Prompting
for few-shot learning

AntNLP 2022 Fall Seminar
Ding Xuanwen

主要内容

- Background
- **PET**
- How Many Data Points is a Prompt Worth?
- **LM-BFF**
- Null-prompt
- True Few-Shot Learning with Language Models

Few-shot for GPT3

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
```

Almost state-of-the art on (some) SuperGLUE tasks with only 32 labeled examples!

Few-shot for GPT3

GPT

Natural language prompts
gigantic model
few in-context examples
no-parameters updated

BERT

110M Parameters
1000x smaller than GPT3
Generic [CLS] Token
Fine-tuned to 2.5 to 400k examples for GLUE tasks

Few-shot for GPT3

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
```

- Needs huge models
- Limited number of labeled example

PET：pattern-exploiting traning

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

Timo Schick[1,2] **Hinrich Schütze**[1]

[1] Center for Information and Language Processing, LMU Munich, Germany
[2] Sulzer GmbH, Munich, Germany

schickt@cis.lmu.de

Abstract

Some NLP tasks can be solved in a fully unsupervised fashion by providing a pretrained language model with "task descriptions" in natural language (e.g., Radford et al., 2019). While this approach underperforms its supervised counterpart, we show in this work that the two ideas can be combined: We introduce Pattern-Exploiting Training (PET), a semi-supervised training procedure that reformulates input examples as cloze-style phrases to help language models understand a given task. These phrases are then used to assign soft labels to a large set of unlabeled examples. Finally, standard supervised training is performed on the resulting training set. For several tasks and languages, PET outperforms supervised training and strong semi-supervised approaches in low-resource settings by a large margin.[1]

1 Introduction

Learning from examples is the predominant approach for many NLP tasks: A model is trained

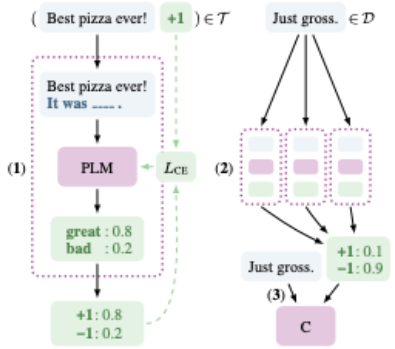


Figure 1: PET for sentiment classification. **(1)** A number of patterns encoding some form of task description are created to convert training examples to cloze questions; for each pattern, a pretrained language model is finetuned. **(2)** The ensemble of trained models annotates unlabeled data. **(3)** A classifier is trained on the resulting soft-labeled dataset.

- Needs normal-sized models
  - Needs huge models
- Unlimited number of labeled examples
  - Limited number of labeled example

Pattern & verbalizer

<Notation>

- M : Masked Language Model
- V : Vocabulary
- $\_ \in V$: Mask Token
- $\mathcal{L}, A$: Labels,Target classification task
- $x = (s_1, \ldots, s_k)$: Input
- P : Pattern
- $v : \mathcal{L} \rightarrow V$
- (P, v) : Pattern-verbalizer pair(PVP)

<Example>

$P(a, b) = a? \_, b.$

$v: y_0 \rightarrow Yes, y_1 \rightarrow No$

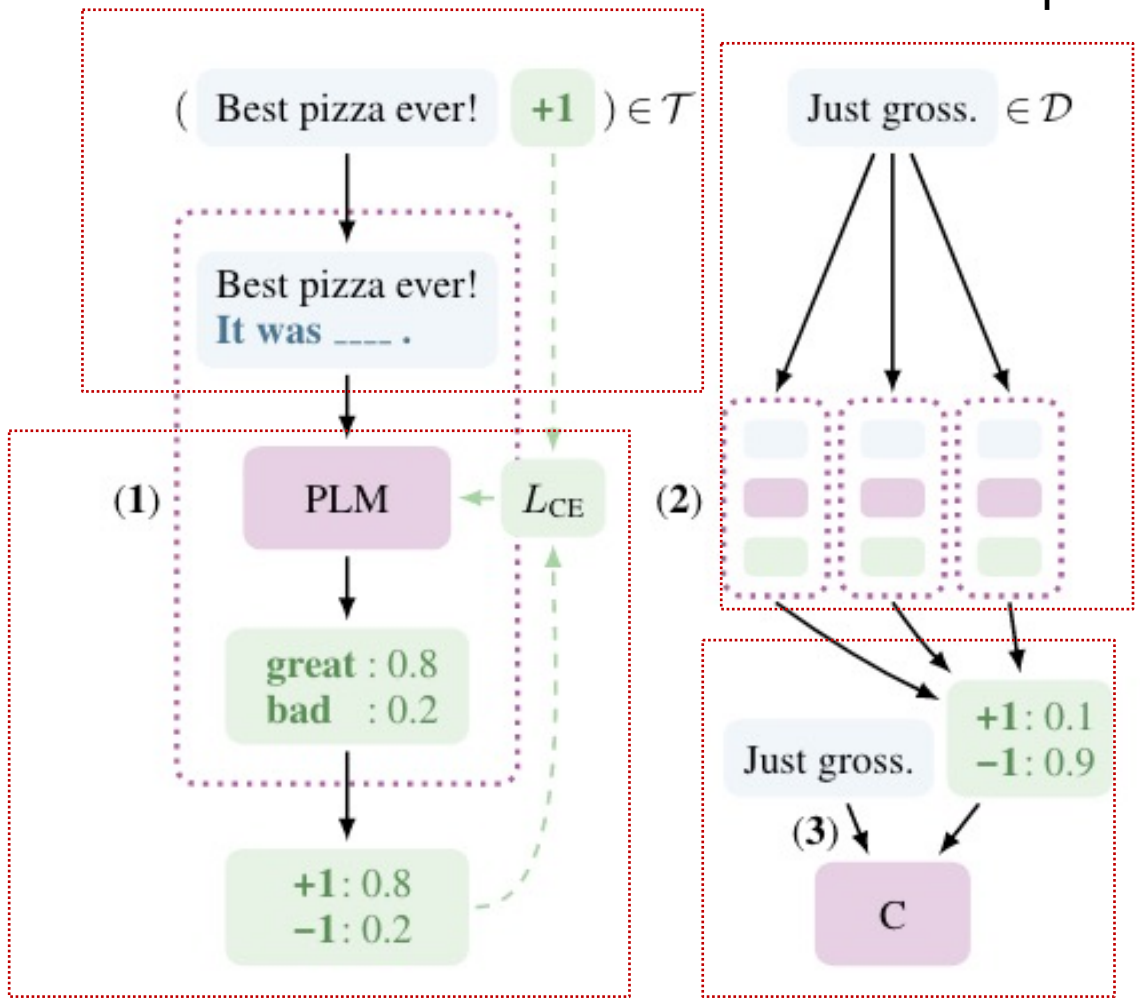
x=(Mia likes pies,Mia hates pie)

$P(x)$=Mia likes pie?__,Mia hates pie.

Overview

step1

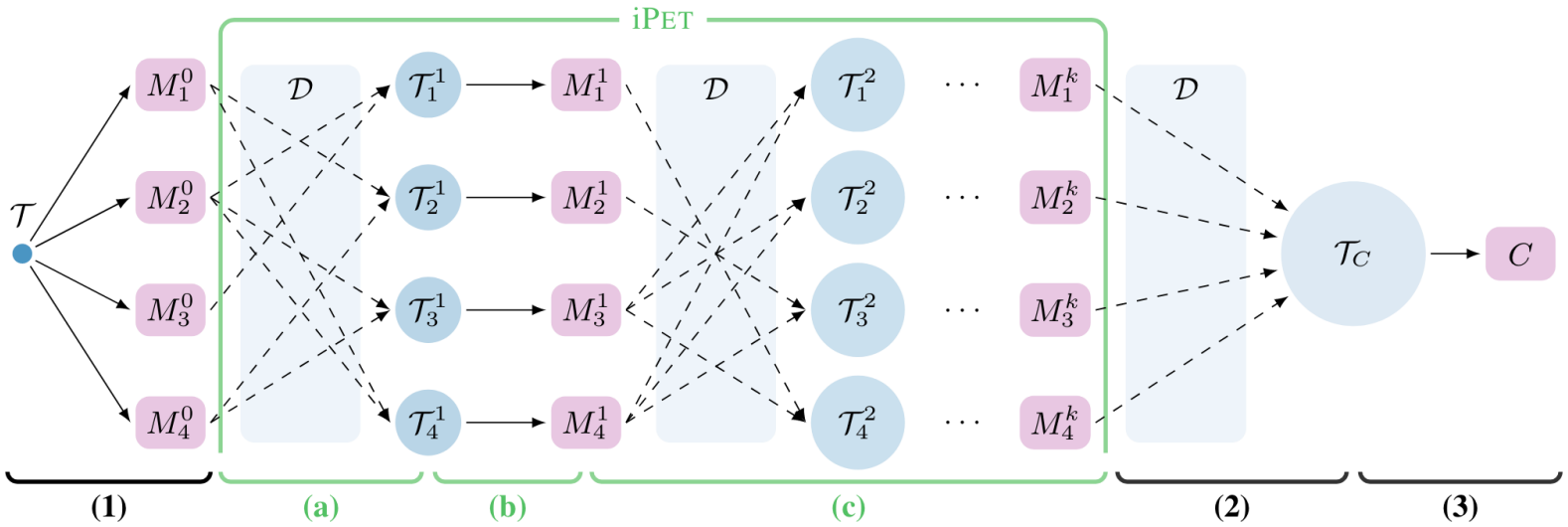
step3



step4

$$L = (1 - \alpha) \cdot L_{\text{CE}} + \alpha \cdot L_{\text{MLM}}$$

PET & iPET



(1)设定初始模型集

$$M^0 = \{M_1^0, \ldots, M_n^0\}$$

(a)创建一个soft-labeled dataset

score：

$$s_{\mathbf{p}}(l \mid \mathbf{x}) = M(v(l) \mid P(\mathbf{x}))$$

$$q_{\mathbf{p}}(l \mid \mathbf{x}) = \frac{e^{s_{\mathbf{p}}(l|\mathbf{x})}}{\sum_{l' \in \mathcal{L}} e^{s_{\mathbf{p}}(l'|\mathbf{x})}} \qquad s_{\mathcal{M}}(l \mid \mathbf{x}) = \frac{1}{Z} \sum_{\mathbf{p} \in \mathcal{P}} w(\mathbf{p}) \cdot s_{\mathbf{p}}(l \mid \mathbf{x})$$

模型选择：$\mathcal{N} \subset \mathcal{M}^{j-1} \setminus \{M_i^{j-1}\}$

产生dataset: $\mathcal{T}_{\mathcal{N}} = \{(\mathbf{x}, \arg\max_{l \in \mathcal{L}} s_{\mathcal{N}}(l \mid \mathbf{x})) \mid \mathbf{x} \in \mathcal{D}\}$

新dataset大小：$c_j(l) = \hat{d} \cdot c_{j-1}(l)$

Experiments

Dataset： Yelp Reviews , AG’s News , Yahoo Questions , MNLI , x-stance

Yelp Reviews

$P_1(a) =$ It was ____. a $\quad P_2(a) =$ Just ____! $\|$ a

$P_3(a) =$ a. All in all, it was ____.

$P_4(a) =$ $a \parallel$ In summary, the restaurant is ____.

$v(1) =$ terrible $\quad v(2) =$ bad $\quad v(3) =$ okay

$v(4) =$ good $\quad v(5) =$ great

AG’s News

$P_1(\mathbf{x}) =$ ____: a b $\quad P_2(\mathbf{x}) =$ a (____) b

$P_3(\mathbf{x}) =$ ____ – a b $\quad P_4(\mathbf{x}) =$ a b (____)

$P_5(\mathbf{x}) =$ ____ News: a b

$P_6(\mathbf{x}) =$ [Category: ____] a b

We use a verbalizer that maps 1–4 to “World”, “Sports”, “Business” and “Tech”, respectively.

Experiments

Line	Examples	Method	Yelp	AG's	Yahoo	MNLI (m/mm)
1	$\|\mathcal{T}\| = 0$	unsupervised (avg)	33.8 ±9.6	69.5 ±7.2	44.0 ±9.1	39.1 ±4.3 / 39.8 ±5.1
2		unsupervised (max)	40.8 ±0.0	79.4 ±0.0	56.4 ±0.0	43.8 ±0.0 / 45.0 ±0.0
3		iPET	**56.7** ±0.2	**87.5** ±0.1	**70.7** ±0.1	**53.6** ±0.1 / **54.2** ±0.1
4	$\|\mathcal{T}\| = 10$	supervised	21.1 ±1.6	25.0 ±0.1	10.1 ±0.1	34.2 ±2.1 / 34.1 ±2.0
5		PET	52.9 ±0.1	87.5 ±0.0	63.8 ±0.2	41.8 ±0.1 / 41.5 ±0.2
6		iPET	**57.6** ±0.0	**89.3** ±0.1	**70.7** ±0.1	**43.2** ±0.0 / **45.7** ±0.1
7	$\|\mathcal{T}\| = 50$	supervised	44.8 ±2.7	82.1 ±2.5	52.5 ±3.1	45.6 ±1.8 / 47.6 ±2.4
8		PET	60.0 ±0.1	86.3 ±0.0	66.2 ±0.1	63.9 ±0.0 / 64.2 ±0.0
9		iPET	**60.7** ±0.1	**88.4** ±0.1	**69.7** ±0.0	**67.4** ±0.3 / **68.3** ±0.3
10	$\|\mathcal{T}\| = 100$	supervised	53.0 ±3.1	86.0 ±0.7	62.9 ±0.9	47.9 ±2.8 / 51.2 ±2.6
11		PET	61.9 ±0.0	88.3 ±0.1	69.2 ±0.0	74.7 ±0.3 / 75.9 ±0.4
12		iPET	**62.9** ±0.0	**89.6** ±0.1	**71.2** ±0.1	**78.4** ±0.7 / **78.6** ±0.5
13	$\|\mathcal{T}\| = 1000$	supervised	63.0 ±0.5	**86.9** ±0.4	70.5 ±0.3	73.1 ±0.2 / 74.8 ±0.3
14		PET	**64.8** ±0.1	**86.9** ±0.2	**72.7** ±0.0	**85.3** ±0.2 / **85.5** ±0.4

Table 1: Average accuracy and standard deviation for RoBERTa (large) on Yelp, AG's News, Yahoo and MNLI (m:matched/mm:mismatched) for five training set sizes $|\mathcal{T}|$.

Ex.	Method	Yelp	AG's	Yahoo	MNLI
$\|\mathcal{T}\| = 10$	UDA	27.3	72.6	36.7	34.7
	MixText	20.4	81.1	20.6	32.9
	PET	48.8	84.1	59.0	39.5
	iPET	**52.9**	**87.5**	**67.0**	**42.1**
$\|\mathcal{T}\| = 50$	UDA	46.6	83.0	60.2	40.8
	MixText	31.3	84.8	61.5	34.8
	PET	55.3	86.4	63.3	55.1
	iPET	**56.7**	**87.3**	**66.4**	**56.3**

Table 2: Comparison of PET with two state-of-the-art semi-supervised methods using RoBERTa (base)

Experiments

Examples	Method	De	Fr	It
$\lvert\mathcal{T}\rvert = 1000$	supervised	43.3	49.5	41.0
	PET	**66.4**	**68.7**	**64.7**
$\lvert\mathcal{T}\rvert = 2000$	supervised	57.4	62.1	52.8
	PET	**69.5**	**71.7**	**67.3**
$\lvert\mathcal{T}\rvert = 4000$	supervised	63.2	66.7	58.7
	PET	**71.7**	**74.0**	**69.5**
$\mathcal{T}_{\text{De}}$, $\mathcal{T}_{\text{Fr}}$	supervised	76.6	76.0	71.0
	PET	**77.9**	**79.0**	**73.6**
$\mathcal{T}_{\text{De}} + \mathcal{T}_{\text{Fr}}$	sup. (*)	76.8	76.7	70.2
	supervised	77.6	79.1	75.9
	PET	**78.8**	**80.6**	**77.2**

Table 3: Results on x-stance intra-target for XLM-R (base) trained on subsets of $\mathcal{T}_{\text{De}}$ and $\mathcal{T}_{\text{Fr}}$ and for joint training on all data ($\mathcal{T}_{\text{De}} + \mathcal{T}_{\text{Fr}}$). (*): Best results for mBERT reported in Vamvas and Sennrich (2020).

Experiments

Auxiliary Language Modeling

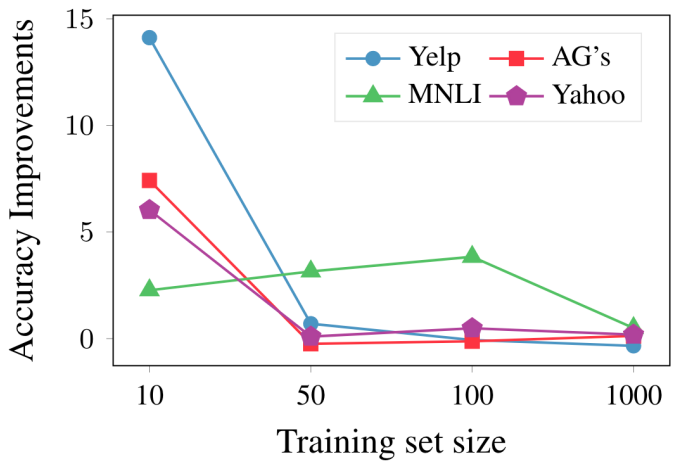


Figure 3: Accuracy improvements for PET due to adding L_{MLM} during training

Iterative PET

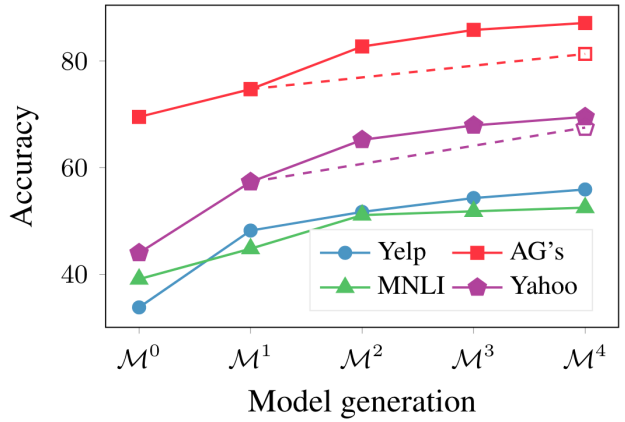


Figure 4: Average accuracy for each generation of models with iPET in a zero-shot setting. Accuracy on AG's News and Yahoo when skipping generation 2 and 3 is indicated through dashed lines.

In-Domain Pretraining

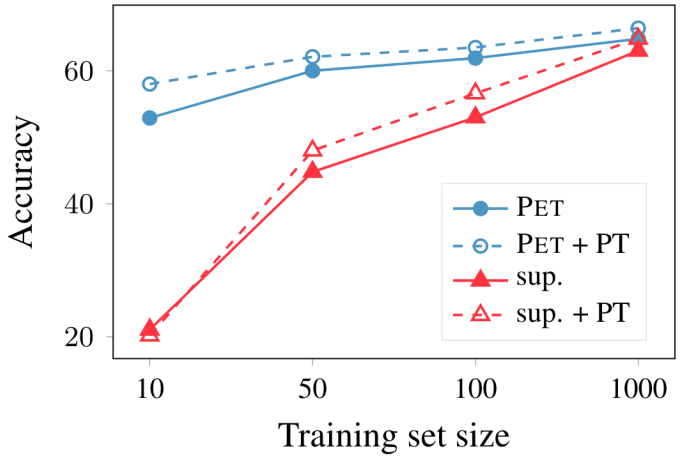


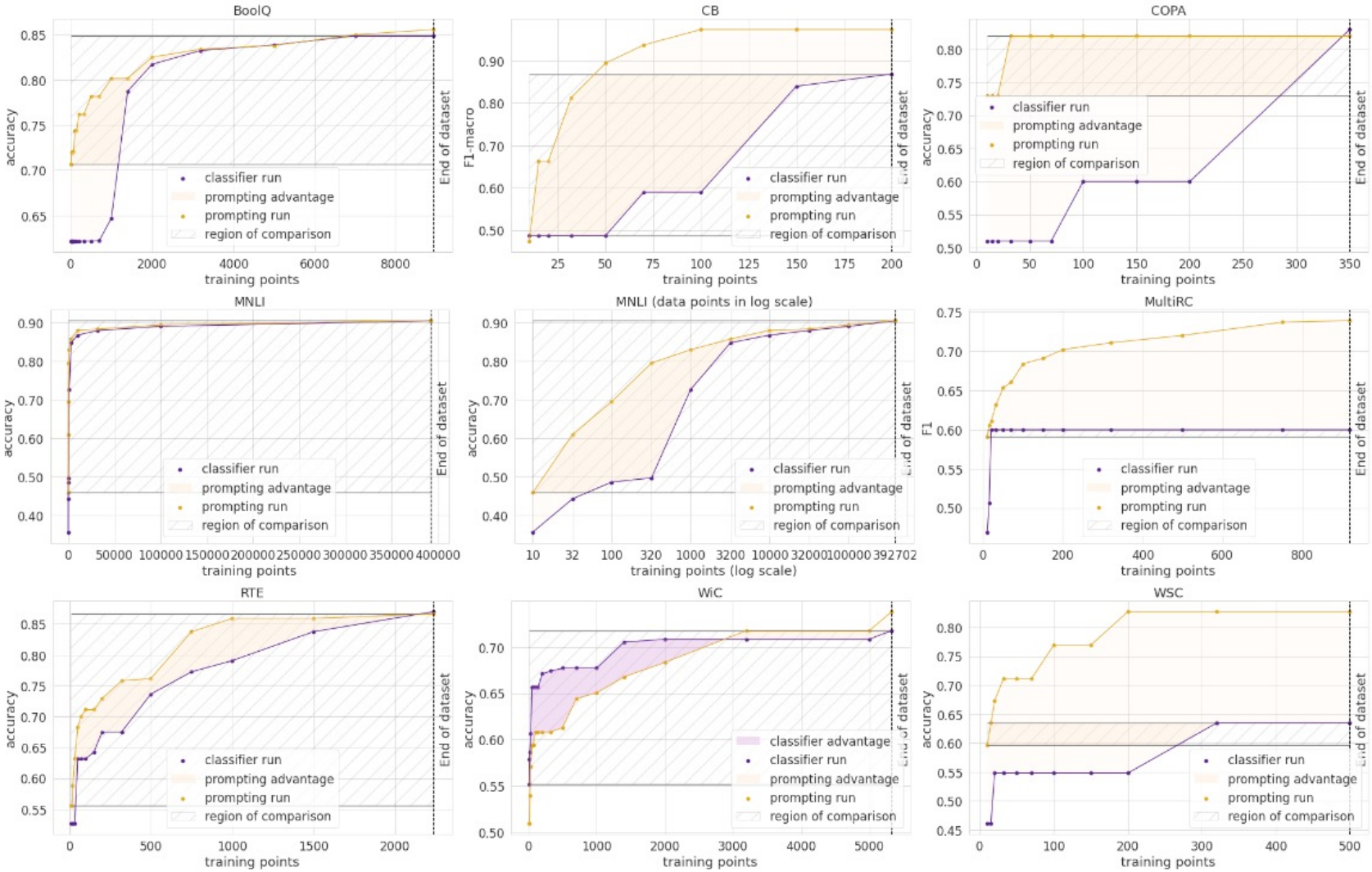
Figure 5: Accuracy of supervised learning (sup.) and PET both with and without pretraining (PT) on Yelp

How Many Data Points is a Prompt Worth?

Teven Le Scao
Hugging Face
teven@huggingface.co

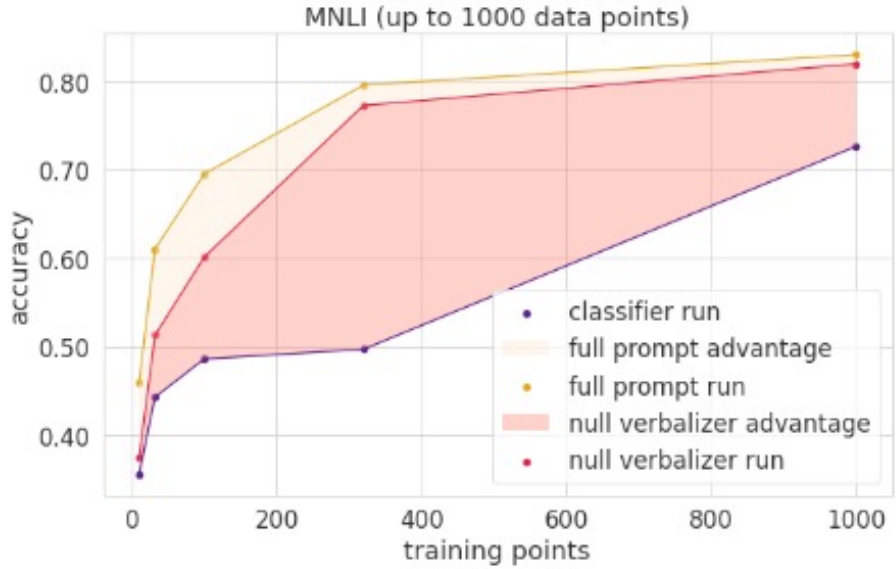
Alexander M. Rush
Hugging Face
sasha@huggingface.co

Heads vs Prompts

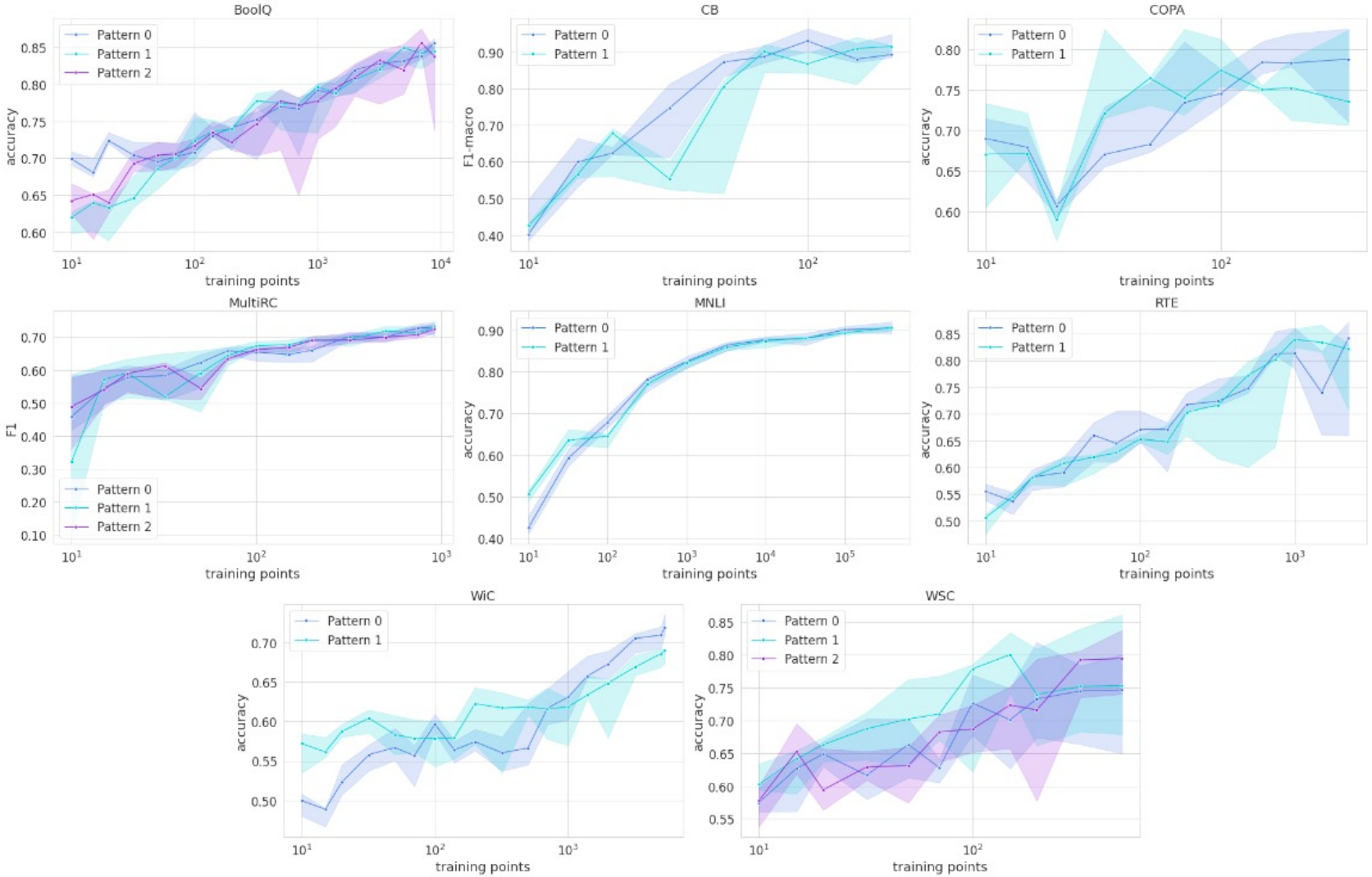


Pattern vs Verbalizer

	Average Advantage (# Training Points)							
	MNLI	BoolQ	CB	COPA	MultiRC*	RTE	WiC	WSC
P vs H	3506 ± 536	752 ± 46	90 ± 2	288 ± 242	384 ± 378	282 ± 34	-424 ± 74	281 ± 137
P vs N	150 ± 252	299 ± 81	78 ± 2	-	74 ± 56	404 ± 68	-354 ± 166	-
N vs H	3355 ± 612	453 ± 90	12 ± 1	-	309 ± 320	-122 ± 62	-70 ± 160	-



Pattern vs Verbalizer



LM-BFF: better few-shot fine-tuning of language models

Making Pre-trained Language Models Better Few-shot Learners

Tianyu Gao[†*] **Adam Fisch**[‡*] **Danqi Chen**[†]
[†]Princeton University [‡]Massachusetts Institute of Technology
{tianyug,danqic}@cs.princeton.edu
fisch@csail.mit.edu

(Small) Language Models are Few-shot Learners

GPT-3

Gigantic language model(up to 175B paramters)

Freezing model parameters

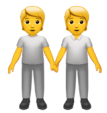
Using manually-designed prompts and demonstrations

PET

Small language model

Fine-tuning model parameters

Using manually-designed prompts



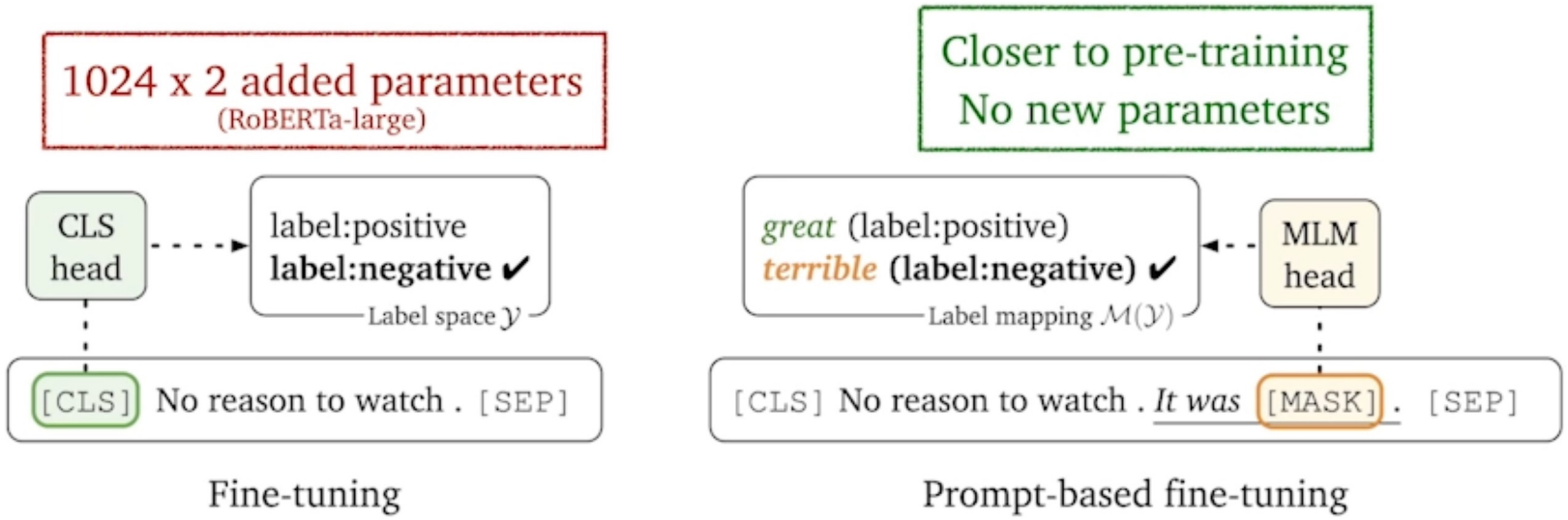
LM-BFF

Small language model

Fine-tuning model parameters

Using automatically-searched prompts and demonstration

Prompt-based Fine-tuning



Manual Label Words : the Good and the Bad

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)

Manual Templates: the Good and the Bad

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)

Automatic Prompt Generation

Recent Work

1. domain expertise
2. an understanding of the language model’ s inner workings
3. manual prompts are likely to be suboptimal

Our goals

1. reduce the human involvement required to design prompts
2. find more optimal settings than those that we manually choose

Automatic selection of label words

1.For each class c, select the top k vocabulary words based on their conditional likelihood

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

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

label : positive
good
great
perfect
…

label : negative
awful
bad
terrible
…

2. find the top n assignments over the pruned space that maximize zero-shot accuracy on $\mathcal{D}_{train}$

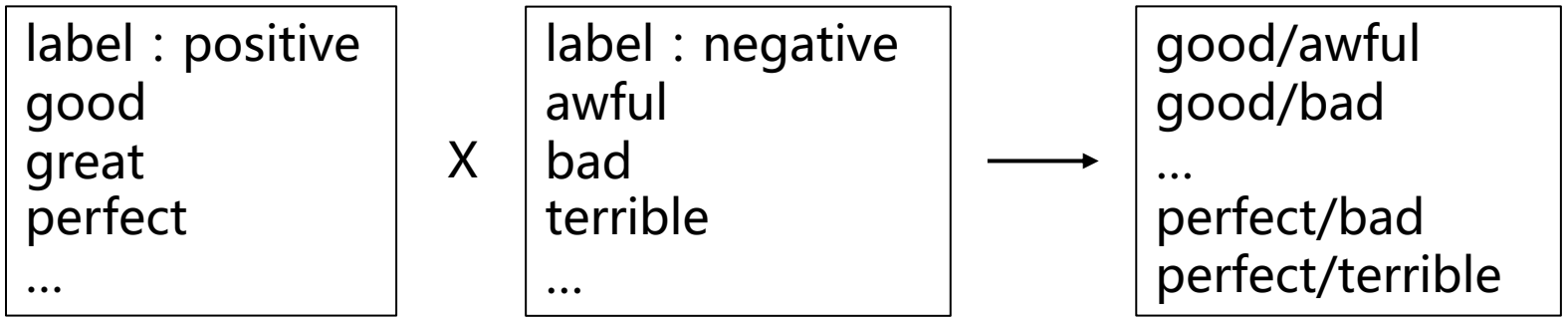
3. fine-tune all top n assignments, and re-rank to find the best one using $\mathcal{D}_{dev}$

Automatic selection of label words

1.For each class c, select the top k vocabulary words based on their conditional likelihood

2. find the top n assignments over the pruned space that maximize zero-shot accuracy on $\mathcal{D}_{train}$

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



3. fine-tune all top n assignments, and re-rank to find the best one using $\mathcal{D}_{dev}$

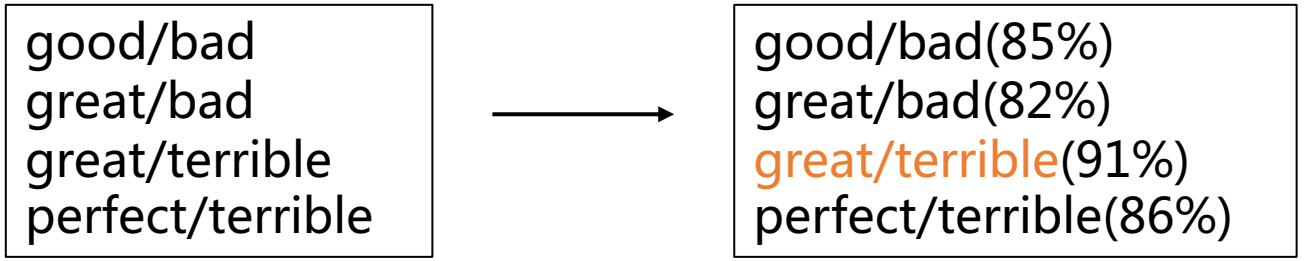
Automatic selection of label words

1.For each class c, select the top k vocabulary words based on their conditional likelihood

2. find the top n assignments over the pruned space that maximize zero-shot accuracy on $\mathcal{D}_{train}$

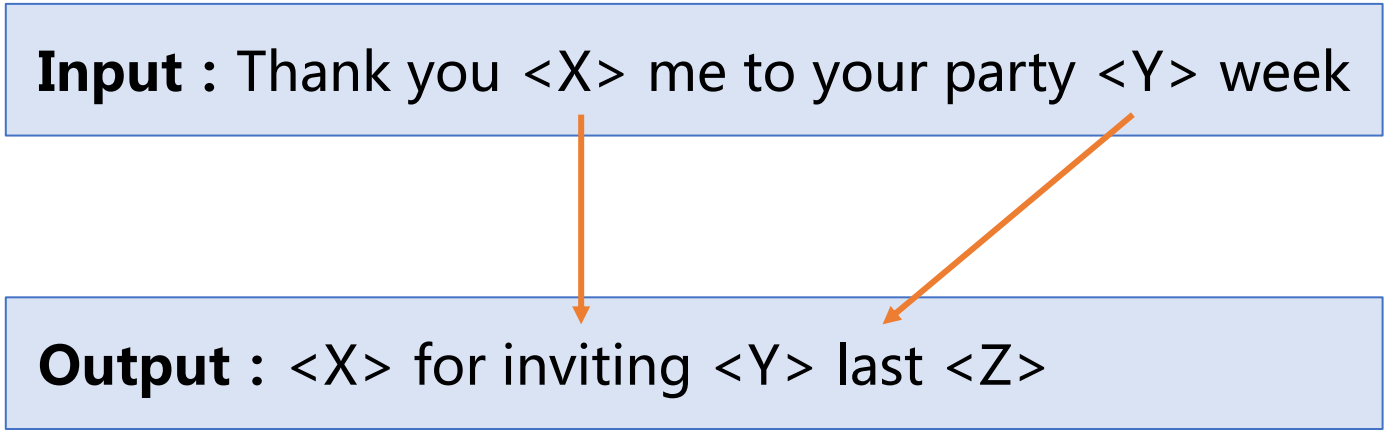
3. fine-tune all top n assignments, and re-rank to find the best one using $\mathcal{D}_{dev}$

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



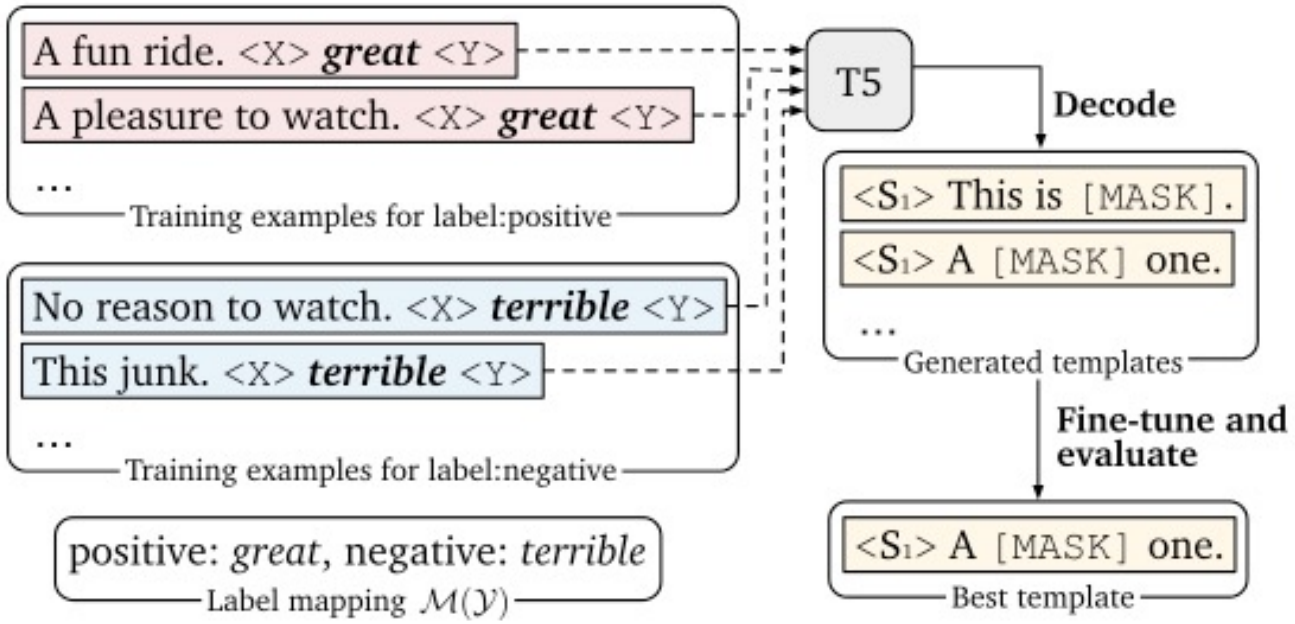
Automatic generation of templates

T5



T5 takes a fill-in-the blank pretraining objective.

Automatic generation of templates



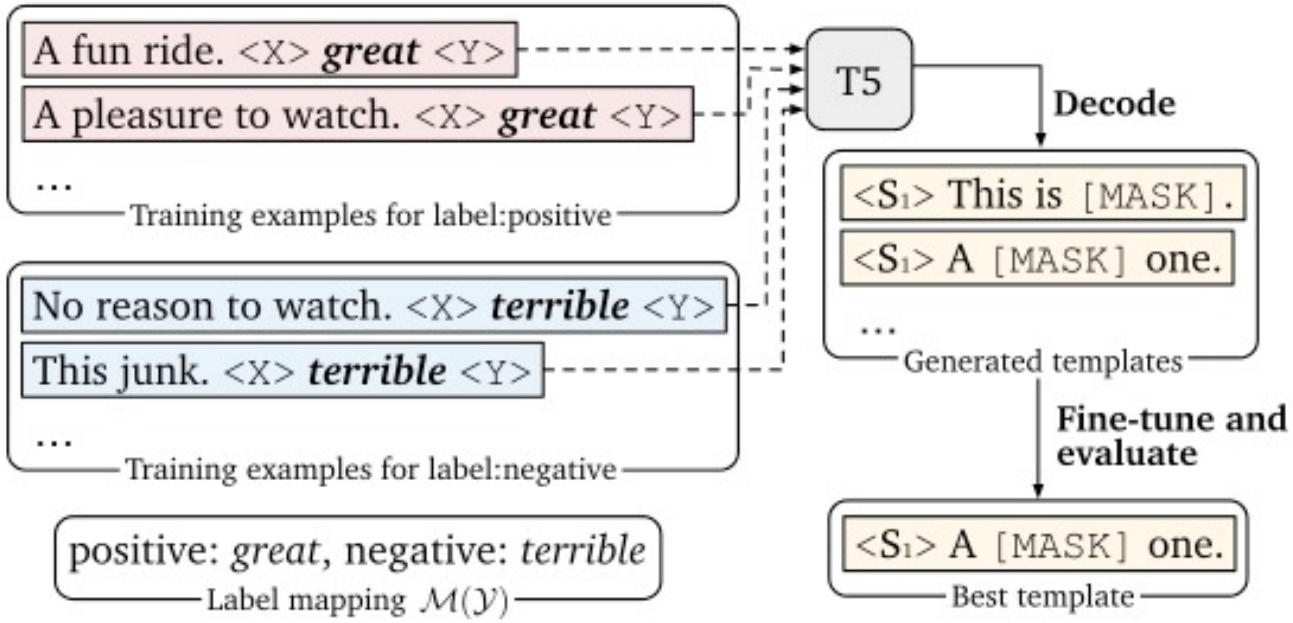
$\mathcal{T}_{\mathrm{g}}(x_{\mathrm{in}}, y)$:

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

condition the model on all training examples at each position in auto-regressive decoding

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

Automatic generation of templates

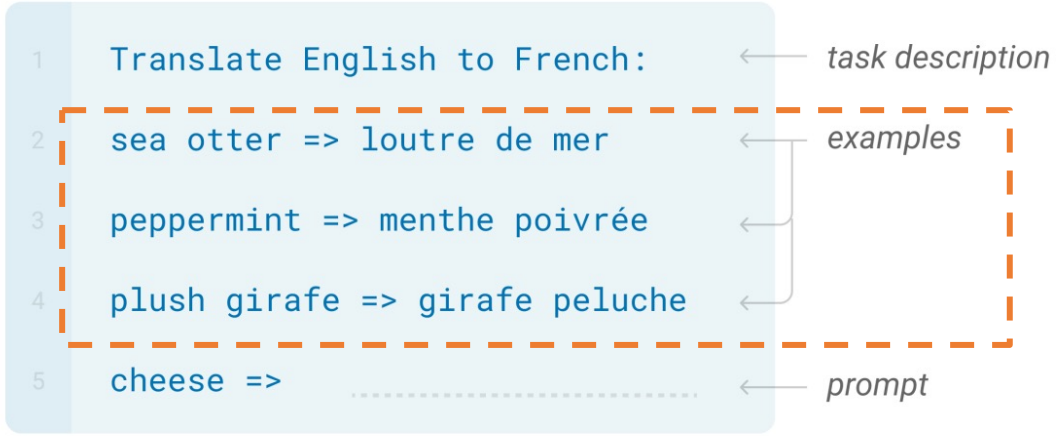


take beam search with a large width (=100) to get high-quality templates

Fine-tuning with Demonstrations

Few-shot

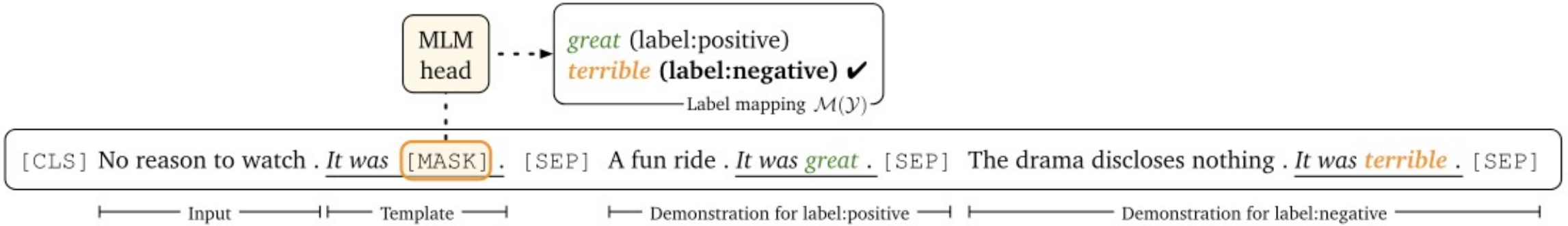
In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



GPT-3 style in-context learning:

Randomly sample examples and fill them in the context

Fine-tuning with Demonstrations



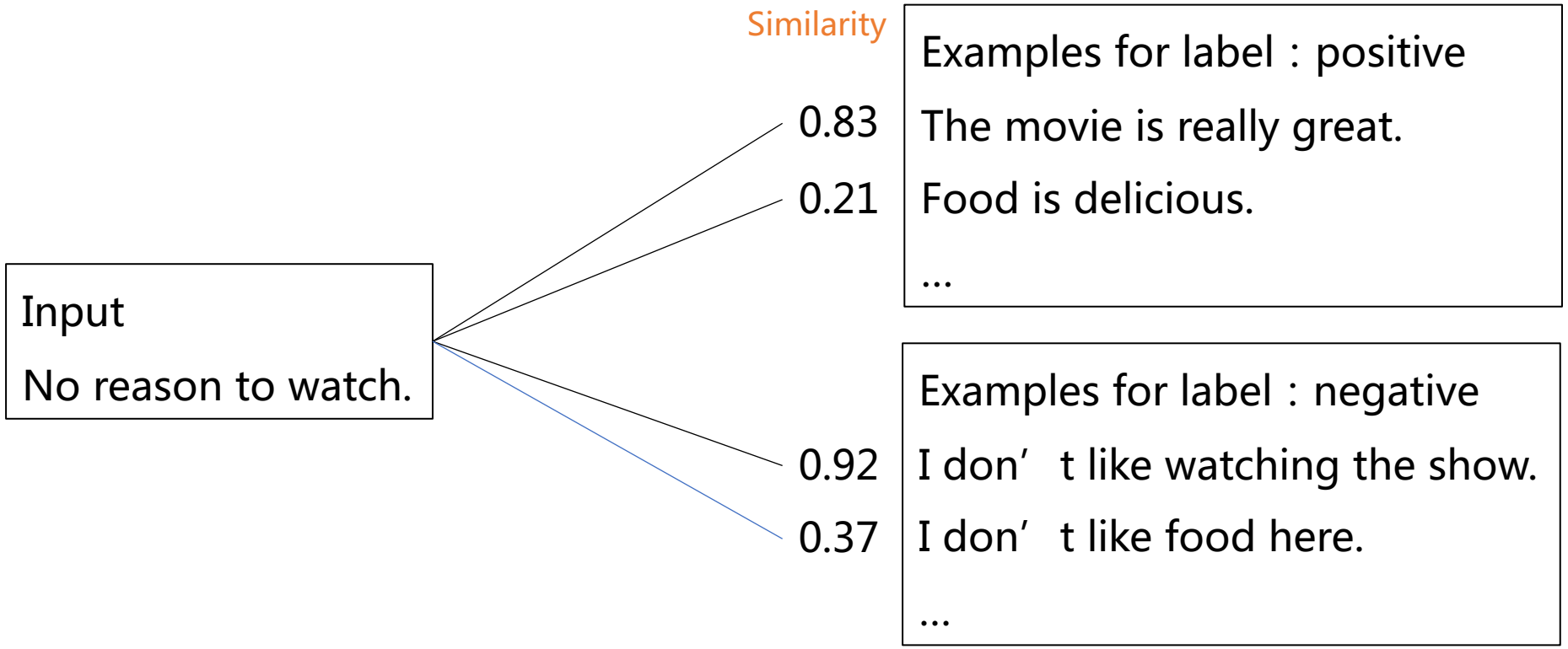
(c) Prompt-based fine-tuning with demonstrations (our approach)

Selectively Sampling Demonstrations

select demonstrations that are semantically close to the input :

measure the cosine similarities between the input and all the training examples

by using a pre-trained sentence encoder(Sentence-BERT)



Experiments Setup

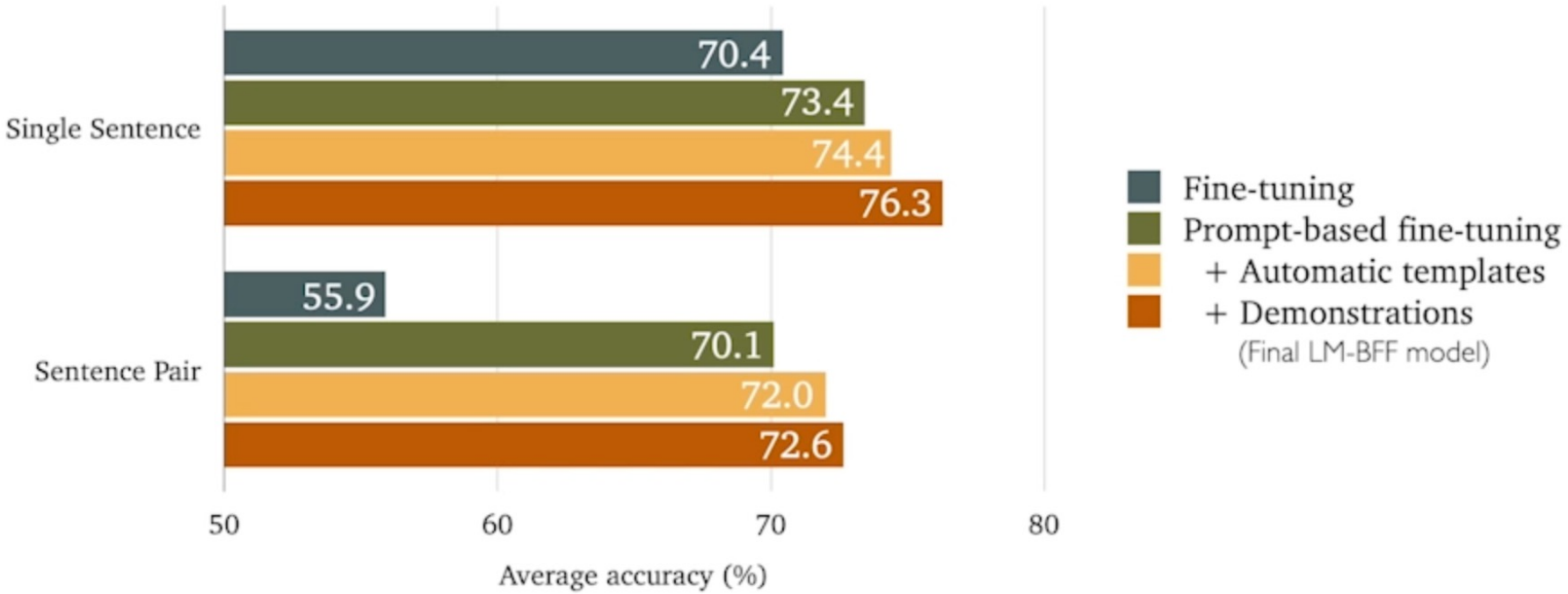
16 examples per class for both training and development sets

- original dev. set : not true few-shot learning
- no dev. set : "shooting in the dark" for hyper-parameters and early stopping

Sample 5 few-shot sets for each dataset and average the results

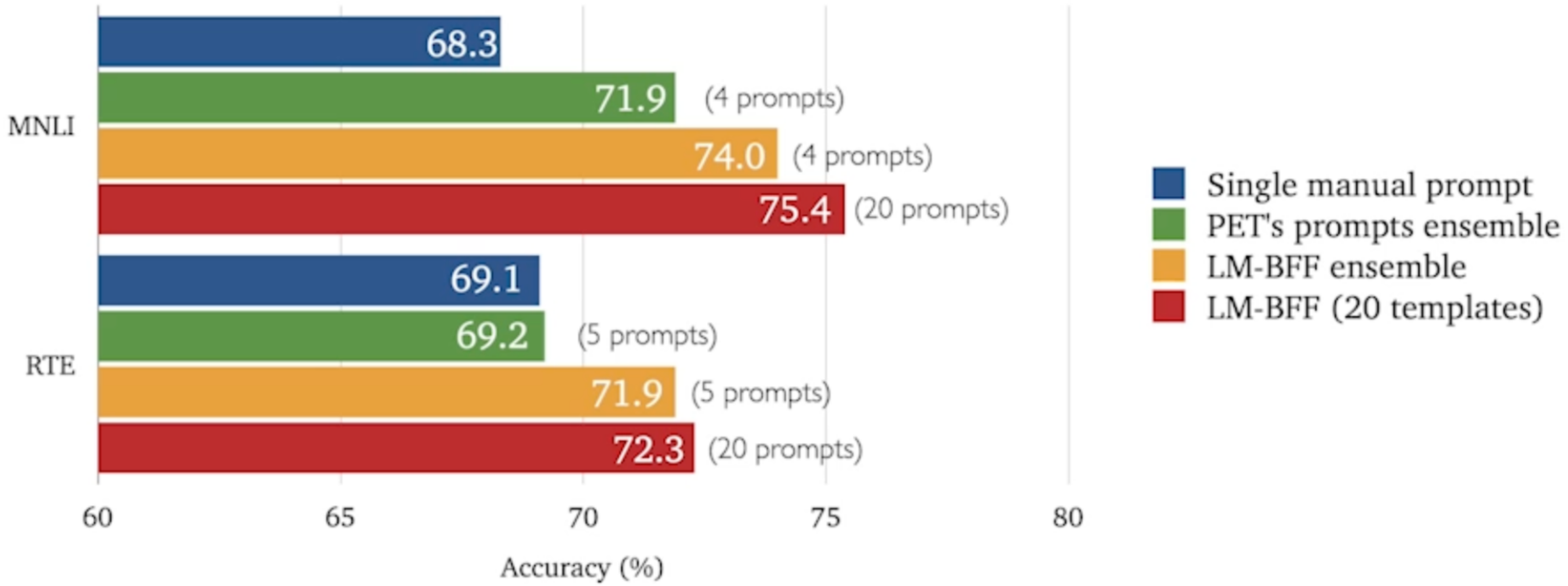
- performance has a large variance given different few-shot sets.

Main Results: Singal prompts

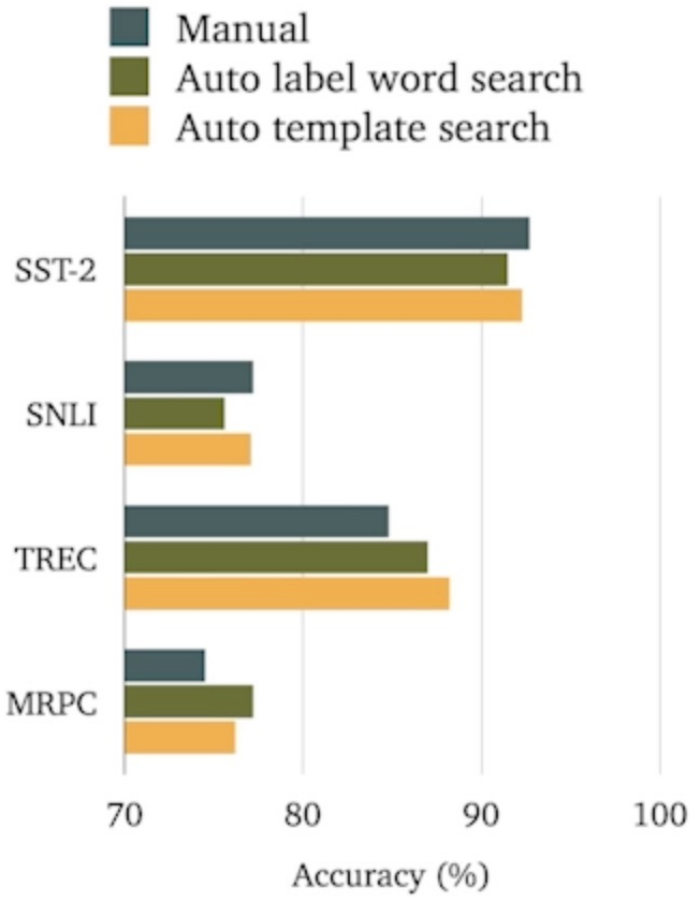


Main Results: Singal prompts

Main Results: Ensemble



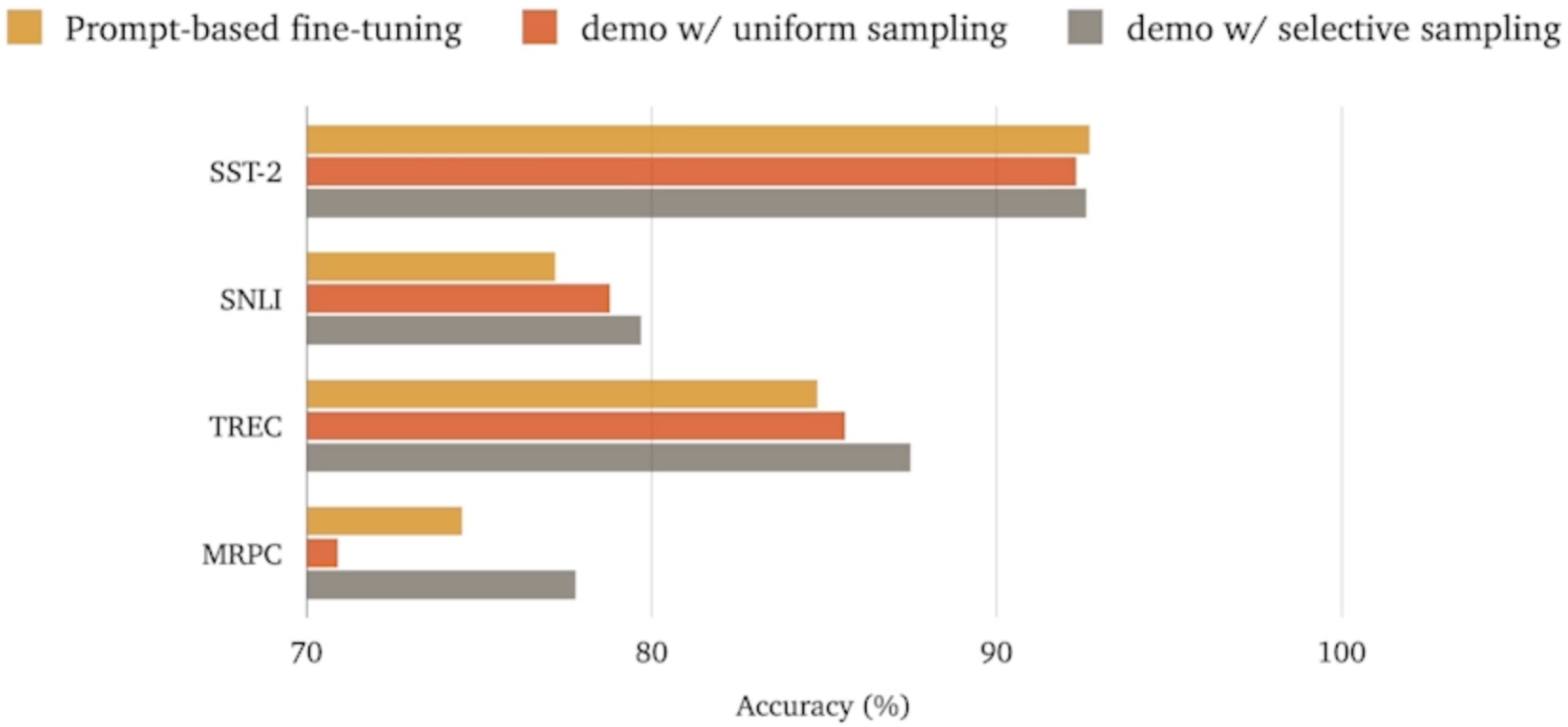
Ablation Study: Automatic Prompt Search



SST-2	(positive/negative)
	$\mathcal{M}(\mathcal{Y})$ = {great, terrible} #1. $<S_1>$ A `[MASK]` one . #2. $<S_1>$ A `[MASK]` piece . #3. $<S_1>$ All in all `[MASK]` .

SNLI	(entailment/neutral/contradiction)
	$\mathcal{M}(\mathcal{Y})$ = {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 $<S_2>$

Ablation Study: Sampling Demonstrations



Summary

- LM-BFF
  - Automatically-searched prompt-based fine-tuning
  - A tailored way for incorporating demonstrations
- A rigorous few-shot evaluation on 15 tasks
  - LM-BFF dramatically outperforms standard fine-tuning
- Limitations
  - Large variance
  - Automatic search still relies on either manual templates or label words
  - Favors tasks that can be posed as fill-in-the-blanks

Cutting Down on Prompts and Parameters: Simple Few-Shot Learning with Language Models

Robert L. Logan IV[1] **Ivana Balažević**[*2] **Eric Wallace**[3]
Fabio Petroni[4] **Sameer Singh**[1] **Sebastian Riedel**[4,5]
[1]UC Irvine [2]University of Edinburgh
[3]UC Berkeley [4]Facebook AI Research [5]University College London
{rlogan,sameer}@uci.edu ivana.balazevic@ed.ac.uk
ericwallace@berkeley.edu {fabiopetroni,sriedel}@fb.com

Motivation

Data Efficiency

Search Efficiency

Memory Efficiency

Improve

Few-Shot Language Model Performance

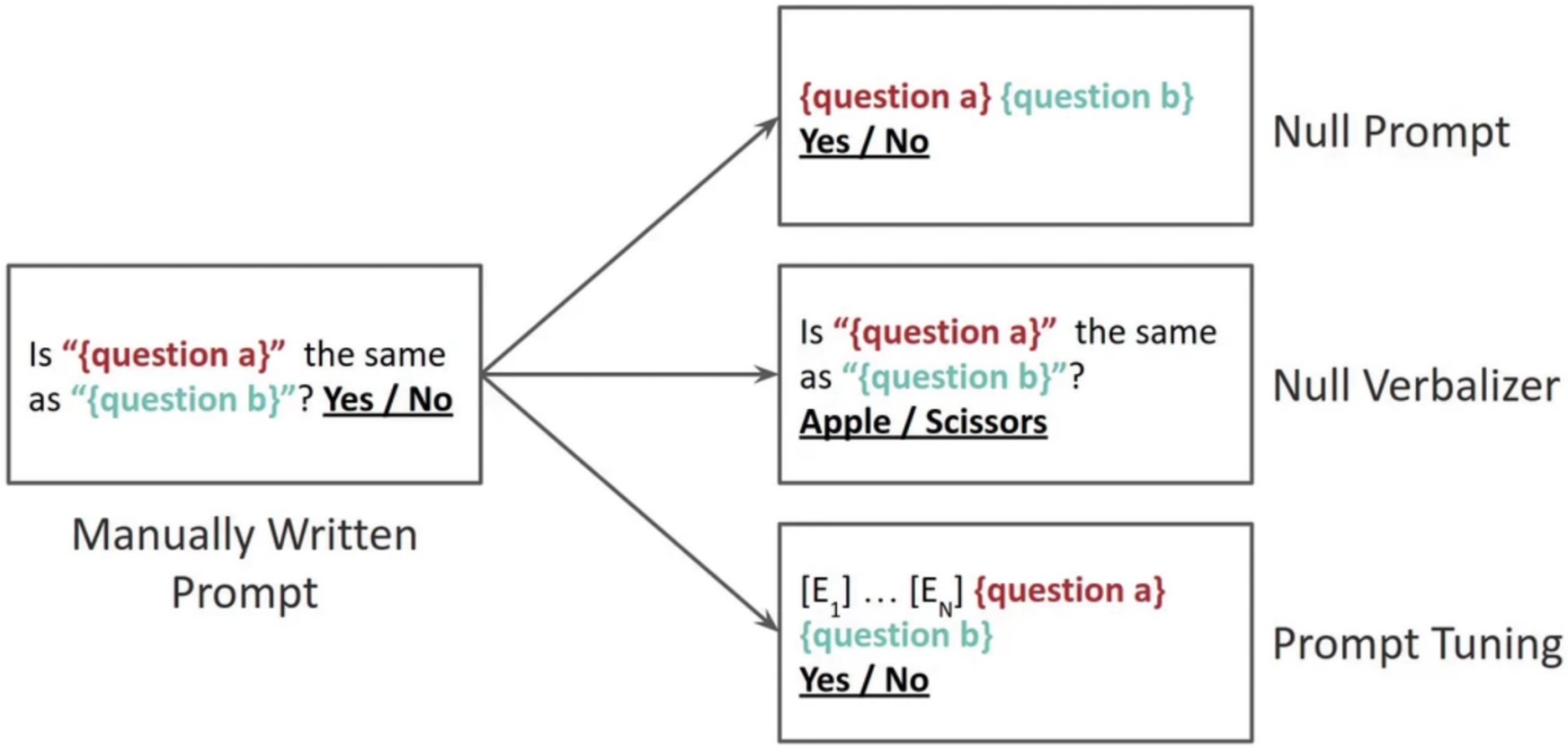
by

Streamlining Prompt Engineering

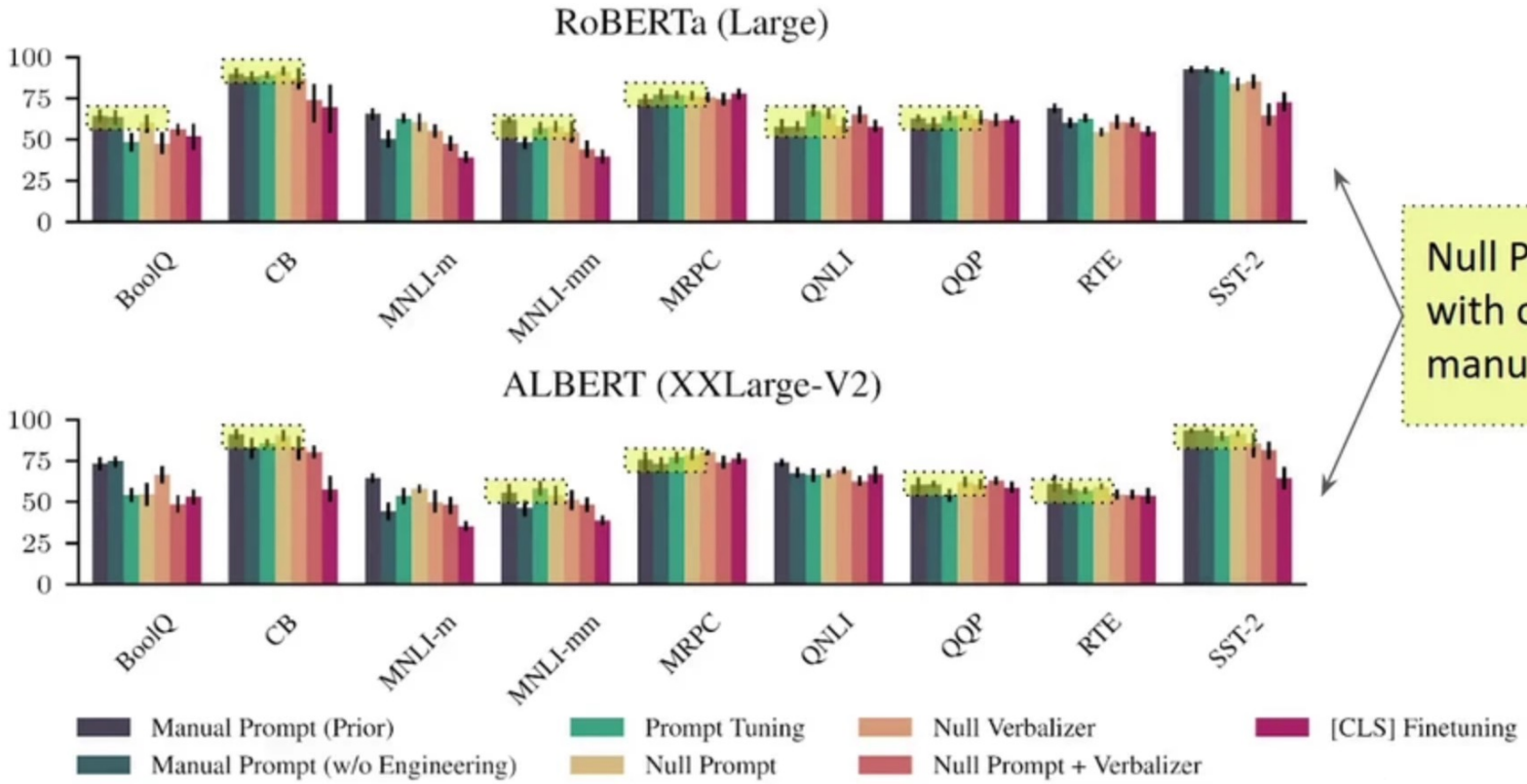
and

Reducing Parameter Counts

Simplifying Prompt Engineering

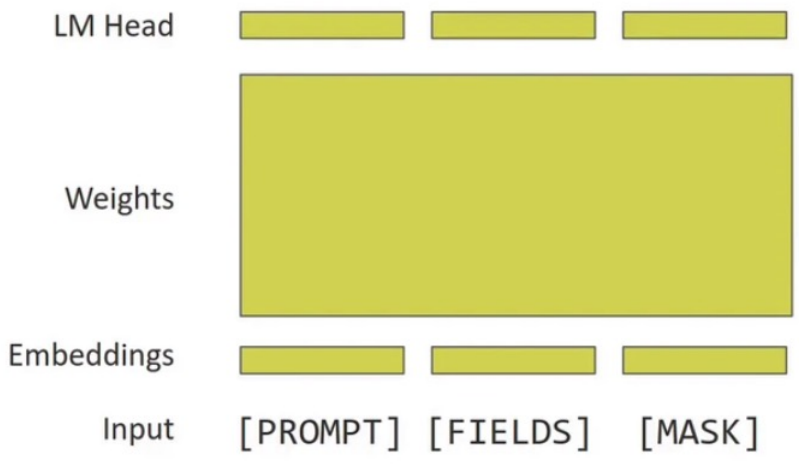


Simplifying Prompt Engineering

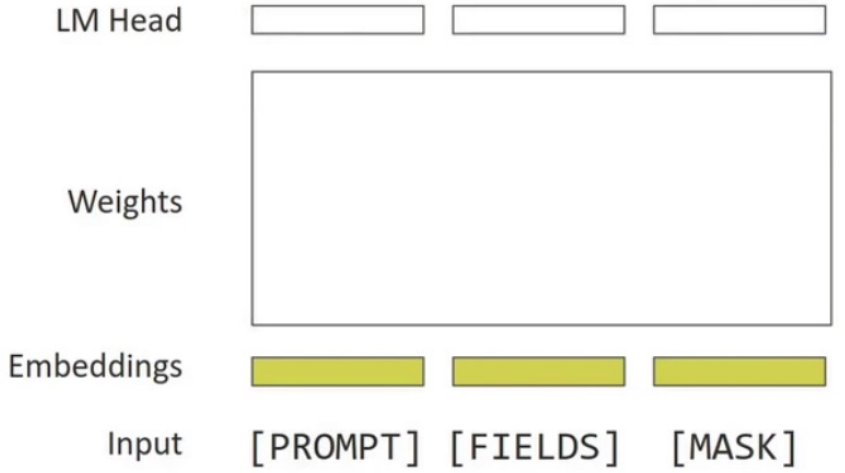


Null Prompts often on-par with or better than manually written prompts

Improving Parameter Efficiency

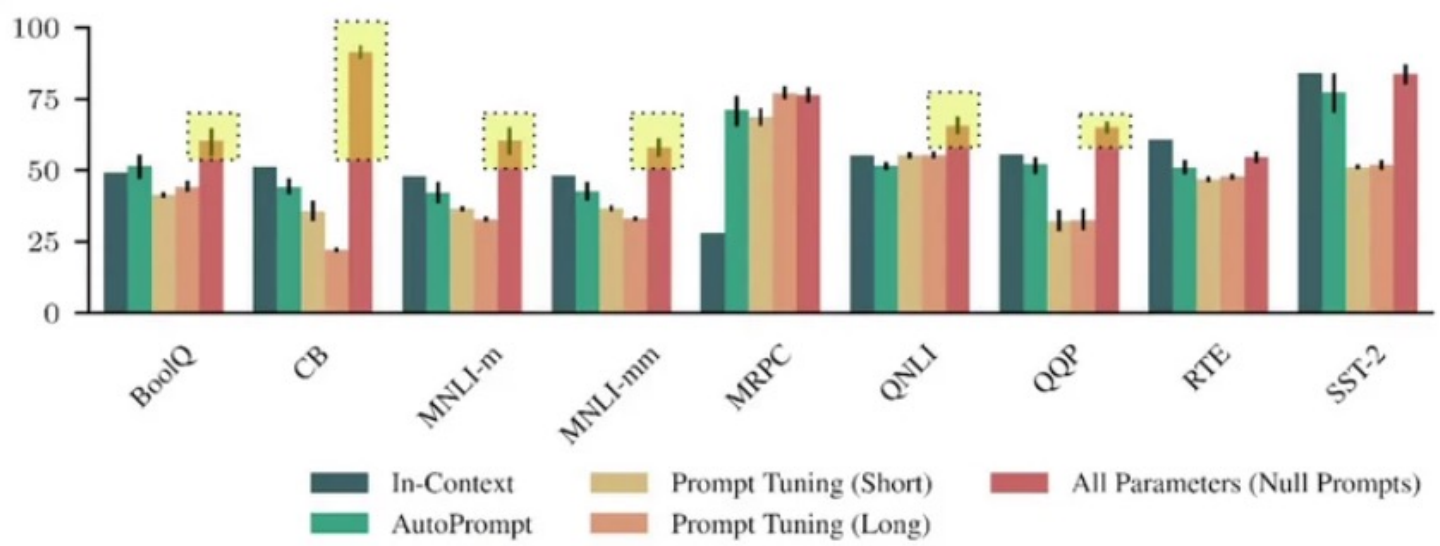


Prompt-Based Finetuning



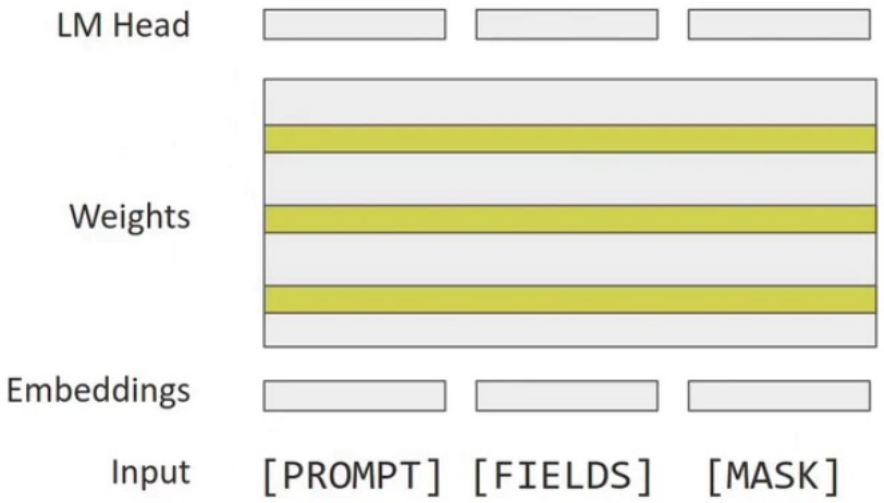
Prompt-Only Tuning

Improving Parameter Efficiency



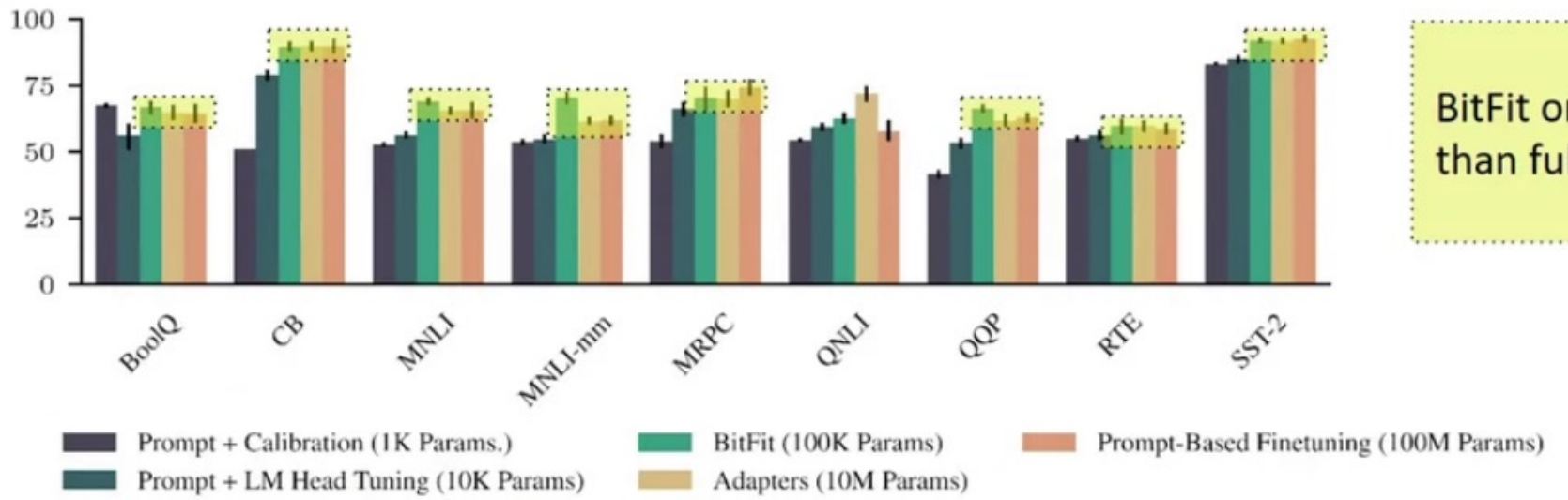
Prompt-Only Tuning Substantially Decreases Performance

Improving Parameter Efficiency



BitFit
&
Adapters

Improving Parameter Efficiency



BitFit on-par with or better than full finetuning

True Few-Shot Learning with Language Models

Ethan Perez[1], **Douwe Kiela**[2], **Kyunghyun Cho**[13]
[1]New York University, [2]Facebook AI Research,
[3]CIFAR Fellow in Learning in Machines & Brains
perez@nyu.edu

Learning Scenario	Many Train Distributions	Many Train Examples	Many Val Examples
Data-Rich Supervised	✗	✓	✓
Multi-Dist. Few-Shot	✓	✗	✗
Tuned Few-Shot	✗	✗	✓
True Few-Shot	✗	✗	✗

model selection criteria

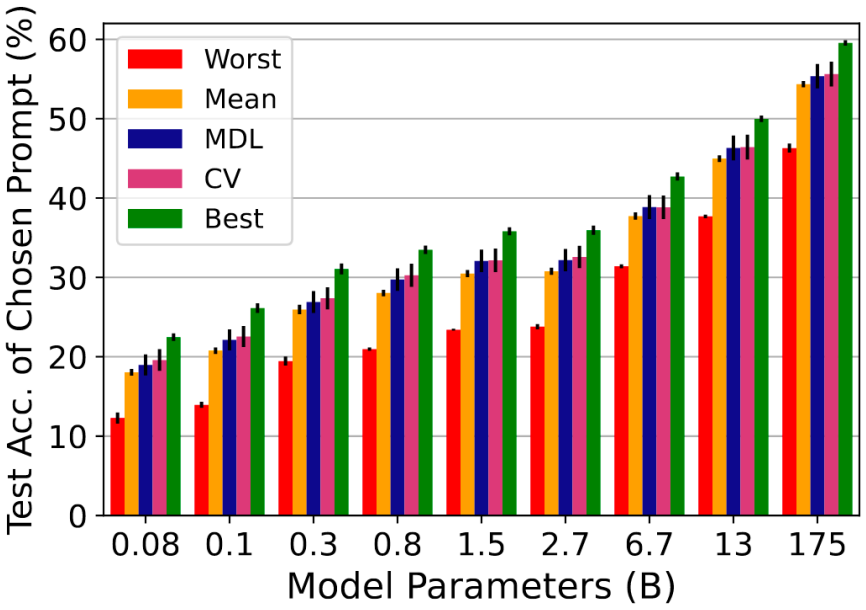
Cross-Validation (CV)

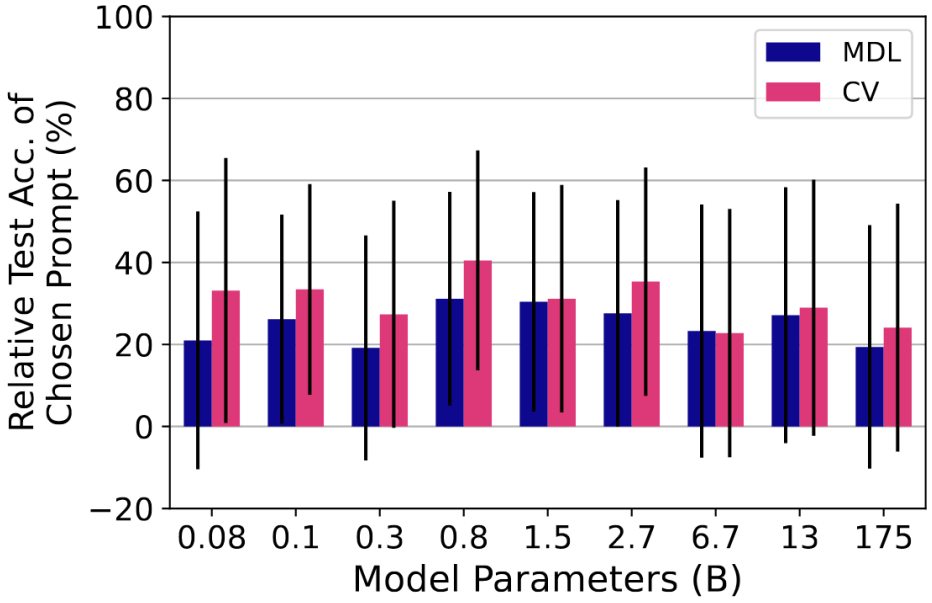
$$\mathrm{CV}(\mathcal{A}, R, F) = \mathbb{E}_{k \sim \mathrm{Unif}(1,K)} \left[\mathcal{L}\Big(\mathcal{A}(F(D_{\mathrm{train}})_{\neg k}, R); F(D_{\mathrm{train}})_k \Big) \right]$$

Minimum description length（MDL）

$$\mathrm{MDL}(\mathcal{A}, R, F) = \mathbb{E}_{k \sim \mathrm{Unif}(1,K)} \left[\mathcal{L}\Big(\mathcal{A}(F(D_{\mathrm{train}})_{1:k-1}, R); F(D_{\mathrm{train}})_k \Big) \right]$$

How well does prompt selection do in true few-shot learning?





True Few-Shot Hyperparameter Selection

	BoolQ Acc	CB Acc/F1	COPA Acc	RTE Acc	WiC Acc	WSC Acc	MultiRC EM/F1	ReCoRD EM/F1	Avg
Worst	$75.0_{4.8}$	$79.5_{2.3}$/$67.3_{7.8}$	$76.8_{2.2}$	$63.2_{4.0}$	$49.0_{1.3}$	$77.2_{1.8}$	$38.5_{7.4}$/$80.0_{2.9}$	$76.2_{1.8}$/$86.5_{1.2}$	$69.4_{1.5}$
Mean	$79.0_{1.5}$	$85.9_{2.3}$/$74.5_{11.0}$	$81.1_{2.9}$	$70.8_{2.5}$	$51.5_{1.8}$	$82.5_{2.7}$	$44.2_{6.6}$/$82.3_{2.7}$	$78.3_{1.3}$/$87.8_{0.8}$	$73.9_{1.2}$
MDL	$76.5_{5.8}$	$85.7_{5.6}$/$74.8_{13.4}$	$82.0_{2.9}$	$70.4_{8.5}$	$52.2_{3.0}$	$82.0_{3.1}$	$39.7_{8.1}$/$80.6_{3.2}$	$78.9_{0.7}$/$88.2_{0.4}$	$73.4_{2.8}$
CV	$78.9_{2.4}$	$83.9_{5.3}$/$69.2_{10.3}$	$80.5_{3.3}$	$68.7_{7.0}$	$51.1_{1.6}$	$83.1_{2.6}$	$41.9_{7.2}$/$81.4_{3.1}$	$78.7_{1.6}$/$88.1_{1.0}$	$73.0_{2.1}$
Best	$80.9_{1.0}$	$89.8_{3.1}$/$79.8_{13.4}$	$84.8_{4.5}$	$76.7_{1.8}$	$54.1_{2.3}$	$86.6_{1.8}$	$46.8_{6.9}$/$83.4_{2.9}$	$80.4_{1.1}$/$89.2_{0.7}$	$77.2_{0.9}$
ADAPET [12]	80.3	89.3 / 86.8	89.0	76.5	54.4	81.7	39.2 / 80.1	85.4 / 92.1	77.3
iPET [9]	80.6	92.9 / 92.4	95.0	74.0	52.2	80.1	33.0 / 74.0	86.0 / 86.5	76.8
PET [9]	79.4	85.1 / 59.4	95.0	69.8	52.4	80.1	37.9 / 77.3	86.0 / 86.5	74.1
GPT-3 [2]	77.5	82.1 / 57.2	92.0	72.9	55.3	75.0	32.5 / 74.8	89.0 / 90.1	73.2