Continually Learning Self-Supervised Representations with Projected Functional Regularization

Alex Gomez-Villa[1], Bartlomiej Twardowski[1], Lu Yu[2], Andrew D. Bagdanov[3], Joost van de Weijer[1]
[1]Computer Vision Center, Universitat Autonoma de Barcelona, Barcelona, Spain
[2]School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, China
[3]MICC, University of Florence, Florence, Italy
{agomezvi,btwardowski,luyu,joost}@cvc.uab.es, andrew.bagdanov@unifi.it

Abstract

Recent self-supervised learning methods are able to learn high-quality image representations and are closing the gap with supervised approaches. However, these methods are unable to acquire new knowledge incrementally – they are, in fact, mostly used only as a pre-training phase over IID data. In this work we investigate self-supervised methods in continual learning regimes without any replay mechanism. We show that naive functional regularization, also known as feature distillation, leads to lower plasticity and limits continual learning performance. Instead, we propose Projected Functional Regularization in which a separate temporal projection network ensures that the newly learned feature space preserves information of the previous one, while at the same time allowing for the learning of new features. This prevents forgetting while maintaining the plasticity of the learner. Comparison with other incremental learning approaches applied to self-supervision demonstrates that our method obtains competitive performance in different scenarios and on multiple datasets.

1. Introduction

Self-supervised learning aims to learn high-quality image representations without the need for human annotations. A recent set of works has shown that self-supervised learning can achieve performance close to that of supervised learning [5, 10, 12, 23], and that learned representations transferred to downstream tasks are sometimes even superior to fully-supervised representation learning [6]. These methods learn representations that are invariant with respect to a set of data augmentations. They are typically trained with contrastive losses in which multiple views of the same image (computed by applying different data augmentations) are mapped close together, whereas representations of other images are mapped far away. However, several methods show that only encouraging similarity between views from the same image (without any explicit loss to promote the distancing of negative pairs) can also obtain excellent performance [12, 23]. These methods apply various mechanisms to prevent trivial solutions, including asymmetric architectures and the use of momentum updates of the model.

Recent works on self-supervised learning have in common that they assume that all training data is available during the training process. However, in many real-world applications the learner must cope with non-stationary data in which they are exposed to tasks with varying distributions of data. Continual learning relaxes the IID assumption that underlies most learning methods and studies the design of algorithms that learn from data with shifting distributions. Naively training a learner on such data, for example by simply continuing stochastic gradient descent, leads to catastrophic forgetting [46]. A variety of approaches have been proposed including various types of regularization [1, 32, 37, 69], data replay [7, 30, 53, 65], pseudo replay [58, 64], and growing architectures [55]. Even though there is some work on unsupervised continual learning [21, 36, 43, 55], the vast majority of existing work is on supervised continual learning [49, 51].

Earlier works on self-supervised learning was based on pretext tasks like predicting rotation [22], determining patch position [17], or solving jigsaw puzzles in images [47]. Labels for these discriminative pretext tasks can be automatically computed and allow learning of meaningful feature representations of images. Recently, researchers are adapting contrastive methods for unlabeled data and operating more at an instance-level augmentation while looking for similarity or contrastive samples [5, 10, 23, 68]. These methods rely heavily on stochastic data augmentation to produce enough similar examples to learn representations. Negative examples are randomly sampled or not used at all [12]. The results are impressive and are competitive with many supervised methods on downstream tasks [6].

We propose an approach to *continual* self-supervised learning that is able to learn high-quality visual feature representations from non-IID data. The learner is exposed to a changing distribution and, while learning new features on current task data, should prevent forgetting of previously acquired knowledge. These representations should, at the end of training, be applicable to a wide range of downstream tasks. We focus on the more restrictive, memory-free continual learning setting in which the learner is not allowed to store any samples from previous tasks. This scenario is realistic in many scenarios where data privacy and security is fundamental and often legislatively regulated.

The main contributions of this work are twofold. First, we propose a new method, called *projected functional regularization*, to alleviate forgetting during unsupervised representation learning without the need for an external memory of samples from previous tasks. This technique is an extension of Learning without Forgetting (LwF) and distillation in feature space. To improve the plasticity of the method we introduce a *temporal projection network* that provides more freedom to learn features from the current task. Secondly, we propose a set of experiments over benchmark datasets to compare with other state-of-the-art methods and use different scenarios to evaluate the functional projection role in the context of continual self-supervised representation learning. We show that the additional projection to past tasks results in better representation learning during class incremental training sessions. Without any adjustment, evaluation on a truly class incremental scenario – with only a single class per task, where many class incremental methods cannot be directly applied – our method still prevents forgetting and is able to progressively learn new features. Furthermore, we confirm that our method is generic and the results are not restricted to a particular self-supervised learning approach. In a variety of experimental settings the transferability of the learned features to different downstream tasks is maintained, confirming that the network is incrementally learning more robust representations. The influence of the regularization strength is analyzed for different regularization methods applied to self-supervised continual learning and results clearly shows the benefit of the proposed additional projector resulting both in improved plasticity (i.e. lower intransigence) and less forgetting.

2. Related Work

Both self-supervised and continual learning have gathered increasing interest in recent years. We briefly review the literature on both topics before articulating our contribution which combines elements of both in the form of continual self-supervised representation learning.

Self-supervised learning. Self-supervised learning has proved useful for many applications. In order to learn representations useful for a downstream task, a self-supervised pretext task can be introduced to avoid supervision. Many pretext tasks were investigated for learning image representations, including rotation prediction [22], solving jigsaw puzzles [47], determining relative patch positions [17], predicting surrogate classes [18], and image colorization [71]).

In the last few years, the gap between supervised and self-supervised learning is being closed. This is primarily due to methods based on data augmentation and contrastive-like learning in which two samples are considered either similar or different to each other. This has links to earlier contrastive methods used in metric learning [25] and some extensions using triplet losses [63]. However, in the unsupervised setting without labels, different approaches must be used for creating such pairs. In SimCLR [10], similar samples are created by augmenting an input image with a random distortion, while dissimilar ones are chosen by random. To make contrastive training more efficient, the MoCo method [11, 26] uses a memory bank for learned embeddings which enables efficient sampling. This memory is kept synchronized with the rest of the network during training by using a momentum encoder. The SwAV approach uses online clustering over the embedded samples [5]. SwAV does not sample negative exemplars, however, other cluster prototypes can play the role of negative examples.

Interesting are methods without any explicit contrastive pairs. The BYOL approach proposed by [23] is based on an asymmetric network with an additional MLP predictor between two outputs of the two branches. One branch is kept "offline" and updated by a momentum encoder. SimSiam [12] goes even further and offers a simplified solution without a momentum encoder and moreover works well without a very large mini-batch size. BarlowTwins is another simplified solution like SimSiam which uses a loss function based on correlations between each pair in a current training mini-batch [68]. Negatives are implicitly assumed to be available in each mini-batch. No asymmetry is used by the BarlowTwins network, but a larger embedding size and bigger mini-batches are preferred in this method in comparison to SimSiam.

Continual learning. Existing continual learning methods can be broadly divided into replay-based, architecture-based, and regularization-based methods [14, 44]. Replay-based methods save a small amount of data from previously seen tasks [4, 9] or generate synthetic data with a generative model [62, 70]. Architecture-based method activate different subsets of network parameters for different tasks by allowing model parameters to grow linearly with the number of tasks. Previous works following this strategy include DER [66], Piggyback [42], PackNet [43], DAN [54], HAT [56], Ternary Masks [45] and PathNet [21]. Regularization-based methods add an additional regulariza-

tion term derived from knowledge of previous tasks to the training loss. This can be done by either regularizing the weight space (constraining important parameters) [57, 60] or the functional space (constraining predictions or intermediate features) [13, 19, 31]. EWC [32], MAS [1], REWC [38], SI [69], and RWalk [8] constrain the importance of network parameters to prevent forgetting. Methods such as LwF [37], LwM [16] and BiC [65] instead leverage knowledge distillation to regularize features or predictions.

Our approach, called *Projected Functional Regularization* is a functional regularization approach. Normally, these approaches distill information at the class-prediction level between an old and new model. However, in self-supervised learning this has to be applied to the embedding output. Regularizing the embedding layer is known to undermine plasticity [19], we therefore propose an additional temporal projection network that maps between the latent spaces of current and previous model. We show that this regularization prevents forgetting while obtaining improved plasticity.

Continual representation learning. Continual unsupervised representation learning was investigated by [52] with an approach based on variational autoencoders and a Gaussian mixture model. Still, detection of new clusters and model expansion is necessary. In a very recent paper, the authors used contrastive self-supervised learning with a memory buffer for storing exemplars [40]. They proposed the LUMP method in which images from the current task and previous tasks are combined with CutMix for continual training. One of the main differences between LUMP and our approach is that ours does not require storing any data from previous tasks.

Our contribution is fundamentally different from methods using self-supervised learning to improve the learning of a sequence of supervised tasks [24, 72]. Their objective is not to learn from unlabeled data, but rather to use self-supervised learning to further enrich the feature representation. The hypothesis of these works is that, for class incremental learning scenario, the features learned via self-supervision will be more generic than ones learned from task-bounded discrimination problems.

3. Continual Self-supervised Representation Learning

We begin with a discussion of self-supervised representation learning, and then describe our proposed Projected Functional Regularization (PFR) approach for continual learning of self-supervised representations without the need of any memory or replay.

3.1. Self-supervised representation learning

In recent works on self-supervised learning the aim is to learn a network $f_\theta : \mathcal{X} \rightarrow \mathcal{F}$ that maps from input space $\mathcal{X}$ to output feature representation space $\mathcal{F}$. This network is learned on unlabeled input data x drawn from distribution $\mathcal{D}$. The aim is then to exploit the learned feature representation to perform any variety of downstream tasks. As an example, for the downstream task of classification on some target domain, we have training data $\mathcal{D}^t = \{x_i^t, y_i^t\}$ on which we learn a classifier $g_\phi : \mathcal{F} \rightarrow \mathcal{Y}$ (with $\mathcal{Y}$ being the output space) that minimizes a loss $\mathcal{L} = \ell\left(y^t, \hat{y}^t = g_\phi\left(f_\theta\left(x^t\right)\right)\right)$. Adaptation to the target domain might only optimize the weights ϕ while keeping θ fixed on the target data, or instead might also allow θ to be fine-tuned on the target data.

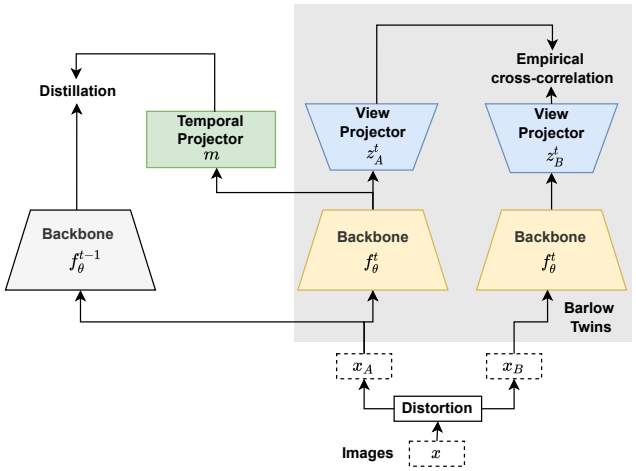


Figure 1. Self-supervised continual learning with Projected Functional Regularization. Instead of performing feature distillation directly between the previous task backbone and the new one, we use a *learned temporal projection* between the two feature spaces.

In this paper we apply the BarlowTwins [68] approach to self-supervised learning of the representation network f_θ. However, the proposed method is general and can be applied to other self-supervised methods. BarlowTwins does not require explicit negative samples and achieves competitive performance while remaining computationally efficient, assuming that negatives are available in each mini-batch to calculate correlation between all samples in it. The BarlowTwins architecture has two branches (see the shaded area in Fig. 1). In both branches a projector network $z : \mathcal{F} \rightarrow \mathcal{Z}$ is used. For the sake of notational simplicity, we do not make explicit the parameters of the network z since it is not used by downstream tasks. The parameters in the backbone and projector layer are shared between the branches.

The network is trained by minimizing an invariance and a redundancy reduction term in the loss function [68]. Here, different augmented views X_A and X_B of the same data samples X are taken from the set of data augmentations $\mathcal{D}^*$.

This leads to the loss defined as:

$$\mathcal{L}_c = \mathbb{E}_{X_A, X_B \sim \mathcal{D}^*} \left[\sum_i (1 - \mathcal{C}_{ii})^2 + \lambda \sum_i \sum_{j \neq i} {\mathcal{C}_{ij}}^2 \right], \quad (1)$$

where λ is a positive constant trade-off parameter between both terms, and where $\mathcal{C}$ is the cross-correlation matrix computed between the representations z of all samples X_A and X_B in a mini-batch indexed by b:

$$\mathcal{C}_{ij} = \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_b \left(z^A_{b,i}\right)^2} \sqrt{\sum_b \left(z^B_{b,j}\right)^2}}, \quad (2)$$

The cross-correlation matrix $\mathcal{C}$ has values in the range of -1.0 (worst) to 1.0 (best) correlation between the projector's outputs: $Z_A = z(f_\theta(X_A))$ and $Z_B = z(f_\theta(X_B))$. The invariance term of the loss function encourages the diagonal elements to be 1. As such the learned embedding will be invariant to the applied data augmentations. At the same time, the second term (redundancy reduction) keeps the off-diagonal elements close to zero and decorrelates the outputs of non-related images.

3.2. Projected Functional Regularization

Current work on self-supervised learning considers the above scenario where the learner has access to a single, large dataset which can be revisited multiple times to learn the optimal feature extractor f_θ. However, for many real-world scenarios this is an unrealistic setup and the learner will have to learn the optimal feature extractor f_θ from a stream of data drawn from a distribution that varies over time.

We consider scenarios in which the learner must learn from a set of tasks, each containing data drawn from a different distribution. We consider the tasks $T = \{1...c\}$ where c is the current task and the data of task t follows the distributions $\mathcal{D}_t$. In this case we would like to find the parameters θ of the feature extractor f_θ that minimize the summed loss over all tasks up to the current one c:

$$\arg\min_\theta \sum_{t=1}^{c} \mathcal{L}^t_c, \quad (3)$$

where $\mathcal{L}^t_c = \mathbb{E}_{X_A, X_B \sim \mathcal{D}^*_t}[\mathcal{L}_c]$ and $\mathcal{L}_c$ is defined as in Eq. 1. Again, $\mathcal{D}^*_t$ refers to the set of augmented samples from $\mathcal{D}_t$ (i.e. the data from task t). However, during continual training we only have access to the data of one task, meaning that the optimal parameters must be found using only the current data $\mathcal{D}_c$. Naive fine-tuning results in parameters optimal for task c, however leads to catastrophic forgetting of knowledge acquired during previous tasks.

Regularization methods are among the most successful at addressing catastrophic forgetting, especially for scenarios where storing of any data from previous tasks is prohibited (which is the objective in this article). Regularization methods can be divided into two important groups: weight regularization approaches [1, 32, 69], which aim to find a set of weights that is both good for the current task while incurring only a small increase in loss on previous tasks, and functional regularization methods (also known as data regularization methods) which optimize weights for new tasks while incurring only minimal changes in the network outputs on previous tasks [29, 48, 61].

The canonical example of functional regularization, called Learning without Forgetting (LwF), was introduced in [29] and is based on knowledge distillation [28]. It was proposed for supervised continual learning and introduces an additional loss that prevents the class predictions of previous tasks on the current data from undergoing large changes while training on the current task data. This loss cannot be directly applied to self-supervised learning since it requires class predictions. However, several continual learning works have extended this idea to feature layers by replacing the modified cross-entropy distillation loss with a distance (typically L1 or L2) which can be applied to any layer output [19, 39, 67]. We will refer to this as *feature distillation* (FD) and it leads to the following loss when training task t:

$$\mathcal{L}^t_c + \lambda_{fd} \mathbb{E}_{x_a, x_b \sim \mathcal{D}^*_i} [\| f_{\theta^t}(x_a) - f_{\theta^{t-1}}(x_a) \| + \| f_{\theta^t}(x_b) - f_{\theta^{t-1}}(x_b) \|], \quad (4)$$

where θ^{t-1} refers to the parameters learned after training up to task $t-1$, and λ_{fd} defines the importance of the regularization term.

The regularization imposed on class predictions in the original LwF paper [29] is not very restrictive: the weights can still significantly vary as long as the final network predictions do not significantly vary. It has been observed in the literature, however, that feature distillation is very restrictive and leads to continual learning methods with low plasticity [19]. In addition, this loss directly penalizes the learning of new features since these would lead to a difference between the new and old model output $\| f_{\theta^t}(x) - f_{\theta^{t-1}}(x) \|$. To address this problem we propose *Projected Functional Regularization* (PFR).

We would like the network to retain previous feature representation while allowing it to learn new features learned on new tasks. These new features should not be directly penalized by regularization. To do so, we introduce a *temporal projection network* $m : \mathcal{Z} \to \mathcal{Z}$ that maps the embedding learned on the current task back to the embedding learned on the previous ones (see Figure 1). The new loss is:

$$\mathcal{L}^t_c + \lambda_{pfr} \mathbb{E}_{x_a, x_b \sim \mathcal{D}^*_i} [\mathcal{S}(m(f_{\theta^t}(x_a)), f_{\theta^{t-1}}(x_a)) + \mathcal{S}(m(f_{\theta^t}(x_b)), f_{\theta^{t-1}}(x_b))] \quad (5)$$

where $S(\cdot,\cdot)$ is a cosine similarity:

$$\mathcal{S}(a,b) = -\frac{a^T b}{||a||_2||b||_2}. \quad (6)$$

New features learned in $f_{\theta^t}(x)$ do not directly result in an increased loss as long as they lie in the null-space of m. As a consequence this loss prevents forgetting of information of previous tasks while maintaining plasticity to adapt to new tasks.

4. Experimental Results

Here we report on a variety of experiments performed to evaluate the performance of Projected Functional Regularization for continual self-supervised representation learning without the need of an exemplar memory and replay.

4.1. Datasets

We use the following datasets in our evaluation:

- **CIFAR-100**: Proposed by [35], this dataset consists 100 object classes in 45,000 images for training, 5,000 for validation, and 10,000 for test with 100 classes. All images are 32×32 pixels.
- **Tiny ImageNet**: A rescaled subset of 200 ImageNet [15] classes used in [59] and containing 64×64 pixel images. Each class has 500 training images, 50 validation images and 50 test images.
- **SVHN**: contains 32×32 pixel images of from house numbers. There are 10 classes with 73,257 training images and 26,032 test images. From we split 5% of the training images to use as a validation set.
- **Cars**: Was introduced in [34]. contains 16,185 images of 196 cars classes which includes 8,144 as train set and 8,041 as test set.
- **Aircraft**: Was proposed in [41] and consists 6,667 images for training and 3,333 for testing of 100 classes.

The last three datasets are used for evaluating our proposed method on downstream tasks. We downscale images to 64×64 for Cars and Aircraft in our experiments.

4.2. Training procedure and baseline methods

In all experiments, we train a ResNet-18 [27] using SGD with an initial learning rate of 0.06 and a weight decay of 0.0001. The network is trained with cosine annealing for the first 1500 epochs. After these epochs of cosine annealing, the learning rate is reduced by a factor of 0.8 for the both projectors (view and temporal) and 0.4 for the backbone. The data augmentation process is the same as in BarlowTwins (which was taken from SimCLR [10]). As a temporal projector, we use an MLP with a linear layer of 512 neurons followed by a batch normalization, Relu and a second linear layer of 256 neurons for CIFAR-100 and a linear layer with 512 neurons followed by a ReLu for TinyImageNet.

Downstream task classifiers are by default linear with a cross-entropy loss and are trained with Adam optimizer with a learning rate $5e$-3 for CIFAR-100 and $5e$-2 for TinyImageNet. We use validation data to implement a patience scheme that lowers the learning rate by a factor of 0.3 up to three times while training a downstream task classifier.

In our experiments we compare with the following baseline methods:

- **Fine-tuning (FT)**: The network is trained sequentially on each task without access to previous data and with no mitigation of catastrophic forgetting.
- **Single Task**: We perform joint training with fine-tuning on all data which provides an upper bound.
- **Continual Joint Training (CJ)**: We continually perform joint training on the entire dataset seen so far. This is provides a tighter upper bound than Single Task [3].
- **Feature Distillation (FD)**: Knowledge distillation is used as in LwF [37] to retain representation from previous tasks. We use the L2 distance as the regularization term, as is also proposed by other methods performing knowledge distillation on feature embeddings [19, 39, 67].
- **Elastic Weight Consolidation (EWC)**: We use the regularization method from [32] with a contrastive loss used to estimate the diagonal of the Fisher Information Matrix.

Note that we only compare to exemplar-free methods and exclude methods that require replay from our comparison. [1]

4.3. Continual representation learning

In this experiment we evaluate all methods in the incremental representation learning setting. The most straightforward way of doing this is to use the class incremental learning setting without access to labels. Specifically, we split datasets into four equal task as done in [53]. In each task we learn a self-supervised representation and in the evaluation phase we train a linear classifier using the trained backbone encoder. In order to assess the learned representation, we use all available test data to obtain the overall task-agnostic performance evaluation[2]

[1] code available at https://github.com/alviur/CVPR_PFR.git

[2] Note that we use *task agnostic* in this paper to refer to the class-incremental learning evaluation [44].

Table 1. (top) Accuracy on CIFAR-100 with 4 tasks of 25 classes for incremental, self-supervised training. The learned representation is evaluated using a linear classifier over all classes after each task. (bottom) Evaluation of the same trained models on a different downstream task - classification using SVHN dataset. Mean and standard deviation over five runs are provided.

CIFAR100				
Method	**Task 1**	**Task 2**	**Task 3**	**Task 4**
Single	-	-	-	65.4±1.4
CJ	53.3±0.7	58.4±0.8	60.8±1.1	63.6±1.5
FD	53.3±0.4	55.6±0.5	56.8±1.0	57.8±0.5
EWC	53.0±0.3	53.1±0.3	53.8±0.6	55.0±0.4
PFR	53.2±0.4	56.4±0.4	58.2±0.3	**59.7**±0.3
FT	53.4±0.5	53.0±0.7	54.6±0.2	54.8±1.0
SVHN				
Method	**Task 1**	**Task 2**	**Task 3**	**Task 4**
Single	-	-	-	64.0±1.4
CJ	60.6±2.6	63.2±1.7	63.4±1.3	63.0±1.3
FD	60.4±2.7	62.7±2.2	64.3±1.7	65.8±1.5
EWC	61.3±2.3	62.4±1.8	64.1±1.2	64.5±1.5
PFR	60.4±2.7	63.5±2.3	66.2±1.9	**68.0**±1.5
FT	60.4±2.7	61.0±1.4	63.8±1.1	64.5±1.1

In Table 1 we present the results for all methods. After the final task, the upper bound CJ obtains 60.6%, while a simple fine-tuning (FT) method reaches 56.8%. This is the gap where methods using regularization can improve. Joint training on all data at once outperforms CJ by 2.4%. PFR obtains an accuracy after the final task of 60.1%, while other regularization methods FD and EWC reach 57.2% and 55.8%, respectively.

In addition, we show the task-aware results on CIFAR 100 of the incrementally learned representations in Figure 2. Here the models are the same as those used in Table 1. Note that all other results in the paper are task-agnostic (no task-ID given during inference). Here, we also use training data from future tasks to train the classifier: this allows us to also evaluate the performance of the tasks that have not been seen by the feature extractor (see above diagonal elements). We observe that all methods, including FT, incrementally improve results. Only our proposed PFR method considerably outperforms FT in this setting. It is also interesting to observe that PFR obtains positive backward transfer, since the performance on task 1 and 2 improves during the consecutive training sessions.

Learned representations. In addition to evaluating accuracy on downstream classification tasks, we compare learned representation similarity with a Centered Kernel Alignment (CKA) [33]. The results are given in Figure 3. When the task is learned and immediately evaluated, the similarity is equal to one. When we finetune the model with new data, we start experiencing representation degradation – seen in decreasing values in the columns in Figure 3, left. With PFR, representation forgetting progresses much more slowly. FT is the worst, having the first task similarity after the last one with CKA equal 0.69, next is FD with value 0.72, and the best is PFR with 0.82.

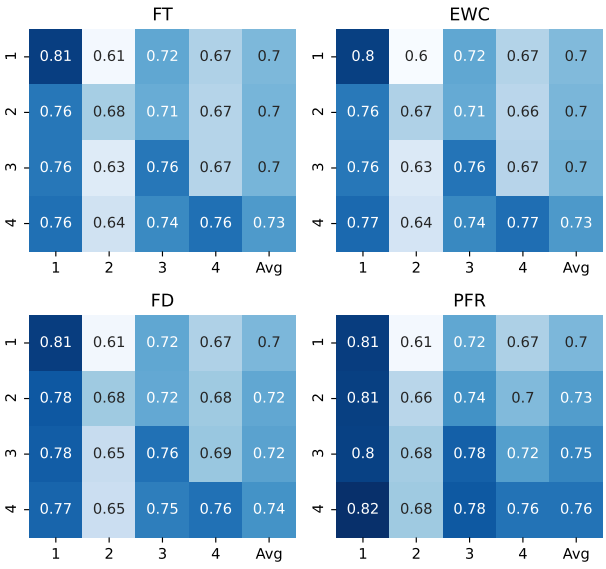


Figure 2. Task-aware performance after the four consecutive tasks on CIFAR-100 for several methods. Each row reports the results after each task, and columns represent on which task the model is evaluated. The last column reports average accuracy after each trained task.

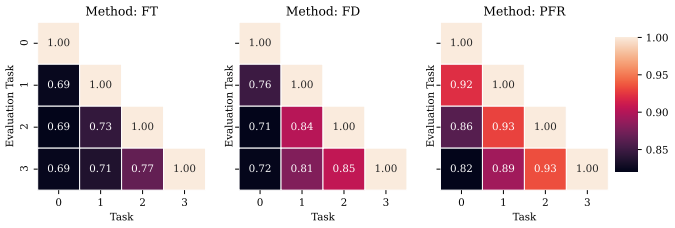


Figure 3. Representation similarity compared with CKA for FT, FD, and PFR during incremental training.

Many task scenario. Here we consider a challenging setting with longer sequences – i.e. with more tasks. We experimented with our PFR method, fine-tuning(FT) and feature distillation(FD) on CIFAR-100 split into 50 or 100 tasks. In the case of 100 tasks, we only have a single class per task,

which is an interesting setting since there are no negative classes, forcing the network to learn representations that are discriminative at the instance level. Results are presented in Figure 4a, where accuracy over all classes is given per training session for each method. Without any mitigation of forgetting, FT cannot maintain the learned representations in longer tasks sequences, dropping closely to the level of a randomly initialized backbone (23%) in the extreme case of 100 tasks. FD is also struggling on the longer sequence of 100 tasks. Only, our method shows stable results, preventing forgetting of the learned representation and progressing steadily. Similar results can be observed for 50 tasks, however the differences are not as pronounced as for the 100 task sequence.

4.4. The influence of regularization

Each regularization method is applied differently to the self-supervised network. To assess the influence of regularization and quantify its effect, we use the *forgetting* and *intransigence* measures defined in [8]. Forgetting measures the average drop in accuracy per-task during continual learning. Intransigence describes the inability of a model to learn a new task. Formally, it is the difference between a referential model accuracy at task t – for us jointly trained self-supervised model up to task t – and the current task accuracy measured using held-out data.

In Figure 4b all methods with different λ parameter values are shown for CIFAR-100 dataset. FT is a point of reference here since it uses no regularization and maximum forgetting is expected. EWC is a weight regularization method that regularizes network weight changes using the Fisher Information Matrix. With larger lambda, forgetting is lower, but at the same time we pay the price of larger intransigence. As in the other experiments, we found that weight regularization does not obtain satisfactory results when applied to continual self-supervised learning. Feature distillation represents a better trade-off. The closer the results are to the bottom left corner, the better they are, and here PFR is the clear winner. The PFR results are consistently better than FD by some margin, which implies that the additional flexibility of the model introduced by the temporal projection network indeed leads to higher plasticity while at the same time keeping forgetting low. These results are confirmed on the larger Tiny ImageNet dataset (see Figure 4c). Here, the gap between FD and PFR is much larger, showing that projected functional regularization suffers from much lower forgetting at equal intransigence values.

4.5. Generality of the Approach

In order to verify that PFR generalizes to other self-supervised approaches, we conducted a series of experiments with SimCLR [10], SimSiam [12], and BarlowTwins [68]. In Table 2 we present results of fine-tuning and PFR for a ten task scenario. PFR results in 5.4%, 8.4%, and 8.1% improvement over FT after the final task. For this longer task sequence scenario the gain of our method with respect to FT is much larger when compared with results in Table 1. Starting from the second task, the effect of projected regularization is clear. In the ten task scenario, SimSiam after the first five tasks begins to outperform BarlowTwins, which is reflected by the results in the Task 10 column.

Table 2. Accuracy of SimCLR, SimSiam and BarlowTwins on CIFAR-100 split into 10 tasks of 10 classes.

Method	Task 1	Task 2	Task 5	Task 10
SimCLR PFR	40.4	44.7	47.2	48.2
SimSiam PFR	43.1	50.2	53.1	55.1
BarlowT PFR	45.1	50.6	54.7	**55.4**
SimCLR FT	40.4	41.2	40.6	42.8
SimSiam FT	43.1	46.0	47.0	46.7
BarlowT FT	45.1	48.8	48.9	**47.0**

4.6. Transfer to downstream tasks

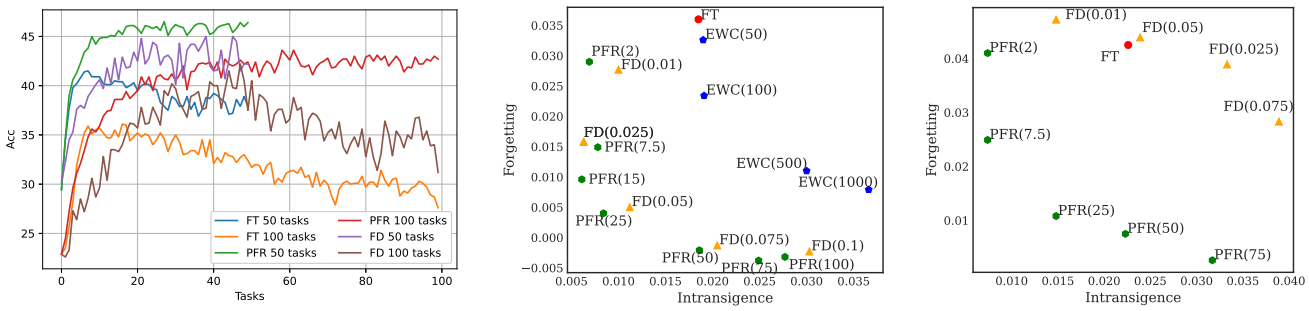
To better asses the quality of the trained representation, we evaluated all methods on a series of different downstream datasets. This allows us to evaluate the transferability of the learned features during the continual training process. The results for the smaller sized (32x32 images) CIFAR-100 and SVHN datasets are in Table 1 (bottom table). We observe a similar outcome as in the source dataset evaluation. The best results use our PFR method, followed by FD, EWC, and finally FT. The results are consistent during incremental learning: the better the representation is on a source classification task, the better it is on the target (SVHN) dataset after each task.

Furthermore, we trained all the methods on a larger dataset (Tiny ImageNet). We use the same procedure as for CIFAR-100 with four tasks, but with bigger images (64×64) and more classes (200). The results are presented in Table 3. In this dataset our method (PFR) surpasses FD, which is followed by the modified EWC and FT.

As in CIFAR100 we evaluate our networks trained on Tiny Imagenet on different downstream datasets (Cars and Aircraft). The results are given in Table 4. For these datasets, as in CIFAR100, the accuracy of the various techniques follows the same pattern: PFR yields the best results, followed by FD, EWC, and finally FT.

5. Conclusions and Future directions

In this paper, we proposed a method for incremental self-supervised learning without the need for any stored examples of previous tasks. Most existing regularization



(a) (b) (c)

Figure 4. (a): Performance of several methods for different numbers of tasks on CIFAR-100. (b, c): Influence of regularization on forgetting and intransigence for EWC, FD, and PFR. FT uses no regularization and represents a point of reference. The regularization hyperparameter λ is given in parentheses, (b) presents results for CIFAR-100 dataset, (c) for Tiny ImageNet.

Table 3. Accuracy on TinyImageNet split into four tasks. The learned representation is evaluated using a linear classifier over all classes after each task.

Method	**Task 1**	**Task 2**	**Task 3**	**Task 4**
Joint (no CL)	-	-	-	46.0
FD	35.3	35.9	36.6	36.6
EWC	35.3	37.4	36.9	38.2
PFR (Ours)	35.3	38.6	39.9	**42.3**
FT	35.3	38.3	38.5	39.1

Table 4. Transfer Learning to downstream tasks.

Cars				
Method	**Task 1**	**Task 2**	**Task 3**	**Task 4**
Joint (no CL)	-	-	-	34.5
FD	27.1	30.6	31.8	31.1
EWC	27.1	28.3	27.7	27.8
PFR	27.1	31.1	33.1	**33.8**
FT	27.1	29.5	29.2	31.5
Aircraft				
Method	**Task 1**	**Task 2**	**Task 3**	**Task 4**
Joint (no CL)	-	-	-	30.0
FD	24.6	25.6	27.0	27.0
EWC	24.6	23.8	25.3	25.1
PFR	24.6	26.2	27.0	**28.4**
FT	24.6	25.6	26.9	27.0

methods for continual learning are applied to class predictions or logits. Such approaches applied to self-supervised representation learning result in low plasticity. To address this, we propose Projected Functional Regularization via a temporal projection network that ensures that the newly learned feature space preserves information of the previous one, while still allowing for the learning of new features, resulting in higher plasticity. Extensive results on CIFAR100 and Tiny ImageNet demonstrate that our approach outperforms standard feature distillation by a considerable margin.

Finally, there are several limitations and future directions we discuss here. First, due to the high computational demands, experiments have been performed on low-resolution images, and they need to be confirmed for higher resolution data. Next, our method assumes access to task boundaries and cannot be directly applied in the task-free setting (without task boundaries) [2]. We think that this can be addressed by replacing the regularization based on the model from the previous task, with a regularization model that is updated with momentum. Next, continual learning of transformer architectures has only recently started [20, 50]. Self-supervised learning is a key ingredient of the transformer network training, and integrating our theory with these attention-based architectures for their continual learning is especially interesting. Finally, extending the theory with a limited replay buffer is of interest and would allow to directly report class-incremental learning results where the buffer is used to compute the classifier layer.

Acknowledgements

We acknowledge the support from Huawei Kirin Solution, and the Spanish Gouvernement funded project PID2019-104174GB-I00/ AEI / 10.13039/501100011033.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision*, 2018. 1, 3, 4

[2] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Conference on Computer Vision and Pattern Recognition*, 2019. 8

[3] Jordan Ash and Ryan P Adams. On warm-starting neural network training. *Advances in Neural Information Processing Systems*, 33:3884–3894, 2020. 5

[4] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8218–8227, 2021. 2

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020. 1, 2

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *CoRR*, abs/2104.14294, 2021. 1

[7] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *European Conference on Computer Vision*, 2018. 1

[8] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: understanding forgetting and intransigence. In *European Conference on Computer Vision*, 2018. 3, 7

[9] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019. 2

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 2, 5, 7

[11] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2

[12] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 1, 2, 7

[13] Ali Cheraghian, Shafin Rahman, Pengfei Fang, Soumava Kumar Roy, Lars Petersson, and Mehrtash Harandi. Semantic-aware knowledge distillation for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2534–2543, 2021. 3

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

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, 2009. 5

[16] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Conference on Computer Vision and Pattern Recognition*, 2019. 3

[17] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 1, 2

[18] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. *Advances in neural information processing systems*, 27:766–774, 2014. 2

[19] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 86–102. Springer, 2020. 3, 4, 5

[20] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. *CoRR*, abs/2111.11326, 2021. 8

[21] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv*, 2017. 1, 2

[22] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 1, 2

[23] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. 1, 2

[24] Linting Guan and Yan Wu. Reduce the difficulty of incremental learning with self-supervised learning. *IEEE Access*, 2021. 3

[25] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006. 2

[26] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 2

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016. 5

[28] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning Workshop*, 2014. 4

[29] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *European Conference on Computer Vision*, 2018. 4

[30] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *International Conference on Computer Vision*, 2019. 1

[31] Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3957–3966, 2021. 3

[32] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *National Academy of Sciences*, 2017. 1, 3, 4, 5

[33] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019. 6

[34] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013. 5

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

[36] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, 2019. 1

[37] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 1, 3, 5

[38] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *International Conference on Pattern Recognition*, 2018. 3

[39] Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D Bagdanov, Shangling Jui, and Joost van de Weijer. Generative feature replay for class-incremental learning. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2020. 4, 5

[40] Divyam Madaan, Jaehong Yoon, Yuanchun Li, Yunxin Liu, and Sung Ju Hwang. Rethinking the representational continuity: Towards unsupervised continual learning. *arXiv preprint arXiv:2110.06976*, 2021. 3

[41] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv*, 2013. 5

[42] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *European Conference on Computer Vision*, 2018. 2

[43] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2

[44] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *arXiv preprint arXiv:2010.15277*, 2020. 2, 5

[45] Marc Masana, Tinne Tuytelaars, and Joost Van de Weijer. Ternary feature masks: zero-forgetting for task-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3570–3579, 2021. 2

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

[47] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016. 1, 2

[48] Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard E Turner, and Mohammad Emtiyaz Khan. Continual deep learning by functional regularisation of memorable past. In *Proc. Adv. Neural Inf. Process. Syst.*, 2020. 4

[49] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019. 1

[50] Francesco Pelosin, Saurav Jha, Andrea Torsello, Bogdan Raducanu, and Joost van de Weijer. Towards exemplar-free continual learning in vision transformers: an account of attention, functional and weight regularization. *arXiv preprint arXiv:2203.13167*, 2022. 8

[51] Benedikt Pfülb and Alexander Gepperth. A comprehensive, application-oriented study of catastrophic forgetting in dnns. In *International Conference on Learning Representations*, 2019. 1

[52] Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In *Advances in Neural Information Processing Systems*, 2019. 3

[53] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Conference on Computer Vision and Pattern Recognition*, 2017. 1, 5

[54] Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 2

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

[56] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, 2018. 2

[57] Yujun Shi, Li Yuan, Yunpeng Chen, and Jiashi Feng. Continual learning via bit-level information preserving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16674–16683, 2021. 3

[58] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, 2017. 1

[59] Stanford. Tiny imagenet challenge, cs231n course., CS231N. 5

[60] Shixiang Tang, Dapeng Chen, Jinguo Zhu, Shijie Yu, and Wanli Ouyang. Layerwise optimization by gradient decomposition for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9634–9643, 2021. 3

[61] Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with gaussian processes. In *Proc. Int. Conf. Learn. Repres.*, 2020. 4

[62] Liyuan Wang, Kuo Yang, Chongxuan Li, Lanqing Hong, Zhenguo Li, and Jun Zhu. Ordisco: Effective and efficient usage of incremental unlabeled data for semi-supervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5383–5392, 2021. 2

[63] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2006. 2

[64] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. Memory replay GANs: learning to generate images from new categories without forgetting. In *Advances in Neural Information Processing Systems*, 2018. 1

[65] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Conference on Computer Vision and Pattern Recognition*, 2019. 1, 3

[66] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2021. 2

[67] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6982–6991, 2020. 4, 5

[68] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021. 1, 2, 3, 7

[69] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, 2017. 1, 3, 4

[70] Mengyao Zhai, Lei Chen, and Greg Mori. Hyperlifelonggan: Scalable lifelong learning for image conditioned generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2246–2255, 2021. 2

[71] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 2

[72] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5871–5880, 2021. 3