DISCRIMINATIVE REPRESENTATION LOSS (DRL): A MORE EFFICIENT APPROACH THAN GRADIENT RE-PROJECTION IN CONTINUAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

The use of episodic memories in continual learning has been shown to be effective in terms of alleviating catastrophic forgetting. In recent studies, several gradient-based approaches have been developed to make more efficient use of compact episodic memories, which constrain the gradients resulting from new samples with those from memorized samples, aiming to reduce the diversity of gradients from different tasks. In this paper, we reveal the relation between diversity of gradients and discriminativeness of representations, demonstrating connections between Deep Metric Learning and continual learning. Based on these findings, we propose a simple yet highly efficient method – Discriminative Representation Loss (DRL) – for continual learning. In comparison with several state-of-the-art methods, DRL shows effectiveness with low computational cost on multiple benchmark experiments in the setting of online continual learning.

1 INTRODUCTION

In the real world, we are often faced with situations where data distributions are changing over time, and we would like to update our models by new data in time, with bounded growth in system size. These situations fall under the umbrella of "continual learning", which has many practical applications, such as recommender systems, retail supply chain optimization, and robotics (Lesort et al., 2019; Diethe et al., 2018; Tian et al., 2018). Comparisons have also been made with the way that humans are able to learn new tasks without forgetting previously learned ones, using common knowledge shared across different skills. The fundamental problem in continual learning is *catastrophic forgetting* (McCloskey & Cohen, 1989; Kirkpatrick et al., 2017), *i.e.* (neural network) models have a tendency to forget previously learned tasks while learning new ones.

There are two main categories of methods for alleviating forgetting in continual learning: *i*) preserving knowledge of models of previous tasks, including methods for parameter regularization (Kirkpatrick et al., 2017; Zenke et al., 2017; Nguyen et al., 2018) and methods for incrementally evolving the model (Schwarz et al., 2018; Hung et al., 2019); *ii*) preserving the knowledge of data distributions of previous tasks, including replay-based methods (Shin et al., 2017; Rolnick et al., 2019), methods for generating compact episodic memories (Chen et al., 2018; Aljundi et al., 2019), and methods using episodic memories to refine gradients when updating model parameters (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019a; Riemer et al., 2019; Farajtabar et al., 2020).

Gradient-based approaches using episodic memories, in particular, have been receiving increasing attention. The essential idea is to use gradients produced by samples from episodic memories to constrain the gradients produced by new samples, *e.g.* by ensuring the inner product of the pair of gradients is non-negative (Lopez-Paz & Ranzato, 2017) as follows:

$$\langle g_t, g_k \rangle = \left\langle \frac{\partial \mathcal{L}(x_t, \theta)}{\partial \theta}, \frac{\partial \mathcal{L}(x_k, \theta)}{\partial \theta} \right\rangle \geq 0, \ \ \forall k < t \tag{1}$$

where t and k are time indices, x_t denotes a new sample from the current task, and x_k denotes a sample from the episodic memory. Thus, the updates of parameters are forced to preserve the performance on previous tasks as much as possible.

In Gradient Episodic Memory (GEM) (Lopez-Paz & Ranzato, 2017), g_t is projected to a direction that closest to it in L_2-norm whilst also satisfying Eq. (1):

$$\min_{\tilde{g}} \frac{1}{2}||g_t - \tilde{g}||_2^2, \;\; s.t. \langle \tilde{g}, g_k \rangle \geq 0, \;\; \forall k < t \tag{2}$$

Optimization of this objective requires a high-dimensional quadratic program and thus is computationally expensive. Averaged-GEM (A-GEM) (Chaudhry et al., 2019a) alleviates the computational burden of GEM by using the averaged gradient over a batch of samples instead of individual gradients of samples in the episodic memory. This not only simplifies the computation, but also obtains comparable performance with GEM. Meta-Experience Replay (MER) (Riemer et al., 2019) integrates the inner product of gradients into the loss function through a Reptile-style algorithm (Nichol & Schulman, 2018), which was originally formulated for the purposes of meta-learning. Orthogonal Gradient Descent (OGD) (Farajtabar et al., 2020) projects g_t to the direction that is perpendicular to the surface formed by $\{g_k | k < t\}$. Finally, Aljundi et al. (2019) propose Gradient-based Sample Selection (GSS), which selects samples that produce gradients with maximum diversity to store in episodic memory. Here diversity is measured by the cosine similarity between gradients. Since the cosine similarity is computed using the inner product of two normalized gradients, GSS embodies the same principle as other gradient-based approaches with episodic memories. Although GSS suggests the samples with most diverse gradients are important for generalization across tasks, Chaudhry et al. (2019b) show that the average gradient over a small set of random samples may be able to obtain good generalization as well.

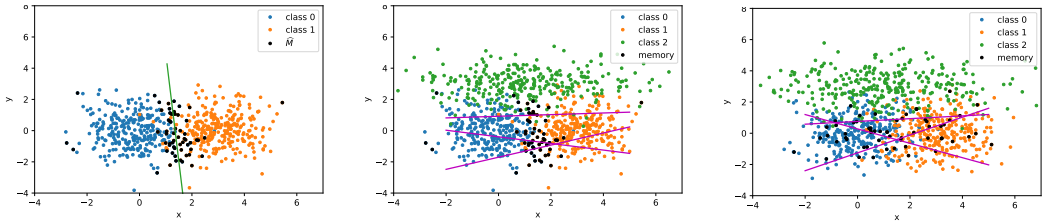
In this paper, we answer the following questions: *i*) Which samples tend to produce diverse gradients that strongly conflict with other samples and why are such samples able to help with generalization? *ii*) Why does a small set of randomly chosen samples also help with generalization? *iii*) Can we reduce the diversity of gradients in a more efficient way? Our answers reveal the relation between diversity of gradients and discriminativeness of representations, and further show connections between Deep Metric Learning (DML) (Kaya & Bilge, 2019; Roth et al., 2020) and continual learning. Drawing on these findings we propose a new approach, Discriminative Representation Loss (DRL), for classification tasks in continual learning. Our methods show improved performance with relatively low computational cost when compared to several state-of-the-art (SOTA) methods across multiple benchmark tasks in the setting of online continual learning.

2 A New Perspective of Reducing Diversity of Gradients

According to Eq. (1), larger cosine similarities between gradients produced by current and previous tasks result in better performance in continual learning. This can be interpreted from the perspective of constrained optimization as discussed by Aljundi et al. (2019). Moreover, the diversity of gradients relates to the Gradient Signal to Noise Ratio (GSNR) (Liu et al., 2020), which plays a crucial role in the model's generalization ability. Intuitively, when more of the gradients point in the same direction, the variance will be smaller, leading to a larger GSNR, and consequently, improved test-time performance. This in turn indicates that samples that lead to the most diverse gradients provide the most difficulty of generalization.

2.1 The source of gradient diversity

We first conducted a simple experiment on classification tasks of 2-D Gaussian distributions, and tried to identify samples with most diverse gradients in the 2-D feature space. We trained a linear model on the first task to discriminate between two classes (blue and orange dots in Fig. 1a). We then applied the algorithm Gradient-based Sample Selection with Interger Quadratic Programming (GSS-IQP) (Aljundi et al., 2019) to select 10% of the samples of training data that produce gradients with the lowest similarity (black dots in Fig. 1a), and denote this set of samples as $\widehat{M} = \min_M \sum_{i,j \in M} \frac{\langle g_i, g_j \rangle}{||g_i|| \cdot ||g_j||}$. It is clear from Fig. 1a that the samples in $\widehat{M}$ are mostly around the decision boundary between the two classes. Increasing the size of $\widehat{M}$ results in the inclusion of samples that trace the outer edges of the data distributions from each class. *Clearly the gradients can be strongly opposed when samples from different classes are very similar*. Samples close to decision boundaries are most likely to exhibit this characteristic. Intuitively, storing the decision boundaries of previously learned classes should be an effective way to preserve classification performance on those classes. However,



(a) Samples with most diverse gradients ($\widehat{M}$) after learning task 1, the green line is the decision boundary.

(b) Learned decision boundaries (purple lines) after task 2. Here the episodic memory includes samples in $\widehat{M}$.

(c) Learned decision boundaries (purple lines) after task 2. Here the episodic memory consists of random samples.

Figure 1: 2-D classification examples, the x and y axis are the coordinates (also features) of samples. We sequentially train a logistic regression model on two tasks: the first task is to classify two classes as shown in (a); the second class is to incrementally classify a third class as shown in (b) and (c). The solid lines are decision boundaries between classes.

if the episodic memory only includes samples representing the learned boundaries, it may miss important information when the model is required to incrementally learn new classes. We show this by introducing a second task - training the model above on a third class (green dots). We display the decision boundaries (which split the feature space in a one *vs.* all manner) learned by the model after task 2 with $\widehat{M}$ (Fig. 1b) and a random set of samples (Fig. 1c) from task 1 as the episodic memory. The random episodic memory shows better performance than the one selected by GSS-IQP, since the new decision boundaries rely on samples not included in $\widehat{M}$. It explains why randomly selected memories may generalize better in continual learning. Ideally, with $\widehat{M}$ large enough, the model can remember all edges of each class, and hence learn much more accurate decision boundaries sequentially. However, memory size is often limited in practice, especially for high-dimensional data. A more efficient way could be learning more informative representations. The experimental results indicate that: *1) more similar representations in different classes result in more diverse gradients. 2) more diverse representations in a same class may help with learning new tasks incrementally.*

Now we formalise the connection between the diversity of gradients and the discriminativeness of representations for the linear model (proofs are in Appx. A). **Notations**: Let $\mathcal{L}$ represent the softmax cross entropy loss, $\mathbf{W} \in \mathbb{R}^{D\times K}$ is the weight matrix of the linear model, and $\mathbf{x}_n \in \mathbb{R}^D$ denotes the input data, $\mathbf{y}_n \in \mathbb{R}^K$ is a one-hot vector that denotes the label of $\mathbf{x}_n$, D is the dimension of representations, K is the number of classes. Let $\boldsymbol{p}_n = softmax(\mathbf{o}_n)$, where $\mathbf{o}_n = \mathbf{W}^T\mathbf{x}_n$, the gradient $\boldsymbol{g}_n = \nabla_{\mathbf{W}}\mathcal{L}(\mathbf{x}_n, \mathbf{y}_n; \mathbf{W})$. x_n, x_m are two different samples when $n \neq m$.

Theorem 1. *Suppose $\mathbf{y}_n \neq \mathbf{y}_m$, and let c_n denote the class index of $\mathbf{x}_n$ (i.e. $\mathbf{y}_{n,c_n} = 1, \mathbf{y}_{n,i} = 0, \forall i \neq c_n$). Let $\boldsymbol{\alpha} \triangleq ||\boldsymbol{p}_n||^2 + ||\boldsymbol{p}_m||^2$, $\boldsymbol{\beta} \triangleq \boldsymbol{p}_{n,c_m} + \boldsymbol{p}_{m,c_n}$ and $\boldsymbol{\delta} \triangleq ||\boldsymbol{p}_n - \boldsymbol{p}_m||_2^2$, then:*

$$\Pr\left(sign(\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle) = sign(-\langle \mathbf{x}_n, \mathbf{x}_m\rangle)\right) = \Pr(2\boldsymbol{\beta} + \boldsymbol{\delta} > \boldsymbol{\alpha}),$$

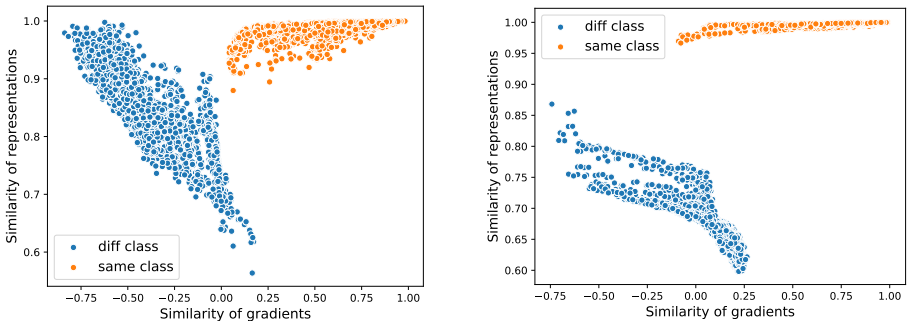
This theorem explains why samples close to a decision boundary tend to produce negative inner products of their gradients. In such a case, $\langle \mathbf{x}_n, \mathbf{x}_m\rangle$ is likely positive, $\boldsymbol{\beta}$ is close to 1 because the predictions are likely to assign relatively large probabilities to both classes, since $\boldsymbol{\delta} \geq 0, \boldsymbol{\alpha} \leq 2$, it will result in $\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle < 0$ with a high probability. We demonstrate visualized results under several circumstances in Fig. 4 in Appx. A.

Theorem 2. *Suppose $\mathbf{y}_n = \mathbf{y}_m$, when $\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle \neq 0$, we have:*

$$sign(\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle) = sign(\langle \mathbf{x}_n, \mathbf{x}_m\rangle),$$

Theorem 2 says $\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle$ can be negative only if $\langle \mathbf{x}_n, \mathbf{x}_m\rangle$ is negative when the two samples from a same class. In other words, if we can guarantee the inner product of representations in a same class always be non-negative, then their gradients will never conflict with each other. A deep neural network can be viewed as a representation extractor on the top of a linear model. Using activation functions such as ReLU satisfies this non-negative condition.

Extending this theoretical analysis based on a linear model, we will provide an empirical study of a non-linear model (a neural network) trained by higher-dimensional data in the following and



(a) Similarities of gradients *vs.* representations (class 7 & 9)

(b) Similarities of gradients *vs.* representations (class 0 & 1)

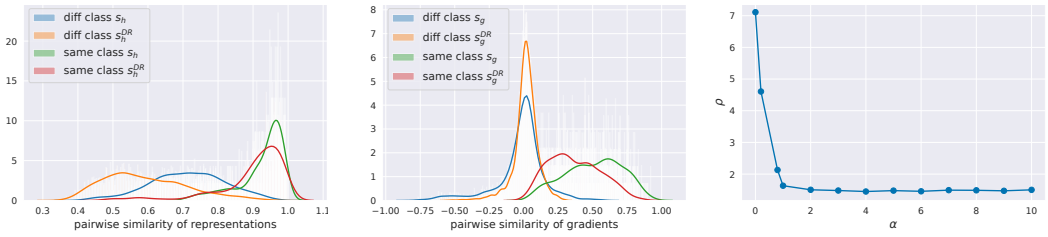
Figure 2: Similarities of gradients and representations of two classes in the MNIST dataset. The x and y axis are the cosine similarity of gradients and representations, respectively. Blue dots indicate the similarity of *negative pairs* (two samples from two different classes), while orange dots indicate that of *positive pairs* (the two samples are from the same class).

show that Theorems 1 and 2 are highly consistent with these results. We first trained two binary classifiers for two groups of MNIST classes ($\{0, 1\}$ and $\{7, 9\}$). The classifiers have two hidden dense layers each with 100 hidden units and ReLU activations. We randomly chose 100 test samples from each group, and computed the pairwise cosine similarities of gradients and representations after the model is trained. Representations are obtained by concatenating the output of all layers of the neural network. We display the similarities in Fig. 2, where blue dots indicate ***negative pairs*** (two samples from two different classes), while orange dots indicate ***positive pairs*** (two samples from the same class). The x and y axes are similarity of gradients and representations, respectively. In Figs. 2a and 2b, the correlation coefficients of blue dots are -0.86 and -0.85, which of orange dots are 0.71 and 0.79. In all cases, the similarities of representations show strong correlations with the similarities of gradients and it is especially true for blue dots. The similarities of gradients of orange dots are almost non-negative in both cases. These results obviously align with Theorems 1 and 2. In addition, the blue and orange dots are perfectly separable on the y axis in Fig. 2b, which indicates that the classifier for class 0 and 1 has learnt strongly discriminative representations, and as a result achieves nearly perfect (99.95%) accuracy on the test set. In comparison, the classifier for class 7 and 9 has learnt less discriminative representations, resulting in lower test accuracy (96.25%).

2.2 Connecting Deep Metric Learning to Continual Learning

The discriminativeness of representations can be interpreted in terms of margins between classes in the representation space, and learning larger margins has been an active research area for a few decades. For example, Kernel Fisher Discriminant analysis (KFD) (Mika et al., 1999) and distance metric learning (Weinberger et al., 2006) aim to learn kernels that can obtain larger margins in an implicit representation space, whereas Deep Metric Learning (DML) (Kaya & Bilge, 2019; Roth et al., 2020) leverages deep neural networks to learn embeddings that maximize margins in an explicit representation space. In this sense, DML has the potential to help with reducing the diversity of gradients in continual learning.

However, we must consider preserving unused information in the representation space for later use since the model is required to learn new classes/instances sequentially in continual learning. Regarding our findings so far, continual learning shares the interests of maximizing margins in DML but prefers less compact representation space. For example, in the 2-D experiments (Fig. 1) we would prefer keeping larger variance within each class in task 1 for a better compatibility with task 2. Most methods of DML in principle try to minimize the similarities between different classes and maximize the similarities within the same class (Schroff et al., 2015; Wang et al., 2017; Deng et al., 2019). In the setting of continual learning, we suggest an opposite way regarding the intra-class compactness: minimizing the similarities within the same class for obtaining less compact representation space. Roth et al. (2020) proposed a ρ-spectrum metric to measure the information entropy contained in the representation space (details are provided in Appx. C) and introduced a ρ-regularization method to restrain over-compression of representations. The ρ-regularization method randomly replaces



(a) Similarities of representations with and without $\mathcal{L}_{DRL}$

(b) Similarities of gradients with and without $\mathcal{L}_{DRL}$

(c) Relation between α and ρ-spectrum.

Figure 3: Effects of $\mathcal{L}_{DRL}$ on reducing diveristy of gradients and ρ-spectrum. (a) and (b) display distributions of similarities of representations and gradients. s_h^{DR} and s_h denote similarities of representations with and without $\mathcal{L}_{DRL}$, respectively, s_g^{DR} and s_g denote similarities of gradients with and without $\mathcal{L}_{DRL}$, respectively. (c) demonstrates increasing α in $\mathcal{L}_{DRL}$ can reduce ρ effectively.

negative pairs by positive pairs with a pre-selected probability p_ρ. Nevertheless, switching pairs is inefficient or even detrimental to the performance in an online setting because some negative pairs may never be learned in this way. Thus, we propose a different approach to restrain the compression of representations which will be introduced in the following.

3 Discriminative Representation Loss

Based on our findings in the above section, we propose an auxiliary objective Discriminative Representation Loss (DRL) for classification tasks in continual learning, which is straightforward, robust, and efficient. Instead of explicitly re-projecting gradients during training process, DRL helps with decreasing gradient diversity by optimizing the representations. As defined in Eq. (3), DRL consists of two parts: one is for minimizing the similarities of representations from different classes ($\mathcal{L}_{bt}$) which can reduce the diversity of gradients from different classes, the other is for minimizing the similarities of representations from a same class ($\mathcal{L}_{wi}$) which helps preserve discriminative information for future tasks in continual learning.

$$\min_{\Theta} \mathcal{L}_{DRL} = \min_{\Theta}(\mathcal{L}_{bt} + \alpha\mathcal{L}_{wi}), \quad \alpha > 0,$$

$$\mathcal{L}_{bt} = \frac{1}{N_{bt}} \sum_{i=1}^{B} \sum_{j \neq i, y_j \neq y_i}^{B} \langle h_i, h_j \rangle, \quad \mathcal{L}_{wi} = \frac{1}{N_{wi}} \sum_{i=1}^{B} \sum_{j \neq i, y_j = y_i}^{B} \langle h_i, h_j \rangle, \tag{3}$$

where Θ denotes the parameters of the model, B is training batch size. N_{bt}, N_{wi} are the number of negative and positive pairs, respectively. α is a hyperparameter controlling the strength of $\mathcal{L}_{wi}$, h_i is the representation of x_i, y_i is the label of x_i. The final loss function combines the commonly used softmax cross entropy loss for classification tasks ($\mathcal{L}$) with DRL ($\mathcal{L}_{DRL}$) as shown in Eq. (4),

$$\widehat{\mathcal{L}} = \mathcal{L} + \lambda\mathcal{L}_{DRL}, \quad \lambda > 0, \tag{4}$$

where λ is a hyperparameter controlling the strength of $\mathcal{L}_{DRL}$, which is larger for increased resistance to forgetting, and smaller for greater elasticity. We verify the effects of $\mathcal{L}_{DRL}$ by training a model with/without $\mathcal{L}_{DRL}$ on Split-MNIST tasks: Fig. 3a shows that $\mathcal{L}_{DRL}$ notably reduces the similarities of representations from different classes while making representations from a same class less similar; Fig. 3b shows the analogous effect on gradients from different classes and a same class. Fig. 3c demonstrates increasing α can effectively decrease ρ-spectrum to a low-value level, where lower values of ρ indicate higher variance of the representations and hence more information entropy retained. We will show the correlation between ρ-spectrum and the model performance in Sec. 5.

4 Online memory update and Balanced Experience Replay

We follow the *online setting* of continual learning as was done for other gradient-based approaches with episodic memories (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019a; Aljundi et al., 2019), in which the model only trained with one epoch on the training data.

We update the episodic memories by the basic ring buffer strategy: keep the last n_c samples of class c in the memory buffer, where n_c is the memory size of a seen class c. We have deployed the episodic memories with a fixed size, implying a fixed budget for the memory cost. Further, we maintain a uniform distribution over all seen classes in the memory. The buffer may not be evenly allocated to each class before enough samples are acquired for newly arriving classes. We show pseudo-code of the memory update strategy in Alg. 1 in Appx. B for a clearer explanation. For class-incremental learning, this strategy can work without knowing task boundaries. Since DRL and methods of DML depend on the pairwise similarities of samples, we would prefer the training batch to include as wide a variety of different classes as possible to obtain sufficient discriminative information. Hence, we adjust the Experience Replay (ER) strategy (Chaudhry et al., 2019b) for the needs of such methods. The idea is to uniformly sample from seen classes in the memory buffer to form a training batch, so that this batch can contain as many seen classes as possible. Moreover, we ensure the training batch includes at least one positive pair of each selected class (minimum 2 samples in each class) to enable the parts computed by positive pairs in the loss. In addition, we also ensure the training batch includes at least one class from the current task. We call this Balanced Experience Replay (BER). The pseudo code is in Alg. 2 of Appx. B. Note that we update the memory and form the training batch based on the task ID instead of class ID for instance-incremental tasks (e.g. permuted MNIST tasks), as in this case each task always includes the same set of classes.

5 EXPERIMENTS

In this section we evaluate our methods on multiple benchmark tasks by comparing with several baseline methods in the setting of online continual learning.

Benchmark tasks: We have conducted experiments on the following benchmark tasks:

Permuted MNIST: 10 tasks using the MNIST dataset (LeCun et al., 2010), each task includes the same 10 classes with different permutation of features. The training size is 1000 samples per task;

Split MNIST: 5 tasks using the MNIST dataset, each task includes two classes which are disjoint from the other tasks. The training size is 1000 samples per task;

Split Fashion-MNIST: 5 tasks using the Fashion-MNIST dataset (Xiao et al., 2017), each task includes two classes which are disjoint from the other tasks. The training size is 1000 samples per task;

Split CIFAR-10: 5 tasks using the CIFAR-10 dataset (Krizhevsky et al., 2009), each task includes two classes which are disjoint from other tasks. The training size is 3000 samples per task;

Split CIFAR-100: 10 tasks using the CIFAR-100 dataset (Krizhevsky et al., 2009), each task includes 10 classes which are disjoint from other tasks. The training size is 5000 samples per task.

N.B.: We use *single-head* (shared output) models in all of our experiments, meaning that we do not require a task identifier at testing time. Such settings are more difficult for continual learning but more practical in real applications.

Baselines: We compare our methods with the following related methods, including gradient-based approaches of continual learning and two SOTA methods of DML:

A-GEM (Chaudhry et al., 2019a): refines gradients using samples from episodic memory, similar to GEM Lopez-Paz & Ranzato (2017) but with lower computational cost.

GSS-greedy (Aljundi et al., 2019): based on gradient diversities for selecting samples into the episodic memory. It is a variant of GSS-IQP with lower computational cost but with similar performance.

ER (Chaudhry et al., 2019b): a basic experience replay strategy, yet was shown to achieve better performance than A-GEM and MER (Riemer et al., 2019) in the online continual learning setting. It simply composes a training batch divided equally between samples from the episodic memory and samples from the current task. We consider this as a baseline of replay-based methods.

Multisimilarity (Wang et al., 2019): A SOTA method of DML which has shown outstanding performance in a comprehensive empirical study of DML (Roth et al., 2020).

R-Margin (Roth et al., 2020): A SOTA method of DML which deploy the ρ regularization method for Margin loss (Wu et al., 2017) and has shown outstanding performance in Roth et al. (2020).

N.B.: We deploy the losses of Multisimilarity and R-Margin as auxiliary objectives as the same as DRL because using standalone such losses causes difficulties of convergence in our experimental settings. We provide the definitions of these two losses in Appx. C.

Performance measures: We use the following measures to evaluate the performance of all methods:

Average accuracy, which is evaluated after learning all tasks: $\bar{a}_t = \frac{1}{t}\sum_{i=1}^{t} a_{t,i}$, where t is the index of the latest task, $a_{t,i}$ is the accuracy of task i after learning task t.

Average forgetting (Chaudhry et al., 2018), which measures average accuracy drop of all tasks after learning the whole task sequence: $\bar{f}_t = \frac{1}{t-1}\sum_{i=1}^{t-1} \max_{j\in\{i,\dots,t-1\}}(a_{j,i} - a_{t,i})$.

Average intransigence (Chaudhry et al., 2018), which measures the inability of a model learning new tasks: $\bar{I}_t = \frac{1}{t}\sum_{i=1}^{t} a_i^* - a_i$, where a_i is the accuracy of task i at time i. We use the best accuracy among all compared models as a_i^* instead of the accuracy obtained by an extra model that is solely trained on task i.

Experimental settings: We use the vanilla SGD optimizer for all experiments without any scheduling. For tasks on MNIST and Fashion-MNIST, we use a dense neural network with two hidden layers and ReLU activations, and each layer has 100 hidden units. For tasks on CIFAR datasets, we use the same reduced Resnet18 as used in Chaudhry et al. (2019a). All networks are trained from scratch without regularization scheme. For the dense network, representations are the concatenation of outputs of all layers; for reduced Resnet18, representations are the concatenation of the input of the final linear layer and output logits. We deploy BER as the replay strategy for DRL, Multisimilarity, and R-Margin. The memory size for tasks on MNIST and Fashin-MNIST is 300 samples. For tasks on CIFAR-10 and CIFAR-100 the memory size is 2000 and 5000 samples, respectively. The standard deviation shown in all results are evaluated over 10 runs with different random seeds. We use 10% of training set as validation set for choosing hyperparameters by cross validation. More details of experimental settings and hyperparameters are given in Appx. E.

Tabs. 1 to 3 give the average accuracy, average forgetting, and average intransigence of all methods on all benchmark tasks, respectively. As we can see, the forgetting and intransigence often conflict with each other which is the most common phenomenon in continual learning. Our method DRL is able to get a better trade-off between them and thus outperforms other methods over most benchmark tasks in terms of average accuracy. In addition, Multisimilarity and R-Margin both have shown relatively good performance, which indicate learning a better representation is a more efficient way than direct gradient re-projection. It is worth noting that the ρ-spectrum exhibits strong correlation with the average accuracy on all tasks except Split MNIST (Tab. 4). Regarding Split MNIST, the ρ-spectrum highly correlates with the average intransigence and consequently affect the average forgetting in an opposite direction so that causes a cancellation of effects on average accuracy. In addition, We found that GSS often obtains a smaller ρ than other methods without getting a better performance. In general, the ρ-spectrum is the smaller the better because it indicates the representations are more informative. However, it may be detrimental to the performance when ρ is too small as the learned representations are too noisy. DRL is robust to this issue because ρ keeps relatively stable when α is larger than a certain value as shown in Fig. 3c.

The computational complexity of DRL is $O(B^2H)$, where B is training batch size, H is the dimension of representations. B is small (10 or 20 in our experiments) and $H \ll W$, commonly $B^2H \ll W$ in neural networks, where W is the number of network parameters. We compare the training time of all methods on MNIST tasks in Tab. 5 in Appx. D, which shows the representation-based methods require much lower computational cost than gradient-based approaches.

6 Conclusion

The two fundamental problems of continual learning with small episodic memories are: (*i*) how to make the best use of a small set of samples; and (*ii*) how to construct a small set of samples that are most representative of a large dataset. Gradient-based approaches have shown that the diversity of gradients computed on data from different tasks is a key to generalization over these tasks.

In this paper we demonstrate that the most diverse gradients are from samples that are close to class boundaries; moreover, those samples further away from learned boundaries but lie on the

Table 1: Average accuracy, the bold font indicates the best performance on this criterion

	Permuted MNIST	Split MNIST	Split Fashion	Split CIFAR10	Split CIFAR100
DRL	$\mathbf{0.787 \pm 0.004}$	$\mathbf{0.882 \pm 0.006}$	$\mathbf{0.782 \pm 0.004}$	0.461 ± 0.012	$\mathbf{0.171 \pm 0.001}$
BER	0.758 ± 0.003	0.864 ± 0.007	0.769 ± 0.006	0.442 ± 0.011	0.153 ± 0.006
ER	0.761 ± 0.003	0.840 ± 0.008	0.756 ± 0.012	0.421 ± 0.02	0.145 ± 0.008
A-GEM	0.759 ± 0.011	0.854 ± 0.008	0.606 ± 0.025	0.331 ± 0.021	0.098 ± 0.003
GSS	0.771 ± 0.003	0.828 ± 0.018	0.725 ± 0.009	0.420 ± 0.030	0.139 ± 0.010
Multisim	0.781 ± 0.002	$\mathbf{0.881 \pm 0.006}$	0.776 ± 0.005	$\mathbf{0.495 \pm 0.008}$	0.162 ± 0.003
R-Margin	0.758 ± 0.004	0.860 ± 0.012	0.770 ± 0.006	0.460 ± 0.013	0.169 ± 0.005

Table 2: Average forgetting, the bold font indicates the best performance on this criterion

	Permuted MNIST	Split MNIST	Split Fashion	Split CIFAR10	Split CIFAR100
DRL	0.060 ± 0.003	$\mathbf{0.084 \pm 0.009}$	0.167 ± 0.015	0.322 ± 0.050	0.208 ± 0.009
BER	0.071 ± 0.002	0.114 ± 0.010	0.174 ± 0.019	0.433 ± 0.021	0.206 ± 0.003
ER	0.084 ± 0.003	0.156 ± 0.015	0.235 ± 0.017	0.486 ± 0.030	0.366 ± 0.005
A-GEM	$\mathbf{0.054 \pm 0.011}$	0.107 ± 0.009	0.461 ± 0.034	0.347 ± 0.025	$\mathbf{0.182 \pm 0.011}$
GSS	0.076 ± 0.002	0.179 ± 0.024	0.274 ± 0.022	$\mathbf{0.109 \pm 0.034}$	0.186 ± 0.007
Multisim	0.059 ± 0.003	0.096 ± 0.009	0.183 ± 0.017	0.333 ± 0.021	0.259 ± 0.012
R-Margin	0.069 ± 0.002	0.096 ± 0.014	$\mathbf{0.140 \pm 0.020}$	0.396 ± 0.045	0.246 ± 0.014

Table 3: Average intransigence, the bold font indicates the best performance on this criterion

	Permuted MNIST	Split MNIST	Split Fashion	Split CIFAR10	Split CIFAR100
DRL	$\mathbf{0.020 \pm 0.001}$	0.026 ± 0.003	0.069 ± 0.010	0.121 ± 0.042	0.139 ± 0.008
BER	0.040 ± 0.002	0.019 ± 0.002	0.077 ± 0.012	0.046 ± 0.012	0.159 ± 0.005
ER	0.025 ± 0.001	0.009 ± 0.002	0.041 ± 0.006	$\mathbf{0.026 \pm 0.007}$	$\mathbf{0.023 \pm 0.007}$
A-GEM	0.075 ± 0.005	0.031 ± 0.003	$\mathbf{0.010 \pm 0.003}$	0.227 ± 0.011	0.235 ± 0.010
GSS	0.076 ± 0.002	$\mathbf{0.009 \pm 0.003}$	0.274 ± 0.022	0.357 ± 0.022	0.203 ± 0.014
Multisim	0.026 ± 0.002	0.018 ± 0.003	0.062 ± 0.010	0.075 ± 0.011	0.103 ± 0.012
R-Margin	0.041 ± 0.002	0.030 ± 0.003	0.105 ± 0.017	0.059 ± 0.026	0.107 ± 0.011

Table 4: Correlation between model performance and ρ-spectrum on all benchmark tasks

Coefficient	Permuted MNIST	Split MNIST	Split Fashion	Split CIFAR10	Split CIFAR100
Avg. Acc.	-0.8379	0.0461	-0.5553	-0.7689	-0.7103
Avg. Forg.	0.2616	-0.3879	0.4331	0.1005	0.0028
Avg. Intran.	0.4659	0.7206	-0.0978	0.2463	0.2229

class edges are also important to unseen tasks. We formally connect the diversity of gradients to discriminativeness of representations, which leads to an alternative way to reduce the diversity of gradients in continual learning. We subsequently exploit ideas from DML for learning more discriminative representations, and furthermore identify the shared and different interests between continual learning and DML. In continual learning we would prefer larger margins between classes as the same as in DML. The difference is that continual learning requires less compact representations for better compatibility with future tasks. Based on these findings, we provide a simple yet efficient approach to solving the first problem listed above. Our findings also shed light on the second problem: it would be better for the memorized samples to preserve as much variance as possible. In most of our experiments, randomly chosen samples outperform those selected by gradient diversity (GSS) due to the limit on memory size in practice. It could be helpful to select memorized samples by separately considering the representativeness of inter- and intra-class samples, i.e., those representing margins and edges. We will leave this for future work.

REFERENCES

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, pp. 11816–11825, 2019.

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019a. URL `https://openreview.net/forum?id=Hkf2_sC5FX`.

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

Yu Chen, Tom Diethe, and Neil Lawrence. Facilitating bayesian continual learning by natural gradients and stein gradients. *Continual Learning Workshop of 32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, 2018.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2019.

Tom Diethe, Tom Borchert, Eno Thereska, Borja de Balle Pigem, and Neil Lawrence. Continual learning in practice. In *Continual Learning Workshop of 32nd Converence on Neural Information Processing Systems (NeurIPS 2018)*, 2018.

Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 3762–3773. PMLR, 2020.

Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. In *Advances in Neural Information Processing Systems*, pp. 13647–13657, 2019.

Mahmut Kaya and Hasan Şakir Bilge. Deep metric learning: A survey. *Symmetry*, 11(9):1066, 2019.

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. *Proceedings of the national academy of sciences*, pp. 201611835, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

Yann LeCun, Corinna Cortes, and Christopher JC Burges. MNIST handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz-Rodríguez. Continual learning for robotics. *arXiv preprint arXiv:1907.00182*, 2019.

Jinlong Liu, Yunzhi Bai, Guoqing Jiang, Ting Chen, and Huayan Wang. Understanding why neural networks generalize well through GSNR of parameters. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=HyevIJStwH`.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pp. 6467–6476, 2017.

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

Sebastian Mika, Gunnar Ratsch, Jason Weston, Bernhard Scholkopf, and Klaus-Robert Mullers. Fisher discriminant analysis with kernels. In *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*, pp. 41–48. Ieee, 1999.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

Alex Nichol and John Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2, 2018.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, , and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=B1gTShAct7`.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, pp. 348–358, 2019.

Karsten Roth, Timo Milbich, Samarth Sinha, Prateek Gupta, Bjoern Ommer, and Joseph Paul Cohen. Revisiting training strategies and generalization performance in deep metric learning. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. *arXiv preprint arXiv:1805.06370*, 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems*, pp. 2990–2999, 2017.

Huangshi Tian, Minchen Yu, and Wei Wang. Continuum: A platform for cost-aware, low-latency continual learning. In *Proceedings of the ACM Symposium on Cloud Computing*, pp. 26–40, 2018.

Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. Deep metric learning with angular loss. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2593–2601, 2017.

Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5022–5030, 2019.

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

Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2840–2848, 2017.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pp. 3987–3995, 2017.

A PROOF OF THEOREMS

Notations: Let $\mathcal{L}$ represent the softmax cross entropy loss, $\mathbf{W} \in \mathbb{R}^{D\times K}$ is the weight matrix of the linear model, and $\mathbf{x}_n \in \mathbb{R}^D$ denotes the input data, $\mathbf{y}_n \in \mathbb{R}^K$ is a one-hot vector that denotes the label of $\mathbf{x}_n$, D is the dimension of representations, K is the number of classes. Let $\boldsymbol{p}_n = softmax(\mathbf{o}_n)$, where $\mathbf{o}_n = \mathbf{W}^T\mathbf{x}_n$, the gradient $\boldsymbol{g}_n = \nabla_{\mathbf{W}}\mathcal{L}(\mathbf{x}_n, \mathbf{y}_n; \mathbf{W})$. x_n, x_m are two different samples when $n \neq m$.

Fact 1. *Let $\boldsymbol{\ell}_n^{'} = \partial\mathcal{L}(\mathbf{x}_n, \mathbf{y}_n; \mathbf{W})/\partial\mathbf{o}_n$, by the chain rule, we have:*

$$\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle = \langle \mathbf{x}_n, \mathbf{x}_m\rangle\langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'}\rangle,$$

Theorem 1. *Suppose $\mathbf{y}_n \neq \mathbf{y}_m$, let c_n denote the class index of $\mathbf{x}_n$ (i.e. $\mathbf{y}_{n,c_n} = 1, \mathbf{y}_{n,i} = 0, \forall i \neq c_n$). Let $\boldsymbol{\alpha} \triangleq ||\boldsymbol{p}_n||^2 + ||\boldsymbol{p}_m||^2$, $\boldsymbol{\beta} \triangleq \boldsymbol{p}_{n,c_m} + \boldsymbol{p}_{m,c_n}$ and $\boldsymbol{\delta} \triangleq ||\boldsymbol{p}_n - \boldsymbol{p}_m||_2^2$, then:*

$$\Pr\left(sign(\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle) = sign(-\langle \mathbf{x}_n, \mathbf{x}_m\rangle)\right) = \Pr(2\boldsymbol{\beta} + \boldsymbol{\delta} > \boldsymbol{\alpha}),$$

Proof. By the definition of $\mathcal{L}$, we can find:

$$\boldsymbol{\ell}_n^{'} = \boldsymbol{p}_n - \mathbf{y}_n, \tag{5}$$

As $\mathbf{y}_n \neq \mathbf{y}_m$, we have

$$\langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'}\rangle = \langle \boldsymbol{p}_n, \boldsymbol{p}_m\rangle - \boldsymbol{p}_{n,c_m} - \boldsymbol{p}_{m,c_n}$$

And

$$\langle \boldsymbol{p}_n, \boldsymbol{p}_m\rangle = \frac{1}{2}(||\boldsymbol{p}_n||^2 + ||\boldsymbol{p}_m||^2 - ||\boldsymbol{p}_n - \boldsymbol{p}_m||^2) = \frac{1}{2}(\boldsymbol{\alpha} - \boldsymbol{\delta})$$

which gives $\langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'}\rangle = \frac{1}{2}(\boldsymbol{\alpha} - \boldsymbol{\delta}) - \boldsymbol{\beta}$. When $2\boldsymbol{\beta} + \boldsymbol{\delta} > \boldsymbol{\alpha}$, we must have $\langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'}\rangle < 0$. According to Fact 1, we prove this theorem. □

In Fig. 4, we demonstrate the distribution of the inner product $\mathcal{S}_{n,m} = \langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'}\rangle$ for a 5-class classification task in several situations under the assumption that $\boldsymbol{p}_n, \boldsymbol{p}_m$ are random variables of two Dirichlet distributions, where we set $\mathbf{y}_n = (0, 0, 1, 0, 0),), \mathbf{y}_m = (0, 0, 0, 1, 0)$ and the model predictions: (a) $\boldsymbol{p}_n \sim Dir(0.05, 0.05, 0.45, 0.4, 0.05), \boldsymbol{p}_m \sim Dir(0.05, 0.05, 0.4, 0.45, 0.05)$, it simulates the scenario that two samples are close to the decision boundary of their true classes; (b) $\boldsymbol{p}_n \sim Dir(0.05, 0.05, 0.8, 0.05, 0.05), \boldsymbol{p}_m \sim Dir(0.05, 0.05, 0.4, 0.45, 0.05)$, it simulates the scenario that $\mathbf{x}_n, \mathbf{x}_m$ are away from and close to the decision boundary, respectively; (c) $\boldsymbol{p}_n \sim Dir(0.05, 0.05, 0.8, 0.05, 0.05), \boldsymbol{p}_m \sim Dir(0.05, 0.05, 0.05, 0.8, 0.05)$, it simulates the scenario that two samples are both away from the decision boundary. We see that $\mathcal{S}_{n,m}$ is negative with a much higher probability in Fig. 4a than in Figs. 4b and 4c, which consistent with the results in Fig. 1.

Theorem 2. *Suppose $\mathbf{y}_n = \mathbf{y}_m$, when $\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle \neq 0$, we have:*

$$sign(\langle \boldsymbol{g}_n, \boldsymbol{g}_m\rangle) = sign(\langle \mathbf{x}_n, \mathbf{x}_m\rangle),$$

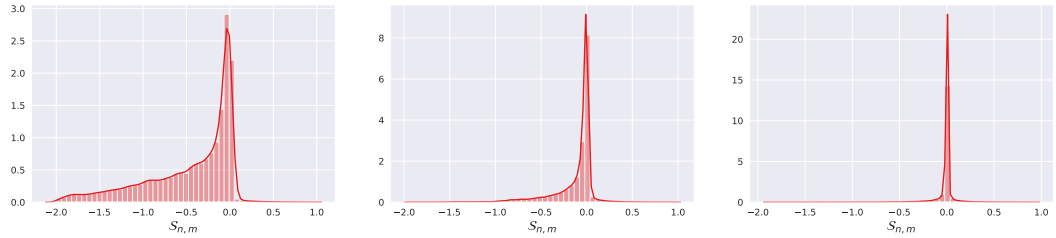
Proof. Because $\sum_{k=1}^{K} \boldsymbol{p}_{n,k} = 1, \boldsymbol{p}_{n,k} \geq 0, \forall k$, and $c_n = c_m = c$,

$$\langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'}\rangle = \sum_{k\neq c}^{K} \boldsymbol{p}_{n,k}\boldsymbol{p}_{m,k} + (\boldsymbol{p}_{n,c} - 1)(\boldsymbol{p}_{m,c} - 1) \geq 0 \tag{6}$$

According to Fact 1, we prove the theorem. □

B ALGORITHMS OF ONLINE MEMORY UPDATE

We provide the details of online ring buffer update and Balanced Experience Replay (BER) in Algs. 1 to 3. We directly load new data batches into the memory buffer without a separate buffer for the current task. The memory buffer works like a sliding window for each class in the data stream and we draw training batches from the memory buffer instead of directly from the data stream. In this case, one sample may not be seen only once as long as it stays in the memory buffer. This strategy is a more efficient use of the memory when $|\mathcal{B}| < n_c$, where $|\mathcal{B}|$ is the loading batch size of the data stream (i.e. the number of new samples added into the memory buffer at each iteration), we set $|\mathcal{B}|$ to 1 in all experiments (see Appx. E for a discussion of this).



(a) $\mathbf{x}_n, \mathbf{x}_m$ are both close to the decision boundary

(b) $\mathbf{x}_n, \mathbf{x}_m$ are away from and close to the decision boundary, respectively

(c) $\mathbf{x}_n, \mathbf{x}_m$ are both away from the decision boundary

Figure 4: Distribution of $\mathcal{S}_{n,m} = \langle \boldsymbol{\ell}_n^{'}, \boldsymbol{\ell}_m^{'} \rangle$ for a 5-class task under the assumptions: (a) the two samples are close to the decision boundary of their true classes and the model is reasonably trained, i.e. $\boldsymbol{p}_n, \boldsymbol{p}_m$ concentrate on two classes; (b) the model just gives random guessing, i.e. $\boldsymbol{p}_n, \boldsymbol{p}_m$ are uniformly distributed.

Algorithm 1 Ring Buffer Update with Fixed Buffer Size

Input: $\mathcal{B}_t$ - current data batch of the data stream, $\mathcal{C}_t$ - the set of classes in $\mathcal{B}_t$, $\mathcal{M}$ - memory buffer, $\mathcal{C}$ - the set of classes in $\mathcal{M}$, K - memory buffer size.
for c **in** $\mathcal{C}_t$ **do**
  Get $\mathcal{B}_{t,c}$ - samples of class c in $\mathcal{B}_t$,
  $\mathcal{M}_c$ - samples of class c in $\mathcal{M}$,
  if c **in** $\mathcal{C}$ **then**
    $\mathcal{M}_c = \mathcal{M}_c \cup \mathcal{B}_c$
  else
    $\mathcal{M}_c = \mathcal{B}_c, \quad \mathcal{C} = \mathcal{C} \cup \{c\}$
  end if
end for
$R = |\mathcal{M}| + |\mathcal{B}| - K$
while $R > 0$ **do**
  $c' = \arg\max_c |\mathcal{M}_c|$
  remove the first sample in $\mathcal{M}_{c'}, \quad R = R - 1$
end while
return $\mathcal{M}$

Algorithm 2 Balanced Experience Replay

Input: $\mathcal{M}$ - memory buffer, $\mathcal{C}$ - the set of classes in $\mathcal{M}$, B - training batch size, Θ - model parameters, $\mathcal{L}_\Theta$ - loss function, $\mathcal{B}_t$ - current data batch from the data stream, $\mathcal{C}_t$ - the set of classes in $\mathcal{B}_t$, K - memory buffer size.

$\mathcal{M} \leftarrow$ MemoryUpdate($\mathcal{B}_t, \mathcal{C}_t, \mathcal{M}, \mathcal{C}, K$)
$n_c, \mathcal{C}_s, \mathcal{C}_r \leftarrow$ ClassSelection($\mathcal{C}_t, \mathcal{C}, B$)
$\mathcal{B}_{train} = \emptyset$
for c **in** $\mathcal{C}_s$ **do**
  if c **in** $\mathcal{C}_r$ **then**
    $m_c = n_c + 1$
  else
    $m_c = n_c$
  end if
  Get $\mathcal{M}_c$ - samples of class c in $\mathcal{M}$,
  $\mathcal{B}_c \overset{m_c}{\sim} \mathcal{M}_c \triangleleft$ sample m_c samples from $\mathcal{M}_c$
  $\mathcal{B}_{train} = \mathcal{B}_{train} \cup \mathcal{B}_c$
end for
$\Theta \leftarrow$ Optimizer($\mathcal{B}_{train}, \Theta, \mathcal{L}_\Theta$)

C Related methods from DML

ρ-spectrum metric (Roth et al., 2020): $\rho = KL(\mathcal{U}||S_{\Phi_\mathcal{X}})$, which is proposed to measure the information entropy contained in the representation space. The ρ-spectrum computes the KL-divergence between a discrete uniform distribution $\mathcal{U}$ and the spectrum of data representations $S_{\Phi_\mathcal{X}}$, where $S_{\Phi_\mathcal{X}}$ is normalized and sorted singular values of $\Phi(\mathcal{X})$, Φ denotes the representation extractor (e.g. a neural network) and $\mathcal{X}$ is input data samples. Lower values of ρ indicate higher variance of the representations and hence more information entropy retained.

Multisimilarity(Wang et al., 2019): we adopt the loss function of Multisimilarity as an auxiliary objective in classfication tasks of continual learning, the batch mining process is omitted because we

Algorithm 3 Class Selection for BER

Input: $\mathcal{C}_t$ - the set of classes in current data batch $\mathcal{B}_t$, $\mathcal{C}$ - the set of classes in $\mathcal{M}$, B - training batch size, m_p - minimum number of positive pairs of each selected class ($m_p \in \{0, 1\}$) .
$\mathcal{B}_{train} = \emptyset$, $n_c = \lfloor B/|\mathcal{C}| \rfloor$, $r_c = B \mod |\mathcal{C}|$,
if $n_c > 1$ or $m_p == 0$ **then**
  $\mathcal{C}_r \overset{r_c}{\sim} \mathcal{C}$ $\lhd$ sample r_c classes from all seen classes without replacement.
  $\mathcal{C}_s = \mathcal{C}$
else
  $\mathcal{C}_r = \emptyset$, $n_c = 2$, $n_s = \lfloor B/2 \rfloor - |\mathcal{C}_t|$, $\lhd$ we ensure the training batch include samples from the current task.
  $\mathcal{C}_s \overset{n_s}{\sim} (\mathcal{C} - \mathcal{C}_t)$ $\lhd$ sample n_s classes from all seen classes except classes in $\mathcal{C}_t$.
  $\mathcal{C}_s = \mathcal{C}_s \bigcup \mathcal{C}_t$
  if $B \mod 2 > 0$ **then**
    $\mathcal{C}_r \overset{1}{\sim} \mathcal{C}_s$ $\lhd$ sample one class in $\mathcal{C}_s$ to have an extra sample.
  end if
end if
Return: $n_c, \mathcal{C}_s, \mathcal{C}_r$

use labels for choosing positive and negative pairs. So the loss function is $\widehat{\mathcal{L}} = \mathcal{L} + \lambda \mathcal{L}_{multi}$, and:

$$\mathcal{L}_{multi} = \frac{1}{B} \sum_{i=1}^{B} \left[\frac{1}{\alpha} \log[1 + \sum_{j \neq i, y_j = y_i} \exp\left(-\alpha(s_c(h_i, h_j) - \gamma)\right)] + \frac{1}{\beta} \log\left[1 + \sum_{y_j \neq y_i} \exp\left(\beta(s_c(h_i, h_j) - \gamma)\right)\right] \right] \tag{7}$$

where $s_c(\cdot, \cdot)$ is cosine similarity, α, β, γ are hyperparameters. In all of our experiments we set $\alpha = 2, \beta = 40, \gamma = 0.5$ as the same as in Roth et al. (2020).

R-Margin(Roth et al., 2020): we similarly deploy R-Margin for continual learning as the above, which uses the Margin loss (Wu et al., 2017) with the ρ regularization (Roth et al., 2020) as introduced in Sec. 2.2. So the loss function is $\widehat{\mathcal{L}} = \mathcal{L} + \lambda \mathcal{L}_{margin}$, and:

$$\mathcal{L}_{margin} = \sum_{i=1}^{B} \sum_{j=1}^{B} \gamma + \mathbb{I}_{j \neq i, y_j = y_i}(d(h_i, h_j) - \beta) - \mathbb{I}_{y_j \neq y_i}(d(h_i, h_j) - \beta) \tag{8}$$

where $d(\cdot, \cdot)$ is Euclidean distance, β is a trainable variable and γ is a hyperparameter. We follow the setting in Roth et al. (2020): $\gamma = 0.2$, the initialization of β is 0.6. We set $p_\rho = 0.2$ in ρ regularization.

D COMPARING TRAINING TIME

Tab. 5 compares the training time of MNIST tasks. All representation-based methods are much faster than gradient-based methods and close to the replay-based methods.

Table 5: Training time (in seconds) of the whole task sequence of MNIST tasks, which have been tested on a laptop with an 8-core Intel CPU and 32G RAM.

	DRL	BER	ER	A-GEM	GSS	Multisim	R-Margin
Permuted	12.48 ± 0.16	11.17 ± 0.18	$\mathbf{10.38 \pm 0.05}$	28.0 ± 0.09	33.98 ± 0.6	12.91 ± 0.13	13.45 ± 0.14
Split	5.6 ± 0.14	5.29 ± 0.08	$\mathbf{5.25 \pm 0.02}$	13.41 ± 0.47	19.07 ± 1.31	5.89 ± 0.09	6.29 ± 0.43

E HYPER-PARAMETERS IN EXPERIMENTS

To make a fair comparison of all methods, we use following settings: *i*) The configurations of GSS-greedy are as suggested in Aljundi et al. (2019), with batch size set to 10 and each batch receives 10 iterations. *ii*) For the other methods, we use the ring buffer memory as described in Alg. 1, the loading batch size is set to 1, following with one iteration, the training batch size is provided in Tab. 6. More hyperparameters are given in Tab. 6 as well.

In the setting of limited training data in online continual learning, we either use a small batch size or iterate on one batch several times to obtain necessary steps for gradient optimization. We chose a small batch size with one iteration instead of larger batch size with multiple iterations because by our memory update strategy (Alg. 1) it achieves similar performance with fewer hyperparameters. Since GSS-greedy has a different strategy for updating memories, we leave it at its default settings.

Table 6: Hyperparameters of all experiments

	Permuted MNIST	Split MNIST	Split Fashion	Split CIFAR-10	Split CIFAR-100
training batch size	20	20	20	10	10
learning rate	0.02	0.2	0.2	0.1	0.2
ref batch size (A-GEM)	256	256	256	256	1500
α of DRL	2	2	2	0.1	0.1
λ of DRL	1×10^{-2}	5×10^{-4}	5×10^{-4}	2×10^{-4}	2×10^{-5}
λ of Multisim	5	1	1	2	0.1
λ of R-Margin	2×10^{-5}	1×10^{-3}	1×10^{-3}	1×10^{-4}	2×10^{-4}