RIDER: SUPPORT-AWARE ANALYTICAL EVIDENCE CONSTRUCTION FOR REHEARSAL-FREE CLASS-INCREMENTAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recently, rehearsal-free class-incremental learning (CIL) based on pre-trained models has attracted increasing attention, and low-rank adaptation further enables parameter-efficient continual learning. However, when task identity is unavailable at test time and class scores from different task branches participate in a unified decision, the model must resolve both competition across tasks and ambiguity among classes within each task. To better understand this issue, we diagnose the final prediction errors by decomposing them into cross-task errors and within-task classification errors, and find that cross-task errors dominate the final errors in three of the four evaluated settings. Motivated by this finding, we propose RIDER with a Support-Aware Analytical Evidence Construction mechanism that addresses the problem from the perspective of evidence construction. Specifically, RIDER combines calibrated prototype evidence, an all-class analytical global reference, and support-valid branch-native evidence to construct global predictions, without completing unsupported branch–class statistics. In addition, we constrain shared-update drift through importance-weighted regularization, decouple task-specific directions from their magnitudes, and progressively reduce the ranks of newly introduced directions to limit incremental parameter growth. Extensive experiments across multiple CIL benchmarks demonstrate the effectiveness of RIDER. In particular, both final accuracy and average incremental accuracy improve in all 55 strictly paired runs.

1 INTRODUCTION

Class-incremental learning (CIL) requires a model to learn new classes sequentially and recognize all seen classes without task identity at inference (Rebuffi et al., 2017). In the rehearsal-free setting, examples from earlier tasks are unavailable during subsequent learning. Low-rank adaptation of a pre-trained ViT backbone (Dosovitskiy et al., 2021) provides a parameter-efficient way to learn task-specific representations (Hu et al., 2022; Liang & Li, 2024; He et al., 2025). In branch-based learners, these representations must ultimately support a single decision over all seen classes. The challenge therefore extends from learning and retaining useful representations to determining how evidence from different branches should enter a global prediction.

Good class rankings within each branch alone cannot determine the relative competition of scores across different branches. A sample can be correctly classified within its true task yet lose the global competition to a class from another task. This distinction between within-task prediction and task identification motivates examining the two aspects separately (Kim et al., 2022; Wang et al., 2023a). Figure 1 partitions final prediction errors into cross-task errors, where the predicted class belongs to an incorrect task, and within-task errors, where task membership is correct, but class identity is not. Cross-task errors dominate in the evaluated CIFAR-100, ImageNet-R, and ImageNet-A configurations. An oracle evaluation with access to the true task provides a complementary diagnostic. For a fixed scoring rule, restricting the original scores to the true task identifies which errors disappear when competing tasks are excluded. The resulting gap measures the benefit of true task information under that rule; it does not establish the improvement achievable by calibration.

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.

VTAB illustrates why both error components matter. RIDER incurs more cross-task errors than CaRE Lou et al. (2026) (0.46% versus 0.16%), but fewer within-task errors (3.88% versus 5.64%), yielding a lower total error. Its advantage in this setting therefore cannot be attributed solely to reducing cross-task errors. Together, these observations motivate studying how global predictions use the available evidence while retaining discrimination within tasks.

Constructing such predictions also requires accounting for which class statistics are available in each feature space. Consider a learner that adds a task-specific branch and freezes it after training. Earlier branches can consequently accumulate statistics for later classes in unchanged feature spaces. However, a newly created branch cannot directly collect statistics for earlier classes whose training examples are no longer available. Previous approaches address missing class representations and cross-task comparison through prototype completion (Zhou et al., 2024b), distribution approximation (Li et al., 2026), or feature and classifier alignment (He et al., 2026). We investigate a design organized around directly observed statistics: *How can fixed task branches construct global predictions over all seen classes, using only the class statistics each branch directly acquires and without revisiting earlier training data?*

We propose **RIDER**, which combines continual branch learning with *Support-Aware Analytical Evidence Construction*. Its prediction rule combines three sources of evidence. First, it calibrates prototype scores from the branch learned for each class's task, adjusting branch offsets while preserving within-task rankings. Second, a dedicated ridge classifier in the fixed first-task branch accumulates statistics as each task arrives, providing a common reference that covers all seen classes. Third, native ridge classifiers in the individual branches contribute scores only for classes supported by their own training observations. The analytical heads operate in fixed random-feature spaces and are updated through accumulated sufficient statistics. This construction combines an all-class reference with task-adapted evidence over the observed support, without completing missing branch-class statistics. While offset calibration preserves within-task prototype rankings, incorporating additional analytical evidence can change both within-task rankings and cross-task comparisons.

RIDER learns its branches with shared and task-specific low-rank adapters He et al. (2025). We combine importance-weighted regularization of effective shared updates with reuse of historical task directions, and reduce the ranks of newly introduced directions to limit task-specific adapter growth. Completed branches remain frozen, allowing statistics collected at different stages to remain expressed in each branch's original feature space. Our reported prediction protocol uses unlabeled evaluation inputs for score normalization and is therefore transductive with respect to these statistics. Our contributions are threefold:

- We develop support-aware analytical evidence construction for global prediction from fixed task branches with different observed class coverage. The construction combines calibrated prototype scores, an all-class reference, and branch-native evidence restricted to classes with directly observed statistics, without completing missing branch–class evidence.
- We integrate importance-constrained shared adaptation with reusable task-specific directions and a decreasing rank schedule for continual branch learning. Completed branches provide fixed feature spaces for subsequent statistical accumulation.
- Through comprehensive evaluation across multiple CIL benchmarks, our method achieves new state-of-the-art performance, with both metrics improving in all 55 strictly paired fixed-branch runs.

2 Related Work

Class-incremental learning. Class-incremental learning (Hersche et al., 2022) requires a model to learn new classes sequentially while preserving previously acquired knowledge and recognizing all seen classes at test time without relying on task identity. To mitigate catastrophic forgetting, traditional CIL methods have explored several directions, including rehearsal, parameter regularization, and model expansion. Rehearsal-based methods (Belouadah & Popescu, 2019; Buzzega et al., 2020; Chaudhry et al., 2018; Hou et al., 2019; Liu et al., 2020a;b) retain representative samples from previously learned classes and jointly use them with new-class data during subsequent training to preserve recognition ability on old classes. Parameter-regularization methods (Wang et al., 2023b; Zheng et al., 2026; Kirkpatrick et al., 2017; Li & Hoiem, 2017; Liu et al., 2018) constrain changes to

important weights according to their estimated importance, while also alleviating practical concerns associated with storing real training samples. Model-expansion methods (Liu et al., 2021; Douillard et al., 2020; Mallya et al., 2018; Ebrahimi et al., 2020; Ramesh & Chaudhari, 2021), in contrast, adapt to new tasks by expanding or modifying model components. However, these traditional CIL approaches often suffer from fundamental performance limitations because they typically require training from scratch under limited data.

Parameter-efficient continual learning. Parameter-efficient continual learning (PECL) (Qiao & Mahdavi, 2024) extends parameter-efficient fine-tuning to sequential tasks, typically freezing the pre-trained backbone and optimizing a small set of additional parameters to balance new-task adaptation with knowledge retention. Existing approaches mainly employ prompt-based conditioning or lightweight adaptation modules. Prompt-based methods introduce task-related information through learnable tokens: L2P retrieves relevant prompts from a learnable pool by matching input features to prompt keys (Wang et al., 2022b). DualPrompt learns complementary general and expert prompts to encode task-invariant and task-specific knowledge, respectively (Wang et al., 2022a). CODA-Prompt combines learnable prompt components through attention-based weighting, enabling end-to-end optimization of prompt composition (Smith et al., 2023). Adapter-based methods (Gao et al., 2023; 2024) introduce lightweight feature transformations within the network: SSIAT continually updates a shared adapter and maintains old-class prototypes through semantic-shift compensation (Tan et al., 2024). For low-rank adaptation, O-LoRA and InfLoRA constrain update subspaces to reduce task interference (Wang et al., 2023b; Liang & Li, 2024). CL-LoRA combines shared and task-specific modules to support cross-task knowledge sharing and specialization (He et al., 2025). SD-LoRA decouples update directions from magnitudes, reusing frozen historical directions through learnable reweighting (Wu et al., 2025). EWC-LoRA estimates Fisher information in the effective weight-update space and preserves prior knowledge through importance-weighted regularization (Zheng et al., 2026).

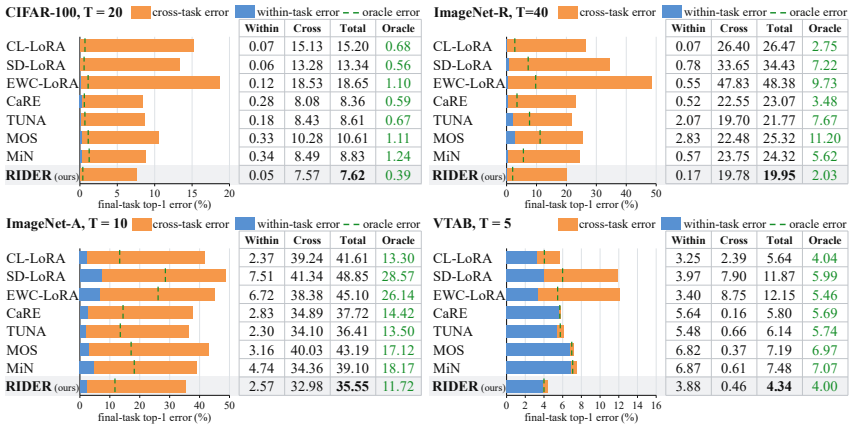
Cross-task prediction and evidence fusion. Beyond representation adaptation, CIL requires reliable decisions across tasks without task identity. Kim et al. (2022) probabilistically decompose CIL into within-task prediction and task-id prediction, relating the latter to out-of-distribution detection. TPL estimates task membership through likelihood ratios between a target task and the remaining learned tasks, using stored examples for distribution estimation (Lin et al., 2024). For all-class classifier construction, RanPAC applies nonlinear random projections to fixed features and fits a ridge classifier from accumulated feature statistics (McDonnell et al., 2023). EASE ensembles task-specific subspaces and completes old-class prototypes in newly added spaces to enable joint prediction (Zhou et al., 2024b). DUKAE trains subspace-specific classifiers through Gaussian sampling and missing-distribution approximation, then selectively ensembles class outputs from branches with directly observed distribution statistics (Li et al., 2026). CKAA combines feature- and classifier-level alignment with task-confidence-guided adapter feature aggregation to improve robustness to incorrect task identification (He et al., 2026). In contrast, RIDER is a PECL method that directly exploits sufficient statistics derived from actual observations to fit classifiers in fixed native feature spaces, without completing missing branch-class statistics or requiring cross-branch feature alignment. Our experiments further demonstrate the effectiveness of this design.

3 Preliminaries and Diagnosis

Diagonal prototype rule. At task t, rehearsal-free CIL provides only the current training set $\mathcal{D}_t$ with classes $\mathcal{C}_t$, where $\mathcal{C}_i \cap \mathcal{C}_j = \varnothing$ for $i \neq j$. Prediction covers $\mathcal{C}_{1:t} = \bigcup_{j=1}^{t} \mathcal{C}_j$ without an input task identity. Let $\tau(c)$ denote the task introducing class c, and let $f_s(x)$ be the final-layer `[CLS]` feature of branch s. The owner prototype $p_{s,c}$ averages $f_s(x)$ over training examples of $c \in \mathcal{C}_s$. The uncalibrated diagonal rule (Zhou et al., 2024b; He et al., 2025) scores each class only in its owner branch:

$$s^{\mathrm{diag}}(x, c) = \cos\big(z_{\tau(c)}(x), p_{\tau(c),c}\big), \qquad \hat{y}^{(t)}_{\mathrm{diag}}(x) = \arg\max_{c \in \mathcal{C}_{1:t}} s^{\mathrm{diag}}(x, c), \tag{1}$$

where $\mathcal{C}_{1:t} = \bigcup_{b=1}^{t} \mathcal{C}_b$, $p_{\tau(c),c}$ denotes the prototype of class c computed in the feature space of its corresponding task branch. Although within-task discrimination is performed in the corresponding task branch, the final prediction still requires scores from different branches to compete globally.



CIFAR-100, T = 20	Within	Cross	Total	Oracle
CL-LoRA	0.07	15.13	15.20	0.68
SD-LoRA	0.06	13.28	13.34	0.56
EWC-LoRA	0.12	18.53	18.65	1.10
CaRE	0.28	8.08	8.36	0.59
TUNA	0.18	8.43	8.61	0.67
MOS	0.33	10.28	10.61	1.11
MiN	0.34	8.49	8.83	1.24
RIDER (ours)	0.05	7.57	**7.62**	0.39

ImageNet-R, T=40	Within	Cross	Total	Oracle
CL-LoRA	0.07	26.40	26.47	2.75
SD-LoRA	0.78	33.65	34.43	7.22
EWC-LoRA	0.55	47.83	48.38	9.73
CaRE	0.52	22.55	23.07	3.48
TUNA	2.07	19.70	21.77	7.67
MOS	2.83	22.48	25.32	11.20
MiN	0.57	23.75	24.32	5.62
RIDER (ours)	0.17	19.78	**19.95**	2.03

ImageNet-A, T = 10	Within	Cross	Total	Oracle
CL-LoRA	2.37	39.24	41.61	13.30
SD-LoRA	7.51	41.34	48.85	28.57
EWC-LoRA	6.72	38.38	45.10	26.14
CaRE	2.83	34.89	37.72	14.42
TUNA	2.30	34.10	36.41	13.50
MOS	3.16	40.03	43.19	17.12
MiN	4.74	34.36	39.10	18.17
RIDER (ours)	2.57	32.98	**35.55**	11.72

VTAB, T = 5	Within	Cross	Total	Oracle
CL-LoRA	3.25	2.39	5.64	4.04
SD-LoRA	3.97	7.90	11.87	5.99
EWC-LoRA	3.40	8.75	12.15	5.46
CaRE	5.64	0.16	5.80	5.69
TUNA	5.48	0.66	6.14	5.74
MOS	6.82	0.37	7.19	6.97
MiN	6.87	0.61	7.48	7.07
RIDER (ours)	3.88	0.46	**4.34**	4.00

Figure 1: Final-task Top-1 errors are decomposed into the cross-task error, which counts predictions in incorrect tasks, and within-task classification error, which counts wrong-class predictions in the correct task. Lower is better. Oracle error is the remaining error when prediction is restricted to the true task with unchanged original class scores. It includes both the original within-task errors and cross-task errors that remain misclassified after this restriction. The Total-Oracle gap measures errors corrected by knowing the true task, motivating better global comparisons.

Error decomposition. Motivated by the task-ID/within-task perspective Kim et al. (2022); Wang et al. (2023a); Lin et al. (2024), we partition classification errors according to whether the predicted and true classes belong to the same task. For an evaluation sample (x, y), let $\hat{y}(x) = \hat{y}^{(t)}_{\text{diag}}(x)$. A **cross-task error** occurs when the predicted class belongs to an incorrect task, $\tau(\hat{y}(x)) \neq \tau(y)$. A **within-task classification error** occurs when the predicted and true classes belong to the same task, but $\hat{y}(x) \neq y$. So the total error admits the exact decomposition: $\mathbf{e}_{\text{cross}} = \Pr[\tau(\hat{y}(x)) \neq \tau(y)]$, $\mathbf{e}_{\text{within}} = \Pr[\tau(\hat{y}(x)) = \tau(y),\ \hat{y}(x) \neq y]$, $\mathbf{e}_{\text{total}} = \Pr[\hat{y}(x) \neq y] = \mathbf{e}_{\text{cross}} + \mathbf{e}_{\text{within}}$,.

To further separate the two effects, we restrict prediction to the ground-truth task while keeping the original scores unchanged: $\hat{y}_{\text{oracle}}(x) = \arg\max_{c \in \mathcal{C}_{\tau(y)}} s^{\text{diag}}(x, c)$. The gap between task-agnostic and oracle prediction thus measures the potential gain obtained by eliminating cross-task competition without changing the underlying representation. We therefore next examine which classes are actually supported by observations in each fixed branch.

Asymmetric branch-class support. In our protocol, each branch is frozen after training, and current data pass through all available branches before being discarded. Thus branch s directly observes exactly $\mathcal{C}_{s:t} = \bigcup_{j=s}^{t} \mathcal{C}_j$ at stage t. For class c, only branches $s \leq \tau(c)$ have observed training examples; later branches cannot revisit their earlier data. RIDER constructs evidence within this support boundary.

4 Method

4.1 Overview of RIDER

Section 3 reveals cross-task decision as a major source of errors and asymmetric branch-class observation support in rehearsal-free CIL. This motivates improving cross-task comparisons alongside preserving useful representations. We introduce **RIDER**, which couples continual branch learning with global prediction over the observed branch-class support (Figure 2). Given a ViT with N Transformer blocks, while the pre-trained backbone remains frozen. And fixed branches $f_1, \ldots, f_t$, Section 4.2 constructs three complementary scores: calibrated owner prototypes, an all-class reference in the first branch, and support-valid native classifiers in task-adapted branches. Section 4.3 describes how these branches are learned. The execution order is training, freezing the completed branch, collecting statistics from current data, and discarding those data; Section 4.4 specifies the complete procedure.

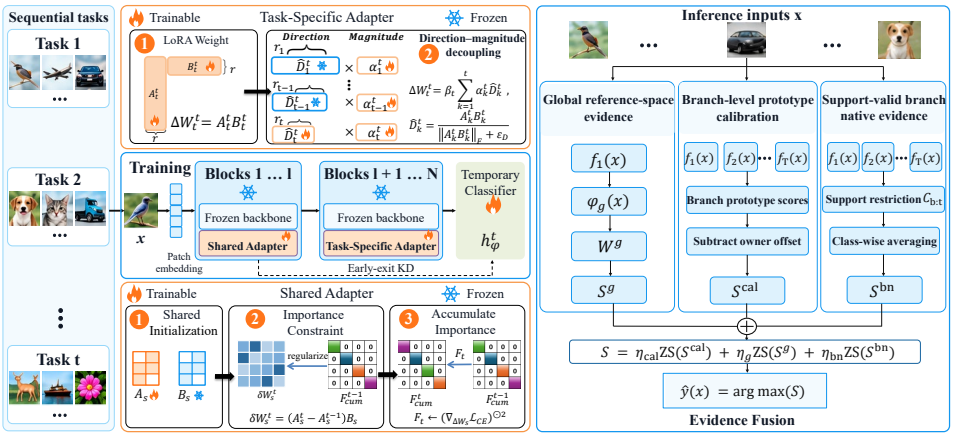


Figure 2: **Overview of RIDER.** Continual dual-adapter learning produces a fixed branch after each task. Before the current data are discarded, all available branches accumulate analytical statistics. Prediction combines calibrated prototype scores, a global reference in the first branch's feature space, and analytical scores from branches that have observed the corresponding classes.

4.2 Support-Aware Analytical Evidence Construction

Branch-level prototype calibration. The diagonal rule compares cosine scores produced in different branch spaces. Their offsets can differ even when each branch separates its own classes well. For a completed branch s, we estimate its mean maximum owner-class score on $\mathcal{D}_s$ and subtract it from all scores assigned to its classes: $\mu_b = \frac{1}{|\mathcal{D}_b|}\sum_{(x,y)\in\mathcal{D}_b} \max_{c\in\mathcal{C}_b} s^{\mathrm{diag}}(x,c)$, $S^{\mathrm{cal}}_{x,c} = s^{\mathrm{diag}}(x,c) - \mu_{\tau(c)}$. The score $s^{\mathrm{diag}}(x,c)$ is defined in Eq. 1. We compute and store μ_b before discarding $\mathcal{D}_b$. Since classes in the same task receive the same offset, their pairwise score differences are unchanged. Calibration therefore adjusts cross-task comparisons while preserving within-task rankings; a formal proof is given in Appendix C.1.

Random-feature analytical classifiers. Calibration uses only owner-branch evidence. To incorporate the additional observations in $\mathcal{C}_{s:t} = \bigcup_{j=s}^{t}\mathcal{C}_j = \{c \in \mathcal{C}_{1:t} : \tau(c) \geq s\}.$, each fixed branch maintains a ridge classifier in a random-feature space:

$$\phi_b(x) = \mathrm{ReLU}\left(R_b^\top \frac{f_b(x)}{\max(\|f_b(x)\|_2, \varepsilon_f)}\right). \tag{2}$$

Here, R_b is randomly initialized and remains fixed thereafter with f_b. For a data task $j \geq s$, let $H_{b,s}$ contain rows $\phi_b(x)^\top$, and let Y_s contain the corresponding one-hot labels in the global class order. Label matrices and accumulated class statistics are zero-padded as new classes arrive. At stage t, the sufficient statistics and classifier are:

$$\begin{aligned} G_b^{(t)} &= \sum_{s=b}^{t} H_{b,s}^\top H_{b,s}, \qquad & C_b^{(t)} &= \sum_{s=b}^{t} H_{b,s}^\top Y_s, \\ W_b^{(t)} &= \left(G_b^{(t)} + \lambda I\right)^{-1} C_b^{(t)}, \quad & S^b_{x,c} &= \phi_b(x)^\top [W_b^{(t)}]_{:,c}. \end{aligned} \tag{3}$$

The additive statistics give the same ridge solution as batch fitting on the observed features when the feature maps and regularization are fixed in the fixed branch. These statistics can be updated without retaining historical training examples; see Appendix C.2.

Global reference evidence. Among the task branches, only f_1 observes every class as it arrives. It therefore provides a reference with complete class coverage, complementing native heads with different supports (Appendix C.4). Its feature map uses an independent fixed random projection R_g: $\phi_g(x) = \mathrm{ReLU}\left(R_g^\top \frac{f_1(x)}{\max(\|f_1(x)\|_2, \varepsilon_f)}\right)$. Its sufficient statistics accumulate observations from

all data tasks $j = 1, \ldots, t$, and its ridge solution uses coefficient $\lambda_g > 0$. The resulting scores are: $S^g_{x,c} = \phi_g(x)^\top [W^g]_{:,c}, c \in \mathcal{C}_{1:t}$. This reference head is fitted independently of branch 1's native head.

Support-valid branch-native evidence. Branch b contributes only to classes in $\mathcal{C}_{b:t}$. Unsupported class columns are excluded before normalization and aggregation. At a fixed stage t, score-matrix rows index evaluation inputs, while columns retain their global class labels. We aggregate the eligible branch scores as:

$$S^{\mathrm{bn}}_{x,c} = \frac{1}{|\mathcal{B}_c|} \sum_{b \in \mathcal{B}_c} \mathrm{ZS}\big(S^b_{:,\mathcal{C}_{b:t}}\big)_{x,c}, c \in \mathcal{C}_{1:t}, \qquad \mathrm{ZS}(M) = \frac{M - \mathrm{mean}(M)}{\mathrm{std}(M) + \varepsilon_z}. \tag{4}$$

ZS computes scalar moments over all evaluation inputs and supported columns, using sample standard deviation and $\varepsilon_z = 10^{-8}$. Eligibility follows class ownership, not the input's task identity. Dividing by $|\mathcal{B}_c| = \tau(c)$ removes direct scaling by vote count; unequal supports can still affect pairwise comparisons (Appendix C.4).

Evidence fusion. Finally, RIDER places the calibrated prototype evidence, global reference evidence, and support-valid branch-native evidence on a common score scale and combines them as:

$$S = \eta_{\mathrm{cal}} \mathrm{ZS}(S^{\mathrm{cal}}) + \eta_g \mathrm{ZS}(S^g) + \eta_{\mathrm{bn}} \mathrm{ZS}(S^{\mathrm{bn}}), \qquad \hat{y}(x) \quad = \arg\max_{c \in \mathcal{C}_{1:t}} S_{x,c}. \tag{5}$$

We use the fixed weights $(\eta_{\mathrm{cal}}, \eta_g, \eta_{\mathrm{bn}}) = (0.70, 0.15, 0.15)$ across datasets; sensitivity results are provided in Appendix B.4. Branch-wise normalization in Eq. 4 precedes the second normalization of the aggregated native scores in Eq. 5. At each task boundary, we collect the new owner prototypes and calibration offset, update G_s, C_s for every available branch $s \leq t$, and update the global statistics. This completes the transition from learning a branch to maintaining its evidence and making task-agnostic predictions.

4.3 Continual Dual-Adapter Learning

Under asymmetric support, each branch supplies owner-task evidence and later serves as a fixed feature extractor for arriving classes. We combine shared and task-specific adaptation to retain reusable transformations across successive branches while learning task-discriminative features.

Importance-constrained shared adaptation. The shared update $\Delta W^{i,(t)}_{\mathrm{sh},m} = A^{i,(t)}_{\mathrm{sh},m} B^i_{\mathrm{sh},m}$ uses trainable A and fixed orthonormal-row B, with constant rank r_{sh}. To retain reusable information, we penalize drift of the effective update using accumulated importance (Kirkpatrick et al., 2017; Zheng et al., 2026):

$$\mathcal{L}^{(t)}_E = \frac{\lambda_E}{2} \sum_{i=1}^{l} \sum_{m \in \{q,v\}} \left\langle F^{i,(t-1)}_{\mathrm{cum},m}, \left(\Delta W^{i,(t)}_{\mathrm{sh},m} - \Delta W^{i,(t-1)}_{\mathrm{sh},m}\right)^{\odot 2} \right\rangle_F. \tag{6}$$

Here, $m \in \{q, v\}$ indexes the query (q) and value (v) projections in self-attention, $F^{i,(t-1)}_{\mathrm{cum},m} \geq 0$ stores accumulated importance of entries of AB, $\lambda_E \geq 0$ controls regularization, $\odot 2$ denotes element-wise squaring, and $\langle \cdot, \cdot \rangle_F$ is the Frobenius inner product. we initialize $F^{(0)}_{\mathrm{cum}} = 0$ and update $F^{(t)}_{\mathrm{cum}} = \mathrm{clip}\Big(\gamma F^{(t-1)}_{\mathrm{cum}} + F_t, F_{\max}\Big), t \geq 1$. This penalizes shared-update drift while leaving task-specific directions available for adaptation. The details are shown in Appendix A.

Direction-magnitude task-specific adaptation. For the specific adapters, retaining learned directions allows a new task to reuse previous adaptations while learning an additional direction. Therefore, we separate each direction from its contribution to the current task:

$$\widehat{D}^i_{b,m} = \frac{A^i_{b,m} B^i_{b,m}}{\|A^i_{b,m} B^i_{b,m}\|_F + \varepsilon_D}, \qquad \Delta W^{i,(t)}_{\mathrm{spec},m} = \beta^i_{t,m} \sum_{b=1}^{t} \alpha^i_{t,b} \widehat{D}^i_{b,m}, \qquad l < i \leq N. \tag{7}$$

where b identifies the task that introduced a direction. Historical factors remain frozen; A_t, B_t, coefficients $\alpha_{t,1:t}$, and scales β_t are learned. Query/value projections share α within each block

but have separate β. Only new directions follow $r_t = \max(r_{\min}, \lfloor r_0 \rho^{t-1} \rfloor)$, with $0 < \rho \leq 1$ and $r_0 \geq r_{\min} \geq 1$; historical ranks remain unchanged. Initialization is detailed in Appendix A. After training, we freeze the branch's complete shared and task-specific state. Freezing preserves feature-space consistency for subsequent sufficient-statistic accumulation (Appendix C.2), while shared adaptation continues for future branches.

4.4 Training and Inference

At task t, RIDER optimizes the current trainable adapters together with a temporary cosine classifier over the new classes $\mathcal{C}_t$. The overall objective is: $\mathcal{L}_t = \mathcal{L}_{\mathrm{CE}}^{(t)} + \lambda_{\mathrm{KD}} \mathcal{L}_{\mathrm{KD}}^{(t)} + \mathcal{L}_E^{(t)}$, where $\mathcal{L}_{\mathrm{CE}}^{(t)}$ is the classification cross-entropy, $\mathcal{L}_{\mathrm{KD}}^{(t)}$ is the early-exit distillation loss in CL-LoRA, the importance regularizer $\mathcal{L}_E^{(t)}$ is defined in Eq. 6. We omit distillation and importance regularization for task 1.

Task-end evidence updates. After optimizing task t, a classification-loss pass estimates shared-update importance without an optimizer step. We then save and freeze the complete adapter state defining branch f_t, including its shared and task-specific components. Using current data with gradients disabled, we collect owner prototypes and μ_t, update every available native head's statistics and the global reference statistics, which can be accumulated consistently and give rise to the asymmetric support. The temporary classifier is retired.

Inference. Inference inputs are processed by the fixed branches $f_1, \ldots, f_t$. These branches produce calibrated owner-prototype scores, global-reference scores, and support-valid branch-native scores, and combine them using Eq. 5. Prediction selects the highest-scoring class in $\mathcal{C}_{1:t}$ without using the input's task identity. The complete procedure is provided in Appendix A.1.

5 Experiments

5.1 Setup

Benchmarks. We evaluate on five CIL benchmarks: CIFAR-100 (Krizhevsky & Hinton, 2009), ImageNet-R (Hendrycks et al., 2021a), ImageNet-A (Hendrycks et al., 2021b), VTAB (Zhai et al., 2019), and DomainNet (Peng et al., 2019). We use two published task-split protocols without modification. Following He et al. (2025), Protocol A uses CIFAR-100 (20×5), ImageNet-R (40×5), ImageNet-A (10×20), and VTAB (5×10). Following Zheng et al. (2026), Protocol B uses CIFAR-100 (10×10), ImageNet-R (10×20), ImageNet-A (10×20), and DomainNet (5×69). The results and detailed analysis for Protocol B are reported in the appendix B.2.

Compared methods. We compare against prompt-based methods (L2P, DualPrompt, CODA-Prompt, LAE (Gao et al., 2023)), prototype- and adapter-based methods (APER (Zhou et al., 2024a), RanPAC, EASE), the LoRA-based line this work belongs to (O-LoRA, InfLoRA, BiLoRA (Zhu et al., 2025), SD-LoRA, CL-LoRA, EWC-LoRA), and the two strongest recent methods, MiN (Jiang et al., 2025) and CaRE (Lou et al., 2026). BiLoRA, however, is included only in the Protocol B experiments. Unmarked rows are quoted from the paper the splits come from; † rows we ran ourselves under exactly those splits and that metric.

Evaluation. RIDER adapts the CL-LoRA (He et al., 2025) as the baseline model, and freezes an ImageNet-21k-pretrained ViT-B/16 (Dosovitskiy et al., 2021); our runs use one RTX 3090 (24 GB). After task t, A_t measures accuracy over all seen classes. We report final accuracy $\bar{A}_T$ and $\mathrm{Avg.} = T^{-1} \sum_{t=1}^{T} A_t$ as mean $\pm$ standard deviation across seeds. By default, RIDER normalizes scores jointly over unlabeled evaluation inputs: this is transductive, without evaluation labels or input task identities. Detailed implementation procedures are provided in the appendix A.

5.2 Main Results

Results across protocols. RIDER attains the highest mean final and average accuracy among the compared methods in all eight dataset-protocol entries (Table 1; Appendix B.2). Under A, final-accuracy gains over the strongest alternative range from 1.01 to 4.11 percentage points; under B,

Table 1: Protocol A: final and average incremental accuracy (%). †: our ten-seed runs (mean ± std.); unmarked rows: results from He et al. (2025). Bold and underlining denote the best and second-best means.

	CIFAR-100		ImageNet-R		ImageNet-A		VTAB	
Method	$\bar{A}_{20}$	Avg.	$\bar{A}_{40}$	Avg.	$\bar{A}_{10}$	Avg.	$\bar{A}_{5}$	Avg.
L2P	79.51±0.67	85.50±1.23	60.62±1.12	65.82±0.71	37.62±1.89	39.81±1.36	76.41±2.26	78.96±1.62
CODA-Prompt	81.36±0.88	88.17±0.61	63.93±0.82	70.39±0.49	51.61±0.63	60.70±0.94	89.49±0.42	92.27±0.61
RanPAC	87.62±0.16	91.63±0.28	71.06±0.71	78.53±0.73	54.85±1.36	66.14±1.54	88.85±1.36	89.61±4.21
EASE	85.71±0.76	90.96±0.83	71.43±0.18	78.04±0.67	59.25±0.88	68.92±2.06	92.85±0.88	93.01±0.33
InfLoRA	80.97±0.74	88.84±0.90	64.51±1.25	73.22±1.12	47.04±0.90	56.91±1.27	87.16±1.17	88.83±0.94
CL-LoRA	85.32±0.08	91.02±0.12	74.51±0.14	81.58±0.59	60.54±0.63	<u>70.15</u>±2.23	<u>94.29</u>±0.34	<u>94.57</u>±0.60
SD-LoRA†	85.59±1.54	90.44±1.19	65.29±0.59	71.27±0.39	51.40±0.87	63.04±0.86	87.60±1.21	90.30±0.38
EWC-LoRA†	84.18±1.62	89.72±1.04	64.52±0.86	72.47±0.69	55.83±0.77	66.63±0.43	85.20±2.60	89.78±1.08
MiN†	90.88±0.63	93.85±0.46	<u>77.51</u>±1.18	<u>83.61</u>±1.08	60.21±1.19	69.84±1.56	93.16±0.39	94.16±0.34
CaRE†	<u>90.93</u>±0.36	<u>94.33</u>±0.32	76.32±0.84	82.20±0.86	<u>60.74</u>±1.27	69.38±1.92	94.24±0.11	94.34±0.17
RIDER†	**91.94**±0.21	**94.89**±0.30	**80.14**±0.32	**85.30**±0.55	**64.85**±0.72	**72.81**±1.52	**95.62**±0.25	**95.80**±0.25

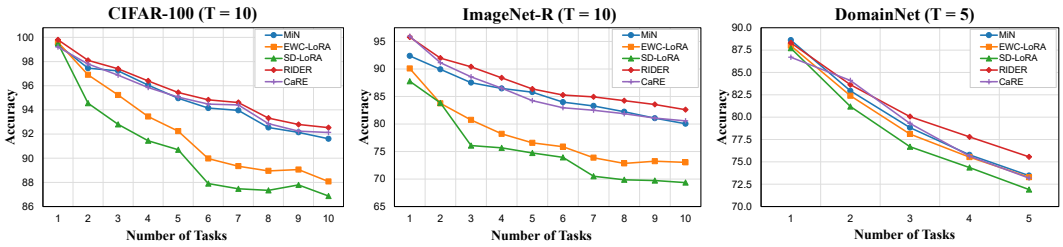


Figure 3: Task-wise top-1 accuracy (%). Results are shown under Protocol B using seed 1993.

from 0.11 to 4.16. These rankings describe the reported means, with a small margin on CIFAR-100 under B. We next separate training improvements from changes in prediction. We further evaluate RIDER's performance advantage throughout the incremental learning process. Figure 3 reports task-wise top-1 accuracy for CIFAR-100 ($T = 10$), ImageNet-R ($T = 10$), and DomainNet ($T = 5$) under Protocol B.RIDER remains competitive during the early stages and retains higher accuracy in later stages, with the clearest visual separation on ImageNet-R and DomainNet.

5.3 Ablations

We conduct ablations along the two stages of RIDER: continual branch learning and support-aware cross-task prediction. Unless stated otherwise, ablations use three seeds (1993,1994,1995) on CIFAR-100 ($T = 20$) and ImageNet-R ($T = 40$).

Continual branch learning. Table 2 isolates training components using the uncalibrated diagonal prototype rule, without analytical evidence fusion. Adding either importance-constrained shared adaptation (ICA) or direction-magnitude task-specific adaptation (DMA) improves both mean metrics; their combination yields the highest means, raising final accuracy over baseline by 0.97 and 0.94 points on the two datasets.

Evidence construction. Table 3 fixes the learned RIDER branches and changes only inference to evaluate the inference-side components; fusion weights are not retrained for individual variants. So the differences arise only from how cross-task evidence is constructed and combined. On CIFAR-100/ImageNet-R, calibration alone improves final accuracy over the prototype baseline by 0.56/0.45 points, compared with 6.14/5.63 for full fusion. Thus, offsets alone do not explain the improvement. Full fusion also exceeds ridge-only fusion by 1.80/1.97 points and outperforms each leave-one-component-out variant in both metrics. Thus, the calibrated prototypes and two ridge sources contribute complementary evidence in this configuration. Appendix B.1 reports all variants.

Table 2: Continual branch learning ablation: Final and average incremental accuracy (%) are reported for three seeds. ICA: importance-constrained shared adaptation, DMA: direction-magnitude task-specific adaptation.

Training configuration		CIFAR-100		ImageNet-R	
DMA	ICA	$\bar{A}_{20}$	Avg.	$\bar{A}_{40}$	Avg.
–	–	84.80±0.95	90.56±0.90	73.53±1.04	80.71±0.97
✓	–	85.20±0.83	91.16±0.80	73.91±0.98	81.02±0.92
–	✓	85.11±0.72	90.93±0.69	74.03±0.84	81.05±0.82
✓	✓	**85.77**±0.99	**91.45**±0.75	**74.47**±0.69	**81.74**±0.48

Table 3: Evidence ablation on identical learned branches (three seeds). Accuracy (%), mean ± std. Removing calibration retains raw prototype scores; ridge-only fusion removes prototype evidence.

	CIFAR-100		ImageNet-R	
Variant	$\bar{A}_{20}$	Avg.	$\bar{A}_{40}$	Avg.
Prototype baseline	85.77±0.99	91.45±0.75	74.47±0.69	81.74±0.48
Calibration only	86.33±0.83	92.03±1.09	74.92±0.67	82.05±0.50
Ridge-only fusion	90.11±0.52	93.51±0.62	78.13±0.37	83.60±0.44
w/o branch-native evidence	90.39±0.45	93.63±0.58	77.97±0.36	83.65±0.53
w/o global reference	90.07±0.23	93.52±0.30	78.62±0.39	84.16±0.58
w/o prototype calibration	90.51±0.29	93.65±0.37	78.70±0.55	84.18±0.70
Full RIDER	**91.91**±0.19	**95.02**±0.28	**80.10**±0.34	**85.51**±0.52

Paired effect on cross-task prediction. Across seven distinct settings, we rescore each trained RIDER model with uncalibrated prototypes and full evidence fusion, without retraining. Both metrics improve in all 55 paired runs. Mean gains across settings range from 1.75 to 6.70 points in final accuracy and 1.23 to 5.25 in Avg.; the smallest individual gains are 1.30 and 0.84 points (Appendix B.3). These comparisons isolate the complete prediction rule, whose effects can include both within-task and cross-task decisions. In addition, we conduct additional ablations, including the accuracy-parameter trade-off of rank reduction in B.5, and defer inductive normalization alternatives to the appendix B.6.

Table 4: Gains of 55 paired runs

			Final gain		Avg. gain		
Dataset	T	n	Mean	Min.	Mean	Min.	Both improved
CIFAR-100	10	5	+3.39	+2.88	+2.26	+1.75	5/5
CIFAR-100	20	10	+5.41	+4.86	+3.25	+2.57	10/10
ImageNet-R	10	5	+2.53	+2.29	+2.09	+1.71	5/5
ImageNet-R	40	10	+6.70	+5.33	+5.25	+4.30	10/10
ImageNet-A	10	10	+6.52	+5.19	+4.41	+3.82	10/10
VTAB	5	10	+1.75	+1.30	+1.23	+0.84	10/10
DomainNet	5	5	+2.03	+1.96	+1.69	+1.62	5/5

6 Conclusion

We presented RIDER, a rehearsal-free CIL framework that couples continual branch learning with support-aware analytical evidence construction. By combining calibrated prototypes, an all-class reference, and branch-native evidence over observed support, RIDER improves global prediction without completing unavailable branch–class statistics. RIDER achieves state-of-the-art performance on four public benchmarks under the reported evaluation protocols. These findings highlight the importance of expanding and comparing observed class evidence, beyond preserving task-specific representations alone.

AI USE STATEMENT

Generative AI tools were used to assist with research planning and literature search. All reported experiments were executed with the authors' code and hardware. The authors reviewed the AI-assisted work, checked the reported claims and results against the underlying artifacts, and take responsibility for the final content.

ETHICS STATEMENT

This work uses five public image-classification benchmarks (Krizhevsky & Hinton, 2009; Hendrycks et al., 2021a;b; Zhai et al., 2019; Peng et al., 2019); it collects no new data and involves no human subjects. RIDER stores adapter snapshots, prototypes, importance estimates, and aggregate Gram and class-target statistics, rather than raw examples for rehearsal. These retained states are not a formal privacy guarantee. Evaluation-set normalization uses unlabeled samples jointly and should not be presented as independent online inference. The benchmarks inherit source-data biases, and their accuracies should not be interpreted as deployed performance estimates.

REPRODUCIBILITY STATEMENT

Section 5.1 specifies datasets, published splits, metrics and implementation. Section 4 separates branch optimization, task-end statistics collection, and analytical prediction; Appendix A gives detailed training hyperparameters, Fisher settings, and pseudocode. One evidence construction configuration was frozen jointly before the eight-cell evaluation and was not tuned per dataset or split; because these protocols provide no held-out validation split, this describes configuration selection rather than validation-set model selection. evidence construction-off/on results are two scorings of the same trained networks, not comparisons of separately trained representations. Anonymous code, configurations, and aggregation scripts will be released.

REFERENCES

Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 583–592. IEEE, 2019.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018.

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.

Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *European conference on computer vision*, pp. 86–102. Springer, 2020.

Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In *European conference on computer vision*, pp. 386–402. Springer, 2020.

Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11449–11459. IEEE, 2023.

Xinyuan Gao, Songlin Dong, Yuhang He, Qiang Wang, and Yihong Gong. Beyond prompt learning: Continual adapter for efficient rehearsal-free continual learning. In *European Conference on Computer Vision*, pp. 89–106. Springer, 2024.

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

Lingfeng He, De Cheng, Zhiheng Ma, Huaijie Wang, Dingwen Zhang, Nannan Wang, and Xinbo Gao. Ckaa: Cross-subspace knowledge alignment and aggregation for robust continual learning. *IEEE Transactions on Image Processing*, 2026.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. 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.

Michael Hersche, Geethan Karunaratne, Giovanni Cherubini, Luca Benini, Abu Sebastian, and Abbas Rahimi. Constrained few-shot class-incremental learning. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9047–9057. IEEE, 2022.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 831–839. IEEE, 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.

Kai Jiang, Zhengyan Shi, Dell Zhang, Hongyuan Zhang, and Xuelong Li. Mixture of noise for pre-trained model-based class-incremental learning. In *Advances in Neural Information Processing Systems*, 2025.

Gyuhak Kim, Changnan Xiao, Tatsuya Konishi, Zixuan Ke, and Bing Liu. A theoretical study on solving continual learning. *Advances in neural information processing systems*, 35:5065–5079, 2022.

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.

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

Songze Li, Tonghua Su, Xu-Yao Zhang, Qixing Xu, and Zhongjie Wang. Dukae: Dual-level knowledge accumulation and ensemble for pre-trained model-based continual learning. *Pattern Recognition*, 179:113860, 2026.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

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.

Haowei Lin, Yijia Shao, Weinan Qian, Ningxin Pan, Yiduo Guo, and Bing Liu. Class incremental learning via likelihood ratio based task prediction. In *International Conference on Learning Representations*, volume 2024, pp. 50989–51028, 2024.

Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. *arXiv preprint arXiv:1802.02950*, 2018.

Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D Bagdanov, Shangling Jui, and Joost Van de Weijer. Generative feature replay for class-incremental learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 915–924. IEEE, 2020a.

Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 12242–12251. IEEE, 2020b.

Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2544–2553. IEEE, 2021.

Meng Lou, Yunxiang Fu, and Yizhou Yu. Scaling continual learning to 300+ tasks with bi-level cross-task prediction mixture-of-experts. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *PMLR*, 2026.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision*, pp. 72–88. Springer, 2018.

Mark D. McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. RanPAC: Random projections and pre-trained models for continual learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

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.

Fuli Qiao and Mehrdad Mahdavi. Learn more, but bother less: parameter efficient continual learning. *Advances in Neural Information Processing Systems*, 37:97476–97498, 2024.

Rahul Ramesh and Pratik Chaudhari. Model zoo: A growing" brain" that learns continually. *arXiv preprint arXiv:2106.03027*, 2021.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

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.

Yuwen Tan, Qinhao Zhou, Xiang Xiang, Ke Wang, Yuchuan Wu, and Yongbin Li. Semantically-shifted incremental adapter-tuning is a continual vitransformer. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23252–23262. IEEE, 2024.

Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. *Advances in Neural Information Processing Systems*, 36:69054–69076, 2023a.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. *arXiv preprint*, 2023b.

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, 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.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

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, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, pp. 1–21, 2024a.

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, 2024b.

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

A Implementation Details

Optimisation. The backbone is a frozen ViT-B/16 pre-trained on ImageNet-21k, while we place shared LoRA adapters in blocks $1, \ldots, l$ and task-specific adapters in blocks $l+1, \ldots, N$ (in this work, we set $l = 6$). Both components adapt the query and value projections, indexed by $m \in \{q, v\}$. We train with SGD with momentum 0.9, an initial learning rate of 0.05, and cosine annealing over the epochs of each task. The random projection matrices are independently sampled from a Gaussian distribution with a fixed random seed,

$$R_{ij} \sim \mathcal{N}\left(0, \frac{1}{768}\right),$$

corresponding to a standard deviation of $1/\sqrt{768}$, and remain fixed after initialization. During training, images are processed using a 224×224 random resized crop with an area scale of $[0.05, 1.0]$, an aspect-ratio range of $[3/4, 4/3]$, and bilinear interpolation, followed by horizontal flipping with probability 0.5. For testing and analytical-statistics fitting, images are resized so that the shorter side is 256 using bicubic interpolation and then center-cropped to 224×224. Pixel values are scaled to $[0, 1]$, with no additional channel-wise normalization. Ridge sufficient statistics are accumulated in FP32, while the corresponding linear systems are solved on the CPU in FP64 using torch.linalg.solve. Table 5 lists the dataset-specific settings. The task-specific component starts from rank r_0, and the rank allocated to each newly introduced direction progressively decreases toward $r_{\min}$. This results in a mean of 0.158M, 0.224M, 0.232M, and 0.240M gradient-updated parameters per task on CIFAR-100, ImageNet-R, ImageNet-A, and VTAB, respectively, corresponding to approximately 0.18%–0.28% of the backbone parameters. ten seeds:1993, 1994, 1995, 0, 1, 2, 3, 4, 5, 6, and five seeds:1993, 1994, 1995, 0, 1.

Adapter initialization and amplitude inheritance. For the shared adapters, we initialize $A_s^{(0)} = 0$ and initialize B_s with orthonormal rows. The shared B_s factors remain frozen throughout the task sequence, while A_s is continuously updated. For a new task t, the rank

$$r_t = \max\big(r_{\min}, \lfloor r_0 \rho^{t-1} \rfloor\big) \tag{8}$$

is used only to allocate the newly introduced task-specific factors. At each task-specific query/value site, we initialize

$$A_t \sim \mathcal{N}(0, \sigma_A^2), \qquad B_t B_t^\top = I_{r_t}, \tag{9}$$

where (σ_A) denotes denotes the standard deviation used for task-specific initialization. We set it to 0.005 on CIFAR-100 and 0.03 on VTAB, and use 0.001 consistently on ImageNet-R, ImageNet-A, and DomainNet. the second relation denotes orthonormal-row initialization. Historical factors $\{A_b, B_b\}_{b<t}$ are frozen once their corresponding tasks have been completed and retain their original ranks.

For task $t > 1$, the amplitudes associated with historical directions are initialized from those learned by the previous task,

$$\alpha_{t,b}^i \leftarrow \alpha_{t-1,b}^i, \qquad b < t, \tag{10}$$

while the newly introduced direction is initialized as

$$\alpha_{t,t}^i = \alpha_{\text{init}}. \tag{11}$$

All entries of the current amplitude vector remain trainable during task t, and query and value projections share the amplitude vector within each task-specific block.

Following the block-wise mechanism inherited from CL-LoRA, we additionally initialize $U_t \in \mathbb{R}^{3\times(N-l)}$ from $\mathcal{U}(0.5, 1.5)$ and normalize it as

$$\widetilde{U}_t = \frac{U_t}{\text{mean}\,|U_t| + \varepsilon_U}. \tag{12}$$

The corresponding query/value scale is

$$\beta_{t,m}^i = (\widetilde{U}_t)_{\iota(m),\, i-l}, \qquad \iota(q) = 1, \quad \iota(v) = 3. \tag{13}$$

Only the query and value rows directly scale adapter outputs.

Early-exit distillation and gradient reassignment. For task $t > 1$, RIDER retains the early-exit distillation mechanism of CL-LoRA. Let $z_t^l(x)$ denote the representation produced at block l by the current shared adapter, and let $\widetilde{z}_{t-1}^l(x)$ denote the representation obtained using the frozen shared adapter preserved after task $t-1$. Both representations are mapped by the same current-task local classifier h_ϕ^t. With temperature ϑ, we compute

$$p_{t,c}(x) = \text{softmax}\left(\frac{h_\phi^t(z_t^l(x))}{\vartheta}\right)_c, \qquad \widetilde{p}_{t-1,c}(x) = \text{softmax}\left(\frac{h_\phi^t(\widetilde{z}_{t-1}^l(x))}{\vartheta}\right)_c. \tag{14}$$

For a current minibatch $\mathcal{M}$, the distillation loss is

$$\mathcal{L}_{\text{KD}}^{(t)} = -\frac{1}{|\mathcal{M}|} \sum_{x\in\mathcal{M}} \sum_{c\in\mathcal{C}_t} \widetilde{p}_{t-1,c}(x) \log p_{t,c}(x). \tag{15}$$

The previous shared adapter is frozen, while the same current local head is used for both paths. In our implementation, the teacher probabilities are not explicitly detached.

The current-task objective is therefore

$$\mathcal{L}_t = \mathcal{L}_{\text{CE}}^{(t)} + \lambda_{\text{KD}} \mathcal{L}_{\text{KD}}^{(t)} + \mathcal{L}_E^{(t)}. \tag{16}$$

A single backward pass is performed for this objective. Before the optimizer step, CL-LoRA's gradient-reassignment rule rescales each row of the shared A_s gradient using

$$\omega_j = \frac{d\|a_{s,j}^{(t-1)}\|_2}{\sum_{k=1}^d \|a_{s,k}^{(t-1)}\|_2}, \tag{17}$$

and $g_{A_s[j,:]} \leftarrow \omega_j g_{A_s[j,:]}$. Both distillation and shared importance regularization are omitted for the first task.

Fisher estimation and importance accumulation. After completing the gradient-based optimization of task t, we estimate the importance of the shared effective update using a separate CE-only pass with no optimizer update. For each shared query/value adapter site, let $\mathcal{M}_j$ denote the j-th Fisher minibatch and let $\overline{\mathcal{L}}_{\mathrm{CE}}(\mathcal{M}_j)$ denote its minibatch-mean cross-entropy loss. The diagonal Fisher-style importance is estimated as

$$F_t = \frac{1}{J_t} \sum_{j=1}^{J_t} \left(\nabla_{\Delta W_s} \overline{\mathcal{L}}_{\mathrm{CE}}(\mathcal{M}_j) \right)^{\odot 2}, \tag{18}$$

where J_t is the number of minibatches used for estimation. The gradients are taken with respect to the effective shared update $\Delta W_s = A_s B_s$, rather than to A_s and B_s independently.

We initialize $F_{\mathrm{cum}}^{(0)} = 0$ and, suppressing block and projection indices, accumulate $F_{\mathrm{cum}}^{(t)} = \gamma F_{\mathrm{cum}}^{(t-1)} + F_t$. When clipping is enabled, we subsequently apply $F_{\mathrm{cum}}^{(t)} \leftarrow \min(F_{\mathrm{cum}}^{(t)}, F_{\max})$ element-wise. Thus γ discounts historical importance, while $F_{\max}$ also limits the current estimate's contribution; their settings are reported separately in Table 5.

In particular, $F_{\max} = 2.0$ is used as a numerical safeguard against a single task producing unusually large Fisher values. This effect cannot be controlled by γ alone because γ discounts only the previous accumulation before the current estimate F_t is added. We therefore do not treat $F_{\max}$ as interchangeable with γ or assume that clipping is inactive when enabled.

Table 5: Per-dataset hyper-parameters. Srank denotes Shared rank. FS is Fisher samples and FB is Fisher Batch. γ, $F_{\max}$, and λ_E denote the EWC accumulation decay, clipping threshold, and regularization strength, respectively. "–" indicates that clipping is disabled. TRank denotes the task-specific. $r_0 \rightarrow r_{\min}$ schedule.

Dataset	Epochs	Batch	FS	FB	Srank	λ_E	γ	$F_{\max}$	TRank	α_{init}	ρ
CIFAR-100	30	64	1024	64	8	180	1.0	–	$8 \rightarrow 4$	0.10	0.9
ImageNet-R	25	32	1024	32	10	130	0.9	2.0	$10 \rightarrow 7$	0.13	0.9
ImageNet-A	25	32	1024	32	10	130	0.9	2.0	$10 \rightarrow 7$	0.13	0.9
VTAB	45	32	1024	32	10	200	1.0	–	$10 \rightarrow 6$	0.80	0.9
DomainNet	15	128	1024	32	10	500	0.9	2.0	$30 \rightarrow 15$	0.50	0.9

Task-end updates. After Fisher estimation, the completed adapter state is preserved as the new fixed branch f_t. All saved branches are then switched to evaluation mode and the remaining task-end operations are performed without gradients.

The current-task training data $\mathcal{D}_t$ are passed through every available fixed branch f_b, $b \leq t$. For the newly introduced classes, the owner branch f_t is used to construct their class prototypes and the branch-level calibration statistic μ_t^+. At the same time, each branch that has directly observed the current classes updates its random-feature ridge sufficient statistics. Existing class-sum matrices are zero-padded when the output dimension expands to include new classes. Because branch 1 has been available since the first task, its dedicated global reference head is also updated using every task as it arrives.

After sufficient-statistic accumulation, the corresponding ridge systems are solved to update the analytical classifier weights. These task-end operations use only the current-task data: previous-task images and per-example features are neither revisited nor retained. Temporary image, feature, label, and random-feature buffers are discarded before advancing to the next task, and the temporary current-task cosine classifier is retired.

Analytical heads and cross-task prediction configuration. A single RIDER inference configuration is used across datasets and task counts. The all-class reference head uses a random-feature dimension of $D_g = 16384$ on the fixed first-task branch, while each branch-native head uses $D_\ell = 4096$. The corresponding ridge coefficients are $\lambda_g = \lambda_\ell = 0.1$. All random projections are sampled once and then fixed, and their sufficient statistics are accumulated in a streaming fashion. Thus, the analytical heads require neither gradient-based fitting nor stored exemplars.

The manuscript indexes tasks and branches from 1; the first-task reference branch therefore corresponds to branch 1 in the notation used in the paper, although it may appear as branch 0 under zero-based code indexing. The fusion weights are fixed to $(\eta_{\text{cal}}, \eta_g, \eta_{\text{bn}}) = (0.70, 0.15, 0.15)$ for all reported experiments. Score standardization in the main setting uses the complete unlabeled evaluation-score matrix, yielding the transductive inference rule described in Section 4.2.

The remaining training constants are $\vartheta = 2$ and $\lambda_{\text{KD}} = 5$.

A.1 Execution Pseudocode

Algorithms 1–3 provide the complete execution procedures of RIDER. Algorithm 1 details current-task dual-adapter optimization, including the initialization of new task-specific directions, early-exit distillation, importance regularization, and shared-gradient reassignment. At the end of each task, it invokes the TaskEndUpdate procedure in Algorithm 2, which estimates and accumulates shared-update importance, freezes the learned branch, constructs class prototypes and calibration statistics, and updates analytical heads from streaming sufficient statistics. Algorithm 3 then performs task-agnostic prediction by combining calibrated prototype evidence, global reference-space evidence, and support-valid branch-native evidence.

Tasks are indexed from 1, and the notation follows Section 4. Training and task-end statistics collection use only current-task data; previous-task images and per-example features are not retained. Once a branch is saved, its feature mapping remains fixed for subsequent statistics collection and inference. At inference, each branch contributes analytical evidence only for classes observed since its creation. Score standardization is performed over the complete unlabeled evaluation set rather than independently for each input or minibatch, following the transductive rule described in the main text.

B Additional Ablation and Analysis

This appendix provides additional experiments and implementation-level analyses that complement the ablations reported in the main text. We first expand the inference-side component study to include alternative combinations of the three evidence sources, and then report the complete paired comparison between prototype-only prediction and the full evidence construction across all benchmark–task settings. We further examine sensitivity to the fusion weights, the accuracy–parameter trade-off of progressive rank reduction, evaluation-set-independent normalization alternatives, and task-wise incremental trajectories.

Table 6: **CIFAR-100, ImageNet-R, ImageNet-A and DomainNet**, five seeds . Results are mean ± std.; higher is better for both metrics. Unmarked rows are quoted from Zheng et al. (2026). ImageNet-A is the same split as in Table 1, and the † rows here restrict those runs to the five seeds above.

	CIFAR-100		ImageNet-R		ImageNet-A		DomainNet	
Method	$\bar{A}_{10}$	Avg.	$\bar{A}_{10}$	Avg.	$\bar{A}_{10}$	Avg.	$\bar{A}_5$	Avg.
Finetune	79.09±1.53	88.17±0.45	60.42±1.64	73.18±0.32	32.85±1.53	54.55±1.44	65.57±0.20	75.12±0.12
L2P	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	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	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
InfLoRA	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	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	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	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
Vanilla LoRA	82.99±0.84	89.74±0.58	64.87±0.73	75.57±0.23	40.01±1.32	58.28±0.85	69.79±0.11	77.44±0.08
EWC-LoRA	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
MiN†	91.92±0.19	95.00±0.16	80.13±0.10	85.32±0.63	60.65±0.72	69.90±1.35	73.40±0.06	79.88±0.05
CaRE†	92.31±0.59	94.95±0.31	80.40±0.17	85.56±0.49	60.47±0.73	69.15±0.99	73.32±0.12	79.75±0.08
RIDER†	**92.42**±0.15	**95.31**±0.14	**82.55**±0.19	**87.00**±0.35	**64.81**±1.00	**72.92**±1.70	**75.55**±0.02	**81.09**±0.01

Algorithm 1 Class-Incremental Training with RIDER

Require: Task datasets $\{\mathcal{D}_t\}_{t=1}^{T}$; frozen backbone f_θ; shared/specific block split l; shared rank r_s
Require: Rank settings $(r_0, r_{\min}, \rho)$; task epochs E_t; optimizer and loss settings
Ensure: Fixed branches $\{f_b\}_{b=1}^{T}$ and analytical inference state $\mathcal{R}_T$

Initialize $A_s^{(0)} \leftarrow 0$ and $B_s \leftarrow \textsc{OrthogonalRows}(r_s, d)$; freeze θ and B_s
$F_{\text{cum}}^{(0)} \leftarrow 0$; $\mathcal{R}_0 \leftarrow \varnothing$
for $t = 1, \dots, T$ **do**
  $r_t \leftarrow \max\big(r_{\min}, \lfloor r_0 \rho^{t-1} \rfloor\big)$
  $A_s^{(t)} \leftarrow \textsc{Copy}(A_s^{(t-1)})$
  Initialize new task-specific factors (A_t, B_t) with rank r_t
  Initialize current amplitudes $\boldsymbol{\alpha}_t$, block weights U_t, and temporary local head h_ϕ^t
  $\Theta_t \leftarrow \{A_s^{(t)}, A_t, B_t, \boldsymbol{\alpha}_t, U_t, \phi_t\}$
  Freeze historical task-specific factors $\{A_k, B_k\}_{k<t}$ and all previously saved branches
  Set the current training path and h_ϕ^t to training mode
  for $e = 1, \dots, E_t$ **do**
    for minibatch $\mathcal{M} \subset \mathcal{D}_t$ **do**
      $\Delta W_s \leftarrow A_s^{(t)} B_s$
      $\widehat{D}_k \leftarrow A_k B_k / (\|A_k B_k\|_F + \varepsilon_D)$, for $k = 1, \dots, t$
      $\Delta W_{\text{spec}} \leftarrow \beta_t \sum_{k=1}^{t} \alpha_{t,k} \widehat{D}_k$
      $(z_t^N, z_t^l) \leftarrow \textsc{DualAdapterForward}(\mathcal{M}; \Delta W_s, \Delta W_{\text{spec}})$
      $\mathcal{L}_{\text{CE}} \leftarrow \textsc{MeanCE}(h_\phi^t(z_t^N[\text{CLS}]), y_{\text{local}})$
      $\mathcal{L}_{\text{KD}} \leftarrow 0$; $\mathcal{L}_E \leftarrow 0$
      if $t > 1$ **then**
        $\widetilde{z}_{t-1}^l \leftarrow \textsc{SharedForward}(\mathcal{M}; A_s^{(t-1)}, B_s)$
        $p_t \leftarrow \text{softmax}\big(h_\phi^t(z_t^l[\text{CLS}])/\vartheta\big)$
        $\widetilde{p}_{t-1} \leftarrow \text{softmax}\big(h_\phi^t(\widetilde{z}_{t-1}^l[\text{CLS}])/\vartheta\big)$
        $\mathcal{L}_{\text{KD}} \leftarrow -|\mathcal{M}|^{-1} \sum_{x \in \mathcal{M}} \sum_{c \in \mathcal{C}_t} \widetilde{p}_{t-1,c}(x) \log p_{t,c}(x)$
        $\delta W_s \leftarrow (A_s^{(t)} - A_s^{(t-1)}) B_s$
        $\mathcal{L}_E \leftarrow \frac{\lambda_E}{2} \sum_{\text{shared sites}} \left\langle F_{\text{cum}}^{(t-1)}, (\delta W_s)^{\odot 2} \right\rangle_F$
      end if
      $\mathcal{L}_t \leftarrow \mathcal{L}_{\text{CE}} + \lambda_{\text{KD}} \mathcal{L}_{\text{KD}} + \mathcal{L}_E$
      Clear gradients; $\{g_\xi\}_{\xi \in \Theta_t} \leftarrow \nabla_{\Theta_t} \mathcal{L}_t$ ▷ One backward pass
      if $t > 1$ **then**
        $g_{A_s[j,:]} \leftarrow \omega_j g_{A_s[j,:]}$ for every shared A_s and row j
      end if
      Update Θ_t with one optimizer step
    end for
  end for
  $(F_{\text{cum}}^{(t)}, f_t, \mathcal{R}_t) \leftarrow \textsc{TaskEndUpdate}(t, \mathcal{D}_t, \Theta_t, F_{\text{cum}}^{(t-1)}, \mathcal{R}_{t-1})$
  Preserve $A_s^{(t)}$ as the frozen shared reference for task $t+1$
  Retire the temporary classifier h_ϕ^t; retain no raw images or per-example features from $\mathcal{D}_t$
end for
return $\{f_b\}_{b=1}^{T}, \mathcal{R}_T$

DUALADAPTERFORWARD adds shared LoRA updates to blocks $1{:}l$ and task-specific updates to blocks $(l+1){:}N$ of the frozen backbone, and returns the final and early-exit representations. The same current-task classifier h_ϕ^t is used on both distillation paths. The previous shared parameters are frozen. Initialization, amplitude inheritance, β_t, and the shared-gradient multiplier ω_j are detailed in Appendix A.

B.1 Extended Evidence-Component Ablation

B.2 Protocol B: Results and Discussion

Under Protocol B, as shown in Table 6, RIDER improves on the second-best method by $+2.15/+1.44$ over CaRE on ImageNet-R. On ImageNet-A, it gains $+4.16$ in $\bar{A}_T$ over MiN and $+2.16$ in Avg. over CL-LoRA; on DomainNet, it gains $+2.09$ in $\bar{A}_T$ over EWC-LoRA and $+1.21$ in Avg. over MiN. On CIFAR-100, RIDER also ranks first on both metrics, exceeding CaRE by $+0.11$ in $\bar{A}_T$ and MiN by $+0.31$ in Avg., although these margins are comparatively small.

Algorithm 2 TASKENDUPDATE: Importance Estimation and Analytical Evidence Construction

Require: Stage t; current-task data $\mathcal{D}_t$; trained current path and local head; previous fixed branches
Require: $F_{\text{cum}}^{(t-1)}, \mathcal{R}_{t-1}; \gamma, F_{\max}, J_t; D_g, D_\ell, \lambda_g, \lambda_\ell$
Ensure: Updated importance $F_{\text{cum}}^{(t)}$, new fixed branch f_t, and analytical inference state $\mathcal{R}_t$

Load retained prototypes, calibration offsets, random projections, and sufficient statistics from $\mathcal{R}_{t-1}$
$F_t \leftarrow 0$ at every shared query/value adapter site; $F_{\text{cum}}^{(t)} = 0$
for $j = 1, \ldots, J_t$ **do**
  $\mathcal{M}_j \leftarrow$ Fisher minibatch sampled from $\mathcal{D}_t$
  $g_j \leftarrow \nabla_{\Delta W_s} \overline{\mathcal{L}}_{\text{CE}}(\mathcal{M}_j; \Theta_t)$ ▷ CE-only; no optimizer step
  $F_t \leftarrow F_t + g_j^{\odot 2}/J_t$
end for
$F_{\text{cum}}^{(t)} \leftarrow \gamma F_{\text{cum}}^{(t-1)} + F_t$
if importance clipping is enabled **then**
  $F_{\text{cum}}^{(t)} \leftarrow \min\left(F_{\text{cum}}^{(t)}, F_{\max}\right)$ ▷ Element-wise; also applied when $t = 1$
end if
$f_t \leftarrow$ FROZENSNAPSHOT$(A_s^{(t)}, B_s, \{A_k, B_k\}_{k \le t}, \boldsymbol{\alpha}_t, U_t; f_\theta)$
Switch all saved branches to evaluation mode and disable gradients for the remaining task-end operations
$K_t \leftarrow |\mathcal{C}_{1:t}|$; $Y_t \leftarrow$ global K_t-class one-hot labels for $\mathcal{D}_t$
if $t = 1$ **then**
  Sample fixed global projection $R_g \in \mathbb{R}^{d \times D_g}$
  $G^g \leftarrow 0$; $C^g \leftarrow 0_{D_g \times K_t}$
end if
Sample a distinct fixed branch-native projection $R_t \in \mathbb{R}^{d \times D_\ell}$
$G^t \leftarrow 0$; $C^t \leftarrow 0_{D_\ell \times K_t}$
Zero-pad the existing class-sum statistics C^g and $\{C^b\}_{b<t}$ to K_t columns
for $b = 1, \ldots, t$ **do**
  $Z_{b,t} \leftarrow [\, f_b(x)^\top \,]_{x \in \mathcal{D}_t}$ using evaluation preprocessing
  if $b = t$ **then**
    $p_c^t \leftarrow |\mathcal{D}_{t,c}|^{-1} \sum_{(x,y) \in \mathcal{D}_{t,c}} f_t(x)$ for every $c \in \mathcal{C}_t$
    $\mu_t^+ \leftarrow |\mathcal{D}_t|^{-1} \sum_{(x,y) \in \mathcal{D}_t} \max_{c \in \mathcal{C}_t} \cos(f_t(x), p_c^t)$
  end if
  $H_{b,t} \leftarrow \mathrm{ReLU}(\mathrm{RowNorm}(Z_{b,t}) R_b)$
  $G^b \leftarrow G^b + H_{b,t}^\top H_{b,t}$
  $C^b \leftarrow C^b + H_{b,t}^\top Y_t$
  $W^b \leftarrow$ SOLVE$(G^b + \lambda_\ell I_{D_\ell}, C^b)$
  if $b = 1$ **then**
    $H_{g,t} \leftarrow \mathrm{ReLU}(\mathrm{RowNorm}(Z_{1,t}) R_g)$
    $G^g \leftarrow G^g + H_{g,t}^\top H_{g,t}$
    $C^g \leftarrow C^g + H_{g,t}^\top Y_t$
    $W^g \leftarrow$ SOLVE$(G^g + \lambda_g I_{D_g}, C^g)$
  end if
end for
Store

$$\mathcal{R}_t = \{p_c^{\tau(c)}, \mu_b^+, R_g, G^g, C^g, W^g, R_b, G^b, C^b, W^b\}_{b \le t,\ c \in \mathcal{C}_{1:t}}$$

Discard temporary images, labels, features, and random-feature buffers
return $F_{\text{cum}}^{(t)}, f_t, \mathcal{R}_t$

$\mathcal{D}_{t,c} = \{(x, y) \in \mathcal{D}_t : y = c\}$, and $\tau(c)$ denotes the task in which class c was introduced. $\mathrm{RowNorm}$ divides every feature row by $\max(\|\cdot\|_2, \varepsilon_f)$. SOLVE$(A, C)$ solves the linear system $AW = C$ rather than explicitly forming A^{-1}. The sufficient statistics above are accumulated minibatch by minibatch using only current-task data, and no per-example buffer survives the task boundary. The global reference head and the branch-native head of branch 1 use the same fixed branch representation f_1 but distinct random projections and independently fitted ridge solutions.

The main text reports the core leave-one-component-out ablation of RIDER's inference stage. Here we provide the complete comparison among prototype calibration, the global reference ridge head, and the support-valid branch-native ridge ensemble.

All variants in this subsection are evaluated on exactly the same learned RIDER branches. Therefore, the differences below are attributable to the inference-side construction and combination of evidence

Algorithm 3 Task-Agnostic Inference with Support-Aware Analytical Evidence

Require: Unlabeled evaluation set $\mathcal{X}_{\text{eval}}$; stage t; fixed branches $\{f_b\}_{b\le t}$; analytical state $\mathcal{R}_t$
Require: Fusion weights $(\eta_{\text{cal}}, \eta_g, \eta_{\text{bn}}) = (0.70, 0.15, 0.15)$; $\varepsilon_z = 10^{-8}$
Ensure: Predictions $\{\hat{y}(x) : x \in \mathcal{X}_{\text{eval}}\}$ over $\mathcal{C}_{1:t}$

Set all fixed branches to evaluation mode and disable gradients
for $b = 1, \dots, t$ **do**
  $Z_b \leftarrow [\, f_b(x)^\top \,]_{x\in\mathcal{X}_{\text{eval}}}$ using evaluation preprocessing
  for $c \in \mathcal{C}_b$ **do**
    $S^{\text{cal}}_{x,c} \leftarrow \cos(f_b(x), p^b_c) - \mu^+_b$ for all $x \in \mathcal{X}_{\text{eval}}$
  end for
  $H_b \leftarrow \text{ReLU}(\text{RowNorm}(Z_b)R_b)$
  $S^b \leftarrow H_b W^b$
  $\mathcal{V}^{(t)}_b \leftarrow \{c \in \mathcal{C}_{1:t} : \tau(c) \ge b\}$ ▷ Observed support of branch b
  $Q^b \leftarrow \text{ZS}\left(S^b_{:,\mathcal{V}^{(t)}_b}\right)$ ▷ Normalize valid entries only
  if $b = 1$ **then**
    $H_g \leftarrow \text{ReLU}(\text{RowNorm}(Z_1)R_g)$
    $S^g \leftarrow H_g W^g$
  end if
end for
for $c \in \mathcal{C}_{1:t}$ **do**
  $\mathcal{B}_c \leftarrow \{1, \dots, \tau(c)\}$
  $S^{\text{bn}}_{:,c} \leftarrow |\mathcal{B}_c|^{-1} \sum_{b\in\mathcal{B}_c} Q^b_{:,c}$
end for
$S \leftarrow \eta_{\text{cal}}\,\text{ZS}(S^{\text{cal}}) + \eta_g\,\text{ZS}(S^g) + \eta_{\text{bn}}\,\text{ZS}(S^{\text{bn}})$
$\hat{y}(x) \leftarrow \arg\max_{c\in\mathcal{C}_{1:t}} S_{x,c}$ for every $x \in \mathcal{X}_{\text{eval}}$
return $\{\hat{y}(x) : x \in \mathcal{X}_{\text{eval}}\}$

$\text{ZS}(M) = (M - \text{mean}(M))/(\text{std}(M) + \varepsilon_z)$. For each branch-native score matrix, the moments are computed over the full evaluation set and only the valid columns $\mathcal{V}^{(t)}_b$. After branch-native aggregation, source-level standardization uses the full evaluation set and all seen-class columns of each complete evidence source. Feature extraction may be performed in minibatches, but the standardization moments are not reset per minibatch. Thus, the main rule above is evaluation-set-standardized and transductive; evaluation-set-independent alternatives are described in Appendix B.6.

rather than to changes in branch training. We use seeds 1993, 1994, and 1995 on CIFAR-100 ($T = 20$) and ImageNet-R ($T = 40$), and report mean final and average incremental accuracy together with the corresponding standard deviation.

Unless explicitly stated otherwise, the fusion weights are not retuned for individual variants; removing a source simply removes its corresponding term from the fusion rule.

For clarity, the variants in Table 7 are:

- **R0: Prototype baseline.** The uncalibrated diagonal prototype rule in Eq. 1, without analytical ridge evidence.
- **R1: Calibration only.** The diagonal prototype scores are shifted by their branch-level calibration offsets, but no ridge evidence is added.
- **R2: w/o branch-native evidence.** The calibrated prototype evidence and global reference ridge head are retained, while the branch-native ridge ensemble is removed.
- **R3: w/o global reference.** The calibrated prototype evidence and branch-native ridge ensemble are retained, while the all-class global reference head is removed.
- **R4: Ridge-only fusion.** Only the global and branch-native ridge sources are fused; prototype evidence is excluded.
- **R5: Global ridge only.** Prediction uses only the all-class ridge classifier constructed in the fixed first-task reference space.
- **R6: Branch-native ridge only.** Prediction uses only the support-valid ensemble of native-space ridge heads.

Table 7: **Extended evidence-component ablation.** Final and average incremental accuracy (%) are reported as mean ± standard deviation over seeds 1993, 1994, and 1995. All variants use exactly the same learned RIDER branches.

Variant		CIFAR-100		ImageNet-R	
		$\bar{A}_{20}$	Avg.	$\bar{A}_{40}$	Avg.
R0	Prototype baseline	85.77±0.99	91.45±0.75	74.47±0.69	81.74±0.48
R1	Calibration only	86.33±0.83	92.03±1.09	74.92±0.67	82.05±0.50
R2	w/o branch-native evidence	90.39±0.45	93.63±0.58	77.97±0.36	83.65±0.53
R3	w/o global reference	90.07±0.23	93.52±0.30	78.62±0.39	84.16±0.58
R4	Ridge-only fusion	90.11±0.52	93.51±0.62	78.13±0.37	83.60±0.44
R5	Global ridge only	89.89±0.68	93.11±0.77	74.33±0.45	82.57±0.62
R6	Branch-native ridge only	89.97±0.74	92.95±0.83	78.03±0.75	83.13±0.82
R7	w/o calibration	90.51±0.29	93.65±0.37	78.70±0.55	84.18±0.70
R8	Full evidence construction	**91.91**±0.19	**95.02**±0.28	**80.10**±0.34	**85.51**±0.52

- **R7: w/o calibration.** The two analytical ridge sources are combined with the original uncalibrated prototype scores.
- **R8: Full evidence construction.** The complete RIDER inference rule combines calibrated prototype evidence, global reference evidence, and support-valid branch-native evidence.

Contribution of the complete evidence construction. Relative to the prototype-only baseline R0, the complete inference rule R8 improves final/average accuracy by $+6.14/+3.57$ percentage points on CIFAR-100 and $+5.63/+3.77$ points on ImageNet-R. Because R0 and R8 use the same learned branches, these gains isolate the effect of the inference-side evidence construction from representation learning.

Leave-one-component-out analysis. Removing any of the three main evidence components decreases both metrics. Compared with R8, removing the branch-native ensemble (R2) decreases final/average accuracy by $1.52/1.39$ points on CIFAR-100 and $2.13/1.86$ points on ImageNet-R. Removing the global reference (R3) produces drops of $1.84/1.50$ and $1.48/1.35$ points, respectively. Likewise, retaining the two ridge sources but removing prototype calibration (R7) reduces the two metrics by $1.40/1.37$ points on CIFAR-100 and $1.40/1.33$ points on ImageNet-R.

These results support different roles for the three sources. The global reference supplies an all-class comparison in a single fixed feature space, the branch-native heads retain task-adapted evidence within their observed support, and prototype calibration adjusts task-dependent score baselines without changing within-task prototype ordering.

Complementarity of the two ridge sources. Variants R4–R6 remove prototype evidence entirely and therefore isolate the relationship between the two analytical sources. On CIFAR-100, combining the two ridge sources (R4) improves final/average accuracy over the global head alone (R5) by $+0.22/+0.40$ points and over the branch-native ensemble alone (R6) by $+0.14/+0.56$ points. On ImageNet-R, the corresponding improvements are $+3.80/+1.03$ over R5 and $+0.10/+0.47$ over R6.

The especially large final-accuracy difference between R4 and R5 on ImageNet-R indicates that the all-class reference does not replace the task-adapted information retained by the branch-native heads. Conversely, the positive R4–R6 differences show that the common reference provides additional information beyond native-space voting.

Contribution of prototype evidence. Comparing R8 with the ridge-only configuration R4 shows that calibrated prototype evidence remains useful even after both ridge sources are available. It increases final/average accuracy by $+1.80/+1.51$ points on CIFAR-100 and $+1.97/+1.91$ points on ImageNet-R. Calibration alone (R1 versus R0) provides a smaller improvement, showing that simple baseline correction does not account for the full gain of the analytical evidence construction.

The strongest results are obtained only when calibrated prototype evidence, the all-class reference, and support-valid branch-native evidence are combined.

B.3 Paired Effect of Full Evidence Construction

The component ablation above uses three seeds on two representative long-sequence settings. We next evaluate the inference-side gain more broadly by comparing prototype-only prediction with the full evidence construction on exactly the same learned branches for every available benchmark–task setting.

For each trained RIDER run, we evaluate two inference rules:

1. **Prototype-only**: the uncalibrated diagonal prototype rule used in the diagnosis; and
2. **Full evidence**: calibrated prototype scores combined with the global reference and support-valid branch-native analytical evidence.

No model parameter is retrained between the two evaluations. Consequently, each pair controls for random initialization and the learned branch representations and directly measures the change induced by the inference rule.

Table 4 summarizes 55 distinct paired runs across seven benchmark–task settings. The five-seed ImageNet-A subset used under the second reporting protocol is drawn from the corresponding ten-seed runs and is therefore not counted a second time when reporting the total number of distinct runs.

Consistency across runs. Both final and average incremental accuracy increase in every one of the 55 paired runs. Importantly, even the minimum gain within every benchmark–task setting remains positive for both metrics. Thus, the average improvement is not driven by a small subset of favorable seeds.

Shorter- and longer-sequence settings. CIFAR-100 and ImageNet-R are each evaluated under two published task partitions. On CIFAR-100, the mean final/average gain is $+3.39/+2.26$ points at $T = 10$ and $+5.41/+3.25$ points at $T = 20$. On ImageNet-R, it is $+2.53/+2.09$ at $T = 10$ and $+6.70/+5.25$ at $T = 40$.

Hence, larger numerical gains are observed under the evaluated longer-sequence settings of both benchmarks. However, the two protocols also change the number of classes introduced per task (10×10 versus 20×5 on CIFAR-100 and 10×20 versus 40×5 on ImageNet-R). We therefore treat this as an empirical pattern across the published protocols rather than as a controlled causal effect of task count or branch count.

B.4 Sensitivity to Fusion Weights

The main experiments use a single fixed fusion configuration,

$$(\eta_{\mathrm{cal}}, \eta_g, \eta_{\mathrm{bn}}) = (0.70, 0.15, 0.15),$$

for calibrated prototype evidence, global reference evidence, and support-valid branch-native evidence, respectively. To examine whether the performance depends strongly on this choice, we vary only the fusion weights while keeping the learned branches, prototype statistics, analytical heads, random projections, and normalization procedure unchanged.

We vary the contribution of calibrated prototype evidence:

$$\eta_{\mathrm{cal}} \in \{0.50, 0.60, 0.70, 0.80\}, \qquad \eta_g = \eta_{\mathrm{bn}} = \frac{1 - \eta_{\mathrm{cal}}}{2}.$$

Therefore, the two analytical evidence sources always receive the same weight, and the experiment only adjusts the relative contribution between prototype evidence and analytical evidence.

Table 8: **Sensitivity to fusion weights.** Final and average incremental accuracy (%) are reported as mean ± sample standard deviation over ten seeds.

Fusion weights			CIFAR-100 ($T = 20$)		ImageNet-R ($T = 40$)	
η_{cal}	η_g	η_{bn}	Final	Avg.	Final	Avg.
0.70	0.15	0.15	91.939 ± 0.208	94.888 ± 0.297	80.143 ± 0.318	85.298 ± 0.547
0.50	0.25	0.25	92.127 ± 0.261	95.031 ± 0.322	80.072 ± 0.299	85.279 ± 0.519
0.60	0.20	0.20	92.069 ± 0.231	94.974 ± 0.316	80.188 ± 0.281	85.325 ± 0.546
0.80	0.10	0.10	91.664 ± 0.202	94.683 ± 0.298	79.913 ± 0.354	85.072 ± 0.597

Sensitivity analysis. Changing the fusion weights within the evaluated range results in only minor performance variations. Relative to the default configuration $(0.70, 0.15, 0.15)$, the maximum change in mean final accuracy is within 0.28 percentage points, while the maximum change in mean average accuracy is within 0.23 points across the two benchmarks.

On CIFAR-100, increasing the contribution of analytical evidence slightly improves the mean performance, with $(\eta_{\text{cal}}, \eta_g, \eta_{\text{bn}}) = (0.50, 0.25, 0.25)$ achieving the highest tested mean accuracy. On ImageNet-R, the highest tested mean is obtained with $(0.60, 0.20, 0.20)$. However, the differences among these configurations remain small compared with the overall improvement introduced by RIDER, indicating that the performance gain does not rely on a narrowly tuned fusion ratio.

Importantly, we retain the same default configuration $(\eta_{\text{cal}}, \eta_g, \eta_{\text{bn}}) = (0.70, 0.15, 0.15)$ for all reported experiments rather than selecting a dataset-specific weight combination. This provides a consistent evaluation protocol across benchmarks and avoids additional dataset-level hyperparameter tuning.

Seed-wise behavior. Aggregate means may hide seed-specific variations. Therefore, Figure 4 reports paired changes in final accuracy for each seed relative to the default fusion configuration. Because each comparison uses the same trained RIDER model and differs only in the final evidence weights, the observed differences isolate the effect of fusion-weight selection.

On CIFAR-100, increasing the contribution of analytical evidence with

$$(\eta_{\text{cal}}, \eta_g, \eta_{\text{bn}}) = (0.50, 0.25, 0.25) \quad \text{or} \quad (0.60, 0.20, 0.20)$$

improves final accuracy in all ten paired seeds, whereas assigning a larger weight to calibrated prototype evidence $(0.80, 0.10, 0.10)$ decreases final accuracy in all ten seeds.

The seed-wise behavior on ImageNet-R is less uniform. For the configuration

$$(\eta_{\text{cal}}, \eta_g, \eta_{\text{bn}}) = (0.60, 0.20, 0.20),$$

five seeds improve, three remain unchanged, and two decrease relative to the default setting. Therefore, the positive average gain observed for this configuration should be interpreted as an aggregate tendency rather than a universal improvement for every individual seed.

B.5 Accuracy–Parameter Trade-off of Progressive Rank Reduction

RIDER applies the rank schedule

$$r_t = \max\big(r_{\min}, \lfloor r_0 \rho^{t-1} \rfloor\big)$$

only when allocating the newly introduced task-specific direction at task t. Previously learned directions retain their original ranks and are not modified. Therefore, the purpose of rank reduction is to control the growth of newly introduced trainable parameters rather than to compress historical task-specific directions.

To quantify this trade-off, we compare the full RIDER configuration with a variant that disables progressive rank reduction while keeping all other components unchanged. Both variants use seed 1993. Trainable parameter counts are averaged across tasks and include the temporary training classifier.

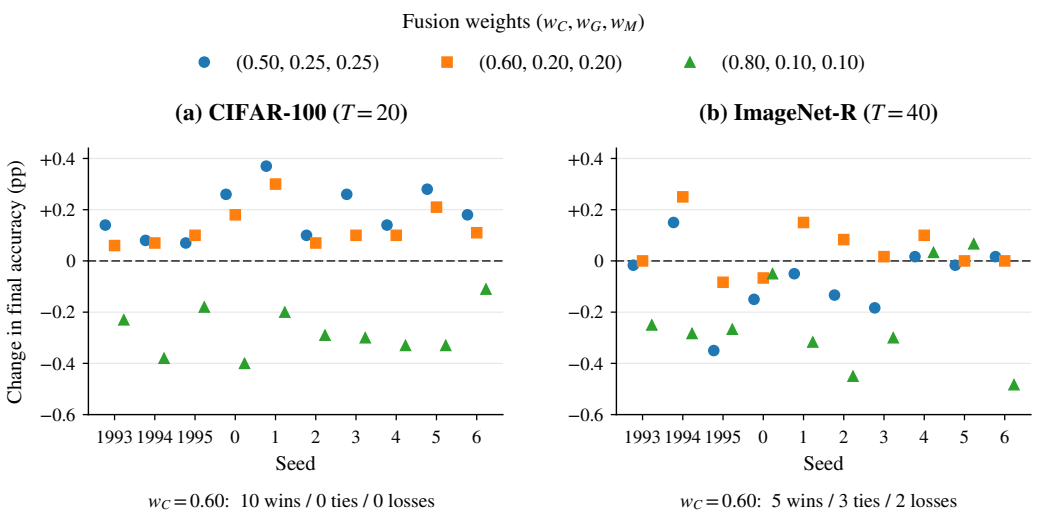


Figure 4: **Seed-wise sensitivity to final fusion weights.** Each marker reports the change in final accuracy (percentage points) relative to $(w_C, w_G, w_M) = (0.70, 0.15, 0.15)$ for the same trained run on CIFAR-100 ($T = 20$) or ImageNet-R ($T = 40$). The weights correspond to calibrated prototype evidence, global reference evidence, and branch-native evidence, respectively. Positive values favor the alternative weight configuration and the dashed line denotes zero change. The panels use the same vertical scale and seed ordering; small horizontal offsets separate configurations at each seed.

Table 9: **Accuracy–parameter trade-off of progressive rank reduction** (seed 1993). Final and average incremental accuracy (%) are reported together with the average number of trainable parameters across tasks.

Dataset	Variant	Trainable (M)	Reduction (%)	$\bar{A}_T$	Avg.
CIFAR-100 ($T = 20$)	w/o rank decay	0.2251	–	92.03	94.96
	RIDER	0.1615	28.25	91.81	94.92
ImageNet-R ($T = 40$)	w/o rank decay	0.2805	–	80.38	85.64
	RIDER	0.2279	18.73	80.05	85.46
ImageNet-A ($T = 10$)	w/o rank decay	0.2919	–	64.45	74.29
	RIDER	0.2477	15.16	63.92	73.81
VTAB ($T = 5$)	w/o rank decay	0.2842	–	95.90	96.08
	RIDER	0.2473	12.97	95.66	96.05
DomainNet ($T = 5$)	w/o rank decay	0.7060	–	75.81	81.15
	RIDER	0.4056	42.55	75.56	81.07

Progressive rank reduction reduces the average number of trainable parameters across all evaluated benchmarks. Specifically, it decreases the trainable parameters by 28.25%, 18.73%, 15.16%, 12.97%, and 42.55% on CIFAR-100, ImageNet-R, ImageNet-A, VTAB, and DomainNet, respectively.

The corresponding changes in final/average incremental accuracy relative to disabling rank reduction are $-0.22/-0.04$, $-0.33/-0.18$, $-0.53/-0.48$, $-0.24/-0.03$, and $-0.25/-0.08$ percentage points, respectively.

These results show that progressive rank reduction substantially reduces the number of newly introduced trainable parameters while maintaining comparable incremental accuracy. Since this comparison uses a single seed, it is intended to illustrate the parameter–accuracy trade-off of rank reduction rather than provide a statistical evaluation of its performance effect.

Table 10: **Effect of score normalization on final accuracy (%).** Results are mean ± sample standard deviation across ten paired seeds. The learned branches, analytical heads, and fusion weights are identical across normalization rules.

Normalization rule	CIFAR-100 ($T = 20$)	ImageNet-R ($T = 40$)
Evaluation-set standardization	91.939 ± 0.208	80.143 ± 0.318
Fixed training statistics	91.978 ± 0.196	$\mathbf{80.155} \pm 0.350$
Per-sample standardization	$\mathbf{92.073} \pm 0.260$	80.077 ± 0.278

B.6 Inductive Normalization Analysis

The main RIDER inference rule uses z-score statistics computed from the unlabeled evaluation-score matrices. This produces a transductive normalization rule because the score of one evaluation input is standardized using moments that depend on other inputs in the same evaluation set.

To determine whether the reported performance critically depends on this choice, we evaluate two alternatives that do not use cross-example statistics from the evaluation set:

1. **Per-sample standardization**, which computes score moments independently for each evaluation input; and
2. **Fixed training-reference standardization**, which estimates all normalization moments once from a fixed set of final-task training inputs and freezes them before evaluation.

The learned branches, class prototypes, analytical classifiers, random projections, and fusion weights are held fixed. Only the normalization operators are replaced.

B.6.1 Results

Across ten paired seeds, fixed training-reference statistics change mean final accuracy by $+0.039$ and $+0.012$ percentage points on CIFAR-100 ($T = 20$) and ImageNet-R ($T = 40$), respectively. Per-sample standardization changes the corresponding means by $+0.134$ and -0.067 points.

Figure 5 reports the corresponding paired seed-wise differences and pointwise 95% confidence intervals. None of the four comparisons is significant after Holm correction. These results show that evaluation-set-independent normalization is feasible in the two evaluated settings, but they do not establish statistical equivalence between the normalization rules.

B.6.2 Notation and valid branch support

We consider the final stage T. Let $\mathcal{R}$ denote the fixed reference inputs drawn from the final task's training data, and let $\mathcal{C}_{1:T}$ denote all classes seen up to the final stage.

For branch b, the set of classes supported by direct observations is

$$\mathcal{V}_b = \{c \in \mathcal{C}_{1:T} : \tau(c) \geq b\}, \tag{19}$$

which is equivalent to the support definition in section 4.2. For class c, the eligible branches are

$$\mathcal{B}_c = \{b : c \in \mathcal{V}_b\} = \{1, \ldots, \tau(c)\}. \tag{20}$$

For a score matrix Q, every mean and standard deviation below is computed over all entries of the explicitly indicated row or submatrix. Invalid branch–class entries are never included in moment estimation or branch voting.

B.6.3 Per-sample standardization

For an evaluation input x and a valid class set $\mathcal{V}$, we define

$$\mathcal{Z}_{\mathcal{V}}^{\mathrm{row}}(Q)_{x,c} = \frac{Q_{x,c} - \mathrm{mean}(Q_{x,\mathcal{V}})}{\mathrm{std}(Q_{x,\mathcal{V}}) + \varepsilon_z}, \qquad c \in \mathcal{V}. \tag{21}$$

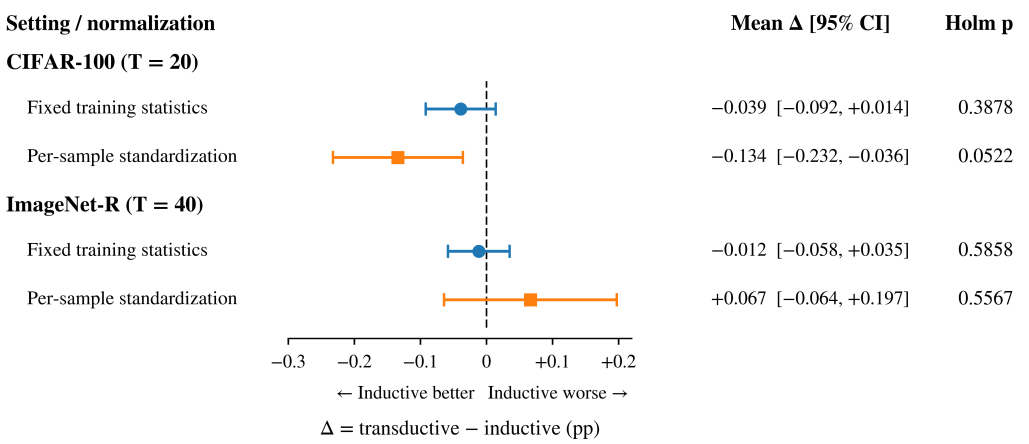


Figure 5: **Paired final-accuracy differences between evaluation-set standardization and two evaluation-set-independent alternatives.** Each point reports the mean paired difference over ten seeds and the error bar denotes its pointwise 95% confidence interval. Differences are defined as evaluation-set standardization minus the corresponding alternative; negative values therefore favor the alternative.

The mean and standard deviation are computed only from the score vector of the current input x. Therefore, the prediction for one input does not depend on the presence or scores of any other evaluation example.

At the branch level, Eq. 21 is applied across the valid class columns $\mathcal{V}_b$ of each branch. After branch-native aggregation, the three complete evidence sources are standardized again, independently for the same input, across all seen classes.

B.6.4 Fixed training-reference statistics

The second alternative estimates one fixed scalar mean and standard deviation for each score source and valid class set from the reference inputs $\mathcal{R}$:

$$\begin{aligned} \mu^{\mathrm{ref}}_{Q,\mathcal{V}} &= \mathrm{mean}(Q_{\mathcal{R},\mathcal{V}}), \\ \sigma^{\mathrm{ref}}_{Q,\mathcal{V}} &= \mathrm{std}(Q_{\mathcal{R},\mathcal{V}}), \\ \mathcal{Z}^{\mathrm{ref}}_{\mathcal{V}}(Q)_{x,c} &= \frac{Q_{x,c} - \mu^{\mathrm{ref}}_{Q,\mathcal{V}}}{\sigma^{\mathrm{ref}}_{Q,\mathcal{V}} + \varepsilon_z}. \end{aligned} \tag{22}$$

The resulting moments are frozen before evaluation and reused for every test input.

In the two evaluated settings, the final task contains five classes, and $\mathcal{R}$ is drawn from those final-task training examples. Importantly, these reference examples also participated in fitting the analytical heads; they do *not* constitute a held-out calibration set. The purpose of this experiment is therefore specifically to test whether inference requires statistics from the evaluation set, rather than to evaluate a separately held-out calibration protocol.

Although the reference inputs themselves belong to only the five final-task classes, the score columns used to estimate $\mu^{\mathrm{ref}}_{Q,\mathcal{V}}$ and $\sigma^{\mathrm{ref}}_{Q,\mathcal{V}}$ are not restricted to those five classes. Each branch uses all of its valid columns $\mathcal{V}_b$, and the fusion-level sources use all seen-class columns $\mathcal{C}_{1:T}$.

B.6.5 BRANCH AGGREGATION

For either normalization rule $\nu \in \{\mathrm{row}, \mathrm{ref}\}$, we first normalize each branch only over its valid support and then aggregate the eligible votes for each class:

$$S^{\mathrm{bn},\nu}_{x,c} = \frac{1}{|\mathcal{B}_c|} \sum_{b \in \mathcal{B}_c} \mathcal{Z}^{\nu}_{\mathcal{V}_b}(S^b)_{x,c}. \tag{23}$$

This preserves exactly the same support restriction as the main method: a branch contributes to class c only if that class has been directly observed in the branch's fixed feature space.

B.6.6 FUSION-LEVEL NORMALIZATION

After branch aggregation, each of the three complete evidence sources is normalized separately over all seen classes and fused with the same weights used in the main experiments:

$$\begin{aligned} S^{\nu} &= 0.70\, \mathcal{Z}^{\nu}_{\mathcal{C}_{1:T}}(S^{\mathrm{cal}}) + 0.15\, \mathcal{Z}^{\nu}_{\mathcal{C}_{1:T}}(S^{g}) \\ &\quad + 0.15\, \mathcal{Z}^{\nu}_{\mathcal{C}_{1:T}}(S^{\mathrm{bn},\nu}), \\ \hat{y}^{\nu}(x) &= \arg\max_{c \in \mathcal{C}_{1:T}} S^{\nu}_{x,c}. \end{aligned} \tag{24}$$

No fusion weight or classifier parameter is retuned for either normalization alternative.

B.6.7 PROCESSING ORDER

For fixed-reference normalization, the processing order is important. We first estimate and freeze the branch-level moments on $\mathcal{R}$. Using those frozen branch moments, we then reconstruct the branch-native ensemble scores for the same reference inputs. Only after this reconstruction do we estimate the three fusion-level moment pairs for calibrated prototype evidence, global reference evidence, and branch-native evidence.

Evaluation then uses the frozen branch-level and fusion-level moments without consulting the evaluation set.

For per-sample normalization, the same sequence of operations is applied independently to every input: branch-level normalization, branch-native aggregation, source-level normalization, and finally weighted fusion.

C THEORETICAL PROPERTIES OF SUPPORT-AWARE ANALYTICAL EVIDENCE

This section establishes three properties of RIDER's analytical evidence construction. First, branch-level prototype calibration exactly preserves within-task class rankings. Second, sufficient-statistic accumulation in a fixed branch space recovers the corresponding batch ridge-regression solution. Third, a positive weighted combination of source-wise margins provides a sufficient condition for correct fused prediction. The first two results are exact algebraic properties, whereas the third characterizes when complementary evidence sources are sufficient to correct a class prediction. These results concern the stated scoring rules on fixed representations and should not be interpreted as an unconditional accuracy guarantee for the complete continual-learning system.

Notation. At stage t, the seen classes are

$$\mathcal{C}_{1:t} = \bigcup_{s=1}^{t} \mathcal{C}_s,$$

where the task-specific class sets $\mathcal{C}_s$ are disjoint and nonempty, and $K_t = |\mathcal{C}_{1:t}|$. The owner task of class c is denoted by $\tau(c)$. All score comparisons assume finite real-valued scores and use the same deterministic class ordering to resolve ties. Whenever evaluation-set standardization is involved, the evaluation set $\mathcal{X}_{\mathrm{eval}}$ is treated as fixed.

C.1 WITHIN-TASK RANKING INVARIANCE

RIDER calibrates the diagonal prototype score by subtracting the branch-level positive-response mean:

$$S^{\mathrm{cal}}_{x,c} = s^{\mathrm{diag}}(x,c) - \mu^{+}_{\tau(c)}, \qquad \mu^{+}_{b} = \frac{1}{|\mathcal{D}_b|} \sum_{(x',y')\in\mathcal{D}_b} \max_{c'\in\mathcal{C}_b} s^{\mathrm{diag}}(x',c'). \tag{25}$$

Because all classes belonging to the same task receive the same additive shift, the following property does not require any distributional assumption on the scores.

Lemma C.1 (Within-task ranking invariance) *For every input x, branch b, and pair $c_1, c_2 \in \mathcal{C}_b$,*

$$S^{\mathrm{cal}}_{x,c_1} - S^{\mathrm{cal}}_{x,c_2} = s^{\mathrm{diag}}(x,c_1) - s^{\mathrm{diag}}(x,c_2). \tag{26}$$

Consequently, calibration preserves the complete class ordering within task b. Under the common tie-breaking rule,

$$\arg\max_{c\in\mathcal{C}_b} S^{\mathrm{cal}}_{x,c} = \arg\max_{c\in\mathcal{C}_b} s^{\mathrm{diag}}(x,c). \tag{27}$$

Proof. Since $c_1, c_2 \in \mathcal{C}_b$, we have $\tau(c_1) = \tau(c_2) = b$. Therefore both scores receive the same shift:

$$\begin{aligned} S^{\mathrm{cal}}_{x,c_1} - S^{\mathrm{cal}}_{x,c_2} &= \left(s^{\mathrm{diag}}(x,c_1) - \mu^{+}_{b}\right) - \left(s^{\mathrm{diag}}(x,c_2) - \mu^{+}_{b}\right) \\ &= s^{\mathrm{diag}}(x,c_1) - s^{\mathrm{diag}}(x,c_2). \end{aligned}$$

Hence every pairwise score difference, including a zero difference, is unchanged. The within-task ordering and set of maximizers are therefore identical before and after calibration. Applying the same deterministic tie-breaking rule gives the same within-task prediction. □

Implication for the diagnosis. If prediction is restricted to the ground-truth task, calibrated and uncalibrated prototype scores yield exactly the same oracle-task prediction for every sample and therefore the same oracle accuracy. Branch-level prototype calibration can thus alter the prototype-based final prediction only through competition among classes belonging to different tasks; it cannot change the class ordering inside any individual task.

This invariance does not imply that the diagnostic e_{within} term must remain numerically unchanged after calibration, because that term is defined jointly with the event that the predicted class belongs to the correct task, and the set of samples satisfying that event can change. Nor does the lemma guarantee that calibration improves cross-task decisions. Finally, the result applies only to the branch-level prototype shift; the complete RIDER evidence fusion also includes analytical ridge scores and can therefore alter the final within-task ordering.

C.2 INCREMENTAL–BATCH EQUIVALENCE IN FIXED BRANCH SPACES

Consider a fixed branch b and one ridge classifier defined in its feature space. Let

$$h_b(x) \in \mathbb{R}^D$$

denote its complete analytical feature map, including the frozen branch mapping, feature normalization, and fixed random projection. Throughout the accumulation considered below, h_b and the feature-extraction preprocessing remain fixed.

For each task $s \in \{b, \ldots, t\}$, let $H_{b,s} \in \mathbb{R}^{n_s \times D}$ contain these features as rows, where $n_s = |\mathcal{D}_s|$. Let $Y_s \in \mathbb{R}^{n_s \times K_t}$ contain the corresponding global one-hot labels. When new classes are introduced, previously accumulated target statistics are zero-padded to the enlarged K_t-class output space.

Proposition C.2 (Incremental–batch equivalence) *For a fixed feature map h_b and regularization coefficient $\lambda > 0$, define*

$$G_{b,t} = \sum_{s=b}^{t} H_{b,s}^{\top} H_{b,s}, \qquad C_{b,t} = \sum_{s=b}^{t} H_{b,s}^{\top} Y_s. \tag{28}$$

In exact arithmetic, the incrementally constructed ridge head

$$W_{b,t}^{\mathrm{inc}} = (G_{b,t} + \lambda I_D)^{-1} C_{b,t} \tag{29}$$

is the unique minimizer of

$$\mathcal{J}_{b,t}(W) = \sum_{s=b}^{t} \|H_{b,s}W - Y_s\|_F^2 + \lambda \|W\|_F^2. \tag{30}$$

Therefore,

$$W_{b,t}^{\mathrm{inc}} = W_{b,t}^{\mathrm{batch}} \tag{31}$$

for the same observed features, target columns, preprocessing, and regularization coefficient.

Proof. Starting from zero statistics, processing task s adds $H_{b,s}^\top H_{b,s}$ to the Gram matrix and $H_{b,s}^\top Y_s$ to the feature–target matrix. Induction over the observed tasks therefore yields Eq. 28. Zero-padding previously accumulated target statistics preserves all existing class columns and assigns zero historical contribution to newly introduced columns, exactly matching the global one-hot encoding of the corresponding batch data.

Differentiating Eq. 30 with respect to W gives

$$\nabla_W \mathcal{J}_{b,t}(W) = 2\,(G_{b,t} + \lambda I_D)\,W - 2C_{b,t}. \tag{32}$$

For any nonzero vector $v \in \mathbb{R}^D$,

$$v^\top (G_{b,t} + \lambda I_D)\, v = \sum_{s=b}^{t} \|H_{b,s}v\|_2^2 + \lambda \|v\|_2^2 > 0.$$

Thus $G_{b,t} + \lambda I_D$ is positive definite and $\mathcal{J}_{b,t}$ is strictly convex. Its unique stationary point therefore satisfies

$$(G_{b,t} + \lambda I_D)\, W = C_{b,t},$$

which yields Eq. 29.

For comparison, define the hypothetical batch matrices

$$\overline{H}_{b,t} = \begin{bmatrix} H_{b,b} \\ H_{b,b+1} \\ \vdots \\ H_{b,t} \end{bmatrix}, \qquad \overline{Y}_{b,t} = \begin{bmatrix} Y_b \\ Y_{b+1} \\ \vdots \\ Y_t \end{bmatrix}.$$

They satisfy

$$\overline{H}_{b,t}^\top \overline{H}_{b,t} = G_{b,t}, \qquad \overline{H}_{b,t}^\top \overline{Y}_{b,t} = C_{b,t}.$$

Hence batch fitting produces exactly the same normal equations and the same unique ridge solution as sequential sufficient-statistic accumulation. □

Implication for RIDER. Once branch b is saved, its representation and random projection remain fixed. The two additive statistics $G_{b,t}$ and $C_{b,t}$ therefore retain all information required to solve the corresponding fixed-feature ridge objective without storing previous raw examples or per-example features.

The proposition applies independently to each branch-native head and to the global reference head, with each head using its own fixed random projection and ridge regularization. In particular, the global reference head and the branch-native head associated with branch 1 are separate classifiers even though both are constructed from the same fixed branch-1 representation.

For a branch created at task b, the equivalence covers only the tasks $b, \ldots, t$ that are actually observed in its feature space. It does not justify constructing or using branch-native evidence for classes from tasks preceding b. The result is an equivalence between incremental and batch fitting of a fixed-feature ridge classifier; it is not an equivalence to end-to-end joint training and does not imply that old-class accuracy remains unchanged as new classes are introduced. Finite-precision linear solvers may also introduce small numerical differences from the exact-arithmetic identity.

C.3 Sufficient Margin Conditions for Evidence Fusion

For a fixed evaluation set with $K_t \geq 2$, let

$$S^{\mathrm{cal}}, \qquad S^{g}, \qquad S^{\mathrm{bn}} \in \mathbb{R}^{n_{\mathrm{eval}} \times K_t}$$

denote the three complete evidence matrices: calibrated prototype evidence, global reference evidence, and support-valid branch-native evidence. Here S^{bn} already includes branch-level standardization and aggregation over eligible branches.

Define the source-standardized evidence by

$$Z^r = \mathrm{ZS}(S^r), \qquad \mathrm{ZS}(M) = \frac{M - \mathrm{mean}(M)}{\mathrm{std}(M) + \varepsilon_z}, \qquad r \in \mathcal{R} = \{\mathrm{cal}, g, \mathrm{bn}\}. \tag{33}$$

The mean and standard deviation are computed over all entries of the corresponding complete score matrix, consistent with the main evaluation-set-standardized inference rule.

Let nonnegative fusion weights η_r satisfy

$$\sum_{r \in \mathcal{R}} \eta_r = 1,$$

and define

$$S_{x,c} = \sum_{r \in \mathcal{R}} \eta_r Z^r_{x,c}. \tag{34}$$

For a sample x with true class y, define the source-specific and fused classification margins as

$$\begin{aligned} m_r(x) &= Z^r_{x,y} - \max_{c \in \mathcal{C}_{1:t} \setminus \{y\}} Z^r_{x,c}, \\ m_S(x) &= S_{x,y} - \max_{c \in \mathcal{C}_{1:t} \setminus \{y\}} S_{x,c}. \end{aligned} \tag{35}$$

Proposition C.3 (Sufficient margin condition) *For every sample in the fixed evaluation set,*

$$m_S(x) \geq \sum_{r \in \mathcal{R}} \eta_r m_r(x). \tag{36}$$

Consequently,

$$\sum_{r \in \mathcal{R}} \eta_r m_r(x) > 0 \quad \Longrightarrow \quad \arg\max_{c \in \mathcal{C}_{1:t}} S_{x,c} = y. \tag{37}$$

For the RIDER weights used in the main experiments, a sufficient condition is

$$0.70\, m_{\mathrm{cal}}(x) + 0.15\, m_g(x) + 0.15\, m_{\mathrm{bn}}(x) > 0. \tag{38}$$

Proof. For any incorrect class $c \neq y$ and evidence source r,

$$Z^r_{x,y} - Z^r_{x,c} \geq Z^r_{x,y} - \max_{c' \neq y} Z^r_{x,c'} = m_r(x).$$

Multiplying by the nonnegative weights and summing over evidence sources gives

$$\begin{aligned} S_{x,y} - S_{x,c} &= \sum_{r \in \mathcal{R}} \eta_r \left(Z^r_{x,y} - Z^r_{x,c} \right) \\ &\geq \sum_{r \in \mathcal{R}} \eta_r m_r(x). \end{aligned}$$

Taking the minimum over all incorrect classes $c \neq y$ yields Eq. 36. If its right-hand side is strictly positive, then $S_{x,y} > S_{x,c}$ for every incorrect class, and y is therefore the unique maximizer. Substituting the fusion weights used by RIDER gives Eq. 38. □

Implication for complementary evidence. Suppose the calibrated prototype source alone misclassifies a sample, so that $m_{\mathrm{cal}}(x) < 0$. For RIDER's implemented weights, a sufficient condition for the two analytical evidence sources to overcome this incorrect preference is

$$0.15\, m_g(x) + 0.15\, m_{\mathrm{bn}}(x) > -0.70\, m_{\mathrm{cal}}(x). \tag{39}$$

Thus, sufficiently strong positive analytical evidence can compensate for a negative margin from the calibrated prototype source.

This condition is sufficient but not necessary. Different evidence sources may attain their worst pairwise margins against different incorrect classes, so a nonpositive value of the lower bound in Eq. 36 does not imply that the fused prediction is incorrect. The result requires no independence assumption among the three evidence sources. It also does not establish optimality of the chosen fusion weights or guarantee that fusion outperforms every individual source on every sample.

Evaluation-set scope. The margins above are defined after applying the normalization used by the main RIDER inference rule. Because the corresponding normalization moments depend on the complete unlabeled evaluation set, changing the evaluation set can change the standardized scores of an otherwise unchanged sample. The proposition is therefore a deterministic statement for a fixed evaluation set, rather than a batch-independent or out-of-sample generalization guarantee.

All margins in this subsection concern the final fused class scores. Unlike the branch-level calibration result in Lemma C.1, the complete evidence fusion can alter both cross-task competition and within-task class rankings.

C.4 Support Coverage and Pairwise Score Structure

This subsection characterizes how observed support enters the native ensemble and motivates a reference with complete class coverage.

Coverage within the constructed branch family. At stage t, branch b observes exactly $\mathcal{C}_{b:t}$. Therefore, branch 1 covers $\mathcal{C}_{1:t}$, whereas every branch $b > 1$ lacks the nonempty class set $\mathcal{C}_{1:b-1}$. Branch 1 is thus the unique all-class branch within the sequentially constructed family considered here. This statement does not imply that its representation is the best reference space among other possible constructions.

Support-count normalization. Fix the evaluation set and define

$$z_b(x,c) = \left[\mathrm{ZS}\left(S^b_{:,\mathcal{C}_{b:t}}\right)\right]_{x,c}, \qquad a_c(x) = \frac{1}{\tau(c)} \sum_{b=1}^{\tau(c)} z_b(x,c). \tag{40}$$

Here $z_b(x,c)$ is defined only on observed support, and $a_c(x) = S^{\mathrm{bn}}_{x,c}$ precedes the source-level standardization in Eq. (5). Division by $\tau(c)$ makes aggregation an arithmetic mean: if every eligible branch assigns the same value q_c, then $a_c = q_c$, irrespective of the number of contributing branches. This removes direct multiplication by support count, but does not equalize score reliability or ensure comparable classwise uncertainty.

Proposition C.4 (Support-composition identity). For two classes y, c with $m = \tau(y) < n = \tau(c)$,

$$\begin{aligned} a_y(x) - a_c(x) = \frac{1}{m} \sum_{b=1}^{m} \big(z_b(x,y) - z_b(x,c)\big) \\ + \frac{n-m}{n} \left(\frac{1}{m} \sum_{b=1}^{m} z_b(x,c) - \frac{1}{n-m} \sum_{b=m+1}^{n} z_b(x,c) \right). \end{aligned} \tag{41}$$

Proof. Split the sum for a_c into its common and additional supporting branches. Add and subtract the mean of $z_b(x,c)$ over branches $1{:}m$, then rearrange to obtain Eq. (41). □

The first term compares both classes within their common supporting branches. The second records how the competitor's mean score differs between common and additional supporting branches; its

sign is unrestricted. Every quantity is directly supported, and no score for y in branches $b > m$ is required. When $m = n$, both classes use the same branches and their difference is simply the average pairwise difference; branch-level additive centering also cancels within each pair. The case $m > n$ follows by exchanging the classes and negating the identity.