WHEN LOCAL CLASSIFIERS BECOME REPRESENTATIONS: THE PLASTICITY TAX OF UNSUPPORTED OUTPUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

In class-incremental local learning, classifier outputs can become downstream representations before all classes are observed. Outputs for classes without positive examples remain unsupported, yet participate in local softmax competition and receive updates. We investigate the representation-level plasticity tax through matched interventions on unsupported-output count, competition strength, exposure depth, and training phase, holding capacity, training data, replay, and optimization budgets fixed. Local softmax analysis motivates Anchor-Forward, an exact intervention on the unsupported-coordinate update path. At a common input and model state, it preserves forward computation and supported-logit gradients while blocking direct local gradients through unsupported coordinates. In DeeperForward, depth and phase interventions identify the largest losses in shallow layers and during new-task acquisition. On CIFAR-100, Anchor-Forward improves arriving-task accuracy by 27.66 percentage points over matched competition in a two-task experiment with 80 unsupported outputs. In a ten-task stream, it recovers 16.13 points in mean arriving-task accuracy. Experiments with HCL-FF and GTSRB demonstrate plasticity recovery across architectures and domains. These findings indicate that output support is part of representation design when local classifier responses are propagated to subsequent learners.

1 INTRODUCTION

In class-incremental learning, new classes arrive while prediction covers the classes observed so far (Rebuffi et al., 2017; van de Ven & Tolias, 2019). A preallocated classifier may already contain outputs for classes that have never supplied positive examples. We refer to these coordinates as *unsupported outputs*. Their role becomes especially consequential when classifier responses are reused as inputs to subsequent modules. These responses then serve two purposes: they support local prediction and provide representations for downstream learning. Updating an unsupported coordinate can therefore affect both the classifier that produces it and the learners that consume it.

This dual role arises in propagated local learning. The Forward-Forward (FF) algorithm trains layers through local goodness objectives rather than end-to-end credit assignment (Hinton, 2022). Subsequent work introduces class-wise competition, deeper residual architectures, and hierarchical or metric-based objectives (Papachristodoulou et al., 2024; Sun et al., 2025; Yao et al., 2026; Xu et al., 2026). In architectures such as DeeperForward and HCL-FF, locally supervised class-wise responses can continue along the forward path. Figure 1 highlights the unsupported-coordinate update path in this setting. This dual role raises a specific question: how does the support of propagated outputs affect the ability to learn an arriving task?

Unsupported outputs participate in local softmax competition even before their classes appear. Each receives a positive loss derivative despite having no positive examples. When the resulting class-wise responses are propagated, the consequences of these updates extend beyond a local prediction. They alter the representation available to subsequent modules. Exposure at different depths can therefore affect different numbers of downstream learners. We call the resulting loss of new-task adaptability the *plasticity tax of unsupported outputs*. This perspective makes output support a property of representation design, rather than merely a choice of classification capacity.

This view suggests that output support is part of the learning architecture itself. Forward competition and coordinate updates play different roles, although standard training couples them. We separate

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.

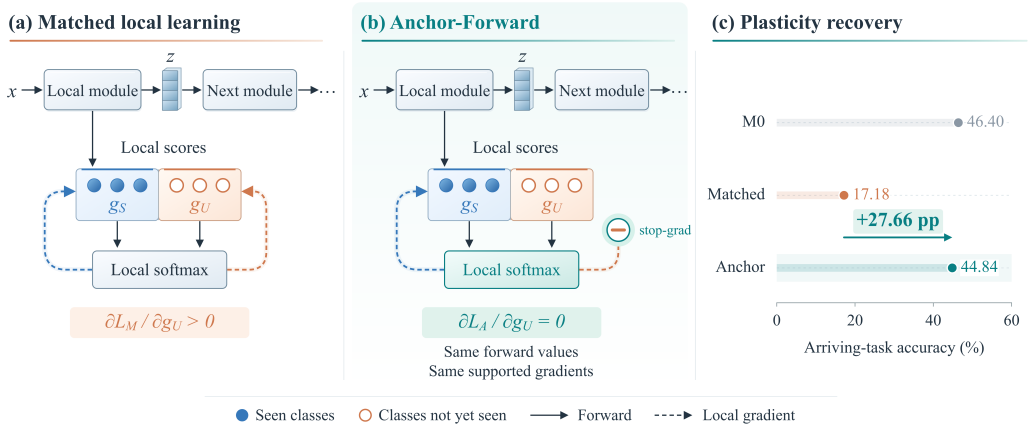

Figure 1. Anchor-Forward preserves forward competition and recovers plasticity. (a) Matched local learning updates both supported and unsupported score coordinates. (b) Anchor retains forward values and supported-score gradients while blocking the direct local-loss gradient through unsupported coordinates, at a common input and model state. (c) Mean arriving-task accuracy for DeeperForward on CIFAR-100 2×10 with $M = 80$. Anchor improves matched accuracy by 27.66 percentage points; M0 is the supported-only reference.

these effects with Anchor-Forward, an exact intervention on the unsupported-coordinate backward path. At a common model state, Anchor-Forward keeps the same logits, local softmax, and scalar loss. It also preserves all supported-coordinate gradients. Only gradients through unsupported coordinates are removed. Their forward values are retained and continue to propagate downstream. The intervention therefore isolates how unsupported-coordinate updates influence later representation learning. Figure 1 summarizes the propagated local-learning setting, the unsupported-coordinate update path, and the Anchor-Forward intervention.

The experiments reveal a clear pattern. In DeeperForward, unsupported outputs can impose a large cost on learning new classes. The effect is strongest when unsupported coordinates are updated in shallow layers during new-task acquisition. Removing this path recovers much of the lost plasticity. With 80 unsupported outputs on CIFAR-100, Anchor-Forward improves arriving-task accuracy by 27.66 percentage points over the matched condition. The same effect persists over a ten-task stream, where Anchor-Forward recovers 16.13 points in mean arriving-task accuracy. HCL-FF and GTSRB show the same recovery pattern under another propagated-local objective and a different data domain.

Our contributions are threefold:

- We identify a representation-level plasticity tax caused by unsupported outputs in propagated local learning and localize its dominant source to shallow unsupported-coordinate updates during new-task acquisition.
- We introduce Anchor-Forward, an exact intervention that decouples forward competition from gradient support. At a common model state, it preserves the forward computation and supported-logit learning signals while removing only the direct gradient path through unsupported coordinates.
- We establish output support as a representation-design factor in propagated local learning. Through controlled studies across architectures, continual-learning regimes, class orders, and data domains, we show that unsupported-coordinate updates systematically shape downstream representations and new-task plasticity.

2 Unsupported Outputs in Propagated Local Learning

2.1 Supported and Unsupported Coordinates

Output support changes as a stream introduces new classes. Let S_t contain classes with positive examples by task t. Let U_t contain selected classes that remain unobserved, with $M = |U_t|$. Each local module produces class-wise scores $\mathbf{g}_\ell(x) \in \mathbb{R}^K$. The output capacity K remains fixed as support grows. Thus, varying M changes local competition within the same network.

2.2 UNSUPPORTED OUTPUTS ENTER LOCAL COMPETITION

The local loss couples support scope to the softmax denominator. We omit the layer index and assume $y \in S_t$ and $T > 0$. Define the supported and unsupported partition sums as

$$Z_S = \sum_{c \in S_t} e^{g_c/T}, \qquad Z_U = \sum_{u \in U_t} e^{g_u/T}. \tag{1}$$

The supported-only reference, denoted M0, uses $\mathcal{L}_S = -g_y/T + \log Z_S$. The matched condition includes both sets:

$$\mathcal{L}_M = -\frac{g_y}{T} + \log(Z_S + Z_U), \qquad \mathcal{L}_M - \mathcal{L}_S = \log\left(1 + \frac{Z_U}{Z_S}\right). \tag{2}$$

Both conditions retain the same output capacity. M0 selects the supported denominator; matched competition adds the selected unsupported coordinates.

Unsupported softmax mass describes competition at a given model state. Let $q_U = Z_U/(Z_S + Z_U)$ and $p_{S,k} = e^{g_k/T}/Z_S$. The local derivatives are

$$\frac{\partial \mathcal{L}_M}{\partial g_k} = \frac{(1 - q_U)p_{S,k} - \mathbf{1}[k = y]}{T}, \quad k \in S_t; \qquad \frac{\partial \mathcal{L}_M}{\partial g_u} = \frac{p_{M,u}}{T} > 0, \quad u \in U_t. \tag{3}$$

Every unsupported coordinate receives a positive loss derivative despite having no positive examples. Gradient descent therefore creates a local pressure to lower these scores. Output count, temperature, and score geometry all influence this pressure. The count and temperature interventions examine how this exposure affects acquisition.

2.3 FROM LOCAL UPDATES TO REPRESENTATION CHANGES

A local classifier can supply both predictions and features. In a terminal classifier, output updates mainly affect the final decision. In a propagated local classifier, they also change representations consumed by later modules. Each module learns from its own objective, while its numerical outputs feed the next learner.

Exposure depth therefore changes which learners receive the affected representation. Earlier affected representations are consumed by more downstream modules. This forward route connects shallow local updates to later representation learning. Can forward competition be retained while the unsupported-coordinate update path is removed?

3 ANCHOR-FORWARD: SEPARATING COMPETITION FROM UPDATES

Anchor-Forward separates forward competition from unsupported-coordinate updates. It preserves the full local softmax and supported-logit gradients while blocking direct gradients through unsupported coordinates. Figure 2 summarizes layer-local deployment, changing class support, and local-loss construction.

3.1 AN EXACT LOCAL BACKWARD INTERVENTION

Let sg denote stop-gradient: identity forward evaluation with a prescribed zero backward Jacobian. Anchor evaluates

$$\mathcal{L}_A = -\frac{g_y}{T} + \log\left(\sum_{c \in S_t} e^{g_c/T} + \sum_{u \in U_t} e^{\mathrm{sg}(g_u)/T}\right). \tag{4}$$

For a fixed input and model state, its automatic-differentiation identities are

$$\boxed{\mathcal{L}_A = \mathcal{L}_M, \qquad \nabla_{\mathbf{g}_S}\mathcal{L}_A = \nabla_{\mathbf{g}_S}\mathcal{L}_M, \qquad \nabla_{\mathbf{g}_U}\mathcal{L}_A = 0.} \tag{5}$$

The identities preserve the scalar loss, full softmax, and supported-logit learning signals. They remove the unsupported-coordinate contribution to the local update. The complete derivation appears in Appendix A.

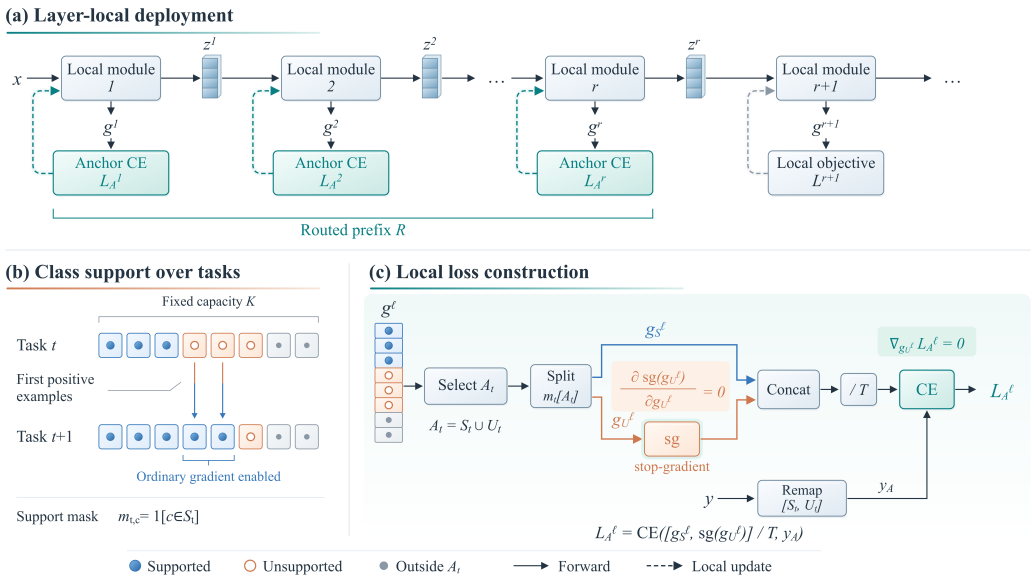

Figure 2. Deploying Anchor across layers and tasks. (a) Anchor modifies local classification losses in the routed prefix $\mathcal{R}$. (b) Class support expands when positive examples arrive, while output capacity K stays fixed. (c) Scores indexed by $A_t = S_t \cup U_t$ are split by the support mask. Stop-gradient blocks gradients through unsupported coordinates while retaining their forward values in cross-entropy. Targets y_A follow the concatenated order $[S_t, U_t]$. Solid arrows denote forward computation; dashed arrows denote local updates.

These identities describe coordinates in the local computation graph. With shared parameters θ, their Jacobians determine the parameter update:

$$\nabla_\theta \mathcal{L}_M = J_S^\top \nabla_{\mathbf{g}_S} \mathcal{L}_M + J_U^\top \nabla_{\mathbf{g}_U} \mathcal{L}_M, \qquad \nabla_\theta \mathcal{L}_A = J_S^\top \nabla_{\mathbf{g}_S} \mathcal{L}_M. \tag{6}$$

Here $J_S = \partial \mathbf{g}_S / \partial \theta$ and $J_U = \partial \mathbf{g}_U / \partial \theta$. Shared-parameter updates can still change unsupported values over training. Anchor removes their direct loss path while allowing the representation to evolve through supported learning.

3.2 Applying Anchor in Propagated Local Training

The intervention acts only on selected local losses. Supported coordinates remain live, whereas unsupported coordinates retain their forward values but are detached from the local loss graph. Algorithm 1 summarizes the training rule. Anchor modifies only the loss-side graph: unsupported values remain part of the propagated representation, while their direct local gradient path is removed. Later modules retain their original local objectives and gradient isolation.

The tensor tests exercise this rule in both implementations. Forward-loss error and maximum supported-gradient error are zero. Unsupported-gradient norms change from 0.06172 to zero in DeeperForward and from 0.07015 to zero in HCL-FF.

4 Experimental Design

4.1 Streams, Budgets, and Measurements

We study two recent representative FF methods, DeeperForward and HCL-FF (Sun et al., 2025; Yao et al., 2026), both of which propagate class-wise local responses into downstream learning. The primary experiments use CIFAR-100 streams with 10 or 20 classes per task (Krizhevsky & Hinton, 2009). Each class contributes 200 selected training-split images, including 20 reserved for validation. Evaluation uses 100 official test images per class. Training runs for 15 epochs per task with batch size 128. Replay stores uint8 images within 512 KiB. DeeperForward has 17 local layers and uses inference layers 0 through 16. The routed prefix contains eight layers in the main Anchor and long-stream configurations.

Algorithm 1 Anchor-Forward in propagated local training

Require: Minibatch (x, y); supported set S_t; unsupported set U_t; routed layers $\mathcal{R}$; temperature T

$A_t \leftarrow S_t \cup U_t$
$h \leftarrow x, \quad y_A \leftarrow \mathrm{remap}(y, A_t)$
for $\ell = 1, \ldots, L$ **do**
  $g^\ell \leftarrow f_\ell(h)$
  if $\ell \in \mathcal{R}$ **then**
    $\tilde{g}^\ell \leftarrow g^\ell$
    $\tilde{g}^\ell[U_t] \leftarrow \mathrm{sg}\big(g^\ell[U_t]\big)$
    $\mathcal{L}_\ell \leftarrow \mathrm{CE}\big(\tilde{g}^\ell[A_t]/T,\, y_A\big)$
  else
    $\mathcal{L}_\ell \leftarrow \mathrm{LocalObjective}(g^\ell, y)$
  end if
  $\theta_\ell \leftarrow \mathrm{LocalUpdate}(\theta_\ell, \mathcal{L}_\ell)$
  $h \leftarrow \mathrm{Propagate}(g^\ell)$
end for

Evaluation covers all observed classes without task identity. This follows class-incremental learning (Rebuffi et al., 2017; van de Ven & Tolias, 2019). Let $a_{i,j}$ denote accuracy on task j after task i. Arriving-task accuracy is $a_{i,i}$; old-task accuracy measures earlier tasks. For two equal-sized tasks, final class-IL accuracy is $(a_{2,1} + a_{2,2})/2$. The ten-task result averages arriving accuracy over tasks 2 through 10. We also report average incremental accuracy and final class-IL accuracy to show acquisition and retention together.

4.2 Matched interventions

Table 1 separates the dimensions of unsupported exposure. Paired branches share seed, class order, selected data, replay policy, optimizer schedule, and applicable task-prefix checkpoint. Capacity and inference remain fixed. Current-task interventions branch from a common task-1 state; pre-arrival experiments vary exposure before that state forms.

Cardinality and temperature curves use three paired seeds. Depth, two-task Anchor, canonical ten-task, HCL-FF, and GTSRB results use five seeds. The four-phase comparison uses three seeds; two added seeds extend current-task exposure. Two additional class orders each use two paired seeds. Tables report means and sample standard deviations where available. Paired effects are computed before aggregation. Statistical interval estimates are collected in Appendix H. The design separates exposure scale, location, timing, and the unsupported backward path.

Table 1. Interventions and their experimental roles. All contrasts retain the same capacity and data budget within their protocol.

Intervention	Variable	Comparison
Cardinality	Unsupported count	$M = 0, 30, 80$ for 2×10; $M = 0, 30, 60$ for 2×20.
Competition	Temperature	$T = 0.5$ vs. 2.0, each with its own M0 reference.
Depth	Exposure depth	Eight shallow vs. eight deep local heads.
Phase	Exposure timing	None, pre-arrival only, current-task only, or both.
Backward path	Unsupported gradient	Matched vs. Anchor with identical forward computation.

5 Empirical Analysis

5.1 Plasticity loss grows with unsupported-output exposure

Arriving-task accuracy decreases with unsupported count in both CIFAR-100 task granularities. In 2×10, arriving accuracy falls from $46.07 \pm 3.86\%$ at M0 to $32.00 \pm 2.95\%$ at M30. It reaches $12.63 \pm 5.43\%$ at M80. Every seed decreases across the three counts. The paired endpoint loss is 33.43 ± 2.07 percentage points. In 2×20, the corresponding means are 49.82%, 32.43%, and 18.92%. Figure 3 places these curves beside the temperature, depth, and phase interventions.

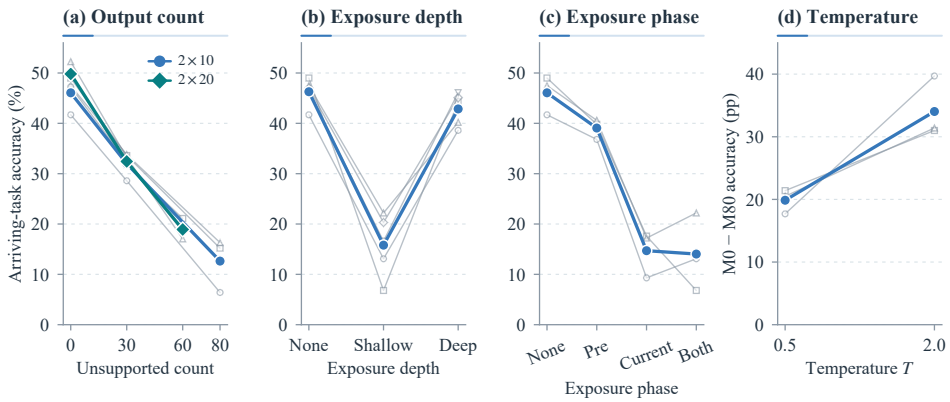

Figure 3. Four views of the plasticity tax in DeeperForward. (a) Arriving accuracy decreases with unsupported count in two task granularities. (b) Eight shallow exposed heads cause greater loss than eight deep heads. (c) The original three-seed phase comparison places the largest phase-specific loss during acquisition. (d) M80 damage is larger at $T = 2.0$ than at 0.5.

The acquisition loss exceeds the change in old-task accuracy. In the three-seed 2×10 comparison, M80 reduces arriving accuracy by 33.43 points and old accuracy by 10.97 points. This asymmetric change identifies new-class learning as the larger source of performance loss. The larger effect on arriving classes motivates the focus on acquisition plasticity.

Temperature changes the effect at a fixed unsupported count. Relative to same-temperature M0, M80 causes losses of 17.7, 21.4, and 20.5 points at $T = 0.5$. At $T = 2.0$, the losses are 39.7, 31.0, and 31.4 points. The paired increase averages 14.17 ± 6.81 points. Temperature changes all local softmax derivatives, so this contrast measures how competition settings moderate the count effect. Together, these results establish an acquisition cost that varies with unsupported exposure.

5.2 Shallow exposure during acquisition carries the largest cost

Equal-dose depth exposure places the main loss near the start of the network. Across five seeds, no exposure yields 46.28% arriving accuracy. Eight shallow exposed heads yield 15.80%, while eight deep heads yield 42.86%. The shallow-to-deep gap is 27.06 points. Both branches expose eight heads, so exposure depth distinguishes their effects. Earlier affected representations are consumed by more downstream local learners.

Exposure timing places the main cost during new-task acquisition. In the three-seed phase comparison, pre-arrival-only exposure reduces arriving accuracy by 7.00 points. Current-task-only exposure reduces it by 31.37 points, giving a 24.37-point difference. Two added seeds extend current-task exposure; its five-seed mean loss is 30.08 points. The factorial interaction changes sign across seeds. The dominant contrast places the cost during acquisition, when exposure overlaps with learning new classes. Appendix D.2 reports every row and the initialization used by the extensions.

5.3 Anchor isolates and removes the harmful update path

Removing the unsupported backward path recovers most of the acquisition loss while retaining the original forward competition. The five-seed M0 reference reaches $46.40 \pm 1.61\%$. At M30, matched competition achieves $33.90 \pm 1.72\%$, whereas Anchor reaches $45.74 \pm 2.13\%$. At M80, the corresponding accuracies are $17.18 \pm 2.99\%$ and $44.84 \pm 2.49\%$, yielding paired gains of 11.84 and 27.66 percentage points. Table 2 summarizes plasticity recovery; Figure 4 shows the corresponding restoration of representation geometry.

Anchor isolates the direct unsupported-coordinate update path: at a common model state, matched and Anchor share the same numerical responses, full softmax denominator, and supported-logit gradients, differing only in the direct gradient through unsupported coordinates (Eq. 5). Figure 4 further shows that matched competition alters the pairwise representation geometry relative to M0, whereas Anchor largely restores it. Because unsupported responses remain in the forward computation, this recovery

Table 2. Arriving-task plasticity with Anchor-Forward. Values are percentages, reported as mean ± sample SD over five seeds. Gain is the paired Anchor-minus-matched mean in percentage points. Each row uses its own matched protocol.

Setting	M0	Matched	Anchor	Gain
DeeperForward, M30	46.40 ± 1.61	33.90 ± 1.72	45.74 ± 2.13	**+11.84**
DeeperForward, M80	46.40 ± 1.61	17.18 ± 2.99	44.84 ± 2.49	**+27.66**
HCL-FF, M80	37.68 ± 1.95	22.88 ± 0.69	34.14 ± 3.40	**+11.26**
GTSRB, M10	48.73 ± 2.75	30.80 ± 3.38	48.13 ± 4.28	**+17.33**

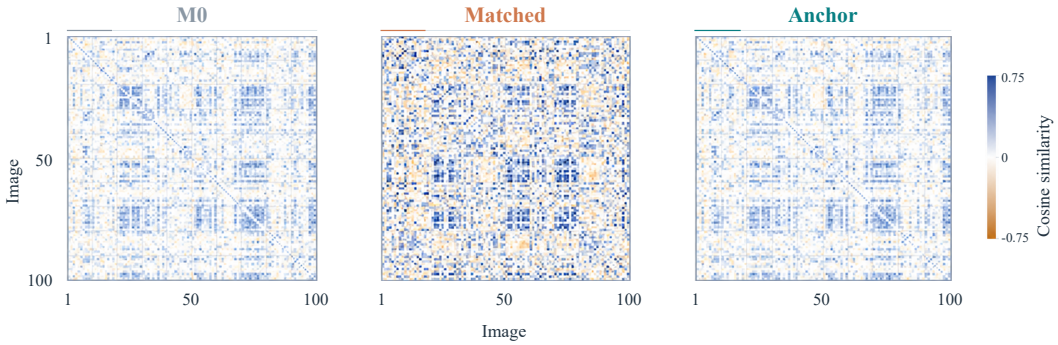

Figure 4. Anchor largely restores representation geometry. Pairwise cosine similarities are computed from the propagated representation after the eighth local layer for the same 100 class-balanced Task-2 test images from seed 3, with identical sample ordering across M0, Matched, and Anchor.

links the observed representation change and acquisition loss to unsupported-coordinate updates rather than forward participation alone.

5.4 Recovery persists over repeated class acquisition

The ten-task stream tests the intervention through repeated updates of the supported set. All remaining unsupported classes enter matched competition. For tasks 2 through 10, M decreases from 80 to zero. Five paired seeds use the protocol in Section 4. Figure 5 shows arriving accuracy and paired effects at each task.

Anchor recovers 16.13 points in mean arriving accuracy. M0 reaches $61.34 \pm 1.04\%$, matched competition reaches $42.43 \pm 0.78\%$, and Anchor reaches $58.56 \pm 0.74\%$. Every seed shows recovery. The gain exceeds five points at eight of nine task means. Average incremental accuracy rises from 17.16% to 22.46%, compared with 22.84% for M0. Final class-IL accuracies are 7.40%, 7.44%, and 7.89%, respectively. The task matrices in Appendix E show how acquisition and later retention contribute to these summaries.

The final task helps distinguish current support from training history. With $M = 0$, matched and Anchor use the same local gradient rule at a common state. Their trained states differ because earlier updates differed. Arriving accuracy is therefore 62.50% for matched and 63.58% for Anchor, with 66.80% for M0. The remaining 1.08-point Anchor gain reflects those trajectories. Task index and remaining count change together in this stream; the controlled count experiments provide the separate cardinality comparison.

Additional class orders preserve the direction of the path effect. Fixed permutations 101 and 202 each use two paired seeds. Mean Anchor gains are 19.06 and 19.46 points, respectively. Each paired matched effect is negative, and each Anchor gain is positive. Appendix E.3 gives the permutations, routing settings, and per-seed results.

5.5 Recovery generalizes across architectures and domains

HCL-FF tests the path in another recent strong FF baseline with a different propagated-local objective. Its evaluation uses the ensemble head, eight routed layers, and class supervision without a label hierarchy. Five paired seeds yield 37.68% arriving accuracy for M0, 22.88% for matched, and 34.14%

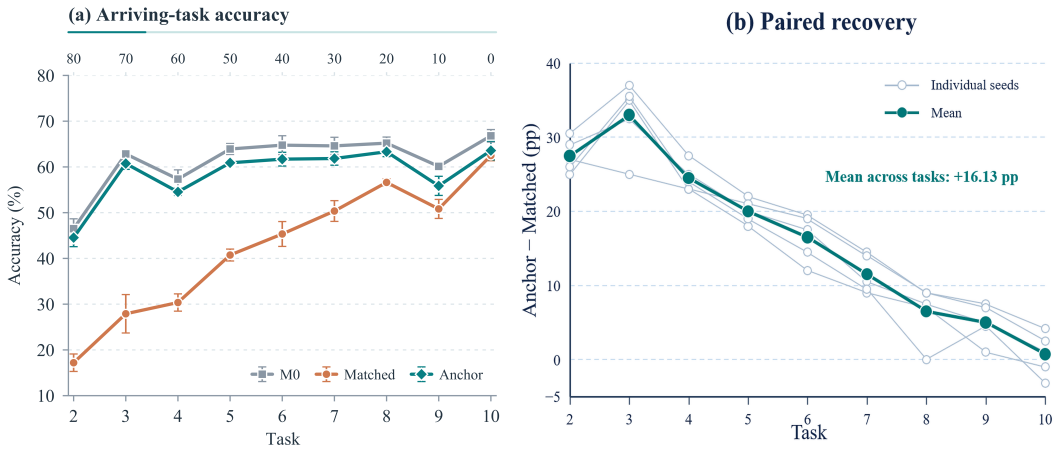

Figure 5. Plasticity recovery across the continual stream. (a) Mean arriving-task accuracy and sample SD for M0, matched, and Anchor on CIFAR-100 10×10. Upper ticks show the unsupported count M. (b) Mean paired Anchor-minus-matched recovery by task. The annotation averages tasks 2–10; recovery generally declines as the unsupported set shrinks.

for Anchor. Every seed improves, with a mean gain of 11.26 points. Final class-IL accuracy changes from 28.76% to 34.51%, compared with 36.64% for M0.

GTSRB tests domain transfer using traffic-sign images. The experiment uses a 20-class output scope and a 2×5 stream. Across five seeds, M0, M5, and M10 yield 48.73%, 37.47%, and 30.80% arriving accuracy. Anchor-M10 reaches 48.13%, recovering 17.33 points. Figure 6 shows the architecture and domain transfer results. Appendix F supplies the corresponding outcome details.

The same recovery pattern appears under another local objective and in another data domain. Together with the controlled interventions, these results connect exposure location and timing to the consequences of the unsupported backward path.

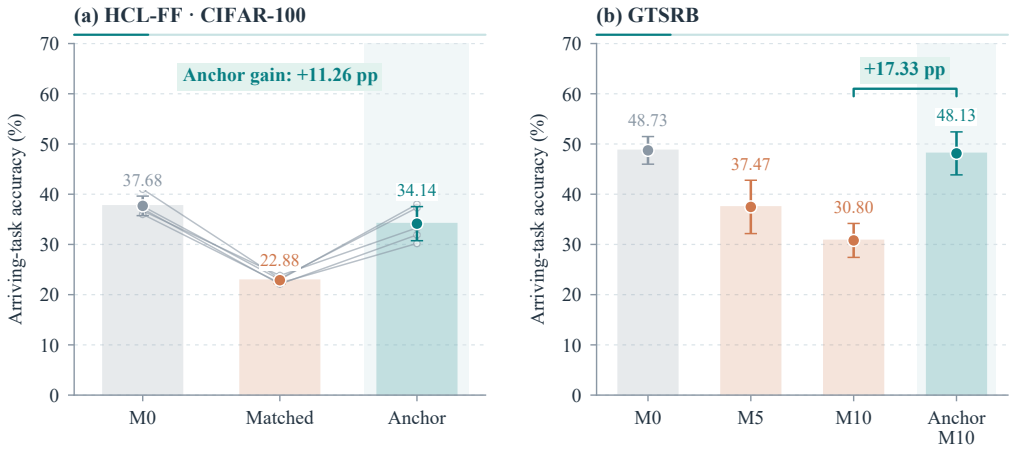

Figure 6. Recovery across an additional architecture and data domain. (a) HCL-FF on CIFAR-100 2×10 with $M = 80$. (b) DeeperForward on GTSRB 2×5 with a 20-class output scope and unsupported counts $M = 0, 5, 10$; Anchor acts at M10. Bars and error bars show mean and sample SD over five runs. Thin lines in (a) connect paired runs. Gains are paired Anchor-minus-matched means.

6 Output Support as Representation Design

6.1 Forward Support and Gradient Support are Distinct

A local learning rule chooses both forward participation and direct gradient paths. Forward support selects coordinates in the denominator. Gradient support selects coordinates receiving direct local

derivatives. Anchor separates these choices: the full denominator remains, while the unsupported direct gradient is zero.

6.2 Exposure depth determines downstream influence

A classifier's position changes the consequences of its updates. A terminal classifier supplies decisions; a propagated local classifier also supplies representations. Earlier affected representations reach more downstream modules. The shallow-versus-deep comparison in Section 5 shows why this exposure depth matters in DeeperForward.

6.3 Implications for propagated local learning

Support scope belongs in the definition of a propagated local learning rule. Its design includes forward support, gradient support, and exposure depth and timing. Anchor separates forward competition from direct unsupported updates; the phase result links timing to new-class acquisition. When classifier outputs are reused as representations, output support becomes part of representation design.

7 Related Work

Forward and local learning. Forward-Forward trains layers with local goodness objectives (Hinton, 2022). Channel-wise competition extends this approach to convolutional networks (Papachristodoulou et al., 2024). DeeperForward is a recent strong FF baseline for deep residual local training (Sun et al., 2025). HCL-FF is another recent strong FF baseline, developed around hierarchical and contrastive learning (Yao et al., 2026). Distance-Forward refines deep local representations through metric-based objectives (Xu et al., 2026). Our study focuses on output support when these locally supervised class-wise responses are propagated.

Intermediate supervision. Local-error and gradient-isolated learning develop alternatives to global error transport (Nøkland & Eidnes, 2019; Löwe et al., 2019). Auxiliary networks, dictionary contrastive learning, and gradient reconciliation improve local training (Ma et al., 2024; Choi et al., 2024; Yang et al., 2024). Auxiliary classifiers also improve continual-learning efficiency (Szatkowski et al., 2025). The present question concerns the class-wise coordinates reused as downstream inputs. Their support affects both local supervision and the representation available to later modules.

Continual classifiers and replay. Fixed classifiers and forward-compatible learning use future output structure during class-incremental learning (Pernici et al., 2020; Zhou et al., 2022). Classifier expansion, asymmetric competition, and logit adjustment modify the classifier as support changes (Liu et al., 2024; Caccia et al., 2022; Huang et al., 2024). SCR and LPR work on replay geometry and layer trajectories (Mai et al., 2021; Yoo et al., 2024). Dark experience replay retains output information from earlier data (Buzzega et al., 2020). SFFA brings contrastive Forward-Forward learning to continual tasks (Terres-Escudero et al., 2025). These approaches motivate separate acquisition and retention measurements, as discussed by Cha et al. (2025). Anchor studies a complementary operation: retaining unsupported forward competition while removing its local coordinate gradient.

8 Conclusion

When local classifier outputs are propagated as representations, unsupported outputs can impose a substantial plasticity tax on new-task learning. Our analysis shows that this cost is driven most strongly by shallow unsupported-coordinate updates during representation acquisition. Anchor-Forward isolates this path by preserving forward competition and supported-logit learning signals while removing direct updates through unsupported coordinates. Across DeeperForward, a ten-task continual stream, HCL-FF, and GTSRB, this intervention consistently recovers a large fraction of the lost arriving-task plasticity. These results reveal that the support scope of a local classifier affects more than its prediction space: it also shapes the representations available to downstream learners. When local classifiers become representations, output support becomes part of representation design.

Reproducibility statement. The appendices provide derivations, experimental settings, per-seed results, class orders, and statistical details. The accompanying archive contains the records and scripts needed to reproduce the main analyses.

AI use statement. Generative AI tools were used only for language polishing, formatting assistance, and limited code drafting. All research ideas, experimental design, results, analyses, and conclusions were developed and verified by the authors.

REFERENCES

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: A strong, simple baseline. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. In *International Conference on Learning Representations*, 2022.

Sungmin Cha, Jihwan Kwak, Dongsub Shim, Hyunwoo Kim, Moontae Lee, Honglak Lee, and Taesup Moon. Towards more diverse evaluation of class incremental learning: Representation learning perspective. In *Proceedings of the Conference on Lifelong Learning Agents*, volume 274 of *Proceedings of Machine Learning Research*, pp. 86–101, 2025.

Suhwan Choi, Myeongho Jeon, Yeonjung Hwang, Jeonglyul Oh, Sungjun Lim, Joonseok Lee, and Myungjoo Kang. Dictionary contrastive learning for efficient local supervision without auxiliary networks. In *International Conference on Learning Representations*, 2024.

Geoffrey Hinton. The Forward-Forward algorithm: Some preliminary investigations. *arXiv preprint arXiv:2212.13345*, 2022.

Zhehao Huang, Tao Li, Chenhe Yuan, Yingwen Wu, and Xiaolin Huang. Online continual learning via logit adjusted softmax. *Transactions on Machine Learning Research*, 2024.

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

Xinchen Liu, Hongbo Wang, Yingjian Tian, and Linyao Xie. Overcoming catastrophic forgetting with classifier expander. In *Proceedings of the Asian Conference on Machine Learning*, volume 222 of *Proceedings of Machine Learning Research*, pp. 803–817, 2024.

Sindy Löwe, Peter O'Connor, and Bastiaan Veeling. Putting an end to end-to-end: Gradient-isolated learning of representations. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Chenxiang Ma, Jibin Wu, Chenyang Si, and Kay Chen Tan. Scaling supervised local learning with augmented auxiliary networks. In *International Conference on Learning Representations*, 2024.

Zheda Mai, Ruiwen Li, Hyunwoo Kim, and Scott Sanner. Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 3589–3599, 2021.

Arild Nøkland and Lars Hiller Eidnes. Training neural networks with local error signals. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4839–4850, 2019.

Andreas Papachristodoulou, Christos Kyrkou, Stelios Timotheou, and Theocharis Theocharides. Convolutional channel-wise competitive learning for the Forward-Forward algorithm. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 14536–14544, 2024.

Federico Pernici, Matteo Bruni, Claudio Baecchi, Francesco Turchini, and Alberto Del Bimbo. Class-incremental learning with pre-allocated fixed classifiers. *arXiv preprint arXiv:2010.08657*, 2020.

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.

Liang Sun, Yang Zhang, Weizhao He, Jiajun Wen, Linlin Shen, and Weicheng Xie. DeeperForward: Enhanced Forward-Forward training for deeper and better performance. In *International Conference on Learning Representations*, 2025.

Filip Szatkowski, Yaoyue Zheng, Fei Yang, Tomasz Trzcinski, Bartlomiej Twardowski, and Joost van de Weijer. Improving continual learning performance and efficiency with auxiliary classifiers. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, 2025.

Erik B. Terres-Escudero, Javier Del Ser, and Pablo Garcia-Bringas. A contrastive symmetric Forward-Forward algorithm for continual learning tasks. In *Proceedings of the Conference on Lifelong Learning Agents*, volume 274 of *Proceedings of Machine Learning Research*, pp. 49–69, 2025.

Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Siyuan Xu, Yujie Wu, Jibin Wu, Lei Deng, Mingkun Xu, Qinghao Wen, and Guoqi Li. Advancing the Forward-Forward algorithm towards high-performance deep local learning. *Neural Networks*, 200:108765, 2026. doi: 10.1016/j.neunet.2026.108765.

Yibo Yang, Xiaojie Li, Motasem Alfarra, Hasan Abed Al Kader Hammoud, Adel Bibi, Philip Torr, and Bernard Ghanem. Towards interpretable deep local learning with successive gradient reconciliation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 56196–56215, 2024.

Jie-En Yao, Hong-En Chen, and C.-C. Jay Kuo. HCL-FF: Hierarchical and contrastive learning for Forward-Forward algorithm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27429–27438, 2026.

Jinsoo Yoo, Yunpeng Liu, Frank Wood, and Geoff Pleiss. Layerwise proximal replay: A proximal point method for online continual learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57199–57216, 2024.

Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, Liang Ma, Shiliang Pu, and De-Chuan Zhan. Forward compatible few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9046–9056, 2022.

A Theory and Exactness

The local derivation separates forward evaluation from automatic differentiation. Throughout this appendix, $T > 0$, $y \in S$, and $S \cap U = \varnothing$. All logits are finite. We write $\mathbf{g} = (\mathbf{g}_S, \mathbf{g}_U)$ and omit the stream index for clarity. The same argument applies independently at each exposed local loss.

A.1 Partition sums and loss difference

The full denominator can be factored into a supported term and an unsupported ratio. From Eq. 1,

$$\mathcal{L}_M = -g_y/T + \log(Z_S + Z_U) \tag{7}$$
$$= -g_y/T + \log Z_S + \log(1 + Z_U/Z_S) \tag{8}$$
$$= \mathcal{L}_S + \log(1 + Z_U/Z_S). \tag{9}$$

Thus, adding a nonempty set of unsupported coordinates changes the loss even before an update. Writing $q_U = Z_U/(Z_S + Z_U)$ gives a second form:

$$\mathcal{L}_M - \mathcal{L}_S = -\log(1 - q_U). \tag{10}$$

This expression describes a fixed state. Across training, the logits and their partition sums also change.

A.2 Derivatives of matched competition

For a supported logit g_k, differentiating the target term and log-partition gives

$$\frac{\partial \mathcal{L}_M}{\partial g_k} = -\frac{\mathbf{1}[k = y]}{T} + \frac{e^{g_k/T}}{T(Z_S + Z_U)} \tag{11}$$
$$= \frac{(1 - q_U)p_{S,k} - \mathbf{1}[k = y]}{T}. \tag{12}$$

For an unsupported logit g_u, the target term contributes zero because $y \in S$. Its derivative is

$$\frac{\partial \mathcal{L}_M}{\partial g_u} = \frac{e^{g_u/T}}{T(Z_S + Z_U)} = \frac{p_{M,u}}{T} > 0. \tag{13}$$

Summing these unsupported derivatives yields q_U/T. The aggregate unsupported-coordinate derivative is therefore q_U/T, coupling the local update pressure to both temperature and the realized score distribution.

A.3 The stop-gradient identities

Stop-gradient defines a forward value and a backward rule. In a computational graph, $\mathrm{sg}(z)$ returns z and sends zero derivative to its input. This is an automatic-differentiation operator. Applying it to $\mathbf{g}_U$ leaves the numerical partition sum unchanged:

$$\sum_{u \in U} e^{\mathrm{sg}(g_u)/T} = Z_U. \tag{14}$$

The forward equality immediately gives $\mathcal{L}_A = \mathcal{L}_M$. Supported logits enter through their original live graph, so their derivatives remain

$$\frac{\partial \mathcal{L}_A}{\partial g_k} = \frac{e^{g_k/T}}{T(Z_S + Z_U)} - \frac{\mathbf{1}[k = y]}{T}. \tag{15}$$

For unsupported logits, the backward chain rule instead gives

$$\frac{\partial \mathcal{L}_A}{\partial g_u} = \frac{\partial \mathcal{L}_A}{\partial\, \mathrm{sg}(g_u)} \frac{\partial\, \mathrm{sg}(g_u)}{\partial g_u} = \frac{p_{M,u}}{T} \cdot 0 = 0. \tag{16}$$

Proposition 1 (Exact local intervention)**.** *For a common input and model state, Anchor preserves the matched forward loss and all supported-logit derivatives. Its unsupported-logit derivatives are zero under the prescribed backward rule:*

$$\boxed{\mathcal{L}_A = \mathcal{L}_M, \qquad \nabla_{\mathbf{g}_S} \mathcal{L}_A = \nabla_{\mathbf{g}_S} \mathcal{L}_M, \qquad \nabla_{\mathbf{g}_U} \mathcal{L}_A = 0.} \tag{17}$$

The preceding derivative calculation proves the proposition. The identities also hold for a minibatch mean because averaging preserves equality. The same scaling applies to both branches.

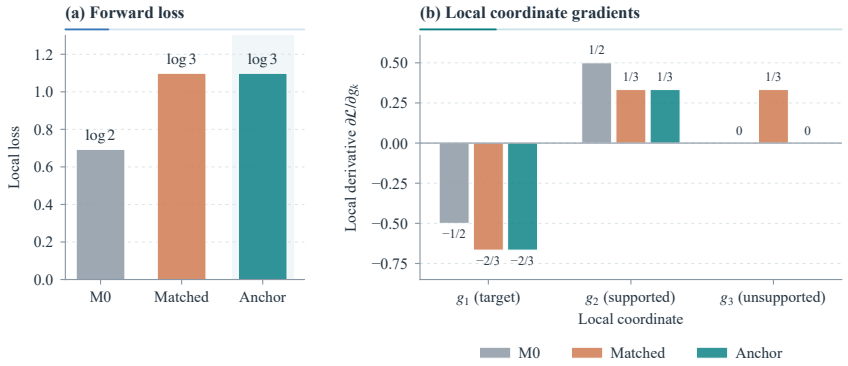

Figure S1. Why Anchor is not supported-only softmax. An analytical three-coordinate example uses $S = \{1, 2\}$, $U = \{3\}$, $y = 1$, $T = 1$, and zero logits. (a) Matched and Anchor have loss $\log 3$, while M0 has $\log 2$. (b) Anchor retains both matched supported derivatives and removes the unsupported derivative. M0 changes the supported derivatives as well. These are exact graph derivatives, not experimental measurements.

A.4 A DIRECT COMPARISON WITH SUPPORTED-ONLY SOFTMAX

Supported-only softmax changes the denominator itself. For $k \in S$, its derivative is $(p_{S,k} - \mathbf{1}[k = y])/T$. Subtracting it from the Anchor derivative gives

$$\frac{\partial \mathcal{L}_A}{\partial g_k} - \frac{\partial \mathcal{L}_S}{\partial g_k} = \frac{(1 - q_U)p_{S,k} - p_{S,k}}{T} = -\frac{q_U p_{S,k}}{T}. \tag{18}$$

For nonempty U, finite logits imply $Z_U > 0$, $q_U > 0$, and $p_{S,k} > 0$. Anchor therefore differs from supported-only training on every supported-logit derivative. Table 3 makes the three operations explicit.

Table 3. Three local learning rules at a common state. Unsupported values remain available to the network in all capacity-matched conditions.

Rule	Denominator	Supported derivative	Unsupported derivative
M0	Z_S	$(p_{S,k} - \mathbf{1}[k = y])/T$	0
Matched	$Z_S + Z_U$	$(p_{M,k} - \mathbf{1}[k = y])/T$	$p_{M,u}/T$
Anchor	$Z_S + Z_U$	$(p_{M,k} - \mathbf{1}[k = y])/T$	0

A three-coordinate example shows the distinction without parameterization. Let $S = \{1, 2\}$, $U = \{3\}$, $y = 1$, $T = 1$, and $\mathbf{g} = (0, 0, 0)$. Matched and Anchor both have forward loss $\log 3$, while M0 has $\log 2$. Their graph gradients are

$$\nabla_{\mathbf{g}} \mathcal{L}_M = (-2/3, 1/3, 1/3), \qquad \nabla_{\mathbf{g}} \mathcal{L}_A = (-2/3, 1/3, 0), \qquad \nabla_{\mathbf{g}} \mathcal{L}_S = (-1/2, 1/2, 0). \tag{19}$$

Anchor keeps the first two matched derivatives and removes the third. M0 changes the first two derivatives by changing forward competition. Figure S1 displays these values.

A.5 SHARED PARAMETERS AND LATER TRAINING STATES

The coordinate identity determines which contribution reaches a local parameter tensor. For $\mathbf{g} = f_\theta(h)$, the matched-to-Anchor parameter-gradient difference is

$$\nabla_\theta \mathcal{L}_M - \nabla_\theta \mathcal{L}_A = J_U^\top \nabla_{\mathbf{g}_U} \mathcal{L}_M. \tag{20}$$

The equality identifies the removed path for shared parameters. Supported learning can still change g_U through those parameters. Regularization and optimizer state can also affect the update. Anchor specifies which local loss contribution reaches the parameters.

For a plain SGD step from the same parameter state, the difference has a simple form. With learning rate η and no other update terms,

$$\theta_A^+ - \theta_M^+ = \eta J_U^\top \nabla_{\mathbf{g}_U} \mathcal{L}_M. \tag{21}$$

The two branches can therefore generate different representations after their updates. Later learners consume those numerical representations under the existing local-learning schedule. This is how a same-state backward intervention changes later acquisition.

When $U = \varnothing$, $Z_U = 0$ and $q_U = 0$. All three rules coincide at the same state. Stream branches can still differ after earlier exposure, as the final-task results demonstrate. The empty-set case links the exact local identities to the observed long-stream trajectories.

B Implementation and Tensor-Level Verification

B.1 Local loss construction

Algorithm 2 gives the minibatch computation at one routed classification loss. The ordered list $A_t = [S_t, U_t]$ places supported coordinates first and selected unsupported coordinates second; targets are remapped to this order. Supported scores remain on the live computation graph, while stop-gradient retains unsupported forward values and blocks their local backward path.

Algorithm 2 Anchor local classification update

Require: Minibatch scores g from local parameters θ; labels y; sets S_t, U_t; temperature T
1: $A_t \leftarrow [S_t, U_t], \qquad y_A \leftarrow \mathrm{remap}(y, A_t)$
2: $g_S \leftarrow g[:, S_t], \qquad g_U \leftarrow g[:, U_t]$
3: $\tilde{g}_A \leftarrow [g_S, \mathrm{sg}(g_U)]$
4: $\mathcal{L}_A \leftarrow \mathrm{CE}(\tilde{g}_A / T, y_A)$
5: $d \leftarrow \nabla_\theta \mathcal{L}_A$
6: $\theta \leftarrow \mathrm{OptimizerStep}(\theta, d)$

The selected coordinates define forward support, while class support determines which scores receive gradients. A Boolean support mask implements the same choice directly on the score tensor. Coordinates outside A_t contribute no loss term. Matched competition uses live scores on the same list; M0 includes supported coordinates alone.

The support mask follows the stream prefix. A class becomes supported when it begins providing positive examples. Its coordinate then receives the ordinary supported gradient. Routing selects which local losses use the intervention. Other local losses, replay handling, and inference retain their configured settings.

B.2 Local propagation and objective scope

The intervention is applied to the class-wise cross-entropy path. Numerical class-wise responses continue into later modules through the architecture's existing forward computation. A separate objective may contribute its own gradients. Thus, Eq. 5 refers to the intervened local objective and its score coordinates. This is also the scope exercised by the exactness scripts.

DeeperForward applies matched competition or Anchor to the first eight local layers during stream continuation and uses inference layers 0 through 16. HCL-FF likewise applies Anchor to eight routed layers and evaluates with the ensemble head. Its evaluated configuration uses class supervision without a label hierarchy.

Figure S2 shows how Anchor integrates with both backbones. DeeperForward trains each routed module with the Anchor classification loss. HCL-FF also uses its supervised contrastive objective on normalized features, giving $\mathcal{L}^{\ell}_{\mathrm{local}} = \mathcal{L}^{\ell}_{A} + \lambda \mathcal{L}^{\ell}_{\mathrm{SupCon}}$.

B.3 Same-state tensor checks

The two test scripts compare cloned score tensors before an optimizer step. DeeperForward uses a 7×100 tensor, ten supported coordinates, and 80 selected unsupported coordinates. Labels lie within the supported set. HCL-FF uses a 12×40 float64 tensor, ten supported coordinates, and 20 selected unsupported coordinates. Each test evaluates both losses and backpropagates through the score tensors.

At the same score tensor, both implementations reproduce the exact identities in Eq. 5: the forward-loss difference and the maximum supported-gradient difference are zero. We therefore omit these repeated zero-valued checks from Table 4 and report the nontrivial path quantity instead.

Matched competition produces a nonzero unsupported-coordinate gradient in both implementations, whereas Anchor removes this direct path exactly. Table 4 places this implementation-level check beside the corresponding full-training acquisition outcomes. The local path removal is accompanied by substantial recovery in both DeeperForward and HCL-FF.

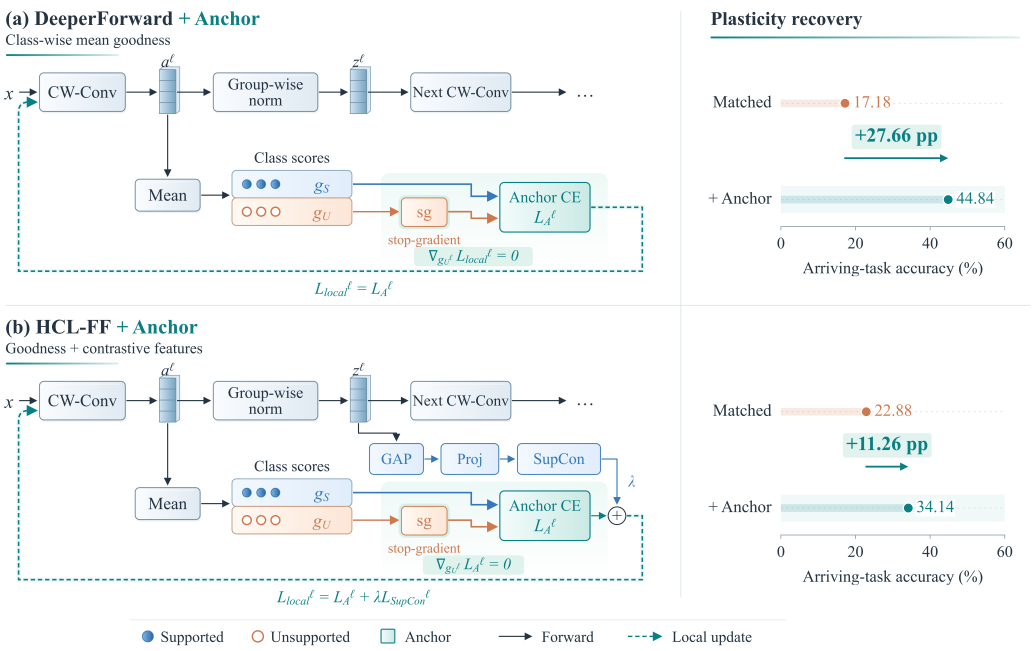

Figure S2. Integrating Anchor with DeeperForward and HCL-FF. (a) DeeperForward retains class-wise mean scores and group-wise feature normalization. (b) HCL-FF adds global average pooling, projection, and supervised contrastive learning on normalized features. Anchor blocks unsupported-score gradients in the local classification loss. On CIFAR-100 2×10 with $M = 80$, mean arriving-task accuracy rises from 17.18% to 44.84% for DeeperForward (+27.66 points), and from 22.88% to 34.14% for HCL-FF (+11.26 points). The diagrams show representative routed modules; HCL-FF uses class supervision without a label hierarchy.

Table 4. Implementation-level path verification and acquisition recovery. The unsupported-gradient column reports the same-state tensor check (Matched → Anchor). The remaining columns report five-seed CIFAR-100 2×10 arriving-task accuracy at $M = 80$. Forward-loss error and maximum supported-gradient error are exactly zero in both implementations and are therefore omitted.

Implementation	$\|\nabla_{g_U} \mathcal{L}\|_2$	M0	Matched	Anchor	Gain (pp)
DeeperForward	$0.06172 \rightarrow \mathbf{0}$	46.40 ± 1.61	17.18 ± 2.99	44.84 ± 2.49	**+27.66**
HCL-FF	$0.07015 \rightarrow \mathbf{0}$	37.68 ± 1.95	22.88 ± 0.69	34.14 ± 3.40	**+11.26**

B.4 Intervention invariants and branch divergence

The matched-to-Anchor comparison is exact at a common input and model state. The two rules share the same network parameters, active coordinates $A_t = S_t \cup U_t$, numerical class-wise scores, temperature, softmax denominator, scalar local loss, and supported-coordinate derivatives. The numerical response propagated to the next module is also unchanged for that forward evaluation. The only removed local contribution is the direct derivative from unsupported coordinates into the selected cross-entropy objective.

The full-training branches are paired under the protocol in Appendix C: seed, class order, selected data, replay policy, optimizer schedule, routed layers, and inference rule are held fixed within each comparison. After the first differing update, parameter states are allowed to diverge. This divergence is the intended consequence of removing the unsupported-coordinate path; the exactness claim does not require the two training trajectories to remain numerically identical after that update. The same-state tensor checks above verify the intervention before trajectory divergence occurs.

C Experimental Protocols and Metrics

C.1 CIFAR-100 data and replay

The primary protocol selects 200 training-split images per class and reserves 20 for validation. This leaves 180 selected images per class for optimization. Testing uses 100 official test images per class.

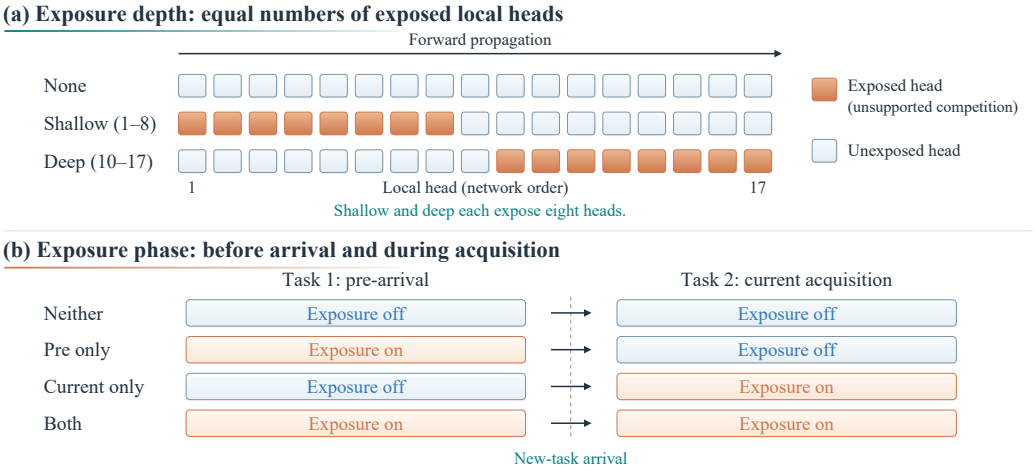

Figure S3. Where and when exposure is applied. (a) Depth compares heads 1–8 with heads 10–17 of the 17-head network. Copper heads use unsupported competition; each branch exposes eight heads. (b) The four-phase design crosses pre-arrival and current-task exposure. Each branch carries its task-1 state into task 2. Branches with the same pre-arrival setting share the applicable checkpoint. The four-phase comparison uses seeds 0–2; seeds 3 and 4 extend current-task exposure from a no-exposure task-1 checkpoint.

Two-task streams use either ten or twenty classes per task. The long stream uses ten tasks of ten classes.

Replay uses a fixed byte budget throughout the compared streams. The budget is 524,288 bytes, and the DeeperForward allocation is 523,600 bytes for 170 images. The capacity is shared across observed classes under the experiment's replay policy. Holding bytes fixed keeps memory use comparable as the supported set grows.

Table 5. Configuration for the full-budget DeeperForward and HCL-FF experiments.

Setting	DeeperForward	HCL-FF
Selected / validation / test per class	200 / 20 / 100	200 / 20 / 100
Epochs per task / batch size	15 / 128	15 / 128
Replay budget (bytes)	524,288	524,288
Initial / minimum learning rate	0.08 / 0.008	0.08 / 0.008
Weight decay	0.0005	0.0001
Dropout	0.2	0
Restart scheduler each task	Yes	Yes
Routed layers	First 8	First 8
Inference interval	0 to 16	0 to 16
Evaluation head	Goodness	Ensemble
Hierarchy configuration	Not applicable	None
Main extension seeds	3, 4, 5, 6, 7	3, 4, 5, 6, 7

C.2 Checkpoints and Exposure

Current-task path tests begin from a common task-1 checkpoint within each seed. M0, matched, and Anchor then follow their own updates. In the ten-task protocol, that checkpoint is continued under the full ten-task configuration. The class-order experiments use the full-stream checkpoint policy and the eight-layer routing prefix.

Depth exposure compares an eight-layer prefix with an eight-layer suffix. The number of exposed losses remains eight in both branches. Phase exposure distinguishes updates before the second task from updates during its acquisition. Appendix D.2 specifies the initialization for the four-phase comparison and current-task extensions. Figure S3 shows both interventions.

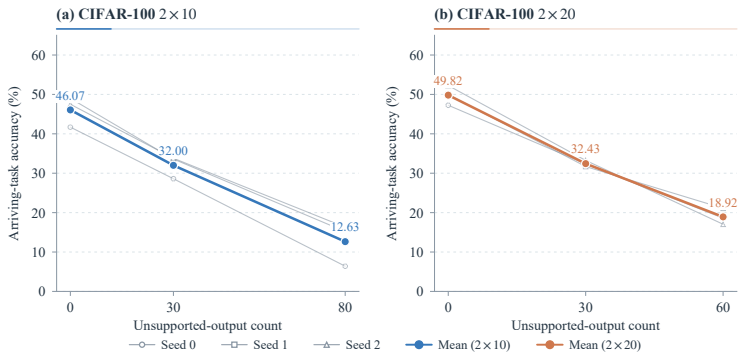

Figure S4. Unsupported-count responses within each seed. Arriving accuracy in CIFAR-100 (a) 2×10 and (b) 2×20. Thin lines show the three seeds; thick lines show their means. Each seed decreases as unsupported count increases.

C.3 Metrics and Aggregation

The task-accuracy matrix records both acquisition and retention. With N tasks of equal size, class-IL accuracy after task i is

$$A_i = \frac{1}{i}\sum_{j=1}^{i} a_{i,j}, \qquad \text{AIA} = \frac{1}{N}\sum_{i=1}^{N} A_i. \tag{22}$$

Final class-IL accuracy is A_N. The two-task old and arriving metrics are $a_{2,1}$ and $a_{2,2}$. Two-task forgetting is $a_{1,1} - a_{2,1}$. The long-stream arriving summary is

$$\overline{A}_{\text{arr}} = \frac{1}{N-1}\sum_{i=2}^{N} a_{i,i}. \tag{23}$$

It excludes task 1, which supplies the shared checkpoint in the main stream comparison.

Effects are calculated within each paired seed. Matched effect is matched minus M0; Anchor gain is Anchor minus matched. Positive gain means better acquisition with Anchor. Task-level effects compare matching task entries before averaging. The reported standard deviation is the sample standard deviation across seeds.

D Additional Controlled Interventions

D.1 Cardinality

The cardinality experiment changes the active unsupported count within a fixed-capacity model. The three-seed curves are reported separately from the five-seed Anchor experiment. They answer how acquisition changes as competition includes more unobserved classes. Tables 6 and 7 give the per-seed values. Figure S4 shows their within-seed trajectories.

Table 6. CIFAR-100 2×10 cardinality experiment. Entries are arriving accuracy in percent.

Seed	M0	M30	M80
0	41.70	28.60	6.40
1	49.00	33.60	15.20
2	47.50	33.80	16.30
Mean ± SD	46.07 ± 3.86	32.00 ± 2.95	12.63 ± 5.43

The fixed-subset control also changes unsupported identities at a fixed count of ten. Its outcome does not attribute the count effect to subset identity. Whole-training unsupported softmax mass gives inconsistent ordering between M30 and M80. These observations distinguish the controlled count response from a single scalar summary of pressure.

Table 7. CIFAR-100 2×20 cardinality experiment. Entries are arriving accuracy in percent.

Seed	M0	M30	M60
0	47.25	32.35	18.60
1	49.95	31.70	21.10
2	52.25	33.25	17.05
Mean ± SD	49.82 ± 2.50	32.43 ± 0.78	18.92 ± 2.04

D.2 Equal-dose depth exposure

The depth comparison holds the number of exposed local heads at eight. Table 8 shows no exposure, shallow exposure, and deep exposure. The five-seed shallow-to-deep gap averages −27.06 points. Deep exposure is 3.42 points below no exposure, while shallow exposure is 30.48 points below it.

Figure 3(b) summarizes the depth contrast, and Figure S5(a) shows the five paired runs. At the same exposure count, shallow exposure consistently reduces arriving-task accuracy more than deep exposure.

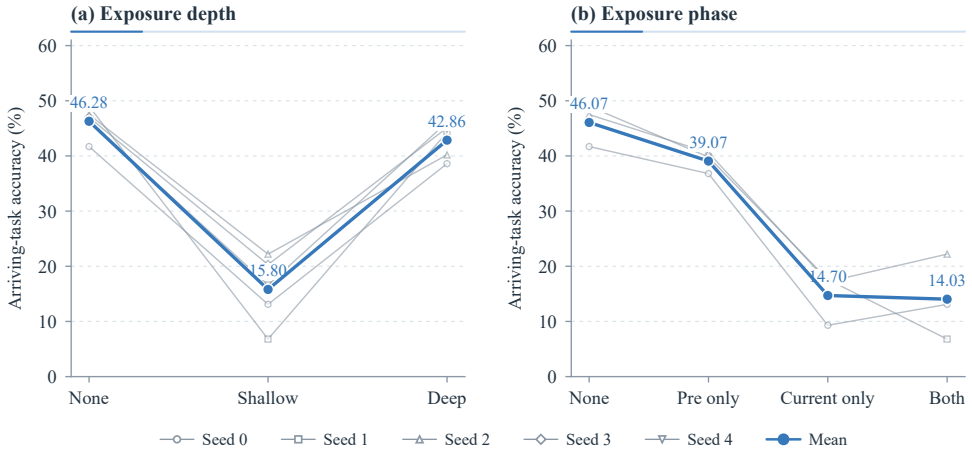

Figure S5. Seedwise depth and phase outcomes. (a) Five paired seeds compare no exposure, eight shallow heads, and eight deep heads. (b) The original three seeds compare neither, pre-arrival-only, current-task-only, and joint exposure. Thin lines connect results within a seed; larger markers show means. The additional current-task extensions remain separately identified in Table 9.

Table 8. Equal-dose depth exposure on CIFAR-100 2×10. Accuracy is in percent; the last column is a paired difference in percentage points.

Seed	None	Shallow 8	Deep 8	Shallow − deep
0	41.7	13.1	38.6	−25.5
1	49.0	6.8	44.3	−37.5
2	47.5	22.2	40.2	−18.0
3	47.0	20.3	45.1	−24.8
4	46.2	16.6	46.1	−29.5
Mean	46.28	15.80	42.86	−27.06

D.3 Phase exposure and branch initialization

The four-phase experiment crosses pre-arrival and current-task exposure. Table 9 gives the seedwise outcomes. Seeds 0 through 2 form the four-phase comparison. Their pre-only loss averages 7.00 points; current-only loss averages 31.37 points. The current-versus-pre difference is 24.37 points.

The current-task extensions resume seeds 3 and 4 from a task-1 checkpoint trained without unsupported-output exposure, then apply the selected phase setting to the resumed training. The pre-only branch therefore matches neither exposure, and both matches current-only in these rows. These added seeds extend the current-task comparison. The first three seeds provide the four-phase

Table 9. Phase outcomes on CIFAR-100 2×10. Accuracy is in percent. Seeds 3 and 4 resume a no-exposure task-1 checkpoint, as detailed below.

Seed	Neither	Pre only	Current only	Both	Current − pre
0	41.7	36.8	9.3	13.1	−27.5
1	49.0	39.8	17.6	6.8	−22.2
2	47.5	40.6	17.2	22.2	−23.4
3	47.0	47.0	20.3	20.3	−26.7
4	46.2	46.2	16.6	16.6	−29.6
Mean, seeds 0–2	46.07	39.07	14.70	14.03	−24.37
Mean, five rows	46.28	42.08	16.20	15.80	−25.88

comparison. The pooled five-row current-only loss is 30.08 points; the pooled pre-only loss is 4.20 points.

The factorial interaction varies across seeds. Using arriving accuracies A, define $I = A_{\text{both}} - A_{\text{pre}} - A_{\text{current}} + A_{\text{neither}}$. Its values for seeds 0 through 2 are 8.7, −1.6, and 11.9 points. The dominant phase result is the much larger loss during acquisition. The per-seed table keeps that comparison separate from the joint-exposure interaction. Figure S5 places the depth and original phase outcomes side by side.

E Full Continual-Stream Results

E.1 Task-wise acquisition results

The full stream follows ten groups of ten CIFAR-100 classes. Five paired seeds share their task-1 prefix and then continue under M0, matched, or Anchor. Unsupported counts follow the remaining future classes. Table 10 reports each arriving task from the run matrices.

Table 10. Arriving accuracy by task in the five-seed CIFAR-100 10×10 stream. Values are percentages; effects are percentage points.

Task	M	M0	Matched	Anchor	Effect	Gain
2	80	46.50±2.16	17.20±1.91	44.54±1.97	-29.30	+27.34
3	70	62.84±0.44	27.90±4.20	60.74±1.27	-34.94	+32.84
4	60	57.34±2.02	30.36±1.89	54.54±0.77	-26.98	+24.18
5	50	63.92±1.20	40.74±1.30	60.90±0.37	-23.18	+20.16
6	40	64.76±2.06	45.34±2.72	61.70±1.50	-19.42	+16.36
7	30	64.60±1.88	50.36±2.27	61.84±1.46	-14.24	+11.48
8	20	65.20±1.33	56.62±0.77	63.32±1.07	-8.58	+6.70
9	10	60.12±0.44	50.82±2.08	55.86±2.09	-9.30	+5.04
10	0	66.80±1.37	62.50±1.15	63.58±2.05	-4.30	+1.08

The largest gains occur while many unsupported outputs remain. Task 3 gives a 32.84-point Anchor gain, followed by 24.18 points at task 4. At task 9, ten unsupported outputs remain and the gain is 5.04 points. At task 10, the local rule coincides at a common state and the trajectory difference is 1.08 points. These entries describe the stream's progression alongside the separate controlled count study.

Each seed shows a positive mean arriving gain. The seedwise gains range from 14.70 to 17.21 points. Averaging these paired differences yields 16.13 points. AIA follows the same overall direction, as shown in Table 11.

E.2 Complete task-accuracy matrices

The task matrices separate immediate acquisition from later performance on the same classes. Their diagonal gives arriving accuracy. Entries below the diagonal show each task after subsequent learning. All entries use class-incremental evaluation over the classes observed at that stage. Figure S6 reports means across the five seeds.

The matrices show that most final-stage accuracy comes from the most recent task. The final class-IL means are 7.89% for M0, 7.40% for matched, and 7.44% for Anchor. In contrast, their mean

Table 11. Per-seed mean arriving accuracy and average incremental accuracy (AIA). All values are percentages except gain, in percentage points.

Seed	M0 arr.	Match arr.	Anchor arr.	Gain	M0 AIA	Match AIA	Anchor AIA
3	60.88	42.82	57.52	+14.70	22.71	17.09	22.13
4	60.99	43.62	59.26	+15.63	23.09	17.52	22.91
5	62.97	41.76	58.97	+17.21	23.24	16.69	22.59
6	60.23	41.93	59.00	+17.07	22.92	17.62	22.80
7	61.64	42.00	58.04	+16.04	22.25	16.89	21.87

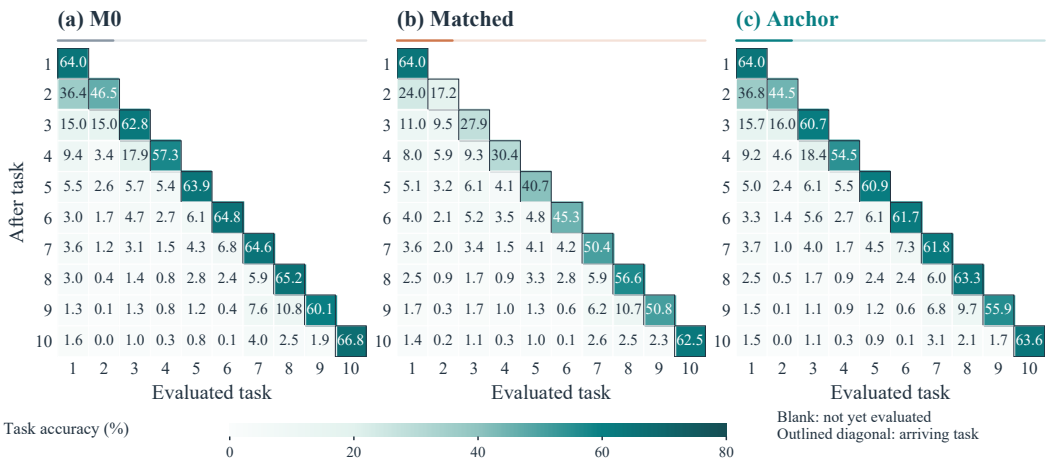

Figure S6. Acquisition and retention throughout the stream. Each matrix shows mean class-incremental accuracy across five seeds. Rows indicate the last trained task; columns indicate the evaluated task. Diagonal cells measure arrival, and lower cells track later performance on the same task. All panels use the same color scale. Unobserved tasks are blank; measured cells are labeled to one decimal place.

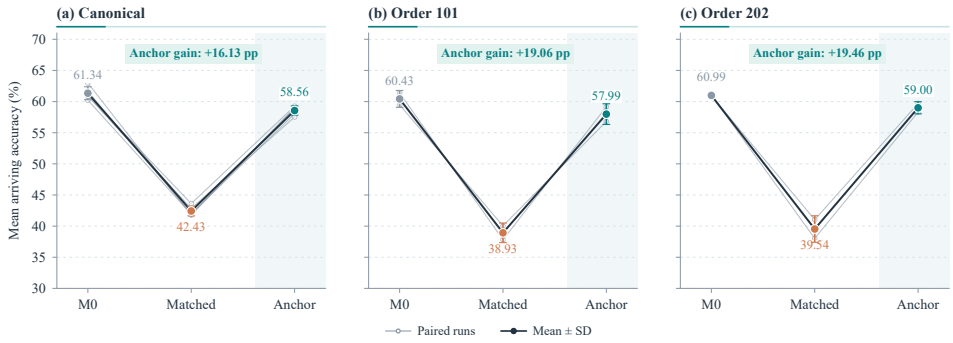

Figure S7. Paired acquisition outcomes under alternative class orders. The canonical order uses five paired seeds. Results are grouped by order. Individual points retain the paired repetitions; larger markers and error bars show means and sample SD. Alternative-order values come from the full-stream checkpoint runs. The permutations and exact outcomes are provided below.

arriving accuracies are 61.34%, 42.43%, and 58.56%. These are different measurements of the same trajectories.

E.3 Class-order experiments

Fixed alternative permutations test whether the path effect changes direction under another stream order. Orders 101 and 202 each use seeds 3 and 4. These runs use the full-stream checkpoint policy and an eight-layer routed prefix. The per-seed means appear in Table 12. Figure S7 compares the paired outcomes by order.

Both orders show lower acquisition under matched competition and higher acquisition with Anchor. Mean gains are 19.06 points for order 101 and 19.46 points for order 202. The canonical five-seed gain is 16.13 points. These fixed-order comparisons use the same task size and training budget.

Table 12. Mean arriving accuracy under fixed class orders. Canonical uses five seeds; orders 101 and 202 use two each.

Order	Seed	M0	Matched	Anchor	Gain
canonical	3	60.88	42.82	57.52	14.70
canonical	4	60.99	43.62	59.26	15.63
canonical	5	62.97	41.76	58.97	17.21
canonical	6	60.23	41.93	59.00	17.07
canonical	7	61.64	42.00	58.04	16.04
101	3	59.47	40.04	56.81	16.77
101	4	61.39	37.81	59.16	21.34
202	3	60.94	41.04	59.69	18.64
202	4	61.04	38.03	58.31	20.28

E.3.1 Class permutations

The sequences below list the canonical and alternative class orders used in the experiments. Consecutive groups of ten define tasks. The permutation identifiers refer to the order seeds, while model seeds control the paired repetitions.

Canonical order. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99.

Permutation 101. 31, 7, 40, 3, 48, 96, 67, 10, 65, 35, 66, 99, 89, 71, 76, 33, 87, 19, 72, 15, 18, 80, 79, 88, 58, 12, 49, 61, 82, 90, 5, 53, 2, 83, 16, 44, 37, 23, 1, 22, 94, 34, 13, 29, 0, 97, 57, 47, 78, 4, 92, 63, 75, 50, 25, 17, 70, 85, 21, 43, 39, 52, 73, 14, 86, 38, 95, 55, 8, 26, 54, 51, 30, 41, 60, 81, 46, 68, 93, 11, 20, 98, 32, 9, 56, 42, 91, 62, 36, 28, 77, 27, 64, 84, 6, 59, 45, 69, 24, 74.

Permutation 202. 26, 37, 44, 53, 68, 70, 10, 23, 91, 33, 9, 3, 28, 17, 43, 88, 46, 5, 48, 20, 99, 64, 76, 85, 36, 83, 72, 35, 1, 95, 65, 39, 29, 21, 18, 38, 42, 45, 19, 15, 27, 47, 75, 14, 8, 22, 90, 12, 13, 58, 0, 73, 77, 62, 6, 98, 40, 55, 11, 69, 71, 2, 61, 34, 81, 74, 87, 32, 31, 79, 92, 59, 89, 16, 96, 84, 51, 24, 66, 78, 80, 63, 30, 93, 41, 67, 4, 57, 56, 54, 7, 50, 86, 94, 25, 52, 60, 82, 49, 97.

F Architecture and Domain Transfer

F.1 HCL-FF per-seed outcomes

HCL-FF uses class-wise local responses in downstream learning. The tested configuration routes eight layers and evaluates with its ensemble head. It retains the CIFAR-100 2×10 data, replay, and epoch settings. Table 13 gives every paired seed for acquisition and final class-IL accuracy. Figure S8 shows both outcome profiles.

Table 13. HCL-FF per-seed arriving and final class-IL accuracy. Gain is Anchor minus matched (percentage points).

Seed	M0 arr.	Match arr.	Anchor arr.	Gain	M0 final	Match final	Anchor final
3	36.70	23.80	33.30	9.50	35.40	28.55	33.80
4	41.00	23.20	37.30	14.10	38.20	32.20	37.00
5	37.10	22.10	31.90	9.80	36.80	28.30	33.40
6	36.00	22.30	30.20	7.90	35.40	27.00	31.90
7	37.60	23.00	38.00	15.00	37.40	27.75	36.45

The arriving means are 37.68% for M0, 22.88% for matched, and 34.14% for Anchor. Final means are 36.64%, 28.76%, and 34.51%, respectively. The M0 final mean is obtained from the five per-seed entries in Table 13. Anchor gains range from 7.9 to 15.0 points in arriving accuracy.

The HCL-FF location screen gives a 4.5-point shallow-to-deep gap at seed 0. This is smaller than the DeeperForward depth effect. The five-seed path experiment still gives recovery in every seed.

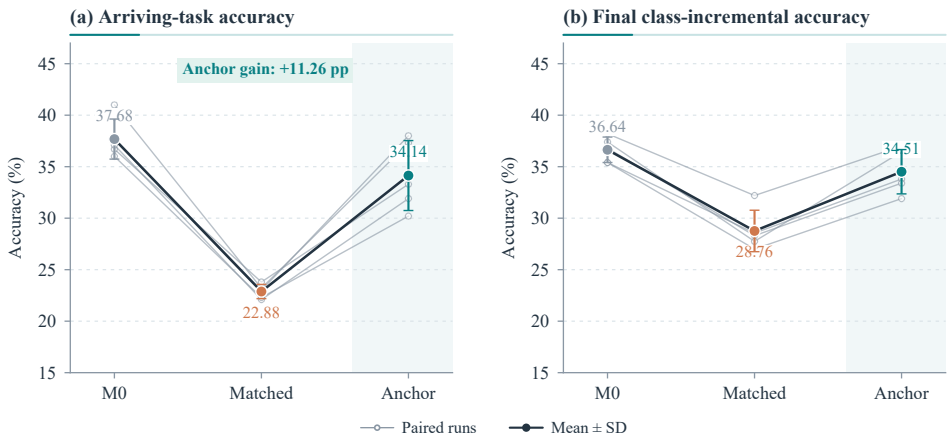

Figure S8. HCL-FF acquisition and final outcome profiles. Five seeds compare M0, matched competition, and Anchor on CIFAR-100 2×10. (a) Arriving-task accuracy. (b) Final class-incremental accuracy. Lines connect each seed across the three rules; larger markers and error bars show means and sample SD. Both panels use the ensemble head, eight routed layers, and class supervision without a label hierarchy.

The architecture comparison therefore separates the presence of the path effect from the size of depth amplification.

F.2 GTSRB OUTPUT-COUNT AND PATH EXPERIMENT

The GTSRB experiment uses a 20-class output scope and two tasks of five classes. The selected unsupported counts are zero, five, and ten. The cardinality curve decreases strictly in every one of the five seeds. Table 14 gives the aggregate values.

Table 14. GTSRB 2×5 arriving accuracy over five paired seeds. The output scope contains 20 classes.

Condition	Arriving accuracy (%)	Gain over M10 (pp)
M0	48.73 ± 2.75	
Matched M5	37.47 ± 5.31	
Matched M10	30.80 ± 3.38	
Anchor M10	48.13 ± 4.28	17.33 ± 6.05

The mean Anchor gain is 17.33 points. One seed has old-task accuracy 4.67 points below M0; the retention tolerance was four points. This outcome places the acquisition result beside its measured retention behavior. The aggregate experiment supplies the independent-domain comparison in the main text.

G ADDITIONAL OUTCOME PROFILES

G.1 PRESSURE SUMMARIES AND PARTIAL INTERVENTIONS

The pressure diagnostic measures a short early-training signal. In one CIFAR-100 2×20 cell, its pressure ratio was 0.0077, below the selected threshold. The later count curve was nevertheless strictly decreasing at 49.9%, 47.35%, and 40.75%. Whole-training mean unsupported mass also failed to consistently order M30 and M80. The count experiment and the mass summaries therefore provide different information.

Partial corrections changed the balance between acquisition and retention. Log-mean-exp background competition, deep routing, and weak support-calibrated variants did not meet the joint outcome criteria. Delaying weak deep competition until the last one, three, or five task-2 epochs gave no joint gain. Replay-only competition improved new-task accuracy by 4.1 points and final accuracy by 1.65 points. It also reduced old-task accuracy and increased forgetting.

A three-seed asymmetric interpolation gave a different outcome profile. Mean old accuracy changed by $+2.93$ points, arriving accuracy by -3.53, and final accuracy by -0.30. Forgetting changed by -4.70 points. A first-k versus last-k Anchor sweep and an unsupported-gradient multiplier sweep were non-monotone. These outcomes show how partial changes affect the same learning system.

G.2 BACKPROPAGATION REFERENCES

The BP references use 180 effective training images per class, 15 epochs per task, and at most 512 KiB replay. They provide acquisition and retention context at the same data and memory scale.

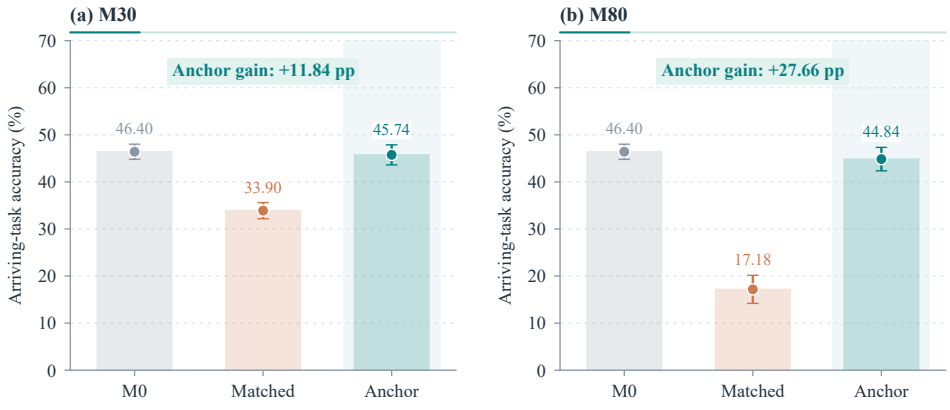

Figure S9. Removing the unsupported path restores two-task acquisition. Arriving-task accuracy of DeeperForward on CIFAR-100 2×10 with (a) 30 and (b) 80 unsupported outputs. Bars show means and sample SD over five runs.

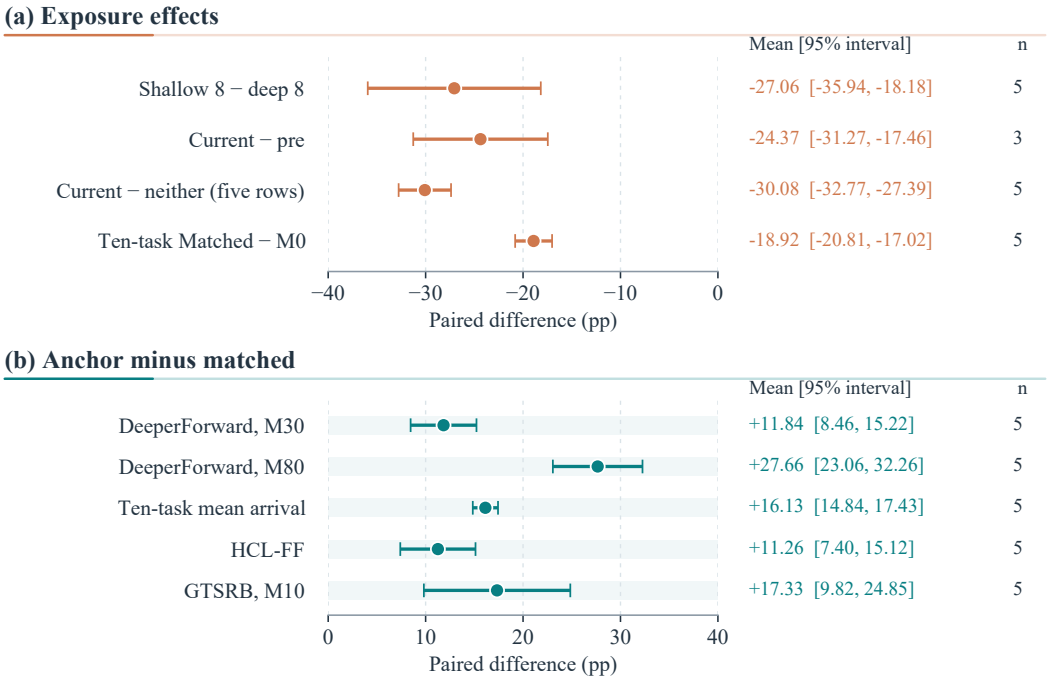

Figure S10. Paired effects across the evaluated protocols. Points show means and horizontal bars show two-sided 95% confidence intervals. (a) Exposure contrasts preserve the stated subtraction direction. (b) Positive values indicate higher arriving accuracy with Anchor. Replication counts refer to seeds. The ten-task estimate first averages tasks 2–10 within each seed. Current minus pre uses the original three-seed phase comparison; current minus neither also includes the two current-task extensions.

Their architecture, optimizer, and credit assignment differ from DeeperForward. Table 15 gives their outcome profiles.

Table 15. ResNet-18 replay references in the two-task setting. Values are mean ± sample SD, in percent.

Method	Old accuracy	Arriving accuracy	Final class-IL
ER	39.40 ± 6.62	30.67 ± 3.27	35.03 ± 3.59
DER++	31.10 ± 3.83	48.00 ± 3.65	39.55 ± 1.84

ER emphasizes old-task accuracy in this comparison, while DER++ reaches higher arriving accuracy. The matched conventional BP count control lacks the stable future-count trend reported for the local model. Together, these reference outcomes locate the unsupported-output effect within the studied learning designs.

H Statistical Analysis and Reproducibility

H.1 Paired effects and interval estimates

The experimental seed is the unit of replication for the paired summaries. For seedwise differences $d_1, \ldots, d_n$, let $\bar{d}$ be their mean and s_d their sample standard deviation. The reported two-sided 95% confidence interval is

$$\bar{d} \pm t_{0.975,n-1} \frac{s_d}{\sqrt{n}}. \tag{24}$$

For the ten-task summary, each seed first averages its differences over tasks 2 through 10. The interval then uses the five seedwise means. Individual tasks are not treated as independent replications.

Figure S10 collects the paired estimates.

Interval widths reflect seedwise variation in the paired effects. The three-seed cardinality and temperature studies are reported with their individual values in Appendix D. The two additional class orders are reported as two paired repetitions per order. Their rows remain separate from the canonical five-seed summary.

H.2 Outcome criteria

The two-task Anchor analysis uses margins of 3, 3, and 2 points for arriving, old, and final accuracy. The analysis reports that M30 and M80 satisfy the supported-reference criteria. The GTSRB retention tolerance was four points, and one seed exceeded it by 0.67 points. HCL-FF and the ten-task results are presented through their measured acquisition and final outcomes. These criteria define how the paired outcomes are assessed.

H.3 Memory and runtime

The five-seed long-stream runs have identical peak CUDA allocation in all three branches: 1,256,806,912 bytes. Parameter storage is 122,410,800 bytes, and replay occupies 523,600 bytes. Table 16 gives runtime and throughput for the continuation runs under this execution environment.

Table 16. Resource use in the five-seed ten-task experiment. Time and throughput are means across continuation runs.

Branch	Parameter bytes	Replay bytes	Peak CUDA bytes	Seconds	Images/s
M0	122,410,800	523,600	1,256,806,912	399.4	1220.9
Matched	122,410,800	523,600	1,256,806,912	462.6	1068.0
Anchor	122,410,800	523,600	1,256,806,912	447.5	1101.4

The BP ResNet-18 reference uses 44,880,528 parameter bytes. Its ER buffer stores 170 images in 523,600 bytes. DER++ stores 159 images and FP16 logits in 521,520 bytes. These quantities describe the reference configurations used in Table 15.

H.4 Environment and result records

Experiments were run on Windows 10 Pro with Python 3.11.7, PyTorch 2.5.1+cu121, torchvision 0.20.1+cu121, and NumPy 1.26.4. The machine used two NVIDIA RTX A6000 GPUs with driver version 571.96. Each run records its configuration, random seed, evaluation metrics, and resource measurements in JSON format. The accompanying experiment archive contains the run-level records and aggregate summaries used to produce the tables and figures in this paper.