Model Breadcrumbs: Scaling Multi-Task Model Merging with Sparse Masks

MohammadReza Davari and Eugene Belilovsky

Concordia University and Mila – Quebec AI Institute
{mohammadreza.davari,eugene.belilovsky}@concordia.ca

Abstract. The rapid development of AI systems has been greatly influenced by the emergence of foundation models. A common approach for targeted problems involves fine-tuning these pre-trained foundation models for specific target tasks, resulting in a rapid spread of models fine-tuned across a diverse array of tasks. This work focuses on the problem of merging multiple fine-tunings of the same foundation model derived from a spectrum of auxiliary tasks. We introduce a new simple method, Model Breadcrumbs, which consists of a sparsely defined weight set that guides model adaptation within the weight space of a pre-trained model. These breadcrumbs are constructed by subtracting the weights from a pre-trained model before and after fine-tuning, followed by a sparsification process that eliminates weight outliers and negligible perturbations. Our experiments demonstrate the effectiveness of Model Breadcrumbs to simultaneously improve performance across multiple tasks. This contribution aligns with the evolving paradigm of updatable machine learning, reminiscent of the collaborative principles underlying open-source software development, fostering a community-driven effort to reliably update machine learning models. Our method is shown to be more efficient and unlike previous proposals does not require hyperparameter tuning for each new task added. Through extensive experimentation involving various models, tasks, and modalities we establish that integrating Model Breadcrumbs offers a simple, efficient, and highly effective approach for constructing multi-task models and facilitating updates to foundation models. [1]

Keywords: Model Merging · Transfer Learning · Foundation Models

1 Introduction

In recent years, foundational models [3] have become instrumental tools, exhibiting unprecedented efficacy across multiple domains. These models are characterized by their extensive scale, generality, and capacity to learn and generalize knowledge from vast datasets, offering promising solutions to a diverse range of problems. The inherent ability of foundational models to be fine-tuned has led to

---

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

advancements in natural language processing (NLP) [14,30,32,43–45], computer vision [5,25,34,42,49], and other related fields [50,51,62].

On one hand, the scalability of expanding foundational models to increase the number of tasks they can perform in practice poses a significant challenge as approaches such as joint training are limited in many practical scenarios [8,9]. In domains such as healthcare, stringent data privacy concerns often prohibit access to the underlying training data, even when the fine-tuned model on the said data is publicly accessible, rendering joint training infeasible [2,10]. Even in scenarios where access to training data is possible, the computational demands of simultaneous training on a multitude of tasks becomes restraining.

On the other hand, the widespread adoption of foundational models has led to a certain homogenization in the field [3]. Both the training approach, commonly transfer learning from a popular foundational model [41], and the model architecture itself have become standardized, typically following a few popular foundation models. This standardization has resulted in a proliferation of publicly available fine-tuned models, all sharing the same architecture [12, 18, 56, 61]. However, beyond their conventional use for model inference, these numerous fine-tuned models remain **largely untapped**, representing a missed opportunity [48].

To address the challenges of scalability, practical constraints, and unlock the untapped potential of the growing pool of publicly available fine-tuned models, recent developments in neural network weight averaging techniques have gained attention [6, 11, 16, 21, 23, 39, 48, 57, 58, 60]. These approaches enable the practitioners to re-purpose the increasingly valuable publicly available fine-tuned models.

Closer to our approach, *Task Arithmetic* were introduced by Ilharco *et al.* [21]. In their method, a foundation model is refined by incorporating the scaled average of the differences between multiple fine-tuned models and the foundation model. This allows for the creation of a multi-task model without the need for additional training or access to the original training data. However, despite its potential, the Task Arithmetic method [21] encounters limitations when dealing with numerous tasks. This is mainly due to its dependence on hyperparameter tuning through validation set performance, a process that becomes computationally impractical at scale, coupled with an increasing accumulation of noise as more tasks are merged to the foundation model.

To address these challenges and to capitalize on the untapped resources within the field, our paper introduces Model Breadcrumbs, a simple solution designed to tackle scalability, noise reduction in merging tasks, and hyperparameter generalization issues. Model Breadcrumbs constructs multi-task models from pre-existing fine-tuned models (see Figure 1), surpassing limitations faced by existing methods. We demonstrate that Model Breadcrumbs not only yields competitive multi-task models but also provides hyperparameters that generalize effectively as the number of tasks increases. In Section 2, we provide context through a review of related work. Sections 3 and 4 present our framework and its

evaluation. Finally, Section 5 outlines the scope and limitations of our proposed method. Our key contributions and findings are summarized as follows:

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

Model Merging   Recent studies in the literature have explored the merging of models trained from scratch with different initializations [1, 53]. 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. Furthermore, our investigation is focused on the scalability of merging methods, exploring the dynamics when multiple models are involved in the merger process.

Neyshabur *et al.* [39] 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.* [57] 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.* [6] merged models from multiple trained models in order to create a better pre-trained 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 [36] 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 [37]. 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.* [21], 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.* [60] presents a method named TIES. Like the Task Arithmetic method [21], 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 [36].

Federated Learning The concept of initiating learning with a pre-trained model has been explored in the federated learning literature, as seen in recent works such as [29, 40]. These studies focused on a single downstream task where data is distributed across multiple clients. In their approach, each client periodically aggregates models during the training process. It's important to note that this differs from our approach, which deals with multi-task learning involving multiple downstream tasks rather than a single task distributed across clients.

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_t^d.

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

Note that θ_t^d contains both (a) large outliers, indicating substantial deviations from the pre-trained starting point, and (b) negligible differences representing
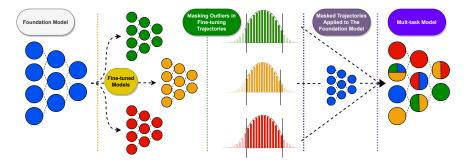
Fig. 1: Method overview. We start with a foundational model that has undergone fine-tuning on various tasks. Next, we build a fine-tuning trajectory for each fine-tuned model by subtracting the pre-trained model weights from each of the fine-tuned models (task vectors). We then, at each layer, apply a masking operation over the absolute value of the the resulting trajectory, eliminating both outliers and small values. Finally, these masked task vectors are aggregated and combined with the reference pre-trained model to create a unified multi-task model.

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.

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. Algorithm 1 describes the overall procedure of the Model Breadcrumbs merging strategy.

Algorithm 1: Model merging via Breadcrumbs.

Data: Foundation model θ, Fine-tuned models $\{\theta'_t\}_{t=1}^{n}$, α, β, and γ
Result: Multi-task model θ^*
for $t \leftarrow 1$ **to** n **do**

    /* Create task direction.   */

    $\theta_t^d \leftarrow \theta'_t - \theta$

    for layer $\in$ Layers(θ) **do**

        /* Record the absolute value of the task direction at the current layer   */

        $p \leftarrow |\theta_{t,\text{layer}}^d|$

        /* Generate a mask for top k percent of the weights   */

        $m_{t,\text{layer}}^{\gamma} \leftarrow \text{mask\_topk\_percent}(p, k = \gamma)$

        /* Generate a mask for the bottom k percent of the weights   */

        $m_{t,\text{layer}}^{\beta} \leftarrow \text{mask\_bottomk\_percent}(p, k = \beta)$

        $m_{t,\text{layer}}^{\beta,\gamma} \leftarrow \text{merge\_masks}(m_{t,\text{layer}}^{\beta}, m_{t,\text{layer}}^{\gamma})$

    end

    /* Generate 1 mask per fine-tuned model   */

    $m_t^{\beta,\gamma} \leftarrow \text{stack\_masks}\left(\left\{m_{t,\text{layer}}^{\beta,\gamma}\right\}_{\text{layer}\in\text{Layers}(\theta)}\right)$

end
/* Generate the multi-task model   */
$\theta^* \leftarrow \theta + \alpha \sum_{t\in T} m_t^{\beta,\gamma}.\theta_t^d$
return θ^*

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

4 Experiments

In this section, we conduct a series of experiments to evaluate the Model Breadcrumbs framework. Our experiments focus on the following key aspects: 1. **Merging Model Breadcrumbs**: We incrementally add tasks, totalling 8 in our investigation, to assess the scalability and performance of merged Model Breadcrumbs as the number of tasks increases. 2. **Generalization of Hyperparameters**: We explore how the hyperparameters introduced by Model Breadcrumbs—α, β, and γ—generalize over the number of datasets. 3. **Effect of Scale**: We investigate the impact of the scale and complexity of the foundation models on the Model Breadcrumbs' adaptability and robustness. 4. **Target Task Improvement**: We examine the potential of enhancing the performance a fine-tuned model on a target task by merging related tasks into it. 5. **Ablation Study**: We study the importance of the design choices introduced by Model Breadcrumbs for successful and competitive model merging.

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 [36] lags behind both Task Arithmetic [21] and Model Breadcrumbs, it also requires significantly more computational resources.

Method	Avg. Normalized Acc.
Breadcrumbs	**83.35**
Task Arithmetic [21]	77.66
Fisher Merging [36]	75.11
Random Sparsed Task Arithmetic	74.00

4.1 Data, Metrics, and Models

In our analysis, we follow the benchmarks and settings used by Ilharco *et al.* [21] for a more meaningful comparison with existing reports. We present results using normalized accuracy, a metric calculated as the ratio of accuracy achieved by the merged model to that of the fine-tuned model.

$$\text{Normalized Accuracy} = \frac{\text{Accuracy of Merged Model}}{\text{Accuracy of Fine-tuned Model}} \quad (4)$$

It is noteworthy that the fine-tuned model establishes the upper bound with a normalized accuracy value of 1. Subsequently, the concept of average normalized accuracy is introduced, representing the mean normalized accuracy across multiple tasks. In Section 4.2, 4.3, 4.4, and 4.6, we assess our findings using an extensive set of 8 datasets: Cars [26], DTD [7], EuroSAT [19], GTSRB [20], MNIST [28], RESISC45 [4], SUN397 [59], and SVHN [38]. We fine-tune various CLIP models [42] (ViT-B-32, ViT-B-16, and ViT-L-14) to explore model merging. For more information on the datasets and the fine-tuning process see Appendix A.

In Section 4.5, we apply our approach to the NLP domain, specifically investigating four GLUE tasks [54] (MRPC [15], RTE [54], CoLA [55], and SST-2 [52]) based on the benchmarks used by [21, 57]. Our process involves fine-tuning the T5-base model [46] on these datasets and subsequently merging publicly available fine-tuned models from other datasets (IMDB [35], RACE [27], QASC [24], MultiNews [17], SQuAD [47], and CommonGen [31]) into each of them. Appendix B provides additional details on the datasets, the fine-tuning process, and the publicly available fine-tuned models we used in our experiments.

4.2 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 Section 4.1. 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

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

(b) At each point, evaluation is performed only over the observed tasks.
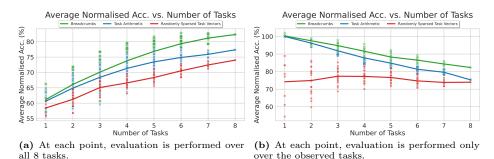
Fig. 2: The solid line is the averaged normalized accuracy across all evaluation points. Each data point corresponds to an experiment involving a subset of the 8 tasks under study. Notably, it is evident that the Model Breadcrumbs (with 90% sparsity), consistently outperform the Task Arithmetic [21]. Specifically, in the experiment involving all eight tasks, the Model Breadcrumbs outperform the Task Arithmetic by a substantial margin of 5.7%.

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 90% sparsity ($\beta = 90\%$, $\gamma = 99\%$), the recently proposed Task Arithmetic [21], and Fisher Merging [36] across 8 tasks, using ViT-B-32 model. Model Breadcrumbs outperforms all considered methods by a substantial margin. Fisher Merging [36] lags behind both Task Arithmetic [21] 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 2, we assess all possible task subsets of the 8 tasks detailed in Section 4.1, 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 (90% sparsity) results in superior multi-task models compared to the Task Arithmetic [21]. 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 prove to be more performance that Task Arithmetic [21] in this setting.

4.3   Validation-Free Setting

In Section 4.2, we compared Model Breadcrumbs and Task Arithmetic [21] under their respective optimal hyperparameters. These hyperparameters were fine-tuned based on model performance on the validation dataset for each subset of
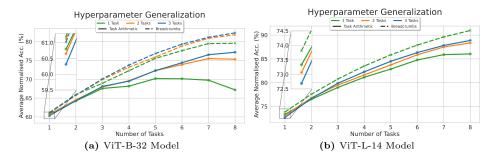
Fig. 3: Validation Free Setting. For the ViT-B-32 model, we tune the hyperparameters of each method (Breadcrumbs and Task Arithmetic) based on the first 1, 2, or 3 tasks and add additional tasks using those hyperparameters (validation set free). For the ViT-L-14 model, the Breadcrumbs method was only 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.

tasks following [21]. 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.

The results are shown in Figure 3. Remarkably, our experiments reveal that the hyperparameters of Model Breadcrumbs exhibit a high degree of generalizability. Specifically, for the ViT-B-32 model when considering scenarios involving three tasks and beyond, up to the 8-task scenario, the optimal hyperparameters remain consistent. Moreover, for the ViT-L-14 model, the hyperparameters do not change beyond the 1 task scenario. This remarkable stability underscores the robustness and versatility of Model Breadcrumbs. We observer that on the other hand the approach Task Arithmetic [21] can quickly collapse in performance.

Motivated by these results, we extended the evaluation to a much longer task sequence using ViT-L-14 model. We split the ImageNet data [13] into 200 tasks, each classifying 5 classes. After finding optimal hyperparameters for both Model Breadcrumbs and Task Arithmetic using 10 tasks, we kept these hyperparameters and incrementally merged all 200 tasks to create a multi-task model. As seen in Figure 4, the observed trend remains, with Model Breadcrumbs (85% sparsity: $\beta = 85\%$, $\gamma = 99.3\%$) consistently outperforming Task Arithmetic [21] by a significant margin as the number of tasks increases. This showcases the generalizability of the hyperparameters for the Model Breadcrumbs approach.

The practical implication of this stability in hyperparameter settings is that, in practice, we can rely on a relatively small number of tasks to determine optimal hyperparameters when applying Model Breadcrumbs to diverse multi-task
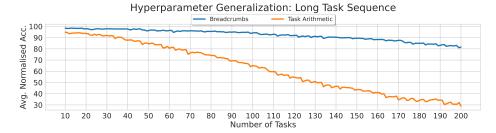
Fig. 4: The 200-task sequence originates from the ImageNet dataset [13], created by dividing the data into 200 5-class classification tasks. After encountering 10 tasks using the ViT-L-14 model, the best hyperparameters for each method (Breadcrumbs with 85% sparsity and Task Arithmetic [21]) are selected 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 [21] by a substantial margin, highlighting the robustness of hyperparameters in the Model Breadcrumbs approach.

learning scenarios. This simplifies the implementation process, reduces the need for extensive hyperparameter tuning, and contributes to the framework's practicality and ease of use. In contrast, Task Arithmetic [21] do not exhibit the same level of hyperparameter stability. Consequently, this fundamental divergence between Model Breadcrumbs and Task Arithmetic [21] underlines the substantial advantage of Model Breadcrumbs in real-world multi-task learning scenarios.

4.4 Effect of Scale

In this section, we explores the impact of using larger CLIP models on our analysis, comparing the performance of ViT-B-32, ViT-B-16, and ViT-L-14 models. For each model type, the optimal Model Breadcrumbs were found at 90% ($\beta = 90\%$, $\gamma = 99\%$), 90% ($\beta = 90\%$, $\gamma = 99.2\%$), 85% ($\beta = 85\%$, $\gamma = 99\%$) sparsity respectively. As shown in Figure 5, the adoption of larger models significantly improves the performance of both our proposed Model Breadcrumbs method and the Task Arithmetic [21] baseline. Moreover, as more tasks are introduced, the capacity to construct better-performing multi-task models grows, with larger-scale models demonstrating superior results.

Specifically, we observe in Figure 5a, when utilizing the ViT-L-14 model and considering 8 tasks, merging Model Breadcrumbs produces a single multi-task model with an average performance that reaches 91.48% of the performance achieved by employing 8 individual fine-tuned models (i.e., one per task). The shift from 8 fine-tuned models to a single multi-task model substantially reduces inference time and compute resources, accompanied by only a minor relative loss in performance. This underscores the practical advantages of our approach.

Moreover, Figure 5b highlights that the performance decline observed when merging either Model Breadcrumbs or Task Arithmetic [21] can be significantly mitigated by adopting larger-scale models. Notably, for the ViT-L-14 model,
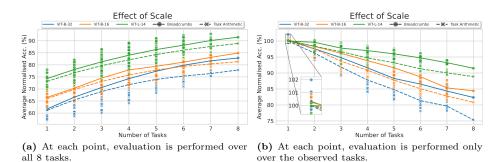
(a) At each point, evaluation is performed over all 8 tasks.

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

Fig. 5: Comparative performance analysis of Model Breadcrumbs and Task Arithmetic [21] 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.

merging Model Breadcrumbs for certain tasks can result in multi-task models that either match or surpass the performance of individual fine-tuned models. To delve deeper into this phenomenon, we conducted a closer examination of task merger for ViT-L-14, considering the two tasks scenario.

As we can see in Figure 6, when adding pairs of tasks via Model Breadcrumbs and Task Arithmetic [21], the merger generally leads to improved performance on both tasks, resulting in a single model that is competitive and often superior to using two specialized fine-tuned models. Furthermore, for the same task pairs, Model Breadcrumbs consistently produces multi-task models that surpass their equivalent Task Arithmetic [21] versions. Notably, Model Breadcrumbs merg-
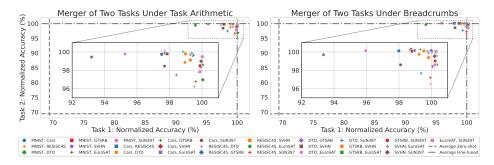

Fig. 6: Comparison of Model Breadcrumbs and Task Arithmetic [21] in the merger of task pairs, revealing improved accuracy on both tasks and a higher frequency of multi-task models surpassing individual fine-tuned accuracy levels when employing Model Breadcrumbs.

Table 2: Model merging enhances the fine-tuned models. Specifically, the merger of Breadcrumbs 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.

Method / Dataset	MRPC	RTE	CoLA	SST-2
Zero-shot	74.8	52.7	8.29	92.7
Fine-tuned	87.9 $\pm$0.68	76.2 $\pm$0.46	51.6 $\pm$0.37	93.3 $\pm$0.35
Fine-tuned + Task Arithmetic	88.6 $\pm$0.59	76.4 $\pm$0.40	52.1 $\pm$0.32	93.5 $\pm$0.31
Fine-tuned + Breadcrumbs	**90.0 $\pm$0.50**	**77.5 $\pm$0.38**	**53.2 $\pm$0.34**	**94.4 $\pm$0.32**

ers generate a higher number of multi-task models where both tasks exceeded their respective fine-tuned accuracy levels. This highlights the potential of Model Breadcrumbs not only to maintain but also to enhance task-specific performance within a multi-task framework. We further examine this concept in the next section.

4.5 Target Task Improvement via Model Merging

Motivated by the insights from Figure 6, we explore the potential of enhancing the performance of a fine-tuned model for a specific target task solely through model merging. We fine-tune the T5-base model [46] for four GLUE tasks [54] (discussed in Section 4.1) based on benchmarks used by [21, 57]. We then merge six publicly available T5-base models (IMDB [35], RACE [27], QASC [24], Multi-News [17], SQuAD [47], and CommonGen [31]) with each of them. Appendix B provides more details on fine-tuned models and the fine-tuning process.

The results, presented in Table 2, evaluates various approaches, including Zeroshot, pure fine-tuning, utilizing Task Arithmetic [21] derived from newly added tasks in the fine-tuned models, and employing Breadcrumbs of the new tasks added to the fine-tuned models. The findings demonstrate that incorporating Breadcrumbs from these new tasks effectively enhances the performance of our fine-tuned models, surpassing all other considered approaches. Importantly, this improvement is achieved without the need for additional training or requiring data from the exact same dataset. This approach underscores the versatility and effectiveness of utilizing Model Breadcrumbs to improve task performance across diverse tasks.

4.6 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
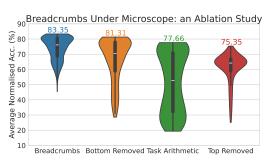
Fig. 7: Performance comparison of the Model Breadcrumbs against alternative masking choices, reveals: Model Breadcrumbs yields a higher distribution of high-performance multi-task models, underlining its robustness towards hyperparameter perturbations. Model Breadcrumbs produces the highest performing multi-task model. The number on top of each violin indicates the performance of the highest performing model of that setting.

Fig. 8: 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.

per layer. 2. Top-Weight Masking: Masking only the 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 [21] 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 4.1. The results are shown in Figure 7.

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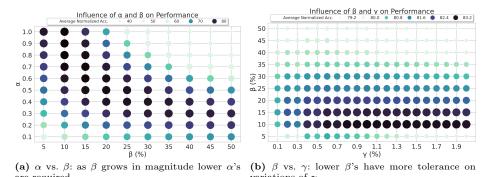
(a) α vs. β: as β grows in magnitude lower α's are required.

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

Fig. 9: 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.

tions underscore the robustness of Model Breadcrumbs to variations in hyperparameter settings, further enhancing its practicality and reliability in real-world applications.

In Figure 8, we examine the cosine similarity between tasks using Model Breadcrumbs and Task Arithmetic [21]. Most tasks show orthogonality, indicating minimal side effects upon merging. However, upon closer inspection, semantically similar tasks (e.g., MNIST [28], SVHN [38], and GTSRB [20]) 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 9, 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 4.1. For ease of readability, in this part we use β and γ to represent how much weights have been masked. Figure 9a 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 9b, 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 7.

5   Conclusions

In this paper, we introduced Model Breadcrumbs, a simple yet effective approach to constructing multi-task models from pre-existing fine-tuned foundation models. Our extensive experiments showcase the method's capability to enhance performance across multiple tasks, demonstrating stable and generalizable hyperparameters. This simplicity makes Model Breadcrumbs practical for real-world multi-task learning scenarios. Additionally, scaling experiments indicate that larger models further benefit from the approach, narrowing the performance gap between merged models and individual fine-tuned ones. Notably, our exploration in NLP data highlights the method's versatility across different modalities.

While promising, Model Breadcrumbs has limitations. Its performance hinges on the quality of the initial fine-tuned models; issues like poor generalization or severe overfitting can propagate. Future research can delve into mitigating these limitations and exploring more sophisticated aggregation techniques for multiple trajectories. Additionally, as the number of tasks increases, considering the expansion of model capacity becomes crucial for sustained high performance.

In conclusion, Model Breadcrumbs stands out for its simplicity, efficiency, and effectiveness in constructing multi-task models. Leveraging publicly available fine-tuned models, it aligns with the trend of updatable machine learning, supporting community-driven model refinement efforts. We anticipate that Model Breadcrumbs will contribute to the development of efficient and scalable multi-task learning solutions in the future.

Acknowledgements

References

1. Ainsworth, S.K., Hayase, J., Srinivasa, S.: Git re-basin: Merging models modulo permutation symmetries. arXiv preprint arXiv:2209.04836 (2022)
2. Asadi, N., Davari, M., Mudur, S., Aljundi, R., Belilovsky, E.: Prototype-sample relation distillation: Towards replay-free continual learning. In: International Conference on Machine Learning. pp. 1093–1106. PMLR (2023)
3. 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)
4. Cheng, G., Han, J., Lu, X.: Remote sensing image scene classification: Benchmark and state of the art. Proceedings of the IEEE **105**, 1865–1883 (Oct 2017)
5. Cho, J., Lei, J., Tan, H., Bansal, M.: Unifying vision-and-language tasks via text generation. In: International Conference on Machine Learning. pp. 1931–1942. PMLR (2021)
6. Choshen, L., Venezian, E., Slonim, N., Katz, Y.: Fusing finetuned models for better pretraining. arXiv preprint arXiv:2204.03044 (2022)
7. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , Vedaldi, A.: Describing textures in the wild. In: Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2014)
8. Cossu, A., Tuytelaars, T., Carta, A., Passaro, L., Lomonaco, V., Bacciu, D.: Continual pre-training mitigates forgetting in language and vision. arXiv preprint arXiv:2205.09357 (2022)
9. Davari, M., Asadi, N., Mudur, S., Aljundi, R., Belilovsky, E.: Probing representation forgetting in supervised and unsupervised continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16712–16721 (2022)
10. Davari, M., Belilovsky, E.: Probing representation forgetting in continual learning. In: NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications (2021)
11. Davari, M., Belilovsky, E.: Model breadcrumbs: Scalable upcycling of finetuned foundation models via sparse task vectors merging. In: ICML 2024 Workshop on Foundation Models in the Wild (2024)
12. Davari, M., Kosseim, L., 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)
13. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., 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)
14. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
15. Dolan, B., Brockett, C.: Automatically constructing a corpus of sentential paraphrases. In: Third International Workshop on Paraphrasing (IWP2005) (2005)
16. Don-Yehiya, S., Venezian, E., Raffel, C., Slonim, N., Katz, Y., Choshen, L.: Cold fusion: Collaborative descent for distributed multitask finetuning. arXiv preprint arXiv:2212.01378 (2022)
17. Fabbri, A.R., Li, I., She, T., Li, S., Radev, D.R.: Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. arXiv preprint arXiv:1906.01749 (2019)

18. Farahnak, F., Mohammadi, E., Davari, M., Kosseim, L.: Semantic similarity matching using contextualized representations. In: Canadian Conference on AI (2021)
19. Helber, P., Bischke, B., Dengel, A., 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)
20. Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., Igel, C.: Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In: International Joint Conference on Neural Networks (2013)
21. Ilharco, G., Ribeiro, M.T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., Farhadi, A.: Editing models with task arithmetic. arXiv preprint arXiv:2212.04089 (2022)
22. Ilharco, G., Wortsman, M., Gadre, S.Y., Song, S., Hajishirzi, H., Kornblith, S., Farhadi, A., Schmidt, L.: Patching open-vocabulary models by interpolating weights. Advances in Neural Information Processing Systems **35**, 29262–29277 (2022)
23. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1803.05407 (2018)
24. Khot, T., Clark, P., Guerquin, M., Jansen, P., Sabharwal, A.: Qasc: A dataset for question answering via sentence composition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 8082–8090 (2020)
25. Kim, W., Son, B., Kim, I.: Vilt: Vision-and-language transformer without convolution or region supervision. In: International Conference on Machine Learning. pp. 5583–5594. PMLR (2021)
26. Krause, J., Stark, M., Deng, J., 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)
27. Lai, G., Xie, Q., Liu, H., Yang, Y., Hovy, E.: Race: Large-scale reading comprehension dataset from examinations. arXiv preprint arXiv:1704.04683 (2017)
28. LeCun, Y., Cortes, C., Burges, C.: Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist **2** (2010)
29. Legate, G., Bernier, N., Caccia, L., Oyallon, E., Belilovsky, E.: Guiding the last layer in federated learning with pre-trained models. arXiv preprint arXiv:2306.03937 (2023)
30. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461 (2019)
31. Lin, B.Y., Zhou, W., Shen, M., Zhou, P., Bhagavatula, C., Choi, Y., Ren, X.: Commongen: A constrained text generation challenge for generative commonsense reasoning. arXiv preprint arXiv:1911.03705 (2019)
32. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
33. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
34. Luo, H., Ji, L., Shi, B., Huang, H., Duan, N., Li, T., Li, J., Bharti, T., Zhou, M.: Univl: A unified video and language pre-training model for multimodal understanding and generation. arXiv preprint arXiv:2002.06353 (2020)

35. Maas, A., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., 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)

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

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

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

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

40. Nguyen, J., Malik, K., Sanjabi, M., Rabbat, M.: Where to begin? exploring the impact of pre-training and initialization in federated learning. arXiv preprint arXiv:2206.15387 (2022)

41. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1717–1724 (2014)

42. 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)

43. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding with unsupervised learning. OpenAI blog (2018)

44. 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)

45. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., 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)

46. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., 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)

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

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

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

50. 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)

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

52. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A.Y., 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)
53. Stoica, G., Bolya, D., Bjorner, J., Hearn, T., Hoffman, J.: Zipit! merging models from different tasks without training. arXiv preprint arXiv:2305.03053 (2023)
54. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.: GLUE: A multi-task benchmark and analysis platform for natural language understanding. In: Linzen, T., Chrupała, G., Alishahi, A. (eds.) Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. pp. 353–355. Association for Computational Linguistics, Brussels, Belgium (Nov 2018). https://doi.org/10.18653/v1/W18-5446, https://aclanthology.org/W18-5446
55. Warstadt, A., Singh, A., Bowman, S.R.: Neural network acceptability judgments. Transactions of the Association for Computational Linguistics **7**, 625–641 (2019)
56. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Transformers: State-of-the-art natural language processing. In: Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations. pp. 38–45 (2020)
57. 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)
58. Wortsman, M., Ilharco, G., Kim, J.W., Li, M., Kornblith, S., Roelofs, R., Lopes, R.G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al.: Robust fine-tuning of zero-shot models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7959–7971 (2022)
59. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., 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)
60. Yadav, P., Tam, D., Choshen, L., Raffel, C., Bansal, M.: Ties-merging: Resolving interference when merging models. In: Thirty-seventh Conference on Neural Information Processing Systems (2023)
61. Yang, Z., Maricar, Y., Davari, M., Grenon-Godbout, N., Rabbany, R.: Toxbuster: In-game chat toxicity buster with bert. arXiv preprint arXiv:2305.12542 (2023)
62. Yin, P., Neubig, G., Yih, W.t., Riedel, S.: Tabert: Pretraining for joint understanding of textual and tabular data. arXiv preprint arXiv:2005.08314 (2020)

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

Table 3: Data statistics.

A   Vision: Data and Training

We fine-tune CLIP models [42] using the datasets specified in Section 4.1 (more details are given in Tabel 3). Following a procedure akin to [22], 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 [33] 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 [22].

B   NLP: Data and Training

We fine-tune the T5-base model [46] on MRPC [15], RTE [54], CoLA [55], and SST-2 [52]. Our fine-tuning process utilizes a batch size of 32, a learning rate of 1e-5, and lasts for 5 epochs 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   TIES Merging

A concurrent study by Yadav *et al.* [60] presents a method named TIES. Like the Task Arithmetic method [21], TIES initially constructs a set of Task Vectors. These vectors undergo a masking process to eliminate interfering weights,

---

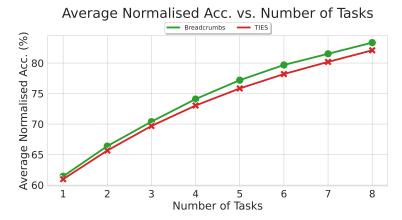[2] `https://huggingface.co/models`

Fig. 10: 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.

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 [36].

In Figure 10, we compare our method to TIES [60], 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 4.1, 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 10, 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.