Gradual Divergence for Seamless Adaptation: A Novel Domain Incremental Learning Method

Kishaan Jeeveswaran [1] **Elahe Arani** [* 1 2] **Bahram Zonooz** [* 1]

Abstract

Domain incremental learning (DIL) poses a significant challenge in real-world scenarios, as models need to be sequentially trained on diverse domains over time, all the while avoiding catastrophic forgetting. Mitigating representation drift, which refers to the phenomenon of learned representations undergoing changes as the model adapts to new tasks, can help alleviate catastrophic forgetting. In this study, we propose a novel DIL method named *DARE*, featuring a three-stage training process: Divergence, Adaptation, and REfinement. This process gradually adapts the representations associated with new tasks into the feature space spanned by samples from previous tasks, simultaneously integrating task-specific decision boundaries. Additionally, we introduce a novel strategy for buffer sampling and demonstrate the effectiveness of our proposed method, combined with this sampling strategy, in reducing representation drift within the feature encoder. This contribution effectively alleviates catastrophic forgetting across multiple DIL benchmarks. Furthermore, our approach prevents sudden representation drift at task boundaries, resulting in a well-calibrated DIL model that maintains the performance on previous tasks. [1]

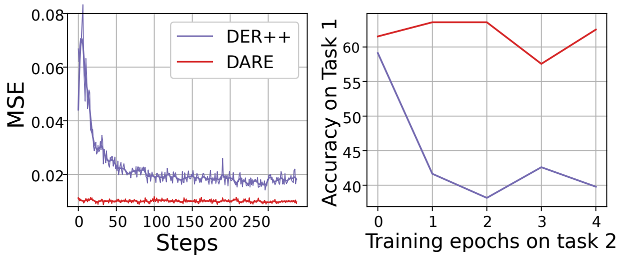

Figure 1. Relationship between representation drift and task 1 accuracy on DN4IL dataset with buffer size 50. The representations of buffered samples, mainly belonging to the first domain, experience an abrupt drift at the task boundary, which is directly associated with the decrease in accuracy.

1. Introduction

Domain incremental learning (DIL) is a subset of continual learning (CL) that addresses the challenge of acquiring knowledge from new domains or tasks in an incremental manner without forgetting previously acquired knowledge. DIL holds significance in real-world applications like autonomous driving and robotics, where data distribution can shift due to factors like changing weather condition and location (Mirza et al., 2022). Deep neural networks (DNNs) suffer from the problem of catastrophic forgetting, where the weights of the network associated with old tasks are overwritten by new information, resulting in a decline in performance for previously learned tasks.

Various approaches have been proposed to alleviate catastrophic forgetting, which can be grouped into three main categories: interleaving past task samples during new task learning (Experience Replay) (Ratcliff, 1990; Rebuffi et al., 2017), constraining the change in weights of DNNs pertinent to past tasks (Regularization) (Kirkpatrick et al., 2017; Li & Hoiem, 2017), or expanding the architecture with new branches for learning new tasks without overwriting existing task parameters (Architecture Expansion) (Rusu et al., 2016; Fernando et al., 2017). Although experience replay has been shown to effectively mitigate catastrophic forgetting, it does not explicitly address drift in representations at task boundaries caused by the disruption of clustered representations corresponding to previously learned tasks (Caccia et al., 2022). Representation drift is directly correlated with performance drop on old tasks and contributes significantly to catastrophic forgetting (see Figure 1).

DNNs aim to acquire clustering representations for similar classes in each task. However, the clusters formed by previous tasks may shift when learning new classes, resulting in

[*] Equal contribution [1] Dep. of Mathematics and Computer Science, Eindhoven University of Technology, NL [2] Wayve Technologies Ltd, London, UK. Correspondence to: <j.kishaan@tue.nl>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1] Code at https://github.com/NeurAI-Lab/DARE.

a decline in accuracy for old tasks, as observed in new domain learning in DIL (Figure 1). The issue of representation drift is addressed in CL literature through various methods. Caccia et al. (2022) proposed isolating new samples from the old buffer samples and employing distinct loss functions to mitigate drift. This approach hinges on separating old and new classes and the quality of negative samples for the proposed semisupervised loss, limiting its applicability in DIL scenarios. Furthermore, representation drift in DIL remains unexplored in the existing literature. To address this, we suggest a three-stage training process to gradually adapt the learning model to new domain sample representations.

Concretely, we propose a novel approach for mitigating abrupt representation drift and catastrophic forgetting in DIL by adapting the representations of new domain samples into the feature space spanned by the old domains. The proposed method employs a three-stage training process (**D**ivergence, **A**daptation, **RE**finement) while learning new domains. During the Divergence and Adaptation stages, the model clusters the representations of new domain samples into the feature space spanned by the first domain, while the Refinement stage helps the model learn the new domain samples. Our method, DARE, helps to mitigate changes to the representations of old domains, resulting in better overall accuracy. Furthermore, we propose an effective buffer sampling strategy to integrate the proposed algorithm into the CL framework. By using this strategy, we can store important samples in the buffer that capture the maximum information about the "dark knowledge" between data samples. Specifically, our contributions are as follows:

- We propose a novel domain incremental learning method to effectively adapt the representations of the new domain into the feature space spanned by the prior domains.
- We, through extensive analyses, demonstrate the effectiveness of our method in mitigating forgetting, task recency bias, and suppressing detrimental representation drifts at task boundaries in DIL.
- We propose and employ an effective buffer sampling strategy that maximizes the information stored in the buffer without significant memory overhead.

2. Related Works

Domain Incremental Learning. DIL studies the ability of DNNs to continually adapt to new domain data while preserving performance on prior domains, such as adapting to different weather conditions (Mirza et al., 2022). Many approaches in DIL rely on learning task-specific information and plugging it during inference. DISC (Mirza et al., 2022) stores domain-specific batch norm statistics and uses them during inference to detect objects under different weather conditions. Garg et al. (2022) use a dynamic architecture for domain incremental segmentation by learning both domain-invariant and domain-specific parameters. However, these approaches require task-id during inference, which violates the CL desiderata (Farquhar & Gal, 2018).

Approaches to address catastrophic forgetting in CL can be broadly divided into three categories: regularization-based (Kirkpatrick et al., 2017; Li & Hoiem, 2017), parameter isolation (Rusu et al., 2016; Fernando et al., 2017), and rehearsal-based (Ratcliff, 1990; Rebuffi et al., 2017) methods. Regularization-based methods can be viewed as a way to shield the weights and therefore the learned representations for previous tasks from interference while learning new tasks. However, these methods are often overly focused on previous tasks, and the limited capacity of Deep Neural Networks (DNNs) makes them inflexible for learning new tasks (Parisi et al., 2019). Rehearsal-based methods are more popular in the literature due to their simplicity and superior performance (Buzzega et al., 2021; Cha et al., 2021). However, they lack a mechanism to explicitly tackle representation drift (Caccia et al., 2022). Parameter-isolation methods allocate a distinct set of parameters for new tasks, but they become memory-intensive as the number of tasks increases. Overall, while each category of methods has its own advantages and disadvantages, none of them fully address representation drift efficiently. Therefore, novel strategies are required to effectively tackle this challenge, upholding performance across new and old tasks.

Representation Drift. In the context of CL, representation drift is a phenomenon in which previously learned representation clusters tend to drift while learning new tasks. Murata et al. (2020) propose a representation-based evaluation framework to evaluate the impact of representation drift by freezing different layers after CL and retraining the remaining layers on all tasks. Caccia et al. (2022) propose two loss functions to mitigate representation drift in online class incremental learning (CIL) by learning new task samples separate from the buffered task samples. However, these methods are not directly applicable to DIL, where there is no separation between the seen and the new classes. To address this issue, Yu et al. (2020) use metric learning to mitigate feature drift in CIL, while Zhang et al. (2022) propose a framework to quantify feature forgetting in CL and learn separate task-wise adapters to combat feature forgetting. However, these approaches entail memory overhead, which grows with the number of tasks.

Domain Adaptation. Domain Adaptation (DA) aims to transfer knowledge learned from a source dataset to a target dataset with a related domain. Saito et al. (2018) propose a dual classifier setup for DA. The training process alternates between maximizing the discrepancy between classifiers for out-of-domain samples and minimizing the discrepancy for in-domain samples by freezing the classifiers and learn-

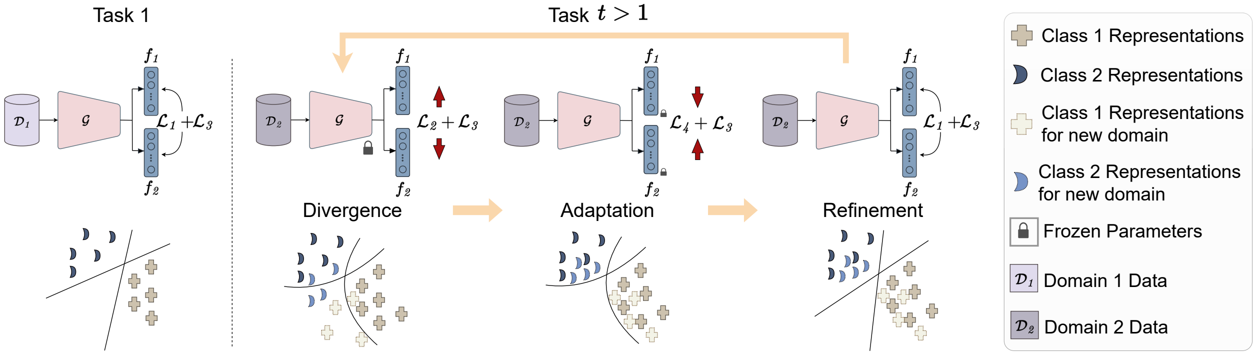

Figure 2. Our proposed method, *DARE*, assimilates the knowledge about the new task while preserving the representations from earlier tasks by adopting a three-stage learning process in DIL. In the first two stages, *Divergence* and *Adaptation*, the model learns the representations of new domains within the cluster of old ones (rather than the other way around, which can exacerbate catastrophic forgetting). The final stage, *Refinement*, helps the model learn the new domain samples.

ing the feature encoder to predict the same logits in both classifiers. In this way, the model adapts the representation of new domains in the space of old ones. Subsequently, Yang et al. (2021) proposed an extension with more than two classifiers to improve accuracy. Following the dual classifier approach, Lv et al. (2022) propose a causally inspired framework with dual classifiers for DA. This framework aims to learn domain-independent representations in the encoder and to learn dual classifiers on complementary features using an adversarial mask. While DA focuses on the forward transfer of knowledge from old tasks to new tasks, DIL focuses on both forward and backward transfers, where the model must retain the knowledge of old tasks.

3. Methodology

Domain incremental learning (DIL) involves a sequence of T tasks that become progressively available over time, with each task representing a shift in the input data distribution while the classes remain constant across tasks. During each task $t \in \{1, 2, .., T\}$, samples and their corresponding labels $\{(x_i, y_i)\}_{i=1}^{N}$ are drawn from the task-specific distribution $\mathcal{D}_t$ (Van de Ven & Tolias, 2019). The CL model is optimized sequentially on each domain, and inference is carried out on all domains seen so far. An optimal CL model would learn to predict new distributions of input samples while retaining its knowledge of the initial tasks.

3.1. Proposed Method - DARE

Our method aims to enable efficient DIL by mitigating the abrupt representation drift at the task boundaries and adapting the learned representations to consolidate new information in a manner that reduces interference. To this end, we propose *DARE* which employs a three-stage (*Divergence*, *Adaptation*, and *REfinement*) learning mechanism that encourages the model to learn the representations of the samples belonging to the new task in the subspace spanned by the representations of the old tasks. This allows the model to consolidate new information without considerably disrupting the representations of the old tasks and adapt the decision boundary for the consolidated representations.

DARE utilizes an encoder g to extract semantically meaningful representations from the input image, and dual classifiers f_1 and f_2 to project these representations to the class distribution (as depicted in Figure 2). To learn more general and robust representations, and enforce the two classifiers to have different decision mechanisms, we employ the cross-entropy loss in the first classifier f_1 and the supervised contrastive loss (Khosla et al., 2020) in the second classifier, f_2. This equips our method with multiple viewpoints of the input data and ensures that the two classifiers sufficiently diverge. Furthermore, supervised contrastive loss provides the benefit of learning generalizable features across different domains while facilitating the learning of discriminative features across different classes (Cha et al., 2021). Hence, the two learning objectives complement each other.

In addition, we employ a buffer with bounded memory in which we store a portion of the learning task samples, labels, and logits from both classifiers. To this end, we propose an effective buffer sampling strategy, called "*Intermediary Reservoir Sampling*" strategy (see Section 3.2) throughout the training process to sample from the current task data and store them in a buffer. This way, the memory buffer contains samples from past tasks that are replayed later during the training process.

Concretely, the first task is learned with the combination of cross-entropy loss on f_1 and supervised contrastive loss on f_2 on shared representations, where $z_i = g(x_i)$:

$$\mathcal{L}_1 \triangleq \mathop{\mathbb{E}}_{(x_i,y_i)\sim\mathcal{D}_t} \left[\mathcal{L}_{ce}(f_1(z_i), y_i) + \mathcal{L}_{sup}(f_2(z_i), y_i) \right] \quad (1)$$

For subsequent tasks, learning unfolds in three stages that enable CL by helping the model effectively adapt to new tasks while preserving prior knowledge.

3.1.1. Divergence

The divergence stage aims to tighten the decision boundaries of the two classifiers around the representation space spanned by the samples of already learned tasks. This involves maximizing the divergence between the two classifiers so that they can identify incoming samples from the new tasks whose representations do not lie in the space spanned by the learned representations.

Specifically, we fix the parameters of the encoder, g, and maximize the distance between the ℓ_2-normalized logits predicted by f_1 and f_2. The discrepancy loss (Tan et al., 2022) measures the disparity in the distributions of pairwise distances of the classifier outputs $f_1(z)$ and $f_2(z)$. Let d_1 be the pairwise distance between the ℓ_2-normalized logits predicted by classifier f_1 for a batch of input samples $\mathcal{X}$:

$$d_1 = \|f_1(z_i) - f_1(z_j)\| \tag{2}$$

where $\|.\|$ denotes Euclidean distance. The similarity metrics $p(.)$ can then be modeled as a normal distribution:

$$p(d_1) = C_1 \frac{1}{\sigma_1\sqrt{2\pi}} exp\left[-\frac{1}{2}\frac{(d_1-\mu_1)^2}{\sigma_1^2}\right] \tag{3}$$

where C_1 is a constant and μ_1, and σ_1^2 are set to 0 and $\frac{1}{2}$ following (Tan et al., 2022). Letting d_2 be the pairwise distance between the ℓ_2-normalized logits predicted by the classifier f_2 and $q(.)$ be the corresponding similarity metrics, the discrepancy loss defined as:

$$\mathcal{L}_2 \triangleq \underset{\mathcal{X}\sim D_t}{\mathbb{E}}\left[p(d_1)\text{log}q(d_2) + (1-p(d_1))\text{log}(1-q(d_2))\right] \tag{4}$$

Intuitively, as illustrated in Figure 2, the divergence stage tightens the decision boundaries around the previous task samples by forcing the classifiers to maximize the distance in the predictions for the samples from the new task, while maintaining correct predictions on the previous tasks. This is achieved by minimizing the cross-entropy loss (f_1), supervised contrastive loss (f_2), and a consistency loss on the buffer samples in addition to maximizing the discrepancy loss on the new task samples:

$$\begin{aligned}\mathcal{L}_3 \triangleq \underset{(x_i',y_i',\zeta_{1,2}')\sim D_m}{\mathbb{E}} &\big[\alpha\|\zeta_{1,2}' - f_{1,2}(z_i')\|_2 \\ &+ \mathcal{L}_{ce}(f_1(z_i'), y_i) + \mathcal{L}_{sup}(f_2(z_i'), y_i)\big]\end{aligned} \tag{5}$$

where ζ_1 and ζ_2 are f_1 and f_2's saved logits in the buffer, and α is a weighting parameter. The consistency loss encourages the classifiers to adapt their decision boundaries while maintaining the semantic relationships between the classes and enforces them to remain close to the optimal solution found for previous task samples in the memory buffer. Hence, the overall loss for the divergence stage is given by $\mathcal{L}_{\mathcal{D}} = \mathcal{L}_2 + \mathcal{L}_3$.

3.1.2. Adaptation

The Divergence stage is followed by the Adaptation stage, which aims to adapt the encoder g so that the representations of the new task samples are adapted within the subspace spanned by the already learned representations of previous tasks. Hence, the goal is to learn a consolidated representation space that supports the samples of the new tasks while remaining close to the optimal representations for the previously learned tasks. This is achieved by freezing the classifiers f_1 and f_2, and minimizing the discrepancy between their predictions. This enforces the encoder g to adapt the representations so that the two classifiers agree on their predictions. The corresponding minimization loss for the discrepancy between f_1 and f_2 is given by;

$$\mathcal{L}_4 \triangleq -\underset{\mathcal{X}\sim D_t}{\mathbb{E}}\left[p(d_1)\text{log}q(d_2) + (1-p(d_1))\text{log}(1-q(d_2))\right] \tag{6}$$

Divergence and Adaptation can also be interpreted as a form of adversarial learning in which, first, the discriminators f_1 and f_2 are trained to discriminate the samples of the new task from those belonging to the old tasks by maximizing the discrepancy between the classifiers. Consequently, the generator, g, is trained to deceive the discriminators by extracting features that minimize the discrepancy between the two classifiers. Thus, during the course of learning the new task, the representations of the new task samples are gradually adapted to lie within the support spanned by the representations of previously learned tasks, rather than the other way around, which effectively reduces the drift in representation, and hence mitigates forgetting. The total loss for this stage is given by $\mathcal{L}_{\mathcal{A}} = \mathcal{L}_3 + \mathcal{L}_4$.

3.1.3. Refinement

Finally, the Refinement stage aims to refine the encoder and classifiers to effectively consolidate the new task information with the previously learned knowledge such that a learned consolidated representation space and decision boundary perform well for all the tasks seen so far. This involves training the encoder g, and the classifiers f_1 and f_2 to predict the correct classes for new task samples, while also minimizing a consistency loss with respect to the stored samples in the buffer. This encourages the model to learn the new task while maintaining previously acquired knowledge. The loss used at this stage is $\mathcal{L}_{\mathcal{R}} = \mathcal{L}_1 + \mathcal{L}_3$.

Note that we iterate through the three stages multiple times while learning each task. This enables the model to gradually adapt the representations and decision boundary to acquire and consolidate information from the new task while mitigating the drift in representations and hence forgetting. Our proposed method is detailed in Algorithm 1.

3.2. Intermediary Reservoir Sampling

The proposed method to populate the replay buffer in DER utilizes Reservoir Sampling (Vitter, 1985). However, this uniform distribution throughout the learning trajectory does not optimize the storage of the exemplars. There is a nontrivial probability that logits are stored in the buffer at the very beginning or end of learning a task, leading to suboptimal performance. To improve this, we propose the "*Intermediary Reservoir Sampling (IRS)*" strategy, which employs a normal distribution over the learning trajectory of each task. The mean of the distribution is set to the intermediate stages, and the buffer is populated accordingly. This incentivizes the storage of logits with more "dark knowledge" about the current task, which in turn propagates the knowledge across future tasks through distillation. This approach aligns with recent research in knowledge distillation (Wang et al., 2022), which suggests distilling with respect to an intermediate teacher model to capture maximum information on the "dark knowledge" between data samples. We defer Algorithm 2 and ablation studies for IRS (Table 3) to Appendix.

4. Experimental Setup

We address the issue of sudden changes in data representation that occur with the introduction of new domains. To tackle this problem, we propose a novel approach to mitigate drift, and our results demonstrate that addressing this issue leads to improved performance on standard DIL benchmarks. To perform our experiments, we use the *mammoth* framework (Buzzega et al., 2020) to emulate DIL scenarios and implement our approach on top of the ResNet-18 architecture (He et al., 2016), following previous works (Buzzega et al., 2020; Rebuffi et al., 2017). We modify the network to include our proposed approach, in which the encoder g retains the default ResNet-18 structure, and the classification heads f_1 and f_2 are linear layers projecting the encoded representations from g to a number of classes C, such that $f_{1,2} : \mathbb{R}^d \to \mathbb{R}^C$, where d is the dimension of flattened representations from the encoder. We train our method with a batch size of 32, for 50 epochs per task on all datasets.

We evaluate our proposed method in DIL setting (Van de Ven & Tolias, 2019) on two diverse datasets. **DN4IL** (DomainNet for Domain-IL) is a challenging dataset consisting of six vastly diverse domains and samples belonging to 100 classes (Gowda et al., 2023). On the other hand, **iCIFAR-20** (Xie et al., 2022) is the DIL setup of the CIFAR-100 dataset (Krizhevsky et al., 2009), where the 20 supercategories are considered actual classes and the five subcategories are considered new domains. We focus on evaluating models on datasets that closely mimic real-world domain shifts, as opposed to the transformed versions of MNIST commonly used in the literature. More information about the datasets and training is deferred to Appendix.

Algorithm 1 Learning Algorithm for DARE

```
input: Data streams $\mathcal{D}_t$, model with backbone $g$ and two classifiers $f_1$, $f_2$ parameterized by $\theta$, memory buffer $\mathcal{M} = \{\}$
for all tasks $t \in \{1, 2, .., T\}$ do
  for epochs $e \in \{1, 2, .., E\}$ do
    if $t = 1$ then
      for batch $(x_t, y_t) \in \mathcal{D}_t$ do
        Compute $\mathcal{L}_1$
        Update the model based on $\nabla_\theta \mathcal{L}_1$
      end for
    else
      if $e\%3 == 0$ then
                                                  ▷ Divergence
        for batch $(x_t, y_t) \in \mathcal{D}_t$ do
          Freeze the encoder $g(.)$
          $\zeta_1, \zeta_2 = f_1(x_t), f_2(x_t)$
          Sample batch $(x', y', \zeta'_{1,2}) \in \mathcal{M}$
          Compute $\mathcal{L}_\mathcal{D} = \mathcal{L}_3 + \mathcal{L}_2$
          Update classifiers based on $\nabla_\theta \mathcal{L}_\mathcal{D}$
        end for
      else if $e\%3 == 1$ then
                                                  ▷ Adaptation
        Unfreeze the model
        for batch $(x_t, y_t) \in \mathcal{D}_t$ do
          Freeze the classifiers $f_1, f_2$
          $\zeta_1, \zeta_2 = f_1(x_t), f_2(x_t)$
          Sample batch $(x', y', \zeta'_{1,2}) \in \mathcal{M}$
          Compute $\mathcal{L}_\mathcal{A} = \mathcal{L}_3 + \mathcal{L}_4$
          Update the backbone based on $\nabla_\theta \mathcal{L}_\mathcal{A}$
        end for
      else if $e\%3 == 2$ then
                                                  ▷ Refinement
        Unfreeze the model
        for batch $(x_t, y_t) \in \mathcal{D}_t$ do
          $\zeta_1, \zeta_2 = f_1(x_t), f_2(x_t)$
          Sample batch $(x', y', \zeta'_{1,2}) \in \mathcal{M}$
          Compute $\mathcal{L}_\mathcal{R} = \mathcal{L}_1 + \mathcal{L}_3$
          Update the model based on $\nabla_\theta \mathcal{L}_\mathcal{R}$
        end for
      end if
    end if
    Update $\mathcal{M} \leftarrow$ IRS$(x, y, \zeta_{1,2})$        ▷ Algorithm 2
  end for
end for
return: model $\theta$
```

Table 1. Results on DIL benchmarks learned with varying buffer sizes, averaged over 3 runs. Accuracy determines the performance on all tasks learned by the model, and backward transfer (BWT) quantifies the degree to which learning a new task improves performance on previously learned tasks. #P denotes the total count of trainable parameters (expressed in millions). [3]

Buffer Size	Method	iCIFAR-20			DN4IL		
		#P ↓	BWT ↑	Last Accuracy ↑	#P ↓	BWT ↑	Last Accuracy ↑
-	Joint	11.18	-	79.61±0.13	11.22	-	59.93±1.07
	SGD	11.18	-43.72±1.07	49.40±0.53	11.22	-42.42±0.00	21.63±0.42
50	ER	11.18	-42.03±0.27	50.23±0.94	11.22	-36.11±0.26	24.24±0.34
	DER++	11.18	-40.63±0.49	52.68±1.10	11.22	-29.05±1.35	28.08±0.99
	DARE	11.19	**-34.98**±1.52	**53.66**±0.59	11.27	**-22.98**±0.62	**32.32**±0.53
	CLS-ER	33.57	-	**63.01**±0.80	33.81	-	37.90±1.15
	DUCA	33.57	-	61.48±0.25	33.81	-	38.91±2.12
	DARE++	22.38	-	62.43±0.37	22.54	-	**40.51**±0.17
100	ER	11.18	-41.88±0.59	50.85±0.73	11.22	-35.28±1.20	24.67±0.86
	DER++	11.18	-37.33±1.47	55.32±0.69	11.22	-27.78±0.90	32.06±1.05
	DARE	11.19	**-33.20**±0.09	**56.01**±0.22	11.27	**-19.37**±0.43	**37.16**±0.62
	CLS-ER	33.57	-	64.31±0.43	33.81	-	39.30±0.74
	DUCA	33.57	-	62.59±0.27	33.81	-	43.09±0.14
	DARE++	22.38	-	**64.59**±0.24	22.54	-	**43.27**±0.37
200	ER	11.18	-38.98±0.74	52.57±0.79	11.22	-32.35±0.51	27.45±0.94
	DER++	11.18	-33.61±0.64	58.39±0.38	11.22	-23.99±0.74	35.74±0.67
	DARE	11.19	**-30.22**±1.84	**58.53**±1.25	11.27	**-14.69**±0.19	**40.59**±0.73
	CLS-ER	33.57	-	**66.40**±0.81	33.81	-	41.70±1.41
	DUCA	33.57	-	66.04±0.36	33.81	-	**44.45**±0.18
	DARE++	22.38	-	65.79±0.92	22.54	-	44.11±0.98

5. Empirical Results

We compare our approach with state-of-the-art rehearsal-based methods in CL literature under uniform experimental settings, focusing on the challenging low buffer regime where representation drift is most pronounced (Caccia et al., 2022). For a comprehensive study, we selected standard methods such as ER (Riemer et al., 2018), DER++ (Buzzega et al., 2020), CLS-ER (Arani et al., 2022), and DUCA (Gowda et al., 2023). CLS-ER uses slow and fast learners to distill knowledge from past tasks, while DUCA is a multimemory system that integrates shape cognitive bias. To consolidate learned knowledge, we employed a semantic memory, an exponential moving average (EMA) of the learning model, comparing it with CLS-ER and DUCA. Our proposed method is 'DARE,' and 'DARE++' represents the results of the EMA model in an extended dual-memory version. We also report both the upper bound, denoted *Joint*, where training uses the entire dataset, and the lower bound, denoted *SGD*, where training progresses through new domains without an additional buffer.

Table 1 presents the performance of DARE and other baselines on DIL benchmarks. The results indicate consistent improvements in final accuracy (over all seen tasks) and backward transfer (BWT) when using a single learning model (DARE) with different buffer sizes. Additionally, DARE++ achieves comparable or even better results than other multi-memory based approaches. Notably, CLS-ER and DUCA require storing all multi-memory models in the device, leading to high memory requirements reflected in the number of parameters in the framework. The efficacy of DARE++ is evident from its performance on par with other multi-memory systems with significantly lower parameters.

In the challenging scenario where the buffer size is limited to 50 in DN4IL, our proposed method, DARE, demonstrates significant improvements of 33.3% and 15.1% in accuracy over ER and DER++, respectively. Additionally, DARE trained with a smaller buffer outperforms the DER++ counterparts trained on larger buffer sizes. Similarly, DARE++ trained with a smaller buffer size outperforms CLS-ER trained with a larger buffer size. DN4IL is a highly challenging DIL dataset with significant domain shifts, and these improvements demonstrate the effectiveness of DARE.

Our results clearly indicate that DARE can effectively learn new domains while maintaining high performance on old tasks, even under complex and memory-restrictive settings. This can be attributed to our learning algorithm preserving representations of old tasks while acquiring new ones.

[3] BWT numbers for methods that include an EMA model are not mentioned due to the stochastic nature of the EMA update.

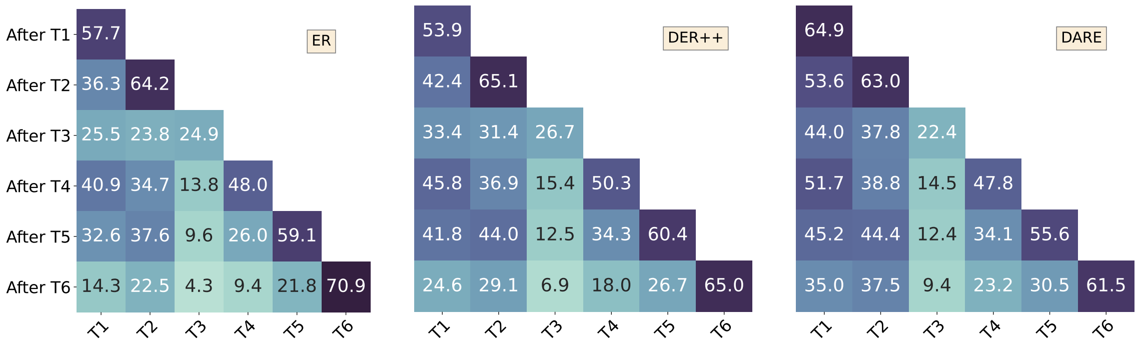

Figure 3. Task-wise accuracy of different CL models while learning new tasks with buffer size 50. DARE retains more performance on seen domains compared to ER and DER++.

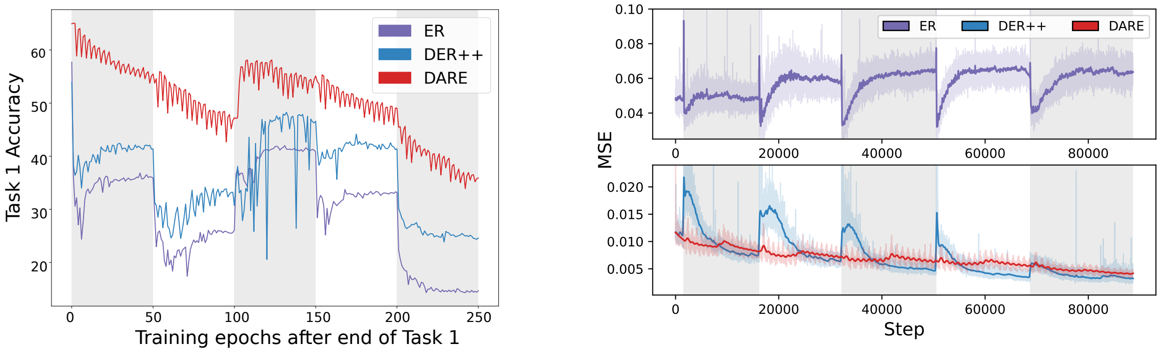

Figure 4. **Representation drift analysis**. Left: Epoch-wise accuracy on Task 1 samples, while learning future tasks (shaded regions indicate new tasks). Right: Iteration-wise drifts for buffered samples for CL methods trained with a buffer size of 50. It is evident that DARE effectively reduces representation drift compared to other methods.

6. Model Analysis

We evaluate the effectiveness of our proposed approach in challenging scenarios through various analysis experiments, comparing its performance to single-model approaches.

6.1. Task-wise Performance

The extreme difference between the domains in every task warrants the study of the model's knowledge about the seen tasks while learning new tasks. Figure 3 shows the task-wise accuracy of different CL approaches while learning new tasks. DARE retains the accuracy of old tasks better compared to ER and DER++. Furthermore, learning task 4 helps improve performance on task 1 across all CL algorithms, and this can be attributed to the similar nature of task 4 (painting) to task 1 (real). It is worth noting that DARE achieves higher performance in task 1 compared to DER++ which employs almost a similar learning algorithm except for the proposed IRS strategy. The consistency loss with respect to intermediate-stage checkpoints helps to learn the first task better than other approaches.

6.2. Study of Representation Drift

The representations learned for previous tasks in the backbone denote the knowledge of the model about the relationship between the input samples and the labels drawn from the data distribution of previous tasks. Modification to important weights in the network for the previous task while learning the new task is deemed to contribute to catastrophic forgetting in CL (McCloskey & Cohen, 1989). Thus, analyzing the change in the representations of past tasks would shed some light on the amount of catastrophic forgetting.

We analyze the representation drift of early task samples in two ways. First, we plot the accuracy of the task 1 validation set over the course of training in the other domains in Figure 4 (left). This represents the disruptive nature of representation drift at task boundaries and their detrimental effect on the accuracy of seen tasks. It can be seen that both ER and DER++ undergo a significant decrease in performance for task 1 samples at the beginning of learning task 2. The same behavior is observed at the beginning of tasks 3, 5, and 6. However, DARE prevents such loss inaccuracy for task 1

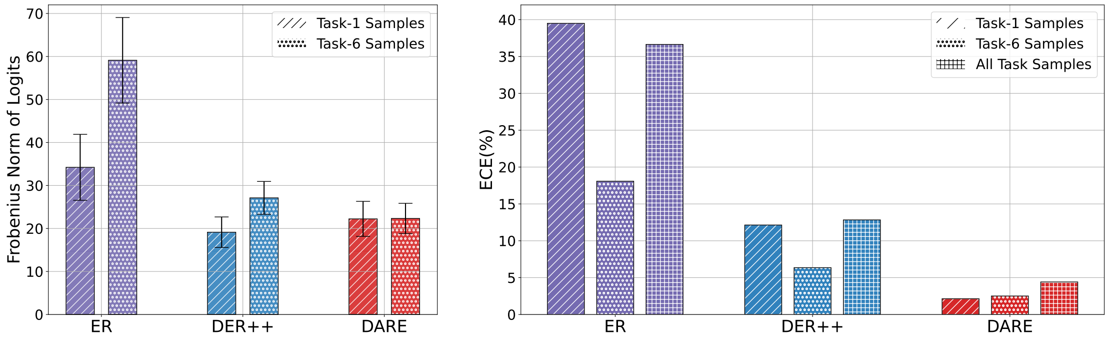

Figure 5. Model calibration and task recency bias analyses of different CL approaches learned with buffer size 200. Left: Logit norm analysis shows that DARE predicts logits with magnitudes smaller than DER++ (less overconfident) for recent task samples. Right: DARE has a lower calibration error compared to DER++ on samples belonging to different tasks.

samples while learning new tasks, as it inhibits disruptive updates to learned representation by design. We observe an increase in task 1 accuracy at the beginning of task 4 in all methods. This is explained by the backward transfer of task 4 (painting), which has features similar to task 1 (real) compared to other domains. DARE is flexible enough to allow for such a backward transfer of knowledge.

Second, we examine the iteration-wise representation drift of the buffered samples in Figure 4 (right). The plot reveals that buffered sample representations experience a sudden shift at task boundaries (indicated by alternating shaded regions). However, DARE, with and without IRS, doesn't exhibit a comparable representation change. As we adjust future task representations into the initial task's representation space, the drift is minimal and gradual, contributing to reduced forgetting. Preventing harmful alterations in representations for samples from the same class set but different domains supports accurate sample classification.

6.3. Task Recency and Model Calibration

Task recency bias is an important concern in CL, wherein model predictions are biased more towards recent tasks and result in more forgetting for earlier task samples (Wu et al., 2019). In DIL, task recency bias can have severe consequences, particularly in safety-critical applications such as autonomous driving, where forgetting knowledge about old tasks can lead to misclassifications. Therefore, it is imperative to develop effective strategies to evaluate and address task recency bias in DIL, especially to ensure the reliability and safety of deployed models.

Table 2. Ablation study on the effect of loss components on the last accuracy of DARE, averaged over 3 runs.

Buffer size	DARE	- $\mathcal{L}_1$	- $\mathcal{L}_2$	- $\mathcal{L}_3$	- $\mathcal{L}_4$
50	**32.32**$\pm$0.53	11.22$\pm$2.32	31.11$\pm$1.09	24.69$\pm$0.34	31.63$\pm$0.98
200	**40.59**$\pm$0.73	22.26$\pm$0.46	38.69$\pm$0.29	24.55$\pm$0.83	38.92$\pm$0.48

Although task recency bias has been extensively studied in CIL (Wu et al., 2019; Masana et al., 2022; Hou et al., 2019; Arani et al., 2022) and is straightforward to analyze as classes are distinct between different tasks, it has not been widely studied in the DIL scenario due to its inherent aspect where all tasks share the same set of classes. As a step forward in studying this bias in DIL, we analyze the logit norms of different CL approaches. DNNs that predict logits with a larger magnitude or norm directly translate into overconfident predictions (Chrysakis & Moens, 2023), which in turn can indicate the bias of a model toward samples belonging to a certain task. Figure 5 (left) illustrates the logit norms predicted by different CL models on the first and last domain samples. While ER and DER++ are more confident on the last task samples compared to the samples belonging to the first task, DARE achieves more uniformly distributed confidence over old and new task samples. Additionally, Figure 5 (right) illustrates the calibration error (Guo et al., 2017) of the model on the initial, last, and all task samples. It is further evident that training with DARE achieves a lower calibration error compared to other CL methods.

6.4. Effectiveness of Individual Components

The effectiveness of our method is demonstrated through an ablation study (Table 2), where three interconnected loss functions are crucial. Removing any loss function results in performance decline, especially after removing $\mathcal{L}_1$ and $\mathcal{L}_3$, impacting accuracies for both current and previous tasks. In particular, $\mathcal{L}_2$ and $\mathcal{L}_4$ contribute to gains of 3.89% and 2.18% for buffer size 50, maximizing Divergence and minimizing Adaptation steps. The synergy among the three stages—Divergence, Adaptation, and Refinement—optimizes the representation space for new tasks, ensuring optimal learning. Their interdependence is crucial; isolated analysis risks divergence, and the absence of $\mathcal{L}_3$

during Divergence leads to catastrophic forgetting. This approach also minimizes representation drift (see Figure 4).

7. Conclusion

We proposed a novel method to address representation drift in domain-incremental learning. Our proposed method, DARE, mitigates representation drift at task boundaries and effectively assimilates new domain information into the feature space of old task samples. The inclusion of an effective buffer sampling strategy allows the preservation of the dark knowledge learned on old tasks when learning new ones. Our empirical evaluation demonstrated that DARE outperforms existing methods across different DIL benchmarks, with less forgetting and improved performance on seen domains. Furthermore, DARE exhibits efficient memory and computational usage, reduces bias towards recent task samples, and inhibits abrupt representation drift at task boundaries. These results demonstrate DARE's efficacy and the potential for practical applications in continual learning.

Limitations and Future Work

One particular area of focus for enhancement that we endeavor to tackle pertains to the enhancement of our methodology to lessen the reliance on task-id, which is presently vital for the IRS buffer sampling strategy. Nevertheless, we can integrate mechanisms for independently identifying task transitions, such as monitoring variations in loss metrics.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Arani, E., Sarfraz, F., and Zonooz, B. Learning fast, learning slow: A general continual learning method based on complementary learning system. *arXiv preprint arXiv:2201.12604*, 2022.

Benjamin, A. S., Rolnick, D., and Kording, K. Measuring and regularizing networks in function space. *arXiv preprint arXiv:1805.08289*, 2018.

Buzzega, P., Boschini, M., Porrello, A., Abati, D., and Calderara, S. Dark experience for general continual learning: a strong, simple baseline. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020.

Buzzega, P., Boschini, M., Porrello, A., and Calderara, S. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2180–2187. IEEE, 2021.

Caccia, L., Aljundi, R., Asadi, N., Tuytelaars, T., Pineau, J., and Belilovsky, E. New insights on reducing abrupt representation change in online continual learning. *arXiv preprint arXiv:2203.03798*, 2022.

Cha, H., Lee, J., and Shin, J. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pp. 9516–9525, 2021.

Chaudhry, A., Dokania, P. K., Ajanthan, T., and Torr, P. H. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

Chrysakis, A. and Moens, M.-F. Online bias correction for task-free continual learning. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=18XzeuYZh_.

Douillard, A., Ramé, A., Couairon, G., and Cord, M. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295, 2022.

Farquhar, S. and Gal, Y. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

Garg, P., Saluja, R., Balasubramanian, V. N., Arora, C., Subramanian, A., and Jawahar, C. Multi-domain incremental learning for semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 761–771, 2022.

Gowda, S., Zonooz, B., and Arani, E. Dual cognitive architecture: Incorporating biases and multi-memory systems for lifelong learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=PEyVq0hlO3.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hinton, G., Vinyals, O., and Dean, J. Dark knowledge. *Presented as the keynote in BayLearn*, 2(2), 2014.

Hou, S., Pan, X., Loy, C. C., Wang, Z., and Lin, D. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019.

Hu, G., Zhang, W., Ding, H., and Zhu, W. Gradient episodic memory with a soft constraint for continual learning, 2020. URL https://arxiv.org/abs/2011.07801.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

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

Lv, F., Liang, J., Li, S., Zang, B., Liu, C. H., Wang, Z., and Liu, D. Causality inspired representation learning for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8046–8056, 2022.

Masana, M., Liu, X., Twardowski, B., Menta, M., Bagdanov, A. D., and van de Weijer, J. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Michieli, U. and Zanuttigh, P. Knowledge distillation for incremental learning in semantic segmentation. *CoRR*, abs/1911.03462, 2019. URL http://arxiv.org/abs/1911.03462.

Mirza, M. J., Masana, M., Possegger, H., and Bischof, H. An efficient domain-incremental learning approach to drive in all weather conditions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3001–3011, 2022.

Murata, K., Toyota, T., and Ohara, K. What is happening inside a continual learning model? a representation-based evaluation of representational forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 234–235, 2020.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., and Wang, B. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1406–1415, 2019.

Ratcliff, R. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., and Tesauro, G. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

Saito, K., Watanabe, K., Ushiku, Y., and Harada, T. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3723–3732, 2018.

Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y. W., Pascanu, R., and Hadsell, R. Progress & compress: A scalable framework for continual learning. In *International conference on machine learning*, pp. 4528–4537. PMLR, 2018.

Tan, C., Gao, Z., Wu, L., Li, S., and Li, S. Z. Hyperspherical consistency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7244–7255, 2022.

Van de Ven, G. M. and Tolias, A. S. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*, 2019.

Vitter, J. S. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1): 37–57, 1985.

Wang, C., Yang, Q., Huang, R., Song, S., and Huang, G. Efficient knowledge distillation from model checkpoints. *arXiv preprint arXiv:2210.06458*, 2022.

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., and Fu, Y. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 374–382, 2019.

Xie, J., Yan, S., and He, X. General incremental learning with domain-aware categorical representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14351–14360, 2022.

Yang, Y., Kim, T., and Wang, G. Multiple classifiers based maximum classifier discrepancy for unsupervised domain adaptation. *arXiv preprint arXiv:2108.00610*, 2021.

Yu, L., Twardowski, B., Liu, X., Herranz, L., Wang, K., Cheng, Y., Jui, S., and Weijer, J. v. d. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6982–6991, 2020.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995. PMLR, 2017.

Zhang, X., Dou, D., and Wu, J. Feature forgetting in continual representation learning. *arXiv preprint arXiv:2205.13359*, 2022.

A. Appendix

A.1. Evaluation Metrics

To evaluate the performance of different models under different settings, we selected two main metrics widely used in the CL literature. We formalize each metric below.

1. **Last Accuracy** defines the final performance of the CL model on the validation set of all the tasks seen so far. Concretely, given that tasks are sampled from a set $t \in 1, 2..., T$, where T is the total number of tasks and $a_{k,j}$ is the accuracy of a CL model on the validation set of the task k after learning task j, last accuracy A_{last} is as follows:

$$A_{last} = \frac{1}{T} \sum_{k=1}^{T} a_{k,T} \tag{7}$$

2. **Backward Transfer (BWT)** defines the influence of the learning task t on previously seen tasks $k < t$. Positive BWT implies that the learning task t increased performance on previous tasks, while negative BWT indicates that the learning task t affected the performance of the model on previous tasks. Formally, BWT is as follows:

$$BWT = \frac{1}{T-1} \sum_{j=1}^{T-1} a_{T,j} - a_{j,j} \tag{8}$$

A.2. Datasets

DN4IL (Gowda et al., 2023) is a subset of the standard DomainNet dataset (Peng et al., 2019) proposed for large-scale unsupervised domain adaptation composed of 345 categories. DN4IL is a class-balanced dataset with 20 supercategories and five classes under each supercategory. In total, the dataset consists of 100 categories spanning over 6 different domains namely, 'sketch', 'real', 'quickdraw', 'painting', 'infograph', and 'clipart' with approximately 67k training images and 19k test images of shape 64×64. Domain-incremental learning (DIL) scenario on DN4IL is more challenging compared to other datasets conventionally used for DIL as the distribution shift is more prominent. Figure 6 shows some examples of different domains in the dataset.

iCIFAR-20 (Xie et al., 2022) is the DIL version of CIFAR-100 (Krizhevsky et al., 2009) dataset. iCIFAR-20 is a class-balanced dataset with 20 supercategories and five classes under each supercategory. The 20 supercategories are considered as real labels, and the five subcategories under each label is considered as a new domain. The dataset consists of approximately 50k training images and 10k test images of size 32×32.

A.3. IRS - Intermediary Reservoir Sampling

Motivated by its effectiveness in Reinforcement Learning (Rolnick et al., 2019), rehearsal-based methods in continual learning settings store a subset of input samples/exemplars and their corresponding labels in the replay buffer and interleave them while learning new tasks. Ideally, the replay buffer is expected to model the data distribution of all previous tasks, and the training algorithm samples exemplars from the buffer and interleaves them with the current task samples while learning a new task, thus mitigating forgetting the knowledge of old tasks. Rehearsal-based methods are widely used in CL and different approaches have been proposed to populate the buffer (Rebuffi et al., 2017; Chaudhry et al., 2018).

Dark Experience Replay (DER) (Buzzega et al., 2020) proposes to store logits along with exemplars and to learn the model on new tasks while emulating their earlier responses to old task samples. Analogous to logit replay, many works have tried distilling logits from a teacher model, typically a snapshot of the model at task boundaries (Douillard et al., 2022; Li & Hoiem, 2017; Michieli & Zanuttigh, 2019) or exponential moving average of the model (Arani et al., 2022) to mitigate forgetting. Concretely, the regularization loss on the logits distills the 'dark knowledge' (Hinton et al., 2014) learned by the model in the previous tasks into the weights of the model being trained. This dark knowledge constitutes more information about the relationships among different classes of input, thus guiding the learning model better discriminate among samples belonging to different tasks and different classes as well. Thus, the information contained in the logits contributes significantly to the accuracy of the learning model on all the seen tasks.

We propose the "*Intermediary Reservoir Sampling (IRS)*" strategy, which employs a normal distribution on the learning trajectory of each task. The mean of the distribution is set to the intermediate stages, and the buffer is populated accordingly.

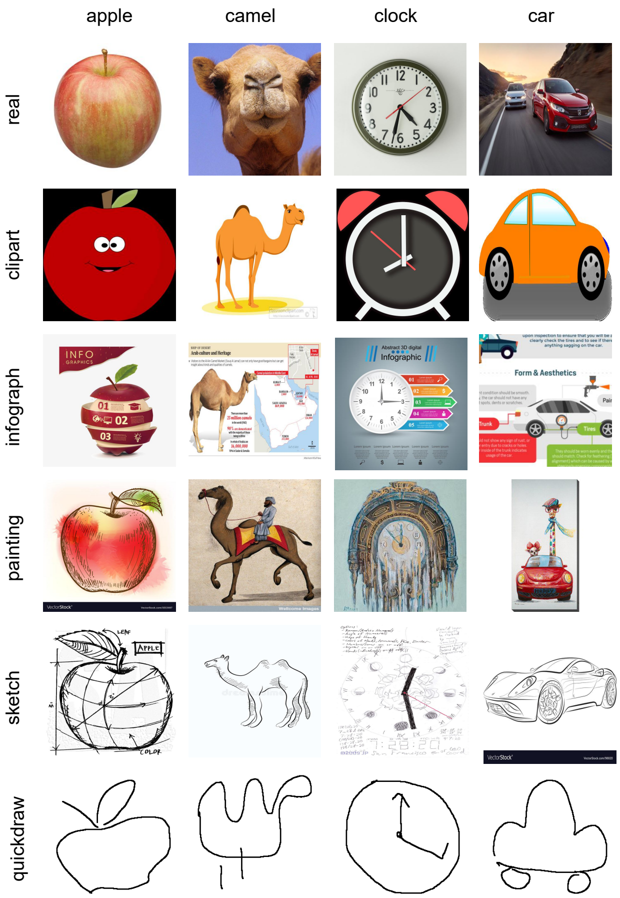

Figure 6. Visualization of domain shifts in the DN4IL dataset.

This incentivizes the storage of logits with more ”dark knowledge” about the current task, which in turn propagates the knowledge across future tasks through distillation. Algorithm 2 describes the IRS strategy.

Furthermore, we probe the improvements brought about by the proposed approach over DARE and DER++ in Table 3. It is evident that the proposed IRS strategy improves both DER++ and DARE in the most challenging learning setting with a buffer size of 50. IRS improves DER++ by ∼5% and DARE by ∼12%.

A.4. Task-wise Performance for Multi-Memory Methods

We analyzed the task-wise accuracies of the single-model versions in the main text, and here we compare DARE++ with other multi-memory methods like CLS-ER and DUCA. Figure 7 shows the task-wise accuracy of different CL approaches while learning new tasks. It can be seen that DARE++ retains accuracy on the initial tasks much better than CLS-ER and

Table 3. Comparison of the reservoir sampling and the proposed IRS buffer sampling strategy. Results are on DN4IL dataset trained with buffer size 50 for six tasks.

Metric	Method	Reservoir Sampling	IRS
BWT	DER++	-23.99$\pm$0.74	-22.69$\pm$3.71
	DARE	-15.77$\pm$0.69	**-14.69**$\pm$0.19
Last Accuracy	DER++	35.74$\pm$0.67	37.60$\pm$1.21
	DARE	36.17$\pm$0.38	**40.59**$\pm$0.73

Algorithm 2 Intermediary Reservoir Sampling (IRS)

```
input: Data streams $\mathcal{D}_t \forall \{t = 1, ..., T\}$, model $f_\theta$, memory buffer $\mathcal{M}$ , number of seen examples $N$, input sample $x$, label $y$, logit $z$.
for all tasks $t \in \{1, 2, .., T\}$ do
    for epochs $ep \in \{1, 2, .., E\}$ do
        for minibatch $\mathcal{B} \rightarrow (x_t, y_t) \in \mathcal{D}_t$ of size $|\mathcal{B}|$ do
            $z_t = f_\theta(x_t)$
            if uniform $[0, 1] < \frac{1}{\sigma\sqrt{2\pi}} e^{-\left(ep-\frac{E}{2}\right)^2/2\sigma^2}$ then
                if $|\mathcal{M}| < N$ then
                    $\mathcal{M}[N] \leftarrow (x_t, y_t, z_t)$
                else
                    $i \sim [0, N]$                                    ▷ Random index
                    if $i < |\mathcal{M}|$ then
                        $\mathcal{M}[i] \leftarrow (x_t, y_t, z_t)$
                    end if
                end if
            end if$N$ += $|\mathcal{B}|$                 ▷ Increment number of seen samples with minibatch size
        end for
    end for
end for
return: updated memory buffer $\mathcal{M}$
```

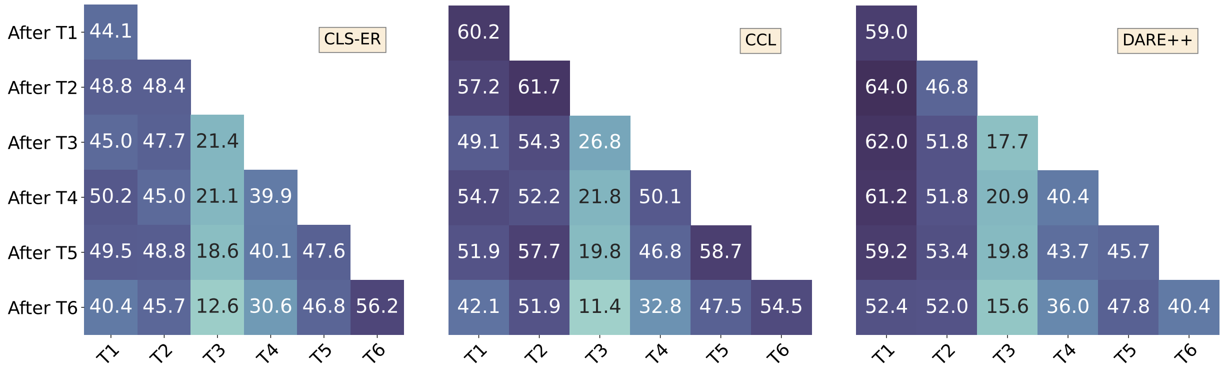

Figure 7. Task-wise accuracy of different multi-memory CL models while learning new tasks with buffer size 50 on DN4IL. DARE++ performs on par with the Stable Model of CLS-ER and the Semantic Memory of DUCA, which require more training memory and computations.

DUCA. DARE++ effectively consolidates the knowledge about the past tasks from the working model compared to other memory-intensive approaches, and this can be attributed to DARE inhibiting the excessive representation of past tasks, and thus retaining the performance on them. It should be noted that DARE++ performs well despite requiring less memory and training computations, as evident from the number of parameters in Table 1. This makes it more efficient and effective in real-time applications.

A.5. Extended Results with Conventional CL Methods

In addition to the comparisons in the main text, we compare DARE with conventional regularization- and replay-based methods. Online EWC (Schwarz et al., 2018) and Synaptic Intelligence (Zenke et al., 2017) fall under regularization-based methods, where changes to important parameters in the network for old tasks are penalized. Averaged-Gradient Episodic Memory (A-GEM) (Hu et al., 2020) and Function Distance Regularization (FDR) (Benjamin et al., 2018) fall into replay-based methods. A-GEM learns new tasks with an optimization constraint such that the gradients for new tasks are projected to the orthogonal subspace of the gradients for old task samples, thus retaining the performance on old tasks. FDR applies distillation loss with respect to network outputs stored in the buffer for past task samples.

Table 4 compares DARE with various baselines on different datasets and buffer sizes. Regularization-based methods (oEWC and SI) face challenges in the DIL scenario, while A-GEM and FDR demonstrate performance comparable to ER. It is

Table 4. Results on DIL benchmarks with varying buffer sizes averaged over three runs. DARE achieves a consistent improvement over the other methods across different metrics, i.e., accuracy and BWT. Accuracy determines the performance on all tasks learned by the model, and backward transfer quantifies the degree to which learning a new task improves performance on previously learned tasks.

Buffer Size	Method	iCIFAR-20			DN4IL		
		#P ↓	BWT ↑	Last Accuracy ↑	#P ↓	BWT ↑	Last Accuracy ↑
-	Joint	11.18	-	79.61±0.13	11.22	-	59.93±1.07
	SGD	11.18	-43.72±1.07	49.40±0.53	11.22	-42.42±0.00	21.63±0.42
	oEWC	11.18	-41.35±2.01	47.39±2.00	11.22	-38.42±0.57	19.56±1.05
	SI	11.18	-41.44±2.75	45.94±2.48	11.22	-25.20±2.75	21.67±1.47
50	ER	11.18	-42.03±0.27	50.23±0.94	11.22	-36.11±0.26	24.24±0.34
	A-GEM	11.18	-43.02±0.88	50.02±0.14	11.22	-35.38±0.35	27.06±0.35
	FDR	11.18	-42.05±1.57	51.07±0.58	11.22	-38.48±1.02	25.09±0.66
	DER++	11.18	-40.63±0.49	52.68±1.10	11.22	-29.05±1.35	28.08±0.99
	DARE	11.19	**-35.64**±0.00	**53.18**±1.00	11.27	**-22.98**±0.62	**32.32**±0.53
100	ER	11.18	-41.88±0.59	50.85±0.73	11.22	-35.28±1.20	24.67±0.86
	A-GEM	11.18	-42.98±0.80	50.43±0.57	11.22	-35.78±0.08	27.15±0.33
	FDR	11.18	-41.22±0.58	52.37±0.39	11.22	-37.26±0.56	26.08±0.65
	DER++	11.18	-37.33±1.47	55.32±0.69	11.22	-27.78±0.90	32.06±1.05
	DARE	11.19	**-33.20**±0.09	**56.01**±0.22	11.27	**-19.37**±0.43	**37.16**±0.62
200	ER	11.18	-38.98±0.74	52.57±0.79	11.22	-32.35±0.51	27.45±0.94
	A-GEM	11.18	-41.49±0.75	51.12±0.76	11.22	-35.65±0.05	27.44±0.39
	FDR	11.18	-38.82±0.85	54.06±0.61	11.22	-36.26±0.55	27.21±0.53
	DER++	11.18	-33.61±0.64	58.39±0.38	11.22	-23.99±0.74	35.74±0.67
	DARE	11.19	**-30.22**±1.84	**58.53**±1.25	11.27	**-14.69**±0.19	**40.59**±0.73

Table 5. Selected hyperparameters for DARE and DARE++.

Dataset	Method	Buffer Size	lr	α	β	sw	st	r	α
iCIFAR-20	DARE	50	0.04	0.3	0.1	0.05	1.2	-	-
		100	0.03	0.5	0.1	0.08	1.0	-	-
		200	0.06	0.5	0.1	0.08	0.99	-	-
	DARE++	50	0.04	0.3	0.1	0.05	1.2	0.055	0.999
		100	0.03	0.5	0.1	0.08	1.1	0.058	0.999
		200	0.06	0.5	0.2	0.08	1.1	0.045	0.999
DN4IL	DARE	50	0.04	0.1	0.2	0.05	0.8	-	-
		100	0.04	0.1	1.0	0.05	0.8	-	-
		200	0.04	0.1	1.0	0.05	0.8	-	-
	DARE++	50	0.04	0.1	0.2	0.05	0.8	0.050	0.999
		100	0.04	0.1	1.0	0.05	0.8	0.050	0.999
		200	0.04	0.1	1.0	0.05	0.8	0.090	0.999

evident that DARE surpasses all other baselines in all settings.

A.6. Hyperparameters

We enumerate the best hyperparameters chosen for the evaluation of different methods in the main paper and Table 4. lr denotes the learning rate for the entire learning trajectory in each task. We fixed the batch size to 32 for both the current task and old task samples (in buffer memory). We used grid search to find the best hyperparameters and took reference from *mammoth* (Buzzega et al., 2020) repository for the search for experiments on iCIFAR-20 dataset and DUCA (Gowda et al., 2023) for the search for experiments on DN4IL dataset, respectively. We trained all methods using SGD optimizer for 50 epochs per task.

Table 5 outlines the hyperparameters chosen for DARE and DARE++, while Table 6 lists the hyperparameters selected for various CL baselines in our study. r denotes the frequency of updating the semantic model from the working model, and d denotes the rate at which the weights of the EMA model are updated. r_p and r_s stand for the update frequency for the plastic and stable model, respectively, in CLS-ER. λ refers to the weighting parameter for the knowledge distillation from the semantic model to the working model in DUCA and CLS-ER. Furthermore, sw and st denote the weight and temperature used in the supervised contrastive loss.

The losses corresponding to *Divergence* and *Adaptation* steps in DARE/DARE++ were weighted by 0.1 and 1, respectively,

Table 6. Selected hyperparameters for all baselines.

Dataset	Buffer Size	Method	Hyperparameters
iCIFAR-20	-	oEWC	lr=0.03, λ=10, γ=1
		SI	lr=0.03, c=1, ξ=0.9
	50	ER	lr=0.1
		A-GEM	lr=0.05
		FDR	lr=0.03, α=0.1
		DER++	lr=0.03, α=0.1, β=0.2
		CLS-ER	lr=0.05, λ=0.1, r_p=0.06, r_s=0.02, d_p=0.999, d_s=0.999
		DUCA	lr=0.05, λ=0.1, r=0.08, d=0.999
	100	ER	lr=0.1
		A-GEM	lr=0.06
		FDR	lr=0.03, α=0.2
		DER++	lr=0.05, α=0.1, β=0.1
		CLS-ER	lr=0.03, λ=0.1, r_p=0.08, r_s=0.04, d_p=0.999, d_s=0.999
		DUCA	lr=0.04, λ=0.1, r=0.09, d=0.999
	200	ER	lr=0.1
		A-GEM	lr=0.04
		FDR	lr=0.03, α=0.5
		DER++	lr=0.03, α=0.2, β=0.1
		CLS-ER	lr=0.05, λ=0.1, r_p=0.12, r_s=0.04, d_p=0.999, d_s=0.999
		DUCA	lr=0.04, λ=0.1, r=0.08, d=0.999
DN4IL	-	oEWC	lr=0.05, λ=50, γ=1
		SI	lr=0.05, c=0.5, ξ=0.5
	50	ER	lr=0.1
		A-GEM	lr=0.05
		FDR	lr=0.03, α=0.5
		DER++	lr=0.01, α=0.1, β=0.1
		CLS-ER	lr=0.05, λ=0.1, r_p=0.06, r_s=0.04, d_p=0.999, d_s=0.999
		DUCA	lr=0.04, λ=0.1, r=0.06, d=0.999
	100	ER	lr=0.1
		A-GEM	lr=0.04
		FDR	lr=0.05, α=0.5
		DER++	lr=0.03, α=0.2, β=0.5
		CLS-ER	lr=0.05, λ=0.1, r_p=0.14, r_s=0.04, d_p=0.999, d_s=0.999
		DUCA	lr=0.05, λ=0.1, r=0.06, d=0.999
	200	ER	lr=0.1
		A-GEM	lr=0.04
		FDR	lr=0.05, α=0.1
		DER++	lr=0.03, α=0.1, β=1.0
		CLS-ER	lr=0.05, λ=0.1, r_p=0.08, r_s=0.04, d_p=0.999, d_s=0.999
		DUCA	lr=0.03, λ=0.1, r=0.06, d=0.999

in all datasets and buffer sizes. It is also evident from Table 5 that DARE and DARE++ do not need extensive finetuning, except for the dataset-specific learning rates. The other hyperparameters are mostly stable across different settings.