Learning Equi-angular Representations for Online Continual Learning

Minhyuk Seo[1,†] Hyunseo Koh[1] Wonje Jeung[1] Minjae Lee[1] San Kim[1] Hankook Lee[2,3]
Sungjun Cho[2] Sungik Choi[2] Hyunwoo Kim[4,*] Jonghyun Choi[5,*]

[1]Yonsei Univ. [2]LG AI Research [3]Sungkyunkwan Univ. [4]Zhejiang Lab [5]Seoul National Univ.

{dbd0508, specific0924, reccos1020, nasmik419}@yonsei.ac.kr
khs8157@gmail.com, {sungik.choi,sungjun.cho}@lgresearch.ai
hankook.lee@skku.edu, hwkim@zhejianglab.com, jonghyunchoi@snu.ac.kr

Abstract

*Online continual learning suffers from an underfitted solution due to insufficient training for prompt model update (*e.g.*, single-epoch training). To address the challenge, we propose an efficient online continual learning method using the neural collapse phenomenon. In particular, we induce neural collapse to form a simplex equiangular tight frame (ETF) structure in the representation space so that the continuously learned model with a single epoch can better fit to the streamed data by proposing preparatory data training and residual correction in the representation space. With an extensive set of empirical validations using CIFAR-10/100, TinyImageNet, ImageNet-200, and ImageNet-1K, we show that our proposed method outperforms state-of-the-art methods by a noticeable margin in various online continual learning scenarios such as disjoint and Gaussian scheduled continuous (*i.e.*, boundary-free) data setups. Code is available at https://github.com/yonseivnl/earl.*

1. Introduction

A growing interest in continuous learning (CL) involves training the model using continuous data streams. Mostly, CL research has focused on the *offline* scenario that assumes the model can be trained in multiple epochs for the current task [8, 42, 54]. However, substantial storage and computational complexity are required to store all data to train a model for multiple epochs. Recently, there has been significant interest in *online* CL as a more realistic set-up with a lower computational overhead of allowing a single training pass through the data stream [1, 7, 27]. Learning a model in streamed data by a single training pass is challenging, since the temporal distribution at each intermediate time point is likely imbalanced, even if the overall distribution of a dataset is balanced. Imbalanced data distributions would cause several problems, such as bias towards the major classes [24, 61] and the hindrance to generalization [53].

Recently, *minority collapse* [14], the phenomenon in which the angles between classifier vectors for minor classes (*i.e.*, the classes that have a relatively small number of samples) become narrow, has been proposed as a fundamental issue in training with imbalanced data, making the classification of minor classes considerably more challenging. In contrast, for balanced data sets, it was proven that classifier vectors and the last layer activations for all classes converge into an optimal geometric structure, named the simplex *equiangular tight frame* (ETF) structure, where all pairwise angles between classes are equal and maximally widened when using cross entropy (CE) [23, 30, 52, 65] or mean squared error (MSE) [32, 41, 48, 64] loss. This phenomenon is called *neural collapse* [34].

Although neural collapse naturally occurs only in balanced training, several recent studies attempted to induce neural collapse in imbalanced training to address the minority collapse problem using a fixed ETF classifier [55, 62]. When employing a fixed ETF classifier, the output features are pulled toward the corresponding classifier vector during training, since the classifier is fixed. More recently, research has also been extended to induce neural collapse in offline CL using a fixed ETF classifier [56].

However, unlike offline CL, there are a number of challenges in inducing neural collapse in online CL, even with a fixed ETF classifier. The prerequisite for neural collapse is reaching the *terminal phase of training* (TPT) by sufficient training [34]. In offline CL, the model can reach the TPT phase for each task by multi-epoch training. In contrast, a single-pass training constraint often prevents the online CL from reaching TPT. As shown in Fig. 1, the offline CL quickly reaches TPT shortly after the arrival of novel task data, while online CL (vanilla ETF) does not.

†: Work done while interning at LG AI Research.
*: Indicates corresponding authors.

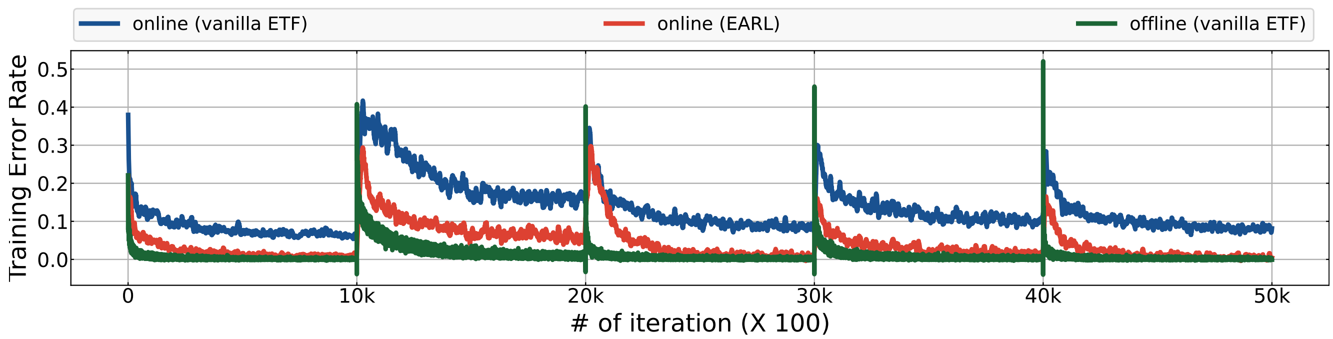


Figure 1. Comparison of training error rates between online CL and offline CL in the CIFAR-10 disjoint setup, where two novel classes are added every 10k samples. Vanilla ETF refers to a method where both preparatory data training and residual correction are removed from our proposed EARL.

Recently, the importance of anytime inference in online CL has been emphasized [6, 17, 27, 36], since a model should be available for inference not only at the end of a task, but also at any point during training to be practical for real-world applications. Hence, not only reaching TPT but also achieving faster convergence is necessary when using neural collapse in online CL.

However, we observe that the phenomenon in which the features of the new class become biased toward the features of the existing classes [5] hinders fast convergence of the last-layer features into the ETF structure, which we call the 'bias problem.' When features of old and new classes overlap (*i.e.*, biased) and are trained with the same objective, well-clustered features of old classes disperse (or perturb) [5], leading to destruction of the ETF structure formed by features of the old classes.

Contributions. To address the bias problem, we train the model to distinguish out-of-distribution (OOD) samples from existing classes' samples. Specifically, we synthesize *preparatory data* that play the role of OOD data, by applying negative data augmentation [25, 47, 50] to samples from existing classes, refer to Sec. 4.2 for details. Since samples from new classes are OOD in the perspective of existing classes, training with preparatory data encourages the representation of new classes to be distinguished from existing classes in advance, thereby mitigating the bias problem. This promotes fast and stable convergence into the ETF structure.

Despite these efforts, however, the continuous stream of new samples cause ongoing data distribution shifts, which prevent the model from reaching the TPT and fully converging to the ETF structure. To address this, we propose to store the residuals between the target ETF classifier and the features during training. During inference, we correct the inference output using the stored residual to compensate for insufficient convergence in the training.

By accelerating convergence by preparatory data training and additional correction using residual, we noticeably improve anytime inference performance than state of the arts. We name our method **Equi-Angular Representation Learning** (**EARL**). We empirically demonstrate the effectiveness of our framework on CIFAR-10, CIFAR-100, TinyImageNet, Imagenet-200, and ImageNet-1K. In particular, our framework outperforms various CL methods by a significant margin (+4.0% gain of A_{auc} in ImageNet-1K).

We summarize our contributions as follows:

- Proposing to induce neural collapse for online CL.
- Proposing 'preparatory data training' to address the 'bias problem' that the new classes are biased toward the existing classes, promoting faster induction of neural collapse.
- Proposing 'residual correction' scheme to compensate for not fully reaching neural collapse at inference to further improve anytime inference accuracy.

2. Related work

Continual learning methods. Various continual learning methods are being researched to prevent forgetting past tasks, broadly categorized into replay, parameter isolation, and regularization. Replay methods involve storing a small portion of data from previous tasks in episodic memory [1, 2, 20, 27, 60] or storing a generative model trained on data from previous tasks [39, 46]. By replaying the samples stored in episodic memory or generated from the stored generative model, the model prevents forgetting past tasks during subsequent learning of novel tasks. Furthermore, Boschini et al. [3], Buzzega et al. [4], Li and Hoiem [29], Wu et al. [54] used replay samples to distill information about past tasks.

Regularization methods [26, 28] apply a penalty to the change of important model parameters during the process of learning new tasks, allowing the model to retain information about previous tasks. Parameter isolation methods [10, 44, 63] expand the network by allocating specific layers for each task. This enables the network to store information about individual tasks and preserves their knowledge without forgetting.

Neural collapse. Neural collapse is a phenomenon in which the activations of the last layer and the classifier vectors form a simplex *equiangular tight frame* (ETF) structure at the terminal phase of training (TPT) in a balanced dataset. [34]. Neural collapse has been demonstrated as the global optimum of balanced training using cross entropy (CE) loss [23, 30, 52, 65] and MSE [32, 41, 48, 64] loss functions, within a simplified model focused solely on last-layer optimization. Inducing neural collapse in imbalanced datasets poses challenges due to minority collapse [14] where minor classes are not well distinguished.

However, using a fixed ETF classifier, it is empirically and theoretically shown that neural collapse is induced even in imbalanced datasets [55]. Continual learning also needs to address imbalanced data since there is an imbalance in the data between novel classes and existing classes. Therefore, in offline CL, NC-FSCIL [38, 56] used a fixed ETF classifier to induce neural collapse. Meanwhile, online CL often fails to induce neural collapse compared to offline CL, *e.g.*, FSCIL, since it lacks sufficient training in multiple epochs, causing failure to reach TPT.

Please refer to the supplementary material for more comprehensive literature reviews including topics related to anytime inference.

3. Preliminaries

We here describe the problem statement for the online continual learning (Sec. 3.1), neural collapse (Sec. 3.2), and the equiangular tight frame classifier (Sec. 3.3).

3.1. Problem Statement of Online CL

Continual learning (CL) aims to learn from a stream of data, rather than a fixed dataset as in standard learning. Formally, given a sequence of tasks $\mathcal{T} = (T_1, T_2, \ldots)$ where each task T_i is a training dataset $D_i = \{(x_1^{(i)}, y_1^{(i)}), (x_2^{(i)}, y_2^{(i)}), \ldots\}$, an offline CL algorithm A_{CL} updates the model parameter θ based on the current task D_k, *i.e.*, $\theta_k = A_{\text{CL}}(\theta_{k-1}, D_k)$, starting from the initial parameter θ_0.

To avoid the forgetting issue in CL, many CL setups allow the use of episodic memory M_k, which is a limited-size subset of training data from previous tasks, *i.e.*, $(\theta_k, M_k) = A_{\text{CL}}(\theta_{k-1}, M_{k-1}, D_k)$. The objective is to minimize the error of θ_k in all observed tasks $\{T_i\}_{i=1}^k$.

Unlike offline CL where all training samples in D_k are given as input, in online CL, the input is provided as a stream of samples $(x_1^{(k)}, y_1^{(k)}), (x_2^{(k)}, y_2^{(k)}), \cdots$. Therefore, an online CL algorithm A_{OCL} is defined as:

$$(\theta_{k,t}, M_{k,t}) = A_{\text{OCL}}\left(\theta_{k,t-1}, M_{k,t-1}, (x_t^{(k)}, y_t^{(k)})\right), \quad (1)$$

with the same objective of minimizing the error of $\theta_{k,t}$ on $\{T_i\}_{i=1}^k$. Since the algorithm does not have access to previous samples $\{(x_s^{(k)}, y_s^{(k)}), s < t\}$ at time t, multi-epoch training with D_k is unavailable in online CL.

3.2. Neural Collapse

Neural collapse [34] is a phenomenon of the penultimate features after the convergence of training on a balanced dataset. When neural collapse (NC) occurs, the collection of K classifier vectors $\mathbf{W}_{\text{ETF}} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_K] \in \mathbb{R}^{d\times K}$ forms a simplex equiangular tight frame (ETF), which satisfies:

$$\mathbf{w}_i^T \mathbf{w}_j = \begin{cases} 1, & i = j \\ -\frac{1}{K-1}, & i \neq j \end{cases}, \quad \forall i, j \in [1, ..., K], \quad (2)$$

where $K - 1 \leq d$, and the penultimate feature of a training sample collapses into an ETF vector $\mathbf{w}_i$.

3.3. Equiangular Tight Frame (ETF) Classifier

Inspired by neural collapse as described in Sec. 3.2, a fixed ETF classifier has been utilized for inducing neural collapse in imbalanced datasets [55, 56, 65]. Here, the classifier is initialized by the ETF structure $\mathbf{W}_{\text{ETF}}$ at the beginning of training and fixed during training to induce the penultimate feature $f(x)$ to converge to the ideal balanced scenario. For training $f(x)$, it is only required to attract $f(x)$ to the corresponding classifier vector for convergence, since the classifier is fixed during training. Therefore, following Yang et al. [55], we use the *dot regression* (DR) loss as a training objective, as it shows to outperform cross entropy (CE) loss when using a fixed ETF classifier in imbalanced datasets [55].[1] The DR loss can be written as follows:

$$\mathcal{L}_{\text{DR}}(\hat{f}(x), y; \mathbf{W}_{\text{ETF}}) = \frac{1}{2}\left(\mathbf{w}_y \hat{f}(x) - 1\right)^2, \quad (3)$$

where $\hat{f}(x) = f(x)/\|f(x)\|_2$ is the L_2 normalized feature of the model f, y is the label of the input x, and $\mathbf{w}_y$ is a classifier vector in $\mathbf{W}_{\text{ETF}}$ for the label y.

4. Approach

Despite the success of the fixed ETF classifier in both imbalanced training [55, 62] and offline CL [56], the ETF classifier has not yet been explored for online CL due to the necessity of sufficient training for neural collapse. To be specific, streamed data are trained only once in online CL, which makes it harder to induce neural collapse than in offline CL which supports multi-epoch training.

To learn a better converged model without multi-epoch training for online CL, we propose two novel methods, each for the training phase and the inference phase, respectively. In the training phase, we accelerate convergence by proposing *preparatory data training* (Sec. 4.2). In the inference

[1] If the training objective includes a contrastive term between different classes like cross entropy, it could cause incorrect gradients [55].

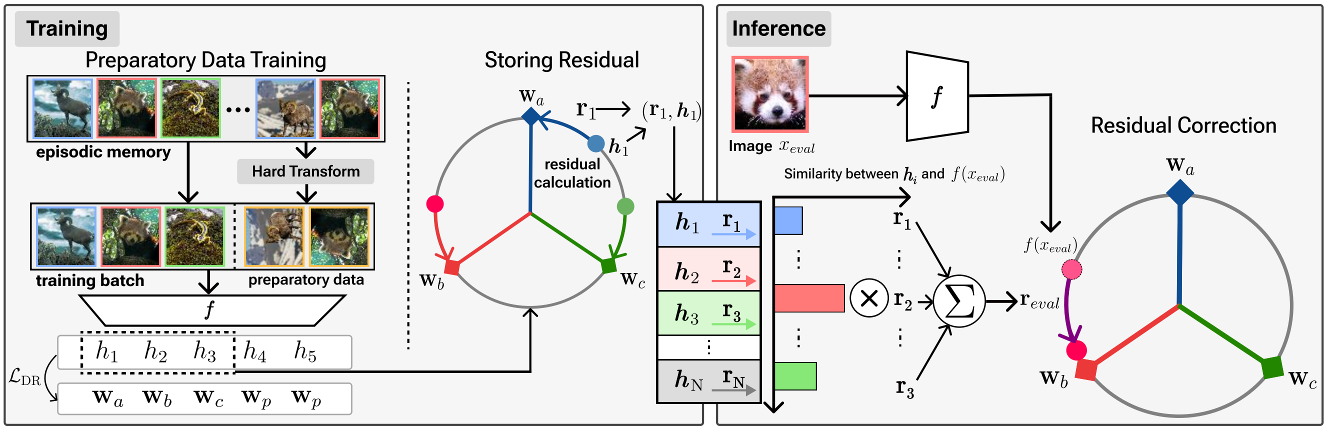


Figure 2. Overview of EARL. $\mathbf{w}_i$ denotes the ETF classifier vector for class i. h_a denotes the output of the model. The colors of the data denote the class to which the data belong. The arrow $\mathbf{r}_i$ denotes the residual between the last layer activation h_i and the classifier vector $\mathbf{w}_i$ for class i. During training, both memory and preparatory data are used for replaying, and the residuals between h_i and $\mathbf{w}_i$ are stored in feature-residual memory. During inference, using the similarity between $f(x_{\text{eval}})$ and h_i in feature-residual memory, $\mathbf{r}_{\text{eval}}$ is obtained by a weighted sum of $\mathbf{r}_i$'s. Finally, by adding $\mathbf{r}_{\text{eval}}$, $f(x_{\text{eval}})$ is corrected. The purple arrow indicates 'residual correction' (Sec. 4.3).

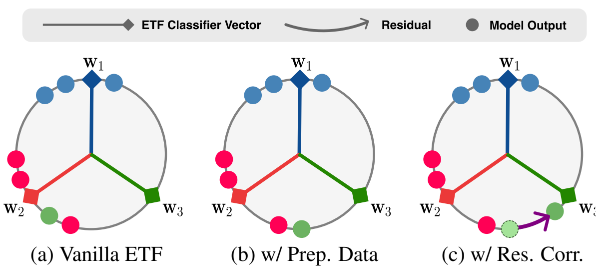


Figure 3. Illustrative effects for each component of EARL. (a) In online CL, features of novel classes are biased towards the features of the previous class. (b) By training with preparatory data (**w/ Prep. Data**, Sec. 4.2), we address the bias problem. (c) In inference, for features that do not fully converge to an ETF classifier, we add residuals (**w/ Res. Corr.**, Sec. 4.3) to features that have not yet reached the corresponding classifier vectors, making features aligned with them. Purple arrow: the 'residual correction', Colors: classes.

phase, we propose to correct the remaining discrepancy between the classifiers and the features using *residual correction* (Sec. 4.3). We illustrate an overview of EARL in Fig. 2 and the effect of EARL in Fig. 3.

4.1. Inducing Neural Collapse for Online CL

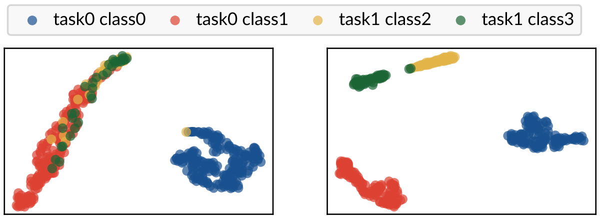
For online CL, we first try to induce neural collapse with a fixed ETF classifier: $\mathbf{W}_{\text{ETF}} \in R^{d\times K}$ where d is the dimension of the embedding space and K is the number of ETF vectors. While prior works [55, 56] use prior knowledge of the total number of classes that will be encountered for setting K, it is unrealistic to know the knowledge (*i.e.*, the exact number of classes) as it evolves continuously over time under realistic CL scenarios.

To address the challenge, we propose using the maximum possible number of classifier vectors for K. Based on the simplex ETF property that the maximum number of ETF vectors is $d+1$ [16, 55], we define our ETF classifier $\mathbf{W}_{\text{ETF}} \in \mathbb{R}^{d\times(d+1)}$ by letting $K := d+1$.

4.2. Preparatory data at training

Since novel classes continuously arrive in CL, both the data from previous tasks (x_{old}) and the current task (x_{new}) coexist. While $\hat{f}(x_{\text{old}})$ are placed closer to their corresponding ETF classifier vectors w_{new} by training, $\hat{f}(x_{\text{new}})$ are biased toward the cluster of $\hat{f}(x_{\text{old}})$, as we can see in Fig. 4-(a), where $\hat{f}(x_{\text{new}})$ and $\hat{f}(x_{\text{old}})$ are the outputs of the model for inputs x_{new} and x_{old}, respectively. We call this '*bias problem*.' When $\hat{f}(x_{\text{new}})$ and $\hat{f}(x_{\text{old}})$ overlap due to bias and are optimized with the same objective function, training on the new class interferes with the representations of the old classes, dispersing the well-clustered $\hat{f}(x_{\text{old}})$ [5]. It destroys the ETF structure formed by $\hat{f}(x_{\text{old}})$, which hinders convergence towards neural collapse. Although the bias problem exists even without a fixed ETF classifier [5, 12, 54], it poses a greater problem when using a fixed ETF classifier, since the dispersed feature of old classes has to be restored to the corresponding fixed classifier vector. In contrast, when using a learnable classifier, the dispersed representation of the old classes does not have to be restored to the original position, since the learnable classifier vector will be re-optimized from the dispersed feature. However, the angles between the classifier vector may become narrow, *i.e.*, minority collapse occurs.

To accelerate convergence in the ETF structure, we propose to prevent novel classes from being biased towards existing classes when introduced, mitigating the bias problem. Specifically, we train the model to avoid making predictions in favor of the existing classes for images that do not be-



(a) 100 iterations after Task 1 introduced

(b) 10,000 iterations after Task 1 introduced

Figure 4. t-SNE visualization of 'bias-problem' in data distributions (class 0 to 3). (a) Only after 100 iterations of training after task 1 appears, learning is likely insufficient, and we can see that the features of new classes (class 2, 3) are biased towards the feature cluster of the existing class (*i.e.*, class 1). (b) With more training iterations (10,000 iter), the features are well clustered by class.

long to them. To this end, we propose to use preparatory data x_p that is different from existing classes, obtained by transforming the samples in the episodic memory. By training with the preparatory data so that their representations differ from existing classes, we prevent biased predictions towards existing classes when a new class arrives.

For preparatory data to have a different representation from existing classes when trained, their image semantics should be distinguishable from existing classes, provided that they contain enough semantic information to be considered as an image (*i.e.*, not noise). There are various approaches to synthesize images with modified semantics, such as using generative models [11, 19, 58] or negative transformations [47, 50]. We use negative transformations, as generative models are expensive in computation and storage [13], which is undesirable for CL scenarios with limited memory and computations. On the contrary, negative transformations are relatively efficient in computation and storage [33] and is reported to create images with semantics different from the original image [50]. We empirically observed that data generated through negative transformations have a distinct semantic class from the original class while preserving its semantic information as an image (*i.e.*, not as random noise), and we summarize the results in the supplementary material.

Formally, for a set of existing classes Y and the set of possible transformations G, we randomly select $y \in Y$ and $g \in G$, and randomly retrieve a sample of class y from memory and apply the transformation g to obtain *preparatory data* x_p. We assign labels of the unseen classes to the preparatory data by a mapping function $m : Y \times G \rightarrow Y'$ where Y' denotes a set of unseen classes, *i.e.*, $Y' = \{y | y \notin Y,\ 1 \leq y \leq K\}$ and K is the total number of classifier vectors in $\mathbf{W}_{\text{ETF}} \in \mathbb{R}^{d \times K}$. Thus, preparatory data x_p from class y and transformation g is pulled towards the classifier vector $\mathbf{w}_p$, where $p = m(y, g)$. When a new class y_{new} is added to Y, we update m by randomly assigning a new mapping $m(y_{\text{new}}, g)$ for y_{new} and $g \in G$.

We compose the transformation set G as a negative transformation that can modify semantic information (*i.e.*, change the label), used in the self-supervised literature [15, 18] and out-of-distribution (OOD) detection [25, 47, 50]. Specifically, we use rotation by 90, 180, and 270 degrees [15, 18] as our negative transformation. Since the vertical information of the image is more important than the horizontal information [22, 49], *e.g.*, images of a car facing left or right are common, but a car flipped upside down is very rare compared to a car standing right, rotation by a large angle causes loss of semantic information of images [50]. Thus, we can use rotated data as semantically different images from images of existing classes, *i.e.*, suitable for preparatory data. A more detailed analysis of various negative transformations is provided in the Supple.

We jointly train the model using real data (*i.e.*, samples of existing classes) and preparatory data. Formally, we write the objective for the model parameter θ as:

$$\arg\min_{\theta} \Big[\mathbb{E}_{(x,y)\sim\mathcal{M}} \mathcal{L}_{\text{DR}} \left(\hat{f}_\theta(x), y; \mathbf{W}_{\text{ETF}} \right) + \lambda\, \mathbb{E}_{(x',y')\sim\mathcal{M}, g\sim G} \mathcal{L}_{\text{DR}} \left(\hat{f}_\theta\left(g(x')\right), m(y', g); \mathbf{W}_{\text{ETF}} \right) \Big], \quad (4)$$

where $\mathcal{M}$ denotes episodic memory, (x, y) and (x', y') denote the samples in $\mathcal{M}$, G denotes the set of possible transformations, g denotes a transformation in G, m denotes the mapping function, and λ is a hyperparameter for balancing real data and preparatory data.

4.3. Residual correction at inference

Despite preparatory data training that accelerates convergence towards ETF, in online CL, new samples in a data stream hinder models from reaching the Terminal Phase of Training (TPT) and fully converging to the ETF during single-epoch training. When the output of the model $f(x)$ does not fully converge to the ETF classifier, the model would not perform well at all times, leading to poor anytime inference accuracy (A_{AUC}).

To address this issue, we want to correct the residual between $f(x)$ and the corresponding classifier vector $\mathbf{w}_y$ during inference, where y is label of input x.

However, the discrepancy between the prediction on test data x_{eval} and the ground-truth ETF classifier vector is not available in the inference phase. In a similar situation, ResMem [57] stores the residual obtained from training data and uses them during inference, but they assume two-stage learning algorithms, *i.e.*, standard training and inference rather than incremental learning and anytime inference.

Inspired by ResMem [57], which uses residuals from the training in the inference stage, we propose to use the residual obtained from the training process to compensate for the

Method	CIFAR-10				CIFAR-100			
	Disjoint		Gaussian-Scheduled		Disjoint		Gaussian-Scheduled	
	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑
EWC (Kirkpatrick *et al.*, 2017)	75.25±0.78	60.80±2.20	59.62±0.31	64.24±1.97	52.08±0.83	41.55±0.85	38.22±0.50	42.52±0.58
ER (Rolnick et al., 2019)	75.94±0.86	63.56±1.32	60.13±0.56	64.81±2.70	52.95±1.25	42.82±0.05	41.12±0.56	42.74±1.09
ER-MIR (Aljundi *et al.*, 2019)	75.89±1.02	61.93±0.93	60.39±0.48	61.64±3.86	52.93±1.44	42.47±0.13	41.19±0.63	42.93±1.18
REMIND (Hayes *et al.*, 2020)	69.55±0.91	53.34±1.01	58.01±0.72	59.27±1.86	40.87±0.76	36.17±1.83	23.40±2.25	28.78±1.71
DER++ (Buzzega *et al.*, 2020)	74.78±0.72	59.20±0.95	59.44±0.28	66.11±1.80	38.16±1.57	38.55±2.11	29.38±2.58	38.20±3.13
SCR (Mai *et al.*, 2021)	75.61±0.93	56.52±0.52	60.62±0.43	58.41±2.39	41.84±0.74	36.00±0.83	31.33±0.41	32.11±0.39
ODDL (Ye *et al.*, 2022)	75.03±1.00	61.61±3.55	65.46±0.46	66.19±2.08	40.26±0.50	41.88±4.52	38.82±0.49	41.35±1.08
MEMO (Zhou *et al.*, 2023)	73.21±0.49	62.47±3.38	59.26±0.90	62.01±1.17	40.60±1.11	39.87±0.46	23.41±1.63	32.74±2.11
X-DER (Boschini *et al.*, 2023)	77.59±0.62	65.40±4.79	58.28±1.17	59.76±4.28	52.80±1.61	43.73±0.86	41.94±0.57	46.00±1.04
EARL (Ours)	**78.31±0.72**	**65.71±2.26**	**69.52±0.19**	**70.41±1.97**	**57.12±1.22**	**44.40±0.68**	**47.89±0.61**	**46.09±0.26**

Method	TinyImageNet				ImageNet-200			
	Disjoint		Gaussian-Scheduled		Disjoint		Gaussian-Scheduled	
	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑
EWC (Kirkpatrick *et al.*, 2017)	37.95±0.93	27.50±0.80	25.29±0.81	26.06±0.52	41.84±0.64	31.57±0.80	30.71±0.27	33.33±0.98
ER (Rolnick *et al.*, 2019)	37.43±1.05	27.47±0.63	26.37±0.89	25.79±0.44	41.51±0.76	30.87±0.72	32.39±0.36	33.09±0.37
ER-MIR (Aljundi *et al.*, 2019)	37.81±1.06	26.72±0.86	26.22±0.69	25.11±1.04	38.28±0.38	33.12±0.73	32.17±0.44	33.85±0.93
REMIND (Hayes *et al.*, 2020)	28.37±0.13	27.68±0.45	10.19±0.60	14.90±1.49	39.25±0.93	31.98±0.84	30.23±0.62	33.98±0.09
DER++ (Buzzega *et al.*, 2020)	39.38±0.60	29.36±0.81	27.23±1.94	31.53±0.80	43.50±0.31	34.56±0.50	35.22±0.26	38.38±0.97
SCR (Mai *et al.*, 2021)	34.65±1.08	22.18±0.32	25.86±0.94	22.54±0.59	41.90±0.40	28.92±0.40	33.24±0.32	30.98±0.28
MEMO (Zhou *et al.*, 2023)	27.36±0.61	27.57±0.52	10.82±1.23	18.03±1.36	41.55±0.23	34.19±1.47	32.54±0.39	36.11±1.06
X-DER (Boschini *et al.*, 2023)	35.15±2.12	26.67±0.52	29.71±0.86	28.10±0.50	43.41±0.47	34.14±0.98	36.31±0.17	38.61±0.55
EARL (Ours)	**41.77±1.26**	**29.65±0.20**	**35.08±0.70**	**32.49±1.21**	**44.88±0.29**	**34.79±0.55**	**39.14±0.47**	**38.83±0.35**

Table 1. Comparison of online CL methods on Disjoint and Gaussian Scheduled Setup for CIFAR-10, CIFAR-100, TinyImageNet and ImageNet-200.

remaining discrepancy between the classifiers and the features at inference.

To select which of the stored residuals to use during inference, we not only store the residual, but also $f(x)$ to choose the stored residual closest to $f(x_{\text{infer}})$. Therefore, we retain 'feature-residual' pairs in a 'feature-residual memory' $M = \{(\hat{h}_i, \mathbf{r}_i)\}_{i=1}^N$, where $\hat{h}_i = \hat{f}(x_i)$, $\mathbf{r}_i = \mathbf{w}_{y_i} - \hat{f}(x_i)$, where N is the size of feature-residual memory, and $\mathbf{w}_{y_i}$ is the classifier vector for class y_i.

During inference, we select the k nearest neighbor $\hat{h}$'s, *i.e.*, $\{\hat{h}_{n_1}, \hat{h}_{n_1}, \ldots, \hat{h}_{n_k}\}$ from $\{\hat{h}_i\}_{i=1}^N$, since using only the nearest residual for correction may lead to incorrect inference predictions if a wrong residual is selected from a different class. Finally, following ResMem [57], we calculate the residual correction term $\mathbf{r}_{\text{eval}}$ by a weighted average of the corresponding residuals $\{\mathbf{r}_{n_1}, \mathbf{r}_{n_2}, \ldots, \mathbf{r}_{n_k}\}$, with weights $\{s_1, s_2, \ldots, s_k\}$ that are inversely proportional to the distance from $\hat{f}(x_{\text{eval}})$, as:

$$\mathbf{r}_{\text{eval}} = \sum_{i=1}^{k} s_i \mathbf{r}_i, \quad s_i = \frac{e^{-(\hat{f}(x_{\text{eval}}) - \hat{h}_i)/\tau}}{\sum_{j=1}^{k} e^{-(\hat{f}(x_{\text{eval}}) - \hat{h}_j)/\tau}}, \tag{5}$$

where τ is a temperature hyperparameter. We add the residual-correcting term $\mathbf{r}_{\text{eval}}$ to the model output $\hat{f}(x_{\text{eval}})$ to obtain the corrected output $\hat{f}(x_{\text{eval}})_{\text{corrected}}$ as:

$$\hat{f}(x_{\text{eval}})_{\text{corrected}} = \hat{f}(x_{\text{eval}}) + \mathbf{r}_{\text{eval}}. \tag{6}$$

5. Experiments

5.1. Experimental setup

We perform experiments on four datasets: CIFAR-10, CIFAR-100, TinyImageNet, ImageNet-200, and ImageNet-1K. For all datasets, our experiments are conducted on both a disjoint setup [35] and a Gaussian scheduled setup [45, 51]. We report the average and standard deviation results in three different seeds, except ImageNet-1k due to computational resource constraints [2, 27].

To evaluate anytime inference performance, we use the area under the curve accuracy(A_{auc}) [6, 27], which measures the area under the accuracy curve. We also use last accuracy(A_{last}) which measures accuracy at the end of training. For detailed information about the experimental setup, refer to the Supplementary.

Baselines. We compare EARL with various CL methods in different categories: regularization (EWC [26]), network expansion (MEMO [63]), replay (ER [43], ER-MIR [1], REMIND [20] , SCR [31], ODDL [3]), and distillation

Method	ImageNet-1K			
	Disjoint		Gaussian-Scheduled	
	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑
EWC (Kirkpatrick *et al.*, 2017)	30.17	18.78	17.09	20.15
ER (Rolnick *et al.*, 2019)	30.18	18.96	17.17	18.39
ER-MIR (Aljundi *et al.*, 2019)	31.68	19.87	19.37	16.30
REMIND (Hayes *et al.*, 2020)	32.47	**24.59**	17.42	19.79
DER++ (Buzzega *et al.*, 2020)	31.08	18.98	23.73	23.14
SCR (Mai *et al.*, 2021)	26.72	13.60	19.82	15.84
MEMO (Zhou *et al.*, 2023)	30.45	19.08	14.06	20.12
X-DER (Boschini *et al.*, 2023)	31.67	18.18	25.18	12.51
EARL (Ours)	**34.33**	23.19	**30.53**	**29.48**

Table 2. Comparison of online CL methods on Disjoint and Gaussian Scheduled Setup for ImageNet-1K.

(DER++ [4], X-DER [3]). For more details on the implementation of these methods and a comparison with NC-FSCIL [56], which attempts to induce neural collapse in the context of few-shot class incremental learning, please refer to the supplementary material.

Implementation details. We use three components to architect our model: a backbone network $g(\cdot)$, a projection MLP $p(\cdot)$, and a fixed ETF classifier $\mathbf{W}_{\text{ETF}}$ (*i.e.*, our model f can be defined as $f(x) = p \circ g(x)$). For the projection layer, we attach an MLP projection layer p_{θ_p} to the output of the backbone network g_{θ_g}, where θ_g and θ_p denote the parameters of the backbone network and the projection layer, respectively (*i.e.*, the model f can be defined as $f(x) = p \circ g(x)$), following [9, 37, 56]. For all methods, we use ResNet-18 [21] as the backbone network.

Following [27, 59], we employ memory-only training, where a random batch is selected from memory at each iteration. Furthermore, for episodic memory sampling, EARL uses the Greedy Balanced Sampling strategy [40]. We describe the details about hyperparameters and the pseudocode of EARL in Supplementary for the sake of space.

5.2. Results

We first compare the accuracy of online continual learning methods, including EARL, and summarize the results in Table 1. As shown in the table, EARL outperforms other baselines on all benchmarks, both in disjoint and Gaussian-scheduled setups, except A_{last} in ImageNet-1K disjoint setup. In particular, high A_{AUC} suggests that EARL outperforms other methods for all the time that the data stream is provided to the model, which implies that it can be used for inference at anytime.

Furthermore, EARL, SCR, ER, MIR, DER, and ODDL do not use task boundary information during training, *i.e.*, *task-free*, in contrast to EWC, X-DER, MEMO, and REMIND which use task-boundary information, *i.e.*, *task-aware*. Nevertheless, EARL outperforms these methods even in the disjoint setup where utilizing task boundary information is advantageous due to abrupt distribution shifts at the task boundaries, except MEMO in ImageNet-1K disjoint. Since MEMO not only uses the task boundary information but also expands the network per task, this extra advantage of using a larger model is emphasized in large-scale datasets such as ImageNet which require a large learning capacity of the model. Please refer to the supplementary material for details of the computational budget comparison among the baseline methods.

Memory budget. We consider not only the size of episodic memory, but also the size of the model, logits, and other components, following MEMO [63]. In the case of the memory budget for CIFAR-10 and the architecture of ResNet-18 (size of 42.6MB), EARL incurs an additional memory cost of 0.2MB to store feature-residual pairs. EWC requires an additional cost of 85.2MB due to the need to store Fisher Information per parameter and the previous model. MEMO, which expands the network per task, incurs an additional cost of 32MB per task. DER also requires an additional cost of 0.1MB to store logits during distillation.

5.3. Ablation study

We conduct an ablation study on the two components of EARL, preparatory data training and residual correction, and summarize the results in Table 3. Although both components contribute to performance improvement, preparatory data training shows a larger gain in performance. Training with preparatory data improves the baseline by 2.7% $\sim$ 3.3% in A_{auc} and 2.1% $\sim$ 4.4% in A_{last}. When combined with residual correction, we observe further gains across all metrics in both disjoint and Gaussian-scheduled setups.

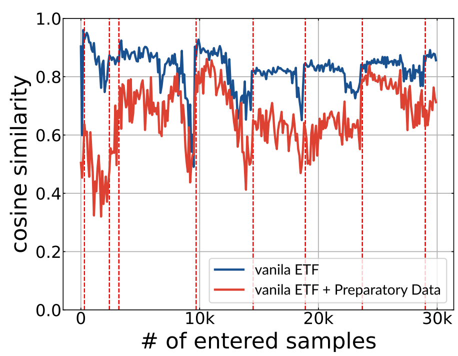
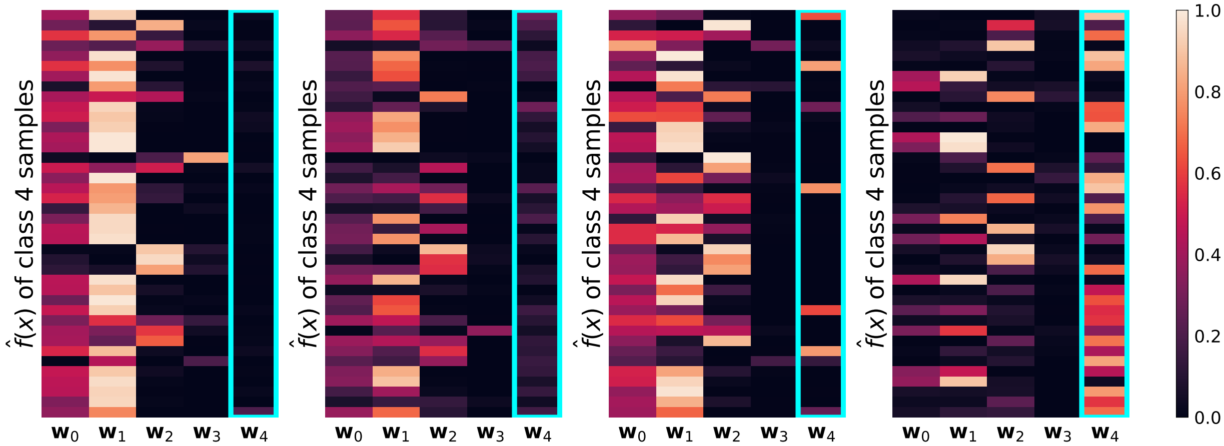


Figure 5. Average similarity between features of the most recently added class's samples and the closest classifier vectors of the old classes (CIFAR-10, Gaussian Scheduled). Baseline is a vanilla ETF model trained only using episodic memory.

For further analysis, Fig. 6 visualizes the cosine similarity between the output features of 50 randomly selected test set samples of novel class '4' and the classifier vectors $\mathbf{w}_i$. In baseline (a), the features of the class 4 are strongly biased towards the classifier vectors of the old classes (*i.e.*, $\mathbf{w}_0$, $\mathbf{w}_1$, $\mathbf{w}_2$, $\mathbf{w}_3$), rather than the correct classifier, $\mathbf{w}_4$. (c)

METHOD	CIFAR-10				CIFAR-100			
	Disjoint		Gaussian-Scheduled		Disjoint		Gaussian-Scheduled	
	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑	A_{AUC} ↑	A_{last} ↑
EARL (Ours)	**78.61±0.72**	**66.01±2.26**	**69.62±0.19**	**70.91±1.97**	**57.42±1.24**	**44.60±0.65**	**48.19±0.61**	**46.10±0.26**
(-) RC	77.78±0.52	65.52±1.63	68.37±0.10	70.17±1.08	56.28±1.40	44.29±1.04	47.05±0.71	46.07±0.52
(-) RC & PDT	74.77±0.77	61.10±3.12	65.60±0.25	67.39±0.78	52.91±1.05	41.08±0.60	44.34±0.55	44.45±0.48

Table 3. Ablation Study. RC and PDT refer to preparatory data training (Sec. 4.2) and the residual correction (Sec. 4.3), respectively.



(a) baseline (b) baseline + PDT (c) baseline + RC (d) baseline + PDT + RC

Figure 6. Cosine similarity between features $\hat{f}(x)$ for class 4 and the ETF classifier vectors $\mathbf{w}_i$ at the 50^{th} iteration after the introduction of class 4 in the Gaussian Scheduled CIFAR-10 setup. As we can see in the cyan highlighted box, EARL promotes the convergence of $\hat{f}(x)$ for class 4 toward the ground truth classifier vector $\mathbf{w}_4$.

When residual correction is used, some samples show high similarity with $\mathbf{w}_4$ compared to the baseline.

However, since incorrect residuals can be added due to the bias problem, more samples have high similarity with $\mathbf{w}_2$ than the baseline (*i.e.*, wrong residuals are added). (b) When using preparatory data training, the bias toward $\mathbf{w}_0$ and $\mathbf{w}_1$ classes significantly decreases compared to the baseline. Fig. 5 also shows the effect of preparatory data, which reduces the similarity between the novel class features and existing classes. (d) Using both residual correction and preparatory data training shows a remarkable alignment with the ground truth classifier $\mathbf{w}_4$. A more detailed analysis of the ablation results is provided in the Supple.

6. Conclusion

To better learn online data in a continuous data stream without multiple epoch training, we propose to induce neural collapse, which aligns last layer activations to the corresponding classifier vectors in the representation space. Unlike in offline CL, it is challenging to induce neural collapse in online CL due to insufficient training epochs and continuously streamed new data. We first observe that the bias of the new class towards existing classes slows the convergence of features toward neural collapse.

To mitigate the bias, we propose synthesizing preparatory data for unseen classes by transforming the samples of existing classes. Using the preparatory data for training, we accelerate neural collapse in an online CL scenario. Additionally, we propose residual correction to resolve the remaining discrepancy toward neural collapse at inference, which arises due to the continuous stream of new data. In our empirical evaluations, the proposed methods outperform state-of-the-art online CL methods in various datasets and setups, especially with high performance on anytime inference.

Limitations and Future Work. Since our work uses ETF structure, it has an inherent limitation that the number of possible classifier vectors in the ETF classifier is limited by the dimension of the embedding space. Considering life-long learning, where the number of new classes goes to infinity, it is interesting to explore the idea of dynamically expanding the ETF structure so that the model can continually learn the ever-increasing number of concepts in the real world.

Acknowledgement. This work was partly supported by the NRF grant (No.2022R1A2C4002300, 20%) and IITP grants (No.2022-0-00077 (10%), No.2022-0-00113 (25%), No.2022-0-00959 (10%), No.2022-0-00871 (10%), No.2020-0-01361 (5%, Yonsei AI), No.2021-0-01343 (5%, SNU AI), No.2021-0-02068 (5%, AI Innov. Hub), No.2022-0-00951 (5%)) funded by the Korea government (MSIT) and CARAI grant funded by DAPA and ADD (UD230017TD) 5%. This work was done while J. Choi was with Yonsei University and partially while H. Lee, M. Seo and H. Kim was with LG AI Research.

References

[1] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *CVPR*, 2019. 1, 2, 6

[2] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, 2021. 2, 6

[3] Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *IEEE TPAMI*, 45(5): 5497–5512, 2022. 2, 6, 7

[4] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *NeurIPS*, 2020. 2, 7

[5] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. In *ICLR*, 2021. 2, 4

[6] Lucas Caccia, Jing Xu, Myle Ott, Marcaurelio Ranzato, and Ludovic Denoyer. On anytime learning at macroscale. In *CoLLAs*. PMLR, 2022. 2, 6

[7] Zhipeng Cai, Ozan Sener, and Vladlen Koltun. Online continual learning with natural distribution shifts: An empirical study with visual data. In *ICCV*, 2021. 1

[8] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, 2018. 1

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*. PMLR, 2020. 7

[10] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno Olshausen. Superposition of many models into one. In *NeurIPS*, 2019. 2

[11] Florent Chiaroni, Ghazaleh Khodabandelou, Mohamed-Cherif Rahal, Nicolas Hueber, and Frederic Dufaux. Counter-examples generation from a positive unlabeled image dataset. In *CVPR*. Elsevier, 2020. 5

[12] Aristotelis Chrysakis and Marie-Francine Moens. Online bias correction for task-free continual learning. In *ICLR*, 2023. 4

[13] Mehmet Dedeoglu, Sen Lin, Zhaofeng Zhang, and Junshan Zhang. Continual learning of generative models with limited data: From wasserstein-1 barycenter to adaptive coalescence. *IEEE TNNLS*, 2023. 5

[14] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *PNAS*, 118(43): e2103091118, 2021. 1, 3

[15] Zeyu Feng, Chang Xu, and Dacheng Tao. Self-supervised representation learning by rotation feature decoupling. In *CVPR*, 2019. 5

[16] Matthew Fickus, John Jasper, Emily J King, and Dustin G Mixon. Equiangular tight frames that contain regular simplices. *Linear Algebra and its applications*, 555:98–138, 2018. 4

[17] Yasir Ghunaim, Adel Bibi, Kumail Alhamoud, Motasem Alfarra, Hasan Abed Al Kader Hammoud, Ameya Prabhu, Philip HS Torr, and Bernard Ghanem. Real-time evaluation in online continual learning: A new hope. In *CVPR*, 2023. 2

[18] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 5

[19] Gabriele Graffieti, Davide Maltoni, Lorenzo Pellegrini, and Vincenzo Lomonaco. Generative negative replay for continual learning. *Neural Networks*, 162:369–383, 2023. 5

[20] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. Remind your neural network to prevent catastrophic forgetting. In *ECCV*. Springer, 2020. 2, 6

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 7

[22] Dong-Hwan Jang, Sanghyeok Chu, Joonhyuk Kim, and Bohyung Han. Pooling revisited: Your receptive field is suboptimal. In *CVPR*, 2022. 5

[23] Wenlong Ji, Yiping Lu, Yiliang Zhang, Zhun Deng, and Weijie J Su. An unconstrained layer-peeled perspective on neural collapse. In *ICLR*, 2021. 1, 3

[24] Haeyong Kang, Thang Vu, and Chang D Yoo. Learning imbalanced datasets with maximum margin loss. In *IEEE ICIP*. IEEE, 2021. 1

[25] Gyuhak Kim, Sepideh Esmaeilpour, Changnan Xiao, and Bing Liu. Continual learning based on ood detection and task masking. In *CVPR*, 2022. 2, 5

[26] 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. *PNAS*, 114(13):3521–3526, 2017. 2, 6

[27] Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on class incremental blurry task configuration with anytime inference. In *ICLR*, 2022. 1, 2, 6, 7

[28] Timothée Lesort, Andrei Stoian, and David Filliat. Regularization shortcomings for continual learning. *arXiv preprint arXiv:1912.03049*, 2019. 2

[29] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017. 2

[30] Jianfeng Lu and Stefan Steinerberger. Neural collapse with cross-entropy loss. *arXiv preprint arXiv:2012.08465*, 2020. 1, 3

[31] Zheda Mai, Ruiwen Li, Hyunwoo Kim, and Scott Sanner. Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *CVPRW*, 2021. 6

[32] Dustin G Mixon, Hans Parshall, and Jianzong Pi. Neural collapse with unconstrained features. *arXiv preprint arXiv:2011.11619*, 2020. 1, 3

[33] Sergey Pankov. Learning image transformations without training examples. In *ISVC*. Springer, 2011. 5

[34] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *PNAS*, 117(40):24652–24663, 2020. 1, 3

[35] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019. 6

[36] Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. Latent replay for real-time continual learning. In *IROS*. IEEE, 2020. 2

[37] Can Peng, Kun Zhao, Tianren Wang, Meng Li, and Brian C Lovell. Few-shot class-incremental learning from an open-set perspective. In *ECCV*, pages 382–397. Springer, 2022. 7

[38] Federico Pernici, Matteo Bruni, Claudio Baecchi, Francesco Turchini, and Alberto Del Bimbo. Class-incremental learning with pre-allocated fixed classifiers. In *ICPR*. IEEE, 2021. 3

[39] Jary Pomponi, Simone Scardapane, and Aurelio Uncini. Continual learning with invertible generative models. *Neural Networks*, 164:606–616, 2023. 2

[40] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *ECCV*. Springer, 2020. 7

[41] Akshay Rangamani and Andrzej Banburski-Fahey. Neural collapse in deep homogeneous classifiers and the role of weight decay. In *ICASSP*. IEEE, 2022. 1, 3

[42] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017. 1

[43] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *NeurIPS*, 2019. 6

[44] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 2

[45] Murray Shanahan, Christos Kaplanis, and Jovana Mitrović. Encoders and ensembles for task-free continual learning. *arXiv preprint arXiv:2105.13327*, 2021. 6

[46] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NeurIPS*, 2017. 2

[47] Abhishek Sinha, Kumar Ayush, Jiaming Song, Burak Uzkent, Hongxia Jin, and Stefano Ermon. Negative data augmentation. *arXiv preprint arXiv:2102.05113*, 2021. 2, 5

[48] Tom Tirer and Joan Bruna. Extended unconstrained features model for exploring deep neural collapse. In *ICML*. PMLR, 2022. 1, 3

[49] Alice Barbora Tumpach and Peter Kán. Temporal alignment of human motion data: A geometric point of view. In *GSI*. Springer, 2023. 5

[50] Chenyang Wang, Junjun Jiang, Xiong Zhou, and Xianming Liu. Resmooth: Detecting and utilizing ood samples when training with data augmentation. *IEEE TNNLS*, 2022. 2, 5

[51] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *CVPR*, 2022. 6

[52] Stephan Wojtowytsch. On the emergence of simplex symmetry in the final and penultimate layers of neural network classifiers. In *MSML*, pages 1–21. PMLR, 2021. 1, 3

[53] Ou Wu. Rethinking class imbalance in machine learning. *arXiv preprint arXiv:2305.03900*, 2023. 1

[54] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, pages 374–382, 2019. 1, 2, 4

[55] Yibo Yang, Shixiang Chen, Xiangtai Li, Liang Xie, Zhouchen Lin, and Dacheng Tao. Inducing neural collapse in imbalanced learning: Do we really need a learnable classifier at the end of deep neural network? In *NeuIPS*, 2022. 1, 3, 4

[56] Yibo Yang, Haobo Yuan, Xiangtai Li, Zhouchen Lin, Philip Torr, and Dacheng Tao. Neural collapse inspired feature-classifier alignment for few-shot class incremental learning. In *ICLR*, 2023. 1, 3, 4, 7

[57] Zitong Yang, Michal Lukasik, Vaishnavh Nagarajan, Zonglin Li, Ankit Singh Rawat, Manzil Zaheer, Aditya Krishna Menon, and Sanjiv Kumar. Resmem: Learn what you can and memorize the rest. *arXiv preprint arXiv:2302.01576*, 2023. 5, 6

[58] Zhi Yang, Jiwei Qin, Chuan Lin, Yanping Chen, Ruizhang Huang, and Yongbin Qin. Ganrec: A negative sampling model with generative adversarial network for recommendation. *Expert Systems with Applications*, 214:119155, 2023. 5

[59] Fei Ye and Adrian G Bors. Task-free continual learning via online discrepancy distance learning. In *NeurIPS*, 2022. 7

[60] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In *ICLR*, 2021. 2

[61] Bowen Zhao, Chen Chen, Qi Ju, and ShuTao Xia. Energy aligning for biased models. *arXiv preprint arXiv:2106.03343*, 2021. 1

[62] Zhisheng Zhong, Jiequan Cui, Yibo Yang, Xiaoyang Wu, Xiaojuan Qi, Xiangyu Zhang, and Jiaya Jia. Understanding imbalanced semantic segmentation through neural collapse. In *CVPR*, 2023. 1, 3

[63] Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning. In *ICLR*, 2022. 2, 6, 7

[64] Jinxin Zhou, Xiao Li, Tianyu Ding, Chong You, Qing Qu, and Zhihui Zhu. On the optimization landscape of neural collapse under mse loss: Global optimality with unconstrained features. In *ICML*. PMLR, 2022. 1, 3

[65] Zhihui Zhu, Tianyu Ding, Jinxin Zhou, Xiao Li, Chong You, Jeremias Sulam, and Qing Qu. A geometric analysis of neural collapse with unconstrained features. In *NeurIPS*, 2021. 1, 3