Adaptive LoRA Merging for Efficient Domain Incremental Learning

Eric Nuertey Coleman[*]
Department of Computer Science
University of Pisa
eric.coleman@phd.unipi.it

Luigi Quarantiello[*]
Department of Computer Science
University of Pisa
luigi.quarantiello@phd.unipi.it

Julio Hurtado
Centre for Applications of Mathematical & Computing Sciences
University of Warwick
julio.hurtado@warwick.ac.uk

Vincenzo Lomonaco
Department of Computer Science
University of Pisa
vincenzo.lomonaco@unipi.it

Abstract

Merging Low-Rank Adaptation (LoRA) has become an essential centre for research and development. However, it is unclear how these methods behave in dynamic scenarios like Domain Incremental Learning (DIL). Here, we address a key limitation of current merging algorithms: their overreliance on fixed weights that usually assume equal importance across tasks. Our method dynamically computes the coefficient for merging, allowing for continuous adaptation to new domains while adjusting the influence of previous ones. We evaluated our approach against some current state-of-the-art merging algorithms on two DIL benchmarks: PACS and OfficeHome. Our results show that the adaptive merging technique achieves performance comparable to or superior to fixed-weight methods while eliminating the need for manual weight selection. In particular, our method maintains high accuracy with minimal memory requirements, using as little as one sample per class for coefficient learning. This work showcases a promising use of LoRA adapters and merging algorithms in continual learning, providing a valuable direction for future research.

1 Introduction

Large pre-trained models have become the default solutions for several applications that require Machine Learning, mainly given the great diversity of problems in which these models can perform effectively. However, these models need to be adapted to perform well in most downstream tasks, due to the fixed and limited training distribution. In classical fine-tuning, all network weights are modified to adapt to the new task, which is highly time-consuming and computationally expensive, considering the massive size of these models. Additionally, the large number of weights that need to be adjusted could easily lead to overfitting.

[*]Equal contribution

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

One alternative to classical fine-tuning are *Parameter-Efficient Fine-Tuning* (**PEFT**) methods. PEFT techniques aim to adapt large models to specific tasks by updating a tiny portion of the network's weights, generally obtaining comparable performance to the fine-tuning approach but much more efficiently [2]. As traditional learning methods, PEFT works under the strong IID assumption, meaning that the training distribution must be kept constant throughout the training process, supposing that the whole set of samples is available from the beginning. This limitation makes it difficult for current models to be used in dynamic contexts, where future distributions can change.

To tackle such constraint, *Continual Learning* (**CL**) [5] focuses on continuously training a model as the training distribution changes. The goal is to make machine learning models capable of absorbing new knowledge while reducing the *catastrophic forgetting* of the previous tasks. Although some papers relate CL to PEFT, they mainly focus on Prompts, leaving out other methods such as LoRA [3].

In this work, we focus on a *Domain-Incremental Learning* (**DIL**) scenario and how to use adapters merging techniques. While different methods have been proposed for merging models, it still needs to be determined which method is most effective in CL settings. This study examines and compares several approaches for merging these continually trained adapters. We also propose a new method to dynamically find the importance of the task when merging the models that adapt to dynamic environments. Our results show that we can achieve comparable or even better results than state-of-the-art methods but with less hyperparameter selection and more robustness in terms of performance.

2 Background

Low Rank Adaptation (LoRA): **LoRA** [3] is a PEFT technique that allows large pretrained models to adapt to downstream tasks by modifying only a small subset of the model's original parameters. LoRA introduces trainable matrices $A \in \mathbb{R}^{r \times d}$ and $B \in \mathbb{R}^{d \times r}$ to model weight updates:

$$W = W_0 + BA \tag{1}$$

where W_0 is the pre-trained weight and $r \ll \min(d, k)$ is the adapter rank. By training these low-rank matrices, LoRA significantly reduces the number of trainable parameters while maintaining comparable performance to full fine-tuning.

Merging Algorithms: Several algorithms have been developed for merging specialized models:

- **DARE** (Drop And REscale) [12]: This technique addresses parameter redundancy by sparsifying and merging fine-tuned models. It focuses on delta parameters ($\delta = W_{fine-tuned} - W_{pre-trained}$), randomly dropping a high percentage (e.g., 90% or 99%) and rescaling the remaining ones by $1/(1-p)$, where p is the drop ratio. DARE enables the merging of multiple task-specific models, potentially outperforming individual source models.
- **TIES-MERGE** [11]: This algorithm addresses interference when combining multiple fine-tuned models through a three-step process: 1) Trimming insignificant parameter changes, 2) Electing signs to resolve conflicts, and 3) Merging parameters aligned with the agreed-upon sign. This approach minimizes interference between task-specific adaptations while maintaining model coherence.
- **Task Arithmetic** [4]: This method introduces Task Vectors, defined as $\Delta W = W_{ft} - W_{pt}$, where W_{ft} and W_{pt} are fine-tuned and pre-trained weights, respectively. These vectors enable operations like negation ($-\Delta W$), addition ($\Delta W_A + \Delta W_B$), and analogy ($(\Delta W_B - \Delta W_A) + \Delta W_C \approx \Delta W_D$), allowing for flexible model behavior modification and knowledge transfer between related tasks without direct training.

Continual Learning CL addresses the challenge of adapting models to a dynamic data distribution arriving in a sequence, mitigating forgetting previously acquired knowledge. CL provides context for how models should learn and adapt to a continuous data stream over time.

This paper focuses on the DIL scenario [8], where a model learns to adapt to a sequence of domains over time. Each domain represents a variation of the same task, such as recognizing objects in different lighting conditions or weather scenarios. DIL is particularly relevant in applications where the same set of classes needs to be recognized across varying domains or conditions, making it crucial for real-world adaptive systems.

3 Merging LoRAs

We propose an adaptive LoRA merging approach for DIL tasks. Specifically, we want to address a limitation of current merging algorithms, *i.e.* the fact that they require fixed weights that are often difficult to select by hand and mostly assume that each task has equal importance, which significantly impacts the network performance.

Given a pre-trained model M and a sequence of domains $\mathcal{D} = \{D_1, D_2, ..., D_N\}$, our goal is to learn a set of LoRA adapters $\mathcal{A} = \{A_1, A_2, ..., A_N\}$ and merging coefficients $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, ..., \alpha_N\}$ that optimize performance across all seen domains. The core of our approach is the dynamical computation of the merging coefficients. For a set of n adapters, our method learns coefficients $\boldsymbol{\alpha} \in \mathbb{R}^n$ is computed as:

$$\alpha_i = \sigma(c_i) \cdot b \tag{2}$$

Algorithm 1 Adaptive LoRA Merging

for $D_k \in \{D_1, \ldots, D_n\}$ **do**
  Train a LoRA adapter A_k on D_k
  Update the merging coefficients to include A_k
  Construct balanced dataset D_{bal} from $\{ D_1, \ldots, D_k \}$
  Learn new coefficients $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_k\}$ by minimizing:

$$\mathcal{L}_{\text{coeff}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{bal}}}[\mathcal{L}_{\text{CE}}(\underbrace{M(x; \sum_{i=1}^{k} \alpha_i A_i)}_{\text{model + merged adapters}}, y)]$$

  Merge adapters: $A_{\text{merged}} = \sum_{i=1}^{k} \alpha_i A_i$
  Evaluate A_{merged} on all seen domains $D_1, \ldots, D_k$
end for

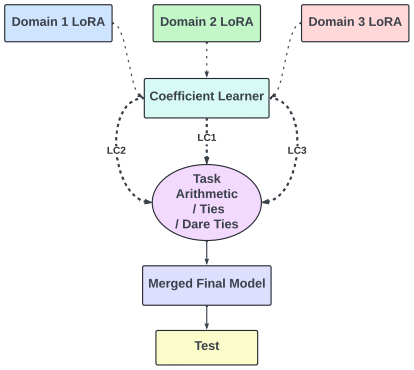

Figure 1: Workflow of the approach

where $\sigma(\cdot)$ denotes the sigmoid function and c_i are learnable parameters. The bias term b is constrained to be positive and bounded:

$$b = 1 + (b_{\text{max}} - 1) \cdot \sigma(b_{\text{raw}}) \tag{3}$$

where b_{raw} is a learnable parameter and b_{max} is a hyperparameter defining the maximum possible bias. This formulation ensures that $0 < \alpha_i < b_{\text{max}}$, allowing for adaptive scaling while preventing excessive growth of coefficient values. Such weights are learned via standard backpropagation, using a balanced subset across all seen domains.

Our incremental learning process is described formally in Algorithm 1, while a graphical overview of the workflow can be seen in Figure 1. This process allows for continuous adaptation to new domains while dynamically adjusting the influence of previous ones. Unlike batch learning approaches, our method doesn't require retraining on data from all domains and maintains a compact model size by merging adapters.

In summary, our adaptive merging can be seen as a generalization of existing methods:

$$A_{\text{adaptive}} = \sum_{i=1}^{k} \alpha_i(\theta) A_i \tag{4}$$

where $\alpha_i(\theta)$ are learned, domain-specific coefficients parameterized by θ. In contrast, linear merging uses fixed weights, while TIES and DARE-TIES use binary or random masks with pruning operators. This adaptive nature allows our method to better capture the relationships between domains and adjust the merging strategy as new domains are encountered, leading to improved performance in domain incremental learning scenarios.

Table 1: Average accuracies (%, ↑) on PACS and OfficeHome. **W** indicates the value of the fixed coefficient for each adapter, while $\boldsymbol{b}_{\max}$ is the maximum possible bias. We used 1 sample per class (*SPC*) to learn the adaptive coefficients. The results are obtained by averaging the accuracies across 3 different runs.

	SPC = 1	*Fixed Coefficients*			*Adaptive Coefficients*		
	Merging Algorithm	**W** = 1	**W** = 3	**W** = 5	$\boldsymbol{b}_{\max} = 1.0$	$\boldsymbol{b}_{\max} = 3.0$	$\boldsymbol{b}_{\max} = 5.0$
PACS	Task Arithmetic	29.19 ± 9.99	11.38 ± 4.76	13.34 ± 6.57	76.48 ± 14.65	29.72 ± 10.15	17.84 ± 4.44
	TIES	82.42 ± 11.76	25.19 ± 9.86	16.14 ± 4.90	62.62 ± 10.10	$\mathbf{82.62 \pm 11.39}$	80.44 ± 12.38
	DARE TIES	20.37 ± 5.02	42.58 ± 8.88	63.25 ± 14.10	16.42 ± 4.93	20.41 ± 5.27	25.02 ± 5.69
OfficeHome	Task Arithmetic	1.83 ± 0.63	1.50 ± 0.29	1.77 ± 0.34	42.09 ± 5.95	2.17 ± 0.58	1.58 ± 0.53
	TIES	$\mathbf{66.83 \pm 6.92}$	2.17 ± 0.42	2.16 ± 0.65	24.88 ± 3.51	$\mathbf{62.95 \pm 5.17}$	62.74 ± 5.83
	DARE TIES	2.65 ± 0.30	13.50 ± 2.32	34.12 ± 4.85	1.68 ± 0.20	2.72 ± 0.21	3.18 ± 0.21

4 Results and Experimental Setup

4.1 Implementation Details

In our experiments, we use a pretrained on ImageNet Vision Transformer (ViT-B/16) [1], obtained through the `timm` library [10]. As baselines, we select three *state-of-the-art* methods: Linear, TIES and DARE+TIES, using the implementation available in the `peft` library from HuggingFace [7]. We assessed existing merging algorithms and our approach in two benchmarks designed explicitly for DIL, **PACS** [6] and **OfficeHome** [9] two real-world datasets with four distinct domains related to the same classification task. Each baseline was evaluated using fixed coefficients (W) for merging, where W represents the weight assigned to each adapter's contribution. We tested $W \in \{1, 3, 5\}$ to demonstrate the sensitivity to this hyperparameter.

4.2 Experimental Results

Firstly, by examining the experiments with fixed coefficients, we can observe in Table 1 that, in both cases, TIES achieves the best overall result, with the value for the weights equal to 1. It is important to note that different coefficient values significantly impact accuracy, and if picked wrongly, this can lead to poor results when working in a DIL scenario. Such lack of robustness requires an extensive manual selection of the hyperparameters, which can be even worse when it is necessary to identify different coefficients for each single task.
To overcome this limitation, our approach learns the importance of each task using a small buffer. Like the fixed coefficients of the baselines, we experimented with different values of $\boldsymbol{b}_{\max}$. The right-hand side of Table 1 shows that TIES achieves the best accuracy, improving the results of the fixed coefficient algorithm in the case of PACS. These results are obtained by saving only 1 example per class in the buffer, showing the efficiency of the proposed method.

To thoroughly analyze our approach, we tested it also with a bigger memory buffer, using 5 samples per class (SPC). The results are reported in Table 2. These results show a slight improvement in most cases, though it comes at a higher computational cost. Nonetheless, such findings suggest that our method does not require large buffers and can perform nearly optimally even with a limited number of samples.

One of the benefits of our method is its dynamic adaptability to different scenarios, identifying the most relevant tasks within the buffer distribution through its learned coefficients. Table 3 shows an example of the evolution of the coefficients value when training our proposed approach with TIES. Our experimental results consistently demonstrate that the cartoon and sketch domains present the most significant challenges in adapter merging, requiring an higher attention from the model, hence higher coefficients (typically $W > 1.0$). Such phenomenon can be primarily attributed to the substantial distribution shift between these testing domains and the natural images used in pre-training. Unlike photos or paintings, which preserve color information and texture details, cartoon images depict objects in a highly abstract fashion, while sketches represent concepts through minimal line drawings. In both cases, most low-level visual features that deep networks rely on are absent, resulting in higher merging coefficients and slightly lower accuracies.

Additionally, standard merging methods have a large hyperparameter space, significantly increasing the computational cost of manually selecting them. Our approach removes the need to select by hand the values of the weights by introducing $\boldsymbol{b}_{\max}$ as a single new value. Looking at the results, we argue

Table 2: Average accuracies (%, ↑) for the PACS dataset. We used 5 samples per class (*SPC*) to learn the adaptive coefficients. The results are obtained by averaging the accuracies across 3 different runs.

SPC = 5	PACS			OfficeHome		
Merging Algorithm	$\boldsymbol{b}_{\max} = 1.0$	$\boldsymbol{b}_{\max} = 3.0$	$\boldsymbol{b}_{\max} = 5.0$	$\boldsymbol{b}_{\max} = 1.0$	$\boldsymbol{b}_{\max} = 3.0$	$\boldsymbol{b}_{\max} = 5.0$
Task Arithmetic	77.88 ± 14.08	30.64 ± 9.86	18.01 ± 4.14	58.91 ± 4.87	42.61 ± 5.13	13.25 ± 2.33
TIES	61.61 ± 8.35	$\mathbf{82.84 \pm 10.53}$	81.63 ± 11.54	18.29 ± 2.29	25.26 ± 2.83	40.29 ± 3.74
DARE TIES	16.35 ± 4.92	20.18 ± 5.20	24.51 ± 5.52	1.61 ± 0.22	1.69 ± 0.20	1.83 ± 0.21

Table 3: Results when merging with TIES and our proposal. Results were obtained using 5 samples per class and $b_{max} = 3$. Experiences represent the sequential addition of domains. Dashes (–) indicate the domain has not yet been added.

Experience	Adapter Weights($\boldsymbol{b}_{\max} = 3.0$)				Domain Accuracy (%)			
	Photo	Cartoon	Sketch	Art	Photo	Cartoon	Sketch	Art
1	1.09	–	–	–	100.00	–	–	–
2	0.94	1.08	–	–	98.98	90.36	–	–
3	0.88	1.05	1.01	–	96.05	86.60	85.19	–
4	0.85	1.07	1.02	0.86	97.72	80.72	76.23	87.40

that $\boldsymbol{b}_{\max}$ makes the model more robust to the hyperparameter selection compared to $\mathbf{W}$, leading to smaller performance variations and higher accuracy in the worst-case scenario. Additionally, our method reduces the complexity of hyperparameter selection by replacing adapter-specific weights with a single $\boldsymbol{b}_{\max}$, significantly decreasing the number of hyperparameters.

5 Conclusion and Future Work

In this study, we explored applying PEFT techniques to DIL tasks. In particular, this work represents the first attempt at applying LoRA adapters and merging algorithms to CL scenarios.
We first analyzed the behavior of existing *state-of-the-art* merging methods when applied to dynamic contexts, finding that TIES achieves the highest accuracy over the other two algorithms. Nonetheless, we observed that such approaches require fixed coefficients, which are challenging to select manually and do not adapt well to evolving environments.
To address this problem, we proposed a *end-to-end* learning algorithm, which learns to select the values of the coefficients based on the relevance of each adapter. We showed that our method performs similarly or even better to the fixed merging approach without needing complex and tedious hyperparameter selection.

This study marks a preliminary step in applying LoRA adapters to CL tasks. In future work, we intend to extend our method to other CL scenarios, such as Class-Incremental Learning, and more complex benchmarks featuring more classes and domains.

6 Ackwoledgements

Work supported by Leonardo Labs, EU EIC project EMERGE (Grant No. 101070918) and PNRR - M4C2 - Investimento 1.3, Partenariato Esteso PE00000013 - "FAIR - Future Artificial Intelligence Research" - Spoke 1 "Human-centered AI", funded by the European Commission under the NextGeneration EU programme.

References

[1] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[2] Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 12799–12807, 2023.

[3] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[4] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic, 2023.

[5] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information fusion*, 58:52–68, 2020.

[6] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

[7] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022.

[8] Haizhou Shi and Hao Wang. A unified approach to domain incremental learning with memory: Theory and algorithm, 2023.

[9] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.

[10] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

[11] Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023.

[12] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch, 2024.