CARES: COVERAGE-AWARE REPLAY EXEMPLAR SELECTION FOR CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Replay-based Continual Learning mitigates catastrophic forgetting by keeping a buffer of past samples, making buffer composition critical under a fixed memory budget. Recent coreset-based replay methods improve buffer quality using model-derived gradients, losses, or auxiliary optimization, but incur substantial computational overhead as memory is repeatedly updated. We propose CARES, a continual model-optimization-free approach for buffer construction that instead exploits the geometry of a frozen pretrained feature space using class-aware, density-weighted coverage. We evaluate CARES across balanced, imbalanced, and label-noisy streams under online class- and task-incremental settings. On Tiny-ImageNet, CARES is 2.3–8.8$\times$ faster than existing coreset-based selectors. Although CARES achieves this speedup using no training signal from the continual learner, it ranks among the state of the art across balanced, imbalanced, and label-noisy streams under online class- and task-incremental settings, outperforming the strongest baseline by up to 7.58 percentage points under class imbalance while also attaining higher average accuracy on Tiny-ImageNet. Code will be released upon publication.

1 INTRODUCTION

Continual learning (CL) studies how a model can acquire knowledge while being trained on a sequence of tasks without losing performance on previously learned tasks. This setting reflects realistic learning scenarios, where data does not arrive all at once but instead appears as a stream over time (Wang et al., 2024). However, deep neural networks trained under this constraint suffer from *catastrophic forgetting* (McCloskey & Cohen, 1989; Robins, 1995), in which updates on newer tasks overwrite representations useful for older ones. This problem is particularly challenging in class-incremental learning, where the model must distinguish among all classes observed so far without receiving the task identity at inference time.

Methods for mitigating catastrophic forgetting are commonly grouped into regularization-, architecture-, and rehearsal-based approaches. *Regularization* methods constrain model updates to preserve previously acquired knowledge (Kirkpatrick et al., 2017; Zhang et al., 2019; Aljundi et al., 2018; Chaudhry et al., 2018; Ritter et al., 2018; Serra et al., 2018). *Architecture-based* approaches modify the network architecture through expansion or parameter isolation (Rusu et al., 2022; Schwarz et al., 2018; Mallya & Lazebnik, 2018; Xu & Zhu, 2018; Yoon et al., 2020; Hung et al., 2019; Sokar et al., 2021). Finally, *replay-based* methods maintain a buffer of samples from past tasks, which are replayed during training to address forgetting (Chaudhry et al., 2019a;b; Buzzega et al., 2020; Aljundi et al., 2019b;a; Lopez-Paz & Ranzato, 2022).

Under a fixed memory budget, the effectiveness of rehearsal depends not only on the number of samples stored but also on which samples are retained in the buffer (Rebuffi et al., 2017; Yoon et al., 2022; Aljundi et al., 2019b). Methods such as experience replay (Chaudhry et al., 2019b) provide inexpensive buffer updates but do not select samples that accurately represent the previously seen data. Other works employ distillation losses or constrain updates in addition to replaying past samples (Buzzega et al., 2020; Chaudhry et al., 2018; 2019a). Consequently, recent work has formulated buffer construction in rehearsal as a coreset selection problem (Yoon et al., 2022; Tiwari et al., 2022; Jie Hao, 2023).

In a continual learning stream, buffer updates happen repeatedly and must keep pace with incoming data; selection is not a one-time cost, but one incurred at every task boundary or even every incoming

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

batch (Wang et al., 2024). Gradient- and loss-based criteria (Aljundi et al., 2019b;a; Tiwari et al., 2022; Yoon et al., 2022; Jie Hao, 2023) produce effective buffers, particularly under extreme memory limits, but they rely on additional computationally heavy optimization steps. Because this cost recurs at every update rather than once, it accumulates over the lifetime of the stream and can become a substantial fraction of total training time. This computational burden creates a practical tension: the very methods designed to make rehearsal more sample-efficient often introduce overhead that undermines the efficiency gains they seek to provide, particularly when buffer updates must keep pace with a continuous data stream (Tiwari et al., 2022; Aljundi et al., 2019a).

These limitations raise a more fundamental question: *must a continually updated model also provide the signal that determines what it should remember?* We investigate a complementary hypothesis: a replay buffer can instead be constructed by preserving the representative structure of the observed data in a stable feature space. Features from a frozen encoder provide a continual model-independent representation space in which examples can be compared throughout the stream. Within this space, geometric coverage encourages the buffer to represent distinct regions of the data distribution. However, pure coverage may overemphasize isolated candidates, whereas prioritizing only dense regions risks selecting redundant examples from dominant modes. Density-aware coverage provides a natural compromise between these two objectives.

SubZeroCore (Moser et al., 2025) addresses this coverage-density tension in static coreset selection through a density-aware coverage objective. We build on this rather than classical geometric selectors such as herding (Welling, 2009; Rebuffi et al., 2017), or k-Center Greedy (Sener & Savarese, 2018), which primarily emphasize representativeness or coverage (Moser et al., 2026), because SubZeroCore incorporates density without requiring model-derived training signals. This makes it well suited to the repeated memory updates of online continual learning. However, it assumes simultaneous access to a fixed candidate pool, whereas continual learning requires a bounded buffer that must be updated as new task data arrives and the distribution of available candidates may become strongly imbalanced across classes. Directly applying a global geometric selector can therefore allocate disproportionate memory to classes with more candidates.

To address these challenges, we propose **CARES**, a continual model-optimization-free approach to replay-buffer construction for online continual learning. At each memory-update step, CARES combines the existing replay memory with a staging pool of newly observed examples and embeds the resulting candidate set using a frozen pretrained encoder. It then allocates the fixed replay budget across observed classes and applies density-aware coverage independently within each class before reconstructing the memory. In this way, CARES adapts training-free geometric coreset selection to a continually evolving replay buffer of fixed capacity.

Crucially, memory construction in CARES is decoupled from optimization of the continual learner: selection requires no model-derived losses, gradients, or auxiliary model training. On Split-Tiny-ImageNet, this yields memory updates that are 2.3–8.8$\times$ faster than existing coreset-based replay methods, while CARES attains higher final accuracy. This favorable accuracy–efficiency trade-off holds across balanced, imbalanced, and label-noisy online continual-learning settings, where CARES remains competitive with considerably more expensive selectors and is particularly effective under class imbalance.

2 Related Work

Replay-based CL: Experience Replay (ER) (Chaudhry et al., 2019b) trains jointly on current and stored examples. Beyond random storage, replay methods select or retrieve examples using interference, uncertainty, gradients, or diversity. Dark Experience Replay (DER) and DER++ (Buzzega et al., 2020) apply distillation-style objectives, whereas others (Lopez-Paz & Ranzato, 2022; Chaudhry et al., 2019a) constrain gradient updates to preserve previous model behavior. GSS (Aljundi et al., 2019b), promotes gradient diversity among buffer samples, while MIR (Aljundi et al., 2019a) retrieves examples predicted to suffer the largest loss increase after an update. Collectively, these methods underscore that buffer composition, not just buffer size, drives rehearsal effectiveness.

Feature-Space Buffer Construction: A separate line of work sidesteps learner-derived signals entirely, relying instead on a fixed feature extractor. iCaRL (Rebuffi et al., 2017) uses the feature extractor of the continually trained network to select exemplars by herding. REMIND (Hayes et al.,

Algorithm 1 CARES Replay Memory Update

Require: Task stream $\mathcal{D}_{1:T}$, replay capacity K, staging capacity P, target coverage γ, frozen encoder g_ϕ

$\mathcal{M}_0 \leftarrow \emptyset$
for $t = 1, \ldots, T$ **do**
  $\mathcal{P}_t \leftarrow \emptyset$
  for each minibatch $\mathcal{B}_t \subset \mathcal{D}_t$ **do**
    Sample a replay minibatch $\mathcal{B}_t^{\mathcal{M}}$ uniformly from $\mathcal{M}_{t-1}$, if non-empty
    Update θ using cross-entropy on $\mathcal{B}_t \uplus \mathcal{B}_t^{\mathcal{M}}$
    Update $\mathcal{P}_t$ with examples in $\mathcal{B}_t$ using reservoir sampling
  end for
  $\mathcal{A}_t \leftarrow \mathcal{M}_{t-1} \uplus \mathcal{P}_t$
  if $|\mathcal{A}_t| \leq K$ **then**
    $\mathcal{M}_t \leftarrow \mathcal{A}_t$
  else
    $\mathcal{Z}_t \leftarrow \{(z_i, y_i) : z_i = g_\phi(x_i)/\|g_\phi(x_i)\|_2, (x_i, y_i) \in \mathcal{A}_t\}$
    $\mathcal{I}_t \leftarrow \text{CARES-SELECT}(\mathcal{Z}_t, K, \gamma)$
    $\mathcal{M}_t \leftarrow \mathcal{A}_t[\mathcal{I}_t]$
  end if
end for

2020), instead, stores compressed hidden representations from frozen early network layers for later replay, while Rainbow Memory (Bang et al., 2021) selects diverse exemplars based on classification uncertainty computed directly in feature space. Unlike these methods, CARES formulates buffer construction as an explicit coreset-selection objective over a frozen embedding space, combining class-aware allocation with density-aware coverage rather than uncertainty- or compression-driven heuristics.

Coreset-Selection for Replay: A coreset is a small, representative subset of a larger dataset that approximates the full-data objective for a given learning task. Classical geometric constructions include clustering (Gonzalez, 1985), farthest-first traversal (Har-Peled & Mazumdar, 2004), and facility location (Cornuéjols et al., 1977). Depending on the task, coresets may be selected to preserve geometric coverage, class balance, diversity, uncertainty, gradient information, or loss characteristics (Moser et al., 2026). In memory-constrained settings, coreset selection provides a principled way to decide which samples should be retained when only a limited number of examples can be stored.

Applying coreset selection to buffer construction has gained traction in recent years. Recent CL methods optimize model-dependent proxies for buffer construction. GCR (Tiwari et al., 2022) selects examples whose gradients approximate those of all data seen so far. OCS (Yoon et al., 2022) applies representativeness, diversity, and affinity criteria during online training. On the other hand, BCSR (Jie Hao, 2023) formulates selection as a bilevel problem over a learned sample distribution, while CSReL (Tong et al., 2025) prioritizes examples through a reducible-loss signal obtained with a holdout model. Across these works, a common thread emerges: effective rehearsal depends on formalizing which samples to keep as an explicit optimization problem rather than a heuristic rule, a principle that motivates the coreset-based approach taken in this work but, as we discuss next, this formalization typically comes at the cost of repeated continual model-dependent computations.

3 CARES

3.1 Setting and Problem Definition

We consider continual learning over a sequence of tasks with labeled data $\mathcal{D}_1, \ldots, \mathcal{D}_T$. In task-incremental learning (TIL), the task identity is available at inference and predictions are restricted to the corresponding class set. In class-incremental learning (CIL), task identity is unavailable and the model must predict among all classes observed so far.

Replay-buffer construction maintains a replay memory $\mathcal{M}_t$ of capacity K after every task t, drawn from the data observed so far:

$$\mathcal{M}_0 = \emptyset, \qquad \mathcal{M}_t \subseteq \mathcal{D}_{1:t}, \qquad |\mathcal{M}_t| \leq K, \qquad \mathcal{D}_{1:t} = \bigcup_{s=1}^{t} \mathcal{D}_s. \tag{1}$$

During task t, an incoming minibatch $\mathcal{B}_t$ is combined with a replay minibatch $\mathcal{B}_t^{\mathcal{M}}$ sampled from $\mathcal{M}_{t-1}$, the memory built after the previous task. CARES changes only how $\mathcal{M}_t$ is constructed; it does not modify the continual model's training objective or inference rule.

Ideally, after observing task $t < T$, replay-memory construction would select the subset of all data seen so far whose replay best preserves performance on those tasks once training continues:

$$\mathcal{M}_t^* \in \underset{\mathcal{M} \subseteq \mathcal{D}_{1:t},\ |\mathcal{M}| \leq K}{\arg\min} \mathcal{L}_{1:t}(\theta_{t+1}(\mathcal{M})), \tag{2}$$

where $\theta_{t+1}(\mathcal{M})$ denotes the continual model obtained by training on $\mathcal{D}_{t+1}$ with $\mathcal{M}$ as replay memory, and $\mathcal{L}_{1:t}$ measures performance over all tasks observed so far. The memory built after task t can only influence training from task $t+1$ onward, so its value is realized in the future rather than in the model θ_t that has already been trained.

This ideal objective is not directly available in online continual learning. The next task $\mathcal{D}_{t+1}$ has not been observed when $\mathcal{M}_t$ must be fixed, the complete history $\mathcal{D}_{1:t}$ cannot be retained under the memory constraint, and evaluating candidate memories through $\mathcal{L}_{1:t}$ would require repeatedly optimizing the continual model. Practical methods therefore rely on tractable selection proxies computed from the data available at each memory-update point.

3.2 Continual Replay-Memory Construction

CARES adapts the density-aware geometric criterion of SubZeroCore (Moser et al., 2025) to online continual replay under a fixed budget. SubZeroCore assumes simultaneous access to a fixed candidate set. Continual replay instead requires newly observed examples to be collected without storing the complete stream, compared with previously retained examples, and selected under one persistent memory capacity. CARES addresses these requirements through candidate staging, repeated memory reconstruction, and class-aware budget allocation.

For notational simplicity, t indexes a protocol-defined memory-update stage. During stage t, the replay memory $\mathcal{M}_{t-1}$ remains available for training, while newly observed examples are accumulated in a temporary staging pool $\mathcal{P}_t$ of capacity P. Once more than P examples have been observed, the staging pool is maintained using reservoir sampling. This operation only bounds the current-stage candidate collection; it does not determine the final replay memory.

At the memory-update point, CARES combines the previous replay memory and the staging pool:

$$\mathcal{A}_t = \mathcal{M}_{t-1} \uplus \mathcal{P}_t, \tag{3}$$

where $\uplus$ denotes the disjoint union of the two example collections. They share no examples because $\mathcal{P}_t \subseteq \mathcal{D}_t$ and $\mathcal{M}_{t-1} \subseteq \mathcal{D}_{1:t-1}$. Previously retained examples therefore compete directly with newly observed candidates.

Each candidate is represented using a frozen pretrained encoder g_ϕ:

$$z_i = \frac{g_\phi(x_i)}{\|g_\phi(x_i)\|_2}, \qquad \mathcal{Z}_t = \{(z_i, y_i) : (x_i, y_i) \in \mathcal{A}_t\}. \tag{4}$$

Because g_ϕ remains fixed throughout the stream, selection is independent of changes in the continually trained model f_θ. In particular, CARES uses no predictions, losses, gradients, or parameters from f_θ, and trains no auxiliary model for buffer construction.

If $|\mathcal{A}_t| \leq K$, CARES retains all candidates. Otherwise, it divides the replay capacity approximately equally across the classes represented in $\mathcal{A}_t$, applies density-aware coverage selection independently within each class, and replaces the previous memory with the candidates at the selected indices:

$$\mathcal{I}_t = \text{CARES-Select}(\mathcal{Z}_t, K, \gamma), \qquad \mathcal{M}_t = \mathcal{A}_t[\mathcal{I}_t]. \tag{5}$$

In both cases $\mathcal{M}_t \subseteq \mathcal{A}_t \subseteq \mathcal{D}_{1:t}$ and $|\mathcal{M}_t| \leq K$, so the update preserves Equation 1 by induction over t. The staging pool is then cleared before the next update stage. Algorithm 1 summarizes the complete procedure.

Table 1: Experimental results on Split-Tiny-ImageNet. CARES achieves the highest average accuracy (A_T) across all three stream conditions, with particularly large gains under label noise.

Method	Balanced		Imbalanced		Label Noise	
	$A_T \uparrow$	$FGT_T \downarrow$	$A_T \uparrow$	$FGT_T \downarrow$	$A_T \uparrow$	$FGT_T \downarrow$
K-means Embedding	41.48±1.21	0.129±0.007	36.29±0.23	0.085±0.003	36.01±1.51	0.083±0.005
Uniform	42.11±0.52	0.129±0.002	37.07±0.53	0.083±0.009	37.14±1.05	0.099±0.003
iCaRL	43.84±0.09	0.114±0.004	37.65±0.84	0.058±0.003	38.52±0.25	0.063±0.006
Greedy Coreset	41.02±0.33	0.119±0.017	33.43±0.86	0.103±0.002	36.37±0.16	0.079±0.006
GCR	41.45±0.35	0.125±0.008	36.08±0.62	0.072±0.017	37.46±0.40	0.115±0.011
OCS	41.29±0.09	0.112±0.001	35.09±0.60	**0.036±0.011**	35.36±0.94	0.061±0.005
BCSR	44.13±0.33	0.106±0.001	38.59±0.11	0.070±0.004	40.72±0.56	**0.055±0.006**
CARES (Ours)	**49.49±0.41**	**0.099±0.015**	**42.94±0.40**	0.104±0.011	**49.37±0.95**	0.078±0.021

3.3 Density-Aware Coverage Selection

The density score and facility-location objective described below are adopted from SubZero-Core (Moser et al., 2025). CARES contributes their continual application through bounded staging, repeated competition between old and new candidates, and class-aware allocation of the persistent replay capacity.

Consider a candidate collection with normalized embeddings $\{z_i\}_{i=1}^N$, from which $m < N$ examples must be selected. For a coverage target $\gamma \in (0, 1]$, the neighborhood order is defined as

$$\kappa = \min \left\{ k \in \{1, \dots, N-m\} : 1 - \prod_{q=1}^{k} \frac{N-m-q}{N-q} \geq \gamma \right\}. \tag{6}$$

The product equals $\binom{N-1-k}{m} / \binom{N-1}{m}$, the probability that a uniformly random selection of m of the other $N-1$ candidates contains none of the k nearest neighbors of a given candidate. Equation 6 therefore chooses the smallest neighborhood through which a random selection of size m covers a non-selected candidate with probability at least γ. The set is non-empty for every $\gamma \leq 1$ because the product vanishes at $k = N - m$. Let r_i be the Euclidean distance from z_i to its κ-th nearest neighbor among the other candidates in the normalized embedding space. Given the empirical mean μ and standard deviation σ of these radii, each candidate receives the density score

$$s_i = \exp\left(-\frac{(r_i - \mu)^2}{2\bar{\sigma}^2} \right), \qquad \bar{\sigma} = \max(\sigma, \epsilon), \tag{7}$$

where $\epsilon > 0$ prevents numerical instability. This score assigns lower weights to candidates with unusually large radii, which are relatively isolated, and to candidates with unusually small radii, which lie in highly redundant regions.

Because the embeddings are normalized, the non-negative cosine similarity used by CARES is

$$\mathrm{sim}(z_i, z_j) = \max\left(0, z_i^\top z_j\right). \tag{8}$$

For a selected index set $\mathcal{S} \subseteq \{1, \dots, N\}$, the density-weighted facility-location objective is

$$F(\mathcal{S}) = \sum_{i=1}^{N} \max_{j \in \mathcal{S}} \left[s_j \ \mathrm{sim}(z_i, z_j) \right]. \tag{9}$$

CARES approximately maximizes Equation 9 under the cardinality constraint $|\mathcal{S}| \leq m$ using greedy marginal-gain selection.

Class-aware allocation. Directly optimizing Equation 9 over all candidates can allow classes with more available examples to occupy a disproportionate share of the replay memory. CARES therefore allocates the memory budget across classes before applying geometric selection.

Let $\mathcal{Y}_t$ be the labels represented in $\mathcal{A}_t$, let $J_t = |\mathcal{Y}_t|$, and let N_c denote the number of candidates belonging to class c. Selection is only invoked when $|\mathcal{A}_t| > K$, so the full capacity K is distributed. Defining

$$b = \left\lfloor \frac{K}{J_t} \right\rfloor, \qquad q = K \bmod J_t, \tag{10}$$

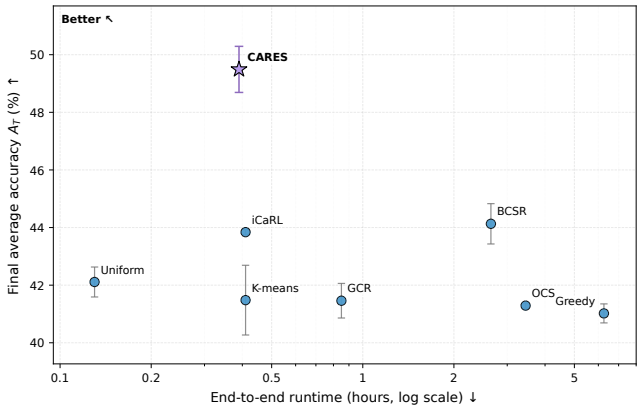


Figure 1: Accuracy–runtime trade-off on Split-Tiny-ImageNet under the balanced setting in Protocol 1 at $K = 200$. CARES achieves substantially higher final accuracy at a runtime comparable to lightweight selection methods.

each class receives an initial allocation $b_c \in \{b, b+1\}$, where the q classes with the largest N_c receive the additional slot, so that $\sum_c b_c = K$. The effective target for class c is

$$m_c = \min(b_c, N_c). \tag{11}$$

Classes with $m_c = N_c$ are retained in full. For every other class, CARES evaluates Equation 6–Equation 9 on that class's candidates with $N = N_c$ and $m = m_c$, and takes the union of the resulting class-wise subsets. The resulting replay memory contains $\sum_c m_c \leq K$ examples, with equality unless some class has fewer candidates than its allocation, and prevents classes with larger candidate pools from automatically dominating the global geometric objective.

4 Experiments

We evaluate CARES under two complementary online continual-learning protocols, measuring robustness to non-ideal streams, performance across memory capacities, and computational efficiency. Let $a_{l,j}$ denote the test accuracy on task j after training on task l. We report final average accuracy and final average forgetting (Chaudhry et al., 2018),

$$\begin{aligned} A_T &= \frac{1}{T}\sum_{j=1}^{T} a_{T,j}, \\ \mathrm{FGT}_T &= \frac{1}{T-1}\sum_{j=1}^{T-1} \max_{l\in\{j,\ldots,T-1\}} \left(a_{l,j} - a_{T,j}\right). \end{aligned} \tag{12}$$

Throughout, A_T is reported in percent and FGT_T as a fraction in $[0, 1]$.

4.1 Experimental Setup

Protocol 1: Following BCSR (Jie Hao, 2023), we evaluate online TIL under balanced, long-tailed imbalanced, and noisy-label streams. Split CIFAR-100 (Krizhevsky, 2009a) and Split Tiny-ImageNet (Le & Yang, 2015) divide the CIFAR-100 (Krizhevsky, 2009b) and TinyImageNet (Pouransari & Ghili, 2014) datasets into 20 tasks of 5 and 10 disjoint classes, respectively, with replay capacities $K = 100$ and $K = 200$. CARES replaces only the replay-memory selection procedure while retaining the reference training and evaluation protocol.

Protocol 2: Following GCR (Tiwari et al., 2022), we evaluate online CIL on Split-CIFAR-10, Split-CIFAR-100, and Split-TinyImageNet. Split-CIFAR-10 and Split-CIFAR-100 are split into 5 tasks of 2 and 20 non-overlapping classes respectively whereas Split-TinyImageNet is split into 10 tasks of 20 classes. The model receives a single pass over streams with blurred task boundaries and is evaluated at $K \in \{200, 500, 2000\}$, without task identity at inference.

Table 2: Results on Split-CIFAR-100. CARES matches the strongest baseline on the balanced stream and achieves the highest final average accuracy under imbalance and label noise settings.

Method	Balanced		Imbalanced		Label Noise	
	$A_T \uparrow$	$FGT_T \downarrow$	$A_T \uparrow$	$FGT_T \downarrow$	$A_T \uparrow$	$FGT_T \downarrow$
K-means Embedding	59.77±0.24	0.061±0.001	43.91±0.15	0.044±0.001	57.92±1.25	0.091±0.016
Uniform	58.99±0.54	0.074±0.004	44.73±0.11	0.033±0.007	58.76±1.07	0.087±0.006
iCaRL	60.74±0.09	**0.044±0.026**	44.25±2.04	0.042±0.019	59.70±0.70	0.071±0.010
Greedy Coreset	59.39±0.16	0.066±0.017	43.80±0.01	0.039±0.007	58.22±0.16	0.066±0.001
GCR	58.73±0.43	0.073±0.013	44.48±0.05	0.035±0.005	58.72±0.63	0.081±0.005
OCS	52.57±0.37	0.088±0.001	46.54±0.34	**0.022±0.003**	51.77±0.81	0.103±0.007
BCSR	**61.60±0.14**	0.051±0.015	47.30±0.57	**0.022±0.005**	60.70±0.08	**0.059±0.013**
CARES (Ours)	61.52±1.48	0.111±0.074	**54.88±0.33**	0.082±0.083	**62.12±1.14**	0.110±0.013

Baselines. The comparison set is protocol-specific. For each protocol, we retain the baselines, data splits, learner architecture, replay objective, and training configuration of the corresponding reference setting. We therefore interpret differences only within a shared protocol and do not compare absolute accuracies across tables.

Implementation and reproducibility details. Unless otherwise stated, the continual model uses a ResNet-18 backbone, while CARES constructs the replay memory using ℓ_2-normalized features from a frozen ImageNet-pretrained Inception-V3 encoder. Replay minibatches match the current-training minibatch size (10/20 for Split CIFAR-100/Tiny-ImageNet in Protocol 1 and 32 in Protocol 2). Runtime measurements are obtained on a single NVIDIA RTX A6000 GPU with four CPU cores. Full dataset-specific hyperparameters are listed in Appendix D.

4.2 Robustness to Imbalance and Label Noise

One of the main advantages of CARES is particularly pronounced under non-ideal streams, as demonstrated in Tables 1 and 2. On Split CIFAR-100, CARES reaches 54.88% under class imbalance, outperforming the strongest baseline, BCSR, by 7.58 percentage points, while remaining competitive on the balanced stream. On Tiny-ImageNet, CARES achieves the highest accuracy under all three conditions, improving over BCSR by 5.36, 4.35, and 8.65 points on balanced, imbalanced, and label-noisy streams, respectively. These results indicate that learner-independent, class-aware geometric selection remains effective when the stream distribution is skewed, or labels are corrupted.

CARES does not uniformly minimize FGT_T, but this does not contradict its stronger end-of-stream performance. While A_T measures the absolute accuracy retained after the full stream, FGT_T measures the decline of each task relative to its best historical accuracy. Consequently, a method can exhibit a larger peak-to-final drop while still retaining higher absolute accuracy at the end of training. We therefore view CARES's primary advantage in Protocol 1 as improved final retained performance, with A_T and FGT_T capturing complementary aspects of continual-learning behavior.

4.3 Performance Across Memory Capacities

Table 3 evaluates CARES under online CIL at different replay capacities. At $K = 200$, CARES is competitive on Split-CIFAR-10 and Split-CIFAR-100 and achieves the highest accuracy on Split-TinyImageNet, reaching 6.60% compared with 6.19% for GCR. At $K = 2000$, CARES achieves the highest mean accuracy on Split-CIFAR-10 and Split-CIFAR-100, reaching 54.88% and 25.17%, respectively. These results indicate that CARES benefits strongly from additional replay capacity while remaining competitive under tighter memory constraints. Results for the intermediate $K = 500$ setting are provided in Appendix C (Table 9).

4.4 Runtime Efficiency

Figure 1 compare accuracy and end-to-end runtime on Split-Tiny-ImageNet under Protocol 1. CARES completes the experiment in 0.39 hours, comparable to K-means Embedding and iCaRL (0.41 hours), while being $2.3\times$, $6.8\times$, and $8.8\times$ faster than GCR, BCSR, and OCS, respectively. Importantly,

Table 3: Online CIL average accuracy (15 seeds) under small ($K = 200$) and large ($K = 2000$) replay budgets. CARES is competitive under tight memory and becomes increasingly effective as capacity grows, achieving the highest mean accuracy on Split-CIFAR-10 and Split-CIFAR-100 at $K = 2000$.

Method	Split-CIFAR-10		Split-CIFAR-100		Split-TinyImageNet	
	$K = 200$	$K = 2000$	$K = 200$	$K = 2000$	$K = 200$	$K = 2000$
ER	32.63±5.11	49.31±7.45	11.72±1.25	20.78±1.48	4.36±1.02	11.30±1.45
GEM	16.81±1.17	20.35±2.44	10.42±1.66	12.38±3.81	5.15±0.71	9.56±2.63
GSS	33.83±3.95	41.70±4.56	12.31±0.57	16.24±0.80	5.82±0.38	8.44±0.48
iCaRL	38.66±2.03	39.56±1.53	11.23±0.31	12.28±0.39	4.39±0.19	6.66±0.30
MIR	21.50±0.70	46.40±1.81	10.30±0.30	16.02±0.50	4.90±0.17	7.05±0.30
DER	38.26±5.08	49.90±7.55	10.68±2.03	24.83±1.58	4.81±0.57	**15.97±0.95**
GCR	**42.26±7.60**	52.30±7.01	**14.88±0.94**	22.47±2.01	6.19±0.36	11.83±1.26
CARES (Ours)	40.26±4.67	**54.88±6.34**	14.28±0.84	**25.17±1.39**	**6.60±0.59**	15.67±0.85

CARES also achieves higher final accuracy than these methods, yielding a favorable accuracy–efficiency trade-off without learner-derived selection signals.

4.5 Sensitivity Analysis

Table 4 studies sensitivity to the coverage parameter γ and staging-pool capacity P on balanced Split-CIFAR-100. Performance remains relatively stable across a broad range of coverage values, while increasing the staging pool beyond a moderate multiple of the replay capacity provides no consistent gain. This indicates that CARES does not require precise tuning or an excessively large temporary candidate pool.

Table 4: Hyperparameter sensitivity of CARES on balanced Split CIFAR-100. Final task-incremental accuracy (%) when varying the coverage parameter γ and the staging-pool size P, expressed as a multiple of the replay capacity K. All other hyperparameters are held fixed.

γ	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	
A_T	59.8	59.0	61.3	59.5	60.0	58.7	60.6	59.8	60.8	60.1	
P/K	1	2	3	4	5	6	7	8	9	10	20
A_T	59.8	60.2	59.4	60.7	58.7	59.7	57.3	58.5	58.3	58.8	60.7

Table 5 further evaluates the frozen representation used for selection. CARES exhibits similar performance across different pretrained encoders, indicating that its effectiveness is not tied to a particular feature extractor.

Table 5: Sensitivity to the frozen encoder. Final class-incremental accuracy (%) of CARES on Split-CIFAR-100 under Protocol 2 with different pretrained encoders used for selection. CARES remains consistent across encoders, indicating limited sensitivity to the choice of representation used for coreset selection.

Selection encoder	$K = 200$	$K = 500$	$K = 2000$
InceptionV3	15.0	19.1	23.5
ResNet-50	14.4	19.6	23.7
CLIP ViT-B/32	14.3	19.1	23.3

4.6 Ablation Studies

Table 6 isolates the contributions of class-aware allocation, geometric coverage, and density weighting. CB-Random preserves class-balanced allocation but selects samples randomly, Global Facility applies coverage globally, Global SubZeroCore adds density weighting to global coverage, and Class-wise Facility performs geometric selection independently within each class.

Table 6: Component ablation of CARES on Split-CIFAR-100. Class-aware allocation provides the dominant gain under class imbalance, substantially outperforming global geometric selection, while density weighting provides additional gains on balanced and label-noisy streams.

Variant	Class-aware	Coverage	Density	Balanced	Imbalanced	Label Noise
Uniform	×	×	×	58.99±0.54	44.73±0.11	58.76±1.07
CB-Random	✓	×	×	59.50±1.63	52.89±0.41	59.37±0.72
Global SubZeroCore	×	✓	✓	59.77±1.94	46.72±1.24	60.71±1.50
Global Facility	×	✓	×	59.83±1.50	46.29±2.07	61.36±1.09
Class-wise Facility	✓	✓	×	60.04±1.61	**55.12±0.37**	61.37±1.41
CARES	✓	✓	✓	**61.52±1.48**	54.88±0.33	**62.12±1.14**

The largest effect appears under class imbalance: class-wise selection substantially outperforms both global variants, showing that explicit allocation of replay capacity across classes is critical when candidate frequencies are skewed. Density weighting provides complementary gains on balanced and label-noisy streams, although its benefit is not uniform across all conditions. Overall, the ablation shows that class-aware allocation is the key continual-learning adaptation, with density-aware coverage providing an additional refinement.

5 Limitations and Future Work

CARES is less consistently advantageous under very small replay budgets. When only a few exemplars can be retained, learner-dependent selection signals can provide more targeted sample choices, whereas CARES becomes increasingly competitive as additional replay capacity allows the selected set to represent the feature-space geometry more fully. Developing lightweight hybrid criteria that combine geometric coverage with inexpensive uncertainty or learner-derived information may improve this small-budget regime without sacrificing the efficiency of CARES.

CARES also assumes access to a suitable frozen pretrained encoder. Our sensitivity analysis indicates that performance is relatively stable across the evaluated representations, but selection quality ultimately depends on the structure induced by the pretrained feature space. The staging pool introduces temporary memory beyond the permanent replay budget; however, our sensitivity analysis shows no consistent benefit from excessively large candidate pools, suggesting that moderate staging capacities are sufficient in the evaluated setting.

6 Conclusion

We presented CARES, a learner-optimization-free approach to replay-buffer construction that selects exemplars using class-aware, density-weighted coverage in a frozen pretrained feature space. By decoupling memory selection from the continually trained model, CARES requires no continual learner-derived losses, gradients, backward passes, or auxiliary model training during buffer construction.

Across online task- and class-incremental settings, CARES achieves competitive or superior final accuracy while substantially reducing the cost of memory selection compared with learner-dependent coreset methods. Its advantage is particularly pronounced under class imbalance, and its relative performance improves as additional replay capacity becomes available. Our ablations further show that class-aware allocation is critical for geometric selection under imbalanced streams, while density weighting provides complementary gains in balanced and label-noisy settings.

These results demonstrate that effective replay-buffer construction need not depend on signals from the continual learner itself. Stable pretrained representation geometry provides a simple and computationally efficient alternative, particularly when memory selection must be performed repeatedly throughout an online data stream.

AI USE STATEMENT

In this work, we used generative AI tools to assist with different aspects of the research and manuscript preparation. Specifically, it was used to check and refine the mathematical formulation of the method in Section 3, to provide feedback on the design and presentation of experiments, and to assist in refining and debugging parts of the implementation. Generative AI was also used to edit the manuscript for clarity and readability and to assist with formatting the paper and its references for this template.

The authors reviewed all AI-assisted suggestions before being incorporated. The mathematical formulation was checked against the reference implementation, and all experimental procedures, code changes, results, and interpretations were verified by the authors. We take full responsibility for the final content of this work, including any text, code, claims, or other artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work uses only publicly available image-classification benchmarks (Split CIFAR-10, Split CIFAR-100, and Split Tiny-ImageNet) and involves no human subjects and no personally identifiable data. The proposed method changes which examples a continual learner replays and raises no ethical concerns beyond those of the underlying datasets and of continual-learning applications in general.

REPRODUCIBILITY STATEMENT

Algorithm 1 and Appendix A give the complete memory-update procedure, and Section 3 states every equation the selector evaluates. Appendix B describes both evaluation protocols, including dataset splits, replay capacities, and the class-imbalance and label-noise procedures. Appendix D lists all hyperparameters (Tables 10 and 11), and the evaluation metrics are defined in equation 12. Code will be released upon publication.

REFERENCES

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget, 2018. URL https://arxiv.org/abs/1711.09601.

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval, 2019a. URL https://arxiv.org/abs/1908.04742.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning, 2019b. URL https://arxiv.org/abs/1903.08671.

Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples, 2021. URL https://arxiv.org/abs/2103.17230.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline, 2020. URL https://arxiv.org/abs/2004.07211.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Computer Vision – ECCV 2018*. Springer International Publishing, 2018. ISBN 9783030012526. doi: 10.1007/978-3-030-01252-6_33. URL http://dx.doi.org/10.1007/978-3-030-01252-6_33.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem, 2019a. URL https://arxiv.org/abs/1812.00420.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning, 2019b. URL https://arxiv.org/abs/1902.10486.

Gérard Cornuéjols, Marshall L. Fisher, and George L. Nemhauser. Location of bank accounts to optimize float: An analytic study of exact and approximate algorithms. *Management Science*, 23 (8):789–810, 1977.

Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, STOC, pp. 291–300, 2004.

Tyler L. Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting, 2020. URL https://arxiv.org/abs/1910.02509.

Steven C. Y. Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning, 2019. URL https://arxiv.org/abs/1910.06562.

Mingrui Liu Jie Hao, Kaiyi Ji. Bilevel coreset selection in continual learning: A new formulation and algorithm. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=2dtU9ZbgSN.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, March 2017. ISSN 1091-6490. doi: 10.1073/pnas.1611835114. URL http://dx.doi.org/10.1073/pnas.1611835114.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2009a.

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

Ya Le and Xuan S. Yang. Tiny imagenet visual recognition challenge, 2015. URL https://api.semanticscholar.org/CorpusID:16664790. CS231N course report, Stanford University.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning, 2022. URL https://arxiv.org/abs/1706.08840.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning, 2018. URL https://arxiv.org/abs/1711.05769.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989. URL https://api.semanticscholar.org/CorpusID:61019113.

Brian B. Moser, Tobias C. Nauen, Arundhati S. Shanbhag, Federico Raue, Stanislav Frolov, Joachim Folz, and Andreas Dengel. Subzerocore: A submodular approach with zero training for coreset selection, 2025. URL https://arxiv.org/abs/2509.21748.

Brian B. Moser, Arundhati S. Shanbhag, Stanislav Frolov, Federico Raue, Joachim Folz, and Andreas Dengel. A coreset selection of coreset selection literature: Introduction and recent advances, 2026. URL https://arxiv.org/abs/2505.17799.

Hadi Pouransari and Saman Ghili. Tiny imagenet visual recognition challenge, 2014.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning, 2017. URL https://arxiv.org/abs/1611.07725.

Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting, 2018. URL https://arxiv.org/abs/1805.07810.

Anthony V. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connect. Sci.*, 7:123–146, 1995. URL https://api.semanticscholar.org/CorpusID:22882861.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks, 2022. URL https://arxiv.org/abs/1606.04671.

Jonathan Schwarz, Jelena Luketina, Wojciech M. Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning, 2018. URL https://arxiv.org/abs/1805.06370.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach, 2018. URL https://arxiv.org/abs/1708.00489.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task, 2018. URL https://arxiv.org/abs/1801.01423.

Ghada Sokar, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Spacenet: Make free space for continual learning. *Neurocomputing*, 439, 2021. ISSN 0925-2312. doi: 10.1016/j.neucom.2021.01.078. URL http://dx.doi.org/10.1016/j.neucom.2021.01.078.

Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. Gcr: Gradient coreset based replay buffer selection for continual learning, 2022. URL https://arxiv.org/abs/2111.11210.

Ruilin Tong, Yuhang Liu, Javen Qinfeng Shi, and Dong Gong. Coreset selection via reducible loss in continual learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=mAztx8QO3B.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application, 2024. URL https://arxiv.org/abs/2302.00487.

Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 1121–1128, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553517. URL https://doi.org/10.1145/1553374.1553517.

Ju Xu and Zhanxing Zhu. Reinforced continual learning, 2018. URL https://arxiv.org/abs/1805.12369.

Jaehong Yoon, Saehoon Kim, Eunho Yang, and Sung Ju Hwang. Scalable and order-robust continual learning with additive parameter decomposition, 2020. URL https://arxiv.org/abs/1902.09432.

Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/pdf?id=f9D-5WNG4Nv.

Guodong Zhang, James Martens, and Roger Grosse. Fast convergence of natural gradient descent for overparameterized neural networks, 2019. URL https://arxiv.org/abs/1905.10961.

SUPPLEMENTARY MATERIAL

A CARES ALGORITHM

Algorithm 2 provides the complete memory-update procedure used by CARES. During task t, incoming examples are maintained in a bounded staging pool $\mathcal{P}_t$ using reservoir sampling. At the memory-update point, this pool is combined with the previous replay memory $\mathcal{M}_{t-1}$. If the resulting candidate set exceeds the replay-memory budget K, CARES embeds the candidates using a frozen feature extractor and applies the class-aware, density-weighted coverage criterion described in the main paper.

Algorithm 2 CARES: Coverage-Aware Replay Exemplar Selection

Require: Task sequence $\{\mathcal{D}_t\}_{t=1}^{T}$, replay capacity K, staging-pool capacity P, target coverage γ, frozen feature extractor g_ϕ
Ensure: Continually updated learner f_θ and replay memory $\mathcal{M}_T$

Initialize replay memory $\mathcal{M}_0 \leftarrow \emptyset$
for $t = 1, \ldots, T$ **do**
  Initialize staging pool $\mathcal{P}_t \leftarrow \emptyset$
  for each minibatch $\mathcal{B}_t \subset \mathcal{D}_t$ **do**
    ▷ Standard rehearsal training
    if $\mathcal{M}_{t-1} \neq \emptyset$ **then**
      Sample replay minibatch $\mathcal{B}_{\mathcal{M}} \sim \mathcal{M}_{t-1}$
      $\mathcal{B} \leftarrow \mathcal{B}_t \uplus \mathcal{B}_{\mathcal{M}}$
    else
      $\mathcal{B} \leftarrow \mathcal{B}_t$
    end if
    Update θ using cross-entropy loss on $\mathcal{B}$
    ▷ Bounded current-task staging
    for each $(x, y) \in \mathcal{B}_t$ **do**
      $\mathcal{P}_t \leftarrow \textsc{ReservoirUpdate}(\mathcal{P}_t, (x, y), P)$
    end for
  end for
  ▷ Memory reconstruction
  $\mathcal{A}_t \leftarrow \mathcal{M}_{t-1} \uplus \mathcal{P}_t$
  if $|\mathcal{A}_t| \leq K$ **then**
    $\mathcal{M}_t \leftarrow \mathcal{A}_t$
  else
    Compute normalized frozen embeddings $z_i \leftarrow g_\phi(x_i)/\|g_\phi(x_i)\|_2$ for all $(x_i, y_i) \in \mathcal{A}_t$
    $\mathcal{I}_t \leftarrow \textsc{CARES-Select}(\{(z_i, y_i)\}_{i=1}^{|\mathcal{A}_t|}, K, \gamma)$
    ▷ Selection criterion defined in the main paper
    $\mathcal{M}_t \leftarrow \mathcal{A}_t[\mathcal{I}_t]$
  end if
end for

B EXPERIMENTAL PROTOCOL DETAILS

B.1 PROTOCOL 1:

Protocol 1 follows the experimental setting of BCSR (Jie Hao, 2023). Split CIFAR-100 is divided into 20 tasks containing 5 disjoint classes each, while Tiny-ImageNet is divided into 20 tasks containing 10 disjoint classes each. The total replay-memory capacity is $K = 100$ for Split CIFAR-100 and $K = 200$ for Tiny-ImageNet. We follow the exact imbalance and noise procedures as Jie Hao (2023). Incoming stream batches contain 50 samples on Split CIFAR-100 and 100 samples on Tiny-ImageNet.

Task identity is available at inference, corresponding to task-incremental learning (TIL). For the robustness experiments, we follow the same class-imbalance and label-noise generation procedures as BCSR (Jie Hao, 2023). CARES changes only the replay-memory selection procedure; the backbone,

continual-model optimization, data stream, and evaluation protocol are otherwise kept consistent with the reference setting. Hyperparameters specific to BCSR's bilevel optimization are used only for that baseline and are not applicable to CARES. Results are aggregated over three independent runs.

B.2 Protocol 2

Protocol 2 follows the online-streaming setting of GCR (Tiwari et al., 2022). Split-CIFAR-10 is divided into 5 tasks of 2 classes, Split-CIFAR-100 into 10 tasks of 10 classes, and Split-TinyImageNet into ten tasks of 20 classes.

Data are presented in streams of 6,250 samples with blurred task boundaries, and the learner receives a single pass over each stream. Evaluation is class-incremental: task identity is unavailable at inference, and the model predicts over all classes observed so far. We use a ResNet-18 learner with minibatches of 32 current examples and 32 replay examples, and evaluate replay-memory capacities $K \in \{200, 500, 2000\}$. Following Tiwari et al. (2022), methods that require boundary information may receive it during training, while evaluation remains class-incremental. Results are aggregated over 15 independent runs.

C Additional Results

Table 7: Results on Split-CIFAR-100 under balanced, imbalanced, and label-noise settings. Higher A_T is better, while lower FGT_T is better. Results are reported as mean ± standard deviation. Best results are shown in **bold**.

Method	Balanced		Imbalanced		Label Noise	
	$A_T \uparrow$	$FGT_T \downarrow$	$A_T \uparrow$	$FGT_T \downarrow$	$A_T \uparrow$	$FGT_T \downarrow$
K-means Features	57.82±0.69	0.070±0.003	45.44±0.76	0.037±0.002	57.38±1.26	0.098±0.003
K-means Embedding	59.77±0.24	0.061±0.001	43.91±0.15	0.044±0.001	57.92±1.25	0.091±0.016
Uniform	58.99±0.54	0.074±0.004	44.73±0.11	0.033±0.007	58.76±1.07	0.087±0.006
iCaRL	60.74±0.09	**0.044±0.026**	44.25±2.04	0.042±0.019	59.70±0.70	0.071±0.010
Grad Matching	59.17±0.38	0.067±0.003	45.44±0.64	0.038±0.001	59.58±0.28	0.073±0.008
SPR	59.56±0.73	0.143±0.064	44.45±0.55	0.086±0.023	58.74±0.63	0.073±0.010
MetaSP	60.14±0.25	0.056±0.230	43.74±0.36	0.079±0.014	57.43±0.54	0.086±0.007
Greedy Coreset	59.39±0.16	0.066±0.017	43.80±0.01	0.039±0.007	58.22±0.16	0.066±0.001
GCR	58.73±0.43	0.073±0.013	44.48±0.05	0.035±0.005	58.72±0.63	0.081±0.005
PBCS	55.64±2.26	0.062±0.001	39.87±1.12	0.076±0.011	56.93±0.14	0.100±0.003
OCS	52.57±0.37	0.088±0.001	46.54±0.34	**0.022±0.003**	51.77±0.81	0.103±0.007
BCSR	**61.60±0.14**	0.051±0.015	47.30±0.57	**0.022±0.005**	60.70±0.08	**0.059±0.013**
CARES (Ours)	61.52±1.48	0.111±0.074	**54.88±0.33**	0.0822±0.083	**62.12±1.14**	0.110±0.013

D Implementation Details

Tables 10 and 11 show the hyperparameter values of the best results under each protocol. All experiments were run on a single NVIDIA RTX A6000 GPU with four CPU cores. Unless otherwise stated, the same hardware configuration was used across all methods and experimental settings.

E Sensitivity Analysis

E.1 Replay and Staging-Pool Scaling

We additionally examine how replay capacity K and staging-pool capacity P affect CARES on Split CIFAR-100 and Split-Tiny-ImageNet under Protocol 1. Increasing replay capacity substantially improves performance: at a fixed staging-pool size of $P = 200$, increasing K from 200 to 1000 raises final CIL accuracy from 9.83% to 19.88% and TIL accuracy from 63.60% to 73.63%. In contrast, increasing the staging-pool size at a fixed replay budget provides comparatively modest gains. For $K = 500$, increasing P from 200 to 1000 improves CIL accuracy from 16.06% to 16.72% and TIL accuracy from 69.26% to 70.12%, while increasing the memory-update overhead from 11.2 s to 23.5 s.

Table 8: Results on Split-Tiny-ImageNet under balanced, imbalanced, and label-noise settings. Higher average accuracy is better, while lower forgetting is better. Results are reported as mean ± standard deviation.

Method	Balanced		Imbalanced		Label Noise	
	AVG ACC ↑	FGT ↓	AVG ACC ↑	FGT ↓	AVG ACC ↑	FGT ↓
K-means Embedding	41.48±1.21	0.129±0.007	36.29±0.23	0.085±0.003	36.01±1.51	0.083±0.005
Uniform	42.11±0.52	0.129±0.002	37.07±0.53	0.083±0.009	37.14±1.05	0.099±0.003
iCaRL	43.84±0.09	0.114±0.004	37.65±0.84	0.058±0.003	38.52±0.25	0.063±0.006
Greedy Coreset	41.02±0.33	0.119±0.017	33.43±0.86	0.103±0.002	36.37±0.16	0.079±0.006
GCR	41.45±0.35	0.125±0.008	36.08±0.62	0.072±0.017	37.46±0.40	0.115±0.011
OCS	41.29±0.09	0.112±0.001	35.09±0.60	0.036±0.011	35.36±0.94	0.061±0.005
BCSR	44.13±0.33	0.106±0.001	38.59±0.11	0.070±0.004	40.72±0.56	0.055±0.006
CARES (Ours)	**49.49±0.41**	**0.099±0.015**	**42.94±0.40**	0.104±0.011	**49.37±0.95**	0.078±0.021

Table 9: Online CIL average accuracy (15 seeds). Best results are shown in bold.

Method	S-CIFAR-10			S-CIFAR-100			S-TinyImageNet		
	$K = 200$	$K = 500$	$K = 2000$	$K = 200$	$K = 500$	$K = 2000$	$K = 200$	$K = 500$	$K = 2000$
ER	32.63±5.11	43.32±4.75	49.31±7.45	11.72±1.25	14.94±1.69	20.78±1.48	4.36±1.02	6.69±0.53	11.3±1.45
GEM	16.81±1.17	17.73±0.98	20.35±2.44	10.42±1.66	9.18±1.8	12.38±3.81	5.15±0.71	6.14±1.01	9.56±2.63
GSS	33.83±3.95	40.59±4.20	41.7±4.56	12.31±0.57	13.84±0.57	16.24±0.80	5.82±0.38	6.78±0.41	8.44±0.48
iCaRL	38.66±2.03	33.53±3.17	39.56±1.53	11.23±0.31	11.81±0.29	12.28±0.39	4.39±0.19	5.36±0.27	6.66±0.30
MIR	21.5±0.7	30.52±1.22	46.4±1.81	10.3±0.3	11.4±0.38	16.02±0.5	4.9±0.17	5.1±0.33	7.05±0.30
DER	38.26±5.08	45.56±6.02	49.9±7.55	10.68±2.03	14.43±2.44	24.83±1.58	4.81±0.57	8.49±0.48	**15.97±0.95**
GCR	**42.26±7.6**	**51.17±3.5**	52.3±7.01	**14.88±0.94**	18.89±1.5	22.47±2.01	6.19±0.36	**8.77±0.5**	11.83±1.26
CARES (ours)	40.26±4.67	48.49±5.83	**54.88±6.34**	14.28±0.84	**19.12±1.28**	**25.17±1.39**	**6.60±0.59**	8.46±0.78	15.67±0.85

Table 10: Hyperparameter settings for Protocol 1. Settings shared across stream conditions are reported once per dataset; learning rate, weight decay, coverage, and staging-pool size are selected separately for the balanced, imbalanced, and label-noise settings.

Hyperparameter	Split CIFAR-100			Split Tiny-ImageNet		
	Bal.	Imb.	Noise	Bal.	Imb.	Noise
Number of tasks	20			20		
Classes per task	5			10		
Replay capacity K	100			200		
Incoming stream batch size	50			100		
Training batch size	*10*			*20*		
Epochs per stream batch	1			1		
Backbone	ResNet-18			ResNet-18		
Number of runs	3			3		
Learning rate	*0.03*	*0.03*	*0.05*	*0.03*	*0.03*	*0.05*
Weight decay	*1e-5*	*1e-5*	*1e-5*	*1e-5*	*5e-5*	*5e-5*
Coverage γ	*0.4*	*0.4*	*0.4*	*0.2*	*0.4*	*0.2*
Staging-pool size P	*200*	*200*	*200*	*400*	*400*	*400*

These results suggest that CARES benefits primarily from additional persistent replay capacity, while very large staging pools offer diminishing returns.

E.2 Joint Sensitivity to Coverage and Staging-Pool Capacity

We further study the interaction between the coverage parameter γ and staging-pool capacity P on balanced Split CIFAR-100. We fix the replay capacity to $K = 100$, the learning rate to 0.02, and weight decay to 5×10^{-5}, while jointly varying $\gamma \in \{0.2, 0.3, 0.4, 0.6, 0.8\}$ and $P \in \{K, 2K, 5K, 10K, 20K\}$.

Figure 2 shows that strong performance is obtained over a relatively broad region of the parameter space rather than at a single narrow operating point. In particular, staging pools of $2K$–$5K$ consistently achieve high final TIL accuracy across multiple coverage values. The highest mean

Table 11: Hyperparameter settings for Protocol 2.

Hyperparameter	**Split CIFAR-10**	**Split CIFAR-100**	**Split Tiny-ImageNet**
Number of tasks	5	10	10
Replay capacity K	{200, 500, 2000}	{200, 500, 2000}	{200, 500, 2000}
Training batch size	32	32	32
Incoming stream batch size	6250	6250	6250
Training epochs	1	1	1
Learning rate	0.05	0.05	0.05
Optimizer	SGD	SGD	SGD
Weight decay	1e-5	1e-5	1e-5
Coverage γ	0.4	0.4	0.4
Staging-pool size P	400	400	400
Backbone	ResNet-18	ResNet-18	ResNet-18
Number of runs	15	15	15

Table 12: Scaling of CARES with replay capacity K and staging-pool capacity P on Split CIFAR-100 under Protocol 1. We report final class-incremental (CIL) and task-incremental (TIL) accuracy and the cumulative memory-update overhead. All other hyperparameters are held fixed. Results are reported as mean $\pm$ standard deviation over n completed runs.

K	P	**CIL Acc.** $\uparrow$	**TIL Acc.** $\uparrow$	**Update Overhead (s)** $\downarrow$
200	200	9.83±0.42	63.60±0.70	6.8±1.0
	500	9.78±0.63	64.16±1.90	12.4±2.1
	1000	10.47±0.00	62.46±0.00	17.6±0.0
500	200	16.06±0.90	69.26±0.56	11.2±1.9
	500	16.24±1.33	69.37±1.02	15.1±7.3
	1000	16.72±1.50	70.12±0.97	23.5±13.4
1000	200	19.88±0.14	73.63±0.85	14.7±3.1
	500	20.08±0.00	73.90±0.11	23.2±7.2
	1000	20.05±0.00	73.92±0.00	44.8±0.0

accuracy is obtained at $\gamma = 0.4$ and $P = 2K$, but several neighboring configurations perform similarly. Increasing the staging pool further to $10K$ or $20K$ does not provide a consistent accuracy improvement. These results suggest that CARES does not require precise tuning of either γ or P, and that a moderately sized staging pool is sufficient to obtain strong performance.

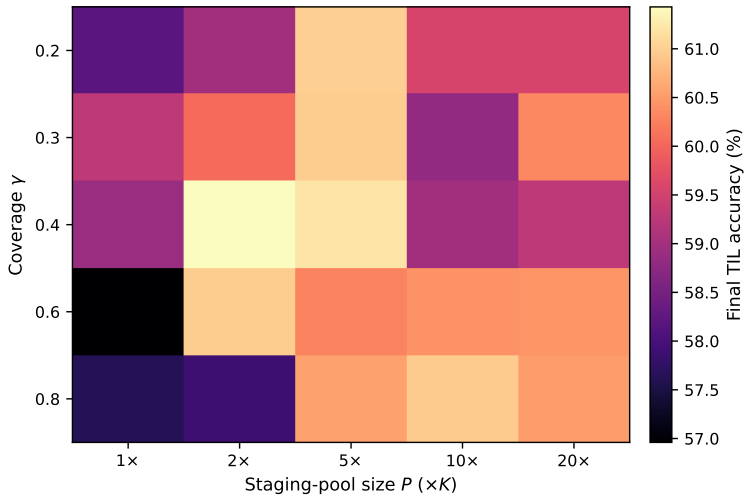


Figure 2: Joint sensitivity of CARES to the coverage parameter γ and staging-pool capacity P on balanced Split CIFAR-100. Each cell represents final task-incremental accuracy, with P expressed relative to replay capacity $K = 100$. CARES exhibits a broad region of strong performance, while increasing the staging pool beyond moderate sizes provides no consistent benefit.