Learning Bayesian Sparse Networks with Full Experience Replay for Continual Learning

Qingsen Yan[1†], Dong Gong[2,1†], Yuhang Liu[1], Anton van den Hengel[1], Javen Qinfeng Shi[1*]
[1]The Australian Institute for Machine Learning, The University of Adelaide, Australia
[2]School of Computer Science and Engineering, The University of New South Wales, Australia

2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) | 978-1-6654-6946-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CVPR52688.2022.00021

Abstract

Continual Learning (CL) methods aim to enable machine learning models to learn new tasks without catastrophic forgetting of those that have been previously mastered. Existing CL approaches often keep a buffer of previously-seen samples, perform knowledge distillation, or use regularization techniques towards this goal. Despite their performance, they still suffer from interference across tasks which leads to catastrophic forgetting. To ameliorate this problem, we propose to only activate and select sparse neurons for learning current and past tasks at any stage. More parameters space and model capacity can thus be reserved for the future tasks. This minimizes the interference between parameters for different tasks. To do so, we propose a Sparse neural Network for Continual Learning (SNCL), which employs variational Bayesian sparsity priors on the activations of the neurons in all layers. Full Experience Replay (FER) provides effective supervision in learning the sparse activations of the neurons in different layers. A loss-aware reservoir-sampling strategy is developed to maintain the memory buffer. The proposed method is agnostic as to the network structures and the task boundaries. Experiments on different datasets show that SNCL achieves state-of-the-art result for mitigating forgetting.

1. Introduction

Humans continually acquire new skills and knowledge while maintaining the ability to perform tasks they learned in childhood. Continual Learning (CL) [26, 28] aims to mimic this process, enabling learning of new tasks sequentially without storing all of the associated data. However, even with the help of CL, deep neural networks often suffer from serious performance degradation on previously learned tasks when learning new ones. This phenomenon is known as catastrophic forgetting [23].

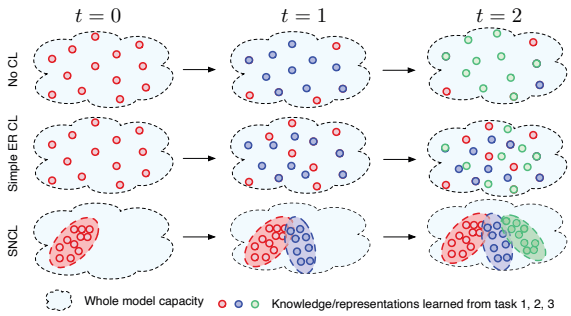


Figure 1. Without continual learning, training on a new task interferes with the knowledge learned in mastering previous tasks, thus introducing the risk of catastrophic forgetting. Simple experience replay approaches can help preserve the learned knowledge. However, the regularizations work on the network parameters as a whole, invariably leading to forgetting. The proposed SNCL learns sparse networks at all stages, enabling more intrinsic and common representations with less model capacity. It reserves more capacity/parameter space for future tasks, resulting in less interference and forgetting.

The above issues motivate more effective CL algorithms [6, 8, 24, 36] to mitigate catastrophic forgetting when the number of tasks and associated knowledge increases over time. For this purpose, existing approaches modify the the network learning strategies in different ways, such as the regularization methods [17, 20, 39], memory-based experience replay [4, 7, 8, 22], and dynamic modular approaches [1, 29, 37]. Specifically, Experience Replay (ER) methods [4, 9, 22] mitigate catastrophic forgetting by replaying a selection of previously seen data maintained in a small episodic memory along with the new task data. Knowledge Distillation (KD) [12, 14] based methods [4, 20, 26] alleviate forgetting and encourage knowledge transfer by applying the old models as teachers. All these approaches learn all sequential tasks with the same whole parameter space. The learner potentially uses all neurons in the over-complete network to decrease the loss on current data. These characteristics inhibit the effectiveness in reducing the interference and invariably lead to forgetting (See Fig. 1).

*† indicates equal contribution. D. Gong (edgong01@gmail.com) is the corresponding author. This work was partly supported by the Centre for Augmented Reasoning at the Australian Institute for Machine Learning.

978-1-6654-6946-3/22/$31.00 ©2022 IEEE
DOI 10.1109/CVPR52688.2022.00021

In this work, we propose to learn Sparse neural Networks for Continual Learning (SNCL). We enforce the sparsity of the network neurons in the learning stages to reserve more parameters for future tasks. This prevents learning a new task from interfering with the parameters critical to achieving previously learned tasks (See Fig. 1). Specifically, we introduce to use variational Bayesian sparsity (VBS) priors on activations to induce a sparse network in a principled manner. The sparse prior helps the model to learn a given task in the most compact set of neurons. This allows far greater control over the process of learning, and forgetting. It particularly enables preserving of resources for future learning, and alleviates potential interference when new tasks are learned. A small replay buffer allows the commonality between new and previous tasks to be exploited, thus increasing the network's total learning capacity and reducing the risk of forgetting.

The variational Bayesian framework flexibly allows learning the sparse networks by only relying on the back-propagation of the supervision signal (*e.g.*, the classification loss). As a result, instead of performing experience replay with only data labels, we propose a more effective experience replay strategy by enforcing the stability of the intermediate representations of old samples. We label this approach *Full Experience Replay* (FER). To achieve this we store not only the past data in the memory but also the corresponding logits and intermediate layer features. FER thus helps activate the appropriate neurons in each layer more effectively, which helps avoid representation drift. Note that, in contrast to many similar knowledge distillation-based methods [20, 26], the proposed approach is not limited to the case where the task boundaries are known. We also introduce an effective Loss-aware Reservoir Sampling (LRS) strategy to maintain the memory, which selects samples on the basis of their ability to improve performance. In summary, our contributions are as follows:

- We propose to learn sparse networks for continual learning with variational Bayesian sparsity priors on the neurons (SNCL). The proposed method alleviates the catastrophic forgetting and interference by enforcing sparsity of the networks to reserve parameters for future tasks. The memory based experience reply allows the commonality between new and previous tasks to be exploited, thus increasing the network's total learning capacity and reducing the risk of forgetting.
- We propose Full Experience Replay (FER) to store and replay with old samples' intermediate layer features generated when observed in data steam, unlike previous methods with only labels. It provides more effective supervision on learning the sparse activation of the neurons at different layers. A loss-aware reservoir sampling strategy is proposed to maintain the memory.
- The proposed approaches can generally improve the performance of the ER based methods, achieving state-of-the-art results under the same setting. The proposed method is agnostic and general to the network structures and task boundaries.

2. Related Work

2.1. Continual Learning Settings

Early work in CL focussed on Task Incremental Learning (**Task-IL**) [10,17,32,39] whereby models are trained on a series of distinct tasks with clear boundaries and no overlap. Recent work has focused on a more realistic setting whereby tasks may change gradually, named Class Incremental Learning (**Class-IL**) [15, 25, 26, 40]. In Class-IL, models are not provided with task boundaries during testing and must the infer task identity from the data. Although huge advance has been obtained, the majority of Task-IL and Class-IL approaches are unsuited for real-world applications, where task boundaries are unclear and overlap. Recently, General Continual Learning (**GCL**) [4, 10] has been introduced whereby: (i) the training and testing phases do not rely on boundaries between tasks; (ii) the testing phase does not use task identifiers; (iii) Memory size is limited. These challenging guidelines guarantee that GCL can be applied to practical scenarios.

2.2. Continual Learning Approaches

Regularization Based Methods. Regularization based methods use regularization terms in the loss function to encourage stability on previous tasks and prevent forgetting. Kirkpatrick *et al*. proposed Elastic Weight Consolidation (EWC) [17] which evaluates the importance of parameters using the Fisher information matrix. Li *et al*. [20] regularized the network using knowledge distillation [14]. To alleviate forgetting, Tao *et al*. [33] propose maintaining the topology of the network's feature space. Yu *et al*. [38], in contrast, approximate the drift in previous task performance based on the drift that is experienced by current task data.

Parameter Isolation Based Methods. Parameter isolation based methods are only employed in Task-IL. To alleviate the forgetting, related approaches allocate each task with different subnetwork's parameters. Rusu *et al*. [29] proposed progressive networks to leverage prior knowledge via lateral connections to previously learned features. Xu *et al*. [36] searched for the best neural network architecture for each coming task by leveraging reinforcement learning strategies. Saha *et al*. [30] enabled a network to learn continually and efficiently by partitioning the learned space into a core space and a residual space. Golkar *et al*. [11] isolate the parameters for different tasks via neural pruning and regain model capacity with graceful forgetting for performance. In [3], neural inhibition is used to impose sparsity in neural networks for CL.

Replay Based Methods. To reduce the forgetting, replay-based methods store a subset of previously-seen data, or important gradient spaces from the previous tasks in a memory buffer, or produce synthetic data using generative models for pseudo-rehearsal. Chaudhry *et al*. [8] proposed Experience Replay (ER) that jointly trains the model with the data from the new tasks and memory. Buzzega *et al*. [4] improved ER, named Dark Experience Replay (DER), by mixing rehearsal with knowledge distillation and regularization. Gradient Episodic Memory (GEM) [22] and Averaged GEM (A-GEM) [7] used samples from the memory buffer to estimate gradient constraint so that loss on the previous task does not increase.

3. Learning Sparse Networks for Continual Learning

A CL problem can be formulated as learning from an ordered sequence of datasets, each from one of T tasks $(\mathcal{D}^1, ..., \mathcal{D}^T)$, where $\mathcal{D}^t = \{(x_i^t, y_i^t)\}_{i=1}^{N_t}$ represents the data examples of t-th task. During each task t, the samples in $\mathcal{D}^t = \{(x_i^t, y_i^t)\}_{i=1}^{N_t}$ are drawn from an i.i.d. distribution associated with the task. The goal is to carry out sequential training while maintaining the performance of the previous tasks. The task boundaries can be available or unavailable (*e.g*., the GCL setting) during both training and testing stages under different settings. CL aims to learn a function f_θ, with parameters θ, for all the tasks by being optimized on one task at a time in a sequential manner.

3.1. Overview

We focus on the CL classification problem in this paper. Given any sample x, we let $z_\theta(x)$ represent the output logits of the f_θ. For classification, predictions for x is made using $f_\theta(x) = \text{softmax}(z_\theta(x))$. Formally, the objective of CL can be formulated as learning an f_θ, *i.e*., the parameter θ, that can minimize the classification loss on all the tasks $t = \{1, ..., T\}$:

$$\mathcal{L}_{\text{CE}}^t = \mathbb{E}_{(x,y)\sim\mathcal{D}^t} \ell(f_\theta(x), y), \tag{1}$$

where $\ell(\cdot, \cdot)$ represents cross entropy loss. However, it is challenging since the datasets D^t's are observed in a sequential manner. When we train the model on any task t, all the previous data samples are unavailable. The knowledge/representations learned on past tasks are easily flushed by the new task data with distribution shifting, leading to catastrophic forgetting. To encourage the parameters to adapt to new tasks and maintain the knowledge from past tasks, as many CL approaches [8, 9, 22], we employ a replay buffer $\mathcal{M}$ as an episodic memory to preserve a small subset of old experiences, *i.e*., samples in old tasks. During training, we use the newly arriving data of the current task and the examples sampled from the reply buffer to optimize the parameters. In our work, memory is updated in a modified Reservoir Sampling method [9], *i.e*. LRS, being agnostic to the task boundaries. The cross entropy based classification loss on memory can be formulated as

$$\mathcal{L}_{\text{CE-M}} = \mathbb{E}_{(x,y)\sim\mathcal{M}} \ell(f_\theta(x), y). \tag{2}$$

In a sense, the replay memory can help the model be aware of the past tasks and provide some hints about the old knowledge. By applying only the loss functions in Eq. (1) and (2), we can arrive at the basic experience replay (ER) based CL methods [8, 22].

Although simply replaying the old data can help the model to preserve the performance on old tasks, the methods [4, 22] still learn all the sequential data with the same whole parameter space. To decrease the loss on current data, the learner potentially uses all neurons and capacities in the over-complete network, when there is no restriction. Although the memory provides some gradients of the past tasks, interference on the parameters and forgetting can still easily happen, as shown in Fig. 1. This motivates us to study more effective capacity/parameter management mechanism CL. We thus propose to enforce sparsity on the network f_θ as shown in Fig. 2. In this way, the learner uses the most sustainable way to learn the current (and past) data, and thus reserves more capacity for the future tasks, alleviating possible interference in the future. As shown in Sec. 3.2, we learn the sparse network by applying variational Bayesian sparsity priors on the neurons. Relying on the supervision from the current data and past data in the memory, the proposed method automatically activates only a sparse set of neurons to maintain current and learned knowledge *jointly*. It also leads to more intrinsic and common knowledge for the seen tasks so far. To further enhance the supervisions for learning the sparse neurons across different layers, we propose the Full Experience Replay (FER) by directly enforcing the stability of the intermediate representations of the old samples (See Sec. 3.3).

3.2. Bayesian Sparse Networks

To learn sparse networks for CL, we introduce variational Bayesian sparse (VBS) prior on the activations, *i.e*., neuron responses on x, in the neural network f_θ. For a network f_θ with L layers, we let $h_{\theta,l,c}(x)$ denote the c-th neuron at the l-th layer, where $l = \{1, ..., L\}$ and $c = \{1, ..., C_l\}$ denote the index of the layers and and the neuron at the l-th layers, respectively. For a convolution layer l, $h_{\theta,l,c}(x)$ represents the c-th channel of the produced feature map (with C_l channels at l-th layer). For a fully connected linear layer, $h_{\theta,l,c}(x)$ represents the c-th column of the produced feature matrix.

To apply the VBS priors on the neurons, we firstly introduce a scalar indicator $\tau_{l,c}$ for the sparsity of the neuron at l-th layer c-th channel. We define $\tilde{h}_{\theta,l,c}(x)$ to denote the

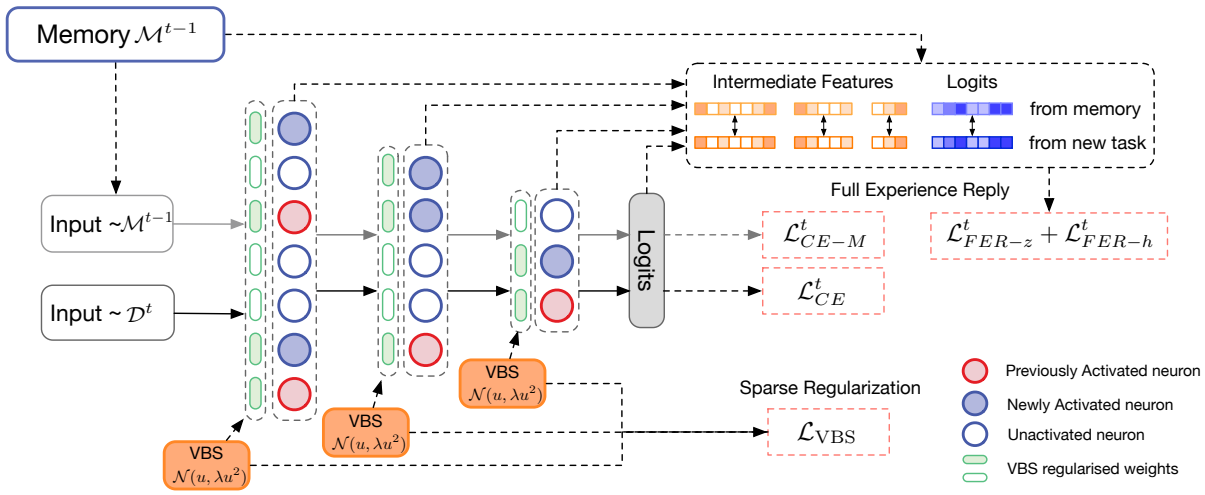


Figure 2. The framework of the proposed SNCL. It uses Variational Bayesian Sparse (VBS) and Full Experience Replay (FER) to learn new tasks and alleviate forgetting. $\mathcal{L}^t_{\text{CE}}$ is the t-th task loss. $\mathcal{L}_{\text{VBS}}$ alleviates the catastrophic forgetting and interference by enforcing sparsity of the networks. Full experience replay loss provides more effective supervision on learning the sparse activation of the neurons at different layers. The memory is updated by the proposed loss-aware reservoir sampling strategy.

non-sparse version of $h_{\theta,l,c}(x)$ produced in the network before applying the sparsity-inducing operation. For simplicity, without losing the generality, we induce the sparsity of the neurons by applying the scalar product on the activation representations [21] as follows:

$$h_{\theta,l,c}(x) = \tilde{h}_{\theta,l,c}(x) \odot \tau_{l,c}, \tag{3}$$

where $\odot$ denotes the pointwise product between the scalar $\tau_{l,c}$ and the activations $\tilde{h}_{\theta,l,c}(x)$. By enforcing sparsity on $\tau_{l,c}$'s, $h_{\theta,l,c}(x)$'s are sparse activations as shown in Fig. 2. $\tau_{l,c}$ plays a key role to activate the nonzero $h_{\theta,l,c}(x)$. To enforce sparsity on $\tau_{l,c}$, we impose a hierarchical prior with a zero-mean Gaussian distribution with the variance $\sigma_{l,c}$ sampled from a uniform hyper-prior (*i.e.*, $\mathcal{U}(a,b)$) [21]:

$$p(\tau_{l,c}) = \mathcal{N}(0, \sigma_{l,c}), \qquad p(\sigma_{l,c}) = \mathcal{U}(a,b). \tag{4}$$

Together with the following variational posterior with mean $\mu_{l,c}$ and variance $\lambda_{l,c}\mu^2_{l,c}$:

$$q(\tau_{l,c}) = \mathcal{N}(\mu_{l,c}, \lambda_{l,c}\mu^2_{l,c}), \tag{5}$$

we arrive at the following regularization term to encourage sparsity on $\mu_{l,c}$ (*i.e.*, $\tau_{l,c}$):

$$\mathcal{L}_{\text{VBS}} = 0.5 \sum_l^L \sum_c^{C_l} \log(1 + \lambda^{-1}_{l,c}). \tag{6}$$

As the posterior mean of $\tau_{c,l}$, $\mu_{l,c}$ can be regarded as the actual sparse weights on the activations. The two parameters, $\mu_{l,c}$ and $\lambda_{l,c}$, can be adaptively optimized via backpropagation from the classification, FER, and the sparse regularizer Eq. (6). $\lambda_{l,c}$ directly represents the sparsity level of each neuron. Specifically, when $\lambda^{-1}_{l,c} \to 0$, the activation $h_{\theta,l,c}(x)$ and the corresponding neuron will be set as zero and inactivated. The term in Eq. (6) is a concave, non-decreasing function on the domain $[0, +\infty)$, with respect to $\lambda^{-1}_{l,c}$. Since the proposed layer is implemented in a Bayesian framework, the sparsity of network is obtained to faithfully and adaptively fit the data in a principled way.

3.3. Full Experience Replay

In ER based methods, a memory buffer $\mathcal{M}$ is used to store the past samples. It stores the input x and label y of the samples and is maintained via sampling on the past data.

We learn the adaptive sparsity on the neurons only relying on the backpropagation of the loss. It is flexible but also renders the inefficient supervisions on the neurons at different layers, since the losses are added on the final output of the network. Instead of the ER only with original labels, we propose a Full Experience Replay (FER) strategy which retains the data, labels and features to the replay buffer. We formulate the replay buffer as $\mathcal{M} = \{\mathbf{m}[(x_i, y_i)]\}_{i=1}^M$ with $\mathbf{m}[(x_i, y_i)] = (x_i, y_i, z_{\hat{\theta}}(x_i), \{h_{\hat{\theta},l,c}(x_i)\})$, where $\hat{\theta}$ is the old model parameter, $z_{\hat{\theta}}(x_i)$ and $\{h_{\hat{\theta},l,c}(x_i)\}$ denote the logits and intermediate layer features of the past samples, and M denotes the buffer size. $\hat{\theta}$ denotes the parameter when each sample is trained with the model. So $\hat{\theta}$ for different samples should be different (with a slight notation abuse here). In this way the memory can maintain the knowledge from diverse status [4].

Relying on the memory with logits and features, we define the loss function of FER with the following components. $\mathcal{L}_{\text{FER-z}}$ preserves the logits of the past samples under the response of current model θ and old model $\hat{\theta}$:

$$\mathcal{L}_{\text{FER-z}} = \mathbb{E}_{(x, z_{\hat{\theta}}(x)) \sim \mathcal{M}} \| z_\theta(x) - z_{\hat{\theta}}(x) \|_2^2. \tag{7}$$

To provide more effectively supervision on learning the sparse activation of the neurons at different layers, $\mathcal{L}_{\text{FER-h}}$ is employed on all intermediate features:

$$\mathcal{L}_{\text{FER-h}} = \mathbb{E}_{(x,\{h_{\hat{\theta},l,c}(x)\})\sim\mathcal{M}} \sum_{l}^{L}\sum_{c}^{C_l} \|h_{\theta,l,c}(x) - h_{\hat{\theta},l,c}(x)\|_2^2. \tag{8}$$

Together with the ER loss on labels in Eq. (2), we arrive at the FER total loss as: The total loss of FER can be written as:

$$\mathcal{L}_{\text{FER}} = \alpha\mathcal{L}_{\text{CE-M}} + \beta\mathcal{L}_{\text{FER-z}} + \gamma\mathcal{L}_{\text{FER-h}}. \tag{9}$$

Although the formulates of FER loss are similar to the KD techniques, *e.g.*, LWF [20] and iCaRL [26], they are intrinsically different. The KD based LWF does not use a replay buffer and cannot use previous samples. Different to iCaRL employing the outputs of the network at the end of each task, we store both features and logits from the model when the sample is observed in the training process, helping to preserve diverse model status and be free to task boundries. Different to DER [4] only relying on the logits to play the past knowledge, our FER obtains the logits $z_{\hat{\theta}}(x)$ and features $\{h_{\hat{\theta},l,c}(x)\}$ for a full replay of the whole experience, which is tailored for promoting the estimation of the sparsity across all layers. Moreover, FER is also powerful for directly reliving the representation drifting of the past samples. With FER, SNCL is encouraged to learn shared features for new and old tasks under the sparse regularization, instead of simply dividing the networks into isolated sparse components.

3.4. The Overall Loss

In summary, the cross entropy loss $\mathcal{L}_{\text{CE}}^t$ focuses on learning the current data; $\mathcal{L}_{\text{VBS}}$ provides an adaptive sparsity regularization on the network neurons; $\mathcal{L}_{\text{FER}}$ helps to replay the full knowledge of the past samples. By integrating them together, the overall loss of SNCL can be written as:

$$\mathcal{L} = \sum\nolimits_{t=1}^{T} \mathcal{L}_{\text{CE}}^t + \eta\mathcal{L}_{\text{VBS}} + \mathcal{L}_{\text{FER}}. \tag{10}$$

3.5. Loss-aware Reservoir Sampling

Like all CL methods with ER, we maintain a small memory buffer for past data. For updating the memory, we rely on reservoir-type sampling (RS) [4,5,35] to avoid using the task boundary information. Under the setting of FER, we use the memory buffer $\mathcal{M}^k = \{\mathbf{m}_i\}_{i=1}^{M}$ to store the full knowledge of the sampled set of data seen until k-th learning step. When the memory is full, RS randomly overwrites the memory slot with the randomly selected new samples [9]. We modify the standard RS as Loss-aware Reservoir Sample (LRS) to maintain memory with more balanced distributions on different classes and training loss values (*i.e.*, learning difficulties).

Algorithm 1: Loss-aware Memory Updating

Input: Memory from previous stage $\mathcal{M}^{k-1}$, the sample set to store $\mathcal{B}$, memory size M
Output: Updated memory buffer $\mathcal{M}^k$

1 **if** $|\mathcal{M}^{k-1}| < M$ **then**
2 $\quad \mathcal{M}^k = \mathcal{M}^{k-1} \cup \{\mathbf{m}[(x,y)] | (x,y) \in \mathcal{B}\}$;
3 **else**
4 $\quad$ Obtain the number of classes in $\mathcal{M}^{t-1} \cup \mathcal{B}$ as R^k and initilize $\mathcal{M}^t = \{\} * M$;
5 $\quad$ **for** $r = 1, 2, ..., R^k$ **do**
6 $\quad\quad \mathcal{S}_r = \{(x,y) | y = r, (x,y) \in \mathcal{B} \cup \mathcal{M}^{k-1}\}$;
7 $\quad\quad$ Sort $\mathcal{S}_r$ w.r.t. the training loss values;
8 $\quad\quad$ Obtain $\mathcal{S}_r^*$ containing (M/R^k) samples with diverse loss values by sampling in the sorted $\mathcal{S}_r$ with interval $|\mathcal{S}_r|/(M/R^k)$;
9 $\quad\quad$ Update $\mathcal{M}^k = \mathcal{M}^k \cup \{\mathbf{m}[(x,y)] | (x,y) \in \mathcal{S}_r^*\}$;

Given a new data batch, LRS randomly samples a set of data $\mathcal{B}$ to store in the memory. Algorithm 1 summarizes the loss-aware memory updating process. If the memory is full, LRS sorts the data points w.r.t. the loss values, specifically for each class seen so far (step 7), and then samples the data with diverse loss values (step 8). As a result, the updated memory can cover balanced classes and learning difficulties. We abuse the notation $\cup$ in step 6 for simplicity.

4. Experiments

Datasets. We test the proposed method on several public datasets generally used in continual learning: sequential CIFAR-10 [22], sequential Tiny ImageNet [8], Permuted MNIST [17], Rotated MNIST [22] and MNIST-360 [4].

CIFAR-10 [18] includes 10 classes, each class has 5000 training examples and 1000 test examples. Tiny ImageNet [19] is a subset of ImageNet, which contains 200 classes, and each class has 500 training examples. Sequential CIFAR-10 [22] consists 5 sequential tasks, each of which has 2 classes. Sequential Tiny ImageNet is built by splitting 100 classes of Tiny ImageNet into 20 tasks where each task has 5 classes. Permuted MNIST [17], Rotated MNIST [22] both have 20 subsequent tasks, Permuted MNIST employs a random permutation to the pixels, Rotated MNIST rotates the inputs by a random angle in the interval $[0; \pi)$. Buzzega *et al*. randomly choose two MNIST numbers from 0 to 8 (*e.g.*, (0, 1), (1, 2), etc.) and rotates those numbers by an increasing angle. Then, MNIST-360 [4] designs a stream of data presenting batches of two consecutive digits at a time.

Experimental Protocol. We evaluate our method under different settings. We use Sequential CIFAR-10 and Sequential Tiny ImageNet to validate the proposed method on *Task-IL* setting. Following [4, 10], Sequential CIFAR-

Table 1. Classification results for standard CL benchmarks. Average accuracy across 10 runs.

Buffer	Method	S-CIFAR-10		S-Tiny-ImageNet		P-MNIST	R-MNIST
		Class-IL(↑)	Task-IL(↑)	Class-IL(↑)	Task-IL(↑)	Domain-IL(↑)	Domain-IL(↑)
-	JOINT	92.20±0.15	98.31±0.12	59.99±0.19	82.04±0.10	94.33±0.17	95.76±0.04
	SGD	19.62±0.05	61.02±3.33	7.92±0.26	18.31±0.68	40.70±2.33	67.66±8.53
-	oEWC [31]	19.49±0.12	68.29±3.92	7.58±0.10	19.20±0.31	**75.79±2.25**	**77.35±5.77**
	SI [39]	19.48±0.17	68.05±5.91	6.58±0.31	36.32±0.13	65.86±1.57	71.91±.83
	LwF [20]	**19.61±0.05**	63.29±2.35	**8.46±0.22**	15.85±0.58	-	-
	PNN [29]	-	**95.13±0.72**	-	**67.84±0.29**	-	-
200	ER [27]	44.79±1.86	91.19±0.94	8.49±0.16	38.17±2.00	72.37±0.87	85.01±1.90
	GEM [22]	25.54±0.76	90.44±0.94	-	-	66.93±1.25	80.80±1.15
	A-GEM [7]	20.04±0.34	83.88±1.49	8.07±0.08	22.77±0.03	66.42±4.00	81.91±0.76
	iCaRL [26]	49.02±3.20	88.99±2.13	7.53±0.79	28.19±1.47	-	-
	HAL [6]	32.36±2.70	82.51±3.20	-	-	74.15±1.65	84.02±0.98
	DER [4]	61.93±1.79	91.40±0.92	11.57±0.78	40.22±0.67	81.74±1.07	90.04±2.61
	SNCL (Ours)	**66.16±1.48**	**92.91±0.81**	**12.85±0.69**	**43.01±1.67**	**86.23±0.20**	**91.54±2.58**
500	ER [27]	57.74±0.27	93.61±0.27	9.99±0.29	48.64±0.46	80.60±0.86	88.91±1.44
	GEM [22]	26.20±1.26	92.16±0.69	-	-	76.88±0.52	81.15±1.98
	A-GEM [7]	22.67±0.57	89.48±1.45	8.06±0.04	25.33±0.49	67.56±1.28	80.31±6.29
	iCaRL [26]	47.55±3.95	88.22±2.62	9.38±1.53	31.55±3.27	-	-
	HAL [6]	41.79±4.46	84.54±2.36	-	-	80.13±0.49	85.00±0.96
	DER [4]	70.51±1.67	93.40±0.39	17.75±1.14	51.78±0.88	87.29±0.46	92.24±1.12
	SNCL (Ours)	**76.35±1.21**	**94.02±0.43**	**20.27±0.76**	**52.85±0.67**	**88.53±0.41**	**93.05±1.02**
5120	ER [27]	82.47±0.52	96.98±0.17	27.40±0.31	67.29±0.23	89.90±0.13	93.45±0.56
	GEM [22]	25.26±3.46	95.55±0.02	-	-	87.42±0.95	88.57±0.40
	A-GEM [7]	21.99±2.29	90.10±2.09	7.96±0.13	26.22±0.65	73.32±1.12	80.18±5.52
	iCaRL [26]	55.07±1.55	92.23±0.84	14.08±1.92	40.83±3.11	-	-
	HAL [6]	59.12±4.41	88.51±3.32	-	-	89.20±0.14	91.17±0.31
	DER [4]	83.81±0.33	95.43±0.33	36.73±0.64	69.50±0.26	91.66±0.11	94.14±0.31
	SNCL (Ours)	**90.41±0.46**	**97.11±0.19**	**39.83±0.52**	**70.52±0.37**	**92.93±0.11**	**94.83±0.34**

10 and Sequential Tiny ImageNet are employed as dataset for verifying *Class-IL*. For *Domain-IL*, we compare with other methods on Permuted MNIST and Rotated MNIST datasets. We use MNIST-360 dataset to test the performance on *GCL* setting.

Architectures. We use the same baseline network as in other existing methods. For CIFAR and Tiny ImageNet, we leverage a standard ResNet18 [13] without pretraining as the baseline architecture, similar to [26]. Following the exact settings in [4, 22, 27], we deploy a fully connected network with two hidden layers for variants of the MNIST dataset. All networks employ ReLU in the hidden layers and softmax with cross-entropy loss in the final layer.

Augmentation. Following [4], we adopt random crops and horizontal flips to examples from the current task and replay buffer. Note that data augmentation provides an implicit constraint to network.

Configuration and Hyperparameters. All models are trained using Stochastic Gradient Descent (SGD) optimizer. The number of epoch for MNIST-based setting is 1. On sequential CIFAR-10 and sequential Tiny ImageNet, we train the model for 50 and 100 epochs in each phase, respectively. Each batch consists of half the new task's samples and half samples from the replay buffer. The replay buffer is updated with the proposed LRS at the end of each batch.

4.1. Experimental Results

We compare the proposed SNCL with multiple baseline methods in the CL setup. Regularization-based methods: oEWC [31] and SI [39]; Knowledge distillation-based methods: iCaRL [26] and LwF [20]; Dynamic modular approach: PNN [29]; Rehearsal-based methods: ER [27], GEM [22], A-GEM [7], GSS [2], HAL [6], DER [4]. We include two non-continual learning baselines: SGD (lower bound) and JOINT (upper bound). All experiments are averaged over 10 runs using different initialization.

Three CL settings with task boundaries. Table 1 summarizes the average accuracy of all tasks after training. The proposed method significantly outperforms the other methods on the different settings. Regularization-based methods, oEWC and SI, obtain terrible results, which shows that regularization towards old sets of parameters is not an effective method to alleviate forgetting. The reason is that the weight importance is calculated in previous tasks, the importance will be changed with the following task.

Compared with replay-based methods, DER and the proposed SNCL significantly outperform the other methods, especially in the Domain-IL setting. For domain-IL set-

Table 2. Accuracy on the test set for MNIST-360.

JOINT	SGD	Buffer	ER [27]	MER [27]	A-GEM-R [7]	GSS [2]	DER [4]	SNCL (Ours)
82.98±3.24	19.09±0.69	200	49.27±2.25	48.58±1.07	28.34±2.24	43.92±2.43	55.22±1.67	**62.37±1.52**
		500	65.04±1.53	62.21±1.36	28.13±2.62	54.45±3.14	69.11±1.66	**74.92±1.73**
		1000	75.18±1.50	70.91±0.76	29.21±2.62	63.84±2.09	75.97±2.08	**78.86±1.25**

Table 3. Ablation studies of different components in SNCL.

Variants	S-CIFAR-10		S-Tiny-ImageNet		P-MNIST	R-MNIST
	Class-IL	Task-IL	Class-IL	Task-IL	Domain-IL	Domain-IL
Baseline	61.93	91.40	11.57	40.22	81.74	90.04
+VBS	64.87	91.65	11.97	40.93	84.52	90.38
+FER	65.85	92.13	12.24	42.23	85.53	90.68
+LRS	63.07	91.49	11.66	40.64	83.63	90.12
+VBS+FER	66.02	92.48	12.61	42.81	86.02	91.32
+VBS+LRS	64.93	91.73	12.05	41.33	84.94	90.40
+FER+LRS	65.91	92.19	12.54	42.63	85.61	90.76
SNCL	66.16	92.91	12.85	43.01	86.23	91.54

tings, a shift occurs within the input domain, but not within the classes: hence, the relations among them also likely persist. Other methods struggle for learning the similarity relations through the data-stream, and decrease the performance. The proposed method can obtain shareable and generalized knowledge from the data-stream and achieve the best performance. In Class-IL setting, we observe the proposed method obtains a significant improvement in terms of average accuracy. For the S-Tiny-ImageNet dataset, the proposed SNCL outperforms all the methods significantly with the small buffer size.

Backward Transfer (BWT) is an important metric to monitor catastrophic forgetting [24]. We also evaluate BWT of SNCL and compare it with other methods on P-MNIST, our method (BWT is -11.62) outperforms GEM (-30.36), HAL (-17.12), and DER (-13.20), showing less forgetting in SNCL.

GCL setting. In [4], MNIST-360 dataset is proposed to validate general continual learning setting, where the task boundaries are not available. We compare our method with ER, MER, GSS, and A-GEM-R. A-GEM-R is a variant of A-GEM with a reservoir replay buffer. The results are reported in Table 2, we observe that the proposed SNCL significantly outperforms the other methods on different buffer sizes. With the small buffer size, the performance of the proposed method improves 13% compared with DER. This result demonstrates that the proposed method can effectively prevent catastrophic forgetting.

4.2. Ablation Study

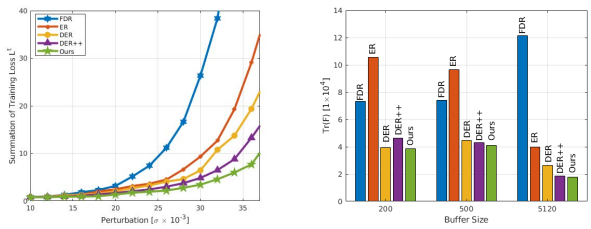
Bayesian sparse networks. To verify the impact of variational Bayesian sparsity in the SNCL, we conduct several different experiments. As reported in Table 3, DER is compared as the baseline. From Table 3, the results of the variant baseline+VBS significantly outperform baseline.

Table 4. Ablation studies w.r.t. different layers.

Variants	P-MNIST	R-MNIST
Baseline	81.74	90.04
FER-on-layer1	82.98	90.17
VBS + FER-on-layer1	83.41	90.61
FER-on-layer2	85.27	90.34
VBS + FER-on-layer2	85.73	90.93
FER-on-layer1&2	85.53	90.68
VBS + FER-on-layer1&2 (SNCL)	86.23	91.54

VBS can also further improve the performance of baseline+FER, which shows the effectiveness of the proposed network sparsity with VBS. We further study the effectiveness of VBS on different layers by operating only on each layer of the network. We report the results in in Table 4, from which the variants with VBS achieve higher numerical results. For example, compared with FER-on-layer1 (using FER on layer1 of baseline), its combination with VBS promotes performance. From Table 4, we can find that employing VBS on all layers obtains the best performance. In practice, if with memory restriction, we can use FER on fewer layers; Thus, we deem VBS restricts the sparsity of the network at an early stage and reserves capacity for the latter tasks.

Full experience replay. To verify the effectiveness of full experience replay (FER), we organize a series of experiences in Table 3 and 4. As shown in Table 3, compared with baseline, the variant baseline+FER has obvious improvement. Unlike previous methods that only replay samples and logits/labels, FER provides effective replay on features and can learn aligned knowledge at each layer. In addition, we also conduct ablation studies on different layer's features $h_{\theta,l,c}(x)$ as shown in Table 4, from which each layer with FER will promote the performance. It demonstrates that the stored intermediate features have contributions to



(a) Perturbation [↓] (b) Fisher Eigenvalues [↓]

Figure 3. Analyses of the robust and flat minima.

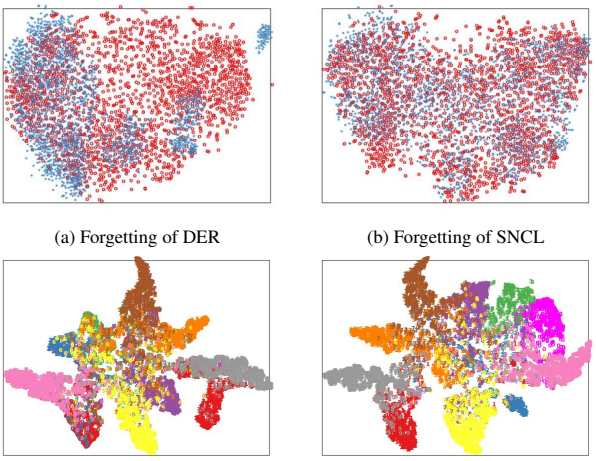
the results. We also observe that the impacts of high-level features (FER-on-layer2) are larger than that of low-level features (FER-on-layer1). In addition, employing FER on all layers will improve the results further, as reported in Table 4. This result attributes to the full experience replay strategy effectively providing more effective supervision on learning knowledge at different layers.

Loss-aware reservoir sampling. To evaluate the impact of LRS, we further conduct ablation studies. As shown in Table 3, the baseline with LRS brings improvement in performance. Compared with baseline+FER, the combination between FER and LRS achieves better results. The reason is that LRS maintains memory with more balanced distributions on different classes and training loss.

Generalization, robustness, and flat minima. Generalization and robustness are desirable properties for CL approaches. As discussed in [4, 16, 17], these properties link to the flatness of the attained minimum. If the model can converge to a flat minimum w.r.t. any task, the model tends to find the joint minimum of all tasks easily. It coincides with the motivation of our SNCL at this viewpoint of optimization geometry. We first measure the sensitivity of the model to the local perturbations [4] on learned parameters. As shown in Fig. 3 (a), the proposed model has a high tolerance towards the perturbations, which indicates that the model converges to the flat and robust minima. Following [4], we compute the empirical Fisher Information Matrix on all training data in S-CIFAR-10. Fig. 3 shows that our method produces the lowest sum eigenvalue of F, implying flatter minima achieved by SNCL.

4.3. Visualization Analysis

We visualizes features of each class after CL training via t-SNE [34] to show how the proposed SNCL performs intuitively. In Fig. 4 (a) and (b), we show that the proposed method suffers from less catastrophic forgetting. We visualize the learned presentation of task #3 after learning all 10 tasks in S-CIFAR-10 under the CL setting in blue color. We also directly train the network *only* on task #3 as the reference of no forgetting. By taking the no-forgetting results as a reference, Fig. 4 (a) shows the representations of DER drift severely after the CL process. On the contrary, in Fig. 4 (b), after learning other tasks, the representations on



(a) Forgetting of DER (b) Forgetting of SNCL

(c) Learned representation of DER (d) Learned representation of SNCL

Figure 4. Visualizing t-SNE embeddings on S-CIFAR-10. (a) and (b) analyze the forgetting issue. Blue points are from a task after learning other tasks in CL; Red points are obtained from a single-task no-forgetting training as reference. (c) and (d) show the representations of all the classes after CL.

task #3 (in blue) of SNCL keep similar to the no-forgetting reference, implying SNCL suffers less forgetting.

Fig. 4 (c) and (d) visualize the representations of all the classes (on the testing data) after the CL learning. The representations of SNCL on different classes (in Fig. 4 (d)) can be separated better than DER (in (c)), which shows that there are less interference and forgetting in SNCL. DER forgets more severely due to that the past tasks can be more easily interfered by the new tasks, such as the green and red points in Fig. 4 (c).

5. Conclusion

In this paper, we proposed to learn sparse networks for continual learning with variational Bayesian sparsity priors on the neurons (SNCL). The proposed method alleviates the catastrophic forgetting and interference by enforcing the sparsity of the network to reserve parameters for future tasks. We propose FER to store and replay with old samples' intermediate layer features generated when observed in the data stream. A loss-aware reservoir sampling strategy is proposed to maintain the memory. Extensive experimental analysis shows that the proposed SNCL framework outperforms the state-of-the-art method on several datasets under different settings. As a main *limitation*, SNCL does not model the more fine-grained sparsity in an element-wise way. And the sparsity structure correlation across the neurons is not considered. As with other existing CL methods, SNCL cannot guarantee to avoid forgetting in CL, stimulating further studies for real-world applications in the future. We will also study how to find the best balanced trade-off between memory usage and accuracy.

References

[1] Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3931–3940, 2020. 1

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. *arXiv preprint arXiv:1903.08671*, 2019. 6, 7

[3] Rahaf Aljundi, Marcus Rohrbach, and Tinne Tuytelaars. Selfless sequential learning. In *International Conference on Learning Representations*, 2019. 2

[4] 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*, pages 15920–15930, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[5] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and Simone Calderara. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2180–2187. IEEE, 2021. 5

[6] Arslan Chaudhry, Albert Gordo, Puneet Kumar Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. *arXiv preprint arXiv:2002.08165*, 2(7), 2020. 1, 6

[7] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018. 1, 3, 6, 7

[8] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet Kumar Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. Continual learning with tiny episodic memories. 2019. 1, 3, 5

[9] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019. 1, 3, 5

[10] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2, 5

[11] Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual learning via neural pruning. *arXiv preprint arXiv:1903.04476*, 2019. 2

[12] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819, 2021. 1

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1, 2

[15] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 2

[16] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016. 8

[17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 1, 2, 5, 8

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[19] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 5

[20] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1, 2, 5, 6

[21] Yuhang Liu, Wenyong Dong, Lei Zhang, Dong Gong, and Qinfeng Shi. Variational bayesian dropout with a hierarchical prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7124–7133, 2019. 4

[22] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30:6467–6476, 2017. 1, 3, 5, 6

[23] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 1

[24] Guy Oren and Lior Wolf. In defense of the learning without forgetting for task incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2209–2218, 2021. 1, 7

[25] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. itaml: An incremental task-agnostic meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13588–13597, 2020. 2

[26] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 1, 2, 5, 6

[27] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018. 6, 7

[28] Mark B Ring. Child: A first step towards continual learning. In *Learning to learn*, pages 261–292. Springer, 1998. 1

[29] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 1, 2, 6

[30] Gobinda Saha, Isha Garg, Aayush Ankit, and Kaushik Roy. Structured compression and sharing of representational space for continual learning. *arXiv e-prints*, pages arXiv–2001, 2020. 2

[31] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, pages 4528–4537. PMLR, 2018. 6

[32] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *arXiv preprint arXiv:1705.08690*, 2017. 2

[33] Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *European Conference on Computer Vision*, pages 254–270. Springer, 2020. 2

[34] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 8

[35] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985. 5

[36] Ju Xu and Zhanxing Zhu. Reinforced continual learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 907–916, 2018. 1, 2

[37] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017. 1

[38] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6982–6991, 2020. 2

[39] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017. 1, 2, 6

[40] Chen Zeno, Itay Golan, Elad Hoffer, and Daniel Soudry. Task agnostic continual learning using online variational bayes. *arXiv preprint arXiv:1803.10123*, 2018. 2