Biological Cybernetics (2023) 117:345–361
https://doi.org/10.1007/s00422-023-00973-w

Biological Cybernetics

ORIGINAL ARTICLE

Bio-inspired, task-free continual learning through activity regularization

Francesco Lässig[1] · Pau Vilimelis Aceituno[1] · Martino Sorbaro[1,2] · Benjamin F. Grewe[1,2]

Received: 30 November 2022 / Accepted: 6 August 2023 / Published online: 17 August 2023

Abstract

The ability to sequentially learn multiple tasks without forgetting is a key skill of biological brains, whereas it represents a major challenge to the field of deep learning. To avoid catastrophic forgetting, various continual learning (CL) approaches have been devised. However, these usually require discrete task boundaries. This requirement seems biologically implausible and often limits the application of CL methods in the real world where tasks are not always well defined. Here, we take inspiration from neuroscience, where sparse, non-overlapping neuronal representations have been suggested to prevent catastrophic forgetting. As in the brain, we argue that these sparse representations should be chosen on the basis of feed forward (stimulus-specific) as well as top-down (context-specific) information. To implement such selective sparsity, we use a bio-plausible form of hierarchical credit assignment known as Deep Feedback Control (DFC) and combine it with a winner-take-all sparsity mechanism. In addition to sparsity, we introduce lateral recurrent connections within each layer to further protect previously learned representations. We evaluate the new sparse-recurrent version of DFC on the split-MNIST computer vision benchmark and show that only the combination of sparsity and intra-layer recurrent connections improves CL performance with respect to standard backpropagation. Our method achieves similar performance to well-known CL methods, such as Elastic Weight Consolidation and Synaptic Intelligence, without requiring information about task boundaries. Overall, we showcase the idea of adopting computational principles from the brain to derive new, task-free learning algorithms for CL.

Keywords Continual learning · Bio-inspired · Sparsity · Feedback · Lateral inhibition · Activity regularization

1 Introduction

The mammalian brain has an astonishing ability to continually form new memories while preserving previous ones. In contrast, artificial neural networks are prone to *catastrophic forgetting* when trained on a sequence of tasks or datasets (McCloskey and Cohen 1989). This is true even if the tasks are very similar to each other and are likely to benefit from similar features. For example, learning to recognize different pairs of hand-written digits in sequence is notoriously difficult for artificial neural networks trained with backpropagation (Van de Ven and Tolias 2019).

For multi-layer artificial neural networks, a range of continual learning (CL) approaches have been devised that include modifications to the network architecture, loss function, or the implicit or explicit storage of previous task data (Van de Ven and Tolias 2019). Usually, these methods require external information about a task switch. This is in stark contrast to natural environments, where tasks are usually not well defined and need to be inferred from context.

To address the CL problem, brain-inspired approaches have been developed (Kudithipudi et al. 2022; Parisi et al. 2019). For example, French (1991) pointed out that the problem of catastrophic forgetting might not be intrinsic to biological neural networks, but is rather an effect of distributed and overlapping task representations that emerge when using the standard backpropagation (BP) algorithm. In line with this idea, it has been suggested that biological

Communicated by Benjamin Lindner.

This article is published as part of the Special Issue on 'What can Computer Vision learn from Visual Neuroscience?'.

✉ Francesco Lässig
flaessig@ethz.ch

Benjamin F. Grewe
grewe@ethz.ch

[1] Institute of Neuroinformatics University of Zürich and ETH, Zürich, Switzerland

[2] AI Center, ETH, Zürich, Switzerland

networks might avoid catastrophic forgetting by representing information through a sparse, but task-specific subset of neurons and synapses to which learning is restricted (Lin et al. 2014; Manneschi et al. 2021; French 1991). Other approaches relax the idea of restricting learning to sub-populations to the more general notion of learning within restricted subspaces (Duncker et al. 2020).

In this work, we exploit the idea of restricting learning to task-specific, sparse representations with the goal to derive a novel, bio-inspired task-free CL method. In line with the pervasive recurrence observed in the visual cortex (van Bergen and Kriegeskorte 2020), we argue that a task-specific sparsity mechanism should not only incorporate feedforward information (bottom-up) coming from lower hierarchical layers but also error feedback information coming from higher areas (top-down). To render both forms of information usable for such informed sparsity, we adopt Deep Feedback Control (DFC), a bio-plausible deep learning framework in which every neuron integrates inputs from the previous layer, as well as top-down error feedback during learning (Meulemans et al. 2022). To enforce sparsity, we combine DFC with a winner-take-all (WTA) mechanism and restrict learning of the feedforward weights to active neurons. To stabilize and protect previously learned representations, we further introduce intra-layer recurrent weights that are updated through a Hebbian-type learning rule. In the following, we term this new, combined method *sparse-recurrent* DFC.

To explain the basics of our algorithm, we first present related work in Sect. 2. Then, in Sect. 3, we provide implementation details on how we modified the DFC learning dynamics to integrate the two major factors required for CL—sparsity and intra-layer recurrent connections. In Sect. 4, we show that the introduction of these additional bio-plausible elements helps to stabilize learning and to reduce forgetting by regularizing neural activity. We compare our approach with other established regularization-based CL methods and show that sparse-recurrent DFC performs comparably well despite completely lacking information on task boundaries. Finally, we analyse the resulting task representations in order to better understand the mechanisms behind the observed improvement in CL performance.

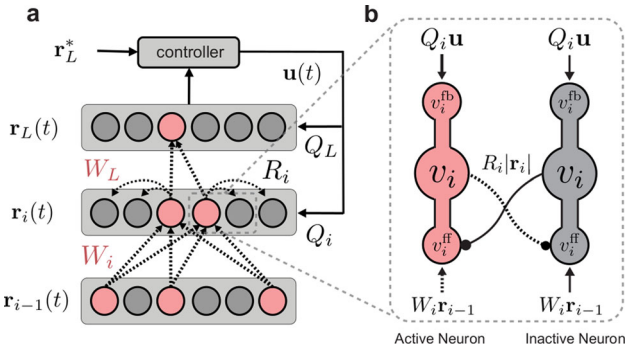

Fig. 1 **a** Schematic of the sparse-recurrent DFC network and its top-down feedback controller. The $r_i(t)$ values denote neuron activation vectors for layer i, whereas r_L^* represents the desired network output. Learning is based on a dynamic process during which neurons integrate feedforward and feedback signals until the network converges to a sparse target representation minimizing the loss. Weight updates (dashed lines) of forward weights W_i are restricted to neurons that are active at convergence (red). Lateral recurrent weights R_i into inactive neurons are updated via a Hebbian-like learning rule. The Q_i values denote feedback weights, and $u(t)$ refers to the control signal. **b** Detailed zoom into layer i showing one active (pink) and one suppressed (grey) neuron. v_i^{ff}, v_i^{fb}, and v_i represent feedforward, feedback and combined activity, respectively. The solid lines represent weights that will not be changed, whereas dashed lines show weights which will be updated

2 Background

2.1 Computational strategies for continual learning

To overcome catastrophic forgetting, researchers developed a variety of different strategies that can roughly be classified into three categories:

(1) *Replay methods* rely on implicitly or explicitly storing and revisiting previous data while learning new tasks. This can be accomplished by storing small subsets of previously seen data in a memory buffer, or by training a generative model (Shin et al. 2017). However, we do not consider data replay in this work, since we are interested in methods based on bio-plausible plasticity, without relying on external data storage.

(2) *Regularization methods* constrain learning to preserve parameters that are important for previous tasks, usually by adding specialized loss terms. Elastic Weight Consolidation (EWC) and Synaptic Intelligence (SI) are commonly used representatives of this family, which we adopt as comparison benchmarks. In EWC (Kirkpatrick et al. 2017), after the network converges on a task, the Fisher information of the first task's loss is computed through a sampling mechanism. The Fisher term contains information on parameter importance relative to the first loss and is added as a regularization term to the loss for the following task. Synaptic Intelligence (Zenke et al. 2017) works through a similar mechanism, but parameter importance is estimated online based on how much of the decrease in loss can be attributed to the variation of each given parameter. In both cases, the regularization term is added to the loss at the end of each task, and information on task boundaries is therefore required.

(3) *Architectural methods* are based on structural changes such as freezing weights, or adding and removing neurons (Rusu et al. 2016). Alternatively, neurons can be dynamically gated based on context (Masse et al. 2018; von Oswald et al. 2020). Context, however, is usually

externally provided rather than inferred by the network itself, which is a strong assumption that may not always hold for real-world scenarios. In another approach, a dedicated system, inspired by the role of the prefrontal cortex, is used to detect contextual information instead (Zeng et al. 2019). In this work, we adopt a similar gating-based approach, in which, conversely, gating is provided by recurrent activity independently of external task information.

2.2 Continual learning in the brain

Although CL in the brain is not well understood, it is likely that various mechanisms are at play simultaneously, with some being loosely connected to the three CL strategies described above (Kudithipudi et al. 2022).

In neuroscience, the trade-off between fast learning and slow forgetting is known as the stability-plasticity dilemma. To avoid this issue, the interaction between a more plastic system, the hippocampus, and a more stable system, the neocortex, has been suggested as a long-term memory storage mechanism, akin to a data replay strategy (van de Ven et al. 2020). On the other hand, biological networks might control the stability/plasticity of individual synapses through mechanisms collectively referred to as *metaplasticity*. Through metaplasticity, synapses that are particularly important for solving previously learned tasks are left unaltered when learning new tasks, while less relevant synapses are made available to store new information, analogously to certain regularization-based approaches in CL (Jedlicka et al. 2022).

Next, *neurogenesis*, the birth of new neurons, is sometimes considered equivalent to architectural approaches that gradually grow the network. However, neurogenesis is believed to be limited to very specific brain areas, with small numbers of new neurons, and it is unclear whether it occurs in adult humans. It is therefore contested whether neurogenesis plays a role in CL (Parisi et al. 2018).

Finally, animal brains heavily rely on *context* to flexibly switch between tasks and to direct learning to task-specific neurons and synapses. For example, previous studies have shown that afferents of the olfactory nucleus in rats provide contextual input from other brain areas, thereby enabling dynamic and flexible task learning (Levinson et al. 2020). This not only enables context-specific gating of neuronal responses to the same stimulus for different environments or tasks but it also facilitates forward-generalization. Similarly, the release of specific neuromodulators (e.g. dopamine) has been linked to the gating of activity and to learning based on context (Kudithipudi et al. 2022). Overall, it is likely that in biological networks the modulation of neuronal activities, either through hierarchical top-down feedback or specific neuromodulators, directs learning to the most salient aspects of the task, while protecting older memories that are irrelevant in the current context.

2.3 Task-free continual learning

Van de Ven and Tolias (2019) defined three CL scenarios for which training is organized sequentially on each task and performance is evaluated as the average accuracy on all previously learned tasks:

(1) in *task-incremental learning* (task-IL), the task ID is available during training and at test time;
(2) in *domain-IL*, the task ID is available during training but not at test time;
(3) in *class-IL*, the task ID is available during training, but at test time the model must report the task ID alongside solving the task.

In all these scenarios, however, information on *task boundaries* is provided during training, i.e. the model knows when training on one task i ends and training on a new task $i+1$ begins. Most CL strategies need this information to update the loss or the network structure in preparation for the new task. However, such discrete changes in the loss or network structure do not seem biologically plausible. Therefore, in this paper, we focus on domain-IL, and on the more challenging class-IL, but in a setting where task information is entirely omitted during both training testing.

This so-called *task-free* form of continual learning is generally less studied, although a few examples have appeared in recent years. The majority of these follow a data storage and replay paradigm (Aljundi et al. 2019b; Wang et al. 2022; Rao et al. 2019), which we do not consider in this work. Lee et al. (2020) adopt an architectural approach, based on an expanding set of experts which, in turn, deal with new tasks. Among regularization-based methods, Laborieux et al. (2021) propose a metaplasticity-inspired mechanism, but so far limited to feedforward, binary networks. Aljundi et al. (2019a) circumvent the problem of task boundaries by heuristically detecting plateaus in the evolution of the loss, which signal the end of learning for a task, and use a mixed replay and regularization strategy. Finally, Pourcel et al. (2022) mix an architectural method with replay using a dynamic content-addressable memory for online class-IL.

To clarify how our method fits into this landscape of brain-inspired algorithms, we next provide details on our CL approach, which combines DFC, sparsity, and recurrent Hebbian-like connections.

3 Activity regularization through sparsity and recurrent gating

3.1 Deep feedback control

During training, the neuronal dynamics within the DFC network (Meulemans et al. 2022) can be described by a differential equation that takes into account the feedforward inputs v_i^{ff} as well as the feedback control signal v_i^{fb} according to

$$\begin{aligned}\tau_v \dot{v}_i(t) &= -v_i(t) + v_i^{\text{ff}}(t) + v_i^{\text{fb}}(t) \\ &= -v_i(t) + W_i\phi\big(v_{i-1}(t)\big) + Q_i u(t)\end{aligned} \tag{1}$$

where the pre-nonlinearity neuron activations in layer i at time t are denoted by $v_i(t)$, and the incoming weights by W_i. ϕ refers to the activation function, while the neuron output is given by $r_i = \phi(v_i(t))$. The feedback signal $u(t)$ is calculated by summing the integral and proportional parts of the network output error $e(t)$ as described by Meulemans et al. (2022). $u(t)$ is then fed back to each neuron of the network via the feedback weights Q_i. During learning, the feedforward network and the feedback controller constitute a recurrent dynamical system that converges to a stable state (ss) at which the neuron activity $v_{i,\text{ss}}$ minimizes the output error and stabilizes the feedback signal $u(t)$. In practice, we simulate the dynamics for a set number of iterations and utilize the final activations as stable state values. The number of iterations is chosen to be high enough such that most simulations converge.

The final neuron activations $r_{i,\text{ss}} = \phi(v_{i,\text{ss}})$ are referred to as 'targets' or 'target activations' since they represent the values we want the network to produce without feedback. To achieve this, the forward weights are learned by comparing each neuron's target activation $r_{i,\text{ss}}$ to its feedforward-driven activation $\phi(v_{i,\text{ss}}^{\text{ff}})$ upon converging to the stable state:

$$\Delta W_i = \eta(r_{i,\text{ss}} - \phi(v_{i,\text{ss}}^{\text{ff}}))r_{i-1,\text{ss}}^T \tag{2}$$

where $r_{i-1,\text{ss}}$ is the presynaptic, post-nonlinearity activity with controller feedback, $r_{i,\text{ss}}$ is the activity of the neuron with feedback and $\phi(v_{i,\text{ss}}^{\text{ff}})$ is the postsynaptic neuron activity without feedback. In sparse-recurrent DFC, we additionally centre each weight update to have zero mean before applying it. This is done in order to prevent a small group of neurons to be more excitable and dominate the winner-take-all mechanism described in the next subsection. The feedback weights Q_i can be learned (Meulemans et al. 2021, 2022), but we simplify the learning of the feedback pathway and re-initialize Q_i as the Jacobian of the loss with respect to the neuron activations for every data point.

The update rule from Eq. 2 implements a learning paradigm where weight updates are determined by neural activity. This opens the possibility of regularizing weight updates indirectly by modulating neural activity. We will refer to this strategy as activity regularization. In the next sections, we describe how activity regularization (e.g. sparsity and recurrent gating) can be utilized to reduce interfering weight updates between representations of different inputs belonging to different tasks.

3.2 Dynamic sparsity

To gradually modulate the network activations towards sparse, non-overlapping representations, we add a winner-take-all mechanism on top of the existing DFC network. At each time step t, we set a fraction $s_i(t)$ of neurons to be zero. $s_i(t)$ is initialized to zero at $t = 0$ and incrementally grows over time until it reaches the desired sparsity for the stable state $s_{i,\text{ss}}$, which is a hyperparameter fixed for each layer i. We refer to these hyperparameters as sparsity levels. As long as different inputs to the network lead to sufficiently different activation profiles, this technique should lead to a reduction in overlap between active populations pertaining to different data points. As a result, interference during learning should be reduced by only updating the weights of active populations.

However, the network cannot learn to suppress specific neurons because forward connections to inactivated neurons are frozen. This is an issue because, while we aim to decrease overlap between representations of different classes, inputs belonging to the same class should be represented similarly. WTA sparsity based on feedforward and feedback activity alone does not ensure this. Our intuition is that, if neurons keep dropping in and out of active populations during training, no consistent representations can be learned, leading to forgetting. To address this problem, we introduce an additional set of connections with the aim of learning which neurons are allowed to fire together, and which neurons are mutually exclusive. This way, we provide a way for the network to stabilize and protect the neuron populations that together constitute specific representations.

3.3 Gating neuron activity through lateral recurrent connections

We stabilize neuron populations involved in learned representations by introducing lateral recurrent connections. Because we want to strongly influence which neurons are active, we implement lateral connections with a gating effect that multiplies activations by a factor between 0 and 1, similar to 'forget' gates used in LSTMs (Hochreiter and Schmidhuber 1997). We then calculate the neuron feedforward activity before the nonlinearity as

$$v_i^{\text{ff}}(t) = W_i\phi\left(v_{i-1}(t)\right) \odot \sigma\left(R_i|r_i(t)|\right) \tag{3}$$

where R_i refers to the recurrent weight matrix in the i-th layer, σ to the sigmoid function, and ϕ to the same activation function as used in Eq. 1. After applying the effect of the recurrent gating, we re-scale the population activity to have the same overall magnitude as before applying the gating. We thus only change the distribution, but not the total level of activity. At convergence, we learn the recurrent gating weights according to a rule inspired by the feedforward updates from Eq. 2

$$\Delta R_i = \eta(|r_{i,\text{ss}}| - |\phi(v^{\text{ff}}_{i,\text{ss}})|)|r_{i,\text{ss}}|^T \tag{4}$$

where $r_{i,\text{ss}}$ are the target activations of the presynaptic neurons in the same layer. Because our multiplicative gating mechanism affects the magnitude, but not the sign of the activity, we render this inhibition to depend on the magnitude of presynaptic activity. We therefore use absolute values of activity in both the dynamics (Eq. 3) and the update rule (Eq. 4). Like forward weight updates, we normalize recurrent weight updates to zero mean. In contrast to the feedforward weights, however, we only update incoming weights of inactivated neurons (i.e. neurons with activity set to zero by the winner-take-all sparsity mechanism). This lets us simplify the above equation to a Hebbian-like update rule for suppressed neurons:

$$\Delta R_i = -\eta|\phi(v^{\text{ff}}_{i,\text{ss}})||r_{i,\text{ss}}|^T. \tag{5}$$

As a result, we only update incoming recurrent weights for inactive neurons within the target representation, while for active neurons, we only update the incoming feedforward weights. Figure 1 (dashed lines) summarizes the weight updates. As in standard DFC, we use a simple feedforward pass during test time, for which neither top-down feedback nor lateral recurrent effects are taken into account. Therefore, the number of parameters of the trained model is equivalent to a conventional feedforward network with the same number of neurons (see "Appendix A.3" for a further discussion on model complexity).

Please note that gating through lateral connections, while crucially influencing the WTA selection of the active neuron population by modulating neuron activity, does not determine the level of sparsity. WTA sparsity and lateral connections are interconnected, but distinct mechanisms.

4 Experiments

To test the CL capabilities of our approach, we train sparse-recurrent DFC on the split-MNIST dataset, according to the domain-IL and class-IL paradigms (Van de Ven and Tolias 2019). Split-MNIST is a simple computer vision CL benchmark in which five pairs of consecutive digits are presented as a sequence of individual supervised learning tasks. In domain-IL, all tasks involve predicting the parity (even/odd) of the input digit, meaning that the output labels stay the same across tasks, but the input data changes. In class-IL, a different class has to be predicted for every digit, so that, across tasks, both the input digits and the class labels change.

4.1 Performance

To establish whether sparse-recurrent DFC actually succeeds at CL, we compare its performance against other learning algorithms, namely Synaptic Intelligence (SI), Elastic Weight Consolidation (EWC), as well as standard BP as baseline. Previous studies evaluated models at a fixed learning rate (LR) for a fixed number of epochs (Kirkpatrick et al. 2017; Van de Ven and Tolias 2019), however, we consider this problematic. Both the LR and the number of epochs can be seen as indicators for how much a network learns, thus pointing to an inherent trade-off between learning the current task well and forgetting previous tasks. Less learning generally leads to less forgetting, while at the same time not allowing the training to converge on the current task. Comparing CL algorithms at a single LR for a fixed number of training samples is problematic for two reasons. First, it does not account for different (model-specific) optimal amounts of training. Second, it fails to capture how robust a CL approach is to more learning, beyond its optimum LR and number of training samples per task. To overcome this issue, we evaluate learning algorithms in two different scenarios. In the first scenario we fix the number of epochs and vary the LR. In the second scenario we fix the LR and vary the training accuracy that we expect on the current task, before training on the next task, which results in different numbers of batches trained on for different models on different tasks. In both scenarios, we cover a wide spectrum between minimizing forgetting, and optimizing the current task.

4.1.1 Learning rate performance evaluation

Figure 2a and 2b shows performance for a fixed number of training samples across a range of LRs for domain-IL and class-IL, respectively. The initial rise of performance followed by a decay can be explained by the fact that very small LRs (left of the peak) generally prevent sufficient learning while high LRs (right of peak) lead to catastrophic forgetting. These CL performance profiles confirm our initial intuition that choosing a single LR to compare CL methods might lead to overestimating one method over another. We regard good performance in this setting as a function of both peak accuracy and the degree to which accuracy can be maintained once the optimal LR is reached. In domain-IL, sparse-recurrent DFC significantly outperforms BP and achieves a similar performance profile to EWC. Compared to SI, our approach

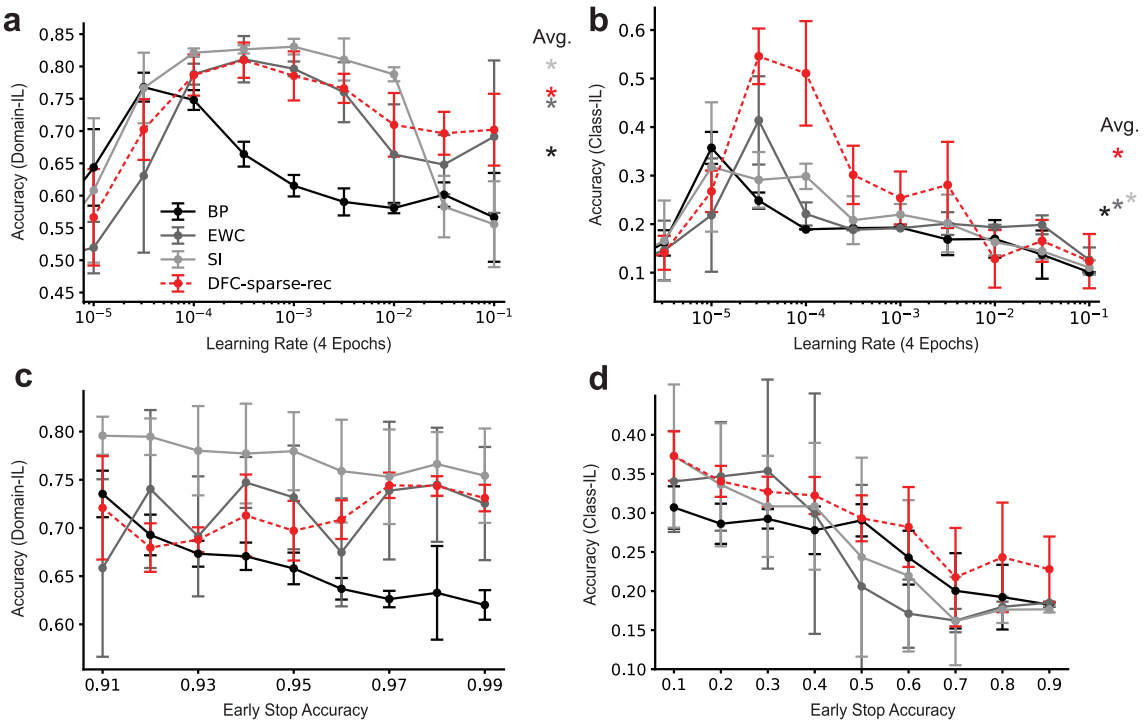

Fig. 2 Performance evaluation of split-MNIST for BP, EWC, SI, and DFC-sparse-rec for domain-IL (left column) and class-IL (right column). Error bars represent standard deviations using five random seeds. **a** Split-MNIST accuracy at the end of training in the domain-IL paradigm on the whole test set (all digits) for a range of learning rates (LRs). The number of training iterations is fixed at four epochs. Stars indicate average performance on an accuracy-maximizing window of six LRs. **b** Accuracy of models at the end of training in the class-IL paradigm on the whole test set for every LR. Stars indicate average performance on an accuracy-maximizing window of six LRs. **c** Accuracy of models at the end of training in the domain-IL paradigm on the whole test set for a range of minimum early stop accuracies. The LR is fixed, and training is stopped at every task once the train accuracy for the current batch reaches the given minimum accuracy value. The maximal number of epochs trained for is 10. **d** Accuracy of models at end of training in the class-IL paradigm on the whole test set for a range of minimum early stop accuracies

performs worse in terms of peak accuracy, but maintains accuracy over 70% for a wider range of LRs. In class-IL, sparse-recurrent DFC outperforms all other methods both in peak accuracy and average accuracy.

4.1.2 Early stop performance evaluation

Figures 2c and 2d show performance for a fixed LR across a range of early stop accuracies for domain-IL and class-IL, respectively. In domain-IL, sparse-recurrent DFC outperforms BP for almost all minimum accuracies. However, it is most competitive when we train each task to convergence. For training up to very high accuracies, sparse-recurrent DFC is comparable to both EWC and SI. In class-IL sparse-recurrent DFC outperforms all other CL algorithms for the majority of accuracies.

Overall, we conclude that sparse-recurrent DFC represents a competitive CL method that shows a robust performance independent of the amount of learning on each individual task. In the next section, we investigate in more detail the effects on accuracy with respect to the main components of our method: feedback, sparsity and intra-layer recurrency.

4.2 Integrating feedback signals facilitates CL

A major difference between standard BP and DFC is that in DFC, the activity of each neuron during training reflects feedforward as well as feedback (error) signals coming from the top-down controller. As a result, target representations $r_{i,\mathrm{ss}}$ are specific to both input and output, with data points exhibiting larger overlaps in active neuron populations if these have similar features *or* the same label. Figure 3a shows that CL performance is improved across a wide range of LRs if we take into account feedback signals when selecting the remaining active population within the sparse target. Although the combination of equal parts feedforward and feedback activity yields the best results overall, feedback activity alone achieves high accuracy for $LR = 1e-3.5$. We hypothesize that low LRs lead to less training of forward weights, rendering input selectivity less useful. Thus, it may be beneficial for the network to rely solely on feedback when determining the

active population. This is consistent with our idea that incorporating feedback signals generally facilitates the sparsity selection process, allowing the learning of more task-specific representations.

4.3 Sparsity and recurrent gating are required for CL

We next investigate whether both sparsity and intra-layer recurrence in the DFC framework are crucial for CL. We compare the accuracy of sparse-recurrent DFC against standard DFC, sparse DFC and recurrent DFC. As opposed to sparse-recurrent DFC, recurrent DFC has no inactivated neurons to constrain the recurrent weight updates to. We thus apply the recurrent weight update rule from Eq. 4 to all neurons. Figure 3b shows that neither sparsity nor recurrent gating alone significantly alters CL performance across LRs. However, the combination of the two leads to better performance across a wide range of LRs.

Figure 3c shows accuracy as a function of the sparsity parameters $s_{i,\mathrm{ss}}$. For the first hidden layer, a small but nonzero sparsity level yields the best performance, while for the second hidden layer, higher sparsity levels work best. This dependence on layer depth is expected, because the early layers of multi-layer neural networks encode low-level features common to multiple classes and class-selectivity is a disadvantage for these neurons (Morcos et al. 2018), while the later layers encode higher-level features which are more specific to individual classes (Zeiler and Fergus 2014; Mahendran and Vedaldi 2016).

4.4 Aligning sparse, separated representations across tasks facilitates domain-IL

Next, we test whether the combination of sparsity and recurrent gating facilitates CL by reducing representational overlap, in a domain-IL setting. We compute the reduction in overlap (i.e. separation) between last hidden layer representations of all pairs of digits, at the end of training. We distinguish between intra-label separation (MNIST digits with the same parity label) and inter-label separation (digits with different parity labels), as shown in Fig. 3f. We compute representational separation between digits as

$$s(d_1, d_2) = 1 - \frac{a_l^{d_1} \cdot a_l^{d_2}}{\|a_l^{d_1}\| \|a_l^{d_2}\|}; \qquad a_l^d = \sum_{j=1}^{n} |r_{l,j}^d| \tag{6}$$

where $r_{l,j}^d$ represents the activations in layer l elicited by the j'th sample of digit d. Figure 3d shows the averages of inter- and intra-label representational separations for DFC variants. Interestingly, sparse DFC shows high representational separation, but does not yield significantly higher accuracies compared to standard DFC or BP. This suggests that overall increases in representational separation alone do not account for performance improvements that we observe in Fig. 3b.

To better understand this result, we next devise a new measure of separation, which we term normalized inter-label separation and that is defined as the average difference between inter-label separation and intra-label separation. Figure 3e shows this separation metric over a wide range of LRs. For the LRs where sparse-recurrent DFC yields higher normalized inter-label separation, we also observe better CL performance (compare to Fig. 3b), suggesting that the relative degree of digit representational overlap can explain the CL performance profile that we observe for sparse-recurrent DFC. This indicates that sparse-recurrent DFC facilitates domain-IL performance by representing even and odd digits in two partially separated neuron populations that are reused across tasks. As a first result, we conclude that, although sparsity is necessary to create non-overlapping representations, sparsity alone is not sufficient for aligning these across tasks. Such alignment, however, seems beneficial for domain-IL, where several digits are represented by the same label. We next investigate how recurrent gating helps to learn representations that are compatible across tasks.

The final hidden layer of a network has to learn representations of the input that are linearly separable by its readout weights. One possible way to prevent catastrophic forgetting is to ensure two things. **Condition 1**: The hyperplane separating representations of different labels (implemented in the network by the readout layer) needs to stay the same, or similar across tasks. **Condition 2**: Data points represented in the final hidden layer need to stay on the same side of the classification hyperplane that was initially learned as we train on subsequent tasks. We measure feedforward activations $\phi(v_{L-1}^{\mathrm{ff}})$ (no recurrent gating) and target activations $r_{L,\mathrm{ss}}$ (including effects of controller and recurrent gating) to test whether recurrent gating helps to achieve this. Regarding condition 1, Fig. 4b shows that, if we classify target activations at training onset of a new task according to the previously learned separation boundary, sparse-recurrent DFC consistently yields higher classification accuracies than sparse DFC. This suggests that lateral connections regularize new target activations such that they better align with previously learned task boundaries. This idea is illustrated in Fig. 4a, showing that target activations of the second task are separated by the same hyperplane that divides targets of the first task. Regarding condition 2, we measure the direction of movement of feedforward activations from the beginning to the end of training. We next quantify how much the data points move towards the initially learned separation boundary. Figure 4c suggests that sparse-recurrent DFC reduces the movement towards the previous decision boundary compared to sparse DFC. Taken together, our results suggest that recurrent gating helps fulfil both conditions. For more details

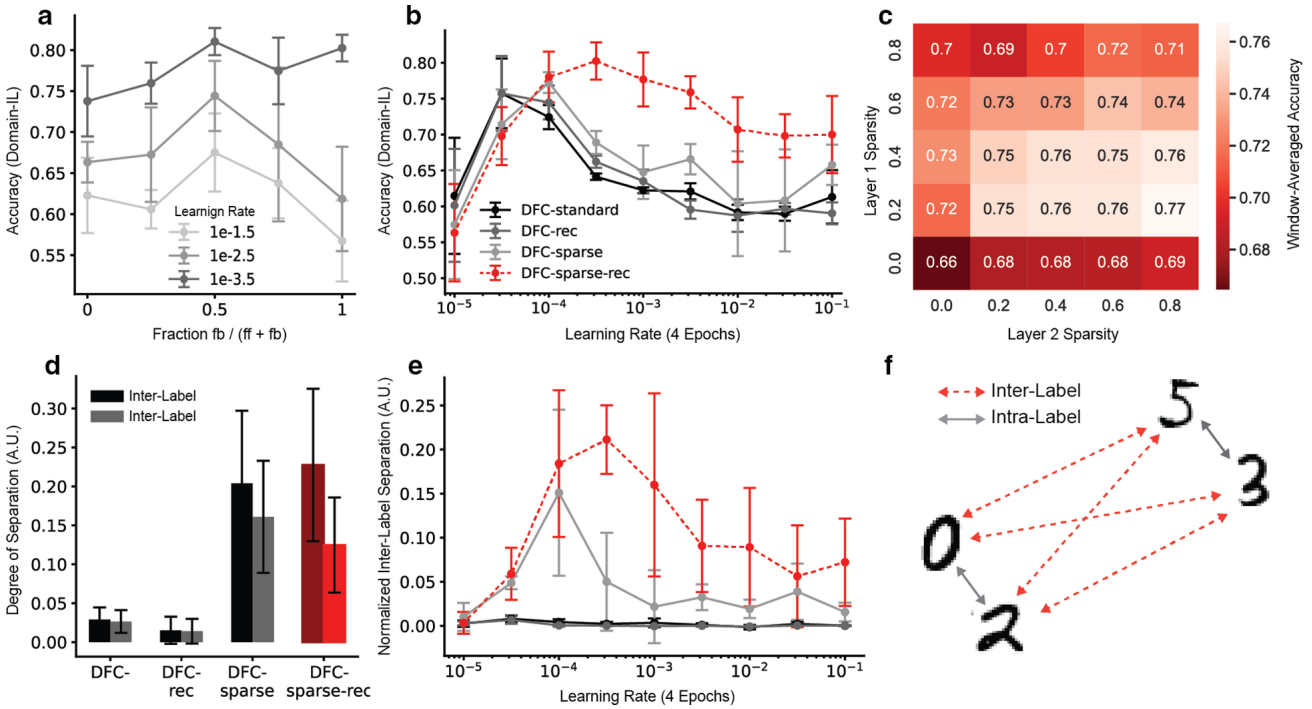

Fig. 3 Necessity of sparse-recurrent DFC components and activity separation analysis for domain-IL. Error bars represent standard deviations using five random seeds. **a** Split-MNIST performance when using varying ratios of feedforward and feedback activity to select the suppressed population for different learning rates (LRs). The x-axis represents the fraction of feedback activity used for the selection of neurons to be suppressed. A value of 0 means only feedforward activity (ff) is considered, a value of 1 means only feedback (fb) is taken into consideration, and 0.5 corresponds to an equal mix of the two activities. This activity mix is only used for selecting the active neuron population, but the activity flowing through the neurons corresponds to the normal network activity given by Eq. 1. **b** Cross-LR evaluation for all DFC variants. The plot reflects the overall performance on all split-MNIST digits at the end of training. **c** Cross-LR accuracy for different combinations of hidden layer sparsity levels. The accuracies were aggregated to single numbers by averaging over a contiguous window of six LRs that maximizes average performance, and over five random seeds. **d** Inter- and intra-label separations for DFC variants after all five tasks have been learned. Intra-label separations are calculated for all digit pairs with the same label, inter-label separations for all pairs of digits with different labels. Results are averaged over nine LR values. **e** Normalized inter-label separation calculated as the difference between inter-label separation and intra-label separation at the end of training across a range of LRs. **f** Visualization of intra- and inter-label distances in the space of activity separation

on the calculation of these metrics involving hyperplanes, see "Appendix C".

4.5 Learning within separate subspaces facilitates class-IL

One possible strategy to address class-IL is to enforce sparse, non-overlapping representations of different digits, thereby preventing interfering weight updates between classes. To test whether sparse-recurrent DFC utilizes this strategy, we record target activities of different digits after they are first learned and measure the representational overlap of all pairs of digits using Eq. 6. Figure 5a shows that, while sparse DFC leads to some increase in representational separation, sparse-recurrent DFC maximizes separation across all LRs compared to other DFC variants. These results are consistent with our initial idea of reduced representational overlap facilitating CL. Intuitively, if different neurons are used for different tasks, weights of neurons that were important in early tasks are less likely to be changed. Similar to domain-IL, sparsity in class-IL can thus be seen as a necessary condition for the formation of non-overlapping representations.

To gain a better understanding of why recurrent gating helps to increase representational separation in class-IL, we next analyse its effect on altering the dimensionality of targets. Figure 5b shows the effective dimensionality (Roy and Vetterli 2007) of the target activations of different tasks after learning for recurrent DFC, sparse DFC and sparse-recurrent DFC. The results suggest that the combination of sparsity and recurrent gating leads to a significant decrease in effective dimensionality of the target activations. This led us to hypothesize that representations learned for a new task are less likely to affect dimensions that were important for previous tasks. To investigate if recurrent gating leads to a reduction in reuse of previously learned subspaces, we compute the fraction of

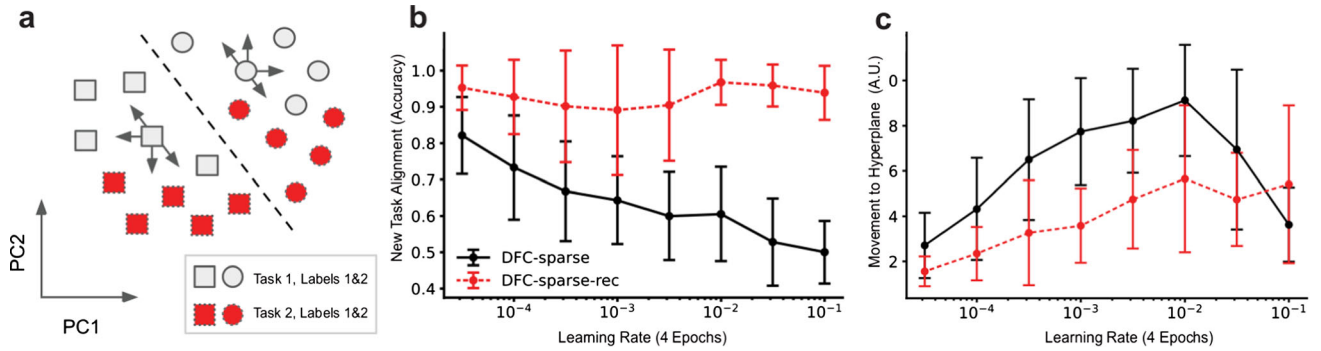

Fig. 4 Effects of recurrent gating on last hidden layer targets $r_{L-1,\text{ss}}$ and feedforward activations $\phi(v^{\text{ff}}_{L-1})$ during learning. Error bars represent standard deviations using five random seeds. **a** Schematic of task 1 and 2 representations with respect to the hyperplane (dashed line) dividing task 1 target activations (grey) according to their label. This diagram illustrates two things: First, the new target representations align with the previously learned hyperplane in terms of label separation (supported by b). In other words, the hyperplane that separates task 1 targets also separates task 2 targets. Second, task 1 representations generally move less towards the separating hyperplane as subsequent tasks are learned (supported by c). This is represented by the arrows. **b** Alignment of new task target activations with previous hyperplanes. This is measured as the fraction of initial target representations ($r_{L-1,\text{ss}}$, before learning task i), of the new task i that are correctly separated according to the hyperplane learned on the previous task $i-1$. **c** Movement of feedforward activations $\phi(v^{\text{ff}}_{L-1})$ of previous tasks towards the hyperplane after learning subsequent tasks, normalized by movement in all directions

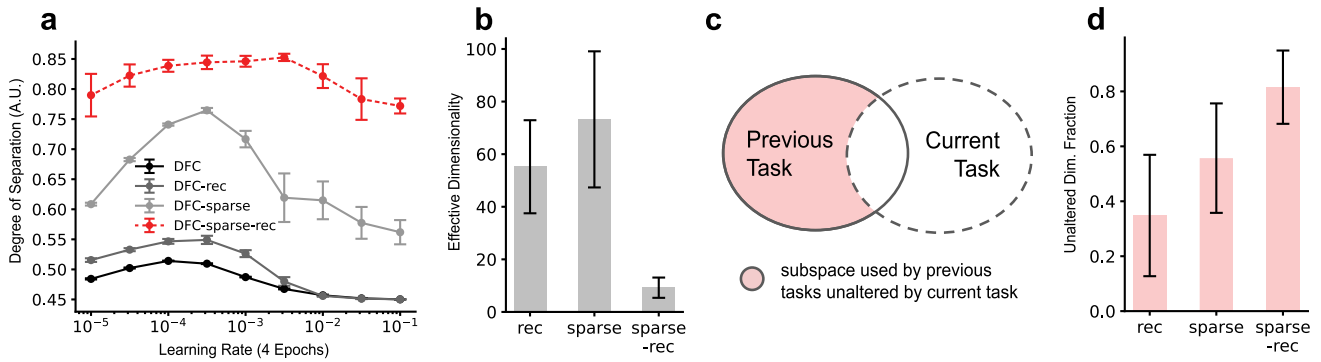

Fig. 5 Last hidden layer target activation ($r_{L-1,\text{ss}}$ belonging to task i, after learning task i) analysis for class-IL. Error bars represent standard deviations over five random seeds. **a** Representational separation (Eq. 6) between pairs of digits for DFC variants for a range of learning rates (LRs). **b** Effective dimensionality (Roy and Vetterli 2007) of targets averaged over tasks and random seeds for DFC variants for LR $= 0.001$. **c** Visualization of the 'unaltered dimensionality fraction' γ measure described in "Appendix D". The left and right ellipses represent the subspace used by the first $i-1$ tasks, and by task i, respectively. γ quantifies the dimensionality of the coloured area as a fraction of the dimensionality of the area of the left ellipse. **d** Unaltered dimensionality fraction γ (described in Eq. 18 from "Appendix D" and visualized in subplot c) for DFC variants

the effective dimensionality used by previous tasks that is left unaltered by the current task (Fig. 5c). For more details on the calculation of this metric, see 'Appendix D". Figure 5d validates our hypothesis that recurrent gating reduces the fraction of dimensions that are altered by new tasks, thus reducing the extent to which new weight updates interfere with parameters important for previous tasks.

5 Discussion

In summary, we have presented a new, bio-inspired, task-free CL approach that yields competitive performance compared to other CL methods on a simple computer vision benchmark. To restrict learning to a reduced set of task-specific parameters, our method (sparse-recurrent DFC) integrates feedforward and feedback information to constrain activity to a sub-population of neurons. In addition to being more biologically plausible, we show that including top-down signals is beneficial for CL. Our results are consistent with the idea that sparsity is a requirement for reducing representational overlap, but suggest that sparsity alone is insufficient for protecting previously learned model parameters. We show that intra-layer recurrent connections, when combined with sparsity, facilitate the protection of old task representations, leading to competitive CL performance of

DFC on split-MNIST. For both domain- and class-IL, recurrent gating in combination with sparsity restricts learning to low-dimensional subspaces. In domain-IL, the same subspace consisting of two separated neuron populations is shared across tasks; in class-IL learning is restricted to multiple distinct subspaces.

From a neuroscience perspective, our findings might allow experimental researchers to derive new hypotheses about how the brain minimizes catastrophic forgetting. One prediction of our sparse-recurrent DFC network is that intra-layer recurrent connections are only critical during learning but not inference, since we only use recurrence at training time. Although this is surprising, there are data suggesting that biological brains do this as well. Van Rullen et al. (1998) argue that, given the short response time in face recognition tasks, neurons do not have the time to emit much more than one spike at each processing stage. This would imply that initial inference can happen before recurrence takes effect. Based on our work, neuroscientists could, for example, manipulate recurrent communication within cortical hierarchies, to test if an animal's ability to perform inference or to learn multiple tasks sequentially is affected.

From a machine learning perspective, our new method is relevant because it is based on a novel set of working principles to achieve CL. As sparse-recurrent DFC naturally infers non-overlapping representations and thus non-interfering parameter updates, it does not require any task boundaries or task information either during training or testing. While other task-free CL methods exist and achieve competitive performance, they are not exclusively based on specialized weight update rules, as they use either data replay or expanding architectures. The only exception we could find is limited to binary networks (Laborieux et al. 2021). Moreover, in future work, our approach could be combined with other task-free CL methods (replay and non-replay-based) which might lead to even better CL performances. Although the current implementation of sparse-recurrent DFC is computationally less efficient when compared to standard CL algorithms running on GPUs, DFC is ideally suited for a neuromorphic hardware implementation that might be more energy-efficient. Finally, we want to acknowledge the limitations of our experimental paradigm: MNIST is a simple dataset, and the number of tasks is limited. While results from additional experiments suggest that our method generalizes to other datasets (Appendix A.1) and more tasks (Appendix A.2), performance gains are diminished when considering a mixed-dataset training paradigm (Appendix A.2). This suggests a need for an overlap in useful features between tasks for sparse-recurrent DFC to facilitate CL.

Overall, our work showcases the idea of adopting biological principles of neural computation and learning to derive new CL methods that not only perform significantly better than BP, but also show performance comparable to existing CL algorithms.

Author Contributions F.L., B.F.G., P.V.A. conceptualized the project idea. F.L. and M.S. carried out all CL experiments and simulations. F.L., B.F.G., P.A. and M.S. wrote the manuscript. F.L. and B.F.G. made the figures and plots.

Funding Open access funding provided by Swiss Federal Institute of Technology Zurich. This work was supported by the Swiss National Science Foundation (B.F.G. CRSII5-173721 and 315230_189251), ETH project funding (B.F.G. ETH-20 19-01), the Human Frontiers Science Program (RGY0072/2019) and funding from the Swiss Data Science Center (B.F.G, C17-18). P.V.A. was supported by an ETH Zürich Postdoc fellowship. M.S. was supported by an ETH AI Center postdoctoral fellowship.

Code availability The implementation of all of the models, training pipeline and analysis code are available on GitHub(https://github.com/pennfranc/bio-inspired-continual-learning) and archived(https://doi.org/10.5281/zenodo.7414720). Additionally, a modified version of the *hypnettorch* library adapted from von Oswald et al. (2020) used to load split-MNIST data is likewise available on GitHub(https://github.com/pennfranc/hypnettorch) and archived(https://doi.org/10.5281/zenodo.7361495). The implementations of EWC and SI were adapted from Hsu et al. (2018) to be compatible with our codebase.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Appendix A: Additional performance results

A.1 Fashion-MNIST

To assess whether performance gains achieved by sparse-recurrent DFC generalize to other datasets, we repeated our performance experiments on the Fashion-MNIST dataset (Xiao et al. 2017). Due to the high similarity of certain pairs of classes (e.g. sandal/sneaker or pullover/coat) and big differences between others (top/sneaker), domain-IL task performance becomes highly dependent on the order of the classes. To approximate general CL performance, we tested all models on the same 10 random permutations of classes,

which were learned in the same pair-wise fashion as split-MNIST. Averages and standard deviations are thus computed over the 10 permutations of classes. For this performance comparison, we retuned the regularization coefficients for EWC and SI, but did not retune sparsity levels and recurrent learning rate for sparse-recurrent DFC. At most, this would give our approach a slight disadvantage.

Figure 6 shows that our approach still yields performance at least as good as EWC and SI. For domain-IL, the improvements are less significant than the ones we obtained with split-MNIST. However, the same can be said for EWC and SI. From these results, we conclude that our approach generalizes beyond the MNIST distribution. Moreover, not needing to retune the hyperparameters of our model to maintain the performance gains suggests that our approach is quite robust. However, for a thorough assessment of robustness to different datasets, more work has to be done.

A.2 Combining MNIST and fashion-MNIST

To test the continual learning models on a larger number of tasks, as well as to test their robustness to sequential learning across different datasets, we developed the following new continual learning task: We first train the network under consideration sequentially on x pairs of MNIST digits, and then on x pairs of fashion-MNIST pictures. We ran this experiment for $x = 2$ and $x = 4$, resulting in a total of 4 and 8 tasks, respectively. Considering both scenarios allows us to analyse the effect of a sequence task increase (doubling of the number of tasks) in addition to comparing performance of learning algorithms on a new cross-dataset training paradigm. For the same reason as discussed in Appendix A.1, we evaluate CL performance over 10 random permutations of class orderings. The results are shown in Fig. 7.

In the case of 2 tasks per dataset, Fig. 7a shows that sparse-recurrent DFC improves upon BP when it comes to accuracy on dataset 1 after having been trained on both datasets. The improvements are seen in the high learning regime (low LRs), which we would expect to see in learning methods that are more robust against catastrophic forgetting. However, SI and EWC show more significant improvements in accuracy overall. Moreover, because sparse-recurrent DFC does not learn dataset 2 as well as the other methods (Fig. 7b), its performance improvements over BP on both datasets (Fig. 7c) are low, and clearly not as good as EWC and SI.

Doubling the number of tasks in both datasets leads to a similar situation. Figure 7d shows significant and consistent improvements of sparse-recurrent DFC over BP accuracy in high LR regimes, although overall accuracy is lower than that for EWC and SI. Figure 7e shows that, unlike in the case of 2 tasks per dataset, sparse-recurrent DFC generally shows better accuracy on dataset 2 than all other methods. Combining these results, Fig. 7f shows that sparse-recurrent DFC yields significantly higher accuracy than BP over most LRs, as well as similar (although still lower) accuracy as EWC and SI.

Overall, the combination of two different datasets within one sequence task diminishes performance of sparse-recurrent DFC relative to other learning algorithms when compared to the case of just one dataset as seen in Sect. 4.1 and Appendix A.1. However, the reduction in forgetting compared to BP in high learning regimes and on average are still reliable. We suspect that, in the case of more drastic shifts in input distribution (as is the case when shifting from MNIST to fashion-MNIST), explicit task-boundary information is especially useful to consolidate weights important for the first dataset, which would give EWC and SI an advantage over our method. Moreover, the superior single-dataset accuracy of our method also suggests that sparse-recurrent DFC succeeds, especially in those situations where representations learned in one task can be reused in subsequent tasks. While this might sound obvious, the successful reuse of representations is not trivial, as is seen when looking at the CL performance of BP.

To summarize, this experiment shows that, while CL performance improvements over BP can be maintained for both cross-dataset learning, as well as an increase in the sequence task length, it also points to a limit of our method: A certain overlap in terms of useful features between tasks may be a prerequisite for our method to protect against forgetting. To make it more resistant against more drastic changes in input distributions, adaptations may be necessary. In any case, we believe that requiring different tasks to benefit from similar features is not an artificial limitation, as biological brains are likely to reuse common features of the natural environment (e.g. edges, textures,...) when learning different tasks. While MNIST and fashion-MNIST are both visual, MNIST is most likely too simple and idiosyncratic to learn representations that will also be useful for transfer learning on fashion-MNIST.

A.3 Role of model complexity

It might be argued that comparing sparse-recurrent DFC to other models with the same number of neurons is unfair due to the added complexity of lateral and feedback connections. However, this should not be a concern, since the additional connections of sparse-recurrent DFC are only in use during training. For testing, we only use feedforward weights. In other words, after training, recurrent and feedback weights are "discarded" and our model has the exact same complexity as models trained with BP, EWC or SI. Moreover, the notion that additional parameters alone cannot account for performance improvements is also supported by Fig. 3b showing that lateral connections alone do not produce gains in accuracy.

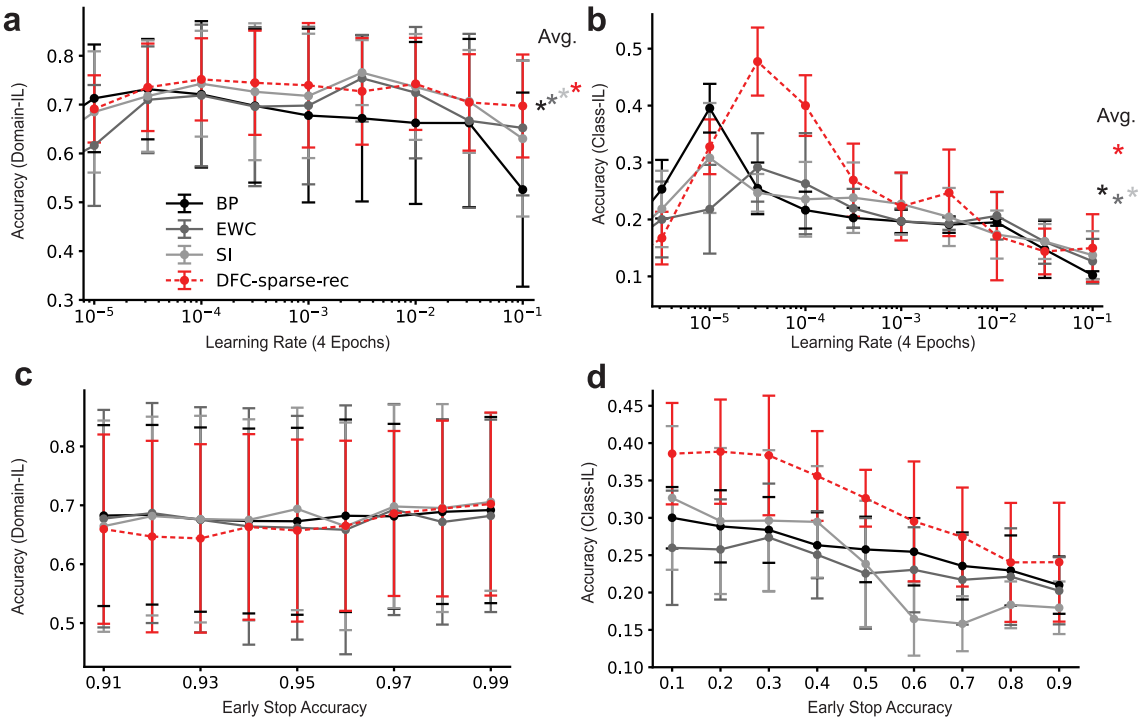

Fig. 6 Performance evaluation of fashion-MNIST for BP, EWC, SI, and DFC-sparse-rec for domain-IL (left column) and class-IL (right column). Error bars represent standard deviations using ten different permutations of the class ordering. **a** Fashion-MNIST accuracy at the end of training in the domain-IL paradigm on the whole test set (all classes) for a range of learning rates (LRs). The number of training iterations is fixed at four epochs. Stars indicate average performance on an accuracy-maximizing window of six LRs. **b** Accuracy of models at the end of training in the class-IL paradigm on the whole test set for every LR. Stars indicate average performance on an accuracy-maximizing window of six LRs. **c** Accuracy of models at the end of training in the domain-IL paradigm on the whole test set for a range of minimum early stop accuracies. The LR is fixed, and training is stopped at every task once the train accuracy for the current batch reaches the given minimum accuracy value. The maximal number of epochs trained for is 10. **d** Accuracy of models at end of training in the class-IL paradigm on the whole test set for a range of minimum early stop accuracies

Regardless, to ascertain that our improvements in CL performance cannot simply be matched by an increase in model complexity, we compared the performance of sparse-recurrent DFC to a bigger BP-trained network, as well as a bigger winner-take-all DFC model (DFC-sparse). Equation 7 shows how the number of parameters in a feedforward network in our setting can be computed as a function of the number of units per hidden layer x. The first, second, and third term represent weights and biases of the first hidden layer, second hidden layer and output layer, respectively. Equation 8 shows how the number of parameters in our recurrent network (including feedback and lateral connections) can be computed. For this, we add the lateral connections in the second term and the feedback connections in the third term.

$$n_{\text{ff}}(x) = (784x + x) + (x^2 + x) + (2x + 2) \tag{7}$$

$$n_{\text{rec}}(x) = n_{\text{ff}}(x) + 2x^2 + 2x \tag{8}$$

To compare our sparse-recurrent DFC model, with 20 units per hidden layer, against a feedforward networks with at least as many parameters during training, we need to find x such that $n_{\text{ff}}(x) > n_{\text{rec}}(20)$, which is already achieved by setting $x = 21$. Unsurprisingly, this will lead to no significant performance improvement compared to $x = 20$, so to show the effect of increased model complexity for BP, we will use $x = 30$. Figure 8a shows how sparse-recurrent DFC compares to the bigger BP network, as well as the default BP network for reference. The same is shown in Fig. 8b for DFC-sparse.

We can see that, while the increased size does improve performance somewhat for low LRs, accuracy decreases at high LRs. Overall, changes in average performance are negligible and the gains in accuracy achieved by DFC-sparse-rec cannot be attributed to increased model complexity during training.

A.4 Accuracy vs. tasks learned

In Sect. 4.1, we evaluate the performance of models on split-MNIST by recording the test accuracies after training on all tasks. To analyse how cumulative performance develops as

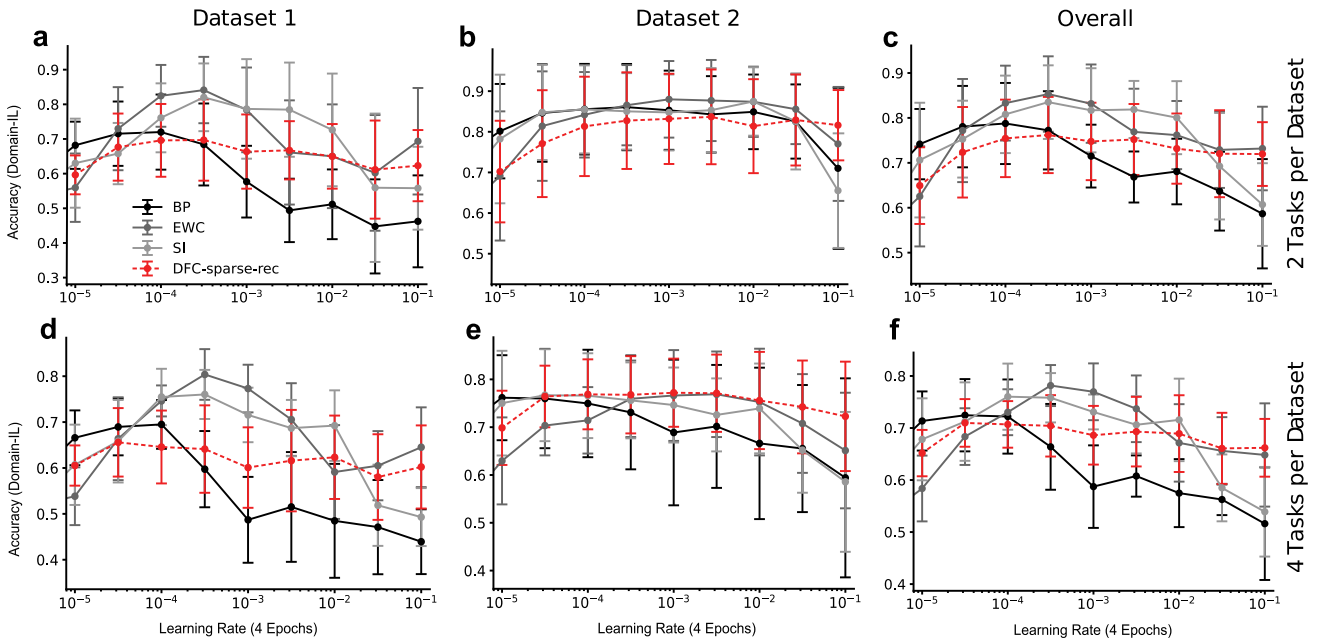

Fig. 7 Performance evaluation of DFC-sparse-rec, BP, EWC, SI on mixed dataset consisting of pairs of MNIST digits and pairs fashion-MNIST pictures. MNIST tasks are presented first as dataset 1, and fashion-MNIST tasks are presented subsequently as dataset 2. Performance is evaluated both for 2 tasks per dataset (first row), as well as 4 tasks per dataset (second row). Accuracies are evaluated at the end of training on the whole sequence task on dataset 1 (first column), dataset 2 (second column) and the combined dataset (third column). Error bars represent standard deviations using ten different permutations of the class orderings. **a** Cross-LR test set accuracy on dataset 1 after learning 2 tasks of both datasets in sequence. **b** Cross-LR test set accuracy on dataset 2 after learning 2 tasks of both datasets in sequence. **c** Cross-LR test set accuracy on the combined dataset after learning 2 tasks of both datasets in sequence. **d** Cross-LR test set accuracy on dataset 1 after learning 4 tasks of both datasets in sequence. **e** Cross-LR test set accuracy on dataset 2 after learning 4 tasks of both datasets in sequence. **f** Cross-LR test set accuracy on the combined dataset after learning 4 tasks of both datasets in sequence

more tasks are learned, we plot the mean accuracy of the first i tasks after training on task i (Fig. 9). Each model was evaluated with its optimal LR for 4 epochs. Curves that start with low accuracies for task 1 can be explained by the fact that choosing an LR that leads to convergence on task 1 is not optimal for the final accuracy on all tasks. Moreover, the increase in cumulative accuracy for task 4 in domain-IL can be attributed to the similarity of the digit pairs 0/1 and 6/7.

Appendix B: Hyperparameters

Our approach for choosing hyperparameters in sparse-recurrent DFC is to start with a configuration that is optimized to solve normal MNIST classification (non-CL) (Meulemans et al. 2022), and to leave all existing parameters unaltered for split-MNIST. Adding sparsity and recurrent gating introduces layer-wise sparsity levels and recurrent learning rate, respectively, as new hyperparameters. These new hyperparameters were tuned separately for domain-IL and class-IL. For EWC and SI, we tuned the regularization coefficient. The overarching principle here is that we only tune hyperparameters specifically associated with solving CL. Table 1 shows all tuned hyperparameters, as well as the activation function (which was not tuned). Table 2 shows the remaining hyperparameters shared by all models. Hyperparameter tuning was performed with respect to maximal average accuracy over a consecutive window of 6 LRs in cross-LR evaluation. The same hyperparameters were used for minimum accuracy evaluation.

Appendix C Hyperplane metrics

In Sect. 4.4 we compute two quantities that involve the use of hyperplanes dividing datapoints into two classes, as per the domain-IL setup (Van de Ven and Tolias 2019). In both cases we obtain the separation hyperplane by fitting a logistic regression model to a set of target activations of the last hidden layer $\{r_{L-1}^{k,j}\}_{k\in t_i}$, where t_i refers to a set of indices of datapoints belonging to task i. $r_{L-1}^{k,j}$ represents the last hidden layer target activations induced by datapoint k after that network has been trained on task j. Let $h_{i,j}$ denote the hyperplane obtained by fitting a logistic regression model to classify $\{r_{L-1}^{k,j}\}_{k\in t_i}$ according to the domain-IL class labels. We use an L1 penalty for the logistic regression model to

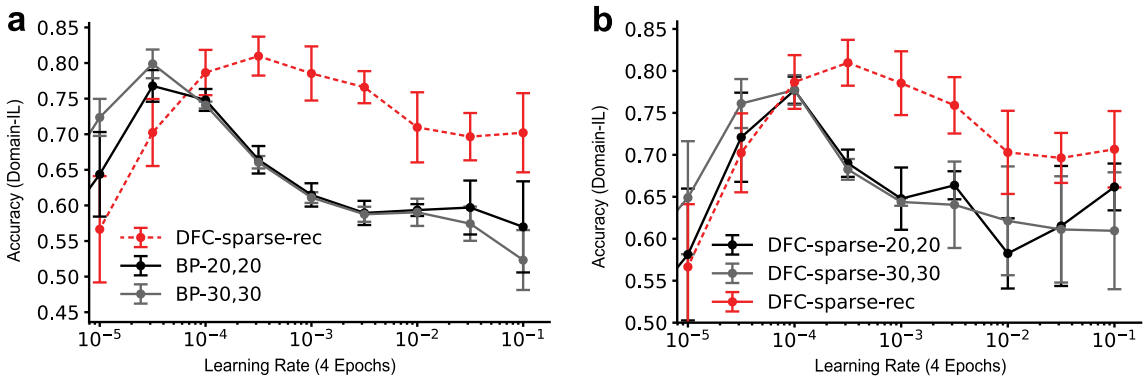

Fig. 8 Analysis of model capacities on domain-IL split-MNIST accuracies. **a** Performance of DFC-sparse-rec network (as used in our previous experiments), the default BP network, and a bigger BP network with 30 units per hidden layer instead of 20. **b** Performance of DFC-sparse-rec network (as used in our previous experiments), DFC-sparse with the same number of neurons, and a bigger DFC-sparse network with 30 units per hidden layer instead of 20

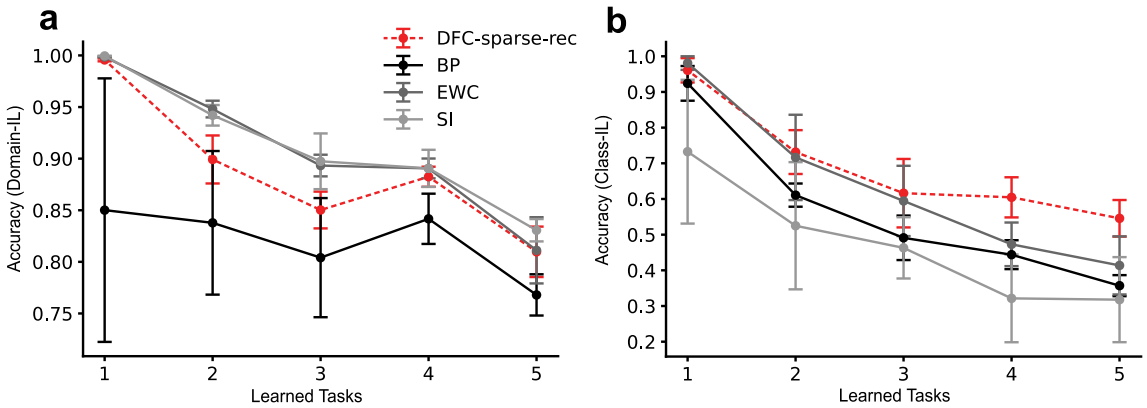

Fig. 9 Average accuracy of first i tasks after training on i'th task. LRs were chosen for each model individually to maximize performance. All models were trained for four epochs. **a** Cumulative domain-IL accuracies for first i tasks on test set. **b** Cumulative class-IL accuracies for first i tasks on test set

Table 1 Model-specific hyperparameters

	DFC	BP	EWC	SI
$s_{1-3,\text{ss}}$ Domain-IL	0.4,0.8,0.5	–	–	–
$s_{1-3,\text{ss}}$ Class-IL	0.2,0.8,0.0	–	–	–
Recurrent LR	40	–	–	–
Reg. coef. domain-IL	–	–	$10^5 \times 2^{-9}$, (10^3)	10^1, (10^1)
Reg. coef. class-IL	–	–	$10^5 \times 2^{-9}$, (10^3)	10^2, (10^3)
Activation function	tanh	relu	relu	relu

Except for the activation function, all of these hyperparameters were tuned for performing well on split-MNIST in a cross-LR evaluation paradigm. Additionally, the regularization coefficients of EWC and SI were retuned for fashion-MNIST in the same cross-LR evaluation paradigm (values in brackets). The three numbers in the sparsity level rows correspond to the 2 hidden layers and the output layer, respectively. The effect of sparsity in the output layer is solely to freeze weights of inactive neurons during training for wrong labels

Table 2 Hyperparameters shared between all used models

	All models
# Hidden layers	2
Hidden layer sizes domain-IL	20,20
Hidden layer sizes combined dataset (domain-IL)	50,50
Hidden layer sizes class-IL	200,200
Learning rate (outside of LR evaluation)	0.001
Batch size	512
Epochs	4
Optimizer	Adam
Forward weight initialization	Xavier

encourage sparse hyperplanes, otherwise we use the default parameters from the sci-kit learn library (Pedregosa et al. 2011).

C.1 Hyperplane alignment

Here we measure the extent to which $\{r_{L-1}^{k,i-1}\}_{k\in t_i}$ are correctly separated by $h_{i-1,i-1}$, that is how well a hyperplane from a previously learned task $i-1$ divides targets of new tasks i, before the network has been fit on the new task. If we represent classification accuracy of $h_{i,j}$ on $\{r_{L-1}^{k,u}\}_{k\in t_v}$ (i, j and u, v representing arbitrary task indices) as $h_{i,j}(\{r_{L-1}^{k,u}\}_{k\in t_v})$, then the hyperplane alignment metric α is given by Eq. 9.

$$\alpha = \frac{1}{4}\sum_{i=2}^{5} h_{i-1,i-1}(\{r_{L-1}^{k,i-1}\}_{k\in t_i}) \tag{9}$$

α values are further averaged over 5 random seeds.

C.2 Movement towards hyperplane

For this metric, we consider distances travelled of feedforward activations, which we would normally refer to as $\phi(v_i^{\text{ff}})$. But because we are running out of space for superscripts, we will refer to $\tilde{r}_{L-1}^{k,j}$ as the last hidden layer feedforward activations induced by datapoint k after that network has been trained on task j. Please note, however, that $h_{i,j}$ is still computed as before, using target activations (including controller and recurrent effects). We quantify the distance of feedforward activations travelled from when they are first learned, to when task 5 training has been finished, with respect to the initially learned hyperplane. More precisely, for all task indices i, we compute the difference of the projections of $\{\tilde{r}_{L-1}^{k,i}\}_{k\in t_i}$ and $\{\tilde{r}_{L-1}^{k,i-1}\}_{k\in t_5}$ on the normal of $h_{i,i}$, which we denote as $n_{i,i}$. Let $T_{i,j}^{c}$ denote the matrix that contains as rows all elements of $\{\tilde{r}_{L-1}^{k,j}\}_{k\in t_i}$ which have c as their correct class label, where $c \in \{0, 1\}$. From these matrices, we can compute the L1 distances travelled by datapoints with class c from task i projected onto the hyperplane normal $n_{i,i}$ as seen in Eq. 10.

$$\tilde{d}_i^c = (-1)^c \cdot (T_{i,5}^c - T_{i,i}^c) n_{i,i} \tag{10}$$

The $(-1)^c$ factor is important to ensure inverted signs of travelled distances in the two classes. We need this because directions towards the hyperplane for one class are directions away from the hyperplane for the other. Because we only want to quantify distance travelled towards the hyperplane direction, and not away from it, we clip the distance vectors to only have positive values.

$$d_i^c = clip(\tilde{d}_i^c, 0, \infty) \tag{11}$$

Finally, we obtain the mean normalized movement towards the hyperplane of activations from task i by dividing the average distance travelled towards $h_{i,i}$ by the average absolute distance travelled in any principal direction, as shown in Eq. 12.

$$\beta_i = \frac{\langle d_i^c \rangle_{c\in\{0,1\}}}{\frac{1}{2}\langle |T_{i,5}^{0,1} - T_{i,i}^{0,1}| \rangle} \tag{12}$$

We need to divide the normalizing factor in the denominator by 2 because we are technically averaging over twice as many directions as there are matrix entries. This is because we consider both positive and negative directions for each principle dimension. The β_i values are averaged over tasks i and 5 random seeds.

Appendix D: Fraction of unaltered subspace

With the unaltered subspace metric γ we attempt to approximate the idea of the fraction of dimensions used by previous tasks that are left unaltered by the current task, as visualized by Fig. 5c. We reuse the notation from the previous section, where $\{r_{L-1}^{k,j}\}_{k\in t_i}$ refers to the set of target activations $r_{L,\text{ss}}$

elicited by datapoints of task i upon learning task j. To quantify the dimensionality of a set of neural activity vectors of a given layer, we utilize the effective rank metric proposed by Roy and Vetterli (2007). The effective rank of a matrix A with positive singular values $\sigma_1, \geq \sigma_2 \geq ... \geq \sigma_Q$ is calculated using Shannon entropy H as shown in Eqs. 13 and 14.

$$p_k = \frac{\sigma_k}{\sum_{k=1}^{Q} |\sigma_k|} \tag{13}$$

$$\text{erank}(A) = exp(H(p_1, ..., p_Q)) \tag{14}$$

We compute the effective rank of the matrix containing activity vectors as rows to quantify the effective dimensionality of the representations. We calculate the effective dimensionality of previously learned tasks (up to but without task i), the current task i, and the combination of previous tasks and the current task as shown in Eqs. 15, 16, 17, respectively.

$$\dim_{\text{prev}}(i) = \text{erank}(\{r_{L-1}^{k,j}\}_{k \in \bigcup_{l=1}^{i-1} t_l}) \tag{15}$$

$$\dim_{\text{curr}}(i) = \text{erank}(r^{t_i}) \tag{16}$$

$$\dim_{\text{cum}}(i) = \max(\text{erank}([r^{t_1}, ..., r^{t_i}]), \dim_{\text{prev}}(i), \dim_{\text{curr}}(i)) \tag{17}$$

Effective rank as a function of sets of target activations does not guarantee monotonicity, which means that the effective rank of a subset of targets can be larger than the effective rank of the superset. To avoid invalid fractions, we guarantee monotonicity between previous, current and cumulative dimensionality by making sure $\dim_{\text{cum}}$ is at least as big as $\dim_{\text{prev}}$ and $\dim_{\text{curr}}$. If we subtract the cumulative dimensionality from the sum of the previous and the current one, we get the intersection of the two, i.e. the dimensionality that is affected by the current task. To quantify the unaltered fraction of previous dimensionality γ, we subtract the fraction of the intersection divided by the previous dimensionality from 1 as shown in Eqs. 18.

$$\gamma = 1 - \frac{\dim_{\text{prev}}(i) + \dim_{\text{curr}}(i) - \dim_{\text{cum}}(i)}{\dim_{\text{prev}}(i)} \tag{18}$$

References

Aljundi R, Kelchtermans K, Tuytelaars T (2019) Task-free continual learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11254–11263

Aljundi R, Lin M, Goujaud B, Bengio Y (2019) Gradient based sample selection for online continual learning. Adv Neural Inf Process Syst 32

Duncker L, Driscoll L, Shenoy KV, Sahani M, Sussillo D (2020) Organizing recurrent network dynamics by task-computation to enable continual learning. Adv Neural Inf Process Syst 33:14387–14397

French RM (1991) Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In: Proceedings of the 13th annual cognitive science society conference, vol 1, pp. 173–178

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Hsu Y-C, Liu Y-C, Ramasamy A, Kira Z (2018) Re-evaluating continual learning scenarios: a categorization and case for strong baselines. arXiv preprint arXiv:1810.12488

Jedlicka P, Tomko M, Robins A, Abraham WC (2022) Contributions by metaplasticity to solving the catastrophic forgetting problem. Trends Neurosci

Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A et al (2017) Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci 114(13):3521–3526

Kudithipudi D, Aguilar-Simon M, Babb J, Bazhenov M, Blackiston D, Bongard J, Brna AP, Raja SC, Cheney N, Clune J et al (2022) Biological underpinnings for lifelong learning machines. Nat Mach Intell 4(3):196–210

Laborieux A, Ernoult M, Hirtzlin T, Querlioz D (2021) Synaptic metaplasticity in binarized neural networks. Nat Commun. https://doi.org/10.1038/s41467-021-22768-y

Lee S, Ha J, Zhang D, Kim G (2020) A neural Dirichlet process mixture model for task-free continual learning. In: International conference on learning representations

Levinson M, Kolenda JP, Alexandrou GJ, Escanilla O, Cleland TA, Smith DM, Linster C (2020) Context-dependent odor learning requires the anterior olfactory nucleus. Behav Neurosci 134(4):332

Lin AC, Bygrave AM, De Calignon A, Lee T, Miesenböck G (2014) Sparse, decorrelated odor coding in the mushroom body enhances learned odor discrimination. Nat Neurosci 17(4):559–568

Mahendran A, Vedaldi A (2016) Visualizing deep convolutional neural networks using natural pre-images. Int J Comput Vis 120(3):233–255

Manneschi L, Lin AC, Vasilaki E (2021) Sparce: improved learning of reservoir computing systems through sparse representations. IEEE Trans Neural Netw Learn Syst

Masse NY, Grant GD, Freedman DJ (2018) Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. Proc Natl Acad Sci 115(44):E10467–E10475

Michael McCloskey, Cohen Neal J (1989) Catastrophic interference in connectionist networks: the sequential learning problem. Psychol Learn Motivat 24:109–165

Meulemans A, Farinha MT, Ordonez JG, Aceituno PV, Sacramento J, Grewe BF (2021) Credit assignment in neural networks through deep feedback control. Adv Neural Inf Process Syst 34:4674–4687

Meulemans A, Farinha MT, Cervera MR, Sacramento J, Grewe BF (2022) Minimizing control for credit assignment with strong feedback. In: KC, Stefanie J, Le S, Csaba S, Gang N, Sivan S (eds) Proceedings of the 39th international conference on machine learning, vol 162 of *Proceedings of machine learning research*, pp 15458–15483. 17–23 PMLR

Morcos AS, Barrett DGT, Rabinowitz NC, Botvinick M (2018) On the importance of single directions for generalization. In: International conference on learning representations. https://openreview.net/forum?id=r1iuQjxCZ

Parisi GI, Ji X, Wermter S (2018) On the role of neurogenesis in overcoming catastrophic forgetting. arXiv preprint arXiv:1811.02113

Parisi GI, Kemker R, Part JL, Kanan C, Wermter S (2019) Continual lifelong learning with neural networks: a review. Neural Netw 113:54–71

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E

(2011) Scikit-learn: machine learning in Python. J Mach Learn Res 12:2825–2830
Pourcel J, Vu N-S, French RM (2022) Online task-free continual learning with dynamic sparse distributed memory. In: European conference on computer vision. Springer, pp 739–756
Rao D, Visin F, Rusu A, Pascanu R, Teh YW, Hadsell R (2019) Continual unsupervised representation learning. In: Advances in neural information processing systems, 32
Roy O, Vetterli M (2007) The effective rank: a measure of effective dimensionality. In: 2007 15th European signal processing conference. IEEE, pp 606–610
Rusu AA, Rabinowitz NC, Desjardins G, Soyer H, Kirkpatrick J, Kavukcuoglu K, Pascanu R, Hadsell R (2016) Progressive neural networks. arXiv preprint arXiv:1606.04671
Shin H, Lee JK, Kim J, Kim J (2017) Continual learning with deep generative replay. Adv Neural Inf Process Syst 30
van Bergen RS, Kriegeskorte N (2020) Going in circles is the way forward: the role of recurrence in visual inference. Curr Opin Neurobiol 65:176–193
Van de Ven GM, Tolias AS (2019) Three scenarios for continual learning. arXiv preprint arXiv:1904.07734
van de Ven GM, Siegelmann HT, Tolias AS (2020) Brain-inspired replay for continual learning with artificial neural networks. Nat Commun 11(1):1–14
Van Rullen R, Gautrais J, Delorme A, Thorpe S (1998) Face processing using one spike per neurone. Biosystems 48(1–3):229–239
von Oswald J, Henning C, Sacramento J, Grewe BF (2020) Continual learning with hypernetworks. In: International conference on learning representations. https://arxiv.org/abs/1906.00695
Wang Z, Shen L, Fang L, Suo Q, Duan T, Gao M (2022) Improving task-free continual learning by distributionally robust memory evolution. In: International conference on machine learning, pp 22985–22998. PMLR
Xiao H, Rasul K, Vollgraf R (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747
Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer, pp 818–833
Zeng G, Chen Y, Cui B, Shan Yu (2019) Continual learning of context-dependent processing in neural networks. Nat Mach Intell 1(8):364–372
Zenke F, Poole B, Ganguli S (2017) Continual learning through synaptic intelligence. In: International conference on machine learning. PMLR, pp 3987–3995