Mixture of Virtual Prompts: Efficient Fine-tuning with Virtual Prompts for Natural Language Understanding

Jiaao Chen jchen896@gatech.edu
William Held wheld3@gatech.edu
Diyi Yang diyi.yang@cc.gatech.edu

Abstract

Fine-tuning large-scale pre-trained language models tasks has become the de facto standard for downstream NLP tasks. However, fine-tuning all parameters of these large models brings expensive computation and storage cost. In this work, inspired by model prompting, we propose Mixture of Virtual Prompts adaptation (MVP adaptation). We first learn virtual prompts downstream tasks to inject task or language specific knowledge, while the pre-trained models are frozen to provide general knowledge. We extend virtual prompting to deal with varying complexity inputs, learning a set of virtual prompts with distinct capacities to learn different levels of knowledge, and dynamically aggregating them to adapt for each individual sequence. Experiments conducted on 5 natural language understanding tasks demonstrate that our methods could achieve comparable performances relative to the full model. Importantly, we show the superiority of MVP in terms of linguistic variation settings which allows efficient adaption to different dialects. We outperform strong efficient finetuning baselines using only 0.03% of its parameters.

1 Introduction

Recently, large pre-trained language models (PLMs) have obtained great success and, through fine-tuning on task/dialect-specific labeled data, achieved state-of-the-art performances across a wide variety of downstream natural language processing tasks (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019; Joshi et al., 2019; Sun et al., 2019; Clark et al., 2019; Lewis et al., 2020; Bao et al., 2020; He et al., 2020; Raffel et al., 2020; Ziems et al., 2022a). However, fine-tuning large models and storing them separately for different tasks/dialects are inherently expensive in terms of both training time and storage overhead (*e.g.*, 355M parameters for RoBERTa (Liu et al., 2019) and 175B parameters for GPT-3 (Brown et al., 2020)), making them difficult to be deployed in real-world NLP systems (which are usually composed of multiple tasks / multiple language variations).

In order to improve the efficiency, different sets of approaches have been proposed where only a small number of parameters are tuned while the pre-trained models are frozen to adapt general knowledge in PLMs to specific down-stream tasks (e.g., sentiment classification) or language variations (e.g., African American Vernacular English, AAVE) (Houlsby et al., 2019a; Pfeiffer et al., 2021; Li and Liang, 2021; Brown et al., 2020; Lester et al., 2021; Schick and Schütze, 2021; Ziems et al., 2022a). Specifically, Houlsby et al. (2019a) introduced extra parameters that needed to be learned via adding extra linear layers to every transformer layer (Houlsby et al., 2019a; Pfeiffer et al., 2021).

To further enable a smaller set of task-specific parameters, well-designed and generated prompts are appended to the input and learned (Brown et al., 2020; Shin et al., 2020a; Lester et al., 2021; Hambardzumyan et al., 2021; Le Scao and Rush, 2021), or be viewed as activation matrix and added per transformer layer (Li and Liang, 2021). While achieving great success, the prompt-based methods usually require carefully-designed modules to generate prompts (Brown et al., 2020; Shin et al., 2020a; Hambardzumyan et al., 2021; Liu et al., 2021b). Moreover, the added prompts are often fixed for all the data points (Hambardzumyan et al., 2021; Li and Liang, 2021), treating each data point equally despite its unique characteristics. We argue that these prompts might need to be empowered with *different capacities* to accomplish specific tasks/dialects depending on the complexities of specific input sentences and the differences from standard English.

In this paper, we introduce **M**ixture of **V**irtual **P**rompts adaptation (MVP adaptation), where a set of virtual prompts are appended to the input

of a frozen PLM that is shared across different tasks/languages variations to provide general information. Virtual prompts with different capacities are learned jointly and dynamically aggregated based on the input sequence, which composes knowledge of different capacities for individual data points. To demonstrate the effectiveness of our proposed methods, we compare the proposed approach with model fine-tuning and automated-prompt-based tuning methods on 5 single-sentence natural language understanding tasks. By only adding less than 0.03% parameters, our MVP adaptation achieves comparable performance (around 97%) to full model fine-tuning. Importantly, we show the superiority of our MVP in terms of dialect adaptations by adapting models trained on standard English to dialects, where all the dialects share one base model trained on standard English while only several virtual prompts are leaned to make adaptations.

Our contributions in this paper are as follows: (1) We propose using multiple end-to-end learned virtual prompts to adapt large-scale language models to natural language understanding tasks and language variations. (2) A novel method jointly learn and dynamically combine virtual prompts to allow adaptation to different data points. (3) Experiments conducted on five natural language understanding tasks show that MVP provides a strong efficiency to performance trade-off compared to several competitive baselines on parameter-efficient fine-tuning. (4) Experiments conducted on a dialectal benchmark which highlights the ability of MVP to enable models tailored to specific linguistic variation to run in parallel.

2 Related Work

2.1 Prompts for In-context Learning

Many extremely large language models such as GPT-3 (Brown et al., 2020) leverage in-context prompts as the sole mechanism to efficiently adapt behavior. However such prompts are typically hand-crafted, giving rise to unstable empirical results with large variance (Gao et al., 2020; Liu et al., 2021a) as well as subjective heuristics. Different strategies are proposed to generate high-quality prompts by focusing on prompt format, example selection and even order of examples (Liu et al., 2021a; Zhao et al., 2021; Lu et al., 2021; Jiang et al., 2020a). These prompt engineering strategies have been explored for both natural language generation and understanding tasks in prior works (Sun and Lai, 2020; Mao et al., 2020; Cai et al., 2018; Le Scao and Rush, 2021).

Shin et al. (2020b) develops AutoPrompt, which creates prompts for different tasks based on gradient-guided search. Chen et al. (2021) introduces AdaPrompt, which constructs customized prompts for relation extraction by exploiting the knowledge within large pre-trained model. Jiang et al. (2020b) proposes to probe the knowledge in language model by generating better prompts using mining-based/paraphrasing-based methods. Our work aims to replicate the benefits of multiple-example prompting, inspired by these prompts-based learning. However MVP directly learns virtual prompts with different capacities in an end-to-end manner without the need to design or generate any new prompts.

2.2 Parameter-efficient Fine-tuning

As pre-trained models increase in size, storing fine-tuned models becomes exceedingly expensive and fine-tuning becomes infeasible for those without extremely high compute resources. A growing body of research has been devoted to find parameter-efficient alternatives for adapting large-scale pre-trained models with reduced memory and storage cost. Houlsby et al. (2019b) propose to adapt large models using bottleneck layers (with skip-connection) between each layer. This idea has been extended in many domains (Stickland and Murray, 2019; Pfeiffer et al., 2020a; Rebuffi et al., 2017; Lin et al., 2020). Other works have aimed to avoid introducing additional parameters by identifying and training only a subset of all model parameters (Zhao et al., 2020; Guo et al., 2020; Mallya et al., 2018; Radiya-Dixit and Wang, 2020; Sung et al., 2021). Recent work also explored injecting trainable rank decomposition matrices into each layer (Hu et al., 2021; He et al., 2021).

Li and Liang (2021) introduces prefix-tuning that prepends a set of prefixes to autoregressive language models or prepends prefixes for both encoder and decoder. The prefix parameters are updated while the pre-trained parameters are fixed. Empirically, the resulting model matches the performance of full fine-tuning with only 0.1% parameters for various generation tasks. Lester et al. (2021) proposes a similar method, but only adds virtual tokens at the embedding layer of very large scale models. We extend the idea of prompting by using multiple

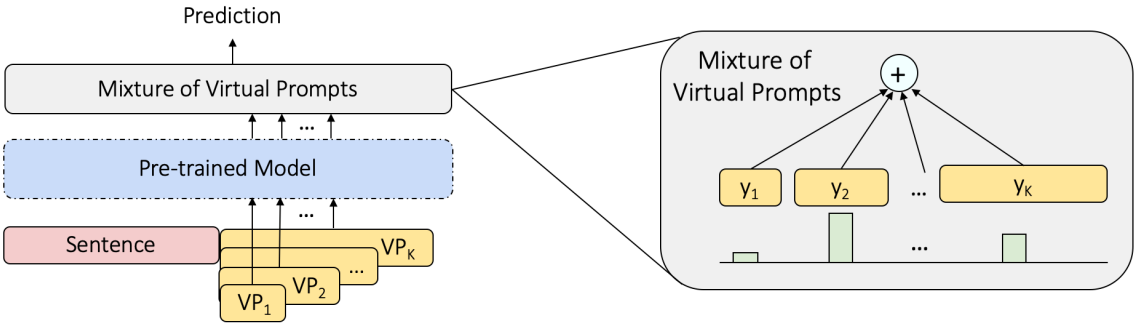

Figure 1: Overall Architecture for Mixture of Virtual Prompts (MVP). During training, the parameters in pre-trained model is fixed. Virtual prompts of different length, $VP_{\{1:K\}}$, are appended to the input sentence and encoded separately first. Then the encoded results y_i from different virtual prompts are aggregated to make the predictions.

virtual prompts to allow each virtual prompt can learn its specialization, which can be used to better adapt to each individual data point.

3 Methods

To more efficiently fine-tune large pre-trained language models on down-stream tasks, we introduce virtual prompts in Section 3.1 where we learn virtual prompts with limited length. Since different data points exhibit different complexities, which might require different model capacities, we further propose the **M**ixture of **V**irtual **P**rompts adaptation (MVP adaptation) in Section 3.2 to dynamically aggregate task knowledge from virtual prompts with different capacities for different inputs. The overall model architecture is shown in Figure 1.

3.1 Virtual Prompts

Our virtual prompt method trains only continuous embeddings and a final classifier [1] to adapt a shared frozen model with parameters θ^* to task knowledge. Specifically, after the embedding layer in the pre-trained language model, the input sentences are first embedded into a hidden matrix $H \in R^{N\times D}$, where D is the number of hidden dimensions:

$$H = f_0(s;\theta^*) \quad (1)$$

We then append a randomly initialized virtual prompts matrix $V \in R^{L\times D}$ to H, and continue the forward passing:

$$[H^M; V^M] = f_{1:M}([H;V];\theta^*) \quad (2)$$

During training, we only update this virtual prompt matrix V to maximize the overall log-likelihood objective, leaving θ^* fixed:

$$max_{V,g} \sum \log p(y|g(f_{1:M}([H;V];\theta^*)) \quad (3)$$

Here, $\tilde{y} = g([H^M; V^M])$ is the final classifier. After training, only the parameters V and $g(.)$ need to be stored for specific tasks while θ^* in pre-trained models will be shared across all the tasks. Detailed analyses are discussed in Section 3.3.

Compared to previous prompt-based tuning (Shin et al., 2020a; Hambardzumyan et al., 2021; Jiang et al., 2020a) where prompts are either specially designed or found through adversarial process, our virtual prompts are initialized and directly learned in an end-to-end fashion during training. Our virtual prompts extends prior work on prefix-tuning (Li and Liang, 2021) and prompt-tuning (Lester et al., 2021).

3.2 Mixture of Virtual Prompts

While updating virtual prompts can effectively bring in task-specific knowledge, work on hard prompting highlights that different levels of knowledge might be useful for input sentences with various complexities (Xin et al., 2020; Zhou et al., 2020). In order to allow virtual prompts adapt to different data points more strategically and more specifically, we introduce **M**ixture of **V**irtual **P**rompts (MVP) based on the virtual prompts where virtual prompts of different lengths are introduced and learned simultaneously, and are dynamically aggregated based on the input sentence. In MVP, a series of virtual prompts $\{V_1, ..., V_K\}$ with different lengths (suppose V_1 is the virtual prompts of length L_1, the length for V_i is set $\frac{L_1}{i}$.) are first appended separately to the hidden sentence matrix

[1] We omit the updates of the top classifiers for simplicity in the following sections.

H to obtain the individual predictions:

$$[H^M; V_i^M] = f_{1:M}([H; V_i]; \theta^*), i \in [1, K] \quad (4)$$

$$\tilde{y}_i = g([H^M; V_i^M]), \quad i \in [1, K] \quad (5)$$

Predictions from models with different virtual prompts are then combined to get the final output:

$$\tilde{y} = \sum_{i=1}^{K} G([H^M; V_i^M]) \cdot \tilde{y}_i, \quad (6)$$

$V_{1:K}$ will then be updated to minimize the supervised objective function:

$$\mathcal{L} = -y \log(\tilde{y}) \quad (7)$$

Here $G(.)$ is a gating function to assign soft weights α_i to virtual prompts V_i for one given sentence based on the encoded representations V_i^M from the last layer of the pre-trained model:

$$v_i^M = \text{pooling}(V_i^M) \quad (8)$$

$$u_i = \tanh(W_V v_i^M + b_V) \quad (9)$$

$$\alpha_i = \frac{\exp(u_i^\intercal u_g)}{\sum_i \exp(u_i^\intercal u_g)} \quad (10)$$

where W_v, b_V are parameters and u_g is a randomly initialized global vector, the pooling function transform the $V_i^M \in R^{L \times D}$ to vector $v_i^M \in R^D$ through averaging over L.

3.2.1 Balancing Utilization in MVP

Our empirical results show that the gating function tends to converge to a state where it always assign higher weights to the same virtual prompts. This imbalance phenomenon is self-reinforced because the favored virtual prompts are updated more rapidly during the training stage and thus receiving even higher weights from the gating function (Eigen et al., 2013; Bengio et al., 2016; Shazeer et al., 2017; Lin et al., 2019).

To balance utilization of different virtual prompts, we add two additional constraints to regularize the virtual prompts and the gating function: (i) each virtual prompt should learn its own specialization and capture different information, (ii) all virtual prompts should receive equal importance. For (i), we directly minimize the pairwise cosine similarities between hidden representations of different virtual prompts:

$$\mathcal{L}_c = \sum_{i,j \in [1,k]} \frac{v_i^{M\intercal} v_j^M}{||v_i^M|| \cdot ||v_j^M||} \quad (11)$$

Methods	# of Parameters to Store
Full fine-tuning	$O(NM)$
Adapters	$O(M + NN'DD)$
Prefix	$O(M + NN'LD)$
Virtual Prompts	$O(M + NLD)$
MVP	$O(M + NKLD)$

Table 1: The number of parameters required for finetuning and storage for different tuning approaches for N tasks. A pre-trained language model with M parameters is used as the base model. N' is the number of layers in language models (usually 12 or 24). D is the hidden dimensions (usually 768 or 1024). L is the length of prefix or virtual prompts. Here, K is the number of virtual prompts in MVP.

To deal with (ii), inspired by Shazeer et al. (2017), we define the importance score of one virtual prompt within a batch as the sum of weights assigned by the gating function in a batch of sentences as follows:

$$I_i = \sum_j G([H_j^M; V_i^M]) \quad (12)$$

where i means the i-th virtual prompts and j means the j-th sentence in a batch. During training, we minimize the variance of the importance scores across different virtual prompts:

$$\mathcal{L}_v = \delta^2(I_{1:K}) \quad (13)$$

3.2.2 Training Objective

The overall learning objective of MVP combines the supervised loss and regularization terms:

$$\mathcal{L}_{MVP} = \mathcal{L} + \gamma \mathcal{L}_c + \beta \mathcal{L}_v \quad (14)$$

where γ and β are hyper-parameters to control the magnitude of regularization.

3.3 Model Complexity Analysis

Here we discuss the number of parameters required for finetuning and storage for MVP. The comparison with different baseline approaches is shown in Table 1. Suppose we are building NLP systems for N different tasks using a large pre-trained language model (N' layers and D hidden dimensions, with a total M parameters) as the base model.

For the *full fine-tuning* where the whole language model is fine-tuned separately for different tasks, the number of parameters needed to be stored grows linearly (require storing at least $O(NM)$ parameters). When N becomes large in NLP systems, it takes significant expensive space cost for

the systems. To this end, methods like *adapters* (Houlsby et al., 2019b) and *prefix* (Li and Liang, 2021) freeze the pre-trained parameters for different tasks while only updating a small number of extra parameters added through linear layers or prefix matrix. However, the size often grows linearly with the number of layers and hidden dimension in pre-trained models. Compared to *adapters* and *prefix*, our *MVP* only requires $O(KLD)$ parameters for a single task where K is the number of prompts and usually smaller than 5, which significantly alleviate the storage cost. In the meanwhile, since the pre-trained model is frozen during training, the training speed of *MVP* also increases (roughly 3 times faster) compared to *full fine-tuning*.

4 Experimental Setup

4.1 Datasets

We evaluate our methods using a wide range of natural language understanding tasks including:

Multiple Sentences Inference: Multi-Genre Natural Language Inference (MNLI) (Wang et al., 2018) which classifies the relationships between two sentences into entailment, contradiction, or neutral, and Question Natural Language Inference (QNLI) (Wang et al., 2018) which predict whether a given sentence is the correct answer to a given question.

Sentiment Analysis: Stanford Sentiment Treebank (SST-2) (Wang et al., 2018) which predicts the sentiment of movie reviews to be positive or negative.

Topic Classification: AG News (Zhang et al., 2015) which predict the topics for given news, and Yahoo! Answers (Chang et al., 2008) which classifies the main topics of online question-answer interactions.

Dialect Adaptation: Finally, to assess the ability of MVP to model linguistic variation we evaluate on aligned African American English-like (AAE) versions of SST-2, MNLI, and QNLI using the splits released in the Vernacular Language Understanding Benchmark (Ziems et al., 2022b).

Accuracy was used as the evaluation metric across all the datasets. The dataset statistics including the task type, the number of classes and the detailed data split are presented in Table 2.

4.2 Baselines

We compare our MVP with several state-of-the-art baselines in efficient fine-tuning as follows:

RoBERTa [2] (Liu et al., 2019) is one of the state-of-the-art pre-trained language model for natural language understanding. In this work, we use off-the-shelf RoBERTa as our base model for all the methods. For downstream tasks, all the model parameters are fine-tuned.

Classifier-only where only the top classifier head is fine-tuned while the whole pre-trained RoBERTa encoder is fixed during training.

Adapter [3] (Houlsby et al., 2019a) adds adapter modules which consist of linear layers and are added to each transformer layer twice, with one after the projection following multi-headed attention and another after the two feed-forward layers.

Prefix [4] (Li and Liang, 2021) is a method proposed for generation tasks which optimizes a set of small continuous task-specific vectors prepended to every transformer layer for each task.

4.3 Model Settings

We used the pre-trained RoBERTa-base model (Liu et al., 2019) to initialize all the methods. Note that our MVP method is agnostic to other types of pre-trained language models as well. We followed Liu et al. (2019) to set the linear decay scheduler with a warmup ratio of 0.06 for training. The batch size was 64. The maximum learning rate was selected from $\{1e-4, 5e-4, 1e-3, 2e-3\}$ and the maximum number of training epochs was set to be either 5 or 10. The maximum length of virtual prompts was set $L = 50$ after a grid search from $\{5, 10, 50, 100, 120\}$. The number of virtual prompts was set $K = 2$ after a grid search from $\{2, 3, 4, 5\}$. The weights γ and β in our objective function were 0.01 and 0.1. All the experiments were performed using 8 NVIDIA Tesla V100.

5 Experimental Results

5.1 Downstream Task Evaluation

Table 3 describes the accuracy of different models on the 5 standard English benchmark datasets. We observe that, compared to *RoBERTa-base*, *Classifier-only* where only the top classification head is fine-tuned only achieves 71.40 average performances as the whole encoder is not learned from specific tasks, and *Adapter* that used 4% of its

[2] https://huggingface.co/transformers/model_doc/roberta.html

[3] https://adapterhub.ml/

[4] https://github.com/XiangLi1999/PrefixTuning

Dataset	Task	# Classes	# Train	# Test
MNLI	Natural Language Inference	3	393k	20k
QNLI	Natural Language Inference (QA)	2	105k	5.4k
SST-2	Sentiment Classification	2	67k	1.8k
AG News	News Topic Classification	4	120k	7.6k
Yahoo! Answer	QA Topic Classification	10	1400k	60k

Table 2: Dataset statistics for 5 benchmark datasets including the task type, the number of classes, as well as the size of training and testing set.

Method	MNLI	QNLI	SST-2	AG News	Yahoo!	Average	% Param.
RoBERTa-base	87.6	92.8	94.8	87.2	76.8	87.84	-
Classifier-only	55.4	71.9	83.6	78.5	67.6	71.40	0.01%
Adapter	85.2	91.6	94.2	86.3	75.6	86.60	4.00%
Adapter	82.8	88.2	92.0	83.4	73.6	84.04	0.08%
Prefix	82.1	89.6	93.4	84.5	74.4	84.80	0.06%
Prefix	82.0	89.0	93.3	83.7	74.5	84.43	0.03%
MVP †	**83.8**	**90.4**	**94.0**	**84.9**	**75.4**	**85.72**	0.03%
Δ comp. RoBERTa	*95.7%*	*97.4%*	*99.2%*	*97.4%*	*98.2%*	*97.6%*	

Table 3: Performances on the benchmark datasets, as well as the number of parameters needed to be tuned and stored compared to RoBERTa-base (135M). † is our proposed method. Last row shows the performance of MVP compared to RoBERTa.

Method	MNLI	QNLI	SST-2
RoBERTa-AAE	83.5	91.8	93.0
RoBERTa-SAE	82.8	**91.4**	92.4
Adapter	83.3	91.3	93.3
Prefix	81.5	86.2	93.1
MVP †	**83.8**	89.6	**93.7**
Δ comp. AAE	*100.3%*	*97.6%*	*100.7%*

Table 4: Performances on dialect adaptation. †is our proposed method. Last row shows the performance of MVP compared to RoBERTa finetuned on AAE data.

K	1	2	3	4
MNLI	82.02	**83.84**	83.24	81.52
SST-2	92.89	**94.04**	93.69	93.12
AG News	83.72	**84.93**	84.63	84.25

Table 5: Performances on MNLI, SST-2 and AG News with different number of virtual prompts in MVP.

trainable parameters achieved an average accuracy of 0.866, i.e., 98.5% of the original performances (0.878) averaged across five different datasets, by only updating extra linear layers added to every transformer layer. When we lowered down the hidden dimensions of the added linear layers (only 0.08% parameters are used), the performances of *Adapter* dropped while still maintained 96% of *RoBERTa-base*'s averaged accuracy. Surprisingly, *Prefix* with 0.03% parameters has comparable results as using 0.06% parameters; both are slightly better than *Adapter*. It is worth mentioning that prefix here could be viewed as a special case of our MVP, where the number of virtual prompts was set 1 and virtual prompts were prepended to every transform layer.

In contrast, MVP achieved better performances compared to *Prefix* when both using the same amount of trainable parameters. Such improvements are consistent across five datasets, and even better compared to *Adapter* with 0.08% parameter size. Specifically, compared to *RoBERTa-base*, MVP maintained 95.7% of its performance on MNLI, 97.4% on QNLI, 99.2% on SST-2, 97.4% on AG News, and 98.2% on Yahoo! Answer, when only 0.03% parameters were involved. Overall, this demonstrated the effectiveness of MVP in capturing task specific knowledge of multiple representative NLU tasks, via a mixture of virtual prompts with various learning capacities.

5.2 Dialect Adaptation Evaluation

Once MVP has prepended it's prompts to the input, a prompt modified input can be batched with

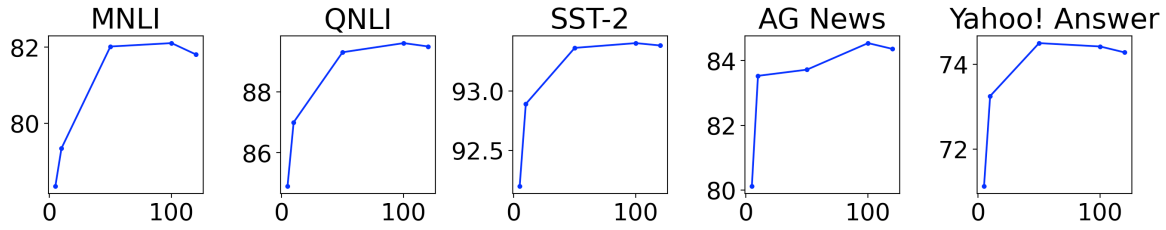

Figure 2: Performances on MNLI, QNLI, SST-2, AG News and Yahoo! Answer with the length of virtual prompt varying from 5 to 120.

Method	MNLI	QNLI	SST-2	AG News	Yahoo!	Average
RoBERTa-base	87.6	92.8	94.8	87.2	76.8	87.84
MVP *w. len = {100}*	82.1	89.6	93.4	84.5	74.4	84.80
MVP *w. len = {50, 50}*	82.1	89.4	93.5	84.3	74.6	84.76
MVP *w. len = {50, 25}*	**83.8**	**90.4**	**94.0**	**84.9**	**75.4**	**85.72**

Table 6: Performance comparisons between mixture of virtual prompts of the same length vs different lengths.

Datasets		Virtual Prompt	
		len = 50	len = 25
MNLI	# Sentences	6788	3044
	# Words	39.41	32.51
	Flesch Score	62.18	67.43
QNLI	# Sentences	4173	1290
	# Words	69.82	40.84
	Flesch Score	46.60	57.38
SST-2	# Sentences	654	218
	# Words	20.03	18.12
	Flesch Score	39.29	42.11
AG News	# Sentences	2730	4870
	# Words	48.02	63.72
	Flesh Score	27.21	12.91
Yahoo! Answer	# Sentences	10925	49075
	# Words	75.8	43.1
	Flesch Score	56.43	65.83

Table 7: The average number of words and Flesh reading ease score of sentences preferred by two different virtual prompts from MVP.

unmodified data without a performance cost. Both adapters and prefix-tuning require injecting additional parameters at intermediate stages, limiting benefits from cross-task batching. In the case of linguistic variation, this allows MVP models trained on two language variants to be run in parallel. This allows systems to add improvements for underserved language variants without major performance impacts.

We report the effectiveness of MVP for this scenario in Table 4. Surprisingly, in two out of three tasks MVP outperforms even finetuning on the dialect specific data indicating that MVP has a high capability to learn patterns of linguistic variation. On QNLI however, all efficient finetuning methods are outperformed by the SAE Roberta model which adds no additional dialect specific features.

5.3 Varying the Length of Virtual Prompts

To examine how the length of virtual prompts affects our model performances, we varied the length L of the used virtual prompts from 5 to 120 . Here, we set the number of virtual prompts as 1, and described the results in Figure 2. We observed that, when L increases from 5 to 50, model performances increased significantly, reaching a peak around L=50. Such trend is consistent across five different datasets. The increases in performances started to converge and even degraded after certain length (L $>$ 50). Thus we set the maximum length of virtual prompts as 50 for MVP in this work. This suggests that for different NLU tasks, there exists an optimal length needed by the virtual prompts, depending on the task complexity itself.

5.4 Varying the Number of Virtual Prompts

Similarly, we looked at the optimal number of virtual prompts needed in MVP. We searched the number of experts (virtual prompts) over {1, 2, 3, 4} (the maximum length of virtual prompts was set 50) and summarized the results in Table 5. Here, MNLI, SST-2 and AG News were used as case studies for different types of tasks, as the other two datasets often exhibited similar properties. When increasing the number of virtual prompts from 1 to

2, we observed an increase in the corresponding F1 scores. More virtual prompts after $K = 3$ did not further help. This communicates to us that more virtual prompts might be needed for more complicated natural language understanding tasks, and for relatively simple classification tasks, two to three virtual prompts seem to be enough.

5.5 Influence of Mixture of Virtual Prompts

To investigate whether a mixture of virtual prompts with different capacities is necessary, we compared three settings, including *MVP w. len={100}* that used only one single virtual prompts with length 100, *MVP w. len = {50, 50}* that used two virtual prompts with the same length (50 and 50), and *MVP w. len = {50, 25}* which utilized the same number of virtual prompts but with different lengths (50 and 25 respectively). As shown in Table 6, a mixture of virtual prompts with different capacities led to significant increases to accuracy over using one single prompts or multiple prompts of the same lengths. This suggested that, virtual prompts with different lengths seem to demonstrate different levels of learning capacities and specializations, thus being more powerful in learning task specific knowledge from multiple perspectives compared to virtual prompts with same lengths.

5.6 Visualizing Individual Virtual Prompt

To test our aforementioned hypothesis that different virtual prompts might have different level of learning capacities and specializations, we took a closer look at the learned virtual prompts. As K was set as 2 based on our prior ablation studies, we first examined which prompt received higher weight on what type of data point. Then we looked at certain properties of these sentences, mainly the number of words per data point and the Flesch reading ease score (Coleman and Liau, 1975) which was a widely used formula to assess the readability of given text (higher scores represent lower difficulties). From Table 7, we observed that, when comparing the virtual prompts of length 25 and of length 50, the sentences that each individual prompt paid attention to demonstrated interesting patterns. Overall, shorter prompt tended to dominate when it came to shorter and easier-to-understand sentences (e.g., 12.9 words with 12.91 Flesch score on AG News) and the relatively longer prompt seemed to work better on longer sentences (e.g., 27.2 words with 48.02 Flesch score on AG News). This demonstrated that, our MVP had a diverse set of capabilities, with each virtual prompt learning its own task-specific specialization.

6 Conclusion and Future Work

In this work, we proposed a simple yet effective tuning technique, Mixture of Virtual Prompts (MVP) adaptation), where virtual prompts are automatically learned from downstream tasks to inject task-specific knowledge while the pre-trained models are frozen to provide general knowledge. To deal with input sentences that might have different complexities, MVP adaptation learns a set of virtual prompts with different capacities and dynamically aggregates them to adapt to every individual sequence. Experiments conducted on 5 natural language understanding tasks demonstrate that our methods could achieve comparable performances with full fine-tuning with only 0.03% parameters. The efficient and dynamic characteristics of MVP also inspire several future research directions:

Zero-Shot Linguistic Transfer Pfeiffer et al. (2020b) showed the effectiveness of separating linguistic and task specific information into discrete modules. Our experiments on dialect adaptation show that MVP can capture linguistic variation as well as task information. This opens opportunities to combine prompts to be used to perform multi-task transfer across dialects and languages, with all the benefits of shared batching at prediction time.

Efficient Continual Learning Through virtual prompts adaptation, task-specific knowledge is naturally disentangled from general language knowledge in pre-trained models, which could inspire more efficient continual learning like Huang et al. (2021) where task-agnostic parameters (pre-trained models) are fixed and only task-specific parameters (virtual prompts) are updated based on novel data in order to mitigate the catastrophic forgetting previous knowledge.

Federated Learning and Data Privacy Mixture of virtual prompts could allow training various prompt with different sets of data separately first and then dynamically ensemble them, which fits the need of federated learning (Kairouz et al., 2019) to collaboratively train a model with decentralized training data. In the meanwhile, it could also protect the data privacy of every individual client as virtual prompts of personalized capacities could be trained separately with client's own data and then shared or ensembled across all the clients.

7 Limitations

Our work only evaluates MVP on the RoBERTa base model. While MVP is not constrained to this architecture and model size, models of different sizes show dramatically different few-shot capabilities using hard prompts so the extent to which our model extends to extremely large models is unclear. Lester et al. (2021) utilizes extremely large scale language models as the basis for much of their comparison, so we are unable to compare directly to their best results.

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, et al. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. *arXiv preprint arXiv:2002.12804*.

Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. 2016. Conditional computation in neural networks for faster models.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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.

Deng Cai, Yan Wang, Victoria Bi, Zhaopeng Tu, Xiaojiang Liu, Wai Lam, and Shuming Shi. 2018. Skeleton-to-response: Dialogue generation guided by retrieval memory. *arXiv preprint arXiv:1809.05296*.

Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Aaai*, volume 2, pages 830–835.

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

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Meri Coleman and Ta Lin Liau. 1975. A computer readability formula designed for machine scoring. *Journal of Applied Psychology*, 60(2):283.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. 2013. Learning factored representations in a deep mixture of experts.

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

Demi Guo, Alexander M Rush, and Yoon Kim. 2020. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. Warp: Word-level adversarial reprogramming.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019a. 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.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019b. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, 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.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. 2021. Continual learning for text classification with information disentanglement based regularization.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020a. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, Rafael G. L. D'Oliveira, Hubert Eichner, Salim El Rouayheb, David Evans, Josh Gardner, Zachary Garrett, Adrià Gascón, Badih Ghazi, Phillip B. Gibbons, Marco Gruteser, Zaid Harchaoui, Chaoyang He, Lie He, Zhouyuan Huo, Ben Hutchinson, Justin Hsu, Martin Jaggi, Tara Javidi, Gauri Joshi, Mikhail Khodak, Jakub Konečný, Aleksandra Korolova, Farinaz Koushanfar, Sanmi Koyejo, Tancrède Lepoint, Yang Liu, Prateek Mittal, Mehryar Mohri, Richard Nock, Ayfer Özgür, Rasmus Pagh, Mariana Raykova, Hang Qi, Daniel Ramage, Ramesh Raskar, Dawn Song, Weikang Song, Sebastian U. Stich, Ziteng Sun, Ananda Theertha Suresh, Florian Tramèr, Praneeth Vepakomma, Jianyu Wang, Li Xiong, Zheng Xu, Qiang Yang, Felix X. Yu, Han Yu, and Sen Zhao. 2019. Advances and open problems in federated learning.

Teven Le Scao and Alexander Rush. 2021. How many data points is a prompt worth? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2627–2636, Online. Association for Computational Linguistics.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *SCL*.

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

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. *arXiv preprint arXiv:2004.03829*.

Zhaojiang Lin, Andrea Madotto, Jamin Shin, Peng Xu, and Pascale Fung. 2019. MoEL: Mixture of empathetic listeners. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 121–132, Hong Kong, China. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021a. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too.

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*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. 2018. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82.

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2020. Generation-augmented retrieval for open-domain question answering. *arXiv preprint arXiv:2009.08553*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2020a. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Evani Radiya-Dixit and Xin Wang. 2020. How fine can fine-tuning be? learning efficient language models. In *International Conference on Artificial Intelligence and Statistics*, pages 2435–2443. PMLR.

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.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *arXiv preprint arXiv:1705.08045*.

Timo Schick and Hinrich Schütze. 2021. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020a. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

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

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.

Fan-Keng Sun and Cheng-I Lai. 2020. Conditioned natural language generation using only unconditioned language model: An exploration. *arXiv preprint arXiv:2011.07347*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Yi-Lin Sung, Varun Nair, and Colin A Raffel. 2021. Training neural networks with fixed sparse masks. In *Advances in Neural Information Processing Systems*, volume 34, pages 24193–24205. Curran Associates, Inc.

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. In *BlackboxNLP@EMNLP*.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. DeeBERT: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 649–657.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. *arXiv preprint arXiv:2004.12406*.

Tony Z Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. *arXiv preprint arXiv:2102.09690*.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit.

Caleb Ziems, Jiaao Chen, Camille Harris, Jessica Anderson, and Diyi Yang. 2022a. VALUE: Understanding dialect disparity in NLU. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3701–3720, Dublin, Ireland. Association for Computational Linguistics.

Caleb Ziems, Jiaao Chen, Camille Harris, Jessica Anderson, and Diyi Yang. 2022b. VALUE: Understanding dialect disparity in NLU. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3701–3720, Dublin, Ireland. Association for Computational Linguistics.