How to Better Transfer the Pre-trained Language Model?

余阳　2022.03.23

Outline

- **Effectiveness**
  - Finetuning
  - Domain-adaptive finetuning
- **Efficiency**
  - Prompt-based learning
  - Parameter-efficient finetuning
- Conclusion

Outline

- Effectiveness
  - Finetuning
  - Domain-adaptive finetuning
- Efficiency
  - Prompt-based learning
  - Parameter-efficient finetuning
- Conclusion

Finetuning

- Pre-train -> Finetune (**NLP Paradigm #3**)
- Some tricks

  "The lower layer of the PLM may contain more general information."

  - **Layer selection**: select the most effective layer(s) for the downstream task.
  - **Layer-wise decreasing learning rate**

$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta)$$

$$\eta^{k-1} = \boxed{\xi} \cdot \eta^k$$

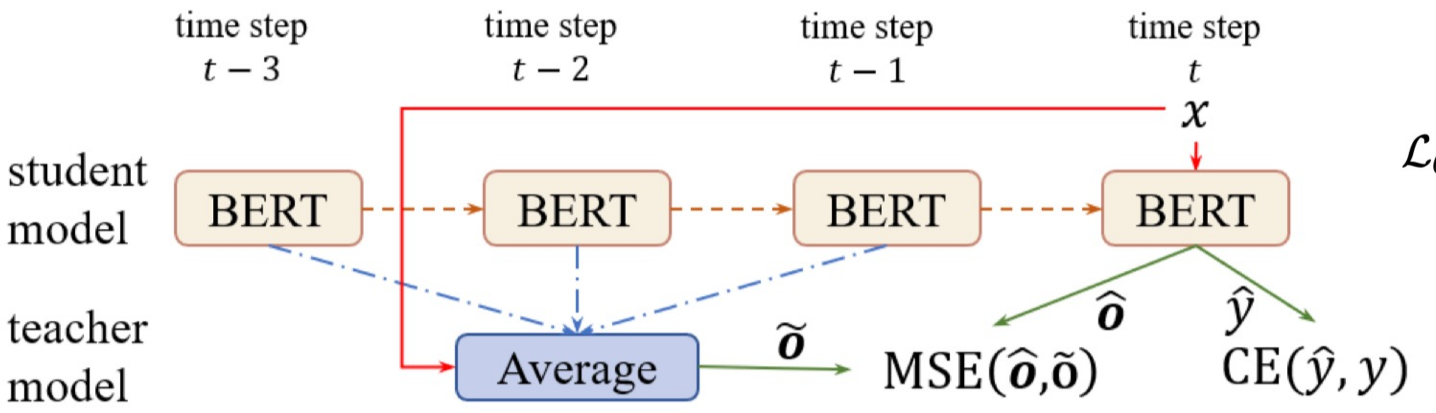
How to Fine-Tune BERT for Text Classification. Sun et al. 2019.

Finetuning

- Some tricks
  - **Self-ensemble & Self-distillation**: improve the stability of finetuning.
    - **Self-ensemble**

$$\mathrm{BERT}_{\mathrm{SE}}(x;\bar{\theta}) = \mathrm{BERT}(x;\frac{1}{T}\sum_{\tau=1}^{T}\theta_t)$$

    - **Self-distillation**

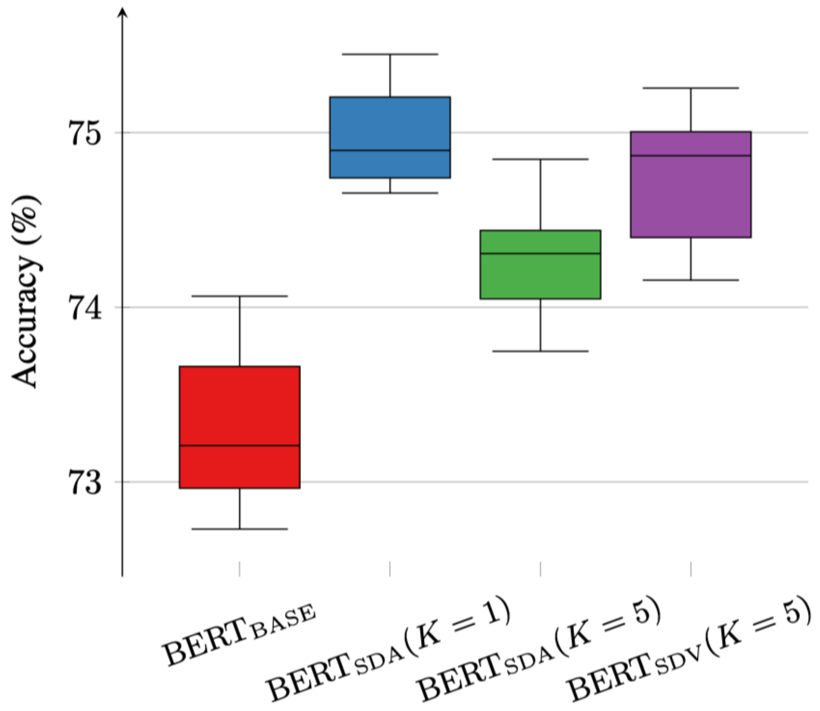


$$\mathcal{L}_\theta(x,y) = \mathrm{CE}\Big(\mathrm{BERT}(x,\theta),y\Big) + \lambda\mathrm{MSE}\Big(\mathrm{BERT}(x,\theta),\mathrm{BERT}(x,\bar{\theta})\Big)$$

Improving BERT Fine-Tuning via Self-Ensemble and Self-Distillation. Xu et al. 2020.

Finetuning

- Some tricks
  - **Self-ensemble & Self-distillation**: improve the stability of finetuning.



Improving BERT Fine-Tuning via Self-Ensemble and Self-Distillation. Xu et al. 2020.

Outline

- Effectiveness
  - Finetuning
  - **Domain-adaptive finetuning**
- Efficiency
  - Prompt-based learning
  - Parameter-efficient finetuning
- Conclusion

Domain-adaptive Finetuning

- The PLM is usually pre-trained on general corpora.
  - BookCorpus
  - Wikipedia
  - CCNews, OpenWebText, CommonCrawl, etc.
- Narrow the **data distribution gap** between the pre-training data and the downstream task data.

	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

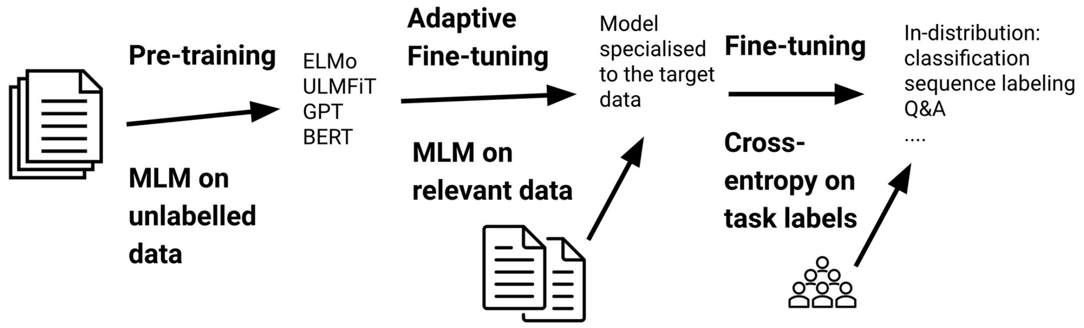
Figure 2: Vocabulary overlap (%) between domains. PT denotes a sample from sources similar to ROBERTA's pretraining corpus. Vocabularies for each domain are created by considering the top 10K most frequent words (excluding stopwords) in documents sampled from each domain.

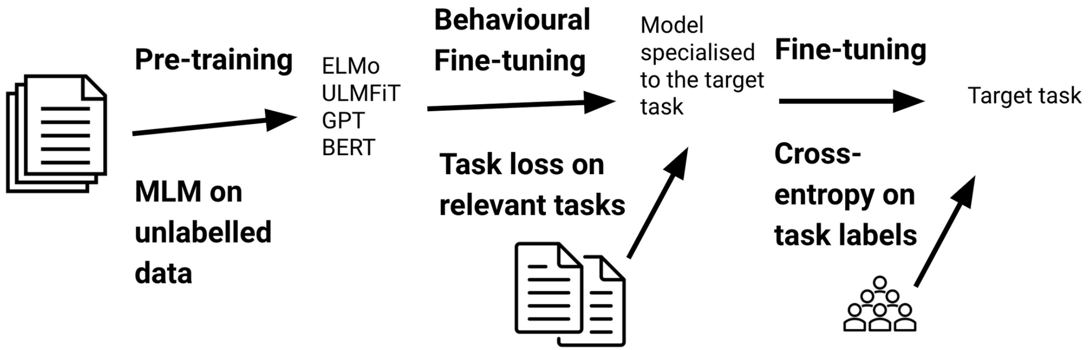
Domain	**Pretraining Corpus**	**# Tokens**	**Size**
BIOMED	2.68M full-text papers from S2ORC (Lo et al., 2020)	7.55B	47GB
CS	2.22M full-text papers from S2ORC (Lo et al., 2020)	8.10B	48GB
NEWS	11.90M articles from REALNEWS (Zellers et al., 2019)	6.66B	39GB
REVIEWS	24.75M AMAZON reviews (He and McAuley, 2016)	2.11B	11GB

Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. Gururangan et al, ACL 2020 Honorable Mention Papers.

Domain-adaptive Finetuning

- **Self-supervised task on downstream domain corpora**
  - Domain-adaptive pre-training (DAPT) / Task-adaptive pre-training (TAPT)
  - Contrastive Learning, etc.
- **Supervised relevant task**
  - Sequential
  - Parallel (i.e., multi-task finetuning)





Domain	Task	RoBERTa	Additional Pretraining Phases: DAPT	TAPT	DAPT + TAPT
BioMed	ChemProt	$81.9_{1.0}$	$84.2_{0.2}$	$82.6_{0.4}$	**$84.4_{0.4}$**
	†RCT	$87.2_{0.1}$	$87.6_{0.1}$	$87.7_{0.1}$	**$87.8_{0.1}$**
CS	ACL-ARC	$63.0_{5.8}$	$75.4_{2.5}$	$67.4_{1.8}$	**$75.6_{3.8}$**
	SciERC	$77.3_{1.9}$	$80.8_{1.5}$	$79.3_{1.5}$	**$81.3_{1.8}$**
News	HyperPartisan	$86.6_{0.9}$	$88.2_{5.9}$	**$90.4_{5.2}$**	$90.0_{6.6}$
	†AGNews	$93.9_{0.2}$	$93.9_{0.2}$	$94.5_{0.1}$	**$94.6_{0.1}$**
Reviews	†Helpfulness	$65.1_{3.4}$	$66.5_{1.4}$	$68.5_{1.9}$	**$68.7_{1.8}$**
	†IMDB	$95.0_{0.2}$	$95.4_{0.1}$	$95.5_{0.1}$	**$95.6_{0.1}$**

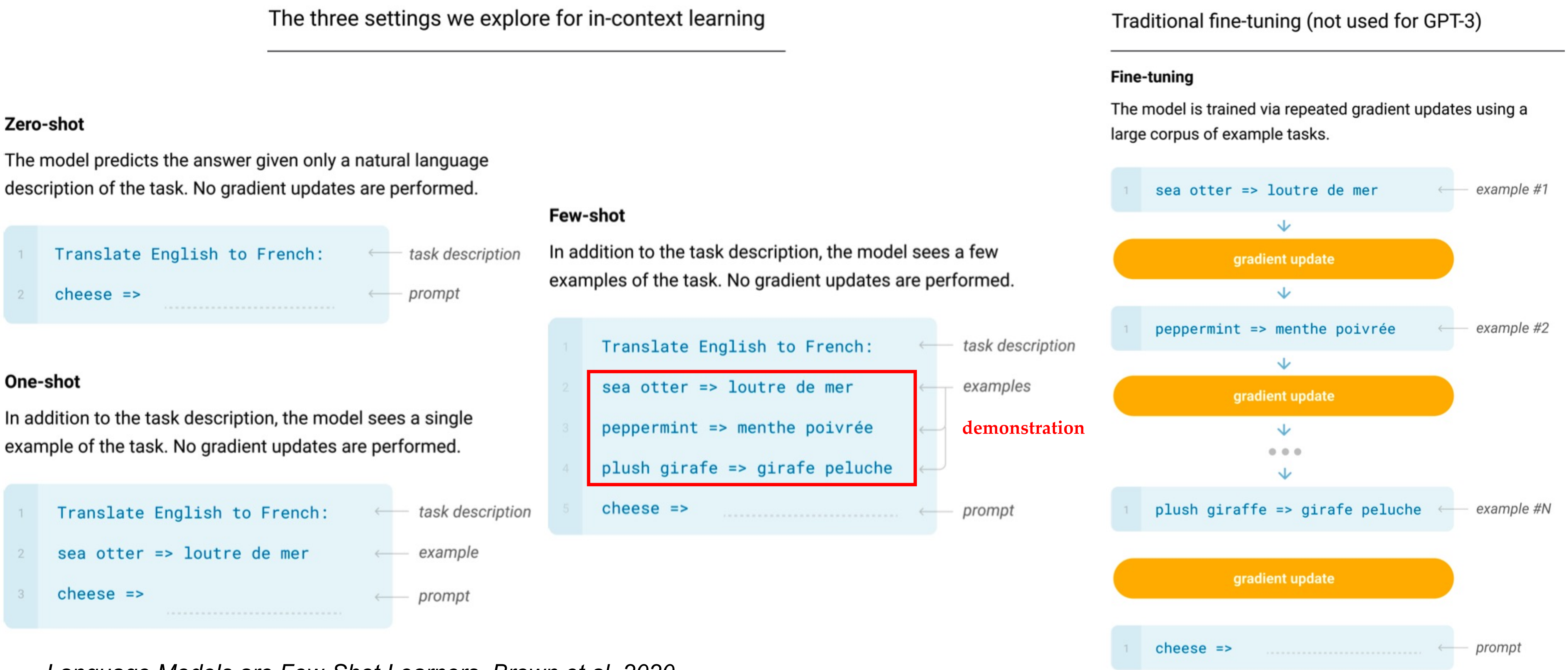
Outline

- Effectiveness
  - Finetuning
  - Domain-adaptive finetuning
- Efficiency
  - **Prompt-based learning**
  - Parameter-efficient finetuning
- Conclusion

Prompt-based Learning

- **Motivation**
  - The PLM need to be finetuned for every new downstream task.
  - Finetuning is costly for extremely large PLMs.
    - E.g., T5 (11B), GPT-3 (175B)
  - Finetuning requires thousands to hundred of thousands task-specific examples.
    - Human can perform a new language task with a few examples or simple instructions.
- **Solution**
  - In-context learning (prompt + demonstration) -> prompt-based learning

Prompt-based Learning



Language Models are Few-Shot Learners. Brown et al. 2020.

Prompt-based Learning

- **Definition**: Prompt is the technique of making better use of the knowledge from the PLM by adding additional texts to the input.
- **Basic framework**

Step I. Prompt Addition — **Step II. Answer Search** — **Step III. Answer Mapping**

input text x $\xrightarrow{\text{Step I}}$ prompt x' $\xrightarrow{\text{Step II}}$ filled prompt $\hat{x}$ $\xrightarrow{\text{Step III}}$ output $\hat{y}$

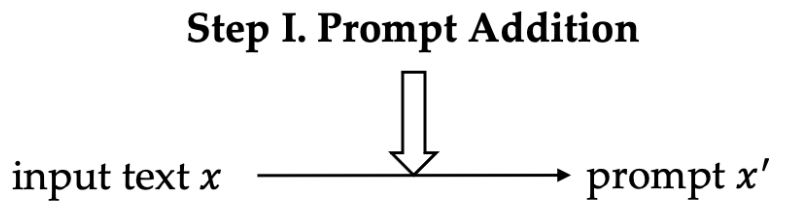
Prompt-based Learning

- **Prompt addition**
  - Design a **template** with two slots: input slot [X] and answer slot [Z].
  - Fill slot [X] with the input text x.

  - E.g., sentiment analysis (**prefix prompt**)

    template = "[X] The movie is [Z]."
    input text x = "I love this movie."
    prompt x' = "I love this movie. The movie is [Z]."
  - E.g., named entity recognition (**cloze prompt**)

    template = "[X1] [X2] is a [Z] entity."
    input text [X1] = "Mike went to Paris." [X2] = "Paris"
    prompt x' = "Mike went to Paris. Paris is a [Z] entity."



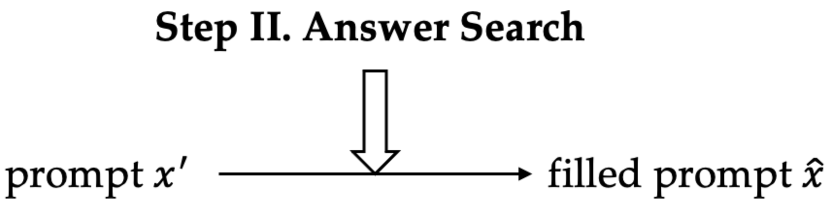
Prompt-based Learning

- **Answer search**
  - Search for text $\hat{z} \in \mathcal{Z}$ that maximize the score of a PLM $P(\cdot;\theta)$.

$$\hat{\boldsymbol{z}} = \underset{\boldsymbol{z} \in \mathcal{Z}}{\text{search}}\, P(f_{\text{fill}}(\boldsymbol{x}', \boldsymbol{z}); \theta)$$

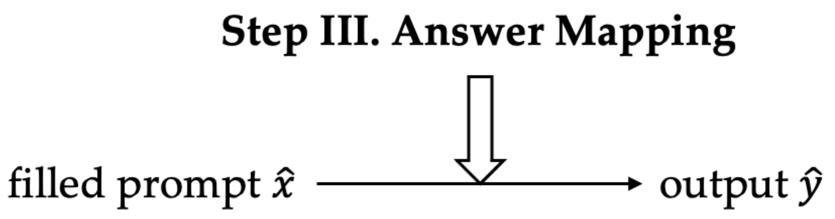
  - $\mathcal{Z}$ can be the entire vocabulary set, or a small subset specific to the target task.
  - The search function can be implemented as *argmax* or *sampling*.

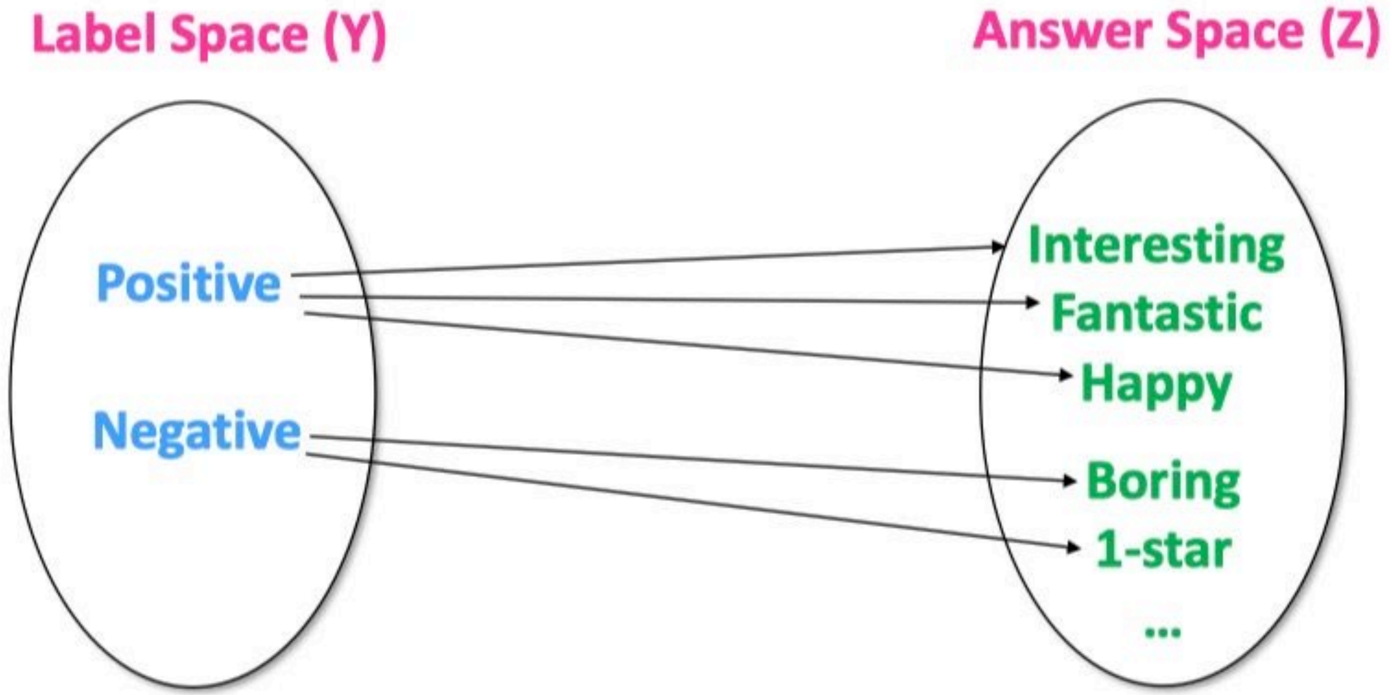
  - E.g., $\mathcal{Z}$ = {"excellent", "good", "OK", "bad", "horrible"} for sentiment analysis.



Prompt-based Learning

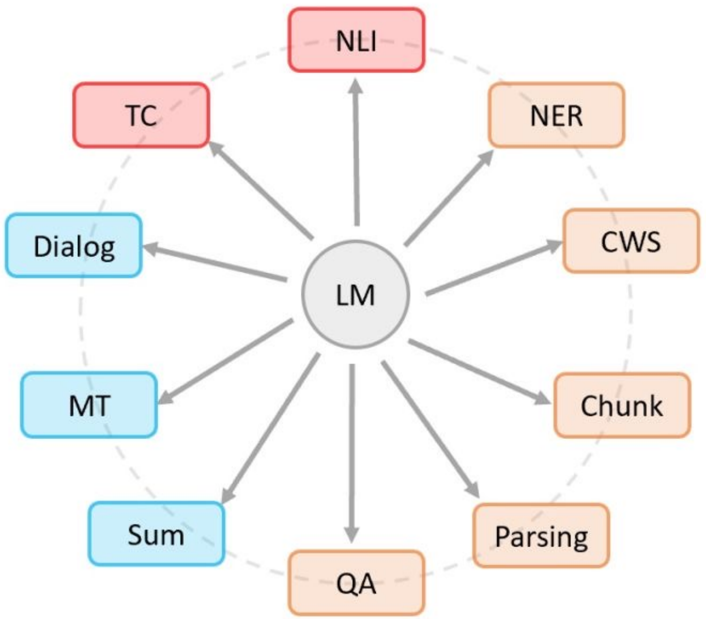
- **Answer mapping**
  - Map the answer $\hat{z}$ to the output $\hat{y}$.
  - Multiple answers can result in the same output.



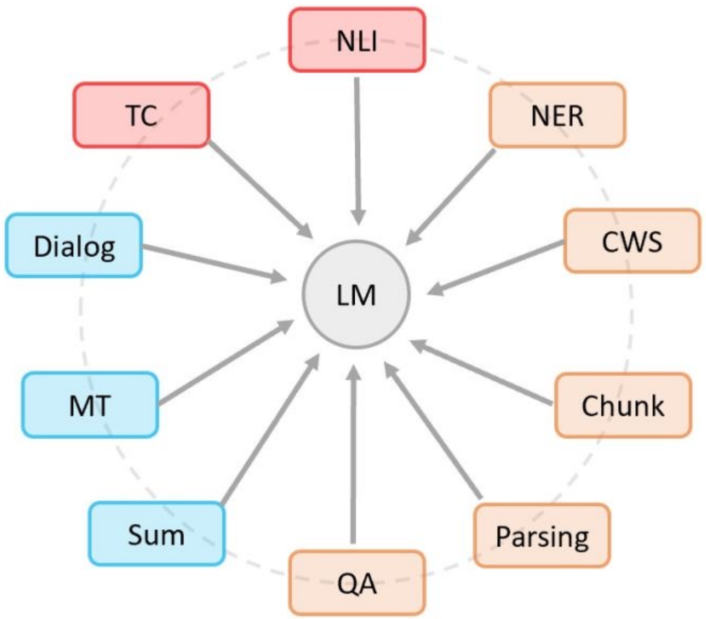


Prompt-based Learning

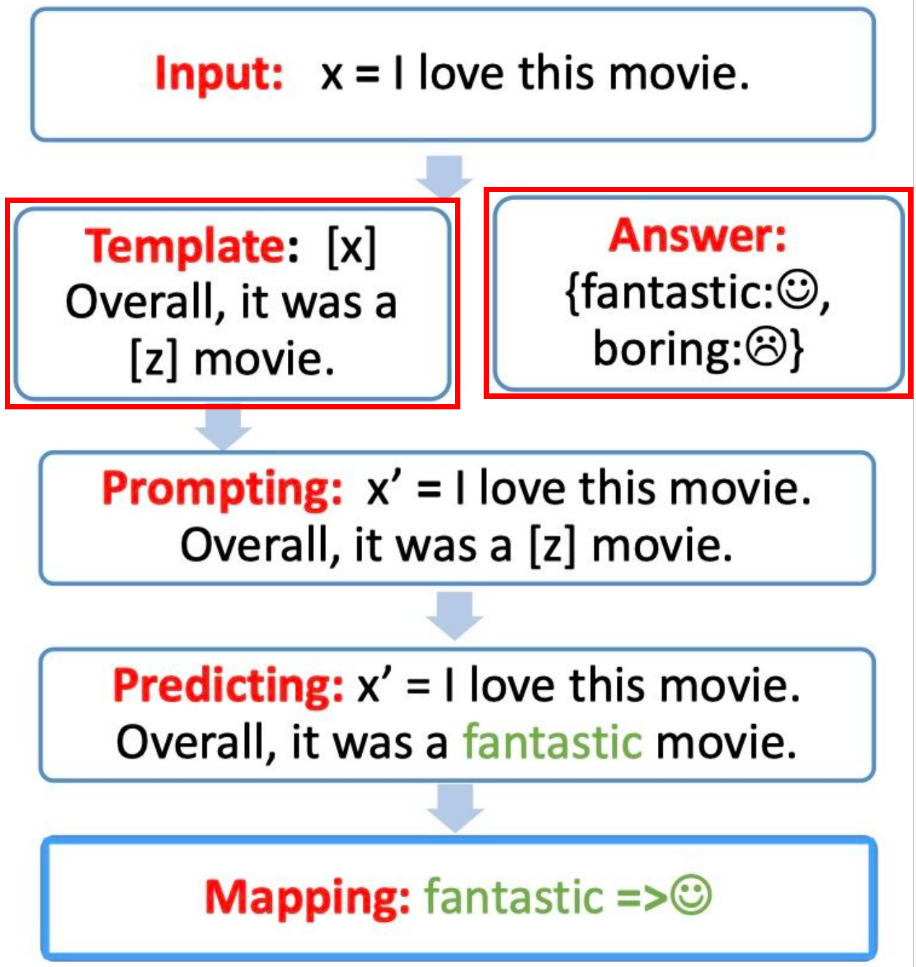
- Pre-train -> Prompt -> Predict (**NLP Paradigm #4**)
- Narrow the gap between the pre-training task and the downstream task.



Finetuning

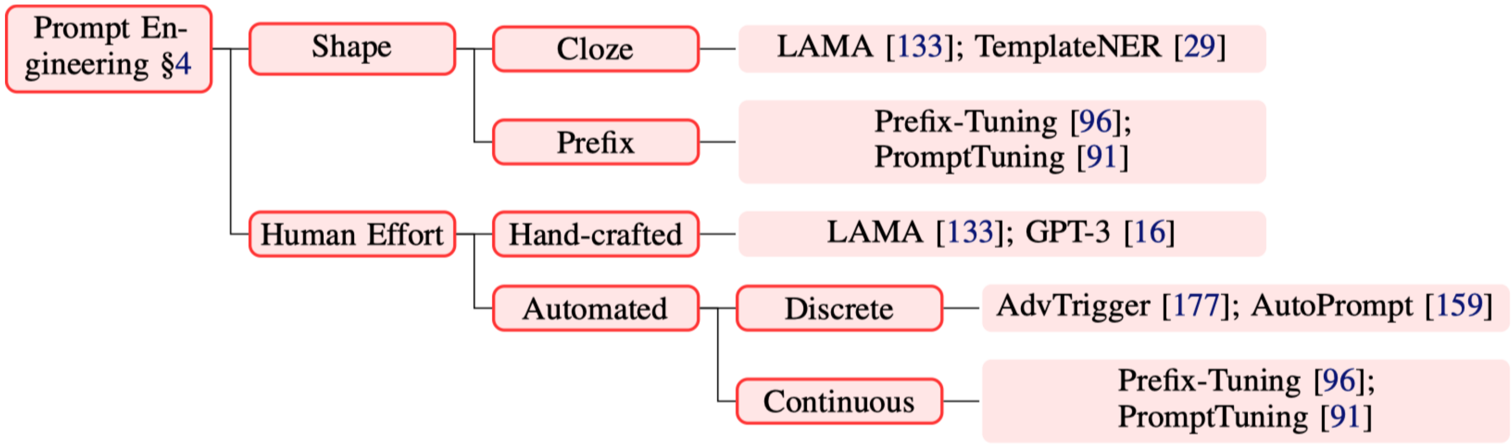


Prompting

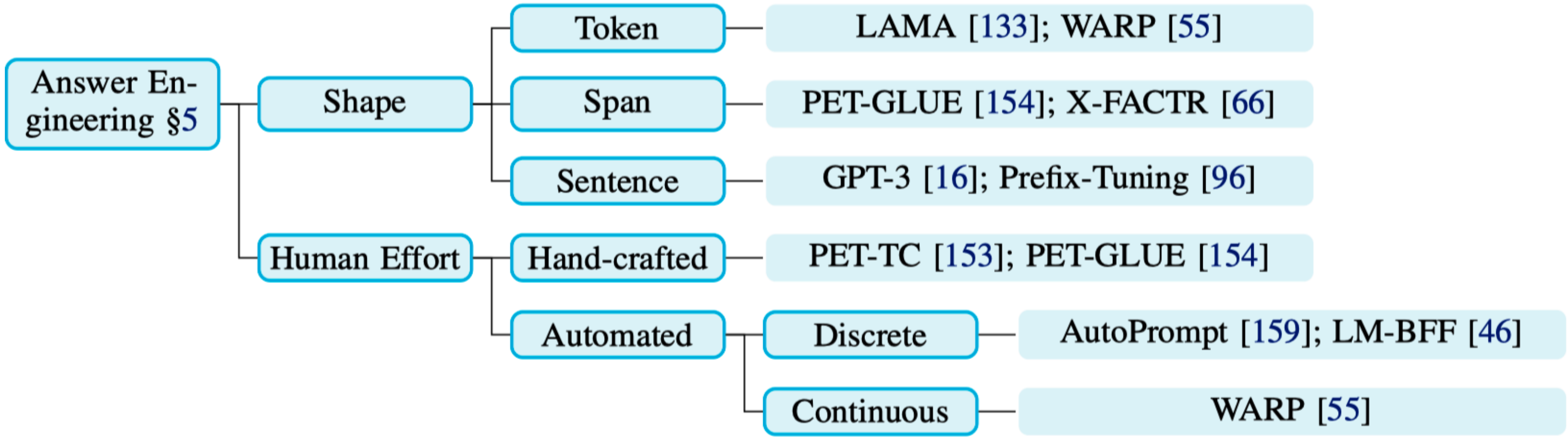


Prompt-based Learning

- Human efforts
  - **Prompt engineering**



  - **Answer engineering**



Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. Liu et al. 2021.

Prompt-based Learning

- Human efforts
  - **Prompt-based training strategies**

zero-shot

Strategy	**LM Params**	**Prompt Params**		**Example**
		Additional	**Tuned**	
Promptless Fine-tuning	Tuned	-		ELMo [130], BERT [32], BART [94]
Tuning-free Prompting	Frozen	✗	✗	GPT-3 [16], AutoPrompt [159], LAMA [133]
Fixed-LM Prompt Tuning	Frozen	✓	Tuned	Prefix-Tuning [96], Prompt-Tuning [91]
Fixed-prompt LM Tuning	Tuned	✗	✗	PET-TC [153], PET-Gen [152], LM-BFF [46]
Prompt+LM Fine-tuning	Tuned	✓	Tuned	PADA [8], P-Tuning [103], PTR [56]

few-shot

Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. Liu et al. 2021.

Prompt-based Learning

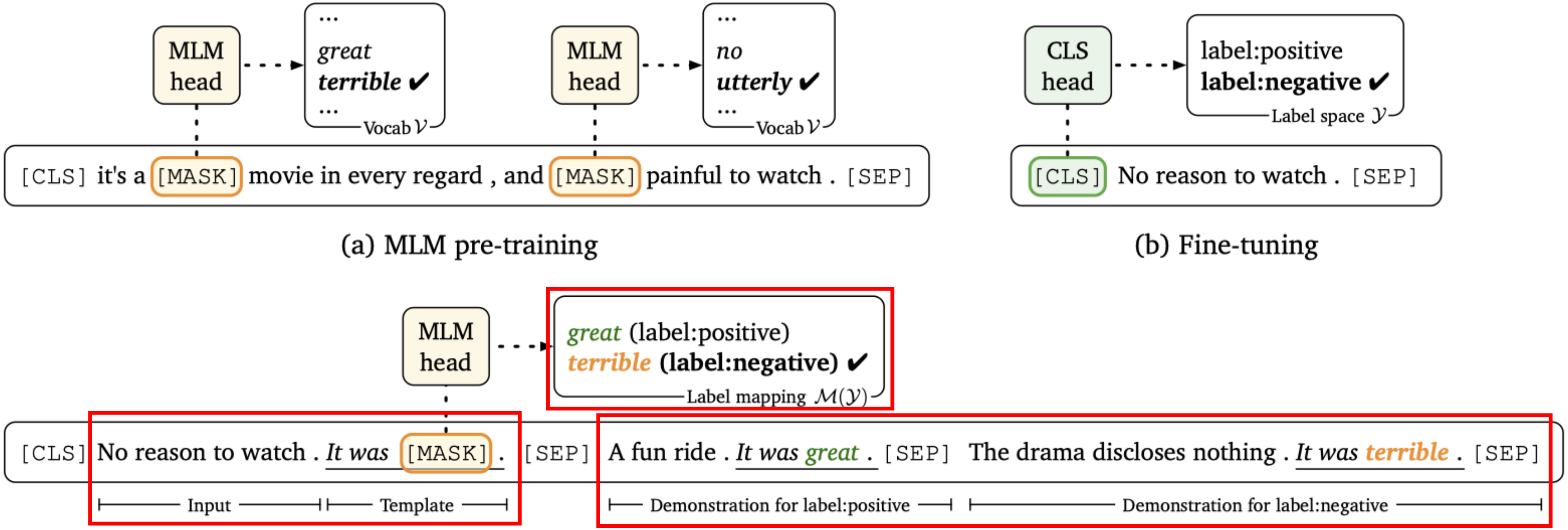
- **Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)**
  - "Finding the right prompts, however, is an art – requiring both domain expertise and an understanding of the language model's inner workings."

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)

Making Pre-trained Language Models Better Few-shot Learners. Gao et al. ACL 2021.

Prompt-based Learning

Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)



(a) MLM pre-training

(b) Fine-tuning

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

Making Pre-trained Language Models Better Few-shot Learners. Gao et al. ACL 2021.

Prompt-based Learning

- **Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)**
  - **Automatic selection of label words** (given a fixed template $\mathcal{T}$)
    - For each class c, select top k words that maximize the total probability of D^{c}_{train} using the initial PLM.

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

    - Further find the top n words that maximize zero-shot accuracy on D_{train}.
    - Finetune all top n assignments and select the best one on D_{dev}.

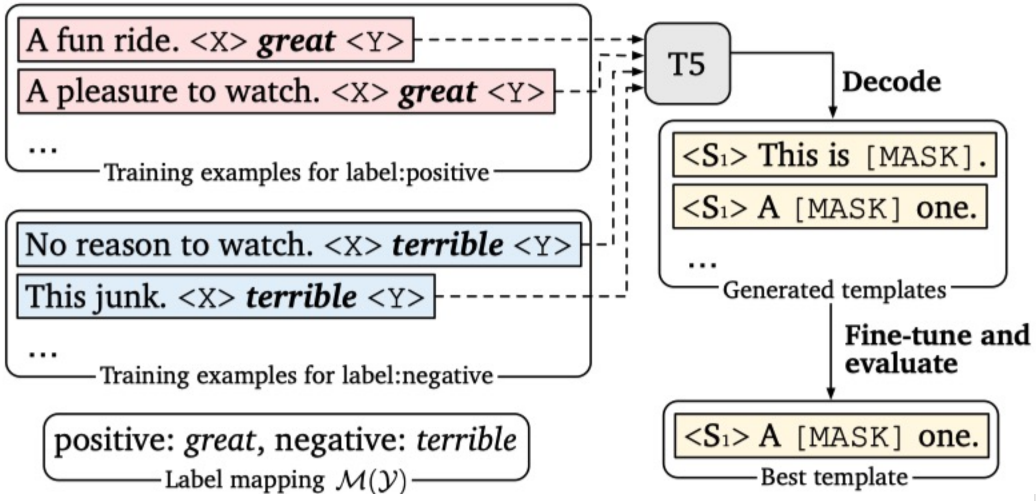
Making Pre-trained Language Models Better Few-shot Learners. Gao et al. ACL 2021.

Prompt-based Learning

- **Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)**
  - **Automatic generation of templates** (given a fixed set of label words $\mathcal{M}(\mathcal{Y})$)
    - Conduct simple conversions to each training sample $(x_{in}, y) \in D_{train}$.

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

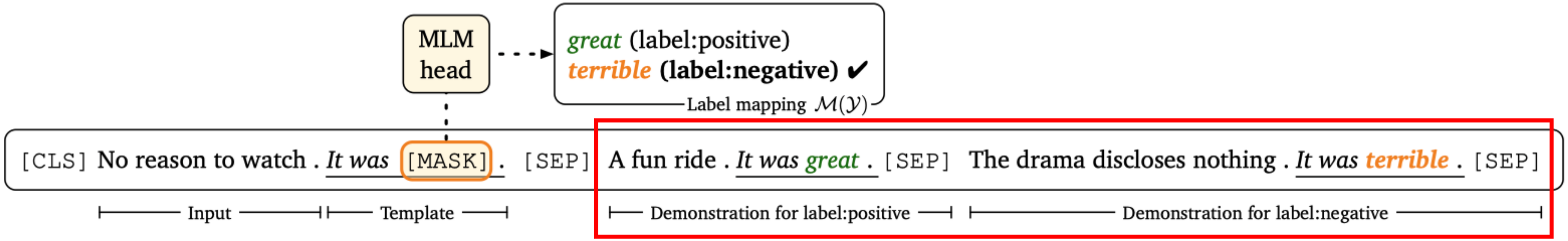
    - Use T5 to fill in missing spans.
    - Use beam search to decode multiple templates.
    - Finetune each generated templated on D_{train} and select the best one on D_{dev}.



Making Pre-trained Language Models Better Few-shot Learners. Gao et al. ACL 2021.

Prompt-based Learning

- **Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)**
  - **Finetune with demonstrations**
    - Use the pre-trained SBERT to generate the embeddings of each training sample $x_{in}^{(c)}$.
    - Randomly sample one example $(x_{in}^{(c)}, y^{(c)})$ from the top 50% samples that most similar to x_{in}.
    - Convert to filled prompt $\tilde{\mathcal{T}}(x_{in}^{(c)}, y^{(c)})$ and concatenate with x_{in}.



Making Pre-trained Language Models Better Few-shot Learners. Gao et al. ACL 2021.

Prompt-based Learning

- **Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)**

Task	Auto template	Auto label words
SST-2	(positive/negative)	
	$<S_1>$ A [MASK] one .	irresistible/pathetic
	$<S_1>$ A [MASK] piece .	wonderful/bad
	$<S_1>$ All in all [MASK] .	delicious/bad
SST-5	(very positive/positive/neutral/negative/very negative)	
	$<S_1>$ The movie is [MASK] .	wonderful/remarkable/hilarious/better/awful
	$<S_1>$ The music is [MASK] .	wonderful/perfect/hilarious/better/awful
	$<S_1>$ But it is [MASK] .	unforgettable/extraordinary/good/better/terrible
MR	(positive/negative)	
	It was [MASK] ! $<S_1>$	epic/terrible
	$<S_1>$ It's [MASK] .	epic/awful
	$<S_1>$ A [MASK] piece of work .	exquisite/horrible
CR	(positive/negative)	
	$<S_1>$ It's [MASK] !	fantastic/horrible
	$<S_1>$ The quality is [MASK] .	neat/pointless
	$<S_1>$ That is [MASK] .	magnificent/unacceptable

Prompt-based Learning

- **Making Pre-trained Language Models Better Few-shot Learners. (ACL 2021)**

	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*

Outline

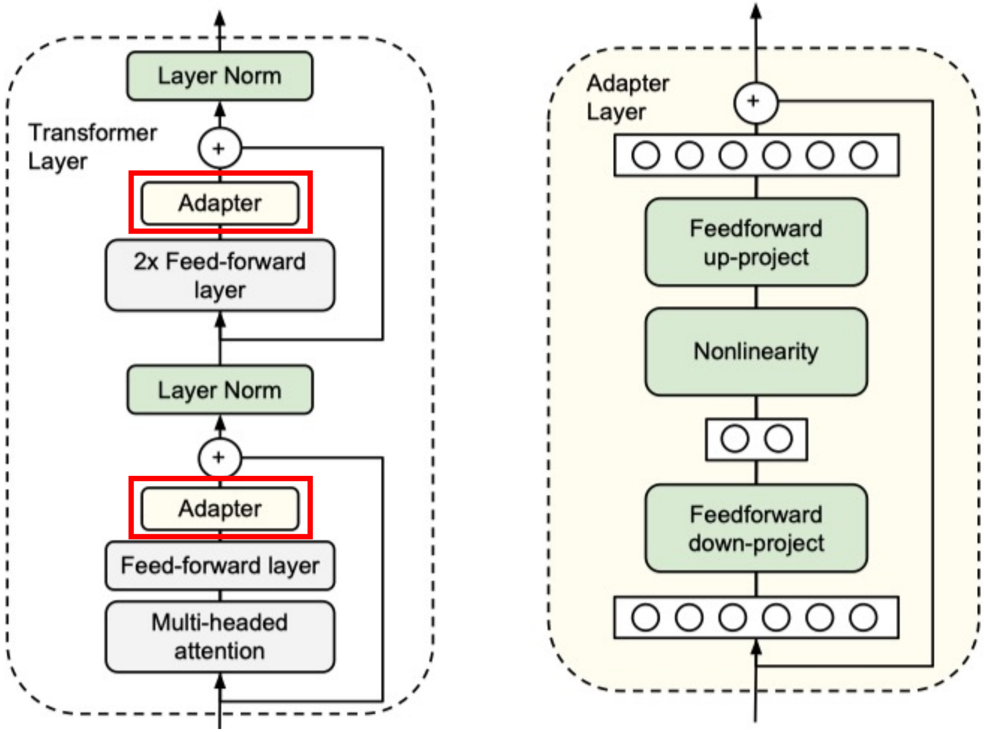
- Effectiveness
  - Finetuning
  - Domain-adaptive finetuning
- Efficiency
  - Prompt-based learning
  - **Parameter-efficient finetuning**
- Conclusion

Parameter-efficient Finetuning

- **Motivation**: Finetuning the entire PLM is parameter inefficient.
- **Solution**: Fix the PLM and only finetune a few additional parameters.
  - Adapter
  - Prefix-tuning & Prompt-tuning
  - Low-rank adaptation

Parameter-efficient Finetuning

- **Parameter-Efficient Transfer Learning for NLP (ICML 2019)**
  - Add some small adapter modules between layers of the PLM.
  - A new set of adapters are added and finetuned for every new task.
  - Adapter modules
    - **small number of parameters**
      - origin dimension d, projection dimension m
      - total number of parameters = $2md + d + m$
    - **near-identity initialization**
      - skip-connection
      - near-zero initialization for projection layers

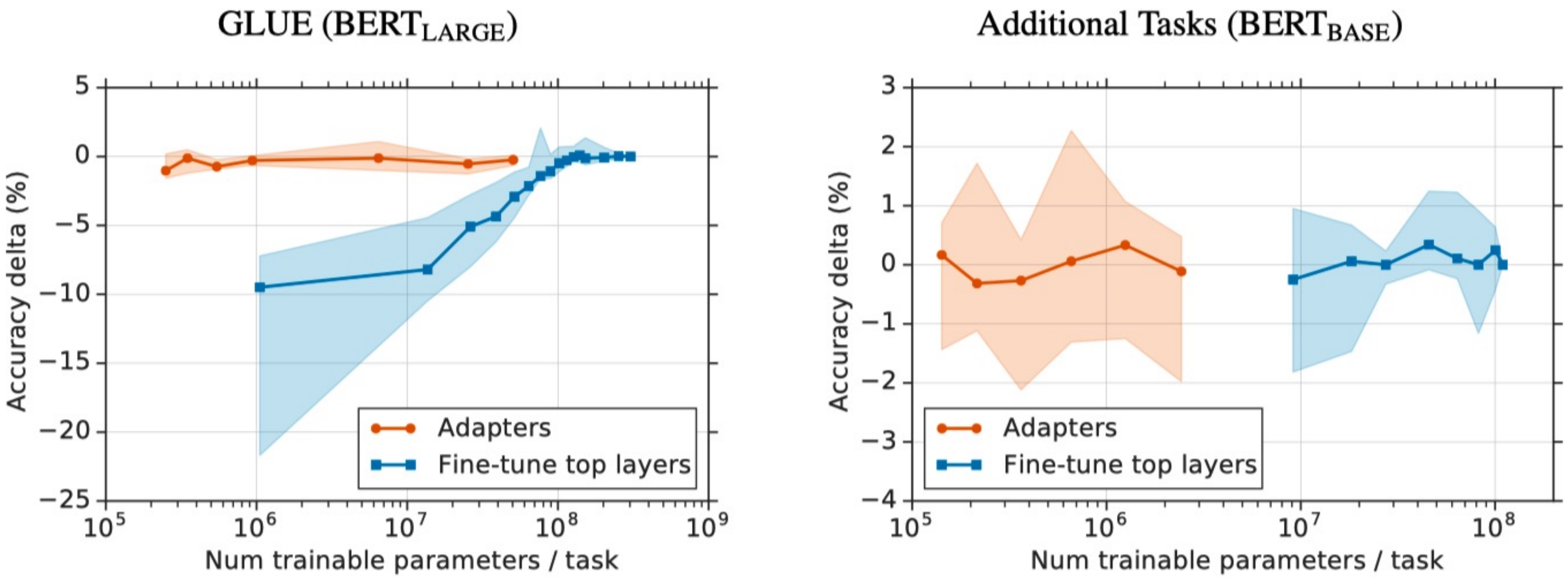


Parameter-Efficient Transfer Learning for NLP. Houlsby et al. ICML 2019.

Parameter-efficient Finetuning

- Parameter-Efficient Transfer Learning for NLP (ICML 2019)

	Total num params	Trained params / task	CoLA	SST	MRPC	STS-B	QQP	MNLI_{m}	MNLI_{mm}	QNLI	RTE	Total
$\text{BERT}_{\text{LARGE}}$	9.0×	100%	60.5	94.9	89.3	87.6	72.1	86.7	85.9	91.1	70.1	80.4
Adapters (8-256)	1.3×	3.6%	59.5	94.0	89.5	86.9	71.8	84.9	85.1	90.7	71.5	80.0
Adapters (64)	1.2×	2.1%	56.9	94.2	89.6	87.3	71.8	85.3	84.6	91.4	68.8	79.6



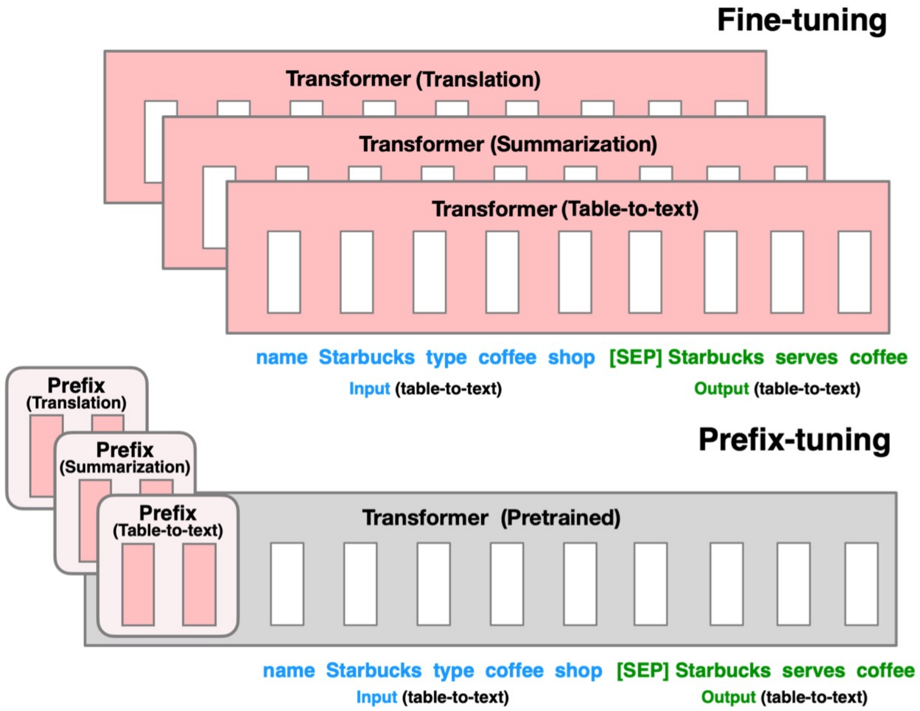
Parameter-Efficient Transfer Learning for NLP. Houlsby et al. ICML 2019.

Parameter-efficient Finetuning

- **Prefix-Tuning**
  - Inspired by prompting: having a proper context can steer the LM without changing its parameters.
  - Optimize a small **continuous** task-specific vector (called the **prefix**).

$$h_i = \begin{cases} P_\theta[i,:], & \text{if } i \in \mathsf{P}_{\text{idx}}, \\ \mathrm{LM}_\phi(z_i, h_{<i}), & \text{otherwise.} \end{cases}$$

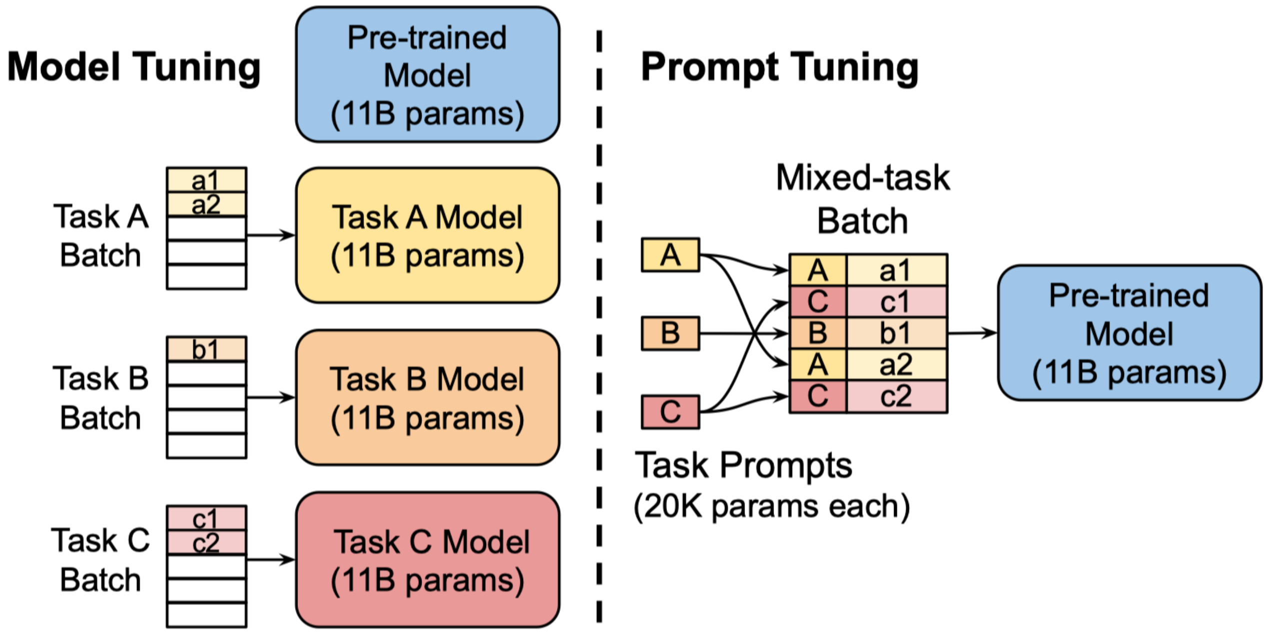
$$P_\theta[i,:] = \mathrm{MLP}_\theta(P'_\theta[i,:])$$



Prefix-Tuning: Optimizing Continuous Prompts for Generation. Li et al. ACL 2021.

Parameter-efficient Finetuning

- **Prompt-Tuning**
  - A simplification of prefix-tuning.
  - Only allow an additional *k* tunable tokens per downstream task to be prepended to the input text.



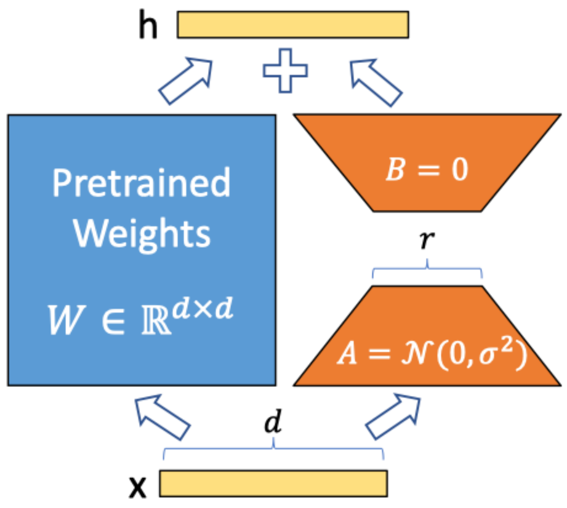
The Power of Scale for Parameter-Efficient Prompt Tuning. Lester et al. EMNLP 2021.

Parameter-efficient Finetuning

- **LoRA: Low-Rank Adaptation of Language Models (2021)**
  - **Hypothesize**: The change in weights during model adaptation has a low "intrinsic rank".
  - Inject trainable rank decomposition matrices into each layer of the PLM.

$$W_0 + \Delta W = W_0 + BA$$

$$W_0 \in \mathbb{R}^{d \times k} \quad A \in \mathbb{R}^{r \times k} \quad B \in \mathbb{R}^{d \times r}$$



  - Only adapt the attention weights (i.e., W_q, W_k, W_v, W_o).
  - No additional inference latency.
  - Cannot put samples of different tasks into the same batch.

LoRA: Low-Rank Adaptation of Large Language Models. Hu et al. 2021.

Parameter-efficient Finetuning

- **Low-Rank Adaptation of Language Models (2021)**

Model & Method	# Trainable Parameters	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
RoB_{base} (FT)*	125.0M	**87.6**	94.8	90.2	**63.6**	92.8	**91.9**	78.7	91.2	86.4
RoB_{base} (BitFit)*	0.1M	84.7	93.7	**92.7**	62.0	91.8	84.0	81.5	90.8	85.2
RoB_{base} (Adpt^{D})*	0.3M	$87.1_{\pm.0}$	$94.2_{\pm.1}$	$88.5_{\pm1.1}$	$60.8_{\pm.4}$	$93.1_{\pm.1}$	$90.2_{\pm.0}$	$71.5_{\pm2.7}$	$89.7_{\pm.3}$	84.4
RoB_{base} (Adpt^{D})*	0.9M	$87.3_{\pm.1}$	$94.7_{\pm.3}$	$88.4_{\pm.1}$	$62.6_{\pm.9}$	$93.0_{\pm.2}$	$90.6_{\pm.0}$	$75.9_{\pm2.2}$	$90.3_{\pm.1}$	85.4
RoB_{base} (LoRA)	0.3M	$87.5_{\pm.3}$	$\mathbf{95.1}_{\pm.2}$	$89.7_{\pm.7}$	$63.4_{\pm1.2}$	$\mathbf{93.3}_{\pm.3}$	$90.8_{\pm.1}$	$\mathbf{86.6}_{\pm.7}$	$\mathbf{91.5}_{\pm.2}$	**87.2**
$\text{RoB}_{\text{large}}$ (FT)*	355.0M	90.2	**96.4**	**90.9**	68.0	94.7	**92.2**	86.6	92.4	88.9
$\text{RoB}_{\text{large}}$ (LoRA)	0.8M	$\mathbf{90.6}_{\pm.2}$	$96.2_{\pm.5}$	$\mathbf{90.9}_{\pm1.2}$	$\mathbf{68.2}_{\pm1.9}$	$\mathbf{94.9}_{\pm.3}$	$91.6_{\pm.1}$	$\mathbf{87.4}_{\pm2.5}$	$\mathbf{92.6}_{\pm.2}$	**89.0**
$\text{RoB}_{\text{large}}$ (Adpt^{P})†	3.0M	$90.2_{\pm.3}$	$96.1_{\pm.3}$	$90.2_{\pm.7}$	$\mathbf{68.3}_{\pm1.0}$	$\mathbf{94.8}_{\pm.2}$	$\mathbf{91.9}_{\pm.1}$	$83.8_{\pm2.9}$	$92.1_{\pm.7}$	88.4
$\text{RoB}_{\text{large}}$ (Adpt^{P})†	0.8M	$\mathbf{90.5}_{\pm.3}$	$\mathbf{96.6}_{\pm.2}$	$89.7_{\pm1.2}$	$67.8_{\pm2.5}$	$\mathbf{94.8}_{\pm.3}$	$91.7_{\pm.2}$	$80.1_{\pm2.9}$	$91.9_{\pm.4}$	87.9
$\text{RoB}_{\text{large}}$ (Adpt^{H})†	6.0M	$89.9_{\pm.5}$	$96.2_{\pm.3}$	$88.7_{\pm2.9}$	$66.5_{\pm4.4}$	$94.7_{\pm.2}$	$92.1_{\pm.1}$	$83.4_{\pm1.1}$	$91.0_{\pm1.7}$	87.8
$\text{RoB}_{\text{large}}$ (Adpt^{H})†	0.8M	$90.3_{\pm.3}$	$96.3_{\pm.5}$	$87.7_{\pm1.7}$	$66.3_{\pm2.0}$	$94.7_{\pm.2}$	$91.5_{\pm.1}$	$72.9_{\pm2.9}$	$91.5_{\pm.5}$	86.4
$\text{RoB}_{\text{large}}$ (LoRA)†	0.8M	$\mathbf{90.6}_{\pm.2}$	$96.2_{\pm.5}$	$\mathbf{90.2}_{\pm1.0}$	$68.2_{\pm1.9}$	$\mathbf{94.8}_{\pm.3}$	$91.6_{\pm.2}$	$\mathbf{85.2}_{\pm1.1}$	$\mathbf{92.3}_{\pm.5}$	**88.6**
DeB_{XXL} (FT)*	1500.0M	91.8	**97.2**	92.0	72.0	**96.0**	92.7	93.9	92.9	91.1
DeB_{XXL} (LoRA)	4.7M	$\mathbf{91.9}_{\pm.2}$	$96.9_{\pm.2}$	$\mathbf{92.6}_{\pm.6}$	$\mathbf{72.4}_{\pm1.1}$	$\mathbf{96.0}_{\pm.1}$	$\mathbf{92.9}_{\pm.1}$	$\mathbf{94.9}_{\pm.4}$	$\mathbf{93.0}_{\pm.2}$	**91.3**

Outline

- Effectiveness
  - Finetuning
  - Domain-adaptive finetuning
- Efficiency
  - Prompt-based learning
  - Parameter-efficient finetuning
- Conclusion

Conclusion

- **Finetuning** requires carefully selected learning rate and output layers.
- **Domain-adaptive finetuning** is usually helpful when applying the PLM to a new downstream domain.
- **Prompt-based learning** is a hot new paradigm with little theoretical analysis. How to design better templates and answers is still an open question.
- **Parameter-efficient finetuning** is also a hot research topic due to its high efficiency and applicability in few-shot settings.

Reference

- How to Fine-Tune BERT for Text Classification? Sun et al. 2019. *arXiv preprint arXiv: 1905.05583*.
- Improving BERT Fine-Tuning via Self-Ensemble and Self-Distillation. Xu et al. 2020. *arXiv preprint arXiv: 2002.10345*.
- Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. Gururangan et al. 2020. In *ACL*.
- Language Models are Few-Shot Learners. Brown et al. 2020. *arXiv preprint arXiv: 2005.14165*.
- Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. Liu et al. 2021. *arXiv preprint arXiv: 2107.13586*.
- Making Pre-trained Language Models Better Few-shot Learners. Gao et al. 2021. In *ACL*.
- Parameter-Efficient Transfer Learning for NLP. Houlsby et al. 2019. In *ICML*.
- Prefix-Tuning: Optimizing Continuous Prompts for Generation. Li et al. 2021. In *ACL*.
- The Power of Scale for Parameter-Efficient Prompt Tuning. Lester et al. 2021. In *EMNLP*.
- LoRA: Low-Rank Adaptation of Large Language Models. Hu et al. 2021. *arXiv preprint arXiv: 2106.09685*.

Thanks

Q&A