GAP: GRADIENT-ALIGNED DUAL PERTURBATION FOR OFFSITE-TUNING LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adapting language models across organizations sometimes requires raw data and complete original models to remain within their respective owners. Offsite-tuning addresses this situation. The model owner supplies an offsite model with a weakened emulator and trainable adapters; the data owner freezes the emulator, tunes only the adapters locally, and returns them for insertion into the complete model. The challenge is to preserve plug-in utility while maintaining an obvious performance gap over the offsite model and the tuned offsite model, which encourages the upload of the adapters to get better performance. Alignment between the emulator and the complete model is central to this trade-off. Prior approaches align representations or preserve gradients through prescribed compression operations. We introduce Gradient-Aligned dual Perturbation (**GAP**), a dual-perturbation approach that expands optimization beyond limited compression choices. Training-based gradient-aligned perturbation learns adversarial weight changes while matching adapter gradients at virtual gradient descent steps; compression-based gradient-aligned perturbation uses Block Influence, feature-direction change between the input and output of a layer, to select low-disruption layer for deletions. Both stages use only support set at the model owner side. Across eight benchmarks, GAP improves macro Plug-in performance over OT, the strongest baseline by 3.59% and 4.53% on Llama-3.2-3B and Llama-3.1-8B respectively with larger performance gap. We theoretically justify GAP's effectiveness and empirically demonstrate its generalization across dense and MoE backbones.

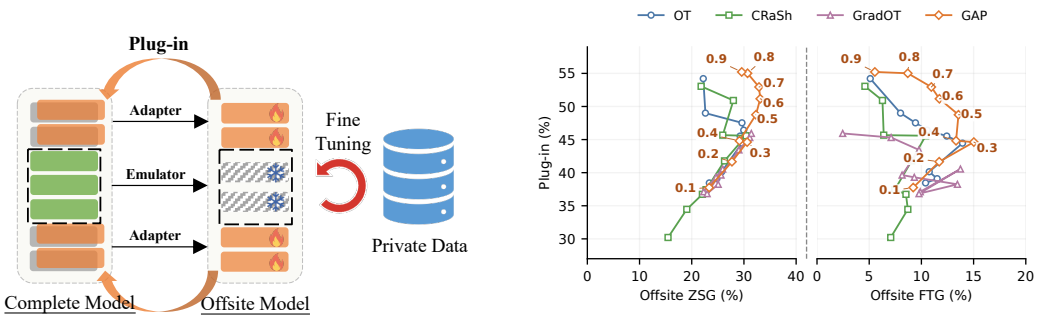
1 INTRODUCTION

Large language models perform strongly across many tasks (Devlin et al., 2019; Radford et al., 2019; Touvron et al., 2023; Wies et al., 2023), yet deploying them in privacy-sensitive domains often requires training on domain-specific data that must remain with their owner. Centralized fine-tuning is frequently infeasible because data owners cannot share raw data under privacy regulations, and model owners cannot release complete proprietary weights due to commercial constraints (Ye et al., 2024; Chua et al., 2023; Li et al., 2020a). This mismatch creates a persistent deployment bottleneck in sensitive, real-world settings.

As shown in Figure 1a, offsite-tuning follows an exchange protocol that keeps raw downstream data with the data owner and the complete original model with the model owner. The model owner derives a shareable offsite model by replacing the pretrained model's middle layers with a weaker **emulator**, while keeping the top and bottom layers as trainable **adapters**. The data owner receives this offsite model and fine-tunes only the adapters on local private data. The updated adapters are then returned to the model owner and plugged back into the original full model to produce the final plug-in model.

We evaluate this offsite-tuning process with four checkpoints: (i) *Pretrained Zero-Shot (ZS)*, the zero-shot performance of the original complete model; (ii) *Offsite ZS*, the zero-shot performance of the offsite model before adapter tuning on private data; (iii) *Offsite Fine-Tuned (FT)*, the performance of the offsite model after adapter tuning; (iv) *Plug-in*, the performance of the complete model after inserting the returned adapters. Furthermore, we define (v) *Offsite Zero-Shot Gap (ZSG)* $=$ *Plug-in* $-$ *Offsite ZS*, the performance gap between the final plug-in model and the untuned offsite model, which encourages fine-tuning on the offsite model; (vi) *Offsite Fine-Tuned Gap (FTG)* $=$ *Plug-in* $-$ *Offsite FT*, the benefit of returning the adapters instead of using the tuned offsite model.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Offsite-tuning.

(b) Gap-utility trade-off.

Figure 1: (a) Offsite-tuning includes weakened emulator construction, offsite model fine-tuning and adapter plug-in. (b) Plug-in scores versus Offsite ZSG (left) and Offsite FTG (right), averaged over OpenBookQA, ARC-E/C, and WebQuestions. Gaps are defined in Section 1 and expressed in %. Each curve contains nine emulator ratios from 0.1 to 0.9, labeled on GAP.

This setup creates a **gap-utility trade-off**. We seek high plug-in performance together with an obvious gain over the available reference models. Perturbing the emulator blindly is indeed possible to create larger gaps by reducing the capabilities of the offsite model, but can also disrupt the plug-in utility of the updates learned around it, which in fact determines whether it is worthwhile to perform offsite-tuning rather than directly using the original complete model. How to construct the perturbed emulator while maintaining alignment between the offsite model and the complete model is therefore central to improving the gap-utility trade-off.

Existing methods mainly construct emulators through compression-based aligned perturbation, which constructs the weakened emulator by compressing the middle layers of the complete model. OT (Xiao et al., 2023) combines uniform layer dropping with hidden-state distillation, while CRaSh (Zhang et al., 2023) uses representation-based clustering and layer sharing. These representation criteria characterize forward alignment. Gradient alignment attempts to align the gradients of the adapters when trained on the offsite model and the complete model, which more directly characterizes plug-in utility of the updates learned on the emulator. GradOT (Yao et al., 2025a) targets gradient preservation through local compression scores for rank decomposition and channel pruning. In existing methods, the solution space of perturbations are constrained by these compression operations, which can only design a scoring method for a fixed set of compression combinations to select the relatively better ones, thus leads to suboptimal performance. We instead introduce training-based gradient-aligned perturbation: optimizing the middle-layer weights directly provides additional degrees of freedom for weakening the emulator while maintaining the utility of adapter updates.

We propose Gradient-Aligned Dual Perturbation **GAP** with two complementary stages: (1) training-based gradient-aligned perturbation learns adversarial middle-layer changes while aligning adapter gradients at temporary tuning states; (2) compression-based gradient-aligned perturbation further compresses the perturbed model by deleting layers with low Block Influence (BI), feature-direction change between the input of a layer and the input of the next layer, for limiting the disruption caused by layer deletion. Our contributions are threefold:

- We introduce training-based gradient-aligned perturbation and combine it with lightweight compression-based gradient-aligned perturbation, expanding the perturbation family beyond prescribed compression operations to improve the gap-utility trade-off as shown in Figure 1b.
- We establish bounds on adapter-gradient mismatch between the offsite model and the complete model throughout a downstream fine-tuning region, connecting the two perturbation stages to transferred updates.
- Across eight benchmarks, GAP improves mean Plug-in performance over the strongest baseline by 3.59% and 4.53% on the two main Llama models, with Offsite ZSG and FTG at the same level. Experiments further demonstrate GAP's generalization across dense and MoE backbones.

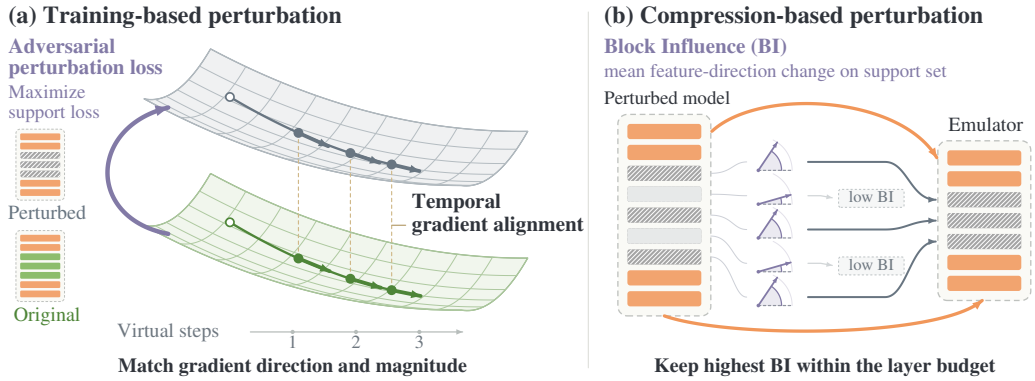


Figure 2: Mechanisms of GAP. (a) Adversarial perturbation raises the support loss by changing middle-layer weights, while temporal alignment matches gradient directions and magnitudes at virtual gradient descent steps of adapters. Dashed links connect matched steps on schematic loss surfaces. (b) BI averages the input-output cosine dissimilarity over support tokens. The highest-BI middle layers are retained within the budget while low-BI layers are removed. Adapters and the retained layers' order are preserved.

2 METHOD

As shown in Figure 2, GAP constructs an emulator through two complementary stages: training-based gradient-aligned perturbation explores adversarial middle-layer weight changes, and compression-based gradient-aligned perturbation selects which perturbed layers to retain. Together they target plug-in utility and the performance gaps defined in Section 1. All model-owner-side operations use only a support set, without downstream training, validation, or test data.

2.1 TRAINING-BASED GRADIENT-ALIGNED PERTURBATION

Write the original model as (a_0, θ_0), where a_0 denotes the boundary adapter layers and θ_0 the middle layers. We initialize $\theta = \theta_0$ and optimize only the middle layers. For support batch B, let $\ell_B(a, \theta)$ denote the language-modeling loss.

To align the gradient beyond the initial adapter state, we vary the number of virtual gradient descent steps:

$$\widetilde{a}^{(k)} = \mathcal{U}^{(k)}_{\theta,B}(a_0), \qquad k \in \{1, 2, 3\}, \tag{1}$$

where $\mathcal{U}^{(k)}_{\theta,B}$ denotes the adapters after training k virtual gradient descent steps on the emulator. The step count cycles across batches; each batch evaluates alignment once at its selected k. Identical adapter increments are applied to the reference model so that both gradients are measured at the same state.

For adapter layer i, define $u_i = \nabla_{a^{(i)}} \ell_B(\widetilde{a}^{(k)}, \theta_0)$ and $v_i = \nabla_{a^{(i)}} \ell_B(\widetilde{a}^{(k)}, \theta)$. The gradient alignment loss matches their directions and magnitudes using squared discrepancies:

$$\mathcal{L}^{(k)}_{\text{grad}} = \frac{1}{q} \sum_{i=1}^{q} \left[\big(1 - \cos(u_i, v_i)\big)^2 + \left(\log \frac{\|v_i\|}{\|u_i\|} \right)^2 \right]. \tag{2}$$

Equation 2 highlights the two alignment targets, while the form for stable training used in experiments is given in Appendix D.

The training-based gradient-aligned perturbation combines the gradient alignment loss with the adversarial perturbation loss:

$$\mathcal{J}_B(\theta) = -\ell_B(\widetilde{a}^{(k)}, \theta) + \lambda \mathcal{L}^{(k)}_{\text{grad}}. \tag{3}$$

The adversarial perturbation loss $-\ell_B$ learns adversarial weight changes to explore strong perturbation with low offsite model capacity throughout the entire middle layers parameter space, while the solution

space of existing methods are restricted by their compression operations. Meanwhile, the gradient alignment loss encourages transferable adapter updates by keeping the gradient alignment between the offsite model and the complete model during the adversarial perturbation. The reference gradients and virtual gradients will not actually update the adapters. The adapters are restored after each B. Only θ is permanently changed.

2.2 Compression-Based Gradient-Aligned Perturbation

The second stage introduces a structural perturbation on the model perturbed by training-based gradient-alignment perturbation for lower Offsite ZS and Offsite FT. Let $h_{l,t}$ represent the input of layer l. We score each layer of the perturbed model using Block Influence (BI) (Men et al., 2024)

$$\mathrm{BI}_l = \mathbb{E}_t[1 - \cos(h_{l,t}, h_{l+1,t})]\,, \tag{4}$$

where the expectation is computed over all tokens t in the support set on the model owner side. Let $\mathcal{I}$ be the middle-layer indices and b the retained-layer budget. We select

$$K \in \underset{J\subseteq\mathcal{I},\,|J|=b}{\arg\max} \sum_{l\in J} \mathrm{BI}_l, \qquad \theta_{\mathrm{emu}} = \mathrm{Prune}_K(\theta), \tag{5}$$

where pruning retains the selected middle blocks in their original order and keeps all boundary adapters. Low-BI deletion targets a small disturbance to the trained emulator. Under the regularity conditions in Appendix D, it contributes to controlling the resulting gradient-alignment error of adapters. Appendix E explains the advantages of layer selection with BI relative to uniform OT and clustering-based CRaSh.

The data owner tunes only the adapters around the frozen emulator and returns them to the model owner, who inserts them into the original complete model. Raw downstream data remain with the data owner, and the complete original model remains with the model owner. Higher Plug-in performance than Offsite ZS provides an incentive for fine-tuning while higher Plug-in performance than Offsite FT provides an incentive for this return. Algorithm 1 summarizes the emulator construction process. Section 3 analyzes gradient alignment of GAP during downstream fine-tuning, with proofs in Appendix D.

Algorithm 1 GAP Emulator Construction

1: **Input:** original model (a_0, θ_0), owner support $\mathcal{S}$, middle-layer budget b; initialize $\theta \leftarrow \theta_0$
2: **Training-based perturbation** (Equations 1–3)
3: **for** each support batch B **do**
4: Choose the scheduled step count k from Equation 1
5: Form matched virtual adapters $\widetilde{a}^{(k)}$ by Equation 1
6: Compute the gradient alignment loss by Equation 2
7: Combine the adversarial perturbation loss and gradient alignment loss; update only θ to minimize Equation 3; restore adapters to a_0
8: **end for**
9: **Compression-based perturbation** (Equations 4–5)
10: Compute middle-block BI on $\mathcal{S}$ by Equation 4
11: Select K and prune by Equation 5, keeping all adapters
12: **Return:** $(a_0, \theta_{\mathrm{emu}})$ from Equation 5

3 Gradient Alignment during Downstream Fine-Tuning

We analyze gradient alignment in the region of adapter states visited during downstream fine-tuning. Training-based perturbation targets alignment at the adapter states obtained by virtual gradient descent steps on the support set (Equation 1); compression adds error from layer removal. Appendix D provides the detailed assumptions and proofs. Write $g(a) = \nabla_a L_D(a, \theta_0)$ for the original model's adapter gradient on a downstream distribution D, and $\widetilde{g}(a) = \nabla_a L_D^K(a, \theta)$ for that of the compressed noisy model at the same adapter state.

Proposition 1 (Gradient alignment and transferred updates) *Fix the finalized noisy model before compression. Suppose its adapter gradients differ from the original model's by at most E_A at each virtual adapter state, evaluated on its associated support batch. For every downstream adapter state in the region considered, assume there is such a virtual state within distance r. On the corresponding support batch, assume the gradient difference changes by at most L_Δ times the distance between these states. Replacing that batch with downstream data changes this gradient difference by at most δ. If layer removal changes the downstream adapter gradient by at most E_C, then*

$$\|\widetilde{g}(a) - g(a)\| \le \epsilon_{\text{tot}} := E_A + L_\Delta r + \delta + E_C. \tag{6}$$

For a plug-in loss $F(a) = L_D(a, \theta_0)$ that is L-smooth on the update segment (its gradient changes by at most L times the change in adapter parameters), a transferred update with step size $\alpha > 0$ satisfies

$$F(a - \alpha\widetilde{g}) \le F(a) - \alpha\|g\|(\|g\| - \epsilon_{\text{tot}}) + \frac{L\alpha^2}{2}(\|g\| + \epsilon_{\text{tot}})^2. \tag{7}$$

Thus, if $\epsilon_{\text{tot}} < \|g\|$, a sufficiently small positive step decreases the original model's loss.

Proof idea and intuition. Matching gradient directions and magnitudes bounds E_A (Proposition D.1). The triangle inequality adds the effects of changing the adapter state ($L_\Delta r$), changing from support to downstream data (δ), and removing layers (E_C). BI controls the last term under the appendix's bounds on layer outputs and their derivatives. Smoothness then gives Equation 7: a sufficiently small emulator update decreases the original model's loss when the gradient error is smaller than $\|g\|$. For fine-tuning runs starting from the same adapters and using the same step sizes, the appendix's gradient-smoothness assumptions also bound how far their adapter parameters diverge over successive updates (Equation 19), provided both runs remain in this region.

4 EXPERIMENT

Table 1: Scores and gaps are shown in %. Comparison on eight datasets with Llama-3.2-3B. GAP achieves the highest Plug-in macro while matching or exceeding every baseline's macro Offsite ZSG and Offsite FTG. Best and second-best Plug-in values are **bold** and underlined, respectively.

Method	*Setting*	OBQA	PIQA	ARC-E	ARC-C	HellaSwag	SciQ	WebQs	RACE	Avg.
Llama-3.2-3B ZS		$31.60_{\pm 2.95}$	$77.15_{\pm 1.39}$	$75.08_{\pm 1.26}$	$42.66_{\pm 2.04}$	$58.02_{\pm 0.70}$	$94.60_{\pm 1.01}$	$12.20_{\pm 1.03}$	$39.27_{\pm 2.14}$	53.82
Pretrained FT		$47.20_{\pm 3.16}$	$80.74_{\pm 1.30}$	$79.12_{\pm 1.18}$	$48.81_{\pm 2.07}$	$61.34_{\pm 0.69}$	$95.60_{\pm 0.92}$	$45.28_{\pm 1.56}$	$46.17_{\pm 2.18}$	63.03
OT	Offsite ZS ↓	$15.60_{\pm 2.30}$	$53.43_{\pm 1.65}$	$32.07_{\pm 1.35}$	$22.18_{\pm 1.72}$	$24.82_{\pm 0.61}$	$31.40_{\pm 2.08}$	$0.00_{\pm 0.00}$	$22.61_{\pm 1.83}$	25.26
	Offsite FT	$28.80_{\pm 2.87}$	$69.86_{\pm 1.51}$	$50.51_{\pm 1.45}$	$24.23_{\pm 1.77}$	$30.67_{\pm 0.65}$	$91.20_{\pm 1.27}$	$30.51_{\pm 1.45}$	$39.27_{\pm 2.14}$	45.63
	Plug-in ↑	$\mathbf{37.60}_{\pm 3.07}$	$\underline{77.26}_{\pm 1.38}$	$\underline{73.57}_{\pm 1.28}$	$\underline{43.00}_{\pm 2.05}$	$\underline{50.49}_{\pm 0.71}$	$\underline{96.00}_{\pm 0.88}$	$\mathbf{29.43}_{\pm 1.43}$	$42.53_{\pm 2.17}$	$\underline{56.24}$
	Offsite ZSG ↑	$\mathbf{22.00}_{\pm 3.84}$	$\mathbf{23.83}_{\pm 2.15}$	$41.50_{\pm 1.86}$	$20.82_{\pm 2.68}$	$25.67_{\pm 0.94}$	$64.60_{\pm 2.26}$	$\mathbf{29.43}_{\pm 1.43}$	$19.92_{\pm 2.84}$	**30.98**
	Offsite FTG ↑	$8.80_{\pm 4.20}$	$7.40_{\pm 2.05}$	$23.06_{\pm 1.93}$	$18.77_{\pm 2.71}$	$\mathbf{19.82}_{\pm 0.96}$	$4.80_{\pm 1.55}$	$\mathbf{-1.08}_{\pm 2.04}$	$3.26_{\pm 3.05}$	10.61
CRaSh	Offsite ZS ↓	$17.20_{\pm 2.39}$	$53.21_{\pm 1.65}$	$28.45_{\pm 1.31}$	$22.53_{\pm 1.73}$	$24.40_{\pm 0.61}$	$28.40_{\pm 2.02}$	$0.00_{\pm 0.00}$	$19.73_{\pm 1.74}$	24.24
	Offsite FT	$27.20_{\pm 2.82}$	$70.73_{\pm 1.50}$	$50.08_{\pm 1.45}$	$23.55_{\pm 1.75}$	$32.38_{\pm 0.66}$	$91.60_{\pm 1.24}$	$28.94_{\pm 1.42}$	$38.31_{\pm 2.13}$	45.35
	Plug-in ↑	$\mathbf{37.60}_{\pm 3.07}$	$72.69_{\pm 1.47}$	$73.40_{\pm 1.28}$	$41.81_{\pm 2.04}$	$46.07_{\pm 0.70}$	$94.20_{\pm 1.05}$	$20.57_{\pm 1.27}$	$\underline{43.68}_{\pm 2.17}$	53.75
	Offsite ZSG ↑	$20.40_{\pm 3.89}$	$19.48_{\pm 2.21}$	$44.95_{\pm 1.83}$	$19.28_{\pm 2.67}$	$21.67_{\pm 0.93}$	$65.80_{\pm 2.28}$	$20.57_{\pm 1.27}$	$\mathbf{23.95}_{\pm 2.78}$	29.51
	Offsite FTG ↑	$10.40_{\pm 4.17}$	$1.96_{\pm 2.10}$	$23.32_{\pm 1.93}$	$18.26_{\pm 2.69}$	$13.69_{\pm 0.96}$	$2.60_{\pm 1.62}$	$-8.37_{\pm 1.91}$	$5.37_{\pm 3.04}$	8.40
GradOT	Offsite ZS ↓	$18.80_{\pm 2.48}$	$54.52_{\pm 1.64}$	$25.67_{\pm 1.27}$	$21.84_{\pm 1.71}$	$23.64_{\pm 0.60}$	$19.80_{\pm 1.78}$	$0.00_{\pm 0.00}$	$20.11_{\pm 1.76}$	23.05
	Offsite FT	$23.20_{\pm 2.68}$	$70.18_{\pm 1.51}$	$46.13_{\pm 1.45}$	$23.04_{\pm 1.74}$	$29.81_{\pm 0.65}$	$90.60_{\pm 1.31}$	$26.18_{\pm 1.38}$	$38.89_{\pm 2.14}$	43.50
	Plug-in ↑	$34.40_{\pm 3.01}$	$73.78_{\pm 1.45}$	$70.20_{\pm 1.33}$	$40.61_{\pm 2.03}$	$45.05_{\pm 0.70}$	$94.60_{\pm 1.01}$	$21.75_{\pm 1.29}$	$38.70_{\pm 2.13}$	52.39
	Offsite ZSG ↑	$15.60_{\pm 3.90}$	$19.26_{\pm 2.19}$	$44.53_{\pm 1.84}$	$18.77_{\pm 2.65}$	$21.41_{\pm 0.92}$	$\mathbf{74.80}_{\pm 2.05}$	$21.75_{\pm 1.29}$	$18.59_{\pm 2.76}$	29.34
	Offsite FTG ↑	$\mathbf{11.20}_{\pm 4.03}$	$3.60_{\pm 2.09}$	$24.07_{\pm 1.97}$	$17.57_{\pm 2.67}$	$15.24_{\pm 0.96}$	$4.00_{\pm 1.65}$	$-4.43_{\pm 1.89}$	$-0.19_{\pm 3.02}$	8.89
GAP	Offsite ZS ↓	$15.20_{\pm 2.28}$	$58.98_{\pm 1.62}$	$32.49_{\pm 1.36}$	$22.87_{\pm 1.74}$	$29.34_{\pm 0.64}$	$48.20_{\pm 2.24}$	$0.00_{\pm 0.00}$	$23.75_{\pm 1.86}$	28.85
	Offsite FT	$26.00_{\pm 2.78}$	$71.93_{\pm 1.48}$	$54.21_{\pm 1.45}$	$26.28_{\pm 1.82}$	$39.89_{\pm 0.69}$	$91.20_{\pm 1.27}$	$31.30_{\pm 1.46}$	$38.70_{\pm 2.13}$	47.44
	Plug-in ↑	$\underline{37.20}_{\pm 3.06}$	$\mathbf{80.20}_{\pm 1.32}$	$\mathbf{79.63}_{\pm 1.17}$	$\mathbf{50.51}_{\pm 2.07}$	$\mathbf{59.11}_{\pm 0.69}$	$\mathbf{97.00}_{\pm 0.76}$	$\underline{28.44}_{\pm 1.42}$	$\mathbf{46.55}_{\pm 2.19}$	**59.83**
	Offsite ZSG ↑	$\mathbf{22.00}_{\pm 3.82}$	$21.22_{\pm 2.09}$	$\mathbf{47.14}_{\pm 1.79}$	$\mathbf{27.64}_{\pm 2.70}$	$\mathbf{29.77}_{\pm 0.94}$	$48.80_{\pm 2.37}$	$28.44_{\pm 1.42}$	$22.80_{\pm 2.87}$	**30.98**
	Offsite FTG ↑	$\mathbf{11.20}_{\pm 4.13}$	$\mathbf{8.27}_{\pm 1.98}$	$\mathbf{25.42}_{\pm 1.86}$	$\mathbf{24.23}_{\pm 2.76}$	$19.22_{\pm 0.98}$	$\mathbf{5.80}_{\pm 1.48}$	$-2.86_{\pm 2.04}$	$\mathbf{7.85}_{\pm 3.05}$	**12.39**

4.1 SETUP

Datasets. Following CRaSh (Zhang et al., 2023), we evaluate eight datasets across three categories: (i) *Multi-Choice QA*: OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), and RACE (Lai et al., 2017); (ii) *Closed-Book QA*: ARC-Easy/Challenge (Clark et al.,

Table 2: Scores and gaps are shown in %. Comparison on eight datasets with Llama-3.1-8B. GAP achieves the highest macro Plug-in, Offsite ZSG, and Offsite FTG, improving the gap-utility trade-off. Best and second-best Plug-in values are **bold** and underlined, respectively.

Method	*Setting*	OBQA	PIQA	ARC-E	ARC-C	HellaSwag	SciQ	WebQs	RACE	Avg.
Llama-3.1-8B ZS		$34.40_{\pm 3.01}$	$79.65_{\pm 1.33}$	$82.49_{\pm 1.10}$	$49.15_{\pm 2.07}$	$64.03_{\pm 0.68}$	$96.40_{\pm 0.83}$	$11.52_{\pm 1.00}$	$39.85_{\pm 2.14}$	57.19
Pretrained FT		$46.40_{\pm 3.16}$	$80.41_{\pm 1.31}$	$82.15_{\pm 1.11}$	$51.54_{\pm 2.07}$	$63.73_{\pm 0.68}$	$95.80_{\pm 0.90}$	$45.18_{\pm 1.56}$	$49.23_{\pm 2.19}$	64.31
OT	Offsite ZS ↓	$10.00_{\pm 1.90}$	$52.34_{\pm 1.65}$	$26.52_{\pm 1.28}$	$19.62_{\pm 1.64}$	$28.22_{\pm 0.64}$	$25.40_{\pm 1.95}$	$0.00_{\pm 0.00}$	$24.52_{\pm 1.88}$	23.33
	Offsite FT	$30.40_{\pm 2.92}$	$72.03_{\pm 1.48}$	$55.98_{\pm 1.44}$	$24.74_{\pm 1.78}$	$36.13_{\pm 0.68}$	$90.40_{\pm 1.32}$	$31.20_{\pm 1.45}$	$36.59_{\pm 2.11}$	47.18
	Plug-in ↑	$\underline{42.80}_{\pm 3.14}$	$\underline{80.63}_{\pm 1.30}$	$\underline{79.55}_{\pm 1.17}$	$\underline{48.98}_{\pm 2.07}$	$\underline{55.75}_{\pm 0.70}$	$\mathbf{97.00}_{\pm 0.76}$	$26.77_{\pm 1.39}$	$\underline{46.93}_{\pm 2.19}$	$\underline{59.80}$
	Offsite ZSG ↑	$\mathbf{32.80}_{\pm 3.67}$	$\mathbf{28.29}_{\pm 2.10}$	$53.03_{\pm 1.73}$	$29.36_{\pm 2.64}$	$27.53_{\pm 0.95}$	$71.60_{\pm 2.09}$	$26.77_{\pm 1.39}$	$22.41_{\pm 2.89}$	36.47
	Offsite FTG ↑	$12.40_{\pm 4.29}$	$8.60_{\pm 1.97}$	$23.57_{\pm 1.86}$	$\mathbf{24.24}_{\pm 2.73}$	$\mathbf{19.62}_{\pm 0.98}$	$\mathbf{6.60}_{\pm 1.52}$	$-4.43_{\pm 2.01}$	$\mathbf{10.34}_{\pm 3.04}$	12.62
CRaSh	Offsite ZS ↓	$15.60_{\pm 2.30}$	$52.99_{\pm 1.65}$	$26.60_{\pm 1.28}$	$22.53_{\pm 1.73}$	$29.36_{\pm 0.64}$	$26.00_{\pm 1.96}$	$0.00_{\pm 0.00}$	$21.26_{\pm 1.79}$	24.29
	Offsite FT	$30.00_{\pm 2.90}$	$65.51_{\pm 1.57}$	$57.15_{\pm 1.44}$	$26.45_{\pm 1.82}$	$38.94_{\pm 0.69}$	$92.40_{\pm 1.19}$	$26.87_{\pm 1.39}$	$38.89_{\pm 2.14}$	47.03
	Plug-in ↑	$37.60_{\pm 3.07}$	$79.00_{\pm 1.34}$	$74.24_{\pm 1.27}$	$42.66_{\pm 2.04}$	$54.91_{\pm 0.70}$	$95.40_{\pm 0.94}$	$\underline{29.43}_{\pm 1.43}$	$44.64_{\pm 2.18}$	57.24
	Offsite ZSG ↑	$22.00_{\pm 3.84}$	$26.01_{\pm 2.13}$	$47.64_{\pm 1.80}$	$20.13_{\pm 2.67}$	$25.55_{\pm 0.95}$	$69.40_{\pm 2.17}$	$29.43_{\pm 1.43}$	$23.38_{\pm 2.82}$	32.95
	Offsite FTG ↑	$7.60_{\pm 4.22}$	$\mathbf{13.49}_{\pm 2.06}$	$17.09_{\pm 1.92}$	$16.21_{\pm 2.73}$	$15.97_{\pm 0.98}$	$3.00_{\pm 1.52}$	$2.56_{\pm 1.99}$	$5.75_{\pm 3.05}$	10.21
GradOT	Offsite ZS ↓	$13.20_{\pm 2.15}$	$51.69_{\pm 1.65}$	$24.07_{\pm 1.24}$	$22.70_{\pm 1.73}$	$27.54_{\pm 0.63}$	$18.40_{\pm 1.73}$	$0.00_{\pm 0.00}$	$21.46_{\pm 1.80}$	22.38
	Offsite FT	$21.60_{\pm 2.61}$	$72.03_{\pm 1.48}$	$49.07_{\pm 1.45}$	$20.65_{\pm 1.67}$	$34.22_{\pm 0.67}$	$90.20_{\pm 1.33}$	$25.10_{\pm 1.36}$	$40.42_{\pm 2.15}$	44.16
	Plug-in ↑	$25.20_{\pm 2.75}$	$77.26_{\pm 1.38}$	$74.41_{\pm 1.27}$	$24.57_{\pm 1.78}$	$48.93_{\pm 0.71}$	$\underline{96.20}_{\pm 0.86}$	$10.73_{\pm 0.97}$	$39.46_{\pm 2.14}$	49.60
	Offsite ZSG ↑	$12.00_{\pm 3.49}$	$25.57_{\pm 2.15}$	$50.34_{\pm 1.77}$	$1.87_{\pm 2.48}$	$21.39_{\pm 0.95}$	$\mathbf{77.80}_{\pm 1.93}$	$10.73_{\pm 0.97}$	$18.00_{\pm 2.80}$	27.22
	Offsite FTG ↑	$3.60_{\pm 3.79}$	$5.23_{\pm 2.02}$	$25.34_{\pm 1.93}$	$3.92_{\pm 2.44}$	$14.71_{\pm 0.98}$	$6.00_{\pm 1.58}$	$-14.37_{\pm 1.67}$	$-0.96_{\pm 3.03}$	5.44
GAP	Offsite ZS ↓	$17.60_{\pm 2.41}$	$57.02_{\pm 1.63}$	$24.24_{\pm 1.24}$	$21.67_{\pm 1.70}$	$26.37_{\pm 0.62}$	$20.60_{\pm 1.81}$	$0.00_{\pm 0.00}$	$21.46_{\pm 1.80}$	23.62
	Offsite FT	$30.40_{\pm 2.92}$	$74.32_{\pm 1.44}$	$55.72_{\pm 1.44}$	$29.01_{\pm 1.88}$	$46.58_{\pm 0.70}$	$93.00_{\pm 1.14}$	$30.12_{\pm 1.44}$	$43.10_{\pm 2.17}$	50.28
	Plug-in ↑	$\mathbf{46.40}_{\pm 3.16}$	$\mathbf{83.24}_{\pm 1.23}$	$\mathbf{84.60}_{\pm 1.05}$	$\mathbf{53.07}_{\pm 2.06}$	$\mathbf{64.43}_{\pm 0.68}$	$\mathbf{97.00}_{\pm 0.76}$	$\mathbf{37.01}_{\pm 1.52}$	$\mathbf{48.85}_{\pm 2.19}$	**64.33**
	Offsite ZSG ↑	$28.80_{\pm 3.97}$	$26.22_{\pm 2.04}$	$\mathbf{60.36}_{\pm 1.62}$	$\mathbf{31.40}_{\pm 2.67}$	$\mathbf{38.06}_{\pm 0.92}$	$76.40_{\pm 1.96}$	$\mathbf{37.01}_{\pm 1.52}$	$\mathbf{27.39}_{\pm 2.83}$	**40.71**
	Offsite FTG ↑	$\mathbf{16.00}_{\pm 4.30}$	$8.92_{\pm 1.89}$	$\mathbf{28.88}_{\pm 1.78}$	$24.06_{\pm 2.79}$	$17.85_{\pm 0.98}$	$4.00_{\pm 1.37}$	$\mathbf{6.89}_{\pm 2.09}$	$5.75_{\pm 3.08}$	**14.05**

2018) and WebQuestions (Berant et al., 2013); and (iii) *Sentence Completion*: HellaSwag (Zellers et al., 2019). We use `lm-evaluation-harness`[1] (Gao et al., 2024), with matching training and evaluation prompt templates. Following GradOT (Yao et al., 2025a), emulator construction uses 1,500 support examples drawn equally from BoolQ (Clark et al., 2019), TriviaQA (Joshi et al., 2017), and CoPA (Wang et al., 2019) (BCT), without downstream training, validation, or test data. Raw downstream data remain with the data owner. Dataset details appear in Appendix A.1.

Models. We evaluate Llama-3.2-3B, Llama-3.1-8B, Qwen2.5-3B, and Qwen2.5-7B (Yang et al., 2024), with supplementary experiments on Qwen2.5-32B and the MoE model DeepSeek-V2-Lite (DeepSeek-AI, 2024). All original references are Base checkpoints, not instruction-tuned variants. The first two and last two layers form the trainable adapters, and the remaining middle layers are used to construct the emulator.

Methods. We compare against OT (Xiao et al., 2023), CRaSh (Zhang et al., 2023), and GradOT (Yao et al., 2025a). Unless specified otherwise, baselines use an Emulator Ratio of 50%, denoting retained emulator capacity relative to the original middle layers. OT uniformly drops half the middle layers, without post-compression distillation due to computational limits. CRaSh forms half as many clusters as middle layers and shares each cluster center's parameters among its members. GradOT uses compression rates of 0.5 for both MHA and MLP modules. The same BCT support set is used for CRaSh clustering, GradOT compression, and GAP training and BI calibration.

GAP uses $\lambda = 25$, learning rate $5\mathrm{e}{-5}$, and one training epoch on support set, followed by BI pruning without further optimization. The default Llama-3.2-3B and Llama-3.1-8B emulators retain 16/28 and 18/32 total layers, respectively. For each reported depth, one emulator is shared across tasks.

Implementation Details. We use AdamW (Loshchilov & Hutter, 2019) with cosine learning-rate decay and linear warm-up. The standard downstream recipe selects the best Offsite FT validation checkpoint from 25 learning-rate/epoch candidates, while auxiliary budgets can differ. Validation uses the first 50% of the labeled evaluation split without shuffling, and final reporting uses the latter 50%. All compression curves are computed on the validation half. Checkpoint selection does not use Plug-in or test scores; exploratory depth adjustments and prior test exposure are disclosed with the corresponding results. Pretrained FT directly tunes the same boundary adapter layers of the

[1] `https://github.com/EleutherAI/lm-evaluation-harness`

full Base model, without noise or pruning, using fixed learning rate $5e{-}5$ and five epochs. This full-model-access reference is reused across tables for each model/task. Search grids and uncertainty definitions appear in Appendix A.2 and Appendix A.1.

4.2 GAP-UTILITY TRADE-OFF ANALYSIS

Figure 1b compares nominal emulator ratios from 0.1 to 0.9 on Llama-3.2-3B, averaged over OpenBookQA, ARC-Easy, ARC-Challenge, and WebQuestions. Each setting uses the standard Offsite-validation checkpoint selection and reports Plug-in and both gaps.

At ratio 0.5, GAP achieves 48.75% Plug-in, 32.29% Offsite ZSG, and 13.55% Offsite FTG, exceeding all three baselines on each metric at the same ratio. At ratio 0.8, GAP also exceeds OT's highest-utility evaluated point (ratio 0.9) in Plug-in and both gaps. These comparisons show an improved gap-utility trade-off at the default and higher-utility operating points.

4.3 MAIN RESULTS

Tables 1 and 2 report the eight-task Llama results. GAP achieves the highest Plug-in macro average on both backbones while matching or exceeding every baseline's macro Offsite ZSG and Offsite FTG. Relative to OT, the strongest baseline, Plug-in improves by 3.59% and 4.53% on Llama-3.2-3B and Llama-3.1-8B, respectively. The former gain comes at the same Offsite ZSG and a larger Offsite FTG; the latter improves both gaps. GAP also exceeds Pretrained ZS on both backbones, supporting useful transferred adaptation rather than merely weakening the offsite model.

Additional results on Qwen2.5 models appear in Appendix B.1, with larger-scale and MoE results in Appendix B.2, demonstrating generalization beyond the main Llama backbones. On the medical ATC-Easy subset of MedConceptsQA (Shoham & Rappoport, 2024), GAP achieves Plug-in 87.60% using Qwen2.5-7B, with positive Offsite ZSG and FTG despite using only general-domain owner support set (Appendix B.3). Appendix B.4 reports runtime and memory across four scales of Qwen2.5 models and the allocated GPUs. On Llama-3.2-3B, GAP is $3.47\times$ faster than GradOT in the measured construction configuration.

4.4 ABLATION STUDY

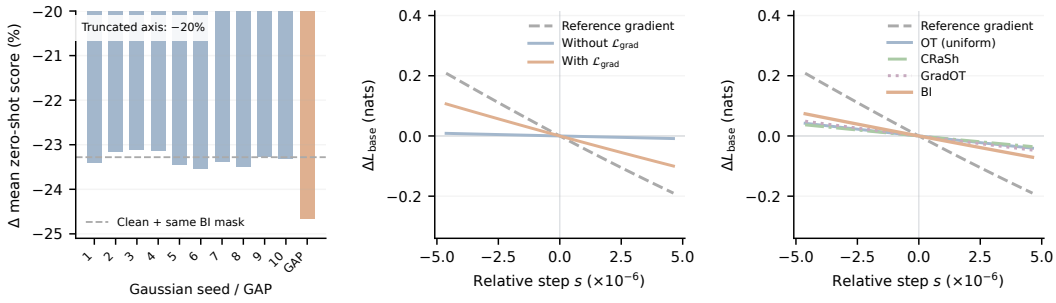
We examine learned versus random perturbations, construction components, and local update transfer. Hyper-parameter and support-set ablations appear in Appendix C.

Adversarial noise vs. norm-aligned random noise. We compare the learned perturbation of the Llama-3.2-3B model in Table 1 against ten Gaussian controls, matching the perturbation norm separately for each middle-layer tensor. All controls retain the same 12 middle layers and four adapters, using a BI mask frozen from owner-support data before downstream evaluation. A clean-pruned control uses the original weights with this mask to separate compression from perturbation effects. We evaluate equal-weight eight-task macro ZS. Algorithm 2 specifies the norm-matching procedure, with seeds and precision details in Appendix C. Figure 3a shows that the learned perturbation has a perturbation effect beyond norm-matched random noise under the same compression mask.

Construction components improve the gap-utility trade-off. Table 3 compares Llama-3.2-3B variants at the same 16-layer depth: removing the adversarial perturbation loss, removing the gradient alignment loss, replacing BI with uniform selection, or using single-step alignment ($k \in \{1\}$) instead of cycling through $k \in \{1, 2, 3\}$. BI is recomputed using only BCT support. Construction details are in Appendix A.2.

The full method achieves the highest Plug-in (59.83%) and Offsite FTG (12.39%) among these variants. Both gradient alignment and BI improve Plug-in over their respective ablations, while cycling the number of virtual gradient descent steps improves Plug-in and both gaps over single-step alignment. Together, the comparisons support combining learned perturbation, gradient alignment, and BI for useful adapter transfer.

BI pruning and gradient alignment loss improve loss landscape alignment. Figure 3(b,c) measures transferability through the change ΔL in the original Base model's loss, with zero denoting



(a) Adversarial noise. (b) $\mathcal{L}_{\text{grad}}$. (c) BI.

Figure 3: Component ablations. (a) Eight-task macro ZS differences from the full pretrained model; all controls share the same 16-layer BI mask, and the dashed line is the clean-pruned control. The vertical axis is truncated at -20%; the full-model reference at zero is outside the displayed range. (b,c) Local Base-loss slices along normalized adapter-gradient directions, with the same measured step grid and vertical scale; both positive and negative steps are shown. Zero denotes the unchanged Base loss. OT denotes the matched-depth uniform-pruning control of the same perturbed model. Panel (c) uses the corrected BI values and the reported pointwise GradOT/CRaSh ratios.

Table 3: Matched-capacity component ablations on Llama-3.2-3B. All variants retain 16 of 28 total layers. Scores and gaps are based on unweighted eight-task macro averages (%).

Variant	Offsite ZSG ↑	Offsite FTG ↑	Plug-in ↑
GAP	30.98	**12.39**	**59.83**
w/o Adversarial perturbation loss	29.24	11.54	58.74
w/o Gradient alignment loss	**34.47**	11.54	58.96
Gradient alignment loss $k \in \{1\}$	30.23	11.71	59.10
BI replaced by uniform	32.05	11.88	57.04

its loss before the update. For an adapter gradient g at starting state a, the diagnostic is $\Delta L(s) = L_{\text{Base}}(a - s\|a\|_2 g/\|g\|_2) - L_{\text{Base}}(a)$, evaluated on the same support examples. Because the gradient directions are normalized and share the same relative step s, a larger loss decrease for positive s indicates a more effective direction for updating the Base model. Figure 3b shows that adding $\mathcal{L}_{\text{grad}}$ produces a steeper loss decrease than omitting it, indicating better local loss-slice alignment and update transfer. Figure 3c shows a larger loss decrease for BI than for OT, CRaSh, and GradOT at the displayed positive steps, consistent with more effective local update transfer.

5 Related Work

5.1 Offsite Tuning

Offsite-tuning (Xiao et al., 2023) enables adaptation while keeping raw downstream data with the data owner and the complete original model with the model owner. The model owner supplies a weakened emulator and adapters; the data owner tunes only the adapters locally and returns them for plug-in into the complete model.

Existing methods mainly construct emulators through compression-based aligned perturbation. OT (Xiao et al., 2023) uniformly drops middle layers and optionally applies hidden-state distillation. CRaSh (Zhang et al., 2023) clusters layers by representation similarity and shares cluster-center layers to preserve functionality. ScaleOT (Yao et al., 2025b) uses reinforcement learning to identify less important layers, replaces them with lightweight harmonizers, and selectively compresses attention modules. GradOT (Yao et al., 2025a) incorporates second-order information through a gradient-preserving compression score for rank decomposition and channel pruning.

These approaches align representations or gradients within prescribed compression operations. GAP extends this construction space with training-based gradient-aligned perturbation, directly learning adversarial middle-layer changes while matching adapter gradients. A complementary compression-based stage uses BI to select low-disruption layer deletions for gradient alignment. Together, the two stages target the gap-utility trade-off without transferring raw downstream data or the complete original model.

5.2 Machine Learning without Centralizing Raw Data and Raw Models

Cross-owner learning can be organized around what each participant exchanges rather than centralizing its original assets. Here, raw models refer to the complete original models, as distinct from the weakened emulators shared in offsite-tuning. Related approaches address different parts of this exchange constraint.

Federated Learning. Federated learning (McMahan et al., 2017) leaves training examples on participating devices and aggregates locally computed model updates. FedProx (Li et al., 2020b) and SCAFFOLD (Karimireddy et al., 2020) address heterogeneous client data and update drift. For language models, FedRW (Ye et al., 2025) uses data reweighting to improve federated fine-tuning under non-IID data. This setting avoids transferring raw training examples but typically distributes a shared model for local optimization. Offsite-tuning additionally keeps the complete original model with its owner and distributes a weakened emulator instead. EncryptedLLM (de Castro et al., 2025) addresses a related inference setting by computing on encrypted queries, avoiding disclosure of the raw input to the inference service.

Machine Unlearning. Machine unlearning removes the influence of designated examples from a trained model while retaining behavior on other data. Gradient-ascent methods increase the loss on a forget set: Jang et al. (2023) apply this operation to target token sequences, while Yao et al. (2024) combine output-token losses with mismatch objectives and retention regularization. Wang et al. (2025) use weighted gradient ascent to moderate excessive forgetting. These methods change what a model retains. Offsite-tuning instead changes what is exchanged during adaptation, keeping raw downstream examples and the complete original model within their owners. GAP uses adversarial loss to weaken a shareable emulator, coupled with temporal gradient alignment to preserve the usefulness of returned adapters, rather than to erase a designated dataset from the original model.

6 Conclusion

We present GAP, a dual-perturbation approach to offsite-tuning. Training-based gradient-aligned perturbation learns adversarial weight changes while matching adapter gradients between the offsite model and the complete model at temporary tuning states. Compression-based gradient-aligned perturbation uses support-set-only BI selection to further compress the emulator. Their combination expands the emulator construction choices beyond prescribed compression operations in existing methods and targets the gap-utility trade-off. Our theoretical analysis bounds gradient mismatch in GAP over downstream fine-tuning regions and connects it to useful transferred updates. Experiments demonstrate higher Plug-in macro performance on eight benchmarks, with positive gains over both Pretrained ZS, Offsite ZS and Offsite FT, and generalization to additional dense and MoE backbones. Throughout the exchange, raw downstream data remain with the data owner and the complete original model remains with the model owner.

Limitations

GAP can enable fine-tuning when the raw data remains on the data owner's side and the complete model remains on the model owner's side. However, it does not guarantee that the raw data cannot be extracted through model inversion or other methods, as is the case in privacy-preserving research. GAP is a fine-tuning approach designed for practical regulatory constraints on the data owner side and the model owner side.

AI USE STATEMENT

In this work, we used generative AI tools for assist in the writing of proofs, implement methods. We have not used generative AI tools for theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, interpret results. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (eds.), *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1160/.

Yonatan Bisk, Rowan Zellers, Ronan Le bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7432–7439, Apr. 2020. doi: 10.1609/aaai.v34i05.6239. URL https://ojs.aaai.org/index.php/AAAI/article/view/6239.

Terence Jie Chua, Wenhan Yu, Jun Zhao, and Kwok-Yan Lam. Fedpeat: Convergence of federated learning, parameter-efficient fine tuning, and emulator assisted tuning for artificial intelligence foundation models with mobile edge computing. *arXiv preprint arXiv:2310.17491*, 2023.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *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, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Leo de Castro, Daniel Escudero, Adya Agrawal, Antigoni Polychroniadou, and Manuela Veloso. EncryptedLLM: Privacy-preserving large language model inference via GPU-accelerated fully homomorphic encryption. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=PGNff6H1TV.

DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 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. 4171–4186, 2019.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting*

of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 14389–14408, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.805. URL https://aclanthology.org/2023.acl-long.805/.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel (eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1082. URL https://aclanthology.org/D17-1082/.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. Bert-attack: Adversarial attack against bert using bert. *arXiv preprint arXiv:2004.09984*, 2020a.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*, 2024. URL https://arxiv.org/abs/2403.03853.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260/.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Ofir Ben Shoham and Nadav Rappoport. Medconceptsqa: Open source medical concepts qa benchmark. *Computers in Biology and Medicine*, 182:109089, 2024. ISSN 0010-4825. doi: https://doi.org/10.1016/j.compbiomed.2024.109089. URL https://www.sciencedirect.com/science/article/pii/S0010482524011740.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/4496bf24afe7fab6f046bf4923da8de6-Paper.pdf.

Qizhou Wang, Jin Peng Zhou, Zhanke Zhou, Saebyeol Shin, Bo Han, and Kilian Q Weinberger. Rethinking LLM unlearning objectives: A gradient perspective and go beyond. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=huo8MqVH6t.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In Leon Derczynski, Wei Xu, Alan Ritter, and Tim Baldwin (eds.), *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413/.

Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning. *Advances in Neural Information Processing Systems*, 36:36637–36651, 2023.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In Qun Liu and David Schlangen (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Guangxuan Xiao, Ji Lin, and Song Han. Offsite-Tuning: Transfer learning without full model, 2023. URL https://arxiv.org/abs/2302.04870.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Kai Yao, Zhaorui Tan, Penglei Gao, Lichun Li, Kaixin Wu, Yinggui Wang, Yuan Zhao, Yixin Ji, Jianke Zhu, and Wei Wang. GradOT: Training-free gradient-preserving offsite-tuning for large language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5115–5130, Vienna, Austria, July 2025a. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.255. URL https://aclanthology.org/2025.acl-long.255/.

Kai Yao, Zhaorui Tan, Tiandi Ye, Lichun Li, Yuan Zhao, Wenyan Liu, Wei Wang, and Jianke Zhu. ScaleOT: Privacy-utility-scalable offsite-tuning with dynamic layerreplace and selective rank compression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(21):22074–22082, Apr. 2025b. doi: 10.1609/aaai.v39i21.34360. URL https://ojs.aaai.org/index.php/AAAI/article/view/34360.

Yuanshun Yao, Xiaojun Xu, and YangLiu. Large language model unlearning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 105425–105475. Curran Associates, Inc., 2024. doi: 10.52202/079017-3346. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/be52acf6bccf4a8c0a90fe2f5cfcead3-Paper-Conference.pdf.

Pukang Ye, Junwei Luo, Xiaolei Dong, and Yunbo Yang. Fedrw: Efficient privacy-preserving data reweighting for enhancing federated learning of language models. *arXiv preprint arXiv:2511.07505*, 2025.

Tiandi Ye, Cen Chen, Yinggui Wang, Xiang Li, and Ming Gao. Bapfl: You can backdoor personalized federated learning. *ACM Transactions on Knowledge Discovery from Data*, 18(7):1–17, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472/.

Kaiyan Zhang, Ning Ding, Biqing Qi, Xuekai Zhu, Xinwei Long, and Bowen Zhou. CRaSh: Clustering, removing, and sharing enhance fine-tuning without full large language model. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9612–9637, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.597. URL https://aclanthology.org/2023.emnlp-main.597/.

A More Experimental Details

A.1 Dataset Splits Details

Some datasets used in our experiments either lack an official test set or provide test sets without publicly available labels. By default, `lm-eval-harness` evaluates on the labeled test split when it is available. When such a test split is unavailable, the framework instead performs evaluation on the labeled validation split. However, hyper-parameter search requires each dataset to have both a labeled validation set and a labeled test set. Therefore, we split the evaluation split used by `lm-eval-harness` for each dataset into two equal halves without shuffling.

The first 50% of the samples are used as the validation set for hyperparameter search, while the remaining 50% serve as the test set for final reporting. As a result, the validation and test sets used in our experiments are custom splits rather than the official standard splits. Per-dataset uncertainties are finite-sample standard errors (SEs), not training-seed standard deviations; gap SEs are approximate and do not establish significance. Macro scores are unweighted averages. Gap means subtract the displayed paired scores, so rounding can create small differences from averaging rounded task gaps. Tables 3 and 11, as well as the Avg. gap entries, report means only. Table 4 presents the detailed statistics for these datasets.

Table 4: Number of samples in the training, validation, and test sets of eight datasets used in our experiments.

Dataset	Train	Validation	Test
OpenBookQA	4,957	250	250
PIQA	16,113	919	919
SciQ	11,679	500	500
RACE	62,445	523	522
ARC-E	2,251	1,188	1,188
ARC-C	1,119	586	586
WebQs	3,589	1,016	1,016
HellaSwag	39,905	5,021	5,021

A.2 Implementation Details

We conduct experiments using `DeepSpeed`[2] and `HuggingFace Transformers`[3] (Wolf et al., 2020) library. Throughout all experiments, the first two and last two Transformer layers of each model are designated as the adapter. The remaining middle layers are used to construct the emulator. The baseline configurations are specified in Section 4; in particular, our OT baseline omits post-compression knowledge distillation and is not a reproduction of every step of the original OT training recipe.

Downstream selection. The standard 25-candidate grid combines learning rates in $\{5e{-}6, 8e{-}6, 1e{-}5, 2e{-}5, 5e{-}5\}$ with epoch counts in $\{1, 2, 3, 4, 5\}$. Selection maximizes the offsite model's own validation score, not Plug-in or test performance. Frozen-recipe depth probes and auxiliary experiments can use different budgets, as disclosed with their results. Bold gap values denote the best displayed score within each comparison, including ties.

Component controls. The established Llama-3.2-3B component owners use 188 first-stage steps and 48 BCT support examples for BI calibration, without post-pruning training. For the zero-gradient-weight control, the virtual-update and CE microbatch computation are retained, while the gradient-alignment contribution to the owner objective is exactly zero. Loss-ablation masks are recomputed using support data only; downstream examples and scores never select individual layers.

[2] `https://github.com/deepspeedai/DeepSpeed`

[3] `https://github.com/huggingface/transformers`

Table 5: Scores and gaps are shown in %. Comparison results on eight datasets with Qwen2.5-3B. The best results are shown in **bold**, and the second-best results are underlined.

Method	*Setting*	OBQA	PIQA	ARC-E	ARC-C	HellaSwag	SciQ	WebQs	RACE	Avg.
Qwen2.5-3B ZS		$30.80_{\pm 2.93}$	$79.43_{\pm 1.33}$	$77.95_{\pm 1.20}$	$44.71_{\pm 2.06}$	$57.52_{\pm 0.70}$	$95.40_{\pm 0.94}$	$10.73_{\pm 0.97}$	$39.27_{\pm 2.14}$	54.48
Pretrained FT		$37.60_{\pm 3.07}$	$78.35_{\pm 1.36}$	$80.47_{\pm 1.15}$	$49.32_{\pm 2.07}$	$61.82_{\pm 0.69}$	$97.00_{\pm 0.76}$	$42.13_{\pm 1.55}$	$38.51_{\pm 2.13}$	60.65
OT	Offsite ZS ↓	$16.40_{\pm 2.35}$	$56.91_{\pm 1.63}$	$28.87_{\pm 1.32}$	$19.28_{\pm 1.63}$	$25.05_{\pm 0.61}$	$47.80_{\pm 2.24}$	$0.00_{\pm 0.00}$	$21.46_{\pm 1.80}$	26.97
	Offsite FT	$23.60_{\pm 2.69}$	$68.77_{\pm 1.53}$	$54.46_{\pm 1.45}$	$22.01_{\pm 1.71}$	$31.57_{\pm 0.66}$	$90.00_{\pm 1.34}$	$23.33_{\pm 1.33}$	$36.02_{\pm 2.10}$	43.72
	Plug-in ↑	$31.60_{\pm 2.95}$	$\underline{76.82}_{\pm 1.39}$	$\underline{77.10}_{\pm 1.22}$	$\underline{44.37}_{\pm 2.05}$	$\underline{51.34}_{\pm 0.71}$	$\mathbf{96.20}_{\pm 0.86}$	$\underline{20.28}_{\pm 1.26}$	$\underline{38.89}_{\pm 2.14}$	$\underline{54.58}$
	Offsite ZSG ↑	$15.20_{\pm 3.77}$	$19.91_{\pm 2.14}$	$48.23_{\pm 1.80}$	$\mathbf{25.09}_{\pm 2.62}$	$\mathbf{26.29}_{\pm 0.94}$	$48.40_{\pm 2.40}$	$20.28_{\pm 1.26}$	$\mathbf{17.43}_{\pm 2.80}$	**27.61**
	Offsite FTG ↑	$8.00_{\pm 3.99}$	$\mathbf{8.05}_{\pm 2.07}$	$22.64_{\pm 1.89}$	$\mathbf{22.36}_{\pm 2.67}$	$19.77_{\pm 0.97}$	$6.20_{\pm 1.59}$	$-3.05_{\pm 1.83}$	$\mathbf{2.87}_{\pm 3.00}$	**10.86**
CRaSh	Offsite ZS ↓	$17.60_{\pm 2.41}$	$51.58_{\pm 1.65}$	$26.18_{\pm 1.28}$	$22.70_{\pm 1.73}$	$23.56_{\pm 0.60}$	$20.60_{\pm 1.81}$	$0.00_{\pm 0.00}$	$22.03_{\pm 1.82}$	23.03
	Offsite FT	$25.20_{\pm 2.75}$	$70.73_{\pm 1.50}$	$47.64_{\pm 1.45}$	$22.87_{\pm 1.74}$	$35.71_{\pm 0.68}$	$90.40_{\pm 1.32}$	$14.37_{\pm 1.10}$	$36.21_{\pm 2.11}$	42.89
	Plug-in ↑	$21.60_{\pm 2.61}$	$73.56_{\pm 1.46}$	$64.23_{\pm 1.39}$	$\mathbf{44.54}_{\pm 2.05}$	$45.05_{\pm 0.70}$	$89.20_{\pm 1.39}$	$2.56_{\pm 0.50}$	$36.97_{\pm 2.11}$	47.21
	Offsite ZSG ↑	$4.00_{\pm 3.55}$	$21.98_{\pm 2.20}$	$38.05_{\pm 1.89}$	$21.84_{\pm 2.68}$	$21.49_{\pm 0.92}$	$\mathbf{68.60}_{\pm 2.28}$	$2.56_{\pm 0.50}$	$14.94_{\pm 2.79}$	24.18
	Offsite FTG ↑	$-3.60_{\pm 3.79}$	$2.83_{\pm 2.09}$	$16.59_{\pm 2.01}$	$21.67_{\pm 2.69}$	$9.34_{\pm 0.98}$	$-1.20_{\pm 1.92}$	$-11.81_{\pm 1.21}$	$0.76_{\pm 2.98}$	4.32
GradOT	Offsite ZS ↓	$16.40_{\pm 2.35}$	$53.43_{\pm 1.65}$	$27.19_{\pm 1.29}$	$21.67_{\pm 1.70}$	$24.02_{\pm 0.60}$	$29.40_{\pm 2.04}$	$0.00_{\pm 0.00}$	$24.14_{\pm 1.87}$	24.53
	Offsite FT	$20.80_{\pm 2.57}$	$69.42_{\pm 1.52}$	$44.44_{\pm 1.44}$	$19.80_{\pm 1.65}$	$29.72_{\pm 0.65}$	$86.60_{\pm 1.52}$	$18.70_{\pm 1.22}$	$36.97_{\pm 2.11}$	40.81
	Plug-in ↑	$\underline{32.80}_{\pm 2.98}$	$76.50_{\pm 1.40}$	$\mathbf{77.44}_{\pm 1.21}$	$24.91_{\pm 1.79}$	$50.13_{\pm 0.71}$	$93.60_{\pm 1.10}$	$8.66_{\pm 0.88}$	$36.97_{\pm 2.11}$	50.13
	Offsite ZSG ↑	$\mathbf{16.40}_{\pm 3.80}$	$\mathbf{23.07}_{\pm 2.16}$	$\mathbf{50.25}_{\pm 1.77}$	$3.24_{\pm 2.47}$	$26.11_{\pm 0.93}$	$64.20_{\pm 2.32}$	$8.66_{\pm 0.88}$	$12.83_{\pm 2.82}$	25.60
	Offsite FTG ↑	$\mathbf{12.00}_{\pm 3.94}$	$7.08_{\pm 2.07}$	$\mathbf{33.00}_{\pm 1.88}$	$5.11_{\pm 2.43}$	$\mathbf{20.41}_{\pm 0.96}$	$\mathbf{7.00}_{\pm 1.88}$	$-10.04_{\pm 1.50}$	$0.00_{\pm 2.98}$	9.32
GAP	Offsite ZS ↓	$23.20_{\pm 2.68}$	$67.68_{\pm 1.54}$	$61.36_{\pm 1.41}$	$29.52_{\pm 1.89}$	$32.76_{\pm 0.66}$	$79.80_{\pm 1.80}$	$1.08_{\pm 0.32}$	$29.89_{\pm 2.01}$	40.66
	Offsite FT	$36.00_{\pm 3.04}$	$76.71_{\pm 1.39}$	$71.97_{\pm 1.30}$	$38.23_{\pm 2.01}$	$45.63_{\pm 0.70}$	$94.60_{\pm 1.01}$	$35.14_{\pm 1.50}$	$38.51_{\pm 2.13}$	54.60
	Plug-in ↑	$\mathbf{37.20}_{\pm 3.06}$	$\mathbf{79.11}_{\pm 1.34}$	$74.41_{\pm 1.27}$	$39.08_{\pm 2.02}$	$\mathbf{58.63}_{\pm 0.70}$	$\underline{96.00}_{\pm 0.88}$	$\mathbf{34.45}_{\pm 1.49}$	$\mathbf{39.85}_{\pm 2.14}$	**57.34**
	Offsite ZSG ↑	$14.00_{\pm 4.07}$	$11.43_{\pm 2.04}$	$13.05_{\pm 1.90}$	$9.56_{\pm 2.77}$	$25.87_{\pm 0.96}$	$16.20_{\pm 2.00}$	$\mathbf{33.37}_{\pm 1.52}$	$9.96_{\pm 2.94}$	16.68
	Offsite FTG ↑	$1.20_{\pm 4.31}$	$2.40_{\pm 1.93}$	$2.44_{\pm 1.82}$	$0.85_{\pm 2.85}$	$13.00_{\pm 0.99}$	$1.40_{\pm 1.34}$	$\mathbf{-0.69}_{\pm 2.11}$	$1.34_{\pm 3.02}$	2.74

Table 6: Scores and gaps are shown in %. Comparison results on eight datasets with Qwen2.5-7B. The best results are shown in **bold**, and the second-best results are underlined.

Method	*Setting*	OBQA	PIQA	ARC-E	ARC-C	HellaSwag	SciQ	WebQs	RACE	Avg.
Qwen2.5-7B ZS		$35.60_{\pm 3.03}$	$78.35_{\pm 1.36}$	$80.98_{\pm 1.14}$	$47.10_{\pm 2.06}$	$62.84_{\pm 0.68}$	$95.80_{\pm 0.90}$	$15.94_{\pm 1.15}$	$41.57_{\pm 2.16}$	57.27
Pretrained FT		$44.40_{\pm 3.15}$	$79.43_{\pm 1.33}$	$83.50_{\pm 1.08}$	$54.61_{\pm 2.06}$	$65.21_{\pm 0.67}$	$97.40_{\pm 0.71}$	$43.90_{\pm 1.56}$	$41.76_{\pm 2.16}$	63.78
OT	Offsite ZS ↓	$11.60_{\pm 2.03}$	$54.73_{\pm 1.64}$	$29.63_{\pm 1.33}$	$18.77_{\pm 1.61}$	$29.22_{\pm 0.64}$	$41.00_{\pm 2.20}$	$0.00_{\pm 0.00}$	$22.03_{\pm 1.82}$	25.87
	Offsite FT	$28.80_{\pm 2.87}$	$71.06_{\pm 1.50}$	$56.90_{\pm 1.44}$	$25.43_{\pm 1.80}$	$37.72_{\pm 0.68}$	$92.80_{\pm 1.16}$	$30.22_{\pm 1.44}$	$40.80_{\pm 2.15}$	47.97
	Plug-in ↑	$\mathbf{42.00}_{\pm 3.13}$	$\mathbf{80.30}_{\pm 1.31}$	$\underline{79.63}_{\pm 1.17}$	$\mathbf{48.98}_{\pm 2.07}$	$\underline{63.11}_{\pm 0.68}$	$\mathbf{97.20}_{\pm 0.74}$	$23.62_{\pm 1.33}$	$40.04_{\pm 2.15}$	$\underline{59.36}$
	Offsite ZSG ↑	$\mathbf{30.40}_{\pm 3.73}$	$25.57_{\pm 2.10}$	$50.00_{\pm 1.77}$	$\mathbf{30.21}_{\pm 2.62}$	$\mathbf{33.89}_{\pm 0.93}$	$56.20_{\pm 2.32}$	$23.62_{\pm 1.33}$	$18.01_{\pm 2.82}$	33.49
	Offsite FTG ↑	$13.20_{\pm 4.25}$	$\mathbf{9.24}_{\pm 1.99}$	$\mathbf{22.73}_{\pm 1.86}$	$23.55_{\pm 2.74}$	$\mathbf{25.39}_{\pm 0.96}$	$\mathbf{4.40}_{\pm 1.38}$	$-6.60_{\pm 1.96}$	$-0.76_{\pm 3.04}$	11.39
CRaSh	Offsite ZS ↓	$14.80_{\pm 2.25}$	$53.54_{\pm 1.65}$	$24.66_{\pm 1.25}$	$21.16_{\pm 1.69}$	$27.68_{\pm 0.63}$	$21.60_{\pm 1.84}$	$0.00_{\pm 0.00}$	$23.37_{\pm 1.85}$	23.35
	Offsite FT	$22.80_{\pm 2.66}$	$72.36_{\pm 1.48}$	$54.63_{\pm 1.45}$	$25.26_{\pm 1.80}$	$37.92_{\pm 0.68}$	$92.80_{\pm 1.16}$	$28.15_{\pm 1.41}$	$35.63_{\pm 2.10}$	46.19
	Plug-in ↑	$36.80_{\pm 3.06}$	$77.58_{\pm 1.38}$	$75.67_{\pm 1.25}$	$47.10_{\pm 2.06}$	$55.83_{\pm 0.70}$	$95.60_{\pm 0.92}$	$16.93_{\pm 1.18}$	$39.46_{\pm 2.14}$	55.62
	Offsite ZSG ↑	$22.00_{\pm 3.80}$	$24.04_{\pm 2.15}$	$\mathbf{51.01}_{\pm 1.77}$	$25.94_{\pm 2.66}$	$28.15_{\pm 0.94}$	$\mathbf{74.00}_{\pm 2.06}$	$16.93_{\pm 1.18}$	$16.09_{\pm 2.83}$	32.27
	Offsite FTG ↑	$14.00_{\pm 4.05}$	$5.22_{\pm 2.02}$	$21.04_{\pm 1.91}$	$21.84_{\pm 2.74}$	$17.91_{\pm 0.98}$	$2.80_{\pm 1.48}$	$-11.22_{\pm 1.84}$	$3.83_{\pm 3.00}$	9.43
gradOT	Offsite ZS ↓	$12.40_{\pm 2.09}$	$52.01_{\pm 1.65}$	$24.49_{\pm 1.25}$	$23.38_{\pm 1.75}$	$28.06_{\pm 0.63}$	$32.20_{\pm 2.09}$	$0.00_{\pm 0.00}$	$21.84_{\pm 1.81}$	24.30
	Offsite FT	$21.60_{\pm 2.61}$	$70.18_{\pm 1.51}$	$52.10_{\pm 1.45}$	$22.01_{\pm 1.71}$	$34.71_{\pm 0.67}$	$91.60_{\pm 1.24}$	$26.77_{\pm 1.39}$	$40.04_{\pm 2.15}$	44.88
	Plug-in ↑	$37.60_{\pm 3.07}$	$79.22_{\pm 1.34}$	$74.33_{\pm 1.27}$	$\underline{48.63}_{\pm 2.07}$	$58.73_{\pm 0.69}$	$\underline{95.80}_{\pm 0.90}$	$\underline{28.44}_{\pm 1.42}$	$\underline{40.80}_{\pm 2.15}$	57.94
	Offsite ZSG ↑	$25.20_{\pm 3.71}$	$\mathbf{27.21}_{\pm 2.13}$	$49.84_{\pm 1.78}$	$25.25_{\pm 2.71}$	$30.67_{\pm 0.93}$	$63.60_{\pm 2.28}$	$28.44_{\pm 1.42}$	$\mathbf{18.96}_{\pm 2.81}$	**33.64**
	Offsite FTG ↑	$\mathbf{16.00}_{\pm 4.03}$	$9.04_{\pm 2.02}$	$22.23_{\pm 1.93}$	$\mathbf{26.62}_{\pm 2.68}$	$24.02_{\pm 0.96}$	$4.20_{\pm 1.53}$	$\mathbf{1.67}_{\pm 1.99}$	$0.76_{\pm 3.04}$	**13.06**
GAP	Offsite ZS ↓	$26.00_{\pm 2.78}$	$72.69_{\pm 1.47}$	$67.17_{\pm 1.36}$	$32.25_{\pm 1.93}$	$40.81_{\pm 0.69}$	$85.60_{\pm 1.57}$	$1.38_{\pm 0.37}$	$32.18_{\pm 2.05}$	44.76
	Offsite FT	$39.60_{\pm 3.10}$	$78.45_{\pm 1.36}$	$75.51_{\pm 1.25}$	$39.59_{\pm 2.02}$	$50.23_{\pm 0.71}$	$96.40_{\pm 0.83}$	$40.35_{\pm 1.54}$	$38.31_{\pm 2.13}$	57.31
	Plug-in ↑	$\underline{41.60}_{\pm 3.12}$	$\underline{80.09}_{\pm 1.32}$	$\mathbf{80.39}_{\pm 1.15}$	$42.32_{\pm 2.04}$	$\mathbf{63.77}_{\pm 0.68}$	$\mathbf{97.20}_{\pm 0.74}$	$\mathbf{39.07}_{\pm 1.53}$	$\mathbf{44.83}_{\pm 2.18}$	**61.16**
	Offsite ZSG ↑	$15.60_{\pm 4.18}$	$7.40_{\pm 1.98}$	$13.22_{\pm 1.78}$	$10.07_{\pm 2.81}$	$22.96_{\pm 0.97}$	$11.60_{\pm 1.74}$	$\mathbf{37.69}_{\pm 1.57}$	$12.65_{\pm 2.99}$	16.40
	Offsite FTG ↑	$2.00_{\pm 4.40}$	$1.64_{\pm 1.90}$	$4.88_{\pm 1.70}$	$2.73_{\pm 2.87}$	$13.54_{\pm 0.98}$	$0.80_{\pm 1.11}$	$-1.28_{\pm 2.17}$	$\mathbf{6.52}_{\pm 3.05}$	3.85

B Additional Experiment Results

B.1 Results on Qwen2.5-3B and Qwen2.5-7B

The experimental results for Qwen2.5-3B and Qwen2.5-7B are presented in Tables 5 and 6, respectively. GAP achieves Plug-in macro averages of 57.34% and 61.16%, exceeding the strongest baseline, OT, by 2.76 and 1.80 percentage points, respectively. Both also exceed Pretrained ZS, by 2.86 and 3.89 points. These higher-utility operating points retain positive Offsite ZSG (16.68% and 16.40%) and Offsite FTG (2.74% and 3.85%), extending the plug-in utility gains to the Qwen family.

B.2 Experiments on Larger Models and More Diverse Architecture

To further examine whether the effectiveness of GAP is specific to the model scales and architectures used in our main experiments, we conduct additional evaluations on a larger dense language model and a Mixture-of-Experts (MoE) model. These experiments are designed to test the scalability and architectural generality of GAP under substantially different backbone settings.

We compare GAP with OT, the strongest baseline on the main Llama backbones, using a fixed fine-tuning configuration with a learning rate of $5e{-}5$ and 5 epochs for both additional models.

Scaling to a larger dense model. Table 7 reports Qwen2.5-32B results on OBQA, ARC-Easy, ARC-Challenge, and WebQuestions. GAP achieves a four-task Plug-in macro of 51.66%, improving over OT's 46.89% by 4.77 percentage points and over Pretrained ZS (47.80%) by 3.86 points. This operating point combines higher average utility with positive Offsite ZSG 25.69% and Offsite FTG 6.24%. GAP improves Plug-in over OT on three of the four tasks, supporting the applicability of the construction at a larger dense-model scale.

Generalization to MoE architectures. We also evaluate GAP on DeepSeek-V2-Lite (DeepSeek-AI, 2024), a 16B-parameter MoE model using Multi-head Latent Attention (MLA) and DeepSeekMoE. In Table 8, GAP raises the four-task Plug-in macro from OT's 48.31% to 50.47%, while increasing Offsite ZSG from 25.49% to 29.39% and Offsite FTG from 11.31% to 14.55%. Its Plug-in macro also exceeds Pretrained ZS (40.41%) by 10.06 percentage points. The simultaneous increases in utility and both gaps demonstrate a favorable aggregate gap-utility trade-off on this MoE backbone.

Together, these experiments show that the same construction and adapter-transfer procedure yields higher average Plug-in utility than OT and Pretrained ZS on both the larger dense and MoE backbones, with positive aggregate performance gaps.

Table 7: Scores and gaps are shown in %. Comparison results on four datasets with Qwen2.5-32B. The best Plug-in results are shown in **bold**.

Method	*Setting*	OBQA	ARC-E	ARC-C	WebQs	Avg.
Qwen2.5-32B ZS		$35.20_{\pm 3.03}$	$81.31_{\pm 1.13}$	$52.73_{\pm 2.06}$	$21.95_{\pm 1.30}$	47.80
Pretrained FT		$45.60_{\pm 3.16}$	$86.36_{\pm 1.00}$	$62.46_{\pm 2.00}$	$49.70_{\pm 1.57}$	61.03
OT	Offsite ZS ↓	$20.00_{\pm 2.53}$	$35.61_{\pm 1.39}$	$22.53_{\pm 1.73}$	$0.10_{\pm 0.10}$	19.56
	Offsite FT	$24.00_{\pm 2.71}$	$60.86_{\pm 1.42}$	$22.70_{\pm 1.73}$	$22.24_{\pm 1.31}$	32.45
	Plug-in ↑	$36.00_{\pm 3.04}$	$\mathbf{80.72}_{\pm 1.14}$	$41.13_{\pm 2.03}$	$29.72_{\pm 1.43}$	46.89
	Offsite ZSG ↑	$16.00_{\pm 3.96}$	$\mathbf{45.11}_{\pm 1.80}$	$18.60_{\pm 2.67}$	$29.62_{\pm 1.43}$	**27.33**
	Offsite FTG ↑	$\mathbf{12.00}_{\pm 4.07}$	$\mathbf{19.86}_{\pm 1.82}$	$\mathbf{18.43}_{\pm 2.67}$	$\mathbf{7.48}_{\pm 1.94}$	**14.44**
GAP	Offsite ZS ↓	$21.20_{\pm 2.59}$	$55.39_{\pm 1.44}$	$27.30_{\pm 1.84}$	$0.00_{\pm 0.00}$	25.97
	Offsite FT	$36.80_{\pm 3.06}$	$71.89_{\pm 1.30}$	$35.67_{\pm 1.98}$	$37.30_{\pm 1.52}$	45.42
	Plug-in ↑	$\mathbf{40.80}_{\pm 3.11}$	$79.29_{\pm 1.18}$	$\mathbf{51.88}_{\pm 2.07}$	$\mathbf{34.65}_{\pm 1.49}$	**51.66**
	Offsite ZSG ↑	$\mathbf{19.60}_{\pm 4.05}$	$23.90_{\pm 1.86}$	$\mathbf{24.58}_{\pm 2.77}$	$\mathbf{34.65}_{\pm 1.49}$	25.69
	Offsite FTG ↑	$4.00_{\pm 4.36}$	$7.40_{\pm 1.76}$	$16.21_{\pm 2.86}$	$-2.65_{\pm 2.13}$	6.24

B.3 Results on Domain-Specific Data

To further evaluate GAP in a domain-specific setting, we conduct additional experiments on the ATC-Easy subset of MedConceptsQA (Shoham & Rappoport, 2024) using Qwen2.5-7B. MedConceptsQA is a medical concept question-answering benchmark, and ATC-Easy focuses on the Anatomical Therapeutic Chemical classification. The task is formulated as four-way multiple-choice question answering.

Table 9 shows Plug-in 87.60%, an improvement of 59.60 percentage points over Pretrained ZS (28.00%) and equal to the Pretrained FT reference. GAP combines this domain-adaptation utility with Offsite ZSG 60.40% and Offsite FTG 10.40%. The positive FTG provides a task-utility incentive to return the locally trained adapters, while the owner construction continues to use only general-domain support data.

Table 8: Scores and gaps are shown in %. Comparison results on four datasets with DeepSeek-V2-Lite. The best Plug-in results are shown in **bold**.

Method	*Setting*	OBQA	ARC-E	ARC-C	WebQs	Avg.
DeepSeek-V2-Lite ZS		$32.40_{\pm 2.97}$	$76.18_{\pm 1.24}$	$43.69_{\pm 2.05}$	$9.35_{\pm 0.91}$	40.41
Pretrained FT		$43.20_{\pm 3.14}$	$80.22_{\pm 1.16}$	$48.29_{\pm 2.07}$	$47.93_{\pm 1.57}$	54.91
OT	Offsite ZS ↓	$20.80_{\pm 2.57}$	$45.45_{\pm 1.45}$	$24.91_{\pm 1.79}$	$0.10_{\pm 0.10}$	22.82
	Offsite FT	$32.00_{\pm 2.96}$	$57.74_{\pm 1.43}$	$28.84_{\pm 1.87}$	$29.43_{\pm 1.43}$	37.00
	Plug-in ↑	$39.20_{\pm 3.09}$	$76.52_{\pm 1.23}$	$\mathbf{45.05}_{\pm 2.06}$	$32.48_{\pm 1.47}$	48.31
	Offsite ZSG ↑	$\mathbf{18.40}_{\pm 4.02}$	$31.07_{\pm 1.90}$	$\mathbf{20.14}_{\pm 2.73}$	$32.38_{\pm 1.47}$	25.49
	Offsite FTG ↑	$7.20_{\pm 4.28}$	$18.78_{\pm 1.89}$	$16.21_{\pm 2.78}$	$3.05_{\pm 2.05}$	11.31
GAP	Offsite ZS ↓	$23.20_{\pm 2.68}$	$35.52_{\pm 1.39}$	$25.60_{\pm 1.80}$	$0.00_{\pm 0.00}$	21.08
	Offsite FT	$31.20_{\pm 2.94}$	$53.20_{\pm 1.45}$	$27.30_{\pm 1.84}$	$31.99_{\pm 1.46}$	35.92
	Plug-in ↑	$\mathbf{40.80}_{\pm 3.11}$	$\mathbf{80.47}_{\pm 1.15}$	$44.88_{\pm 2.06}$	$\mathbf{35.73}_{\pm 1.50}$	**50.47**
	Offsite ZSG ↑	$17.60_{\pm 4.11}$	$\mathbf{44.95}_{\pm 1.80}$	$19.28_{\pm 2.74}$	$\mathbf{35.73}_{\pm 1.50}$	**29.39**
	Offsite FTG ↑	$\mathbf{9.60}_{\pm 4.28}$	$\mathbf{27.27}_{\pm 1.85}$	$\mathbf{17.58}_{\pm 2.76}$	$\mathbf{3.74}_{\pm 2.09}$	**14.55**

Owner noise training and BI ranking use only the original 1,500 BCT support examples (BoolQ, TriviaQA, and CoPA), with no medical data. Medical data are used only for downstream fine-tuning and evaluation. The original six-setting grid is retained at all four total depths 16, 20, 24, and 28. At depths 24 and 28, we additionally train learning rates $\{2, 3\} \times 10^{-5}$ for $\{3, 5, 8\}$ epochs, yielding twelve candidates per extended depth. Each depth first selects its own highest-Offsite-validation checkpoint; only then are frozen depth-specific winners compared by Plug-in validation, requiring Offsite validation of at least 50% and a strictly positive validation transfer gain. The selected model has 28 total layers, learning rate 2e-05, and 8 epochs. Neither medical examples nor scores are used to rank individual layers or train the owner.

The native ATC-Easy file uses fixed train/validation/test row ranges [0,5000), [5000,5500), and [5500,6000), respectively. This exploratory extension follows earlier test exposure; unchanged validation measurements and identical model/adapter evaluations are reused, and the pretrained reference is shared across comparisons.

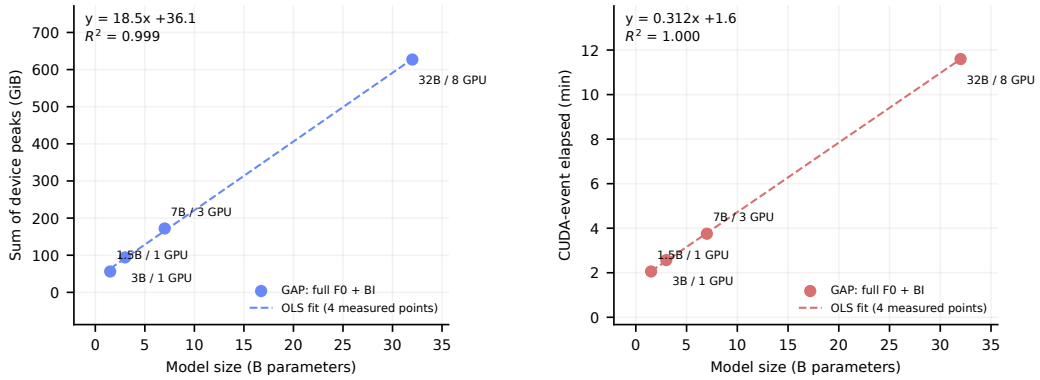
Table 9: Scores and gaps are shown in %. Experimental results of GAP on the ATC-Easy subset of MedConceptsQA using Qwen2.5-7B.

Method	Setting	MedConceptsQA(ATC-Easy)
Qwen2.5-7B	Pretrained ZS	$28.00_{\pm 2.01}$
	Pretrained FT	$87.60_{\pm 1.48}$
GAP	Offsite ZS	$27.20_{\pm 1.99}$
	Offsite FT	$77.20_{\pm 1.88}$
	Plug-in	$\mathbf{87.60}_{\pm 1.48}$
	Offsite ZSG ↑	$\mathbf{60.40}_{\pm 2.48}$
	Offsite FTG ↑	$\mathbf{10.40}_{\pm 2.39}$

B.4 Computational Overhead

We further analyze the computational cost of GAP, focusing on two practical efficiency metrics: peak GPU memory consumption and GPU runtime.

Scaling with model size. Figure 4 reports the four measured Qwen2.5 sizes 1.5, 3, 7, and 32B, using 1, 1, 3, and 8 NVIDIA H200 GPUs, respectively. The measurements cover complete F0 noise training, BI pruning, and final artifact construction, excluding initial model/tokenizer loading and post-hoc CPU audits. Memory is the sum of per-device peak allocated memory; runtime is the maximum per-device CUDA-event elapsed interval, including host-side gaps. The fitted lines summarize the observed resource costs under the indicated hardware configurations.



(a) Sum of per-device peak allocated memory during full emulator construction.

(b) Maximum per-device CUDA-event elapsed time for full emulator construction.

Figure 4: Computational overhead at four measured Qwen2.5 sizes, using the GPU counts annotated in the plots. The fitted lines summarize the observations under these hardware configurations.

Comparison with GradOT. Table 10 compares emulator-construction costs on Llama-3.2-3B. GAP takes 2.26 minutes versus 7.84 minutes for GradOT, giving a $3.47\times$ runtime speedup in the measured configuration. The table also reports the memory footprint of each method.

Table 10: Comparison of the GPU memory footprint and runtime between GAP and GradOT on Llama-3.2-3B.

Method	Peak Memory	GPU Runtime
GradOT	32.18 GiB	7.84 min
GAP	84.11 GiB	2.26 min

C Additional Ablation Studies

In this section, we provide additional ablation studies that complement the mechanism-level analysis in Section 4.4. We study the sensitivity of GAP to adversarial-training hyper-parameters. Finally, we analyze how the size and domain composition of the support set affect the gap-utility trade-off. These studies examine how useful plug-in adaptation and performance gaps are maintained across the reported perturbation-training and support-set configurations.

Algorithm 2 Norm-Aligned Gaussian Control with a Frozen BI Mask

Input: Original model $\boldsymbol{\theta}_{\text{orig}}$, uncompressed optimized model $\boldsymbol{\theta}_{\text{adv}}$, seed s, frozen retained-layer indices K
Copy $\boldsymbol{\theta}_{\text{orig}}$; initialize the random generator with seed s
for each parameter tensor $\mathbf{W}$ in the original middle layers **do**
  Compute stored-BF16 difference $\Delta\mathbf{W} = \mathbf{W}_{\text{adv}} - \mathbf{W}_{\text{orig}}$
  Set $N_W = \| \text{float32}(\Delta\mathbf{W}) \|_2$
  Draw a same-shape float32 tensor $\mathbf{R} \sim \mathcal{N}(0, \mathbf{I})$
  Set $\mathbf{R}_{\text{scaled}} = \mathbf{R}\, N_W / \|\mathbf{R}\|_2$
  Set $\mathbf{W}_{\text{noise}} = \mathbf{W}_{\text{orig}} + \text{BF16}(\mathbf{R}_{\text{scaled}})$ using BF16 addition
end for
Keep exactly the layers in K, including the unchanged first two and last two adapters
Return: The compressed random control; never recompute K using the seed or downstream scores

The fixed-mask comparison in Figure 3a uses seeds 42, 1024, 1337, 2026, 3407, 31415, 271828, 299792, 1602176, and 66260701. For each original middle-layer tensor, the target perturbation is

the stored-BF16 difference between the optimized and original weights. Gaussian noise is rescaled in float32 to that difference's flattened L_2 norm, cast to BF16, and added in BF16, as specified in Algorithm 2. Norm matching thus precedes rounding; realized post-addition norms need not be exactly equal. Non-middle parameters remain unchanged, and the same owner-support BI mask is applied to every seed without recomputation.

C.1 Hyper-parameter Sensitivity

We next study whether GAP requires delicate hyper-parameter tuning in the optimization process used to generate the perturbations. This is important because a method that only works under a narrow training configuration may be difficult to apply in practice. We conduct a grid search over learning rates and training epochs. The learning rates are selected from $\{5e{-}6, 8e{-}6, 1e{-}5, 2e{-}5, 5e{-}5\}$, and the number of training epochs ranges from 1 to 5. Evaluations are performed with Llama-3.2-3B on the validation sets of four datasets: ARC-Challenge, ARC-Easy, OpenBookQA, and WebQuestions. For each setting, we select the checkpoint with the highest Offsite FT. Table 11 reports average scores and the two performance gaps across the four datasets.

Across the tested configurations, Plug-in ranges from 47.37% to 49.67%, consistently exceeding the four-task Pretrained ZS reference of 40.52%. Offsite ZSG ranges from 29.99% to 33.53%, and Offsite FTG from 12.71% to 15.40%. Thus the tested settings maintain both improved plug-in utility and positive gaps, supporting a useful gap-utility trade-off across the perturbation-training grid.

Table 11: Scores and gaps are shown in %. Average scores and performance gaps of GAP on Llama-3.2-3B (Base), evaluated on the first-half validation splits of OpenBookQA, ARC-Easy, ARC-Challenge, and WebQuestions under various owner-training hyper-parameter configurations.

LR	Setting	Epoch				
		1	2	3	4	5
5e-6	Offsite ZS↓	17.52	17.46	17.48	17.65	17.38
	Offsite FT	34.97	35.28	34.97	34.49	35.09
	Plug-in↑	48.03	48.10	48.61	47.78	48.28
	Offsite ZSG ↑	30.51	30.64	31.13	30.13	30.90
	Offsite FTG ↑	13.06	12.82	13.64	13.29	13.19
8e-6	Offsite ZS↓	17.68	17.58	17.79	17.28	17.28
	Offsite FT	34.92	34.89	34.94	35.17	34.81
	Plug-in ↑	48.30	48.71	47.84	49.24	48.22
	Offsite ZSG ↑	30.62	31.13	30.05	31.96	30.94
	Offsite FTG ↑	13.38	13.82	12.90	14.07	13.41
1e-5	Offsite ZS↓	17.35	17.51	17.54	17.41	17.25
	Offsite FT	34.78	34.40	34.82	34.26	34.19
	Plug-in↑	48.57	48.79	47.53	48.12	47.37
	Offsite ZSG ↑	31.22	31.28	29.99	30.71	30.12
	Offsite FTG ↑	13.79	14.39	12.71	13.86	13.18
2e-5	Offsite ZS↓	17.10	17.55	17.23	16.71	17.37
	Offsite FT	34.80	34.78	34.80	34.58	34.15
	Plug-in↑	48.92	48.55	48.91	47.96	48.81
	Offsite ZSG ↑	31.82	31.00	31.68	31.25	31.44
	Offsite FTG ↑	14.12	13.77	14.11	13.38	14.66
5e-5	Offsite ZS↓	16.46	17.62	17.28	16.78	16.14
	Offsite FT	35.20	34.95	35.35	35.37	34.27
	Plug-in↑	48.75	48.81	48.83	<u>49.43</u>	**49.67**
	Offsite ZSG ↑	32.29	31.19	31.55	32.65	**33.53**
	Offsite FTG ↑	13.55	13.86	13.48	14.06	**15.40**
Pretrained ZS		40.52				

C.2 Ablation Studies on the Support Set

We further study whether GAP requires a carefully constructed support set. This question is important for practical deployment, where the model owner may only have access to a small support set or to support data whose domain does not exactly match the downstream task. We examine two factors: (i) the size of the support set, and (ii) the domain composition and sample distribution of the support set. Unless otherwise specified, all experiments in this section are conducted with Qwen2.5-3B.

Table 12: Scores and gaps are shown in %. Ablation study on the support set size for GAP with Qwen2.5-3B, using support budgets from 300 to 1800. Plug-in and the two performance gaps characterize the gap-utility trade-off. The best Plug-in result for each dataset is **bolded**.

Dataset	Setting	Support Set Size					
		300	600	900	1200	1500	1800
ARC-Challenge	Pretrained ZS	$44.71_{\pm 2.06}$					
	Pretrained FT	$49.32_{\pm 2.07}$					
	Offsite ZS↓	$20.14_{\pm 1.66}$	$21.67_{\pm 1.70}$	$19.97_{\pm 1.65}$	$19.80_{\pm 1.65}$	$29.52_{\pm 1.89}$	$20.31_{\pm 1.66}$
	Offsite FT	$30.03_{\pm 1.90}$	$32.42_{\pm 1.94}$	$30.38_{\pm 1.90}$	$29.52_{\pm 1.89}$	$38.23_{\pm 2.01}$	$31.74_{\pm 1.92}$
	Plug-in↑	$36.18_{\pm 1.99}$	$42.32_{\pm 2.04}$	$36.69_{\pm 1.99}$	$37.37_{\pm 2.00}$	$39.08_{\pm 2.02}$	$\mathbf{46.08}_{\pm 2.06}$
	Offsite ZSG ↑	$16.04_{\pm 2.59}$	$20.65_{\pm 2.66}$	$16.72_{\pm 2.59}$	$17.57_{\pm 2.59}$	$9.56_{\pm 2.77}$	$\mathbf{25.77}_{\pm 2.65}$
	Offsite FTG ↑	$6.15_{\pm 2.75}$	$9.90_{\pm 2.82}$	$6.31_{\pm 2.75}$	$7.85_{\pm 2.75}$	$0.85_{\pm 2.85}$	$\mathbf{14.34}_{\pm 2.82}$
ARC-Easy	Pretrained ZS	$77.95_{\pm 1.20}$					
	Pretrained FT	$80.47_{\pm 1.15}$					
	Offsite ZS↓	$45.62_{\pm 1.45}$	$47.31_{\pm 1.45}$	$46.46_{\pm 1.45}$	$46.46_{\pm 1.45}$	$61.36_{\pm 1.41}$	$45.20_{\pm 1.44}$
	Offsite FT	$62.88_{\pm 1.40}$	$63.47_{\pm 1.40}$	$62.88_{\pm 1.40}$	$62.88_{\pm 1.40}$	$71.97_{\pm 1.30}$	$62.21_{\pm 1.41}$
	Plug-in↑	$69.61_{\pm 1.33}$	$71.55_{\pm 1.31}$	$71.13_{\pm 1.32}$	$71.46_{\pm 1.31}$	$\mathbf{74.41}_{\pm 1.27}$	$72.98_{\pm 1.29}$
	Offsite ZSG ↑	$23.99_{\pm 1.97}$	$24.24_{\pm 1.95}$	$24.67_{\pm 1.96}$	$25.00_{\pm 1.95}$	$13.05_{\pm 1.90}$	$\mathbf{27.78}_{\pm 1.93}$
	Offsite FTG ↑	$6.73_{\pm 1.93}$	$8.08_{\pm 1.92}$	$8.25_{\pm 1.92}$	$8.58_{\pm 1.92}$	$2.44_{\pm 1.82}$	$\mathbf{10.77}_{\pm 1.91}$
OpenBookQA	Pretrained ZS	$30.80_{\pm 2.93}$					
	Pretrained FT	$37.60_{\pm 3.07}$					
	Offsite ZS↓	$16.80_{\pm 2.37}$	$24.00_{\pm 2.71}$	$17.60_{\pm 2.41}$	$18.00_{\pm 2.43}$	$23.20_{\pm 2.68}$	$20.00_{\pm 2.53}$
	Offsite FT	$28.40_{\pm 2.86}$	$34.40_{\pm 3.01}$	$29.60_{\pm 2.89}$	$26.40_{\pm 2.79}$	$36.00_{\pm 3.04}$	$32.80_{\pm 2.98}$
	Plug-in↑	$33.60_{\pm 2.99}$	$32.40_{\pm 2.97}$	$34.40_{\pm 3.01}$	$33.20_{\pm 2.98}$	$\mathbf{37.20}_{\pm 3.06}$	$34.00_{\pm 3.00}$
	Offsite ZSG ↑	$\mathbf{16.80}_{\pm 3.82}$	$8.40_{\pm 4.02}$	$\mathbf{16.80}_{\pm 3.86}$	$15.20_{\pm 3.85}$	$14.00_{\pm 4.07}$	$14.00_{\pm 3.92}$
	Offsite FTG ↑	$5.20_{\pm 4.14}$	$-2.00_{\pm 4.23}$	$4.80_{\pm 4.17}$	$\mathbf{6.80}_{\pm 4.08}$	$1.20_{\pm 4.31}$	$1.20_{\pm 4.23}$
WebQuestions	Pretrained ZS	$10.73_{\pm 0.97}$					
	Pretrained FT	$42.13_{\pm 1.55}$					
	Offsite ZS↓	$0.20_{\pm 0.14}$	$0.10_{\pm 0.10}$	$0.20_{\pm 0.14}$	$0.30_{\pm 0.17}$	$1.08_{\pm 0.32}$	$0.20_{\pm 0.14}$
	Offsite FT	$31.20_{\pm 1.45}$	$29.23_{\pm 1.43}$	$30.91_{\pm 1.45}$	$30.31_{\pm 1.44}$	$35.14_{\pm 1.50}$	$27.76_{\pm 1.41}$
	Plug-in↑	$29.72_{\pm 1.43}$	$27.76_{\pm 1.41}$	$27.85_{\pm 1.41}$	$27.66_{\pm 1.40}$	$\mathbf{34.45}_{\pm 1.49}$	$26.97_{\pm 1.39}$
	Offsite ZSG ↑	$29.52_{\pm 1.44}$	$27.66_{\pm 1.41}$	$27.65_{\pm 1.42}$	$27.36_{\pm 1.41}$	$\mathbf{33.37}_{\pm 1.52}$	$26.77_{\pm 1.40}$
	Offsite FTG ↑	$-1.48_{\pm 2.04}$	$-1.47_{\pm 2.01}$	$-3.06_{\pm 2.02}$	$-2.65_{\pm 2.01}$	$\mathbf{-0.69}_{\pm 2.11}$	$-0.79_{\pm 1.98}$

Effect of support set size. We first evaluate how GAP is affected by the amount of support data. We vary the support set size over $\{300, 600, 900, 1200, 1500, 1800\}$. To remain consistent with the main experimental setting, support examples are drawn evenly from BoolQ, TriviaQA, and CoPA. Table 12 reports the results on ARC-Easy, ARC-Challenge, OpenBookQA, and WebQuestions.

In Table 12, 1500 examples give the highest Plug-in scores on ARC-Easy, OpenBookQA, and WebQuestions, whereas 1800 gives the highest score on ARC-Challenge. Across the reported support budgets, the four-task Plug-in macro exceeds Pretrained ZS (41.05%), with positive macro Offsite ZSG and Offsite FTG. The four-task Plug-in macro averages for 1500 and 1800 examples are 46.29% and 45.01%, respectively, a modest difference of 1.28 percentage points. Given their similar aggregate utility and the smaller support-data budget, we retain the 1500-example support set used in our main experiments.

Effect of domain matching and sample distribution. We next examine whether GAP is sensitive to the domain composition of the support set. We group candidate support datasets into two broad domains: OpenBookQA and ARC-Easy are treated as science-education domain datasets, while BoolQ and TriviaQA are treated as general domain datasets. Based on these candidates, we construct seven support set combinations: OpenBookQA, ARC-Easy, OpenBookQA+ARC-Easy, BoolQ, TriviaQA, BoolQ+TriviaQA, and all four datasets combined. For each combination, we use a support

set size of 1500, the same as in the main experiments. The support examples are uniformly sampled from the datasets included in the corresponding combination. We evaluate the resulting models on ARC-Challenge, which belongs to the science-education domain, and WebQuestions, which belongs to the general domain. Table 13 reports the results.

Table 13: Scores and gaps are shown in %. Ablation study on the support set domain composition for GAP with Qwen2.5-3B. We construct support sets from science-education datasets, i.e., OpenBookQA and ARC-Easy, general-domain datasets, i.e., TriviaQA and BoolQ, and their combinations. Results on ARC-Challenge and WebQuestions characterize the gap-utility trade-off across support domains.

Dataset	Setting	Support Set Combination						
		OBQA	ARC-E	OBQA+ARC-E	TriviaQA	BoolQ	TriviaQA+BoolQ	All
ARC-C	Pretrained ZS				$44.71_{\pm 2.06}$			
	Pretrained FT				$49.32_{\pm 2.07}$			
	Offsite ZS↓	$22.35_{\pm 1.72}$	$20.99_{\pm 1.68}$	$20.31_{\pm 1.66}$	$23.04_{\pm 1.74}$	$33.28_{\pm 1.95}$	$20.65_{\pm 1.67}$	$20.65_{\pm 1.67}$
	Offsite FT	$44.20_{\pm 2.05}$	$27.99_{\pm 1.86}$	$28.50_{\pm 1.87}$	$41.30_{\pm 2.04}$	$48.81_{\pm 2.07}$	$47.78_{\pm 2.07}$	$43.52_{\pm 2.05}$
	Plug-in↑	$47.44_{\pm 2.06}$	$33.79_{\pm 1.96}$	$40.27_{\pm 2.03}$	$44.54_{\pm 2.05}$	$\mathbf{51.54}_{\pm 2.07}$	$50.51_{\pm 2.07}$	$47.61_{\pm 2.06}$
	Offsite ZSG ↑	$25.09_{\pm 2.68}$	$12.80_{\pm 2.58}$	$19.96_{\pm 2.62}$	$21.50_{\pm 2.69}$	$18.26_{\pm 2.84}$	$\mathbf{29.86}_{\pm 2.66}$	$26.96_{\pm 2.65}$
	Offsite FTG ↑	$3.24_{\pm 2.91}$	$5.80_{\pm 2.70}$	$\mathbf{11.77}_{\pm 2.76}$	$3.24_{\pm 2.89}$	$2.73_{\pm 2.93}$	$2.73_{\pm 2.93}$	$4.09_{\pm 2.91}$
WebQs	Pretrained ZS				$10.73_{\pm 0.97}$			
	Pretrained FT				$42.13_{\pm 1.55}$			
	Offsite ZS↓	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$
	Offsite FT	$40.94_{\pm 1.54}$	$41.44_{\pm 1.55}$	$42.03_{\pm 1.55}$	$34.06_{\pm 1.49}$	$41.73_{\pm 1.55}$	$40.26_{\pm 1.54}$	$40.16_{\pm 1.54}$
	Plug-in↑	$35.63_{\pm 1.50}$	$39.57_{\pm 1.53}$	$30.81_{\pm 1.45}$	$32.87_{\pm 1.47}$	$35.04_{\pm 1.50}$	$\mathbf{40.75}_{\pm 1.54}$	$32.78_{\pm 1.47}$
	Offsite ZSG ↑	$35.63_{\pm 1.50}$	$39.57_{\pm 1.53}$	$30.81_{\pm 1.45}$	$32.87_{\pm 1.47}$	$35.04_{\pm 1.50}$	$\mathbf{40.75}_{\pm 1.54}$	$32.78_{\pm 1.47}$
	Offsite FTG ↑	$-5.31_{\pm 2.15}$	$-1.87_{\pm 2.18}$	$-11.22_{\pm 2.12}$	$-1.19_{\pm 2.09}$	$-6.69_{\pm 2.16}$	$\mathbf{0.49}_{\pm 2.18}$	$-7.38_{\pm 2.13}$

The reported general-domain TriviaQA+BoolQ setting achieves Plug-in 50.51% on ARC-Challenge and 40.75% on WebQuestions, exceeding their Pretrained ZS references of 44.71% and 10.73%. It also gives the largest Offsite ZSG in each task (29.86% and 40.75%), together with positive Offsite FTG (2.73% and 0.49%). This comparison illustrates that general-domain support can yield useful plug-in performance and adapter-return gains without requiring a science-education support domain for ARC-Challenge.

D Gradient Alignment during Downstream Fine-Tuning

This appendix establishes the gradient-alignment and transfer results for GAP's two stages: training-based gradient-aligned perturbation followed by compression-based gradient-aligned perturbation. Following Section 3, the analysis bounds gradient differences at virtual adapter states obtained on the support set, extends the bound to the region visited during downstream fine-tuning, and accounts for BI-guided layer removal. The propositions state the regularity and coverage conditions used in each step.

D.1 Training-based perturbation and virtual adapter states

The margin-free expression in Equation 2 corresponds to $\tau = 1$, $\rho = 0$, and $\varepsilon = 0$ in Equation 9 for nonzero gradients. The experiments instead use the stabilized, tolerance-aware form below; the following bounds analyze that implemented form.

Let $a \in \mathbb{R}^p$ collect the parameters of the two bottom and two top adapter layers. Let θ_0 denote the original middle layers, and let $\theta = \theta_0 + \xi$ denote the adversarially perturbed, uncompressed middle layers. For a batch or distribution D, write $L_D(a, \theta)$ for the token-weighted language-model loss and $g_D(a, \theta) = \nabla_a L_D(a, \theta)$. The compressed emulator uses a fixed subset K of middle layers and has loss $L_D^K(a, \theta)$ and gradient $g_D^K(a, \theta)$. Adapter coordinates are identical before and after compression; the original middle layers are restored only when creating the plug-in model. This distinction permits comparison of adapter gradients even when the emulators have different depths.

During owner training, a support batch is partitioned for constructing one to three virtual gradient descent steps. Scheduled step counts cycle across optimizer batches, capped by the number of available inner examples. Each batch computes alignment only at its selected endpoint, not a sum of

losses at all three step counts. For each adapter parameter tensor $a^{(j)}$, the virtual update has the form

$$d_k^{(j)} = -\rho_v \frac{\mathrm{RMS}(a_k^{(j)})}{\mathrm{RMS}(\nabla_{a^{(j)}} L_{B_k}(a_k, \theta)) + \varepsilon} \nabla_{a^{(j)}} L_{B_k}(a_k, \theta), \qquad a_{k+1} = a_k + d_k, \tag{8}$$

with $\rho_v = 0.0025$; a missing or negligible gradient gives a zero step. The same increment is applied to both emulator and reference adapters. These steps are detached from the outer differentiation, and all adapter tensors are restored after forming the outer emulator gradient. The negative loss and gradient alignment are evaluated on the support batch at the resulting virtual state. In particular, with response weight zero, the query partition is not a separate held-out evaluation of the two active losses: gradient statistics use the full batch, split into two microbatches. No downstream examples are used in this construction.

At a virtual state, let u_i, v_i be the token-weighted mean adapter gradients of the reference and noisy models in adapter layer i, respectively. There are $q = 4$ adapter-layer groups. The implemented alignment term is

$$c_{\varepsilon,i} = \frac{\langle u_i, v_i \rangle}{\sqrt{(\|u_i\|^2 + \varepsilon)(\|v_i\|^2 + \varepsilon)}}, \qquad r_{\varepsilon,i} = \frac{1}{2} \log \frac{\|v_i\|^2 + \varepsilon}{\|u_i\|^2 + \varepsilon},$$
$$\mathcal{G} = \frac{1}{q} \sum_{i=1}^{q} \left([\tau - c_{\varepsilon,i}]_+^2 + [|r_{\varepsilon,i}| - \rho]_+^2 \right), \qquad (\tau, \rho) = (0.9, 0.25), \tag{9}$$

where $[z]_+ = \max(z, 0)$ and $\varepsilon = 10^{-12}$. Thus $\mathcal{G}$ includes a norm-ratio constraint, not only a directional constraint. The outer objective is $-L_B(a_k, \theta) + 25\mathcal{G}_B(a_k, \theta)$, with a_k treated as a fixed virtual state for outer differentiation. There is no loss-dependent rescaling of the alignment coefficient.

Proposition D.1 (Alignment loss bounds adapter-gradient error) *At a fixed batch and virtual state, suppose $\mathcal{G} \le \eta$, $s = \sqrt{q\eta} < \tau$, and $\min(\|u_i\|, \|v_i\|) \ge m > 0$ for every group. Define*

$$b = \rho + s + \tfrac{1}{2} \log(1 + \varepsilon/m^2), \qquad E_A = \|u\| \sqrt{(e^b - 1)^2 + 2e^b(1 - \tau + s)}, \tag{10}$$

where $u = (u_1, \ldots, u_q)$ and $v = (v_1, \ldots, v_q)$. Then $\|v - u\| \le E_A$.

Proof. Every nonnegative group penalty is at most $q\eta$, so $c_{\varepsilon,i} \ge \tau - s > 0$ and $|r_{\varepsilon,i}| \le \rho + s$. Positive regularized cosine is no larger than ordinary cosine, hence $c_i \ge \tau - s$. The lower norm bound implies $|\log(\|v_i\|/\|u_i\|) - r_{\varepsilon,i}| \le \frac{1}{2} \log(1 + \varepsilon/m^2)$. Writing $t_i = \|v_i\|/\|u_i\|$ gives $e^{-b} \le t_i \le e^b$. The exact identity

$$\|v_i - u_i\|^2 = \|u_i\|^2 \big((t_i - 1)^2 + 2t_i(1 - c_i) \big)$$

is therefore bounded by the corresponding summand of E_A^2. Sum over the disjoint adapter groups and take a square root. □

The bound retains the alignment tolerances τ and ρ explicitly, linking the achieved loss level η to the magnitude of the adapter-gradient discrepancy.

D.2 Alignment across adapter states and downstream trajectories

For a fixed finalized noisy emulator, define $\Delta_D(a) = g_D(a, \theta) - g_D(a, \theta_0)$ and $\Delta H_D(a) = \nabla_a \Delta_D(a)$. Assuming differentiability, the fundamental theorem of calculus gives

$$\Delta_D(a + hv) - \Delta_D(a) = h \int_0^1 \Delta H_D(a + thv) v \, dt, \qquad \|v\| = 1. \tag{11}$$

Consequently, if the gradient difference is at most E_A at both endpoints, the norm of the averaged directional Hessian difference is at most $2E_A/h$. If ΔH_D is K_H-Lipschitz along the segment, then

$$\|\Delta H_D(a)v\| \le 2E_A/h + K_H h/2. \tag{12}$$

Indeed, subtract $\Delta H_D(a)v$ inside the integral and bound the remainder by $\int_0^1 K_H th \, dt$. This gives a directional-curvature bound from endpoint alignment for the same loss and finalized emulator.

The virtual gradient descent steps in Equation 8 use support gradients of the noisy model to obtain non-initial adapter states, as in Equation 1. We relate these virtual adapter states to downstream adapter states under the following coverage and distribution conditions.

Proposition D.2 (From virtual adapter states to downstream states) *Fix the finalized noisy model θ before compression. Let (a_j, B_j) pair each virtual adapter state a_j with its associated support batch B_j. Suppose $\|\Delta_{B_j}(a_j)\| \le E_A$ for every pair. For each downstream adapter state a in the region $\mathcal{A}$, assume there is such a virtual state a_j and associated batch B_j satisfying*

$$\|a - a_j\| \le r, \qquad \|\Delta_{B_j}(a) - \Delta_{B_j}(a_j)\| \le L_\Delta \|a - a_j\|, \qquad \|\Delta_D(a) - \Delta_{B_j}(a)\| \le \delta.$$

Then $\sup_{a\in\mathcal{A}} \|\Delta_D(a)\| \le E_A + L_\Delta r + \delta$.

Proof. Add and subtract $\Delta_{B_j}(a)$ and $\Delta_{B_j}(a_j)$ and apply the triangle inequality and the three assumed bounds. □

Here E_A is a common upper bound on the gradient error at the virtual adapter states, r bounds the distance from each downstream state to a corresponding virtual state, and L_Δ bounds how quickly the gradient difference changes between them. The term δ combines minibatch-to-support and support-to-downstream discrepancies, as in Section 3. The bounds at virtual states are evaluated for the finalized θ; when using measurements from an earlier θ_j, the corresponding model-drift term is included in E_A. These coverage and distribution conditions are assumptions on the downstream region, not guarantees from support-set alignment alone. All owner construction and layer selection continue to use only the support set.

D.3 Compression-based perturbation and gradient stability

Following Men et al. (2024), the BI of a middle block B_i is the mean of $1 - \cos(h, B_i(h))$ over calibration tokens. GAP computes this statistic on the support set after training-based perturbation and retains the highest-BI middle layers under the specified retained-layer budget. The default main comparisons use OT-matched depth; the disclosed relaxed-depth probes need not match OT. For nonzero h and $z = B_i(h)$, let $t = \|z\|/\|h\|$ and $b_i(h) = 1 - \cos(h, z)$. The identity

$$\frac{\|B_i(h) - h\|^2}{\|h\|^2} = (t-1)^2 + 2t\, b_i(h) \tag{13}$$

separates directional and norm changes. If $\|h\| \le H$ and $|t - 1| \le \kappa$, then

$$\mathbb{E}\|B_i(h) - h\|^2 \le H^2\big(\kappa^2 + 2(1+\kappa)\mathbb{E}b_i(h)\big). \tag{14}$$

Together with norm control, low BI bounds the forward perturbation. The following result incorporates block-Jacobian regularity to translate this bound into adapter-gradient stability.

Proposition D.3 (Gradient perturbation from deleting one block) *For a fixed example, represent a network's loss as $F(a) = f_a(B(p_a))$, with $h = p_a$ the prefix output and f_a the suffix including the loss. The middle block B has no adapter parameters. After deleting it, $\bar{F}(a) = f_a(p_a)$. Suppose, on the relevant segment from h to $B(h)$,*

$$\begin{aligned}
&\|B(h) - h\| \le d, \quad \|DB(h) - I\| \le J, \quad \|D_a p_a\| \le P,\\
&\|\nabla_h f_a(h)\| \le Q, \quad \|\nabla_h f_a(z) - \nabla_h f_a(h)\| \le L\|z - h\|,\\
&\|\partial_a f_a(z) - \partial_a f_a(h)\| \le A\|z - h\|.
\end{aligned}$$

Then

$$\|\nabla_a F(a) - \nabla_a \bar{F}(a)\| \le Ad + P\big((1+J)Ld + JQ\big). \tag{15}$$

All Jacobian norms are operator norms, and ∂_a holds the suffix input fixed, while ∇_a denotes the total derivative.

Proof. The chain rule yields

$$\begin{aligned}
\nabla_a F &= \partial_a f_a(B(h)) + (D_a p_a)^T DB(h)^T \nabla_h f_a(B(h)),\\
\nabla_a \bar{F} &= \partial_a f_a(h) + (D_a p_a)^T \nabla_h f_a(h).
\end{aligned}$$

In the second term of their difference, add and subtract $(D_a p_a)^T DB(h)^T \nabla_h f_a(h)$. Apply the assumed Lipschitz bounds, $\|DB(h)\| \le 1 + J$, and the triangle inequality. □

For several deleted blocks, consider a sequence of hybrid networks that removes one additional block at each step. Applying Proposition D.3 at each deletion and telescoping bounds the total gradient error by the sum of the right-hand sides of Equation 15. The constants must hold for the hybrid networks, not just the original uncompressed network.

More explicitly, write $\widehat{b}_i$ for the BI of a deleted block measured on the support set and assume the corresponding expected BI at every required hybrid input and adapter state is at most $\widehat{b}_i + \zeta_i$. Here $\zeta_i \geq 0$ accounts for calibration error, distribution shift, adapter drift, and earlier deletions. Under uniform constants as above, define

$$\bar{d}_i = H_i\sqrt{\kappa_i^2 + 2(1+\kappa_i)(\widehat{b}_i + \zeta_i)}, \qquad E_C = \sum_{i \notin K} \left[A_i \bar{d}_i + P_i\big((1+J_i)L_i\bar{d}_i + J_i Q_i\big)\right]. \quad (16)$$

Equations 14 and 15, together with $\mathbb{E}\|r\| \leq \sqrt{\mathbb{E}\|r\|^2}$, give $\|g_D^K(a,\theta) - g_D(a,\theta)\| \leq E_C$. For this expected-gradient statement, differentiation and expectation must commute; the displayed uniform bounds are sufficient under the usual integrability assumptions.

Corollary D.4 (BI ranking minimizes a common gradient-error bound) *If the number of deleted blocks is fixed and all constants $H_i, \kappa_i, \zeta_i, A_i, P_i, J_i, L_i, Q_i$ in Equation 16 can be bounded by the same nonnegative values independently of the deletion set and order, deleting the lowest-BI blocks minimizes this common upper bound.*

Proof. Each summand is the same nondecreasing function of $\widehat{b}_i$. If a selected deletion has higher BI than a retained block, exchanging them cannot increase the bound. Repeated exchanges produce the lowest-BI deletion set. □

The common-constant conditions yield a support-based selection rule: at a fixed deletion budget, the smallest BI scores minimize the displayed bound. The activation-norm, Jacobian, and hybrid-distribution terms quantify the stability conditions entering this ranking result.

D.4 Dual perturbation and downstream gradient alignment

Combining Propositions D.1 and D.2 with Equation 16 gives the sufficient bound

$$\|g_D^K(a,\theta) - g_D(a,\theta_0)\| \leq \epsilon_{\text{tot}} := E_A + L_\Delta r + \delta + E_C, \qquad a \in \mathcal{A}. \quad (17)$$

As in Equation 6, training-based gradient-aligned perturbation targets the gradient error E_A at virtual adapter states. The assumed coverage and gradient-variation conditions contribute $L_\Delta r$, while δ accounts for changing from support to downstream data. Compression-based gradient-aligned perturbation controls the layer-removal term E_C through BI selection and the stated stability conditions.

Proposition D.5 (A transferred update and a finite training trajectory) *Let $F(a) = L_D(a,\theta_0)$ be the plug-in loss, let $g = \nabla F(a)$, and let $\widetilde{g} = g_D^K(a,\theta)$. Suppose F is L-smooth on a region containing the update segment and $\|\widetilde{g} - g\| \leq \epsilon_{\text{tot}}$. Then*

$$F(a - \alpha\widetilde{g}) \leq F(a) - \alpha\|g\|^2 + \alpha\|g\|\epsilon_{\text{tot}} + \frac{L\alpha^2}{2}(\|g\| + \epsilon_{\text{tot}})^2. \quad (18)$$

If $r_g = \|g\| > \epsilon_{\text{tot}}$ and $L > 0$, any $0 < \alpha < 2r_g(r_g - \epsilon_{\text{tot}})/[L(r_g + \epsilon_{\text{tot}})^2]$ strictly decreases this loss.

For gradient-descent fine-tuning on the compressed noisy model and the original model, respectively, consider $\widetilde{a}_{t+1} = \widetilde{a}_t - \alpha_t g_D^K(\widetilde{a}_t,\theta)$ and $a_{t+1} = a_t - \alpha_t g_D(a_t,\theta_0)$, starting from the same adapters and using the same step sizes. As in Section 3, suppose both trajectories remain in the region, the reference gradient is L-Lipschitz there, and the gradient difference at the same adapter state is bounded by ϵ_t. Their distance satisfies

$$\|\widetilde{a}_T - a_T\| \leq \sum_{t=0}^{T-1} \alpha_t\epsilon_t \prod_{j=t+1}^{T-1}(1+\alpha_j L). \quad (19)$$

Proof. Smoothness gives $F(a-\alpha\widetilde{g}) \le F(a)-\alpha\langle g,\widetilde{g}\rangle+L\alpha^2\|\widetilde{g}\|^2/2$. Use $\langle g,\widetilde{g}\rangle \ge \|g\|^2-\|g\|\epsilon_{\mathrm{tot}}$ and $\|\widetilde{g}\| \le \|g\|+\epsilon_{\mathrm{tot}}$; solving the resulting strict quadratic inequality gives the step-size condition. For the trajectory, add and subtract $g_D(\widetilde{a}_t,\theta_0)$ to obtain $e_{t+1} \le (1+\alpha_t L)e_t+\alpha_t\epsilon_t$, with $e_0=0$. Induction yields Equation 19. □

With constant α,ϵ and $L>0$, this gives $e_T \le \epsilon[(1+\alpha L)^T-1]/L$; for $L=0$, it gives $e_T \le T\alpha\epsilon$. These expressions quantify finite-horizon stability as a function of the step sizes and alignment errors. If F is also M-Lipschitz between the two endpoints, the plug-in loss difference is at most Me_T.

Implication for downstream fine-tuning. Under the stated assumptions, the two perturbation stages provide a uniform same-state gradient bound on the downstream region. For gradient-descent fine-tuning, the resulting recurrence controls the deviation between emulator-trained and full-model adapter trajectories. Together with the descent result, this establishes the connection between maintaining gradient alignment during tuning and obtaining useful plug-in updates.

E Why BI Fits Compression-Based Gradient-Aligned Perturbation

Matching the deletion operation. Deleting a residual block $B_l(h)=h+R_l(h)$ replaces it with the identity. BI directly measures the directional change between its input and output:

$$\mathrm{BI}_l = \frac{1}{2}\mathbb{E}\left\| \frac{h_l}{\|h_l\|} - \frac{B_l(h_l)}{\|B_l(h_l)\|} \right\|^2. \tag{20}$$

Following ShortGPT (Men et al., 2024), GAP removes low-BI blocks of the already perturbed model. Equation 14 links this score to the forward change under norm control, and Proposition D.3 incorporates the block Jacobian to bound the resulting adapter-gradient error.

Comparison with uniform OT selection. Uniform layer dropping in OT (Xiao et al., 2023) allocates depth without adapting to the functional contribution of each block. BI instead directs the retained-layer budget toward blocks with larger representational changes. Under the common stability constants in Corollary D.4, deleting the lowest-BI blocks minimizes the same upper bound over all fixed-size deletion sets, including the uniform set. This provides a selection-level justification for using BI within GAP. The matched-capacity comparison in Table 3 evaluates this replacement with the same perturbed owner model.

Comparison with CRaSh representation clustering. CRaSh (Zhang et al., 2023) clusters layers using representation similarity and combines removal with sharing. Its linear CKA criterion preserves similarity under orthogonal changes of feature coordinates: for a token-representation matrix H and orthogonal Q,

$$(HQ)(HQ)^\top = HH^\top. \tag{21}$$

Thus CKA emphasizes the structure of a representation, whereas direct deletion must also preserve compatibility with the surrounding blocks in their existing coordinates. BI compares the input and output directions in those coordinates, making it closely matched to identity replacement. In GAP, this inexpensive structural criterion complements the explicit adapter-gradient alignment learned in the training-based stage. The comparison concerns the selection criteria; the respective complete methods also differ in their construction operations.