Large Language Models

Prompting for Zero-Shot and Few-Shot Learning

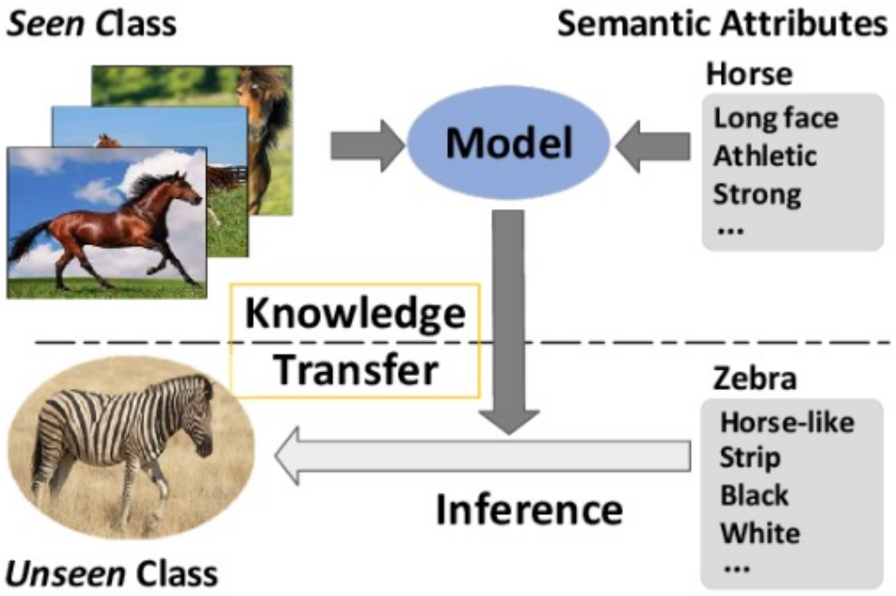
Mohammad Hossein Rohban

Fall 2023

Courtesy: Most of the slides are adopted from the course COS 597G and the paper “Making Pre-trained Language Models Better Few-shot Learners” b Gao et al.

What is Zero-Shot Learning?

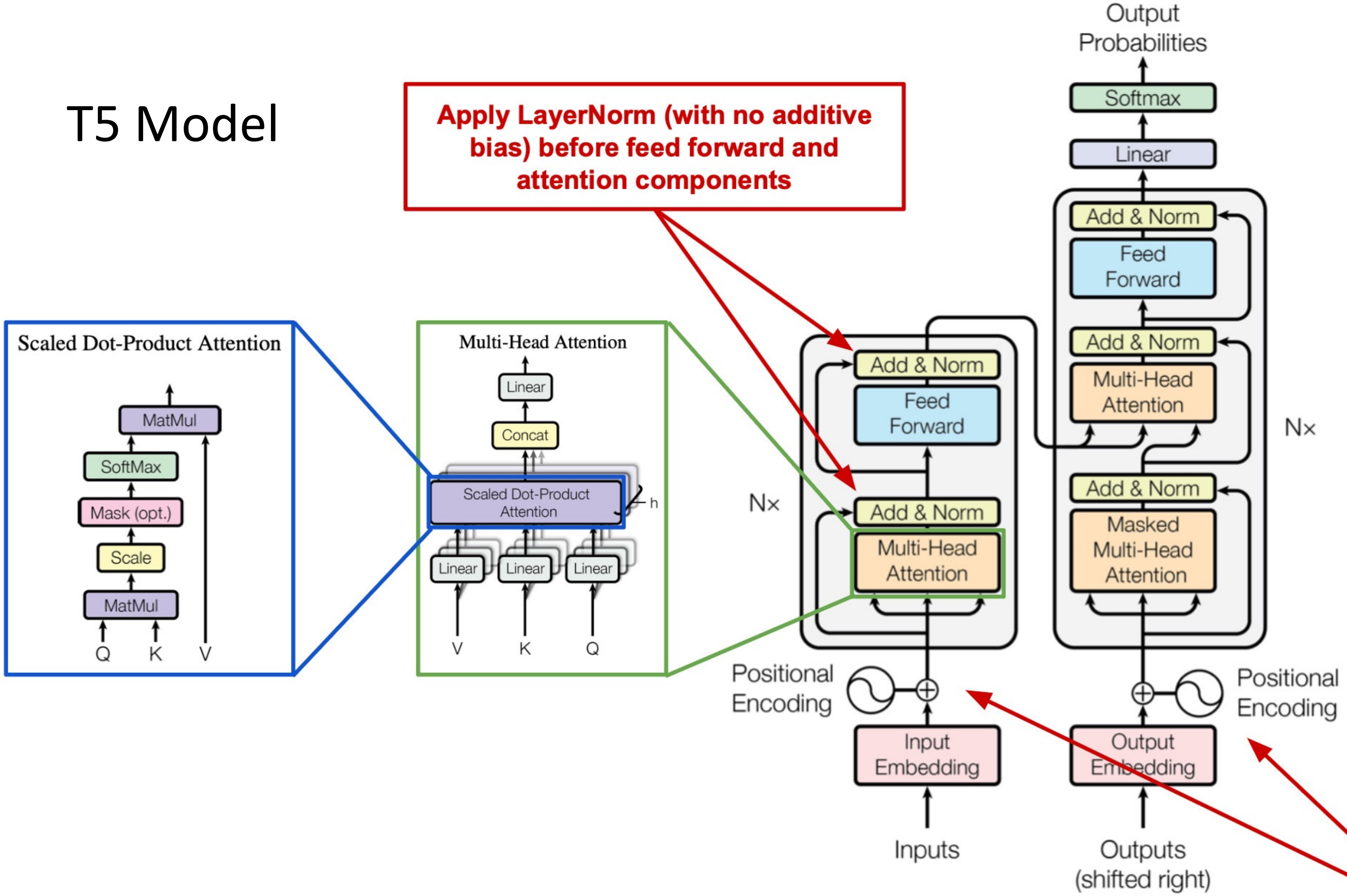
- Zero-Shot Learning (ZSL) [2009-]
  - Unseen test sample classes (or tasks) during training
  - Has to associate observed and non-observed classes
  - Auxiliary information is used to make this happen
  - e.g. a model trained to recognize horses along with textual info of how each animal looks like ☞ can classify zebras too!



ZSL (cont.)

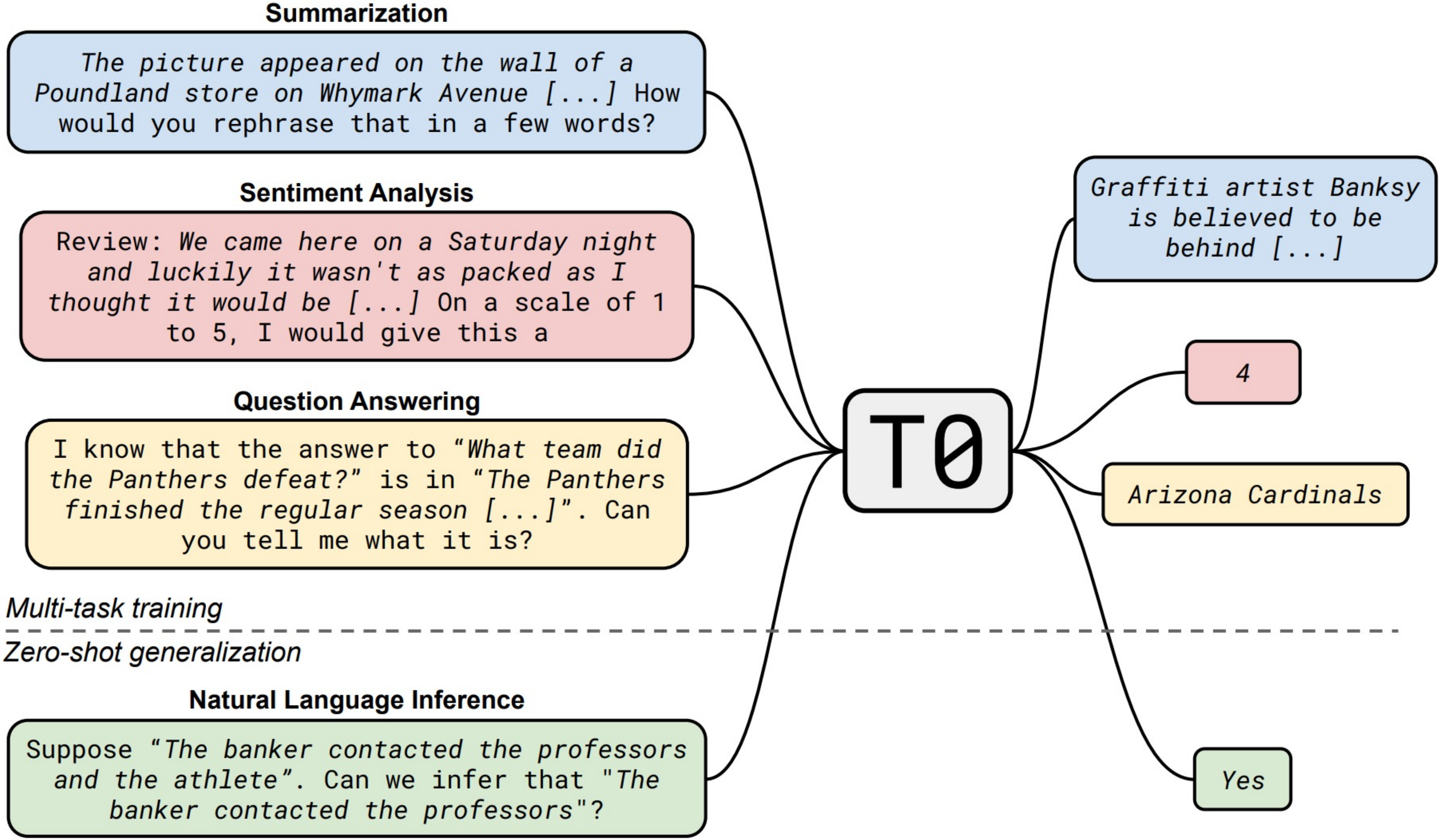
- T0: An encoder-decoder model
  - 16x smaller than GPT-3
  - Can generalize to unseen NLP tasks
  - Explicit multi-task learning to achieve ZSL.
  - Map any NLP task into a readable prompt.
  - Fine-tuned the T5 model on multi-task training dataset.
  - https://bigscience.huggingface.co/blog/t0

T5 Model



Apply LayerNorm (with no additive bias) before feed forward and attention components

Use a simplified positional encoding scheme



Summarization

The picture appeared on the wall of a Poundland store on Whymark Avenue [...] How would you rephrase that in a few words?

Sentiment Analysis

Review: *We came here on a Saturday night and luckily it wasn't as packed as I thought it would be [...]* On a scale of 1 to 5, I would give this a

Question Answering

I know that the answer to “*What team did the Panthers defeat?*” is in “*The Panthers finished the regular season [...]*”. Can you tell me what it is?

Multi-task training

Zero-shot generalization

Natural Language Inference

Suppose “*The banker contacted the professors and the athlete*”. Can we infer that "*The banker contacted the professors*"?

T0

Graffiti artist Banksy is believed to be behind [...]

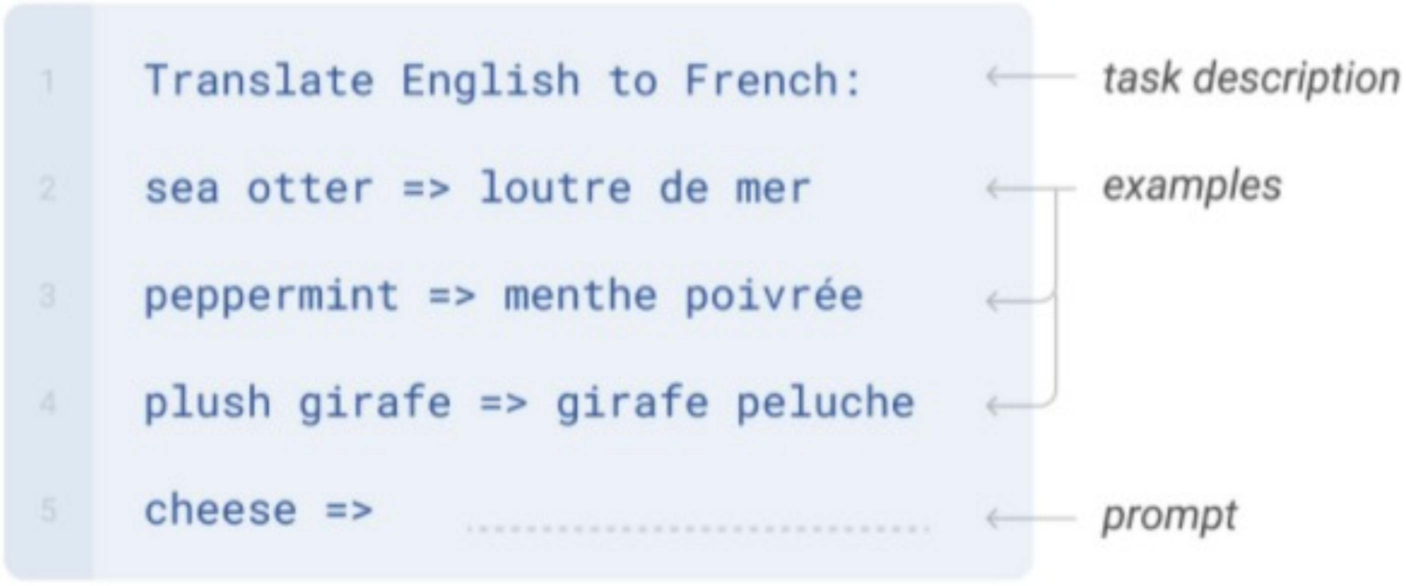
4

Arizona Cardinals

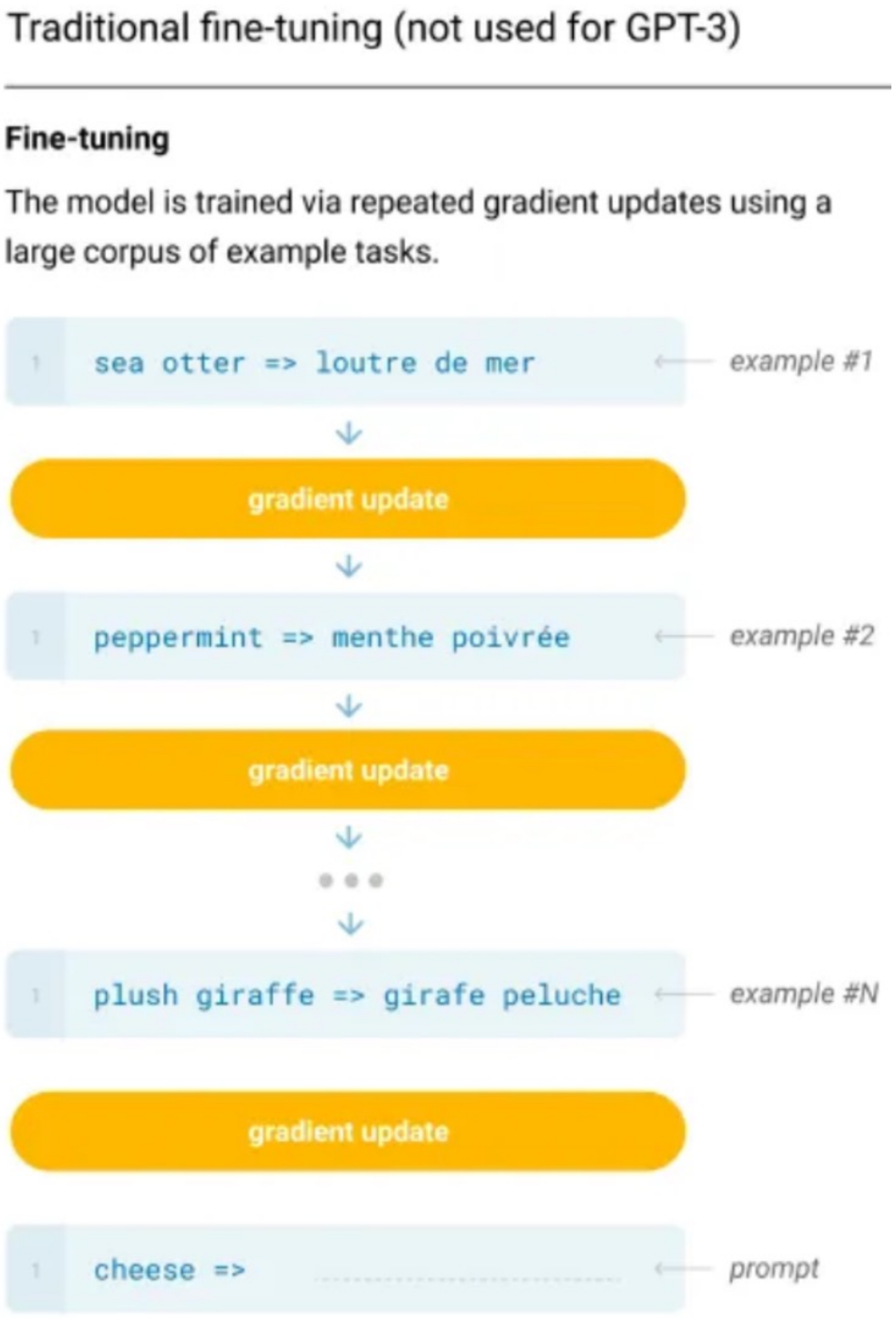
Yes

Few-Shot Learning

- Including few examples of test task at inference time.



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



Traditional fine-tuning (not used for GPT-3)

Fine-tuning

The model is trained via repeated gradient updates using a large corpus of example tasks.

```
sea otter => loutre de mer
```

← example #1

↓

gradient update

↓

```
peppermint => menthe poivrée
```

← example #2

↓

gradient update

↓

• • •

↓

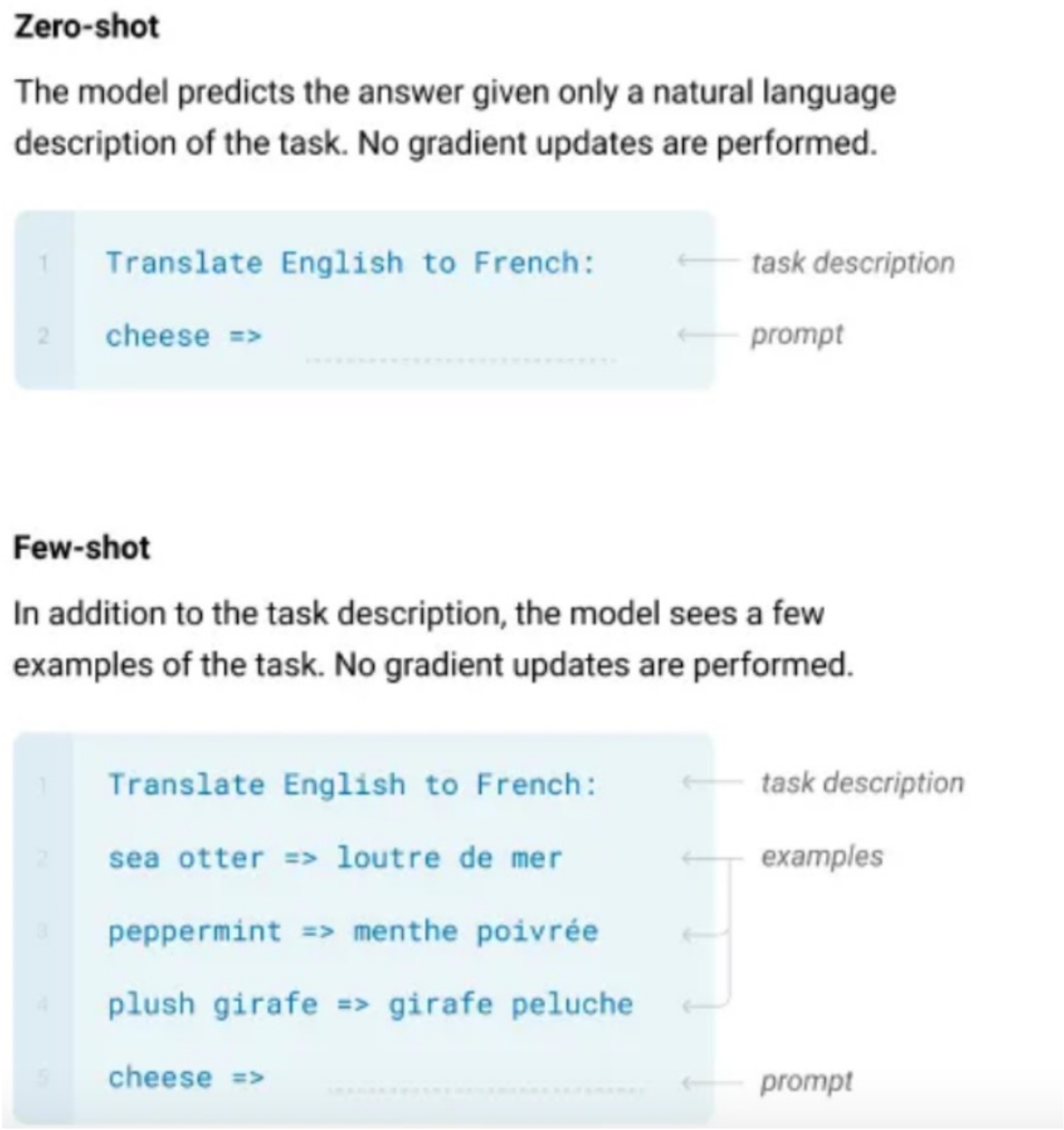
```
plush giraffe => girafe peluche
```

← example #N

gradient update

```
cheese =>
```

← prompt



Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
Translate English to French:
cheese =>
```

← task description

← prompt

Few-shot

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

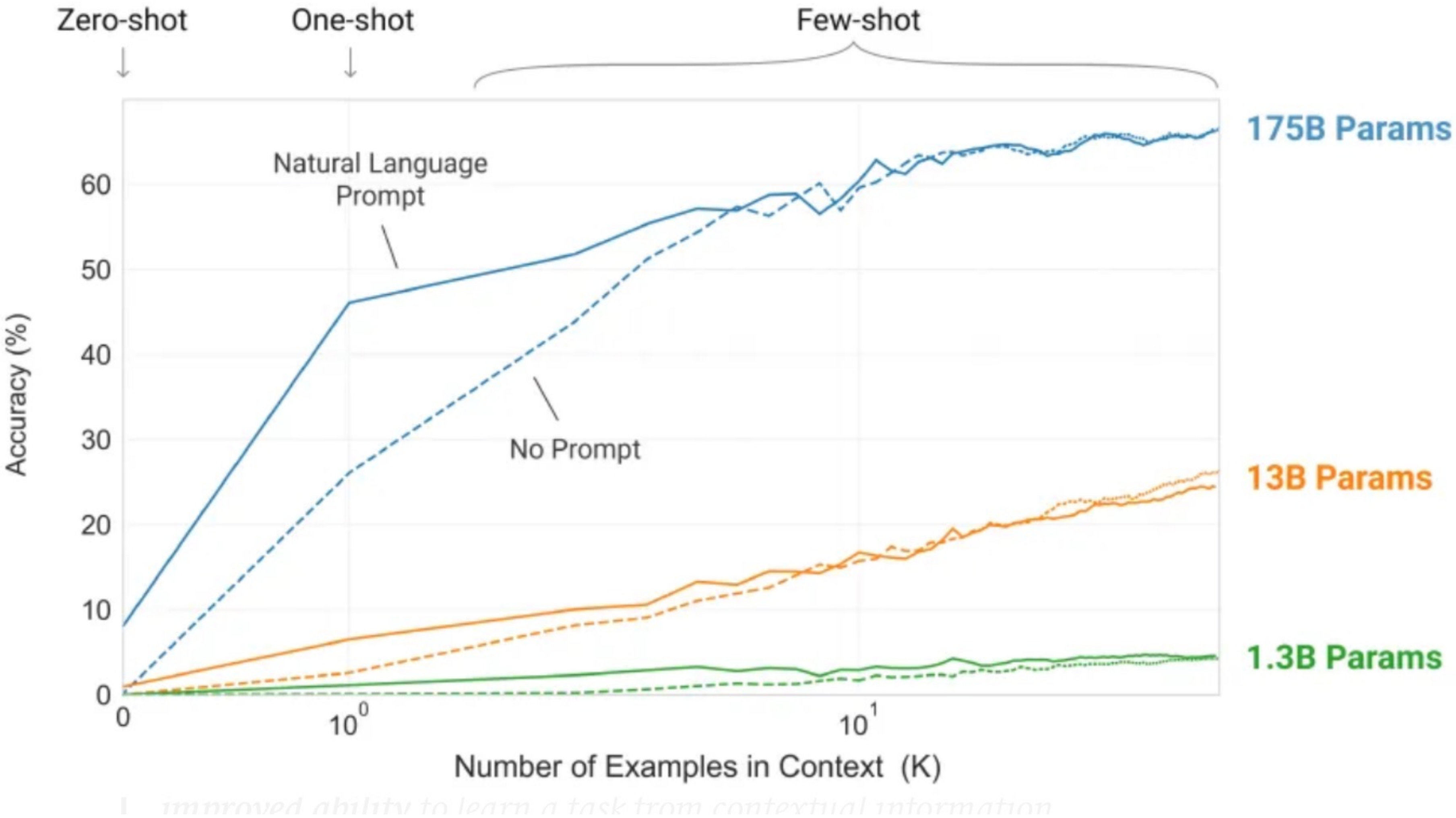
```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

← task description

← examples

← prompt

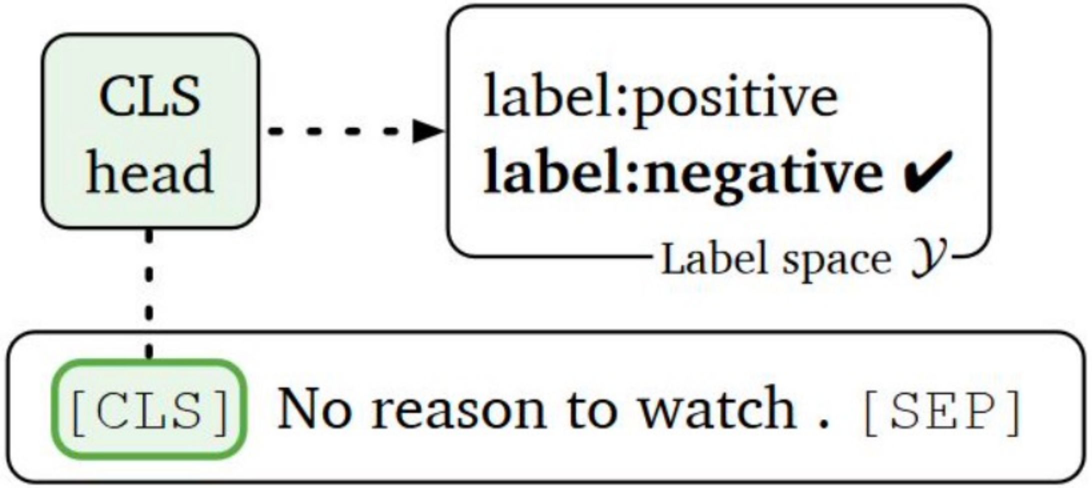
LLMs have ZSL and FSL capabilities



Let’s have a discussion

- Don’t have the luxury of deploying a 100-B parameter.
    - All we can afford is a pre-trained 100-M parameter model.
- Have only a couple of labeled examples from the target task.
    - Let’s say sentiment analysis of movies.
- How to go about this?

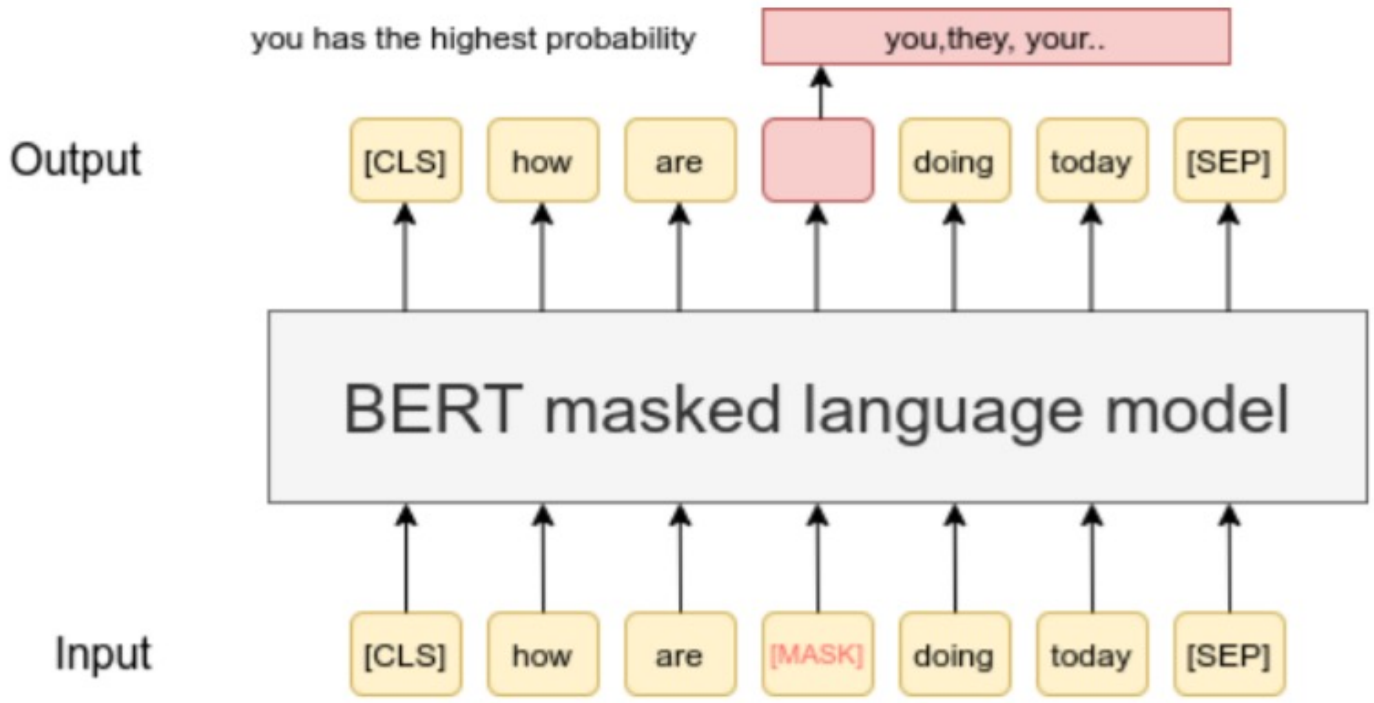
1st Solution: Head-based Fine-Tuning of a MLM



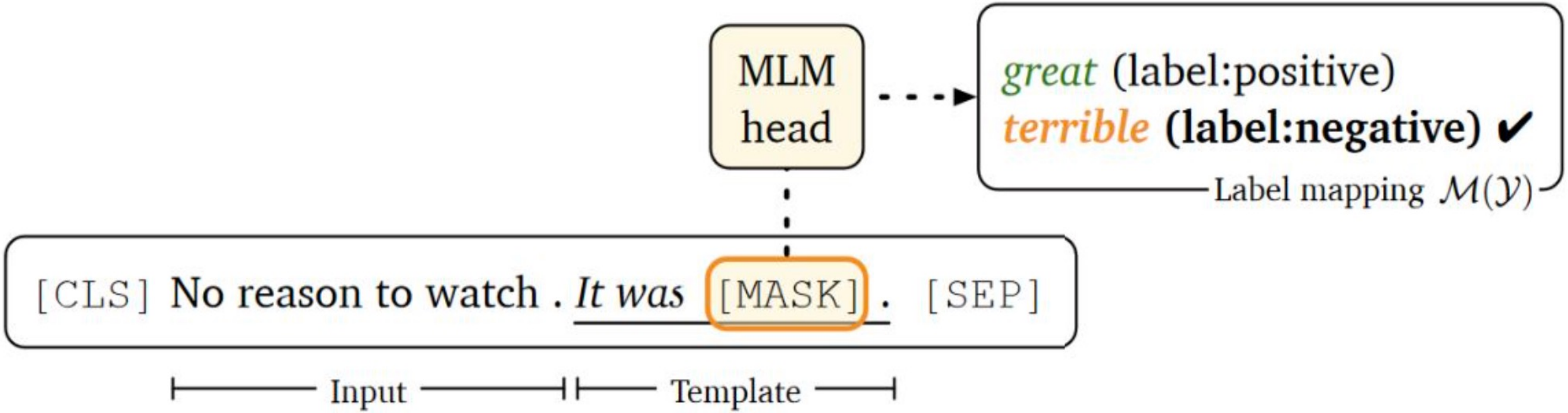
- How many trainable parameters are involved?
- hidden_size × num_classes
- Does it work well when given only ~10 training samples?

What else we can do? Let’s discuss.

- … which better suits the FSL setup?
- Utilizing the masked token prediction capability of the BERT.

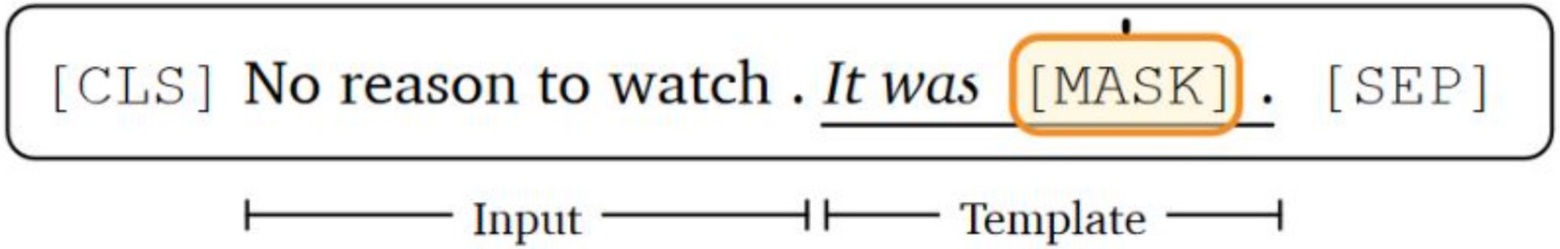


Prompt-based Fine-Tuning



Prompt-based Fine-Tuning (cont.)

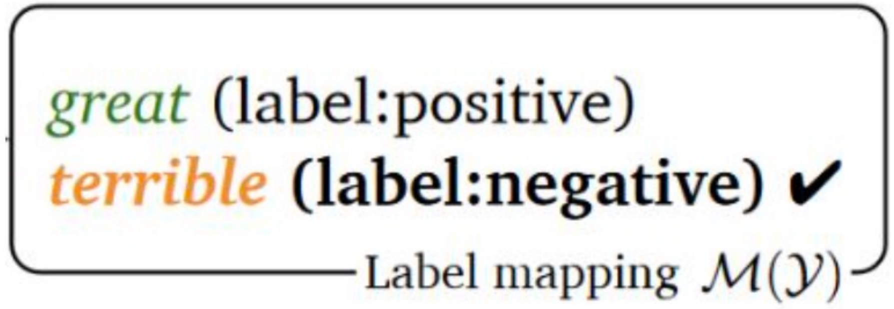
- Step 1: Formulate the task into a masked token prediction through a prompt template:



[CLS] No reason to watch . *It was* [MASK] . [SEP]

Input | Template

- Step 2: Choose a label-word mapping M.



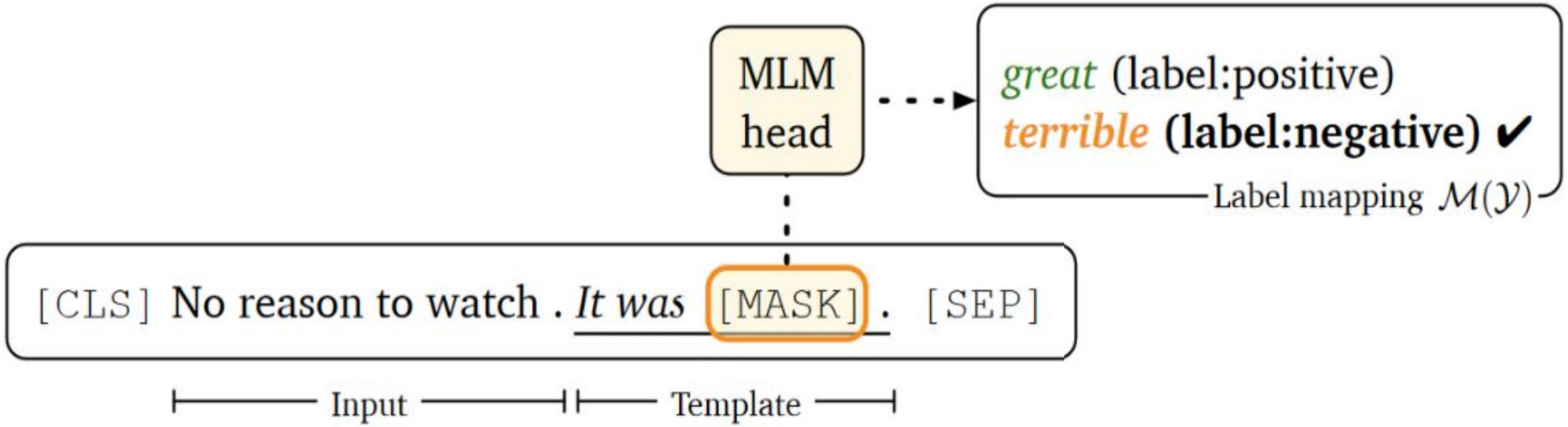
great (label:positive)
terrible **(label:negative)** ✓

Label mapping $\mathcal{M}(\mathcal{Y})$

Prompt-based Fine-Tuning (cont.)

- Step 3: Fine-tune the LM to fill in the correct word

$$p(y \mid x_{\text{in}}) = p\left(\texttt{[MASK]} = \mathcal{M}(y) \mid x_{\text{prompt}}\right)$$
$$= \frac{\exp\left(\mathbf{w}_{\mathcal{M}(y)} \cdot \mathbf{h}_{\texttt{[MASK]}}\right)}{\sum_{y' \in \mathcal{Y}} \exp\left(\mathbf{w}_{\mathcal{M}(y')} \cdot \mathbf{h}_{\texttt{[MASK]}}\right)},$$

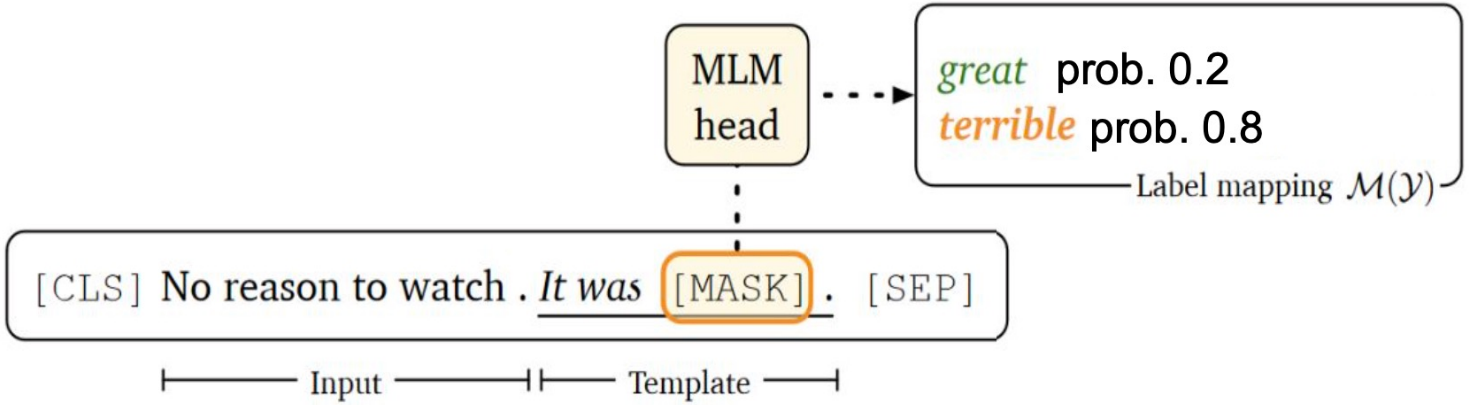


Regression Problem

- Regression: interpolating between two extremes

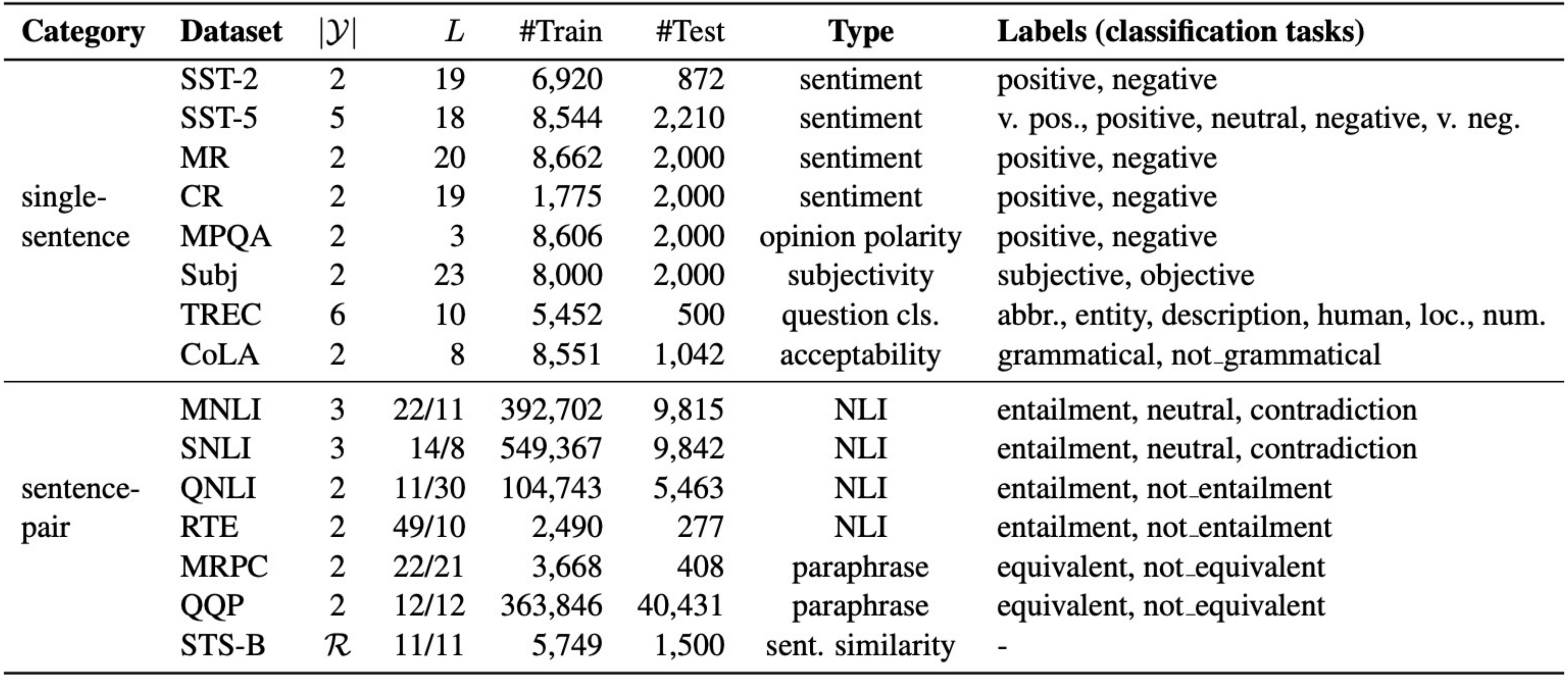
$$y = v_l \cdot p(y_l \mid x_{\text{in}}) + v_u \cdot p(y_u \mid x_{\text{in}})$$

- The LM is fine-tuned to minimize the KL-divergence between the inferred $P(y_u \mid x_{in})$ and $(y - v_l)/(v_u - v_l)$ the observed target.



Predicted y = 0.2

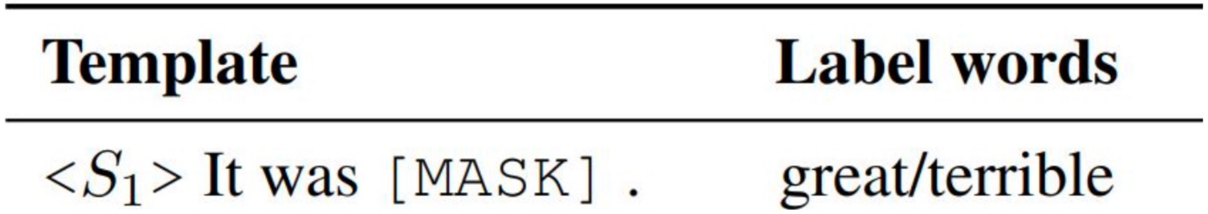
Evaluation Datasets



Category	Dataset	$\lvert\mathcal{Y}\rvert$	L	#Train	#Test	Type	Labels (classification tasks)
single-sentence	SST-2	2	19	6,920	872	sentiment	positive, negative
	SST-5	5	18	8,544	2,210	sentiment	v. pos., positive, neutral, negative, v. neg.
	MR	2	20	8,662	2,000	sentiment	positive, negative
	CR	2	19	1,775	2,000	sentiment	positive, negative
	MPQA	2	3	8,606	2,000	opinion polarity	positive, negative
	Subj	2	23	8,000	2,000	subjectivity	subjective, objective
	TREC	6	10	5,452	500	question cls.	abbr., entity, description, human, loc., num.
	CoLA	2	8	8,551	1,042	acceptability	grammatical, not_grammatical
sentence-pair	MNLI	3	22/11	392,702	9,815	NLI	entailment, neutral, contradiction
	SNLI	3	14/8	549,367	9,842	NLI	entailment, neutral, contradiction
	QNLI	2	11/30	104,743	5,463	NLI	entailment, not_entailment
	RTE	2	49/10	2,490	277	NLI	entailment, not_entailment
	MRPC	2	22/21	3,668	408	paraphrase	equivalent, not_equivalent
	QQP	2	12/12	363,846	40,431	paraphrase	equivalent, not_equivalent
	STS-B	$\mathcal{R}$	11/11	5,749	1,500	sent. similarity	-

Examples

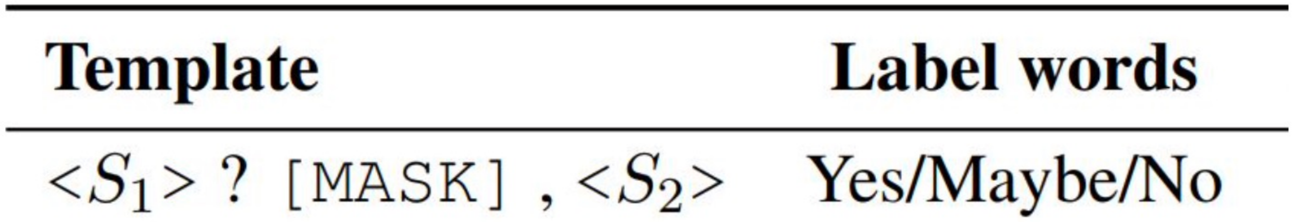
- SST-2: sentiment analysis.
- e.g. S1 = “The movie is ridiculous”. Label: negative.
- Manual prompt:



Template	**Label words**
$<S_1>$ It was `[MASK]` .	great/terrible

Examples (cont.)

- SNLI: Natural Language Inference
- S1 = “A soccer game with multiple males playing”. S2 =“Some men are playing sport”. Label: Entailment.
- Manual prompt:

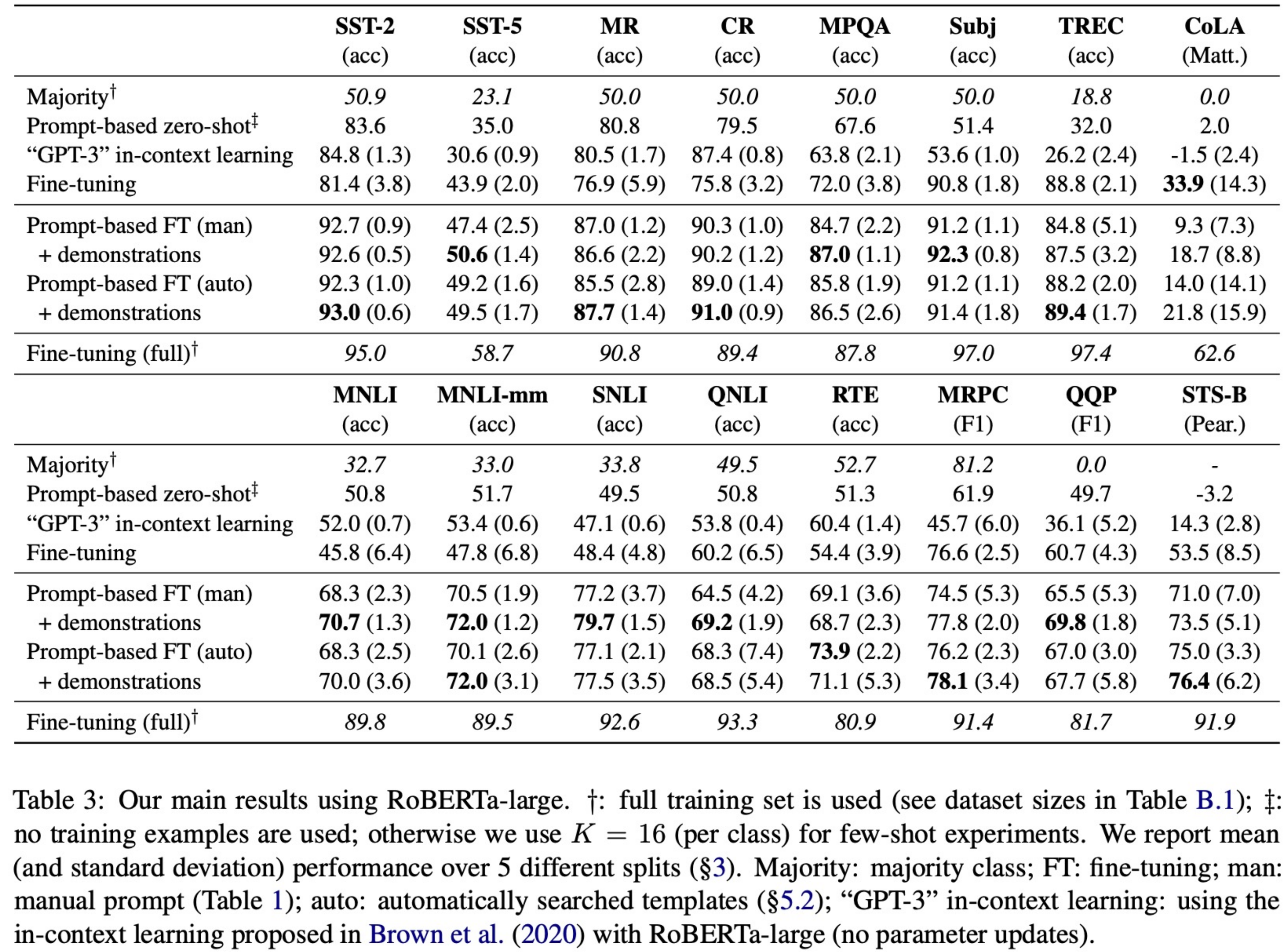


Template	**Label words**
$<S_1>$? `[MASK]` , $<S_2>$	Yes/Maybe/No

Few-shot Learning & Evaluation Protocol

- Training dataset: K=16 examples per class.
- Dev dataset: same size as training dataset.
- Performance measured across 5 random splits of {train, dev} set.

Results

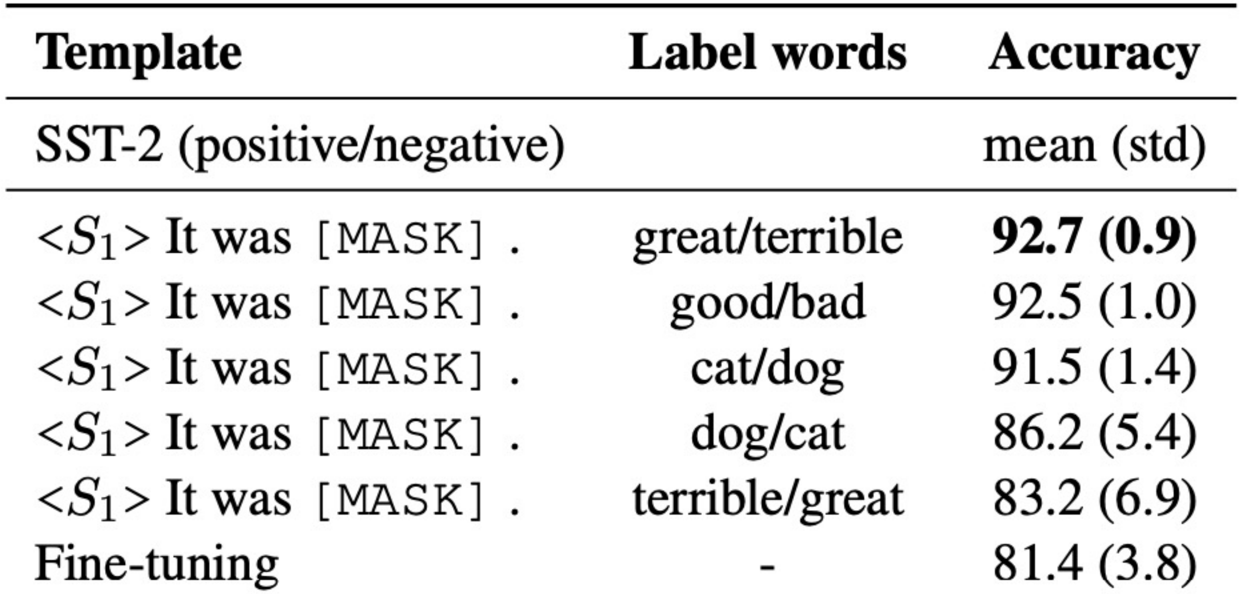


	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)	MPQA (acc)	Subj (acc)	TREC (acc)	CoLA (Matt.)
Majority†	*50.9*	*23.1*	*50.0*	*50.0*	*50.0*	*50.0*	*18.8*	*0.0*
Prompt-based zero-shot‡	83.6	35.0	80.8	79.5	67.6	51.4	32.0	2.0
"GPT-3" in-context learning	84.8 (1.3)	30.6 (0.9)	80.5 (1.7)	87.4 (0.8)	63.8 (2.1)	53.6 (1.0)	26.2 (2.4)	-1.5 (2.4)
Fine-tuning	81.4 (3.8)	43.9 (2.0)	76.9 (5.9)	75.8 (3.2)	72.0 (3.8)	90.8 (1.8)	88.8 (2.1)	**33.9** (14.3)
Prompt-based FT (man)	92.7 (0.9)	47.4 (2.5)	87.0 (1.2)	90.3 (1.0)	84.7 (2.2)	91.2 (1.1)	84.8 (5.1)	9.3 (7.3)
+ demonstrations	92.6 (0.5)	**50.6** (1.4)	86.6 (2.2)	90.2 (1.2)	**87.0** (1.1)	**92.3** (0.8)	87.5 (3.2)	18.7 (8.8)
Prompt-based FT (auto)	92.3 (1.0)	49.2 (1.6)	85.5 (2.8)	89.0 (1.4)	85.8 (1.9)	91.2 (1.1)	88.2 (2.0)	14.0 (14.1)
+ demonstrations	**93.0** (0.6)	49.5 (1.7)	**87.7** (1.4)	**91.0** (0.9)	86.5 (2.6)	91.4 (1.8)	**89.4** (1.7)	21.8 (15.9)
Fine-tuning (full)†	*95.0*	*58.7*	*90.8*	*89.4*	*87.8*	*97.0*	*97.4*	*62.6*

	MNLI (acc)	MNLI-mm (acc)	SNLI (acc)	QNLI (acc)	RTE (acc)	MRPC (F1)	QQP (F1)	STS-B (Pear.)
Majority†	*32.7*	*33.0*	*33.8*	*49.5*	*52.7*	*81.2*	*0.0*	-
Prompt-based zero-shot‡	50.8	51.7	49.5	50.8	51.3	61.9	49.7	-3.2
"GPT-3" in-context learning	52.0 (0.7)	53.4 (0.6)	47.1 (0.6)	53.8 (0.4)	60.4 (1.4)	45.7 (6.0)	36.1 (5.2)	14.3 (2.8)
Fine-tuning	45.8 (6.4)	47.8 (6.8)	48.4 (4.8)	60.2 (6.5)	54.4 (3.9)	76.6 (2.5)	60.7 (4.3)	53.5 (8.5)
Prompt-based FT (man)	68.3 (2.3)	70.5 (1.9)	77.2 (3.7)	64.5 (4.2)	69.1 (3.6)	74.5 (5.3)	65.5 (5.3)	71.0 (7.0)
+ demonstrations	**70.7** (1.3)	**72.0** (1.2)	**79.7** (1.5)	**69.2** (1.9)	68.7 (2.3)	77.8 (2.0)	**69.8** (1.8)	73.5 (5.1)
Prompt-based FT (auto)	68.3 (2.5)	70.1 (2.6)	77.1 (2.1)	68.3 (7.4)	**73.9** (2.2)	76.2 (2.3)	67.0 (3.0)	75.0 (3.3)
+ demonstrations	70.0 (3.6)	**72.0** (3.1)	77.5 (3.5)	68.5 (5.4)	71.1 (5.3)	**78.1** (3.4)	67.7 (5.8)	**76.4** (6.2)
Fine-tuning (full)†	*89.8*	*89.5*	*92.6*	*93.3*	*80.9*	*91.4*	*81.7*	*91.9*

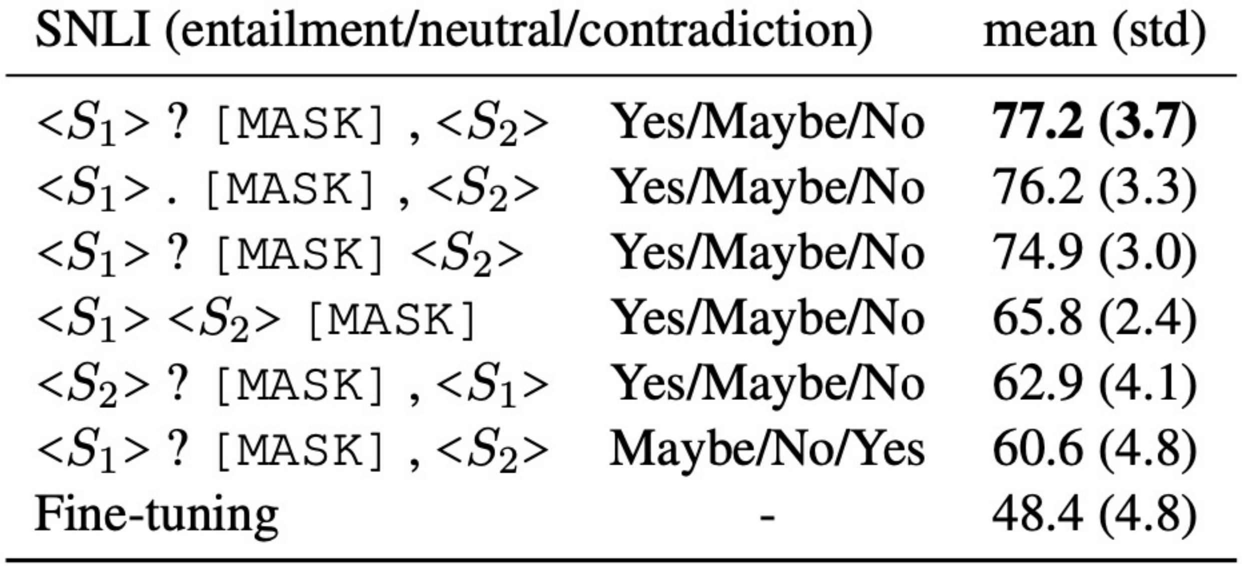
Table 3: Our main results using RoBERTa-large. †: full training set is used (see dataset sizes in Table B.1); ‡: no training examples are used; otherwise we use $K = 16$ (per class) for few-shot experiments. We report mean (and standard deviation) performance over 5 different splits (§3). Majority: majority class; FT: fine-tuning; man: manual prompt (Table 1); auto: automatically searched templates (§5.2); "GPT-3" in-context learning: using the in-context learning proposed in Brown et al. (2020) with RoBERTa-large (no parameter updates).

Effect of Word-Class Mapping



Template	**Label words**	**Accuracy**
SST-2 (positive/negative)		mean (std)
<S_1> It was `[MASK]` .	great/terrible	**92.7 (0.9)**
<S_1> It was `[MASK]` .	good/bad	92.5 (1.0)
<S_1> It was `[MASK]` .	cat/dog	91.5 (1.4)
<S_1> It was `[MASK]` .	dog/cat	86.2 (5.4)
<S_1> It was `[MASK]` .	terrible/great	83.2 (6.9)
Fine-tuning	-	81.4 (3.8)

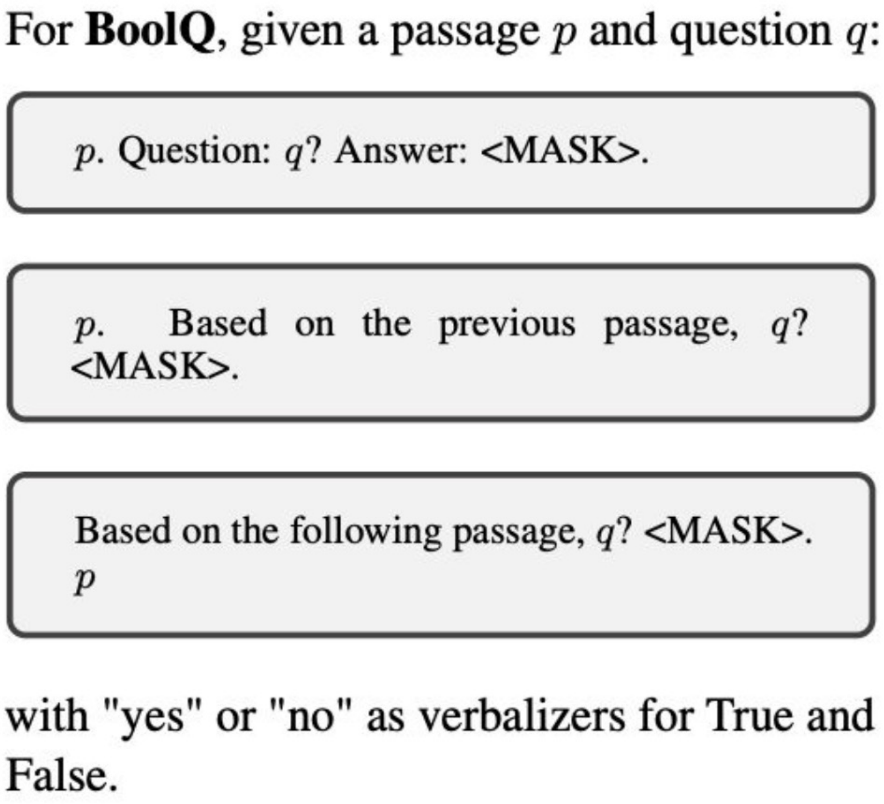
Effect of the Prompt Template



SNLI (entailment/neutral/contradiction)		mean (std)
<S_1> ? `[MASK]` , <S_2>	Yes/Maybe/No	**77.2 (3.7)**
<S_1> . `[MASK]` , <S_2>	Yes/Maybe/No	76.2 (3.3)
<S_1> ? `[MASK]` <S_2>	Yes/Maybe/No	74.9 (3.0)
<S_1> <S_2> `[MASK]`	Yes/Maybe/No	65.8 (2.4)
<S_2> ? `[MASK]` , <S_1>	Yes/Maybe/No	62.9 (4.1)
<S_1> ? `[MASK]` , <S_2>	Maybe/No/Yes	60.6 (4.8)
Fine-tuning	-	48.4 (4.8)

How to design good prompts?

- BoolQ: given a passage q and question p, design a prompt for question answering.



For **BoolQ**, given a passage p and question q:

p. Question: q? Answer: <MASK>.

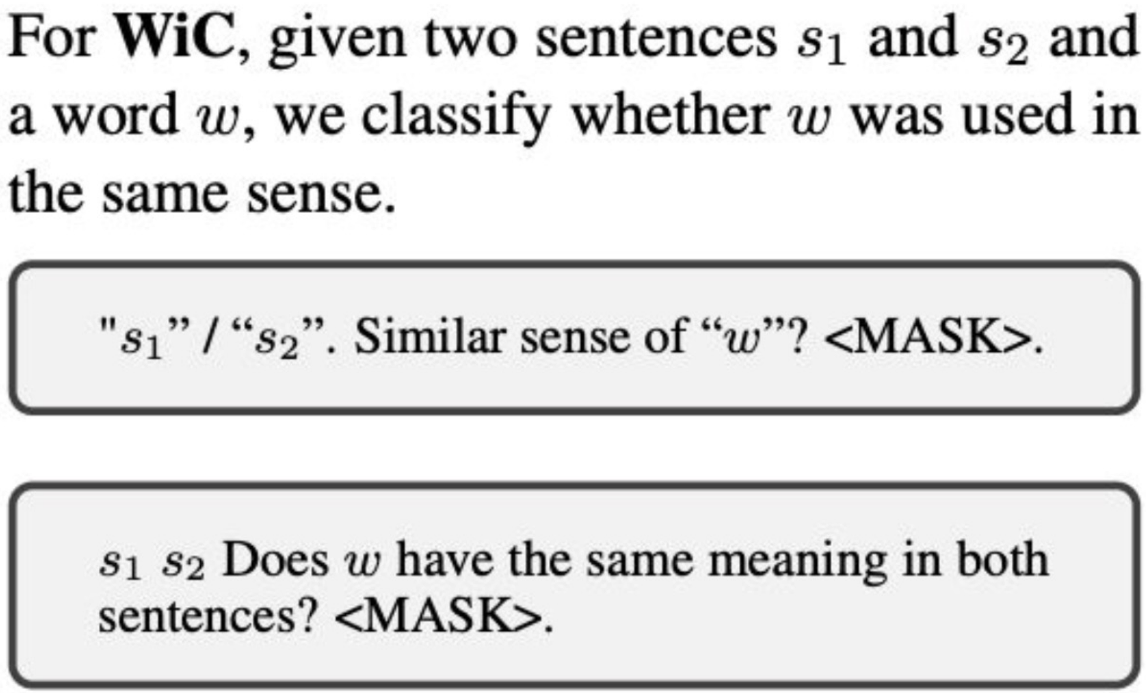
p. Based on the previous passage, q? <MASK>.

Based on the following passage, q? <MASK>.
p

with "yes" or "no" as verbalizers for True and False.

How to design good prompts? (cont.)

- WiC: given two sentences S1 and S2, and a word W, design a prompt to determine whether W was used in the same sense in both sentences.



For **WiC**, given two sentences s_1 and s_2 and a word w, we classify whether w was used in the same sense.

"s_1" / "s_2". Similar sense of "w"? <MASK>.

s_1 s_2 Does w have the same meaning in both sentences? <MASK>.

How to design good prompts? (cont.)

- Manual designing requires some effort.
- The template T and word-class mapping M are not independent.
- Model selection (T, M) is subject to overfitting.

Automatic Selection of Label Words

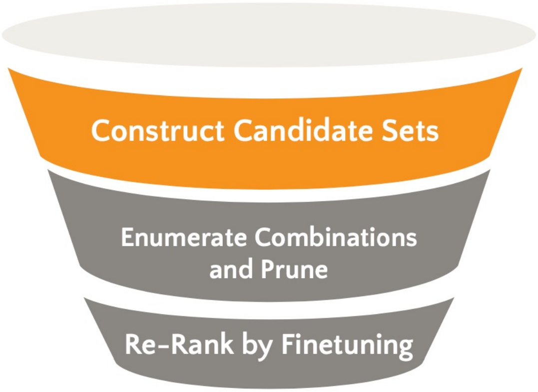
- Why naively searching all possibilities is not working?
- Generally interactable, exponentially large search space.
- Prone to overfitting. May uncover spurious correlations using few samples.
- For each class c, select top k words according to

$$\underset{v \in \mathcal{V}}{\text{Top-}k} \left\{ \sum_{x_{\text{in}} \in \mathcal{D}_{\text{train}}^{c}} \log P_{\mathcal{L}}\Big(\texttt{[MASK]} = v \mid \mathcal{T}(x_{\text{in}}) \Big) \right\}$$

- D^{c}_{train} is training set for the class c.

Automatic Selection of Label Words (cont.)

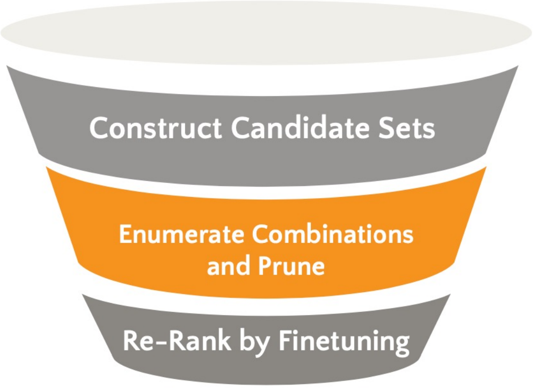
- Enumerate all combinations of top-k words for different classes.
- Prune by zero-shot accuracy on the training set, select top-n tuples.
- Fine-tune based on top-n candidate and select the best one on the dev set.



Given the **manual** template: <S> It was [MASK] .

label:positive
good
great
perfect
…

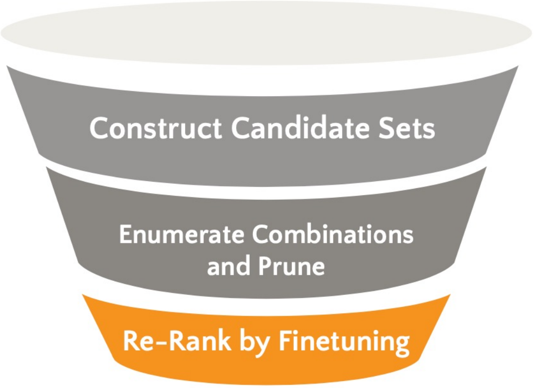
label:negative
awful
bad
terrible
…



good/awful
good/bad
…
perfect/bad
perfect/terrible

Prune →

good/bad
great/bad
great/terrible
perfect/terrible



Given the **manual** template: <S> It was [MASK] .

good/bad
great/bad
great/terrible
perfect/terrible

→ Fine-tune and evaluate on $\mathscr{D}_{dev}$

good/bad (85%)
great/bad (82%)
great/terrible (91%)
perfect/terrible (86%)

Automatic Generation of Templates

- Having fixed M(y), use the T5 model.
  - Trained to fill in multiple tokens.
  - e.g. “Thank you <X> to your party <Y> week” with X = “inviting me” and Y = “last”
- Let $T_g(x_{in}, y)$ be the formulation for making the T5 input:

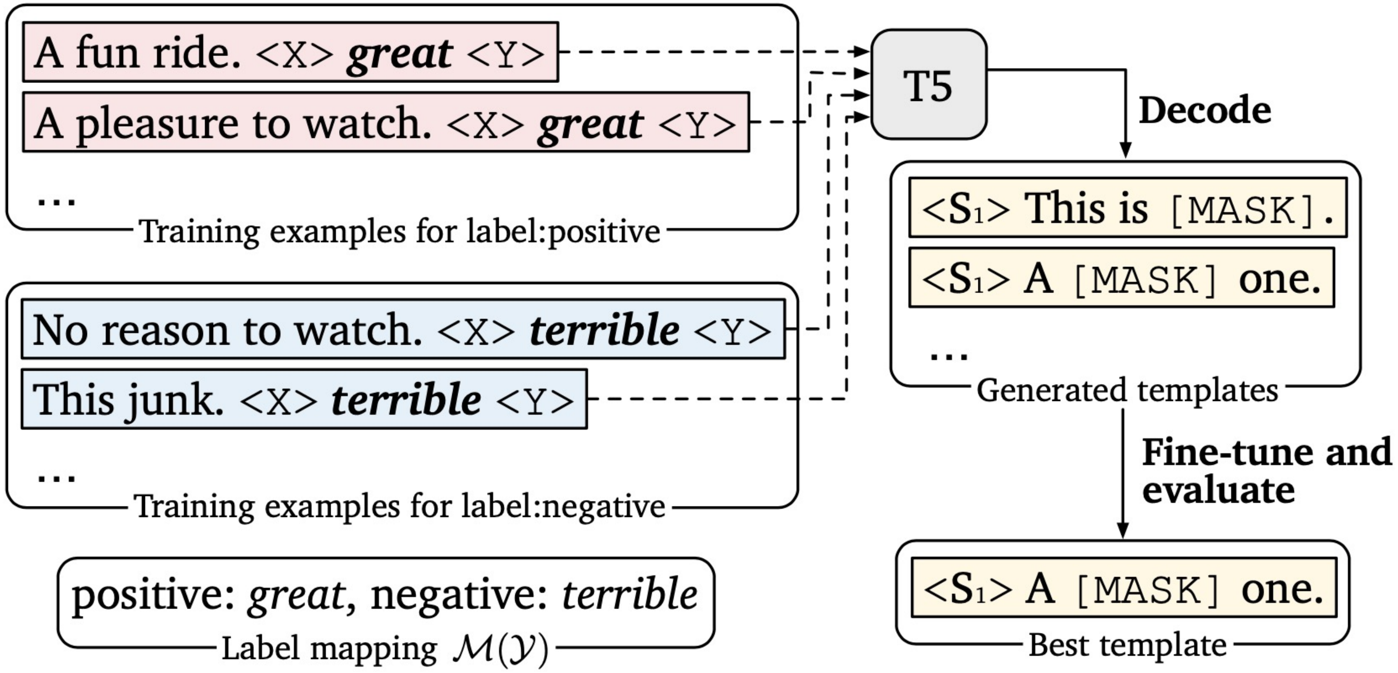
$$
\begin{aligned}
{<}S_1{>} &\longrightarrow\ {<}\mathrm{X}{>}\ \mathcal{M}(y)\ {<}\mathrm{Y}{>}\ {<}S_1{>}, \\
{<}S_1{>} &\longrightarrow\ {<}S_1{>}\ {<}\mathrm{X}{>}\ \mathcal{M}(y)\ {<}\mathrm{Y}{>}, \\
{<}S_1{>}, {<}S_2{>} &\longrightarrow\ {<}S_1{>}\ {<}\mathrm{X}{>}\ \mathcal{M}(y)\ {<}\mathrm{Y}{>}\ {<}S_2{>}.
\end{aligned}
$$

$$\sum\nolimits_{(x_{\text{in}}, y) \in \mathcal{D}_{\text{train}}} \log P_{\text{T5}}(\mathcal{T} \mid \mathcal{T}_{\text{g}}(x_{\text{in}}, y))$$

$$\sum_{j=1}^{|\mathcal{T}|} \sum_{(x_{\text{in}}, y) \in \mathcal{D}_{\text{train}}} \log P_{\text{T5}}\left(t_j \mid t_1, \ldots, t_{j-1}, \mathcal{T}_{\text{g}}\left(x_{\text{in}}, y\right)\right)$$

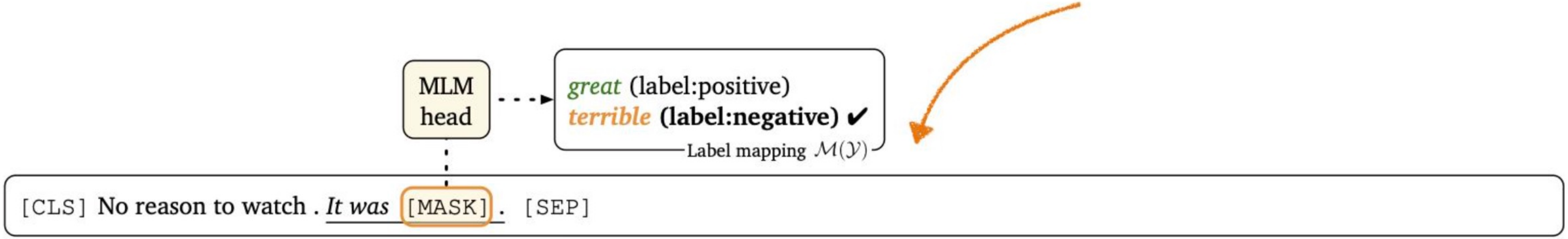
Automatic Generation of Templates (cont.)

- Use a wide (b = 100) beam search to decode <X> and <Y>.
- Finally, fine-tune the model on top-p templates and pick the one with best dev accuracy. c



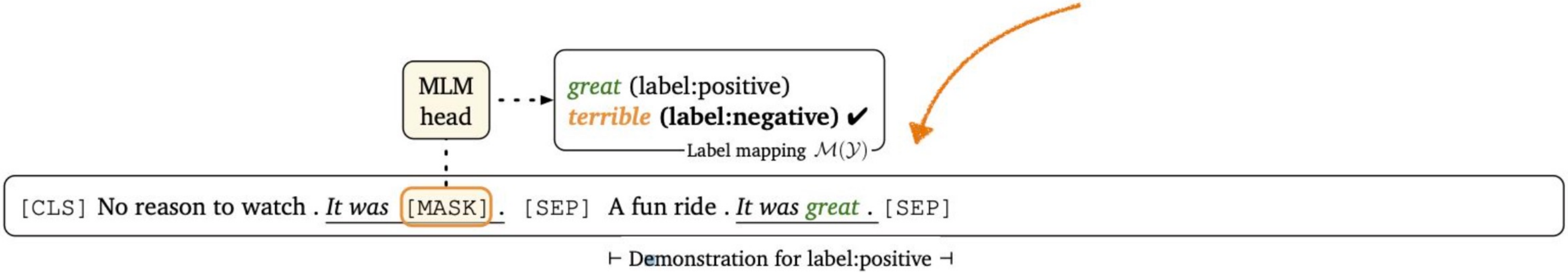
Demonstrations

GPT3 In-context Learning:
Randomly Samples Examples and fills them in context 👎



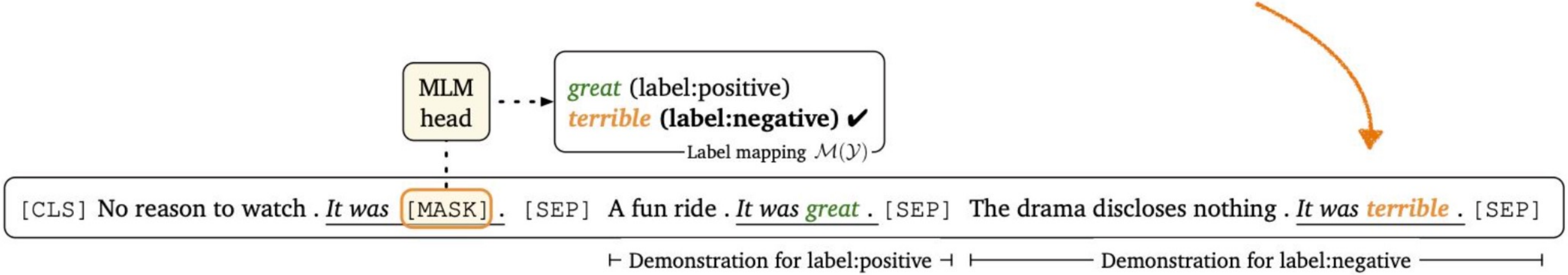
Demonstrations (cont.)

Improved: Selective Sampling, ie. for this example sample from then positive class 😎



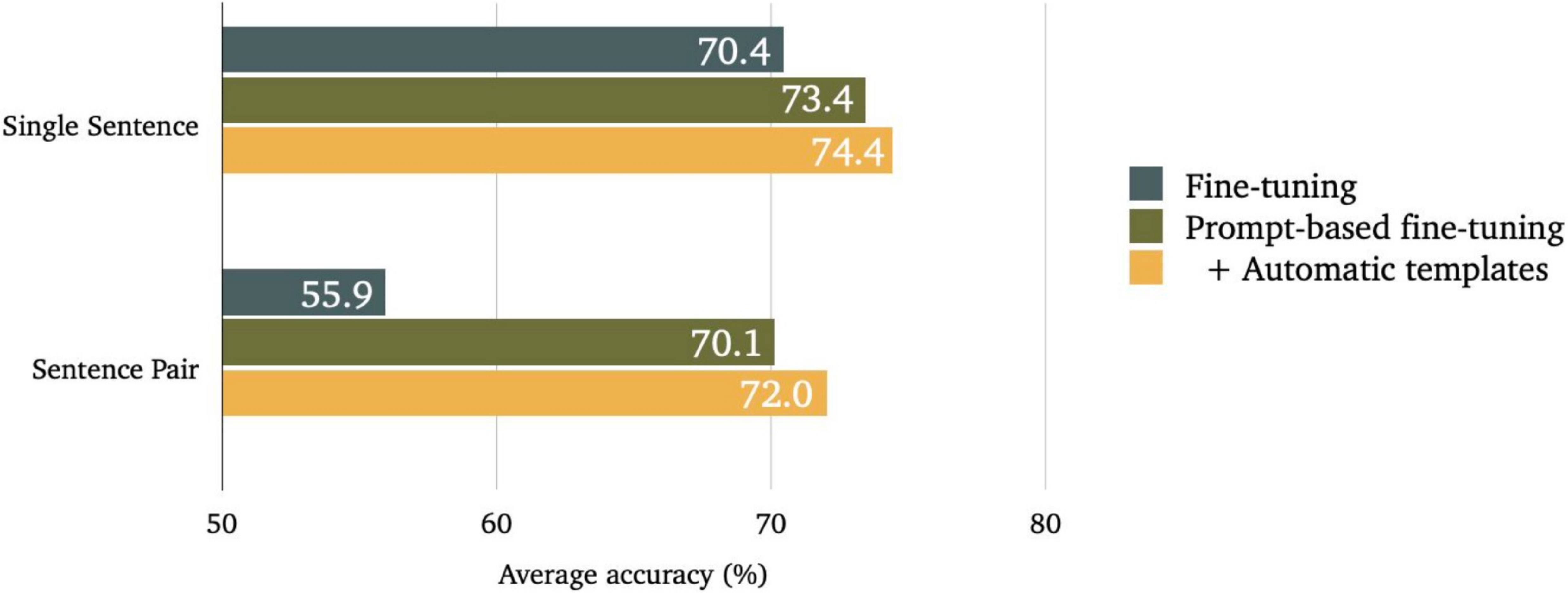
Demonstrations (cont.)

And we can also sample one from a negative training instance



- How to select demo samples?

Ablation Studies



Ablation Studies (cont.)