Domain-Specific NLP

Contents

1. Domain-Specific Models
    a. *Don't Stop Pre-training*
    b. Specialized Models (BioBERT, SciBERT, Galactica)
2. Unsupervised Classification Models
    a. Out-of-the-box representations: limitations
    b. SimCSE, E5, GTE...
    c. Document Representation: DocBERT
3. Learning Long-Range Dependencies
    a. Long-range attention models
    b. State-space models: S4

Domain-Specific Models

Domain-Specific Models

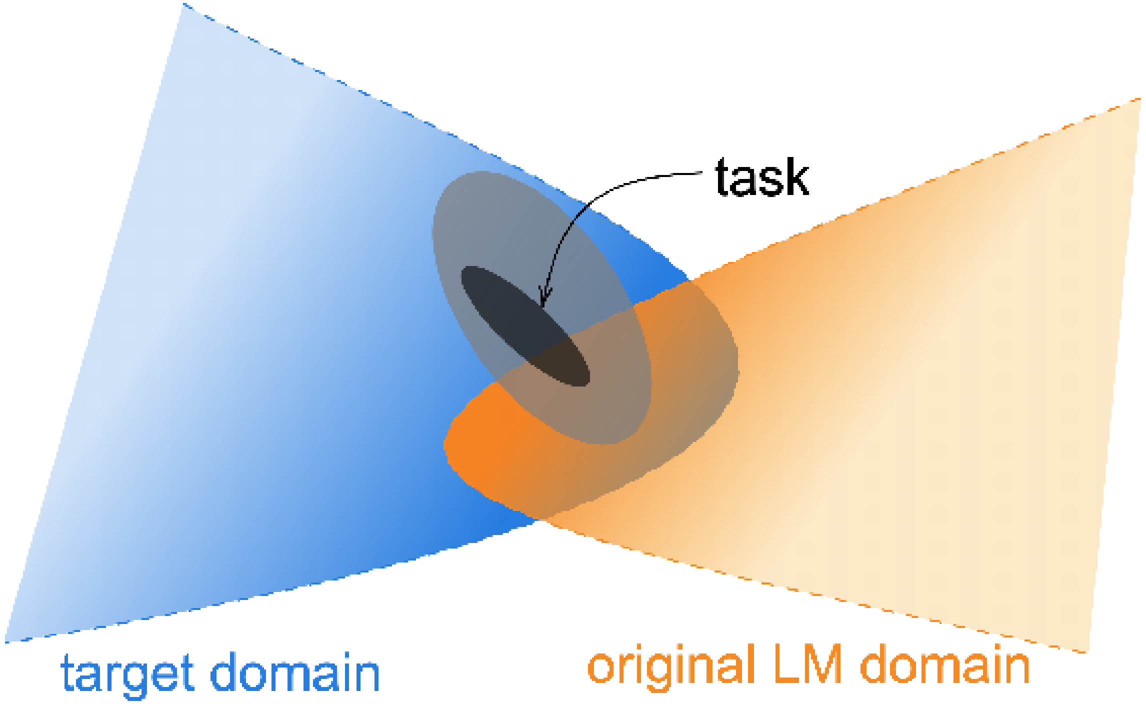
Pretrained (Large) Language Models are trained on content crawled over the internet, books, reports and news papers and are, hence **are open-domain**.

A **textual domain** is the **distribution over language characterizing a given topic or genre** [1].

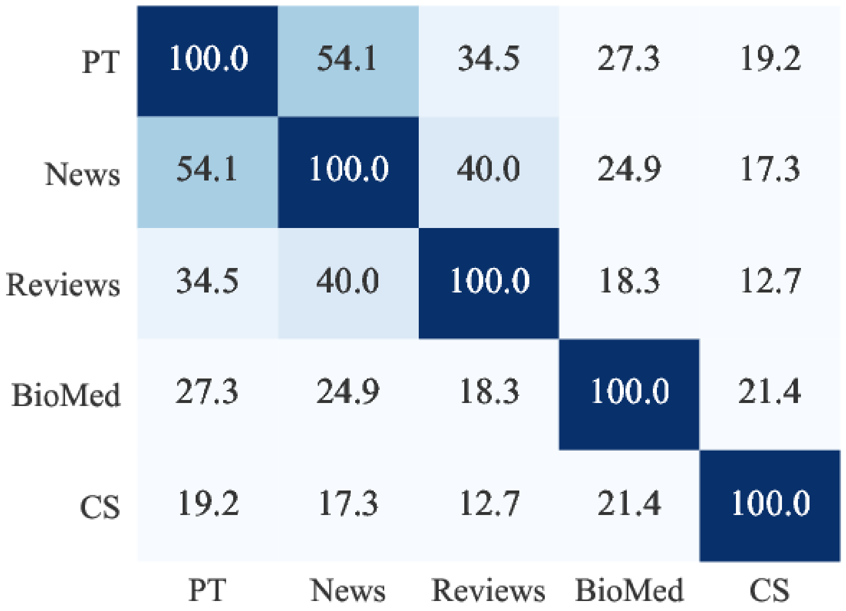
- You are more likely to see the word "integer" in computer science than in news papers.
- An (L)LM will be more perplex to the word "integer" even though the input comes from a StackOverflow post.

Don’t Stop Pretraining

Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. [1]



Don’t Stop Pretraining



	PT	News	Reviews	BioMed	CS
PT	100.0	54.1	34.5	27.3	19.2
News	54.1	100.0	40.0	24.9	17.3
Reviews	34.5	40.0	100.0	18.3	12.7
BioMed	27.3	24.9	18.3	100.0	21.4
CS	19.2	17.3	12.7	21.4	100.0

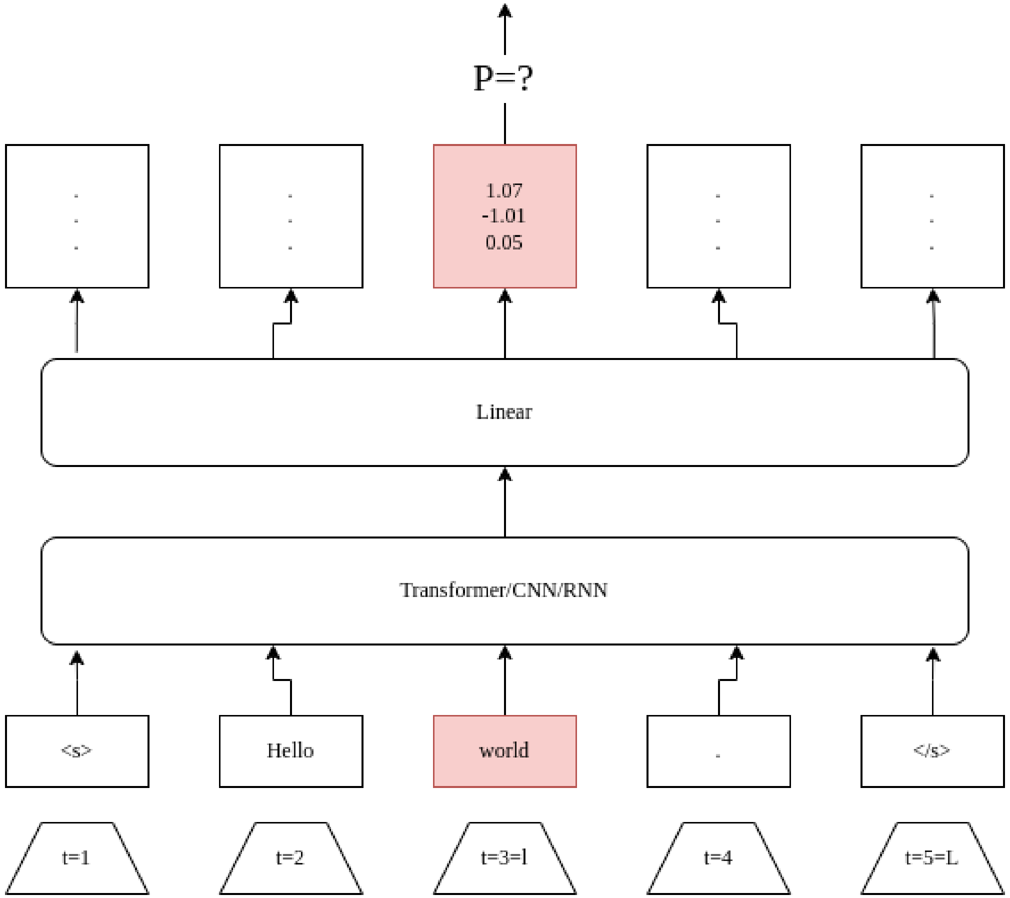
Figure 2: Vocabulary overlap (%) between domains. PT denotes a sample from sources similar to ROBERTA’s pretraining corpus. Vocabularies for each domain are created by considering the top 10K most frequent words (excluding stopwords) in documents sampled from each domain.

Part-of-Speech Tagging (POS)

There several levels of granularity.: using the tag set for english

("He", "likes", "to", "drink", "tea"), → ("PRP", "VBP", "TO", "VB", "NN").

Conditional Random Field (CRF)



Conditional Random Field (CRF)

For each token in a sentence at position l we want to compute a probability p to belong to a class n.

$$p : f(\mathbf{x}, \theta)_l \mapsto ?$$

with $p \in [0, 1]$

Conditional Random Field (CRF)

Using the softmax function?

$$p : f(\mathbf{x}, \theta)_l \mapsto \frac{e^{f(\mathbf{x},\theta)_l^{(n)}}}{\sum_{n'=1}^{N} e^{f(\mathbf{x},\theta)_l^{(n')}}}$$

The probability given by the softmax function will not encode non-local dependencies!

Conditional Random Field (CRF)

We need to take sequential decisions: what if we add transition scores into our softmax?

$$p : f(\mathbf{x}, \theta)_l \mapsto \frac{e^{f(\mathbf{x},\theta)_l^{(n)} + t(y_l^{(n)}, y_{l-1})}}{\sum_{n'=1}^{N} e^{f(\mathbf{x},\theta)_l^{(n')} + t(y_l^{(n')}, y_{l-1})}}$$

But this is the probability for one token to belong to a class, we want to compute the probability of a whole sequence of label at once...

Conditional Random Field (CRF)

$$P(\mathbf{y}|\mathbf{x}) = \prod_{l=2}^{L} p(\mathbf{y}|f(\mathbf{x},\theta)_l)$$

$$= \prod_{l=2}^{L} \frac{e^{f(\mathbf{x},\theta)_l^{(n)} + t(y_l^{(n)}, y_{l-1})}}{\sum_{n'=1}^{N} e^{f(\mathbf{x},\theta)_l^{(n')} + t(y_l^{(n')}, y_{l-1})}}$$

$$P(\mathbf{y}|\mathbf{x}) = \frac{exp[\sum_{l=2}^{L}\left(f(\mathbf{x},\theta)_l^{(n)} + t(y_l^{(n)}, y_{l-1})\right)]}{\sum_{n'=1}^{N} exp[\sum_{l=2}^{L}\left(f(\mathbf{x},\theta)_l^{(n')} + t(y_l^{(n')}, y_{l-1})\right)]}$$

$$= \frac{exp[\sum_{l=2}^{L}\left(U(\mathbf{x}, y_l^{(n)}) + T(y_l^{(n)}, y_{l-1})\right)]}{\sum_{n'=1}^{N} exp[\sum_{l=2}^{L}\left(U(\mathbf{x}, y_l^{(n')}) + T(y_l^{(n')}, y_{l-1})\right)]}$$

$$= \frac{exp[\sum_{l=2}^{L}\left(U(\mathbf{x}, y_l^{(n)}) + T(y_l^{(n)}, y_{l-1})\right)]}{Z(\mathbf{x})}$$

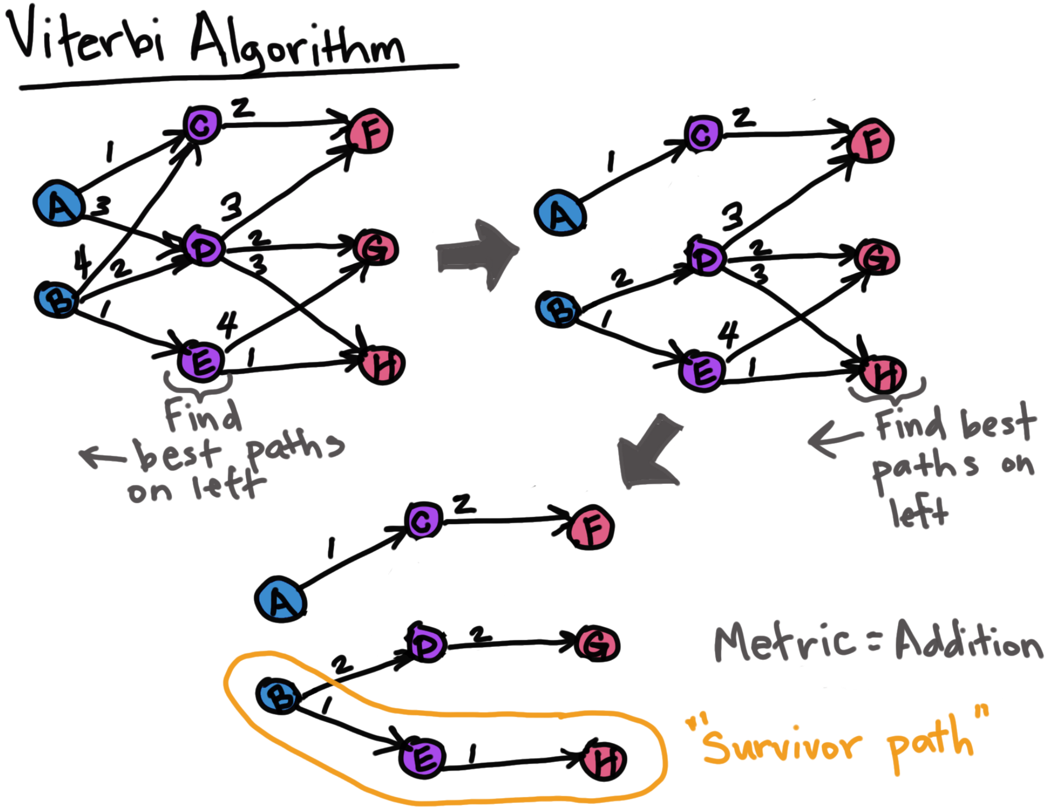
Conditional Random Field (CRF)

$Z(\mathbf{x})$ is commonly referred as the partition function. However, its not trivial to compute: we'll end up with a complexity of $\mathcal{O}(N^L)$.

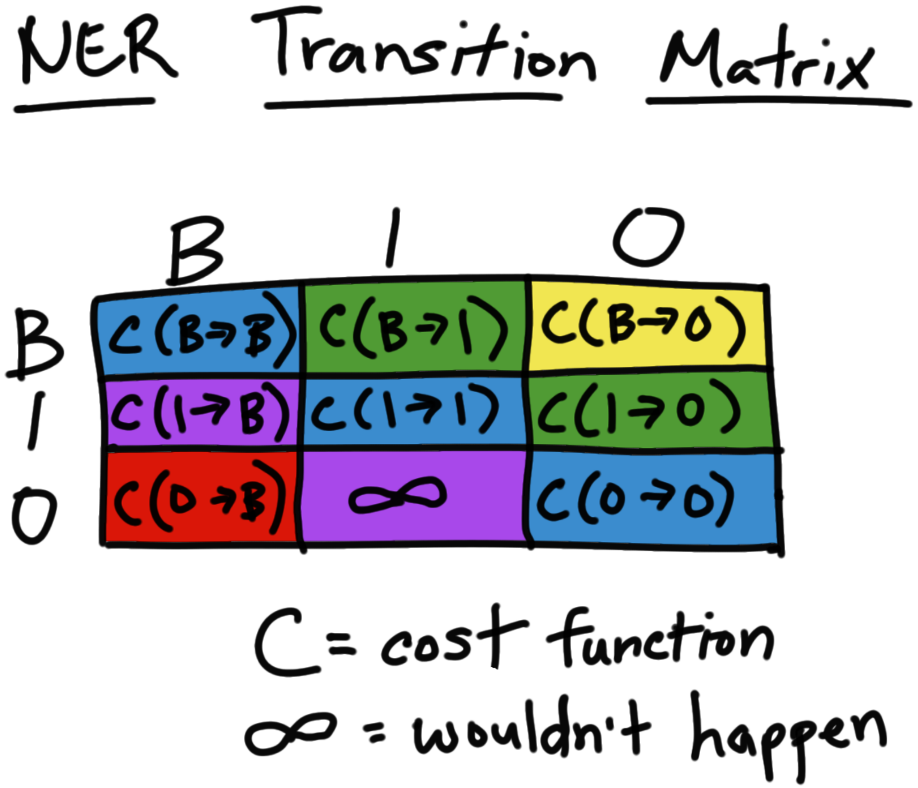
Where N is the number of possible labels and L the sequence length.

How do we proceed?

Conditional Random Field (CRF)



Conditional Random Field (CRF)



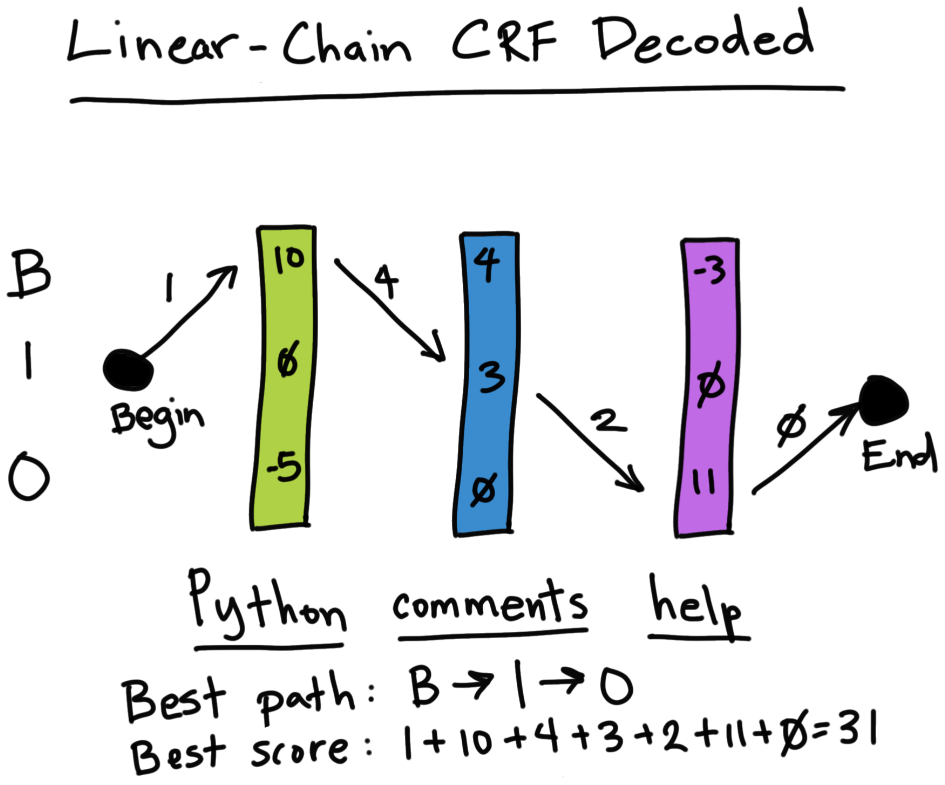
NER Transition Matrix

	B	I	O
B	c(B→B)	C(B→I)	C(B→O)
I	c(I→B)	c(I→I)	c(I→O)
O	c(O→B)	∞	c(O→O)

C = cost function

∞ = wouldn't happen

Conditional Random Field (CRF)



Conditional Random Field (CRF)

Negative log-likelihood:

$$\begin{aligned}
\mathcal{L} &= -log(P(\mathbf{y}|\mathbf{x})) \\
&= -log(\frac{exp[\sum_{l=2}^{L}(U(\mathbf{x}, y_l^{(n)}) + T(y_l^{(n)}, y_{l-1}))]}{Z(\mathbf{x})}) \\
&= -[log(exp[\sum_{l=2}^{L}(U(\mathbf{x}, y_l^{(n)}) + T(y_l^{(n)}, y_{l-1}))]) - log(Z(\mathbf{x}))] \\
&= log(Z(\mathbf{x})) - \sum_{l=2}^{L}(U(\mathbf{x}, y_l^{(n)}) + T(y_l^{(n)}, y_{l-1}))
\end{aligned}$$

Conditional Random Field (CRF)

There is an effective way to compute $log(Z(\mathbf{x}))$ with a complexity of $\mathcal{O}(L)$ using the Log-Sum-Exp trick.

$$log(Z(\mathbf{x})) = log(\sum_{n'=1}^{N} exp[\sum_{l=2}^{L} (U(\mathbf{x}, y_l^{(n')}) + T(y_l^{(n')}, y_{l-1}))])$$

$$= c + log(\sum_{n'=1}^{N} exp[\sum_{l=2}^{L} (U(\mathbf{x}, y_l^{(n')}) + T(y_l^{(n')}, y_{l-1})) - c])$$

Conditional Random Field (CRF)

If we fix $c = max\{U(\mathbf{x}, y_l^{(1)}) + T(y_l^{(1)}, y_{l-1}), \ldots, U(\mathbf{x}, y_l^{(N)}) + T(y_l^{(N)}, y_{l-1})\}$ we ensure that the largest positive exponentiated term is $exp(0) = 1$.

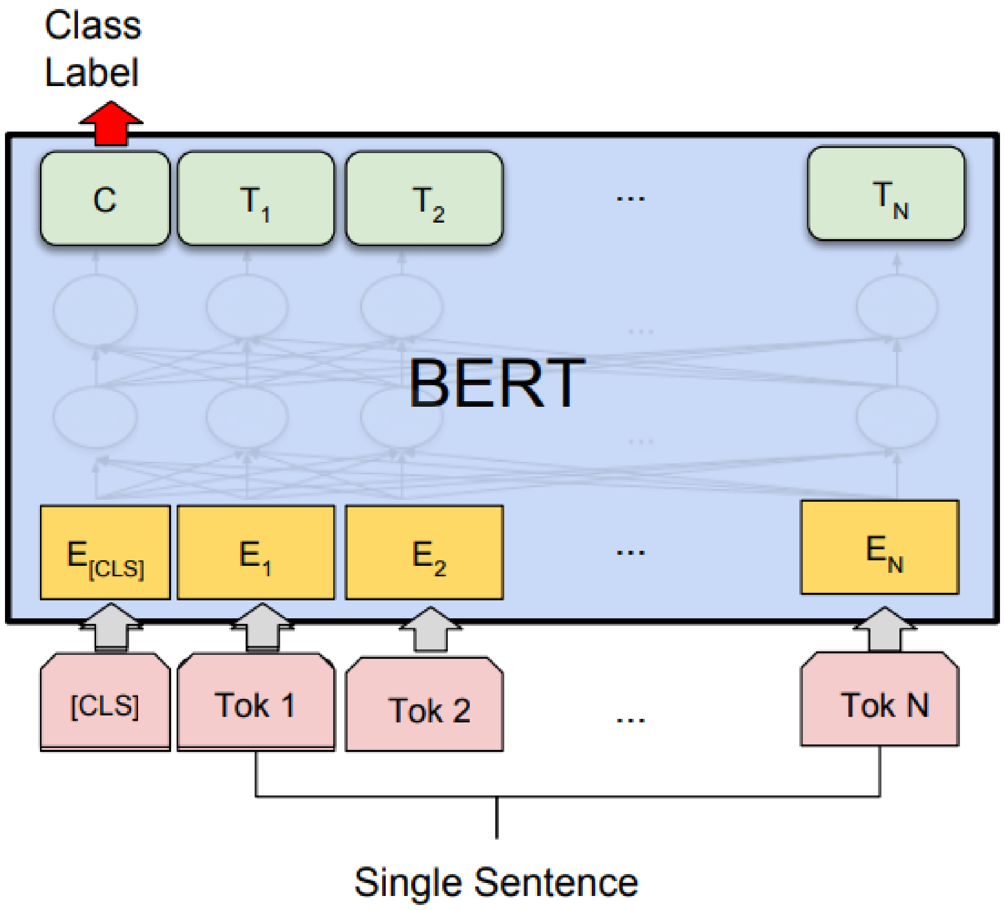
Sentiment Analysis

Sentiment Analysis

Sentiment analysis is a sentence classification task aiming at **automatically mapping data to their sentiment**.

It can be **binary** classification (e.g., positive or negative) or **multiclass** (e.g., enthusiasm, anger, etc)

Sentiment Analysis



Sentiment Analysis

The loss can be the likes of cross-entropy (CE), binary cross-entropy (BCE) or KL-Divergence (KL).

$$\mathcal{L}_{CE} = -\frac{1}{N}\sum_{n'=1}^{N} y^{(n)}.\, log(f(\mathbf{x},\theta)^{(n)})$$

$$\mathcal{L}_{BCE} = -y^{(n)}.\, log(f(\mathbf{x},\theta)^{(n)}) + (1 - y^{(n)}).\,(1 - f(\mathbf{x},\theta)^{(n)})$$

$$\mathcal{L}_{KL} = -\frac{1}{N}\sum_{n'=1}^{N} y^{(n)}.\, log(\frac{y^{(n)}}{f(\mathbf{x},\theta)^{(n)}})$$

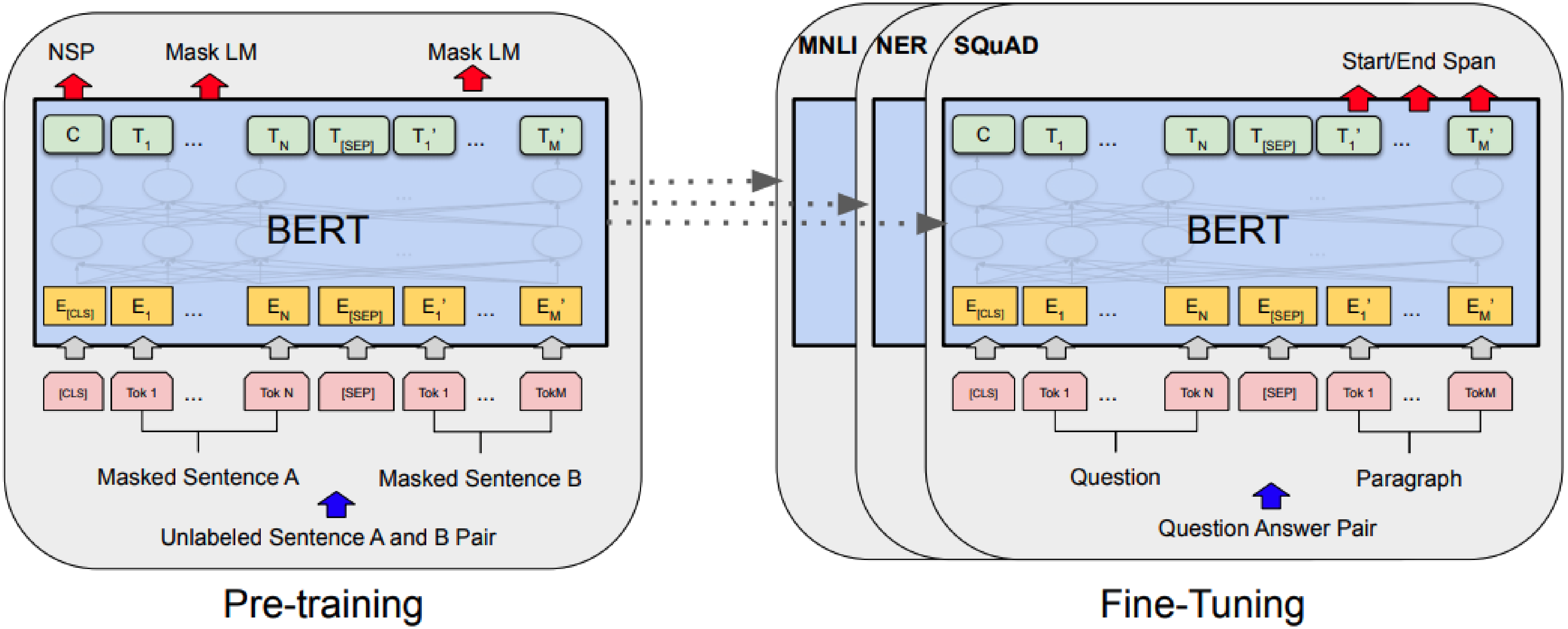
Question Answering (QA)

Question Answering (QA)

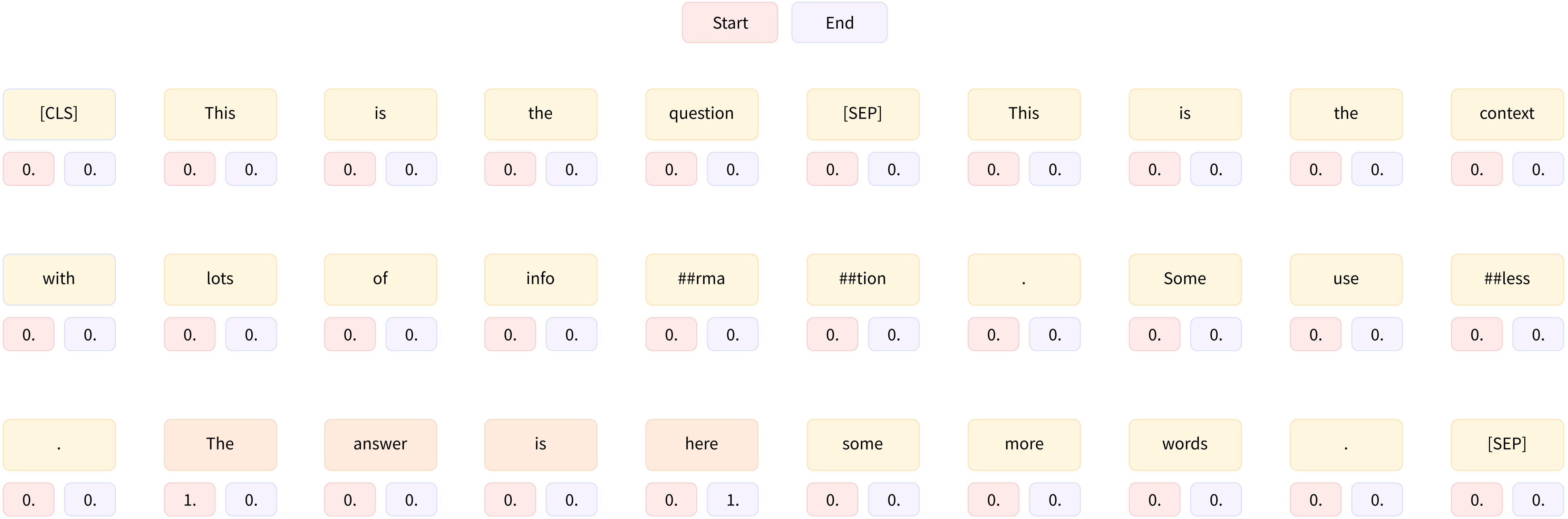
QA is the task of **retrieving a span of text from a context** that is best suited to answer a question.

This task is extractive -> **information retrieval**

Question Answering (QA)



Question Answering (QA)



	Start	End
[CLS]	0.	0.
This	0.	0.
is	0.	0.
the	0.	0.
question	0.	0.
[SEP]	0.	0.
This	0.	0.
is	0.	0.
the	0.	0.
context	0.	0.
with	0.	0.
lots	0.	0.
of	0.	0.
info	0.	0.
##rma	0.	0.
##tion	0.	0.
.	0.	0.
Some	0.	0.
use	0.	0.
##less	0.	0.
.	0.	0.
The	1.	0.
answer	0.	0.
is	0.	0.
here	0.	1.
some	0.	0.
more	0.	0.
words	0.	0.
.	0.	0.
[SEP]	0.	0.

Question Answering (QA)

The loss is the cross entropy over the output of the starting token and the ending one:

$$\mathcal{L}_{CE_{QA}} = \mathcal{L}_{CE_{start}} + \mathcal{L}_{CE_{end}}$$

Natural Language Inference (NLI)

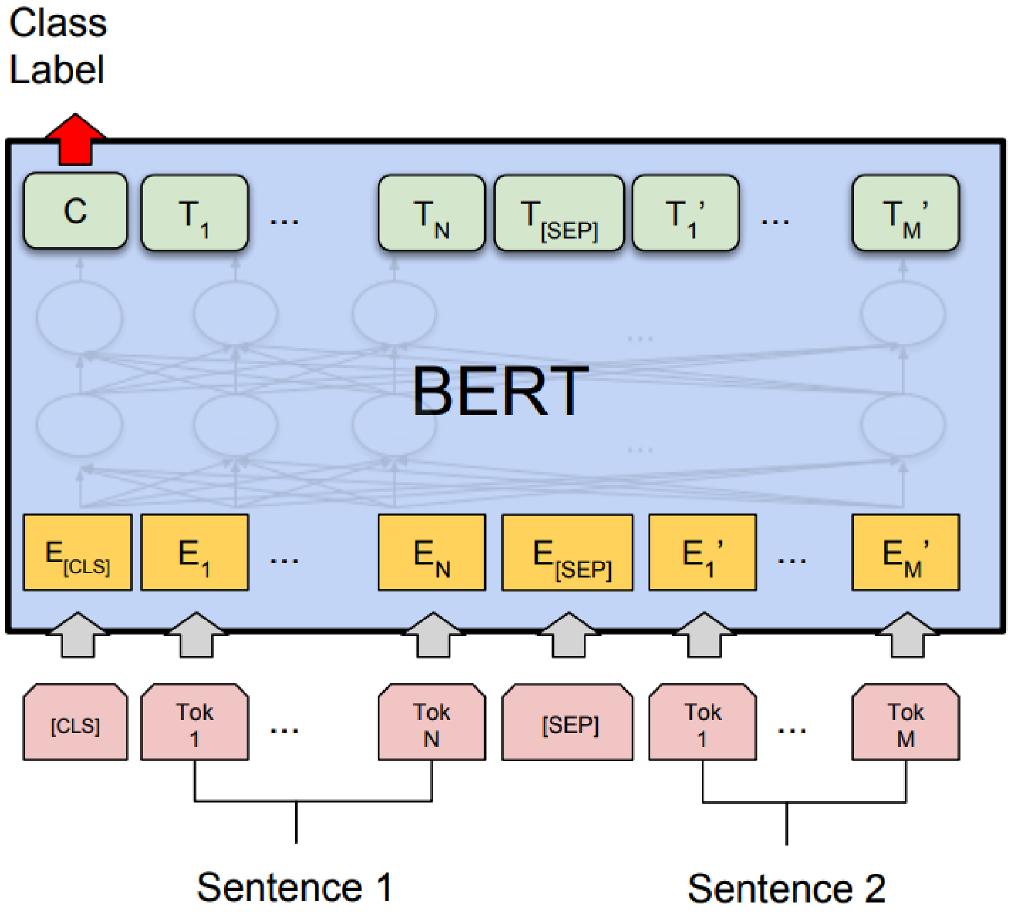
Natural Language Inference (NLI)

NLI is the task of **determining whether a "hypothesis" is true (entailment), false (contradiction), or undetermined (neutral)** given a "premise". [1]

Natural Language Inference (NLI)

Premise	Label	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction	The man is sleeping.
An older and younger man smiling.	neutral	Two men are smiling and laughing at the cats playing on the floor.
A soccer game with multiple males playing.	entailment	Some men are playing a sport.

Natural Language Inference (NLI)



Natural Language Inference (NLI)

The loss is simply the cross entropy or the divergence over the output of the `CLS` token and the true label.

$$\mathcal{L}_{NLI} = \mathcal{L}_{CE_{CLS}}$$

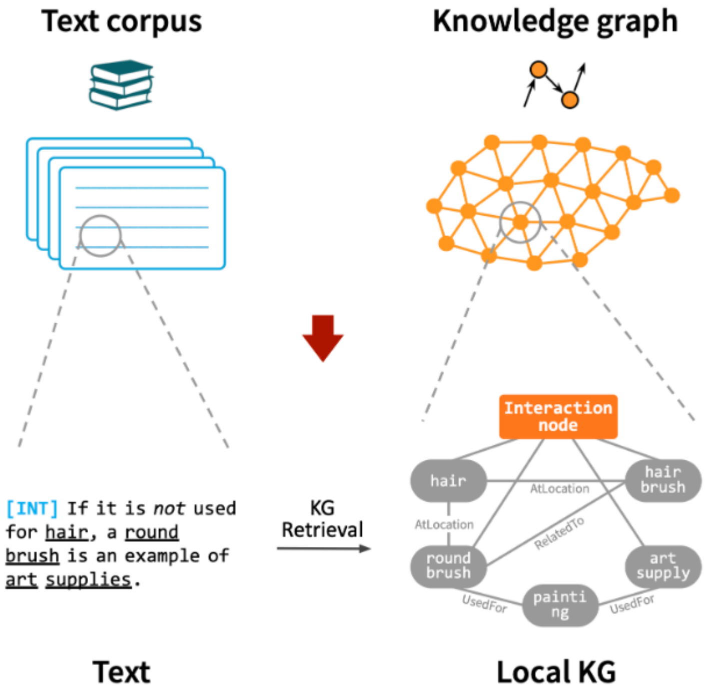
We are trying to compress the information about both sentence in one `CLS` token via attention and decide about their relationship.

Is it possible to help the model infering more information with les text data?

Going Further: LM as Knowledge Graphs

Yasunaga, M., Bosselut, A., Ren, H., Zhang, X., Manning, C. D., Liang, P. S., & Leskovec, J. (2022). Deep bidirectional language-knowledge graph pretraining. Advances in Neural Information Processing Systems, 35, 37309-37323.

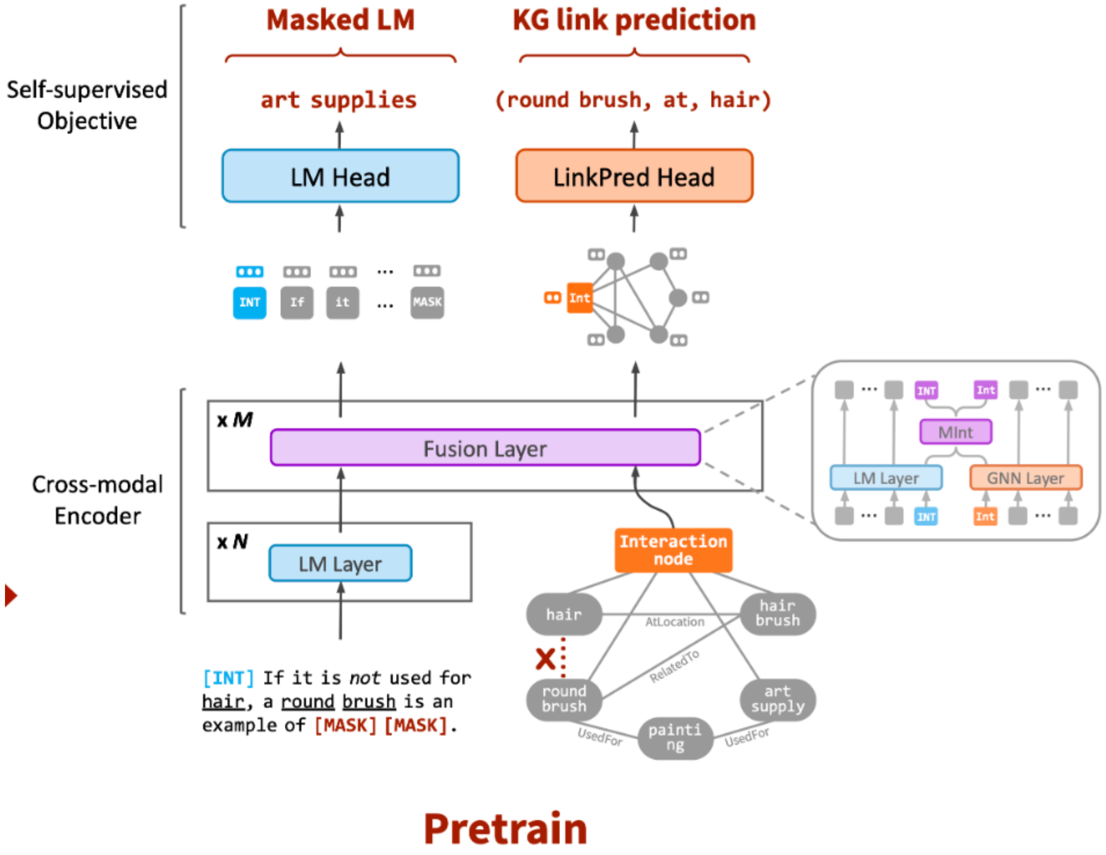
Going Further: LM as Knowledge Graphs



[INT] If it is *not* used for hair, a round brush is an example of art supplies.

Raw data

Going Further: LM as Knowledge Graphs



Going Further: LM as Knowledge Graphs

This architecture ***involves a KG ready to use beforeheaad and pre-training from scratch***. How can we better **perform NLP task without having to retrain or fine-tune** a model?

Exploit LLMs capacities: Chain-of-thoughts & In context Learning

Exploit LLMs capacities

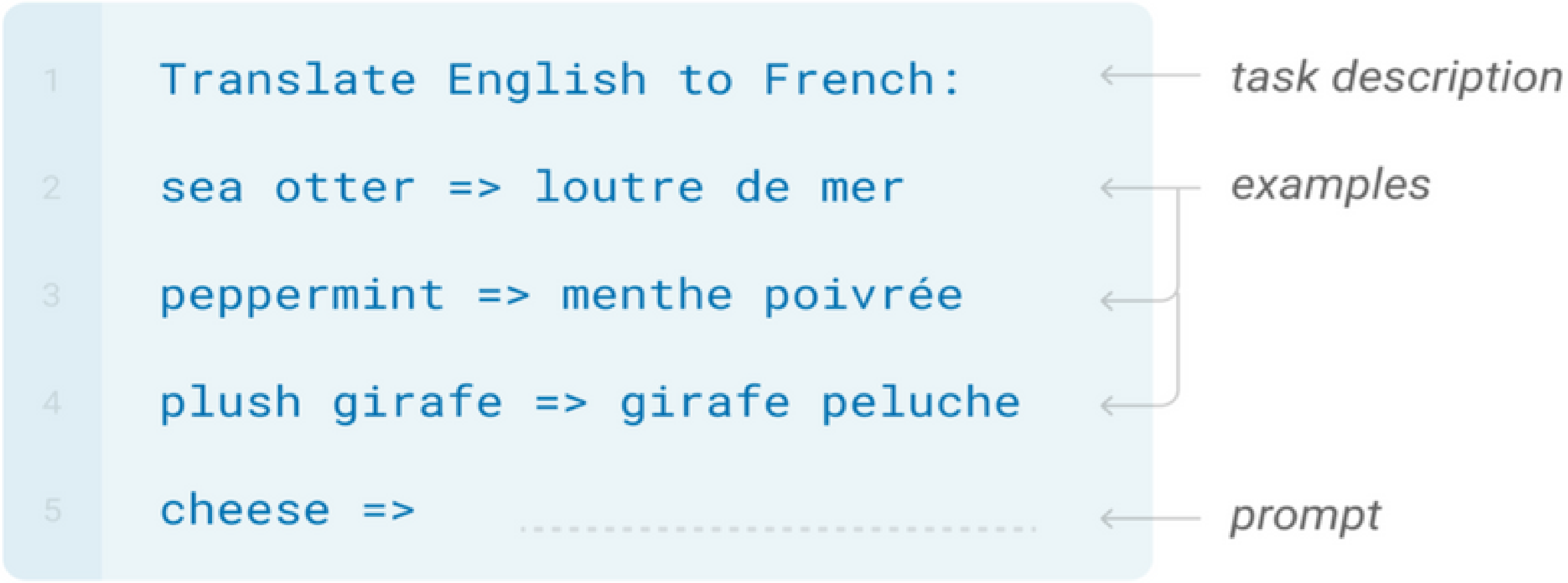
ICL enables LLMs to learn new tasks using natural language prompts without explicit retraining or fine-tuning.

The efficacy of ICL is closely tied to the model's scale, training data quality, and domain specificity.

Exploit LLMs capacities

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

task description
examples
prompt

Exploit LLMs capacities

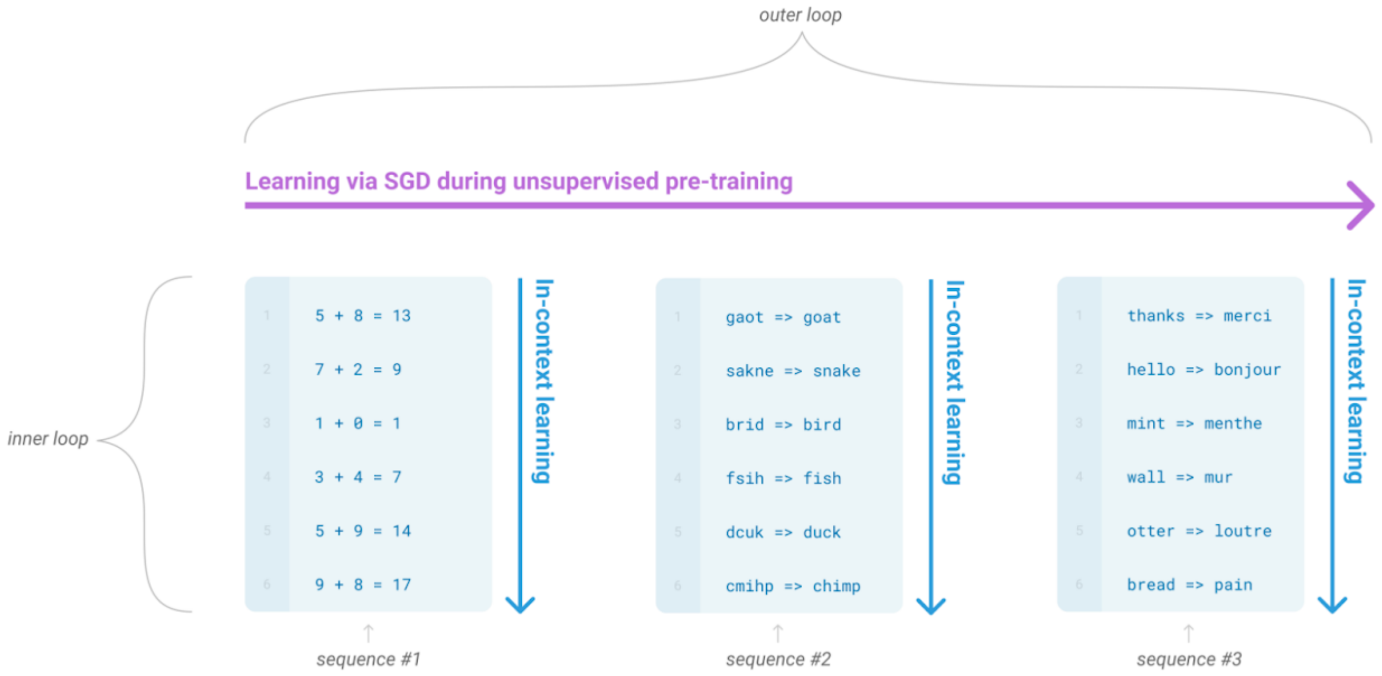


Figure 1.1: Language model meta-learning. During unsupervised pre-training, a language model develops a broad set of skills and pattern recognition abilities. It then uses these abilities at inference time to rapidly adapt to or recognize the desired task. We use the term "in-context learning" to describe the inner loop of this process, which occurs within the forward-pass upon each sequence. The sequences in this diagram are not intended to be representative of the data a model would see during pre-training, but are intended to show that there are sometimes repeated sub-tasks embedded within a single sequence.

Exploit LLMs capacities

$$p(\text{output}|\text{prompt}) = \int_{\text{concept}} p(\text{output}|\text{concept}, \text{prompt})\, p(\text{concept}|\text{prompt})\, d(\text{concept}). \qquad (1)$$

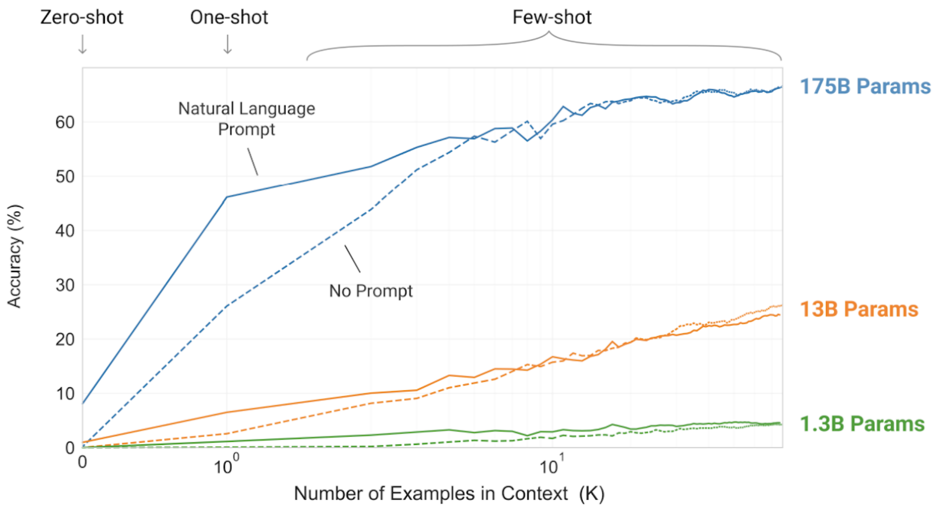
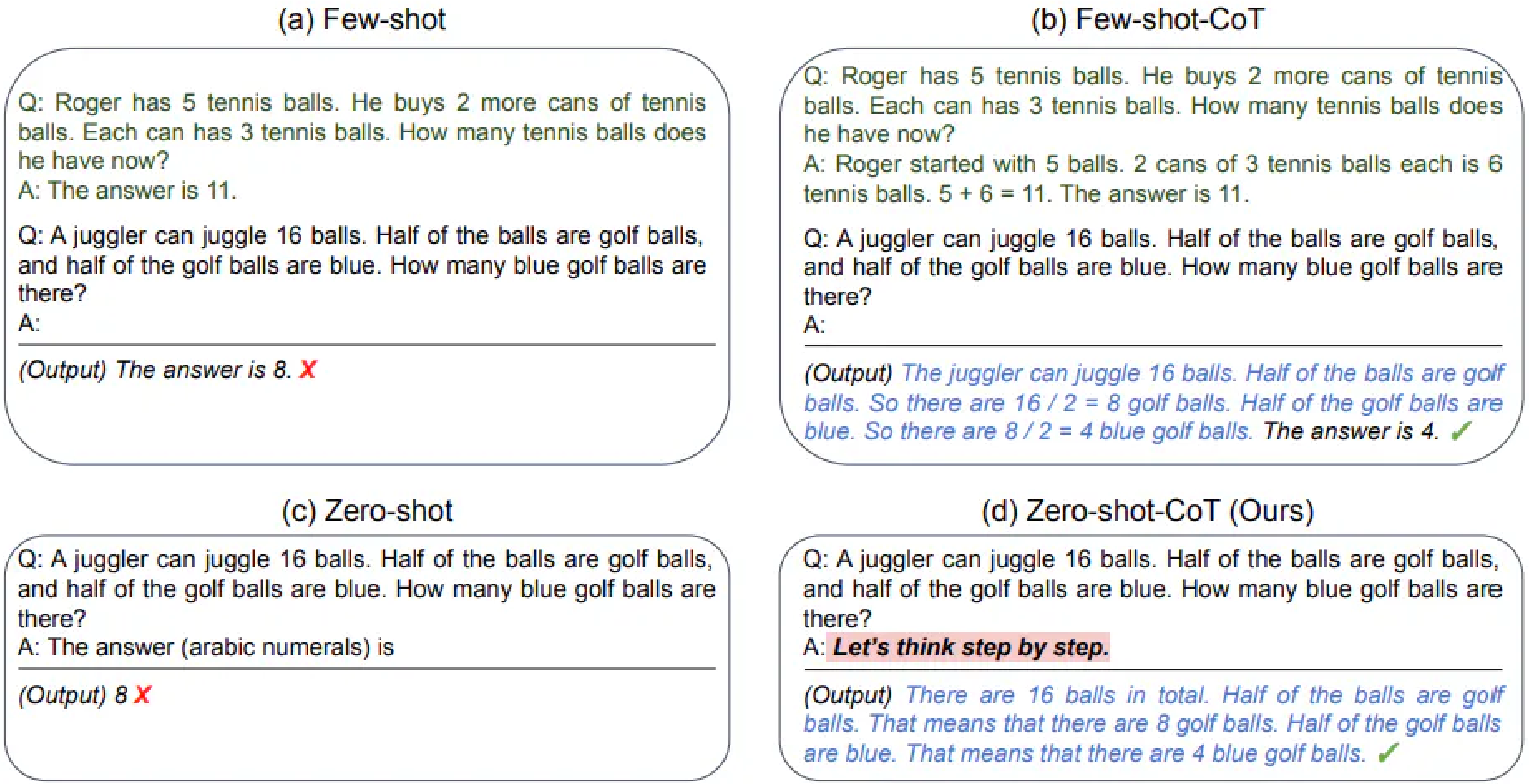


Figure 1.2: Larger models make increasingly efficient use of in-context information. We show in-context learning performance on a simple task requiring the model to remove random symbols from a word, both with and without a natural language task description (see Sec. 3.9.2). The steeper "in-context learning curves" for large models demonstrate improved ability to learn a task from contextual information. We see qualitatively similar behavior across a wide range of tasks.

Exploit LLMs capacities



(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. **X**

(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 **X**

(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. ✓

Questions?

References

[1] Gururangan, Suchin, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks." In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, edited by Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, 8342–60. Online: Association for Computational Linguistics, 2020.