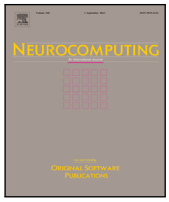
Neurocomputing 598 (2024) 127935

Contents lists available at ScienceDirect

Neurocomputing

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



Projected Latent Distillation for Data-Agnostic Consolidation in distributed continual learning

Antonio Carta [a,*], Andrea Cossu [a], Vincenzo Lomonaco [a], Davide Bacciu [a], Joost van de Weijer [b]

[a] Department of Computer Science, University of Pisa, Pisa, Italy
[b] Computer Vision Center, Barcelona, Spain

ARTICLE INFO

Communicated by S. He

Keywords:
Continual learning
Model consolidation
Distributed continual learning

ABSTRACT

In continual learning applications on-the-edge multiple *self-centered devices* (SCD) learn different local tasks independently, with each SCD only optimizing its own task. *Can we achieve (almost) zero-cost collaboration between different devices?* We formalize this problem as a *Distributed Continual Learning* (DCL) scenario, where SCDs greedily adapt to their own local tasks and a separate continual learning (CL) model perform a sparse and asynchronous consolidation step that combines the SCD models sequentially into a single multi-task model without using the original data. Unfortunately, current CL methods are not directly applicable to this scenario. We propose Data-Agnostic Consolidation (DAC), a novel double knowledge distillation method which performs distillation in the latent space via a novel Projected Latent Distillation loss. Experimental results show that DAC enables forward transfer between SCDs and reaches state-of-the-art accuracy on Split CIFAR100, CORe50 and Split TinyImageNet, both in single device and distributed CL scenarios. Somewhat surprisingly, a single out-of-distribution image is sufficient as the only source of data for DAC.

1. Introduction

In a typical continual learning application deployed on-the-edge, a machine learning model is continuously adapted with a nonstationary stream of data. Usually, multiple devices are deployed, resulting in a fleet of models which are trained in parallel and specialized at their own local task or domain. We call such devices *self-centered devices (SCDs)*, to highlight that (1) they greedily optimize their performance only over a specialized domain and (2) they do not want to waste model capacity or computational resources to learn other tasks. SCDs are "selfish actors" that do not want to share their private data or spend their limited computational budget to support other SCDs.

Given such a restricted setting, this paper seeks an answer to the question: *How can we allow collaboration between SCDs at (almost) zero cost for them?* To satisfy the strict SCD requirements, we train each SCD independently without any synchronization mechanism. Only two operations are allowed: a *model initialization* phase, where a new SCD asks for a model initialization (the consolidated model). The consolidated model is a multi-task model learned independently to provide positive forward transfer between the SCDs. Only once and after its local training is completed, the SCD sends the model to update the consolidated model during a *model consolidation* phase (Fig. 1).

In this paper, we formalize this problem as a *Distributed Continual Learning* (DCL) scenario. Training the consolidated model is a hard continual learning (CL) problem [1] that requires to incrementally update the consolidated model to learn the new task whenever the parameters of a new SCD models are received. Crucially, the consolidated model never sees the original data, which means that classic CL methods are not applicable to this setting. Consolidating models asynchronously is a difficult problem. For example, federated learning methods [2] require multiple rounds of frequent communication to work [3].

We propose a novel method for training the consolidated CL model, called *Data-Agnostic Consolidation (DAC)*, that performs double knowledge distillation from the previous consolidated model and current SCD model without using the original data. DAC consolidates the two models using a single out-of-distribution image and heavy augmentations [4,5]. As a consequence, DAC can iteratively update the consolidated model without ever looking at the original data. A limitation of naive double distillation is that it is not possible to perform feature distillation because the two teachers compute two different latent representations. DAC solves this problem via our novel *Projected Latent Distillation* loss. By using the same loss for new and old tasks, DAC easily balance stability and plasticity [6], resulting in good performance even for old tasks, something that most methods struggle with. The experimental results show that DAC consolidates the knowledge from independent SCD models even when they are trained on different tasks and only a single out-of-distribution image is available.

The main contributions of the paper can be summarized as follows:

* Corresponding author.
E-mail address: antonio.carta@unipi.it (A. Carta).

https://doi.org/10.1016/j.neucom.2024.127935
Received 7 September 2023; Received in revised form 18 December 2023; Accepted 27 May 2024
Available online 13 June 2024

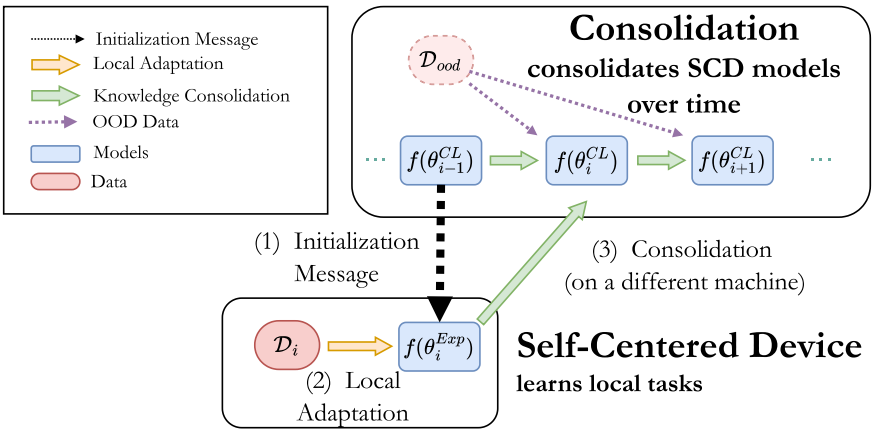


Fig. 1. In our Distributed Continual Learning scenario, whenever a self-centered device (bottom) wants to learn a single task, it (1) asks and receives a model initialization from the consolidated model (top), (2) performs a local adaptation step to learn the new task, and (3) sends the learned model to the cloud for the consolidation. This process allows to continually learn from SCDs that do not wish to dedicate resources to prevent forgetting, nor want to share their data.

- a formal characterization of distributed continual learning. To the best of our knowledge, this is the first work to formalize continual learning in a distributed setting and the consolidation problem (Section 2);
- *Data-Agnostic Consolidation*, a novel strategy which performs a data-agnostic double knowledge distillation in the output and latent space via Projected Latent Distillation (Section 3);
- state-of-the-art results in the classic rehearsal-free and the novel DCL scenario for task-aware SplitCIFAR100 (Table 1a, +3.9% on 10 Tasks), task-aware Split TinyImageNet, and task-agnostic CORe50-NI[1];
- we show that DAC allows positive forward transfer between the SCDs and that simple sources of data are sufficient for consolidation when coupled with heavy augmentations (Section 4).

2. Distributed continual learning

In Distributed Continual Learning (DCL) we have two types of models interacting with each other: *SCD models*, denoted as f^{SCD}, and the *consolidated* model, denoted as f^{CL}. SCD models learn a single task, while the CL model is a multi-task model learned by consolidating the SCD models sequentially. SCDs have four characteristics: (i) they want to train independently, (ii) they never share the data, (iii) they want minimal communication, and (iv) they do not care about learning or forgetting other tasks. In DCL we have two different learning problems. Knowledge adaptation is the local adaptation to the task solved by each SCD, while knowledge consolidation is the continual learning problem of consolidating the SCD models without access to the original data. More formally:

Knowledge adaptation. is an algorithm $\mathcal{A}^{ADA} : \langle f_{i-1}^0, \mathcal{D}_i^{train} \rangle \rightarrow \langle f_i^{SCD} \rangle$ that takes an initialization f_{i-1}^0 and a batch of data $\mathcal{D}_i^{train}$ and returns a model f_i^{SCD}. This is a simple finetuning procedure where only the local loss $\mathcal{L}(f_i^{SCD}, \mathcal{D}_i)$ is minimized. We call the resulting model f_i^{SCD} the self-centered model for task i.

Knowledge consolidation. is an algorithm $\mathcal{A}^{KC} : \langle f_{i-1}^{CL}, f_i^{SCD}, \mathcal{D}^{ood} \rangle \rightarrow \langle f_i^{CL} \rangle$ that takes as input two models, the previous CL model f_{i-1}^{CL} and the current SCD model f_i^{SCD}, and a small dataset $\mathcal{D}^{ood}$ and combines them into a single model f_i^{CL}. The resulting model is optimized to solve all the task encountered by the SCD models so far. The dataset $\mathcal{D}^{ood}$ used during consolidation is different from the original data used to train the SCD models.

[1] Available as supplementary material. The source code will be publicly released after acceptance.

To ensure a fixed memory occupation, we assume that a knowledge consolidation algorithm can store only two models at the same time, the current SCD model and the previous consolidation model.

Two minimal schemes: The sequential and independent settings

Communication between the consolidated and self-centered models, shown in Fig. 1, consists of two messages: the *initialization message*, sent from the consolidated model to the SCD before training, and the *consolidation message*, sent from the SCD to the consolidated model after training. Communication occurs only at the beginning and at the end of training for each SCD, allowing complete freedom during training. The initialization message provides a good initialization (i.e. positive transfer between SCDs), while the consolidation message updates the consolidated CL model with the new task.

The interaction between the SCD and CL model happens in this order: (1) Before training, the SCD receives the CL model to use as its initialization. (2) The SCD performs a *knowledge adaptation* step to learn its task. (3) After training, the SCD sends its trained model for the consolidation. (4) The CL model performs a *knowledge consolidation* step to consolidate the new model and the previous CL model, learning a new task.

Under this communication scheme, the CL model receives a sequence of expert models $f_0^{SCD}, \dots, f_i^{SCD}$ and aggregates them via a knowledge consolidation algorithm. Notice that the global model never receives the original data. During this interaction, the ordering between the messages arriving the SCDs is critical since it determines how much transfer there can be between the SCD models. We focus on two extremes of the range of possibilities: the *sequential* and *independent* scenario.

In the *sequential scenario*, SCD models are trained sequentially. At time i, the SCD i receives f_{i-1}^{CL} from the CL model. The next SCD $(i+1)$ will start training only after the consolidation of the SCD model f_i^{SCD}. This is the scenario where there is the maximum opportunity for transfer.

In the *independent scenario*, all the SCDs start training in parallel at time $t = 0$. Therefore, they receive the same random initialization f_0. In this setting, there is no transfer between the SCD models. We use this scenario to evaluate the consolidation method and to evaluate the impact of knowledge transfer between SCDs in the sequential setting (Section 4.2).

Relationship with classic continual and federated learning

Continual learning is the problem of learning from a non-stationary stream of experiences $S = e_1, \dots, e_n$. In classification problems, each experience e_i provides a batch of samples $D_i = \{\langle x_m, y_m, t_m \rangle\}$, where $x_m \in \mathbb{R}^X$ is the input, $y_m \in \mathcal{Y}_i$ the target label, and $t_m \in \mathbb{N}$ an optional

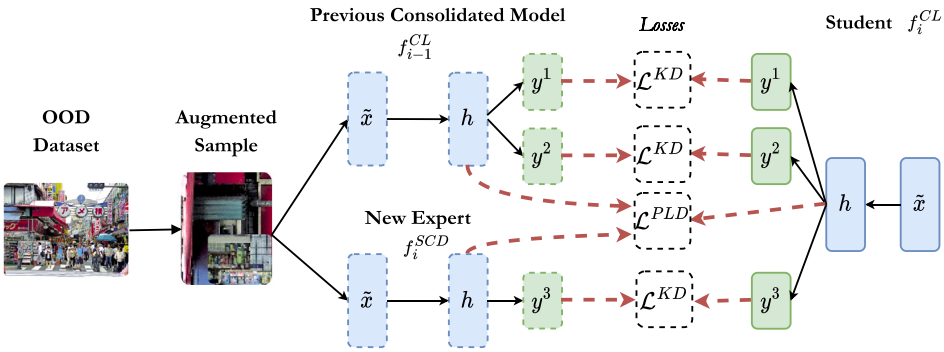


Fig. 2. Data-Agnostic Consolidation. The method uses double distillation in the output and latent space, using heavily augmented samples as input. Dashed green and blue blocks represent the teachers' modules, while solid blocks are the student's modules.

task label. A continual learning model f^{CL} learns on each experience sequentially. In the rehearsal-free setting at time i data $\mathcal{D}_j, j < i$ from previous experiences is not available [7,8]. A rehearsal-free continual learning algorithm $\mathcal{A}^{CL} : \langle f_{i-1}^{CL}, \mathcal{D}_i^{train} \rangle \rightarrow \langle f_i^{CL} \rangle$ takes as input the previous model f_{i-1}^{CL} and the new data $\mathcal{D}_i^{train}$ and returns the updated model f_i^{CL}. Each experience k has a separate loss $\mathcal{L}^{i,k} = \mathcal{L}(f_i^{CL}, \mathcal{D}_k)$ and the objective of the algorithm is to minimize the loss over the entire stream $\mathcal{L}^S = \sum_{k=1}^{n} \mathcal{L}^{n,k}$.

In general, DCL is more constrained than other CL scenario, even rehearsal-free scenarios since the consolidated model will never see any real data. As a result, most CL strategies cannot be applied to DCL. Conversely, DCL methods can be applied to single devices by performing an adaptation followed by the consolidation for each new experience. In fact, our experiments show that the proposed method is competitive even for single devices (Section 4).

DCL is also different from federated learning (FL) [9]. In FL, a single model is trained using multiple devices. Critically, the training loop is characterized by multiple rounds of communication, while in DCL we assume a single round for each SCD. While this may seem a small difference, it is known that FL methods are unable to deal with large drifts, either caused by the data distribution, by sparse communication, or by asynchronous interactions between clients and server [3], which are fundamental properties of the scenario we propose in this paper.

3. Data-Agnostic consolidation

As discussed above, DCL requires two algorithms: knowledge adaptation and knowledge consolidation. Since knowledge adaptation aims for optimal plasticity in the SCD task, we solve it by finetuning on the local data $\mathcal{D}_i$ via stochastic gradient descent. To solve knowledge consolidation we propose *Data-Agnostic Consolidation (DAC)*, which we describe below. DAC is a method for knowledge consolidation that matches the output and feature space of the two models via a double knowledge distillation (Section 3.1). At each iteration, DAC samples from a source of data and uses heavy augmentations to increase the data diversity (Section 3.2). Finally, DAC leverages a novel approach called *Projected Latent Distillation (PLD)* to distill the latent spaces of the two models (Section 3.3). While our main objective with DAC is to solve DCL, the method can be also be used in the classic single device setting (shown in Section 4).

3.1. Knowledge consolidation objective

In knowledge consolidation, we combine the previous CL model f_{i-1}^{CL} and the current SCD model f_i^{SCD}. The resulting model f_i^{CL} consolidates the knowledge of both models. As discussed before, we do not have access to $\mathcal{D}_i$ during the consolidation. Therefore, we use an out-of-distribution dataset $\mathcal{D}^{ood}$, as discussed in the next section.

We assume that f_i^{CL}, the current CL model, is a multi-head architecture [10,11] which uses a shared feature extractor for all the tasks and a separate linear classifier $\texttt{softmax}(\mathbf{W}_i\mathbf{h} + \mathbf{b}_i)$ for each task, where $\mathbf{W}_i, \mathbf{b}_i$ are the parameters of the head for task i. This architecture allows knowledge transfer between tasks via sharing a common feature extractor and task-specific classification heads. The disadvantage is that it requires task labels to select the correct head at inference time. It is possible to remove this requirement by performing task inference, i.e. predicting the task given the current input. This approach would be completely orthogonal to our method. For example, in the CORe50 experiments we do not use task labels and we simply average the outputs of all the heads.

At step i, we want to consolidate a multi-head model f_{i-1}^{CL} with $i-1$ heads and a single-head model f_i^{SCD}. The desired result is a multi-head model f_i^{CL} with i heads such that

$$f_i^{CL}(\boldsymbol{x}, k) = f_{i-1}^{CL}(\boldsymbol{x}, k), \quad \forall \boldsymbol{x} \in \mathbb{R}^N, k \neq i \tag{1}$$

$$f_i^{CL}(\boldsymbol{x}, i) = f_i^{SCD}(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathbb{R}^N. \tag{2}$$

Notice that, since we want to match the two teachers exactly on every input, the objective does not depend on the original data. We can optimize θ_i^{CL} by stochastic gradient descent minimizing a double knowledge distillation loss

$$E_{\boldsymbol{x} \sim \mathcal{D}^{ood}}[\mathcal{L}^{DKD}(\boldsymbol{x})] = \mathcal{L}^{KD}(f_i^{CL}(\boldsymbol{x}, i), f_i^{SCD}(\boldsymbol{x})) \tag{3}$$

$$+ \sum_{k=1}^{i-1} \mathcal{L}^{KD}(f_i^{CL}(\boldsymbol{x}, k), f_{i-1}^{CL}(\boldsymbol{x}, k)) \tag{4}$$

where $\mathcal{L}^{KD}(\boldsymbol{y}_S, \boldsymbol{y}_T)$ is the KL-divergence between the student's output $\boldsymbol{y}_S$ and the teacher's output $\boldsymbol{y}_T$. $\mathcal{L}^{DKD}$ balances each task contribution by summing the head's errors.

3.2. Sampling data for consolidation

While having the original data would help knowledge consolidation, the objective defined in Eq. (2) is data-agnostic. In DCL we assume to not have access to the original data, and we will use a small set of out-of-distribution samples $\mathcal{D}_{ood}$ instead. Our experiments confirmed that DAC does not need the original data. In fact, we have found a single image to be sufficient in many experimental settings. Following [4,5], we use a large set of stochastic augmentations, such as jittering, rotation, crop, resize, flip, CutMix [12], to create a dataset of highly diverse samples from a small number of images. The resulting images will be heavily distorted but we can still use them to match the output targets of the two teachers in Eq. (2). Augmentations ensure that, even with a small number of samples, we have enough diversity in our data, which as we will show in Section 4 is critical for success. While the preprocessing pipeline can become the bottleneck of the training process with such a large number of augmentations, it is possible to trade-off memory to save computation by precomputing a large number of pre-processed images.

3.3. Projected latent distillation

The knowledge distillation loss in Eq. (4) matches the outputs for each teacher's head. However, we would like to also match the latent space of the two teachers. This is more problematic because given an input $\boldsymbol{x}$, while the outputs $\boldsymbol{y}_k$ are computed by separate heads, and therefore have no interference, the hidden activations share the same units in the consolidated model (as shown in Fig. 2). As a result, for a specific hidden layer we have two different targets $\boldsymbol{h}_{i-1}^{CL}$ and $\boldsymbol{h}_i^{SCD}$ and a single activation vector $\boldsymbol{h}_i^{CL}$ for the student. In general, since the two teachers are trained on different tasks, for most inputs we expect $\boldsymbol{h}_{i-1}^{CL} \neq \boldsymbol{h}_i^{SCD}$, which means that $\boldsymbol{h}_i^{CL}$ cannot be equal to both at the same time. To solve this issue, we propose *Projected Latent Distillation (PLD)*. The underlying intuition is that while we cannot enforce an exact match, we can match the two hidden states up to a linear transformation. During the consolidation phase, we optimize two linear transformations $\boldsymbol{W}^{SC}$ and $\boldsymbol{W}^{CL}$ that map the teachers' hidden states to the student's hidden states. The loss is then defined as

$$\mathcal{L}^{PLD}(\boldsymbol{h}_i^{CL}, \boldsymbol{h}_{i-1}^{CL}, \boldsymbol{h}_i^{SCD}) = \|\boldsymbol{W}^{SC}\boldsymbol{h}_i^{CL} - \boldsymbol{h}_i^{SCD}\|_2^2 \tag{5}$$

$$+(i-1)\|\boldsymbol{W}^{CL}\boldsymbol{h}_i^{CL} - \boldsymbol{h}_{i-1}^{CL}\|_2^2, \tag{6}$$

where $\boldsymbol{h}_i^{CL}$ is the student, $\boldsymbol{h}_{i-1}^{CL}$ the previous CL model (already trained on $i-1$ tasks) and $\boldsymbol{h}_i^{SCD}$ the SCD model's hidden states. $\boldsymbol{W}^{SC}$ and $\boldsymbol{W}^{CL}$ are initialized to the identity matrix and optimized via end-to-end backpropagation during the consolidation phase, together with the parameters of the new consolidated model. The two teachers are kept frozen during training. The loss encourages the student model to also match the teachers' hidden states. Notice that the loss for f_{i-1}^{CL} is multiplied by $i-1$ to give the same weight to each task. The same effect is present in the distillation in the output space defined by Eq. (4), since we sum all the heads (one for each task). The total loss of DAC is $\mathcal{L}(\boldsymbol{x}, \boldsymbol{h}_i^{CL}, \boldsymbol{h}_{i-1}^{CL}, \boldsymbol{h}_i^{SCD}) = \mathcal{L}^{DKD}(\boldsymbol{x}) + \lambda\mathcal{L}^{PLD}(\boldsymbol{h}_i^{CL}, \boldsymbol{h}_{i-1}^{CL}, \boldsymbol{h}_i^{SCD})$, where λ controls the ratio between the output and latent distillation losses.

A schematic view of DAC is shown in Fig. 2.

4. Experiments

We show experimental results in the sequential and independent setting introduced in Section 2. We use CIFAR100 [13], Tiny ImageNet [14] and CORe50 [15] to create our benchmarks. The source code is implemented in Avalanche [16] and publicly available.[2] More details about the experiments can be found in the Appendix.

We use SplitCIFAR100 in the task-incremental setting using 10 and 20 tasks, where the stream is divided into experiences of 10 and 5 classes, respectively. We use a slimmed ResNet18 as defined in [17]. We also use Split Tiny ImageNet (10 Task) in the task-incremental setting with a WideVGG architecture following [11]. Finally, we use CORe50 [15], a task agnostic-benchmark, in the joint and domain-incremental (NI) settings. CORe50 images are scaled to 128×128. We use a MobileNetv2 [18] pretrained on ImageNet, as [19]. We use these streams in a DCL scenario by learning each task in a different SCD. SplitCIFAR100 and Split Tiny ImageNet are sequential DCL scenario, while CoRE50 is an independent DCL scenario. The default data source for DAC is a single image ("animals", shown in the Appendix).

Distributed continual learning evaluation

We point out that many methods that we compare against do not satisfy the DCL constraints. This is due to the fact that most CL methods are unable to learn only from the stream of pretrained models. In the following results, we denote in the DCL columns of Tables 1a, 1b and 2 whether a method satisfies DCL constraints (✓), rehearsal-free constraints (RF), or it is simply a baselines that uses rehearsal data or keeps all the teachers in memory (B). The comparison against rehearsal-free baselines is used to show that DAC is a state-of-the-art method even in the single device setting. All the results are the average of 5 runs on different seeds.

The consolidated CL model is evaluated with the average accuracy over the entire test stream. Given $A_{t,i}$ as accuracy for task i of the model obtained after training on task t, the average accuracy is defined as $A_t = \sum_{i=1}^{t} A_{t,i}$. This choice also allows us to compare DAC in a more classic single device scenario and to evaluate the quality of the consolidated model in the distributed setting. We also evaluate the SCD models by measuring their local task accuracy (Section 4.2) and linear probing of the last layer to evaluate the finetuning performance on all the tasks. This allows to evaluate whether the SCDs benefit from the usage of the consolidated model, which is our main objective.

4.1. Single device and distributed continual learning

In this section we show how DAC performs in single device and distributed settings. We also show the performance of state-of-the-art rehearsal-free baselines. Although these are not applicable to DCL, the results suggest that DAC is a good choice even in a single device setting. Results for SplitCIFAR100 in the task-aware sequential setting are shown in Table 1a. We use the results in [8] as baselines. DAC shows state of the art performance on the 10 task setting. Despite the limited data source, a single out-of-domain image ("animals"), DAC outperforms LwF, which uses the real data $\mathcal{D}_i$ for distillation. We argue that there are two properties of DAC which justify this improvement. First, the use of heavy augmentations improves distillation even with limited data, which confirms previous observations in the offline setting [4,5]. Furthermore, during the consolidation DAC weighs the current task and all the previous ones in the same way by using the same loss for all tasks and summing them (Eq. (4), (6)). Instead, LwF uses the cross-entropy for the current task and the KL divergence for the previous ones, which makes it more difficult to balance stability and plasticity (the well known stability-plasticity dilemma [6]). We evaluated DAC on the 20 task setting (5 classes per task), which results in the average test accuracy of $86.2_{\pm 1.0}$, even higher than the 10 task setting, suggesting that DAC can scale well to longer streams. Results on Tiny ImageNet in the sequential setting (Table 1b) are consistent with Split CIFAR100, with DAC as the best performing method with a lower performance when PLD is not used ($\lambda = 0$).

Experiments on CORe50 in the independent setting are shown in Table 2. Notice that while our method uses a multi-head, CORe50 is a task-agnostic benchmark. Therefore, we convert the final multi-head model into a task-agnostic model by averaging the output of all the heads. CORe50 is also a more challenging benchmark due to the higher image resolution (128×128). We compare against the methods in [21], which perform knowledge distillation using synthetic data or the entire ImageNet dataset. Overall, DAC obtains the best performance, even outperforming Aux. Data ED, which uses ImageNet (with more than 1 million images) for distillation.

Fig. 3 shows the learning curves for SplitCIFAR100 (10 Tasks) and CORe50-NI. We notice that on Split CIFAR100 the forgetting, i.e. the difference between the accuracy after training on task i and the task at the end of training on the entire stream, is very low. This means that the consolidation process works with minimal forgetting. On CORe50, we see a bigger drop over time, but the old and new tasks have a similar performance. This suggests that the drop may be the result of the problem becoming more difficult with the introduction of new instances while keeping a fixed capacity.

4.2. PLD improves forward transfer

In this section, we study how the sequential initialization of the SCD models in DAC and PLD affects the SCD model's accuracy with respect

[2] Available as supplementary material. The source code will be publicly released after acceptance.

Table 1
Average test set accuracy A_t. The column DCL shows whether a strategy satisfies the DCL constraints (✓), it is a rehearsal-free strategy (RF), or it is a baseline which uses replay data or keeps multiple models in memory (B).

Task-incremental SplitCIFAR100 after task 5 and 10. Baselines denoted by † are taken from [8], while results denoted with ‡ are taken from [20]. Best in bold, send best underlined.

	SplitCIFAR100		
	DCL	5 Tasks	10 Tasks
Naive†	RF	49.8	38.3
EWC†	RF	60.2	56.7
PathInt†	RF	57.3	53.1
MAS†	RF	61.8	58.6
RWalk†	RF	56.3	49.3
LwM†	RF	76.2	70.4
LwF†	RF	76.7	76.6
EWC-GI‡	RF	–	63.3
WSN‡	RF	–	69.3
SPG‡	RF	–	67.7
DMC†	✓	72.3	66.7
DAC($\lambda = 0$)	✓	$\underline{77.6_{\pm 1.7}}$	$\underline{77.5_{\pm 0.6}}$
DAC	✓	**81.4**$_{\pm 1.6}$	**80.5**$_{\pm 0.8}$

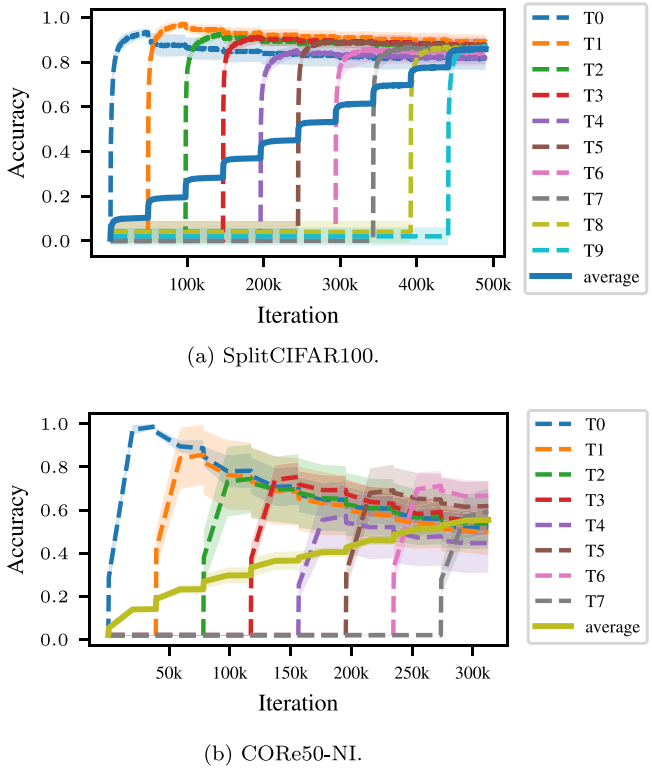
Task-incremental Split Tiny ImageNet. Baselines denoted by † are taken from [11], while results denoted with ‡ are taken from [20]. Best in bold, send best underlined.

	Split Tiny ImageNet	
	DCL	ACC
Joint†	B	57.29
Naive†	RF	25.28
PackNet†	RF	47.64
HAT†	RF	43.78
SI†	RF	33.86
EWC†	RF	31.10
MAS†	RF	45.08
LwF†	RF	46.79
EBLL†	RF	46.25
mode-IMM†	RF	36.42
EWC-GI‡	RF	$\underline{48.3}$
WSN‡	RF	47.8
SPG‡	RF	**48.4**
DAC($\lambda = 0$)	✓	40.46$_{\pm 9.2}$
DAC	✓	$\underline{48.3_{\pm 0.5}}$

Table 2
Average test set accuracy A_t for CORe50 in a task-agnostic independent DCL scenario. The column DCL shows whether a strategy satisfies the DCL constraints (✓), it is a rehearsal-free strategy (RF), or it is a baseline which uses replay data or keeps multiple models in memory (B).

	DCL	CORe50	
		Joint	NI
Oracle[a]	B	85.7$_{\pm 0.2}$	–
Min. Entropy[a]	B	–	61.3$_{\pm 1.8}$
Output Avg.[a]	B	–	69.9$_{\pm 0.7}$
Replay ED[a]	B	87.4$_{\pm 0.2}$	83.7$_{\pm 0.5}$
Parameter Avg.[a]	✓	–	2.0$_{\pm 0.0}$
MI-ED[a]	✓	50.0$_{\pm 2.7}$	44.3$_{\pm 4.9}$
DI-ED[a]	✓	52.9$_{\pm 2.0}$	43.2$_{\pm 2.3}$
Aux. Data ED[a]	✓	81.8$_{\pm 0.2}$	44.5$_{\pm 2.9}$
DAC($\lambda = 0$)	✓	82.2$_{\pm 0.2}$	41.5$_{\pm 9.07}$
DAC	✓	**84.9**$_{\pm 0.2}$	**48.9**$_{\pm 1.9}$

[a] Baselines are taken from [21].



(a) SplitCIFAR100.

(b) CORe50-NI.

Fig. 3. Task-specific and stream average accuracy on the training set during training.

to 3 dimensions: forward transfer, the generalization of the hidden representations to other tasks, and the representation similarity between different SCD models. We use SCD models trained on SplitCIFAR100 (10 Tasks). We use a common random initialization as a baseline, which is the configuration in which there is zero forward transfer.

The sequential initialization should encourage forward transfer between the CL model and the SCD model, i.e. it should improve the SCD model's performance compared to the random independent initialization. Fig. 5 shows the accuracy of the 10 SCD models on the task they have seen during training (left) and the average accuracy over all tasks for a task-agnostic linear probe which uses the final layer's representation, finetuned on all the tasks (right). Surprisingly, while DAC obtains a higher accuracy than the random initialization on the linear probing, the average local task accuracy is actually lower. Therefore, it seems that without latent distillation there is negative forward transfer. Instead, *PLD has positive forward transfer, improving both the local task and the linear probing on unseen tasks* (Fig. 5(b)).

In Fig. 4, we measure representation similarity with the CKA [22]. The figure shows the CKA between the first (Task 0) and last (Task 9) SCD models.[3] In general, the CKA similarity is relatively high for the different configurations since they all share the same architecture and are trained on similar data. Somewhat surprisingly, the similarity between PLD models is lower than the similarity of DAC SCD models. In principle, we expected to find a positive correlation between the representation similarity and the forward transfer. This appears not to be the case since PLD has better forward transfer but DAC shows closer similarity. We hypothesize that the PLD loss encourages the consolidated model to learn more diverse representations, decreasing the representation similarity over time but increasing the forward transfer thanks to the richer representation.

[3] The CKA for the entire stream is shown in the Supplementary Material.

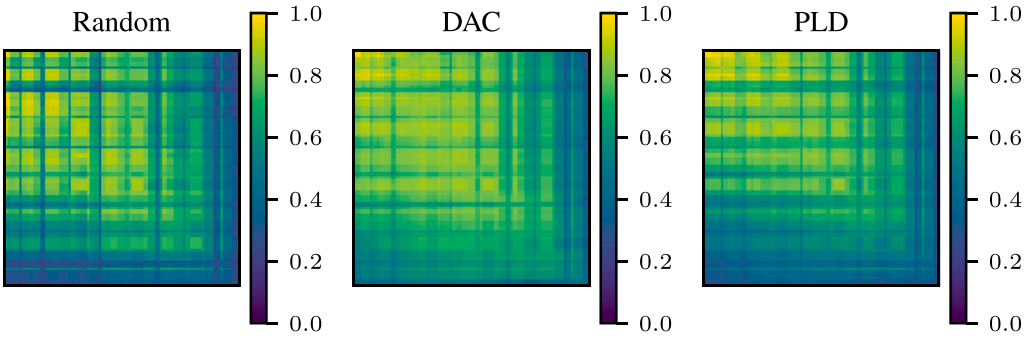


Fig. 4. CKA between the first and last SCD model of SplitCIFAR100 (10 Tasks) computed using the last task.

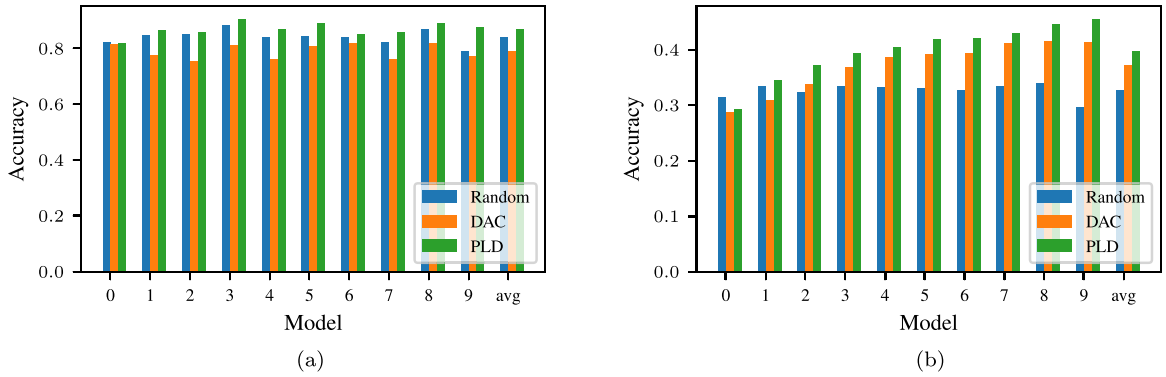


Fig. 5. Accuracy of each SCD model on its local task (left) and task-agnostic linear probing on all tasks (right).

4.3. Comparison between different data sources

We have already shown that DAC with a single out-of-distribution image is competitive with state of the art methods. In this section, we compare different data sources to evaluate the impact of different properties of the data on the consolidation performance (Table 3). We use data taken from the original tasks, single images vs full dataset, and natural images against other domains, synthetic images and static noise:

- city: high-resolution (around 2560 × 1920, 1.85 MB) image of a japanese market;
- animals: medium-resolution (600 × 225, 338 kB) poster with several animals;
- hubble: high-resolution image (2300 × 2100, 6.90 MB) from the Hubble telescope;
- bridge: Image of the San Francisco Golden Gate Bridge (1165 × 585, 1.17 MB);
- ImageNet: samples from ImageNet (using only CutMix);
- DeepInversion: synthetic images generated via DeepInversion [23];
- noise: static noise.

Images and samples are shown in Appendix C. In general, we notice that using a single image is sufficient to reach a very high performance. However, there is still a large gap between the use of real data (D_i) and a single out-of-distribution image. It is important to notice that DAC helps even when using the real data since we obtain an accuracy of 84.2 against the 65.8 of LwF. Moreover, even very different domains ("hubble") still work better than completely random data ("noise"). Finally, diversity, given either by a large $\mathcal{D}^{ood}$ or by heavy augmentations seems to be the most important factor since using ImageNet (more than 1M images) is slightly better than using data from the current task (5000 images).

Table 3
Test accuracy of DAC with different data sources on SplitCIFAR100 (10 Tasks).

	Original domain	Single image	Natural	ACC
Current data (D_i)	✓		✓	$84.2_{\pm 0.5}$
ImageNet			✓	$84.8_{\pm 0.6}$
animals		✓	✓	$82.1_{\pm 0.5}$
city		✓	✓	$80.5_{\pm 0.8}$
bridge		✓	✓	$79.0_{\pm 0.6}$
hubble		✓		$65.1_{\pm 0.3}$
DeepInversion				$64.9_{\pm 3.3}$
noise		✓		$10.7_{\pm 0.5}$

4.4. Analysis

We can summarize the main findings as follow:

DAC provides a good stability-plasticity tradeoff. The separate consolidation step provides a better stability-plasticity tradeoff, higher accuracy, and better scaling w.r.t. the number of tasks, even in rehearsal-free scenarios.

PLD improves forward transfer between models. In the sequential setting, DAC and the PLD loss shows positive forward transfer. This result suggests that latent distillation helps to combine models with different representations and to learn richer latent representations.

Small data sources are sufficient. Limited data sources, such as a single out-of-distribution image, show a good performance due to the heavy augmentations. This opens up to the opportunity of offloading the consolidation to a server without sharing the data.

Limitations. The main limitation of DAC is the number of epochs required for the consolidation. While the computational cost of a single iteration is comparable to a classic knowledge distillation step, the use of heavy augmentations and out-of-distribution images slow down convergence. This concern may be mitigated by collecting a small

amount of data, if possible, or trading off accuracy for computational benefits by reducing the number of epochs. Regarding the experiments, we had to limit the scope of our experimental analysis to two scenarios (the sequential and independent) even though most realistic scenarios will be a combination between the two. Furthermore, most CL methods are not applicable to DCL, which limits the number of baselines that we can fairly compare against. We compared against rehearsal-free baselines to mitigate this limitation.

5. Related works

Recent work on knowledge distillation (KD) [24] explores the possibility of distillation with limited data. Data-free KD is applied in the offline training setting by creating synthetic images with generators in [25]. More relevant to our work, [26] propose handcrafted noise and simple procedurally generated images, while [27] create image by combining together slices of several images. [5] show the beneficial effect of heavy augmentations and [4] additionally show the benefit of long training schedules. More recently, [28] proposes multiple distillation stages via channel, response, and cross-stage review distillation, while [29] introduces multilevel attention maps to construct sample correlations for knowledge distillation (MASCKD).

In continual learning, many popular methods are based on knowledge distillation [10,30]. Progress and Compress [31] uses an adaptation and compression step reminescent of our consolidation, but it still needs the original data. [32] applies projected feature distillation in a continual self-supervised setting. Similar to us, they use a projection network to distill the features, however their method is purely unsupervised and neither can it handle distillation from multiple networks as we consider in this paper. In continual learning, exemplar-free scenarios are very popular, especially in the class-incremental setting (DFCIL). This scenario is different from our setting since the model still has access to $\mathcal{D}_i$ (see also Appendix B). Many strategies address DFCIL by using alternative sources of data. [21] uses synthetic data generated via model inversion, [33] uses external data, and [7] uses a data-free training process similar to GANs. [34] can optionally use external data for model calibration. [35,36] uses external data for semantic segmentation and object detection, respectively, by exploiting the notion of background on these tasks, which provide a neutral label, which makes them inapplicable for classification tasks. Among all these strategies, only [21] (ED, Table 2) and [33] (DMC, Table 1a) are fully applicable to DCL.

6. Conclusion

In this paper, we studied the problem of distributed continual learning (DCL), a scenario which provides two challenges: zero-cost transfer learning between SCDs and incremental learning from a stream of models. We proposed DAC as a general DCL method that satisfy the privacy and communication constraints. We showed that DAC reaches state-of-the-art performance using a single out-of-distribution image in single device and distributed CL. Additionally, we showed that DAC enables positive transfer between SCDs.

DCL is a very general setting that opens up many research avenues. In this paper, we focused on the computational problem of consolidation, but many other aspects are worth of study, such as studying different communication protocols or sending other forms of encoded knowledge instead of the parameters. Knowledge consolidation also opens up the possibility of a continuous interaction between SCDs and large continually pretrained models.

To conclude, we would like to point out that improvements in DCL can easily transfer to the single-model/single-device scenario, as it happened for DAC, which improved the state-of-the-art performance on the task-incremental Split CIFAR100 and Split TinyImageNet. We hope that the research in distributed continual learning scenarios will provide insights about questions such as the relationship between adaptation and consolidation explored in this paper.

CRediT authorship contribution statement

Antonio Carta: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing, Validation. **Andrea Cossu:** Conceptualization, Writing – review & editing, Methodology. **Vincenzo Lomonaco:** Conceptualization, Writing – review & editing, Methodology. **Davide Bacciu:** Conceptualization, Writing – review & editing, Methodology. **Joost van de Weijer:** Conceptualization, Methodology, Writing – review & editing.

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

Data will be made available on request.

Acknowledgments

This project was partially supported by the Ministry of University and Research (MUR), Italy as part of the FSE REACT-EU - PON 2014–2020 “Research and Innovation” resources – Innovation Action - DM MUR 1062/2021.

Appendix A. Reproducibility

We release our source code (anonymized in the supplementary material for the review, public on github afterwards). All of our experiments use Avalanche [16], a continual learning library based on PyTorch [37]. We release the experiments’ configurations using Hydra [38](hierarchical yaml configuration files), which means that each experiment in the paper can be reproduced by running a main python script with the desired configuration, as detailed in the README of the source code. Experimental details relevant for independent implementations are available in the following sections.

Appendix B. Comparison between related learning scenarios

The distributed continual learning scenario present some similarities with other scenarios in the literature. We provide a more detailed discussion of their differences here hoping to highlight their difference:

continual learning : a single model learning from a nonstationary stream of data. Knowledge sharing between models is not possible and current data is always available. Plasticity is suboptimal due to the stability/plasticity tradeoff.

rehearsal-free CL : includes scenarios such as data-free class-incremental learning (DFCIL), where a single model learns from a nonstationary stream of data. Data from previous experiences is unavailable due to privacy constraints or severe storage limitations. Knowledge sharing is not possible and current data is always available. Plasticity is suboptimal due to the stability/plasticity tradeoff.

federated : client–server organization with a single centralized controller. All the clients are learning the same task, possibly on different data. The server has full control over the training process and the client synchronize every few training iterations, resulting in multiple communication rounds. SCD data is private but plasticity is suboptimal because SCD are optimized on the global instead of the local task.

Table B.4
Summary of the main properties of different scenarios related to distributed continual learning.

	Past data privacy	Current data privacy	Optimal plasticity	Multiple devices	Single round of communication
Continual learning					
Rehearsal-free continual learning	✓				
Federated learning	✓			✓	
Ours	✓	✓	✓	✓	✓

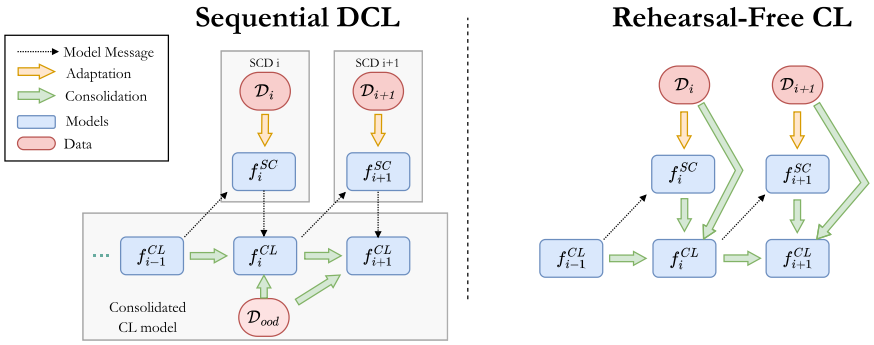


Fig. B.6. Schematic comparison of Sequential DCL vs Rehearsal-free CL, showing a possible implementation of DAC in rehearsal-free. Notice that in rehearsal-free CL we have the possibility to use the original data during consolidation and all the learning steps can happen on the same device.

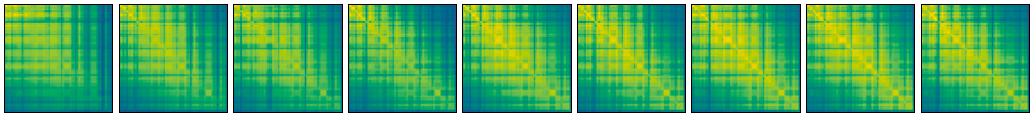


Fig. E.7. CKA for DAC($\lambda = 0$).

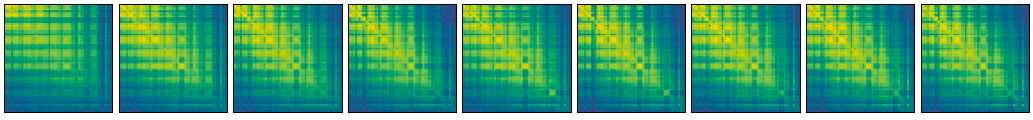


Fig. E.8. CKA for DAC.

DCL : Each SCD learn its own task, keeping all the data private. Plasticity is optimal because SCD are optimizing their own task. Communication is minimized by using a single round of communication.

Fig. B.6 shows the training process of the four different scenarios, mapping the adaptation and consolidation phases defined in Section 2 to the other scenarios. Table B.4 summarizes the properties of the different scenario.

Appendix C. Data sources

In this section, we show samples from the images used for knowledge distillation. *city*: high-resolution (around 2560 × 1920, 1.85 MB) image of a japanese market;

Appendix D. Hyperparameters

Splitcifar100:. We use a slimmed ResNet18 as a backbone for both the teacher and consolidated model. During the consolidation, we use Adam with learning rate set to 0.0001, with a batch size of 512 and 500′000 iterations. We use a temperature of 0.5 for distillation. For the PLD loss, we set $\lambda = 0.01$ and apply the loss at `layer4.0` and `linear` (logits).

CORe50:. We use a MobileNet v2 pretrained on ImageNet as a backbone for both the teacher and consolidated model. During the consolidation, we use Adam with learning rate set to 0.0001, with a batch size of 128 and 100′000 iterations. We use a temperature of 1.0 for distillation. For the PLD loss, we set $\lambda = 100.0$ and apply the loss at `classifier` (logits).

Appendix E. CKA

In this section, we show the CKA, as described in Section 4.2, for the entire stream. We compute the CKA between the first expert and the expert after experience i (see Figs. E.7 and E.8).

References

[1] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, N. Díaz-Rodríguez, Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges, Inf. Fusion 58 (2020) 52–68, http://dx.doi.org/10.1016/j.inffus.2019.12.004, arXiv:1907.00182.

[2] T. Li, A.K. Sahu, A. Talwalkar, V. Smith, Federated learning: Challenges, methods, and future directions, IEEE Signal Process. Mag. 37 (3) (2020) 50–60, http://dx.doi.org/10.1109/MSP.2020.2975749.

[3] G. Legate, L. Caccia, E. Belilovsky, Re-weighted softmax cross-entropy to control forgetting in federated learning, 2023, http://dx.doi.org/10.48550/arXiv.2304.05260, arXiv:2304.05260.

[4] L. Beyer, X. Zhai, A. Royer, L. Markeeva, R. Anil, A. Kolesnikov, Knowledge distillation: A good teacher is patient and consistent, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10925–10934, arXiv:2106.05237.

[5] Y.M. Asano, A. Saeed, Extrapolating from a single image to a thousand classes using distillation, 2022, arXiv:2112.00725 [cs]. arXiv:2112.00725.

[6] R.M. French, Catastrophic forgetting in connectionist networks, Trends in Cognitive Sciences 3 (4) (1999) 128–135, http://dx.doi.org/10.1016/S1364-6613(99)01294-2.

[7] J. Smith, Y.-C. Hsu, J. Balloch, Y. Shen, H. Jin, Z. Kira, Always be dreaming: A new approach for data-free class-incremental learning, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 9374–9384, arXiv:2106.09701.

[8] M. Masana, X. Liu, B. Twardowski, M. Menta, A.D. Bagdanov, J. van de Weijer, Class-incremental learning: Survey and performance evaluation on image classification, IEEE Trans. Pattern Anal. Mach. Intell. (2022) 1–20, http://dx.doi.org/10.1109/TPAMI.2022.3213473, arXiv:2010.15277.

[9] H.B. McMahan, E. Moore, D. Ramage, S. Hampson, B.A. y Arcas, Communication-efficient learning of deep networks from decentralized data, 2017, arXiv:1602.05629 [cs]. arXiv:1602.05629.

[10] Z. Li, D. Hoiem, Learning without forgetting, IEEE Trans. Pattern Anal. Mach. Intell. 40 (12) (2018) 2935–2947, http://dx.doi.org/10.1109/TPAMI.2017.2773081, arXiv:1606.09282.

[11] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, T. Tuytelaars, A continual learning survey: Defying forgetting in classification tasks, IEEE Trans. Pattern Anal. Mach. Intell. 44 (7) (2022) 3366–3385, http://dx.doi.org/10.1109/TPAMI.2021.3057446.

[12] S. Yun, D. Han, S.J. Oh, S. Chun, J. Choe, Y. Yoo, CutMix: Regularization strategy to train strong classifiers with localizable features, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6023–6032, arXiv:1905.04899.

[13] A. Krizhevsky, Learning Multiple Layers of Features from Tiny Images, Tech. rep., University of Toronto, 2009, p. 60.

[14] P. Chrabaszcz, I. Loshchilov, F. Hutter, A downsampled variant of ImageNet as an alternative to the CIFAR datasets, 2017, http://dx.doi.org/10.48550/arXiv.1707.08819, arXiv:1707.08819.

[15] V. Lomonaco, D. Maltoni, CORe50: A new dataset and benchmark for continuous object recognition, in: S. Levine, V. Vanhoucke, K. Goldberg (Eds.), Proceedings of the 1st Annual Conference on Robot Learning, in: Proceedings of Machine Learning Research, vol. 78, PMLR, 2017, pp. 17–26.

[16] V. Lomonaco, L. Pellegrini, A. Cossu, A. Carta, G. Graffieti, T.L. Hayes, M. De Lange, M. Masana, J. Pomponi, G.M. van de Ven, M. Mundt, Q. She, K. Cooper, J. Forest, E. Belouadah, S. Calderara, G.I. Parisi, F. Cuzzolin, A.S. Tolias, S. Scardapane, L. Antiga, S. Ahmad, A. Popescu, C. Kanan, J. van de Weijer, T. Tuytelaars, D. Bacciu, D. Maltoni, Avalanche: An end-to-end library for continual learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 3600–3610.

[17] D. Lopez-Paz, M. Ranzato, Gradient episodic memory for continual learning, Adv. Neural Inf. Process. Syst. 2017-Decem (Nips) (2017) 6468–6477.

[18] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017, arXiv:1704.04861 [cs]. arXiv:1704.04861.

[19] D. Maltoni, V. Lomonaco, Continuous learning in single-incremental-task scenarios, Neural Netw. 116 (2019) 56–73, http://dx.doi.org/10.1016/j.neunet.2019.03.010.

[20] T. Konishi, M. Kurokawa, C. Ono, Z. Ke, G. Kim, B. Liu, Parameter-level soft-masking for continual learning, in: Proceedings of the 40th International Conference on Machine Learning, PMLR, 2023, pp. 17492–17505.

[21] A. Carta, A. Cossu, V. Lomonaco, D. Bacciu, Ex-model: Continual learning from a stream of trained models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 3790–3799.

[22] T. Nguyen, M. Raghu, S. Kornblith, Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth, in: International Conference on Learning Representations, 2022.

[23] H. Yin, P. Molchanov, J.M. Alvarez, Z. Li, A. Mallya, D. Hoiem, N.K. Jha, J. Kautz, Dreaming to distill: Data-free knowledge transfer via DeepInversion, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 8715–8724, arXiv:1912.08795.

[24] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, 2015, http://dx.doi.org/10.48550/arXiv.1503.02531, Arxiv preprint. arXiv:1503.02531.

[25] Y. Liu, W. Zhang, J. Wang, J. Wang, Data-free knowledge transfer: A survey, 2021, arXiv:2112.15278 [cs]. arXiv:2112.15278.

[26] M. Baradad Jurjo, J. Wulff, T. Wang, P. Isola, A. Torralba, Learning to see by looking at noise, in: Advances in Neural Information Processing Systems, Vol. 34, Curran Associates, Inc., 2021, pp. 2556–2569, arXiv:2106.05963.

[27] G. Fang, Y. Bao, J. Song, X. Wang, D. Xie, C. Shen, M. Song, Mosaicking to distill: Knowledge distillation from out-of-domain data, in: Thirty-Fifth Conference on Neural Information Processing Systems, 2021.

[28] J. Gou, X. Xiong, B. Yu, L. Du, Y. Zhan, D. Tao, Multi-target knowledge distillation via student self-reflection, Int. J. Comput. Vis. 131 (7) (2023) 1857–1874, http://dx.doi.org/10.1007/s11263-023-01792-z.

[29] J. Gou, L. Sun, B. Yu, S. Wan, W. Ou, Z. Yi, Multilevel attention-based sample correlations for knowledge distillation, IEEE Trans. Ind. Inform. 19 (5) (2023) 7099–7109, http://dx.doi.org/10.1109/TII.2022.3209672.

[30] P. Buzzega, M. Boschini, A. Porrello, D. Abati, S. Calderara, Dark experience for general continual learning: A strong, simple baseline, in: Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 15920–15930, arXiv:2004.07211.

[31] J. Schwarz, J. Luketina, W.M. Czarnecki, A. Grabska-Barwinska, Y.W. Teh, R. Pascanu, R. Hadsell, Progress & compress: A scalable framework for continual learning, in: 35th International Conference on Machine Learning, ICML 2018, Vol. 10, 2018, pp. 7199–7208.

[32] A. Gomez-Villa, B. Twardowski, L. Yu, A.D. Bagdanov, J. van de Weijer, Continually learning self-supervised representations with projected functional regularization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 3867–3877, arXiv:2112.15022.

[33] J. Zhang, J. Zhang, S. Ghosh, D. Li, S. Tasci, L. Heck, H. Zhang, C.-C.J. Kuo, Class-incremental learning via deep model consolidation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2020, pp. 1131–1140, arXiv:1903.07864.

[34] K. Lee, K. Lee, J. Shin, H. Lee, Overcoming catastrophic forgetting with unlabeled data in the wild, 2019, pp. 312–321, http://dx.doi.org/10.48550/arXiv.1903.12648, arXiv:1903.12648.

[35] L. Yu, X. Liu, J. van de Weijer, Self-training for class-incremental semantic segmentation, IEEE Trans. Neural Netw. Learn. Syst. (2022) 1–12, http://dx.doi.org/10.1109/TNNLS.2022.3155746, arXiv:2012.03362.

[36] N. Dong, Y. Zhang, M. Ding, G.H. Lee, Bridging non co-occurrence with unlabeled in-the-wild data for incremental object detection, in: Advances in Neural Information Processing Systems, Vol. 34, Curran Associates, Inc., 2021, pp. 30492–30503, arXiv:2110.15017.

[37] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, PyTorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.

[38] O. Yadan, Hydra - A Framework for Elegantly Configuring Complex Applications, Github, 2019.