OPT2024: 16th Annual Workshop on Optimization for Machine Learning

WASH: Train your Ensemble with Communication-Efficient Weight Shuffling, then Average

Louis Fournier LOUIS.FOURNIER@ISIR.UPMC.FR
ISIR - Sorbonne Université, Paris - France

Adel Nabli ADEL.NABLI@SORBONNE-UNIVERSITE.FR
Mila, Concordia University, Sorbonne Université

Masih Aminbeidokhti
Marco Pedersoli
École de technologie supérieure, Montréal - Québec

Eugene Belilovsky
Concordia University, Mila – Quebec AI Institute, Montréal - Québec

Edouard Oyallon
Center for Computational Mathematics, Flatiron Institute, New York, USA

Abstract

Deep neural networks' performance is enhanced by ensemble methods, averaging the output of several models at an increased inference cost. Weight averaging methods aim at avoiding this issue by merging the models, but naive averaging results in poor performance for models in different loss basins. Distributed training methods like DART and PAPA have been proposed to train several models in parallel in the same basin but at the cost of ensembling accuracy and significant communication costs between models. We introduce WASH, a novel distributed method that outperforms previous approaches by randomly shuffling a small percentage of model weights during training, for a much lower communication cost.

1. Introduction

In order to enhance the accuracy of a given class of models, the answers of multiple instances trained in parallel can be aggregated via model *ensembling*. This can lead to significant improvements in modern deep learning models [8], at the cost of evaluating multiple instances of a given model during inference, increasing memory and computational requirements, resources that can be critical for on-device inference [27]. To reduce the inference cost back to a single model, the population of models can be fused by averaging their weights [45].

However, there are limits to this method. For models that are too dissimilar, the performance of the averaged model may be no better than chance [14]. This can be mitigated [10, 29] but at the expanse of the ensemble diversity, revealing a trade-off. Inspired by distributed training, techniques such as DART [15] and PAPA [16] propose to train a population of models in parallel on heterogeneous data while communicating to balance this trade-off. DART, similarly to LocalSGD [38], periodically averages all models to avoid model divergence. PAPA controls the diversity of the models more finely by pushing them toward the averaged parameters using an Exponential Moving Average (EMA) like EASGD [48], achieving better performance. In particular, they show that training a population in this way results in better models than using the same compute to train a

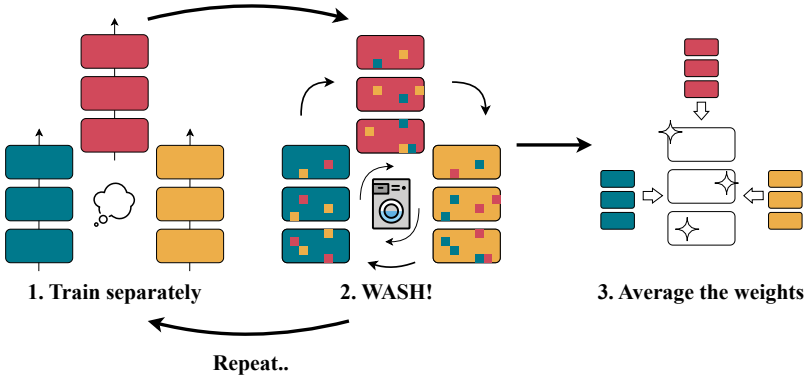

Figure 1: **Representation of training with WASH**. A population of models is being trained separately. **(1)** After each training step, **(2)** a small percentage of the parameters are permuted between models. **(3)** At the end of the training, the model weights are averaged, resulting in a high performance model.

single model, showing the potential of these approaches. However, they generally require a regular computation of the average model using an all-reduce operation, either for the periodic averaging of DART or the EMA used in PAPA. This results in a high communication cost during the parallel training of the model population [31], which hampers the scalability of these approaches as the population size increases [30].

In this work, we propose a novel method to train a population of models in parallel while keeping their weights within the same basin. It requires a fraction of the communication cost of PAPA but exhibits greater model diversity during training, increasing the final averaging accuracy. Our main idea is to randomly shuffle some parameters between models during training, forcing them to learn using the others' parameters. We refer to this idea as "parameter shuffling", distinct from the intra-model weight permutations of [1]. We denote our method, which achieves **W**eight **A**veraging using parameter **SH**uffling, as **WASH**, and represent it schematically in Fig. 1.

Contributions. Our contributions are as follows: **(1)** We propose a novel method to train a population of models that can be weight-averaged. **(2)** We find that it outperforms other methods, at a fraction of their required communication volume. **(3)** We provide experiments and ablations of WASH, notably related to the distance between models and their diversity. **(4)** At the time of publication, we will release an open implementation of WASH.

2. Related work

Ensemble and weight averaging. Ensemble methods significantly improve the ability of a predictive system to make accurate generalizations [6, 21], especially when models exhibit diversity [8, 12]. Model merging by averaging was first explored before deep learning [3, 22, 33]. Then, [14] established that weight averaging is a first-order approximation of the ensemble when models are close in weight space (e.g., with the same pre-trained initialization [29]). Following mode connectivity [10, 11] and the observation that many optima of independent models are connectable, [2, 44] proposed learning simplexes in the parameter space and [34, 46] to train simultaneously several model branches with different last-layer initialization. For models not amenable to weight

averaging, neuron alignment techniques [1, 13, 32, 36] try to match the units of the models, but rarely work in practical scenarios [17]. DART [15] and Branch-Train-Merge (BTM) [23] propose a three-phase training pipeline, with an initial shared training phase followed by parallel training of the models diversified by their data before merging. Iterative refinement of the last 2 stages improves generalization. To enhance the diversity among the models, PAPA [16] gradually adjusts the model weights towards the population average throughout the training process, starting from random initialization. However, these approaches can result in significant communication costs during training, which we address with WASH by only permuting a small fraction of the parameters.

Distributed and federated learning. In the distributed training of deep learning models, the tradeoff between communication and model performance is a core concern [9, 18, 30, 40]. With a limited communication budget, the average distance to consensus is a key metric in decentralized optimization [19, 28, 35, 39, 43]. The techniques discussed earlier for training a population of models for weight averaging are similar to methods in the LocalSGD [25, 37, 38, 42, 48] and Federated Learning [4, 18, 24, 26, 41, 47] literature. Finally, cross-gradient aggregation [7] can be seen as a way of locally shuffling gradients for distributed learning.

3. Parameter shuffling in an ensemble for weight averaging

Motivation of our training procedure. We aim to balance the benefits of model ensembling with the computational efficiency of using a single model for inference via weight averaging. A set of N model parameters $\{\theta_n\}_{n\leq N} \subset \mathbb{R}^d$ are trained in parallel on the same dataset, with different data ordering and possibly different data augmentations and regularizations. To avoid divergence between the models, PAPA [16] applies an EMA every T training steps and produces the following update

$$\tilde{\theta}_n \leftarrow \alpha\theta_n + (1-\alpha)\bar{\theta}\,, \tag{1}$$

where $\bar{\theta} \triangleq \frac{1}{N}\sum_{n=1}^{N}\theta_n$ represents the average of the model weights (the *consensus*), and $\alpha \in [0,1]$ is weighted according to the learning rate. Despite its advantages, this method has drawbacks, including the need for synchronized global communication across all models and the potential reduction in model diversity due to the EMA. Note that after each EMA step

$$\sum_n \|\tilde{\theta}_n - \bar{\theta}\|^2 = \alpha^2 \sum_n \|\theta_n - \bar{\theta}\|^2 < \sum_n \|\theta_n - \bar{\theta}\|^2\,, \tag{2}$$

showing that the models' distance from the consensus is reduced, hindering their diversity.

Proposed method: WASH. To address these challenges, we propose the following stochastic parameter shuffling step instead of the EMA, defined for each individual parameter $\theta_n^j \in \mathbb{R}$ of a model $\theta_n = [\theta_n^j]_{j=1}^d$ by

$$\hat{\theta}_n^i \leftarrow \begin{cases} \theta^i_{\pi_i(n)} & \text{with probability } p, \\ \theta^i_n & \text{otherwise,} \end{cases} \tag{3}$$

where π_i denotes a random permutation of the indices $\{1, ..., N\}$, chosen uniformly at each iteration for each parameter index $i \in \{1, ..., d\}$, and independently from the Bernoulli variable of Eq. (3). Notably, this parameter shuffling reduces in expectation to

$$\mathbb{E}[\hat{\theta}_n] = (1-p)\theta_n + p\bar{\theta}\,. \tag{4}$$

Thus, WASH aligns, in expectation, with the EMA of Eq. (1) for $p = (1 - \alpha)$. The expected number of parameters communicated by each model at each step is thus $p \times d$ while for PAPA, each model communicating all of its parameters every T steps, this amounts to $\frac{d}{T}$. Thus, if $p \ll \frac{1}{T}$, the communication overhead is significantly reduced. However, the model diversity is higher as WASH preserves the consensus distance, as shown by

$$\sum_n \|\hat{\theta}_n - \bar{\theta}\|^2 = \sum_n \sum_i (\hat{\theta}_n^i - \bar{\theta}^i)^2 = \sum_i \sum_n (\theta_n^i - \bar{\theta}^i)^2 = \sum_n \|\theta_n - \bar{\theta}\|^2 \,. \tag{5}$$

Still, the following optimization step will affect the consensus distance, as we will see later.

Layer-wise adaptation via WASH. We also introduce a specific and linearly decreasing layer probability. Assuming L layers in the network, for each layer $l \in [0, L-1]$ we set

$$p_l = p \left(1 - \frac{l}{L-1}\right) \,, \tag{6}$$

where p is a base probability. Thus, the faster-to-train early layers will be shuffled more often than the final ones, which also halves the communication volume. Other adaptations are studied in the Appendix.

Full procedure. Alg. 1 in the Appendix presents the training of a population of N models using WASH. Starting from the same initialization, our training procedure alternates between local gradient computation and shuffling communication. At inference, we average the weights of the models to obtain a single model with parameters $\bar{\theta}$. We found techniques such as REPAIR [17] or activation alignment [1] to be unnecessary to achieve high accuracy.

4. Experiments

Training methods. We present the capabilities of WASH for training a population of models on standard image classification tasks. We compare to a Baseline where the population is trained separately, as in [16]. We also compare WASH to PAPA [16] on the same tasks, to show our improvement despite the lower communication volume. We do not provide comparisons with DART [15] or the variants of PAPA (like PAPA-all, equivalent to DART) as their performances are generally inferior [16]. We also propose a variant of WASH called WASH+Opt, which also permutes the optimizer state associated with the shuffled parameter (here, SGD's momentum), doubling the communication volume.

Communication cost. While PAPA requires an all-reduce operation every $T = 10$ training steps, WASH only requires, in expectation, a shuffling of $p/2$ of the parameters. In practice, in our experiments, p is equal to 0.001 or 0.05, ensuring a reduction in communication volume of 200 or 4. We summarize in Tab. 1 the communication volume and inference costs required for the 4 methods.

Evaluation strategy. We evaluate in three ways the resulting population of models trained. By averaging the predictions, we evaluate the Ensemble as a baseline. By averaging the weights, we refer to the merged model as Averaged (i.e. UniformSoup [45] or AvgSoup [16]). Finally, a more elaborate averaging scheme, GreedySoup [45], averages models to greedily increase the validation accuracy. As reported in [16], this corresponds in practice to only the accuracy of the best model, and thus only report it for the Baseline.

Table 1: **Communication volume and inference costs** of the 4 methods. The Baseline Ensemble requires a linearly increasing inference cost but no communication. WASH and WASH+Opt require much less communication than PAPA for training.

	Communication volume		
Technique	**CIFAR-10/100**	**ImageNet**	**Inference cost**
Ensemble	0	0	N
PAPA	1	1	1
WASH	$1/200$	$1/4$	1
WASH+Opt	$1/100$	$1/2$	1

Table 2: **Ensemble and Averaged Model accuracy for a heterogeneous population of models.** We compare models trained separately (Baseline), with PAPA, and our methods WASH and WASH+Opt. The best Ensemble (black) and Averaged (blue) accuracy are reported in bold. Except on CIFAR-10, WASH and WASH+Opt outperform PAPA and provide performances comparable to the Ensembles.

Method		**Baseline (trained separately)**			**PAPA**		**WASH (ours)**		**WASH+Opt (ours)**	
Config	**#N**	Ensemble	Averaged	GreedySoup	Ensemble	Averaged	Ensemble	Averaged	Ensemble	Averaged
CIFAR-10										
VGG-16	3	95.98±.42	10.00±.00	95.26±.05	**96.12±.34**	**96.13±.24**	95.89±.23	95.97±.24	95.91±.36	95.85±.27
	5	**96.28±.40**	10.00±.00	95.42±.10	96.24±.17	**96.21±.13**	96.15±.10	**96.20±.10**	96.00±.21	96.04±.14
	10	**96.47±.07**	10.00±.00	95.39±.24	96.32±.13	**96.31±.13**	96.27±.10	96.18±.13	96.14±.08	96.20±.05
ResNet18	3	97.15±.28	10.17±.29	96.62±.38	**97.33±.05**	**97.24±.05**	97.21±.19	97.19±.17	97.22±.07	**97.25±.14**
	5	**97.33±.08**	10.09±.16	96.61±.03	**97.35±.12**	**97.31±.06**	97.21±.10	97.25±.12	97.18±.09	97.16±.07
	10	**97.59±.01**	9.26±1.28	96.79±.14	97.39±.13	**97.34±.06**	97.30±.10	97.28±.04	97.20±.13	97.16±.13
CIFAR-100										
VGG-16	3	**80.36±.15**	1.00±.00	77.92±.22	78.89±.10	78.77±.16	79.10±.88	79.05±.68	79.15±.61	**79.15±.41**
	5	**81.32±.56**	1.00±.00	77.81±.25	79.51±.38	79.24±.43	79.65±.27	79.39±.21	79.75±.21	**79.71±.20**
	10	**82.24±.15**	1.00±.00	77.83±.65	79.95±.11	79.64±.13	80.05±.18	79.70±.25	80.03±.11	**79.76±.13**
ResNet18	3	**82.84±.48**	1.00±.01	80.06±1.5	81.58±.12	81.53±.13	81.91±.34	81.90±.36	81.99±.06	**82.08±.09**
	5	**83.72±.49**	1.00±.00	80.72±.52	82.09±.30	82.01±.34	82.16±.42	81.97±.28	82.35±.17	**82.17±.15**
	10	**84.18±.20**	1.00±.00	80.61±.43	82.32±.09	82.15±.14	82.43±.32	**82.31±.38**	82.42±.31	82.18±.22
ImageNet										
ResNet50	3	**76.16±.28**	0.10±.00	74.15±.11	75.62±.15	10.32±2.4	74.39±.14	**74.34±.18**	74.30±.22	74.18±.26
	5	**76.68±.06**	0.10±.00	74.47±.06	75.80±.21	0.13±0.04	74.63±.11	**74.59±.07**	74.44±.21	74.39±.21

4.1. Main experiments

Experimental setup. We showcase the performance of WASH on image classification tasks on the same framework as [16] for a fair comparison, either on heterogeneous or homogeneous dataset environments. The training details are in the Appendix.

Main results. We report in Tab. 2 the average test accuracies of 3 runs for the Ensemble, the Averaged model, and the GreedySoup for the Baseline in heterogeneous settings. The homogeneous results are in the Appendix. As in [16], the Baseline models have a high Ensemble accuracy but perform as random when Averaged. PAPA and WASH result in lower Ensemble accuracy, but almost no difference with the Averaged accuracy. In general, WASH and WASH+Opt outperform PAPA, for less communication. On ImageNet, we were not able to reproduce PAPA's baseline, possibly due to

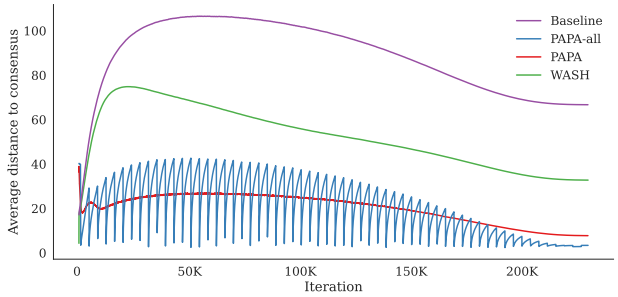

Figure 2: **Average distance to the consensus** during training. Starting at 0 distance, models initially diverge before converging back, mainly due to weight decay. WASH incurs a smaller distance than the Baseline, putting the models in the same basin. Training with PAPA-all (i.e., periodically averaging the models) forbids the population from reaching high diversity. The EMA of PAPA shows a similar strong pulling effect towards consensus.

a mistake in their reported hyperparameters (See Appendix E.3). Both of our methods reduce the accuracy gap with the Baseline Ensemble, indicating that WASH hinders less the models' diversity while maintaining averagability. The remaining gap may be inherent to having the models in the same basin. WASH and WASH+Opt have very similar results, with the simpler WASH being better in the homogeneous case and WASH+Opt in the heterogeneous one.

4.2. Why do shuffling parameters help?

We propose now to explain why our parameter shuffling approach improves other previous mechanisms, by showing that WASH reduces the distance to consensus enough for weight averagability, while also inducing diversity in the models.

Reduced distance to consensus. To better analyze the WASH models' diversity, we report in Fig. 2 the distance of the models to the consensus during training as a proxy for the diversity metric (see Appendix D), as [14, 46] showed it is correlated to the performance gap between the Ensemble and Averaged model. WASH results in a consistently lower distance to consensus than the Baseline, despite the non-compressive shuffling. In comparison, PAPA and PAPA-all (i.e. DART) pull the models toward the consensus too strongly.

Encouraging diversity. The weight shuffling is a weak perturbation of the models, as only a few parameters are affected, and the consensus distance is unchanged. We illustrate in Appendix E.2 how the shuffling increases the diversity of the models' optimization trajectories on a 2D optimization example. We find that while PAPA may allow points to reach consensus, they can get stuck on local minima. In contrast, WASH allows them to reach the global minimum due to a greater path diversity.

Conclusion

We proposed a novel distributed training method, WASH, training a population of models that can be averaged into a high-performance model. Our new parameter shuffling approach does not explicitly reduce the distance between models and increase their diversity, for a fraction of the cost of traditional approaches.

References

[1] Samuel K. Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries, 2022.

[2] Gregory Benton, Wesley Maddox, Sanae Lotfi, and Andrew Gordon Gordon Wilson. Loss surface simplexes for mode connecting volumes and fast ensembling. In *International Conference on Machine Learning*, pages 769–779. PMLR, 2021.

[3] Léon Bottou, Frank E Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM review*, 60(2):223–311, 2018.

[4] Minghui Chen, Meirui Jiang, Qi Dou, Zehua Wang, and Xiaoxiao Li. Fedsoup: Improving generalization and personalization in federated learning via selective model interpolation, 2023.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[6] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[7] Yasaman Esfandiari, Sin Yong Tan, Zhanhong Jiang, Aditya Balu, Ethan Herron, Chinmay Hegde, and Soumik Sarkar. Cross-gradient aggregation for decentralized learning from non-iid data. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3036–3046. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/esfandiari21a.html.

[8] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[9] Louis Fournier and Edouard Oyallon. Cyclic data parallelism for efficient parallelism of deep neural networks, 2024.

[10] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis, 2020.

[11] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/be3087e74e9100d4bc4c6268cdbe8456-Paper.pdf.

[12] Raphael Gontijo-Lopes, Yann Dauphin, and Ekin D Cubuk. No one representation to rule them all: Overlapping features of training methods. *arXiv preprint arXiv:2110.12899*, 2021.

[13] Stefan Horoi, Albert Manuel Orozco Camacho, Eugene Belilovsky, and Guy Wolf. Harmony in diversity: Merging neural networks with canonical correlation analysis. In *International Conference on Machine Learning (ICML)*, 2024.

[14] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization, 2019.

[15] Samyak Jain, Sravanti Addepalli, Pawan Sahu, Priyam Dey, and R. Venkatesh Babu. Dart: Diversify-aggregate-repeat training improves generalization of neural networks, 2023.

[16] Alexia Jolicoeur-Martineau, Emy Gervais, Kilian Fatras, Yan Zhang, and Simon Lacoste-Julien. Population parameter averaging (papa), 2023.

[17] Keller Jordan, Hanie Sedghi, Olga Saukh, Rahim Entezari, and Behnam Neyshabur. Repair: Renormalizing permuted activations for interpolation repair, 2023.

[18] Jakub Konecný, H. B. McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *ArXiv*, abs/1610.02527, 2016. URL https://api.semanticscholar.org/CorpusID:2549272.

[19] Lingjing Kong, Tao Lin, Anastasia Koloskova, Martin Jaggi, and Sebastian Stich. Consensus control for decentralized deep learning. In *International Conference on Machine Learning*, pages 5686–5696. PMLR, 2021.

[20] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. URL https://api.semanticscholar.org/CorpusID:18268744.

[21] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[22] Chandrashekar Lakshminarayanan and Csaba Szepesvari. Linear stochastic approximation: How far does constant step-size and iterate averaging go? In *International Conference on Artificial Intelligence and Statistics*, pages 1347–1355. PMLR, 2018.

[23] 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, 2022.

[24] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization for heterogeneous networks. In *ICML Workshop on Adaptive & Multitask Learning: Algorithms & Systems*, 2019. URL https://openreview.net/forum?id=SkgwE5Ss3N.

[25] Tao Lin, Sebastian U. Stich, Kumar Kshitij Patel, and Martin Jaggi. Don't use large mini-batches, use local sgd. In *International Conference on Learning Representations*, 2020.

[26] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017. URL https://proceedings.mlr.press/v54/mcmahan17a.html.

[27] Gaurav Menghani. Efficient deep learning: A survey on making deep learning models smaller, faster, and better. *ACM Computing Surveys*, 55(12):1–37, 2023.

[28] Adel Nabli, Eugene Belilovsky, and Edouard Oyallon. A2cid2: Accelerating asynchronous communication in decentralized deep learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=YE04aRkeZb.

[29] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 512–523. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0607f4c705595b911a4f3e7a127b44e0-Paper.pdf.

[30] Jose Javier Gonzalez Ortiz, Jonathan Frankle, Mike Rabbat, Ari Morcos, and Nicolas Ballas. Trade-offs of local sgd at scale: An empirical study, 2021.

[31] Suchita Pati, Shaizeen Aga, Mahzabeen Islam, Nuwan Jayasena, and Matthew D. Sinclair. Computation vs. communication scaling for future transformers on future hardware, 2023.

[32] Fidel A. Guerrero Peña, Heitor Rapela Medeiros, Thomas Dubail, Masih Aminbeidokhti, Eric Granger, and Marco Pedersoli. Re-basin via implicit sinkhorn differentiation, 2022.

[33] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[34] Alexandre Ramé, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization, 2022.

[35] Wei Shi, Qing Ling, Gang Wu, and Wotao Yin. Extra: An exact first-order algorithm for decentralized consensus optimization, 2014.

[36] Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport, 2023.

[37] Artin Spiridonoff, Alex Olshevsky, and Ioannis Ch. Paschalidis. Communication-efficient sgd: From local sgd to one-shot averaging, 2021.

[38] Sebastian U. Stich. Local sgd converges fast and communicates little, 2019.

[39] Hanlin Tang, Xiangru Lian, Ming Yan, Ce Zhang, and Ji Liu. D^2: Decentralized training over decentralized data, 2018.

[40] Guanhua Wang, Heyang Qin, Sam Ade Jacobs, Connor Holmes, Samyam Rajbhandari, Olatunji Ruwase, Feng Yan, Lei Yang, and Yuxiong He. Zero++: Extremely efficient collective communication for giant model training, 2023.

[41] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging, 2020.

[42] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkxJ8REYPH.

[43] Zhiguo Wang, Jiawei Zhang, Tsung-Hui Chang, Jian Li, and Zhi-Quan Luo. Distributed stochastic consensus optimization with momentum for nonconvex nonsmooth problems. *IEEE Transactions on Signal Processing*, 69:4486–4501, 2021. ISSN 1941-0476. doi: 10.1109/tsp.2021.3097211. URL http://dx.doi.org/10.1109/TSP.2021.3097211.

[44] Mitchell Wortsman, Maxwell Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. Learning neural network subspaces, 2021.

[45] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/wortsman22a.html.

[46] Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo-Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models, 2022.

[47] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7252–7261. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/yurochkin19a.html.

[48] Sixin Zhang, Anna Choromanska, and Yann LeCun. Deep learning with elastic averaging sgd. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 685–693, Cambridge, MA, USA, 2015. MIT Press.

Appendix A. Algorithm

In Alg. 1, we showcase the distributed training procedure with WASH.

Algorithm 1 Training with WASH

Input: Datasets D_i, number of models N, initial parameters θ_0, training steps T, number of layers L, base probability p
Initialize parameters $(\theta_n)_n \leftarrow \theta_0$ and optimizers OPT_i
for $t = 1$ to T **do**
  # Training step
  for $n = 1$ to N, in parallel **do**
    $(x_n, y_n) \leftarrow D_n$ *# Sample data*
    $\theta_n \leftarrow \text{OPT}_n(x_n, y_n, \theta_n)$ *# Update the model n*
  # Shuffling step
  for layer $l = 0$ to $L - 1$ **do**
    for parameter θ^i in layer l **do**
      With probability $p(1 - \frac{l}{L-1})$,
        $\pi_i \leftarrow$ Random permutation
        $(\theta^i_n)_n \leftarrow (\theta^i_{\pi_i(n)})_n$ *# Send and permute the parameter*
Output: the averaged model $\frac{1}{N}\sum_{n=1}^{N} \theta_n$

Appendix B. Experimental setup

B.1. Framework details

We showcase the performance of WASH for training neural networks on image classification tasks on the CIFAR-10, CIFAR-100 [20], and ImageNet [5] datasets. We use the same training framework as [16] for a fair comparison. We train a population of N models for $N \in \{3, 5, 10\}$, on the ResNet-18, 50 and VGG-16 architectures. 2% of the training data is kept as validation for computing the GreedySoup. As in [16], we consider one framework with heterogeneous models, learning with different data augmentations and regularizations, and one homogeneous setting with no data augmentations, where the only difference between the models' trainings is the dataset shuffling. Details are presented in the Appendix. The models are trained with SGD with momentum, a weight decay of 10^{-4}, and a cosine annealing scheduler with initial and minimum learning rates of 0.1 and 10^{-4}. For CIFAR-10/100, we train over 300 epochs with a batch size of 64, and 90 epochs with a batch size of 256 for ImageNet. For WASH and WASH-Opt we initialize the models with the same parameters and choose p with cross-validation to be equal to 0.001 when training on CIFAR-10/100 or 0.05 for ImageNet. We do not require any alignment technique such as REPAIR [17]. For simplicity, we do not permute or recompute the running statistics of the BatchNorm layers. The Baseline models start with the same initialization compared to PAPA, but we found this change to have no major impact. PAPA's models start on different initialization.

For our experiments, we needed a single A100 GPU for up to 14 hours to train up to a population of 10 models, and up to 40 hours for a population of 20 models. Similarly, we needed 16 A100 GPUs to train a population of 5 models on ImageNet in parallel.

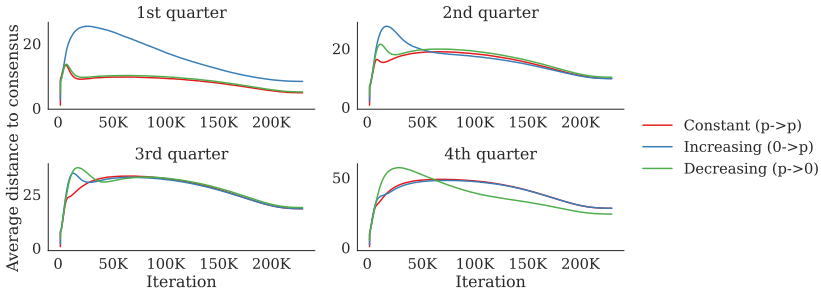

Figure 3: **Average distance to the consensus for different layer-wise adaptations of WASH**, for different slices of the model parameters. Keeping the probability constant across layers ensures the lowest distance to consensus for the first quarters. Surprisingly, in the last quarter of parameters, the 'decreasing probability' adaptation, despite starting with a higher distance to consensus, shows a lower distance to consensus later in training; even though shuffling is less frequent than in the other schedules. The 'increasing probability' adaptation shows how early layers are useful for shuffling.

B.2. Augmentations and regularization used

For a fair comparison, we follow the same data augmentations and regularisations used in [16]. We use Mixup (random draw from {0, 0.5, 1.0} for CIFAR-10/100 or from {0, 0.2} for ImageNet), Label Smoothing (random draw from {0, 0.05, 0.1} for CIFAR-10/100 or from {0, 0. 1} for ImageNet), CutMix (randomly drawn from {0, 0.5, 1.0} for CIFAR-10/100 or from {0, 1.0} for ImageNet), and Random Erasing (randomly drawn from {0, 0.15, 0.35} for CIFAR-10/100 or from {0, 0.35} for ImageNet).

Appendix C. Ablations

In this section, we present ablations to better understand the effect of the parameter shuffling, varying the layer-wise probability adaptation, the base probability value, and the shuffling period. In all cases, we consider 5 ResNet-18 models trained on CIFAR-100 in a heterogeneous environment.

Layer-wise adaptation variations. For WASH, we found that decreasing probability with depth gave the best results. We showcase in Tab. 3 the performances for alternatives where the probability either remains constant or increases with depth. We find lower performances for both alternatives. In Fig. 3 we show the distances of the models to the consensus for all three schedules. More specifically, we report the distances for different slices of the models' parameters, showing the effect of shuffling as a function of depth. As predicted, shuffling all layers equally results in the lowest distance to the consensus, except for the last quarter of parameters. Here, surprisingly, our base 'decreasing' adaptation shows a lower distance to the consensus despite less frequent shuffling. We also observe a particularly strong effect of the shuffling for the early layers, as the distance in the first quarter is more pronounced between the 'increasing' curve and the others.

Base probability variation. We present in Fig. 4 the Ensemble and Averaged for different values of p, the base shuffling probability of the first layer. Rather than a smooth increase in the accuracy of the Averaged model, we observe a phase transition between a phase where the accuracy of the Averaged model is not improved by the shuffling and a sudden increase in the accuracy where it

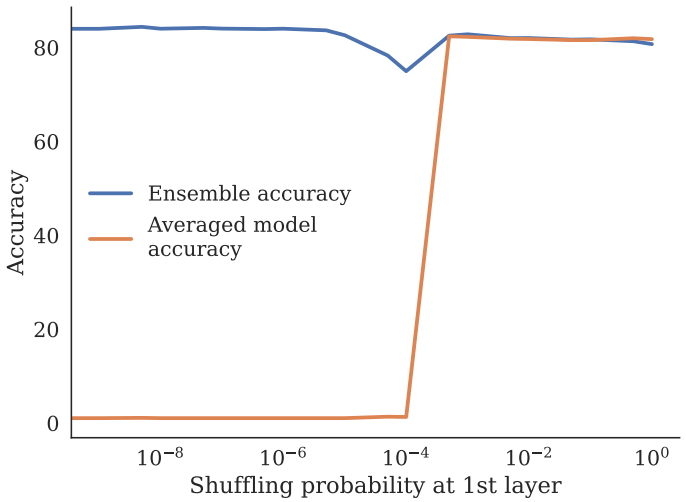

Figure 4: **Ensemble and Averaged accuracy for varying base probability values.** We observe a phase transition as the base probability increases between a phase where permuting does not improve the averaged model accuracy and a phase where the ensemble accuracy is equal to the averaged model accuracy. Between the phases, the ensemble accuracy decreases.

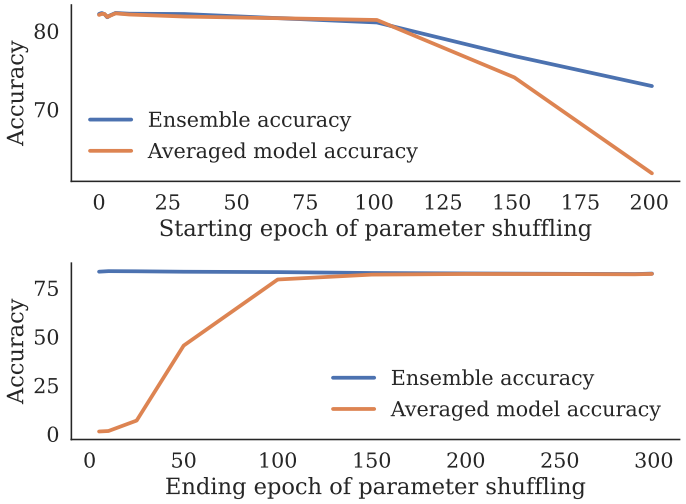

Figure 5: **Ensemble and Averaged accuracy depending on the starting or ending epoch of the shuffling.** The parameter shuffling is beneficial both at the beginning and at the end of training. Note that ending early, at epoch 150 out of 300, has less impact on performance than starting permuting at epoch 150, showing that WASH is more important early in training.

reaches the accuracy of the Ensemble. Just before the transition, the accuracy of the Ensemble decreases, before increasing again back to its previous performance. The accuracy decreases only slightly even when the shuffling probability is increased to 1, indicating the resilience of the models to heavy shuffling.

Table 3: **Test accuracies of WASH with variants of the shuffling probability per depth.** Trained with a population of 5 models on CIFAR-100 using a ResNet-18. The results show that permuting the first layers is more important than permuting the later layers. However, keeping the probability constant across layers does not significantly reduce the performance of WASH.

Proba. at layer 0	to	L-1	Technique Ensemble	Averaged	GreedySoup	Best model	Worst model
10^{-3}	$\searrow$	**0**	82.22±.38	82.15±.22	81.94± 0.25	80.89±.03	78.80±.77
10^{-3}	$\rightarrow$	10^{-3}	82.04±.19	81.94±.15	81.69±.23	80.60±.16	78.67±.89
0	$\nearrow$	10^{-3}	81.75±.35	81.37±.10	81.14±.20	80.08±.40	78.55±.70

Shuffling is beneficial at every step. Finally, we propose to show the impact of the parameter shuffling at different steps of the training by varying the epoch at which the shuffling either starts or stops. In Fig. 5, we show that there is no improvement by having a warmup or slowdown period in parameter shuffling, indicating that all phases of the training are improved by WASH. Furthermore, stopping parameter shuffling early results in a much smaller loss of Averaged accuracy compared to starting shuffling late. In other words, shuffling at the beginning of training before the models start to converge is more impactful as the models may still reside in different loss basins.

C.1. Interpolation heatmap

Here, we propose to display a heatmap showing the accuracy of more different interpolations between 5 models trained separately, with WASH, or WASH+Opt. We observe how models trained with WASH and WASH+Opt converge to the same loss basin, and that a large number of possible interpolations lead to high accuracy. The heatmaps are presented in Fig. 6. The performance of each individual model is represented at the five extremities of the heatmaps (see a. notably). Then, each other performance represented in the heatmap circle is for a model with its parameters interpolated between the 5 models. The interpolation weights are computed by normalizing the distance (from a Gaussian kernel) between the point in the circle and the 5 points at the extremities. The center of the heatmap represents an equally weighted average of the models, as implemented in WASH and the other methods considered.

Appendix D. Additional metrics

Disagreement in function space. To support our use of the distance to consensus as an accurate metric of diversity in our paper, we also report a more established metric, the model prediction disagreement, as proposed by [8]. This value corresponds to the fraction of examples in the validation set where two models disagree on the prediction. In Fig. 9, we report the disagreement for models trained on the four methods considered in this work: the Baseline without communication, PAPA, WASH, and WASH+Opt. We observe the same ranking in the methods as in the distance to consensus: the Baseline models have the highest disagreement, followed by our methods, and PAPA has the lowest. This confirms that WASH produces more diverse models than PAPA. Note that the Baseline has the highest disagreement, but the models cannot be successfully averaged.

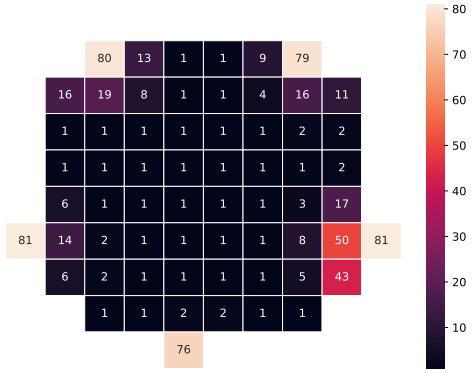

Figure 6: **Accuracy heatmap of the Baseline.** The interpolated models' performance is equal to random ones.

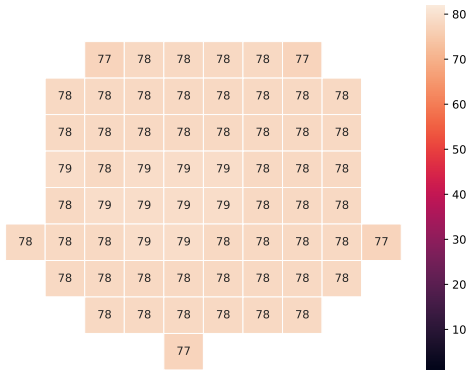

Figure 7: **Accuracy heatmap of WASH.** The accuracy is very similar for various interpolations.

Expected Calibration Error. In Tab. 4, we report the ECE for all four methods, showing that WASH provides better calibrated models than PAPA.

Method	**Indv.**	**Ens.**	**Avg.**
Baseline	0.377	0.368	0.180
WASH	0.374	0.372	0.376
WASH+Opt	0.374	0.373	0.375
PAPA	0.376	0.376	0.378

Table 4: **Expected Calibration Error (ECE) for all four methods**, for 5 ResNets trained on CIFAR-100 on heterogeneous data. We report the ECE for the individual models (Indv., averaged for the 5 models), the Ensemble model (Ens.) and the Averaged (Avg.) one. The ECE is the one obtained for the optimal temperature. Our method has a lower ECE than WASH in all cases, showing that it is better calibrated. The very low ECE for the Averaged baseline is due to the fact that the model is close to random.

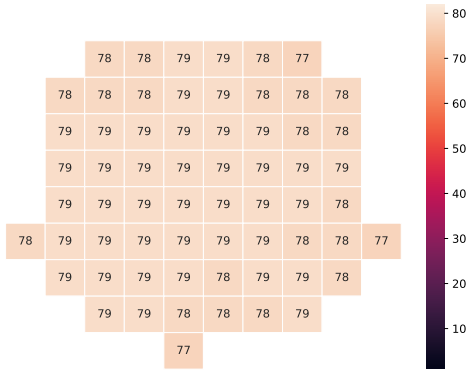

Figure 8: **Accuracy heatmap of WASH+Opt.** The results are similar to WASH.

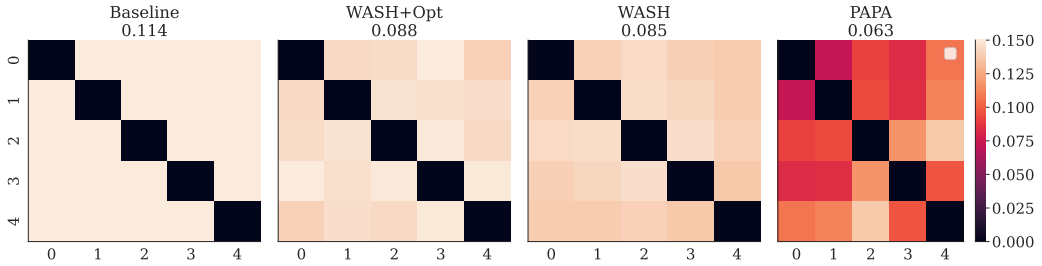

Figure 9: **Disagreement in function space**, for 5 ResNets trained on CIFAR-100 on heterogeneous data. The mean disagreement value for models with different indices is reported on top of the heatmaps. WASH has a higher disagreement between the model predictions (and thus better diversity) than PAPA.

Appendix E. Additional results

E.1. Homogeneous framework results

The results for an homogeneous framework (where data augmentations are not different between models) are presented in Tab. 5.

E.2. 2D optimization example

The loss function we consider is a highly simplified version of the Ackley function. With a minimum in (x_m, y_m) defined by

$$g(x, y, x_m, y_m, \lambda) = \exp\left(-\lambda\sqrt{0.5((x - x_m)^2 + (y - y_m)^2)}\right), \tag{7}$$

the function we consider in our example is

$$f(x, y) = -10g(x, y, 10, 10, 0.1) - 5g(x, y, 8, 3, 0.3) - 5g(x, y, 3, 8, 0.3). \tag{8}$$

This function has a 2 local minima at $(3, 8)$ and $(8, 3)$ and a global minimum at $(10, 10)$. In all three cases, the starting points are $(0, 5)$ and $(5, 0)$. We compute SGD by first computing the exact gradient of the function and then adding Gaussian noise to the gradient. The learning rate is 0.1 and

we optimize for 1000 steps. For PAPA, we consider $\alpha = 0.99$. For WASH, the shuffling probability is the same for both coordinates and is equal to 0.01.

We represent the trajectories of two points trained with SGD either separately or jointly with PAPA or WASH, on a loss function with 2 local and 1 global minima, in Fig. 10. Trained separately, they converge to a separate minimum. With PAPA, they reach a consensus but also reach a local minimum. In contrast, WASH allows them to reach the global minimum due to the greater diversity of paths. More details are in the Appendix.

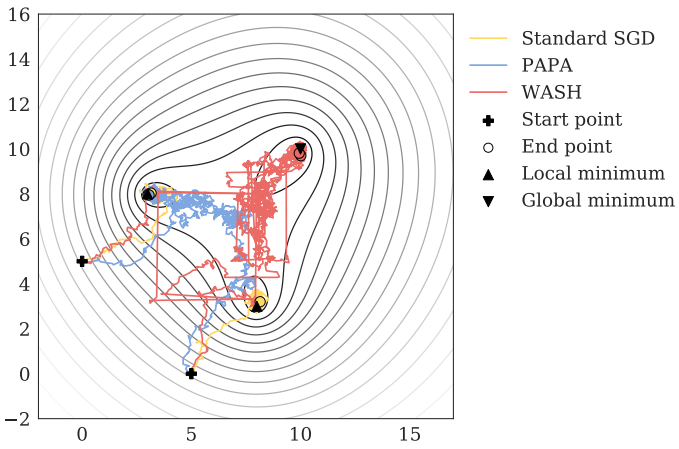

Figure 10: **2D optimization example.** We train 2 points on a function from two starting points. They converge to local minima separately when trained apart (yellow) or together when trained with PAPA (blue); while they reach the global minimum with WASH (red) due to the shuffling (orthogonal lines).

E.3. ImageNet32x32.

In Tab. 6, we report the accuracy for the dataset ImageNet32x32, showing that a lower PAPA EMA frequency compared to what was reported in their article and code ($T = 10$), results in a better Averaged performance, reproducing their results but still resulting in worse results than WASH. We also find similar results by decreasing the value of the EMA α. This confirms that the low performance of our replication of PAPA on ImageNet mainly stems from its hyperparameters, and reinforces our conclusion on the improvements provided by WASH.

E.4. GreedySoup.

In Tab. 7, we report the accuracy of GreedySoup for WASH, WASH+Opt, and PAPA, showing that it provides worse accuracy than the Averaged model, in accordance with the findings of PAPA.

E.5. REPAIR.

In Tab. 8, we show that the addition of REPAIR further reduces the gap between WASH and the Baseline ensemble accuracy, demonstrating that further post-training techniques (like self-distillation or SWA) could further improve our method.

Table 5: **Ensemble and Averaged Model accuracy for a homogeneous population of models.** We compare models trained separately (Baseline), with PAPA, or with our methods WASH and WASH+Opt. The best Ensemble (black) and Averaged (blue) accuracy are reported in bold. We observe the same results in this setting, with WASH in particular coming close to the Ensemble performance.

Method		**Baseline (trained separately)**			**PAPA**		**WASH (ours)**		**WASH+Opt (ours)**	
Config	**#N**	Ensemble	Averaged	GreedySoup	Ensemble	Averaged	Ensemble	Averaged	Ensemble	Averaged
CIFAR-10										
VGG-16	3	**94.93±.06**	10.00±.00	93.60±.41	94.38±.14	94.34±.18	94.41±.23	**94.58±.17**	94.45±.05	94.47±.02
	5	**95.29±.05**	10.00±.00	93.82±.30	94.55±.12	94.58±.12	94.72±.08	**94.70±.17**	94.63±.11	**94.68±.14**
	10	**95.23±.06**	10.00±.00	93.82±.06	94.79±.18	**94.78±.20**	94.66±.03	94.54±.07	94.71±.07	94.61±.13
ResNet18	3	**96.14±.10**	10.00±.00	95.42±.27	95.89±.04	**95.89±.06**	95.77±.12	95.77±.17	95.85±.04	**95.87±.10**
	5	**96.19±.16**	10.00±.00	95.31±.09	95.99±.08	**95.99±.08**	95.96±.08	**95.98±.05**	95.94±.12	**95.98±.12**
	10	**96.34±.02**	10.00±.00	95.26±.11	96.10±.25	**96.11±.24**	96.08±.07	**96.12±.09**	96.07±.07	96.08±.14
CIFAR-100										
VGG-16	3	**77.63±.24**	1.00±.00	73.76±.35	75.10±.11	75.09±.16	76.30±.37	**76.04±.58**	76.04±.03	75.96±.18
	5	**78.52±.10**	1.00±.00	73.76±.18	75.56±.16	75.55±.14	76.63±.27	**76.48±.23**	76.64±.15	76.13±.18
	10	**79.26±.06**	1.00±.00	73.99±.26	76.24±.44	76.26±.43	77.06±.12	**76.43±.18**	76.72±.15	75.94±.26
ResNet18	3	**79.54±.17**	1.00±.00	76.84±.54	77.83±.26	77.86±.30	78.90±.17	**78.76±.25**	78.66±.08	78.56±.21
	5	**80.11±.23**	1.00±.00	76.83±.45	77.94±.16	77.92±.19	79.24±.32	79.09±.43	79.32±.19	**79.19±.15**
	10	**80.55±.13**	1.00±.00	76.80±.41	78.40±.15	78.44±.22	79.65±.17	**79.43±.16**	79.34±.34	79.19±.45

Method	**Baseline**	**WASH**	**WASH+Opt**	**PAPA** ($T = 10$)	$T = 9$	$T = 5$
Ensemble	**74.95**±0.95	67.55±0.22	67.95±0.66	61.01±0.31	61.34±0.19	61.52±0.45
Averaged	0.1±0.0	67.80±0.16	**68.22±0.71**	1.98±1.54	35.43±13.67	61.05±0.32

Table 6: **Performance on ImageNet32**, for all methods on 3 ResNet-50 trained on heterogeneous data. $p = 0.05$ like on ImageNet. We find similar results for PAPA. However, reducing the EMA frequency T allows for a better Averaged accuracy, while still being heavily under WASH's performance.

N	**Method**	**WASH**	**WASH+Opt**	**PAPA**
3	**Averaged**	**81.90±0.19**	**82.08±0.09**	**81.53±0.13**
	GreedySoup	81.73±0.27	81.42±0.55	80.91±0.74
5	**Averaged**	**81.97±0.28**	**82.17±0.15**	**82.01±0.34**
	GreedySoup	81.83±0.26	81.49±0.91	81.67±1.03
10	**Averaged**	**82.31±0.38**	**82.18±0.22**	**82.15±0.14**
	GreedySoup	81.92±0.53	81.99±0.17	81.92±0.22

Table 7: **GreedySoup performances** for WASH and its variant and PAPA, for Resnets-18 trained on CIFAR-100 in the heterogeneous case. GreedySoup is the same method as Diwa. In the case here where averaging all models provides the best results, GreedySoup may only keep a subpar subset of weights to average (generally only one).

Method	**Ens.**	**Avg.**	**+REPAIR**
Baseline	**83.8**	0.01	0.01
WASH	82.7	82.5	82.7
WASH+Opt	82.4	82.5	**82.8**
PAPA	81.8	81.8	82.3

Table 8: **Effect of REPAIR on the four methods**, for 5 ResNets trained on CIFAR-100 on heterogeneous data. We note that REPAIR has no effect on the Baseline models. Our method's performance can be improved even closer to the baseline Ensemble by using post-training methods like REPAIR.