Model Breadcrumbs: Scalable Upcycling of Finetuned Foundation Models via Sparse Task Vectors Merging

MohammadReza Davari [1] **Eugene Belilovsky** [1]

Abstract

The rapid development of AI systems has been greatly influenced by foundation models. Typically, these models are fine-tuned for specific tasks, leading to numerous task-specific versions. This paper addresses the challenge of merging and upcycling these fine-tuned models. We introduce Model Breadcrumbs, a simple method using sparse weight trajectories to guide model adaptation within a pre-trained model's weight space. Our approach improves performance across multiple tasks without the need for hyperparameter tuning for each new task. Extensive experiments, involving various models, tasks, and modalities, demonstrate that Model Breadcrumbs provides an efficient and effective solution for creating and updating multi-task models, promoting a community-driven effort for updatable machine learning. [1]

1. Introduction

Foundational models (Bommasani et al., 2021) have become instrumental across multiple domains due to their extensive scale, generality, and capacity to generalize to vast datasets. They have driven advancements in NLP (Radford et al., 2018; 2019; Devlin et al., 2018; Marks et al., 2024; Farahnak et al., 2021; Davari et al., 2020; Davari, 2020), computer vision (Radford et al., 2021; Ramesh et al., 2021; Luo et al., 2020), and other fields (Rives et al., 2021; Yin et al., 2020; Rothchild et al., 2021) through fine-tuning for specific tasks. However, scaling these models to perform multiple tasks presents challenges, particularly in practical scenarios where joint training is infeasible due to data privacy or computational constraints (Cossu et al., 2022).

[1]Concordia University and Mila – Quebec AI Institute. Correspondence to: MohammadReza Davari <mohammadreza.davari@mail.concordia.ca>.

Published at ICML 2024 Workshop on Foundation Models in the Wild. Copyright 2024 by the author(s).

[1]The code to reproduce our results is publicly available at: https://github.com/rezazzr/breadcrumbs

The widespread adoption of foundational models has led to a proliferation of fine-tuned models with standardized architectures (Bommasani et al., 2021). These models, while useful for inference, remain largely untapped beyond their conventional uses. Recent developments in neural network weight averaging techniques have sought to repurpose these fine-tuned models, enabling scalable and efficient model merging (Ramé et al., 2022; Ilharco et al., 2022a; Choshen et al., 2022; Yadav et al., 2023).

Task Arithmetic, introduced by Ilharco *et al.* (2022a), refines a foundation model by averaging the differences between multiple fine-tuned models and the foundation model. Despite its potential, this method faces limitations with numerous tasks due to its reliance on hyperparameter tuning and the accumulation of noise.

To address these challenges, we propose Model Breadcrumbs, a method designed to tackle scalability, reduce noise in merging tasks, and generalize hyperparameters effectively. Model Breadcrumbs constructs multi-task models from pre-existing fine-tuned models, overcoming the limitations of existing methods (see Figure 1). We demonstrate its effectiveness through extensive evaluations, showing that Model Breadcrumbs yields competitive multi-task models with robust performance across varying tasks. Our key contributions are: 1. Introducing a simple and scalable approach for merging models and reusing pre-existing fine-tuned models to construct multi-task models, often outperforming their individual fine-tuned counterparts. 2. We empirically show the robustness of our approach to hyperparameter variations and its ability to generalize with the increasing number of tasks.

2. Related Work

Recent studies in the literature have explored the merging of models trained from scratch with different initializations (Ainsworth et al., 2022; Stoica et al., 2023). One of the main challenges in this type of model merging is aligning the models before the actual merger. Therefore, research in this branch primarily focuses on finding permutations between networks to bring them into alignment with a reference model, enabling the subsequent merger of the

two models in weight space. Our work, on the other hand, distinguishes itself from this line of research, as we concentrate on the model merging of networks that share the same initialization, specifically initialized by a foundation model.

Neyshabur *et al.* (2020) highlighted the benefits of linearly interpolating two fine-tuned models originating from the same pre-trained model. They showed that this technique often yields a model that outperforms both of the original fine-tuned models. This discovery sparked subsequent investigations into the merging of fine-tuned models derived from a single foundation model, exploring its potential and practical applications. Wortsman *et al.* (2022) demonstrated that models fine-tuned on the same dataset with different hyperparameters can be combined together using a weighted average to yield an overall higher performing model. Unlike our work they did not consider merging models from different datasets and tasks. Choshen *et al.* (2022) merged models from multiple trained models in order to create a better pretrained model to be used for downstream tasks. Unlike our work they do not demonstrate or study the creation of multi-task ability through the merging. Matena and Raffel (2022) considered merging of multiple fine-tuned models originating from the same pre-trained model, trained on diverse datasets. The merger operation combines a series of fine-tuned models using a weighted average determined by the Fisher information matrix (Myung, 2003). However, computing the Fisher information matrix, as well as finding other required hyperparameters for this approach, becomes increasingly computationally expensive as the number of models to be merged grows. Therefore, it faces challenges when applied at scale. In contrast, our approach is computationally efficient, and as we will show in Section 4, its hyperparameters exhibit the ability to generalize to the scenarios where a large number of models are to be merged.

A related study to ours is conducted by Ilharco *et al.* (2022a), introducing a method named Task Arithmetic for model merging. Their approach begins by forming *Task Vectors*, representing the weight differences between pre-trained and fine-tuned weights for each task. The merged model's weights are then obtained by adding a scaled sum of these task vectors to the pre-trained weights. However, their approach necessitates a validation set for each new task, which adds complexity and computational overhead, coupled with an increasing accumulation of noise as more tasks are merged to the foundation model. A concurrent study by Yadav *et al.* (2023) presents a method named TIES. Like the Task Arithmetic method (Ilharco et al., 2022a), TIES initially constructs a set of Task Vectors. These vectors undergo a masking process to eliminate interfering weights, identified as a percentage of overall weights with low magnitudes. The remaining unmasked weights undergo a sign alignment operation to determine their polarity. Finally, a scaled sum merges the task vectors with the pre-trained model. Our approach differs from TIES in two key aspects. Firstly, we apply masking to both very large and small magnitude weights of the task vectors to minimize interference, whereas TIES focuses solely on small magnitude weights. Secondly, our masking strategy employs layer-wise masking as opposed to overall masking. Notably, in the context of task vectors, overall masking of small magnitude weights typically targets weights in the early layers (Matena & Raffel, 2022).

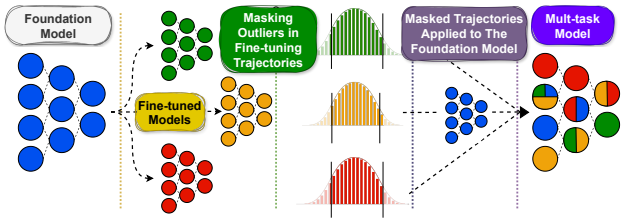


Figure 1. Method overview. Starting with a fine-tuned foundational model, we form task vectors by subtracting pre-trained model weights from each fine-tuned model. We then apply a masking operation to each layer, filtering out outliers and small values. Finally, the masked task vectors are aggregated and combined with the pre-trained model to create a unified multi-task model.

3. Model Breadcrumbs Framework

The Model Breadcrumbs framework is designed to enable the construction of multi-task models from pre-existing fine-tuned foundation models without the need for further training. The central idea is to merge models and aggregate valuable *knowledge* for the resulting multi-task model while filtering out potential harmful perturbations that could impact its performance. This section provides an overview of the process for acquiring and merging Model Breadcrumbs.

To start generating Model Breadcrumbs, we begin with a pre-trained foundation model that has undergone fine-tuning for various auxiliary tasks. Denoting the weights of the foundation model as θ, after fine-tuning on a specific task t, the weights are transformed into θ'_t. The initial step involves creating *task vectors* by calculating the weight differences between θ'_t and θ, resulting in θ^d_t.

$$\theta^d_t = \theta'_t - \theta \tag{1}$$

Note that θ^d_t contains both (a) large outliers, indicating substantial deviations from the pre-trained starting point, and (b) negligible differences representing minor perturbations from the foundation model's weights. The presence of these extremes can impact the effectiveness of the resulting multi-task model upon merging. To address this concern, we implement a masking process that filters out both large outliers and small perturbations.

Table 1. The evaluation of the above merging strategies over 8 tasks using ViT-B-32 reveals the advantage of Breadcrumbs over other merging methods. Note that not only Fisher Merging (Matena & Raffel, 2022) lags behind both Task Arithmetic (Ilharco et al., 2022a) and Model Breadcrumbs, it also requires significantly more computational resources.

Method	Avg. Normalized Acc.
Breadcrumbs	**83.35**
Task Arithmetic	77.66
Fisher Merging	75.11
Task Arithmetic$^{\text{w/ random masking}}$	74.00

In each layer L, we mask out the extreme tails of the absolute magnitude distribution of θ_t^d, using γ and β as thresholds for the right and left tails, respectively. Let w_i^L represent the index of the weights sorted by their absolute magnitude in layer L and i the order in the sort (lowest to highest). The mask $m_L^{\beta,\gamma}$ for the layer L is defined as:

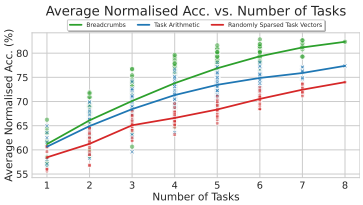
$$m_L^{\beta,\gamma}[w_i] = \begin{cases} 0 & \text{if } i \leq \beta \text{ or } i \geq \gamma \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

The masked weights are set to zero in θ_t^d or returned to their respective pre-training weights in θ_t'. Aggregating $m_L^{\beta,\gamma}$ over all layers, for task t, results in the final mask $m_t^{\beta,\gamma}$. Next, we apply the mask $m_t^{\beta,\gamma}$ to the task vectors θ_t^d. We now have a set of weight differences that define a trajectory within the weight space of the foundation model. Traversing this trajectory allows us to effectively transfer the knowledge accumulated during fine-tuning across tasks, while filtering out the harmful perturbations. For a total of T tasks, we assemble a multi-task model θ^* by following the trajectories defined by the Model Breadcrumbs with a specific strength parameter α. The formation of this multi-task model is expressed in Eq. 3. Furthermore, Appendix A describes the pseudocode of our algorithm.

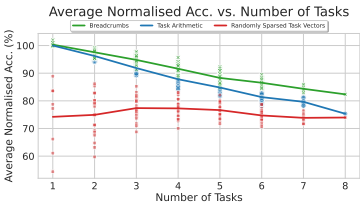
$$\theta^* = \theta + \alpha \sum_{t \in T} m_t^{\beta,\gamma}.\theta_t^d \tag{3}$$

4. Experiments

In this section, we evaluate the Model Breadcrumbs framework through a series of experiments focusing on three key aspects: 1. **Scalability and Performance**: Incrementally adding up to 8 vision tasks to assess the performance of merged Model Breadcrumbs. 2. **Hyperparameter Generalization**: We explore how the hyperparameters introduced by Model Breadcrumbs—α, β, and γ—generalize over the number of datasets. 3. **Target Task Improvement**: Enhancing a fine-tuned model's performance on a target task by merging related tasks into it.



(a) At each point, evaluation is performed over all 8 tasks.



(b) At each point, evaluation is performed only over the observed tasks.

Figure 3. The solid line is the avg. normalized accuracy across all evaluation points. Each data point corresponds to an experiment involving a subset of the 8 tasks under study. Notably, it is evident that the Model Breadcrumbs (with 91% sparsity), consistently outperform the Task Arithmetic (Ilharco et al., 2022a). Specifically, in the experiment involving all eight tasks, the Model Breadcrumbs outperform the Task Arithmetic by a substantial margin of 5.7%.

We follow the benchmarks and settings used by Ilharco *et al.* (2022a) for meaningful comparison. Results are presented using normalized accuracy, calculated as the ratio of accuracy achieved by the merged model to that of the fine-tuned model:

$$\text{Normalized Accuracy} = \frac{\text{Accuracy of Merged Model}}{\text{Accuracy of Fine-tuned Model}}$$

A fine-tuned model has a normalized accuracy value of 1. Average normalized accuracy is the mean normalized accuracy across multiple tasks. Our experiments cover both Vision and NLP domains. For details on datasets, models, and configurations, see Appendix B.

4.1. Merging Model Breadcrumbs

In this section, we explore the scalability and performance of merged Model Breadcrumbs as we progressively include tasks, reaching a total of 8 in our investigation, as detailed in Appendix B. Merging enables the creation of multi-task models that can excel across various tasks simultaneously. This versatility is valuable both in scenarios where we have multiple privately fine-tuned models as well as in cases where we have access to publicly available fine-tuned models. This allows the extraction of existing knowledge from these models without the need for extra training or access to additional training data. Table 1 presents a comparison between Model Breadcrumbs with 91% sparsity ($\beta = 90\%$, $\gamma = 99\%$), the recently proposed Task Arithmetic (Ilharco et al., 2022a), and Fisher Merging (Matena & Raffel, 2022) across 8 tasks, using ViT-B-32 model. Model Breadcrumbs

Algo. / Data	MRPC	RTE	CoLA	SST-2
Zero-shot	74.8	52.7	8.29	92.7
FT	87.9 ±0.68	76.2 ±0.46	51.6 ±0.37	93.3 ±0.35
FT + TA	88.6 ±0.59	76.4 ±0.40	52.1 ±0.32	93.5 ±0.31
FT + Ours	**90.0 ±0.50**	**77.5 ±0.38**	**53.2 ±0.34**	**94.4 ±0.32**

Table 2. Model merging enhances the fine-tuned (FT in table) models. Specifically, the merger of Breadcrumbs (ours in table) yields higher-performing models without requiring additional training data or combining with models trained on similar data. Values represent the average performance over 20 runs, followed by the standard error. TA stands for Task Arithmetic.

outperforms all considered methods by a substantial margin. Fisher Merging (Matena & Raffel, 2022) lags behind both Task Arithmetic (Ilharco et al., 2022a) and Model Breadcrumbs, and notably, it requires significantly more computational resources. Therefore, we proceed with the rest of our studies without evaluating Fisher Merging.

In Figure 3, we assess all possible task subsets of the 8 tasks detailed in Section B, amounting to a total of $256 = 2^8$ combinations, under two settings: 1. evaluation over all 8 tasks and, 2. evaluation only on the subset of tasks that have been observed. As we can see in Figure 2a merging Model Breadcrumbs (91% sparsity) results in superior multi-task models compared to the Task Arithmetic (Ilharco et al., 2022a). Furthermore, the performance gap between these two approaches increases as more tasks are observed, resulting in vastly superior multi-task models when more Model Breadcrumbs are available. In Figure 2b we can see that for small task numbers the resulting merged model performs closely to that of the multiple fine-tuned models although the gap increases as more tasks are added. Model Breadcrumbs again proves to be more performance that Task Arithmetic (Ilharco et al., 2022a) in this setting.

4.2. Validation-Free Setting

In Section 4.1, we compared Model Breadcrumbs and Task Arithmetic (Ilharco et al., 2022a) under their respective optimal hyperparameters. These hyperparameters were tuned based on model performance on the validation dataset for each subset of tasks following (Ilharco et al., 2022a). However, as the number of tasks increases, the search for optimal hyperparameters becomes increasingly resource-intensive. Furthermore, the need for a validation set from each task being added can be restrictive due to privacy concerns or due to the unavailability of additional validation data. Thus, we consider a new setting where hyperparamters are tuned based on a few tasks, and subsequent tasks are added using these pre-determined hyperparameters.

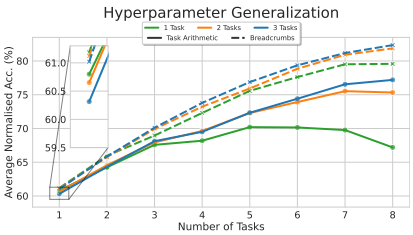


Figure 4. Validation-Free Setting. Using the ViT-B-32 model, we tune hyperparameters for Breadcrumbs and Task Arithmetic based on the first few tasks. Subsequent tasks are added without further validation. Breadcrumbs significantly outperforms Task Arithmetic in this setting.

The results, shown in Figure 4, reveal that the hyperparameters of Model Breadcrumbs exhibit a high degree of generalizability. For the ViT-B-32 model, optimal hyperparameters remain consistent across scenarios with three or more tasks, up to the 8-task scenario. This stability in hyperparameter settings persists with increasing model scale, as demonstrated in additional experiments detailed in Appendix C. Motivated by these results, we extended the evaluation to a longer task sequence of 200 tasks using the ViT-L-14 model, which is detailed in Appendix C. Despite the larger task sequence, Model Breadcrumbs consistently outperforms Task Arithmetic (Ilharco et al., 2022a), highlighting the robustness and generalizability of its hyperparameters. The stability in hyperparameter settings simplifies implementation, reduces the need for extensive tuning, and enhances the practicality and ease of use of Model Breadcrumbs in real-world multi-task learning scenarios. This fundamental divergence underscores the substantial advantage of Model Breadcrumbs over Task Arithmetic (Ilharco et al., 2022a).

4.3. Target Task Improvement via Model Merging

We explore the potential of improving a single target task's performance through model merging, switching to NLP tasks to demonstrate the method's cross-modality effectiveness. We fine-tuned the T5-base model (Raffel et al., 2020) for 4 GLUE tasks (Wang et al., 2018) and merged six publicly available fine-tuned T5-base models with each of them (details in Appendix B). Table 2 compares Zeroshot, pure fine-tuning, Task Arithmetic (Ilharco et al., 2022a), and Breadcrumbs. The results show that Breadcrumbs consistently enhances fine-tuned model performance, outperforming other methods without additional training or data from the same dataset. This highlights Model Breadcrumbs' versatility and effectiveness across diverse tasks.

5. Conclusions

We introduced Model Breadcrumbs, a simple and effective method for building multi-task models from existing fine-tuned foundation models. Our experiments demonstrate

its ability to enhance performance across multiple tasks with stable and generalizable hyperparameters, making it practical for real-world scenarios. Our experiments show that larger models benefit more, narrowing the performance gap with individual fine-tuned models. Additionally, our NLP experiments highlight its versatility across different modalities.

Model Breadcrumbs' performance depends on the quality of the initial fine-tuned models, with issues such as poor generalization potentially propagating. Future research should focus on mitigating these limitations and exploring advanced aggregation techniques. As tasks increase, expanding model capacity will be crucial for maintaining performance. In conclusion, Model Breadcrumbs excels in simplicity, efficiency, and effectiveness for multi-task models. By leveraging publicly available fine-tuned models, it supports community-driven model refinement and updatable machine learning, contributing to efficient and scalable multi-task learning solutions.

Acknowledgements

We acknowledge funding from the NSERC Discovery Grant RGPIN-2021-04104 and FRQNT New Scholar. This research was enabled in part by compute resources provided by Digital Research Alliance of Canada (the Alliance) and Calcul Québec.

References

Ainsworth, S. K., Hayase, J., and Srinivasa, S. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022.

Asadi, N., Davari, M., Mudur, S., Aljundi, R., and Belilovsky, E. Prototype-sample relation distillation: Towards replay-free continual learning. In *International Conference on Machine Learning*, pp. 1093–1106. PMLR, 2023.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Cheng, G., Han, J., and Lu, X. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105:1865–1883, October 2017.

Choshen, L., Venezian, E., Slonim, N., and Katz, Y. Fusing finetuned models for better pretraining. *arXiv preprint arXiv:2204.03044*, 2022.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , and Vedaldi, A. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Cossu, A., Tuytelaars, T., Carta, A., Passaro, L., Lomonaco, V., and Bacciu, D. Continual pre-training mitigates forgetting in language and vision. *arXiv preprint arXiv:2205.09357*, 2022.

Davari, M. *Neural Network Approaches to Medical Toponym Recognition*. PhD thesis, Concordia University, 2020.

Davari, M. and Belilovsky, E. Probing representation forgetting in continual learning. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

Davari, M., Kosseim, L., and Bui, T. Timbert: toponym identifier for the medical domain based on bert. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 662–668, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dolan, B. and Brockett, C. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*, 2005.

Fabbri, A. R., Li, I., She, T., Li, S., and Radev, D. R. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *arXiv preprint arXiv:1906.01749*, 2019.

Farahnak, F., Mohammadi, E., Davari, M., and Kosseim, L. Semantic similarity matching using contextualized representations. In *Canadian Conference on AI*, 2021.

Helber, P., Bischke, B., Dengel, A., and Borth, D. 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.

Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., and Igel, C. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*, 2013.

Ilharco, G., Ribeiro, M. T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., and Farhadi, A. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022a.

Ilharco, G., Wortsman, M., Gadre, S. Y., Song, S., Hajishirzi, H., Kornblith, S., Farhadi, A., and Schmidt, L. Patching open-vocabulary models by interpolating weights. *Advances in Neural Information Processing Systems*, 35: 29262–29277, 2022b.

Khot, T., Clark, P., Guerquin, M., Jansen, P., and Sabharwal, A. Qasc: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8082–8090, 2020.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

Lai, G., Xie, Q., Liu, H., Yang, Y., and Hovy, E. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Lin, B. Y., Zhou, W., Shen, M., Zhou, P., Bhagavatula, C., Choi, Y., and Ren, X. Commongen: A constrained text generation challenge for generative commonsense reasoning. *arXiv preprint arXiv:1911.03705*, 2019.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Luo, H., Ji, L., Shi, B., Huang, H., Duan, N., Li, T., Li, J., Bharti, T., and Zhou, M. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*, 2020.

Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Marks, J., Davari, M., and Kosseim, L. CLaC at SemEval-2024 task 2: Faithful clinical trial inference. In Ojha, A. K., Doğruöz, A. S., Tayyar Madabushi, H., Da San Martino, G., Rosenthal, S., and Rosá, A. (eds.), *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

Matena, M. S. and Raffel, C. A. Merging models with fisher-weighted averaging. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17703–17716. Curran Associates, Inc., 2022.

Myung, I. J. Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology*, 47(1):90–100, 2003.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. *Advances in Neural Information Processing Systems (NIPS)*, 2011.

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

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding with unsupervised learning. *OpenAI blog*, 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Ramé, A., Ahuja, K., Zhang, J., Cord, M., Bottou, L., and Lopez-Paz, D. Model ratatouille: Recycling diverse models for out-of-distribution generalization. *arXiv preprint arXiv:2212.10445*, 2022.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR, 2021.

Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., et al. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021.

Rothchild, D., Tamkin, A., Yu, J., Misra, U., and Gonzalez, J. C5t5: Controllable generation of organic molecules with transformers. *arXiv preprint arXiv:2108.10307*, 2021.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Stoica, G., Bolya, D., Bjorner, J., Hearn, T., and Hoffman, J. Zipit! merging models from different tasks without training. *arXiv preprint arXiv:2305.03053*, 2023.

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

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pp. 23965–23998. PMLR, 2022.

Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. 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, 2010.

Yadav, P., Tam, D., Choshen, L., Raffel, C., and Bansal, M. Ties-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Yin, P., Neubig, G., Yih, W.-t., and Riedel, S. Tabert: Pretraining for joint understanding of textual and tabular data. *arXiv preprint arXiv:2005.08314*, 2020.

Table 3. Data statistics.

Dataset	Training	Validation	Testing	Classes
Cars	7,330	814	8041	196
DTD	3,384	376	1,880	47
EuroSAT	21,600	2,700	2,700	10
GTSRB	23,976	2,664	12,630	43
MNIST	55,000	5,000	10,000	10
RESISC45	17,010	1,890	6,300	45
SUN397	17,865	1,985	19,850	397
SVHN	68,257	5,000	26,032	10

A. Algorithm: Model Breadcrumbs

In this section, we have a closer look at the Model Breadcrumbs algorithm for model merging, which was introduced in Section 3. The algorithm is given below.

Algorithm 1 Model merging via Breadcrumbs.

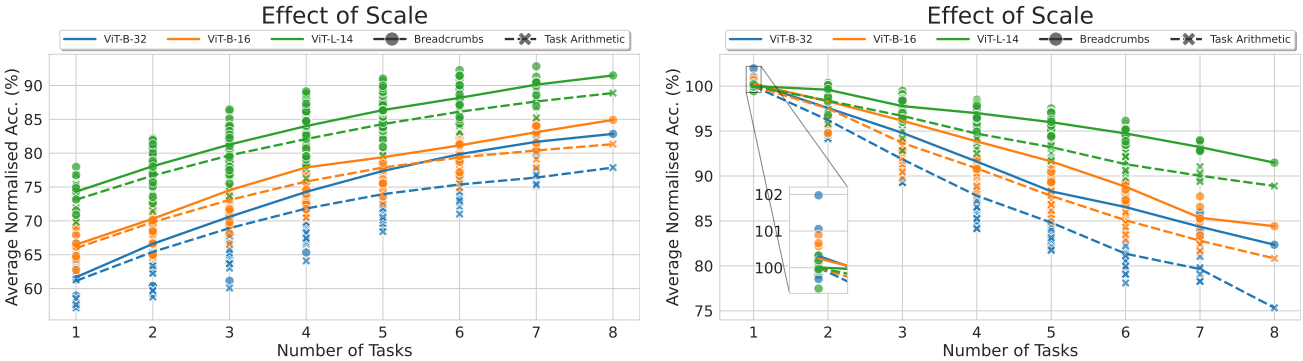
Input: Foundation model θ, Fine-tuned models $\{\theta'_t\}_{t=1}^{n}$, α, β, and γ
Output: Multi-task model θ^*
for $t = 1$ **to** n **do**
  $\theta_t^d \leftarrow \theta'_t - \theta$ **Create task direction.**
  for each layer layer in $\text{Layers}(\theta)$ **do**
    $p \leftarrow |\theta^d_{t,\text{layer}}|$ **Record abs. value of task direction at current layer**
    $m^{\gamma}_{t,\text{layer}} \leftarrow \text{mask_topk_percent}(p, k = \gamma)$ **Generate mask for top k percent of weights**
    $m^{\beta}_{t,\text{layer}} \leftarrow \text{mask_bottomk_percent}(p, k = \beta)$ **Generate mask for bottom k percent of weights**
    $m^{\beta,\gamma}_{t,\text{layer}} \leftarrow \text{merge_masks}(m^{\beta}_{t,\text{layer}}, m^{\gamma}_{t,\text{layer}})$
  end for
  $m_t^{\beta,\gamma} \leftarrow \text{stack_masks}\left(\left\{m^{\beta,\gamma}_{t,\text{layer}}\right\}_{\text{layer}\in\text{Layers}(\theta)}\right)$ **Generate 1 mask per fine-tuned model**
end for
$\theta^* \leftarrow \theta + \alpha \sum_{t\in T} m_t^{\beta,\gamma} \cdot \theta_t^d$ **Generate the multi-task model**
Return θ^*

B. Data, Models, and Configurations

In Section 4.1, 4.2, C, and D, we assess our findings using an extensive set of 8 datasets: Cars (Krause et al., 2013), DTD (Cimpoi et al., 2014), EuroSAT (Helber et al., 2019), GTSRB (Houben et al., 2013), MNIST (LeCun et al., 2010), RESISC45 (Cheng et al., 2017), SUN397 (Xiao et al., 2010), and SVHN (Netzer et al., 2011) (see Tabel 3 for more details). We fine-tune various CLIP models (Radford et al., 2021) (ViT-B-32, ViT-B-16, and ViT-L-14) following a procedure similar to (Ilharco et al., 2022b), our fine-tuning comprises 2000 iterations with a batch size of 128, a learning rate set to 1e-5, and a cosine annealing learning rate schedule with 200 warm-up steps. The AdamW optimizer (Loshchilov & Hutter, 2017) with a weight decay of 0.1 is employed for optimization.

Throughout the fine-tuning process, we freeze the weights of CLIP's text encoder classification layer. This ensures no introduction of additional learnable parameters, a strategy validated in prior work (Ilharco et al., 2022b).

In Section 4.3, we apply our approach to the NLP domain, specifically investigating four GLUE tasks (Wang et al., 2018) (MRPC (Dolan & Brockett, 2005), RTE (Wang et al., 2018), CoLA (Warstadt et al., 2019), and SST-2 (Socher et al., 2013)) based on the benchmarks used by (Ilharco et al., 2022a; Wortsman et al., 2022). Our process involves fine-tuning the T5-base model (Raffel et al., 2020) on these datasets and subsequently merging publicly available fine-tuned models from other datasets (IMDB (Maas et al., 2011), RACE (Lai et al., 2017), QASC (Khot et al., 2020), MultiNews (Fabbri et al., 2019), SQuAD (Rajpurkar et al., 2016), and CommonGen (Lin et al., 2019)) into each of them. We fine-tune the T5-base model (Raffel et al., 2020) on MRPC (Dolan & Brockett, 2005), RTE (Wang et al., 2018), CoLA (Warstadt et al., 2019), and SST-2 (Socher et al., 2013). Our fine-tuning process utilizes a batch size of 32, a learning rate of 1e-5, and lasts for 5 epochs



(a) At each point, evaluation is performed over all 8 tasks (i.e currently merged and the remaining unmerged tasks).

(b) At each point, evaluation is performed only over the observed tasks.

Figure 5. Comparative performance analysis of Model Breadcrumbs and Task Arithmetic (Ilharco et al., 2022a) methods across varying CLIP model scales (ViT-B-32, ViT-B-16, and ViT-L-14) as the number of tasks increases. The solid line represents the averaged normalized accuracy across all evaluation points. Each data point corresponds to an experiment involving a subset of the 8 tasks under study. Our findings highlight the potential of larger-scale models to mitigate performance degradation and, as seen in Figure 5b, the capability of Model Breadcrumbs to produce multi-task models that surpass individual fine-tuned models for specific tasks.

using the AdamW optimizer with a linear learning rate schedule. We use a create a validation set from the training data equal in size to the test set to pick the best model. The publicly available fine-tuned models are sourced from the Hugging Face hub[2], and the specific models can be accessed via the following links:

- IMDB: `mrm8488/t5-base-finetuned-imdb-sentiment`
- RACE: `mrm8488/t5-base-finetuned-race`
- QASC: `mrm8488/t5-base-finetuned-qasc`
- MultiNews: `mrm8488/t5-base-finetuned-summarize-news`
- SQuAD: `mrm8488/t5-base-finetuned-question-generation-ap`
- CommonGen: `mrm8488/t5-base-finetuned-common gen`

C. Effect of Scale

We explore the impact of using larger CLIP models by comparing ViT-B-32, ViT-B-16, and ViT-L-14 models. Optimal Model Breadcrumbs were found at 90% ($\beta = 90\%$, $\gamma = 99\%$), 90% ($\beta = 90\%$, $\gamma = 99.2\%$), and 85% ($\beta = 85\%$, $\gamma = 99\%$) sparsity respectively. As shown in Figure 5, larger models significantly enhance performance for both Model Breadcrumbs and the Task Arithmetic (Ilharco et al., 2022a) baseline. Adding more tasks improves the capacity to construct better multi-task models, with larger models showing superior results.

In Figure 5a, using the ViT-L-14 model and considering 8 tasks, merging Model Breadcrumbs achieves 91.48% of the performance of 8 individual fine-tuned models, significantly reducing inference time and compute resources with only a minor performance loss. Figure 5b indicates that performance decline when merging tasks can be mitigated by using larger models. For ViT-L-14, Model Breadcrumbs often match or exceed the performance of individual fine-tuned models when merging tasks.

We further examined task merging for ViT-L-14 with two tasks. Figure 6 shows that merging tasks with Model Breadcrumbs generally improves performance on both tasks, outperforming Task Arithmetic (Ilharco et al., 2022a). Model Breadcrumbs consistently produce multi-task models that surpass Task Arithmetic versions, highlighting its potential to enhance task-specific performance.

[2] https://huggingface.co/models

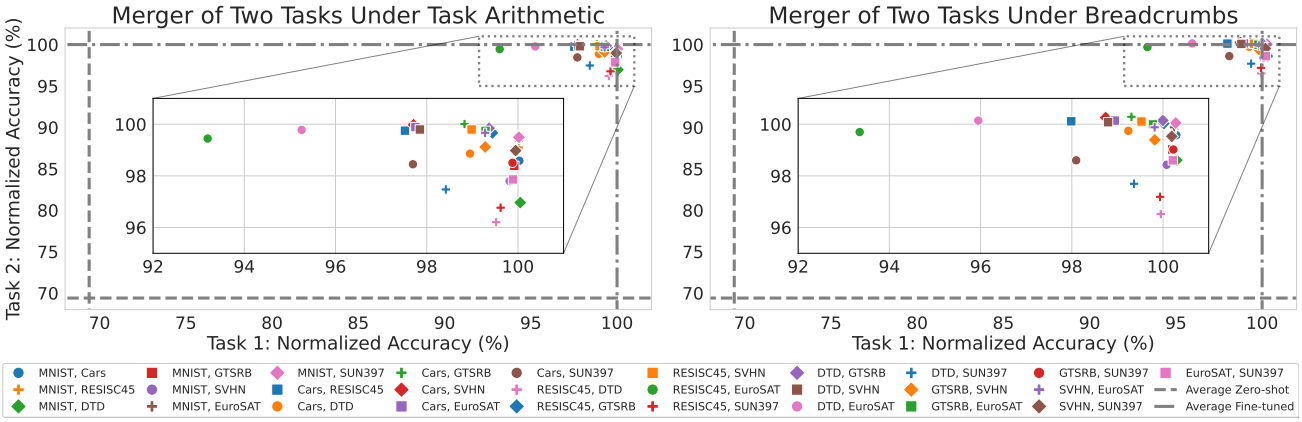


Figure 6. Comparison of Model Breadcrumbs and Task Arithmetic (Ilharco et al., 2022a) in the merger of task pairs, revealing improved accuracy on both tasks and a higher frequency of multi-task models surpassing individual fine-tuned accuracy levels when employing Model Breadcrumbs.

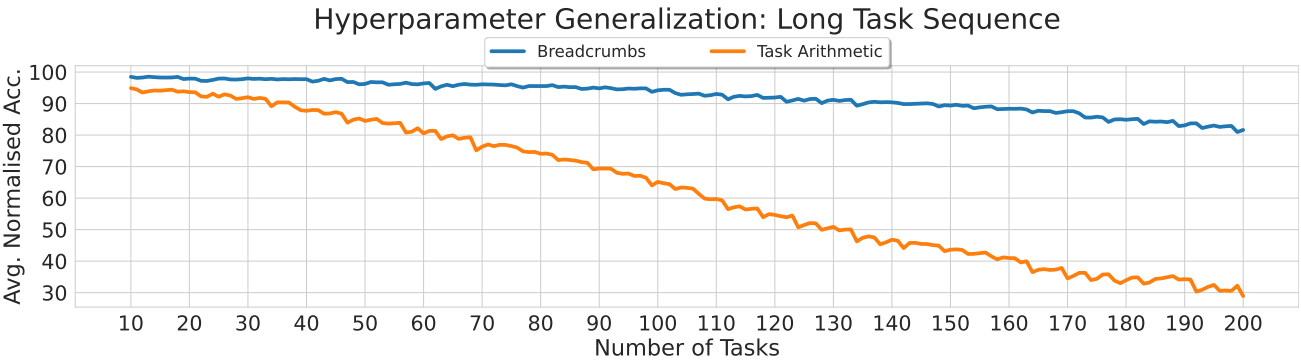


Figure 7. The 200-task sequence originates from the ImageNet dataset (Deng et al., 2009), created by dividing the data into 200 5-class classification tasks. After encountering 10 tasks using the ViT-L-14 model, the best hyperparameters for each method (Breadcrumbs with 85% sparsity and Task Arithmetic (Ilharco et al., 2022a)) are selected and fixed. Each point on the plot represents the evaluation of the method over all tasks observed up to that point. With an increasing number of tasks, Model Breadcrumbs consistently outperforms Task Arithmetic (Ilharco et al., 2022a) by a substantial margin, highlighting the robustness of hyperparameters in the Model Breadcrumbs approach.

We also studied the generalization of hyperparameters with larger models. Figure 8 shows that Model Breadcrumbs hyperparameters are highly generalizable across tasks, consistent with our results for ViT-B-32 (see Section 4.2). For ViT-L-14, the hyperparameters remain stable beyond one task, underscoring the robustness and versatility of Model Breadcrumbs. Task Arithmetic (Ilharco et al., 2022a) quickly collapses in performance as tasks increase.

To further test generalizability, we split ImageNet data (Deng et al., 2009) into 200 tasks to study a the effect of model merging in long task sequences (Davari & Belilovsky, 2021; Asadi et al., 2023). After finding optimal hyperparameters using 10 tasks, we incrementally merged all 200 tasks. Figure 7 shows Model Breadcrumbs (85% sparsity: $\beta = 85\%$, $\gamma = 99.3\%$) consistently outperforming Task Arithmetic, demonstrating the approach's scalability and robustness.

In summary, Model Breadcrumbs exhibit stable hyperparameters, simplifying implementation and reducing the need for extensive tuning. This stability offers a substantial advantage in real-world multi-task learning scenarios compared to Task Arithmetic.

D. Ablations

In this section, we perform ablations to examine alternative design decisions within the Model Breadcrumbs method. Specifically, we explore different approaches for constructing the masking operation, namely: 1. Bottom-Weight Masking: Masking only the bottom-most smallest absolute magnitude weights per layer. 2. Top-Weight Masking: Masking only the

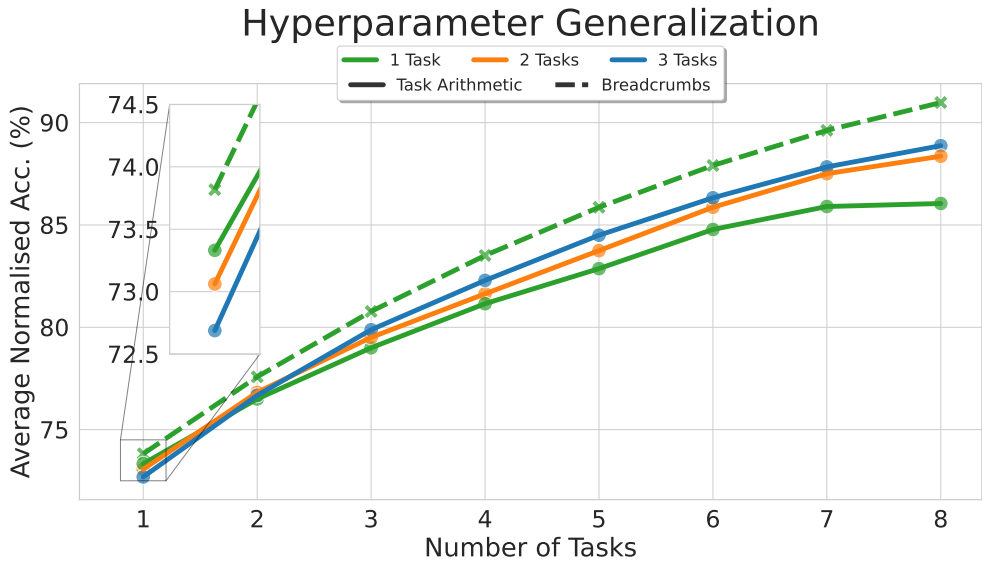


Figure 8. Validation free setting using the ViT-L-14 model. The Model Breadcrumbs method was only tune for the 1 task scenario and evaluate on the additional tasks using those hyperparameters, though the Task Arithmetic appraoch was given more chances to adjust its hyperparameters (task 1, 2, and 3). We observe that Breadcrumbs substantially outperforms task vectors in this setting.

top largest absolute magnitude weights per layer. We compare these alternatives to the full Model Breadcrumbs approach, which encompasses both (1) and (2), as well as the Task Arithmetic (Ilharco et al., 2022a) method, which lacks any masking. In our investigation, we conduct a grid search to identify the optimal hyperparameters for each of the four configurations. We assess the resulting multi-task models on 8 tasks discussed in Section B. The results are shown in Figure 9.

Our findings reveal two key insights: (i) both forms of weight masking, as employed in Model Breadcrumbs, are essential for achieving competitive performance. Model Breadcrumbs, which combines both bottom and top weight masking, emerges as the most effective approach. (ii) The grid search for hyperparameters within the Model Breadcrumbs approach yields a higher distribution of high-performance multi-task models compared to the other three settings. Furthermore, there is much lower variation in the overall performance distribution of the multi-task models produced by the Model Breadcrumbs. These observations underscore the robustness of Model Breadcrumbs to variations in hyperparameter settings, further enhancing its practicality and reliability in real-world applications.

In Figure 10, we examine the cosine similarity between tasks using Model Breadcrumbs and Task Arithmetic (Ilharco et al.,

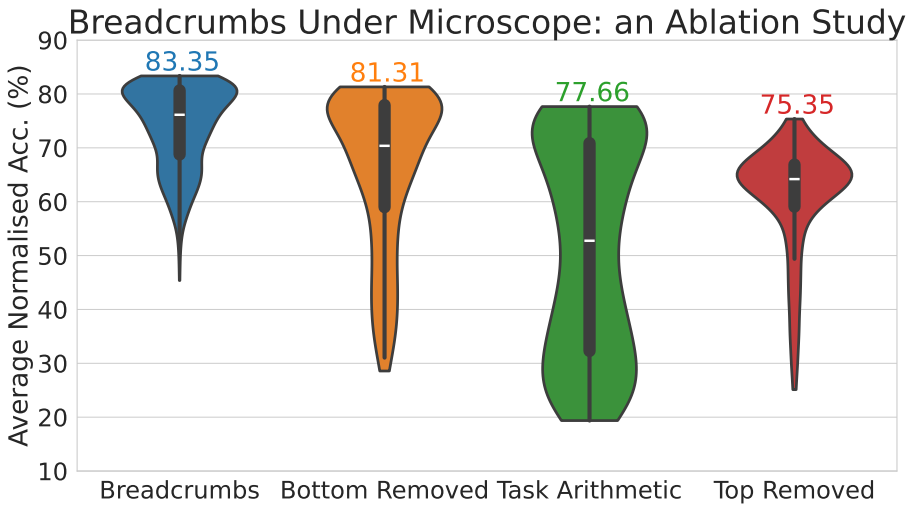


Figure 9. Performance comparison of the Model Breadcrumbs against alternative masking choices, reveals: Model Breadcrumbs yields a higher distribution of high-performance multi-task models, underlining its robustness towards hyperparameter perturbations. Model Breadcrumbs produces the highest performing multi-task model. The number on top of each violin indicates the performance of the highest performing model of that setting.

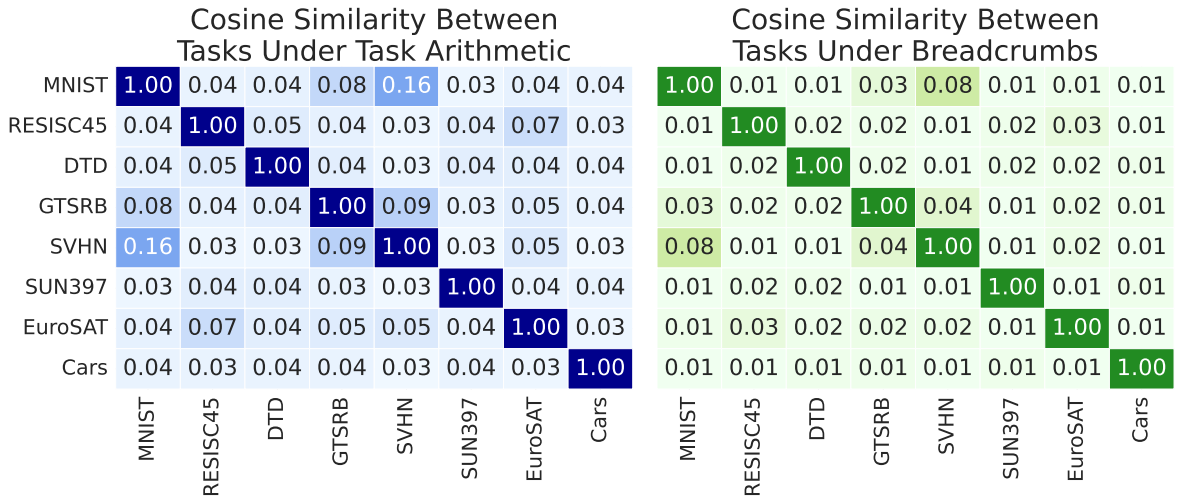
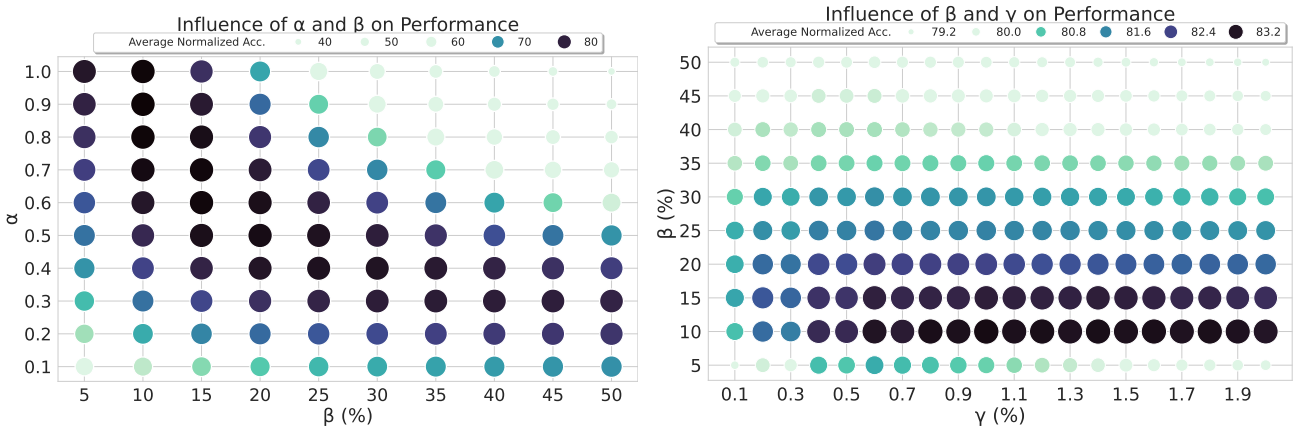


Figure 10. Comparison of Cosine Similarity Between Tasks in Model Breadcrumbs and Task Arithmetic. The figure illustrates the cosine similarity distribution among tasks, highlighting how Model Breadcrumbs enforces greater orthogonality, leading to reduced interference during model merging.



(a) α vs. β: as β grows in magnitude lower α's are required.

(b) β vs. γ: lower β's have more tolerance on variations of γ.

Figure 11. Influence of hyperparameters on model performance using the ViT-B-32 model across eight vision tasks. For ease of readability, in this part we use β and γ to represent how much weights have been masked. It shows the relationships between α and β, and between β and γ, highlighting the stability of hyperparameters across the possible combinations.

2022a). Most tasks show orthogonality, indicating minimal side effects upon merging. However, upon closer inspection, semantically similar tasks (e.g., MNIST (LeCun et al., 2010), SVHN (Netzer et al., 2011), and GTSRB (Houben et al., 2013)) exhibit higher cosine similarity, suggesting non-orthogonality. This similarity could introduce interference during merging. In contrast, Model Breadcrumbs pushes all cosine similarity values closer to zero, reinforcing orthogonality. This reduction in interference could explain the enhance performance of the resulting multi-task models when using Breadcrumbs.

In Figure 11, we demonstrate the impact of hyperparameters on the performance of models using the ViT-B-32 model, assessed across eight vision tasks outlined in Section B. For ease of readability, in this part we use β and γ to represent how much weights have been masked. Figure 11a examines the relationship between α and β, the primary determinants of task vector sparsity. As β increases and more weights are masked, large alphas, which amplify the remaining weights' contributions, become less tolerable, necessitating lower α's as β grows. Upon identifying optimal α and β values, we investigate gamma. In Figure 11b, we depict the relationship between β and gamma. Regardless of beta's value, the γ that optimizes a combination of α and β tends to hover around 1%, with lower betas allowing for higher gammas and vice versa. Across both figures, we consistently observe that numerous combinations of alpha, beta, and γ result in high-performing merged models, as previously noted in Figure 9.

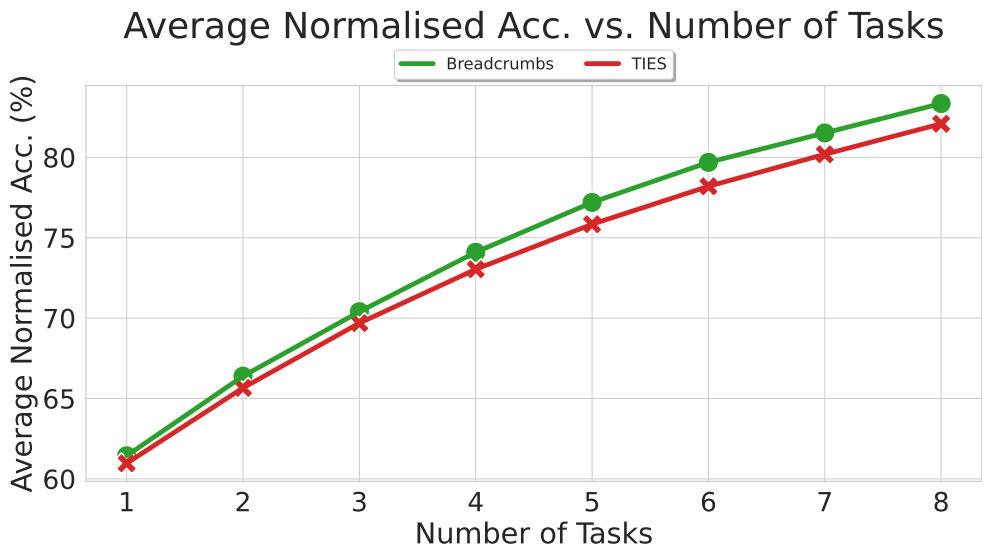


Figure 12. Comparison of Model Breadcrumbs and TIES merging methods across tasks, illustrating Model Breadcrumbs' consistent outperformance, with the performance gap widening as tasks increase. The results underscore the superior practical performance gains of Model Breadcrumbs at scale.

E. TIES Merging

A concurrent study by Yadav *et al.* (Yadav et al., 2023) presents a method named TIES. Like the Task Arithmetic method (Ilharco et al., 2022a), TIES initially constructs a set of Task Vectors. These vectors undergo a masking process to eliminate interfering weights, identified as a percentage of overall weights with low magnitudes. The remaining unmasked weights undergo a sign alignment operation to determine their polarity. Finally, a scaled sum merges the task vectors with the pre-trained model.

Our approach differs from TIES in two key aspects. Firstly, we apply masking to both very large and small magnitude weights of the task vectors to minimize interference, whereas TIES focuses solely on small magnitude weights. Secondly, our masking strategy employs layer-wise masking, wherein a certain percentage of weights are masked based on their magnitude relative to the weights within that layer, as opposed to overall masking, which ranks all model weights by magnitude and masks the smallest ones. Notably, in the context of task vectors, overall masking typically targets weights in the early layers (Matena & Raffel, 2022).

In Figure 12, we compare our method to TIES (Yadav et al., 2023), where at each point: (I) We show results for `ViT-B-32` model where we found the best hyper-parameters for that specific number of tasks for each method. (II) We show the average normalized accuracy over all subsets of the 8 tasks detailed in Section B, amounting to a total of $256 = 2^8$ combinations. (III) The evaluation is performed over all 8 tasks at each point. As we can see from Figure 12, Model Breadcrumbs merging consistently outperforms the TIES method at each point, with the performance gap widening as more tasks are considered. This highlights the significant practical performance advantages of Model Breadcrumbs on a larger scale.