Transfer without Forgetting

Matteo Boschini[1], Lorenzo Bonicelli[1], Angelo Porrello[1],
Giovanni Bellitto[2], Matteo Pennisi[2], Simone Palazzo[2],
Concetto Spampinato[2], and Simone Calderara[1]

[1] AImageLab, University of Modena and Reggio Emilia, Italy
`firstname.lastname@unimore.it`
[2] PeRCeiVe Lab, University of Catania, Italy
`firstname.lastname@unict.it`

Abstract. This work investigates the entanglement between Continual Learning (CL) and Transfer Learning (TL). In particular, we shed light on the widespread application of network pretraining, highlighting that it is itself subject to catastrophic forgetting. Unfortunately, this issue leads to the under-exploitation of knowledge transfer during later tasks. On this ground, we propose Transfer without Forgetting (TwF), a hybrid approach building upon a fixed pretrained sibling network, which continuously propagates the knowledge inherent in the source domain through a layer-wise loss term. Our experiments indicate that TwF steadily outperforms other CL methods across a variety of settings, averaging a 4.81% gain in Class-Incremental accuracy over a variety of datasets and different buffer sizes. Our code is available at `https://github.com/mbosc/twf`.

Keywords: Continual Learning, Lifelong Learning, Experience Replay, Transfer Learning, Pretraining, Attention

1 Introduction

Thanks to the enthusiastic development carried out by the scientific community, there exist myriad widely available deep learning models that can be either readily deployed or easily adapted to perform complex tasks [24,60,64,48,4]. However, the desiderata of practical applications [59] often overstep the boundaries of the typical *i.i.d.* paradigm, fostering the study of different learning approaches.

In contrast with the natural tendency of biological intelligence to seamlessly acquire new skills and notions, deep models are prone to an issue known as *catastrophic forgetting* [39], *i.e.*, they fit the current input data distribution to the detriment of previously acquired knowledge. In light of this limitation, the sub-field of Continual Learning (CL) [16,46,63] aspires to train models capable of adaptation and lifelong learning when facing a sequence of changing tasks, either through appositely designed architectures [56,57,38], targeted regularization [34,31,70] or by storing and replaying previous data points [50,52,8,14].

On a similar note, human intelligence is especially versatile in that it excels in contrasting and incorporating knowledge coming from multiple domains. Instead, the application of deep supervised learning algorithms typically demands

large annotated datasets, whose collection has significant costs and may be impractical. To address this issue, Transfer Learning (TL) techniques are typically applied with the purpose of transferring and re-using knowledge across different data domains. In this setting, the simplest technique is to pretrain the model on a huge labeled dataset (*i.e.* the source) and then finetune it on the *target* task [51,23,19]. Such a simple schema has been recently overcome by more sophisticated domain adaptation algorithms [15,36,35] mainly based on the concept of *feature alignment*: here, the goal is to reduce the shift between the feature distributions of target and source domains. Unfortunately, these approaches often require the availability of the source dataset during training, which clashes with the usual constraints imposed in the CL scenarios.

In this work, we explore the interactions between pretraining and CL and highlight a blind spot of continual learners. Previous work underlined that naive pretraining is beneficial as it leads the learner to reduced forgetting [40]. However, we detect that the pretraining task itself is swiftly and catastrophically forgotten as the model veers towards the newly introduced stream of data. This matter is not really detrimental if all target classes are available at once (*i.e.*, joint training): as their exemplars can be accessed simultaneously, the learner can discover a joint feature alignment that works well for all of them while leaving its pretraining initialization. However, if classes are shown in a sequential manner, we argue that transfer mostly concerns the early encountered tasks: as a consequence, pretraining ends up being fully beneficial only for the former classes. For the later ones, since pretraining features are swiftly overwritten, the benefit of pretraining is instead lowered, thus undermining the advantages of the source knowledge. In support of this argument, this work reports several experimental analyses (Sec. 3.1) revealing that state-of-the-art CL methods do not take full advantage of pretraining knowledge.

To account for such a disparity and let all tasks profit equally from pretraining, this work sets up a framework based on Transfer Learning techniques. We show that the Continual Learning setting requires specific and *ad-hoc* strategies to fully exploit the source knowledge without incurring its forgetting. Consequently, we propose an approach termed **Transfer without Forgetting (TwF)** that equips the base model with a pretrained and fixed sibling network, which continuously propagates its internal representations to the former network through a per-layer strategy based on *knowledge distillation* [27]. We show that our proposal is more effective than alternative approaches (*i.e.*, extending anti-forgetting regularization to the pretraining initialization) and beneficial even if the data used for pretraining is strongly dissimilar w.r.t. to the target task.

2   Related Work

Continual Learning (CL) [16,46] is an increasingly popular field of machine learning that deals with the mitigation of catastrophic forgetting [39]. CL methods are usually grouped as follows, according to the approach they take.

Regularization-based methods [31,37,13,12] typically identify subsets of weights that are highly functional for the representations of previous tasks, with the purpose to prevent their drastic modification through apposite optimization constraints. Alternatively, they consolidate the previous knowledge by using past models as soft teachers while learning the current task [34].

Architectural approaches dedicate distinct sets of parameters to each task, often resorting to network expansion as new tasks arrive [56,38,58]. While capable of high performance, they are mostly limited to the Task-IL scenario (described in Sec. 4.1) as they require task-identifiers at inference time.

Rehearsal-based methods employ a fixed-size buffer to store a fraction of the old data. ER [49,53] interleaves training samples from the current task with previous samples: notably, several works [20,8] point out that such a simple strategy can effectively mitigate forgetting and achieve superior performance. This method has hence inspired several works: DER [8] and its extension X-DER [6] also store past model responses and pin them as an additional teaching signal. MER [52] combines replay and meta-learning [21,44] to maximize transfer from the past while minimizing interference. Other works [3,9] propose different sample-selection strategies to include in the buffer, while GEM [37] and its relaxation A-GEM [13] employ old training data to minimize interference. On a final note, recent works [7,61] exploit the memory buffer to address semi-supervised settings where examples can be either labeled or not.

Transfer Learning (TL) [45] is a machine learning methodology aiming at using the knowledge acquired on a prior task to solve a distinct target task. In its classical formulation [68], a model is trained on the source dataset and then finetuned on the (possibly much smaller) target dataset to adapt the previously learned features. Alternatively, transfer can be induced via multi-level Knowledge Distillation, guided by meta-learning [30], attention [66] or higher-level descriptions of the flow of information within the model [67].

3   Method

Setting. In CL, a classification model $f_{(\theta,\phi)}$ (composed of a multilayered feature extractor $h_\theta = h_{\theta_l}^{(l)} \circ h_{\theta_{l-1}}^{(l-1)} \circ \cdots \circ h_{\theta_1}^{(1)}$ and a classifier g_ϕ, $f_{(\theta,\phi)} = g_\phi \circ h_\theta$) is trained on a sequence of N tasks $\mathcal{T}_i = \{(x_j^i, y_j^i)\}_{j=1}^{|\mathcal{T}_i|}$. The objective of $f_{(\theta,\phi)}$ is minimizing the classification error across all seen tasks:

$$\min_{\theta,\phi} \mathcal{L} = \mathbb{E}_i \left[\mathbb{E}_{(x,y)\sim\mathcal{T}_i} \left[\ell(y, f_{(\theta,\phi)}(x)) \right] \right], \tag{1}$$

where ℓ is a suitable loss function. Unfortunately, the problem framed by Eq. 1 cannot be directly optimized due to the following key assumptions: *i)* while learning the current task $\mathcal{T}_c$, examples and labels of previous tasks are inaccessible; *ii)* the label space of distinct tasks is disjoint ($y_m^i \neq y_n^j \ \forall i \neq j$) *i.e.,* classes learned previously cannot recur in later phases. Therefore, Eq. 1 can only be approximated, seeking adequate performance on previously seen tasks (*stability*), while remaining flexible enough to adapt to upcoming data (*plasticity*).
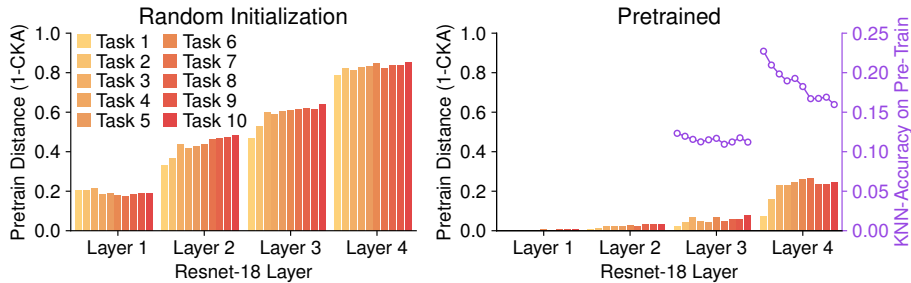
Fig. 1. Forgetting of the initialization, measured as the distance from the pretrain $(1-\mathrm{CKA}$ [32]) (lower is better) and kNN accuracy (higher is better). Features extracted by a pretrained model remain closer to the initialization w.r.t. a randomly initialized model. Furthermore, the steady decrease in kNN accuracy as training progresses reveals that features become less specific for past tasks.

3.1   Pretraining incurs Catastrophic Forgetting

Mehta et al. [40] have investigated the entanglement between continual learning and pretraining, highlighting that the latter leads the optimization towards wider minima of the loss landscape. As deeply discussed in [8,6], such property is strictly linked to a reduced tendency in incurring forgetting.

On this latter point, we therefore provide an alternate experimental proof of the benefits deriving from pretraining initialization. In particular, we focus on ResNet-18 trained with ER [53] on Split CIFAR-100[3] and measure how each individual layer differs from its initialization. It can be observed that a randomly initialized backbone (Fig. 1, *left*) significantly alters its parameters at all layers while tasks progress, resulting in a very low Centered Kernel Alignment [32] similarity score already at the first CL task. On the contrary, a backbone pretrained on Tiny ImageNet (Fig. 1, *right*) undergoes limited parameter variations in its layers, with the exception of the last residual layer (although to a lesser extent w.r.t. random init.). This latter finding indicates that its pretraining parametrization requires relevant modifications to fit the current training data. This leads to the *catastrophic forgetting* of the source pretraining task: namely, the latter is swiftly forgotten as the network focuses on the initial CL tasks. This is corroborated by the decreasing accuracy for pretraining data of a kNN classifier trained on top of *Layer 3* and *Layer 4* representations in Fig. 1 (*right*).

To sum up, while pretraining is certainly beneficial, the model drifts away from it one task after the other. Hence, only the first task takes full advantage of it; the optimization of later tasks, instead, starts from an initialization that increasingly differs from the one attained by pretraining. This is detrimental, as classes introduced later might be likewise advantaged by the reuse of different pieces of the initial knowledge.

---

[3] This preliminary experiment follows the same setting presented in Sec. 4.1
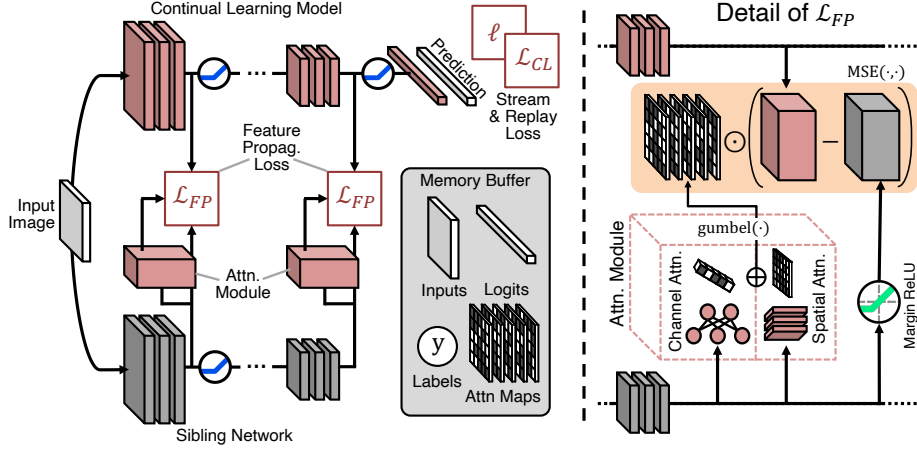
Fig. 2. Overview of TwF and detail of $\mathcal{L}_{\mathrm{FP}}$: Given a batch of samples from the current task or from $\mathcal{B}$, we *i)* extract intermediate features from both the student and fixed sibling backbones at multiple layers; *ii)* compute the corresponding binarized attention maps $\mathbb{M}(\cdot)$; *iii)* pull the attention-masked representations of the two models closer.

3.2 Transfer without Forgetting

To mitigate the issue above, we propose a strategy that enables a continuous transfer between the source task and the incrementally learned target problem.

Feature Propagation. As the training progresses, the input stream introduces new classes that might benefit from the adaptation of specific features of the pretrained model. To enable feature transfer without incurring pretraining forgetting, we maintain a copy of it (the *sibling* model) and adopt an intermediate feature knowledge distillation [55,2,66,26,41] objective. Considering a subset of L layers, we seek to minimize the distance between the activations of the base network $h_\theta^{(l)} \triangleq h_\theta^{(l)}(x)$ and those from its pretrained sibling $\widehat{h}^{(l)} \triangleq h_{\theta^t}^{(l)}(x)$:

$$\mathbb{E}_{x \sim \mathcal{T}_c} \left[\sum_{l=1}^{L} ||h_\theta^{(l)} - \mathrm{ReLU}_{\mathrm{m}}(\widehat{h}^{(l)})||_2^2 \right], \tag{2}$$

where c is the current task and $\mathrm{ReLU}_{\mathrm{m}}(\cdot)$ indicates the application of a margin ReLU activation [26]. It is noted that the objective outlined by Eq. 2 leads the CL model to focus on mirroring the internal representations of the pretrained teacher and maximizing transfer. However, focusing on the latter solely can lead to excessive rigidity, thus preventing the model from fitting the data from the current task altogether. On these grounds, we take inspiration from [66] and use a weighted version of Eq. 2. In particular, an apposite learnable module computes a gating attention map $\mathbb{M}(\cdot)$ over the feature maps of the sibling, which serves as

a binary mask selecting which spatial regions have to be aligned. The resulting objective is consequently updated as follows:

$$\mathbb{E}_{x \sim \mathcal{T}_c}\left[\sum_{l=1}^{L}||\mathbb{M}(\widehat{h}^{(l)}) \odot \left(h_\theta^{(l)} - \mathrm{ReLU}_{\mathrm{m}}(\widehat{h}^{(l)})\right)||_2^2\right], \tag{3}$$

where $\odot$ indicates the Hadamard product between two tensors of the same dimensions. The attention maps $\mathbb{M}(\cdot)$ are computed through specific layers, whose architectural design follows the insights provided in [47]. Specifically, they forward the input activation maps into two parallel branches, producing respectively a Channel Attention $\mathbb{M}_{\mathrm{Ch}}(\cdot)$ map and a Spatial Attention $\mathbb{M}_{\mathrm{Sp}}(\cdot)$ map. These two intermediate results are summed and then activated through a binary Gumbel-Softmax sampling [29], which allows us to model discrete *on-off* decisions regarding which information we want to propagate. In formal terms:

$$\mathbb{M}(\widehat{h}^{(l)}) \triangleq \mathrm{gumbel}(\mathbb{M}_{\mathrm{Ch}}(\widehat{h}^{(l)}) + \mathbb{M}_{\mathrm{Sp}}(\widehat{h}^{(l)})). \tag{4}$$

The Spatial Attention $\mathbb{M}_{\mathrm{Sp}}(\widehat{h}^{(l)})$ regulates the propagation of spatially localized information and is obtained by stacking four convolutional layers [47] with different configurations (*i.e.*, the kernel sizes and dilation rates – please refer to supplementary materials for additional details):

$$\mathbb{M}_{\mathrm{Sp}}(\widehat{h}^{(l)}) \triangleq \mathrm{C}_{1\times1} \circ \mathrm{C}_{3\times3} \circ \mathrm{C}_{3\times3} \circ \mathrm{C}_{1\times1}(\widehat{h}^{(l)}), \tag{5}$$

where C denotes a sequence of convolutional, batch normalization, and ReLU activation layers. On the other hand, the Channel Attention $\mathbb{M}_{\mathrm{Ch}}(\widehat{h}^{(l)})$ estimates the information across the channels of $\widehat{h}^{(l)}$; in its design, we draw inspiration from the formulation proposed in [28]. Formally, considering the result $\widehat{h}^{(l)}_{\mathrm{GAP}}$ of the Global Average Pooling (GAP) applied on top of $\widehat{h}^{(l)}$, we have:

$$\mathbb{M}_{\mathrm{Ch}}(\widehat{h}^{(l)}) \triangleq \tanh(\mathrm{BN}(W_1^\mathsf{T}\widehat{h}^{(l)}_{\mathrm{GAP}})) \cdot \sigma(\mathrm{BN}(W_2^\mathsf{T}\widehat{h}^{(l)}_{\mathrm{GAP}})) + W_3^\mathsf{T}\widehat{h}^{(l)}_{\mathrm{GAP}}, \tag{6}$$

where W_1, W_2, and W_3 are the weights of three fully connected layers organized in parallel and BN indicates the application of batch normalization.

Diversity loss. Without a specific loss term supervising the attention maps, we could incur in useless behaviors, *e.g.*, all binary gates being either on or off, or some channels being always propagated and some others not. While recent works provide a target expected activation ratio [1,58] as a countermeasure, we encourage the auxiliary modules to assign different propagation gating masks to different examples. The intuition is that each example has its own preferred subset of channels to be forwarded from the sibling. To do so, we include an additional auxiliary loss term [42] as follows:

$$\mathcal{L}_{\mathrm{AUX}} \triangleq -\lambda \sum_{l=1}^{L} \mathbb{E}_{x_1,\ldots,x_n \sim \mathcal{T}_c}\left[\sum_{j=1}^{n} \log \frac{e^{g_{ij}^\mathsf{T} g_{ij}/T}}{\frac{1}{n}\sum_{k=1}^{n} e^{g_{ij}^\mathsf{T} g_{ik}/T}}\right], \tag{7}$$

$$g_{ij} \triangleq \mathrm{NORM}(\mathrm{GAP}(\mathbb{M}(\widehat{h}^{(l)}(x_j)))),$$

where n indicates the batch size, NORM a normalization layer, T a temperature and finally λ is a scalar weighting the contribution of this loss term to the overall objective. In practice, we ask each vector containing channel-wise average activity to have a low dot product with vectors of other examples.

3.3   Knowledge Replay

The training objective of Eq. 3 is devised to facilitate selective feature transfer between the in-training model and the immutable sibling. However, to prevent forgetting tied to previous CL tasks to the greatest extent, the model should also be provided with a targeted strategy. We thus equip the continual learner with a small memory buffer $\mathcal{B}$ (populated with examples from the input stream via *reservoir sampling* [65]) and adopt the simple labels and logits replay strategy proposed in [8]:

$$\mathcal{L}_{\text{CL}} \triangleq \mathop{\mathbb{E}}_{(x,y,l)\sim\mathcal{B}} \left[\alpha \cdot ||f_{(\theta,\phi)}(x) - l||_2^2 + \beta \cdot \ell(y, f_{(\theta,\phi)}(x)) \right], \qquad (8)$$

where (x, y, l) is a triplet of example, label and original network responses $l = f(x)$ recorded at the time of sampling and α, β are scalar hyperparameters. Although extremely beneficial, we remark that the model need not optimize $\mathcal{L}_{\text{CL}}$ to achieve basic robustness against catastrophic forgetting (as shown in Sec. 5): preserving pretraining features already serves this purpose.

Replaying past propagation masks. With the purpose of protecting the feature propagation formulated in Eq. 3 from forgetting, we also extend it to replay examples stored in memory. It must be noted that doing so requires taking additional steps to prevent cross-task interference; indeed, simply applying Eq. 3 to replay items would apply the feature propagation procedure unchanged to all tasks, regardless of the classes thereby included. For this reason, we take an extra step and make all batch normalization and fully connected layers in Eq. 4, 5 and 6 conditioned [17] w.r.t. the CL task. Consequently, we add to $\mathcal{B}$ for each example x both its task label t and its corresponding set of binary attention maps $m = (m^1, ..., m^l)$ generated at the time of sampling. Eq. 3 is finally updated as:

$$\mathcal{L}_{\text{FP}} \triangleq \mathop{\mathbb{E}}_{\substack{(x,t=c)\sim\mathcal{T}_c \\ (x;t)\sim\mathcal{B}}} \left[\sum_{l=1}^{L} ||\mathbb{M}(\widehat{h}^{(l)}; t) \odot \left(h^{(l)} - \text{ReLU}_{\text{m}}(\widehat{h}^{(l)}) \right)||_2^2 \right]$$
$$+ \mathop{\mathbb{E}}_{\substack{(x,t,m)\sim\mathcal{B} \\ l=1,...,L}} \left[\text{BCE}\left(\mathbb{M}(\widehat{h}^{(l)}; t), m^{(l)} \right) \right], \qquad (9)$$

where the second term is an additional replay contribution distilling past attention maps, with BCE indicating the binary cross entropy criterion.

Overall objective. Our proposal – dubbed **Transfer without Forgetting (TwF)** – optimizes the following training objective, also summarized in Fig. 2:

$$\min_{\theta,\phi} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}_c} \left[\ell(y_j^i, f_{(\theta,\phi)}(x_j^i))\right] + \mathcal{L}_{\text{CL}} + \mathcal{L}_{\text{FP}} + \mathcal{L}_{\text{AUX}}. \tag{10}$$

We remark that: *i)* while TwF requires keeping a copy of the pretrained model during training, this does not hold at inference time; *ii)* similarly, task labels t are not needed during inference but only while training, which makes TwF capable of operating under both the Task-IL and Class-IL CL settings [63]; *iii)* the addition of t and m in $\mathcal{B}$ induces a limited memory overhead: t can be obtained from the stored labels y for typical classification tasks with a fixed number of classes per task, while m is a set of Boolean maps that is robust to moderate re-scaling (as we demonstrate by storing m at half resolution for our experiments in Sec. 4). We finally point out that, as maps m take discrete binary values, one could profit from lossless compression algorithms (such as Run-Length Encoding [54] or LZ77 [71]) and thus store a compressed representation into the memory buffer. We leave the comprehensive investigation of this application to future works.

4   Experiments

4.1   Experimental Setting

Metrics. We assess the overall performance of the models in terms of *Final Average Accuracy* (FAA), defined as the average accuracy on all seen classes after learning the last task, and *Final Forgetting* [12] (FF), defined as:

$$\text{FF} \triangleq \frac{1}{T-1} \sum_{i=0}^{T-2} \max_{t\in\{0,...,T-2\}} \{a_i^t - a_i^{T-1}\}, \tag{11}$$

where a_i^t denotes the accuracy on task τ_i after training on the tth task.

Settings. We report results on two common protocols [63]: *Task-Incremental Learning* (Task-IL), where the model must learn to classify samples only from within each task, and *Class-Incremental Learning* (Class-IL), where the model must gradually learn the overall classification problem. The former scenario is a relaxation of the latter, as it provides the model with the task identifier of each sample at test time; for this reason, we focus our evaluation mainly on the Class-IL protocol, highlighted as a more realistic and challenging benchmark [20,3].

Datasets. We initially describe a scenario where the transfer of knowledge from the pretrain is facilitated by the similarity between the two distributions. Precisely, we use **CIFAR-100** [33] as the pretrain dataset and then evaluate the models on **Split CIFAR-10** [70] (5 binary tasks) (see Tab. 1). In Tab. 2 we envision a second and more challenging benchmark, which relies on **Split**

CIFAR-100 [70] with the opportunity to benefit from the knowledge previously learned on **Tiny ImageNet** [62]. Due to the size mismatch between CIFAR-100 and the samples from Tiny ImageNet, we resize the latter to 32×32 during pretraining. The last scenario (Tab. 3) involves pretraining on ImageNet [18] and learning incrementally **Split CUB-200** [13,69], split into 10 tasks of 20 classes each. With an average of only 29.97 images per class and the use of higher-resolution input samples (resized to 224×224), this benchmark is the most challenging. We use ResNet18 [25] for all experiments involving Split CIFAR-10 and Split CIFAR-100, as in [50,8], while opting for ResNet50 on Split CUB-200. The supplementary materials report other details on the experimental protocols.

Competitors. We focus our comparison on state-of-the-art rehearsal algorithms, as they prevail on most benchmarks in literature [14,8,63].

- **Experience Replay (ER)** [49,53] is the first embodiment of a rehearsal strategy that features a small memory buffer containing an *i.i.d.* view of all the tasks seen so far. During training, data from the stream is complemented with data sampled from the buffer. While this represents the most straightforward use of a memory in a CL scenario, ER remains a strong baseline, albeit with a non-negligible memory footprint.
- **Dark Experience Replay (DER)** [8] envisions a self-distillation [22] constraint on data stored in the memory buffer and represents a simple extension to the basic rehearsal strategy of ER. In this work, we compare against DER++, which includes both ER and DER objectives.
- **Incremental Classifier and Representation Learning (iCaRL)** [50] tackle catastrophic forgetting by distilling the responses of the model at the previous task boundary and storing samples that better represent the current task. In addition to simple replay, those *exemplars* are used to compute class-mean prototypes for nearest-neighbor classification.
- **ER with Asymmetric Cross-Entropy (ER-ACE)** [10] recently introduced a method to alleviate class imbalances to ER. The authors obtain a major gain in accuracy by simply separating the cross-entropy contribution of the classes in the current batch and that of the ones in the memory buffer.
- **Contrastive Continual Learning (CO^2L)** [11] proposes to facilitate knowledge transfer from samples stored in the buffer by optimizing a contrastive learning objective, avoiding any potential bias introduced by a cross-entropy objective. To perform classification, a linear classifier needs to be first trained on the exemplars stored in the buffer.

In addition, we also include results from two popular regularization methods. **Online Elastic Weight Consolidation (oEWC)** [31] penalizes changes on the most important parameters by means of an online estimate of the Fisher Information Matrix evaluated at task boundaries. **Learning without Forgetting (LwF)** [34] includes a distillation target similar to iCaRL but does not store any exemplars. We remark that **all competitors undergo an initial pretraining phase** prior to CL, thus ensuring a fair comparison.

Table 1. Final Average Accuracy (FAA) [↑] and Final Forgetting (FF) [↓] on Split CIFAR-10 w. pretrain on CIFAR-100.

FAA (FF)	**Split CIFAR-10** (*pretr. CIFAR-100*)			
Method	**Class-IL**		**Task-IL**	
Joint (UB)	92.89 (−)		98.38 (−)	
Finetune	19.76 (98.11)		84.05 (17.75)	
oEwC [57]	26.10 (88.85)		81.84 (19.50)	
LwF [34]	19.80 (97.96)		86.41 (14.35)	
Buffer Size	500	5120	500	5120
ER [53]	67.24 (38.24)	86.27 (13.68)	96.27 (2.23)	97.89 (0.55)
CO^2L [11]	75.47 (21.80)	87.59 (9.61)	96.77 (1.23)	97.82 (0.53)
iCaRL [50]	76.73 (14.70)	77.95 (12.90)	97.25 (0.74)	97.52 (0.15)
DER++ [8]	78.42 (20.18)	87.88 (8.02)	94.25 (4.46)	96.42 (1.99)
ER-ACE [10]	77.83 (10.63)	86.20 (5.58)	96.41 (2.11)	97.60 (0.66)
TwF (ours)	**83.65 (11.59)**	**89.55 (6.85)**	**97.49 (0.86)**	**98.35 (0.17)**

To gain a clearer understanding of the results, all the experiments include the performance of the upper bound (**Joint**), obtained by jointly training on all classes in a non-continua fashion. We also report the results of the model obtained by training sequentially on each task (**Finetune**), *i.e.*, without any countermeasure to forgetting.

4.2   Comparison with State-Of-The-Art

Regularization methods. Across the board, non-rehearsal methods (oEWC and LwF) manifest a profound inability to effectively use the features learned during the pretrain. As those methods are not designed to extract and reuse any useful features from the initialization, the latter is rapidly forgotten, thus negating any knowledge transfer in later tasks. This is particularly true for oEWC, whose objective proves to be both too strict to effectively learn the current task and insufficient to retain the initialization. Most notably, on Split CUB-200 oEWC shows performance lower than Finetune on both Task- and Class-IL.

Rehearsal methods. In contrast, rehearsal models that feature some form of distillation (DER++ and iCaRL) manage to be competitive on all benchmarks. In particular, iCaRL proves especially effective on Split CIFAR-100, where it reaches the second highest FAA even when equipped with a small memory thanks to its *herding* buffer construction strategy. However, this effect is less pronounced on Split CIFAR-10 and Split CUB-200, where the role of pretraining is far more essential due to the similarity of the two distributions for the former and the higher difficulty of the latter. In these settings, we see iCaRL fall short of DER++, which better manages to maintain and reuse the features

Table 2. Accuracy (forgetting) on Split CIFAR-100 w. pretrain on Tiny ImageNet.

FAA (FF)	**Split CIFAR-100** (*pretr. Tiny ImageNet*)			
Method	**Class-IL**		**Task-IL**	
Joint (UB)	75.20 (−)		93.40 (−)	
Finetune	09.52 (92.31)		73.50 (20.53)	
oEwC [57]	10.95 (81.71)		65.56 (21.33)	
LwF [34]	10.83 (90.87)		86.19 (4.77)	
Buffer Size	500	2000	500	2000
ER [53]	31.30 (65.40)	46.80 (46.95)	85.98 (6.14)	87.59 (4.85)
CO^2L [11]	33.40 (45.21)	50.95 (31.20)	68.51 (21.51)	82.96 (8.53)
iCaRL [50]	56.00 (19.27)	58.10 (16.89)	**89.99 (2.32)**	90.75 (1.68)
DER++ [8]	43.65 (48.72)	58.05 (29.65)	73.86 (20.08)	86.63 (6.86)
ER-ACE [10]	53.38 (21.63)	57.73 (17.12)	87.21 (3.33)	88.46 (2.46)
TwF (ours)	**56.83 (23.89)**	**64.46 (15.23)**	89.82 (3.06)	**91.11 (2.24)**

available from its initialization. Moreover, we remark that iCaRL and DER++ show ranging Class-IL performance in different tasks, whereas our method is much less sensitive to the specific task at hand.

While it proves effective on the easier Split CIFAR-10 benchmark, CO^2L does not reach satisfactory results on either Split CIFAR-100 or Split CUB-200. We ascribe this result to the high sensitivity of this model to the specifics of its training process (*e.g.*, to the applied transforms and the number of epochs required to effectively train the feature extractor with a contrastive loss). Remarkably, while we extended the size of the batch in all experiments with CO^2L to 256 to provide a large enough pool of negative samples, it still shows off only a minor improvement on non-rehearsal methods for Split CUB-200.

Interestingly, while both ER and ER-ACE do not feature distillation, we find their performance to be competitive for large enough buffers. In particular, the asymmetric objective of ER-ACE appears less sensitive to a small memory buffer but always falls short of DER++ when this constraint is less severe.

Transfer without Forgetting. Finally, results across all proposed benchmarks depict our method (TwF) as consistently outperforming all the competitors, with an average gain of 4.81% for the Class-IL setting and 2.77% for the Task-IL setting, w.r.t. the second-best performer across all datasets (DER++ and ER-ACE, respectively). This effect is especially pronounced for smaller buffers on Split CIFAR-10 and Split CUB-200, for which the pretrain provides a valuable source of knowledge to be transferred. We argue that this proves the efficacy of our proposal to retain and adapt features available from initialization through distillation. Moreover, we remark that its performance gain is consistent in all settings, further attesting to the resilience of the proposed approach.

Table 3. Accuracy (forgetting) on Split CUB-200 w. pretrain on ImageNet.

FAA (FF)	Split CUB-200	(*pretr. ImageNet*)		
Method	**Class-IL**		**Task-IL**	
Joint (UB)	78.54 (−)		86.48 (−)	
Finetune	8.56 (82.38)		36.84 (50.95)	
oEwC [57]	8.20 (71.46)		33.94 (40.36)	
LwF [34]	8.59 (82.14)		22.17 (67.08)	
Buffer Size	400	1000	400	1000
ER [53]	45.82 (40.76)	59.88 (25.65)	75.26 (9.82)	80.19 (4.52)
CO^{2}L [11]	8.96 (32.04)	16.53 (20.99)	22.91 (26.42)	35.79 (16.61)
iCaRL [50]	46.55 (12.48)	49.07 (11.24)	68.90 (3.14)	70.57 (3.03)
DER++ [8]	56.38 (26.59)	67.35 (13.47)	77.16 (7.74)	82.00 (3.25)
ER-ACE [10]	48.18 (25.79)	58.19 (16.56)	74.34 (9.78)	78.27 (6.09)
TwF (ours)	**57.78 (18.32)**	**68.32 (6.74)**	**79.35 (5.77)**	**82.81 (2.14)**

Table 4. Impact of each loss term and of using no memory buffer on TwF. Results given in the Class-IL scenario following the same experimental settings as Tab.1-3.

$\mathcal{L}_{CL}$	$\mathcal{L}_{FP}$	$\mathcal{L}_{AUX}$	Split CIFAR-10			Split CIFAR-100			Split CUB-200		
Buffer Size			w/o/buf.	500	5120	w/o/buf.	500	2000	w/o/buf.	400	1000
✓	✓	✓	−	**83.65**	**89.55**	−	**56.83**	**64.46**	−	**59.67**	**68.32**
✓	✗	✗	−	75.79	87.54	−	44.01	57.84	−	56.53	67.29
✓	✓	✗	−	83.29	89.53	−	55.50	63.53	−	59.06	67.83
✗	✓	✗	60.07	62.63	62.75	49.14	50.20	50.22	37.57	38.43	38.93
✗	✓	✓	60.90	63.19	63.79	49.74	50.88	50.52	37.99	39.20	39.31

5   Ablation Studies

Breakdown of the individual terms of TwF. To better understand the importance of the distinct loss terms in Eq. 10 and their connection, we explore their individual contribution to the final accuracy of TwF in Tab. 4. Based on these results, we make the following observations: *i)* $\mathcal{L}_{CL}$ is the most influential loss term and it is indispensable to achieve results in line with the SOTA; *ii)* $\mathcal{L}_{FP}$ applied on top of $\mathcal{L}_{CL}$ induces better handling of pretraining transfer, as testified by the increased accuracy; *iii)* $\mathcal{L}_{AUX}$ on top of $\mathcal{L}_{FP}$ reduces activation overlapping and brings a small but consistent improvement.

Further, in the columns labeled as w/o/buf., we consider what happens if TwF is allowed **no replay example at all** and only optimizes $\mathcal{L}_{FP}$ and $\mathcal{L}_{AUX}$ on current task examples. Compared to oEwC in Tab. 1-3 – the best non-replay method in our experiments – we clearly see preserving pretraining features is in itself a much more effective approach, even with rehearsal is out of the picture.
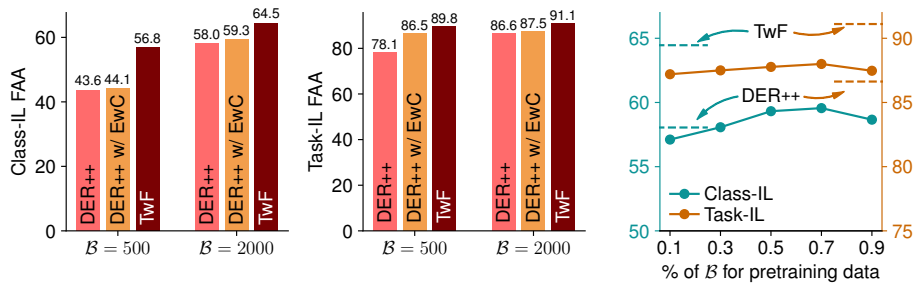
Fig. 3. Class-IL (left) and Task-IL (center) FAA performance comparison of our proposal with different possible methods to retain knowledge from pretrain. (Right) Influence of different allocation rates of pretrain examples in $\mathcal{B}$ for DER++, $|\mathcal{B}| = 2000$.

Alternatives for the preservation of pretraining knowledge. TwF is designed to both preserve pretraining knowledge and facilitate its transfer. However, other approaches could be envisioned for the same purpose. Hence, we compare here TwF with two alternative baselines for pretraining preservation.

Pretraining preservation with EwC. We complement a strong approach such as DER++ with an additional regularization term based on EwC:

$$\mathcal{L}_{\text{EwC}} = \lambda(\theta - \theta^t)^T \operatorname{diag}(F)(\theta - \theta^t), \tag{12}$$

where $\operatorname{diag}(F)$ indicates the diagonal of the empirical Fisher Information Matrix, estimated on the pretraining data at the optimum θ^t. When equipped with this additional loss term, DER++ is anchored to its initialization and prevented from changing its pretraining weights significantly, while its replay-based loss term prevents forgetting of knowledge acquired in previous tasks. As shown by Fig. 3 (left, center), the EwC loss allows DER++ to improve its accuracy on Split CIFAR-100 with Tiny ImageNet pretraining (especially in the Task-IL setting). However, this improvement is not actively incentivizing feature reuse and thus falls short of TwF. We finally remark that TwF and DER++ w/ EwC have a comparable memory footprint (both retain the initialization checkpoint).

Pretraining preservation through rehearsal. An alternative for preserving the source knowledge is to assume that pretraining data is available and can be treated as an auxiliary data stream [5]. To evaluate this strategy with a bounded memory footprint, we test our baseline method (DER++) on Split CIFAR-100 with different percentages of the buffer dedicated to pretraining images (from Tiny ImageNet). The results shown in Fig. 3 (right) confirm our main claim: DER++ coupled with pretraining rehearsal improves over DER++ with only pretraining. This finding proves that, if pretraining is available, it is beneficial to guard it against catastrophic forgetting.

Table 5. Dissimilar pretrain tasks: accuracy on CIFAR-100 pretrained on SVHN.

FAA (FF)	Class-IL		Task-IL	
Buffer size	500	2000	500	2000
iCaRL [50]	39.59 (21.81)	42.02 (18.78)	78.89 (4.04)	80.65 (2.24)
DER++ [8]	36.46 (53.47)	52.29 (24.04)	75.05 (16.22)	83.36 (8.04)
TwF (ours)	**43.56 (40.02)**	**56.15 (21.51)**	**80.89 (10.12)**	**87.30 (3.12)**

Furthermore, we highlight that TwF outperforms the baseline introduced here. When replaying pretraining data, indeed, the model has to maintain its predictive capabilities on the classes of the source task, *i.e.*, we enforce both backward and forward transfer. TwF, instead, allows the model to disregard the classes of the source dataset, as long as the transfer of its internal representations favors the learning of new tasks ($\Rightarrow$ **it only enforces forward transfer**). This substantial distinction helps to understand the merits of TwF: namely, a full but still functional exploitation of the pretraining knowledge.

Role of pretraining datasets. Here, we seek to gain further proof of our claim about the ability of TwF to adapt features from the pretrain. Specifically, we study a scenario where the source data distribution and the target one are highly dissimilar: namely, we first pretrain a ResNet18 backbone on SVHN [43] and then follow with Split CIFAR-100. We compare our model with the second-best performer from Tab. 2, *i.e.*, iCaRL, and DER++. The results, reported in Tab. 5, suggest that our method outranks the competitors not only when pretrained on a similar dataset – as in Tab. 2 – but also when the tasks are very dissimilar. We argue that this result further shows the ability of TwF to identify which pretraining features are really advantageous to transfer.

6   Conclusions

We introduced Transfer without Forgetting, a hybrid method combining Rehearsal and Feature transfer, designed to exploit pretrained weights in an incremental scenario. It encourages feature sharing throughout all tasks, yielding a stable performance gain across multiple settings. We also show that TwF outperforms other hybrid methods based on rehearsal and regularization and that it is able to profit even from pretraining on a largely dissimilar dataset.

References

1. Abati, D., Tomczak, J., Blankevoort, T., Calderara, S., Cucchiara, R., Bejnordi, B.E.: Conditional channel gated networks for task-aware continual learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2020)
2. Aguilar, G., Ling, Y., Zhang, Y., Yao, B., Fan, X., Guo, C.: Knowledge distillation from internal representations. In: Proceedings of the AAAI Conference on Artificial Intelligence (2020)
3. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient based sample selection for online continual learning. In: Advances in Neural Information Processing Systems (2019)
4. Allegretti, S., Bolelli, F., Pollastri, F., Longhitano, S., Pellacani, G., Grana, C.: Supporting Skin Lesion Diagnosis with Content-Based Image Retrieval. In: International Conference on Pattern Recognition (2021)
5. Bellitto, G., Pennisi, M., Palazzo, S., Bonicelli, L., Boschini, M., Calderara, S., Spampinato, C.: Effects of auxiliary knowledge on continual learning. In: International Conference on Pattern Recognition (2022)
6. Boschini, M., Bonicelli, L., Buzzega, P., Porrello, A., Calderara, S.: Class-incremental continual learning into the extended der-verse. arXiv preprint arXiv:2201.00766 (2022)
7. Boschini, M., Buzzega, P., Bonicelli, L., Porrello, A., Calderara, S.: Continual semi-supervised learning through contrastive interpolation consistency. arXiv preprint arXiv:2108.06552 (2021)
8. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark Experience for General Continual Learning: a Strong, Simple Baseline. In: Advances in Neural Information Processing Systems (2020)
9. Buzzega, P., Boschini, M., Porrello, A., Calderara, S.: Rethinking Experience Replay: a Bag of Tricks for Continual Learning. In: International Conference on Pattern Recognition (2020)
10. Caccia, L., Aljundi, R., Asadi, N., Tuytelaars, T., Pineau, J., Belilovsky, E.: New Insights on Reducing Abrupt Representation Change in Online Continual Learning. In: International Conference on Learning Representations (2022)
11. Cha, H., Lee, J., Shin, J.: Co2l: Contrastive continual learning. In: IEEE International Conference on Computer Vision (2021)
12. Chaudhry, A., Dokania, P.K., Ajanthan, T., Torr, P.H.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: Proceedings of the European Conference on Computer Vision (2018)
13. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient Lifelong Learning with A-GEM. In: International Conference on Learning Representations (2019)
14. Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P.K., Torr, P.H., Ranzato, M.: On tiny episodic memories in continual learning. In: International Conference on Machine Learning Workshop (2019)
15. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning (2020)
16. De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)

17. De Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O., Courville, A.C.: Modulating early visual processing by language. In: Advances in Neural Information Processing Systems (2017)
18. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2009)
19. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (2019)
20. Farquhar, S., Gal, Y.: Towards Robust Evaluations of Continual Learning. In: International Conference on Machine Learning Workshop (2018)
21. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning (2017)
22. Furlanello, T., Lipton, Z.C., Tschannen, M., Itti, L., Anandkumar, A.: Born again neural networks. In: International Conference on Machine Learning (2018)
23. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: IEEE International Conference on Computer Vision (2017)
24. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: IEEE International Conference on Computer Vision (2015)
25. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2016)
26. Heo, B., Kim, J., Yun, S., Park, H., Kwak, N., Choi, J.Y.: A comprehensive overhaul of feature distillation. In: IEEE International Conference on Computer Vision (2019)
27. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: Neural Information Processing Systems Workshops (2015)
28. Ilse, M., Tomczak, J., Welling, M.: Attention-based deep multiple instance learning. In: International Conference on Machine Learning (2018)
29. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: International Conference on Learning Representations (2017)
30. Jang, Y., Lee, H., Hwang, S.J., Shin, J.: Learning what and where to transfer. In: International Conference on Machine Learning (2019)
31. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences (2017)
32. Kornblith, S., Norouzi, M., Lee, H., Hinton, G.: Similarity of neural network representations revisited. In: International Conference on Machine Learning (2019)
33. Krizhevsky, A., et al.: Learning multiple layers of features from tiny images. Tech. rep., Citeseer (2009)
34. Li, Z., Hoiem, D.: Learning without forgetting. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
35. Long, M., Cao, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: Advances in Neural Information Processing Systems (2018)
36. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. In: International Conference on Machine Learning (2017)
37. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. In: Advances in Neural Information Processing Systems (2017)

38. Mallya, A., Lazebnik, S.: Packnet: Adding multiple tasks to a single network by iterative pruning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2018)
39. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. Psychology of learning and motivation (1989)
40. Mehta, S.V., Patil, D., Chandar, S., Strubell, E.: An empirical investigation of the role of pre-training in lifelong learning. In: International Conference on Machine Learning (2021)
41. Monti, A., Porrello, A., Calderara, S., Coscia, P., Ballan, L., Cucchiara, R.: How many observations are enough? knowledge distillation for trajectory forecasting. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2022)
42. Müller, R., Kornblith, S., Hinton, G.: Subclass distillation. arXiv preprint arXiv:2002.03936 (2020)
43. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: Advances in Neural Information Processing Systems (2011)
44. Nichol, A., Schulman, J.: On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999 (2018)
45. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering (2009)
46. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: A review. Neural Networks (2019)
47. Park, J., Woo, S., Lee, J.Y., Kweon, I.S.: Bam: Bottleneck attention module. In: British Machine Vision Conference (2018)
48. Porrello, A., Vincenzi, S., Buzzega, P., Calderara, S., Conte, A., Ippoliti, C., Candeloro, L., Di Lorenzo, A., Dondona, A.C.: Spotting insects from satellites: modeling the presence of culicoides imicola through deep cnns. In: International Conference on Signal-Image Technology & Internet-Based Systems (2019)
49. Ratcliff, R.: Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. Psychological Review (1990)
50. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2017)
51. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (2015)
52. Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., Tesauro, G.: Learning to Learn without Forgetting by Maximizing Transfer and Minimizing Interference. In: International Conference on Learning Representations (2019)
53. Robins, A.: Catastrophic forgetting, rehearsal and pseudorehearsal. Connection Science (1995)
54. Robinson, A.H., Cherry, C.: Results of a prototype television bandwidth compression scheme. In: Proceedings of the IEEE (1967)
55. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. In: International Conference on Learning Representations (2015)
56. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)

57. Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R., Hadsell, R.: Progress & compress: A scalable framework for continual learning. In: International Conference on Machine Learning (2018)
58. Serra, J., Suris, D., Miron, M., Karatzoglou, A.: Overcoming Catastrophic Forgetting with Hard Attention to the Task. In: International Conference on Machine Learning (2018)
59. Shaheen, K., Hanif, M.A., Hasan, O., Shafique, M.: Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks. Journal of Intelligent & Robotic Systems (2022)
60. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. Nature (2016)
61. Smith, J., Balloch, J., Hsu, Y.C., Kira, Z.: Memory-efficient semi-supervised continual learning: The world is its own replay buffer. In: International Joint Conference on Neural Networks (2021)
62. Stanford: Tiny ImageNet Challenge (CS231n) (2015), `https://www.kaggle.com/c/tiny-imagenet`
63. van de Ven, G.M., Tolias, A.S.: Three continual learning scenarios. In: Neural Information Processing Systems Workshops (2018)
64. Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., et al.: Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature (2019)
65. Vitter, J.S.: Random sampling with a reservoir. ACM Transactions on Mathematical Software (1985)
66. Wang, K., Gao, X., Zhao, Y., Li, X., Dou, D., Xu, C.Z.: Pay attention to features, transfer learn faster cnns. In: International Conference on Learning Representations (2019)
67. Yim, J., Joo, D., Bae, J., Kim, J.: A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2017)
68. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems (2014)
69. Yu, L., Twardowski, B., Liu, X., Herranz, L., Wang, K., Cheng, Y., Jui, S., Weijer, J.v.d.: Semantic drift compensation for class-incremental learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (2020)
70. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. In: International Conference on Machine Learning (2017)
71. Ziv, J., Lempel, A.: A universal algorithm for sequential data compression. IEEE Transactions on information theory (1977)