SHERPA: LEVERAGING NEURON ALIGNMENT FOR KNOWLEDGE-PRESERVING FINE-TUNING

Dongkyu Cho, Jinseok Yang, Jun Seo, Seohui Bae, Dongwan Kang, Hyeokjun Choe, Woohyung Lim*
LG AI Research
Seoul, South Korea
{dongkyu.cho, jinseok.yang, jun.seo, seohui.bae,
dongwan.kang, hyeokjun.choe, w.lim}@lgresearch.ai

ABSTRACT

Machine learning models commonly face challenges in maintaining robustness under distribution shifts. An effective strategy to address this issue involves fine-tuning selected layers of pre-trained models for adaptation. However, the lack of clear criteria for selecting trainable layers poses a significant obstacle, inevitably distorting pre-trained features amidst fine-tuning. This paper proposes a novel approach to this problem by analyzing the loss landscape of trained networks. By drawing insight from recent studies on neuron alignment, we conjecture that aligning models in their loss landscape will minimize the knowledge distortion during the fine-tuning process. Reflecting this, we introduce a novel fine-tuning framework, named SHERPA (Shifted basin for Enhanced Robustness via Permuted Activations), which shifts the training model towards the loss basin of a trained anchor model to encourage the preservation of pre-trained features. Empirical results demonstrate the effectiveness of SHERPA in enhancing Out-of-Distribution (OOD) robustness in multiple benchmarks (PACS, Terra Incognita, VLCS), without incurring additional overhead from gradient computations. Our work provides fresh perspectives in understanding how neural networks preserve and tune knowledge in the face of distribution shifts.

1 INTRODUCTION

An underlying assumption of machine learning is that the model will be applied to a target that is independent and identically distributed (i.e. i.i.d) to the trained data. In reality, this presumption is commonly violated by a discrepancy in distribution. This discrepancy, alias *distribution shift*, frequently hinders the performance of trained models (Kurakin et al., 2018). To mitigate this issue, a plethora of work is dedicated to learning robust models. An effective method is to fine-tune selective layers of pre-trained models to preserve general knowledge learned from the pre-trained distribution, while also reflecting the target distribution (Zhuang et al., 2020). Expanding this, more recent works suggest that tuning certain layers can outperform the entire model Lee et al. (2022). Yet, a clear criterion for the selection of trainable layers remains unclear, owing to our insufficient understanding of how neural networks preserve knowledge.

In this paper, we suggest a novel outlook to this question, centered on the loss landscape of trained networks (Simsek et al., 2021). Deriving from previous works on loss landscapes, we show that models sharing a loss basin share more pre-trained features (Neyshabur et al., 2020). Reflecting this, we conjecture that shifting the training model towards the basin of a well-trained model would help preserve pre-trained features that are critical for model robustness. Next, we revisit recent works on neuron alignment, which aligns individual models in their loss landscapes via permutation, to devise SHERPA (Shifted basin for Enhanced Robustness via Permuted Activations), a 2 stage fine-tuning framework that performs neuron alignment before the fine-tuning stage. Specifically, SHERPA shifts the training model towards the robust basin of the pre-trained anchor model to preserve pre-trained knowledge. We show that SHERPA effectively enhances the OOD robustness

*Corresponding Author

across multiple benchmark datasets, without the additional cost of gradient computations. Furthermore, we provide an extensive analysis of the effect of neuron alignment in the parameter space, and the loss geometry of trained models.

We state our contributions threefold. (1) We reveal that neuron alignment can help preserve pre-trained knowledge amidst fine-tuning by exploiting the loss basin of trained models (2) We present a 2 stage fine-tuning method SHERPA that enhances OOD generalizability without the additional cost of gradient computation. We show the strength of SHERPA in boosting OOD performance under severe distribution shifts. (3) We demonstrate that neuron alignment offers insights into how neural networks preserve and tune knowledge, revealing promising avenues for further exploration.

2 RELATED WORKS

Exploiting pre-trained models Leveraging the knowledge of pre-trained models is a longstanding area of investigation (Tan et al., 2018). The key motivation behind utilizing pre-trained models lies in their effectiveness in improving model performance under both in-distribution (ID) and out-of-distribution (OOD) settings. A well-accepted idea is that fine-tuning an entire model inevitably distorts pre-obtained knowledge (Kumar et al., 2022), leading to a performance drop in OOD settings. Reflecting this, recent works in transfer learning tune selected layers of a model to preserve knowledge during model adaptation (Kirichenko et al., 2022; Lee et al., 2022; Kaplun et al., 2023).

Neuron Alignment The idea behind neuron alignment revolves around finding the optimal method to merge multiple models in their weight space (i.e. Model Fusion). Initially introduced in the federated learning literature (Wang et al., 2020), model fusion has gained considerable attention for its generalization capability (Wortsman et al., 2022; Rame et al., 2022; 2023). A crucial condition for model fusion is that the models being merged must occupy the same loss basins (Neyshabur et al., 2020; Gontijo-Lopes et al., 2021) to ensure Linear Mode Connectivity (LMC) (Frankle et al., 2020; Juneja et al., 2022). Recent works aim to overcome this constraint via neuron alignment, capitalizing on the permutation-invariance property of neural networks (Entezari et al., 2021). Here, the *permutation invariance* of neural architectures refers to the phenomenon that replacing the i^{th} weight matrix W_i in the model with $P \cdot W_i$, where P is a permutation matrix, and the $i + 1^{th}$ weight matrix as $W_{i+1} \cdot P^{-1}$ to reflect the permutation in the previous layer, can represent the identical function as before, which can be utilized to align individual models in their loss landscapes (Ainsworth et al., 2022; Jordan et al., 2022; Nguyen et al., 2023; Stoica et al., 2023).

3 METHOD: LEVERAGING NEURON ALIGNMENT FOR ROBUST FINE-TUNING

We present a simple fine-tuning framework SHERPA that leverages neuron alignment for the preservation of pre-trained knowledge. SHERPA is a 2 stage framework that (1) aligns the training model with the teacher model in their loss landscape and then (2) fine-tunes the model with the source data.

Notation We begin by defining the notations. In our problem formulation, we aim to train a model M with the source distribution P_{src} and test M's robustness across both the source P_{src} and the OOD target distribution P_{tgt}, similar to the domain generalization setup (Gulrajani & Lopez-Paz, 2020). In our setting, M is a pre-trained model trained on pre-training distribution P_{pt}. Finally, we introduce an anchor model A which is pre-trained on the distribution P_A and then trained on P_{src}.

Problem Formulation & Setup Without any additional procedures, model M pre-trained on P_{pt} and fine-tuned on the target P_{src} tends to display unexpectedly high OOD performance on the target data P_{tgt} (Gulrajani & Lopez-Paz, 2020). The generalizability of M is likely the result of the knowledge learned from P_{pt}, as discussed in Kumar et al. (2022); Li et al. (2022). However, there are cases where P_{pt} is largely different from the source and target distribution, such that naively fine-tuning on P_{src} will distort pre-trained knowledge, limiting M's generalizability on target P_{tgt}.

Reflecting on this, this paper aims to devise a method that can minimize the distortion of pre-trained knowledge while fine-tuning M, such that its OOD robustness is maintained. To reflect realistic settings, we add some additional constraints: (1) *Limited Data*: Similar to the domain generalization setting (Zhou et al., 2022), we limit our available data to P_{src}. In other words, M can only be trained on limited source data. This constraint is added to display that our manipulations during the

fine-tuning process have clear effects on the model's post-training OOD generalizability. (2) *Limited Resources*: In this setting, we can imagine leveraging auxiliary information for guidance. A possible method would be to use an additional model A to regularize the learning procedure, namely in the form of knowledge distillation. Yet, this approach normally requires excessive computation costs for gradient updates. In contrast, we aim to limit the use of computing resources.

3.1 SHERPA: SHIFTED BASIN FOR ENHANCED ROBUSTNESS VIA PERMUTED ACTIVATIONS

ShERPA aims to preserve pre-trained knowledge by leveraging neuron alignment. The fundamental of our method is rooted in the idea that models closely located in the loss landscape share more pre-trained features (Neyshabur et al., 2020). In this sense, we conjecture that keeping a model M in the vicinity of a trained *anchor* model A's loss basin will minimize the distortion of pre-trained knowledge during the training of M. Following this, we adopt an activation-matching method for neuron alignment (Li et al., 2016; Jordan et al., 2022), which shifts the model towards the loss basin of the anchor (Entezari et al., 2021).

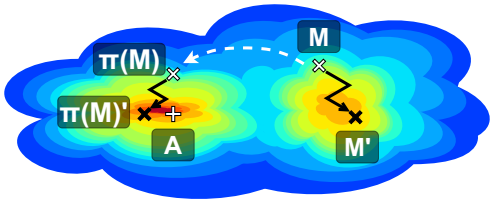


Figure 1: **Illustration of SHERPA.** SHERPA leverages neuron alignment to shift the model M towards the loss basin of the anchor model A, using permutation π. SHERPA aims to exploit the rich loss basin of A during training.

Let us denote the model weight of the anchor model A and the training model M as Θ_A and Θ_M, respectively. Our neuron alignment algorithm aims to find a set of permutations $\pi = (P_1, P_2 \dots P_L)$ that aligns L-layer networks A and $\pi(M)$ in their weight space. In specific, for a batch of samples, we search for a permutation P_l that maximizes Equation (1) for i^{th} hidden units in the l^{th} layer:

$$\sum_i \text{corr}\big(X^A_{(l,i)}, X^M_{(l,P_l(i))}\big), \tag{1}$$

where $X^A_{(l,i)}, X^M_{(l,P_l(i))}$ refers to the random variables representing the activations of the i^{th} hidden units in the l^{th} layer in models A and M, respectively (Jordan et al., 2022). The effectiveness of using correlation values to measure the degree of relation between the units was studied in Li et al. (2016). Optimizing Equation (1) maximizes the sum of correlations between the activations between the two models, which is a Linear Assignment Problem (Bertsekas, 1998) that can be solved at polynomial-time using combinatorial optimization methods (e.g., Hungarian algorithm (Kuhn, 1955), Jonker-Volgenant algorithm (Jonker & Volgenant, 1988)) without excess computation.

Intuitively, the neuron alignment between the anchor A and the model M is similar to shifting M towards the loss basin of A, as empirically shown in Entezari et al. (2021). Now, consider a scenario where A is a well-trained model with a flat and wide loss landscape that possesses advantages in out-of-distribution (OOD) generalization (Hochreiter & Schmidhuber, 1997; Cha et al., 2021; Iyer et al., 2023). If neuron alignment transports a model into another model's robust basin, we conjecture that aligning the model before fine-tuning can enhance the robustness of M by minimizing the distortion of pre-trained features.

Deriving from this, SHERPA first aligns the new training model M with the trained model A, such that the permuted model ($\pi(M)$) resides in the loss basin of A. Afterward, the permuted model $\pi(M)$ is trained on the source distribution P_{src} with the ERM loss (Gulrajani & Lopez-Paz, 2020), which is the cross-entropy loss for the image classification task written as:

$$L_{ce}(\mathbf{y}, \hat{\mathbf{y}}) = -\sum\nolimits_i y_i \log(\hat{y}_i), \tag{2}$$

with $\mathbf{y}$ the ground truth, $\hat{\mathbf{y}}$ the softmax prediction of the model. An overview of our method is illustrated in Figure 1. We conjecture that the neuron aligned $\pi(M)$ will benefit from training on the loss basin of A, preserving more of its pre-trained features. Specifically, we expect that the fine-tuned $\pi(M)'$ will display higher OOD robustness. In Section 4, we present that empirical results support our conjecture, consistent with prior research on neuron alignment (Singh & Jaggi, 2020; Entezari et al., 2021). Furthermore, we provide extensive analysis on SHERPA in Appendix A to show SHERPA's effect in preserving pre-trained features.

4 EXPERIMENTS

Datasets & Implementation To test the effect of neuron alignment on model robustness, we select Domain Generalization (DG) benchmarks (e.f., PACS (Li et al., 2017), Terra Incognita (Beery et al., 2018) and VLCS (Fang et al., 2013)) that display significant distribution shifts across multiple datasets. Details regarding the dataset are reported in Appendix C.1. Following the general DG setting (Gulrajani & Lopez-Paz, 2020), we fix the architecture of the model M and the anchor A as ResNet-50 (He et al., 2016). To clarify, the model M is not yet trained on the source dataset P_{src}, while the anchor A is trained on the source P_{src}. Implementation details are reported in Appendix C.

4.1 EXPERIMENTAL RESULTS AND ANALYSIS

Experiment on PACS The PACS experiment aims to display the effect of neuron alignment on enhancing the robustness of the trained model. The results are reported in Table 1, where A[1], C, P, and S refer to the unseen target dataset. We find that aligning M with the anchor A indeed boosts OOD accuracy. Specifically, we compare the average OOD accuracy of the baseline ERM (84.3) with SHERPA (86.9), where the two methods only differ in their use of neuron alignment. Our method also outperformed LP-FT (Kumar et al., 2022), while falling behind an ensemble of 6 models, which we consider as the upper bound owing to its innate strength in generalizability (Arpit et al., 2022). Yet ensembles are heavier in terms of computation cost, while our method is much lighter. Notably, we find that the performance of A has no significant impact on SHERPA, which are discussed in Appendix A. Furthermore, we observe that random permutation affects the OOD performance while displaying high fluctuations compared to aligning towards A (SHERPA), which is in line with the idea of Entezari et al. (2021) that permutation is functionally equivalent to changing the initialization, transporting models to a different loss basin.

Table 1: Accuracy on PACS.

Method	A	C	P	S	Avg.
ERM	91.22	80.63	98.03	67.32	84.3 ±0.2
Ensemble (m=6)	91.19	82.47	98.84	77.90	**87.6**
LP-FT (Kumar et al., 2022)	91.17	81.21	98.45	73.57	86.1 ±0.5
Random Perm.	87.80	84.64	97.85	71.06	85.3 ±**2.1**
SHERPA (Ours)	90.00	83.53	97.62	76.48	**86.9** ±0.1

Experiment on Terra Incognita Here, we present the results of the Terra Incognita experiment at Table 2, where L100, L38, L43, and L46 refer to the unseen target datasets. Similar to the PACS experiment, models fine-tuned after neuron alignment showed stronger OOD accuracy (48.3) compared to the vanilla fine-tuned baseline (47.4). Yet unlike PACS, our method was behind the ensemble model (52.3) by a large margin. However, we would like to note that the reported performance of the ensemble model in Rame et al. (2022) (49.2) was lower than our run (52.3). Like other experiments, applying random permutation affected the OOD accuracy of the permuted model, displaying high fluctuations in OOD performance. This fluctuation is an expected behavior, as random permutation is similar to aligning with a random anchor. In the Terra Incognita dataset, our method slightly fell behind LP-FT but displayed stronger stability with smaller fluctuations ($\pm 0.2 < \pm 0.5$).

Table 2: Accuracy on Terra Incognita.

Method	L100	L38	L43	L46	Avg.
ERM	61.11	40.15	48.54	40.00	47.4 ±0.4
Ensemble (m=6)	57.73	46.16	61.46	43.75	52.3
LP-FT (Kumar et al., 2022)	64.17	42.71	44.98	42.24	**48.5** ±0.5
Random Perm.	62.56	42.87	46.41	40.37	48.1 ±**0.7**
SHERPA (Ours)	64.63	41.28	45.47	41.78	**48.3** ±**0.2**

Experiment on VLCS In Table 3, we report the experimental results in the VLCS dataset. Here, C, L, V, and S refer to the target dataset. Notably, we observe that a similar pattern is repeated in the VLCS dataset, where neuron alignment positively affected OOD accuracy compared to the baseline ($80.8 > 80.5$), while the performance gap between our method and the baseline is smaller than in other experiments. Furthermore, the effect of random permutation was observed similar to other experiments, suffering a strong level of fluctuation in the OOD accuracy. In VLCS, our method showed very similar results to LP-FT, while we believe our method to be slightly more reliable.

Table 3: Accuracy on VLCS.

Method	C	L	S	V	Avg.
ERM	98.59	66.53	76.51	80.24	80.5 ±0.3
Ensemble(m=6)	98.02	66.11	78.55	81.61	81.0
LP-FT (Kumar et al., 2022)	99.08	67.10	76.44	80.58	**80.8** ±0.3
Random Perm.	97.40	63.00	72.50	76.30	77.3 ±**3.8**
SHERPA (Ours)	99.22	66.19	75.47	82.43	**80.8** ±0.2

Effect of Neuron Alignment on OOD performance In Section 3.1, we conjectured that if neuron alignment algorithms transport model M into a trained anchor A's loss basin, the permuted model $\pi(M)$ would be able to preserve its pre-trained knowledge amidst training on the source distribution P_{src}. In specific, we expected that the fine-tuned $\pi(M)'$ would display superior OOD robustness than the fine-tuned M', as it supposedly preserved pre-trained features (Neyshabur et al., 2020).

[1]Please note that this is not the anchor A.

In the above experiments, we have empirically shown that applying neuron alignment before fine-tuning changes the OOD accuracy of trained models. Similar to our expectations, when the anchor A is a trained model, the OOD accuracy of the fine-tuned $\pi(M)'$ (SHERPA) surpassed the baseline M' (ERM), as seen in Table 1. We interpret that these results support our conjecture above. Furthermore, when we applied random permutations to M, the permuted model showed a varying performance after fine-tuning. Once again, this display of randomness coincides with our expectations that neuron alignment is indeed functionally equivalent to transporting the loss basin. Furthermore, these results support the conjectures made by Entezari et al. (2021) that applying random permutations to an SGD solution approximates training models with varying initialization. In Appendix A, we provide a deeper analysis of the effect of neuron alignment on the model (e.g., model parameters, feature representations, loss geometry).

5 DISCUSSION

Limitations In this section, we discuss the limitations of our work. Our framework uses an anchor model of an identical architecture as the training model and is fine-tuned on the target dataset. The first issue is the availability of an anchor model of identical architecture. Please note that for neuron alignment, a model of identical architecture is necessary to ensure a 1:1 matching between individual neurons of the anchor and the model Li et al. (2016); Ainsworth et al. (2022). However, in reality, such models may not be readily available. We believe that further study is required to perform neuron alignment between models of varying sizes Imfeld et al. (2024). Another limitation is the redundancy of using a fine-tuned anchor model to guide the fine-tuning process. Whilst we believe that the OOD performance boost effect justifies this small redundancy, we view that there is room for further investigation.

Future Work In this section, we suggest a possible future work. In specific, we are interested in using neuron alignment to develop a new layer-selection criterion for selective fine-tuning. It is a well-established idea that selectively fine-tuning subset layers of a model is sufficient to adapt the entire model, as described in previous works (Kirichenko et al., 2022; Rosenfeld et al., 2022). Notably, (Lee et al., 2022) demonstrated that the selection of fine-tuning layers is influenced by the type of distribution shift. We believe that there is room for improvement in designing an effective criterion for layer selection. Our idea is that analyzing the effect of neuron alignment can provide insights into how neural networks change amidst distribution shifts. In specific, we compare each layer of the original model M and the permuted model $\pi(M)$ and compute their distance. We conjecture that the distance can be used as a criterion for layer selection. In our setting, the two corresponding layers in M and $\pi(M)$ are composed of the same weights, but with different orders. Reflecting this, we adopt the Kendall-Tau rank distance as a distance metric, which measures the number of permutations (swaps) required to transform M into $\pi(M)$ (Kendall, 1938). Interestingly, we find that selectively fine-tuning layers with high Kendall-Tau distance before/after neuron alignment tend to display higher OOD performance, with some exceptions. We find potential in extending this observation to design a layer-selection criterion for selective (Lee et al., 2022) fine-tuning.

6 CONCLUSION

This paper proposes a novel fine-tuning framework for model robustness, namely SHERPA, which leverages neuron alignment to preserve useful pre-trained knowledge amidst fine-tuning. The effectiveness of our method in increasing OOD performance is successfully demonstrated across three benchmarks (PACS, Terra Incognita, VLCS). We also show the effect of neuron alignment on the model's parameter space and illustrate its effect through a visualization of the loss surface. The ablation study reinforces the effectiveness of our method across varying environments and hyperparameters. More importantly, our research offers new insights into understanding how neural networks behave against distribution shifts. Exploring neuron alignment for enhancing model robustness holds promise for further development, including establishing layer-selection criteria for fine-tuning.

REFERENCES

Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representa-*

tions, 2022.

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. Ensemble of averages: Improving model selection and boosting performance in domain generalization, 2022.

Sara Beery, Grant van Horn, and Pietro Perona. Recognition in terra incognita, 2018.

Dimitri Bertsekas. *Network optimization: continuous and discrete models*, volume 8. Athena Scientific, 1998.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. In *International Conference on Learning Representations*, 2021.

Daniel Falbel. *torchvision: Models, Datasets and Transformations for Images*, 2023. https://torchvision.mlverse.org, https://github.com/mlverse/torchvision.

Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1657–1664, 2013.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pp. 3259–3269. PMLR, 2020.

Raphael Gontijo-Lopes, Yann Dauphin, and Ekin Dogus Cubuk. No one representation to rule them all: Overlapping features of training methods. In *International Conference on Learning Representations*, 2021.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2020.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, Los Alamitos, CA, USA, jun 2016. IEEE Computer Society. doi: 10.1109/CVPR.2016.90.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

Moritz Imfeld, Jacopo Graldi, Marco Giordano, Thomas Hofmann, Sotiris Anagnostidis, and Sidak Pal Singh. Transformer fusion with optimal transport, 2024.

Nikhil Iyer, V Thejas, Nipun Kwatra, Ramachandran Ramjee, and Muthian Sivathanu. Wide-minima density hypothesis and the explore-exploit learning rate schedule. *Journal of Machine Learning Research*, 24(65):1–37, 2023.

Roy Jonker and Ton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. In *DGOR/NSOR: Papers of the 16th Annual Meeting of DGOR in Cooperation with NSOR/Vorträge der 16. Jahrestagung der DGOR zusammen mit der NSOR*, pp. 622–622. Springer, 1988.

Keller Jordan, Hanie Sedghi, Olga Saukh, Rahim Entezari, and Behnam Neyshabur. Repair: Renormalizing permuted activations for interpolation repair. *arXiv preprint arXiv:2211.08403*, 2022.

Jeevesh Juneja, Rachit Bansal, Kyunghyun Cho, João Sedoc, and Naomi Saphra. Linear connectivity reveals generalization strategies. In *The Eleventh International Conference on Learning Representations*, 2022.

Gal Kaplun, Andrey Gurevich, Tal Swisa, Mazor David, Shai Shalev-Shwartz, and Eran Malach. Subtuning: Efficient finetuning for multi-task learning. *arXiv preprint arXiv:2302.06354*, 2023.

Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMLR, 2019.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pp. 99–112. Chapman and Hall/CRC, 2018.

Yoonho Lee, Annie S Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. *arXiv preprint arXiv:2210.11466*, 2022.

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

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets, 2018.

Yixuan Li, Jason Yosinski, Jeff Clune, Hod Lipson, and John Hopcroft. Convergent learning: Do different neural networks learn the same representations?, 2016.

Ziyue Li, Kan Ren, Xinyang Jiang, Bo Li, Haipeng Zhang, and Dongsheng Li. Domain generalization using pretrained models without fine-tuning, 2022.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.

Dang Nguyen, Trang Nguyen, Khai Nguyen, Dinh Phung, Hung Bui, and Nhat Ho. On cross-layer alignment for model fusion of heterogeneous neural networks. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Alexandre Rame, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. Diverse weight averaging for out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 35:10821–10836, 2022.

Alexandre Rame, Kartik Ahuja, Jianyu Zhang, Matthieu Cord, Léon Bottou, and David Lopez-Paz. Model ratatouille: Recycling diverse models for out-of-distribution generalization. *PMLR*, 2023.

Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

Berfin Simsek, François Ged, Arthur Jacot, Francesco Spadaro, Clément Hongler, Wulfram Gerstner, and Johanni Brea. Geometry of the loss landscape in overparameterized neural networks: Symmetries and invariances. In *International Conference on Machine Learning*, pp. 9722–9732. PMLR, 2021.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33:22045–22055, 2020.

George Stoica, Daniel Bolya, Jakob Bjorner, Taylor Hearn, and Judy Hoffman. Zipit! merging models from different tasks without training. *arXiv preprint arXiv:2305.03053*, 2023.

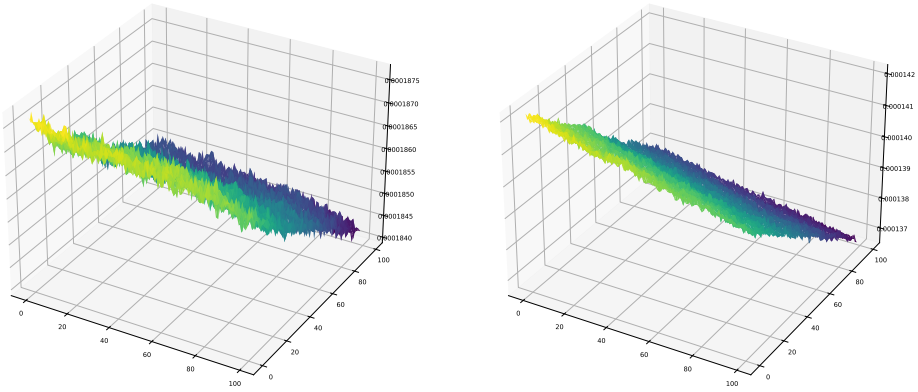
Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, pp. 270–279. Springer, 2018.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *International Conference on Learning Representations*, 2020.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pp. 23965–23998. PMLR, 2022.

Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2022. ISSN 1939-3539. doi: 10.1109/tpami.2022.3195549. URL http://dx.doi.org/10.1109/TPAMI.2022.3195549.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1): 43–76, 2020.



(a) Vanilla Fine-tuning (ERM) (b) SHERPA

Figure 2: The loss surface of trained models

A ANALYSIS

In this section, we present our analysis on SHERPA. Specifically, we illustrate our view on how neuron alignment affects the learning process, as well as the model's loss landscape.

Effect of Neuron Alignment on Model Parameters To display that neuron alignment is helpful for the preservation of pre-trained knowledge amidst fine-tuning, we present an analysis of the model in the parameter space. In specific, we analyze the ℓ_2 distance of the models before and after fine-tuning, following the practice of Neyshabur et al. (2020).

Table 4: ℓ_2 distance of ResNet-50 parameters before/after fine-tuning

Method	Conv1	Layer1	Layer2	Layer3	Layer4
Epochs=1					
ERM	0.0195	0.159	0.210	0.702	0.814
SHERPA (Ours)	**0.0159**	**0.127**	0.266	0.858	**0.674**
Epochs=10					
ERM	0.0395	0.282	0.631	2.235	2.257
SHERPA (Ours)	**0.0263**	0.289	0.669	**2.125**	**2.006**
Epochs=30					
ERM	0.0333	0.367	0.753	3.776	2.696
SHERPA (Ours)	**0.0293**	**0.343**	1.141	**3.736**	**2.417**

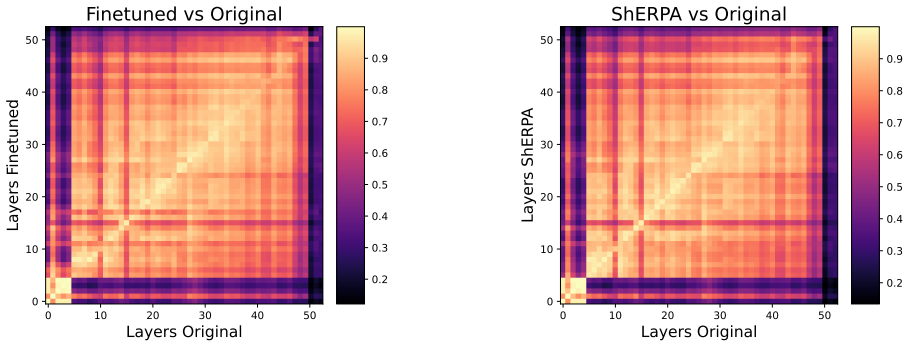
We compare the ℓ_2 distance between the layers/blocks of (1) the model before fine-tuning and (2) the model after fine-tuning. Specifically, we compare the 1^{st} convolution layer (Conv1), the 1^{st}, 2^{nd}, 3^{rd}, and 4^{th} layers (block) of the ResNet-50. The fine-tuning setting follows the general setting in Section 4, with a learning rate of 0.001 using the SGD optimizer with a momentum of 0.9, as depicted in Appendix C. We repeat this experiment across various training epochs, to see how the training model deviates from the original model during fine-tuning.

We share the results in Table 4. In the 1^{st} fine-tuning epoch, our method shows a slightly reduced ℓ_2 distance compared to the baseline in Conv1, Layer1, and Layer4. In the 10^{th} fine-tuning epoch, we observe that the gap has grown. Notably in Layer1, there is a very small ℓ_2 gap (0.007). On the other hand, Layer2 has changed significantly in our method(1.141) compared to that of the baseline (0.753). Lastly, in the 30^{th} epoch, we observe that a similar pattern is repeated, where Layer2 has significantly changed in our method (1.141). ℓ_2 distance of Layer1 (0.343), on the other hand, has become smaller than that of the baseline ERM (0.367). We believe that this result displays how the neuron-aligned model preserves knowledge by selectively updating subsets of the model.

It is a well-established idea that selectively fine-tuning subset layers of a model is sufficient to adapt the entire model, as described in previous works (Kirichenko et al., 2022; Rosenfeld et al., 2022). Notably, Lee et al. (2022) demonstrated that the type of distribution shift influences the selection of fine-tuning layers. We find potential in connecting this observation to recent studies in selective fine-tuning (Lee et al., 2022), specifically in designing a layer-selection criterion for layer-selective fine-tuning.

Table 5: Feature Similarity (CKA) of ResNet-50 before/after fine-tuning

Method	Conv1	Layer1	Layer2	Layer3	Layer4
Epochs=30					
ERM	1.0000	0.8784	0.9610	0.8452	0.3622
SHERPA (Ours)	1.0000	**0.8998**	**0.9628**	0.8089	0.3547



(a) Vanilla Fine-tuning (ERM) (b) SHERPA

Figure 3: Layer-wise Feature Similarity (CKA) of ResNet-50 before/after fine-tuning

Effect of Neuron Alignment on the Feature Representations In this paragraph, we provide an analysis of the effect of neuron alignment on the feature representations. We follow the practice of Neyshabur et al. (2020) and compute the Centered Kernel Alignment (CKA) metric (Kornblith et al., 2019) between the model before and after fine-tuning.

The CKA metric measures the similarity between two feature representations given two models. In our case, we compare two cases (1) *ERM (Baseline)*: CKA similarity of the model before/after fine-tuning ($M \leftrightarrow M'$), (2) SHERPA *(Ours)*: CKA similarity of the neuron-aligned model before/after fine-tuning ($\pi(M) \leftrightarrow \pi(M)'$). In specific, we compute the feature similarity for different layers of the network, which is the ResNet-50 for our experimental setting. The feature similarity was computed on the PACS dataset, under the same setting as our experiment in Section 4 (30 epochs fine-tuning; a learning rate of 0.001; using the SGD optimizer with a momentum of 0.9). We find that applying neuron alignment alters the feature similarity of the trained model before/after fine-tuning, while further investigation is required. The experimental results of layer-wise feature similarity analysis are reported in Table 5. The results are also visualized in Figure 3. Here, the diagonal values of the matrix indicate the feature similarity between the corresponding layers of the model before/after fine-tuning.

Effect of Neuron Alignment on the loss geometry We can also visually observe the effect of neuron alignment on the loss landscape. We find that applying neuron alignment smoothens the loss surface of the model. As visualized in Figure 2b, the sharpness of the loss landscape is significantly different from that of the baseline method in Figure 2a. The smoothening effect of SHERPA can also be seen in the loss contour plot Figure 4b, in comparison to that of the baseline in Figure 4a. We view this as evidence that neuron alignment transports the model across the loss landscape. For visualization, we adopted a filter normalization-based method introduced in Li et al. (2018).

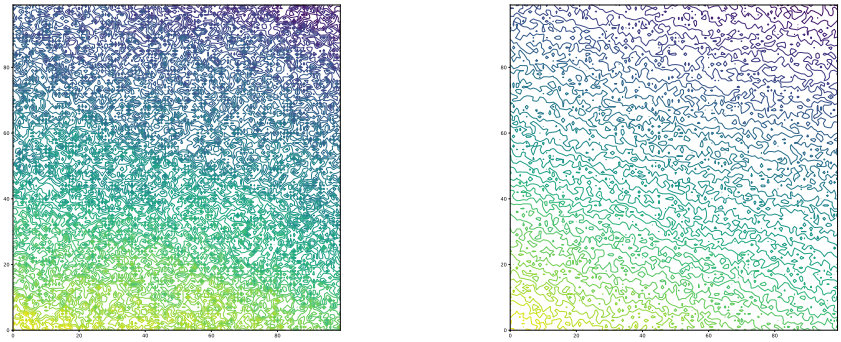
B ABLATION STUDY

B.1 STUDY ON ANCHOR

Here we report experimental results showing that SHERPA's effects are not limited by the performance of the anchor A. Specifically, we compare the OOD accuracy of SHERPA-trained models under *varying* anchors: (1) *Random Permutation*: Instead of an anchor, we randomly permute the model. (2) *Single Model (A=1)*: The anchor is a pre-trained model fine-tuned on the source domain P_{src}. (3) *Ensemble (A=6)*: Weight-averaged ensemble of 6 models sharing a pre-trained backbone, each fine-tuned on the source domain P_{src}. In all cases, the anchor A uses a different pre-trained backbone as the model M, to ensure that they are

Table 6: Ablation study on Anchor (PACS)

Method	A	C	P	S	Avg.
ERM	91.22	80.63	98.03	67.32	84.3 ±0.2
Random Perm.	87.80	84.64	97.85	71.06	85.3 ±**2.1**
SHERPA (Ours, A=1)	90.00	83.53	97.62	76.48	**86.9** ±0.1
SHERPA (Ours, A=6)	89.91	84.27	97.70	76.91	**87.2** ±0.2



(a) Vanilla Fine-tuning (ERM) (b) SHERPA

Figure 4: The loss contour of trained models

located on a different basin before neuron alignment. In terms of performance, the ensemble anchor (A=6) outperforms the single anchor (A=1).

In Table 6, we share the results of the anchor experiment, performed on the PACS dataset. Interestingly, we find that the anchor's performance does not directly affect M's performance. Specifically, using a better-performing anchor (87.2) did not show superior performance than using a single anchor (86.9). While this can be partially explained by the idea that the loss landscape is shared across multiple anchors, we believe further analysis is required.

B.2 Study on Hyperparameters

In this section, we share our study on hyperparameters. In essence, the first stage of SHERPA does not require specific hyperparameters, as neuron alignment (i.e. Activation Matching) requires no additional hyperparameters. For the second-stage fine-tuning, we perform an ablation study on the *learning rate* and the number of *training epochs*. Experimental results show that the effect of SHERPA on OOD accuracy is present across all experimental settings.

Table 7: Ablation Study on Learning Rate (PACS)

Learning Rate	A	C	P	S	Avg.
ERM* ($lr = 0.001$)	91.22	80.63	98.03	67.32	84.3 ±0.2
ERM ($lr = 0.0005$)	86.73	74.83	98.15	67.52	81.8 ±0.3
ERM ($lr = 0.0001$)	88.19	68.22	98.09	55.41	77.5 ±0.2
Ours* ($lr = 0.001$)	90.00	83.53	97.62	76.48	**86.9** ±0.1
Ours ($lr = 0.0005$)	88.04	82.25	98.63	69.56	**84.6** ±0.1
Ours ($lr = 0.0001$)	85.41	68.90	98.45	62.23	**78.8** ±0.2

Learning Rate To test the reliability of our method (SHERPA), we perform an ablation study on the learning rate of SHERPA's second stage fine-tuning. Here, we report results under varying learning rates $(0.001, 0.0005, 0.0001)$ for 30 epochs in the PACS benchmark. Note that, for our experiments in Section 4, we have set the learning rate as 0.001. As seen in Table 7, we observed that in all cases, our method SHERPA outperformed the baseline ERM. In Table 7, the results marked with an asterisk are the results reported in Table 1.

Table 8: Ablation Study on Training Epochs (PACS)

Epochs	A	C	P	S	Avg.
ERM (5)	87.99	75.00	96.90	66.33	81.6 ±0.3
ERM (10)	87.75	80.46	98.57	71.46	84.6 ±0.4
ERM (20)	89.85	79.39	98.09	70.07	84.4 ±0.2
ERM* (30)	91.22	80.63	98.03	67.32	84.3 ±0.2
Ours (5)	84.13	76.83	97.68	72.38	**82.8** ±0.2
Ours (10)	87.75	81.74	98.15	70.86	**84.7** ±0.1
Ours (20)	89.21	81.57	97.85	70.98	**84.9** ±0.1
Ours* (30)	90.00	83.53	97.62	76.48	**86.9** ±0.1

Training Epochs In this section, we show that SHERPA consistently outperforms the baseline regardless of changes in training epochs. In Table 8, we report experimental results under varying training epochs $(5, 10, 20, 30)$ in the PACS benchmark. In all cases, we find that applying neuron alignment positively affects the OOD performance of the trained model. Notably, We find that the gap between our method and the baseline is the largest when the training epochs are set as 30. In Table 8, the results marked with an asterisk are the results reported in Table 1.

C IMPLEMENTATION DETAILS

In this section, we report the implementation details of our experiments.

C.1 DATASET

Here, we elaborate on the datasets used in our work. All three datasets are benchmark datasets used to test model generalizability in tasks such as Domain Generalization and Single-source Domain Generalization (Gulrajani & Lopez-Paz, 2020; Zhou et al., 2022).

PACS PACS is a common benchmark in the field of domain generalization. PACS combines images categorized into 7 classes from 4 datasets (Photo (P), Art Painting (A), Cartoon (C), and Sketch (S)). Generally, the model is trained on the source domain, and tested on the target domain, where domain refers to the distribution of the dataset. In our experiments, we adopt the leave-one-out method to measure OOD robustness, which refers to selecting one dataset (e.g., A) as the target domain, while the rest are used as the source domain. The model is first trained on the 3 source domains (e.g., P, C, S), and tested on the unseen target domain A. The target domain is switched for cross-validation, and the average OOD accuracy is used to measure a model's OOD robustness, as reported in Table 1.

Terra Incognita The Terra Incognita benchmark is designed to test the OOD robustness of a trained model. Terra Incognita is a collection of multiple datasets, where each collected was from different camera traps in the wild. The camera traps were placed stationary, hence there is minimal room for spurious correlation with the background. The Terra Incognita experiment in our work uses 4 datasets ($L100$, $L38$, $L43$, $L46$) from 4 locations, which is commonly used in the DG setting. Like PACS, the Terra Incognita experiment was performed using the leave-one-out method as in Table 2.

VLCS The VLCS benchmark is also a domain generalization benchmark. Similar to PACS, VLCS comprises samples from 4 datasets (VOC2007, LabelMe, Caltech-101, and SUN) across 5 overlapping classes. The VLCS experiment also adopts the leave-one-out method for evaluation, as displayed in Table 3.

C.2 MODEL ARCHITECTURE

Here, we report the architectural details of the models used in our experiments.

Model (M) In our experiments, we used the ResNet-50 model as our training model M, which is a standard model architecture in the domain generalization literature. In specific, we adopted the model and its pre-trained weights provided in the torchvision library (Falbel, 2023). Model M is pre-trained on the Imagenet (Russakovsky et al., 2014) dataset.

Anchor Model (A) The anchor model is a frozen ResNet-50 model with the same architecture as our training model M. This is due to innate limitations of the neuron alignment algorithms that require models to be of the same design for alignment. For our experiments, the anchor is a pre-trained ResNet-50, similar to M but with different backbone initialization. The anchor A is then fine-tuned on the source domains. Notably, the anchor model has no significant differences in performance compared to the baseline model. We also tested using a weight-averaged model (Wortsman et al., 2022) created from 6 models as the anchor. An ablation study on the anchor model on SHERPA is reported in Appendix B.

C.3 TRAINING (FINE-TUNING)

In this section, we present the details of the fine-tuning procedure. In our reported experiments, we train the model M (ERM) and the aligned model $\pi(M)$ (SHERPA) on the source domain for 30 epochs, with a learning rate of 0.001 with the SDG optimizer, its momentum set as 0.9, and a batch size of 32. We run the LP-FT (Linear Probing-Fine-tuning) framework for 5 epochs of linear probing and 15 epochs of fine-tuning (Kumar et al., 2022). In Appendix B, we provided a detailed study of the hyperparameters.