Abstract

This paper presents SPeCiaL: a method for unsupervised pretraining of representations tailored for continual learning. Our approach devises a meta-learning objective that differentiates through a sequential learning process. Specifically, we train a linear model over the representations to match different augmented views of the same image together, each view presented sequentially. The linear model is then evaluated on both its ability to classify images it just saw, and also on images from previous iterations. This gives rise to representations that favor quick knowledge retention with minimal forgetting. We evaluate SPeCiaL in the Continual Few-Shot Learning setting, and show that it can match or outperform other supervised pretraining approaches.

1 Introduction

In Machine Learning, practitioners often rely on the ubiquitous assumption that the training data is sampled independently and identically from a fixed distribution. This iid assumption is a convenient requirement, spanning from classical approaches to Deep Learning based methods. Yet, the world that humans encounter is far from static: the data we observe is temporally correlated, and changes constantly. Therefore many practical applications, from autonomous driving to conversational chatbots, can greatly benefit from the ability to learn from non-stationary data.

To this end, the field of Continual Learning [French, 1991; Kirkpatrick et al., 2017] tackles the problem setting where an agent faces an online and non-stationary data stream. The core of CL methods have been intently focused on directly minimizing catastrophic interference, a phenomenon whereby previously acquired knowledge gets overwritten as the agent learns new skills.

In this paper, we are interested in the deployment of CL systems [Caccia et al., 2020b]. Namely, we are interested in preparing a system before its deployment on non-stationary data where it will need to make predictions and accumulate knowledge in an online fashion. We believe this setting captures better the deployment of CL systems in real life, as it would be more realistic to deploy an agent with some - albeit limited - knowledge of the world.

Therefore, we ask ourselves how can we equip agents with data representations that are amenable to learning under non-stationarity? An innovative solution was first proposed by [Javed and White, 2019], coined Online-Aware Meta Learning (OML). This objective differentiates through a supervised sequential learning task and optimizes for minimal forgetting. While ingenious, OML requires access to labelled data, which can be difficult or expensive to obtain for many CL scenarios. For example, is it fairly cheap to obtain large unlabelled datasets from a dashboard camera in the hopes of training a self-driving agent. However, manually annotating each frame - or even deciding which types of annotation - can be time consuming and costly.

In this work, we instead aim to learn representations amenable to CL purely from unlabelled data. Recently, the field of unsupervised representation learning (or self-supervised learning) has shown tremendous progress [Chen et al., 2020a], closing the gap with fully supervised methods on large-scale image datasets. Moreover, SSL methods have shown better transfer capabilities [Chen et al., 2020a] to out-of-distribution data than supervised methods. To this end, we leverage key components of recent SSL methods, enabling us to devise an efficient label-free representation learning algorithm for CL. Concretely, our contributions are as follows:

- We propose a novel strategy, called SPeCiaL, for unsupervised pretraining of representations amenable to continual learning.
- Our algorithm efficiently reuses past computations, reducing its computational and memory footprint.
- We show that SPeCiaL can match or outperform OML on several continual few-shot learning benchmarks.

2 Related Work

Our work builds upon several research fields, which we summarize here.

Self-Supervised Learning The field of Unsupervised Learning of Visual Representations has made significant
progress over the last few years, reducing the gap with supervised pretraining on many downstream tasks. By using data augmentations, SSL methods can generate pseudo-labels, matching different augmented views of the same image together. Instance-level classifiers [Chen et al., 2020a; Chen et al., 2020b] treat each image - and its augmented versions - as its own class, and rely on noise contrastive estimation to compare latent representations. Clustering Methods [Caron et al., 2020] instead group latent instances together, either using the cluster id as a label, or ensuring cluster consistency across different views of the same image. In general, SSL methods are not designed to be deployed in non-stationary settings, which can lead to suboptimal performance (see section 5). Our proposed method addresses this specific issue.

Continual Learning The supervised CL literature is often divided into several families of methods, all with the principal objective of reducing catastrophic forgetting. In the fixed architecture setting we have prior based methods [Kirkpatrick et al., 2017; Zenke et al., 2017; Nguyen et al., 2017], which regularize the model to limit the change of parameters deemed important in previous tasks. Retrieval based methods [Chaudhry et al., 2019; Aljundi et al., 2019a], fight forgetting by reusing past samples alongside the incoming data. This data can be store in a small episodic memory [Aljundi et al., 2019b] or sampled from a generative model also trained continually [Shin et al., 2017; Van de Ven and Tolias, 2018]. Some replay based methods further aim at directly optimizing for the reduction of negative interference across tasks [Lopez-Paz and Ranzato, 2017; Riener et al., 2018; Chaudhry et al., 2018; Gupta et al., 2020].

Unsupervised Continual Learning Several works [Rao et al., 2019; Smith et al., 2019; Achille et al., 2018] learn representations in an unsupervised manner over non-iid data. We also note that some generative replay methods also fit into this category, namely ones where the generator does not leverage labelled data [Caccia et al., 2020a; Shin et al., 2017]. This line of work differs from the setting explored in this paper, as the representation learning occurs not during pretraining (where we assume iid access to some data) but over a stream of non-stationary data.

Meta-Learning The Meta-Learning [Ravi and Larochelle, 2016; Snell et al., 2017; Vinyals et al., 2016] - or learning to learn - field aims to design models which can acquire new skills or knowledge fast, via a small set of examples. Most of the literature has focused on the few-shot supervised learning setting, where agents are first pretrained offline and evaluated on their ability to label new classes given small support sets. Model-Agnostic Meta-Learning (MAML) [Finn et al., 2017] enables agents to learn a parameter initialization suitable for fast adaption, by directly optimizing through the update procedure performed by the model at deployment time. The flexibility of MAML has led its use to reach beyond few-shot learning.

Meta-Continual Learning Most relevant to our work is Online-Aware Meta-Learning (OML) [Javed and White, 2019], which designs a meta-objective using catastrophic interference as a training signal. In the pretraining step, the OML objective artificially creates a temporally correlated sequence through which the agent must learn. Follow up work proposed ANML [Beaulieu et al., 2020], a neuromodulated architecture which meta-learns a gating mechanism to reduce forgetting. Differently, recent works have propose the use of meta-learning objectives at continual-learning time [Gupta et al., 2020; Caccia et al., 2020b].

Unsupervised Meta-Learning Variants of MAML which do not require labelled data in the pretraining phase have been proposed [Hsu et al., 2018; Antoniou and Storkey, 2019; Khodadadeh et al., 2018]. These methods however are not designed for continual learning, as they have no mechanism to prevent catastrophic forgetting. Lastly and closest to our work is [Bertugli et al., 2021], where the authors propose a unsupervised pretraining strategy for learning representations in the continual few shot learning setting. Similar to [Hsu et al., 2018], the authors use an unsupervised embedding function with clustering to label examples, which are then used within the OML algorithm, resulting in a 3-step procedure for the pretraining phase. Our work instead proposes a simplified (single step) pretraining procedure, which is both conceptually simpler, and is easier to extend.

3 Background: Meta Continual Learning

In this section we formalize the learning setting under which our proposed method operates. We begin by first defining the general continual learning setting, in which (potentially pretrained) agents are deployed. We then describe meta-continual learning, a framework to pretrain agents before deployment in a continual learning setting.

We follow [Lopez-Paz and Ranzato, 2017; Javed and White, 2019] and define a Continual Learning Problem (CLP) as a potentially infinite stream of samples:

$$S = (X_1, Y_1), (X_2, Y_2), \ldots , (X_t, Y_t), \ldots ,$$

where input $X_t$ and output $Y_t$ belong to sets $\mathcal{X}$ and $\mathcal{Y}$ respectively. Crucially, examples are not drawn i.i.d from a fixed distribution over $(X, Y)$ pairs. Instead, we assume that at each timestep $t$, input samples are drawn according to the conditional distribution $P(X_t|C_t)$, where $C_t \in \mathcal{C}$ is a hidden time-dependent context variable [Caccia et al., 2020b], denoting the state of the distribution at time $t$. In practice, $C_t$ can be the previous frames of a video, or the cards currently dealt in a poker game. Given this context variable, samples can be drawn i.i.d. from $P(X_t|C_t)$, in other words the $C_t$ encapsulates all the non-stationarity of the CLP.

Our goal is to learn a $\theta$-parameterized predictor $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ which can be queried at any given time on samples belonging to previously seen contexts $C_{\leq t}$. For simplicity, we assume that the target function is deterministic and only depends on the input, i.e. $p(Y|X, C) = p(Y|X)$. Our goal is to minimize the following objective after $T$ timesteps

$$L_{\text{CLP}}(\theta) \overset{\text{def}}{=} \mathbb{E}[\ell(f_\theta(X), Y)]$$

$$= \sum_{t=1}^{T} \mathbb{E}_{p(x|C_t)}[\ell(f_\theta(X), Y)]$$
where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ denotes a loss function between the model predictions and labels.

**Meta CL.** In Meta Continual Learning, the goal is to pre-train the predictor $f_0$ for better performance, or transfer, on the CLP objective. In the pretraining phase, we aim to learn parameters $\theta$, such that at deployment time, the model can quickly adapt to any given CLP. To do so, we adopt the nested optimization procedure described in MAML [Finn et al., 2017] for the pretraining step. In the inner loop, we sample a sequence of contexts $C_1, ..., C_k$ from a distribution over context sequences $P(C_k)$. From this sequence, we then generate a trajectory $S_k = (X_1, Y_1), ..., (X_k, Y_k)$ with $X_i \sim P(X|C_i)$. This trajectory is equivalent to the support set in MAML, as it mimics the non stationary learning conditions encountered at deployment time. Starting from initial parameters $\theta$, we obtain fast parameters $\theta^k$ by processing the trajectory one (potentially batched) example at a time, performing $k$ SGD steps, with loss $\ell(f_\theta(X_i), Y_i)$ at each inner step. Denoting the one-step SGD operator with learning rate $\alpha$ as

$$U(\theta^0) = \theta^0 - \alpha \nabla_{\theta^0} \ell_0(\theta^0, S[0]) = \theta^1,$$  \hspace{1cm} (3)

the fast parameters $\theta_k$ are given by

$$\theta^k = U \circ ... \circ U(\theta^0).$$ \hspace{1cm} (4)

We proceed to evaluate parameters $\theta^k$ on a query set $Q$, obtaining the outer loss $\ell_{out}$. We recall that two key properties of a continual learner are its ability to leverage past skills to quickly learn new concepts and to learn without negative interference on previous knowledge. To do so, the OML [Javed and White, 2019] objective insightfully proposes to compose $Q$ with samples from current inner loop distribution $P(X|C_{ik})$ but also from concepts seen in previous iterations of the pretraining phase, which we denote by $C_{old}$. The full objective can be written as

$$\min_{\theta} \sum_{C_1, k \sim P(C_k)} \sum_{S_k \sim P(S_k|C_{1:k})} \ell_{out}(U(\theta, S_k), Q).$$ \hspace{1cm} (5)

Once $\ell_{out}$ has been evaluated, we backpropagate through the inner learning procedure, all the way to the initial $\theta$. Using this gradient we calculate a new set of initial weights $\theta := U(\theta, \ell_{out})$ and iterate again over the whole process until convergence.

In practice, [Javed and White, 2019] partition the full model $f_\theta : \mathcal{X} \to \mathcal{Y}$ into a model base (or trunk) $b_{\theta_0} : \mathcal{X} \to \mathbb{R}^d$ which extracts a learned representation, and a model head $h_{\theta_b} : \mathbb{R}^d \to \mathcal{Y}$ which predicts the final output from the learned representation. In other words, the model can be represented as $f_\theta = h_{\theta_b} \circ b_{\theta_0}$. In practice, the network base typically contains all convolutional blocks, while the head is a small multi-layer perceptron.

During the inner loop (and deployment), only the head parameters $\theta_h$ are updated via SGD. The parameters $\theta_b$ are only updated in the outer loop, meaning that latent representations for all inputs can be extracted in parallel, since $\theta_b$ is fixed. This modification also leads to training optimization procedure that is mode stable, leading to significantly better results.

Again, the key contribution of OML is that one can approximate the full $\mathcal{L}_{CLP}$ by artificially creating short non-stationary sequences and evaluating knowledge retention on a sample of past data. Crucially, the loss $\ell$ used in both inner and outer loop, is a supervised loss. Moreover, the non-stationary sequences are created using labelled information. In this work, we design a new algorithm for the pretraining phase which bypasses the need for labelled data. We discuss our approach next.
4 Proposed Method

In this section, we introduce the core components of our method, SPeCiaL. We first discuss our unsupervised labelling procedure, enabling SPeCiaL to work on unlabelled datasets. We then discuss our mechanism to fight catastrophic forgetting. Lastly, we discuss several modifications reducing the computational and memory footprint of our method, a key requirement when scaling to larger datasets. A full workflow of our method is shown in Figure 1.

4.1 Generating Pseudo Labels

Our labelling procedure relies on data augmentation in pixel space, in order to generate multiple transformed images, or views, that share common semantic features. Similar to instance-level classifiers, we propose to assign each image (and its generated views) to a unique label. However, unlike most instance-level classifiers, we do not require a large final linear layer, nor do we use noise contrastive estimation. Similar to OML, we reset class prototypes once their lifespan is complete, allowing for a tractable solution (more information in section 4.3). We follow the augmentation pipeline described in [Chen et al., 2020a], yielding a stochastic transformation operator consisting of three base augmentations: random cropping which is then resized back to the original size, random color distortions, and random Gaussian blur. We highlight that this procedure can operate online, meaning it does not require a separate isolated clustering step, as in [Bertugli et al., 2021; Hsu et al., 2018]. Moreover, by incorporating the labelling procedure within our variant of OML, labels are not static, hence the model is less prone to overfitting, and can benefit from longer training (see section 5.1).

4.2 Maximizing parallel computations

Processing the incoming batch of data sequentially hinders the ability to run computations in parallel. We make two modifications to circumvent this issue. First, following [Beaulieu et al., 2020], we partition the full model such that the model head contains only the last linear classification layer. Since only the head parameters have to be executed sequentially, this enables us to run the feature extraction process in parallel, which represents a significant portion of the computation. Second, we train on multiple sequences in parallel. During the inner loop procedure, instead of processing a single example per timestep, we instead process $M$ examples every step. This is equivalent to first generating $M$ independent streams, and processing (in batch) one example per stream at each timestep. This further maximizes the proportion of parallelizable operations. We refer to the number of sequences ($M$) in the inner loop as the meta batch size.

4.3 Delayed Feature Recall

The original OML objective proposes to sample across all previously seen contexts $C_{old}$ (or previously seen classes) to obtain samples for the outer loop, such that the model is incentivized to learn an update procedure which is robust to forgetting. In our current framework, this is not currently possible as it would require to keep in memory a number of vectors in the output layer that is proportional to the number of images seen. Instead, we propose to sample only from the last $N$ seen samples, by storing these recent samples in a queue. This way, only the output vectors of the samples in the queue need to be kept additionally in memory. As a consequence, the lifespan of a given label starts when an image first appears in the inner loop, and persists until the sample is fully discarded from the queue. Therefore, we only require that our model’s output size be larger than the sum of the number of unique images seen within a meta-training step, and the queue size. We note that when sampling a random label for an incoming image, we make sure to exclude already assigned labels in the queue to avoid collisions.

Furthermore, instead of storing raw images in the queue, we instead store the representation produced by the model trunk $b : \mathcal{X} \rightarrow \mathbb{R}^{d}$. This reduces both the computational and memory footprint of our algorithm, since we are reusing previously computed features. We note that in practice, using a small queue works best. Therefore the representations are stored in the queue for a few iterations only, and do not become stale, so no additional mechanism to counter staleness [He et al., 2020] is needed. The full workflow of our method can be seen in Figure 1.

5 Experiments

We design our experiments to evaluate how well the learned representations can transfer when learning new classes from a common domain.

Pretraining In practice, for each meta-train step we instantiate a Continual Learning Problem as follows: we first sample a set of (e.g. 5) context variables $\{C_1, \ldots, C_5\}$. From this set, we create a sequence of correlated contexts by repeating each context $n$ times: $(C_1^{(1)}, \ldots, C_1^{(n)}, C_2^{(1)}, \ldots, C_5^{(n)})$. In the supervised case, we create a context $C_i$ for each distinct label $Y_i$ in the meta-training set. Therefore, sampling from $P(X_i|C_i)$ is equivalent to sampling from the class-conditional distribution $P(X_i|Y_i)$. For the unsupervised case, we create a context $C_j$ for each image in the dataset, and sampling from $P(X_j|C_j)$ yields different augmented views of the $j$-th image in the meta-training set. For all methods considered, we cross validate the optimal value for the meta batch size.
Deployment: Few Shot Continual Learning Following [Javed and White, 2019; Beaulieu et al., 2020; Bertugli et al., 2021], our instantiation of a CLP follows the continual few shot learning procedure. At metatest time, we fix all layers except the last one (i.e. the RLN layers), as done in the inner loop procedure. We then proceed to learn a linear classifier (PLN) from a stream of $C$ classes, with $N_c$ samples for each class. The data is streamed example by example, one class at a time. In other words, the learner does $N_c \times C$ updates, seeing $N_c \times C$ samples (once), and observing $N_c − 1$ distribution shifts.

Datasets We experiment on two datasets. We first employ the MiniImagenet [Vinyals et al., 2016] dataset, commonly used in few shot learning. It comprises 100 classes, with 64 used for meta-training, 16 for meta-validation and 20 for meta-testing. Images are resized to 84 × 84. We also use the CIFAR-10/100[Krizhevsky et al., 2009] datasets. All the meta-training is performed on CIFAR-10, and we do an 50-50 split for CIFAR100 for meta-validation and meta-test respectively. Images are resized to 32 × 32.

Architecture For all baselines considered, we use a 4 block residual network. Each block consists of two computation paths, each with a single convolution. We use Weight Standardized [Qiao et al., 2019] convolutions with Group Normalization [Wu and He, 2018], a normalization approach with does not use batch-level statistics and has shown strong performance in small batch settings. We use MaxPooling to downsample our input when required, i.e. in the all blocks for MiniImagenet, and the first three blocks for CIFAR. The block’s architecture is illustrated in Figure 2. The key differences with previous architectures commonly used in Meta Learning, is the use of a computational path with no activations. We further discuss the impact of these architectural changes in section 5.2.

Baselines We mainly compare against OML, where supervised pretraining is used. Our OML implementation can optionally leverage tools specific to the SSL pipeline, namely the use of augmentations. We also benchmark against SimCLR [Chen et al., 2020a], a strong and flexible SSL model. For all approaches, we use the same architecture and training budget, allowing us to compare methods on equal footing and better isolate the contribution of our method.

Pretraining Models are trained for 350,000 optimization steps. For OML and SPeCiaL, a step corresponds to a full inner / outer loop, while for regular models it accounts for the processing of one minibatch. We found that for meta models, the AdamW optimizer with gradient clipping yielded best results. For other models, we used SGD with momentum. All models use a learning rate scheduler consisting of a linear warmup with cosine decay. For OML and SPeCiaL, we used cross-validation to determine the optimal sequence length, meta batch size and the number of distinct images per sequence. For SimCLR, we use the largest batch size which fits in memory, and the (batch size) adaptive learning rate procedure described in the paper. All experiments are executed on one Tesla V100 GPU.

MetaTesting Once the models are trained, we evaluate their performance on the downstream Continual Few Shot Learning Task. As in [Javed and White, 2019; Bertugli et al., 2021] we set the number of examples per class to 30. We follow [Javed and White, 2019] and cross-validate the learning rate for each metatest trajectory length reported.

5.1 Results

We first present results for the CIFAR experiment in Fig 3. In this experiment, SPeCiaL is actually able to outperform its fully supervised counterpart. Moreover, this performance gap widens as more tasks are being learned, suggesting that the representations learned by our model are better adapted for longer (and more challenging) scenarios. We generally found that self supervised method, e.g. SimCLR and SPeCiaL, can greatly benefit from longer training. Unlike OML, which starts overfitting after 100K steps, SPeCiaL continues to improve when given more computation. We illustrate this in Figure 4. This observation is consistent with [Assran et al., 2020], who note that when combining supervised and SSL losses, after a fixed number of epochs the supervised signal needs to be removed to avoid a decrease in performance.

We proceed with the MiniImagenet experiment, shown on the right of Fig 3. Here SPeCiaL is able to match the performance of OML. We note here that OML does not overfit as on the smaller scale CIFAR dataset. Instead, performance stagnates after 150K steps.

To summarize, early in training OML outperforms other
unsupervised methods. As training progresses, SpeCial is able to close the performance gap, and benefits more from longer training.

5.2 Sparse Representation is Architecture-Dependent

In previous Meta Continual Learning paper, it was shown that the model learns sparse representations as a way to mitigate catastrophic forgetting. The idea of using sparse representations has been proposed in earlier work [French, 1991], and revisited in modern CL methods [Liu et al., 2019; Aljundi et al., 2018]. Therefore, seeing sparsity emerge in a data-driven way from MCL algorithms, as reported in [Javed and White, 2019; Beaulieu et al., 2020], confirms that it is a good inductive bias for continual learning. In our work however, we found that the OML objective alone is not sufficient to obtain sparse representations: one must also use a suitable architecture for sparsity to naturally emerge. In the original work, the architecture used consisted of Convolutions and ReLU activations, hence the last operation performed by the RLN was a nonlinearity favorable to cancelling neurons.

We run an experiment where we add an additional ReLU activation after the final block of the RLN. We compare the activations obtained with this variant in Figure 5. We see that indeed sparsity emerges, which is not the case under the base architecture. Moreover, in the base architecture used in our experiments (see Fig. 2), one computational path finishes with a ReLU + MaxPool operation. Therefore the model could in practice only leverage this path and obtain sparse representations, without the explicit ReLU after the skip connection. Yet, we found that this skip connection is leveraged by the model, and overall stabilized training. We therefore conclude that the architecture used plays an important role in shaping the activation patterns. Moreover, we found that this additional ReLU activation after the final skip connection gave a model which was harder to optimize, and therefore led to worse performance.

6 Conclusion

In this work we proposed a novel unsupervised meta objective to learn representations suitable for Continual Learning. We showed that when deployed in a Continual Learning setting, the learned representations can match or exceed the performance of fully supervised meta algorithms. Furthermore, we showed that sparsity does not always occur when using Meta Continual Learning algorithms, and that the architecture of the model plays a significant role in this context.

Our work takes a first step towards leveraging unlabelled data to prepare a system before its deployment on non-stationary data. An important next step for this work is to investigate how we can scale up meta algorithms presented in this work, and see if they can excel in the large data regime like Self-Supervised Learning methods.
References

[Achille et al., 2018] Alessandro Achille, Tom Eccles, Loic Matthey, Christopher P Burgess, Nick Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. arXiv preprint arXiv:1808.06508, 2018.

[Aljundi et al., 2018] Rahaf Aljundi, Marcus Rohrbach, and Tinne Tuytelaars. Selfless sequential learning. arXiv preprint arXiv:1806.05421, 2018.

[Aljundi et al., 2019a] Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. arXiv preprint arXiv:1908.04742, 2019.

[Aljundi et al., 2019b] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. arXiv preprint arXiv:1903.08671, 2019.

[Antoniou and Storkey, 2019] Antreas Antoniou and Amos Storkey. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. arXiv preprint arXiv:1902.09884, 2019.

[Bertugli et al., 2021] Alessia Bertugli, Stefano Vincenzi, Simone Calderara, and Andrea Passerini. Generalising via meta-examples for continual learning in the wild. arXiv preprint arXiv:2101.12081, 2021.

[Caccia et al., 2020a] Lucas Caccia, Eugene Belilovsky, Massimo Caccia, and Joelle Pineau. Online learned continual compression with adaptive quantization modules. In International Conference on Machine Learning, pages 1240–1250. PMLR, 2020.

[Caccia et al., 2020b] Massimo Caccia, Pau Rodriguez, Oleksiy Ostapenko, Fabrice Normandin, Min Lin, Lucas Caccia, Issam Laradji, Irina Rish, Alexandre Lacoste, David Vazquez, et al. Online fast adaptation and knowledge accumulation: a new approach to continual learning. arXiv preprint arXiv:2003.05886, 2020.

[Caron et al., 2020] 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.

[Chaudhry et al., 2018] Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. arXiv preprint arXiv:1812.00420, 2018.

[Chaudhry et al., 2019] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and M Ranzato. Continual learning with tiny episodic memories. 2019.

[Chen et al., 2020a] 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.

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

[Finn et al., 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In International Conference on Machine Learning, pages 1126–1135. PMLR, 2017.

[French, 1991] Robert M French. Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In Proceedings of the 13th annual cognitive science society conference, volume 1, pages 173–178, 1991.

[Gupta et al., 2020] Gunshi Gupta, Karmesh Yadav, and Liam Paull. Look-ahead meta learning for continual learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, Advances in Neural Information Processing Systems, volume 33, pages 11588–11598. Curran Associates, Inc., 2020.

[He et al., 2020] 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.

[Hsu et al., 2018] Kyle Hsu, Sergey Levine, and Chelsea Finn. Unsupervised learning via meta-learning. arXiv preprint arXiv:1810.02334, 2018.

[Javed and White, 2019] Khurram Javed and Martha White. Meta-learning representations for continual learning. arXiv preprint arXiv:1905.12588, 2019.

[Khodadadeh et al., 2018] Siavash Khodadadeh, Ladiislaub Bölöni, and Mubarak Shah. Unsupervised meta-learning for few-shot image classification. arXiv preprint arXiv:1811.11819, 2018.

[Kirkpatrick et al., 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, André A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 114(13):3521–3526, 2017.

[Krizhevsky et al., 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.
[Liu et al., 2019] Vincent Liu, Raksha Kumaraswamy, Lei Le, and Martha White. The utility of sparse representations for control in reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 4384–4391, 2019.

[Lopez-Paz and Ranzato, 2017] David Lopez-Paz and Marc’Aurelio Ranzato. Gradient episodic memory for continual learning. arXiv preprint arXiv:1706.08840, 2017.

[Nguyen et al., 2017] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. arXiv preprint arXiv:1710.10628, 2017.

[Qiao et al., 2019] Siyuan Qiao, Huiyu Wang, Chenxi Liu, Wei Shen, and Alan Yuille. Weight standardization. arXiv preprint arXiv:1903.10520, 2019.

[Rao et al., 2019] Dushyant Rao, Francesco Visin, Andrei A Rusu, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Continual unsupervised representation learning. arXiv preprint arXiv:1910.14481, 2019.

[Ravi and Larochelle, 2016] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

[Riemer et al., 2018] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. arXiv preprint arXiv:1810.11910, 2018.

[Shin et al., 2017] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. arXiv preprint arXiv:1705.08690, 2017.

[Smith et al., 2019] James Smith, Seth Baer, Cameron Taylor, and Constantine Dovrolis. Unsupervised progressive learning and the stam architecture. arXiv preprint arXiv:1904.02021, 2019.

[Snell et al., 2017] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. arXiv preprint arXiv:1703.05175, 2017.

[Van de Ven and Tolias, 2018] Gido M Van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning. arXiv preprint arXiv:1809.10635, 2018.

[Vinyals et al., 2016] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. arXiv preprint arXiv:1606.04080, 2016.

[Wu and He, 2018] Yuxin Wu and Kaiming He. Group normalization. In Proceedings of the European conference on computer vision (ECCV), pages 3–19, 2018.

[Zenke et al., 2017] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In International Conference on Machine Learning, pages 3987–3995. PMLR, 2017.