MODEL RATATOUILLE: RECYCLING DIVERSE MODELS FOR OUT-OF-DISTRIBUTION GENERALIZATION

Alexandre Ramé[1,2], **Kartik Ahuja**[1], **Jianyu Zhang**[1,3], **Matthieu Cord**[2,4], **Léon Bottou**[1,3], **David Lopez-Paz**[1]

[1]Meta AI, Paris, France [2]Sorbonne Université, CNRS, ISIR, Paris, France
[3]NYU, New-York, USA [4]Valeo.ai, Paris, France

ABSTRACT

Foundation models are redefining how AI systems are built. Practitioners now follow a standard procedure to build their machine learning solutions: from a pre-trained foundation model, they fine-tune the weights on the target task of interest. So, the Internet is swarmed by a handful of foundation models fine-tuned on many diverse tasks: these individual fine-tunings exist in isolation without benefiting from each other. In our opinion, this is a missed opportunity, as these specialized models contain *rich and diverse* features. In this paper, we thus propose *model ratatouille*, a new strategy to recycle the multiple fine-tunings of the same foundation model on diverse auxiliary tasks. Specifically, we repurpose these auxiliary weights as initializations for multiple parallel fine-tunings on the target task; then, we average all fine-tuned weights to obtain the final model. This recycling strategy aims at maximizing the diversity in weights by leveraging the diversity in auxiliary tasks. Empirically, it improves the state of the art on the reference DomainBed benchmark for out-of-distribution generalization. Looking forward, this work contributes to the emerging paradigm of *updatable machine learning* where the community collaborates to reliably update machine learning models. Our code is released at `https://github.com/facebookresearch/ModelRatatouille/`.

1 INTRODUCTION

The framework of *foundation models* (Bommasani et al., 2021) is fueling a spectacular adoption of machine learning solutions for real-world applications: also known as pre-trained models, these machine learning systems are trained on large-and-diverse data (Fang et al., 2022; Nguyen et al., 2022; Abnar et al., 2022) and easy to adapt to downstream tasks. Having ditched the "training from scratch" mentality, practitioners now follow the two-step transfer learning strategy (Oquab et al., 2014). They first download a foundation model and then fine-tune the weights on their target task by leveraging a limited amount of in-house data. Unfortunately, each of these fine-tunings risks latching onto specific patterns (Arjovsky et al., 2019; Miller et al., 2020). Thus, these shortsighted models struggle to generalize on out-of-distribution (OOD) test examples (Hendrycks & Dietterich, 2019; Taori et al., 2020), which can negatively impacts human lives (Taylor et al., 2016; Zech et al., 2018).

How to best fine-tune foundation models for OOD generalization is thus becoming a central topic of research. In particular, the recently discovered ability to average neural networks' weights (Izmailov et al., 2018; Neyshabur et al., 2020) has inspired a plethora of modern fine-tuning approaches. We illustrate some of them in Figure 1, such as moving averages (Izmailov et al., 2018), WiSE fine-tuning (Wortsman et al., 2022c), model soups (Wortsman et al., 2022b) and DiWA (Ramé et al., 2022). However, these strategies cannot accommodate the swarms of specialized fine-tunings of the same foundation model increasingly available in the Internet. Recent inter-training (Phang et al., 2018; Pruksachatkun et al., 2020) and fusing (Choshen et al., 2022b; Don-Yehiya et al., 2022) strategies recycle intermediate fine-tunings on auxiliary tasks to enrich the features before fine-tuning on the target task. However, the success of these recycling strategies usually depend on the similarity between the auxiliary and target tasks. We also argue in Section 2 that these strategies fail to fully leverage the diversity in auxiliary tasks, even though feature diversity improves OOD generalization (Laakom et al., 2021; Nayman et al., 2022; Jain et al., 2022; Zhang et al., 2022).

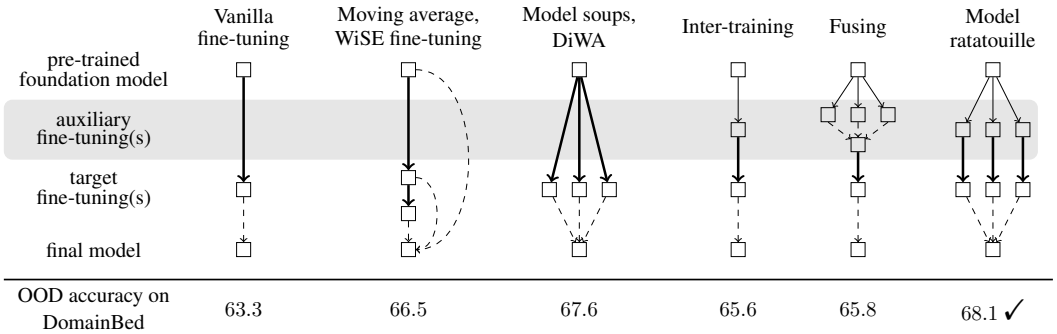


Figure 1: The different fine-tuning strategies discussed in this paper: vanilla fine-tuning (Oquab et al., 2014), moving average (Izmailov et al., 2018) and variants (Wortsman et al., 2022c), model soups (Wortsman et al., 2022b) and DiWA (Ramé et al., 2022), inter-training (Phang et al., 2018), fusing (Choshen et al., 2022b) and our proposed *model ratatouille*. They all start with a pre-trained foundation model. Some strategies fine-tune the pre-trained model on auxiliary tasks (thin solid arrows $\longrightarrow$): these auxiliary fine-tunings can be performed by different contributors of the community on their own data. Then, all strategies perform fine-tuning on the target task of interest (thick solid arrows $\longrightarrow$). Finally, the weights fine-tuned on the target task are used as is, or are averaged (dashed arrows $\dashrightarrow$) into a final model. Ratatouille (i) enables compute parallelism throughout training, (ii) maximizes the amount of diversity in models' predictions, (iii) achieves state-of-the-art performance in DomainBed (Gulrajani & Lopez-Paz, 2021), the standard computer vision benchmark for OOD generalization and (iv) does not incur any inference or training overhead compared to a traditional hyperparameter search.

Thus, the central question of this paper is: *how can we best recycle diverse fine-tunings of a given foundation model towards strong out-of-distribution performance on our target task* ? Our answer is a simple fine-tuning strategy we named *model ratatouille*,[1] illustrated in Figure 1 and described in Section 3. In a similar fashion to converting waste into reusable material for new uses, we take fine-tunings of the same foundation model on diverse auxiliary tasks and recycle them as initializations for multiple fine-tunings on the target task. Specifically, we (i) fine-tune a copy of the foundation model on each auxiliary task, (ii) fine-tune each auxiliary model on the target task, and (iii) return as the final model the average of all fine-tuned weights. In brief, while model soups (Wortsman et al., 2022b) averages multiple weights fine-tuned from a shared initialization, model ratatouille averages multiple weights fine-tuned from different initializations each inter-trained (Phang et al., 2018) on different auxiliary tasks. As we will see, ratatouille works because the fine-tunings remain linearly connected (Frankle et al., 2020; Mirzadeh et al., 2021) in the loss landscape (despite having different initializations) and thus can be averaged for improved performance.

We show the efficacy of model ratatouille in Section 4, where we set a new state of the art on DomainBed (Gulrajani & Lopez-Paz, 2021), the reference benchmark evaluating OOD generalization. We will show how we leverage the diversity across the auxiliary tasks to construct a final model with decreased over-fitting to task-specific patterns. As we discuss in Appendix A, this work contributes to the emerging paradigm of *updatable machine learning* (Raffel, 2021), where practitioners work in collaboration towards incrementally and reliably updating the capabilities of a machine learning model. As also highlighted in recent works (Matena & Raffel, 2022; Li et al., 2022a), we envision a future where deep neural networks are trained by following similar pipelines to the ones in open-source development with version control systems.

2 Fine-tuning for OOD generalization

Setup. We train a deep model $f_\theta = f_w \circ f_\phi$, where the featurizer f_ϕ is with weights ϕ and the classifier f_w is with weights w. We are dealing with out-of-distribution (OOD) generalization where there exists a distribution shift between train and test: our aim is to find θ maximizing the test accuracy $\mathrm{acc}_{\mathrm{te}}(\theta)$. The standard strategy is transfer learning (Oquab et al., 2014; Kirsch et al., 2022; Wenzel et al., 2022) with empirical risk minimization (ERM) (Vapnik, 1992).

[1] We named our method after this traditional French dish for two main reasons. Firstly, the ratatouille is often used as a way to recycle leftover vegetables. Secondly, the ratatouille is better prepared by cooking each ingredient separately before mixing them: this technique ensures that each ingredient "will taste truly of itself", as noted by chef Joël Robuchon (Monaco, 2020).

Weight averaging over epochs. Recently, *weight averaging* strategies came to the foreground (Szegedy et al., 2016; Izmailov et al., 2018; Draxler et al., 2018). While fine-tuning a pre-trained model, they saved and averaged checkpoints every few epochs to build the final model. Due to the nonlinear nature of deep neural networks, the efficacy of weight averaging (Arpit et al., 2021; Cha et al., 2021; Wortsman et al., 2022c; Kaddour, 2022) was a surprising observation, that Frankle et al. (2020) later called the linear mode connectivity.

Observation 1 (LMC with different epochs (Izmailov et al., 2018))**.** *Two weights θ_a and θ_b, obtained at two different epochs of the same fine-tuning, satisfy the linear mode connectivity (LMC): for all $\lambda \in [0, 1]$, $\mathrm{acc_{te}}((1-\lambda)\cdot\theta_a + \lambda\cdot\theta_b) \gtrsim (1-\lambda)\cdot\mathrm{acc_{te}}(\theta_a) + \lambda\cdot\mathrm{acc_{te}}(\theta_b)$.*

Weight averaging over runs. Perhaps motivated by these results, Neyshabur et al. (2020) (along with similar works (Nagarajan & Kolter, 2019; Frankle et al., 2020)) pushed the envelope of weight averaging. They observed that two *independent* fine-tunings—pre-trained similarly but differing in hyperparameter choices, data orders or other stochastic factors—also satisfy the LMC!

Observation 2 (LMC with different runs (Neyshabur et al., 2020))**.** *The LMC holds between θ_a and θ_b fine-tuned on the target task initialized from a shared pre-trained model.*

See Figure 2a for an illustration of Observations 1 and 2. Observation 2 inspired model soups (Wortsman et al., 2022b) and DiWA (Ramé et al., 2022)—the current state-of-the-art approaches for OOD generalization—to average all the weights obtained from a standard ERM hyperparameter search. However, the shared initialization constraint limits models diversity (Kuncheva & Whitaker, 2003; Aksela, 2003), especially when compared to methods that can combine arbitrary networks, for example via prediction averaging in deep ensembles (Lakshminarayanan et al., 2017).

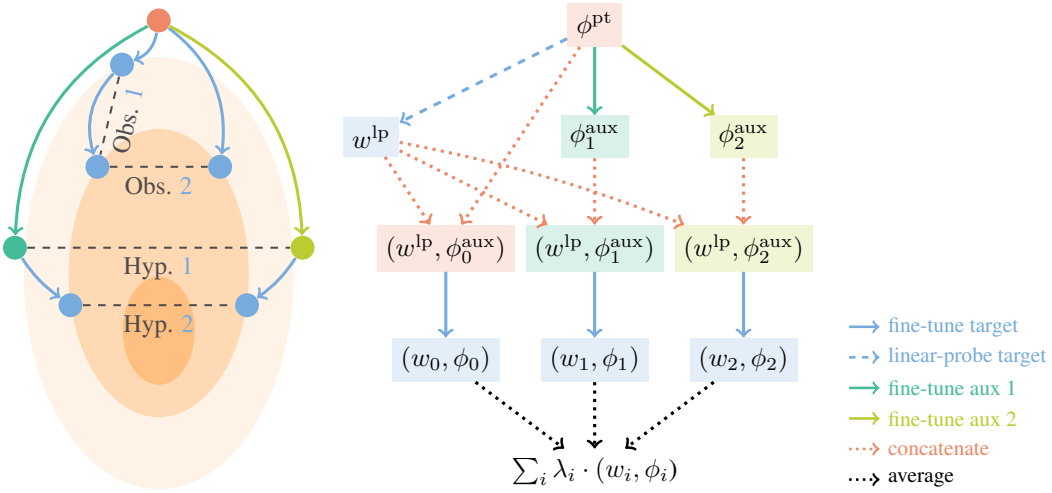
Weight averaging over tasks. *Inter-training* (Phang et al., 2018; Pruksachatkun et al., 2020; Choshen et al., 2022a) performs an intermediate fine-tuning of the pre-trained model on some auxiliary task, before tackling the target task. However, the sequential nature of inter-training leads to catastrophic forgetting (Rebuffi et al., 2017) of some knowledge from the pre-trained model. Moreover, the choice of the auxiliary task plays a determinant role, since "when the wrong task is chosen, inter-training hurts results" (Choshen et al., 2022b). To address these shortcomings, recent works (Choshen et al., 2022b; Don-Yehiya et al., 2022; Li et al., 2022a; Matena & Raffel, 2022) proposed to recycle weights fine-tuned on various auxiliary tasks. In particular, concurrent Choshen et al. (2022b) operates *fusing* to combine into one single initialization the knowledge from multiple auxiliary tasks; they average the auxiliary weights before a single fine-tuning on the target task. Yet, fusing performs badly for OOD generalization on DomainBed in Section 4. We posit that fusing weight average prematurely, destroying most diversity from auxiliary tasks even before the target task can benefit from it. To address these issues, next we propose *ratatouille* which performs one target fine-tuning per auxiliary weights, and averages weights only as the very last step.

3 Model Ratatouille

3.1 Recycling diverse initializations

Our model ratatouille is a proposal to recycle diverse auxiliary fine-tunings of the same pre-trained model; it is compared against other fine-tuning strategies in Figure 1 and outlined in detail in Figure 2b. Ratatouille recycles these fine-tunings as diverse initializations to parallel fine-tunings on the target task, as summarized in these five steps.

1. Download a featurizer ϕ^{pt} pre-trained on task T_0.
2. Fine-tune ϕ^{pt} on each auxiliary task T_i, obtaining $(w_i^{\mathrm{aux}}, \phi_i^{\mathrm{aux}})$ for $0 \leq i < M$.
3. Replace each w_i^{aux} by w^{lp}, obtained by linear probing ϕ^{pt} on the target task T.
4. Fine-tune each $(w^{\mathrm{lp}}, \phi_i^{\mathrm{aux}})$ on the target task T, obtaining $\theta_i = (w_i, \phi_i)$ for $0 \leq i < M$.
5. Return as final model $\sum_{i=0}^{M-1} \lambda_i \cdot \theta_i$. To select the interpolating coefficients, we use the two strategies from model soups (Wortsman et al., 2022b; Ramé et al., 2022). The first "uniform" averages all weights with $\lambda_i = \frac{1}{M}$. The second "greedy" sorts the θ_i by descending accuracy on the in-distribution (ID) validation set, before greedily constructing an uniform average containing θ_i if and only if its addition lowers the ID validation accuracy.



(a) LMC conditions (b) Diagram of model ratatouille. (c) Arrow legend

Figure 2: Illustrations of (a) different linear mode connectivity (LMC) conditions, and (b) model ratatouille. In subplot (a), we illustrate Observation 1, about LMC between two checkpoints along the same target fine-tuning; Observation 2, about LMC between two target fine-tunings; Hypothesis 1, about LMC between two auxiliary fine-tunings; and Hypothesis 2, about LMC between two target fine-tunings initialized from auxiliary weights satisfying Hypothesis 1. Subplot (b) represents our proposed recycling strategy, where we (i) fine-tune a pre-trained model on auxiliary tasks, (ii) plug a linear probe on the auxiliary fine-tunings, (iii) fine-tune on the target task from each auxiliary weights, and (iv) return their weight average as the final model.

Rather than relying on intrinsically good auxiliary tasks (Choshen et al., 2022a), we show in Appendix C that we mostly count on the diversity of initializations and their complementary knowledge. Note that we consider the pre-training task as the auxiliary task "number zero" T_0; this resembles WiSE fine-tuning (Wortsman et al., 2022c) and aims at preserving the general-purpose knowledge contained in the original pre-trained model. More generally, ratatouille connects some fine-tuning strategies from Section 2 while overcoming their limitations. When compared to model soups, model ratatouille removes the shared initialization constraint and thus benefits from the additional diversity brought by the different initializations specialized on various auxiliary tasks. When compared to inter-training (Phang et al., 2018) and fusing (Choshen et al., 2022b), model ratatouille avoids the difficult choice of choosing one single initialization (Choshen et al., 2022a). When compared to fusing, ratatouille delays the weight averaging, and thus the destruction of diversity.

3.2 Novel linear mode connectivity hypotheses

For ratatouille to work, it requires a relaxation of the conditions under which the LMC holds. First, we introduce Hypothesis 1 that posits LMC between two models whose featurizers were fine-tuned on different auxiliary tasks.

Hypothesis 1 (LMC with different tasks)**.** *The LMC holds between* (w, ϕ_a^{aux}) *and* (w, ϕ_b^{aux}) *if* ϕ_a^{aux} *and* ϕ_b^{aux} *are featurizers fine-tuned on two auxiliary tasks initialized from the same pre-trained featurizer* ϕ^{pt}*. Here,* w *is the linear probe of* ϕ^{pt} *on the target task.*

Though this Hypothesis 1 was never formulated explicitly, it underlies fusing (Choshen et al., 2022b) strategy. Ratatouille relies on the following Hypothesis 2, which adds on top of Hypothesis 1 independent fine-tuning steps on the target task.

Hypothesis 2 (LMC with different auxiliary initializations)**.** *The LMC holds between* θ_a *and* θ_b *fine-tuned on the target task starting from initializations* (w, ϕ_a^{aux}) *and* (w, ϕ_b^{aux}) *satisfying Hypothesis 1.*

Hypothesis 2 is the first to posit the LMC between weights with different initializations. We show in Appendix D that Hypotheses 1 and 2 to hold as long as the different tasks are sufficiently similar. In this case, and as diversity was shown to be positively correlated with strong generalization, we expect ratatouille to improve generalization abilities.

4 EXPERIMENTS: SOTA PERFORMANCE ON DOMAINBED

Table 1: Accuracies (%, ↑) on the DomainBed (Gulrajani & Lopez-Paz, 2021) benchmark evaluating OOD generalization. Ratatouille sets a new SoTA by leveraging auxiliary tasks' diversity. The selection column indicates the weight selection strategy. The symbol "$*$" indicates inference overhead in functional ensembling. The symbol "†" indicates the averaging of all weights across 3 data splits.

	Algorithm	Selection	PACS	VLCS	OfficeHome	TerraInc	DomainNet	Avg
	Vanilla fine-tuning	ID val	85.5 ± 0.2	77.5 ± 0.4	66.5 ± 0.3	46.1 ± 1.8	40.9 ± 0.1	63.3
	CORAL (Sun et al., 2016)	ID val	86.2 ± 0.3	78.8 ± 0.6	68.7 ± 0.3	47.6 ± 1.0	41.5 ± 0.1	64.6
	SWAD (Cha et al., 2021)	Loss-aware trajectory	88.1 ± 0.1	**79.1** ± 0.1	70.6 ± 0.2	50.0 ± 0.3	46.5 ± 0.1	66.9
	MA (Arpit et al., 2021)	Uniform trajectory	87.5 ± 0.2	78.2 ± 0.2	70.6 ± 0.1	50.3 ± 0.5	46.0 ± 0.1	66.5
	Deep ensembles* (Arpit et al., 2021)	Uniform	87.6	78.5	70.8	49.2	**47.7**	66.8
DiWA runs	Vanilla fine-tuning	ID val	85.9 ± 0.6	78.1 ± 0.5	69.4 ± 0.2	50.4 ± 1.8	44.3 ± 0.2	65.6
DiWA runs	Ensemble*	Uniform	88.1 ± 0.3	78.5 ± 0.1	71.7 ± 0.1	50.8 ± 0.5	47.0 ± 0.2	67.2
DiWA runs	Model soups	Uniform	88.7 ± 0.2	78.4 ± 0.2	72.1 ± 0.2	51.4 ± 0.6	47.4 ± 0.2	67.6
DiWA runs	Model soups	Greedy	88.0 ± 0.3	78.5 ± 0.1	71.5 ± 0.2	51.6 ± 0.9	**47.7** ± 0.1	67.5
DiWA runs	Model soups†	Uniform†	89.0	78.6	72.8	<u>51.9</u>	**47.7**	68.0
Our runs	Inter-training (Phang et al., 2018)	ID val	89.0 ± 0.0	77.7 ± 0.0	69.9 ± 0.6	46.7 ± 0.1	44.5 ± 0.1	65.6
Our runs	Ensemble* of inter-training	Uniform	89.2 ± 0.1	<u>79.0</u> ± 0.2	72.7 ± 0.1	51.1 ± 0.3	47.2 ± 0.1	67.8
Our runs	Fusing (Choshen et al., 2022b)	ID val	88.0 ± 1.0	78.5 ± 0.8	71.5 ± 0.5	46.7 ± 1.8	44.4 ± 0.2	65.8
Our runs	Model ratatouille	Uniform	89.5 ± 0.1	78.5 ± 0.1	73.1 ± 0.1	51.8 ± 0.4	47.5 ± 0.1	<u>68.1</u>
Our runs	Model ratatouille	Greedy	**90.5** ± 0.2	78.7 ± 0.2	<u>73.4</u> ± 0.3	49.2 ± 0.9	**47.7** ± 0.0	67.9
Our runs	Model ratatouille†	Uniform†	<u>89.8</u>	78.3	**73.5**	**52.0**	**47.7**	**68.3**

Setup. Table 1 shows our main experiment on DomainBed (Gulrajani & Lopez-Paz, 2021), the reference benchmark evaluating OOD generalization with five real-world datasets: PACS (Li et al., 2017), VLCS (Fang et al., 2013), OfficeHome (Venkateswara et al., 2017), TerraIncognita (Beery et al., 2018) and DomainNet (Peng et al., 2019). Each contains multiple domains about the same classification task: each domain is successively considered as the test while others are for training. Standard deviations are obtained on 3 different random data splits. The network is a ResNet-50 (He et al., 2016) pre-trained on ImageNet (Russakovsky et al., 2015). Each strategy leverages 20 runs with hyperparameters sampled from Table 2.

Approaches. Model soups (Wortsman et al., 2022b; Ramé et al., 2022) only differs from vanilla fine-tuning by the selection strategy: rather than selecting the model with highest ID validation accuracy out of the 20 runs, model soups either uniformly averages all weights or greedily selects some—as described in Section 3. For strategies leveraging auxiliary trainings, given a target dataset, we consider the other DomainBed's datasets as the auxiliary tasks. For example when tackling OfficeHome, out of the 20 runs, 4 are inter-trained on PACS, 4 on VLCS, 4 on TerraIncognita, 4 on DomainNet and 4 are directly transferred from ImageNet. Then, *model ratatouille is to inter-training as model soups is to vanilla fine-tuning*. In other words, while inter-training selects the best run based on ID accuracy, ratatouille applies the uniform or the greedy selection. Thus ratatouille provides a single weight averaged network without any inference overhead. For real-world applications, auxiliary weights may be shared by the community; in that case, ratatouille is without training overhead, except when marked by the "†". Indeed, "†" symbol marks methods averaging $60 = 20 \times 3$ weights from 3 data splits, and thus benefiting from larger training budget. We further discuss ratatouille's training cost in Appendix B, and show in Appendix B.3 that ratatouile already performs well with only 5 runs. Ensembling strategies (marked by the symbol "$*$") average predictions with large inference overhead. For example, "ensemble* of inter-training" averages the predictions of the $M = 20$ models ratatouille averages in weights. The experimental setup and the other approaches are described in Appendix F.

Results. Table 1 shows that ratatouille achieves a new SoTA on DomainBed: with uniform selection, it achieves 68.1 and improves model soups by 0.5 points after averaging over all datasets. Precisely, model ratatouille beats model soups by 0.8 and 1.0 points on PACS and OfficeHome with uniform selection, and by 2.5 and 1.9 with greedy selection. On these two datasets, inter-training and fusing also succeed, yet they fail on TerraIncognita (both reach 46.7%) as all auxiliary tasks are distant from photos of animals in the wild; in contrast on TerraIncognita, ratatouille (51.8%) with uniform selection matches model soups (51.4%). This highlights the key strength of our ratatouille w.r.t. other recycling strategies such as fusing: namely, the robustness to the choice of auxiliary tasks. On VLCS, ratatouille is also generally beneficial. For DomainNet, ratatouille is SoTA though the gains are small w.r.t. model soups: we suspect this is because the initialization strategy becomes less critical for larger datasets (Chang & Lu, 2021) with more training epochs (see Figure 6b). In conclusion, even in the absence of similar auxiliary tasks, ratatouille consistently improves generalization on DomainBed.

REFERENCES

Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. In *ICLR*, 2022. (p. 1)

Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint*, 2022. (p. 11)

Matti Aksela. Comparison of classifier selection methods for improving committee performance. In *MCS*, 2003. (p. 3)

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint*, 2019. (p. 1)

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. Ensemble of averages: Improving model selection and boosting performance in domain generalization. In *NeurIPS*, 2021. (pp. 3, 5, 17, 18, and 19)

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in Terra Incognita. In *ECCV*, 2018. (pp. 5 and 17)

Rishi Bommasani and Percy Liang. Reflections on foundation models. https://hai.stanford.edu/news/reflections-foundation-models, 2021. (p. 11)

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint*, 2021. (pp. 1 and 11)

Alexander Borzunov, Dmitry Baranchuk, Tim Dettmers, Max Ryabinin, Younes Belkada, Artem Chumachenko, Pavel Samygin, and Colin Raffel. Petals: Collaborative inference and fine-tuning of large models. *arXiv preprint*, 2022. (p. 11)

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. SWAD: Domain generalization by seeking flat minima. In *NeurIPS*, 2021. (pp. 3, 5, 17, 18, and 19)

Ting-Yun Chang and Chi-Jen Lu. Rethinking why intermediate-task fine-tuning works. *arXiv preprint*, 2021. (p. 5)

Leshem Choshen, Elad Venezian, Shachar Don-Yehia, Noam Slonim, and Yoav Katz. Where to start? analyzing the potential value of intermediate models. *arXiv preprint*, 2022a. (pp. 3, 4, and 11)

Leshem Choshen, Elad Venezian, Noam Slonim, and Yoav Katz. Fusing finetuned models for better pretraining. *arXiv preprint*, 2022b. (pp. 1, 2, 3, 4, 5, 18, and 19)

Shachar Don-Yehiya, Elad Venezian, Colin Raffel, Noam Slonim, Yoav Katz, and Leshem Choshen. ColD fusion: Collaborative descent for distributed multitask finetuning. *arXiv preprint*, 2022. (pp. 1, 3, and 11)

Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht. Essentially no barriers in neural network energy landscape. In *ICML*, 2018. (p. 3)

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. In *ICLR*, 2022. (p. 11)

Alex Fang, Gabriel Ilharco, Mitchell Wortsman, Yuhao Wan, Vaishaal Shankar, Achal Dave, and Ludwig Schmidt. Data determines distributional robustness in contrastive language image pre-training (CLIP). In *ICML*, 2022. (p. 1)

Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *ICCV*, 2013. (pp. 5 and 17)

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *ICML*, 2020. (pp. 2 and 3)

Raphael Gontijo-Lopes, Yann Dauphin, and Ekin Dogus Cubuk. No one representation to rule them all: Overlapping features of training methods. In *ICLR*, 2022. (p. 14)

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2021. (pp. 2, 5, 13, and 18)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. (pp. 5 and 17)

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019. (p. 1)

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *UAI*, 2018. (pp. 1, 2, 3, and 16)

Saachi Jain, Dimitris Tsipras, and Aleksander Madry. Combining diverse feature priors. In *ICML*, 2022. (p. 1)

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint*, 2022. (p. 11)

Keller Jordan, Hanie Sedghi, Olga Saukh, Rahim Entezari, and Behnam Neyshabur. Repair: Renormalizing permuted activations for interpolation repair. In *ICLR*, 2023. (p. 11)

Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. Linear connectivity reveals generalization strategies. *arXiv preprint*, 2022. (p. 15)

Jean Kaddour. Stop wasting my time! saving days of imagenet and BERT training with latest weight averaging. In *NeurIPS Workshop*, 2022. (p. 3)

Andrej Karpathy. Software 2.0. https://karpathy.medium.com/software-2-0-a64152b37c35, 2017. (p. 11)

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. (p. 17)

Andreas Christian Kirsch, Balaji Lakshminarayanan, Clara Huiyi Hu, D. Sculley, Du Phan, Dustin Tran, Jasper Roland Snoek, Jeremiah Liu, Jie Jessie Ren, Joost van Amersfoort, Kehang Han, Kelly Buchanan, Kevin Patrick Murphy, Mark Patrick Collier, Michael W. Dusenberry, Neil Band, Nithum Thain, Rodolphe Jenatton, Tim G. J. Rudner, Yarin Gal, Zachary Nado, Zelda Mariet, Zi Wang, and Zoubin Ghahramani. Plex: Towards reliability using pretrained large model extensions. In *ICML Workshop*, 2022. (p. 2)

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, Tony Lee, Etienne David, Ian Stavness, Wei Guo, Berton Earnshaw, Imran Haque, Sara M Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *ICML*, 2021. (p. 14)

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *ICLR*, 2022. (pp. 15 and 17)

Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 2003. (pp. 3, 13, and 17)

Firas Laakom, Jenni Raitoharju, Alexandros Iosifidis, and Moncef Gabbouj. Within-layer diversity reduces generalization gap. In *ICML Workshop*, 2021. (p. 1)

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017. (p. 3)

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017. (pp. 5 and 17)

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A Smith, and Luke Zettlemoyer. Branch-Train-Merge: Embarrassingly parallel training of expert language models. *arXiv preprint*, 2022a. (pp. 2, 3, 11, and 15)

Qinbin Li, Zeyi Wen, and Bingsheng He. Federated learning systems: Vision, hype and reality for data privacy and protection. *arXiv preprint*, 2019. (p. 11)

Tao Li, Zhehao Huang, Qinghua Tao, Yingwen Wu, and Xiaolin Huang. Trainable weight averaging for fast convergence and better generalization. *arXiv preprint*, 2022b. (p. 11)

Wei-Hong Li, Xialei Liu, and Hakan Bilen. Universal representation learning from multiple domains for few-shot classification. In *ICCV*, 2021. (p. 14)

David Lopez-Paz, Diane Bouchacourt, Levent Sagun, and Nicolas Usunier. Measuring and signing fairness as performance under multiple stakeholder distributions. *arXiv preprint*, 2022. (p. 11)

Ekdeep Singh Lubana, Eric J Bigelow, Robert Dick, David Krueger, and Hidenori Tanaka. Mechanistic lens on mode connectivity. In *NeurIPS Workshop*, 2022. (p. 15)

Michael Matena and Colin Raffel. Merging models with Fisher-weighted averaging. In *NeurIPS*, 2022. (pp. 2, 3, and 11)

John Miller, Karl Krauth, Benjamin Recht, and Ludwig Schmidt. The effect of natural distribution shift on question answering models. In *ICML*, 2020. (p. 1)

John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *ICML*, 2021. (p. 16)

Seyed Iman Mirzadeh, Mehrdad Farajtabar, Dilan Gorur, Razvan Pascanu, and Hassan Ghasemzadeh. Linear mode connectivity in multitask and continual learning. In *ICLR*, 2021. (p. 2)

Emily Monaco. The right way to make ratatouille. `https://www.bbc.com/travel/article/20200812-the-right-way-to-make-ratatouille`, 2020. (p. 2)

Vaishnavh Nagarajan and J Zico Kolter. Uniform convergence may be unable to explain generalization in deep learning. *NeurIPS*, 2019. (p. 3)

Niv Nayman, Avram Golbert, Asaf Noy, Tan Ping, and Lihi Zelnik-Manor. Diverse ImageNet models transfer better. *arXiv preprint*, 2022. (p. 1)

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In *NeurIPS*, 2020. (pp. 1 and 3)

Thao Nguyen, Gabriel Ilharco, Mitchell Wortsman, Sewoong Oh, and Ludwig Schmidt. Quality not quantity: On the interaction between dataset design and robustness of CLIP. In *NeurIPS*, 2022. (p. 1)

Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014. (pp. 1 and 2)

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019. (pp. 5 and 17)

Jason Phang, Thibault Févry, and Samuel R Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint*, 2018. (pp. 1, 2, 3, 4, 5, 18, and 19)

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel Bowman. Intermediate-task transfer learning with pretrained language models: When and why does it work? In *ACL*, 2020. (pp. 1 and 3)

Yujia Qin, Cheng Qian, Jing Yi, Weize Chen, Yankai Lin, Xu Han, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Exploring mode connectivity for pre-trained language models. In *EMNLP*, 2022. (p. 15)

Colin Raffel. A call to build models like we build open-source software. https://colinraffel.com/blog/a-call-to-build-models-like-we-build-open-source-software.html, 2021. (pp. 2 and 11)

Alexandre Ramé and Matthieu Cord. DICE: Diversity in deep ensembles via conditional redundancy adversarial estimation. In *ICLR*, 2021. (p. 13)

Alexandre Ramé, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization. In *NeurIPS*, 2022. (pp. 1, 2, 3, 5, 13, 14, 16, 17, and 18)

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental classifier and representation learning. In *CVPR*, 2017. (p. 3)

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. In *IJCV*, 2015. (p. 5)

Max Ryabinin and Anton Gusev. Towards crowdsourced training of large neural networks using decentralized mixture-of-experts. *NeurIPS*, 2020. (p. 11)

Will Sackfield. SOTAMoon. https://github.com/8W9aG/SOTAMoon, 2021. (p. 11)

Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016. (pp. 5, 18, and 19)

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. (p. 3)

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. In *NeurIPS*, 2020. (p. 1)

J Taylor, B Earnshaw, B Mabey, M Victors, and J Yosinski. RxRx1: An image set for cellular morphological variation across many experimental batches. In *ICLR Workshop*, 2019. (p. 14)

Jessica Taylor, Eliezer Yudkowsky, Patrick LaVictoire, and Andrew Critch. Alignment for advanced machine learning systems. *Ethics of Artificial Intelligence*, 2016. (p. 1)

Damien Teney, Yong Lin, Seong Joon Oh, and Ehsan Abbasnejad. ID and OOD performance are sometimes inversely correlated on real-world datasets. *arXiv preprint*, 2022. (p. 16)

V. Vapnik. Principles of risk minimization for learning theory. In *NeurIPS*, 1992. (p. 2)

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. (pp. 5 and 17)

Florian Wenzel, Andrea Dittadi, Peter Vincent Gehler, Carl-Johann Simon-Gabriel, Max Horn, Dominik Zietlow, David Kernert, Chris Russell, Thomas Brox, Bernt Schiele, Bernhard Schölkopf, and Francesco Locatello. Assaying out-of-distribution generalization in transfer learning. In *NeurIPS*, 2022. (pp. 2 and 16)

Mitchell Wortsman, Suchin Gururangan, Shen Li, Ali Farhadi, Ludwig Schmidt, Michael Rabbat, and Ari S Morcos. lo-fi: distributed fine-tuning without communication. *arXiv preprint*, 2022a. (p. 11)

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, 2022b. (pp. 1, 2, 3, 5, 14, 16, and 18)

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Hanna Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models. In *CVPR*, 2022c. (pp. 1, 2, 3, and 4)

George Udny Yule. On the association of attributes in statistics. *Philosophical Transactions of the Royal Society of London.*, 1900. (p. 13)

John R. Zech, Marcus A. Badgeley, Manway Liu, Anthony B. Costa, Joseph J. Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLOS Medicine*, 2018. (p. 1)

Jianyu Zhang, David Lopez-Paz, and Léon Bottou. Rich feature construction for the optimization-generalization dilemma. In *ICML*, 2022. (p. 1)

Supplementary material

This supplementary material is organized as follows:

- Appendix A discusses the updatable machine learning paradigm.
- Appendix B analyzes ratatouille's components: Appendix B.1 ablates the number of auxiliary tasks, Appendix B.2 ablates the number of target fine-tuning steps and Appendix B.3 ablates the number of target fine-tuning runs.
- Appendix C enriches our diversity experiments.
- Appendix D further empirically analyzes the validity of Hypotheses 1 and 2.
- Appendix E discusses the interest of ratatouille for ID tasks.
- Appendix F further describes our experimental setup on DomainBed.

A Discussion: towards updatable machine learning

In the grand scheme of things, we see model recycling within the emerging *updatable machine learning* (Raffel, 2021) paradigm. The goal is to develop machine learning systems that can be incrementally improved and recombined, allowing for the collaborative creation of increasingly sophisticated AI systems. The core idea is to consider networks as pieces of software (Karpathy, 2017) and mirror the open-source development of software engineering via version control. Could it be possible that, someday, we could build decentralized open-source repositories, where we can clone, commit and merge neural networks towards an ever-improving AI system?

Recent works (Matena & Raffel, 2022; Li et al., 2022a; Don-Yehiya et al., 2022; Choshen et al., 2022a) and the proposed ratatouille give some primitives to learn neural networks in collaboration. Here, (i) cloning is simply weights downloading, (ii) commits are fine-tunings performed by individual contributors on their specific tasks, and (iii) branch merging is replaced by weight averaging. Advanced merging operations (Matena & Raffel, 2022; Li et al., 2022b; Jin et al., 2022) could help to better select the interpolating coefficients λ_i; neuron permutations strategies (Entezari et al., 2022; Ainsworth et al., 2022; Jordan et al., 2023) could remove the need for a shared pre-training, though (so far) these permutations have not improved models' accuracy.

In terms of privacy, such a federated learning setup (Li et al., 2019) where datasets can be kept private does indeed seem desirable. In terms of computation and sustainability, minimal communication across servers enable embarrassingly simple parallelization (Li et al., 2022a; Wortsman et al., 2022a) and could reduce costs and CO2 emissions when training on multiple servers. This paradigm could also leverage the utilization of volunteer computing with single-GPU desktop machines, and complement approaches like Learning@home (Ryabinin & Gusev, 2020) or Petals (Borzunov et al., 2022). Finally, the contributors may potentially be incentivized financially through a system similar to blockchain technology (Sackfield, 2021).

If collaboration is the way forward, how can we ensure the *recyclability* of the shared models? In software engineering, practices such as unit tests greatly reduce the failure modes of programs; how can we borrow these ideas to *specify and test* neural networks? To measure models' shortcomings, we may leverage datasets as *test certificates* (Lopez-Paz et al., 2022). The community would monitor statistics on these datasets, e.g., accuracy, forgetting, and robustness against spurious correlations. Then, the reported scores could guide the choice of what models to clone, fine-tune, and merge. However, bad actors could directly include these datasets in their training data; then, should these external datasets be kept secret by some certifying authority?

These questions are all the more important as traditional foundation models (Bommasani et al., 2021) come with centralization and monetization, raise data privacy concerns, and lack transparency and reproducibility (Bommasani & Liang, 2021), which may hinder the democratization of AI. The ability to collaboratively improve weights represents a shift from *proprietary network training* to

open-source collaborative network building, and could lead to the development of more responsible and reliable AI systems. We see this as an exciting possibility for the future of AI.

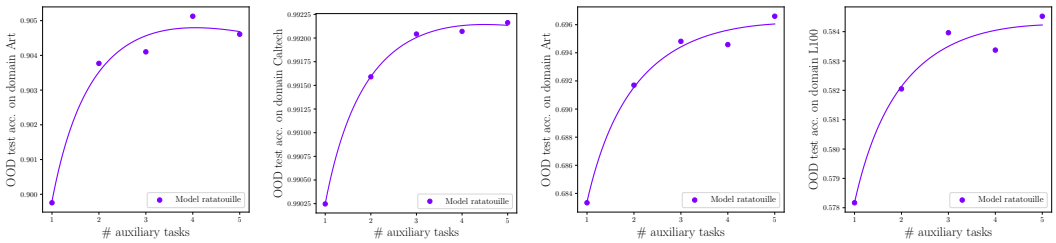
B RATATOUILLE'S COMPONENTS ANALYSIS

In this section we try to refine our understanding of various components in ratatouille.

Remark 1. *If auxiliary weights are shared by the community, recycling strategies cost no more than other fine-tuning strategies: recycling simply benefits from weights that would otherwise ignore each other and be discarded.*

B.1 ANALYSIS OF THE NUMBER OF AUXILIARY TASKS

In our Table 1, ratatouille leverages 5 auxiliary tasks for simplicity: ImageNet (which we consider as the auxiliary task "number zero"), and the 4 other datasets from DomainBed (out of the 5, as we leave out the target task to prevent any information leakage). In following Figure 3, we report the scores obtained using 1 to 5 auxiliary tasks: we always average $M = 20$ weights, the only difference is how they were initialized. When we have 1 auxiliary task, they were all inter-trained on this auxiliary task: when we have 2 auxiliary tasks, 10 are inter-trained on the first auxiliary task, 10 on the second: and etc. This validates that a greater number of auxiliary tasks leads to an increase in expected OOD accuracy. In the paper, we argue that this improvement is a result of the diversity gained through different specializations on different auxiliary tasks. We expect that further increasing the number of auxiliary datasets—beyond those from DomainBed—would further improve results.



(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraIncognita.

Figure 3: OOD accuracy (↑) for model ratatouille when increasing the number of auxiliary tasks and uniformly averaging all fine-tuned weights. For each target task, we consider the first domain as the test OOD; the other domains are used for training.

B.2 ANALYSIS OF THE NUMBER OF TARGET FINE-TUNING STEPS

One could argue that recycling auxiliary weights only benefit from longer training, part of which is delegated to the community. To invalidate this hypothesis, we ablate the number of training steps for model soups and ratatouille in Figure 4, on OfficeHome with "Art" as the OOD domain. We observe that even with unlimited number of training steps, model soups can not beat ratatouille. Therefore ratatouille's gains are made possible by fine-tuning on auxiliary datasets. We also observe that after a large number of epochs, the initialization becomes less important and thus model ratatouille's gain over model soups decreases. In short, using the standard number of training steps (5000) provided by Domainbed is close to optimal.

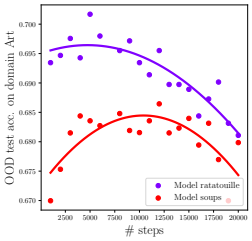
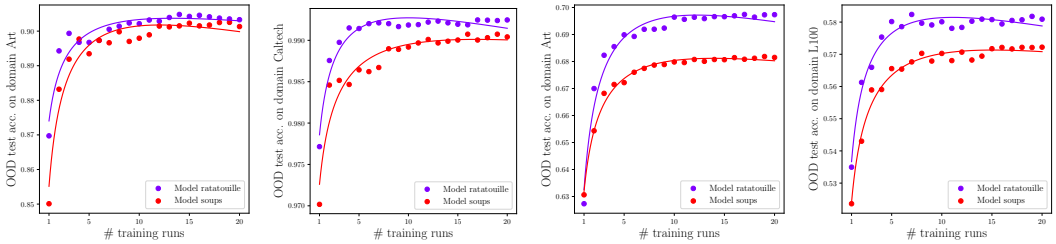


Figure 4: OfficeHome OOD accuracy with uniform averaging at different training steps.

B.3 Analysis of the number of target fine-tuning runs

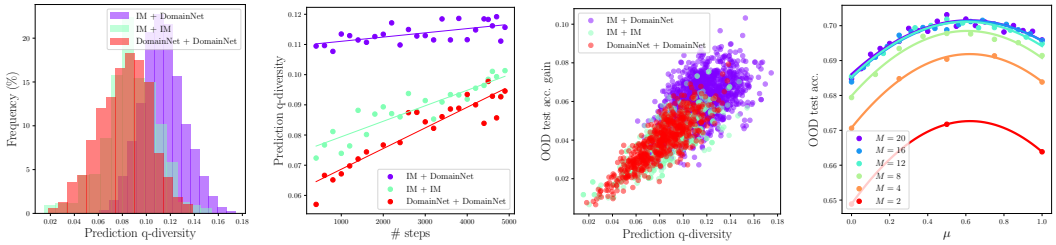
In our main experiment from Table 1, we train and average $M = 20$ independent weights, as 20 is the standard number of hyperparameter trials in DomainBed (Gulrajani & Lopez-Paz, 2021). In Figure 5 we ablate this value. We observe that a larger number of runs improves performance. If reducing the training budget is critical, one could already benefit from significant gains over model soups (and vanilla fine-tuning) with only 5 runs on the target task.



(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraIncognita.

Figure 5: OOD accuracy (↑) for model ratatouille and model soups, when increasing the number of training runs and uniformly averaging all fine-tuned weights. For each target task, we consider the first domain as the test OOD; the other domains are used for training.

C Diversity experiments: increased diversity by recycling



(a) Diversity frequency. (b) Diversity vs. ft steps. (c) Acc. gain vs. diversity. (d) Acc. vs. mixing ratio.

Figure 6: Explorations on q-diversity (Kuncheva & Whitaker, 2003) and its positive impact on accuracy for the OOD test domain "Art" from OfficeHome. In (a), we compute the diversity between pairs of models either directly fine-tuned from ImageNet, either inter-trained on DomainNet: having one model from each initialization increases diversity. In (b), we plot this diversity along the 5k training steps. In (c), we observe that the more diverse the models, the higher the accuracy gain of their weight average compared to the average of their individual accuracies. In (d), we average M models: a proportion $(1 - \mu)$ start directly from ImageNet, the others μ are inter-trained on DomainNet. The accuracy of the weight average is maximized when $\mu \approx 0.5$.

In Figure 6, we investigate how the diversity across models fine-tuned on the target task influences the OOD performance of their weight average. Here, we measure diversity with the prediction q-diversity, which increases when models fail on different examples. Specifically, it is defined by $1 - Q$, where $Q = \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}$ and N^{ij} is the number of times that the first classifier is (correct if $i = 1$ or wrong if $i = 0$) and the second classifier is (correct if $j = 1$ or wrong if $j = 0$). For example, N^{10} is the number of times that the first classifier is correct but not the second. This was introduced in Yule (1900), brought up to date in Kuncheva & Whitaker (2003) and also used in Ramé & Cord (2021).

Following DiWA (Ramé et al., 2022), let the target task be OfficeHome, with "Art" as the test OOD domain; we thus train on the "ClipArt", "Product" and "Photo" domains. We consider models either only pre-trained on ImageNet or also inter-trained on DomainNet.

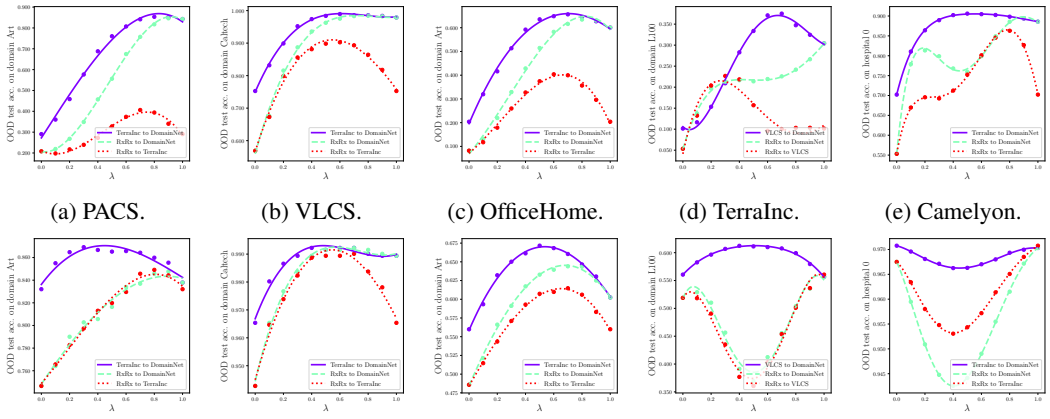
First, we verify that inter-training influences the diversity across fine-tuned models. Specifically, Figure 6a confirms that networks with different initializations are more diverse than networks

initialized similarly. Then, Figure 6b verifies that this diversity gain comes from their initialization and remains along fine-tuning on the target task. Moreover, Figure 6c shows that diversity is positively linearly correlated with OOD generalization: specifically, we observe that having different initializations improves diversity and thus the accuracy of their weight average. Finally, in Figure 6d, we consider averaging M weights: a proportion $(1-\mu)$ start directly from ImageNet, the others μ were inter-trained on DomainNet. In the simplest case $M = 2$, using one model from each initialization leads to maximum accuracy; best performances are obtained around $\mu \approx 0.5$, where the final weight average has access to diverse initializations.

In conclusion, each auxiliary task fosters the learning of diverse features (Li et al., 2021; Gontijo-Lopes et al., 2022). Model ratatouille increases diversity and improves performance by removing a key limitation of model soups approaches (Wortsman et al., 2022b; Ramé et al., 2022); the need for all fine-tunings to start from a shared initialization.

D Linear mode connectivity experiments

D.1 Why ratatouille works



(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraInc. (e) Camelyon.

(f) PACS. (g) VLCS. (h) OfficeHome. (i) TerraInc. (j) Camelyon.

Figure 7: Figures 7a to 7e validate Hypothesis 1 by plotting $\lambda \to \text{acc}_{\text{te}}\big(\big(w^{\text{lp}}, (1-\lambda)\cdot\phi_a^{\text{aux}} + \lambda\cdot\phi_b^{\text{aux}}\big)\big)$, where w^{lp} is the linear probe of $\phi_{\text{IM}}^{\text{pt}}$, and ϕ_a^{aux} and ϕ_b^{aux} are fine-tuned on the two auxiliary datasets in the legend "Dataset$_a$ to Dataset$_b$". Figures 7f to 7j support Hypothesis 2 by plotting $\lambda \to \text{acc}_{\text{te}}\big((1-\lambda)\cdot\theta_a + \lambda\cdot\theta_b\big)$ where θ_a and θ_b are fine-tuned on the target task starting respectively from $(w^{\text{lp}}, \phi_a^{\text{aux}})$ and $(w^{\text{lp}}, \phi_b^{\text{aux}})$. We encounter two exceptions to Hypothesis 2 (Figures 7i and 7j), due to the fact that *neither* the auxiliary *nor* the target task bear enough similarity with the pre-training task.

In Figure 7, we validate Hypotheses 1 and 2 when considering the five datasets from DomainBed. For the sake of completeness, we also analyze some successes and failure cases in "extreme" conditions when considering two distant unrelated medical datasets; RxRx (Taylor et al., 2019) and Camelyon (Koh et al., 2021) from the WILDS (Koh et al., 2021) benchmark. For each target task, we first consider the first domain as the test OOD; the other domains are used for training.

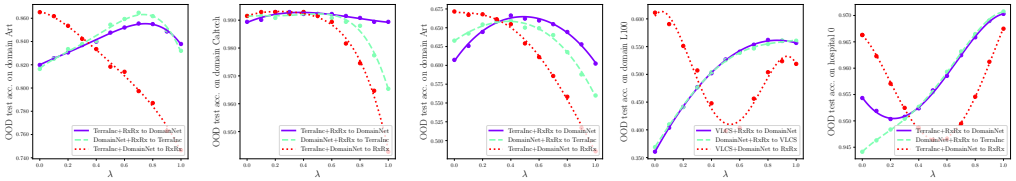
We validate Hypothesis 1 in Figures 7a to 7e. For each dataset, we plot the test OOD accuracy for the weights $\big(w^{\text{lp}}, (1-\lambda)\cdot\phi_a^{\text{aux}} + \lambda\cdot\phi_b^{\text{aux}}\big)$, where the classifier w^{lp} is a linear probe of the ImageNet pre-trained featurizer $\phi_{\text{IM}}^{\text{pt}}$, and $\lambda \in [0,1]$ interpolates between ϕ_a^{aux} and ϕ_b^{aux}, obtained by fine-tuning on two auxiliary tasks initialized from $\phi_{\text{IM}}^{\text{pt}}$. First, we observe that task similarity influences OOD generalization since the test accuracies in Figure 7c agree with the fact that OfficeHome is most similar to DomainNet, not as similar to TerraIncognita, and most dissimilar to the medical dataset RxRx. Second, *the accuracy of the interpolated weights is above the interpolated accuracy*: this validates Hypothesis 1. The accuracy is even usually concave in λ.

Similarly, we empirically support Hypothesis 2 in Figures 7f to 7j. For each dataset, we plot the test OOD accuracy obtained with weights $(1-\lambda)\cdot\theta_a + \lambda\cdot\theta_b$, where the coefficient $\lambda \in [0,1]$ interpolates between θ_a and θ_b, fine-tuned on the target task respectively starting from $(w^{\text{lp}}, \phi_a^{\text{aux}})$ and $(w^{\text{lp}}, \phi_b^{\text{aux}})$. We observe that Hypothesis 2 usually holds: for example, even recycling RxRx can

help for OfficeHome on Figure 7h. Yet, Hypothesis 2 breaks on TerraIncognita and Camelyon in Figures 7i and 7j when RxRx is one of the two auxiliary tasks. In light of these results, we argue *that Hypothesis 2 holds as long as either the auxiliary or the target task is sufficiently similar to the pre-training task*. We speculate this prevents feature distortion (Kumar et al., 2022) and escaping a shared loss valley. Better understanding when LMC breaks is a promising research direction (Juneja et al., 2022; Lubana et al., 2022); among other factors, we speculate that larger pre-training corpus (as in Qin et al. (2022)) or larger architectures (as in Li et al. (2022a)) may favor weight averaging strategies. In Appendix D, we further analyze Hypotheses 1 and 2, notably in a more complex setup where the intermediate tasks are successive fine-tunings on several auxiliary datasets.

D.2 Linear mode connectivity across three weights

In the practical settings from Section 4, model ratatouille averages more than two weight inter-trained on different auxiliary tasks. For consistency, in Figure 8 we thus analyze LMC when interpolating across three fine-tuned weights. We observe the same successes, but also the same occasional failures when the target and task datasets are both simultaneously distant from the pre-training task, i.e., with RxRx as the auxiliary target and either TerraIncognita or Camelyon as the target task.

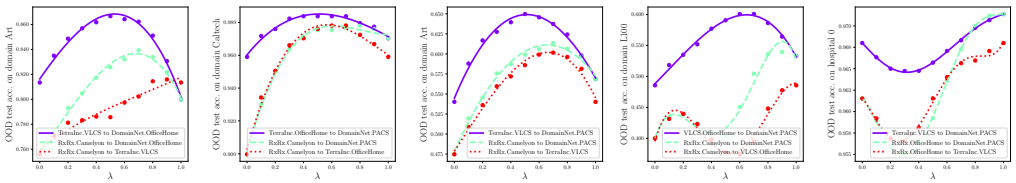


(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraIncognita. (e) Camelyon.

Figure 8: Empirical analysis of LMC when combining three weights. "Dataset$_a$+Dataset$_b$ to Dataset$_c$" means that the model for $\lambda = 0$ is the uniform weight average of θ_a and θ_b (fine-tuned on Dataset$_a$ and respectively on Dataset$_b$ before the target task) while the model θ_c for $\lambda = 1$ was fine-tuned on Dataset$_c$ before the target task; $0 < \lambda < 1$ interpolates between those three fine-tuned weights as $(1-\lambda)/2 \cdot \theta_a + (1-\lambda)/2 \cdot \theta_b + \lambda \cdot \theta_c$. On each target task, we consider the first domain as the test OOD domain.

D.3 Recycling of weights fine-tuned sequentially on multiple datasets

In Figure 9, we empirically analyze Hypothesis 2 when the intermediate tasks are themselves several successive trainings on different auxiliary datasets. Thus the initialization for "TerraInc.VLCS" in Figure 9c was sequentially fine-tuned on TerraIncognita and then on VLCS, before tackling the target task (OfficeHome). The concavity of the curves validates the LMC in most setups. It hints towards a more general inheritance property of LMC: if two initializations satisfy the LMC, then the two fine-tuned weights too. Yet, analysis of this inheritance property is left for future work.



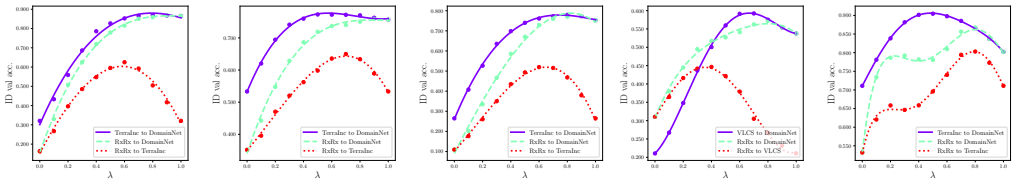
(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraIncognita. (e) Camelyon.

Figure 9: Empirical analysis of Hypothesis 2 when the intermediate tasks are themselves several successive fine-tunings on different auxiliary datasets. "Dataset$_a$.Dataset$_b$ to Dataset$_c$.Dataset$_d$" means that the model for $\lambda = 0$ was sequentially fine-tuned on Dataset$_a$ then Dataset$_b$ before fine-tuning on the target task, while the model for $\lambda = 1$ was sequentially fine-tuned on Dataset$_c$ then Dataset$_d$ before fine-tuning on the target task; $0 < \lambda < 1$ interpolates between those two fine-tuned weights.

E RATATOUILLE FOR ID TASKS

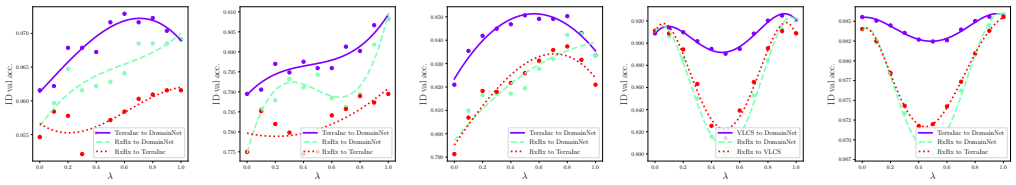
Like previous weight averaging strategies (Izmailov et al., 2018; Wortsman et al., 2022b), ratatouille also works for ID tasks, yet with smaller gains than in OOD. This is what we validate in Figures 10 and 11, where the LMC usually holds in ID (when RxRx is not the auxiliary task), yet with curves less concave than in OOD. These smaller gains when interpolating is because variance reduction via weight averaging is less beneficial in ID than in OOD. Theoretically, this is because, variance is smaller without distribution shift, as explained in Ramé et al. (2022). Empirically, this is consistent with models' diversity being smaller in ID, as shown in Figure 12a. Overall, diversity procedures are less useful in ID than in OOD. The conclusion is a lack of correlation between ID and OOD accuracies (Teney et al., 2022), as shown in Figures 12b and 12c. This finding contrasts with recent works (Miller et al., 2021) and goes against the prescription in Wenzel et al. (2022) that, "to make the model more robust on OOD data, the main focus should be to improve the ID classification error".

In conclusion, when aiming at OOD with ensembling strategies, our experiments suggest that there exists a trade-off between diversity and ID accuracy. This is critical for end-users as OOD is arguably more relevant than ID to ensure applicability in real-world applications, where train and test hardly ever follow the same distributions. This explains occasional failures of the greedy selection (notably for TerraIncognita in Table 1): based on the ID validation accuracy, only a few runs are selected and averaged, causing smaller OOD accuracy than with the uniform selection.



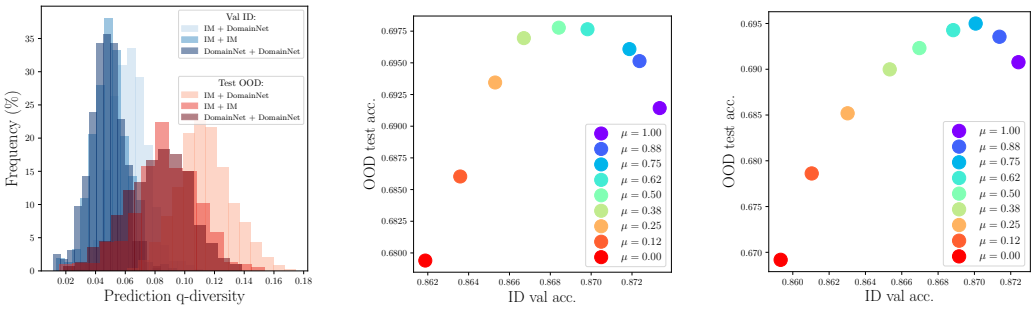
(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraIncognita. (e) Camelyon.

Figure 10: Analysis of Hypothesis 1 on the ID validation split, mirroring the setup from Figures 7a to 7e.



(a) PACS. (b) VLCS. (c) OfficeHome. (d) TerraIncognita. (e) Camelyon.

Figure 11: Analysis of Hypothesis 2 on the ID validation split, mirroring the setup from Figures 7f to 7j.



(a) Diversity frequency. (b) OOD vs. ID acc. (c) OOD vs. ID acc.

Figure 12: Relations between diversity, ID and OOD accuracies. The models were trained on ID domains "Clipart", "Product", and "Photo" from OfficeHome, thus "Art" is the OOD domain. In subplot (a), we compute the diversity (Kuncheva & Whitaker, 2003) between models either directly fine-tuned from ImageNet, either inter-trained on DomainNet. Though having different initializations increases diversity both in ID and in OOD, the diversity in ID remains smaller. In subplot (b), we report the mean results when averaging $M = 8$ weights: $(1 - \mu)$ are fine-tuned on OfficeHome directly from ImageNet, the others μ are inter-trained on DomainNet. We observe a lack of correlation between ID and OOD accuracies in (b). We observe a similar trend in (c), which mirrors the experiment from subplot (b) with the only difference that the proportion $(1 - \mu)$ are inter-trained on PACS (rather than just transferred from ImageNet).

F DomainBed

F.1 Experimental details

Datasets. We tackle the PACS (Li et al., 2017), VLCS (Fang et al., 2013), OfficeHome (Venkateswara et al., 2017), TerraIncognita (Beery et al., 2018) and DomainNet (Peng et al., 2019) datasets. Domains are split into 80% (used as training and evaluation) and 20% (used as validation). When considered as the target task, each domain is successively considered as the test domain while others are for training. When considered as an auxiliary task, we train on all domains simultaneously. Critically, the procedure to obtain the pool of specialized initializations is agnostic to the target task or the test domain, and thus is done only once; for real-world applications, we envision that these weights may be downloaded from collaborative open-source repositories of neural networks.

Training protocol. In all cases, we follow the training protocol from DomainBed. For each dataset, we perform a random search of 20 trials on the mild hyperparameter distributions described in Table 2. We use a ResNet-50 (He et al., 2016) pre-trained on ImageNet, with a dropout layer before the newly added dense layer and fine-tuned with frozen batch normalization layers. The optimizer is Adam (Kingma & Ba, 2015). The linear probe classifier are obtained with default hyperparameters from Table 2 and features extracted from the ImageNet pre-trained featurizer. All runs are trained for 5k steps, except on DomainNet for 15k steps as done in concurrent works (Arpit et al., 2021; Cha et al., 2021; Ramé et al., 2022). When the featurizer was inter-trained on auxiliary datasets, it remains frozen during the first 200 steps to prevent feature distortion (Kumar et al., 2022). As in Ramé et al. (2022); Cha et al. (2021), validation accuracy is calculated every 50 steps for VLCS, 500 steps for DomainNet and 100 steps for others. Our code is released at `https://github.com/facebookresearch/ModelRatatouille/`.

Table 2: Hyperparameters, their default values and distributions for random search.

Hyperparameter	**Default value**	**Random distribution**	
		(DomainBed)	(Ours, DiWA and SWAD)
Learning rate	$5 \cdot 10^{-5}$	$10^{\mathcal{U}(-5,-3.5)}$	$[1, 3, 5] \cdot 10^{-5}$
Batch size	32	$2^{\mathcal{U}(3,5.5)}$	32
ResNet dropout	0	$[0, 0.1, 0.5]$	$[0, 0.1, 0.5]$
Weight decay	0	$10^{\mathcal{U}(-6,-2)}$	$[10^{-6}, 10^{-4}]$

Baselines. Vanilla fine-tuning was named Empirical Risk Minimization in previous papers; CORAL (Sun et al., 2016) is the best invariance-based approach; their scores are taken from DomainBed (Gulrajani & Lopez-Paz, 2021). MA (Arpit et al., 2021) and SWAD (Cha et al., 2021) average weights along the trajectory of a vanilla fine-tuning; their scores are taken from their respective papers. Deep ensembles* averages the predictions of $M = 6$ models, each trained with different classifier initializations on different data splits; the scores are taken from Arpit et al. (2021). Model soups (Wortsman et al., 2022b) averages the weights obtained from different vanilla fine-tunings; for fair comparison, we report the scores achieved in DiWA (Ramé et al., 2022) with linear probing. Fusing averages at initialization 5 auxiliary weights ϕ_i^{aux}; for each of the 20 runs and $0 \leq i < 5$, we sample $\kappa_i \sim \text{Unif}(0, 4)$ and choose $\lambda_i = \frac{e^{\kappa_i}}{\sum_{j=0}^{4} e^{\kappa_j}}$, i.e., the featurizer is initialized from $\sum_{i=0}^{4} \frac{e^{\kappa_i}}{\sum_{j=0}^{4} e^{\kappa_j}} \phi_i^{\text{aux}}$.

Model and weight selection. We consider the training-domain validation set protocol. From each run, we thus take the weights at the epoch with maximum accuracy on the ID validation dataset. The greedy weight selection is also based on this ID validation set. This greedy strategy is not possible for † approaches, that average uniformly the $M = 20 \times 3 = 60$ weights from the 3 data splits: indeed, there is no shared ID validation dataset.

F.2 Results per target dataset and domain

Tables below detail results per domain for the 5 datasets from DomainBed. The average scores were reported in Table 1.

Table 3: Accuracy (%, ↑) on PACS (best in **bold** and second underlined).

	Algorithm	Selection	Art	Cartoon	Photo	Sketch	Avg
	Vanilla fine-tuning	ID val	84.7 ± 0.4	80.8 ± 0.6	97.2 ± 0.3	79.3 ± 1.0	85.5 ± 0.2
	CORAL (Sun et al., 2016)	ID val	88.3 ± 0.2	80.0 ± 0.5	97.5 ± 0.3	78.8 ± 1.3	86.2 ± 0.3
	SWAD (Cha et al., 2021)	Loss-aware trajectory	89.3 ± 0.5	83.4 ± 0.6	97.3 ± 0.3	82.5 ± 0.8	88.1 ± 0.1
	MA (Arpit et al., 2021)	Uniform trajectory	89.1 ± 0.1	82.6 ± 0.2	97.6 ± 0.0	80.5 ± 0.9	87.5 ± 0.2
	Deep ensembles* (Arpit et al., 2021)	Uniform	88.3	83.6	96.5	81.9	87.6
DiWA runs	Vanilla fine-tuning	ID val	86.8 ± 0.8	80.6 ± 1.0	97.4 ± 0.4	78.7 ± 2.0	85.9 ± 0.6
DiWA runs	Ensemble*	Uniform	89.6 ± 0.2	81.6 ± 0.3	97.8 ± 0.2	83.5 ± 0.5	88.1 ± 0.3
DiWA runs	Model soups	Uniform	90.1 ± 0.2	82.8 ± 0.6	98.3 ± 0.1	83.3 ± 0.4	88.7 ± 0.2
DiWA runs	Model soups	Greedy	89.3 ± 0.2	82.8 ± 0.2	98.0 ± 0.1	82.0 ± 0.9	88.0 ± 0.3
DiWA runs	Model soups†	Uniform†	90.6	83.4	98.2	83.8	89.0
Our runs	Inter-training (Phang et al., 2018)	ID val	89.2 ± 1.0	85.3 ± 0.7	97.5 ± 0.0	84.2 ± 0.2	89.0 ± 0.0
Our runs	Ensemble* of inter-training	Uniform	90.4 ± 0.2	83.7 ± 0.3	97.9 ± 0.2	84.9 ± 0.3	89.2 ± 0.1
Our runs	Fusing (Choshen et al., 2022b)	ID val	90.8 ± 0.1	79.1 ± 1.4	98.0 ± 0.4	84.1 ± 2.1	88.0 ± 1.0
Our runs	Model ratatouille	Uniform	90.3 ± 0.2	84.4 ± 0.1	98.7 ± 0.1	84.8 ± 0.1	89.5 ± 0.1
Our runs	Model ratatouille	Greedy	**90.9** ± 0.1	**86.5** ± 1.1	98.6 ± 0.0	**85.9** ± 0.4	**90.5** ± 0.2
Our runs	Model ratatouille†	Uniform†	90.6	84.7	**98.8**	85.0	89.8

Table 4: Accuracy (%, ↑) on VLCS (best in **bold** and second underlined).

	Algorithm	Selection	Caltech	LabelMe	SUN	VOC	Avg
	Vanilla fine-tuning	ID val	97.7 ± 0.4	64.3 ± 0.9	73.4 ± 0.5	74.6 ± 1.3	77.5 ± 0.4
	CORAL (Sun et al., 2016)	ID val	98.3 ± 0.1	**66.1** ± 1.2	73.4 ± 0.3	77.5 ± 1.2	78.8 ± 0.6
	SWAD (Cha et al., 2021)	Loss-aware trajectory	98.8 ± 0.1	63.3 ± 0.3	**75.3** ± 0.5	79.2 ± 0.6	**79.1** ± 0.1
	MA (Arpit et al., 2021)	Uniform trajectory	99.0 ± 0.2	63.0 ± 0.2	74.5 ± 0.3	76.4 ± 1.1	78.2 ± 0.2
	Deep ensembles* (Arpit et al., 2021)	Uniform	98.7	64.5	72.1	78.9	78.5
DiWA runs	Vanilla fine-tuning	ID val	98.1 ± 0.3	64.4 ± 0.3	72.5 ± 0.5	77.7 ± 1.3	78.1 ± 0.5
DiWA runs	Ensemble*	Uniform	98.5 ± 0.1	64.9 ± 0.1	73.4 ± 0.4	77.2 ± 0.4	78.5 ± 0.1
DiWA runs	Model soups	Uniform	98.8 ± 0.1	62.8 ± 0.2	73.9 ± 0.3	78.3 ± 0.1	78.4 ± 0.2
DiWA runs	Model soups	Greedy	98.4 ± 0.0	64.1 ± 0.2	73.3 ± 0.4	78.1 ± 0.8	78.5 ± 0.1
DiWA runs	Model soups†	Uniform†	98.9	62.4	73.9	78.9	78.6
Our runs	Inter-training (Phang et al., 2018)	ID val	98.2 ± 0.0	63.8 ± 0.5	72.3 ± 0.5	76.6 ± 0.2	77.7 ± 0.0
Our runs	Ensemble* of inter-training	Uniform	98.9 ± 0.1	64.7 ± 0.4	73.8 ± 0.5	78.6 ± 0.2	79.0 ± 0.2
Our runs	Fusing (Choshen et al., 2022b)	ID val	98.4 ± 0.4	64.8 ± 1.2	72.2 ± 0.9	78.5 ± 0.6	78.5 ± 0.8
Our runs	Model ratatouille	Uniform	**99.3** ± 0.0	60.8 ± 0.3	74.3 ± 0.3	**79.5** ± 0.3	78.5 ± 0.1
Our runs	Model ratatouille	Greedy	99.0 ± 0.0	62.4 ± 0.5	73.8 ± 0.3	**79.5** ± 0.1	78.7 ± 0.2
Our runs	Model ratatouille†	Uniform†	**99.3**	60.4	73.9	**79.5**	78.3

Table 5: Accuracy (%, ↑) on OfficeHome (best in **bold** and second underlined).

	Algorithm	Selection	Art	Clipart	Product	Photo	Avg
	Vanilla fine-tuning	ID val	61.3 ± 0.7	52.4 ± 0.3	75.8 ± 0.1	76.6 ± 0.3	66.5 ± 0.3
	CORAL (Sun et al., 2016)	ID val	65.3 ± 0.4	54.4 ± 0.5	76.5 ± 0.1	78.4 ± 0.5	68.7 ± 0.3
	SWAD (Cha et al., 2021)	Loss-aware trajectory	66.1 ± 0.4	57.7 ± 0.4	78.4 ± 0.1	80.2 ± 0.2	70.6 ± 0.2
	MA (Arpit et al., 2021)	Uniform trajectory	66.7 ± 0.5	57.1 ± 0.1	78.6 ± 0.1	80.0 ± 0.0	70.6 ± 0.1
	Deep ensembles* (Arpit et al., 2021)	Uniform	65.6	58.5	78.7	80.5	70.8
DiWA runs	Vanilla fine-tuning	ID val	63.9 ± 1.2	54.8 ± 0.6	78.7 ± 0.1	80.4 ± 0.2	69.4 ± 0.2
DiWA runs	Ensemble*	Uniform	67.0 ± 0.1	57.9 ± 0.4	80.0 ± 0.2	81.7 ± 0.3	71.7 ± 0.1
DiWA runs	Model soups	Uniform	68.4 ± 0.2	58.2 ± 0.5	80.0 ± 0.1	81.7 ± 0.3	72.1 ± 0.2
DiWA runs	Model soups	Greedy	67.8 ± 0.5	57.2 ± 0.5	79.6 ± 0.1	81.4 ± 0.4	71.5 ± 0.2
DiWA runs	Model soups†	Uniform†	69.2	59.0	**80.6**	82.2	72.8
Our runs	Inter-training (Phang et al., 2018)	ID val	65.3 ± 0.3	55.8 ± 2.2	78.6 ± 0.1	80.1 ± 0.2	69.9 ± 0.6
Our runs	Ensemble* of inter-training	Uniform	67.8 ± 0.1	60.5 ± 0.1	80.5 ± 0.2	82.0 ± 0.2	72.7 ± 0.1
Our runs	Fusing (Choshen et al., 2022b)	ID val	66.4 ± 0.5	59.8 ± 1.2	78.8 ± 0.2	81.0 ± 0.3	71.5 ± 0.5
Our runs	Model ratatouille	Uniform	69.8 ± 0.1	60.3 ± 0.2	80.4 ± 0.1	81.8 ± 0.2	73.1 ± 0.1
Our runs	Model ratatouille	Greedy	70.0 ± 0.2	**60.8** ± 1.0	**80.6** ± 0.1	82.0 ± 0.2	73.4 ± 0.3
Our runs	Model ratatouille†	Uniform†	**70.4**	60.7	**80.6**	**82.3**	**73.5**

Table 6: Accuracy (%, ↑) on TerraIncognita (best in **bold** and second underlined).

	Algorithm	Selection	L100	L38	L43	L46	Avg
	Vanilla fine-tuning	ID val	49.8 ± 4.4	42.1 ± 1.4	56.9 ± 1.8	35.7 ± 3.9	46.1 ± 1.8
	CORAL (Sun et al., 2016)	ID val	51.6 ± 2.4	42.2 ± 1.0	57.0 ± 1.0	39.8 ± 2.9	47.6 ± 1.0
	SWAD (Cha et al., 2021)	Loss-aware trajectory	55.4 ± 0.0	44.9 ± 1.1	59.7 ± 0.4	39.9 ± 0.2	50.0 ± 0.3
	MA (Arpit et al., 2021)	Uniform trajectory	54.9 ± 0.4	45.5 ± 0.6	60.1 ± 1.5	40.5 ± 0.4	50.3 ± 0.5
	Deep ensembles* (Arpit et al., 2021)	Uniform	53.0	42.6	60.5	40.8	49.2
DiWA runs	Vanilla fine-tuning	ID val	**59.9** ± 4.2	46.9 ± 0.9	54.6 ± 0.3	40.1 ± 2.2	50.4 ± 1.8
DiWA runs	Ensemble*	Uniform	55.6 ± 1.4	45.4 ± 0.4	**61.0** ± 0.4	**41.3** ± 0.3	50.8 ± 0.5
DiWA runs	Model soups	Uniform	56.3 ± 1.9	49.4 ± 0.7	59.9 ± 0.4	39.8 ± 0.5	51.4 ± 0.6
DiWA runs	Model soups	Greedy	58.5 ± 2.2	48.2 ± 0.3	58.5 ± 0.3	41.1 ± 1.2	51.6 ± 0.9
DiWA runs	Model soups†	Uniform†	57.2	50.1	60.3	39.8	51.9
Our runs	Inter-training (Phang et al., 2018)	ID val	49.9 ± 1.7	44.3 ± 1.6	54.7 ± 0.4	37.9 ± 1.1	46.7 ± 0.1
Our runs	Ensemble* of inter-training	Uniform	58.1 ± 0.2	43.8 ± 0.4	**61.0** ± 0.2	**41.3** ± 0.4	51.1 ± 0.3
Our runs	Fusing (Choshen et al., 2022b)	ID val	52.8 ± 3.2	43.2 ± 2.3	55.2 ± 1.3	35.5 ± 0.3	46.7 ± 1.8
Our runs	Model ratatouille	Uniform	57.9 ± 0.2	50.1 ± 0.7	59.8 ± 0.1	38.9 ± 0.5	51.8 ± 0.4
Our runs	Model ratatouille	Greedy	54.0 ± 2.0	47.7 ± 0.8	57.3 ± 0.8	37.9 ± 1.2	49.2 ± 0.9
Our runs	Model ratatouille†	Uniform†	57.9	**50.6**	60.2	39.2	**52.0**

Table 7: Accuracy (%, ↑) on DomainNet (best in **bold** and second underlined).

	Algorithm	Selection	Clipart	Info	Painting	QuickDraw	Photo	Sketch	Avg
	Vanilla fine-tuning	ID val	58.1 ± 0.3	18.8 ± 0.3	46.7 ± 0.3	12.2 ± 0.4	59.6 ± 0.1	49.8 ± 0.4	40.9 ± 0.1
	CORAL (Sun et al., 2016)	ID val	59.2 ± 0.1	19.7 ± 0.2	46.6 ± 0.3	13.4 ± 0.4	59.8 ± 0.2	50.1 ± 0.6	41.5 ± 0.1
	SWAD (Cha et al., 2021)	Loss-aware trajectory	66.0 ± 0.1	22.4 ± 0.3	53.5 ± 0.1	16.1 ± 0.2	65.8 ± 0.4	55.5 ± 0.3	46.5 ± 0.1
	MA (Arpit et al., 2021)	Uniform trajectory	64.4 ± 0.3	22.4 ± 0.2	53.4 ± 0.3	15.4 ± 0.1	64.7 ± 0.2	55.5 ± 0.1	46.0 ± 0.1
	Deep ensembles* (Arpit et al., 2021)	Uniform	68.3	23.1	54.5	16.3	66.9	**57.0**	**47.7**
DiWA runs	Vanilla fine-tuning	ID val	63.4 ± 0.2	21.1 ± 0.4	50.7 ± 0.3	13.5 ± 0.4	64.8 ± 0.4	52.4 ± 0.1	44.3 ± 0.2
DiWA runs	Ensemble*	Uniform	66.7 ± 0.4	22.2 ± 0.1	54.1 ± 0.2	15.1 ± 0.2	68.4 ± 0.1	55.7 ± 0.2	47.0 ± 0.2
DiWA runs	Model soups	Uniform	65.9 ± 0.4	23.0 ± 0.2	55.0 ± 0.3	16.1 ± 0.2	68.4 ± 0.1	55.7 ± 0.4	47.4 ± 0.2
DiWA runs	Model soups	Greedy	66.7 ± 0.2	**23.3** ± 0.2	55.3 ± 0.1	16.3 ± 0.2	68.2 ± 0.0	56.2 ± 0.1	**47.7** ± 0.1
DiWA runs	Model soups†	Uniform†	66.2	**23.3**	55.4	16.5	**68.7**	56.0	**47.7**
Our runs	Inter-training (Phang et al., 2018)	ID val	63.5 ± 0.1	21.1 ± 0.1	51.2 ± 0.2	14.2 ± 0.2	64.7 ± 0.3	52.1 ± 0.1	44.5 ± 0.1
Our runs	Ensemble* of inter-training	Uniform	**66.8** ± 0.2	22.3 ± 0.0	54.2 ± 0.2	15.4 ± 0.2	68.3 ± 0.0	55.8 ± 0.2	47.2 ± 0.1
Our runs	Fusing (Choshen et al., 2022b)	ID val	63.6 ± 0.1	21.3 ± 0.1	51.4 ± 0.2	14.0 ± 0.2	64.1 ± 0.4	52.1 ± 0.3	44.4 ± 0.2
Our runs	Model ratatouille	Uniform	65.9 ± 0.2	23.0 ± 0.1	55.1 ± 0.0	16.5 ± 0.1	68.3 ± 0.0	55.8 ± 0.0	47.5 ± 0.1
Our runs	Model ratatouille	Greedy	66.5 ± 0.1	23.2 ± 0.1	55.3 ± 0.0	**16.7** ± 0.1	68.0 ± 0.0	56.0 ± 0.0	**47.7** ± 0.0
Our runs	Model ratatouille†	Uniform†	66.1	23.1	**55.5**	**16.7**	68.5	56.0	**47.7**