Different Tunes Played with Equal Skill:
Exploring a Unified Optimization Subspace for Delta Tuning

Jing Yi\textsuperscript{1}\textsuperscript{*}, Weize Chen\textsuperscript{1}\textsuperscript{*}, Yujia Qin\textsuperscript{1}\textsuperscript{*}, Yankai Lin\textsuperscript{2,3}, Ning Ding\textsuperscript{1}, Xu Han\textsuperscript{1},
Zhiyuan Liu\textsuperscript{1,4,5†}, Maosong Sun\textsuperscript{1,4,5†}, Jie Zhou\textsuperscript{6}

\textsuperscript{1}NLP Group, DCST, IAI, BNRIST, Tsinghua University, Beijing
\textsuperscript{2}Gaoling School of Artificial Intelligence, Renmin University of China, Beijing
\textsuperscript{3}Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing
\textsuperscript{4}International Innovation Center of Tsinghua University, Shanghai
\textsuperscript{5}Quan Cheng Laboratory \textsuperscript{6}Pattern Recognition Center, WeChat AI, Tencent Inc.

\{yi-j20, chenwz21, qyj20\}@mails.tsinghua.edu.cn
\{liuzy, sms\}@tsinghua.edu.cn

Abstract
Delta tuning (DET, also known as parameter-efficient tuning) is deemed as the new paradigm for using pre-trained language models (PLMs). Up to now, various DETs with distinct design elements have been proposed, achieving performance on par with fine-tuning. However, the mechanisms behind the above success are still under-explored, especially the connections among various DETs. To fathom the mystery, we hypothesize that the adaptations of different DETs could all be reparameterized as low-dimensional optimizations in a unified optimization subspace, which could be found by jointly decomposing independent solutions of different DETs. Then we explore the connections among different DETs by conducting optimization within the subspace. In experiments, we find that, for a certain DET, conducting optimization simply in the subspace could achieve comparable performance to its original space, and the found solution in the subspace could be transferred to another DET and achieve non-trivial performance. We also visualize the performance landscape of the subspace, and find that, there exists a substantial region where different DETs all perform well. Finally, we extend our analysis and show the strong connections between fine-tuning and DETs. The codes are publicly available at https://github.com/thunlp/Unified-DeltaTuning.

1 Introduction
Serving as the critical backbone for NLP, pre-trained language models (PLMs) achieve superior performance when adapted to downstream tasks (Han et al., 2021). Conventionally, the dominant way for such an adaptation is fine-tuning, which requires updating and storing all the parameters in PLMs. Consequently, with ever-larger PLMs continually being proposed (Raffel et al., 2019; Brown et al., 2020), fine-tuning becomes extremely computationally expensive. As an alternative, various delta tuning algorithms (DETs) spring up, which freeze most of the parameters and only optimize minimal adaptive parameters (Ding et al., 2022). Up to now, various DETs have been proposed, including introducing extra tunable neuron modules (Houlsby et al., 2019a), specifying partial parameters to be tunable (Ben Zaken et al., 2021) and re-parameterizing part of existing modules in PLMs (Hu et al., 2021b), etc. DETs extensively reduce the number of tunable parameters, and still achieves comparable downstream performance to fine-tuning.

Despite the success of DETs, the mechanism behind it remains unclear. An essential question is: how could the PLM adaptation using different DETs relate to each other? To answer this question, a direct exploration of the connections among different DETs is needed, but this would run into a problem: due to the versatile designs of DETs, the parameter space of various DETs is inherently different. To address the issue and investigate the above research question, we hypothesize that the adaptations of different DETs could be reparameterized as low-dimensional optimizations in a unified optimization subspace. In this sense, optimizing various DETs can all be viewed as finding optimal solutions within the same subspace. Our hypothesis is inspired by recent findings that despite owning huge amounts of parameters, PLMs have an extremely low intrinsic dimension (Aghajanyan et al., 2021; Qin et al., 2021). In this regard, optimizing a certain PET, which is typically a high-
dimensional optimization problem, could be equivalently re-parameterized as a low-dimensional optimization problem, while achieving non-trivial performance.

To find evidence for our hypothesis, we design an analysis pipeline as follows: we first independently obtain solutions for different DETs on a set of tasks. Then we learn to project these solutions to a desired subspace. Meanwhile, we also define a mapping from the subspace to each DET’s original space. We contend that if the found subspace is indeed shared among various DETs, then two conditions should be satisfied: (1) the optimizations of different DETs could be equivalently conducted in the found subspace and achieve non-trivial performance, and (2) the local optima of various DETs have a substantial intersection in the subspace, which means the solution obtained in the subspace using a certain DET could be directly transferred to other DETs. If both conditions are well-established for the found subspace, then we could validate the existence of the unified optimization subspace for DETs.

We conduct experiments on a series of representative NLP tasks, and demonstrate that in the found subspace:

- **Solutions are transferable.** The solution of a DET in the found subspace not only achieves comparable performance to that in its original DET space, but can be directly transferred to another DET, achieving non-trivial performance.

- **Local optima of DETs greatly overlap.** When visualizing the performance landscape, we find that there exists a substantial region where different DETs all perform well, indicating the close connections among different DETs.

- **Fine-tuning has strong connection with DETs.** We extend the above analysis to fine-tuning and show the strong connections between fine-tuning and DETs.

In general, our study is the first work to reveal the connections among different DETs and fine-tuning from the perspective of subspace optimization, and uncovers the underlying mechanism of PLMs’ downstream adaptation. We believe many applications such as the ensemble and transfer among various DETs can be well empowered by the unified optimization subspace. Our findings can be of interest to researchers who are working on designing better DETs, and may provide some guidance for using DETs in many real-world scenarios.

2 Background

**Delta Tuning.** DET has been regarded as the new paradigm for PLM adaptation. By training lightweight parameters, DET yields a compact and extensible model, and could achieve comparable performance to full-parameter fine-tuning. Up to now, various DET designs have sprung up. For instance, some introduce additional tunable modules after the feed-forward and attention modules in a PLM (Houlsby et al., 2019a; Pfeiffer et al., 2021); others prepend tunable prompt tokens into each attention layer (Li and Liang, 2021a) or only the embedding layer (Lester et al., 2021). Another line of work re-parameterizes existing modules with low-rank decompositions (Hu et al., 2021b). Recently, researchers demonstrate that existing DET algorithms can be combined simultaneously and achieve better performance (He et al., 2021; Mao et al., 2021).

To fathom the mechanisms behind DET, He et al. (2021) pioneered to explore the connections among different DETs. They formalize various DETs as different ways to compute the modifications on the hidden states and unify different DETs in terms of formulas. However, the unification in the formula does not reveal the essence of DETs’ success, and does not indicate that their internal mechanisms are unified. Our paper differs from theirs in that we explore whether DETs can be unified in terms of internal mechanisms through the lens of optimization. Specifically, we investigate whether the optimization of different DETs can be unified in a certain subspace.

**Intrinsic Dimension.** Intrinsic dimension (Li et al., 2018) estimates the minimum number of tunable parameters needed to reach a satisfying performance for neural networks. Instead of training networks in their native parameter space, they linearly re-parameterize all the tunable parameters $\theta_0$ in a randomly oriented subspace: $\theta \leftarrow \theta_0 + \text{Proj}(\theta_I)$, where $\text{Proj}: \mathbb{R}^{|\theta_I|} \rightarrow \mathbb{R}^{|\theta_0|}$ denotes a random projection ($|\theta_I| \ll |\theta_0|$). During optimization, only the low-dimensional vector $\theta_I$ is tuned. Considering that $|\theta_0|$ could be extremely large, making computation of the projection intractable, Aghajanyan
et al. (2021) reduce the computational complexity using Fastfood transformation (Le et al., 2013). In experiments, they find that for PLMs, a low-dimensional (e.g., $|\theta_0| \sim 10^5$) re-parameterization could achieve over 85% performance of fine-tuning ($|\theta_0|$ exceeds millions or even billions). Further, Qin et al. (2021) extend the tuning method from fine-tuning to prompt tuning (Lester et al., 2021). They demonstrate that the projection $\mathbf{P}_{\text{roj}}$ can be trained in order to approximate a better optimization subspace. Based on previous explorations of intrinsic subspace, we aim to validate the existence of a unified subspace for various tuning methods.

3 Preliminary

Following He et al. (2021), we investigate three representative DET algorithms to validate our hypothesis, including Adapter (Houlsby et al., 2019a), Prefix-tuning (Li and Liang, 2021a), and LoRA (Hu et al., 2021b). We will first recap the Transformer layer (Vaswani et al., 2017), and then give a brief review of the three DETs.

Transformer layer. PLMs generally have multiple Transformer layers, each consisting of a multi-head attention (MHA) and a feed-forward network (FFN). MHA is composed of $N_h$ attention heads, each containing a query / key / value weight matrix $\mathbf{W}_{q}^{(i)} / \mathbf{W}_{k}^{(i)} / \mathbf{W}_{v}^{(i)} \in \mathbb{R}^{d \times d_h}$, where $d$ denotes the model dimension and $d_h = d/N_h$. Given a sequence of $n$ vectors $\mathbf{X} \in \mathbb{R}^{n \times d}$, MHA parameterizes them into queries ($\mathbf{Q}^{(i)}$), keys ($\mathbf{K}^{(i)}$) and values ($\mathbf{V}^{(i)}$) as follows:

$$\mathbf{Q}^{(i)} = \mathbf{X} \mathbf{W}_{q}^{(i)}, \mathbf{K}^{(i)} = \mathbf{X} \mathbf{W}_{k}^{(i)}, \mathbf{V}^{(i)} = \mathbf{X} \mathbf{W}_{v}^{(i)}.$$

Each ($\mathbf{Q}^{(i)}, \mathbf{K}^{(i)}, \mathbf{V}^{(i)}$) triple is then fed into a self-attention function to obtain the $i$-th head’s representation $\mathbf{H}_i$. All head representations are then concatenated and combined using an output weight matrix $\mathbf{W}_o \in \mathbb{R}^{d \times d}$:

$$\mathbf{H}_i = \text{softmax}\left(\frac{\mathbf{Q}^{(i)} (\mathbf{K}^{(i)})^T}{\sqrt{d_h}} \mathbf{V}^{(i)}\right),$$

$$\mathbf{H} = \text{concat}(\mathbf{H}_1, ..., \mathbf{H}_{N_h}) \mathbf{W}_o.$$

The FFN module is a two-layer MLP:

$$\text{FFN}(\mathbf{H}) = \sigma(\mathbf{H} \mathbf{W}_1 + \mathbf{b}_1) \mathbf{W}_2 + \mathbf{b}_2,$$

where $\mathbf{W}_1 \in \mathbb{R}^{d \times d_m}$, $\mathbf{d} \in \mathbb{R}^{d_m}$, $\mathbf{W}_2 \in \mathbb{R}^{d_m \times d}$ and $\mathbf{b}_2 \in \mathbb{R}^{d}$. $d_m$ is often chosen larger than $d$.

Adapter. Adapter (Houlsby et al., 2019a) plugs in light-weight feed-forward networks in Transformer layers (after the MHA module and the FFN module). Every adapter layer typically consists of a down-projection matrix $\mathbf{W}_{\text{down}} \in \mathbb{R}^{d \times r_A}$, a non-linear activation function $f(\cdot)$, and an up-projection matrix $\mathbf{W}_{\text{up}} \in \mathbb{R}^{r_A \times d}$, where $r_A$ denotes the bottleneck dimension. Denote the input as $\mathbf{X} \in \mathbb{R}^{n \times d}$, adapter applies a residual connection as follows:

$$\mathbf{X} \leftarrow \mathbf{X} + f(\mathbf{X} \mathbf{W}_{\text{down}}) \mathbf{W}_{\text{up}}.$$

Prefix-tuning. Prefix-tuning (Li and Liang, 2021a) extends the queries $\mathbf{K}^{(i)}$ / the values $\mathbf{V}^{(i)}$ in every MHA module by prepping learnable prefix vectors $\mathbf{P}^{(i)}_k / \mathbf{P}^{(i)}_v \in \mathbb{R}^{m \times d_h}$ before them, where $m$ denotes the number of virtual tokens. The output of an attention head $\mathbf{H}_i$ can be re-formulated as:

$$\mathbf{H}'_i = \text{ATT}(\mathbf{Q}^{(i)}, [\mathbf{P}^{(i)}_k; \mathbf{K}^{(i)}], [\mathbf{P}^{(i)}_v; \mathbf{V}^{(i)}]),$$

where $[\cdot; \cdot]$ denotes concatenation.

LoRA. LoRA (Hu et al., 2021b) re-parameterizes the weight updates $\Delta \mathbf{W}$ of the weight matrix $\mathbf{W}$ in the MHA module with low-rank decompositions, i.e., $\Delta \mathbf{W} = \mathbf{W}_A \mathbf{W}_B$, where $\mathbf{W}_A \in \mathbb{R}^{d \times r_L}$ and $\mathbf{W}_B \in \mathbb{R}^{r_L \times d}$ are two learnable low-rank matrices, with $r_L$ being typically a small integer. For an input $\mathbf{X} \in \mathbb{R}^{n \times d}$, LoRA is formulated as:

$$\mathbf{X} \leftarrow \mathbf{X} + s \cdot \mathbf{X} \mathbf{W}_A \mathbf{W}_B,$$

where $s \geq 1$ is a scaling hyper-parameter.

4 Analysis Pipeline

As mentioned before, we consider three representative DETs: Adapter ($t_a$), Prefix-tuning ($t_p$), and LoRA ($t_l$). Each DET $t_k$ defines a set of tunable parameters $\theta_{t_k}$. To adapt a PLM to a specific downstream task $T$, we optimize $\theta_{t_k}$ to minimize the loss function $\mathcal{L}_{\text{task}}(\theta_{t_k} | \theta_0)$ defined by $T$, where $\theta_0$ denotes the pre-trained weights. To verify our hypothesis that there exists a unified optimization subspace where all DETs can achieve non-trivial performance, we propose a three-stage analysis pipeline (visualized in Figure 1), where the first stage is designed to approximate the desired subspace, so that in the second stage, the optimizations for different DETs could all be conducted in this subspace. This makes it possible to explore the connections of different DETs in the third stage. Following Qin et al. (2021), to validate the generality of the found subspace and avoid information
leakage, we approximate the subspace with a series of training tasks $T_{\text{train}}$, and conduct subsequent subspace optimization on unseen tasks $T_{\text{test}}$.

**Subspace Approximation.** To approximate the desired subspace, we decompose and then reconstruct independent DeT solutions of $T_{\text{train}}$. We first train DeTs in their original space, and for each task $T_i \in T_{\text{train}}$, we obtain three independent solutions: $\theta_{i_A}$, $\theta_{i_p}$, and $\theta_{i_L}$. Then we assign a down-projection $\text{Proj}_{i_A} : \mathbb{R}^{||\theta_{i_A}||} \rightarrow \mathbb{R}^y$ and an up-projection $\text{Proj}_{i_A}^+ : \mathbb{R}^y \rightarrow \mathbb{R}^{||\theta_{i_A}||}$ for each DeT $t_i$, where $y$ is the dimension of the intrinsic subspace. In practice, both down-projection and up-projection are MLP layers. Each down-projection decomposes a DeT solution into a low-dimensional intrinsic vector $I_{i_A} \in \mathbb{R}^y$:

$$I_{i_A} = \text{Proj}_{i_A}^i(\theta_{i_A}).$$

Three intrinsic vectors $I_{i_A}^i$, $I_{i_p}^i$, $I_{i_L}^i$ represent different local minima of $T_i$ in the same subspace. Ideally, if three DeTs can be unified in the subspace, then each vector $I_{i_A}^i$ could be used to reconstruct any DeT solution ($\theta_{i_A}^i$, $\theta_{i_p}^i$, or $\theta_{i_L}^i$). Therefore, to approximate such a subspace, we facilitate the interaction among different DeTs efficiently by dynamically sampling two random ratios $\alpha \in [0, 1], \beta \in [0, 1 - \alpha]$, and computing an interpolation of three intrinsic vectors of $T_i$:

$$I_{i_A}^i = \alpha \cdot I_{i_A}^i + \beta \cdot I_{i_p}^i + (1 - \alpha - \beta) \cdot I_{i_L}^i.$$

The interpolation is mapped by each up-projection $\text{Proj}_{i_A}^+$ to reconstruct the task solution for each DeT by minimizing the following loss function:

$$L_{\text{dist}}^i(\theta_{i_A}^i) = ||\theta_{i_A}^i - \theta_{i_A}^i||^2, \quad \theta_{i_A}^i = \text{Proj}_{i_A}^+(I_{i_A}^i).$$

To properly guide the reconstructed $\theta_{i_A}^i$ to solve task $T_i$, we also incorporate the original task loss $L_{\text{task}}$. The overall training objective can be formulated as follows:

$$L_{\text{pet}} = \sum_{i=1}^{||T_{\text{test}}||} \sum_{t_i \in \{t_A, t_p, t_L\}} L_{\text{dist}}^i(\theta_{i_A}^i) + L_{\text{task}}(\theta_{i_A}^i | \theta_0).$$

During this stage, only the down-projections and up-projections are optimized, and other parameters are kept frozen. When this stage finishes, the two projections can be seen as mappings between the unified subspace and each DeT’s original space.

**Subspace Optimization.** In the second stage, we investigate whether the optimization in the subspace could achieve comparable performance to
the optimization in the original space for unseen tasks $T_{test}$. If this holds, then we could empirically validate that the optimizations of different DeTs could be equivalently mapped in this subspace with a low level of error, and it is possible to explore the connections among DeTs in the next stage.

Specifically, we only retain the up-projection $\text{Proj}^\uparrow_{t}$ trained in the first stage. $\text{Proj}^\uparrow_{t}$ defines the mapping from the found subspace to the original DeT space. We keep both PLM and $\text{Proj}^\uparrow_{t}$ frozen during subspace optimization. After that, for each task $T_i \in T_{test}$, the optimization of $t_s$ can be conducted within the subspace defined by $\text{Proj}^\uparrow_{t}$ by merely tuning a randomly initialized intrinsic vector $I_{t,s}$, which is formulated as:

$$I_{t,s} = \arg \min_{I_t} \mathcal{L}_{i,\theta}^t(\text{Proj}^\uparrow_{t}(I_{t,s})) = \arg \min_{I_t} \mathcal{L}_{i,\theta}^t(I_{t,s}).$$

**Subspace Solution Transfer.** If the found subspace is shared among different DeTs, then the found solution $I_{t,s}$ in the subspace could be directly transferred to another DeT and achieve nontrivial performance. Taking the transferring between $t_A$ and $t_P$ as an example, for a task $T_i \in T_{test}$, we first conduct subspace optimization for $t_A$ and obtain a well-tuned intrinsic vector $I_{t,s}^*$. Then we directly transfer $I_{t,s}^*$ to $t_P$ utilizing its up-projection $\text{Proj}^\uparrow_{t}$, and obtain a $t_P$’s solution $\theta_{i,t_A\rightarrow t_P} = \text{Proj}^\uparrow_{t}(I_{t,s}^*)$.

5 Experiment

We conduct experiments on representative NLP tasks. We first introduce the experimental setups in §5.1, next we approximate the subspace and present the analysis in §5.2. Lastly, we explore the connection between DeTs and fine-tuning in §5.3.

5.1 Experimental Setups

**Training Setups.** We conduct experiments with both single-task and multi-task settings.

In the **single-task setting**, we approximate the unified optimization subspace using only one dataset, i.e., $|T_{train}| = 1$. Then we perform the subspace optimization and subspace solution transfer on unseen tasks. However, the subspace approximated with only one task may not generalize well to diverse unseen tasks (Qin et al., 2021). For example, the subspace approximated using a NLI task can hardly be generalized to a QA task. Therefore, for the single-task setting, we only evaluate the found subspace using the unseen tasks belonging to the same category of $T_{train}$.

Besides, we also experiment on the **multi-task setting**, where the unified subspace is approximated with diverse training tasks, i.e., $|T_{train}| > 1$. The unified DeT subspace found in the multi-task setting is expected to generalize to more diverse tasks than that in the single-task setting.

During subspace solution transfer, we choose the subspace solution that achieves the best transferring performance using the development set, and report its performance on the test set.

**Tasks and Datasets.** In the single-task setting, we experiment with 6 types of tasks, including:

- **Sentiment Analysis (SA):** SST-2 (Socher et al., 2013), Rotten Tomatoes (Pang and Lee, 2005), and Amazon Review (McAuley and Leskovec, 2013).

- **Natural Language Inference (NLI):** SciTail (Khot et al., 2018), MNLI (Williams et al., 2018), and RTE (Dagan et al., 2005).

- **Text Classification (TC):** WiC (Pilehvar and Camacho-Collados, 2019), and WSC (Levesque et al., 2012).

- **Paraphrase Detection (PD):** QQP(link), and MRPC (Dolan and Brockett, 2005).

- **Long-form QA (LF-QA):** ELI5-ELI5, ELI5-Askh, and ELI5-Asks (Fan et al., 2019).

- **Multiple-choice QA (MC-QA):** CoPA (Gordon et al., 2012), DREAM (Saha et al., 2018), QuaRTz (Tafjord et al., 2019b), and CODAH (Chen et al., 2019).

We include more diverse datasets in the multi-task setting, and randomly partition them into 60 training tasks $T_{train}$ and 9 test tasks $T_{test}$. More details are left in Appendix A.1.

**Evaluation Metrics.** For each dataset, we use the common evaluation metric, e.g., ROUGE-L for LF-QA, F1 for SA and NLI, ACCURACY for TC, PD and MC-QA. Denote $E_{ori}$ as the performance achieved by DeT in the original space, and $E_{sub}$ as the performance achieved by optimization within the subspace, we report the relative recovering performance ($\%$), i.e., $\frac{E_{sub}}{E_{ori}}$ in all experiments.
### Models

We use $T5_{\text{BASE}}$ (Raffel et al., 2020) as the backbone model, and unify all tasks into a text-to-text format without loss of generality. We set the dimension of the subspace to $4$ in single-task setting and $100$ in multi-task setting. During subspace optimization, only $4$ or $100$ free parameters are tuned, compared with $220M$ for fine-tuning. We choose the intrinsic dimension according to our preliminary experiment. The single-task performances of different intrinsic dimensions in $\{4, 8, 16\}$ do not vary much. The multi-task performance gets better when the intrinsic dimension increases. Practically, we find a dimension of $100$ strikes a satisfying balance between performance and computational resources. The details of implementation are shown in Appendix A.2.

### 5.2 Experimental Results

#### 5.2.1 Single-task Setting

**Subspace Optimization.** The results of subspace optimization are presented in Table 1. On average, for all three $D_T$Es, optimization within the subspace can recover more than $80\%$ performance of the original space. Among three $D_T$Es, Adapter achieves the best recovering performance ($\approx 90\%$), despite only tuning $4$ free parameters. This indicates that we have found a satisfying optimization subspace that could recover most of the performance of the original space$^1$, and the subspace can be generalized to unseen tasks $T_{\text{test}}$ belonging to the same category of $T_{\text{Train}}$.

---

$^1$Aghajanyan et al. (2021) deem $85\%$ as a satisfying recovering performance for an intrinsic subspace. Although the performance of our method could be a bit lower under certain cases, we contend that the performance is already non-trivial.

---

| $T_{\text{Train}}$ | $T_{\text{Test}}$ | Adapter | LoRA | Prefix | Avg. |
|-------------------|-----------------|---------|------|--------|------|
| SST-2 Rotten Tomatoes | 101.8 | 98.0 | 99.3 | 100.4 |
| Amazon Review | 100.1 | 96.9 | 98.2 | 97.7 |
| MNLI SciTail | 82.9 | 79.8 | 84.4 | 82.4 |
| RTE | 95.4 | 68.2 | 80.2 | 81.3 |
| WiC WSC | 70.6 | 57.6 | 77.1 | 68.4 |
| QQP MRPC | 85.4 | 84.3 | 83.1 | 84.3 |
| ELIS-ELI5 | 91.4 | 87.6 | 80.1 | 86.4 |
| ELI5-Askh | 96.6 | 95.0 | 94.9 | 95.5 |
| DREAM | 77.4 | 70.4 | 74.0 | 73.9 |
| QuaRTz CODAH | 75.6 | 78.5 | 74.7 | 76.3 |
| CoPA | 98.3 | 71.6 | 92.5 | 87.5 |
| Avg. | 88.5 | 80.9 | 85.3 | 84.9 |

Table 1: Relative performance (%) for subspace optimization under the single-task setting.

#### Subspace Solution Transfer.

Then we transfer the solution found with a source $D_T$ to other $D_T$Es. The results are presented in Table 2. On $6$ out of the $11$ tasks, transferring the subspace solution from a source $D_T$ to a target $D_T$ achieves more than $80\%$ recovering performance, and achieves $82.6\%$ on average across all tasks. This demonstrates that the transferred intrinsic vector yields $D_T$Es with non-trivial performance. In particular, in the category of sentiment analysis, the subspace of three $D_T$Es approximated on SST-2 serves as an excellent optimization subspace for similar tasks (R. Tomatoes and A. Review). Performing optimization in this subspace with an arbitrary source $D_T$ and transfer the found intrinsic vector to other $D_T$Es yield performance comparable to or even surpass the original $D_T$ space.

However, we also observe that the transferred $D_T$ on WSC does not perform very well, achieving only $64.9\%$ recovering performance. We argue that this may be due to the inherent difference between WiC ($T_{\text{train}}$) and WSC ($T_{\text{test}}$): WiC evaluates the quality of context-sensitive representations, while WSC is a coreference resolution task, which requires slightly distinct language skills from WiC. Besides, the performances of the transferred $D_T$ on DREAM are also slightly below the expectation, this may due to the domain differences between DREAM and the target tasks. Although they all belong to multi-choice QA, their domains differ significantly. In fact, these unwanted transferred performance can be substantially improved.

| $T_{\text{Train}}$ | $T_{\text{Test}}$ | $A \rightarrow L \rightarrow A \rightarrow P \rightarrow L$ | $A \rightarrow P \rightarrow A \rightarrow L \rightarrow P$ | $P \rightarrow A \rightarrow P \rightarrow L$ | Avg. |
|-------------------|-----------------|-----------------|-----------------|-----------------|------|
| SST-2 R. Tomatoes | 100.6 | 99.0 | 100.7 | 98.8 | 101.0 | 100.8 | 100.2 |
| A. Review | 97.1 | 97.7 | 97.7 | 98.1 | 97.6 | 96.7 | 97.5 |
| MNLI SciTail | 81.7 | 83.0 | 83.2 | 83.6 | 83.4 | 80.9 | 82.6 |
| RTE | 62.7 | 74.3 | 81.7 | 78.2 | 55.0 | 80.0 | 72.0 |
| WiC WSC | 72.7 | 54.3 | 58.8 | 57.1 | 76.5 | 69.7 | 64.9 |
| QQP MRPC | 83.7 | 65.7 | 83.7 | 69.1 | 84.8 | 85.4 | 78.7 |
| ELIS-ELI5 ELI5-Askh | 88.0 | 79.4 | 91.3 | 78.1 | 90.3 | 87.0 | 85.7 |
| ELIS-Asks | 95.9 | 95.4 | 91.3 | 92.5 | 97.8 | 96.1 | 94.8 |
| DREAM | 76.2 | 71.9 | 76.0 | 74.5 | 76.4 | 77.2 | 75.4 |
| QuaRTz CODAH | 63.0 | 61.6 | 74.3 | 58.1 | 83.9 | 69.0 | 68.3 |
| CoPA | 77.3 | 100.3 | 96.0 | 103.1 | 91.7 | 65.6 | 89.0 |
| Avg. | 81.7 | 80.2 | 85.0 | 81.0 | 85.3 | 82.6 | 82.6 |

Table 2: Relative performance (%) for subspace solution transfer under the single-task setting. $A$, $P$, and $L$ refer to Adapter, Prefix-tuning, and LoRA, respectively. As an example, $A \rightarrow L$ means we obtain $I_t^L$ by conducting subspace optimization with Adapter (source $D_T$), and then transfer the subspace solution to LoRA (target $D_T$) with the fixed up-projection, i.e., $\text{Proj}_{\text{LoRA}}^L(I_t^L)$.
in multi-task setting, as we will see in the next section.

In addition, we do not find a significant difference in the transferability among different DETs. In general, when serving as the source DET, Adapter has slightly worse transferability than other two DETs. Besides, the transferability of a DET seems to have a weak correlation to its performance of subspace optimization.

### 5.2.2 Multi-task Setting

To improve the subspace’s task-level generalization, we propose to approximate the subspace in a multi-task manner (60 training tasks in total), and test the generalization ability of the approximated subspace on 6 categories of unseen tasks. Note for the multi-task setting, the subspace optimization and subspace solution transfer are carried out within the same subspace for all the unseen tasks.

**Subspace Optimization.** The results of subspace optimization under the multi-task setting are shown in Table 3. In general, three DETs achieve non-trivial (88.7%) performance during subspace optimization on unseen tasks. Among three DETs, Adapter still performs the best, achieving 95.1% of its original performance. However, the performance of Prefix-tuning is about 10% poorer than Adapter and LoRA. We observe that when approximating the subspace, the loss of Prefix-tuning converges much slower than Adapter and LoRA, which may partially explain the poorer performance of Prefix-tuning. We leave further exploration of this phenomenon as future work.

**Subspace Solution Transfer.** The results are presented in Table 4. On 8 out of 9 tasks, DETs recover around or more than 80% their performance in the original space. The non-trivial results demonstrate that (1) for most of the investigated unseen tasks, the local optima found by a source DET can be directly transferred to a target DET and achieve non-trivial performance; (2) the subspace approximated with multiple training tasks can be well generalized to diverse unseen tasks. Both findings provide strong evidence for our hypothesis that different DETs can be re-parameterized into a unified optimization subspace.

We also observe that, the transferring performance on WSC is far better than that in the single-task setting, demonstrating the benefits of including diverse training tasks in subspace approximation. However, we still find that there are cases where the subspace solution of a source DET has poor transferability to another DET. For instance, the transferring performance of different DETs on ETHOS-Gender is only 63.0%. We conjecture that it is due to the gap between ETHOS-Gender and the training tasks. As demonstrated by Qin et al. (2021), increasing the diversity and number of training tasks could significantly improve the generalization ability of the subspace on unseen tasks. We expect future works to apply our analysis to more diverse training tasks.

Furthermore, comparing the transferability of different DETs, we find that similar to the single-task setting, Adapter is still slightly worse than the other two DETs. We also find there is no symmetry in the transferability. For example, the average transferring performance from Prefix-tuning to Adapter achieves 90.9% of its performance, while the performance in the opposite direction only reaches 76.5%.

### 5.2.3 Performance Landscape Visualization

From Tables 2 and 4, we observe non-trivial transferring performance among different DETs. These

| $T_{train}$ | Adapter | LoRA | Prefix | Avg. |
|-------------|---------|------|--------|------|
| Rotten Tomatoes | 99.7 | 98.2 | 100.3 | 99.4 |
| Yelp Polarity | 99.5 | 99.5 | 98.7 | 99.2 |
| WSC | 88.2 | 75.8 | 80.0 | 81.3 |
| AI2 ARC | 93.2 | 87.9 | 78.8 | 86.6 |
| QASC | 99.3 | 71.4 | 90.9 | 87.2 |
| QaRTz | 96.6 | 86.9 | 77.3 | 86.9 |
| BLiMP-ANA | 100.0 | 100.0 | 51.0 | 83.7 |
| ELI5-Asks | 99.9 | 99.6 | 94.3 | 97.6 |
| ETHOS-Gender | 79.6 | 88.9 | 59.0 | 75.8 |
| Avg. | 95.1 | 89.8 | 81.1 | 88.7 |

| $T_{test}$ | A→L | A→P | L→A | L→P | P→A | P→L | Avg. |
|-------------|------|------|------|------|------|------|------|
| Rotten Tomatoes | 97.5 | 98.6 | 98.5 | 95.9 | 98.3 | 96.2 | 97.2 |
| Yelp Polarity | 99.1 | 97.5 | 99.4 | 98.1 | 98.5 | 98.4 | 98.5 |
| WSC | 90.9 | 85.7 | 97.1 | 94.3 | 91.2 | 93.9 | 92.2 |
| AI2 ARC | 78.7 | 79.3 | 87.7 | 76.9 | 85.2 | 87.9 | 82.6 |
| QASC | 65.1 | 63.6 | 98.7 | 72.1 | 105.2 | 68.0 | 78.8 |
| QaRTz | 77.4 | 71.7 | 90.9 | 73.9 | 83.4 | 78.1 | 79.3 |
| BLiMP-ANA | 98.0 | 47.0 | 95.0 | 49.0 | 92.0 | 95.0 | 79.3 |
| ELI5-Asks | 90.6 | 89.7 | 89.6 | 95.0 | 95.8 | 87.4 | 91.4 |
| ETHOS-Gender | 55.5 | 57.8 | 53.0 | 73.3 | 68.7 | 69.8 | 63.0 |
| Avg. | 83.6 | 76.5 | 90.0 | 80.9 | 90.9 | 86.1 | 84.7 |
results demonstrate that the local optima of different DETs have a substantial overlap in the approximated subspace for the investigated unseen tasks. It is natural to be concerned about how large this overlap area is since a larger overlap may indicate closer connection of different DETs. Therefore, for both single-task and multi-task settings, we visualize the performance landscape to understand to what extent the local optima of different DETs in intrinsic subspace overlap with each other.

Specifically, denote \( I_0 \) as an origin, and \( u, v \) as two orthogonal directions. Let \( \alpha, \beta \) be two coordinates in the 2-dimensional space spanned by \( u \) and \( v \). Each solution \( I_0 + \alpha u + \beta v \) in the subspace can be mapped by the up-projection of DET \( t_s \) to the solution \( \text{Proj}^+_T(I_0 + \alpha u + \beta v) \) in the DET space. Denote \( E(\text{Proj}^+_T(I_0 + \alpha u + \beta v)) \) as the performance of the recovered DET, and \( E_{\text{PET}} \) as the average performance of the three DETs in their original space, we plot the relative performance along these two directions as follows:

\[
\mathcal{P} = \frac{1}{3} \sum_{t_s \in \{t_A,t_p,t_L\}} \frac{E(\text{Proj}^+_T(I_0 + \alpha u + \beta v))}{E_{\text{PET}}}. 
\]

If \( \mathcal{P} \) is high at \((\alpha, \beta)\), then it means three DETs all correspond to high performance at this point. Let \( I_{t_A}, I_{t_p}, I_{t_L} \) denote the optimal solution obtained by tuning each DET in the subspace independently. We visualize the performance landscape around these optimal solutions. Without loss of generality, we choose \( I_{t_A} \) as the origin, and select two orthogonal axes \( u, v \) as follows:

\[
u = I_{t_p} - I_{t_A}, \quad \bar{v} = I_{t_L} - I_{t_A}, \quad v = \frac{\bar{v} - \langle \bar{v}, u \rangle u}{\|\bar{v} - \langle \bar{v}, u \rangle u\|}.
\]

We traverse \( \alpha \) and \( \beta \) from \(-4 \) to \(4\) with a step size of \(0.4\). Due to the length limit, we only show in Fig. 2 the performance landscape on (1) SciTail of the single-task setting, (2) Yelp Polarity and AI2 ARC under the multi-task setting.

We observe that the subspace solutions of different DETs almost lie in the same optimal region for each task. Comparing the landscape of both single-task and multi-task settings, the highland area is much wider in the multi-task setting. This may explain the better transferability under the multi-task setting. In general, the above results demonstrate that the optimal solutions of different DETs indeed have a large overlap, otherwise there should not exist such a flat performance highland.

5.3 Extension to Fine-tuning

Finally, we extend our analysis pipeline to fine-tuning, and investigate its connection with DETs. However, directly training a down-projection and up-projection for fine-tuning encounters difficulty: if we still use an MLP layer (as introduced in §4) to implement the projections, the number of trainable parameters will be \( \mathcal{O}(yN) \), where \( N \) is the number of parameters of the PLM, and \( y \) is the dimension of our intrinsic subspace. Since PLMs generally contain tremendous parameters, it is intractable to train such an MLP. To alleviate the problem, we turn to using Fastfood transformation (Yang et al., 2015; Aghajanyan et al., 2021) as an alternative. It is an approximation for the linear projection, but requires far fewer parameters. Specifically, the up-

### Table 5: Relative performance (%) for subspace optimization and subspace solution transfer for different tuning methods on Rotten Tomatoes. The subspace is approximated on SST-2. We transfer the subspace solution from a source tuning method to a target one.

| Src. | Tgt. | Adapter | Prefix | LoRA | Fine-tune |
|------|------|---------|--------|------|----------|
| Adapter | 100.0 | 100.1 | 97.7 | 95.5 |
| Prefix | 101.2 | 100.4 | 97.7 | 95.0 |
| LoRA | 101.1 | 100.2 | 97.5 | 95.8 |
| Fine-tune | 101.1 | 99.7 | 97.1 | 96.1 |

Figure 2: Performance landscape visualization on three datasets (SciTail, Yelp Polarity, and AI2 ARC). We highlight the subspace solutions \((I_{t_A}, I_{t_L}, I_{t_p})\) found independently by the three DETs.
projection using Fastfood transformation can be formalized as follows:
\[ \tilde{\theta} = \theta_0 + IM, \quad M = HG\Pi HB, \]
where \( \tilde{\theta} \) denotes the tunable parameters in the original space, \( \theta_0 \) is the pre-trained weights, and \( I \) is the intrinsic vector. The Fastfood matrix \( M \) can be factorized as a Hadamard matrix \( H \), a random permutation matrix \( \Pi \), a diagonal matrix \( G \) with each element sampled from a standard normal distribution, and a diagonal matrix \( B \) with each element being \( \pm 1 \) with equal probability. Unlike previous work that uses a random and frozen Fastfood matrix (Yang et al., 2015; Aghajanyan et al., 2021), we optimize the matrix \( G \) to better approximate the desired subspace. For \( \text{DETs} \), the projection is implemented the same as before. More detailed implementations are described in Appendix A.4.

We perform subspace approximation on SST-2, and report the results of subspace optimization and subspace solution transfer on Rotten Tomatoes. As shown in Table 5, we find that: (1) all the \( \text{DETs} \) and fine-tuning can achieve satisfactory results in subspace optimization, and the solution found by any source tuning method can be transferred to other target tuning methods and achieve non-trivial performance. This demonstrates the close connection between \( \text{DETs} \) and fine-tuning in the approximated subspace. (2) Since the Fastfood transformation is an approximation for linear projection, its representation ability may be limited. Therefore, the transferring performance of fine-tuning is slightly inferior to other \( \text{DETs} \).

In general, the above finding implicates that all tuning methods may be re-parameterized to a unified optimization subspace, which also sheds light on the reason why different \( \text{DETs} \) optimize distinct sets of parameters, but all achieve comparable downstream performance to fine-tuning.

6 Conclusion

In this work, we explore the hypothesis that the adaptations of different delta tuning methods could all be re-parameterized as low-dimensional optimizations in a unified optimization subspace. The empirical results provide strong evidence for our hypothesis. We also extend our analysis to find the connection between fine-tuning and delta tuning. We hope our findings could provide insights for future research in designing better tuning methods and understanding the mechanisms behind PLM adaptation.

Acknowledgments

This work is supported by the National Key R&D Program of China (No. 2020AAA0106502) and Institute Guo Qiang at Tsinghua University.

Yujia Qin, Weize Chen, and Jing Yi designed the methods and wrote the paper. Jing Yi, Weize Chen, and Yujia Qin conducted the experiments. Yankai Lin, Zhiyuan Liu, Maosong Sun, and Jie Zhou advised the project. All authors participated in the discussion.

Limitations

The limitations of this paper are listed as follows:
- Under some settings, the recovering performance of the subspace optimization and subspace solution transfer can still be improved.
- We only conduct the experiments using \( T5_{\text{BASE}} \) model. We expect future works to apply our analysis pipeline to other kinds of PLMs, and PLMs with larger sizes.

References

Aarmen 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.

Tiago A. Almeida, José María G. Hidalgo, and Akebo Yamakami. 2011. Contributions to the study of sms spam filtering: New collection and results. In Proceedings of the 11th ACM Symposium on Document Engineering, DocEng ’11, page 259–262, New York, NY, USA. Association for Computing Machinery.

Victor Zhong an. 2017. Seq2sql: Generating structured queries from natural language usin. ArXiv preprint, abs/1709.00103.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. In Proceedings of the second PASCAL challenges workshop on recognising textual entailment, volume 6, pages 6–4. Venice.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinoza Anke, and Leonardo Neves. 2020. TweetEval: Unified benchmark and comparative evaluation for tweet classification. In Findings of the Association
Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. arXiv e-prints, pages arXiv–2106.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. In Proceedings of Text Analysis Conference.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Michael Boratko, Xiang Li, Tim O’Gorman, Rajarshi Das, Dan Le, and Andrew McCallum. 2020. ProtoQA: A question answering dataset for prototypical common-sense reasoning. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1122–1136, Online. Association for Computational Linguistics.

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

Michael Chen, Mike D’Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. CODAH: An adversarially-authored question answering dataset for common sense. In Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP, pages 63–69, Minneapolis, USA. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. 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 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. ArXiv preprint, abs/1803.05457.

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

Pradeep Dasigi, Nelson F. Liu, Ana Marasović, Noah A. Smith, and Matt Gardner. 2019. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. 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 5925–5932, Hong Kong, China. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. arXiv preprint arXiv:2203.06904.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In Proceedings of IWP Workshop.

Matthew Dunn, Levent Sagun, Mike Higgins, V. U. Güney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. ArXiv preprint, abs/1704.05179.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and ablative hierarchical model. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Manaal Faruqui and Dipanjan Das. 2018. Identifying well-formed natural language questions. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 798–803, Brussels, Belgium. Association for Computational Linguistics.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third pascal recognizing textual entailment challenge. In Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, pages 1–9, Prague. Association for Computational Linguistics.
Andrew Gordon, Zornitsa Kozareva, and Melissa Roemmele. 2012. SemEval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 394–398, Montréal, Canada. Association for Computational Linguistics.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhiqong Qu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open*, 2:225–250.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzabela, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyana, and Sylvain Gelly. 2019a. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzabela, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyana, and Sylvain Gelly. 2019b. Parameter-efficient transfer learning for NLP. In *ICML*.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chinyew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021a. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021b. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural CRF model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960, Online. Association for Computational Linguistics.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. 2020. Qasc: A dataset for question answering via sentence composition.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018, pages 5189–5197. AAAI Press.

Neema Kotonya and Francesca Toni. 2020. Explainable automated fact-checking for public health claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7740–7754, Online. Association for Computational Linguistics.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Quoc Le, Tamás Sarlós, Alex Smola, et al. 2013. Fastfood-approximating kernel expansions in log-linear time. In *Proceedings of the international conference on machine learning*, volume 85, page 8.

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

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR’12, page 552–561. AAAI Press.

Quentin Lhoest, Albert Villa nova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. *arXiv preprint arXiv:2109.02846*. 

3358
Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. 2018. Measuring the intrinsic dimension of objective landscapes. arXiv preprint arXiv:1804.08838.

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

Xiang Lisa Li and Percy Liang. 2021b. 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.

Xin Li and Dan Roth. 2002. Learning question classifiers. In COLING 2002: The 19th International Conference on Computational Linguistics.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-Trained Language Models. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6862–6868, Online. Association for Computational Linguistics.

Annie Louis, Dan Roth, and Filip Radlinski. 2020. “i’d rather just go to bed”: Understanding indirect answers. arXiv preprint arXiv:2010.03450.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Walenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. Journal of the Association for Information Science and Technology, 65(4):782–796.

Yuning Mao, Lambert Mathias, Rui Hou, Anjnad Almahairi, Hao Ma, Jiawei Han, Wen-tau Yih, and Madian Khabas. 2021. Unipel: A unified framework for parameter-efficient language model tuning. arXiv preprint arXiv:2110.07577.

Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2020. Hatesplain: A benchmark dataset for explainable hate speech detection. ArXiv preprint, abs/2012.10269.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In Proceedings of the 7th ACM conference on Recommender systems, pages 165–172.

Clara H. McCreeery, Namit Katariya, Anitha Kannan, Manish Chablanii, and Xavier Amatriain. 2020. Effective transfer learning for identifying similar questions: Matching user questions to COVID-19 FAQs. In KDD ’20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 21-27, 2020, pages 3458–3465. ACM.

Ioannis Mollas, Zoe Chrysoupolou, Stamatios Karlos, and Grigoris Tsoumakas. 2020. Ethos: an online hate speech detection dataset. ArXiv preprint, abs/2006.08328.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 1797–1807, Brussels, Belgium. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL ’05), pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models’ factual predictions. In Automated Knowledge Base Construction.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jonas Pfeiffer, Ashwarya Kamath, Andreas Rücklélé, 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.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. 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 1267–1273, Minneapolis, Minnesota. Association for Computational Linguistics.

Amir Pouran Ben Beyesh, Franck Dernoncourt, Quan Hung Tran, and Thien Huu Nguyen. 2020.
What does this acronym mean? introducing a new dataset for acronym identification and disambiguation. In Proceedings of the 28th International Conference on Computational Linguistics, pages 3285–3301. Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yujia Qin, Xiaozhi Wang, Yusheng Su, Yankai Lin, Ning Ding, Zhiyuan Liu, Juanzi Li, Lei Hou, Peng Li, Maosong Sun, et al. 2021. Exploring low-dimensional intrinsic task subspace via prompt tuning. arXiv preprint arXiv:2110.07867.

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.

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

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Anna Rogers, Olga Kovaleva, Matthew Downey, and Anna Rumschisky. 2020. Getting closer to ai complete question answering: A set of prerequisite real tasks. Proceedings of the AAAI Conference on Artificial Intelligence, 34(05):8722–8731.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. Proceedings of the AAAI Conference on Artificial Intelligence, 34:8732–8740.

Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. 2018. CARER: Contextualized affect representations for emotion recognition. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3687–3697, Brussels, Belgium. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In International Conference on Machine Learning, pages 4596–4604. PMLR.

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

Oyvind Tafjord, Peter Clark, Matt Gardner, Wen-tau Yih, and Ashish Sabharwal. 2019a. Quarel: A dataset and models for answering questions about qualitative relationships. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01):7063–7071.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019b. QuaRTz: An open-domain dataset of qualitative relationship questions. 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 5941–5946, Hong Kong, China. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Sowmya Vajjala and Ivana Lučić. 2018. Onestopenglish corpus: A new corpus for automatic readability assessment and text simplification. In Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications, pages 297–304.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Ilia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008.

William Yang Wang. 2017. “liar, liar pants on fire”: A new benchmark dataset for fake news detection. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 432–426, Vancouver, Canada. Association for Computational Linguistics.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BliMP: The benchmark of linguistic minimal pairs for English. Transactions of the
Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. In Proceedings of the 3rd Workshop on Noisy User-generated Text, pages 94–106, Copenhagen, Denmark. Association for Computational Linguistics.

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

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

Zichao Yang, Marcin Moczulski, Misha Denil, Nando De Freitas, Alex Smola, Le Song, and Ziyu Wang. 2015. Deep fried convnets. In Proceedings of the IEEE international conference on computer vision, pages 1476–1483.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Hao Zhang, Jae Ro, and Richard Sproat. 2020. Semi-supervised URL segmentation with recurrent neural networks pre-trained on knowledge graph entities. In Proceedings of the 28th International Conference on Computational Linguistics, pages 4667–4675, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sheng Zhang, X. Liu, J. Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. Record: Bridging the gap between human and machine commonsense reading comprehension. ArXiv preprint, abs/1810.12885.
A Appendices

A.1 Datasets
All the datasets used in the multi-task setting are listed in Table 12 and Table 13. All these datasets are downloaded from Huggingface Datasets (Lhoest et al., 2021).

A.2 Hyper-parameters and Network Structure for §5.2

The down-projection $\text{Proj}_1^{\downarrow}$ is a two-layer MLP, with the first linear layer $f_{\text{down-1}} : \mathbb{R}^{|\theta_t|} \rightarrow \mathbb{R}^y$, and the second linear layer $f_{\text{down-2}} : \mathbb{R}^y \rightarrow \mathbb{R}^y$, where $y$ is the dimension of intrinsic subspace.

We use $\text{tanh}$ as the activation function between $f_{\text{down-1}}$ and $f_{\text{down-2}}$. The up-projection $\text{Proj}_1^{\uparrow}$ is a single linear layer $f_{\text{up}} : \mathbb{R}^y \rightarrow \mathbb{R}^{|\theta_t|}$. Note for all the linear layers of the projections, we do not include the bias term.

To ensure that the number of parameters is consistent across the three DETs, we set $r_A$ as 12, $r_L$ as 10, $m$ as 120, and $d_p$ as 24. $d_p$ refers to the hidden dimension of the two-layer MLP that is used to re-parameterize Prefix vectors $F_K^{(i)}/P_V^{(i)}$, see Li and Liang 2021b for more details. The meanings of other notations are the same as those in §3. In this way, the number of parameters of $\theta_{1A}^{\uparrow}, \theta_{1P}^{\uparrow}$ and $\theta_{1L}^{\uparrow}$ are all 1105920. Moreover, following Houlsby et al. 2019b, we choose a SiLU activation function for Adapter. Following Hu et al. 2021a, we set the scaling factor $s$ in LoRA as 1.6. In our implementation, LoRA is applied to Q and V matrices in the MHA module.

During subspace approximation, for the multi-task setting, we randomly sample 20000 instances from the original training set of each dataset, and blend them together to form the multi-task training set. Similarly, we sample another 240 instances from each dataset to form the validation set. We set the learning rate as $1 \times 10^{-4}$, batch size as 4. We train the model for 1 epoch and evaluate on validation set for every 1000 steps. For the single-task setting, we perform grid search using the learning rates in $\{1 \times 10^{-5}, 5 \times 10^{-5}\}$ and set the batch size as 8. We train the model for a maximum step of 100000, and evaluate on validation set for every 1000 steps. When conducting subspace optimization, we perform grid search using the learning rate in $\{1 \times 10^{-2}, 5 \times 10^{-2}\}$ and set the batch size as 8. We train the model for a maximum of 5000 steps and evaluate on validation set every 500 steps.

| $\tau_{\text{train}}$ | $\tau_{\text{test}}$ | Adapter | LoRA | Prefix | Avg. |
|----------------------|---------------------|---------|------|--------|------|
| SST-2                | Rotten Tomatoes     | 99.3    | 98.5 | 95.5   | 97.8 |
| Amazon Review        |                     | 96.7    | 96.5 | 97.1   | 96.8 |
| MNLI                 | SciTail             | 82.0    | 74.1 | 73.3   | 76.5 |
| RTE                  |                     | 71.5    | 80.0 | 71.3   | 74.3 |
| WSC                  | WSC                 | 61.8    | 84.8 | 85.7   | 77.4 |
| QQP                  | MRPC                | 85.4    | 81.5 | 68.5   | 78.5 |
| ELIS-ELIS            | ELIS-Askh           | 89.1    | 85.1 | 77.5   | 83.9 |
| ELIS-ELIS            | ELIS-Aks            | 93.9    | 94.4 | 88.2   | 92.2 |
| DREAM                | CODAH               | 63.6    | 63.9 | 61.6   | 63.0 |
|                     | QuaRTs              | 74.2    | 72.4 | 73.9   | 73.5 |
|                     | CoPA                | 90.4    | 66.0 | 101.4  | 85.9 |
| Avg.                 |                     | 82.5    | 81.6 | 81.3   | 81.8 |

Table 6: Relative performance (%) for subspace optimization under single-task setting with constructed subspace.

| $\tau_{\text{test}}$ | Adapter | LoRA | Prefix | Avg. |
|----------------------|---------|------|--------|------|
| Rotten Tomatoes       | 92.9    | 88.5 | 66.4   | 82.6 |
| Yelp Polarity         | 97.0    | 96.2 | 90.0   | 94.4 |
| WSC                   | 94.1    | 90.9 | 82.9   | 89.3 |
| A2 ARC                | 77.2    | 73.4 | 75.9   | 75.5 |
| QASC                  | 73.8    | 64.0 | 59.8   | 65.9 |
| QuaRTs                | 78.0    | 78.5 | 74.3   | 76.9 |
| BLIMP-ANA             | 94.0    | 98.0 | 47.0   | 79.7 |
| ELIS-Asks             | 83.9    | 84.6 | 79.5   | 82.7 |
| ETHOS-Gender          | 54.6    | 55.4 | 74.3   | 61.4 |
| Avg.                  | 82.8    | 81.1 | 72.2   | 78.7 |

Table 7: Relative performance (%) for subspace solution transfer under under multi-task setting with constructed subspace.

We train the model using Adafactor (Shazeer and Stern, 2018) with a constant learning rate in all experiments. Intrinsic dimension $y$ (the dimension of the approximated subspace) is set to 4 in single-task setting and 100 in multi-task setting. We additionally set a ratio $\alpha = 10$ to balance the reconstruction loss $L_{\text{dist}}(\theta_t)$ and original task loss $L_{\text{task}}(\theta_t, \theta_0)$, i.e., $L_{\text{pet}} = 10 \times L_{\text{dist}}(\theta_t) + L_{\text{task}}(\theta_t, \theta_0)$. All experiments are carried out on NVIDIA 32GB V100 GPU.

A.3 Simplification of Subspace Approximation

From a standpoint of analysis, it is necessary to start our pipeline from independent solutions of different DETs and then explore their connections. Now that we have validated the existence of the unified optimization subspace, for practical uses, we could simplify the original pipeline by enforcing different DETs to share the same intrinsic vector. Specifically, we jointly train three DETs, and
The steps of analysis is the same as in §5.2. In the experiment of subspace approximation on glue-
we minimize the loss:

\[ \mathcal{L} = \frac{1}{3} \sum_{i=1}^{\lvert T \rvert_{\text{max}}} \mathcal{L}_{\text{task}}^i(\theta_{\text{ts}}^i | \theta_0). \]

In this way, we can directly approximate the desired unified subspace, omitting the procedure of first obtaining solutions for different DETs, and we do not need to assign a down-projection for each DET. Subspace optimization and subspace solution transfer can then be carried out using this subspace.

We present the results of subspace optimization and subspace solution transfer for both the single-
task and multi-task settings in Tables 6 to 9. We find that in general, all DETs still achieve non-
trivial performance in both subspace optimization and subspace solution transfer, which means the simplification does not influence the representation ability of the found subspace. This simplified pipeline is the cornerstone of our analysis of the connection between fine-tuning and DETs. Since the simplified procedure does not require training a down-projection, the total number of tunable parameters can be further reduced.

### A.4 HyperParameters of simplified approximation experiments

In the simplified approximation experiments, we use different learning rates for the shared intrinsic vector and DETs in subspace approximation. We set the learning rate as $5 \times 10^{-5}$ for the shared intrinsic vector, and $1 \times 10^{-4}$ for DETs. The batch size is 16 in the single-task setting and 8 in the multi-task setting. We train the model for a maximum of 100000 and validate every 1000 steps. For subspace optimization, we perform grid search on learning rate in \{5 $\times 10^{-2}$, 1 $\times 10^{-2}$, 5 $\times 10^{-3}$, 1 $\times 10^{-3}$\}. We set batch size as 16. To keep 3 DETs’ number of parameters consistent, we set $r_A$ as 12, $r_L$ as 10, $m$ as 24, and $dp$ as 120. Other hyperparameters are kept consistent with the main experiments.

### A.5 Implementation Details for Extension to Fine-tuning

We use the simplified pipeline introduced in Appendix A.3 to further reduce the number of trainable parameters. That is, an intrinsic vector $I^t$ for the $i$-th task is set to be a trainable parameter, and is shared among fine-tuning and different DETs. The steps of analysis is the same as in §5.2. In the experiment of subspace approximation on glue-

---

Table 8: Relative performance (%) for subspace solution transfer under single-task setting with constructed subspace.

| Task        | $T_{\text{train}}$ | $T_{\text{test}}$ | A→L | A→P | L→A | L→P | P→A | P→L | Average |
|-------------|---------------------|-------------------|-----|-----|-----|-----|-----|-----|---------|
| SST-2       | R. Tomatoes         | 99.7              | 98.5| 100.9| 98.8| 101.2| 99.7| 99.8|         |
|             | A. Review           | 98.2              | 98.2| 98.3 | 98.2| 98.1 | 97.3| 98.1|         |
| MNLI        | SciTail             | 74.9              | 77.4| 83.5 | 77.0| 83.9 | 74.8| 78.6|         |
|             | RTE                 | 89.1              | 80.2| 58.7 | 87.1| 70.6 | 87.3| 78.8|         |
| WiC         | WSC                 | 84.8              | 74.3| 70.7 | 74.3| 82.3 | 72.8| 76.5|         |
| QQP         | MRPC                | 84.8              | 67.4| 79.8 | 76.4| 78.0 | 5.6 | 65.3|         |
| ELIS-ELI5   | ELI5-ELI5          | 85.1              | 79.9| 93.4 | 79.9| 92.6 | 85.9| 86.1|         |
|             | ELI5-Askh           | 93.1              | 91.5| 97.9 | 91.5| 97.3 | 93.8| 94.2|         |
| DREAM       | QuaRTz              | 80.0              | 74.9| 77.7 | 74.2| 78.1 | 80.2| 77.5|         |
|             | CODAH               | 73.0              | 77.8| 75.9 | 78.2| 80.3 | 70.9| 76.0|         |
|             | CoPA                | 67.7              | 107.5| 86.8 | 103.8| 84.4 | 63.8| 85.7|         |
| Avg.        |                     | 84.6              | 84.3| 84.0 | 85.4| 86.1 | 75.6| 83.3|         |

Table 9: Relative performance (%) for subspace solution transfer under multi-task setting with constructed subspace.

| Task        | $T_{\text{train}}$ | $T_{\text{test}}$ | A→L | A→P | L→A | L→P | P→A | P→L | Average |
|-------------|---------------------|-------------------|-----|-----|-----|-----|-----|-----|---------|
| Rotten Tomatoes | 93.6              | 88.0              | 83.5| 87.9 | 84.6| 95.0 | 88.8|     |         |
| Yelp-Polarity | 97.5              | 92.0              | 98.0| 87.0 | 97.9 | 97.1 | 94.9|     |         |
| WiC         | 93.9              | 94.3              | 58.9| 94.3 | 64.7| 93.9 | 83.3|     |         |
| A2 ARC      | 82.7              | 78.3              | 88.4| 77.7 | 83.6| 81.4 | 82.0|     |         |
| QASC        | 85.2              | 68.2              | 103.2| 70.7 | 99.9 | 85.2 | 85.4|     |         |
| QuaRTz      | 79.1              | 76.9              | 81.4| 76.4 | 81.4| 77.4 | 78.8|     |         |
| BLMP-ANA    | 99.0              | 51.0              | 96.0| 52.0 | 96.0 | 96.0 | 81.7|     |         |
| ELIS-Asks   | 90.5              | 89.5              | 89.9| 88.8 | 87.9 | 90.5 | 89.5|     |         |
| ETHOS-Gender | 52.8              | 0.0               | 51.1| 0.0  | 50.1 | 57.4 | 35.2|     |         |
| Avg.        | 86.0              | 70.9              | 83.4| 70.5 | 82.9 | 86.0 | 80.0|     |         |

Table 10: Absolute performance for different tuning methods under the single-task setting.

| Task        | $T_{\text{train}}$ | $T_{\text{test}}$ | Adapter | LoRA | Prefix | Fine-tune | Average |
|-------------|---------------------|-------------------|---------|------|--------|----------|---------|
| SST-2       | Rotten Tomatoes     | 89.2              | 89.3 | 90.0 | 90.0  | 89.8    |         |
|             | Amazon Review       | 96.2              | 96.6 | 96.6 | 97.0  |         |         |
| MNLI        | SciTail             | 94.0              | 93.8 | 93.0 | 94.8  |         |         |
|             | RTE                 | 78.4              | 79.1 | 72.7 | 80.4  |         |         |
| WiC         | WSC                 | 65.4              | 63.5 | 67.3 | 67.3  |         |         |
| QQP         | MRPC                | 87.3              | 87.3 | 87.3 | 89.7  |         |         |
| ELIS-ELI5   | ELI5-ELI5          | 11.5              | 12.0 | 12.6 | 13.0  |         |         |
|             | ELI5-Askh           | 15.0              | 15.2 | 15.1 | 15.3  |         |         |
| DREAM       | CODAH               | 41.2              | 43.0 | 45.0 | 45.2  |         |         |
|             | QuaRTz              | 67.0              | 67.1 | 68.5 | 69.4  |         |         |
|             | CoPA                | 60.4              | 56.4 | 58.4 | 59.2  |         |         |
| Avg.        |                     | 65.6              | 65.2 | 65.0 | 66.6  |         |         |
A.6 Absolute Performance for Different Tuning Methods

In the main paper, we report the relative performance of subspace optimization. In this section, we list the absolute performance of different tuning methods (Adapter, Prefix-Tuning, and LoRA) in Table 10 and Table 11 for reference.

| Task         | Adapter | LoRA | Prefix | Fine-tune |
|--------------|---------|------|--------|-----------|
| Rotten Tomatoes | 89.2    | 89.3 | 90.0   | 89.8      |
| Yelp Polarity  | 97.3    | 97.4 | 97.8   | 97.9      |
| WSC           | 65.4    | 63.5 | 67.3   | 67.3      |
| AI2 ARC       | 31.2    | 32.4 | 32.2   | 31.3      |
| QASC          | 33.0    | 37.8 | 33.3   | 43.6      |
| QuaRTz        | 67.0    | 67.1 | 68.5   | 69.4      |
| BLiMP-ANA     | 100.0   | 100.0| 100.0  | 100.0     |
| ELI5-Asks     | 15.0    | 15.2 | 15.1   | 15.3      |
| ETHOS-Gender  | 79.9    | 79.9 | 77.4   | 74.5      |
| Avg.          | 65.6    | 66.2 | 65.4   | 67.0      |

Table 11: Absolute performance for different tuning methods under the multi-task setting.
Table 12: The training tasks involved in our multi-task setting.

| Split | Task Name                        | Reference                                      |
|-------|----------------------------------|------------------------------------------------|
|       |amazon review                    | McAuley and Leskovec 2013                      |
|       |financial_phrasebank             | Malo et al. 2014                              |
|       |glue-sst2                        | Socher et al. 2013                            |
|       |imdb                             | Maas et al. 2011                              |
|       |emotion                          | Saravia et al. 2018                           |
|       |tweet_eval-offensive             | Barbieri et al. 2020                          |
|       |tweet_eval-stance_climate        | Barbieri et al. 2020                          |
|       |ethos-directed_vs_generalized    | Mollas et al. 2020                            |
|       |ethos-race                       | Mollas et al. 2020                            |
|       |hateplain                        | Mathew et al. 2020                            |
|       |glue-mnli                        | Williams et al. 2018                          |
|       |glue-qnli                        | Rajpurkar et al. 2016                         |
|       |glue-wnli                        | Faruqui and Das 2018                          |
|       |superglue-re                      | Giampiccolo et al. 2007; Bentivogli et al. 2009|
|       |health_fact                      | Kotonya and Toni 2020                         |
|       |liar                             | Wang 2017                                     |
|       |glue-qqp                         | (link)                                        |
|       |medical_questions_pairs          | McCreery et al. 2020                          |
|       |paws                             | Zhang et al. 2019                             |
|       |circa                            | Louis et al. 2020                             |
|       |onestop_english                  | Vajjala and Lučić 2018                        |
|       |trec-finegrained                 | Li and Roth 2002; Hovy et al. 2001            |
|       |wiki_auto                        | Jiang et al. 2020                             |
|       |google_wellformed_query          | Faruqui and Das 2018                          |
|       |sms_spam                         | Almeida et al. 2011                           |
|       |superglue-wic                    | Pilehvar and Camacho-Collados 2019            |
|       |lama-google_re                   | Petroni et al. 2019, 2020                     |
|       |numer sense                      | Lin et al. 2020                               |
|       |search_qa                        | Dunn et al. 2017                              |
|       |web_questions                    | Berant et al. 2013                            |
|       |boolq                            | Clark et al. 2019                             |
|       |codah                            | Chen et al. 2019                              |
|       |commonsense_qa                   | Talmor et al. 2019                            |
|       |cosmos_qa                        | Huang et al. 2019                             |
|       |dream                            | Saha et al. 2018                              |
|       |hellaswag                        | Zellers et al. 2019                           |
|       |sciq                             | Welbl et al. 2017                             |
|       |quail                            | Rogers et al. 2020                            |
|       |quaref                           | Tajford et al. 2019a                          |
|       |race-high                        | Lai et al. 2017                                |
|       |superglue-copa                   | Gordon et al. 2012                            |
|       |wino_grande                      | Sakaguchi et al. 2020                         |
|       |eli5-eli5                        | Fan et al. 2019                                |
|       |hotpot_qa                        | Yang et al. 2018                               |
|       |quoref                           | Dasigi et al. 2019                            |
|       |superglue-record                 | Zhang et al. 2018                             |
|       |multi_news                       | Fabbri et al. 2019                            |
|       |xsum                             | Narayanan et al. 2018                         |
|       |spider                           | Yu et al. 2018                                 |
|       |wikisql                          | an 2017                                       |
|       |blimp-anaphor_gender_agreement   | Warstadt et al. 2020                          |
|       |blimp-ellipsis_n_bar_1           | Warstadt et al. 2020                          |
|       |blimp-irregular_past PARTICIPLE_ADJECTIVES | Warstadt et al. 2020                      |
|       |blimp-wh_questions_object_gap    | Warstadt et al. 2020                          |
|       |cos_e                            | Rajani et al. 2019                             |
|       |acronym_identification            | Pouran Ben Veyseh et al. 2020                 |
|       |crawl_domain                     | Zhang et al. 2020                             |
|       |proto_qa                         | Boratko et al. 2020                           |
|       |qa_srl                           | He et al. 2015                                 |
Table 13: The test tasks involved in our multi-task setting.

| Split       | Task Name                          | Reference              |
|-------------|------------------------------------|------------------------|
| Test tasks  | rotten_tomatoes                    | Pang and Lee 2005      |
|             | yelp_polarity                       | Zhang et al. 2015      |
|             | ethos-gender                        | Mollas et al. 2020     |
|             | superglue-wsc                       | Levesque et al. 2012   |
|             | ai2_arc                             | Clark et al. 2018      |
|             | qasc                                | Khot et al. 2020       |
|             | quartz-no_knowledge                 | Tafjord et al. 2019b   |
|             | eli5-asks                           | Fan et al. 2019        |
|             | blimp-anaphor_number_agreement      | Warstadt et al. 2020   |