

GRASP: GRouped Activation Shared Parameterization for Parameter-Efficient Fine-Tuning and Robust Inference of Transformers

Malyaban Bal

School of EECS

The Pennsylvania State University
University Park, PA, USA
mjb7906@psu.edu

Abhroni Sengupta

School of EECS

The Pennsylvania State University
University Park, PA, USA
sengupta@psu.edu

Abstract—Parameter-efficient fine-tuning (PEFT) provides a scalable alternative to full-model adaptation by updating only a small subset of parameters in large pre-trained models. We introduce GRASP — GRouped Activation Shared Parameterization — a lightweight PEFT framework that partitions the D -dimensional token representations of selected layers into $K \ll D$ groups and learns a shared scaling and shifting vector for each group. This grouped modulation reduces the number of trainable parameters significantly while preserving the ability of the model to learn task-specific features. Building on this formulation, we further propose StochGRASP, which learns Gaussian distributions as perturbations to the pre-trained weights rather than deterministic values. This probabilistic parameterization along with a noise-aware loss function formulation enables modelling hardware-level variability in programmed weights and significantly improves robustness under non-ideal inference conditions—an important requirement for deployment on edge-based emerging AI hardware. Across GLUE (RoBERTa-base & RoBERTa-large) and E2E NLG (GPT-2 Medium), GRASP matches or exceeds the performance of established PEFT methods while achieving an order of magnitude reduction in trainable parameters compared to LoRA and BitFit. Under varying levels of noise, StochGRASP consistently outperforms deterministic variants, demonstrating its suitability for energy-efficient and noise-prone hardware platforms.

Index Terms—Large Language Models, Parameter-Efficient Fine-Tuning, Stochastic Modeling

I. INTRODUCTION

Pre-trained transformers form the backbone of modern natural language processing [1], but their large number of trainable parameters presents significant challenges for adaptation—particularly in low-resource scenarios [2]. Moreover, on tasks with limited data, full fine-tuning often leads to overfitting, resulting in reduced generalization and lower inference-time accuracy [3].

To tackle these challenges, parameter-efficient fine-tuning (PEFT) [4] methods have emerged as effective strategies for adapting large models by updating only a small subset of parameters. This approach significantly reduces computational costs, storage requirements, and training time. Widely adopted PEFT techniques such as LoRA [2] and Adapters [5] achieve strong performance but introduce additional matrix multipli-

cations during training and typically require more parameters than methods based on lightweight, element-wise shifting & scaling operations. Notable alternatives in this category include BitFit [6], which fine-tunes only the bias terms, and $(IA)^3$ [3], which learns scaling vectors for selected layers. Although these methods achieve a favorable balance between efficiency and performance, they still incur a trainable parameter cost of the order of $\mathcal{O}(n \times D)$, where n is the number of layers and D represents the hidden dimension (either the model or intermediate size), indicating potential for further compression.

In this paper, instead of learning separate linear scale & shift parameters for each of the D components, we group them into K clusters and assign shared parameters within each group. For chosen linear projection layers where we apply GRASP, we modulate the input activations to the layer using this shared set of parameters. This grouped modulation approach reduces the number of trainable parameters from $\mathcal{O}(n \times D)$ to $\mathcal{O}(n \times K)$, where $K \ll D$. Analogous to how LoRA demonstrates that fine-tuning can be achieved by learning low-rank ($r \ll D$) updates to weight matrices, our method learns only K group-wise scaling and shifting parameters. While LoRA introduces approximately $\mathcal{O}(D \times r)$ trainable parameters per layer (where r is the low rank), our approach requires only $\mathcal{O}(K)$ parameters per layer, offering an even more compact and efficient alternative for PEFT.

While the primary contribution of GRASP lies in reducing the computational overhead of PEFT by drastically lowering the number of trainable parameters on GPUs, a deeper analysis of the layer-wise parameter distributions reveals a consistent emergence of structured, multimodal patterns. Motivated by this observation, we reinterpret PEFT as the task of learning *underlying perturbation distributions* that can be sampled to adapt the frozen pre-trained weights to a new dataset.

Building on this perspective, we introduce **StochGRASP**, a PEFT strategy that learns Gaussian parameter distributions instead of fixed deterministic updates. This distributional formulation yields perturbations that are inherently more resilient to device/circuit-level variability—an important requirement for deployment on noisy or non-ideal hardware. For instance,

emerging energy-efficient AI hardware accelerators based on novel non-volatile memories suffer from multiple sources of noise (device-to-device variations, cycle-to-cycle variations, drift effects) and mitigation of such non-idealities is an active area of research [7], [8]. To enable robust variation-immune inference on such edge AI hardware platforms, we explore a noise-aware fine-tuning objective that regularizes the learned standard deviations toward a target noise profile, encouraging the model to maintain stable performance across a range of hardware noise conditions. The primary contributions of this work are:

- 1) We introduce **GRASP** (GRouped Activation Shared Parameterization), a lightweight PEFT method that partitions the D -dimensional hidden representation into K groups and learns only a shared scaling and shifting pair per group, significantly reducing the number of trainable parameters.
- 2) Through a layer-wise analysis of parameter distributions, we show that decreasing group size transforms the learned parameters from a unimodal to a multimodal structure, highlighting GRASP’s ability to retain task-specific expressivity despite aggressive compression.
- 3) GRASP achieves competitive or superior performance compared to popular PEFT methods (LoRA, BitFit, $(IA)^3$) on both masked (GLUE with RoBERTa-base/large) and causal (E2E with GPT-2 Medium) language modeling tasks, while using orders-of-magnitude fewer parameters.
- 4) Motivated by the multimodal structure, we reformulate PEFT as learning *perturbation distributions* instead of deterministic values, and develop a stochastic variant of GRASP that exhibits improved robustness under hardware-induced noise.

II. RELATED WORKS

Parameter-efficient fine-tuning (PEFT) has emerged as a powerful strategy for adapting large pre-trained models while updating only a small fraction of parameters. Existing PEFT techniques can be broadly categorized into *layer-selective*, *module-based*, and *element-wise* approaches.

Layer-selective methods fine-tune only a subset of transformer layers to reduce training cost. For example, FFTop2 [2] updates only the top two layers of the model while keeping the remaining parameters frozen.

Module-based PEFT introduces small trainable components into the network. Adapters [5] insert bottleneck layers inside transformer blocks and update only these modules during fine-tuning. LoRA [2] replaces full-rank updates with low-rank matrices, substantially reducing the number of trainable parameters while preserving representational flexibility. Other techniques, such as Prefix Tuning [9], optimize continuous prefix vectors at the embedding layer rather than modifying internal weights.

Element-wise modulation methods operate directly on hidden activations and avoid introducing additional matrix multiplications. BitFit [6] tunes only bias terms, whereas

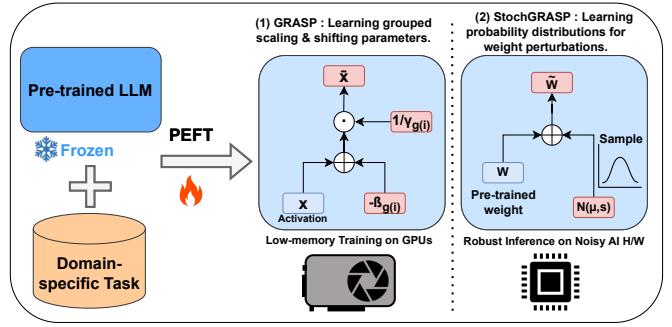


Fig. 1. High-level overview of the proposed PEFT framework. (1) GRASP performs parameter-efficient fine-tuning by learning grouped scaling and shifting parameters over activations, enabling low-memory training with deterministic modulation. (2) StochGRASP extends this concept by learning Gaussian perturbation distributions over weight updates rather than fixed values, coupled with a noise-aware objective that yields significantly more robust inference under hardware-level noise than deterministic baselines.

$(IA)^3$ [3] learns multiplicative scaling vectors applied to selected projections. RED [10] extends this family by learning both scaling and shifting parameters on transformer activations.

Uncertainty-aware or stochastic modeling within transformers is also an active area of research. BayesFormer [11] and related variational dropout methods estimate predictive uncertainty, but their focus is on improving generalization rather than enabling robustness under device-level noise or analog hardware variability. While there have been works on developing variation-resilient models for such analog AI hardware platforms [7], [8], the existing probabilistic models remain limited on small scale vision-datasets and simple convolutional networks. **Our work uniquely bridges PEFT with variation-aware robust inference, addressing a critical gap in existing approaches.**

III. GROUPED ACTIVATION SHARED PARAMETERIZATION

GRASP aims to adapt pre-trained transformer models by freezing the majority of model weights and applying lightweight linear scaling and shifting operations to the hidden state activity of a sub-layer that we want to fine-tune. Unlike prior methods such as BitFit/ $(IA)^3$, which learn individual shift/scale parameters across the entire hidden dimension, GRASP learns only K - pairs of parameters where $K \ll D$. As a result, all D/K dimensions within the same group share the same modulation parameters (Fig. 2a), drastically reducing the number of trainable parameters while achieving competitive results.

Our proposed method introduces no additional matrix multiplications during training, resulting in lower computational overhead compared to methods such as LoRA and Adapters—while also significantly reducing the number of trainable parameters. For each projection layer selected for fine-tuning, we modulate the input activation as follows:

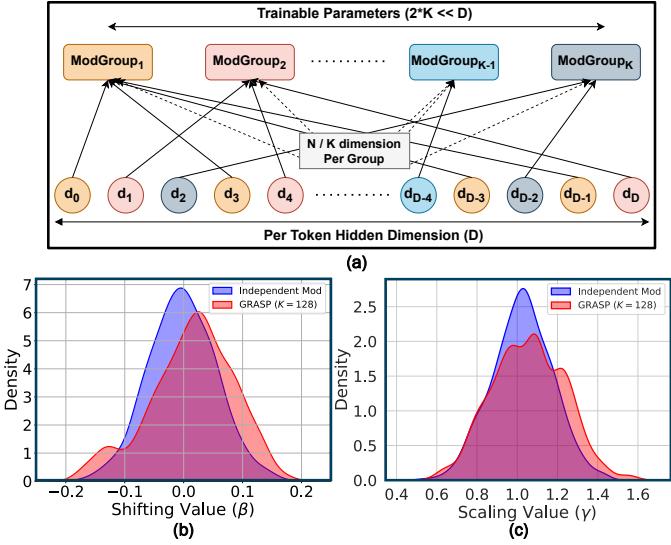


Fig. 2. (a) Random grouping of per-token D -dimensional input activation to a layer into K groups. GRASP learns scaling and shifting parameters per group rather than independently for each component. (b & c) Kernel Density Estimation (KDE) plot showing (b) the distribution of shifting parameter (β) and (c) the distribution of scaling parameter (γ) learned in a selected projection layer (Key) when parameters are learned independently versus using GRASP with $K = 128$. Results are from GLUE CoLA dataset.

$$\tilde{x}_i^\tau = \frac{x_i^\tau - \beta_{g(i)}}{\gamma_{g(i)}} \quad \text{for } i = 1, 2, \dots, D_{in}, \quad (1)$$

$$\tilde{Y} = \tilde{X}\tilde{W}$$

where x_i^τ denotes input activation, i.e., the output of the previous layer, for the i -th dimension of token τ , and \tilde{x}_i^τ represents the entire modulated input. $\tilde{X} \in \mathbb{R}^{L \times D_{in}}$ is the modulated input, $\tilde{W} \in \mathbb{R}^{D_{in} \times D_{out}}$ is the frozen layer weight and $\tilde{Y} \in \mathbb{R}^{L \times D_{out}}$ is the output, where L is sequence length, D_{in} is the activation dimension of the previous layer and D_{out} is the activation dimension of the current layer. $g(i) \in \{1, 2, \dots, K\}$ maps each dimension component in D_{in} to one of K groups, with $\gamma_{g(i)}$ and $\beta_{g(i)}$ denoting the learnable scaling and shifting parameters shared across all components within group $g(i)$.

A. Random Grouping

Instead of doing any computationally expensive pre-processing to learn the grouping function ($g(i)$), we propose an effective strategy to randomly group dimensions. The method used in this paper is described below,

$$\begin{aligned} \tilde{\gamma} &= (\gamma^{\oplus m})_{1:D}, \quad \tilde{\beta} = (\beta^{\oplus m})_{1:D} \\ \hat{\gamma} &= \tilde{\gamma}[\pi], \quad \hat{\beta} = \tilde{\beta}[\pi] \end{aligned} \quad (2)$$

Here, $\gamma, \beta \in \mathbb{R}^K$ are the learned scaling and shifting parameters for $K \ll D$ groups, where D represents the input hidden dimension. These vectors are first repeated $m = \lceil D/K \rceil$

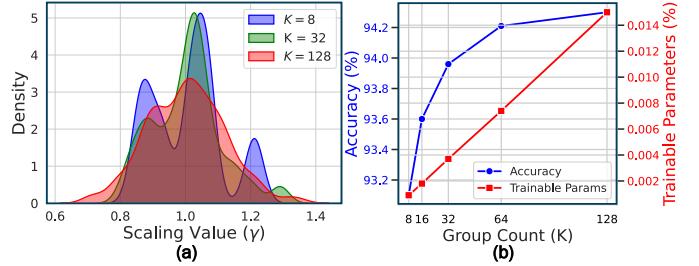


Fig. 3. (a) KDE plots of scaling parameter (γ) distribution in the same projection layer (Key) for GRASP with varying group sizes $K \in \{8, 32, 128\}$. Smaller K values produce distinct parameter clusters (modes), enabling the model to efficiently learn downstream task (GLUE SST-2) with significantly fewer trainable parameters. (b) Graph showing trade-off between accuracy and trainable parameters (%) on SST-2 task using GRASP with different values of K .

times and truncated to length D , yielding $\tilde{\gamma}, \tilde{\beta} \in \mathbb{R}^D$. A fixed permutation π over $\{1, \dots, D\}$ is then applied to obtain the permuted vectors $\hat{\gamma} = \tilde{\gamma}[\pi]$ and $\hat{\beta} = \tilde{\beta}[\pi]$. This effectively groups D/K neurons to share a pair of learnable linear scaling and shifting parameters. For stable training, we initialize all γ values to 1 and all β values to 0.

B. Group Granularity

The number of groups (K) play a critical role in controlling the number of trainable parameters, as further explored in the Results section. Interestingly, reducing K not only reduces trainable parameters but also encourages it to capture more salient features of the underlying data. We empirically validate this in Fig. 2b & c and Fig 3a. When parameters are learned independently (no grouping), their layer-wise distribution is approximately uni-modal Gaussian. As K decreases, multi-modes (mixture of Gaussians) arises in the parameter distribution, suggesting that the model is discovering and encoding distinct latent structures.

This effect becomes increasingly pronounced as K is reduced from 128 to 8 (Fig. 3a), requiring the model to learn critical task-relevant features under tighter parameter constraints. In Fig. 3b, we demonstrate the trade-off between total trainable parameters and downstream task accuracy. Due to the observed non-linear relationship, we find that permitting a small drop in task-specific accuracy (i.e., $\leq 0.7\%$, from 94.3% to 93.6% on SST-2) enables a substantial additional reduction in trainable parameters—over 8×, achieved by decreasing $K = 128$ to $K = 16$.

C. Stochastic GRASP

1) Motivation: The multi-modal feature space, which can be seen in Fig. 3a, becomes more prominent as we reduce K and motivated our stochastic GRASP formulation namely, StochGRASP. From empirical analysis, we hypothesize that the pre-trained weights of each modulated layer are perturbed by K -learnt Gaussians upon fine-tuning and our goal is to learn its parameters, i.e. mean (μ) and standard deviation (σ). In formulating StochGRASP, we aim to understand whether

TABLE I

GLUE BENCHMARK COMPARISON OF GRASP-BASED FINE-TUNING ON ROBERTA-BASE (125M) AND ROBERTA-LARGE (355M) AGAINST STATE-OF-THE-ART PARAMETER-EFFICIENT FINE-TUNING METHODS. %PARAM DENOTES THE PERCENTAGE OF TRAINABLE PARAMETERS RELATIVE TO THE TOTAL MODEL PARAMETERS.

Model	Method	%Param	QNLI	SST-2	MNLI	CoLA	MRPC	STS-B	RTE	QQP	Avg.
RoBERTa-Base	Full-FT [12]	100%	92.3	94.2	86.7	61.1	92.5	90.6	77.4	91.9	85.8
	Adapters [2]	0.3%	93.1	94.2	87.1	60.8	88.5	89.7	71.5	90.2	84.4
	LoRA [2]	0.3%	93.3	95.1	87.5	63.4	89.7	91.5	86.6	90.8	87.2
	BitFit [12]	0.09%	91.3	93.7	85.0	61.8	92.0	90.8	77.8	87.3	84.9
	(IA) ³ [10]	0.05%	91.1	93.4	85.4	57.8	86.4	88.5	73.5	88.5	83.1
	RED [10]	0.02%	90.7	93.9	83.9	61.0	89.2	90.4	78.0	87.2	84.3
	GRASP ($K = 128$)	0.015%	92.0	94.3	85.2	64.2	92.2	90.9	79.1	89.5	85.9
	GRASP ($K = 32$)	0.004%	90.4	93.8	83.4	61.9	91.5	90.3	77.2	88.2	84.6
RoBERTa-Large	Full-FT [12]	100%	94.7	96.4	90.2	68.0	94.0	92.4	86.6	92.2	89.3
	LoRA [2]	0.3%	94.9	96.2	90.6	68.2	93.4	92.6	87.4	91.6	89.4
	RED [10]	0.02%	93.5	96.0	89.5	68.1	90.3	91.3	86.2	88.9	87.9
	VeRA [13]	0.02%	94.4	96.1	—	68.0	90.9	91.7	85.9	—	—
	GRASP ($K = 64$)	0.005%	94.5	96.1	88.9	67.0	93.0	91.8	85.9	89.8	88.4

distributional modeling of perturbations can yield robustness similar to that of probabilistic architectures.

Our stochastic formulation, relates GRASP’s activation-space shifts to an equivalent structured perturbation in weight parameter space. Consider a frozen linear sub-layer

$$Y = X\mathbf{W} + \mathbf{b}, \quad W \in \mathbb{R}^{D_{\text{in}} \times D_{\text{out}}}, \quad b \in \mathbb{R}^{D_{\text{out}}}. \quad (3)$$

In the simplified (shift-only) GRASP setting, the input activations are modified group-wise before the linear map. Let each input dimension j belong to a group $g(j) \in \{1, \dots, K\}$, with a learned shift $t_{g(j)}$. Defining the shift vector $t \in \mathbb{R}^{D_{\text{in}}}$ via

$$\tilde{x}_j = x_j + t_{g(j)} \implies \tilde{X} = X + T,$$

the transformed output becomes

$$\tilde{Y} = \tilde{X}\mathbf{W} + \mathbf{b} = (X + T)\mathbf{W} + \mathbf{b} = X\mathbf{W} + \underbrace{(TW + \mathbf{b})}_{b_{\text{eff}}}. \quad (4)$$

Thus, applying group-wise shifts in the activation space is *functionally equivalent* to introducing an effective bias term $b_{\text{eff}} = \mathbf{b} + TW$, which is broadcast across all input rows, while \mathbf{W} and \mathbf{b} remains frozen.

2) *Formulation:* Empirically, as the group size decreases, the learned GRASP shifts $\{t_{g(j)}\}_{j=1}^{D_{\text{in}}}$ exhibit clear multi-modal structure across layers. Since the mapping $T \mapsto TW$ in Eqn. 4 is linear, any multi-modal distribution over the group-wise shifts T induces a corresponding multi-modal distribution over the effective bias b_{eff} . This observation motivates extending GRASP beyond learning deterministic values to instead learning distributions of perturbations that are injected into the pre-trained weight parameters.

While Eqn. 1 applies deterministic scaling and shifting to activations, our proposed StochGRASP instead models the trainable update as a distribution over weight perturbations. We assume the learnt perturbation to follow underlying *Gaussian*

Distributions. After training, the effective weights can be written as:

$$\begin{aligned} \tilde{W}_{i,j} &= \mathbf{W}_{i,j} + \Delta\mathbf{W}_{i,j}, \quad i \in \{1, \dots, D_{\text{in}}\}, \quad j \in \{1, \dots, D_{\text{out}}\}, \\ \Delta\mathbf{W}_{i,j} &\sim \mathcal{N}(\mu_{f(i), g(j)}, \sigma_{f(i), g(j)}), \\ \tilde{W}_{i,j} &\sim \mathcal{N}(\mathbf{W}_{i,j} + \mu_{f(i), g(j)}, \sigma_{f(i), g(j)}). \end{aligned} \quad (5)$$

where $g(j) \in \{1, \dots, K\}$ and $f(i)$ controls how many distinct Gaussian distributions are learned along rows of weight matrix: Setting $f(i) = i$ learns a separate distribution for each input row, yielding $O(D * K)$ trainable parameters. Setting $f(i) = 1$ shares a single distribution across all rows, with parameter count $O(K)$.

Thus, rather than learning deterministic weight updates, we formulate PEFT as the stochastic task of learning Gaussian distributions whose samples generate perturbations to align (fine-tune) the pre-trained model with a given dataset. This not only provides a principled explanation for the emergent multi-modal behavior but also yields a model that is inherently more resilient to noise, particularly in edge-level or compute-constrained hardware settings.

IV. RESULTS

Following prior work [6], we evaluate our proposed method on both masked and causal language modeling tasks. Specifically, we fine-tune RoBERTa-Base and RoBERTa-Large [14] on eight tasks of the GLUE benchmark [15] and GPT2-Medium [16] on the E2E NLG Challenge [17], a data-to-text generation task. The experiments were done on Nvidia RTX A5000 GPUs (8) each with 24GB memory.

A. Natural Language Understanding Tasks

We present the results on GLUE benchmark using RoBERTa-Base and RoBERTa-Large models in Table I. We report results on single sentence tasks such as CoLA and SST-2, on similarity-based tasks such as STS-B, MRPC and QQP and natural language inference tasks such as RTE, QNLI, MNLI. We report matthews correlation for CoLA, pearson

TABLE II
COMPARISON OF DIFFERENT FINE-TUNING METHODS ON E2E NLG CHALLENGE USING GPT-2 MEDIUM. %PARAM IS RELATIVE TO FULL FINE-TUNING OF GPT-2 MEDIUM (354.9M). * RESULTS TAKEN FROM [2]. † RESULTS TAKEN FROM [10].

Method	%Param	BLEU	NIST	METEOR	ROUGE-L	CIDEr
Full-FT*	100%	68.2	8.62	46.2	71.00	2.47
FTTop2*	7.10%	68.1	8.59	46.00	70.80	2.41
AdapterL*	3.12%	68.9	8.71	46.10	71.30	2.47
LoRA*	0.1%	70.4	8.85	46.80	71.80	2.53
Prefix Tuning†	0.23%	63.92	8.26	41.81	66.86	2.03
(IA) ³ †	0.05%	63.63	7.99	40.49	66.36	1.89
GRASP ($K=384$)	0.02%	68.64	8.76	45.00	69.47	2.37
Adapter (rank 1)†	0.07%	63.76	8.37	42.74	66.70	2.09
Adapter-FFN (rank 1)†	0.02%	62.99	8.09	40.88	66.39	1.98
LoRA (rank 1)†	0.03%	64.51	8.38	44.78	67.35	2.28
RED†	0.01%	64.86	8.36	44.99	67.62	2.28
GRASP ($K=64$)	0.003%	66.03	8.49	43.37	68.15	2.27

correlation for STS-B, F1-score for MRPC and accuracy for all other tasks.

For all the tasks in GLUE, we keep the maximum sequence length as 128. We use batch size of 16 for RTE, MRPC, MNLI and batch size of 32 for the rest of the datasets. The learning rate is set to 8×10^{-4} for CoLA, RTE, and STS-B; 5×10^{-4} for MNLI, QNLI, and QQP; and 1×10^{-4} for SST-2 and MRPC. RoBERTa-base has 12 layers with hidden dimension 768 and the intermediate dimension being 3072. RoBERTa-large has 24 layers with a hidden dimension of 1024 and an intermediate dimension of 4096.

As highlighted in Table I, our method surpasses prior approaches that rely on element-wise modulation, such as BitFit, (IA)³, and RED. Remarkably, with only $K = 32$, GRASP attains comparable or superior performance while reducing the number of trainable parameters by $75 \times$ relative to LoRA, $22.5 \times$ relative to BitFit, and $12.5 \times$ relative to (IA)³.

B. Natural Language Generation Tasks

E2E dataset comprises data from restaurant domain and consists of 42K training, 4.6K validation and 4.6K testing samples. It is primarily used as data-to-text generation task.

For the E2E NLG Challenge, we use a train batch size of 4 and learning rate of 5×10^{-4} . We do beam-search during inference with a beam width of 10. The model dimension in GPT-2 Medium is 1024 and the intermediate dimension is 4096.

In the first part of Table II, we compare GRASP with several popular PEFT strategies and demonstrate that it achieves competitive performance on generative tasks (E2E NLG Challenge), while using significantly fewer trainable parameters. To emphasize suitability of GRASP for low-resource environments, we compare it against low-rank LoRA and Adapters (with rank = 1), as well as the RED method in second part of Table II. GRASP outperforms all of these approaches while requiring substantially fewer trainable parameters—up to **10x fewer than LoRA with rank = 1**.

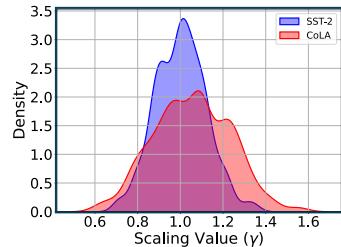


Fig. 4. (a) KDE plots of scaling parameter distribution of the same projection layer (Key) for GRASP on CoLA and SST-2.

C. GRASP: Ablation Studies

We perform ablation studies to isolate the impact of different factors—namely, random grouping, the selection of layers for GRASP and the use of scaling versus shifting —on overall performance. The results are summarized next.

TABLE III
PERFORMANCE OF GRASP ($K = 128$) ON SELECTED GLUE TASKS USING ROBERTA-BASE, FOR 3 DIFFERENT INITIAL SEEDS RESULTING IN DIFFERENT GROUPING CONFIGURATIONS.

Method	STS-B	QNLI	CoLA	SST-2
GRASP	$90.89 \pm .03$	$92.00 \pm .19$	$64.19 \pm .15$	$94.27 \pm .11$

1) **Random Grouping Robustness:** Since we employ a random grouping strategy where random dimensions are grouped together, we do further exploration on what role different initial seeds play in model performance. We empirically report in Table III that the performance of the learnt model remains consistent across different starting configurations. This validates that our proposed method enables dynamic adaption of learnt parameters based on the groups formalized without any degradation in performance.

2) **Layer Selection:** In Table IV, we conduct an experiment to compare the performance of GRASP when applied in three different ways: (1) to all linear sub-layers of an encoder block—similar to BitFit, but excluding LayerNorm layers; (2) selectively to specific components such as the key (K), value (V), and second feedforward projection ($FF2$) layers, as in

TABLE IV
ABLATION STUDY ON THREE GLUE DATASETS EVALUATING THE IMPACT OF APPLYING GRASP ($K=128$) TO DIFFERENT LAYERS.

Layers	% Params	RTE	SST-2	STS-B
All Linear	0.015%	79.1	94.3	90.9
$K, V, FF2$	0.008%	78.3	94.1	90.4
$FF2$	0.003%	78.0	94.0	90.0

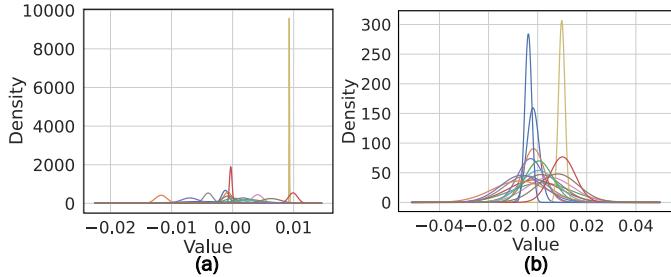


Fig. 5. (a) Gaussian distributions (perturbations) learnt without the modified objective, and (b) distributions learned with the proposed objective (Eqn. 6); both learn $K = 16$ distributions per layer.

(IA)³; and (3) only to the final $FF2$ layer. By only limiting GRASP to $FF2$ layer, we achieve competitive performance while reducing total trainable parameters significantly.

TABLE V
ABLATION STUDY ON THREE GLUE DATASETS ISOLATING THE IMPACT OF SCALING & SHIFTING USING GRASP ($K=128$).

Method	RTE	SST-2	STS-B
without shifting	77.9	93.0	90.1
without scaling	78.7	93.8	90.4

3) **Effect of Scaling & Shifting:** In Table V, we isolate the effect of learning scaling and shifting parameters. We find learning only the shift parameter results in improved performance compared to learning only scaling.

4) **Effect of Tasks on Parameter Distribution:** We further analyze how the per-layer parameter distributions vary across tasks of different difficulty levels. As shown in Fig. 4a comparatively harder task such as CoLA induces a parameter distribution with more pronounced multi-modality than an easier task like SST-2, even when using the same group size ($K = 128$). This suggests that GRASP adapts its learned perturbations to the underlying task complexity, effectively capturing task-critical structure despite operating with a limited number of trainable parameters.

D. Robustness of StochGRASP

As discussed in Section III-C, StochGRASP reformulates PEFT as the task of learning a probability distribution over the weight perturbations required for downstream adaptation. To further align the learned perturbations with hardware-level noise characteristics (e.g., variability in analog non-volatile memory devices), we introduce an additional regularization term that encourages the learned standard deviations to match

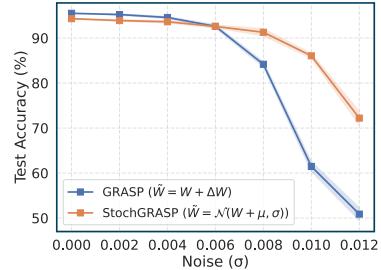


Fig. 6. Robustness comparison between StochGRASP and a deterministic variant of GRASP on the SST-2 evaluation set using RoBERTa-large. We inject varying levels of hardware-level noise during inference and report performance averaged over three random seeds, each corresponding to a distinct noise realization.

a hardware-motivated target value. The modified training objective becomes:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \sum_{\ell=1}^L \sum_{i=1}^{D_{\text{in}}^{(\ell)}} \sum_{j=1}^{D_{\text{out}}^{(\ell)}} \left(\sigma_{f(i),g(j)}^{(\ell)} - \sigma_{\text{target}} \right)^2 \quad (6)$$

where $\mathcal{L}_{\text{task}}$ denotes the original fine-tuning loss, $\sigma^{(\ell)}$ represents the learned standard deviation vector for layer ℓ , σ_{target} is the desired hardware-aligned noise level, and $\lambda \in [0.01, 0.1]$ controls the strength of the regularization. During training, we initialize the mean of the learned Gaussian perturbations as 0, and set the standard deviation as $\sigma \in [0.001, 0.005]$, resulting in a stable initialization. Fig. 5b shows the learned parameter distributions after applying the proposed loss function (with $\sigma_{\text{target}} = 0.05$). Compared to Fig. 5a (without the modified objective), we observe a clear increase in the learned standard deviations (σ), indicating that the objective explicitly encourages broader distributions that provide greater robustness to parameter variability (noise) during inference.

In Fig. 6, we assess the robustness of StochGRASP under varying levels of simulated noise and compare it against a deterministic variant of the GRASP baseline. For this experiment, we train two separate models : (i) a deterministic GRASP model that learns $K = 64$ fixed shift parameters per row of each modulated linear layer, and (ii) our StochGRASP variant, which instead learns K Gaussian distributions per row of each modulated layer. During inference, both models are evaluated under varying levels of injected Gaussian noise. For StochGRASP, instead of sampling from the learned standard deviation, we replace it with the standard deviation corresponding to the injected noise level. This situation is equivalent to deploying the learnt mean values of the fine-tuned weight parameters to an edge AI hardware accelerator that is associated with an independent noise for each programmed weight. Each setting is tested over three random seeds to capture variability in noise realizations. Since, StochGRASP learns *distributions* rather than fixed values, it models weight perturbations during training and internalizes this variability. Specifically under normal noise ($\sigma = 0.01$) StochGRASP maintains accuracy of $\geq 86\%$ while the deterministic

variants deteriorates to 61.5% on SST-2 dataset using RoBERTa-large model. As a result, StochGRASP demonstrates markedly higher tolerance to hardware-level noise, yielding significantly better performance across all disturbance levels.

V. CONCLUSIONS

We introduced GRASP, a lightweight PEFT method that uses group-wise activation modulation to drastically reduce trainable parameters while preserving strong performance. We showed that aggressive compression (small K) yields multimodal parameter structures, motivating a stochastic reformulation. Building on this, StochGRASP learns Gaussian perturbation distributions and, when paired with a noise-aware objective, provides markedly improved robustness under inference-time noise. Together, GRASP and StochGRASP offer scalable, hardware-resilient PEFT strategies well-suited for efficient deployment of large pre-trained models. Future work can involve moving beyond simulated noise and explore training StochGRASP using hardware-informed noise-variability models, enabling deployment on real edge AI hardware chips.

REFERENCES

- [1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.
- [2] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “Lora: Low-rank adaptation of large language models. arxiv 2021,” *arXiv preprint arXiv:2106.09685*, 2021.
- [3] H. Liu, D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C. A. Raffel, “Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 1950–1965, 2022.
- [4] L. Xu, H. Xie, S.-Z. J. Qin, X. Tao, and F. L. Wang, “Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment,” *arXiv preprint arXiv:2312.12148*, 2023.
- [5] C. Poth, H. Sterz, I. Paul, S. Purkayastha, L. Engländer, T. Imhof, I. Vulić, S. Ruder, I. Gurevych, and J. Pfeiffer, “Adapters: A unified library for parameter-efficient and modular transfer learning,” *arXiv preprint arXiv:2311.11077*, 2023.
- [6] E. B. Zaken, S. Ravfogel, and Y. Goldberg, “Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models,” *arXiv preprint arXiv:2106.10199*, 2021.
- [7] O. Yousuf, B. D. Hoskins, K. Ramu, M. Fream, W. A. Borders, A. Madhavan, M. W. Daniels, A. Dienstfrey, J. J. McClelland, M. Lueker-Boden *et al.*, “Layer ensemble averaging for fault tolerance in memristive neural networks,” *Nature Communications*, vol. 16, no. 1, p. 1250, 2025.
- [8] B. Manna, A. Saha, Z. Jiang, K. Ni, and A. Sengupta, “Variation-resilient fetef-based in-memory computing leveraging probabilistic deep learning,” *IEEE Transactions on Electron Devices*, vol. 71, no. 5, pp. 2963–2969, 2024.
- [9] X. L. Li and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” *arXiv preprint arXiv:2101.00190*, 2021.
- [10] M. Wu, W. Liu, X. Wang, T. Li, C. Lv, Z. Ling, J. Zhu, C. Zhang, X. Zheng, and X. Huang, “Advancing parameter efficiency in fine-tuning via representation editing,” *arXiv preprint arXiv:2402.15179*, 2024.
- [11] K. A. Sankararaman, S. Wang, and H. Fang, “Bayesformer: Transformer with uncertainty estimation,” *arXiv preprint arXiv:2206.00826*, 2022.
- [12] E. B. Zaken, Y. Goldberg, and S. Ravfogel, “BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, S. Muresan, P. Nakov, and A. Villavicencio, Eds. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 1–9. [Online]. Available: <https://aclanthology.org/2022.acl-short.1/>
- [13] D. J. Kopiczko, T. Blankevoort, and Y. M. Asano, “Vera: Vector-based random matrix adaptation,” *arXiv preprint arXiv:2310.11454*, 2023.
- [14] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019.
- [15] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “Glue: A multi-task benchmark and analysis platform for natural language understanding,” *arXiv preprint arXiv:1804.07461*, 2018.
- [16] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [17] O. Dušek, J. Novikova, and V. Rieser, “Evaluating the state-of-the-art of end-to-end natural language generation: The e2e nlg challenge,” *Computer Speech & Language*, vol. 59, pp. 123–156, 2020.