GEM: A Scale-Aware and Distribution-Sensitive Sparse Fine-Tuning Framework for Effective Downstream Adaptation

Sungmin Kang[1], **Jisoo Kim**[2], **Salman Avestimehr**[1], **Sunwoo Lee**[2*]

[1]University of Southern California, Los Angeles, CA, USA
[2]Inha University, Incheon, Republic of Korea
kangsung@usc.edu, starprin3@inha.edu, avestime@usc.edu, sunwool@inha.ac.kr

Abstract

Parameter-efficient fine-tuning (PEFT) has become a popular way to adapt large pre-trained models to new tasks. Most PEFT methods update only a small subset of parameters while freezing the rest, avoiding redundant computation. As they maximize the absolute size of the updates without regard to the parameters' original scale, the resulting changes in model behavior can be minimal. In contrast, we maximize updates relative to each parameter's scale, yielding more meaningful downstream adaptation. We propose **G**radient-to-Weight Ratio and **E**ntropy-guided **M**asking (GEM), a parameter scale-aware, distribution-sensitive sparse fine-tuning framework. GEM prioritizes parameters whose updates are significant in proportion to their initial pre-trained values. It also adaptively determines how many parameters to tune at each layer based on the entropy of parameter values, thereby making the most effective use of the computational budget in PEFT. Our empirical study demonstrates the efficacy of GEM on both general-domain tasks (GLUE and SuperGLUE) and domain-specific tasks (GSM8k and MBPP), achieving up to a 1.6% improvement in fine-tuning accuracy over full fine-tuning while updating only 0.1% of model parameters.

1 Introduction

With the success of large-scale pre-trained models (Brown et al. 2020; Zhang et al. 2022; Touvron et al. 2023; Gunasekar et al. 2023), fine-tuning has been an efficient yet effective way of adapting large models to downstream tasks using fewer training data. Despite the success, fine-tuning the entire large-scale model demands intensive resource and time, which poses significant challenges to adapt the model to multiple target tasks. To reduce the computational burden, parameter-efficient fine-tuning (PEFT) methods have been proposed with the aim to adapt models by updating only a small subset of parameters while keeping the majority of weights fixed.

Several lines of research have been proposed to address the high cost of fine-tuning, such as Low-rank Adaptation (LoRA) (Hu et al. 2022), Adapter (Houlsby et al. 2019), and prompt tuning (Li and Liang 2021). Parameter masking-based sparse fine-tuning methods (Zhao et al. 2020) have also been proposed, showing that most parameters can be frozen, with only a small portion being fine-tuned to achieve comparable accuracy to full fine-tuning (FFT). These methods construct masks by estimating the importance of individual parameters and selecting those with the highest scores, using criteria such as gradient magnitude (Li et al. 2025; Zhang et al. 2024), Fisher matrix (Agarwal et al. 2024; Sung, Nair, and Raffel 2021; Das et al. 2023), or Hessian-based sensitivity measures (Xu and Bu 2025). However, existing approaches overlook the scale of pre-trained parameter weights during parameter selection.

A common goal of PEFT is to maximize downstream adaptation within a constrained budget. Here, we re-define this goal as maximizing downstream adaptation not by the absolute magnitude of parameter updates, but as the extent to which parameters change *relative to their pre-trained values*. In other words, a parameter that undergoes a large proportional shift from its initial value is considered to contribute more significantly to task-specific adaptation, even if its absolute update is small. Therefore, maximizing downstream adaptation under a limited budget entails identifying and updating parameters that exhibit the greatest relative change. This motivates our focus on relative parameter change $|\Delta w|/|w|$ as a principled criterion for identifying the most adaptable parameters under tight tuning budgets.

In this work, we propose GEM, a general PEFT framework built upon a novel parameter prioritization metric and an entropy-guided parameter selection method. The main principle of our framework is to select a small portion of parameters that have the most critical impact on model behavior and fine-tune only those parameters efficiently. First, we investigate the impact of a parameter selection strategy based on the gradient magnitude relative to the parameter values. We empirically explore and demonstrate why the gradient-to-weight ratio identifies important parameters more effectively than the gradient magnitude alone. Second, we investigate how to exploit entropy information at each network layer to determine the number of parameters to be tuned per layer. Our study empirically proves that entropy is a useful metric for identifying layers with an irregular distribution of parameter importance. We can safely freeze more parameters in layers with irregular distributions, expecting a greater reduction in computational cost while maintaining fine-tuning accuracy.

*Corresponding Author

Method	Model Agnostic	No Extra Train Param	No Extra Infer Param	Param-wise Importance	Layer-wise Importance	Parameter Scale-aware	Tuning Budget Controllable	Static Mask
LoRA	✗	✗	✓	✗	✗	✗	✗	✓
BitFit	✓	✓	✓	✗	✗	✗	✗	✓
Adapter	✗	✗	✗	✗	✗	✗	✗	✓
AdaLoRA	✗	✗	✓	✓	✓	✗	✗	✗
Random Mask	✓	✓	✓	✗	✗	✗	✓	✓
Top-Grad Mask	✓	✓	✓	✓	✓*	✗	✓	✓
GEM	✓	✓	✓	✓	✓	✓	✓	✓

* Some implementations of Top-Grad Masking optionally consider per-layer allocation.

Table 1: Comparison of GEM with representative PEFT baselines. GEM is a versatile method that combines scale-awareness, hierarchical selectivity, and model-agnostic design, while being static and parameter-efficient.

To evaluate the performance of GEM, we conduct benchmark experiments on several popular datasets, including RTE, SST-2, WiC, BoolQ, MultiRC, CoPA, and SQuAD. We directly compare our proposed method to a variety of SOTA PEFT methods such as LoRA, BitFit, Adapter, AdaLoRA, random masking, and top gradient masking. Our extensive experiments consistently demonstrate that the gradient-to-weight ratio and the entropy-aware parameter selection strategy achieve the best fine-tuning accuracy while significantly reducing the computational cost. We also present a few ablation study results to further evaluate the efficacy of our method. Based on our observations and analyses, we conclude that GEM is a highly efficient PEFT framework that can be readily applied to a wide range of real-world LLM-based applications.

Our main contributions are summarized as follows:

- We propose a novel parameter prioritization method that maximizes weight changes relative to their original scale and analyze its impact on fine-tuning performance.
- We propose an entropy-guided, layer-wise parameter-allocation method that dynamically selects the number of tunable parameters in each layer, and we analyze how exploiting both the magnitude and distribution of learning signals maximizes fine-tuning performance.
- Building on these two methods, we introduce a general PEFT framework, GEM, and empirically validate its effectiveness through extensive experiments across diverse tasks, including both general-domain and domain-specific benchmarks.

2 Related Works

Reparameterization-based Methods. A key family of PEFT techniques reparameterizes pre-trained models by modifying internal parameterizations, either by injecting lightweight modules, or prepending tunable inputs, while keeping the original model weights frozen.

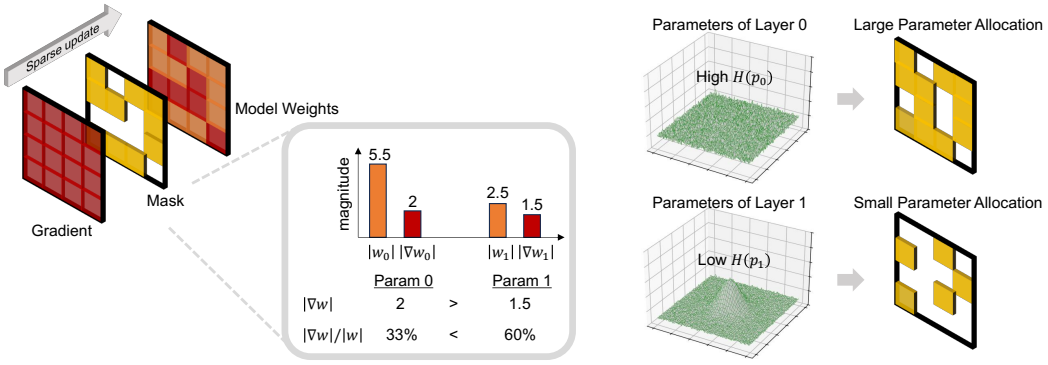
Low-Rank Adaptation (LoRA) (Hu et al. 2022) is a popular PEFT method, which utilizes a low-rank decomposition to efficiently fine-tune models with minimal parameter overhead. There are adaptive rank allocation methods such as AdaLoRA (Zhang et al. 2023b) and ALoRA (Liu et al. 2024). GoRA (He et al. 2025) and SoRA (Ding et al. 2023) utilize gradient-based signals to control rank allocation. LoRA-FA (Zhang et al. 2023a), VeRA (Kopiczko, Blankevoort, and Asano 2024), and S-LoRA (EV et al. 2024) enhance memory efficiency by limiting the number, or the extent, of trainable parameters.

Introduced by Houlsby et al. (Houlsby et al. 2019), adapters are lightweight modules inserted within transformer layers to enable task-specific adaptation, significantly reducing the number of tunable parameters. Parallel adapters (Pfeiffer et al. 2020) allow adapters to be inserted alongside the main path for better gradient flow, while AdapterFusion (Pfeiffer et al. 2021) enables multi-task adaptation by learning to combine knowledge from multiple adapters. Compacter (Karimi Mahabadi, Henderson, and Ruder 2021) factorizes adapter weights into shared low-rank components.

Sparsification and Masking-based Methods. Motivated by the lottery ticket hypothesis (Frankle and Carbin 2019) and structured pruning techniques (Guo, Rush, and Kim 2021), sparsification and masking have been widely applied to PEFT. These methods identify a subset of parameters based on importance measures and fine-tune the subset only. Another line of work leverages gradient magnitude as a proxy for parameter importance, either by pre-selecting weights with large gradients or dynamically masking them during training (Zhang et al. 2024; Li et al. 2025; Song et al. 2024). Child-Tuning (Xu et al. 2021) restricts updates to a subnetwork selected either randomly or based on accumulated gradient norms. More recently, random masking (Xu and Zhang 2024), which selects a random subset to update, has also shown surprisingly competitive performance.

Quantization- and Pruning-based Methods. Several quantization-based methods reduce the precision of model parameters to decrease the memory footprint and accelerate computation. For example, QLoRA (Dettmers et al. 2023) enables 4-bit fine-tuning of large language models with minimal performance degradation, while OPTQ (Frantar et al. 2023) and AWQ (Lin et al. 2024) provide post-training quantization techniques for compressing LLMs efficiently. These methods allow parameter-efficient tuning of LLMs under severe memory constraints.

Pruning-based methods explicitly remove a subset of parameters to construct a sparse subnetwork, reducing training



(a) Gradient-to-Weight Ratio-based Masking **(b) Entropy-aware Parameter Allocation**

Figure 1: Schematic illustration of GEM framework's key components. (a) We prioritize parameters that maximize the relative weight change, rather than the absolute weight change commonly used in prior methods. This figure illustrates a simple example in which the important parameter varies with the metric used. While existing methods select parameter 0 based on its larger gradient magnitude, our method selects parameter 1 due to its larger gradient relative to its original weight. (b) The number of tunable parameters in each layer is determined by the degree to which the learning signal is concentrated. We quantify layer-wise allocations using entropy-based importance scores.

cost and inference overhead. DiffPruning (Guo, Rush, and Kim 2021) learns a binary mask that selects only a small fraction of weights to update during fine-tuning, while keeping the rest frozen. Light-PEFT (Gu et al. 2024) adopts an early-stage pruning strategy that estimates parameter importance in the initial training phase, enabling the removal of less critical components before full adaptation.

3 Method

In this section, we propose a binary masking framework for selecting tunable parameters to achieve accurate and parameter-efficient fine-tuning. Our framework comprises two key components: (i) parameter prioritization based on the gradient-to-weight ratio, and (ii) entropy-guided layer-wise parameter selection. Based on these two methods, we construct a general PEFT framework, GEM. Figure 1 shows an overview of GEM, illustrating how it selects the most critical and minimal subset of parameters to tune.

3.1 Gradient-to-Weight Ratio for Parameter Prioritization

Parameter Prioritization using a Scale-Invariant Signal. Consider a standard first-order update rule: $w_{t+1} = w_t - \eta \cdot \nabla_w \mathcal{L}$, where η is the learning rate. The weight update magnitude is $|\Delta w| = \eta \cdot |\nabla_w \mathcal{L}|$, and its relative scale with respect to the parameter is given by

$$\frac{|\Delta w|}{|w|} = \eta \cdot \frac{|\nabla_w \mathcal{L}|}{|w|}. \tag{1}$$

Motivated by the analysis above, we define a scale-invariant signal for parameter prioritization, referred to as the gradient-to-weight ratio (GWR), as

$$\rho^{(i)} := \left| \frac{\nabla_{w^{(i)}} \mathcal{L}}{w^{(i)}} \right|, \tag{2}$$

where $w^{(i)}$ denotes the scalar i-th element of the weight tensor W. We consider parameters with a high ρ value to be critical in the fine-tuning process. Even if two parameters have gradients of comparable magnitude, the parameter with smaller weight will experience a larger relative update, potentially causing greater change in the model's behavior. This suggests that relying solely on gradient magnitude may overlook the relative scale of updates, while the gradient-to-weight ratio provides a more accurate and scale-aware measure of parameter influence. Note that this signal is naturally available during training, as both the numerator and the denominator are computed at every iteration, incurring almost no additional computational cost.

Theoretical Interpretation of Gradient-to-Weight Ratio. The gradient-to-weight ratio can be interpreted as a good metric for quantifying parameter importance. Let us begin with a first-order Taylor approximation of the loss $\mathcal{L}(w)$:

$$\mathcal{L}(w + \Delta w) \approx \mathcal{L}(w) + \nabla_w \mathcal{L}(w)^T \Delta w \tag{3}$$

$$\Delta \mathcal{L} \approx \nabla_w \mathcal{L}(w)^T \Delta w \tag{4}$$

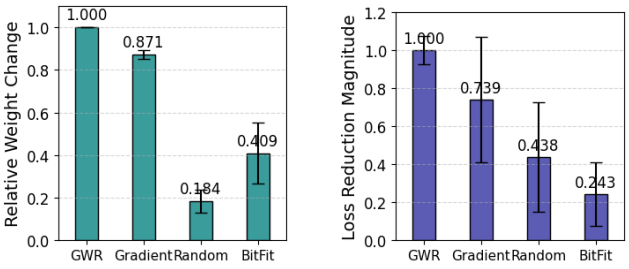
Assuming a gradient descent update $\Delta w = -\eta \nabla_w \mathcal{L}(w)$, the change in loss becomes:

$$\Delta \mathcal{L} \approx -\eta \|\nabla_w \mathcal{L}(w)\|^2 \;\Rightarrow\; \frac{|\Delta \mathcal{L}|}{\|\nabla_w \mathcal{L}(w)\|} \approx \eta \|\nabla_w \mathcal{L}(w)\|.$$

This shows that the quantity $|\Delta \mathcal{L}|/\|\nabla_w \mathcal{L}(w)\|$ characterizes the reduction in loss per unit of gradient norm when taking a step along the negative gradient direction. In other words, it measures how efficiently the loss is decreased relative to the local steepness of the loss landscape at the current parameter setting.

Meanwhile, the norm of the update satisfies $\|\Delta w\| = \eta \|\nabla_w \mathcal{L}(w)\|$, leading to:

$$\frac{|\Delta \mathcal{L}|/\|\nabla_w \mathcal{L}(w)\|}{\|w\|} \approx \frac{\|\Delta w\|}{\|w\|}.$$



(a) Relative weight change (b) Loss reduction

Figure 2: **Gradient-to-weight ratio identifies critical parameters by prioritizing updates that better align with loss reduction.** We fine-tune OPT-125M on WiC for 10 epochs, testing four masking strategies over three seeds. Results are normalized. Plot (a) shows total deviation from the pre-trained weights; plot (b) shows corresponding loss reduction. Gradient-to-weight masking produces both the largest relative weight shifts and the greatest loss drop.

Thus, the left-hand side can be interpreted as a scale-invariant measure of effective loss reduction, quantifying optimization progress relative to both gradient magnitude and parameter scale. The right-hand side represents the relative parameter update, and this equivalence highlights a direct connection between the extent of loss reduction and the magnitude of parameter changes relative to their scale. Based on this, we conclude that the gradient-to-weight ratio effectively captures how significantly each parameter contributes to reducing the loss.

Empirical Interpretation of Gradient-to-Weight Ratio. Beyond our theoretical understanding, we empirically investigate the effectiveness of the gradient-to-weight ratio as a metric for quantifying parameter importance. To assess the impact of parameter selection methods on fine-tuning efficiency, we evaluate two metrics: (1) total relative weight change with respect to the initial pre-trained weights, measured as $\left\| \frac{w_t - w_0}{w_0} \right\|$, and (2) the magnitude of loss reduction, computed as $|\Delta\mathcal{L}| \approx |-\nabla_w \mathcal{L}_0 \cdot (w_t - w_0)|$, where w_0 and w_t denote the initial and fine-tuned weights of the selected parameters, respectively, after t epochs.

Figure 2(a) shows that selecting parameters based on the gradient-to-weight ratio, ρ, leads to the largest relative weight change, indicating that this metric enables us to effectively prioritize the parameters that undergo meaningful updates relative to their scale. This observation is consistent with the intended design of our method. In addition, Figure 2(b) shows that the proposed parameter selection method achieves the best loss reduction among all the methods. This result verifies that the gradient-to-weight ratio not only identifies parameters that are significantly updated, but also selects those that contribute more directly and effectively to optimization progress. These empirical results align well with the theoretical interpretation presented earlier, which showed that the gradient-to-weight ratio captures both the relative scale of parameter updates and their contribution to loss reduction.

3.2 Entropy-Guided Layer-wise Parameter Selection

Strength- and Distribution-Aware Parameter Allocation. Here, we propose a principled approach to quantifying layer importance by analyzing the distribution of parameter-wise gradient-to-weight ratios in each layer. Our approach allocates the number of tunable parameters in each layer based on the quantified layer importance. This method plays a key role in reducing the number of tunable parameters while maintaining high accuracy.

For a given layer ℓ, we first normalize the gradient-to-weight ratio scores across its parameters to obtain a probability distribution:

$$p_\ell^{(i)} = \frac{\rho_\ell^{(i)}}{\sum_j \rho_\ell^{(j)}}, \tag{5}$$

where i indicates a specific parameter, and j iterates over all the parameters at layer ℓ. Based on this distribution, we compute the entropy of the layer:

$$\mathcal{H}(\boldsymbol{p}_\ell) = -\sum_i p_\ell^{(i)} \log p_\ell^{(i)}, \tag{6}$$

which serves as an indicator of how the learning signal is distributed across the parameters in a layer. A low entropy indicates that the learning signal is concentrated in a small subset of parameters, enabling a more selective tuning strategy. In contrast, a high entropy suggests that the signal is more evenly spread, implying that a larger number of parameters are collectively important for the task.

Here, we define a layer importance metric which reflects both the overall magnitude of updates and the internal distribution of the learning signal as follows:

$$\alpha_\ell = \|\boldsymbol{\rho}_\ell\|_2 \cdot \mathcal{H}(\boldsymbol{p}_\ell). \tag{7}$$

Here, the norm term captures the total relative update magnitude of the layer, reflecting how strongly the layer contributes to learning. The entropy term scales this magnitude based on how concentrated or spread out the learning signal is across the parameters. If the signal is sharply concentrated (i.e., low entropy), fewer parameters can be selected without compromising learning capacity. If the signal is spread out (i.e., high entropy), more parameters must be tuned to account for the distributed importance.

This entropy-guided importance metric enables layer-specific and efficient allocation of the tuning budget across layers. After computing the importance scores α_ℓ for all tunable layers, we normalize them across layers to form a proportional allocation:

$$\gamma_\ell = \frac{\alpha_\ell}{\sum_j \alpha_j}, \tag{8}$$

where γ_ℓ represents the relative share of the total parameter budget allocated to layer ℓ. Given the total tunable parameter ratio r and the total number of parameters N, we determine the number of selected parameters in each layer as

$$k_\ell = \lfloor r \cdot N \cdot \gamma_\ell \rfloor. \tag{9}$$

We then select the top-k_ℓ parameters for each layer ℓ, ranked by their gradient-to-weight ratio scores $\rho_\ell^{(i)}$.

Algorithm 1: GEM (Gradient-to-Weight Ratio and Entropy-guided Masking)

Input: Pretrained weight W_0, gradient $\nabla\mathcal{L}(W_0)$, tuning ratio r
Identify tunable modules (e.g., query, value projections)
for each tunable layer ℓ **do**
  for each parameter $w_\ell^{(i)}$ in layer ℓ **do**
    Compute gradient-to-weight ratio $\rho_\ell^{(i)}$ // Eq. (2)
  end for
  Normalize to get probabilities $p_\ell^{(i)}$ // Eq. (5)
  Compute layer entropy $\mathcal{H}(\boldsymbol{p}_\ell)$ // Eq. (6)
  Compute layer importance α_ℓ // Eq. (7)
end for
Normalize importance scores across layers: γ_ℓ // Eq. (8)
for each layer ℓ **do**
  Compute number of tunable parameters k_ℓ // Eq. (9)
  Select top-k_ℓ parameters with highest $\rho_\ell^{(i)}$
  Construct binary mask M_ℓ where $[M_\ell]_{m,n} = 1$ if $w_\ell^{(i)}$ is selected for tuning, 0 otherwise
end for
Output: Binary masks M_ℓ for each tunable layer ℓ

Experimental Analysis of Entropy-aware Layer Score. We validate our entropy-guided importance score α_ℓ defined in Equation 7. Table 2 summarizes the results across three GLUE-style tasks (SST-2, BoolQ, and MultiRC), comparing three layer-wise parameter selection strategies: (1) *Uniform*, where each layer receives the same number of trainable parameters; (2) *Norm only*, which allocates parameters to layers solely based on the magnitude of their gradient-to-weight ratio norm $\|\boldsymbol{\rho}_\ell\|_2$; and (3) *Norm* $\times$ *Entropy*, our proposed metric α_ℓ, which jointly considers both the magnitude and the intra-layer distribution of learning signals.

Table 2 (top) shows the share of the total gradient-to-weight signal each method captures. *Norm only* outperforms the uniform baseline, but adding entropy covers even more signal with the same parameter budget, confirming that intra-layer diversity helps identify more informative parameters. Table 2 (bottom) reports validation accuracy for each strategy. Methods that capture more of the gradient-to-weight signal achieve higher accuracy. Our entropy-guided approach beats both *Uniform* and *Norm only* on every task, showing the value of modeling both signal strength and diversity in PEFT.

Based on the two key observations above, we conclude that the proposed entropy-guided layer importance metric, α_ℓ, is a comprehensive measure for determining how many parameters should be fine-tuned at each layer, consistently selecting more informative parameters. In particular, our empirical study demonstrates that entropy serves as a useful layer-wise information metric, enabling us to maximize PEFT efficiency and improve downstream performance.

3.3 GEM Framework for Sparse Fine-Tuning

Algorithm 1 presents pseudo code of our GEM framework built upon the proposed parameter prioritization and selection methods. Here, we define how to build masks for PEFT at each layer. Let $W_\ell \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ denote a weight matrix in layer ℓ and $M_\ell \in \{0,1\}^{d_{\text{in}} \times d_{\text{out}}}$ be the corresponding binary mask. We define M_ℓ elementwise as:

$$[M_\ell]_{m,n} = \begin{cases} 1, & \text{if } [W_\ell]_{m,n} \text{ is selected for tuning,} \\ 0, & \text{otherwise,} \end{cases}$$

and apply the mask during weight updates as follows:

$$W_\ell \leftarrow W_\ell - \eta \cdot \nabla_{W_\ell}\mathcal{L} \odot M \tag{10}$$

where η is the learning rate and $\odot$ denotes element-wise multiplication. This ensures that only the selected parameters receive updates, while the rest remain frozen, enabling sparse and efficient fine-tuning.

Task	Uniform	Norm only	Norm $\times$ Entropy
Captured GWR by Selected Parameters (% of Total)			
SST-2	49.69%	68.25%	**71.70%**
BoolQ	49.52%	71.32%	**74.22%**
MultiRC	47.67%	68.52%	**72.36%**
Validation Accuracy			
SST-2	91.74%	93.11%	**94.84%**
BoolQ	68.33%	69.91%	**72.39%**
MultiRC	65.08%	66.56%	**70.31%**

Table 2: **Entropy-aware strategy captures more informative parameters across layers.** Top: share of the total gradient-to-weight magnitude captured by the chosen parameters (how well each method covers important weights under a fixed budget). Bottom: corresponding validation accuracy. We trained OPT-1.3B on SST-2, BoolQ, and MultiRC, fine-tuning 0.1% of parameters using the gradient-to-weight ratio as the importance metric.

4 Experiments

4.1 Experimental Setup

Datasets and Models. We evaluate GEM on seven SuperGLUE (Wang et al. 2020) and GLUE (Wang et al. 2018) benchmarks: RTE (Dagan, Glickman, and Magnini 2005), SST-2 (Socher et al. 2013), WiC (Pilehvar and Camacho-Collados 2018), BoolQ (Clark et al. 2019), MultiRC (Khashabi et al. 2018), COPA (Roemmele, Bejan, and Gordon 2011), and SQuAD v2.0 (Rajpurkar, Jia, and Liang 2018). We report accuracy for the classification tasks and F1 for MultiRC and SQuAD v2.0.

We use three pre-trained language models with varying model capacities: OPT-125M, OPT-1.3B (Zhang et al. 2022), and Microsoft Phi-2 (Javaheripi et al. 2023). Microsoft Phi-2 is a larger model with 2.7 billion parameters, known for its strong language modeling performance and reasoning tasks.

Baseline Methods. To evaluate the effectiveness of GEM, we compare it to several representative PEFT methods: LoRA (Hu et al. 2022), BitFit (Zaken, Ravfogel, and Goldberg 2022), Adapter (Houlsby et al. 2019), AdaLoRA (Zhang et al. 2023b), Random Masking (Xu and Zhang 2024), and Top Gradient Masking (Zhang et al. 2024; Li et al. 2025). Detailed settings for each method are provided in the Appendix.

Model	Method	Params	RTE	SST-2	WiC	BoolQ	MultiRC	COPA	SQuAD	Avg.
OPT-125M	FFT	100%	61.21±0.6%	89.33±0.5%	60.56±1.2%	62.81±0.7%	64.35±0.7%	69.67±0.5%	62.71±0.9%	67.23
	LoRA	0.235%	60.17±0.8%	89.49±0.8%	59.40±0.7%	63.17±0.6%	64.79±1.0%	69.00±0.7%	62.70±1.1%	66.96
	BitFit	0.082%	61.13±0.7%	88.69±1.1%	58.31±0.7%	63.26±0.7%	66.52±0.6%	69.33±0.4%	57.98±0.6%	66.46
	Adapter	0.25%	60.65±0.7%	89.07±1.2%	60.14±1.5%	62.52±1.1%	65.53±0.9%	**69.67**±1.2%	60.47±0.8%	66.86
	AdaLoRA	0.246%	61.61±0.5%	88.76±0.8%	58.52±0.8%	63.42±1.0%	66.59±0.6%	69.33±0.7%	61.98±0.9%	67.17
	Random Mask	0.1%	62.82±1.5%	89.33±0.9%	59.87±0.9%	63.01±0.8%	60.95±1.3%	66.33±1.3%	61.69±0.9%	66.29
	Top Gradient Mask	0.1%	51.26±0.7%	88.07±0.6%	57.68±1.0%	62.38±0.9%	63.83±0.8%	66.33±0.9%	61.03±0.5%	64.37
	GEM	0.1%	**65.10**±0.9%	**89.53**±0.7%	**64.16**±1.2%	**63.65**±0.9%	**66.89**±0.7%	**69.67**±0.6%	**62.98**±0.7%	**68.85**
OPT-1.3B	FFT	100%	73.66±0.7%	93.42±0.6%	67.13±0.9%	72.05±0.8%	68.31±0.9%	82.33±0.6%	81.83±0.8%	76.96
	LoRA	0.120%	72.56±0.6%	93.65±1.2%	63.01±0.9%	72.01±0.7%	68.90±0.4%	**83.00**±0.9%	82.22±0.8%	76.48
	BitFit	0.041%	72.08±0.8%	92.65±0.6%	62.07±0.6%	71.10±0.6%	68.43±0.9%	79.67±0.4%	80.24±0.6%	75.18
	Adapter	0.127%	72.80±1.1%	92.34±0.8%	63.90±0.7%	69.02±0.6%	69.57±1.2%	79.00±0.8%	82.21±0.7%	75.55
	AdaLoRA	0.125%	70.16±0.7%	93.71±0.9%	65.20±0.6%	72.52±0.9%	68.89±0.5%	82.67±1.0%	**82.61**±0.7%	76.54
	Random Mask	0.1%	69.55±1.1%	93.15±0.8%	63.22±0.8%	69.76±1.6%	66.90±0.7%	81.33±0.7%	81.22±1.2%	75.02
	Top Gradient Mask	0.1%	67.75±0.9%	91.74±0.9%	61.23±0.5%	62.17±0.8%	59.54±1.0%	77.67±0.6%	79.83±0.8%	71.42
	GEM	0.1%	**74.73**±0.7%	**93.84**±0.9%	**65.47**±0.6%	**72.73**±0.9%	**70.31**±0.7%	**83.00**±0.4%	82.23±0.8%	**77.47**
Microsoft Phi-2 2.7B	FFT	100%	83.49±0.8%	93.58±0.7%	69.75±0.7%	84.10±0.8%	83.51±1.0%	92.00±0.0%	90.46±0.7%	85.27
	LoRA	0.094%	76.35±0.9%	93.06±0.9%	66.97±0.8%	83.58±0.8%	79.17±0.7%	90.67±0.6%	90.48±1.2%	82.90
	BitFit	0.009%	74.73±0.9%	92.83±0.6%	60.86±0.9%	83.27±0.8%	78.87±0.9%	87.67±0.5%	88.52±0.5%	80.96
	Adapter	0.200%	74.01±0.8%	92.43±0.9%	61.88±0.5%	83.10±0.7%	77.87±0.9%	84.33±1.1%	89.02±0.7%	80.38
	AdaLoRA	0.138%	81.49±0.9%	94.04±0.8%	67.82±0.9%	84.77±0.6%	82.92±0.6%	91.00±0.8%	90.44±0.7%	84.64
	Random Mask	0.1%	79.60±1.2%	94.09±0.9%	68.07±0.8%	84.48±0.9%	81.96±0.9%	90.67±0.6%	90.57±0.8%	84.21
	Top Gradient Mask	0.1%	77.98±0.6%	94.32±0.9%	66.11±0.8%	83.82±0.8%	82.92±0.9%	89.00±0.7%	89.92±0.8%	83.44
	GEM	0.1%	**83.94**±0.7%	**94.44**±0.6%	**69.56**±0.6%	**85.02**±0.9%	**83.46**±0.9%	**91.33**±0.8%	**90.89**±0.7%	**85.52**

Table 3: Performance comparison across various PEFT methods and baselines on different NLP tasks. Accuracy is reported along with the corresponding standard deviation over three random seeds.

4.2 Performance Comparison

Table 3 compares the fine-tuning performance of PEFT methods on seven language tasks. As shown in the **Avg.** column, which reports mean accuracy across all seven tasks, GEM outperforms all other SOTA methods with OPT-125M, OPT-1.3B, and Phi-2, consistently surpassing sparse fine-tuning baselines at comparable tuning ratios. Notably, GEM surpasses Top Gradient Masking by 4.48% on OPT-125M and 6.05% on OPT-1.3B, and outperforms Random Masking by 2.56% and 2.45%, respectively. We also see that Random Masking exhibits a higher standard deviation (up to 1.6%) compared to other methods, which indicates unstable training behavior. Note that Adapter shows significantly lower performance on Phi-2 compared to OPT models, which may be attributed to its sensitivity to residual scaling mismatch in pre-norm architectures like Phi-2, whereas it achieves reasonable performance on post-norm models such as OPT.

GEM also outperforms LoRA, BitFit, and Adapter on every task except only three (OPT-125M and OPT-1.3B on COPA, and Phi-2 on SQuAD). Notably, GEM outperforms AdaLoRA across all model sizes, even though it uses fewer tunable parameters and fixed masks, showing that a well-designed static selection can surpass more complex adaptive schemes.

Our extensive empirical study highlights two key strengths of GEM. First, it shows strong performance across diverse task types including classification (SST-2, WiC), natural language inference (RTE, COPA), multi-sentence reading comprehension (BoolQ, MultiRC), and extractive QA (SQuAD), showing versatility and robustness. Second, GEM demonstrates effectiveness across both lightweight (OPT-125M) and mid-sized (Phi-2) models, ranking first on all seven tasks and achieving the highest average score, highlighting its applicability under a range of computational constraints. Therefore, we conclude that the proposed parameter selection strategy is a practical and effective PEFT method that can be readily applied to real-world applications.

Comparative Study on Masking Strategies. Here, we specifically compare our masking strategy against Random Masking and Top Gradient Masking under the same tuning ratio (0.1%) as reported in Table 3. Random Masking selects parameters uniformly at random without any parameter prioritization criteria. While previous work (Xu and Zhang 2024) demonstrated that such random selection can still yield competitive performance, the absence of an importance-based selection mechanism fundamentally limits its effectiveness. Top Gradient Masking prioritizes parameters solely based on the gradient magnitude, disregarding the weight scale and update direction—factors critical for meaningful parameter prioritization. Furthermore, both Random Masking and Top Gradient Masking adopt a uniform selection scheme across layers, assigning equal number of updating parameters per layer. This layer-agnostic strategy further restricts their ability to allocate updates effectively. In contrast, GEM selects the number of tuning parameters per layer based on entropy-guided layer scoring, which enables more effective update allocation and leads to superior performance.

Model	Task	Uniform	Norm Only	Entropy Only	GEM
OPT-125M	SST-2	85.34	87.12	88.17	**89.53**
	SQuAD	59.52	60.76	60.69	**62.98**
OPT-1.3B	SST-2	91.98	92.84	91.92	**93.84**
	SQuAD	80.48	81.72	80.69	**82.23**
Phi-2	SST-2	93.37	92.66	92.41	**94.44**
	SQuAD	88.70	89.44	87.84	**90.89**

Table 4: Comparison of different layer importance strategies for parameter allocation.

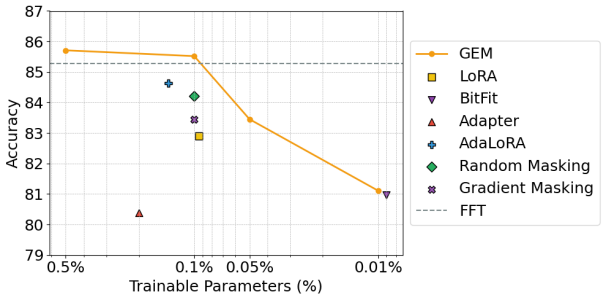


Figure 3: Average performance of PEFT methods using Phi-2 on seven tasks. GEM shows superior performance over methods with a comparable 0.1% parameter budget.

4.3 Ablation Study and Discussion

Comparison of Layer Importance Metrics. Table 4 compares different strategies for layer-wise importance estimation. The in-layer parameter selection criterion is fixed to the gradient-to-weight ratio. The *Uniform* strategy all layers equally, giving each the same number of tunable parameters. The *Norm Only* approach scores layers by the ℓ_2 norm of their gradient-to-weight ratios, capturing update strength. The *Entropy Only* approach ignores magnitude and instead measures how focused those ratios are within each layer. Finally, our proposed method, GEM, combines both the norm and entropy, targeting layers whose learning signals are both strong and focused.

GEM outperforms every baseline across tasks and models. These results show that layer importance is uneven: allocating the same parameter count to each layer wastes tuning budget. Instead, parameters should scale with each layer's contribution, considering both signal strength (norm) and signal spread (entropy). Relying on either one alone fails to capture the complex learning dynamics across layers. GEM's superior performance validates that jointly modeling both the magnitude and the concentration of learning signals enables more expressive parameter selection.

Effect of Varying Tunable Parameter Ratios. We investigate the performance of GEM under varying tunable parameter ratios: {0.5%, 0.1%, 0.05%, 0.01%}. Figure 3 presents the results on Microsoft Phi-2 across seven tasks, alongside baseline methods reported in Table 3. GEM outperforms all PEFT baselines at roughly the same budget (0.1%). Performance drops sharply below 0.1%, indicating that overly sparse updates cannot adequately adapt the model. GEM even surpasses full fine-tuning performance when tuning only 0.5% or 0.1% of the model parameters. This observation aligns with recent studies (Xu and Zhang 2024; Li et al. 2025) suggesting that structured and prioritized tuning parameter selection can lead to better generalization than indiscriminate full fine-tuning, which may suffer from overfitting or suboptimal adaptation.

Model	Dataset	FFT	LoRA	AdaLoRA	Random	GEM
Phi-2	GSM8k	53.41	42.82	41.47	47.96	**48.78**
	MBPP	47.83	40.24	42.94	40.18	**43.58**

Table 5: Evaluation on domain-specific tasks.

Transferability to Domain-Specific Tasks. Table 5 presents performance comparisons on domain-specific tasks, including mathematical reasoning and code generation. These tasks often require knowledge or reasoning beyond what is captured during pre-training, making them particularly challenging for LLMs to solve without task-specific adaptation. Following Javaheripi et al. (2023), we evaluated GSM8k (Cobbe et al. 2021) in a zero-shot setting using the exact match (EM) metric, and MBPP (Austin et al. 2021) with 5-shot prompting using the pass@3 metric. We exclude OPT-125M and OPT-1.3B from specific task evaluations as these models lack sufficient reasoning and code generation capabilities, resulting in negligible performance.

Across both tasks, all PEFT methods see substantial performance drops relative to full fine-tuning since only about 0.1% of the 2.7-billion parameters in Phi-2 are updated while the model faces strong distribution shifts. Nevertheless, GEM consistently outperforms other PEFT baselines, demonstrating robustness to distribution shifts. Unlike conventional PEFT methods, GEM prioritizes parameters by their gradient-to-weight ratio, measuring how much each should change relative to its original scale, and updates those with the greatest functional impact. This scale-aware, context-adaptive strategy yields superior performance.

5 Conclusion

Our paper studies the impact of two key concepts on fine-tuning performance: scale-aware parameter prioritization and distribution-sensitive, layer-wise parameter budget allocation. We find that the relative weight change is correlated with loss reduction, suggesting that the gradient-to-weight ratio is an effective metric for parameter prioritization. In addition, our study finds that entropy-guided parameter budget allocation improves fine-tuning performance by effectively allocating tunable parameters based on how learning signals are distributed within each layer. Based on our findings, we develop a practical and effective PEFT framework, GEM, and demonstrate its effectiveness through extensive comparative studies. We believe that the proposed GEM framework provides a practical fine-tuning strategy for many real-world LLM applications. An important direction for future work is to develop a theoretical understanding of how gradient-to-weight ratio affects generalization performance.

Acknowledgments

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.RS-2022-00155915, Artificial Intelligence Convergence Innovation Human Resources Development (Inha University)) and National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00452914).

References

Agarwal, A.; Ramesh, S. K.; Sengupta, A.; and Chakraborty, T. 2024. Step-by-Step Unmasking for Parameter-Efficient Fine-tuning of Large Language Models. *CoRR*, abs/2408.14470.

Austin, J.; Odena, A.; Nye, M.; Bosma, M.; Michalewski, H.; Dohan, D.; Jiang, E.; Cai, C.; Terry, M.; Le, Q.; and Sutton, C. 2021. Program Synthesis with Large Language Models. arXiv:2108.07732.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165.

Clark, C.; Lee, K.; Chang, M.-W.; Kwiatkowski, T.; Collins, M.; and Toutanova, K. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of NAACL-HLT 2019*.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. arXiv:2110.14168.

Dagan, I.; Glickman, O.; and Magnini, B. 2005. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges Workshop*, 177–190. Springer.

Das, S. S. S.; Zhang, R. H.; Shi, P.; Yin, W.; and Zhang, R. 2023. Unified Low-Resource Sequence Labeling by Sample-Aware Dynamic Sparse Finetuning. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 6998–7010. Singapore: Association for Computational Linguistics.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems*, volume 36, 10088–10115. Curran Associates, Inc.

Ding, N.; Lv, X.; Wang, Q.; Chen, Y.; Zhou, B.; Liu, Z.; and Sun, M. 2023. Sparse Low-rank Adaptation of Pre-trained Language Models. arXiv:2311.11696.

EV, A.; Biswas, A.; Kalra, S. A.; Mitra, P.; and BASU, B. 2024. XoRA: Expander Adapted LoRA Finetuning. In *NeurIPS 2024 Workshop on Fine-Tuning in Modern Machine Learning: Principles and Scalability*.

Frankle, J.; and Carbin, M. 2019. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations*.

Frantar, E.; Ashkboos, S.; Hoefler, T.; and Alistarh, D. 2023. OPTQ: Accurate Quantization for Generative Pre-trained Transformers. In *The Eleventh International Conference on Learning Representations*.

Gu, N.; Fu, P.; Liu, X.; Shen, B.; Lin, Z.; and Wang, W. 2024. Light-PEFT: Lightening Parameter-Efficient Fine-Tuning via Early Pruning. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 7528–7541. Bangkok, Thailand: Association for Computational Linguistics.

Gunasekar, S.; Zhang, Y.; Aneja, J.; Mendes, C. C. T.; Giorno, A. D.; Gopi, S.; Javaheripi, M.; Kauffmann, P.; de Rosa, G.; Saarikivi, O.; Salim, A.; Shah, S.; Behl, H. S.; Wang, X.; Bubeck, S.; Eldan, R.; Kalai, A. T.; Lee, Y. T.; and Li, Y. 2023. Textbooks Are All You Need. arXiv:2306.11644.

Guo, D.; Rush, A.; and Kim, Y. 2021. Parameter-Efficient Transfer Learning with Diff Pruning. In Zong, C.; Xia, F.; Li, W.; and Navigli, R., eds., *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4884–4896. Online: Association for Computational Linguistics.

He, H.; Ye, P.; Ren, Y.; Yuan, Y.; and Chen, L. 2025. GoRA: Gradient-driven Adaptive Low Rank Adaptation. arXiv:2502.12171.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-Efficient Transfer Learning for NLP. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2790–2799. PMLR.

Hu, E. J.; yelong shen; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Javaheripi, M.; Bubeck, S.; Abdin, M.; Aneja, J.; Bubeck, S.; Mendes, C. C. T.; Chen, W.; Del Giorno, A.; Eldan, R.; Gopi, S.; et al. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Karimi Mahabadi, R.; Henderson, J.; and Ruder, S. 2021. Compacter: Efficient Low-Rank Hypercomplex Adapter Layers. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 1022–1035. Curran Associates, Inc.

Khashabi, D.; Chaturvedi, S.; Roth, M.; Upadhyay, S.; and Roth, D. 2018. Looking Beyond the Surface: A Challenge Set for Reading Comprehension over Multiple Sentences. In *Proceedings of North American Chapter of the Association for Computational Linguistics (NAACL)*.

Kopiczko, D. J.; Blankevoort, T.; and Asano, Y. M. 2024. VeRA: Vector-based Random Matrix Adaptation. In *The Twelfth International Conference on Learning Representations*.

Li, H.; Zhang, X.; Liu, X.; Gong, Y.; Wang, Y.; Chen, Q.; and Cheng, P. 2025. Enhancing Large Language Model Performance with Gradient-Based Parameter Selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(23): 24431–24439.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. arXiv:2101.00190.

Lin, J.; Tang, J.; Tang, H.; Yang, S.; Chen, W.-M.; Wang, W.-C.; Xiao, G.; Dang, X.; Gan, C.; and Han, S. 2024. AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration. In Gibbons, P.; Pekhimenko, G.; and Sa, C. D., eds., *Proceedings of Machine Learning and Systems*, volume 6, 87–100.

Liu, Z.; Lyn, J.; Zhu, W.; Tian, X.; and Graham, Y. 2024. ALoRA: Allocating Low-Rank Adaptation for Fine-tuning Large Language Models. arXiv:2403.16187.

Pfeiffer, J.; Kamath, A.; Rücklé, A.; Cho, K.; and Gurevych, I. 2021. AdapterFusion: Non-Destructive Task Composition for Transfer Learning. In Merlo, P.; Tiedemann, J.; and Tsarfaty, R., eds., *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 487–503. Online: Association for Computational Linguistics.

Pfeiffer, J.; Rücklé, A.; Poth, C.; Kamath, A.; Vulić, I.; Ruder, S.; Cho, K.; and Gurevych, I. 2020. AdapterHub: A Framework for Adapting Transformers. In Liu, Q.; and Schlangen, D., eds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 46–54. Online: Association for Computational Linguistics.

Pilehvar, M. T.; and Camacho-Collados, J. 2018. WiC: 10,000 Example Pairs for Evaluating Context-Sensitive Representations. *CoRR*, abs/1808.09121.

Rajpurkar, P.; Jia, R.; and Liang, P. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In Gurevych, I.; and Miyao, Y., eds., *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 784–789. Melbourne, Australia: Association for Computational Linguistics.

Roemmele, M.; Bejan, C. A.; and Gordon, A. S. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, 1631–1642.

Song, W.; Li, Z.; Zhang, L.; hai zhao; and Du, B. 2024. Sparse is Enough in Fine-tuning Pre-trained Large Language Models. In *Forty-first International Conference on Machine Learning*.

Sung, Y.-L.; Nair, V.; and Raffel, C. 2021. Training neural networks with fixed sparse masks. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713845393.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

Wang, A.; Pruksachatkun, Y.; Nangia, N.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2020. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. arXiv:1905.00537.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*.

Xu, J.; and Zhang, J. 2024. Random Masking Finds Winning Tickets for Parameter Efficient Fine-tuning. In *Forty-first International Conference on Machine Learning*.

Xu, R.; Luo, F.; Zhang, Z.; Tan, C.; Chang, B.; Huang, S.; and Huang, F. 2021. Raise a Child in Large Language Model: Towards Effective and Generalizable Fine-tuning. arXiv:2109.05687.

Xu, S.; and Bu, Z. 2025. Adaptive parameter-efficient fine-tuning via Hessian-informed subset selection. arXiv:2505.12579.

Zaken, E. B.; Ravfogel, S.; and Goldberg, Y. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. arXiv:2106.10199.

Zhang, L.; Zhang, L.; Shi, S.; Chu, X.; and Li, B. 2023a. LoRA-FA: Memory-efficient Low-rank Adaptation for Large Language Models Fine-tuning. arXiv:2308.03303.

Zhang, Q.; Chen, M.; Bukharin, A.; He, P.; Cheng, Y.; Chen, W.; and Zhao, T. 2023b. Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. In *The Eleventh International Conference on Learning Representations*.

Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X. V.; Mihaylov, T.; Ott, M.; Shleifer, S.; Shuster, K.; Simig, D.; Koura, P. S.; Sridhar, A.; Wang, T.; and Zettlemoyer, L. 2022. OPT: Open Pre-trained Transformer Language Models. arXiv:2205.01068.

Zhang, Z.; Zhang, Q.; Gao, Z.; Zhang, R.; Shutova, E.; Zhou, S.; and Zhang, S. 2024. Gradient-based Parameter Selection for Efficient Fine-Tuning. arXiv:2312.10136.

Zhao, M.; Lin, T.; Mi, F.; Jaggi, M.; and Schütze, H. 2020. Masking as an Efficient Alternative to Finetuning for Pretrained Language Models. In Webber, B.; Cohn, T.; He, Y.; and Liu, Y., eds., *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2226–2241. Online: Association for Computational Linguistics.