Constrained Few-shot Class-incremental Learning

Michael Hersche[1,2] her@zurich.ibm.com
Geethan Karunaratne[1,2] kar@zurich.ibm.com
Giovanni Cherubini[1] cbi@zurich.ibm.com
Luca Benini[2] lbenini@iis.ee.ethz.com
Abu Sebastian[1] ase@zurich.ibm.com
Abbas Rahimi[1] abr@zurich.ibm.com

[1]IBM Research-Zurich, [2]ETH Zurich

Abstract

Continually learning new classes from fresh data without forgetting previous knowledge of old classes is a very challenging research problem. Moreover, it is imperative that such learning must respect certain memory and computational constraints such as (i) training samples are limited to only a few per class, (ii) the computational cost of learning a novel class remains constant, and (iii) the memory footprint of the model grows at most linearly with the number of classes observed. To meet the above constraints, we propose C-FSCIL, which is architecturally composed of a frozen meta-learned feature extractor, a trainable fixed-size fully connected layer, and a rewritable dynamically growing memory that stores as many vectors as the number of encountered classes. C-FSCIL provides three update modes that offer a trade-off between accuracy and compute-memory cost of learning novel classes. C-FSCIL exploits hyperdimensional embedding that allows to continually express many more classes than the fixed dimensions in the vector space, with minimal interference. The quality of class vector representations is further improved by aligning them quasi-orthogonally to each other by means of novel loss functions. Experiments on the CIFAR100, mini-ImageNet, and Omniglot datasets show that C-FSCIL outperforms the baselines with remarkable accuracy and compression. It also scales up to the largest problem size ever tried in this few-shot setting by learning 423 novel classes on top of 1200 base classes with less than 1.6% accuracy drop. Our code is available at https://github.com/IBM/constrained-FSCIL.

1. Introduction

Deep convolutional neural networks (CNNs) have achieved remarkable success in various computer vision tasks, such as image classification [15,24,26,47], stemming from the availability of large curated datasets as well as huge computational and memory resources. This, however, poses significant challenges for their applicability to smart agents deployed in new and dynamic environments, where there is a need to continually learn about novel classes from very few training samples, and under resource constraints. We consider the challenging scenario of learning from an online stream of data, including never-seen-before-classes, where we impose constraints on the sample size, computational cost, and memory size.

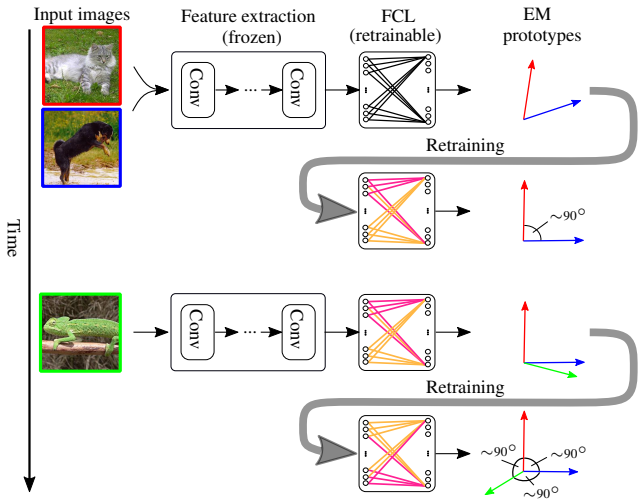

Figure 1. Overview of C-FSCIL which maps input images to quasi-orthogonal prototypes such that the prototypes of different classes encounter small interference.

Let us first focus on the sampling constraint of training data. Inspired by human-like sequential learning, classical connectionist networks can be naively trained sequentially, e.g., first on a set of old classes, and then on a set of novel classes, whereby the training dataset of old classes is no longer available. As a result, the new learning may interfere catastrophically with the old learning by overwriting weights involved in representing the old learning (and thereby forgetting) [33]. This effect is known as *catastrophic interference*, or *catastrophic forgetting*, and causes the classification accuracy to deteriorate [12,33]. To address the catastrophic forgetting problem, research efforts have

been directed to, e.g., freezing parts of the network weights, while simultaneously growing other parts of the network to extend the learning ability. Among them, class-incremental learning (CIL) [4, 18, 34, 39, 52] aims to learn a unified classifier in which the encountered novel classes—that were not seen before in the continual data stream—are added into the recognition tasks without forgetting the previously observed classes. One step further, very recently, few-shot CIL (FSCIL) [2, 5–7, 40, 44, 48, 56] algorithms have been proposed to continually extend learning to novel classes with only a few data samples. Requiring FSCIL to be trained with very few novel training samples makes it more challenging compared to CIL, which usually learns new classes from large-scale training samples.

To facilitate few-shot learning, memory-augmented neural networks (MANNs) *separate* information processing from memory storage [13, 14, 21, 23, 42, 46, 51]. MANNs incorporate an *explicit* memory (EM) into an embedding network such that the network can write the embedding of few data samples to the memory as class prototypes, and read from these individual entries during inference. This separation enables the network to offload new prototypes to the EM, where they do not endanger the previously learned prototypes to be overwritten, leading to remarkable accuracy with few classes (typically 20 classes). However, this neat feature of MANNs has not been exploited in CIL or FSCIL. This is mainly due to the fact that the interference between different class prototypes increases with a growing number of classes, as experienced in FSCIL. To allow continual learning in MANNs, the representational power of the embedding network should be naturally extensible to a large number of classes with minimal interference. This critical requirement can be met by exploiting hyperdimensional computing [10, 19, 35], where classes can be represented by random high-dimensional vectors with a dimensionality in the order of thousands. This new combination of MANNs and hyperdimensional computing has been recently developed in [21] for few-shot learning.

Hyperdimensional computing is characterized by the following properties: (i) A randomly chosen vector is quasi-orthogonal to other random vectors with very high probability (the "curse" of dimensionality), therefore the representation of a novel class is not only incremental to the old learning but also causes minimal interference. This phenomenon is known as concentration of measure [28], with the peculiar property that pseudo-orthogonality converges to exact orthogonality with increasing dimensionality. (ii) The number of such quasi-orthogonal vectors grows exponentially with the dimensionality, which provides a sufficiently large capacity to accommodate novel classes over time. (iii) Counterintuitively, quasi-orthogonal vectors can still encode semantic information. We can describe a concept in a scene (e.g., a black dog) with a vector by binding quasi-orthogonal atomic vectors ($\mathbf{x}_{\text{black}} \circledast \mathbf{x}_{\text{dog}}$), which is quasi-orthogonal to all other involved vectors (atomic and composite). The bound vector can be decomposed to $\mathbf{x}_{\text{black}}$ and $\mathbf{x}_{\text{dog}}$, revealing the semantic relation between a black dog and a black cat (both include $\mathbf{x}_{\text{black}}$) [9, 16]. In fact, fixed quasi-orthogonal vectors have been successfully used as class vectors achieving improved performance in the supervised classification tasks [16, 17].

In this paper, we enhance MANNs by exploiting the representational power of hyperdimensional computing to perform FSCIL. During FSCIL operations, it is constrained to either no gradient updates or a small constant number of iterations for learning novel classes, and a linear growth in the memory size with respect to the encountered classes. The contributions of this work are summarized as follows:

First, we propose C-FSCIL, which is architecturally composed of a frozen feature extractor, a trainable fixed-size fully connected layer, and a rewritable dynamically growing EM that stores as many vectors as the number of classes encountered so far (See Fig. 1). The frozen part is separated from the growing part by inserting the fully connected layer, which outputs class vectors in a hyperdimensional embedding space whose dimensionality remains fixed, and is therefore independent of the number of classes in the past and future. The feature extractor is a CNN that is meta-learned by proper sharpened attention, which strives to represent dissimilar images with quasi-orthogonal vectors. The C-FSCIL architecture is presented in Section 4.1.

Second, C-FSCIL with three update modes offers a trade-off between accuracy and the compute-memory cost of learning a novel class. The simple yet powerful Mode 1 creates and updates an averaged prototype vector as the mean-of-exemplars in the EM, without any gradient computation (see Section 4.2.1). Mode 2 bipolarizes the prototype vectors, and retrains the fully connected layer without exceeding a small constant number of iterations; for retraining it requires storing an averaged activation pattern for every class in a globally averaged activation (GAA) memory (see Section 4.2.2). Mode 3 nudges the averaged prototype vectors to align them quasi-orthogonally to each other, while remaining in close proximity to the original averaged prototypes, using a combination of novel loss functions. It then retrains the fully connected layer without exceeding the constant number of iterations (see Section 4.2.3).

Third, C-FSCIL leads to higher accuracy, compute-memory efficiency, and scalability compared to FSCIL baselines, as shown in Section 5. Experiments on the CIFAR100, miniImageNet, and Omniglot datasets demonstrate that C-FSCIL outperforms the baselines even in Mode 1 where prototypes are simply obtained in one pass without any gradient-based parameter updates. In Omniglot, C-FSCIL scales up to 1623 classes, whereby 423 novel classes are incrementally added to 1200 base classes, with less than 2.6%, 1.4%, and 1.6% accuracy drops when using Modes 1, 2, and 3, respectively. Thanks to the quasi-orthogonality of the class prototypes in the EM, they can be compressed by 2×, causing 1.7%–3.5% accuracy drop during the course of FSCIL.

2. Related Work

2.1. MANNs

MANNs have been used in one- and few-shot learning tasks [21, 23, 42, 50, 51]. First, their embedding network is meta-learned on a set of initial categories of so-called base classes. After meta-learning, the network is ready to provide new representations for never-seen-before categories (i.e., novel classes) that can be independently stored, or retrieved via the EM. In these works, training is done on the base classes and inference is done solely on the novel classes, as opposed to the union of the two sets of classes as required by FSCIL. Further, the size of the EM is set by c vectors [42, 50], or $c \times k$ vectors [21, 53, 54] where c and k represent the number of classes and training examples per class, respectively. The contents of the EM can be compressed using outer products with randomized labels [23]. In a similar vein, C-FSCIL superposes two class vectors to store $c/2$ vectors in the EM, as described in the following.

2.2. Rehearsal and Pattern Replay

To address the catastrophic forgetting issue in CIL, rehearsal methods store some old training data, and replay them while learning novel classes during the incremental stage [1, 3, 31, 32, 36, 39]. For instance, a flexible number of exemplars per class are stored in an *episodic memory*. It provides the training data of the old classes as well as the currently available classes to a classifier to incrementally learn novel classes. The memory footprint of the episodic memory is kept bounded by relying on entry/exit criteria [1, 3, 31, 36, 39]. Alternatively, the exemplars of past classes can be generated using GANs with extra computational cost [8]. The storage and computational overhead of the rehearsal methods can be reduced by *latent replay* [34] and its TinyML evolution [38]. Instead of storing a portion of past exemplars in the input space, the latent replay method stores the corresponding activation patterns from an intermediate layer [34]. The stored activations are replayed to retrain all the layers above the latent replay layer.

Our C-FSCIL avoids the rehearsal and pattern replay overheads because it does not need any access to the previous data in any form. Instead, it maintains a highly compressed, minimal amount of past knowledge, either in the EM (Mode 1) or in the GAA memory (Modes 2-3), which is similar or smaller than the compared methods. Note that in Modes 2-3 only storing the GAA memory is sufficient as it contains c compressed activations, from which the EM can be *rematerialized* on the fly. For discriminating c classes, one needs to maintain at least c prototypes (class vectors); thus, a linear increase with the number of classes is unavoidable. In addition, if the memory constraints become even tighter, our quasi-orthogonal design comes to rescue to further compress either of the memories by binding each vector with a randomly drawn key and superimposing two key-prototype pairs [35], yielding twofold compression with moderate accuracy drop (See Appendix A.3).

2.3. Class Imbalance and Reparameterizations

Another issue in CIL is the class imbalance problem, where the norm or the bias can be unbalanced for the classes observed later in the incremental stage, causing the network's prediction to be biased towards novel classes [18, 52]. Learning novel classes might also interfere with the past classes. To mitigate the impact of class imbalance, recent CIL approaches adopt the cosine distance metric [18], or add a bias correction layer [52] to avoid the norm and bias imbalance. In a similar vein, a weight aligning method has been proposed to align the norms of the weight vectors for novel classes to those for past classes [57]. Another source of perturbation is the gradient from novel class observations that can affect past classes to change their weights. Regularization strategies have been aimed at avoiding such forgetting [22], but they have been recently shown to be insufficient in CIL [30]. Various maskings have been proposed to apply the gradient only to a subset of classes during backpropagation [29].

We avoid these issues systematically in C-FSCIL, where hyperdimensional quasi-orthogonal vectors are assigned to each and every class with the aim of reducing interference. This cannot be achieved with other methods that replace the fully connected layer with a fixed Hadamard [17] or identity [37] matrix, because they fail to support a larger number of classes than the vector dimensionality of the layer they are connected to. Our prototypes are stored in the EM, the cosine similarity is used for their comparison, and they can be selectively updated. The alignment to prototypes can be improved by proper retraining of the fully connected layer whose structure remains fixed and independent of the number of classes (See Fig. 1).

2.4. Few-shot Class-incremental Learning (FSCIL)

Very recently, FSCIL [5–7, 44, 48, 56] has been proposed for tackling CIL with very few training samples through a number of incremental sessions. Various solutions have been proposed, such as exploiting a neural gas network [48], a graph attention network [56], semantic word embeddings [6], and vector quantization [5]. Specifically, a pretrained backbone is decoupled from a non-parametric class mean classifier whose weights can be progressively adapted across sessions by a graph attention network [56]. C-FSCIL, even with the simple Mode 1, which does not need retraining, sets a new state-of-the-art accuracy compared to the previous FSCIL [5–7, 44, 48, 56], and demonstrates its extensible representation by supporting the maximum number of encountered classes (1623) in this setting. In fact, all previous works have used up to 200 classes. To be able to do retraining in the other modes, C-FSCIL requires the GAA memory to store as many compressed activation patterns as there are encountered classes. It allows to stably retrain with any combination of sessions or classes over time, and with an improved accuracy compared to Mode 1.

3. Notations and Preliminaries

3.1. Problem Formulation

FSCIL sequentially provides training sets $\mathcal{D}^{(1)}$, $\mathcal{D}^{(2)}$, ..., $\mathcal{D}^{(s)}$,...,$\mathcal{D}^{(S)}$, where $\mathcal{D}^{(s)} := \{(\mathbf{x}_n^{(s)}, y_n^{(s)})\}_{n=1}^{|\mathcal{D}^{(s)}|}$ with input data $\mathbf{x}_n^{(s)}$, e.g., an image, and corresponding ground-truth labels $y_n^{(s)}$. The labels $y_n^{(s)} \in \mathcal{C}^{(s)}$ are mutually exclusive across different training sets, i.e., $\forall i \neq j, \mathcal{C}^{(i)} \cap \mathcal{C}^{(j)} = \emptyset$. The training sets are denoted as *sessions*, whereby the first training set is denoted as the *base session*, providing a larger number of training examples and classes. The goal of the base session is to learn a meaningful representation, where we can distinguish between the ways (i.e., classes) in $\mathcal{C}^{(1)}$. The following sessions provide training sets $\mathcal{D}^{(s)}$ of size $|\mathcal{D}^{(s)}| = c \cdot k$, where $c = |\mathcal{C}^{(s)}|$ is the number of ways and k the number of training samples per way, hence we call it c-way k-shot. In a given session s, one only has access to the corresponding training set $\mathcal{D}^{(s)}$; the training sets of the previous sessions $1, 2, ..., s-1$ are not available anymore. The model is tested on a session-specific evaluation set $\mathcal{E}^{(s)}$ that contains samples from all previous sessions as well as the current one, i.e., $\forall j < i, \mathcal{E}^{(j)} \subset \mathcal{E}^{(i)}$. We denote the set of classes that have to be covered during session s as $\tilde{\mathcal{C}}^{(s)} := \cup_{i=1}^{s} \mathcal{C}^{(i)}$.

4. Proposed Method: C-FSCIL

4.1. Architecture

C-FSCIL consists of three main components: a feature extractor, a fully connected layer, and an EM. It can also have an auxiliary GAA memory, which will be described in Section 4.2.1. The feature extractor maps the samples from the input domain $\mathcal{X}$ to a feature space: $f_{\theta_1} : \mathcal{X} \rightarrow \mathbb{R}^{d_f}$, where θ_1 are the feature extractor's learnable parameters. As the feature extractor, we use a five-layer CNN, or a ResNet-12, to map an input image to the feature space. To form the embedding network with a hyperdimensional distributed representation, the feature extractor is connected to a fully connected layer $g_{\theta_2} : \mathbb{R}^{d_f} \rightarrow \mathbb{R}^{d}$, containing $\theta_2 \in \mathbb{R}^{d \times d_f}$ trainable parameters where $d \leq 512$. Note that d should be large enough to ensure that the expected similarity between randomly drawn d-dimensional vectors is approximately zero with a very high probability [19], but preferably $d < |\tilde{\mathcal{C}}^{(S)}|$. We denote the union of the trainable parameters in the feature extractor and the fully connected layer as $\theta = (\theta_1, \theta_2)$.

The fully connected layer produces a support vector for every training input. These support vectors are combined to compute a set of d-dimensional prototype vectors to be stored in the EM (more details in Section 4.2). Besides storing the prototype vectors, the EM contains a value memory of one-hot labels, and it is overall referred to as a key-value memory. The prototype vectors stored in the EM are not accessed by stating a discrete address, but by comparing the cosine similarities between a query from the embedding network and all the prototype vectors. Given a query $\mathbf{x} \in \mathcal{E}^{(s)}$ and prototypes $\mathbf{P}^{(s)} := (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_{|\tilde{\mathcal{C}}^{(s)}|})$, we compute the score l_i for class $i \in \tilde{\mathcal{C}}^{(s)}$ as follows:

$$l_i = \cos(\tanh(g_{\theta_2}(f_{\theta_1}(\mathbf{x}))), (\tanh(\mathbf{p}_i))), \qquad (1)$$

where $\tanh(\cdot)$ is the hyperbolic tangent function and $\cos(\cdot, \cdot)$ the cosine similarity. The hyperbolic tangent has proven to be a useful non-linearity in the hyperdimensional MANNs [21], limiting the norm of activated prototypes and embedding outputs. Moreover, the cosine similarity resolves the norm and bias issues usually encountered in FSCIL by focusing on the angle between the activated prototypes and embedding outputs, while ignoring their norm [18, 29]. Overall, this forms a content-based attention mechanism between the embedding network and the EM by computing a similarity score for each memory entry with respect to a given query. This attention vector is sharpened by a soft absolute sharpening function $\epsilon(\cdot)$ leading to

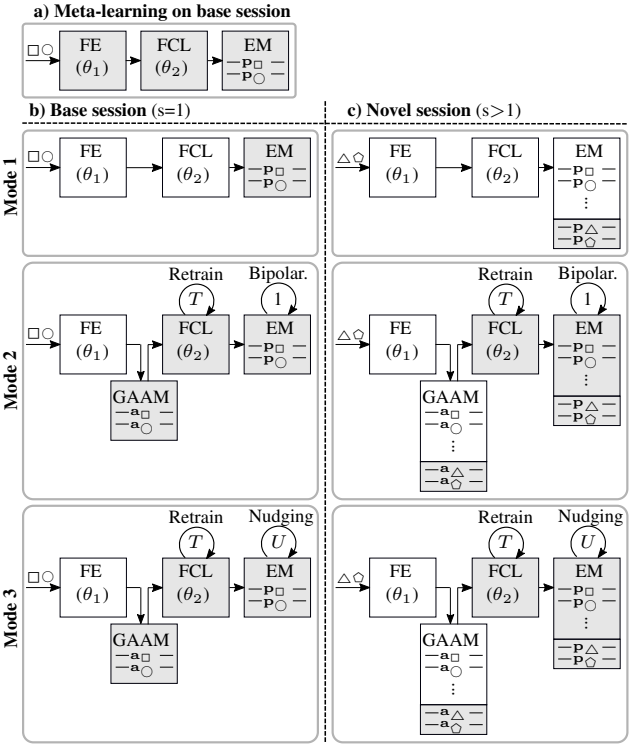

Figure 2. The C-FSCIL architecture consists of a feature extraction (FE), a fully connected layer (FCL), and an explicit memory (EM). It also requires a globally average activation memory (GAAM) only for Modes 2 and 3. It is shown how these components will evolve during **a)** meta-learning, **b)** base session, and **c)** novel sessions; the updated components are marked in gray while the frozen ones are kept in white. The three update modes are shown as well, with emphasis on which components they update and for how many iterations (1, T, and U), and which memory they expand linearly with the number of classes. All three Modes require meta-learning prior to their base session.

quasi-orthogonality [21]. See Appendix A.2.2. The resulting attention vector serves to read out the value memory.

The embedding network (θ_1, θ_2) and the EM are meta-learned by solving various few-shot problem sets that gradually enhance the quality of the mapping (see Fig. 2a). This is done by gradually learning to assign nearly quasi-orthogonal vectors to different image classes, and mapping them far away from each other in the hyperdimensional space. This space does not run out of such dissimilar vectors for newly encountered classes. To improve inter-class separation, C-FSCIL builds on top of the meta-learning setup and offers three modes for updating that are described in the following section.

4.2. Update Modes

C-FSCIL does not need to keep track of all class representations in a centralized fashion. Rather, it relies on the hyperdimensional embedding space to incrementally create maximally separable classes with three different techniques that trade off memory-compute cost for improved accuracy. In the first mode, the embedding network is already enforced (during meta-learning) to provide a valid quasi-orthogonal vector for a novel class, as described in Section 4.2.1. This mode learns online without any gradient-based parameter update. In the second mode, the averaged prototypes are bipolarized to achieve better quasi-orthogonality, followed by the retraining of the fully connected layer (Section 4.2.2). In the third mode, the quality of separation is further improved by prototype nudging and fine-tuning of the fully connected layer (Section 4.2.3). The three modes are illustrated in Fig. 2. None of them update the weights in the feature extractor after the meta-learning.

4.2.1 Mode 1: Averaged Prototypes

Earlier works on few-shot learning networks [45, 50] have proposed the use of class means as a single prototype per class, and then classified a query to the nearest prototype. Inspired by these networks, we also represent each class by a single prototype vector by computing the class-wise average over all the support vectors that belong to a class. The prototypes are stored in the EM, which continuously expands over time as new prototypes are computed in the new sessions (see Fig. 2b and c; Mode 1). The EM stores prototypes $\mathbf{P}^{(s)} = (\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_{|\tilde{\mathcal{C}}^{(s)}|}), \mathbf{P}^{(s)} \in \mathbb{R}^{d \times |\tilde{\mathcal{C}}^{(s)}|}$ of all classes that have been exposed to the model so far, whereby a prototype for class i is determined as follows:

$$\mathbf{p}_i(\mathcal{D}^{(s)}, \theta) = \frac{1}{k} \sum_{\substack{n=1 \\ \text{s.t. } y_n^{(s)}=i}}^{|\mathcal{D}^{(s)}|} g_{\theta_2}\left(f_{\theta_1}\left(\mathbf{x}_n^{(s)}\right)\right) \tag{2}$$

$$= g_{\theta_2}\left(\frac{1}{k} \sum_{\substack{n=1 \\ \text{s.t. } y_n^{(s)}=i}}^{|\mathcal{D}^{(s)}|} f_{\theta_1}\left(\mathbf{x}_n^{(s)}\right)\right) \tag{3}$$

$$= g_{\theta_2}\left(\mathbf{a}_i(\mathcal{D}^{(s)}, \theta_1)\right), \forall i \in \mathcal{C}^{(s)}. \tag{4}$$

The linearity of $g_{\theta_2}(\cdot)$ enables the step from (2) to (3). Each $\mathbf{a}_i$, as a d_f-dimensional compressed vector, represents the globally averaged activations of class i, and allows the determination of the corresponding prototype using $g_{\theta_2}(\cdot)$. The globally averaged activation (GAA) memory keeps track of all past averaged activations $\mathbf{A}^{(s)} := \left(\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_{|\tilde{\mathcal{C}}^{(s)}|}\right), \mathbf{A}^{(s)} \in \mathbb{R}^{d_f \times |\tilde{\mathcal{C}}^{(s)}|}$. Mode 1 does not need the GAA memory, because the GAA memory maintains the compressed activations of the past classes that will be needed for retraining in the next two modes.

4.2.2 Mode 2: Retraining on Bipolarized Prototypes

As the number of prototypes increases with the new sessions, they fall short in discriminating between different classes due to insufficient inter-class separability. To this end, we adjust the prototypes and the fully connected layer to better guide the activations that are provided by the frozen feature extractor. We follow a two-stage strategy, where we first adjust the prototypes and then retrain the fully connected layer to align the averaged activations with the newly adjusted prototypes.

The first step tries to create separation between nearby prototype pairs, which optimally yields close to zero cross-correlation between the prototypes pairs. A computationally cheap yet effective option is to add some sort of noise to the prototypes, e.g., quantization noise. We propose to quantize the prototypes to bipolar vectors by applying the element-wise sign operation, defined as $\mathbf{K}^* = \text{sign}(\mathbf{P}^{(s)})$.

Next, the embedding has to be retrained such that its output aligns with the bipolarized prototypes. Instead of attempting to optimize every training sample, we aim to align the globally averaged activations available in the GAA memory, defined in (2), with the bipolarized prototypes ($\mathbf{K}^*$). The final fully connected layer of the embedding network has the task of mapping localist features from the feature extractor to a distributed representation. Hence, we find that exclusively updating the parameters of the fully connected layer θ_2 is sufficient, while the parameters of the feature extraction θ_1 are kept frozen during retraining. Due to the averaged prototype-based retraining and the linearity of the fully connected layer, it is sufficient to pass the averaged activations from the GAA memory through the fully connected layer.

The fine-tuning of the fully connected layer is based on iterative updates using a loss function that strives to maximize the similarity between the averaged activations and the

bipolar prototypes:

$$\mathcal{L}_{\mathrm{F}}\Big(\theta_2^{(t)}, \mathbf{K}^*, \mathbf{A}\Big) = -\sum_{i=1}^{|\tilde{\mathcal{C}}^{(s)}|} \cos\Big(\tanh(\mathbf{k}_i^*), \tanh(g_{\theta_2^{(t)}}(\mathbf{a}_i))\Big) \tag{5}$$

$$\theta_2^{(t+1)} = \theta_2^{(t)} - \beta \frac{\partial \mathcal{L}_{\mathrm{F}}\Big(\theta_2^{(t)}, \mathbf{K}^*, \mathbf{A}\Big)}{\partial \theta_2^{(t)}} \tag{6}$$

where β is the update rate.

After T iterations of parameter updates, the final prototypes $\mathbf{P}^*$ are determined by passing the globally averaged activations through the fully connected layer one last time. The fully connected layer was already fine-tuned on the quasi-orthogonal bipolarized prototypes and therefore these final generated prototypes also tend to be quasi-orthogonal. Moreover, the final prototypes provide a better alignment than the earlier bipolarized prototypes. Fig. 2 illustrates this process in Mode 2.

4.2.3 Mode 3: Retraining on Nudged Prototypes

This mode proposes an improved prototype alignment strategy based on solving an optimization problem instead of simply bipolarizing the prototypes. Given a set of initial prototypes, generated by the meta-learned embedding network, C-FSCIL first optimizes them to yield the nudged prototypes. C-FSCIL then fine-tunes the fully connected layer to align it with the nudged prototypes.

We aim to find the nudged prototypes that simultaneously a) improve the inter-class separability by attaining a lower similarity between the pairs of nudged prototype vectors, and b) remain close to the initial averaged prototypes generated by the meta-learned embedding network. We set the initial nudged prototypes to the current prototypes stored in the EM, i.e., $\mathbf{K}^{(0)} := \mathbf{P}^{(s)}$. The nudged prototypes are then updated U times in a training loop to find an optimal set of prototypes unique to the given $\mathbf{A}^{(s)}$ available in the GAA memory (see Fig. 2 Mode 3).

The updates to the prototypes are based on two distinct loss functions that aim to meet the two aforementioned objectives. The first main objective is to decrease the inter-class similarity, which is achieved by minimizing the cross-correlation between the prototypes:

$$\mathcal{L}_{\mathrm{O}}(\mathbf{K}^{(u)}) = \sum_{\substack{i,j=1 \\ i \neq j}}^{|\tilde{\mathcal{C}}^{(s)}|} \sigma\Big(\cos\Big(\tanh(\mathbf{k}_i^{(u)}), \tanh(\mathbf{k}_j^{(u)})\Big)\Big), \tag{7}$$

The activation function $\sigma(\cdot)$ penalizes prototype pairs with large absolute cross-correlations:

$$\sigma(c) := e^{\alpha \cdot c} + e^{-\alpha \cdot c} - 2, \forall c \in \mathbb{R}, \tag{8}$$

where $\alpha = 4$ controls the steepness of the loss. Optimally, all of the prototypes would be orthogonal due to having zero cross-correlation, which is not possible when the growing number of classes exceeds the number of dimensions. Instead, we focus on finding the quasi-orthogonal vectors that can meet the constraint of $d < |\tilde{\mathcal{C}}^{(s)}|$. There is a large number of quasi-orthogonal vectors that can be found by minimizing the proposed loss function $\mathcal{L}_{\mathrm{O}}$.

The second objective is to keep the updated prototypes similar to the initial prototypes $\mathbf{K}^{(0)}$. This avoids significant deviations from the original representations of the initial base categories on which the embedding network was trained during meta-learning. We retain high similarity between the currently updated and the initial prototypes by adding the following loss function:

$$\mathcal{L}_{\mathrm{M}}(\mathbf{K}^{(u)}, \mathbf{K}^0) = -\sum_{i=1}^{|\tilde{\mathcal{C}}^{(s)}|} \cos\Big(\tanh(\mathbf{k}_i^{(u)}), \tanh(\mathbf{k}_i^{(0)})\Big). \tag{9}$$

Finally, the nudged prototypes are updated for U iterations, whereby one update is defined as:

$$\mathbf{K}^{(u+1)} = \mathbf{K}^{(u)} - \gamma \frac{\partial(\mathcal{L}_{\mathrm{O}}(\mathbf{K}^{(u)}) + \mathcal{L}_{\mathrm{M}}(\mathbf{K}^{(u)}, \mathbf{K}^{(0)}))}{\partial \mathbf{K}^{(u)}}, \tag{10}$$

where γ denotes the update rate. The final nudged prototypes $\mathbf{K}^* := \mathbf{K}^{(U)}$ will be used to retrain the fully connected layer for T iterations using the loss (5) and the update rule (6). Akin to Mode 2, the final prototypes are determined by passing the globally averaged activations through the fully connected layer after retraining.

5. Experiments

5.1. Datasets

We evaluate our methods on miniImageNet [41], CIFAR100 [25], and Omniglot [27]. For the evaluation on miniImageNet and CIFAR100, we follow the same FSCIL procedure as in [48], dividing the dataset into a base session with 60 classes and eight novel sessions with a 5-way 5-shot problem each. For Omniglot, we propose a new split that follows the common practice in FSCIL [48]. It contains a base session with 1200 classes and nine novel sessions with a 47-way 5-shot problem each, yielding 423 novel classes overall. See Appendix A.1 for more details.

5.2. Experimental Setup

miniImageNet and CIFAR100. For the natural image datasets, we use a Resnet-12 architecture as feature extractor [11,55]. It consists of four residual blocks with block dimensions [64, 160, 320, 640], each containing three convolutional layers with batchnorm and ReLU activation. After the final global average pooling, we get output activations

Table 1. Classification accuracy (%) on miniImageNet in the 5-way 5-shot FSCIL setting. [∗]: Upper bound based on the visual illustration in the corresponding work.

Session (s)	1	2	3	4	5	6	7	8	9
No. of classes $\lvert\tilde{\mathcal{C}}^{(s)}\rvert$	60	65	70	75	80	85	90	95	100
AL-MML [48]	61.31	50.09	45.17	41.16	37.48	35.52	32.19	29.46	24.42
IDLVQ-C [5]	64.77	59.87	55.93	52.62	49.88	47.55	44.83	43.14	41.84
Semantic KD* [6]	<62	<59	<54	<50	<49	<45	<42	<40	<39
VAE* [7]	<62	<60	<54	<52	<50	<49	<46	<44	<43
F2M [44]	67.28	63.80	60.38	57.06	54.08	51.39	48.82	46.58	44.65
CEC [56]	72.00	66.83	62.97	59.43	56.70	53.73	51.19	49.24	47.63
C-FSCIL Mode 1 d=512 (ours)	76.37	70.94	66.36	62.64	59.31	56.02	53.14	51.04	48.87
C-FSCIL Mode 2 d=512 (ours)	**76.45**	**71.23**	**66.71**	63.01	60.09	56.73	53.94	52.01	50.08
C-FSCIL Mode 3 d=512 (ours)	76.40	71.14	66.46	**63.29**	**60.42**	**57.46**	**54.78**	**53.11**	**51.41**

Table 2. Classification accuracy (%) on CIFAR100 in the 5-way 5-shot FSCIL setting. [∗]: Upper bound based on the visual illustration in the corresponding work.

Session (s)	1	2	3	4	5	6	7	8	9
No. of classes $\lvert\tilde{\mathcal{C}}^{(s)}\rvert$	60	65	70	75	80	85	90	95	100
AL-MML [48]	64.10	55.88	47.07	45.16	40.11	36.38	33.96	31.55	29.37
Semantic KD* [6]	<64	<57	<51	<46	<43	<41	<39	<37	<35
VAE* [7]	<62	<58	<57	<52	<51	<49	<46	<45	<42
F2M [44]	64.71	62.05	59.01	55.58	52.55	49.96	48.08	46.67	44.67
CEC [56]	73.07	68.88	65.26	61.19	58.09	55.57	53.22	51.34	49.14
C-FSCIL Mode 1 d=512 (ours)	77.47	72.20	67.53	63.23	59.58	56.67	53.94	51.55	49.36
C-FSCIL Mode 2 d=512 (ours)	**77.50**	**72.45**	**67.94**	**63.80**	**60.24**	**57.34**	**54.61**	52.41	50.23
C-FSCIL Mode 3 d=512 (ours)	77.47	72.40	67.47	63.25	59.84	56.95	54.42	**52.47**	**50.47**

Table 3. Classification accuracy (%) on Omniglot in the 47-way 5-shot FSCIL setting. [∗]: Reproduced baselines.

Session (s)	1	2	3	4	5	6	7	8	9	10
No. of classes $\lvert\tilde{\mathcal{C}}^{(s)}\rvert$	1200	1247	1294	1341	1388	1435	1482	1529	1576	1623
ProtoNet* [45]	70.61	70.20	70.01	69.68	69.48	68.99	68.74	68.07	67.60	67.41
CEC* [56]	78.91	79.07	78.74	78.60	77.94	77.55	77.18	76.77	76.39	76.11
C-FSCIL Mode 1 d=512 (ours)	84.16	83.82	83.69	83.32	83.22	82.78	82.70	82.32	81.77	81.56
C-FSCIL Mode 2 d=512 (ours)	86.87	86.77	86.57	86.44	86.40	86.20	**86.25**	85.96	85.63	85.49
C-FSCIL Mode 3 d=512 (ours)	**87.21**	**87.03**	**86.89**	**86.60**	**86.43**	**86.32**	86.13	**85.98**	**85.59**	**85.70**

of dimension $d_f = 640$. The fully connected layer has dimension $d = 512$. Motivated by [49] to derive good visual representations in the embedding, we pretrain the feature extractor in the standard supervised classification on the base session by replacing the EM with an additional auxiliary fully connected layer of dimension $d \times 60$. The additional fully connected layer is discarded and replaced by the EM after the pretraining. This pretraining step improved the overall accuracy on the base session by up to 15%. Next, the meta-learning is attained by drawing a new 60-way 5-shot problem in every iteration and updating the model based on ten queries per way. The model is trained for 70,000 iterations using a stochastic gradient descent (SGD) with momentum 0.9 and weight decay 5×10^{-4}. The learning rate is initially set to 0.01 and reduced by $10\times$ at iterations 30,000 and 60,000.

In Mode 2, the fully connected layer is retrained for $T = 10$ iterations at an update rate $\beta = 0.01$. In Mode 3, the prototype nudging is done for $U = 100$ iterations at an update rate $\gamma = 0.01$, and the fully connected layer is retrained for $T = 50$ iterations at an update rate $\beta = 0.01$.

Omniglot. For the Omniglot dataset, we use a feature extractor which involves 4 convolutional layers with 128

channels and 2×2 maxpooling at the end of the second and fourth layer, followed by a fully connected layer that resizes the flattened embedding to $d_f = 512$ before feeding to the retrainable fully connected layer ($d = 512$) that outputs the prototypes. During the meta-learning the model is trained for 70,000 iterations with an Adam optimizer with a learning rate of 10^{-4}. In Mode 2, the fully connected layer is retrained for $T = 20$ iterations with a learning rate $\beta = 10^{-4}$. In Mode 3, the prototype nudging is done for $U = 20$ iterations with an update rate $\gamma = 0.01$, followed by a similar setting for retraining the final fully connected layer.

5.3. Comparative Results

miniImageNet and CIFAR100. We compare our performance on the two natural image datasets with different state-of-the-art methods [5–7, 44, 48, 56], as shown in Table 1 and Table 2. Our method sets the new state-of-the-art on both datasets. Notably, even the simple prototype averaging (Mode 1) outperforms all other methods on both miniImageNet and CIFAR100. Note that Mode 1 does not involve any retraining or the use of auxiliary GAA memory. The EM can also be compressed as shown in Table A9.

In the other two modes, the prototype quasi-orthogonalization consistently improves the accuracy. The prototype bipolarization (Mode 2) is more effective for a lower number of classes (sessions $s \leq 3$ on miniImagenet and sessions $s \leq 8$ on CIFAR100), whereas the prototype nudging (Mode 3) outperforms all other methods for a large number of classes. Hence, the experimental results suggest that the classification of relatively simple problems (low number of ways) requires computationally cheap updates (e.g., Mode 1 or 2) for the best performance. On the other hand, harder problems (large number of ways) benefit from more sophisticated update mechanisms (i.e., Mode 3).

Omniglot. Table 3 compares the accuracy on the Omniglot dataset. As no prior works evaluated their methods on Omniglot in the FSCIL setting, we adapted ProtoNet [45] and CEC [56] as additional baselines. See Appendix A.4 for more details. C-FSCIL starts with 84.16% accuracy in the base session and ends with 81.56% in the last session using Mode 1. This small accuracy drop is further reduced by the other modes. In Mode 3, C-FSCIL achieves an accuracy of 87.21% in the base session and of 85.70% in session 10, outperforming both the ProtoNet and CEC baselines by a large margin of $\geq$16.99% and $\geq$8.30%, respectively.

For an additional comparison on Omniglot, we consider an alternative continual incremental learning setting developed by [2], that arranges 600 instead of the previous 423 classes in the novel sessions, but does not consider the evaluation on the base classes. As shown in Appendix A.4.3, we find that all modes of C-FSCIL performs consistently better than the best baseline, ANML [2].

5.4. Ablation study

We conduct extensive ablation experiments on the dimension d, the attention function, and the feature extractor. Here, we list the main findings of the ablation. Detailed results and discussions are available in Appendix A.2.

Dimension. We analyze the effect of reduced dimensions in Tables A1–A3. We find that C-FSCIL in Mode 3 allows to reduce the dimension below the number of classes ($d < |\tilde{\mathcal{C}}^{(S)}|$), e.g., $d = 64$ for miniImageNet and CIFAR100 or $d = 128$ for Omniglot, yielding marginal accuracy degradation while still outperforming all baseline methods.

Attention function. We compare the soft absolute [21] attention with the exponential attention, which is commonly used in softmax. Tables A4–A6 show the superiority of the soft absolute attention function, which particularly improves the accuracy when novel classes are encountered.

Feature extractor. Most baseline methods on miniImageNet and CIFAR100 use a ResNet-18 as feature extractor [5–7, 44, 48], requiring slightly fewer parameters compared to our ResNet-12 (12.4 M vs. 11. 2M). Tables A7–A8 show that a reduced ResNet-12 (8.0 M parameters) maintains a high accuracy (<1% drop) and still outperforms all baseline methods.

6. Conclusion and Outlook

We propose C-FSCIL for few-shot class-incremental learning in which the model is either built rapidly in one pass without gradient-based updates (Mode 1), or retrained with a small constant number of iterations (Modes 2 and 3). The C-FSCIL memory grows at most linearly with the number of encountered classes. The simple averaged prototypes in Mode 1 outperform all other methods for CIFAR100, miniImageNet, and Omniglot. In Modes 2 and 3, the optimization of the prototypes and the fixed-sized fully connected layer, through a maximum of 50 iterations, leads to higher accuracy (up to 4%) when the maximum number of classes is encountered.

Moreover, simply training with class prototypes having large inter-class separation provides robustness against adversarial perturbations, without requiring any adversarial training [43]. In a similar vein, C-FSCIL naturally pushes the meta-learned prototypes towards quasi-orthogonality. Furthermore, the precision of such robust prototypes can be reduced, as confirmed by the bipolarization in Mode 2, which makes them ideal for implementation on emerging hardware technologies exploiting non-volatile memory for in-memory computation [20, 21].

Acknowledgements

We acknowledge support from the IBM Research AI Hardware Center, and the Center for Computational Innovation at Rensselaer Polytechnic Institute for computational resources on the AiMOS Supercomputer.

References

[1] 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 (CVPR)*, June 2021. 3

[2] Shawn Beaulieu, Lapo Frati, Thomas Miconi, Joel Lehman, Kenneth O Stanley, Jeff Clune, and Nick Cheney. Learning to continually learn. In *European Conference on Artificial Intelligence (ECAI)*, 2020. 2, 8, 14, 19

[3] Zalán Borsos, Mojmir Mutny, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[4] Francisco M. Castro, Manuel J. Marin-Jimenez, Nicolas Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 2

[5] Kuilin Chen and Chi-Guhn Lee. Incremental few-shot learning via vector quantization in deep embedded space. In *International Conference on Learning Representations (ICLR)*, 2021. 2, 3, 7, 8, 12, 13, 17

[6] 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 (CVPR)*, June 2021. 2, 3, 7, 8, 12, 13, 17, 18

[7] Ali Cheraghian, Shafin Rahman, Sameera Ramasinghe, Pengfei Fang, Christian Simon, Lars Petersson, and Mehrtash Harandi. Synthesized feature based few-shot class-incremental learning on a mixture of subspaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 2, 3, 7, 8, 12, 13, 17, 18

[8] Yulai Cong, Miaoyun Zhao, Jianqiao Li, Sijia Wang, and Lawrence Carin. Gan memory with no forgetting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[9] E Paxon Frady, Spencer J Kent, Bruno A Olshausen, and Friedrich T Sommer. Resonator networks, 1: An efficient solution for factoring high-dimensional, distributed representations of data structures. *Neural computation*, 32(12):2311–2331, 2020. 2

[10] Ross W. Gayler. Vector symbolic architectures answer Jackendoff's challenges for cognitive neuroscience. In *Proceedings of the Joint International Conference on Cognitive Science. ICCS/ASCS*, pages 133–138, 2003. 2

[11] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 6, 13

[12] Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgeting in gradient-based neural networks. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2014. 1

[13] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural Turing Machines. *arXiv:1410.5401*, 2014. 2

[14] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid Computing Using a Neural Network with Dynamic External Memory. *Nature*, 538(7626):471–476, 2016. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 1

[16] Michael Hersche, Mustafa Zeqiri, Luca Benini, Abu Sebastian, and Abbas Rahimi. A neuro-vector-symbolic architecture for solving Raven's progressive matrices. *arXiv:2203.04571*, 2022. 2

[17] Elad Hoffer, Itay Hubara, and Daniel Soudry. Fix your classifier: the marginal value of training the last weight layer. In *International Conference on Learning Representations (ICLR)*, 2018. 2, 3

[18] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 3, 4

[19] P. Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1(2):139–159, 2009. 2, 4

[20] Geethan Karunaratne, Manuel Le Gallo, Giovanni Cherubini, Luca Benini, Abbas Rahimi, and Abu Sebastian. In-memory Hyperdimensional Computing. *Nature Electronics*, 3(6):327–337, 2020. 8

[21] Geethan Karunaratne, Manuel Schmuck, Manuel Le Gallo, Giovanni Cherubini, Luca Benini, Abu Sebastian, and Abbas Rahimi. Robust High-dimensional Memory-augmented Neural Networks. *Nature Communications*, 12(1):1–12, 2021. 2, 3, 4, 5, 8, 12

[22] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. 3

[23] Denis Kleyko, Geethan Karunaratne, Jan Rabaey, Abu Sebastian, and Abbas Rahimi. Generalized Key-Value Memory to Flexibly Adjust Redundancy in Memory-Augmented Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. 2, 3

[24] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1

[25] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 2009. 6, 12

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 1

[27] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350:1332–1338, 2015. 6, 12

[28] M. Ledoux. *The Concentration of Measure Phenomenon*. American Mathematical Society, 2001. 2

[29] Timothée Lesort, Thomas George, and Irina Rish. Continual learning in deep networks: an analysis of the last layer. *arXiv:2106.01834*, 2021. 3, 4

[30] Timothée Lesort, Andrei Stoian, and David Filliat. Regularization shortcomings for continual learning. *arXiv:1912.03049*, 2021. 3

[31] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[32] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 3

[33] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989. 1

[34] Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. Latent replay for real-time continual learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2020. 2, 3

[35] T. A. Plate. Holographic reduced representations. *IEEE Transactions on Neural Networks*, 6(3):623–641, 1995. 2, 3, 13, 14

[36] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[37] Zhongchao Qian, Tyler L. Hayes, Kushal Kafle, and Christopher Kanan. Do we need fully connected output layers in convolutional networks? *arXiv:2004.13587*, 2020. 3

[38] Leonardo Ravaglia, Manuele Rusci, Davide Nadalini, Alessandro Capotondi, Francesco Conti, and Luca Benini. A tinyml platform for on-device continual learning with quantized latent replays. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2021. 3

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 3

[40] Mengye Ren, Renjie Liao, Ethan Fetaya, and Richard S. Zemel. Incremental few-shot learning with attention attractor networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 6, 12

[42] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of The 33rd International Conference on Machine Learning*. PMLR, 2016. 2, 3

[43] Alex Serban, Erik Poll, and Joost Visser. Deep repulsive prototypes for adversarial robustness. *arXiv:2105.12427*, 2021. 8

[44] Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3, 7, 8, 12, 13, 17, 18

[45] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical Networks for Few-Shot Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 5, 7, 8, 12, 14, 16

[46] Sainbayar Sukhbaatar, arthur szlam, Jason Weston, and Rob Fergus. End-To-End Memory Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015. 2

[47] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1

[48] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 3, 6, 7, 8, 12, 13, 17, 18

[49] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: A good embedding is all you need? In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 7

[50] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 3, 5, 12

[51] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory Networks. In *International Conference on Learning Representations (ICLR)*, 2015. 2, 3

[52] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 3

[53] Zhirong Wu, Alexei A. Efros, and Stella X. Yu. Improving Generalization via Scalable Neighborhood Component Analysis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 3

[54] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised Feature Learning via Non-Parametric Instance Discrimination. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[55] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 6, 13

[56] Chi Zhang, Nan Song, Guosheng Lin, Yun Zheng, Pan Pan, and Yinghui Xu. Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 2, 3, 7, 8, 12, 13, 14, 16, 17, 18

[57] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3