EPT: EXPLOSIVE PROMPT TUNING For Parameter-Efficient with Large Norm Prompt

Anonymous EMNLP submission

Abstract

Prompt tuning introduces additional learnable tokens, known as *soft prompts*, to frozen pre-trained language models for parameter-efficient tuning. Unlike fine-tuning, only these soft prompts are trained on downstream tasks rather than all model parameters. While recent prompt tuning approaches that introduce a reparameterization network have shown comparable performance to fine-tuning, they still require a large number of parameters for the soft prompts. In this paper, we empirically show the characteristics of the recent prompt tuning methods, such as the large norms of trained soft prompts and their significant similarity to each other. Inspired by these observations, we propose simple yet effective modifications to the reparameterization network for efficient prompt tuning, which involves inducing large norm, replacing overparameterization with underparameterization, and focusing on a single prompt. This approach preserves the advantageous characteristics of the soft prompts while significantly reducing the number of parameters. Our comprehensive experiments across 21 diverse NLP datasets show that our method called EPT: EXPLOSIVE PROMPT TUNING, significantly outperforms prompt tuning and achieves comparable performance full fine-tuning or other parameter-efficient tuning, with only 2.3K parameters during training on T5-base.

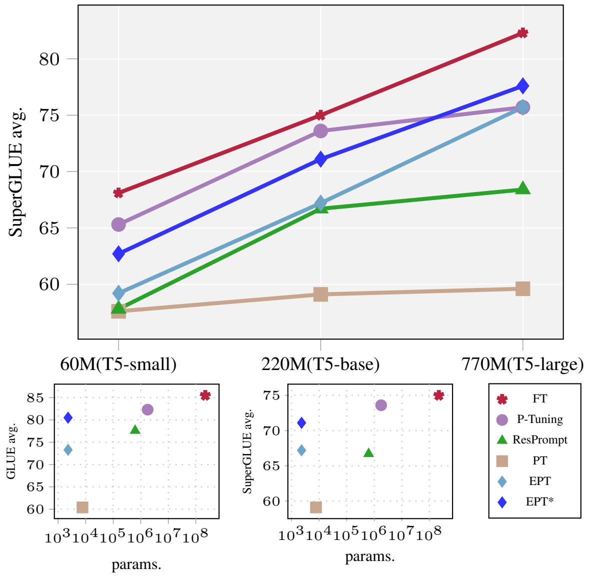


Figure 1: EPT and its variant EPT* outperform Prompt-Tuning (PT; Lester et al., 2021) and ResPrompt (Razdaibiedina et al., 2023), and reduce the gap with P-Tuning (Liu et al., 2021) and Fine-tuning (FT) on SuperGLUE tasks by using a *single* soft prompt in T5-series models (**Top**). In terms of parameter efficiency, using an *underparameterized* MLP allows for a significantly reduced the number of training parameters (**Bottom**).

1 Introduction

Pre-trained Language Models (PLMs) (Devlin et al., 2019; Radford et al., 2019; Raffel et al., 2020) have demonstrated remarkable performance in various natural language processing (NLP) tasks, typically using the *pretrain-then-finetune* paradigm (Liu et al., 2019). However, fine-tuning all the model parameters for individual downstream tasks requires a substantial memory footprint and training time, making it inefficient for large-scale PLMs.

To address the computational expense of fine-tuning, researchers have explored prompt-tuning (PT) in large-scale PLMs. PT is an efficient method that prepends learnable prompt vectors (referred to as *soft prompts* or *continuous prompts*) to the input embeddings of the model. It updates only the soft prompts while freezing the rest of the model parameters. These learned soft prompts provide PLMs with task-specific information for each downstream task.

Recently, some studies have introduced reparameterization networks to soft prompts to reduce the performance gap between vanilla PT (Lester et al., 2021) and fine-tuning in moderate-scale models (less than 11 billion parameters). P-tuning (Liu et al., 2021) employs a long short-term memory (LSTM) network and a multi-layer perception (MLP) as the reparameterization network to promote the discreteness of the soft prompts in continuous space. ResPrompt (Razdaibiedina et al., 2023) applies layer normalization (Ba et al., 2016) and a residual connection (He et al., 2016) to the reparameterization network to alleviate poor performance with shorter prompt lengths and sensitivity to hyper-parameters. While the reparameterization-based PT (Liu et al., 2021; Razdaibiedina et al., 2023) has successfully improved the performance of vanilla PT, their reparameterization networks generally require overparameterization compared to vanilla PT. Additionally, we find that the reparameterization networks lead to redundancy problems in the soft prompts.

In this paper, we analyze the trained soft prompts of both vanilla PT and reparameterization-based PT models through pilot experiments to gain key insights for designing more efficient soft prompts. We find that (1) reparameterized soft prompts have large norms. (2) After discovering high cosine similarities indicating redundancy among these prompts, we find that using a broadcast prompt initialized from a subset can achieve the original performance level as using full soft prompts. Reflecting on this, (3) focusing on a single large norm prompt shows insensitivity to network size. Based on these observations, We propose EPT: EXPLOSIVE PROMPT TUNING which modifies the reparameterization-based PT network to induce large norms with gradient exploding. This allows us to replace the overparameterized network with an underparameterized network, improving parameter efficiency while eliminating the redundancy of highly similar soft prompts. It also enables the use of a single prompt or a broadcast prompts, while maintaining the advantages of reparameterized prompts.

To verify the effectiveness of EPT, we conduct comprehensive experiments on GLUE, SuperGLUE, MRQA, and Others benchmark of Asai et al., 2022 with T5-small, T5-base, and T5-Large models (Raffel et al., 2020). As shown in Figure 1 and Table 1, EPT and its variant EPT* outperform ResPrompt and Prompt-Tuning and achieve comparable or outperforming performance with P-Tuning and fine-tuning, despite using a single prompt and tuning significantly fewer parameters, 2.3K for T5-base and 3K for T5-Large. Our approach is also effective on few-shot settings (i.e. 4-32 shots) (Table 3).

2 Related Works

2.1 Parameter-Efficient Tuning Methods

Pre-trained Language Models (PLMs) have demonstrated remarkable performance across various Natural Language Processing (NLP) tasks, leading to widespread adoption. Since the emergence of Transformer-based model (Vaswani et al., 2017) such as BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), T5 (Raffel et al., 2020), recently advancements have introduced large-scale PLMs (Brown et al., 2020; Chowdhery et al., 2022). However, fine-tuning has become parameter-inefficient as it requires updating all model parameters due to the exponential increase in the parameters of PLMs. Moreover, it is computationally expensive in terms of time and memory to store and deploy all model parameters of adapted PLMs for each task. In response to the challenges of fine-tuning, Parameter-Efficient Tuning (PEFT) methods have emerged as promising alternatives. It updates only a subset of model parameters while adapting to various downstream tasks, demonstrating competitive performance to full fine-tuning.

We divide PEFT methods into *parameter composition*, *extra module*, and *input composition*. Parameter composition methods involve simply combining task-specific parameters with model parameters, as shown in approaches like BitFiT (Ben Zaken et al., 2022) and LoRA (Hu et al., 2021). Extra module methods introduce task-specific modules, such as Adapters (Houlsby et al., 2019) and IA^3 (Liu et al., 2022). Input composition methods prepend task-specific learnable prompt to the model's input, as demonstrated in approaches like Prefix-Tuning (Li and Liang, 2021), P-Tuning (Liu et al., 2021), and Prompt-Tuning (Lester et al., 2021).

2.2 Prompt Tuning

Lester et al. (2021) achieved competitive performance by simply prepending *soft prompt* into the input sequence of PLMs without modifying the model parameters. The tuned prompt consists of less than 0.1% of the total parameters, making it reduces the cost of copying, storing, and deploying. However, comparable performance to full fine-tuning is typically achieved only with large-scale PLMs or by using long soft prompts, which can lead to increased training and inference times.

Recently, there exist methods using a reparameterization network with soft prompts (Liu et al., 2021; Razdaibiedina et al., 2023). Liu et al., 2021 employs the reparameterization network consisting of LSTM or MLP layer to promote the discreteness of continuous prompts, aiming to address the underperformance of decoder-only models on NLU tasks. In the case of Razdaibiedina et al., 2023, a residual connection is employed in the reparameterization network (i.e., MLP) to enhance the performance robustness to the hyper-parameter associated with prompt tuning using shorter soft prompts. However, these reparameterization-based prompt tuning require tens or hundreds more parameters. More recently, SPoT (Vu et al., 2022), ATTEMPT (Asai et al., 2022), and MPT (Wang et al., 2023) propose a prompt-based transfer learning. SPoT improves the performance of prompt tuning by initializing soft prompts for the target task after training prompts on one or more source tasks. ATTEMPT trains an attention module to interpolate between the source prompts and the target prompts. MPT distills knowledge from multiple task source prompts into target prompts. These methods require pre-training on any source tasks to obtain a collection of source prompts.

3 Method

3.1 Preliminaries

Vanilla Prompt Tuning Lester et al. (2021) proposed prompt tuning that defines a length m sequence of soft prompts $P = \{P_1, \dots, P_m\} \in \mathbb{R}^{m \times e}$, which is prepended to the input embeddings X, and learn only them for adaptation to downstream tasks. The model parameters θ are frozen, and only soft prompt parameters θ_P are stored after training. The training objective of prompt tuning is as follows:

$$\arg\max_{\theta_P} \log p_\theta(Y \mid [P; X]), \tag{1}$$

Reparameterization-based Prompt Tuning Liu et al., 2021; Razdaibiedina et al., 2023 proposed that use a reparameterization network $\phi_{\mathcal{O}}$ (see Figure 6 (a)). Before prepending the soft prompts to the input embeddings, they project it into reparameterized soft prompts P' as follows.

$$P' = [P'_1, \dots, P'_m] = \Phi_{\mathcal{O}}(P), \tag{2}$$

where $\Phi_{\mathcal{O}}(\cdot)$ is a reparameterization function composed of the network $\phi_{\mathcal{O}}$ with overparameterization. P-Tuning (Liu et al., 2021) configures $\phi_{\mathcal{O}}$ using LSTM or MLP, and ResPrompt constructs $\phi_{\mathcal{O}}$ by adding residual connection and layer norm to a bottleneck-structured MLP. They train only the soft prompt parameters θ_P and the reparameterization network parameters $\theta_{\phi_{\mathcal{O}}}$. The training objective of reparameterization-based prompt tuning is as follows:

$$\arg\max_{\theta_P, \theta_{\phi_{\mathcal{O}}}} \log p_\theta(Y \mid [P'; X]), \tag{3}$$

After training, they discard the reparameterization network and use the projected prompt P' during inference.

3.2 Reparameterized Soft Prompt

In this section, we observe the l_2 norm and similarity of soft prompts for the vanilla PT and reparameterization-based PT methods. One important aspect of the comparison is discussing the characteristics of the soft prompts in reparameterization-based PT that contribute to performance improvement over vanilla PT.

Reparameterized soft prompts show a growing large norms during training. We take the perspective of understanding the differences in the dynamics of gradient descent between vanilla PT and reparameterization-based PT for soft prompts. We observe the trend of l_2 norm over time steps t for each method during the training as potential empirical observation. To measure this trend for soft prompts norm at each time step t, we calculate the average norm of each prompt.

$$\mu_t = \frac{1}{n} \sum_i \|\theta_{P_i^t}\|_2 \tag{4}$$

where μ_t denotes the average norm of the soft prompts at time step t, n is the prompt length, and $\theta_{P_i^t}$ represents the parameter of the i-th prompt at t-th step.

As shown in Figure 2, reparameterization-based PT shows an increasing trend in μ_t and evaluation accuracy for Boolq and RTE tasks as the training step t progresses. In contrast, vanilla PT shows no noticeable increase in either metric. This suggests that training to increase the norm of the soft prompt can lead to better performance compared to vanilla PT.

Reparameterized soft prompts contain redundant representations. To identify the representation of the trained soft prompt, we estimate the

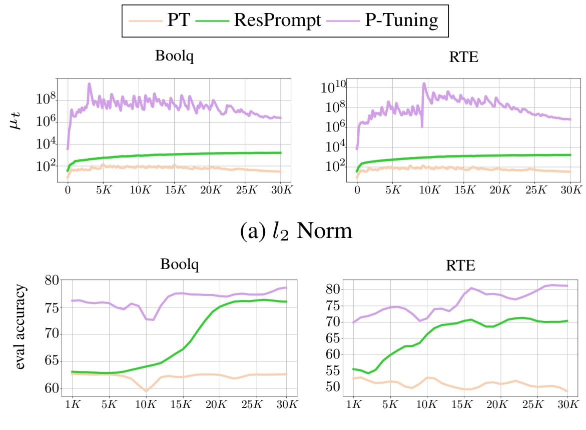


Figure 2: Illustration depicting the growth trends of the average norm μ_t in Eq. (4) **(a)** and evaluation accuracy on the validation set **(b)** during 30K training steps t for PT, ResPrompt, and P-Tuning on Boolq and RTE tasks using T5-base. The norm was measured at every step, while the accuracy was measured every 1K steps.

similarity between each prompt by measuring the cosine similarities. Figure 3 shows a correlation heatmap of the cosine similarities. We observe distinct trends between vanilla PT, which exhibits diverse representations, and reparameterization-based PT, which shows significantly high similarity[1]. This suggests that the reparameterized soft prompt serves as a representation that guides the model towards output labels, exhibiting redundancy. See Appendix C for additional results.

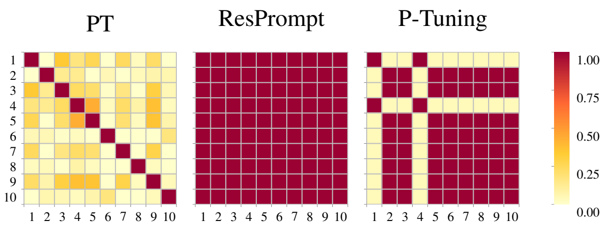


Figure 3: Illustration of correlation heatmap between learned soft prompts of length 10 for PT, ResPrompt and P-Tuning on Boolq task with T5-base. Each score represents the cosine similarities between each token.

The subset of reparameterized soft prompts achieves comparable performance to the original soft prompts. We identify the potential for eliminating redundant representations by initializing subsets of reparameterized soft prompts, and compare their performance. As shown in Figure 4, we observe the performance based on the concatenation length l by selecting one P_i from original soft prompts P and concatenating it to form $P_c = [P_i, P_i, ..., P_i] \in \mathbb{R}^{l \times e}$. For P-Tuning, which exhibits the largest norm, reaches the original performance level with a shorter prompt length compared to ResPrompt – Initializing with a length of 3 achieves 95.1% of the original accuracy on Boolq and 93.0% on RTE. For ResPrompt, initializing with a length of 4 achieves 90.0% on Boolq, while a length of 7 surpasses the original accuracy on RTE. In contrast, vanilla PT shows no performance improvement. This suggests that an optimal prompt can be achieved using a single instance from reparameterizing, allowing for a single or broadcast prompt.

[1]The dual representation of P-Tuning and the same representation of ResPrompt are discussed in Appendix A.1

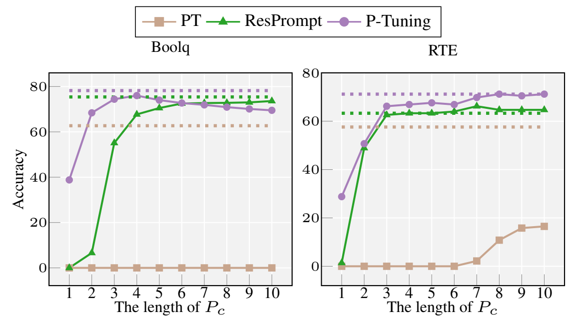


Figure 4: The performance comparison of PT, ResPrompt, and P-Tuning with T5-base on Boolq and RTE tasks using the concatenated soft prompts P_c chosen one from the original soft prompts. The solid lines represent the performance based on the concatenated length of P_c. The dotted lines indicate the original accuracy when using the full soft prompts.

Overparameterization. We apply the above observations to induce large norms with gradient exploding and eliminate redundancy, focusing on a single prompt (see Figure 6 (c)), and show the performance on Boolq and RTE tasks for various dimensions $\{1, 5, 10, 50, 100, 500, 1000\}$ of the reparameterization network in Figure 5. In contrast to the trend observed in Razdaibiedina et al., 2023, where performance declined with smaller prompt lengths while improved with increasing MLP dimension, the single prompt with large norm does not exhibit significant performance improvement or decline across various dimensions. This indicates that performance enhancement through reparameterization networks inducing norm growth does not scale proportionally with the parameter size of the network. See Appendix D for additional results.

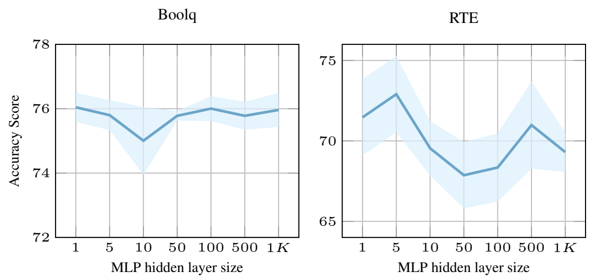


Figure 5: The performance of EPT with T5-base for Boolq and RTE tasks based on MLP hidden size. The blue line and shadow represent the average and standard devidations respectively over 3 runs.

3.3 Explosive Prompt Tuning

Our approach includes modifying the reparameterization network as shown in Figure 6 while preserving the advantageous characteristics of the soft prompts observed in reparameterization-based PT on Section 3.2. First, we construct a simple linear network using only the down-up feedforward layer, under the assumption that the layer norm and non-linearity in ResPrompt suppress the growth of norms, as shown in Figure 2. This assumption is important because ResPrompt does not reach the performance of P-Tuning, which exhibits extreme norm growth. Secondly, we focus on a single prompt based on the observations from Figure 3 and Figure 4 showing the potential for eliminating redundancy in reparameterized soft prompts.

We project the single prompt P_1 into a reparameterized soft prompt P' through the modified network $\phi_{\mathcal{U}}$ with underparameterization. The broadcast prompts P'_b is constructed by concatenating n reparameterized prompts $[P'_1, P'_1, \dots, P'_1] \in \mathbb{R}^{n \times e}$, then is prepended to the input embeddings X. We train the single prompt parameters θ_{P_1} and the modified MLP parameters $\theta_{\phi_{\mathcal{U}}}$. The training objective of EPT is as follows:

$$\underset{\theta_{P_1}\theta_{\phi_{\mathcal{U}}}}{\arg\max} \log p_\theta(Y \,|[\phi_{\mathcal{U}}(P'_b); X] \quad (5)$$

4 Experiments

4.1 Datasets

To cover diverse NLP tasks in our comprehensive experiments, we evaluate our method EPT on 21 tasks including linguistic acceptability, entailment, similarity and paraphrase detection, sentiment analysis, question answering, commonsense reasoning,

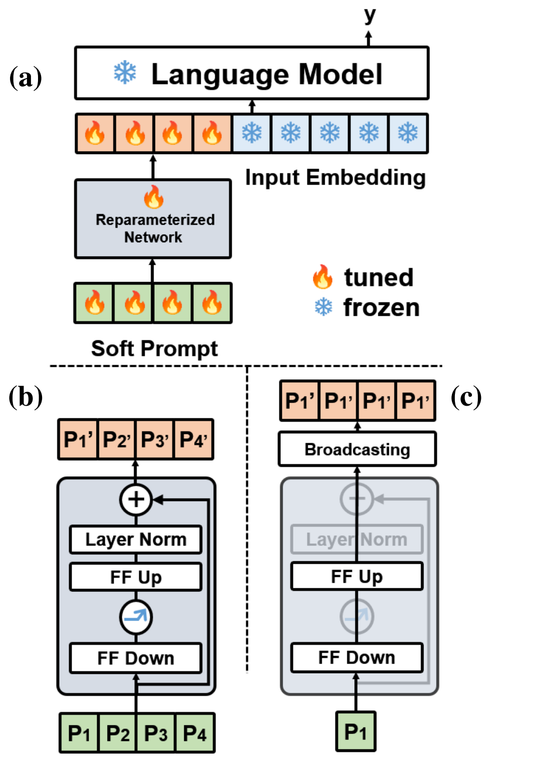


Figure 6: **(a)** Illustration of prompt tuning with a reparameterization network. **(b)** The reparameterization network (i.e., MLP) used in ResPrompt. The structures and flow in **(c)** related to reducing norm are removed. It enables flexible broadcasting to extend the representation of a single prompt.

and natural language inference. More details are provided in the Appendix B.1.

GLUE and SuperGLUE. We use 5 SuperGLUE (Wang et al., 2019a) and 8 GLUE (Wang et al., 2018) tasks to test NLU ability: Boolq (Clark et al., 2019), CB (de Marneffe et al., 2019), WiC (Pilehvar and Camacho-Collados, 2018), WSC (Levesque et al., 2012) and MutliRC (Khashabi et al., 2018); CoLA (Warstadt et al., 2019), MRPC (Dolan and Brockett, 2005), RTE (Giampiccolo et al., 2007), STSB (Cer et al., 2017), MNLI (Williams et al., 2018), QNLI (Demszky et al., 2018), QQP (Wang et al., 2019b), and SST-2 (Socher et al., 2013).

MRQA and Others. We use 4 MRQA 2019 tasks (Fisch et al., 2019) to test on large-scale QA dataset: Natural Questions (NQ; Trischler et al., 2017), NewsQA(News; Trischler et al., 2017), SearchQA(SQA; Dunn et al., 2017), and HotpotQA(HQ; Yang et al., 2018). Additionally, we experiment on 4 "Others" benchmark in Asai et al., 2022: WinoGrande (WG; Sakaguchi et al., 2021), Yelp-2 (Zhang et al., 2015), SciTail (Khot et al., 2018), and PAWS-Wiki (Zhang et al., 2019).

4.2 Pre-trained Models

We experiment using the publicly available pre-trained models on the HuggingFace (Wolf et al., 2020) of T5 (Raffel et al., 2020). We consider T5-small (60M), T5-Base (220M), and T5-Large (770M) to cover moderate scales.

4.3 Baselines

Fine-Tuning. Full Fine-Tuning is the standard approach (Raffel et al., 2020; Aribandi et al., 2022) of T5, where all the pre-trained parameters are updated on each downstream task.

Prompt-Tuning. The vanilla prompt tuning of Lester et al., 2021 is an approach that prepends the soft prompts to the input sequence embeddings.

P-Tuning. Liu et al., 2021 employs an encoder composed of LSTM or MLP as a reparameterization network. The soft prompts pass through the encoder to optimize the prompt in a continuous space.

ResPrompt. Razdaibiedina et al., 2023 adds residual connection and layerNorm to the reparameterization network composed of bottleneck design to improve the performance and stability.

4.4 Implementation

In our study, we translate all datasets into a text-to-text format following (Raffel et al., 2020). Since most datasets do not publicly release their test set, we generate the test set by constructing or sampling from the validation set. In the main results, we use pre-trained T5 checkpoints across three scales: Small, Base, and Large with 60M, 220M, and 770M parameters, respectively, as the LMs for EPT and all of the baselines. For other experiments, we use T5-base as the base LM. Excluding the few-shot setting, we train for 30K steps with batch size of 16. We experiment on short prompt length 10 for vanilla prompt tuning (Lester et al., 2021) and reparameterization-based prompt tuning (Liu et al., 2021; Razdaibiedina et al., 2023), and single prompt for EPT to demonstrate our approach. In the case of EPT*, this is a variant where a single prompt is concatenated to extend it to a length of 10. More detailed implementations and hyper-parameters are in Appendix B.2.

5 Results

5.1 Main Results

EPT significantly improves the performance of prompt tuning with fewer parameters. We compare EPT with prompt tuning and reparameterization-based prompt tuning. First, Table 1 presents the results on SuperGLUE and GLUE. Under the same model scale and short prompt length settings, reparameterization-based prompt tuning significantly outperforms prompt tuning. Moreover, EPT surpasses ResPrompt using much fewer parameters except for T5-base and T5-small on GLUE, and matches the performance of P-Tuning for T5-Large on SuperGLUE, using approximately a thousand times fewer parameters. Second, Table 2 shows the results on MRQA and Others. EPT achieves 66.3 average F1 on MRQA, matching ResPrompt, and yields 77.3 average accuracy on Others, outperforming ResPrompt (76.7).

EPT* largely closes the gap with P-Tuning and fine-tuning, maintaining fewer parameters. EPT still does not improve the performance of prompt tuning on several datasets such as CoLA (0%), and MultiRC (59.7%) in T5-base. Based on the observations in Figure 4, we introduce a variant of EPT, named EPT*, which maintains efficiency by concatenating a single soft prompt ten times into broadcast prompts instead of using ten individual soft prompts that exhibits redundant representations. Since it involves training a concatenation of single soft prompts, the training and inference parameters remain unchanged. First, the performance results of the broadcast prompts for all lengths on CoLA and MultiRC are shown in Figure 7. We observe a significant performance improvement when the broadcast prompts length exceeds 2. Second, from the results in Table 1 and Table 2, EPT* outperforms ResPrompt and Prompt-Tuning across all tasks and model scales while maintaining efficiency, and even surpasses P-Tuning for T5-Large on SuperGLUE.

5.2 Few-shot Adaptations

We conduct additional experiments in few-shot settings to measure the effectiveness of EPT in low-resource scenarios and evaluate its generalization capabilities. The training data consists of k ($k = \{4, 16, 32\}$) randomly selected sampled, with the number of classes sampled consistently for 13 NLP tasks from SuperGLUE and GLUE. As shown

Method		Param /task	GLUE CoLA Matt	MRPC Acc	RTE Acc	STS-B Pearson	MNLI Acc	QNLI Acc	QQP Acc	SST-2 Acc	Avg -	SuperGLUE Boolq Acc	CB F1/Acc	WiC Acc	WSC Acc	Multi F1	Avg -
T5-Large	FT	770M	61.7	89.4	88.5	91.9	89.7	94.4	91.4	95.9	87.9	85.8	88.6	71.9	85.3	79.9	82.3
	P-Tuning	3.1M	**58.0**	88.1	85.4	91.2	**88.2**	**94.1**	90.9	95.2	**86.4**	**82.9**	87.0	68.8	60.9	**79.1**	75.7
	ResPrompt	832K	54.8	**88.2**	**85.9**	**91.5**	66.1	93.8	90.8	95.0	83.2	82.1	51.2	**69.2**	61.5	78.2	68.4
	PT	10K	0.7	74.9	50.8	91.2	35.7	89.7	88.5	87.4	64.9	62.9	58.6	55.7	60.9	59.8	59.6
	EPT	**3K**	56.8	87.9	85.1	91.1	84.4	93.6	**91.0**	**95.4**	85.7	82.2	89.1	67.2	61.5	78.7	75.7
	EPT*	**3K**	54.3	87.9	85.4	91.2	87.2	93.8	**91.0**	95.3	85.8	82.4	**95.4**	69.1	**62.2**	**79.1**	**77.6**
T5-base	FT	220M	59.8	87.9	81.8	90.6	85.9	93.1	90.4	94.2	85.5	82.8	87.6	67.9	61.5	74.9	75.0
	P-Tuning	1.7M	**47.2**	85.7	**76.3**	89.7	**83.0**	**92.8**	90.3	93.5	**82.3**	**78.8**	**88.3**	**67.0**	60.3	**73.6**	**73.6**
	ResPrompt	624K	34.1	**85.8**	65.7	**90.3**	72.6	92.6	89.4	90.3	77.6	76.0	64.2	65.1	61.5	66.4	66.7
	PT	7.6K	0.0	67.6	56.6	90.2	48.8	69.7	65.9	84.7	60.4	62.1	59.2	53.0	61.5	59.6	59.1
	EPT	**2.3K**	0.0	82.9	68.8	89.7	69.3	92.7	90.2	93.0	73.3	75.5	73.1	65.6	**62.2**	59.7	67.2
	EPT*	**2.3K**	41.0	85.0	75.8	89.7	76.1	92.6	**90.4**	**93.8**	80.5	77.0	82.7	65.3	58.3	72.4	71.1
T5-small	FT	60M	36.3	85.9	67.9	88.8	78.8	90.4	88.1	91.1	78.4	76.5	77.8	66.6	51.3	68.4	68.1
	P-Tuning	793K	0.0	**84.2**	65.0	**86.6**	**75.5**	**89.4**	**88.3**	**89.1**	**72.3**	**71.1**	**64.5**	**65.6**	59.6	**65.7**	**65.3**
	ResPrompt	416K	0.0	80.9	61.4	85.9	62.1	88.9	88.2	88.1	69.4	61.9	58.4	50.6	59.0	59.3	57.8
	PT	5.1K	**2.8**	75.6	52.3	85.6	41.5	87.2	85.8	82.0	64.1	61.8	55.2	48.7	**62.2**	60.0	57.6
	EPT	**1.5K**	0.8	78.0	57.6	85.6	62.7	87.7	87.8	83.4	68.0	61.9	59.1	51.8	59.6	63.7	59.2
	EPT*	**1.5K**	0.0	82.8	**65.9**	86.3	68.7	89.0	88.2	88.8	71.2	68.0	59.2	60.4	60.3	65.5	62.7

Table 1: Main results on GLUE and SuperGLUE tasks, averaged over 3 runs. We use Pearson Correlation for STS-B, Matthews Correlation for CoLA, F1 for MultiRC (Multi), the average score for tasks with two metrics, and accuracy for other tasks as metrics. "param/task" represents the number of trainable parameters for each task. Excluding fine-tuning, The best results are in bold with underline representing the second best ones.

Method	Param /task	MRQA NQ F1	HP F1	SQA F1	News F1	Avg -	Others WG Acc	Yelp Acc	SciTail Acc	PAWS Acc	Avg -
FT	220M	75.0	78.7	80.2	66.7	75.1	59.1	97.0	94.9	94.2	86.3
P-Tuning	1.7M	**66.1**	72.9	**71.3**	**61.8**	**68.0**	48.6	**95.8**	**91.3**	**89.0**	**81.2**
ResPrompt	624K	64.9	72.7	67.5	60.3	66.3	**48.9**	95.1	88.4	74.6	76.7
PT	7.6K	64.7	71.6	66.7	60.4	65.8	48.5	93.7	68.0	55.7	66.5
EPT	2.3K	64.4	72.3	68.2	60.2	66.3	**48.9**	93.8	89.0	77.2	77.3
EPT*	2.3K	65.0	**73.1**	69.5	60.4	67.0	48.2	95.4	90.1	84.8	79.6

Table 2: Results on MRQA QA datasets, WinoGrande (WG), Yelp, Scitail, and PAWS, averaged over 3 runs. We use F1 for MRQA and accuracy for others. "param/task" represents the number of trainable parameters for each task. Excluding fine-tuning, The best results are in bold with underline representing the second best ones.

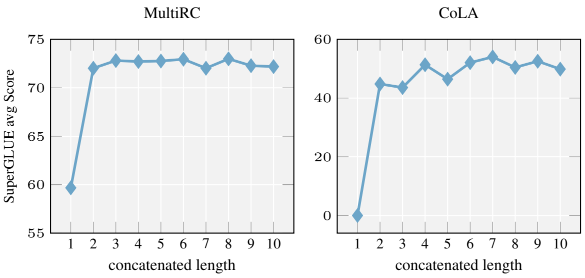


Figure 7: Performance comparison for concatenated soft prompt lengths from 1 to 10 on T5-base for MultiRC and CoLA tasks.

k-shot		FT	P-Tuning	ResPrompt	PT	EPT	EPT*
SuperGLUE	4	53.1	**56.3**	53.0	53.9	55.8	54.3
	16	56.8	**60.2**	51.2	54.2	57.9	57.1
	32	58.5	**59.9**	54.0	52.9	58.2	59.3
GLUE	4	57.2	**63.9**	46.8	48.9	60.7	60.8
	16	59.0	64.2	49.8	50.6	65.3	**65.7**
	32	61.8	69.1	56.2	52.9	**69.6**	69.0

Table 3: Few-shot ($k = \{4, 16, 32\}$) results on SuperGLUE and GLUE tasks for T5-base model, averaged over 6 runs.

in Table 3, we observe that ResPrompt and Prompt-Tuning underperform compared to fine-tuning, facing difficulties in few-shot adaptation. P-Tuning, EPT, and EPT* are effective in few-shot settings, outperforming ResPrompt, Prompt-Tuning, and fine-tuning except for EPT in 32-shot. Notably, EPT and EPT* surpass P-Tuning in the 16-shot and 32-shot settings on SuperGLUE, while using still only 2.3K parameters.

5.3 Robust learning rate selection

We compare the performance of EPT and EPT* with Prompt-Tuning and ResPrompt across various learning rates. We evaluate 5 SuperGLUE tasks with learning rates $\{0.003, 0.01, 0.03, 0.1, 0.3\}$. As

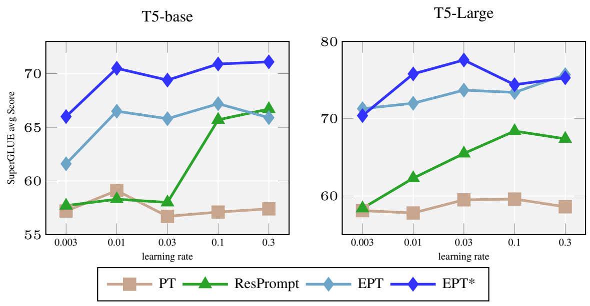


Figure 8: Performance on 5 SuperGLUE tasks with different learning rates on T5-base.

shown in Figure 8, our experiments indicate that Prompt-Tuning does not achieve consistent performance improvement across different learning rates on both T5-base and T5-Large. In contrast, ResPrompt demonstrates progressively better performance with increasing learning rates. For EPT and EPT*, it can be observed that they are robust to the selection of learning rate, despite using fewer parameters.

5.4 Other parameter-efficient fine-tuning methods

Method	Params	Multi	Bool	WiC	WSC	CB	Avg
Fine-tuning	220M/-	73.9	81.5	70.8	62.2	75.0	72.7
Adapter	1.9M/-	**75.5**	82.1	67.0	61.5	**81.0**	**73.4**
AdapterDrop	1.1M/-	75.3	**82.3**	67.7	**62.2**	73.8	72.3
BitFit	280K/-	74.7	79.9	**68.2**	54.5	72.6	70.0
ATTEMPT	232K/-	72.9	77.0	66.9	52.5	72.6	68.4
PT	77K/-	56.5	61.9	48.1	**62.2**	50	55.7
SPoT	77K/-	73.0	77.4	58.4	61.5	36.9	61.4
EPT	2.3K/768	67.9	70.3	55.8	61.5	61.9	63.5
EPT*	2.3K/768	73.3	76.7	65.2	**62.2**	67.9	69.1

Table 4: Results on 5 SuperGLUE tasks for T5-base model with other PEFT methods, averaged over 3 runs in Asai et al., 2022. Params column represents the training parameters/inference parameters for each method. Params with "-" indicate equivalence with training parameters.

We compare EPT and EPT* with other PEFT methods in the experimental setup of Asai et al., 2022 on 5 SuperGLUE tasks with T5-Base. Following the ATTEMPT, prompt-based tuning is trained with the prompt length of 100, and EPT and EPT* are trained with a single prompt of the configuration as the main results. In this setup, considering that the training is conducted for 10 or 20 epochs depending on the dataset size, an additional MLP layer is stacked for EPT and EPT* due to the fewer steps. The results show that EPT outperforms PT by 7.8 points and SPoT by 2.1 points, and EPT* surpasses ATTEMPT by 0.7 points and closes BitFit by 0.9 points difference. Notably, our approach reaches the performances using significantly fewer parameters and a single prompt, without requiring pre-trained source prompts like SPoT and ATTEMPT.

5.5 Ablation Studies

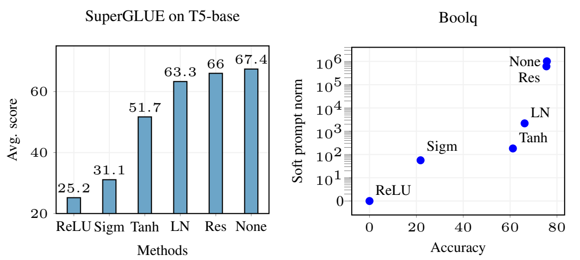


Figure 9: **(Left)** Performance of EPT with additional structures across 5 SuperGLUE tasks with T5-base. **(Right)** Comparison of additional structures in terms of accuracy (x-axis) and norm (y-axis) in the Boolq task.

We compare the performance of EPT, which does not add additional structures (None), with layer norm, residual connection, and three types of non-linearity across 5 SuperGLUE tasks. As shown in Figure 9, we observe that not adding any structures contributes to norm growth and performance improvement. All three types of non-linearity result in significant performance drops, particularly with ReLU, in which dead neurons in the underparameterized MLP lead to persistent performance degradation. Layer norm causes a decrease of 5.5 points. With the residual connections, the large discrepancy between the soft prompt before and after passing through the MLP results in no direct performance gain.

6 Conclusion

This work first shows that reparameterized soft prompts exhibit large norms and unnecessary redundancy. Inspired by the observations, we proposed EXPLOSIVE PROMPT TUNING (EPT), which intentionally induces large norms and eliminate the redundant prompts to significantly reduce the parameters of the reparameterization network and soft prompts. Extensive experiments on 21 NLP tasks demonstrate the comparable performance of our method with much fewer parameters.

Limitations

We have discovered the advantageous characteristics that contribute to the performance enhancement of reparameterized soft prompts. Therefore, a key question that remains underexplored is understanding the task-specific optimal representation of soft prompts having with those features. The interpretability of such soft prompts remains a limitation in the line of research work focusing prompt-based tuning. We believe that EPT offers a critial direction to enhance new interpretability of soft prompts. Moreover, we plan to explore scenarios such as multi-task learning (Wang et al., 2023) and transfer learning (Vu et al., 2022; Asai et al., 2022) with the reparameterized soft prompt. We leave these for future research targets.

References

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. Ext5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*.

Akari Asai, Mohammadreza Salehi, Matthew Peters, and Hannaneh Hajishirzi. 2022. ATTEMPT: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6655–6672, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions.

Marie-Catherine de Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. *Proceedings of Sinn und Bedeutung*, 23(2):107–124.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new qa dataset augmented with context from a search engine.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 1950–1965. Curran Associates, Inc.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT understands, too. *CoRR*, abs/2103.10385.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Anastasiia Razdaibiedina, Yuning Mao, Madian Khabsa, Mike Lewis, Rui Hou, Jimmy Ba, and Amjad Almahairi. 2023. Residual prompt tuning: improving prompt tuning with residual reparameterization. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6740–6757, Toronto, Canada. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. 2022. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059, Dublin, Ireland. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. Glue: A multi-task benchmark and analysis platform for natural language understanding.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. 2023. Multitask prompt tuning enables parameter-efficient transfer learning. *arXiv preprint arXiv:2303.02861*.

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

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. *arXiv preprint arXiv:1904.01130*.

Appendix

A More Analysis of Trained Soft Prompts

A.1 Interpretability

Soft prompts operate in a continuous embedding space rather than a discrete token space which makes them challenging to interpret. By measuring their nearest vocabulary neighbors using cosine similarity, Lester et al., 2021 observed the interpretability of learned soft prompts, often finding class labels. In our experiments, soft prompts learned by vanilla PT also often show close to class labels. Reparameterized soft prompts frequently show class labels. Specifically, in P-Tuning without layer Norm structure (Figure 3), two distinct directions of prompt vectors were identified: one closely aligned with output labels (positive prompts contribute significantly to performance) and the other not closely aligned with output labels (negative prompts do not contribute to performance). This underscores the notion that prompts guide and focus the model towards output labels. This feature is pronounced in reparameterization-based PT.

A.2 Zero-shot generalization

We conduct zero-shot transfer experiments across 21 NLP tasks. In Figure 10, we measure the cosine similarity between tasks using the soft prompts learned by EPT*. We then represent the zero-shot performance as a percentage of the original task performance, scaled from 0 to 1. As mentioned in Appendix A.1, we address the characteristics of prompts guiding the model towards output labels. This becomes more prominent when examining task similarities. For instance, classification tasks predicting 0 or 1 (e.g., e.g., GLUE, SuperGLUE, excluding STS-B, Others bechmark), regression tasks predicting the similarity between two sentences (STS-B), and question-answering tasks (MRQA) exhibit low similarity due to differing output labels.

B Experimental Setup

B.1 Dataset Details

Table 5 shows detailed settings for SuperGLUE, GLUE, MRQA, and Others datasets. We utilize the HuggingFace dataset (Lhoest et al., 2021). Since most datasets do not provide a test set, we split the training set or validation set to use as the test set. For small-scale datasets that have less than 10K in the training set, we split the validation set in half, using one half as the test set and the other as the validation set. For moderate-scale datasets that have less than 100K in the training set, we sample 1K from the training set to use as the validation set. For large-scale datasets that have more than 100K in the training set, we sample 10K from the training set to use as the validation set. The validation set is used as the test set in the moderate and large-scale datasets. We translate all tasks in both the SuperGLUE and GLUE datasets into text-to-text.

B.2 Implementation Details

Our code is implemented using HuggingFace Transformers (Wolf et al., 2020) and PEFT (Mangrulkar et al., 2022). We train the models on 10 NVIDIA RTX A6000 GPUs. We explore different learning rates for robustness in SuperGLUE. For the main results on SuperGLUE datasets, we search the learning rate from $\{1e^{-5}, 1e^{-4}, 1e^{-3}\}$ for fine-tuning, and $\{3e^{-3}, 1e^{-2}, 3e^{-2}, 1e^{-1}, 3e^{-1}\}$ for prompt-based tuning. For the other experiments except few-shot setting, we use a learning rate of $1e^{-5}$ for fine-tuning and $3e^{-1}$ for prompt-based tuning. For the few-shot experiments, we use a learning rate of $1e^{-3}$ for fine-tuning and $3e^{-1}$ for prompt-based tuning.

For all experiments except the few-shot setting, we train 30,000 steps, while for few-shot experiments, we train 10,000 steps, and select the best checkpoint based on the optimal performance on the validation set every 1,000 steps. We set the batch size to 16, and the input length to 256 for all tasks, except MultiRC has input length of 348 and MRQA tasks have input length of 512. We use AdamW (Loshchilov and Hutter, 2019) optimizer with weight decay of 0.01.

B.3 Prompt-based Tuning

In all experiments, vanilla and reparameterization based PT, used short soft prompts with 10 virtual tokens, while EPT and EPT* used 1 virtual token. For the encoder setup, P-Tuning employs an MLP consisting of 3 linear layers of the embedding dimension from the default of HuggingFace PEFT. ResPrompt stacked two encoders with a shared MLP of 400 hidden size, following Razdaibiedina et al., 2023. EPT utilizes a single hidden size for the underparameterized MLP.

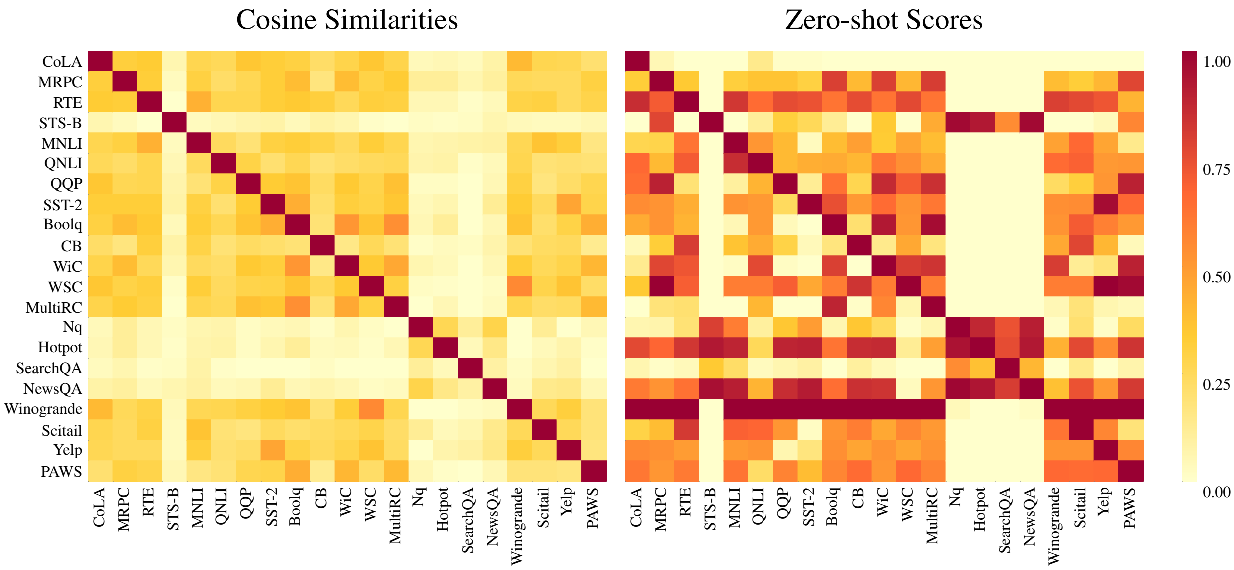


Figure 10: **(Left)** Illustration of cosine similarity between learned for each task using EPT*. **(Right)** Illustration of the ratio of the score achieved through zero-shot transfer for each task-specific soft prompt compared to the original score. The task transfer is performed along the x-axis towards the right.

		SuperGLUE				
Dataset	**Text Sources**	**Task**	**Train**	**Valid**	**Test**	**Metric**
Boolq	Google queries, Wikipedia	QA	9,427	1,635	1,635	acc.
CB	various	NLI	250	28	28	F1, acc.
Multirc	various	QA	26,243	1,000	4,848	F1
WiC	WordNet, VerbNet, Wiktionary	WSD	5,428	319	319	acc.
WSC	fiction books	coref.	554	52	52	acc.
		GLUE				
CoLA	misc.	Acceptability	8,551	521	522	matt.
SST-2	movie reviews	Sentiment	66,349	1,000	872	acc.
MRPC	news	Paraphrase	3,668	408	1,725	acc.
STS-B	misc.	Sentence Similarity	5,749	750	750	pearson
QQP	social QA questions	Paraphrase	353,846	10,000	40,430	acc.
MNLI	misc.	NLI	382,702	10,000	9,815	acc.
QNLI	Wikipedia	QA/NLI	94,743	10,000	5,463	acc.
RTE	news, Wikipedia	NLI	2,490	138	139	acc.
		MRQA				
NQ	Wikipedia	QA	94,071	10,000	12,836	F1
HotpotQA	Wkipedia reviews	QA	72,928	2,950	2,951	F1
SearchQA	Search snippets	QA	107,384	10,000	16,980	F1
NewsQA	News article	QA	74,160	2,106	2,106	F1
		Others				
WinoGrande	WikiHow	coref. / common.	40,398	633	634	acc.
Yelp	Yelp reviews	Sentiment	100,000	10,000	38,000	acc.
SciTail	science exams	NLI	23,596	652	652	acc.
PAWS-Wiki	Wikipedia	Paraphrase	49,401	4,000	4,000	acc.

Table 5: The details of 5 SuperGLUE tasks and 8 GLUE tasks in our experiments. *NLI* is natural language inference, *coref.* is coreference resolution, *common.* is commonsense, *QA* is question answering, and *WSC* is word sense disambiguation.

B.4 Prompt Initialization

In our experiments, we initialize the virtual embeddings by sampling uniformly from [-0.5, 0.5] following Lester et al., 2021. We also explore the different prompt initializations, and Table 6 compares the default uniform sampling with random vocabulary and class vocabulary initialization.

Init. / Task	Boolq	CB	RTE	WSC	Avg.
Random	75.7	72.5	70.5	65.4	71.0
Sample	75.4	77.6	69.8	63.5	71.6
Label	75.1	75.5	67.6	63.5	70.4

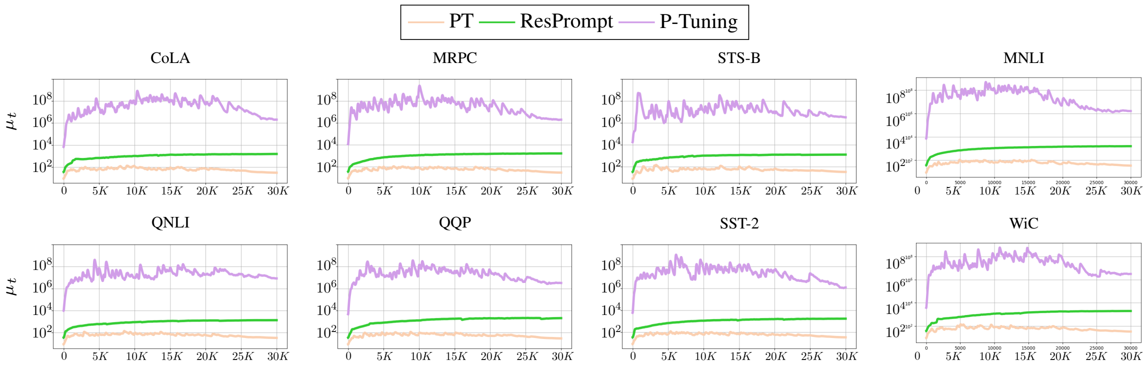
Table 6: We present results on three prompt initializations for T5-base in EPT: random uniform within the range of [-0.5, 0.5], sampled vocabulary, and label vocabulary.

C More Results for Norms of Soft Prompts

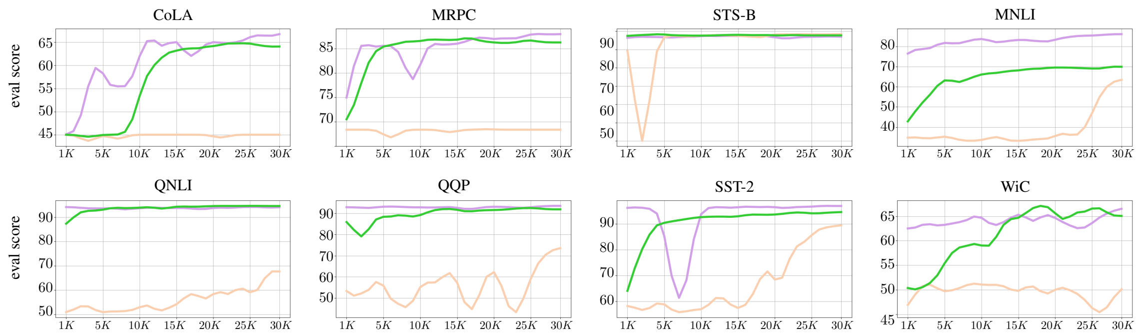
In Figure 11, we provide further observations on the norms of soft prompts in datasets with more than 1K training samples, focusing on the GLUE 7 tasks and WiC task from the SuperGLUE. The score metrics for each task are consistent with the main results. Reparameterization-based PT significantly outperforms vanilla PT on all tasks except for STS-B. At the same time, P-Tuning shows significantly faster convergence of both score and norm than ResPrompt on all tasks except for STS-B.

D More Results for Overparameterization

We provide further observations on the performance of the reparameterization network based on parameter size for the SuperGLUE tasks. As shown in Figure 12, the reparameterization network does not exhibit performance improvements with increased parameters for low-resource datasets such as CB (250 samples) and WSC (554 samples). Additionally, for WiC (less than 10K samples), which has a similar number of training samples as Boolq and RTE, the underparameterized network does not show a performance gap compared to the overparameterized network, and also exhibits low variance.



(a) l_2 Norm



(b) Score

Figure 11: The illustration depicts the trends of the average soft prompt norm μ_t in Eq. 4 **(a)** and the evaluation score **(b)** during 30K training steps for PT, ResPrompt, and P-Tuning using T5-base. The norm was measured at every step, while the scores was recorded every 1K steps.

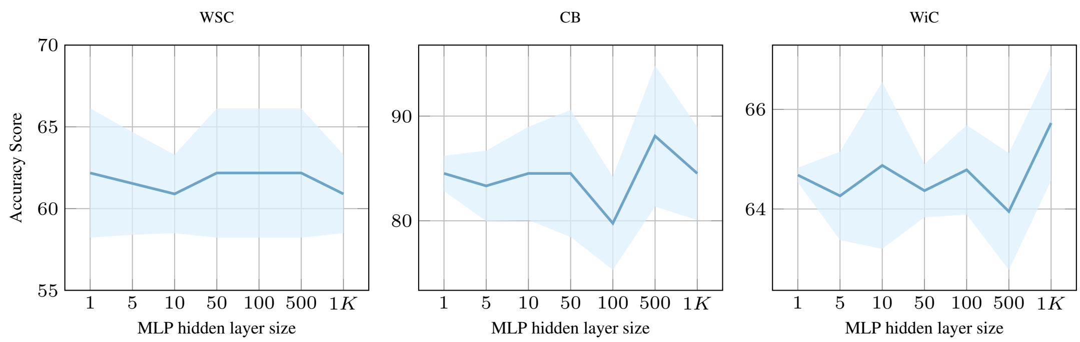


Figure 12: The performance of EPT with T5-base for WSC, CB and WiC tasks from the SuperGLUE based on MLP hidden size. The blue line and shadow represent the average and standard devidations respectively over 3 runs.