P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks

Xiao Liu[1,2*], **Kaixuan Ji**[1*], **Yicheng Fu**[1*], **Zhengxiao Du**[1,2], **Zhilin Yang**[1,2†], **Jie Tang**[1,2†]
[1]Tsinghua University, Beijing, China
[2]Beijing Academy of Artificial Intelligence (BAAI), Beijing, China
{liuxiao21,jkx19,fyc19}@mails.tsinghua.edu.cn

Abstract

Prompt tuning, which only tunes continuous prompts with a frozen language model, substantially reduces per-task storage and memory usage at training. However, in the context of NLU, prior work reveals that prompt tuning does not perform well for normal-sized pretrained models. We also find that existing methods of prompt tuning cannot handle hard sequence tagging tasks, indicating a lack of universality. We present a novel empirical finding that properly optimized prompt tuning can be universally effective across a wide range of model scales and NLU tasks. It matches the performance of finetuning while having only 0.1%-3% tuned parameters. Our method P-Tuning v2 is not a new method, but a version of prefix-tuning (Li and Liang, 2021) optimized and adapted for NLU. Given the universality and simplicity of P-Tuning v2, we believe it can serve as an alternative to finetuning and a strong baseline for future research.[1]

1 Introduction

Pretrained language models (Han et al., 2021a) improve performance on a wide range of natural language understanding (NLU) tasks such as question answering (Rajpurkar et al., 2016) and textual entailment (Dagan et al., 2005). A widely-used method, **fine-tuning**, updates the entire set of model parameters for a target task. While fine-tuning obtains good performance, it is memory-consuming during training because gradients and optimizer states for all parameters must be stored. Moreover, fine-tuning requires keeping a copy of model parameters for each task during inference, which is inconvenient since pretrained models are usually large.

Figure 1: Average scores on RTE, BoolQ and CB of SuperGLUE dev. With 0.1% task-specific parameters, P-tuning v2 can be comparable to fine-tuning across different scales of pre-trained models, while Lester et al. (2021) & P-tuning can only do so at the 10B scale.

Prompting, on the other hand, freezes all parameters of a pretrained model and uses a natural language prompt to query a language model (Brown et al., 2020). For example, for sentiment analysis, we can concatenate a sample with a prompt "This movie is [MASK]" and ask the pretrained language model to predict the masked token. We can then use the predicted probabilities of "good" and "bad" being the masked token to predict the sample's label. Prompting requires no training at all and stores one single copy of model parameters. However, prompting can lead to suboptimal performance in many cases compared to fine-tuning (Liu et al., 2021b; Lester et al., 2021).

Prompt tuning[2] is an idea of tuning only the continuous prompts. Specifically, Liu et al. (2021b); Lester et al. (2021) proposed to add trainable continuous embeddings to the original sequence of input word embeddings. These continuous embeddings (also called continuous prompts) are analogous to discrete manually designed prompts in prompting. Only the continuous prompts are updated during training. While prompt tuning improves over prompting on many tasks (Liu et al., 2021b; Lester et al., 2021), it still underperforms fine-tuning when the model size is small, specifically less than 10 billion parameters

[1]Codes will be at https://github.com/THUDM/P-tuning-v2
† corresponding to: Zhilin Yang (zhiliny@tsinghua.edu.cn) and Jie Tang (jietang@tsinghua.edu.cn)
* indicates equal contribution.

[2]We use "prompt tuning" to refer to a class of methods rather than a particular method.

(Lester et al., 2021). Moreover, as shown in our experiments, prompt tuning performs poorly compared to fine-tuning on several hard sequence tasks such as extractive question answering and sequence tagging (Cf. Section 4.3).

Our main contribution in this paper is a novel empirical finding that properly optimized prompt tuning can be comparable to fine-tuning universally across various model scales and NLU tasks. In contrast to observations in prior work, our discovery reveals the universality and massive potential of prompt tuning for NLU.

Technically, our approach P-tuning v2 can be viewed as an optimized version of prefix-tuning (Li and Liang, 2021), a method designed for generation and adapted to NLU. The most significant improvement originates from using **deep prompt tuning**, which is to apply continuous prompts for every layer of the pretrained model (Li and Liang, 2021; Qin and Eisner, 2021). Deep prompt tuning increases the capacity of continuous prompts and closes the gap to fine-tuning across various settings, especially for small models and hard tasks. Moreover, we present a few details of optimization and implementation for further enhancement of the results.

Experimental results show that P-tuning v2 matches the performance of fine-tuning at different model scales ranging from 300M to 10B parameters and on various hard NLU tasks such as question answering and sequence tagging. P-tuning v2 has 0.1% to 3% trainable parameters per task compared to fine-tuning, which substantially reduces training time memory cost and per-task storage cost.

2 Preliminaries

2.1 NLU Tasks

In this work, we categorize NLU challenges into two families: simple tasks and hard sequence tasks.

- **Simple NLU tasks** involve classification over a single label. Most datasets from GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019), including Text Classification (e.g., SST-2), Natural Language Inference (NLI, e.g., MNLI-m, RTE), Multiple-choice Question Answering (e.g., BoolQ), and so on, are in this category.

- **Hard sequence NLU tasks** involve classification over a sequence of labels. They are problems mostly related to information extraction, such as Open Information Extraction, Named Entity Recognition, Extractive Question Answering, and Semantic Role Labeling.

2.2 Prompt Tuning

Prompt tuning (Lester et al., 2021), or P-tuning (Liu et al., 2021b), introduces trainable continuous prompts as a substitution to natural language prompts for NLU when backbone language models' parameters are frozen. For example, let $\mathcal{V}$ refers to the vocabulary of a language model $\mathcal{M}$ and $\mathbf{e}$ serves as the embedding function for $\mathcal{M}$.

To classify a film review $\mathbf{x}$ ="Amazing movie!" as positive or negative, it is natural to think of appending a prompt "It is [MASK]" to the review and generating the conditional probabilities of the mask token being predicted as "good" or "bad" as the classification. In this case, prompt tokens {"It", "is", "[MASK]"} belong to the model's vocabulary $\mathcal{V}$, and the input embedding sequence would be

$$[\mathbf{e}(\mathbf{x}), \mathbf{e}(\text{"It"}), \mathbf{e}(\text{"is"}), \mathbf{e}(\text{"[MASK]"})] \quad (1)$$

However, since the model $\mathcal{M}$ is intrinsically continuous, from the perspective of optimization, one can never achieve the optimum with discrete natural prompts. P-tuning, instead, proposes to replace prompt tokens with trainable continuous embeddings $[h_0, ..., h_i]$ and turn the input sequence into

$$[\mathbf{e}(\mathbf{x}), h_0, ..., h_i, \mathbf{e}(\text{"[MASK]"})] \quad (2)$$

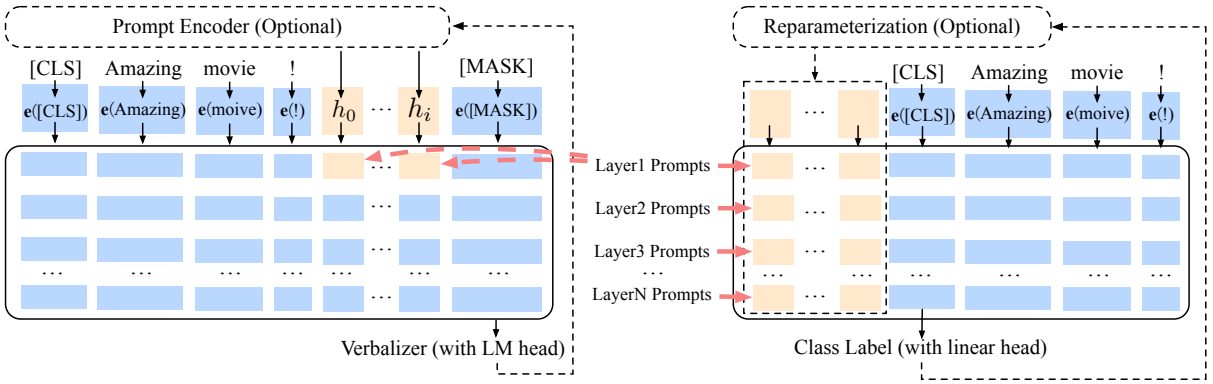
and therefore can be differentially optimized (Cf. Figure 2 (a)). Under the strict constraint that backbone pre-trained models' parameters are frozen, prompt tuning has been proved to have comparable performance to fine-tuning on 10-billion-parameter models in simple NLU tasks (Lester et al., 2021; Kim et al., 2021) and knowledge probing (Liu et al., 2021b).

3 P-Tuning v2

3.1 Lack of Universality

Prompt tuning and P-tuning have been proved quite effective in many NLP applications (Cf. Section 5). Nevertheless, P-tuning is not yet a comprehensive alternative to fine-tuning, considering the following lack of universality.

Lack of universality across scales. Lester et al. (2021) shows that prompt tuning can be comparable to fine-tuning when the model scales to over 10



(a) Lester et al. & P-tuning (Frozen, 10-billion-scale, simple tasks) (b) P-tuning v2 (Frozen, most scales, most tasks)

Figure 2: From Lester et al. (2021) & P-tuning to P-tuning v2. Orange tokens (include h_0, h_i) refer to prompt embeddings we add; blue tokens are embeddings stored or computed by frozen pre-trained language models. Compared to Lester et al. (2021), P-tuning v2 adds trainable continuous prompts to inputs of every transformer layer independently (as prefix-tuning (Li and Liang, 2021) does). Additionally, P-tuning v2 removes verbalizers with LM head and returns to the traditional class labels with ordinary linear head to allow its task-universality.

billion parameters. But for those smaller models (from 100M to 1B), there is a significant discrepancy between performances of prompt tuning and fine-tuning, which significantly limits the applicability of prompt tuning.

Lack of universality across tasks. Though Lester et al. (2021) and P-tuning have shown superiority on NLU benchmarks such as GLUE and SuperGLUE, their effectiveness on another large family of hard sequence NLU tasks (i.e., sequence tagging) is not verified. First, sequence tagging requires predicting a sequence of labels rather than a single label. Second, sequence tagging usually predicts no-actual-meaning labels, which could be challenging to turn into effective verbalizers (Schick and Schütze, 2020). In our experiment (Cf. Section 4.3 and Table 3), we show that Lester et al. (2021) & P-tuning performs poorly on typical sequence tagging tasks compared to fine-tuning.

Considering these challenges, we propose P-tuning v2, which adapts prefix-tuning as a universal solution across scales and NLU tasks.

3.2 Deep Prompt Tuning

Prefix-tuning (Li and Liang, 2021) was originally proposed for natural language generation (NLG) tasks, but we find it very effective for NLU as well. We describe a version of prefix-tuning adapted to NLU.

In (Lester et al., 2021) and P-tuning, continuous prompts are only inserted into the sequence of input embeddings (Cf. Figure 2 (a)) for the transformer's first layer. In the following transformer layers, embeddings at positions where continuous prompts are inserted are **computed** by previous transformer layers, which may lead to two possible optimizing challenges.

1. Limited amount of parameters to tune. Most language models currently can only support a maximum sequence length of 512 (due to the cost of attention's quadratic computational complexity). If we additionally deduct the length of our context (e.g., a sentence to be classified), there is a limited length for us to fill with continuous prompts.

2. Limited stability when tuning with very deep transformers. As the transformer growing deeper, the impact of prompts from the first transformer layer can be unexpected due to many intermediate layers' computation (with nonlinear activation functions), making our optimization not a very smooth one.

In light of the challenges, P-tuning v2 leverages multi-layer prompts (i.e., **deep prompt tuning**) as in prefix-tuning (Li and Liang, 2021) (Cf. Figure 2 (b)), as a major improvement over P-tuning and Lester et al. (2021). Prompts in different layers are added as prefix tokens in the input sequence and are **independent** to each other interlayers (rather than being computed by previous transformer layers). On the one hand, in this way, P-tuning v2 has a larger number of tunable task-specific parameters (from 0.01% to 0.1%-3%) to allow for more per-task capacity, while it is still much smaller than full pre-trained language models; on the other

hand, prompts added to deeper layers (e.g., LayerN Prompts in Figure 2) can have more direct and significant impacts on output predictions with fewer intermediate transformer layers (Cf. Section 4.4).

3.3 Optimization and Implementation

There are also a few helpful optimization and implementation details:

Optimization: Reparameterization. Previous methods leverage the reparameterization function to increase training speed, robustness, and performance (e.g., MLP for prefix-tuning and LSTM for P-tuning). However, for NLU tasks, we discover the benefit of this technique depends on tasks and datasets. For some datasets (e.g., RTE and CoNLL04), MLP reparameterization brings a consistent improvement over embedding; for others, reparameterization may show no effect (e.g., BoolQ), sometimes even worse (e.g., CoNLL12). See our ablation study in Section 4.4.

Optimization: Prompt length. Prompt length plays a central role in the hyper-parameter search of prompt tuning methods. In our experiments, we find different understanding tasks usually achieve their best performance with different prompt lengths, which accords with findings in prefix-tuning (Li and Liang, 2021) where different text generation tasks may have different optimal prompt lengths. See discussions in Section 4.4.

Optimization: Multi-task learning. Multi-task learning is optional for our method but could be quite helpful. On the one hand, the random initialization of continuous prompts brings in difficulties for optimization, which can be alleviated with more training data or task-related unsupervised pre-training (Gu et al., 2021); on the other hand, continuous prompts serve as perfect carriers of task-specific knowledge across tasks and datasets. Our experiment shows that multi-task learning can be a useful complement to P-tuning v2 in some hard sequence tasks, denoted as MPT-2 (Cf. Table 2,3,4).

Implementation: [CLS] and token classification, rather than verbalizers. Verbalizer (Schick and Schütze, 2020) has been a central component of prompt tuning, which turns one-hot class labels into meaningful words to make use of the pre-trained language model head. Despite its potential necessity in a few-shot setting, the verbalizer is not a must in a full-data supervised setting. It hinders the application of prompt tuning to scenarios where we need no-actual-meaning labels and sentence embeddings. Therefore, P-tuning v2 returns to the conventional [CLS] label classification (Cf. Figure 2) paradigm with random-initialized linear heads. See the comparison in Section 4.4.

4 Experiments

4.1 Setup

We conduct extensive experiments over different commonly-used pre-trained models and NLU tasks to verify the effectiveness of P-tuning v2.

Evaluation Setting. In this work, all results of "prompt tuning", "P-tuning", "P-tuning v2", and "Multitask P-tuning v2" are obtained by freezing the parameters of the transformer and only tuning the continuous prompts. Ratios of task-specific parameters (e.g., 0.1%) are derived from comparing continuous prompts' parameters with transformers' parameters. Only results of "fine-tuning" are obtained by tuning transformers' parameters (without using continuous prompts).

Another thing to notice is that our experiments are all conducted in the full-data supervised learning setting rather than few-shot learning, which is important because some properties we leverage (e.g., use class labels with linear heads instead of verbalizers with LM heads) are only likely to work in the supervised setting.

NLU Tasks. First, we include part of datasets from GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) benchmarks to test P-tuning v2's general NLU ability including SST-2, MNLI-m, RTE, BoolQ and CB. More importantly, we introduce a suite of tasks in the form of sequence tagging, which require language model to predict the class of every token in the input sequence, including Named Entity Recognition (CoNLL03 (Sang and De Meulder, 2003), OntoNotes 5.0 (Weischedel et al., 2013) and CoNLL04 (Carreras and Màrquez, 2004)), Extractive Question Answering (SQuAD 1.1 and SQuAD 2.0 (Rajpurkar et al., 2016)) and Semantic Role Labeling (CoNLL05 (Carreras and Màrquez, 2005) and CoNLL12 (Pradhan et al., 2012)).

Pre-trained Models. We include BERT-large (Devlin et al., 2018), RoBERTa-large (Liu et al., 2019), DeBERTa-xlarge (He et al., 2020), GLM-xlarge/xxlarge (Du et al., 2021) for evaluation. They are all bidirectional models designed for NLU

	#Size	GLUE dev						SuperGLUE dev								
		SST-2			MNLI-m			RTE			BoolQ			CB		
		FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2	FT	PT	PT-2
$\text{BERT}_{\text{large}}$	335M	93.2	92.4	**93.6** (+0.4)	**86.6**	75.6	85.8	70.4	53.5	**78.3** (+7.9)	**77.7**	67.2	75.8	**94.6**	80.4	**94.6** (+0.0)
$\text{RoBERTa}_{\text{large}}$	355M	**96.4**	95.3	96.3	90.2	83.8	**90.4** (+0.2)	86.6	58.8	**88.4** (+1.8)	**86.9**	62.3	84.8	98.2	71.4	**100** (+1.8)
$\text{GLM}_{\text{xlarge}}$	2B	-	-	-	-	-	-	**90.3**	85.6	**90.3** (+0.0)	**88.3**	79.7	87.0	**96.4**	76.4	**96.4** (+0.0)
$\text{GLM}_{\text{xxlarge}}$	10B	-	-	-	-	-	-	**93.1**	89.9	**93.1** (+0.0)	88.7	**88.8**	**88.8** (+0.1)	**98.7**	98.2	96.4

Table 1: Results on part of GLUE and SuperGLUE development set (all metrics are Accuracy). P-tuning v2 significantly surpasses P-tuning & Lester et al. (2021) on models smaller than 10B and matches the performance of fine-tuning. (FT: fine-tuning; PT: P-tuning & Lester et al. (2021); PT-2: P-tuning v2).

	#Size	CoNLL03				OntoNotes 5.0				CoNLL04			
		FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2
$\text{BERT}_{\text{large}}$	335M	**92.8**	81.9	90.2	91.0	**89.2**	74.6	86.4	86.3	85.6	73.6	84.5	**86.6** (+1.0)
$\text{RoBERTa}_{\text{large}}$	355M	92.6	86.1	92.4	**92.8** (+0.2)	**89.8**	80.8	89.4	**89.8** (+0.0)	88.8	76.2	87.8	**90.6** (+0.8)
$\text{DeBERTa}_{\text{xlarge}}$	750M	**93.1**	90.2	**93.1** (+0.0)	**93.1** (+0.0)	90.4	85.1	90.4 (+0.0)	**90.5** (+0.1)	89.1	82.4	86.5	**90.1** (+1.0)

Table 2: Results on Named Entity Recognition (NER) test set (all metrics are micro-f1 score). P-tuning v2 is generally comparable to fine-tuning, and multitask P-tuning v2 can bring in a further improvement. (FT: fine-tuning; PT: P-tuning & Lester et al. (2021); PT-2: P-tuning v2; MPT-2: Multi-task P-tuning v2)

purposes, covering a wide range of sizes from about 300M to 10B.

Comparison Methods. We compare our P-tuning v2 (PT-2) with vanilla fine-tuning (FT), P-tuning & Lester et al. (2021) (PT). Additionally, for hard tasks regarding the sequence tagging, we present our results on multi-task P-tuning v2 (MPT-2) with more details presented in Section 4.3.

4.2 P-tuning v2: Across Scales

Table 1 presents P-tuning v2's performances across different model scales. For simple NLU tasks such as SST-2 (single sentence classification), Lester et al. (2021) & P-tuning do not show a significant disadvantage at a smaller scale. But when it comes to complicated challenges such as Natural Language Inference (RTE) and Multiple-choice Question Answering (BoolQ), their performance can be very poor. On the contrary, P-tuning v2 matches fine-tuning performance in all the tasks at a smaller scale. To our surprise, P-tuning v2 significantly outperforms fine-tuning in RTE, especially for BERT.

In terms of larger scales (2B to 10B) with GLM (Du et al., 2021), the gap between P-tuning & Lester et al. (2021) and fine-tuning is gradually narrowed down. On 10B scale, we have a similar observation as is reported in (Lester et al., 2021), that prompt tuning becomes competitive to fine-tuning. However, P-tuning v2 is always comparable to fine-tuning at all scales but with only 0.1% task-specific parameters needed comparing to fine-tuning.

Additionally, we observe that in some datasets, RoBERTa-large has poorer performance than BERT-large. Part of the reason is that we empirically find prompt tuning can be quite sensitive to hyper-parameters, and sometimes the tuning just gets trapped. P-tuning v2 can be more stable and robust during tuning. For more details about hyper-parameters, please refer to our code repository.

4.3 P-tuning v2: Across Tasks

In Section 4.2, we discuss P-tuning v2's consistent, comparable performance to fine-tuning whatever the scales. However, most tasks on GLUE and SuperGLUE are comparatively simple NLU problems. Another important family of hard NLU challenges lies in sequence tagging, which relates to some more high-level NLP applications, including open information extraction, reading comprehension, and so on.

To evaluate P-tuning v2's ability on these hard NLU challenges, we select three typical sequence

	#Size	SQuAD 1.1 dev								SQuAD 2.0 dev							
		FT		PT		PT-2		MPT-2		FT		PT		PT-2		MPT-2	
		EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
$\text{BERT}_{\text{large}}$	335M	**84.2**	**91.1**	1.0	8.5	77.8	86.0	82.3	89.6	**78.7**	**81.9**	50.2	50.2	69.7	73.5	72.7	75.9
$\text{RoBERTa}_{\text{large}}$	355M	**88.9**	**94.6**	1.2	12.0	88.1	94.1	88.0	94.1	**86.5**	**89.4**	50.2	50.2	82.1	85.5	83.4	86.7
$\text{DeBERTa}_{\text{xlarge}}$	750M	90.1	95.5	2.4	19.0	**90.4** (+0.3)	**95.7** (+0.2)	89.6	95.4	88.3	91.1	50.2	50.2	**88.4** (+0.1)	**91.1** (+0.0)	88.1	90.8

Table 3: Results on Question Answering (Extractive QA). Prompt tuning & P-tuning performs extremely poor on question answering, while P-tuning v2's performance is generally reasonable, and can be better than fine-tuning with DeBERTa-xlarge. (FT: fine-tuning; PT: P-tuning & Lester et al. (2021); PT-2: P-tuning v2; MPT-2: Multi-task P-tuning v2)

	#Size	CoNLL12				CoNLL05 WSJ				CoNLL05 Brown			
		FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2	FT	PT	PT-2	MPT-2
$\text{BERT}_{\text{large}}$	335M	84.9	64.5.	83.2	**85.1** (+0.2)	**88.5**	76.0	86.3	**88.5** (+0.0)	82.7	70.0	80.7	**83.1** (+0.4)
$\text{RoBERTa}_{\text{large}}$	355M	**86.5**	67.2	84.6	86.2	**90.2**	76.8	88.3	90.0	85.6	70.7	84.2	**85.7** (+0.1)
$\text{DeBERTa}_{\text{xlarge}}$	750M	86.5	74.1	85.7	**87.1** (+0.6)	**91.2**	82.3	90.6	91.1	86.9	77.7	86.3	**87.3** (+0.4)

Table 4: Results on Semantic Role Labeling (SRL). P-tuning v2 shows a consistent improvement over Lester et al. (2021) & P-tuning on SRL. (FT: fine-tuning; PT: P-tuning & Lester et al. (2021); PT-2: P-tuning v2; MPT-2: Multi-task P-tuning v2)

	SST-2	RTE	BoolQ	CB
CLS & linear head	96.3	88.4	84.8	96.4
Verbalizer & LM head	95.8	86.6	84.6	94.6

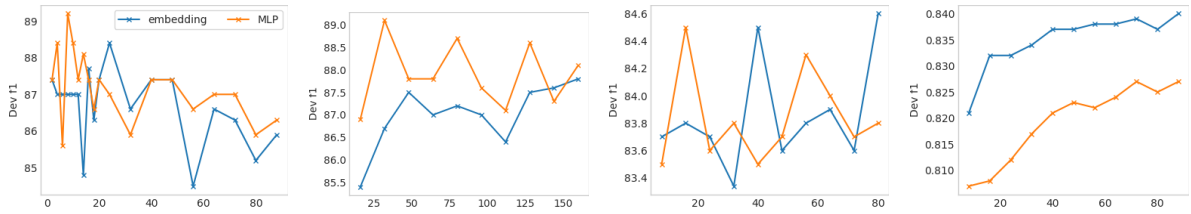
Table 5: Comparison between [CLS] label with linear head and verbalizer with LM head on RoBERTa-large.

tagging tasks: Name Entity Recognition, Extractive Question Answering (QA), and Semantic Role Labeling (SRL), altogether eight datasets.

Name entity recognition (NER). NER aims to predict all spans of words that represent some given classes of entity with a sentence. We adopted CoNLL03 (Sang and De Meulder, 2003), OntoNotes 5.0 (Weischedel et al., 2013) and CoNLL04 (Carreras and Màrquez, 2004). For CoNLL03 and CoNLL04, we trained our model on the standard train-develop-test split. For OntoNotes 5.0, we use the same train, develop, test split as (Xu et al., 2021b). All the datasets are labeled in IOB2 format. We use sequence tagging to solve NER tasks by assigning labels marking the beginning and inside some classes of entity. The language models generate a representation for each token, and we use a linear classifier to predict the labels. We use the official scripts to evaluate the results. For the multi-task setting, we combine the training set of the three datasets for pre-training. We use different linear classifiers for each dataset while sharing the continuous prompts.

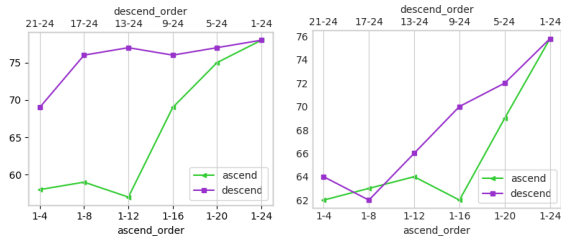
(Extractive) Question Answering (QA). Extractive QA is designed to extract the answer from the context given the context and a question. We use SQuAD (Rajpurkar et al., 2016) 1.1 and 2.0, in which each answer is within a continuous span of the context. Following tradition, we formulate the problem as sequence tagging by assigning one of the two labels: 'start' or 'end' to each token and at last selecting the span of the most confident start-end pair as the extracted answer. If the probability of the most confident pair is lower than a threshold, the model will assume the question unanswerable. For the multi-task setting, our training set for pre-training combines the training sets of SQuAD 1.1 and 2.0. When pre-training, we assume that all the questions, regardless of their origin, are possibly unanswerable.

Semantic Role Labeling (SRL). SRL assigns labels to words or phrases in a sentence that indicates their semantic role in the sentence. We evaluate P-tuning v2 on CoNLL05 (Carreras and Màrquez, 2005) and CoNLL12 (Pradhan et al., 2012). Since a sentence can have multiple verbs, we add the target verb token to the end of each sentence to help recognize which verb is used for prediction. We classify each word using a linear classifier based on the cor-



(a) NLI: RTE (b) NER: CoNLL04 (c) MQA: BoolQ (d) SRL: CoNLL12

Figure 3: Ablation study on prompt length and reparamerization using RoBERTa-large. The conclusion can be very different given certain NLU task and dataset. (MQA: Multiple-choice QA)



(a) RTE (b) BoolQ

Figure 4: Ablation study on prompt depth using BERT-large. "[x-y]" refers to the layer-interval we add continuous prompts (e.g., "21-24" means we are add prompts to transformer layers from 21 to 24). Same amount of continuous prompts added to deeper transformer layers (i.e., more close to the output layer) can yield a better performance than those added to beginning layers.

responding semantic role representation. For multi-task setting, the pre-train training set is a combination of the training set of CoNLL05 (Carreras and Màrquez, 2005), CoNLL12 (Pradhan et al., 2012) and propbank-release (a common extend data used for training SRL). The multi-task training strategy is similar to NER.

Results. From Table 2,3,4, we observe that P-tuning v2 can be generally comparable to fine-tuning on all tasks. P-tuning & Lester et al. (2021) show much poorer performance, especially on QA, which might be the most difficult challenge of the three tasks. We also notice that there are some abnormal results presented in SQuAD 2.0 (BERT/RoBERTa/DeBERTa show the same performance using Lester et al. (2021) & P-tuning). This is probably because compared to SQuAD 1.1, SQuAD 2.0 contains unanswerable questions, and the Lester et al. (2021) & P-tuning could possibly get the trivial solution.

Multi-task P-tuning v2 generally brings in significant improvement overall tasks except for QA (which might still be the consequence of mixing all-answerable SQuAD 1.1 and not-answerable SQuAD 2.0), which implies that randomly initialized prompts' potential is under-explored.

4.4 Ablation Study

We study some important hyper-parameters and architecture designs that may play a central role in P-tuning v2.

Prompt depth. The main difference between Lester et al. (2021) & P-tuning and P-tuning v2 is the multi-layer continuous prompts we introduce. Intuitively, due to the many non-linear activation functions in intermediate transformer layers, the deeper the transformer layer a prompt locates in, the more direct its impact on the output predictions. To verify its exact influence, given a certain number of k to add prompts, we select k layers in both ascending and descending order to add prompts as prefix tokens; for the rest layers, we change their attention masks for disallowing their prefix prompts to involve in the computation.

As shown in Figure 4, with the same amount of parameters (i.e., num of transformer layers to add prompts), adding them in the descending order is always better than in the ascending order. In the RTE case, only adding prompts to layers 17-24 can yield a very close performance to all layers, further cutting down parameters we may need to tune for matching fine-tuning.

Embedding v.s. MLP reparameterization. In both prefix-tuning (Li and Liang, 2021) and P-tuning (Liu et al., 2021b), authors discover the reparameterization to be useful in improving training speed, robustness and performance. However, we conduct experiments to show that the reparameterization effect is inconsistent across different NLU tasks and datasets.

As shown in Figure 3, in RTE and CoNLL04, MLP reparameterization generally indicates better performance than embedding for almost all prompt lengths. However, in BoolQ, MLP and embed-

ding's results are competitive; in CoNLL12, the embedding consistently outperforms MLP.

Prompt Length. Prompt length is yet another influential hyper-parameter for P-tuning v2, and its optimal value varies from task to task. From Figure 3, we observe that for simple NLU tasks, usually, a shorter prompt is enough for the best performance; for hard sequence tasks, usually, a longer prompt than 100 would be helpful.

We also discover that reparameterization has a close bond with optimal prompt length. For example, in RTE, CoNLL04, and BoolQ, MLP reparameterization achieves its optimal result earlier than embedding. This conclusion may contribute some thoughts on P-tuning's optimization properties.

Verbalizer with LM head v.s. [CLS] label with linear head. Verbalizer with LM head has been a central component in previous prompt tuning approaches. However, for P-tuning v2 in a supervised setting, it is affordable to tune a linear head with about several thousand parameters. We present our comparison in Table 5, where we keep other hyper-parameters and only change [CLS] label with linear head to verbalizer with LM head. Here, for simplicity, we use "true" and "false" for SST-2, RTE and BoolQ; "true", "false" and "neutral" for CB. Results indicate that there is no significant difference between performances of verbalizer and [CLS].

5 Related Work

Pre-trained Language Models. Self-supervised (Liu et al., 2020) pre-trained language models (Han et al., 2021a) has become the backbone of natural language processing. From early stage when GPT (Radford et al., 2019), BERT (Devlin et al., 2018), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019) has limited amount of parameters (less than 350M), the advent of T5 (Raffel et al., 2019) and GPT-3 (Brown et al., 2020) boosts the development of giant language models with billion and even trillions of parameters.

Prompting. Prompting (Liu et al., 2021a) refers to leverage special templates in the input context to aid the language model prediction with respect to both understanding and generation. Recently, thanks to the success of GPT-3 (Brown et al., 2020), various prompting strategies including discrete natural language prompt (Shin et al., 2020; Gao et al., 2020), continuous prompts (Liu et al., 2021b; Li and Liang, 2021; Lester et al., 2021; Qin and Eisner, 2021; Zhong et al., 2021), tuning bias (Logan IV et al., 2021) and many other prompting strategies have appeared.

Prompting's advantage and effectiveness in a wide range of NLP applications has been verified in recent literature, including text classification (Hu et al., 2021; Min et al., 2021; Sun et al., 2021; Li et al., 2021; Zhang et al., 2021b), entity typing (Ding et al., 2021), few-shot learning (Zheng et al., 2021; Xu et al., 2021a; Zhao et al., 2021; Gu et al., 2021; Zhang et al., 2021a), relation extraction (Chen et al., 2021a; Han et al., 2021b; Sainz et al., 2021), knowledge probing (Zhong et al., 2021), named entity recognition (Chen et al., 2021b), machine translation (Tan et al., 2021; Wang et al., 2021b) and dialogue system (Wang et al., 2021a).

In this work, we are especially interested in scaling prompting methods to smaller models and hard sequence NLU tasks.

6 Conclusion

We present P-tuning v2, a prompting method comparable to fine-tuning universally across scales and tasks. P-tuning v2 is not a conceptually new approach but an optimized and adapted prefix-tuning and deep prompt tuning to NLU challenges. P-tuning v2 shows consistent improvements for models ranging from 330M to 10B and outperforms Lester et al. (2021) & P-tuning on hard sequence tasks such as sequence tagging by a large margin. P-tuning v2 could be a comprehensive alternative for fine-tuning and a strong baseline for future work.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*

(CoNLL-2005), pages 152–164, Ann Arbor, Michigan. Association for Computational Linguistics.

Xiang Chen, Xin Xie, Ningyu Zhang, Jiahuan Yan, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021a. Adaprompt: Adaptive prompt-based finetuning for relation extraction. *arXiv preprint arXiv:2104.07650*.

Xiang Chen, Ningyu Zhang, Lei Li, Xin Xie, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021b. Lightner: A lightweight generative framework with prompt-guided attention for low-resource ner. *arXiv preprint arXiv:2109.00720*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv e-prints*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. *arXiv preprint arXiv:2108.10604*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2021. All nlp tasks are generation tasks: A general pretraining framework. *arXiv preprint arXiv:2103.10360*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Yuxian Gu, Xu Han, Zhiyuan Liu, and Minlie Huang. 2021. Ppt: Pre-trained prompt tuning for few-shot learning. *arXiv preprint arXiv:2109.04332*.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Liang Zhang, Wentao Han, Minlie Huang, et al. 2021a. Pre-trained models: Past, present and future. *AI Open*.

Xu Han, Weilin Zhao, Ning Ding, Zhiyuan Liu, and Maosong Sun. 2021b. Ptr: Prompt tuning with rules for text classification. *arXiv preprint arXiv:2105.11259*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Juanzi Li, and Maosong Sun. 2021. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *arXiv preprint arXiv:2108.02035*.

Boseop Kim, HyoungSeok Kim, Sang-Woo Lee, Gichang Lee, Donghyun Kwak, Dong Hyeon Jeon, Sunghyun Park, Sungju Kim, Seonhoon Kim, Dongpil Seo, et al. 2021. What changes can large-scale language models bring? intensive study on hyperclova: Billions-scale korean generative pretrained transformers. *arXiv preprint arXiv:2109.04650*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Chengxi Li, Feiyu Gao, Jiajun Bu, Lu Xu, Xiang Chen, Yu Gu, Zirui Shao, Qi Zheng, Ningyu Zhang, Yongpan Wang, et al. 2021. Sentiprompt: Sentiment knowledge enhanced prompt-tuning for aspect-based sentiment analysis. *arXiv preprint arXiv:2109.08306*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Xiao Liu, Fanjin Zhang, Zhenyu Hou, Zhaoyu Wang, Li Mian, Jing Zhang, and Jie Tang. 2020. Self-supervised learning: Generative or contrastive. *arXiv preprint arXiv:2006.08218*, 1(2).

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *arXiv: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. *arXiv e-prints*.

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. *arXiv preprint arXiv:1907.11692*.

Robert L Logan IV, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models. *arXiv preprint arXiv:2106.13353*.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification. *arXiv preprint arXiv:2108.04106*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Joint Conference on EMNLP and CoNLL - Shared Task*, pages

1–40, Jeju Island, Korea. Association for Computational Linguistics.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. *arXiv preprint arXiv:2104.06599*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 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. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Oscar Sainz, Oier Lopez de Lacalle, Gorka Labaka, Ander Barrena, and Eneko Agirre. 2021. Label verbalization and entailment for effective zero- and few-shot relation extraction. *arXiv preprint arXiv:2109.03659*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

Yi Sun, Yu Zheng, Chao Hao, and Hangping Qiu. 2021. Nsp-bert: A prompt-based zero-shot learner through an original pre-training task–next sentence prediction. *arXiv preprint arXiv:2109.03564*.

Zhixing Tan, Xiangwen Zhang, Shuo Wang, and Yang Liu. 2021. Msp: Multi-stage prompting for making pre-trained language models better translators. *arXiv preprint arXiv:2110.06609*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *NeurIPS 2019*, pages 3261–3275.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *arXiv e-prints*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv e-prints*.

Hongru Wang, Mingyu Cui, Zimo Zhou, Gabriel Pui Cheong Fung, and Kam-Fai Wong. 2021a. Topicrefine: Joint topic prediction and dialogue response generation for multi-turn end-to-end dialogue system. *arXiv preprint arXiv:2109.05187*.

Shuo Wang, Zhaopeng Tu, Zhixing Tan, Wenxuan Wang, Maosong Sun, and Yang Liu. 2021b. Language models are good translators. *arXiv preprint arXiv:2106.13627*.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2013. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23.

Liang Xu, Xiaojing Lu, Chenyang Yuan, Xuanwei Zhang, Hu Yuan, Huilin Xu, Guoao Wei, Xiang Pan, and Hai Hu. 2021a. Fewclue: A chinese few-shot learning evaluation benchmark. *arXiv preprint arXiv:2107.07498*.

Lu Xu, Zhanming Jie, Wei Lu, and Lidong Bing. 2021b. Better feature integration for named entity recognition. *arXiv preprint arXiv:2104.05316*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021a. Differentiable prompt makes pre-trained language models better few-shot learners. *arXiv preprint arXiv:2108.13161*.

Wenxuan Zhang, Yang Deng, Xin Li, Yifei Yuan, Lidong Bing, and Wai Lam. 2021b. Aspect sentiment quad prediction as paraphrase generation. *arXiv preprint arXiv:2110.00796*.

Mengjie Zhao, Yi Zhu, Ehsan Shareghi, Ivan Vulić, Roi Reichart, Anna Korhonen, and Hinrich Schütze. 2021. A closer look at few-shot crosslingual transfer: The choice of shots matters. 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 5751–5767.

Yanan Zheng, Jing Zhou, Yujie Qian, Ming Ding, Jian Li, Ruslan Salakhutdinov, Jie Tang, Sebastian Ruder, and Zhilin Yang. 2021. Fewnlu: Benchmarking state-of-the-art methods for few-shot natural language understanding. *arXiv preprint arXiv:2109.12742*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. *arXiv preprint arXiv:2104.05240*.