Task-Specific Skill Localization in Fine-tuned Language Models

Abhishek Panigrahi [*1] **Nikunj Saunshi** [*1] **Haoyu Zhao** [1] **Sanjeev Arora** [1]

Abstract

Pre-trained language models can be fine-tuned to solve diverse NLP tasks, including in few-shot settings. Thus fine-tuning allows the model to quickly pick up task-specific "skills," but there has been limited study of *where* these newly-learnt skills reside inside the massive model. This paper introduces the term *skill localization* for this problem and proposes a solution. Given the downstream task and a model fine-tuned on that task, a simple optimization is used to identify a very small subset of parameters ($\sim 0.01\%$ of model parameters) responsible for ($> 95\%$) of the model's performance, in the sense that *grafting* the fine-tuned values for just this tiny subset onto the pre-trained model gives a performance almost as well as the fine-tuned model. While reminiscent of recent works on parameter-efficient fine-tuning, the novel aspects here are that: (i) No further re-training is needed on the subset (unlike, say, with lottery tickets). (ii) Notable improvements are seen over vanilla fine-tuning with respect to calibration of predictions in-distribution (40-90% error reduction) as well as the quality of predictions out-of-distribution (OOD). In models trained on multiple tasks, a stronger notion of skill localization is observed, where the sparse regions corresponding to different tasks are almost disjoint, and their overlap (when it happens) is a proxy for task similarity. Experiments suggest that localization via grafting can assist certain forms of continual learning. Our code is available at Skill-Localization-by-grafting[1].

[*]Equal contribution [1]Department of Computer Science, Princeton University. Correspondence to: Abhishek Panigrahi <ap34@princeton.edu>, Nikunj Saunshi <nsaunshi@google.com>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

[1]https://github.com/abhishekpanigrahi1996/Skill-Localization-by-grafting

1. Introduction

Pre-trained language models (Liu et al., 2019b; Devlin et al., 2019) have shown huge success after fine-tuning (FT) on many downstream tasks. With as few as 32 training examples, fine-tuning these giant models beats head tuning (Gao et al., 2021b). Thus fine-tuning is quick to acquire new "skills" like sentiment identification, but locating where these skills reside in the net is an open question. *A priori*, one expects them to be diffused and not be "localized" in a meaningful way. Better understanding of the skill's location could improve fine-tuned models – say, with respect to accuracy, calibration, or out-of-domain generalization— or help mitigate catastrophic forgetting in multi-task settings. Existing methods for parameter-efficient fine-tuning (e.g., using *lottery tickets* (Frankle & Carbin, 2018; Chen et al., 2020)) suggest the existence of more compact descriptions of the fine-tuned model, but they involve re-training the net and do not give insight into where the skills existed after vanilla fine-tuning. (Section 6 discusses past works.).

This paper gives a simple way to pinpoint tiny regions in the pre-trained model where the skills acquired via fine-tuning can be localized. In particular, although fine-tuning could have updated hundreds of millions of parameters, we can identify (Section 2) a tiny subset of (a few thousand) parameters whose values are sufficient to solve the task, in the following sense: "grafting" the values of this tiny subset of parameters onto the pre-trained model without changing any of the other parameters, almost recovers the performance of the original fine-tuned model. We call this tiny subset of parameters a "grafting region." Note that finding sparse grafting regions allows for compact storage of the fine-tuned model, which could be important in settings where the model is being fine-tuned for a large number of tasks or users. Crucially we find that grafted models have other desirable properties like calibration and OOD generalization. Our main contributions are as follows.

- Section 2 formalizes skill localization via *grafting*, and gives a simple optimization procedure to find them. Section 3 shows that in a RoBERTa (GPT-2 resp.) model fine-tuned on GLUE tasks, regions with just 0.01% (0.05% resp.) of parameters can recover 95% accuracy of the fine-tuned models, without further re-training.
- Section 4 shows that our grafted models have much better

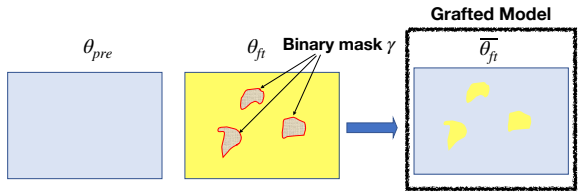


Figure 1. Grafting learns a binary mask $\boldsymbol{\gamma}$ using the fine-tuned ($\boldsymbol{\theta}_{\text{ft}}$) and pre-trained ($\boldsymbol{\theta}_{\text{pre}}$) models, and creates a grafted model $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$. For parameters in the region corresponding to $\boldsymbol{\gamma}$, $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$ gets its values from $\boldsymbol{\theta}_{\text{ft}}$, while all other parameters default to $\boldsymbol{\theta}_{\text{pre}}$.

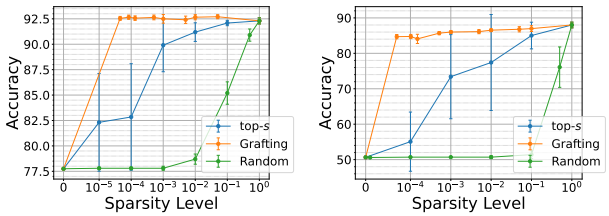
calibrated outputs than vanilla fine-tuning — a significant result because calibrating models can be difficult on small datasets. Grafted models, without re-training, also often have much better out-of-distribution (OOD) performance. However, re-training a sparse region (including prior parameter-efficient fine-tuning methods like BitFit) does not afford the same OOD benefits. These findings suggest that retaining sparse grafting regions provide a purer, more transferable way to capture the skill, while avoiding over-fitting to idiosyncrasies of the specific dataset. The section also discusses the generalization mystery of fine-tuning and how the graft regions begin to explain it.

- Section 5 explores consequences of our skill localization in multi-task and continual learning settings. We show that when FT learns multiple tasks together, skills from different tasks localize in somewhat disjoint regions, where the degree of overlap between regions for two tasks seems to correlate with their intuitive similarity. We also observed some degree of compositionality: grafting the pre-trained net using regions of a subset of tasks works well for only those (and related) tasks but not others.

2. Skill Localization through Model Grafting

Humans have concise ways of describing their skills in a modular fashion using natural language, typically as a combination of more basic skills. Such descriptions are challenging for skills in deep nets. In context of fine-tuning, recent papers have approached by equating "skill" with the ability to perform a specific fine-tuning task. They correlate this ability with activations of certain subsets of neurons; for instance, Wang et al. (2022) find "skill neurons" in prompt-tuned nets (not FT nets). While interesting, such notions suffer from the limitation that the activations depend on both the input to the net, as well as on a large set of parameters. Ideally, we would pinpoint the skill for the entire task in terms of specific net parameters, and in a compact way.

A naive attempt at a parameter-centered formalization would be to identify parameters that change a lot during fine-tuning. However, this turns out to be neither concise nor closely connected to the task in question; see Figure 2.



(a) SST-2, 4096-shot (b) QNLI, 4096-shot

Figure 2. Accuracies of the grafting regions learned using our procedure in Section 2.3, regions corresponding to the top-s parameters based on the magnitude of movement during FT, and random regions. The learned region performs much better at low sparsity levels.

2.1. Model Grafting

To formalize skill localization, we introduce *model grafting*. Given a pre-trained model with parameters $\boldsymbol{\theta}_{\text{pre}}$, fine-tuned model $\boldsymbol{\theta}_{\text{ft}}$, we can think of a binary mask $\boldsymbol{\gamma} \in \{0,1\}^{|\boldsymbol{\theta}_{\text{ft}}|}$ as identifying a subset of parameters, also called a "region." A *grafting* of $\boldsymbol{\theta}_{\text{ft}}$ in the region $\boldsymbol{\gamma}$ onto $\boldsymbol{\theta}_{\text{pre}}$ is defined as

$$\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma}) = \boldsymbol{\gamma} \odot \boldsymbol{\theta}_{\text{ft}} + (1 - \boldsymbol{\gamma}) \odot \boldsymbol{\theta}_{\text{pre}}. \tag{1}$$

In other words, for parameters in the region corresponding to $\boldsymbol{\gamma}$, $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$ gets its values from $\boldsymbol{\theta}_{\text{ft}}$, while all other parameters default to $\boldsymbol{\theta}_{\text{pre}}$, yielding a "grafted model." This is reminiscent of model stitching (Bansal et al., 2021), where layers of one model are grafted onto the remaining layers of another model. But we allow any subset of parameters to be grafted, thus potentially affecting very tiny fraction of parameters. We desire two competing properties:

Good Localization: the region $\boldsymbol{\gamma}$ is sparse, i.e. $\|\boldsymbol{\gamma}\|_0$ is tiny

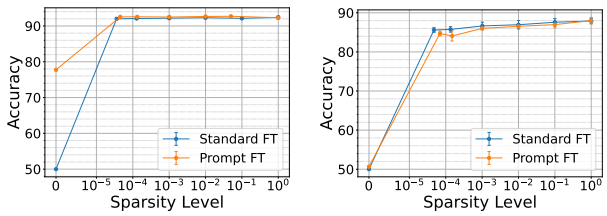
Skill retention: $\mathcal{L}_{\mathcal{T}}(\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})) \approx \mathcal{L}_{\mathcal{T}}(\boldsymbol{\theta}_{\text{ft}})$ are both small, i.e., grafted model recovers the fine-tuning performance

where $\mathcal{L}_{\mathcal{T}}$ is some metric for performance on $\mathcal{T}$ (e.g. classification error or logistic loss). We note that we can rewrite Equation (1) as $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma}) = \boldsymbol{\theta}_{\text{pre}} + \boldsymbol{\gamma} \odot (\boldsymbol{\theta}_{\text{ft}} - \boldsymbol{\theta}_{\text{pre}})$. Thus while $\boldsymbol{\gamma}$ denotes the location of the skill, $\boldsymbol{\gamma} \odot (\boldsymbol{\theta}_{\text{ft}} - \boldsymbol{\theta}_{\text{pre}})$ gives a succinct representation of the core skills acquired. This characterization suggests a natural way to learn a grafting region $\boldsymbol{\gamma}$ as well; see Section 2.3.

2.2. Differences from Sparsity-based Fine-Tuning

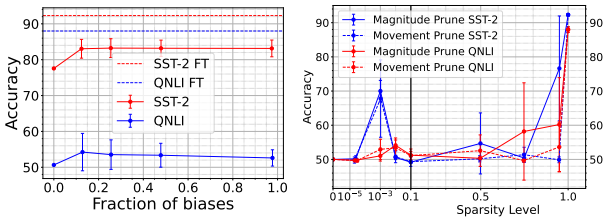
Grafting with sparse regions (aka *sparse grafting*), while reminiscent of efficient FT methods has key differences.

Lottery tickets: The lottery ticket hypothesis (LTH) (Frankle & Carbin, 2018) aims to prune to model by finding a sparse sub-network that when re-trained from scratch can recover the performance of vanilla training. Grafting is fundamentally different in two ways: (a) parameters outside



(a) SST-2, 4096-shot (b) QNLI, 4096-shot

Figure 3. Testing existence of sparse grafting regions for prompt-based FT and standard FT fine-tuning (uses a linear head on top of [CLS] token). Skill localization is equally good for FT approaches.



(a) Bias-only grafting (b) Lottery tickets pruning

Figure 4. (a) Grafting regions that only contain biases of the model don't give good skill localization. (b) Localizing with lottery ticket pruning (setting remaining parameters to 0) does not perform well at any sparsity level without re-training.

the grafting region are set to pre-trained values rather than 0, and (b) no re-training is needed. Our underlying motivation is to find skills in the standard fine-tuned model, whereas re-training can change the training mechanism itself.

Parameter-efficient fine-tuning: Methods like BitFit (Ben Zaken et al., 2022) find that updating only a small subset of parameters during fine-tuning (e.g. just biases) is very effective. However, the mechanism of parameter-efficient FT is very different from vanilla FT (e.g. biases are not important for vanilla FT). Furthermore, the sparsity of biases (0.1%) is $10\times$ than what task-dependent grafting can achieve. Besides, BitFit fails to provide the calibration benefits of grafting; see Table 1.

2.3. Optimization Procedure to Learn Grafted Models

We now describe a simple optimization procedure to learn a mask $\boldsymbol{\gamma}$ such that the grafted model $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$ from Equation (1) retains the skills to do well on the task $\mathcal{T}$:

$$\underset{\boldsymbol{\gamma}\in\{0,1\}^{|\boldsymbol{\theta}_{\text{ft}}|}:\|\boldsymbol{\gamma}\|_0\leq s}{\operatorname{argmin}} \quad \mathcal{L}_{\mathcal{T}}(\boldsymbol{\gamma}\odot\boldsymbol{\theta}_{\text{ft}}+(1-\boldsymbol{\gamma})\odot\boldsymbol{\theta}_{\text{pre}}) \tag{2}$$

Due to optimization considerations, we re-parametrize $\boldsymbol{\gamma}$ as a sigmoid of a real-valued vector $\boldsymbol{S}$, i.e. $\boldsymbol{\gamma}=\sigma(\boldsymbol{S})$. Furthermore to control the sparsity level s, we build the mask $\boldsymbol{\gamma}$ on top of an initial candidate mask $\boldsymbol{\gamma}_{\text{base}}\in\{0,1\}^{|\boldsymbol{\theta}_{\text{ft}}|}$. So the general optimization problem reduces to solving

$$\underset{\boldsymbol{S}\in\mathbb{R}^{|\boldsymbol{\theta}_{\text{ft}}|}}{\operatorname{argmin}} \quad \mathcal{L}_{T}(\boldsymbol{\gamma}\odot\boldsymbol{\theta}_{\text{ft}}+(1-\boldsymbol{\gamma})\odot\boldsymbol{\theta}_{\text{pre}}) \tag{3}$$

$$\boldsymbol{\gamma}\coloneqq\boldsymbol{\gamma}_{\text{base}}\odot(1-\sigma(\boldsymbol{S}))+(1-\boldsymbol{\gamma}_{\text{base}})\odot\sigma(\boldsymbol{S}) \tag{4}$$

Our optimization procedure aims to make minimal changes (addition or deletion) to $\boldsymbol{\gamma}_{\text{base}}$ while getting low task loss. We achieve minimal changes by initializing $\boldsymbol{S}$ such that $\sigma(\boldsymbol{S})\approx\mathbf{0}$ and by taking only a few gradient steps to train $\boldsymbol{S}$. (One could also use ℓ_1 regularization on $\sigma(\boldsymbol{S})$, but using a few gradient steps seems to suffice.) A natural and effective choice for $\boldsymbol{\gamma}_{\text{base}}$ turns out to be the top few parameters based on their movement $|\boldsymbol{\theta}_{\text{ft}}-\boldsymbol{\theta}_{\text{pre}}|$. While $\boldsymbol{\gamma}_{\text{base}}$ by itself is not great (see Figure 2), it tends to agree with the final localization in many coordinates.

3. Evaluating Grafting for Skill Localization

Experimental Setup. We fine-tuned the pre-trained RoBERTa-base (Liu et al., 2019b) model on 13 different tasks, with the majority from GLUE (Wang et al., 2018), including sentiment analysis, topic classification, natural language inference, and paraphrase detection datasets. All experiments, unless specified otherwise, use prompt-based fine-tuning with the human-generated prompts from Gao et al. (2021a), and SGD optimizer, which achieves similar performance as AdamW (Loshchilov & Hutter, 2019) after fixing the embedding layer (which was also observed in Kumar et al. (2022) but in the vision setting). For 64-shot and 4096-shot experiments, we report performance across 5 randomly sampled datasets. Unless mentioned otherwise, we always report our performance on 4096-shot setting. Other training and hyperparameter details can be found in Appendix A.2.

Model grafting experiments optimize Equation (3) using SGD with batch size 1024 (full-batch GD for 64-shot) for 100 steps with learning rate 10^7. For patches of varying sizes, we use $\boldsymbol{\gamma}_{\text{base}}$ as the top-s fraction of parameters based their movement $|\boldsymbol{\theta}_{\text{ft}}-\boldsymbol{\theta}_{\text{pre}}|$ for s in $[0,1]$.

3.1. Sparse Grafting Retain Skills

The first experiment compares the performance of model grafting with sparse regions versus full model (vanilla) fine-tuning. For each downstream task and prompt fine-tuned model, we learn a grafting region $\boldsymbol{\gamma}$ of sparsity at most 0.01% by building on top of $\boldsymbol{\gamma}_{\text{base}}\coloneqq\text{top-}(10^{-5})$, where top-$s$ selects the top s fraction parameters based on parameter movement. We report the accuracies of these grafted models in Table 2. The main observation is that sparse grafting can recover at least 95% of FT performance for all datasets. Additionally, the grafted models have a high agreement (on test set labels) with the original FT models: 93%

Table 1. Comparing the accuracy (†F1) in % and the calibration error (ECE) for fine-tuned model (FT), model grafting, graft re-training for grafting region with sparsity 0.01%, and BitFit. Results are in the 4096-shot setting. Re-training the grafting regions from scratch is not only good, but performs slightly better than the grafted model, implying that the grafting regions form a sub-network of their own. It also retains the good calibration of non re-trained grafting. BitFit updates all biases (sparsity 0.1%) and achieves slightly better accuracy than grafting, but it has significantly worse calibration error compared to the grafted model.

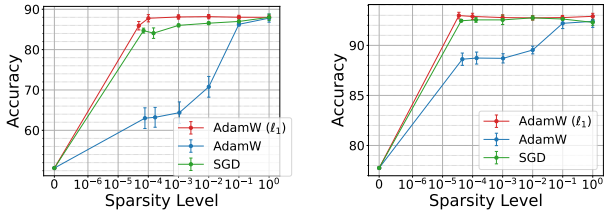
	FT		Grafting		Graft re-training		BitFit	
Dataset	Acc./F1	ECE	Acc./F1	ECE	Acc./F1	ECE	Acc./F1	ECE
SST-2	92.3 (0.3)	7.4 (0.3)	92.4 (0.1)	3.1 (0.4)	92.2 (0.7)	3.9 (0.7)	92.4 (0.6)	6.7 (0.8)
AGNews	92.7 (0.4)	6.8 (0.3)	91.1 (0.9)	0.9 (0.2)	91.2 (0.1)	1.0 (0.2)	93.0 (0.2)	4.4 (0.2)
QNLI	88.0 (0.8)	10.2 (0.0)	84.7 (0.6)	1.0 (0.3)	87.4 (0.5)	2.0 (1.1)	87.8 (0.6)	8.1 (2.3)
QQP†	79.6 (0.1)	10.1 (4.2)	76.3 (0.4)	3.5 (0.7)	78.1 (0.7)	4.3 (1.3)	79.4 (0.3)	9.7 (1.2)

(resp. 86%) for single-sentence (resp. two-sentence) experiments, suggesting good skill "localization." Please see Appendix B.1 for a closer analysis on the distribution of the graft parameters. We have similar observations for models trained with larger training data. Please see Appendix B.2.

Performance of γ_{base}: We compare the performance of the learned regions γ using optimization versus that of γ_{base} using top-s most-changed parameters, at different levels of sparsity in Figure 2. We find that the optimization method is much more effective, especially at lower sparsity levels. Exploring other ways to learn the regions, perhaps directly from the pre-trained model, is an interesting open question.

Gains from re-training the grafting regions. To check whether the learned region forms a meaningful sub-network within the model, we re-train starting from pre-trained initialization, but only make updates to parameters in the learned region γ, akin to parameter-efficient FT. Table 1 shows that re-training the sparse regions from scratch performs well, almost always better than the grafted models. This suggests that the sparse sub-network represented by γ is also trainable while being significantly sparser (0.01%) than the set of biases used in BitFit (0.1%).

Differences with BitFit and lottery tickets: Since BitFit succeeds by only training biases, we check whether biases can form good grafting regions. Figure 4a answers this in the negative, which implies a stark difference in mechanism between standard fine-tuning and BitFit. Furthermore, we check whether lottery tickets style pruning can work, without re-training, i.e. can we learn a sparse region such that setting other parameters to 0 can yield a good model. We find in Figure 4b that even regions as dense as 90% of parameters fail to capture any skill, without re-training. The high density is consistent with prior works on lottery tickets for language model fine-tuning (Chen et al., 2020), where the sparsity is usually higher than 10%, much denser compared to our grafting regions of 0.01%.



(a) QNLI, 4096-shot (b) SST-2, 4096-shot

Figure 5. Grafting accuracy for FT with SGD and AdamW. For both SST-2 and QNLI, the AdamW trained model is much worse at skill localization through grafting. However, a small ℓ_1 regularization on the parameter movement during FT recovers localization.

3.2. Other Fine-Tuning Paradigms

Experiments in the previous sections use prompt-based fine-tuning with SGD optimizer. In this section, we check whether sparse grafting regions (i.e. skill localization) exist in models fine-tuned differently.

Standard FT. Instead of prompt-based FT, we consider fine-tuning with a linear head on top of the [CLS] token representation, which was the *standard FT* approach before prompt-tuning (Liu et al., 2019b). Figure 3 confirms that similar sparse localization is also possible for Standard FT.

AdamW optimizer. In Figure 5 we test skill localization with AdamW (Loshchilov & Hutter, 2019) optimizer on prompt-based FT. Unlike SGD, we find that fine-tuning with AdamW *does not* contain sparse grafted models with good performance. However, adding an explicit ℓ_1 regularization (with strength 0.001) on the parameter movement $\|\boldsymbol{\theta}_{\text{ft}} - \boldsymbol{\theta}_{\text{pre}}\|_1$ can recover sparse grafts. This suggests that ℓ_1 regularization could be a way to encourage skill localization. An extensive exploration of this is left for future work.

4. Calibration, OOD and Generalization

Usually "skill" denotes flexibility and competence at a task, and machine learning has standard notions for testing this.

Table 2. For each downstream task, we learn a grafting region γ using our optimization procedure in Equation (3). The grafting regions for all tasks have sparsity at most 0.01% (< 8500 parameters). We report test accuracy (†F1) and the calibration error using ECE of the fine-tuned model and the grafted model for each task. The main findings are (1) The grafted model can retrieve $> 95\%$ of the FT accuracy, while being better calibrated than the original model itself. For single-sentence tasks (4096-shot) the grafted model shows only a 0.7% drop in accuracy but an improvement of 5% in the calibration error. Similarly for two-sentence tasks, the grafted model shows a 3.4% drop in accuracy with an improvement of 9.6% in the calibration error.

		64-shot				4096-shot				
		FT		Graft		FT		Graft		
	Dataset	Acc.	ECE	Acc.	ECE	Acc.	ECE	Acc.	ECE	Agreement
Single sent. tasks	SST-2	90.5 (0.4)	9.7 (0.3)	89.7 (0.2)	7.8 (0.6)	92.3 (0.3)	7.4 (0.3)	92.4 (0.1)	3.1 (0.4)	95.3 (0.6)
	CR	90.2 (0.6)	8.2 (2.3)	89.5 (1.1)	5.7 (1.9)	91.7 (0.2)	8.0 (0.3)	91.7 (0.5)	5.0 (0.3)	96.6 (0.5)
	MR	85.0 (1.2)	22.9 (2.1)	85.2 (1.7)	10.8 (2.3)	89.7 (0.3)	9.0 (0.6)	89.1 (1.1)	1.5 (0.2)	93.6 (0.8)
	MPQA	85.4 (0.9)	14.2 (0.9)	84.1 (1.2)	11.4 (1.8)	88.9 (0.6)	10.5 (0.6)	88.1 (0.4)	3.3 (0.2)	93.3 (0.2)
	TREC	93.1 (1.7)	6.1 (1.1)	86.8 (0.7)	4.8 (1.2)	-	-	-	-	-
	AGNews	88.2 (0.3)	10.1 (0.5)	86.8 (0.3)	7.1 (0.5)	92.7 (0.4)	6.8 (0.3)	91.1 (0.2)	0.9 (0.2)	95.1 (0.5)
	Subj	91.2 (1.2)	5.9 (1.2)	91.7 (1.2)	2.6 (1.2)	96.7 (0.1)	3.0 (0.1)	95.5 (0.1)	1.2 (0.1)	97.3 (0.2)
	Avg.	89.1	11.0	87.7	7.2	92	7.5	91.3	2.5	95.2
Two sent. tasks	QNLI	77.8 (0.9)	21.1 (0.9)	76.7 (1.5)	12.3 (0.8)	88.0 (0.8)	10.2 (0.0)	84.7 (0.6)	1.0 (0.3)	88.9 (0.3)
	SNLI	76.5 (1.7)	20.8 (1.6)	72.1 (1.9)	14.4 (3.1)	86.4 (0.3)	10.6 (1.7)	82.7 (0.5)	1.1 (0.4)	87.5 (1.5)
	MNLI	67.5 (2.1)	29.5 (2.0)	64.6 (2.5)	20.4 (1.4)	81.8 (0.1)	14.8 (0.7)	78.0 (0.4)	1.5 (0.0)	86.4 (0.4)
	RTE	66.9 (3.5)	31.0 (3.5)	66.2 (5.0)	21.5 (3.9)	80.0 (2.2)	20.2 (1.7)	77.7 (0.5)	8.5 (1.7)	88.6 (1.7)
	MRPC†	82.5 (1.9)	22.9 (2.1)	76.9 (2.4)	19.1 (3.5)	90.0 (0.7)	13.0 (0.8)	86.2 (0.5)	6.2 (0.1)	88.6 (1.7)
	QQP†	68.7 (1.3)	26.5 (3.7)	66.9 (1.0)	17.3 (2.1)	79.6 (0.6)	10.1 (4.2)	76.3 (0.4)	3.5 (0.7)	93.3 (4.9)
	Avg.	73.3	25.3	70.6	17.5	84.3	13.2	80.9	3.6	88.9

4.1. Skill Localization improves Calibration

Human skill in a task usually includes some awareness of when the task has not been performed well. In ML, *calibration* is an attempt to formalize this. Suppose a classification model outputs, given input x and each possibly label y, a probability $\Pr[y|x]$ that the label y is correct for x. In order to be *well-calibrated*, this probability should be meaningful, i.e., among all x, y such that $\Pr[y|x] = a$, the expected fraction of these where y is the correct (ground truth) label should also be around a. It is well-known that usual softmax outputs in ML models are not well-calibrated. (This can be mitigated given enough held-out data by re-calibrating the output.) Could skill localization help with calibration? This could be interesting in low-data settings where re-calibration is impossible.

Table 2 reports the calibration error using the ECE metric (Naeini et al., 2015) (described in Appendix A.3) in a pre-trained model with sparse grafting using tasks from GLUE dataset. Sparsity levels that cause $< 5\%$ reduction in accuracy lead to $40 - 90\%$ reduction in ECE. Vanilla fine-tuning is highly overconfident in wrong predictions, and grafted models avoid this; see histograms in Figure 6,

Comparison with re-training. Our sparse grafting involves no re-training. Does re-training just the grafted parameters affect calibration? Table 1 shows that impressive calibration persists after re-training. This suggests that the sparse grafting region identified by our method is, in some sense, fundamentally suited to the task.

Note that several recent papers have tried to sparsify fine-tuned nets by identifying sub-networks of interest and re-training their parameters – one example is BitFit. Table 1 finds that BitFit is better calibrated than vanilla fine-tuning, but worse than our grafted model after re-training. This suggests that the sparse regions identified by our procedure are better at localizing the skill.

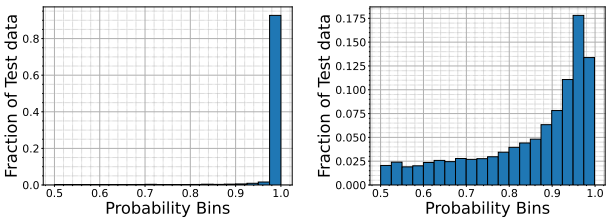


Figure 6. Histogram of top prediction probabilities for FT model and grafted model on QNLI (4096-shot). (**left**) The model assigns high confidence on most examples. (**right**) The grafted model has diverse confidence levels, explaining its superior calibration.

4.2. Out-of-Distribution Generalization

Human skills extend at least a bit to new settings; e.g., skill at throwing a baseball should also lead to ability to throw a tennis ball. We evaluated in-distribution (ID) and

out-of-distribution (OOD) accuracies for grafted models of varying sparsities in Figure 7. We find that grafted models may suffer a little on ID performance but match or significantly outperform vanilla fine-tuning on OOD. This suggests that grafting has indeed captured "core" skills.

Small distribution shifts. When the distribution shift between tasks is intuitively small (e.g. SST-2 to IMDb or MNLI to SNLI) vanilla fine-tuned model itself is robust enough —grafting provides little or no advantage. Similar findings appear in Hendrycks et al. (2020).

Large distribution shifts. Sentiment analysis task MPQA uses text from news articles while SST-2/Yelp/Amazon uses reviews. We find that models fine-tuned on MPQA perform poorly when tested on SST-2 and Yelp. However, the grafted model for MPQA performs at least 5% better than the fine-tuned model. Similar results hold for NLI datasets. QNLI task consists of question/answer pairs whereas MNLI and SNLI[2] have pairs of assertions as inputs. This distribution shift is enough to make vanilla fine-tuning on QNLI perform poorly on MNLI and SNLI, but the grafted model for QNLI again performs around 5% better.

Comparison with WiSE-FT. Often there is no magic bullet for doing well on both ID and OOD generalization. For image data (with model pre-trained on CLIP), it was shown that WiSE-FT (Wortsman et al., 2022), which linearly interpolates between $\boldsymbol{\theta}_{\text{ft}}$ and $\boldsymbol{\theta}_{\text{pre}}$, does best in the ID-OOD trade-off. Figure 7 explores similar ideas for NLP tasks. Model grafting is better than WiSE-FT for one ID-OOD pair, but the opposite is true for a different pair. Applying the WiSE-FT idea on the grafted model (i.e. interpolating between grafted model and pre-trained model), "WiSE Graft," gets competitive ID-OOD tradeoff to Wise FT.

Comparison with parameter-efficient methods. To investigate if the number of updated parameters is the key factor behind the better performance of sparse grafted models in terms of calibration and OOD generalization, we assess the calibration error and OOD accuracy of few parameter-efficient fine-tuning methods. We find that, at comparable sparsity levels, grafting exhibits better calibration and OOD performance. This suggests that the sparsity of the grafting region is not the only factor for its better OOD and calibration performance. Furthermore, the OOD and calibration performance of grafting change gradually with decreasing sparsity. Please see Appendix B.3 for more details.

4.3. Understanding Generalization for Fine-tuning

Fine-tuning a vast pre-trained model on a small dataset seems iffy since we are finding the best model in a very large class of models Θ, and according to the classic understanding of generalization, the error could be as high as $\sup_{\boldsymbol{\theta}\in\Theta} |\mathcal{L}_{\text{test}}(\boldsymbol{\theta}) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta})|$. This bound is too pessimistic for most deep learning settings (Nagarajan & Kolter, 2019), including fine-tuning, since the training data can be easily fit perfectly in these settings.

[2]We consider only contradiction and entailment labels here.

Table 3. Comparing the OOD performance of model grafting, graft re-training and BitFit. For NLI tasks, the OOD accuracy of graft re-training is 5% worse than model grafting. BitFit completely fails on OOD tasks when the distribution shift is large.

OOD task	Grafting	Graft re-training	BitFit
ID task: SST-2			
Yelp	89.5 (0.3)	88.9 (1.0)	89.0 (0.3)
IMDb	81.5 (0.7)	81.2 (1.4)	81.3 (0.7)
ID task: QNLI			
MNLI(0/1)	71.8 (1.8)	67.0 (2.3)	60.5 (4.9)
SNLI(0/1)	80.1 (2.9)	66.4 (7.1)	57.4 (8.0)

Understanding generalization of the grafted model is more tractable because of the small size of the grafting region. Empirically we find that re-training on the grafted parameters fails to make $|\mathcal{L}_{\text{test}}(\boldsymbol{\theta}) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta})|$ higher than 1% once the dataset has a few thousand datapoints, which can be formalized as a "complexity parameter". Appendix D explores this further using classical generalization theory.

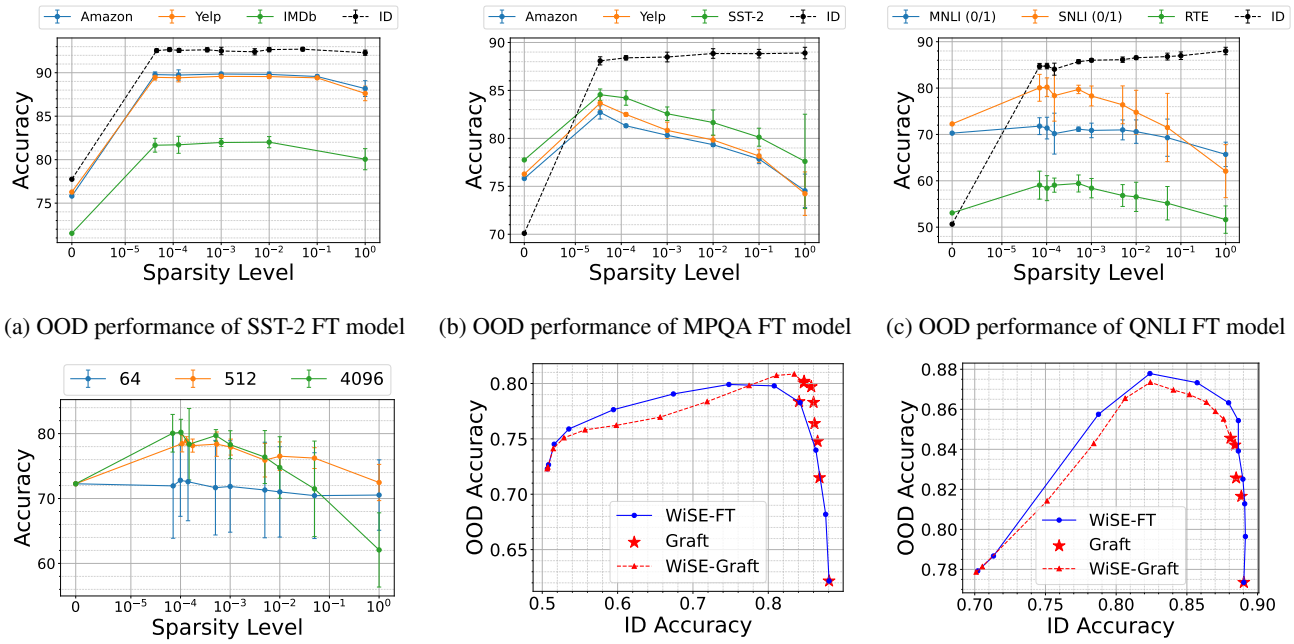
5. Multi-Task and Continual Learning

Previous sections have shown that sparse grafts can localize skills when fine-tuning on a single task. In this section, we test a stronger version of skill localization involving multiple tasks in the following settings: (i) the model is fine-tuned on many tasks together, (ii) continual learning, where the model is fine-tuned on one task at a time.

5.1. Multi-Task Learning

We perform multi-task learning (MT) by fine-tuning a RoBERTa-base model with SGD on 8 different datasets (4096-shot setting for each) simultaneously. The datasets represent four different classes of tasks: NLI, sentiment analysis, paraphrasing, and classification. Firstly, the resulting MT model achieves test accuracy comparable with the task-specific FT models, suggesting no *gradient interference* that is observed in some cases (Yu et al., 2020). For skill localization, we learn task-specific sparse regions: for each task i, we optimize for $\mathcal{L}_i$ (i.e., performance on task i) from Equation (3) using the MT model parameters as $\boldsymbol{\theta}_{\text{ft}}$ and $\boldsymbol{\gamma}_{\text{base}} = \mathbf{0}$. (Note that grafted models for i, j have the same value for parameters that are contained in both $\boldsymbol{\gamma}_i$ and $\boldsymbol{\gamma}_j$.) Results are presented in Figure 8.

We find skill localization continues to exist in multi-task models, and now also provides signal about task simi-



(a) OOD performance of SST-2 FT model (b) OOD performance of MPQA FT model (c) OOD performance of QNLI FT model

(d) QNLI → SNLI (0/1) for varying k-shot (e) ID-OOD curves for QNLI → SNLI (0/1) (f) ID-OOD curves for MPQA → SST-2

Figure 7. Comparing the zero-shot OOD performance of the FT model and grafting in various settings. (b,c) We observe at least a 5% gap between the performance of the two, when the distribution shifts are large. (c) The gap gets worse as the number of available in-distribution samples increases. (e) For transfer in NLI task, the optimal (ID, OOD) point for WiSE-FT is (80.5, 79.6), and for grafting (84.2, 80.0). (f) For transfer in sentiment task, WiSE-FT on sparse grafted models (WiSE-Graft) gets a competitive ID-OOD curve.

larity (through region overlap) and affords interesting compositional properties (through the union of regions).

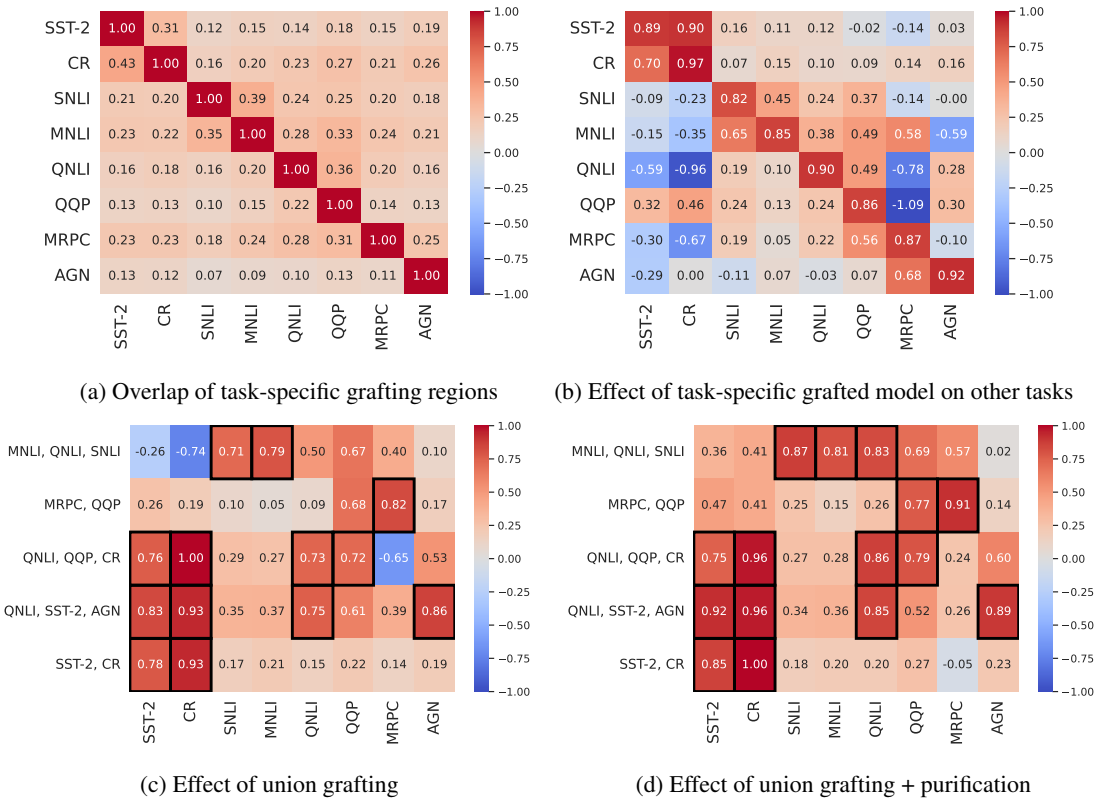
Region overlap and task similarity. Figure 8a shows that the patches for different tasks have very little overlap (defined as $\frac{|\gamma_i \cap \gamma_j|}{|\gamma_j|}$ for tasks i, j). However, similar tasks show slightly more overlap compared to other pairs, e.g. (SST-2, CR), and (SNLI, MNLI). This is some evidence of skill-sharing across similar tasks.

Skill isolation and transfer. In Figure 8b we find that grafted models for a single task, which presumably isolate the skills for just that task, indeed only help that particular task and a few closely related tasks. We measure the effect of task i on task j by grafting only the parameters in the region γ_i, and measuring the performance gain on task j compared to the performance gain of the MT model. For a task t, if $P_{\boldsymbol{\gamma},t}$ is the accuracy of the model grafted with $\boldsymbol{\gamma}$, $P_{\mathbf{0},t}$ is the pre-trained accuracy, and $P_{\mathbf{1},t}$ is the MT model accuracy, then relative performance gain of grafting region $\boldsymbol{\gamma}$ is

$$\text{Rel}_{\boldsymbol{\gamma},t} = (P_{\boldsymbol{\gamma},t} - P_{\mathbf{0},t})/(P_{\mathbf{1},t} - P_{\mathbf{0},t}) \tag{5}$$

We find that some similar pairs of tasks like (SST-2, CR), and (SNLI, MNLI) show transfer, i.e. grafting the region from one task helps with the other. Interestingly, for some tasks that are seemingly similar (e.g. QQP and MRPC) the effect seems to be asymmetric, i.e. $\boldsymbol{\gamma}_{\text{MRPC}}$ helps with QQP, but $\boldsymbol{\gamma}_{\text{QQP}}$ does not help with MRPC. Furthermore, we observe that, γ_{QNLI} helps with QQP paraphrasing, presumably because they both have questions in their inputs.

Skill compositionality through region unions: Since grafting for a single task works for that task as well as related tasks, we ask a more ambitious question: *Can grafting multiple regions lead to skill isolation for that subset of tasks?* A priori one would guess "No," because the grafting regions were independently trained on individual tasks without any compositionality requirements. Surprisingly we find the answer is a qualified "yes." Figure 8c presents compositionality results for 5 groups of tasks. For each group G, we take the union of regions $\boldsymbol{\gamma}_G = \cup_{i \in G} \gamma_i$ and evaluate the relative performance gains for tasks using the grafted model of $\boldsymbol{\gamma}_G$. We indeed find that composing tasks for a subset in this way retains around 70% of the accuracy gains for that subset and related tasks, but not for other tasks. We tried slightly fine-tuning the union region $\boldsymbol{\gamma}_G$ to optimize the joint loss on tasks in G, i.e. $\sum_{i \in G} \mathcal{L}_i(\boldsymbol{\gamma})$, by only taking 10 gradient steps for quick adaptation. Figure 8d shows that this causes the accuracy gain on relevant tasks to be even higher (around $\sim 80\%$) without affecting gains for other tasks by much. The emergence of this compositionality property, we believe, is very interesting and deserves further exploration.



(a) Overlap of task-specific grafting regions

(b) Effect of task-specific grafted model on other tasks

(c) Effect of union grafting

(d) Effect of union grafting + purification

Figure 8. Ablations on task-specific grafting region γ_i for task i, learned by optimizing $\mathcal{L}_i$ on the MT model. Section 5.1 has details of experiments and the metrics being reported. In all figures, we evaluate the effect of graft region of task in row i on the task in row j. Figure (a) measures the assymteric overlap in the regions defined as $\frac{|\gamma_i \cap \gamma_j|}{|\gamma_j|}$ for tasks in row i and column j. Figure (b), (c), (d) evaluate the relative accuracy gain of task in column j using the graft regions of task(s) in row i; refer to Equation (5) for the precise expression. **Observations:** (a) Similar tasks, like (SST-2, CR), and (SNLI, MNLI), show relatively higher overlap in grafting regions. (b) The grafted model of a task only does well on itself and a few similar tasks. (c) The grafted model with union of regions for a subset G of tasks only does well on the tasks in G and similar tasks (all values higher than 0.7). (d) Allowing a few steps of GD to purify the union of grafting regions improves the grafted model's performance on the desired set of tasks.

AdamW MT training: We also check skill localization in a model trained with AdamW (see Figure 13 in the Appendix). Interestingly, we find that the task-specific grafts show small overlap across tasks, and only perform well on similar tasks, indicating localization even in the absence of an explicit ℓ_1 regularization during training. This is in stark contrast to single task trained models, which had failed to show any skill localization without an explicit ℓ_1 regularization (Figures 5a and 5b). We speculate that forcing the model to do well on multiple tasks together naturally encourages the model to localize skills.

5.2. Forget-free Continual Learning

Continual learning aims to train a model sequentially, seeing one task at a time. A frequent complication is *catastrophic forgetting* (see chapter 4 in Chen & Liu (2018)): training on new tasks can greatly hurt performance on earlier tasks. Here skill localization could help: once skills for previous tasks have been localized, we can freeze them and only update the rest of the net for new tasks. We use this localization idea, through our grafting procedure, to perform forget-free continual learning, i.e. without forgetting anything about previous tasks.

Table 4. Continual learning on the sequence of tasks QNLI, AG news, SST-2. The naive continual FT leads to a 20% drop in accuracy for QNLI, owing to catastrophic forgetting. Grafting continual FT (our procedure) can retain the performance on QNLI, while minimally affecting the performance of newer tasks.

Method	QNLI	AGNews	SST-2
FT	88.0 (0.8)	93.1 (0.1)	92.1 (0.1)
ContinualFT	67.5 (5.3)	87.6 (2.5)	92.0 (0.5)
GraftingContinual	86.5 (0.7)	90.8 (0.1)	92.5 (0.3)

The main idea is to only train the parameters in the grafting region for a task, that does not intersect with the grafting regions of the previously encountered tasks. During

inference, inspired by (Kang et al., 2022), we only use the grafted model that takes the union of the grafting regions of the task and the previous tasks. While this requires resetting parameters to pre-trained values for each evaluation, the total memory needed to retain all skills is propositional to Ts instead of Td, where T is the total number of tasks, s ($\sim$ 5000) is the sparsity of the grafting regions and d is the total number of parameters in the model ($\sim$ 100M). Preliminary experiments in Table 4 on a sequence of three tasks suggest a significant benefit of skill localization. Appendix C.3 provides more details on training and evaluation procedures, and includes a discussion on why grafting helps in this case. Other explorations of skill localization and grafting for continual learning is left for future work.

6. Related Work

Knowledge/skills. Li et al. (2022) show that the feed-forward activations are sparse in large pre-trained models. Dai et al. (2022) discover knowledge neurons in BERT, whose activations correlate with specific facts, whereas Burns et al. (2022) find latent knowledge in the internal representations of language models. Furthermore, Meng et al. (2022); Hase et al. (2023) show that the language models localize knowledge in the feed-forward layers of the pre-trained models. Wang et al. (2022) find specific "skill" neurons highly predictive of the downstream task in soft prompt-tuning (Li & Liang, 2021) of language models. Xie et al. (2022) demonstrate that a layerwise analysis of hidden state variations in pre-trained language models can aid in the identification of an optimal subset of task-specific layers for fine-tuning purposes.

Parameter-efficient fine-tuning (PEFT). The goal is to update very few parameters for efficient training. Houlsby et al. (2019) only train a small set of trainable parameters added between layers of the network. Gordon et al. (2020) use an ℓ_0 regularizer to update few layers during fine-tuning. BitFit (Ben Zaken et al., 2022) only updates biases during FT, and performs comparably to vanilla FT. (Our graft regions have fewer parameters.) Grafting only retains updates for few parameters, but is not motivated by efficiency considerations. Leveraging grafting for better PEFT could be an interesting future direction.

Lottery ticket hypothesis (Frankle & Carbin, 2018) asserts that a trained neural network can be re-trained using a small sub-network while setting other parameters to 0 and still reach the same performance. Lottery tickets for pre-trained language models are studied in (Chen et al., 2020; Prasanna et al., 2020; Liang et al., 2021). To our best knowledge, LTH results in sub-networks much denser than graft.

While Gong et al. (2022) claim to find a much sparser "lottery ticket" that transfers to different GLUE tasks, to the best of our understanding, they set parameters outside the ticket to the pre-trained value and not 0. Their aim is to design a PEFT technique, whereas grafting aims to understand the mechanism of fine-tuning and finding *task-specific* regions in fine-tuned models. Since grafting is post-hoc, i.e. involves no further fine-tuning, it provides a lens for a more holistic evaluation of fine-tuning.

OOD generalization and distribution shifts. Diffenderfer et al. (2021); Zhang et al. (2021); Liu et al. (2022) show that re-trained lottery tickets can be more robust to the distribution shift. Lee et al. (2023) alleviate distribution shift by only fine-tuning specific layers of the whole model. Li et al. (2022) show the efficacy of sparsifying the activations of feed-forward layers in the models.

Multi-task training. Misra et al. (2016) stitch networks from different tasks together, Long et al. (2017) learn relations between tasks to enhance performance, and Lu et al. (2017) apply adaptive weight-sharing between the networks for different tasks. Multi-task training in NLP is also studied in (Collobert & Weston, 2008; Liu et al., 2016; Gupta et al., 2016; Lu et al., 2017; Liu et al., 2019a). These involve training additional task-specific parameters whereas our approach does not. Also, we attempt to understand the localization of skills within the model post-hoc.

Saunshi et al. (2021); Wei et al. (2021); Malladi et al. (2022) mathematically study head-tuning, prompt-tuning and fine-tuning of language models for few-shot downstream tasks.

7. Conclusions and future directions

By successfully demonstrating the ability to do a sparse"graft" of the skill on top of the pre-trained model, this paper makes a start on localizing newly acquired skills inside fine-tuned language models. We hope our first-cut method will improve with further work, potentially yielding better understanding as well as applications in multi-task and continual learning, which we also begin to address in Section 5.2. We hope these may yield new insights on how to compose skills, decompose the identified skill into finer skills, and give applications to unlearning and federated learning. One open problem for multi-task setting is a method to find, for any subset $S \subseteq \{1, \ldots, m\}$ of tasks, a model that does well on all tasks in S. (The naive method would train models for all 2^m subsets of tasks.) Our approach with finding task-specific regions and using their unions shows promise for small m.

Acknowledgments. We thank: Danqi Chen for feedback on an earlier draft; Saurabh Garg for pointer to WiSE-FT; Tianyu Gao and Mengzhou Xia for pointers on the prompt-based fine-tuning codebase; the anonymous reviewers for their helpful comments. This work is supported by funding from NSF, ONR, Simons Foundation, DARPA and SRC.

References

Arora, S., Ge, R., Neyshabur, B., and Zhang, Y. Stronger generalization bounds for deep nets via a compression approach. In *International Conference on Machine Learning*. PMLR, 2018.

Bansal, Y., Nakkiran, P., and Barak, B. Revisiting model stitching to compare neural representations. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=ak06J5jNR4.

Ben Zaken, E., Goldberg, Y., and Ravfogel, S. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.1. URL https://aclanthology.org/2022.acl-short.1.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*, 2022.

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Wang, Z., and Carbin, M. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems*, 33:15834–15846, 2020.

Chen, Z. and Liu, B. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2018.

Collobert, R. and Weston, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pp. 160–167, 2008.

Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., and Wei, F. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8493–8502, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.581. URL https://aclanthology.org/2022.acl-long.581.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. 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)*, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. URL https://aclanthology.org/N19-1423.

Diffenderfer, J., Bartoldson, B., Chaganti, S., Zhang, J., and Kailkhura, B. A winning hand: Compressing deep networks can improve out-of-distribution robustness. *Advances in Neural Information Processing Systems*, 34: 664–676, 2021.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

Gao, T., Fisch, A., and Chen, D. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2021a. doi: 10.18653/v1/2021.acl-long.295. URL https://aclanthology.org/2021.acl-long.295.

Gao, Y., Colombo, N., and Wang, W. Adapting by pruning: A case study on bert. *arXiv preprint arXiv:2105.03343*, 2021b.

Gong, Z., He, D., Shen, Y., Liu, T.-Y., Chen, W., Zhao, D., Wen, J.-R., and Yan, R. Finding the dominant winning ticket in pre-trained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1459–1472, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.115. URL https://aclanthology.org/2022.findings-acl.115.

Gordon, M., Duh, K., and Andrews, N. Compressing BERT: Studying the effects of weight pruning on transfer learning. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pp. 143–155, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.repl4nlp-1.18. URL https://aclanthology.org/2020.repl4nlp-1.18.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Gupta, P., Schütze, H., and Andrassy, B. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 2537–2547, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee. URL https://aclanthology.org/C16-1239.

Hase, P., Bansal, M., Kim, B., and Ghandeharioun, A. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *arXiv preprint arXiv:2301.04213*, 2023.

Hendrycks, D., Liu, X., Wallace, E., Dziedzic, A., Krishnan, R., and Song, D. Pretrained transformers improve out-of-distribution robustness. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2744–2751, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.244. URL https://aclanthology.org/2020.acl-main.244.

Holtzman, A., West, P., Shwartz, V., Choi, Y., and Zettlemoyer, L. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7038–7051, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.564. URL https://aclanthology.org/2021.emnlp-main.564.

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., and Gelly, S. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pp. 2790–2799. PMLR, 2019.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Kalimeris, D., Kaplun, G., Nakkiran, P., Edelman, B., Yang, T., Barak, B., and Zhang, H. Sgd on neural networks learns functions of increasing complexity. *Advances in neural information processing systems*, 2019.

Kang, H., Mina, R. J. L., Madjid, S. R. H., Yoon, J., Hasegawa-Johnson, M., Hwang, S. J., and Yoo, C. D. Forget-free continual learning with winning subnetworks. In *International Conference on Machine Learning*. PMLR, 2022.

Kumar, A., Liang, P. S., and Ma, T. Verified uncertainty calibration. *Advances in Neural Information Processing Systems*, 32, 2019.

Kumar, A., Shen, R., Bubeck, S., and Gunasekar, S. How to fine-tune vision models with sgd. *arXiv preprint arXiv:2211.09359*, 2022.

Lee, Y., Chen, A. S., Tajwar, F., Kumar, A., Yao, H., Liang, P., and Finn, C. Surgical fine-tuning improves adaptation to distribution shifts. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=APuPRxjHvZ.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Li, Z., You, C., Bhojanapalli, S., Li, D., Rawat, A. S., Reddi, S. J., Ye, K., Chern, F., Yu, F., Guo, R., et al. Large models are parsimonious learners: Activation sparsity in trained transformers. *arXiv preprint arXiv:2210.06313*, 2022.

Liang, C., Zuo, S., Chen, M., Jiang, H., Liu, X., He, P., Zhao, T., and Chen, W. Super tickets in pre-trained language models: From model compression to improving generalization. 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)*, pp. 6524–6538, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.510. URL https://aclanthology.org/2021.acl-long.510.

Liu, P., Qiu, X., and Huang, X. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 2873–2879, 2016.

Liu, X., He, P., Chen, W., and Gao, J. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4487–4496, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1441. URL https://aclanthology.org/P19-1441.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019b.

Liu, Y., Meng, F., Lin, Z., Li, J., Fu, P., Cao, Y., Wang, W., and Zhou, J. A win-win deal: Towards sparse and robust pre-trained language models. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 19189–19202. Curran Associates, Inc., 2022.

Long, M., Cao, Z., Wang, J., and Yu, P. S. Learning multiple tasks with multilinear relationship networks. *Advances in neural information processing systems*, 30, 2017.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Lu, Y., Kumar, A., Zhai, S., Cheng, Y., Javidi, T., and Feris, R. Fully-adaptive feature sharing in multi-task networks with applications in person attribute classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5334–5343, 2017.

Malladi, S., Wettig, A., Yu, D., Chen, D., and Arora, S. A kernel-based view of language model fine-tuning. *arXiv preprint arXiv:2210.05643*, 2022.

Meng, K., Bau, D., Andonian, A. J., and Belinkov, Y. Locating and editing factual associations in GPT. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=-h6WAS6eE4.

Misra, I., Shrivastava, A., Gupta, A., and Hebert, M. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3994–4003, 2016.

Naeini, M. P., Cooper, G., and Hauskrecht, M. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Nagarajan, V. and Kolter, J. Z. Uniform convergence may be unable to explain generalization in deep learning. In *Advances in Neural Information Processing Systems*, 2019.

Negrea, J., Dziugaite, G. K., and Roy, D. In defense of uniform convergence: Generalization via derandomization with an application to interpolating predictors. In *International Conference on Machine Learning*. PMLR, 2020.

Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., and Gurevych, I. 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*, pp. 487–503, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.39. URL https://aclanthology.org/2021.eacl-main.39.

Prasanna, S., Rogers, A., and Rumshisky, A. When BERT Plays the Lottery, All Tickets Are Winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3208–3229, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.259. URL https://aclanthology.org/2020.emnlp-main.259.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

Saunshi, N., Malladi, S., and Arora, S. A mathematical exploration of why language models help solve downstream tasks. In *International Conference on Learning Representations*, 2021.

Sung, Y.-L., Nair, V., and Raffel, C. A. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34:24193–24205, 2021.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446.

Wang, X., Wen, K., Zhang, Z., Hou, L., Liu, Z., and Li, J. Finding skill neurons in pre-trained transformer-based language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11132–11152, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.765.

Wei, C., Xie, S. M., and Ma, T. Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning. *Advances in Neural Information Processing Systems*, 2021.

Wortsman, M., Ilharco, G., Kim, J. W., Li, M., Kornblith, S., Roelofs, R., Lopes, R. G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7959–7971, 2022.

Xie, S., Qiu, J., Pasad, A., Du, L., Qu, Q., and Mei, H. Hidden state variability of pretrained language models can guide computation reduction for transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 5750–5768, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-emnlp.422.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 2020.

Zhang, D., Ahuja, K., Xu, Y., Wang, Y., and Courville, A. Can subnetwork structure be the key to out-of-distribution generalization? In *International Conference on Machine Learning*. PMLR, 2021.

A. Additional Experimental Details

A.1. Architecture Details on Fine-tuning

We fix the embeddings and the language model head of the pre-trained model during fine-tuning with prompts. We observe an improvement in the model's performance, when we fix the embedding layer while fine-tuning with SGD. This was also observed by Kumar et al. (2022) for vision transformers. We fix the language model head, to stay consistent with the fact that the weights of the language model head are an exact copy of the weights of the embedding layer in the pre-trained model. Furthermore, fixing the head for RoBERTa does as well as learning, but reduces the sizes of regions. For standard fine-tuning experiments, we fix only the embeddings of the pre-trained model.

A.2. Hyperparameter settings

For SGD, we follow the grid $\{2, 4, 8\}$ for batch size and $\{10^{-2}, 5 \times 10^{-3}, 10^{-3}\}$ for learning rate and apply a small weight decay of 10^{-4} on all the model parameters during training. Following (Malladi et al., 2022), we train the model for $16 \times n \times k$ steps in k-shot tuning on a task with n labels. However, to reduce training time for 4096(and higher)-shot experiments, we stop training after $16 \times n \times 512$ steps and observe that the model converges to 100% train accuracy during this time. We fix the randomness of the model during FT by using seed 0. In all our notations, a k-shot dataset for a task refers to a dataset with k examples per classification class. For prompt-based fine-tuning, we do not use demonstrations in the context of each input example as proposed in Gao et al. (2021a).

A.3. Calibration Error

The ECE metric measures the difference between the confidence and accuracy of a model. We use the implementation from Guo et al. (2017) and briefly describe the metric below. Predictions are grouped into M interval bins (each of size $1/M$) and the accuracy of each bin is computed. If B_m is the set of indices of samples whose prediction confidence falls into the interval $I_m = [\frac{m-1}{M}, \frac{m}{M})$, then ECE is given by

$$\sum_{m=1}^{M} \frac{|B_m|}{n} |\text{conf}(B_m) - \text{acc}(B_m)|, \tag{6}$$

where $\text{acc}(B_m)$ and $\text{conf}(B_m)$ denote the accuracy of the model and the average of all the predictions on the examples in the bin B_m respectively.

One of the popular approaches to improve the calibration of a model is to learn a function on top of the logits of a trained model using a hold-out set. However, learning a function with ϵ ECE requires at least $\Omega(1/\epsilon^2)$ samples (Kumar et al., 2019). This is difficult in few-shot settings, where we only have less than 100 examples to train with. Thus, grafting provides a direct way to improve the calibration of the model, with a minimal drop in performance.

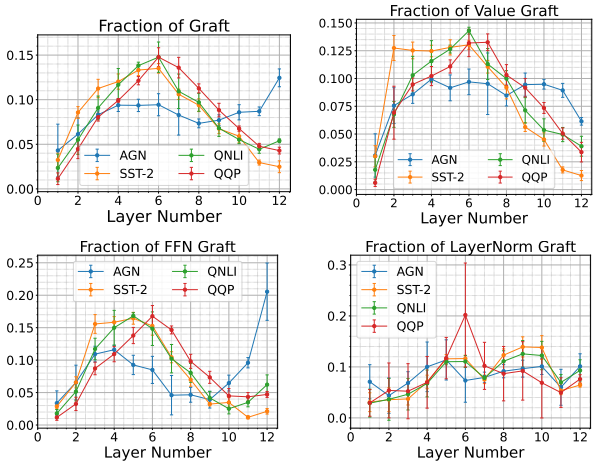


Figure 9. Distribution of graft parameters in different regions of the model across layers. (TL) All parameters, (TR) Value parameters of attention module, (BL) first layer parameters of the feedforward module, (BR) LayerNorm parameters (feedforward and attention combined). For most of the tasks, the graft parameters are concentrated more in the middle layers.

B. Additional experiments

B.1. Distribution of graft parameters

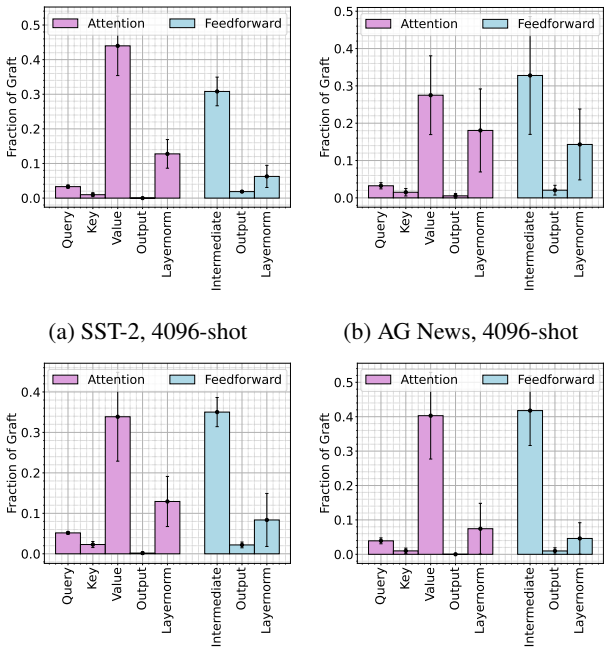
We perform a closer analysis of the distribution of graft parameters for different downstream tasks in Figures 9 and 10. Firstly we observe, in Figure 9, that most of the graft region is concentrated in the middle layers for most tasks; AG News is an exception. Furthermore, a closer look into the distribution of the graft parameters in Figure 10 reveals an interesting pattern. Most of the graft parameters are concentrated in three parameter types: (1) Value parameters of the attention module, (2) the first layer of the feed-forward module, and (3) LayerNorm parameters. This observation could potentially provide a deeper understanding of the mechanism of fine-tuning and the role of pre-training. The LayerNorm parameters in the grafting region are uniformly distributed across layers for all tasks, as evident in the bottom right of Figure 9. The layer-wise distribution of the value parameters and the first layer parameters of the feed-forward module in the grafting region show varying patterns among different tasks.

B.2. Grafts with a larger training data

Most experiments in the previous sections perform fine-tuning in the few-shot or mid-shot settings. Here we verify

Table 5. We consider a larger training set for each downstream task from Table 2, where we allow $50k$ training examples per classification class. The Sentiment dataset represents the combination of MR, CR, MPQA, and SST-2. The main findings are (1) Grafted models with 0.03% sparse grafting regions can retrieve $> 95\%$ of the FT accuracy while being better calibrated than the original model itself. For single-sentence tasks, the grafted model shows only a 0.3% drop in accuracy but an improvement of 6% in the calibration error. Similarly for two-sentence tasks, the grafted model shows a 2.8% drop in accuracy with an improvement of 4.0% in the calibration error.

	FT		0.03% Sparse Graft		0.06% Sparse Graft		0.12% Sparse Graft	
Dataset	Acc.	ECE	Acc.	ECE	Acc.	ECE	Acc.	ECE
Sentiment	89.2	9.3	89.5	2.7	89.6	3.3	89.4	3.6
AG News	92.8	6.1	91.9	0.8	91.8	0.9	91.9	0.8
QNLI	91.8	5.5	88.9	1.2	89.2	1.2	89.6	1.3
SNLI	88.6	5.1	85.9	1.4	86.2	1.3	86.3	1.4



(a) SST-2, 4096-shot (b) AG News, 4096-shot

(c) QNLI, 4096-shot (d) QQP, 4096-shot

Figure 10. Distribution of graft parameters in attention and feedforward modules. For feedforward module, Intermediate denotes its first layer and Output denotes its second layer. Most of the graft parameters are concentrated in the Value parameters of the attention module, the first layer of the feedforward module, and the LayerNorm parameters.

that skill localization via grafting also happens in full-shot models. We repeat the experiments in Table 2 for a few datasets with almost their entire training data, as we allow $50k$ training examples per classification class. We make a random $95\% - 5\%$ split of the training set to have a validation set for hyperparameter tuning. Since all the sentiment analysis tasks (MPQA, SST-2, CR, MR) considered in our paper have at most $10k$ training examples, we merge them to form one Sentiment dataset. For training models, we follow the same hyperparameter tuning procedure from Appendix A.2. With $\sim 10\times$ increase in training examples, we also increase the number of epochs during mask training by $10\times$, while using the same set of hyperparameters as given in Section 3.

We observe that grafted models using 0.03% sparse grafting regions can achieve $> 95\%$ of the final model performance, while also showing an absolute improvement of 4.5% in calibration error. With a $10\times$ increase in the number of training examples, the necessary grafting region however doesn't increase by $10\times$. This indicates that the grafting region doesn't increase proportionally with the number of training samples, which points to some form of task-specific intrinsic dimensionality for fine-tuning. This is left for future exploration.

B.3. OOD and calibration comparisons with parameter-efficient methods

Since calibration and OOD generalization for sparse grafted models are better than fully fine-tuned models, a natural question is whether the number of parameters being updated is the main reason for this good performance. To test this hypothesis, we perform two sets of experiments, (1) evaluate the calibration error and OOD accuracy of parameter-efficient fine-tuning methods, and (2) evaluate these metrics for grafted models at different levels of sparsity.

Parameter-efficient methods. We report the calibration and the OOD performance of three parameter efficient methods, LoRA (Hu et al., 2022) and two adapter-based methods (Houlsby et al., 2019; Pfeiffer et al., 2021) in Tables 6 and 7. We do not change the hyperparameter setting for these techniques from their corresponding papers, i.e. we train these methods with AdamW optimizer and follow the corresponding papers for hyperparameter tuning. We report at different sparsity levels, to allow a fairer comparison across the methods.

Table 6. Comparing the accuracy (†F1) in % and the calibration error (ECE) for model grafting, LoRA (Hu et al., 2022), and two adapter methods (Pfeiffer et al., 2021; Houlsby et al., 2019) at different sparsity levels. Results are in the 4096-shot setting. The major observations are: (a) At similar sparsity levels ($0.05 - 0.1\%$), grafting achieves an improvement of 4.6% in calibration error while being only 1.7% lower in accuracy, implying parameter count is not the only reason behind the calibration results for grafting. (b) For grafting, calibration error, and accuracy change smoothly with the grafting region sparsity.

		SST-2		AGNews		QNLI		QQP†		Average	
	Rank/ Sparsity	Acc.	ECE	Acc.	ECE	Acc.	ECE	F1	ECE	Acc./F1	ECE
LoRA	1 (0.05%)	92.8 (0.5)	6.2 (0.7)	93.1 (0.2)	5.1 (0.4)	87.6 (0.3)	8.0 (1.9)	79.4 (0.2)	9.9 (2.8)	88.2	7.3
	2 (0.1%)	93.0 (0.2)	5.7 (0.8)	93.0 (0.3)	5.1 (0.9)	87.5 (0.4)	9.0 (2.4)	79.2 (0.6)	13.7 (3.6)	88.2	8.4
	8 (0.4%)	93.4 (0.4)	5.8 (0.9)	93.1 (0.1)	5.7 (0.6)	87.6 (0.3)	9.7 (2.1)	79.7 (0.3)	14.3 (2.3)	88.5	8.9
Houlsby et al.	1 (0.025%)	92.9 (0.5)	6.3 (1.0)	93.2 (0.2)	5.5 (0.9)	88.1 (0.6)	10.2 (1.7)	79.7 (0.4)	13.6 (3.0)	88.5	8.9
	2 (0.05%)	92.9 (0.6)	5.8 (1.1)	93.0 (0.4)	5.4 (1.4)	88.0 (0.4)	9.8 (1.8)	79.3 (02)	13.7 (2.8)	88.3	8.7
	16 (0.4%)	93.2 (0.3)	6.2 (0.9)	93.3 (0.2)	5.5 (0.8)	88.4 (0.8)	11.3 (0.6)	80.0 (0.3)	15.1 (1.3)	88.7	9.5
	64 (1.6%)	92.6 (0.6)	6.9 (0.8)	93.3 (0.1)	6.1 (0.5)	88.7 (0.2)	10.3 (1.6)	80.1 (0.3)	15.8 (0.2)	88.7	9.8
Pfeiffer et al.	1 (0.025%)	92.8 (0.6)	6.4 (1.0)	93.2 (0.2)	5.4 (0.9)	88.0 (0.7)	10.1 (1.8)	79.7 (0.4)	13.7 (3.1)	88.4	8.9
	2 (0.05%)	93.1 (0.4)	6.0 (1.4)	93.0 (0.2)	4.5 (0.6)	87.4 (0.4)	7.7 (2.5)	78.9 (0.2)	12.5 (4.2)	88.1	7.7
	16 (0.4%)	92.5 (0.4)	6.2 (0.8)	93.1 (0.2)	5.1 (0.8)	87.8 (0.4)	9.5 (1.6)	79.6 (0.4)	13.5 (2.2)	88.3	8.6
	64 (1.6%)	93.1 (0.5)	6.7 (0.5)	93.1 (0.1)	5.8 (0.7)	87.9 (0.3)	9.9 (2.2)	79.6 (0.4)	14.9 (3.5)	88.5	9.4
Grafting	0.01%	92.4 (0.1)	3.2 (0.4)	91.1 (0.9)	0.9 (0.2)	84.7 (0.6)	1.0 (0.3)	76.3 (0.4)	3.5 (0.7)	86.1	2.2
	0.05%	92.5 (0.5)	4.0 (0.9)	89.9 (0.5)	0.9 (0.1)	86.0 (0.4)	1.9 (0.8)	77.6 (0.4)	3.9 (0.8)	86.5	2.7
	0.1%	92.5 (0.4)	4.4 (0.7)	90.8 (0.1)	1.1 (0.3)	86.0 (0.4)	2.2 (0.9)	78.2 (0.0)	4.5 (1.3)	86.9	3.1
	1%	92.3 (0.3)	4.6 (0.7)	91.0 (0.3)	1.2 (0.2)	86.5 (0.4)	3.1 (1.1)	78.7 (0.2)	5.4 (1.7)	87.2	3.6
	100%	92.3 (0.3)	7.4 (0.3)	92.7 (0.4)	6.8 (0.3)	88.0 (0.8)	10.2 (0.0)	79.6 (0.6)	10.1 (4.2)	88.2	8.6

Table 7. Comparing the zero-shot OOD performance for model grafting, LoRA (Hu et al., 2022), and two adapter methods (Pfeiffer et al., 2021; Houlsby et al., 2019) at different sparsity levels. Results are in the 4096-shot setting. The major observations are: (a) At similar sparsity levels ($0.05 - 1\%$), grafting achieves an improvement of 7.9% in OOD performance while being a 1.0% lower in ID accuracy, implying parameter count is not the only reason behind the OOD results for grafting. (b) For grafting, the OOD and ID accuracy change smoothly with the grafting region sparsity.

		ID task: MPQA			ID task: QNLI				
Method	Rank/Sparsity	MPQA	Yelp	SST-2	QNLI	MNLI (0/1)	SNLI (0/1)	Avg. ID	Avg. OOD
LoRA	1 (0.05%)	89.1 (0.7)	79.8 (4.2)	64.7 (5.7)	87.6 (0.3)	57.2 (4.4)	57.9 (4.8)	88.3	64.9
	2 (0.1%)	89.1 (0.4)	74.3 (7.7)	65.5 (5.0)	87.5 (0.4)	54.7 (3.7)	54.0 (4.3)	88.3	62.1
	8 (0.4%)	89.2 (0.7)	83.9 (1.5)	79.6 (3.5)	87.6 (0.3)	64.2 (3.9)	68.5 (4.9)	88.4	74.1
Houlsby et al.	1 (0.025%)	88.6 (0.7)	81.6 (1.8)	74.8 (2.0)	88.1 (0.6)	58.4 (3.8)	57.8 (7.2)	88.4	68.2
	2 (0.05%)	88.7 (0.6)	83.0 (0.9)	77.5 (4.4)	88.0 (0.4)	60.6 (0.7)	62.7 (3.8)	88.4	71.0
	16 (0.4%)	88.1 (0.9)	82.7 (2.3)	77.9 (3.1)	88.4 (0.8)	61.9 (5.3)	61.8 (7.8)	88.3	71.1
	64 (1.6%)	88.1 (0.6)	80.7 (3.3)	75.5 (4.8)	88.7 (0.2)	63.1 (3.2)	63.2 (5.1)	88.4	70.6
Pfeiffer et al.	1 (0.025%)	88.6 (1.0)	77.0 (2.2)	71.6 (2.4)	88.0 (0.7)	56.9 (5.4)	58.3 (8.3)	88.3	66.0
	2 (0.05%)	88.6 (0.9)	77.1 (6.2)	70.0 (7.1)	87.4 (0.4)	61.5 (3.9)	63.8 (5.5)	88.0	68.1
	16 (0.4%)	88.4 (0.5)	80.3 (1.4)	75.9 (2.1)	87.8 (0.4)	66.5 (1.8)	68.1 (5.3)	88.1	71.2
	64 (1.6%)	88.2 (0.9)	78.2 (6.4)	75.4 (6.8)	87.9 (0.3)	66.3 (2.7)	66.7 (7.9)	88.1	71.7
Grafting	0.01%	88.1 (0.4)	83.7 (0.7)	84.6 (0.6)	84.7 (0.6)	71.8 (1.8)	80.1 (2.9)	86.4	80.1
	0.05%	88.8 (0.5)	81.6 (0.6)	83.1 (1.1)	86.0 (0.4)	71.1 (0.5)	79.7 (0.9)	87.4	78.9
	0.1%	88.5 (0.5)	80.8 (0.9)	82.6 (0.5)	86.0 (0.4)	70.9 (1.57)	78.3 (2.2)	87.3	78.2
	1%	88.9 (0.5)	79.8 (0.7)	81.7 (1.3)	86.5 (0.4)	70.6 (2.5)	74.8 (4.7)	87.7	76.7
	100%	88.9 (0.6)	74.2 (2.3)	77.6 (4.9)	88.0 (0.8)	65.7 (2.6)	62.1 (5.7)	88.5	69.9

We observe that at similar sparsity levels, grafting shows a better calibration and OOD performance with a small accuracy drop, compared to the parameter-efficient methods under consideration. This shows that the sparsity of the grafting region is not the only reason behind its better OOD and calibration performance. Other factors like optimization algorithm and the grafting procedure could also be responsible and a deeper exploration of various factors is left for

future work.

Grafts with different sparsity. In Tables 6 and 7, we show the OOD and calibration performance of grafting at different levels of sparsity. They show a smooth decrease with the sparsity of the grafting region, with the grafted model's performance increasing smoothly. Thus for grafting, the calibration error and OOD generalization correlates with number of the size of the graft.

B.4. Overlap with FISH (Sung et al., 2021) mask

Sung et al. (2021) proposed a parameter-efficient technique called Fisher-Induced Sparse uncHanging (FISH) Mask training. The authors select a sparse set of parameters containing the largest Fisher information in the pre-trained model to fine-tune for a specific task and the FISH mask represents the selected parameters. We compute the overlap of our grafting regions with the FISH mask at different sparsity levels in Table 8. If γ_{FISH} denotes the FISH mask, and γ denotes the grafting region given by Equation (3), then overlap score is defined as the intersection over union measure, i.e. $(|\gamma_{FISH} \cap \gamma| / |\gamma_{FISH} \cup \gamma|)$. Furthermore, we report the performances of the grafted models formed with the FISH mask and grafted models with a random mask as a baseline. Our observations can be summarized as follows: (a) There is a minimal overlap ($< 10\%$) between the FISH mask and our grafting regions. (b) Sparse FISH masks do not effectively serve as suitable grafting regions, while the performance of grafted models using FISH masks improves with decreasing sparsity. This shows that full model fine-tuning need not use the top parameters with the largest Fisher information at pre-training.

B.5. Experiments with auto-regressive language modeling

Given the tremendous recent success of auto-regressive models (Brown et al., 2020), we fine-tune a pre-trained GPT-2 (small) model (Radford et al., 2019) on GLUE tasks using prompts from Holtzman et al. (2021). Firstly, we find that skill localization is possible for GPT-2 based fine-tuning as well, albeit requiring denser regions with 0.05% parameters as opposed to 0.01% required by the RoBERTa model; Table 9 summarizes the results. Overall the performance of GPT-2 fine-tuned models is worse than a similarly sized RoBERTa model, which is consistent with prior work. GPT-2 requiring denser regions and having worse generalization is consistent with our connection between sparsity and generalization from Section 4.3.

C. Multi-task and Continual Learning

C.1. Training details for multi-task training.

Each dataset was downsampled to 4096-shot and SGD was used to fine-tune the model on all the tasks (with batches being picked from a randomly chosen dataset), but with 8 times as many iterations. Using same hyperparameter grid as before, we select the model that performs the best on average on the validation sets of each of these tasks.

C.2. Additional Multi-task Experiments

Varying γ_{base}. We conduct further experiments on SGD trained MT models in Figure 11 and Figure 12. In these experiments, we vary the initialization γ_{base} for the task specific graft regions in Equation (3). Increasing the size of the γ_{base} leads to increasing overlap among the task-specific grafts. However, we still observe a similar pattern, albeit dimishing, as Figure 8 for the effect of the task specific grafts on other tasks.

AdamW optimizer. We also conduct experiments on MT models trained with AdamW in Figure 13 and Figure 14. Interestingly, we observe skill localization in the model, without an explicit use of ℓ_1 regularization. This is in stark contrast to single task models (Figures 5a and 5b), where an explicit ℓ_1 regularization was necessary to get sparse grafts.

C.3. Details for Continual Learning

Formally, let task t_n represent the task arrives at order n in the sequence of tasks $t_1, t_2, \cdots, t_n, \cdots$. Suppose we have identified γ_{t_n} as the grafting region for the task t_n from an independent run of the model on the task t_n. Then, when t_n arrives, we train the parameters of the model in the region $\gamma_{t_n} \setminus \cup_{i<n} \gamma_{t_i}$. When we are asked to infer for task t_n, we construct the grafted model using $\cup_{i \leq n} \gamma_{t_i}$.

As a simple experiment, we train a model on QNLI, AG news, and SST-2, with each dataset represented by a 4096-shot training set, in sequential order. After the model has been trained on SST-2, we check the model's performance on QNLI and AG news. We compare the performance of the model when trained with the SGD algorithm naively, and when it is trained with our continual learning algorithm in Table 4. We observe that our proposed continual method can retain the performance on QNLI, while having a minimal effect on the performance of other tasks.

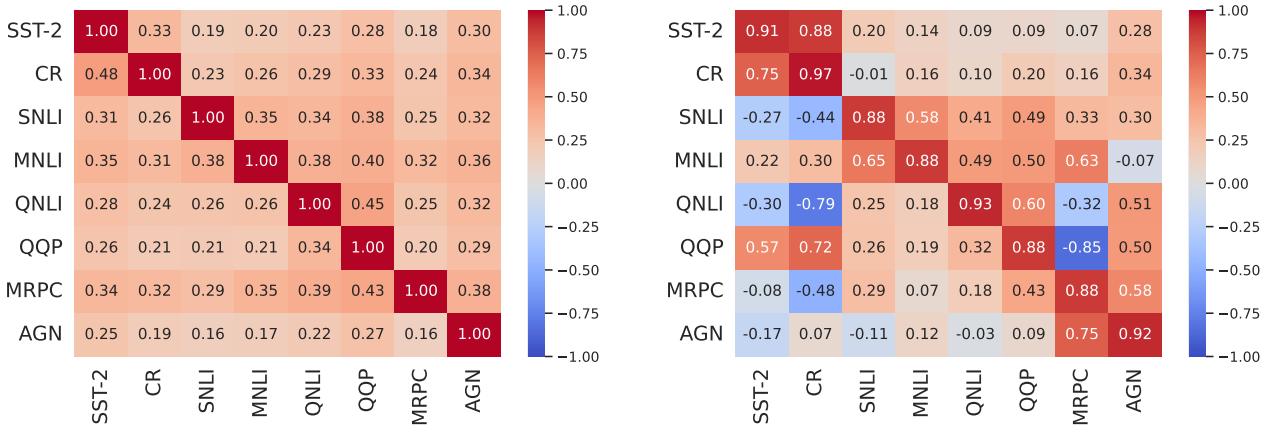
The main advantage of our proposed method is that we can retain performance by memorizing the grafting region and the arrival sequence number for each task. This method does not lead to a blowup of the memory required, since each parameter in the memory can be indexed by the sequence number of the task it belongs to. However, the major disadvantage of this method is that we need to create the grafted

Table 8. Measuring overlap between FISH mask (Sung et al., 2021) and model grafting at different sparsity levels. If γ_{FISH} denotes the FISH mask, and γ denotes our grafting region (computed using with Equation (3)), then overlap is defined by $(|\gamma_{FISH} \cap \gamma| / |\gamma_{FISH} \cup \gamma|)$. Furthermore, we report the accuracy (†F1) in % for our grafted models, grafted models with FISH mask, and grafted models with random grafting regions. Our observations are: (a) overlap is $< 10\%$ between the FISH mask and our grafting regio, and(b) There is an average 10% gap between the performance of our grafted models and grafted models with FISH mask at 0.01% sparsity, which reduces with decreasing sparsity levels.

Dataset	Sparsity	Overlap (in %)	Graft		Graft with FISH mask		Random Graft	
			Acc/F1	ECE	Acc/F1	ECE	Acc/F1	ECE
SST-2	0.01%	9.2 (1.3)	92.4 (0.1)	3.1 (0.4)	90.8 (0.3)	8.5 (0.6)	77.8 (0.1)	4.7 (0.1)
	0.1%	4.1 (1.7)	92.5 (0.4)	4.4 (0.7)	91.5 (0.3)	8.1 (0.4)	77.8 (0.2)	4.8 (0.4)
	1%	1.9 (0.5)	92.3 (0.3)	4.6 (0.7)	91.6 (0.2)	8.0 (0.5)	78.7 (0.5)	4.5 (0.5)
AGNews	0.01%	7.6 (3.3)	91.1 (0.9)	0.9 (0.2)	80.3 (5.8)	5.1 (1.6)	65.8 (0.1)	6.8 (0.1)
	0.1%	1.8 (0.2)	90.8 (0.1)	1.1 (0.3)	82.4 (4.8)	5.1 (2.7)	66.0 (0.2)	6.7 (0.1)
	1%	1.1 (0.1)	91.0 (0.3)	1.2 (0.2)	71.8 (13.2)	7.9 (6.0)	67.8 (0.7)	6.7 (0.3)
QNLI	0.01%	8.1 (3.1)	84.7 (0.6)	1.0 (0.3)	65.1 (8.9)	22.5 (11.8)	50.7 (0.0)	34.5 (0.0)
	0.1%	4.8 (2.0)	86.0 (0.4)	2.2 (0.9)	81.3 (5.4)	12.7 (3.5)	50.7 (0.1)	34.5 (0.2)
	1%	1.9 (0.6)	86.5 (0.4)	3.1 (1.1)	78.9 (12.2)	16.6 (10.4)	50.7 (0.1)	35.2 (0.3)
QQP†	0.01%	11.4 (5.9)	76.3 (0.4)	3.5 (0.7)	57.0 (10.5)	16.9 (4.9)	51.6 (0.0)	36.2 (0.0)
	0.1%	6.6 (4.7)	78.2 (0.0)	4.5 (1.3)	73.8 (1.2)	14.3 (1.0)	51.6 (0.1)	36.1 (0.2)
	1%	2.3 (1.1)	78.7 (0.2)	5.4 (1.7)	75.9 (0.2)	14.1 (0.6)	51.8 (0.2)	35.6 (0.2)

Table 9. We fine-tune GPT-2 models (small) (Radford et al., 2019) on different downstream tasks using prompts from (Holtzman et al., 2021). FT on GPT-2 (small) performs worse than RoBERTa-base in both 64-shot and 4096-shot settings. In fact, we observe that FT gives close to random guessing performance on most two sentence tasks in the 64-shot setting. For each downstream task, we learn a grafting region γ using our optimization procedure in Equation (3). The grafting regions for all tasks have sparsity at most 0.05% (< 42500 parameters). We report test accuracy (†F1) and the calibration error using ECE of the fine-tuned model and the grafted model for each task. The main findings are (1) The grafted model can retrieve $> 95\%$ of the FT accuracy while being better calibrated than the original model itself. For single-sentence tasks (4096-shot) the grafted model shows only a 1.2% drop in accuracy but an improvement of 3.2% in the calibration error. Similarly for two-sentence tasks, the grafted model shows a 3.4% drop in accuracy with an improvement of 7.3% in the calibration error.

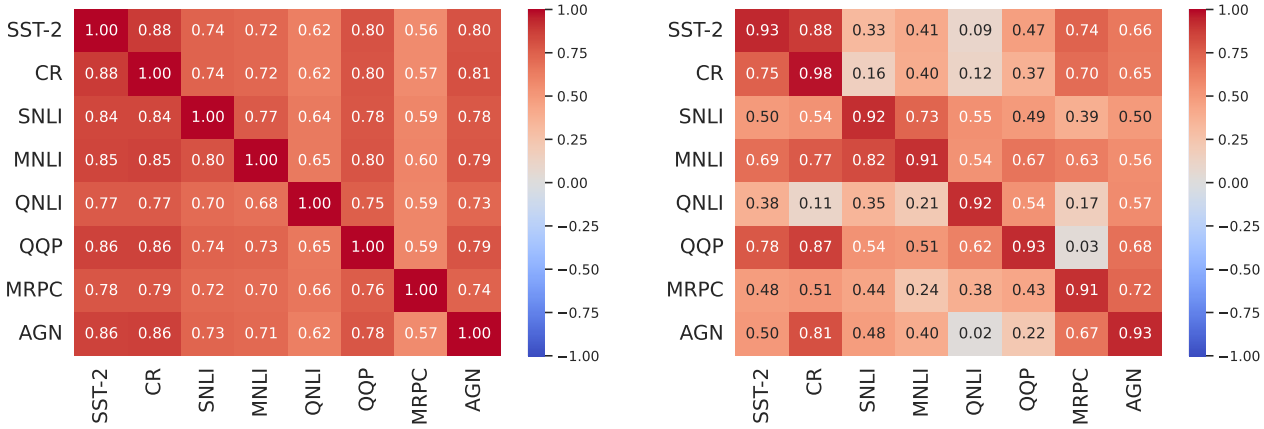
	64-shot				4096-shot				
	FT		Graft		FT		Graft		
Dataset	Acc.	ECE	Acc.	ECE	Acc.	ECE	Acc.	ECE	Agreement
Single sentence tasks									
SST-2	85.0 (0.9)	11.8 (2.0)	84.8 (0.8)	7.3 (2.1)	90.6 (0.6)	5.8 (1.7)	88.8 (0.5)	3.4 (0.8)	93.6 (0.9)
CR	87.4 (1.0)	11.1 (1.1)	87.5 (0.8)	7.4 (0.9)	87.3 (1.5)	11.2 (1.3)	88.5 (0.6)	5.4 (1.6)	92.8 (1.6)
MR	80.2 (2.0)	17.0 (2.0)	80.5 (1.6)	11.0 (1.4)	86.7 (0.2)	6.0 (2.2)	86.0 (0.4)	2.0 (0.2)	91.6 (1.0)
MPQA	85.2 (1.2)	11.9 (1.3)	85.5 (0.5)	6.7 (1.9)	88.4 (0.6)	7.6 (2.1)	88.1 (0.2)	3.7 (1.4)	93.7 (1.3)
AGNews	87.2 (0.8)	9.6 (0.5)	84.3 (0.8)	6.0 (0.8)	92.2 (0.4)	2.8 (0.9)	88.6 (0.9)	1.6 (0.3)	91.9 (0.7)
Subj	89.5 (1.1)	7.5 (0.4)	85.0 (2.2)	2.8 (1.3)	95.8 (0.2)	3.3 (0.3)	93.5 (0.3)	1.5 (0.6)	95.2 (0.5)
Two sentence tasks									
QNLI	53.9 (3.3)	30.5 (13.0)	54.1 (3.2)	18.4 (7.3)	81.9 (0.3)	6.1 (4.8)	79.8 (0.2)	2.5 (0.9)	87.8 (1.6)
SNLI	56.0 (3.2)	33.4 (4.3)	50.4 (3.1)	23.3 (4.4)	80.7 (0.6)	8.2 (2.7)	76.3 (0.5)	1.7 (0.7)	84.0 (1.2)
MNLI	45.1 (3.5)	41.3 (2.1)	42.2 (5.0)	30.7 (3.1)	72.3 (0.8)	12.2 (4.5)	67.6 (0.6)	4.3 (2.6)	80.1 (1.2)
RTE	51.8 (3.5)	36.8 (4.5)	51.9 (2.9)	23.8 (2.5)	66.7 (1.5)	27.3 (2.3)	65.7 (1.5)	16.8 (1.8)	79.6 (2.9)
MRPC†	81.1 (0.2)	13.2 (7.5)	78.5 (1.7)	6.3 (3.1)	85.2 (0.5)	15.8 (3.5)	80.3 (1.5)	8.3 (2.2)	82.9 (2.2)
QQP†	56.8 (0.7)	25.6 (7.9)	54.6 (1.8)	16.1 (5.9)	76.1 (0.2)	13.5 (2.4)	73.0 (0.5)	5.9 (1.9)	87.4 (1.1)



(a) Overlap of task-specific grafting regions

(b) Effect of task-specific grafted model on other tasks

Figure 11. Ablations on task-specific grafting region γ_i for task i, learned by optimizing $\mathcal{L}_i$ on the MT model, trained with SGD. Here, we learn the task specific grafting regions $\gamma_{\mathcal{T}}$ by solving $\mathcal{L}_{\mathcal{T}}$ on the MT model with γ_{base} set to the top-10^{-5} moving parameters in the model. In all figures, we evaluate the effect of graft region of task in row i on the task in row j. Figure (a) measures the assymteric overlap in the regions defined as $\frac{|\gamma_i \cap \gamma_j|}{|\gamma_j|}$ for tasks in row i and column j. Figure (b) evaluates the relative accuracy gain of task in column j using the graft regions of task in row i; refer to Equation (5) for the precise expression. **Observations:** We have similar observations as Figure 8. However, we have an increasing amount of intersection between the task-specific graftings, as γ_{base} is common among all the tasks for optimization.



(a) Overlap of task-specific grafting regions

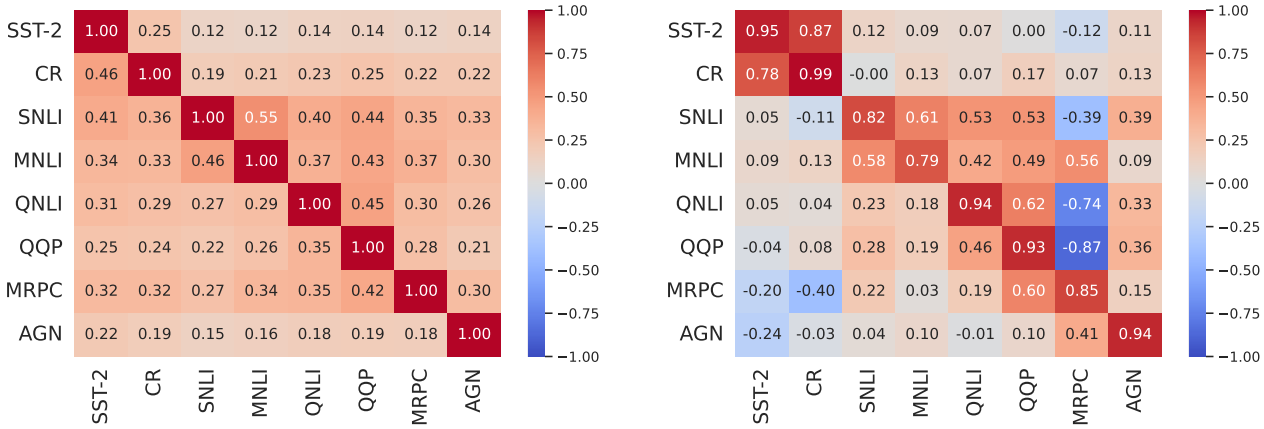
(b) Effect of task-specific grafted model on other tasks

Figure 12. Ablations on task-specific grafting region γ_i for task i, learned by optimizing $\mathcal{L}_i$ on the MT model, trained with SGD. Here, we learn the task specific grafting regions $\gamma_{\mathcal{T}}$ by solving $\mathcal{L}_{\mathcal{T}}$ on the MT model with γ_{base} set to the top-10^{-4} moving parameters in the model. In all figures, we evaluate the effect of graft region of task in row i on the task in row j. Figure (a) measures the assymteric overlap in the regions defined as $\frac{|\gamma_i \cap \gamma_j|}{|\gamma_j|}$ for tasks in row i and column j. Figure (b) evaluates the relative accuracy gain of task in column j using the graft regions of task in row i; refer to Equation (5) for the precise expression. **Observations:** We have similar observations as Figure 8. However, we have an increasing amount of intersection between the task-specific graftings, as γ_{base} is common among all the tasks for optimization.

model for a task when we need to infer for the particular task. Thus, an important question that we aim to explore for future work is whether we can use the concept of grafting to get a single model that does well in continual learning.

D. Discussions on Generalization Theory

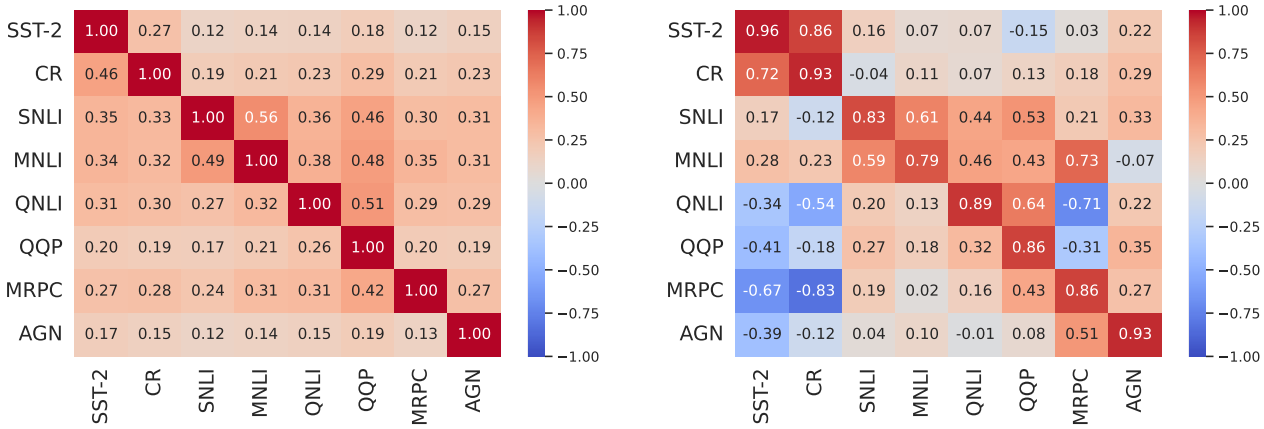
Classical tools like uniform convergence (UC) seemingly fail to explain generalization in deep learning (Nagarajan &



(a) Overlap of task-specific grafting regions

(b) Effect of task-specific grafted model on other tasks

Figure 13. Ablations on task-specific grafting region $\boldsymbol{\gamma}_i$ for task i, learned by optimizing $\mathcal{L}_i$ on the MT model, trained with AdamW. Here, we learn the task specific grafting regions $\boldsymbol{\gamma}_{\mathcal{T}}$ by solving $\mathcal{L}_{\mathcal{T}}$ on the MT model with $\boldsymbol{\gamma}_{\text{base}}$ set to 0. In all figures, we evaluate the effect of graft region of task in row i on the task in row j. Figure (a) measures the assymteric overlap in the regions defined as $\frac{|\boldsymbol{\gamma}_i \cap \boldsymbol{\gamma}_j|}{|\boldsymbol{\gamma}_j|}$ for tasks in row i and column j. Figure (b) evaluates the relative accuracy gain of task in column j using the graft regions of task in row i; refer to Equation (5) for the precise expression. **Observations:** Interestingly, we observe that AdamW shows better skill localization in the MT model, as opposed to task-specific models (Figures 5a and 5b).



(a) Overlap of task-specific grafting regions

(b) Effect of task-specific grafted model on other tasks

Figure 14. Ablations on task-specific grafting region $\boldsymbol{\gamma}_i$ for task i, learned by optimizing $\mathcal{L}_i$ on the MT model, trained with AdamW with an explicit 0.001-ℓ_1 regularization on the movement of the parameters during fine-tuning. Here, we learn the task specific grafting regions $\boldsymbol{\gamma}_{\mathcal{T}}$ by solving $\mathcal{L}_{\mathcal{T}}$ on the MT model with $\boldsymbol{\gamma}_{\text{base}}$ set to 0. In all figures, we evaluate the effect of graft region of task in row i on the task in row j. Figure (a) measures the assymteric overlap in the regions defined as $\frac{|\boldsymbol{\gamma}_i \cap \boldsymbol{\gamma}_j|}{|\boldsymbol{\gamma}_j|}$ for tasks in row i and column j. Figure (b) evaluates the relative accuracy gain of task in column j using the graft regions of task in row i; refer to Equation (5) for the precise expression. **Observations:** We find that adding an explicit ℓ_1 regularization doesn't affect or improve the skill localization of AdamW in the MT model, observed in Figure 13.

Kolter, 2019) since they bound the generalization error[3] as

$$\mathcal{L}_{\text{test}}(\boldsymbol{\theta}_{\text{ft}}) \lesssim \inf_{\boldsymbol{\theta}^\star \in \Theta} \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^\star) + \sup_{\boldsymbol{\theta} \in \Theta} |\mathcal{L}_{\text{test}}(\boldsymbol{\theta}) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta})|$$

[3]This is a rough picture just to present the high-level idea.

The second term is further upper bounded by $\frac{\mathcal{C}(\Theta)}{\sqrt{n}}$ where $\mathcal{C}(\Theta)$ denotes a complexity measure for the class (e.g. Rademacher complexity). This approach necessarily fails for vanilla fine-tuning because the class Θ is very large (100M of parameters) and expressive enough to perfectly fit the training set with a large train-test gap, thus giving

vacuous bounds – fine-tuning on QNLI in the 4096-shot setting get 100% training and $\sim 88\%$ test accuracy, making the second term $\geq 12\%$..

One way to get around this vacuous-ness of UC is to use properties of the training algorithm (here fine-tuning) to transform the learned model into a "surrogate model" that belongs to a much simpler class of functions, for which uniform convergence is more like to succeed, e.g. low rank compressions (Arora et al., 2018), denoising classifiers (Negrea et al., 2020). The existence of sparse graft models with fine-tuning precisely gives us a surrogate model class that allows for a tighter UC-based bound instead. For a grafting region γ the relevant UC bound will looks as follows:

$$\sup_{\theta \in \Theta} \left| \mathcal{L}_{\text{test}}(\overline{\theta}(\gamma)) - \mathcal{L}_{\text{test}}(\overline{\theta}(\gamma)) \right|$$

The sparsity of the region γ can be further leveraged to upper bound this UC term by $\widetilde{\mathcal{O}}\left(\sqrt{\frac{s}{n}}\right)$ using standard arguments, where $s = \|\gamma\|_0$, n is the training set size and $\widetilde{\mathcal{O}}$ hides logarithmic factors. Besides the theoretical benefit of the sparsity, we also empirically find that the train-test gap of the graft model using just ~ 5000 parameters is small ($\sim 1\%$) since a sparse graft model is unable to perfectly fit the training sets even in the 4096-shot setting. Crucially this small train-test gap comes at a very small impact on the expressivity, i.e. there exist sparse grafting models that can do very well on the task. Leveraging these observations, we show a generalization bound for a sparse re-training based procedure inspired by model grafting. In fact, our results hold generally for any parameter efficient fine-tuning method that only updates a small subset of parameters, e.g. BitFit (Ben Zaken et al., 2022).

D.1. Generalization Bound for Graft Re-training and Parameter-Efficient FT

We first clarify some notations. Let $\overline{\theta}_{\text{ft}}(\gamma)$ denote the graft model learned after the fine-tuning procedure, $\theta(\gamma)$ to denote the model re-trained on n samples by only updating the parameters in γ, $\theta^*(\gamma)$ to denote the model with best test performance while only changing the parameters in γ, and θ^* to denote the model with the best test performance. For simplicity, we also assume that the loss $\mathcal{L}$ is bounded by $[0, 1]$. To bound the complexity of the function class, we assume that each parameter can only take the value from q numbers, and the quantized network has performance close to the original network. This assumption is very close to the definition of compressible classifier in Arora et al. (2018). Formally, we have

Assumption D.1. For any model θ, we define $\bar{q}(\theta)$ to be the model that quantized every parameter into the q given values. Then there exist $\varepsilon_1 > 0$ such that for all θ and any data point x, we have

$$|\mathcal{L}(x; \bar{q}(\theta)) - \mathcal{L}(x; \theta)| \leq \varepsilon_1.$$

Remark. We only need this condition for quantization of the parameters in γ and not necessarily all parameters. Furthermore, fixed precision floating point numbers are anyway quantized in practice, with $q = 2^{32}$ for 32-bit floats.

We denote Θ_n to be the number of different regions γ that can be found from Equation (3) when trained on different training datasets from the same task. While Θ_n can be trivially bounded by d^s where d is the parameters in the model and $s = \|\gamma\|_0$, empirically Θ_n should be small ($\log \Theta_n \ll s$) since the regions γ we find for different training sets of the same task have large overlap. Additionally, for the case of parameter-efficient fine-tuning algorithms where a fixed set of parameters are chosen to be updated (e.g. BitFit (Ben Zaken et al., 2022)), this term Θ_n is 1.

Then we have the following theorem for the generalization bound of the re-trained graft model $\theta(\gamma)$ on n samples.

Theorem D.2. *Under Assumption D.1, we have*

$$\begin{aligned}
&\mathcal{L}_{test}(\theta(\gamma)) - \mathcal{L}_{test}(\theta^*) \\
&\leq \underbrace{2 \sup_{\widetilde{\theta}} \left| \mathcal{L}_{test}(\bar{q}(\widetilde{\theta}(\gamma))) - \mathcal{L}_{train}(\bar{q}(\widetilde{\theta}(\gamma))) \right|}_{\text{variance term}} \\
&+ \underbrace{\mathcal{L}_{test}(\theta^*(\gamma)) - \mathcal{L}_{test}(\theta^*)}_{\text{bias term}} + \underbrace{4\varepsilon_1}_{\text{quantization error}}.
\end{aligned}$$

Moreover, with probability at least $1 - \delta$, the variance term can be bounded by

$$\begin{aligned}
&2 \sup_{\widetilde{\theta}} \left| \mathcal{L}_{test}(\bar{q}(\widetilde{\theta}(\gamma))) - \mathcal{L}_{train}(\bar{q}(\widetilde{\theta}(\gamma))) \right| \\
&\leq 2\sqrt{\frac{s \log q + \log \Theta_n + \log(1/\delta)}{n}}.
\end{aligned}$$

The first term captures the variance of the re-trained graft model, which can be further bounded by $\widetilde{O}(\sqrt{s/n})$ in our setting using uniform convergence over all possible grafting regions γ, since the total number of the grafted region can be bounded as $\Theta_n \leq d^s$. Thus the sparser the graft region, the more sample efficient fine-tuning will be. Empirically, because the (re-trained) grafting model has a very small train-test gap in practice, we know that the term $\mathcal{L}_{\text{test}}(\theta(\gamma)) - \mathcal{L}_{\text{train}}(\theta(\gamma))$ is small. Note that $\theta(\gamma)$ is the re-trained model on the grafted region γ using the training data, so so one would expect the model $\theta(\gamma)$ to have nearly the largest train-test gap compared to all grafted models. This is because, intuitively, the solution found by ERM should not have better generalization compared to other solutions. This leads us to believe that for all models on grafted regions $\widetilde{\theta}(\gamma)$, the train-test gap is also small, which means that the variance term $\sup_{\widetilde{\theta}} \left| \mathcal{L}_{\text{test}}(\bar{q}(\widetilde{\theta}(\gamma))) - \mathcal{L}_{\text{train}}(\bar{q}(\widetilde{\theta}(\gamma))) \right|$ should be small empirically. In practice, the train-test gap for re-trained models is of the order of 1%.

The second term $(\mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*))$ actually characterizes the capacity of the grafted region $\boldsymbol{\gamma}$. If we let the sparsity level goes to 1 (the full model), this term becomes 0 since it is as expressive as the whole model, and if the graft $\boldsymbol{\gamma}$ is too sparse, this term may be large. Thus if there exists a good localized way to do well on the given task, we expect this term to be small.

Remark. It is important to note that setting $\boldsymbol{\gamma}$ to be the all ones vector in the above result, i.e. including all parameters, will reduce to the naive uniform convergence bound for fine-tuning the entire network. In this case $s = d$ (all parameters), and thus the bias term is exactly 0. However the variance term, in practice and in theory, is far from being small because fine-tuning the entire network can fit the training set perfectly, leading to the variance term being at least as large as $\sim 10\%$. This shows the utility of using model grafting as a "surrogate" to show much tighter generalization guarantees.

Theorem D.2 gives an end-to-end bound for the re-training of the grafted model $\boldsymbol{\theta}(\boldsymbol{\gamma})$, and also provides us with the freedom to balance the bias and variance. In fact this bound also works for any parameter-efficient fine-tuning regime like BitFit. However the sparsity level of model grafting is much lower than previous parameter efficient methods.

D.2. Generalization Bound for Grafted Models

The previous discussion works for re-training a graft model $\boldsymbol{\theta}(\boldsymbol{\gamma})$. In order to get a generalization bound for the graft model $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$ without re-training (the graft model obtained directly after the fine-tuning), we need to add the following assumption.

Assumption D.3. We assume that the graft model $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$ is nearly ERM on n training samples among all grafted models, i.e., there exist $\varepsilon_2 > 0$ such that

$$\mathcal{L}_{\text{train}}(\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})) \leq \mathcal{L}_{\text{train}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) + \varepsilon_2.$$

Although this assumption seems a little bit strong at first glance, it is actually validated by our empirical findings. If we further have Assumption D.3, we can directly have the generalization bound for the grafted model $\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})$ from Theorem D.2.

Theorem D.4. *Under Assumption D.1 and Assumption D.3, we have*

$$\begin{aligned}
&\mathcal{L}_{test}(\overline{\boldsymbol{\theta}_{ft}}(\boldsymbol{\gamma})) - \mathcal{L}_{test}(\boldsymbol{\theta}^*) \\
&\leq \underbrace{2\sup_{\widetilde{\boldsymbol{\theta}}} \left|\mathcal{L}_{test}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma}))) - \mathcal{L}_{train}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma})))\right|}_{\textit{variance term}} \\
&+ \underbrace{\mathcal{L}_{test}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})) - \mathcal{L}_{test}(\boldsymbol{\theta}^*)}_{\textit{bias term}} + \underbrace{4\varepsilon_1 + \varepsilon_2}_{\textit{error term}}.
\end{aligned}$$

Moreover, with probability at least $1-\delta$, the first term can be bounded by

$$\begin{aligned}
&2\sup_{\widetilde{\boldsymbol{\theta}}} \left|\mathcal{L}_{test}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma}))) - \mathcal{L}_{train}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma})))\right| \\
&\leq 2\sqrt{\frac{s\log q + \log \Theta_n + \log(2/\delta)}{n}}.
\end{aligned}$$

D.3. Proof of Theorem D.2

First, we can decompose the term we interested $\mathcal{L}_{\text{test}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*)$ into seven terms.

$$\begin{aligned}
&\mathcal{L}_{\text{test}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*) \\
&\leq (\mathcal{L}_{\text{test}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma})))) \\
&+ (\mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma})))) \\
&+ (\mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta}(\boldsymbol{\gamma}))) \\
&+ (\mathcal{L}_{\text{train}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))) \\
&+ (\mathcal{L}_{\text{train}}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})) - \mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})))) \\
&+ (\mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})))) \\
&+ (\mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*)).
\end{aligned}$$

Now we bound each term one by one. For the first term $\mathcal{L}_{\text{test}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma})))$, it is bounded by ε_1 because of Assumption D.1. Similarly, the third term $\mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta}(\boldsymbol{\gamma}))$ is also bounded by ε_1. As for the fourth term $\mathcal{L}_{\text{train}}(\boldsymbol{\theta}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{train}}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))$, it is at most 0 since $\boldsymbol{\theta}(\boldsymbol{\gamma})$ is the re-trained model by only updating the parameters in the region $\boldsymbol{\gamma}$ on the samples. As for the fifth term $(\mathcal{L}_{\text{train}}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})) - \mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))))$ and the seventh term $(\mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*))$, they are also bounded by ε_1 according to Assumption D.1.

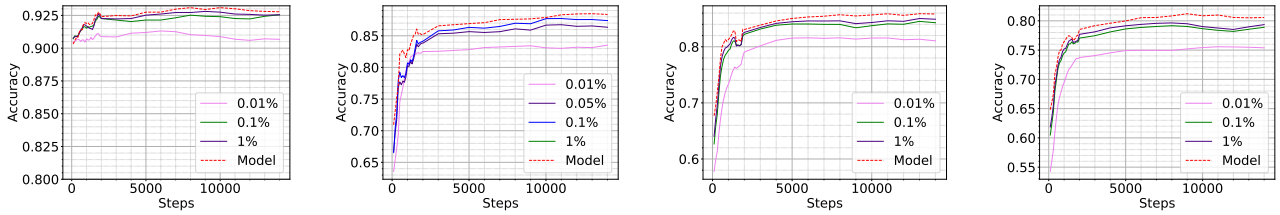
For the second term $(\mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma}))))$ and the sixth term $(\mathcal{L}_{\text{train}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{test}}(\bar{q}(\boldsymbol{\theta}^*(\boldsymbol{\gamma}))))$, both of them can be bounded by

$$\sup_{\widetilde{\boldsymbol{\theta}}} \left|\mathcal{L}_{\text{test}}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma})))\right|.$$

Thus, we show that under Assumption D.1, we have

$$\begin{aligned}
&\mathcal{L}_{\text{test}}(\overline{\boldsymbol{\theta}_{\text{ft}}}(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*) \\
&\leq \underbrace{2\sup_{\widetilde{\boldsymbol{\theta}}} \left|\mathcal{L}_{\text{test}}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma})))\right|}_{\text{variance term}} \\
&+ \underbrace{\mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*(\boldsymbol{\gamma})) - \mathcal{L}_{\text{test}}(\boldsymbol{\theta}^*)}_{\text{bias term}} + \underbrace{4\varepsilon_1 + \varepsilon_2}_{\text{error term}}.
\end{aligned}$$

Now we bound the variance term. Note that this can be bounded by standard generalization theory: because the



(a) SST-2, 4096-shot. Checkpoint selected at step 500.

(b) QNLI, 4096-shot. Checkpoint selected at step 2000.

(c) SNLI, 4096-shot. Checkpoint selected at step 3000.

(d) MNLI, 4096-shot. Checkpoint selected at step 2000.

Figure 15. We track the evolution of the grafting regions of various sizes of an intermediate checkpoint over time. We created grafted models with the model checkpoints at every step, using the grafting regions. We observe that the grafting region tracks the model's performance over training before saturating.

number of possible models for $\bar{q}(\boldsymbol{\theta}(\boldsymbol{\gamma}))$ is bounded by $\Theta_n \cdot q^s$, after applying for the union bound, we have with probability at least $1-\delta$,

$$\sup_{\widetilde{\boldsymbol{\theta}}} \left| \mathcal{L}_{\text{test}}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma}))) - \mathcal{L}_{\text{train}}(\bar{q}(\widetilde{\boldsymbol{\theta}}(\boldsymbol{\gamma}))) \right| \leq \sqrt{\frac{s \log q + \log \Theta_n + \log(1/\delta)}{n}},$$

and we complete the proof of Theorem D.2.

D.4. Core skills are learned first.

We track the accuracy of grafted models of varying sparsity as fine-tuning progresses in Figure 15. To do so, we select an intermediate checkpoint to compute grafting regions at different sparsity levels that can match the labels of the model rather than the ground-truth labels. For each run, we selected the checkpoint, where the train and test accuracy plots seemed to diverge, implying the phase where the model starts to overfit on the training dataset. We construct grafted model at each model checkpoint using the pre-computed grafting regions and track their performance. As we can observe, the grafting region tracks the model's performance over training before saturating earlier. This aligns with the observations of Kalimeris et al. (2019) that SGD learns simple functions first.