arXiv:2607.25531v1 [cs.LG] 28 Jul 2026

# Multi-Scale Structural Features for Continual, Comprehensible Visual Recognition in a Developmental Learning Framework

**Zeki Doruk Erden**
Sabanci University, Istanbul, Turkey
doruk.erden@sabanciuniv.edu

**Abstract**

Contemporary machine learning struggles to learn continually, reuse prior knowledge, and expose a comprehensible internal structure. A recently proposed developmental, gradient-free learning framework addresses these limitations by learning a discrete, topological model of its inputs through local variation and selection, yielding an inherent continual-learning guarantee: new observations refine existing structure without overwriting past knowledge, and without replay buffers or predefined task boundaries. Its extension to visual inputs demonstrated this principle on shape recognition, but relied on a feature representation of limited expressivity that capped recognition accuracy. We introduce a new visual feature representation that encodes shape structure across multiple scales, capturing edge and contour features together with their spatial relations, and integrate it with the network-refinement learning process; we further improve the learning dynamics and the read-out used to predict from the learned model. The study targets two-dimensional shape, with class-incremental MNIST as a controlled, interpretable benchmark in which continual-learning behavior can be measured directly. Our approach substantially increases accuracy over the prior representation, matching or exceeding replay- and regularisation-based baselines at comparable storage while storing no past data, and preserves the framework's defining behavior: earlier-learned classes are retained as new ones are introduced, with no destructive adaptation, and the learned representations remain human-interpretable. What separates the methods is retention: the baselines surrender most of a just-trained class within its own cycle and relearn it afterwards, which ours does not. The significance lies in the manner of learning. The system integrates information one sample at a time while provably preserving its responses to past observations, without gradients, replay, or task boundaries — a qualitative capability outside the reach of statistical, gradient-based learning. These results show that this developmental modelling approach can be made markedly more accurate while retaining the properties that distinguish it from neural networks.

## 1 Introduction

Continual learning asks a system to learn from a non-stationary stream without overwriting what it already knows. Gradient-trained networks struggle by construction: past and present competence are carried by overlapping parameters in a shared weight space, and an update reducing error on the current data carries no constraint to preserve those supporting past competence (McCloskey and Cohen 1989; French 1999; van de Ven, Soures, and Kudithipudi 2025). We call this *destructive adaptation*.[1] The standard remedies—replay buffers, regularization anchors, task boundaries—mitigate the symptom while conceding the premise, each importing an assumption sequential experience does not offer: retained data, known boundaries, or a routing signal.

A different design avoids the problem at its root. A recently proposed *developmental*, gradient-free framework grows a discrete, topological model of its observations by local variation and selection, so integrating an observation adds or refines structure locally and—under a formal condition—leaves the responses to all past observations locally unchanged (Erden and Faltings 2026; Erden 2025). That method, termed the *Modeller*, integrates each observation exactly once, stores no data, needs no task boundaries, and yields models that are readily and inherently comprehensible—a *property set* statistical learning does not provide at any task scale. Its extension to vision reconceives an observation as a relational network and learns by refining it, but the representation it used was minimal and capped accuracy (Erden and Faltings 2026).

This paper removes that cap on the representational side while keeping the learner and its guarantee intact. Our central contribution is a *multi-scale structural feature representation*: an image becomes a single relational network describing the shape at *every* level of a coarsening hierarchy at once, so long-range and local relations are present together over shared nodes. This matters twice over. It gives refinement material at more than one granularity, so whichever scale recurs across a class's instances is retained while the rest is reduced away—structure accumulates rather than eroding. And it makes correspondence (matching) tractable: the atomic features are generic and recur many times within and across shapes, so the matcher needs relations constraining placement over the whole shape. We also improve the *read-out*: a local, gradient-free procedure consuming the spatial statistics the matcher already computes, which treats the *absence* of an expected unit as evidence. The construction rests on

[1] We prefer this to "catastrophic forgetting," which connotes an unintended, passive loss; the process is an *active overwriting* of past information by the update on the current data.

a general principle—*a continuous entity can be represented by its points of change, typed by the order and character of that change*—of which we build the simplest instantiation: first-order change points on a binarized contour, restricting the system to 2D shape. That restriction lies in the conversion from images to networks, *not* in the learner, which is indifferent to what its nodes and edges mean; we take it deliberately, so the learning machinery can be studied on inputs whose structure is available without first solving a second hard problem (Sec. 4).

On class-incremental MNIST the system learns all ten classes from a replay-free, boundary-free stream—in a stronger sense than "continual" usually carries, since learning proceeds strictly sample by sample with no batching—reaching $0.87$ held-out accuracy, matching or exceeding replay- and regularisation-based baselines at comparable storage while storing nothing, retaining what it learns where they do not, and producing models that can simply be drawn; well above the $\approx 0.50$ previously reported (Erden and Faltings 2026; Erden 2025). Our overall contribution is the development of the framework of Erden and Faltings (2026) for visual recognition; along that path we contribute the multi-scale representation the learner refines directly (Sec. 4), an explicit account of the matching step both learning and prediction rest on (Sec. 5), the local absence-aware read-out (Sec. 6), and an evaluation covering retention, neural baselines under their respective assumptions, a multi-scale ablation, model-size growth and readable models (Sec. 7).

## 2 Related Work

Describing visual structure across a hierarchy of scales is classical—scale-space filtering (Witkin 1983), the deep structure of images (Koenderink 1984), automatic scale selection (Lindeberg 1998), pyramids (Burt and Adelson 1983), wavelet multiresolution (Mallat 1989), and, closest to us, graph pyramids (Kropatsch 1995; Montanvert, Meer, and Rosenfeld 1991)—while local descriptors such as SIFT (Lowe 2004) and shape context (Belongie, Malik, and Puzicha 2002) reduce a shape to fixed-length vectors compared against stored exemplars. We inherit the principle but realize it structurally, accumulating all levels into a *single unified network* with shared nodes rather than a stack of separate descriptions. More fundamentally, across this work—and across the layered architectures from Marr (Marr 1982) through the Neocognitron (Fukushima 1980) and HMAX (Riesenhuber and Poggio 1999) to deep networks—the multi-scale representation is a *front end* handed to a *separate* learner; we know of no method in which it is integral to a *non-gradient, structural* learner that learns *by refining that very representation*.

Representing shape as attributed relational structure and recognizing by graph matching is likewise long-established (Messmer and Bunke 1998; Siddiqi et al. 1999; Biederman 1987; Felzenszwalb et al. 2010), with neural models more recently learning over graphs directly (Kipf and Welling 2017); here matching becomes a step *inside online learning*. Outside the gradient paradigm, Adaptive Resonance Theory (Carpenter and Grossberg 1987) treats stability–plasticity via a vigilance test, incremental concept formation (Fisher 1987) and growing self-organizing models (Fritzke 1995)—unlike the fixed-lattice self-organizing map (Kohonen 1982)—add units as learning proceeds, and Bayesian program learning (Lake, Salakhutdinov, and Tenenbaum 2015) shares our emphasis on structured concepts. Within class-incremental learning, replay (Rebuffi et al. 2017; Rolnick et al. 2019), distillation (Li and Hoiem 2018), regularization (Kirkpatrick et al. 2017) and expansion (Aljundi, Chakravarty, and Tuytelaars 2017; Erden, Gasmi, and Faltings 2025) each buy retention at the cost of an assumption online learning does not grant—stored data, task boundaries, or a routing signal (surveys: De Lange et al. 2022; Masana et al. 2023; van de Ven, Soures, and Kudithipudi 2025). The dominant recent direction, prompt-based continual learning over a frozen pretrained backbone (Wang et al. 2022b,a; Smith et al. 2023), leads current benchmarks but is unavailable here: it presupposes large-scale pretraining, supplying from the outset the representation whose *formation* is what we study. Appendix Q treats related work at more length.

## 3 A Brief Review of the Modeller

We give a compact, self-contained account of the learning methodology as instantiated here; it follows the design of Erden and Faltings (2026), to which we refer for the conceptual grounding, and Appendix B records where this instantiation differs.

**What "learning" means here.** Learning here differs fundamentally from the statistical process. There, learning extracts statistics from a *batch*—a gradient estimate followed by a small step, repeated over epochs. Here there is no batch and no gradient: the model sees *one sample at a time* and responds with discrete, local *structural* changes—creating a unit, removing part of one, spawning a more specific one—that encode what the observation revealed. Each sample is processed once and never revisited. This is not a batch procedure adapted to arrive one sample at a time; continual learning is not a setting it tolerates but the setting it is native to.

**State variables.** The model is built from *state variables* (SVs), whose state is *active* (1), *inactive* ($-1$) or *unobserved* (0). Classes are *target* SVs. Internal structure consists of *conditioning* SVs (csvs); each carries a *source* pattern extracted from the image, a single *target* (a target SV or another csv), and a signed *polarity*: a positive csv represents a configuration whose presence supports its target's activity, a negative one a configuration accounting for its inactivity. Each csv accumulates presence statistics ($n_{\text{active}}, n_{\text{inactive}}$), from which its *reliability* $P(\text{target} \mid \text{source present})$ and per-upstream $P(s \mid t)$ derive. Multiple csvs conditioning one another thereby form a hierarchical, multi-layer representation.

**Observations as networks, and refinement.** An observation is a *state network*: a directed graph whose typed nodes are feature instances and whose edges are relations, several relation types kept as parallel layers (a *state polynetwork*). When a target is active and no csv accounts for it, a new positive csv is created whose source is the observed network itself (*variation*). Learning then reduces each source

toward what recurs (*selection*): when the source matches a new observation in which the target recurs, source elements absent from that observation are removed; a match below the coverage threshold counts as *absence* rather than a reason to strip structure. This already exposes the operation everything depends on: source and observation are two *separate* networks, so the learner must establish *where* the source sits inside the observation—a correspondence (match) searched for at every learning and prediction step (Secs. 4 and 5).

The reduction carries the property the framework is built around. **Because refinement only ever *removes* requirements from a source and never adds any, an observation that satisfied a csv before a modification still satisfies it afterwards.** The unit's response to what it has already accounted for is preserved exactly—the local preservation guaranteed by Theorem 1 of Erden and Faltings (2026), from which system-wide continual learning follows emergently rather than via any mechanism added on top.

**Differentiation and structure formation.** Refinement alone would collapse each class to its least common denominator, so the methodology also grows structure: when a csv fully matches but its target contradicts it, or a partial match leaves an unexplained residual, *refinement upstreams* are spawned—new csvs conditioned on the existing one that carve out the residual (Algorithm 4, supplement; Figs. 1 and 2 of Erden and Faltings 2026 illustrate the refinement operation and upstream creation respectively), forming chains of progressively more specific csvs atop more general ones. Two maintenance operations use the statistics, mirror images acting on the two ends of one quantity—how often a conditioner is satisfied when its target occurs: *removal* discards one almost never satisfied, *reintegration* merges one almost always satisfied back into the downstream it conditions. Both are sampled per step rather than applied as hard tests (Appendix B states both bands exactly). A chain is evaluated coherently—a specific upstream assessed in the context of where its downstream matched—by the procedure of Sec. 5, the same at learning and prediction time. For simplicity each csv conditions a single target and negative evidence is externalized into separate negative csvs; an economy, not a design position (Appendix B).

## 4 Multi-Scale Structural Features

We now describe the representational contribution: the conversion of a shape image into the network the learner consumes. We take the figure as binary (thresholding at $0.5$) and describe the first-order instantiation for 2D shape; it is a proof of concept for a general principle, not a final descriptor.

**The principle: regions of constancy as points of change.** The representation embodies a single general principle: *regions of constancy are represented by their points of change*. An extended stretch continuing flatly collapses to the change points bounding it, and those points—typed by the order, direction and character of the change—become the nodes of a network. Nothing about this is image-specific: it applies to any signal representable as a continuous entity, and extends to higher orders of change where more detail is needed, the informativeness of each order diminishing rapidly. To sketch a function by hand one marks its extreme points and interpolates between them. The features are in this sense "designed," but designed *once*, from first principles, and universally applicable—a candidate primitive for arbitrarily complex visual structure rather than a task-tuned descriptor, *which is why we treat the change-point construction as part of the methodology itself rather than as engineering around it*. Two boundaries of the present instantiation are boundaries of the instantiation, not of the principle: (1) we read change points off the *contour* of a binarized figure, i.e. we apply the construction to a one-dimensional signal traced through the plane; the same construction applies directly to the un-thresholded image treated as a two-dimensional intensity *surface*, identifying its directional extrema and change lines instead of a curve's turning points—as one sketches a surface by its ridges and saddles; and (2) we use only the first order of change, where higher orders would enter as additional node types wherever finer discrimination is wanted. Both keep the principle intact and neither is trivial to implement well, so we defer them and develop the first-order, contour-based case here, which suffices to study the learning machinery.

**From contour to features.** The image is binarized and its contours extracted (outer boundaries and holes distinguished). Each contour is *traversed* and the points where the traversal direction changes sign along either image axis—the directional extrema—are recorded as nodes, typed by the axis and direction of the change and the local convexity (Fig. 1b); no smoothing is applied. These points form a sparse, largely style-invariant skeleton, capturing where the boundary turns while discarding how slowly or with what stroke width it does so. Nodes are connected by directed edges kept as parallel layers: *contour* edges linking successive change points, and *spatial* edges expressing relative position, split into horizontal and vertical variants whose direction encodes order along that axis. Every node also carries the *position* observed and every edge the *displacement* between its endpoints; these give matching something to check beyond type compatibility and the read-out its class-conditional evidence, and in a *learned* source are running estimates over the instances matched.

**Multi-scale coarsening into one network.** A single-scale network describes the shape at one granularity only, and fixing it in advance decides, before any learning happens, which distinctions the learner can make (Fig. 13, supplement). We therefore do not fix one. The network is coarsened iteratively—at each level the least-extent same-type maximum/minimum pair is removed, contour edges are created between the removed elements' surviving neighbours, and the spatial layers are recomputed—and the representation the learner receives is the *accumulation* of all levels into one augmented network, nodes shared across levels and the edges of every level coexisting as parallel structure (Fig. 1). What is outlined here is a brief form, given the space; the procedure in full is Algorithm 1 in Appendix A.

Relations of every extent thus coexist over the same nodes—long-range ones stating how distant parts of the

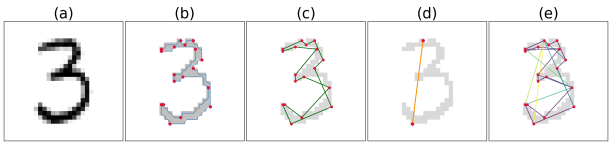


Figure 1: Construction of the multi-scale representation (actual pipeline). (a) Input. (b) Binarized contours with orientation-change points (red) that become nodes. (c) The finest contour network. (d) The coarsest level: one long-range relation summarizing the shape. (e) The unified augmented network: edges of *every* level coexist over the shared node set (color = level). Spatial layers omitted for legibility.

shape stand to each other ("this turning point precedes that concavity"), short-range ones stating local detail—with *no* priority attached to either: which a class comes to depend on is settled by learning, and the intuitive reading—longer relations carrying what generalizes, shorter ones what separates—is an *outcome* of refinement rather than an assumption of the method.

Carrying all levels at once has three consequences. *Refinement gains a choice of granularity*: relations failing to recur are reduced away and those recurring retained, so a mature csv is expressed at whatever extents proved reliable. *The algorithm becomes simpler*: when an element is refined away, longer-range edges spanning the gap are already present, so relational continuity survives without a repair step—which makes the base methodology's *rerelation* operation unnecessary here (Appendix B). And *matching becomes well-determined*, as shape-spanning relations constrain a placement local information cannot settle.

**Scope, and what is being claimed.** The learner is blind to the dimensionality and semantics of the network it refines—a network over 3D feature points would be processed identically—but the conversion from images to networks is not: obtaining a structured 3D description from photographs is a substantial problem in its own right, and the alternative of learning appearance across many stored 2D views is what this methodology exists to avoid. We take our motivation from a structural account of recognition with a long history in the study of human vision, though not a settled one, on which recognition rests on structural descriptions rather than accumulated 2D appearances (Marr and Nishihara 1978; Biederman 1987). We therefore restrict this work to intrinsically 2D shape, for which MNIST is canonical, and treat structured 3D feature extraction as the enabling step for future extensions (see also Erden and Faltings 2026). A harder benchmark would not test the properties at issue—whether each observation is integrated once, every past response preserved, and nothing stored are properties of the learning *process*—but rather the feature vocabulary, already the limiting factor on accuracy (Sec. 8). The representation is correspondingly minimal and is not offered as a competitive digit descriptor; the claim is the *abstract design*, that multi-scale visual structure and its cross-scale relations can be expressed as one relational network on which a structural continual learner operates directly. We do, however, defend the underlying principle more strongly than we can demonstrate it here: representing a continuous entity by its points of change is in our view capable of furnishing a *complete* feature description of shape to whatever fidelity is wanted, as further orders are admitted, and likewise for the surface formulation. We do not claim this as an experimental result, but state it as the position the representation is built on, and one we consider both defensible and testable (Appendix O).

## 5 Matching a Pattern to an Observation

Both learning (Sec. 3) and prediction (Sec. 6) rest on the same operation: locating where a learned pattern—a csv's source, or a chain of them—sits within an observed network. We describe it here, before either use, because it is the step at which the representation of Sec. 4 does its work.

**The problem.** The pattern (a learned internal representation) and the observation are separate networks of typed nodes. A candidate correspondence assigns pattern nodes to observation nodes of compatible type and must be consistent with the pattern's edges. Because the node types are generic—a given kind of orientation change occurs many times in one digit—type compatibility alone leaves many candidates, and short-range relations do not reduce them much: a small configuration of neighbouring change points typically recurs, in near-identical local geometry, at several places in the same shape. What distinguishes the correct placement from these near-duplicates is how the configuration sits relative to the rest of the shape (Fig. 11, supplement), which is what the long-range relations of the augmented network state. This is where the prior instantiation fell short: correspondence was decided by ranking candidates on positional proximity and judging them by how much structure survived the assignment, preserving the largest fit (Erden and Faltings 2026)—which, on a single-scale network, admits assignments locally plausible everywhere and globally wrong—visible in the $\approx 0.50$ ten-class accuracy reported there. Rather than searching harder for the largest fit, we make the observation carry relations whose agreement a correct placement must reproduce.

**The procedure.** Every node of a pattern carries a running estimate of its position and every edge one of its *displacement*, maintained online as means over the observations in which that element matched. The matcher grows a correspondence *incrementally* (Algorithm 2, Appendix A): it seeds a pairing of compatible type and extends to a neighbour only when the observed displacement agrees, within a spatial tolerance, with the learned one. Relations of every extent participate, so a placement is constrained by local detail and by relations reaching across the shape at once. The tolerance is swept strict to lenient and nothing is rejected outright on distance—agreement enters as a score falling off with disagreement, so the sweep *orders* the search rather than filtering it.

A match is *full* when the entire pattern is placed and *partial* otherwise. Matching is *chain-aware* (pinned). A *mid-chain* csv is one whose target is another csv rather than a class: its own source is the additional structure carved on

top of the configuration the downstream already represents (Sec. 3; in the base methodology such an upstream instead *contained* its downstream's source, whereas here the two are separate sources joined at *anchor* nodes, Appendix B). It is placed relative to where that downstream matched, so a chain is matched as one coherent object rather than piecewise—which is what lets deep chains discriminate on topology. At learning time the same procedure drives refinement and updates the surviving geometry toward the observed values.

## 6 Read-Out: Structure to Decision

Learning produces, per class, positive csv chains and negative suppressive structure. Classification requires a *read-out*: a procedure mapping a trained model and a new observation to a class. We keep it strictly separate from learning—it reads the structure and statistics and mutates nothing—so it operates on already-learned models and cannot affect the continual-learning guarantee.

The read-out consumes what matching (Sec. 5) makes available: which csvs are present, and, for those that are, where their nodes and edges were placed—resolved chain-wise rather than by matching each csv independently (Appendix C). The learned spatial statistics are the informative part here. Position is unreliable as a criterion for *deciding* a correspondence, but once a correspondence has been fixed on structural grounds, the positions and orientations at which a csv's elements landed are class-conditional evidence: a csv of class $k$ typically matches its own class at a characteristic place and orientation, and matches other classes elsewhere. Our read-out is built on this distinction, and on treating the *absence* of a csv as informative rather than as no evidence at all.

**The read-out mechanism.** Each csv $c$ accumulates only its own statistics, and only for its own class $k_c$: two firing counters, a streaming mean and variance of each matched node's *position*, and a circular mean of each matched edge's *orientation*, alongside a class-agnostic *pool* of the same geometry as a background. A csv's class is read off its immediate target, resolved one hop at a time. These accumulators are distinct from the matcher's own estimates, which keep one unconditioned mean per element to *decide* a correspondence, whereas these keep a mean *and* a spread to weigh one already decided. Each csv contributes to the score of $k_c$ alone; the only operation comparing classes is a final $\arg\max$, with no global classifier, learned weights or gradient. Let $P_c^o, P_c^f$ be $c$'s smoothed firing rates when its own class is present or absent. If $c$ is *absent* it still contributes $s_{k_c} \mathrel{+}= \log \frac{1-P_c^o}{1-P_c^f}$, so a csv that usually fires for its class but is silent here *demotes* that class. If *present* it contributes a firing term plus geometry terms comparing matched geometry to $c$'s own-class anchors versus the pooled background:

$$s_{k_c} \mathrel{+}= \log \frac{P_c^o}{P_c^f} + w_p \sum_n \big[\ell(p_n; a^o_{c,n}) - \ell(p_n; a^{\text{pool}}_{c,n})\big] + w_o\, \kappa \sum_e \big[\cos 2\Delta^o_{c,e} - \cos 2\Delta^{\text{pool}}_{c,e}\big], \quad (1)$$

with $\ell(\cdot; a)$ a diagonal-Gaussian log-likelihood (self-calibrating spread), $\Delta$ the circular deviation from the anchor mean, and $w_p, w_o, \kappa$ fixed. Every component is textbook; the contribution is that this information is present in the learned structure and that a read-out respecting the paradigm's locality recovers it with no global classifier, fitted weights or gradient step, and, being read-only, leaves the guarantee untouched. Appendix C gives the details and pseudocode (Algorithm 5); Appendix D reports a simpler read-out consulting *only* which csvs matched, ablating the geometric information.

## 7 Experiments

### Protocol

We use class-incremental MNIST, filtered for topological consistency (59,595 of 70,000 images; all methods see the identical subset; see the reasoning and exact criterion in Appendix V). A *cycle* presents the ten classes in fixed order, five samples each, and we run twenty cycles. Three properties define what is being tested: **exactly one online learning step per sample** (no batches), **no re-presentation and no replay buffer**, and **no task- or cycle-boundary signal**. None of this is difficulty imposed for its own sake; it is what a stream of experience looks like when it is *not* first collected into a dataset, and the setting in which learning from the world actually occurs. The comparison in Sec. 7 is therefore between methods operating under *different* assumptions, and we are explicit there about which each requires. Held-out evaluation uses twenty images per class at a disjoint offset, after each class block of every cycle; all Modeller results average ten seeds, and the *baseline* configuration is the full system with the local geometric read-out (Appendix V records the full configuration, Appendix M its sensitivity to the governing thresholds).

### Continual Learning of the Modeller

Once the Modeller has learned a class, the classes that follow do not destroy it. Figure 4 in the supplement shows this directly, and is worth consulting: it tracks every class's held-out accuracy along the whole stream, cycle band by cycle band, for the Modeller (top) against a plain network (bottom). Each class's accuracy rises when the class is introduced and then *holds* through every later class. The remaining dips are transient and self-correcting: a class typically falls to around 90% of its just-trained accuracy at its lowest point within a cycle and recovers by its end, an entirely different order from the baselines in the same figure, where a class drops to near zero the moment other classes train. Our dips are residual interference around a retained representation; the baseline's collapses are the representation itself being overwritten.

End-of-cycle accuracy climbs steeply over the first cycles ($0.244, 0.641, 0.709$ for cycles 0–2; the Modeller's trace in Fig. 2), plateaus by cycle 4–5 and holds with no downward trend, ending at **0.874** ($\pm 0.018$) against the $\approx 0.50$ previously reported (Erden and Faltings 2026; Erden 2025). Mean retention loss (per-class peak minus final) is **0.097**, most of it transient. We report per class block, since a block is where the task changes; the sample-by-sample view (Fig. 15, supplement) confirms nothing is concealed between them. Per-class accuracies lie between 0.80 and 0.96 for nine of ten classes

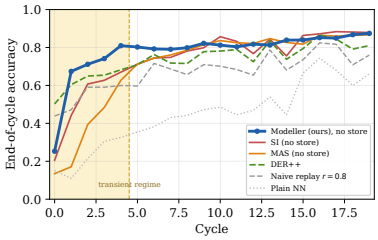


Figure 2: End-of-cycle accuracy over the stream (ten-seed means, identical stream). The shaded band marks cycles 0–4, where continual learning is under test: there a method either holds what it has learned or gives it up. Past it, repeated re-exposure turns the procedure into stochastic gradient descent with a slow time constant, and everything that learns at all converges. Per-class breakdown behind each curve: Figs. 4–7, 9, 10; remaining baselines and enlarged buffers: Appendices J, L, K.

(Table 11, supplement); the absence term levels them, since a csv that usually fires for its class but is silent here counts *against* it, demoting over-general csvs whose source also occurs *inside* other digits. Class 4 is the exception at 0.63, confused with 9—a representational residual.

## Comparison with Neural Baselines

We compare against representative neural continual-learning baselines on the identical stream (Table 1, Figs. 2 and 5), spanning the three families needing no visible task boundary. *Regularisation*, storing only per-parameter statistics: SI (Zenke, Poole, and Ganguli 2017), MAS (Aljundi et al. 2018) and LwF (Li and Hoiem 2018). *Replay*: a plain fully-connected network; the same with naive experience replay at $r \in \{0.2, 0.5, 0.8\}$; DER++ (Buzzega et al. 2020), which also stores the logits emitted when each example was inserted; and ER-ACE (Caccia et al. 2022). *Expansion*: a small convolutional network and an autoencoder-gated expert model (Erden, Gasmi, and Faltings 2025), which spawns a new expert when the reconstruction error of all existing experts exceeds a threshold and routes each test sample to one, each expert thereby seeing only the class its gate matches. Each is reported at its best configuration over a sweep of its own hyper-parameters—necessary, since the regularisation methods vary by orders of magnitude with penalty strength and are numerically inert at the values usually quoted (Appendix J).

**What each method assumes.** Every method receives the identical stream of *new* observations—five previously-unseen samples of the current class per block, 100 per class—differing in what they additionally need. Replay needs past data *stored and re-presented*: the ratio $r$ sets the buffer relative to a base size (10 here, of comparable scale to the five new samples per block the Modeller sees), so it holds 2, 5 or 8 past examples *at a time*, re-presented with every batch and turning over as the stream proceeds—far more past data is seen than the capacity suggests. DER++ and ER-ACE run at the same capacities. The regularisation methods store no

Table 1: Continual learning under the identical class-incremental stream (ten seeds, 20 held-out/class). *Final*: mean accuracy after cycle 19. *Early*: mean end-of-cycle accuracy over cycles 0–2. *WCR*: within-cycle retention (text), as a fraction of a class's accuracy just after its own block—$\mathrm{WCR}_{\mathrm{end}}$ at the close of that cycle, and $\mathrm{WCR}_{\mathrm{wst}}$ at its lowest point in between. *WCR* measures continual learning as such—how much of a class is still held after learning others—and separates the methods by a factor of two to four where *Final* does not. *Stored*: what is retained besides the model. Bold marks the best in each column. Baselines are at their best configuration over the sweeps in Appendix J, which also reports LwF and ER-ACE; buffers an order of magnitude larger, not comparable to ours, are in Appendix K. On *Final*, SI and MAS do not differ significantly from ours ($p$=1.00, $p$=0.24); every other difference is significant at $p < 0.01$ (Appendix H).

| Method | Final | Early | $\mathrm{WCR}_{\mathrm{end}}$ | $\mathrm{WCR}_{\mathrm{wst}}$ | Stored |
|---|---|---|---|---|---|
| **Modeller (ours)** | .874±.018 | **.536** | **.95** | **.90** | none |
| SI | **.879**±.016 | .418 | .44 | .36 | none |
| MAS | .867±.012 | .233 | .24 | .19 | none |
| DER++ | .810±.040 | .585 | .67 | .63 | buffer |
| Naive replay $r$=0.8 | .760±.069 | .500 | .52 | .40 | buffer |
| Plain NN | .661±.043 | .158 | .16 | .15 | none |
| Expert-AEC $k$=2 | .607±.055 | .198 | .75 | .72 | none |

data but each needs a *consolidation schedule*, which ours does not. None learns *sample-by-sample* either: each block is a batch of five revisited for 20 gradient updates (4,000 over the run, against our 1,000), and a batch supplies contrast between samples that a single observation does not. Rerun in our regime—one sample at a time, same gradient budget—every no-storage method loses accuracy (plain $0.663 \to 0.577$, MAS $0.836 \to 0.659$) and falls further below us on retention (Appendix J): the batch is an advantage they hold and we never had. We do not compare against methods requiring *visible task boundaries*, since on a genuine stream no such signal exists; the expert-routing model (Erden, Gasmi, and Faltings 2025) stands in for the expansion family, run *without* one—though it keeps a weaker assumption of the same family, that a task persists long enough for its expert to stabilize (Appendix P). On final accuracy the two strongest regularisation baselines are statistically indistinguishable from ours (SI $p$=1.00, MAS $p$=0.24); every other difference is significant at $p < 0.01$ (Wilcoxon signed-rank, Appendix H).

**Destructive adaptation in the baselines.** Under the identical stream the baselines contrast sharply (Fig. 2): a fully connected network loses nearly all information on a class as the following classes train, a convolutional network worse. Continual learning is most critical in those early cycles: in the long run the procedure becomes slow-timescale stochastic gradient descent under data abundance, where networks excel—why the plain network reaches 0.661 by cycle 19. We quantify that regime with *within-cycle retention* (WCR): the fraction of a class's just-trained accuracy still held at the

end of the same cycle ($\mathrm{WCR}_{\mathrm{end}}$) and at its lowest point in between ($\mathrm{WCR}_{\mathrm{wst}}$), over cycles 0–2 (Appendix I).

The Modeller retains $0.95$ and $0.90$, the gradient baselines far below—$0.52$ and $0.40$ at the strongest naive replay setting, $0.16$ and $0.15$ for the plain network—a difference of kind: their losses are the representation being overwritten, ours dips that recover within the cycle. The regularisation baselines make this sharpest: SI reaches $0.879$ and MAS $0.867$, neither distinguishable from our $0.874$, while retaining only $0.44$ and $0.24$ (Fig. 6, supplement)—equal accuracy by opposite routes, they relearning each class fast enough to recover by the run's end having given most of it up in between. Final accuracy alone cannot separate these. The one comparably stable baseline is the expert-routing model ($0.75$, $0.72$), which isolates what it has learned and pays for that in end performance.

**What the mitigations pay.** Replay buys its lead at the price of the assumption continual learning questions: a growing store re-trained at every step, and Fig. 5 shows the purchase—each stored example lifts the troughs, yet at every setting the per-class curves still collapse and recover, cycle after cycle: the oscillation is attenuated, not eliminated, while ours shows none of it and stores nothing. Relaxing the assumption further—a much larger buffer, or blocks supplying a class's full data—improves replay in proportion to what it represents, but those settings are not comparable to ours: with a buffer of $80$ each update is dominated by stored rather than arriving data, which is ordinary training on a growing pool rather than continual learning (Appendix K). The expert-routing model is the instructive opposite: isolating past experts brings it closer to our within-cycle stability than replay gets, but it plateaus far below because accuracy is capped by test-time routing rather than overwriting (Appendix L): isolation solves overwriting but breaks integration.

### Model Growth and Where the Discrimination Lives

Because the model *grows* its own structure rather than filling a fixed architecture, its size is an observable of learning rather than a hyperparameter—and unbounded growth is the obvious failure mode for a learner that adds structure in response to what it sees. It does not occur: growth is quick while classes are novel ($\sim 21$ csvs after cycle 0, $\sim 134$ after cycle 5) then saturates near $\sim 187$ from cycle $\sim 11$, fluctuating by 7.7 csvs between checkpoints, even though novel samples keep arriving and none is re-presented. Generation does not stop; what it produces is not *retained* once existing structure accounts for what arrives, so the population settles into an *equilibrium*, stable in size while continually regenerated—a *design property*, since a system that stopped proposing structure would be assuming nothing further will ever need representing. Negative structure comes to dominate ($105$ vs. $82$ positive at cycle $19$): boundaries, not prototypes, are where structure keeps being demanded (Appendices S, E, T). Scaling with class count and the work done per observation: Appendix U.

The read-out makes the split precise. Ignoring every csv with fewer than $M$ lifetime presence observations leaves accuracy flat ($0.8735$ at $M{=}0$, $0.8790$ at $M{=}10$, $0.8735$ at $M{=}40$; ten seeds) while at $M{=}40$ only $92$ of $187$ csvs remain: **half the learned population can be discarded at read-out time with no loss of accuracy.** Those below it are the recently formed and rarely satisfied, so the model is a mature core carrying essentially all the discrimination plus the provisional reserve above (Appendix R).

We ablate every component—the multi-scale representation, each supporting mechanism at read-out time, the geometric information the read-out consumes and the learning variants—each contributing to final performance to varying degrees (Appendices G, F, D, E).

### Comprehensibility of the Learned Structure

Because a csv is an explicit configuration of typed change points and relations it can simply be drawn, each mature one reading as a caricature of its class (Fig. 12, supplement); one can also read off which csvs condition which, and so diagnose *why* a class underperforms. Model and explanation are the same object.

## 8 Conclusion and Limitations

**Limitations.** The system reaches $0.874$—well above the $\approx 0.50$ this methodology previously attained on ten-class MNIST (Erden and Faltings 2026)—while learning in a manner statistical methods cannot. What stands between it and a conventionally trained network is chiefly the feature vocabulary: first-order change points on a binarized contour discard most of the image, and the one weak class, the $4/9$ confusion, is where it supplies no lasting distinction. The approach *presupposes* a representation rather than learning it, but as *a universal principle* applied once (Sec. 4) rather than per-task engineering. Two further limitations: each csv conditions a single target, inflating model size (Appendix U), and the read-out uses absolute node position, suiting centred MNIST. Extensions follow the same principle without altering the learner (Appendix N). The representation developed here is also a component of a wider programme, to be integrated with deliberative behaviour for embodied settings such as robotics (Erden and Faltings 2026; Erden 2025).

**Summary.** We extended a developmental, gradient-free structural learner to visual shape recognition by designing the representation it needed: a multi-scale structural encoding unifying all scales and their cross-scale relations in one network, with a local, absence-aware read-out. On class-incremental MNIST it learns all ten classes one sample at a time from a replay-free, boundary-free stream—a regime no baseline learns in, and under which they all lose accuracy—matches or exceeds them at comparable storage while storing nothing, and does not give up what it learns: continual learning in the sense meant, at an accuracy that makes it a genuine recognizer.

## References

Aljundi, R.; Babiloni, F.; Elhoseiny, M.; Rohrbach, M.; and Tuytelaars, T. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 139–154.

Aljundi, R.; Chakravarty, P.; and Tuytelaars, T. 2017. Expert gate: Lifelong learning with a network of experts. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Belongie, S.; Malik, J.; and Puzicha, J. 2002. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4): 509–522.

Biederman, I. 1987. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94(2): 115–147.

Burt, P. J.; and Adelson, E. H. 1983. The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4): 532–540.

Buzzega, P.; Boschini, M.; Porrello, A.; Abati, D.; and Calderara, S. 2020. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems*, volume 33, 15920–15930.

Caccia, L.; Aljundi, R.; Asadi, N.; Tuytelaars, T.; Pineau, J.; and Belilovsky, E. 2022. New insights on reducing abrupt representation change in online continual learning. In *International Conference on Learning Representations*.

Carpenter, G. A.; and Grossberg, S. 1987. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37(1): 54–115.

De Lange, M.; Aljundi, R.; Masana, M.; Parisot, S.; Jia, X.; Leonardis, A.; Slabaugh, G.; and Tuytelaars, T. 2022. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7): 3366–3385.

Erden, Z. D. 2025. *Foundations of a new learning paradigm in AI grounded in the principles of evolutionary developmental biology*. Ph.D. thesis, EPFL.

Erden, Z. D.; and Faltings, B. 2026. Foundations of a Developmental Design Paradigm for Integrated Continual Learning, Deliberative Behavior, and Comprehensibility. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 10(2): 1738–1752.

Erden, Z. D.; Gasmi, D.; and Faltings, B. 2025. Continual Reinforcement Learning via Autoencoder-Driven Task and New Environment Recognition. In *Adaptive and Learning Agents (ALA) and Autonomous Robots and Multirobot Systems (ARMS) Workshops at the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Felzenszwalb, P. F.; Girshick, R. B.; McAllester, D.; and Ramanan, D. 2010. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9): 1627–1645.

Fisher, D. H. 1987. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2(2): 139–172.

French, R. M. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4): 128–135.

Fritzke, B. 1995. A growing neural gas network learns topologies. In *Advances in Neural Information Processing Systems 7*, 625–632.

Fukushima, K. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4): 193–202.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; Hassabis, D.; Clopath, C.; Kumaran, D.; and Hadsell, R. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13): 3521–3526.

Koenderink, J. J. 1984. The structure of images. *Biological Cybernetics*, 50(5): 363–370.

Kohonen, T. 1982. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1): 59–69.

Kropatsch, W. G. 1995. Building irregular pyramids by dual-graph contraction. *IEE Proceedings—Vision, Image and Signal Processing*, 142(6): 366–374.

Lake, B. M.; Salakhutdinov, R.; and Tenenbaum, J. B. 2015. Human-level concept learning through probabilistic program induction. *Science*, 350(6266): 1332–1338.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Li, Z.; and Hoiem, D. 2018. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12): 2935–2947.

Lindeberg, T. 1998. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2): 79–116.

Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2): 91–110.

Mallat, S. G. 1989. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7): 674–693.

Marr, D. 1982. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman.

Marr, D.; and Nishihara, H. K. 1978. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 200(1140): 269–294.

Masana, M.; Liu, X.; Twardowski, B.; Menta, M.; Bagdanov, A. D.; and van de Weijer, J. 2023. Class-incremental learning: Survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5): 5513–5533.

McCloskey, M.; and Cohen, N. J. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, 109–165. Academic Press.

Messmer, B. T.; and Bunke, H. 1998. A new algorithm for error-tolerant subgraph isomorphism detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(5): 493–504.

Montanvert, A.; Meer, P.; and Rosenfeld, A. 1991. Hierarchical image analysis using irregular tessellations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4): 307–316.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Riesenhuber, M.; and Poggio, T. 1999. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11): 1019–1025.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*.

Sabour, S.; Frosst, N.; and Hinton, G. E. 2017. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, 3856–3866.

Siddiqi, K.; Shokoufandeh, A.; Dickinson, S. J.; and Zucker, S. W. 1999. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1): 13–32.

Smith, J. S.; Karlinsky, L.; Gutta, V.; Cascante-Bonilla, P.; Kim, D.; Arbelle, A.; Panda, R.; Feris, R.; and Kira, Z. 2023. 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 (CVPR)*, 11909–11919.

van de Ven, G. M.; Soures, N.; and Kudithipudi, D. 2025. Continual learning and catastrophic forgetting. In Wixted, J. T., ed., *Learning and Memory: A Comprehensive Reference*, volume 1, 153–168. Academic Press, third edition.

Wang, Z.; Zhang, Z.; Ebrahimi, S.; Sun, R.; Zhang, H.; Lee, C.-Y.; Ren, X.; Su, G.; Perot, V.; Dy, J.; and Pfister, T. 2022a. DualPrompt: Complementary prompting for rehearsal-free continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 631–648.

Wang, Z.; Zhang, Z.; Lee, C.-Y.; Zhang, H.; Sun, R.; Ren, X.; Su, G.; Perot, V.; Dy, J.; and Pfister, T. 2022b. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 139–149.

Witkin, A. P. 1983. Scale-space filtering. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1019–1022.

Zenke, F.; Poole, B.; and Ganguli, S. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, 3987–3995.

**Algorithm 1** Multi-scale structural representation of a shape

**Require:** binary image $I$

extract the contours of $I$, distinguishing outer boundaries from holes
$N \leftarrow$ directional change points of every contour, each typed by change axis, direction, and local convexity
$E \leftarrow$ contour edges between successive change points, plus spatial edges, in horizontal and vertical layers
$\mathcal{A} \leftarrow E$ {accumulator: the edges of every level}
**while** some pair of the kind below remains **do**
    among the nodes of each type, taken in contour order, collect every consecutive pair $(u, v)$ whose subtypes are one maximum and one minimum
    **if** there is no such pair **then** exit
    $(u, v) \leftarrow$ the pair of least extent—measured along $x$ for $x$-typed nodes, along $y$ for $y$-typed ones, and as the distance between them otherwise {an adjacent maximum and minimum lying close together is the smallest remaining excursion of the contour}
    remove *both* $u$ and $v$, reconnecting the surviving contour neighbours of each
    recompute the spatial layers at this level, giving edge set $E$
    $\mathcal{A} \leftarrow \mathcal{A} \cup E$
**end while**
**return** augmented network $(N, \mathcal{A})$: all nodes, and the edges of every level in parallel

**Algorithm 2** Matching a pattern to an observation

**Require:** pattern $P$ (a csv source or chain, with learned node positions and edge displacements); observation $O$; tolerances $\tau_1 < \cdots < \tau_m$, strict to lenient

**if** $P$ is mid-chain **then**
    pin $P$'s anchor nodes to where its downstream matched in $O$
**end if**
$\mathcal{M} \leftarrow$ a population of seed correspondences, each pairing a node of $P$ with a node of $O$ of compatible type, ordered by positional proximity
**for all** tolerances $\tau$ in increasing order **do**
    let every correspondence in $\mathcal{M}$ reconsider the extensions it previously declined {a looser $\tau$ may now admit them}
    **repeat**
        **for all** $M \in \mathcal{M}$, and unplaced nodes $n$ of $P$ adjacent to a node $M$ has placed **do**
            for each $O$-node of compatible type, score placing $n$ there by how closely its displacement from that neighbour agrees with the learned one, the score falling off over $\tau$
            admit each candidate with probability given by its score {nothing is rejected outright on distance}
        **end for**
        where admitted placements compete for the same $O$-node, fork $M$ into alternatives; cap the population, setting the weakest aside
    **until** a round admits nothing
**end for**
$M^\star \leftarrow$ the surviving correspondence of greatest coverage
$\mathrm{degree} \leftarrow$ the smaller of the fraction of $P$'s own nodes placed and the fraction of its own edges placed {anchors excluded: they are pinned, not searched for}
**return** $M^\star$, degree, and $\textsc{full} = [\mathrm{degree} = 1]$

# Supplementary Material

## A Coarsening and Matching Algorithms

The two procedures described in Sec. 4 and Sec. 5 are given here in full.

## B Differences from the Prior Instantiation

This appendix records, for readers familiar with the prior work (Erden and Faltings 2026; Erden 2025), the points at which this system differs from the mechanism as described there; it can be read in place of Sec. 3.

**Rerelation removed.** The prior algorithm (*network refinement with rerelation*) created bridging edges between predecessors and successors of removed elements. Here the algorithm is plain *network refinement*: removal of non-recurring nodes and edges only, identical to the prior algorithm but operating on node/edge lists rather than lists of state variables. Relational continuity across removals is guaranteed by the representation instead—the multi-scale augmented network already contains the coarser-scale edges spanning any removed fine structure (Sec. 4). The prior statistical absence-ratio criterion for noise tolerance is likewise replaced by coverage-thresholded matching (a poorly-matching source is treated as absent, so it is not stripped). The preservation guarantee is unaffected: removal-only refinement is strictly more conservative.

**Composed sources and anchors.** In the prior vision instantiation an upstream csv is a *subvariant* of its downstream: its source network *encompasses* the downstream's, so a chain is a sequence of ever-larger patterns, each containing the one below it. Here a chain is instead a *composition* of distinct sources. An upstream's source holds only the structure it itself contributes, plus *anchors*—nodes that are not owned by it but are references to nodes owned by its downstream—and the boundary edges that join the two. Ownership is explicit and single-valued: each node is owned by exactly one csv, and being an anchor is a role a node plays *in another source*, not a property of the node, so the same node is ordinary content for its owner and an anchor for the upstream that references it. The same construction is used whether the new structure comes from refining a partial match (the residual that was missing) or from generating a conditioner for the part of an observation the chain did not cover (the remainder); in both cases only the new structure is stored, anchored to what is already represented.

Three consequences follow. First, structure within a chain is stored once rather than restated at every level, so the csv

counts reported in Sec. 7 measure disjoint contributions rather than repeated copies. Second, matching is cheaper: matching a chain downstream-first binds an upstream's anchors as a side effect of its downstream's placement—no separate search, and no walk back down the chain, since an anchor is literally the downstream's node—so the work at each level is proportional to the new structure at that level. Anchors are consequently excluded from a csv's coverage, which therefore measures only its own content. Third, each csv is readable on its own as what that level adds, which is what Fig. 12 displays.

The arrangement does impose an invariant: every anchor of a csv must remain owned by some csv in its own downstream chain, since an anchor pointing outside that chain will never be bound and the upstream can then no longer match at all. Refinement, removal, absorption and reintegration all move or delete nodes, so each has to preserve it.

**Feature extraction.** The prior description extracted polygonal approximations of contours; here no polygonal approximation or smoothing is applied—contours are traversed directly and orientation-change points read off the raw contour, typed by axis, direction, and convexity. The multi-scale coarsening and the accumulation of all levels into one augmented network are new in this work.

**Single-target csvs and externalized negatives.** In the base methodology—the non-visual formulation—units condition multiple targets, sharing source subgroups across them, and negative evidence is carried as negative sources inside a unit. This implementation instead assigns each csv one target (duplicating sources across classes as needed) and represents negative evidence as separate negative csvs. It should be noted that this is not a departure introduced here: the visual formulation of the prior work already made both of these choices, so on this point the present system follows it rather than differing from it. Both remain implementation economies; the multi-target, shared-subgroup mechanism would require a substantial refactor orthogonal to this paper's contributions. The visible cost is redundancy—structure shared between classes is stored per class, inflating model size (Sec. 7) and forgoing cross-class reuse; we expect the multi-target mechanism to reduce the csv population substantially at equal competence.

**Statistics and maintenance.** Presence statistics, reliability and $P(s \mid t)$ follow the prior design, as does removal. *Reintegration* has no counterpart there and is introduced here; without it, upstreams that have ceased to carve any distinction accumulate as permanent structure. Both operations act on the same quantity, $P(\mathrm{SS}(C) \mid I(T))$, and are stated exactly here since the main text gives only their sense. In this expression $\mathrm{SS}(C)$ is the event that conditioner $C$'s sources are *satisfied*—its source configuration is matched in full by the observation—and $I(T)$ the event that its target $T$ is in the state $C$ conditions toward, that is, active for a positive conditioner and inactive for a negative one. So $P(\mathrm{SS}(C) \mid I(T))$ reads: given that the target came out the way this conditioner speaks for, how often was the conditioner in fact present. Writing $p = P(\mathrm{SS}(C) \mid I(T))$, removal applies on the low band $p < \tau_{\mathrm{rem}}$ and reintegration on the high band $p > \tau_{\mathrm{int}}$, each firing on a given step with probability

$$\rho_{\mathrm{rem}} \frac{\tau_{\mathrm{rem}} - p}{\tau_{\mathrm{rem}}} \qquad \text{and} \qquad \rho_{\mathrm{int}} \frac{p - \tau_{\mathrm{int}}}{1 - \tau_{\mathrm{int}}}$$

respectively, and zero outside its band. Each rises linearly from zero at its band edge to its rate at the extreme ($p = 0$, $p = 1$), which is what makes the two exact mirrors. By default the bands are placed symmetrically about the significance threshold, $\tau_{\mathrm{rem}} = \epsilon_{\mathrm{sign}}$ and $\tau_{\mathrm{int}} = 1 - \epsilon_{\mathrm{sign}}$. Two further conditions apply to reintegration only: the conditioner must be *positive*, since folding suppressive structure into what it suppresses would combine opposing evidence, and it must condition another csv rather than a class node, since a top-level csv has no pattern to be folded into. Neither operation acts on a csv on its birth step—a freshly spawned upstream inherits $p \approx 1$ and carries no independent evidence yet—and one whose relation gained no evidence on the current step has its probability down-weighted (Appendix V gives the values). Chain-aware anchored matching (Sec. 5) is used at both learning and prediction time. The strictly local geometric read-out (Sec. 6) is likewise new; a simpler presence-only read-out is reported alongside it in Appendix D.

**Learning flow.** Algorithm 3 summarizes one learning step: it composes variation (adopting an observed configuration as a new csv), refinement (statistical removal of non-recurring source structure), differentiation (spawning refinement upstreams on residuals), and the two statistics-driven maintenance operations (removal and reintegration). Matching throughout is chain-aware and anchored.

Algorithm 4 expands the two steps at the centre of that loop—the refinement of a csv's source and the spawning of the upstream that carries what refinement removed—since this is where anchors originate and where the composed-source arrangement above is actually established.

## C Read-Out Details

The read-out is strictly read-only: it inspects the learned structure, statistics, and matcher and computes a prediction, mutating nothing.

**Presence.** Presence is computed once per observation, by a single target-free pass over all csvs using the matching procedure of Sec. 5. Two quantities are recorded per csv: a coverage *degree* in $[0, 1]$—the fraction of its own nodes and edges placed, with edges counted so that the same nodes under a different connectivity are not treated as the same match—and whether the match is *full*. A csv whose source does not match, or whose downstream is not present, counts as absent.

The pass is ordered *downstream-first*, and this is what keeps its cost from scaling with the size of the model, so it is worth being explicit that no csv is matched independently of its chain. Because an upstream meets its downstream only at anchors (Appendix B), a csv whose downstream failed to place has nowhere for its anchors to be bound, and is settled as absent *without the matcher being invoked on it at all*. A chain is therefore descended only as far as the observation supports it, and one failure discharges the whole branch above: the

matching work is proportional to the structure the image actually evokes, not to the number of csvs the model holds. What does range over every csv is the scoring sum, since an absent csv still contributes its absence term—but that is a few arithmetic operations each, not a match. Discarding absent csvs rather than scoring them would therefore save nothing that matters and would yield a different, weaker read-out (Appendix D).

**The terms in full.** Two things are counted per csv, over its whole lifetime. Write $m_c^{\text{own}}$ and $m_c^{\text{tot}}$ for the number of steps at which $c$ existed in the model with its own class active, and at which it existed at all; and $n_c^{\text{own}}$, $n_c^{\text{tot}}$ for the subsets of those steps at which $c$ additionally *fired*, that is, matched the observation in full. Note that the condition is the activity of the class node the csv conditions toward—its own class in the sense of Sec. 6—and not the state of its immediate target, which for a mid-chain csv is another csv and may be inactive while the class is present. These counters are therefore separate from the presence statistics $n_{\text{active}}, n_{\text{inactive}}$ that the learner itself maintains (Sec. 3), which *are* defined relative to a csv's immediate target and which the maturity filter of Sec. 7 uses. The two firing rates of Sec. 6 are then Laplace-smoothed frequencies with $\alpha = \frac{1}{2}$ (Jeffreys),

$$P_c^o = \frac{n_c^{\text{own}} + \alpha}{m_c^{\text{own}} + 2\alpha}, \qquad P_c^f = \frac{(n_c^{\text{tot}} - n_c^{\text{own}}) + \alpha}{(m_c^{\text{tot}} - m_c^{\text{own}}) + 2\alpha},$$

clipped away from 0 and 1. Note that the foreign rate is an *aggregate* over all classes other than $c$'s own: no per-class breakdown is stored, and the smoothing keeps a csv that has never yet fired from contributing an unbounded term.

A position anchor is a streaming Welford accumulator $a = (n, \mu_x, \mu_y, M_{2x}, M_{2y})$ over the positions at which that source node has been placed, one such anchor per node for the csv's own class and one for the class-agnostic pool. Its per-axis spread is $\sigma = \sqrt{M_2/n + \varphi^2}$ with $\varphi = 1.5\,\text{px}$, and

$$\ell(p; a) = -\tfrac{1}{2}\Big[\Big(\tfrac{p_x-\mu_x}{\sigma_x}\Big)^2 + \Big(\tfrac{p_y-\mu_y}{\sigma_y}\Big)^2\Big] - \log(\sigma_x\sigma_y),$$

a diagonal-Gaussian log-density up to an additive constant that cancels in the difference $\ell(p; a^o) - \ell(p; a^{\text{pool}})$. The floor $\varphi$ is what makes the spread self-calibrating: without it a node placed almost identically in every match would have $\sigma \to 0$ and dominate the sum, so $\varphi$ sets the resolution below which positional agreement is not treated as informative.

Orientation is *axial*—a segment and its reverse are the same orientation—so an edge angle $\theta$ enters through its doubled angle. An orientation anchor accumulates $(n, \sum\cos 2\theta, \sum\sin 2\theta)$, giving a mean direction $\bar\theta = \frac{1}{2}\arctan 2(\sum\sin 2\theta, \sum\cos 2\theta)$, and the deviation of an observed $\theta$ from it is the circular distance $\Delta = \min\big(\delta, \pi - \delta\big)$ with $\delta = |\theta - \bar\theta| \bmod \pi$. The term $\kappa(\cos 2\Delta^o - \cos 2\Delta^{\text{pool}})$ is then the log-ratio of two von Mises densities of common concentration $\kappa$ up to their (identical) normalizers.

Both geometric terms are gated on having at least two observations in the anchor and in the pool, so a single placement never sets a class-conditional expectation. Their weights are fixed for all reported runs at $w_p = 3.0$ and $w_o = 1.0$ with $\kappa = 2.0$ (Appendix V).

**Polarity.** The read-out never inspects a csv's polarity. A suppressor nonetheless enters with the correct sign automatically: because it is a unit that fires when its own class is *absent*, its own-class firing rate falls below its foreign rate, $P_c^o < P_c^f$, and its present-term $\log(P_c^o/P_c^f)$ is therefore negative—a veto—while its absent-term is positive. No separate handling of negative structure is required.

**Matching policy.** Anchored (pinned) matching can be replaced by *unified* matching, in which each csv's whole chain is matched fresh as one pattern; on the final models it scores slightly below anchored. Anchored matching is used throughout.

# D Ablating the Geometry: a Presence-Only Read-Out

This appendix reports a second, deliberately impoverished read-out. It is best read as an *ablation of the geometric information* rather than as a competing method: it is given the same trained models and differs from the read-out of Sec. 6 in exactly two respects—it consults only *which* csvs matched, using none of the learned positions or orientations, and it treats a non-matching csv as no evidence rather than as evidence against its class. Its lower accuracy is therefore the point of reporting it: the gap measures what those two sources of information contribute, on identical structure. Each class is scored through its positive top-level csvs; a present csv contributes its reliability $\rho$ and present upstream evidence is fused recursively (positive as support $p_i$, negative as suppression $q_j$) by summation in log-odds, with the reliability term pivoted at the uniform prior $1/K$ over $K$ classes so that a csv aligned with its class only at the base rate contributes nothing:

$$\ell = \text{logit}(\rho) - \text{logit}(1/K) + \textstyle\sum_i \text{logit}(p_i) - \sum_j \text{logit}(q_j). \tag{2}$$

Both read-outs are read-only and differ only in how they score a fixed trained model, so neither requires retraining. On the same final models this scheme reaches $0.697$, against $\mathbf{0.874}$ for the geometric read-out. Variations within it change little: ignoring suppressive structure gives $0.635$, a maturity filter that discards csvs with fewer than $M$ lifetime observations moves it within one standard deviation ($0.697/0.700/0.704$ at $M$=$0/5/10$), and two further variants—a continuous *graded* coverage scheme and a *precedence* scheme—are dominated by the log-odds form. The maturity filter behaves the same way under the geometric read-out, and we report that measurement, which is the one the main text draws on, in Table 13. The spread among all of these is small compared with the difference to the geometric read-out, which indicates that what matters is not the particular decision rule but whether the learned spatial statistics and the informativeness of absence are consulted at all.

# E Learning Variants

We report learning-side variants around the baseline; the baseline (with suppressors, geometric read-out) remains the main version. **Suppressors: dispensable here, but formed by design.** Preventing suppressor formation does *not* hurt

Table 2: Learning variants (ten-seed mean). Baseline is the main version.

| Variant | Accuracy | csvs | Retention loss |
|---|---|---|---|
| **Baseline** | **0.8735** | 187 | 0.097 |
| No suppressors | 0.8805 | 58 | 0.095 |
| Pos-gate | 0.8645 | 186 | 0.108 |
| Pos-skip | 0.8650 | 178 | 0.107 |

on MNIST—it is marginally higher, $0.8805$ vs. $0.8735$, a difference that is not significant over the ten paired seeds ($p = 0.64$), at roughly a third of the model size (58 vs. 187 csvs)—because the discrimination MNIST needs is carried by the positive, geometry-bearing structure. This does not mean suppressors should be removed. The learner does not optimize our accuracy metric: it has no test set, receives no error signal, and never stops changing on the basis of accuracy, by deliberate design. It forms a complete-but-minimal model of what it has seen, and suppressors are part of that completeness; it keeps forming them because they may be discriminative for other data, even where—as on MNIST—they are not needed. **Position-gated matching.** Two variants gate matching or suppressor formation on the match centroid position; neither improves on the baseline ($0.8645$ and $0.8650$; $p = 0.39$ and $p = 0.34$, both non-significant), and a per-node position variant did not either—consistent with Sec. 4: local position is not the lever for discrimination; global structure is.

## F Ablations of Supporting Mechanisms

**Suppressive structure.** The role of suppressors depends on the read-out. Under the *presence-only* read-out, re-scoring the final models with all negative csvs ignored lowers accuracy ($0.697 \rightarrow 0.635$), and the effect is strongly class-specific: some classes depend on suppression to hold their boundary (most sharply $d8$, $0.77 \rightarrow 0.23$ when suppressors are removed), while others are over-suppressed and *improve* without them (most sharply $d2$, $0.57 \rightarrow 0.74$). Under the *geometric* read-out the picture inverts: preventing suppressor formation entirely (Appendix E) is net neutral-to-positive ($0.8735 \rightarrow 0.8805$) at a third of the model size, because the geometry-bearing positive structure carries the discrimination the suppressors provided under the weaker read-out. Suppressors thus remain a designed, load-bearing mechanism in general, even where—as on MNIST with this read-out—they prove unnecessary.

**Read-out policy.** Varying the maturity threshold changes final accuracy by less than one standard deviation, and unified matching scores marginally below anchored (Appendix D); read-out policy is a stable, secondary factor relative to the information source.

## G The Multi-Scale Ablation in Full

Retraining the identical system on the *finest scale only*—no coarsening, no cross-scale relations, everything else including the read-out unchanged—drops accuracy from $0.874$ to $\mathbf{0.730}$ at comparable model size (191 vs. 172 csvs, five seeds), a gap large relative to the seed spread. Multi-scale structure is therefore not a refinement of the representation but a condition on how well the learner does with it: with relations at a single extent, whatever is refined away leaves no longer-range relation spanning the gap, and structure erodes instead of consolidating (Appendix G; the remaining ablations are in Appendix F).

Table 3: Multi-scale ablation: representation $\times$ read-out in one pipeline (five seeds; the full multi-scale row reproduces the ten-seed headline). Final end-of-cycle accuracy.

| | Alternative read-out | Geometric |
|---|---|---|
| Full multi-scale | 0.716 | **0.874** |
| Single-scale | 0.439 | 0.730 |

Appendix G reports the headline of this ablation; here are the details. We retrain the identical system on the *finest scale only*—no coarsening, no cross-scale relations—and score it under both read-outs, five seeds each (Table 3, Fig. 3). Under the geometric read-out accuracy falls from $0.874$ to $0.730$; under the simpler alternative read-out of Appendix D it falls from $0.716$ to $0.439$. Model size is comparable in both cases (191 vs. 172 csvs), so this is not a capacity difference.

Two things are worth noting. First, the gap is present under either read-out and in neither case is it small relative to the spread across seeds, so the contribution of carrying multiple scales is not an artifact of how the structure is decoded. Second, the two contributions are complementary rather than substitutable: the geometric read-out lifts the single-scale model substantially as well ($0.439 \rightarrow 0.730$), and the multi-scale gap narrows under it ($0.28 \rightarrow 0.14$) without closing. Part of what relations at multiple extents provide is thus geometry that a read-out can exploit, and part is the structural continuity that keeps refinement from eroding what it has built—the latter cannot be recovered at read-out time, because the structure in question was never formed.

## H Statistical Tests

Seeds are paired across methods—a given seed fixes the sample draw and the order of presentation, so every method sees the identical stream—which makes a paired test the appropriate family. With ten seeds we do not assume normality, so we use the two-sided Wilcoxon signed-rank test on final end-of-cycle accuracy, one comparison per baseline (Table 4).

Three features of the outcome are worth stating plainly. For most baselines the test statistic is $W = 0$: our system is more accurate on *every one* of the ten paired seeds, which is arguably more informative than the $p$-value itself. With $n = 10$ the smallest attainable two-sided $p$ is $2/2^{10} = 0.00195$, so several comparisons sit at the resolution limit of the test—they cannot be more significant than they are, and the test cannot distinguish "clearly better" from "overwhelmingly better" at this sample size. Third, and against us, two comparisons are *not* significant and one is significant in the opposite direction. On final accuracy SI is indistinguishable from ours ($p = 1.00$; it is ahead on six of ten seeds) and so is MAS

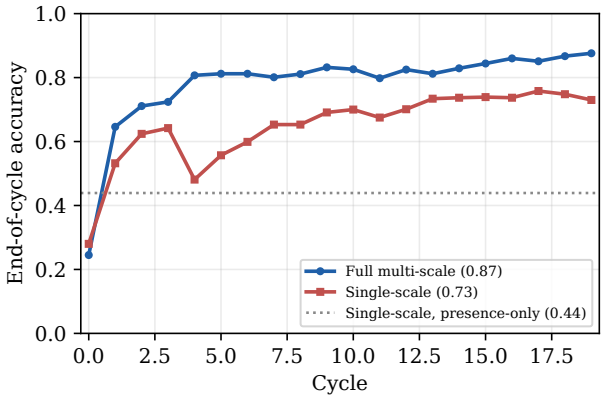


Figure 3: Removing multi-scale structure. Retrained on the *finest scale only*: the full system (blue) reaches 0.87; the single-scale system under the geometric read-out (red) plateaus near 0.73; under the simpler alternative read-out (dotted) it reaches only 0.44.

($p = 0.24$), while DER++ with 80 stored images has $W = 0$ in its own favour, exceeding us on all ten seeds. Retention separates all three from ours decisively where final accuracy does not (Table 1).

We note, in the interest of transparency, that the authors are not specialists in statistical methodology; the tests reported here were selected and carried out with AI assistance, and we have restricted ourselves to a standard paired non-parametric test and to stating its limits rather than attempting anything more elaborate.

We deliberately do not report significance for the multi-scale ablation (Appendix G) or the threshold sweep (Appendix M), which use five and two seeds; a signed-rank test bottoms out at $p = 0.0625$ for $n = 5$ and is meaningless for $n = 2$, so we report those as means with their spread and make no significance claim. For the learning variants (Appendix E) the same test over ten paired seeds returns $p = 0.64$, $p = 0.39$ and $p = 0.34$ for the no-suppressor, pos-gate and pos-skip variants respectively—that is, none of them differs significantly from the baseline, which is what the text there claims.

# I Within-Cycle Retention: Definition and Full Values

**Definition.** Let $a_y(t, c)$ be the held-out accuracy of class $y$ measured after the block in which class $c$ was trained during cycle $t$, so that within cycle $t$ the blocks $c = 0, \ldots, 9$ index successive measurement points. For a class $c$ in cycle $t$ the reference is its accuracy immediately after its own block, $a_c(t, c)$, and we compare against two later points of the same cycle: its accuracy at the end of the cycle, $a_c(t, 9)$, and its lowest accuracy at any block from its own onward, $\min_{c \le j \le 9} a_c(t, j)$. Writing $\mathcal{P}_T = \{(t, c) : t \in T,\ a_c(t, c) > \theta\}$ for the class-cycle pairs entering the average,

$$\mathrm{WCR}_{\mathrm{end}} = \frac{1}{|\mathcal{P}_T|} \sum_{(t,c)\in\mathcal{P}_T} \min\Big(\frac{a_c(t, 9)}{a_c(t, c)},\, 1\Big), \tag{3}$$

$$\mathrm{WCR}_{\mathrm{wst}} = \frac{1}{|\mathcal{P}_T|} \sum_{(t,c)\in\mathcal{P}_T} \min\Big(\frac{\min_{c\le j\le 9} a_c(t, j)}{a_c(t, c)},\, 1\Big), \tag{4}$$

averaged additionally over seeds. In the main text $T = \{0, 1, 2\}$ and $\theta = 0.3$.

**Why these choices.** Three details matter, and each was made to avoid flattering our own system. *The threshold $\theta$.* A class that is still at chance right after its own block has nothing to lose, and its ratio would be $\approx 1$ for that reason alone. This is not hypothetical: in cycle 0 the model has $\sim$20 csvs and eight of the ten classes sit near chance, so without a threshold the measure is dominated by pairs that cannot move. We therefore require $a_c(t, c) > \theta = 0.3$. *The cap at 1.* Early in the stream a class may keep *improving* after its own block, as later classes sharpen the decision among them; such pairs give ratios above 1 that would offset genuine losses elsewhere in the average. Capping each ratio at 1 makes the measure record only what was given up. *The worst-case variant.* Comparing only the endpoints of a cycle is blind to what happens in between: a class can lose half its accuracy mid-cycle and recover by the cycle's end, and $\mathrm{WCR}_{\mathrm{end}}$ would report almost no loss. One class in our own runs does exactly this, falling from 0.80 to 0.40 during cycle 1 and returning to 0.74 by its end. $\mathrm{WCR}_{\mathrm{wst}}$ is reported alongside for that reason.

**Early cycles versus the whole run.** The main text reports $T = \{0, 1, 2\}$ because that is the regime the comparison is about. For completeness Table 5 also gives $T = \{0, \ldots, 19\}$. Over the whole run every method improves, which is the ex-

Table 4: Two-sided Wilcoxon signed-rank tests on final accuracy, our system against each baseline, over the ten paired seeds. *Difference* is ours minus theirs, so a negative value means the baseline is ahead. The first block is the three comparisons that do not favour us on this metric; *seeds ahead* counts the paired seeds on which our accuracy is higher (ties are dropped by the test).

| Comparison | difference | $W$ | $p$ | seeds ahead |
|---|---|---|---|---|
| vs. SI | $-0.005$ | 18 | 1.0000 | 4/10 |
| vs. MAS | $+0.008$ | 12 | 0.2383 | 6/10 |
| vs. DER++ (80 img) | $-0.038$ | 0 | 0.0039 | 0/10 |
| vs. DER++ (8 img) | $+0.065$ | 2 | 0.0059 | 9/10 |
| vs. LwF | $+0.076$ | 0 | 0.0020 | 10/10 |
| vs. Naive replay $r$=0.8 | $+0.115$ | 0 | 0.0039 | 9/10 |
| vs. Naive replay $r$=0.5 | $+0.125$ | 0 | 0.0020 | 10/10 |
| vs. Naive replay $r$=0.2 | $+0.158$ | 0 | 0.0020 | 10/10 |
| vs. Plain NN | $+0.214$ | 0 | 0.0020 | 10/10 |
| vs. Expert-AEC ($k$=2) | $+0.268$ | 0 | 0.0020 | 10/10 |
| vs. ER-ACE (8 img) | $+0.476$ | 0 | 0.0020 | 10/10 |
| vs. CNN | $+0.547$ | 0 | 0.0020 | 10/10 |

Table 5: Within-cycle retention over the early cycles (as in the main text) and over the entire run. Ten seeds; $\theta = 0.3$; ratios capped at 1.

| Method | cycles 0–2 end | cycles 0–2 worst | all 20 cycles end | all 20 cycles worst |
|---|---|---|---|---|
| **Modeller (ours)** | **0.95** | **0.90** | **0.97** | **0.94** |
| Naive replay $r$=0.8 | 0.52 | 0.40 | 0.72 | 0.65 |
| Naive replay $r$=0.5 | 0.46 | 0.34 | 0.69 | 0.62 |
| Naive replay $r$=0.2 | 0.32 | 0.25 | 0.61 | 0.53 |
| Plain NN | 0.16 | 0.15 | 0.48 | 0.40 |
| CNN | 0.11 | 0.11 | 0.24 | 0.21 |
| Expert-AEC ($k$=2) | 0.75 | 0.72 | 0.95 | 0.95 |

pected consequence of repeated re-exposure: the plain network rises from 0.16 to 0.48, and replay at $r$=0.8 from 0.52 to 0.72. The Modeller changes least ($0.95 \rightarrow 0.97$), having had the least to recover from. Averaged over all twenty cycles the gap therefore narrows, which is precisely the point of restricting attention to the early cycles: the long-run figures describe a regime that has become effectively i.i.d.

## J Standard Continual-Learning Baselines

This appendix documents the regularisation and replay baselines of Table 1: what was implemented, how each was tuned, and the sweeps behind the single configuration reported for each. All run on the identical stream, backbone and gradient budget as the other baselines, and were implemented directly rather than taken from a library, so their specification is given here in full.

**Methods.** **SI** (Zenke, Poole, and Ganguli 2017) accumulates a per-parameter path integral $\omega_k \mathrel{+}= -g_k\,\Delta\theta_k$ during training and, at each consolidation, folds $\mathrm{relu}\big(\omega_k/(\Delta_k^2+\xi)\big)$ into a cumulative importance $\Omega_k$, penalising $\sum_k \Omega_k(\theta_k - \tilde{\theta}_k)^2$; $g_k$ is the gradient of the *task* loss, and $\xi$=0.1. **MAS** (Aljundi et al. 2018) instead accumulates $\big|\partial\|f(x)\|^2/\partial\theta_k\big|$ over the block's samples, computed on the logits and requiring no labels, with the same quadratic penalty. **LwF** (Li and Hoiem 2018) keeps a snapshot of the model and adds a distillation term between snapshot and current outputs on the *current* block's inputs, at temperature $T$=2 with the standard $T^2$ scaling. **DER++** (Buzzega et al. 2020) stores $(x, y)$ together with the logits emitted when the example was inserted, drawing two independent buffer batches for $\mathcal{L} = \mathrm{CE}(\text{new}) + \alpha\|f(x_1) - z_1\|^2 + \beta\,\mathrm{CE}(x_2, y_2)$. **ER-ACE** (Caccia et al. 2022) restricts the incoming batch's cross-entropy to the classes present in it plus those not yet seen, leaving the buffer term over the full label space. Buffers are filled by reservoir sampling.

**Penalty strength dominates, and the usual values are inert.** The regularisation methods are far more sensitive to their penalty coefficient $C$ than to anything else. With Adam at $10^{-3}$ the parameters move little within a block, so $(\theta - \tilde{\theta})^2$ is of order $10^{-6}$ and at $C \sim 1$—the scale commonly quoted for these methods on larger models—the penalty gradient is under 1% of the task gra-

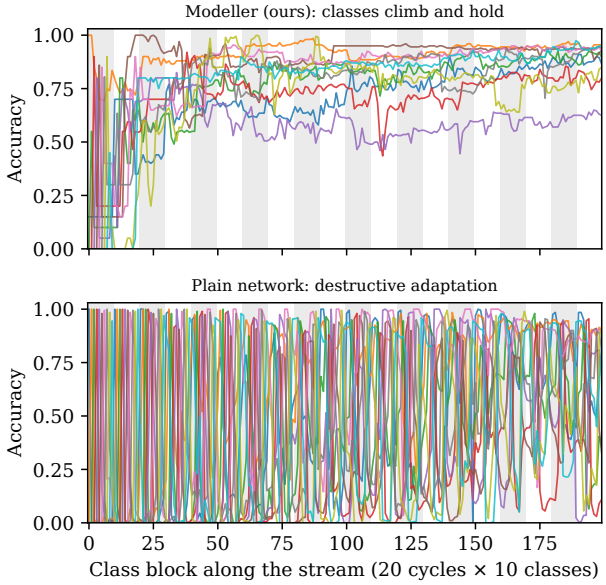


Figure 4: Per-class held-out accuracy along the class-incremental stream (ten-seed means). Each alternating white/grey band is one *cycle*; within a band the ten tick positions are that cycle's ten class blocks in order, so the $k$-th point is the model just after the $k$-th class was trained. Each line follows one class throughout. **Top:** the Modeller—classes climb and *hold*, with transient, recovering dips. **Bottom:** the plain network—each class collapses when the following classes train, the destructive-adaptation signature.

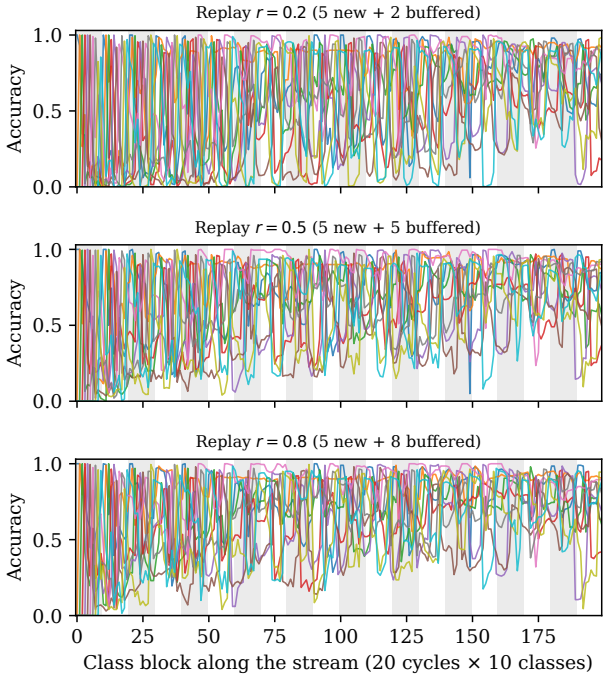


Figure 5: Replay *damps* destructive adaptation but does not remove it. Per-class accuracy along the stream (ten-seed means; axis convention as in Fig. 4) for the three replay buffers. Compare the plain network (Fig. 4, bottom, 0 buffered): each added stored example lifts the troughs, yet at every setting the per-class curves still collapse and recover repeatedly, cycle after cycle—the oscillation is attenuated, not eliminated. The Modeller (Fig. 4, top) shows no such pattern while storing nothing.

dient and the regulariser is numerically switched off: SI, MAS and the plain network then return *identical* results. Sweeping $C$ over five orders of magnitude, each method has a clear and quite different optimum. Reporting a regularisation baseline without this sweep measures nothing. The values tried were $C \in \{0.1, 1, 10, 100, 300, 10^3, 3\times 10^3, 10^4\}$ for SI, $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2\}$ for MAS and $\{0.1, 0.3, 1, 3, 10, 30\}$ for LwF; six $(\alpha, \beta)$ pairs in $[0.1, 1]$ for DER++; the consolidation cadence over $\{1, 2, 3, 7, 10, 13, 20, 200\}$ gradient steps; and the buffer over $\{2, 5, 8, 20, 50, 80\}$ examples.

**The consolidation schedule, and asynchrony.** Each regularisation method must be told *when* to consolidate—when to declare the current parameters the ones to protect. We varied that cadence from every gradient step to once per cycle (20 steps is one class block, 200 one cycle). Cadences that divide 20 are *aligned*: a consolidation interval never spans a class change. Cadences of 3, 7 and 13 do not divide 20, so consolidation drifts through the block and straddles class changes constantly.

Two extremes fail, for different reasons. At every step the anchor $\tilde{\theta}$ is reset to the current parameters, so at the next step $\theta = \tilde{\theta}$ and both the penalty and its gradient are *identically* zero: all three methods reduce exactly to the plain network, and SI, MAS and plain agree to three decimals (0.663). At once per cycle the anchor goes stale across ten class blocks and performance collapses. Between those extremes the choice matters little—and, contrary to what the mechanisms suggest, *alignment does not matter at all*: the asynchronous cadences are as good as or better than the aligned ones for every method. Our system has no cadence to choose.

Table 6: Penalty-strength sweep, bracketing each method's optimum (ten seeds, consolidation per block). WCR is $\text{WCR}_{\text{end}}$.

| Method | $C$ | Final | Early | WCR |
|---|---|---|---|---|
| SI | 300 | 0.806±0.039 | 0.469 | 0.54 |
| SI | 1000 | 0.803±0.063 | 0.474 | 0.56 |
| SI | 3000 | 0.807±0.048 | 0.468 | 0.54 |
| MAS | 0.003 | 0.856±0.018 | 0.174 | 0.18 |
| MAS | 0.01 | 0.836±0.020 | 0.185 | 0.19 |
| MAS | 0.03 | 0.867±0.012 | 0.233 | 0.24 |
| LwF | 1.0 | 0.712±0.041 | 0.136 | 0.14 |
| LwF | 3.0 | 0.730±0.038 | 0.178 | 0.18 |
| LwF | 10 | 0.695±0.061 | 0.228 | 0.25 |

Table 7: Consolidation cadence, in gradient steps, at each method's best $C$ (ten seeds). Each pair is Final and $\text{WCR}_{\text{end}}$. ASYNC cadences do not divide the 20-step block, so they straddle class changes.

| | | SI | | MAS | | LwF | |
|---|---|---|---|---|---|---|---|
| Cadence | Phase | Final | WCR | Final | WCR | Final | WCR |
| 1 | aligned | 0.663 | 0.16 | 0.663 | 0.16 | 0.644 | 0.17 |
| 2 | aligned | 0.850 | 0.30 | 0.818 | 0.17 | 0.675 | 0.14 |
| 3 | ASYNC | 0.864 | 0.37 | 0.830 | 0.17 | 0.726 | 0.14 |
| 7 | ASYNC | 0.879 | 0.44 | 0.830 | 0.18 | 0.798 | 0.19 |
| 10 | aligned | 0.836 | 0.58 | 0.840 | 0.18 | 0.742 | 0.14 |
| 13 | ASYNC | 0.836 | 0.47 | 0.855 | 0.18 | 0.725 | 0.23 |
| 20 (block) | aligned | 0.803 | 0.56 | 0.836 | 0.19 | 0.730 | 0.18 |
| 200 (cycle) | aligned | 0.329 | 0.20 | 0.624 | 0.17 | 0.569 | 0.11 |

**One sample at a time.** Our system integrates one sample at a time; the baselines see each block as a batch of five, revisited. Because that difference is a property of the protocol rather than of the methods, we reran the no-storage baselines in our regime—batch of one, four gradient steps per sample, so that the total gradient budget per block is unchanged and only the *grouping* of the data differs. Every one of them loses accuracy, and their retention stays far below ours. We restrict this comparison to methods that store nothing: for a replay method the constraint does not bind, since the buffer supplies a multi-sample batch at every step regardless. The parity is therefore on presentation order, not on work done per observation, for which our system has no per-sample analogue.

Table 8: Batch of five versus one sample at a time, at an identical total gradient budget (ten seeds). No-storage methods only. WCR is $\mathrm{WCR}_{\mathrm{end}}$.

| | Batch of five | | | One at a time | | |
|---|---|---|---|---|---|---|
| Method | Final | Early | WCR | Final | Early | WCR |
| Plain | 0.663 | 0.158 | 0.16 | 0.577 | 0.155 | 0.16 |
| SI | 0.803 | 0.474 | 0.56 | 0.776 | 0.406 | 0.48 |
| MAS | 0.836 | 0.185 | 0.19 | 0.659 | 0.150 | 0.15 |
| LwF | 0.730 | 0.178 | 0.18 | 0.622 | 0.206 | 0.26 |
| **Modeller** | | — | | **.874** | **.536** | **.95** |

**Replay methods and the size of the store.** DER++ was swept over $\alpha$ and $\beta$ at both buffer capacities, receiving the same tuning as the regularisation methods. At the capacity the main comparison holds fixed (8 images) it reaches $0.810$; at $80$ images—$8\%$ of the entire stream, with the incoming stream unchanged—it reaches $0.912$ with retention above ours. ER-ACE behaves differently: its retention is high at every capacity, because its whole mechanism is to prevent arriving data from suppressing established classes, but its accuracy is low at small buffers. The reason is specific to a cyclic stream: once every class has been seen, the incoming batch of a single-class block admits only its own label, its cross-entropy is exactly zero, and the method learns from its buffer alone. Its numbers here therefore reflect a protocol mismatch rather than a defect of the method.

Table 9: Replay methods at both buffer capacities (ten seeds). Each pair is Final and $\mathrm{WCR}_{\mathrm{end}}$.

| DER++ | | 8 images | | 80 images | |
|---|---|---|---|---|---|
| $\alpha$ | $\beta$ | Final | WCR | Final | WCR |
| 0.1 | 0.5 | 0.799 | 0.50 | 0.883 | 0.76 |
| 0.5 | 0.5 | 0.779 | 0.60 | 0.905 | 0.89 |
| 1.0 | 0.5 | 0.793 | 0.67 | 0.910 | 0.94 |
| 0.5 | 1.0 | 0.790 | 0.62 | 0.912 | 0.91 |
| 1.0 | 1.0 | 0.810 | 0.67 | 0.906 | 0.95 |
| 0.2 | 1.0 | 0.790 | 0.56 | 0.898 | 0.85 |
| ER-ACE | | Final | WCR | Final | WCR |
| $r$=0.2 | | 0.150 | 0.63 | 0.631 | 0.95 |
| $r$=0.5 | | 0.337 | 0.79 | 0.772 | 0.97 |
| $r$=0.8 | | 0.399 | 0.81 | 0.825 | 0.98 |

## K Enlarging the Replay Buffer

Two questions about the baselines of Sec. 7 are worth settling here. The first is whether they are adequately configured at all: the same fully-connected network trained in the ordinary i.i.d. manner reaches $0.979$ test accuracy on full MNIST, and $0.90$ when restricted to the stream's total budget of $100$ samples per class presented i.i.d. Their low continual numbers therefore reflect the class-incremental stream, not the architecture or the optimizer.

The second is how far replay improves when its buffer is no longer kept comparable in scale to the five new samples per block. We repeat it with the buffer enlarged by an order of magnitude—$20, 50, 80$ stored examples at $r$=$0.2, 0.5, 0.8$, against a per-class training budget of only $100$ samples, so the buffer approaches a class's entire data. Final accuracy rises to $0.807$, $0.833$, and $0.882$ respectively (three seeds), up from $0.717$, $0.750$, $0.760$; the largest setting slightly exceeds the Modeller ($0.874$). The crossover therefore lies between $50$ and $80$ retained examples—between roughly $5\%$ and $8\%$ of the entire stream. We lead at every capacity below it, so the comparison is not decided at the $8$-example setting the main text holds fixed. Relaxing the stream further, so that each block supplies a class's full data rather than five samples, raises batch-composition replay to $\sim$0.93.

The same holds for the tuned modern methods. At the $8$-example capacity the comparison fixes, DER++ reaches $0.810$ with $\mathrm{WCR}_{\mathrm{end}}$ $0.67$; at $80$ it reaches $0.912$ with $0.91$, exceeding us on both. That is the only regime in which any baseline overtakes us on both measures, and two things locate it. The first is its composition: $80$ retained examples replayed alongside five new ones is sixteen stored for every one arriving, so roughly $94\%$ of each update is data the learner has already seen—much closer to ordinary training on a growing pooled set than to learning from a stream. The second is its size relative to the stream, which is the only way to compare buffers across datasets of different scale. Our stream is $1{,}000$ samples in total, so $80$ retained is $8\%$ of everything the learner ever sees; the buffers used for Seq-CIFAR-10 by Buzzega et al. (2020)—$200$, $500$ and $5120$ against $50{,}000$ training images—are $0.4\%$, $1\%$ and $10\%$ of theirs. Measured that way our largest setting sits near the top of the range the field itself uses, not below it. We report it because it is the honest upper end of what replay buys, not because it is a comparable setting.

This is the expected behavior, and it locates the trade precisely. Figure 8 shows the trajectories: the Modeller still leads through the early cycles, and the large-buffer runs overtake it only late, once enough of the past has been stored and re-presented. Figure 9 shows why the accuracy improves—with 20–80 examples in the buffer the per-class oscillation of Fig. 5 is strongly suppressed, since most of each class is re-presented at every step. Retention here is therefore purchased directly with stored data: at $r$=$0.8$ the buffer holds up to $80\%$ of every class the network has seen, which is closer to interleaved i.i.d. training than to a boundary-free stream. For reference, the same network trained i.i.d. on this data budget reaches $\sim$0.90, and relaxing the stream so each block supplies a class's full data raises batch-replay to $\sim$0.93. The Modeller attains $0.874$ storing nothing and re-presenting nothing.

## L Expert-Routing Baseline: All Configurations

The expansion baseline spawns a new expert whenever the reconstruction error of every existing expert exceeds a $k$-scaled threshold, so $k$ controls how readily capacity is added; prediction routes each test sample to a single expert. Figure 10 shows the per-class trajectories for every configuration we ran.

The failure mode is qualitatively different from replay's. Because an expert is reached almost exclusively by observations of one class, the curves show none of the replay oscillation: they are smooth and drift slowly upward, consistent with the within-cycle retention of $\approx 1$ reported in Table 14. What limits the method is *routing*: many classes sit at $0.1$–$0.5$ for the whole run despite being retained somewhere in the model, because test-time expert selection sends their samples to the wrong expert. Lowering $k$ spawns more experts, makes routing harder, and lowers the ceiling monotonically ($0.607 \rightarrow 0.482 \rightarrow 0.294 \rightarrow 0.218$). In the two configurations that reach the 30-expert cap the curves flatten into horizontal lines—capacity is exhausted and the system stops improving at all, most starkly in the batch-1 setting. Isolation, in short, removes overwriting at the cost of integration: the knowledge is retained but cannot be brought to bear on a query, which is precisely the failure our single-model, no-routing read-out avoids.

## M Parameter Sensitivity

Four thresholds govern how readily structure is created and retired, and it is worth saying what each controls before reporting the sweep. The *significance threshold* $\epsilon_{\text{sign}}$ is the level of $P(\text{SS}(C) \mid I(T))$ below which a conditioner is discarded as not accounting for its target; lowering it retains more weakly-supported structure. The *reintegration threshold* is how nearly coextensive an upstream's presence must be with its downstream's before the two are merged; *lowering* it merges more readily, and so collapses distinctions that refinement had drawn, while raising it merges less and leaves upstream levels standing after they have stopped carving any distinction. The *removal* and *reintegration rates* set how often each maintenance operation is considered at all, and thus how quickly the population responds to its own statistics. All four were held *fixed* at the values of Appendix V for every run reported in this paper; the sweep below was carried out afterwards, to characterise sensitivity rather than to choose them.

We vary these thresholds one axis at a time around the shipped configuration (two seeds each, geometric read-out; Table 10). The shipped defaults sit at or near the best value on every axis, and no off-default setting exceeds the default meaningfully. Accuracy is fairly stable across the removal rate ($0.848$–$0.873$) and the reintegration rate ($0.855$–$0.863$), and more sensitive to the reintegration threshold ($0.815$–$0.873$) and the significance threshold ($0.795$–$0.873$). The reintegration threshold is the one axis that degrades in *both* directions, and for opposite reasons: at $0.80$ ($0.843$) too much is folded back and distinctions refinement had drawn are collapsed, while at $0.95$ ($0.815$) too little is, and upstreams that have ceased to carve a distinction are left standing as separate levels. Loosening the significance threshold to $0.05$ ($0.795$) instead retains too much spurious structure. The result thus does not hinge on fine-grained tuning, but the two threshold defaults are well-chosen rather than arbitrary. We deliberately do not sweep the reliability gates, which prior analysis shows to be load-bearing: relaxing them causes unbounded structure growth rather than an informative accuracy change.

Table 10: One-axis-at-a-time threshold sweep (two seeds; final accuracy). Default column is the shipped configuration (ten-seed baseline $0.8735$).

| Axis (low/def/high) | low | def | high |
|---|---|---|---|
| Reintegration thresh. ($.80/.90/.95$) | 0.843 | **0.873** | 0.815 |
| Removal rate ($.3/.5/.7$) | 0.873 | **0.873** | 0.848 |
| Reintegration rate ($.3/.5/.7$) | 0.863 | **0.873** | 0.855 |
| Significance thresh. ($.05/.1/.2$) | 0.795 | **0.873** | 0.878 |

## N Discussion in Full

The main text compresses this discussion.

**Accuracy and the representation ceiling.** The system reaches $0.874$—well above the $\approx 0.50$ this methodology previously attained on ten-class MNIST (Erden and Faltings 2026), and at or above every baseline at comparable storage—while learning in a manner statistical methods cannot. Two tuned regularisation baselines converge to the same final figure, for the reason given below: end-of-run accuracy is not what distinguishes these methods. What stands between it and a conventionally trained network is chiefly the feature vocabulary: first-order change points on a binarized contour discard most of the image, and the one clearly weak class, the $4/9$ confusion, is exactly where that vocabulary supplies no distinction holding across instances. The construction extends naturally in that direction without touching the learner, so a richer vocabulary is the most direct route to the remainder.

**Where the effort moves.** A gradient-trained network learns its features from data, paying with sample volume, repeated exposure and the destructive dynamics analyzed above; ours instead *presupposes* a representation—a front-loaded cost, but *a single universal principle* applied once (Sec. 4) rather than per-task engineering. The return appears where the baselines are weakest: because features need not be discovered by optimization, the first exposures already produce stable, retained structure (Fig. 2), while the statistical methods spend those cycles overwriting and relearning, closing the gap only once re-exposure makes the stream effectively i.i.d. A second cost is redundancy: each csv conditions a single target, so structure shared between classes is represented separately, inflating model size—an implementation economy, with multi-target units a substantial refactor orthogonal to this paper (Appendix B).

**The road ahead.** Several extensions follow the same principle without altering the learner: deepening the vocabulary by taking *higher orders of change*; applying the construction to the un-thresholded image as an intensity *surface* rather than a traced contour (Sec. 4); and extending to 3D feature points, which the learner would process identically. Beyond the representation, the most consequential step is hierarchical, compositional structure—the construction here operates *within* a single shape, and the same principle points one level up, to part–whole organization across entities. Two limitations remain: the read-out uses absolute node position as

a class-conditional cue, which suits centred MNIST (position enters only as post-match evidence, never as a matching criterion, but behavior under translation and scale is future work), and wall-clock cost is dominated by graph matching.

## O Scope and Claims in Full

The main text compresses this discussion.

Our interest in this work is shape as structure. The learning algorithm is blind to the dimensionality and semantics of the network it refines: a network over 3D feature points would be processed identically. What is *not* dimension-blind is the conversion from images to networks. Natural photographs are 2D projections of 3D scenes; a structured 3D representation of an object from images is a substantial problem in its own right, and the alternative—learning appearance across many 2D views of each object, which is effectively what statistically trained networks do—is precisely the path this methodology is built to avoid. A structural account of recognition has a long history in the study of human vision, though not a settled one: recognition is argued to rest on structural descriptions rather than on accumulated 2D appearances—object-centred in Marr and Nishihara's formulation (Marr and Nishihara 1978), and organized around viewpoint-invariant volumetric parts in Biederman's (Biederman 1987). We take from this tradition only its motivating intuition, not a claim about how human recognition is settled. We therefore deliberately restrict this work to intrinsically 2D shape, for which MNIST is the canonical benchmark, and treat structured 3D feature extraction as the enabling step for future extensions (see also the corresponding discussion in Erden and Faltings 2026). It is worth being clear about what a harder benchmark would and would not test here. Whether each observation is integrated exactly once, whether the response to every past observation is preserved, and whether anything has to be stored are properties of the learning process, and they do *not* become more or less true on a more difficult dataset. What a harder dataset would stress is the feature vocabulary—which is already the limiting factor on accuracy, by our own account (Sec. 8)—rather than the mechanism under study.

**What is and is not being claimed.** This representation is deliberately minimal: it relies on nothing but edge detection and traversal, and it discards all appearance information (stroke, texture, gray levels) and all boundary detail between orientation changes. It is not proposed as a competitive descriptor for handwritten digits. The claim is the *abstract design*: multi-scale visual structure, together with its cross-scale relations, can be expressed as a single relational network on which a structural continual learner operates directly. Richer instantiations of the same design—denser feature vocabularies, appearance attributes on nodes, finer scale ladders—are natural extensions and can be slotted in without changing the learner.

We do, however, defend the underlying principle more strongly than we can demonstrate it here. Representing a continuous entity by its points of change, typed by the order and character of that change, is in our view capable of furnishing a *complete* feature description of shape, to whatever fidelity is wanted, as further orders of change are admitted—and the same holds for the surface formulation above. We do not claim this as an experimental result: establishing it would require building those higher-order and surface constructions and testing them, which is a substantial piece of work in its own right and outside the scope of this paper. We state it as the position the representation is built on, and one we consider both defensible and testable.

**Why this setting, at greater length.** This work is deliberately a proof of concept on intrinsically 2D shape, and it is worth being clear about why, since the restriction is in the image-to-network conversion and not in the learner (Sec. 4). We chose the setting in which the structural content of an observation is available without first solving a second hard problem: for a flat shape, a network of typed change points can be read off directly, whereas for natural photographs—2D projections of 3D scenes—obtaining a structured description is a substantial research problem in its own right, and the alternative of learning appearance across many stored 2D views is the very approach this methodology exists to avoid. Studying the learning machinery therefore calls for the simplest input that still has genuine structure, which is what MNIST provides.

## P Baseline Assumptions in Full

The main text summarizes this; the fuller statement follows.

The comparison is not between methods on equal footing, and it is more informative to say what each one requires than to normalize them. Every method receives the identical stream of *new* observations—the same five previously-unseen samples of the current class per block, 100 unique samples per class over the run—and they differ in what they additionally need. The regularisation baselines (SI, MAS, LwF) store no past data, but each requires a *consolidation schedule*—a designated moment at which the parameters currently held are declared the ones to protect—and LwF additionally keeps a full copy of the previous model. Our system requires neither (Appendix J). The replay baselines need past data to be *available, stored, and re-presented* to the network; the replay ratio $r$ sets the buffer size relative to the training batch (10 here), so the buffer holds 2, 5, or 8 examples at $r{=}0.2, 0.5, 0.8$, a re-exposure budget of the same order as the five new samples. Beyond storage, and in common with statistical learning generally, none of these baselines learns on a sample-by-sample basis: each block is a small batch revisited for 20 gradient updates (4,000 over the run, against the Modeller's 1,000—one per sample, never revisited). The Modeller stores nothing, re-presents nothing, and needs no buffer.

Further results are collected in the appendices: the remaining expert-routing configurations, spanning its spawn-threshold $k$ (Appendix L, Table 14), a per-class breakdown for every method (Table 15), the significance tests (Appendix H), and the effect of relaxing the replay assumption (Appendix K).

We deliberately do not compare against methods that require *visible task boundaries*—a signal announcing that one task has ended and another begun. On a genuine stream of experience no such signal exists, and assuming it is difficult

to defend: it presupposes exactly the segmentation of experience that a continual learner ought to discover. Instead we include the expert-routing model (Erden, Gasmi, and Faltings 2025) as the representative of the expansion/isolation family, run *without* any task-boundary signal: it must decide for itself, from reconstruction error alone, when a new expert is warranted. That is the same information our system has, and it makes the comparison meaningful rather than merely favorable. It is worth noting that this method still carries a weaker assumption of the same family: although no boundary is announced, a task is expected to persist unchanged long enough for the current expert to stabilize before the input distribution moves on. Our setting grants no such interval—the class in front of the learner changes every five samples—so the assumption is considerably weaker than an explicit boundary but not absent.

For completeness we also examined what happens when the replay assumption is relaxed further—an order-of-magnitude larger buffer, and a variant in which each block supplies a class's full data. Replay improves as expected in proportion to how much past data it stores and re-presents, up to $\sim 0.93$, but those settings are no longer comparable to the Modeller's; we report them, with the corresponding verification that the baselines here are well-configured, in Appendix K.

## Q Related Work in Full

The main text compresses this discussion; the fuller version follows.

**Multi-scale and scale-space representations.** Describing visual structure across a hierarchy of scales—coarse structure stable and general, fine structure specific—is classical: scale-space filtering (Witkin 1983), the deep structure of images (Koenderink 1984), automatic scale selection (Lindeberg 1998), image pyramids (Burt and Adelson 1983), and wavelet multiresolution (Mallat 1989). Our representation inherits the principle but realizes it structurally rather than signal-analytically: we coarsen a *relational graph* and, crucially, do not keep the scales as a stack of separate descriptions—all levels are accumulated into a *single unified network* in which nodes are shared and edges from every level coexist, so relations are expressed both within and *across* scales in one structure. Graph pyramids (Kropatsch 1995; Montanvert, Meer, and Rosenfeld 1991) are the closest graph-domain precedent, but build hierarchies of successively contracted graphs; we collapse the hierarchy into one augmented network serving a continual structural learner. Local descriptors—SIFT (Lowe 2004), built on Gaussian scale space, and shape context (Belongie, Malik, and Puzicha 2002), which achieves scale invariance by normalizing distances—reduce a shape to fixed-length vectors compared against stored exemplars; where correspondence is iterated, as in the latter, what is refined is the alignment of a single pair rather than a model retained across samples. More fundamentally, across this body of work the multi-scale representation is a *front end*: features are computed and handed to a *separate* learner. We are not aware of a method in which a multi-scale representation is integral to a *non-gradient, structural* learner that learns *by refining that very representation*.

**Hierarchical visual architectures.** Layered abstraction of visual features runs from Marr's program (Marr 1982) through the Neocognitron (Fukushima 1980) and HMAX (Riesenhuber and Poggio 1999) to modern deep networks, which realize the hierarchy as gradient-trained filter banks whose internal representation is monolithic and which carry the destructive-adaptation difficulties that motivate this line of work. Our hierarchy is instead explicit, gradient-free, and inspectable at every level.

**Structural shape representation and matching.** Representing shape as attributed relational structure and recognizing by graph matching is a long tradition: error-tolerant attributed-graph matching (Messmer and Bunke 1998) and shock graphs (Siddiqi et al. 1999) do exactly this, and in cognitive science recognition by components (Biederman 1987) gives the structural-description account of the same idea. Deformable part-based models (Felzenszwalb et al. 2010) retain the relational arrangement—parts coupled by spring-like connections—but with learned appearance filters in place of symbolic parts, and infer the model directly against the image rather than matching two graphs. More recently, neural models learn representations over graph-structured data directly (Kipf and Welling 2017). Our state polynetworks sit in this tradition, but matching is reframed as a step *inside online learning*: refinement incrementally computes, with statistical tolerance and a preservation guarantee, the shared substructure of a unit's recurrences, rather than a one-shot comparison or a gradient-trained embedding.

**Compositional and non-gradient concept learning.** Bayesian program learning (Lake, Salakhutdinov, and Tenenbaum 2015) shares our emphasis on structured, comprehensible concept models but is generative over a hand-designed compositional grammar, with its primitives learned from a background set of alphabets, and is not framed for lifelong integration; capsule networks (Sabour, Frosst, and Hinton 2017) pursue part–whole structure with gradient training. Outside the gradient paradigm, Adaptive Resonance Theory (Carpenter and Grossberg 1987) treats the stability–plasticity dilemma via a vigilance test—analogous to our refinement tolerance $T_{\mathrm{ref}}$—while incremental concept formation (Fisher 1987) and growing self-organizing models (Fritzke 1995), which extend the fixed-lattice self-organizing map (Kohonen 1982) by inserting units and connections where the data demand them, add units on demand, as our variation process does. We differ in learning explicitly relational, multi-level structure with a preservation guarantee.

**Continual learning in vision.** Class-incremental learning is active: exemplar replay (Rebuffi et al. 2017; Rolnick et al. 2019; Buzzega et al. 2020; Caccia et al. 2022), distillation (Li and Hoiem 2018), regularization (Kirkpatrick et al. 2017; Zenke, Poole, and Ganguli 2017; Aljundi et al. 2018), and expansion or expert-routing (Aljundi, Chakravarty, and Tuytelaars 2017; Erden, Gasmi, and Faltings 2025); see (De Lange et al. 2022; Masana et al. 2023; van de Ven, Soures, and Kudithipudi 2025) for surveys. Each family buys retention at the cost of an assumption online learning does not grant: replay

stores and re-presents past data; expansion isolates per-task sub-models behind a routing signal. Distillation and regularization are the mildest case and worth stating precisely, since our own measurements sharpen it (Appendix J): they do not require *task boundaries* as such—we find that whether consolidation is aligned with the class changes or deliberately out of phase with them makes no difference—but they do require a *consolidation schedule*, some designated moment at which the parameters currently held become the ones to protect, and they fail at both extremes of it. They are also acutely sensitive to penalty strength, being numerically inert at the values usually quoted for larger models.

The dominant recent direction is prompt-based continual learning over a frozen pretrained backbone (Wang et al. 2022b,a; Smith et al. 2023), which leads current class-incremental benchmarks. We do not compare against it, and the reason is not incidental: it presupposes large-scale pretraining, and therefore supplies from the outset a representation of the kind whose *formation* from a stream is what this work studies. A from-scratch setting is not a limitation of that line of work but a different question from the one it answers.

Our experiments make the comparison along the axes those assumptions tend to hide: behavior within a cycle, behavior in the early cycles, and what must be stored.

## R Model Growth and Turnover in Full

This appendix gives the full form of the argument summarized in Sec. 7.

Because the model *grows* its own structure rather than filling a fixed architecture, its size is an observable of learning, not a hyperparameter (Fig. 14). The learned structure grows fast while classes are novel ($\sim 21$ csvs after cycle 0, $\sim 134$ after cycle 5) and then saturates, plateauing around $\sim 187$ csvs from cycle $\sim$11 onward—the model does not grow without bound, even though novel samples keep arriving at every step and none is ever re-presented (Table 12 in the appendix gives the counts by cycle, split by polarity). This is worth checking rather than assuming, since a learner that adds structure in response to what it observes has no built-in ceiling and unbounded growth is the obvious failure mode. What the plateau shows is that the system does not *retain* the structure it keeps generating when what it already has accounts for what it is seeing: it continues to propose—which is what leaves it able to capture a genuine novelty—while retiring proposals that earn nothing, so the population settles instead of accumulating.

The plateau should not be read as a finished, static model. It is an *equilibrium*: the number stops changing while its *membership* does not. Variation continues to propose structure for whatever each new sample raises, and refinement, removal, and reintegration continue to retire whatever fails to reach significance, so a portion of the population at any moment consists of csvs that are formed and later discarded rather than kept. Some of this turnover is directly measurable: between consecutive checkpoints in the plateau the count changes by 7.7 csvs on average, and by as much as 29, while the level itself stays near 174. That figure is only a *lower bound* on the activity, however, and understates it substantially, because it records the *net* change between two checkpoints. Formation and retirement proceed concurrently, so a csv created and another removed in the same interval cancel in the count and leave no trace in the fluctuation at all—a population could be turning over completely while the number never moved. What is needed alongside it is a measure of how much of the population is *long-lived*, one that does not depend on net change; the maturity filter below supplies exactly that, by counting how many csvs have accumulated a given number of lifetime observations and asking what is lost by consulting only those.

This dynamism, and the transient redundancy that comes with it, is a design property rather than an inefficiency to be tuned away. The learner has no way of knowing that the stream will keep presenting the same ten classes; that is our knowledge of the experiment, not information available to it. A system that stopped proposing new structure once its size stabilized would be committing to the assumption that nothing further will ever need to be represented, and would be unable to react if the stream did in fact turn to something new. The continued churn is precisely the capacity to notice novelty, held in reserve; on a stationary class population it shows up as turnover around a fixed level, and on a non-stationary one it is what would allow the model to grow again.

Negative (suppressive) structure comes to dominate (105 negative vs. 82 positive at cycle 19), reflecting that class boundaries, not class prototypes, are where structure keeps being demanded; the no-suppressor variant makes the split visible directly, saturating near a third of the baseline's population (Fig. 14). That variant also shows that this particular task can be solved with a much smaller model: suppressing their formation altogether costs no measurable accuracy here (Appendix E, Table 2). We do not adopt it as our main configuration, because "suppressive structure is dispensable" is a property of *this* data rather than an assumption the learner could safely make in general—it forms such structure because it is accounting for what it has seen, not because it has been told what will turn out to matter. Notably, accuracy plateaus several cycles *before* size does (cycle $\sim$5 vs. $\sim$11): the late-added structure does not carry the discrimination (Appendix T). This is expected rather than surprising, since the learner has no access to accuracy and is not optimizing it. What it does is account for the statistically significant relationships it encounters, as completely as it can and with as little structure as it can—so it continues to add structure for regularities that are *real* but redundant for telling the ten classes apart.

**How much of the population is load-bearing.** The readout lets us measure this split directly, and the answer sharpens the picture above. We score the same trained models while ignoring every csv with fewer than $M$ lifetime presence observations, varying $M$ and changing nothing else (ten seeds; Table 13 in the appendix). Accuracy is flat throughout: 0.8735 at $M$=0, 0.8790 at $M$=10, and 0.8735 again at $M$=40, every deviation well inside the $\pm$0.018 spread across seeds. What changes is how much of the model is consulted—at $M$=40 only 92 of 187 csvs remain. **Half the**

Table 11: Final per-class held-out accuracy (baseline, ten-seed mean).

| Class | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | **avg** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Acc | .87 | .96 | .90 | .80 | .63 | .93 | .93 | .94 | .85 | .95 | **.874** |

Table 12: Model size (csv count) by cycle: total, positive/negative split, and the count surviving read-out maturity gates $M{\geq}5$ and $M{\geq}10$.

| cycle | total | pos | neg | $M{\geq}5$ | $M{\geq}10$ |
|---|---|---|---|---|---|
| 0 | 21.2 | 16.0 | 5.2 | 8.5 | 3.0 |
| 5 | 134.3 | 62.8 | 71.5 | 95.8 | 71.5 |
| 10 | 168.1 | 72.1 | 96.0 | 130.1 | 113.4 |
| 15 | 177.1 | 78.7 | 98.4 | 149.8 | 138.5 |
| 19 | 186.7 | 81.7 | 105.0 | 154.4 | 143.8 |

**learned population can be discarded at read-out time with no loss of accuracy whatsoever.**

Taken together with the turnover measured above, this identifies what that half *is*. The csvs that fall below the maturity threshold are those with few lifetime observations—the recently formed and the rarely satisfied—which is the transient population that the equilibrium consists of. So the $\sim 187$ csvs are not 187 pieces of predictive structure: they are a persistent, mature core that carries essentially all of the discrimination, plus a comparable body of provisional structure that is being formed and retired around it and contributes nothing to recognizing these ten classes. The two measurements are the same fact seen from two directions—one in time, as churn at a stable level; one in a single frozen model, as a large sub-population whose removal changes nothing. Read together, the two measurements also close the argument of the preceding paragraph. The net fluctuation could not see structure that is created and retired in equal measure; the maturity count sees it directly, as the gap between the full population and the long-lived one. And the provisional body it exposes is not waste to be trimmed. Those csvs are indeed unnecessary for these ten classes—the filter identifies them as such, and discarding them costs nothing—but they are not *errors*: each was a hypothesis about a regularity the data had just presented, and the mechanism that produced it is the same one that would capture a regularity that did matter. They keep being recreated because the system has no way to know, and no basis for assuming, that nothing further will need representing. Pruning the mechanism away permanently would buy a smaller model at the cost of the system's ability to respond to anything new; filtering its products at read-out time, as here, costs nothing and leaves the capacity intact.

## S Additional Figures and Tables

These support claims made in Sec. 7; each is referenced from the main text at the point it bears on.

Table 13: Read-out maturity filter on the geometric read-out (ten seeds). $M$: a csv is consulted only if it has accumulated at least $M$ lifetime presence observations; nothing else changes, and no model is retrained. Accuracy is flat throughout while the consulted population halves.

| $M$ | csvs kept | kept | accuracy | vs. $M{=}0$ |
|---|---|---|---|---|
| 0 | 186.7 | 1.00 | $0.8735 \pm 0.018$ | — |
| 5 | 154.4 | 0.83 | $0.8735 \pm 0.019$ | $+0.000$ |
| 10 | 143.8 | 0.77 | $0.8790 \pm 0.021$ | $+0.006$ |
| 20 | 128.8 | 0.69 | $0.8770 \pm 0.023$ | $+0.004$ |
| 40 | **91.9** | **0.49** | $\mathbf{0.8735} \pm 0.021$ | $+0.000$ |

## T Full Experimental Tables

Accuracy plateaus by cycle $\sim$5 while size saturates only by cycle $\sim$11: the discriminative content is concentrated in the mature core (Sec. 7), as the maturity measurement below makes precise.

Table 13 gives the measurement discussed in Sec. 7. Every value lies within one standard deviation of $M{=}0$, so the filter costs nothing across the whole range; at $M{=}40$ fewer than half the csvs are consulted and accuracy is exactly what it was with all of them. Since low-maturity csvs are the recently formed and the rarely satisfied, this is the static counterpart of the turnover reported in Sec. 7: the population divides into a mature core that carries the discrimination and a comparable transient body that does not.

The filter is also usable rather than merely diagnostic. Because it is a read-out policy and touches nothing the learner does, a maturity threshold can be fixed in advance and applied whenever the model is consulted, which yields a model that is smaller by the transient fraction at no measured cost in accuracy—roughly half the csvs at $M{=}40$. What one gives up is not accuracy on the classes already learned but the *readiness* the transient population represents: those csvs are the ones a genuinely new regularity would be picked up by, and a threshold set high enough excludes them from being consulted while they remain immature.

## U Scaling

Three scaling questions are separable, and the paper answers two of them. The first is whether structure grows without bound as *novel samples* keep arriving: it does not, saturating near 187 csvs while samples continue to arrive and none is re-presented (Appendix R). The second, addressed here, is how size and per-sample cost grow with the *number of classes*. The third—growth with the complexity of the input itself, richer images or a larger feature vocabulary—is not addressed by this work and would require a different dataset; we state it as untested rather than leave it implied by the first two.

**What was measured.** We reran the identical learner and stream with the class count varied, the classes drawn at random per seed (four seeds each). For each finished model we then replayed the held-out set and counted the calls to the single matcher invocation in the read-out. This is the whole of the expensive work: the read-out loop walks every csv, but for

Table 14: Full baseline comparison (ten seeds), grouped by what each method stores. *Final*: end-of-cycle accuracy after cycle 19. *Early*: mean over cycles 0–2. *WCR*: within-cycle retention at the end of the cycle, as in Table 1. Regularisation and replay methods are at their best configuration over the sweeps in Appendix J; the expansion (Expert-AEC) is swept over its spawn threshold $k$, higher $k$ spawning fewer experts (count in *Notes*; "failed" = hit the 30-expert cap). Two things stand out: SI and MAS reach our accuracy while retaining a quarter to a half of what we do, and storage buys retention—the same method at 8 and at 80 images differs more than most methods differ from each other.

| Method | Final | Early | WCR | Notes |
|---|---|---|---|---|
| SI | .879 | .418 | .44 | no store |
| **Modeller (ours)** | **.874** | .536 | **.95** | no store |
| MAS | .867 | .233 | .24 | no store |
| LwF | .798 | .187 | .19 | no store |
| Plain NN ($r$=0) | .661 | .158 | .16 | no store |
| Expert-AEC $k$=2 | .607 | .198 | .75 | 9.2 exp. |
| Expert-AEC $k$=1 | .482 | .176 | .71 | 17.2 exp. |
| CNN | .328 | .105 | .11 | unstable |
| Expert-AEC $k$=0.5 | .294 | .163 | .66 | 27.5 exp. |
| Expert-AEC $k$=0 | .218 | .170 | .65 | 31 exp., failed |
| Expert-AEC (bs 1) | .184 | .184 | .90 | 31 exp., failed |
| DER++ | .810 | .585 | .67 | 8 img |
| Naive replay $r$=0.8 | .760 | .500 | .52 | 8 img |
| Naive replay $r$=0.5 | .749 | .442 | .46 | 5 img |
| Naive replay $r$=0.2 | .717 | .309 | .32 | 2 img |
| ER-ACE | .399 | .383 | .81 | 8 img |
| DER++ | .912 | .748 | .91 | 80 img |
| ER-ACE | .825 | .714 | .98 | 80 img |

one whose anchors are unbound—whose downstream chain did not match the observation—it performs only a constant-time test and skips before any matching. The integration of the surviving detectors' statistics is arithmetic on scalars and is negligible beside it. **Learning needs no separate measurement**: it reaches csvs by the same chain-aware pass and is cut by the same condition, so a csv skipped at read-out is skipped when learning from that sample too.

We measure traversal rather than size because that is where the cost is. A csv that is never reached costs storage and nothing further; the computation is entirely in those that are matched, and the model may in principle be arbitrarily large without that changing. We do not treat the storage itself as the pressing question, and not because it is small. Neural networks are known to be substantially overparameterised for the tasks they solve, and that surplus is not incidental but appears to be necessary for them to be trainable at all; a learner that instead retains only what the evidence requires, and retires what it does not (Appendix R), is attempting the opposite discipline by construction. We would therefore not expect size to be more of a problem here than it already is for the models this work is compared against. Establishing that would require tasks far more complex than this design can currently address, so we offer it as an argument about where the burden lies rather than as an experimental claim.

Table 15: Final per-class accuracy (mean over seeds): our system vs. representative baselines.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Ours | .87 | .96 | .90 | .80 | .63 | .93 | .93 | .94 | .85 | .95 |
| Replay 0.8 | .83 | .91 | .60 | .42 | .69 | .62 | .90 | .80 | .94 | .89 |
| Plain NN | .77 | .90 | .36 | .11 | .69 | .33 | .89 | .81 | .92 | .84 |
| CNN | .15 | .33 | .17 | .12 | .28 | .27 | .58 | .57 | .38 | .45 |

Table 16: Scaling with the number of classes (four seeds each; classes drawn at random per seed). *Matched* is the number of csvs that reach the matcher for one observation, *range* its per-sample minimum and maximum across all seeds, and *frac* the mean fraction of the model matched. The two-class row measures which digit pair was drawn rather than class-count scaling—its four seeds gave 5, 16, 31 and 57 csvs—and should be read as such.

| Classes | csvs | Matched | Range | Frac | Accuracy |
|---|---|---|---|---|---|
| 2 | 27±23 | 23±18 | 3–57 | 0.88 | 0.869±0.072 |
| 4 | 68±15 | 40±7 | 10–74 | 0.59 | 0.878±0.057 |
| 6 | 124±2 | 58±1 | 16–123 | 0.47 | 0.833±0.065 |
| 8 | 145±12 | 68±10 | 20–150 | 0.47 | 0.842±0.025 |
| 10 | 191±18 | 83±5 | 24–199 | 0.43 | 0.874±0.014 |

**Cost grows more slowly than the model.** From four to ten classes—2.5× the classes—the model grows 2.80× while the work per observation grows 2.09×. The fraction of the model matched falls from 0.59 to 0.43 and is flat from six classes on (0.47, 0.47, 0.43, with standard deviations of 0.02–0.03 across seeds). Roughly half the model is skipped before any matching, and that proportion holds as the model grows. Accuracy does not degrade as classes are added (0.869, 0.878, 0.833, 0.842, 0.874), so the flat cost is not bought by the model quietly getting worse. At ten classes these runs give 0.874±0.014 over four seeds against the 0.874±0.018 of Table 1 over ten—an incidental but useful reproduction of the main result through a separately written harness. The per-sample maximum, however, approaches the model size (199 of 216 in the largest run): the mean is the right summary, but the worst case is near-full traversal.

**Why the fraction falls, and why the count is larger than it needs to be.** Both follow from one property of the current design. Each csv conditions a *single* target, so structure shared between classes cannot be referenced from more than one place and is instead rebuilt separately under each class that uses it. For any one observation the copies belonging to other classes fail the anchor test and are cut, so as classes are added a growing share of the model is about classes the observation is not—which is exactly why the matched fraction falls. The same duplication inflates the absolute count.

This identifies the lever. Allowing a csv to condition several immediate targets—a directed acyclic organisation rather than a per-target tree—lets each distinct sub-pattern exist once and be referenced by many parents, which is the minimal shared representation, and it is consistent with the locality the learner already respects (connections to immedi-

ate targets, not to a whole chain). The expected effect is not a faster procedure but a smaller model: fewer CSVs in total and fewer reached per observation. Note that the fraction would likely *rise* rather than fall, since what survives is shared structure that is relevant more often—which is a further reason to treat the absolute count, not the ratio, as the quantity of interest. We flag one condition: the representational change is necessary but not sufficient on its own, since refinement must also recognise an existing sub-pattern and connect to it rather than spawn a fresh copy. We have not implemented this, and report it as the identified direction rather than a measured improvement.

**The traversal is unprioritised.** A second lever is untouched. Within what the cut leaves, the traversal is exhaustive and unordered: wherever a CSV's anchors are bound it is matched, in no particular sequence, with no attempt to visit first those most likely to match or those carrying the most weight in the decision. Nothing in the design requires this. Each CSV already maintains the statistics such an ordering would need—how often it has been satisfied, and how specific it is to its own target—so the traversal could be ordered by expected contribution and stopped once the candidates that remain cannot change the prediction, in the manner of a best-first search with a bound. We have implemented no such scheme, and the consequence is worth stating precisely: the 83 of 191 reported above is what an *exhaustive* traversal costs, and is therefore the pessimistic case rather than the operating cost of a read-out that ordered its work. Together with the sharing above, the two levers act on different terms—one on how many CSVs exist, the other on how many of them need to be visited.

**An independent bound at read-out.** Two further results already bound the cost of consulting the model. Ignoring every CSV with fewer than $M$ lifetime presence observations leaves accuracy flat while removing half the population at $M{=}40$ (Sec. 7), and because that is a read-out policy touching nothing the learner does, it can be fixed in advance. And the profile in Appendix V shows wall-clock dominated by evaluation—matching CSVs against held-out images—rather than by learning, so the quantity measured here is the one that governs runtime.

## V Reproducibility

We record the full configuration used for every reported Modeller result.

**Data.** We use MNIST (LeCun et al. 1998), filtered for topological consistency, at native $28 \times 28$ resolution and binarized at intensity threshold $0.5$; $59{,}595$ of the $70{,}000$ images are kept.

*Why filter.* The filtering is a scope simplification, not a requirement of the method. The present feature construction reads structure off clean contours and does not model topological *gaps* (breaks or disconnections within a stroke), just as it does not model smooth intensity transitions; both are handleable in principle by the same change-point construction—gaps as additional structure, transitions as higher orders of change (Sec. 4)—but incorporating them would be a lengthy extension orthogonal to what this paper tests. We therefore exclude the affected samples and keep only clean, single-structure digits with the expected loops. Crucially, to keep the comparison fair, *all* baselines are trained and evaluated on this same filtered subset; no method sees data the others do not.

*The filtering criterion.* Topology is measured through the model's own image-to-network front end, so that a loop which does not close under the model's binarization genuinely is not there for the model and the sample is dropped. An image is binarized (no Gaussian smoothing, threshold $0.5$) and its contours extracted with a nesting hierarchy, discarding contours shorter than $10$ pixels so that antialiasing speckle is not counted; contours at even nesting depth are outer boundaries (connected components) and those at odd depth are holes. A sample is kept if and only if

1. it has exactly *one* connected component—more than one outer contour means disconnected strokes, and the sample is dropped; *and*
2. its hole count matches the digit, for those digits that are expected to enclose regions: $0 \to 1$, $6 \to 1$, $9 \to 1$, $8 \to 2$. Digits $1, 2, 3, 4, 5, 7$ carry no hole constraint and may have any number.

**Stream and evaluation.** Ten classes (0–9) in fixed order; $20$ cycles; $5$ samples per class per cycle from disjoint indices; one learning step per sample; no re-presentation, replay, or task/cycle-boundary signal. Held-out evaluation uses $20$ images per class at a fixed disjoint offset. All Modeller results average $10$ seeds (0–9), each fixing the sample draw and the model's internal random state. A run is therefore exactly reproducible from its seed, and we verified this rather than assuming it: retraining all ten seeds from scratch reproduced both the final accuracy and the final CSV count of the original run in every case ($10/10$ exact matches). Results are additionally invariant to the order in which internal identifiers are assigned. Trained models can optionally be serialized (per cycle, per class block, or at the end of a run) so that read-out variants such as the maturity filter of Table 13 can be re-scored without retraining.

**Representation.** Contours are extracted from the binarized image (outer boundaries and holes distinguished); each is traversed and orientation-change points read off the raw contour (typed by change axis, direction, convexity; no polygonal approximation). Nodes are wired into *contour* and *spatial* layers, horizontal and vertical variants, and coarsened by successive pair contraction with per-level recomputation of spatial edges; the edges surviving at every level are accumulated into one augmented network. The no-multi-scale ablation uses the finest level only.

**Learning hyperparameters (fixed across all runs).** Significance threshold $\epsilon_{\text{sign}} = 0.1$; removal rate $0.5$; reintegration rate $0.5$; spawn reliability threshold $0.9$; refinement reliability threshold $0.9$; stale-statistics coefficient $0.5$; refinement match threshold $0.5$ on combined node-and-edge coverage; depth-scaled significance enabled. Matching uses the spatial multi-network matcher over the horizontal (`contour_h`,

`spatial_h`) and vertical (`contour_v`, `spatial_v`) layers.

**Read-out.** The local geometric read-out (Sec. 6) uses position weight $w_p = 3.0$ with a variance floor of $1.5$ px on the position anchors, orientation weight $w_o = 1.0$ with concentration $\kappa = 2.0$, and add-$\frac{1}{2}$ smoothing on the firing counters; a node or edge anchor contributes only once it has accumulated at least two observations. The alternative read-out of Appendix D uses the binary-integration log-odds rule (Eq. 2) with base-rate pivot, anchored chain-aware matching, and maturity filter $M = 0$ unless stated. Both are deterministic given a trained model.

**Metrics.** Three quantities are reported. *Accuracy* is the fraction of held-out images classified correctly, computed per class over the twenty held-out images of that class and then averaged over the ten classes, so every class contributes equally regardless of its frequency in MNIST. *End-of-cycle* accuracy is that quantity measured at the last block of a cycle, and *Early* its mean over cycles 0–2. *Retention loss* is computed per class as the difference between that class's highest accuracy at any checkpoint and its accuracy at the end of the run, then averaged over classes—so it measures what was reached and not held, and is zero for a class that never declines. Within-cycle retention is defined in Appendix I.

**Seeds and repetitions.** Headline Modeller results and all neural baselines average ten seeds (0–9); the multi-scale ablation (Appendix G) uses five seeds per cell, the threshold sweep (Appendix M) two seeds per configuration, and the enlarged-buffer replay study (Appendix K) three seeds. Each seed fixes both the sample draw and the model's internal random state, and results are invariant to the order in which internal identifiers are assigned.

**Baselines.** Run under the identical stream and held-out sets. Plain/replay: a fully connected network (Flatten–Dense256–Dense256–Dense10, softmax), batch 10, 20 epochs per class block, replay buffer ratios $r \in \{0, 0.2, 0.5, 0.8\}$. CNN: two $3 \times 3$ conv–pool blocks of 32 filters each, then Dense128–Dense10. Expert-AEC: per expert a predictor (Dense128) and an autoencoder (Dense64, sigmoid output), a new expert spawned when the minimum reconstruction error exceeds a $k$-scaled threshold, capped at 30 experts; configurations vary $k$ and batch size. All hidden units are ReLU. Every classifier is trained with Adam at its framework defaults (learning rate $10^{-3}$) under sparse categorical cross-entropy, and every autoencoder with Adam under mean squared error; each presented batch is one gradient step, so the 20 epochs per class block are 20 such steps. Baselines run in an isolated TensorFlow/Keras environment on CPU.

**Compute.** All experiments run on a single machine: a 6-core (12-thread) Intel Core i7-9750H at 2.6 GHz with 16 GB of RAM, under macOS 15.7. No GPU is used by our system at any point. Seeds are run as independent single-threaded processes (BLAS threading pinned to one thread per process) with several in parallel.

Two software environments are used, and the separation is itself informative: our system requires no deep-learning framework, running on Python 3.13 with NumPy 2.2, NetworkX 3.5, SciPy 1.16 and OpenCV 4.12 (the last only for contour extraction), while TensorFlow 2.16 / Keras 3.15 (Python 3.11, NumPy 1.26) appear *only* in the neural baselines.

For wall-clock, a single twenty-cycle seed takes ∼7–8 minutes when the model is evaluated once at the end, and ∼55–60 minutes when it is evaluated after every one of the 200 class blocks, as in the retention matrices. The difference makes the cost profile clear: the 1,000 learning steps are the cheap part, and wall-clock is dominated by *evaluation*—that is, by repeatedly matching every csv against every held-out image—rather than by learning.

**Algorithm 3** One learning step (Modeller with network refinement)

**Require:** observed multi-scale network $O$; target states $\{S_T\}$

reset each csv's per-step fields, and apply the target states $\{S_T\}$
{**traversal** — upstream from the targets, taken in order of activity}
**while** some csv is still to be processed **do**
  take the next csv $C$ on that path
  match $C$'s source against $O$, its anchors pinned to where its downstream matched (Alg. 2)
  resolve $C$'s state from whether its sources are satisfied and from its target's state
  **if** $C$ matched only in part **then**
    refine $C$ and spawn the upstream carrying its residual (Alg. 4), and add that upstream to the traversal {so it is processed within this same step}
  **end if**
**end while**
{**generation** — only now, with the whole frontier resolved}
**for all** events left unexplained by the traversal **do**
  for an active target or csv with no active positive conditioner, create a positive csv whose source is $O$ itself
  for an inactive csv with no active negative conditioner, create a negative one likewise
**end for**
fold this step into every csv's counts, the just-created ones included, so that their statistics begin from this observation
{**structure dynamics** — two mirrored bands on $P(\mathrm{SS}(C) \mid I(T))$, each sampled per step, neither a hard test}
**removal:** if $P(\mathrm{SS}(C) \mid I(T)) < \tau_{\mathrm{rem}}$, discard the conditioner with probability $\rho\,(\tau_{\mathrm{rem}} - P(\mathrm{SS}(C) \mid I(T)))/\tau_{\mathrm{rem}}$
**reintegration:** if $P(\mathrm{SS}(C) \mid I(T)) > \tau_{\mathrm{int}}$, and the conditioner is positive and conditions another csv, merge it into that downstream with probability $\rho\,(P(\mathrm{SS}(C) \mid I(T)) - \tau_{\mathrm{int}})/(1 - \tau_{\mathrm{int}})$
in both: never act on a csv on its birth step, and down-weight one whose relation gained no evidence this step

**Algorithm 4** Refinement and upstream spawning for one csv

**Require:** csv $C$ with source $S$; observation $O$; the best match $\phi$ of $S$ into $O$; coverage threshold $\tau$

**if** $\mathrm{degree}(\phi) < \tau$ **then**
  treat $C$ as absent and leave $S$ unchanged; **return** {a poor match must not strip structure}
**end if**
$(R, A) \leftarrow$ the part of $S$ that $\phi$ left unplaced, together with the placed nodes whose edges reach into it {$R$: residual nodes and edges, identities preserved; $A$: the boundary nodes, to become anchors}
$S \leftarrow \phi(S)$ {refinement: $S$ shrinks to the placed part, which $\phi$ now matches fully}
**if** $R$ is non-empty **then**
  create $U$, a positive csv conditioning $C$, with source $R \cup A$ and $A$ flagged as anchors
  $U$ inherits $C$'s presence counts, so its statistics begin from $C$'s history rather than from nothing
  re-parent onto $U$ every conditioner that previously conditioned $C$ {they conditioned the larger configuration $U$ now represents}
  transfer ownership of $R$'s nodes to $U$; the anchors $A$ stay owned by $C$
**end if**

**Algorithm 5** Read-out for one observation

**Require:** trained model; observation $O$; weights $w_p, w_o, \kappa$

$s_k \leftarrow 0$ for every class $k$
compute the presence of every csv against $O$ (Alg. 2, downstream-first)
**for all** csvs $c$ **do**
  $k_c \leftarrow$ class of $c$'s immediate target, resolved one hop at a time
  **if** $k_c$ is undefined **then** skip $c$
  $P_c^o, P_c^f \leftarrow$ smoothed own-class and aggregate-foreign firing rates
  **if** $c$ is not fully present **then**
    $s_{k_c}$ += $\log \frac{1-P_c^o}{1-P_c^f}$ {absence counts as evidence}
  **else**
    $s_{k_c}$ += $\log \frac{P_c^o}{P_c^f}$
    **for all** source nodes $n$ of $c$, placed at $p_n$ **do**
      **if** both the own-class and pooled anchors of $n$ have $\geq 2$ observations **then** $s_{k_c}$ += $w_p\big[\ell(p_n; a^o_{c,n}) - \ell(p_n; a^{\mathrm{pool}}_{c,n})\big]$
    **end for**
    **for all** source edges $e$ of $c$, placed at angle $\theta_e$ **do**
      **if** both orientation anchors of $e$ have $\geq 2$ observations **then** $s_{k_c}$ += $w_o\kappa\big[\cos 2\Delta^o_{c,e} - \cos 2\Delta^{\mathrm{pool}}_{c,e}\big]$
    **end for**
  **end if**
**end for**
**return** $\arg\max_k s_k$

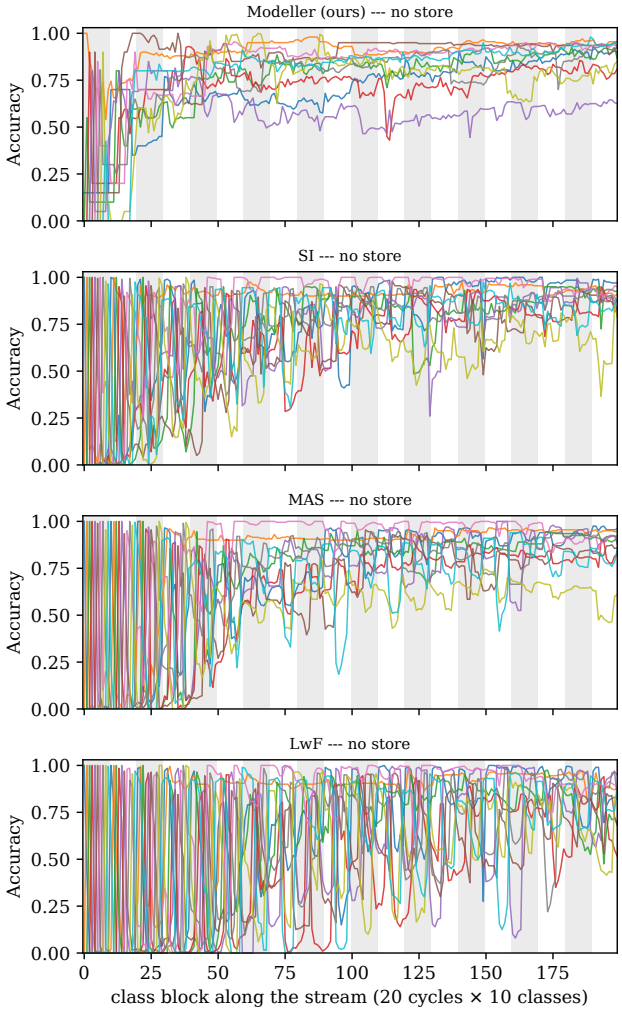


Figure 6: Equal final accuracy, opposite behaviour. Per-class accuracy along the stream (ten-seed means; axis convention as in Fig. 4) for the methods that store no past data. SI and MAS end the run at accuracies indistinguishable from ours (Table 1), but their per-class curves collapse and recover repeatedly—each class is given up and relearned every cycle—whereas ours are held. This is also why final accuracy converges: over twenty cycles of re-exposure the procedure becomes stochastic gradient descent with a slow time constant, under which any method that learns at all approaches the same ceiling. Retention is what still distinguishes them at that horizon.

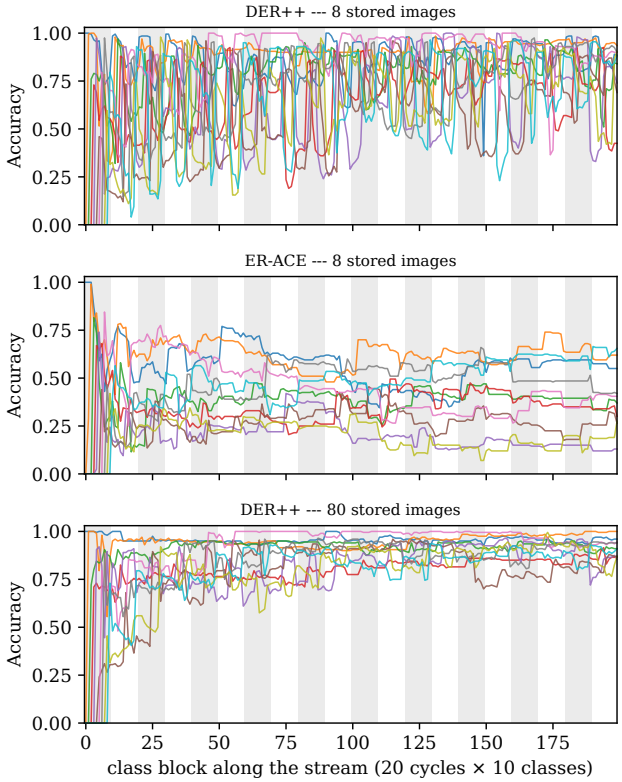


Figure 7: Modern replay at the two buffer capacities. At 8 stored images DER++ still oscillates. At 80 the oscillation is largely gone and it exceeds us on both accuracy and retention—but at that capacity each update is dominated by stored data rather than by the arriving sample, so the procedure is close to ordinary training on a growing pooled batch and is no longer a continual-learning result in the sense at issue here. ER-ACE holds what it has at either capacity but learns little at the small one, for the reason given in the text.

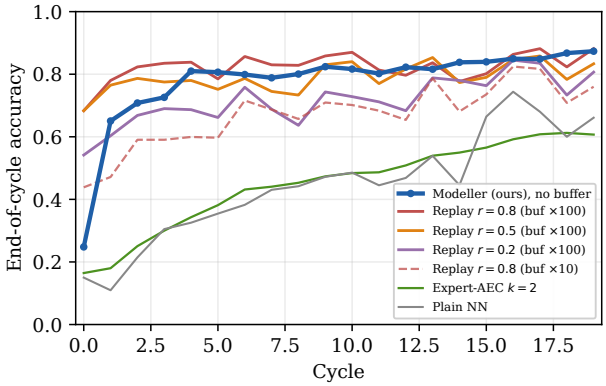


Figure 8: End-of-cycle accuracy with the enlarged buffers ($\times 100$ scale), against the Modeller, the original $\times 10$ setting (dashed), the expert model, and the plain network. Larger buffers lift replay to the Modeller's band, but only late; the Modeller leads the early cycles while storing nothing.

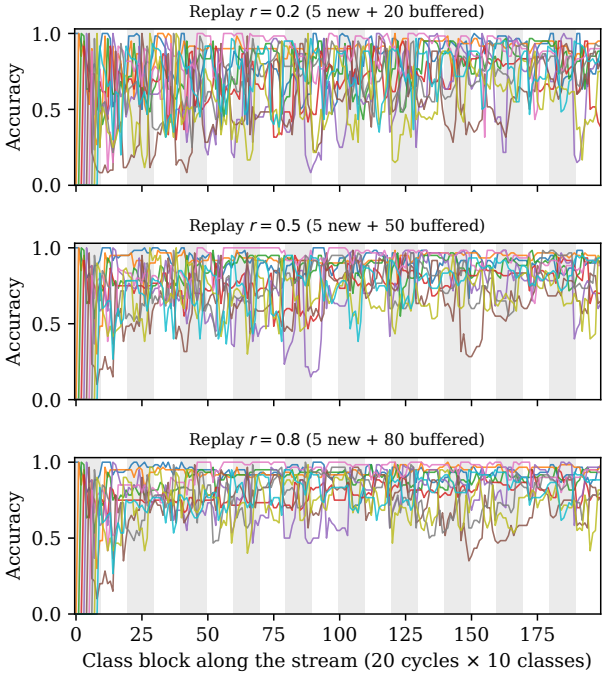


Figure 9: Per-class accuracy along the stream with the enlarged buffers (ten-seed means; axis convention as in Fig. 4). With 20–80 stored examples the oscillation of the small-buffer settings is largely suppressed—destructive adaptation is bought off with stored data, at up to 80% of each class's total budget.

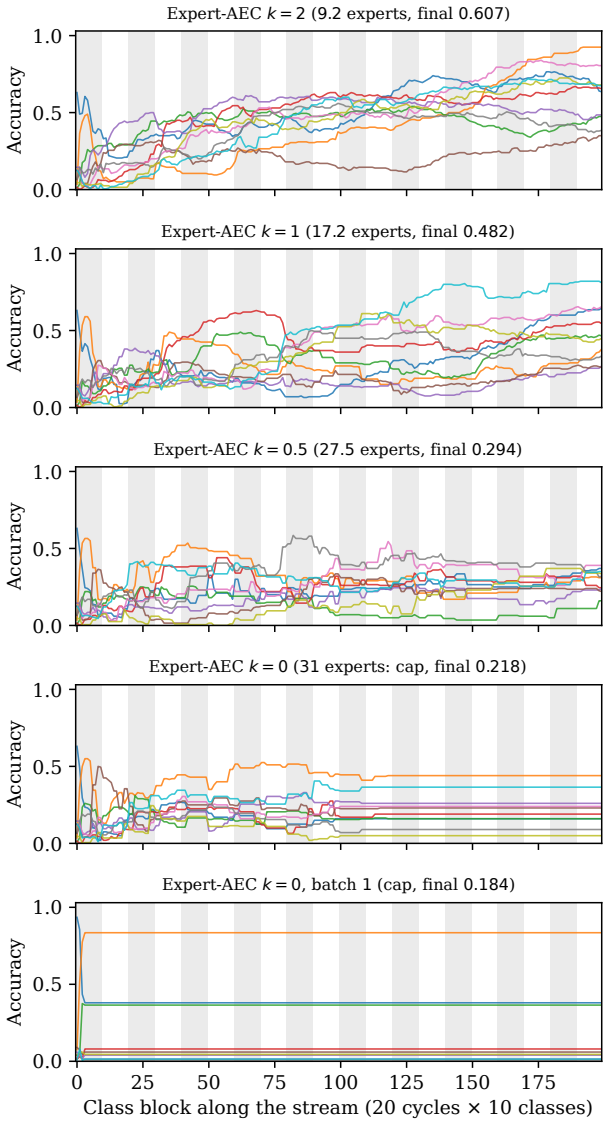


Figure 10: Per-class accuracy along the stream for every expert-routing configuration (ten-seed means; axis convention as in Fig. 4). Curves are smooth rather than oscillating—an expert is reached almost only by its own class—but plateau far below, capped by test-time routing; the two capped settings (bottom) stop improving entirely.

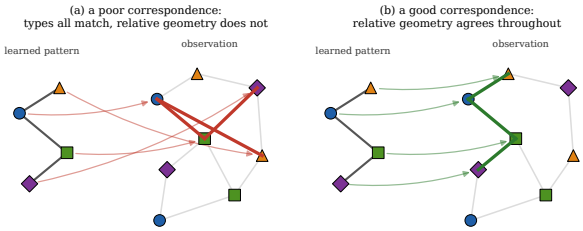


Figure 11: The correspondence problem. A learned pattern (left in each panel) must be placed within an observation (right) whose nodes carry the same feature types; marker shape and colour denote type. **(a)** A poor correspondence: every pattern node is matched to an observation node of the correct type, yet the placed configuration (red) bears no resemblance to the pattern's own geometry—the relative displacements disagree. **(b)** A good correspondence: the placed configuration (green) reproduces the pattern's arrangement, so every relation agrees. Type compatibility alone does not distinguish these; agreement of the relations does.

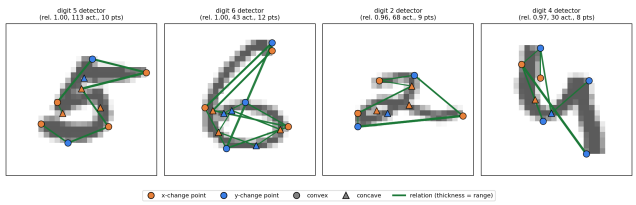

Figure 12: The learned models are their own explanation. Mature csvs for four classes, each overlaid on a held-out image it matches. Nodes are typed orientation-change points landing on meaningful loci; green edges are retained cross-scale relations (thickness $\propto$ range), coarse ones carrying global layout and fine ones local detail. No post-hoc attribution is involved: this *is* the learned representation.

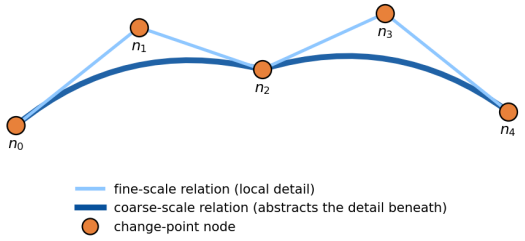


Figure 13: The core idea in miniature. Short-range relations link neighbouring change points (thin); long-range relations span several of them (thick). Both coexist over the same nodes in one network, so relations of different extent are simultaneously present and interconnected, with no ordering or priority between them.

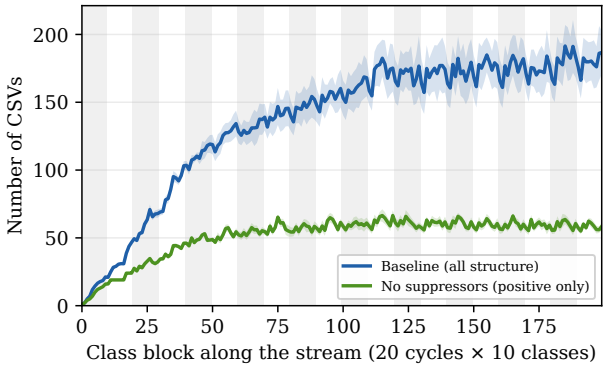


Figure 14: The model grows and then saturates. csv count along the stream (ten seeds; line = mean, band = $\pm 1$ s.d.; alternating shading marks cycles). Growth is steep while classes are novel, decelerates from cycle $\sim$5, and levels off around $\sim$185 csvs—bounded, not unbounded, under a stationary class population. The no-suppressor variant (green) isolates how much of the population is suppressive: it saturates near 60, so roughly two-thirds of the structure the baseline forms is negative, boundary-carving structure.

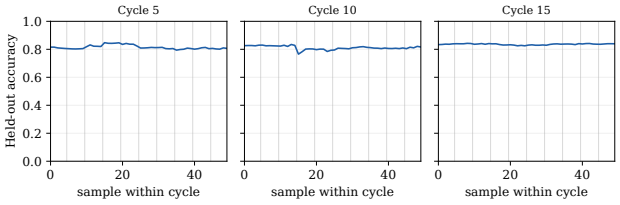


Figure 15: Per-sample learning progression within a cycle, for the baseline configuration at cycles 5, 10 and 15. Those three were fixed in advance rather than selected afterwards; evaluating after every sample of every cycle would multiply an already evaluation-dominated cost (Appendix V). Accuracy is recorded after *each individual sample* rather than only at block ends—fifty checkpoints per cycle instead of ten—and each point is the accuracy over the entire held-out set, *averaged across all ten classes*; ten-seed means. The $x$-axis is the sample's position within the cycle, with thin verticals at the ten class-block boundaries. Accuracy rises as each block's five samples are integrated and does not fall back as the later blocks train, so the per-block sampling used elsewhere in the paper conceals nothing between its checkpoints—the sample-level counterpart of Sec. 7.