ECR-LoRA: Evolving Core LoRA and Complementary Residual LoRAs for Class-Incremental Learning

Anonymous authors
Paper under double-blind review

Abstract

Rehearsal-free parameter-efficient continual learning (PECL) enables frozen pretrained Vision Transformers (ViTs) to adapt to sequentially arriving tasks using lightweight adaptation modules, such as LoRA, without historical exemplars. Existing LoRA-based PECL methods face a fundamental trade-off between knowledge sharing and task specialization: sharing or merging adapters promotes reuse but blurs task-specific representations, whereas isolating updates mitigates forgetting at the cost of reusable knowledge transfer. We propose ECR-LoRA, a split-depth LoRA framework that organizes continual adaptation into an evolving Core LoRA and complementary task-specific Residual LoRAs. Early blocks maintain the fixed-rank Core, which evolves across tasks to accumulate transferable knowledge, while later blocks preserve Residual LoRAs for evidence not sufficiently captured by the Core. Core evolution is controlled by subspace-compatible learning and Fisher-guided consolidation, while core-error reweighting and residual-mix consistency encourage complementary residual learning and robustness to cross-residual variation. During inference, uncertainty from the consolidated core provides an entropy-guided prior for calibrating residual experts without task identities, additional training, or exemplar replay. Experiments on four class-incremental learning benchmarks show that ECR-LoRA achieves the best final accuracy on all four and the best average accuracy on three, with a favorable stability–plasticity trade-off. Code is available in the supplementary material.

1 Introduction

Continual learning (CL) aims to acquire knowledge from a sequence of tasks without catastrophically forgetting previously learned concepts (Parisi et al., 2019; De Lange et al., 2022). With the emergence of large pretrained visual Transformers (Dosovitskiy et al., 2021), recent CL methods increasingly adapt a strong frozen backbone rather than repeatedly updating the entire model. We study rehearsal-free class-incremental learning in this setting, where new classes arrive sequentially and inference covers all previously seen classes without access to task identities or stored exemplars.

Parameter-efficient fine-tuning (PEFT) provides a practical means of adapting such backbones by optimizing only a small set of additional parameters (Houlsby et al., 2019). Among PEFT techniques, Low-Rank Adaptation (LoRA) is particularly attractive because it parameterizes task updates with compact trainable low-rank matrices (Hu et al., 2022). In this continual setting, however, the central challenge is not only how to learn a compact update for the current task, but also how to organize the accumulated LoRA updates over time. Each update may encode both reusable patterns and task-local evidence, yet standard LoRA does not explicitly distinguish between them.

Existing continual LoRA methods mainly organize task updates through consolidation or interference avoidance. Merge-centric approaches promote knowledge reuse by aggregating task adapters, but may blur task-specific distinctions (Li et al., 2025; Qiu et al., 2025). Interference-aware methods instead constrain new updates to orthogonal, decoupled, or low-conflict subspaces, improving stability at the risk of restricting reusable directions (Liang & Li, 2024; Wu et al., 2025; Zhu et al., 2025; Wang et al., 2025; Cheng et al., 2025; Qiu et al., 2026; Chen et al., 2026). Shared-state methods further retain common adaptation components under structural or regularization constraints (He

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.

et al., 2025; Zheng et al., 2026). Despite these advances, reusable and task-specific knowledge is still largely organized at the level of entire adapters or predefined branches. This leaves a more fundamental question unresolved: *what should be shared, and what should remain task-specific?*

Motivated by this observation, we propose ECR-LoRA, a split-depth LoRA framework that organizes continual adaptation into an evolving shared Core LoRA and complementary task-specific Residual LoRAs. Early Transformer blocks maintain the fixed-rank Core, which evolves across tasks to accumulate transferable adaptation knowledge, while later blocks preserve a bank of Residual LoRAs for evidence not sufficiently represented by the Core. After each task, reusable information is selectively consolidated into the Core, whereas the current Residual LoRA remains task-specific rather than being merged into the shared state. This asymmetric organization allows shared knowledge to evolve without forcing task-discriminative information into a single persistent adapter.

To realize this organization, ECR-LoRA comprises three corresponding mechanisms. For Evolving Core LoRA, subspace-compatible candidate learning encourages new updates to reuse directions represented by the existing Core, while Fisher-guided consolidation determines how strongly task-supported information is absorbed under a fixed-rank budget. For Complementary Residual LoRAs, core-error reweighting emphasizes examples insufficiently explained by the shared part, while residual-mix consistency improves robustness to variations induced by historical Residual LoRAs. Such designs encourage complementary role of residuals rather than redundant corrections. During inference, uncertainty from the consolidated Core additionally provides an entropy-guided common reference for calibrating residual experts without task identities or further training.

The main contributions of this work are summarized as follows:

- We propose ECR-LoRA, which builds on a split-depth LoRA organization and turns the shared branch into an evolving fixed-rank Core LoRA, while specializing later-layer adapters as complementary task-specific Residual LoRAs.
- We develop a controlled Core–Residual learning strategy that regulates reusable knowledge accumulation in the Core and explicitly directs Residual LoRAs toward core-complementary, cross-task-robust corrections.
- We further reuse the consolidated Core as an entropy-guided, task-agnostic prior for residual calibration. Experiments on four class-incremental benchmarks demonstrate consistent improvements and a favorable stability–plasticity trade-off.

2 Related Work

PEFT-based continual learning. Large pre-trained models have made frozen-backbone adaptation a practical paradigm for continual learning. PEFT methods update lightweight components while preserving the pre-trained parameters (Houlsby et al., 2019; Hu et al., 2022). Prompt-based approaches such as L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), and CODA-Prompt (Smith et al., 2023) learn or select prompts for rehearsal-free class-incremental learning. Adapter-based methods such as EASE (Zhou et al., 2024) construct expandable task subspaces for joint classification. These works establish lightweight adaptation as an effective continual-learning interface.

Interference-aware continual LoRA. A major line of continual LoRA research controls update directions to protect historical knowledge. InfLoRA constructs interference-free low-rank subspaces (Liang & Li, 2024); SD-LoRA decouples update direction and magnitude (Wu et al., 2025); BiLoRA expands almost-orthogonal parameter spaces (Zhu et al., 2025); and PLAN proactively allocates low-rank bases to reduce conflicts with previous parameters (Wang et al., 2025). CoSO further optimizes within a sequence of gradient-derived subspaces that remain orthogonal to historical directions (Cheng et al., 2025). More recently, SplitLoRA determines a minor historical gradient subspace and confines LoRA updates to that space to balance stability and plasticity (Qiu et al., 2026), whereas Janus-LoRA rectifies the coupled gradients of LoRA factors to enforce historical-subspace orthogonality and adds feature-level margin separation (Chen et al., 2026). These methods effectively reduce interference, but their principal mechanism is still to select, separate, or protect update directions. However, directional protection does not explicitly determine how reusable and task-specific knowledge should be organized.

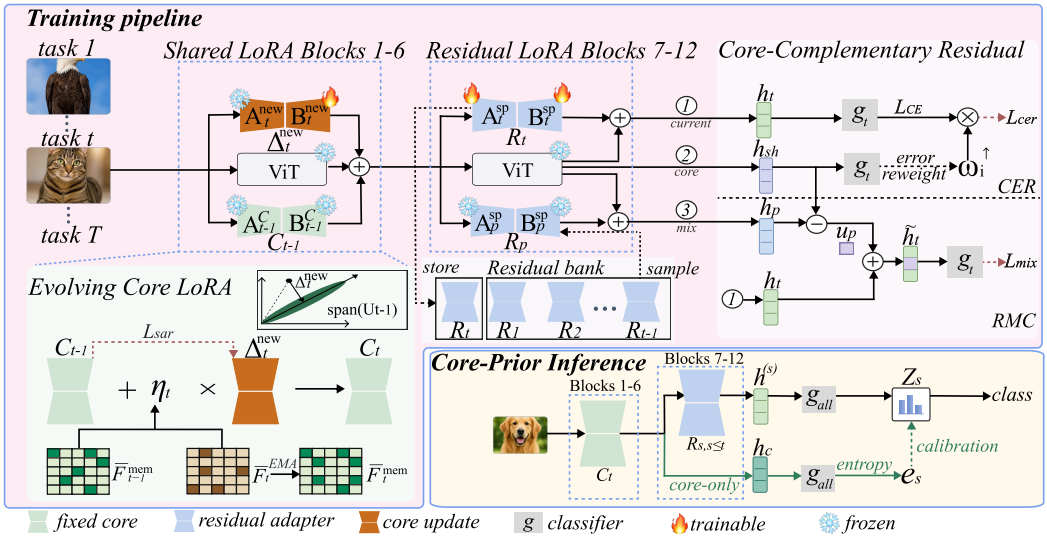


Figure 1: Overview of ECR-LoRA. During task-t training, early transformer blocks maintain an evolving Core LoRA that consolidates reusable knowledge, while later blocks learn task-specific residual LoRAs specialized to core-complementary evidence and historical-residual variation. After training, the consolidated core and residual bank are reused for task-agnostic inference, where core-only entropy calibrates residual-expert predictions. For clarity, the frozen ViT is omitted from the inference panel; the core-only path passes through Blocks 7–12 with residual adapters disabled.

Shared knowledge and continual consolidation. Closely related to our architecture, CL-LoRA combines task-shared and task-specific adapters across transformer depth (He et al., 2025), while LoDA decomposes LoRA update spaces into general and isolated components (He et al., 2026). Our framework adopts the shared/specific perspective but changes how both components evolve: the shared branch becomes a selectively updated fixed-rank core, and task-specific adapters are explicitly specialized to complement that core. Core evolution is also related to importance regularization and model consolidation. EWC (Kirkpatrick et al., 2017) and EWC-LoRA (Zheng et al., 2026) use Fisher information for parameter protection, while CoMA/CoFiMA (Marouf et al., 2024), BECAME (Li et al., 2025), and LoRA-P&M (Qiu et al., 2025) consolidate successive task solutions through fixed, Fisher-weighted, or adaptive merging. Different from globally regularizing all LoRA parameters or collapsing task updates into one model, we apply Fisher-guided absorption only to the evolving core, and further reuse the core as a task-agnostic inference prior.

3 Method

3.1 Problem Definition

We consider rehearsal-free class-incremental learning over a sequence of T tasks $\{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$. Each task $\mathcal{D}_t = \{(x_i, y_i)\}_{i=1}^{n_t}$ introduces a class set $\mathcal{Y}_t$, where $\mathcal{Y}_t \cap \mathcal{Y}_{t'} = \emptyset$ for $t \neq t'$. After learning task t, the model is required to classify samples from all observed classes $\mathcal{Y}_{1:t} = \bigcup_{\tau=1}^{t} \mathcal{Y}_\tau$ without access to task identities or stored exemplars. The pretrained transformer backbone remains frozen throughout continual adaptation. Following the prototype-based classification protocol of CL-LoRA (He et al., 2025), each task uses a cosine classifier $g_t : \mathbb{R}^d \to \mathbb{R}^{|\mathcal{Y}_t|}$, where d is the feature dimension.

3.2 Overall Framework

Figure 1 presents the overall architecture of ECR-LoRA. Following the split-depth shared/task-specific organization of CL-LoRA, we divide Transformer depth into shared and task-specific adaptation stages, with a default 6/6 allocation. Rather than treating the shared component as a conventional shared adapter, ECR-LoRA maintains the LoRA modules in blocks 1–6 as a fixed-rank

shared state that continuously evolves across tasks to accumulate reusable knowledge; we refer to this persistent component as the *core*. The LoRA modules in blocks 7–12 are instead maintained as task-specific residual adapters that preserve complementary evidence not sufficiently represented by the evolving core.

At task t, the consolidated core C_{t-1} and all historical residuals are frozen, while a new core candidate Δ_t^{new}, the current residual R_t, and classifier head g_t are optimized on $\mathcal{D}_t$. For shared knowledge evolution, subspace-compatible learning encourages the candidate to reuse directions represented by the existing core, followed by Fisher-guided consolidation that selectively absorbs the learned update into the next fixed-rank core C_t. In parallel, core-error reweighting emphasizes evidence insufficiently explained by the shared path, while residual-mix consistency promotes compatibility with variations induced by historical residuals. After training, R_t is frozen and appended to the residual bank. At inference, core-only entropy provides a common task-agnostic prior for calibrating residual-expert predictions without task identities or an additional learned router. Sections 3.3–3.5 detail these components.

3.3 Evolving Core LoRA

The evolving core LoRA aims to accumulate transferable adaptation knowledge while maintaining sufficient flexibility for future tasks. Unlike approaches that primarily restrict interactions between new and previous updates to mitigate interference, we allow the shared representation to continuously absorb reusable information while controlling how new knowledge enters the existing core. Core evolution therefore involves two complementary decisions: whether a candidate update is compatible with the current shared space and how strongly the compatible information should be retained. We address these two aspects through subspace-compatible candidate learning and Fisher-guided core consolidation.

Subspace-compatible candidate learning. This stage encourages the new shared candidate to reuse directions already represented by the evolving core. Let $\mathcal{B}_c$ index the Q/V LoRA branches in the Core. We omit the branch index for a generic branch and restore it in branch-wise aggregation. The previous core is represented as $C_{t-1} = B_{t-1}^c A_{t-1}^c$. For $t > 1$, the candidate down-projection A_t^{new} is initialized (seeded) from the previous core and then kept frozen during task-t optimization, while only B_t^{new} is optimized. The resulting candidate is $\Delta_t^{\text{new}} = B_t^{\text{new}} A_t^{\text{new}}$. This core-derived initialization provides a compatible input-side coordinate system without treating the candidate as a copy of the previous parameters. Let $C_{t-1} = U_{t-1}\Sigma_{t-1}V_{t-1}^\top$ be the compact SVD of the previous core, where the columns of U_{t-1} span its output subspace. We isolate directions outside this subspace using the orthogonal-complement projector

$$P_{t-1} = I - U_{t-1}U_{t-1}^\top, \tag{1}$$

and penalize only the out-of-subspace component of the candidate:

$$\mathcal{L}_{\text{sar}}^{(t)} = \frac{1}{|\mathcal{B}_c|} \sum_{j \in \mathcal{B}_c} \left\| P_{t-1,j} \Delta_{t,j}^{\text{new}} \right\|_F^2 . \tag{2}$$

This soft penalty discourages unsupported directions without hard-constraining the candidate to the existing subspace. For $t = 1$, we use standard LoRA initialization and omit $\mathcal{L}_{\text{sar}}^{(1)}$. Appendix A.2 shows that each term equals the squared distance from $\Delta_{t,j}^{\text{new}}$ to the set of matrices whose columns lie in $\text{span}(U_{t-1,j})$.

Fisher-guided core consolidation. Subspace compatibility determines *where* the candidate can evolve, but not how strongly it should be retained. After candidate learning, we accumulate squared gradients of the trainable factor B_t^{new} over several current-task mini-batches. Let $\bar{F}_t$ denote the resulting normalized Fisher support of the candidate and $\bar{F}_{t-1}^{\text{mem}}$ the accumulated support of the existing core, maintained by exponential moving average. For each branch, the absorption coefficient is

$$\eta_t = \frac{\bar{F}_t}{\bar{F}_{t-1}^{\text{mem}} + \bar{F}_t + \epsilon}, \tag{3}$$

where $\epsilon > 0$ is a numerical stabilizer. Stronger current-task support increases candidate absorption, whereas stronger historical support makes the core update more conservative. Given the optimized

candidate $\widehat{\Delta}_t^{\text{new}}$, the core is consolidated as

$$C_t = \Pi_r\Big(C_{t-1} + \eta_t \widehat{\Delta}_t^{\text{new}}\Big), \tag{4}$$

where $\Pi_r(\cdot)$ denotes rank-r truncated-SVD refactorization. Appendix A.2 derives Eq. (3) as the closed-form minimizer of a precision-weighted quadratic and interprets Π_r as the optimal rank-r approximation. Together, subspace compatibility determines *where* the core evolves, while Fisher-guided consolidation determines *how much* task-supported information persists under the fixed-rank budget.

3.4 Core-Complementary Residual LoRA

Although the fixed-rank core accumulates reusable knowledge, it cannot represent all task-dependent variations. Residual specialization therefore serves two coupled requirements: *specificity*, so that task-specific correction is emphasized on evidence not sufficiently explained by the core, and *cross-residual compatibility*, so that such corrections remain reliable under variations induced by stored residuals. We realize these requirements through core-error reweighting (CER) and residual-mix consistency (RMC), respectively.

Specificity via core-error reweighting. Let $\mathcal{B}_r$ denote the LoRA branches assigned to the residual blocks. For each $j \in \mathcal{B}_r$, both LoRA factors are optimized, $\Delta_{t,j}^{\text{sp}} = B_{t,j}^{\text{sp}} A_{t,j}^{\text{sp}}$. We collect these branch-wise updates as $R_t = \{\Delta_{t,j}^{\text{sp}}\}_{j\in\mathcal{B}_r}$. Since CER and RMC operate on features of the complete residual adapter, we omit the branch index below.

To identify evidence that still requires task-specific correction, we compare the current shared-only path with the complete Core–Residual path. For a mini-batch of size N_b, let $h_{\text{sh}}(x)$ denote the feature produced by the frozen previous core and the trainable shared candidate with all residuals disabled, and let $h_t(x)$ denote the full feature after activating R_t. We use ℓ_{CE} for per-sample cross-entropy on g_t's cosine scores, and $\mathcal{L}_{\text{CE}}^{(t)}$ for its mini-batch average on the full Core–Residual path. The shared-only prediction is $\widehat{y}_i^{\text{sh}} = \arg\max_{c\in\mathcal{Y}_t}[g_t(h_{\text{sh}}(x_i))]_c$. Samples misclassified by this path indicate evidence not yet captured by the shared representation and are assigned greater weight:

$$\begin{aligned} \widetilde{\omega}_i &= \begin{cases} 1, & \widehat{y}_i^{\text{sh}} \neq y_i, \\ \omega_{\min}, & \widehat{y}_i^{\text{sh}} = y_i, \end{cases} \\ \omega_i &= \frac{\widetilde{\omega}_i}{\frac{1}{N_b}\sum_{m=1}^{N_b}\widetilde{\omega}_m + \epsilon}. \end{aligned} \tag{5}$$

Here, $\omega_{\min} \in (0, 1]$ keeps shared-path-correct samples active, while normalization stabilizes the loss scale. The weighted objective is evaluated on the complete path:

$$\mathcal{L}_{\text{cer}}^{(t)} = \frac{1}{N_b}\sum_{i=1}^{N_b} \omega_i \, \ell_{\text{CE}}\left(g_t(h_t(x_i)), y_i\right). \tag{6}$$

Thus, the shared-only path supplies the specificity signal, while gradients are computed through the trainable full Core–Residual path, emphasizing evidence that remains unexplained by the core.

Compatibility via residual-mix consistency. Specificity alone does not ensure that residual experts learned at different stages remain reliable when compared at inference. Although all residuals are paired with the same consolidated core, different experts induce distinct feature shifts. For $t > 1$, we therefore sample a frozen historical residual $R_p \in \mathcal{R}_{1:t-1}$ and let $h_p(x)$ denote the feature obtained by combining it with the current shared-only path. We construct

$$\begin{aligned} u_p(x) &= h_p(x) - h_{\text{sh}}(x), \\ \widetilde{h}_t(x) &= h_t(x) + \alpha u_p(x), \end{aligned} \tag{7}$$

where $\alpha \sim \text{Unif}[\alpha_{\min}, \alpha_{\max}]$ controls the exposure strength. Because $u_p(x)$ is induced by an actual stored residual, it captures structured cross-residual variation encountered during residual-bank traversal rather than arbitrary perturbation. We enforce current-task label consistency under

this exposure through

$$\mathcal{L}_{\text{mix}}^{(t)} = \frac{1}{N_b} \sum_{i=1}^{N_b} \ell_{\text{CE}} \left(g_t(\widetilde{h}_t(x_i)), y_i \right). \tag{8}$$

For $t = 1$, no historical residual is available and $\mathcal{L}_{\text{mix}}^{(1)} = 0$.

Together with classification loss and core-compatibility regularization, the complete objective is

$$\mathcal{L}_t = \mathcal{L}_{\text{CE}}^{(t)} + \lambda_{\text{sar}} \mathcal{L}_{\text{sar}}^{(t)} + \lambda_{\text{cer}} \mathcal{L}_{\text{cer}}^{(t)} + \lambda_{\text{mix}} \mathcal{L}_{\text{mix}}^{(t)}. \tag{9}$$

Here, λ_{sar}, λ_{cer}, and λ_{mix} control the corresponding terms. After optimization, the shared candidate is consolidated into the evolving core, whereas R_t is frozen and appended to the residual bank. CER determines *what* evidence should receive task-specific correction, while RMC governs *how* such corrections remain robust to cross-residual variation.

3.5 Core-Prior Inference

Beyond accumulating reusable knowledge during training, the same consolidated core provides a common reference at inference. Without task identities, predictions from residual experts learned at different stages must be compared across task-specific class subsets. Because these residuals are optimized under different task distributions, their confidence scores are not necessarily calibrated. Since the core is shared by every residual path, its uncertainty provides a natural task-level reference for adjusting this cross-expert competition.

For each stored residual R_s, let $h^{(s)}(x)$ denote the feature obtained by combining it with the consolidated core C_t, and define the task-restricted logits as $z_s(x) = [g_{\text{all}}^{(t)}(h^{(s)}(x))]_{\mathcal{Y}_s}$, where $[\cdot]_{\mathcal{Y}_s}$ selects the outputs associated with task s. In parallel, the core-only pathway produces $z^{\text{c}}(x) = g_{\text{all}}^{(t)}(h_c(x))$, where $h_c(x)$ denotes the core-only feature obtained with all residuals disabled. We measure the core uncertainty over each task-specific class subset by the normalized entropy

$$e_s(x) = \frac{H(\text{softmax}\left([z^{\text{c}}(x)]_{\mathcal{Y}_s}\right))}{\log |\mathcal{Y}_s|}, \tag{10}$$

where H denotes Shannon entropy. The normalization makes uncertainty comparable across class subsets of different sizes, and a smaller $e_s(x)$ indicates stronger core support for task s.

We use this shared signal to calibrate each residual expert. For $c \in \mathcal{Y}_s$, the calibrated score is

$$S_{s,c}(x) = [\text{softmax}\left(z_s(x)\right)]_c - \beta_H e_s(x), \tag{11}$$

where $\beta_H \geq 0$ controls the strength of the core prior. The final task-agnostic prediction is

$$\widehat{y} = \underset{\substack{1 \leq s \leq t \\ c \in \mathcal{Y}_s}}{\arg\max} S_{s,c}(x). \tag{12}$$

Because the entropy offset is shared by all classes within the same residual expert, it changes only cross-expert competition while preserving the within-expert class ranking. Thus, the core calibrates *which residual path should be trusted* without altering *what that residual predicts*. Appendix A.3 establishes this ranking-preservation property formally. The procedure requires neither task identities nor stored exemplars and introduces no additional learned router, allowing the shared core to serve both as persistent cross-task memory and as an inference-time reference.

4 Experiments

4.1 Datasets and Metrics

Following the EWC-LoRA protocol (Zheng et al., 2026), we evaluate on four benchmarks for pretrained-model-based class-incremental learning. CIFAR-100 (Krizhevsky, 2009), ImageNet-R (Hendrycks et al., 2021a), and ImageNet-A (Hendrycks et al., 2021b) are each divided into 10 sessions, whereas DomainNet (Peng et al., 2019) uses 5 sessions. These benchmarks cover standard object recognition, natural distribution shifts, and multi-domain variation, enabling evaluation of both transferable knowledge reuse and task-specific adaptation. We report final accuracy A_{Last} over all seen classes after the final session and average accuracy A_{Avg} across all incremental sessions.

Table 1: Comparison on four class-incremental learning benchmarks. We report A_{Last} and A_{Avg} in percentage as mean±standard deviation. Our results are averaged over three seeds. Bold and underline indicate the best and second-best results.

Method	Venue	10S-CIFAR-100		10S-ImageNet-R		10S-ImageNet-A		5S-DomainNet	
		$A_{\text{Last}}\uparrow$	$A_{\text{Avg}}\uparrow$	$A_{\text{Last}}\uparrow$	$A_{\text{Avg}}\uparrow$	$A_{\text{Last}}\uparrow$	$A_{\text{Avg}}\uparrow$	$A_{\text{Last}}\uparrow$	$A_{\text{Avg}}\uparrow$
L2P	CVPR'22	83.18±1.20	87.69±1.05	71.26±0.44	76.13±0.46	42.94±1.27	51.40±1.95	70.26±0.25	75.83±0.98
DualPrompt	ECCV'22	81.48±0.86	86.41±0.66	68.22±0.20	73.81±0.39	45.49±0.96	54.68±1.24	68.26±0.90	73.84±0.45
CODA-Prompt	CVPR'23	86.31±0.12	90.67±0.22	74.05±0.41	78.14±0.39	45.36±0.78	57.03±0.94	70.58±0.53	76.68±0.44
EASE	CVPR'24	87.76±0.44	92.02±0.76	75.91±0.17	81.38±0.29	54.93±1.14	63.92±0.76	–	–
InfLoRA	CVPR'24	86.34±0.76	91.33±0.48	74.41±0.63	80.31±0.60	50.75±1.33	64.36±1.01	71.01±0.05	77.75±0.03
SD-LoRA	ICLR'25	86.77±0.30	90.96±0.30	72.93±2.76	79.80±0.15	55.23±0.94	66.10±0.54	71.27±0.14	77.70±0.07
CL-LoRA	CVPR'25	87.65±0.53	92.25±0.41	78.72±0.44	85.20±0.45	57.62±0.89	70.76±0.63	71.06±0.31	77.76±0.29
BiLoRA	CVPR'25	85.99±0.49	90.62±0.42	74.28±0.92	77.38±0.88	51.05±0.74	62.82±0.65	69.75±0.23	73.86±0.21
EWC-LoRA	ICLR'26	87.91±0.57	92.27±0.39	72.86±0.79	78.95±0.86	59.89±0.26	68.33±0.67	73.46±0.16	79.58±0.10
SplitLoRA	ICLR'26	87.47±0.70	91.56±1.37	81.02±0.11	**86.05±0.73**	62.78±0.51	71.62±1.99	–	–
Janus-LoRA	ICML'26	88.68±0.25	92.58±0.20	75.78±0.16	81.64±0.27	–	–	73.82±0.09	79.67±0.05
ECR-LoRA (All-scan)	–	87.35±0.18	92.20±0.28	80.65±0.19	85.37±1.29	62.29±1.17	70.72±0.94	74.57±0.03	80.11±0.07
ECR-LoRA	–	**89.22±0.12**	**93.17±0.18**	**81.24±0.39**	85.80±1.11	**64.18±1.01**	**71.95±0.94**	**74.68±0.01**	**80.20±0.08**

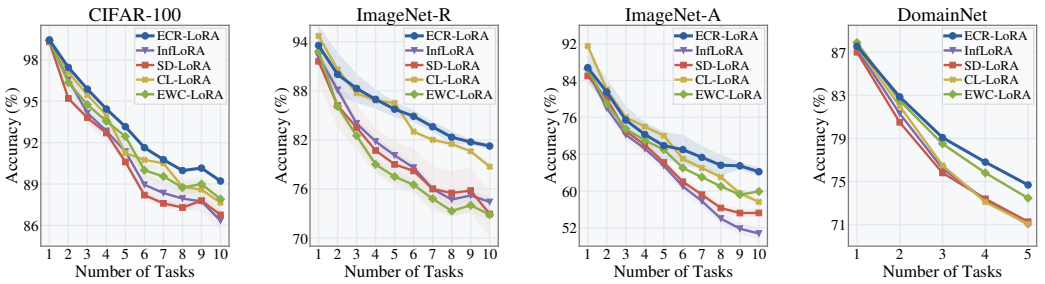


Figure 2: Task-wise accuracy over all observed classes after each incremental session.

4.2 Implementation Details

For ECR-LoRA and the baselines reproduced in our codebase, we use a frozen ViT-B/16 (Dosovitskiy et al., 2021) backbone pretrained on ImageNet-21K (Deng et al., 2009). LoRA modules are inserted into the query and value projections with rank $r = 10$. We use $r = 30$ on DomainNet because of its larger class space. ECR-LoRA follows the CL-LoRA prototype-based cosine classifier, while reproduced baselines retain their original classifier protocols. Our default configuration adopts a 6/6 core–residual split and is trained with SGD and cosine annealing. Entropy-guided core-prior inference is used unless otherwise stated. Experiments were conducted on a single NVIDIA A40 GPU, and all reproduced results are averaged over three random seeds.

4.3 Comparison with State-of-the-Art Methods

We compare ECR-LoRA under the rehearsal-free class-incremental learning setting with representative PEFT-based continual learning approaches, including prompt-based methods L2P (Wang et al., 2022b), DualPrompt (Wang et al., 2022a), and CODA-Prompt (Smith et al., 2023); the expandable subspace-based method EASE (Zhou et al., 2024); and recent LoRA-based methods InfLoRA (Liang & Li, 2024), SD-LoRA (Wu et al., 2025), CL-LoRA (He et al., 2025), BiLoRA (Zhu et al., 2025), EWC-LoRA (Zheng et al., 2026), SplitLoRA (Qiu et al., 2026), and Janus-LoRA (Chen et al., 2026).

Table 1 summarizes the comparison on four class-incremental learning benchmarks. ECR-LoRA achieves the best A_{Last} on all four benchmarks and the best A_{Avg} on three. Compared with EWC-LoRA, it improves A_{Last} and A_{Avg} by an average of 3.80 and 3.00 points, respectively. Importantly, the gains are not confined to one baseline family: ECR-LoRA consistently improves subspace-constrained methods such as InfLoRA and SplitLoRA, and shared or regularized approaches such as CL-LoRA and EWC-LoRA wherever results are available. This breadth supports our motivation that neither indiscriminate merging nor restrictive update separation fully resolves the tension between knowledge reuse and task-specific preservation.

Table 2: Component ablation on three benchmarks. Idx-1 denotes the independent residual bank with All-scan inference. Idx-1 to Idx-5 use All-scan inference, while Idx-6 additionally enables core-prior inference (CP). CER and RMC denote residual specificity and cross-residual compatibility, respectively. We report A_{Last} (%) averaged over three seeds.

	Shared	Residual specialization		Inference	$A_{\text{Last}} \uparrow$		
Index	Core	CER	RMC	CP	ImageNet-A	ImageNet-R	CIFAR-100
1					55.02±0.42	74.80±0.19	86.89±0.21
2	✓				60.22±0.66	79.91±0.83	86.71±0.18
3	✓	✓			60.76±1.29	80.25±0.43	86.86±0.21
4	✓		✓		61.31±1.17	80.29±0.41	87.08±0.25
5	✓	✓	✓		62.29±1.17	80.65±0.19	87.35±0.18
6	✓	✓	✓	✓	**64.18±1.01**	**81.24±0.39**	**89.22±0.12**

The gains are particularly pronounced on ImageNet-R and ImageNet-A, where larger distribution shifts make continual adaptation more challenging. ECR-LoRA improves over EWC-LoRA by 8.38 and 4.29 points in A_{Last} on the two benchmarks, respectively. On these shifted benchmarks, the residual branch preserves task-local evidence not adequately captured by pretrained features, while the evolving Core LoRA continues to accumulate reusable knowledge across tasks. Entropy-guided core-prior inference further improves ECR-LoRA (All-scan) across all four benchmarks, indicating that the consolidated core also provides a useful common reference for comparing residual experts at inference.

Figure 2 presents the task-wise accuracy evolution as more classes are introduced. Although some baselines achieve stronger performance during early sessions, their advantages gradually diminish as the number of tasks increases. ECR-LoRA exhibits slower performance degradation and achieves the highest final accuracy across all benchmarks, indicating that the evolving core preserves transferable knowledge while the residual bank continues to provide sufficient task-specific adaptation capacity. This behavior further supports the effectiveness of balancing shared knowledge accumulation and complementary residual preservation in continual learning.

4.4 Ablation Study

To further evaluate each component of ECR-LoRA, we conduct ablations starting from a fully independent residual-bank baseline (Idx-1), which keeps one adapter per task without cross-task sharing, as shown in Table 2.

Effectiveness of the Core–Residual organization. Our evolving core design(Idx-2) improves A_{Last} by 5.20% and 5.11% on ImageNet-A and ImageNet-R. We attribute these gains to selectively consolidating task-supported updates into reusable cross-task knowledge. CER (Idx-3) and RMC (Idx-4) further address residual specificity and cross-residual compatibility, respectively. Combining them (Idx-5) gives the strongest All-scan result on all three benchmarks, including a 2.07% gain on ImageNet-A. This supports their complementary roles: CER focuses residual capacity on core-unexplained evidence, while RMC improves robustness to historical-residual variation.

Effectiveness of core-prior inference. Building on the full training configuration, CP (Idx-6) further improves A_{Last} on all three benchmarks, without additional learned parameters or stored exemplars. This confirms the dual role of our core: beyond accumulating reusable knowledge during training, it also provides task-level reliability cues for calibrating residual competition at inference.

4.5 Analysis

Effect of core–residual depth allocation. Figure 3 examines how performance changes with the core–residual depth split. The balanced 6/6 configuration performs best on ImageNet-A and ImageNet-R under both all-scan and core-prior inference. The 0/12 setting provides no shared adaptation across tasks, thus we do not report the core-prior result. As more blocks are assigned to the

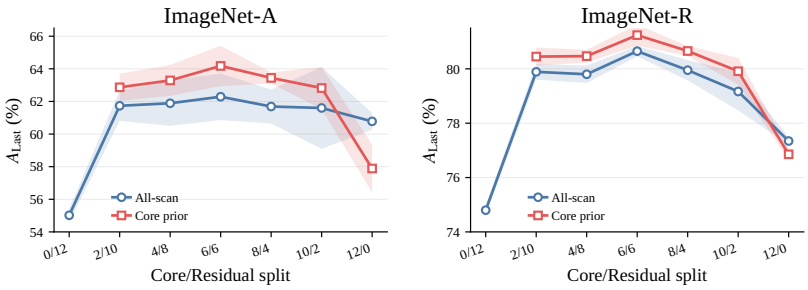


Figure 3: Effect of the core–residual depth allocation on ImageNet-A and ImageNet-R.

Table 3: Long-horizon comparison in A_{Last} (%).

Method	ImageNet-R 5S	ImageNet-R 20S	ImageNet-R 50S	ImageNet-A 20S
InfLoRA	77.52±0.37	71.01±0.45	61.86±0.78	46.81±2.30
SD-LoRA	78.25±0.92	71.26±0.77	67.60±1.22	49.12±1.78
BiLoRA	78.32±0.37	72.41±0.65	55.68±0.93	42.20±0.53
EWC-LoRA	76.36±0.21	70.18±1.06	46.48±1.00	48.87±1.21
CL-LoRA	81.59±0.73	77.58±0.47	71.71±0.67	54.38±0.73
SplitLoRA	**82.43±0.18**	78.57±0.55	71.24±0.12	55.31±1.31
ECR-LoRA	82.27±0.45	**78.97±0.47**	**71.73±0.26**	**57.67±1.28**

Table 4: Stability–plasticity comparison.

Data	Method	Stab.↑	Plast.↑	A_{Last} ↑
CIFAR100	Vanilla	87.56±0.09	**98.86±0.09**	82.99±0.84
	EWC-LoRA	94.45±0.59	97.99±0.50	87.91±0.57
	ECR-LoRA	**95.36±0.02**	94.21±0.15	**89.22±0.12**
ImageNet-R	Vanilla	78.63±1.38	**99.57±0.35**	64.87±0.73
	EWC-LoRA	**95.62±0.42**	93.23±0.34	72.86±0.79
	ECR-LoRA	93.40±0.37	97.25±0.36	**81.24±0.39**
ImageNet-A	Vanilla	85.56±1.82	**97.56±0.77**	40.01±1.32
	EWC-LoRA	**89.52±1.14**	92.78±1.91	59.89±0.26
	ECR-LoRA	87.11±1.24	96.33±0.90	**64.18±1.01**

core, less depth remains available for task-specific residual adaptation. Performance also decreases markedly in the fully shared 12/0 setting. Neither complete isolation nor complete sharing provides the desired balance. This comparison therefore reveals how LoRA adaptation should be organized across transformer depth. The peak at 6/6 supports our split-depth design, which uses depth as an explicit boundary between reusable and task-local knowledge.

Long-horizon robustness. We examine whether the Core–Residual organization remains effective as the task sequence becomes longer, where retaining reusable knowledge without restricting later adaptation becomes increasingly difficult. Table 3 shows that several strong methods are comparable at the short 5-session horizon, while ECR-LoRA maintains the highest mean accuracy as ImageNet-R is extended to 20 and 50 sessions and remains clearly competitive on 20-session ImageNet-A. In contrast, methods such as EWC-LoRA and BiLoRA deteriorate much more sharply with increasing horizon. This trend suggests that the persistent Core can accumulate reusable knowledge without exhausting the task-specific capacity needed for later adaptation.

Stability–plasticity balance. We further examine whether ECR-LoRA improves final accuracy by favoring stability or plasticity. As shown in Table 4, ECR-LoRA retains higher plasticity than EWC-LoRA on ImageNet-R and ImageNet-A at some stability cost, while achieving the strongest stability on CIFAR-100. Despite these different trade-offs, it attains the highest final accuracy on all three benchmarks, supporting the intended Core–Residual balance between knowledge preservation and new-task adaptation.

5 Conclusion

We presented ECR-LoRA, a Core–Residual framework for rehearsal-free continual low-rank adaptation. An evolving fixed-rank core accumulates reusable knowledge, while complementary residuals preserve task-specific evidence that should remain outside the shared state. The same core further serves as a task-agnostic reference for residual calibration at inference. Experiments on four class-incremental benchmarks demonstrate consistent gains over representative PEFT baselines and a favorable stability–plasticity trade-off.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproducibility of ECR-LoRA. The model architecture and learning objectives are described in Section 3, while the complete task-wise training procedure and theoretical derivations are provided in Appendix A. Dataset construction, task splits, training configurations, hyperparameters, evaluation metrics, and the computing environment are documented in Appendix B. Additional sensitivity analyses, computational-cost measurements, and design comparisons are reported in Appendices C–D. We also provide an anonymous code package containing the implementation, configuration files, fixed data splits, and dependency specification used for the reported experiments.

AI USE STATEMENT

In this work, we used generative AI tools (CHATGPT and CLAUDE) for tasks requiring disclosure. These tools were used to discuss and stress-test the theoretical framing of the core–residual decomposition; to check the correctness and completeness of mathematical derivations in Appendix A.2–A.3; to discuss working hypotheses and the design of ablation, sensitivity, and diagnostic experiments; to assist with selected figure-plotting scripts and parts of the residual-objective code; and to discuss and refine interpretations of the reported experimental results. We also used them to assist with translation between Chinese and English and to check that the manuscript accurately described the reported numbers.

For tasks with recommended disclosure, we used generative AI tools to assist with literature search, related-work summarization, brainstorming, manuscript editing for readability, and bibliographic formatting.

All AI-assisted outputs were reviewed by the authors. The authors independently checked the final mathematical derivations, experimental results, cited references, and AI-assisted code used in this work. All reported numerical results were produced by our own implementation and experiments. The authors made the final decisions regarding the research questions, method design, experimental setup, interpretation of results, and manuscript content, and take responsibility for all claims and artifacts in the paper.

REFERENCES

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9650–9660, 2021.

Cheng Chen, Pengpeng Zeng, Yuyu Guo, Lianli Gao, Hengtao Shen, and Jingkuan Song. Janus-LoRA: A balanced low-rank adaptation for continual learning. In *International Conference on Machine Learning*, 2026.

Quan Cheng, Yuanyu Wan, Lingyu Wu, Chenping Hou, and Lijun Zhang. Continuous subspace optimization for continual learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Jiangpeng He, Zhihao Duan, and Fengqing Zhu. CL-LoRA: Continual low-rank adaptation for rehearsal-free class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 30534–30544, 2025.

Lingfeng He, De Cheng, Huaijie Wang, Xi Yang, Nannan Wang, and Xinbo Gao. Task-driven subspace decomposition for knowledge sharing and isolation in LoRA-based continual learning. In *International Conference on Machine Learning*, 2026.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8340–8349, 2021a.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15262–15271, 2021b.

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*, volume 97, pp. 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. In *International Conference on Learning Representations*, 2022.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Mei Li, Yuxiang Lu, Qinyan Dai, Suizhi Huang, Yue Ding, and Hongtao Lu. BECAME: Bayesian continual learning with adaptive model merging. In *International Conference on Machine Learning*, volume 267, pp. 35481–35501. PMLR, 2025.

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

Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. Weighted ensemble models are strong continual learners. In *European Conference on Computer Vision*, pp. 306–324. Springer, 2024.

German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1406–1415, 2019.

Haomiao Qiu, Miao Zhang, Ziyue Qiao, and Liqiang Nie. Train with perturbation, infer after merging: A two-stage framework for continual learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Haomiao Qiu, Miao Zhang, Ziyue Qiao, Weili Guan, Min Zhang, and Liqiang Nie. SplitLoRA: Balancing stability and plasticity in continual learning through gradient space splitting. In *International Conference on Learning Representations*, 2026.

James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. CODA-Prompt: COntinual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11909–11919, 2023.

Xiequn Wang, Zhan Zhuang, and Yu Zhang. PLAN: Proactive low-rank allocation for continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2909–2918, 2025.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, pp. 631–648. Springer, 2022a.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 139–149, 2022b.

Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. SD-LoRA: Scalable decoupled low-rank adaptation for class incremental learning. In *International Conference on Learning Representations*, 2025.

Yaoyue Zheng, Yin Zhang, Joost van de Weijer, Gido M. van de Ven, Shaoyi Du, Xuetao Zhang, and Zhiqiang Tian. Revisiting weight regularization for low-rank continual learning. In *International Conference on Learning Representations*, 2026.

Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23554–23564, 2024.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations*, 2022.

Hao Zhu, Yifei Zhang, Junhao Dong, and Piotr Koniusz. BiLoRA: Almost-orthogonal parameter spaces for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25613–25622, 2025.

APPENDIX

A ADDITIONAL METHOD DETAILS

A.1 TRAINING ALGORITHM

Algorithm A1 summarizes the task-wise training procedure of ECR-LoRA. For task t, the trainable parameters Θ_t include the shared candidate up-projections, the current residual adapter, and the current classifier head. Objectives requiring a previous core or historical residuals are omitted for the first task.

Algorithm A1 Task-wise training pipeline of ECR-LoRA

Require: Tasks $\{\mathcal{D}_t\}_{t=1}^{T}$; frozen backbone $f(\cdot; W)$; branch sets $\mathcal{B}_c, \mathcal{B}_r$; LoRA rank r
Ensure: Core C_T, residual bank $\mathcal{R}_{1:T}$, and classifier heads $\mathcal{G}_{1:T}$

Initialize $C_0 \leftarrow \mathbf{0}$, $\bar{F}_0^{\text{mem}} \leftarrow \mathbf{0}$, $\mathcal{R} \leftarrow \varnothing$, and $\mathcal{G} \leftarrow \varnothing$
for $t = 1, \ldots, T$ **do**
  ▷ *Current-task initialization*
    Initialize A_t^{new} from the previous core if $t > 1$; otherwise use standard LoRA initialization
    Set $B_t^{\text{new}} \leftarrow \mathbf{0}$ and freeze A_t^{new}
    Initialize residual R_t and classifier head g_t
    Freeze C_{t-1}, $\mathcal{R}_{1:t-1}$, and $\mathcal{G}_{1:t-1}$
  ▷ *Joint core–residual optimization*
    for each mini-batch $\mathcal{B} \subset \mathcal{D}_t$ **do**
      Compute the shared-only feature h_{sh} and full feature h_t
      Compute $\mathcal{L}_{\text{CE}}^{(t)}$ and $\mathcal{L}_{\text{sar}}^{(t)}$, where $\mathcal{L}_{\text{sar}}^{(1)} = 0$
      Construct weights from shared-only errors and compute $\mathcal{L}_{\text{cer}}^{(t)}$ on the full path
      if $t > 1$ **then**
        Sample $R_p \in \mathcal{R}_{1:t-1}$ and compute $\mathcal{L}_{\text{mix}}^{(t)}$ using a residual-mixed feature
      else
        Set $\mathcal{L}_{\text{mix}}^{(t)} \leftarrow 0$
      end if
      Update Θ_t using the objective in Eq. (9)
    end for
  ▷ *Fisher-guided core consolidation*
    for each shared branch $j \in \mathcal{B}_c$ **do**
      Estimate the Fisher support of $B_{t,j}^{\text{new}}$
      Compute the absorption coefficient and consolidate the candidate according to Eqs. (3)–(4)
      Update the Fisher memory by EMA
    end for
    Prototype-replace and freeze g_t; append R_t to $\mathcal{R}$ and g_t to $\mathcal{G}$
end for
return C_T, $\mathcal{R}_{1:T}$, and $\mathcal{G}_{1:T}$

A.2 DERIVATION OF CORE LORA EVOLUTION

Shared-core evolution requires three related decisions: which directions a current-task candidate should reuse, how strongly the optimized candidate should be absorbed, and how to retain the result under a fixed-rank budget. We first show that the subspace-compatible objective measures the candidate's squared distance to the output subspace of the previous core. We then derive the Fisher-guided absorption coefficient from a strictly convex precision-weighted surrogate. Finally, we interpret the rank-r refactorization as an optimal low-rank projection.

Notation. For a shared LoRA branch $j \in \mathcal{B}_c$, let

$$\begin{aligned} C_{t-1,j} &= B_{t-1,j}^{c} A_{t-1,j}^{c}, \\ \Delta_{t,j}^{\text{new}} &= B_{t,j}^{\text{new}} A_{t,j}^{\text{new}}. \end{aligned} \tag{A1}$$

This follows the standard LoRA parameterization (Hu et al., 2022). For $t > 1$, $A_{t,j}^{\text{new}}$ is initialized from the consolidated core and remains frozen, whereas $B_{t,j}^{\text{new}}$ is optimized. The candidate after task-t optimization is denoted by $\widehat{\Delta}_{t,j}^{\text{new}}$.

Let

$$q_j = \operatorname{rank}(C_{t-1,j}) \le r. \tag{A2}$$

The compact singular value decomposition is

$$C_{t-1,j} = U_{t-1,j}\Sigma_{t-1,j}V_{t-1,j}^\top, \tag{A3}$$

where $U_{t-1,j} \in \mathbb{R}^{d_{\text{out}} \times q_j}$ and $U_{t-1,j}^\top U_{t-1,j} = I_{q_j}$. The columns of $U_{t-1,j}$ span the output subspace represented by the previous core. Its orthogonal-complement projector is

$$P_{t-1,j} = I_{d_{\text{out}}} - U_{t-1,j}U_{t-1,j}^\top. \tag{A4}$$

Proposition A1 (Subspace-distance interpretation). For any matrix $\Delta \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$,

$$\|P_{t-1,j}\Delta\|_F^2 = \min_Z \|\Delta - U_{t-1,j}Z\|_F^2, \tag{A5}$$

where $Z \in \mathbb{R}^{q_j \times d_{\text{in}}}$. The unique minimizer is

$$Z^\star = U_{t-1,j}^\top \Delta. \tag{A6}$$

Proof. For any Z,

$$\begin{aligned}\Delta - U_{t-1,j}Z = P_{t-1,j}\Delta \\ + U_{t-1,j}\left(U_{t-1,j}^\top\Delta - Z\right).\end{aligned} \tag{A7}$$

The two terms are orthogonal because $P_{t-1,j}U_{t-1,j} = 0$. Hence,

$$\begin{aligned}\|\Delta - U_{t-1,j}Z\|_F^2 = \|P_{t-1,j}\Delta\|_F^2 \\ + \left\|U_{t-1,j}^\top\Delta - Z\right\|_F^2.\end{aligned} \tag{A8}$$

The second term is uniquely minimized at $Z = U_{t-1,j}^\top\Delta$, which proves the result. □

The subspace-compatible objective from the main paper is

$$\mathcal{L}_{\text{sar}}^{(t)} = \frac{1}{|\mathcal{B}_c|}\sum_{j\in\mathcal{B}_c}\left\|P_{t-1,j}\Delta_{t,j}^{\text{new}}\right\|_F^2. \tag{A9}$$

Proposition A1 shows that every summand is the squared distance from the candidate to the output subspace of the previous core. Thus, the loss encourages reuse of represented directions without hard-constraining the candidate to that subspace. For $t = 1$, the term is omitted because no previous core is available.

Precision-weighted candidate absorption. Subspace-compatible learning determines the geometry of the candidate, but not how much of it should enter the persistent core. Let $\bar{F}_{t,j} \ge 0$ and $\bar{F}_{t-1,j}^{\text{mem}} \ge 0$ denote the normalized current and historical support of branch j, respectively. In the implementation, these values are obtained from normalized mini-batch squared gradients of the trainable candidate up-projection. They provide a practical Fisher-like importance approximation, following the use of Fisher information in continual low-rank adaptation (Zheng et al., 2026).

Proposition A2 (Precision-weighted absorption). Let $m_{t,j}$ denote the candidate increment absorbed into branch j. For $\epsilon > 0$, consider

$$\begin{aligned}\Omega_{t,j}(m) = \frac{\bar{F}_{t-1,j}^{\text{mem}} + \epsilon}{2}\|m\|_F^2 \\ + \frac{\bar{F}_{t,j}}{2}\left\|m - \widehat{\Delta}_{t,j}^{\text{new}}\right\|_F^2.\end{aligned} \tag{A10}$$

This objective is strictly convex and has the unique minimizer

$$m_{t,j}^\star = \eta_{t,j}\widehat{\Delta}_{t,j}^{\text{new}}, \tag{A11}$$

where

$$\eta_{t,j} = \frac{\bar{F}_{t,j}}{\bar{F}_{t-1,j}^{\text{mem}} + \bar{F}_{t,j} + \epsilon}. \tag{A12}$$

Proof. Differentiating Eq. (A10) with respect to m gives

$$\nabla_m \Omega_{t,j} = \left(\bar{F}^{\mathrm{mem}}_{t-1,j} + \bar{F}_{t,j} + \epsilon\right) m - \bar{F}_{t,j} \widehat{\Delta}^{\mathrm{new}}_{t,j}. \tag{A13}$$

Since

$$\bar{F}^{\mathrm{mem}}_{t-1,j} + \bar{F}_{t,j} + \epsilon > 0,$$

the quadratic objective is strictly convex. Setting its gradient to zero gives Eqs. (A11)–(A12). □

The implementation evaluates this closed form directly rather than running an additional optimization procedure. Equation (A12) corresponds to Eq. (3).

Corollary A1 (Range and monotonicity). Define

$$D_{t,j} = \bar{F}^{\mathrm{mem}}_{t-1,j} + \bar{F}_{t,j} + \epsilon. \tag{A14}$$

Then $0 \le \eta_{t,j} < 1$, and

$$\frac{\partial \eta_{t,j}}{\partial \bar{F}_{t,j}} = \frac{\bar{F}^{\mathrm{mem}}_{t-1,j} + \epsilon}{D^2_{t,j}} \ge 0, \qquad \frac{\partial \eta_{t,j}}{\partial \bar{F}^{\mathrm{mem}}_{t-1,j}} = -\frac{\bar{F}_{t,j}}{D^2_{t,j}} \le 0. \tag{A15}$$

Hence, stronger current-task support increases candidate absorption, whereas stronger historical support makes consolidation more conservative.

The historical support is maintained by an exponential moving average:

$$\bar{F}^{\mathrm{mem}}_{t,j} = \rho \bar{F}^{\mathrm{mem}}_{t-1,j} + (1-\rho)\bar{F}_{t,j}, \qquad 0 \le \rho < 1. \tag{A16}$$

This stores a constant-size summary rather than separate support statistics for every previous task.

Rank-preserving consolidation. After absorption, define

$$M_{t,j} = C_{t-1,j} + \eta_{t,j} \widehat{\Delta}^{\mathrm{new}}_{t,j}. \tag{A17}$$

The persistent core branch is

$$C_{t,j} = \Pi_r(M_{t,j}), \tag{A18}$$

where Π_r retains the largest r singular components. By the Eckart–Young–Mirsky theorem,

$$\Pi_r(M_{t,j}) = \underset{\mathrm{rank}(X) \le r}{\arg\min} \; \|M_{t,j} - X\|^2_F. \tag{A19}$$

Thus, the persistent core is a best rank-r approximation of the precision-weighted merge under the Frobenius norm. Each shared-core branch retains storage complexity $\mathcal{O}(r(d_{\mathrm{in}} + d_{\mathrm{out}}))$, independent of the number of tasks. The task-wise residual bank is stored separately and grows with the task sequence.

Discussion. The derivation separates the three roles in shared-core evolution. Subspace-compatible learning determines *where* the candidate is encouraged to update. The precision-weighted coefficient determines *how much* of the optimized candidate is retained. Rank-r refactorization determines *how* the consolidated information is stored under a fixed budget. The complete update is

$$C_{t,j} = \Pi_r\left(C_{t-1,j} + \eta_{t,j} \widehat{\Delta}^{\mathrm{new}}_{t,j}\right), \tag{A20}$$

which corresponds to Eq. (4). The projector $P_{t-1,j}$ acts during candidate learning and does not appear in the denominator of $\eta_{t,j}$, matching the implementation used in the reported experiments.

A.3 Properties of Entropy-Guided Core-Prior Calibration

We next establish two properties of the calibration rule in Eqs. (10)–(12): the normalized core uncertainty is bounded, and the core prior preserves the within-expert class ranking.

For residual expert s, let

$$p_s^c(x) = \mathrm{softmax}\left([z^c(x)]_{\mathcal{Y}_s}\right) \tag{A21}$$

be the core-only distribution restricted to its class subset $\mathcal{Y}_s$, where $|\mathcal{Y}_s| \geq 2$. The normalized uncertainty is

$$e_s(x) = \frac{H(p_s^c(x))}{\log |\mathcal{Y}_s|}. \tag{A22}$$

Since the entropy of a distribution over $|\mathcal{Y}_s|$ classes satisfies

$$0 \leq H(p_s^c(x)) \leq \log |\mathcal{Y}_s|, \tag{A23}$$

we immediately obtain

$$0 \leq e_s(x) \leq 1. \tag{A24}$$

Thus, the calibration applied to expert s is bounded by $[0, \beta_H]$.

Let

$$p_{s,c}(x) = \left[\mathrm{softmax}(z_s(x))\right]_c \tag{A25}$$

be the class score produced by residual expert s. The calibrated score is

$$S_{s,c}(x) = p_{s,c}(x) - \beta_H e_s(x), \qquad c \in \mathcal{Y}_s. \tag{A26}$$

Proposition A3 (Within-expert ranking invariance). For any two classes $c_1, c_2 \in \mathcal{Y}_s$,

$$S_{s,c_1}(x) - S_{s,c_2}(x) = p_{s,c_1}(x) - p_{s,c_2}(x). \tag{A27}$$

Consequently,

$$p_{s,c_1}(x) > p_{s,c_2}(x) \iff S_{s,c_1}(x) > S_{s,c_2}(x). \tag{A28}$$

Proof. The entropy term $e_s(x)$ depends on the expert index s but is constant for all classes in $\mathcal{Y}_s$. Therefore,

$$\begin{aligned} S_{s,c_1}(x) - S_{s,c_2}(x) &= [p_{s,c_1}(x) - \beta_H e_s(x)] \\ &\quad - [p_{s,c_2}(x) - \beta_H e_s(x)] \\ &= p_{s,c_1}(x) - p_{s,c_2}(x), \end{aligned} \tag{A29}$$

which proves the result. □

The core prior therefore does not replace the class discrimination learned by each residual expert. Instead, it applies an expert-level shift that only changes competition across task-specific residual paths. A low-entropy core prediction produces a smaller penalty, whereas an uncertain core prediction suppresses the corresponding expert more strongly.

B Implementation Details and Reproducibility

B.1 Dataset and Task Splits

We follow the rehearsal-free class-incremental learning protocol used in the main paper. At each session, only samples from the current class subset are available for training. At inference, the model predicts over all classes observed so far without task identities or stored exemplars.

CIFAR-100 (Krizhevsky, 2009), ImageNet-R (Hendrycks et al., 2021a), and ImageNet-A (Hendrycks et al., 2021b) are divided into 10 sessions. DomainNet (Peng et al., 2019) is divided into 5 sessions. Table A1 summarizes the resulting class-incremental splits.

CIFAR-100 uses its standard 50,000/10,000 train–test split. For ImageNet-R and ImageNet-A, we use fixed stratified 80/20 manifests, containing 24,000/6,000 and 6,000/1,500 training/test images,

Table A1: Class-incremental benchmark configurations. All sessions contain disjoint class subsets.

Dataset	Classes	Sessions	Cls./session
CIFAR-100	100	10	10
ImageNet-R	200	10	20
ImageNet-A	200	10	20
DomainNet	345	5	69

Table A2: Dataset-specific hyperparameters for ECR-LoRA. All settings use SGD with momentum 0.9, cosine annealing, and a 6/6 core–residual split. "lr" and "wd" denote the initial learning rate and weight decay; λ_{sar}, λ_{cer}, and λ_{mix} weight subspace-compatible learning, core-error reweighting, and residual-mix consistency, respectively.

Dataset	Batch	Epochs	Rank	lr	wd	λ_{sar}	λ_{cer}	λ_{mix}
CIFAR-100	128	30	10	0.03	1×10^{-4}	0.50	0.20	0.50
DomainNet	128	15	30	0.04	1×10^{-4}	0.20	0.10	0.25
ImageNet-R	32	20	10	0.05	5×10^{-4}	0.20	0.20	0.50
ImageNet-A	32	25	10	0.05	5×10^{-3}	1.00	0.20	0.50

respectively. The DomainNet split contains 338,804 training images and 84,702 test images. All fixed split files are included in the anonymous code package.

CIFAR-100 and DomainNet use fixed class orders. For ImageNet-R and ImageNet-A, the class order is randomly permuted by the training seed. Within each seed, the same class order and data split are used for all compared methods.

B.2 Training Configuration

All experiments use the frozen ViT-B/16 backbone and query/value LoRA placement described in the main paper. The default core/residual depth split is 6/6. We use SGD with momentum 0.9 and cosine annealing with a minimum learning rate of zero. A single learning rate is applied to all trainable parameters. Dataset-specific configurations are summarized in Table A2.

All reported ECR-LoRA results are averaged over three random seeds, $\{0, 1, 2\}$. Unless otherwise specified, the full model uses entropy-guided core-prior inference with $\beta_H = 4$. The all-scan variant uses the same trained model without core-prior calibration. Remaining implementation constants and training switches are specified in the JSON configuration files provided in the anonymous code package.

Computing environment. Experiments were conducted on a Linux server equipped with two NVIDIA A40 GPUs, while each individual run used a single GPU. The reference software environment consists of Python 3.8.20, PyTorch 1.11.0, CUDA runtime 11.3, torchvision 0.12.0, and timm 0.6.7. The complete dependency specification is included as `environment.yaml` in the anonymous code package.

B.3 Accuracy and Stability–Plasticity Metrics

Let $A_{t,i}$ denote the classification accuracy on the class subset introduced in session i after training through session t, where $1 \leq i \leq t \leq T$. The average accuracy after session t is

$$A_t = \frac{1}{t} \sum_{i=1}^{t} A_{t,i}. \tag{A30}$$

Final and average accuracy are then defined as

$$A_{\text{Last}} = A_T,$$
$$A_{\text{Avg}} = \frac{1}{T}\sum_{t=1}^{T} A_t. \tag{A31}$$

Following EWC-LoRA (Zheng et al., 2026), we additionally report normalized stability and plasticity. For each non-final session i, let

$$A_i^{\max} = \max_{i \le t < T} A_{t,i} \tag{A32}$$

denote its highest accuracy before the final evaluation. Stability is defined as one minus the average normalized forgetting:

$$S = 1 - \frac{1}{T-1}\sum_{i=1}^{T-1} \frac{A_i^{\max} - A_{T,i}}{A_i^{\max}}. \tag{A33}$$

A higher S indicates that the performance attained on previous sessions is better preserved after learning the complete sequence.

Plasticity measures how well each new session is learned relative to a task-specific reference:

$$P = \frac{1}{T}\sum_{i=1}^{T} \frac{A_{i,i}}{A_{i,i}^{\text{ref}}}. \tag{A34}$$

Here, $A_{i,i}$ is the accuracy on session i immediately after learning it, and $A_{i,i}^{\text{ref}}$ is obtained by fine-tuning the same backbone exclusively on session i. A higher P indicates stronger adaptation to newly introduced classes. Following EWC-LoRA, we report both S and P after multiplying them by 100.

C Sensitivity and Design Analysis

C.1 LoRA Rank Sensitivity

ECR-LoRA uses a fixed LoRA rank for both the evolving shared core and the task-specific residual adapters. We evaluate $r \in \{4, 8, 10, 16, 20, 30\}$ on ImageNet-A and ImageNet-R while keeping the frozen backbone, core–residual depth split, training objectives, and Core-prior inference strategy unchanged. The same rank is used by the shared core and all residual adapters throughout the incremental sequence. Results are averaged over three random seeds. In addition to accuracy, we report the number of trainable LoRA parameters, the accumulated adapter storage after ten sessions, and the Core-prior inference cost. The trainable parameter count includes the evolving core and the current task residual, whereas storage includes the consolidated shared core and all ten stored task residuals.

Table A3 shows that ECR-LoRA remains stable across a broad range of LoRA ranks. Across $r = 4$ to 30, the variation in A_{Last} is only 1.25 points on ImageNet-A and 1.47 points on ImageNet-R. The corresponding variation in A_{Avg} remains below 1.0 point on ImageNet-A and approximately 1.3 points on ImageNet-R. Even the low-capacity $r = 4$ configuration retains competitive performance, indicating that the method does not rely on an unusually large adaptation rank. Increasing the rank also does not produce a consistent monotonic improvement. The default $r = 10$ obtains the best final and average accuracy on ImageNet-A. On ImageNet-R, $r = 30$ improves over $r = 10$ by only 0.11 points in A_{Last} and 0.33 points in A_{Avg}. In contrast, the larger rank reduces ImageNet-A performance by 0.80 and 0.98 points, respectively. The preferred rank is therefore dataset dependent, and simply increasing adapter capacity does not consistently improve continual adaptation.

The capacity comparison further supports the use of $r = 10$. Increasing the rank from 10 to 30 triples the accumulated adapter storage from 7.73 MiB to 23.20 MiB and increases the number of trainable LoRA parameters from 0.369M to 1.106M. Although the corresponding Core-prior inference cost grows by only approximately 1.16%, the additional storage and trainable capacity provide no consistent cross-dataset advantage. We therefore select $r = 10$ as a robust operating point that achieves near-optimal performance while retaining a compact continual adapter bank.

Table A3: LoRA rank sensitivity of ECR-LoRA on ImageNet-A and ImageNet-R. Storage is measured in FP32 after ten sessions, and GFLOPs correspond to Core-prior inference over ten residual experts. Results are reported as mean±standard deviation over three seeds; † denotes the default rank.

	Capacity and computation			ImageNet-A		ImageNet-R	
Rank	Params (M)	Storage (MiB)	GFLOPs	A_{Last} ↑	A_{Avg} ↑	A_{Last} ↑	A_{Avg} ↑
4	0.147	3.09	205.49	62.93±0.59	71.29±0.75	79.88±0.29	84.82±0.95
8	0.295	6.19	205.96	63.73±0.96	71.28±0.46	80.52±0.41	85.30±1.20
10†	0.369	7.73	206.20	**64.18±1.01**	**71.95±0.94**	81.24±0.39	85.80±1.11
16	0.590	12.38	206.92	63.51±1.29	71.82±0.92	81.18±0.44	85.79±1.26
20	0.737	15.47	207.40	63.53±1.07	71.48±1.79	81.20±0.33	85.83±1.11
30	1.106	23.20	208.60	63.38±1.19	70.97±1.13	**81.35±0.50**	**86.13±1.14**

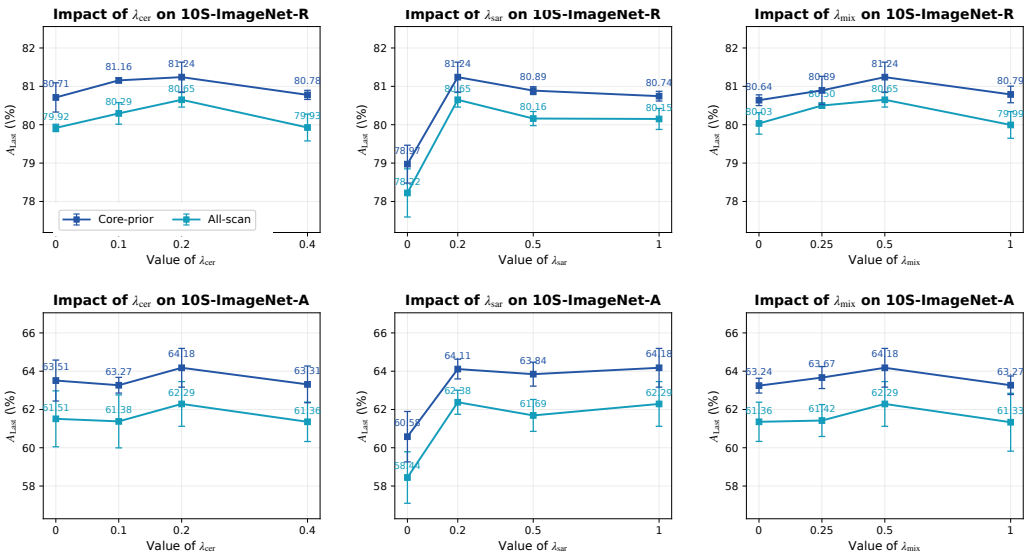


Figure A1: Sensitivity of ECR-LoRA to λ_{cer}, λ_{sar}, and λ_{mix} on 10-session ImageNet-R and ImageNet-A. Curves report A_{Last} under Core-prior and All-scan inference, and error bars denote the standard deviation over three random seeds.

C.2 Loss Weight Sensitivity

We examine the sensitivity of ECR-LoRA to the weights of the core-error reweighting, subspace-alignment regularization, and residual-mix consistency objectives, denoted by λ_{cer}, λ_{sar}, and λ_{mix}, respectively. Each coefficient is varied independently on 10-session ImageNet-R and ImageNet-A, while the remaining coefficients and all other training settings are kept fixed. We report A_{Last} under both All-scan and Core-prior inference, averaged over three random seeds.

Figure A1 shows that the three objectives play complementary but different roles. Moderate values of λ_{cer} and λ_{mix} consistently improve both inference strategies, whereas assigning excessive weight to either objective partially reverses the gain. This behavior is expected because these terms are intended to refine residual specialization rather than dominate the classification objective. Importantly, the curves form relatively broad plateaus around their best values. The small differences among nearby nonzero settings are generally comparable to the variation across random seeds, indicating that the method does not depend on narrowly tuned coefficients.

Table A4: Storage and inference cost as the number of residual experts increases. Overhead denotes the relative cost of Core-prior over All-scan. Latency is reported as mean±standard deviation over five measurement repetitions.

			GFLOPs/sample			Latency (ms/sample)		
Sessions	Residuals	Storage (MiB)	All-scan	Core-prior	Overhead	All-scan	Core-prior	Overhead
5	5	4.22	93.729	112.475	20.00%	59.69±0.28	72.41±0.08	21.32%
10	10	7.73	187.458	206.203	10.00%	118.74±0.09	132.61±0.19	11.68%
20	20	14.77	374.915	393.661	5.00%	238.75±0.19	254.34±0.47	6.53%

The effect of λ_{sar} is more pronounced. Removing subspace-alignment regularization reduces the best nonzero Core-prior result by 2.27 points on ImageNet-R and 3.60 points on ImageNet-A. The corresponding reductions under All-scan inference are 2.43 and 3.94 points. Once the regularizer is enabled, however, performance remains stable over a relatively wide range of nonzero weights. This contrast suggests that subspace alignment is important for preventing unconstrained core updates, while its exact strength is less critical after sufficient regularization is introduced. Core-prior also consistently outperforms All-scan across all tested weights, showing that its inference benefit is preserved under substantial changes to the training objectives. Overall, the selected coefficients lie in stable high-performing regions rather than isolated optima.

D Additional Analysis

D.1 Inference Cost Analysis

We quantify the inference and storage costs associated with the growing residual bank. All measurements use the final ImageNet-R models on a single NVIDIA A40 GPU in FP32 with batch size 1 and an input resolution of 224×224. GFLOPs are measured from the deployed forward path. Latency excludes data loading and host-to-device transfer and is reported over five repetitions, each containing 100 warm-up iterations and 500 synchronized measurements. Storage includes the consolidated shared core and all stored residual adapters in FP32, while excluding the frozen backbone, classifiers, optimizer states, and temporary training parameters.

Both inference strategies evaluate every stored residual expert. Consequently, All-scan computation and latency grow approximately in proportion to the number of sessions. Increasing the residual bank from 5 to 20 experts raises the All-scan cost from 93.729 to 374.915 GFLOPs and the measured latency from 59.69 to 238.75 ms per sample. The consolidated shared core remains fixed in size, while one task-specific residual is retained after each session. Adapter storage therefore grows from 4.22 MiB at 5 sessions to 14.77 MiB at 20 sessions. Core-prior retains the full residual scan and adds one shared-core calibration path rather than pruning residual experts. Its absolute computational increment remains approximately 18.75 GFLOPs across the evaluated horizons. The relative GFLOP overhead therefore decreases from 20.00% at 5 sessions to 5.00% at 20 sessions, while the measured latency overhead decreases from 21.32% to 6.53%. Thus, the calibration overhead becomes a progressively smaller fraction of the total inference cost as the residual bank grows. The approximately linear cost of evaluating all residual experts, however, remains the primary scalability limitation of the current task-agnostic inference.

D.2 Merge Strategy Comparison

We examine whether the performance of ECR-LoRA depends on the specific form of shared-core consolidation. All strategies use the same shared-core–residual architecture, residual objectives, rank, and core-prior inference protocol.

The default method uses the adaptive branch-level Fisher coefficient defined in Eq. (3). We additionally compare it with uniform task averaging, CoMA and CoFiMA (Marouf et al., 2024), and a rank-wise Fisher variant. Uniform averaging weights the current core proposal according to the number of observed tasks. CoMA uses a fixed averaging coefficient, whereas CoFiMA replaces the fixed average with Fisher-weighted model averaging. For CoMA and CoFiMA, we evaluate $w \in \{0.1, 0.3, 0.5\}$ and report the strongest setting, $w = 0.5$. Rank-wise Fisher applies separate

Table A5: Comparison of shared-core consolidation strategies on ImageNet-A. The default row uses the final branch-level Fisher configuration reported in the main paper. For CoMA and CoFiMA, the best result among $w \in \{0.1, 0.3, 0.5\}$ is reported. We show core-prior A_{Last} and A_{Avg} in percentage as mean±standard deviation over three seeds.

Strategy	Weighting level	$A_{\text{Last}} \uparrow$	$A_{\text{Avg}} \uparrow$
Default Fisher	Branch-adaptive	**64.18±1.01**	**71.95±0.94**
Uniform task average	Task-count	63.09±0.32	71.23±0.52
CoMA ($w = 0.5$)	Fixed scalar	63.62±0.55	71.34±0.46
CoFiMA ($w = 0.5$)	Fisher-weighted	63.33±0.41	71.41±0.48
Rank-wise Fisher	Per-rank adaptive	63.09±0.44	70.95±0.63

adaptive coefficients to individual low-rank components instead of sharing one coefficient within each LoRA branch.

The default branch-level Fisher strategy achieves the highest final and average accuracy. CoMA performs competitively with a fixed coefficient, but remains 0.56 points lower in A_{Last} and 0.61 points lower in A_{Avg}. CoFiMA improves slightly over uniform task averaging in A_{Avg}, but its finer Fisher-weighted averaging still does not outperform the proposed branch-level rule. The rank-wise Fisher variant also provides no additional benefit, despite using more granular importance estimates. These results suggest that uniform averaging is insufficiently adaptive, whereas component-level weighting may introduce unnecessary estimation variation. A single adaptive coefficient for each shared LoRA branch provides a simpler and more reliable balance between current-task support and historical core importance. We therefore use branch-level Fisher consolidation in all main experiments.

D.3 Complementary Residual Diagnostics

We provide compact diagnostics of how complementary residual learning changes the allocation and organization of task-specific knowledge. All diagnostics are computed on ImageNet-A after the final session and averaged over three seeds.

CER training-mass allocation. We define core-error samples as those misclassified by the shared-only path, $\mathcal{I}_{\text{err}} = \{i : \widehat{y}_i^{\text{sh}} \neq y_i\}$. We measure their frequency in the diagnostic set and their contribution to the CER-weighted classification objective:

$$S_{\text{sample}} = \frac{|\mathcal{I}_{\text{err}}|}{N}, \qquad S_{\text{mass}} = \frac{\sum_{i \in \mathcal{I}_{\text{err}}} \omega_i \ell_i}{\sum_{i=1}^{N} \omega_i \ell_i}. \tag{A35}$$

Here, N is the number of diagnostic samples, ℓ_i is the full-path cross-entropy loss, and ω_i is the CER weight defined in Eq. (5). Thus, S_{sample} measures how frequently the shared-only path fails, whereas S_{mass} measures the fraction of the weighted optimization objective contributed by these samples. Comparing the two quantities reveals whether CER concentrates residual learning on evidence that is not sufficiently captured by the shared core.

Residual diagnostic scores. We use two complementary scores to characterize the resulting residual bank. For each task residual s and residual LoRA branch j, let $Q_{s,j}$ be an orthonormal basis obtained by applying QR decomposition to the column space of its LoRA-B matrix. Let $\mathcal{B}_r$ denote the set of residual LoRA branches, and let $\mathcal{P} = \{(s, p) : 1 \leq s < p \leq T\}$ contain all unordered pairs of stored task residuals. Residual similarity is defined as

$$S_{\text{sim}} = \frac{1}{|\mathcal{P}|\,|\mathcal{B}_r|} \sum_{(s,p) \in \mathcal{P}} \sum_{j \in \mathcal{B}_r} \frac{\left\| Q_{s,j}^{\top} Q_{p,j} \right\|_F^2}{r_{s,p,j}}, \tag{A36}$$

where $r_{s,p,j}$ is the smaller effective rank of the two bases. Each normalized overlap lies in $[0, 1]$ and represents the mean squared cosine between the two residual subspaces. A lower S_{sim} therefore indicates that stored residuals occupy more distinct task-specific adaptation directions.

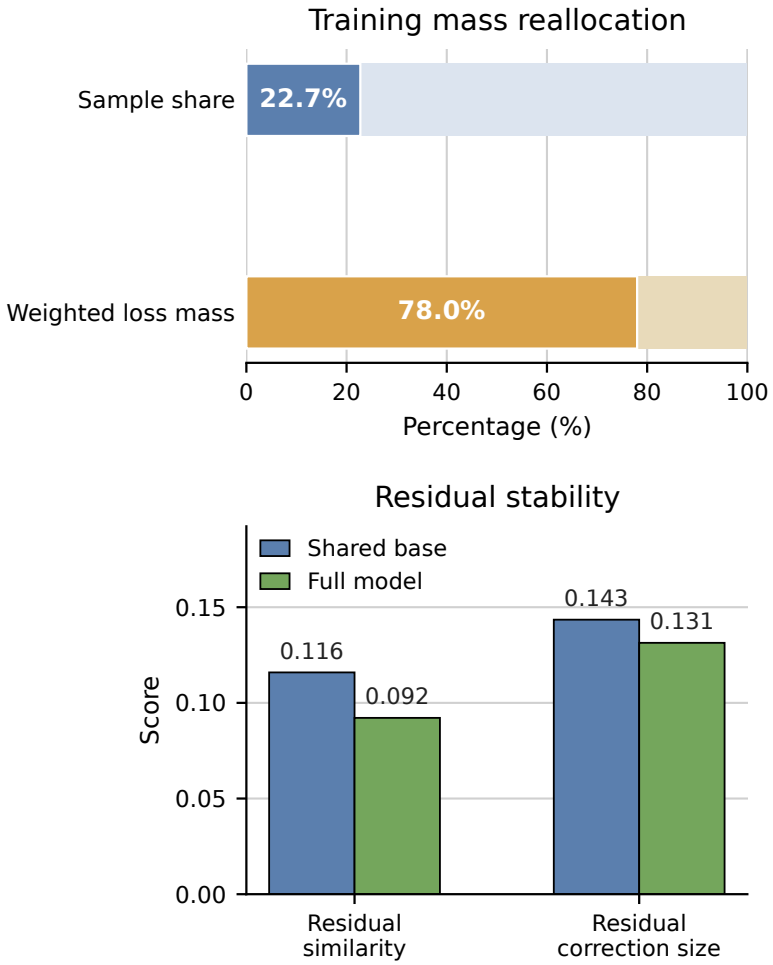


Figure A2: Complementary residual diagnostics on ImageNet-A. Top: core-error sample frequency and CER-weighted loss allocation. Bottom: residual similarity and hard-sample correction size for the shared-core base and the full model. Results are averaged over three seeds.

We additionally measure how strongly the residual path changes uncertain shared predictions. Let $z^{\mathrm{sh}}(x)$ and $z^{\mathrm{full}}(x)$ denote the shared-only and full-path logits over the evaluated class set. The shared-only classification margin is $m^{\mathrm{sh}}(x) = [z^{\mathrm{sh}}(x)]_y - \max_{c \neq y}[z^{\mathrm{sh}}(x)]_c$. Let $\mathcal{H}$ contain the bottom 30% of samples ranked by this margin. The residual correction size is

$$S_{\mathrm{corr}} = \frac{1}{|\mathcal{H}|} \sum_{x \in \mathcal{H}} \frac{\left\| z^{\mathrm{full}}(x) - z^{\mathrm{sh}}(x) \right\|_2}{\sqrt{|\mathcal{Y}_{1:T}|}}. \tag{A37}$$

The normalization by $\sqrt{|\mathcal{Y}_{1:T}|}$ converts the logit distance into a root-mean-square change per observed class. A lower S_{corr} indicates that residual adapters apply smaller and more controlled modifications around difficult shared predictions. Because S_{sim} and S_{corr} quantify different properties, their numerical values are compared only within the same metric.

As shown in Figure A2, core-error samples constitute only 22.7% of the diagnostic set but contribute 78.0% of the CER-weighted classification mass. CER therefore redirects residual optimization toward samples that remain insufficiently explained by the shared representation, rather than uniformly increasing the loss assigned to all samples.

Compared with the shared-core base, the full model reduces S_{sim} from 0.116 to 0.092. This result indicates clearer separation between the column subspaces of different task residuals. The full model

Table A6: Results on 10-session ImageNet-R with self-supervised ViT-B/16 initializations. ECR-LoRA results are averaged over three seeds. Bold and underline indicate the best and second-best results under each initialization. A dash denotes an unreported result.

Method	$A_{\text{Last}} \uparrow$	$A_{\text{Avg}} \uparrow$
DINO-1k		
InfLoRA	68.31 ± 0.28	76.15 ± 0.05
SD-LoRA	69.78 ± 0.63	75.45 ± 0.35
ECR-LoRA (All-scan)	$\underline{73.57 \pm 0.35}$	$\underline{79.62 \pm 0.89}$
ECR-LoRA	$\mathbf{74.07 \pm 0.28}$	$\mathbf{79.95 \pm 0.92}$
iBOT-1k		
InfLoRA	71.84 ± 0.09	78.29 ± 0.09
SD-LoRA	–	–
ECR-LoRA (All-scan)	$\underline{77.12 \pm 0.58}$	$\underline{82.55 \pm 1.23}$
ECR-LoRA	$\mathbf{77.52 \pm 0.27}$	$\mathbf{82.90 \pm 1.17}$

also reduces S_{corr} from 0.143 to 0.131 on low-margin samples, indicating less aggressive logit modification around uncertain shared predictions. Together with the performance gains reported in the main paper, these results suggest that the learned residuals become more task-specific without relying on increasingly large prediction perturbations. Overall, the diagnostics support the intended asymmetric organization: the shared core accumulates reusable knowledge, while the residual bank preserves distinct and controlled task-local corrections.

D.4 Robustness to Backbone Pretraining

We investigate whether ECR-LoRA depends on the supervised ImageNet-21K initialization adopted in the main experiments. Specifically, we replace the frozen backbone with ViT-B/16 models pre-trained on ImageNet-1K using DINO (Caron et al., 2021) and iBOT (Zhou et al., 2022). The backbone architecture, 10-session ImageNet-R protocol, core–residual split, LoRA rank, training objectives, and inference settings remain unchanged. Our results are averaged over three random seeds.

For reference, we include InfLoRA (Liang & Li, 2024) and SD-LoRA (Wu et al., 2025), which are also compared in the main paper. As shown in Table A6, ECR-LoRA remains effective under both self-supervised initializations. Core-prior consistently improves the corresponding All-scan model. With DINO, it increases A_{Last} and A_{Avg} by 0.50 and 0.33 points, respectively. With iBOT, the corresponding improvements are 0.40 and 0.35 points. These consistent gains indicate that the consolidated core continues to provide useful expert-level evidence even when the frozen representation is learned without supervised labels. Under DINO initialization, ECR-LoRA improves over SD-LoRA by 4.29 points in A_{Last} and 4.50 points in A_{Avg}. Compared with InfLoRA, it improves the two metrics by 5.76 and 3.80 points under DINO, and by 5.68 and 4.61 points under iBOT. Together with the consistent Core-prior gains, these results indicate that the core–residual organization remains effective across different self-supervised initializations, although the quality of the frozen representation still influences the continual learning trajectory.

E Limitations and Future Directions

Although ECR-LoRA improves knowledge reuse and task specialization, its task-agnostic inference currently evaluates all stored residual adapters, causing computation and latency to increase with the number of tasks. Future work may incorporate sparse mixture-of-experts routing, adapter selection, and consolidation mechanisms to retain only a compact set of informative residuals and activate a bounded number of adapters for each input. Such extensions could preserve task-specific capacity while keeping the inference cost approximately constant as the continual sequence grows.