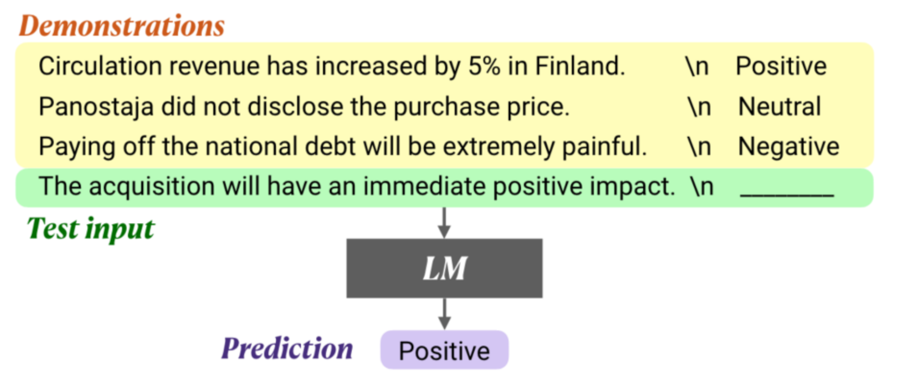
Downstream Adaption of PLMs

It works!

- There are two mainstream ways to adapt PLMs on downstream tasks:
- Fine-tuning (In-Weight Learning):
  - Gradient-based parameter updates.
  - Learn or "remember" class information during fine-tuning.
- In-Context Learning:
  - No parameter updates.
  - Learn with a concatenation of demonstrations.
- It turns out that downstream adaptation of PLMs is effective in terms of both quality and efficiency.
- Why these adaptations could work on PLMs?

	Rank	Name	Model
	1	JDExplore d-team	Vega v2
+	2	Liam Fedus	ST-MoE-32B
	3	Microsoft Alexander v-team	Turing NLR v5
	4	ERNIE Team - Baidu	ERNIE 3.0
	5	Yi Tay	PaLM 540B
+	6	Zirui Wang	T5 + UDG, Single Model (Google Brain)
+	7	DeBERTa Team - Microsoft	DeBERTa / TuringNLRv4
	8	SuperGLUE Human Baselines	SuperGLUE Human Baselines

Fine-tuned LMs outperform human baseline in SuperGLUE natural language understanding benchmark

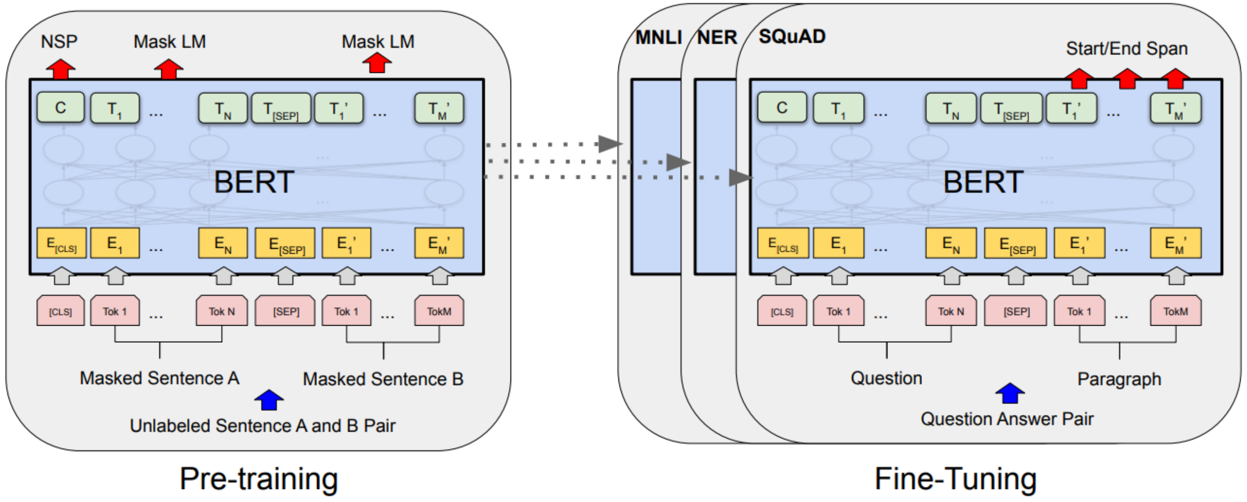


An example of in-context learning

Fine-tune a Pre-trained Masked Language Model

A rough picture on how fine-tuning works

- In the pre-training phase, the MLM first uses a transformer-based text encoder f to get hidden representations $g(\boldsymbol{x})$ of input sequence, then a language modeling head is applied on $g(\boldsymbol{x})$ to get the conditional probability $p_\theta(x_{\pi_k}|\boldsymbol{x}_{-\Pi})$.
- In the fine-tuning phase, a newly initialized classification head is applied to $g(\boldsymbol{x})$ and is jointly optimized with the LM using downstream data $(\boldsymbol{x}, y)$.



Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." (2018).

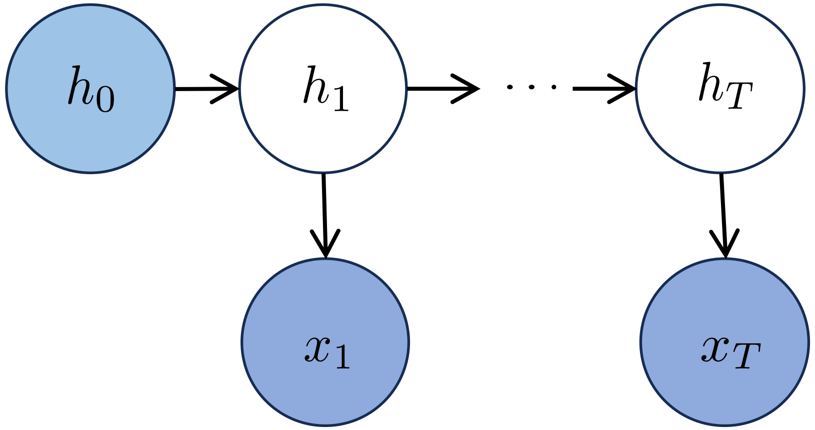
Why PLMs Help in Downstream Task?

- We could get some insights about why fine-tuning a pre-trained MLM is effective by analyzing a simplified setting:
  - Downstream task: text classification.
  - Fine-tuning methods: head tuning (optimize the classification head only).
  - Data generating distribution: Hidden Markov Model.
- Definitions:
  - **Pre-trained Model.** Assume our pre-trained masked language model could **perfectly** compute the conditional probability $G_i(\boldsymbol{x}) = p(x_i|\boldsymbol{x}_{-i}) \in \Delta^{|\mathcal{V}|}$, which is a probability vector over the vocabulary space $\mathcal{V}$.
  - **Downstream Task.** The downstream task contains paired data $(\boldsymbol{x}, F^\star(\boldsymbol{x})) \in \mathcal{X} \times \mathcal{Y}$, where $F^\star : \mathcal{X} \to \mathcal{Y}$ is the ground-truth mapping and $\mathcal{Y}$ is a discrete set of labels.
  - **Head Tuning.** Use a classification head f on top of fixed model outputs, e.g., $F(\boldsymbol{x}) = \mathbf{1}(f(G(\boldsymbol{x})) \geq 0)$ for classification.
    - **Remark.** In practice, the classifier is built on top of contextualized representations.

Wei, Colin, Sang Michael Xie, and Tengyu Ma. "Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning." (NeurIPS 2021).

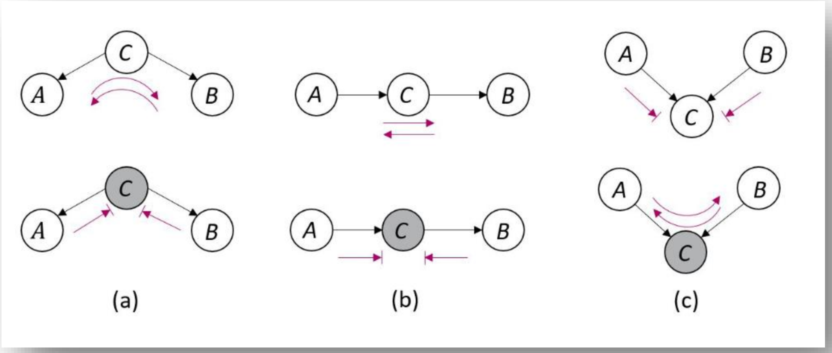
Review: Hidden Markov Model

- Hidden Markov Model (HMM) is a probabilistic graph model:



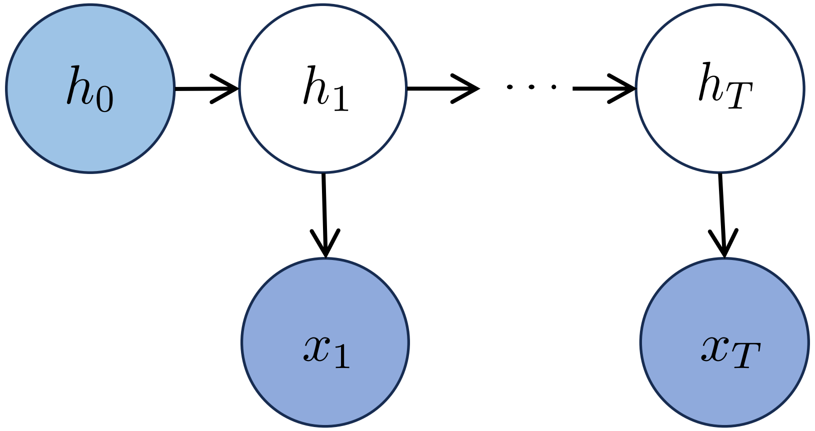
- Joint probability: $p(\boldsymbol{x}, \boldsymbol{h}) = p(h_0) \prod_{i=1}^{T} p(h_i | h_{i-1}) p(x_i | h_i)$

- Conditional independence: $h_{i-1} \perp h_{i+1} | h_i, \; x_i \perp \boldsymbol{x}_{i+1:T} | h_i$



Conditional independence in Bayesian network

Analysis with HMMs



- **Data distribution.**
  - Joint probability: $p(\boldsymbol{x}, \boldsymbol{h}) = p(h_0)\prod_{i=1}^{T} p(h_i|h_{i-1})p(x_i|h_i)$
  - We have time-invariant transition probability for all timesteps $i > 0$, i.e., $p(h_i|h_{i-1}) = A \in \mathbb{R}^{|\mathcal{H}|\times|\mathcal{H}|}$.
  - We have time-invariant token emission probability for all timesteps $i \geq 1$, i.e. $p(x_i|h_i) = W \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{H}|}$.
- **Downstream task.**
  - The ground-truth mapping is assumed to be a linear classifier on the **posterior** $p(h_0|\boldsymbol{x}_{1:T})$:
$$F^\star(\boldsymbol{x}) = \mathbf{1}(\mu^\top p(h_0|\boldsymbol{x}) \geq 0)$$
  - The downstream classifier is built on top of the **conditional probability** $G_i(\boldsymbol{x}) = p(x_i|\boldsymbol{x}_{-i})$:
$$F(\boldsymbol{x}) = \mathbf{1}(b^\top(G(\boldsymbol{x})) \geq 0)$$

Wei, Colin, Sang Michael Xie, and Tengyu Ma. "Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning." (NeurIPS 2021).

Bridge the Gap between Conditional Prob & Posterior (I)

- **Lemma 1.** If the Markov chain $\{h_0, \ldots, h_T\}$ is ergodic and $p(h_0)$ has full support, then for any timestep $t \geq 1$, there exists a diagonal matrix D such that for all sequence $\boldsymbol{v} \in \mathrm{supp}(p(\boldsymbol{x}))$,

$$p(h_i|\boldsymbol{x}_{i+1:i+T} = \boldsymbol{v}) = r_{\boldsymbol{v}} D p(h_0|\boldsymbol{x}_{1:T} = \boldsymbol{v})$$

where $r_{\boldsymbol{v}}$ is a positive scalar.

- **Proof.**

$$\begin{aligned}
p\left(h_i|\boldsymbol{x}_{i+1:T+i} = \boldsymbol{v}\right) &= \frac{p\left(\boldsymbol{x}_{i+1:T+i} = \boldsymbol{v}|h_i\right) \odot p\left(h_i\right)}{p\left(\boldsymbol{x}_{i+1:T+i} = \boldsymbol{v}\right)} \\
&= \frac{p\left(\boldsymbol{x}_{1:T} = \boldsymbol{v}|h_0\right) \odot p\left(h_0\right)}{p\left(\boldsymbol{x}_{i+1:T+1} = \boldsymbol{v}\right)} \odot \frac{p\left(h_i\right)}{p\left(h_0\right)} \quad \text{(by Markovian property of HMMs)} \\
&= p\left(h_0|\boldsymbol{x}_{1:T} = \boldsymbol{v}\right) \odot \frac{p\left(h_i\right)}{p\left(h_0\right)} \cdot \frac{p\left(\boldsymbol{x}_{1:T} = \boldsymbol{v}\right)}{p\left(\boldsymbol{x}_{i+1:T+i} = \boldsymbol{v}\right)}
\end{aligned}$$

Wei, Colin, Sang Michael Xie, and Tengyu Ma. "Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning." (NeurIPS 2021).

Bridge the Gap between Conditional Prob & Posterior (II)

- **Lemma 2.** Let U, V, Z be random variables such that $U \perp V \mid Z$. Then for any v, $P[U|V=v] = P[U|Z] \cdot P[Z|V=v]$. Thus, if $P[U|Z]$ has a left inverse $(P[U|Z])^\dagger$, then $P[Z|V=v] = (P[U|Z])^\dagger P[U|V=v]$.

- Recall that in HMM, we have $x_i \perp \boldsymbol{x}_{i+1:T} | h_i$, apply Lemma 2, let $\boldsymbol{x}' = [x_1', \boldsymbol{v}]$, then:

$$G_1(\boldsymbol{x}') = p(x_1' | \boldsymbol{x}_{2:T}' = \boldsymbol{v}) = W p(h_1 | \boldsymbol{x}_{2:T}' = \boldsymbol{v})$$

- If the token emission probability matrix W has linearly independent columns, then:

$$p(h_1 | \boldsymbol{x}_{2:T}' = \boldsymbol{v}) = W^\dagger G_1(\boldsymbol{x}')$$

- By Lemma 1, we have:

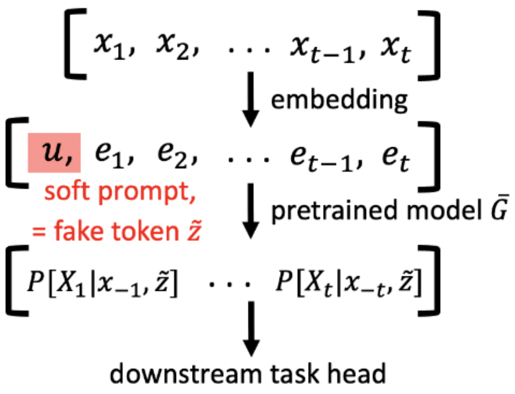
$$p(h_1 | \boldsymbol{x}_{2:T+1}' = \boldsymbol{v}) = r_{\boldsymbol{v}} D p(h_0 | \boldsymbol{x}_{1:T} = \boldsymbol{v})$$

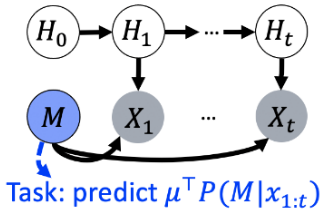
- Hence, we can recover the posterior-based ground-truth mapping using a linear classification head on top of a conditional probability.

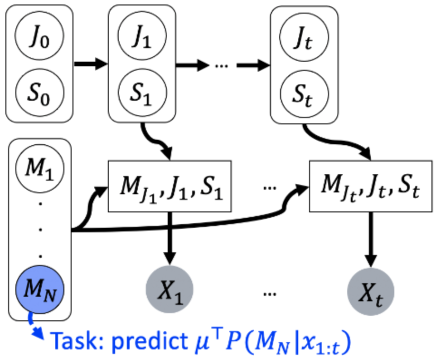
Wei, Colin, Sang Michael Xie, and Tengyu Ma. "Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning." (NeurIPS 2021).

Beyond (Linear) Head Tuning and HMM

- Under above setting, the full column rank assumption on $W \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{H}|}$ implies $|\mathcal{H}| \leq |\mathcal{V}|$, which is unrealistic because we usually adopt a large model to ensure its expressivity.
- This assumption can be further relaxed via:
  - More flexible tuning methods, e.g., soft prompt tuning.
  - More powerful data generating distribution, e.g., memory augmented HMM.



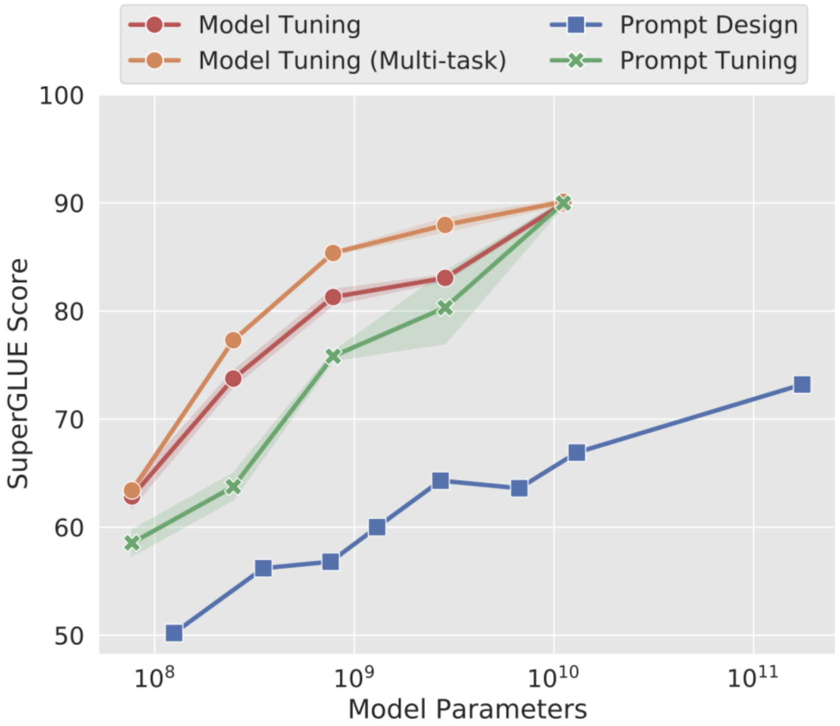




- You can refer to the original paper for further analysis.

Wei, Colin, Sang Michael Xie, and Tengyu Ma. "Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning." (NeurIPS 2021).

Empirical Evidence: Scaling Law of Soft Prompt Tuning



- Soft prompt tuning of T5 matches the quality of standard full parameter fine-tuning as size increases.

Lester, Brian, Rami Al-Rfou, and Noah Constant. "The power of scale for parameter-efficient prompt tuning." (EMNLP 2021).

Summary

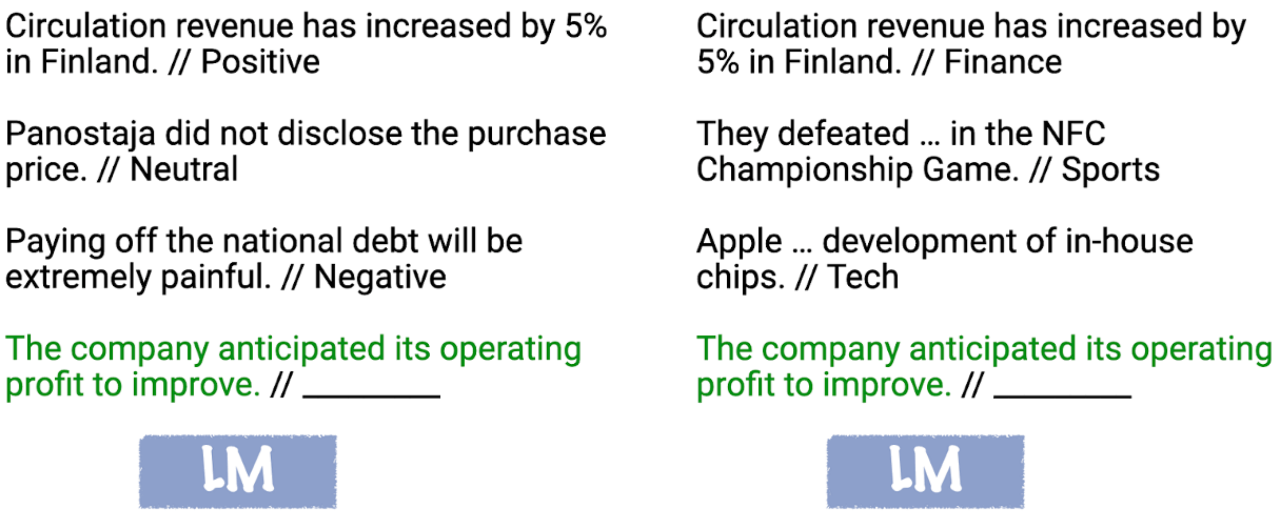
- Pre-training on large-scale datasets with self-supervised learning enables efficient and effective adaptation to downstream tasks.
- There exists some relationship between the self-supervised objective and the performance on downstream tasks, which is seemingly unrelated.

- Further Reading:
  - Closer look to relationship between pre-training loss and downstream performance:
    Liu, Hong, et al. "Same pre-training loss, better downstream: Implicit bias matters for language models." (ICML 2023).

In-Context Learning

An intriguing phenomenon

- **In-Context Learning (ICL)** was popularized in the original GPT-3 paper as an adaptation technique for larger language models to learn tasks given only a few examples.



Brown, Tom, et al. "Language models are few-shot learners." (NeurIPS 2020).
Xie, Sang Michael, and Min, Sewon "How does in-context learning work? A framework for understanding the differences from traditional supervised learning" (2022).

The Mystery of In-Context Learning

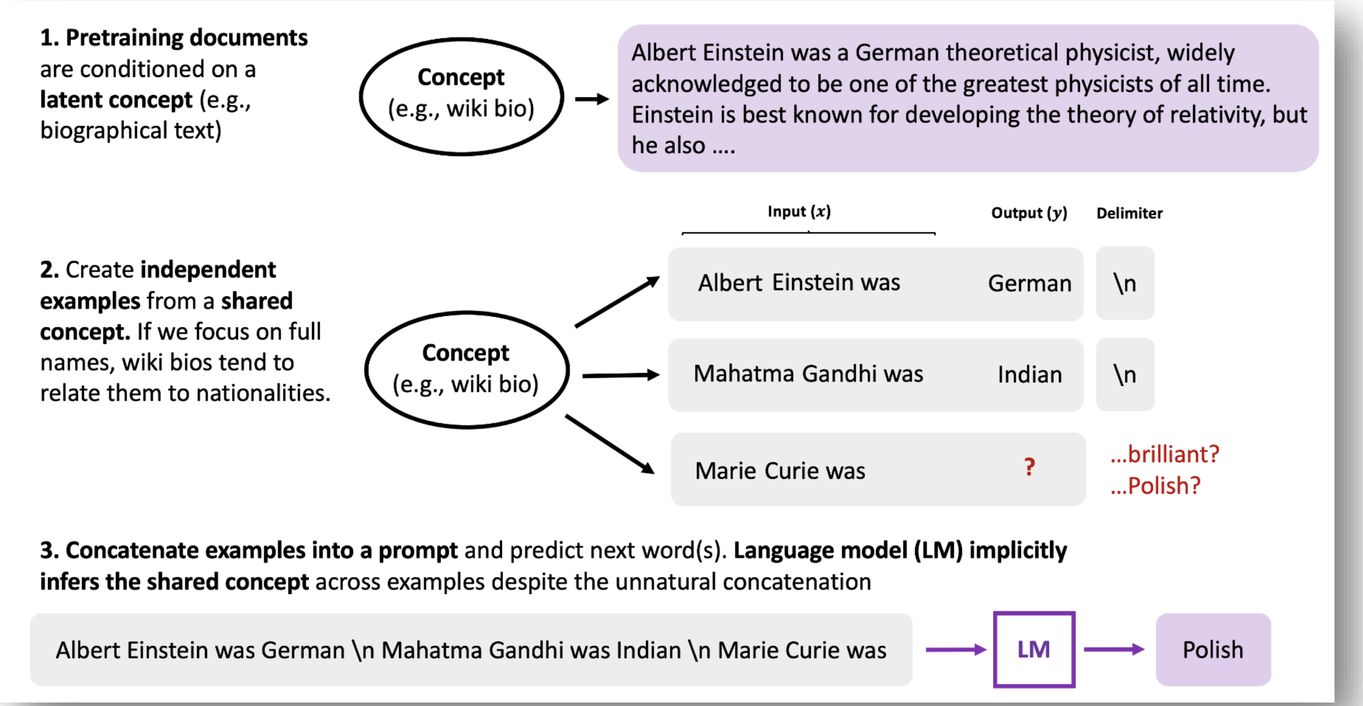
- What can ICL do?
  - On many NLP benchmarks, ICL is **competitive with supervised learning** using less labeled data.
  - ICL has enabled people to build new applications in just a few hours (prompt engineering).

- Why ICL surprising?
  - ICL **does not need any parameter updates**.
  - ICL just emerges from large PLMs, where the model did not explicitly learn with such pattern.

- What the model does when conducting ICL?
  - Indexing into a vast set of known tasks from the training data?
  - Learning new tasks from in-context examples at inference time?

Xie, Sang Michael, and Min, Sewon "How does in-context learning work? A framework for understanding the differences from traditional supervised learning" (2022).

A Framework for ICL as Bayesian Inference



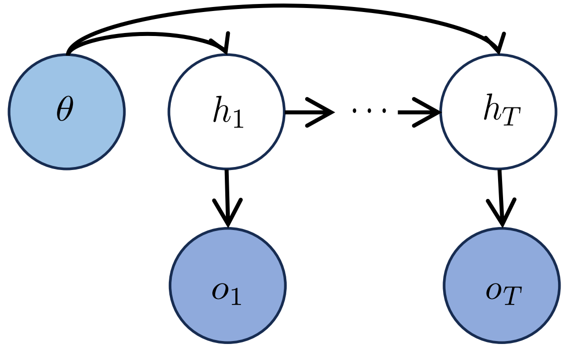
- If the LM fits the pretraining distribution with enough data and expressivity, the question of ICL becomes matching $p(\text{output}|\text{prompt})$ under pretraining distribution and a different distribution p_{prompt} via marginalization:

$$p(\text{output}|\text{prompt}) = \int_{\text{concept}} p(\text{output}|\text{concept}, \text{prompt})p(\text{concept}|\text{prompt})d(\text{concept})$$

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference
Formalizing ICL

- **Pretraining distribution.**
  - A latent concept (task) θ from a family of concepts Θ defines a distribution $p\left(o_1, \ldots, o_T \mid \theta\right)$ over observed tokens o from a vocabulary $\mathcal{V}$.
  - Document generation:
    - Sample $\theta \sim p(\theta)$.
    - Generate the document by $p\left(o_1, \ldots, o_T \mid \theta\right)$, which is defined by a HMM. The concept θ determines the transition probability matrix of HMM between $h_1, \ldots, h_T$ from a hidden state set $\mathcal{H}$.
  - Pretraining: $p\left(o_1, \ldots, o_T\right)=\int_{\theta \in \Theta} p\left(o_1, \ldots, o_T \mid \theta\right) p(\theta) d \theta$



Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference

Formalizing ICL

- **In-Context Prompts.**
  - A prompt example composes an input sequence x and an output token y.
  - The prompts is a concatenation of n independent training examples and a test input $\boldsymbol{x}_{\text{test}}$, which are all conditioned on a shared prompt concept θ^*. The goal is to predict y_{test} .
  - Prompt generation:
    - Generate a start hidden state h_i^{start} from a prompt start distribution p_{prompt}.
    - Given h_i^{start}, generate the example sequence $O_i = [\boldsymbol{x}_i, y_i]$ from the pretraining distribution $p(O_i | h_i^{\text{start}}, \theta^*)$ conditioned on θ^*.
    - A special delimiter token o^{delim} is used to split these examples.
    - The prompt can be written as:

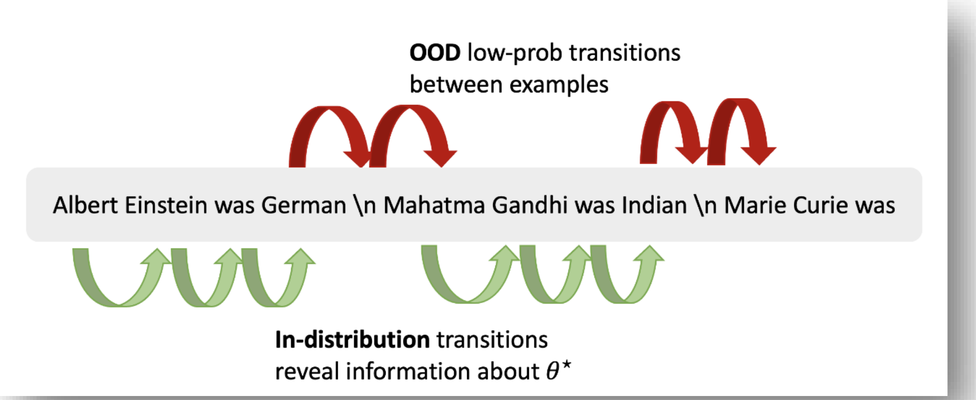
$$[S_n, \boldsymbol{x}_{\text{test}}] = [\boldsymbol{x}_1, y_1, o^{\text{delim}}, \boldsymbol{x}_2, y_2, o^{\text{delim}}, \ldots, \boldsymbol{x}_n, y_n, o^{\text{delim}}, \boldsymbol{x}_{\text{test}}] \sim p_{\text{prompt}}$$

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference

Formalizing ICL

- There exists a mismatch between prompt and pretraining distributions:
  - The transition between ICL examples has low probability in the pretraining distribution.
  - The choice of o^{delim} can also be a source of mismatch.



- Under such mismatch, LLMs can correctly infer the prompt concept from examples.
  - Ground Truth:

$$y_{\text{test}} \sim p_{\text{prompt}}\left(y \mid \boldsymbol{x}_{\text{test}}\right) = \mathbb{E}_{h_{\text{test}}^{\text{start}} \sim p_{\text{prompt}}\left(h_{\text{test}}^{\text{start}} \mid \boldsymbol{x}_{\text{test}}\right)}\left[p\left(y \mid \boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right)\right]$$

  - In-Context predictor: $f_n\left(x_{\text{test}}\right) = \arg\max_y p\left(y \mid S_n, \boldsymbol{x}_{\text{test}}\right)$

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference
High-level Approach

- Our goal is to show $\arg\max_y p\left(y|S_n, \boldsymbol{x}_{\text{test}}\right) \to \arg\max_y p_{\text{prompt}}\left(y|\boldsymbol{x}_{\text{test}}\right)$ as n grows.
- Expanding $p\left(y|S_n, \boldsymbol{x}_{\text{test}}\right)$:

$$p\left(y|S_n, \boldsymbol{x}_{\text{test}}\right) = \int_\theta p\left(y|S_n, \boldsymbol{x}_{\text{test}}, \theta\right) p\left(\theta|S_n, \boldsymbol{x}_{\text{test}}\right) d\theta$$

$$\propto \int_\theta p\left(y|S_n, \boldsymbol{x}_{\text{test}}, \theta\right) p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta\right) p(\theta) d\theta \quad \left(\text{Bayes' rule, drop the constant } \frac{1}{p\left(S_n, \boldsymbol{x}_{\text{test}}\right)}\right)$$

$$\propto \int_\theta \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta\right) p\left(h_{\text{test}}^{\text{start}}|S_n, \boldsymbol{x}_{\text{test}}, \theta\right) \frac{p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta\right)}{p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta^*\right)} p(\theta) d\theta$$

$$\text{(Law of total prob, Markov property, divide by } p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta^*\right) \text{ (a constant))}$$

$$= \int_\theta \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta\right) p\left(h_{\text{test}}^{\text{start}}|S_n, \boldsymbol{x}_{\text{test}}, \theta\right) \exp\left(n \cdot r_n(\theta)\right) p(\theta) d\theta$$

where $r_n(\theta) = \frac{1}{n} \log \frac{p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta\right)}{p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta^*\right)}$.

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference
High-level Approach

- Goal: $\arg\max_y p\left(y|S_n, \boldsymbol{x}_{\text{test}}\right) \rightarrow \arg\max_y p_{\text{prompt}}\left(y|\boldsymbol{x}_{\text{test}}\right)$
- Expanding $p\left(y|S_n, \boldsymbol{x}_{\text{test}}\right)$:

$$p(y|S_n, \boldsymbol{x}_{\text{test}}) \propto \int_\theta \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta\right) p\left(h_{\text{test}}^{\text{start}}|S_n, \boldsymbol{x}_{\text{test}}, \theta\right) \frac{p(S_n, \boldsymbol{x}_{\text{test}}|\theta)}{p(S_n, \boldsymbol{x}_{\text{test}}|\theta^*)} p(\theta) d\theta$$

- If $\frac{p(S_n, x_{\text{test}}|\theta)}{p(S_n, x_{\text{test}}|\theta^*)} \rightarrow 0$ for all concepts θ except the prompt concept θ^*, then the prompt concept θ^* is "selected" as a consequence of Bayesian inference.

- New goal:
  - Concept selection: show the average likelihood ratio $r_n(\theta) = \frac{1}{n} \log \frac{p(S_n, \boldsymbol{x}_{\text{test}}|\theta)}{p(S_n, \boldsymbol{x}_{\text{test}}|\theta^*)}$ converges to a negative constant for all $\theta \neq \theta^*$.
  - Same prediction under θ^*:

$$\arg\max_y \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right) p(h_{\text{test}}^{\text{start}}|S_n, \boldsymbol{x}_{\text{test}}, \theta^*) \rightarrow \arg\max_y p_{\text{prompt}}\left(y|\boldsymbol{x}_{\text{test}}\right)$$

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference

Heuristic derivation

- A main technical challenge in this setting is:
  - First, the in-context examples $O_i = [\boldsymbol{x}_i, y_i]$ are *i.i.d.*
  - However, they are dependent w.r.t. the pretraining distribution in ICL.
- Under some assumptions on bounded $p(\boldsymbol{x}_{\text{test}}|S_n,\theta)$ and $p(h^{\text{delim}}|\theta)$, we can perform factorization with constant error per sample:

$$p\left(S_n, \boldsymbol{x}_{\text{test}}|\theta\right) = p\left(\boldsymbol{x}_{\text{test}}|S_n,\theta\right) p\left(S_n|\theta\right) \approx \prod_{i=1}^{n} O(1) p\left(O_i|\theta\right)$$

- Then:

$$r_n(\theta) \leq \frac{1}{n}\left(O(n) + \sum_{i=1}^{n} \log \frac{p(O_i|\theta)}{p(O_i|\theta^*)}\right) \to O(1) + \mathbb{E}_{O\sim p_{\text{prompt}}}\left[\log \frac{p(O|\theta)}{p(O|\theta^*)}\right]$$

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference
Heuristic derivation

- From $r_n(\theta) \leq \frac{1}{n}\left(O(n) + \sum_{i=1}^{n} \log \frac{p(O_i|\theta)}{p(O_i|\theta^*)}\right) \to O(1) + \mathbb{E}_{O \sim p_{\text{prompt}}}\left[\log \frac{p(O|\theta)}{p(O|\theta^*)}\right]$

- The expectation can be decomposed to two KL terms:

$$\mathbb{E}_{O \sim p_{\text{prompt}}}\left[\log \frac{p(O|\theta)}{p(O|\theta^*)}\right] = \underbrace{D_{\text{KL}}(p_{\text{prompt}}(O) \| p(O|\theta^*))}_{O(1) \text{ error term}} - \underbrace{D_{\text{KL}}(p_{\text{prompt}}(O) \| p(O|\theta))}_{\text{KL term}}$$

- When KL term $>$ Error term for all $\theta \neq \theta^*$, we will get $\frac{p(S_n, x_{\text{test}}|\theta)}{p(S_n, x_{\text{test}}|\theta^*)} \to 0$.

- The prompt should provide enough signal (distinguishability) for Bayesian inference.

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

A Framework for ICL as Bayesian Inference

Heuristic derivation

- Last piece:

$$\arg\max_y \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right) p(h_{\text{test}}^{\text{start}}|S_n, \boldsymbol{x}_{\text{test}}, \theta^*) \to \arg\max_y p_{\text{prompt}}\left(y|\boldsymbol{x}_{\text{test}}\right)$$

- Expanding $p_{\text{prompt}}\left(y|\boldsymbol{x}_{\text{test}}\right)$:

$$p_{\text{prompt}}\left(y|\boldsymbol{x}_{\text{test}}\right) = \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right) p_{\text{prompt}}(h_{\text{test}}^{\text{start}}|\boldsymbol{x}_{\text{test}})$$

$$\propto \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right) p(\boldsymbol{x}_{\text{test}}|h_{\text{test}}^{\text{start}}, \theta^*) p_{\text{prompt}}(h_{\text{test}}^{\text{start}})$$

- Expanding LHS:

$$\text{LHS} = \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right) p(h_{\text{test}}^{\text{start}}|S_n, \boldsymbol{x}_{\text{test}}, \theta^*)$$

$$\propto \sum_{h_{\text{test}}^{\text{start}} \in \mathcal{H}} p\left(y|\boldsymbol{x}_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*\right) p(\boldsymbol{x}_{\text{test}}|h_{\text{test}}^{\text{start}}, \theta^*) p(h_{\text{test}}^{\text{start}}|S_n, \theta^*)$$

- Same argmax when the difference between $p_{\text{prompt}}(h_{\text{test}}^{\text{start}})$ and $p(h_{\text{test}}^{\text{start}}|S_n, \theta^*)$ is moderate.

Xie, Sang Michael, et al. "An explanation of in-context learning as implicit bayesian inference." (ICLR 2022).

Summary

- In-context examples provide noisy evidence for Bayesian inference.
  - The input distribution, label distribution and input-output mapping all provide signal for Bayesian inference.

- ICL is robust to some noise.
  - With a strong signal, some forms of noise (e.g., low-prob transitions between examples, removed input-output mapping) could be tolerable.

Circulation revenue has increased by 5% in Finland. // Finance

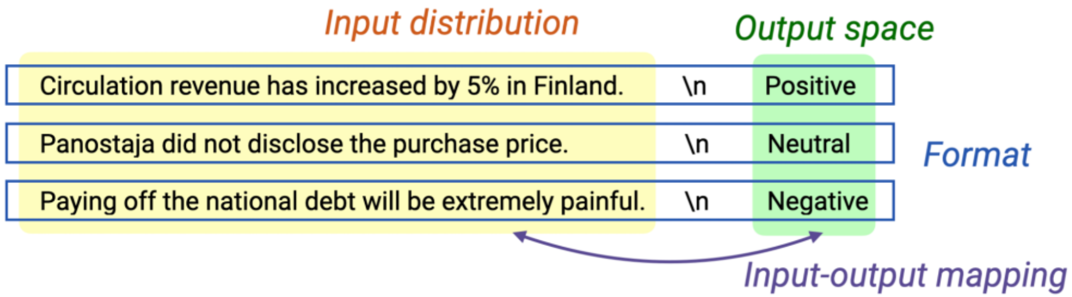
They defeated … in the NFC Championship Game. // Sports

Apple … development of in-house chips. // Tech

Xie, Sang Michael, and Min, Sewon "How does in-context learning work? A framework for understanding the differences from traditional supervised learning" (2022).

Empirical Evidence: Investigating ICL's Components

- Typical in-context examples consists of 4 components:

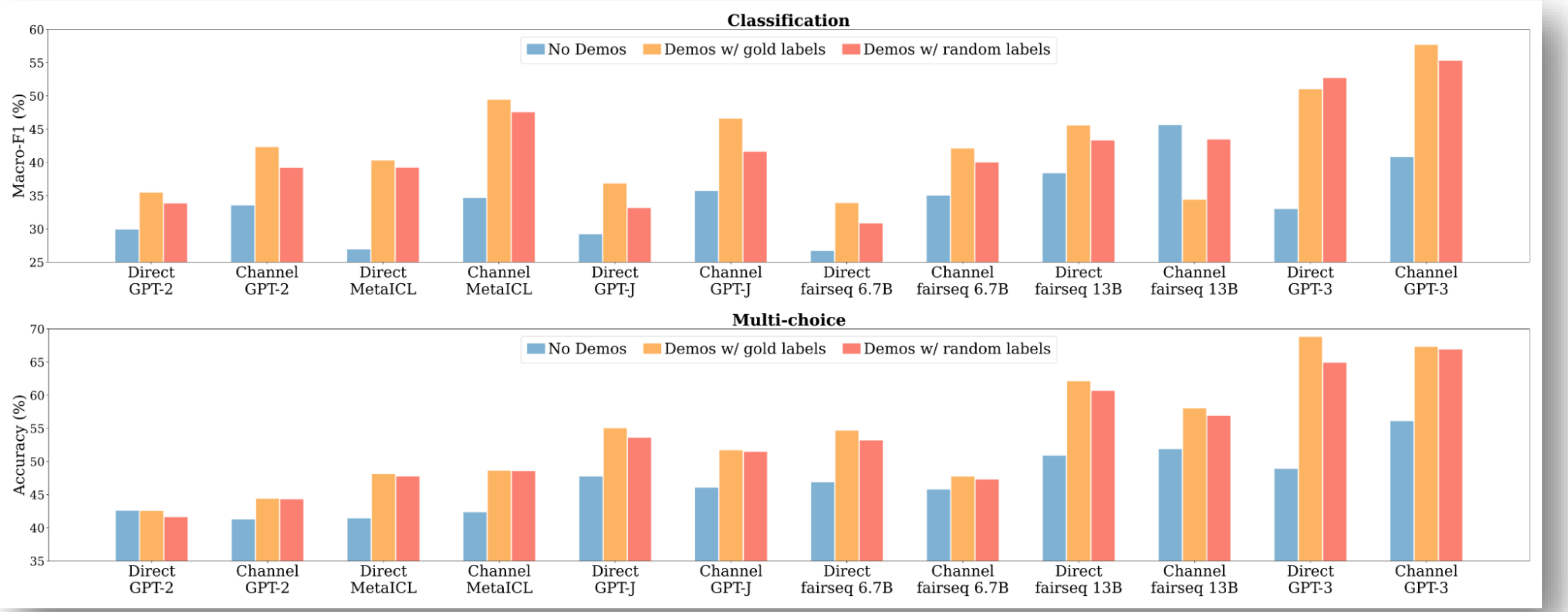


- Examine the role of input-output mapping by:
  - Zero-Shot learning
  - Examples with ground-truth outputs
  - Examples with random outputs

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

Effect of Input-Output Mapping

- Results of models whose sizes range from 774M to 175B

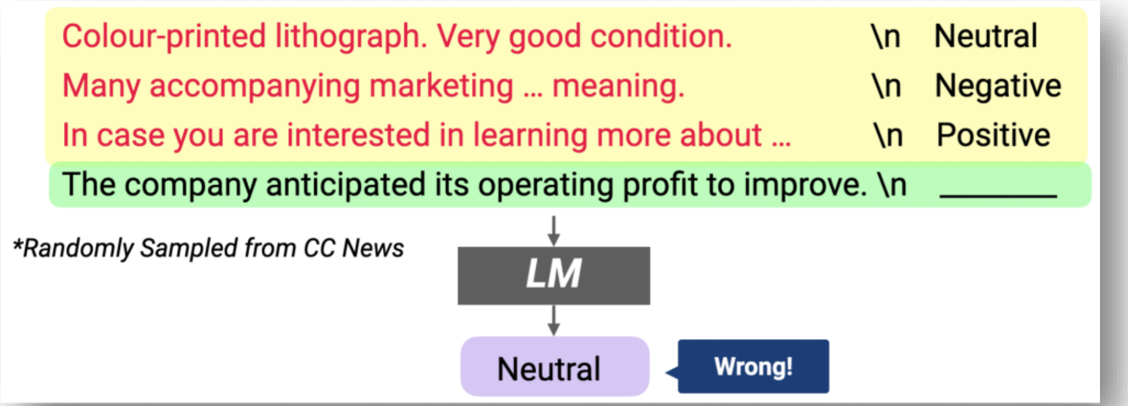


- Correct input-output mapping has a marginal effect on ICL (with implications).

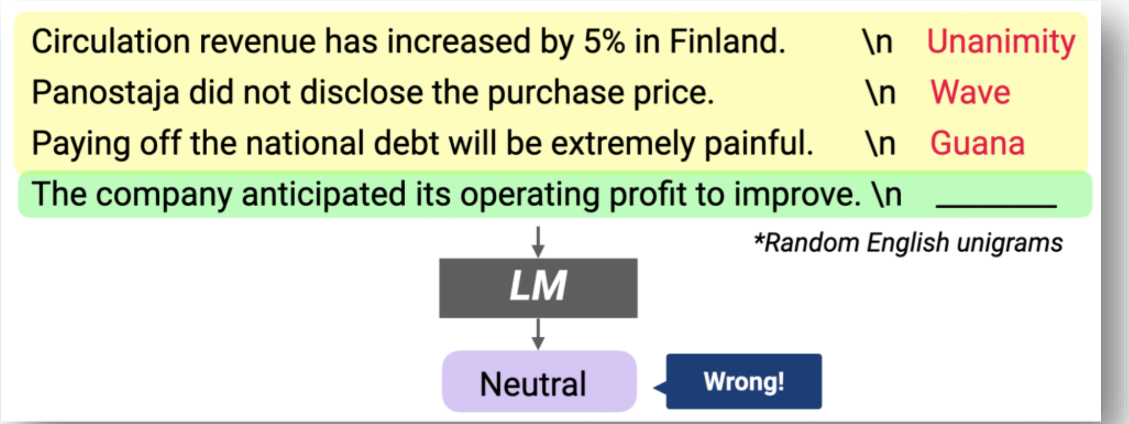
Min, Sewon, et al. "Rethinking the Role of Demonstrations: What Makes In-Context Learning Work?." (EMNLP 2022).

Effect of Input and Label Space

- The input distribution and the label space of in-context examples matter.



Replace the prompt input with random inputs from an external corpus

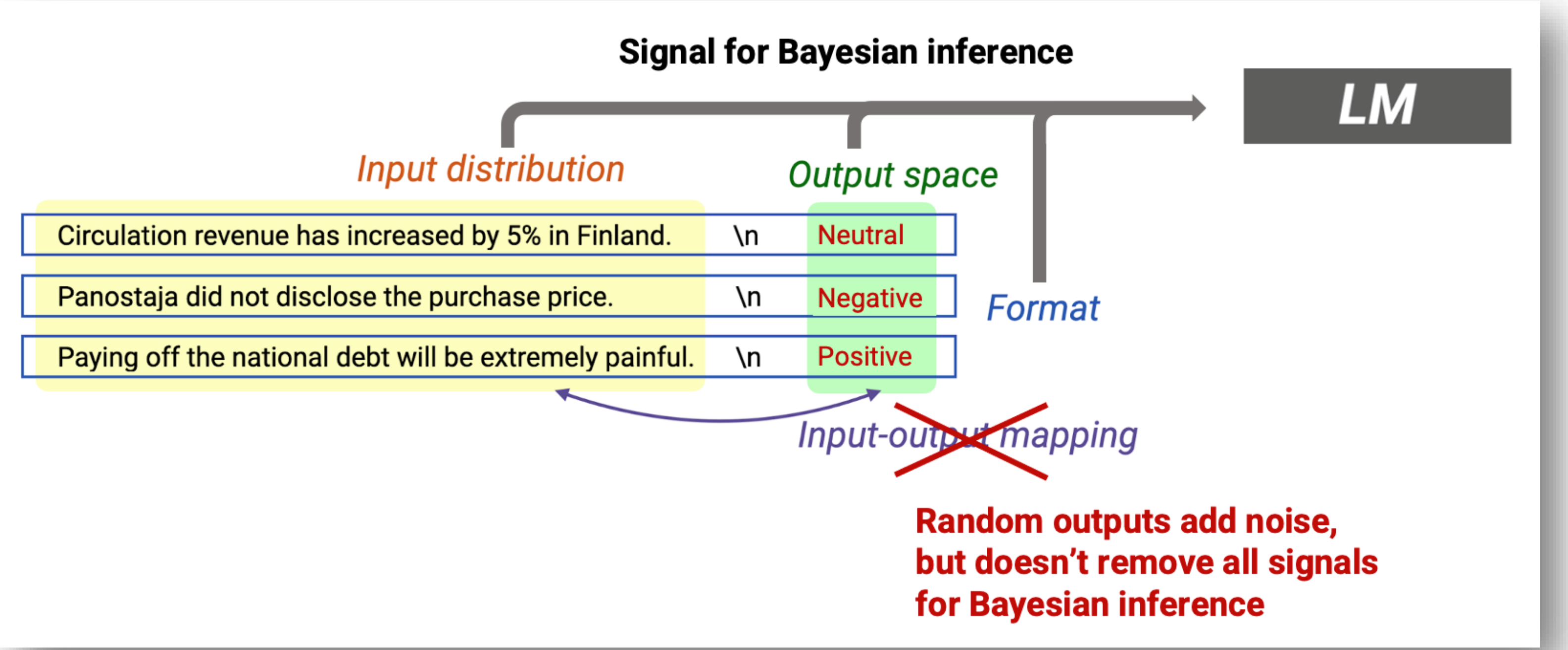


Replace the prompt label with random English unigrams

- Both changes can lead to a significant performance drop.

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

A Framework for ICL as Bayesian Inference

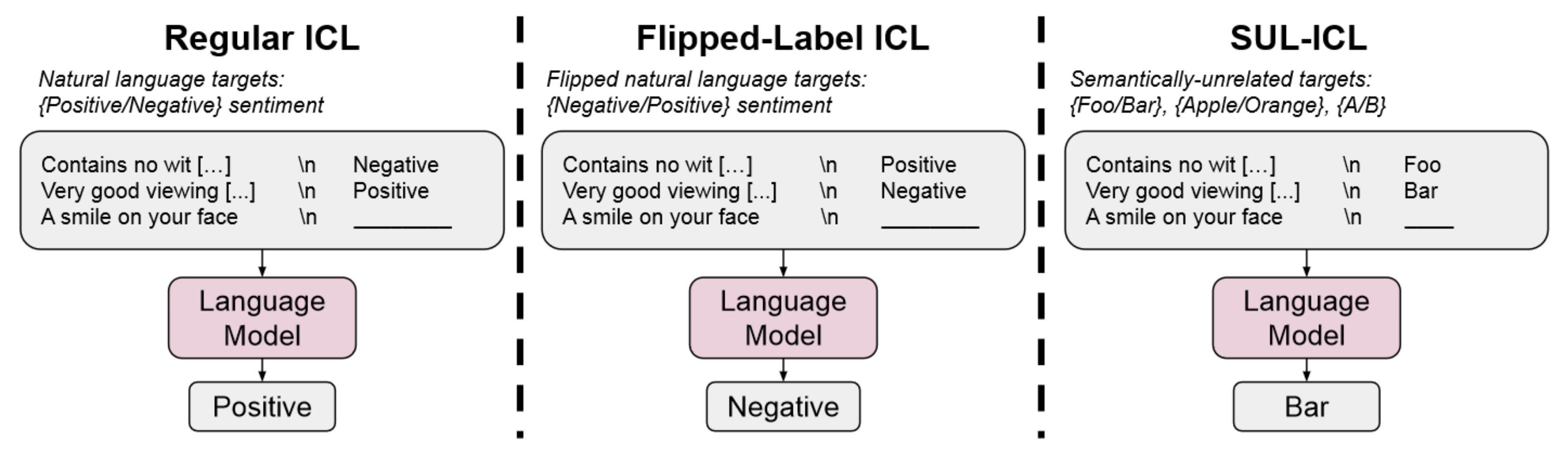


- Is the story ends?

Xie, Sang Michael, and Min, Sewon "How does in-context learning work? A framework for understanding the differences from traditional supervised learning" (2022).

Different Story for **Larger** LMs

- To successfully perform ICL, models can
  - Mostly use semantic prior knowledge to predict labels while following the format of in-context examples
  - Learn the input-label mappings from examples (overriding semantic prior or only exploit the input-output mapping).
- Study how semantic priors and input-label mappings interact in several experimental settings.



Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

Experiment Setting

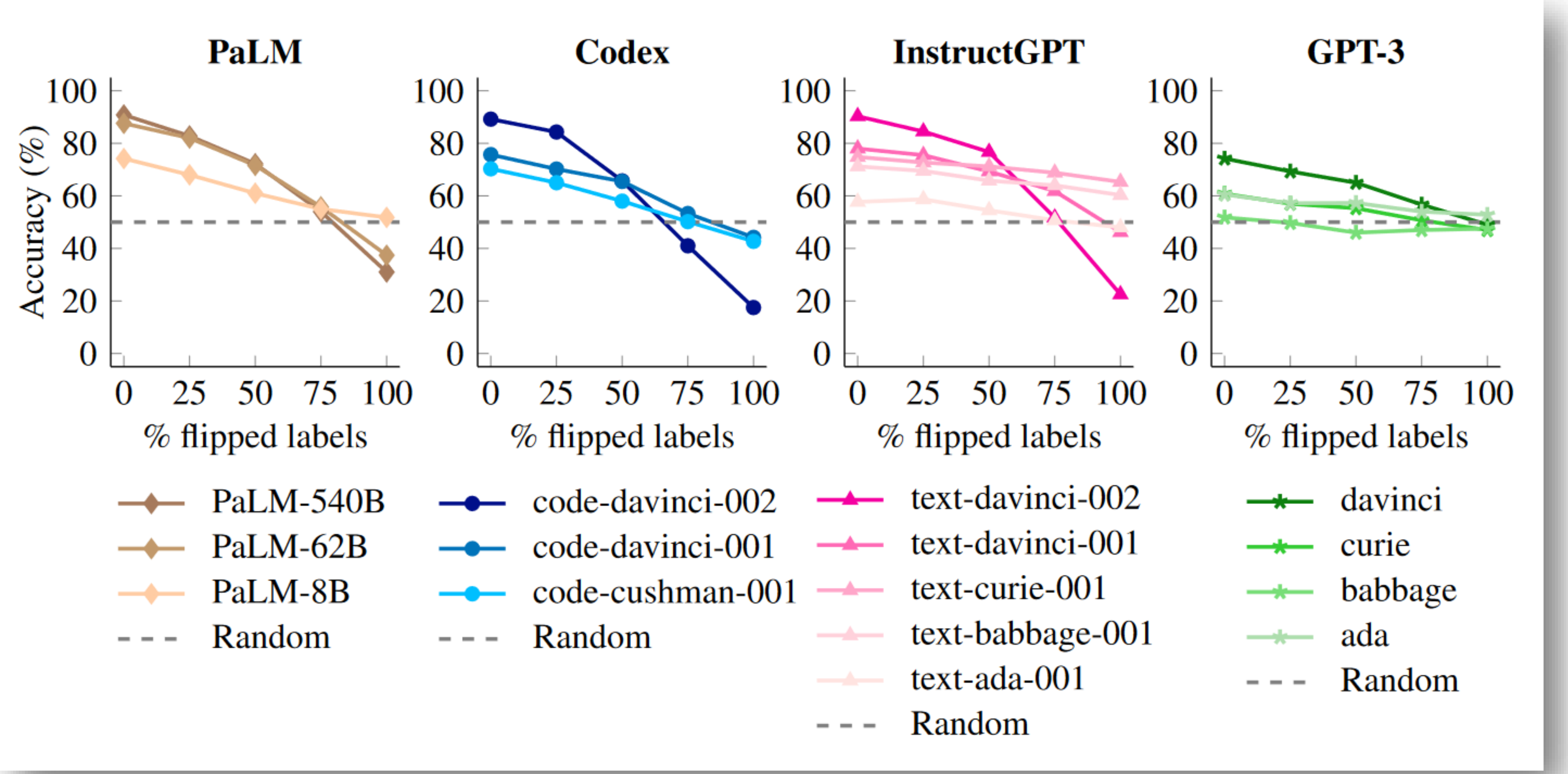
- Tasks: standard NLP classification datasets
- Models: ranging 350M ~ 540B, w/ and w/o instruct tuning.
- Use 16 context examples for each dataset.
- Use 100 randomly sampled evaluation examples per dataset.

Model Family	Model Name (Abbreviation)
GPT-3	ada (a), babbage (b), curie (c), davinci (d)
InstructGPT	text-ada-001 (a-1), text-babbage-001 (b-1), text-curie-001 (c-1), text-davinci-001 (d-1), text-davinci-002 (d-2)
Codex	code-cushman-001 (c-c-1), code-davinci-001 (c-d-1), code-davinci-002 (c-d-2)
PaLM	PaLM-8B, PaLM-62B, PaLM-540B
Flan-PaLM	Flan-PaLM-8B, Flan-PaLM-62B, Flan-PaLM-540B

Table 1: Models used in this paper.

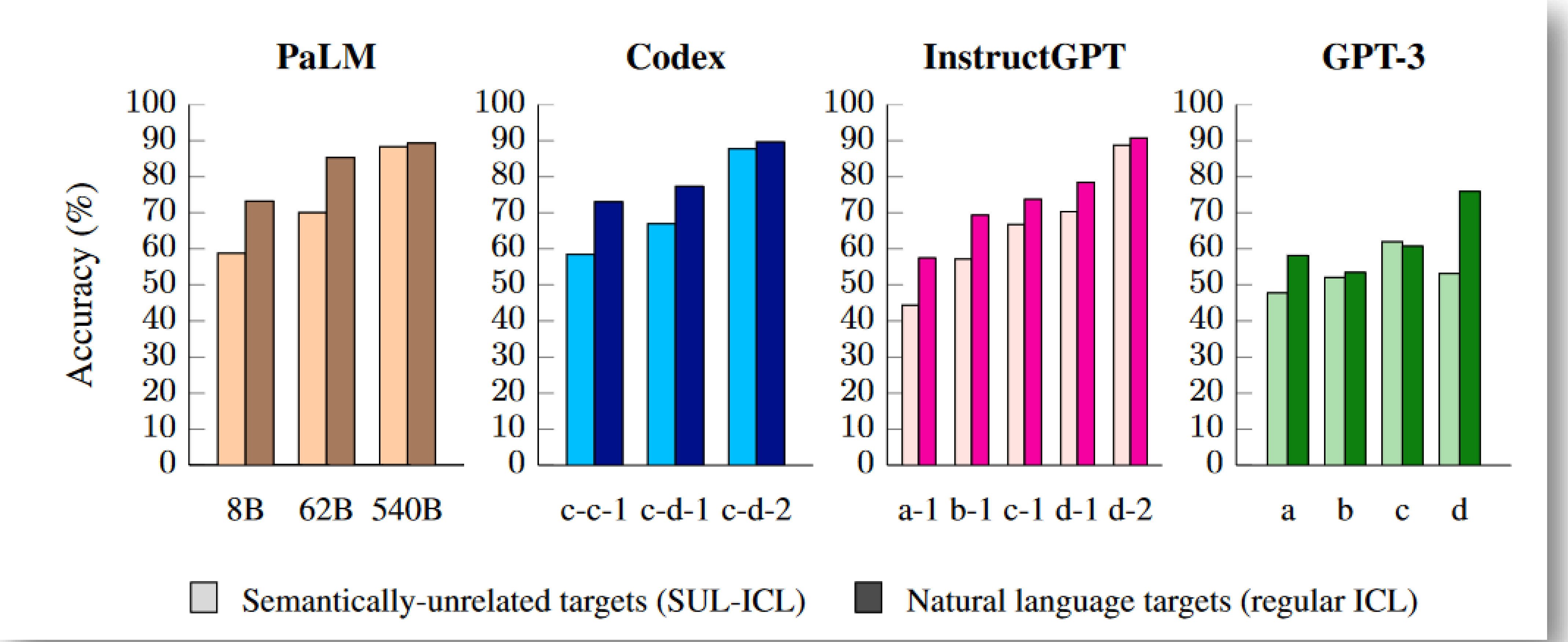
Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

Input-Label Mapping Override Semantic Priors in LLMs



Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

ICL with Semantically Unrelated Labels Emerges with Scale



Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

ICL with Semantically Unrelated Labels Emerges with Scale

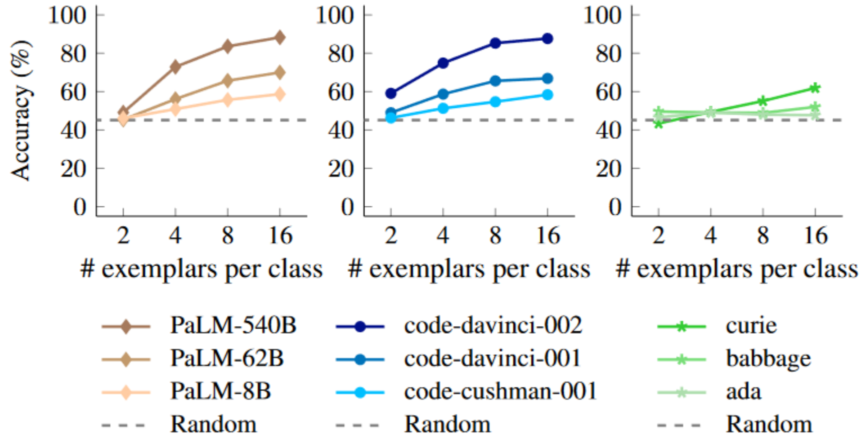


Figure 4: In the SUL-ICL setup, larger models benefit more from additional exemplars than smaller models do. Accuracy is calculated over 100 evaluation examples per dataset and averaged across all datasets. A per-dataset version of this figure is shown in Figure 18 in the Appendix.

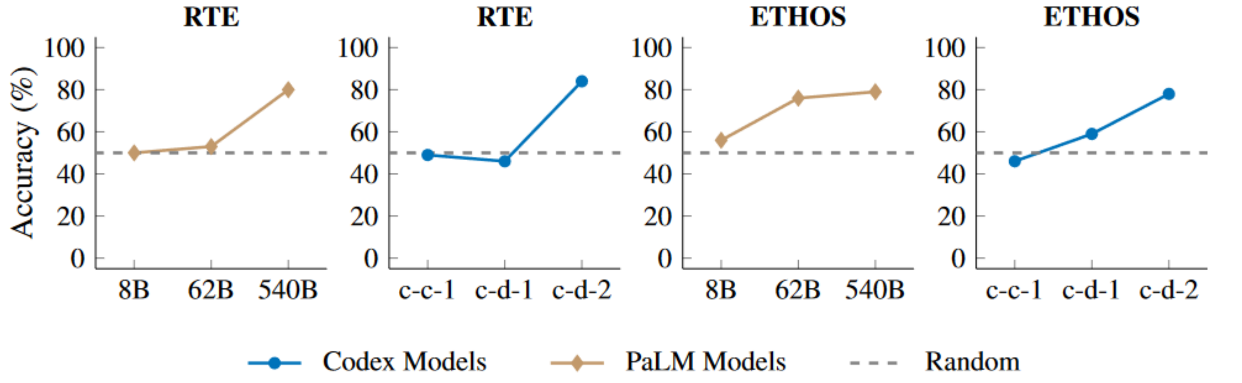
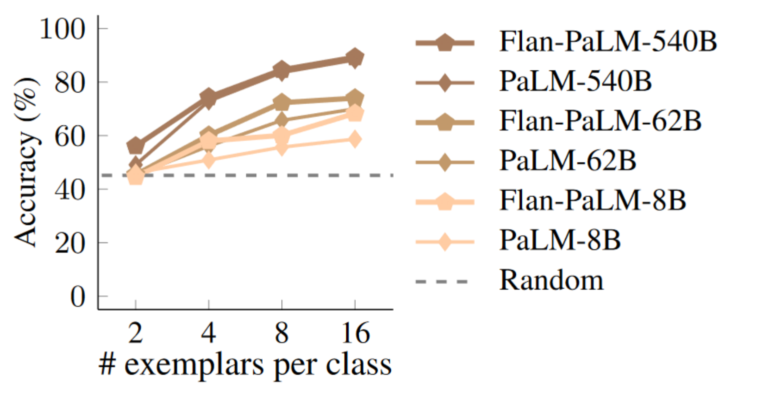


Figure 5: Some tasks in the SUL-ICL setting emerge with scale and can only be successfully performed by large-enough models. These experiments use $k = 8$ in-context exemplars per class. Accuracy is calculated over 100 evaluation examples.

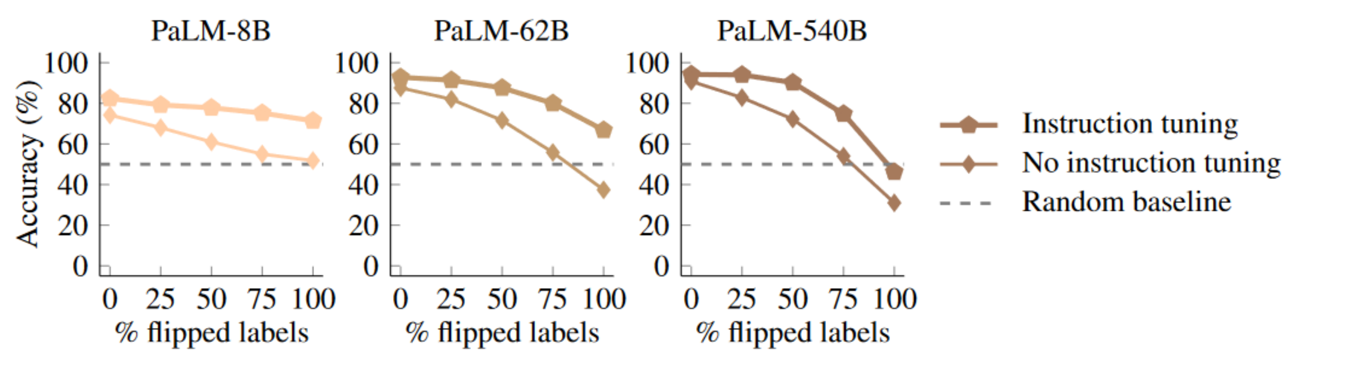
Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

Effect of Instruction Tuning

- Better at learning novel input-output label mapping



- Bad at overriding semantic prior



Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

LLMs can Perform Linear Classification via ICL

Input: 59, 874, 536, 60, 824, 223, 555, 809, 727, 448, 20, 482, 523, 928, 331, 182

Output: Bar

Input: 669, 414, 858, 114, 509, 393, 222, 627, 579, 336, 455, 732, 799, 636, 771, 990

Output: Bar

Input: 405, 146, 99, 760, 880, 778, 922, 555, 170, 600, 843, 358, 323, 654, 501, 603

Output: Bar

Input: 839, 45, 729, 900, 235, 605, 973, 304, 558, 479, 645, 77, 345, 768, 927, 734

Output: Bar

Input: 319, 605, 921, 13, 449, 608, 157, 718, 316, 409, 558, 364, 860, 215, 740, 909

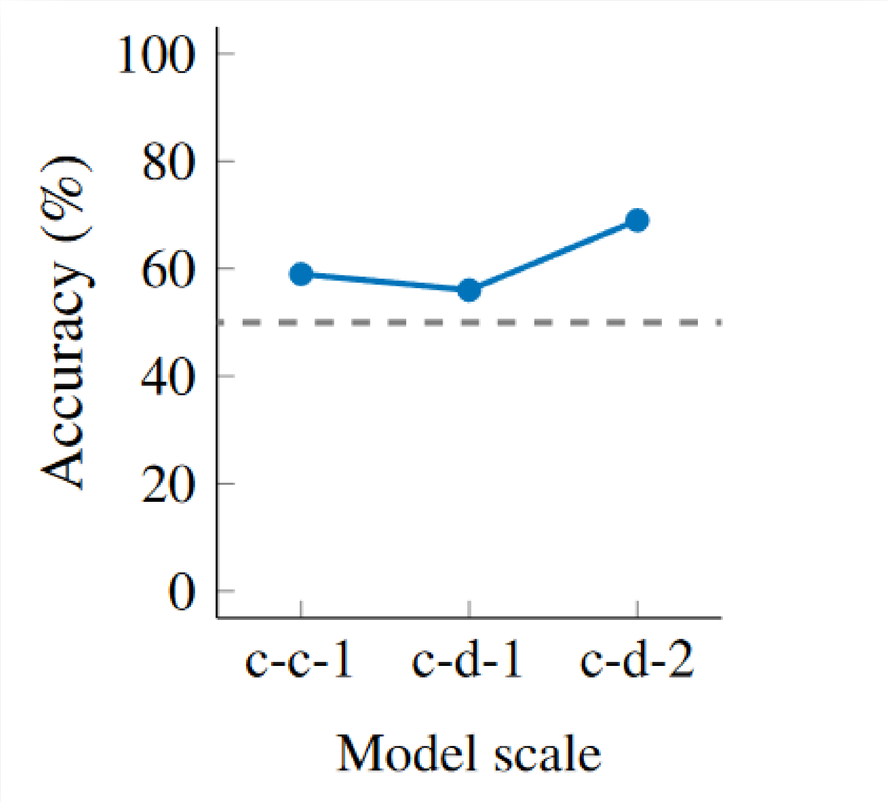
Output: Bar

Input: 101, 969, 495, 149, 394, 964, 428, 946, 542, 814, 240, 467, 435, 987, 297, 466

Output:

Answer:

Bar

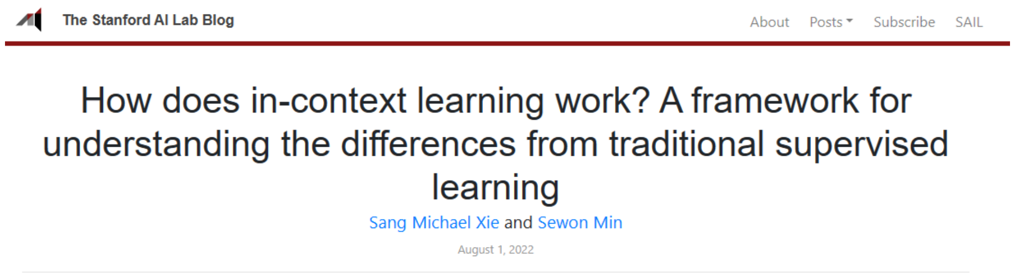


- These phenomenon can not be fully explained by the Bayesian inference framework.
- There are still many questions to be answered about ICL!

Wei, Jerry, et al. "Larger language models do in-context learning differently." arXiv preprint arXiv:2303.03846 (2023).

Summary

- In-Context Learning as an emerging ability from LLM:
  - In-Context Learning is an empirical method that enables efficient 'learning' happens.
  - Understanding how LLMs conduct ICL, e.g., **conducting Bayesian inference to 'locate and extract some pre-trained knowledge** or **learning novel tasks from context.**
  - There are still many unanswered questions about ICL!
- Further Reading:
  - A great blog written by the authors of aforementioned two papers about ICL:


The Stanford AI Lab Blog
About Posts Subscribe SAIL

How does in-context learning work? A framework for understanding the differences from traditional supervised learning

Sang Michael Xie and Sewon Min

August 1, 2022

  - How ICL works with transformer architecture:
    - Garg, Shivam, et al. "What can transformers learn in-context? a case study of simple function classes." (NeurIPS 2022).
    - Akyürek, Ekin, et al. "What learning algorithm is in-context learning? investigations with linear models." (ICLR 2023).

Thanks!