

SpectR: Dynamically Composing LM Experts with Spectral Routing

William Fleshman & Benjamin Van Durme

Johns Hopkins University
will.fleshman@jhu.edu

Abstract

Training large, general-purpose language models poses significant challenges. The growing availability of specialized *expert* models, fine-tuned from pretrained models for specific tasks or domains, offers a promising alternative. Leveraging the potential of these existing expert models in real-world applications requires effective methods to select or merge the models best suited for a given task. This paper introduces SPECTR, an approach for dynamically composing expert models at each time step during inference. Notably, our method requires no additional training and enables flexible, token- and layer-wise model combinations. Our experimental results demonstrate that SPECTR improves routing accuracy over alternative training-free methods, increasing task performance across expert domains.¹

1 Introduction

Language models (LMs) have increasingly become more capable in recent years with model size being an important factor in scaling performance (Kolachina et al., 2012; Hestness et al., 2017; Kaplan et al., 2020; Hoffmann et al., 2022). Unfortunately, large models are inherently more resource-intensive, motivating the development of efficiency-boosting strategies such as knowledge distillation (Hinton et al., 2015; Gou et al., 2021), quantization (Shen et al., 2020; Kim et al., 2021; Wu et al., 2022; Dettmers & Zettlemoyer, 2023), and pruning (Fang et al., 2023; Ma et al., 2023). Architecting models as a Mixture-of-Experts (MoEs) has also become popular for state-of-the-art open-source LMs such as the DeepSeek (DeepSeek-AI et al., 2024) Mixtral (Jiang et al., 2024), and Qwen (Qwen et al., 2025) model families. MoEs gain efficiency by learning to activate a smaller subset of parameters for each input, reducing computation while allowing for larger model sizes (Jacobs et al., 1991; Fedus et al., 2022).

Simultaneously, parameter efficient fine-tuning (Mangrulkar et al., 2022) methods such as adapter-tuning (Bapna & Firat, 2019; Houlsby et al., 2019), prefix-tuning (Li & Liang, 2021), and prompt-tuning (Lester et al., 2021; Liu et al., 2022) have emerged to allow efficient customization of pretrained models for expert-level performance in new domains or tasks. Low-rank adaptation (LoRA) (Hu et al., 2022) is one of the most widely adopted adapter-tuning approaches, resulting in the proliferation of thousands of expert models openly available in repositories such as huggingface (Wolf et al., 2020). Given the abundance of these experts, many strategies have been proposed for merging models to improve multi-task capability (Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024; Stoica et al., 2025).

Model *MoErging* pairs these merging strategies with routing mechanisms akin to MoEs (Yadav et al., 2024). Unfortunately, the majority of existing MoErging methods require training data for learning to route (Pfeiffer et al., 2021; Shnitzer et al., 2023; Huang et al., 2024; Tang et al., 2024) or custom training procedures for enabling adapter compatibility or gathering activation statistics (Chronopoulou et al., 2023; Diao et al., 2023; Belofsky, 2024; Fleshman et al., 2024). These constraints motivate the development of *training-free* MoErging approaches which require no data and allow for the use of externally sourced LoRA experts.

¹Code available at <https://github.com/wfleshman/spectr>

One training-free option is the use of μ -routing, which forgoes selecting specific experts and instead routes every input to all models (Caccia et al., 2023; Ostapenko et al., 2024). While dense linear combinations of experts can be successful (Wortsman et al., 2022; Chronopoulos et al., 2023; Ilharco et al., 2023), merging several LoRAs is especially problematic due to interference among adapters (Ortiz-Jimenez et al., 2023; Tang et al., 2024; Stoica et al., 2025).

Ostapenko et al. (2024) proposed Arrow routing, which uses the singular value decomposition of LoRA adapters for crafting a prototype vector per expert. These prototypes are then used to score input vectors by measuring the magnitude of their dot product, routing then to the top-k experts with the highest score. We recognize Arrow as a significant contribution, while developing an improvement we refer to as Spectral Routing (SPECTR) that addresses limitations in existing training-free methods such as Arrow (Figure 1). Specifically, we:

- Identify and confirm weaknesses in existing training-free routing strategies;
- Propose SPECTR, our approach designed to explicitly address these challenges;
- Measure the impact of adapter rank and task similarity on routing effectiveness, demonstrating improved routing accuracies with SPECTR across 4 LMs;
- Quantify the impact of different routing strategies on multi-task performance, with SPECTR increasing accuracy by up to 15% over alternatives; and
- Discuss trade-offs between SPECTR and other training-free methods.

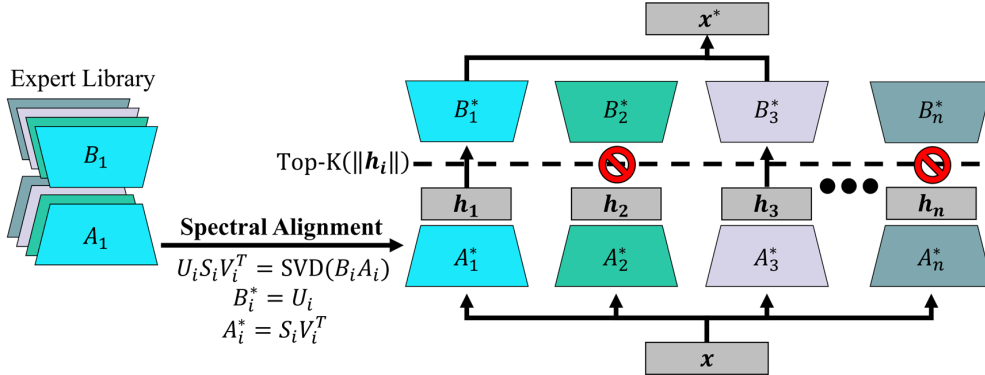


Figure 1: SPECTR uses the SVD to transform adapters into equivalent representations capable of measuring the compatibility of new inputs without relying on expert prototypes or routing networks. Vectors (x) are projected into low-rank representations (h_i) using the eigenvectors (A_i^*) of each adapter covariance matrix. The length of these representations measures the alignment of the input with directions of maximum variation induced by the expert in the space of possible inputs. Routing continues for the top-k compatible experts.

2 Background and Motivation

2.1 Adapters as Expert Models

Adapters enable parameter efficient fine-tuning (PEFT) of existing models to new tasks or domains by adding a small number of trainable parameters to existing models (Bapna & Firat, 2019; Houlsby et al., 2019). Low-rank adaptation (LoRA) is one of the most popular and effective adapter-tuning approaches, accomplished by adding a low-rank matrix product to each adapted layer (Hu et al., 2022). LoRA’s popularity is partly due to technologies such as Mangrulkar et al. (2022)’s peft library and (Wolf et al., 2020)’s huggingface model repository, resulting in wide availability of expert LoRA models trained for a diverse set of tasks and domains (Brüel-Gabrielsson et al., 2024). In this work, we use LoRA adapters to derive several *expert* models, each capable of superior performance on their specific task. We

explore methods for dynamically selecting and combining these experts at inference time with no prior knowledge of the applicability or compatibility of individual expert models.

Mixture of Experts (MoEs) efficiently scales the capacity of LLMs by only activating a subset of the model parameters for each input (Jacobs et al., 1991; Fedus et al., 2022). These models are jointly trained without explicitly assigning individual tasks to each expert. The MoE approach has gained significant use in practice, resulting in multiple state-of-the-art language models in recent years (DeepSeek-AI et al., 2024; Jiang et al., 2024; Qwen et al., 2025). The success of MoE has motivated the development of techniques for mixing task-specific experts such as Ye et al. (2022)’s Task-level MoE and several adapter-based expert approaches (Pfeiffer et al., 2021; Wang et al., 2022; Caccia et al., 2023; Ponti et al., 2023; Fleshman et al., 2024; Huang et al., 2024; Zadouri et al., 2024). Unlike these existing works, we require zero data or training to select and combine our mixture of task-specific experts.

Task-Arithmetic isolates parameters responsible for a specific task by taking the difference between model weights before and after fine-tuning on the task (Ilharco et al., 2023; Fleshman & Van Durme, 2024). Fleshman & Van Durme (2024) demonstrate that these differences can be decomposed into LoRA adapters using the singular value decomposition (SVD). We note that these arithmetic-based adapters allow for arbitrary expert models to be used for adapter mixing, but our experiments in this work focus only on traditional LoRA models.

2.2 Adapter Selection and Merging

Model merging is an established technique for deriving a single multi-task model from individual task-specific models (Matena & Raffel, 2022; Wortsman et al., 2022; Stoica et al., 2024). One popular approach for model merging is to create a linear combination of the expert model weights (Wortsman et al., 2022; Chronopoulou et al., 2023; Ilharco et al., 2023; Fleshman et al., 2024). While simple, these weighted averages can be less effective when using LoRA experts, especially as the number of experts grows (Tang et al., 2024; Stoica et al., 2025). Tang et al. (2024) hypothesizes that LoRA adapters are less disentangled than full-rank fine-tuned models, causing deleterious interference when combined. Chronopoulou et al. (2023) and Fleshman et al. (2024) observed that LoRA averaging worked best when adapters for different tasks were equally initialized, an approach inapplicable when using experts from multiple sources. These challenges have led to many techniques for mitigating adapter interference (Ortiz-Jimenez et al., 2023; Yadav et al., 2023; Tang et al., 2024; Stoica et al., 2025; Yu et al., 2024). Ortiz-Jimenez et al. (2023) and Tang et al. (2024) suggest fine-tuning strategies to discourage entanglement during expert training. Yadav et al. (2023) propose TIES, which resolves adapter interference through an election process during merging. Stoica et al. (2025)’s KnOTS method performs an SVD over the set of adapters and merges them in the new shared subspace. Yu et al. (2024)’s DARE randomly drops and rescales parameters to create sparse approximations of experts. Our approach is compatible with most post-training strategies such as DARE, KnOTS, or TIES, but we find simple averaging with a sparse selection of experts reduces complexity and works well in our experiments.

Routing is the process of selecting relevant experts for a given task or query (Yadav et al., 2024). Yadav et al. (2024) refer to the combination of routing and merging as *MoErging* to highlight the similarity to MoEs while distinguishing methods using existing individual experts. We adopt their MoErging taxonomy which categorizes routing by the type of dataset used to learn routing, input and depth granularity, and the expert selection and aggregation mechanisms (Yadav et al., 2024). For example, μ -routing does not require training data because it aggregates all experts as a uniform linear combination for all queries and model layers (Caccia et al., 2023; Ostapenko et al., 2024). In contrast, Chronopoulou et al. (2023) and Fleshman et al. (2024) perform clustering over dense representations of training data and route new queries to a mixture of experts represented by similar clusters. Our work focuses on situations where there is no data available for learning to route. Therefore, we include μ -routing as one of the appropriate baselines for our MoErging experiments.

2.3 Problem Setting

We study training-free model MoErging methods using experts derived from standard LoRA training procedures (Hu et al., 2022; Ostapenko et al., 2024; Yadav et al., 2024). We assume access to T task-specific LoRAs for the same LM architecture without access to any data associated with their training. For a query q_t related to task t , we would like to select and merge a small number of experts (including the expert trained for t) capable of addressing q_t . We choose this setting to enable the usage of large and distinctly trained expert repositories that may not include associated data. We use LoRA for parameter efficiency (Hu et al., 2022) and choose sparse expert selection to minimize interference while increasing the chances of selecting the correct expert under imperfect routing (Tang et al., 2024; Yadav et al., 2024).

2.4 Arrow Routing

Arrow routing (Ostapenko et al., 2024) is the only existing work covered by the MoErging taxonomy (Yadav et al., 2024) that meets the requirements of our problem setting. Ostapenko et al. (2024) interpret the routing matrix from a standard MoE model as a collection of expert *prototypes* and construct their own prototypes using singular value decomposition (SVD). Recall that a rank- r LoRA for task t maps an input $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ to an output $\mathbf{x}^* \in \mathbb{R}^{d_{\text{out}}}$ with:

$$\mathbf{x}^* = W\mathbf{x} + B_t A_t \mathbf{x}, \quad (1)$$

where $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ are the original layer weights, and $B_t \in \mathbb{R}^{d_{\text{out}} \times r}$ and $A_t \in \mathbb{R}^{r \times d_{\text{in}}}$ are the low-rank LoRA matrices (Hu et al., 2022)². The SVD of the LoRA product:

$$U_t, S_t, V_t = \text{SVD}(B_t A_t), \quad (2)$$

produces the matrix containing the left singular vectors $U_t \in \mathbb{R}^{d_{\text{out}} \times r}$, the diagonal matrix of singular values $S_t \in \mathbb{R}^{r \times r}$, and the matrix of right singular vectors $V_t \in \mathbb{R}^{d_{\text{in}} \times r}$ such that:

$$U_t S_t V_t^T = B_t A_t. \quad (3)$$

The right singular vectors are eigenvectors of the LoRA parameter covariance matrix $(B_t A_t)^T B_t A_t$ and represent orthogonal directions of maximum variation induced by the LoRA expert in the space of input vectors \mathbf{x} . The top eigenvector $\mathbf{v}_t = V_t[:, 0]$ satisfies the following equation over possible unit length input vectors:

$$\mathbf{v}_t = \underset{\mathbf{x}, \|\mathbf{x}\|_2=1}{\text{argmax}} \|B_t A_t \mathbf{x}\|_2. \quad (4)$$

For this reason, Ostapenko et al. (2024) use \mathbf{v}_t as the prototype for expert t , as inputs similar to \mathbf{v}_t tend to produce activations with larger magnitudes. Let $P_\ell \in \mathbb{R}^{T \times d_{\text{in}}}$ be the routing matrix for layer ℓ with row $P_\ell[t] = \mathbf{v}_t$ representing the prototype for expert t . Ostapenko et al. (2024) then route an input vector \mathbf{x} of layer ℓ to the top- k experts using:

$$\text{experts} = \text{arg top-}k(|P_\ell \mathbf{x}|). \quad (5)$$

Ostapenko et al. (2024) use rank-4 LoRAs for their experiments, but we hypothesize that Arrow routing becomes less effective as the rank of experts increases. As the rank increases, the top eigenvector will capture a smaller percentage of the overall variation induced by that expert. We test this hypothesis and create an improved routing mechanism that leverages the entire spectrum of the LoRA covariance matrix without storing additional prototypes.

3 Spectral Routing

Spectral Routing (SPECTR) is our approach for dynamic token- and layer-wise composition of LoRAs, enabling improved multi-task adaptation of a base model without explicitly learning to route from data. SPECTR includes separate initialization and inference procedures.

²LoRA applies a fixed scalar to the matrix product, which we absorb into B for cleaner notation.

3.1 Spectral Alignment

We use an identical initialization procedure for each layer of the base model. Let B_t and A_t be the current layer’s LoRA parameters for expert t . We use the SVD from Equation 2 to compute U_t , S_t , and V_t . This can be done efficiently using low-rank or probabilistic algorithms (Halko et al., 2011; Nakatsukasa, 2019). Following from Equation 3, we reformulate the adapter parameters as:

$$B_t^* = U_t \text{ and } A_t^* = S_t V_t^T, \quad (6)$$

and discard the original parameters. Observe that A_t^* contains scaled eigenvectors of the covariance matrix, the first of which is the prototype used in Arrow routing (Ostapenko et al., 2024). Instead of storing separate prototypes, we capture all covariance structure directly in the new adapter representation. We repeat this process for each layer and adapter.

3.2 Routing Procedure

Given a set of aligned adapters $\{(B_1^*, A_1^*), (B_2^*, A_2^*), \dots, (B_T^*, A_T^*)\}$ and a token vector input to the current layer \mathbf{x} , we compute a low-rank representation \mathbf{h}_t for adapter t using:

$$\mathbf{h}_t = A_t^* \mathbf{x}. \quad (7)$$

Observe that if A_t^* was rank-1, then $\mathbf{h}_t \equiv P_\ell[t] \mathbf{x}$ from Equation 5. SPECTR takes advantage of the full rank of A_t^* and computes \mathbf{h}_t directly from the aligned adapter parameters without requiring the separate prototype. We then compute a routing score for adapter t using:

$$s_t = \|\mathbf{h}_t\|_2, \quad (8)$$

which measures the length of \mathbf{h}_t in the subspace induced by adapter t . Similar to Equation 4, we expect related vectors to maximize this score because the subspace represents directions of maximum variation for vectors used to train adapter t . We route to the top- k experts with:

$$\text{experts} = \arg \text{top-}k_{t \in \{1, \dots, T\}}(s_t). \quad (9)$$

3.3 Merging Procedure

SPECTR is flexible enough to allow for more advanced merging methods such as DARE (Yu et al., 2024), TIES (Yadav et al., 2023), or KnOTS (Stoica et al., 2025), but we use the same linear merging procedures as our baselines for a fair comparison. We discuss alternative merging options and considerations in Appendix A. Given the set of selected experts indexed $1, \dots, k$, we uniformly average their low-rank representations from Equation 7:

$$\mathbf{h}^* = \frac{1}{k} \sum_{i=1}^k \mathbf{h}_i. \quad (10)$$

Similarly, we uniformly merge the selected experts remaining LoRA parameters:

$$\hat{B} = \frac{1}{k} \sum_{i=1}^k B_i^*. \quad (11)$$

Like Ostapenko et al. (2024), a softmax of the routing scores could be used as an alternative to uniform weights. Finally, we compute the output of the current layer for token \mathbf{x} with:

$$\mathbf{x}^* = W\mathbf{x} + \hat{B}\mathbf{h}^*. \quad (12)$$

SPECTR performs these routing and merging procedures on a per-token and per-layer basis, enabling model expressivity on the scale of traditional MoE models.

4 Experiments

We measure the effectiveness of SPECTR as a routing strategy, testing our hypothesis that it improves routing accuracy over Arrow’s rank-1 prototypes. We measure multi-task performance across different methods and explore the confounding impact of task similarity.

4.1 Models and Adapters

We replicate our experiments across four popular instruction-tuned LMs chosen for their diversity in parameter count: Gemma-2B³ (Team et al., 2024), Phi-3.5B⁴ (Abdin et al., 2024), and Llama-3B⁵ and -8B⁶ (Grattafiori et al., 2024). LoRA adapters (Hu et al., 2022) are trained for each model using the peft library (Mangrulkar et al., 2022). We apply rank-8 adapters to all linear layers of each model. A main motivation for SPECTR is to allow the use of externally sourced adapters, so we use default hyperparameters instead of optimizing them for each routing method. We prompt each model using the corresponding template from the huggingface library (Wolf et al., 2020) and include our training details in Appendix B. Keeping with Ostapenko et al. (2024), we use $k = 4$ for Arrow and SPECTR top- k merging.

4.2 Datasets

We measure performance on several public datasets covering a diverse set of tasks: agnews,⁷ cola (Warstadt et al., 2019), dbpedia (Auer et al., 2007), hellaswag (Zellers et al., 2019), mnli (Williams et al., 2018), mrpc (Dolan & Brockett, 2005), qnli (Rajpurkar et al., 2016), qqp,⁸ and sst2 (Socher et al., 2013). We include each model’s out-of-sample accuracy before and after fine-tuning on the datasets in Appendix C.

We measure the cosine similarity between the LoRA weights of different tasks as a proxy for task similarity. Figure 2 displays the pairwise similarity scores averaged over the four LMs. We include the cosine scores for individual models in Appendix D.

We compute an overall similarity score for each task by averaging its pairwise cosine scores (Figure 3). The score for each task is consistent across the four LMs, with a slight drop for Llama-8B likely due to its increased dimensionality. The dbpedia, mnli, and qqp adapters are the least similar to other tasks, while sst2, cola, and mrpc are the most similar.

agnews -		0.40	0.22	0.37	0.25	0.41	0.33	0.25	0.36
cola -	0.40		0.26	0.47	0.31	0.62	0.44	0.33	0.50
dbpedia -	0.22	0.26		0.24	0.17	0.27	0.22	0.17	0.24
hellaswag -	0.37	0.47	0.24		0.28	0.48	0.38	0.28	0.42
mnli -	0.25	0.31	0.17	0.28		0.32	0.28	0.21	0.29
mrpc -	0.41	0.62	0.27	0.48	0.32		0.44	0.34	0.49
qnli -	0.33	0.44	0.22	0.38	0.28	0.44		0.28	0.39
qqp -	0.25	0.33	0.17	0.28	0.21	0.34	0.28		0.30
sst2 -	0.36	0.50	0.24	0.42	0.29	0.49	0.39	0.30	
	agnews	cola	dbpedia	hellaswag	mnli	mrpc	qnli	qqp	sst2

Figure 2: Pairwise cosine similarities between task adapters averaged across four models.

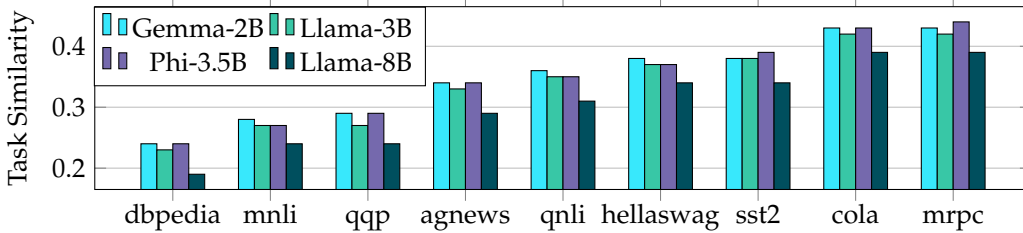


Figure 3: The average cosine similarity between a specific task and all other tasks.

4.3 Routing Performance

We first compare strategies by framing routing as a classification problem. Arrow and SPECTR both compute per-token routing scores at each layer. We label each token from an

³<https://huggingface.co/google/gemma-2-2b-it>

⁴<https://huggingface.co/microsoft/Phi-3.5-mini-instruct>

⁵<https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct>

⁶<https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct>

⁷http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

⁸<https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs>

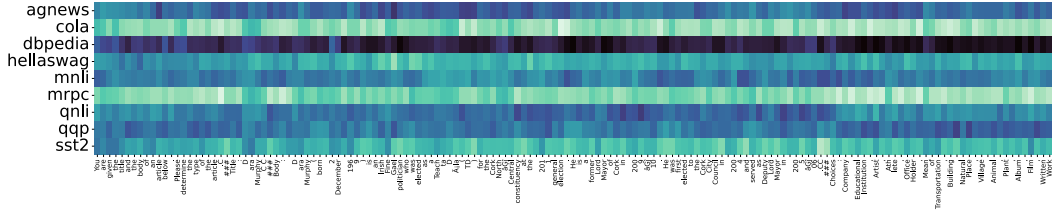


Figure 4: Per-token routing distribution using SPECTR for a sequence of tokens from dbpedia averaged across model layers. Darker areas indicate a higher average routing score.

out-of-sample sequence as belonging to the adapter trained on the corresponding dataset. We visualize the classification of an example sequence in Figure 4, where routing scores have been averaged across all model layers.⁹ We note that a perfect routing accuracy at the token level would be unexpected, as only a subset of tokens are consistent across examples from the same dataset. We therefore focus on the relative difference in accuracies between the two approaches. We mark each token correct if the ground truth

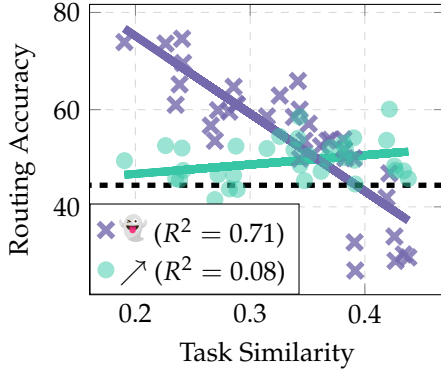


Figure 5: SpectR (🤖) is more accurate at routing than Arrow (↗) except for datasets with higher task similarity. The black dashed line indicates random routing accuracy.

adapter is selected among the top-4 experts and record the routing accuracy in Table 1. SPECTR results in an average top-4 accuracy gain of approximately 4 percentage points, with individual task gains of up to 24 points. We include top-1 accuracies in Appendix E.

We plot routing accuracies against each task similarity score in Figure 5 and see that SPECTR’s largest accuracy gains occur for the more unique tasks such as dbpedia. SPECTR seems to systematically choose alternative adapters for the most similar tasks such as mrpc and cola, resulting in accuracies below random chance. We will explore if these inaccuracies are consequential on downstream performance in Section 4.5, or if SPECTR’s choices still perform well due to selected adapters having been trained for similar tasks. Arrow’s accuracy appears independent of task similarity, albeit at a significantly lower average accuracy.

	Gemma-2B		Llama-3B		Phi-3.5B		Llama-8B	
	↗	🤖	↗	🤖	↗	🤖	↗	🤖
agnews	51.7	58.5	54.1	65.9	54.5	62.9	52.5	61.2
cola	48.4	33.9	47.4	30.2	53.7	41.9	49.6	32.7
dbpedia	47.5	69.6	52.1	74.6	52.6	73.6	49.5	73.9
hswag	50.1	51.8	52.2	53.8	50.0	53.6	48.6	49.9
mnli	43.7	59.6	46.6	59.9	38.0	56.7	45.8	61.0
mrpc	47.5	28.9	45.8	29.6	60.1	46.9	44.7	26.9
qnli	47.3	52.5	50.6	57.1	45.4	53.6	52.0	58.6
qqp	46.5	64.7	43.6	61.0	41.5	53.6	45.6	65.2
sst2	51.8	50.0	54.2	50.0	53.1	53.9	58.5	60.0
AVG	48.3	52.2	49.6	53.6	49.9	55.2	49.6	54.4

Table 1: Top-4 routing accuracies. SPECTR(🤖) outperforms Arrow (↗) on all four models.

⁹We acknowledge the token labels are small, but keep them for curious readers.

4.4 Rank vs. Routing Effectiveness

We recall our hypothesis that Arrow prototypes become less meaningful as the adapter rank increases, reducing the sensitivity of routing scores to anything other than the single direction of maximum variation for each task. We quantify this effect by measuring the ratio of the routing score for the ground truth adapter against the highest routing score among all adapters. A ratio of 1 indicates perfect token routing. We plot these ratios for both methods as a function of adapter rank in Figure 6. The SPECTR routing scores from Equation 8 take advantage of the extra dimensions in the intermediate representation as the rank increases, leading to better discrimination between adapters. In contrast, Arrow uses the rank-1 prototype, which we confirm results in decreased ratio scores at higher ranks.

4.5 Multi-Task Performance

Now that we have established SPECTR improves routing, we confirm the strategy results in good multi-task performance in practice. We compare the base instruction-tuned model with Arrow and SPECTR, and we include μ -routing as an additional strong baseline. Following similar work, we measure the normalized accuracy to control for the difficulty of the task (Yadav et al., 2023; Ilharco et al., 2023; Stoica et al., 2025). Normalized accuracy divides the achieved performance by the accuracy of the oracle model fine-tuned for the specific task.

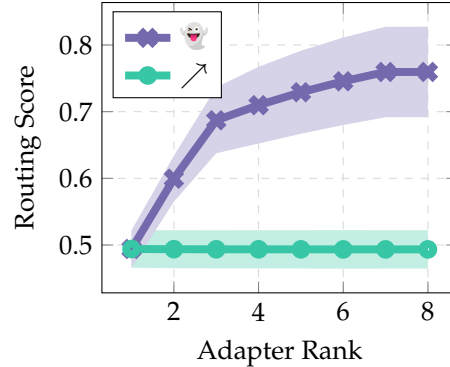


Figure 6: SPECTR (👤) becomes increasingly more effective than Arrow (↗) at selecting the correct adapter as adapter rank increases.

We report the normalized accuracies in Table 2. All three routing methods outperform the baseline models and SPECTR results in the highest average accuracy in all cases.

The magnitude of individual performance differences varied across models and tasks, with SPECTR’s largest gain over μ and Arrow being 15.1 and 10.7 percentage points respectively. SPECTR performed well on tasks even when routing accuracy was low, suggesting that low routing accuracies were partially caused by different datasets having similar tasks. High task similarity resulted in good performance even though the ground truth adapter was used less often. Interestingly, all routing methods and LMs perform poorly on hellaswag, even though the individual adapters fine-tuned for that task do well (Appendix C). This could indicate task interference (Ortiz-Jimenez et al., 2023; Tang et al., 2024), possibly caused by the simple linear merge used by all three methods in our experiments (Appendix A).

5 Trade-offs

While better routing and multi-task performance are desirable, there is no free lunch when choosing between methods (Wolpert & Macready, 1997). We discussed the impact of adapter rank in Section 4.4, and here we highlight three additional dimensions to consider when choosing between training-free strategies: storage cost, VRAM usage, and interference.

Storage costs for μ -routing and SPECTR are identical if the experts are kept isolated for circumstances such as access-control (Fleshman et al., 2024). Otherwise, μ -routing can have zero overhead by merging all experts into the base model ahead of time. SPECTR represents the experts in a different form after spectral alignment (Section 3.1), but the resulting matrices are the same size and produce equivalent products. Arrow routing requires the same library but includes the additional overhead of storing prototype vectors for each expert at every layer in the model (Ostapenko et al., 2024). The additional overhead is equivalent to storing an extra adapter for every $2r$ experts when using LoRAs of rank r .









		agnews	cola	dbpedia	hswag	mnli	mrpc	qnli	qqp	sst2	AVG
Gem-2B	Base	86.2	92.7	88.4	38.3	62.2	90.4	73.2	84.9	95.0	79.0
	μ	87.6	93.2	95.4	33.1	69.9	89.1	60.8	90.1	96.9	79.6
		87.9	92.3	95.2	29.8	53.2	91.9	53.1	91.4	96.6	76.8
		88.2	91.3	96.8	29.7	69.8	90.8	63.8	92.0	96.7	79.9
Llam-3B	Base	79.7	65.3	86.1	39.0	38.0	63.3	72.9	53.5	88.3	65.1
	μ	84.3	90.7	94.1	53.0	55.6	86.4	83.2	82.9	95.7	80.7
		85.3	88.2	95.1	53.9	49.7	84.6	83.8	90.4	94.4	80.6
		84.8	89.4	96.1	52.4	55.3	87.4	83.3	91.9	95.7	81.8
Phi-3.5B	Base	78.9	95.0	94.4	65.6	80.6	94.1	60.8	86.5	94.0	83.3
	μ	84.8	96.2	97.3	69.3	86.3	94.0	59.2	91.9	95.7	86.1
		85.0	96.2	97.7	73.9	88.2	92.5	56.9	92.1	95.9	86.5
		86.0	96.2	98.2	74.3	88.3	94.0	55.9	92.4	95.8	86.8
Llam-8B	Base	88.1	84.5	94.0	52.4	57.3	91.9	81.3	75.9	93.4	79.9
	μ	93.3	87.4	99.1	41.5	73.0	91.8	90.2	94.5	97.7	85.3
		90.2	93.5	97.0	56.4	71.0	93.8	89.0	91.4	97.2	86.6
		88.4	93.2	98.0	56.6	72.2	93.8	90.2	92.7	97.3	86.9

Table 2: Multi-Task performance across models and methods.

VRAM requirements are lowest for μ -routing because inputs are processed by a single merged adapter. Arrow and SPECTR require that all adapters be loaded in GPU memory as selection and merging occur on the fly for each token at each layer. Arrow takes advantage of its rank-1 prototypes to reduce computation in the routing step, while SPECTR requires all adapters to compute low-rank representations before selecting the top-k experts. For a layer with hidden dimension h , routing with SPECTR requires the FLOPS equivalent to a forward pass of the layer for every h/r experts. Arrow allows for h experts at the same cost.

Interference between adapters is most likely when using the dense selection of μ -routing (Caccia et al., 2023; Ostapenko et al., 2024). Both Arrow and SPECTR help mitigate interference by using sparse routing and are compatible with additional mitigations such as DARE (Yu et al., 2024), TIES (Yadav et al., 2023), or KnOTS (Stoica et al., 2025) if necessary.

These considerations can help practitioners choose a strategy under various constraints. Our experiments demonstrate superior performance with SPECTR, but Arrow might be preferable for its increased GPU efficiency as long as the adapters are of low enough rank. μ -routing remains a good option for smaller adapter libraries where routing is unnecessary.

6 Conclusion

In this work, we studied training-free model MoErting approaches for integrating externally trained LoRA experts into expressive models capable of performing well in a multi-task environment. We identified limitations with existing strategies, such as interference and sensitivity to adapter rank. These challenges led to our development of SPECTR, a new approach for per-token, per-layer routing, requiring zero additional data or training to employ. We conducted experimentation across a diverse set of tasks and popular LMs, demonstrating the ability of SPECTR to leverage the extra dimensions of higher rank adapters for increased routing accuracy over alternatives. SPECTR improved routing accuracy by 4 percentage points on average, leading to individual increases in task performance of up to 15%. We also explored the impact of task similarity on routing accuracy, and we found the results for each dataset were consistent across all four LMs under evaluation. SPECTR achieved the highest accuracies on the more unique tasks. We found that the alternate adapters selected by SPECTR for highly similar tasks still performed well regardless of imprecise routing. Finally, we discussed trade-offs to consider when choosing a routing strategy, including storage costs, VRAM requirements, and potential adapter interference. Overall, we hope the effectiveness of SPECTR will allow for broader use of the abundance of expert models available for composition and improve the multi-task capabilities and performance of LMs.

References

- Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, et al. Phi-3 technical report: A highly capable language model locally on your phone, 2024. URL <https://arxiv.org/abs/2404.14219>.
- Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux (eds.), *The Semantic Web*, pp. 722–735, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-76298-0.
- Ankur Bapna and Orhan Firat. Simple, scalable adaptation for neural machine translation. 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. 1538–1548, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1165. URL <https://aclanthology.org/D19-1165/>.
- Joshua Belofsky. Token-level adaptation of lora adapters for downstream task generalization. In *Proceedings of the 2023 6th Artificial Intelligence and Cloud Computing Conference, AICCC '23*, pp. 168–172, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400716225. doi: 10.1145/3639592.3639615. URL <https://doi.org/10.1145/3639592.3639615>.
- Rickard Brüel-Gabrielsson, Jiacheng Zhu, Onkar Bhardwaj, Leshem Choshen, Kristjan Greenewald, Mikhail Yurochkin, and Justin Solomon. Compress then serve: Serving thousands of lora adapters with little overhead, 2024. URL <https://arxiv.org/abs/2407.00066>.
- Lucas Caccia, Edoardo Ponti, Zhan Su, Matheus Pereira, Nicolas Le Roux, and Alessandro Sordoni. Multi-head adapter routing for cross-task generalization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=qcQhBli5Ho>.
- Alexandra Chronopoulou, Matthew Peters, Alexander Fraser, and Jesse Dodge. Adapter-Soup: Weight averaging to improve generalization of pretrained language models. In Andreas Vlachos and Isabelle Augenstein (eds.), *Findings of the Association for Computational Linguistics: EACL 2023*, pp. 2054–2063, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-eacl.153. URL <https://aclanthology.org/2023.findings-eacl.153/>.
- DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyan Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Gao, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, et al. Deepseek-v3 technical report, 2024. URL <https://arxiv.org/abs/2412.19437>.

- Tim Dettmers and Luke Zettlemoyer. The case for 4-bit precision: k-bit inference scaling laws. In *Proceedings of the 40th International Conference on Machine Learning, ICML'23*. JMLR.org, 2023.
- Shizhe Diao, Tianyang Xu, Ruijia Xu, Jiawei Wang, and Tong Zhang. Mixture-of-domain-adapters: Decoupling and injecting domain knowledge to pre-trained language models' memories. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5113–5129, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.280. URL <https://aclanthology.org/2023.acl-long.280/>.
- William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL <https://aclanthology.org/I05-5002/>.
- Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. Depgraph: Towards any structural pruning. *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. 23(1), January 2022. ISSN 1532-4435.
- William Fleshman and Benjamin Van Durme. Re-adapt: Reverse engineered adaptation of large language models, 2024. URL <https://arxiv.org/abs/2405.15007>.
- William Fleshman, Aleem Khan, Marc Marone, and Benjamin Van Durme. Adapterswap: Continuous training of llms with data removal and access-control guarantees. In *Proceedings of Conference on Applied Machine Learning in Information Security (CAMLIS) 2024*, 2024. URL <https://arxiv.org/abs/2404.08417>.
- Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. Knowledge distillation: A survey. *Int. J. Comput. Vision*, 129(6):1789–1819, June 2021. ISSN 0920-5691. doi: 10.1007/s11263-021-01453-z. URL <https://doi.org/10.1007/s11263-021-01453-z>.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, et al. The llama 3 herd of models, 2024. URL <https://arxiv.org/abs/2407.21783>.
- N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2): 217–288, 2011. doi: 10.1137/090771806. URL <https://doi.org/10.1137/090771806>.
- Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically, 2017. URL <https://arxiv.org/abs/1712.00409>.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL <https://arxiv.org/abs/1503.02531>.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Oriol Vinyals, Jack W. Rae, and Laurent Sifre. Training compute-optimal large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS '22*, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Larous-silhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, 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, 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>.
- Robert Jacobs, Michael Jordan, Steven Nowlan, and Geoffrey Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3:79–87, 03 1991. doi: 10.1162/neco.1991.3.1.79.
- Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, L  lio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Th  ophile Gervet, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. Mixtral of experts, 2024. URL <https://arxiv.org/abs/2401.04088>.
- Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL <https://arxiv.org/abs/2001.08361>.
- Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. I-bert: Integer-only bert quantization. *International Conference on Machine Learning (Accepted)*, 2021.
- Prasanth Kolachina, Nicola Cancedda, Marc Dymetman, and Sriram Venkatapathy. Prediction of learning curves in machine translation. In Haizhou Li, Chin-Yew Lin, Miles Osborne, Gary Geunbae Lee, and Jong C. Park (eds.), *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22–30, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL <https://aclanthology.org/P12-1003/>.
- Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL <https://aclanthology.org/2021.emnlp-main.243/>.
- Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *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. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL <https://aclanthology.org/2021.acl-long.353/>.
- Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th*

- Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.8. URL <https://aclanthology.org/2022.acl-short.8/>.
- Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems*, 2023.
- Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. <https://github.com/huggingface/peft>, 2022.
- Michael Matena and Colin Raffel. Merging models with fisher-weighted averaging. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS ’22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.
- Yuji Nakatsukasa. The low-rank eigenvalue problem, 2019. URL <https://arxiv.org/abs/1905.11490>.
- Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=0A9f2jZDGW>.
- 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 *Forty-first International Conference on Machine Learning*, 2024. URL <https://openreview.net/forum?id=0ZFWfeVsaD>.
- Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty (eds.), *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/>.
- Edoardo Maria Ponti, Alessandro Sordoni, Yoshua Bengio, and Siva Reddy. Combining parameter-efficient modules for task-level generalisation. In Andreas Vlachos and Isabelle Augenstein (eds.), *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 687–702, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.49. URL <https://aclanthology.org/2023.eacl-main.49/>.
- Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL <https://arxiv.org/abs/2412.15115>.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL <https://aclanthology.org/D16-1264/>.
- Sheng Shen, Dong Zhen, Jiayu Ye, Linjian Ma, Zhewei Yao, Asghar Gholami, Michael Mahoney, and Kurt Keutzer. Q-bert: Hessian based ultra low precision quantization of bert. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:8815–8821, 04 2020. doi: 10.1609/aaai.v34i05.6409.

- Tal Shnitzer, Anthony Ou, Mírian Silva, Kate Soule, Yuekai Sun, Justin Solomon, Neil Thompson, and Mikhail Yurochkin. Large language model routing with benchmark datasets, 2023. URL <https://arxiv.org/abs/2309.15789>.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL <https://aclanthology.org/D13-1170/>.
- George Stoica, Daniel Bolya, Jakob Brandt Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. Zipit! merging models from different tasks without training. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=LEYUkvUdhq>.
- 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>.
- 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=iynRvVVAH>.
- Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, et al. Gemma 2: Improving open language models at a practical size, 2024. URL <https://arxiv.org/abs/2408.00118>.
- Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5744–5760, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.388. URL <https://aclanthology.org/2022.emnlp-main.388/>.
- Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019. doi: 10.1162/tacl_a.00290. URL <https://aclanthology.org/Q19-1040/>.
- Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL <https://aclanthology.org/N18-1101/>.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface’s transformers: State-of-the-art natural language processing, 2020. URL <https://arxiv.org/abs/1910.03771>.

- D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997. doi: 10.1109/4235.585893.
- Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 23965–23998. PMLR, 17–23 Jul 2022. URL <https://proceedings.mlr.press/v162/wortsman22a.html>.
- Xiaoxia Wu, Zhewei Yao, Minjia Zhang, Conglong Li, and Yuxiong He. Extreme compression for pre-trained transformers made simple and efficient. 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=xNeAhc2CNA1>.
- 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>.
- Qinyuan Ye, Juan Zha, and Xiang Ren. Eliciting and understanding cross-task skills with task-level mixture-of-experts. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 2567–2592, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.189. URL <https://aclanthology.org/2022.findings-emnlp.189/>.
- 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 *ICML*, 2024. URL <https://openreview.net/forum?id=fq0NaiU8Ex>.
- Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=EvDeilV7qc>.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL <https://aclanthology.org/P19-1472/>.

A Alternate Merging Procedures

As mentioned in Section 3.3, we structure Equation 10 and Equation 11 to make comparisons with baselines more fair. However, merging the experts in two separate steps is known to cause issues when the adapters were initialized with different seeds (Chronopoulou et al., 2023; Fleshman et al., 2024), as the merging procedure is sensitive to the ordering of rows and columns in each matrix. The issue is alleviated by merging the final result of each expert instead of doing separate merges:

$$\hat{\mathbf{x}} = \frac{1}{k} \sum_{i=1}^k B_i^* \mathbf{h}_i, \quad (13)$$

and using $\hat{\mathbf{x}}$ in place of $\hat{\mathbf{B}}\mathbf{h}^*$ in Equation 12. This removes the need for experts to have the same permutation of rows and columns and allows for seamless merging of adapters with different ranks, as the resulting products will have the same dimensions. This simple average is also easily generalized to more sophisticated merging algorithms.

B LoRA Training

All adapters were trained on a single Nvidia A100 GPU with 80GB of memory. We used rank-8 LoRAs with a LoRA $\alpha = 16$, LoRA dropout of 0.05, and applied adapters to all linear layers. We used supervised fine-tuning and trained with a batch size of 8 using a learning rate of $1e-4$ with a constant learning rate scheduler.

C Adapter Performance

We computed the raw out-of-sample accuracy achieved by each model before and after fine-tuning the model on the in-sample portion of each dataset (Table 3).

	Gemma-2B		Llama-3B		Phi-3.5B		Llama-8B	
	Base	FT	Base	FT	Base	FT	Base	FT
agnews	80.6	93.5	75.3	94.5	74.9	94.9	83.0	94.2
cola	79.0	85.2	55.5	85.0	81.6	85.9	73.7	87.2
dbpedia	87.5	99.0	85.2	99.0	93.6	99.2	93.2	99.1
hswag	34.4	89.9	34.2	87.6	60.1	91.6	48.3	92.2
mnli	54.8	88.1	33.8	88.9	72.8	90.3	51.5	89.9
mrpc	75.7	83.7	54.1	85.5	77.8	82.7	75.0	81.6
qnli	67.5	92.2	67.5	92.6	56.5	93.0	75.6	93.0
qqp	75.7	89.2	47.8	89.4	77.5	89.6	68.2	89.9
sst2	90.2	94.9	84.1	95.2	89.9	95.6	88.7	95.0

Table 3: Task Performances before and after fine-tuning a task-specific adapter.

D Task Similarities

We display the cosine similarities between adapters for all models in Figure 7.

E Top-1 Routing

We show the top-1 routing accuracies on out-of-sample data in Table 4.









	Gemma-2B		Llama-3B		Phi-3.5B		Llama-8B	
								
agnews	18.4	21.9	20.5	27.0	20.5	25.9	19.2	23.7
cola	14.0	7.5	13.2	7.0	14.7	11.1	14.2	7.1
dbpedia	16.1	36.0	19.7	39.6	20.0	41.6	16.6	38.5
hswag	17.2	17.8	18.7	19.2	19.3	21.5	16.3	15.7
mnli	11.0	20.3	12.8	21.6	10.5	21.3	12.5	22.1
mrpc	11.6	4.9	12.5	6.0	18.0	11.4	11.4	5.6
qnli	13.4	14.4	14.0	15.6	14.4	17.2	15.6	17.4
qqp	15.9	26.3	11.3	21.8	12.9	20.4	12.8	26.1
sst2	16.3	15.4	18.1	14.7	17.8	19.1	20.3	19.4
AVG	14.9	18.3	15.6	19.2	16.5	21.1	15.4	19.5

Table 4: Top-1 routing accuracies.

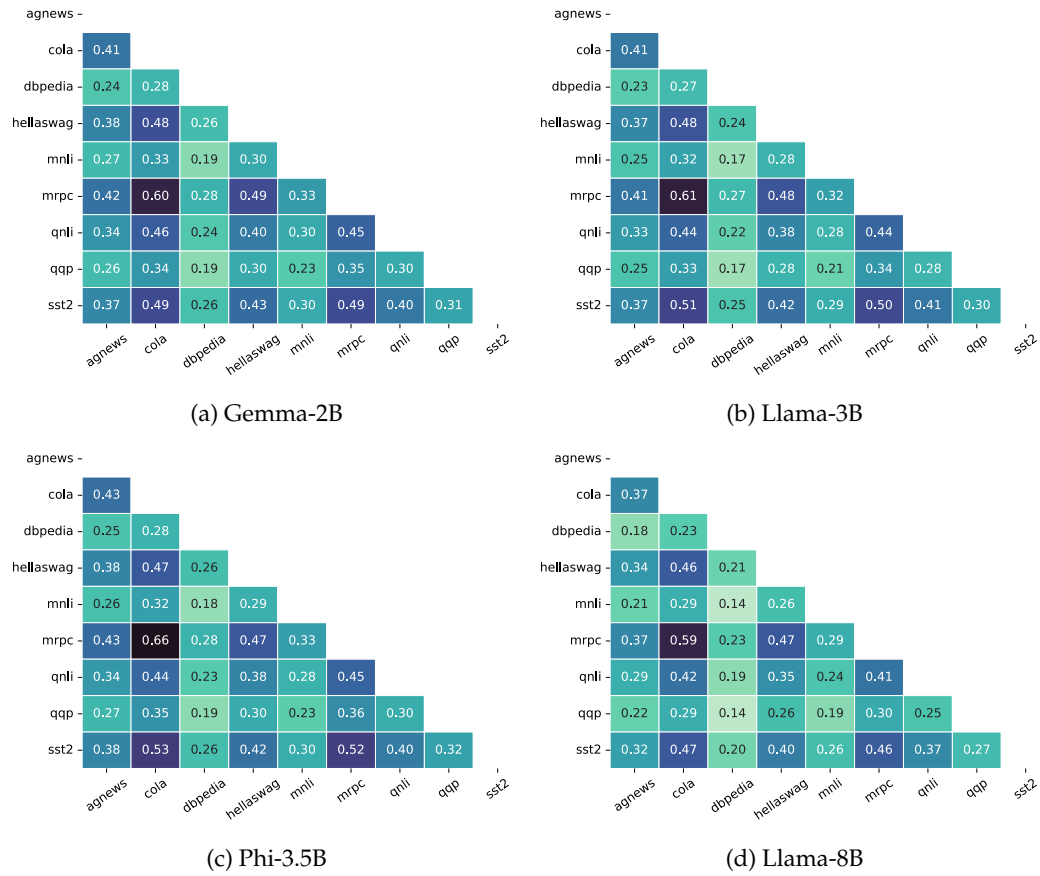


Figure 7: Cosine Similarities for all models.