Exploring Low-dimensional Intrinsic Task Subspace via Prompt Tuning

Yujia Qin[1*], **Xiaozhi Wang**[1*], **Yusheng Su**[1], **Yankai Lin**[2], **Ning Ding**[1], **Zhiyuan Liu**[1], **Juanzi Li**[1], **Lei Hou**[1], **Peng Li**[2], **Maosong Sun**[1], **Jie Zhou**[2]
[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]Pattern Recognition Center, WeChat AI, Tencent Inc.
yujiaqin16@gmail.com, wangxz20@mails.tsinghua.edu.cn

Abstract

How can pre-trained language models (PLMs) learn universal representations and effectively adapt to broad NLP tasks differing a lot superficially? In this work, we empirically find evidences indicating that the adaptations of PLMs to various tasks can be reparameterized as optimizing only a few free parameters in a common low-dimensional *intrinsic task subspace*, which may help us understand why PLMs could easily adapt to various NLP tasks with small-scale data. Specifically, to find such a subspace and examine its universality, we resort to the recent success of prompt tuning and decompose the soft prompts of multiple NLP tasks into the same low-dimensional nonlinear subspace, then we learn to adapt the PLM to unseen tasks or data by only tuning parameters in the subspace. We dub this pipeline as *intrinsic prompt tuning* (IPT). In experiments, we study diverse few-shot NLP tasks and surprisingly find that in a 5-dimensional subspace found with 100 random tasks, by only tuning 5 free parameters, we can recover 87% and 65% of the full prompt tuning performance for 100 seen tasks (using different training data) and 20 unseen tasks, respectively, showing great generalization ability of the found intrinsic task subspace. Besides being an analysis tool, IPT could further bring practical benefits, such as improving the prompt tuning stability[1].

1 Introduction

Pre-trained Language Models (PLMs) have shown dominant performances on various natural language processing (NLP) tasks (Han et al., 2021a; Qiu et al., 2020). After pre-training huge parameters on massive data, a PLM can effectively adapt to diverse downstream NLP tasks with small-scale data compared to its model scale, through full-parameter fine-tuning or even parameter-efficient tuning methods (Lester et al., 2021; Houlsby et al., 2019). Nevertheless, the mechanisms behind such adaptations remain unclear. How can PLMs learn universal representations through task-irrelevant pre-training objectives and easily adapt to different NLP tasks differing a lot? Trying to understand how could PLMs universally perform adaptation at a small cost, in this paper, we hypothesize that the optimization problems of PLM adaptations to various downstream tasks can be reparameterized as optimizing only a few free parameters in the same low-dimensional parameter subspace, which we call *intrinsic task subspace* (Figure 1).

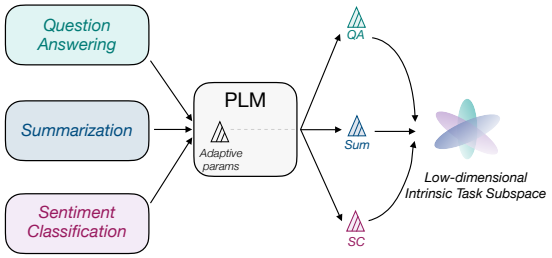

Figure 1: An illustration of a common low-dimensional intrinsic task subspace for diverse NLP tasks. During adaptation, PLMs learn tunable adaptive parameters for each task.

Specifically, the adaptation here is to optimize the tunable *adaptive parameters* so that a PLM could adapt to a certain downstream task, which is typically a very high-dimensional optimization problem. For instance, in conventional fine-tuning, the adaptive parameters are all the PLM parameters and the classification heads, which are more than hundreds of millions. However, Aghajanyan et al. (2021) show that the adaptation to a single task of a PLM can be reparameterized into only optimizing hundreds of free parameters in a low-dimensional subspace and then randomly projecting the tuned parameters back into the full parameter space. This motivates our hypothesis that adaptations to multiple tasks can be reparameterized into optimizations in the same low-dimensional intrinsic task

* The first two authors contributed equally.
[1]Work in progress.

subspace. If this hypothesis holds, the existence of task-specific optimization subspaces explains the universality of PLMs and the low dimensionality explains why the adaptations can be done with relatively small-scale data. From this perspective, the PLMs serve as general *compression frameworks*, which compress the learning complexities of various tasks from very high dimensionalities to low dimensionalities.

To find evidence for the hypothesis, we need to develop methods for finding the intrinsic task subspaces of PLMs. Naturally, the subspace should contain adaptation solutions (i.e., tunable adaptive parameters) for various tasks, hence we can approximate the subspace by training a low-dimensional decomposition of the adaptive parameters using multiple tasks and then examine whether we can learn unseen tasks in the found subspace. However, training a decomposition for all the PLM parameters (the case of fine-tuning) is computationally intractable since the parameters of the projection into subspaces will be hundreds of PLMs. Prompt tuning (PT) provides a parameter-efficient alternative, whose adaptive parameters, i.e., the *soft prompts*, are only tens of thousands. Nevertheless, PT can achieve close performances to fine-tuning on both understanding (Lester et al., 2021; Liu et al., 2021b) and generation (Li and Liang, 2021) tasks. Moreover, PT does not have structural biases since the tuned soft prompts are confined to input embeddings, hence decomposing them is intuitively easier than other parameter-efficient tuning methods like adapter (Houlsby et al., 2019).

In experiments, we explore the common intrinsic subspace through PT under the few-shot learning setting, which ensures the data scales of various tasks are balanced. We name the empirical method used in this paper as intrinsic prompt tuning (IPT), consisting of two phases: multi-task subspace finding (MSF) and intrinsic subspace tuning (IST). During MSF, we first obtain optimized soft prompts for multiple tasks and then learn an auto-encoder by first projecting them into a low-dimensional subspace and then reconstructing them with a back-projection. The optimized auto-encoder defines the desired intrinsic subspace. During IST, we only train the few free parameters for unseen data and tasks in the low-dimensional subspace found by MSF through back-projection.

Surprisingly, we find that the intrinsic task subspace may not only exist but also is extremely low-dimensional. We study diverse few-shot NLP tasks and find that in a 5-dimensional subspace found by 100 random tasks with MSF, we can recover 87% and 65% of the full PT performance with IST for 100 seen tasks (using different training data) and 20 unseen tasks, respectively. Furthermore, we investigate the effectiveness of IPT under different task types as well as the number of training tasks and data. We also show that the found intrinsic task subspace and IPT have some practical uses, such as analyzing task differences and improving prompt tuning stability. We encourage future work to explore how to better find the intrinsic task subspace and develop advanced techniques taking inspiration from low-dimensional reparameterizations of PLM adaptations.

2 Related Work

PLM, Fine-tuning and Prompt tuning. With the success since BERT (Devlin et al., 2019), pre-trained language models (Radford et al., 2018; Yang et al., 2019; Liu et al., 2019; Raffel et al., 2019) bring a new paradigm to NLP, that is to pre-train a massive model as the universal backbone and then adapt the PLMs to specific downstream tasks. The mainstream way for downstream adaption is fine-tuning, which adds task-specific classification heads and tunes all the PLM parameters with supervised data for the downstream tasks.

Recently, researchers found that promising results can also be achieved by casting downstream tasks into the form of pre-training tasks and adding some *prompt* tokens into the input, including human-designed explainable prompts (Brown et al., 2020; Schick and Schütze, 2021a,b; Han et al., 2021b; Ding et al., 2021; Hu et al., 2021) and automatically searched prompts (Jiang et al., 2020; Shin et al., 2020; Gao et al., 2021). Following this line of study, the prompts are extended from tokens in the vocabularies to trainable embeddings, i.e., soft prompts (Li and Liang, 2021; Hambardzumyan et al., 2021; Zhong et al., 2021; Qin and Eisner, 2021; Liu et al., 2021b; Lester et al., 2021). Furthermore, some works (Li and Liang, 2021; Qin and Eisner, 2021; Lester et al., 2021) demonstrate that only tuning soft prompts and keeping PLM parameters frozen can achieve good performances in various tasks. Especially, Lester et al. (2021) show that with the growth of PLM's size, the gap between prompt tuning and fine-tuning becomes narrower and finally disappears.

In this work, we try to take a step towards understanding these phenomena, i.e., how can PLMs learn universal abilities to adapt to various tasks with few data and tunable parameters.

Intrinsic Dimensionality. *Intrinsic dimension* (ID) is the minimal number of variables needed to represent some data or approximate a function. Li et al. (2018) propose to measure the IDs of objective functions optimized by neural networks by randomly projecting all the trainable parameters of neural networks into linear subspaces and find the minimal dimensions that satisfactory solutions (e.g. can achieve 90% performances) appear. Following this, Aghajanyan et al. (2021) show that the IDs of PLM fine-tuning on many NLP tasks can be lower than thousands and the pre-training implicitly optimizes the IDs of downstream tasks, which motivates this work. The random linear projection used in their methods does not involve any additional training for generating low-dimensional subspaces, so that successfully finding solutions in the subspaces provides ample evidence for the existence of effective low-dimensional reparameterizations. The random linear projections inevitably introduce redundant task-irrelevant information and make the generated subspaces not compact for reparameterizing task adaptations. Considering the existence has been given and the hypothesis we want to study is whether the low-dimensional subspace is universal, we resort to stronger subspace-finding method and use supervision from diverse downstream tasks to train a nonlinear low-dimensional decomposition for adaptive parameters.

Unifying Different NLP Tasks. Although various NLP tasks differ a lot on the surface, there has been long-standing attempts to unify different NLP tasks into the same form (Sun et al., 2021) and thus handle them with similar techniques, especially after the success of the prompting methods (Liu et al., 2021a) to cast various tasks into the form of pre-training tasks of PLMs. The analyses in this paper may help us understand how can this be possible and how to better unify different tasks from the perspective of intrinsic task subspace.

3 Methodology

We first introduce essential preliminaries for both fine-tuning and soft prompt tuning in § 3.1, and then we introduce our proposed analysis pipeline **I**ntrinsic **P**rompt **T**uning (**IPT**) in § 3.2, consisting of two training stages: (1) Multi-task Subspace Finding (MSF) and (2) Intrinsic Subspace Tuning (IST). In Figure 2 we visualize the paradigms of fine-tuning, prompt tuning and our IPT.

3.1 Preliminaries

Assume we are given a series of NLP tasks $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_{|\mathcal{T}|}\}$, without loss of generality, following Raffel et al. (2019), we suppose each task $\mathcal{T}_i$ is casted as the unified format conditional generation: $\mathcal{T}_i = \{\mathcal{X}_i, \mathcal{Y}_i\}$, where both the input $\mathcal{X}_i$ and the target $\mathcal{Y}_i$ consist of a series of tokens, i.e., $\mathcal{X}_i = \{w_1, w_2, \ldots, w_{|\mathcal{X}|}\}$, $\mathcal{Y}_i = \{y_1, y_2, \ldots, y_{|\mathcal{Y}_i|}\}$. The goal is to learn a mapping function $\mathcal{F}: \mathcal{X}_i \rightarrow \mathcal{Y}_i$, and the de-facto way is to model $\mathcal{F}$ with a PLM $\mathcal{M}$, which first converts the input $\mathcal{X}_i = \{w_1, w_2, ..., w_{|\mathcal{X}_i|}\}$ into embeddings $\mathbf{E}_i = \{\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{|\mathcal{X}_i|}\} \in \mathbb{R}^{|\mathcal{X}_i| \times d}$, where d denotes the hidden size, then encodes $\mathbf{E}_i$ into hidden representations $\mathbf{H}_i = \{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{|\mathcal{X}_i|}\} \in \mathbb{R}^{|\mathcal{X}_i| \times d}$ and finally decodes $\mathcal{Y}_i$ conditioned on $\mathbf{H}_i$. The goal is to optimize the following objective:

$$\mathcal{L}^i_{\text{LM}} = -\frac{1}{|\mathcal{Y}_i|} \prod_{j=1}^{|\mathcal{Y}_i|} p(y_j | w_1, ..., w_{|\mathcal{X}_i|}, y_1, ..., y_{j-1}). \tag{1}$$

In traditional fine-tuning, all parameters of $\mathcal{M}$ ($\theta_{\mathcal{M}}$) participate in the optimization. Recently, prompt tuning (Lester et al., 2021) springs up as an effective alternative with extensively fewer tunable adaptive parameters. Formally, prompt tuning introduces additional information into the input $\mathcal{X}_i$ by prepending a series of tunable vectors $\mathbf{P}_i = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}$ parameterized by θ_P before $\mathbf{H}_i$, i.e., the modified input embeddings $\mathbf{E}^*_i = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n; \mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{|\mathcal{X}_i|}\} \in \mathbb{R}^{(n+|\mathcal{X}_i|) \times d}$. During prompt tuning, $\theta_{\mathcal{M}}$ remains fixed and only θ_P is updated, satisfying $|\theta_P| = n \times d \ll |\theta_{\mathcal{M}}|$.

3.2 Intrinsic Prompt Tuning

To verify our hypothesis that the adaptations of PLMs to various downstream tasks can be reparameterized as optimization within a low-dimensional *intrinsic task subspace*, we propose a two-phase pipeline called *intrinsic prompt tuning* (IPT). The first phase is multi-task subspace finding (MSF), aiming to find the intrinsic task subspace with multiple tasks' prompts, which are defined by an auto-encoder with a projection function and a back-projection function. The second intrinsic subspace

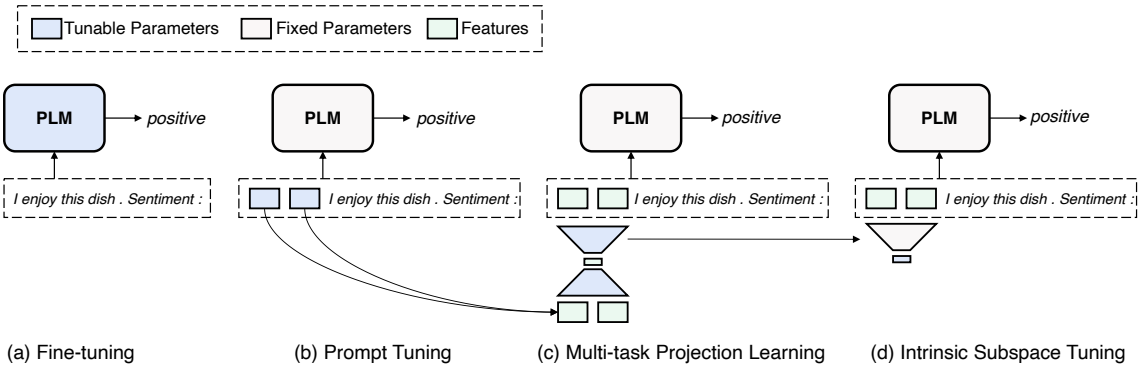

Figure 2: Illustrations for fine-tuning (a), prompt tuning (b) and two components of IPT (c,d). We discriminate tunable parameters, fixed parameters and intermediate features with different colors.

tuning (IST) phase is to tune parameters only in the subspace and then recover them to soft prompts with the back-projection function.

Multi-task Subspace Finding. During MSF, we try to find a satisfactory intrinsic task subspace of a low dimension d_I by learning a decomposition for the matrix $\mathbf{P}_i \in \mathbb{R}^{n\times d}$, which is the trained soft prompts for a downstream task $\mathcal{T}_i$. Inspired by text autoencoders (Bowman et al., 2016), the decomposition consists of a projection function $\mathbf{Proj}(\cdot)$ to project $\mathbf{P}_i$ into the d_I-dimensional subspace and a back-projection function $\mathbf{Proj}_b(\cdot)$ to project the d_I-dimensional vectors back into soft prompts to handle $\mathcal{T}_i$, and we reconstruct $\mathbf{P}_i$ as follows:

$$\begin{aligned} \mathbf{P}_i^* &= \mathbf{Proj}_b(\mathbf{Proj}(\mathbf{P}_i)), \\ \mathcal{L}_{\text{AE}}^i &= ||\mathbf{P}_i^* - \mathbf{P}_i||_2^2, \end{aligned} \tag{2}$$

where $\mathbf{Proj}(\cdot)$ is implemented with a one-layer FFN and $\mathbf{Proj}_b(\cdot)$ is parameterized by a two-layer perceptron as follows:

$$\mathbf{Proj}_b(\mathbf{d}_i) = \mathbf{W}_2(\tanh(\mathbf{W}_1\mathbf{d}_i + \mathbf{b}_1)) + \mathbf{b}_2, \tag{3}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_I'\times d_I}$, $\mathbf{b}_1 \in \mathbb{R}^{d_I'}$, $\mathbf{W}_2 \in \mathbb{R}^{n\times d\times d_I'}$ and $\mathbf{b}_2 \in \mathbb{R}^{n\times d}$ are trainable parameters. Moreover, finding the decomposition for a prompt $\mathbf{P}_i$ of a certain task, which is essentially a matrix, is somewhat trivial. Since the desired intrinsic task subspace we want to find in MSF should work for broad tasks, we introduce multi-task training and also take the task-oriented language modeling losses $\mathcal{L}_{\text{LM}}$ of the reconstructed soft prompts for the training tasks as objective functions. By jointly optimizing the reconstruction loss and the task-oriented losses, the subspace could gain the ability of reparameterizing various task adaptaions. The overall training objective of MSF is shown as follows:

$$\mathcal{L}_{\theta_{\text{proj}}} = \frac{1}{|\mathcal{T}_{\text{train}}|}\sum_{i=1}^{|\mathcal{T}_{\text{train}}|}(\mathcal{L}_{\text{LM}}^i + \alpha\mathcal{L}_{\text{AE}}^i), \tag{4}$$

where α denotes the hyper-parameter controlling the ratio between the two losses, and θ_{proj} denotes the parameters of both $\mathbf{Proj}$ and $\mathbf{Proj}_b$. During $\mathbf{MSF}$, we only optimize $\mathbf{Proj}$ and $\mathbf{Proj}_b$ while keeping other parameters fixed. By introducing downstream task supervision and nonlinearity, we could effectively find more irredundant and stronger subspaces than the random linear subspaces (Li et al., 2018; Aghajanyan et al., 2021).

Intrinsic Subspace Tuning. In this stage, we want to evaluate if the subspace found by MSF is generalizable to unseen training data or unseen tasks. And if the answer is yes, we can say that we successfully find the intrinsic task subspace reparameterizing the adaptations of PLMs to various tasks to some extent. Specifically, we only retain $\mathbf{Proj}_b$ learned during MSF and keep both $\mathbf{Proj}_b$ and $\mathcal{M}$ fixed, then for each task, instead of vanilla prompt tuning, we tune only d_I free parameters (θ_d) in the subspace and project the d_I parameters into soft prompts with $\mathbf{Proj}_b$, the objective function could be formulated as follows:

$$\mathcal{L}_{\theta_d}^i = \mathcal{L}_{\text{LM}}^i. \tag{5}$$

4 Experiment and Analysis

In this section, we first describe the experimental settings in § 4.1, including the tasks and corresponding datasets, evaluation metrics and training

details of this paper. Then we introduce the experimental results and analyses in § 4.2 and § 4.3.

4.1 Experimental Settings

Tasks and Datasets. To cover broad and diverse NLP tasks in our experiments, we randomly choose a wide variety of few-shot NLP tasks $\mathcal{T}$ (120 in total) from the pool of *CrossFit Gym* (Ye et al., 2021). The few-shot setting ensures the data scales of various tasks are balanced, so that the subspace found by MSF will not be easily biased towards data-rich tasks.

For a brief introduction, *CrossFit Gym* consists of various types of few-shot NLP tasks, including text classification (e.g., sentiment analysis and natural language inference), question answering (e.g., machine reading comprehension and multi-choice question answering), conditional generation (e.g., summarization and dialogue), etc. As mentioned in § 3.1, all tasks are processed into a unified sequence-to-sequence format following Raffel et al. (2019) and Khashabi et al. (2020) for ease of handling them with unified text-to-text PLMs. For instance, the input of a multi-choice QA task is formulated as: "`Question: <question> Context: <context> Candidates: <candidates>`", and the PLM is expected to generate the correct answer span from `<candidates>`. Each task $\mathcal{T}_i \in \mathcal{T}$ could be represented as a tuple of ($\mathcal{D}^i_{\text{train}}$, $\mathcal{D}^i_{\text{dev}}$, $\mathcal{D}^i_{\text{test}}$), where the size of the $\mathcal{D}^i_{\text{train}}$/$\mathcal{D}^i_{\text{dev}}$ is set to K to ensure the few-shot setting. For classification and regression tasks, $K = 16$, while for other categories of tasks, $K = 32$. We list task details in § 6.

Evaluation Metrics. Since different tasks have distinct evaluation protocols (e.g., F1 score for discriminative tasks and BLEU for generative tasks typically), we introduce both average absolute performance (E_{abs}) and average relative performance (E_{rel}) as main evaluation metrics. Specifically, let $\mathcal{T} = \{\mathcal{T}_1, ..., \mathcal{T}_{|\mathcal{T}|}\}$ be the tasks to be evaluated, let $E_{\mathcal{T}_i}$ denotes the test score of $\mathcal{T}_i$ for IPT, $E_{\text{abs}} = \frac{1}{|\mathcal{T}|} \sum_{\mathcal{T}_i \in \mathcal{T}} E_{\mathcal{T}_i}$, $E_{\text{rel}} = \frac{1}{|\mathcal{T}|} \sum_{\mathcal{T}_i \in \mathcal{T}} \frac{E_{\mathcal{T}_i}}{E^*_{\mathcal{T}_i}}$, where $E^*_{\mathcal{T}_i}$ denotes the performance with either prompt tuning ($E^{\text{PT}}_{\mathcal{T}_i}$) or fine-tuning ($E^{\text{FT}}_{\mathcal{T}_i}$). In this paper, we use E_{rel} as the main evaluation criterion and choose E_{abs} as the auxiliary one. To properly evaluate the generalization ability achieved by IPT, we randomly sample training tasks $\mathcal{T}_{\text{train}}$ and test tasks $\mathcal{T}_{\text{test}}$ from $\mathcal{T}$, satisfying that $\mathcal{T}_{\text{train}} \bigcap \mathcal{T}_{\text{test}} = \emptyset$.

During the multi-task subspace finding (MSF) stage, PLMs are trained on $\mathcal{T}_{\text{train}}$ only, and we evaluate both E_{abs} and E_{rel} on $\mathcal{T}_{\text{train}}$ to see how much performance we will lose by reconstructing prompts compressed into d_I-dimensional subspace, which will provide an empirical upper bound for the generalization to unseen data and tasks. We also reconstruct the trained soft prompts of $\mathcal{T}_{\text{test}}$ with the auto-encoder learned in MSF and test the performances to investigate the learned auto-encoder's reconstruction ability for unseen soft prompts.

For the intrinsic subspace tuning (IST) stage, we first carry out IST on $\mathcal{T}_{\text{train}}$ using exactly the same $\mathcal{D}^i_{\text{train}}$/$\mathcal{D}^i_{\text{dev}}$ utilized in MSF to investigate to what extent the performance could be recovered by optimizing only in the subspace. After that, we evaluate the generalization ability of IPT to see whether adapations to various tasks are substantially reparameterized into the found subspace with two generalization challenges: (1) *unseen-data challenge* and (2) *unseen-task challenge*.

• For the *unseen-data challenge*, we conduct IST on $\mathcal{T}_{\text{train}}$ with a different K-shot $\mathcal{D}^{i'}_{\text{train}}$/$\mathcal{D}^{i'}_{\text{dev}}$ by re-sampling training data, while keeping $\mathcal{D}^i_{\text{test}}$ the same. Note $\mathcal{D}^{i'}_{\text{train}}$/$\mathcal{D}^{i'}_{\text{dev}}$ and $\mathcal{D}^i_{\text{train}}$/$\mathcal{D}^i_{\text{dev}}$ conform with the i.i.d. assumption. The *unseen-data challenge* is designed to test whether the learned subspace can generalize to unseen data, which will lead to different optimization trajectories.

• For the *unseen-task challenge*, we evaluate the soft prompts obtained through IPT on $\mathcal{T}_{\text{test}}$ to see how well can optimization in the found subspace recover the adaptation of PLM to unseen tasks, which will provide evidences for our hypothesis that the reparameterization subspaces for different task adaptations of PLMs are not independent.

Training Details. Since all tasks are unified into the same sequence-to-sequence format, we use BART$_{BASE}$ (Lewis et al., 2020) for all experiments under the same environment of NVIDIA 32GB V100 GPU. We set the inner hidden size d'_I of $\mathbf{Proj}_b$ to 768. For all experiments, we adopt AdamW (Loshchilov and Hutter, 2019) as the optimizer. For the prompt tuning/fine-tuning baseline, we perform grid search on the combination of a series of learning rates ($\{1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}\}$) and batch sizes ($\{2, 4, 8\}$)[2], choose the best checkpoint using $\mathcal{D}_{\text{dev}}$, and evaluate it on $\mathcal{D}_{\text{test}}$. We set the max step to $10,000/100,000$

[2]The numbers are chosen by pilot experiments on a random subset of tasks

Shorthand	$\mathcal{T}_{\text{train}}$	$\mathcal{T}_{\text{test}}$	Prompt Tuning		Fine-tuning	
			$\mathcal{T}_{\text{train}}$	$\mathcal{T}_{\text{test}}$	$\mathcal{T}_{\text{train}}$	$\mathcal{T}_{\text{test}}$
random	100 random	20 random	32.6	40.1	35.2	40.7
non-cls	35 non-cls.	42 non-cls.($\mathcal{T}_{\text{test}}^{\text{in}}$) / 43 cls.($\mathcal{T}_{\text{test}}^{\text{out}}$)	23.0	28.0/49.0	24.4	29.6/52.2
cls	35 cls.	8 cls.($\mathcal{T}_{\text{test}}^{\text{in}}$) / 77 non-cls.($\mathcal{T}_{\text{test}}^{\text{out}}$)	48.6	50.9/25.7	52.5	51.1/27.2

Table 1: The overall 120 tasks $\mathcal{T}$ consist of 43 classification tasks (cls.) and 77 non-classification tasks (non-cls.). Three task partitions are evaluated in this paper, including *random*, *non-cls* and *cls*, with details listed above, e.g., for *non-cls* partition, 35 non-cls. are chosen as $\mathcal{T}_{\text{train}}$, and 42 non-cls./43 cls. are chosen as $\mathcal{T}_{\text{test}}^{\text{in}}$/$\mathcal{T}_{\text{test}}^{\text{out}}$, respectively. We also list E_{abs} for prompt tuning/fine-tuning on each task split.

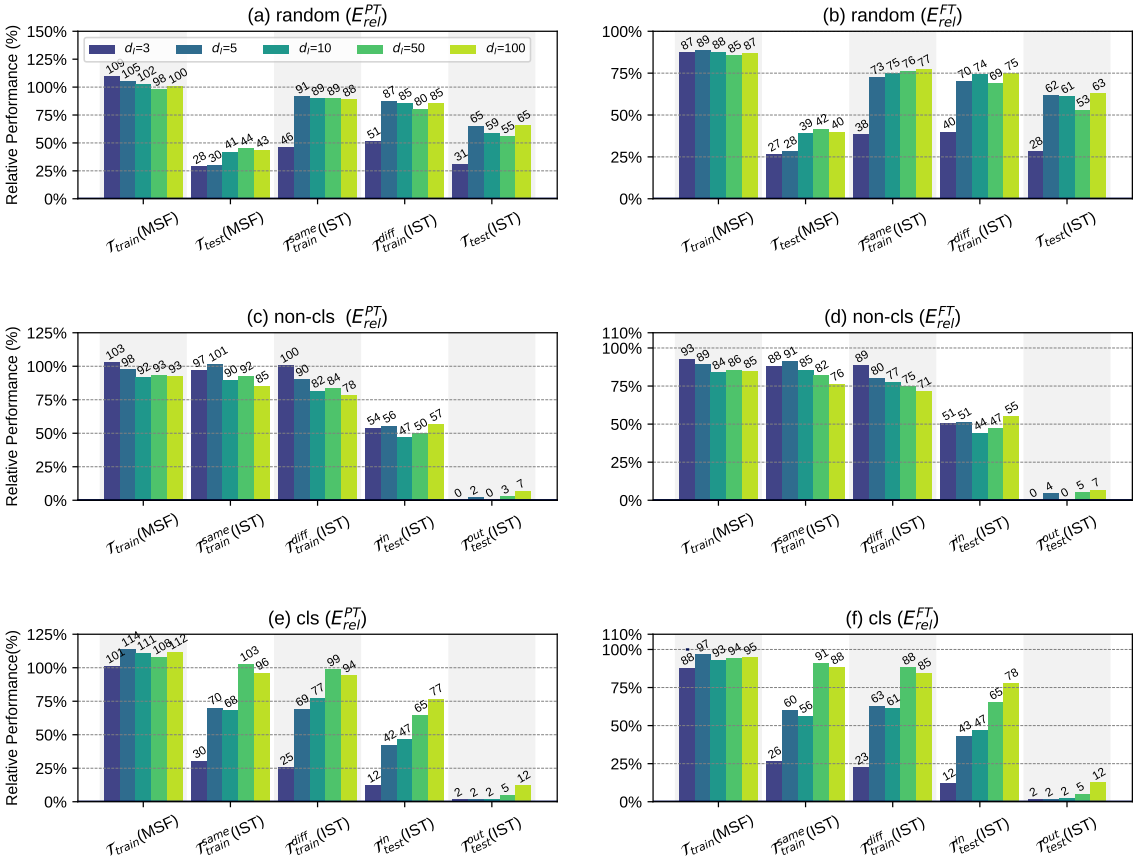

Figure 3: Relative performance for IPT on three task partitions (*random*, *non-cls* and *cls*), comparing both prompt tuning (left, $E_{\text{rel}}^{\text{PT}}$) and fine-tuning (right, $E_{\text{rel}}^{\text{FT}}$).

and validate on $\mathcal{D}_{\text{dev}}$ every 100/1000 steps[3]. We set the number of soft prompts to be 100 for all tasks and randomly initialize them. For IPT, we examine the dimension d_I of $\{3, 5, 10, 50, 100\}$, respectively. During MSF, we only select the prompts that perform best on $\mathcal{D}_{\text{dev}}$ for each task to train the auto-encoder, since we found empirically that involving other prompts leads to worse performance. The hyper-parameters of IST are chosen as the same as prompt tuning for fair comparisons. Note that for fine-tuning/prompt tuning, 139M/76, 800 parameters are tuned during training, while IPT only tunes d_I free parameters.

[3]We found empirically that prompt tuning requires around 10× steps than fine-tuning to converge.

4.2 Main Results

Three task splits are evaluated: *random*, *non-cls* and *cls*, with details listed in Table 1. The experimental results are shown in Figure 3 (E_{rel}), Table 2 (E_{abs}), and Table 3 (E_{abs}). As mentioned before, we choose E_{rel} as the main criterion for analyses. Based on these results, we study the following research questions:

Q1. Do PLMs really reparameterize various task adaptations into a low-dimensional task

Dim (d_I)	3	5	10	50	100
Multi-task Projection Learning					
$\mathcal{T}_{\text{train}}$	29.1	31.8	32.2	32.0	**32.6**
$\mathcal{T}_{\text{test}}$	8.5	10.0	15.0	**16.7**	16.4
Single-task Intrinsic Subspace Tuning					
$\mathcal{T}_{\text{train}}^{\text{same}}$	13.2	25.6	27.8	28.8	**29.6**
$\mathcal{T}_{\text{train}}^{\text{diff}}$	13.0	24.9	27.4	26.7	**28.4**
$\mathcal{T}_{\text{test}}$	9.3	**26.5**	24.7	23.1	25.8

Table 2: E_{abs} on task partition *random*. During MSF, we report E_{abs} on both training tasks ($\mathcal{T}_{\text{train}}$) and the reconstruction performance for test tasks ($\mathcal{T}_{\text{test}}(zs)$). During IST, we first test on $\mathcal{T}_{\text{train}}^{\text{same}}$ to what extent the performance could be recovered by tuning only d_I parameters, and test the generalization ability by carrying out both *unseen-data challenge* and *unseen-task challenge* on $\mathcal{T}_{\text{train}}^{\text{diff}}$ and $\mathcal{T}_{\text{test}}$, respectively.

subspace in the few-shot setting? From the results in Figure 3 (a), we can see that: (1) for the *unseen-data challenge* ($\mathcal{T}_{\text{train}}^{\text{diff}}(IST)$), choosing $d_I \geq 5$ could recover more than 80% of the full prompt tuning performances for the 100 training tasks with IST on unseen i.i.d. data. (2) For the *unseen-task challenge* ($\mathcal{T}_{\text{test}}(IST)$), we can also achieve about 60% performances by only tuning $5 \sim 100$ parameters. From these results, we can say that the low-dimensional reparameterizations in the subspaces found by MSF successfully recover the PLM adapations of $\mathcal{T}_{\text{train}}$ and can also generalize to unseen tasks to some extent, thus non-trivial performances can be achieved by only tuning a few free parameters in these subspaces. This provides evidence for our hypothesis that PLMs reparameterize various task adaptations into the same low-dimensional subspace, or at least the low-dimensional reparameterization subspaces for various task adaptations (Aghajanyan et al., 2021) should have a substantial intersection, otherwise the subspaces found by $\mathcal{T}_{\text{train}}$ will be almost impossible to work for $\mathcal{T}_{\text{test}}$.

Moreover, in the *random* split, we can see that $E_{\text{rel}}^{\text{PT}}$ of both $\mathcal{T}_{\text{train}}^{\text{diff}}(IST)$ and $\mathcal{T}_{\text{test}}(IST)$ are always on the same level when $d_I \geq 5$, indicating that the dimension for the *intrinsic task subspace* should be around 5, which is extremely low for a PLM with hundreds of millions of parameters.

Q2. What limits IPT? Although positive evidence is observed, the effectiveness of IPT is still limited considering only about 60% performances can be recovered for unseen tasks. From the results in Figure 3 (a) and (b), we discuss what factors may limit the effectiveness of IPT and provide insights for improving the analysis pipeline.

1. The performances for $\mathcal{T}_{\text{train}}$ when we directly reconstruct soft prompts using the autoencoder of MSF ($\mathcal{T}_{\text{train}}(MSF)$) are even better than vanilla soft prompt tuning, which demonstrates that (1) the pipeline of MSF can help improve soft prompt tuning by enforcing multi-task skill sharing with the extremely low dimensions, and (2) there at least exist good enough solutions in the subspaces, which have been found by MSF. However, even using exactly the same training data, IST cannot find these good solutions and thus leads to the gap between $\mathcal{T}_{\text{train}}(MSF)$ and $\mathcal{T}_{\text{train}}^{\text{same}}(IST)$, which shows that the limitations of the adopted optimization algorithms will influence the performance of IST.

2. From the comparisons between $\mathcal{T}_{\text{train}}(MSF)$ and $\mathcal{T}_{\text{test}}(MSF)$, we can see that the performances of directly reconstructing soft prompts for unseen tasks are poor. It indicates that the reconstruction ability of the auto-encoders trained in MSF cannot generalize well to unseen inputs (soft prompts), which may limit IPT to some extent. Nevertheless, IST can still find much better solutions within the subspace found by MSF.

3. From the comparisons between results in Figure 3 (a) and (b), we can see that the recovered relative performance of fine-tuning (E_{rel}^{FT}) is always poorer than that of prompt tuning (E_{rel}^{PT}). This is because the soft prompt tuning is slightly inferior than fine-tuning under few-shot setting, and the performance of IPT is bounded by soft prompt tuning, since MSF is designed to reconstruct soft prompts. Ideally, E_{rel}^{FT} could be further improved by designing more advanced prompt tuning algorithms.

Q3. How is the influence of task types? Following CrossFit (Ye et al., 2021), we divide the studied tasks into two parts: *cls* (classification), which are discriminative tasks and *non-cls* (non-classification), which tend to be generative tasks. From the results in Figure 3 (c)-(f), we find that, aligned with the common sense, there exists a great discrepancy between them: (1) there is a huge generalization gap between classification tasks and non-classification tasks. When using only a kind of tasks in MSF (*cls* or *non-cls*), the found subspaces work fine for the same kind of tasks ($\mathcal{T}_{\text{test}}^{\text{in}}(IST)$) but poorly generalize to the other kind of tasks ($\mathcal{T}_{\text{test}}^{\text{out}}(IST)$). (2) When increasing d_I, *non-cls* tasks (Figure 3 (c) and (d)) tend to show decreasing performances in all the settings, but *cls* per-

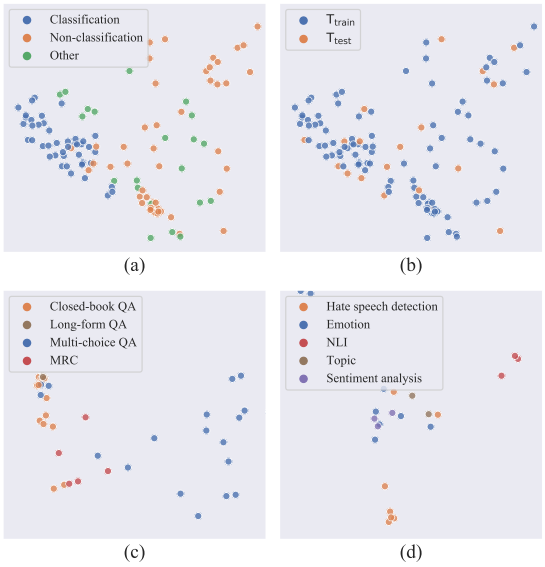

Figure 4: PCA plots of the intrinsic vectors learned during IST. We label points with different colors to represent its corresponding categories. Specifically, we show the clusters of classification and non-classification tasks in (a), $\mathcal{T}_{\text{train}}$ and $\mathcal{T}_{\text{test}}$ in (b), fine-grained categories of QA and text classification tasks in (c) and (d), respectively. Without loss of generality, we choose the task split of *random* and $d_I = 100$.

formances (Figure 3 (e) and (f)) tend to increase. Reasonably, an ideal intrinsic subspace dimension setup for various NLP tasks should at least exceed a threshold (the *intrinsic dimension*) to ensure that the subspace could be substantially depicted, but should not be too large, which may cause over-parameterization[4]. Hence we believe this indicates that, although counter-intuitively, the most appropriate intrinsic subspace dimension for *non-cls* tasks is far smaller than *cls* tasks. We hypothesize this may come from the few-shot setting and will explore it in future.

4.3 Analyses and Properties

Visualization of the Found Intrinsic Subspace. We visualize the intrinsic vectors (vectors consisting of the free parameters learned during IST in the found subspace) using PCA in Figure 4, from which we observe that: (1) there exists an clear dividing line between the clusters of classification tasks and non-classification tasks, indicating that they are highly distinct, which is aligned with the common sense. This also explains why subspaces learned on either cluster generalize poorly to the other cluster. (2) The points of unseen tasks $\mathcal{T}_{\text{test}}$ are mixed with those of $\mathcal{T}_{\text{train}}$, which demonstrates that the found subspaces universally reparameterize various tasks so that PLMs can generalize to unseen tasks. (3) It is also observed from (c) and (d) that the points belonging to the same category exhibit a compact cluster. We argue that the learned intrinsic vectors could be viewed as low-dimensional task representations, helping analyze the similarity and difference for various NLP tasks.

[4]We also observe non-increasing trends for *cls* when d_I is enlarged above 500.

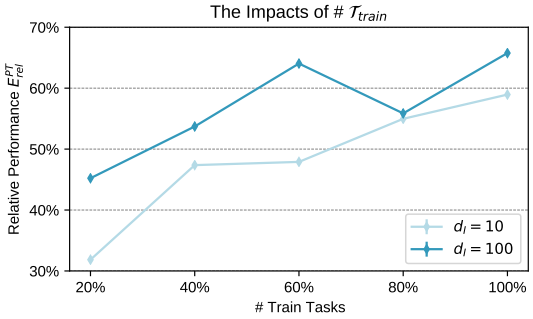

Figure 5: Impacts of the number of train tasks ($\mathcal{T}_{\text{train}}$). We test $E^{\text{PT}}_{\text{rel}}$ by sampling $\{20\%, 40\%, 60\%, 80\%\}$ tasks in $\mathcal{T}_{\text{train}}$ with $d_I = 10$, $d_I = 100$, respectively.

Impacts of the Number of Training Tasks. During MSF, the auto-encoder is optimized to reconstruct the adaptive parameters of various training tasks to reparameterize them into a low-dimensional task subspace. Ideally, the coverage of $\mathcal{T}_{\text{train}}$ would significantly impact the generalization ability of IPT on unseen tasks $\mathcal{T}_{\text{test}}$. To demonstrate this, we choose the task partition *random*, and randomly sample only $\{20\%, 40\%, 60\%, 80\%\}$ tasks in the original $\mathcal{T}_{\text{train}}$ for training the auto-encoder, then evaluate the *unseen-task* challenge on the same $\mathcal{T}_{\text{test}}$. The results are visualized in Figure 5, from which we observe that, with the number of training tasks growing, the generalization ability of the found task subspace is generally improved. This reflects that increasing the coverage and diversity of seen tasks could help IPT find more universal subspaces.

Impacts of the Number of Shots. Although in this paper, we mainly investigate the few-shot setup to control the influence of data amount, it is also interesting to investigate whether IPT's ability will be stronger when more training data is available. Here we take an initial trial by doubling the number of shots K for the task partition *cls*, and conduct experiments to see the performance ($E^{\text{PT}}_{\text{rel}}$) of MSF and IST on $\mathcal{T}^{\text{same}}_{\text{train}}$. Note the denominator when cal-

Train Tasks	Non-classification					Classification				
Dim (d_I)	3	5	10	50	100	3	5	10	50	100
Multi-task Projection Learning										
$\mathcal{T}_{\text{train}}$	**23.3**	23.1	21.9	22.7	22.2	46.0	**50.0**	48.0	49.5	48.7
Single-task Intrinsic Subspace Tuning										
$\mathcal{T}_{\text{train}}^{\text{same}}$	22.1	**23.3**	21.4	20.4	19.5	12.2	32.0	30.3	**48.5**	47.2
$\mathcal{T}_{\text{train}}^{\text{diff}}$	**22.0**	20.5	17.4	19.6	19.7	10.5	33.0	31.9	**46.9**	44.4
$\mathcal{T}_{\text{test}}^{\text{in}}$	16.7	16.4	14.8	17.0	**19.5**	7.8	21.0	24.5	32.7	**38.1**
$\mathcal{T}_{\text{test}}^{\text{out}}$	0.0	1.0	0.8	1.4	**3.9**	0.6	0.7	1.0	2.1	**4.2**

Table 3: E_{abs} on task partition *non-cls* and *cls*. Different from Table 2, during MSF, we only report E_{abs} on training tasks ($\mathcal{T}_{\text{train}}$). During IST, we separately test E_{abs} on $\mathcal{T}_{\text{test}}^{\text{in}}$ and $\mathcal{T}_{\text{test}}^{\text{out}}$ for *unseen-task challenge*.

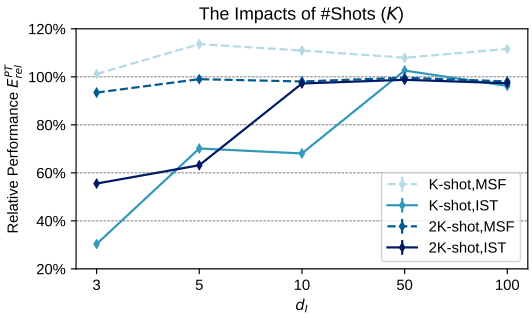

Figure 6: Impacts of the number of shots. We test $E_{\text{rel}}^{\text{PT}}$ when the number of shots is doubled.

Partition	$\mathcal{T}_{\text{train}}$	$\mathcal{T}_{\text{test}}$	$\mathcal{T}_{all}$
FT	2.16	2.40	2.20
PT	3.06	4.19	3.25
IPT	**1.12**	**0.73**	**1.06**

Table 4: Standard deviation (std) of test scores over multiple runs. d_I is chosen to be 10. We observe that std(IPT) $<$ std(fine-tuning) $<$ std(prompt tuning). The significantly lower std for our IPT demonstrates the much stronger stability in the low-dimensional subspace.

culating $E_{\text{rel}}^{\text{PT}}$ is different for K-shot and $2K$-shot experiments. The results are visualized in Figure 6, from which we observe that the gap between MSF and IST is rapidly narrowed in $2K$-shot setting when d_I grows, while it is significantly slower in K-shot setting. This demonstrates that, involving more supervision could benefit the optimization of IST. We also find that when increasing d_I, $\mathcal{T}_{\text{train}}^{\text{same}}$ for $2K$-shot grows very close to 100% but never exceeds it, indicating that when more data is available, jointly reconstructing prompts from multiple tasks does not bring additional benefits as observed in the K-shot setting. In general we believe it is interesting to explore how strong the subspace found by IPT in data-rich scenarios will be.

Improving Prompt Tuning Stability with IPT. In Table 4, we show the mean standard deviation (`std`) of test scores for 120 tasks over 10 runs comparing IPT (d_I = 10), fine-tuning and prompt tuning. We observe that prompt tuning is the most unstable strategy with the highest `std`, while fine-tuning exhibits far more stable. Instability of soft prompt tuning may influence the practical use of this technique. Intuitively, IPT tries to find low-dimensional intrinsic subspace and learn new tasks with only a few free parameters, which will help improve the stability, and IPT surely becomes the most stable methods in Table 4.

To make the stability advantage brought by IPT practical, we propose to use the solutions found by IPT as the initialization for the vanilla prompt tuning. Specifically, we continue the experiments of partition *random* on $\mathcal{T}_{\text{test}}$ choosing $d_I = 10$ and initialize the soft prompts by back-projecting the found solutions in the subspace during IST. Other details are kept the same compared with prompt tuning baseline. We observe that the standard variance achieved in this way is significantly lower than the vanilla prompt tuning (1.65 v.s. 4.19) while in this way we can achieve 103.4% of $E_{\text{rel}}^{\text{PT}}$, i.e., the performance could also be improved from 59% (IST). This indicates that both IPT and prompt tuning could be further combined in a two-stage manner. This experiment also demonstrates that although our IPT pipeline mainly works as an analytical framework in this paper, it can also bring practical benefits. We will explore more practical use of IPT in future.

5 Discussion

Could NLP tasks be reparameterized into the same subspace? In this paper, we find strong evidences that by compressing various tasks' adaptive parameters with downstream supervision, we

could possibly find an extremely low-dimensional subspace containing sub-optimal but non-trivial solutions for adapations of PLMs to unseen tasks. Despite that $E_{\text{rel}}^{\text{PT}}$ exceeds 100% during MSF, we notice that during IST, the generalization on both seen and unseen tasks is still far from perfect (87% v.s. 65% for $E_{\text{rel}}^{\text{PT}}$), and increasing d_I does not significantly improve the performance. Although it may be relevant to the inadequacy of current optimization algorithms, based on current results, we cannot directly conclude that various NLP task adaptations of PLMs could be reparameterized into optimizations within exactly the same subspace. However, at least we have found promising empirical results showing that the low-dimensional reparameterization subspaces of various tasks have a substantial intersection, which could be found by MSF. And conducting IST in this intersection subspace could recover most of the performance for each task. We thus encourage future work to explore (1) whether there exist more efficient and effective optimization algorithms for IST in extremely low-dimensional subspaces, and (2) whether the union of intrinsic subspaces for various tasks is also low-dimensional.

Moreover, it shoud also be noted that the extremely low dimension (about $5 \sim 100$) achieved by IPT should not be regarded exactly as *intrinsic dimension* of Aghajanyan et al. (2021) and Li et al. (2018) since the *intrinsic dimension* should ensure that the overall training process can be done in that low dimension. The dimension achieved with IPT could possibly serve as a rough lower bound for the *intrinsic dimension*.

Relation to the scaling law. Recently, more and more cases for *the power of scale* have been shown. People have found that extremely larger PLMs tend to be much more sample-efficient (Kaplan et al., 2020), parameter-efficient (Lester et al., 2021) and cross-task generalizable (Wei et al., 2021). We think this may be explained with the hypothesis in this paper: the adaptations of larger or better-trained PLMs can be better reparameterized into the same low-dimensional subspace so that the cross-task generalization should be easier, and better pre-training bring lower reparameterization dimensions (Aghajanyan et al., 2021), hence the larger PLM should need less data and tunable parameters.

If this hypothesis holds, we argue that tuning methods similar to IPT should be developed. Larger PLMs provide much stronger compression framework and we can find the low-dimensional intrinsic task subspaces and train various downstream tasks in these subspaces, which will avoid the instability and large generalization gap caused by over-parameterization and also be greener to environments.

6 Conclusion and Future work

In this paper, we study the hypothesis that the adaptations of PLMs to various tasks can be reparameterized as optimizations of a few free parameters in the same low-dimensional *intrinsic task subspace*. We develop an analysis pipeline called IPT, which first finds a subspace by jointly compressing the adaptive parameters of multiple tasks and then tune parameters only in the subspace for unseen data and tasks. The non-trivial performances achieved by IPT in extremely low dimension provide positive evidence for the hypothesis. We also discuss the factors influencing the results and the potential practical use of IPT. In future, we will improve the IPT framework to better verify the *universal low-dimensional reparameterization* hypothesis and develop more practical techniques for more efficient optimization in low-dimensional subspaces.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. 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 7319–7328, Online. Association for Computational Linguistics.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

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. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

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.

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*, 2108.10604.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. 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 3816–3830, Online. Association for Computational Linguistics.

Karen Hambardzumyan, Hrant Khachatrian, and Jonathan May. 2021. WARP: Word-level Adversarial ReProgramming. 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 4921–4933, Online. Association for Computational Linguistics.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Liang Zhang, Wentao Han, Minlie Huang, Qin Jin, Yanyan Lan, Yang Liu, Zhiyuan Liu, Zhiwu Lu, Xipeng Qiu, Ruihua Song, Jie Tang, Ji-Rong Wen, Jinhui Yuan, Wayne Xin Zhao, and Jun Zhu. 2021a. Pre-trained models: Past, present and future. *ArXiv preprint*, abs/2106.07139.

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

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, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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*, 2108.02035.

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

Jared Kaplan, Sam McCandlish, T. J. Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *ArXiv preprint*, abs/2001.08361.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. 2018. Measuring the intrinsic dimension of objective landscapes. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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.

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*, abs/2107.13586.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021b. Gpt understands, too. *ArXiv preprint*, 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. *ArXiv preprint*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5203–5212, Online. Association for Computational Linguistics.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, pages 1–26.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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*, abs/1910.10683.

Timo Schick and Hinrich Schütze. 2021a. 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.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352, Online. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. 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.

Tianxiang Sun, Xiangyang Liu, Xipeng Qiu, and Xuanjing Huang. 2021. Paradigm shift in natural language processing. *ArXiv preprint*, abs/2109.12575.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *ArXiv preprint*, abs/2109.01652.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. Crossfit: A few-shot learning challenge for cross-task generalization in nlp. *ArXiv preprint*, abs/2104.08835.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: Learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5017–5033, Online. Association for Computational Linguistics.

Appendices

Table 5: The tasks evaluated in our experiments. We refer to Ye et al. (2021) for task ontology.

Ontology	Task Name
cls/sentiment analysis	glue-sst2 imdb rotten_tomatoes
cls/emotion	emo tweet_eval-emoji tweet_eval-hate tweet_eval-irony tweet_eval-offensive tweet_eval-sentiment tweet_eval-stance_abortion tweet_eval-stance_atheism tweet_eval-stance_climate tweet_eval-stance_feminist tweet_eval-stance_hillary
cls/hate speech detection	ethos-disability ethos-gender ethos-national_origin ethos-religion ethos-sexual_orientation hate_speech18 hatexplain
cls/NLI	anli glue-mnli glue-qnli glue-rte glue-wnli scitail superglue-rte
cls/fact checking	climate_fever kilt_fever liar
cls/paraphrase	glue-qqp medical_questions_pairs paws
cls/topic	ag_news dbpedia_14
cls/other	ade_corpus_v2-classification discovery glue-cola google_wellformed_query sms_spam superglue-wic superglue-wsc wiki_qa
qa/closed-book qa	freebase_qa jeopardy kilt_hotpotqa kilt_nq kilt_trex kilt_zsre lama-conceptnet lama-google_re lama-squad lama-trex numer_sense search_qa squad-no_context web_questions
qa/multiple-choice qa	ai2_arc aqua_rat codah commonsense_qa cosmos_qa dream hellaswag math_qa openbookqa qasc quail quarel quartz-no_knowledge quartz-with_knowledge race-high race-middle social_i_qa superglue-copa superglue-multirc swag wino_grande
qa/long-form qa	eli5-askh eli5-asks eli5-eli5
qa/MRC	adversarialqa biomrc quoref ropes superglue-record
qa/binary	boolq mc_taco
cg/summarization	gigaword multi_news samsum xsum
cg/dialogue	empathetic_dialogues kilt_wow
cg/other	spider wiki_bio wiki_split wikisql
other/linguistic phenomenon	blimp- anaphor_gender_agreement blimp-ellipsis_n_bar_1 blimp- sentential_negation_npi_scope
other/generate explanation	cos_e
other/slot_filling	ade_corpus_v2-dosage ade_corpus_v2-effect
other/entity linking	kilt_ay2
other/other	acronym_identification art aslg_pc12 break-QDMR break-QDMR-high-level common_gen crawl_domain crows_pairs definite_pronoun_resolution e2e_nlg_cleaned limit piqa proto_qa qa_srl