EPT: Efficient Prompt Tuning by Multi-Space Projection and Prompt Fusion

Pengxiang Lan[1], Enneng Yang[1], Yuting Liu[1], Guibing Guo[1*], Jianzhe Zhao[1], Xingwei Wang[2*]

[1]Software College, Northeastern University, China
[2]School of Computer Science and Engineering, Northeastern University, China
{pengxianglan, ennengyang, yutingliu}@stumail.neu.edu.cn, {guogb, zhaojz}@swc.neu.edu.cn, wangxw@mail.neu.edu.cn

Abstract

Prompt tuning is a promising method to fine-tune a pre-trained language model without retraining its large-scale parameters. Instead, it attaches a soft prompt to the input text, whereby downstream tasks can be well adapted by merely learning the embeddings of prompt tokens. Nevertheless, existing methods still suffer from two challenges: **(i)** they are hard to balance accuracy and efficiency. A longer (shorter) soft prompt generally leads to a better (worse) accuracy but at the cost of more (less) training time. **(ii)** The performance may not be consistent when adapting to different downstream tasks. We attribute it to the same embedding space but responsible for different requirements of downstream tasks. To address these issues, we propose an **E**fficient **P**rompt **T**uning method (**EPT**) by multi-space projection and prompt fusion. Specifically, it decomposes a given soft prompt into a shorter prompt and two low-rank matrices, significantly reducing the training time. Accuracy is also enhanced by leveraging low-rank matrices and the short prompt as additional knowledge sources to enrich the semantics of the original short prompt. In addition, we project the soft prompt into multiple subspaces to improve the performance consistency, and then adaptively learn the combination weights of different spaces through a gating network. Experiments on 13 natural language processing downstream tasks show that our method significantly and consistently outperforms 11 comparison methods with the relative percentage of improvements up to 12.9%, and training time decreased by 14%.

Introduction

Fine-tuning methods have become a growing focus to adapt a pre-trained language model (PLM) to a variety of downstream tasks (Devlin et al. 2019; Radford et al. 2019). However, the continuous expansion of the PLMs scale has led to a significant increase in the number of parameters (Zhang et al. 2022), such as the T5 model (Raffel et al. 2020) containing hundreds of millions of parameters. Therefore, full fine-tuning PLMs on all parameters is unrealistic in practical applications. The discrete phrase-based tuning provides task descriptions in the form of input text (Brown et al. 2020), guiding PLMs to perform corresponding downstream

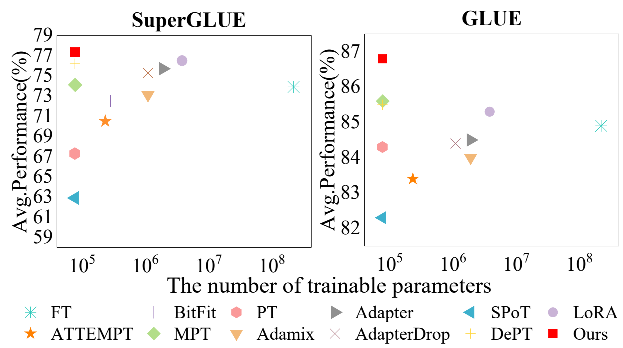

Figure 1: Average performance (y-axis) against the number of trainable parameters (x-axis) on the GLUE and SuperGLUE benchmarks. We utilize the T5-Base for all models.

tasks effectively, avoiding full-parameter fine-tuning. Unfortunately, manually designing an effective set of task prompt phrases heavily relies on experts' domain knowledge, which is still challenging in the face of a wide variety of tasks.

Recently, prompt tuning (PT) (Lester, Al-Rfou, and Constant 2021) based method has become an effective alternative to convert discrete phrases into a set of learnable parameters. PT freezes the parameters of PLMs and only trains the attached soft (continuous) prompt vectors to the input text. Therefore, its parameters do not dramatically scale up with the expansion of the model size, making PT stand out in the parameter-efficient fine-tuning (PEFT) approaches (Shi and Lipani 2024). Recent studies have leveraged some successful approaches to reduce training parameters in PT, such as parameter-efficient transfer learning (PETL) (Vu et al. 2022; Asai et al. 2022), multi-task learning (Wang et al. 2022b), and decomposing soft prompts (Shi and Lipani 2024; Xiao et al. 2023). Despite these PT variants effectively improving soft prompt performance in downstream tasks, PT still faces several limitations that cannot be ignored. **First**, existing PT-based methods encounter the challenge of balancing accuracy and efficiency (Xiao et al. 2023; Lester, Al-Rfou, and Constant 2021; Shi and Lipani 2024). Attaching the soft prompt to the input extends the overall length of the input sequence. Due to the quadratic complexity of the Transformer (Vaswani et al. 2017), lengthening the soft prompt introduces additional training time. PT requires training a

*Guibing Guo and Xingwei Wang are corresponding author.
Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

substantial number of prompt tokens to achieve competitive performance (Lester, Al-Rfou, and Constant 2021); directly shortening the soft prompt to reduce training time may result in sub-optimal performance for PT. **Second**, existing PT-based variants are not well adapted to various downstream tasks and are causing inconsistent performance. This is because they attempt to handle the different needs of various downstream tasks with the same embedding space (Shi and Lipani 2024; Wang et al. 2022b; Asai et al. 2022). However, text information in natural language processing tasks involves different types (Wang et al. 2019) and degrees of difficulty, and models pay limited attention to semantics in the short prompt. For example, on the SuperGLUE (Wang et al. 2019) benchmark, which is more complex than the GLUE (Wang et al. 2018) benchmark, the performance of PT's variants is not very satisfactory.

To tackle the aforementioned knotty issues, we propose a novel efficient prompt tuning (EPT) that consists of two core modules: prompt fusion and multi-space projection. EPT initially decomposes a whole soft prompt into two independent parts: a short prompt and two low-rank matrices. Only the short prompt is attached to the front of the input, to reduce the training time. Low-rank matrices are utilized to update the frozen input text embedding. Next, to offset the semantic loss of the short prompt compared with long ones, we design a prompt fusion module. This module utilizes the attention network by Einstein Summation to capture the knowledge difference between low-rank matrices and the short prompt, and instills this difference into the short prompt to improve the semantic richness of the short prompt. Then, to adapt PT to different downstream tasks more consistently, we leverage a multi-space projection module to project a single soft prompt into multiple subspaces and reweight the soft prompt in these subspaces according to the task through the gating network. Finally, a joint representation of the prompt (obtained from the prompt fusion and multi-space modules) replaces the vanilla prompt.

Contributions. In summary, the main contributions of this paper are as follows:

- We point out that PT-based methods suffer from the trade-off dilemma of "accuracy and efficiency" as well as performance inconsistency. To address these issues, we propose a novel efficient prompt tuning (EPT) method.
- We design two effective modules in EPT, prompt fusion and prompt projection. The former helps to maintain the efficiency of the short prompt and compensate for the semantic missing of the short prompt to enhance performance, and the latter reweights prompts in multiple subspaces to adapt to downstream tasks.
- We comprehensively evaluate EPT on the GLUE and SuperGLUE benchmarks, where EPT outperformed other PEFT methods, including LoRA and multi-task transfer learning-based PT variants (see Figure. 1). In particular, EPT achieves a 14% reduction in training time compared to vanilla PT on the GLUE benchmark.

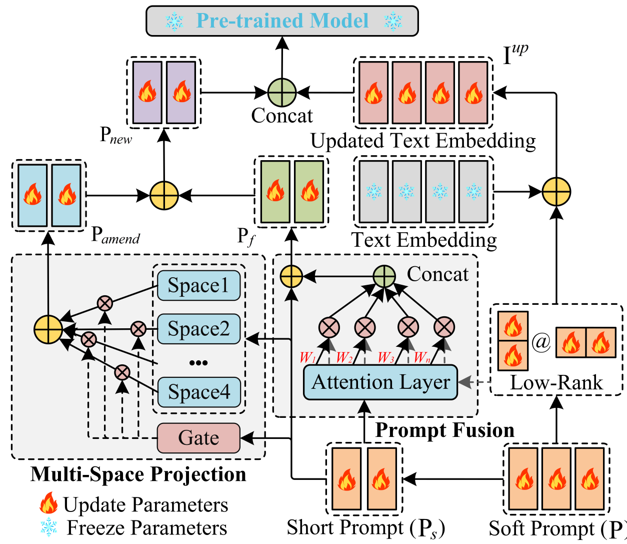

Figure 2: The overview of the EPT model. The whole soft prompt is decomposed into a short prompt and two low-rank matrices. Low-rank matrices are multiplied and added element-wise to the frozen input text embedding. The Multi-Space Projection Module maps the short prompt to multiple subspaces, addressing diverse downstream task requirements, while the Prompt Fusion module enhances its semantic knowledge. Finally, EPT generates a joint prompt representation to supersede the original prompt. The new prompt and the updated input text embedding are concatenated to input into the PLM.

The Proposed Method

In this section, we first introduce the background of the prompt tuning and then elaborate our proposed EPT method as shown in Figure. 2. It consists of four main modules: (1) Prompt Decomposition, (2) Prompt Fusion, (3) Multi-Space Projection, and (4) Reconstructed Prompt.

Background: Prompt Tuning

We first introduce the training method of PT. PT has gained widespread adoption in downstream tasks due to its advantage of the parameters not increasing sharply with the expansion of the model (Shi and Lipani 2024). Let labeled training data $(\boldsymbol{X}, \boldsymbol{Y}) = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$ for one target task $\mathcal{T}$, where N is the number of training data. Given a PLM with parameters Θ and each input text x_i. The embedding of $x_i \in \boldsymbol{X}$ is represented as $\mathbf{E}_i \in \mathbb{R}^{m \times d}$, where m is maximum sequence length and d is the hidden dimension of input text embedding. $\mathbf{P} \in \mathbb{R}^{l \times d}$ is initialized to form a target prompt, l is a hyper-parameter for the length of the soft prompt. It is concatenated with $\mathbf{E}_i \in \mathbb{R}^{m \times d}$, which does not involve gradient updates during training, to form a new input embedding $[\mathbf{P}; \mathbf{E}_i] \in \mathbb{R}^{(l+m) \times d}$. The target task is formulated as follows:

$$\mathcal{L}_{PT} = -\sum_i \log P\left(\mathbf{y}_i | \left[\mathbf{P}; \mathbf{E}_i\right]; \Theta\right) \quad (1)$$

where $\mathcal{L}_{PT}$ is a loss function only optimized with the prompt $\mathbf{P}$. However, the vanilla PT requires training a large number of prompt tokens (i.e., a larger value of l in $\mathbf{P}$) to achieve the expected performance (Lester, Al-Rfou, and Constant 2021; Razdaibiedina et al. 2023).

Prompt Decomposition

Most studies have shown that the performance of PT is comparable to full fine-tuning (Razdaibiedina et al. 2023; Wang et al. 2022b). However, a challenging issue persists: PT requires training a substantial number of prompt tokens to achieve competitive performance, resulting in an increased length of the entire input sequence (Lester, Al-Rfou, and Constant 2021). It causes greater resource consumption in the training/inference phase. We begin by initializing our source prompt $\mathbf{P} \in \mathbb{R}^{l \times d}$ from sampled vocabulary (e.g., the 5000 most common tokens) to ensure that $\mathbf{P}$ is informative content. Inspired by DEPT (Shi and Lipani 2024), we truncate a trainable short prompt $\mathbf{P}_s \in \mathbb{R}^{s \times d}$ with a length of s ($s < l$) from $\mathbf{P}$. Subsequently, we align the dimensions of $\mathbf{P} \in \mathbb{R}^{l \times d}$ with $\mathbf{E} \in \mathbb{R}^{m \times d}$ and then perform Singular Value Decomposition (SVD), retaining the top r two trainable low-rank singular vector matrices ($\mathbf{A} \in \mathbb{R}^{m \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times d}$). Among them, r is the rank in low-rank matrices and $r \ll \min(m, d)$, d is the dimension of input text embedding, m is the maximum sequence length. Due to the transformer's quadratic complexity, the training duration is proportional to the length of the prompt. Therefore, a shorter prompt $\mathbf{P}_s$ can effectively reduce the training time. Notably, unlike the DEPT, its $\mathbf{A}$ and $\mathbf{B}$ are random Gaussian initialization and zero initialization respectively (follow LoRA (Hu et al. 2021)). This operation of randomly initializing results in a complete loss of information about the original longer prompt, $\mathbf{P}$, since it is semantically rich. Therefore, in our approach, $\mathbf{A}$ and $\mathbf{B}$ are obtained by decomposing of $\mathbf{P}$ to preserve the semantic knowledge of original prompt $\mathbf{P}$ as much as possible.

To keep the same amount of trainable parameters, the selection of s and r satisfies the equation $l \times d = s \times d + (m + d) \times r$, where s and r are hyper-parameters and $s < l$ when $r > 0$. For the decomposition of the vanilla PT, the specific values of s and r affect each other. For example, in the T5-base, d (dimension) is 768. If l is 100 and m is 256, when the length of $\mathbf{P}_s$ is 60, r is 30 ($60 \times 768 + (256 + 768) \times 30$). When the length of $\mathbf{P}_s$ is 40, r is 45 ($40 \times 768 + (256 + 768) \times 45$). When $r = 0$, $s = l$, the decomposed PT proposed in this paper degenerates to vanilla PT. The purpose of the low-rank matrices is to update the frozen input word embedding. When $s = 0$, only low-rank matrices are used to update the frozen input word embedding:

$$\mathbf{I}_i^{up} = \mathbf{E}_i + \mathbf{A} \otimes \mathbf{B} \tag{2}$$

where $\mathbf{A} \otimes \mathbf{B}$ represents the multiplication operation of $\mathbf{A}$ and $\mathbf{B}$, $\mathbf{I}_i^{up} \in \mathbb{R}^{m \times d}$ represents the result of adding $\mathbf{A} \otimes \mathbf{B}$ to the frozen input text embedding $\mathbf{E}_i$.

Prompt Fusion

In this section, we design a novel prompt fusion module to keep the short prompt efficiency and further compensate for the semantic loss of the decomposition of the long prompt into a short prompt and two low-rank matrices in the previous section. Specifically, supposing the short prompt $\mathbf{P}_s$ is directly injected into PLMs (the vanilla prompt has the same operation). In that case, although shortening the length of the soft prompt reduces the training time, this still will lead to poor performance of PT because of the lack of knowledge of the original prompt $\mathbf{P}$. PT requires a substantial number of prompt tokens (exceeding 100) to achieve optimal performance (Lester, Al-Rfou, and Constant 2021). Therefore, enriching the knowledge of $\mathbf{P}_s$ becomes exceptionally crucial while reducing the training time.

Building upon this foundation, we first leverage an attention network by Einstein Summation to consider the difference in knowledge richness between low-rank matrices and the short prompt. Then, we add the short prompt with the output of the attention network to enhance the knowledge of the original short prompt:

$$\mathbf{W}_{attn} = \text{softmax}(\frac{1}{\sqrt{d}}\mathbf{P}_s \cdot (\mathbf{A} \otimes \mathbf{B})^\top) \tag{3}$$

$$\mathbf{P}_f = \mathbf{P}_s + Ein(\mathbf{W}_{attn} \cdot \mathbf{P}_s) \tag{4}$$

where $(\mathbf{A} \otimes \mathbf{B})^\top$ is the transpose of $\mathbf{A} \otimes \mathbf{B}$, $\mathbf{W}_{attn}$ is the weighted vector representation, and $Ein(\cdot)$ is the Einstein Summation (the way the dimensions change is $'bpl, bpd \rightarrow bpd'$). The attention mechanism $\mathbf{W}_{attn} \cdot \mathbf{P}_s$ considers the knowledge association between low-rank matrices and $\mathbf{P}_s$. $\mathbf{P}_f \in \mathbb{R}^{m \times d}$ enhances the knowledge within the original short prompt based on reducing the consumption of computing resources.

Multi-Space Projection

In this section, we propose the multi-space projection module to project a single prompt into multiple subspaces to solve the performance inconsistency problem of the original PT only fine-tuning in a single space, which reweights the prompt representations in different spaces through a gating network at each downstream task. Text information in text classification tasks usually involves different types and degrees of difficulty (such as Natural Language Inference, Question Answering, etc.). However, PT is inputted into PLMs in the same embedding space to adapt to downstream tasks, and a single space does not consider the different requirements in downstream tasks. This results in potentially inconsistent performance of PT - as it performs well on some tasks and poorly on others. The Mixture-of-Experts (Jacobs et al. 1991) provides an excellent idea to solve the aforementioned problem. Motivated by this, we map $\mathbf{P}_s$ to distinct spaces and utilize a gating network to control each space's weight distribution. Prompt tokens are assigned different degree weights by achieving the parameter selection:

$$E_i(\mathbf{P}_s) = \text{linear}_1\left(\sigma\left(\text{linear}_2\left(\mathbf{P}_s\right)\right)\right), i \in [1, ..., N_e] \tag{5}$$

where $E_i(\mathbf{P}_s) \in \mathbb{R}^{s \times d}$ is the i-th space, $\text{linear}_1 \in \mathbb{R}^{m \times d}$, $\text{linear}_2 \in \mathbb{R}^{d \times m}$, N_e is the maximum number of spaces, the activation function $\sigma(\cdot)$ is a ReLU (Krizhevsky, Sutskever, and Hinton 2012) function. The gate network is formulated as follows:

$$f_i(\mathbf{P}_s) = \text{linear}(\mathbf{P}_s), i \in [1, ..., N_e]$$
$$G_i(\mathbf{P}_s) = \frac{\exp^{f_i(\mathbf{P}_s)}}{\sum_{i=1}^{N_e} \exp^{f_i(\mathbf{P}_s)}} \tag{6}$$

where $G_i(\mathbf{P}_s) \in \mathbb{R}^{s\times 1}$ is used to control the importance of each space, linear $\in \mathbb{R}^{d\times 1}$. Reweighting each space by leveraging a gating mechanism:

$$\mathbf{P}_{amend} = \sum_{i=1}^{N_e} G_i(\mathbf{P}_s) \cdot E_i(\mathbf{P}_s) \tag{7}$$

where $\mathbf{P}_{amend} \in \mathbb{R}^{s\times d}$ is the result of reweighting $\mathbf{P}_s$. $G_i(\mathbf{P}_s)$ makes one or more spaces in an active state better for different parameter selections.

Reconstructed Prompt

In this section, our EPT method integrates prompt representations of the fusion module and the multi-space module to obtain a joint representation to have both advantages. To be specific, we learn the joint representation $\mathbf{P}_{new}$ of $\mathbf{P}_{amend}$ and $\mathbf{P}_f$. Weights of $\mathbf{P}_{amend}$ are allocated in different spaces, and the soft prompt $\mathbf{P}_f$ in the prompt fusion module. The purpose of learning a joint representation of soft prompts is to replace the original prompt $\mathbf{P}$ with $\mathbf{P}_{new}$:

$$\mathbf{P}_{new} = \mathbf{P}_{amend} + \mathbf{P}_f \tag{8}$$

when the initialized $\mathbf{P}_s$ performs poorly on specific tasks, $\mathbf{P}_{amend}$ and $\mathbf{P}_f$ redistribute the importance of $\mathbf{P}_s$. After learning $\mathbf{P}_{new}$, the constructed network is discarded, and $\mathbf{P}_{new}$ is utilized for training in the PLM. Therefore, the trainable parameters input into the PLMs will remain consistent with the original PT. By $\mathbf{P}_{new}$ and $\mathbf{I}_i^{up}$, Eq.(1) is displaced by:

$$\mathcal{L}_{PT} = -\sum_i \log P(\mathbf{y}_i | [\mathbf{P}_{new}; \mathbf{I}_i^{up}]; \mathbf{P}_{new}) \tag{9}$$

where $[\mathbf{P}_{new}; \mathbf{I}_i^{up}]$ is a input embedding of PLMs through the connection of $\mathbf{P}_{new}$ and $\mathbf{I}_i^{up}$.

Quantization

To reduce GPU memory usage, we employed quantization techniques (Dettmers et al. 2021, 2023) for models with a size of 3B or larger. This process involves rescaling the input tensors by loading the model in 4-bit precision and back-quantizing the values to bf16 during training. We minimize storage consumption by implementing the double quantization method proposed in QLoRA (Dettmers et al. 2023), which approach significantly reduces memory usage while maintaining performance comparable to standard parameter-efficient fine-tuning. Notably, weight gradients are still calculated exclusively on the soft prompt parameters.

Experiments

We conduct extensive experiments to answer these key research questions: **RQ1:** How does EPT compare with state-of-the-art baselines across different datasets? **RQ2:** How do we understand the impact of the critical components of EPT and model scaling on the performance of EPT? **RQ3:** How do the few-shot adaptability and hyper-parameter tuning affect the performance of EPT?

Evaluation Datasets and Source Tasks

We conducted multi-angle experiments on the EPT method to demonstrate its outstanding applicability to 13 publicly available NLP tasks (8 from the GLUE benchmark [1] and 5 from the SuperGLUE benchmark [2]). Specifically, **(1) GLUE** (Wang et al. 2018) is a benchmark for evaluating natural language understanding performance. It consists of diverse tasks that test the model's ability to understand language in different contexts. To fully prove the performance effect of EPT, we maintain consistency with previous work, and the NLP datasets are MNLI (Williams, Nangia, and Bowman 2018), QQP (Wang et al. 2018), QNLI (Rajpurkar et al. 2016), SST-2 (Socher et al. 2013), STS-B (Cer et al. 2017), MRPC (Dolan and Brockett 2005), RTE (Giampiccolo et al. 2007) and CoLA (Warstadt, Singh, and Bowman 2019) from GLUE. **(2) SuperGLUE** (Wang et al. 2019) is an extension of GLUE, that includes more complex and challenging tasks. This paper uses five tasks from SuperGLUE: MultiRC (Khashabi et al. 2018), BoolQ (Clark et al. 2019), WiC (Pilehvar and Camacho-Collados 2019), WSC (Levesque, Davis, and Morgenstern 2012) and CB (De Marneffe, Simons, and Tonhauser 2019). We follow the previous working setup (Su et al. 2022; Asai et al. 2022; Shi and Lipani 2024), which only utilizes ReCoRD (Zhang et al. 2018) and SQuAD (Rajpurkar et al. 2016) in the few-shot experiment.

Baselines

We focus on exploring a high-performance and less training parameter method of PEFT, so the number of training parameters is also an essential factor. Methods such as KronA (Edalati et al. 2022), S4 (Chen et al. 2022), etc. have more training parameters, for example, the training parameter of PT is 0.1% of full fine-tuning, while the training parameter of MAM adapter (He et al. 2021) is 6.7% of full fine-tuning. Therefore, we focus more on the latest methods of PT-type in the baseline selection.

The baselines for comparison with EPT are: **(1) Full Fine-tuning (FT)**, which updates all parameters of PLMs. **(2) PEFT approaches**, including Adapter (Houlsby et al. 2019), AdapterDrop (Rücklé et al. 2021), AdaMix (Wang et al. 2022a), BitFit (Zaken, Goldberg, and Ravfogel 2022), and LoRA (Hu et al. 2021). **(3) PT-based method**, where the vanilla PT (Lester, Al-Rfou, and Constant 2021) updates parameters with prompt prefix to accommodate various downstream tasks, and its variants include SPoT (Vu et al. 2022), ATTEMPT (Asai et al. 2022), MPT (Wang et al. 2022b), and their transfer and multi-task learning variants. SPoT and ATTEMPT find optimal prompt initializations by pre-training prompts on informative source tasks. **(4) Prompt decomposition**, DEPT (Shi and Lipani 2024)

[1] https://huggingface.co/datasets/glue

[2] https://huggingface.co/datasets/super_glue

Model	Param	GLUE									SuperGLUE					
		MNLI (393K)	QQP (364K)	QNLI (105K)	MRPC (3.7K)	STS-B (7K)	SST-2 (67K)	CoLA (8.5K)	RTE (2.5K)	Mean (%)	Multi (5.1K)	WiC (6K)	WSC (554)	BoolQ (9.4K)	CB (250)	Mean (%)
Fine-tuning[1]	220M	**86.8**	**91.6**	93.0	90.2	89.7	**94.6**	61.8	71.9	84.9	72.8	**70.2**	59.6	81.1	85.7	73.9
LoRA[2]	3.8M	86.3	89.0	93.2	90.1	90.9	94.3	63.3	75.5	85.3	72.6	68.3	67.3	81.3	**92.9**	76.5
Adapter[1]	1.9M	86.5	90.2	93.2	85.3	90.7	93.8	64.0	71.9	84.5	**75.9**	67.1	67.3	**82.5**	85.7	75.7
Adamix	1.9M	86.4	90.1	93.0	87.4	91.0	93.9	59.2	70.8	84.0	73.1	66.8	59.3	80.6	85.7	73.1
AdapterDrop[1]	1.1M	86.3	90.2	93.2	86.3	**91.4**	93.6	62.7	71.2	84.4	72.9	68.3	67.3	82.3	85.7	75.3
BitFit[1]	280K	85.3	90.1	93.0	86.8	90.9	94.2	58.2	67.6	83.3	74.5	70.0	59.6	79.6	78.6	72.5
PT	76.8K	83.6	90.3	93.1	87.7	90.2	93.6	59.5	76.2	84.3	67.3	60.5	59.6	70.7	78.6	67.3
ATTEMPT★[1]	232K	84.3	90.3	93.0	85.7	89.7	93.2	57.4	73.4	83.4	74.4	66.8	53.8	78.8	78.6	70.5
MPT★[3]	77.6K	85.9	90.3	93.1	89.1	90.4	93.8	62.4	79.4	85.6	74.8	69.0	67.3	79.6	79.8	74.1
SPoT★[1]	76.8K	85.4	90.1	93.0	79.7	90.0	93.4	57.1	69.8	82.3	74.0	67.0	50.0	77.2	46.4	62.9
DEPT	76.8K	85.1	90.4	**93.3**	89.2	91.0	94.2	62.7	78.4	85.5	74.4	67.1	67.3	79.4	**92.9**	76.2
DPT	9.0K	85.4	90.2	93.1	**90.4**	90.3	94.5	57.8	79.0	85.1	74.0	68.5	67.3	79.4	78.6	73.6
EPT (ours)	76.8K	85.8	90.3	93.2	90.2	91.1	94.5	**67.0**	82.0	**86.8**	74.8	69.0	**69.2**	81.5	**92.9**	**77.5**
ATTEMPT◊★[3]	96.0K	83.7	90.1	93.2	87.3	90.8	94.3	64.3	**82.7**	85.8	74.4	66.5	**69.2**	78.5	82.1	74.1
MPT◊★[3]	10.5K	84.3	90.0	93.0	89.2	90.4	93.3	63.5	**82.7**	85.8	74.8	**70.2**	67.3	79.2	89.3	76.1

Table 1: Performance comparison on GLUE and SuperGLUE benchmark, all experimental results are based on the T5-Base model. The evaluation metrics are Pearson correlation for STS-B, F1 for MultiRC (Multi) and accuracy for other tasks. "Param" represents the amount of trainable parameters for each task. Where ★ indicates that some tasks utilize the PETL method, ◊ indicates that some tasks utilize multi-task learning (resulting in the reduction of trainable parameters). [1] sourced from (Asai et al. 2022). [2] sourced from (Sung, Cho, and Bansal 2022). [3] sourced from (Wang et al. 2022b). The best result is marked in bold. The second-best result is marked with an underline. The numbers under datasets refer to training examples in each dataset.

and DPT (Xiao et al. 2023) are parameter-efficient method that decomposes the soft prompt. DPT effectively reduces the trainable parameters of PT.

Training Detail Settings

Implementation details The main experiments of EPT and baseline are performed using the T5-Base model (Shi and Lipani 2024), which has a parameter size of 220M and the hidden size d is 768. Consistent with the experimental setup of DEPT, we decompose the vanilla prompt (parameter size is 76,800) with the length of prompt tokens of 100. We train for 30,000 steps on small datasets with less than 100k training examples and 300,000 steps on large-size data with more than 100k examples. The batch size is 16 and the number of spaces is 4. For soft prompts, we search for learning rate within the set {3e-1, 4e-1, 5e-1}; for the low-rank matrices, we search for learning rate within the set {1e-04, 5e-4, 5e-03}. Following DEPT (Shi and Lipani 2024), we utilize five source tasks - MNLI, QQP, SST-2, SQuAD, and ReCoRD - for the few-shot experiments. We derive our soft prompt from one of these selected source tasks to initialize our soft prompt and low-rank matrices.

Models Our models for evaluating EPT performance are T5-Base (220M), T5-3B, T5-11B and Llama2-7B (Touvron et al. 2023). In this context, we employed quantization techniques when using T5-3B, T5-11B and Llama2-7B. Notably, PT demonstrates suboptimal performance on smaller models, exhibiting significant sensitivity to hyperparameter configurations (Vu et al. 2022). Consequently, our primary experimental analysis centers on the T5-Base model.

Overall Performance Comparison (RQ1)

Overall, Table 1 shows the results of EPT and other baselines on the GLUE and SuperGLUE benchmarks. Overall, EPT utilizes only a tiny number of trainable parameters yet consistently delivers exceptional performance across various downstream tasks. It surpasses 11 other PEFT methods in average performance on two benchmarks, including PT variants based on multitasking and transfer learning. The visualized results of baselines are shown in Figure. 1.

Among all baselines, although the full fine-tuning performs best in some datasets (MNLI, QQP, SST-2, and SuperGLUE_Wic), the number of parameters required for training is 2,904 times that EPT, making full fine-tuning undoubtedly very computationally resource intensive. SPoT, DEPT, and EPT perform better while keeping the same training parameters as the original PT. This proves that randomly sampled tokens from the vocabulary for initialization and then directly injecting them into PLMs cannot make PT well adaptable to different downstream tasks. EPT and DEPT also utilize decomposing the soft prompt to reduce computing resources. Additionally, compared to the baseline MPT and ATTEMPT, the best-performing transfer learning methods, EPT performs better. EPT does not require additional pretraining source tasks and trains fewer parameters.

Unlike SPoT and ATTEMPT, EPT has consistent performance in downstream tasks with different requirements, whereas they all utilize the attention mechanism. Additionally, SPoT and ATTEMPT only consider the relationship between source prompts of different tasks. EPT enhances the short prompt's semantic knowledge through the prompt fusion module and improves its adaptability to downstream

Prompt Decomposition	Prompt Fusion	Multi-Space Projection	GLUE (%)	Super-GLUE (%)
✗	✗	✗	84.3	67.3
✓	✗	✗	85.8	76.3
✓	✗	✓	86.4	77.1
✓	✓	✗	86.5	76.8
✓	✓	✓	86.8	77.5

Table 2: Performance comparison on the critical components of EPT on GLUE and SuperGLUE benchmarks.

tasks with different requirements by reweighting the short prompt in the multi-space projection module, which is why it performs better than EPT et al. Full fine-tuning performs best in some datasets, such as MNLI and QQP. We analyze that EPT is more efficient in datasets with fewer training samples. Overall, in the GLUE benchmark, our optimal baseline DEPT is only 0.3% higher than MPT in single-task setting. DEPT is only 0.5% better than MPT on multi-task setting on the SuperGLUE benchmark. On the contrary, on the GLUE benchmark, our proposed EPT outperforms DEPT by 1.5% and vanilla PT by 2.9%. On the SuperGLUE benchmark, EPT outperforms DEPT by a relative 1.7% and vanilla PT by a relative 15.2%. Therefore, while training time decreased by 14%, the degree of performance improvement is already very noticeable.

Ablation Experiment Analysis (RQ2)

Analysis the Critical Components of EPT To verify the contribution of each critical component (Prompt Decomposition, Prompt Fusion, and Multi-Space Projection) in EPT. We divided EPT into five different variants for ablation experiments, as shown in Table 2. Overall, the result of EPT considering all critical components (i.e., the last line) is the most outstanding. The lack of any critical component in EPT significantly reduces performance, proving that each critical component positively impacts EPT. When not considering all critical components (i.e., the first line), EPT is a vanilla PT. When using the prompt fusion or multi-space projection module, EPT is superior to only performing the prompt decomposition. This again proves the effectiveness of the

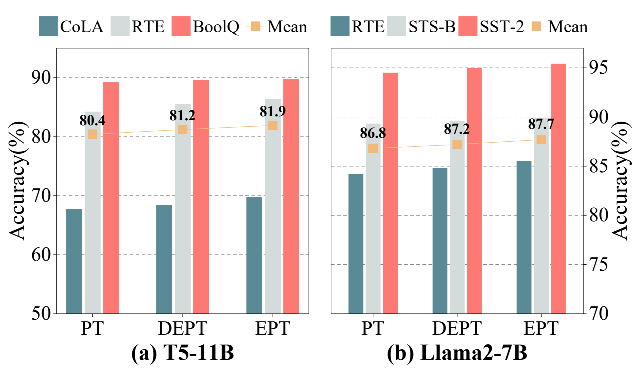

Figure 3: The performance changes of EPT(Ours), DEPT, and PT at different datasets on the T5-11B and Llama2-7B.

Model		PT	DEPT	EPT
BoolQ	(9.4K)	87.0	87.8	87.9
CoLA	(8.5K)	66.1	67.8	68.2
WiC	(6.0K)	70.5	71.2	73.7
MultiRC	(5.1K)	78.0	80.5	80.8
MRPC	(3.7K)	90.7	91.7	92.2
RTE	(2.5K)	82.7	84.2	85.6
WSC	(554)	67.3	67.3	69.2
CB	(250)	75.0	94.6	94.6
Mean	(%)	77.2	80.6	81.5

Table 3: Performance comparison of PT, DEPT and EPT on different datasets for T5-3B.

prompt fusion and multi-space projection module.

Power of Model Scale We conducted an empirical analysis of the impact of model size on performance using different datasets, as detailed in Table 3 (T5-3B) and Figure 3 (T5-11B and Llama2-7B). We choose baselines initialized from a sampled vocabulary for comparison. As illustrated in Table 3 and Figure. 3, EPT outperforms other baselines across various datasets, with an average performance increase of 5.6% on T5-3B compared to the original PT; this advantage persists even in larger models (T5-11B and Llama2-7B). Notably, all methods perform well in larger model scales, resulting in less pronounced performance differences, aligning with previous research findings (Lester, Al-Rfou, and Constant 2021). EPT is also capable of adapting to various downstream tasks in different model architectures.

Indepth Analysis (RQ3)

Few-shot adaptation Following previous work (Asai et al. 2022; Wang et al. 2022b; Shi and Lipani 2024), we pretrained the soft prompt and the low-rank matrices on source tasks. We evaluate the performance of EPT, vanilla PT, and MPT in k-shot (k = 4, 16, 32) on the GLUE benchmark. As shown in Figure. 4(a), the performance improvement of EPT is mainly due to using the PETL framework for pre-training source prompts. EPT outperforms other variants of PT under few-shot learning tasks, which proves its effectiveness.

The Length of Soft Prompt For the EPT method, we maintained the same number of trainable parameters (76,800) as the conventional PT with a length of 100, and compared the training time costs between EPT and PT. Figure. 4(b) shows that EPT takes more training time as the length of the short prompt increases. When the length of the short prompt is 60, EPT has the best performance on the GLUE benchmark, and the training time of EPT is 14% lower than that of the original PT. On the GLUE benchmark, EPT significantly outperforms DEPT and PT at different prompt's lengths (except for length 0). When the length is 0, the source prompt is only decomposed into two low-rank matrices, rendering the prompt fusion and multi-space projection modules in EPT non-functional. Consequently, EPT and DEPT exhibit identical performance. Additionally, the parameters of vanilla PT are frozen and not updated, resulting in no performance outcomes. When the soft prompt

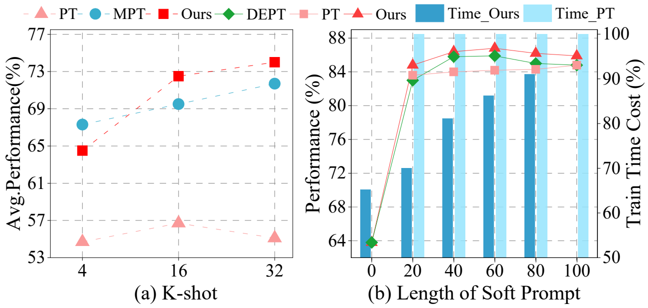

Figure 4: On the GLUE benchmark, (a) The performance changes of EPT(Ours), MPT, and PT at different K-shot. (b) Comparison of training time consumption and the performance changes (EPT, DEPT, and PT) according to different lengths of the short prompt in EPT and DEPT.

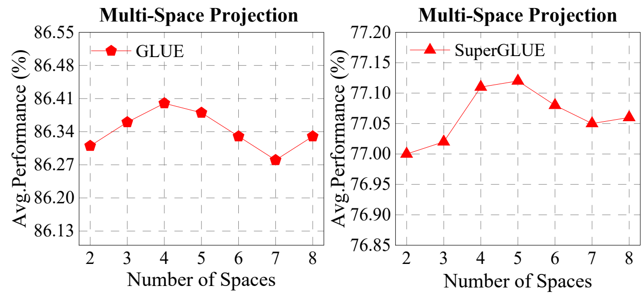

Figure 5: Performance of the number of spaces in the Multi-Space Projection module on the GLUE and SuperGLUE benchmarks.

length is 100, DEPT is conventional PT, and EPT outperforms DEPT as the short prompts are mapped to different subspaces to reweight the prompt tokens, positively influencing EPT. This demonstrates that conventional PT struggles to adapt to downstream tasks with varying requirements through fine-tuning in the same single embedding space.

The Impact of the Number of Spaces To eliminate the noise generated by the prompt fusion module, when analyzing the impact of changes in the number of spaces on performance, we only leverage a multi-space projection module that learns the reweighted short prompt. As shown in Figure. 5, we dynamically alter the number of spaces N from 2 to 8 with a step size of 1 during training. Overall, there are many datasets in both the GLUE and SuperGLUE benchmarks, so the fluctuations in EPT on the two benchmarks are small, and the number of spaces we comprehensively selected is 4.

Related Works

Parameter-efficient Fine-tuning

Parameter-efficient fine-tuning approaches can adapt well to various downstream tasks by updating a limited number of training parameters compared to full fine-tuning. AdapterDrop (Rücklé et al. 2021) dynamically dropping the Adapter reduces the number of model parameters as much as possible and improves the efficiency of model training/inference. Diff pruning (Guo, Rush, and Kim 2021) learns a task-specific "diff" vector that extends the original pre-trained parameters. LoRA (Hu et al. 2021) only updates the parameters of low-rank matrix pairs. BitFit (Zaken, Goldberg, and Ravfogel 2022) only updates the mask layer parameters of PLMs. HyperDecoder (Ivison and Peters 2022) efficient adaptation of parameters for decoder generation using a hyper-network conditioned on encoder output in multi-task. LST (Sung, Cho, and Bansal 2022) aims to reduce the training memory by a ladder-side network for transformers. Prompt tuning (PT) is a promising parameter-efficient fine-tuning (PEFT) approach, as its parameters do not exhibit dramatic growth even when the model size expands significantly.

PT-based Methods

The expansion in PLMs size does not lead to a surge in the training parameters of PT. The recent research aims to improve the performance of PT through various approaches. SPoT (Vu et al. 2022) learns one or more source prompts, constructing the interaction with the target task to initialize the target prompt. ATTEMPT (Asai et al. 2022) considers the impact of knowledge in the source prompts on the input sequence to generate different attention weights, achieving weighting target prompts. MPT (Wang et al. 2022b) decomposes each source prompt into a one-rank matrix, performs Hadamard product with shared prompts to construct student prompts, and then improves the performance of PT through knowledge distillation. DPT (Xiao et al. 2023) initializes a soft prompt to reduce the number of trainable parameters by utilizing two low rank vectors instead of soft prompt. These variants, which are built upon soft prompts, have exhibited remarkable performance. However, these PT-based methods still struggle to balance efficiency and accuracy. Moreover, they typically work in a single space, thus resulting in performance inconsistencies across different downstream tasks.

Conclusions and Future Work

In this work, we propose an efficient soft prompt tuning (EPT) method by prompt fusion and multi-space projection. Specifically, the prompt fusion module can help enhance the semantic of the soft prompt, leading to a balance between accuracy and efficiency. The multi-space module projects a single soft prompt into multiple subspaces with reweighted prompt tokens, improving the performance consistency. Experimental results across two model architectures (T5 and Llama2) demonstrate that EPT reduces training time, achieves optimal and consistent performance using the shorter soft prompt, and validates the effectiveness of critical components in EPT.

For future work, we will address the computational overhead introduced by using two learning rates in EPT for parameter search. Furthermore, we intend to explore the integration of EPT with soft prompt methods based on multitask transfer learning, aiming to reduce training parameters further while maintaining optimal performance.

Acknowledgements

This work is partially supported by the National Natural Science Foundation of China under Grant (No. 62032013, 62102074), the Science and Technology projects in Liaoning Province (No. 2023JH3/10200005).

References

Asai, A.; Salehi, M.; Peters, M. E.; and Hajishirzi, H. 2022. Attempt: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *EMNLP*, 6655–6672.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901.

Cer, D.; Diab, M.; Agirre, E. E.; Lopez-Gazpio, I.; and Specia, L. 2017. SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Cross-lingual Focused Evaluation. In *SemEval*, 1–14.

Chen, J.; Zhang, A.; Shi, X.; Li, M.; Smola, A.; and Yang, D. 2022. Parameter-Efficient Fine-Tuning Design Spaces. In *ICLR*.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *NAACL*, 2924–2936.

De Marneffe, M.-C.; Simons, M.; and Tonhauser, J. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *SuB*, volume 23, 107–124.

Dettmers, T.; Lewis, M.; Shleifer, S.; and Zettlemoyer, L. 2021. 8-bit Optimizers via Block-wise Quantization. In *ICLR*.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLORA: efficient finetuning of quantized LLMs. In *NeurIPS*, 10088–10115.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Dolan, B.; and Brockett, C. 2005. Automatically constructing a corpus of sentential paraphrases. In *IWP*.

Edalati, A.; Tahaei, M.; Kobyzev, I.; Nia, V. P.; Clark, J. J.; and Rezagholizadeh, M. 2022. Krona: Parameter efficient tuning with kronecker adapter. *arXiv preprint arXiv:2212.10650*.

Giampiccolo, D.; Magnini, B.; Dagan, I.; and Dolan, W. B. 2007. The third pascal recognizing textual entailment challenge. In *ACL*, 1–9.

Guo, D.; Rush, A. M.; and Kim, Y. 2021. Parameter-Efficient Transfer Learning with Diff Pruning. In *ACL*, 4884–4896.

He, J.; Zhou, C.; Ma, X.; Berg-Kirkpatrick, T.; and Neubig, G. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-efficient transfer learning for NLP. In *ICML*, 2790–2799.

Hu, E. J.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2021. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*.

Ivison, H.; and Peters, M. E. 2022. Hyperdecoders: Instance-specific decoders for multi-task NLP. In *EMNLP*, 1715–1730.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; and Hinton, G. E. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1): 79–87.

Khashabi, D.; Chaturvedi, S.; Roth, M.; Upadhyay, S.; and Roth, D. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *NAACL*, 252–262.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Lester, B.; Al-Rfou, R.; and Constant, N. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning. In *EMNLP*, 3045–3059.

Levesque, H.; Davis, E.; and Morgenstern, L. 2012. The winograd schema challenge. In *KR*.

Pilehvar, M. T.; and Camacho-Collados, J. 2019. WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations. In *NAACL*, 1267–1273.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8): 9.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140): 1–67.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*, 2383–2392.

Razdaibiedina, A.; Mao, Y.; Khabsa, M.; Lewis, M.; Hou, R.; Ba, J.; and Almahairi, A. 2023. Residual Prompt Tuning: improving prompt tuning with residual reparameterization. In *ACL*, 6740–6757.

Rücklé, A.; Geigle, G.; Glockner, M.; Beck, T.; Pfeiffer, J.; Reimers, N.; and Gurevych, I. 2021. AdapterDrop: On the Efficiency of Adapters in Transformers. In *EMNLP*, 7930–7946.

Shi, Z.; and Lipani, A. 2024. DePT: Decomposed Prompt Tuning for Parameter-Efficient Fine-tuning. In *ICLR*.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 1631–1642.

Su, Y.; Wang, X.; Qin, Y.; Chan, C.-M.; Lin, Y.; Wang, H.; Wen, K.; Liu, Z.; Li, P.; Li, J.; et al. 2022. On Transferability of Prompt Tuning for Natural Language Processing. In *NAACL*, 3949–3969.

Sung, Y.-L.; Cho, J.; and Bansal, M. 2022. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35: 12991–13005.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*, 6000–6010.

Vu, T.; Lester, B.; Constant, N.; Al-Rfou, R.; and Cer, D. 2022. SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer. In *ACL*, 5039–5059.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. SuperGLUE: a stickier benchmark for general-purpose language understanding systems. In *NeurIPS*, 3266–3280.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *EMNLP*, 353–355.

Wang, Y.; Agarwal, S.; Mukherjee, S.; Liu, X.; Gao, J.; Hassan, A.; and Gao, J. 2022a. AdaMix: Mixture-of-Adaptations for Parameter-efficient Model Tuning. In *EMNLP*, 5744–5760.

Wang, Z.; Panda, R.; Karlinsky, L.; Feris, R.; Sun, H.; and Kim, Y. 2022b. Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning. In *ICLR*.

Warstadt, A.; Singh, A.; and Bowman, S. R. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7: 625–641.

Williams, A.; Nangia, N.; and Bowman, S. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *NAACL*, 1112–1122.

Xiao, Y.; Xu, L.; Li, J.; Lu, W.; and Li, X. 2023. Decomposed Prompt Tuning via Low-Rank Reparameterization. In *EMNLP*.

Zaken, E. B.; Goldberg, Y.; and Ravfogel, S. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. In *ACL*, 1–9.

Zhang, S.; Liu, X.; Liu, J.; Gao, J.; Duh, K.; and Van Durme, B. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.