Exploring Mode Connectivity for Pre-trained Language Models

Yujia Qin[1*], **Cheng Qian**[1*], **Jing Yi**[1*], **Weize Chen**[1], **Yankai Lin**[2,3†], **Xu Han**[1], **Zhiyuan Liu**[1,4,5†], **Maosong Sun**[1,4,5†], **Jie Zhou**[6]
[1]NLP Group, DCST, IAI, BNRIST, Tsinghua University, Beijing
[2]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing
[3]Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing
[4]International Innovation Center of Tsinghua University, Shanghai
[5]Quan Cheng Laboratory [6]Pattern Recognition Center, WeChat AI, Tencent Inc.
{qyj20, qianc20, yi-j20}@mails.tsinghua.edu.cn

Abstract

Recent years have witnessed the prevalent application of pre-trained language models (PLMs) in NLP. From the perspective of parameter space, PLMs provide generic initialization, starting from which high-performance minima could be found. Although plenty of works have studied how to effectively and efficiently adapt PLMs to high-performance minima, little is known about the connection of various minima reached under different adaptation configurations. In this paper, we investigate the geometric connections of different minima through the lens of *mode connectivity*, which measures whether two minima can be connected with a low-loss path. We conduct empirical analyses to investigate three questions: (1) how could hyperparameters, specific tuning methods, and training data affect PLM's mode connectivity? (2) How does mode connectivity change during pre-training? (3) How does the PLM's task knowledge change along the path connecting two minima? In general, exploring the mode connectivity of PLMs conduces to understanding the geometric connection of different minima, which may help us fathom the inner workings of PLM downstream adaptation. The codes are publicly available at https://github.com/thunlp/Mode-Connectivity-PLM.

1 Introduction

Recent years have witnessed the prevalent application of pre-trained language models (PLMs) in NLP (Han et al., 2021), with the state-of-the-art across various NLP tasks consistently being pushed (Devlin et al., 2019; Liu et al., 2019b; Raffel et al., 2020). Through large-scale self-supervised training, PLMs acquire versatile semantic (Liu et al., 2019a) and syntactic (Tenney et al., 2019) knowledge, which could be utilized when conducting transfer learning on downstream tasks.

From the perspective of parameter space, PLMs provide generic initialization for downstream adaptation. Starting from the initialization, many high-performance minima can be found through gradient-based optimization. Up to now, plenty of works have studied how to effectively and efficiently adapt PLMs to high-performance minima, including adjusting hyperparameters (Liu and Wang, 2021), conducting transfer learning using auxiliary training data (Pruksachatkun et al., 2020), tuning PLMs in a parameter-efficient way (Ding et al., 2022), etc. Under different adaptation configurations, PLMs may finally reach local minima distributed in highly distinct regions. Although these minima all correspond to excellent performance (low loss), little has been known about their geometric connection in the parameter space.

A straightforward way to explore such geometric connection is to look into the loss landscape around different minima, which is inherently intractable due to the high dimensionality brought by the tremendous parameter size of PLMs. Instead of probing the full landscape, we propose to investigate the relation of different minima through the lens of *mode connectivity* (Garipov et al., 2018), which measures whether two different minima can be connected via a parametric path, along which the loss of the downstream task remains low. Exploring the mode connectivity of PLMs contributes to understanding the geometric connection among different minima. Such connection reflects the inherent relation of various adaptation configurations and may help us fathom the inner workings of PLM downstream adaptation under different settings.

To the best of our knowledge, systematic studies for the mode connectivity of PLMs are still lacking.

[*]Indicates equal contribution.
[†]Corresponding author.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 6726 - 6746

In this paper, we first investigate what factors may affect PLM's mode connectivity by answering the following research questions:

- (*Q1*) How could different adaptation configurations (hyperparameters, tuning methods, and training data) affect PLM's mode connectivity?
- (*Q2*) How does mode connectivity change during pre-training?

We first consider the mode connectivity when different minima are trained on the same dataset. We investigate the effects of several hyperparameters (e.g., training data order, initialization of the tunable parameters, training steps, etc.) on PLM's mode connectivity, and find that among these factors, initialization has the greatest impact. In addition, we show that fine-tuning leads to better mode connectivity than parameter-efficient delta tuning (e.g., adapter (Houlsby et al., 2019)).

Then we extend the connectivity analysis to minima trained on different datasets. We demonstrate that: (1) the mode connectivity is good for two minima trained on data belonging to the same distribution, but without overlap of specific instances. This means instead of memorizing training data, PLMs learn advanced task-level knowledge during training, and mode connectivity originates from the high overlap of task knowledge of two minima. (2) Although two minima trained on different tasks are inherently disconnected, pre-training gradually pulls the optimal regions of different tasks closer in an implicit way. This phenomenon may help explain PLM's excellent cross-task transferability.

Beyond exploring the effects that could affect PLM's mode connectivity, we also study the intrinsic properties of model solutions between two minima, which leads to the third question:

- (*Q3*) How does the PLM's task knowledge change along the path connecting two minima?

In the experiments, we observe that for two minima obtained independently on two tasks, when traversing from the minima trained on a source task to that of a target task, a PLM suffers from catastrophic forgetting (McCloskey and Cohen, 1989) on the source task, and gradually absorbs the knowledge from the target task. Besides, PLMs prioritize forgetting those elusive source knowledge and acquiring easy-to-grasp target knowledge.

In general, to fathom the connection of minima reached under different settings, we conduct empirical studies on the mode connectivity of PLMs. We also show that our findings may have potential significance in broad research areas, such as designing better ensemble methods for PLMs, understanding the task-level transferability of PLMs and revealing the mechanism of PLM downstream adaptation. We expect our evaluation setup and findings could inspire more future works in this field.

2 Related Work

Adaptation Strategies of PLMs. To effectively and efficiently utilize the knowledge learned during pre-training, many strategies have been developed to better tune a PLM, including: (1) *hyperparameter search*, which aims to find an optimal hyperparameter configuration through traditional grid search or modern automated search (Liu and Wang, 2021); (2) *pre-finetuning*, which trains PLMs on intermediate auxiliary tasks before fine-tuning on a target task (Pruksachatkun et al., 2020; Aghajanyan et al., 2021). In this way, PLMs achieve better downstream performance by taking advantage of cross-dataset knowledge transfer; (3) *prompt learning*, which casts downstream tasks into the form of the pre-training objective by leveraging natural language prompts (Brown et al., 2020; Schick and Schütze, 2021a,b). Prompt learning exhibits superior performance especially under the few-shot and zero-shot scenarios; (4) *delta tuning* (also known as parameter-efficient tuning). Optimizing all the parameters of a PLM is computationally cumbersome. As a lightweight alternative, delta tuning optimizes only a few tunable parameters and keeps other parameters frozen, achieving comparable performance to fine-tuning (Ding et al., 2022).

Although plenty of adaptation strategies have been developed to better tune a PLM, little is understood about the connection of local minima reached under different training configurations. In this work, we take the first step by utilizing mode connectivity as the analysis tool.

Mode Connectivity for Neural Networks. Despite being extremely high-dimensional, the loss landscape of neural networks exhibits a simple geometric pattern of mode connectivity (Garipov et al., 2018; Freeman and Bruna, 2017; Draxler et al., 2018). It is shown that starting from different initialization, the local minima obtained by gradient-based optimizations are often connected

by low-loss paths, along which high-performance solutions could be easily found and ensembled to achieve better performance (Garipov et al., 2018). These paths are typically *non-linear* curves, which require a process of curve finding with task supervision. Excellent mode connectivity indicates that different minima are not isolated points in the parameter space, but essentially form a connected manifold (Draxler et al., 2018).

Frankle et al. (2020) further contend that from the same initialization, local minima obtained with different training data order can be connected by a *linear* low-loss path, reducing the burden of curve finding. Such a phenomenon is dubbed as linear mode connectivity, which is closely related to lottery ticket hypothesis (Frankle and Carbin, 2019), and has direct implications for continual learning (Mirzadeh et al., 2020). Compared with the non-linear counterpart, linear mode connectivity is a stronger constraint, requiring that the convex combination of two minima stay in the same loss basin.

Previous works typically study mode connectivity using non-pretrained models in the field of computer vision. Until recently, Neyshabur et al. (2020) observe linear mode connectivity on pre-trained vision models. Despite the great efforts spent, a systematic understanding of the mode connectivity of PLMs is still lacking. In this paper, we focus on investigating the effects that would influence PLM's mode connectivity and analyze the knowledge variation along the connecting path. Different from existing works that study mode connectivity for minima trained on the same dataset, we additionally extend the analysis to different datasets.

3 Mode Connectivity Evaluation

Preliminaries. Consider adapting a PLM on a downstream task, let C_1 and C_2 be two distinct sets of training configurations that may differ in hyperparameters or data. We use C_1 and C_2 to train two copies of the PLM independently. The specific tuning method determines the tunable parameters θ_0. After training, θ_0 is adapted to $\theta_{C_1} \in \mathbb{R}^{|\theta_0|}$ and $\theta_{C_2} \in \mathbb{R}^{|\theta_0|}$, respectively, where $|\theta_0|$ denotes the total number of tunable parameters.

Connecting Path. Based on Frankle et al. (2020), the same initialization generally leads to a linear low-loss path between two minima. Besides, compared with the non-linear counterpart, linearity is a more favorable property, which indicates a closer connection between different minima. Therefore, our first step is to investigate whether PLMs have good *linear* mode connectivity. Specifically, assume a continuous curve $\phi(\alpha) : [0, 1] \rightarrow \mathbb{R}^{|\theta_0|}$ connecting θ_{C_1} and θ_{C_2}, satisfying $\phi(0) = \theta_{C_1}$ and $\phi(1) = \theta_{C_2}$, we consider a linear path as follows:

$$\phi(\alpha) = (1 - \alpha) \cdot \theta_{C_1} + \alpha \cdot \theta_{C_2}. \quad (1)$$

Connectivity Criterion. After defining the curve connecting both minima, we traverse along the curve and evaluate the loss and performance of the interpolations. We deem two minima θ_{C_1} and θ_{C_2} mode connected if there does not exist a significant loss barrier or performance drop along the defined curve between θ_{C_1} and θ_{C_2}. In the experiments, we evaluate evenly distributed interpolations on $\phi(\alpha)$.

4 Empirical Analysis

In this section, we conduct experiments to investigate the aforementioned research questions.

Q1. (a) How could different hyperparameters and the specific tuning method affect the mode connectivity of PLMs?

We first investigate the effects of several hyperparameters that could affect PLM's mode connectivity, including (1) training data order, initialization of tunable parameters, training step (main paper), (2) learning rate and batch size (appendix B.1). To explore the effects of the specific tuning method, we experiment with both fine-tuning and a representative delta tuning method, i.e., adapter (Houlsby et al., 2019). Adapter inserts tunable modules into a PLM and keeps other parameters fixed during adaptation. Unless otherwise specified, we mainly conduct the experiments using T5_{BASE} (Raffel et al., 2020), and choose two representative NLP tasks (MNLI (Williams et al., 2018) and ReCoRD (Zhang et al., 2018)).

In each experiment, the training configurations of the two endpoints only differ in one hyperparameter, while other settings are kept the same for a fair comparison. To explore the effects of training steps, we evaluate the performance when both endpoints are trained for {10k, 30k, 50k} steps, respectively. We evaluate 24 evenly distributed interpolations and 2 endpoints along a linear path, i.e., we evaluate a series of $\phi(\alpha)$, where $\alpha \in \{\frac{0}{25}, \frac{1}{25}, ..., \frac{25}{25}\}$. Since we find that the trends of loss and performance are generally highly correlated (i.e., a performance drop corresponds to a loss barrier), we

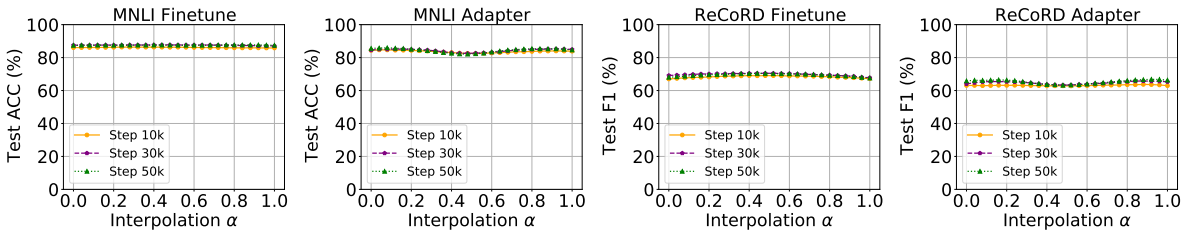


Figure 1: The performance of linear interpolations between two minima trained with different training data order.

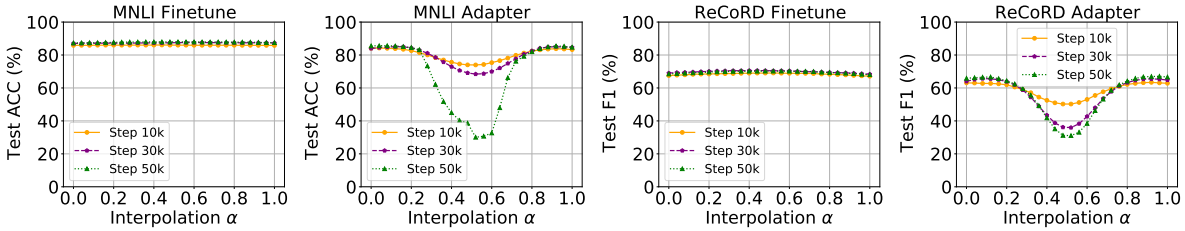


Figure 2: The performance of linear interpolations between two minima trained with different initialization.

report the performance in the main paper and leave the results of loss in appendix D. All experiments are conducted 3 times with random seeds and we report the average results on test sets. For more training details, please refer to appendix C.

Effects of Training Data Order. PLM's downstream adaptation generally involves mini-batch gradient-based optimization, where training samples are learned in a random order. To explore its effect, we adapt two copies of a PLM with two different random data order. Then we visualize the performance of linear interpolations in Figure 1, from which we observe that for fine-tuning, both endpoints are well connected by a linear path; while for adapter tuning, there exists a slight but negligible performance drop near the midpoint. In general, we conclude that **local minima are well connected under different random training data order**.

Effects of Initialization. Before downstream adaptation, additional parameters (e.g., extra modules defined by delta tuning, the classification head, etc.) may be introduced; in addition, Wu et al. (2022) recently show that adding noise to the pre-trained weights improves the fine-tuning performance on downstream tasks. Thus, both fine-tuning and delta tuning require proper initialization for the tunable parameters. Since different initialization could lead to distinct optimization trajectories, we explore the effects of initialization on PLM's mode connectivity.

Specifically, for those newly introduced modules, we randomly initialize them with a Gaussian distribution; for those pre-trained weights that require tuning, we add a random Gaussian noise. Two endpoints are initialized with the same configuration (e.g., mean and standard deviation of the Gaussian distribution), but different random seeds. The linear interpolation results are depicted in Figure 2, from which we observe that the mode connectivity of fine-tuning is generally good; while for adapter tuning, there exists a significant performance drop between two differently initialized minima. This means starting from different initialization, PLMs tend to reach non-connected local minima in the parameter space, especially for delta tuning. In short, **initialization of tunable parameters has a great impact on mode connectivity**.

Effects of Training Step. As mentioned before, the experiments in Figure 1 and 2 are conducted when both minima are trained for {10k, 30k, 50k} steps. Comparing the results at different training steps, we observe that (1) longer training leads to poorer connectivity for adapter tuning under certain cases; while (2) the mode connectivity of fine-tuning is good at different steps. In appendix B.2, we further show that (1) the mode connectivity becomes poorer when one endpoint is trained with more steps while the other is trained with fixed steps, and (2) with the training step increasing, the Euclidean distance between two minima is also prolonged, which may partially explain the poorer mode connectivity.

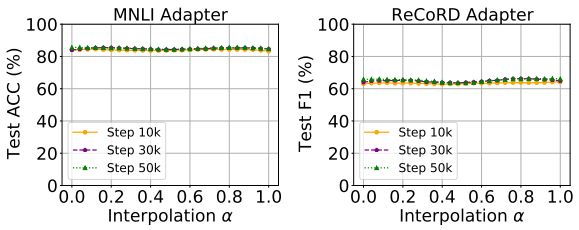


Figure 3: The performance of interpolations along a non-linear path connecting two minima, which are trained with adapter tuning from different initialization.

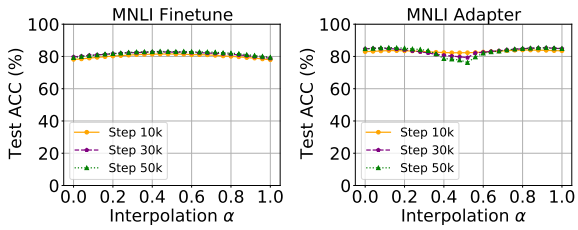


Figure 4: Linear mode connectivity analysis for two minima trained with in-distribution MNLI data. The results on ReCoRD are left in appendix B.4.

Effects of Tuning Method. Comparing the results of fine-tuning and adapter tuning in Figure 1 and 2, we observe that in general, the linear mode connectivity of fine-tuning is better than adapter tuning. In other words, when using fine-tuning, PLMs are more likely to be optimized to linearly-connected minima. A similar phenomenon also occurs for minima trained with different learning rates or batch sizes (see appendix B.1). Considering that adapter optimizes only 2.38% parameters than fine-tuning, we hypothesize that more tunable parameters may yield better mode connectivity and leave more explorations as future work.

Different Minima are Generally Connected by a Non-linear Path. Considering that linearity is a strong constraint for mode connectivity, even if a direct linear path connecting two minima incurs a high loss, both minima may still be connected by a low-loss non-linear path. To explore whether this holds for PLMs, we follow the setting of tuning adapters with different initialization, which has been shown in Figure 2 to have poor linear mode connectivity. We try to use the supervision from the downstream task to find a low-loss parametric path connecting two endpoints θ_{C_1} and θ_{C_2}. Following Garipov et al. (2018), we consider a quadratic Bezier curve defined as follows:

$$\phi_\theta(\alpha) = (1-\alpha)^2 \cdot \theta_{C_1} + 2\alpha(1-\alpha)\theta + \alpha^2 \cdot \theta_{C_2}, \quad (2)$$

where $\theta \in \mathbb{R}^{|\theta_0|}$ denotes tunable parameters of the curve. During curve finding, θ_{C_1} and θ_{C_2} are kept frozen, and only θ is optimized. Denote $\mathcal{L}$ as the loss function of the task, the training objective is to minimize the expectation of loss on the curve over a uniform distribution $\mathrm{U}(0,1)$, i.e., $\mathbb{E}_{\alpha \in \mathrm{U}(0,1)} \mathcal{L}(\phi_\theta(\alpha))$. For more details, please refer to appendix A.

We visualize the performance of the interpolation on the found Bezier curve in Figure 3. We observe that the two minima are well-connected by the found Bezier curve, without a significant performance drop. In fact, such a low-loss Bezier curve exists for minima reached under various different settings (see more experiments in appendix B.3). Given the above results, we conjecture that **there may exist multiple loss basins which are connected via a low-loss non-linear path, instead of a linear path. For most of the minima within the same loss basin, their convex combination also lies in this basin**. In this sense, if two minima are connected linearly, then both of them probably lie in the same basin; otherwise in different basins (e.g., the case of adapter tuning with different initialization).

Q1. (b) What are the effects of training data?

In previous experiments, we focus on the connectivity of two minima trained with the same dataset. From now on, we extend the mode connectivity to two minima trained on different datasets, focusing on two facets: data overlap and data domain.

Effects of Data Overlap. PLMs have been demonstrated to be adept at memorizing the training data (Carlini et al., 2021, 2022). To show that the connectivity of both minima does not originate from PLM's memorization, we explore whether such mode connectivity still exists when two minima are obtained on data belonging to the same distribution, but without overlap of specific training samples. Specifically, we partition the original training data of MNLI into two equal splits. Then we adapt two copies of $\mathrm{T5}_{\mathrm{BASE}}$ on either split using the same training configurations. The experiments are conducted using both fine-tuning and adapter tuning.

The performance of linear interpolations is recorded in Figure 4. The results show that two

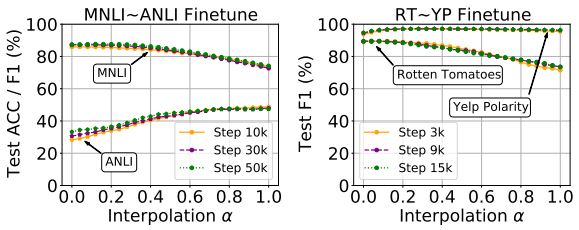


Figure 5: Linear mode connectivity for two minima fine-tuned on different data distributions of the same task. Left: $\alpha = 0$ / $\alpha = 1$ denotes the minimum of MNLI / ANLI. Right: $\alpha = 0$ / $\alpha = 1$ denotes the minimum of Rotten Tomatoes / Yelp Polarity.

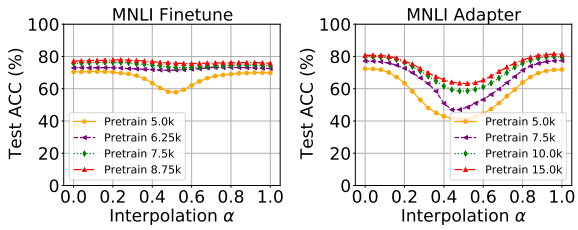


Figure 6: The change of linear mode connectivity at different pre-training steps. We illustrate the performance of linear interpolations of two minima trained on MNLI using different initialization.

minima are well connected for both tuning methods, demonstrating that **mode connectivity does not originate from PLM's memorization of specific training data**; instead, during training, PLMs learn advanced task-level knowledge, and the **connectivity reflects the high overlap of task knowledge of two local minima**.

Effects of Data Domain. PLMs are shown to generalize well on out-of-distribution data (Hendrycks et al., 2020), implying the connection of minima trained with different data distributions. To gain a deeper understanding, we choose two natural language inference datasets (MNLI and ANLI (Nie et al., 2020)), and two sentiment analysis datasets (Rotten Tomatoes (Pang and Lee, 2005) and Yelp Polarity (Zhang et al., 2015)) sourced from different domains. Then we fine-tune two copies of T5_{BASE} on two datasets of the same task, and evaluate the linear mode connectivity between two minima. Note previous works typically study mode connectivity on the same dataset; while in our setting, we extend the analysis by evaluating the interpolations on two datasets.

The results are shown in Figure 5. Take the NLI task as an example, starting from one endpoint ($\alpha = 0$) of a source task (MNLI), with the interpolation approaching the other endpoint ($\alpha = 1$) of the target task (ANLI), the performance on MNLI / ANLI exhibits almost a monotonous drop / rise. Besides, there does not exist a performance valley where the performance is significantly lower than both endpoints. Intuitively, **the performance change reflects the variation of the interpolation's task knowledge along the connecting path**. Due to the difference in data domain, the task knowledge of two endpoints only partially overlap with each other. When traversing from the source minimum to the target minimum, PLM suffers from catastrophic forgetting on the source knowledge, but gradually absorbs target knowledge, leading to the performance drop / rise on the source / target task. We defer more in-depth analyses to Q3.

Q2. How does mode connectivity change during pre-training?

Previous works demonstrate that compared with random initialization, the initialization obtained by pre-training leads to a wider loss basin after downstream adaptation (Hao et al., 2019; Neyshabur et al., 2020). Intuitively, if a local minimum lies in a more flat basin, it should be easier to connect with other minima. In this sense, pre-training may be closely related to mode connectivity. To investigate this, we re-train a $\text{RoBERTa}_{\text{BASE}}$ model from scratch and explore how mode connectivity changes at different pre-training steps. We follow the pre-training setting of Liu et al. (2019b), with more details described in appendix C.4.

Pre-training Facilitates Mode Connectivity. We select a series of checkpoints at different pre-training steps. For each checkpoint, we adapt two copies on MNLI using different initialization, and evaluate the performance of their linear interpolations. From Figure 6, we observe that for both fine-tuning and adapter tuning, with the pre-training step becoming larger, the mode connectivity of the PLM becomes better. This implies that pre-training implicitly facilitates the mode connectivity. Specifically, when using fine-tuning, there does not exist a performance drop for checkpoints pre-trained with more than 6.25k steps. Considering that pre-training with a batch size of 2048 for 6.25k steps corresponds to almost 5% the computational cost of BERT (a batch size of 256 for 1000k steps), we

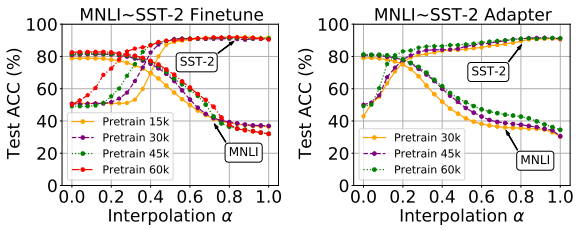


Figure 7: The change of linear mode connectivity at different pre-training steps. We illustrate the performance of linear interpolations of two minima trained on MNLI and SST-2. $\alpha = 0$ / $\alpha = 1$ denotes the minimum of MNLI / SST-2.

conclude that PLMs acquire good mode connectivity at an early stage of pre-training.

Pre-training Pulls Task Boundary Closer. Further, we look into the performance variation along a linear path between two minima trained on two different tasks (MNLI and SST-2 (Socher et al., 2013)). Similarly, we choose a series of checkpoints at different pre-training steps. Then for each checkpoint, we adapt two copies on MNLI and SST-2 under the same setting, and conduct linear interpolation between two adapted weights. We also conduct experiments on MNLI and QQP (link) in appendix B.6. We evaluate the performance of each interpolation on both tasks and illustrate the results in Figure 7. It can be derived that (1) due to the inherent difference of both tasks, the minimum obtained on one task achieves the performance near random guess ($\approx 50\%$ for SST-2 and $\approx 33.3\%$ for MNLI) on another task. This indicates that minima of different tasks are disconnected. (2) In addition, there is a strong general trend that for a checkpoint pre-trained longer, the intersection of both tasks' high-performance regions becomes wider. In other words, the boundaries of both tasks' optimal regions are gradually pulled closer by pre-training. (3) For the checkpoint pre-trained with 60k steps, we do not observe a region where the performance on both tasks is poor. This means starting from the initialization of a PLM pre-trained with enough steps, the optimal regions of various downstream tasks are closely packed. This finding may help explain PLM's cross-task transferability, and we leave more discussions in § 5.

Q3. How does the task knowledge change along the path connecting two minima?

Having shown that mode connectivity reflects the high overlap of task knowledge of different minima, we further investigate the knowledge variation along the path connecting two minima. To quantify a model's task knowledge, we resort to the *memorization* of the training data as a rough estimate. In experiments, we evaluate two minima obtained on data of different distributions as mentioned in Q1. (b)[1].

Specifically, we adapt two copies of T5_{BASE} on MNLI (source task) and ANLI (target task), respectively. Denote θ_s and θ_t as two minima trained on the source dataset $\mathcal{D}_s = \{x_i, y_i\}_{i=1}^{|\mathcal{D}_s|}$ and the target dataset $\mathcal{D}_t = \{x_i, y_i\}_{i=1}^{|\mathcal{D}_t|}$. We investigate the knowledge variation from θ_s to θ_t by choosing 4 evenly distributed linear interpolations (ϕ_1, ϕ_2, ϕ_3, ϕ_4) and 2 endpoints (ϕ_0, ϕ_5), i.e., $\phi_j = \theta_s + \frac{j}{5} \cdot (\theta_t - \theta_s)$, $j \in \{0, 1, ..., 5\}$, where $\phi_0 = \theta_s$, $\phi_5 = \theta_t$. Then we measure whether each source training sample $x_i \in \mathcal{D}_s$ is memorized (correctly classified) by ϕ_j. We find empirically that with ϕ_j approaching θ_t, training samples of $\mathcal{D}_s$ are gradually forgotten, but seldom re-memorized under this setting. Therefore, we only record those newly forgotten samples for ϕ_j (i.e., those classified correctly by ϕ_{j-1} but wrongly by ϕ_j) and denote them as $\mathcal{F}_j$. Similarly, we denote those newly memorized samples of $\mathcal{D}_t$ as $\mathcal{M}_j$ (i.e., those classified wrongly by ϕ_{j-1} but correctly by ϕ_j).

After that, we characterize the role of each sample using *dataset cartography* (Swayamdipta et al., 2020). For a brief introduction, each sample of $\mathcal{D}_s$ ($\mathcal{D}_t$) is characterized by the training dynamics of θ_s (θ_t). Take $\mathcal{D}_s$ as an example, assume we train the PLM for E epochs on $\mathcal{D}_s$, and the weights of the PLM are adapted to $\theta_s(e)$ after the e-th epoch, where $1 \le e \le E$. For each training instance $(x_i, y_i) \in \mathcal{D}_s$, denote $\mathcal{P}_{\theta_s(e)}(y_i|x_i)$ as the probability $\theta_s(e)$ assigns to the true label, we record the PLM's prediction after each epoch and calculate two statistics:

• `confidence` measures how confidently the PLM assigns the true label to a given input, it is defined as the mean probability of the true label:

$$\mu_i = \frac{1}{E} \sum_{e=1}^{E} \mathcal{P}_{\theta_s(e)}(y_i|x_i).$$

• `variability` captures how consistently the PLM

[1]We choose this setting because (1) there does not exist a performance valley between two minima, which means the knowledge is properly combined, and (2) the knowledge of both tasks is diverse enough.

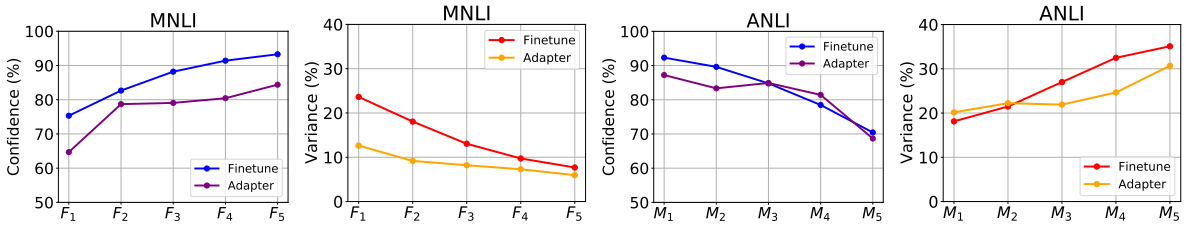


Figure 8: The results of knowledge variation for linear interpolations (ϕ_1, ϕ_2, ϕ_3, ϕ_4) between two minima (ϕ_0, ϕ_5) adapted on MNLI and ANLI. We leave experiments on other tasks as future work.

judges each training instance, it is defined using the standard deviation of the true label's probability:

$$\sigma_i = \sqrt{\frac{\sum_{e=1}^{E}(\mathcal{P}_{\theta_s(e)}(y_i|x_i) - \mu_i)^2}{E}}.$$

After obtaining both statistics for each training sample of $\mathcal{D}_s$ and $\mathcal{D}_t$, we illustrate the average statistics for newly forgotten / memorized samples ($\mathcal{F}_j$ / $\mathcal{M}_j$) in Figure 8. We observe that with ϕ_j approaching θ_t, the average confidence of the newly forgotten data gradually increases, while the variability gradually drops; symmetrically, the average statistics of the newly learned data exhibit an opposite trend. According to Swayamdipta et al. (2020), instances with high confidence but low variability are generally easy-to-learn ones; while those with low confidence are generally ambiguous or hard-to-learn data. In this regard, when gradually leaving the source minimum, **the PLM prioritizes forgetting the source knowledge of those difficult instances, and then forgets the source knowledge of the easy-to-learn data**. On the contrary, **the easy-to-learn target knowledge is learned before the elusive and obscure target knowledge**.

5 Discussion

Weight Averaging. The property of linear mode connectivity is related to recent explorations of weight averaging (Wortsman et al., 2021, 2022; Matena and Raffel, 2021), which combines independently fine-tuned models in the parameter space. In this way, the knowledge from multiple models can be merged. Our findings have direct implications for designing better weight averaging methods: (1) for two minima, weight averaging can be seen as choosing the midpoint on the linear path. We have shown that a non-linear curve may have better mode connectivity under certain cases. This implicates that linear interpolation may not find the optimal combination despite its simplicity; instead, there may exist better methods to ensemble weights (see experiments in appendix B.8); (2) our findings on the effects of different training configurations can also inspire choosing more appropriate models (with better mode connectivity) to ensemble.

Task-level Transferability. Although PLMs are demonstrated to have excellent cross-task transferability (Vu et al., 2020; Pruksachatkun et al., 2020; Poth et al., 2021; Su et al., 2022), it is still under-explored why PLMs have such an ability. Our findings that pre-training implicitly pulls the task boundary closer may help explain this phenomenon. Since the optimal regions of various tasks are packed closely, PLMs are easier to traverse across the task boundary, without getting blocked by a loss barrier.

Knowledge Quantification. Investigating the knowledge variation along the connecting path helps better understand how different model knowledge is merged. Quantifying the task knowledge of various models may also provide insights for research topics like knowledge distillation (Hinton et al., 2015) and knowledge transfer (Weiss et al., 2016). While we use training data memorization as a rough estimate for task knowledge, it would be interesting to explore whether there exist more granular methods, such as knowledge probing (Petroni et al., 2019; Liu et al., 2019a).

6 Conclusions

In this paper, we conduct empirical analyses on the mode connectivity of PLMs, aiming to fathom the connection of minima reached under different settings. We investigate how different downstream adaptation configurations and pre-training affect PLM's mode connectivity. In addition, we explore the knowledge variation along the path connecting different minima. In general, exploring the mode connectivity of PLMs contributes to understanding

the inner workings of PLM downstream adaptation. We expect our evaluation setup and analyses could inspire more future explorations in this field.

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 designed the experiments and wrote the paper. Cheng Qian and Jing Yi conducted the experiments. Yankai Lin, Zhiyuan Liu, Maosong Sun, and Jie Zhou advised the project. All authors participated in the discussion.

The authors would like to thank anonymous reviewers for their valuable feedback.

Limitations

There are some limitations not well addressed in this paper:

- The goal of this paper is to investigate the connection among different minima. However, since we use mode connectivity as the analysis tool, we only investigate the connection between two minima at a time. In this regard, it would be interesting to develop more advanced tools to explore the connection among multiple minima simultaneously, which is left as future work.
- We do not give a strict mathematical definition for "good mode connectivity". For instance, we do not set a specific threshold of performance drop (e.g., $> 5\%$ for the case of "not well-connected minima"). We argue that this is because, all of our experimental results are significant enough, thus there is no need to follow a strict definition.

References

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5799–5811, Online and Punta Cana, Dominican Republic. 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*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. *ArXiv preprint*, abs/2202.07646.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

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, 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*.

Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred A. Hamprecht. 2018. Essentially no barriers in neural network energy landscape. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1308–1317. PMLR.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. 2020. Linear mode connectivity and the lottery ticket hypothesis. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3259–3269. PMLR.

C. Daniel Freeman and Joan Bruna. 2017. Topology and geometry of half-rectified network optimization.

In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P. Vetrov, and Andrew Gordon Wilson. 2018. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8803–8812.

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

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2019. Visualizing and understanding the effectiveness of BERT. 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 4143–4152, Hong Kong, China. Association for Computational Linguistics.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2744–2751, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *ArXiv preprint*, abs/1503.02531.

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.

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.

Quentin Lhoest, Albert Villanova 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*.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019a. Linguistic knowledge and transferability of contextual 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 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Xueqing Liu and Chi Wang. 2021. An empirical study on hyperparameter optimization for fine-tuning pre-trained language models. 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 2286–2300, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. 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.

Michael Matena and Colin Raffel. 2021. Merging models with fisher-weighted averaging. *ArXiv preprint*, abs/2111.09832.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: the sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Dilan Gorur, Razvan Pascanu, and Hassan Ghasemzadeh. 2020. Linear mode connectivity in multitask and continual learning. *arXiv preprint arXiv:2010.04495*.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. 2020. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. 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, 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.

Clifton Poth, Jonas Pfeiffer, Andreas Rücklé, and Iryna Gurevych. 2021. What to pre-train on? Efficient intermediate task selection. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10585–10605, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5231–5247, Online. Association for Computational Linguistics.

Yujia Qin, Yankai Lin, Jing Yi, Jiajie Zhang, Xu Han, Zhengyan Zhang, Yusheng Su, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022a. Knowledge inheritance for pre-trained language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3921–3937, Seattle, United States. Association for Computational Linguistics.

Yujia Qin, Jiajie Zhang, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2022b. ELLE: Efficient lifelong pre-training for emerging data. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2789–2810, Dublin, Ireland. Association for Computational Linguistics.

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.

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.

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.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. 2022. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3949–3969, Seattle, United States. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926, Online. Association for Computational Linguistics.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big data*, 3(1):1–40.

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.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. *ArXiv preprint*, abs/2203.05482.

Mitchell Wortsman, Gabriel Ilharco, Mike Li, Jong Wook Kim, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. 2021. Robust fine-tuning of zero-shot models. *ArXiv preprint*, abs/2109.01903.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2022. NoisyTune: A little noise can help you finetune pretrained language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 680–685, Dublin, Ireland. Association for Computational Linguistics.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018. ReCoRD: Bridging the gap between human and machine commonsense reading comprehension.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

Appendices

A Details for Finding a Bezier Curve

We follow Garipov et al. (2018) to find a quadratic Bezier curve connecting two endpoints θ_{C_1} and θ_{C_2}. The Bezier curve is defined as follows:

$$\phi_\theta(\alpha) = (1-\alpha)^2 \cdot \theta_{C_1} + 2\alpha(1-\alpha)\theta + \alpha^2 \cdot \theta_{C_2}.$$

During curve finding, only $\theta \in \mathbb{R}^{|\theta_0|}$ is optimized to minimize the expectation of loss over a uniform distribution on the curve as follows:

$$\mathcal{L}_{\text{curve}}(\theta) = \frac{\int \mathcal{L}(\phi_\theta) d\phi_\theta}{\int d\phi_\theta} = \frac{\int_0^1 \mathcal{L}(\phi_\theta(\alpha)) ||\phi'_\theta(\alpha)|| d\alpha}{\int_0^1 ||\phi'_\theta(\alpha)|| d\alpha}$$
$$= \int_0^1 \mathcal{L}(\phi_\theta(\alpha)) q_\theta(\alpha) d\alpha = \mathbb{E}_{\alpha \sim q_\theta(\alpha)} \mathcal{L}(\phi_\theta(\alpha)),$$

where $q_\theta(\alpha) = \frac{||\phi'_\theta(\alpha)|| d\alpha}{\int_0^1 ||\phi'_\theta(\alpha)|| d\alpha}$. Since $q_\theta(\alpha)$ is dependent on θ, it is generally intractable to compute the original loss $\mathcal{L}_{\text{curve}}(\theta)$. To this end, Garipov et al. (2018) suggest optimizing a more computationally tractable loss as follows:

$$\mathcal{L}'_{\text{curve}}(\theta) = \mathbb{E}_{\alpha \in \text{U}(0,1)} \mathcal{L}(\phi_\theta(\alpha)),$$

where α is sampled from a uniform distribution $\text{U}(0,1)$ instead of $q_\theta(\alpha)$. In experiments, we initialize θ with $\frac{1}{2}\theta_{C_1} + \frac{1}{2}\theta_{C_2}$ (i.e., starting from a linear curve), which makes training more stable than using randomly initialized weights or the pre-trained weights.

B Additional Experiments

B.1 Effects of the Learning Rate and the Batch Size

We perform experiments to explore the effects of both learning rates and batch sizes. For the former, we evaluate when both endpoints are trained with a learning rate of $\{1 \times 10^{-4}, 5 \times 10^{-4}\}$ and $\{1 \times 10^{-4}, 5 \times 10^{-5}\}$, and the batch size is set to 16; for the latter, we chose a batch size of $\{16, 8\}$ and $\{16, 32\}$, and the learning rate is set to 1×10^{-4}. The experiments are conducted using both full-parameter fine-tuning and adapter tuning on MNLI and SST-2 with T5_{BASE}. For MNLI, we experiment when both endpoints are trained for {10k, 30k, 50k} steps; for SST-2, both endpoints are trained for {3k, 9k, 15k} steps. We illustrate the results of linear interpolation in Figure 9 and Figure 10. We could conclude from both figures that, the minima obtained by fine-tuning are always well-connected by the linear path; however, the connectivity of adapter is poor under certain cases. This is aligned with the finding in the main paper that the mode connectivity of fine-tuning is generally better than delta tuning. We also observe that with the training steps becoming larger, the connectivity of adapter tuning sometimes becomes poorer.

B.2 Additional Experiments for the Effects of Training Steps

In the main paper, when exploring the effects of training steps, we experiment with the setting where both endpoints are trained with the same number of steps. We show that mode connectivity could be poorer when both endpoints are trained for longer steps. To more rigorously investigate the effects of training steps, we experiment when both minima are obtained when using different training steps. Specifically, we adapt T5_{BASE} model on MNLI and ReCoRD using both fine-tuning and adapter tuning. We train two endpoints with different initialization, which is implemented by utilizing different random seeds, but keeping the configuration of the initialization (mean and standard deviation of the normal distribution) the same. After that, one endpoint (denoted as θ_{C_1}) is adapted for 50k steps, while the other endpoint is adapted for {10k, 20k, 30k, 40k} steps, and denoted as $\{\theta_{C_2}^{10\text{k}}, \theta_{C_2}^{20\text{k}}, \theta_{C_2}^{30\text{k}}, \theta_{C_2}^{40\text{k}}\}$, respectively. Then we evaluate the linear interpolations between $\{(\theta_{C_1}$ and $\theta_{C_2}^{10\text{k}})$, $(\theta_{C_1}$ and $\theta_{C_2}^{20\text{k}})$, $(\theta_{C_1}$ and $\theta_{C_2}^{30\text{k}})$, $(\theta_{C_1}$ and $\theta_{C_2}^{40\text{k}})\}$, respectively. The results are shown in Figure 11, from which we observe that, the mode connectivity of fine-tuning is generally good, while two minima of adapter are not well-connected. In addition, with the gap of the training steps between two endpoints becoming larger, the mode connectivity of adapter becomes poorer. These results suggest that the number of training steps can affect PLM's mode connectivity under certain cases.

Euclidean Distance Analysis. To better understand the reason why training steps could affect mode connectivity, we record the Euclidean distance variation of two endpoints during downstream adaptation. Both endpoints start from different initialization. We adapt the PLM on MNLI using both fine-tuning and adapter tuning for {10k, 20k, 30k, 40k, 50k} steps, and obtain a series of checkpoints: $\{(\theta_{C_1}^{10\text{k}}$ and $\theta_{C_2}^{10\text{k}})$, $(\theta_{C_1}^{20\text{k}}$ and $\theta_{C_2}^{20\text{k}})$, $(\theta_{C_1}^{30\text{k}}$ and $\theta_{C_2}^{30\text{k}})$, $(\theta_{C_1}^{40\text{k}}$ and $\theta_{C_2}^{40\text{k}})$, $(\theta_{C_1}^{50\text{k}}$ and $\theta_{C_2}^{50\text{k}})\}$. Then we calculate the Euclidean distance of two endpoints as: $||\theta_{C_1}^* - \theta_{C_2}^*||^2$. The change of Eu-

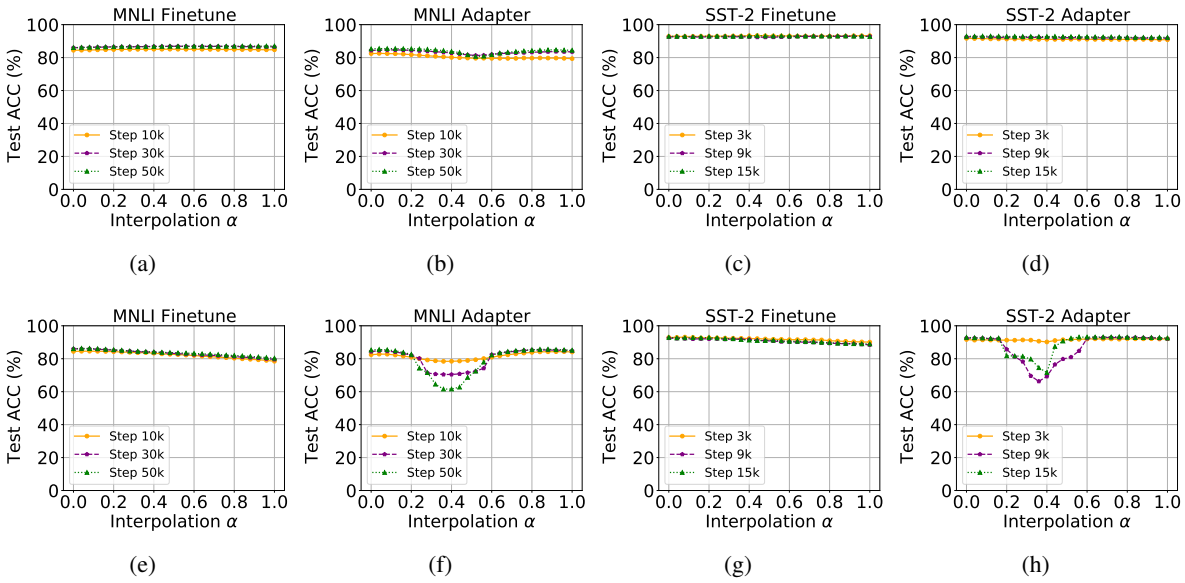


Figure 9: Experiments of the effects of the learning rate. We conduct linear interpolations for MNLI and SST-2, using both fine-tuning and adapter tuning. For (a-d), both minima are obtained with a learning rate of 1×10^{-4} and 5×10^{-5}, respectively; for (e-h), both minima are obtained with a learning rate of 1×10^{-4} and 5×10^{-4}, respectively.

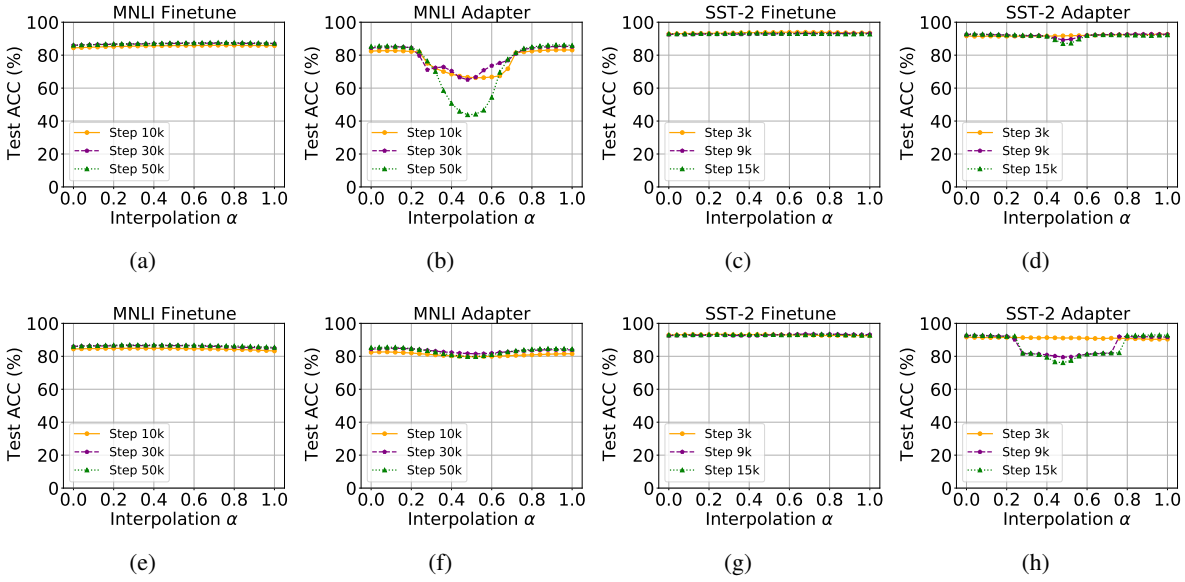


Figure 10: Experiments of the effects of the batch size. We conduct linear interpolations for MNLI and SST-2, using both fine-tuning and adapter tuning. For (a-d), both minima are obtained with a batch size of 16 and 32, respectively; for (e-h), both minima are obtained with a batch size of 16 and 8, respectively.

clidean distance is visualized in Figure 12, from which we observe that, with the training steps becoming larger, the distance between two endpoints is also prolonged. This may partially explain the poorer mode connectivity with the increasing of training steps. We have shown in the main paper that PLMs have multiple loss basins connected by a non-linear path, instead of a linear path. Within the same loss basins, most of the solutions have good linear mode connectivity. However, since the loss basin has a boundary, when the distance between two minima becomes large enough, they may finally cross the border of the loss basin. Under this scenario, the linear path connecting both endpoints would incur a high loss.

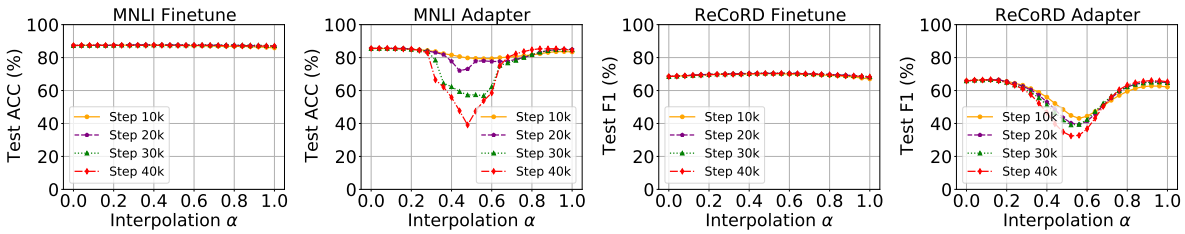


Figure 11: Experiments of the effects of the training steps. We conduct linear interpolations for MNLI and ReCoRD, using both fine-tuning and adapter tuning. One endpoint is trained for 50k steps, while the other endpoint is trained for {10k, 20k, 30k, 40k} steps, respectively.

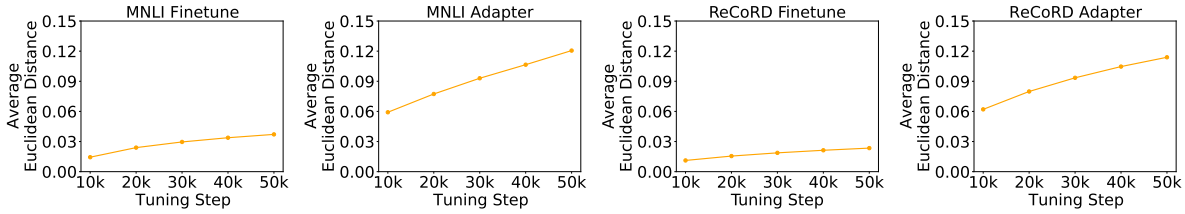


Figure 12: Euclidean distance (per neuron) of two minima at different training steps ({10k, 20k, 30k, 40k}) during downstream adaptation. Two minima are trained from different initialization.

B.3 More Experiments for Mode Connectivity along a Non-linear Path

In the main paper, to explore the connectivity of two minima along a non-linear path, we experiment on the setting of tuning adapters with different initialization. This setting has been shown to have poor linear mode connectivity but good non-linear mode connectivity. In fact, in our pilot experiments, we find that such a low-loss non-linear curve exists for minima reached under various different settings. In this section, we provide some of the experiments to demonstrate the above finding using T5_{BASE}.

Specifically, we experiment with three tasks: MNLI, ReCoRD, and SST-2. For adapter tuning, we choose the setting where two minima are trained with (1) different training data order and (2) data from the same distribution but without specific overlap of training instances. For (2), same as before, we randomly partition the original training dataset into two equal splits, and adapt two copies of PLM on each split. For fine-tuning, we choose the setting where two minima are trained with (1) different training steps (the setting is the same as appendix B.2), and (2) different initialization.

The performance of the interpolations are visualized in Figure 13 for adapter tuning and Figure 14 for fine-tuning. We observe that under all the settings, we do not observe a significant performance drop along the non-linear curve, showing that the connectivity is good. The above results demonstrate that PLMs may have multiple loss basins which are connected via a low-loss non-linear path. In this paper, following Neyshabur et al. (2020), we spend most of the efforts on convex hull and linear interpolation to avoid possibly trivial connectivity results.

B.4 Additional Experiments for the Effects of Data Overlap

In the main paper, when evaluating the effects of data overlap, we present the results on MNLI. In this section, we visualize the results when using ReCoRD and SST-2 in Figure 15. Other settings are kept the same as the main paper.

B.5 Additional Experiments for the Effects of Data Domain

In the main paper, when evaluating the effects of data distributions (data domain), we present the results when using fine-tuning. In this section, we visualize the results when using adapter tuning in Figure 16. Other settings are kept the same as the main paper.

B.6 Additional Experiments for the Change of Mode Connectivity during Pre-training

In the main paper, when evaluating the performance variation between two minima trained on two different tasks, we report the results of MNLI and SST-2.

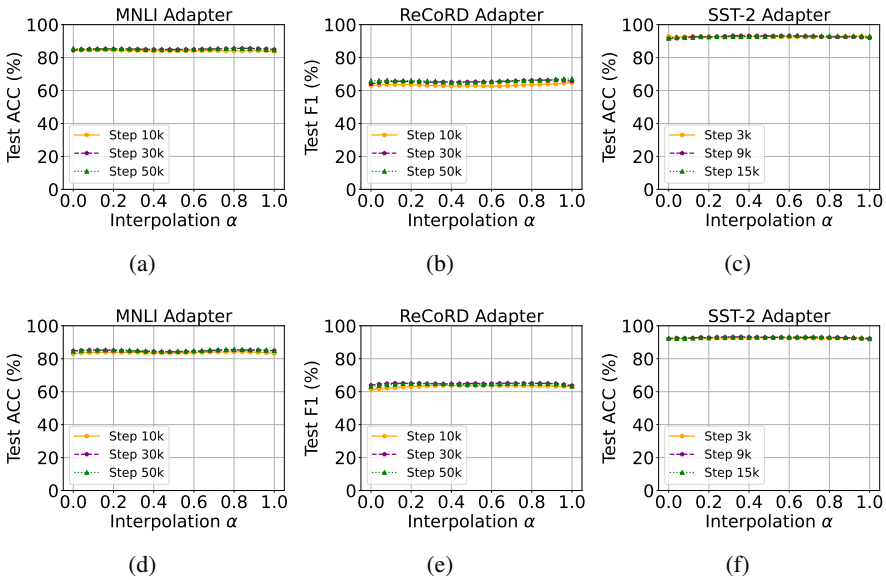


Figure 13: The performance of interpolations along a non-linear path connecting two minima trained with adapter tuning. For (a-c), two minima are trained with different training data order. For (d-f), two minima are trained with in-distribution data of the same task.

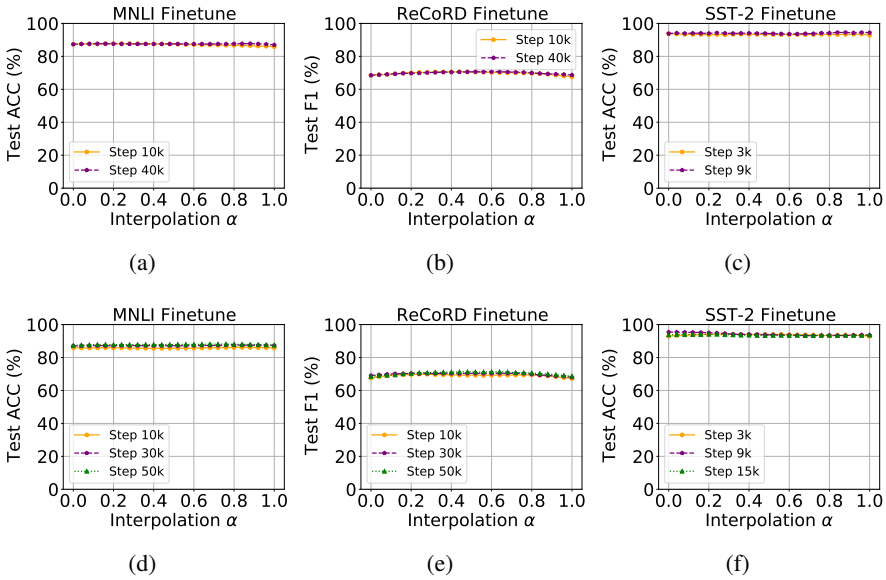


Figure 14: The performance of interpolations along a non-linear path connecting two minima trained with fine-tuning. For (a-c), two minima are trained with different training steps. For (d-f), two minima are trained with different initialization.

In this section, we present the results of MNLI and QQP in Figure 17. In fact, in our pilot studies, we find that the conclusions in diverse tasks are very consistent. Due to the concern about the energy cost, we only report the performance of two pairs of tasks.

B.7 Performance along the Connecting Path

We show that better performance could be achieved by interpolating two independently trained weights in the parameter space. Specifically, we choose the scenario where two copies of PLMs are trained with different training data order. As mentioned in Q1. (a) in the main paper, PLMs have excellent mode connectivity under this setting. We experiment with T5_{BASE} using fine-tuning and adapter tun-

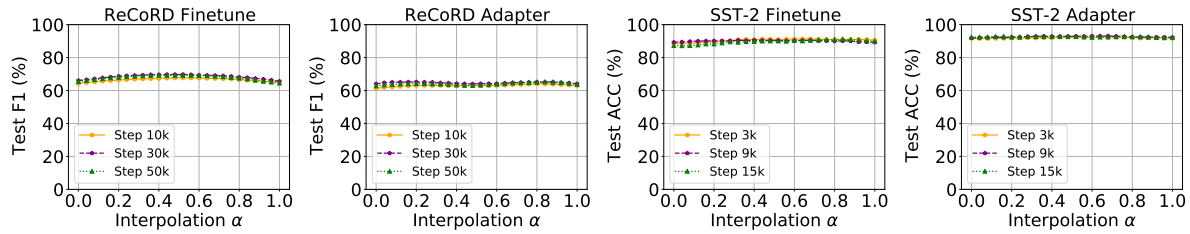


Figure 15: Linear mode connectivity analysis for two minima trained with in-distribution data. In (a-b), we experiment with ReCoRD. In (c-d), we experiment with SST-2.

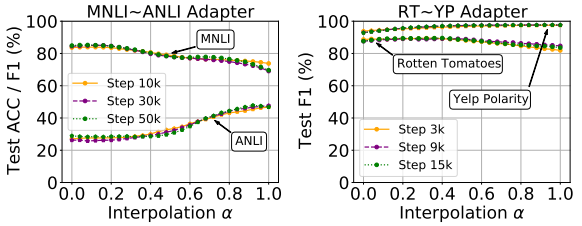


Figure 16: Linear mode connectivity for two minima trained on different data distributions of the same task using adapter tuning. Left: $\alpha = 0$ / $\alpha = 1$ denotes the minimum of MNLI / ANLI. Right: $\alpha = 0$ / $\alpha = 1$ denotes the minimum of Rotten Tomatoes / Yelp Polarity.

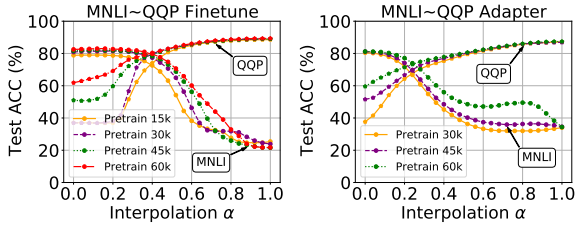


Figure 17: The change of linear mode connectivity at different pre-training steps. We illustrate the performance of linear interpolations of two minima trained on MNLI and QQP. $\alpha = 0$ / $\alpha = 1$ denotes the minimum of MNLI / QQP.

ing on MNLI, and conduct both linear interpolation and curved interpolation. We evaluate the performance of 24 evenly distributed points on the curve on a development set, select the best-performing one and evaluate its performance on the test set. We also compare the interpolation with the endpoints (we report the best performance of the two endpoints). All experiments are conducted 3 times and we report the average test results in Table 1. We observe that by traversing along the connecting curve between two minima, we could find a solution that performs better than both endpoints. In addition, traversing along a linear path finds an interpolation with higher performance than traversing along a curved path[2]. In general, this finding demonstrates that it is promising to combine the knowledge of multiple models through weight averaging.

B.8 Other Strategies for Weight Ensemble

As demonstrated in the main paper, the connectivity on a non-linear path may be better than a linear path under certain cases. This phenomenon demonstrates that despite the simplicity of linear interpolation, there may exist better ways to interpolate two independently trained minima. To demonstrate the existence of a better combination for two minima, we take an initial step and propose to optimize such a combination. Specifically, suppose all the tunable parameters of a PLM can be divided into M components, i.e., $\theta_{C_1} = \{\theta^i_{C_1}\}^M_{i=1}$, $\theta_{C_2} = \{\theta^i_{C_2}\}^M_{i=1}$. We optimize a tunable vector $\boldsymbol{\alpha} \in \mathbb{R}^M$ to combine θ_{C_1} and θ_{C_2} as follows:

$$\theta(\boldsymbol{\alpha}) = \{\sigma(\boldsymbol{\alpha}_i) \cdot \theta^i_{C_1} + (1 - \sigma(\boldsymbol{\alpha}_i)) \cdot \theta^i_{C_2}\}^M_{i=1},$$

where σ denotes a *sigmoid* function. During training, both θ_{C_1} and θ_{C_2} are kept frozen, and only $\boldsymbol{\alpha}$ is tuned. We design three intuitive strategies for parameter division: (1) layer-wise division, where the parameters within the same layer share the same combination ratio; (2) module-wise division, where we discriminate the combination ratio of the feed-forward network module and multi-head attention module in each layer. This means each module in each layer is assigned with an individual combination ratio; (3) matrix-wise division, where each weight matrix in each module is combined individually. Matrix-wise division is the most fine-grained

[2]Although we have shown that the non-linear mode connectivity is generally good for different minima, it does not mean that the best performance on a non-linear curve is always better than that on a linear curve.

Interpolation	Step	Linear	Curved	Endpoint
Adapter	10k	**84.58**	84.14	84.37
	30k	**85.27**	85.14	84.96
	50k	**85.70**	85.23	85.55
Fine-tuning	10k	**85.97**	85.77	85.85
	30k	**87.36**	87.27	87.29
	50k	87.88	**87.88**	87.52

Table 1: Test performance for interpolations of two minima trained on MNLI using different training data order. For both linear interpolation and curved interpolation, we choose the best checkpoint based on the development set performance. For two endpoints, we report the endpoint that performs better on the test set.

Step	Linear	Layer	Module	Matrix	Endpoint
10k	86.0	86.2	**86.3**	86.3	85.9
30k	87.4	87.4	87.4	**87.5**	87.3
50k	87.9	87.7	87.5	**88.0**	87.5

Table 2: Test performance for interpolations of two minima fine-tuned on MNLI using different training data order. For linear interpolation (**Linear**), we choose the best checkpoint based on the development set performance. For two endpoints (**Endpoint**), we report the endpoint that performs better on the test set. **Layer**, **Module**, and **Matrix** denote layer-wise, module-wise, and matrix-wise divisions we proposed.

one among the above three strategies. Since we use the T5_{BASE} model, which consists of 12 encoders and 12 decoders, the number of components M for block-wise, layer-wise, and matrix-wise divisions are 24, 60, and 120 for adapter; and 28, 64, and 282 for fine-tuning. During training, we perform grid search on a series of learning rates $\{0.1, 0.05, 0.01\}$ and set the batch size to 8, and max training steps to 100k.

Two endpoints are obtained by fine-tuning T5_{BASE} on MNLI using different training data order. The results are shown in Table 2, from which we find that, among the proposed three combination strategies, matrix-wise division achieves the best performance. The performance is also better than using linear interpolation. This phenomenon demonstrates that there exist better ways for combining two minima's knowledge than linear interpolation. We hope our findings on mode connectivity in this paper could inspire future works to design better weight ensemble methods.

C Training Details

For the T5_{BASE} model, we use the checkpoint provided by Lester et al. (2021), who conducted additional 100k steps of language modeling adaption on the official checkpoints released by Raffel et al. (2020). Such adaptation is demonstrated to help stabilize downstream adaptation and improve the performance, especially for delta tuning methods (Lester et al., 2021). We use AdamW (Loshchilov and Hutter, 2019) as the optimizer for all the experimented PLMs. All the implementation codes, trained checkpoints and used datasets would be released after publication.

Task	LR	BS	SI
MNLI	5×10^{-5}	32	10k
ReCoRD	1×10^{-4}	32	10k
ANLI	1×10^{-4}	32	10k
SST-2	5×10^{-5}	8	3k
Rotten Tomatoes	5×10^{-5}	32	3k
Yelp Polarity	5×10^{-4}	8	3k

Table 3: Hyperparameters (LR: learning rate, BS: batch size, SI: saving interval) for different tasks during the fine-tuning of T5_{BASE} model.

We download all the experimented datasets from *Huggingface Datasets* (Lhoest et al., 2021). Since some datasets do not contain a test set, we first merge all the data points, and then split them into the new training split, development split, and test split with an approximate ratio of $8:1:1$. The above procedure is conducted on all the experimented datasets.

For different tasks fine-tuned on T5_{BASE}, we first conduct grid search to find an optimal hyperparameter combination. Specifically, the chosen hyperparameter of different tasks for fine-tuning is shown in Table 3; for adapter tuning, in our prior experiments, we find that a learning rate of 5×10^{-4} and a batch size of 16 performs good on all tasks, thus we set them as the default configuration. For both tuning methods, we save 5 checkpoints during training, with different saving intervals for different tasks as shown in Table 3.

C.1 Additional Details for the Effects of Initialization

For adapter tuning, all the modules newly introduced are initialized using a Gaussian distribution. As for fine-tuning, we add Gaussian noise to all the tunable parameters. The mean and standard deviation of the Gaussian distribution are set to 0 and 0.0002, respectively. We use different random seeds to generate different initialization.

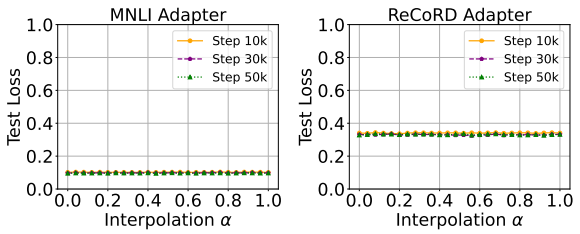


Figure 18: The loss of curved interpolations between two minima trained with adapter tuning from different initialization. The corresponding performance visualization is Figure 3.

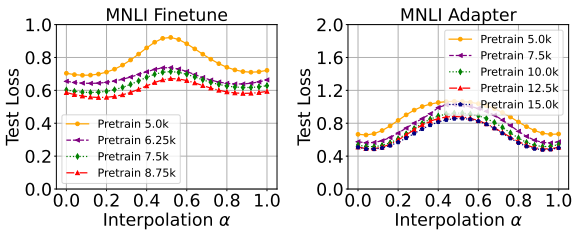


Figure 19: The loss of the change of mode-connectivity at different pre-training steps. We report the results of linear interpolations of two minima trained on MNLI using different initialization. The corresponding performance visualization is Figure 6.

C.2 Additional Details for Curve Finding

When optimizing the Bezier curve, we set the learning rate to 1×10^{-4}, batch size to 8, max training steps to 5k for fine-tuning; and set the learning rate to 1×10^{-4}, batch size to 16, max training steps to 10k for adapter tuning. During curve finding, we evaluate the development performance of the current curve for every 100 steps, using a series interpolations with $\alpha \in \{0.25, 0.5, 0.75\}$.

C.3 Additional Details for Calculating Confidence and Variability

As mentioned in the main paper, we use the training dynamics to characterize each training sample. For MNLI / ANLI, we adapt the model for 8 epochs / 20 epochs to calculate both confidence and variability. We tune more epochs for ANLI because the size of its training dataset is far smaller than that of MNLI.

C.4 Additional Details for Pre-training RoBERTa$_{\text{BASE}}$

We closely follow the pre-training setting of Liu et al. (2019b), except that for pre-training data, we use the concatenation of Wikipedia and BookCorpus (Zhu et al., 2015) same as BERT (Devlin et al., 2019), and we pre-train our model with a batch size of 2048. The pre-training implementations for RoBERTa$_{\text{BASE}}$ are based on those of Qin et al. (2022a,b). Adam (Loshchilov and Hutter, 2019) is chosen as the optimizer. The hyperparameters for the optimizer is set to 1×10^{-6}, 0.9, 0.98 for ϵ, β_1, β_2, respectively. We set the dropout rate to 0.1, weight decay to 0.01 and use linear learning rate decay. The model architecture is the same as the official RoBERTa$_{\text{BASE}}$ model (Liu et al., 2019b). The pre-training is conducted using 8 NVIDIA V100 GPUs.

D The Visualization of Loss for Interpolations

As mentioned before, we record both loss and performance for each interpolation. Since we find that the trends of loss and performance are generally highly correlated, due to the length limit, we only report the performance in the main paper. In this section, we visualize the loss for most of the experiments conducted in this paper, see Figure 18, Figure 19, Figure 20, Figure 21, Figure 22, Figure 23, Figure 24, and Figure 25.

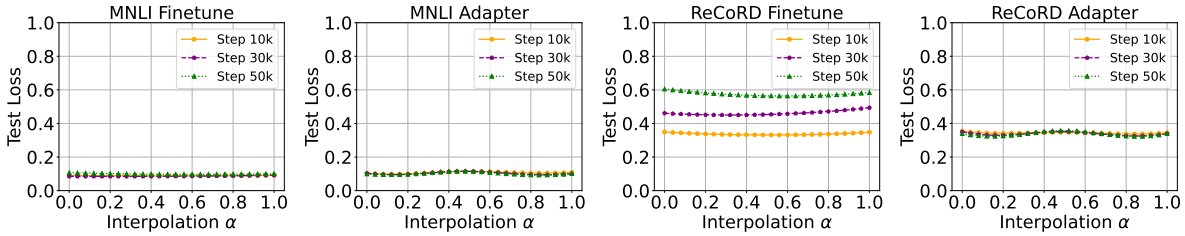


Figure 20: The loss of linear interpolations between two minima trained with different training data order. The corresponding performance visualization is Figure 1.

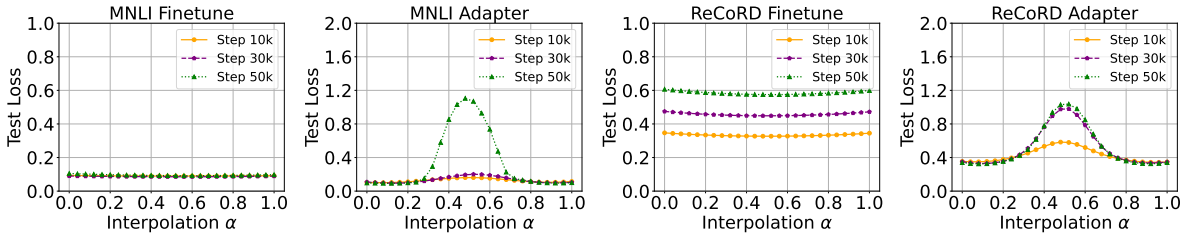


Figure 21: The loss of linear interpolations between two minima trained with different initialization. The corresponding performance visualization is Figure 2.

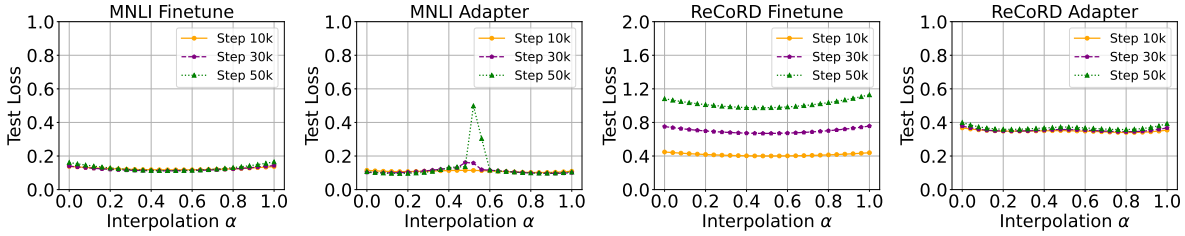
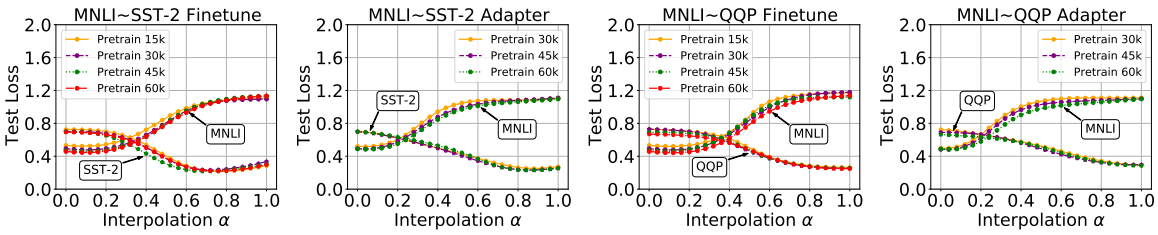


Figure 22: Linear mode connectivity analysis (loss) for two minima trained with in-distribution data. The corresponding performance visualization is Figure 4 and Figure 15.



(a) (b) (c) (d)

Figure 23: The results (loss) of the change of mode-connectivity at different pre-training steps. (a-b) record the results of linear interpolations of two minima trained on MNLI and SST-2, (c-d) record the results of linear interpolations of two minima trained on MNLI and QQP. The corresponding performance visualization is Figure 7 and Figure 17.

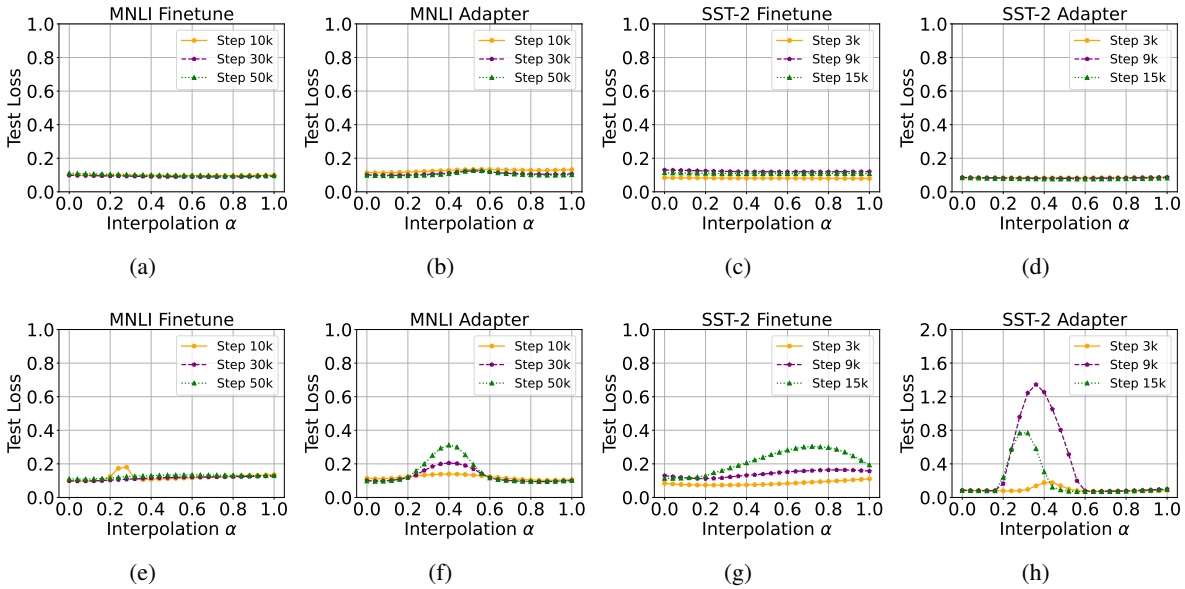


Figure 24: Experiments of the effects of the learning rate. We conduct linear interpolations for MNLI and SST-2, using both fine-tuning and adapter tuning, and visualize their loss. For (a-d), both minima are obtained with a learning rate of 1×10^{-4} and 5×10^{-5}, respectively; for (e-h), both minima are obtained with a learning rate of 1×10^{-4} and 5×10^{-4}, respectively. The corresponding performance visualization is Figure 9.

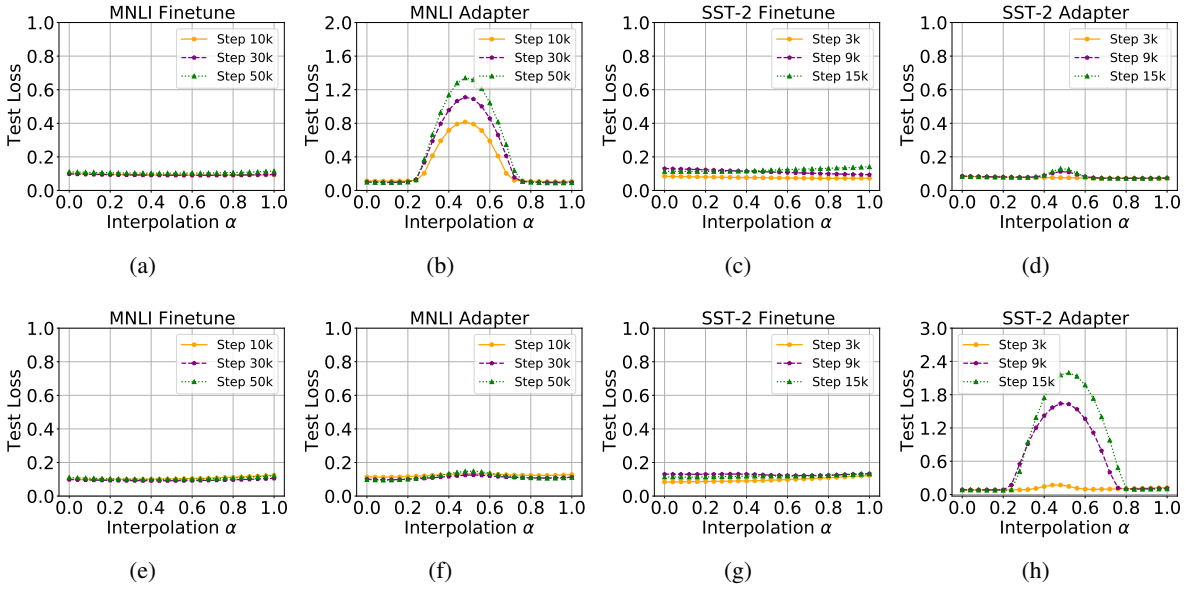


Figure 25: Experiments of the effects of the batch size. We conduct linear interpolations for MNLI and SST-2, using both fine-tuning and adapter tuning, and visualize their loss. For (a-d), both minima are obtained with a batch size of 16 and 32, respectively; for (e-h), both minima are obtained with a batch size of 16 and 8, respectively. The corresponding performance visualization is Figure 10.