Eigenspectrum Analysis of Neural Networks without Aspect Ratio Bias

Yuanzhe Hu [1] **Kinshuk Goel** [2 3] **Vlad Killiakov** [4] **Yaoqing Yang** [2]

Abstract

Diagnosing deep neural networks (DNNs) by analyzing the eigenspectrum of their weights has been an active area of research in recent years. One of the main approaches involves measuring the *heavytailness* of the empirical spectral densities (ESDs) of weight matrices. This analysis has been shown to provide insights to help diagnose whether a model is well-trained or undertrained, and has been used to guide training methods involving layer-wise hyperparameter assignment. In this paper, we address an often-overlooked challenge in estimating the heavytailness of these ESDs: the impact of the aspect ratio of weight matrices. We demonstrate that matrices of varying sizes (and aspect ratios) introduce a non-negligible bias in estimating the heavytailness of ESDs, leading to inaccurate model diagnosis and layer-wise hyperparameter assignment. To overcome this challenge, we propose `FARMS` (**F**ixed-**A**spect-**R**atio **M**atrix **S**ubsampling), a method that normalizes the weight matrices by subsampling submatrices with a fixed aspect ratio. Instead of measuring the heavytailness of the original ESD, we measure the average ESD of these subsampled submatrices. We show that this method effectively mitigates the aspect ratio bias. We validate our approach across various optimization techniques and application domains that involve eigenspectrum analysis of weights, including image classification in computer vision (CV) models, scientific machine learning (SciML) model training, and large language model (LLM) pruning. Our results show that despite its simplicity, `FARMS` uniformly improves the accuracy of eigenspectrum analysis while enabling more effective layer-wise hyperparameter assignment. In one of the LLM pruning experiments, `FARMS` reduces the perplexity of the LLaMA-7B model by 17.3% when compared with state-of-the-art methods.

1. Introduction

Random Matrix Theory (RMT) has long been used as a foundational tool in multiple research domains (Guhr et al., 1998; Tulino & Verdú, 2004; Bai & Silverstein, 2010; Tao, 2012; Nadakuditi & Newman, 2012) and has now been applied to providing precise characterizations of neural networks (NNs) (Pennington & Worah, 2017; Dobriban & Wager, 2018; Adlam & Pennington, 2020; Mei & Montanari, 2022; Hastie et al., 2022; Hu & Lu, 2022; Ba et al., 2022; Couillet & Liao, 2022; Derezinski et al., 2020). Among several emerging topics in this field, Heavy-Tailed Self-Regularization (HT-SR) (Martin & Mahoney, 2021; Mahoney & Martin, 2019; Martin et al., 2021) Theory has been gaining significant attention. Unlike conventional RMT studies, HT-SR Theory focuses on studying weight matrices with strongly correlated elements, a typical characteristic of well-trained, practical DNNs. These strong correlations lead to heavy-tailed (HTed) ESDs (Martin & Mahoney, 2021). Analyzing these HT ESDs provides valuable insights into the quality of trained models, enabling both model diagnostics and improvements in model training. For example, Martin & Mahoney (2021) show that one can locate undertrained layers by finding weight matrices with less HTed ESDs, which provides useful metrics for model selection and ranking (Martin et al., 2021; Yang et al., 2023). Furthermore, one can use a larger learning rate on the undertrained layers (Zhou et al., 2024), which provides an empirically successful training method to balance these layers with other, more well-trained ones. This technique has been developed into a general approach to improve test accuracy, training efficiency, and model interpretability in various applications, such as image classification (Zhou et al., 2024), LLM model pruning (Lu et al., 2024), and SciML model training and fine-tuning (Liu et al., 2024; 2025b).

The core of the HT-SR theory lies in the measurement of the heavytailness of ESDs. In this context, we identified an often-overlooked issue in prior studies: weight matrices

[1]Department of Computer Science and Engineering, University of California, San Diego, CA, USA [2]Department of Computer Science, Dartmouth College, NH, USA [3]Department of Computer Science & Engineering, SRM Institute of Science & Technology, India [4]Independent Researcher, University of California, Berkeley, CA, USA. Correspondence to: Yuanzhe Hu <yuanzhe.hu@dartmouth.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

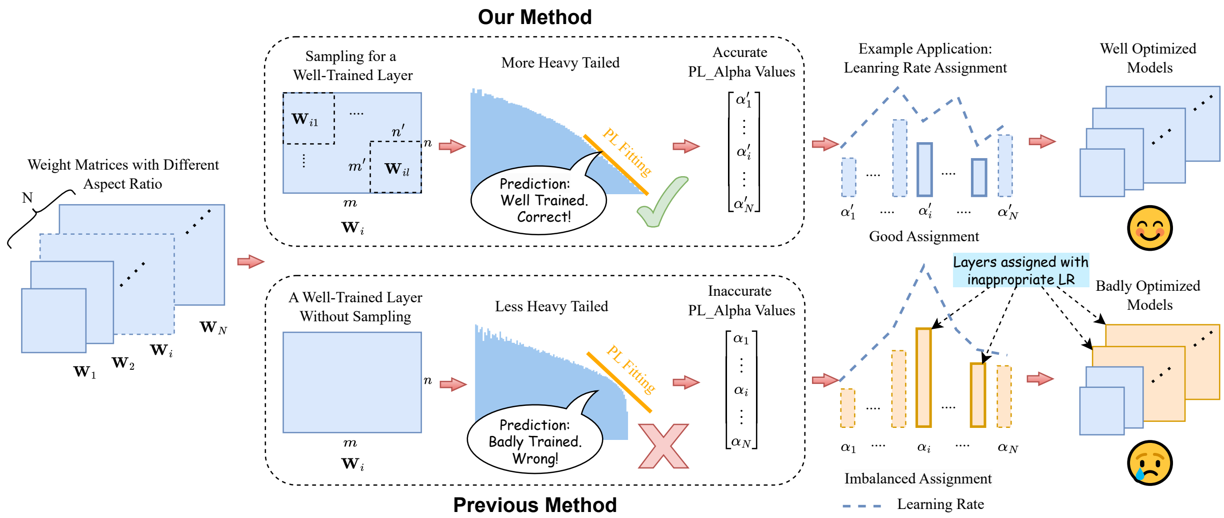


Figure 1. Comparing FARMS with common HT-SR methods in analyzing the weight matrices of a DNN. We use the weight matrix $\mathbf{W}_i$ of a well-trained layer with a large aspect ratio as an example (i.e., $\gamma = m/n$ is much larger than 1). Due to the influence of the aspect ratio $\gamma = m/n \gg 1$, the ESD is more concentrated and less HTed than other layers. As a result, previous methods for fitting a power-law distribution (shown as "PL Fitting") tend to overestimate the heavytailness metric `PL_Alpha` (the HT index) and incorrectly suggest that the layer is poorly trained. Due to this aspect ratio bias, some optimization methods (such as `TempBalance`) cannot accurately assign per-layer hyperparameters, leading to suboptimal training. In contrast, our method, FARMS, which conducts ESD analysis with a fixed aspect ratio, accurately measures the training quality of the layer. See a concrete numerical example in Figure 7 in Appendix A.

of different sizes are analyzed using the same HT measuring procedure, even though *their ESDs can differ in theory*. Specifically, algorithms inspired by the HT-SR theory often begin by measuring the ESD of a weight matrix $\mathbf{W}$, that is, the empirical distribution of eigenvalues of the correlation matrix $\mathbf{W}^\top\mathbf{W}$. The shape of the ESD is then analyzed by fitting it to certain distribution families, such as the power-law (PL) distribution. For weight matrices initialized using the i.i.d. random Gaussian elements, it is well-known that the ESD converges to the Marchenko–Pastur (MP) distribution as the dimensions of the matrix grow infinitely large in the proportional limit, i.e., when the number of rows m and the number of columns n increase but the ratio m/n stays fixed. The MP distribution depends explicitly on the aspect ratio $\gamma = m/n$, since this ratio determines the limiting bulk range, $[(1-\sqrt{\gamma})^2, (1+\sqrt{\gamma})^2]$, and it influences the overall shape of the ESD. As a result, weight matrices of different aspect ratios exhibit distinct shapes in their ESDs, especially when they transition from the i.i.d. initialization to being correlated. Comparing ESDs with different aspect ratios γ directly, as is commonly done in prior work, overlooks this dependency on aspect ratio and can lead to inaccurate shape estimates. Therefore, a careful consideration of the aspect ratio is critical when analyzing or comparing weight matrices, especially in eigenspectrum analysis.

Considering these challenges, we introduce FARMS (**F**ixed-**A**spect-**R**atio **M**atrix **S**ubsampling), a new method for measuring HT characteristics. FARMS partitions each weight matrix into (overlapping) submatrices of fixed aspect ratios and then does an eigenspectrum analysis on the average ESD of these submatrices. This approach enables accurate computation of heavytailness metrics regardless of variations in matrix aspect ratios, ensuring a robust evaluation of layer quality across diverse matrix sizes that one can have in a DNN.

To validate the effectiveness of FARMS, we conduct experiments across various application domains, including CV, SciML, and LLM pruning. In addition, these experiments are conducted on different parameter settings and model architectures. We compare FARMS with several prior HT-SR approaches that use weight eigenspectrum analysis for layer-wise hyperparameter assignment (Zhou et al., 2024; Lu et al., 2024; Liu et al., 2024). We also apply FARMS to measure various post-training and pruned models, making FARMS useful for model compression. Our findings demonstrate that models optimized using FARMS exhibit lower mean and variation in HT-SR metrics across layers, a sign of good-quality training as reported in prior work (Martin et al., 2021; Liu et al., 2024). Our code is available here[1]. Our key contributions are summarized below.

[1] https://github.com/HUST-AI-HYZ/FARMS

- **Mitigating Aspect Ratio Bias in Eigenspectrum Analysis**. FARMS addresses the aspect ratio bias of existing HT-SR eigenspectrum analysis, and it enables better computation of HT metrics on DNN models that have heterogeneous layer types. This improvement is achieved through a subsampling-based HT estimation method independent of the aspect ratio of weight matrices. In particular, we use a numerical example in Figure 6 and multiple real-data experiments in Section 4.3 to convincingly demonstrate that FARMS mitigates the aspect ratio bias in training.
- **Improved Layer-wise Hyperparameter Tuning**. Since HT-SR Theory has been recently applied to various layer-wise hyperparameter tuning methods, FARMS thus improves these methods by offering a more accurate evaluation of HT metrics. In particular, we conduct experiments on DNN training and pruning across various model architectures. In CV models like ResNet and VGG, integrating FARMS into learning rate assignments yields improved accuracy compared to TempBalance, a recently proposed layer-wise learning rate assignment method (Zhou et al., 2024). In LLMs pruning, FARMS improves AlphaPruning (Lu et al., 2024), the SOTA method in assigning layer-wise pruning ratios. Specifically, FARMS can reduce the perplexity of the LLaMA-13B model from 2029.20 to 413.76 using the magnitude pruning method at a 0.7 sparsity ratio. As another example in LLaMA-7B, it reduces the perplexity from 96.02 to 79.42 using the SparseGPT pruning method (Frantar & Alistarh, 2023) at a 0.8 sparsity ratio. In SciML, FARMS helps scientific models achieve up to a 5.66% error reduction during fine-tuning compared to the HT-SR based method TB_Sigmoid (Zhou et al., 2024), even at relatively low L2 relative error (L2RE) levels.

2. Related Work

2.1. Prior Work on HT-SR Theory

In this section, we provide an overview of prior work on HT-SR theory, a framework derived from RMT that is relevant to understanding modern, practical DNNs. HT-SR theory (Martin & Mahoney, 2021; Mahoney & Martin, 2019) originates from RMT but extends well beyond it. The theory was proposed based on the observation that well-trained, state-of-the-art DNNs often exhibit HTed structures in the ESD of each layer. In the meantime, several rigorous theories in SGD relating HT phenomena to generalization performance were established, providing further theoretical support for HT-SR theory (Hodgkinson & Mahoney, 2021; Hodgkinson et al., 2022; Gurbuzbalaban et al., 2021; Simsekli et al., 2019; 2020). Building on the theoretical foundation of HT-SR, a model analysis tool called WeightWatcher (Martin & Mahoney, 2021) was developed. Without accessing any training or test data, methods based on HT-SR Theory can be used to assess the training quality of models across various domains, such as CV and NLP (Martin & Mahoney, 2021; Martin et al., 2021; Yang et al., 2023; Liu et al., 2025b).

2.2. Optimization Methods Based on Eigenspectrum Analysis

Our paper is mainly motivated by HT-SR theory, focusing on the eigenspectrum analysis of weight matrices. Our paper connects to several DNN optimization methods that use eigenspectrum analysis to improve model performance. For example, spectral norm regularization has been applied to improve generalizability of trained models (Yoshida & Miyato, 2017; Miyato et al., 2018; Farnia et al., 2019). Stable rank normalization (SRN) (Sanyal et al., 2020), a normalization technique, scales matrices using their stable rank to enhance training stability in GANs and generalization in DNNs. TE-NAS (Chen et al., 2021) is a training-free neural architecture search framework that identifies high-performing networks by analyzing the neural tangent kernel spectrum and input space linear regions.

Although the aforementioned optimization methods based on eigenspectrum analysis have achieved a certain degree of model improvement, they do not provide fine-grained layer-wise optimization within DNNs. However, eigenspectrum analysis based on HT-SR theory offers a novel perspective by analyzing the shape of the HT ESDs, enabling a more precise estimate of the training quality for each layer. TempBalance (Zhou et al., 2024) performs eigenspectrum analysis on ESDs of the weight matrices of DNNs to assess the training progress of each layer. It enables a balanced adjustment of the learning rate across layers during training—an important parameter that functions as a "temperature-like" parameter within the language of statistical mechanics of learning. AlphaPruning (Lu et al., 2024), on the other hand, uses a similar weight analysis method to evaluate the training quality of each layer in LLMs. It then uses this information to strategically allocate layer-wise sparsity ratios to maintain model performance after pruning.

3. Method

In this section, we first discuss the motivation behind our method FARMS, which is to address the aspect ratio bias of typical HT-SR methods. Then, we describe FARMS in detail and show how FARMS can reduce the aspect ratio bias.

3.1. Typical HT-SR Analysis

Before introducing FARMS, we first review the commonly adopted procedures in HT-SR weight analysis. HT-SR analysis relies on estimating the layer quality based on the HT characteristic of the layer ESDs, which is quantified by HT metric PL_Alpha, introduced below. The ESD of a weight matrix is the histogram of eigenvalues. The ESD of weight matrices evolves during training, transitioning from a bulk-dominated regime to an HTed regime (Martin et al., 2021). The HTed portion can be modeled by a power-law (PL) distribution within an interval (λ_{min}, λ_{max}):

$$p(\lambda) \propto \lambda^{-\alpha}, \lambda_{min} < \lambda < \lambda_{max}, \tag{1}$$

where α, the PL exponent, is a critical metric for analyzing training quality. To fit a PL distribution to the ESD, methods in HT-SR often use the Hill Estimator (Hill, 1975; Zhou et al., 2024; Lu et al., 2024; Liu et al., 2024) [2]. For the i-th layer, suppose the weight matrix is $\mathbf{W}_i$ and the correlation matrix $\mathbf{W}_i^\top \mathbf{W}_i$ has ascending eigenvalues $\{\lambda_i\}_{i=1}^n$. The Hill estimator calculates PL_Alpha_Hill (the PL exponent PL_Alpha estimated using the Hill estimator) as:

$$\texttt{PL\_Alpha\_Hill} = 1 + \frac{k}{\sum_{i=1}^{k} \ln \frac{\lambda_{n-i+1}}{\lambda_{n-k}}}, \tag{2}$$

where k is an adjustable parameter. Changing k essentially changes the lower eigenvalue threshold λ_{min} for (truncated) PL estimation. Various metrics have been proposed to analyze the properties of ESDs, among which shape metrics, which characterize the distributional shape of ESDs, have been shown to effectively predict the training quality of individual layers (Yang et al., 2023). The PL_Alpha_Hill metric is one such shape metrics.

There are several metrics used to quantify the structure of ESDs in HT-SR theory. In this work, we mainly consider the PL_Alpha_Hill, which is empirically shown to be effective for training tasks (Zhou et al., 2024; Lu et al., 2024; Liu et al., 2024; Qing et al., 2024). In general, undertrained layers in DNNs tend to exhibit larger PL_Alpha_Hill values, whereas well-trained or over-trained layers typically have smaller PL_Alpha_Hill values. This metric can also be used for a comprehensive analysis across a series of DNN models with similar architectures to find the best model. Well-trained models tend to exhibit a lower average PL_Alpha_Hill across all layers (Martin et al., 2021; Yang et al., 2023). Moreover, in fine-tuning tasks, a larger amount of data generally results in a lower standard deviation (STD) of PL_Alpha_Hill across the model, indicating a more balanced training progression across different layers (Liu et al., 2024).

[2]One important point to note is that estimating PLs is inherently a challenging task (Clauset et al., 2009). Previous research suggests using the maximum likelihood estimate (Alstott et al., 2014; Martin & Mahoney, 2021). However, empirical evidence indicates that the Hill estimator performs more reliably in DNN optimization applications (Zhou et al., 2024; Lu et al., 2024; Liu et al., 2024).

3.2. Rationale of FARMS: Why Typical HT-SR Methods Are Insufficient

Here, we explain the rationale behind our method FARMS. In Section 3.1, we mentioned that layers exhibiting more HTed ESDs tend to be more well-trained. This observation is the key to using HT-SR methods to measure model quality. However, beyond the training quality of weights, the aspect ratio of a weight matrix also influences the heavytailness of its ESD. Specifically, in RMT, the Marchenko-Pastur (MP) distribution describes the limiting behavior of singular value distributions in large rectangular random matrices. According to the MP distribution, the ESD of the correlation matrix $\mathbf{W}^\top\mathbf{W}$ of an i.i.d. Gaussian random matrix $\mathbf{W}_{m\times n}$ becomes more concentrated as the aspect ratio m/n deviates from 1. An example is presented in Figure 2. Consequently, weight matrices with different aspect ratios naturally exhibit varying ESD shapes, leading to different degrees of heavy-tailness, which, if not analyzed carefully, can interfere with the quality assessment of model layers. We refer to this issue as the ***aspect ratio bias*** in HT-SR methods.

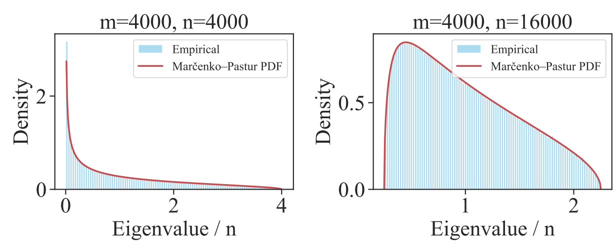


Figure 2. ESD shapes can be biased by the aspect ratio. Here we visualize the eigenvalues of the symmetric matrix $\frac{1}{n}\mathbf{X}_m\mathbf{X}_m^\top$. The left ESD is more HTed, while the right ESD is more concentrated. However, this change in the shape of the ESD is entirely due to the different aspect ratios of these two matrices and is not caused by differences in training quality. See detailed settings and supplementary results in the Figure 8 in Appendix B.

As another example, consider a well-trained layer with a large aspect ratio, such as the final layer of a ResNet 18 model trained on the CIFAR 100 dataset. This weight matrix has size 512×100 and a large aspect ratio of $512/100$. Thus, the ESD may not exhibit an obvious enough HT structure (See a related example in Figure 7 in Appendix A). If previous HT-SR methods are used, such layers may be quantified as poorly trained, but that conclusion results from the aspect ratio bias and is an inaccurate assessments of their training quality. Such misjudgment not only interferes with the evaluation of model performance but also leads to inaccurate tuning of hyperparameters in layer-wise optimization

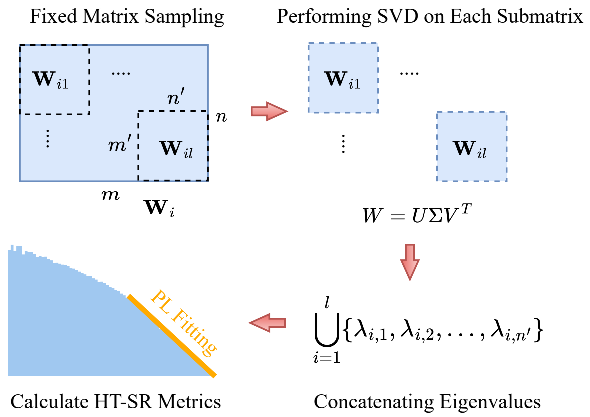


Figure 3. Main Steps in FARMS.

methods (Zhou et al., 2024; Lu et al., 2024), as these methods all assign different layer-wise hyperparameters based on the HT estimates. Therefore, eliminating such aspect ratio bias is crucial for improving HT-SR eigenspectrum analysis.

3.3. Analyzing ESDs Using FARMS

To mitigate the aspect ratio bias in analyzing ESDs, we use a block-wise sampling method when processing weight matrices. We partition each weight matrix into overlapping sub-matrices with a fixed aspect ratio (across all layers) following a predefined scheme described below.

Consider the weight matrix $\mathbf{W}_i$ of the i-th layer, which has a shape of $m \times n$. Without loss of generality, consider the case when $m \geq n$. Note that the shape $m \times n$ changes across layers. To process this weight matrix, we apply a sliding window approach to partition it into several equally sized submatrices (potentially with overlap), denoted as $\mathbf{W}_{i1}, \mathbf{W}_{i2}, \ldots, \mathbf{W}_{il}$, where l represents the number of submatrices. Each submatrix has a shape of $m' \times n'$ and satisfies a fixed aspect ratio $Q = m'/n'$. The parameters m', n', l, and Q are tunable hyperparameters (on which we will provide ablation studies in Section 4.6), allowing flexibility in the partitioning strategy. The key requirement is that even if the aspect ratio of the whole weight matrix m/n changes for different layer index i, that of the submatrices $Q = m'/n'$ is fixed across all layers.

Subsequently, for 2D Linear layers we compute the eigenvalues of the correlation matrices $\mathbf{X}_{ij} = \mathbf{W}_{ij}^\top \mathbf{W}_{ij}$ for each submatrix in the i-th layer. We then average the ESDs of $\mathbf{W}_{i1}, \mathbf{W}_{i2}, \ldots, \mathbf{W}_{il}$ (by merging their corresponding eigenvalue series) and measure the HT metrics of the averaged ESD instead. We merge the eigenvalue series because that's equivalent to averaging the empirical densities. The average of the ESDs leads to a less variant estimate of HT characteristics.

For CNN layers, since they have 4D tensor weight matrices with four dimensions $[C_1, C_2, k_H, k_W]$ (which represent input channels, output channels, height, and width, respectively), we employ a slightly different method for subsampling and measuring. We first flatten the two dimensions representing the convolution kernel size, $[k_H, k_W]$, in the 4D tensor into a single dimension. This results in $l' = k_H \times k_W$ two-dimensional matrices of shape $[C_1, C_2]$. We assume here that $C_1 \geq C_2$ for convenience. Then, for the 3D tensor of shape $l' \times C_1 \times C_2$, we perform subsampling of size $m' \times n'$ on each $C_1 \times C_2$ matrix corresponding to each l'. This results in $[\frac{C_1}{m'}] \times [\frac{C_2}{n'}] \times l'$ submatrices, each of size $m' \times n'$. Next, we average the ESDs by concatenating the rooted singular values for each submatrix along the l' dimension, and then measure the HT metrics of the ESDs. In this way, we obtain $[\frac{C_1}{m'}] \times [\frac{C_2}{n'}]$ HT metrics values. Finally, we take the average of these metrics values to obtain the HT metrics corresponding to the original 2D CNN. We also considered concatenating all the rooted singular values into a single list and calculating the corresponding ESD to measure the degree of heavy-tailedness. However, our experimental results in Table 7 in Appendix C.2 show that averaging the results separately leads to better model performance. The detailed procedures of FARMS are shown in Figure 3.

4. Empirical Results

In this section, we apply FARMS to measure HT metrics. To demonstrate the effectiveness of the new approach, we use these metrics across various optimization methods and tasks from different machine learning subfields.

In Section 4.1, we give full details of the experimental setup. In Section 4.2, we applied FARMS to LLM layer-wise pruning. In Section 4.3, we employ FARMS on training VGGs and ResNets on image classification tasks. In Section 4.4, we validate that FARMS achieves better performance in SciML fine-tuning experiments. In Section 4.5, we apply FARMS to analyze model quality, such as the balance of different layers. Finally, we perform ablation studies in Section 4.6.

4.1. Experimental Setup

Datasets. For image classification, we consider the CIFAR100 dataset (Krizhevsky, 2012). CIFAR100 consists of 50K pictures for the training set and 10K pictures for the testing set with 100 categories. For evaluating LLM pruning methods, we calculate model perplexity on the held-out WikiText (Merity et al., 2017) validation set and use seven tasks, including BoolQ (Clark et al., 2019), RTE (Wang et al., 2019), HellaSwag (Zellers et al., 2019),

Table 1. WikiText validation perplexity for pruned LLaMA-7B and LLaMA-13B models at different sparsity settings. Our method is compared to `AlphaPruning`, each paired with magnitude based pruning, Wanda, and SparseGPT. Lower perplexity indicates improved model performance. For calculating the standard deviation(STD) and demonstrating the stability of our method, we sample different calibration sets with 128 samples using six different seeds [0, 1, 2, 3, 4, 5] in Wanda and SparseGPT.

Sparsity Ratio	Layer-wise Method	LLaMA-7B Magnitude	LLaMA-7B Wanda	LLaMA-7B SparseGPT	LLaMA-13B Magnitude	LLaMA-13B Wanda	LLaMA-13B SparseGPT
0.7	AlphaPruning	231.76	24.30 $\pm$ 0.25	18.66 $\pm$ 0.49	2029.20	14.47 $\pm$ 0.08	13.29 $\pm$ 0.17
	Ours	**173.49**	**22.61** $\pm$ **0.18**	**18.53** $\pm$ **0.40**	**413.76**	**14.20** $\pm$ **0.09**	**13.06** $\pm$ **0.14**
0.75	AlphaPruning	2046.22	104.53 $\pm$ 4.49	36.52 $\pm$ 1.13	2710.49	32.18 $\pm$ 0.31	22.26 $\pm$ 0.59
	Ours	**1704.56**	**71.67** $\pm$ **1.71**	**35.47** $\pm$ **1.01**	**2634.82**	**29.56** $\pm$ **0.33**	**20.80** $\pm$ **0.43**
0.8	AlphaPruning	28865.67	772.20 $\pm$ 78.70	96.02 $\pm$ 1.59	5399.87	160.59 $\pm$ 4.05	47.57 $\pm$ 2.64
	Ours	**12799.58**	**504.58** $\pm$ **23.05**	**79.42** $\pm$ **3.86**	**5026.86**	**127.49** $\pm$ **2.12**	**41.44** $\pm$ **1.58**
0.85	AlphaPruning	71710.96	4609.70 $\pm$ 978.39	272.84 $\pm$ 30.84	38140.95	3144.01 $\pm$ 597.79	122.38 $\pm$ 8.88
	Ours	**66808.51**	**3595.54** $\pm$ **810.54**	**234.46** $\pm$ **16.42**	**37453.06**	**2847.85** $\pm$ **368.10**	**101.06** $\pm$ **4.48**

WinoGrande (Sakaguchi et al., 2021), ARC Easy and Challenge (Clark et al., 2018) and OpenbookQA (Mihaylov et al., 2018) for downstream zero-shot evaluation (Gao et al., 2021). For SciML, we fine-tune the models on simulated solutions of time-dependent PDE dataset 2D Compressible Navier-Stokes (CFD) [3] from PDEBench (Takamoto et al., 2022). All datasets considered in this paper are standard and widely studied.

Models. We consider different types of NNs in various research fields: VGG (Simonyan & Zisserman, 2015), ResNet (He et al., 2016), OPT (Zhang et al., 2022), LLaMA (Touvron et al., 2023a;b; Grattafiori et al., 2024) and DPOT (Hao et al., 2024). For VGG series, we consider VGG16 and VGG19. For ResNet series, we consider ResNet18 and ResNet34. For OPT series, we consider four different model size: OPT-125M/350M/1.3B/6.7B. For LLaMA series, we consider six different moodels: LLaMA-7B/13B, LLaMA-V2-7B/13B and LLaMA-V3-3B/8B. For DPOT series, we consider DPOT-Tiny and DPOT-Small.

Baseline. We considered several model diagnostic and layer-wise hyperparameter scheduling methods as baselines for comparison. `TempBalance` (Zhou et al., 2024) is a layer-wise learning rate allocation algorithm based on HT-SR theory, designed to analyze the training progress of each layer and allocate learning rates accordingly. `TempBalance` measures the HT metrics of all layers, and it assigns a larger learning rate to weight matrices with a more lighted-tailed ESD. `AlphaPruning` (Lu et al., 2024), on the other hand, is a layer-wise pruning method for LLMs based on HT-SR. It assigns a larger pruning ratio to Transformer weight matrices with a more lighted-tailed ESD. Additionally, we compared our method with the `TB_Sigmoid` (Liu et al., 2024) approach, which was applied to experiments on SciML models. All three optimization methods adopt ESD analysis techniques, which are susceptible to aspect ratio bias because the HT metrics are measured on the whole weight matrix. Therefore, one can replace the HT measurement procedures in these methods with `FARMS`. The goal is to evaluate the improvement of `FARMS` compared to the existing HT-SR analysis used in these optimization methods.

[3] CFD means compressible fluid dynamics or, equivalently, the compressible Navier-Stokes equations.

4.2. Improving LLM Pruning with FARMS

To explore the effectiveness of `FARMS`, we conduct experiments in the task of LLM pruning. As we have mentioned, we can replace the HT measurement procedures in `AlphaPruning` with `FARMS`. Therefore, we can compare `AlphaPruning` with `FARMS` and without. We make the comparison with different sparsity ratios in the range {0.7, 0.75, 0.8, 0.85} and different LLM pruning methods, including Magnitude-based (Han et al., 2015), SparseGPT (Frantar & Alistarh, 2023) and Wanda (Sun et al., 2024). We use different LLM pruning methods because `AlphaPruning` assigns only layer-wise ratios, not specific pruning locations, and thus can be applied to various LLM pruning methods. We followed the experimental setup from Lu et al. (2024) and our hyperparameter ranges are reported in Table 14 of Appendix F.

Language Modeling. The results in Table 1 illustrate that using `FARMS` helps `AlphaPruning` achieve better performance under different sparsity ratios and LLM pruning methods. Specifically, when using the Magnitude method, our approach reduces the perplexity of LLaMA-7B from 231.76 to 173.49 at a sparsity ratio of 0.7. Similarly, when employing more advanced pruning methods such as Wanda and SparseGPT, our approach further reduces the perplexity of LLaMA-7B from 96.02 to 79.42 and LLaMA-13B from 47.57 to 41.44 at a sparsity ratio of 0.8. One notable advantage of `AlphaPruning` is that it allows pruning LLMs

Table 2. Comparison of mean zero-shot accuracies (%) for pruned LLaMA-7B and LLaMA-13B models at different sparsity settings. We evaluate our method against `AlphaPruning`, each integrated with magnitude-based pruning, Wanda, and SparseGPT. Higher accuracy values indicate better zero-shot ability.

Sparsity Ratio	Layer-wise Method	LLaMA-7B			LLaMA-13B		
		Magnitude	Wanda	SparseGPT	Magnitude	Wanda	SparseGPT
0.7	AlphaPruning	35.67	43.67	44.79	38.23	47.46	49.07
	Ours	**35.96**	**44.26**	**44.84**	**38.87**	**47.75**	**49.61**
0.75	AlphaPruning	34.59	37.99	40.89	**38.16**	**42.73**	44.17
	Ours	**35.88**	**39.66**	**40.93**	37.68	42.66	**44.47**
0.8	AlphaPruning	33.79	34.06	36.63	35.59	38.42	39.07
	Ours	**34.33**	**35.76**	**37.50**	**36.94**	**39.23**	**40.18**
0.85	AlphaPruning	33.29	31.69	34.86	33.11	32.05	36.73
	Ours	**34.27**	**33.09**	**35.25**	**33.29**	**32.41**	**37.04**

to a sparsity ratio of 0.8 without significantly impacting perplexity. With `FARMS`, this performance can be further improved. We can also find that `FARMS` achieves lower perplexity STD in most settings. In Table 8 and Table 9 in Appendix C, we provide additional experiment results on OPT series models and more recent LLaMA models.

Zero-Shot Tasks. We test the zero-shot ability of pruned LLMs on seven zero-shot downstream tasks mentioned in Section 4.1 with prompting. Results are shown in Table 2, where we report the mean zero-shot accuracy on seven zero-shot tasks of pruned LLaMA-7B and LLaMA-13B models. `FARMS` can outperform `AlphaPruning` in improving accuracy across most settings. For example, although `AlphaPruning` achieved significant improvements over the uniform method, `FARMS` further improved accuracy by values of 1.11% in SparseGPT pruning over `AlphaPruning` at a sparsity ratio of 0.8. We report the detailed performance for each individual task in Table 10 and Table 11 in Appendix C.

4.3. Improving Image Classification with `FARMS`

In this subsection, we compare `FARMS` to `TempBalance` on image classification tasks. As we mentioned, `TempBalance` is a layer-wise learning rate assignment method using HT metrics. In Figure 4, we report the evaluation results of training the ResNet and VGG series models under different optimizing settings. Again, we compare `TempBalance` with or without `FARMS`. “Ours” means replacing the method to measure heavytailness previously employed in the `TempBalance` with `FARMS`.

An important configuration used in the `TempBalance` method is the exclusion of certain layers from the learning rate assignment process. Instead of receiving layer-wise learning rate adjustments as in `TempBalance`, these layers are assigned a global learning rate that decays following the cosine annealing schedule. These excluded layers have “tall-and-skinny matrices,” which are matrices with a large aspect ratio and a relatively small number of eigenvalues. For example, the final layer of the ResNet 18 model has dimensions of 512×100, resulting in a large aspect ratio of $512/100 = 5.12$. This kind of extreme size makes it difficult for previous methods to accurately estimate the heavytailness of the ESDs. This issue further exacerbates the inability of the previous `TempBalance` method to accurately allocate per-layer learning rates, resulting in certain layers remaining at either excessively high or extremely low learning rates. We illustrate this phenomenon in more detail in Figure 12 of Appendix C. Consequently, this imbalance leads to suboptimal model performance after training.

Therefore, according to the experimental setup described in `TempBalance`, the first and last layers of ResNet and VGG models, along with certain layers whose correlation matrix of weight matrix contains a relatively small number of eigenvalues, will be excluded from the layer-wise learning rate scheduling. To ensure a comprehensive study, we compare `FARMS` and `TempBalance` with and without this “layer selection” (LS) heuristic. See Figure 4, when the LS heuristic is not applied (and all layers will be included in the adjustable learning rate scheduling), the performance of the original `TempBalance` method (shown as “TB (no LS)”) deteriorates significantly. For example, compared to the LS-enabled setup (shown as “LS+TB”), the test accuracy of the VGG 19 model trained on CIFAR100 without LS using the `TempBalance` method drops from 74.19% to 73.22%, while the standard deviation increases from 0.159 to 0.277. In contrast, when training models using the `TempBalance` method with `FARMS`, performance remains stable or even improves when LS is disabled. This observation demonstrates that our method is more robust to weight matrices of extreme sizes, and the reason is that our method provides a more accurate assessment of the HTed ESDs independent of the matrix size.

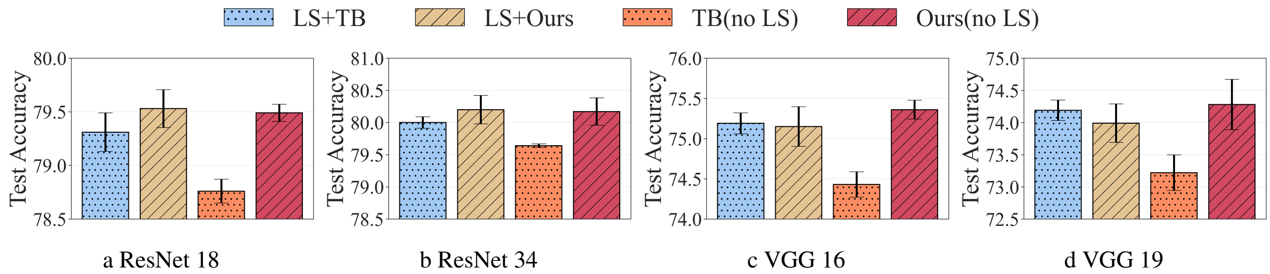


Figure 4. Comparing our method to `TempBalance` at different layer usage settings in training ResNet and VGG series models on CIFAR100. Higher test accuracy values indicate better model performance. See Table 13 in Appendix F for the details in all hyperparameters.

In `TempBalance` algorithm, the layer-wise adjustable learning rate is scaled in the range of $(s_1\eta_t, s_2\eta_t)$, where η_t is the global learning rate at time t, and s_1 and s_2 are two tunable lower and upper limits of learning rates. To demonstrate that `FARMS` can more accurately evaluate the training quality of each model layer, we present the test performance of models trained using the different learning rate scaling ratios (s_1, s_2). in which we consider five different settings for (s_1, s_2): [(0.1, 1.9), (0.2, 1.8), (0.3, 1.7), (0.4, 1.6), (0.5, 1.5)]. Additionally, we consider that these five scaling ratios are generally close to the optimal scaling ratio. We run tasks on CIFAR100 with four VGG and ResNet architectures. Our results in Table 3 show that `FARMS` can improve the test accuracy of models among the five scaling ratios and help models achieve a lower overall standard deviation compared to `TempBalance`. Detailed results can be found in Figure 9, Appendix C.

Table 3. Comparing our method to `TempBalance` among the five scaling ratios. All results are reported as the mean test accuracy and standard deviation obtained across five different scaling ratios. In this experiment, we allowed `TempBalance` baseline to use the LS heuristic to achieve slightly better test accuracy.

Method	**ResNet 18**	**ResNet 34**	**VGG 16**	**VGG 19**
TB	$79.03_{\pm 0.169}$	$79.81_{\pm 0.145}$	$74.87_{\pm 0.214}$	$73.89_{\pm 0.199}$
Ours	$\mathbf{79.35}_{\pm 0.126}$	$\mathbf{80.07}_{\pm 0.097}$	$\mathbf{75.16}_{\pm 0.212}$	$\mathbf{74.03}_{\pm 0.163}$

4.4. Improving SciML Fine-tuning with FARMS

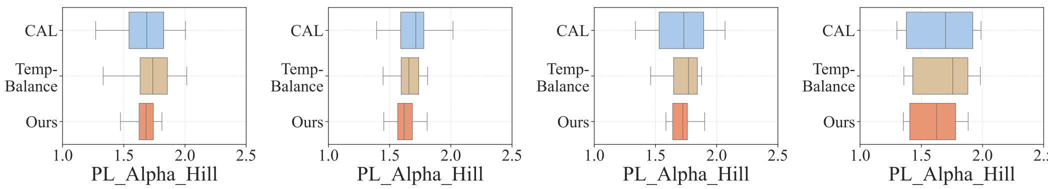
To explore the potential applications of `FARMS` in multiple domains of machine learning research, we also performed SciML fine-tuning experiments using the 2DCFD dataset with DPOT-Tiny and DPOT-Small models and compared `FARMS` with the `TB_Sigmoid` method (Liu et al., 2024). We followed the experimental setup from Liu et al. (2024), and our hyperparameter ranges are detailed in Table 14 of Appendix F. In Table 4, we show the results of comparing `FARMS` with the `TB_Sigmoid` (shown as "TB_Sig") with layer selection and the baseline fine-tuning (shown as "FT") with a uniform learning rate for each layer in different data subsampling ratios in the range {5%, 10%, 25%, 50%, 100%}. Our method consistently outperformed `TB_Sigmoid` across all subsampling ratios. For instance, with a full dataset (100% of the data), our approach reduced the model's L2RE to 1.017e-2. Comparing to the `TB_Sigmoid`, the L2RE decreased by 5.66%.

Table 4. `FARMS` achieves lower L2RE(↓) on the test dataset than `TB_Sigmoid` and baseline fine-tuning method on SciML tasks.

Subsampling Ratio	**Method**	**DPOT-Tiny**	**DPOT-Small**
5%	FT	1.863e-2	1.546e-2
	TB_Sig	1.856e-2	1.539e-2
	Ours	**1.842e-2**	**1.536e-2**
10%	FT	1.747e-2	1.426e-2
	TB_Sig	1.730e-2	1.415e-2
	Ours	**1.706e-2**	**1.407e-2**
25%	FT	1.543e-2	1.226e-2
	TB_Sig	1.517e-2	1.203e-2
	Ours	**1.499e-2**	**1.189e-2**
50%	FT	1.309e-2	1.025e-2
	TB_Sig	1.283e-2	1.005e-2
	Ours	**1.242e-2**	**9.822e-3**
100%	FT	1.096e-2	8.400e-3
	TB_Sig	1.078e-2	8.193e-3
	Ours	**1.017e-2**	**7.949e-3**

4.5. HT Metrics Analysis

In this subsection, we demonstrate that `FARMS` can effectively control the shape of ESDs, resulting in a more balanced distribution of `PL_Alpha_Hill` values among the layers of NNs compared to the previous methods. We assess the models presented in Section 4.3, which include 2D Convolution and 2D Linear layers of ResNets and VGG



a ResNet 18, Cifar100 b ResNet 34, Cifar100 c VGG 16, Cifar100 d VGG 19, Cifar100

Figure 5. Comparing the distribution of `PL_Alpha_Hill` of ResNet and VGG series models. The top blue-shaded section in each subplot of the experiments (shown as "CAL") represents models trained without employing layer-wise optimization methods. The middle brown-shaded section in each subplot indicates models trained using `TempBalance`; Lastly, the bottom orange-shaded section in each subplot corresponds to the results obtained using `FARMS`. Figures 5a, 5b, 5c and 5d present the averaged results obtained from multiple experiments within the optimal parameter range.

networks. Results in Figure 5 show the distribution of `PL_Alpha_Hill` of models trained using baseline cosine annealing (CAL), `TempBalance`, and `TempBalance` using our method `FARMS`. It can be seen that `FARMS` generally leads to a more concentrated distribution and, in most cases, reduces the average `PL_Alpha_Hill` value. These experimental results suggest that our method `FARMS` enables a more balanced training progression across different layers of the model. These observations align well with the findings reported by Martin et al. (2021); Liu et al. (2024), which suggest that a decrease in both average `PL_Alpha_Hill` values and the variance across layers indicate better training quality.

4.6. Ablation Study

Different Aspect Ratios of Weight Matrix. Here, we study the effect of the aspect ratio of the subsampled matrices in `FARMS`. We consider image classification tasks and use `FARMS` to replace the HT metrics in `TempBalance` for assigning layer-wise learning rates. We consider five different aspect ratios in the range $\{0.5, 0.75, 1.0, 1.5, 2.0\}$ and keep other hyperparameters optimal. We again use ResNet18 and VGG16 as our architectures and show results on CIFAR100. Results in Table 5 show that `FARMS` achieves a higher test accuracy when the aspect ratio is 1.0.

Table 5. Comparing the different Aspect ratios settings for submatrices on ResNet 18 and VGG 16 models trained on CIFAR100 Dataset.

Q Ratio	0.5	0.75	**1.0**	1.5	2.0
ResNet 18	$79.13_{\pm 0.158}$	$79.33_{\pm 0.122}$	$\mathbf{79.53_{\pm 0.177}}$	$79.13_{\pm 0.282}$	$79.31_{\pm 0.046}$
VGG 16	$75.05_{\pm 0.285}$	$75.19_{\pm 0.517}$	$\mathbf{75.36_{\pm 0.118}}$	$75.21_{\pm 0.345}$	$75.10_{\pm 0.360}$

Subsampling Window Sizes and Sampling Steps. Here, we study different subsampling window sizes and sampling steps (i.e., the number of subsampled submatrices) in the task of LLaMA-7B pruning. The aspect ratio Q is fixed to be 1.0. For all weight matrices in LLaMA-7B, the smallest dimension is 4096. Therefore, when using a window size of 4096, sliding window downsampling is applied only along the larger dimension. This process generates multiple 4096×4096 submatrices. We report our results in Table 6. We find that using an appropriately sized sampling window, such as 2000×2000, the model achieves a lower perplexity. As a comparison, baseline perplexity achieved by `AlphaPruning` without using `FARMS` is 96.02.

Table 6. Different Window Size and Sampling Steps for pruning the LLaMA-7B at 0.8 sparsity ratio. The pruning method is SparseGPT. The model is evaluated on WikiText dataset with perplexity (↓).

	Sampling Steps			
Window size	5	10	15	20
500	$96.34_{\pm 9.15}$	$95.93_{\pm 2.80}$	$99.23_{\pm 3.53}$	$88.08_{\pm 4.23}$
1000	$84.71_{\pm 4.97}$	$81.96_{\pm 4.22}$	$89.20_{\pm 3.91}$	$84.04_{\pm 2.48}$
2000	$82.33_{\pm 3.18}$	$\mathbf{79.42_{\pm 3.86}}$	$80.14_{\pm 2.32}$	$86.54_{\pm 3.28}$
4096	$82.63_{\pm 2.45}$	$89.61_{\pm 5.25}$	$84.19_{\pm 6.05}$	$84.07_{\pm 5.64}$

5. Conclusion

We propose a subsampling strategy to address the measurement bias introduced by varying aspect ratios of weight matrices in HT-SR eigenspectrum analysis. The main idea of our method is to extract submatrices of the original matrix with a fixed aspect ratio, followed by averaging the ESDs of these submatrices to accurately assess the training quality of each weight matrix. Our extensive experiments demonstrate that our method can precisely estimate layer-wise training quality, improve the performance of layer-wise optimization algorithms, and contribute to more balanced training and efficient pruning. Furthermore, results on visual analytics confirm the effectiveness of our approach, showing that it successfully eliminates the measurement bias caused by aspect ratio variations.

Impact Statement

This paper presents research aimed at advancing the fields of Machine Learning and Deep Learning, particularly in training using information from eigenspectrum analysis. While our work has various potential societal implications, we do not find it necessary to highlight any specific ones here.

Acknowledgments

This work is supported by DOE under Award Number DE-SC0025584 and Dartmouth College.

References

Adlam, B. and Pennington, J. The neural tangent kernel in high dimensions: Triple descent and a multi-scale theory of generalization. In *International Conference on Machine Learning*, pp. 74–84. PMLR, 2020.

Alstott, J., Bullmore, E., and Plenz, D. powerlaw: a python package for analysis of heavy-tailed distributions. *PloS one*, 9(1):e85777, 2014.

Ba, J., Erdogdu, M. A., Suzuki, T., Wang, Z., Wu, D., and Yang, G. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. *Advances in Neural Information Processing Systems*, 35: 37932–37946, 2022.

Bai, Z. and Silverstein, J. W. *Spectral analysis of large dimensional random matrices*, volume 20. Springer, 2010.

Chen, W., Gong, X., and Wang, Z. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. In *International Conference on Learning Representations (ICLR)*, 2021.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Clauset, A., Shalizi, C. R., and Newman, M. E. Power-law distributions in empirical data. *SIAM review*, 51(4): 661–703, 2009.

Couillet, R. and Liao, Z. *Random Matrix Methods for Machine Learning*. Cambridge University Press, 2022.

Derezinski, M., Liang, F. T., and Mahoney, M. W. Exact expressions for double descent and implicit regularization via surrogate random design. *Advances in neural information processing systems*, 33:5152–5164, 2020.

Dobriban, E. and Wager, S. High-dimensional asymptotics of prediction: Ridge regression and classification. *The Annals of Statistics*, 46(1):247–279, 2018.

Farnia, F., Zhang, J., and Tse, D. Generalizable adversarial training via spectral normalization. In *International Conference on Learning Representations*, 2019.

Frantar, E. and Alistarh, D. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pp. 10323–10337. PMLR, 2023.

Gao, L., Tow, J., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., McDonell, K., Muennighoff, N., et al. A framework for few-shot language model evaluation. *Version v0. 0.1. Sept*, 10:8–9, 2021.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Guhr, T., Müller–Groeling, A., and Weidenmüller, H. A. Random-matrix theories in quantum physics: common concepts. *Physics Reports*, 299(4):189–425, June 1998. ISSN 0370-1573. doi: 10.1016/S0370-1573(97)00088-4.

Gurbuzbalaban, M., Simsekli, U., and Zhu, L. The heavy-tail phenomenon in sgd. In *International Conference on Machine Learning*, pp. 3964–3975. PMLR, 2021.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

Hao, Z., Su, C., Liu, S., Berner, J., Ying, C., Su, H., Anandkumar, A., Song, J., and Zhu, J. Dpot: Auto-regressive denoising operator transformer for large-scale pde pre-training. In *International Conference on Machine Learning*, pp. 17616–17635. PMLR, 2024.

Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. Surprises in high-dimensional ridgeless least squares interpolation. *Annals of statistics*, 50(2):949, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hill, B. M. A simple general approach to inference about the tail of a distribution. *The annals of statistics*, pp. 1163–1174, 1975.

Hodgkinson, L. and Mahoney, M. Multiplicative noise and heavy tails in stochastic optimization. In *International Conference on Machine Learning*, pp. 4262–4274. PMLR, 2021.

Hodgkinson, L., Simsekli, U., Khanna, R., and Mahoney, M. Generalization bounds using lower tail exponents in stochastic optimizers. In *International Conference on Machine Learning*, pp. 8774–8795. PMLR, 2022.

Hu, H. and Lu, Y. M. Universality laws for high-dimensional learning with random features. *IEEE Transactions on Information Theory*, 69(3):1932–1964, 2022.

Kothapalli, V., Pang, T., Deng, S., Liu, Z., and Yang, Y. From spikes to heavy tails: Unveiling the spectral evolution of neural networks. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Krizhevsky, A. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Liu, L., Zhao, X., Li, G., Li, D., Wang, Han, Y., Li, X., and Wang, Y. BaWA: Automatic optimizing pruning metric for large language models with balanced weight and activation. In *Proceedings of the 42nd International Conference on Machine Learning*, ICML '25. PMLR, 2025a.

Liu, Z., Hu, Y., Pang, T., Zhou, Y., Ren, P., and Yang, Y. Model balancing helps low-data training and fine-tuning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1311–1331, 2024.

Liu, Z., Pang, T., Balabanov, O., Yang, C., Huang, T., Yin, L., Yang, Y., and Liu, S. Lift the veil for the truth: Principal weights emerge after rank reduction for reasoning-focused supervised fine-tuning. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, Vancouver, Canada, July 2025b.

Lu, H., Zhou, Y., Liu, S., Wang, Z., Mahoney, M. W., and Yang, Y. Alphapruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models. *Advances in neural information processing systems*, 37:9117–9152, 2024.

Mahoney, M. and Martin, C. Traditional and heavy tailed self regularization in neural network models. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4284–4293. PMLR, 09–15 Jun 2019.

Martin, C. H. and Mahoney, M. W. Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning. *Journal of Machine Learning Research*, 22(165):1–73, 2021.

Martin, C. H., Peng, T., and Mahoney, M. W. Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data. *Nature Communications*, 12(1), July 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-24025-8.

Mei, S. and Montanari, A. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, 75(4):667–766, 2022.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks, 2018.

Nadakuditi, R. R. and Newman, M. E. Graph spectra and the detectability of community structure in networks. *Physical review letters*, 108(18):188701, 2012.

Pennington, J. and Worah, P. Nonlinear random matrix theory for deep learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Qing, P., Gao, C., Zhou, Y., Diao, X., Yang, Y., and Vosoughi, S. Alphalora: Assigning lora experts based on layer training quality. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20511–20523, 2024.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sanyal, A., Torr, P. H., and Dokania, P. K. Stable rank normalization for improved generalization in neural networks and gans. In *International Conference on Learning Representations*, 2020.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR 2015)*. Computational and Biological Learning Society, 2015.

Simsekli, U., Sagun, L., and Gurbuzbalaban, M. A tail-index analysis of stochastic gradient noise in deep neural networks. In *International Conference on Machine Learning*, pp. 5827–5837. PMLR, 2019.

Simsekli, U., Sener, O., Deligiannidis, G., and Erdogdu, M. A. Hausdorff dimension, heavy tails, and generalization in neural networks. *Advances in Neural Information Processing Systems*, 33:5138–5151, 2020.

Sun, M., Liu, Z., Bair, A., and Kolter, J. Z. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

Takamoto, M., Praditia, T., Leiteritz, R., MacKinlay, D., Alesiani, F., Pflüger, D., and Niepert, M. Pdebench: An extensive benchmark for scientific machine learning. *Advances in Neural Information Processing Systems*, 35: 1596–1611, 2022.

Tao, T. *Topics in random matrix theory*, volume 132. American Mathematical Soc., 2012.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Tulino, A. M. and Verdú, S. Random Matrix Theory and Wireless Communications. *Foundations and Trends® in Communications and Information Theory*, 1(1):1–182, June 2004. ISSN 1567-2190, 1567-2328. doi: 10.1561/0100000001. Publisher: Now Publishers, Inc.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

Wang, Z., Engel, A., Sarwate, A. D., Dumitriu, I., and Chiang, T. Spectral evolution and invariance in linear-width neural networks. *Advances in neural information processing systems*, 36:20695–20728, 2023.

Yang, G. and Hu, E. J. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*, pp. 11727–11737. PMLR, 2021.

Yang, G., Hu, E., Babuschkin, I., Sidor, S., Liu, X., Farhi, D., Ryder, N., Pachocki, J., Chen, W., and Gao, J. Tuning large neural networks via zero-shot hyperparameter transfer. *Advances in Neural Information Processing Systems*, 34:17084–17097, 2021.

Yang, Y., Theisen, R., Hodgkinson, L., Gonzalez, J. E., Ramchandran, K., Martin, C. H., and Mahoney, M. W. Test accuracy vs. generalization gap: Model selection in nlp without accessing training or testing data. KDD '23, pp. 3011–3021, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701030. doi: 10.1145/3580305.3599518.

Yin, L., Wu, Y., Zhang, Z., Hsieh, C.-Y., Wang, Y., Jia, Y., Li, G., Jaiswal, A. K., Pechenizkiy, M., Liang, Y., et al. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning llms to high sparsity. In *International Conference on Machine Learning*, pp. 57101–57115. PMLR, 2024.

Yoshida, Y. and Miyato, T. Spectral norm regularization for improving the generalizability of deep learning, 2017.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zhou, Y., Pang, T., Liu, K., Mahoney, M. W., Yang, Y., et al. Temperature balancing, layer-wise weight analysis, and neural network training. *Advances in Neural Information Processing Systems*, 36, 2024.

Appendix

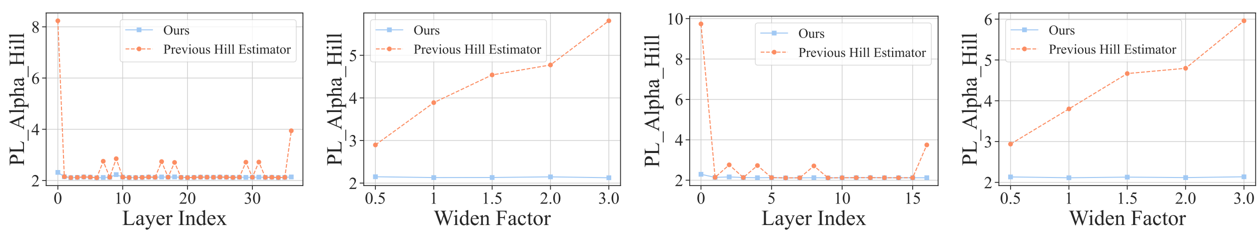
A. Detailed Matrix Analysis

A.1. Random Initialization

In this section, we do weight analysis for ResNet 34 and VGG 19 models that are randomly initialized using methods proposed in He et al. (2015). The weights w are drawn from a normal distribution with mean 0 and variance $2/n_{in}$, where n_{in} represents the number of input neurons. This is represented as:

$$w \sim \mathcal{N}\left(0, \frac{2}{n_{in}}\right)$$

We present our results of comparing FARMS with the previous HT estimation methods in measuring randomly initialized models in Figure 6. In Figure 6a and Figure 6c, all layers in these models should be similarly under-trained (because they are all just initialized). However, previous methods tend to report significantly higher values for PL_Alpha_Hill in certain layers, suggesting that these layers are much more under-trained. In contrast, FARMS produces results where the training quality of all initialized layers is nearly identical, aligning more closely with the expected result. Similarly, in Figure 6b and Figure 6d, for randomly initialized ResNet-34 and VGG-19 models with varying widths, the previous method measures PL_Alpha_Hill values that increase as model width increases. In contrast, FARMS shows PL_Alpha_Hill values that remain unaffected by the aspect ratio bias introduced by model width variations.



a ResNet 34, All Layers   b ResNet 34, Different Width   c VGG 19, All Layers   d VGG 19, Different Width

Figure 6. Comparing FARMS and previous HT-SR methods for measuring the randomly initialized ResNet 34 and VGG 19 weights. Figure 6a and Figure 6c show the PL_Alpha_Hill values for each layer in models (widen factor is 1.0) by using different methods. Figure 6b and Figure 6d show the measured PL_Alpha_Hill values of the final linear layer across different model width factors. As the width increases, the aspect ratio of the weight matrix also becomes larger, leading to bias in previous HT-SR methods.

A.2. Mitigating Aspect Ratio Bias in Specific Layers

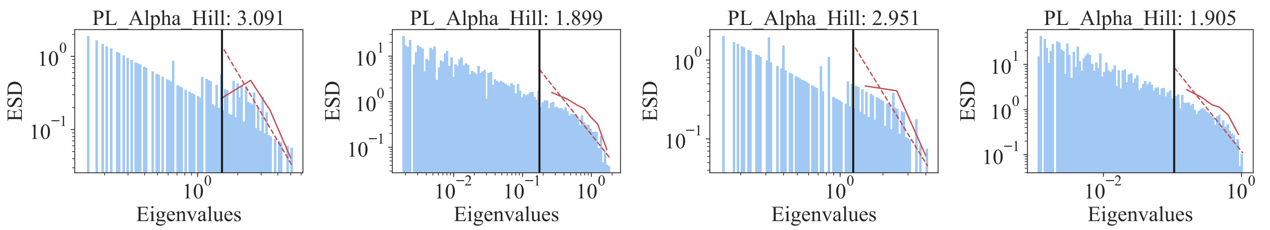
In this section, we do weight analysis for the last layer from the ResNet 34 and VGG 16 models trained on CIFAR100. These models are trained with hyperparameters according to the Appendix F. The weight matrices of the final layers in both models have dimensions of 512×100 (aspect ratio = 5.12). This means that the previous weight matrix analysis method will be significantly affected by aspect ratio bias, leading to inaccurate assessments of the layer's training quality.

In Figure 7, the presented experimental results validate this observation. Figure 7a and Figure 7c show the ESD fitting results obtained using the previous weight analysis method, where the measured PL_Alpha_Hill values are larger than those typically observed in well-trained layers. This leads to the erroneous classification of these layers as poorly trained, ultimately affecting both model performance evaluation and optimization. In contrast, Figure 7b and Figure 7d present the ESD fitting results obtained using FARMS. Our method produces measurements that align more closely with the expected ESD of well-trained layers, providing a more accurate assessment of training quality.

B. Marchenko–Pastur Distribution

In RMT, the Marchenko-Pastur distribution, also known as the Marchenko-Pastur law, characterizes the asymptotic behavior of singular values of large rectangular random matrices.

Let X_{ij}, $1 \le i \le m$, $1 \le j \le n$, be independent random variables with $\mathbb{E}X_{ij} = 0$ and $\mathbb{E}X_{ij}^2 = 1$, and $\mathbf{X}_m =$



a Previous Method, ResNet 34 b `FARMS`, ResNet 34 c Previous Method, VGG 16 d `FARMS`, VGG 16

Figure 7. Comparing `FARMS` and the previous method for measuring the ESD of final layers in ResNet 34 and VGG 16 trained on CIFAR 100.

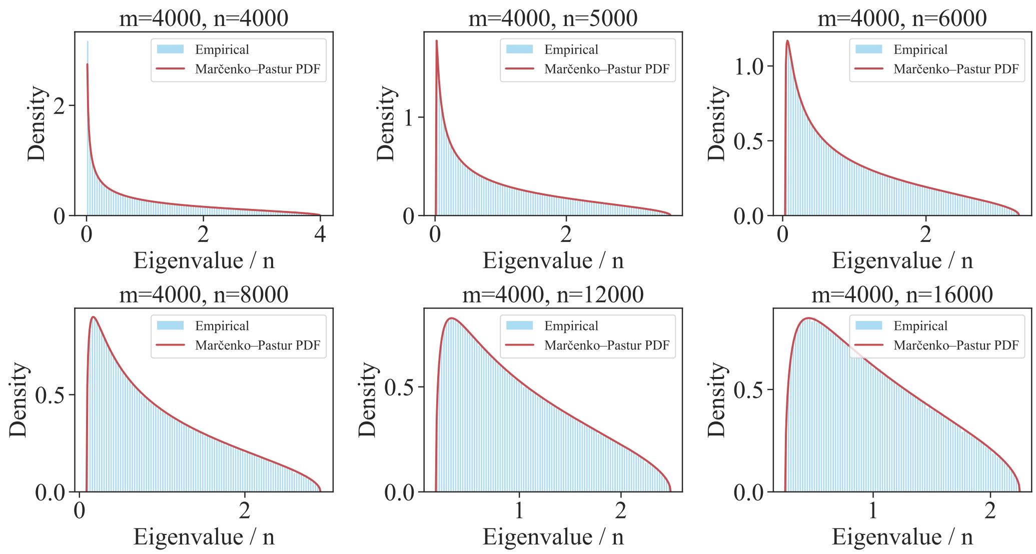


Figure 8. The Marchenko-Pastur (MP) Law for different values of n. The ESD (blue histogram) of the eigenvalues of the sample covariance matrix is compared with the theoretical Marchenko-Pastur probability density function (red curve) for various aspect ratios, where $m = 4000$ is fixed, and n varies from 4000 to 16000.

$(X_{ij})_{1\le i\le m, 1\le j\le n}$. Denote by $\lambda_1 \le \cdots \le \lambda_m$ the eigenvalues of the symmetric matrix[4]

$$\mathbf{W} := \mathbf{W}_m := \frac{1}{n}\mathbf{X}_m\mathbf{X}_m^\top$$

and defined its empirical distribution by

$$F_m(x) = \frac{1}{m}\sum_{k=1}^{m} I_{\{\lambda_k \le x\}},$$

where $I_{\{B\}}$ denotes the indicator of an event B. One often investigates the rate of convergence of the expected spectral distribution $\mathbb{E}F_m(x)$ as well as $F_m(x)$ to the Marchenko–Pastur distribution function $F_y(x)$ with density

$$f_y(x) = \frac{1}{2xy\pi}\sqrt{(b-x)(x-a)}I_{\{[a,b]\}}(x) + I_{\{[1,\infty)\}}(y)(1-y^{-1})\delta(x),$$

where $y = m/n$, $y \in (0,\infty)$ and $a = (1-\sqrt{y})^2$, $b = (1+\sqrt{y})^2$. Here we denote by $\delta(x)$ the Dirac delta function and by $I_{\{[a,b]\}}(x)$ the indicator function of the interval $[a, b]$.

[4]**Note.** The transpose operation applies to the second (right-hand) matrix in this expression, which differs from the transpose placement used in HT-SR Theory, shown as $\mathbf{X}_{ij} = \mathbf{W}_{ij}^\top\mathbf{W}_{ij}$ in Section 3.3. Thus, the "large aspect ratio" regime discussed in HT-SR Theory corresponds to a large value of the ratio n/m in Appendix B.

We visualize the MP distribution and the ESD of the weight matrices in Figure 8. In this figure, we can observe that the empirical distribution converges to the theoretical MP distribution. Additionally, as n increases, the ESD distribution exhibits an increasingly concentrated shape with a reduced degree of HT. Therefore, ignoring the impact of aspect ratio and directly measuring the HT degree of the ESD to estimate a layer's training quality may lead to inaccurate results.

C. Additional Experiment Results

C.1. Detailed Performance on Different Scaling Ratios

We provide detailed results of a hyperparameter study on learning rate scaling ratio (s_1, s_2). We set other hyperparameters (including layer selection, initial learning rate, and so on) as the optimal value for each training set. The results in Figure 9 show that across all tested architectures and most scaling ratios, `FARMS` consistently outperforms `TempBalance`. This demonstrates that `FARMS` provides a more accurate measurement of each layer's training quality. As a result, it helps the model achieve better training performance when using different learning rate scaling ratios.

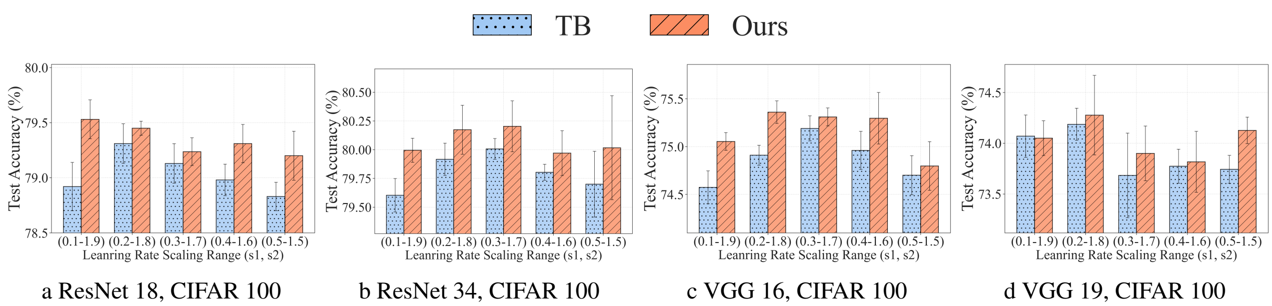


Figure 9. Comparison of test accuracy across different architectures and learning rate scaling ranges. The figure presents the test accuracy (%) of `TempBalance` (blue, dotted) and `FARMS` (orange, hatched). These two methods are evaluated on the CIFAR 100 dataset using ResNet and VGG series architectures. The x-axis represents different learning rate scaling ranges (s_1, s_2), while the y-axis indicates test accuracy. Error bars denote standard deviations across multiple runs.

C.2. Comparison of Different CNN Processing Methods

We compared two different methods for processing the ESD of CNNs. The first method concatenates the rooted singular values of all submatrices and then measures the HT metrics of the resulting ESD. The second method measures the HT metrics of the ESD formed by concatenating the rooted singular values of each of the $C_1 \times C_2$ submatrices separately, and then averages these HT metrics from each group. In Table 7, we provide a performance comparison of training ResNet-18 and VGG-16 models using these two methods. For each experiment, all other hyperparameters are set to their optimal values. We find that although the performance difference between the two methods is not very large, the second method brings more performance improvement.

Table 7. Comparing Two Stratgy in CNN Processing with `FARMS`.

Method	ResNet 18	VGG 16
Calculating the Overall ESD	79.36$\pm$0.101	75.27$\pm$0.201
Averaging ESDs Computed from Subsets	**79.53$\pm$0.177**	**75.36$\pm$0.118**

C.3. Experiment Results for Additional Models

We prune the OPT models (Zhang et al., 2022) and LLaMA-V2/V3 models (Touvron et al., 2023b; Grattafiori et al., 2024) with `AlphaPruning` and compare the model performance with `FARMS`. We report the additional experiment results on LLM pruning in Table 8 and Table 9. For the OPT series models, we use four sparsity ratios: {0.6, 0.7, 0.75, 0.8}. For the LLaMA series models, we select {0.7, 0.75, 0.8, 0.85} as the sparsity ratios. Experimental results show that our method can help `AlphaPruning` achieve better model performance on models such as OPT.

Table 8. WikiText validation perplexity for pruned OPT models at different sparsity settings. Our method is compared to `AlphaPruning`, each paired with Wanda and SparseGPT. Lower perplexity indicates improved model performance.

Sparsity Ratio	Layer-wise Method	OPT-125M		OPT-350M		OPT-1.3B		OPT-6.7B	
		Wanda	SparseGPT	Wanda	SparseGPT	Wanda	SparseGPT	Wanda	SparseGPT
0.6	AlphaPruning	67.12	59.18	87.95	50.94	**27.09**	22.76	15.6	13.71
	Ours	**66.71**	**58.86**	**82.55**	**50.52**	27.15	**22.69**	**15.59**	**13.67**
0.7	AlphaPruning	263.71	196.64	595.26	147.20	101.13	50.30	44.92	20.76
	Ours	**261.84**	**188.31**	**489.75**	**136.80**	**100.74**	**50.01**	**42.03**	**20.76**
0.75	AlphaPruning	**718.69**	515.55	1453.21	330.11	613.78	142.68	245.16	35.27
	Ours	721.52	**491.98**	**1298.22**	**303.07**	**601.24**	**122.01**	**181.92**	**34.88**
0.8	AlphaPruning	1713.45	1525.25	2869.8	921.17	2763.33	511.66	5781.0	98.3
	Ours	**1609.36**	**1412.59**	**2551.46**	**798.25**	**2502.60**	**499.82**	**5212.51**	**90.88**

Table 9. WikiText validation perplexity for pruned LLaMA-V2 and LLaMA-V3 models at different sparsity settings. Our method is compared to `AlphaPruning`, each paired with Wanda and SparseGPT. Lower perplexity indicates improved model performance.

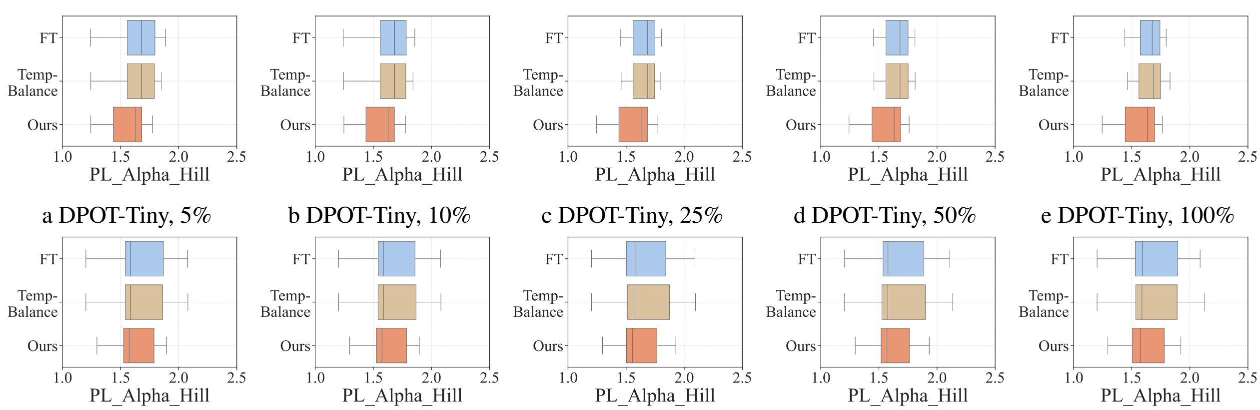
Sparsity Ratio	Layer-wise Method	LLaMA-V2-7B		LLaMA-V2-13B		LLaMA-V3.2-3B		LLaMA-V3.1-8B	
		Wanda	SparseGPT	Wanda	SparseGPT	Wanda	SparseGPT	Wanda	SparseGPT
0.7	AlphaPruning	34.71	20.92	15.37	13.69	123.19	64.33	**105.64**	38.43
	Ours	**31.09**	**20.49**	**15.36**	**13.64**	**120.03**	**63.78**	107.00	**37.81**
0.75	AlphaPruning	170.74	41.39	35.26	23.21	356.69	121.53	240.17	**81.30**
	Ours	**161.58**	**39.98**	**33.67**	**22.98**	**311.73**	**117.19**	**229.63**	81.93
0.8	AlphaPruning	868.66	89.73	150.01	46.32	1412.53	262.44	688.23	180.40
	Ours	**831.37**	**88.25**	**127.69**	**44.74**	**1219.10**	**212.65**	**607.58**	**175.93**
0.85	AlphaPruning	6425.25	217.79	874.97	107.21	5857.39	505.8	**3498.94**	**400.69**
	Ours	**4437.46**	**204.44**	**748.91**	**98.99**	**4407.94**	**463.34**	3766.13	411.35

C.4. Zero-shot Tasks Performance

We demonstrate the task-wise performance in detail in Table 10 and Table 11.

C.5. Detailed `PL_Alpha_Hill` Distribution

We provide additional experimental results on LLM pruning and SciML fine-tuning to support the `PL_Alpha_Hill` distribution analysis presented in Section 4.5. The results in Figure 10 and 11 show that `FARMS` consistently helps models achieve better layer quality across different experimental settings.



a DPOT-Tiny, 5% b DPOT-Tiny, 10% c DPOT-Tiny, 25% d DPOT-Tiny, 50% e DPOT-Tiny, 100%

f DPOT-Small, 5% g DPOT-Small, 10% h DPOT-Small, 25% i DPOT-Small, 50% j DPOT-Small, 100%

Figure 10. Comparing the distribution of PL Alpha Hill of DPOT-Tiny and DPOT-Small in different fine-tuning methods and data ratios.

Table 10. Accuracies (%) of LLaMA-7B for 7 zero-shot tasks with unstructured sparsity from 70% to 85%. We compare `FARMS` with uniform pruning ratios and `AlphaPruning` using Magnitude-based pruning, Wanda and SparseGPT.

Sparsity Ratio	Method	BoolQ	RTE	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Mean
0.7	Magnitude	38.29	52.71	25.59	51.22	26.73	19.62	11.60	32.25
	AlphaPruning $w.$ Magnitude	**41.31**	**52.71**	30.37	51.46	**35.44**	22.01	16.40	35.67
	Ours $w.$ Magnitude	40.09	52.71	**30.49**	**53.83**	34.22	**23.21**	**17.20**	**35.96**
	Wanda	56.06	55.60	28.90	50.99	32.11	18.26	13.60	36.50
	AlphaPruning $w.$ Wanda	64.34	57.40	35.57	**61.09**	**45.58**	24.49	17.20	43.67
	Ours $w.$ Wanda	**64.40**	**59.93**	**36.54**	60.62	45.24	**24.66**	**18.40**	**44.26**
	SparseGPT	65.14	**53.79**	33.95	58.72	44.44	23.98	17.20	42.46
	AlphaPruning $w.$ SparseGPT	**65.74**	53.07	**37.72**	64.01	47.35	**26.88**	18.80	44.79
	Ours $w.$ SparseGPT	65.54	53.07	36.85	**64.33**	**48.11**	26.37	**19.60**	**44.84**
0.75	Magnitude	42.26	52.35	25.88	48.54	26.68	**21.42**	14.00	33.02
	AlphaPruning $w.$ Magnitude	50.67	52.71	26.80	**49.96**	27.19	21.42	13.40	34.59
	Ours $w.$ Magnitude	**55.54**	**53.07**	**27.80**	49.25	**29.21**	21.08	**15.20**	**35.88**
	Wanda	37.83	53.79	27.01	49.96	27.74	19.37	12.60	32.61
	AlphaPruning $w.$ Wanda	62.17	53.43	29.52	53.75	33.04	20.82	13.20	37.99
	Ours $w.$ Wanda	**62.17**	**58.12**	**30.99**	**55.49**	**35.69**	**21.76**	**13.40**	**39.66**
	SparseGPT	62.14	**53.43**	29.76	51.22	33.80	19.11	13.40	37.55
	AlphaPruning $w.$ SparseGPT	63.71	52.71	**33.11**	**59.91**	**38.68**	**23.89**	14.20	40.89
	Ours $w.$ SparseGPT	**64.07**	53.07	32.74	59.83	38.43	23.55	**14.80**	**40.93**
0.8	Magnitude	**48.81**	49.10	25.59	48.78	24.92	22.01	**14.20**	33.34
	AlphaPruning $w.$ Magnitude	44.40	53.07	**26.17**	50.75	**25.67**	22.27	14.20	33.79
	Ours $w.$ Magnitude	46.76	**53.79**	26.08	**52.33**	24.58	**22.95**	13.80	**34.33**
	Wanda	61.69	51.42	25.85	**50.24**	26.23	**20.62**	**14.00**	35.72
	AlphaPruning $w.$ Wanda	52.75	51.62	26.53	48.54	26.81	20.39	11.80	34.06
	Ours $w.$ Wanda	**62.14**	**51.62**	**26.81**	50.20	**27.57**	20.56	11.40	**35.76**
	SparseGPT	43.55	52.71	27.87	48.86	29.34	18.34	13.40	33.44
	AlphaPruning $w.$ SparseGPT	61.62	52.35	28.29	52.25	**30.64**	19.28	12.00	36.63
	Ours $w.$ SparseGPT	**62.26**	**53.43**	**29.29**	**54.22**	30.18	**19.71**	**13.40**	**37.50**
0.85	Magnitude	48.75	51.62	25.58	48.46	**25.76**	**22.61**	**14.60**	33.91
	AlphaPruning $w.$ Magnitude	43.18	**52.71**	**25.87**	49.49	25.00	22.35	14.40	33.29
	Ours $w.$ Magnitude	**56.64**	48.01	25.60	**49.49**	25.08	22.44	12.60	**34.27**
	Wanda	**51.74**	45.85	**26.12**	47.59	25.88	20.48	**15.00**	**33.24**
	AlphaPruning $w.$ Wanda	38.23	47.29	25.87	50.12	25.76	**22.18**	12.40	31.69
	Ours $w.$ Wanda	49.91	**48.38**	25.92	**50.12**	**26.09**	20.22	11.00	33.09
	SparseGPT	37.89	**53.07**	26.69	50.36	**26.98**	19.28	11.40	32.24
	AlphaPruning $w.$ SparseGPT	**57.61**	52.35	27.08	48.70	26.64	19.45	12.20	34.86
	Ours $w.$ SparseGPT	55.69	51.62	**27.19**	**52.88**	26.98	**19.97**	**12.40**	**35.25**

C.6. Layer-wise Visualization over Training

We provide visualization to compare how the `PL_Alpha_Hill` and learning rates are distributed over layers during the training with different layer-wise optimization settings. In Figure 12, we report the learning rate and `PL_Alpha_Hill` every five epochs throughout the 200-epoch training duration. We can find that in Figure 12a, 12e and Figure 12b, 12f, if the previous `TempBalance` does not apply LS to exclude layers with aspect ratio bias (e.g., the first and last layers of the model), the layer-wise learning rate allocation will be inaccurate. This misallocation leads to a few layers having excessively large learning rates while most layers have very small ones. This ultimately results in poor model performance or an imbalanced training process. In contrast, `FARMS` (see results in Figures 12c, 12d, 12g and 12h) effectively eliminates the measurement inaccuracies caused by aspect ratio bias. As a result, it ensures that learning rates are properly allocated regardless of whether LS is applied.

C.7. Computation Cost Analysis

In Table 12, we report the computational cost of eigenspectrum analysis across different models and methods. The most computation-intensive aspect of these layer analysis methods involves performing SVD on weight matrices, which can be optimized using parallel processing techniques. In the LLM pruning task, we use 8 L40 GPUs for weight analysis and record the `PL_Alpha_Hill` values for each layer. We select four different sparsity ratios, three pruning methods, and two evaluation approaches, yielding a total of 24 experimental configurations per model. However, since weight analysis is

Table 11. Accuracies (%) of LLaMA-13B for 7 zero-shot tasks with unstructured sparsity from 70% to 85%. We compare `FARMS` with uniform pruning ratios and `AlphaPruning` using Magnitude-based pruning, Wanda and SparseGPT.

Sparsity Ratio	Method	BoolQ	RTE	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Mean
	Magnitude	52.87	**50.54**	26.57	50.83	28.45	20.56	14.80	34.95
	AlphaPruning w. Magnitude	**61.28**	46.93	30.24	50.43	31.23	26.28	21.20	38.23
	Ours w. Magnitude	57.16	43.32	**33.90**	**53.43**	**34.97**	**26.88**	**22.40**	**38.87**
	Wanda	62.08	52.71	30.38	52.41	40.53	17.49	16.00	38.80
0.7	AlphaPruning w. Wanda	63.43	**53.43**	**42.16**	65.59	57.53	**28.67**	21.40	47.46
	Ours w. Wanda	**66.18**	52.71	41.94	**64.80**	**57.87**	28.58	**22.20**	**47.75**
	SparseGPT	69.17	52.71	37.25	63.22	51.98	24.83	21.60	45.82
	AlphaPruning w. SparseGPT	72.05	54.15	**42.44**	68.35	55.60	28.50	**22.40**	49.07
	Ours w. SparseGPT	**73.03**	**54.51**	41.98	**69.06**	**57.45**	**29.27**	22.00	**49.61**
	Magnitude	45.05	**50.90**	25.93	50.43	25.93	20.82	13.60	33.24
	AlphaPruning w. Magnitude	60.03	50.90	**30.10**	**52.88**	28.20	**25.60**	19.40	**38.16**
	Ours w. Magnitude	**60.06**	44.04	28.29	52.57	**32.53**	23.89	**22.40**	37.68
	Wanda	42.51	52.71	27.88	49.64	29.17	17.83	12.00	33.11
0.75	AlphaPruning w. Wanda	62.17	52.71	**36.09**	**62.67**	44.15	23.29	**18.00**	**42.73**
	Ours w. Wanda	**62.42**	**52.71**	35.41	61.33	**46.93**	**23.46**	16.40	42.66
	SparseGPT	62.69	52.71	31.66	57.06	40.74	20.56	15.00	40.06
	AlphaPruning w. SparseGPT	64.19	**53.07**	**37.44**	**64.88**	45.79	**26.45**	17.40	44.17
	Ours w. SparseGPT	**65.69**	52.71	36.89	64.72	**47.43**	26.02	**17.80**	**44.47**
	Magnitude	38.50	**53.43**	25.95	48.86	25.51	22.10	13.40	32.53
	AlphaPruning w. Magnitude	53.30	51.26	26.67	51.70	26.60	23.38	16.20	35.59
	Ours w. Magnitude	**59.79**	48.38	**28.32**	**54.38**	**27.10**	**24.23**	**16.40**	**36.94**
	Wanda	37.86	52.71	26.64	48.30	27.15	19.97	13.20	32.26
0.8	AlphaPruning w. Wanda	62.17	52.71	29.59	55.09	34.93	20.22	14.20	38.42
	Ours w. Wanda	**62.17**	**52.71**	**30.45**	**57.38**	**36.41**	**21.08**	**14.40**	**39.23**
	SparseGPT	60.80	52.71	28.71	50.28	30.22	18.26	13.00	36.28
	AlphaPruning w. SparseGPT	62.20	52.71	31.46	57.62	35.48	19.62	14.40	39.07
	Ours w. SparseGPT	**62.20**	**52.71**	**32.37**	**59.19**	**37.16**	**22.18**	**15.40**	**40.18**
	Magnitude	38.29	**54.51**	**25.95**	49.96	25.21	**23.12**	14.00	33.01
	AlphaPruning w. Magnitude	39.11	51.26	25.47	**50.83**	**26.39**	22.10	**16.60**	33.11
	Ours w. Magnitude	**41.22**	51.62	25.73	50.43	26.01	22.01	16.00	**33.29**
	Wanda	37.83	52.71	26.09	47.91	25.88	**21.16**	**12.80**	32.05
0.85	AlphaPruning w. Wanda	37.83	52.71	**26.72**	**49.80**	26.09	19.62	11.60	32.05
	Ours w. Wanda	**38.65**	**53.79**	26.31	48.46	**26.77**	20.31	12.60	**32.41**
	SparseGPT	38.41	52.71	27.31	50.04	26.30	17.66	12.60	32.15
	AlphaPruning w. SparseGPT	**62.17**	52.71	**28.64**	51.70	29.67	**18.86**	**13.40**	36.73
	Ours w. SparseGPT	62.17	**52.71**	28.59	**54.14**	**30.09**	18.77	12.80	**37.04**

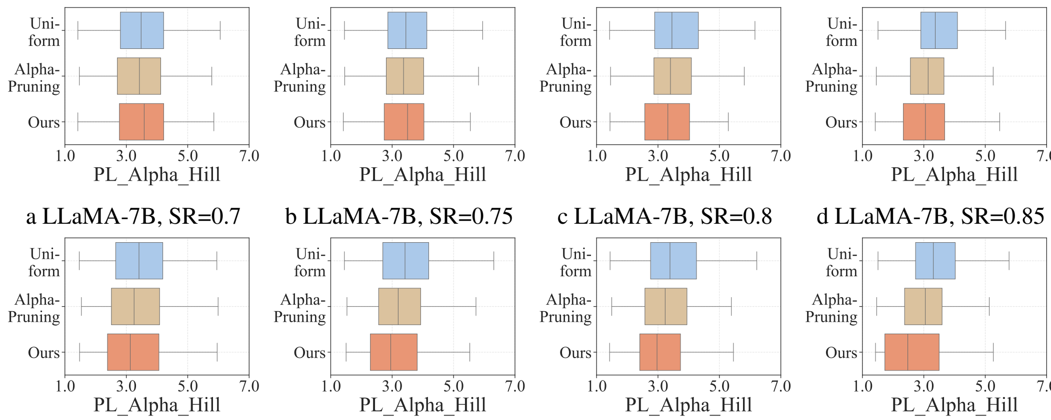
performed only once per model, the additional computational cost per experiment remains minimal. When the number of submatrices becomes particularly large, the additional computational overhead introduced by `FARMS` increases accordingly. In CV and SciML experiments, we use a single L40 GPU and do weight analysis every epoch during the training and fine-tuning. Compared to previous methods, the additional computational cost introduced by `FARMS` is not particularly significant.

We acknowledge that computational cost can indeed be higher than previous methods. But there are certain ways to mitigate the issue, e.g., by using a larger window size (like 4096 for LLaMA 7B/5120 for LLaMA 13B) and a limited number of sampling steps. For example, in Table 6, when we use larger window size(such as 4096) and smaller sampling steps(such as 5), our method can still improve model performance and reduce the computational cost.

C.8. LLM Pruning Stability Analysis

The experiment results from (Yin et al., 2024) show that layer-wise pruning LLMs at high sparsity is a challenging optimization problem. If sparsity ratios are not properly allocated, the performance of the model can become very unstable. Therefore, here we aim to demonstrate that the performance improvements observed in our experiments are due to more accurate layer-wise analysis, rather than accidental factors such as random seeds.

We follow the experiment settings from Table 6 and visualize the sparsity ratio assigned to each transformer block in



a LLaMA-7B, SR=0.7 b LLaMA-7B, SR=0.75 c LLaMA-7B, SR=0.8 d LLaMA-7B, SR=0.85

e LLaMA-13B, SR=0.7 f LLaMA-13B, SR=0.75 g LLaMA-13B, SR=0.8 h LLaMA-13B, SR=0.85

Figure 11. Comparing the distribution of PL Alpha Hill of LLaMA-7B and LLaMA-13B in different layer-wise strategies and sparsity ratios (shown as "SR"). The pruning method is SparseGPT.

Table 12. Computation cost for each experiment.

Model	Experiment Settings	Weight Analysis Time (sec/experiment)
LLaMA-7B	AlphaPruning	2.65
	Ours (4096, 5)	6.08
	Ours (2000, 15×15)	134.5
LLaMA-13B	AlphaPruning	6.23
	Ours (5120, 5)	16.65
	Ours (2000, 15×15)	163.26

Model	Method	Weight Analysis Time (sec/epoch)
ResNet 18	TB	0.887
	Ours	1.054
ResNet 34	TB	1.761
	Ours	1.946
VGG 16	TB	1.048
	Ours	1.200
VGG 19	TB	1.421
	Ours	1.578
DPOT-Tiny	TB_Sig	0.235
	Ours	0.278
DPOT-Small	TB_Sig	1.255
	Ours	1.340

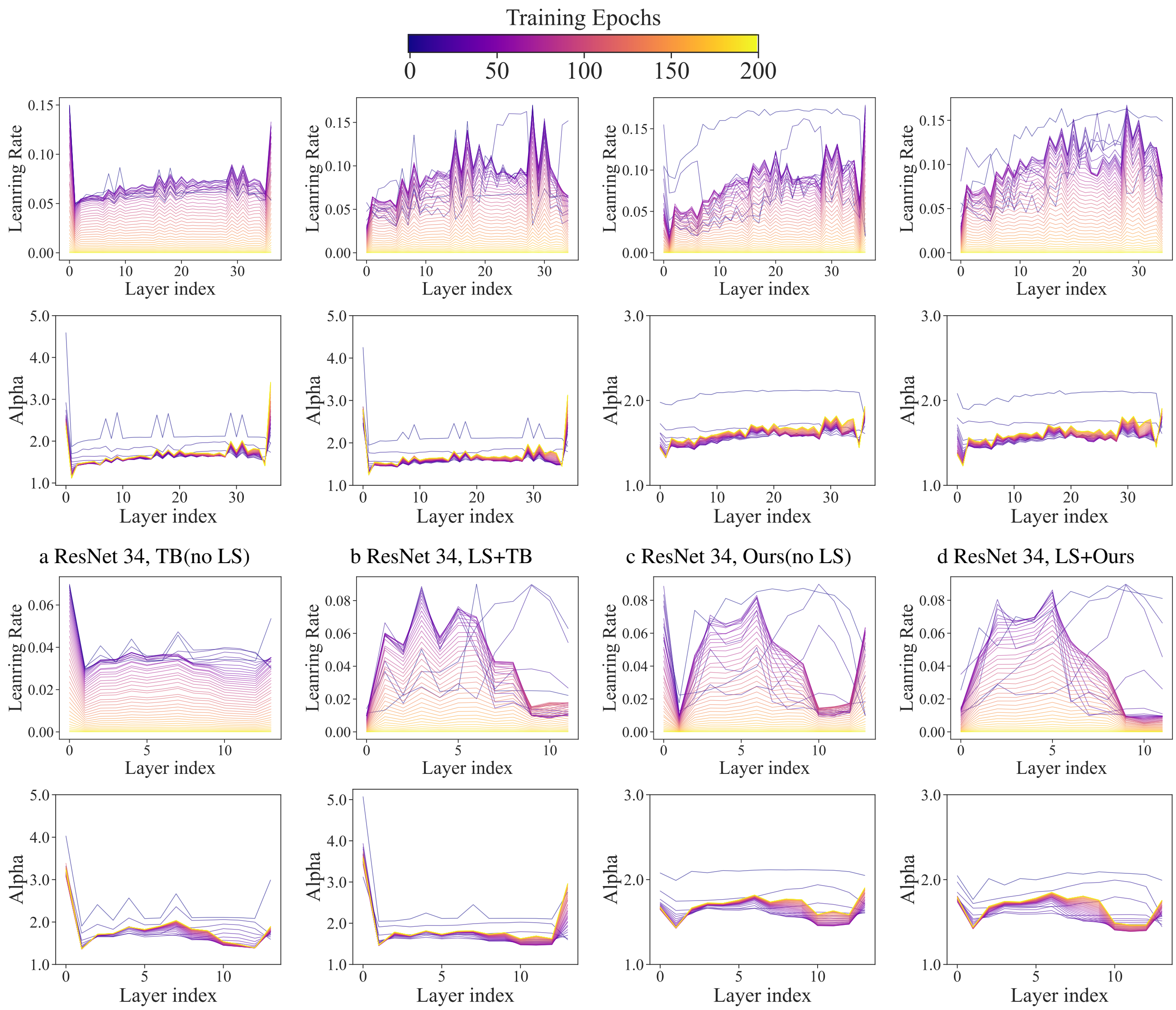
LLaMA-7B under different sampling settings in Figure 13. This visualization indirectly reflects the degree of heavy-tailedness in the ESD of each transformer block's model layer as analyzed by our method. We observe that the differences in sparsity ratio assignments across these settings are not significant. However, the assignment from the best FARMS setting is still distinct from the worst setting, which explains the performance difference in Table 6.

D. Training Quality Analysis

D.1. Previous work on training quality

Previous work on HT-SR has established that the heavy-tailness of ESDs is strongly correlated with the test accuracy of models (Martin & Mahoney, 2021; Martin et al., 2021). While this does not imply that "training quality" is identical to "test accuracy," the correlation between heavy-tailedness and test accuracy has been used to justify HT-SR metrics. Therefore, improving test accuracy or similar performance metrics (e.g., perplexity) remains our primary goal.

Although previous work on HT-SR does not explicitly define "training quality," several related quantities have been mentioned: (1) strong correlation between weight elements (Martin & Mahoney, 2021; Martin et al., 2021) and (2) feature spikes and their alignment with the target teacher model (Ba et al., 2022; Wang et al., 2023). The feature spike, analyzed in the context of a single-index teacher and a two-layer student model (Ba et al., 2022; Wang et al., 2023), is approximately a rank-one update to the model (in the limit of infinite matrix size with fixed aspect ratio) and also persists after matrix subsampling. This is because the specific form of the rank-one update makes it cover the whole matrix with probability one.



a ResNet 34, TB(no LS) b ResNet 34, LS+TB c ResNet 34, Ours(no LS) d ResNet 34, LS+Ours

e VGG 16, TB(no LS) f VGG 16, LS+TB g VGG 16, Ours(no LS) h VGG 16, LS+Ours

Figure 12. Visualization of layer-wise Learning Rate and `PL_Alpha_Hill` (Alpha) over training. The top rows of each subfigure presents the evolution of layer-wise learning rates and the bottom rows presents `PL_Alpha_Hill` during training for different optimizing configurations of ResNet 34 and VGG 16 on CIFAR 100. The color gradient represents training epochs, transitioning from dark purple (early epochs) to yellow (later epochs).

D.2. A toy experiment to measure training quality

We designed a toy experiment to test the correlation between "training quality" and the new HT-SR metric measured using FARMS. Following (Ba et al., 2022; Wang et al., 2023), we use a single-index teacher to generate signals to train a two-layer student. The first layer of the student model is a weight matrix, while the second layer is a weight vector. We only update the weight of first layer. To measure "training quality", during training, we measure the alignment between the weight matrix and the ground truth target vector of the teacher model similar to (Ba et al., 2022; Wang et al., 2023), and we define this alignment to be the "training quality" of the student model.

Throughout the training process, we select the student network checkpoint with the highest alignment and report both the alignment value and the `PL_Alpha_Hill` value. We then vary the sizes of the student model with different weight matrix aspect ratios on a fixed input dimension 500 to conduct multiple experiments. Each experiment provides one

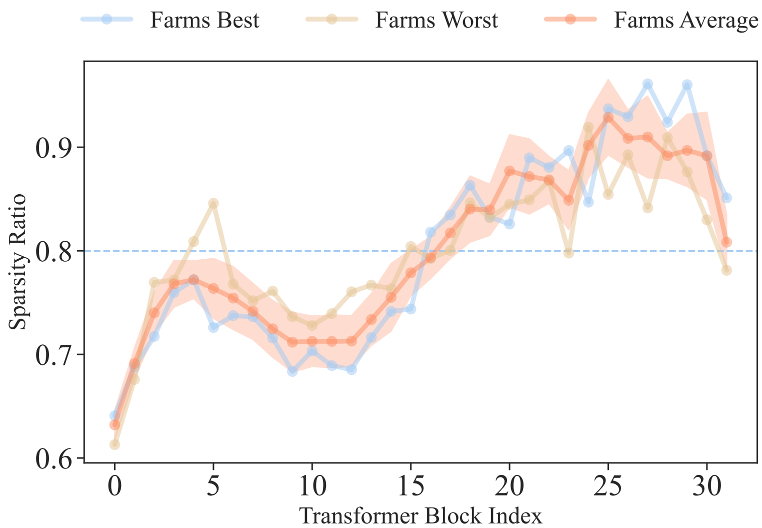


Figure 13. Block-wise Sparsity Ratio for LLaMA-7B assigned by FARMS. We set up FARMS with four different window sizes {500, 1000, 2000, 4096} and four sampling setps {5, 10, 15, 20}. The pruning method is SparseGPT and the base sparsity ratio is 0.8. The FARMS Best indicates using this sparsity ratio distribution makes minimal perplexity **79.42**±**3.86** while the FARMS worst makes largest perplexity **99.23**±**3.53**. The distribution of block-wise sparsity ratios of FARMS Average represents the mean under the 4×4 settings, with the standard deviation intervals also illustrated.

`PL_Alpha_Hill` value and one alignment value. The multiple experiments (conducted using varying sizes) produce one curve for `PL_Alpha_Hill` and one curve for the alignment value.

We then plot the two curves using both existing methods for estimating `PL_Alpha_Hill` and our method FARMS. As shown in Figure 14, FARMS reveals a clear negative correlation between the two curves: the better the training quality, the larger the alignment, and the smaller the `PL_Alpha_Hill`. However, for the existing method, due to the aspect ratio bias, the correlation is incorrect.

E. Broader Discussion

In this section, we discuss more related literature as well as some explanations regarding aspect ratio bias and issues related to our method.

E.1. Relationship with Matrix Shape

One should recognize that the numerical value of the same property measured in different network layers can naturally have different scales, and thus should not be directly compared without proper normalization. In some real-world application scenarios, such as in LLM Pruning (Lu et al., 2024; Liu et al., 2025a), this factor need to be considered. When the network size changes, hyperparameters also need to be changed accordingly. For example, works such as Tensor Programs IV (Yang & Hu, 2021) and Tensor Programs V (Yang et al., 2021) have explored how layer sizes affect the optimal scaling of weight initializations and learning rates. Tensor Programs IV introduced the Maximal Update Parametrization (μP) to ensure feature learning by carefully choosing parameter scaling rules based on these sizes, addressing how existing parametrizations might otherwise collapse to kernel regimes. Building on this, Tensor Programs V demonstrated that μP can enable zero-shot hyperparameter transfer, where hyperparameters tuned on smaller models remain optimal for significantly larger ones. In this context, "shape" awareness pertains to designing size-dependent hyperparameters that maintain stable and effective (non-kernel) training dynamics.

Nevertheless, our work is fine-grained in that we consider the aspect ratios of different layers (number of rows versus columns). We demonstrate that varying aspect ratios can artificially stretch or compress the ESD. This distortion confounds the interpretation of heavy-tailed (HT) metrics. To mitigate this measurement bias, we propose `FARMS` to analyze the average ESD of submatrices sampled at a fixed aspect ratio. This approach provides a normalized HT metric, enabling more reliable comparisons of such spectral diagnostics across different layers within a given network. Thus, while focusing on shape-aware parametrization for training stability and transfer, our contribution lies in a shape-aware analysis technique (FARMS) aimed at correcting measurement bias in spectral diagnostics.

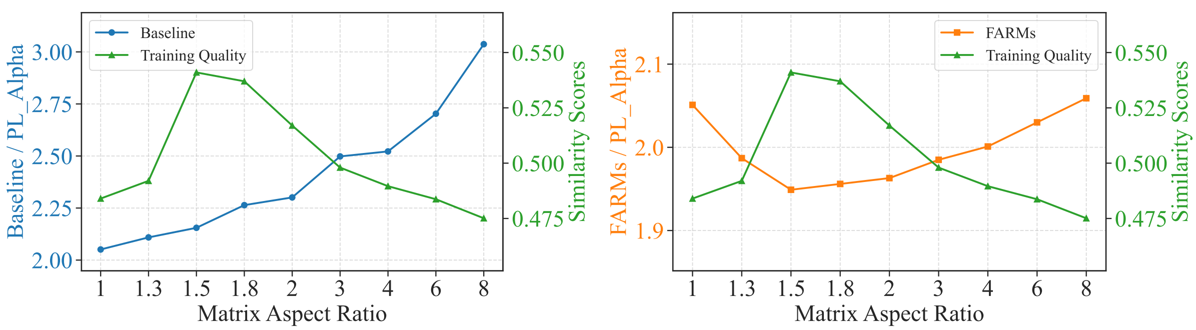
E.2. Why does the FARMS preserve important spectral property about the original matrix?

The goal of measuring heavy-tailness in HT-SR is to evaluate the strength of correlations introduced by training, as established in previous work (Martin & Mahoney, 2021). However, when we subsample a single submatrix and measure correlations only within that submatrix, some correlations between elements in the subsampled matrix and those outside it are inevitably lost. This motivates our approach of using multiple submatrices to capture a broader range of correlations.

E.3. Does this approach introduce additional bias?

Our approach could be viewed as introducing a form of "bias"; however, we interpret this more specifically as achieving partial coverage of the entire matrix. Conceptually, this is similar to the principle behind bootstrap sampling in random forests, where multiple samples, each with potentially limited coverage, are used collectively to mitigate the effects of this partial view and improve overall model robustness.

Further justification for this perspective comes from recent work (Wang et al., 2023; Kothapalli et al., 2025) that aims to theoretically quantify heavy-tailedness. These studies interpret heavy-tailedness as the accumulation and evolution of feature spikes in the ESD that align with the teacher model's features. Critically, these feature spikes are characterized as being approximately rank-one updates to the original matrix Because a rank-one component inherently covers the whole matrix, sampling a submatrix will, with high probability, capture that rank-one component. Therefore, this subsampling process is unlikely to miss the feature spikes, which are identified by previous work as the cause of the heavy-tail structure. We believe this provides substantial evidence that FARMS can preserve important spectral information, specifically as measured by these feature spikes in the ESD.



a Baseline

b FARMS

Figure 14. Compare the PL_Alpha_Hill from FARMS and Baseline in measuring the training quality of a single layer. The Correlation Coefficient between FARMS and training quality is -0.89 and for baseline is -0.51. We can find that FARMS can measure the training quality more precisely.

F. Hyperparameter Adjustment

In this section, we report the hyperparameters that we use in the experiments shown in the main paper (Section 4).

First, we report the common hyperparameters shared by Image Classification experiments (Section 4.3): the optimizer is SGD, batch size 128, number of total training epochs 200, weight decay 5e-4, and momentum 0.9. For each experiment setting, we repeat our experiments with three random seeds {43, 37, 13}. We also report the mean and standard deviation of the test accuracy across these seeds. In Table 13, we report the details of experiments for each model and method. We use the same learning rate range from (Zhou et al., 2024) and we expand the scaling ratio range into [(0.1, 1.9), (0.2, 1.8), (0.3, 1.7), (0.4, 1.6), (0.5, 1.5), (0.6, 1.4), (0.7, 1.3), (0.8, 1.2), (0.9, 1.1)] nine choices.

Second, we provide the hyperparameters used in experiments of LLM pruning and SciML. We follow the common hyperparameter settings as described in Lu et al. (2024); Liu et al. (2024). See more details for other hyperparameters like τ

in LLM pruning and scaling ratios in SciML in Table 14.

Finally, we report the detailed matrix subsampling settings used in every model in Table 15. We cannot use a very small window size or sampling steps because doing so may not cover the entire matrix. Conversely, selecting a very large size would result in too much overlap between sampled matrices. For ResNet, VGG and DPOT series models, we use the minimum dimension of the weight matrices to construct the sampled submatrices based on the parameter Q. We also select the $\lfloor m/n \rfloor$ for the submatrices number, where m, n is the dimension of weight matrix, $m \geq n$. But for the final layer of ResNet and VGG models, we select nine submatrices based on experiments results. For LLaMA series models, we apply sliding window sampling using multiple moderately sized submatrices, resulting in a smoother `PL_Alpha_Hill` estimation.

Table 13. Parameter settings of the experiment reported in Section 4.3.

Model	Method	Initial Learning Rate	Scaling Ratio (s_1, s_2)	Test Accuracy
ResNet 18	CAL	0.05, **0.1**, 0.15	-	$78.23_{\pm 0.087}$
	TB(no LS)	0.1	(0.6, 1.4)	$78.76_{\pm 0.111}$
	LS+TB	0.1	(0.2, 1.8)	$79.31_{\pm 0.180}$
	Ours(no LS)	0.1	(0.1, 1.9)	$79.49_{\pm 0.080}$
	LS+Ours	0.1	(0.1, 1.9)	$\mathbf{79.53_{\pm 0.177}}$
ResNet 34	CAL	**0.05**, 0.1, 0.15	-	$78.99_{\pm 0.137}$
	TB(no LS)	0.1	(0.5, 1.5)	$79.64_{\pm 0.029}$
	LS+TB	0.1	(0.3, 1.7)	$80.00_{\pm 0.090}$
	Ours(no LS)	0.1	(0.2, 1.8)	$80.17_{\pm 0.213}$
	LS+Ours	0.1	(0.3, 1.7)	$\mathbf{80.20_{\pm 0.221}}$
VGG 16	CAL	0.025, **0.05**, 0.1	-	$74.30_{\pm 0.078}$
	TB(no LS)	0.05	(0.6, 1.4)	$74.43_{\pm 0.158}$
	LS+TB	0.05	(0.3, 1.7)	$75.19_{\pm 0.131}$
	Ours(no LS)	0.05	(0.2, 1.8)	$\mathbf{75.36_{\pm 0.118}}$
	LS+Ours	0.05	(0.2, 1.8)	$75.15_{\pm 0.247}$
VGG 19	CAL	0.025, **0.05**, 0.1	-	$73.11_{\pm 0.113}$
	TB(no LS)	0.05	(0.6, 1.4)	$73.22_{\pm 0.277}$
	LS+TB	0.05	(0.2, 1.8)	$74.19_{\pm 0.159}$
	Ours(no LS)	0.05	(0.2, 1.8)	$\mathbf{74.28_{\pm 0.392}}$
	LS+Ours	0.05	(0.4, 1.6)	$73.99_{\pm 0.300}$

Table 14. Hyperparameters for LLaMA and DPOT models. (Left) The range of τ used for LLM pruning. (Right) Learning rate and scaling ratio settings for DPOT series models at different subsampling ratios.

Sparsity Ratio	LLaMA-7B/13B
0.7	0.1, 0.2, 0.3, 0.4, 0.5, 0.6
0.75	0.1, 0.2, 0.3, 0.4, 0.5, 0.6
0.8	0.1, 0.2, 0.25, 0.3, 0.4, 0.5
0.85	0.1, 0.15, 0.2, 0.25, 0.3

Model	DPOT-Tiny		DPOT-Small	
Hyperparameters	Learning Rate	Scaling Ratio	Learning Rate	Scaling Ratio
5%	2.5e-4	(1.0, 1.0)	1e-4	(1.0, 1.0)
10%	2.5e-4	(1.0, 1.0)	1e-4	(1.0, 1.0)
25%	2.5e-4	(1.0, 1.0)	2.5e-4	(1.0, 1.0)
50%	5e-4	(1.0, 1.0)	2.5e-4	(1.0, 1.0)
100%	5e-4	(1.0, 1.0)	2.5e-4	(1.0, 1.0)

Table 15. Subsampling Hyperparameters for different models.

Model	**Aspect Ratio**(Q)	**Window Size**	**Submatrices Number**
ResNet 18/34, VGG 16/19	1.0	Minimum Dimension	$\lfloor m/n \rfloor$, 9
DPOT-Tiny/Small	1.0	Minimum Dimension	$\lfloor m/n \rfloor$
LLaMA-7B	1.0	2000	15×15
	1.0	2000	10×10
	1.0	1000	10×10
LLaMA-13B	1.0	2000	15×15