A NEW FORGETTING-RISK CONSTRAINED OPTIMIZATION FOR MODEL FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Fine-tuning a large pre-trained model is often necessary for strong downstream performance, but can degrade generalization and cause catastrophic forgetting of pre-trained knowledge. Existing forgetting-mitigation methods commonly regularize parameter changes or control selected spectral structures, such as suppressing intruder dimensions or protecting top singular components. Such component-wise control may provide limited spectral coverage, as unprotected directions can become prominent during adaptation and contribute to forgetting. Motivated by the observation that unconstrained adaptation can develop prominent task-specific spectral directions associated with forgetting, we regularize directional flexibility by restricting updates to an admissible subspace. We formulate forgetting-aware fine-tuning with joint adaptation-subspace and update-magnitude constraints, using the dominant column subspace of the pre-trained weights as a data-free, model-informed choice. We develop two realizations: Representation-Space Projected LoRA (RSP-LoRA), which projects LoRA updates onto the selected subspace, and Vector-based Feature Adaptation (VeFA), which satisfies the constraint through feature-side parameterization. Across image classification, natural language understanding, and Qwen2.5-7B mathematical reasoning, our methods substantially mitigate forgetting while maintaining competitive downstream performance. Ablations show that subspace restriction itself reduces forgetting, while the dominant pre-trained subspace generally provides a more favorable stability–plasticity trade-off than dimension-matched random and bottom subspaces.

1 INTRODUCTION

Pre-trained models on large-scale datasets have demonstrated strong generalization and transferable representations across a wide range of domains and tasks (Bommasani et al., 2021; Radford et al., 2021). Due to distributional differences between pre-training and downstream data, fine-tuning is often necessary to achieve strong downstream task performance. Meanwhile, downstream adaptation is typically performed with substantially less data than pre-training, making it important to adapt the model without unnecessarily disrupting broadly transferable capabilities acquired during pre-training.

Standard fine-tuning strategies include full fine-tuning and linear probing, which update all model parameters and only the final prediction layer, respectively (Kumar et al., 2022). To reduce computational cost and improve adaptation efficiency, parameter-efficient fine-tuning (PEFT) methods (Houlsby et al., 2019; Zaken et al., 2021; Hu et al., 2022), such as LoRA, focus on updating only a small number of parameters or injecting low-rank updates into frozen weights. However, fine-tuning can degrade generalization due to over-adaptation to task-specific patterns, leading to catastrophic forgetting of broadly transferable knowledge acquired during pre-training (French, 1999; Kirkpatrick et al., 2017; Kemker et al., 2018; Serra et al., 2018).

A common strategy for mitigating forgetting in fine-tuning is to constrain the distance between the fine-tuned and pre-trained models, thereby regularizing the magnitude of the fine-tuning update (Xuhong et al., 2018; Wortsman et al., 2022; Tian et al., 2023b). Such methods can effectively limit excessive parameter changes, but update magnitude alone may not fully characterize forgetting: updates of similar magnitude can affect the pre-training objective differently depending on their directions. This motivates considering not only how much the model adapts, but also the directional properties of adaptation.

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.

Recent work has shown that forgetting is also closely related to changes in the spectral structure of the pre-trained weights. For example, Xiong & Xie (2026) and Jin et al. (2026) protect selected top singular structures of the pre-trained model during fine-tuning, while Shuttleworth et al. (2024) suppress newly emerged *intruder dimensions* post hoc, i.e., high-ranking fine-tuned singular directions that are nearly orthogonal to the pre-trained singular directions. These findings suggest that adaptation geometry can be closely associated with forgetting, motivating explicit regularization of directional adaptation beyond update magnitude alone.

However, existing spectral-control methods either suppress intruder dimensions post hoc or protect selected top singular structures during fine-tuning. Such component-wise control may provide limited spectral coverage as the importance of singular directions can change during fine-tuning. Motivated by the observation that unconstrained adaptation can produce task-specific spectral changes associated with forgetting (e.g., intruder dimensions), we regularize directional flexibility during fine-tuning. Restricting updates to a fixed subspace reduces feasible update orientations, limiting opportunities for such spectral changes to emerge. The subspace can be chosen in different ways; we use the dominant column subspace of the pre-trained weights as a data-free, model-informed choice that maximizes pre-trained weight-energy coverage under a fixed dimensional budget.

Based on this perspective, we formulate forgetting-aware fine-tuning with joint adaptation-subspace and update-magnitude constraints. We consider two realizations of the subspace constraint. Representation-Space Projected LoRA (RSP-LoRA) projects LoRA updates onto the dominant pre-trained column subspace, while Vector-based Feature Adaptation (VeFA) uses a more restrictive feature-side parameterization that satisfies the constraint by construction and may suit small-data settings. VeFA is further motivated by effect equivalence modeling (EEM) of lurking variables (Huang et al., 2015; Sabbaghi & Huang, 2018), which represents unobserved effects through equivalent transformations of observed features. Both methods can be combined with magnitude regularization to jointly control directional flexibility and update magnitude.

Our contributions are summarized as follows. First, we analyze fine-tuning forgetting risk through update magnitude and directional sensitivity, and formulate forgetting-aware fine-tuning with joint adaptation-subspace and update-magnitude constraints. Second, we develop two realizations, Representation-Space Projected LoRA (RSP-LoRA) and Vector-based Feature Adaptation (VeFA), using the dominant column subspace of the pre-trained weights as a data-free and model-informed adaptation space. Third, experiments across image classification, natural language understanding, and Qwen2.5-7B mathematical reasoning demonstrate substantial forgetting mitigation with competitive downstream performance. Ablation studies show that restricting directional adaptation freedom itself reduces forgetting, while the dominant pre-trained subspace provides a more favorable stability–plasticity trade-off than matched random and bottom subspaces; controlled rank-one studies further isolate the retention benefit of subspace restriction beyond parameter count and update magnitude.

2 Preliminaries on Model Fine-Tuning and Forgetting

2.1 Model Fine-Tuning and its Formulation

We focus on fine-tuning *large-scale pre-trained models* to individual downstream tasks, rather than sequential continual learning. The downstream fine-tuning data are generally substantially smaller than the corpora used for large-scale pre-training, although we do not require every downstream task to follow a strict few-shot setting. Large-scale pre-trained models have learned rich and transferable representations during pre-training. Our goal is to enable effective downstream adaptation while mitigating the forgetting of broadly transferable knowledge acquired during pre-training.

Let $f_{\boldsymbol{W}_{\mathrm{pt}}}$ denote the original pre-trained model with parameters $\boldsymbol{W}_{\mathrm{pt}} = \{\boldsymbol{W}_{\mathrm{pt}}^{(\ell)}\}_{\ell=1}^{L}$. The principles outlined in the problem formulation apply to any dense layers in deep learning models, though we only focus on certain weights in Transformers in our experiments for practical reasons. When no ambiguity arises, we omit the layer index ℓ and use $\boldsymbol{W}_{\mathrm{pt}}$ to denote an individual pre-trained weight matrix. Given a downstream fine-tuning dataset $\mathcal{D}_{\mathrm{ft}}$, standard fine-tuning starts from the pre-trained parameters $\boldsymbol{W}_{\mathrm{pt}}$ and learns adapted parameters

$$\boldsymbol{W}_{\mathrm{ft}} = \arg\min_{\boldsymbol{W}\in\mathcal{A}} \mathcal{L}(\boldsymbol{W};\mathcal{D}_{\mathrm{ft}}), \qquad \mathcal{L}(\boldsymbol{W};\mathcal{D}_{\mathrm{ft}}) = \mathbb{E}_{(\boldsymbol{X},\boldsymbol{y})\sim\mathcal{D}_{\mathrm{ft}}}\big[\ell_{\mathrm{task}}(\boldsymbol{X},\boldsymbol{y};f_{\boldsymbol{W}})\big], \tag{1}$$

where $\mathcal{A}$ denotes the adaptation family, such as full fine-tuning or LoRA, and ℓ_{task} is the task loss.

2.2 Forgetting and Intruder Dimensions

Fine-Tuning aims to improve in-domain performance on $\mathcal{D}_{\text{ft}}$, but may cause forgetting by degrading transferable representations acquired during pre-training. To quantify forgetting, one would ideally evaluate $f_{\boldsymbol{W}_{\text{ft}}}$ on the pre-training distribution and compare its loss with that of $f_{\boldsymbol{W}_{\text{pt}}}$: $R_{\text{F}}(\boldsymbol{W}_{\text{ft}}) = \mathcal{L}(\boldsymbol{W}_{\text{ft}}; \mathcal{D}_{\text{pt}}) - \mathcal{L}(\boldsymbol{W}_{\text{pt}}; \mathcal{D}_{\text{pt}})$, where a more positive $R_{\text{F}}(\boldsymbol{W}_{\text{ft}})$ indicates more forgetting.

However, the original large-scale pre-training data may not always be available. When such data are unavailable, we use performance changes on held-out tasks as an empirical proxy for forgetting (Wortsman et al., 2022; Tian et al., 2023a). After fine-tuning on a source dataset $\mathcal{D}_{\text{ft}}^{(a)}$, we evaluate the fine-tuned model on held-out target datasets $\{\mathcal{D}_{\text{te}}^{(b)} : b \neq a\}$. We measure its performance change relative to $f_{\boldsymbol{W}_{\text{pt}}}$ as $R^{(b)}(\boldsymbol{W}_{\text{ft}}) = \mathcal{M}(f_{\boldsymbol{W}_{\text{ft}}}, \mathcal{D}_{\text{te}}^{(b)}) - \mathcal{M}(f_{\boldsymbol{W}_{\text{pt}}}, \mathcal{D}_{\text{te}}^{(b)})$. Here, $\mathcal{M}$ is a task-specific metric for which larger values are better. A more negative $R^{(b)}(\boldsymbol{W}_{\text{ft}})$ indicates more forgetting. Since all methods share the same pre-trained reference, their retention can equivalently be compared using held-out performance directly.

Most existing forgetting-mitigation methods constrain the distance between $\boldsymbol{W}_{\text{ft}}$ and $\boldsymbol{W}_{\text{pt}}$. This limits parameter drift but does not explicitly preserve the spectral structure of the pre-trained weights. Recent work has shown that fine-tuning can introduce prominent singular directions that are nearly orthogonal to the pre-trained singular structure and are associated with forgetting. Shuttleworth et al. (2024) formalize such directions as intruder dimensions.

Definition 2.1 (Intruder Dimension)**.** Let $\boldsymbol{W}_{\text{pt}} = \sum_j \sigma_{\text{pt},j} \boldsymbol{u}_{\text{pt},j} \boldsymbol{v}_{\text{pt},j}^\top$ and $\boldsymbol{W}_{\text{ft}} = \sum_i \sigma_{\text{ft},i} \boldsymbol{u}_{\text{ft},i} \boldsymbol{v}_{\text{ft},i}^\top$ denote the singular value decompositions (SVDs) of the pre-trained and fine-tuned weight matrices, respectively. A left singular vector $\boldsymbol{u}_{\text{ft},i}$ among the top-k singular directions of $\boldsymbol{W}_{\text{ft}}$ is an *intruder dimension* if $\max_j \left|\boldsymbol{u}_{\text{pt},j}^\top \boldsymbol{u}_{\text{ft},i}\right| < \epsilon$, $i \leq k$, where ϵ is the pairwise cosine-similarity threshold.

Through extensive empirical studies, Shuttleworth et al. (2024) showed that suppressing the highest-ranking intruder dimension after fine-tuning can substantially mitigate forgetting with limited impact on downstream performance.

3 A Constrained Formulation for Model Fine-Tuning with Reduced Forgetting Risk

3.1 Forgetting-Risk Factorization and Subspace-Constrained Adaptation

We begin by analyzing the sources of forgetting risk arising from model fine-tuning. Assuming that the pre-trained parameters are near a stationary point of the pre-training objective, a second-order approximation, together with the Fisher/GGN (Martens, 2020) and layer-wise block-diagonal approximations (Martens & Grosse, 2015), gives

$$R_{\text{F}}(\boldsymbol{W}_{\text{ft}}) \approx \frac{1}{2} \sum_{\ell=1}^{L} \text{vec}(\boldsymbol{\Delta}^{(\ell)})^\top \boldsymbol{F}_{\text{pt}}^{(\ell)} \text{vec}(\boldsymbol{\Delta}^{(\ell)}), \tag{2}$$

where $\boldsymbol{\Delta}^{(\ell)} = \boldsymbol{W}_{\text{ft}}^{(\ell)} - \boldsymbol{W}_{\text{pt}}^{(\ell)}$ denotes the fine-tuning update at layer ℓ, and $\boldsymbol{F}_{\text{pt}}^{(\ell)}$ denotes the layer-wise Fisher/GGN approximation to the curvature of the pre-training objective. When pre-training data are available, $\boldsymbol{F}_{\text{pt}}$ can be estimated and the resulting quadratic forgetting-risk penalty can be incorporated into $\mathcal{L}(\boldsymbol{W}; \mathcal{D}_{\text{ft}})$, as in Fisher/EWC-based regularization (Kirkpatrick et al., 2017; Zheng et al., 2026). In our setting, however, the pre-training data are unavailable.

Equation 2 can be factorized into update magnitude and directional sensitivity. Let $m^{(\ell)} = \|\boldsymbol{\Delta}^{(\ell)}\|_F$ and $\boldsymbol{v}^{(\ell)} = \text{vec}(\boldsymbol{\Delta}^{(\ell)})/m^{(\ell)}$. Defining the total squared update magnitude $M = \sum_{\ell=1}^{L} (m^{(\ell)})^2$ and the magnitude-weighted directional sensitivity $\bar{D} = \frac{1}{M} \sum_{\ell=1}^{L} (m^{(\ell)})^2 \boldsymbol{v}^{(\ell)\top} \boldsymbol{F}_{\text{pt}}^{(\ell)} \boldsymbol{v}^{(\ell)}$, we obtain $R_{\text{F}}(\boldsymbol{W}_{\text{ft}}) \approx \widehat{R}_{\text{F}} = \frac{1}{2} M \bar{D}$. When pre-training data are unavailable, $\bar{D}$ cannot be directly estimated. A common alternative is to regularize update magnitude through penalties toward the pre-trained weights (Xuhong et al., 2018), projection-based regularization (Tian et al., 2023b), or post-hoc weight interpolation (Wortsman et al., 2022).

However, the factorization indicates that updates of similar magnitude can incur different forgetting risks depending on their directional sensitivity $\bar{D}$. Without access to $\boldsymbol{F}_{\mathrm{pt}}$, these sensitive directions cannot generally be identified or explicitly avoided. Existing spectral approaches address adaptation geometry by limiting deviations from the pre-trained spectral structure. Prior work protects selected singular structures during fine-tuning (Xiong & Xie, 2026; Jin et al., 2026) or suppresses newly emerged intruder dimensions post hoc (Shuttleworth et al., 2024). As shown in Fig. 1, intruder scaling reduces M sufficiently to lower $\widehat{R}_{\mathrm{F}}$ and observed forgetting, whereas scaling a nearby non-intruder increases both M and $\bar{D}$, accompanied by increased forgetting. When M is matched to LoRA, the interventions still exhibit different forgetting behavior associated with changes in $\bar{D}$. These observations further motivate controlling adaptation geometry beyond update magnitude alone.

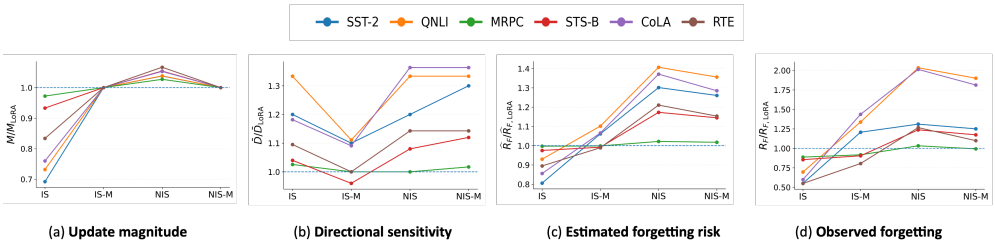


Figure 1: Mechanism analysis of spectral interventions. IS and NIS denote intruder scaling and nearest non-intruder scaling, respectively; IS-M and NIS-M denote their magnitude-matched variants.

However, existing spectral-control methods either suppress newly emerged intruder dimensions post hoc or protect selected top singular structures during fine-tuning. Such component-wise control may provide limited spectral coverage, particularly as the relative importance of singular directions can change during adaptation. Motivated by the observation that unconstrained directional adaptation can lead to highly task-specific spectral changes associated with forgetting, we instead *regularize the directional flexibility available during fine-tuning*. Specifically, restricting updates to a fixed admissible subspace reduces the feasible set of update orientations, thereby limiting the opportunity for highly task-specific spectral changes to emerge during adaptation.

The fixed admissible adaptation subspace can be chosen in different ways, ranging from an arbitrary random subspace to subspaces informed by the pre-trained model. In this work, we use the high-energy spectral subspace of the pre-trained weights as a data-free, deterministic, and model-informed choice. Importantly, this subspace is not assumed to approximate the low-curvature eigenspace of $\boldsymbol{F}_{\mathrm{pt}}$, nor is it assumed to be optimal for minimizing forgetting. Instead, under a fixed dimensional budget, it maximizes the pre-trained weight energy covered by the admissible adaptation space.

Specifically, we define $r_\alpha = \min\{J \in \mathbb{Z}_{>0} : \sum_{j=1}^{J} \sigma_{\mathrm{pt},j}^2 \geq \alpha \|\boldsymbol{W}_{\mathrm{pt}}\|_F^2\}$, where $\sigma_{\mathrm{pt},j}$ is the j-th largest singular value and $\alpha \in (0,1)$ is the retained spectral-energy ratio. We then define $\boldsymbol{W}_{\mathrm{pt}}^{(\alpha)} = \boldsymbol{U}_{\mathrm{pt},r_\alpha} \boldsymbol{\Sigma}_{\mathrm{pt},r_\alpha} \boldsymbol{V}_{\mathrm{pt},r_\alpha}^\top$ and $\mathcal{S}_0^{(\alpha)} = \mathrm{Col}(\boldsymbol{W}_{\mathrm{pt}}^{(\alpha)}) = \mathrm{span}(\boldsymbol{U}_{\mathrm{pt},r_\alpha})$. Among all r_α-dimensional output subspaces,

$$\mathcal{S}_0^{(\alpha)} \in \arg \max_{\substack{\mathcal{S} \subseteq \mathbb{R}^{d_{\mathrm{out}}} \\ \dim(\mathcal{S}) = r_\alpha}} \|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{W}_{\mathrm{pt}}\|_F^2 ,$$

where $\boldsymbol{P}_{\mathcal{S}}$ denotes the orthogonal projector onto $\mathcal{S}$. Thus, while any r_α-dimensional subspace imposes the same directional budget, the dominant pre-trained subspace discards the least pre-trained weight energy. If downstream adaptation reuses directions expressed by the pre-trained mapping, this greater coverage may preserve more plasticity while the restricted budget regularizes adaptation, potentially yielding a better stability–plasticity trade-off. Since weight-energy coverage does not guarantee downstream relevance or low Fisher sensitivity, we evaluate this hypothesis empirically against dimension-matched random and bottom subspaces in section 4.5.

Fine-tuning is then constrained by $\mathrm{Col}(\boldsymbol{W}_{\mathrm{ft}} - \boldsymbol{W}_{\mathrm{pt}}) \subseteq \mathcal{S}_0^{(\alpha)}$, such that adaptation-induced output directions remain within the selected pre-trained spectral subspace while the original pre-trained model remains the base. The retained-energy ratio α serves as a control parameter for the stability–plasticity trade-off: smaller subspaces impose stronger directional regularization but may be insufficient to

express the output corrections required by downstream adaptation, whereas broader subspaces provide greater adaptation flexibility but weaker directional control. Since weight-energy coverage does not guarantee downstream relevance or low Fisher sensitivity, we evaluate this trade-off empirically without assuming that forgetting or downstream performance varies monotonically with α.

3.2 Subspace-Constrained Adaptation

We formulate fine-tuning with joint adaptation-subspace and update-magnitude constraints:

$$\min_{\{\Delta \boldsymbol{W}^{(\ell)}\}} \quad \mathcal{L}\Big(\{\boldsymbol{W}_{\mathrm{pt}}^{(\ell)} + \Delta \boldsymbol{W}^{(\ell)}\}; \mathcal{D}_{\mathrm{ft}}\Big) \tag{3a}$$

$$\text{s.t.} \quad \mathrm{Col}\Big(\Delta \boldsymbol{W}^{(\ell)}\Big) \subseteq \mathcal{S}_0^{(\alpha,\ell)}, \qquad \forall \ell, \tag{3b}$$

$$\Omega^{(\ell)}\Big(\Delta \boldsymbol{W}^{(\ell)}\Big) \leq \gamma^{(\ell)}, \qquad \forall \ell. \tag{3c}$$

Constraint 3b restricts the admissible adaptation subspace, while Constraint 3c controls the update magnitude, where $\Omega^{(\ell)}(\Delta \boldsymbol{W}^{(\ell)})$ denotes a magnitude measure and $\gamma^{(\ell)}$ its upper bound. The subspace constraint controls the directional degrees of freedom available to adaptation but does not guarantee a reduction in the unknown Fisher directional sensitivity. Its effect on forgetting therefore depends on the selected subspace and is evaluated empirically.

We consider two realizations of the adaptation-subspace constraint: projection and parameterization. Projection explicitly maps each update onto $\mathcal{S}_0^{(\alpha,\ell)}$, whereas parameterization satisfies the constraint by construction. These realizations primarily enforce Constraint 3b and can be combined with training-time regularization methods, such as Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) and Fast Trainable Projection (FTP) (Tian et al., 2023b), to regularize the extent of adaptation in practice. In our implementation, EWC importance is estimated using only the fine-tuning data and does not require access to the pre-training distribution.

Representation-Space Projected LoRA (RSP-LoRA). We project the LoRA update onto the retained pre-trained spectral subspace as $\boldsymbol{W}_{\mathrm{ft}}^{(\ell)} = \boldsymbol{W}_{\mathrm{pt}}^{(\ell)} + \boldsymbol{U}_{\mathrm{pt},r_\alpha^{(\ell)}}^{(\ell)} \boldsymbol{U}_{\mathrm{pt},r_\alpha^{(\ell)}}^{(\ell)\top} \boldsymbol{B}^{(\ell)} \boldsymbol{A}^{(\ell)}$, where $r_\alpha^{(\ell)}$ is determined for each layer. Thus, $\mathrm{Col}(\Delta \boldsymbol{W}^{(\ell)}) \subseteq \mathcal{S}_0^{(\alpha,\ell)}$ at each post-update model state, while the update is applied to the pre-trained weight matrix. RSP-LoRA retains the low-rank flexibility of LoRA within the selected output subspace without assuming that this subspace is Fisher-optimal.

Vector-based Feature Adaptation (VeFA). As an alternative to explicit projection, feature adaptation parameterizes the update through the retained pre-trained spectral structure as $\boldsymbol{W}_{\mathrm{ft}}^{(\ell)} = \boldsymbol{W}_{\mathrm{pt}}^{(\ell)} + \boldsymbol{W}_{\mathrm{pt}}^{(\alpha,\ell)} \boldsymbol{M}^{(\ell)}$. Since $\Delta \boldsymbol{W}^{(\ell)} = \boldsymbol{W}_{\mathrm{pt}}^{(\alpha,\ell)} \boldsymbol{M}^{(\ell)}$, it follows that $\mathrm{Col}(\Delta \boldsymbol{W}^{(\ell)}) \subseteq \mathcal{S}_0^{(\alpha,\ell)}$ by construction. We consider a lightweight realization by setting $\boldsymbol{M}^{(\ell)} = \boldsymbol{\Lambda}^{(\ell)} = \mathrm{diag}(\boldsymbol{\lambda}^{(\ell)})$, yielding

$$\boldsymbol{h}^{(\ell)} = \boldsymbol{W}_{\mathrm{pt}}^{(\ell)} \boldsymbol{h}^{(\ell-1)} + \boldsymbol{W}_{\mathrm{pt}}^{(\alpha,\ell)} \left(\boldsymbol{\lambda}^{(\ell)} \odot \boldsymbol{h}^{(\ell-1)}\right). \tag{4}$$

We initialize $\boldsymbol{\lambda}^{(\ell)} = \mathbf{0}$, so fine-tuning starts exactly from the original pre-trained model. VeFA therefore provides a lightweight and parameter-efficient realization of the adaptation-subspace constraint.

Stability–Plasticity tradeoff. RSP-LoRA and VeFA satisfy the same adaptation-subspace constraint through different parameterizations, resulting in distinct stability–plasticity trade-offs. RSP-LoRA permits flexible low-rank updates within the selected subspace, retaining greater adaptation capacity for tasks that require substantial task-specific learning. In contrast, VeFA employs a lightweight and more restrictive feature-wise parameterization that provides stronger structural regularization and may be particularly suitable for few-shot settings where over-adaptation is a primary concern. This stronger constraint, however, may limit downstream adaptation when greater plasticity is required. We evaluate these complementary behaviors empirically across different adaptation regimes.

4 Experiments

4.1 Experiment Setup

Our experiments examine: (1) whether constraining the admissible adaptation subspace mitigates forgetting beyond standard low-rank adaptation and post-hoc spectral correction; (2) whether adaptation-

subspace constraints complement magnitude regularization; and (3) how the dimensionality and construction of the admissible subspace affect the stability–plasticity trade-off.

Pre-trained models and downstream tasks. We evaluate across three pre-trained model families and task domains: CLIP ViT-B/16 on seven image-classification datasets, RoBERTa-Base on six GLUE tasks, and Qwen2.5-7B for mathematical reasoning. For CLIP, forgetting is evaluated on held-out datasets; for RoBERTa, it is measured using MLM pseudo-loss on samples from its pre-training corpora; and for Qwen2.5-7B, retention is evaluated on held-out reasoning benchmarks. Detailed datasets, evaluation protocols, and forgetting measurements are provided in Appendix D.1.

All three model families use Transformer backbones. Unless otherwise specified, all methods use the same target modules within each model setting: query/key/value projections for CLIP, query/value projections for RoBERTa, and selected attention and MLP projections for Qwen2.5-7B, following the OPLoRA setting.

We compare our proposed methods with a comprehensive set of PEFT and forgetting-mitigation baselines. In addition to standard LoRA, we consider two post-hoc methods: LoRA-WISE (Wortsman et al., 2022) interpolates the fine-tuned and pre-trained weights, while Intruder-Scaling-LoRA (Shuttleworth et al., 2024) suppresses intruder dimensions after fine-tuning. We include LoRA-EWC (Zheng et al., 2026) and LoRA-FTP (Tian et al., 2023a) as training-time regularization methods, and OPLoRA (Xiong & Xie, 2026) as a training-time spectral subspace-control method. We further include LoRA$_{r=1}$ and VeRA (Kopiczko et al., 2023) as parameter-efficient baselines with few trainable parameters.

Hyperparameters are selected using only in-domain validation performance before forgetting evaluation. Unless otherwise specified, all LoRA-based methods (i.e., all methods except VeRA and VeFA) follow the same training protocol as the LoRA baseline. Full implementation and hyperparameter details are provided in Appendix D.2.

4.2 Mitigating Forgetting in Fine-Tuning CLIP for Image Classification

We focus on the 16-shot setting, where only 16 training examples per class are used for downstream adaptation. We run each method with three random seeds, report the average in-domain accuracy, and use the median-performing checkpoint selected by in-domain validation accuracy for forgetting evaluation. Unless otherwise specified, all RSP-LoRA and VeFA variants use a fixed retained-energy ratio of $\alpha = 0.9$ across datasets. The results are reported in Tab. 1.

In the 16-shot setting, the compared methods achieve broadly similar in-domain performance, with ID averages mostly concentrated around 88%. When paired with the same type of regularizer, RSP-LoRA consistently improves retention over LoRA, with $\bar{R}$ Avg improving from -5.19 to -2.86 under FTP and from -3.39 to -2.41 under EWC. RSP-LoRA alone also improves $\bar{R}$ Avg from -5.70 to -3.14 while maintaining comparable in-domain performance, supporting the benefit of proactively restricting the admissible adaptation space. Post-hoc Intruder Scaling provides a smaller improvement from -5.70 to -4.77. VeFA achieves the highest ID average of 89.2% while improving $\bar{R}$ Avg to -1.13, demonstrating strong retention without sacrificing downstream adaptation. With EWC, VeFA-EWC further improves $\bar{R}$ Avg to -0.59 while maintaining an ID average of 88.2%. Overall, these results show that constraining the admissible adaptation subspace can mitigate forgetting, while combining the subspace constraint with update regularization can provide further retention benefits.

4.3 Mitigating Forgetting in Fine-Tuning Natural Language Models

We evaluate RoBERTa-Base (Liu et al., 2019) on the GLUE benchmark (Wang et al., 2018). Following the LoRA setup (Hu et al., 2022), we adapt selected self-attention projections and fully train the task-specific classification head. Unless otherwise specified, all LoRA-based methods use rank $r = 8$ to maintain the same number of trainable adapter parameters, while LoRA$_{r=1}$ is included as a low-parameter baseline. For RSP-LoRA variants and VeFA, we use an adaptation-subspace energy ratio of $\alpha = 0.90$ unless otherwise specified. Unlike the original LoRA setup (Hu et al., 2022), we do not initialize any task from an MNLI-fine-tuned checkpoint; all six tasks start from the same original pre-trained model $\boldsymbol{W}_{\mathrm{pt}}$. For each method and task, we run five trials with different random seeds, select the best epoch in each trial according to validation performance, and report the median over

Table 1: In-domain accuracy and held-out forgetting on seven image classification datasets using the CLIP ViT-B/16 backbone.

Method	ID Avg ↑	Caltech101	Food101	Pets	Flowers	EuroSAT	DTD	UCF101	$\bar{R}$ Avg ↑
LoRA	88.2	-6.3	-3.9	-5.4	-7.7	+0.4	-7.1	-9.9	-5.70
Intruder-Scaling-LoRA	88.1	-5.3	-3.4	-4.3	-6.5	+0.3	-6.2	-8.0	-4.77
LoRA-WISE	88.1	-0.8	-0.6	-1.8	**+0.1**	**+0.6**	-1.6	-2.3	-0.91
LoRA-EWC	87.8	-2.7	-2.9	-3.6	-1.8	-1.1	-5.8	-5.8	-3.39
LoRA-FTP	88.2	-5.6	-3.7	-4.6	-6.2	-0.4	-6.3	-9.5	-5.19
VeRA	87.4	-1.0	-0.9	**-0.2**	-1.1	0	-2.2	**-0.4**	-0.83
RSP-LoRA	88.3	-2.5	-1.7	-3.2	-3.1	-0.3	-3.9	-7.2	-3.14
RSP-LoRA-FTP	88.4	-3.2	-1.5	-2.6	-3.2	-1.4	-4.5	-3.6	-2.86
RSP-LoRA-EWC	88.4	-0.8	-0.7	-2.6	-3.2	+0.3	-2.9	-7.0	-2.41
VeFA	**89.2**	**+0.3**	-0.1	-1.6	-1.1	-0.9	-2.3	-2.2	-1.13
VeFA-EWC	88.2	+0.2	**+0.6**	-0.9	-1.2	0.0	**-1.5**	-1.3	**-0.59**

Table 2: In-domain performance and forgetting on GLUE using RoBERTa-Base. "ID Avg" is the average in-domain performance, and forgetting is measured by the change in MLM pseudo-loss on sampled pre-training data relative to the original pre-trained model $\boldsymbol{W}_{\mathrm{pt}}$.

Method	ID Avg ↑	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	Retention Avg ↑
LoRA	**84.2**	-0.281	-0.448	-0.108	-0.262	-0.376	-1.342	-0.469
LoRA$_{r=1}$	83.5	-0.207	-0.224	-0.144	-0.224	-0.124	-0.702	-0.271
Intruder-Scaling-LoRA	82.6	-0.155	-0.397	-0.065	-0.183	-0.206	-1.147	-0.359
LoRA-WISE	81.8	-0.053	-0.317	-0.044	-0.143	-0.132	-0.559	-0.208
LoRA-EWC	83.4	-0.270	-0.105	0.003	-0.410	-0.064	-0.027	-0.145
LoRA-FTP	81.6	-0.310	-0.187	-0.045	-0.355	-0.101	-0.586	-0.264
VeRA	83.8	-0.200	-0.371	-0.100	-0.392	-0.298	-0.385	-0.291
RSP-LoRA	83.6	-0.199	-0.414	-0.109	-0.205	-0.224	-0.443	-0.266
RSP-LoRA-EWC	82.9	-0.049	-0.063	0.011	-0.093	-0.094	-0.015	**-0.050**
VeFA-EWC	81.7	-0.133	-0.051	-0.048	-0.123	-0.068	-0.021	-0.074

the five trials. The checkpoint with median validation performance is used for forgetting evaluation. Full hyperparameter settings are provided in Appendix D.2.

As shown in Tab. 2, the methods exhibit different stability–plasticity trade-offs, with ID averages ranging from 81.6% to 84.2%. Forgetting is measured by the change in MLM pseudo-loss on the sampled pre-training data relative to the original pre-trained model $\boldsymbol{W}_{\mathrm{pt}}$, with larger values indicating less forgetting. Training-time regularization reduces forgetting relative to LoRA, with Retention Avg improving from -0.469 to -0.145 with EWC and -0.264 with FTP. Post-hoc control also reduces forgetting, reaching -0.208 with WISE and -0.359 with Intruder-Scaling-LoRA. Adding the adaptation-subspace constraint to EWC improves Retention Avg from -0.145 to -0.050. Low-parameter methods also exhibit reduced forgetting, with LoRA$_{r=1}$ and VeRA achieving -0.271 and -0.291, respectively.

To examine the forgetting-risk factorization, we analyze the association of update magnitude M and directional sensitivity $\bar{D}$ with observed forgetting across methods and tasks. Directional sensitivity is positively associated with forgetting (Spearman $\rho = 0.586$, $p < 10^{-4}$), whereas update magnitude alone shows no significant association ($\rho = -0.127$, $p = 0.389$). The estimated risk $\widehat{R}_{\mathrm{F}} = \frac{1}{2} M \bar{D}$ is also positively correlated with forgetting ($\rho = 0.554$, $p < 10^{-4}$). These results suggest that adaptation geometry provides complementary information beyond update magnitude alone, without implying a causal relationship. Detailed results are provided in Appendix E.3.

4.4 Mathematical Reasoning

For the mathematical reasoning experiments, we fine-tune Qwen2.5-7B on 10,000 examples sampled from MetaMathQA and compare LoRA, OPLoRA, Intruder-Scaling-LoRA, LoRA-FTP, RSP-LoRA, and RSP-LoRA-FTP under a shared training protocol. For RSP-LoRA and RSP-LoRA-FTP, we use an adaptation-subspace energy ratio of $\alpha = 0.90$. In particular, LoRA-FTP and RSP-LoRA-FTP use identical FTP hyperparameters, allowing us to assess the additional effect of the adaptation-subspace

Table 3: Fine-tuning and held-out capability evaluation on Qwen2.5-7B following the OPLoRA evaluation protocol. Results are reported as mean ± standard deviation over three random seeds. "Overall" denotes the average score across all five benchmarks.

Method	Fine-Tuning Eval (↑)		Held-Out Capability Eval (↑)			Overall (↑)
	MATH	GSM8K	ARC-e	ARC-c	SIQA	
Qwen2.5-7B	13.92	73.62	80.60	48.04	54.81	54.20
LoRA	$43.16_{\pm 0.27}$	$83.85_{\pm 0.44}$	$80.39_{\pm 0.05}$	$49.91_{\pm 0.30}$	$50.87_{\pm 0.16}$	$61.64_{\pm 0.16}$
OPLoRA-128	$42.92_{\pm 0.16}$	$83.85_{\pm 0.40}$	$80.77_{\pm 0.21}$	$49.74_{\pm 0.77}$	$51.02_{\pm 0.10}$	$61.66_{\pm 0.28}$
Intruder-Scaling-LoRA	$43.48_{\pm 0.08}$	$84.31_{\pm 0.27}$	$80.47_{\pm 0.09}$	$49.74_{\pm 0.12}$	$50.67_{\pm 0.14}$	$61.73_{\pm 0.05}$
LoRA-FTP	$43.12_{\pm 0.24}$	$83.85_{\pm 0.26}$	$81.48_{\pm 0.01}$	$50.85_{\pm 0.20}$	$51.79_{\pm 0.19}$	$62.22_{\pm 0.01}$
RSP-LoRA	$42.96_{\pm 0.37}$	$85.05_{\pm 0.13}$	$81.14_{\pm 0.22}$	$50.54_{\pm 0.82}$	$51.05_{\pm 0.13}$	$62.15_{\pm 0.26}$
RSP-LoRA-FTP	$42.50_{\pm 0.04}$	$84.08_{\pm 0.12}$	$81.78_{\pm 0.11}$	$52.13_{\pm 0.23}$	$52.46_{\pm 0.02}$	$62.59_{\pm 0.07}$

constraint under the same magnitude regularization. Full experimental settings are provided in Appendix Tables 12–14.

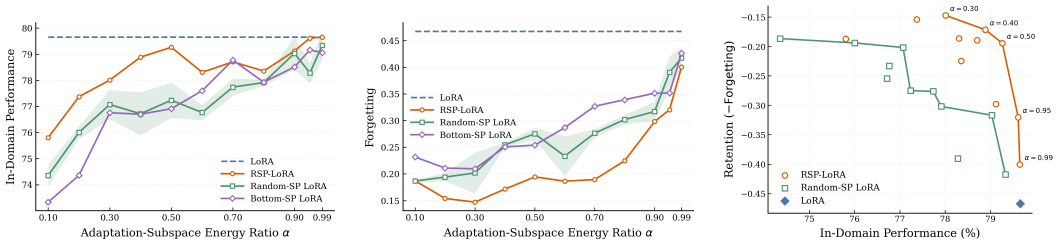
Table 3 reports fine-tuning performance on MATH and GSM8K and held-out capability evaluation on ARC-e, ARC-c, and SIQA. Fine-tuning substantially improves mathematical reasoning over the pre-trained Qwen2.5-7B model. RSP-LoRA maintains comparable fine-tuning performance to LoRA while improving all three held-out scores from $80.39/49.91/50.87$ to $81.14/50.54/51.05$. Under the same FTP regularization, RSP-LoRA-FTP further improves the held-out scores from $81.48/50.85/51.79$ to $81.78/52.13/52.46$, while maintaining comparable fine-tuning performance.

Overall, RSP-LoRA-FTP achieves the highest average score of 62.59, compared with 62.22 for LoRA-FTP, 62.14 for RSP-LoRA, and 61.64 for LoRA. These results show that restricting the admissible adaptation subspace can improve held-out capability retention and complement magnitude regularization while preserving competitive downstream adaptation performance. We analyze the performance–resource trade-off of RSP-LoRA across energy thresholds in Appendix F. The results show that moderate thresholds can provide favorable performance with modest overhead, while larger retained subspaces incur increasing training-memory costs without monotonic performance gains.

4.5 Ablation Study

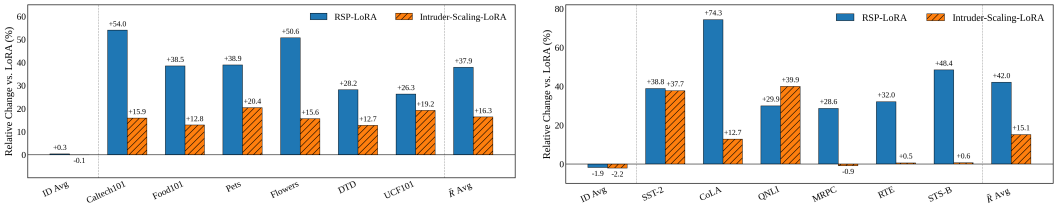
(1) **Effect of adaptation-space budget and subspace construction**. We vary $\alpha \in \{0.1, 0.2, \ldots, 0.9, 0.95, 0.99\}$ and compare RSP-LoRA with dimension-matched Random-SP and Bottom-SP LoRA. For each α, the dominant-subspace energy criterion determines the layer-wise dimension r_α. Random-SP uses a randomly sampled orthonormal basis of dimension r_α, while Bottom-SP uses the r_α left singular vectors associated with the smallest singular values in the thin SVD. Thus, the three methods differ only in subspace construction. For Random-SP, we report the mean and standard deviation over three sampled subspaces. As shown in Fig. 2(a)–(b), both random and bottom subspaces generally reduce forgetting relative to LoRA, confirming that restricting directional flexibility itself provides regularization. However, RSP-LoRA consistently achieves lower forgetting and generally higher in-domain performance than both alternatives. In particular, Bottom-SP often sacrifices more downstream performance while retaining less effectively, showing that dimensional restriction alone does not explain the gains. The dominant pre-trained subspace therefore provides a more favorable stability–plasticity trade-off under the same directional budget.

These gains cannot be explained by reduced downstream learning. At $\alpha = 0.95$, RSP-LoRA nearly preserves LoRA's in-domain performance (79.614% versus 79.657%) while reducing forgetting by 31.5%; at $\alpha = 0.99$, it matches LoRA within 0.01 percentage points while reducing forgetting by 14.3%. Conversely, decreasing α from 0.30 to 0.10 degrades both in-domain performance and retention, showing that tighter constraints do not monotonically improve stability. The Pareto frontiers in Fig. 2(c) further show that RSP-LoRA generally achieves better retention at comparable in-domain performance than Random-SP. Overall, subspace restriction provides general regularization, while alignment with the dominant pre-trained directions better preserves adaptation capacity.



(a) In-domain performance. (b) Forgetting (lower is better). (c) Empirical Pareto frontiers.

Figure 2: Effect of the adaptation-space budget and subspace construction on the stability–plasticity trade-off with LoRA rank $r = 8$.



(a) Image classification with CLIP. (b) NLU with RoBERTa-Base.

Figure 3: Proactive subspace restriction versus post-hoc Intruder Scaling at rank one. RSP-LoRA and LoRA have matched parameter count and update magnitude. Results are relative to LoRA.

(2) Controlled comparison at rank one. We compare $\text{LoRA}_{r=1}$, $\text{RSP-LoRA}_{r=1}$, and post-hoc Intruder Scaling under controlled parameter count and update magnitude. As shown in Fig. 3, RSP-LoRA improves average retention over both baselines with limited changes in in-domain performance. These results provide controlled evidence that the retention benefit of RSP-LoRA cannot be explained solely by parameter count or update magnitude, and are consistent with a benefit from proactively restricting the admissible adaptation subspace. Full experimental details are provided in Appendix E.4.

5 Conclusion

This work studies forgetting in fine-tuning from the joint perspective of adaptation magnitude and directional flexibility. Motivated by the observation that unconstrained directional adaptation can produce highly task-specific spectral changes associated with forgetting, we formulate fine-tuning with adaptation-subspace and update-magnitude constraints. We use the dominant high-energy subspace of the pre-trained weights as a data-free and model-informed adaptation space, and develop two realizations, RSP-LoRA and VeFA, through projection-based and feature-based adaptation.

Across CLIP, RoBERTa, and Qwen2.5-7B, the proposed methods mitigate forgetting while maintaining competitive downstream performance, and can be combined with magnitude regularization for further retention benefits. Ablation studies further clarify the role of the adaptation-space constraint: restricting directional freedom reduces forgetting even with random and bottom subspaces, while the dominant pre-trained subspace generally provides a more favorable stability–plasticity trade-off under the same dimensional budget. Controlled rank-one experiments further show retention benefits beyond parameter count and update magnitude alone.

The main limitation is the stability–plasticity trade-off of restricting the adaptation space. An overly restrictive subspace can limit downstream adaptation without further reducing forgetting, while the dominant pre-trained subspace is not guaranteed to minimize forgetting. Moreover, the original pre-training distribution is typically unavailable, requiring forgetting to be evaluated through held-out capabilities or representative pre-training data. Future work may explore adaptive subspace construction and more general mechanisms for controlling directional flexibility with theoretical guarantees.

AI USE DISCLOSURE

In this work, we used generative AI tools to provide feedback on experiments, assist with the interpretation of results, and support code organization and implementation. We also used generative AI tools to assist with manuscript editing and polishing. We did not use generative AI tools for other applicable tasks requiring disclosure. All AI-assisted work was reviewed and verified by the authors. We take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

REFERENCES

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

George EP Box. Use and abuse of regression. *Technometrics*, 8(4):625–629, 1966.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613, 2014.

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. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pp. 178–178. IEEE, 2004.

Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3 (4):128–135, 1999.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Qiang Huang, Jizhe Zhang, Arman Sabbaghi, and Tirthankar Dasgupta. Optimal offline compensation of shape shrinkage for three-dimensional printing processes. *Iie transactions*, 47(5):431–441, 2015.

William G Hunter and John J Crowley. Hazardous substances, the environment and public health: a statistical overview. *Environmental health perspectives*, 32:241–254, 1979.

Hangzhan Jin, Tianwei Ni, Lu Li, Pierre-Luc Bacon, Mohammad Hamdaqa, and Doina Precup. Rotation-preserving supervised fine-tuning. *arXiv preprint arXiv:2605.10973*, 2026.

Brian L Joiner. Lurking variables: Some examples. *The American Statistician*, 35(4):227–233, 1981.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in neural information processing systems*, 34:1022–1035, 2021.

Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M Rush. Block pruning for faster transformers. *arXiv preprint arXiv:2109.04838*, 2021.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. In *International Conference on Learning Representations*.

Jaeho Lee, Sejun Park, Sangwoo Mo, Sungsoo Ahn, and Jinwoo Shin. Layer-adaptive sparsity for the magnitude-based pruning. *arXiv preprint arXiv:2010.07611*, 2020.

Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pp. 6028–6039. PMLR, 2020.

Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23638–23647. IEEE, 2024.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Hoang-Chau Luong and Lingwei Chen. Why lora fails to forget: Regularized low-rank adaptation against backdoors in language models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 34692–34705, 2026.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 67–82, 2018.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pp. 2408–2417. PMLR, 2015.

Hesham Mostafa and Xin Wang. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pp. 4646–4655. PMLR, 2019.

Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5715–5725, 2017.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008.

Aliakbar Panahi, Seyran Saeedi, and Tom Arodz. Shapeshifter: a parameter-efficient transformer using factorized reshaped matrices. *Advances in Neural Information Processing Systems*, 34: 1337–1350, 2021.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. *arXiv preprint arXiv:2007.07779*, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918*, 2020.

Arman Sabbaghi and Qiang Huang. Model transfer across additive manufacturing processes via mean effect equivalence of lurking variables. *The Annals of Applied Statistics*, 12(4):2409–2429, 2018.

Victor Sanh, Thomas Wolf, and Alexander Rush. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in neural information processing systems*, 33:20378–20389, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pp. 4548–4557. PMLR, 2018.

Reece Shuttleworth, Jacob Andreas, Antonio Torralba, and Pratyusha Sharma. Lora vs full fine-tuning: An illusion of equivalence. *arXiv preprint arXiv:2410.21228*, 2024.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Brady Steele. Subspace geometry governs catastrophic forgetting in low-rank adaptation. *arXiv preprint arXiv:2603.02224*, 2026.

Junjiao Tian, Zecheng He, Xiaoliang Dai, Chih-Yao Ma, Yen-Cheng Liu, and Zsolt Kira. Trainable projected gradient method for robust fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7836–7845, 2023a.

Junjiao Tian, Yen-Cheng Liu, James S Smith, and Zsolt Kira. Fast trainable projection for robust fine-tuning. *Advances in Neural Information Processing Systems*, 36:11374–11393, 2023b.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuan-Jing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 10658–10671, 2023.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7959–7971, 2022.

Yifeng Xiong and Xiaohui Xie. Oplora: Orthogonal projection lora prevents catastrophic forgetting during parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34088–34096, 2026.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*, 2021.

LI Xuhong, Yves Grandvalet, and Franck Davoine. Explicit inductive bias for transfer learning with convolutional networks. In *International conference on machine learning*, pp. 2825–2834. PMLR, 2018.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, et al. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 45040–45061, 2024.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021.

Maxime Zanella and Ismail Ben Ayed. Low-rank few-shot adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1593–1603, 2024.

Yaoyue Zheng, Yin Zhang, Joost van de Weijer, Gido M van de Ven, Shaoyi Du, Xuetao Zhang, and Zhiqiang Tian. Revisiting weight regularization for low-rank continual learning. *arXiv preprint arXiv:2602.17559*, 2026.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

A Related Work

Distance-based forgetting mitigation. A common strategy for mitigating forgetting is to restrict how far the fine-tuned model moves from the pre-trained or previously learned model. Elastic Weight Consolidation (EWC) is a representative regularization-based method: it penalizes changes to parameters that are important for previous tasks using a Fisher-weighted quadratic penalty (Kirkpatrick et al., 2017). Recent work further adapts this idea to low-rank fine-tuning, such as LoRA-EWC, which regularizes LoRA updates through Fisher-based importance estimates (Zheng et al., 2026). Another line of work controls parameter deviation through weight-space interpolation or projection. WiSE-FT interpolates the original pre-trained and fine-tuned weights to recover robustness under distribution shift (Wortsman et al., 2022), while FTP constrains deviation from the pre-trained model through learnable projection constraints during fine-tuning (Tian et al., 2023b). Although these approaches differ in implementation, they generally mitigate forgetting by regularizing parameter changes. In contrast, our work factorizes local forgetting risk into update magnitude and directional sensitivity and explicitly constrains both the admissible adaptation subspace and update magnitude. The subspace constraint restricts where adaptation-induced updates can occur, while update regularization limits the extent of parameter changes.

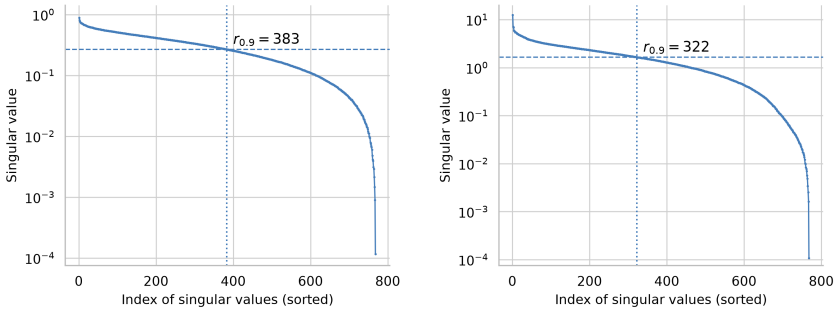
Subspace- and direction-aware adaptation. Recent work has increasingly studied forgetting from a subspace or spectral perspective. Shuttleworth et al. (2024) introduced *intruder dimensions* and showed that changes in leading left singular directions are strongly associated with forgetting. O-LoRA (Wang et al., 2023) and InfLoRA (Liang & Li, 2024) address sequential continual learning by placing different tasks or updates in interference-reducing subspaces. RoRA (Luong & Chen, 2026) and Subspace Geometry Governs Catastrophic Forgetting (Steele, 2026) further study how spectral strength, alignment, and subspace geometry relate to forgetting. Most closely related to our setting, OPLoRA (Xiong & Xie, 2026) protects selected leading singular structures of the pre-trained model by constraining adaptation relative to these directions. In contrast, we directly constrain the admissible adaptation subspace together with update magnitude under a unified forgetting-risk formulation. We use a broader high-energy spectral subspace of the pre-trained weights as a data-free, deterministic, and model-informed choice of the adaptation space.

Parameter-Efficient Fine-Tuning (PEFT). PEFT aims to adapt large pre-trained models by updating only a small subset of parameters or by introducing lightweight trainable modules. Existing PEFT methods differ in how they restrict the trainable components. Randomized approaches, such as Random and Mixout models (Lee et al.), select parameters for fine-tuning without using task-specific information. Rule-based approaches, including BitFit (Zaken et al., 2021), pruning-based methods (Han et al., 2015; Lee et al., 2020; Lagunas et al., 2021), adapters (Houlsby et al., 2019; Pfeiffer et al., 2020; Rücklé et al., 2020), and LoRA-style methods (Hu et al., 2022; Karimi Mahabadi et al., 2021; Panahi et al., 2021), impose predefined structures on the trainable parameters. Among them, LoRA has become a widely used baseline due to its simplicity, hardware efficiency, and strong empirical performance across modalities. Projection- or mask-based approaches, such as DiffPruning (Mallya et al., 2018; Sanh et al., 2020; Lagunas et al., 2021) and ChildPruning (Xu et al., 2021; Mostafa & Wang, 2019), use data-dependent criteria to determine which parameters or directions should be updated. Our RSP-LoRA and VeFA are parameter-efficient realizations within the PEFT setting. However, the main contribution of this work is not a new parameter-selection rule, but a constrained fine-tuning formulation that jointly controls the admissible adaptation subspace and update magnitude. Thus, although our implementations build on parameter-efficient adaptation, the proposed formulation is not limited to PEFT.

Domain Adaptation. Our work is related to domain adaptation (DA), since fine-tuning can be viewed as adapting a pre-trained model to a downstream distribution. However, our setting and objective differ from standard DA approaches (Zhu et al., 2017; Ganin et al., 2016; Liang et al., 2020; Motiian et al., 2017). DA methods typically assume access to both source and target data and learn domain-invariant representations by explicitly aligning the two distributions. In contrast, we do not assume access to the original pre-training data and focus on mitigating forgetting during downstream fine-tuning. Rather than explicitly aligning source and target distributions, we constrain the admissible adaptation subspace and update magnitude to regularize fine-tuning. Therefore, our work focuses on constrained fine-tuning for forgetting mitigation rather than conventional domain adaptation.

B Spectral Analysis of Pre-trained Weight Matrices

Although pre-trained weight matrices are generally full rank, their spectral energy can be concentrated in a subset of singular directions. As illustrated in Fig. 4, the selected weight matrices from CLIP ViT-B/16 and RoBERTa-base require only 383 and 322 of their 768 singular directions, respectively, to capture at least 90% of the total spectral energy, defined as the sum of squared singular values. Thus, approximately half or fewer of the singular directions account for most of the energy in these matrices. This observation motivates defining the admissible adaptation subspace using a cumulative energy threshold α, which controls its dimensionality based solely on the pre-trained weights.



(a) CLIP ViT-B/16 (b) RoBERTa-base

Figure 4: Spectral decay and effective rank of pre-trained Transformer weight matrices.

C Effect Equivalence Modeling of Lurking Variables

Effect equivalence modeling (EEM) provides a statistical motivation for the feature-adaptation realization of our subspace-constrained formulation. Its central idea is to represent the effects of unobserved factors through equivalent transformations of observed inputs (Huang et al., 2015; Sabbaghi & Huang, 2018).

Unobservable factors that induce distributional discrepancies can be described as lurking variables (Box, 1966; Joiner, 1981; Hunter & Crowley, 1979). As illustrated in Fig. 5, lurking variables $\boldsymbol{U}$ may alter the distributions of the observed input $\boldsymbol{X}$, response $\boldsymbol{y}$, or both. In our setting, such effects provide one interpretation of discrepancies between pre-training and downstream data.

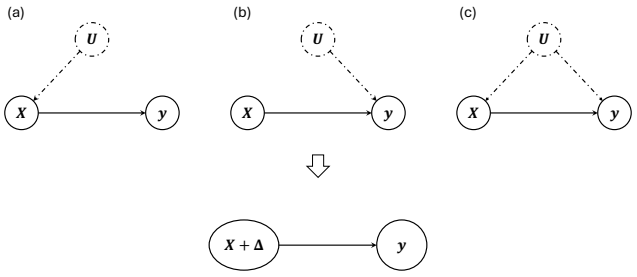


Figure 5: Three types of lurking-variable effects: (a) $\boldsymbol{U}$ alters the distribution of input $\boldsymbol{X}$, (b) alters the response $\boldsymbol{y}$, or (c) alters both.

Let $\boldsymbol{X} \in \mathbb{R}^d$ denote the observed variables and $\boldsymbol{U} \in \mathbb{R}^m$ the lurking variables. When the effects of $\boldsymbol{U}$ are negligible, the response is modeled as

$$\boldsymbol{y} = f(\boldsymbol{X}) + \boldsymbol{\epsilon}, \tag{5}$$

where $f : \mathbb{R}^d \to \mathbb{R}^k$ and $\mathbb{E}[\boldsymbol{\epsilon}] = \mathbf{0}$. When lurking variables exert non-negligible effects, we instead consider

$$\boldsymbol{y} = g(\boldsymbol{X}, \boldsymbol{U}) + \boldsymbol{\epsilon}, \tag{6}$$

where $g : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^k$ and $f(\boldsymbol{X}) = g(\boldsymbol{X}, \mathbf{0})$. EEM locally represents the response discrepancy induced by $\boldsymbol{U}$ through an equivalent transformation of $\boldsymbol{X}$.

Proposition C.1 (Local Effect Equivalence)**.** *Let $g : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}^k$ be continuously differentiable and define $f(\boldsymbol{X}) = g(\boldsymbol{X}, \mathbf{0})$. For a given $(\boldsymbol{X}, \boldsymbol{U})$, define the lurking-variable-induced response discrepancy as*

$$\boldsymbol{\delta}_U = g(\boldsymbol{X}, \boldsymbol{U}) - g(\boldsymbol{X}, \mathbf{0}).$$

If $\boldsymbol{\delta}_U \in \mathrm{Col}(\boldsymbol{J}_f(\boldsymbol{X}))$, where $\boldsymbol{J}_f(\boldsymbol{X}) = \partial f(\boldsymbol{X})/\partial \boldsymbol{X} \in \mathbb{R}^{k\times d}$, then there exists an equivalent feature perturbation $\boldsymbol{\Delta} \in \mathbb{R}^d$ satisfying

$$\boldsymbol{J}_f(\boldsymbol{X})\boldsymbol{\Delta} = \boldsymbol{\delta}_U.$$

For sufficiently small $\|\boldsymbol{\Delta}\|$,

$$g(\boldsymbol{X}, \boldsymbol{U}) = f(\boldsymbol{X}) + \boldsymbol{J}_f(\boldsymbol{X})\boldsymbol{\Delta} = f(\boldsymbol{X} + \boldsymbol{\Delta}) + o(\|\boldsymbol{\Delta}\|). \tag{7}$$

A canonical minimum-norm solution is $\boldsymbol{\Delta}^* = \boldsymbol{J}_f(\boldsymbol{X})^\dagger \boldsymbol{\delta}_U$, where $(\cdot)^\dagger$ denotes the Moore–Penrose pseudoinverse.

Proposition C.1 shows that when the response discrepancy induced by unobserved factors lies locally within the response variations attainable through the observed features and admits a sufficiently small equivalent perturbation, it can be represented to first order through a transformation of the observed features.

Connection to subspace-constrained feature adaptation. For a pre-trained linear map $f(\boldsymbol{x}) = \boldsymbol{W}_{\mathrm{pt}}\boldsymbol{x}$, EEM motivates representing downstream discrepancies through feature-induced response corrections. We retain the original pre-trained response and constrain only the adaptation-induced correction to the prescribed adaptation subspace. Specifically, with a learnable feature transformation $\boldsymbol{M}$, we define

$$\boldsymbol{h} = \boldsymbol{W}_{\mathrm{pt}}\boldsymbol{x} + \boldsymbol{W}_{\mathrm{pt}}^{(\alpha)}\boldsymbol{M}\boldsymbol{x},$$

which corresponds to the effective weight matrix $\boldsymbol{W}_{\mathrm{ft}} = \boldsymbol{W}_{\mathrm{pt}} + \boldsymbol{W}_{\mathrm{pt}}^{(\alpha)}\boldsymbol{M}$. Therefore,

$$\mathrm{Col}(\boldsymbol{W}_{\mathrm{ft}} - \boldsymbol{W}_{\mathrm{pt}}) = \mathrm{Col}(\boldsymbol{W}_{\mathrm{pt}}^{(\alpha)}\boldsymbol{M}) \subseteq \mathcal{S}_0^{(\alpha)}.$$

Thus, EEM motivates adaptation through feature-side corrections, while the adaptation-subspace constraint restricts the induced corrections to the prescribed pre-trained spectral subspace.

D Experimental Details

D.1 Datasets and Evaluation Protocols

We evaluate our methods across three model families and task domains; the datasets and forgetting evaluation protocols for each setting are detailed below.

1. **Image Classification** (CLIP across seven datasets (Radford et al., 2021; Zanella & Ben Ayed, 2024)). We follow the setting of previous work (Zanella & Ben Ayed, 2024; Zhou et al., 2022) and evaluate seven datasets spanning diverse visual domains: EuroSAT (Helber et al., 2019), Food101 (Bossard et al., 2014), Pets (Parkhi et al., 2012), Flowers (Nilsback & Zisserman, 2008), Caltech101 (Fei-Fei et al., 2004), DTD (Cimpoi et al., 2014), and UCF101 (Soomro et al., 2012). For each run, we fine-tune CLIP on one dataset as the in-domain task and treat the remaining six datasets as held-out tasks. Forgetting is measured by the average performance change on these held-out tasks relative to the original pre-trained model $\boldsymbol{W}_{\mathrm{pt}}$. Since CLIP already supports these tasks zero-shot, this directly measures degradation of pre-existing capabilities after fine-tuning.

2. **Natural Language Understanding** (RoBERTa on GLUE (Liu et al., 2019; Wang et al., 2018)). We fine-tune RoBERTa separately on six GLUE tasks: SST-2, MRPC, CoLA, QNLI, RTE, and STS-B. To approximate the RoBERTa pre-training distribution, we randomly sample from four pre-training corpora—CC-News, OpenWebText, Stories, and BookCorpus—and weight them proportionally to their contributions to the original RoBERTa pre-training mixture (Liu et al., 2019). Following Shuttleworth et al. (2024), we reconnect the fine-tuned RoBERTa backbone to the original pre-trained MLM head and measure forgetting by the change in masked-language-model (MLM) pseudo-loss relative to the original pre-trained model $\boldsymbol{W}_{\mathrm{pt}}$.

3. **Mathematical Reasoning** (Qwen2.5-7B (Yang et al., 2024)). We fine-tune Qwen2.5-7B on MetaMathQA (Yu et al., 2024) and evaluate downstream mathematical reasoning on MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021). We further evaluate retention of held-out capabilities on ARC-e, ARC-c (Clark et al., 2018), and SIQA (Sap et al., 2019) after fine-tuning.

D.2 Hyperparameter

D.2.1 Image Classification

For the CLIP image experiments, all LoRA-based methods, including LoRA, Intruder-Scaling-LoRA, LoRA-EWC, LoRA-WISE, LoRA-FTP, RSP-LoRA-EWC, and RSP-LoRA-FTP, use the same base training configuration as the original LoRA baseline, as summarized in Table 4. VeRA, VeFA, and VeFA-EWC use the method-specific learning rates reported in Table 5. All other hyperparameters follow the LoRA baseline unless otherwise specified.

For methods with additional hyperparameters, we select them using only in-domain validation performance and evaluate forgetting using the selected configurations. For LoRA-EWC, RSP-LoRA-EWC, and VeFA-EWC, we tune the Fisher-weighted regularization strength λ_{EWC}, with the selected values reported in Table 5. LoRA-FTP and RSP-LoRA-FTP use the same FTP hyperparameters. For RSP-LoRA variants, VeFA, and VeFA-EWC, we use an adaptation-subspace energy ratio of $\alpha = 0.90$ unless otherwise specified. For LoRA-WISE, we tune the interpolation coefficient α_{WISE} separately for each fine-tuning dataset using in-domain validation performance; the selected coefficients are reported in Table 6.

Table 4: Base hyperparameter settings for CLIP-LoRA image experiments. The same configuration is used for LoRA and all LoRA-based variants unless otherwise specified.

Hyperparameter	**Value**
Backbone	ViT-B/16
Shots	16
Learning rate	2×10^{-4}
Training steps per shot	500
Total optimizer steps	8,000
Batch size	32
LoRA rank r	1
LoRA scaling factor α_{LoRA}	1
Dropout rate	0.25
Adapted attention matrices	q, k, v

D.2.2 Natural Language Understanding

For the GLUE experiments, LoRA and its variants use the base training configurations summarized in Table 7. Unless otherwise specified, Intruder-Scaling-LoRA, LoRA-EWC, LoRA-WISE, LoRA-FTP, RSP-LoRA-EWC, RSP-LoRA-FTP, and LoRA$_{r=1}$ follow the same base LoRA configuration and introduce only their method-specific hyperparameters. For LoRA$_{r=1}$, we keep the same configuration as LoRA and only reduce the rank from $r = 8$ to $r = 1$. VeRA and VeFA-EWC use separate configurations, reported in Tables 8 and 9, respectively.

For methods with additional hyperparameters, we select them using only in-domain validation performance and evaluate forgetting using the selected configurations. For LoRA-EWC and RSP-LoRA-EWC, we use $\lambda_{\mathrm{EWC}} = 10^7$. LoRA-FTP and RSP-LoRA-FTP use the same FTP hyperparameters reported in Table 10. For LoRA-WISE, we tune the interpolation coefficient α_{WISE} for each GLUE task using in-domain validation performance; the selected coefficients are reported in Table 11.

D.2.3 Mathematical Reasoning

For the Qwen experiments, we fine-tune Qwen2.5-7B on MetaMathQA under a limited-data setting. We randomly sample 10,000 examples from a fixed pool of 100,000 training examples using a fixed

Table 5: Method-specific hyperparameters for CLIP image experiments. These hyperparameters are selected using in-domain validation performance only unless otherwise specified.

Method	Hyperparameter	Value
VeRA	Learning rate	5×10^{-3}
LoRA-EWC	λ_{EWC}	10^5
RSP-LoRA-EWC	λ_{EWC}	10^3
VeFA	Learning rate	6×10^{-4}
VeFA-EWC	Learning rate	10^{-3}
	λ_{EWC}	10^3
LoRA-FTP / RSP-LoRA-FTP	κ	1.0
	γ learning rate	10^{-2}
	γ initialization	10^{-8}
	β_1	0.9
	β_2	0.999
	ϵ_{opt}	10^{-8}

Table 6: Dataset-specific LoRA-WISE interpolation coefficients for CLIP image experiments.

Dataset	Caltech101	DTD	EuroSAT	Food101	Flowers	Pets	UCF101
$\boldsymbol{\alpha}_{\mathbf{WISE}}$	0.5	0.6	0.7	0.6	0.7	0.6	0.5

Table 7: Base LoRA hyperparameter settings for RoBERTa-Base on GLUE benchmark tasks. LoRA variants are built on this configuration unless otherwise specified.

Model	Hyper-Parameters	SST-2	MRPC	CoLA	QNLI	RTE	STS-B
Base	Optimizer	AdamW					
	Warmup Ratio	0.06					
	LR Schedule	Linear					
	LoRA Rank r	8					
	LoRA Scaling Factor	16					
	Weight Decay	0.1					
	Batch Size	64					
	Epochs	60	30	80	25	80	40
	Learning Rate	5E-4	4E-4	4E-4	4E-4	5E-4	4E-4
	Max Seq. Len.	512					

Table 8: VeRA hyperparameter settings for RoBERTa-Base on GLUE benchmark tasks.

Model	Hyper-Parameters	SST-2	MRPC	CoLA	QNLI	RTE	STS-B
	Optimizer	AdamW					
	Warmup Ratio	0.06					
	LR Schedule	Linear					
	VeRA Dropout	0.1					
	d_{init}	0.1					
	Weight Decay	0.1					
Base	Epochs	60	30	80	25	160	80
	VeRA Rank r	1024					
	Learning Rate (VeRA)	4E-3	4E-3	1E-2	4E-3	1E-2	1E-2
	Learning Rate (Head)	4E-3	1E-2	1E-2	1E-2	4E-3	1E-2
	Max Seq. Len.	512					
	Batch Size	64					

Table 9: VeFA-EWC hyperparameters for RoBERTa-Base on GLUE.

Hyper-Parameters	SST-2	MRPC	CoLA	QNLI	RTE	STS-B
Optimizer	AdamW					
Warmup Ratio	0.06					
LR Schedule	Linear					
EWC Coefficient (λ_{EWC})	1×10^5					
Epochs	60	30	80	25	80	50
Learning Rate	8E-3	4E-3	8E-3	8E-3	4E-3	4E-3
Weight Decay	0.05					
Max Seq. Len.	512					
Batch Size	64					

Table 10: Method-specific hyperparameters for GLUE experiments. These hyperparameters are selected using in-domain validation performance only.

Method	Hyperparameter	Value
LoRA-EWC / RSP-LoRA-EWC	λ_{EWC}	10^7
	Weight Decay	0
LoRA-FTP / RSP-LoRA-FTP	κ	1.0
	γ learning rate	10^{-2}
	γ initialization	10^{-8}
	β_1	0.9
	β_2	0.999
	ϵ_{opt}	10^{-8}

Table 11: Task-specific LoRA-WISE interpolation coefficients for RoBERTa-Base on GLUE.

	SST-2	QNLI	MRPC	STS-B	CoLA	RTE
α_{WISE}	0.5	0.8	0.9	0.8	0.7	0.7

data seed and use the same subset across all methods and training seeds. All methods start from the same original pre-trained checkpoint.

LoRA, OPLoRA, LoRA-FTP, and RSP-LoRA-FTP share the same training configuration, including the adapter rank, scaling factor, learning rate, target modules, and training budget. The shared hyperparameters are provided in Appendix Table 12. OPLoRA additionally specifies the number of protected singular directions. LoRA-FTP and RSP-LoRA-FTP use identical FTP hyperparameters, while RSP-LoRA-FTP additionally constrains the LoRA update to the prescribed adaptation subspace. This comparison assesses the additional effect of the adaptation-subspace constraint under the same FTP regularization. Method-specific settings are reported in Appendix Table 13.

We also evaluate Intruder-Scaling-LoRA by applying post-hoc intruder suppression to the trained LoRA model using the original pre-trained checkpoint as the reference. For each target matrix, we scale the singular value of the highest-ranking intruder, if present, without additional training. The detection threshold and retention factor are provided in Appendix Table 14.

Table 12: Shared hyperparameter settings for Qwen2.5-7B math fine-tuning. LoRA, OPLoRA, LoRA-FTP, and RSP-LoRA-FTP use the same configuration unless otherwise specified.

Hyperparameter	**Value**
Model	Qwen2.5-7B
Initialization	Original pre-trained checkpoint
Training Dataset	MetaMathQA
Training Pool Size	100,000
Sampling Fraction	0.1
Sampled Training Examples	10,000
Data Sampling Seed	42
Training Seeds	0, 1, 2
Optimizer	AdamW
Learning Rate	3×10^{-4}
LR Schedule	Cosine
Warmup Ratio	0.03
Epochs	1
Microbatch Size	4
Gradient Accumulation Steps	8
Effective Batch Size	32
Maximum Sequence Length	1024
Weight Decay	0
Maximum Gradient Norm	1.0
LoRA Rank r	64
LoRA Scaling Factor	64
Adapter Dropout	0.05
Target Modules	`q_proj`, `v_proj`, `o_proj`, `up_proj`, `down_proj`
Base / Adapter Precision	BF16 / FP32
Gradient Checkpointing	Enabled
Training Prompt	Alpaca
Loss	Response tokens only

D.3 Compute Resources

Unless otherwise specified, all experiments are conducted on a single NVIDIA A100 GPU. For CLIP image experiments, running one method with one seed over the seven 16-shot datasets takes approximately 1.5 GPU-hours. For GLUE experiments, running one method with one seed over the six GLUE tasks takes approximately 4 GPU-hours for RoBERTa-Base. These estimates refer to the experiments reported in the paper.

Table 13: Method-specific settings for Qwen2.5-7B math fine-tuning. LoRA-FTP and RSP-LoRA-FTP share the same FTP hyperparameters; RSP-LoRA-FTP additionally applies the adaptation-subspace constraint.

Method	**Hyperparameter**	**Value**
LoRA	Additional Constraints	None
OPLoRA	Protected Singular Directions k	128
LoRA-FTP / RSP-LoRA-FTP	γ Initialization	5.0
	γ Learning Rate	10^{-2}
	κ	0.05
	β_1	0.9
	β_2	0.999
	ϵ_{opt}	10^{-8}
RSP-LoRA-FTP	Additional Constraint	Adaptation-subspace projection

Table 14: Post-hoc intruder suppression settings for trained LoRA models. Suppression requires no additional fine-tuning.

Setting	**Value**
Reference Model	Original pre-trained checkpoint
Intruder Threshold ϵ_{intr}	0.5
Singular Value Retention Factor λ	0.7
Suppression Selection	Highest-ranking intruder per target matrix
Target Modules	Same as LoRA
Additional Training	None

E Additional Results

E.1 Effectiveness on the Visual Classification Benchmark

Table 15 reports the mean 16-shot in-domain fine-tuning performance over three random seeds on seven image classification datasets. We compare standard LoRA with representative post-hoc, magnitude-control, and direction-control baselines, including Intruder-Scaling-LoRA, LoRA-WISE, LoRA-EWC, LoRA-FTP, and VeRA, together with our RSP-LoRA and VeFA variants. All LoRA-based methods use rank $r = 1$. Overall, fine-tuning substantially improves over the zero-shot CLIP baseline, particularly on Flowers, EuroSAT, DTD, and UCF101. More importantly, the compared methods achieve broadly comparable in-domain performance, with average accuracies ranging from 87.4 to 89.2. RSP-LoRA-FTP, RSP-LoRA-EWC, and RSP-LoRA-Norm achieve average accuracies of 88.4, 88.4, and 88.5, respectively, while VeFA and VeFA-EWC achieve 89.2 and 88.2. These results show that the proposed forgetting-mitigation strategies maintain competitive downstream adaptation performance, providing a comparable basis for evaluating forgetting.

E.2 Effectiveness on the GLUE Benchmark

Table 16 reports the in-domain performance on six GLUE tasks using RoBERTa-Base. Each entry reports the median over five random seeds, with uncertainty shown as half of the range. Overall, all methods achieve effective downstream adaptation, with average scores ranging from 81.6 to 84.2. Standard LoRA achieves the highest average score of 84.2, while RSP-LoRA-EWC and VeFA-EWC achieve 82.9 and 81.7, respectively. Together with the forgetting results in Table 2, these results characterize the trade-off between downstream adaptation and forgetting mitigation.

Table 15: In-domain fine-tuning accuracy on seven image classification datasets using the CLIP ViT-B/16 visual backbone. All LoRA-based methods use rank $r = 1$.

Shots	Method	Caltech101	Food101	Pets	Flowers	EuroSAT	DTD	UCF101	Avg
0	CLIP	93.2	85.2	88.2	67.3	42.3	44.3	65.0	69.4
16	LoRA	96.2	84.1	91.4	97.8	91.5	70.6	85.5	88.2
	Intruder-Scaling-LoRA	96.0	84.2	91.6	97.5	91.4	71.0	85.1	88.1
	LoRA-WISE	96.1	85.6	94.0	96.1	91.5	68.7	84.4	88.1
	LoRA-EWC	95.5	86.5	92.1	97.5	89.7	69.1	84.3	87.8
	LoRA-FTP	96.1	85.2	92.9	97.5	91.2	69.3	85.5	88.2
	VeRA	96.4	86.6	93.6	91.6	91.7	70.9	80.7	87.4
	RSP-LoRA	96.2	85.8	93.1	97.2	91.3	69.7	85.1	88.3
	RSP-LoRA-FTP	96.4	84.5	91.7	97.8	90.2	71.5	86.4	88.4
	RSP-LoRA-EWC	96.4	85.0	92.1	97.2	91.3	70.5	86.2	88.4
	RSP-LoRA-Norm	96.4	84.8	93.1	97.5	90.1	71.6	86.2	88.5
	VeFA	96.1	87.3	94.7	97.0	91.2	71.7	85.7	89.2
	VeFA-EWC	94.6	86.4	93.5	97.2	90.9	70.2	84.3	88.2

Table 16: In-domain performance on GLUE using RoBERTa-Base with different fine-tuning methods. We report Matthews correlation for CoLA, Pearson correlation for STS-B, and accuracy for all other tasks; higher is better.

Method	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	AVG
LoRA	94.6±0.2	88.7±1.0	62.5±1.5	92.8±0.2	75.8±0.7	90.8±0.2	84.2
LoRA$_{r=1}$	93.9±0.2	87.7±0.6	62.6±0.8	92.2±0.2	74.3±0.5	90.2±0.4	83.5
Intruder-Scaling-LoRA	94.6±0.2	88.2±0.7	55.2±1.2	92.7±0.2	74.4±0.5	90.7±0.4	82.6
LoRA-Wise	92.7±0.2	88.5±1.2	51.6±1.6	92.7±0.4	75.8±1.6	89.6±0.4	81.8
LoRA-EWC	94.8±0.5	88.2±0.9	60.6±1.7	93.2±0.2	74.4±0.5	89.0±0.1	83.4
LoRA-FTP	94.7±0.4	86.8±1.0	54.4±1.0	92.5±0.2	72.2±1.1	89.2±0.2	81.6
VeRA	93.6±0.2	88.2±0.6	62.5±1.6	90.4±0.3	77.6±0.7	90.6±0.2	83.8
RSP-LoRA	95.0±0.3	89.0 ±0.6	59.1 ±1.3	92.8 ±0.3	74.4 ±0.6	91.1 ±0.2	83.6
RSP-LoRA-EWC	95.0±0.3	88.0±0.8	60.4±1.2	92.5±0.1	72.2±0.6	89.3±0.1	82.9
VeFA-EWC	93.9±0.1	87.7±0.8	56.1±1.0	89.9±0.2	74.2±0.5	88.2±0.1	81.7

E.3 Analysis of Forgetting Factors

To examine the forgetting-risk factorization, we conduct an exploratory association analysis of update magnitude M and directional sensitivity $\bar{D}$ with observed forgetting across methods and tasks. Directional sensitivity shows a positive association with forgetting (Spearman $\rho = 0.586$, $p < 10^{-4}$), whereas update magnitude alone shows no significant association ($\rho = -0.127$, $p = 0.389$); the estimated risk $\widehat{R}_{\mathrm{F}} = \frac{1}{2} M \bar{D}$ is also positively correlated with forgetting ($\rho = 0.554$, $p < 10^{-4}$). Similar or larger update magnitudes can correspond to substantially different forgetting, indicating that magnitude alone is insufficient to characterize forgetting. As summarized in Table 17, RSP-LoRA-EWC reduces both M and $\bar{D}$ relative to LoRA-EWC, whereas VeFA-EWC achieves the lowest $\bar{D}$ despite the largest M. These associations are consistent with a complementary role of adaptation geometry beyond update magnitude alone, but should not be interpreted as establishing a causal relationship.

Table 17: Mechanism analysis averaged over six GLUE tasks.

Method	M	$\bar{D}$	$\widehat{R}_F$	F	Method	M	$\bar{D}$	$\widehat{R}_F$	F
LoRA	1723	3.22×10^{-5}	.0167	.469	RSP-LoRA-EWC	**1236**	9.52×10^{-6}	**.0049**	**.050**
LoRA-EWC	1554	1.20×10^{-5}	.0093	.145	VeFA-EWC	4379	4.97×10^{-6}	.0119	.074

E.4 Controlled Comparison at Rank One

We compare LoRA$_{r=1}$, RSP-LoRA$_{r=1}$, and post-hoc Intruder Scaling to test whether the subspace constraint remains beneficial when the LoRA rank is already one. RSP-LoRA and LoRA have the same parameter count, with final layer-wise update magnitudes matched to the LoRA checkpoint. For RSP-LoRA, we apply the subspace constraint for the first 5% of training, introduce magnitude matching over the next 5%, and thereafter match the final LoRA update norms. Intruder Scaling retains its original post-hoc operation without magnitude matching, since Fig. 1 shows that magnitude-matched intruder scaling substantially reduces retention.

As shown in Fig. 3, RSP-LoRA improves average retention over both baselines with limited changes in in-domain performance. These results isolate the retention benefit of subspace restriction beyond parameter count and update magnitude, while highlighting the limitation of post-hoc correction compared with proactive adaptation-space control.

F Performance and Resource Trade-offs of Subspace Projection

We examine how the retained spectral energy threshold α affects the performance, computation, and storage costs of RSP-LoRA on Qwen2.5-7B. All configurations preserve the original pretrained weights and use LoRA rank 64, LoRA scaling 64, learning rate 3×10^{-4}, and one epoch on the same fixed 10,000 training examples. Here, α denotes the subspace energy threshold, not the LoRA scaling parameter. No FTP magnitude control is used.

Implementation. We compute FP32 exact economy-size SVD for the query, value, and output attention projections and the up and down MLP projections in each of the 28 transformer layers: 140 matrices in total. For $W_{\mathrm{pt}} = U\Sigma V^\top$, we retain

$$r_\alpha = \min\left\{ k : \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{\min(m,n)} \sigma_i^2} \geq \alpha \right\}, \qquad U_{r_\alpha} = U[:, 1 : r_\alpha]. \tag{8}$$

The implementation uses `torch.linalg.svd` with `full_matrices=False` and the CUDA `gesvd` driver. The decomposition determines the projection basis only; W_{pt} is neither truncated nor reconstructed.

We store the fixed FP32 basis U_{r_α}, not the dense projector $U_{r_\alpha}U_{r_\alpha}^\top$. After each optimizer step, we apply $B \leftarrow U_{r_\alpha}(U_{r_\alpha}^\top B)$. Thus, B is explicitly projected rather than parameterized as $B = U_{r_\alpha}C$. The forward pass retains the standard LoRA form, and projection occurs once per optimizer step, not per gradient-accumulation microbatch.

Performance. Table 18 shows that every reported energy threshold achieves a higher Overall score than the LoRA baseline, although individual benchmarks do not improve uniformly. The strongest Overall and held-out scores in this sweep occur around $\alpha = 0.3$–0.4, while increasing retained energy does not monotonically improve performance.

Measurement protocol. All cost measurements use seed 0 on one NVIDIA RTX PRO 6000 Blackwell Server Edition GPU, with PyTorch 2.13.0, CUDA 13.0, and Transformers 5.17.0. Base-model weights use BF16, while LoRA adapters and the stored subspace bases U_{r_α} use FP32. We use microbatch size 4, gradient accumulation of 8, and gradient checkpointing, yielding 313 optimizer steps. CUDA is synchronized at phase boundaries. Initialization includes SVD, rank selection, and adapter construction, but excludes base loading. Training time excludes initialization, saving, and evaluation. GPU peaks are measured using PyTorch's allocated and reserved memory counters; CPU process-tree RSS is sampled every 0.2 seconds. These are single-run measurements with profiling overhead, not repeated timing estimates.

Preprocessing and training costs. RSP-LoRA initialization takes 86–90 seconds, including approximately 85–87 seconds of SVD, compared with 0.91 seconds for LoRA. Initialization time changes little with α because exact SVD is computed before rank selection. During this phase, the model remains on CPU and matrices are processed sequentially on GPU: peak PyTorch-allocated GPU memory is approximately 1.78 GiB, while sampled CPU RSS increases from 11.09 to 18.09 GiB across the sweep. SVD factors are not cached across runs.

Table 18: Performance under different retained-energy thresholds. Scores are percentages; held-out tasks use unnormalized accuracy. Unless otherwise specified, results are from a single run with seed 0. †Results at $\alpha = 0.9$ are averaged over multiple runs.

Method	MATH	GSM8K	ARC-e	ARC-c	SIQA	In-domain	Held-out	Overall
LoRA	43.22	84.15	80.60	50.60	50.87	63.69	60.69	61.89
RSP-LoRA ($\alpha = 0.1$)	42.90	85.29	79.71	50.26	53.48	64.10	61.15	62.33
RSP-LoRA ($\alpha = 0.2$)	42.26	84.38	80.56	49.74	53.28	63.32	61.19	62.04
RSP-LoRA ($\alpha = 0.3$)	42.38	84.76	81.65	51.37	53.22	63.57	62.08	62.68
RSP-LoRA ($\alpha = 0.4$)	42.56	84.38	81.94	51.28	53.17	63.47	62.13	62.67
RSP-LoRA ($\alpha = 0.5$)	42.68	84.08	81.48	51.45	52.66	63.38	61.86	62.47
RSP-LoRA ($\alpha = 0.6$)	42.72	84.00	81.52	51.88	51.89	63.36	61.76	62.40
RSP-LoRA ($\alpha = 0.7$)	42.90	84.76	81.40	50.60	51.13	63.83	61.04	62.16
RSP-LoRA ($\alpha = 0.8$)	43.22	85.14	81.06	50.85	50.97	64.18	60.96	62.25
RSP-LoRA ($\alpha = 0.9$)†	42.96	85.05	81.14	50.54	51.05	64.01	60.91	62.15

Table 19: Resource costs for one profiled run per configuration. Mean rank averages the 140 target matrices. Basis storage uses actual FP32 tensor sizes. Allocated and reserved memory are training-phase peaks; Δ allocated is relative to LoRA. Initialization and training times are separate. GiB denotes 2^{30} bytes.

Method	Mean r_α	Basis GiB	Allocated GiB	Δ allocated GiB	Reserved GiB	Init. s	Train min
LoRA	–	0.00	23.78	0.00	53.05	0.91	14.23
RSP-LoRA ($\alpha = 0.1$)	70.05	0.33	24.12	0.34	53.24	89.06	15.26
RSP-LoRA ($\alpha = 0.2$)	169.39	0.79	24.59	0.81	53.60	85.91	15.31
RSP-LoRA ($\alpha = 0.3$)	288.07	1.33	25.12	1.34	54.26	86.59	15.31
RSP-LoRA ($\alpha = 0.4$)	426.56	1.95	25.74	1.96	54.89	86.91	15.31
RSP-LoRA ($\alpha = 0.5$)	587.86	2.66	26.44	2.66	55.60	87.32	15.33
RSP-LoRA ($\alpha = 0.6$)	777.96	3.48	27.28	3.50	56.47	87.74	15.34
RSP-LoRA ($\alpha = 0.7$)	1007.56	4.46	28.25	4.47	57.41	88.23	15.33
RSP-LoRA ($\alpha = 0.8$)	1297.79	5.66	29.46	5.68	58.70	89.12	15.36
RSP-LoRA ($\alpha = 0.9$)	1703.98	7.28	31.08	7.30	60.30	89.87	15.35

Table 19 reports the training trade-off. The mean retained rank increases from 70.05 to 1703.98, and FP32 basis storage increases from 0.33 to 7.28 GiB. Although the bases have no gradients or optimizer states, they remain GPU-resident during training. Their storage is $4 \sum_\ell m_\ell r_{\alpha,\ell}$ bytes. Peak allocated training memory consequently increases from 24.12 to 31.08 GiB, compared with 23.78 GiB for LoRA. Reserved memory is larger because it includes the allocator cache; neither metric represents total device memory usage.

The two projection multiplications require approximately $4mr_\alpha r$ FLOPs per matrix per optimizer step. CUDA-event timing around the projection hook totals 1.01–4.94 seconds over the full training run, increasing with α. This time is already included in training wall time. RSP-LoRA training takes 15.26–15.36 minutes versus 14.23 minutes for LoRA, an increase of approximately 7–8%, excluding initialization. This wall-time difference reflects the complete implementations and cannot be attributed solely to projection.

Storage, inference, and implications. All configurations retain 113,770,496 trainable parameters and save approximately 455 MB of FP32 adapter weights. Compact checkpoints store the effective LoRA factors but omit U_{r_α}, so their size does not increase with α. At inference, $W_{\text{merged}} = W_{\text{pt}} + s_{\text{LoRA}} BA$ eliminates the adapter and projection operations; the merged model has the original architecture and requires no subspace basis.

RSP-LoRA is therefore parameter-efficient and mergeable, but its training-memory overhead is not negligible at large retained ranks. In this experiment, $\alpha = 0.3$ improves Overall from 61.89 to 62.68 with approximately 1.34 GiB additional peak allocated memory, whereas $\alpha = 0.9$ requires

approximately 7.30 GiB additional memory. The threshold provides a performance–resource trade-off rather than a uniformly beneficial increase in capacity. This sweep does not, by itself, establish an advantage over rank-matched random subspaces.