# Exploring the Impact of Batch Size on Model Fusion: Generalization, Curvature, and Optimization Dynamics

Alessandro Cabodi   Filippo Rambelli   Nicola Stella   Matas Udris

## Abstract

Combining neural networks together is a standard approach in machine learning to achieve better prediction robustness. A cost-effective approach is Optimal Transport (OT) Model Fusion, which (soft) aligns models' neurons before averaging their weights. Yet, it is still an open question whether the gain in performance imparted by fusion can be attributed to properties of the resulting loss landscape, whose local curvature can be highly predictive of the model's generalization power. This paper investigates the impact of batch size on the generalization capabilities of OT-fused models. Our study presents numerical evidence that OT-fused models retain the curvature properties of their parents, even after finetuning. In particular, we show that training parent models with smaller batch sizes, which lead to flatter minima, result in better performance post-fusion and lower curvature. Furthermore, we demonstrate that fusing together parents trained with different batch sizes further reduces curvature and improves accuracy after finetuning. We corroborate our studies with an investigation on linear mode connectivity (LMC), and show that OT alignment effectively reduces the energy barrier between solutions.

## 1. Introduction

Combining models in machine learning is common, typically through ensembles that average predictions [7]. This reduces the variance introduced by the numerous noise sources present during training, enhancing the results' robustness. However, resource constraints limit ensemble feasibility due to memory and computation costs that grow linearly with the number of base learners [35].

Model fusion offers an efficient alternative by approximating ensembling with a single, intelligently chosen network [1, 9, 19, 31]. Optimal Transport (OT) Model Fusion [31] is the current state-of-the-art approach. It carries out a (soft) alignment of the neurons of the parent networks via OT [34] before averaging them. Alignment is necessary due to the lack of a unique parametrization scheme of the models' weights [24], especially when initialized differently.

OT Model Fusion poses significant computational and storage advantages over ensembling, while offering comparable performance after moderate finetuning. It also performs notably better than naive vanilla averaging of the models' weights for one-shot knowledge transfer [31].

Previous results highlight that, when using stochastic gradient descent (SGD) on deep neural networks, rich generalization is made possible by smaller batch sizes and higher learning rates [16], as they lead to smoother and flatter optima [21]. Whether such generalization properties persist after model fusion remains an open question. The concept of linear mode connectivity (LMC) helps approaching this problem due to its link to high in-domain generalization performance [9, 11, 14].

In this paper, we examine the influence of the batch sizes used to train the parent networks on the generalization capabilities of the fused models. We show that training the parent models with smaller batch sizes, which tend to converge to flatter minima, induces a lower LMC barrier between the respective solutions in parameter space post OT alignment [28]. This, in turn, implies lower performance degradation after fusion.

We investigate the local curvature characteristics of the loss landscape of the finetuned fused model as an indicator of its generalization capabilities [20]. We demonstrate that model fusion preserves the curvature properties of the parent models, as determined by the batch sizes they were trained on. This means that networks which converged to flatter minima, once fused and finetuned, also converge to flatter minima compared to the same networks trained on higher batch sizes. Moreover, we show that when models trained on different batch sizes are fused, the resulting network predominantly maintains the curvature properties of the parent model trained on the smaller batch size. Additionally, we present partial results suggesting that a greater disparity between the parent networks in this scenario may enhance the generalization capabilities of the fused model.

These findings emphasize the advantages of training at least one model in the fusion process with a smaller batch size. They also show promise for combining models trained with distinct batch sizes. Notably, if these trends continue when fusing more networks, it could enable better parallelization [21] by allowing to use larger batches for most models, with only one network requiring training on a small batch.

## 2. Related Work

**OT Model Fusion** [31] utilizes optimal transport to minimize the transportation cost of neurons in individual model layers, based on the similarity of activations or incoming weights. This approach effectively computes the Wasserstein barycenter of the probability measures at the corresponding layers of the parent models, through a layer-wise averaging process. Notably, OT Model Fusion excels in one-shot merging of networks with differing weights, surpassing standard averaging methods. Additionally, with moderate finetuning, OT fusion can serve as an efficient alternative to prediction ensembling.

**Ensembling** [13] enhances the robustness of predictions by smoothing out the decision boundaries. We explore whether such a smoothing effect is observable in networks fused via OT Model Fusion as well. This can be investigated by examining the imparted benefit on the local curvature of the network solution in the parameter space, a factor known to be highly linked with generalization performances [20]. While computing the full Hessian matrix is computationally unfeasible for large models, proxy measures such as top eigenvalue and trace can be approximated efficiently.

**PyHessian** [37] is a Python framework that enables fast computation of Hessian information for deep neural networks. It can efficiently compute the top $k$ eigenvalues (using the power method [36]) and the trace (via the Hutchinson algorithm [2, 3]), as well as the full empirical spectral density (through Stochastic Lanczos Quadrature [25, 33]), providing insights into the loss landscape's local flatness or sharpness.

**Stochastic Gradient Descent (SGD)** [32] uses batches of sampled training data for gradient approximation. While larger batches allow for better parallelization, they often reduce a model's generalization power due to convergence to sharp minima of the loss function [18]. Conversely, small-batch SGD tends to find flatter minima [21].

**Linear mode connectivity (LMC)** [9] has direct implications for ensemble [14] and weight averaging methods [29]. Research in [8, 21, 38] shows that deep neural networks' loss landscapes contain numerous minima, partly due to permutation invariance, which allows multiple parameter representations of the same function [5, 26]. Work in [8, 12, 14] indicates that minima found by SGD are connected by non-linear paths of non-increasing loss, a concept known as mode connectivity. A linear form of mode connectivity can be assessed through instability analysis, a method to determine whether a network is susceptible to SGD noise (e.g. induced by data order or augmentation) [11]. This can be quantified by the highest error increase (energy barrier) along the linear path that connects the two networks resulting from instability analysis [8].

Let $\theta$ denote a network's parameters and $\mathcal{L}(\theta)$ its average loss over the training data. The loss barrier $B(\theta_1, \theta_2)$ along the linear path between $\theta_1$ and $\theta_2$ is defined as:

$$B(\theta_1, \theta_2) = \sup_{\alpha \in [0,1]} \Big[\mathcal{L}(\alpha\theta_1 + (1-\alpha)\theta_2) - (\alpha\mathcal{L}(\theta_1) + (1-\alpha)\mathcal{L}(\theta_2))\Big]$$

This perspective suggests that SGD solutions that are linearly connected (i.e. with negligible barrier), belong to the same loss landscape basin [27]. On the other hand, the conjecture in [9] states that most SGD solutions belong to a set $\mathcal{S}$ whose elements can be permuted in such a way that there is no barrier on the linear interpolation between any two permuted elements in $\mathcal{S}$. This conjecture has significant implications for model ensembling, as it allows for the averaging of models within the same loss landscape basin. In this paper, we explore whether two models indeed belong to the same basin after alignment via OT.

## 3. Methodology

We conducted ablation studies to explore the impact of batch sizes in the context of model fusion. Our exploration was focused on aspects independent of architecture or dataset specifics. The experiments were thus conducted in minimal setups. To stay within time and resource constraints, we trained ResNet-18 [17] and VGG-11 [30] on widely used simple datasets such as MNIST [6] and CIFAR-10 [22].

We set up the experiments by training two identical-architecture parent networks $A$ and $B$ on the same task and dataset, using SGD with momentum as the optimizer. The learning rate was adjusted according to the batch size, in order to achieve comparable performances for all networks [16]. Batch sizes of 32, 128, and 512 were used. More details on hyperparameters can be found in A.2.

We first carried out a series of experiments on LMC, computing the barrier pre and post OT alignment. Instability analysis, based on data order and augmentation [11], was conducted on pairs of models having same and different initialization. Subsequently, an examination of the influence of batch sizes on the LMC barrier was undertaken, along with an assessment of the linear connectivity between the solutions for $A$ and $B$.

We conducted more experiments performing OT Model Fusion by aligning $A$ to $B$ via OT, and compared one-shot knowledge transfer results to those obtained by vanilla averaging. Parent models with corresponding and mixed batch sizes were used, both in the same and different initialization settings. We tested only activation alignment due to GPU memory constraints. Based on model size, we used 200 activation samples for ResNet-18, and 75 samples for VGG-11.

In the case of different initialization, the OT fused models have been finetuned and evaluated against the parent networks as well as our baselines: vanilla averaging (post-finetuning) and prediction ensembling. Regarding finetuning, we generally adhered to the initial hyperparameter configurations used in training the parent models. Any adjustment was made sparingly, only when necessary to reliably exceed the test accuracy achieved by the parent models for both vanilla averaging and OT fusion. Notably, in the mixed batch sizes case, we tested finetuning using both the batch size of the aligned-to parent and an intermediate batch size. More details are provided in A.3.

We proceeded to analyze the local curvature of the fusion solutions pre and post finetuning on the training loss. The max eigenvalue and the trace of the Hessian were calculated to evaluate both the largest and overall steepness [37]. For sake of comparison, the same measurements were repeated on all parent models. In all cases, we compared the best checkpoint in terms of validation accuracy for each model. This is motivated by the validation accuracy being a proxy for generalization capabilities, and by the fact that networks with best validation accuracy would likely be the checkpoints used in practice. In addition, for the fusion solution prior to finetuning we plotted the full Hessian eigenvalue spectral density [15, 37], in order to investigate the presence of descent directions, and compared the results with those for vanilla averaging (see B.3).

PyTorch Lightning [10] was used for the training and evaluation pipeline, which was monitored with Weights & Biases [4], while the PyHessian package [37] was used to study the solutions' curvature. Finally, we employed an external library, namely the one provided by [23], for plotting the loss landscapes (see B.4).

## 4. Results

The instability analysis, as summarized in Table 1, reveals that the loss barrier, both pre and post-alignment, tends to increase with larger batch sizes for both same and different initialization scenarios. When the models share the same initialization, the loss barrier for batch sizes 32 and 128 is observed to be smaller compared to cases where the models are initialized differently, both before and after alignment. Notably, this pattern does not hold for batch size 512.

However, in all cases the results do not allow to conclude that the minima are linearly connected, even though the barrier becomes significantly lower after OT alignment, reaching results that appear to be in line with the permutation selection method in [1] for $1\times$ width ResNet. This in practice translates in significantly lower performance degradation after OT fusion with respect to vanilla averaging, as also shown in Tables 2, 3, 4. In this regard, consistently with findings in [11], it is noted that even with the same initialization, the pre-alignment barrier remains high, underscoring the utility of OT alignment in such scenarios as well.

| Batch | Same init. | | Different init. | |
|---|---|---|---|---|
| size | pre | post | pre | post |
| 32 | 1.3797 | 0.4601 | 2.2496 | 0.5473 |
| 128 | 2.1547 | 0.538 | 2.2544 | 0.6213 |
| 512 | 2.1521 | 0.7731 | 2.2638 | 0.6969 |

*Table 1.* LMC barrier pre / post alignment (ResNet-18, CIFAR-10).

The hypothesis that models trained with a lower batch size converge to flatter minima is supported by both results of the experiments conducted in same and different initialization scenarios (Tables 2, 3). The same effects are observed in the fused models, which therefore appear to retain the properties of the parent models.

The same initialization appears to be beneficial for fusion for batch size 32, both in terms of accuracy and flatness. However, this is not the case for batch size 512, where different initialization achieves better results. Such findings are in line with the pattern manifested for the LMC barrier.

Despite the reduced curvature of vanilla averaging models, the poor test accuracy they achieve (with the exception of batch 32 case with same initialization) suggests that these solutions are not optimal. The plots shown in B.3 indeed support this hypothesis.

In the different initialization case, finetuned models exhibit a retention of the batch-related properties of the parent models. The lower curvature of the finetuned models compared to the parents, coupled with higher generalization performance (as approximated by test accuracy), once again demonstrates the validity of this relationship. In addition, we emphasize that the finetuned OT fused model are consistently superior to finetuned vanilla averaging and demonstrate accuracy comparable to, and at times surpassing, that of prediction ensembling, rendering it a cost-effective alternative.

To better validate the results, the same experiments were performed using ResNet-18 on MNIST and VGG-11 model on CIFAR-10 (B.2). However, in the latter case, replicating similar results proved challenging. Such a discrepancy may stem from several factors, including a relatively lower accuracy levels achieved by the parent models (probably due to a sub-optimal choice of hyperparameters) and a significantly greater memory inefficiency that conflicted with our resource constraints and rendered some approximations necessary, potentially leading to greater instability in curvature analysis.

Finally, for ResNet-18 on CIFAR-10 we tried to fuse together networks trained with different batch sizes. The results, reported in 4), highlight how both post-fusion

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 90.65 | 2.48 | 126.47 | 512 | Model A | 90.38 | 8.03 | 231.17 |
| 32 | Model B | 91.24 | 2.34 | 107.57 | 512 | Model B | 89.84 | 8.31 | 310.12 |
| | Vanilla avg | 55.75 | 3.51 | 66.47 | | Vanilla avg | 19.38 | 7.57 | 42.58 |
| | A aligned to B | 78.55 | 7.79 | 425.16 | | A aligned to B | 70.70 | 26.42 | 743.58 |
| | Pred. avg | 92.10 | – | – | | Pred. avg | 91.39 | – | – |
| | OTFusion | 78.38 | 3.41 | 141.06 | | OTFusion | 63.87 | 10.74 | 245.23 |

*Table 2.* Same initialization for A and B, trained with corresponding batch size (ResNet-18, CIFAR-10). A is aligned to B.

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 90.65 | 2.48 | 126.47 | 512 | Model A | 90.38 | 8.03 | 231.17 |
| 32 | Model B | 91.90 | 2.49 | 104.19 | 512 | Model B | 90.2 | 6.21 | 194.00 |
| | Vanilla avg | 17.9 | -0.19 | 1.06 | | Vanilla avg | 12.49 | 1.04 | 4.53 |
| | A aligned to B | 68.96 | 9.06 | 427.71 | | A aligned to B | 73.47 | 27.33 | 719.33 |
| | Pred. avg | 92.50 | – | – | | Pred. avg | 91.46 | – | – |
| | OTFusion | 72.46 | 3.55 | 126.34 | | OTFusion | 70.22 | 11.04 | 235.02 |
| 32 | Vanilla avg FT | 92.00 | 1.65 | 54.09 | 512 | Vanilla avg FT | 90.98 | 21.12 | 279.48 |
| 32 | OTFusion FT | 92.30 | 3.03 | 43.04 | 512 | OTFusion FT | 91.66 | 6.73 | 96.26 |

*Table 3.* Different initialization for A and B, trained with corresponding batch size (ResNet-18, CIFAR-10). A is aligned to B.

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 91.90 | 2.49 | 104.19 | 512 | Model A | 90.38 | 8.03 | 231.17 |
| 512 | Model B | 90.38 | 8.03 | 231.17 | 32 | Model B | 91.90 | 2.49 | 104.19 |
| | Vanilla avg | 26.75 | -0.37 | 1.33 | | Vanilla avg | 26.75 | -0.37 | 1.33 |
| | A aligned to B | 50.22 | 16.13 | 406.20 | | A aligned to B | 55.52 | 37.11 | 827.00 |
| | Pred. avg | 92.63 | – | – | | Pred. avg | 92.63 | – | – |
| | OTFusion | 49.11 | 7.18 | 115.19 | | OTFusion | 67.53 | 5.39 | 117.40 |
| 256 | OTFusion FT | 92.23 | 2.77 | 37.90 | 256 | OTFusion FT | 92.47 | 4.37 | 30.74 |
| 512 | OTFusion FT | 92.11 | 3.12 | 43.94 | 32 | OTFusion FT | 92.65 | 3.60 | 33.86 |

*Table 4.* Different initialization for A and B, trained with different batch sizes (ResNet-18, CIFAR-10). A is aligned to B.

performance and flatness are somewhat better when aligning with respect to the model with a smaller batch size. In particular, this trend appears not to be affected by the batch size used during finetuning. We speculate that starting finetuning from an advanced training stage (in relation to the performances attained post-fusion) leads SGD to converge to similar solutions. These results appear to be in line with those reported in [11]. The post-finetuning accuracy in the mixed batch sizes case turns out to be the highest, indicating a benefit in fusing models trained with different batch sizes, probably due to their greater disparity, which in this case seems to boost generalization.

## 5. Discussion

The results obtained are promising and could help achieve even better performance with model fusion, especially concerning the merging of models trained with different batch sizes. However, to better assess the robustness of these results, it would be advisable to evaluate their scalability on more complex models or datasets (e.g. CIFAR-100). Furthermore, we hypothesize that differences in curvature and performance could be even more pronounced when using models trained with larger batch sizes, e.g. 4096. We consider these to be promising future research directions.

## 6. Summary

We analyzed local curvature characteristics of the loss landscape of deep neural network models before and after OT Model Fusion with an emphasis on the batch size used to train the parent models. We observed that the curvature characteristics persist not only post-fusion, but also after finetuning. Additionally, we found that fusing models trained with different batch sizes retains the flatness of the model with the lower batch size and results in superior test accuracy. Lastly, we assessed the LMC of the parent solutions and provided evidence that OT alignment significantly lowers the energy barrier between them.

## References

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

[2] Avron, H. and Toledo, S. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *J. ACM*, 58(2), apr 2011. ISSN 0004-5411. doi: 10.1145/1944345.1944349. URL https://doi.org/10.1145/1944345.1944349.

[3] Bai, Z., Fahey, G., and Golub, G. Some large-scale matrix computation problems. *Journal of Computational and Applied Mathematics*, 74(1-2):71–89, 1996.

[4] Biewald, L. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

[5] Brea, J., Simsek, B., Illing, B., and Gerstner, W. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape. *arXiv preprint arXiv:1907.02911*, 2019.

[6] Deng, L. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

[7] Dietterich, T. G. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pp. 1–15, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. ISBN 978-3-540-45014-6.

[8] Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. Essentially no barriers in neural network energy landscape. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1309–1318. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/draxler18a.html.

[9] Entezari, R., Sedghi, H., Saukh, O., and Neyshabur, B. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.

[10] Falcon, W. and The PyTorch Lightning team. PyTorch Lightning, March 2019. URL https://github.com/Lightning-AI/lightning.

[11] Frankle, J., Dziugaite, G. K., Roy, D., and Carbin, M. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020.

[12] Freeman, C. D. and Bruna, J. Topology and geometry of half-rectified network optimization. *arXiv preprint arXiv:1611.01540*, 2016.

[13] Ganaie, M. A., Hu, M., Malik, A., Tanveer, M., and Suganthan, P. Ensemble deep learning: A review. *Engineering Applications of Artificial Intelligence*, 115:105151, 2022.

[14] Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. Loss surfaces, mode connectivity, and fast ensembling of dnns. *Advances in neural information processing systems*, 31, 2018.

[15] Ghorbani, B., Krishnan, S., and Xiao, Y. An investigation into neural net optimization via hessian eigenvalue density, 2019.

[16] He, F., Liu, T., and Tao, D. Control batch size and learning rate to generalize well: Theoretical and empirical evidence. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/dc6a70712a252123c40d2adba6a11d84-Paper.pdf.

[17] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015.

[18] Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural computation*, 9(1):1–42, 1997.

[19] Imfeld, M., Graldi, J., Giordano, M., Hofmann, T., Anagnostidis, S., and Singh, S. P. Transformer fusion with optimal transport, 2023.

[20] Jiang, Y., Neyshabur, B., Mobahi, H., Krishnan, D., and Bengio, S. Fantastic generalization measures and where to find them, 2019.

[21] Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

[22] Krizhevsky, A. Learning multiple layers of features from tiny images, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[23] Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets, 2018.

[24] Li, W., Peng, Y., Zhang, M., Ding, L., Hu, H., and Shen, L. Deep model fusion: A survey. *arXiv preprint arXiv:2309.15698*, 2023.

[25] Lin, L., Saad, Y., and Yang, C. Approximating spectral densities of large matrices. *SIAM review*, 58(1):34–65, 2016.

[26] Neyshabur, B., Salakhutdinov, R. R., and Srebro, N. Path-sgd: Path-normalized optimization in deep neural networks. *Advances in neural information processing systems*, 28, 2015.

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

[28] Sagun, L., Evci, U., Guney, V. U., Dauphin, Y., and Bottou, L. Empirical analysis of the hessian of over-parametrized neural networks, 2018.

[29] Scaman, K., Bach, F., Bubeck, S., Lee, Y. T., and Massoulié, L. Optimal convergence rates for convex distributed optimization in networks. *Journal of Machine Learning Research*, 20(159):1–31, 2019. URL http://jmlr.org/papers/v20/19-543.html.

[30] Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition, 2015.

[31] Singh, S. P. and Jaggi, M. Model fusion via optimal transport, 2023.

[32] Smith, S., Elsen, E., and De, S. On the generalization benefit of noise in stochastic gradient descent. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9058–9067. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/smith20a.html.

[33] Ubaru, S., Chen, J., and Saad, Y. Fast estimation of tr(f(a)) via stochastic lanczos quadrature. *SIAM Journal on Matrix Analysis and Applications*, 38(4): 1075–1099, 2017.

[34] Villani, C. et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

[35] Wen, Y., Tran, D., and Ba, J. Batchensemble: An alternative approach to efficient ensemble and lifelong learning. *CoRR*, abs/2002.06715, 2020. URL https://arxiv.org/abs/2002.06715.

[36] Yao, Z., Gholami, A., Lei, Q., Keutzer, K., and Mahoney, M. W. Hessian-based analysis of large batch training and robustness to adversaries. *Advances in Neural Information Processing Systems*, 31, 2018.

[37] Yao, Z., Gholami, A., Keutzer, K., and Mahoney, M. Pyhessian: Neural networks through the lens of the hessian, 2020.

[38] Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

## Appendix

## A. Technical Specifications

### A.1. Code Repository

The code and implementations of our experiments can be found at `https://github.com/nickstella/model-fusion-curvature-analysis`.

### A.2. Parent Training

The training of the parent models was done using Cross-Entropy loss and SGD with momentum 0.9 and weight decay 0.0001. All models were trained with early stopping based on validation loss with a patience of 15 epochs. The plateau learning rate scheduler based on validation loss was used with a patience of 7 epochs and a learning rate decay factor of 0.1. CIFAR-10 and MNIST datasets were normalized according to the defaults provided by the Lightning-Bolts library [10]. For training on CIFAR-10, standard data augmentation comprised of random cropping and random horizontal flipping was applied. We bounded the epochs between 20 and 100 for MNIST and between 50 and 200 for CIFAR-10. For the unspecified parameters, default values were used. See table 5 below for more details on the hyperparameters.

| Architecture | Dataset | Batch size | Model initialization seed | Dataloader seed | Learning rate |
|---|---|---:|---:|---:|---:|
| ResNet-18 | CIFAR-10 | 32 | 42 | 42 | 0.025 |
| ResNet-18 | CIFAR-10 | 32 | 42 | 43 | 0.025 |
| ResNet-18 | CIFAR-10 | 32 | 43 | 42 | 0.025 |
| ResNet-18 | CIFAR-10 | 128 | 42 | 42 | 0.01 |
| ResNet-18 | CIFAR-10 | 128 | 42 | 43 | 0.01 |
| ResNet-18 | CIFAR-10 | 128 | 43 | 42 | 0.01 |
| ResNet-18 | CIFAR-10 | 512 | 42 | 42 | 0.04 |
| ResNet-18 | CIFAR-10 | 512 | 42 | 43 | 0.04 |
| ResNet-18 | CIFAR-10 | 512 | 43 | 42 | 0.04 |
| ResNet-18 | MNIST | 32 | 42 | 42 | 0.025 |
| ResNet-18 | MNIST | 32 | 42 | 43 | 0.025 |
| ResNet-18 | MNIST | 32 | 43 | 42 | 0.025 |
| ResNet-18 | MNIST | 512 | 42 | 42 | 0.1 |
| ResNet-18 | MNIST | 512 | 42 | 43 | 0.1 |
| ResNet-18 | MNIST | 512 | 43 | 42 | 0.1 |
| VGG-11 | CIFAR-10 | 32 | 42 | 42 | 0.025 |
| VGG-11 | CIFAR-10 | 32 | 42 | 43 | 0.025 |
| VGG-11 | CIFAR-10 | 32 | 43 | 42 | 0.025 |
| VGG-11 | CIFAR-10 | 512 | 42 | 42 | 0.01 |
| VGG-11 | CIFAR-10 | 512 | 42 | 43 | 0.01 |
| VGG-11 | CIFAR-10 | 512 | 43 | 42 | 0.01 |

*Table 5.* Hyperparameters for training the parent models.

## A.3. Finetuning

As for the finetuning of the fused models, the setup mirrors the training setup, with the different hyperparameters listed in table 6. Additionally, the learning rate decay factor for the plateau scheduler was changed to 0.5, and the patience of the early stopping callback was set to 20 epochs. Note that finetuning was only performed for instances where the parent models had different initialization.

| Architecture | Dataset | Fusion method | Batch Size | | | Learning rate | Momentum |
|---|---|---|---|---|---|---|---|
| | | | A | B | FT | | |
| ResNet-18 | CIFAR-10 | Vanilla | 32 | 32 | 32 | 0.01 | 0.95 |
| ResNet-18 | CIFAR-10 | OTF | 32 | 32 | 32 | 0.01 | 0.95 |
| ResNet-18 | CIFAR-10 | Vanilla | 128 | 128 | 128 | 0.05 | 0.9 |
| ResNet-18 | CIFAR-10 | OTF | 128 | 128 | 128 | 0.05 | 0.9 |
| ResNet-18 | CIFAR-10 | Vanilla | 512 | 512 | 512 | 0.05 | 0.9 |
| ResNet-18 | CIFAR-10 | OTF | 512 | 512 | 512 | 0.1 | 0.9 |
| ResNet-18 | CIFAR-10 | OTF | 512 | 32 | 32 | 0.01 | 0.9 |
| ResNet-18 | CIFAR-10 | OTF | 512 | 32 | 256 | 0.1 | 0.9 |
| ResNet-18 | CIFAR-10 | OTF | 32 | 512 | 512 | 0.1 | 0.9 |
| ResNet-18 | CIFAR-10 | OTF | 32 | 512 | 256 | 0.1 | 0.9 |
| ResNet-18 | MNIST | OTF | 32 | 32 | 32 | 0.005 | 0.98 |
| ResNet-18 | MNIST | OTF | 512 | 512 | 512 | 0.1 | 0.9 |
| VGG-11 | CIFAR-10 | OTF | 32 | 32 | 32 | 0.005 | 0.95 |
| VGG-11 | CIFAR-10 | OTF | 512 | 512 | 512 | 0.1 | 0.95 |

*Table 6.* Hyperparameters for finetuning fused models. A and B correspond to parent models $A$ and $B$, and fused models are so that $A$ is aligned to $B$.

## A.4. PyHessian configuration

For estimating Hessian information (max eigenvalue, trace, empirical spectral density), we used 15 batches of size 256, for a total of 3840 data-points. After some empirical investigation it proved to be a good trade-off in terms of results stability and computation time.

## B. More Experiments

### B.1. Linear Mode Connectivity

As shown in tables 7, 8, 9, we report LMC barriers before and after OT alignment for different combinations of models, parent batch sizes, and datasets. Experimental results in 7 suggest that, for parents with different batch sizes, OT-aligning the model with larger batch size onto the one with smaller batch yields smaller LMC barriers in both initialization scenarios.

Tables 8, 9 do not fully align with previous results. As for VGG, we believe the reason is that time and resource limits did not allow for a proper training of the models. Further investigation will be needed for the discrepancies in 8, but for now we deem them to be fortuitous.

| Batch size | | Same init | | Diff init | |
|---|---|---|---|---|---|
| A | B | pre | post | pre | post |
| 32 | 512 | 1.6574 | 0.85299 | 2.2562 | 0.7883 |
| 512 | 32 | 1.6574 | 0.7337 | 2.2562 | 0.7363 |

*Table 7.* LMC barrier pre and post alignment for mixed batch sizes (ResNet-18, CIFAR-10).

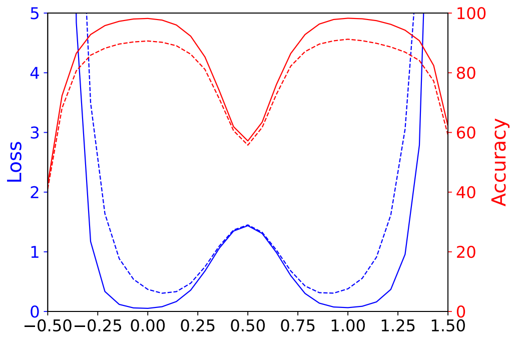
| Batch | Same init | | Diff init | |
|---|---|---|---|---|
| size | pre | post | pre | post |
| 32 | 0.9541 | 0.4597 | 2.2870 | 1.0208 |
| 512 | 2.1808 | 0.7466 | 2.2843 | 0.3133 |

*Table 8.* LMC barrier pre and post alignment (ResNet-18, MNIST).

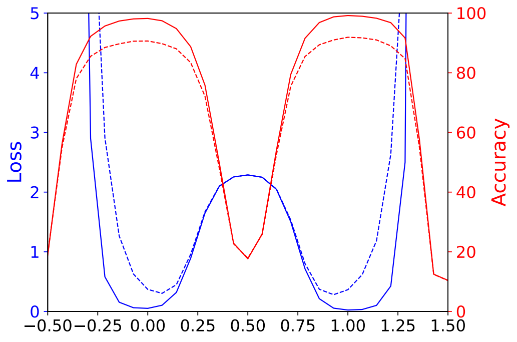
| Batch | Same init | | Diff init | |
|---|---|---|---|---|
| size | pre | post | pre | post |
| 32 | 2.1462 | 1.6953 | 2.1728 | 1.7161 |
| 512 | 2.1715 | 1.6433 | 2.1888 | 1.7099 |

*Table 9.* LMC barrier pre and post alignment (VGG-11, CIFAR-10).

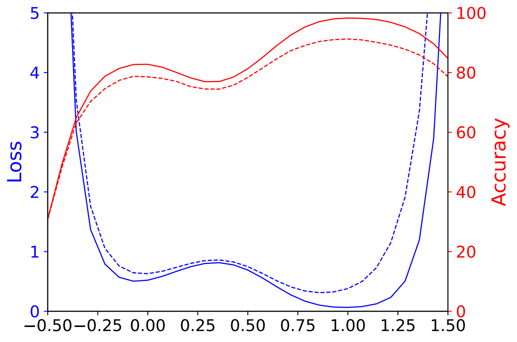
Plots in figures 1, 2 were created making use of the publicly accessible loss-landscape repository [23]. One can notice that vanilla averaging solutions are located on local maxima of the loss with respect to the interpolation direction, and are hence characterized by an almost null curvature. This is not the case for the post-fusion solutions, which clearly exhibit a non-zero eigenvalue on the interpolation direction after alignment. Moreover, it is worth mentioning that the losses on the interpolation direction are considerably lower in the same initialization case with batch size 32 compared to others. As a direct consequence, the accuracy has a significantly lower degradation. The figures highlight how, in all cases, OT alignment substantially reduces the barrier.



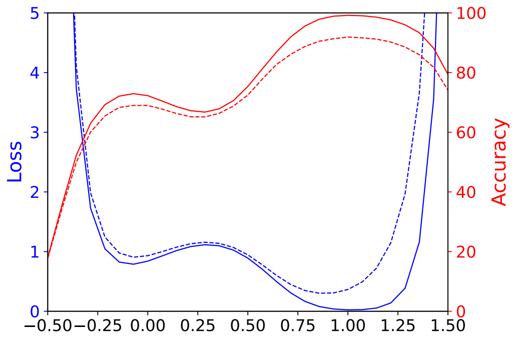
(a) Same initialization, pre alignment



(b) Different initialization, pre alignment

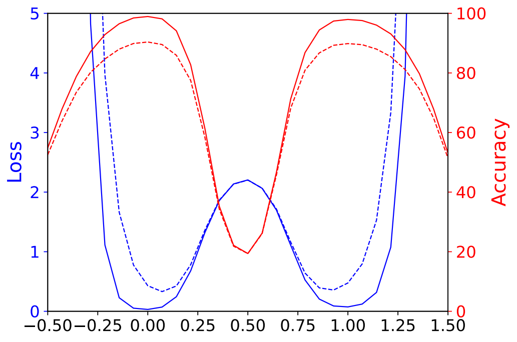


(c) Same initialization, post alignment

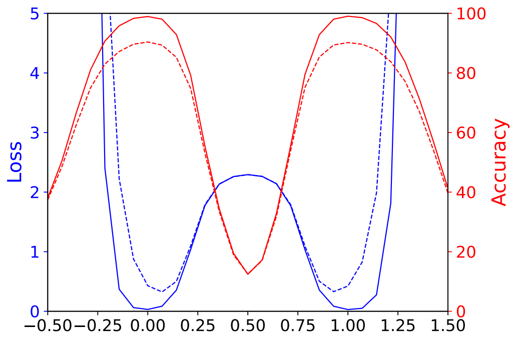


(d) Different initialization, post alignment

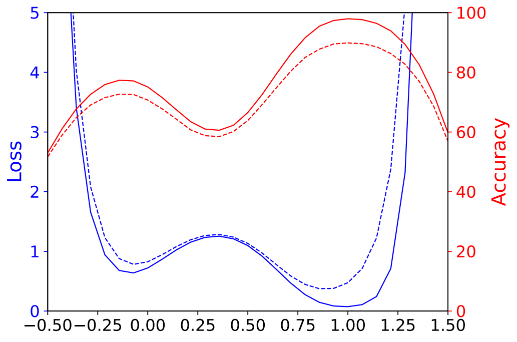
*Figure 1.* Train loss (solid blue) and train accuracy (solid red) on the linear path between A (x=0) and B (x=1) are plotted. Dashed lines are the same metrics evaluated on the test set. Vanilla averaging (x=0.5, pre alignment) and OT fusion (x=0.5, post alignment) are also displayed. The batch size used to train A and B is 32, A is aligned to B (ResNet-18, CIFAR-10).



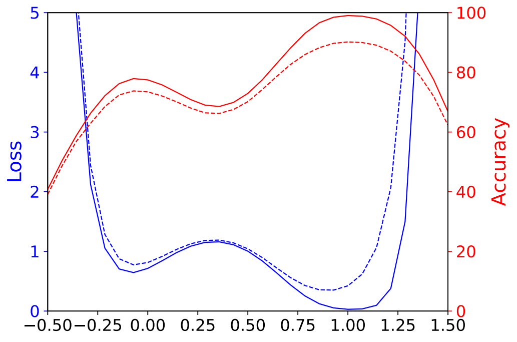
(a) Same initialization, pre alignment



(b) Different initialization, pre alignment



(c) Same initialization, post alignment



(d) Different initialization, post alignment

*Figure 2.* Train loss (solid blue) and train accuracy (solid red) on the linear path between A (x=0) and B (x=1) are plotted. Dashed lines are the same metrics evaluated on the test set. Vanilla averaging (x=0.5, pre alignment) and OT fusion (x=0.5, post alignment) are also displayed. The batch size used to train A and B is 512, A has been aligned to B (ResNet-18, CIFAR-10).

### B.2. OT Fusion and Curvature Analysis

Tables 10, 11, 12, 13, 14 show metrics on performance and flatness of the loss landscapes of our fused models (along with the finetuned models in the different initialization scenario), allowing comparison with the parent networks and with the baselines (vanilla and prediction averaging).

The trend for ResNet-18 on MNIST (Table 11, 12) seems to follow that reported for ResNet-18 on CIFAR-10, though the very high accuracy achieved makes it more difficult to discern clear differences between the small and large batch cases.

Conversely, we observed that reproducing similar results with VGG-11 on CIFAR-10 presented difficulties. This variation could be attributed to multiple factors. One possibility is the comparatively lower accuracy levels attained by the parent models, which might be a result of sub-optimal hyperparameter selection. Additionally, there was a notable increase in memory inefficiency, which conflicted with our resource limitations. This necessitated some approximations (e.g. reduced number of activation samples), which in turn could have contributed to increased instability in the curvature analysis (see A.4.

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 90.65 | 2.47 | 126.47 | 512 | Model A | 90.38 | 8.03 | 231.17 |
| 512 | Model B | 90.38 | 8.03 | 231.17 | 32 | Model B | 90.65 | 2.48 | 126.47 |
| | Vanilla avg | 49.04 | 4.22 | 62.64 | | Vanilla avg | 49.04 | 4.22 | 62.64 |
| | A aligned to B | 63.05 | 13.85 | 490.30 | | A aligned to B | 69.08 | 25.36 | 729.51 |
| | Pred. avg | 91.96 | – | – | | Pred. avg | 91.96 | – | – |
| | OTFusion | 55.67 | 6.99 | 119.45 | | OTFusion | 74.6 | 3.92 | 127.80 |

*Table 10.* Same initialization for A and B, trained with mixed batch sizes (ResNet-18, CIFAR-10).

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 99.49 | 0.123 | 1.017 | 512 | Model A | 99.14 | 0.82 | 6.36 |
| 32 | Model B | 99.42 | 0.57 | 3.98 | 512 | Model B | 99.18 | 3.73 | 26.67 |
| | Vanilla avg | 96.50 | 2.40 | 38.17 | | Vanilla avg | 11.19 | 15.52 | 64.61 |
| | A aligned to B | 98.57 | 9.19 | 72.35 | | A aligned to B | 71.78 | 211.68 | 1665.69 |
| | Pred. avg | 99.50 | – | – | | Pred. avg | 99.33 | – | – |
| | OTFusion | 98.08 | 3.75 | 82.6 | | OTFusion | 60.84 | 51.78 | 322.12 |

*Table 11.* Same initialization for A and B, trained with corresponding batch size (ResNet-18, MNIST).

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 99.49 | 0.13 | 1.02 | 512 | Model A | 99.14 | 0.82 | 6.36 |
| 32 | Model B | 99.32 | 0.07 | 0.57 | 512 | Model B | 99.05 | 0.26 | 2.14 |
| | Vanilla avg | 17.38 | 0.41 | -0.25 | | Vanilla avg | 23.23 | -1.02 | 0.46 |
| | A aligned to B | 93.74 | 6.63 | 103.89 | | A aligned to B | 68.7 | 285.84 | 1228.00 |
| | Pred. avg | 99.49 | – | – | | Pred. avg | 99.24 | – | – |
| | OTFusion | 96.96 | -2.36 | 58.78 | | OTFusion | 69.31 | 71.75 | 378.58 |
| 32 | OTFusion FT | 99.46 | 0.09 | 0.42 | 512 | OTFusion FT | 99.19 | 0.11 | 0.52 |

*Table 12.* Different initialization for A and B, trained with corresponding batch size (ResNet-18, MNIST).

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 85.13 | 13.98 | 584.52 | 512 | Model A | 86.93 | 6.37 | 367.13 |
| 32 | Model B | 85.24 | 14.5 | 659.64 | 512 | Model B | 85.88 | 8.23 | 540.82 |
| | Vanilla avg | 32.76 | -0.01 | -0.01 | | Vanilla avg | 21.57 | -0.01 | 0.02 |
| | A aligned to B | 85.1 | 13.95 | 616.11 | | A aligned to B | 86.94 | 6.37 | 370.31 |
| | Pred. avg | 87.15 | – | – | | Pred. avg | 87.97 | – | – |
| | OTFusion | 68.37 | 3.45 | 22.46 | | OTFusion | 71.63 | 3.91 | 40.59 |

*Table 13.* Same initialization for A and B, trained with corresponding batch size (VGG-11, CIFAR-10).

| Batch | Model | Test acc., % | Top $\lambda$ | Trace | Batch | Model | Test acc., % | Top $\lambda$ | Trace |
|---|---|---|---|---|---|---|---|---|---|
| 32 | Model A | 85.13 | 13.98 | 584.52 | 512 | Model A | 86.93 | 6.37 | 367.13 |
| 32 | Model B | 85.38 | 15.61 | 611.05 | 512 | Model B | 86.29 | 8.17 | 506.03 |
| | Vanilla avg | 14.24 | 0.00 | 0.00 | | Vanilla avg | 22.07 | 0.00 | 0.00 |
| | A aligned to B | 85.11 | 13.97 | 609.28 | | A aligned to B | 86.94 | 6.38 | 353.85 |
| | Pred. avg | 87.56 | – | – | | Pred. avg | 88.29 | – | – |
| | OTFusion | 68.88 | 2.11 | 27.63 | | OTFusion | 75.38 | 3.36 | 53.58 |
| 32 | OTFusion FT | 87.8 | 4.92 | 291.77 | 512 | OTFusion FT | 87.95 | 4.05 | 242.58 |

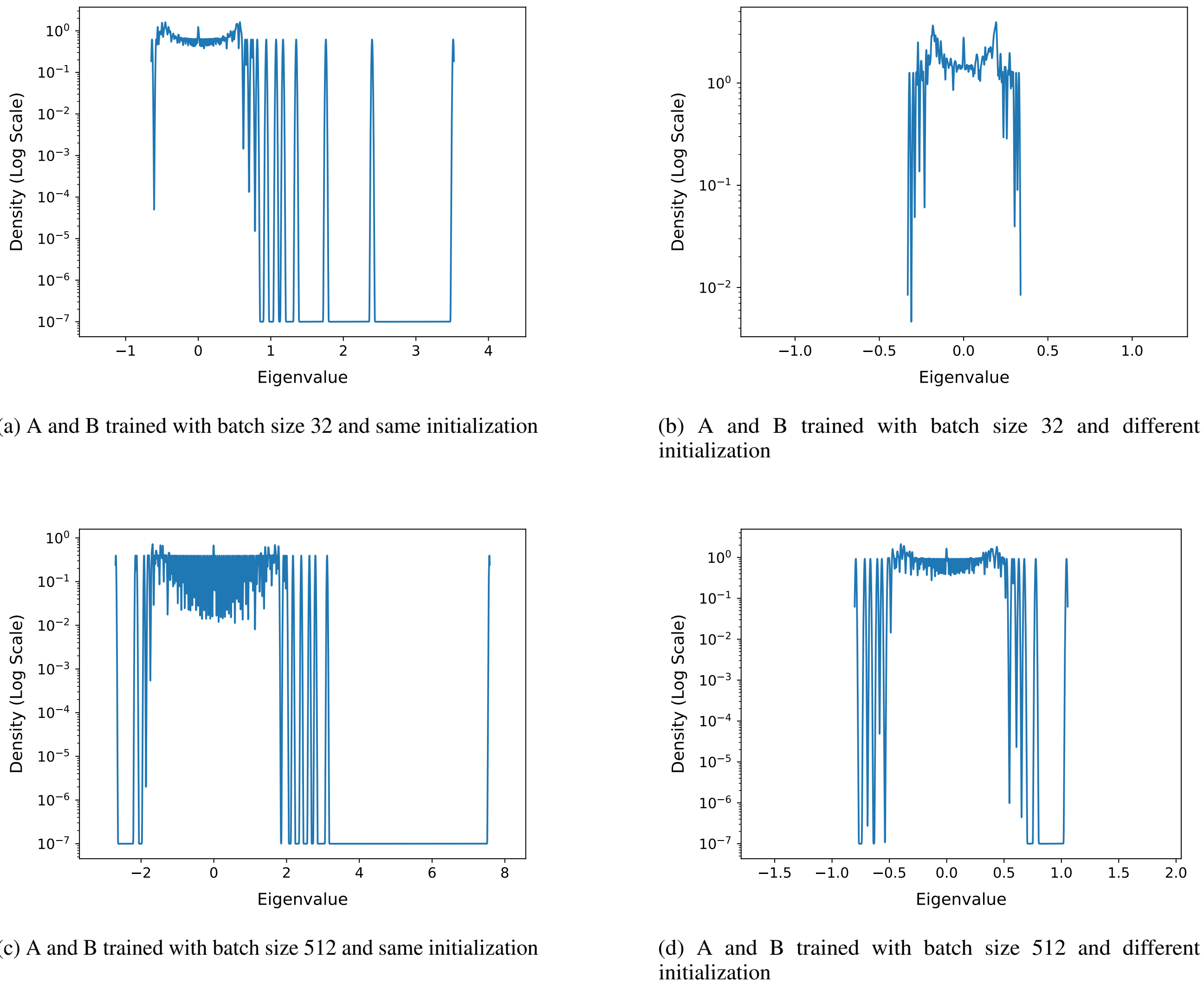
*Table 14.* Different initialization for A and B, trained with corresponding batch size (VGG-11, CIFAR-10).

## B.3. Eigenvalues Spectral Density

The following plots for the full eigenvalue spectral density (figures 3, 4, 5) were created making use of the publicly accessible PyHessian library [37]. They allow to gain more insights into the curvature of the loss landscape post-fusion, highlighting the differences between OT model fusion and vanilla averaging. In particular, they enable to study the presence of descent directions, helping in guiding the finetuning process. Specifically we looked for peaks or high-density regions in the negative part of the spectrum, which indicate the directions in which the loss function decreases. Moreover, the magnitude of negative eigenvalues can give an idea about the steepness of the descent.

### B.3.1. Vanilla Averaging

We first studied vanilla averaging. Most of the eigenvalues of the Hessian, as shown in figure 3, are in a neighborhood of 0, evidencing a relatively flat loss landscape post-averaging. Still, pronounced differences can be observed between the same and different initialization scenarios. In particular, the spectral densities for the different initialization case showcase notably shorter tails, whose symmetry is reflected in the almost null trace measured. These distinctions are expected, by looking at the difference in trace for vanilla averaging in Table 2 and 3. The amount of negative eigenvalues, though small in magnitude, is considerable and indicates the presence of several descent directions to explore.



(a) A and B trained with batch size 32 and same initialization

(b) A and B trained with batch size 32 and different initialization

(c) A and B trained with batch size 512 and same initialization

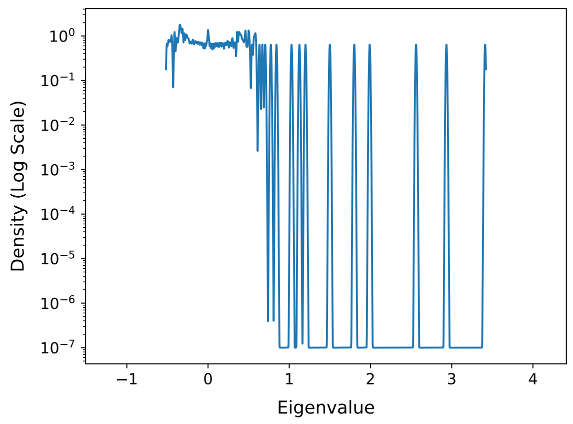
(d) A and B trained with batch size 512 and different initialization

*Figure 3.* Full Hessian Eigenvalues Spectral Density (ESD) for the vanilla averaging model on log-scale. Mind the different axis scales (ResNet18, CIFAR-10).

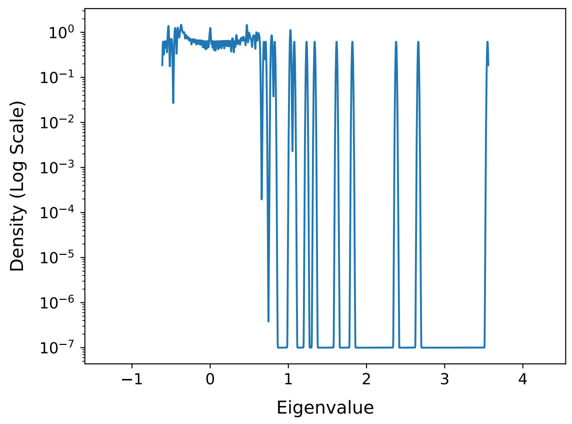
### B.3.2. OT Model Fusion

Again, most of the eigenvalues of the Hessian are in a neighborhood of 0, evidencing a relatively flat loss landscape post-fusion. Still, a greater amount of large magnitude positive eigenvalues, indicating ascent directions, can be observed. Furthermore, non-negligible presence of negative eigenvalues ensures the presence of descent directions to pursue during finetuning. One may notice that, differently from vanilla averaging, there are no tangible distinctions between the two initialization scenarios for what concerns OT model fusion. This is in line with the similarity of results in Table 2, 3. It can be seen that batch size 512 yields a slightly more pronounced right tail with respect to both batch 32 plots, confirming the greater steepness.

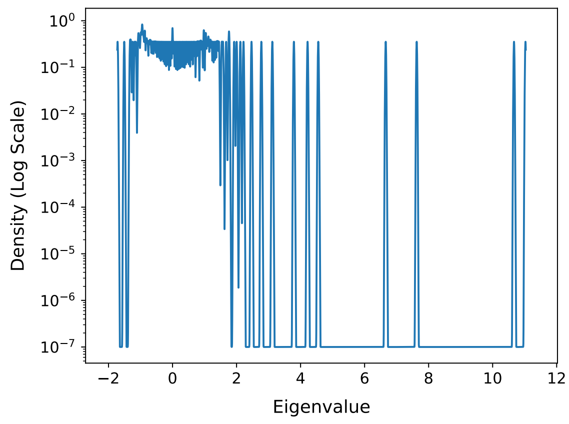
The mixed batch size cases in figure 5 shows an intermediate behavior, in comparison to the cases in figure 4. When the larger batch size parent is aligned onto the other one, the left-tail of the plot is slightly heavier.



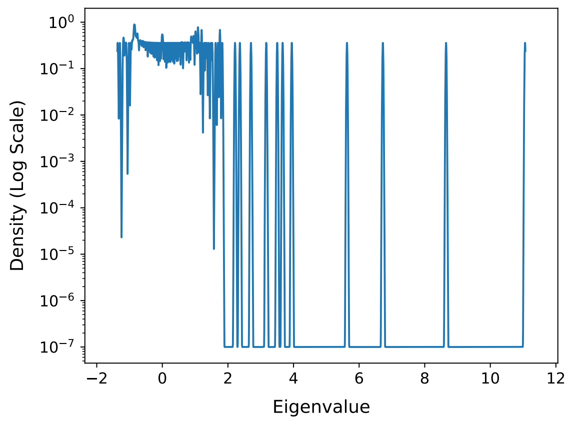
(a) A and B trained with batch size 32 and same initialization



(b) A and B trained with batch size 32 and different initialization

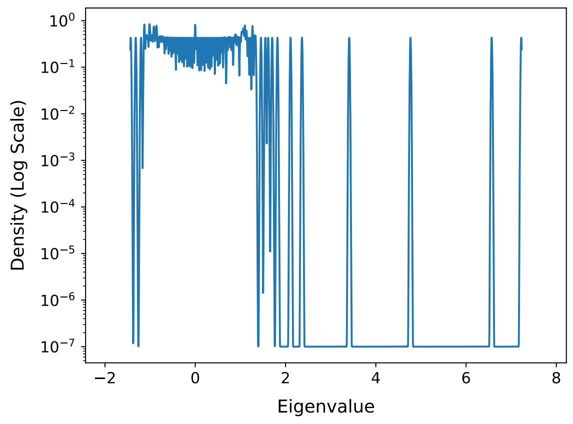


(c) A and B trained with batch size 512 and same initialization

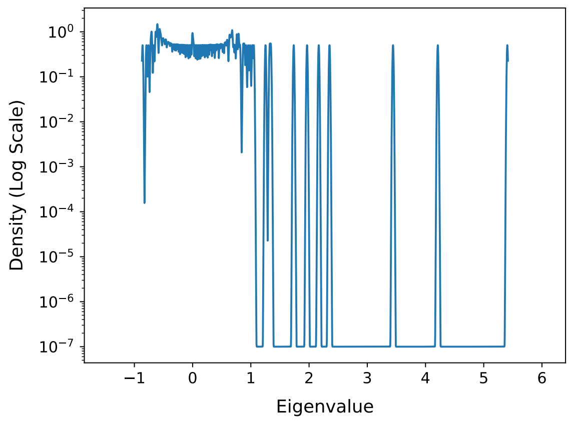


(d) A and B trained with batch size 512 and different initialization.

*Figure 4.* Full Hessian Eigenvalues Spectral Density (ESD) for the OT fused model on log-scale. A is aligned to B. Mind the different axis scales (ResNet18, CIFAR-10).



(a) A and B trained with batch size 32 and 512 respectively and different initialization

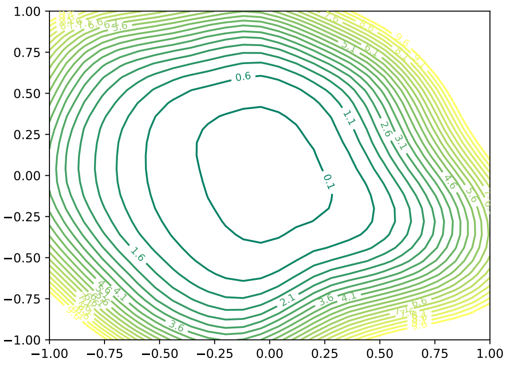


(b) A and B trained with batch size 512 and 32 respectively and different initialization

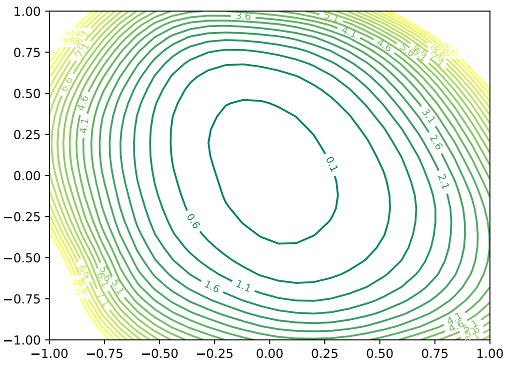
*Figure 5.* Full Hessian Eigenvalues Spectral Density (ESD) for the OT fused model on log-scale for the mixed batch sizes case. A is aligned to B. Mind the different axis scales (ResNet-18, CIFAR-10).

## B.4. Loss random directions

Making again use of [23], it was possible to visualize the training loss contours along 2 random directions for the OT fusion finetuned solutions. Despite the corresponding traces differ by a factor larger than 2 (43.04 vs 96.26, see 3), it was not possible to spot differences in the steepness on these 2 particular directions.



(a) Batch size 32



(b) Batch size 512

*Figure 6.* Training loss contours of finetuned solutions obtained after aligning and fusing models trained with different initialization (ResNet-18, CIFAR-10).