LESS IS MORE: UNDERTRAINING EXPERTS IMPROVES MODEL UPCYCLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern deep learning is increasingly characterized by the use of open-weight foundation models that can be fine-tuned on specialized datasets. This has led to a proliferation of expert models and adapters, often shared via platforms like HuggingFace and AdapterHub. To leverage these resources, numerous model upcycling methods have emerged, enabling the reuse of fine-tuned models in multi-task systems. A natural pipeline has thus formed to harness the benefits of transfer learning and amortize sunk training costs: models are pre-trained on general data, fine-tuned on specific tasks, and then upcycled into more general-purpose systems. A prevailing assumption is that improvements at one stage of this pipeline propagate downstream, leading to gains at subsequent steps. In this work, we challenge that assumption by examining how expert fine-tuning affects model upcycling. We show that long fine-tuning of experts that optimizes for their individual performance leads to degraded merging performance, both for fully fine-tuned and LoRA-adapted models, and to worse downstream results when LoRA adapters are upcycled into MoE layers. We trace this degradation to the memorization of a small set of difficult examples that dominate late fine-tuning steps and are subsequently forgotten during merging. Finally, we demonstrate that a task-dependent aggressive early stopping strategy can significantly improve upcycling performance.

1 INTRODUCTION

The rise of open-weight foundation models, such as CLIP (Radford et al., 2021; Ilharco et al., 2021), T5 (Raffel et al., 2020) and the more recent Gemma (Team, 2025), Llama (Grattafiori et al., 2024) and DeepSeek (DeepSeek-AI, 2024), has caused a paradigm shift in the field of machine learning. Instead of training a model from scratch as was previously the norm, it is now increasingly common for practitioners and researchers alike to start with a pre-trained foundation model and then fine-tune it on a task of interest (Stanford-CRFM, 2021). This approach leverages the benefits of transfer-learning, leading to performance and robustness gains. The proposal of multiple parameter-efficient fine-tuning (PEFT) methods (Hu et al., 2022; Liu et al., 2022), which reduce the computational costs of fine-tuning and limit catastrophic forgetting by only updating a subset of the model parameters, further enables this approach. This has lead to a proliferation of different versions of these foundation models and of PEFT adapters, fine-tuned on a variety of downstream tasks, which are openly accessible on public model repositories such as Hugging Face (Wolf et al., 2019) and Adapter Hub (Pfeiffer et al., 2020).

Model *upcycling*, the practice of reusing existing models to create new, more capable deep learning systems (Zhang et al., 2024; He et al., 2024), capitalizes on this proliferation of fine-tuned models and adapters. Two upcycling strategies stand out: *model merging*, and *model MoErging*. Model merging methods combine multiple fine-tuned versions of the same foundational model into one, preserving the size and therefore the computational and memory requirements of the original pre-trained model while infusing it with multiple new capabilities (Matena & Raffel, 2022; Jin et al., 2023; Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024; Davari & Belilovsky, 2024). The advent of model merging techniques and open-source libraries for merging (Kandpal et al., 2023; Goddard et al., 2024) has had an important impact on the deep learning community, providing a simple, training-free way to create better models from already existing checkpoints and adapters.

In the past year, many of the top performing models on HuggingFace's Open LLM Leaderboard (Beeching et al., 2023) have resulted from the merging of fine-tuned checkpoints (Yu et al., 2024).

Model MoErging (Yadav et al., 2024) similarly combines multiple adapted experts, but instead of fusing the parameters directly, MoErging approaches such as Ostapenko et al. (2024); Muqeeth et al. (2024) combine adapters into modular, mixture-of-experts (MoE) type layers (Shazeer et al., 2017) expanding the model's size and capabilities. A routing mechanism determines which input, or part of the input, gets processed by which expert modules. For this upcycling strategy further training is often required to let the router and expert adapters learn how to interact with one another.

A natural pipeline has therefore emerged to leverage the benefits of transfer-learning and amortize past sunk training costs: large models are *pre-trained* in an unsupervised fashion on large amounts of general, unlabeled data; these foundational models are then *fine-tuned*, potentially using PEFT techniques, on specialized datasets or tasks; finally these fine-tuned expert checkpoints or adapters are *upcycled* and combined to create more capable, often multi-task models.

A common assumption is that *increased performance at one stage of this pipeline will propagate downstream*. In other words, a stronger pre-trained model should yield a stronger fine-tuned model, and similarly, stronger fine-tuned experts should produce a stronger merged / MoErged model. We challenge this assumption in this work by studying the following questions: *How does expert training affect upcycling?* and *Do all capabilities and knowledge transfer equally well?*

We find that long fine-tuning that optimizes for expert performance can substantially hurt model upcycling, a phenomenon to which we refer as "overtraining" in the context of this paper. While overtrained experts might be better on their respective fine-tuning tasks, they lead to worse performance when merged or when used as initializations for model MoErging. We validate this phenomenon across diverse settings, including merging fully fine-tuned and PEFT models, performing MoErging with LoRA adapters, in both vision and language domains and across different model sizes. Additionally, we identify what type of knowledge gets preserved during model merging. We find that easy examples are correctly classified by merged models while harder data points are overwhelmingly forgotten during the merging process. While some recent work has hinted that undertraining experts can benefit merging performance (Pari et al., 2024; Zhou et al., 2025), our work provides a systematic analysis of this phenomenon, and demonstrates how a simple early stopping strategy can significantly improve the efficacy of existing merging and MoErging techniques. Our research introduces a critical new dimension to model upcycling, showing how careful expert training, and targeted checkpoint release can unlock improved performance.

Concretely, our contributions are the following:

- We show that overtraining full fine-tuned (FFT) models produces sub-optimal merges (Section 3.1), and that the negative impact is even stronger when using LoRA adapters for parameter-efficient fine-tuning (Section 3.2);
- We explain this phenomenon through the lens of data difficulty in Section 4, showing that later training steps are primarily guided by the loss of a small fraction of difficult examples which are predominantly forgotten when merging.
- We show that for model MoErging, overtraining the constituent experts leads to lower final accuracy after further multi-task training of the modular model (Section 3.3).
- We show that a task-dependent training time of experts can bring a further boost in upcycling performance. We propose a simple early stopping strategy that favors expert undertraining. This strategy effectively adapts the training duration for each task, and can recover optimal upcycling accuracy (Section 5).

2 Preliminaries and Methodology

2.1 Model merging

Model merging has recently gained a lot of popularity as a means to combine the abilities of multiple fine-tuned versions of the same pre-trained model into one, preserving the model architecture and size. Formally, a model merging method, $Merge$, takes the parameters θ_0 of the pre-trained foundation model, and parameters $\{\theta_t\}_{t\in\mathcal{T}}$ of the multiple *experts*, which are fine-tuned models on each task t from a set $\mathcal{T}$, and outputs the parameters of the merged model $\bar{\theta} = Merge(\theta_0, \{\theta_t\}_{t\in\mathcal{T}})$. A simple example of this combination step is averaging the different fine-tuned models' parameters:

$$\bar{\theta} = \frac{1}{|\mathcal{T}|} \sum_{t\in\mathcal{T}} \theta_t. \tag{1}$$

A common challenge in model merging is the observed performance degradation of the merged model $\bar{\theta}$ on individual tasks $t \in \mathcal{T}$, relative to the original fine-tuned model θ_t. This phenomenon has been coined "interference", and a plethora of merging methods have been proposed to reduce interference when merging models and to preserve as much of the accuracy of the expert models as possible (Matena & Raffel, 2022; Jin et al., 2023; Yadav et al., 2023; Yu et al., 2024; Deep et al., 2024; Davari & Belilovsky, 2024). These methods have mainly focused on modifying the experts parameters $\{\theta_t\}_{t\in\mathcal{T}}$ or the respective *task vectors* $\{\tau_t\}_{t\in\mathcal{T}}$, where $\tau_t = \theta_t - \theta_0$, and / or changing the combination step. We consider 4 popular merging methods:

- **Average** simply averages the parameters of all fine-tuned models following Equation (1);
- **Task Arithmetic (TA)** (Ilharco et al., 2023) scales the sum of the task vectors by a tuned scalar λ, and adds it to the pre-trained model parameters, returning $\theta_0 + \lambda \sum_{t\in\mathcal{T}} \tau_t$;
- **TIES** (Yadav et al., 2023) prunes low magnitude parameters from each task vector, then only averages the parameters from each sparse task vector that have the same sign as the weighted majority;
- **DARE** (Yu et al., 2024) randomly prunes a fraction of each task vector parameters; the remaining sparse task vectors are then rescaled based on the pruning fraction, and are combined as in the TA method.

2.2 Model MoErging

Another popular class of upcycling strategies besides model merging are model MoErging techniques. MoErging methods aggregate multiple fine-tuned experts with the use of modular architectures to build stronger deep learning systems. The large design space of these methods, paired with their effectiveness has led to the rapid development of many new methods in the recent past (Yadav et al., 2024). A key feature of MoErging approaches is modularity; multiple experts are considered simultaneously and a routing mechanism decides which input, or which part of an input, is processed by which expert.

In this work we consider per-token and per-layer routing, following recent works which suggest this leads to better performance relative to other possible configurations (Ostapenko et al., 2024; Muqeeth et al., 2024). Concretely, let $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}, b \in \mathbb{R}^{d_{\text{out}}}$ denote the weight matrix and bias of a pre-trained linear layer, whose original output is $\mathbf{W}x + b$. We assume the availability of a fine-tuned expert module $E_t(\cdot)$ for each target task $t \in \mathcal{T}$ and we replace the original linear layer with a MoE layer. A router π parameterized by matrix $R \in \mathbb{R}^{|\mathcal{T}| \times d_{\text{in}}}$ computes routing logits Rx and applies softmax $\sigma(\cdot)$ to obtain the routing probabilities. The outputs of the experts with top k highest probabilities are then computed and weight-averaged. The resulting MoE layer output is:

$$y \;=\; \mathbf{W}x + b \;+\; \frac{\sum_{t\in I_k(x)} \pi(x)_t \, E_t(x)}{\sum_{t\in I_k(x)} \pi(x)_t}, \tag{2}$$

where $I_k(x) = \{t \mid \pi(x)_t \in \text{top k elements of } \pi(x)\}$. We use $k = 2$ for our experiments.

We consider the "multi-task" setting where we assume access to all the datasets the experts were trained on. After updating every linear layer of the pre-trained model with available adapters, we continue training the MoE-fied model on the multi-task mixture of data by freezing the original model parameters and only updating the router and the expert modules.

2.3 Low-rank adaptation

Modern foundation models have tens, if not hundreds, of billions of parameters, making full fine-tuning impractical on typical hardware (Grattafiori et al., 2024; DeepSeek-AI, 2024; Team, 2025). Parameter-Efficient Fine-Tuning (PEFT) updates only a small subset of the parameters to ease the computational burden and curb catastrophic forgetting (Hu et al., 2022; Liu et al., 2022). Low-Rank Adaptation (LoRA) (Hu et al., 2022), has emerged as one of the most popular PEFT methods due to its simplicity and effectiveness. LoRA inserts two low-rank matrices $\mathbf{A}$ and $\mathbf{B}$ into selected linear layers of a model. If the input and output dimension at that layer are n_{in} and n_{out}, LoRA uses a rank $r \ll \min(n_{in}, n_{out})$ to define matrices $\mathbf{A} \in \mathbb{R}^{r\times n_{in}}$ and $\mathbf{B} \in \mathbb{R}^{n_{out}\times r}$. The output of that layer then becomes $(\mathbf{W}\mathbf{x} + \mathbf{b}) + \frac{\alpha}{r}\mathbf{B}\mathbf{A}\mathbf{x}$ where α is a scaling hyperparameter. During fine-tuning, the original model parameters are frozen and only the LoRA's $\mathbf{A}, \mathbf{B}$ matrices are updated.

Merging LoRA adapters At each layer, the weight update induced by LoRA is exactly $\Delta W = W_{\text{fine-tuned}} - W_{\text{pre-trained}} = \frac{\alpha}{r}\mathbf{BA}$. Consequently, standard merging techniques can be directly applied to LoRA-adapted models if the updates $\frac{\alpha}{r}\mathbf{BA}$ are added to the pre-trained weights or if they are directly used to compute the task vectors. Merging the LoRA $\mathbf{A}$ and $\mathbf{B}$ matrices separately is not recommended since this can lead to mismatched representation spaces resulting in poor performance (Stoica et al., 2025). Nevertheless, recent work has observed that merging LoRA-adapted models is harder than merging FFT models (Tang et al., 2024; Stoica et al., 2025), often leading to significant performance degradation.

Model MoErging with LoRA adapters Using LoRA adapters for model MoErging is straightforward, with each adapter being used to define one expert module in the MoE layer. Let $\mathbf{A}_t$ and $\mathbf{B}_t$ denote the LoRA low-rank matrices obtained from fine-tuning on task t, then we can define the expert modules in Equation (2) as $E_t(x) = \frac{\alpha}{r}\mathbf{B}_t\mathbf{A}_t x$ for each task of interest $t \in \mathcal{T}$.

2.4 Data difficulty

Prior work has examined how individual data points influence neural network training dynamics and properties such as generalization, memorization, and privacy, leading to the development of various data difficulty scores (Kwok et al., 2024). These scores have been used for data pruning, i.e. removing certain examples from the training set, without harming test performance (Paul et al., 2021). In particular, large fractions of easy examples can be pruned since they contribute little to learning, while removing a small fraction of the hardest examples can improve generalization, as these are likely to be outliers with uncommon features (Toneva et al., 2019), or examples with noisy / incorrect labels (Paul et al., 2021). (Sorscher et al., 2022) further showed that appropriate data pruning can yield better-than-power-law error scaling with dataset size. A natural relationship exists between data difficulty and deep learning generalization and memorization. For instance, Sorscher et al. (2022) found a 0.78 Spearman rank correlation between EL2N scores (Paul et al., 2021) and the memorization score presented by Feldman & Zhang (2020). This indicates that, in order to classify difficult examples, models often need to memorize them. This relationship between memorization and generalization has been further substantiated with theoretical results in simpler settings (Attias et al., 2024; Feldman, 2020).

We utilize data difficulty scores to identify which knowledge is transferred during upcycling. Specifically, we use the EL2N score proposed by Paul et al. (2021) which is the norm of the error vector, i.e. the predicted class probabilities minus the one-hot label encoding. The EL2N score of a training example x with one-hot encoded label y is defined to be $\mathbb{E}\|p(\theta, x) - y\|_2$, where $p(\theta, x)$ are the predicted class probabilities of example x by a deep learning model with parameters θ.

2.5 Models and datasets

Vision domain We evaluate merging performance in a standard vision benchmark setting using the official codebase from Ilharco et al. (2023): a CLIP (Radford et al., 2021) pre-trained ViT-B-32 model (Dosovitskiy et al., 2021) is fine-tuned on 8 image classification tasks: Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Stallkamp et al., 2012), MNIST (Deng, 2012), RESISC45 (Cheng et al., 2017), SUN397 (Xiao et al., 2010) and SVHN (Netzer et al., 2011). The fine-tuning is done with a batch size of 128, the AdamW optimizer (Loshchilov & Hutter, 2019; Paszke et al., 2019) and a learning rate of 1e-5. We use a learning-rate scheduler with linear warm-up for the first 10% of training, followed by cosine annealing. When evaluating merged models, we use the corresponding frozen classification head for each task.

Language domain For our natural language processing (NLP) experiments, we adopt the setting of the TIES paper (Yadav et al., 2023) and use their released code. We use pre-trained T5-Base models (Raffel et al., 2020) which we fine-tune on 7 tasks: QASC (Khot et al., 2020), WikiQA (Yang et al., 2015) and QuaRTz (Tafjord et al., 2019) for question answering; PAWS (Zhang et al., 2019) for paraphrase identification; Story Cloze (Sharma et al., 2018) for sentence completion and Winogrande (Sakaguchi et al., 2020) and WSC (Levesque et al., 2012) for coreference resolution. We use the AdamW (Loshchilov & Hutter, 2019) optimizer with a batch size of 256, a constant lr of 0.0001 and no weight decay. bfloat16 mixed precision training is used to reduce GPU utilization.

Evaluation For all our experiments we report the raw, un-normalized test accuracy averaged across the multiple considered tasks. We chose not to use the popular *normalized accuracy* metric because the set of experts being merged here differs across experiments, which also changes the

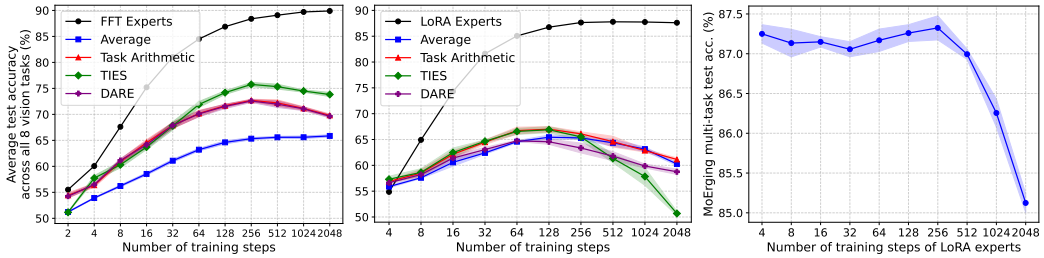

Figure 1: Average test accuracy across the 8 vision classification tasks for merged and MoErged ViT-B-32 experts. **Left:** merging fully fine-tuned experts, we plot the average accuracy of the expert models evaluated on their respective tasks as well as merging accuracies for multiple methods; **Center:** merging LoRA-adapted experts; **Right:** final multi-task accuracy of MoE-fied models vs. LoRA training steps used for initialization. Shaded regions show mean±std over 3 random seeds.

normalization factor and makes comparisons inconsistent. A more detailed justification is provided in Appendix A. Our experiments are ran using the PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2019) open source machine learning frameworks on an Nvidia Quadro RTX 8000 GPU with 48GB of memory.

3 Longer fine-tuning hurts model upcycling

In this section, we present results challenging the common assumption that better fine-tuned models lead to better upcycling results. We show that overtrained experts lead to worse merged models for both FFT and LoRA, as well as lower accuracy when used to initialize MoErging methods.

3.1 Merging fully fine-tuned models

While a multitude of model merging methods have been proposed, the influence of the fine-tuning procedure itself on merging remains understudied. Most prior works have used similar fine-tuning protocols, typically training for a fixed 2000 steps in the vision setting described in Section 2.5. Instead of proposing yet another model merging method, we take a look at how the number of training iterations affects merging. We fine-tune our vision and NLP models for varying number of training steps $s \in \{2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048\}$ on every considered dataset. Each merge combines either 8 vision or 7 NLP experts (one per task) all trained for the same duration.

Undertrained experts result in better merging Figure 1 (left) shows that, except for Average, all methods achieve better merging performance when the ViT experts are trained for just 256 training steps, only ∼1/8 of the commonly used 2000. TA, TIES, and DARE yield models with ∼3% higher accuracy at 256 steps compared to 2048, a gain comparable to the 3.4% gap between TA and the more sophisticated TIES at 2048 steps. The same conclusions hold in the NLP setting (Figure 2 left), with both TA and TIES peaking around 256–512 training steps. Further training leads to a drop in merging performance of over 3% for both merging methods. Notably, merging undertrained experts with TA outperforms merging experts trained for longer with TIES. Average is the only method that seems to benefit from training the experts longer, but it consistently underperforms overall. Moreover, TA, TIES, and DARE show similar trends across training durations, suggesting that training length itself, rather than the merging method, plays a key role in merging performance.

Better experts do not necessarily lead to better merging The black lines in the left and central panels of Figures 1 and 2 show the average accuracy of the expert models on their respective fine-tuning tasks. In both the vision and NLP settings, we observe that higher expert accuracy does not necessarily translate into better merging performance. In the vision setting, expert models trained for 256 steps achieve an average accuracy of 88.4%, which is 1.6% lower than at 2048 steps (90.0%). Nevertheless, merging after 256 steps yields models with approximately 3% higher accuracy than merging after 2048 steps. The discrepancy is even more pronounced in the NLP setting. Expert accuracy improves from 78.2% at 256 steps to 82.4% at 1024 steps, a 4% gain, yet the merging accuracy of TA and TIES drops by around 3% over the same interval.

Effect of model scale In the right panel of Figure 2, we compare Task Arithmetic merging accuracy across different model sizes in the T5 family: T5-Base (220M parameters), T5-Large (770M),

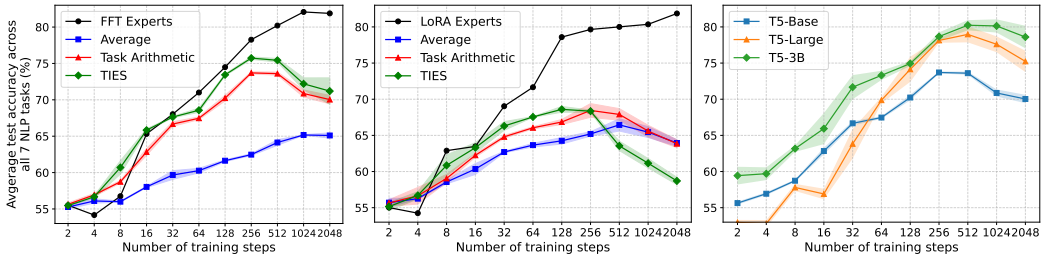

Figure 2: Average test accuracy across all 7 NLP tasks for fully fine-tuned (**left**) and LoRA-adapted (**center**) T5-Base models. We plot the average accuracy of the expert models evaluated on their respective tasks as well as merging accuracies for multiple methods. **Right:** Task Arithmetic merging accuracy for different T5 model sizes. Shaded regions show mean±std over 3 random seeds.

and T5-3B (3B). We observe that the same trend persists across scales: upcycling performance peaks at an intermediate number of training steps before degrading with longer fine-tuning. Additional merging results are provided in Appendix D.

3.2 Merging LoRA adapters

We now extend our previous results to the highly relevant setting of merging LoRA adapters. We find that long training of LoRA experts hurts merging performance even more than in the FFT case. We add LoRA adapters at every linear layer of the original ViT-B-32 and T5-Base models. We use LoRA rank $r = 8$, scaling parameter $\alpha = 32$ and learning rates 1e-4 and 5e-4 for the ViT and T5 models respectively. We train the LoRAs for different number of steps s to evaluate the impact of training duration on accuracy and mergeability. The parameters of the base model are kept frozen.

Overtraining severely impairs LoRA merging The center panels of Figures 1 and 2 show expert and merging accuracies for our vision and NLP LoRA models, respectively. For the ViT models, merging performance peaks at 128 training steps (64 for DARE), with accuracies ranging from 65–67% across all methods. Although further training improves expert accuracy by about 1%, it significantly degrades merging performance, with accuracy drops of 5–6% for Average, TA, and DARE, and nearly 17% for TIES. In the NLP setting, different methods reach peak merging performance at different training durations: 512 steps for Average (66.5%), 256 for TA (68.5%), and 128 for TIES (68.6%). Expert models, however, continue to improve, reaching an average accuracy of 81.9% at 2048 steps. Despite this, merging at 2048 steps harms performance, with drops of 2.5%, 4.6%, and 9.9% for Average, TA, and TIES, respectively. In Appendix E, we examine the impact of LoRA rank and show that higher ranks lead to smaller performance degradations when merging.

3.3 Model MoErging with LoRA experts

We next analyze how the performance of MoE-fied models, initialized with LoRA experts, is affected by the training time of these experts. We use the LoRA adapters obtained in Section 3.2 with different number of training steps to initialize our MoE experts, one LoRA for each task. The routing mechanism is initialized using Arrow (Ostapenko et al., 2024), where the weight vector associated with each expert is the first right-singular vector of the BA matrix multiplication. These vectors are assumed to determine the direction of most variance induced by expert E_t for $t \in \mathcal{T}$ in the space of hidden states induced by data from task t.

We create one MoE-fied model for each number of steps s, i.e. for each different model we initialize the MoE layers with the expert LoRAs for each task, all trained for s steps. Once the MoE-fied model has been initialized using the fine-tuned LoRAs, we further train the routing mechanism and the LoRA experts in a multi-task fashion for 4000 steps with a peak learning rate of 1e-5, with the base parameters frozen. We report the final, multi-task, accuracies over the 8 classification tasks in the right panel of Figure 1.

We observe that the MoE-fied models initialized with overtrained LoRA experts reach about 2% lower final multi-task accuracy than the models initialized with experts trained for less. Even expert LoRAs trained for as little as 4 steps on their respective tasks reach a higher final multi-task accuracy than those overtrained. We conclude that overtraining experts can hurt downstream MoErging.

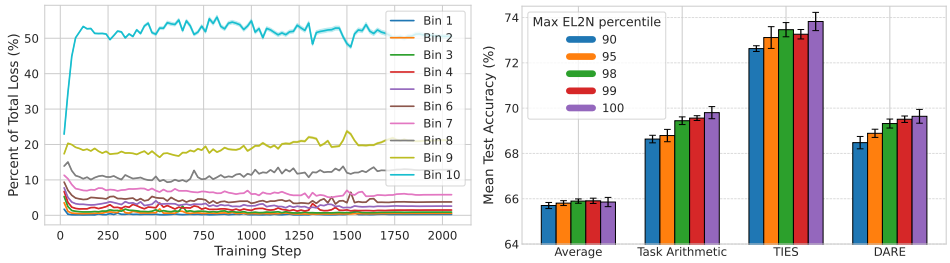

Figure 3: **Left:** Percentage of total loss for examples in different data difficulty bins. Bin 1 represents 10% easiest examples (lowest EL2N scores), bin 10 represents 10% hardes examples (highest EL2N scores). Mean across all 8 vision datasets shown. **Right:** Merging accuracy for experts trained without the hardest examples. Experts are trained on data with EL2N scores from percentile 0 to varying max percentiles in $\{90, 95, 98, 99, 100\}$.

4 Why is undertraining beneficial for merging?

Easy examples are learned early during training while harder examples are learned later. To link our main observation to the training duration of the expert models we track the loss of the training examples during training, these results are shown in the left panel of Figure 3. We group the training examples into 10 bins according to their data difficulty scores, the 10% of examples with the lowest EL2N scores are in bin 1, etc. EL2N scores are computed early in fine-tuning, after only 32 steps, across 10 different seeds. We observe that easy examples, which have more common features, are learned early in training. The rest of training is dedicated to learning the more difficult examples. In fact, the top 10% of hardest examples account for over 50% of the total loss during most of training. As discussed in Section 2.4, these results imply that in later training steps models try to memorize difficult examples with uncommon features or noisy labels.

Model merging leads to the forgetting of difficult examples. To analyze why merging benefits from less training of the expert models we take a look at which examples are forgotten during merging, i.e. which examples from the training set are correctly classified by the expert models but incorrectly classified once these models are merged. We hypothesize that merging primarily affects the classification of difficult examples. Memorizing such examples, with uncommon features or noisy labels, is likely to yield parameter updates which are unique from one dataset to the other, and which will be destroyed by the aggregation step of model merging.

Figure 4 shows pie charts of the examples which are forgotten during merging, with each "slice" representing one of ten data difficulty bins. Hard examples are overwhelmingly forgotten when merging, with over 50% of forgotten data points being in the top 30% in terms of data difficulty.

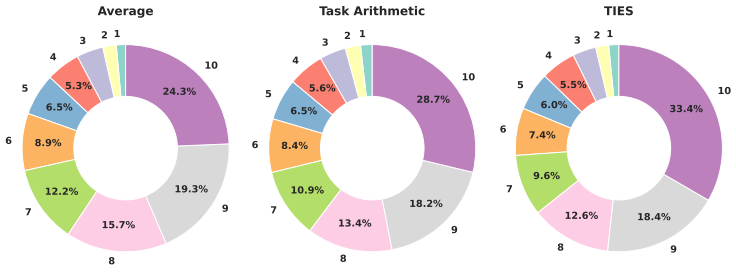

Figure 4: Proportion of forgotten examples in each data difficulty bin for three different model merging methods. Bin 1 represents 10% easiest examples (lowest EL2N scores), bin 10 represents 10% hardest examples (highest EL2N scores). Hard examples are overwhelmingly forgotten when merging with all methods, with the 30% hardest examples representing over 50% of forgotten examples.

From these 2 observations, we conclude that fine-tuning for longer, which mainly helps the experts memorize difficult examples, is not beneficial to merging since those harder examples will most likely be forgotten during the merging procedure.

Difficult examples are still necessary for good generalization. We remove difficult examples from expert training to see how this effects merging performance. Past work has determined that removing a small percentage of the most difficult examples can help generalization (Toneva et al., 2019; Paul et al., 2021). We remove the top 1, 2, 5 or 10% most difficult examples from training to see the impact on downstream merging, the results are shown in the right panel of Figure 3. We see that the best merging results are achieved when the entire available data is used for training. Removing a fraction of the most difficult examples consistently yields lower merging performance, with more data removed leading to greater performance loss. This suggests that some amount of memorization of hard examples / uncommon features during fine-tuning is beneficial for merging.

5 AGGRESSIVE EARLY STOPPING IMPROVES UPCYCLING RESULTS

We next examine the variability of optimal expert training time among different tasks. We find that upcycling can be further improved if the stopping time is optimized for a specific task, and propose a strategy on when to stop training.

The learning rate scheduler we use in Section 3, i.e. linear warm-up followed by cosine decay, is a popular choice in the literature for training vision models, and has been extensively used in recent model merging papers. Both the warm-up and the decay phases are beneficial for performance since the former provides stability at the start of training while the latter helps convergence with smaller steps at the end of training. Therefore, our early stopping strategy uses a learning rate scheduler with warm-up and decay phases which can adapt to the varying training length induced by early stopping. Altogether, our proposed early stopping strategy uses a simple learning rate scheduler paired with an early stopping condition: a linear warm-up phase of a fixed number of steps followed by a "reduce learning rate on plateau" phase which gradually decreases the learning rate when a plateau is observed in the validation accuracy. Once the learning rate is decreased below a certain threshold, training is stopped.

Table 1: Merging accuracy (%) for the overtrained, optimal and early stopped experts. Mean and standard deviation across 3 random seeds shown.

	Average	Task Arithmetic	TIES	DARE
FFT 2048 steps	65.9 ±0.2	69.8 ±0.27	73.8 ±0.4	69.6 ±0.3
FFT best (# steps)	65.9 ±0.2 (2048)	72.7 ±0.2 (256)	75.8 ±0.4 (256)	72.6 ±0.3 (256)
FFT early stop	64.5 ±0.1	72.6 ±0.5	74.7 ±0.3	72.5 ±0.5
LoRAs 2048 steps	60.3 ±0.3	61.1 ±0.3	50.7 ±0.4	58.8 ±0.5
LoRAs best (# steps)	65.4 ±0.4 (128)	67.0 ±0.5 (128)	66.9 ±0.6 (128)	64.7 ±0.1 (64)
LoRAs early stop	65.6 ±0.4	68.0 ±0.8	67.1 ±0.6	65.3 ±0.3

We fine-tune FFT and LoRA models on the 8 considered vision tasks. We use a fixed number of 50 steps for the linear warm-up, then we evaluate accuracy on a validation set every 5 training steps and multiply the learning rate by a factor of 0.5 when the validation accuracy has not improved for 3 consecutive validation rounds. The peak learning rates are 1e-5 and 1e-4 for the FFT and LoRA models respectively. In Table 1, we report the merged model's average accuracy across eight tasks. We compare the merging of early stopped experts to two baselines from Section 3: merging "overtrained" models (trained for 2048 steps) and merging the checkpoints that achieved the highest accuracy among all training durations. We see that the models trained using our simple task-dependent early stopped strategy yield merges that are better than those of overtrained models and as good, if not better, than the best merged experts obtained from a single stopping time, as presented in Sections 3.1 and 3.2. Early stopping seems to work especially well for LoRA adaptation, yield results on average better than the best ones from Section 3.2.

Table 2: Early stopping MoErging results

Expert initialization	Avg. accuracy
2048 steps LoRAs	85.1 ± 0.1
Best LoRAs (256 steps)	87.3 ± 0.2
Early stop LoRAs	87.3 ± 0.1

We also use the early-stopped LoRAs to initialize MoE layers and continue training in a multi-task fashion, as in Section 3.3. As shown in Table 2, the MoErged models initialized with the early stop LoRAs achieve the same accuracy as the best LoRAs across all training steps.

6 Related work

Model merging Combining multiple versions of a model into a single, more capable one has been a powerful technique in deep learning, and a very active research area (Yang et al., 2024). We review some of the popular methods in Appendix B. These merging methods often rely on the so-called *linear mode connectivity* (Frankle et al., 2020; Sharma et al., 2024), i.e., minima in a deep learning model's loss landscape that are connected by a low loss linear path in parameter space. Models that share a significant part of their training trajectories were found to be linearly mode connected (Frankle et al., 2020; Neyshabur et al., 2020). Therefore, it is generally assumed that different fine-tuned versions of the same pre-trained model can be merged successfully. Sharma et al. (2024) goes beyond that, exploring merging of experts that were trained from different or poorly performing pre-trained models. However, little attention has been paid to how the expert fine-tuning procedure itself, specifically its duration, affects merging performance.

Model MoErging Model MoErging methods propose to re-use expert modules by combining them into mixture-of-experts (MoE) layers (Shazeer et al., 2017), with a router deciding which input, or part of an input, is processed by which expert module. Numerous model MoErging techniques have been proposed, with varying expert, router and application design choices (Yadav et al., 2024; Huang et al., 2024; Ostapenko et al., 2024; Muqeeth et al., 2024). While existing surveys and methods focus on routing algorithms and module selection, none examine how expert overtraining influences downstream MoErging efficacy to our knowledge. Our MoErging setup is comparable to Ostapenko et al. (2024), where LoRA experts are combined into MoE layers and the router is initialized with Arrow, except that we assume access to training data and continue training.

Expert training time Most model merging and MoErging papers do not examine how expert fine-tuning affects downstream upcycling, there are however two notable exceptions. Zhou et al. (2025) show that the effectiveness of taskvector based approaches is largely driven by first-epoch gradients and therefore propose alternating 1-epoch fine-tuning and merging. While they note that less training can improve accuracy, they only test 1 epoch. Given the disparity in dataset sizes, 1 epoch of training can yield either overtrained experts (on large datasets) or undertrained experts (on small datasets). Secondly, Pari et al. (2024) observe representational incompatibilities when merging highly specialized experts but study only two-model merges and their solution is to bypass merging altogether and use MoErging instead. To our knowledge, we are the first to systematically link expert training duration to downstream merging and MoErging outcomes, to analyze merging through example difficulty, and to propose an early-stopping strategy that adapts to dataset heterogeneity. Finally, although the TIES Merging paper (Yadav et al., 2023) uses early stopping, it is only used to avoid expert overfitting and its effect on merging is not studied.

Analogous to our work, others have studied how scaling pre-training impacts downstream fine-tuning. A large scale study on vision models, Abnar et al. (2022) found that as pre-training accuracy improves, fine-tuning saturates. More recently, Springer et al. (2025) show that over-training LLMs during pre-training can harm fine-tuned performance on both in- and out-of-distribution tasks.

7 Conclusion

In this paper, we challenged the assumption that better fine-tuned experts yield better upcycling performance. Across multiple merging methods, model sizes and for both fully fine-tuned and LoRA-adapted models, we found that optimal merging occurs well before full convergence, often when experts are less accurate on their original tasks. For MoErging, continued fine-tuning of LoRA experts even degrades downstream multi-task performance. We attribute this to a shift in training dynamics: as fine-tuning progresses, the training loss is dominated by a small subset of difficult examples whose memorization does not survive merging, an insight supported by tools from the data difficulty literature. Finally, we show that a simple early stopping strategy mitigates overtraining and restores near-optimal upcycling performance.

While our findings do not offer actionable insights for existing adapters (e.g., HuggingFace, AdapterHub), they have important implications for publishing new adapters and evaluating upcycling pipelines. **Publish intermediate checkpoints:** Releasing not only final but also intermediate checkpoints is crucial, as the best upcycling point may precede convergence. **Prioritize early-stopped experts:** When training experts in-house, aggressive early stopping can outperform convergence for downstream upcycling. Since upcycling reuses checkpoints and amortizes sunk costs, our findings can help reduce the future computational and environmental footprint of training AI models.

8 Reproducibility Statement

We have taken several steps to ensure the reproducibility of our work. In Section 2.5 we describe the exact models, datasets, and codebases used, as well as the machine learning frameworks and hardware employed. All frameworks and codebases are open-sourced and publicly available, and our exact codebase is provided as supplementary material. In addition, the main text and Appendix C include all relevant details and a description of our hyperparameter tuning procedures, ensuring that our experiments can be fully reproduced.

References

Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=V3C8p78sDa.

Idan Attias, Gintare Karolina Dziugaite, Mahdi Haghifam, Roi Livni, and Daniel M Roy. Information complexity of stochastic convex optimization: Applications to generalization and memorization. *arXiv preprint arXiv:2402.09327*, 2024.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open LLM Leaderboard (2023–2024). https://huggingface.co/spaces/open-llm-leaderboard-old/open_llm_leaderboard, 2023. Accessed: 2025-04-29.

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, Oct 2017. ISSN 1558-2256. doi: 10.1109/jproc.2017.2675998. URL http://dx.doi.org/10.1109/JPROC.2017.2675998.

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. Fusing finetuned models for better pretraining, 2022. URL https://arxiv.org/abs/2204.03044.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

MohammadReza Davari and Eugene Belilovsky. Model breadcrumbs: Scaling multi-task model merging with sparse masks. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part LXXV*, pp. 270–287, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-73225-6. doi: 10.1007/978-3-031-73226-3_16. URL https://doi.org/10.1007/978-3-031-73226-3_16.

Pala Tej Deep, Rishabh Bhardwaj, and Soujanya Poria. Della-merging: Reducing interference in model merging through magnitude-based sampling, 2024. URL https://arxiv.org/abs/2406.11617.

DeepSeek-AI. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 954–959, 2020.

Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 2881–2891. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1e14bfe2714193e7af5abc64ecbd6b46-Paper.pdf.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3259–3269. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/frankle20a.html.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, Vladimir Karpukhin, Brian Benedict, Mark McQuade, and Jacob Solawetz. Arcee's MergeKit: A toolkit for merging large language models. In Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 477–485, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-industry.36. URL https://aclanthology.org/2024.emnlp-industry.36.

Aaron Grattafiori et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Ethan He, Abhinav Khattar, Ryan Prenger, Vijay Korthikanti, Zijie Yan, Tong Liu, Shiqing Fan, Ashwath Aithal, Mohammad Shoeybi, and Bryan Catanzaro. Upcycling large language models into mixture of experts, 2024. URL https://arxiv.org/abs/2410.07524.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2019.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic loRA composition. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=TrloAXEJ2B.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=6t0Kwf8-jrj.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew G. Wilson. Averaging weights leads to wider optima and better generalization. In Amir Globerson and Ricardo Silva (eds.), *Uncertainty in Artificial Intelligence*. AUAI Press, 2018.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FCnohuR6AnM.

Nikhil Kandpal, Brian Lester, Mohammed Muqeeth, Anisha Mascarenhas, Monty Evans, Vishal Baskaran, Tenghao Huang, Haokun Liu, and Colin Raffel. Git-theta: A git extension for collaborative development of machine learning models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning*

Research, pp. 15708–15719. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/kandpal23b.html.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. Qasc: A dataset for question answering via sentence composition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8082–8090, Apr. 2020. doi: 10.1609/aaai.v34i05.6319. URL https://ojs.aaai.org/index.php/AAAI/article/view/6319.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

Devin Kwok, Nikhil Anand, Jonathan Frankle, Gintare Karolina Dziugaite, and David Rolnick. Dataset difficulty and the role of inductive bias. *arXiv preprint arXiv:2401.01867*, 2024.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, pp. 552–561. AAAI Press, 2012. ISBN 9781577355601.

Haokun Liu, Derek Tam, Muqeeth Mohammed, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=rBCvMG-JsPd.

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

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17703–17716. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/70c26937fbf3d4600b69a129031b66ec-Paper-Conference.pdf.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu (eds.), *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282. PMLR, 20–22 Apr 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

Mohammed Muqeeth, Haokun Liu, Yufan Liu, and Colin Raffel. Learning to route among specialized experts for zero-shot generalization, 2024. URL https://arxiv.org/abs/2402.05859.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 512–523. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/0607f4c705595b911a4f3e7a127b44e0-Paper.pdf.

Oleksiy Ostapenko, Zhan Su, Edoardo Ponti, Laurent Charlin, Nicolas Le Roux, Lucas Caccia, and Alessandro Sordoni. Towards modular LLMs by building and reusing a library of LoRAs. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 38885–38904. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ostapenko24a.html.

Jyothish Pari, Samy Jelassi, and Pulkit Agrawal. Collective model intelligence requires compatible specialization, 2024. URL https://arxiv.org/abs/2411.02207.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 20596–20607. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/ac56f8fe9eea3e4a365f29f0f1957c55-Paper.pdf.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. AdapterHub: A framework for adapting transformers. In Qun Liu and David Schlangen (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 46–54, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.7. URL https://aclanthology.org/2020.emnlp-demos.7/.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8732–8740, Apr. 2020. doi: 10.1609/aaai.v34i05.6399. URL https://ojs.aaai.org/index.php/AAAI/article/view/6399.

Ekansh Sharma, Daniel M Roy, and Gintare Karolina Dziugaite. The non-local model merging problem: Permutation symmetries and variance collapse. *arXiv preprint arXiv:2410.12766*, 2024.

Rishi Sharma, James Allen, Omid Bakhshandeh, and Nasrin Mostafazadeh. Tackling the story ending biases in the story cloze test. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 752–757, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2119. URL https://aclanthology.org/P18-2119/.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 19523–19536. Curran Associates, Inc.,

2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/7b75da9b61eda40fa35453ee5d077df6-Paper-Conference.pdf.

Jacob Mitchell Springer, Sachin Goyal, Kaiyue Wen, Tanishq Kumar, Xiang Yue, Sadhika Malladi, Graham Neubig, and Aditi Raghunathan. Overtrained language models are harder to fine-tune, 2025. URL https://arxiv.org/abs/2503.19206.

J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32(0):–, 2012. ISSN 0893-6080. doi: 10.1016/j.neunet.2012.02.016. URL http://www.sciencedirect.com/science/article/pii/S0893608012000457.

Stanford-CRFM. On the opportunities and risks of foundation models. *ArXiv*, 2021. URL https://crfm.stanford.edu/assets/report.pdf.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the knots. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=67X93aZHII.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. QuaRTz: An open-domain dataset of qualitative relationship questions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 5941–5946, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1608. URL https://aclanthology.org/D19-1608/.

Derek Tam, Mohit Bansal, and Colin Raffel. Merging by matching models in task parameter subspaces. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=qNGo6ghWFB.

Anke Tang, Li Shen, Yong Luo, Yibing Zhan, Han Hu, Bo Du, Yixin Chen, and Dacheng Tao. Parameter-efficient multi-task model fusion with partial linearization. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=iynRvVVAmH.

Gemma Team. Gemma 3 technical report, 2025. URL https://arxiv.org/abs/2503.19786.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJlxm30cKm.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019. URL http://arxiv.org/abs/1910.03771.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7959–7971, June 2022.

J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3485–3492, June 2010. doi: 10.1109/CVPR.2010.5539970.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=xtaX3WyCj1.

Prateek Yadav, Colin Raffel, Mohammed Muqeeth, Lucas Caccia, Haokun Liu, Tianlong Chen, Mohit Bansal, Leshem Choshen, and Alessandro Sordoni. A survey on model moerging: Recycling and routing among specialized experts for collaborative learning, 2024. URL https://arxiv.org/abs/2408.07057.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*, 2024.

Yi Yang, Wen-tau Yih, and Christopher Meek. WikiQA: A challenge dataset for open-domain question answering. In Lluís Màrquez, Chris Callison-Burch, and Jian Su (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2013–2018, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1237. URL https://aclanthology.org/D15-1237/.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57755–57775. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/yu24p.html.

Qizhen Zhang, Nikolas Gritsch, Dwaraknath Gnaneshwar, Simon Guo, David Cairuz, Bharat Venkitesh, Jakob Nicolaus Foerster, Phil Blunsom, Sebastian Ruder, Ahmet Üstün, and Acyr Locatelli. BAM! just like that: Simple and efficient parameter upcycling for mixture of experts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=BDrWQTrfyI.

Yuan Zhang, Jason Baldridge, and Luheng He. PAWS: Paraphrase adversaries from word scrambling. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *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)*, pp. 1298–1308, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1131. URL https://aclanthology.org/N19-1131/.

Luca Zhou, Daniele Solombrino, Donato Crisostomi, Maria Sofia Bucarelli, Fabrizio Silvestri, and Emanuele Rodolà. Atm: Improving model merging by alternating tuning and merging, 2025. URL https://arxiv.org/abs/2411.03055.

A USING THE RAW, UN-NORMALIZED ACCURACY

The *normalized accuracy* is a very common metric used to compare model merging methods (Ilharco et al., 2023; Yadav et al., 2023). However, because the normalized accuracy depends on both the merged model's performance and that of the experts, it isn't suitable for settings like ours where different sets of experts are used and compared.

The core issue is that normalized accuracy, defined as (merged_accuracy / expert_accuracy), is a relative metric designed to compare different merging methods when the set of experts (the denominator) is fixed. Papers that propose a novel model merging method are justified in using this metric by the fact that they have a fixed set of experts and they are comparing merging methods, therefore only the numerator changes. In our study, the experts themselves are the primary variable, as their training duration and performance change in each experiment, therefore the denominator changes from one merging experiment to another. This creates paradoxical situations that make the metric misleading for our purposes. For example consider the following scenario:

- **Case 1 (Undertraining):** Experts trained for only a few steps have very low absolute accuracy (e.g., 60%). When merged, they interfere very little since they're all relatively close in parameter space to the zeroshot model, so the merged model also achieves around 60% accuracy. This yields a normalized accuracy near 100%, despite the models being bad at solving the considered tasks.
- **Case 2 (Optimal Training):** Experts trained for longer have high accuracy (e.g., 90%). Merging them results in a high-performing model with 85% absolute accuracy. However, the normalized accuracy is only $85/90 = 94.4\%$ due to negative interference caused by longer training.

Comparing the "useless" 100% from Case 1 with the "useful" 94.4% from Case 2 is meaningless. Absolute, un-normalized accuracy on the other hand allows for a fair and interpretable comparison of the final upcycled model's quality across different expert training durations.

B ADDITIONAL RELATED WORK

The simplest approach, parameter averaging, was shown to lead to better generalization when used on checkpoints from the same training trajectory (Izmailov et al., 2018) and was popularized in federated learning with FedAvg (McMahan et al., 2017). Recently, parameter averaging was also shown to be useful in the context of robust fine-tuning (Wortsman et al., 2022) and to obtain better pre-trained models (Choshen et al., 2022). When merging multiple fine-tuned versions of the same pre-trained model, Fisher-weighted averaging (Matena & Raffel, 2022) and related methods improve upon this simple averaging by adjusting per-parameter contributions (Jin et al., 2023; Tam et al., 2024). Task arithmetic based methods rely on the computation of task vectors, which are then summed, scaled and added back to the pretrained model (Ilharco et al., 2023) to give it multi-task capabilities. Pruning the task vector parameters (Yadav et al., 2023; Davari & Belilovsky, 2024; Yu et al., 2024; Deep et al., 2024) and selectively combining them to reduce negative interference (Yadav et al., 2023) further benefits performance.

C TUNING MERGING HYPERPARAMETERS

Several merging methods require careful hyperparameter tuning to achieve optimal performance. In particular, Task Arithmetic, TIES, and DARE each apply a scaling factor α to their task-vector sums before adding them to the pretrained weights; TIES and DARE additionally specify a percentage k of weights to retain after pruning. As is standard, we select the best $\alpha,\ k$ values by maximizing merging accuracy on a held-out validation set. All merging accuracies reported in the main text are evaluated on the test set using hyperparameters selected via validation performance. We followed the hyperparameter configurations from the original papers (Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024), adjusting them as needed to optimize performance in our experimental settings.

Vision setting: Following (Ilharco et al., 2023), we reserve 10% of the training data for validation and train the ViT models on the remaining 90%. We tune the following hyperparameter values using the validation set:

- **Task Arithmetic:** $\alpha \in \{0.05, 0.1, \ldots, 1\}$
- **TIES:** $\alpha \in \{0.5, 0.6, \ldots, 1.5\}$ and $k \in \{10, 20, 30\}$
- **DARE:** $\alpha \in \{0.05, 0.1, \ldots, 0.55\}$ and $k \in \{10, 20, 30\}$

NLP setting: We adopt the validation splits from (Yadav et al., 2023) and evaluate the following hyperparameter values:

- **Task Arithmetic:** $\alpha \in \{0.1, 0.2, \ldots, 1\}$
- **TIES:** $\alpha \in \{0.8, 0.9, \ldots, 2.1\}$ and $k \in \{10, 20, 30\}$

D Effect of model scale

In the right panel of Figure 2, we have investigated how model size influences the merging dynamics by comparing Task Arithmetic merging results across T5-Base (220M parameters), T5-Large (770M), and T5-3B (3B). In Figure 5 we also show the average expert accuracy on their respective tasks as well as the merging accuracies for Average, Task Arithmetic and TIES methods. The purpose of these experiments is to test whether the decrease in merging accuracy observed after extended fine-tuning in smaller models also occurs at larger scales. We find that the same phenomenon persists: for both Task Arithmetic and TIES, merging accuracy peaks at an intermediate number of training steps and then degrades as fine-tuning continues, even though the absolute merging accuracy is generally higher for the larger models. Interestingly, Average merging appears robust to this degradation, but its overall accuracy remains comparatively low.

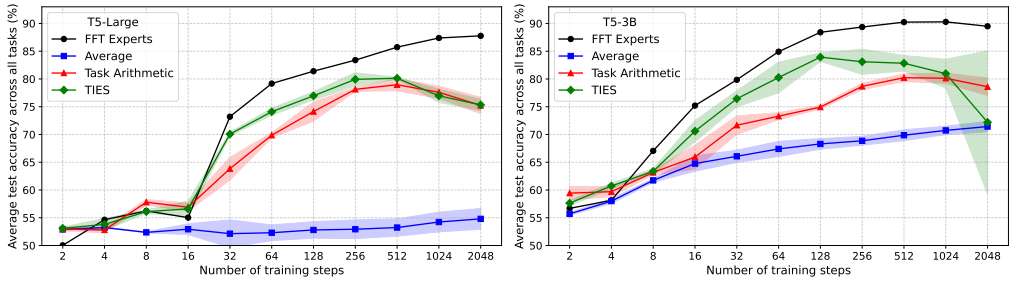

Figure 5: Average test accuracy across all 7 NLP tasks for fully fine-tuned T5-Large (**left**) and T5-3B (**right**) models. We plot the average accuracy of the expert models evaluated on their respective tasks as well as merging accuracies for multiple methods. Shaded regions show mean±std over 3 random seeds.

E Effect of LoRA rank

In this section, we examine how the choice of LoRA rank affects the degradation effect reported in the main paper. We find that increasing the LoRA rank mitigates the loss in merging accuracy that occurs as experts are trained for longer.

We fine-tune ViT-B-32 models on the eight image-classification tasks from Section 2.5, applying LoRA adapters to every linear layer while systematically varying the adapter rank r. We employ square-root scaling for the LoRA factor α, choosing $(r, \alpha) \in \{(16, 45), (32, 64), (64, 90), (128, 128), (256, 181)\}$. The models are trained for varying number of steps $s \in \{8, 32, 128, 512, 2048\}$ to assess how training duration interacts with rank. When merging, we combine LoRA-adapted models with the same rank and trained for the same number of steps. The resulting accuracies are plotted in Figure 6.

Across all three merging methods (Average, Task Arithmetic, and TIES) increasing the LoRA adapter rank consistently raises merging accuracy at every training duration. Moreover, higher

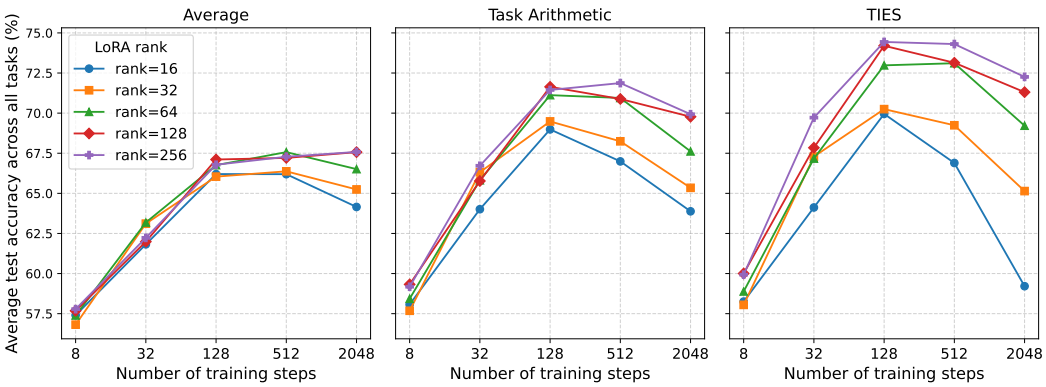

Figure 6: Average test accuracy across all 8 vision classification tasks as a function of the number of fine-tuning steps for different LoRA ranks and three merging methods. Each panel shows one method: Average (left), Task Arithmetic (center) and TIES (right). Colored solid lines and distinct markers denote the different LoRA adapter ranks. The x-axis is in $\log_2$ scale.

ranks substantially attenuate the accuracy drop associated with extended training: as the number of fine-tuning steps grows, models with larger ranks exhibit smaller declines from their peak merging performance.