Continually Learning Self-Supervised Representations with Projected Functional Regularization

Alex Gomez-Villa,* Bartlomiej Twardowski,* Lu Yu,* Andrew D. Bagdanov,† Joost van de Weijer*

{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 methods. However, these methods are unable to acquire new knowledge incrementally—they are, in fact, mostly used only as a pre-training phase with IID data. In this work we investigate self-supervised methods in continual learning regimes without additional memory or replay. To prevent forgetting of previous knowledge, we propose the usage of functional regularization. We will show that naive functional regularization, also known as feature distillation, leads to low plasticity and therefore seriously limits continual learning performance. To address this problem, we propose Projected Functional Regularization where a separate projection network ensures that the newly learned feature space preserves information of the previous feature space, while allowing for the learning of new features. This allows us to prevent forgetting while maintaining the plasticity of the learner. Evaluation against 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 [3,8,10,20], and that when learned representations are transferred to downstream tasks are sometimes even superior [4]. These methods learn representations that are invariant with respect to a set of data augmentations. They are typically trained with contrastive losses where 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 have shown 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 [10, 20]. These methods apply various mechanisms to prevent the trivial solution, 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 to 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 [41]. A variety of approaches have been proposed including various types of regularization [1, 29, 34, 65], data replay [5, 27, 47, 61], pseudo replay [52, 60], and growing architectures [49]. Even though there is some work on unsupervised continual learning [18, 33, 39, 49], the vast majority of existing work is on supervised continual learning [44,45].

Earlier works on self-supervised learning was based on pretext tasks like predicting rotation [19], determining patch position [15], or solving jigsaw puzzles in images [42]. Labels for these discriminative pretext tasks can be automatically computed, and allow to learn 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 [3, 8, 20, 64]. 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 [10]. The results are impressive and are competitive with many supervised methods on downstream tasks [4].

In this paper, we propose an approach to continual self-supervised learning that is able to learn high-quality vi-
ual 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 previous 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 of an external memory of samples of 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 projection network that provides more freedom to the learner 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 rest of the paper is organized as follows: First, related work is mentioned in section 2. In section 3 self-supervised learning and our projected functional regularization are described. Section 4 describes performed experiments and the results analysis. Finally, in section 5 conclusions and future work are presented.

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 [19], solving jigsaw puzzles [42], determining relative patch positions [15], predicting surrogate classes [16], and image colorization [67]).

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 [22] and some extensions using triplet losses [59]. However, in the unsupervised setting without labels, different approaches must be used for creating such pairs. In SimCLR [8], 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 [9, 23] 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 [3]. 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 [20] 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 [10] 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 [64]. 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 approaches can be broadly divided into replay-based, architecture-based, and regularization-based methods [12, 40]. Replay-based methods save a small amount of data from previously seen tasks [2, 7] or generate synthetic data with a generative model [58, 66]. 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 [62], Piggyback [38], PackNet [39], DAN [48], HAT [50], and PathNet [18]. Regularization-based methods add an additional regularization term derived from knowledge of previous tasks to the
training loss. This can be done by either regularizing the weight space (constraining important parameters) [51, 54] or the functional space (constraining predictions or intermediate features) [11, 17, 28]. EWC [29], MAS [1], REWC [35], SI [65], and RWalk [6] constrain the importance of network parameters to prevent forgetting. Methods such as LwF [34], LwM [14] and BiC [61] 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 [17], we therefore propose an additional 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 [46] with an approach based on variational autoencoders and a Gaussian mixture model. The encoding part and replay for knowledge retention play together. Still, detection of new clusters and model expansion is necessary.

Our contribution is fundamentally different than methods using self-supervised learning to improve the learning of a sequence of supervised tasks [21, 68]. 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.

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 in some target domain, we have training data \( \mathcal{D}^t = \{x_i^t, y_i^t\} \) on which we learn a classifier \( g_\theta : \mathcal{F} \rightarrow \mathcal{Y} \) (with \( \mathcal{Y} \) being the output space) that minimizes a loss \( \mathcal{L} = \ell(y^t, y^t = g_\theta(f_\theta(x^t))) \). Adaptation to the target domain might only optimize the weights \( \phi \) while keeping \( \theta \) fixed on the target data, or instead might also allow \( \theta \) to be fine-tuned on the target data.

Here we consider the SimSiam [10] approach to self-supervised learning of the representation network \( f_\theta \). SimSiam does not require explicit negative samples and achieves competitive performance while remaining computationally efficient, primarily by allowing smaller mini-batches and obviating the need for a momentum encoder. The SimSiam architecture has two branches (see the gray area in Fig. 1). In SimSiam an asymmetric network architecture is used that contains a projector network \( z : \mathcal{F} \rightarrow \mathcal{Z} \) in both branches, and an additional predictor network \( p : \mathcal{Z} \rightarrow \mathcal{Z} \) in one. For the sake of notational simplicity, we do not make explicit the parameters of the networks \( z \) and \( p \) since they are not used by downstream tasks. In the branch without predictor the gradient is not backpropagated during training, which was found to be crucial in preventing collapse to trivial solutions [55]. The parameters in the backbone and projector layer are shared between the branches. The network is trained by minimizing the distance (or maximizing the similarity) of two different augmented views \( x_1 \) and \( x_2 \) of the same data sample \( x \). We use the notation \( \mathcal{D}^* \) to identify the set of augmented samples, and \( b_\theta(x) = z(f_\theta(x)) \). The training loss is defined as:

\[
\mathcal{L}_c = \mathbb{E}_{x_1, x_2 \sim \mathcal{D}^*} [S(p(b_\theta(x_1)), b_\theta(x_2))/2 + S(p(b_\theta(x_2)), b_\theta(x_1))/2],
\]

where

\[
S(a, b) = -\frac{a^T b}{||a|| ||b||}.
\]

Note that there is no contrastive term in Eq. 1. The loss
lacks explicit negative pairs and only similarity is enforced during training.

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 \( f_\theta \). However, for many real-world scenarios this is an unrealistic setup and the learner will have to learn the optimal feature extractor \( f_\theta \) from a stream of data drawn from a distribution that varies over time.

In the considered setup, 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 \( D_t \). In this case we would like to find the parameters \( \theta \) of the feature extractor \( f_\theta \) that minimize the summed loss over all tasks up to the current one \( c \):

\[
\arg \min_{\theta} \sum_{t=1}^{c} \mathcal{L}^t_c,
\]

where

\[
\mathcal{L}^t_c = \mathbb{E}_{x_1,x_2 \sim D^*_t} \left[ S(p(b_\theta(x_1)), b_\theta(x_2)) / 2 + S(p(b_\theta(x_2)), b_\theta(x_1)) / 2 \right] \]

Again, \( D^*_t \) refers to the set of augmented samples from \( D_t \) (i.e., the data from task \( i \)). However, during the continual training we only have access to the data of one task at a time, meaning that the optimal parameters must be found while only having access to the current data \( D_c \). 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 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 [1,29,65], 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 [26,43,56].

The canonical example of functional regularization, called Learning without Forgetting (LwF), was introduced in [26] and is based on knowledge distillation [25]. 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 [17,36,63]. We will refer to this as feature distillation (FD) and it leads to the following loss when training task \( t \):

\[
\mathcal{L}^t_c + \lambda_{fd} \mathbb{E}_{x_1,x_2 \sim D^*_t} \left[ \| b_\theta(x_1) - b_{\theta_{t-1}}(x_1) \| + \| b_\theta(x_2) - b_{\theta_{t-1}}(x_2) \| \right],
\]

where \( \theta^{t-1} \) refers to the parameters learned after training up to task \( t - 1 \), and \( \lambda_{fd} \) defines the importance of the regularization term.

The regularization imposed on class predictions in the original LwF paper [26] 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 [17]. 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 \( \| b_\theta(x) - b_{\theta_{t-1}}(x) \| \). 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 model projection network \( m : \mathcal{Z} \rightarrow \mathcal{Z} \) 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:

\[
\mathcal{L}^t_c + \lambda_{pfr} \mathbb{E}_{x_1,x_2 \sim D^*_t} \left[ S(m(b_\theta(x_1)), b_{\theta_{t-1}}(x_1)) + S(m(b_\theta(x_2)), b_{\theta_{t-1}}(x_2)) \right]
\]

(6)

New features learned in \( b_\theta(x) \) 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

In this section we report on a variety of experiments performed to evaluate the performance of Projected Functional Regularization for continual self-supervised representation learning.

4.1. Datasets

We use the following datasets for our experimental evaluation:
• **CIFAR-100**: proposed by [32], 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.

• **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.

• **Tiny ImageNet**: a rescaled subset of 200 ImageNet [13] classes used in [53] and containing 64×64 pixel images. Each class has 500 training images, 50 validation images and 50 test images.

• **Cars**: was introduced in [31]. It 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 [37] 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. Images are re-sized to 64×64 in our experiments.

### 4.2. Training procedure and baseline methods

In all experiments we train a ResNet-18 architecture [24] 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, like in SimSiam [10]. After these epochs of cosine annealing, the learning rate is reduced by a factor of 0.6. The data augmentation process is also taken from SimSiam and follows SimCLR [8]. For small datasets like CIFAR-100, the LARS optimizer is not used.

Downstream task classifier is by default linear one, trained with Adam optimizer with a learning rate 1e-4. We use validation data for patience schema – lowering 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, one after another, without access to previous data and with no mitigation of catastrophic forgetting.

• **Feature Distillation (FD)**: knowledge distillation is used as in LwF [34] to retain representation from a previous task. We use L2 distance as the regularization term.

• **Continual Joint Training (CJ)**: contrary to a single training session over the entire dataset, a joint training is conducted on the entire dataset seen so far.

• **Elastic Weight Consolidation (EWC)**: the regularization method from work of [29] where contrastive loss is used to estimate the diagonal of the Fisher Information Matrix.

### 4.3. Continual representation learning

In this experiment we evaluate all methods in the representation incremental learning setting. The most straightforward way of doing this is to use class incremental learning setting, without access to labels. Specifically, we split dataset into ten equal tasks, following [47]. In each task we learn a self-supervised representation. 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 get the overall task-agnostic performance evaluation.

In Table 1 the results for all methods at several selected tasks are presented. The first task accuracy is for all methods the same and equal to 48.9% when evaluated on all classes, we omit that in the table. After the final task, the upper bound CJ obtains 65.7%, while a simple fine-tuning (FT) method obtains 46.7%. This presents the gap, where methods with a regularization can be placed. Joint train on all data at once gets a higher results than CJ by 0.11%. Our method PFR received an accuracy after the final task of 55.1%, while other regularization methods FD and EWC have respectively 50.1% and 55.5%. The Figure 2 shows a detailed evolution of the accuracy during the continual learning process. Our PFR method obtains superior performance than fine-tuning (FT) and Feature Distillation (FD) along with 10 tasks training.

All regularization methods help with getting better results in incremental tasks. In both, second and fifth tasks, presented in the table application of EWC receives the best results, while PFR is a second best. In Figure 3 presents differences between accuracies of each task for PFR and FT methods. It is expected that by maintaining the knowledge of the old-representation (stability) a price is paid in terms of plasticity. That is seen in the red diagonal, while old tasks are having surplus in a accuracy – green area with bigger than zero values below the red diagonal.

**Learned representation** In addition to checking accuracy on a downstream classification task, we compared learned representation similarity with a Centered Kernel Alignment (CKA) proposed by [30]. The results for FT and PFR are presented in Figure 4. 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 - going in the column down in Figure 4 left. With PFR, representation forgetting progresses much slower. The worst similarity for PFR and the longest evaluated first task is the same as the best when we go from second task to the third one in FT and is equal to
Table 1. Accuracy for CIFAR-100, 10 tasks with 10 classes training. Representation is evaluated with a linear classifier over all classes.

CIFAR100

| Method | Task 2 | Task 5 | Task 10 |
|--------|--------|--------|---------|
| Single task | - | - | 0.668 |
| CJ | 0.512 | 0.589 | 0.657 |
| FD | 0.488 | 0.492 | 0.501 |
| EWC | 0.492 | 0.534 | 0.555 |
| PFR | 0.502 | 0.531 | **0.551** |
| FT | 0.460 | 0.470 | 0.467 |

SVHN

| Method | Task 2 | Task 5 | Task 10 |
|--------|--------|--------|---------|
| Single task | - | - | 0.633 |
| CJ | 0.651 | 0.648 | 0.653 |
| FD | 0.700 | 0.709 | 0.733 |
| EWC | 0.705 | 0.736 | **0.755** |
| PFR | 0.735 | 0.744 | 0.740 |
| FT | 0.640 | 0.647 | 0.631 |

Moreover, in Figure 5 we present t-Distributed Stochastic Neighbor Embedding (t-SNE) [57] plots of the first task evaluated after tasks: 2, 6, and 10 for three of the methods presented in Table 1. FT (first column) presents mixed classes without any clear structure for all the evaluated tasks. FD (second column) shows clusters of classes in all the evaluated task with a greater number of clusters after the last task. As in FD, the PFR method (third column) present clusters of classes in all the evaluated task. However, PFR has more well-formed clusters than FD.

![Figure 2](image2.png)

Figure 2. Different methods accuracy of a linear classifier during the class incremental learning session with CIFAR-100 dataset split in ten tasks.

![Figure 3](image3.png)

Figure 3. A difference matrix of accuracies of evaluating methods against fine-tuning (left PFR: Acc(PFR) - Acc(FT), right FD: Acc(FD) - Acc(FT)) on different tasks during a train session with CIFAR-100. The last column present difference of average accuracy difference after each trained task.

![Figure 4](image4.png)

Figure 4. Representation similarities comparison with CKA method for FT and PFR during incremental training.

**More tasks scenario.** Here we consider the challenging setting with longer sequences, i.e. with more tasks. We experimented with our PFR method with CIFAR-100 splitting for more than ten tasks - 50, 100. In 100 tasks, we only have a single class per task, what is a hard setting for several class-IL methods. Results are presented in Figure 6, where accuracy over all classes is presented during a training session for each method. As expected, the best CJ constantly improves representation. Interestingly, without additional measurement for alleviate forgetting FT cannot even maintain learned representation in longer tasks sequences, dropping even to the random level in an extreme case of 100 tasks. Our PFR method presents stable results, preventing forgetting of the learned representation and progressing steadily.

**4.4. Generality of the Approach**

In order to verify if our approach generalize to other self-supervised approaches, we conduct a series of experiments with SimCLR [8] and Barlow Twins [64] methods. In Table 2 results for fine-tuning and PFR are presented. The proposed method results in 5.4% and 4.5% improvement over FT after final task. Starting from second task, the effect of
Figure 5. CIFAR-100 t-SNE plots for Task 1 test datapoints during the incremental training of representations for methods FT, FD, and PFR.

Figure 6. Different number of tasks for CIFAR-100 and evaluated methods.

Table 2. Accuracy for CIFAR-100, 10 tasks with 10 classes training. SimCLR and Barlow Twins

| Method       | Task 2 | Task 5 | Task 10 |
|--------------|--------|--------|---------|
| SimCLR PFR   | 44.7   | 47.2   | 48.2    |
| BarlowT PFR  | 31.5   | 35.9   | 35.6    |
| SimCLR FT    | 41.2   | 40.6   | 42.8    |
| BarlowT FT   | 30.6   | 33.9   | 31.1    |

4.5. Transfer Learning to downstream tasks

To better assess the quality of the trained representation, we evaluated all the methods with 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 im-
Table 3. Accuracy for TinyImageNet, 10 tasks with 20 classes training. Representation is evaluated with linear classifier over all classes.

| Method          | Task 2 | Task 5 | Task 10 |
|-----------------|--------|--------|---------|
| Joint (no CL)   | -      | -      | 37.4    |
| FD              | 33.3   | 34.5   | 34.9    |
| EWC             | 30.2   | 31.5   | 31.7    |
| PFR (Ours)      | 33.7   | 34.9   | **35.1**|
| FT              | 33.1   | 32.9   | 33.1    |

Table 4. Transfer Learning to downstream tasks

| Method          | Task 2 | Task 5 | Task 10 |
|-----------------|--------|--------|---------|
| Single task     | -      | -      | 28.5    |
| FD              | 34.3   | 36.0   | 28.0    |
| EWC             | 31.0   | 32.3   | 32.7    |
| PFR             | 32.9   | 35.2   | 36.5    |
| FT              | 28.1   | 31.6   | 27.7    |

| Method          | Task 2 | Task 5 | Task 10 |
|-----------------|--------|--------|---------|
| Single task     | -      | -      | 24.6    |
| FD              | 24.6   | 26.5   | 23.3    |
| EWC             | 23.8   | 24.0   | 25.1    |
| PFR             | 23.9   | 25.1   | 26.4    |
| FT              | 22.2   | 23.4   | 22.6    |

5. Conclusions

In this paper we propose a method for incremental self-supervised learning without the need for any stored examples of previous tasks. Most existing regularization 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 projection network that ensures that newly learned feature space preserves information of the previous feature space, while allowing for the learning of new features. Consequently, our method has significantly higher plasticity. Extensive results on CIFAR100 and Tiny ImageNet demonstrate that our approach outperforms standard feature distillation by a large margin.

Acknowledgements

We acknowledge the support from Huawei Kirin Solution.

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] 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
[3] 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
[4] 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
[5] 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
[6] 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] 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
[8] 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, 6

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

[10] 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, 3, 5

[11] Ali Cheraghian, Shafin Rahman, Pengfei Fang, Soumava Kumar Roy, Lars Petersson, and Mehrtaash 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

[12] Matthias Delange, Rafah 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

[13] 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

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

[15] 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

[16] 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

[17] 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

[18] 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

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

[20] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tal Lee, 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

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

[22] 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

[23] 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

[24] 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

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

[26] 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

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

[28] Xinting Hu, Kaihu Tang, Chunyuan 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

[29] 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

[30] 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. 5

[31] 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

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

[33] 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

[34] Zhizhong Li and Derek Hoiem. Learning without forgetting. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017. 1, 3, 5
[35] 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

[36] 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

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

[38] 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

[39] 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

[40] 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

[41] 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

[42] 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

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

[44] 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

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

[46] 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

[47] Sylvester-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

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

[49] 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

[50] 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

[51] 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

[52] 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

[53] Stanford. Tiny imagenet challenge, cs231n course., CS231N.

[54] 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

[55] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. *arXiv preprint arXiv:2102.06810*, 2021. 3

[56] 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

[57] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 6

[58] Liyuan Wang, Kuo Yang, Chongxuan Li, Lanqing Hong, Zhenguo Li, and Jun Zhu. Oridiscard: 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

[59] 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

[60] 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

[61] 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

[62] 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

[63] 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

[64] 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, 6

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

[66] 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

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

[68] 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