FISHER-GUIDED PROGRESSIVE PARAMETER SELECTION FOR ADAPTIVE FINE-TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient fine-tuning often selects trainable parameters before adaptation using architectural heuristics, without accounting for their varying importance during training. We introduce **FisherAdapTune**, which progressively selects parameter groups based on temporal drift in their Fisher information. Under a local Gaussian approximation, we bound the divergence between the fine-tuned posterior and pretrained prior by accumulated Fisher-weighted update costs, motivating curvature-aware selection. FisherAdapTune measures Jensen-Shannon distance between successive Fisher-value distributions and uses an adaptive threshold to freeze stabilized groups. Across VTAB-1k classification tasks, it achieves the highest macro Top-1 accuracy among the compared methods with a smaller average trainable set than full fine-tuning. Across four segmentation backbones, it maintains competitive in-distribution performance and improves zero-shot transfer in several settings. Its selections reveal architecture-dependent patterns where input, output, and normalization parameters can remain trainable as attention and MLP groups freeze at different rates. The results support Fisher structural drift as a task-dependent signal for allocating updates during adaptation.

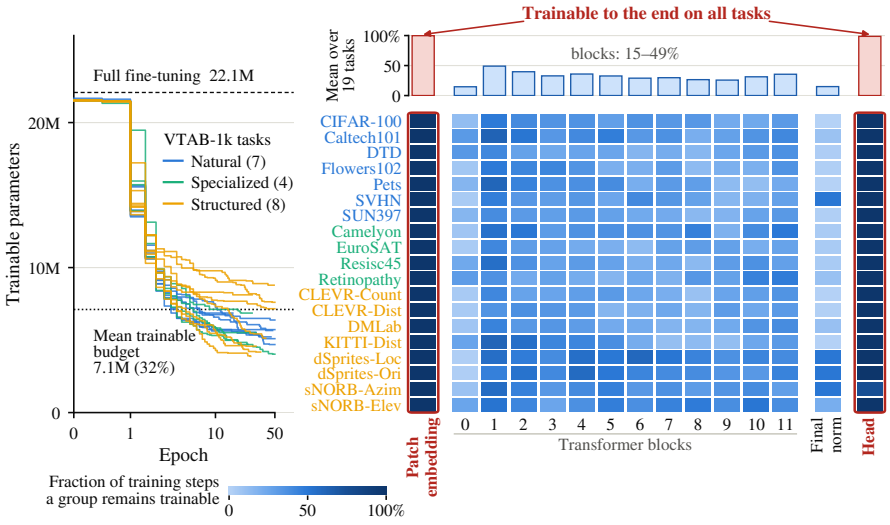


Figure 1: **FisherAdapTune uses variations in Fisher structure to progressively select parameters for each task and architecture.** Across VTAB-1k tasks with DINOv2 ViT-S/14, input and output layers remain trainable the longest, consistent with adaptation to input distribution shifts and task-specific outputs. See Sections 3 and Appendix F.2 for further discussion.

1 INTRODUCTION

Adapting pretrained deep neural networks to downstream tasks is a central problem in modern machine learning. While large-scale pretraining enables models to learn rich and transferable representations, leveraging these representations under limited data, distribution shifts, and computational constraints remains challenging (Zhuang et al., 2021; Kumar et al., 2021). In practice, downstream environments are dynamic where data distributions evolve, tasks change, and models must be updated frequently. This makes efficient and robust adaptation essential for scalable deployment, especially

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.

when retraining from scratch is infeasible. Full fine-tuning, where all parameters are updated for the target task, is computationally expensive and memory-intensive, especially for large models, and can lead to overfitting or degradation in generalization under low-data regimes (Zhang et al., 2025). Moreover, recent studies suggest that classical scaling laws do not directly transfer to fine-tuning settings, with performance depending on complex interactions between model size, pretraining, and data regime (Tay et al., 2021; Lin et al., 2024; Park and Kim, 2021). These observations highlight the need for principled adaptation strategies beyond naïve full fine-tuning.

To address these challenges, parameter-efficient fine-tuning reduces the trainable parameter set while seeking to preserve performance. Existing approaches include low-rank reparameterization, such as LoRA (Wang et al., 2024; He et al., 2023; Zhang et al., 2023), learnable prompts (Bahng et al., 2022; Huang et al., 2023; Jia et al., 2022), lightweight adapters (Houlsby et al., 2019; Chen et al., 2022a;b; Jie et al., 2024), and selective tuning of predefined subsets, such as biases or normalization layers (Zaken et al., 2022; Zhao et al., 2023). As summarized in Table 1, these methods can achieve competitive performance with fewer trainable parameters. Yet many rely on fixed architectural choices that cannot account for how parameter relevance evolves during training. Although data-driven selection has been explored (Zhao et al., 2020), a principled framework for allocating trainable capacity to task-relevant parameters throughout fine-tuning remains an open challenge.

Table 1: Comparison of PEFT categories. [1]Parameter Efficiency, [2]Task-Aware Configuration, [3]Architectural Modifications, [4]Theory-Grounded, [5]Generalization, [6]Inference Overhead.

Method	Param. Eff.[1]	Task-Aware[2]	Arch. Mod.[3]	Theory.[4]	Gen.[5]	Inf. OH[6]
Full Fine-Tuning	✗	✗	✗	✗	✗	✗
Reparameterization (Hu et al., 2021)	✓	✗	✗	✓	✗	✗
Selective (Zaken et al., 2022)	✓	✗	✗	✗	✗	✗
Prompt Tuning (Bahng et al., 2022)	✓	✗	✓	✗	✗	✓
Adapter-based (Chen et al., 2022a)	✓	✗	✓	✗	✗	✓
Ours (FisherAdapTune)	✓	✓	✗	✓	✓	✗

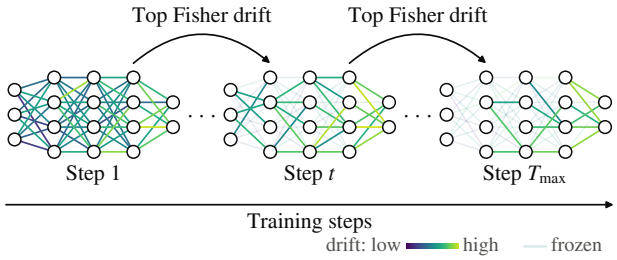


Figure 2: **FisherAdapTune** periodically measures Fisher information reorientation via Jensen-Shannon distance between consecutive Fisher distributions to identify stabilized parameter groups. Parameter groups with high Fisher drift remain tunable, while stabilized groups are frozen.

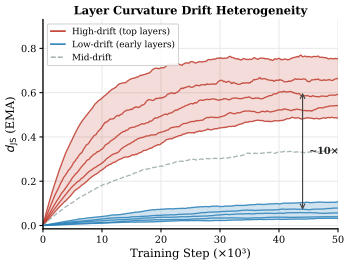


Figure 3: Layer-wise Fisher drift during full fine-tuning is uneven, motivating FisherAdapTune's progressive freezing of stabilized layers.

In this work, we propose **FisherAdapTune**, a Fisher-guided Adaptive Fine-Tuning framework for dynamic parameter selection during fine-tuning (Figure 2). A PAC-Bayesian analysis relates divergence from the pretrained prior to accumulated Fisher-weighted updates, motivating curvature-aware parameter selection during fine-tuning. Building upon this, our approach is grounded in the temporal evolution of the Fisher Information Matrix, which captures local curvature and parameter sensitivity. Instead of relying on static heuristics, we measure the *Jensen-Shannon (JS) distance* between successive Fisher distributions to track how parameter groups evolve during training (Figure 3). This provides a data-driven signal for tracking how parameter groups evolve: groups with persistent Fisher drift remain candidates for continued updating, whereas groups with low drift become candidates for freezing. This yields a dynamic selective fine-tuning strategy that continuously adjusts the trainable parameter set throughout optimization. By leveraging Fisher *structural drift*, FisherAdapTune identifies task-relevant parameters in a principled manner and avoids unnecessary updates. We validate FisherAdapTune on VTAB-1k classification tasks and a domain-specific dense prediction task, demonstrating that it achieves competitive or superior performance compared to existing PEFT baselines. Furthermore, our approach improves robustness under distribution shifts, highlighting the importance of adaptive, task-aware parameter selection for generalization.

Our **main contributions** are:

- We derive a PAC-Bayesian bound that relates divergence from the pretrained prior to accumulated Fisher-weighted updates, providing a theoretical motivation for curvature-aware selection.
- We develop a task-dependent selection criterion that measures temporal changes in group-wise Fisher-value distributions using Jensen-Shannon distance.
- We introduce **FisherAdapTune**, an adaptive, model-agnostic fine-tuning framework that progressively freezes groups with stabilized Fisher drift, discovering architecture-dependent trainable sets without a prescribed layer-freezing order.
- Evaluate FisherAdapTune on VTAB-1k classification tasks and a downstream segmentation task across four backbones, achieving the highest macro classification accuracy among the evaluated methods with a reduced trainable budget, alongside competitive segmentation and improved zero-shot transfer in several settings.

2 Related Work

Parameter-Efficient Fine-Tuning aims to adapt pretrained models with reduced computational and memory cost while maintaining performance (Zhang et al., 2025; Han et al., 2024). Selective fine-tuning methods, inspired in part by the Lottery Ticket Hypothesis (Frankle and Carbin, 2019), update only a subset of parameters to mitigate overfitting and improve efficiency (Aghajanyan et al., 2020). Existing approaches identify important parameters heuristically using magnitude of updates (e.g., LT-SFT (Ansell et al., 2023)), architectural choices (e.g., LayerNorm tuning (Zhao et al., 2023)), or probabilistic criteria (e.g., BayesTune (Kim and Hospedales, 2023), while others rely on learnable masks (Masking (Zhao et al., 2020), Diff-Pruning (Guo et al., 2021) or gradient dynamics (AutoFreeze (Liu et al., 2021)) to guide selection. Despite promising empirical results, these methods largely depend on heuristic signals and lack a principled, theoretically grounded mechanism for identifying task-relevant parameters. On the other hand, **Fisher-guided methods** use the Fisher Information Matrix to estimate parameter sensitivity (Kirkpatrick et al., 2017; Mi et al., 2022). Child-Tuning (Xu et al., 2021) and FISH-Mask (Sung et al., 2021) use diagonal Fisher estimates to select sparse parameter subsets prior to fine-tuning. More recently, FisherTune (Zhao et al., 2025) uses a domain-related Fisher formulation to identify parameters sensitive to task and domain shifts, with the goal of improving cross-domain generalization in semantic segmentation. Despite these advances, principled allocation of trainable capacity throughout fine-tuning remains an open challenge.

3 Methodology

Effective selective fine-tuning requires identifying both *where adaptation remains necessary* and *when parameter groups can be frozen*. We introduce **FisherAdapTune**, a selective fine-tuning framework, which uses the temporal evolution of Fisher information to guide these decisions. Section 3.1 establishes the theoretical foundation by connecting Fisher-weighted updates to a PAC-Bayesian generalization bound. Section 3.2 decomposes this cost across layers and motivates Fisher drift as an adaptation signal. Finally, Section 3.3 introduces a Jensen-Shannon distance criterion for measuring this drift, presents empirical evidence of its heterogeneity across layers and parameter groups, and develops the progressive freezing algorithm, with an analysis of the resulting parameter selections.

3.1 Role of Fisher Information in Generalization and Adaptation

We revisit Probably Approximately Correct (PAC)-Bayesian learning theory (McAllester, 1999a;b; Dziugaite and Roy, 2017; Zhou et al., 2018; Lotfi et al., 2022) to derive a principled basis for fine-tuning. Let P denote a prior distribution over model parameters, instantiated by the model's pretrained weights, and let Q_T denote the posterior after T steps of fine-tuning. With probability at least $1-\delta$ over the training sample, the theorem bounds the generalization error as follows:

$$\mathbb{E}_{Q_T}[\mathcal{L}_{\text{out}}] \;\le\; \mathbb{E}_{Q_T}[\mathcal{L}_{\text{in}}] \;+\; \sqrt{\frac{D_{\text{KL}}(Q_T\|P) + \log(N/\delta)}{2N}}, \tag{1}$$

where $\mathcal{L}_{\text{in}}$ is the empirical risk on N training samples drawn i.i.d. from a distribution $\mathcal{D}$, and $\mathcal{L}_{\text{out}}$ is the population risk under the same $\mathcal{D}$. The bound is governed by $D_{\text{KL}}(Q_T\|P)$ where excessive deviation from the pretrained prior loosens the guarantee (Lotfi et al., 2022).

To understand how fine-tuning updates affect this divergence, we consider how a small change in parameters shifts the model's output distribution. Let $p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})$ denote the predictive distribution over $\mathbf{y}$ produced by a model with parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{|\boldsymbol{\theta}|}$ given input $\mathbf{x}$. A natural measure of the effect of a perturbation $\boldsymbol{\delta} \in \mathbb{R}^{|\boldsymbol{\theta}|}$ on the model's predictions is $D_{\mathrm{KL}}(p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x}) \| p_{\boldsymbol{\theta}+\boldsymbol{\delta}}(\mathbf{y} \mid \mathbf{x}))$. It can be shown (Dangel et al., 2025; Amari, 2016; Soen and Sun, 2024) that as $\|\boldsymbol{\delta}\| \to 0$, a second-order expansion averaged over the input distribution yields:

$$\mathbb{E}_{\mathbf{x}}[D_{\mathrm{KL}}(p_{\boldsymbol{\theta}+\boldsymbol{\delta}} \parallel p_{\boldsymbol{\theta}})] = \tfrac{1}{2}\, \boldsymbol{\delta}^\top \mathbf{F}_{\boldsymbol{\theta}} \boldsymbol{\delta} + \mathcal{O}(\|\boldsymbol{\delta}\|^3), \tag{2}$$

where $\mathbf{F}_{\boldsymbol{\theta}}$ is the Fisher Information Matrix (FIM) (Amari, 1998), capturing the local curvature of the predictive distribution $\mathbf{F}_{\boldsymbol{\theta}} = \mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathbf{y}\sim p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})}\left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})\, \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})^\top\right]\right]$. To obtain a tractable approximation of $D_{\mathrm{KL}}(Q_T|P)$ in terms of parameter updates that occur during iterative fine-tuning, we adopt a standard Laplace approximation around the pretrained solution (MacKay, 1992; Ritter et al., 2018; Kirkpatrick et al., 2017). Under this approximation, the weight-space distributions are locally modeled using the pretrained Fisher information as a common precision matrix: $Q_t = \mathcal{N}(\boldsymbol{\theta}_t, \mathbf{F}_{\boldsymbol{\theta}_0}^{-1})$ for $t = 0, \ldots, T$, with $Q_0 = P$. Writing $\boldsymbol{\delta}_t = \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t$ for the parameter update at iteration t, the shared-precision model gives $D_{\mathrm{KL}}(Q_T \| P) = \frac{1}{2}\left\|\sum_{t=0}^{T-1} \boldsymbol{\delta}_t\right\|^2_{\mathbf{F}_{\boldsymbol{\theta}_0}}$. Applying the triangle and Cauchy-Schwarz inequalities in the Fisher norm bounds the divergence by the fine-tuning path's curvature-weighted energy:

Generalization divergence governed by Fisher information

$$D_{\mathrm{KL}}(Q_T \| P) \;\le\; \frac{\bar{\gamma}\, T}{2} \sum_{t=0}^{T-1} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t, \tag{3}$$

where $\bar{\gamma}$ is the smallest constant satisfying $\boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t \le \bar{\gamma}\, \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t$ for all t, with $\bar{\gamma} = 1$ when the curvature along the update directions is unchanged.

Equation 3 bounds the divergence from the pretrained prior by an accumulation of curvature-weighted updates along the fine-tuning trajectory. Rather than depending solely on parameter displacement, each update is weighted by the local Fisher geometry, emphasizing changes along directions to which the model is most sensitive. Consequently, the KL term governing the PAC-Bayes bound in Eq. 1 is linked to *curvature-aligned adaptation*, providing a principled basis for using Fisher information to characterize how different parameter groups contribute to fine-tuning. This motivates tracking the evolution of their Fisher structure to determine where continued adaptation remains necessary and where parameters can be progressively frozen. The full derivation and empirical validation are provided in Appendices A.1, A.1.3 and A.5.

3.2 Curvature Shift as an Adaptation Signal

To determine where continued adaptation remains necessary, we next decompose the Fisher-governed KL cost in Eq. 3 across layers. Following standard structured approximations to the Fisher Information Matrix (FIM) (Martens and Grosse, 2015), we use a block-diagonal approximation $\mathbf{F}_{\boldsymbol{\theta}} \approx \mathrm{diag}(\mathbf{F}_{\boldsymbol{\theta}_1}, \ldots, \mathbf{F}_{\boldsymbol{\theta}_L})$ under which the Eq. 3 decomposes across layers as:

$$D_{\mathrm{KL}}(Q_T \| P) \;\le\; \frac{\bar{\gamma}\, T}{2} \sum_{t=0}^{T-1} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t \;\approx\; \frac{\bar{\gamma}\, T}{2} \sum_{t=0}^{T-1} \sum_{\ell=1}^{L} c_{t,\ell}, \qquad c_{t,\ell} := \boldsymbol{\delta}_{t,\ell}^\top \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} \boldsymbol{\delta}_{t,\ell}, \tag{4}$$

where the approximation is due solely to the block-diagonal structure. Since the FIM is positive semi-definite, $c_{t,\ell} \ge 0$. Consequently, freezing layer ℓ at iteration t^* removes its subsequent Fisher-weighted contributions, $\Delta\mathcal{C}_\ell = \frac{\bar{\gamma} T}{2} \sum_{t=t^*}^{T-1} c_{t,\ell} \ge 0$, thereby preventing further accumulation of that layer's contribution to the KL bound and, consequently, to the PAC-Bayes generalization bound. However, reducing this cost alone does not justify freezing, since premature removal of a layer may suppress task-relevant adaptation. The key question is therefore *which layers have adapted sufficiently such that further updates provide diminishing benefit?* We characterize this condition through the *stabilization* of a layer's Fisher-weighted contribution. Importantly, stabilization concerns the temporal evolution of the contribution rather than its absolute magnitude, where a layer may retain a nonzero cost while its role in ongoing adaptation has effectively ceased to evolve. Accordingly, when $c_{t,\ell} - c_{t-1,\ell} \approx 0$, the layer's contribution to the Fisher-weighted adaptation cost changes only

marginally across successive iterations, indicating saturation of its ongoing adaptation dynamics. We therefore define the per-layer contribution drift as:

$$\Delta c_{t,\ell} := c_{t,\ell} - c_{t-1,\ell}, \tag{5}$$

which quantifies the temporal evolution of each layer's contribution to the Fisher-weighted adaptation cost. Under locally stable update magnitudes, the update-dependent component varies slowly, leaving changes in the local Fisher geometry as the dominant source of variation in $\Delta c_{t,\ell}$. This yields:

Adaptation governed by FIM drift

$$c_{t,\ell} - c_{t-1,\ell} \;\approx\; \boldsymbol{\delta}_{t,\ell}^{\top} \Delta \mathbf{F}_{t,\ell} \boldsymbol{\delta}_{t,\ell}, \quad \Delta \mathbf{F}_{t,\ell} := \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} - \mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}. \tag{6}$$

Equation 6 links the evolution of a layer's adaptation cost directly to temporal changes in its Fisher geometry. When the Fisher structure stabilizes, $\Delta \mathbf{F}_{t,\ell} \approx 0$, the layer's contribution changes only marginally, indicating diminishing ongoing adaptation. Such layers become candidates for freezing, whereas layers exhibiting persistent Fisher drift remain actively involved in reshaping task-specific geometry and should continue to adapt (see the empirical analysis in Figures 4 and 5). The complete decomposition and underlying local stability assumptions are provided in Appendix A.3. These assumptions are empirically assessed across four VTAB-1k tasks in Appendix A.5. Fisher-scale changes remain small (0.10-0.17%), while structural Fisher drift persists, supporting the distinction between magnitude stability and structural evolution.

3.3 Fisher-Guided Adaptive Selective Fine-Tuning Algorithm (FisherAdapTune)

A Scale-Robust Measure of Fisher Structural Drift. The preceding analysis motivates the temporal variation of the FIM as a principled signal for parameter selection. To design a criterion that captures structural changes in Fisher information while remaining robust to scale, for each parameter group, we transform its Fisher values as $\tilde{f}_i = \log_{10}(|f_i| + \varepsilon)$, and construct probability distributions from histograms with shared bin edges for consecutive Fisher states. We apply additive smoothing and normalize the histogram counts, as detailed in Appendix A.4. This transformation mitigates the effect of large-magnitude entries while preserving the ordering of the Fisher entries. To quantify the evolution of this distribution across training iterations, we measure the divergence between consecutive Fisher-induced distributions using the Jensen-Shannon (JS) divergence. For two probability mass functions p and q, the JS divergence is defined as:

$$\mathrm{JS}(p, q) = \frac{1}{2}\mathrm{KL}(p \parallel M) + \frac{1}{2}\mathrm{KL}(q \parallel M), \quad M = \frac{1}{2}(p + q). \tag{7}$$

And the JS distance is:

$$d_{\mathrm{JS}}(p, q) = \sqrt{\mathrm{JS}(p, q)}. \tag{8}$$

which is symmetric and bounded. Appendix A.4 provides the computational details and characterizes its sensitivity to changes in the Fisher-value distribution and its robustness to uniform Fisher scaling.

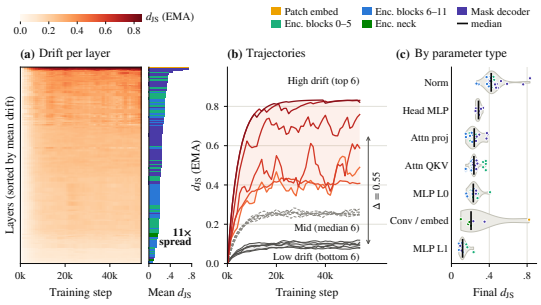


Figure 4: Layer-wise Jensen-Shannon distance between consecutive Fisher distributions in SAM2-Tiny during full fine-tuning on crack segmentation; $EMA[d^t_{\mathrm{JS}_i}(p(\tilde{\mathbf{F}}_i^{t-1}), p(\tilde{\mathbf{F}}_i^t))]$.

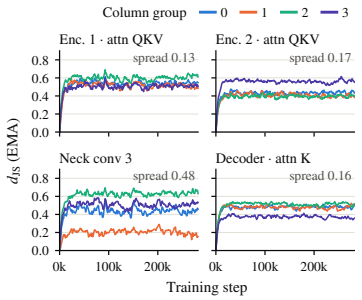


Figure 5: Intra-layer Fisher structural drift heterogeneity in SAM2-Tiny during full fine-tuning on crack segmentation.

Empirical Analysis: Fisher structural drift is heterogeneous across layers and parameter groups. As shown in Appendix Figure 12, Fisher structure can continue to evolve despite locally stable Fisher

scales and update magnitudes. We next examine whether this structural drift is distributed uniformly across layers and parameter groups. Using the JS distance defined above and the efficient diagonal Kronecker approximation introduced by AdaFisher (Gomes et al., 2024), we track Fisher structural drift during full fine-tuning. The approximation is given as:

$$\tilde{\mathbf{F}}_{\ell}^{D} = \mathbf{H}_{D_{\ell-1}} \otimes \mathbf{S}_{D_{\ell}}, \tag{9}$$

where $\mathbf{H}_{D_{\ell-1}}$ and $\mathbf{S}_{D_{\ell}}$ denote diagonal activation- and gradient-correlation factors, respectively. This formulation reduces the computational and memory complexity of Fisher estimation to linear in the number of parameters. We measure $d_{\mathrm{JS}}(p_{t,\ell}, p_{t-1,\ell})$ throughout full fine-tuning of SAM2 (Ravi et al., 2024) and SegFormer (Xie et al., 2021) on crack segmentation. Figure 4 reveals evident layer-wise heterogeneity in SAM2-Tiny, with an $11\times$ difference in mean drift between high-drift and near-stationary layers. This separation emerges early ($\sim$5k steps), persists throughout training, and varies systematically across parameter types. A near-zero d_{JS} indicates a stable Fisher-value distribution, whereas persistent drift indicates continued structural change. These observations support Fisher structural drift as an operational signal of continued adaptation. The parameter-group analysis in Figure 5, shown in full in Appendix Figure 21, further reveals substantial intra-layer heterogeneity: drift differences between groups exceed 0.5 in some layers and 0.1 across much of the network. Similar behavior is observed for SegFormer in Appendix Figure 22. Collectively, these results show that *Fisher structural drift is heterogeneous across layers and parameter groups*, motivating adaptive group-wise selection rather than a uniform freezing strategy.

Fisher Structural Drift Reveals Where Adaptation Persists

Our theoretical analysis and empirical observations exhibit that Fisher structural drift is non-uniform across layers and parameter groups, indicating that only a subset of parameters remain dynamically relevant throughout fine-tuning. Therefore, stabilized groups can be frozen, while groups with persistent curvature drift remain trainable.

FisherAdapTune. Building upon the finding above, we propose *FisherAdapTune*, a Fisher-guided Adaptive Fine-Tuning framework that updates only parameter groups that remain useful for adaptation. The motivation is the non-uniform *Fisher structural drift* observed across layers and parameter blocks (Figure 21) where some groups continue to reorient their local curvature, while others quickly stabilize. FisherAdapTune tracks this evolution with the Jensen-Shannon (JS) distance between consecutive Fisher distributions. Groups with low stable JS distance are treated as saturated and progressively frozen, where groups with persistent drift remain trainable. All parameters are trainable at initialization, which creates an implicit warm-up phase before the first freezing decision. Fisher statistics are collected every m steps and smoothed over time, while freezing is evaluated only every n steps. This separates noisy curvature estimation from parameter selection and prevents premature freezing. At each freezing point, FisherAdapTune applies an adaptive threshold over the smoothed JS scores, yielding a data-driven active set without manual layer schedules. Algorithm 1 summarizes the procedure, and Appendix B discusses additional implementation details and training overhead.

Algorithm 1 FisherAdapTune Algorithm

Require: Pre-Trained Model θ, dataset $\mathcal{D}$, hyperparameters: EMA factor β, threshold scale λ, block count K, fisher collection interval m, freezing interval n, total steps $T_{\max}$.

for $t = 1, \ldots, T_{\max}$ **do**
  Compute $\nabla_\theta \mathcal{L}(\mathcal{D}; \theta)$ via forward/backward pass
  if $t \bmod m = 0$ **then**
    Estimate per-layer Fisher curvature $\tilde{\mathbf{F}}^{t}_{D_i}$ via Eq. 9
    Partition $\tilde{\mathbf{F}}^{t}_{D_i}$ into K blocks $\{\tilde{\mathbf{F}}^{t}_{w_{ik}}\}_{k=1}^{K}$
    Build PMFs $p(\tilde{\mathbf{F}}^{t}_{w_{ik}})$ via Appendix Eq. 22
    Measure curvature shift $d^{t}_{\mathrm{JS}_{ik}}(p_{t-1}, p_t)$ via Eq 8
    Smooth via EMA: $\hat{d}^{t}_{\mathrm{JS}_{ik}} \leftarrow \beta\, \hat{d}^{t-1}_{\mathrm{JS}_{ik}} + (1-\beta)\, d^{t}_{\mathrm{JS}_{ik}}$
    Compute mean curvature shift $\bar{d}_{\mathrm{JS}_{ik}}$
  if $t \bmod n = 0$ **then**
    Compute global statistics of JS distances $\mu \leftarrow \mathrm{mean}_{i,k}(\bar{d}_{\mathrm{JS}_{ik}})$; $\sigma \leftarrow \mathrm{std}_{i,k}(\bar{d}_{\mathrm{JS}_{ik}})$
    Set adaptive threshold $\tau \leftarrow \mu + \lambda\sigma$
    Freeze stabilized parameter groups ($\bar{d}_{\mathrm{JS}_{ik}} < \tau$)
  Update only the remaining tunable parameter groups $\theta \leftarrow \theta - \eta\, \nabla_\theta \mathcal{L}(\mathcal{D}; \theta)$
end for

Which Parameters Does FisherAdapTune Select? Figure 6 shows how the trainable set evolves for SAM2-Tiny on the downstream crack segmentation task. The budget falls from 31M to 271K parameters, averaging 6.95M over training.

Large encoder attention and MLP groups freeze relatively early, whereas patch embedding, several mask-decoder groups, and small normalization groups remain trainable longer. This suggests that trainable lifetime is not determined simply by parameter count. Compact groups can continue adapting after much larger groups have stabilized. The broader pattern depends on architecture and task. DINOv2 consistently retains patch embedding and the classification head across VTAB-1k tasks, while SegFormer-MiT-B3 sustains adaptation in intermediate encoder groups (Appendix F.2). These patterns are consistent with the observation of Lee et al. (2023) that useful adaptation locations depend on the distribution shift. FisherAdapTune discovers these differences during training without prescribing which layers should remain trainable.

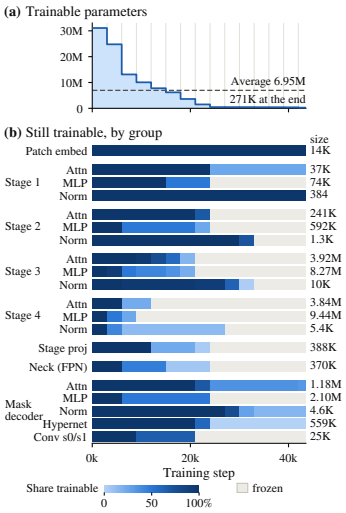


Figure 6: Trainable parameter budget and group lifetimes for SAM2-Tiny in crack segmentation.

4 Experiments & Results

Experimental Setup. We evaluate FisherAdapTune across all 19 VTAB-1k classification tasks, including the natural, specialized, and structured categories (Zhai et al., 2019), using DINOv2 ViT-S/14. Our objective is to compare adaptation strategies under a common experimental protocol. We further consider crack segmentation, a challenging dense-prediction task involving thin, curvilinear structures and substantial distribution shift from pretraining data (Chen et al., 2024; 2023), using SAM2-Tiny, SAM2-Large, SegFormer-MiT-B0, and SegFormer-MiT-B3. Models are fine-tuned on OmniCrack30k (Benz and Rodehorst, 2024) and evaluated both in-distribution and zero-shot on Concrete3k, Facade390, and Road420 (Li et al., 2023; Wang and Su, 2022; Ge et al., 2024). We compare against full fine-tuning, LoRA (Hu et al., 2021), BitFit (Zaken et al., 2022), LayerNorm tuning (Zhao et al., 2023), decoder-only tuning, AutoFreeze (Liu et al., 2021), Child-Tuning (Xu et al., 2021), FISH-Mask (Sung et al., 2021), and random parameter selection. Full experimental details are provided in Appendix C.

Effective Number of Trainable Parameters. Since FisherAdapTune progressively changes the active parameter set, we report the average number of parameters updated per iteration $\bar{N} = T^{-1} \sum_{t=1}^{T} N_t$, where N_t is the number of trainable parameters at iteration t and T is the number of training iterations. For static PEFT methods, the same parameters remain trainable throughout training, so $\bar{N}$ equals their fixed trainable count. For FisherAdapTune, the trainable set changes as parameter groups are progressively frozen, so $\bar{N}$ is the average number of trainable parameters across training iterations.

4.1 Comparative Results

Image Classification. The results in Table 2 show that FisherAdapTune achieves the highest macro Top-1 accuracy across VTAB-1k tasks (61.62%), surpassing full fine-tuning (59.97%) while using a smaller effective trainable parameter budget. Unlike static PEFT methods and AutoFreeze, which follow a predefined layer-freezing order, FisherAdapTune automatically identifies which parameter groups to freeze based on the stabilization of their Fisher drift.

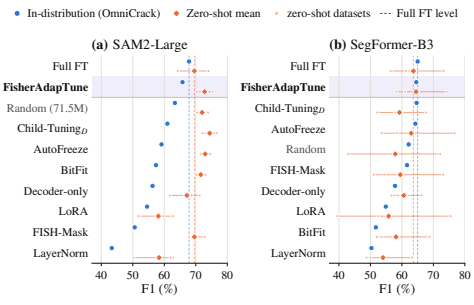


Figure 7: Segmentation F1-scores for (a) SAM2-Large and (b) SegFormer-B3.

Semantic Segmentation. FisherAdapTune achieves a balance between in-distribution performance and zero-shot generalization. It remains competitive on the in-distribution test set while improving zero-shot performance in several settings (Figure 7). Complete results are provided in Appendix Figure 14 and Table 4). This suggests that its adaptive parameter selection enables effective task adaptation without sacrificing transferable representations. Its improvement over matched-budget random selection

further shows that this balance depends on *which* parameters are updated, rather than on parameter sparsity alone. Multi-seed results in Appendix Table 5 further support the robustness of these findings.

Table 2: PEFT comparison across VTAB-1k tasks using DINOv2 ViT-S/14. "Trainable" denotes the effective number of trainable parameters. Bold and underlined values denote the best and second-best Top-1 accuracy per task, respectively.

		Natural							*Specialized*				*Structured*									
Method	Metric	CIFAR-100	Caltech-101	DTD	Flowers-102	Oxford Pets	SVHN	SUN397	EuroSAT	Patch-Cam	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Elev	Macro mean ± task SD
Full FT	Top-1 (%)	56.76	85.15	62.66	95.89	86.45	**88.88**	38.49	89.08	84.30	85.14	**76.91**	42.86	46.71	**53.59**	47.40	**23.47**	**31.66**	14.95	29.01	$59.97_{\pm 25.91}$
	Trainable (M)	22.09	22.10	22.07	22.10	22.07	22.06	22.21	22.06	22.06	22.07	22.06	22.06	22.06	22.06	22.06	22.06	22.06	22.06	22.06	$22.08_{\pm 0.04}$
LoRA	Top-1 (%)	**62.96**	85.88	**75.80**	**97.54**	**92.34**	53.77	**46.71**	93.91	82.97	82.37	75.47	52.73	50.39	46.44	53.45	7.25	26.30	12.66	27.37	$59.28_{\pm 27.49}$
	Trainable (M)	0.08	0.08	0.06	0.08	0.05	0.04	0.19	0.04	0.04	0.05	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	$0.06_{\pm 0.04}$
BitFit	Top-1 (%)	53.95	77.83	68.14	95.74	89.07	51.17	34.74	91.34	79.75	72.41	73.83	34.95	47.59	39.87	**55.41**	10.88	20.20	9.96	26.24	$54.37_{\pm 27.10}$
	Trainable (M)	0.09	0.09	0.07	0.09	0.07	0.06	0.20	0.06	0.05	0.07	0.05	0.05	0.05	0.05	0.05	0.06	0.06	0.06	0.05	$0.07_{\pm 0.03}$
LayerNorm	Top-1 (%)	54.02	78.96	68.14	96.75	90.22	34.51	40.71	87.08	79.00	69.06	73.58	28.15	25.13	34.90	46.69	6.77	15.81	6.80	20.77	$50.37_{\pm 29.38}$
	Trainable (M)	0.06	0.06	0.04	0.06	0.03	0.02	0.17	0.02	0.02	0.04	0.02	0.02	0.02	0.02	0.02	0.03	0.03	0.03	0.02	$0.04_{\pm 0.04}$
Lin. Probe	Top-1 (%)	45.02	74.60	65.59	95.46	88.66	29.63	38.79	84.55	78.36	61.78	73.58	25.27	24.35	32.81	38.40	6.52	15.12	6.52	16.59	$47.45_{\pm 29.21}$
	Trainable (M)	0.04	0.04	0.02	0.04	0.01	<0.01	0.15	<0.01	<0.01	0.02	<0.01	<0.01	<0.01	<0.01	<0.01	0.01	0.01	0.01	<0.01	$0.02_{\pm 0.04}$
AutoFreeze	Top-1 (%)	55.28	**88.41**	68.94	95.09	85.94	84.47	36.91	94.71	84.32	**85.95**	75.32	39.92	**54.66**	45.18	51.48	16.77	9.94	13.10	**32.21**	$58.87_{\pm 28.57}$
	Trainable (M)	4.19	9.58	4.41	2.29	1.81	4.47	4.40	3.72	4.75	3.82	1.26	1.39	3.39	1.24	2.58	3.08	2.67	1.64	7.51	$3.59_{\pm 2.13}$
Child-Tuning$_D$	Top-1 (%)	48.86	84.51	67.50	95.61	88.09	76.44	38.61	95.28	85.40	85.81	73.60	50.61	51.74	51.59	53.16	20.93	26.96	14.27	30.55	$59.97_{\pm 25.99}$
	Trainable (M)	6.66	6.66	6.63	6.66	6.63	6.62	6.77	6.62	6.62	6.63	6.62	6.62	6.62	6.62	6.62	6.62	6.62	6.62	6.62	$6.64_{\pm 0.04}$
FISH-Mask (2%)	Top-1 (%)	39.37	75.85	71.01	95.28	90.24	71.27	28.58	94.50	81.38	79.00	76.37	**57.30**	53.03	47.25	53.31	19.71	18.92	8.51	24.32	$57.12_{\pm 27.58}$
	Trainable (M)	0.46	0.46	0.44	0.45	0.44	0.44	0.56	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	0.44	$0.45_{\pm 0.03}$
FisherAdapTune	Top-1 (%)	57.11	81.93	71.65	96.03	85.20	87.67	36.24	**95.76**	**85.83**	84.44	76.86	56.01	51.17	52.50	54.15	21.28	29.90	**15.49**	31.59	$\mathbf{61.62}_{\pm \mathbf{25.87}}$
	Trainable (M)	6.25	9.30	5.62	6.61	6.63	7.56	6.02	6.65	7.22	6.62	6.57	5.78	5.75	6.09	7.08	9.57	8.85	7.98	8.93	$7.11_{\pm 1.25}$

Computational Cost. FisherAdapTune introduces a controllable runtime overhead (Figure 8). With frequent Fisher collection (m=2), its per-step cost is $1.14\times$ that of full fine-tuning. Increasing the collection interval to m=8 reduces this overhead to $1.06\times$, while decreasing steady-state peak memory from 5.56 GB to 4.13 GB (26%). The lifetime peak remains comparable to full fine-tuning since FisherAdapTune begins with an all-trainable phase. Accuracy remains stable across the evaluated collection intervals (Section 4.2 and Appendix E.2). Thus, collecting Fisher statistics less frequently reduces per-step cost with minimal change in accuracy across the evaluated intervals. Complete timing, memory, and baseline comparisons are provided in Appendix B.4.

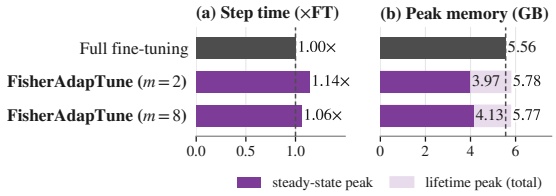


Figure 8: FisherAdapTune adds modest per-step overhead while reducing steady-state peak memory relative to full fine-tuning.

4.2 Ablation Studies

Parameter-Selection Signals. To further assess whether changes in Fisher structure provide a useful criterion for progressive freezing, we replace Fisher structural drift with Fisher magnitude, Fisher-magnitude drift, gradient RMS, update RMS, and random selection while keeping the remaining algorithm fixed. Figure 9 shows that Fisher structural drift remains heterogeneous across layers, distinguishing groups with ongoing adaptation from those approaching stabilization. On the other hand, gradient RMS is dominated by a few layers and declines broadly during training. As a first-order magnitude measure, it may diminish even while the approximate second-order structure continues to change. Consistent with this distinction, Fisher structural drift achieves the strongest overall performance across eight VTAB-1k tasks (Table 3 and Appendix Figure 15). Full signal trajectories, comparisons with Fisher variants, and experimental details are provided in Appendix E.1.

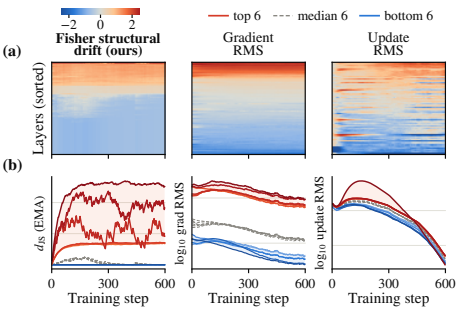


Figure 9: (a) Layer-wise signal heatmaps and (b) representative trajectories, monitored during full fine-tuning of DINOv2 on CIFAR-100. Fisher structural drift remains heterogeneous as gradient and update RMS decay.

Hyperparameter Sensitivity. We analyze the sensitivity of FisherAdapTune to its main hyperparameters, including the parameter-group count K, Fisher interval m, freezing interval n, EMA factor β, and threshold scale λ. Appendix Table 6 and Figure 10 show stable performance across m, n, and β, with larger m and n reducing computational overhead as the frequency of Fisher collection and freezing decision reduces. In contrast, K and λ control the performance-efficiency trade-off, with λ determining which adapting groups remain trainable. In additional segmentation experiments with SAM2 and SegFormer (Figure 11), smaller λ values retain more groups and improve in-distribution performance, while zero-shot performance peaks at intermediate budgets.

Table 3: Signal ablations on eight VTAB-1k tasks. λ is tuned to match the trainable budget of Fisher structural drift. Trainable (M) is the effective trainable parameter count. Bold and underlined values mark the best and second-best Top-1 accuracy per task.

			Natural						*Specialized*		
Signal	λ	Metric	CIFAR-100	Caltech-101	DTD	Flowers-102	Oxford Pets	SVHN	EuroSAT	Patch-Cam	Macro mean $\pm$ task SD
Random	-1.25	Top-1 (%)	23.12	56.22	59.04	70.11	68.74	67.24	90.48	75.80	$63.84_{\pm 19.52}$
		Trainable (M)	3.72	5.25	3.95	3.72	3.71	3.71	3.83	3.71	$3.95_{\pm 0.53}$
Gradient RMS	-0.85	Top-1 (%)	49.36	80.07	65.43	77.22	21.72	**88.96**	<u>95.72</u>	**87.03**	$70.69_{\pm 24.61}$
		Trainable (M)	7.96	6.88	6.14	3.58	2.42	13.46	10.74	13.29	$8.06_{\pm 4.15}$
Update RMS	-1.0	Top-1 (%)	<u>52.12</u>	67.96	66.97	95.35	<u>86.45</u>	75.33	95.63	82.49	$\underline{77.79}_{\pm 15.13}$
		Trainable (M)	5.26	5.44	5.40	9.75	8.73	2.61	4.01	3.34	$\underline{5.57}_{\pm 2.50}$
Fisher magnitude	-0.8	Top-1 (%)	16.53	**86.14**	<u>67.98</u>	<u>95.90</u>	**88.23**	77.95	92.63	82.86	$76.03_{\pm 25.57}$
		Trainable (M)	1.68	21.56	21.54	21.56	21.53	1.80	1.40	1.44	$11.56_{\pm 10.67}$
Fisher magnitude drift	-0.2	Top-1 (%)	10.03	28.46	26.17	16.04	38.13	26.56	74.56	77.26	$37.15_{\pm 25.35}$
		Trainable (M)	1.51	1.79	0.69	0.71	0.73	1.58	0.61	0.75	$1.04_{\pm 0.49}$
Fisher structural drift (ours)	-0.65	Top-1 (%)	**57.11**	<u>81.93</u>	**71.65**	**96.03**	85.20	<u>87.67</u>	**95.76**	<u>85.83</u>	$\mathbf{82.65}_{\pm \mathbf{12.91}}$
		Trainable (M)	6.25	9.30	5.62	6.61	6.63	7.56	6.65	7.22	$6.98_{\pm 1.10}$

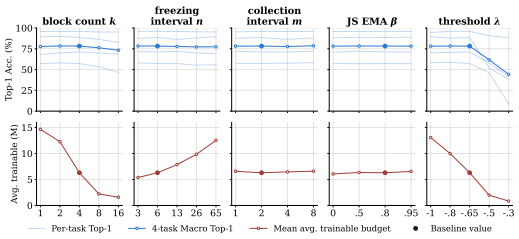
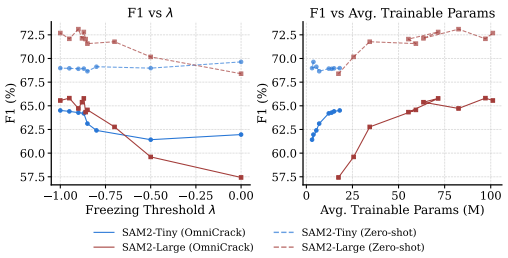
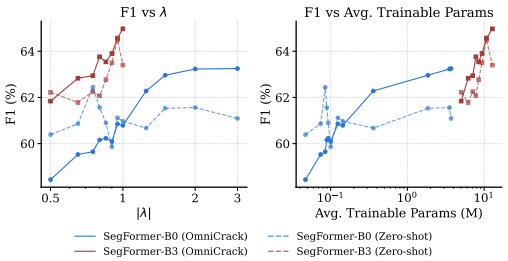


Figure 10: FisherAdapTune hyperparameter sensitivity. Top: macro Top-1 (light blue: individual tasks). Bottom: mean trainable budget. Accuracy is relatively insensitive to n, m, and β, while λ and K govern the accuracy-efficiency trade-off.



(a) SAM2



(b) SegFormer

Figure 11: Ablation over the freezing threshold λ in crack segmentation task. FisherAdapTune balances in-distribution adaptation and zero-shot generalization by avoiding both premature freezing and excessive source-task specialization.

5 Conclusion

We introduced FisherAdapTune, a Fisher-guided adaptive fine-tuning framework that progressively freezes parameter groups as their Fisher structure stabilizes, selecting the trainable set without a prescribed layer schedule. Our PAC-Bayesian analysis relates Fisher-weighted updates to a generalization bound, motivating curvature-aware parameter selection during fine-tuning. Across VTAB-1k classification tasks and crack segmentation, FisherAdapTune achieves the highest macro classification accuracy among the evaluated methods and competitive in-distribution and zero-shot segmentation performance while reducing the effective trainable-parameter budget. Interestingly, across different architectures and tasks, FisherAdapTune retains different combinations of input, output, and normalization groups trainable while freezing attention and MLP groups at architecture-dependent rates. The results support Fisher drift as a task-aware signal for allocating updates. Future work will evaluate FisherAdapTune across broader architectures, datasets, and downstream tasks.

AI USE STATEMENT

In this work, generative AI tools were used to assist with language editing, including improving the clarity, grammar, organization, and conciseness of author-written text. They were also used to support the preparation and refinement of visualizations, including their layout, presentation, and captions. All research ideas, hypotheses, methodological choices, experimental designs, analyses, and interpretations originated from the authors. The AI tools did not independently formulate the research contributions or determine the conclusions of this work. All AI-assisted text and visual materials were carefully reviewed, revised, and verified by the authors. The authors take full responsibility for the accuracy, originality, and final content of the manuscript.

REPRODUCIBILITY STATEMENT

We support reproducibility by providing a complete description and pseudocode of FisherAdapTune in the main paper, together with the theoretical assumptions and derivations underlying the method. The experimental setup specifies the datasets, model architectures, adaptation baselines, evaluation metrics, and training protocol. The appendix further reports implementation details, hyperparameter settings and sensitivity analyses, Fisher-drift observations, ablation studies, and training-time and memory overhead. These materials are intended to provide the information necessary to reproduce the theoretical and empirical results reported in this work.

REFERENCES

Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning, December 2020.

Shun-ichi Amari. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10(2): 251–276, February 1998. ISSN 0899-7667. doi: 10.1162/089976698300017746.

Shun-ichi Amari. *Information Geometry and Its Applications*. Springer, February 2016. ISBN 978-4-431-55978-8.

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. Composable Sparse Fine-Tuning for Cross-Lingual Transfer, February 2023.

Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring Visual Prompts for Adapting Large-Scale Models, June 2022.

Christian Benz and Volker Rodehorst. Omni-Crack30k: A Benchmark for Crack Segmentation and the Reasonable Effectiveness of Transfer Learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3876–3886, June 2024. doi: 10.1109/CVPRW63382.2024.00392.

Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. AdaptFormer: Adapting Vision Transformers for Scalable Visual Recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, December 2022a.

Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision Transformer Adapter for Dense Predictions. In *The Eleventh International Conference on Learning Representations*, September 2022b.

Zhuangzhuang Chen, Jin Zhang, Zhuonan Lai, Guanming Zhu, Zun Liu, Jie Chen, and Jianqiang Li. The Devil is in the Crack Orientation: A New Perspective for Crack Detection. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6630–6640, October 2023. doi: 10.1109/ICCV51070.2023.00612.

Zhuangzhuang Chen, Zhuonan Lai, Jie Chen, and Jianqiang Li. Mind marginal non-crack regions: Clustering-inspired representation learning for crack segmentation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12698–12708, June 2024. doi: 10.1109/CVPR52733.2024.01207.

Felix Dangel, Bálint Mucsányi, Tobias Weber, and Runa Eschenhagen. Kronecker-factored Approximate Curvature (KFAC) From Scratch, July 2025.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Dominik M. Endres and Johannes E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, 2003.

Jonathan Frankle and Michael Carbin. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks, March 2019.

K. Ge, C. Wang, Y. T. Guo, Y. S. Tang, Z. Z. Hu, and H. B. Chen. Fine-tuning vision foundation model for crack segmentation in civil infrastructures. *Construction and Building Materials*, 431: 136573, June 2024. ISSN 0950-0618. doi: 10.1016/j.conbuildmat.2024.136573.

Damien MARTINS Gomes, Yanlei Zhang, Eugene Belilovsky, Guy Wolf, and Mahdi S. Hosseini. AdaFisher: Adaptive Second Order Optimization via Fisher Information. In *The Thirteenth International Conference on Learning Representations*, October 2024.

Demi Guo, Alexander M. Rush, and Yoon Kim. Parameter-Efficient Transfer Learning with Diff Pruning, June 2021.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey, April 2024.

Xuehai He, Chunyuan Li, Pengchuan Zhang, Jianwei Yang, and Xin Eric Wang. Parameter-Efficient Model Adaptation for Vision Transformers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1):817–825, June 2023. ISSN 2374-3468. doi: 10.1609/aaai.v37i1.25160.

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*, pages 2790–2799. PMLR, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models, October 2021.

Qidong Huang, Xiaoyi Dong, Dongdong Chen, Weiming Zhang, Feifei Wang, Gang Hua, and Nenghai Yu. Diversity-Aware Meta Visual Prompting. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10878–10887, Vancouver, BC, Canada, June 2023. IEEE. ISBN 979-8-3503-0129-8. doi: 10.1109/CVPR52729.2023.01047.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual Prompt Tuning, July 2022.

Shibo Jie, Zhi-Hong Deng, Shixuan Chen, and Zhijuan Jin. Convolutional Bypasses Are Better Vision Transformer Adapters. In *ECAI 2024*, pages 202–209. IOS Press, 2024. doi: 10.3233/FAIA240489.

Minyoung Kim and Timothy Hospedales. BayesTune: Bayesian Sparse Deep Model Fine-tuning. *Advances in Neural Information Processing Systems*, 36:65317–65365, December 2023.

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.

Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution. In *International Conference on Learning Representations*, October 2021.

Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2210.11466.

Yongshang Li, Ronggui Ma, Han Liu, and Gaoli Cheng. Real-time high-resolution neural network with semantic guidance for crack segmentation. *Automation in Construction*, 156:105112, December 2023. ISSN 0926-5805. doi: 10.1016/j.autcon.2023.105112.

Haowei Lin, Baizhou Huang, Haotian Ye, Qinyu Chen, Zihao Wang, Sujian Li, Jianzhu Ma, Xiaojun Wan, James Zou, and Yitao Liang. Selecting Large Language Model to Fine-tune via Rectified Scaling Law. In *Proceedings of the 41st International Conference on Machine Learning*, pages 30080–30107. PMLR, July 2024.

Yuhan Liu, Saurabh Agarwal, and Shivaram Venkataraman. AutoFreeze: Automatically Freezing Model Blocks to Accelerate Fine-tuning, April 2021.

Sanae Lotfi, Marc Anton Finzi, Sanyam Kapoor, Andres Potapczynski, Micah Goldblum, and Andrew Gordon Wilson. PAC-Bayes compression bounds so tight that they can explain generalization. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=o8nYuR8ekFm.

David J. C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992. doi: 10.1162/neco.1992.4.3.448.

James Martens and Roger Grosse. Optimizing Neural Networks with Kronecker-factored Approximate Curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2408–2417. PMLR, June 2015.

David A. McAllester. PAC-Bayesian model averaging. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, pages 164–170, Santa Cruz California USA, July 1999a. ACM. ISBN 978-1-58113-167-3. doi: 10.1145/307400.307435.

David A. McAllester. Some PAC-Bayesian Theorems. *Machine Learning*, 37(3):355–363, December 1999b. ISSN 1573-0565. doi: 10.1023/A:1007618624809.

Peng Mi, Li Shen, Tianhe Ren, Yiyi Zhou, Xiaoshuai Sun, Rongrong Ji, and Dacheng Tao. Make Sharpness-Aware Minimization Stronger: A Sparsified Perturbation Approach. *Advances in Neural Information Processing Systems*, 35:30950–30962, December 2022.

Namuk Park and Songkuk Kim. How Do Vision Transformers Work? In *International Conference on Learning Representations*, October 2021.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment Anything in Images and Videos, October 2024.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable Laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Skdvd2xAZ.

Alexander Soen and Ke Sun. Trade-Offs of Diagonal Fisher Information Matrix Estimators. *Advances in Neural Information Processing Systems*, 37:5870–5912, December 2024.

Yi-Lin Sung, Varun Nair, and Colin Raffel. Training Neural Networks with Fixed Sparse Masks - Preprint, November 2021.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. Scale Efficiently: Insights from Pretraining and Finetuning Transformers. In *International Conference on Learning Representations*, October 2021.

Flemming Topsøe. Some inequalities for information divergence and related measures of discrimination. *IEEE Transactions on Information Theory*, 46(4):1602–1609, 2000.

Wenjun Wang and Chao Su. Automatic concrete crack segmentation model based on transformer. *Automation in Construction*, 139:104275, July 2022. ISSN 09265805. doi: 10.1016/j.autcon.2022.104275.

Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. LoRA-Pro: Are Low-Rank Adapters Properly Optimized? In *The Thirteenth International Conference on Learning Representations*, October 2024.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers. In *Advances in Neural Information Processing Systems*, volume 34, pages 12077–12090. Curran Associates, Inc., 2021.

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, September 2021.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models, September 2022.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

Dan Zhang, Tao Feng, Lilong Xue, Yuandong Wang, Yuxiao Dong, and Jie Tang. Parameter-Efficient Fine-Tuning for Foundation Models, January 2025.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023.

Bingchen Zhao, Haoqin Tu, Chen Wei, Jieru Mei, and Cihang Xie. Tuning LayerNorm in Attention: Towards Efficient Multi-Modal LLM Finetuning. In *The Twelfth International Conference on Learning Representations*, October 2023.

Dong Zhao, Jinlong Li, Shuang Wang, Mengyao Wu, Qi Zang, Nicu Sebe, and Zhun Zhong. FisherTune: Fisher-Guided Robust Tuning of Vision Foundation Models for Domain Generalized Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15043–15054, 2025.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. Masking as an Efficient Alternative to Finetuning for Pretrained Language Models, October 2020.

Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: a pac-bayesian compression approach. *arXiv preprint arXiv:1804.05862*, 2018.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE*, 109(1):43–76, January 2021. ISSN 1558-2256. doi: 10.1109/JPROC.2020.3004555.

APPENDIX CONTENTS

A Theoretical Details and Derivations **15**

A.1 PAC-Bayes Derivation and Fisher-Based KL Approximation 15

A.2 Layer-Wise KL Decomposition and Freezing 17

A.3 Curvature Shift as an Adaptation Signal 18

A.4 Quantifying Fisher Structural Drift with Jensen-Shannon Distance 18

A.5 Empirical Assessment of the Theoretical Assumptions 20

B FisherAdapTune: Algorithm and Implementation **21**

B.1 Parameter Grouping 21

B.2 Drift Smoothing and Adaptive Freezing 22

B.3 Complete Algorithm 23

B.4 Training-Time and Memory Overhead 23

C Experimental Setup **24**

C.1 VTAB-1k Classification 24

C.2 Crack Segmentation 25

D Comparative Results **25**

D.1 In-Distribution and Zero-Shot Segmentation Results 25

E Ablation Studies and Hyperparameter Sensitivity **27**

E.1 Comparison of Parameter-Selection Signals 27

E.2 Hyperparameter Sensitivity on VTAB-1k 28

E.3 Freezing-Threshold Sensitivity on Crack Segmentation 30

E.4 Active-Parameter Trajectories Across Thresholds 32

F Fisher Drift and Parameter-Selection Dynamics **34**

F.1 Heterogeneity Across Layers and Parameter Groups 34

F.2 Architectural Patterns in the Selected Parameters 36

A Theoretical Details and Derivations

This appendix collects the assumptions, results, and derivations supporting the main-text theory. Appendix A.1 presents the Fisher-based KL result together with its decomposition proof. Appendices A.2 and A.3 develop the layer-wise decomposition, freezing argument, and contribution-drift derivation. Appendix A.4 describes the JS drift measure, and Appendix A.5 collects the empirical assessments of the assumptions.

A.1 PAC-Bayes Derivation and Fisher-Based KL Approximation

A.1.1 Local Fisher-Scale Stability

Assumption A.1 (Local Fisher-scale stability)**.** The overall Fisher scale varies slowly during fine-tuning: $\|\mathbf{F}_{\boldsymbol{\theta}_t}\|_F \approx \|\mathbf{F}_{\boldsymbol{\theta}_0}\|_F$ uniformly over $t \in \{0, \dots, T-1\}$. This holds when the model is updated with a small learning rate from a pretrained stationary point. Crucially, this assumption constrains only the Fisher *magnitude*; the normalized shape $\mathbf{P}_\ell^t = \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} / \|\mathbf{F}_{\boldsymbol{\theta}_{t,\ell}}\|_F$ is free to reorient across layers and iterations as the importance mass may redistribute across parameters while the layer's aggregate curvature stays constant.

The empirical assessment of Assumption A.1 is provided in Appendix A.5. This condition is not required for the bound in Proposition A.2, which handles the change from $\mathbf{F}_{\boldsymbol{\theta}_0}$ to $\mathbf{F}_{\boldsymbol{\theta}_t}$ through the constant $\bar{\gamma}$; its role is to establish that the overall Fisher magnitude is stable during fine-tuning, which motivates tracking changes in Fisher *structure* rather than magnitude as the adaptation signal.

A.1.2 KL Accumulation Result

Proposition A.2 (KL accumulation along the fine-tuning trajectory)**.** *Let $Q_0 = P = \mathcal{N}(\boldsymbol{\theta}_0, \mathbf{F}_{\boldsymbol{\theta}_0}^{-1})$ denote the pretrained prior, let all posteriors share the pretrained precision, $Q_t = \mathcal{N}(\boldsymbol{\theta}_t, \mathbf{F}_{\boldsymbol{\theta}_0}^{-1})$, and let $\boldsymbol{\delta}_t = \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t$. The following statements hold for any update rule generating $\{\boldsymbol{\delta}_t\}_{t=0}^{T-1}$, irrespective of the optimizer and of the form of regularization (proofs in Appendix A.1.3).*

1. Exact decomposition.

$$D_{\mathrm{KL}}(Q_T \| P) \;=\; \sum_{t=0}^{T-1} \left[\frac{1}{2}\, \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t \;+\; \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0) \right], \tag{10}$$

where the cross terms collect all pairwise interactions between updates:

$$\sum_{t=0}^{T-1} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0) \;=\; \sum_{0 \le s < t < T} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_s. \tag{11}$$

2. Alignment factor. *Equivalently, $D_{\mathrm{KL}}(Q_T \| P) = \kappa_T \sum_t \frac{1}{2}\, \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t$ with*

$$\kappa_T \;:=\; \frac{\big\| \sum_{t=0}^{T-1} \boldsymbol{\delta}_t \big\|_{\mathbf{F}_{\boldsymbol{\theta}_0}}^2}{\sum_{t=0}^{T-1} \|\boldsymbol{\delta}_t\|_{\mathbf{F}_{\boldsymbol{\theta}_0}}^2} \;\in\; [0, T]. \tag{12}$$

$\kappa_T = 1$ when the updates are mutually $\mathbf{F}_{\boldsymbol{\theta}_0}$-orthogonal, $\kappa_T < 1$ only when the pairwise terms in Eq. 11 are negative on net, and $\kappa_T = T$ when all updates coincide in the $\mathbf{F}_{\boldsymbol{\theta}_0}$-geometry.

3. Path-energy bound.

$$D_{\mathrm{KL}}(Q_T \| P) \;\le\; \frac{1}{2} \Big(\sum_{t=0}^{T-1} \|\boldsymbol{\delta}_t\|_{\mathbf{F}_{\boldsymbol{\theta}_0}} \Big)^2 \;\le\; \frac{T}{2} \sum_{t=0}^{T-1} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t. \tag{13}$$

Generalization bound governed by FIM

Let $\bar{\gamma}$ be the smallest constant such that $\boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t \le \bar{\gamma}\, \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t$ for all t. Expressing the path energy in the local Fisher geometry gives:

$$D_{\mathrm{KL}}(Q_T \| P) \;\le\; \frac{\bar{\gamma}\, T}{2} \sum_{t=0}^{T-1} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t. \tag{14}$$

Remark A.3 (On the factor T and the constant $\bar{\gamma}$). By item 2, the unweighted quadratic sum $\sum_t \frac{1}{2} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t$ is itself an upper bound on $D_{\mathrm{KL}}(Q_T \| P)$ only when $\kappa_T \le 1$, i.e., when successive updates are orthogonal or anti-aligned on net. Fine-tuning away from the pretrained initialization operates in the opposite regime: consecutive updates are strongly aligned (mean cosine similarity 0.87-0.89, Appendix A.5), so the pairwise terms in Eq. 11 are predominantly positive and $\kappa_T > 1$. Eq. 13 replaces κ_T by its largest possible value. The factor T is attained for aligned updates of equal $\mathbf{F}_{\boldsymbol{\theta}_0}$-norm and therefore cannot be removed without assumptions on the trajectory. Since T and $\bar{\gamma}$ multiply every per-step term equally, they do not affect how the bound is attributed across layers or iterations. The constant $\bar{\gamma}$ equals one when the curvature along the update directions is unchanged from $\mathbf{F}_{\boldsymbol{\theta}_0}$. Assumption A.1 controls the overall scale of $\mathbf{F}_{\boldsymbol{\theta}_t}$ relative to $\mathbf{F}_{\boldsymbol{\theta}_0}$, whereas $\bar{\gamma}$ additionally absorbs any reorientation of the Fisher along $\boldsymbol{\delta}_t$; it is finite whenever $\boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t > 0$ for every nonzero update, which holds for the damped diagonal Fisher estimates used in practice. The right-hand side of Eq. 14 is therefore the energy of the fine-tuning path measured under the evolving Fisher metric, which upper-bounds the Mahalanobis displacement from the prior under the fixed metric $\mathbf{F}_{\boldsymbol{\theta}_0}$ up to the factor $\bar{\gamma}T$; the two quantities take the same form only when $\mathbf{F}_{\boldsymbol{\theta}_t} = \mathbf{F}_{\boldsymbol{\theta}_0}$ along the trajectory. All posteriors Q_t retain the prior precision $\mathbf{F}_{\boldsymbol{\theta}_0}$, so the Gaussian KL remains a pure quadratic form, and $\mathbf{F}_{\boldsymbol{\theta}_t}$ enters only through this upper bound.

A.1.3 Proof of the KL Decomposition

$$D_{\mathrm{KL}}(Q_T \| P) \;=\; \sum_{t=0}^{T-1} \left[\frac{1}{2}\, \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} \boldsymbol{\delta}_t \;+\; \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_0} (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0) \right].$$

Step 1: KL between two Gaussians with the same covariance. Since $Q_T = \mathcal{N}(\boldsymbol{\theta}_T, \mathbf{F}_{\boldsymbol{\theta}_0}^{-1})$ and $P = \mathcal{N}(\boldsymbol{\theta}_0, \mathbf{F}_{\boldsymbol{\theta}_0}^{-1})$ share the same covariance, the standard KL formula for multivariate Gaussians reduces to a pure quadratic in the difference of means (all log-determinant and trace terms cancel):

$$D_{\mathrm{KL}}(Q_T \| P) \;=\; \frac{1}{2} (\boldsymbol{\theta}_T - \boldsymbol{\theta}_0)^\top \mathbf{F}_{\boldsymbol{\theta}_0} (\boldsymbol{\theta}_T - \boldsymbol{\theta}_0) \;=\; \frac{1}{2} \|\boldsymbol{\theta}_T - \boldsymbol{\theta}_0\|^2_{\mathbf{F}_{\boldsymbol{\theta}_0}}.$$

Where the $|\cdot|^2_{\mathbf{F}_{\boldsymbol{\theta}_0}}$ is squared Mahalanobis norm weighted by the shared precision matrix. The distance measures how far the fine-tuned parameters $\boldsymbol{\theta}_T$ have drifted from the pretrained $\boldsymbol{\theta}_0$ in Fisher-geometry terms.

Step 2: Expressing the total displacement as a telescoping sum. Define the per-step update $\boldsymbol{\delta}_t = \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t$. Summing over all steps:

$$\boldsymbol{\theta}_T - \boldsymbol{\theta}_0 \;=\; \sum_{t=0}^{T-1} \boldsymbol{\delta}_t.$$

Substituting into Step 1:

$$D_{\mathrm{KL}}(Q_T \| P) \;=\; \frac{1}{2} \left\| \sum_{t=0}^{T-1} \boldsymbol{\delta}_t \right\|^2_{\mathbf{F}_{\boldsymbol{\theta}_0}}.$$

Step 3: Expanding the squared norm via a telescoping identity. Let $\mathbf{F}_0 \equiv \mathbf{F}_{\boldsymbol{\theta}_0}$ for brevity. Define the partial-sum vector $\mathbf{s}_t = \sum_{s=0}^{t-1} \boldsymbol{\delta}_s = \boldsymbol{\theta}_t - \boldsymbol{\theta}_0$, with $\mathbf{s}_0 = \mathbf{0}$ and $\mathbf{s}_T = \boldsymbol{\theta}_T - \boldsymbol{\theta}_0$. Then:

$$\frac{1}{2} \|\mathbf{s}_T\|^2_{\mathbf{F}_0} \;=\; \sum_{t=0}^{T-1} \left(\frac{1}{2} \|\mathbf{s}_{t+1}\|^2_{\mathbf{F}_0} - \frac{1}{2} \|\mathbf{s}_t\|^2_{\mathbf{F}_0} \right).$$

Since $\mathbf{s}_{t+1} = \mathbf{s}_t + \boldsymbol{\delta}_t$, each difference expands as:

$$\begin{aligned}\frac{1}{2}\|\mathbf{s}_t + \boldsymbol{\delta}_t\|^2_{\mathbf{F}_0} - \frac{1}{2}\|\mathbf{s}_t\|^2_{\mathbf{F}_0} &= \frac{1}{2}\big(\mathbf{s}_t^\top \mathbf{F}_0 \mathbf{s}_t + 2\,\mathbf{s}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_t + \boldsymbol{\delta}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_t\big) - \frac{1}{2}\mathbf{s}_t^\top \mathbf{F}_0 \mathbf{s}_t \\ &= \frac{1}{2}\,\boldsymbol{\delta}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_t \;+\; \boldsymbol{\delta}_t^\top \mathbf{F}_0\, \mathbf{s}_t.\end{aligned}$$

Substituting $\mathbf{s}_t = \boldsymbol{\theta}_t - \boldsymbol{\theta}_0$:

$$\frac{1}{2}\|\mathbf{s}_{t+1}\|^2_{\mathbf{F}_0} - \frac{1}{2}\|\mathbf{s}_t\|^2_{\mathbf{F}_0} \;=\; \frac{1}{2}\,\boldsymbol{\delta}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_t \;+\; \boldsymbol{\delta}_t^\top \mathbf{F}_0 (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0).$$

Step 4: Summing over all steps. Summing from $t = 0$ to $T - 1$ and using the result of Step 1:

$$D_{\mathrm{KL}}(Q_T \| P) \;=\; \frac{1}{2}\|\mathbf{s}_T\|^2_{\mathbf{F}_0} \;=\; \sum_{t=0}^{T-1}\left[\frac{1}{2}\,\boldsymbol{\delta}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_t \;+\; \boldsymbol{\delta}_t^\top \mathbf{F}_0 (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0)\right],$$

which gives Eq. 10.

Step 5: Pairwise form of the cross terms and the alignment factor. Substituting $\boldsymbol{\theta}_t - \boldsymbol{\theta}_0 = \sum_{s=0}^{t-1} \boldsymbol{\delta}_s$ into each cross term gives $\boldsymbol{\delta}_t^\top \mathbf{F}_0 (\boldsymbol{\theta}_t - \boldsymbol{\theta}_0) = \sum_{s<t} \boldsymbol{\delta}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_s$, and summing over t yields Eq. 11. Dividing the result of Step 2 by $\sum_t \frac{1}{2}\|\boldsymbol{\delta}_t\|^2_{\mathbf{F}_0}$ gives Eq. 12; the range $\kappa_T \in [0, T]$ follows from positive semi-definiteness and from Step 6.

Step 6: Path-energy bound. Since $\mathbf{F}_0$ is positive semi-definite, $\|\cdot\|_{\mathbf{F}_0}$ is a seminorm and satisfies the triangle inequality:

$$\Big\|\sum_{t=0}^{T-1} \boldsymbol{\delta}_t\Big\|_{\mathbf{F}_0} \;\le\; \sum_{t=0}^{T-1} \|\boldsymbol{\delta}_t\|_{\mathbf{F}_0},$$

with equality when the vectors $\mathbf{F}_0^{1/2}\boldsymbol{\delta}_t$ are non-negative multiples of a common direction. Squaring and applying the Cauchy–Schwarz inequality, $\big(\sum_{t=0}^{T-1} a_t\big)^2 \le T \sum_{t=0}^{T-1} a_t^2$ with $a_t = \|\boldsymbol{\delta}_t\|_{\mathbf{F}_0}$ (equality when all a_t are equal), gives Eq. 13 after multiplying by $\frac{1}{2}$ and using Step 2.

Step 7: Local Fisher geometry. By the definition of $\bar{\gamma}$, $\boldsymbol{\delta}_t^\top \mathbf{F}_0 \boldsymbol{\delta}_t \le \bar{\gamma}\, \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t$ for every t. Substituting into the right-hand side of Eq. 13 gives Eq. 14. None of the steps uses the form of the update rule, so the bound holds for adaptive optimizers and for coupled or decoupled weight decay alike.

A.2 Layer-Wise KL Decomposition and Freezing

We decompose the Fisher-governed KL cost across layers and use the resulting curvature shifts as a proxy for adaptation relevance. Following standard structured approximations to the Fisher Information Matrix (Martens and Grosse, 2015), we use a block-diagonal decomposition across L layers:

$$\mathbf{F}_{\boldsymbol{\theta}} \approx \mathrm{diag}(\mathbf{F}_{\boldsymbol{\theta}_1}, \ldots, \mathbf{F}_{\boldsymbol{\theta}_L}). \tag{15}$$

This approximation enables the Fisher-weighted contribution to be decomposed layer-wise and is used only for tractable attribution of the global curvature cost.

Definition A.4 (Per-layer KL cost)**.** The per-iteration per-layer contribution to the KL bound in Eq. 14 is $c_{t,\ell} := \boldsymbol{\delta}_{t,\ell}^\top \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} \boldsymbol{\delta}_{t,\ell}$, and its cumulative contribution is $\mathcal{C}_\ell := \sum_{t=0}^{T-1} c_{t,\ell}$.

Proposition A.5 (Layer-wise decomposition of the KL bound)**.** *Under the layer-wise Fisher approximation in Eq. 15, the right-hand side of Eq. 14 decomposes as:*

$$D_{\mathrm{KL}}(Q_T \| P) \;\le\; \frac{\bar{\gamma}\, T}{2} \sum_{t=0}^{T-1} \boldsymbol{\delta}_t^\top \mathbf{F}_{\boldsymbol{\theta}_t} \boldsymbol{\delta}_t \;\approx\; \frac{\bar{\gamma}\, T}{2} \sum_{\ell=1}^{L} \mathcal{C}_\ell, \tag{16}$$

where the approximation is due solely to the block-diagonal structure. Since each $c_{t,\ell} = \boldsymbol{\delta}_{t,\ell}^\top \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} \boldsymbol{\delta}_{t,\ell} \ge 0$ (a quadratic form under a positive semi-definite matrix), every term contributes non-negatively to the bound.

Corollary A.6 (Freezing stabilized layers reduces the KL bound)**.** *If layer ℓ is frozen at iteration t^*, then $\boldsymbol{\delta}_{t,\ell} = \mathbf{0}$ and hence $c_{t,\ell} = 0$ for all $t \ge t^*$. Holding the updates of the remaining layers fixed, the layer-wise bound in Eq. 16 is reduced by $\Delta\mathcal{C}_\ell = \frac{\bar{\gamma} T}{2} \sum_{t=t^*}^{T-1} c_{t,\ell} \ge 0$, since each $c_{t,\ell} \ge 0$.*

A.3 Curvature Shift as an Adaptation Signal

Corollary A.6 shows that freezing a layer removes a non-negative contribution to the Fisher-weighted upper bound on KL divergence, holding the remaining updates fixed. This provides a curvature-based incentive for freezing, however, freezing may also suppress task adaptation. Therefore, the key question is *which layers can be frozen while preserving essential adaptation dynamics*. We posit that layers whose contributions have effectively *stabilized* are promising candidates. Importantly, stabilization concerns the *evolution* of the contribution rather than its absolute value. If a layer's contribution changes only marginally across iterations, $c_{t,\ell} - c_{t-1,\ell} \approx 0$, its curvature-weighted role is becoming more stable, motivating a closer examination of whether continued updates are needed. To characterize this behavior, we define the per-layer contribution drift:

$$c_{t,\ell} - c_{t-1,\ell} = \boldsymbol{\delta}_{t,\ell}^\top \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} \boldsymbol{\delta}_{t,\ell} - \boldsymbol{\delta}_{t-1,\ell}^\top \mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}} \boldsymbol{\delta}_{t-1,\ell}.$$

A.3.1 Local Update-Magnitude Stability

Assumption A.7 (Local update-magnitude stability)**.** Consecutive layer-wise updates have comparable magnitude in the local Fisher metric, $\|\boldsymbol{\delta}_{t,\ell}\|_{\mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}} \approx \|\boldsymbol{\delta}_{t-1,\ell}\|_{\mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}}$, where $|\mathbf{v}|_{\mathbf{A}}^2 := \mathbf{v}^\top \mathbf{A}\mathbf{v}$. Such local magnitude stability can arise during fine-tuning with small learning rates and adaptive or momentum-based optimizers, for which the scale of parameter updates may vary slowly across consecutive iterations. Importantly, comparable update magnitudes do not imply that adaptation has converged, rather the update direction and the local geometry experienced by the parameters may continue to evolve.

Distinguishing magnitude stability from convergence is important as a layer may continue receiving updates of similar magnitude while responding to an evolving task-specific geometry. Assumption A.7 therefore constrains only the *magnitude* of consecutive updates, and it does not require $\boldsymbol{\delta}_{t,\ell}$ and $\boldsymbol{\delta}_{t-1,\ell}$ to be close as vectors. The empirical assessment of Assumption A.7 is provided in Appendix A.5.

A.3.2 Derivation of the Contribution Drift

Under this local magnitude-stability condition, the temporal change in the per-layer contribution decomposes as:

$$c_{t,\ell} - c_{t-1,\ell} \;=\; \underbrace{\boldsymbol{\delta}_{t,\ell}^\top \Delta\mathbf{F}_{t,\ell} \boldsymbol{\delta}_{t,\ell}}_{\text{curvature shift}} \;+\; \underbrace{\left(\|\boldsymbol{\delta}_{t,\ell}\|^2_{\mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}} - \|\boldsymbol{\delta}_{t-1,\ell}\|^2_{\mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}} \right)}_{\text{magnitude term}}$$

where $\Delta\mathbf{F}_{t,\ell} := \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} - \mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}$. Therefore, under Assumption A.7:

Adaptation governed by FIM drift

$$c_{t,\ell} - c_{t-1,\ell} \;\approx\; \boldsymbol{\delta}_{t,\ell}^\top \Delta\mathbf{F}_{t,\ell} \boldsymbol{\delta}_{t,\ell}, \quad \Delta\mathbf{F}_{t,\ell} := \mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} - \mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}. \tag{17}$$

Equation 17 shows that, under local update-magnitude stability, the evolution of a layer's Fisher-weighted contribution is approximately governed by the *temporal variation of its Fisher information*. When the Fisher stabilizes, $\mathbf{F}_{\boldsymbol{\theta}_{t,\ell}} \approx \mathbf{F}_{\boldsymbol{\theta}_{t-1,\ell}}$, the contribution drift $c_{t,\ell} - c_{t-1,\ell}$ becomes small, even though the contribution itself may remain nonzero. This makes the layer a candidate for freezing: its curvature-weighted role changes only marginally, while continued updates still incur divergence cost. Conversely, persistent Fisher drift suggests that a layer's role in adaptation is still evolving and motivates keeping it trainable.

A.4 Quantifying Fisher Structural Drift with Jensen-Shannon Distance

To compare Fisher structural drift across layers and training stages, for each layer ℓ, we concatenate the Fisher values of its weights and biases into a single vector:

$$F_\ell = \{f_i\}_{i=1}^{N_\ell}. \tag{18}$$

To stabilize heavy-tailed magnitudes, we first transform their magnitudes to log scale:

$$\tilde{F}_\ell = \log_{10}(|F_\ell| + \varepsilon), \qquad \varepsilon = 10^{-12}. \tag{19}$$

To compare two consecutive Fisher states, we construct histograms using a shared set of B bins. The bin edges $\{b_k\}_{k=0}^{B}$ are determined from the joint 1st–99th percentile range of the two log-transformed vectors, ensuring aligned histogram support. The count in each bin is:

$$c_k = \#\left\{ i \;\middle|\; b_k \le \tilde{f}_i < b_{k+1} \right\}, \qquad k = 0, \dots, B-1, \tag{20}$$

where $\tilde{f}_i \in \tilde{F}_\ell$. To avoid zero-probability bins and ensure numerical stability in divergence computations, we apply additive smoothing and normalize:

$$p_k = \frac{c_k + \varepsilon}{\sum_{j=0}^{B-1}(c_j + \varepsilon)}, \qquad \sum_{k=0}^{B-1} p_k = 1. \tag{21}$$

The resulting probability mass function (PMF) for layer ℓ is:

$$p_\ell = (p_0, \dots, p_{B-1}) \in \mathbb{R}^B. \tag{22}$$

Distributional differences are then measured using the Jensen-Shannon (JS) distance. The JS divergence between two PMFs p and q is defined as:

$$\mathrm{JS}(p, q) = \frac{1}{2}\mathrm{KL}(p \parallel m) + \frac{1}{2}\mathrm{KL}(q \parallel m), \qquad m = \frac{1}{2}(p + q), \tag{23}$$

with the corresponding JS distance:

$$d_{\mathrm{JS}}(p, q) = \sqrt{\mathrm{JS}(p, q)}. \tag{24}$$

This symmetric and bounded distance measures redistribution in the Fisher-value distribution during fine-tuning. Since $p_{t,\ell}$ is a binned, one-dimensional summary of the diagonal Fisher entries, d_{JS} is a proxy for change in the Fisher tensor rather than an exact test of tensor equality. The following proposition formalizes the information captured, and omitted, by this proxy. Let $w = (b_B - b_0)/B$ denote the bin width on the $\log_{10}$ scale, and for two PMFs p, q let

$$\Delta(p, q) = \sum_{k=0}^{B-1} \frac{(p_k - q_k)^2}{p_k + q_k}$$

denote their triangular discrimination.

Proposition A.8 (JS distance as a proxy for Fisher structural drift)**.** *Let $p_{t,\ell}$ and $p_{t-1,\ell}$ be the PMFs of Eq. 22 constructed from $\mathbf{F}_\ell^t$ and $\mathbf{F}_\ell^{t-1}$ on the shared bin grid $\{b_k\}_{k=0}^{B}$. The JS distance $d_{\mathrm{JS}}(p_{t,\ell}, p_{t-1,\ell})$ satisfies:*

1. Distributional identity. *d_{JS} is a metric on PMFs, hence $d_{\mathrm{JS}}(p_{t,\ell}, p_{t-1,\ell}) = 0$ if and only if $p_{t,\ell} = p_{t-1,\ell}$. In particular, $d_{\mathrm{JS}} = 0$ whenever $\mathbf{F}_\ell^t = \mathbf{F}_\ell^{t-1}$. The converse does not hold: d_{JS} is unchanged by any permutation of the entries within the group and by any perturbation that moves no entry across a bin edge. Thus $d_{\mathrm{JS}} > 0$ certifies that the distribution of log-sensitivities has changed, whereas $d_{\mathrm{JS}} \approx 0$ is a necessary, but not sufficient, condition for a stationary Fisher tensor.*
2. Bounded sensitivity. *Let m be the number of entries whose bin assignment differs between the two states (counting entries that enter or leave $[b_0, b_B]$), let n_t, n_{t-1} be the in-range counts, and set $\rho = m/\max(n_t, n_{t-1})$. Then:*

$$d_{\mathrm{JS}}^2(p_{t,\ell}, p_{t-1,\ell}) \;\le\; \frac{\ln 2}{2}\,\Delta(p_{t,\ell}, p_{t-1,\ell}) \;\le\; \ln 2 \cdot \min(1, \rho). \tag{25}$$

3. Approximate scale invariance. *Suppose the normalized shape tensor $\mathbf{P}_\ell^t = \mathbf{F}_\ell^t / \|\mathbf{F}_\ell^t\|_F$ is unchanged, i.e., $\mathbf{F}_\ell^t = \alpha\,\mathbf{F}_\ell^{t-1}$ for some $\alpha > 0$, and let $\delta = |\log_{10}\alpha|$. Every log-transformed entry then shifts by at most δ, so only entries lying within δ of one of the $B+1$ bin edges can change bins, and Eq. 25 holds with $m \le \#\{i : \min_k |\tilde{f}_i^{\,t-1} - b_k| \le \delta\}$. Hence d_{JS} is not exactly invariant to uniform rescaling, but the scale-induced component is controlled by the ratio δ/w of the log-scale shift to the bin width.*

Consequently, a near-zero JS distance indicates a stable Fisher-value distribution up to within-bin and permutation ambiguity, whereas persistent JS distance that exceeds the scale-induced component of item 3 indicates continued redistribution of curvature. Under the local stability conditions underlying Eq. 17, this distance provides an operational proxy for whether a layer or parameter group continues to contribute to adaptation.

Proof. Item 1. The square root of the JS divergence is a metric on probability distributions (Endres and Schindelin, 2003), which gives the equivalence with $p_{t,\ell} = p_{t-1,\ell}$. The counts c_k depend on $\mathbf{F}_\ell$ only through the multiset $\{\tilde{f}_i\}$ and only through the bin containing each $\tilde{f}_i$, which gives the two invariances.

Item 2. The first inequality is the upper bound of Topsøe (2000), $2\,\mathrm{JS}(p,q) \le \ln 2 \cdot \Delta(p,q)$, with JS in nats. For the second, $|p_k - q_k| \le p_k + q_k$ gives $\Delta(p,q) \le \|p-q\|_1 \le 2$. Write $p_k = (c_k + \varepsilon)/Z$ and $q_k = (c'_k + \varepsilon)/Z'$ with $Z = n + B\varepsilon$, $Z' = n' + B\varepsilon$, and label the two states so that $Z \ge Z'$. From $p_k - q_k = (c_k - c'_k)/Z + (c'_k + \varepsilon)(1/Z - 1/Z')$ we obtain

$$\|p - q\|_1 \;\le\; \frac{\sum_k |c_k - c'_k| + |n - n'|}{Z}. \tag{26}$$

An entry that moves between two in-range bins contributes 2 to $\sum_k |c_k - c'_k|$ and 0 to $|n - n'|$; an entry that enters or leaves $[b_0, b_B]$ contributes at most 1 to each. The numerator is therefore at most $2m$, so $\|p-q\|_1 \le 2m/Z \le 2\rho$, and $\frac{\ln 2}{2}\Delta \le \ln 2 \cdot \min(1, \rho)$.

Item 3. $\mathbf{P}_\ell^t = \mathbf{P}_\ell^{t-1}$ holds if and only if $\mathbf{F}_\ell^t = \alpha \mathbf{F}_\ell^{t-1}$ with $\alpha = \|\mathbf{F}_\ell^t\|_F / \|\mathbf{F}_\ell^{t-1}\|_F$. For $\alpha \ge 1$ and any f_i, $0 \le \log_{10}(\alpha|f_i| + \varepsilon) - \log_{10}(|f_i| + \varepsilon) \le \log_{10}\alpha$, and symmetrically for $\alpha < 1$, so each $\tilde{f}_i$ moves by at most δ. An entry changes its bin assignment only if a bin edge lies between its old and new position, which requires the entry to be within δ of that edge. The result then follows directly from item 2. □

A.5 Empirical Assessment of the Theoretical Assumptions

The experimental protocol and metric definitions for the empirical assessment of Assumptions A.1 and A.7 are reported in Figure 12. We analyze one full fine-tuning trajectory for each of CIFAR-100, DTD, SVHN, and EuroSAT from VTAB-1k. For each task, a pretrained DINOv2 ViT-S/14 is fine-tuned for 50 epochs (600 optimizer steps) using AdamW with batch size 64, a peak learning rate of 5×10^{-5}, linear warmup followed by cosine decay without weight decay. Update and Fisher diagnostics are recorded from the same training trajectory.

Update-magnitude stability. We empirically assess this behavior in Figure 12. As a tractable surrogate for variation in the Fisher-weighted update magnitude, we measure the relative change in Euclidean update norm: $\frac{\left|\|\boldsymbol{\delta}_{t,\ell}\|_2 - \|\boldsymbol{\delta}_{t-1,\ell}\|_2\right|}{\|\boldsymbol{\delta}_{t-1,\ell}\|_2}$. Across four VTAB-1k tasks, its median ranges from 0.045 to 0.073, indicating gradually varying update magnitudes. However, consecutive updates are not directionally identical, with mean cosine similarities of 0.87-0.89. Update-magnitude stability alone therefore does not imply that adaptation has stabilized, motivating a signal that captures changes in task-specific geometry.

For parameter tensor ℓ we define the update at iteration t as $\boldsymbol{\delta}_{t,\ell} = \boldsymbol{\theta}_{t+1,\ell} - \boldsymbol{\theta}_{t,\ell}$ and report the relative change in its magnitude:

$$\rho_{t,\ell} \;:=\; \frac{\big|\,\|\boldsymbol{\delta}_{t,\ell}\|_2 - \|\boldsymbol{\delta}_{t-1,\ell}\|_2\,\big|}{\|\boldsymbol{\delta}_{t-1,\ell}\|_2} \;=\; |r_{t,\ell} - 1|, \qquad r_{t,\ell} \;:=\; \frac{\|\boldsymbol{\delta}_{t,\ell}\|_2}{\|\boldsymbol{\delta}_{t-1,\ell}\|_2}, \tag{27}$$

where the ratio $r_{t,\ell}$ is retained as shorthand for use below. The quantity $\rho_{t,\ell}$ is the Euclidean surrogate for the Fisher-metric ratio constrained by Assumption A.7. Across the four tasks the median of $\rho_{t,\ell}$ is 0.046 (CIFAR-100), 0.067 (DTD), 0.045 (SVHN) and 0.073 (EuroSAT), with 90th percentiles of 0.13-0.19 and no upward trend over training.

We additionally report the directional diagnostic $s_{t,\ell} = \boldsymbol{\delta}_{t,\ell}^\top \boldsymbol{\delta}_{t-1,\ell} / (\|\boldsymbol{\delta}_{t,\ell}\|_2 \|\boldsymbol{\delta}_{t-1,\ell}\|_2)$, which their mean is 0.87-0.89, corresponding to a median rotation of $24°$-$28°$ per optimizer step. The two

diagnostics combine through the exact identity:

$$\frac{\|\boldsymbol{\delta}_{t,\ell} - \boldsymbol{\delta}_{t-1,\ell}\|_2^2}{\|\boldsymbol{\delta}_{t-1,\ell}\|_2^2} = \underbrace{\rho_{t,\ell}^2}_{\text{radial}} + \underbrace{2r_{t,\ell}\big(1 - s_{t,\ell}\big)}_{\text{angular}}, \tag{28}$$

in which the two terms have medians of 0.045-0.073 (radial) and 0.41-0.47 (angular) respectively. Figure 12 (top) reports the two underlying diagnostics directly, $\rho_{t,\ell}$ and $s_{t,\ell}$, rather than the terms of Eq. 28. Note that these are opposite in sense ($s_{t,\ell} \to 1$ and $\rho_{t,\ell} \to 0$ both denote an unchanged update), so the vertical separation between the two curves in that panel carries no meaning. We emphasize that Assumption A.7 does *not* require consecutive updates to be close as vectors. The total relative discrepancy has a median of 0.41-0.47 on these trajectories, and is dominated by the angular part (0.41-0.47) rather than the radial part (0.045-0.073), a factor of 6-10. The magnitude term in Eq. 17 depends only on the radial part, which is why Assumption A.7 is stated as a constraint on update scale and not on update direction.

Local Fisher-scale stability. We empirically assess this assumption during DINOv2 ViT-S/14 fine-tuning on four VTAB-1k tasks (Figure 12). We approximate the FIM using AdaFisher's diagonal Kronecker form (Gomes et al., 2024):

$$\tilde{\mathbf{F}}_\ell^D = \mathbf{H}_{D_{\ell-1}} \otimes \mathbf{S}_{D_\ell}, \tag{29}$$

where $\mathbf{H}_{D_{\ell-1}}$ and $\mathbf{S}_{D_\ell}$ are diagonal activation and gradient correlation factors, yielding linear computational and memory complexity. We measure scale variation by the relative change in mean absolute Fisher magnitude and structural change by the Jensen-Shannon (JS) distance between consecutive Fisher-induced distributions. Metric definitions are given below, and the JS construction is detailed in Appendix A.4. Across all four tasks, Fisher-scale change is only 0.10-0.17%, while structural drift persists, supporting locally stable scale without stationary Fisher structure.

To examine the slowly varying Fisher-scale assumption, we define the mean absolute Fisher magnitude of tensor ℓ at iteration t as:

$$\bar{F}_{t,\ell} = \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} |F_\ell^t(i)|, \tag{30}$$

and its relative temporal change as:

$$r_{t,\ell} = \frac{|\bar{F}_{t,\ell} - \bar{F}_{t-1,\ell}|}{\bar{F}_{t-1,\ell} + \varepsilon}. \tag{31}$$

The Fisher is estimated using the efficient diagonal Kronecker approximation proposed by AdaFisher Gomes et al. (2024), collecting it every five optimizer steps. We additionally compute the JS distance between consecutive Fisher-value distributions, which is robust to uniform rescaling (Appendix A.4), to separate changes in Fisher *structure* from changes in overall *scale*. Both quantities are smoothed using an exponential moving average with decay 0.8, and Figure 12 reports their cross-layer means.

B FisherAdapTune: Algorithm and Implementation

B.1 Parameter Grouping

Algorithm 2 expands the main-text Algorithm 1. FisherAdapTune operates on parameter groups rather than freezing an entire tensor at once. For a matrix-shaped parameter tensor $\mathbf{W}_i \in \mathbb{R}^{d_i^{\text{out}} \times d_i^{\text{in}}}$, we divide its columns into K contiguous, non-overlapping blocks. Let

$$b_k = \left\lfloor \frac{k\, d_i^{\text{in}}}{K} \right\rfloor, \qquad k = 0, \ldots, K.$$

The kth group is then defined as

$$\mathbf{W}_{ik} = \mathbf{W}_i[:,\, b_{k-1} : b_k], \qquad k = 1, \ldots, K,$$

with the corresponding Fisher block

$$\tilde{\mathbf{F}}_{\mathbf{W}_{ik}}^t = \tilde{\mathbf{F}}_{D_i}^t[:,\, b_{k-1} : b_k].$$

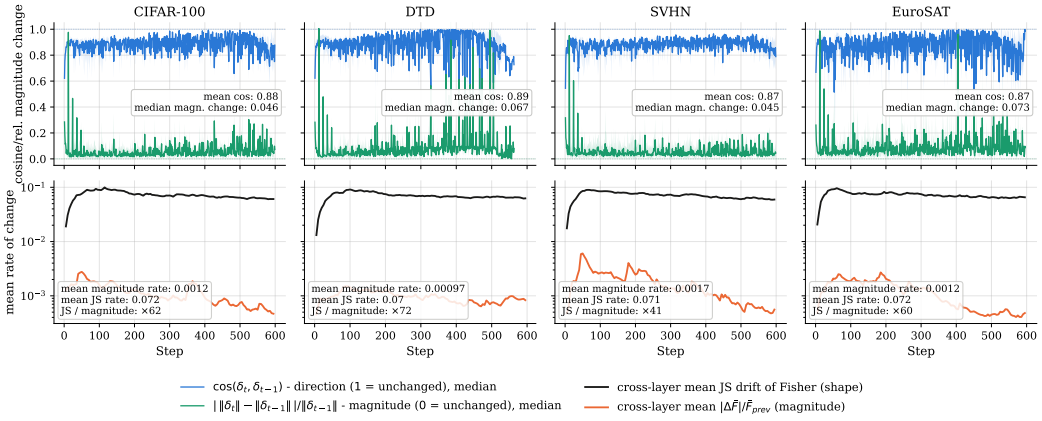


Figure 12: Empirical validation of Assumptions A.1 and A.7 during full fine-tuning of DINOv2 ViT-S/14 on four VTAB-1k tasks. *Top:* relative change in update magnitude and cosine similarity between consecutive updates (median and 10-90th percentiles across parameter tensors). Update magnitude remains stable (median 0.045-0.073), while showing modest directional variation. *Bottom:* relative Fisher-scale change and structural Fisher drift. Fisher scale varies by only 0.10-0.17%, while structural drift is 41-$72\times$ larger. See Appendix A.5 for more details.

Thus, every group contains all output rows but only a subset of the input columns. The blocks collectively cover the tensor without overlap, and their numbers of columns differ by at most one when d_i^{in} is not divisible by K. Convolutional kernels are first viewed in matrix form by flattening the input-channel and spatial dimensions, while one-dimensional tensors are partitioned into contiguous segments using the same rule. Each resulting block is treated as an independent member (i, k) of the active set $\mathcal{S}$ and may therefore be frozen at a different training step.

The block count K controls the granularity of parameter selection. A small K produces coarse groups and lower monitoring overhead, whereas a larger K permits more localized freezing at the cost of maintaining more Fisher distributions and drift statistics. Setting $K = 1$ reduces the procedure to tensor-wise selection. In practice, the effective number of blocks for a dimension of size d is $\min(K, d)$, preventing empty groups.

B.2 Drift Smoothing and Adaptive Freezing

The EMA factor $\gamma \in [0, 1)$ controls the temporal smoothing of the AdaFisher activation and gradient statistics. Larger values yield more stable Fisher estimates while responding more slowly to changes during fine-tuning. The second EMA factor $\beta \in [0, 1)$ independently smooths the consecutive Jensen–Shannon (JS) drift values, where larger β suppresses short-term fluctuations, and smaller β makes the freezing criterion more responsive. The Fisher interval m specifies how often the Fisher statistics and group-wise JS drift are updated. The freeze interval n specifies how often the active groups are evaluated for freezing, with $m < n$ allowing multiple Fisher observations to be collected between consecutive freezing decisions. At each freezing step, the threshold scale λ determines the data-adaptive cutoff:

$$\tau = \mu + \lambda\sigma,$$

where μ and σ are computed from the running mean JS drifts of the currently active groups. Groups with $\bar{d}_{\text{JS},ik} < \tau$ are regarded as stabilized since their Fisher distributions change relatively little, and are removed from $\mathcal{S}$. Decreasing λ lowers the threshold and therefore yields more conservative freezing, whereas increasing λ raises the threshold and freezes more groups. Finally, $T_{\max}$ is the total training budget and η is the optimizer learning rate. Once a group is removed from $\mathcal{S}$, its gradient update is masked for all subsequent iterations, while the remaining active groups continue to be optimized (Appendix B.4 details when masked tensors are released).

B.3 Complete Algorithm

Algorithm 2 FisherAdapTune Algorithm

Require: Parameters $\theta = \{\theta_i\}_{i=1}^{L}$, dataset $\mathcal{D}$, EMA factors γ, β, threshold scale λ, slice count K, Fisher interval m, freeze interval n, total steps $T_{\max}$.

Initialize active parameter groups set $\mathcal{S}$ to all supported groups.
for $t = 1$ **to** $T_{\max}$ **do**
  Compute loss $\mathcal{L}(\mathcal{D}; \theta)$ and gradients $\nabla_\theta \mathcal{L}$.
  if $t \bmod m = 0$ **then**
    Update AdaFisher activation and gradient statistics:

$$\mathbf{H}^t_{D_{i-1}} \leftarrow \gamma \mathbf{H}^{t-1}_{D_{i-1}} + (1-\gamma)\widehat{\mathbf{H}}^t_{D_{i-1}}, \quad \mathbf{S}^t_{D_i} \leftarrow \gamma \mathbf{S}^{t-1}_{D_i} + (1-\gamma)\widehat{\mathbf{S}}^t_{D_i}.$$

    Estimate layer Fisher tensors $\tilde{\mathbf{F}}^t_{D_i} = \mathbf{H}^t_{D_{i-1}} \otimes \mathbf{S}^t_{D_i}$.
    Partition each $\tilde{\mathbf{F}}^t_{D_i}$ into K column-wise parameter groups $\{\tilde{\mathbf{F}}^t_{w_{ik}}\}_k$.
    Build PMFs $p^t_{ik} = p(\tilde{\mathbf{F}}^t_{w_{ik}})$ using Eqs. 19-22.
    Compute consecutive Fisher drift:

$$\tilde{d}^t_{\mathrm{JS},ik} = d_{\mathrm{JS}}(p^{t-1}_{ik}, p^t_{ik}) = \sqrt{\tfrac{1}{2}\mathrm{KL}(p^{t-1}_{ik} \| M^t_{ik}) + \tfrac{1}{2}\mathrm{KL}(p^t_{ik} \| M^t_{ik})}, \quad M^t_{ik} = \tfrac{1}{2}(p^{t-1}_{ik} + p^t_{ik}).$$

    Smooth and accumulate the drift: $\hat{d}^t_{\mathrm{JS},ik} \leftarrow \beta \hat{d}^{t-1}_{\mathrm{JS},ik} + (1-\beta)\tilde{d}^t_{\mathrm{JS},ik}$, $\mathcal{J}_{ik} \leftarrow \mathcal{J}_{ik} \cup \{\hat{d}^t_{\mathrm{JS},ik}\}$.
    Compute running mean $\bar{d}_{\mathrm{JS},ik} \leftarrow |\mathcal{J}_{ik}|^{-1} \sum_{d \in \mathcal{J}_{ik}} d$.
  if $t \bmod n = 0$ **then**
    Compute global drift statistics over active parameter groups: $\mu \leftarrow \mathrm{mean}_{(i,k)\in\mathcal{S}}(\bar{d}_{\mathrm{JS},ik})$, $\sigma \leftarrow \mathrm{std}_{(i,k)\in\mathcal{S}}(\bar{d}_{\mathrm{JS},ik})$.
    Set $\tau \leftarrow \mu + \lambda\sigma$ and freeze stabilized parameter groups: $\mathcal{S} \leftarrow \{(i,k) \in \mathcal{S} : \bar{d}_{\mathrm{JS},ik} \geq \tau\}$.
  Update only active parameter groups: $\theta_{\mathcal{S}} \leftarrow \theta_{\mathcal{S}} - \eta \nabla_{\theta_{\mathcal{S}}} \mathcal{L}(\mathcal{D}; \theta)$.
end for

B.4 Training-Time and Memory Overhead

Figure 13 reports per-step cost and peak memory on CIFAR-100, as well as taskwise training-loop times. Fisher statistics and JS distances are computed periodically. Until all K blocks of a tensor are frozen, block-level masking zeros the corresponding gradients and optimizer moments, while autograd continues to process the tensor. Once all blocks are frozen, the tensor is set to non-trainable, releasing its gradient computation, saved activations, and optimizer state. With DINOv2 ViT-S/14, at m=2, 142 tensors containing 10.2M of 22.1M parameters are fully frozen by the end of the timed region. Without tensor-level release, peak memory is 5.73 GB.

With m=2, FisherAdapTune requires $1.14\times$ the per-step time of full fine-tuning while reducing steady-state peak memory from 5.56 GB to 3.97 GB. Its lifetime peak is 5.78 GB since training begins with all parameters active. Increasing the collection interval to m=8 reduces the per-step ratio to $1.06\times$, comparable to AutoFreeze, while maintaining a 4.13 GB steady-state peak. Performance remains stable across the evaluated collection intervals (Appendix E.2). Taskwise training-loop times are also reported, dependent on both per-step cost and the early-stopping epoch. The per-step measurements therefore provide the more controlled comparison of computational overhead.

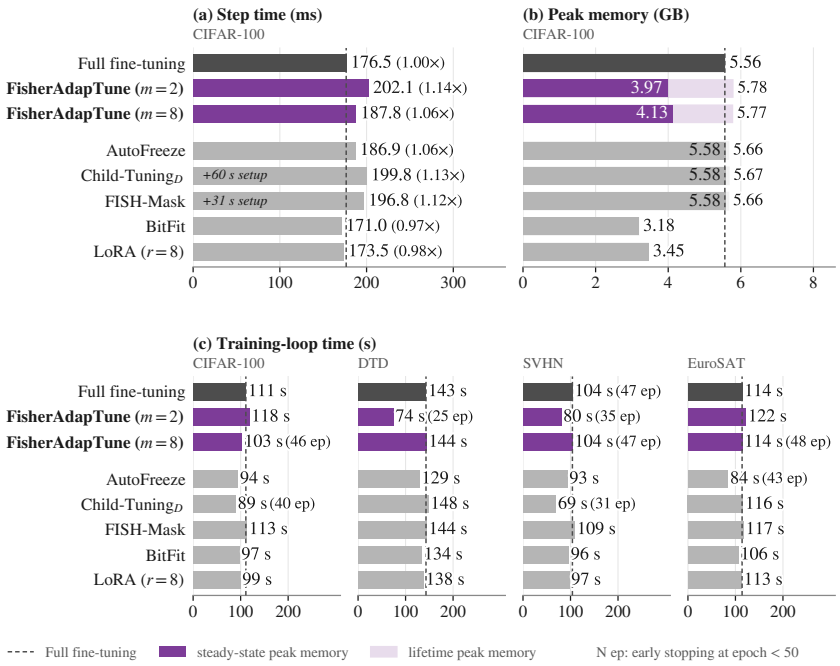


Figure 13: Training-time and memory overhead with DINOv2 ViT-S/14 (batch size 64, one RTX 6000 Ada). **(a)** CUDA-synchronized per-step time on CIFAR-100, excluding validation; one-time setup costs are annotated separately and excluded from step timing. **(b)** Steady-state and lifetime peak allocated memory on CIFAR-100; the lifetime peak includes the initial all-trainable phase. **(c)** Taskwise training-loop time, excluding setup, validation, and evaluation.

C Experimental Setup

C.1 VTAB-1k Classification

The VTAB-1k evaluation comprises all 19 tasks across its three categories: seven natural-image tasks, four specialized tasks, and eight structured tasks. Our objective is to compare adaptation strategies under a common experimental protocol. We report Top-1 accuracy for each task, with macro results computed as the unweighted mean across all 19 tasks. We use a DINOv2 ViT-S/14 backbone pretrained on LVD-142M (`timm` model `vit_small_patch14_dinov2.lvd142m`) and initialize a task-specific linear classification head for each dataset. Following the low-data VTAB-1k setting, each task uses 800 training samples, with a held-out or official validation split for model selection and early stopping. Evaluation is performed on the full official test split when available, and on the held-out remainder otherwise. All inputs are processed at 224×224 resolution and normalized using ImageNet statistics. During training, we apply random resized cropping, horizontal flipping, and color jitter; validation and test images are resized and center-cropped.

Training minimizes cross-entropy loss with label smoothing of 0.1 using AdamW with a learning rate of 5×10^{-5}, weight decay of 10^{-2}, and batch size 64 for up to 50 epochs. The learning rate is linearly warmed up for the first 5 epochs and subsequently cosine-decayed. Early stopping is applied when the validation loss does not improve for 10 consecutive epochs. Unless otherwise specified in the ablation studies, FisherAdapTune uses k=4, m=2, n=6, β=0.8, and λ= -0.65. Full fine-tuning and FisherAdapTune share the same data pipeline, task-head initialization, and optimization recipe. FisherAdapTune differs only through its progressive restriction of the trainable backbone parameters. Experiments were run on Digital Research Alliance of Canada NVIDIA H100 GPUs and local NVIDIA RTX 6000 Ada Generation GPUs.

C.2 Crack Segmentation

The OmniCrack30k dataset contains approximately 30,000 images aggregated from more than 20 crack datasets and covers concrete, asphalt, ceramic, masonry, and steel surfaces. Its held-out test set is used for in-distribution evaluation. Zero-shot generalization is evaluated on Concrete3k, Facade390, and Road420, which represent distinct acquisition conditions and surface domains. We report F1-score and IoU for each dataset and their unweighted mean across the three zero-shot datasets. All crack-segmentation experiments use AdamW with a cosine learning-rate schedule and batch size 2. We use a learning rate of 10^{-6} for SAM2 and 10^{-4} for SegFormer, and optimize the sum of BCE-with-logits and Dice losses. Images and masks are resized to 256×256 for both training and evaluation. Within each backbone, FisherAdapTune uses the same initialization, data splits, optimizer, loss, and schedule as the baselines. FisherAdapTune differs only by progressively removing parameter groups from the optimizer once their smoothed Fisher JS drift satisfies the adaptive freezing criterion.

D Comparative Results

D.1 In-Distribution and Zero-Shot Segmentation Results

Table 4 and Figure 14 compare fine-tuning methods across four backbones on crack segmentation. FisherAdapTune provides a strong balance between in-distribution performance, zero-shot generalization, and parameter efficiency. It consistently outperforms matched-budget random selection and achieves the highest average zero-shot performance for SegFormer-MiT-B3 while remaining competitive across the other backbones. Table 5 further reports three-seed results for FisherAdapTune and matched-budget random selection. FisherAdapTune consistently improves in-distribution performance and exhibits substantially lower variability. Its zero-shot gains are strongest for SegFormer, while remaining comparable to random selection for SAM2, confirming that Fisher-guided selection is stable across seeds and more effective than sparsity alone.

Table 4: Performance comparison of fine-tuning methods for crack segmentation in terms of in-distribution performance (OmniCrack test) and zero-shot generalization. Bold entries show the best per column within each model block; underline shows the second best. Effect. Params denotes number of Effective Tunable Parameters.

Model	Fine-tune Method	Effect. Params	OmniCrack		Zero-shot F1 (%)				Zero-shot IoU (%)			
			F1	IoU	Conc.	Fac.	Road	Avg.±Std	Conc.	Fac.	Road	Avg.±Std
SAM2-Tiny	Freeze	0.0M	4.37	2.24	8.21	4.13	4.47	$5.60_{\pm 2.27}$	4.28	2.11	2.28	$2.89_{\pm 1.21}$
	Full FT	38.9M	**64.67**	**47.79**	**73.34**	70.39	64.32	$69.35_{\pm 4.60}$	**57.91**	54.31	47.40	$53.21_{\pm 5.34}$
	LoRA	0.28M	56.79	39.65	69.07	67.51	58.76	$65.11_{\pm 5.56}$	52.75	50.96	41.60	$48.44_{\pm 5.99}$
	BitFit	0.11M	57.85	40.69	70.84	69.17	67.33	$69.11_{\pm 1.76}$	54.85	52.87	50.74	$52.82_{\pm 2.05}$
	LayerNorm	0.03M	50.06	33.39	64.15	65.38	61.16	$63.56_{\pm 1.77}$	47.23	48.56	44.05	$46.61_{\pm 1.89}$
	Decoder-only	4.2M	58.53	41.48	69.26	69.36	64.35	$67.66_{\pm 2.86}$	52.98	53.09	47.44	$51.17_{\pm 3.23}$
	Random	0.3M	48.30	31.84	56.00	65.36	58.93	$60.09_{\pm 3.91}$	38.88	48.54	41.77	$43.07_{\pm 4.05}$
	Random	1.0M	55.14	38.07	66.41	69.44	61.88	$65.91_{\pm 3.11}$	49.71	53.19	44.80	$49.24_{\pm 3.44}$
	Random	6.95M	57.57	40.42	69.41	70.83	63.15	$67.80_{\pm 3.34}$	53.15	54.83	46.14	$51.38_{\pm 3.76}$
	AutoFreeze	21.40M	59.25	42.10	72.41	**73.48**	67.75	$\mathbf{71.21_{\pm 3.05}}$	56.76	**58.08**	51.23	$\mathbf{55.36_{\pm 3.63}}$
	Child-Tuning$_D$	9.67M	58.43	41.27	70.07	71.45	**69.20**	$70.24_{\pm 1.13}$	53.93	55.58	**52.91**	$54.14_{\pm 1.35}$
	FISH-Mask	0.19M	57.01	39.87	67.67	66.73	60.67	$65.02_{\pm 3.80}$	51.14	50.07	43.54	$48.25_{\pm 4.11}$
	FisherAdapTune	6.95M	63.12	46.11	72.43	69.10	64.42	$68.65_{\pm 4.02}$	56.77	52.79	47.52	$52.36_{\pm 4.64}$
SAM2-Large	Freeze	0.0M	3.54	1.80	6.29	2.41	3.93	$4.21_{\pm 1.95}$	3.25	1.22	2.00	$2.16_{\pm 1.02}$
	Full FT	224M	**67.86**	**51.35**	**74.03**	64.25	70.39	$69.56_{\pm 5.00}$	**58.77**	47.33	54.31	$53.47_{\pm 5.8}$
	LoRA	0.221M	54.54	37.49	59.72	62.82	51.81	$58.11_{\pm 5.68}$	42.57	45.79	34.96	$41.11_{\pm 5.56}$
	BitFit	0.41M	57.35	40.20	73.05	70.13	71.55	$71.58_{\pm 1.19}$	57.54	54.00	55.70	$55.75_{\pm 1.44}$
	LayerNorm	0.12M	43.33	27.66	50.29	62.75	61.99	$58.34_{\pm 5.70}$	33.59	45.72	44.92	$41.41_{\pm 5.54}$
	Decoder-only	4.2M	56.28	39.16	71.26	68.61	61.71	$67.19_{\pm 4.93}$	55.35	52.22	44.62	$50.73_{\pm 5.52}$
	Random	71.49M	63.36	46.37	73.90	70.35	71.79	$72.01_{\pm 1.46}$	58.60	54.26	55.99	$56.29_{\pm 1.79}$
	AutoFreeze	90.28M	59.12	41.97	71.66	74.54	72.73	$72.98_{\pm 1.46}$	55.83	59.42	57.14	$57.46_{\pm 1.82}$
	Child-Tuning$_D$	46.76M	60.98	43.86	72.11	**76.66**	**74.47**	$\mathbf{74.41_{\pm 2.28}}$	56.38	**62.15**	**59.33**	$\mathbf{59.29_{\pm 2.89}}$
	FISH-Mask	1.12M	50.60	33.87	67.98	67.94	72.79	$69.57_{\pm 2.79}$	51.50	51.45	57.22	$53.39_{\pm 3.32}$
	FisherAdapTune	71.49M	65.78	49.01	72.81	75.30	70.29	$72.80_{\pm 2.04}$	57.24	60.38	54.19	$57.27_{\pm 2.53}$
SegFormer -MiT-B0	Full FT	3.71M	**63.35**	**46.36**	73.71	57.14	**54.00**	$\mathbf{61.61_{\pm 8.64}}$	58.37	40.00	**36.98**	$\mathbf{45.11_{\pm 9.45}}$
	LoRA	0.23M	58.83	41.68	71.85	55.38	44.31	$57.18_{\pm 11.31}$	56.06	38.29	28.46	$40.94_{\pm 11.42}$
	BitFit	0.02M	44.99	29.02	61.86	45.95	43.23	$50.34_{\pm 8.21}$	44.79	29.83	27.57	$34.06_{\pm 7.64}$
	LayerNorm	0.01M	45.56	29.50	67.07	44.07	42.41	$51.18_{\pm 11.25}$	50.46	28.26	26.91	$35.21_{\pm 10.79}$
	Decoder-only	0.40M	55.11	38.04	63.47	54.91	52.58	$56.98_{\pm 4.68}$	46.48	37.85	35.66	$40.00_{\pm 4.67}$
	Random	0.12M	52.08	35.21	63.81	51.33	52.45	$55.87_{\pm 5.64}$	46.85	34.53	35.55	$38.98_{\pm 5.58}$
	AutoFreeze	1.43M	52.82	35.89	**74.53**	49.37	53.60	$59.17_{\pm 13.47}$	**59.40**	32.77	36.61	$42.93_{\pm 14.39}$
	Child-Tuning$_D$	1.39M	57.85	40.70	71.60	51.93	51.11	$58.21_{\pm 11.60}$	55.77	35.08	34.33	$41.73_{\pm 12.17}$
	FISH-Mask	0.02M	52.63	35.71	73.18	49.46	44.30	$55.65_{\pm 15.40}$	57.70	32.85	28.46	$39.67_{\pm 15.77}$
	FisherAdapTune	0.12M	60.85	43.73	70.97	**58.77**	53.63	$61.12_{\pm 7.27}$	55.01	**41.62**	36.64	$44.42_{\pm 7.75}$
SegFormer -MiT-B3	Full FT	44.60M	**64.99**	**48.14**	73.20	**61.31**	56.42	$63.64_{\pm 7.04}$	57.72	**44.20**	39.30	$47.07_{\pm 7.78}$
	LoRA	1.79M	54.88	37.82	75.47	52.39	39.53	$55.79_{\pm 14.87}$	60.60	35.49	24.63	$40.24_{\pm 15.06}$
	BitFit	0.14M	51.73	34.89	68.79	52.04	53.53	$58.12_{\pm 7.56}$	52.42	35.17	36.55	$41.38_{\pm 7.82}$
	LayerNorm	0.05M	50.33	33.63	63.24	50.01	48.79	$54.01_{\pm 6.54}$	46.24	33.34	32.27	$37.28_{\pm 6.34}$
	Decoder-only	0.53M	57.83	40.67	66.28	56.71	**58.72**	$60.57_{\pm 4.12}$	49.57	39.57	**41.57**	$43.57_{\pm 4.32}$
	Random	10.73M	62.15	45.09	72.16	58.56	42.96	$57.89_{\pm 11.93}$	56.45	41.40	27.36	$41.74_{\pm 11.88}$
	AutoFreeze	22.72M	64.29	47.38	**76.71**	53.58	58.53	$62.94_{\pm 12.18}$	**62.22**	36.59	41.37	$46.73_{\pm 13.63}$
	Child-Tuning$_D$	9.34M	64.67	47.79	67.71	57.87	52.19	$59.26_{\pm 7.85}$	51.18	40.72	35.31	$42.40_{\pm 8.07}$
	FISH-Mask	0.22M	61.63	44.54	73.12	54.36	51.02	$59.50_{\pm 11.91}$	57.63	37.32	34.25	$43.07_{\pm 12.71}$
	FisherAdapTune	10.73M	64.57	47.68	74.19	61.04	58.26	$\mathbf{64.49_{\pm 6.94}}$	58.97	43.93	41.11	$\mathbf{48.00_{\pm 7.83}}$

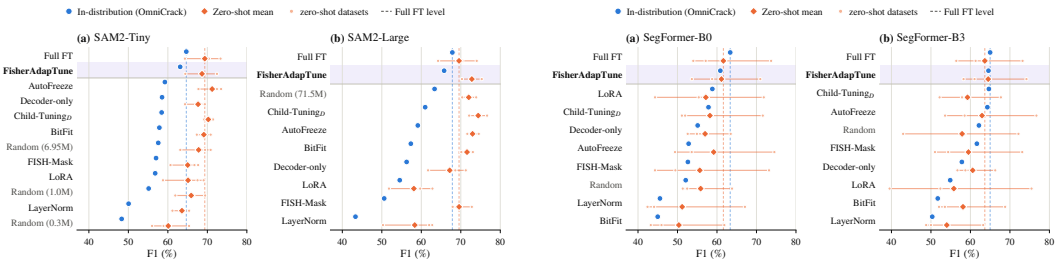


Figure 14: Comparison of in-distribution and zero-shot crack-segmentation performance across fine-tuning methods for SAM2 and SegFormer architectures.

Table 5: Comparison of FisherAdapTune and matched-budget random selection across four crack-segmentation experiments using three random seeds. In-distribution results are reported on Omni-Crack30K, and zero-shot results are averaged across Concrete3k, Facade390, and Road420. Values denote mean ± standard deviation across seeds.

	In-distribution		Zero-shot mean	
Method	F1	IoU	F1	IoU
SegFormer-B0				
FisherAdapTune ($\lambda = -0.95$)	$61.21_{\pm 0.48}$	$44.11_{\pm 0.50}$	$61.24_{\pm 1.40}$	$44.55_{\pm 1.36}$
Random subset (0.124M)	$43.38_{\pm 18.64}$	$28.84_{\pm 14.40}$	$49.35_{\pm 12.45}$	$34.37_{\pm 9.51}$
SegFormer-B3				
FisherAdapTune ($\lambda = -0.95$)	$64.41_{\pm 0.14}$	$47.51_{\pm 0.15}$	$64.40_{\pm 0.90}$	$48.03_{\pm 0.95}$
Random subset (10.73M)	$63.84_{\pm 1.71}$	$46.91_{\pm 1.84}$	$61.40_{\pm 3.26}$	$44.98_{\pm 3.04}$
SAM2-Tiny				
FisherAdapTune ($\lambda = -0.85$)	$63.20_{\pm 0.16}$	$46.20_{\pm 0.17}$	$68.88_{\pm 0.21}$	$52.65_{\pm 0.25}$
Random subset (6.95M)	$60.70_{\pm 2.71}$	$43.61_{\pm 2.76}$	$68.82_{\pm 0.93}$	$52.54_{\pm 1.06}$
SAM2-Large				
FisherAdapTune ($\lambda = -0.87$)	$66.02_{\pm 0.31}$	$49.28_{\pm 0.34}$	$72.63_{\pm 0.16}$	$57.06_{\pm 0.19}$
Random subset (71.49M)	$63.47_{\pm 0.58}$	$46.49_{\pm 0.62}$	$72.83_{\pm 0.75}$	$57.32_{\pm 0.94}$

E Ablation Studies and Hyperparameter Sensitivity

E.1 Comparison of Parameter-Selection Signals

To assess the selection signal, we replace Fisher structural drift with Fisher magnitude, Fisher-magnitude drift, gradient RMS, update RMS, and random selection while keeping the remaining algorithm unchanged. For parameter group g with d_g parameters, gradient RMS is defined as $\sqrt{d_g^{-1} \sum_{i=1}^{d_g} (\partial \mathcal{L} / \partial \theta_{g,i})^2}$, update RMS replaces the gradients with the corresponding parameter updates $\Delta\theta_{g,i}$. Figure 16 monitors these signals across DINOv2 layers during fine-tuning on CIFAR-100. Fisher structural drift separates groups whose Fisher distributions continue to change from groups whose drift approaches zero. Fisher magnitude also differs across layers, while its ordering is largely persistent. Fisher-magnitude drift is small for most groups apart from transient peaks. Gradient RMS is concentrated in a few layers yet declines broadly during training, while update RMS falls similarly and the group trajectories converge. Thus, variation across layers alone does not establish that a signal can distinguish *continued adaptation* from stabilization over time. Each alternative is reported at the tuned λ where trainable budget comes closest to that of Fisher structural drift. For some signals, no λ produces a close match and further adjustment retains all groups as trainable instead of yielding finer progressive freezing. Figure 15 shows the resulting budget trajectories, and Table 3 reports their performance across eight VTAB-1k tasks. Taken together, these results support Fisher structural drift as a useful selection signal beyond the effect of reducing the trainable budget.

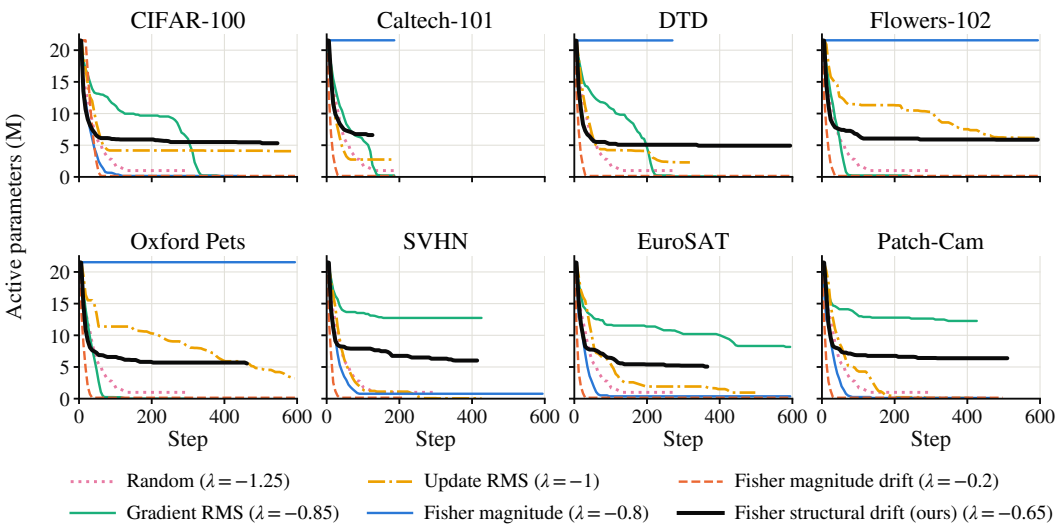


Figure 15: Evolution of the active trainable-parameter budget across freezing events for each selection signal on the eight VTAB-1k tasks. Each alternative is shown at the tuned λ where the parameter budget is closest to that of Fisher structural drift, with all other components of FisherAdapTune kept fixed (Table 3).

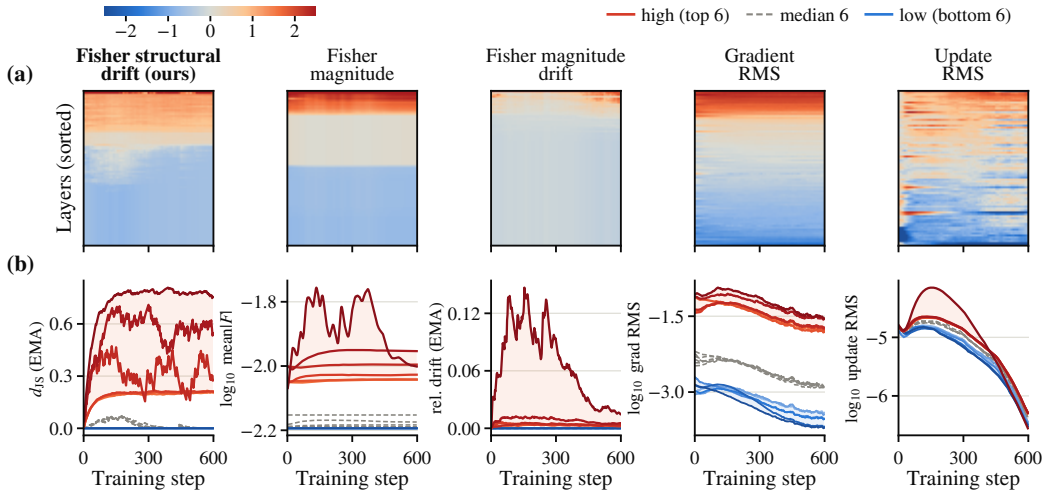


Figure 16: Parameter-selection signals monitored across DINOv2 layers during CIFAR-100 full fine-tuning. (a) Layerwise signal trajectories, with layers sorted within each signal. (b) Trajectories for six high-, six median-, and six low-signal layers. Fisher structural drift distinguishes groups with sustained drift from those that stabilize, while the other signals exhibit different layerwise and temporal patterns.

E.2 Hyperparameter Sensitivity on VTAB-1k

We examine the sensitivity of FisherAdapTune in two complementary settings. First, we conduct a one-at-a-time sensitivity analysis on four representative VTAB-1k classification tasks using DINOv2 ViT-S/14. This analysis considers the parameter-group count k, Fisher collection interval m, freezing interval n, EMA smoothing factor β, and freezing-threshold scale λ. Second, we study the effect of λ more extensively on the downstream crack-segmentation task using SAM2-Tiny, SAM2-Large, SegFormer-MiT-B0, and SegFormer-MiT-B3. The classification experiments assess general robustness to the algorithmic settings, whereas the segmentation experiments examine how the freezing threshold controls adaptation, parameter efficiency, and zero-shot generalization across model families.

Table 6 reports the one-at-a-time sensitivity analysis on CIFAR-100, DTD, SVHN, and EuroSAT. Each hyperparameter is varied while the remaining values are fixed to the base configuration (k=4, m=2, n=6, β=0.8, and λ= $-$ 0.65). The results show that m, n, and β have relatively limited effects on the performance. Across the evaluated collection intervals, macro accuracy varies from 77.76% to 78.58%, indicating that Fisher statistics do not need to be collected at every opportunity (i.e., iteration) to maintain performance. Increasing m also reduces the frequency and computational cost of Fisher estimation. Similarly, varying n produces a macro-accuracy range of 77.23%-78.34%. Although a larger n reduces the frequency of freezing decisions, it delays parameter removal and therefore increases the average trainable budget from 5.38M at n=3 to 12.53M at n=65. The small variation across β values further indicates that the method is not highly sensitive to the precise degree of temporal smoothing.

In contrast, k and λ directly determine the performance-efficiency operating point. Increasing k creates finer-grained parameter groups, allowing FisherAdapTune to isolate and freeze smaller subsets of parameters. This reduces the mean trainable budget from 14.64M at k=1 to 1.59M at k=16, but overly fine grouping degrades macro accuracy. The selected value k=4 retains roughly half the average trainable parameters of k=2 while achieving comparable accuracy, providing a favorable balance between granularity and performance. The threshold scale λ has an even stronger effect on the parameter budget. Lower values apply a more conservative freezing criterion and retain more parameter groups, whereas larger values freeze groups more aggressively. Values between -1.0 and -0.65 yield similar macro accuracy, while λ= $-$ 0.65 reduces the mean trainable budget to 6.52M. Increasing the threshold to -0.5 or -0.3 causes premature freezing and a substantial loss in accuracy. Thus, tuning λ allows FisherAdapTune to retain groups with continuing Fisher geometry variation while removing groups that have stabilized.

Table 6: Hyperparameter sensitivity on four VTAB-1k datasets and DINOv2 ViT-S/14. Each block varies one hyperparameter from a base configuration (K=4, m=2, n=6, β=0.8, λ= − 0.65). * marks the value adopted in this work for further experiments; the starred rows are the run of Table 2 (weight decay 10^{-2}, early-stopping patience 10), whereas the remaining rows use weight decay 0 and patience 15. M denotes the trajectory-averaged trainable-parameter budget in millions.

	CIFAR-100		DTD		SVHN		EuroSAT		Macro	Mean
Setting	Top-1	M	Top-1	M	Top-1	M	Top-1	M	Top-1	M
Block count K										
$k = 1$	57.31	14.73	68.19	14.52	89.52	14.85	95.75	14.47	77.69	14.64
$k = 2$	58.01	12.94	70.05	10.68	89.88	13.11	95.76	12.39	78.42	12.28
$k = 4^\star$	57.11	6.25	71.65	5.62	87.67	7.56	95.76	6.65	78.05	6.52
$k = 8$	53.35	2.58	70.00	2.02	86.34	2.67	95.17	1.61	76.22	2.22
$k = 16$	46.57	1.63	68.56	1.39	82.54	1.88	95.24	1.47	73.23	1.59
Freezing interval n										
$n = 3$	57.83	5.88	71.91	4.77	87.63	4.85	96.00	6.04	78.34	5.38
$n = 6^\star$	57.11	6.25	71.65	5.62	87.67	7.56	95.76	6.65	78.05	6.52
$n = 13$	57.22	8.16	71.86	7.10	86.29	9.48	95.80	6.68	77.79	7.85
$n = 26$	55.36	9.32	70.05	9.80	88.02	10.40	95.51	9.90	77.23	9.85
$n = 65$	55.78	12.11	69.10	12.17	89.47	12.50	95.29	13.33	77.41	12.53
Collection interval m										
$m = 1$	57.45	7.20	72.02	5.07	87.75	6.81	95.39	7.23	78.15	6.58
$m = 2^\star$	57.11	6.25	71.65	5.62	87.67	7.56	95.76	6.65	78.05	6.52
$m = 4$	57.73	6.82	71.17	5.42	86.35	7.83	95.80	5.78	77.76	6.46
$m = 8$	58.29	6.57	72.07	6.51	88.06	7.24	95.90	6.08	78.58	6.60
JS smoothing β										
$\beta = 0$	57.47	6.92	70.90	5.87	88.41	6.41	95.79	5.08	78.14	6.07
$\beta = 0.5$	57.03	6.97	72.02	5.82	88.71	6.80	95.78	5.79	78.38	6.35
$\beta = 0.8^\star$	57.11	6.25	71.65	5.62	87.67	7.56	95.76	6.65	78.05	6.52
$\beta = 0.95$	57.08	6.89	71.17	5.67	88.79	7.30	95.80	6.32	78.21	6.55
Freezing threshold λ										
$\lambda = -1.0$	58.19	12.69	70.16	12.53	89.49	13.04	94.55	14.03	78.10	13.07
$\lambda = -0.8$	58.72	9.79	70.64	9.50	87.90	10.60	95.98	10.06	78.31	9.99
$\lambda = -0.65^\star$	57.11	6.25	71.65	5.62	87.67	7.56	95.76	6.65	78.05	6.52
$\lambda = -0.5$	46.18	2.85	57.82	0.94	52.11	2.57	90.72	1.56	61.71	1.98
$\lambda = -0.3$	8.56	1.04	41.65	0.76	38.43	0.84	88.37	0.75	44.25	0.85

E.3 Freezing-Threshold Sensitivity on Crack Segmentation

We further evaluate the freezing threshold, λ, on crack segmentation as it is the main hyperparameter controlling which parameter groups remain active. These experiments use OmniCrack30k for downstream adaptation and evaluate zero-shot transfer on Concrete3k, Facade390, and Road420. The Fisher collection and freezing intervals are fixed at m=300 and n=3000 training iterations, respectively. Tables 7 and 8 report the results for the SAM2 and SegFormer model families. We report both final and effective number of trainable parameters. The final count represents the parameter groups that remain active at the end of training, whereas the effective number of trainable parameters captures the average trainable-parameter count over the complete optimization. The latter accounts for parameters that contribute updates before being frozen and therefore provides a more representative measure of training-time parameter efficiency.

Table 7 shows that more conservative thresholds generally improve in-distribution performance at the cost of larger trainable budgets. For SAM2-Tiny, moving from $\lambda = 0$ to -1 increases OmniCrack F1 from 61.96% to 64.51% while raises the average budget from 3.82M to 18.07M. The selected $\lambda = -0.85$ maintains 63.12% F1 with only 0.27M parameters active at the end of training. For SAM2-Large, $\lambda = -0.87$ achieves 65.78% F1, 49.01% IoU, and strong zero-shot performance with a final budget of 3.0M. Comparable or slightly better results at $\lambda = -0.9$ and -0.95 require substantially larger final budgets of 56.8M and 74.5M, respectively. Table 8 shows similar trends for SegFormer. For MiT-B0, the selected $\lambda = -0.95$ achieves 60.85% F1 with an average budget of 0.124M and only 128 parameters remaining active, whereas further in-distribution gains require

much larger budgets. For MiT-B3, $\lambda = -0.95$ provides the best overall balance, achieving 64.57% F1 and 47.68% IoU on OmniCrack and the highest mean zero-shot F1 and IoU of 64.49% and 48.00%. Although $\lambda = -1.0$ slightly improves in-distribution performance, it increases the average budget and reduces zero-shot performance, indicating that retaining more parameters does not necessarily improve generalization.

Table 7: Ablation on the freezing threshold λ for **FisherAdapTune** applied to crack segmentation using SAM2-Tiny and SAM2-Large. Larger $|\lambda|$ retains more trainable parameter groups. Bold entries show the best per column within each model block; underline shows second best. Shaded rows indicate the selected λ. Effect. Params denotes number of Effective Tunable Parameters.

Model	λ	Final Tunable Params	Effect. Params	OmniCrack		Zero-shot F1 (%)				Zero-shot IoU (%)			
				F1	IoU	Conc.	Fac.	Road	Avg.±Std	Conc.	Fac.	Road	Avg.±Std
SAM2-Tiny	0.0	4.7K	3.82M	61.96	44.89	72.77	**69.65**	**66.49**	$\mathbf{69.64}_{\pm 3.14}$	57.20	**53.43**	**49.80**	$\mathbf{53.48}_{\pm 3.70}$
	-0.5	4.7K	3.08M	61.42	44.32	71.87	69.22	65.85	$68.98_{\pm 3.02}$	56.09	52.93	49.09	$52.70_{\pm 3.51}$
	-0.8	0.20M	5.40M	62.40	45.35	72.51	68.58	66.29	$69.13_{\pm 3.15}$	56.88	52.19	49.58	$52.88_{\pm 3.70}$
	-0.85	0.27M	6.95M	63.12	46.11	72.43	69.10	64.42	$68.65_{\pm 4.02}$	56.77	52.79	47.52	$52.36_{\pm 4.64}$
	-0.87	9.9M	12.13M	64.20	47.27	72.98	68.84	64.98	$68.93_{\pm 4.00}$	57.45	52.48	48.14	$52.69_{\pm 4.67}$
	-0.9	11.8M	13.69M	64.28	47.36	**72.99**	68.74	64.99	$68.91_{\pm 4.00}$	**57.47**	52.37	48.14	$52.66_{\pm 4.67}$
	-0.95	13.1M	15.02M	64.41	47.50	72.92	68.99	64.97	$68.96_{\pm 3.98}$	57.39	52.67	48.11	$52.72_{\pm 4.64}$
	-1.0	16.6M	18.07M	**64.51**	**47.61**	72.94	69.05	64.97	$68.98_{\pm 3.99}$	57.40	52.73	48.11	$52.75_{\pm 4.65}$
SAM2-Large	0.0	7.1K	17.41M	57.43	40.29	65.17	73.55	66.44	$68.39_{\pm 3.69}$	48.33	58.17	49.74	$52.08_{\pm 4.34}$
	-0.5	7.1K	25.62M	59.60	42.45	68.94	73.93	67.63	$70.17_{\pm 2.72}$	52.60	58.64	51.09	$54.11_{\pm 3.26}$
	-0.7	1.3M	34.36M	62.77	45.74	72.63	74.10	68.58	$71.77_{\pm 2.33}$	57.02	58.85	52.18	$56.02_{\pm 2.81}$
	-0.85	2.0M	59.40M	64.57	47.67	71.99	74.69	68.04	$71.57_{\pm 2.73}$	56.25	59.61	51.56	$55.80_{\pm 3.30}$
	-0.86	2.1M	55.36M	64.33	47.41	71.81	**75.79**	68.54	$72.05_{\pm 2.96}$	56.02	**61.01**	52.14	$56.39_{\pm 3.63}$
	-0.87	3.0M	71.49M	65.78	49.01	72.81	75.30	70.29	$72.80_{\pm 2.04}$	57.24	60.38	54.19	$57.27_{\pm 2.53}$
	-0.88	2.5M	63.52M	65.38	48.57	72.17	74.90	69.33	$72.13_{\pm 2.28}$	56.46	59.87	53.05	$56.46_{\pm 2.78}$
	-0.9	56.8M	82.54M	64.72	47.84	**73.32**	74.72	**71.27**	$\mathbf{73.10}_{\pm 1.42}$	**57.88**	59.64	**55.36**	$\mathbf{57.63}_{\pm 1.76}$
	-0.95	74.5M	97.12M	**65.81**	**49.04**	73.13	73.89	69.24	$72.08_{\pm 2.03}$	57.64	58.59	52.95	$56.39_{\pm 2.46}$
	-1.0	81.9M	101.14M	65.56	48.77	72.10	75.37	70.60	$72.69_{\pm 1.99}$	56.38	60.48	54.56	$57.14_{\pm 2.48}$

Table 8: Ablation on the freezing threshold λ for **FisherAdapTune** applied to crack segmentation using SegFormer-MiT-B0 and SegFormer-MiT-B3. Larger $|\lambda|$ retains more trainable parameter groups. Bold shows best per column within each model block; underline shows second best. Shaded rows indicate the selected λ. Effect. Params denotes number of Effective Tunable Parameters.

Model	λ	Final Tunable Params	Effect. Params	OmniCrack		Zero-shot F1 (%)				Zero-shot IoU (%)			
				F1	IoU	Conc.	Fac.	Road	Avg.±Std	Conc.	Fac.	Road	Avg.±Std
SegFormer-B0	−0.50	0.032K	0.047M	58.44	41.28	70.18	58.70	52.30	$60.39_{\pm 7.39}$	54.06	41.54	35.41	$43.66_{\pm 7.76}$
	−0.65	0.032K	0.074M	59.53	42.37	71.40	59.17	52.03	$60.86_{\pm 7.99}$	55.52	42.02	35.17	$44.23_{\pm 8.45}$
	−0.75	0.032K	0.085M	59.65	42.50	71.36	**59.92**	**56.06**	$\mathbf{62.44_{\pm 6.49}}$	55.47	**42.77**	**38.95**	$\mathbf{45.73_{\pm 7.06}}$
	−0.80	0.064K	0.089M	60.16	43.02	70.71	58.99	54.99	$61.56_{\pm 6.67}$	54.70	41.83	37.92	$44.81_{\pm 7.16}$
	−0.85	0.064K	0.093M	60.23	43.09	69.66	58.54	54.52	$60.90_{\pm 6.40}$	53.44	41.38	37.48	$44.09_{\pm 6.79}$
	−0.90	0.096K	0.100M	60.09	42.95	68.77	58.50	52.33	$59.86_{\pm 6.78}$	52.41	41.35	35.43	$43.06_{\pm 7.03}$
	−0.95	0.128K	0.124M	60.85	43.73	70.97	58.77	53.63	$61.12_{\pm 7.27}$	55.01	41.62	36.64	$44.42_{\pm 7.75}$
	−1.00	0.160K	0.144M	60.79	43.67	70.62	59.02	53.29	$60.97_{\pm 7.20}$	54.59	41.87	36.32	$44.26_{\pm 7.64}$
	−1.25	0.048M	0.360M	62.28	45.22	70.52	58.89	52.59	$60.67_{\pm 7.43}$	54.46	41.73	35.67	$43.96_{\pm 7.83}$
	−1.5	1.31M	1.85M	62.96	45.95	**73.03**	56.58	54.98	$61.53_{\pm 8.16}$	**57.52**	39.45	37.91	$44.96_{\pm 8.90}$
	−2	3.37M	3.54M	63.23	46.23	73.02	58.01	53.64	$61.56_{\pm 8.30}$	57.50	40.85	36.65	$45.00_{\pm 9.00}$
	−3	3.71M	3.71M	**63.25**	**46.25**	72.76	56.96	53.56	$61.09_{\pm 8.36}$	57.18	39.82	36.57	$44.52_{\pm 9.05}$
SegFormer-B3	−0.50	3K	5.02M	61.84	44.76	73.21	60.80	52.66	$62.22_{\pm 8.44}$	57.74	43.68	35.74	$45.72_{\pm 9.09}$
	−0.65	4K	6.19M	62.83	45.81	72.90	60.73	51.72	$61.78_{\pm 8.67}$	57.36	43.61	34.88	$45.28_{\pm 9.25}$
	−0.75	6K	7.10M	62.94	45.92	73.12	60.19	53.45	$62.25_{\pm 8.16}$	57.63	43.05	36.47	$45.71_{\pm 8.84}$
	−0.80	7K	7.80M	63.76	46.80	73.64	59.21	53.38	$62.07_{\pm 8.51}$	58.29	42.06	36.40	$45.58_{\pm 9.27}$
	−0.85	7K	8.48M	63.54	46.57	73.04	59.89	55.38	$62.77_{\pm 7.49}$	57.53	42.74	38.30	$46.19_{\pm 8.22}$
	−0.90	7K	9.38M	63.90	46.95	73.35	59.39	57.76	$63.50_{\pm 6.99}$	57.91	42.24	40.60	$46.91_{\pm 7.80}$
	−0.95	11K	10.73M	64.57	47.68	74.19	**61.04**	**58.26**	$\mathbf{64.49_{\pm 6.94}}$	58.97	**43.93**	**41.11**	$\mathbf{48.00_{\pm 7.83}}$
	−1.00	14K	12.81M	**64.97**	**48.11**	**75.85**	57.83	56.52	$63.40_{\pm 8.81}$	**61.10**	40.68	39.40	$47.06_{\pm 9.94}$

Figure 11 summarizes the threshold trends across both model families. More conservative thresholds retain more parameters and generally improve in-distribution performance, whereas zero-shot performance often peaks at an intermediate budget. Thus, the optimal threshold balances downstream adaptation, zero-shot generalization, and parameter efficiency.

E.4 Active-Parameter Trajectories Across Thresholds

Figures 17-20 show how the active trainable-parameter budget evolves throughout optimization. These trajectories complement the aggregate counts in Tables 7 and 8 by showing when parameter groups are frozen. Thresholds with smaller absolute values satisfy the freezing criterion earlier, producing a rapid decline in the active budget. More negative thresholds apply a more conservative criterion, keeping a larger portion of the model trainable for longer. The trajectories also reveal architecture-dependent transitions in the final active set. In SAM2, small changes near the selected thresholds can cause large differences in the number of retained parameters, particularly for the larger backbone. SegFormer exhibits the same general relationship between λ and budget, although its finer parameter grouping permits very small final active sets. Overall, the plots confirm that λ controls not only the final number of trainable parameters but also the duration for which different parameter groups contribute to adaptation.

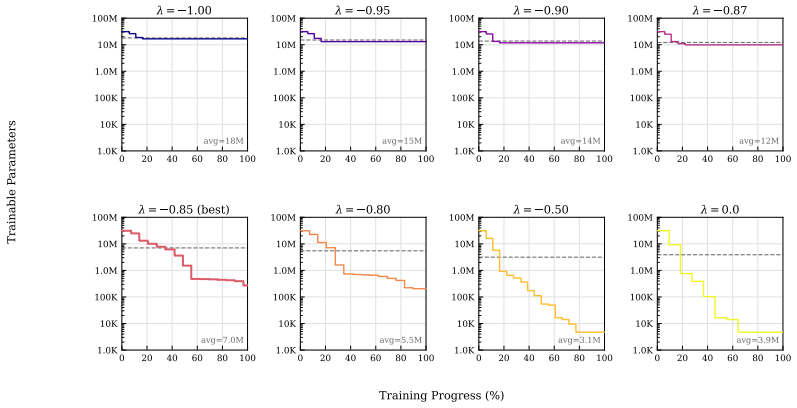


Figure 17: Trainable-parameter evolution for SAM2-Tiny across freezing thresholds λ. "avg." denotes the effective number of trainable parameters.

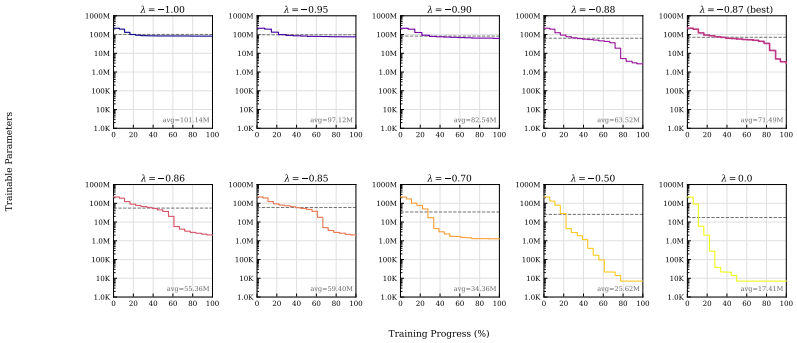


Figure 18: Trainable-parameter evolution for SAM2-Large across freezing thresholds λ. "avg." denotes the effective number of trainable parameters.

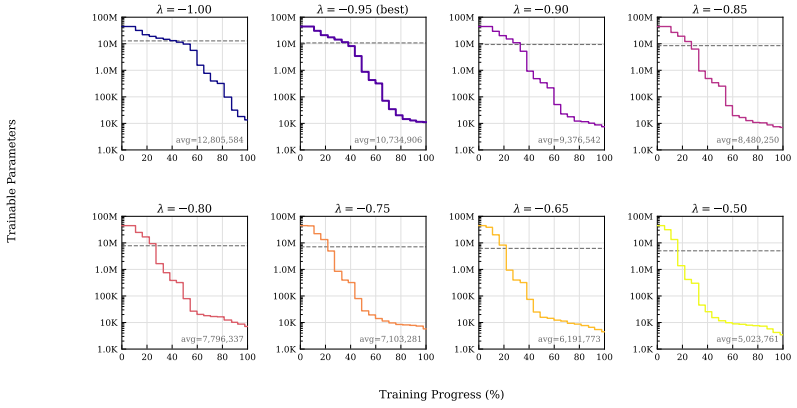


Figure 19: Trainable-parameter evolution for SegFormer-MiT-B3 across freezing thresholds λ. "avg." denotes the effective number of trainable parameters.

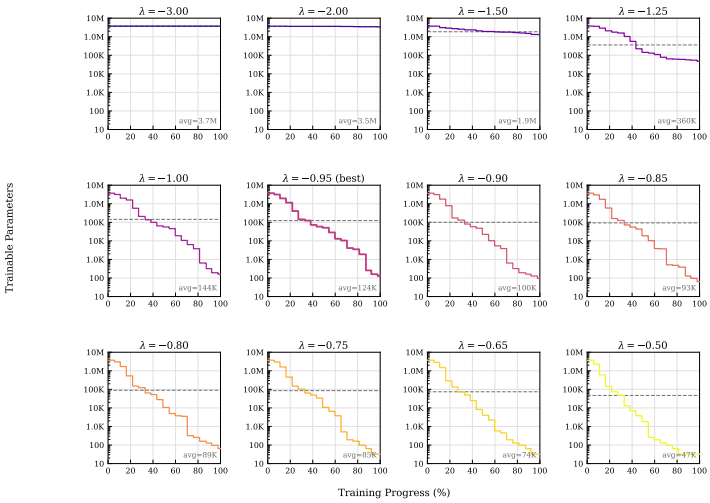


Figure 20: Trainable-parameter evolution for SegFormer-MiT-B0 across freezing thresholds λ. "avg." denotes the effective number of trainable parameters.

F Fisher Drift and Parameter-Selection Dynamics

F.1 Heterogeneity Across Layers and Parameter Groups

To examine whether Fisher geometry evolves uniformly during adaptation, we tracked the exponential moving average of the Jensen-Shannon distance between consecutive Fisher-value distributions throughout full fine-tuning. Figures 21 and 22 reveal evident heterogeneity at multiple levels. In Figure 21, Fisher drift varies not only across layers but also among column-wise parameter groups within the same layer, with some groups exhibiting different convergence levels. SegFormer-B3 similarly shows a broad layer-wise spread in drift, while normalization, attention, and MLP/fully connected parameters form distinct drift distributions. These observations indicate that parameter groups stabilize at different rates and that a single architecture-wide freezing schedule cannot capture their heterogeneous adaptation dynamics, motivating FisherAdapTune's progressive, group-specific selection strategy.

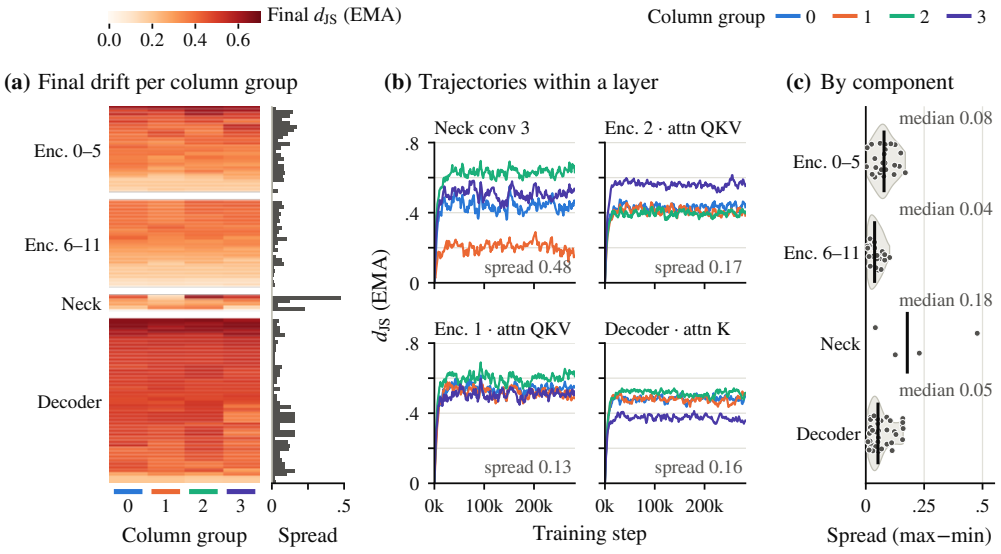


Figure 21: Parameter group-wise Jensen-Shannon (JS) distance between consecutive Fisher distributions in SAM2-Tiny during full fine-tuning on crack segmentation; $EMA[d^t_{\mathrm{JS}_i}(p(\tilde{\mathbf{F}}_i^{t-1}), p(\tilde{\mathbf{F}}_i^t))]$.

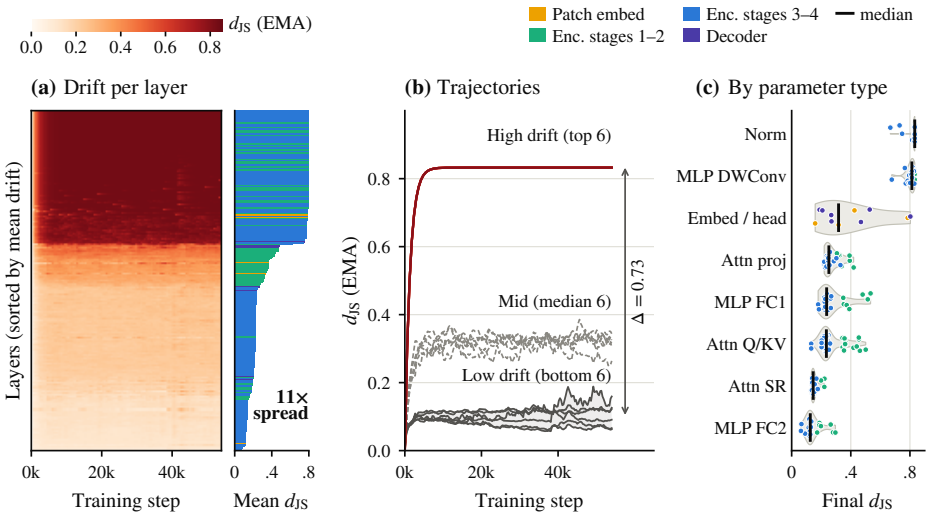


Figure 22: Layer-wise Jensen-Shannon (JS) distance between consecutive Fisher distributions in SegFormer during full fine-tuning on crack segmentation; $EMA[d^t_{\mathrm{JS}_i}(p(\tilde{\mathbf{F}}_i^{t-1}), p(\tilde{\mathbf{F}}_i^t))]$.

F.2 Architectural Patterns in the Selected Parameters

The selected groups reveal different adaptation trajectories across architectures. For SAM2-Tiny, large encoder attention and MLP groups freeze relatively early, whereas patch embedding and several mask-decoder groups remain trainable longer (Figure 6). Several normalization groups also have long trainable lifetimes despite their small parameter counts. Thus, the reduction in trainable budget is driven largely by freezing high-capacity encoder groups. It does not imply that all components in a stage stop adapting together. Across all VTAB-1k tasks, DINOv2 exhibits a recurring pattern: patch embedding and the classification head have the longest trainable lifetimes, while transformer-block groups freeze at different rates (Figure 24). This is consistent with continued adjustment at the input and task-specific output, although trainable lifetime alone does not establish the function of those updates. SegFormer-MiT-B3 shows a different allocation where intermediate encoder groups, particularly Stage 3 attention and MLP, remain active longer, alongside selected normalization groups (Figure 23). Overall, the patterns show that FisherAdapTune does not recover a fixed freezing order or a specific block-type training strategy. Instead, the location and duration of adaptation vary with the architecture and downstream task.

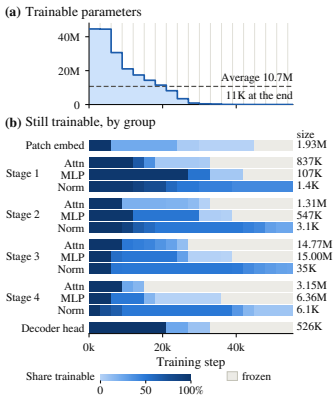


Figure 23: Trainable parameter budget and group lifetimes for SegFormer-MiT-B3.

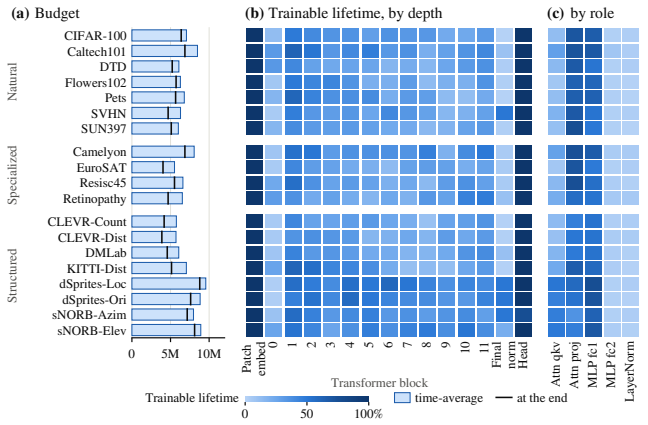


Figure 24: Trainable parameter budgets and group lifetimes for DINOv2 across 19 VTAB-1k tasks.