NICE: Neurogenesis Inspired Contextual Encoding for Replay-free Class Incremental Learning

Mustafa Burak Gurbuz
Georgia Institute of Technology, USA
mgurbuz6@gatech.edu

Jean Michael Moorman
Georgia Institute of Technology, USA
jmoorman9@gatech.edu

Constantine Dovrolis
The Cyprus Institute, Cyprus
Georgia Institute of Technology, USA
constantine@gatech.edu

Abstract

Deep neural networks (DNNs) struggle to learn in dynamic settings because they mainly rely on static datasets. Continual learning (CL) aims to overcome this limitation by enabling DNNs to incrementally accumulate knowledge. A widely adopted scenario in CL is class-incremental learning (CIL), where DNNs are required to sequentially learn more classes. Among the various strategies in CL, replay methods, which revisit previous classes, stand out as the only effective ones in CIL. Other strategies, such as architectural modifications to segregate information across weights and protect them from change, are ineffective in CIL. This is because they need additional information during testing to select the correct network parts to use. In this paper, we propose NICE, Neurogenesis Inspired Contextual Encoding, a replay-free architectural method inspired by adult neurogenesis in the hippocampus. NICE groups neurons in the DNN based on different maturation stages and infers which neurons to use during testing without any additional signal. Through extensive experiments across 6 datasets and 3 architectures, we show that NICE performs on par with or often outperforms replay methods. We also make the case that neurons exhibit highly distinctive activation patterns for the classes in which they specialize, enabling us to determine when they should be used. The code is available at https://github.com/BurakGurbuz97/NICE.

1. Introduction

Deep neural networks (DNNs) are typically trained on static data distributions, where training batches are drawn from a fixed dataset adhering to the independent and identically distributed (i.i.d.) assumption. While this approach has advanced the field, it overlooks a key facet of intelligence: the ability to continuously learn from non-stationary distributions over time. In such continual learning (CL) scenarios, when DNNs process new data, they tend to suffer from catastrophic forgetting (CF) of prior knowledge [25, 44].

Recent interest in CL has led to various scenarios [21, 48, 71]. Widely adopted are class-incremental learning (CIL) and task-incremental learning (TIL) [67, 68]. In both, learners process series of datasets and classify test samples into previously seen classes. The difference between them lies in the handling of test samples. In TIL, a task identifier indicates the originating dataset of the test sample. In contrast, CIL lacks this information, requiring learners to also distinguish classes appearing at different timepoints.

This subtle distinction leads to significant variations in performance. Many TIL approaches either fail or show substantial drops in performance when adapted to CIL. Among the various strategies to address CL, replay stand out as the only effective ones in the CIL setting [23, 31, 36, 67]. In their basic form, replay methods retain a subset of samples from all previously seen classes [19]. When introduced to a new class, they augment the training batches with samples from earlier classes and train on all classes simultaneously.

Despite the success of replay methods, they have faced some criticism. For example, [50] proposed the GDumb algorithm, which keeps a subset of samples and retrains a model from scratch on these samples at test time, as a critique of prevailing replay methods. Surprisingly, this straightforward approach often outperforms well-known replay methods, suggesting that the performance of these methods may largely depend on the samples retained in memory. Another work [16] indicates that in the Experience Replay (ER) algorithm [19], which has inspired numerous replay methods, representations of older classes are rapidly disrupted when training on new classes. Consequently, the

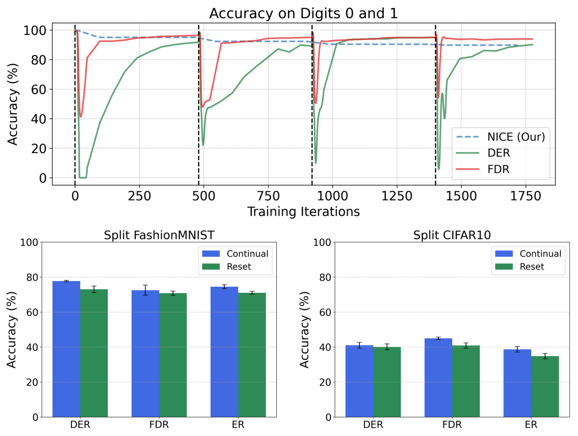

Figure 1. Top: DER & FDR trained on MNIST, with vertical dashed lines indicating the introduction of two novel classes. Bottom: Accuracy across all classes after sequentially learning FashionMNIST and CIFAR10. Results are averaged across 3 seeds.

model must relearn representations for these older classes.

Following these critiques, we evaluated some common replay methods and observed similar issues. For instance, Figure 1 shows the performance of the Dark Experience Replay (DER) [14] and Function Distance Regularization (FDR) [9] methods on the first two classes of MNIST as we introduce more classes. Both methods exhibit a drastic drop in accuracy for the first two classes after few updates with new data, despite using replay. After this drop, they recover performance on previous classes with the aid of replay. Supplementary Material Section-D.1 presents similar observations on larger datasets. This drop in performance does not necessarily mean complete forgetting, as knowledge may be kept in earlier layers. To investigate this, we reset the weights of the replay methods each time new classes were introduced. As shown in Figure 1, they experienced a mild drop in accuracy due to the weight reset. This mild drop indicates that information is not entirely forgotten. However, these results also suggest that the accuracy of replay methods fluctuates erratically with the new data.

We argue that this issue arises because the architecture does not recognize the sequential nature of learning. Replay primarily focuses on inputs and outputs. With this approach, the network distributes information across all weights, optimizing for i.i.d. input. However, in CL, input is not i.i.d., and new classes can cause rapid disruption in representations, leading to performance fluctuations and abrupt forgetting. For instance, when new data is introduced, neurons representing old classes are likely to make overconfident, incorrect predictions because standard DNNs suffer from a calibration problem – that is, a model's predicted class probabilities do not accurately estimate the true likelihood of correctness [26, 27]. Thus, output neurons of old classes receive significant penalties, and class representations rapidly change, especially in the top layers.

Consequently, we are motivated to develop an architecture that recognizes the sequential aspect of CL. Our intention is not to discredit replay. Biology has taught us that the replay of neural patterns is essential in memory formation, retrieval, and consolidation in the brain. It stands as a valuable model, inspired by nature, for achieving CL desiderata [29, 30, 38, 66]. However, nature also offers architectural insights to preserve knowledge while continuing to learn progressively. Hence, this paper seeks to identify neuro-inspired architectural strategies to address CIL. We do not perform any replay and focus on the architecture alone. In the long run, architectures with CIL capabilities can serve as a backbone for hybrid systems that use other CL strategies, including replay – briefly discussed in Section 5.

Note that while architectural strategies are common in CL, their effectiveness is largely limited to TIL [31, 36]. The few methods developed for CIL still tend to significantly underperform in comparison to even simple replay [62, 63]. We propose an architectural approach that not only rivals but often outperforms replay methods in CIL.

Our approach, Neurogenesis Inspired Contextual Encoding (NICE), is inspired by adult neurogenesis observed in the hippocampus. This process continuously introduces adult-born granule cells into the dentate gyrus, providing a unique form of plasticity not found in most other brain regions [2, 3, 46]. As these newly introduced cells mature, they help to form disentangled representations for experiences and play a crucial role in contextual discrimination [20, 47, 65, 74]. This process facilitates the efficient integration of new information without causing forgetting. In DNNs, we emulate this by introducing neurons at different maturation stages. Furthermore, we have devised a context-detector that identifies which neurons are most predictive for a sample. During testing, this detector predicts the context and sends inhibitory signals to certain neurons that would otherwise cause interference. A comprehensive discussion of the neuroscientific underpinnings of NICE is presented in the Supplementary Material Section-E.

2. Related Work

More detailed overviews can be found in [21, 48, 71].

2.1. Replay Methods for CL

Raw Replay: The most common form of replay is revisiting old examples [6, 11, 12, 14, 16, 19, 41, 50, 52, 60, 76]. They employ an extra memory and fill it by sampling from seen examples. The batches are formed by mixing examples from new dataset with memory examples. Although it is an effective strategy, raw replay may not be feasible in settings where the indefinite retention of previous data is prohibited due to regulatory restrictions or compute limitations.

Generative Replay: Some studies have suggested using a generative model to produce samples that resemble seen examples [7, 8, 15, 35, 51, 54, 61]. They address data retention concerns, but with high computational costs. Training generative models is challenging since they also forget [66].

Activation Replay: Some works suggested replaying activations [29, 49, 66, 70]. They pass examples through a frozen feature extractor, storing the activations output by the feature extractor rather than the examples. Thus, they can store more within the same memory budget, and they do not access old data. However, they depend on pretraining and freezing. Without these steps, stored activations become outdated once the feature extractors are updated.

Lastly, replay methods revisit all classes to create batches, which introduces significant computational overhead. Furthermore, they focus on engineering the input rather than addressing forgetting within the architecture.

2.2. Replay-free Approaches in CL

Regularization Methods: Regularization methods modulate updates to protect important weights and preserve past knowledge [5, 17, 37, 40, 55, 58, 59, 78]. However, they primarily work in TIL and perform poorly in CIL settings [31, 36, 67], as their training does not enforce discrimination across classes that appear at different time points.

Architectural Methods: Some work modifies the DNN structure to perform CL. They allocate distinct parameters for tasks (e.g., set of classes) to segregate the knowledge of different classes. This segregation is often achieved by expanding new branches/modules [4, 45, 56, 57, 69, 75] or partitioning existing connections through weight pruning [1, 28, 33, 42, 62, 63]. Additionally, another approach entails learning unique masks for each task to select a subset of weights for a given task [34, 43, 72, 73]. A common issue with these architectural methods is their heavy reliance on task identifiers to determine which branches, modules, or masks to use for predictions, rendering them effective only in TIL settings. As far as we are aware, only SpaceNet [62] and AFAF [63] demonstrate good performance in CIL scenarios among architectural methods. However, they significantly lag behind replay methods. Our approach, NICE, is an architectural method; however, it does not rely on task-identifiers and often outperforms replay methods.

3. Neurogenesis Inspired Contextual Encoding

3.1. Problem Formulation and Notation

We focus on CIL through a series of E episodes where each episode e has a training dataset D_e. Although the term *task* is commonly used in the literature, we avoid using it due to its various meanings in different contexts. A neural network f with L layers is trained sequentially on one dataset per episode. At test time, f is given a new example x and must predict its label from the set of classes observed so far.

In NICE, neurons in f are denoted as N and are assigned ages. These ages start at 0 and can increase up to the number of episodes observed, reflecting the neurons' maturation levels. We represent the neuron i in layer l as $n_i^l \in N^l$. Additionally, we use subscripts to denote neurons with certain ages; for example, $N_{=1}^l$ represents all neurons in layer l that are age 1. Furthermore, $A(n_i^l, x, \alpha)$ denotes the activation of the neuron n_i^l for the sample x, considering only incoming activations from previous layer neurons of age α.

3.2. Neuron Ages

Neurogenesis (from age 0 to 1): NICE draws inspiration from biological neurogenesis, which continually integrates new neurons into neural circuits. However, the growing architectures introduce complexity and computational overhead. Thus, we have adapted the neurogenesis idea to a fixed architecture. In this model, neurons are initially assigned an age of 0, indicating that they are in surplus capacity (i.e., neurons to be added to the architecture). As network learns, we periodically transition neurons between age-0 and age-1 to determine which neurons should be permanently integrated into the network. When learning begins with the new dataset D_e, all neurons at age 0 are temporarily set to age 1. Subsequently, every p epochs, we revert some neurons back to age 0 based on their activations. Here, p is NICE's only additional hyperparameter. To revert neurons back to age-0, first, we calculate the activations (after ReLU) of the neurons at age 1 in each layer, using a subset of the episode examples (1024 for experiments):

$$A_{=1}^l = \sum_{x_k \sim D_e} \sum_{n_i^l \in N_{=1}^l} A(n_i^l, x_k, 1) \tag{1}$$

Here, $A_{=1}^l$ the total activation for layer l, considering only neurons at age 1. Next, we select age-1 neurons to keep at layer l denoted by set S_1^l by solving the following discrete constrained optimization problem:

$$\min_{S_1^l \subseteq N_{=1}^l} |S_1^l| \text{ subject to } \sum_{x_k \sim D_e} \sum_{n_i^l \in S_1^l} A(n_i^l, x_k, 1) \geq \tau A_{=1}^l \tag{2}$$

Simply put, we aim to keep the smallest subset of age-1 neurons that, in combination, contribute to at least a τ fraction (set at 0.95 in our experiments) of the layer's total activation. The problem can be solved optimally using a greedy algorithm. First, sort the neurons based on their activations in descending order. Then, select neurons with the highest activation until the target is achieved or exceeded (see Supplementary Material Section-C for the proof).

We assume that the activation of a neuron is a reliable indicator of its importance in learning. This assumption is supported by previous research in CL [24, 28, 33], network pruning [32, 39, 53], and model interpretation [22, 77]. Nevertheless, it is possible to substitute this score of importance with more complex scores. Note that the τ value

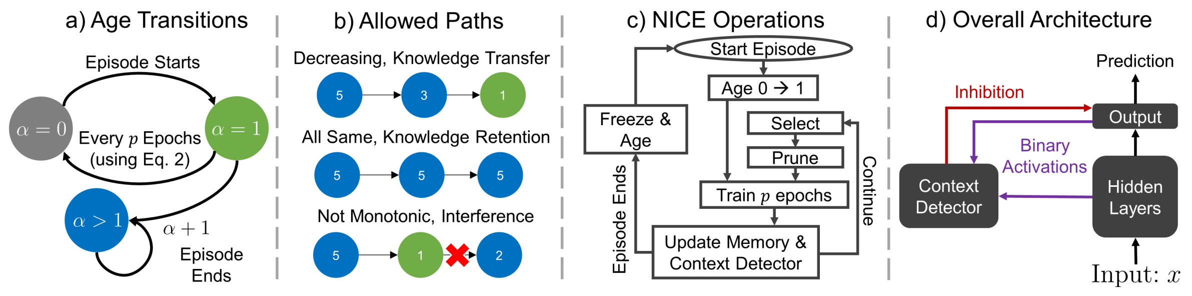

Figure 2. a) Transition diagram of neuron ages. b) Paths within the network and their roles. c) Diagram of NICE operations. d) Overview structure of the NICE. Gray neurons are reserved for the future, green neurons are receiving gradient updates, and blue neurons are frozen.

that is set to 0.95 may require adjustment for different importance scores. Excessively high values of τ could result in the selection of too many neurons, thereby diminishing the capacity available for future tasks. Conversely, setting τ too low could lead to aggressive elimination, which might not allocate sufficient capacity to learn current classes.

Maturation (from age 1 to beyond): At the end of each episode, a subset of neurons maintains age-1, while the remainder revert to $N_{=0}$ because of the iterative elimination. We then age all neurons above age-0 by one, allowing those that survived the elimination to mature to age-2. These neurons will continue to age incrementally after each episode, as shown in **Figure** 2-a. As neurons age, they lose plasticity and connections, which we will explain next. By design, input neurons always have the maximum age (i.e., E), and output neurons are aged to 1 when their class is first seen.

3.3. Avoiding Interference Between Neurons

We group neurons by their age, which indicates the episode during which they were introduced. For example, in episode 3, age-1 neurons are newly selected, while age-2 and age-3 neurons were selected at the end of episodes 2 and 1, respectively. The formula is: at episode E, the neurons responsible for episode e are at the age of $E - e + 1$.

Now, we will avoid modifying neurons age-2 or older so that NICE does not forget earlier episodes. Learning modifies a neuron in two ways: (a) by directly changing the incoming connections and (b) by altering the connections of ancestors. To address (a), we freeze incoming connections to neurons older than 1. We tackle (b) by pruning connections from age-u to age-v neurons where $u < v$, applying this every p epochs. So, neurons cannot have younger ancestors that will receive updates. This does not completely isolate neurons of different ages; we keep connections from older to younger ones to allow knowledge transfer between episodes. The combination of these two operations ensures neurons age-2 or older never change. Note that when training with cross-entropy loss, we only consider the activations of output neurons that correspond to batch classes prior to the softmax. This is a pivotal step as it avoids including frozen neurons whose classes are not in the batch. These frozen neurons, if included, cannot decrease in activation due to their state. This could potentially lead to an unintended escalation in the activity of other neurons as they try to compensate, which could cause training instability.

To summarize, age-0 neurons act as excess capacity, age-1 neurons are immature neurons that are plastic and learn, and neurons older than age 1 are mature neurons serving as memory. Another way to view this is in terms of paths from layer-1 to outputs. If we traverse along the paths, we can only have monotonically non-increasing paths in terms of ages. Here, decreasing paths are knowledge transfer paths that carry information from older episodes to newer ones. Paths that form only same age neurons are backbones for particular episode's knowledge, while increasing paths are not permitted because of pruning (see **Figure** 2-b).

In NICE, a subnetwork comprising neurons of a certain age and older is isolated from future updates by means of freezing and pruning, ensuring that these neurons remain unchanged (i.e., they neither forget nor learn). Consequently, this approach guarantees zero forgetting, provided that we are able to identify the ages associated with predicting the class of a test example. Let us assume we are in episode E, we encounter a test example that pertains to a class taught in episode e (although this is unknown to the model). Then the correct output neuron would be one from age $E - e + 1$. Therefore, our focus is on the subnetwork connected to these output neurons (comprising neurons $N_{\geq E-e+1}$). In the following, we explain how to infer which neuron ages should be utilized during prediction.

3.4. Context-Detector

NICE infers which ages to use during predictions through a context-detector, akin to the hippocampus where neurons at different maturation stages enable contextual discrimination. However, the underlying mechanisms involve complex recurrent relationships between hippocampal regions [20, 47, 65, 74]. We adopt a simplified perspective and use the network's global activation patterns to infer context.

Firstly, we periodically store information on neurons $N_{>0}$ that exhibit high activation for some recently seen examples. Specifically, every p epochs, we input m randomly

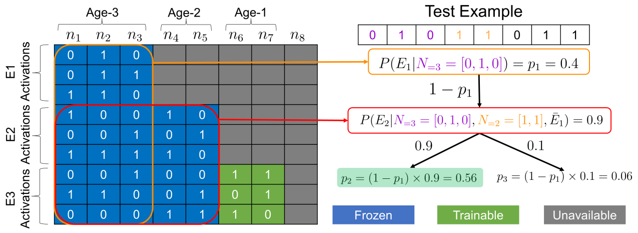
Age-3
Age-2
Age-1
n_1 n_2 n_3 n_4 n_5 n_6 n_7 n_8
E1 Activations
E2 Activations
E3 Activations
Test Example
0 1 0 1 1 0 1 1
$P(E_1|N_{=3}=[0,1,0]) = p_1 = 0.4$
$1-p_1$
$P(E_2|N_{=3}=[0,1,0], N_{=2}=[1,1], \bar{E}_1) = 0.9$
0.9
0.1
$p_2 = (1-p_1)\times 0.9 = 0.56$
$p_3 = (1-p_1)\times 0.1 = 0.06$
Frozen
Trainable
Unavailable

Figure 3. Left: Example memory for a network with 8 neurons and $m = 3$. Right: Illustration of chaining conditional probabilities.

selected examples into the f and get activations. Next, we apply threshold t_l on each layer activations, with t_l defined as the mean plus one standard deviation of the activations in layer l after episode-1 (for simplicity, we rely on statistics from the first episode). This yields a binary vector, representing the neurons that were notably active for the samples in the current episode. To illustrate this with a simplified example, imagine we have 8 neurons, and during episode 3, our neurons are grouped as follows: $N_{=3} = \{n_1, n_2, n_3\}$, $N_{=2} = \{n_4, n_5\}$, $N_{=1} = \{n_6, n_7\}$, and $N_{=0} = \{n_8\}$. Also, let's assume that $m = 3$. An example memory state for this case is shown on the left side of **Figure** 3. Note that if we already have memory allocated for current episode classes, incoming ones will replace the existing ones.

Next, we learn a set of conditional probabilities using this binary memory of activations. We model these conditional probabilities by fitting a logistic regression model. Suppose we are at episode E. First, we calculate the conditional probability that a sample belongs to episode 1 ($e = 1$) given observations $N_{=E}$, or $P(E_1|N_{=E})$, using memory (recall formula $E - e + 1$). For instance, in our example, the first three rows and columns act as positive samples, and the remaining six rows as negative samples. Then, we determine the probability for episode 2, given it is not episode 1, or $P(E_2|N_{=E}, N_{=E-1}, \bar{E}_1)$, using memory samples excluding episode 1. In our example, this involves using the first five columns, with rows 4-6 as positive and 7-9 as negative samples. We continue this process up to Episode $E-1$. The derived conditional probabilities are then chained, as depicted in **Figure** 3. More specifically, the probability of a sample belonging to episode e (except when $E = e$) is:

$$p_e = P(E_e|N_{\geq E-e+1}, \bar{E}_1, \ldots, \bar{E}_{e-1}) \tag{3}$$

$$\times \prod_{i=1}^{e-1} (1 - P(E_i|N_{\geq E-i+1}, \bar{E}_1, \ldots, \bar{E}_{i-1})) \tag{4}$$

Here, the second term represents the probability of the sample not belonging to earlier episodes, and the first term is the probability of the sample belonging to episode e given that it does not belong to any of the earlier episodes. If we are calculating p_1, the second term is omitted. If we are calculating p_E, the formula is simply $p_E = 1 - \sum_{i=1}^{E-1} p_i$.

Given a test sample, we input it to the network, generate a binary activation vector, and evaluate it using our chained probabilities. This results in a probability distribution across episodes, helping us identify the most probable episode and inhibiting irrelevant outputs. Note that the neurons used for learning these conditional probabilities are already frozen and isolated from the younger ones. Therefore, they remain constant despite training, ensuring that memory entries do not become outdated. While this method requires fitting logistic regressions, the cost is negligible compared to training DNN on older classes (i.e., replay).

3.5. Summary

Over the course of training, we select neurons (Section 3.2), update connections (Section 3.3), train for p epochs, and update the memory and context detector (Section 3.4). We repeat this cycle until an episode ends. Once an episode ends, we increment the ages and freeze certain neurons (Section 3.3). **Figure** 2-c shows a diagram of these operations. Note training always ends with memory and context detector update. The pseudo-code for the NICE is provided in the Supplementary Material Section-B. At test time, we input the test example x into the network, generate activations and then feed the thresholded binary vector into the context-detector to infer the episode (see **Figure** 2-d). Suppose the current episode is E, and our context indicates that the test sample x belongs to E_p; in that case, we mask out output neurons except for those of age-$(E-E_p+1)$. This let us use the subnetwork that is composed of neurons $N_{\geq E-E_p+1}$. For instance, the context-detector predicts that an input image is an animal, then the subnetwork deduces it is a cat.

4. Experimental Results

Our experiments cover six datasets: MNIST, FashionMNIST, EMNIST, CIFAR10, CIFAR100, and Tiny ImageNet. For the first three, we utilize a network comprising two convolutional layers with 32 filters each and a fully connected layer with 500 neurons. For CIFAR10 and CIFAR100, we employ a modified version of VGG11 to adapt it to smaller datasets. For Tiny ImageNet, we use ResNet18. **Table** 1 shows how we slice the datasets to create learning episodes. We keep the original class order, and episodes have the same number of non-overlapping classes. For more details on experiments, see the Supplementary Material Section-A.

In convolutional layers, 3D filters replace neurons, and 2D kernels replace connections. We define a convolutional neuron's activation as the average of the feature map it produces. Freezing a neuron entails freezing the corresponding dimensions in the batch normalization, which includes learnable parameters, running mean, and variance. Note that logistic regressions introduce extra parameters; however, pruning reduces NICE's parameters. So, the number of parameters in NICE is lower than those of a standard

Table 1. Episode details and memory budgets (denote with M).

	Datasets			
Details	MNIST/Fashion	EMNIST	CIFAR10/100	TinyImgNet
# Classes	10	26	10/100	200
# Episodes	5	13	5/10	5
# Neurons	575	591	3437/3527	4107
M (activations)	500	1300	1500/5000	5000
M (byte)	35.2 KB	91.2 KB	0.6/2.1 MB	2.5 MB
M (images)	46	120	210/718	210

dense network—see Supplementary Material Section-A.5. Furthermore, the training time for logistic regression is negligible compared to the replay, which effectively doubles training costs by increasing the batch size with replay samples. Lastly, to maintain a fair comparison, the NICE algorithm uses all age-0 neurons in the final episode.

4.1. Comparison with Replay Methods

We compare NICE with eight replay methods[1]. We only consider raw replay methods because generative and activation replay often attempt to approximate raw replay. We establish a lower bound by employing SGD without any measures to prevent forgetting, and we denote an upper bound as Joint, which involves training on all classes simultaneously. We ensure that the same memory budget is maintained for both NICE and the replay methods. This is done by computing NICE's memory usage in the worst-case scenario. For instance, in the early episodes, we store only activations of neurons of certain age rather than all. However, we calculate memory usage assuming that each memory entry occupies a number of bits equal to the total number of neurons. Therefore, NICE's actual memory usage remains significantly below the allocated budget.

Table 2 presents the average accuracy across all classes following the last episode. We report results for a standard memory budget (refer to **Table** 1) and for increased budgets (doubled, denoted as $\times 2$, and quadrupled, denoted as $\times 4$). Note that NICE does not require a large memory, and our estimates are based on a worst-case, while actual memory usage is significantly lower. Nevertheless, to provide a comprehensive overview, we include experiments with increased memory budgets. Also, **Figure** 4 shows the accuracy after each learning episode across three datasets for the top-performing methods (NICE, iCaRL, TAMiL, x-DER).

We observe that NICE outperforms all baselines when utilizing standard and $\times 2$ budgets in terms of final accuracy (**Table** 2). The only exceptions are on CIFAR100 and Tiny ImageNet with the $\times 2$ budget. Furthermore, in the Supplementary Material Table 5, we present forgetting scores for the top-performing methods. We observe that NICE exhibits less forgetting across all datasets on both standard and $\times 2$ budgets. To our knowledge, this is the first instance of a replay-free CL approach outperforming replay-based methods in a CIL context, an achievement that we believe will encourage further exploration into architectural methods functioning without artificial task identifiers.

Surprisingly, iCaRL, one of the earliest replay methods in the literature, shows robust performance across all datasets and memory budgets. For instance, iCaRL is among the top-3 performing methods on all datasets for standard and $\times 2$ memory budgets, while more recent baselines such as x-DER and TAMiL lag behind significantly on some datasets like CIFAR10. The robustness of iCaRL can be attributed to its sample selection strategy, which selects images that closely approximate the class mean embedding. As a result, iCaRL identifies a core subset of examples that best represent the seen classes, optimizing the use of the memory budget. Furthermore, its knowledge distillation component avoids abrupt changes in old class knowledge. Although these results suggest iCaRL is a robust choice for replay, it has a practical limitation. iCaRL uses a nearest-mean-of-exemplars classifier for predictions, which necessitates passing all replay samples through the network to compute the class mean embeddings before making predictions. This introduces considerable overhead and may be impractical in scenarios, where the model must be ready to classify at any moment – not only at episode boundaries.

With a budget quadrupled ($\times 4$), replay methods begin to outperform NICE. In such a high-memory scenario, replay is expected to dominate, as the setting starts to resemble i.i.d. conditions where models can access a substantial number of previously seen examples.

4.2. Relationship between activations and context

NICE infers context through activations, although neurons are not explicitly trained to respond distinctly to different episode classes. This poses an intriguing question: How do neuron activations enable context detection?

To explore this, we ran NICE on the CIFAR10 and scrutinize the oldest neurons linked with episode-1 classes. We examined these neurons' responses to familiar episode-1 classes and the novel classes that appeared after the neurons had been frozen. We analyzed 300 thresholded activations from each episode. Given the extensive number of neurons, we used a Random Forest to pick the top-250 neurons by feature importance for visualization. Figure 5 presents the contrasting activation patterns for episode-1 and the other classes. Remarkably, the top-250 neurons showed stronger activations for learned classes, weaker for unfamiliar, which allows distinguishing between the two types of classes. We noted similar findings across different datasets and architectures (see the Supplementary Material Section-D.2).

This phenomenon aligns with a recent discovery in the out-of-distribution (OOD) detection that inspired an OOD

[1] We implemented all the baselines by customizing the PyTorch continual learning framework, Mammoth [14].

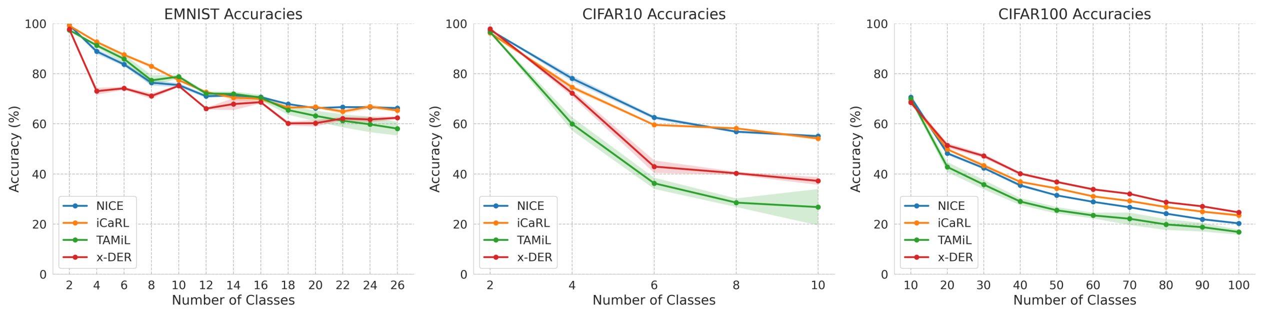

Figure 4. Accuracies for seen classes on EMNIST, CIFAR10, and CIFAR100 throughout training. Standard memory budget.

Table 2. Accuracy after all episodes on three memory budgets. Results averaged across three seeds. C stands for CIFAR. Standard deviations presented in Supplementary Material Section-D.

	Datasets					
Method	MNIST	Fashion	EMNIST	C10	C100	TinyImgNet
Joint	98.4	86.8	92.1	75.8	41.3	38.0
NICE	**83.4**	**73.9**	**66.3**	**55.1**	20.3	**11.8**
iCaRL [52]	79.7	68.7	65.3	54.2	23.5	10.3
TAMiL [10]	81.1	68.2	58.1	26.8	16.8	7.2
x-DER [13]	77.6	65.3	62.4	37.2	**24.7**	11.6
DER [14]	67.6	65.4	47.5	27.3	13.1	9.6
GDumb [50]	60.3	62.6	43.8	30.2	10.4	2.7
ER [19]	66.5	58.3	39.0	23.1	8.9	8.8
FDR [9]	64.3	52.7	42.2	27.5	9.6	10.2
A-GEM [18]	33.9	42.6	15.5	21.2	7.7	10.2
NICE (×2)	**85.9**	**75.1**	**69.4**	**56.5**	22.0	12.2
iCaRL (×2)	84.6	74.5	67.9	56.3	24.6	13.6
TAMiL (×2)	84.4	70.6	67.4	34.1	21.7	11.6
x-DER (×2)	85.6	71.9	68.6	42.6	**25.7**	**14.8**
DER (×2)	80.7	73.4	63.6	35.4	18.5	9.8
GDumb (×2)	73.4	66.4	56.7	37.0	14.5	3.8
ER (×2)	79.7	66.6	53.8	31.5	12.7	8.6
FDR (×2)	75.9	64.8	54.0	35.5	12.3	10.1
A-GEM (×2)	64.8	45.5	17.6	21.9	7.8	10.4
NICE (×4)	87.7	76.3	71.7	57.6	23.0	12.3
iCaRL (×4)	85.8	76.1	69.1	**59.1**	25.8	15.8
TAMiL (×4)	**92.7**	77.2	**78.6**	37.2	24.8	13.3
x-DER (×4)	91.6	75.8	74.8	50.0	**28.5**	**16.8**
DER (×4)	90.6	**77.7**	75.3	41.1	25.2	10.0
GDumb (×4)	86.4	74.2	65.8	41.8	20.8	4.7
ER (×4)	88.2	74.9	64.7	38.7	17.8	8.8
FDR (×4)	84.8	72.6	64.8	45.0	18.2	10.1
A-GEM (×4)	71.6	44.5	15.8	21.1	7.7	10.5
SGD	19.9	20.0	6.4	19.1	7.2	10.2

detection method ReACT [64]. They demonstrated that neurons present highly distinctive activation signatures for OOD samples. Specifically, for in-distribution (ID) data, neurons show stable activations, while their responses to OOD data are more variable. ReACT also provides a theoretical explanation for why thresholding reduces OOD activations more than ID activations, increasing separability. Even though all classes originate from the same dataset, such as CIFAR10, we can regard all classes other than episode 1 classes as OOD samples for the oldest neurons since they never learned these classes. Consequently, NICE may be using novelty as a means to determine context.

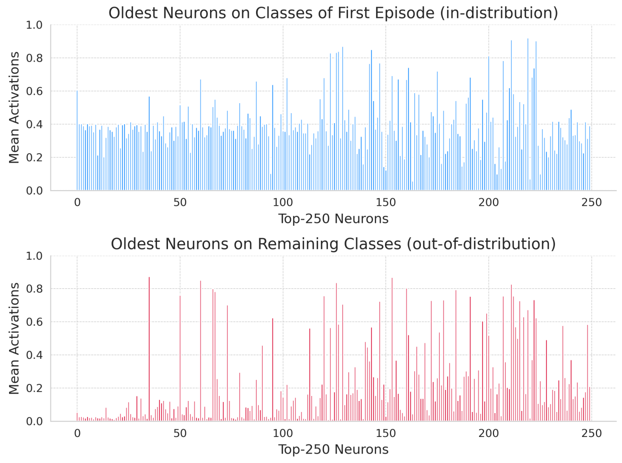

Figure 5. Average activations of the top 250 neurons for episode-1 classes (top) and for the remaining classes (bottom).

4.3. Which layers are important for context?

Throughout the experiments, we stored the activations of all neurons across all layers. This raises an important question: Is a single layer, or just a few, sufficient to achieve high accuracy in context detection? If this were the case, we could store the neurons at those layers and reduce memory cost. To test this, we ran NICE on the CIFAR10 and CIFAR100 separately, using only one layer of the VGG11 architecture to detect context. Figure 6 (left) displays these results. It is evident that none of the individual layers matched the performance achieved by using all layers.

Furthermore, Figure 6 (right) shows results where we considered several layers from input to output (indicated by the blue line) and from output to input (indicated by the red line). We observe that overall, we need most of the layers to perform comparably to using all layers. It seems that we could avoid storing 1-2 layers from either the bottom or top of the network without a significant drop in performance. However, determining this in a CL setting is challenging, so we suggest using all layers for context detection. Nevertheless, in a scenario with limited memory, one could

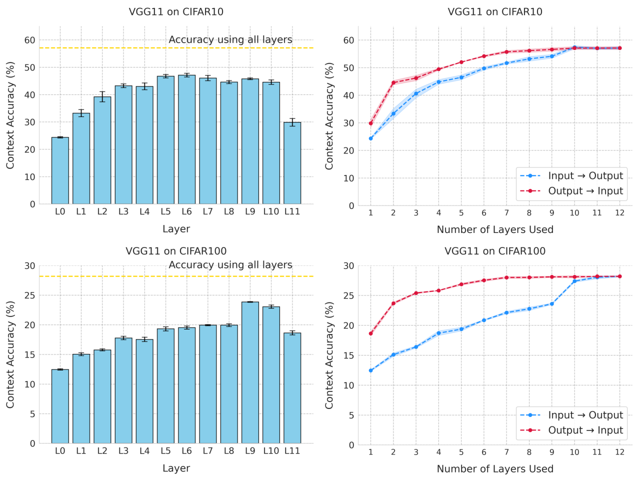
Figure 6. Left: Context detection accuracy when using only one layer. "L0" denotes the input layer, or in simpler terms, the average channel value of the input images. Right: Context detection accuracy using multiple layers, either cumulatively from input to output or vice versa. All results are the average of three seeds.

potentially use the coefficients of fitted logistic regression models to identify the most useful neurons throughout the architecture and purge the rest from memory. Additionally, compression algorithms such as Huffman coding could be employed to represent the memory using fewer bits.

4.4. Which layers are depleted first?

As neurons mature, they lose plasticity and become frozen. Consequently, NICE will eventually exhaust the available neurons as we learn more and more classes. Figure 7 illustrates the proportion of mature (frozen) neurons as we sequentially learn the classes of CIFAR10 and CIFAR100. We note that the earlier layers become frozen quickly and are depleted early in training, while the later layers remain available to learn additional classes. It is believed that the early layers of DNNs capture low-level features that are generalizable, while the later layers specialize in memorizing samples to map them to the classes. Therefore, the depletion of early layers should not impair performance if future classes are from the same domain. For example, once the first few layers have learned 20 CIFAR100 classes, their representations should be useful for the remaining classes. However, if the domain of future classes changes significantly, NICE may need to add neurons into the earlier layers, which can be accomplished on-the-fly.

5. Limitations and Future Work

Despite the promising results, NICE has several limitations. Firstly, chaining multiple logistic regressions represents a simplistic method of modeling the relationship between activations and episode classes. A issue is that obser-

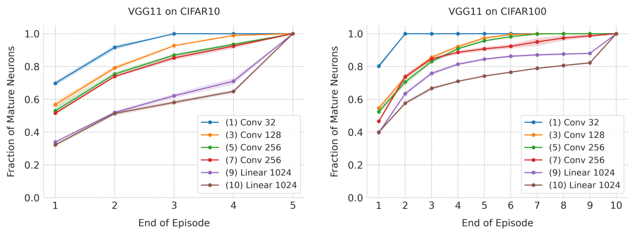
Figure 7. Fraction of mature neurons in the network for CIFAR10 and CIFAR100 experiments. Results are average of three seeds.

vations corresponding to each episode vary in dimensionality, making it challenging to fit a single model effectively. However, sequential learning techniques, such as attention mechanisms, could be utilized to learn from these variable-sized observations. Secondly, NICE eventually exhausts the available neurons. A potential strategy to mitigate this issue could be to reduce the age of some neurons to reallocate capacity, allowing for the graceful forgetting of earlier classes.

Another question for exploration is how to integrate replay techniques with NICE. For example, age transitions and freezing rules can be adjusted to incorporate replay within a limited window (e.g., replaying classes from the most recent one or two episodes). Thus, replay does not need to revisit all classes. Also, NICE could start with raw replay, and as layers become frozen, it can use these layers as a frozen feature extractor and switch to activation replay. So, NICE can enable activation replay without pretraining. Furthermore, we could explore using replay to encourage age-1 neurons to fire differently for replayed and current samples. This could potentially enhance the context detector's ability to discriminate between episodes.

6. Conclusion

NICE is a CL method inspired by adult neurogenesis in the dentate gyrus. It labels neurons according to their stages of maturation and learns from their binary activations to predict the appropriate episodic context. Our results demonstrate that NICE surpasses state-of-the-art replay methods across various datasets. Additionally, we have analyzed the activations across different classes and posited that neurons exhibit unique activation patterns for familiar and unfamiliar classes. To the best of our knowledge, NICE is the first replay-free method to outperform replay methods in Class Incremental Learning (CIL). We hope this achievement will inspire further research in developing CL architectures that do not rely on task identifiers.

Acknowledgements

This work was supported by the National Science Foundation (Award: 2039741). We are grateful to CiCi (Xingyu) Zheng and the CVPR 2024 reviewers for their constructive comments.

References

[1] Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. In *Advances in Neural Information Processing Systems 32*, pages 4394–4404. Curran Associates, Inc., 2019. 3

[2] James Aimone, Janet Wiles, and Fred Gage. Computational influence of adult neurogenesis on memory encoding. *Neuron*, 61:187–202, 2009. 2

[3] James Aimone, Wei Deng, and Fred Gage. Resolving new memories: A critical look at the dentate gyrus, adult neurogenesis, and pattern separation. *Neuron*, 70:589–96, 2011. 2

[4] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3366–3375. IEEE, 2017. 3

[5] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *The European Conference on Computer Vision (ECCV)*, 2018. 3

[6] Elahe Arani, Fahad Sarfraz, and Bahram Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. In *International Conference on Learning Representations*, 2022. 2

[7] Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony V. Robins. Pseudo-recursal: Solving the catastrophic forgetting problem in deep neural networks. *arXiv preprint*, abs/1802.03875, 2018. 3

[8] Ali Ayub and Alan Wagner. {EEC}: Learning to encode and regenerate images for continual learning. In *International Conference on Learning Representations*, 2021. 3

[9] Ari S. Benjamin, David Rolnick, and Konrad P. Körding. Measuring and regularizing networks in function space. *CoRR*, abs/1805.08289, 2018. 2, 7

[10] Prashant Shivaram Bhat, Bahram Zonooz, and Elahe Arani. Task-aware information routing from common representation space in lifelong learning. In *The Eleventh International Conference on Learning Representations*, 2023. 7

[11] Zalán Borsos, Mojmir Mutny, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. In *Advances in Neural Information Processing Systems*, pages 14879–14890. Curran Associates, Inc., 2020. 2

[12] Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *CoRR*, abs/2201.00766, 2022. 2

[13] M. Boschini, L. Bonicelli, P. Buzzega, A. Porrello, and S. Calderara. Class-incremental continual learning into the extended der-verse. *IEEE Transactions on Pattern Analysis amp; Machine Intelligence*, 45(05), 2023. 7

[14] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems*, 2020. 2, 6, 7

[15] Lucas Caccia, Eugene Belilovsky, Massimo Caccia, and Joelle Pineau. Online learned continual compression with adaptive quantization modules. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1240–1250. PMLR, 2020. 3

[16] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. In *International Conference on Learning Representations*, 2022. 1, 2

[17] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 3

[18] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2019. 7

[19] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. Continual learning with tiny episodic memories. In *Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019. 1, 2, 7

[20] C.D. Clelland, Minee-Liane Choi, Carola Romberg, Gregory Clemenson, Alexandra Fragniere, Pamela Tyers, S Jessberger, Lisa Saksida, Roger Barker, F.H. Gage, and Tim Bussey. A functional role for adult hippocampal neurogenesis in spatial pattern separation. *Science (New York, N.Y.)*, 325:210–3, 2009. 2, 4

[21] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021. 1, 2

[22] Dumitru Erhan, Y. Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Technical Report, Univeristé de Montréal*, 2009. 3

[23] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning, 2019. 1

[24] Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual learning via neural pruning. *arXiv preprint*, abs/1903.04476, 2019. 3

[25] Ian J. Goodfellow, Mehdi Mirza, Xia Da, Aaron C. Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *International Conference on Learning Representations*, 2014. 1

[26] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, 2017. 2

[27] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017. 2

[28] Mustafa Burak Gurbuz and Constantine Dovrolis. Nispa: Neuro-inspired stability-plasticity adaptation for continual

learning in sparse networks. *International Conference on Machine Learning*, 30, 2022. 3

[29] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2, 3

[30] Tyler L Hayes, Giri P Krishnan, Maxim Bazhenov, Hava T Siegelmann, Terrence J Sejnowski, and Christopher Kanan. Replay in deep learning: Current approaches and missing biological elements. *Neural Computation*, 33(11):2908–2950, 2021. 2

[31] Yen-Chang Hsu, Yen-Cheng Liu, and Zsolt Kira. Re-evaluating continual learning scenarios: A categorization and case for strong baselines. 2018. 1, 2, 3

[32] Hengyuan Hu, Rui Peng, Yu-Wing Tai, and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint*, abs/1607.03250, 2016. 3

[33] Sangwon Jung, Hongjoon Ahn, Sungmin Cha, and Taesup Moon. Continual learning with node-importance based adaptive group sparse regularization. In *Advances in Neural Information Processing Systems*, 2020. 3

[34] Haeyong Kang, Rusty John Lloyd Mina, Sultan Rizky Hikmawan Madjid, Jaehong Yoon, Mark Hasegawa-Johnson, Sung Ju Hwang, and Chang D. Yoo. Forget-free continual learning with winning subnetworks. In *Proceedings of the 39th International Conference on Machine Learning*, pages 10734–10750. PMLR, 2022. 3

[35] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *International Conference on Learning Representations*, 2018. 3

[36] Ronald Kemker, Marc Mcclure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. Measuring catastrophic forgetting in neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32, 2017. 1, 2, 3

[37] 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, 2017. 3

[38] Dhireesha Kudithipudi, Mario Aguilar-Simon, Jonathan Babb, Maxim Bazhenov, Douglas Blackiston, Josh Bongard, Andrew Brna, Suraj Chakravarthi Raja, Nick Cheney, Jeff Clune, Anurag Daram, Stefano Fusi, Peter Helfer, Leslie Kay, Nicholas Ketz, Zsolt Kira, Soheil Kolouri, Jeff Krichmar, Sam Kriegman, and Hava Siegelmann. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4:196–210, 2022. 2

[39] Mark Kurtz, Justin Kopinsky, Rati Gelashvili, Alexander Matveev, John Carr, Michael Goin, William Leiserson, Sage Moore, Nir Shavit, and Dan Alistarh. Inducing and exploiting activation sparsity for fast neural network inference. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 3

[40] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 3

[41] Zheda Mai, Ruiwen Li, Hyunwoo Kim, and Scott Sanner. Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3589–3599, 2021. 2

[42] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018. 3

[43] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82, 2018. 3

[44] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 1989. 1

[45] Jorge A Mendez and ERIC EATON. Lifelong learning of compositional structures. In *International Conference on Learning Representations*, 2021. 3

[46] Matías Mugnaini, Mariela F. Trinchero, Alejandro F. Schinder, Verónica C. Piatti, and Emilio Kropff. Unique potential of immature adult-born neurons for the remodeling of ca3 spatial maps. *Cell Reports*, 42(9):113086, 2023. 2

[47] Toshiaki Nakashiba, Jesse Cushman, Kenneth Pelkey, Sophie Renaudineau, Derek Buhl, Thomas Mchugh, Vanessa Rodriguez Barrera, Ramesh Chittajallu, Keisuke Iwamoto, Chris McBain, Michael Fanselow, and Susumu Tonegawa. Young dentate granule cells mediate pattern separation, whereas old granule cells facilitate pattern completion. *Cell*, 149:188–201, 2012. 2, 4

[48] German Parisi, Ronald Kemker, Jose Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019. 1, 2

[49] Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. Fetril: Feature translation for exemplar-free class-incremental learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3911–3920, 2023. 3

[50] Ameya Prabhu, Philip Torr, and Puneet Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *The European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 7

[51] Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *Neurocomputing*, 404, 2020. 3

[52] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph Lampert. icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 7

[53] Minsoo Rhu, Mike O'Connor, Niladrish Chatterjee, Jeff Pool, Youngeun Kwon, and Stephen Keckler. Compressing dma engine: Leveraging activation sparsity for training deep

neural networks. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2018. 3

[54] Matthew Riemer, Tim Klinger, Djallel Bouneffouf, and Michele Franceschini. Scalable recollections for continual lifelong learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:1352–1359, 2019. 3

[55] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pages 3738–3748, 2018. 3

[56] Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 42, 2018. 3

[57] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint*, abs/1606.04671, 2016. 3

[58] 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. In *ICML*, 2018. 3

[59] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, 2018. 3

[60] Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9630–9638, 2021. 2

[61] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017. 3

[62] Ghada Sokar, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Spacenet: Make free space for continual learning. *Neurocomputing*, 439, 2021. 2, 3

[63] Ghada Sokar, Decebal Constantin Mocanu, and Mykola Pechenizkiy. Avoiding forgetting and allowing forward transfer in continual learning via sparse networks. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2022. 2, 3

[64] Yiyou Sun, Chuan Guo, and Yixuan Li. React: Out-of-distribution detection with rectified activations. In *Advances in Neural Information Processing Systems*, 2021. 7

[65] Sophie Tronel, Laure Belnoue, Noëlle Grosjean, Jean-Michel Revest, pier-vincenzo Piazza, Muriel Koehl, and Djoher Abrous. Adult-born neurons are necessary for extended contextual discrimination. *Hippocampus*, 22:292–8, 2012. 2, 4

[66] Gido van de Ven, Hava Siegelmann, and Andreas Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications*, 11, 2020. 2, 3

[67] Gido van de Ven, Tinne Tuytelaars, and Andreas Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4:1–13, 2022. 1, 3

[68] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. *arXiv preprint*, abs/1904.07734, 2019. 1

[69] Tom Veniat, Ludovic Denoyer, and MarcAurelio Ranzato. Efficient continual learning with modular networks and task-driven priors. In *International Conference on Learning Representations*, 2021. 3

[70] Kai Wang, Joost van de Weijer, and Luis Herranz. Acae-remind for online continual learning with compressed feature replay. *Pattern Recognition Letters*, 150:122–129, 2021. 3

[71] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application, 2023. 1, 2

[72] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. In *Advances in Neural Information Processing Systems*, 2020. 3

[73] Mengqi Xue, Haofei Zhang, Jie Song, and Mingli Song. Meta-attention for vit-backed continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 150–159. IEEE/CVF, 2022. 3

[74] Michael Yassa and Craig Stark. Pattern separation in the hippocampus. *Trends in neurosciences*, 34:515–25, 2011. 2, 4

[75] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018. 3

[76] 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. 2

[77] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 2014. 3

[78] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, 2017. 3