References

- Rethinking the Role of Demonstrations: What Makes In-Context Learning Work? (https://arxiv.org/pdf/2202.12837.pdf)

What makes In-context Learning work ?

- blog (http://ai.stanford.edu/blog/understanding-incontext/)
- paper (https://arxiv.org/pdf/2202.12837.pdf)
    - more empirical
    - various models
        - MetaICL: trained with InContextLearning objective
        - 2 methods: Direct vs Channel ???
    - gold-label vs random (uniform sampling) label: ground-truth not necessary
        - gold improves over zero shot
        - random: small decrease vs gold
            - **very** small for MetaICL
        - sampling for true label distribution: smaller decrease

How does In Context Learning work ?

In-context Learning describes a means of using a fixed LLM to solve a task

- by supplying some number k of *exemplars* (or *demonstrations*) of the new task
- as a *pre-prompt*
- and the presenting a prompt $\mathbf{x}$ to the model
- expecting the model to produce a $\mathbf{y}$
- that is the correct "response" to the task on input $\mathbf{x}$

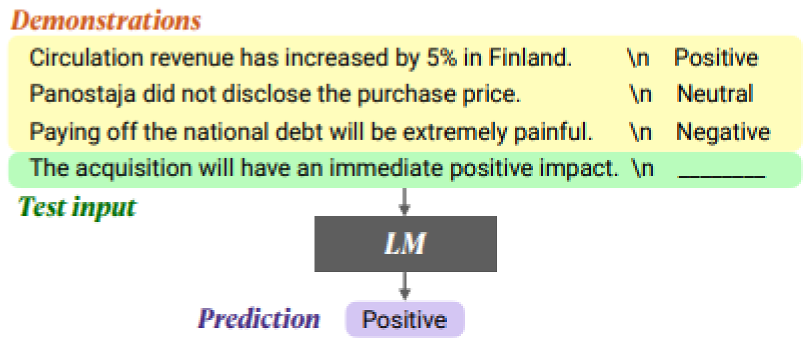

Demonstrations
Circulation revenue has increased by 5% in Finland. \n Positive
Panostaja did not disclose the purchase price. \n Neutral
Paying off the national debt will be extremely painful. \n Negative
The acquisition will have an immediate positive impact. \n ______
Test input
LM
Prediction
Positive

Figure 2: An overview of in-context learning. The demonstrations consist of k input-label pairs from the training data ($k = 3$ in the figure).

Atttribution: [https://arxiv.org/pdf/2202.12837.pdf#page=2 (https://arxiv.org/pdf/2202.12837.pdf#page=2)](https://arxiv.org/pdf/2202.12837.pdf#page=2)

In-Context Learning appears to be a way

- of extending a LM
- *without* further training
    - as opposed to Fine-Tuning
- since
    - the exemplars are given at *test* time
    - no parameter updates to the LLM occur

In-Context Learning uses a pre-trained LLM and the trick of using the Universal Text API

- to turn the new task
- into a text-continuation ("predict the next") task

That is:

- given some number k of exemplars: $\langle \mathbf{x}^{(1)}, \mathbf{y}^{(1)} \rangle, \ldots, \langle \mathbf{x}^{(k)}, \mathbf{y}^{(k)} \rangle$
- the prompt string $\mathbf{x}$

we create a sequence $\dot{\mathbf{x}}$ encoding the exemplars and prompt

- and ask the LLM model to predict
$$p(\mathbf{y}|\dot{\mathbf{x}})$$

A common way to create the sequence $\dot{\mathbf{x}}$ by concatenating the exemplars and prompt, using separator characters to as delimiters.

$$\begin{aligned}\dot{\mathbf{x}} = \text{concat}(\quad & \mathbf{x}^{(1)}, \langle \text{SEP}_1 \rangle, \mathbf{y}^{(1)}, \langle \text{SEP}_2 \rangle, \\ & \vdots \\ & \mathbf{x}^{(k)}, \langle \text{SEP}_1 \rangle, \mathbf{y}^{(k)}, \langle \text{SEP}_2 \rangle, \\ & \mathbf{x} \\ &)\end{aligned}$$

The LLM then computes

$$p(\mathbf{y}|\dot{\mathbf{x}})$$

For notational convenience, we will omit writing the concatenation

- and just write this as the conditional probability
$$p(\mathbf{y}|\mathbf{x}, \mathbf{x}^{(1)}, \mathbf{y}^{(1)}, \ldots, \mathbf{x}^{(k)}, \mathbf{y}^{(k)})$$

But why should this work ?

More interestingly

- what is a good theory
- and how can we test it

We will present a paper (https://arxiv.org/pdf/2202.12837.pdf) that attempts to present some insights into the process.

Testing some theories

In order to test a theory

- various aspects of the exemplars are proposed as variables
- one variable at a time is perturbed
- the effect of the perturbations is measured across a range of benchmarks

- **and compare to measurements before the perturbation**

The results are summarized in the following diagram

- that we will subsequently refer to for each experiment

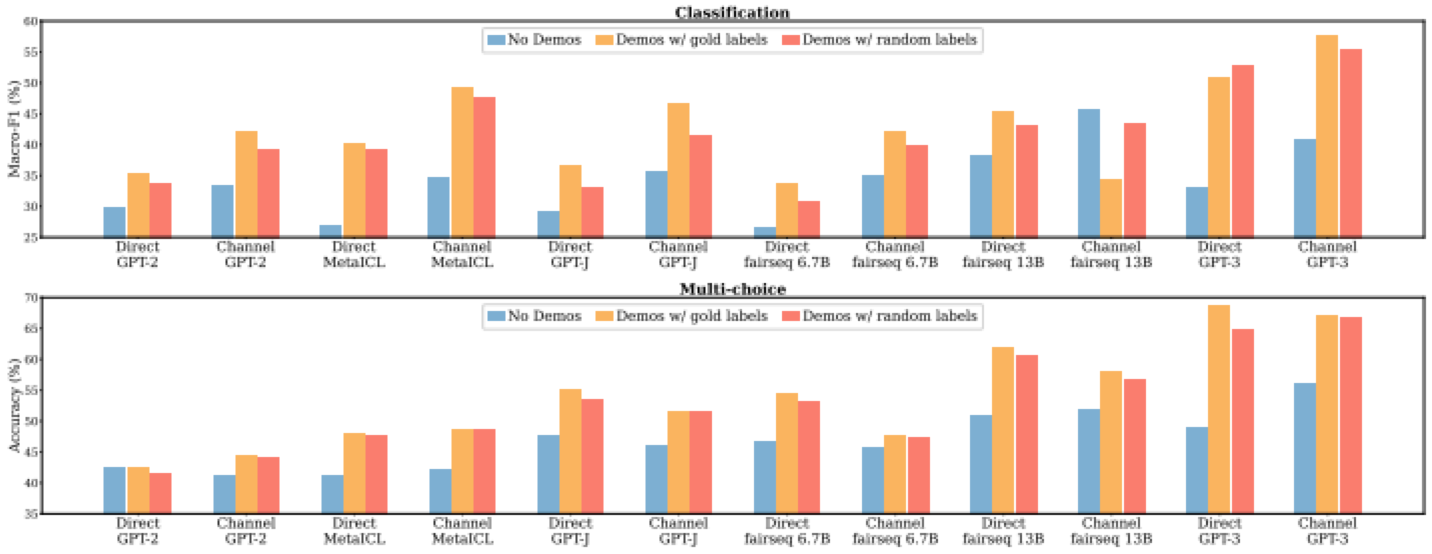


Figure 3: Results when using no-demonstrations, demonstrations with gold labels, and demonstrations with random labels in classification (top) and multi-choice tasks (bottom). The first eight models are evaluated on 16 classification and 10 multi-choice datasets, and the last four models are evaluated on 3 classification and 3 multi-choice datasets. See Figure 11 for numbers comparable across all models. **Model performance with random labels is very close to performance with gold labels** (more discussion in Section 4.1).

This chart shows the result of perturbations

- run across a variety of models
- of sizes ranging from 774M to 175B parameters
- each experiment is averaged across multiple benchmarks

The number of demonstrations, when present, is $k = 16$.

Zero shot verus $k \geq 1$ shot

The first experiment measures the effect of the presence/absence of exemplars.

In the diagram, compare

- "No demos": the blue bar
- "Gold labels": the gold bar

Conclusion

$k \geq 1$ exemplars *improves performance* relative to zero-shot.

Parts of the Context

The next set of experiments varies parts of the context (exemplars and prompt).

Given exemplars $\langle \mathbf{x}^{(1)}, \mathbf{y}^{(1)} \rangle, \ldots, \langle \mathbf{x}^{(k)}, \mathbf{y}^{(k)} \rangle$ the authors posit some salient characteristics

- the *input distribution* I from which the exemplar *features* are drawn $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}$
- the distribution L of the exemplar *labels* $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(k)}$
- the feature/label mapping relationship M
    - i.e., the pair of $\mathbf{x}^{\mathbf{(i)}}$ and $\mathbf{y}^{\mathbf{(i)}}$, for $1 \le i \le k$
- formatting
    - the encoding of the exemplars and prompt into $\dot{\mathbf{x}}$

Feature/label mapping relationship

Let $\mathcal{C}$ denote the set from which exemplar labels are drawn.

In this experiment, replace

- correct label $\mathbf{y^{(i)}}$ for exemplar i
- with label $\tilde{\mathbf{y}}^{\mathbf{(i)}}$ drawn at random (uniformly) from $\mathcal{C}$.

That is, we preserve I and L, but break M.

In the diagram, compare

- "Gold labels": the gold bar (true labels)
- "Random labels": the reddish bar

Conclusions

- Correct ("gold") labels improve performance over random labels
    - but not as much as expected, perhaps
- Random labels *improves performance* over *no exemplars*
    - "Ground truth" matters surprisingly little !

The fact that an *incorrect* M improves performance relative to no exemplars is surprising.

This suggests

- that the exemplars are used to infer the *task to be performed*
- once the task has been identified
    - the exemplar mis-labeling is ignored
- the model is able to perform the task as it was *trained* during training

See the [Signifier theory in the module (Prompt_Engineering_Suggestions.ipynb#Signifier:-direct-specification)](Prompt_Engineering_Suggestions.ipynb#Signifier:-direct-specification)

Input distribution

In this experiment

- each exemplar input $\mathbf{x}^{(\mathbf{i})}$ is replaced by
- a random $\mathbf{x}^{(\mathbf{i})}_{\text{rand}}$ drawn from a text corpus *other than the one used for Training*

We note that this experiment *also* breaks the feature/label relationship M

- we preserve the original labels $\mathbf{y}^{(\mathbf{i})}$ for exemplar i
- which is not necessarily related to $\mathbf{x}^{(\mathbf{i})}_{\text{rand}}$

We can contrast the results of this experiment the effect of breaking M alone.

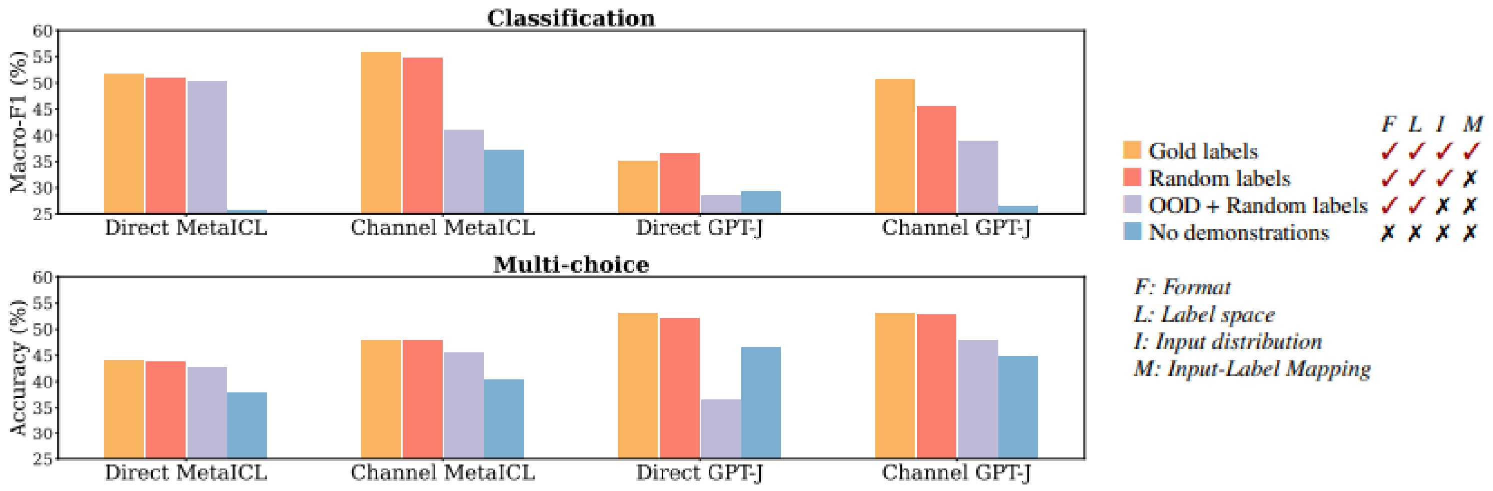


Figure 8: Impact of the distribution of the inputs. Evaluated in classification (top) and multi-choice (bottom). The impact of the distribution of the input text can be measured by comparing 🟥 and 🟪. The gap is substantial, with an exception in Direct MetaICL (discussion in Section 5.1).

In the above diagram, compare

- the lavender (third bar from left): perturbed I and M
- the red bar (second bar from left): perturbed M alone

Conclusions

The M relationship is broken in both cases. But

- preserving the original distribution I of exemplar features
- improves performance relative to changing the distribution

Why might this be ?

The suggestion is that the model was trained with the LLM objective ("predict the next")

- from a training distribution
- and $\mathbf{x}_{\text{rand}}$ is from a *different* distribution
- so the model struggles on non-training input

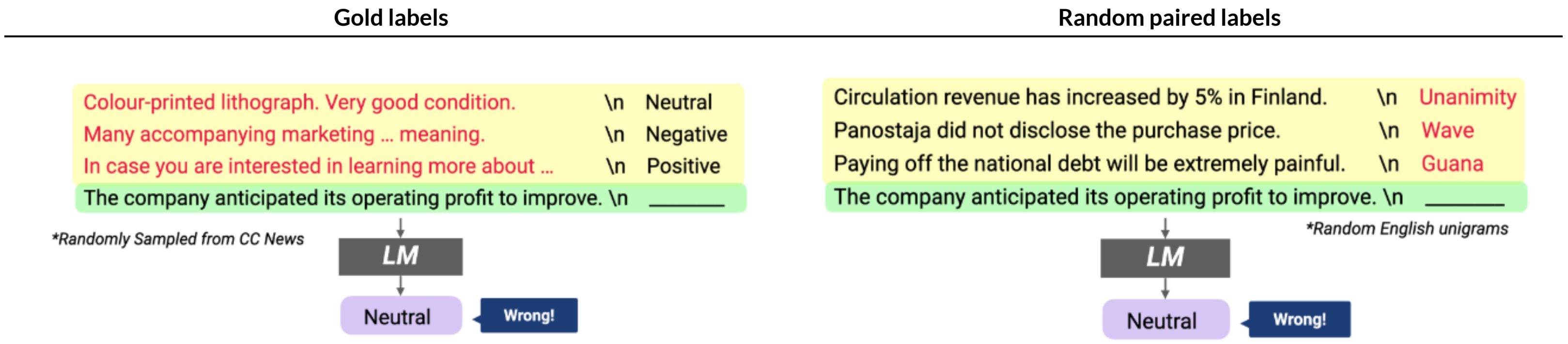
Output distribution

In this experiment

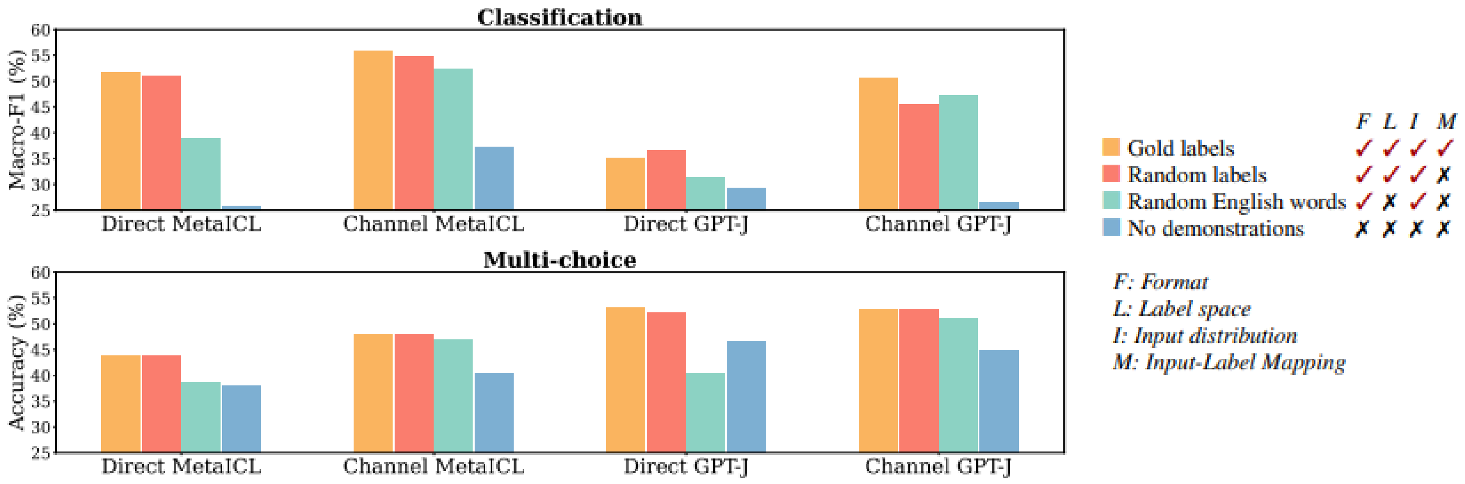
- each exemplar label $\mathbf{y}^{(\mathbf{i})}$ (from $\mathcal{C}$) is replaced by
- $\tilde{\mathbf{y}}^{(\mathbf{i})}$ a *random* English word from $\mathcal{C}_{\text{rand}}$

Note that we *also break* M

- e.g., $\mathbf{y}^{(\mathbf{i})} = \mathbf{y}^{(i')}$ does not imply $\tilde{\mathbf{y}}^{(\mathbf{i})} = \tilde{\mathbf{y}}^{(i)}$


Gold labels
Random paired labels
Colour-printed lithograph. Very good condition. \n Neutral
Many accompanying marketing ... meaning. \n Negative
In case you are interested in learning more about ... \n Positive
The company anticipated its operating profit to improve. \n ______
*Randomly Sampled from CC News
LM
Neutral
Wrong!
Circulation revenue has increased by 5% in Finland. \n Unanimity
Panostaja did not disclose the purchase price. \n Wave
Paying off the national debt will be extremely painful. \n Guana
The company anticipated its operating profit to improve. \n ______
*Random English unigrams
LM
Neutral
Wrong!

Attribution: [http://ai.stanford.edu/blog/understanding-incontext/ (http://ai.stanford.edu/blog/understanding-incontext/)](http://ai.stanford.edu/blog/understanding-incontext/)



Classification
Macro-F1 (%)
Multi-choice
Accuracy (%)
Direct MetaICL
Channel MetaICL
Direct GPT-J
Channel GPT-J
F L I M
Gold labels ✓ ✓ ✓ ✓
Random labels ✓ ✓ ✓ ✗
Random English words ✓ ✗ ✓ ✗
No demonstrations ✗ ✗ ✗ ✗
F: Format
L: Label space
I: Input distribution
M: Input-Label Mapping

In the above diagram, compare

- the red bar: random labels
- the turquoise bar (third from left): random words

The difference: Although we break M in both cases

- in the "random labels" case: the labels are chosen from the correct output distribution $\mathcal{C}$
- in the "random words" case: the labels come from a distribution other than $\mathcal{C}$

Conclusions

The M relationship is broken in both cases. But

- preserving the original distribution L of exemplar labels
- improves performance relative to changing the distribution of labels

Formatting

In this experiment

- *format* is defined as the *pairing* of a feature and label within an exemplar
- not necessarily a *correct pairing*: mapping M not necessarily correct

One experiment is run

- with *only* exemplar features (and no exemplar labels): $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}$
- natural comparison is with experiment of correct format
  - $\mathbf{x^{(i)}}$
  - paired with random English words (from $\mathcal{C}_{\mathrm{rand}}$) as labels

A second experiment is run

- with *only* exemplar labels (and no exemplar features): $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(k)}$
- natural comparison is with experiment of correct format
    - a random $\mathbf{x}^{(\mathbf{i})}_{\text{rand}}$ drawn from a text corpus
    - paired with $\mathbf{y}^{(\mathbf{i})}$

Both comparison experiments

- preserve the format: feature/exemplar pairs
- without preserving M
- or the distribution I in the first case, and L in the second case

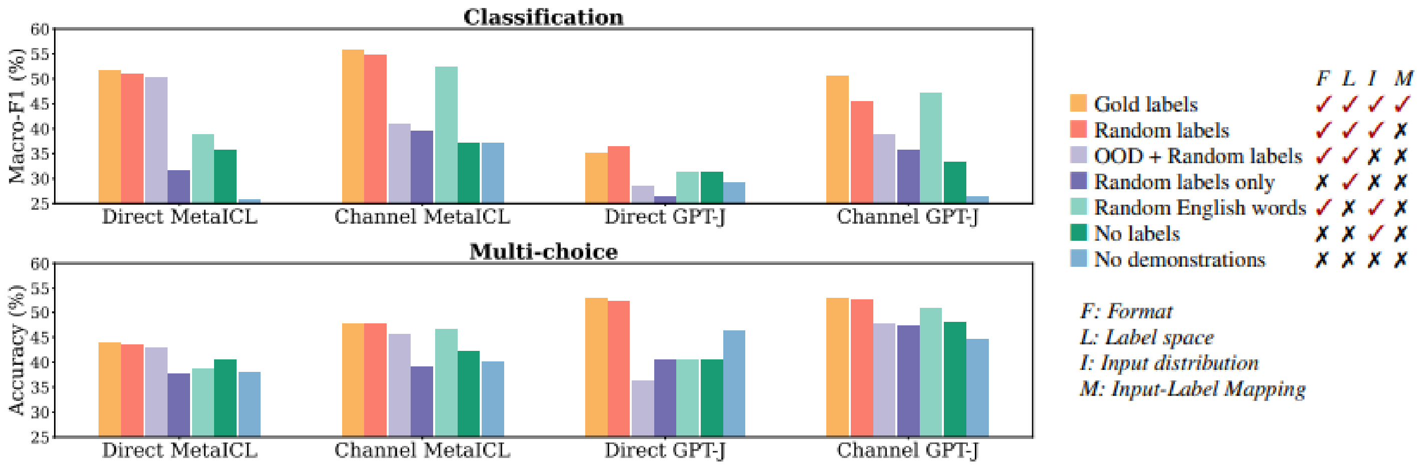


Figure 10: Impact of the format, i.e., the use of the input-label pairs. Evaluated in classification (top) and multi-choice (bottom). Variants of demonstrations without keeping the format (🟪 and 🟩) are overall not better than no demonstrations (🟦). Keeping the format is especially significant when it is possible to achieve substantial gains with the label space but without the inputs (🟪 vs. 🟪 in Direct MetaICL), or with the input distribution but without the labels (🟩 vs. 🟩 in Channel MetaICL and Channel GPT-J). More discussion in Section 5.3.

Conclusions

- Not keeping the format has performance *on par* with **no demonstrations** at all
- Keeping the format retains most of the benefit achievable with either
    - correct I (but incorrect M)
    - or correct L (but incorrect M)

The suggestion is that correct format an important feature

- to enable the LLM to recognize the task from exemplars

Exemplars that differ from the LLM

The on-line articles makes another interesting observation.

Observe the encoding of

- the exemplars $\langle \mathbf{x}^{(1)}, \mathbf{y}^{(1)} \rangle$, $\ldots, \langle \mathbf{x}^{(k)}, \mathbf{y}^{(k)} \rangle$
- the prompt string $\mathbf{x}$

into the string $\dot{\mathbf{x}}$ that is the input to the LLM

$$\begin{aligned}\dot{\mathbf{x}} = \text{concat}(\quad & \mathbf{x}^{(1)}, \langle \text{SEP}_1 \rangle, \mathbf{y}^{(1)}, \langle \text{SEP}_2 \rangle, \\ & \vdots \\ & \mathbf{x}^{(k)}, \langle \text{SEP}_1 \rangle, \mathbf{y}^{(k)}, \langle \text{SEP}_2 \rangle, \\ & \mathbf{x} \\ &)\end{aligned}$$

The distribution from which encoded $\dot{\mathbf{x}}$ is drawn

- is probably *much different* than
- the distribution (Internet text documents) on which the LLM was trained

in several ways

- syntax
    - sentences (e.g., exemplars)
    - are not naturally separated by an inter-example separator $\langle \text{SEP} \rangle$ (whatever is chosen) in the training distribution
- coherence
    - exemplars i and $i+1$
    - many not naturally follow one another in the training distribution
        - may be different topics
        - but demonstrate the same concept (that is why they were chosen as exemplars)

The article posits that

- these encoding anomalies
- are low-frequency noise
- that the LLM is able to ignore
- providing there is more "signal" in the exemplars

A theory of In-Context Learning

A more theoretical paper (https://arxiv.org/pdf/2111.02080.pdf) and accompanying online article (http://ai.stanford.edu/blog/understanding-incontext/)

- combine these experimental insights
- into a theory
- and mathematical model of the theory
- that is consistent with the experimental results

The authors posit

- during training, the LLM learns "concepts", for example
    - abstract ideas
        - question answering
        - sentiment
    - plans
        - how to solve a multi-step task: travel directions

Concepts

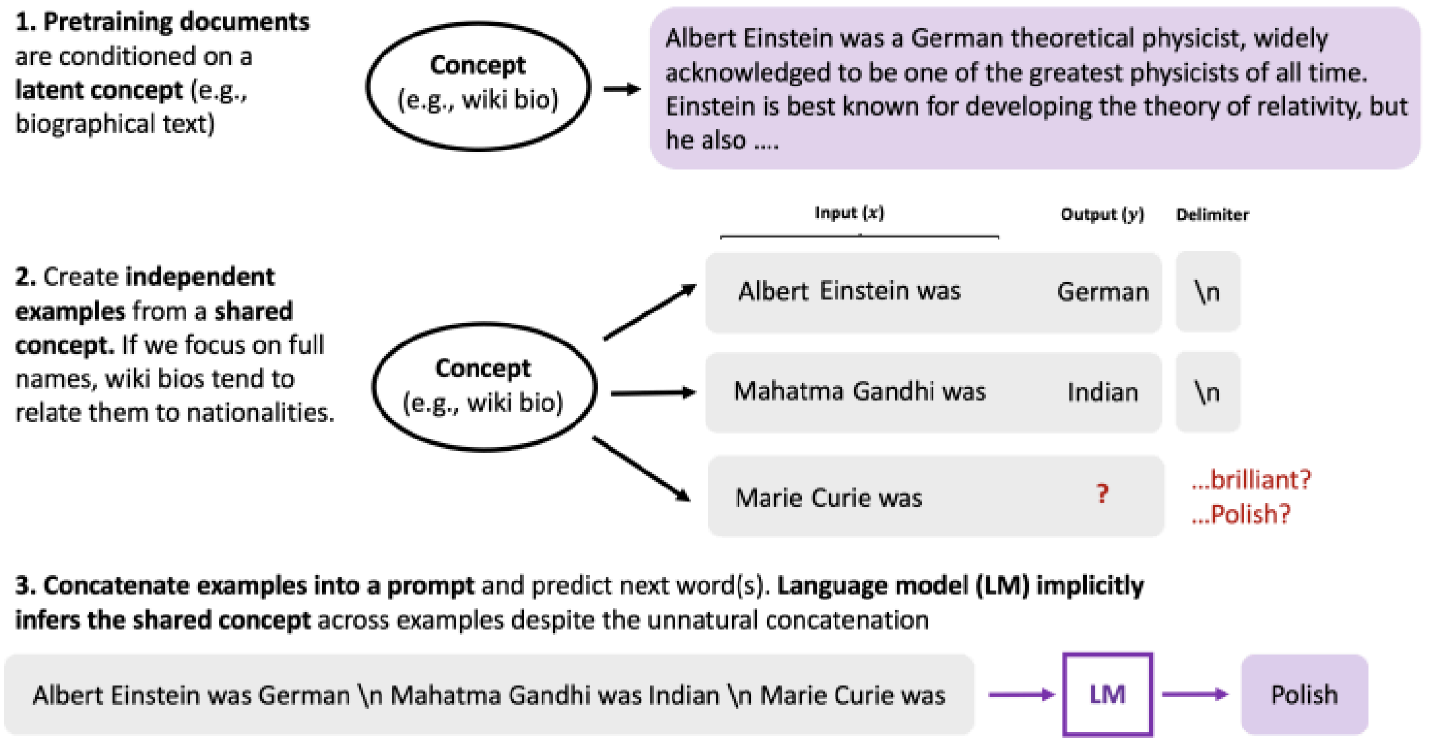

1. Pretraining documents are conditioned on a **latent concept** (e.g., biographical text)
Concept (e.g., wiki bio)
Albert Einstein was a German theoretical physicist, widely acknowledged to be one of the greatest physicists of all time. Einstein is best known for developing the theory of relativity, but he also ....
Input (x)
Output (y)
Delimiter
2. Create **independent examples** from a **shared concept.** If we focus on full names, wiki bios tend to relate them to nationalities.
Concept (e.g., wiki bio)
Albert Einstein was
German
\n
Mahatma Gandhi was
Indian
\n
Marie Curie was
?
...brilliant?
...Polish?
3. Concatenate examples into a prompt and predict next word(s). **Language model (LM) implicitly infers the shared concept** across examples despite the unnatural concatenation
Albert Einstein was German \n Mahatma Gandhi was Indian \n Marie Curie was
LM
Polish

Figure 1: In-context learning can emerge from modeling long-range coherence in the pretraining data. During pretraining, the language model (LM) implicitly learns to infer a latent concept (e.g., wiki bios, which typically transition between name (Albert Einstein) → nationality (German) → occupation (physicist) → ...) shared across sentences in a document. Although prompts are unnatural sequences that concatenate independent examples, in-context learning occurs if the LM can still infer the shared concept across examples to do the task (name → nationality, which is part of wiki bios).

Attribution: [https://arxiv.org/pdf/2111.02080.pdf#page=2 (https://arxiv.org/pdf/2111.02080.pdf#page=2)](https://arxiv.org/pdf/2111.02080.pdf#page=2)

The model's LLM "predict the next" training objective did not specify to goal of learning concepts.

But

- summarizing a large number of similar training documents (e.g., collection of biographies)
- in a parameters-constrained model
- logically suggests that concepts emerge as a way of reducing parameter usage

The authors suggest that the LLM's probability of outputting $\mathbf{y}$ given prompt $\mathbf{x}$ is formed by

$$p(\mathbf{y}|\mathbf{x}) = \int_{c \in \text{Concepts}} p(\mathbf{y}|\mathbf{x}, c) p(c) \ d(c)$$

That is, the output

- is the sum over all concepts
- of the probability of outputting $\mathbf{y}$ given prompt $\mathbf{x}$ and concept c

Furthermore: the context (i.e., exemplars) of in-context learning

- helps the LLM identify the concept c
- to which the prompt $\mathbf{x}$ implicitly refers

The experimental results seem to suggest that the exemplars

- don't need to be fully accurate
    - the model tolerates inaccurate mappings M between feature input space I and label space L
- that correctly identifying I and L through the exemplars is
    - advantageous
    - but not completely necessary
- that the *format* of the exemplar
    - paired features and labels
    - is important

Under this theory

- the exemplars **are not teaching** new concepts
    - hence M can be inaccurate
- but serving to help the LLM **identify** a concept learned in training

That is, the encoded exemplars in $\dot{\mathbf{x}}$

- are related to $p(c)$

Once the concept c is identified, the output $\mathbf{y}$ depends

- on the distributions I and L
- on the mapping M

that were learned during training.

The actual output $\mathbf{y}$

$$p(\mathbf{y}|\mathbf{x}, \mathbf{c})$$

depends on training examples

- the exemplars
  $\langle \mathbf{x}^{(1)}, \mathbf{y}^{(1)} \rangle, \ldots, \langle \mathbf{x}^{(k)}, \mathbf{y}^{(k)} \rangle$
  do not appear in the equation

```
In [2]: print("Done")
```

```
Done
```