Soft Computing Fusion with Applications

www.scfa.reapress.com

Soft. Comput. Fusion. Appl. Vol. 3, No. 4 (2026) 272–283.

Paper Type: Original Article

Replay-Based Continual Learning on Sequential Digit Tasks: Buffer Size, Task Similarity, and the Effect of Interleaving

Anna L. Marchetti[1,*], Jan H. Novak[2], Hiroshi Tanabe[3]

[1] Dipartimento di Informatica, Universita di Pisa, Pisa, Italy; a.marchetti@unipi.it.

[2] Faculty of Information Technology, Czech Technical University in Prague, Praha, Czech Republic; j.novak@gmail.com.

[3] RIKEN Center for Computational Science, Kobe, Japan; hitanabe@toku.jp.

Citation:

Received: 13 April 2026 *Revised: 27 July 2026* *Accepted: 06 September 2026*	Marchetti, A. L., Novak, J. H., & Tanabe, H. (2026). Replay-based continual learning on sequential digit tasks: Buffer size, task similarity, and the effect of interleaving. *Soft Computing Fusion with Applications, 3*(4), 272-283.

Abstract

We study replay-based mitigation of forgetting on five binary digit tasks using the 1797-sample subset of UCI Optical Recognition of Handwritten Digits distributed by scikit-learn. A shared 64-16-16 ReLU network with task-specific sigmoid heads is trained under sequential learning, Experience Replay (ER), past-task-balanced Interleaved Replay (IR), and joint training with all tasks available from the outset. Reported means over ten seeds give Backward Transfer (BWT) -0.170 and final accuracy 0.861 for naive training, versus joint-reference accuracy 0.996. In the natural-layout ER condition, nominal per-class capacities 25 to 200 improve BWT from -0.035 to -0.010 and accuracy from 0.969 to 0.988; capacity 200 exceeds the available training examples per class and stores the full observed class data. IR does not significantly outperform ER in BWT at any capacity by the reported unadjusted tests ($p \geq 0.058$). ER and IR also differ in current-task inclusion and expected past-task rehearsal dose, so this contrast does not isolate temporal interleaving alone. Per-task feature permutations reduce measured mean-image correlation and are associated with less forgetting in several exploratory contrasts. The twenty-six reported tests are unadjusted; applying Holm to their rounded p-values leaves none below 0.05. Results characterize one small task-incremental benchmark, and mechanism, execution, and reproducibility claims require the original code and per-run outputs.

Keywords: Continual learning, Catastrophic forgetting, Experience replay, Complementary learning systems, Task similarity, Optdigits.

1 | Introduction

Artificial learning systems are increasingly expected to acquire knowledge sequentially: new users appear, sensor calibrations drift, and novel categories must be learned while earlier ones remain in service. Networks trained with gradient descent handle this demand poorly. When the training distribution shifts from one task

Corresponding Author: a.marchetti@unipi.it

https://doi.org/10.22105/scfa.v3i4.116

Licensee System Analytics. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0).

to the next, the updates that fit the new task partially overwrite the parameters that encoded the previous one, so that competence acquired earlier degrades rapidly. This failure mode, known as catastrophic interference, was demonstrated convincingly in the early connectionist literature: McCloskey and Cohen [1] showed that a network trained by back-propagation of errors [2] on one paired-associate list and then a second lost almost all retention of the first, and they argued that the problem was intrinsic to the way distributed representations are updated rather than an artifact of a particular architecture.

The most influential theoretical account of how biological brains sidestep this problem is the Complementary Learning Systems (CLS) framework of McClelland, McNaughton, and O'Reilly [3]. CLS theory posits two interacting memories: a fast, sparse hippocampal store that binds the specifics of new episodes, and a slow neocortical system that acquires the statistical structure of the environment through gradual, interleaved re-exposure to old and new material. The computational translation of this account is rehearsal: retain some record of past experience and re-present it, interleaved with new data, while learning continues. Robins [4] demonstrated that rehearsal with a small store of true exemplars — and, more remarkably, pseudorehearsal with internally generated pseudo-exemplars — substantially reduces catastrophic forgetting in small networks. Rehearsal-style buffers have since become a standard tool, most visibly in reinforcement learning, where Experience Replay (ER) stabilizes learning from non-stationary streams [5].

Contemporary continual learning research has broadened this toolkit considerably. Regularization approaches such as elastic weight consolidation identify parameters important to old tasks and penalize their modification [6]; architectural approaches allocate or grow task-specific capacity; and replay-based approaches store, condense, or generate past data and mix it into current training. Recent surveys [7], [8] and overview chapters [9] catalogue dozens of variants and report that replay methods with even modest memory remain among the most robust baselines across benchmarks. At the same time, the empirical picture is fragmented: reported comparisons typically change several design factors at once — the amount of stored data, the rule by which stored samples are drawn into training, and the relatedness of the tasks themselves — so that the individual contribution of each factor is difficult to isolate.

The study uses the 1797-sample digit subset bundled by scikit-learn [10], [11], partitioned into five tasks discriminating digit pairs 2i and 2i+1. ER draws uniformly from a class-balanced buffer including the current task; IR uses the same buffer but samples only past tasks. Thus the compared distributions also change past-task rehearsal dose and current-task duplication. Joint training provides a future-data reference, not a mathematical upper bound. Per-task input permutations manipulate layout statistics, and ten seeded replications support conditional exploratory comparisons.

The design varies nominal buffer capacity, replay sampling distribution and input layout. Shared splits, initialization and buffer snapshots are reported controls, subject to code confirmation. The evidence concerns aggregate dose response and exploratory layout effects; it neither proves a pure scheduling effect nor establishes a biological memory mechanism.

2 | Related Work

Catastrophic interference was characterized long before it became a benchmarking concern. McCloskey and Cohen [1] trained back-propagation networks [2] on paired-associate lists and observed that learning a second list drove retention of the first toward zero, in sharp contrast to human subjects, who showed moderate proactive and retroactive interference but no wholesale erasure. Their analysis located the problem in the overlap of internal representations: because the same hidden units participate in encoding both lists, gradients from the new task move weights that the old task depends on. Interference is therefore expected to scale with representational overlap between sequentially learned tasks — an expectation that directly motivates our third hypothesis.

The CLS framework [3] motivates retaining episodes and re-exposing a shared learner to them. Robins [4] studied literal and pseudo-rehearsal. The present ER-IR comparison varies current-task inclusion and past-

task sample dose along with task balancing, so it tests a combined replay distribution rather than pure temporal scheduling.

Regularization methods such as elastic weight consolidation protect important parameters [6]; surveys review replay and other continual-learning families [7], [8]. A recent chapter places these methods within learning and memory [9]. Reinforcement-learning replay serves additional purposes such as decorrelation [5]. This experiment investigates exemplar memory and task sampling in a supervised task-incremental network, rather than testing all rehearsal mechanisms.

Continual learning involves sequential exposure to changing task data. Active learning concerns selective acquisition of labels and need not assume a stationary pool [12]; classical pooled prediction [13] addresses another evaluation setting. These distinctions motivate specifying the task sequence and memory-access assumptions directly.

This is a controlled empirical comparison rather than a new algorithm. Capacity, current-task inclusion, past-task sampling and layout statistics are explicit factors. Shared buffer contents improve the comparison, but differing replay doses and future-data access must be accounted for before attributing effects solely to scheduling. The present review does not establish that this particular factorization has never been studied.

3 | Research Gap and Questions

The study asks how nominal capacity relates to retention, how two replay distributions differ, and how per-task layout permutations affect interference. These are operational questions in a small network; the design does not equate layout correlation with semantic task similarity or measured hidden-representation overlap.

RQ1 (capacity). How does replay buffer capacity, from 25 to 200 stored samples per class, govern Backward Transfer (BWT) and final average accuracy on sequential digit tasks?

H1. forgetting decreases monotonically — BWT rises toward zero — as capacity grows.

RQ2 (scheduling). Does distributing a fixed replay budget uniformly over past tasks (Interleaved Replay (IR)) reduce forgetting relative to drawing the same number of samples uniformly from the entire buffer (ER)?

H2. IR achieves higher BWT than same-budget ER, with the benefit expected at moderate buffer sizes (50–100 samples per class).

RQ3 (similarity). Does the degree of input-layout overlap between tasks modulate interference?

H3. streams whose tasks share a common input layout (natural digit images) suffer more interference than streams whose tasks are decorrelated by per-task feature permutations, consistent with overlap-based accounts of catastrophic interference [1], [3].

The scope is one dataset subset and a compact task-incremental network. Task identity is supplied at test time. Shared controls and paired seeds support conditional comparisons, while the full pipeline and run-level statistics require artifact confirmation.

4 | Materials and Methods

4.1 | Dataset and Task Construction

The scikit-learn load_digits benchmark contains 1797 images with 64 intensity features in an 8x8 grid and ten classes [11]. It is the test subset of the larger UCI Optical Recognition of Handwritten Digits collection, not the complete 5620-example corpus [10]. The source reports per-class counts 174-183, adjacent-pair tasks of 354-363 samples, 80/20 per-task splits with about 283-290 training and 71-73 test examples, and identical splits across conditions. Pixel intensities 0-16 are scaled to 0-1. Task identity is provided during training and evaluation.

4.2 | Network Architecture and Training Objective

The learner is a task-incremental multilayer perceptron implemented from scratch in NumPy [13]. The 64-dimensional input feeds a shared body of two fully connected hidden layers of 16 rectified-linear units each, followed by T = 5 task-specific readout heads, each a single logistic unit. Training a task activates only that task's head, so cross-task interference arises exclusively through the shared body — the component that continual learning must protect. The loss is binary cross-entropy computed on the active head; when a training batch contains replayed samples from earlier tasks, each replayed sample contributes through its own task's head while gradients still flow into the shared body. Optimization is mini-batch stochastic gradient descent with a fixed learning rate of 0.3 and a current-stream batch of 32 samples; no momentum, weight decay, or adaptive step sizes are used, keeping the dynamics transparent. Each task is trained for 1,000 gradient steps, which at the given batch size corresponds to roughly 111 epochs over the ~287 training samples of a task; this deliberately over-trained regime drives the current task to near-zero loss before the stream advances, maximizing the opportunity for interference. Weights are initialized with He scaling from a per-seed generator that is shared by all strategies, so that every strategy within a seed starts from an identical model.

4.3 | Replay Memory

The source reports class-balanced reservoirs built during the first exposure epoch, with nominal capacities 25, 50, 100, 200 per class and identical snapshot sequences for ER and IR. A reservoir stores at most min(c, n_class_train) distinct examples; with roughly 139-146 train examples per class, c=200 stores all available examples rather than 200 distinct examples. Actual retained counts and insertion timing must be reported. The standard replacement probability c/n_seen applies after capacity is reached; code must confirm that repeated epochs do not create duplicated reservoir entries.

4.4 | Continual-Learning Strategies

Naive training uses current-task batches. ER appends 32 samples drawn with replacement from the whole buffer, including the current task. IR draws 32 samples from past tasks only, balancing the chosen past task before selecting its stored example. With i completed tasks and equal full class buffers, ER's expected past-task dose is 32i/(i+1), while IR's is 32; actual partially filled buffers change the ER fraction. At the first task IR has no past examples, so its fallback and batch size require specification. Joint training draws from all tasks from the outset. Equal 5000 update counts do not imply equal sample computation or information access, and shared current-task batch sequences apply to the sequential conditions rather than joint pooled batches.

Replay supplies extra sample evaluations per update relative to naive learning. The two replay distributions alter the proportion of old and current samples, and may differ at the empty-history boundary. Joint training has access to future-task data and changes the data distribution from the start. These are treatment differences that constrain an interpretation as pure temporal ordering or equal-compute superiority.

4.5 | Task-Similarity Manipulation

Natural tasks share the canonical coordinate frame, with reported mean pairwise correlation 0.82 between task-average images. Independent fixed task permutations lower this summary to -0.002. An invertible coordinate permutation preserves each task's information, but does not eliminate every cross-task relation or guarantee identical optimization difficulty. The observed layout manipulation is not a direct measurement of gradient conflict or hidden representation overlap.

4.6 | Evaluation Protocol

After task i, A[i, j] is held-out accuracy on task j. For zero-based task indices, BWT averages final accuracy minus just-trained accuracy over the four earlier tasks, while final average accuracy averages the final row. Negative BWT indicates forgetting; zero BWT may also mask compensating task gains and losses. Above-

diagonal accuracy in non-joint runs calibrates untrained heads, whereas joint training has already accessed all tasks.

$$\mathrm{BWT} = \frac{\sum_{j=0}^{T-2}[A_{T-1,j} - A_{j,j}]}{T-1}, \qquad \mathrm{AA}_{\mathrm{final}} = \frac{\sum_{j=0}^{T-1} A_{T-1,j}}{T} \tag{1}$$

4.7 | Fairness and Controls

Reported controls use common splits, initialization and reservoir snapshots with separate per-seed generators. Update counts are shared, but sample evaluations and future-task access differ. ER-versus-IR comparisons therefore concern the stated sampling distributions rather than identical doses of past experience. First-task fallback and actual batch normalization require code confirmation.

5 | Experimental Setup

The source reports 200 runs: ten seeds 42-51, two layouts and ten conditions per layout, with 186 s runtime. NumPy [13], scikit-learn [11], SciPy [14] and Matplotlib [15] are named components. The original script and results file are needed to verify execution, controls, timings, and programmatic insertion of values.

Reported checks include first-task loss changing from 0.551 to 0.00016 and mean untrained accuracy 0.49 over non-joint runs. These are plausible sanity summaries, but pooled mean accuracy does not prove unbiased heads or absence of forward transfer on each task. Run-level loss and accuracy matrices are required to confirm the checks.

The source reports a balanced two-layout by ten-condition grid per seed, yielding 200 runs. Naive, ER/IR at four caps, and joint have ten replications per cell. Shared splits, initialization and snapshots support pairing among sequential strategies; original artifacts must confirm completeness, exclusions and exact controls.

Paired two-sided Wilcoxon signed-rank tests summarize differences over ten seeds; rank-biserial effects and percentile bootstrap intervals are reported. The source uses 2000 bootstrap draws with generator 1234. The twenty-six comparisons are exploratory and unadjusted in the original tables. A Holm sensitivity calculation on the displayed rounded p-values gives minimum adjusted $p = 0.052$ across the complete 26-test family, so none reaches 0.05. Exact nonzero counts, ties and rounding require the raw results before assigning exact adjusted probabilities.

Table 1. Architecture, training hyperparameters, and experimental design as reported in the source manuscript; execution requires the original code and run-level record.

Setting	Value
Dataset	Scikit-learn load_digits subset of UCI Optdigits: 1,797 samples, 64 features, 10 classes
Task partition	T = 5 two-class tasks; task i discriminates digits {2i, 2i+1}
Split	Fixed stratified 80/20 split per task, shared across strategies and regimes
Architecture	64-16-16 ReLU shared body + 5 task-specific sigmoid heads (task-incremental)
Loss	Binary cross-entropy on the active task head
Optimizer	Mini-batch SGD, fixed learning rate 0.3, no momentum/decay
Current batch size	32

Table 1. Continued.

Setting	Value
Replay batch size	32 (appended per step for ER/IR)
Steps per task	1,000 (identical for all strategies)
Buffer capacities	Nominal 25 / 50 / 100 / 200 per class; actual retained count min(capacity, available class training examples)
Similarity regimes	Natural (shared layout); permuted (per-task fixed feature permutation)
Seeds	42–51 (10 independent replications per condition)
Total runs	200
Runtime	186 s on a single workstation CPU

6 | Results

6.1 | Catastrophic Forgetting under Naive Sequential Training

Fig. 1 shows strong retention losses in naive sequential training. The reported natural-layout BWT is -0.170 and final accuracy 0.861, versus joint-reference 0.996; task 0 falls from 1.000 just-trained to 0.754 after the fifth task. The other earlier-task losses are 13.1, 18.8 and 11.7 percentage points. Above-diagonal values are near binary chance in aggregate, which does not exclude individual-task forward transfer. Permuted naive BWT is -0.135 with larger seed variation.

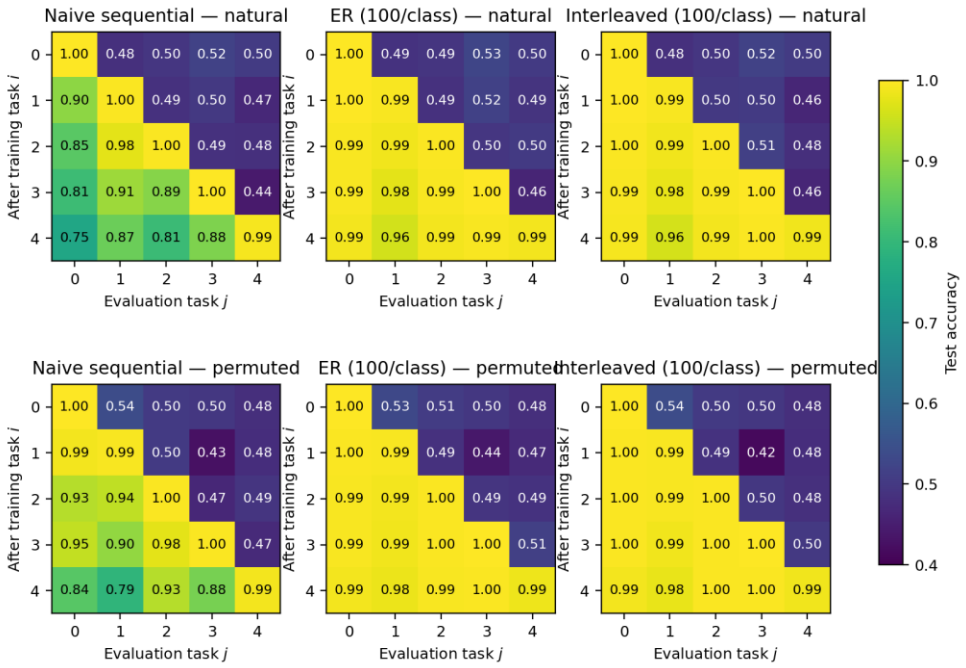


Fig. 1. Mean test-accuracy matrices A[i,j] over 10 seeds — accuracy on task j (columns) after completing task i (rows) — for naive sequential training, ER (100 samples per class), and IR (100 per class), in the natural (top row) and permuted (bottom row) regimes.

Lower-triangular degradation relative to the diagonal shows forgetting; above-diagonal entries sit at the binary chance level (*Fig. 1*).

6.2 | Effect of Replay Buffer Size

Natural-layout ER means improve with nominal capacity: BWT -0.035, -0.024, -0.015, -0.010 at 25, 50, 100, 200, and accuracy 0.969, 0.977, 0.984, 0.988. Capacity 100 stores about 1000 examples, roughly 70 percent of the full training stream, and closes approximately 91 percent of the naive-to-joint accuracy gap. Capacity 200 saturates the available class training data. Monotonicity is an aggregate ER pattern, not a universal law:

permuted IR BWT changes from -0.010 at 25 to -0.011 at 50 and its accuracy from 0.987 to 0.986. No formal paired monotone-trend test is provided.

Table 2. Reported final accuracy and BWT over ten seeds.

Regime	Strategy	Final Avg. Accuracy (Mean ± SD)	95% CI	BWT (mean ± SD)	95% CI
Natural	Naive (no memory)	0.861 ± 0.047	[0.833, 0.887]	-0.170 ± 0.060	[-0.206, -0.137]
Natural	ER 25	0.969 ± 0.017	[0.959, 0.978]	-0.035 ± 0.020	[-0.047, -0.024]
Natural	IR 25	0.971 ± 0.013	[0.963, 0.978]	-0.032 ± 0.016	[-0.041, -0.023]
Natural	ER 50	0.977 ± 0.010	[0.971, 0.982]	-0.024 ± 0.012	[-0.031, -0.017]
Natural	IR 50	0.978 ± 0.009	[0.973, 0.984]	-0.022 ± 0.011	[-0.029, -0.016]
Natural	ER 100	0.984 ± 0.005	[0.981, 0.987]	-0.015 ± 0.007	[-0.018, -0.011]
Natural	IR 100	0.986 ± 0.006	[0.982, 0.989]	-0.012 ± 0.006	[-0.016, -0.009]
Natural	ER 200	0.988 ± 0.006	[0.984, 0.992]	-0.010 ± 0.006	[-0.014, -0.006]
Natural	IR 200	0.989 ± 0.006	[0.986, 0.992]	-0.008 ± 0.007	[-0.012, -0.004]
Natural	Joint (future-data reference)	0.996 ± 0.004	[0.994, 0.998]	-0.000 ± 0.002	[-0.001, 0.001]
Permuted	Naive (no memory)	0.887 ± 0.085	[0.834, 0.937]	-0.135 ± 0.107	[-0.204, -0.073]
Permuted	ER 25	0.984 ± 0.009	[0.978, 0.989]	-0.014 ± 0.010	[-0.020, -0.008]
Permuted	IR 25	0.987 ± 0.008	[0.982, 0.991]	-0.010 ± 0.008	[-0.015, -0.006]
Permuted	ER 50	0.987 ± 0.009	[0.981, 0.991]	-0.010 ± 0.010	[-0.017, -0.005]
Permuted	IR 50	0.986 ± 0.008	[0.981, 0.990]	-0.011 ± 0.009	[-0.018, -0.007]
Permuted	ER 100	0.990 ± 0.004	[0.987, 0.992]	-0.006 ± 0.006	[-0.009, -0.002]
Permuted	IR 100	0.990 ± 0.004	[0.988, 0.993]	-0.006 ± 0.006	[-0.010, -0.002]
Permuted	ER 200	0.991 ± 0.003	[0.989, 0.992]	-0.005 ± 0.003	[-0.007, -0.003]
Permuted	IR 200	0.990 ± 0.003	[0.988, 0.992]	-0.006 ± 0.004	[-0.008, -0.003]
Permuted	Joint (future-data reference)	0.996 ± 0.002	[0.995, 0.997]	0.000 ± 0.001	[0.000, 0.001]

Note: Numbers 25, 50, 100, 200 denote nominal capacity per class, not necessarily retained sample counts; capacity 200 exceeds available training examples per class. Brackets are source percentile bootstrap intervals.

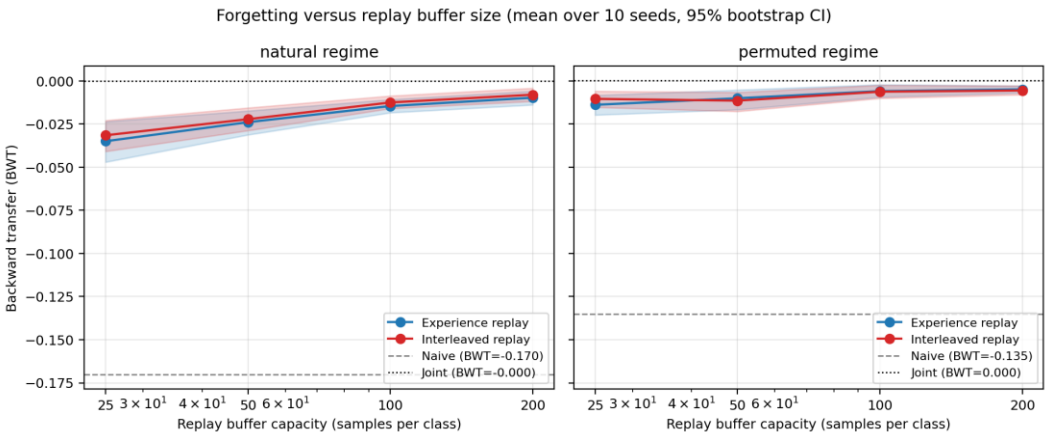


Fig. 2. BWT versus replay buffer capacity (mean over 10 seeds; shaded bands are 95% bootstrap confidence intervals) for ER and IR in the natural (left) and permuted (right) regimes.

Dashed and dotted horizontal lines mark the naive and joint-training BWT levels (*Fig. 2*).

6.3 | Learning Trajectories Across the Stream

Fig. 3 reports mean accuracy over encountered tasks. Naive means decline from 1.000 after the first task to 0.861 after the fifth, with intermediate 0.947, 0.939, 0.903. Replay curves remain near ceiling but do not all

rise: seen-task averages can decrease as retained tasks accumulate. These averages mix current acquisition and past retention; a separate plasticity endpoint is required to claim replay does not impede new-task learning.

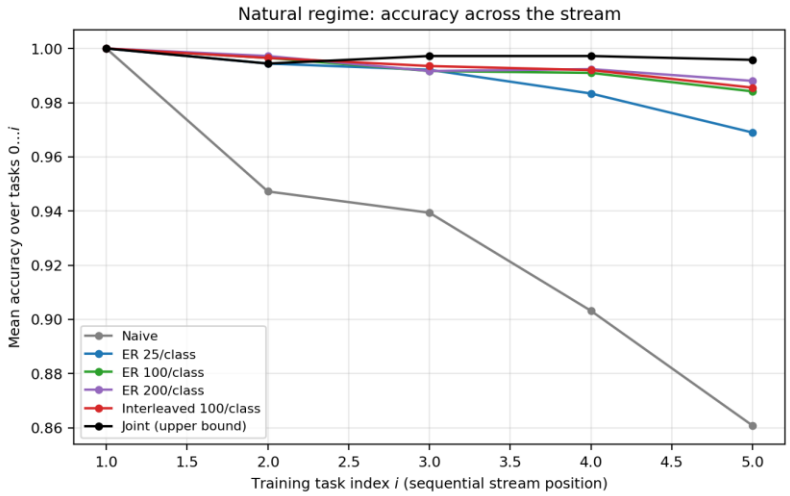


Fig. 3. Natural-regime learning curves: Mean accuracy over all tasks encountered so far, measured after completing each training task, for naive training, replay at three capacities, IR (100 per class), and joint training (mean over 10 seeds).

6.4 | Interleaved Versus Uniform Replay

Table 3 reports small ER-versus-IR differences. None of the eight BWT comparisons reaches an unadjusted 0.05 threshold; the smallest reported p is 0.058. One of sixteen interleaving comparisons, permuted accuracy at capacity 25, has unadjusted p = 0.036 but does not survive multiplicity control. These are failures to detect benefit under the reported protocol, not equivalence. Since ER includes current-task samples while IR excludes them, the contrast combines old-task dose with sampling allocation rather than isolating temporal scheduling.

Table 3. Paired Wilcoxon signed-rank tests, IR versus ER, over 10 seeds per comparison. W = test statistic; p = two-sided p-value; r = matched-pairs rank-biserial correlation (positive values favor IR).

Regime	Buffer/Class	Metric	W	p	Rank-Biserial r
Natural	25	BWT	11.0	0.326	+0.39
Natural	25	Accuracy	11.0	0.327	+0.39
Natural	50	BWT	10.5	0.293	+0.42
Natural	50	Accuracy	21.0	0.557	+0.24
Natural	100	BWT	4.0	0.168	+0.62
Natural	100	Accuracy	7.0	0.463	+0.33
Natural	200	BWT	3.0	0.112	+0.71
Natural	200	Accuracy	3.0	0.461	+0.40
Permuted	25	BWT	4.5	0.058	+0.75
Permuted	25	Accuracy	3.0	0.036	+0.83
Permuted	50	BWT	4.0	0.168	-0.62
Permuted	50	Accuracy	15.0	0.673	+0.17
Permuted	100	BWT	6.5	0.786	-0.13
Permuted	100	Accuracy	12.5	0.799	+0.11
Permuted	200	BWT	5.0	0.496	-0.33
Permuted	200	Accuracy	6.5	0.786	-0.13

6.5 | Effect of Task Similarity

Per-task permutation is associated with higher BWT in several unadjusted comparisons. Natural ER 25 has BWT -0.035 versus -0.014 under permutation (reported p = 0.002); naive and IR 200 do not reach unadjusted 0.05. Seven comparisons are below 0.05 before correction. Across the whole 26-test family none survives Holm on the rounded table values; treating the 10 layout tests as a separately prespecified family would retain only ER 25, IR 25 and ER 100. Such family choice is a sensitivity convention here, not retrospective preregistration. The observed effect is specific to coordinate layout and does not identify a general semantic-similarity or gradient mechanism.

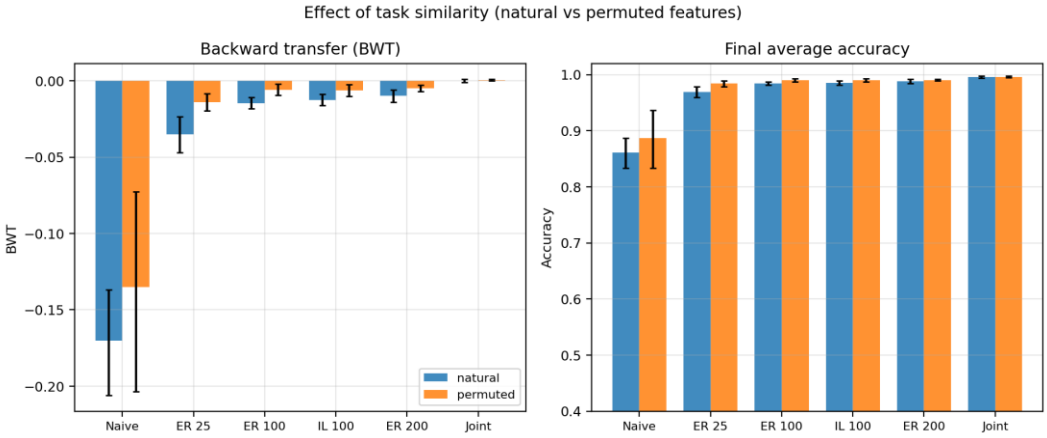


Fig. 4. Effect of task similarity on BWT (left) and final average accuracy (right) for selected conditions (mean over 10 seeds; error bars are 95% bootstrap confidence intervals).

Natural = shared input layout; permuted = per-task feature permutations (natural and permuted variants of each condition shown side by side) (*Fig. 4*).

Table 4. Task-similarity effect on BWT: Natural versus permuted regime (paired two-sided Wilcoxon signed-rank tests over 10 seeds).

Condition	BWT Natural	BWT Permuted	W	p	Rank-Biserial r
Naive	-0.170	-0.135	18.0	0.375	-0.35
ER 25	-0.035	-0.014	0.0	0.002	-1.00
IR 25	-0.032	-0.010	0.0	0.002	-1.00
ER 50	-0.024	-0.010	4.5	0.020	-0.84
IR 50	-0.022	-0.011	4.0	0.014	-0.85
ER 100	-0.015	-0.006	1.0	0.004	-0.96
IR 100	-0.012	-0.006	5.0	0.020	-0.82
ER 200	-0.010	-0.005	1.0	0.011	-0.96
IR 200	-0.008	-0.006	10.5	0.154	-0.53
Joint	-0.000	0.000	1.5	0.414	-0.50

Note: r = rank-biserial for natural-minus-permuted differences; negative values indicate less forgetting (higher BWT) in the permuted regime.

7 | Statistical Analysis

Tables 3 and *4* retain the twenty-six reported unadjusted paired tests: sixteen ER-IR comparisons and ten layout comparisons. One ER-IR accuracy contrast has p = 0.036, whereas all BWT contrasts have p ≥ 0.058. Holm correction on displayed values across 26 tests gives no adjusted p below 0.05 (minimum 0.052); separating the exploratory questions into 16- and 10-test families leaves no ER-IR result and only three layout contrasts below 0.05. Exact adjustment should use unrounded raw probabilities and a justified family definition.

Unadjusted layout p-values range from 0.002 to 0.020 for the seven favourable contrasts, with large reported rank-biserial magnitudes. Separate marginal bootstrap intervals are not a paired confidence interval for the

contrast, and nonoverlap cannot replace a multiplicity-aware paired test. Conversely, overlapping intervals do not prove equivalence or absence of a practically relevant difference.

Ten nonzero, untied pairs permit a minimum exact two-sided signed-rank probability 2/2^10=0.001953125. Zeros or ties alter the effective sample and implementation; the raw comparisons are needed to reconcile p-values and effect sizes. No formal equivalence margin, TOST, or preregistered testing family is supplied. Small observed mean differences do not establish a bound on the true paired effect.

8 | Discussion

Memory capacity is associated with most of the measured accuracy recovery in natural-layout ER, with diminishing increments at larger nominal caps. Capacity 100 retains roughly 70 percent of the training examples; capacity 200 retains the complete available class data. These are sizeable memory fractions on a small benchmark, so they should not be described as an arbitrary 200-sample reservoir comparison or a deployment provisioning rule.

The replay distributions differ in both current-task duplication and past-task dose. In a filled class-balanced buffer after i prior tasks, ER includes the current task with probability 1/(i+1); that fraction declines as tasks accumulate. IR samples only prior tasks and assigns them equal task probabilities. At early stages and while reservoirs fill, these differences can be larger. The absence of a detectable ER-IR benefit does not test pure temporal ordering at an identical past-task dose or prove a specific complementary-learning-systems mechanism. A matched past-dose and current-duplication experiment would isolate those questions.

Input permutations reduce task-average image correlation and are associated with improved retention in some exploratory conditions. Gradient conflict is a possible explanation, but no gradient dot products, hidden-state overlaps, or parameter-importance trajectories are measured. Invertible task encodings preserve information but can change optimization and representation geometry; both may contribute. The evidence therefore supports a benchmark-specific association with layout rather than a verified interference mechanism.

The layout contrast is not significant for naive training, where seed variation is largest. A formal interaction test is needed before concluding that layout effects require partial memory. Separate significances across capacities do not by themselves establish an interaction.

This task-incremental study compares exemplar replay distributions only. It does not benchmark against regularization methods [6], generative replay, larger networks or class-incremental inference. Shared reported controls improve comparison, but implementation verification and the replay-dose distinction remain necessary for strong internal-validity claims.

9 | Implications

The experiment separates nominal memory capacity, a combined replay-distribution treatment, and coordinate layout. It finds a marked capacity association and no multiplicity-supported ER-IR advantage. Joint future-data access has a different information regime, so its performance cannot identify biological interleaving as the causal mechanism. Layout is a useful controlled benchmark variable, while generalization and mechanistic attribution require new measurements.

On this small digit subset, a nominal cap 100 retains about 70 percent of the training stream and yields ER accuracy 0.984 versus joint-reference 0.996; retaining all class training examples at cap 200 yields 0.988. These fractions and task-identity assumptions limit direct use for edge-memory provisioning. Per-task remapping may affect optimization and interface requirements, so a deployment recommendation cannot follow from this single layout experiment.

10 | Limitations

The learner is small and uses task-specific heads with known task identity. The dataset is the 1797-example scikit-learn subset rather than the full UCI corpus. Coordinate permutations modify layout statistics without changing task labels, but do not remove every shared representation. Nominal cap 200 exceeds available class training examples. The ER-IR contrast changes past-task sample dose and current-task inclusion, and the first-task fallback needs confirmation.

The seed-level tests are exploratory, contain ties and zeros, and lack a preregistered family, paired-effect interval, power analysis or equivalence margin. Multiplicity sensitivity limits strong significance claims. The source reports 1000 updates per task, around 111 current-sample exposures in epoch units, with additional replay work; normalization and actual counts require code. Hyperparameters are shared, and no regularization or generative baselines are evaluated.

11 | Reproducibility and Data Availability

The source specifies code/experiment.py, results/results.json and code/build_docx.py as the generation workflow, with seeds 42-51 and bootstrap seed 1234. It reports Python 3.12.14, NumPy 2.1.3, SciPy 1.14.1, scikit-learn 1.5.2 and Matplotlib 3.9.2, and 186 s runtime. A stable repository or supplied code/result package is required to verify these commands, environments and generation claims; it is not identified here.

The public dataset is the bundled scikit-learn digit subset [10], [11]. Exact splits, permutations, reservoirs, fallback batches and the claimed 200 accuracy matrices require the seeded original script and machine-readable output. No new human or animal participants are described, but applicable dataset-use conditions and declarations should be confirmed by the authors.

12 | Conclusion

The reported task-incremental experiment compares exemplar replay on a small real digit subset. Natural-layout ER has improving mean retention with larger nominal capacity, from BWT -0.170 without replay to -0.010 when the available training examples are retained, while final accuracy rises from 0.861 to 0.988 against a joint future-data reference of 0.996. These means are descriptive of the stated protocol rather than a monotonicity theorem.

No reported BWT contrast detects an ER-IR advantage at unadjusted 0.05, and none survives whole-family multiplicity control. Because past-task dose and current-task inclusion differ, this is a comparison of sampling distributions rather than a pure interleaving test. Layout permutations have favourable exploratory contrasts, but no whole-family Holm significance or measured gradient mechanism is established. The results provide a bounded empirical baseline whose reproduction and broader relevance require code, raw matrices and larger, matched-treatment studies.

Data Availability

The study uses the 1797-example load_digits subset of UCI Optical Recognition of Handwritten Digits. Splits and permutations are experimental transformations; their exact reconstruction requires the original seed-controlled code.

The study describes no newly recruited human or animal participants. Dataset permissions and any applicable review requirements must be confirmed by the authors.

Funding

This work received no external funding.

Competing Interests

none declared.

Author Contributions

Actual contributions and author approval must be confirmed by the named researchers. A program or AI tool cannot establish the author roles or scientific accountability.

AI Disclosure

The source reports AI-assisted drafting. The authors must confirm the tools and tasks actually used, the origin of experiment code and data, and responsibility for independent scientific verification.

References

[1] McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation* (Vol. 24, pp. 109–165). Elsevier. https://doi.org/10.1016/S0079-7421(08)60536-8

[2] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. https://doi.org/10.1038/323533a0%0A%0A

[3] McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, *102*(3), 419. https://doi.org/10.1037/0033-295X.102.3.419

[4] Robins, A. (1995). Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection science*, *7*(2), 123–146. https://doi.org/10.1080/09540099550039318

[5] Sutton, R. S., Barto, A. G., & Barto, A. (1998). *Reinforcement learning: An introduction* (Vol. 1). MIT press, Cambridge. https://doi.org/10.1017/S0263574799271172

[6] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., & Rusu, A. A. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521–3526. https://doi.org/10.1073/pnas.1611835114

[7] De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., … & Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, *44*(7), 3366–3385. https://doi.org/10.1109/TPAMI.2021.3057446

[8] Wang, L., Zhang, X., Su, H., & Zhu, J. (2024). A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence*, *46*(8), 5362–5383. https://doi.org/10.1109/TPAMI.2024.3367329

[9] de Ven, G. M., Soures, N., & Kudithipudi, D. (2024). *Continual learning and catastrophic forgetting*. https://doi.org/10.1016/B978-0-443-15754-7.00073-0

[10] Alpaydin, E., & Kaynak, C. (1998). *Optical recognition of handwritten digits*. UCI Machine Learning Repository. https://doi.org/10.24432/C50P49

[11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., & Grisel, O. (2011). Scikit-learn: Machine learning in Python. *The journal of machine learning research*, *12*, 2825–2830. https://jmlr.org/papers/v12/pedregosa11a.html

[12] Settles, B. (2009). *Active learning literature survey*. https://burrsettles.com/pub/settles.activelearning.pdf

[13] Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., & Cournapeau, D. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

[14] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., & Cournapeau, D. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[15] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in science & engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55