Neural Networks 168 (2023) 615–630

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

A biologically inspired architecture with switching units can learn to generalize across backgrounds☆

Doris Voina [a,*], Eric Shea-Brown [a,b], Stefan Mihalas [a,b]

[a] Department of Applied Mathematics, Computational Neuroscience Center, University of Washington, Seattle, WA 98195, USA
[b] Allen Institute for Brain Science, 615 Westlake Ave N, Seattle, WA 98109, USA

ARTICLE INFO

Keywords:
Switching network
Bio-inspired
Context
Generalization
Continual learning
Domain adaptation

ABSTRACT

Humans and other animals navigate different environments effortlessly, their brains rapidly and accurately generalizing across contexts. Despite recent progress in deep learning, this flexibility remains a challenge for many artificial systems. Here, we show how a bio-inspired network motif can explicitly address this issue. We do this using a dataset of MNIST digits of varying transparency, set on one of two backgrounds of different statistics that define two contexts: a pixel-wise noise or a more naturalistic background from the CIFAR-10 dataset. After learning digit classification when both contexts are shown sequentially, we find that both shallow and deep networks have sharply decreased performance when returning to the first background — an instance of the catastrophic forgetting phenomenon known from continual learning. To overcome this, we propose the bottleneck-switching network or switching network for short. This is a bio-inspired architecture analogous to a well-studied network motif in the visual cortex, with additional "switching" units that are activated in the presence of a new background, assuming a priori a contextual signal to turn these units on or off. Intriguingly, only a few of these switching units are sufficient to enable the network to learn the new context without catastrophic forgetting through inhibition of redundant background features. Further, the bottleneck-switching network can generalize to novel contexts similar to contexts it has learned. Importantly, we find that — again as in the underlying biological network motif, *recurrently* connecting the switching units to network layers is advantageous for context generalization.

1. Introduction

The ability to adapt to changes in context while preserving relevant information that is contextually invariant is one of the traits that make biological brains so effective and robust in complex, natural environments. In the visual domain, for example, humans are particularly adept at generalization across contexts and will react similarly when seeing a familiar face during a remote video call or during an interaction in person. Translating this capacity for contextual adaptation to artificially intelligent agents is essential if, for example, we are to upgrade current visual learning algorithms to generalize across new environments and abstract visual concepts (Beery, Horn, & Perona, 2018). A central motivation of this study is to find the relevant network mechanisms that allow artificial neural networks to accomplish adaptation and switching to different contexts. One difficulty in this setting is that most current deep feedforward architectures learn the background and foreground in tandem, without abstracting objects of interest from their surround. For example, recent studies show networks that repeatedly misclassify familiar objects set on new backgrounds (Beery et al., 2018), while select adversarial backgrounds may fool even deep state-of-the-networks up to 87% of the time (Xiao, Engstrom, Ilyas, & Madry, 2020). As we will show, simply relying on network depth is insufficient. Instead, novel architectures supporting context-dependent computations are required.

In general, datasets used to train current state-of-the-art neural networks are shown to be biased (Amir, Zemel, & Tsotsos, 2018; Choi, Torralba, & Willsky, 2012; de Vries, Misra, Wang, & van der Maaten, 2019; Wang, Narayanan, & Russakovsky, 2020), with object class correlating with background, or the background statistics being constrained to belong to a particular distribution. For example, certain datasets may predominantly contain street or nature scenes, or have a preferred viewing angle (Torralba & Efros, 2011); in the case of ImageNet, the dataset predominantly contains centered objects with limited

☆ This work was supported in part by National Institute of Health Training Grant 5 R90 DA033461-0, and in part by the Allen Institute for Brain Science. We thank Paul G. Allen, the founder of the Allen Institute for Brain Science, for his vision, encouragement, and support.

* Correspondence to: Department of Applied Mathematics University of Washington, Lewis Hall #201, Box 353925, Seattle, WA 98195-3925, USA.
E-mail address: dvoina@uw.edu (D. Voina).

https://doi.org/10.1016/j.neunet.2023.09.014
Received 6 July 2022; Received in revised form 24 August 2023; Accepted 7 September 2023
Available online 17 September 2023
0893-6080/© 2023 Elsevier Ltd. All rights reserved.

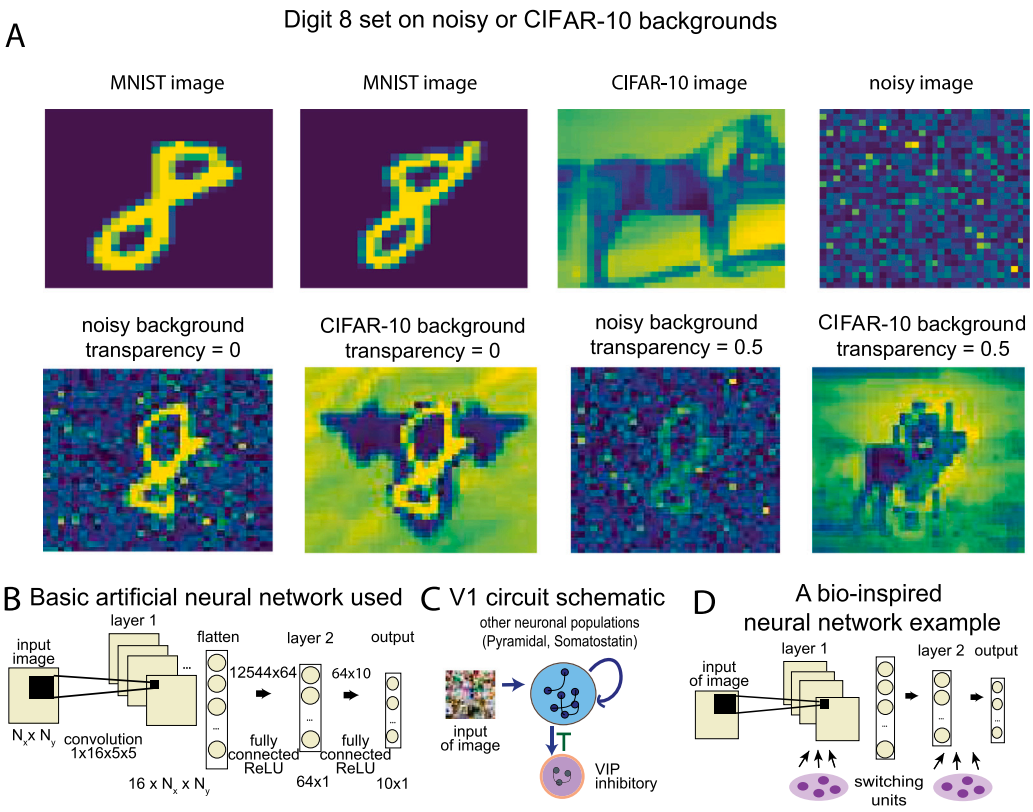

Fig. 1. Contexts and networks used for sequential context switching. (A) Digit "8" set on two different backgrounds (noisy and CIFAR-10 background) with different levels of transparency T. (B) Basic artificial neural network architecture without switching units used for the toy examples presented below. (C) Schematic of the biological circuit in V1 with the VIP population acting as binary switch that turns ON/OFF due to the stationary and moving states of the animal (Yu et al., 2014). (D) A bio-inspired neural network with switching units.

background clutter, while the PascalVOC dataset depicts more complex scenes with multiple objects and a significant amount of background clutter (Oquab, Bottou, Laptev, & Sivic, 2014). Such biases prevent even the most sophisticated deep architectures from generalizing across contexts so that when the network is tested on a dataset with different background statistics the accuracy can be significantly affected (Barbu et al., 2019). Few studies so far, such as Barbu et al. (2019), Beery et al. (2018), have disentangled different dataset biases to separately address how tasks like object classification are affected by variations of context.

To address these challenges systematically, we first construct a simple dataset where context is clearly defined and without other complicating biases. The dataset consists of MNIST digits set on either pixel-wise noisy backgrounds or more naturalistic backgrounds of images from the CIFAR-10 dataset (Fig. 1A). To parametrically vary the difficulty of this task, we control the transparency of the MNIST digits. We use this dataset in a biologically realistic setting where agents learn to classify digits set on different backgrounds but are exposed to these contexts sequentially and without having access to previous data points. This is the continual learning framework, where a major challenge is that old tasks are forgotten when network weights are overwritten to solve new tasks, a phenomenon called *catastrophic forgetting* (McCloskey & Cohen, 1989; Ratcliff, 1990). We show that both deep and shallow networks trained to classify MNIST digits set on the two contexts fail to sequentially learn without catastrophic forgetting. We refer to this classification task with varying context as *sequential context switching*.

We introduce a new network architecture for solving sequential context switching that is roughly inspired by a recently characterized local circuit in the mouse visual cortex, as described in Voina, Recanatesi, Hu, Shea-Brown, and Mihalas (2022). This circuit in the mouse primary visual cortex (V1) modulates its activity whenever an animal shifts from stationary to moving behavior via a neuronal population expressing Vasoactive Intestinal Peptide – the VIP population. It has been hypothesized that this neuronal population, which is recurrently connected to other neurons in V1 (e.g., Pyramidal and Somatostatin neurons), turns ON like a switch and reconfigures the circuit dynamics to efficiently process the corresponding static vs. moving visual scenes (Voina et al., 2022; Yu et al., 2014). Inspired by this biological circuit motif and without emulating other features of the V1 circuit, we propose a network architecture – the *bottleneck-switching network* or *switching network*, for short – where, using a feedforward neural network to start (Fig. 1B), we add units that are OFF when the network is presented the first context but turn ON for the second context, similarly to the VIP (Fig. 1C-D). We refer to these added units as *switching units*. Crucially, our architecture is constrained by a contextual signal that modulates the switching units and turns them ON or OFF; such a signal is either assumed as given or can be inferred through a separate module (see Section 3.4). Due to this network's architecture and the training method implemented (Section 3.4), *catastrophic forgetting is guaranteed not to occur*.

Our paper makes the following main points: (1) We introduce and share a simple dataset for the classification of MNIST images set on contexts of different (parametric and nonparametric) statistics. (Section 3.1, Fig. 1A). (2) Using this dataset, we demonstrate systematically how catastrophic forgetting occurs in sequential context switching for both deep and shallow neural networks (Section 3.2, Fig. 2). (3) We then test a set of basic network architectures that incorporate context in different ways, and find that they fail to achieve sequential context switching (Section 3.3, Supplementary Figure 3). (4) We propose a bio-inspired architecture, the *bottleneck-switching network*, that succeeds in sequential context switching while also using a few additional units (Section 3.4, Fig. 3). (5) We show how the *switching network* can improve performance relative to other established methods by assessing the performance of EWC (Kirkpatrick et al., 2017) and ProgNet, (Rusu

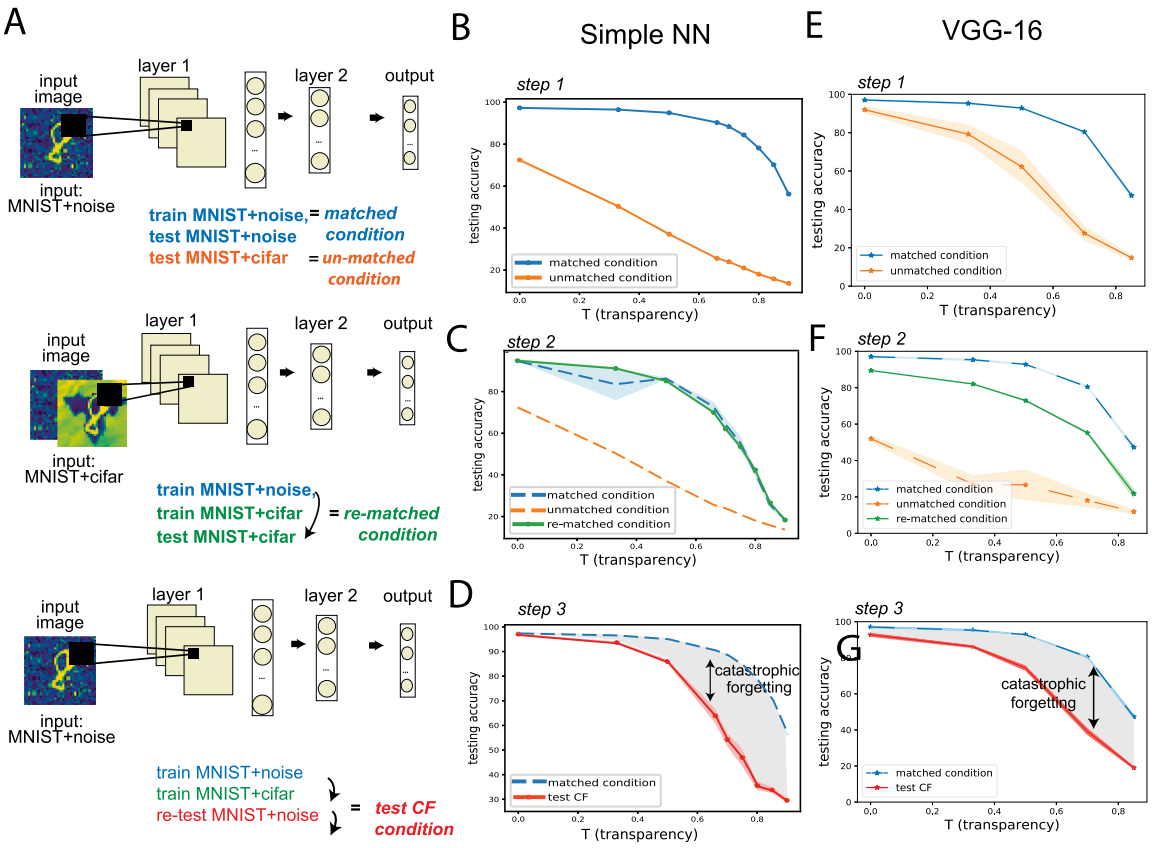

Fig. 2. The catastrophic forgetting phenomenon occurs for sequential context switching in both deep and shallow neural networks. (A) Schematic of training and testing for sequential context switching to test catastrophic forgetting. (B) Average performance on the *matched condition* for MNIST+noise (blue) and on the *unmatched condition* for MNIST+cifar (orange) as transparency T increases. (C) The MNIST+cifar dataset is less accurately classified by a network trained on MNIST+noise (*unmatched condition*, orange line) than one trained on MNIST+cifar (*matched condition*, blue line). However, once we re-train the MNIST+noise-trained NN on MNIST+cifar (Step 2), the accuracy approaches that for the *matched condition* (*re-matched condition*, green line). (D) The performance when re-testing MNIST+cifar on a network that was first trained on MNIST+cifar, then on MNIST+noise, is reduced (*test CF*, red line) compared to the *matched condition*, therefore catastrophic forgetting occurs. (E)-(G) same as (B)-(D) using the VGG-16 network instead of the basic network.

et al., 2016) in the sequential context switching task (Section 3.5, Fig. 4). (6) We propose a mechanism behind switching networks' high performance, via a sparsification of the initial network activities, specifically through inhibition of background features (Section 3.6, Fig. 5). (7) Testing digit-background pairs that have not been trained on the switching network, we find that our network has superior performance to a feedforward network trained on one context (Section 3.7, Fig. 6A-B). (8) Further, we show that switching networks can generalize well to distinct contexts whose image statistics are similar to the ones it has trained on (Section 3.8, Fig. 6C-E).

2. Related work

Transfer learning (TL), domain adaptation (DA), multi-task learning (MTL), and continual learning (CL) are allied fields that have the broad goal of training networks on two or more datasets or tasks, with the common objective of using the knowledge learned from one task (source task) to efficiently learn a different but related task (target task). General strategies (Weiss, Khoshgoftaar, & Wang, 2016) applied to this end include trying to correct for the input marginal distribution difference (Duan, Tsang & Xu, 2012; Glorot, Bordes, & Bengio, 2011; Li, Pan, Jin, Yang, & Zhu, 2012; Oquab et al., 2014; Pan, Tsang, Kwok, & Yang, 2009; Shi & Sha, 2012) or the conditional distribution difference (Chattopadhyay, Ye, Panchanathan, Fan, & Davidson, 2011; Daume, 2007; Duan, Xu & Chang, 2012; Long, Wang, Ding, Sun, & Yu, 2013; Tommasi, Orabona, & Caputo, 2010; Xia, Zong, Hu, & Cambria, 2013; Yao & Doretto, 2010) between the source and target tasks. In this paper, we focus on context switching, when only the features and statistics of the background change.

This scenario is most similar to domain adaptation (DA), a particular case of TL (specifically, heterogeneous, transductive TL, Csurka, 2017; Weng & Deng, 2018) when the source and target tasks are identical but the source and target features, as well as their distribution are different due to selection bias or distribution mismatch. A rich literature spanning decades describes DA methods aiming to solve the domain shift between the source and target domains (Bousmalis, Trigeorgis, Silberman, Krishnan, & Erhan, 2016; Bruzzone & Marconcini, 2010; Chu, de la Torre, & Cohn, 2013; Gong, Grauman, & Sha, 2013; Li, Wang, Shi, Liu, & Hou, 2016; Liu & Tuzel, 2016; Long, Wang, & Jordan, 2016; Pan et al., 2009; Sun & Saenko, 2016; Xiao, Li, Ouyang, & Wang, 2016). Here we also focus on the case when the data is observed incrementally as a continuous stream —- the CL setting. Unlike learning in TL and MTL, the network in this case does not have access to old data that can be interleaved with the new data, but rather agents continually learn new knowledge across time while retaining previously learned information (Parisi, Kemker, Part, Kanan, & Wermter, 2019). While the dataset of different contexts we have chosen distinctly requires us to consider DA as an appropriate framework, we have chosen a task (sequential context switching) whose structure promotes the formation of context-independent features through the CL framework, assuming that classification without catastrophic forgetting is conducive to context invariance and generalization. Datasets/contexts and tasks requiring DA are not always studied in the CL setting, however our study of sequential context switching implies that our work lies at the intersection of the two fields.

Models in CL have taken inspiration from neurophysiological principles (e.g. Hebbian plasticity, compensatory homeostatic plasticity, (Abraham & Robins, 2005; Zenke, Gerstner & Ganguli, 2017), and from

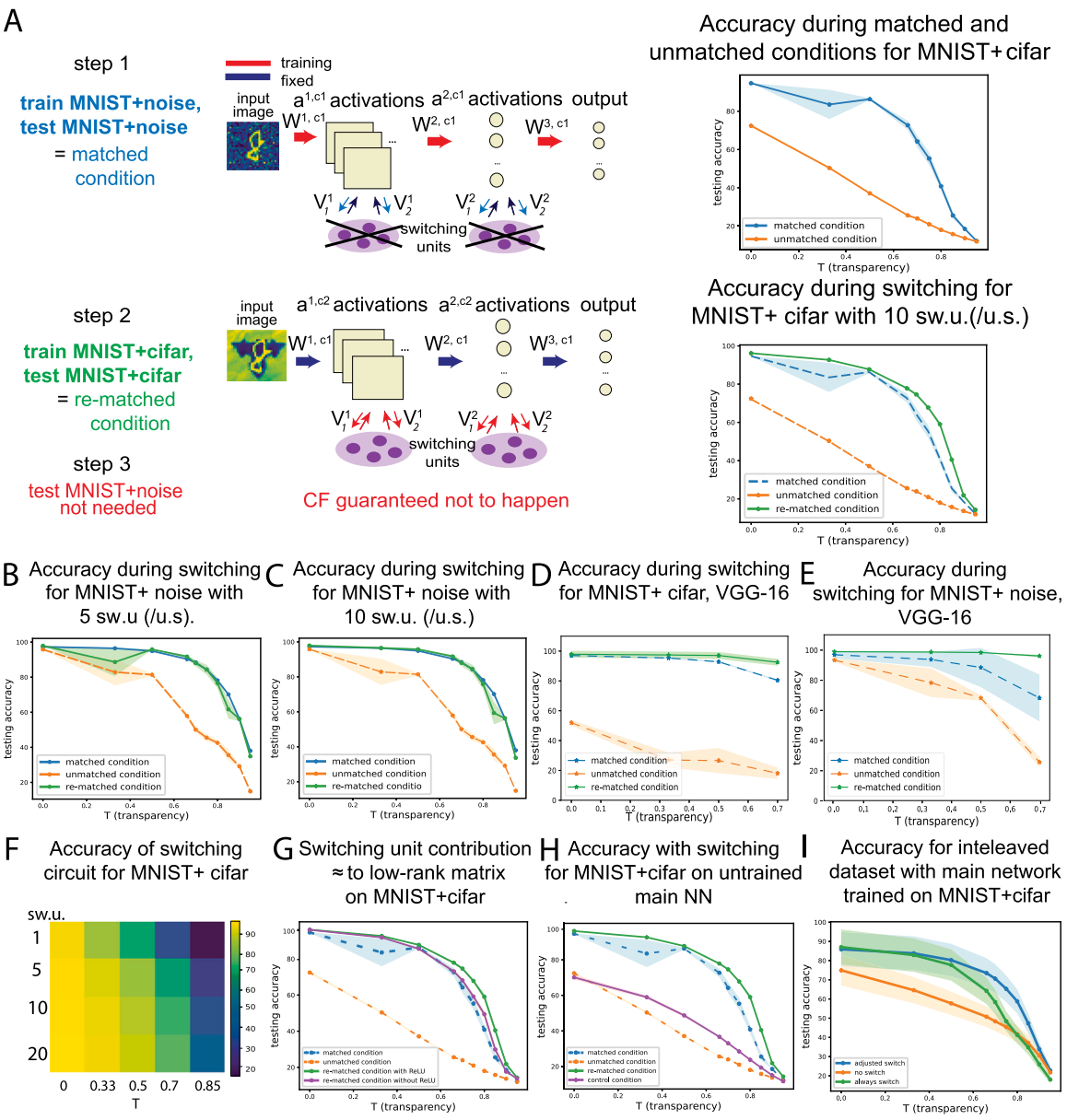

Fig. 3. The bottleneck-switching network is successful in solving the continual learning problem and uses additional units that amount to a low-rank contribution to activities. (A). Summary and schematic of the training procedure for the bottleneck-switching network. (B)-(E). Testing accuracy on MNIST+noise/cifar vs transparency T: after training in the *matched condition* (step 1, blue line); after training in the *unmatched condition* (orange line); after training in the *re-matched condition* with switching units (per unit space for convolutional layers; step 2, green line). "sw.u." stands for switching unit(s), "u.s." stands for unit space. We always add the recurrent connections to the switching units and test the corresponding context using 1, 5, or 10 switching units (see Supplementary Figure 4A, for a network using 1 switching unit per unit space). (D)-(E). Testing accuracy on the VGG-16 with switching units. (F). Heat map of testing accuracy on the basic network across number of switching units and transparency T. (G). A linearized version of the switching network that is equivalent to a low rank perturbation on weights is applied on the MNIST+cifar dataset (green line) and performs comparably well to the matched condition (blue line) and to the switching network as described in main text Eq. (1) (purple line). (H). Accuracy for the bottleneck-switching network on MNIST+noise (purple line), when training weights to and from the switching units, but keeping weights in the main network randomly initialized. (I). Accuracy using the network with a built-in downstream module that classifies contexts and turns the switching units ON or OFF automatically; main network is trained on MNIST+cifar and switching units are trained on MNIST+noise.

computational learning models (complementary learning systems theory, (Kumaran, Hassabis, & McClelland, 2016; McClelland, McNaughton, & O'Reilly, 1995)). There are several categories of computational approaches to CL that permit good generalization and avoid catastrophic forgetting (Parisi et al., 2019): (i) learning models that regulate levels of plasticity to protect consolidated knowledge through regularization (Donahue et al., 2014; Kim, Yoo, Park, & Kim, 2021; Li & Hoiem, 2016; Razavian, Azizpour, Sullivan, & Carlsson, 2014; Su et al., 2020; Volpi, Larlus, & Rogez, 2021); (ii) adding additional neural resources such as neurons to learn new information (Draelos et al., 2017; Rebuffi, Kolesnikov, Sperl, & Lampert, 2016; Wang, Fink, Gool, & Dai, 2022; Xiao, Zhang, Yang, Peng, & Zhang, 2014; Yoon, Yang, Lee, & Hwang, 2018; Zhou, Sohn, & Lee, 2012); (iii) using complementary learning systems for memory consolidation and experience replay (French, 1997; Hinton & Plaut, 1987; Hofmanninger et al., 2020; Kemker & Kanan, 2018; Kim et al., 2021; Rostami, 2021; Shin, Lee, Kim, & Kim, 2017; Taufique, Jahan, & Savakis, 2022). In the case of regularization methods, a successful strategy has been applied in Kirkpatrick et al. (2017), describing the EWC, and in Zenke, Poole and Ganguli (2017) by which more influential parameters from previous tasks are pulled back towards a reference weight with good performance on previous tasks. An example of the second approach where the network dynamically adds neuronal resources is the ProgNet architecture (Rusu et al., 2016).

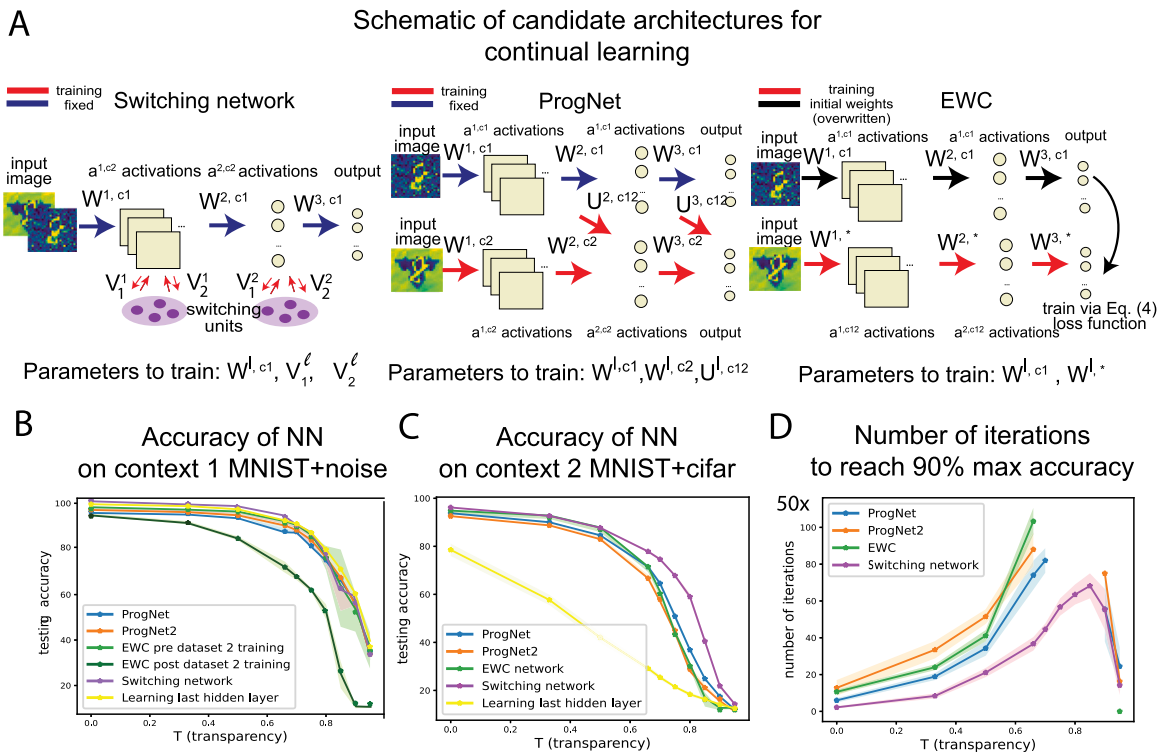

Fig. 4. The bottleneck-switching network compares favorably to other continual learning solutions. (A) Schematic of network architectures: bottleneck-switching network, ProgNet, and EWC. (B) Testing accuracy for context 1 (MNIST+noise) for the five NNs. Only the EWC (dark green) on context 1 after training on context 2 (MNIST+cifar) shows a poorer performance because it cannot overcome catastrophic forgetting. All plot lines, except for the dark green line corresponding to EWC post-dataset 2 training, largely overlap, but are plotted with larger spacing, for clarity. (C). Testing accuracies on context 2 (MNIST + cifar) for various NNs. We note that the switching network has a slight edge across all transparencies. The plot lines corresponding to ProgNet, ProgNet2, and the EWC network largely overlap, but are plotted with slightly larger spacing, for clarity. (D). Number of iterations (in steps of 50) for each NN, during training on MNIST+cifar, to reach 90% of the maximum peak accuracy over all the methods. Several data points are not shown because the respective network does not reach 90% of this peak accuracy.

This architecture expands through the allocation of new "column" networks, trained on novel information, and receiving lateral connections from the other columns. We will describe ProgNet as presented in Rusu et al. (2016), and EWC as introduced in Kirkpatrick et al. (2017) in more detail below.

Recently developed methods have successfully addressed both CL and DA. In Taufique et al. (2022), the authors propose a method based on episodic memory replay with buffer management. A "contrastive" loss is incorporated for better alignment of the buffer samples and the continual stream of batches that represent the gradually evolving stream of new data. In a novel study aimed at improving medical imaging applications through machine learning, a dynamic memory enables rehearsal on a subset of diverse training data to mitigate forgetting while enabling models to expand to new domains (Hofmanninger et al., 2020). In yet another study (Kim et al., 2021) utilizing memory replay, in this case for semantic segmentation, authors implement a lightweight sub-memory called Target-specific Memory (TM) which is initiated, trained, and stored for each target domain. To overcome catastrophic forgetting, each TM contains unique information corresponding to each domain discrepancy so that the semantic segmentation network can adapt to the current target domain while preserving the knowledge learned on previous target domains. By optimizing a Double Hinge Adversarial loss function, the segmentation network aligns the source and target domain data while considering geometric relations between them. A method aimed at solving both CL and unsupervised domain adaptation (UDA) (Rostami, 2021) consolidates the learned internal distribution such that all learned tasks share a similar distribution in the embedding space, while catastrophic forgetting is mitigated using experience replay by storing and then replaying the input samples that are more informative for estimating the internally learned distribution.

Other popular methods at the intersection of CL and DA use regularization: for instance, authors in Volpi et al. (2021) devise a meta-learning strategy where a regularizer explicitly penalizes any loss associated with transferring the model from the current domain to different "auxiliary" meta-domains, while also easing adaptation to them. In the case of vision tasks, these meta-domains are constructed by randomizing the current domain's distribution with heavy image manipulations. Another example is Gradient Regularized Contrastive Learning (GRCL) (Su et al., 2020) which uses a source discriminative constraint formulated to constrain that the gradient of the parameters should be positively correlated to the gradient of the classification loss for the source domain; and a target memorization constraint that constrains the gradient of the parameters to be positively correlated to the gradient of classification loss for every old target domain. Alternatively, in continual test-time adaptation approach (CoTTA) the error accumulation due to the change of tasks is reduced by using weight-averaged and augmentation-averaged predictions, which are often more accurate than the pseudo-labels used in UDA (Wang et al., 2022). To avoid catastrophic forgetting, this method proposes to stochastically restore a small part of the neurons to the source pre-trained weights during each iteration to help preserve source knowledge in the long-term.

However, these methods do not specifically address generalization across background and are not tested against a common benchmark. Furthermore, while these network architectures have performed impressively in Machine Learning tasks, to the best of our knowledge they have not been related to specific biological circuits.

Our work shares commonality with a minority of bio-inspired architectures performing continual learning, such as Kirkpatrick et al. (2017), Zenke, Poole et al. (2017). Other related architectures worth mentioning here are continual learning studies that successfully make use of bio-inspired context-specific components, such as Ellefsen, Mouret,

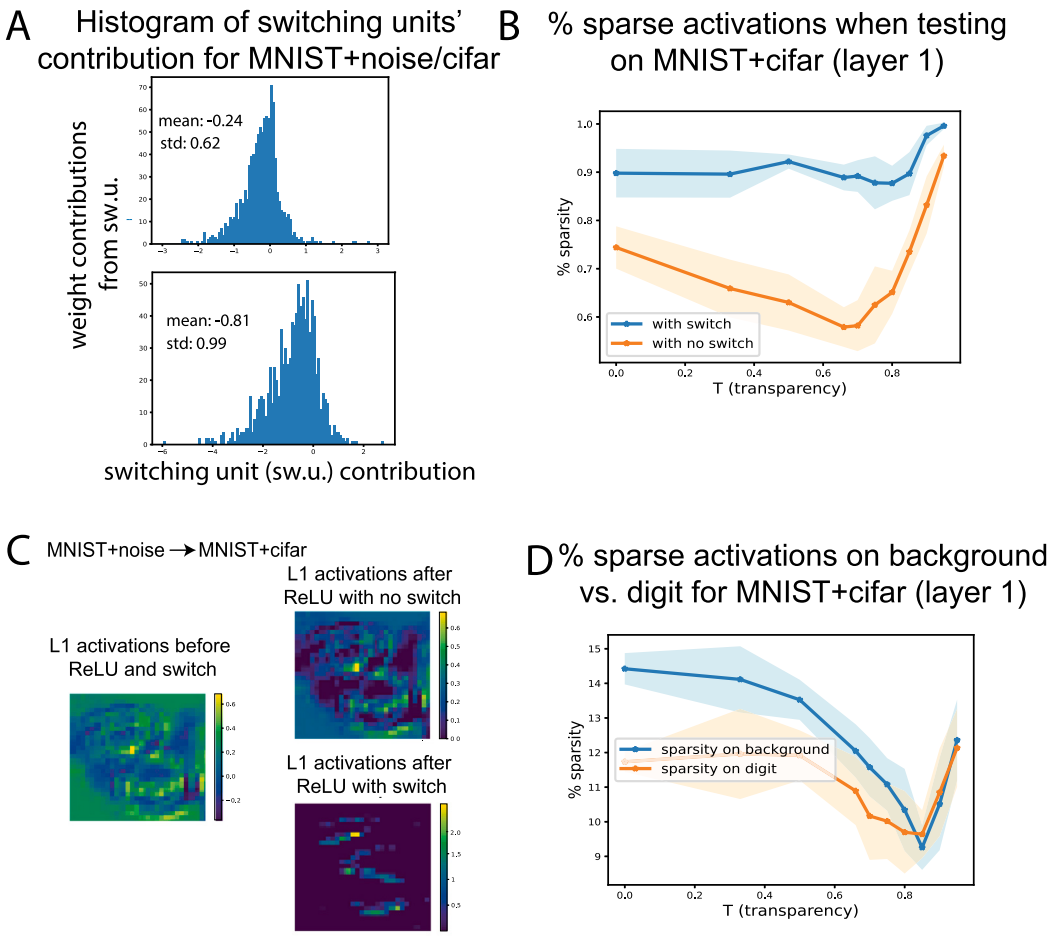

Fig. 5. The switching mechanism leads to sparser activation patterns, specifically an inhibition of redundant background features. (A) Histograms of switching unit contributions to the main network (for switching to MNIST+noise, up; for switching to MNIST+cifar, down). (B) Percentage layer 1 sparse activations with or without adding the switching contributions (blue vs orange line), and after applying ReLU. (C) Examples of (convolutional) layer 1 activation patterns with or without the switching contribution. (D) Percentage layer 1 sparsity (zero activations) within the background (blue) and within the digit (orange) when switching to MNIST+cifar after adding the switching contributions and the ReLU non-linearity.

and Clune (2015), Masse, Grant, and Freedman (2018). In Ellefsen et al. (2015), continual learning capability can be boosted by implementing modular networks which intuitively reduce learning interference between tasks by isolating functionality into physically distinct network modules where learning can be alternatively turned on or off. To produce modular networks, i.e. networks that have many clusters or modules of highly connected neurons that are only sparsely connected to other neurons, these architectures are evolved with a cost for neural connections. Another pertinent study relevant to our own (Masse et al., 2018) uses context-dependent gating in tandem with the synaptic stabilization implemented in Kirkpatrick et al. (2017), Zenke, Poole et al. (2017) to achieve continual learning. This context-dependent signal enables only sparse, mostly non-overlapping subsets of units to be active for any given task. Our own work complements these few studies implementing biologically inspired architectures and learning to solve the continual learning problem.

An important limitation in our study is that context is either known a priori or can be trivially inferred. For many related continual learning and domain adaptation problems, context identification (or task inference) is in fact the main challenge and has been successfully addressed (Henning et al., 2021; van de Ven & Tolias, 2019). In the continual learning literature this setting is referred to as "task-incremental learning" (van de Ven & Tolias, 2019). Authors in van de Ven and Tolias (2019) extensively compare continual learning methods on split and permuted MNIST dataset protocols, specifically testing scenarios when the task identity is provided and when it is not. They show that, when task identity needs to be inferred, regularization based approaches like EWC (Kirkpatrick et al., 2017) fail while replaying representations of previous experiences successfully achieves continual learning.

Direct comparison between these methods has been difficult because of a lack of established benchmark datasets and metrics (van de Ven & Tolias, 2019). Many training and evaluation protocols have shifted from MNIST or CIFAR-10 datasets (Jung, Ju, Jung, & Kim, 2016; Kirkpatrick et al., 2017; Zeng, Chen, Cui, & Yu, 2019; Zenke, Poole et al., 2017), to more challenging datasets (ImageNet, Russakovsky et al., 2015; MS COCO, Lin et al., 2014; OpenImages, Kuznetsova et al., 2018), similar in complexity to realistic settings (Kaiser et al., 2017; Kirkpatrick et al., 2017; Li & Hoiem, 2016; Mallya, Davis, & Lazebnik, 2018; Mallya & Lazebnik, 2018; Misra, Shrivastava, Gupta, & Hebert, 2016). However, large datasets also contain substantial contextual biases, in background/context, rotation, viewpoint, etc., and have insufficient controls to ensure networks do not exploit trivial correlations in the data (Barbu et al., 2019; Sagawa, Pang, Tatsunori, & Percy, 2020; Shetty, Fritz, & Schiele, 2018; Xiao et al., 2020; Zhu, Xie, & Yuille, 2017). For instance, in Xiao et al. (2020) authors find that changing backgrounds in ImageNet significantly decreases average performance, and that choosing backgrounds in an adversarial manner can lead to misclassifying 87.5% of the images.

Therefore, an important undertaking is to carefully analyze generalization (or lack thereof) in detection and classification tasks as discussed in Beery et al. (2018), dissecting the biases neural networks can abuse or misuse (variations in lightning, viewpoints, context/background etc.). To that end, our work distinctively addresses

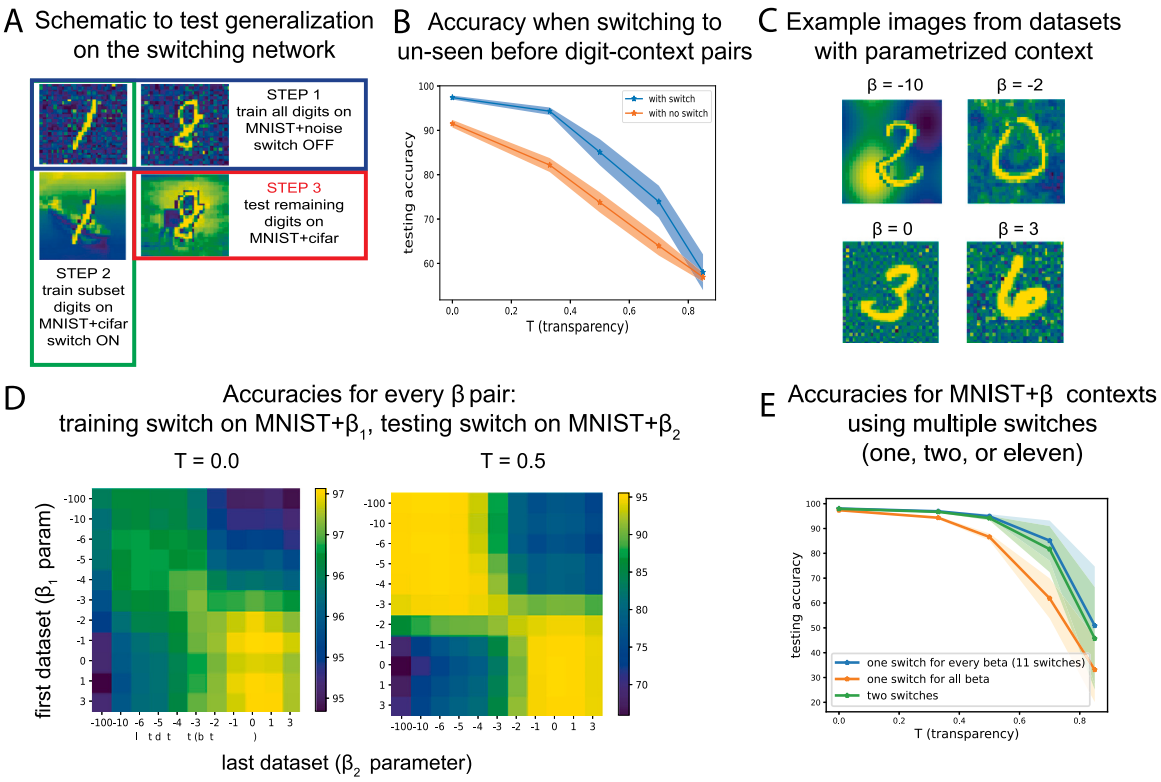

Fig. 6. The switching network improves accuracy for digit-background pairs that the network has not been trained on, compared to simple feedforward networks trained on one context. (A) Schematic of training the switching network to test generalization performance for digit-background pairs that switching units' weights have not been trained on. (B) Overall accuracy, averaged over ten randomly chosen digit pairs, of switching network (with switch ON, blue line; with switch OFF, orange line) when tested on digit-context pairs that the switch has not been trained on. (C) Example images from MNIST + β parametrized backgrounds. (D) Test accuracies for different β pairs and two values of T ($T = 0.0, 0.5$): training the main network on MNIST+cifar, training the weights to and from the switching units on MNIST+β_1, testing the switching network (with switch ON) on MNIST+β_2. (E). Testing accuracies for MNIST+β datasets when using one (orange line), two (green line), or eleven switches (blue line). We show that using two switches (i.e. two sets of switching units trained on particular subsets of MNIST+β contexts) is as effective as training 11 switches (i.e. a set of switching units for every context MNIST+β).

variations of context for a classification task, when the network has to generalize by ignoring contexts and abstracting MNIST digits. More precisely, this work sets out to (1) systematically address context generalization for parametric and non-parametric backgrounds, and specifically when the statistics of the background is varied in a controlled way (see Section 3.8); (2) do a thorough comparison between our bio-inspired architecture, the bottleneck-switching network, and relevant prior methods like EWC and Prognet, or other simple architectures.

3. Results

We introduce a simple dataset for the classification of MNIST images set on contexts of different statistics (Section 3.1). Using this dataset, we demonstrate systematically how catastrophic forgetting occurs in a sequential context switching task and subsequently propose a bio-inspired architecture, the bottleneck-switching network, that succeeds in sequential context switching by using a few additional units (Sections 3.2–3.4). The addition of few units to the network is equivalent to a low rank perturbation to the original ANN weights (Section 3.4). Compared to other methods, like ProgNet or EWC, the bottleneck-switching network can improve on performance and learn in fewer iterations (Section 3.5). The mechanism behind the bottleneck-switching networks' high performance is through inhibition of background features (Section 3.6). Moreover, the network can generalize well to digit-background pairs that have not been shown during training (Section 3.7). Finally, we show that distinct sets of switching units for each context are not required, rather the switching architecture can generalize to different contexts whose image statistics are within a range to the ones the switch has trained on (Section 3.8).

3.1. A simple dataset for sequential context switching

To address the problem of sequential context switching, we create a dataset whose only confounding feature is context. We make this dataset publicly available to test generalization across contexts. The dataset consists of MNIST digits with varying degrees of transparency set on either noisy backgrounds, with noise being the absolute value of a Gaussian random variable normalized appropriately, or MNIST digits set on a more naturalistic background from the CIFAR-10 dataset (Fig. 1A, Methods Sec. 1.1). We refer to the subset of MNIST digits set on noisy backgrounds as "MNIST+noise" and to the subset of MNIST digits set on CIFAR-10 backgrounds as "MNIST+ cifar". The goal is to perform image classification so that neural networks (NN) correctly identify the MNIST digit despite the different backgrounds. A parameter that makes the task more difficult is digit transparency; as we increase transparency, the background interferes with the digit and the identity of the digit becomes more ambiguous (Fig. 1A, right).

3.2. Catastrophic forgetting in the MNIST+noise and MNIST+cifar datasets

To systematically address the problem of sequential context switching and infer suitable network architectures, we start by first choosing a basic NN (Fig. 1B) that solves the digit classification task with a performance above 90% on either MNIST+noise (96% accuracy) or MNIST+cifar (93% accuracy) datasets with no digit transparency (Fig. 2B, Supplementary Figure 1B). The classic MNIST classification task can be performed at high accuracies using even simple linear classifiers, with accuracies as high as 92.4% reported in LeCun, Bottou, Bengio, and Haffner (2013). However, the dataset we focus on here

contains backgrounds and additional transparency T causing the backgrounds to obscure the digits, making this a more difficult task. For instance, for intermediate levels of transparency ($T = 0.5$) accuracies are 94% and 85% for MNIST+noise and MNIST+cifar, respectively. For high levels of transparency ($T = 0.85$), the same accuracies are 67% and 25%, respectively.

For details on how the training of the basic NN and other networks is run, see Supplementary Material, Methods Sec. 1.2.2.

Turning to the problem of continual learning applied to context switching, we ask whether this NN learns digit classification on MNIST+noise and MNIST+cifar, sequentially and in either order. To achieve continual learning, we implement the following steps (Fig. 2A, Supplementary Figure 1A):

- at **step 1**, we train the network on context 1, then test on context 1 – this is the *matched condition*;
  - we test context 2 on the network trained on context 1 – this is the *unmatched* condition;
- at **step 2**, we retrain the NN on context 2 and then test on context 2 – this is the *re-matched condition*;
- at **step 3**, we re-test context 1 – this is the test catastrophic forgetting or *test CF condition*, where context 1,2 are either MNIST+noise or MNIST+cifar.

Our initial findings are as follows:

- at **step 1**, networks learn digit classification on context 1 with high accuracies (Fig. 2B, Supplementary Figure 1B), in this case up to 96% on the test set (Fig. 2B, inset), with reduced accuracy at increased transparency.
  - Testing the NN in the *unmatched condition* using context 2, the accuracy is comparatively poor (orange vs blue lines in Fig. 2C). The difference between the matched and unmatched conditions is statistically significant ($p < 0.05$, one-sided t-test, see Supplementary Material Sec. 1.2.3). This is the effect of changing context: weights from the prior context can no longer be used for accurate classification in the novel context.
- At **step 2**, for the *re-matched condition*, we obtain a performance (green line, Fig. 2C, Supplementary Figure 1C) comparable to the *matched condition*, as if we trained on context 2 from scratch (blue line).
- At **step 3**, in the *test CF condition*, the network significantly fails to remember the original dataset MNIST+noise after being retrained on MNIST+cifar (Fig. 2D) and vice versa (Supplementary Figure 1D, $p < 0.05$, one-sided t-test). This is especially true in cases of increased transparency. See Supplementary Material Sec. 1.2.3 for a detailed report on statistical significance.

Moreover, increasing either the depth or the width of the network does not alleviate this forgetting. For depth, we use the well-known VGG-16 (Simonyan & Zisserman, 2015). For width, we increase twofold the number of filters or hidden units in the convolutional and linear layers, respectively. Fig. 2E-G, Supplementary Figure 1E-G, and Supplementary Figures 2A,B all show that the accuracy of the network in the *test CF condition* (orange line) remains lower than in the *matched condition* (blue line). In summary, Fig. 2B-G demonstrates that CF occurs when context changes in a step-wise manner, indicating a failure of standard feedforward NN's in the present continual learning setting.

3.3. Simple networks with contextual input, output, or feedforward switching units fail to perform sequential context switching

We next test three simple modifications to the networks that could avoid catastrophic forgetting.

The first strategy is to add a binary contextual input to the NN, similar to the input received in some widely used recurrent neural network models (e.g., see Mante, Sussillo, Shenoy, & Newsome, 2013). The input is added to the hidden layers (Supplementary Figure 3A, top network) and is set to 0 for context 1 (e.g., MNIST+noise) and 1 for context 2 (e.g., MNIST+cifar). For simplicity, we assume the contextual input perfectly matches the true context, and does not have to be inferred from the input images. We implement the following steps to test this architecture:

- at **step 1**, we test context 1 on a network trained on context 1, without any contributions from binary units, which are set to 0 – this is the *matched condition*;
- at **step 2**, we train the NN on context 2 while setting the binary units to 1 and learning weights from these units in tandem with other network weights;
- at **step 3**, the binary units are set to 0 once again and we test the performance of the network on context 1 – this is the *test CF condition*.

Eq. (1) describes the activities in this simple network's hidden layers:

$$\begin{aligned} a^{l,c_1} &= \sigma(W^l a^{l-1,c_1}), \\ a^{l,c_2} &= \sigma(W^l a^{l-1,c_2} + V^l \mathbb{1}). \end{aligned} \tag{1}$$

where σ is a ReLU non-linearity, $\mathbf{a}^{l,c}$ are the activities at layer l for contexts $c = c_{1,2}$ (c_1 for context 1; c_2 for context 2), $\mathbf{W}^l$ are the weights between layers $l-1$ and l, layer $l = 0$ is the input layer, and $\mathbf{a}^{0,c}$ is the input. $\mathbb{1}$ is a vector of ones, $\mathbf{V}^l$ is a matrix of (learned) weights, therefore $V^l \mathbb{1}$ is a vector of constants added at c_2. At step 1, only $\mathbf{W}^l$ is learned; at step 2, both $\mathbf{W}^l$ and $\mathbf{V}^l$ are learned in tandem.

Essentially, the additional binary units in this network effectively add a learnable, context-specific bias to each hidden neuron every time context 2 is shown.

We find that in this case catastrophic forgetting still occurs (red vs blue lines in Supplementary Figure 3B,E).

The second strategy was to require a separate binary *output* for context: 0 for context 1 and 1 for context 2 (Supplementary Figure 3A, middle network). The steps we go through here are similar, except for replacing the added binary units with binary outputs which are required to be 0 when training/testing context 1 (steps 1,3) and 1 for context 2 (step 2). Eq. (2) describes this simple network's hidden layer activations:

$$a^l = \sigma(W^l a^{l-1}), a^L \in \mathbb{R}^{11} \tag{2}$$

where σ is the ReLU non-linearity, $\mathbf{a}^l$ are the activations at the hidden layer l, $\mathbf{W}^l$ are the weights between layers $l-1$ and l, a^0 is the input, and a^L are the activations at the last hidden layer. $\mathbf{a}^L$ is 11-dimensional, where the first ten dimensions are the output digit classification units, and the eleventh dimension is the output context.

Again, this network fails to reduce catastrophic forgetting (red vs blue lines in Supplementary Figure 3C,F). The idea underpinning these strategies is that by either directly providing knowledge about the identity of the context (noisy or CIFAR-10 context) – or explicitly requiring the network to represent this context in its output – the NN would be able to separate contextual information from digit information, and would learn to use only the latter in the classification task.

The third strategy takes inspiration from the local circuit in the mouse visual cortex discussed above (Voina et al., 2022; Yu et al., 2014), where the circuit modulates its activity whenever an animal shifts from stationary to moving behavior via the "VIP" neuronal population. It has been hypothesized that this neuronal population, which is recurrently connected to other neurons in V1 (e.g., Pyramidal and Somatostatin neurons), turns ON like a switch and reconfigures the circuit dynamics to efficiently process the corresponding static vs. moving visual scenes (Voina et al., 2022; Yu et al., 2014).

In our first, simplest model inspired by this circuit, we add *switching units* akin to the VIP neurons: they are OFF for the first context and ON for the second context, while abstracting away other VIP properties (e.g. their inhibitory effect and their recurrent connectivities). This architecture is the "feedforward switching network", with the NN from Fig. 1B as the main network, while the switching units added are auxiliary units (Supplementary Figure 3A, bottom network). We describe the step-by-step training applied to the feedforward switching network in the following:

- at **step 1**, we train then test context 1 with the switching units OFF – this is *matched condition*.
- at **step 2**, we hold the weights learned at step 1, when the switching units were OFF, fixed. We then train the NN on context 2 while setting the switching units to 1 and learning weights only from these units. The contribution of the switching units gets added to the activations of the main network hidden layer before it gets through a ReLU non-linearity. For the convolutional layers, we add the switching units' contributions to the flattened hidden activations of the network. The resulting performance is the *re-matched condition*.
- we no longer require a separate **step 3**, as catastrophic forgetting is guaranteed not to happen. To re-test digit classification on context 1, we simply turn the switching units OFF, returning the network (and therefore its performance) to an identical state to step 1's *matched condition*.

This simple network with switching units can be described through an equation which is in fact equivalent to Eq. (1):

$$\begin{aligned} a^{l,c_1} &= \sigma(W^l a^{l-1,c_1} + V^l), \\ a^{l,c_2} &= \sigma(W^l a^{l-1,c_2} + V^l \mathbb{1}). \end{aligned} \tag{3}$$

where we utilize the same general notation as in Eq. (1), and $\mathbf{V}^l$ may be interpreted here as being the weights from the switching units to the main network. We note that this is a similar network to the one with binary inputs described above, with one important difference: after training for context 1 with the switching units OFF, we *freeze* the weights already learned and only changing the weights from the switching units. In short, for context 1, we train only the weights $\mathbf{W}^l$, while for context 2 we train only $\mathbf{V}^l$, and keep $\mathbf{W}^l$ constant.

As before in the first network we described, the switching units in this architecture effectively add a learnable, context-specific bias to each hidden neuron every time context 2 is shown.

For this switching network we emphasize a key fact: we are guaranteed to return to original performance on the first learned context by simply turning the switching units OFF. Thus, such a network will automatically overcome catastrophic forgetting — if it can solve the task on context 2 by only learning the weights from the switching units. However, we find that this does not occur. The feedforward switching network achieves accuracies on context 2 (Supplementary Figure 3D, G, green line) only slightly higher than the *unmatched condition* when using a network trained on context 1 alone (orange line), and far below the reference *matched* condition (blue line). (This is consistent with allied findings in Voina et al., 2022). As before, the conclusions hold when context 1 is for either MNIST+noise or MNIST+cifar (Supplementary Figure 3D,G), and we assume this is the case throughout the paper unless otherwise noted. In sum, while the feedforward switching network does not, by construction, show catastrophic forgetting, it has inferior performance on the second context.

We conclude that none of the three strategies considered above is successful in sequential context switching. Our first and third approaches are akin to the context signal implemented in Masse et al. (2018), which was also ineffective at a continual learning problem that attempted to learn different permutations of MNIST digits for each task. We next attempt to continue in the same line of work as Masse et al. (2018) and Ellefsen et al. (2015), which successfully achieved continual learning by making use of bio-inspired, context-specific components. In Masse et al. (2018), the algorithm used a contextual signal that was unique for each task and projected onto all hidden neurons, such that sparse, largely non-overlapping patterns of units were co-activated for every task. Alternatively, Ellefsen et al. (2015) developed modular neural networks through an evolutionary algorithm such that learning interference between tasks was naturally reduced. In the next section, we expand on these current architectures and introduce an improvement to the feedforward switching network that attains success.

3.4. Networks with recurrent switching units succeed in performing sequential context switching

We next show that, given a contextual signal which is known a priori or that we can readily infer through a separate module, we may improve the performance of the feedforward switching network by adding recurrent connections to the switching units (blue or red arrows from the switching units, Fig. 3A, left). These connections are directly inspired by biological data and confirmed by a prior computational model of the V1 circuit described in Voina et al. (2022), which suggests recurrence from switching units is necessary and sufficient for context-dependent computations. The goal is to study a bio-inspired circuit motif that incorporates into a feedforward artificial NN architecture a specific cell type reproducing the simplified activity of the VIP neural population. We then assess whether this motif enables sequential context switching between backgrounds of different statistics, analogously to the V1 circuit switching between static and moving contexts.

Inspired by the overall structure of the V1 circuit motif we propose a network architecture — the *bottleneck-switching network* (or *switching network*). This is the same as the feedforward switching network above (Supplementary Figure 3A, bottom network) but allows recurrent connections back to switching units. In detail, using a feedforward NN to start (Fig. 1B), we add *switching units* akin to the feedforward switching units from the previous section (Section 3.3), only that they are recurrently connected to the main network units. As before, the switching units are OFF when the network is presented the first context, but turn ON for the second context, similarly to the VIP (Fig. 1C,D). Our main assumption here is that the contextual signal is either given a priori or can be trivially inferred through a separate module (see below). Due to this network's architecture and the training method implemented, *catastrophic forgetting is guaranteed not to occur*.

As for the feedforward switching network, we follow the same sequence of steps:

- at **step 1**, we train then test context 1 with the switching units OFF — this is the *matched condition*.
- at **step 2**, we hold the weights learned at step 1, when the switching units were OFF, fixed. We then train the NN on context 2 while setting the switching units ON and learning weights to and from these units. The contribution of the switching units gets added to the activations of the main network hidden layer before it gets through a ReLU non-linearity. The resulting performance is the *re-matched condition*.
- we no longer require a separate **step 3**, as catastrophic forgetting is guaranteed not to happen.

When adding switching units to convolutional hidden layers, we aim to perform convolutional operations from the hidden layers to the switching units and back to the main network, so we essentially add kernels. Each kernel is a square matrix that extracts spatial features from the images, so we refer to the number of kernels added as switching units per unit space (sw.u/u.s.).

The activities of this improved network at layer l for contexts c_1, c_2 can be expressed as:

$$\begin{aligned} a^{l,c_1} &= \sigma(W^{l,c_1} a^{l-1,c_1}) \\ a^{l,c_2} &= \sigma(W^{l,c_1} a^{l-1,c_2} + V_2^l \sigma(V_1^l W^{l,c_1} a^{l-1,c_2})). \end{aligned} \tag{4}$$

where W^{l,c_1} are the weights between layers $l-1$ and l of the main network for context 1 (c_1), σ is the non-linearity (ReLU), a^{l,c_i} are the activations at layer l for context c_i, a^{0,c_i} is the input image, and V_1^l, V_2^l are the weights to and from the switching units at layer l, as shown in Fig. 3A. For classifying digits set on the first context c_1, we simply use the basic NN, a feedforward neural network whose activities at layer l are computed via the first equation by applying a linear operation to the activities at the previous layer/input using the weights W^{l,c_1}, then implementing a non-linear function σ (Fig. 3A (step 1)). For classifying digits set on c_2, we apply a similar equation to compute the activities in the network layers, except we add an additional term from the switching units, $V_2^l\sigma(V_1^l W^{l,c_1} a^{l-1,c_2})$ that essentially projects the activations in the current layer, $W^{l,c_1} a^{l-1,c_2}$ onto a lower dimensional space (the switching units), applies a non-linear function, then finally projects back onto the layer l activities via the matrix V_2^l (Fig. 3A (step 2)).

When the switching units are ON, we learn connections to and from these units, intending to classify context 2. This strategy succeeds: we find that the accuracy on context 2 for different values of T is much improved when adding even one switching unit per unit space at the first layer (green line in Supplementary Figure 4A, see also Fig. 3B-E). We can compare this performance (*re-matched condition*) to the *unmatched* and *matched conditions*: the *unmatched condition* entails testing context 2 on the NN trained on context 1 (orange lines) and represents a lower bound, while the *matched condition* implies testing context 2 on the NN trained on context 2 (blue lines) and is the accuracy we want to reach or exceed. As we add more switching units, the performance of the bottleneck-switching network on context 2 approaches or even surpasses the *matched condition*. Using 5 switching units (5 sw.u./u.s. for convolutional layers and 5 simple units for regular hidden layers), the performance reaches that of the *matched condition* (Fig. 3B), with $p < 0.06$ when testing the difference between the re-matched and matched condition on MNIST+cifar (one-sided t-test). There is an even more pronounced boost in performance as we increase the number of switching units to 10 (Fig. 3A,C), with $p < 0.03$ when testing the difference between re-matched and matched conditions on MNIST+cifar (one-sided t-test). Supplementary Material Sec. 1.2.3 contains a more detailed report on statistical significance. When adding 10 switching units we use small 3×3 kernels, which causes the total number of parameters to be comparatively reduced.

Similar conclusions hold when we test the switching mechanism on 3 and 4-layer neural networks similar to the bottleneck-switching network implemented above (Supplementary Figure 7A-C). Switching units applied to the convolutional layers of the much deeper VGG-16 network also achieve sequential context switching, with superior accuracy in the *re-matched condition* using switching units to classify MNIST+cifar or MNIST+noise (Fig. 3D-E).

Thus, we see that generally few switching units are required for the highest accuracy to be achieved. Using the basic NN as the main network, the accuracy either plateaus or increases slowly, such that with 5-10 switching units we are close to peak performance (Supplementary Figure 4B,C). Using the same basic network, a summary of how testing performance varies with the number of switching units and transparency shows the same tendency of the accuracy to plateau as switching units increase (Fig. 3F, Supplementary Figure 4D). For the VGG-16, we use approximately a tenth of the number sw.u/u.s. present in the VGG main network, and solely for a subset of the convolutional layers (see Methods Sec. 1.2.1,), hence comparatively fewer switching units are used for the deep network as well.

To further analyze the switching network motif, we next study a version of this network simplified by eliminating the ReLU non-linearity σ after the switching contributions are computed. In other words, we study a linearized version of the recurrent switching component of the network. The activities of the NN after this simplification can now be expressed as:

$$\begin{aligned} a^{l,c_2} &= \sigma(W^{l,c_1} a^{l-1,c_2} + V_2^l V_1^l W^{l,c_1} a^{l-1,c_2}) \\ &= \sigma((W^{l,c_1} + V_2^l V_1^l W^{l,c_1}) a^{l-1,c_2}). \end{aligned} \tag{5}$$

The contribution from the switching units is $V_2^l V_1^l W^{l,c_1} a^{l-1,c_1}$. This contribution becomes a low-rank addition to the activities of the NN if few switching units are required, i.e. V_1^l, V_2^l are lower dimensional in one of the two dimensions of the matrix. We find that this low-rank contribution successfully performs sequential context switching ($p < 3e{-}10$ for MNIST+cifar, comparing the unmatched and re-matched conditions using a one-sided t-test). Fig. 3G shows the performance of the bottleneck-switching network on context 2 (*re-matched condition*) approaching and surpassing the *matched condition*, just as before, even as accuracies are reduced compared to when the ReLU non-linearity is used. This suggests that varied network mechanisms that implement low-rank updates to weight matrices may be sufficient to support contextual switches. Further, we can link our work to a rich body of literature that emphasizes the effectiveness of low-rank weights in NNs trained for a variety of tasks (Bau, Liu, Wang, Zhu, & Torralba, 2020; Swaminathan, Garg, Kannan, & Andres, 2020).

We then ran a series of controls to further test the effectiveness of the bottleneck-switching network.

First, we shuffled labels for images set on context 2, while images set on context 1 had typically assigned labels ("1" for a written digit of 1, etc.). We find that switching units do not work as well when the context is unchanged, but the task has different input–output dependencies (Supplementary Figures 5A,B). This suggests that for sufficiently different datasets — differing through more than the change of context features, the switching network fails to reach similarly high performance.

Second, we assessed a network where only weights to and from the switching units are learned, while the weights in the main network are fixed and randomly initialized (Fig. 3H, Supplementary Figure 5C). The goal is to establish baseline performance when the main network does not contribute features from context 1 to the task. To infer this baseline performance, we train weights to and from the switching units (turned ON) on context 2, without modifying weights in the main network. We then test this network on context 2 to find that the accuracy on context 2 is low, barely surpassing the *unmatched condition*, especially when the switching network with switching units ON learns MNIST+cifar (purple line, Fig. 3H). The accuracies for the matched condition are significantly higher than the accuracies for this control condition ($p < 2e-7$ for MNIST+cifar, one-sided t-test). We conclude that main network weights and activations from training on a similar dataset are necessary for sequential context switching. Hence the switching network utilizes a type of transfer learning, where previous knowledge from training on images of digits is used to train on a similar dataset of digits set on a different context.

Furthermore, to test whether the presence of recurrent connections is a critical feature of the proposed approach, we look at another alternative architecture that uses the switching units in a purely feedforward fashion. This architecture is more complex than the feedforward switching network of Section 3.3, but implements a bottleneck module without the recurrency. We refer to this network as the *feedforward bottleneck-switching network* and offer a schematic of this architecture in Supplementary Figure 6A. We can express the activities of this network at layer l for contexts c_1, c_2 through the following equation:

$$\begin{aligned} a^{l,c_1} &= \sigma(W^{l,c_1} a^{l-1,c_1}) \\ a^{l,c_2} &= \sigma(W^{l,c_1} a^{l-1,c_2} + V_2^l \sigma(V_1^l a^{l-1,c_2})). \end{aligned} \tag{6}$$

We find that the feedforward bottleneck-switching network has a similar performance to the bottleneck-switching network on the re-matched condition tested on MNIST+cifar, but has superior performance for higher levels of transparency when tested on MNIST+noise (Supplementary Figure 6B,C). In addition, we note that the bottleneck-switching network has much fewer parameters to learn than its feedforward counterpart (an order of over 120,000 fewer parameters). We thus find the recurrence within our bottleneck-switching network is a beneficial feature that makes the network's performance competitive when compared to other similar architectures.

Finally, we address a simplification we made in the framework presented above: that switching units turn ON/OFF based on perfect knowledge of the context presented. For a more realistic case, in which contexts are detected from images, the network can include a downstream module consisting of a one-layer network to identify contexts through a binary classification using supervised learning, with *a priori* training on the noisy and CIFAR-10 backgrounds. In this setting, the switching network includes binary multiplicative inputs from the one-layer network that enable the switching units to turn ON and OFF automatically based on context. Augmenting such a module to classify contexts to the bottleneck-switching network does not deteriorate performance ($p < 0.05$ for higher T on MNIST+cifar, one-sided t-test; see Fig. 3I, Supplementary Figures 15A,B), as can be seen by comparing the accuracy with an augmented trained module (green line), compared to the accuracies when the network never turns ON the switching units (orange line), or when the switching units are always OFF (blue line).

We conclude that relatively few switching units that are recurrently connected to the main network layers can provide substantial performance improvement in the sequential context switching task and that the switching unit contribution can be approximated to a low-rank perturbation to the weights.

3.5. A comparison between the bottleneck-switching network and two established continual learning methods

We next compare the bottleneck-switching network with two other learning methods that could achieve generalization across contexts: Progressive Networks and Elastic Weight Consolidation. Both these methods assume the context of the input images is known and provided to the network. Whereas many studies in the literature describe these methods' ability to perform continual learning by switching between *tasks*, we are focused on switching between *contexts*, and will test the noisy and CIFAR-10 contexts in our framework. A detailed summary of the results below, along with the relevant statistical tests can be found in the Supplementary Material, Sec. 1.2.3.

Progressive Network (ProgNet). ProgNets (Rusu et al., 2016) maintain a set of pre-trained networks ("columns") for each task (or context) and learn lateral connections between these columns to extract useful features from related tasks (contexts) (Fig. 4A middle, Methods sec. 1.2.3). During learning, the parameters of columns trained on previous contexts are kept constant, so weights are not overwritten and the network is immune to catastrophic forgetting by design. Considering only two columns corresponding to classification of different contexts (noise/CIFAR-10), the activations in column 2 of the ProgNet can be expressed as:

$$a^{l,c_2} = \sigma(W^{l,c_2} a^{l-1,c_2} + U^{l,c_{12}} a^{l-1,c_1}), \tag{7}$$

where a^{l,c_k} are the activations of layer l of column c_k, $c_k \in \{1,2\}$, $W^{l,c_k} \in \mathbb{R}^{n_l \times n_{l-1}}$ is the weight matrix of layer l and column c_k, n_l are the number of units in these hidden layers, $U^{l,c_{12}} \in \mathbb{R}^{n_{l-1} \times n_l}$ are the lateral connections from layer $l-1$ of column 1 to layer l of column 2, a^{0,c_k} is the network input for column c_k, and σ is an element-wise non-linearity (ReLU).

Additionally, we introduce another version of ProgNet, which we call "ProgNet2" where the lateral connections from previous column layers are initialized to be the same as the feedforward connections between the corresponding column layers ($U^{l,c_{12}} = W^{l,c_1}$).

A drawback of these approaches is the growth in the number of parameters with the number of tasks, as we detail below.

Elastic Weight Consolidation (EWC). EWC (Kirkpatrick et al., 2017) averts forgetting of old tasks by constraining learning on the weights important for those tasks (or contexts). The importance of parameters for a particular task (context) is quantified by the diagonal of the Fisher information matrix. The important weights stay close to their old values, keeping the parameters in a region of low error for context 1, centered around W^{c_1} (the weights for context 1) while learning context 2. The crux of EWC lies in the loss function used for training, which contains an additional regularization term keeping weights close to their old values:

$$L(W^*) = L_2(W^*) + \lambda/2 \sum_i F_i (W_i^* - W_i^{c_1})^2 \tag{8}$$

where L is the loss function that prevents catastrophic forgetting (CF), L_2 is the loss functions for context 2 without considering CF, λ sets how important the old task is compared with the new one, i labels each parameter, F_i is the ith entry of the Fisher information matrix diagonal corresponding to parameter W_i, $W_i^{c_1}$ is the solution found for the ith parameter when learning context 1, W_i^* is the solution for the ith parameter found by optimizing Eq. (8) as in Kirkpatrick et al. (2017).

Architecture comparison. Our network architecture, the switching network, is most similar to ProgNet, an architecture that similarly adjoins units with each context learned. A few important differences between these architectures stand out. First, the operation we implement in the switching network constrains it to use the input representation pre-nonlinearity at that layer and transform it via the addition of a low-rank modification of this representation (Eq. (5)). Second, our findings with respect to the small number of switching units required suggest that there are fewer weights to learn, i.e. fewer parameters to learn for $V_1^{l_1}, V_2^{l_1}, V_1^{l_2}, V_2^{l_2}$ ($V_{1,2}^{l_{1,2}}$ for short), than for $W^{l,c_2}, U^{l,c_{12}}$, which can imply fewer training iterations required to train our switching network. An explicit parameter count is presented below. The EWC, in contrast does not add any extra units to the network, instead relying on a clever regularization strategy as shown in (Eq. (8)). The switching network has an intermediate number of parameters to learn, provided there are indeed a very few number of switching units to be added for classification on context 2 (i.e. fewer weights to learn for $V_{1,2}^{l_1}$ than for $W^{l,*}$).

Performance comparison. We compare the performance of the bottleneck-switching network with the performance of ProgNet and EWC. All networks have the same architecture in common as the main network (Fig. 1B) of our switching network. The networks we compare against are: the main network implementing EWC during learning ("EWC network"); a ProgNet with two columns like the main network and lateral connections ("ProgNet"); a network like ProgNet, save for how lateral connections are initialized ("ProgNet2"); a main network with two separate last hidden layers for each context.

We first apply training as before, by first training/testing the basic NN on context 1, the *matched condition* (Fig. 4B). For EWC, we train on this network using the cross-entropy loss function L_1 that does not include the regularization term. When networks are trained on the *matched condition* for MNIST+noise, they perform equally well since initially all of them use the same architecture, the basic NN. These plot lines (except for the dark green line corresponding to EWC post-dataset 2 training, which will be discussed in the next paragraph) largely overlap, but are plotted with larger spacing, for clarity. At the next step, we train on context 2 by activating the switching units (purple line for the bottleneck-switching network); implementing a new column and learning weights from the column trained on context 1 (blue, orange lines for the ProgNets); learning using the basic NN as before using the loss function L_2 as shown in Eq. (8) (green line for EWC); training separate connections from the last hidden layer (yellow line). We show the corresponding accuracies on context 2 in Fig. 4C. As before, plot lines corresponding to ProgNet, ProgNet2, and the EWC network largely overlap, but are plotted with slightly larger spacing, for clarity.

The switching network and both ProgNets avoid catastrophic forgetting by design, but that is not the case for the EWC network (dark green line, Fig. 4B). Varying λ, the trade-off hyperparameter that balances how well the network performs on context 2 versus how close the

weights for context 1 and 2 are, we were unable to find a regime for EWC where both accuracies for the *test CF condition* and the *re-matched condition* were high. We conclude that, at least for the specific architecture and contexts studied here, EWC cannot perform sequential context switching, as Fig. 4B shows catastrophic forgetting for EWC (dark green line). A possible explanation for this problem is the high complexity of the task compared to the complexity of the architecture used, which leads to a large fraction of the weights being essential for the first context, leaving few unimportant weights to learn the second context.

When testing on context 2 (e.g., MNIST+cifar), we find that the switching network performs slightly better than the ProgNets across all transparencies, with statistically significant results (Fig. 4C, see Supplementary Material sec. 1.2.3. for exact details on statistical significance). This is unexpected, as Prognets could achieve *matched condition* performance by training the second column on context 2 and setting all lateral connections to 0. However, the switching network evaluated in Fig. 4C surpasses *matched condition* performance. It is possible that the ProgNets are over-parametrized and suffer from overfitting, or that training settles into a local minimum. The switching network also performs significantly better compared to a network where only the last layer has been trained on context 2, while the previous layer weights are constant, set to the weights for an NN trained on context 1 (yellow line, Fig. 4C, $p < 0.05$, one-sided t-test).

Furthermore, bottleneck-switching networks are learning faster than the ProgNets on transparencies ≤ 0.85 ($p < 7e-8$ and $p < 4e-7$ on MNIST+cifar, for ProgNet and ProgNet2, respectively, one-sided t-test; see Fig. 4D), when considering the number of iterations to reach 90% of the peak accuracy for that transparency, maximized between the switching network, ProgNet, ProgNet2, and EWC. Several data points are not shown for ProgNet, ProgNet2, and EWC because the respective network does not reach 90% of this peak accuracy. Alternatively, the switching network is the fastest on average for all $T \leq 0.66$, if we consider the number of iterations to reach 90% converging accuracy for each method independently (Supplementary Figure 8D-E). The comparatively rapid increase to peak accuracy could be explained because the switching network has fewer weights to learn than the ProgNet, in addition to taking advantage of the features learned from context 1. For learning two tasks, we are using 10 switching units for each layer (sw.u/u.s., with 3×3 kernels for the convolutional layers), and so we add an additional $10{\times}3{\times}3{\times}2{+}10{\times}2 = 200$ parameters to learn, compared to $O(100,000)$ for ProgNets. This over-parametrization might be the reason that for higher transparencies T these networks never reach the accuracy levels of the switching network, as the learning procedure could be stuck in a local minima. As the number of contexts (tasks) increases, the number of parameters for ProgNet remains much higher than for the switching network, even if we add switching units for every new context (task) (in some cases, adding new switching units is not necessary — see Section 3.8, Bottleneck-switching network for contexts of different statistics — however a careful analysis of the generality of the switching units is beyond the scope of this study). In conclusion, for most transparencies, our switching network shows enhanced performance at sequential context switching in terms of combined accuracy and learning speed (Fig. 4, Supplementary Figure 8A-I; see Supplementary Material Sec. 1.2.3. for detailed statistical tests).

3.6. Analysis of a switching network mechanism for sequential context switching

Having established the bottleneck-switching network as a suitable architecture for sequential context switching, we ask what mechanisms underlie its performance. We first observe that switching units in the first layer have a predominantly negative effect — that is, reducing the activations of the main network, as seen from the histograms of contributions for switching to MNIST+noise and MNIST+cifar (Fig. 5A). These predominantly negative weight contributions from switching units were not built into the network but emerged during learning of the second context.

We next make an allied observation about network representations. The negative contributions from switching units favor a sparsification of activations (after applying the ReLU non-linearity). Specifically, the activations after switching are much more sparse than those without switching ($p < 2e-6$ for MNIST+cifar, one-sided t-test; Fig. 5B and Supplementary Figure 10A), and we hypothesize that, at least for lower transparencies, this allows the network to highlight features useful for digit classification while inhibiting redundant features from the background (Fig. 5C, Supplementary Figure 10B). The validity of this hypothesis for lower transparencies is reinforced by evidence that the switching units inhibit a larger percentage of the background than of the digit ($p < 0.07$ on MNIST+cifar, one-sided t-test; Fig. 5D, Supplementary Figure 10C), indicating that the background features are suppressed preferentially. Additionally, when we constrain the weights to and from the switching units to be positive we obtain lower accuracy than when these weights are strictly negative (Supplementary Figure 9B,C). See Supplementary Material Sec. 1.2.3. for a detailed summary of these results and the statistical tests applied.

Furthermore, when we shift MNIST digits several pixels to the right without changing the background, we find an analogous result. Within the basic NN with two hidden layers, we train weights to the last layers (second hidden layer and output layer), while keeping weights to the first hidden layer and the corresponding weights to and from the switching units constant. We find that an analogous sparsification of digits appears in the first hidden layer (Supplementary Figure 11A-C), driven by the switching units that inhibit the background despite the shift in the position of the digits. Accuracy in this input regime without training the first hidden layer and the switching unit connectivities is high (Supplementary Figure 11D).

We next test alternative (simpler) mechanisms that could enable switching between contexts. For instance, a sparsification of the activities by using L1 regularization during training may be sufficient for sequential context switching. Using L1 regularization, the testing accuracy for context 1 is close to the *matched condition* accuracy (Supplementary Figure 12A). We hypothesize that L1 regularization may enable background inhibition, analogous to the behavior of the switching units. However, testing on context 2, we see that context 2 has substantially decreased performance compared to the *re-matched condition* using the bottleneck-switching network (Supplementary Figure 12B). This conclusion still holds as we vary the hyperparameter λ that controls how much the L1 norm is weighed. Alternatively, after training on both context 1 (e.g., MNIST+noise) and on context 2 (MNIST+cifar) using L1 regularization, we may test context 1 for the catastrophic forgetting phenomenon (*test CF condition*). As shown in Supplementary Figure 12C, this strategy is not effective in preventing catastrophic forgetting, given lower accuracies for the *test CF condition* (green line). Additionally, accuracies for the *re-matched condition* testing MNIST+cifar after L1 regularization (green line, Supplementary Figure 12D) are also lower than accuracies for the *re-matched condition* using the bottleneck-switching network.

We further examine the switching unit contributions from the simpler, feedforward switching units (Section 3.3). Interestingly, we find that the distribution of switching unit contributions is similarly left skewed, with weights on average inhibitory (Supplementary Figure 12E). However, upon closer examination, we find that the first hidden layer activations for the simple network from Section 3.3 are not sparsified, with the background inhibited similarly as the bottleneck-switching network that has recurrent switching units (Supplementary Figure 12F). This reinforces our finding that only by using appropriately, recurrently connected switching units, can switching networks be capable of sequential context switching by inhibiting the redundant features from the background.

3.7. Bottleneck-switching network for digit class and background pairs not used in training

As described above, switching unit contributions inhibit the background whenever digits set in context 2 are shown to the switching network, boosting the accuracy of the network in the new context. Can switching boost accuracy when the weights to and from the switching units have not been trained on specific digit-background pairs?

To investigate this problem, we choose a random pair of digits (e.g., 8,9) that will not be part of the training of the switching unit weights, then train the switching network in the manner shown in Fig. 6A. First, we train the main network (with the switching units OFF) in context 1 (e.g., MNIST+noise), using all digits 0-9. Then, we train the switching network (with switching units ON) on the second context (e.g., MNIST+cifar), using the digits not included in the preselected pair (e.g., 0–7). Finally, we employ binary classification on the chosen pair of digits set on context 2 and compare the performance of the switching network with the switching units turned ON, with the performance when switching units are OFF (Fig. 6B, Supplementary Figure 13). Presumably, the switching unit contributions will inhibit the background, but is training on digit identity necessary when learning weights to and from the switching units?

We find that, on average, switching units' contribution increases accuracy even for digit-background pairs never-before seen by the switching network, i.e. digit-background pairs for which the weights to and from the switching units have not been trained (Fig. 6B, Supplementary Figure 13). This is shown for select pairs of digits (Supplementary Figure 13), and when averaging over ten randomly chosen pairs of digits (Fig. 6B). We conclude that the bottleneck-switching network is capable of generalizing to digit-background pairs it has not trained on, given that the main network and the switching units have been separately trained on the digits and the background, respectively. This is a compelling generalization property of the switching network, showing how classification in different contexts can arise synergistically. More work is needed to infer what the minimal training set for the switching unit weights is that still improves performance for sequential context switching.

3.8. Bottleneck-switching network for contexts of different statistics

More generally, we would like to investigate switching between contexts of any statistics. So far, we have seen how a switching network can be trained to switch between MNIST+noise (a pixel-wise noisy context) and MNIST+cifar (a realistic and non-parametrized context). To study the more general problem, we parametrize contexts using β, the spectral distribution, such that the spectral density is $S(f) = N f^{\beta}$, where f is the frequency, and N is the normalization coefficient. With this parametrization, $\beta = 0$ denotes a white noise context; $\beta = -1$ denotes a pink noise context; $\beta = -2$ denotes a Brownian noise context; $\beta = 1$ represents a blue noise context; $\beta = 2$ denotes a violet noise context. More details on how these contexts are generated can be found in Supplementary Materials, Sec. 1.2.5.

We super-impose MNIST digits on backgrounds of different statistics to obtain different datasets representing different contexts (Fig. 6C). We refer to these datasets as "MNIST + β" (or "MNIST + β" - parametrized contexts) for different values of β. Using these datasets, we seek to investigate two main questions. First, given contexts of different statistics, how well can the switching units that are trained on a particular statistics help – without further training – with the classification of a context with different statistics? Second, are some contexts "closer" in some sense to others (with respect to a certain distance or metric), hence enabling the switch to readily generalize to different statistics? If so, is this distance symmetric?[1]

To answer these questions, we first train the bottleneck-switching network using a familiar approach: the main network is trained on MNIST + cifar with the switching units OFF; we then turn the switching units ON, and train on MNIST+noise while keeping the weights of the main network constant. Using this trained network, we can now test its performance on the MNIST+β-parametrized contexts. With the switching units ON, we assess the accuracy of the network to identify which dataset of parametrized contexts can utilize the switching units best to achieve digit classification. We find there is maximum accuracy when $\beta = 0$, with generally high accuracies for $\beta \geq 0$ (Supplementary Figure 14A). This occurs because MNIST+noise corresponds to an MNIST+β-parametrized context with $\beta = 0$. While the peak at $\beta = 0$ is explainable, the high performance for $\beta > 0$ demonstrates an anti-symmetry between the capacity of the switching network to generalize well to particular contextual statistics, while not to others. We conclude that switching networks whose weights have been trained on MNIST+noise $\equiv$ MNIST + $\beta = 0$ (images with pixel-wise noisy contexts) generalize well to higher frequency contexts ($\beta > 0$).

The same results hold when training the main network on MNIST+cifar, training the switching units on MNIST+β, and then testing the switching network on MNIST+noise. As before, we find there is maximum accuracy when $\beta = 0$. (Supplementary Figure 14B). Accuracies for contexts with $\beta \geq 0$ are high, with the accuracy plateauing for higher frequency contexts.

Finally, we would like to gain insight into the issue of distance in the space of contexts with different statistics, where contexts that exhibit the closest statistics may benefit most from the same switching units (or from the same low-rank perturbation to the weights). To investigate this problem, we use the following training procedure: we train the main network on MNIST+cifar (with switching units OFF); we then turn the switching units ON and train the switching weights on a context with some fixed β, keeping weights in the main network fixed; lastly, we test different MNIST+β-parametrized contexts by varying β. We can find generalization accuracies for different pairs of β corresponding to pairs of contexts used for training and testing the switch, respectively. The matrix obtained (Fig. 6D) shows how efficient the switch is across different contexts (for $T = 0.0, 0.5$) and can be interpreted as the inverse of the distance between two contexts: the higher the accuracy is, the better we can classify utilizing the same set of switching weights, suggesting that these contexts are "close" in some sense.

We find a bimodal behavior of the switching network: when contexts corresponding to (β_1, β_2)-pairs are trained on the network, we obtain high testing accuracies when both β_1, β_2 have higher or lower frequencies, i.e. $\beta_1, \beta_2 \leq -3$ or $\beta_1, \beta_2 \geq -2$. When $\beta_1 \geq -2$ and $\beta_2 \leq -3$ (or vice versa), the generalization accuracy is poor. This is particularly evident for higher transparencies (Fig. 6D, $T = 0.5$).

The bimodal behavior of the bottleneck-switching network suggests that using two switches may be sufficient to effectively classify all MNIST+β contexts. We define a "switch" in this particular framework to be a set of switching units with connections to and from the main network. These connections are learned by training on a particular MNIST+β dataset. We ask how many of these "switches" we would need to use to accurately classify every MNIST+β dataset. In the worst case scenario, we would need a switch for every MNIST+β context. In the best case scenario, we would need only one switch that can work across all contexts. To test these alternatives, we consider the accuracies shown in Fig. 6D and first try to find one β such that the average accuracies across all other β values is maximized. This represents the

[1] Viewing the switching unit contribution as essentially a low-rank perturbation to the weights (at least to a linear approximation), an interesting problem is whether there is a metric in the space of contexts, that can determine whether low-rank perturbations to the weights of the main network enable switching contexts with different statistics while maintaining the task. Additionally, if two contextual statistics are "close enough", we hypothesize that the low-rank perturbation is capable of generalizing to enable classification in a similar context.

accuracy when using only one switch. We next look for two β values such that switches trained on the corresponding β contexts would be most effective in classifying all the different MNIST+β datasets. Finally, we consider accuracies on the diagonal of matrices such as those shown in Fig. 6D. These represent the accuracies given a switch for each of the MNIST+β contexts (i.e. one switch for every β).

We find that having only two switches for the eleven contexts, with β values spanning an interval from -100 to 3, is sufficient to give an accuracy close to that of the upper bound (Fig. 6E). The upper bound represents the accuracy when every MNIST+β context is trained and tested on a separate switch. Having a switch for every context may give the highest accuracy, but would indeed be quite impractical and very costly in terms of training time. The accuracies when using two switches are significantly higher then when using only one switch ($p < 3e-4$, one sided t-test). More details on the statistical significance of these results can be found in the Supplementary Material, Sec. 1.2.3.

Different results hold when training a feedforward network without switching units in the following way: we train the basic NN we have so far employed on an MNIST+ β-parametrized context, and then test on a different MNIST+β-parametrized context. We found distinct conditions for satisfactory generalization: it is best, whichever context the network was trained on, to test on contexts with higher-frequency statistics. It is also satisfactory to first train in low-frequency contexts. The instance most prone to (generalization) error is for training on higher frequency contexts (positive β) and testing on low-frequency contexts (negative β), as seen in Supplementary Figure 14D.

4. Conclusion and future work

We study a biologically inspired switching network that is capable of domain adaptation between backgrounds (contexts) with different statistical properties, while the basic task structure is maintained. We construct and share a parametrized dataset to test this adaptation, by overlaying MNIST digits with different transparencies on noisy or CIFAR-10 backgrounds. The switching network can leverage features learned in one context and use them for digit classification in a second context, without forgetting classification in the first context. Furthermore, the network has a higher performance classifying the second context than a network trained only on the second context. We refer to this network as the bottleneck-switching network because it has only relatively few switching units that are recurrently connected to the main layers. This structure compresses network activations through a bottleneck, then relays this compressed information back to the main network. However, it assumes context is either known a priori to the network, or can be trivially inferred.

While we do not claim to have identified a precise mechanism for how the switching allows the network to perform well under both contexts with so few neurons, a set of analyses allows us to speculate. We observe a sparsification of the NN representation with the switch ON (Fig. 5B,C). We believe this allows the network to select the features relevant for the task; these features are different from the features in the new background, which serve as distractors. We show that this can be achieved with relatively few switching neurons. We also show that a low-rank change in the connectivity matrix can result in a similar performance (Fig. 3G, Supplementary Figure 9A). We connect these findings by noting that a small number of neurons recurrently connected can provide the mechanism for a low-rank matrix change in network connectivity.

We also note similarities and differences to the biology of the VIP circuitry in the visual cortex. We find that the small number of switching units in the first NN layer are inhibitory towards the hidden layer activations of the main network. While this invites direct comparison to the underlying biology, as VIP neurons are inhibitory, our framework omits the disinhibitory motif, and specific inhibitory/excitatory neuronal populations. Specifically, while the switching units roughly reproduce the behavior of the VIP neurons by turning ON/OFF contingent on the context, and by recurrently interacting with other network units to modulate network activity, these added units are not constrained to have any sign in particular. Moreover, the inhibitory dominance is not apparent in the second layer, although we note that the switching units at the first convolutional layer are the main drivers of high performance (Supplementary Figure 7D-F). In sum, we do not claim to have implemented in this paper a precise model of the VIP circuitry, but rather to have used a bio-inspired circuit motif with VIP-like, bottlenecked, and recurrent connectivity.

Future research will be able to improve on our current network architecture. First, we opted for a simple benchmark dataset (MNIST digits) to remove other potential confounds, but future work should concentrate on more complex datasets. Second, while we found high performance with relatively low numbers of switching units, especially for the VGG network, future research may be able to find theoretical guarantees on the number of switching units sufficient for sequential context switching. Moreover, beyond our main studies with a pair of contexts (noise and CIFAR-10), we have studied the transition between multiple contexts using only a simple continuously parametrized noise setting (the MNIST+β contexts). More work is needed to address the problem of continuous contexts in general, although the suppression of the contextual features shown above opens the possibility that multiple context switches can be incorporated. Finally, we showed that a switching network can readily include a downstream module to perform binary classification of contexts using supervised learning, with *a priori* training to different backgrounds (Fig. 3I, Supplementary Figure 15). The binary output is then used by the VIP neurons to turn them ON and OFF in a switch-like manner, automating the sequential context switching task. However, a more general framework would enable the module to detect a novel context in an unsupervised way, and this is another important avenue to explore in future work. Finally, just like the brain is believed to implement various strategies in order to perform multiple tasks, our switching units may be combined with other methods, for instance regularization-based methods like Kirkpatrick et al. (2017), similar to what is shown in Masse et al. (2018) for the context-dependent gating.

Our study has commonalities, but also a few key differences with the work of Masse et al. (2018) that implements context-dependent gating (XdG). Importantly, both our architecture and theirs use a context-dependent signal that is either known *a priori* or is computed by a separate module. As Masse et al. correctly point out, these context-dependent signals likely originate from areas such as the prefrontal cortex, and project to various cortical areas in order to allow neural circuits to process incoming information in a task-dependent manner. While Masse et al. (2018) test their network on classification tasks which constitute permuted sets of MNIST digits or sets of ImageNet classes, and additionally on cognitive tasks similar to the ones used in neuroscience experiments, we focus on the problem of sequential context (or background) switching, which they do not address. In addition, XdG implements gating which silences large sets of neurons in each of the layers ($\approx 80\%$ of all the units), while our bottleneck-switching network is not constrained to sparsify the hidden layers, but rather learns to find sparse solutions where the unimportant (background-related) features are inhibited. Another important difference is that our architecture is a form of transfer learning, where the weights learned for context 1 are used when learning the new context, therefore our performance for context 2 is higher than that of a network trained only on context 2. This is distinct from Masse et al. (2018) where largely non-overlapping sets of units are activated for each task.

In principle, bottleneck-switching networks employ a general circuit motif that implements a context-dependent computation, therefore an advantage is that switching units could be integrated into any neural network in order to address the problem of adaptation to context. The switching units can also be combined with other complementary learning mechanisms that improve continual learning capabilities, such as Kirkpatrick et al. (2017), Zenke, Poole et al. (2017). A possible

application area of high social importance is in addressing biases of background, in which networks can learn unwanted correlations between context and objects of interest (de Vries et al., 2019).

Finally, we note that this is one of few studies to address generalization across context in a bio-inspired architecture while focusing specifically on background variation and removing other confounds (Beery et al., 2018). In future years, we look forward to progress in understanding the context-dependent computations that enable the extraordinary versatility of biological agents to adapt and switch environments so effortlessly. Deciphering the general principles behind contextual generalization will enable our current networks to develop from being sophisticated "pattern-matching machines" to more intelligent, human-like learners capable of abstracting visual concepts (Beery et al., 2018).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared a link to my data in the manuscript and the Supplementary Material.

Acknowledgments

This work was supported by the Swartz Foundation Center for Theoretical Neuroscience at the University of Washington; and the National Institutes of Health [grant number 5 R90 DA 033461-08]. We thank the Allen Institute for Brain Science founder, Paul G. Allen, for his vision, encouragement, and support.

Funding sources have no direct involvement in the study design, in the analysis, interpretation of results, or in the writing of the manuscript.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.neunet.2023.09.014. Supplemental Materials are included with the main manuscript. A repository with the code to generate the figures in the main paper can be found at https://github.com/dvoina13/switching_network_ANN.git.

The MNIST+noise/cifar datsets can be found in Voina, Shea-Brown, and Mihalas (2021).

References

Abraham, W., & Robins, A. (2005). Memory retention and weight plasticity in ANN simulations. *Trends in Neurosciences*, *28*(2), 73–78.

Amir, R., Zemel, R., & Tsotsos, J. (2018). The elephant in the room.

Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., et al. (2019). ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Vol. 32*, In *Advances in neural information processing systems* (pp. 9448–9458).

Bau, D., Liu, S., Wang, T., Zhu, J.-Y., & Torralba, A. (2020). Rewriting a deep generative model. arXiv:2007.15646.

Beery, S., Horn, G. V., & Perona, P. (2018). Recognition in terra incognita. In *European conference on computer vision (ECCV)* (pp. 472–489).

Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., & Erhan, D. (2016). Domain separation networks. In *Advances in neural information processing systems* (pp. 343–351).

Bruzzone, L., & Marconcini, M. (2010). Domain adaptation problems: A dasvm classification technique and a circular validation strategy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(5), 770–787.

Chattopadhyay, R., Ye, J., Panchanathan, S., Fan, W., & Davidson, I. (2011). Multi-source domain adaptation and its application to early detection of fatigue. *Vol. 6*, In *Proc. KDD* (4), (pp. 717–725).

Choi, M., Torralba, A., & Willsky, A. (2012). Context models and out-of-context objects. *Pattern Recognition Letters*, *33*(7), 853–862.

Chu, W., de la Torre, F., & Cohn, J. (2013). Selective transfer machine for personalized facial action unit detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3515–3522).

Csurka, G. (2017). Domain adaptation for visual applications: A comprehensive survey. In G. Csurka (Ed.), *Domain adaptation in computer vision applications. Advances in computer vision and pattern recognition*. Springer, Cham.

Daume, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of ACL* (pp. 256–263).

de Vries, T., Misra, I., Wang, C., & van der Maaten, L. (2019). Does object recognition work for everyone? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR) workshops* (pp. 52–59).

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., et al. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. *32*, In *ICML: Proceedings of the 31st international conference on international conference on machine learning*.

Draelos, T., Miner, N., Lamb, C., Vineyard, C., Carlson, K., James, C., et al. (2017). Neurogenesis deep learning. In *International joint conference on neural networks (IJCNN)* (pp. 526–533).

Duan, L., Tsang, I., & Xu, D. (2012). Domain transfer multiple kernel learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 465–479.

Duan, L., Xu, D., & Chang, S. (2012). Exploiting web images for event recognition in consumer videos: a multiple source domain adaptation approach. In *IEEE 2012 conference on computer vision and pattern recognition* (pp. 1338–1345).

Ellefsen, K. O., Mouret, J.-B., & Clune, J. (2015). Neural modularity helps organisms evolve to learn new skills without forgetting old skills. *PLoS Computational Biology*, *11*(4), 1–24. http://dx.doi.org/10.1371/journal.pcbi.1004128.

French, R. (1997). Pseudo-recurrent connectionist networks: An approach to the sensitivity-stability dilemma. *Connection Science*, *9*(4), 353–380.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. *Vol. 27*, In *Proceedings of the 28th international conference on machine learning* (pp. 97–110).

Gong, B., Grauman, K., & Sha, F. (2013). Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International conference on machine learning* (pp. 222–230).

Henning, C., Cervera, M. R., D'Angelo, F., von Oswald, J., Traber, R., Ehret, B., et al. (2021). Posterior meta-replay for continual learning. arXiv:2103.01133.

Hinton, G., & Plaut, D. (1987). Using fast weights to deblur old memories. In *Proceedings of the annual conference of the cognitive science society* (pp. 177–186).

Hofmanninger, J., Perkonigg, M., Brink, J. A., Pianykh, O., Herold, C., & Langs, G. (2020). Dynamic memory to alleviate catastrophic forgetting in continuous learning settings. arXiv:2007.02639.

Jung, H., Ju, J., Jung, M., & Kim, J. (2016). Less-forgetting learning in deep neural networks.

Kaiser, L., Gomez, A., Shazeer, N., Vaswani, A., Parmar, N., Jones, L., et al. (2017). One model to learn them all.

Kemker, R., & Kanan, C. (2018). Fearnet: Brain-inspired model for incremental learning. In *International conference on learning representations (ICLR)*.

Kim, J., Yoo, S.-M., Park, G.-M., & Kim, J.-H. (2021). Continual unsupervised domain adaptation for semantic segmentation. arXiv:2010.09236.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, *114*(13), 3521–3526.

Kumaran, D., Hassabis, D., & McClelland, J. (2016). What learning systems do intelligent agents need? Complementary learning systems theory updated. *Trends in Cognitive Sciences*, *20*(7), 512–534.

Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., et al. (2018). The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (2013). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Li, Z., & Hoiem, D. (2016). Learning without forgetting. In *ECCV 2016: Computer vision* (pp. 614–629).

Li, F., Pan, S., Jin, O., Yang, Q., & Zhu, X. (2012). Cross-domain co-extraction of sentiment and topic lexicons. In *Proceedings of the 50th annual meeting of the association for computational linguistics long papers* (pp. 410–419).

Li, Y., Wang, N., Shi, J., Liu, J., & Hou, X. (2016). Revisiting batch normalization for practical domain adaptation.

Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft COCO: Common objects in context. In *European conference on computer vision (ECCV)*.

Liu, M., & Tuzel, O. (2016). Coupled generative adversarial networks. In *Advances in neural information processing systems* (pp. 469–477).

Long, M., Wang, J., Ding, G., Sun, J., & Yu, P. (2013). Transfer feature learning with joint distribution adaptation. In *Proceedings of the 2013 IEEE international conference on computer vision* (pp. 2200–2207).

Long, M., Wang, J., & Jordan, M. (2016). Deep transfer learning with joint adaptation networks.

Mallya, A., Davis, D., & Lazebnik, S. (2018). Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *ECCV 2018: Computer vision* (pp. 72–88).

Mallya, A., & Lazebnik, S. (2018). PackNet: Adding multiple tasks to a single network by iterative pruning. In *IEEE/CVF conference on computer vision and pattern recognition* (pp. 7765–7773).

Mante, V., Sussillo, D., Shenoy, K., & Newsome, W. (2013). Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, *503*, 78–84.

Masse, N. Y., Grant, G. D., & Freedman, D. J. (2018). Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, *115*(44), http://dx.doi.org/10.1073/pnas.1803839115.

McClelland, J., McNaughton, B., & O'Reilly, R. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, *102*(419–457).

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, *24*(C), 109–165.

Misra, I., Shrivastava, A., Gupta, A., & Hebert, M. (2016). Cross-stitch networks for multi-task learning. In *IEEE conference on computer vision and pattern recognition (CVPR)*.

Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *IEEE conference on computer vision and pattern recognition* (pp. 1717–1724).

Pan, S., Tsang, I., Kwok, J., & Yang, Q. (2009). Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, *22*(2), 199–210.

Parisi, G., Kemker, R., Part, J., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, *113*, 57–71.

Ratcliff, R. (1990). Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*, *97*(2), 285–308.

Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off-the-shelf: An astounding baseline for recognition. In *IEEE conference on computer vision and pattern recognition workshops* (pp. 512–519).

Rebuffi, S., Kolesnikov, A., Sperl, G., & Lampert, C. (2016). iCaRL: Incremental classifier and representation learning.

Rostami, M. (2021). Lifelong domain adaptation via consolidated internal distribution. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Vol. 34*, *Advances in neural information processing systems* (pp. 11172–11183). Curran Associates, Inc., URL https://proceedings.neurips.cc/paper_files/paper/2021/file/5caf41d62364d5b41a893adc1a9dd5d4-Paper.pdf.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211–252.

Rusu, A., Rabinowitz, N., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., et al. (2016). Progressive neural networks.

Sagawa, S., Pang, W., Tatsunori, H., & Percy, L. (2020). Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International conference on learning representations*.

Shetty, R., Fritz, M., & Schiele, B. (2018). Adversarial scene editing: Automatic object removal from weak supervision. In *Neural information processing systems (NeurIPS)*.

Shi, Y., & Sha, F. (2012). Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. In *Proceedings of the 29th international conference on machine learning* (pp. 1–8).

Shin, H., Lee, J., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. In *NIPS: Proceedings of the 31st international conference on neural information processing systems* (pp. 2994–3003).

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. CoRR arXiv:1409.1556.

Su, P., Tang, S., Gao, P., Qiu, D., Zhao, N., & Wang, X. (2020). Gradient regularized contrastive learning for continual domain adaptation. arXiv:2007.12942.

Sun, B., & Saenko, K. (2016). Deep coral: Correlation alignment for deep domain adaptation. In *Computer vision–ECCV 2016 workshops* (pp. 443–450). Springer.

Swaminathan, S., Garg, D., Kannan, R., & Andres, F. (2020). Sparse low rank factorization for deep neural network compression. *Neurocomputing*, *398*, 185–196. http://dx.doi.org/10.1016/j.neucom.2020.02.035, URL https://www.sciencedirect.com/science/article/pii/S0925231220302253.

Taufique, A. M. N., Jahan, C. S., & Savakis, A. (2022). Unsupervised continual learning for gradually varying domains. In *2022 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW)* (pp. 3739–3749). http://dx.doi.org/10.1109/CVPRW56347.2022.00418.

Tommasi, T., Orabona, F., & Caputo, B. (2010). Safety in numbers: learning categories from few examples with multi model knowledge transfer. In *IEEE conf comput vision pattern recog.* (pp. 3081–3088).

Torralba, A., & Efros, A. (2011). Unbiased look at dataset bias. *CVPR*, 1521–1528.

van de Ven, G. M., & Tolias, A. S. (2019). Three scenarios for continual learning. arXiv:1904.07734.

Voina, D., Recanatesi, S., Hu, B., Shea-Brown, E., & Mihalas, S. (2022). Single circuit in V1 capable of switching contexts during movement using VIP population as a switch. *Neural Computation*, *34*(3), 541–594.

Voina, D., Shea-Brown, E., & Mihalas, S. (2021). mnist+context. URL https://figshare.com/articles/dataset/mnist_context/14703639.

Volpi, R., Larlus, D., & Rogez, G. (2021). Continual adaptation of visual representations via domain randomization and meta-learning. arXiv:2012.04324.

Wang, Q., Fink, O., Gool, L. V., & Dai, D. (2022). Continual test-time domain adaptation. arXiv:2203.13591.

Wang, A., Narayanan, A., & Russakovsky, O. (2020). REVISE: A tool for measuring and mitigating bias in visual datasets. In *European conference on computer vision (ECCV)*.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. D. (2016). A survey of transfer learning. *Journal of Big Data*, *3*(1), 9.

Weng, M., & Deng, W. (2018). Deep visual domain adaptation: A survey. *Neurocomputing*, *312*, 135–153.

Xia, R., Zong, C., Hu, X., & Cambria, E. (2013). Feature ensemble plus sample selection: domain adaptation for sentiment classification. *IEEE Intelligent Systems*, *28*(3), 10–18.

Xiao, K., Engstrom, L., Ilyas, A., & Madry, A. (2020). Noise or signal: The role of image backgrounds in object recognition.

Xiao, T., Li, H., Ouyang, W., & Wang, X. (2016). Learning deep feature representations with domain guided dropout for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1249–1258).

Xiao, T., Zhang, J., Yang, K., Peng, Y., & Zhang, Z. (2014). Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Proceedings of the ACM international conference on multimedia* (pp. 177–186).

Yao, Y., & Doretto, G. (2010). Boosting for transfer learning with multiple sources. In *Proceedings of the IEEE computer society conference on computer vision and pattern recognition* (pp. 1855–1862).

Yoon, J., Yang, E., Lee, J., & Hwang, S. (2018). Lifelong learning with dynamically expandable networks. In *International conference on learning representations (ICLR)*.

Yu, F., Tucciarone, J., Espinosa, J., Sheng, N., Darcy, D., Nicoll, R., et al. (2014). A cortical circuit for gain control by behavioral state. *Cell*, *156*(6), 1139–1152.

Zeng, G., Chen, Y., Cui, B., & Yu, S. (2019). Continuous learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, *1*, 364–372.

Zenke, F., Gerstner, W., & Ganguli, S. (2017). The temporal paradox of hebbian learning and homeostatic plasticity. *Vol. 43*, In *Neurobiology* (pp. 166–176).

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. *Vol. 70*, In *ICML: Proceedings of the 34th international conference on machine learning*.

Zhou, G., Sohn, K., & Lee, H. (2012). Online incremental feature learning with denoising autoencoders. In *International conference on artificial intelligence and statistics* (pp. 1453–1461).

Zhu, Z., Xie, L., & Yuille, A. (2017). Object recognition without and without objects. In *International joint conference on artificial intelligence*.