Making Pre-trained Language Models Better Few-shot Learners

Tianyu Gao*, Adam Fisch*, Danqi Chen

ACL 2021

Reporter: Xiachong Feng

Authors

Tianyu Gao

Adam Fisch

Danqi Chen

Why?

manually created prompts

Prompt-tuning
Exploiting Cloze Questions for Few Shot Text Classification and Natural Language Inference

Automatically created prompts

Mining-based and Paraphrasing-based
How Can We Know What Language Models Know?

Gradient Searching
AUTOPROMPT: Eliciting Knowledge from Language Models with Automatically Generated Prompts

Using Separate Model
Making pre-trained language models better few-shot learners.

GPT Understands, Too

Learning How to Ask: Querying Ms with Mixtures of Soft Prompts

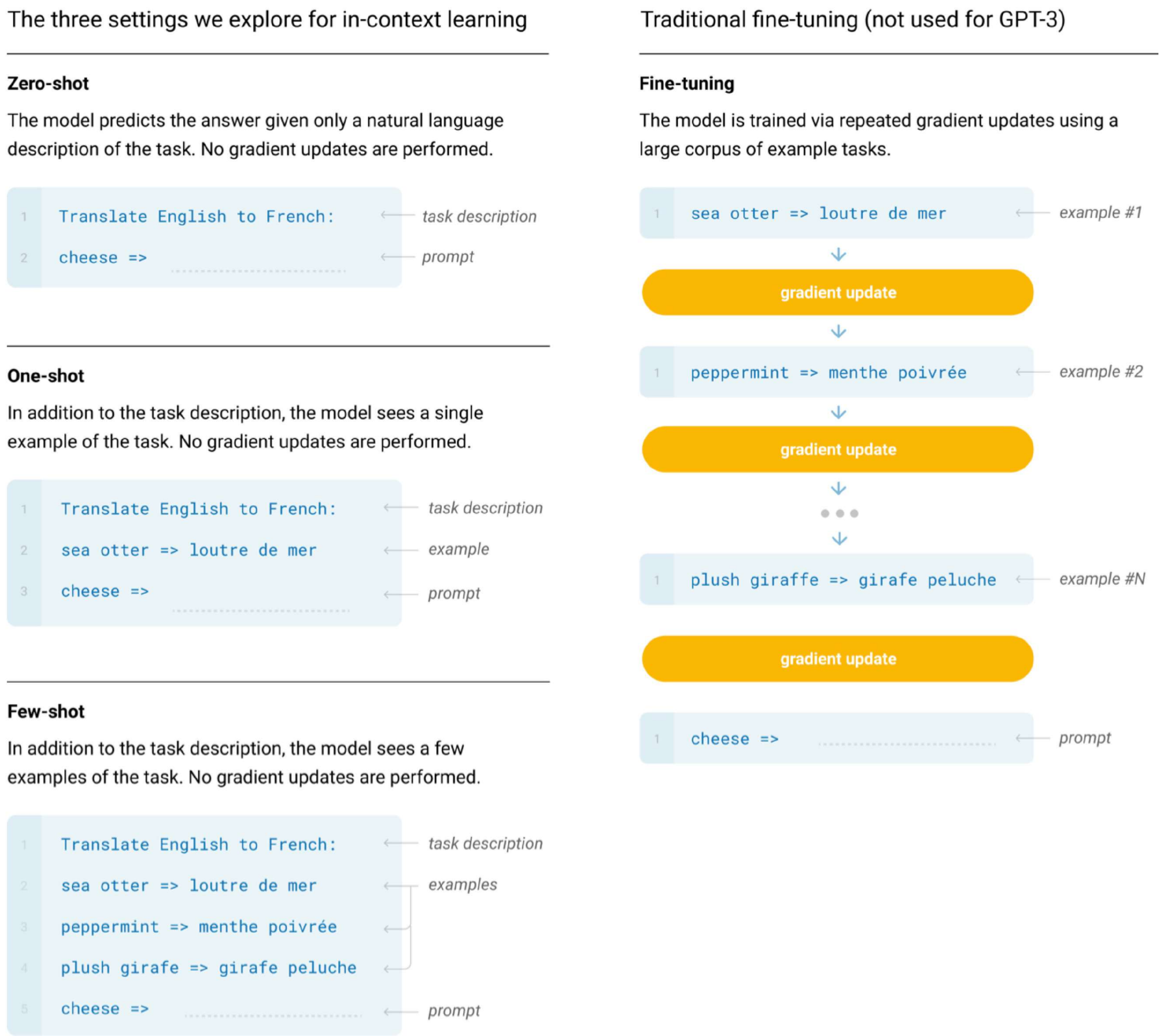
Factual Probing Is [MASK]: Learning vs. Learning to Recall

Differentially Optimized

In-context learning
Language Models are Few-Shot Learners (GPT-3)

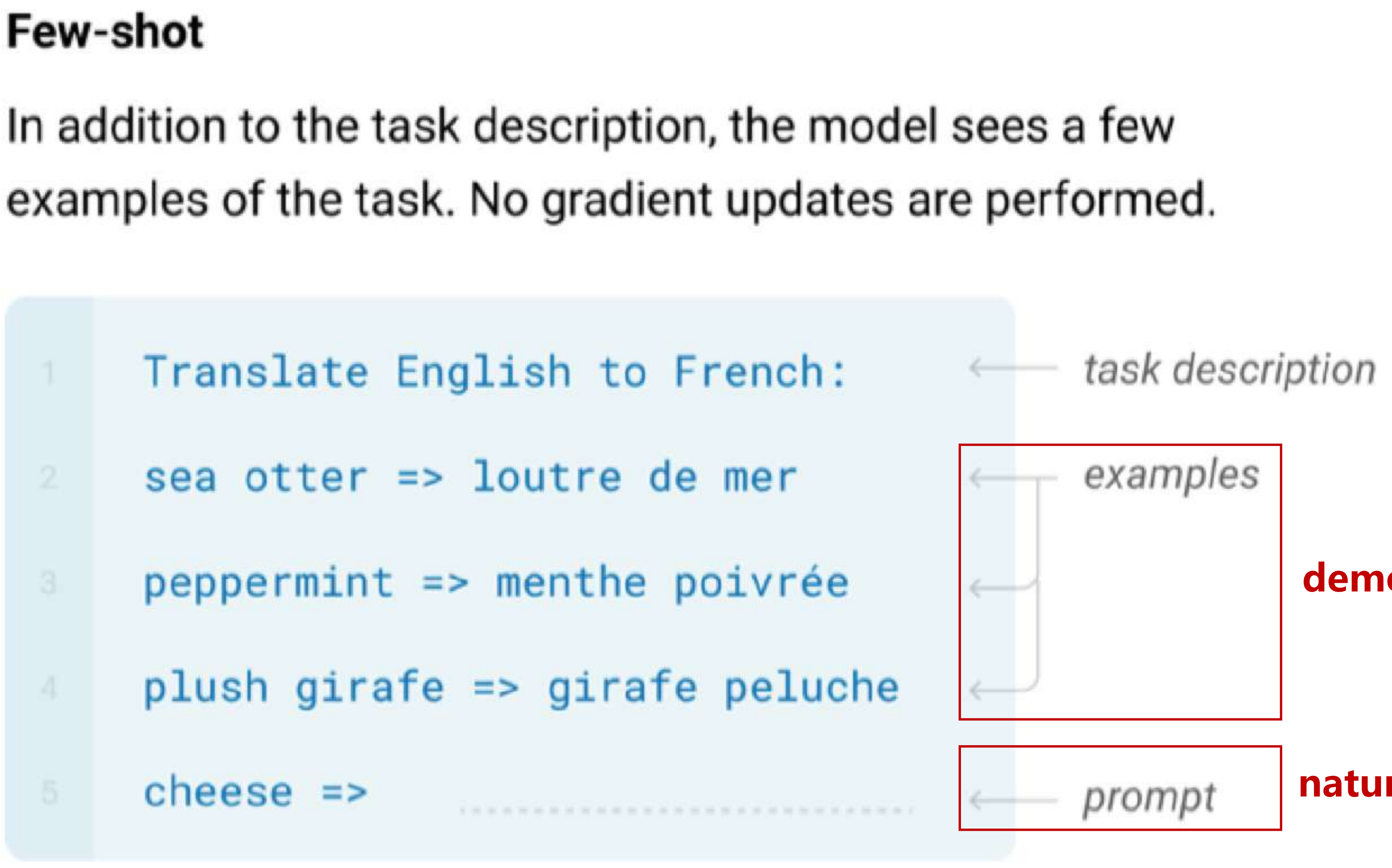
In-context learning

GPT-3



The three settings we explore for in-context learning
Zero-shot
The model predicts the answer given only a natural language description of the task. No gradient updates are performed.
1 Translate English to French:
task description
2 cheese =>
prompt
One-shot
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.
1 Translate English to French:
task description
2 sea otter => loutre de mer
example
3 cheese =>
prompt
Few-shot
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.
1 Translate English to French:
task description
2 sea otter => loutre de mer
examples
3 peppermint => menthe poivrée
4 plush girafe => girafe peluche
5 cheese =>
prompt
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.
1 sea otter => loutre de mer
example #1
gradient update
1 peppermint => menthe poivrée
example #2
gradient update
1 plush giraffe => girafe peluche
example #N
gradient update
1 cheese =>
prompt

GPT-3



demonstrations

natural language prompt

without updating any of the weights !

Overview

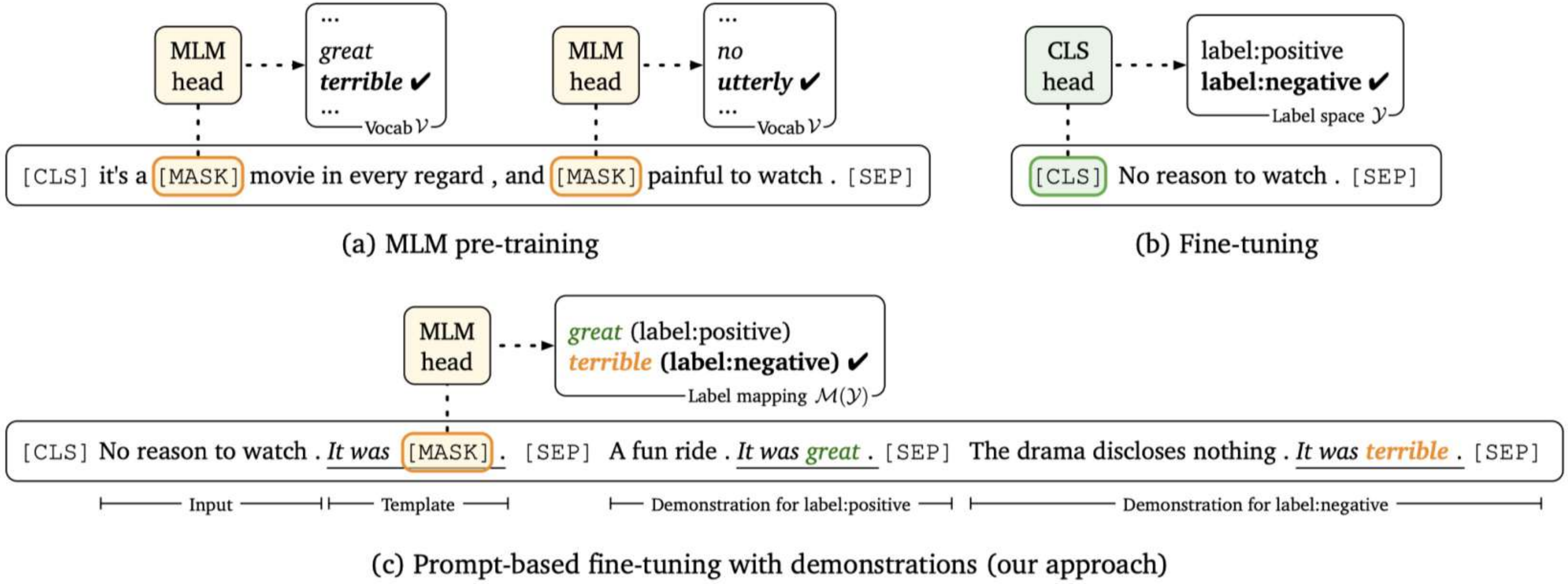
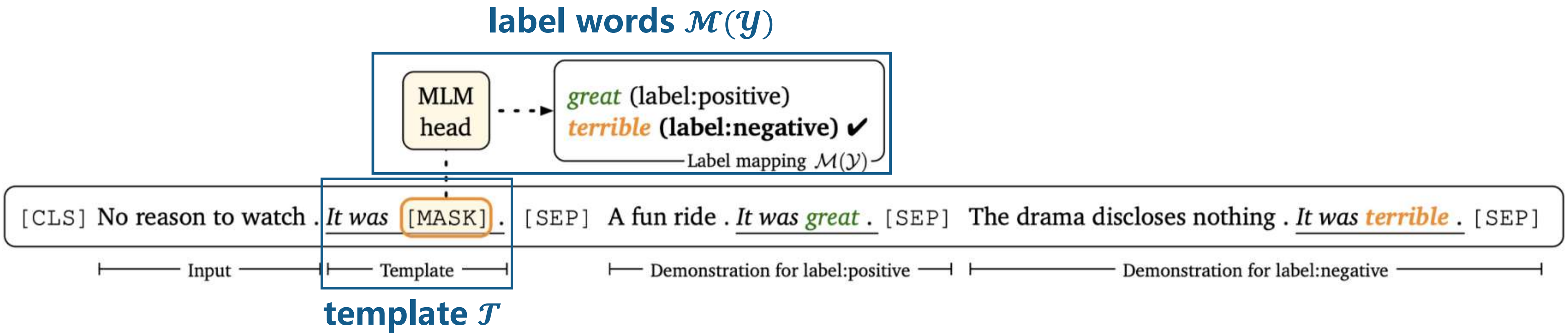


Figure 1: An illustration of (a) masked language model (MLM) pre-training, (b) standard fine-tuning, and (c) our proposed LM-BFF using prompt-based fine-tuning with demonstrations. The underlined text is the task-specific *template*, and colored words are *label words*.

Automatic Prompt Generation

What is a prompt?

- The key challenge is to construct the **template $\mathcal{T}$** and **label words $\mathcal{M}(\mathcal{Y})$** we refer to these two together as a **prompt $\mathcal{P}$.**
  1. Automatic generation of templates
  2. Automatic selection of label words



Manual Prompts

Task	Template	Label words
SST-2	$<S_1>$ It was `[MASK]` .	positive: great, negative: terrible
SST-5	$<S_1>$ It was `[MASK]` .	v.positive: great, positive: good, neutral: okay, negative: bad, v.negative: terrible
MR	$<S_1>$ It was `[MASK]` .	positive: great, negative: terrible
CR	$<S_1>$ It was `[MASK]` .	positive: great, negative: terrible
Subj	$<S_1>$ This is `[MASK]` .	subjective: subjective, objective: objective
TREC	`[MASK]` : $<S_1>$	abbreviation: Expression, entity: Entity, description: Description human: Human, location: Location, numeric: Number
COLA	$<S_1>$ This is `[MASK]` .	grammatical: correct, not_grammatical: incorrect
MNLI	$<S_1>$? `[MASK]` , $<S_2>$	entailment: Yes, netural: Maybe, contradiction: No
SNLI	$<S_1>$? `[MASK]` , $<S_2>$	entailment: Yes, netural: Maybe, contradiction: No
QNLI	$<S_1>$? `[MASK]` , $<S_2>$	entailment: Yes, not_entailment: No
RTE	$<S_1>$? `[MASK]` , $<S_2>$	entailment: Yes, not_entailment: No
MRPC	$<S_1>$ `[MASK]` , $<S_2>$	equivalent: Yes, not_equivalent: No
QQP	$<S_1>$ `[MASK]` , $<S_2>$	equivalent: Yes, not_equivalent: No
STS-B	$<S_1>$ `[MASK]` , $<S_2>$	y_u: Yes, y_l: No

Pilot Study

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)
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)

Table 2: The impact of templates and label words on prompt-based fine-tuning ($K = 16$).

when a template is fixed, the better the label words match the "semantic classes", the better the final accuracy is (great/terrible > good/bad > cat/dog).

In extreme cases where we swap plausible label words (e.g., terrible/great), we achieve the worst overall performance

with the same set of label words, even a small change in the template can make a difference.

Automatic Selection of Label Words

K

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

$\boxed{\mathcal{D}^c_{\text{train}} \subset \mathcal{D}_{\text{train}}}$

$\boxed{\text{output probability distribution of } \mathcal{L}}$

N

We find the **top n** assignments over the pruned space that maximize zero-shot accuracy on D_{train}

1

We fine-tune all top n assignments, and rerank to find the **best one** using D_{dev}.

Automatic Generation of Templates

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

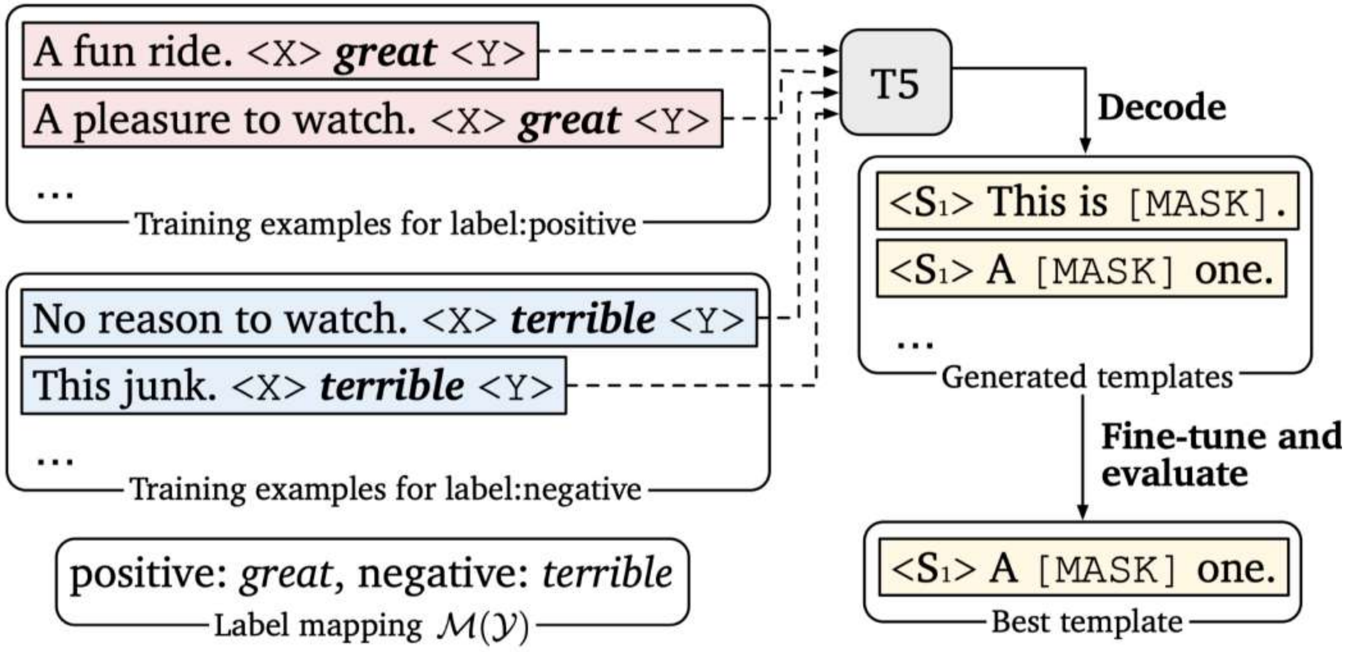


Figure 2: Our approach for template generation.

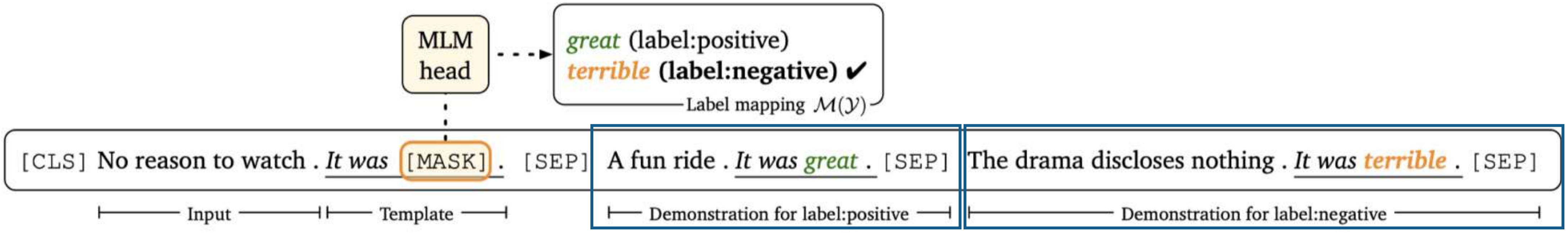
Fine-tuning with Demonstrations

Fine-tuning with Demonstrations

- GPT-3
  - concatenating the input with up to 32 examples randomly drawn from the training set.
- Drawbacks
  - the number of available demonstrations is bounded by the model’s maximum input length
  - mixing numerous random examples from different classes together creates extremely long contexts which can be hard to leverage, especially for a smaller model.

Fine-tuning with Demonstrations

- We randomly sample a single example at a time from each class to create multiple, minimal demonstration sets.



(c) Prompt-based fine-tuning with demonstrations (our approach)

Fine-tuning with Demonstrations

Sampling similar demonstrations

- We only sample examples that are semantically close to x_{in}.
- Pre-trained SBERT
- Sample from the top r = 50% instances for each class to use as demonstrations.

Experiments

Task Formulation

For Train

$$\mathcal{D}_{train} = \{(x_{in}^{i}, y^{i})\}_{i=1}^{K_{tot}}$$

$$K_{tot} = K \times |\mathcal{Y}|$$

$\mathcal{Y}$:label space, K training examples per class (=16)

For Dev: true few-shot

$$|\mathcal{D}_{dev}| = |\mathcal{D}_{train}|$$

Classification

$$\mathcal{M}: \mathcal{Y} \rightarrow \mathcal{V}$$

mapping from the task label space to individual words

$$x_{\text{prompt}} = \mathcal{T}(x_{\text{in}})$$

masked language modeling (MLM) input
which contains one [MASK] token.

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

Re-uses the pre-trained weights W_v and does not introduce any new parameters.

Regression

$[v_l, v_u]$ label space Y as a bounded interval

Sentiment Analysis: “terrible” (v_l = 0) and “great” (v_u = 1)

$[v_l, v_u]$ => {*terrible*(y_l), *great*(y_u)}

$$\boldsymbol{y} = v_l \cdot p(y_l|x_{in}) + v_u \cdot p(y_u|x_{in})$$
$$p(y_l|x_{in}) = 1 - p(y_u|x_{in})$$

$$\mathcal{M}: \{y_l, y_u\} \rightarrow \mathcal{V}$$

e.g. (*terrible*->bad)

Evaluation Datasets

Tasks

- 8 single-sentence and 7 sentence-pair English tasks
  - 8 tasks from the GLUE benchmark, SNLI
  - 6 other popular sentence classification tasks (SST-5, MR, CR, MPQA, Subj, TREC)

For robustness

- measure average performance across 5 different randomly sampled $|\mathcal{D}_{train}|$ and $|\mathcal{D}_{dev}|$ splits.

Single-prompt 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*

Single-prompt 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 (m	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	8 (5.1)	9.3 (7.3)
+ demonstrations	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	5 (3.2)	18.7 (8.8)
Prompt-based FT (a	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	2 (2.0)	14.0 (14.1)
+ demonstrations	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	**4** (1.7)	21.8 (15.9)
Fine-tuning (full)†	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	*97.4*	*62.6*

	(acc)	(acc)	(acc)	(acc)	(acc)	(F1)	QP (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*

① prompt-based zero-shot prediction achieves much better performance than the majority class, showing the pre-encoded knowledge in RoBERTa.

Single-prompt 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*

								STS-B (Pear.)
Majority†								-
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*

② prompt-based fine-tuning can greatly outperform standard fine-tuning

Single-prompt 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*

② using demonstrations in context leads to consistent gains in a majority of tasks.

								TS-B (ear.)
Majo								-
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*

Ensemble Results

Prompt-based Fine-tuning	**MNLI**	**RTE**
Our single manual $\mathcal{P}$	68.3 (2.3)	69.1 (3.6)
$\mathcal{P}_{\text{PET}}$	71.9 (1.5)	69.2 (4.0)
$\mathcal{P}_{\text{ours}}$, $\lvert\mathcal{P}_{\text{ours}}\rvert = \lvert\mathcal{P}_{\text{PET}}\rvert$	70.4 (3.1)	73.0 (3.2)
+ demonstrations	74.0 (1.9)	71.9 (4.6)
$\mathcal{P}_{\text{ours}}$, $\lvert\mathcal{P}_{\text{ours}}\rvert = 20$	72.7 (2.5)	**73.1** (3.3)
+ demonstrations	**75.4** (1.6)	72.3 (4.5)

Table 4: Ensemble models using manual prompts from PET (Schick and Schütze, 2021a,b) and our automatic templates. PET uses 4 prompts for MNLI and 5 for RTE. We also use an equal number of templates in $\lvert\mathcal{P}_{\text{ours}}\rvert = \lvert\mathcal{P}_{\text{PET}}\rvert$ for a fair comparison.

Analysis of Generated Prompts

	SST-2	SNLI	TREC	MRPC
Manual	**92.7**	**77.2**	84.8	74.5
Auto T	92.3	77.1	88.2	76.2
Auto L	91.5	75.6	87.0	**77.2**
Auto T + L	92.1	77.0	**89.2**	74.0

Table 5: Comparison between manual prompts and different automatic prompt generation methods: auto-generated templates (Auto T), auto-generated label words (Auto L), and their combination (Auto T + L).

SST-2	(positive/negative)
Auto T	$\mathcal{M}(\mathcal{Y})$ = {great, terrible} #1. $<S_1>$ A `[MASK]` one . #2. $<S_1>$ A `[MASK]` piece . #3. $<S_1>$ All in all `[MASK]` .
Auto L	$\mathcal{T}(x_{\text{in}})$ = $<S_1>$ It was `[MASK]`. #1. irresistible/pathetic #2. wonderful/bad #3. delicious/bad
SNLI	(entailment/neutral/contradiction)
Auto T	$\mathcal{M}(\mathcal{Y})$ = {Yes, Maybe, No} #1. $<S_1>$. `[MASK]` , no , $<S_2>$ #2. $<S_1>$. `[MASK]` , in this case $<S_2>$ #3. $<S_1>$. `[MASK]` this time $<S_2>$
Auto L	$\mathcal{T}(x_{\text{in}})$ = $<S_1>$? `[MASK]` , $<S_2>$ #1. Alright/Watch/Except #2. Hi/Watch/Worse #3. Regardless/Fortunately/Unless

Table 6: Examples of our automatically generated templates (Auto T) and label words (Auto L).

Analysis of Demonstration Sampling

	SST-2	SNLI	TREC	MRPC
Prompt-based FT	**92.7**	77.2	84.8	74.5
Uniform sampling	92.3	78.8	85.6	70.9
+ RoBERTa sel.	**92.7**	79.5	83.4	76.6
+ SBERT sel.	92.6	**79.7**	**87.5**	**77.8**

Table 7: Impact of demonstration sampling strategies. Uniform sampling randomly samples demonstrations, while selective (sel.) sampling only takes top sentences measured by the sentence encoders (§6).

Sample Efficiency

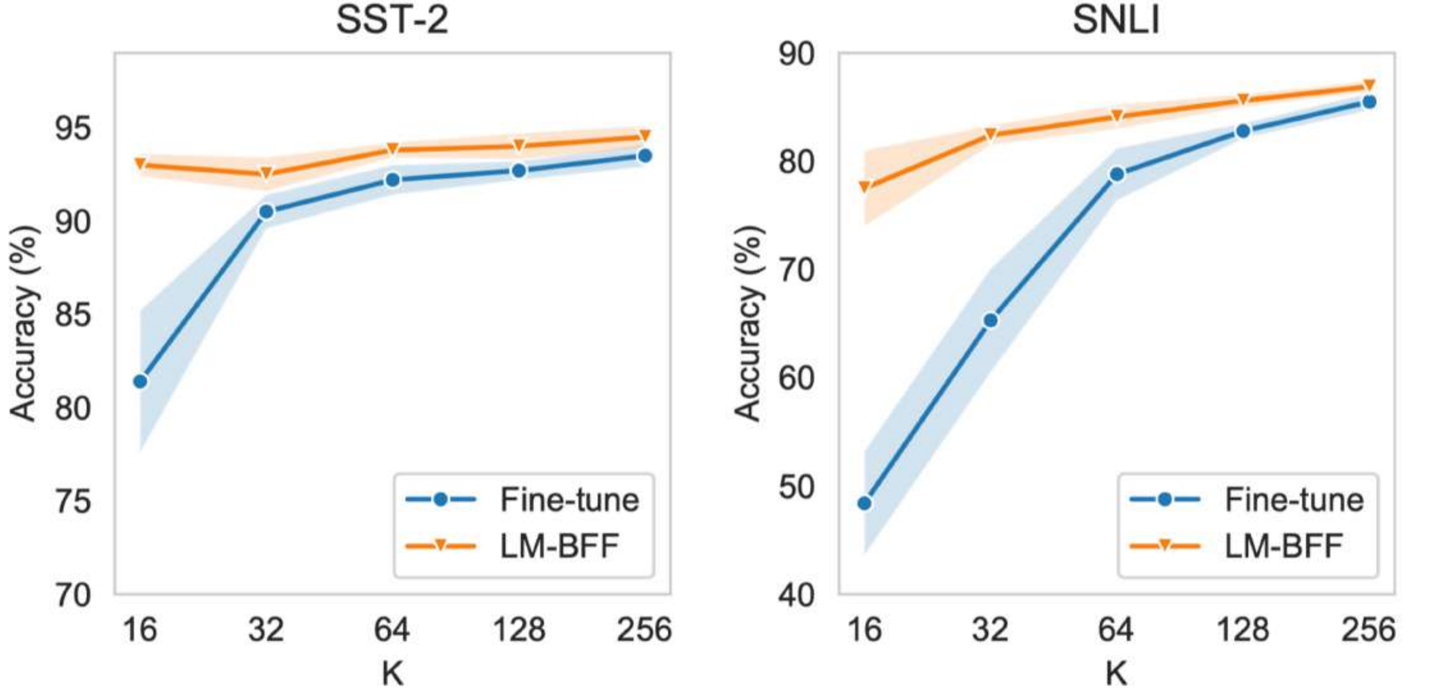


Figure 3: Standard fine-tuning vs our LM-BFF as a function of K (# instances per class). For lower K, our method consistently outperforms standard fine-tuning.

Discussion and Conclusion

Discussion

LM-BFF

1. Can be naturally posed as a “fill-in-the-blank” problem.
2. Have relatively short input sequences.
3. Do not contain many output classes. *(for structured prediction?)*

Conclusion

- Use prompt-based fine-tuning with automatically searched prompts;
- Include selected task demonstrations;
- LM-BFF outperforms vanilla fine-tuning by up to 30% (and 11% on average).

Thanks~