

Model Stock: All we need is just a few fine-tuned models

Dong-Hwan Jang^{1,2*} Sangdoo Yun^{1†} Dongyoon Han^{1†}

¹ NAVER AI Lab ² Samsung Advanced Institute of Technology (SAIT)
† corresponding authors

Abstract. This paper introduces an efficient fine-tuning method for large pre-trained models, offering strong in-distribution (ID) and out-of-distribution (OOD) performance. Breaking away from traditional practices that need a multitude of fine-tuned models for averaging, our approach employs significantly fewer models to achieve final weights yet yield superior accuracy. Drawing from key insights in the weight space of fine-tuned weights, we uncover a strong link between the performance and proximity to the center of weight space. Based on this, we introduce a method that approximates a center-close weight using only two fine-tuned models, applicable during or after training. Our innovative layer-wise weight averaging technique surpasses state-of-the-art model methods such as Model Soup, utilizing only two fine-tuned models. This strategy can be aptly coined *Model Stock*, highlighting its reliance on selecting a minimal number of models to draw a more optimized-averaged model. We demonstrate the efficacy of Model Stock with fine-tuned models based upon pre-trained CLIP architectures, achieving remarkable performance on both ID and OOD tasks on the standard benchmarks, all while barely bringing extra computational demands. Our code and pre-trained models are available at <https://github.com/naver-ai/model-stock>.

1 Introduction

Pre-train/fine-tune paradigm [18, 22, 37, 40, 41] has proven to be a strong framework for training models to reach state-of-the-art performance. This approach, especially pivotal in fine-tuning pre-trained models, involves models acquiring general knowledge during pre-training and task-specific knowledge during fine-tuning. How we perform a fine-tuning stage is crucial, affecting task performance and robustness against distribution shifts.

Recent advancements, notably Model Soup [40], which merges weights from multiple fine-tuned models trained under different training setups, have shown impressive performance without increasing inference costs. This method is believed to be effective because these models often reside in the same loss basin, and their merging results in a lower and flat loss basin. However, Model Soup’s requirement for multiple fine-tuned models (more than dozens) raises concerns

* Work done during an internship at NAVER AI Lab.

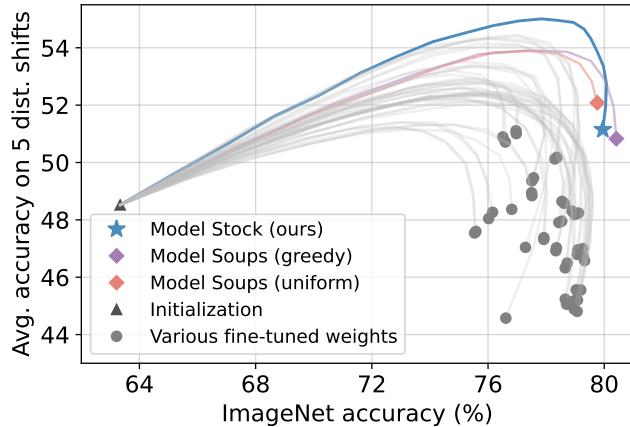


Fig. 1: Model Stock vs. Model Soup. Model Stock consistently enjoys improved accuracy on ImageNet (x-axis) and distribution shift benchmarks (y-axis) against individual fine-tuned models (gray circles). We plot WiSE-FT [41] curves to each fine-tuned model, highlighting Model Stock’s better performance on distribution shifts compared to Model Soup [40]. Note that Model Stock has much smaller computational costs than Model Soups ($24\times$ smaller), i.e., Model Stock requires two fine-tuning procedures, whereas Model Soups are leveraging 48 various fine-tuned models in this experiment.

about efficiency and practicality in general scenarios where the models need to be prepared from scratch. Thus, our question is: Is there an efficient way to achieve an effective merged weight from very few fine-tuned models?

We initially explore the dynamics of fine-tuned weights under the simplest scenario: varying the random seeds while maintaining the other training setups. It reveals that the fine-tuned weights with different random seeds reside on a very thin shell layer-wise during and after training. We then delve into the impact of a model-soup-like weight averaging approach. Our findings show that the closer proximity of the averaged weights correlates with improved In-Distribution (ID) and Out-Of-Distribution (OOD) performance.

Building upon the findings, we propose a novel approach of fine-tuning method coined **Model Stock**, analogous to chicken stock in cooking, distinguishing it from what Model Soup intended. Now, the answer to our question is indeed affirmative: Model Stock approximates the merged weight using just a few fine-tuned models, leveraging the weight space’s geometric properties and a pre-trained model’s anchoring effect. This strategy offers a more computationally efficient alternative to the labor-intensive averaging of fine-tuned models, streamlining the process while enhancing model performance. Fig. 1 illustrates our brief comparison of Model Stock vs. Model Soup [40]. We reproduce the Model Soup experiments¹ based on the CLIP ViT-B/32 initialization by fine-tuning 48 models with various hyper-parameters, which is defined as *zero-shot* initialization

¹ We follow the standard grid hyper-parameter sweep [40], fine-tuning for 10 epochs. Our results align with those presented in Fig. D.1 (right) of the original paper [40].

setting. Fig. 1 shows that Model Stock outperforms Model Soups with much smaller computational costs.

Our comprehensive experiments demonstrate the effectiveness of Model Stock. We achieve performance comparable to, or even surpassing, that of the more resource-intensive methods such as Model Soup [40], using only a fraction of the models. Specifically, our method achieves 87.8% ImageNet top-1 accuracy (ID) and averaged 74.9% in five distribution shift benchmarks (OOD) on ViT-L/14 fairly compared with the prior arts using the CLIP pre-trained weight. We believe that our study not only underscores Model Stock’s practicability but also opens new directions in the pre-train/fine-tune paradigm for superior performance across various tasks.

2 Analyzing Fine-tuned Weights

Our study is driven by two fundamental findings related to the performance and robustness of fine-tuned models. The first one is that model weights are fine-tuned on different random seeds² lie on *thin shell* in weight space layer-wise. The second posits that closer proximity to the center of this thin shell is beneficial for improving performance across the ImageNet and distribution shift benchmarks. The substantiation and implications of these observations are discussed in the subsequent sections.

2.1 Geometric Properties Between Weights

Angle and norm of weights. We begin by examining the intrinsic properties of the weights in fine-tuned models. We define the weight vector of the fine-tuned model at k -th layer as $\mathbf{w}^{(k)} \in \mathbb{R}^{n^{(k)}}$ where $n^{(k)}$ is the number of weight parameters at k -th layer, and the origin $\mathbf{0}$ as the pre-trained model weight $\mathbf{w}_0^{(k)}$ at k -th layer. Then the angle $\theta^{(k)}$ between two weight $\mathbf{w}_1^{(k)}$ and $\mathbf{w}_2^{(k)}$ is defined by: $\theta^{(k)} = \arccos \left(\frac{\mathbf{w}_1^{(k)} \cdot \mathbf{w}_2^{(k)}}{\|\mathbf{w}_1^{(k)}\| \|\mathbf{w}_2^{(k)}\|} \right)$, where the Euclidean l_2 -norm $\|\mathbf{w}^{(k)}\|$ of a $n^{(k)}$ defined as $\|\mathbf{w}^{(k)}\| = \sqrt{\sum_{i=1}^{n^{(k)}} w_i^{(k)2}}$. Angle and norm will provide a geometric view of the weights at k -th layer between fine-tuned models.

Observation 1: Angle and norm consistency among fine-tuned weights. We investigate the weight space of models fine-tuned on ImageNet from a pre-trained model with various random seeds. Our first observation is that both angle $\theta^{(k)}$ between two different models and norm $\|\mathbf{w}^{(k)}\|$ of a weight exhibit *consistent* values with very low standard deviations, as shown in Fig. 2. This consistency can be mathematically represented as follows: For all i and $j \in [1, N]$ when the number of fine-tuned weights, N , is sufficiently large, the following holds:

$$\mathbf{w}_i^{(k)} \cdot \mathbf{w}_j^{(k)} = \begin{cases} (l^{(k)})^2 & \text{if } i = j, \\ (l^{(k)})^2 \cos \theta^{(k)} & \text{otherwise,} \end{cases} \quad (1)$$

² Random seed influences training randomness, such as training data shuffling and data augmentation parameters.

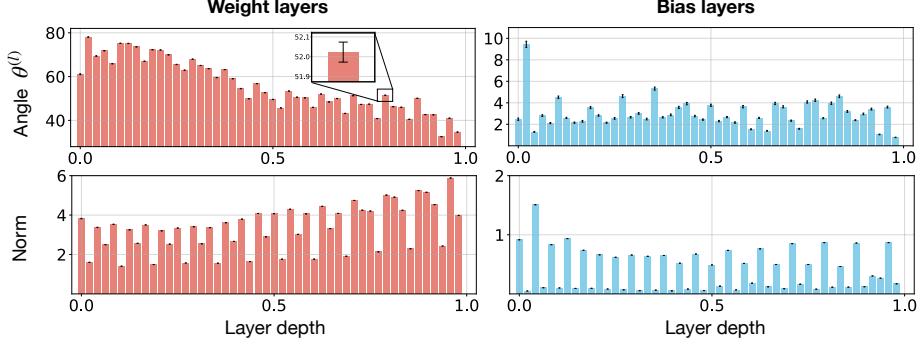


Fig. 2: Layer-wise angle and norm of fine-tuned models. We measure the angle $\theta^{(k)}$ (degree) and norm $\frac{\|\mathbf{w}^{(k)}\|}{\sqrt{n^{(k)}}}$ for 50 distinct weights, fine-tuned under different random seeds. We separately visualize weight layers (red bars) and bias layers (blue bars), where bias layers have much smaller angles. We plot the mean angle and norm values with standard deviation (black error bar). The results show that any two fine-tuned weights have *layer-wise consistent* angle and norm with extremely low standard deviation.

where $l^{(k)}$ and $\theta^{(k)}$ are constants that describe the magnitude and angle between weights at k -th layer, respectively. Henceforth, to simplify notation, we will omit the superscript (k) indicating the layer index.

Interestingly, these consistencies in angle and norm are observed 1) across diverse setups and 2) both during and after training. Fig. 2 shows this consistency over 50 fine-tuned CLIP ViT-B/32³ [30] models. It illustrates that the *layer-wise* norm and angle of these models exhibit almost constant values with extremely minimal error. While this figure depicts a specific model (*i.e.*, CLIP ViT-B/32), we establish that such regularity is not confined to a single model or setting but is consistent across various CLIP fine-tuning scenarios. We conjecture this holds irrespective of networks (ViT [5], Hybrid-ViT [5], ResNet [9], ConvNext [24]), optimizers (SGD, AdamW [25]), data augmentations (RRC [35], RandAug [4]), datasets (CIFAR [17], ImageNet [34]), or initialization of the classifier (zero-shot, LP-FT [18]). Remarkably, this regularity also holds for fine-tuned weights at each step during training as well as after training. A comprehensive analysis supporting these findings is presented in the Appendix A.

Based on the observation, we presume the distribution of the fine-tuned weights. The center of the fine-tuned weights is defined as $\boldsymbol{\mu} = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=1}^N \mathbf{w}_i$. We then deduce the following properties among fine-tuned weights: (i) $\|\mathbf{w}_i - \boldsymbol{\mu}\| = \text{constant}$, indicating a *thin shell* distribution; (ii) $(\mathbf{w}_0 - \boldsymbol{\mu}) \perp (\mathbf{w}_i - \boldsymbol{\mu})$; and (iii) $(\mathbf{w}_i - \boldsymbol{\mu}) \perp (\mathbf{w}_j - \boldsymbol{\mu})$ for all $i, j \in [1, N]$. These properties are depicted in Fig. 5 for better understanding. The detailed proof is in the Appendix B.

³ <https://github.com/openai/CLIP>

Table 1: Distance from the center (i.e., $\|\mathbf{w} - \mu\|$) vs. performance. We report the ImageNet and distribution shift performance with the distance from the center of weights μ for fine-tuned and averaged models. We observe 1) both models consistently maintain a nearly constant distance from μ with remarkably small standard deviation; 2) averaging more models approaches μ , boosting ID and OOD performance.

	$\ \mathbf{w} - \mu\ $	ImageNet	Avg. shifts
Fine-tuned	13.133±.004	79.72	46.37
$\mathbf{w}_{avr}^{(2)}$	9.192±.003	80.24	47.76
$\mathbf{w}_{avr}^{(3)}$	7.439±.025	80.35	48.18
$\mathbf{w}_{avr}^{(5)}$	5.633±.014	80.47	48.53
$\mathbf{w}_{avr}^{(50)} \simeq \mu$	~0	80.59	48.85

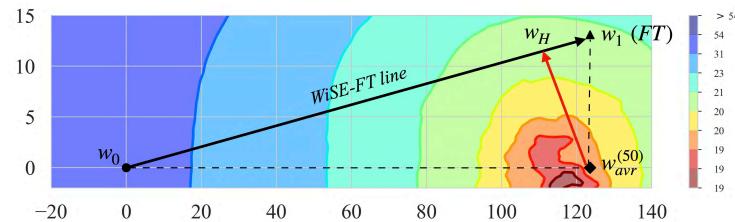


Fig. 3: Test error landscape. We visualize a test error landscape with three weight points: a pre-trained model (\mathbf{w}_0), a fine-tuned model (\mathbf{w}_1), and the averaged weights of 50 fine-tuned models ($\mathbf{w}_{avr}^{(50)}$). \mathbf{w}_{avr} locates near the lowest error basin. A better solution (\mathbf{w}_H) with lower error than \mathbf{w}_1 can be easily found utilizing those three weights (§3).

2.2 Center of Weights and Performance

We proceed to explore the relationship between the proximity to the center of fine-tuned weights and their performance on ID and OOD datasets. Given that computing the exact center is infeasible, we approximate it by averaging differently seeded 50 fine-tuned weights, using it as a pseudo-center (i.e. $\mu \simeq \mathbf{w}_{avr}^{(50)}$).

Observation 2: Distance from the center of weights and performance. Table 1 offers quantitative observations about the fine-tuned weights and their performance using CLIP ViT-B/32. The results include distances from the weight center (μ) of fine-tuned models and their averaged counterparts with their ID and OOD performances. Going closer to the center by averaging the weights leads to improving both performances. Interestingly, the standard deviation of the distances is less than 0.1% of the mean distance, suggesting highly consistent fine-tuned weight distances from the center across different weights. This suggests that fine-tuned weights occupy a *thin shell* as discussed in §2.1.

Observation 3: Fine-tuned weights occupy local minima edges in the test error landscape. We present an additional observation regarding the test error landscape, which relates to the performance around the center of weights. Fig. 3 depicts the test error landscape on the ImageNet test dataset within a two-dimensional plane. This plane includes a pre-trained model’s weight (\mathbf{w}_0), a single fine-tuned model (\mathbf{w}_1), and the pseudo-center ($\mathbf{w}_{avr}^{(50)}$). This landscape

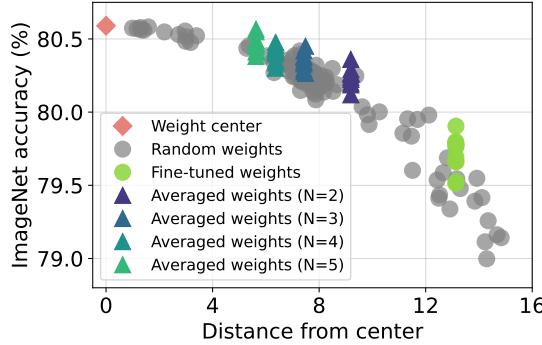


Fig. 4: Weights closer to the center enjoy higher performances. We show the ImageNet accuracy of weights with their distance from the center (μ), which is approximated with $\mathbf{w}_{\text{avr}}^{(50)}$. Averaged weights are closer to the center than individual weights, and their accuracy increases as the averaging number of models (N) increases. Gray circles are the weights randomly sampled from the Gaussian distribution centered at $\mathbf{w}_{\text{avr}}^{(50)}$. Even the random weights also achieve higher performance as they reach the center. The results indicate the critical role of proximity to the center on performance.

reveals that a fine-tuned model typically occupies the boundary of test error regions. On the other hand, centered near pseudo-center ($\mathbf{w}_{\text{avr}}^{(50)}$), the test error is the lowest and gets higher as the weights get far from the center. Interestingly, along the line segment $\overline{\mathbf{w}_0 \mathbf{w}_1}$ (*i.e.*, a WiSE-FT curve [41]), the fine-tuned weight \mathbf{w}_1 is neither the point closest to the pseudo-center nor the one with the lowest test error. We will connect this observation in §3 to find \mathbf{w}_H , the weight on the line closest to the center.

Observation 4: Randomly perturbed weights nearing the center also merit high performance. To further investigate the impact of proximity to the center on performance, we conduct a toy experiment to measure ImageNet performance using random weights generated by adding layer-wise noise to the weight at the center. The standard deviation of the noise for each layer is adjusted to align with the distribution of fine-tuned weights. Fig. 4 presents a scatter plot correlating the distance of model weights from the distribution’s center with corresponding ImageNet accuracies. Remarkably, randomly sampled weights demonstrate performance comparable to fine-tuned and averaged models, highlighting the importance of center proximity.

Finally, the above observations naturally raise a question: Why do fine-tuned weights through optimization not reach the center, staying constantly close instead? Previous studies [3, 15] tell us that optimization steps might struggle to guide fine-tuned weights to the center of the weight distribution due to the many stationary points in the loss surface. Alternatively, averaging independently fine-tuned models is a unique solution but both laborious and resource-intensive. It appears that there are no better alternatives for getting closer to the center, as

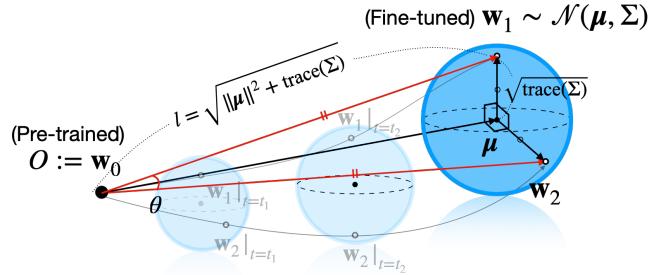


Fig. 5: Comprehensive illustration of the geometric dynamics of fine-tuned weights. This figure illustrates the behavior of fine-tuned weights that lie on a thin shell, supporting our Gaussian distribution hypothesis. Each sphere represents a thin shell that fine-tuned models lie on at each training step, with \mathbf{w}_0 denoting the pre-trained model. The curved lines trace the fine-tuning trajectory from \mathbf{w}_0 , while the red vectors indicate that fine-tuned models at each time step are equidistant from \mathbf{w}_0 .

optimization proves ineffective near these flat local minima. Could there be a faster approach to reaching the center? This question will be addressed in §3.

2.3 Our Hypothesis

The observed geometric patterns in weight distributions closely align with mathematical properties of Gaussian distributions, represented as $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Therefore, a plausible reason for the particular geometric pattern of fine-tuned weights could be the influence of Gaussian noise within the weight space. In high-dimensional spaces, vectors sampled from such distributions tend to have nearly identical norms, specifically $\sqrt{|\boldsymbol{\mu}|^2 + \text{trace}(\boldsymbol{\Sigma})}$ and consistent in-between angles, due to the *concentration of measure phenomenon* [19]. The likelihood of the squared norm significantly deviating from this expected value is exponentially negligible in high-dimensional spaces, like a weight space.

In other words, the vectors sampled from high-dimensional Gaussian distribution lie on a very thin shell with the radius $\approx \sqrt{\text{trace}(\boldsymbol{\Sigma})}$ around the center $\boldsymbol{\mu}$. Consequently, we hypothesize that fine-tuned weights follow a Gaussian distribution in a layer-wise manner. While not necessary for our observation, this hypothesis provides a sufficient condition for understanding the geometric dynamics of fine-tuned weights. For example, it aids in the intuitive understanding of the fact that the distance of $\mathbf{w}_{\text{avr}}^{(N)}$ from the weight center is proportional to $1/\sqrt{N}$, signifying the reduction of variance. Fig. 5 comprehensively illustrates the observations and our hypothesis discussed in §2.

3 Method

This section introduces our method, Model Stock, a cost-efficient weight merging method. As discussed in §2.2, getting closer to the center of weights $\boldsymbol{\mu}$ induces improved model performance. A straightforward method to approximate

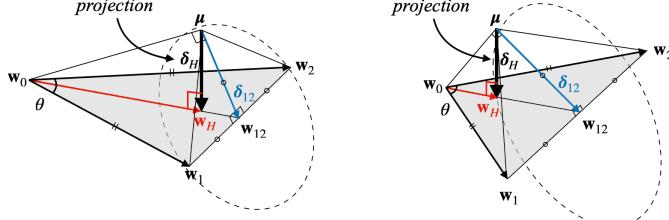


Fig. 6: Schematic concept of our Model Stock. We present two scenarios with a small angle (left) and a large angle (right). Given a pre-trained weight (\mathbf{w}_0) and two fine-tuned weights (\mathbf{w}_1 , and \mathbf{w}_2), we visualize a gray triangle representing the span of three weights. We consider this triangle area our search space spanned by three weights. We aim to find the best weight point on the triangle nearest the ideal center μ . We find that the perpendicular foot \mathbf{w}_H from point μ to the plane is the nearest point, which can be specified solely using the angle between the fine-tuned models, even without knowing the exact position of the center μ . We utilize \mathbf{w}_H as the merged weight of Model Stock. Intuitively, when the angle θ is large (*i.e.*, two weights are diverse), as in the right figure, \mathbf{w}_H will rely more on the pre-trained weight(\mathbf{w}_0), vice versa.

μ is averaging multiple model weights, which can be computationally expensive. We propose an efficient alternative method by leveraging *the pre-trained model weights*, an aspect previously neglected by existing weight-merging methods. A pre-trained model usually possesses general knowledge and shows robust and reliable performance in out-of-distribution (OOD) cases [30]. Therefore, a pre-trained model can become a *robust anchor point*. As shown in Fig. 3, we could readily identify a weight (\mathbf{w}_H) that is closer to the center—and thus better—by interpolating with the anchor. Building on this concept, we propose a method to approximate the center of weights more accurately with only a few fine-tuned weights. Again, for readability, we omit the layer notation (k), but the subsequent method is applied layer-wise.

On two fine-tuned models. We observed in §2 that two fine-tuned models with different random seeds have almost constant norms and the angle between them. Based on this, we define a plane connecting the pre-trained model and two fine-tuned models as shown in Fig. 6 (depicted as a gray triangular area). Any weight vector on this plane can be expressed as a linear combination of the pre-trained model and two fine-tuned models. Our goal is to find the weight closest to the center of fine-tuned weights on this plane, which is the perpendicular foot (\mathbf{w}_H) from the center of distribution (μ) to the plane.

Even without knowing the exact position of the center μ , \mathbf{w}_H can be specified solely using the angle between the fine-tuned models with the following two conditions that μ must satisfy. First, as mentioned in §2.1, $(\mathbf{w}_1 - \mu) \perp (\mathbf{w}_2 - \mu)$ and $\|\mathbf{w}_i - \mu\| = \|\mathbf{w}_2 - \mu\|$ hold, implying that $\Delta \mathbf{w}_1 \mu \mathbf{w}_2$ forms an isosceles right triangle. In Fig. 6, μ should lie on the dotted hyper-circle. Another condition is that the condition $(\mathbf{w}_0 - \mu) \perp (\mathbf{w}_{12} - \mu)$ must be satisfied, where $\mathbf{w}_{12} = \frac{\mathbf{w}_1 + \mathbf{w}_2}{2}$. This condition arises from the second property in §2.1, where $(\mathbf{w}_0 - \mu) \perp (\mathbf{w}_i - \mu)$. Combining these two conditions, μ is at the point where the line starting from \mathbf{w}_0

is tangent to the hyper-circle. We precisely determine the position of μ within a 3D volume slice that encompasses both μ and $\Delta\mathbf{w}_0\mathbf{w}_1\mathbf{w}_2$. Consequently, we can find the position of the closest weight \mathbf{w}_H to the distribution's center on the plane using straightforward geometric principles. The position of the perpendicular foot is determined as follows:

$$\mathbf{w}_H = \frac{2 \cos \theta}{1 + \cos \theta} \cdot \mathbf{w}_{12} + \left(1 - \frac{2 \cos \theta}{1 + \cos \theta}\right) \cdot \mathbf{w}_0. \quad (2)$$

For more detailed proof, please refer to the Appendix D. Note that the interpolation ratio $t = \frac{2 \cos \theta}{1 + \cos \theta}$ is solely determined by the angle θ between two fine-tuned models. Crucially, unlike previous methods, determining t does not require extra training [22, 36, 37, 40] or heuristic hyper-parameter settings [6, 40], thereby simplifying the process and enhancing its accessibility and efficiency.

As the θ decreases, the pre-trained model is less utilized for merging, as shown in Fig 6 (left). Coupled with the observation in Fig. 2, this indicates that bias layers rely less on pre-trained models and focus more on fine-tuned models, whereas weight layers depend more on pre-trained models. This observation extends the findings of previous works such as BitFit [43] and LP-FT [18]. In the case of BitFit and LP (*i.e.*, the first step of LP-FT), bias and classifier layers fully utilize fine-tuning, while other weight layers (attention and MLP) rely on pre-trained models. We present an additional analysis in the Appendix F.

On N fine-tuned models. We further extend the previous derivation to N fine-tuned models to move even closer to the weight center. Let us denote $\mathbf{w}_{\text{avr}}^{(N)}$ as the N -averaged weight, $\sum_{i=1}^N \mathbf{w}_i / N$, and $\mathbf{w}_H^{(N)}$ as the weight in the $\text{span}(\mathbf{w}_0, \mathbf{w}_1, \dots, \mathbf{w}_N)$ closest to the distribution's center. Then, we derive the position of $\mathbf{w}_H^{(N)}$ as:

$$\mathbf{w}_H^{(N)} = t \cdot \mathbf{w}_{\text{avr}}^{(N)} + (1 - t) \cdot \mathbf{w}_0, \quad \text{s.t. } t = \frac{N \cos \theta}{1 + (N - 1) \cos \theta}. \quad (3)$$

Detailed proof is in the Appendix D. Similar to the case of two fine-tuned models, the interpolation ratio t depends solely on the angle θ between the pre-trained model and the N fine-tuned models.

Periodic merging. To move one step forward with our method, we propose *periodic merging*, which is performed between the fine-tuned models and the pre-trained model during training. As the geometric properties of weights are also applicable to weights during training (refer to Appendix A.2 for more details), we fully utilize this phenomenon here.

We strategically merge weights at the end of every epoch, allowing for the fine-tuning process to be parallelized with merging. The interpolation ratio for merging is determined by the angle θ between the pre-trained model and the fine-tuned models at the current epoch. Fig. 7 visualizes this periodic merging process. We argue that employing the periodic merging can approximate the

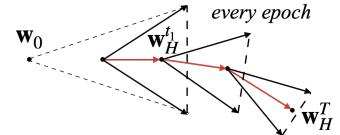


Fig. 7: Periodic merging for Model Stock. Weights are merged every epoch.

center of weights more accurately. In §4.3, we empirically show that the periodic merging yields superior performance and achieves a closer distance to the center.

4 Experiment

We present the key experimental results in this section. We first provide our experimental setups in §4.1. Then, we present the main results in §4.2, and ablation studies in §4.3. More detailed experimental setups, analysis, and further results are in the Appendix G. We will release our codes and weights publicly.

4.1 Experimental Setup

Models. We conduct the experiments on CLIP ViT-B/32, CLIP ViT-B/16, and CLIP ViT-L/14 models. We set the number of fine-tuning models for Model Stock as two. We compare Model Stock against various fine-tuning techniques, including Model Soups [40], LP-FT [18], CAR-FT [27], FTP [37], and FLYP [7]. We use CLIP ViT-B/32 for ablation studies.

Datasets. We fine-tune models on the ImageNet-1K [34] training dataset. We report the ImageNet-1K [34] top-1 accuracy for evaluating in-distribution (ID) performance. For distribution shift scenarios, we consider five out-of-distribution (OOD) benchmarks including ImageNet-V2 [33], ImageNet-R [10], ImageNet-Sketch [39], ImageNet-A [11], and ObjectNet [1]. Since previous methods, except Model Soups [40], have not been evaluated on ObjectNet, we omit the ObjectNet results when comparing against them.

Training setup. We initialize the classifier weights (*e.g.*, 1000 classes for ImageNet) using the text encoder of CLIP and text prompts following Model Soup [40]. We use AdamW [25] with batch size 512 and weight decay 0.1 for all the experiments, including vanilla fine-tuning, Model Soup reproduction, and Model Stock⁴. We employ two training setups for comparisons: (1) training 10 epochs with minimal data augmentation, following Model Soup’s zero-shot initialization setup, and (2) training 16 epochs with strong data augmentation, following one of Model Soup’s LP initialization setup [40], denoting with *. These setups enable a balanced comparison of Model Stock against zero-shot and LP-initialized Model Soups. Detailed hyper-parameters are in the Appendix G.

4.2 Main Results

CLIP ViT-B/32. Table 2 shows the results of Model Stock on the pre-trained CLIP ViT-B/32 model by comparing it with Model Soups. ‘Avg. shifts’ denotes the average accuracy of the five OOD benchmark scores. Our Model Stock and Model Stock* show competitive performance with Model Soups. Furthermore, Model Stock* achieves state-of-the-art performance on ImageNet with **81.19%** top-1 accuracy. As described in Fig. 1, Model Stock with WiSE-FT [41] enjoys a

⁴ We reduce the batch size to 64 on CLIP ViT-L/14 due to memory limitation.

Table 2: Comparison against Model Soups [40] on CLIP ViT-B/32. We report the performance and relative fine-tuning costs on the CLIP ViT-B/32 scenario. α denotes the cost for LP initialization. Model Stock shows comparable performance with Model Soups with significantly reduced training costs.

Method	ImageNet	Avg. shifts	Cost
<i>Comparing with Model Soups from zero-shot init.</i>			
CLIP zero-shot Initialization	63.34	48.51	0
Vanilla FT	78.35	47.03	1
Uniform Model Soup (from zero-shot)	79.76	52.08	48
Greedy Model Soup (from zero-shot)	80.42	50.83	48
Model Stock	<u>79.89</u>	<u>50.99</u>	2
<i>Comparing with Model Soups from LP init.</i>			
CLIP LP initialization	75.57	47.21	α
Vanilla FT*	79.72	46.37	1
Uniform Model Soup (from LP init)	79.97	51.45	71+ α
Greedy Model Soup (from LP init)	81.03	50.75	71+ α
Model Stock*	81.19	48.69	2

Table 3: Model Stock on CLIP ViT-B/16. Model Stock shows competitive performance against previous fine-tuning methods on ImageNet and distribution shifts.

Method	ImageNet	Distribution shifts				
		Avg. shifts	IN-V2	IN-R	IN-A	IN-Sketch
Zero-shot	68.3	59.5	62.0	77.7	<u>49.9</u>	48.3
Vanilla FT	82.8	57.7	72.9	66.4	43.7	48.0
Vanilla FT*	83.7	57.4	73.5	67.6	40.0	48.6
LP [18]	79.7	48.1	71.5	52.4	27.8	40.5
LP-FT [18]	81.7	<u>60.5</u>	71.6	<u>72.9</u>	49.1	48.4
CAR-FT [27]	83.2	59.4	73.0	71.3	43.7	49.5
FTP [37]	<u>84.2</u>	49.7	74.6	47.2	26.5	50.2
FLYP [7]	82.6	<u>60.5</u>	73.0	71.4	48.1	49.6
Model Stock	84.1	62.4	<u>74.8</u>	71.8	51.2	51.8
Model Stock*	85.2	60.1	75.3	68.7	45.0	<u>51.3</u>

superior ID-OOD performance curve compared to Model Soup and its WiSE-FT curves. Note that Model Soups require dozens of fine-tuned models (*e.g.*, zero-shot and LP-init Model Soups use 48 and 71 models, respectively), highlighting the effectiveness of Model Stock along with efficiency utilizing only two models. We provide further comparison results with WiSE-FT [41] curves on LP-init Model Soups in the Appendix H.

CLIP ViT-B/16. Table 3 presents a comprehensive comparison of different fine-tuning methods applied to CLIP ViT-B/16. Previous works [18, 36, 37] lack ObjectNet [1] results; therefore, we omit ObjectNet and report the other four OOD benchmarks. Complete results with ObjectNet and ImageNet-Real [2] are in the Appendix H.2. The results show Model Stock exhibits exceptional performance on the ImageNet accuracy, *e.g.*, Model Stock* achieves 85.2% top-1

Table 5: Impact of the number of fine-tuning models (N) on Model Stock. IN denote ImageNet accuracy.

	IN	Avg. Shifts	$\ \mathbf{w} - \boldsymbol{\mu}\ $
FT	79.7	46.7	13.13
$N=2$	80.1	48.8	10.01
$N=3$	80.2	48.8	9.05
$N=4$	80.4	48.9	8.45

Table 6: Impact of merging period on Model Stock. IN denotes ImageNet accuracy.

Period	IN	Avg. Shifts
1000 iters	79.8	48.7
5000 iters	79.9	48.5
1 epoch	80.1	48.8

accuracy on ImageNet, which is a state-of-the-art level. Model Stock also shows robust performance across diverse distribution shift scenarios.

CLIP ViT-L/14. Table 4 shows the results of Model Stock on the CLIP ViT-L/14 model. The results show that Model Stock can push the limit of benchmark scores with large-size backbone architecture. We remark that Model Stock* achieves state-of-the-art performance with 87.7% ImageNet top-1 accuracy, implying that Model Stock is still effective in a scale-up scenario. The results consistently demonstrate the high efficacy and robustness of Model Stock across diverse scales of models and various benchmark scenarios, reaffirming its potential in practical applications.

Table 4: Model Stock on CLIP ViT-L/14.

	IN	Avg. shifts
Zero-shot	75.0	63.0
Vanilla FT	85.8	66.8
Vanilla FT*	87.1	68.0
TPGM [36]	87.0	69.4
CAR-FT [27]	87.1	67.8
Model Stock	87.0	71.6
Model Stock*	87.7	73.5

4.3 Ablation studies and analysis of Model Stock

We conduct ablation studies on CLIP ViT-B/32. We train vanilla fine-tuned models and Model Stock for 16 and 8 epochs, respectively; thus, the training cost of Model Stock matches with a single fine-tuning process.

Experiments on the number of fine-tuned model N . Table 5 shows the effect of the number of fine-tuned models. The results show that Model Stock obtains enhanced performance and closer distance from the (pseudo-) center ($\|\mathbf{w} - \boldsymbol{\mu}\|$) as the number of merging models increases. Considering the trade-off between the performance and training cost induced by increased N , our setting ($N=2$) shows the best for Model Stock.

Study on the merging period of Model Stock. Table 6 shows the results of various merging periods, including 1000 and 5000 iterations settings. Note that 1 epoch is ~ 2500 iterations in our experiment. Model Stock shows consistent performance with various periods.

The post-training merging strategy of Model Stock. We study an alternative of Model Stock that merges fine-tuned weights only once after each fine-tuning process is finished, similar to Model Soups [40]. We denote it as Model

Table 7: Post-training merging strategy of Model Stock. We present ImageNet accuracy, distribution shifts, and distance from the center with the results of uniform averaging, a straightforward baseline.

	Uniform averaging ($\mathbf{w}_{\text{avg}}^N$)			Model Stock (post-training)		
	ImageNet	Avg. Shifts	$\ \mathbf{w} - \mu\ $	ImageNet	Avg. Shifts	$\ \mathbf{w} - \mu\ $
$N=2$	80.2	47.8	9.19	80.3(+0.1)	50.4(+2.6)	7.62(-1.57)
$N=3$	80.4	48.2	7.44	80.4(+0.0)	50.2(+2.0)	6.49(-0.95)
$N=4$	80.5	48.5	5.63	80.5(+0.0)	49.8(+1.4)	5.16(-0.47)

Stock (post-training). We utilize the individually fine-tuned weights as we conducted in §2, using the same training settings with different random seeds for each model. We report the performance and distance from the pseudo-center of Model Stock (post-training) in Table 7. On the left side of the table, we provide the performance of its counterpart, a uniform averaging of N models ($\mathbf{w}_{\text{avg}}^N$). The improvements from the uniform averaging to Model Stock (post-training) are denoted in the table’s parentheses. The results show that Model Stock (post-training) archives improved distribution shift scores with closer distances toward the center than its counterpart.

5 Related Work

We discuss related works and highlight how our method differs and contributes to the existing works.

Model Soups [40] is a straightforward weight averaging method that merges weights from various fine-tuned models trained with different hyper-parameters. It demonstrates improved in-distribution (ID) and out-of-distribution (OOD) performance. While effective, model soups typically require a large number of fine-tuned models. Our method aims to achieve similar or superior performance improvements more efficiently, utilizing significantly fewer fine-tuning costs. We provide further discussion about Model Soups with our new interpretation in Appendix E.

Robust Fine-tuning. When fine-tuning generalist models like CLIP, we often observe the fine-tuned models lose the generalization ability of the original CLIP model, with decreased OOD performance. To address this issue, several robust fine-tuning approaches have been proposed. LP-FT [18] attempts to preserve pre-trained weights by initially training only a linear probing layer. WiSE-FT [41] improves OOD performance through linear interpolation between fine-tuned and pre-trained weights. While our method shares similarities with WiSE-FT in using pre-trained weights, our method determines interpolation ratios layer-wise based on geometric properties. Focusing on OOD performance, methods suggesting

improved training objectives [7, 27–29, 36, 37] has been proposed. Our approach differs from these methods as we do not propose a new fine-tuning loss. Instead, we perform two fine-tunings and achieve robust performance by merging them.

Weight Center and Flat minima Recent machine learning research has extensively explored the significance of finding flat minima for improved generalization [3, 15, 21, 26]. Keskar *et al.* [16] and Hochreiter & Schmidhuber [12] demonstrated that sharp optima by large batch SGD have steep, harmful directions, while broader optima enhance generalization. Stochastic Weight Averaging (SWA) [15] targets the center of flat minima, enhancing robustness against shifts in the loss landscape between training and test datasets. SWAG [26] builds on SWA by incorporating Bayesian model averaging with a Gaussian posterior to further boost performance. SWAD [3] found that the generalization gap between flat and sharp minima is more pronounced in OOD scenarios than in ID ones. Our method theoretically extends these approaches by efficiently identifying the center of flat minima with novel geometric properties, leading to significant improvements in both ID and OOD performances.

Model Weight Merging. Recent research has explored merging models fine-tuned on various tasks. Methods such as Task Arithmetic [14] and TIES [42] have been proposed. They are also based on the difference between fine-tuned and pre-trained weights (often referred to as the “task vector”). However, our method distinguishes itself through geometric analysis for weight merging. In the domain of Large Language Models (LLMs), merging techniques like WARM [31] and WARP [32] have emerged. Our method has the potential for extension to these areas, offering new avenues for future research.

6 Conclusion

Our study illuminated the fine-tuning process in machine learning, revealing that fine-tuned models’ weights generally exhibit the properties of a Gaussian distribution. The proximity of these models to the center of weights was crucial for improved performance in target domains like ImageNet and under diverse distribution shifts. Utilizing a pre-trained model as a robust anchor point, we efficiently minimized the variance with fewer fine-tuned models, eliminating the need for additional training to find the optimal interpolation ratio.

Additionally, our findings suggested further knowledge and applicability to the models near flat minima and will offer new insights on model weight merging methods. As the pretraining-finetuning paradigm gains more prominence, our insights will provide a foundation for better understanding and optimizing the fine-tuning process in both academia and industry.

Limitation. Due to resource limitations, we could not conduct larger-scale models such as ViT-G. Exploring this will be part of our future work.

References

1. Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., Katz, B.: Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. NeurIPS (2019)
2. Beyer, L., Hénaff, O.J., Kolesnikov, A., Zhai, X., Oord, A.v.d.: Are we done with imagenet? arXiv preprint arXiv:2006.07159 (2020)
3. Cha, J., Chun, S., Lee, K., Cho, H.C., Park, S., Lee, Y., Park, S.: Swad: Domain generalization by seeking flat minima. NeurIPS **34**, 22405–22418 (2021)
4. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: CVPRW. pp. 702–703 (2020)
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021)
6. Gouk, H., Hospedales, T.M., Pontil, M.: Distance-based regularisation of deep networks for fine-tuning. arXiv preprint arXiv:2002.08253 (2020)
7. Goyal, S., Kumar, A., Garg, S., Kolter, Z., Raghunathan, A.: Finetune like you pretrain: Improved finetuning of zero-shot vision models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 19338–19347 (2023)
8. He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., Neubig, G.: Towards a unified view of parameter-efficient transfer learning. arXiv preprint arXiv:2110.04366 (2021)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
10. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: ICCV. pp. 8340–8349 (2021)
11. Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., Song, D.: Natural adversarial examples. In: CVPR. pp. 15262–15271 (2021)
12. Hochreiter, S., Schmidhuber, J.: Flat minima. Neural computation **9**(1), 1–42 (1997)
13. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)
14. Ilharco, G., Ribeiro, M.T., Wortsman, M., Schmidt, L., Hajishirzi, H., Farhadi, A.: Editing models with task arithmetic. In: The Eleventh International Conference on Learning Representations (2023), <https://openreview.net/forum?id=6t0Kwf8-jrj>
15. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv:1803.05407 (2018)
16. Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: Generalization gap and sharp minima. In: International Conference on Learning Representations (2017), <https://openreview.net/forum?id=H1oyR1Ygg>
17. Krizhevsky, A.: Learning multiple layers of features from tiny images. In: Tech Report (2009)
18. Kumar, A., Raghunathan, A., Jones, R., Ma, T., Liang, P.: Fine-tuning can distort pretrained features and underperform out-of-distribution. arXiv preprint arXiv:2202.10054 (2022)

19. Ledoux, M.: The concentration of measure phenomenon. No. 89, American Mathematical Soc. (2001)
20. Lee, Y., Chen, A.S., Tajwar, F., Kumar, A., Yao, H., Liang, P., Finn, C.: Surgical fine-tuning improves adaptation to distribution shifts. In: ICLR (2022)
21. Li, H., Xu, Z., Taylor, G., Studer, C., Goldstein, T.: Visualizing the loss landscape of neural nets. NeurIPS **31** (2018)
22. Li, T., Huang, Z., Tao, Q., Wu, Y., Huang, X.: Trainable weight averaging: Efficient training by optimizing historical solutions. In: ICLR (2022)
23. Lian, D., Zhou, D., Feng, J., Wang, X.: Scaling & shifting your features: A new baseline for efficient model tuning. NeurIPS **35**, 109–123 (2022)
24. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: CVPR. pp. 11976–11986 (2022)
25. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
26. Maddox, W.J., Izmailov, P., Garipov, T., Vetrov, D.P., Wilson, A.G.: A simple baseline for bayesian uncertainty in deep learning. NeurIPS **32** (2019)
27. Mao, X., Chen, Y., Jia, X., Zhang, R., Xue, H., Li, Z.: Context-aware robust fine-tuning. IJCV (12 2023). <https://doi.org/10.1007/s11263-023-01951-2>
28. Nam, G., Heo, B., Lee, J.: Lipsum-ft: Robust fine-tuning of zero-shot models using random text guidance. In: The Twelfth International Conference on Learning Representations (2024)
29. Oh, C., Lim, H., Kim, M., Han, D., Yun, S., Choo, J., Hauptmann, A., Cheng, Z.Q., Song, K.: Towards calibrated robust fine-tuning of vision-language models (2024), <https://arxiv.org/abs/2311.01723>
30. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: ICML (2021)
31. Rame, A., Vieillard, N., Hussenot, L., Dadashi, R., Cideron, G., Bachem, O., Ferret, J.: WARM: On the benefits of weight averaged reward models. In: Forty-first International Conference on Machine Learning (2024), <https://openreview.net/forum?id=s7RDnNUJy6>
32. Ramé, A., Ferret, J., Vieillard, N., Dadashi, R., Hussenot, L., Cedoz, P.L., Sessa, P.G., Girgin, S., Douillard, A., Bachem, O.: Warp: On the benefits of weight averaged rewarded policies (2024), <https://arxiv.org/abs/2406.16768>
33. Recht, B., Roelofs, R., Schmidt, L., Shankar, V.: Do imagenet classifiers generalize to imagenet? In: ICML (2019)
34. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. IJCV **115**(3), 211–252 (2015)
35. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR. pp. 1–9 (2015)
36. Tian, J., He, Z., Dai, X., Ma, C.Y., Liu, Y.C., Kira, Z.: Trainable projected gradient method for robust fine-tuning. In: CVPR. pp. 7836–7845 (2023)
37. Tian, J., Liu, Y.C., Smith, J.S., Kira, Z.: Fast trainable projection for robust fine-tuning. In: NeurIPS (2023)
38. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers & distillation through attention. In: ICML. vol. 139, pp. 10347–10357 (July 2021)
39. Wang, H., Ge, S., Lipton, Z., Xing, E.P.: Learning robust global representations by penalizing local predictive power. In: NeurIPS (2019)

40. Wortsman, M., Ilharco, G., Gadre, S.Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A.S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al.: Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In: ICML. pp. 23965–23998. PMLR (2022)
41. Wortsman, M., Ilharco, G., Kim, J.W., Li, M., Kornblith, S., Roelofs, R., Lopes, R.G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al.: Robust fine-tuning of zero-shot models. In: CVPR. pp. 7959–7971 (2022)
42. Yadav, P., Tam, D., Choshen, L., Raffel, C., Bansal, M.: TIES-merging: Resolving interference when merging models. In: Thirty-seventh Conference on Neural Information Processing Systems (2023), <https://openreview.net/forum?id=xtaX3WyCj1>
43. Zaken, E.B., Ravfogel, S., Goldberg, Y.: Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. arXiv preprint arXiv:2106.10199 (2021)
44. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. ICLR (2018)

Appendix

In this Appendix, we provide in-depth analysis and additional insights to complement the main text of our study on Model Stock, our novel approach to fine-tuning and weight merging. The contents are summarized as follows:

- We examine the angle norm consistency of fine-tuned weights across various settings in §A, extending the observations discussed in §2.1.
- We provide detailed proofs of geometric properties of fine-tuned weights in §B.
- We study the importance of reducing variance for performance in out-of-distribution scenarios in §C, showcasing the test error landscape across various datasets and elaborating on the explanations in §2.2.
- We provide detailed proofs in §D for the optimal interpolation ratio in our method §3.
- We discuss prior studies through the lens of our findings in §E.
- We provide an additional analysis of the interpolation ratio in §F.
- We present experimental settings of §4 in §G.
- We present additional experiments of Model Stock in §H

Each section aims to offer a comprehensive understanding of our method’s underlying principles and its broad applicability in machine learning.

A Angle and Norm Consistency

We argue that, as discussed in §2.1, angles and norms of fine-tuned weights would remain consistent across fine-tuned models, independent of various factors. These factors include architecture type (ViTs [5], ResNet [9], ConvNeXt [24]), optimizers (SGD, AdamW [25]), augmentations (RRC [35], RandAug [4]), datasets (CIFAR [17], ImageNet [34]), or the initialization of the classifier (zero-shot, LP as in LP-FT [18]). We depict the layer-wise angle and norm of 5 fine-tuned weights for each category based on different random seeds. We give detailed illustrations for each setting at the end of the Appendix to enhance readability (refer to Fig. H–O). Across all these settings, the angle and norm of weights exhibit a surprising level of consistency.

A.1 Analysis on layer-wise tendency

The layer-wise angle and norm across various settings are shown in Fig. H–L. We visualize with every weight of attentions/convolutions (Attention/Conv), multi-layer perceptrons (MLP), normalizations (LayerNorm and BatchNorm), a classifier (Classifier), individual bias (Bias), and the remaining layers (*i.e.*, the patchification layer, positional embedding, class embedding, and projection layer). We further display *All* in each figure, which denotes the concatenation of the weights of entire layers.

The layer-wise analysis reveals an interesting trend: Bias and classifier layers demonstrate smaller angles than attention and MLP layers. In other words, bias and classifier layers exhibit lower randomness and more reliable updates than attention and MLP layers. It is important to note that as the angle decreases, the pre-trained model is less utilized for merging (refer to Eq. (2)). This indicates that bias and classifier layers focus more on fine-tuned models and rely less on the pre-trained model, whereas attention and MLP layers depend less on the fine-tuned model (*i.e.*, $t_{\text{bias}}, t_{\text{clf}} > t_{\text{attn}}, t_{\text{mlp}}$). This observation extends the findings of previous works such as BitFit [43] and LP-FT [18]. In the case of BitFit and LP (*i.e.*, the first step of LP-FT), bias and classifier layers fully utilize fine-tuning, while other layers (attention and MLP) rely on pre-trained models.

These traits could offer new insights into parameter-efficient transfer learning (PETL) [8, 13, 23, 43] and layer-wise fine-tuning [18, 20, 36, 37]. Maintaining weights with high randomness (higher angles) while updating on biases and classifier weights with lower randomness and fewer parameters would be an efficient fine-tuning strategy. PETL has been exploring this direction but has not yet provided solid reasons why certain layers are more effective than others. Our analysis suggests that one reason could be the lower randomness (or variance) of these layers, as indicated by the angle trend per layer.

A.2 Maintaining consistency during training

We further argue that the consistency we observed is maintained while training progresses, as illustrated by multiple thin shells in Fig. 5. To demonstrate that the angle and norm of fine-tuned models remain consistent during the entire training process, we plot their relationship across weights for every epoch in Fig. M. Please note that the angle is consistent across differently seeded models at the same timestamp (*i.e.*, $\mathbf{w}_1|_{t=t_1}$ and $\mathbf{w}_2|_{t=t_1}$), not across models at different timestamps (*i.e.*, $\mathbf{w}_1|_{t=t_1}$ and $\mathbf{w}_1|_{t=t_2}$). The observed trend is as follows: as training progresses, the angle between weights steadily decreases. This analysis uses the CLIP ViT-B/32 model fine-tuned on ImageNet-1K with five random seeds.

A.3 Filter-wise analysis of weights

Li *et al.* [21] showed that when evaluating the robustness of a neural network by adding random noise to certain weights, performance analysis based on adding filter-wise noise (*i.e.*, adding noise for each row in all weight matrices) aligns more closely with the generalization performance than adding layer-wise noise does. Inspired by this observation, we investigate the possibility that the weight distribution may follow a filter-wise Gaussian distribution and adapt this concept to our method (see the performance analysis in §H.5). Fig. N illustrates the angle distribution filter-wise. The angle exhibits much larger standard deviations than the layer-wise distribution. This could be attributed to the reduction in dimensionality. As the number of dimensions decreases, it becomes challenging to approximate the norm as a constant value.

A.4 Analysis on non-CLIP models

To verify if this key observation also applies to non-CLIP models, we analyze the geometric patterns of fine-tuned weights trained using the DeiT [38] method (*i.e.*, pre-trained on ImageNet-21K). Fig. O displays the angle and norm of 10 DeiT-base models first pre-trained on ImageNet-21K [34] and then fine-tuned on ImageNet-1K. We find that weights pre-trained with ImageNet-21K also exhibit *consistent angle and norm*, indicating that our observation may be valid beyond CLIP fine-tuning scenarios as well.

B Detailed Proof for Geometric Properties of Fine-tuned Weights

For all indices i, j within the set $[1, N]$, where N denotes the sufficiently large number of fine-tuned weights, we derive one lemma and three propositions based on the foundational observation described in Eq. (1):

Lemma: $\mathbf{w}_i \cdot \boldsymbol{\mu} = \boldsymbol{\mu} \cdot \boldsymbol{\mu} = l^2 \cos \theta$.

Proof:

$$\begin{aligned} \mathbf{w}_i \cdot \boldsymbol{\mu} &= \lim_{N \rightarrow \infty} \frac{1}{N} \mathbf{w}_i \cdot \sum_{k=1}^N \mathbf{w}_k = \lim_{N \rightarrow \infty} \frac{1}{N} (l^2 + (N-1) * l^2 \cos \theta) \\ &= l^2 \cos \theta. \end{aligned}$$

Similarly,

$$\begin{aligned} \boldsymbol{\mu} \cdot \boldsymbol{\mu} &= \lim_{N \rightarrow \infty} \frac{1}{N^2} \sum_{k=1}^N \mathbf{w}_k \cdot \sum_{l=1}^N \mathbf{w}_l = \lim_{N \rightarrow \infty} \frac{1}{N^2} (N * l^2 + N(N-1) * l^2 \cos \theta) \\ &= l^2 \cos \theta. \quad \square \end{aligned}$$

Proposition 1: $\|\mathbf{w}_i - \boldsymbol{\mu}\| = \text{constant}$.

Proof:

$$\begin{aligned} \|\mathbf{w}_i - \boldsymbol{\mu}\|^2 &= (\mathbf{w}_i - \boldsymbol{\mu}) \cdot (\mathbf{w}_i - \boldsymbol{\mu}) \\ &= \mathbf{w}_i \cdot \mathbf{w}_i - 2\mathbf{w}_i \cdot \boldsymbol{\mu} + \boldsymbol{\mu} \cdot \boldsymbol{\mu} \\ &= l^2 - 2l^2 \cos \theta + l^2 \cos \theta \quad (\text{by Lemma}) \\ &= l^2(1 - \cos \theta) \quad (\text{constant}) \quad \square \end{aligned}$$

Proposition 2: $(\mathbf{w}_0 - \boldsymbol{\mu}) \perp (\mathbf{w}_i - \boldsymbol{\mu})$.

Proof:

$$\begin{aligned} (\mathbf{w}_0 - \boldsymbol{\mu}) \cdot (\mathbf{w}_i - \boldsymbol{\mu}) &= -\boldsymbol{\mu} \cdot (\mathbf{w}_i - \boldsymbol{\mu}) \\ &= 0 \quad (\text{by Lemma}) \quad \square \end{aligned}$$

Proposition 3: $(\mathbf{w}_i - \boldsymbol{\mu}) \perp (\mathbf{w}_j - \boldsymbol{\mu})$.

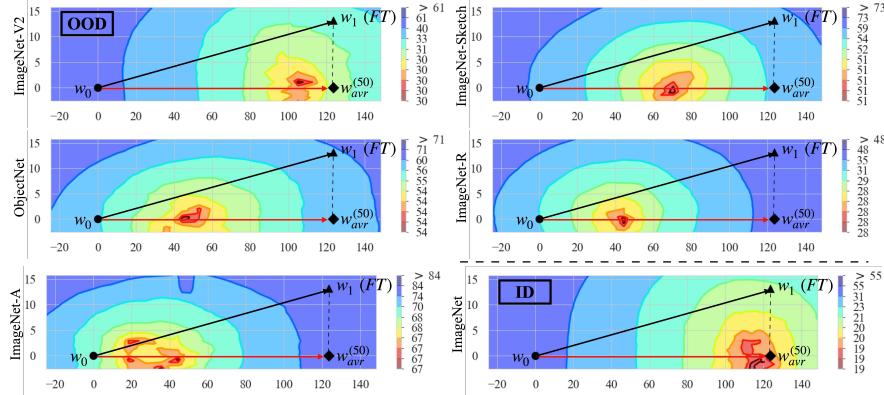


Fig. A: Test error landscape on OOD datasets. We depict the test error landscape on ImageNet-V2, -Sketch, ObjectNet, ImageNet-R, and -A (from left to right, from top to bottom, respectively) on the plane containing pre-trained model (\mathbf{w}_0), fine-tuned model (\mathbf{w}_1), and the pseudo-center of fine-tuned weights ($\mathbf{w}_{\text{avr}}^{(50)}$). The local optima for the OOD datasets always lie on the line segment $\mathbf{w}_0\mathbf{w}_{\text{avr}}^{(50)}$.

Proof:

$$\begin{aligned} (\mathbf{w}_i - \boldsymbol{\mu}) \cdot (\mathbf{w}_j - \boldsymbol{\mu}) &= \mathbf{w}_i \cdot \mathbf{w}_j - \mathbf{w}_i \cdot \boldsymbol{\mu} - \mathbf{w}_j \cdot \boldsymbol{\mu} + \boldsymbol{\mu} \cdot \boldsymbol{\mu} \\ &= 0 \quad (\text{by Eq. (1) \& Lemma}) \quad \square \end{aligned}$$

C Importance of Reducing Weight Variance on Performance under Distribution Shifts

In demonstrating the significance of variance reduction for robustness in out-of-distribution (OOD) scenarios, we analyze the test error landscape as in §2.2. As shown in Fig. A, we examine the error landscape across various OOD datasets, including ImageNet-V2, ImageNet-Sketch, ObjectNet, ImageNet-R, and ImageNet-A (from top to bottom). This landscape is plotted on a plane defined by the weights of a pre-trained model (\mathbf{w}_0), a fine-tuned model (\mathbf{w}_1), and the center of the fine-tuned weights, which is approximated by averaging 50 fine-tuned weights ($\mathbf{w}_{\text{avr}}^{(50)}$). A notable pattern emerges where the local optima for these datasets consistently align with the line segment connecting \mathbf{w}_0 and $\mathbf{w}_{\text{avr}}^{(50)}$.

Though the exact location of local minima differs depending on the dataset type, it has a common point that the minima are aligned on the line between the weight center and pre-trained model rather than the line between the fine-tuned weight and pre-trained model. Consequently, not only does the averaged weight exhibit higher performance on distribution shifts compared to the fine-tuned model, but the WiSE-FT [41] curves corresponding to the averaged weights also demonstrate better ID/OOD trade-off than the WiSE-FT curve of the fine-tuned model, as illustrated in Fig. B. This indicates the importance of getting closer to the weight center, even for OOD datasets.

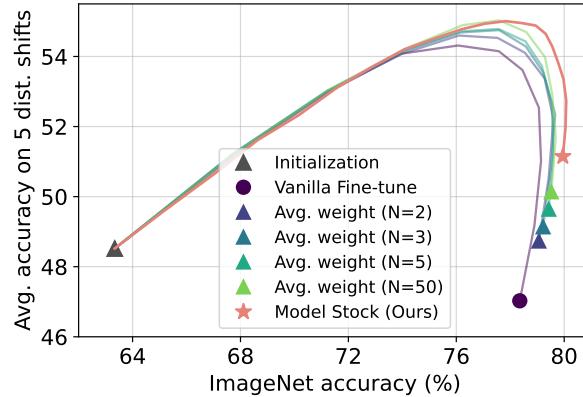


Fig. B: ID vs. OOD accuracy along WiSE-FT [41] curves for averaged models. As the number of weights used for averaging increases, the corresponding WiSE-FT curves demonstrate improvements in the ID-OOD trade-off.

Another interesting point is that depending on the traits of datasets, the position of local minima differs. ImageNet-V2 has a similar dataset distribution to ImageNet since it shares the same data collection and categorization policy, and its local optima lies close to that of ImageNet. On the other hand, on the datasets with harsh variations (*e.g.*, ImageNet-A), the local minima are positioned much closer to the pre-trained model than the original ImageNet or ImageNet-V2. This loss landscape gives an intuitive insight into the similarity between OOD datasets and ImageNet.

In conclusion, there is no universal interpolation ratio optimal for every distribution shift. However, all the local minima lie on the line between the weight center and the pre-trained model. This implies the importance of proximity to the weight center in achieving a better WiSE-FT line.

D Detailed Proof of Model Stock

Here, we present detailed proof of Model Stock introduced in §3. We first show the case with two fine-tuned models and extend our proof toward N fine-tuned models.

On two fine-tuned models. We will prove step-by-step how the optimal interpolation ratio t in Eq. (2) in the main paper is derived. Using the same notation as in §3, we denote the magnitude and the angle between the fine-tuned weights as l and θ , respectively. Starting from the fact that $\triangle \mu w_1 w_2$ is a right isosceles triangle, we can derive the following relations from Fig. C:

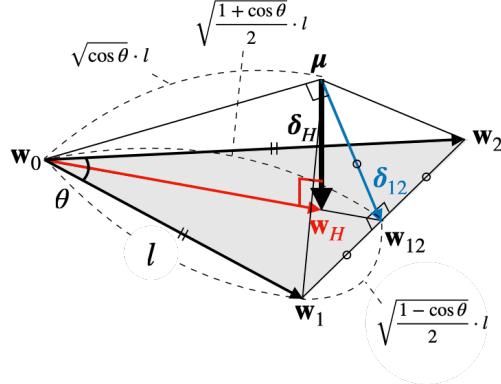


Fig. C: Model Stock with two fine-tuned models. We reference the illustration in Fig. 6 to more understandably substantiate merging two fine-tuned models.

$$\begin{aligned} \overline{\mathbf{w}_{12}\mathbf{w}_1} &= \overline{\mathbf{w}_{12}\mathbf{w}_2} = \overline{\mathbf{w}_{12}\boldsymbol{\mu}} \\ &= \sqrt{\frac{1 - \cos\theta}{2}} \cdot l \quad (\text{from } \triangle \mathbf{w}_0\mathbf{w}_1\mathbf{w}_2 \text{ and } \triangle \boldsymbol{\mu}\mathbf{w}_1\mathbf{w}_2) \end{aligned} \quad (4)$$

$$\begin{aligned} \Rightarrow \overline{\mathbf{w}_{12}\mathbf{w}_0} &= \sqrt{\overline{\mathbf{w}_1\mathbf{w}_0}^2 - \overline{\mathbf{w}_{12}\mathbf{w}_1}^2} \\ &= \sqrt{1^2 - \frac{1 - \cos\theta}{2}} \cdot l \\ &= \sqrt{\frac{1 + \cos\theta}{2}} \cdot l \quad (\text{from } \triangle \mathbf{w}_0\mathbf{w}_1\mathbf{w}_{12} \text{ and Eq. (4)}) \end{aligned} \quad (5)$$

$$\begin{aligned} \Rightarrow \overline{\mathbf{w}_0\boldsymbol{\mu}} &= \sqrt{\overline{\mathbf{w}_{12}\mathbf{w}_0}^2 - \overline{\mathbf{w}_{12}\boldsymbol{\mu}}^2} \\ &= \sqrt{\frac{1 + \cos\theta}{2} - \frac{1 - \cos\theta}{2}} \cdot l \\ &= \sqrt{\cos\theta} \cdot l \quad (\text{from } \triangle \mathbf{w}_0\boldsymbol{\mu}\mathbf{w}_{12}, \text{ Eq. (4) and Eq. (5)}) \end{aligned} \quad (6)$$

$$\begin{aligned} \Rightarrow t &:= \frac{\overline{\mathbf{w}_H\mathbf{w}_0}}{\overline{\mathbf{w}_{12}\mathbf{w}_0}} = \frac{\overline{\mathbf{w}_H\mathbf{w}_0}}{\overline{\mathbf{w}_0\boldsymbol{\mu}}} \cdot \frac{\overline{\mathbf{w}_0\boldsymbol{\mu}}}{\overline{\mathbf{w}_{12}\mathbf{w}_0}} \\ &= \left(\frac{\overline{\mathbf{w}_0\boldsymbol{\mu}}}{\overline{\mathbf{w}_{12}\mathbf{w}_0}} \right)^2 \quad (\text{from } \triangle \mathbf{w}_0\boldsymbol{\mu}\mathbf{w}_{12} \sim \triangle \mathbf{w}_0\mathbf{w}_H\boldsymbol{\mu}) \\ &= \frac{2 \cos\theta}{1 + \cos\theta} \quad (\text{from Eq. (5) and Eq. (6)}) \end{aligned} \quad (7)$$

□

Interestingly, \mathbf{w}_H is located at an orthocenter of the triangle $\triangle \mathbf{w}_0\mathbf{w}_1\mathbf{w}_2$ with the given optimal ratio t .

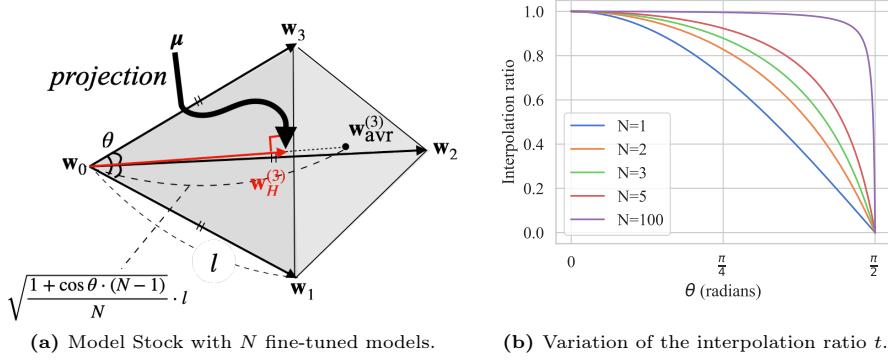


Fig. D: Model Stock with N fine-tuned models and Interpolation Ratio Variation. (a) We visualize a special case of $N = 3$ (tetrahedron) for better understanding. (b) The trend towards $t = 1$ with increasing N illustrates that $\mathbf{w}_H^{(N)}$ on the N -dimensional simplex approaches $\mathbf{w}_{\text{avr}}^{(N)}$, reflecting a growing dependence on the number of fine-tuned models.

On N fine-tuned models. Similarly, we can derive a more generalized interpolation ratio for $N \geq 2$. Our goal is to find the weight $\mathbf{w}_{\text{avr}}^{(N)}$ that is on the hyper-plane spanned by $\mathbf{w}_0, \mathbf{w}_1, \dots, \mathbf{w}_N$ and closest to the weight center $\boldsymbol{\mu}$, as described in Fig. Da. Again, for simplicity, we treat \mathbf{w}_0 as the origin \mathbf{O} .

Based on the observation, we presume that the following two conditions hold:

$$\begin{cases} \mathbf{w}_H^{(N)} = t \cdot \mathbf{w}_{\text{avr}}^{(N)} \\ (\mathbf{w}_{\text{avr}}^{(N)} - \mathbf{w}_H^{(N)}) \cdot (\boldsymbol{\mu} - \mathbf{w}_H^{(N)}) = 0. \end{cases} \quad (8)$$

The first condition comes from the symmetry of an N -simplex structure, and the second condition holds since the orthogonal projection is the minimal distance from $\boldsymbol{\mu}$. Then, we can derive t as follows:

By substituting the first condition into the second condition from Eq. (8),

$$\begin{aligned} & (\mathbf{w}_{\text{avr}}^{(N)} - \mathbf{w}_H^{(N)}) \cdot (\boldsymbol{\mu} - \mathbf{w}_H^{(N)}) = 0 \\ & \Rightarrow \mathbf{w}_{\text{avr}}^{(N)} \cdot \boldsymbol{\mu} - t \cdot \|\mathbf{w}_{\text{avr}}^{(N)}\|^2 = 0 \\ & \Rightarrow t = \frac{\boldsymbol{\mu} \cdot \mathbf{w}_{\text{avr}}^{(N)}}{\|\mathbf{w}_{\text{avr}}^{(N)}\|^2}. \end{aligned} \quad (9)$$

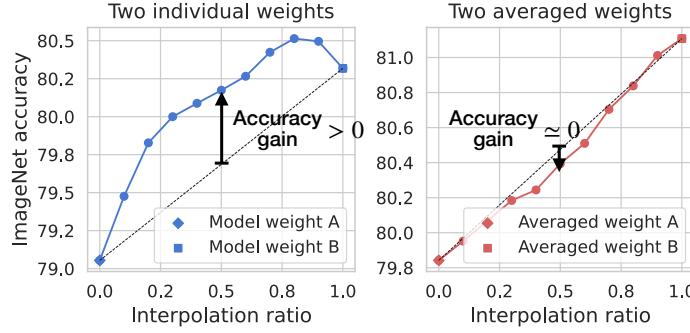


Fig. E: Ensembling impact disappears when interpolating between two averaged weights. We plot the ImageNet performance of interpolated weights between two selected fine-tuned models in Model Soup [40] (left) and between their corresponding weight centers (right).

Note that the norm of the N -averaged fine-tuned weights can be derived as follows:

$$\begin{aligned} \|\mathbf{w}_{\text{avr}}^{(N)}\|^2 &= \frac{1}{N^2}(\mathbf{w}_1 + \dots + \mathbf{w}_N) \cdot (\mathbf{w}_1 + \dots + \mathbf{w}_N) \\ &= \frac{1}{N^2}(l^2 + l^2 \cos \theta \cdot (N - 1)) \cdot N \\ &= \frac{l^2}{N}(1 + \cos \theta \cdot (N - 1)), \end{aligned} \quad (10)$$

while the term $\boldsymbol{\mu} \cdot \mathbf{w}_{\text{avr}}^{(N)}$ can be simplified as

$$\boldsymbol{\mu} \cdot \mathbf{w}_{\text{avr}}^{(N)} = \frac{1}{N} \sum_{i=1}^N (\boldsymbol{\mu} \cdot \mathbf{w}_i) = l^2 \cos \theta \quad (\text{from Lemma}). \quad (11)$$

By substituting Eq. (10) and Eq. (11) into Eq. (9), we can finally derive the optimal interpolation ratio t as follows:

$$t = \frac{N \cos \theta}{1 + (N - 1) \cos \theta} \quad \square \quad (12)$$

Fig. Db displays how the optimal interpolation ratio t varies as a function of θ with different numbers of fine-tuned models. As N increases, t trends towards 1, indicating that $\mathbf{w}_H^{(N)}$ on the N -dimensional simplex gets closer to $\mathbf{w}_{\text{avr}}^{(N)}$. This shows increasing dependence on fine-tuned models as their number grows.

E Discussion — Rethinking Pivotal Prior Studies

In this section, we extend our findings to reinterpret the underlying mechanics in prior studies, WiSE-FT [41] and Model Soups [40], through a consistent rationale to illuminate their effectiveness.

WiSE-FT [41] is a state-of-the-art robust fine-tuning method for CLIP-based models. It demonstrates that linearly combining weights of the pre-trained and fine-tuned models achieves significant accuracy gain on distribution shifts. We argue that the WiSE-FT model’s superiority over a fine-tuned model can be interpreted by its weights being closer to the center of the corresponding weight distribution. Fig. 3 already showed fine-tuned models typically lie on the periphery of flat minima. Given that the angle $\angle \mathbf{w}_0 \mathbf{w}_{\text{avr}}^{(50)} \mathbf{w}_1$ is nearly a right angle, along the line $\overline{\mathbf{w}_0 \mathbf{w}_1}$, multiple weight points are closer to the center than a single fine-tuned model, thereby enhancing performance. Note that \mathbf{w}_H is the closest to the center among the line $\overline{\mathbf{w}_0 \mathbf{w}_1}$. More discussions on performance boosts observed in distribution shifts are provided in the Appendix C.

Model Soup [40] merges various fine-tuned models’ weights trained from varied hyper-parameters. It has been credited with delivering enhanced performance across ImageNet and distribution shifts. Here, we interpret the performance improvements of Model Soup as the result of the proximity to the center of weight distribution. Consider two weight vectors, \mathbf{w}_A and \mathbf{w}_B , fine-tuned with different hyper-parameters and following Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_A, \Sigma_A)$ and $\mathcal{N}(\boldsymbol{\mu}_B, \Sigma_B)$ respectively. Then, the interpolated weight vector $\mathbf{w}_{AB} = t \cdot \mathbf{w}_A + (1 - t) \cdot \mathbf{w}_B$ also follows a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{AB}, \Sigma_{AB})$. The expected squared distance from the interpolated weight vector to its mean $\boldsymbol{\mu}_{AB}$ is minimized to $\frac{\text{trace}(\Sigma_A)\text{trace}(\Sigma_B)}{\text{trace}(\Sigma_A)+\text{trace}(\Sigma_B)}$ when t is chosen to $\frac{\text{trace}(\Sigma_B)}{\text{trace}(\Sigma_A)+\text{trace}(\Sigma_B)}$, indicating the reduction of variance through weight interpolation (*i.e.*, the distance between \mathbf{w}_{AB} and $\boldsymbol{\mu}_{AB}$ might be closer than each weight’s distance). For example, if $\text{trace}(\Sigma_B)$ is equal to $\text{trace}(\Sigma_A)$, this minimum squared distance is exactly half of the sum of the individual traces when $t = 0.5$. This insight suggests that the performance gains realized by Model Soup could be due to reduced variance resulting from merging numerous weights.

We set up a toy experiment to evaluate the effect of variance reduction in the Model Soup scenario by comparing the interpolation of fine-tuned weights with the interpolation of their corresponding weight centers, when $N = 2$. In the former case, variance reduction exists along with the effect of merging diverse hyper-parameters, while in the latter case, performance gain would only come from hyper-parameter diversity. If the diversity of hyper-parameters is a major factor, the performance gain from interpolation of central weights should remain the same. To test this, we assessed the ImageNet performance of interpolated weights between pairs of fine-tuned models within Greedy Model Soup⁵ [40] and compared it to interpolations between their central weights, calculated as the average of 20 differently seeded models. Fig. E shows that, unlike interpolations between individual models, using the centers does not significantly improve performance. This suggests that proximity to the center of the weight distribution may play a more critical role than hyper-parameter diversity in weight ensemble methods in this case.

⁵ We opt for Greedy Model Soup to show that even the interpolation of models from the best merging combination does not benefit from the impact of weight diversity.

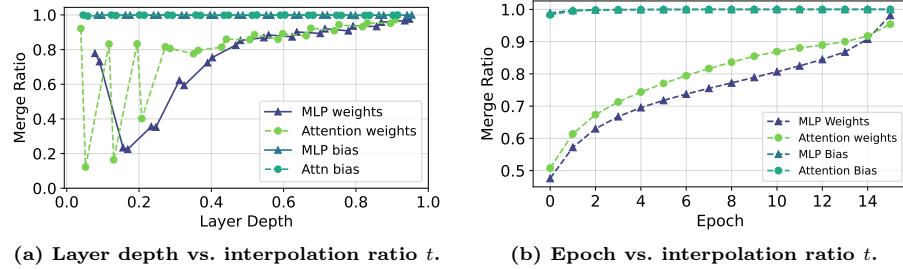


Fig. F: Trend of interpolation ratio t during Model Stock training.

It is also worth noting that μ_{AB} always surpasses the performance of \mathbf{w}_{AB} for the same interpolation ratio t , indicating that the importance of proximity to the center remains consistent for interpolated weights. With extensive future research, this understanding could provide valuable insights for developing more generalizable and effective weight-merging techniques.

F Analysis of the interpolation ratio t

We analyze the interpolation ratio $t = \frac{2\cos\theta}{1+\cos\theta}$ in a layer-wise manner. During a Model Stock experiment on CLIP ViT-B/32 with 16 epoch training, we log the layer-wise merge ratios at every merging period. Figure F visualizes the averaged interpolation ratio during Model Stock training. We plot two trends of the interpolation ratio for the layer depth and training step. Our overall observation indicates the bias layers have high merge ratios t ($\simeq 1$) with small angles θ ($\simeq 0$), implying that the bias layers do not need to enjoy the pre-trained model, similar to our discussion in §2 and §3. Focusing on the weight layers, Figure Fa shows a U-shape tendency as the layer depth increases, implying the weights of intermediate layers can be more diverse (*i.e.*, larger angle θ) than those of early and later layers. Our intuition here is that since the early and later layers are directly connected to input data and output labels, respectively, they may not demand the advantage of the pre-trained weight. Figure Fb presents that the models at the early training stage are more diverse and they enjoy the pre-trained weights more than those of the later training stage. As the model approaches convergence, the diversity of fine-tuning models decreases (*i.e.*, smaller angle θ).

G Experimental setup

Here, we present detailed setups for the experiments in §4. We utilize AdamW optimizer [25] with a weight decay of 0.1. We employ two training setups for Model Stock. The first is training Model Stock with a learning rate of 3×10^{-5} in 10 epochs with minimal data augmentation. The minimal data augmentation utilizes random resize crop augmentation with a minimum crop ratio of 0.9,

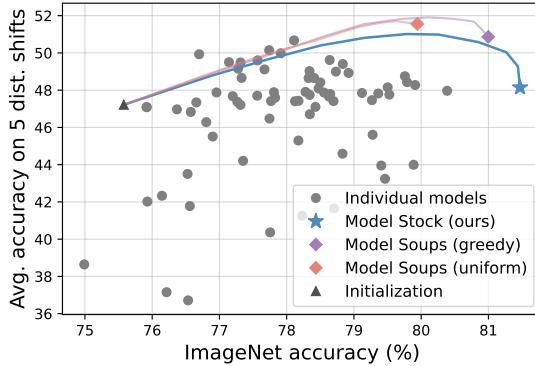


Fig. G: Results on LP initialization. We plot in-distribution ImageNet accuracy (x-axis) and distribution shift results (y-axis) with individual fine-tuned models (gray circles) and Model Soups [40]. Note that Model Stock has much smaller ($35\times$ smaller) computational costs than Model Soups, leveraging 71 various fine-tuned models as in the original paper.

mixup [44] augmentation with $\beta=0.5$, following Model Soup’s “standard grid search” setting. The other is training Model Stock with a learning rate of 2×10^{-5} in 16 epochs with strong data augmentation. The strong data augmentation utilizes random resize crop augmentation with a minimum crop ratio of 0.08 and random augmentation [4] ($N = 2$, $M = 10$) following Model Soup’s “random search” setting. When experimenting with the ViT-B/16 and ViT-L/14 models, we adjusted the learning rate and batch size to accommodate the GPU memory constraints.

H Additional Experiments

We present additional experimental studies to verify the effectiveness and applicability of Model Stock.

H.1 Experiments with LP initialization

We conduct Model Stock with LP initialization and compare it with Model Soups that are initialized from LP. The results are in Fig. G. In this experiment, we use the 16-epoch training setup with strong data augmentation for training Model Stock. As shown in Fig. G, Model Stock outperforms the individual fine-tuned models⁶ (gray dots) on ImageNet accuracy. Model Stock also demonstrates competitive performance against Model Soups considering WiSE-FT curves. Note that Model Stock is much more efficient ($35\times$) than Model Soups, which utilize 71 models in this experiment.

⁶ All the individual model checkpoints are from the official Model Soup repository.

Table A: Complete results of Table 3 with ObjectNet [1] and ImageNet-Real [2].

Method	In-distribution		Distribution shifts				
	ImageNet	IN-RealL	IN-V2	IN-R	IN-A	IN-Sketch	ObjectNet
Zero-shot	68.3	75.1	62.0	77.7	49.9	48.3	54.2
Vanilla FT	82.8	87.8	72.9	66.4	43.7	48.0	51.8
Vanilla FT*	83.7	87.8	73.5	67.6	40.0	48.6	50.1
LP [18]	79.7	-	71.5	52.4	27.8	40.5	-
LP-FT [18]	81.7	-	71.6	<u>72.9</u>	49.1	48.4	-
CAR-FT [27]	83.2	-	73.0	71.3	43.7	49.5	-
FTP [37]	<u>84.2</u>	-	74.6	47.2	26.5	50.2	-
FLYP [7]	82.6	-	73.0	71.4	48.1	49.6	58.7
Lipsum-FT [28]	83.3	-	73.6	75.9	49.9	51.4	54.4
CaRot [29]	83.1	-	74.1	77.7	51.6	52.7	56.6
Model Stock	84.1	<u>88.8</u>	74.8	71.8	<u>51.2</u>	<u>51.8</u>	55.0
Model Stock*	85.2	89.1	75.3	68.7	45.0	51.3	52.3

Table B: Comparison against Model Soups [40] on CLIP ViT-B/16. Model Stock shows comparable performance with Model Soups.

Method	ImageNet	Avg. shifts
CLIP zero-shot Init.	68.3	58.4
Vanilla FT	82.8	56.6
Vanilla FT*	83.7	55.9
Uniform Model Soup	<u>84.4</u>	62.7
Greedy Model Soup	84.3	60.4
Model Stock	84.1	<u>61.0</u>
Model Stock*	85.2	58.5

H.2 Complete comparison results on CLIP ViT-B/16

In the main paper, we omit the results of ObjectNet [1] on CLIP ViT-B/16 experiments since the comparison methods such as LP-FT [18], FTP [37] have not evaluated on ObjectNet benchmark. We here show the results with ObjectNet [1] and ImageNet-Real [2] of CLIP ViT-B/16 in Table A. We additionally compare Model Stock with recent fine-tuning methods including FLYP [7], Lipsum-FT [28], and CaRot [29] Model Stock consistently demonstrates its effectiveness with ObjectNet and ImageNet-Real as well.

H.3 Model Stock vs. Model Soups on CLIP ViT-B/16

Table B shows the performance of Model Stock on the pretrained CLIP ViT-B/16 model. Since the original Model Soups paper [40] only provides CLIP ViT-B/32 models, we replicate Model Soups experiments on CLIP ViT-B/16. We fine-tuned 48 models from CLIP ViT-B/16 initialization following the standard grid hyper-parameter sweep (*i.e.*, zero-shot initialization setting). Model Stock shows

Table C: Model Stock with different hyper-parameters on CLIP ViT-B/32.

Method	ImageNet	Avg. shifts
Model Stock	79.89	50.99
Model Stock w/ different hyper-parameters	79.75±0.45	50.40±0.84

Table D: Performance comparison of merging units in Model Stock. This table presents the overall performance of Model Stock using different merging units: entire weight merging, entire weight merging based on transformer block angle, layer-wise merging, and filter-wise merging. It highlights the effectiveness of each strategy in approaching the weight center and their impact on the model’s performance.

Merging Unit	Target		Avg. Shifts
	IN	IN-ReaL	
Entire weights	79.69	85.39	46.40
Entire weights (rep. blocks only)	79.64	85.38	48.28
Layer-wise (ours)	80.12	<u>85.65</u>	48.84
Filter-wise	<u>80.10</u>	85.67	48.72

comparable performance against Model soups. Note that Model Soups requires 24× more training cost than Model Stock.

H.4 Model Stock with different hyper-parameters

To verify the validity of Model Stock beyond the setup of the main paper (*i.e.*, different random seeds with the same hyper-parameters), we conduct Model Stock with different hyper-parameters. In detail, when we fine-tune two models for Model Stock, we choose different hyper-parameter for each model (*e.g.*, learning rate, data augmentation.). To ensure the basic assumption of Model Stock, we use the same batch size and training epochs. C shows the experimental results on CLIP ViT-B/32. We repeat 5 runs and report accuracy with standard deviation. Model Stock with different hyper-parameters shows comparable performance to the original one.

H.5 Ablation study on merging unit

We investigate the efficacy of different merging units within our method, Model Stock. Our default approach employs layer-wise merging, but alternatives include merging based on the angle between 1) entire weights, 2) weights of the entire repetitive transformer blocks following [40], or 3) using a filter-wise approach as discussed in §A.3. The results of these ablations are summarized in Table D, where we assess the overall performance based on the chosen merging unit.

Our analysis reveals that the accuracy of noise distribution estimation is critical in approaching the weight center. When assuming weight noise across the

entire model, our method does not approximate the weight center as effectively as it does with layer-wise merging, leading to suboptimal overall performance. Similarly, the merging performance based on the angle of transformer blocks was insufficient. Conversely, while filter-wise noise demonstrates a larger standard deviation in angle, as depicted in Fig. N, this increased variance results in a more significant error in Gaussian distribution approximation. Consequently, the overall performance under filter-wise merging is slightly inferior to layer-wise one.

These findings underscore the importance of accurately modeling noise distribution in enhancing the performance of Model Stock. As our understanding and ability to model this noise distribution improve, we anticipate further increases in the efficacy and robustness of our approach.



Fig. H: Layer-wise angle and norm across different model architectures. The angle and norm for CLIP ViT-L/14, CLIP ResNet50, and OpenCLIP ConvNeXt are displayed from top to bottom. These metrics demonstrate consistency regardless of the model type from left (first layer) to right (last layer). It is important to note that we also depict the error bars for each layer in all figures, but they are not visible in most layers due to the small standard deviation.



Fig. I: Layer-wise angle and norm across different optimizers. Displayed from top to bottom are the angle and norm for models trained with SGD and SGD with momentum, respectively. These metrics demonstrate consistency regardless of the optimization strategy from left (first layer) to right (last layer).



Fig. J: Layer-wise angle and norm across different augmentations. Displayed from top to bottom are the angle and norm for the vanilla model (10 epochs + no augmentation), +longer epochs (16 epochs), and +RRC. Each augmentation is applied incrementally. These metrics demonstrate consistency regardless of the augmentations from left (first layer) to right (last layer).

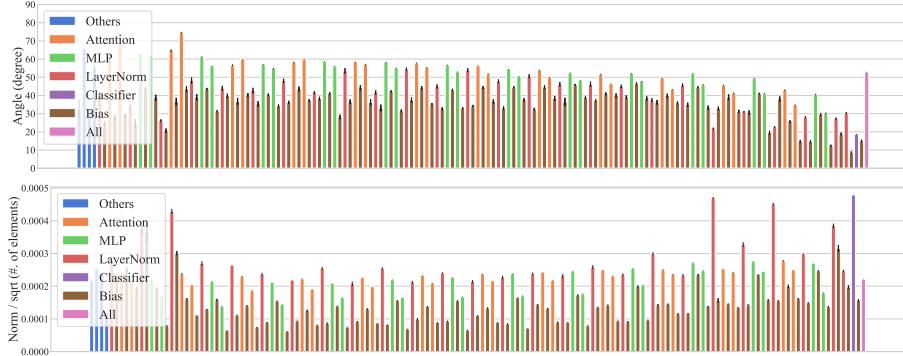


Fig. K: Layer-wise angle and norm across different datasets. The angle and norm for models trained on different datasets, including CIFAR [17] are displayed from top to bottom. These metrics demonstrate consistency regardless of the dataset type from left (first layer) to right (last layer).

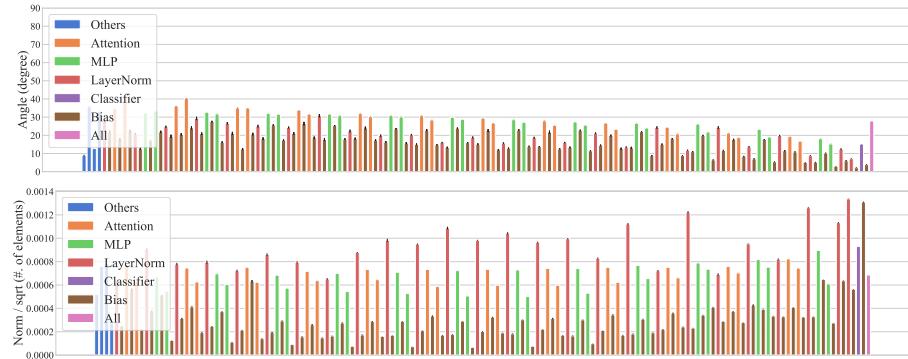


Fig. L: Layer-wise angle and norm across different classifier initializations. The angle and norm for models trained with differently initialized networks following the LP-FT [18] method are displayed from top to bottom. These metrics demonstrate consistency regardless of the initialization method from left (first layer) to right (last layer).

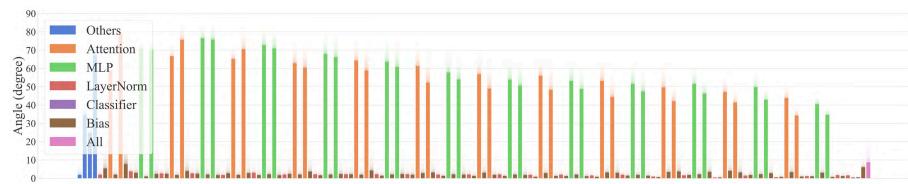


Fig. M: Layer-wise angle during training. Displayed are the overlapped angles across models trained with different random seeds at each timestamp. Even during training, the angle remains highly consistent, decreasing as training progresses.

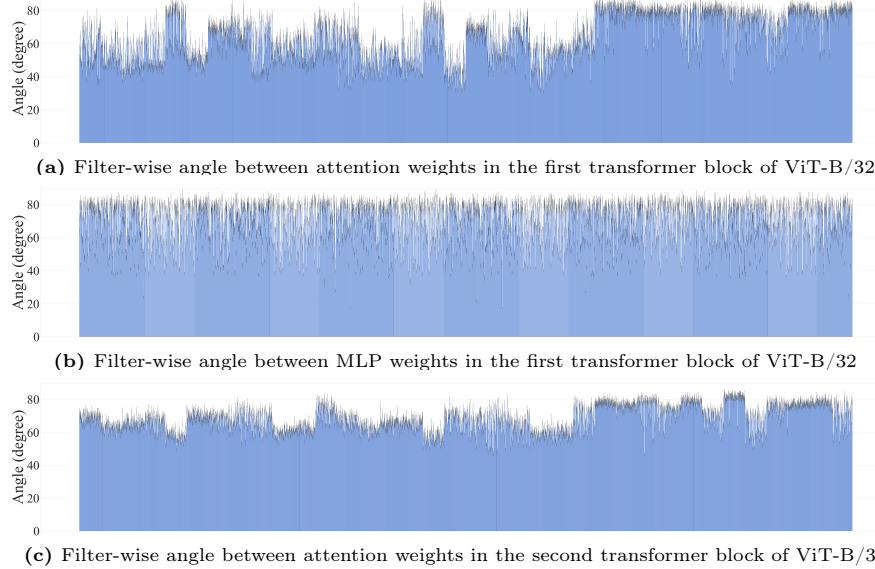


Fig. N: Filter-wise angle for attention and MLP layers in ViT-B/32. We display filter-wise angles for each layer. Each bar represents each row (*i.e.*, filter) in the given layer. Interestingly, the angles between the filters of the fine-tuned weights exhibit similar values, while the standard deviation between each filter is notably larger than that of the angle between each layer. Due to the large number of layers, only representative layers are selected for display.

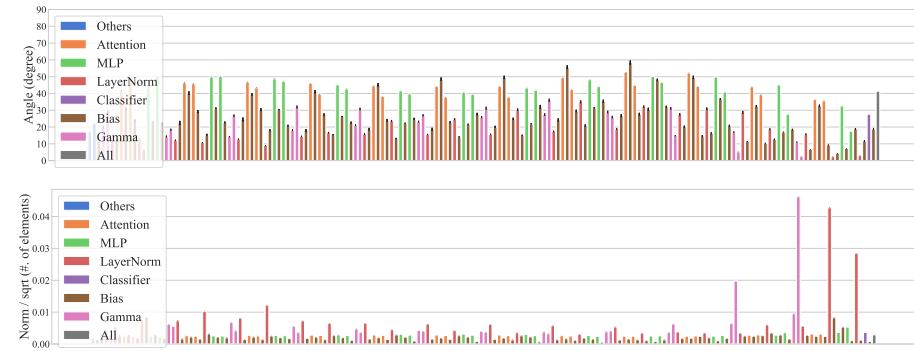


Fig. O: Layer-wise angle and norm for DeiT. The angle and norm for DeiT-base models are displayed, each trained with different random seeds. These models are initially pre-trained on ImageNet-21K [34] and then fine-tuned on ImageNet-1K. The consistency observed in the metrics is maintained even in the DeiT training setting.