Continual Learning with Dual Regularizations

Xuejun Han[1]✉ and Yuhong Guo[1,2]✉

[1] Carleton University, Ottawa, Canada
[2] Canada CIFAR AI Chair, Amii
xuejunhan@cmail.carleton.ca, yuhong.guo@carleton.ca

Abstract. Continual learning (CL) has received a great amount of attention in recent years and a multitude of continual learning approaches arose. In this paper, we propose a continual learning approach with dual regularizations to alleviate the well-known issue of catastrophic forgetting in a challenging continual learning scenario – domain incremental learning. We reserve a buffer of past examples, dubbed memory set, to retain some information about previous tasks. The key idea is to regularize the learned representation space as well as the model outputs by utilizing the memory set based on interleaving the memory examples into the current training process. We verify our approach on four CL dataset benchmarks. Our experimental results demonstrate that the proposed approach is consistently superior to the compared methods on all benchmarks, especially in the case of small buffer size.

Keywords: Continual Learning · Representation Regularization · Functional Regularization.

1 Introduction

Ideally, the intelligent system should be capable of coping with and adapting to continually changing environments like humans. Unfortunately, most of existing machine learning algorithms are not provided with such ability. To address this shortcoming, a problem setting known as *continual learning* [18] or *lifelong learning* [22] came into being in recent years, prompting machine learners to behave more like humans by fast acquiring new knowledge without forgetting what has been learned in the past. In this setting, the learner is presented with a sequence of similar or dissimilar tasks and the goal is to train a learner to work well on all seen tasks.

Generally, continual learning can be split into three scenarios: *task incremental learning*, *domain incremental learning* and *class incremental learning* [7, 24]. In *task incremental learning*, the task identities are always available and hence it admits model architectures with task specific components such as a multi-head output layer. In contrast, the task identities are unknown at test time in *incremental domain learning*. Single-head networks are typically exploited for this scenario and the output layer stays the same with exposure to new tasks. Different from the above two scenarios, *class incremental learning* deals with the

circumstance where each task in the sequence only contains a subset of classes and new classes progressively emerge with new tasks arriving. In the case that the output spaces of sequential tasks can be treated as an identical setting, e.g., all binary tasks, class incremental learning has been viewed as a special case of domain incremental learning, where the domain shifts between different tasks are relatively substantial and the task identities are no long available.

The most critical issue in continual learning nevertheless is the *catastrophic forgetting* [11, 5] for previous learned tasks when the model is retrained for new ones. To address this problem, many continual learning methods have been developed. Some methods overcome the forgetting problem by regularizing the model parameters or outputs [8, 26, 9, 2, 1], which is known as *regularization-based methods*. Some methods resort to a memory set consisting of previous seen examples to preserve certain knowledge about past tasks [15, 19, 10, 3, 4, 17, 14, 13], which is called *memory-based methods*. These methods typically keep a constant network architecture during the overall learning phase. By contrast, *model-based methods* propose to revise the network architecture by dynamically adding new neurons for new tasks or revising some specific neurons for early tasks. However, such kind of methods may result in a network with excessive size as the number of tasks increases.

Among the exiting regularization-based and memory-based approaches, it is worth noting that some regularization-based models such as [8] can completely fail even on the simple Split MNIST dataset in the domain incremental learning scenario, while many memory-based models can work extremely well in the task incremental learning scenario but perform poorly in the domain incremental learning setting. Some memory-based models such as [10] behave well in both settings but are computational inefficient.

In this paper, we propose a novel continual learning approach for the domain incremental learning setting, which can be viewed as a hybrid of regularization and memory based methods and does not require much computational and memory cost. The key idea is to regularize the model outputs as well as the learned representation space by use of a set of past examples, known as memory set. First, we interleave memory examples with current data throughout model training to learn shared feature representations. On this basis, we enforce the model outputs on both current and memory data to be close to previous ones via knowledge distillation so as to preserve information about previous model and thereby overcome forgetting. Meanwhile, feature selection is implemented in the learned representation space to further minimize the domain discrepancy of current and memory datasets. Finally, mixup of current and memory data on the representation level is exploited to yield a beneficial effect on the generalization performance. We verify our approach on four dataset benchmarks – Split MNIST, Permuted MNIST, Split CIFAR-10 and Split CIFAR-100 and the proposed approach consistently outperforms the compared methods on all the benchmarks. Notably, our approach is empirically much more effective when the size of memory set is small.

2 Related work

Generally speaking, continual learning methods can be grouped into three categories: regularization-based methods, memory-based methods and model-based methods. Our approach is a combination of regularization-based and memory-based methods. We hence will briefly review prior work for these two categories.

Regularization-based methods. To alleviate the issue of catastrophic forgetting, the regularization-based methods equip the loss function with a regularization, either on weights or functions outpits, to protect the model from unwanted changes. *Elastic Weight Consolidation (EWC)* [8] penalizes the parameter changes in terms of the importance of parameters for old tasks measured by Fisher information matrix. In a similar way, *Synaptic Intelligence (SI)* [26] proposed a regularization penalty but in an online manner and along the entire training trajectory. *Online EWC* [20] is a modified variant of *EWC* by treating all old tasks equivalently which are able to gracefully forget old tasks when the model is out of capacity. Based on *Online EWC*, *Riemmanian Walk (RWalk)* [2] further replaces the Euclidean distance between parameters by KL divergence, i.e. distance in the Riemannian manifold. *Memory Aware Synapses (MAS)* [1] measures the importance of parameters based on the sensitivity to model predictions in an unsupervised and online setting. In addition to weight-regularization approaches, there also exists a group of functional-regularization methods, such as *Learning without Forgetting (LwF)* [9] which enforces the model outputs to be close to the previous ones by use of *knowledge distillation*.

Memory-based methods. The memory-based continual learning methods can be divided into two types based on whether interleaving memory examples with current data in the model training. *Gradient Episodic Memory (GEM)* [10] does not absorb the memory set into the training data. Specifically, It incorporates the memory set into the objective as $t-1$ inequality constraints where $t-1$ is the number of seen tasks, such that the loss on previous tasks cannot increase. *Averaged GEM (A-GEM)* [3] is a simplified version of *GEM* by combining $t-1$ task-specific constraints into one constraint for all old tasks together. In contrast, the manner that the memory data participates directly in the model training is named as *Experience Replay (ER)* [15, 19], which we believe makes the best use of all data on hand. [4] gave a comprehensive study for *experience replay* and evaluated a group of different memory selection strategies. Based on ER, many CL techniques were put forward. *Variational Continual Learning (VCL)* [12] extends online variational inference to deal with continual learning tasks. *Meta-Experience Replay (MER)* [17] combines experience replay with the optimization-based meta-learning. *Dark Experience Replay (DER)* [14] encourages the model outputs to mimic the previous ones on memory examples by minimizing their KL divergence. *Functional Regularization of Memorable Past (FROMP)* [13] regularizes the model outputs at a few memory examples with a Gaussian process formulation of deep neural networks. Moreover, instead of reserving a subset from previous datasets directly, *Deep Generative Replay (DGR)*

[21] and *Replay through Feedback (RtF)* [23] train generative models on old tasks and rehearsal pseudo-examples during new task training. However, it is difficult to make them work well on complicated datasets.

In a nutshell, the proposed approach falls into the category of a hybrid of regularization-based and memory-based CL methods, and is expected to integrate strengths of both.

3 Methodology

3.1 Problem Formulation

Continual learning copes with a sequence of similar or dissimilar tasks and the model is optimized on one task at a time without accessing to previous data in general. The goal is to train a model that works well on all tasks, which naturally involves a severe problem of catastrophic forgetting. In practice, a small buffer is realizable to preserve some information about old tasks, which is crucial to alleviate the forgetting problem. Formally, let the task index be $t \in \{1, \ldots, T\}$, with corresponding dataset $\mathcal{D}_t$. In addition, we keep a memory set $\mathcal{M}$ of size m consisting of examples from previous tasks, and allocate $m/(t-1)$ memories to each of the $t-1$ previous tasks. By default we assume $m \geq T$. Furthermore, to make the notation of memory set more explicitly, we denote the memory set used during task t by $\mathcal{M}_{t-1}$, which contains $t-1$ subsets, specifically, $\mathcal{M}_{t-1} = \bigcup_{i=1}^{t-1} \mathcal{M}_{t-1}^{i}$.

When the task t arrives, the model f_θ attempts to minimize the loss over current data $\mathcal{D}_t$. Out of sample efficiency, we also incorporate memory data $\mathcal{M}_{t-1}$ into the training dataset. Thus, the loss function is defined as follows,

$$\mathcal{L}_{CE} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_t \bigcup \mathcal{M}_{t-1}} l(f_\theta(\mathbf{x}), y) \tag{1}$$

where l is the classification loss function, in particular the cross-entropy loss.

It is worth noticing that the memory set is equally allocated to each seen tasks, so as to ensure there is as few as one example for each previous task. Notably, the joint training on current data and stored previous data falls into the scope of *experience repay* [15, 19], which was comprehensively explored by [4] and demonstrated to be a very simple and strong CL baseline.

3.2 Proposed Approach

The key idea of the proposed approach is to regularize the model outputs as well as the learned representation space by use of the memory set through knowledge distillation and feature selection, respectively. Moreover, the representation regularization and the memory set usage are further strengthened by deploying a manifold mixup component. In this section, we provide a progressive and comprehensible exposition of the proposed approach.

Knowledge Distillation The phenomenon of catastrophic forgetting usually occurs when the model changes drastically after retraining for new tasks. For this reason, preventing the model from dramatic drifts is one of coping strategies, which was investigated in [8, 26] by regularizing the model parameters. Besides, a more straightforward and effective alternative is to directly regularize the model outputs [9, 16, 13, 14], since what ultimately matters is model predictions rather than learned representations. To achieve this goal, it is effective to make the outputs of new model to be close to the ones of previous model by means of knowledge distillation.

We use $f_{\theta_{t-1}}$ to denote the prediction model the model after learning tasks $\{1 \ldots, t-1\}$. To preserve the knowledge about these previous tasks, a knowledge distillation loss can be incorporated into the objective of the classification loss:

$$\mathcal{L}_{KD} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_t \bigcup \mathcal{M}_{t-1}} l(f'_\theta(\mathbf{x}), f'_{\theta_{t-1}}(\mathbf{x})) \tag{2}$$

where l denotes the cross-entropy loss, $f'_\theta(\mathbf{x})$ and $f'_{\theta_{t-1}}(\mathbf{x})$ are modified versions of current and previous model predictions, respectively. Explicitly, let $h_\theta(\mathbf{x})$ and $h_{\theta_{t-1}}(\mathbf{x})$ be pre-softmax outputs (i.e. logits) of of $f_\theta(\mathbf{x})$ and $f_{\theta_{t-1}}(\mathbf{x})$ on example $\mathbf{x}$, then

$$f'_\theta(\mathbf{x}) = \text{softmax}(h_\theta(\mathbf{x})/\mathcal{T}) \tag{3}$$

where $\mathcal{T}$ is a temperature parameter to adjust soft target distributions. Same definition applies for $f'_{\theta_{t-1}}(\mathbf{x})$.

We empirically found that letting $\mathcal{T} = 2$ is slightly better than just leaving $\mathcal{T} = 1$, therefore, we will use $\mathcal{T} = 2$ during experiments in this paper, which aligns with the choices of [6, 9]. It is worth noting that, different from [9], we adopted a memory set and encouraged the similar predictions of current and old models not only on current data but also on memory examples. Moreover, [9] pointed out the similar performance between the cross-entropy loss and other reasonable loss functions for knowledge distillation, as long as trying to force the outputs of previous model to stay steady.

Feature Selection via Sparse Regularization The performance deterioration of old tasks arising from model retaining for new tasks is mainly caused by the domain drifts between the new and old tasks. By interleaving the memory data with current data in the training stage, the learned representation space are to some extent shared across the current and previous tasks. Ideally, the distribution gap between current and previous tasks is expected to be eliminated in the shared representation space. However, such aspiration is strenuous to realize especially when the memory set is small. Therefore, an elaborately filtration among features in the representation space is worthy of consideration to alleviate such issue. On this account, we incorporate a regularization on the classifier to implicitly select features that behave similarly in both current and previous tasks, so that in the selected feature space, the discrepancy between the current and previous task domains is able to be further diminished.

To begin with, let us rephrase the model parameters $\theta = [\theta^G, \theta^F]$, where θ^G and θ^F are parameters for the feature extractor G and the classifier F, respectively. The dimension of the feature space is d.

To realize feature selection, we first resort to a statistic – Pearson correlation coefficient (PCC) – that measures the correlation between two variables such as X and Y. The definition is as follows,

$$\rho_{X,Y} = \frac{\mathbb{E}[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

It is worth noticing that $\rho_{X,Y} \in [-1, 1]$ with a value of 1 if X and Y are perfectly correlated and -1 otherwise.

Here we aim to capture the correlation between features and the class labels via PCC, which can reflect the prediction power of the corresponding features. Specifically, we denote the PCC between the prediction of the classifier based on the feature ψ of datapoint $\mathbf{x}$ and the label y for all pairs $(\mathbf{x}, y)$ from $\mathcal{D}_t$ as $\rho_{\mathcal{D}_t}(F_\psi(G(\mathbf{x})), y)$, where F_ψ uses only the parameters related to feature ψ. The larger the value of $\rho_{\mathcal{D}_t}(F_\psi(G(\mathbf{x})), y)$, the greater the compatibility degree of the prediction based on the feature ψ and the label. Similarly, let us denote the PCC for all pairs in $\mathcal{M}_{t-1}$ as $\rho_{\mathcal{M}_{t-1}}(F_\psi(G(\mathbf{x})), y)$. Ideally, we desire to select features that are similarly predictive in both the current and memory datasets. For this purpose, let us define the *compatibility* of the prediction based on the feature ψ and the label in two datasets as the product $\rho_{\mathcal{D}_t}(F_\psi(G(\mathbf{x})), y) \cdot \rho_{\mathcal{M}_{t-1}}(F_\psi(G(\mathbf{x})), y)$. while the *incompatibility* is

$$\Delta(\psi) = 1 - \rho_{\mathcal{D}_t}(F_\psi(G(\mathbf{x})), y)\rho_{\mathcal{M}_{t-1}}(F_\psi(G(\mathbf{x})), y) \tag{4}$$

which represents the undesirableness of a specific feature ψ for bridging the domain discrepancy in terms of the classification problem. It is straightforward to see that $\Delta(\psi) \in [0, 2]$. By using the incompatibility values of features as weights for l_1 norm regularization, we obtain the following sparse regularizer of interest:

$$\mathcal{R}(\theta) = \sum_{\psi=1}^{d} \Delta(\psi)|\theta_\psi^F| \tag{5}$$

which enforces stronger regularization on more incompatible features so as to achieve the goal of selecting features that are more compatible across the current and previous tasks.

Manifold Mixup Enhancement We reserve a memory set to preserve some knowledge about previous tasks, however, such memory set merely provides partial information especially when the buffer size is small. In the selected representation space via feature selection, we expect the domain discrepancy between current and old tasks to be diminished. However, oscillations are inevitable when predicting the examples from old tasks outside the memory set. Out of concern for this issue, we recall a data augmentation technique – *mixup* [27], which is

Algorithm 1 Training Protocol

Input: dataset $\mathcal{D}_t$, memory set $\mathcal{M}_{t-1}$, previous model θ_{t-1}, batch size b, number of iterations num_iter, learning rate τ

Output: model θ

```
Initialize θ ← θ_{t−1}
for iter = 1 : num_iter do
    B_D ← random sample(D, b)
    B_M = ⋃_{i=1}^{t−1} B_{M_i} and B_{M_i} ← random sample(M_{t−1}^i, b/(t−1))
    B_mix = Mixup_G(B_D, B_M) acc. to Eq. (6).
    compute loss l acc. to Eq. (8)
    update θ ← θ − τ∇_θ L_θ
end for
return θ
```

to extend the training dataset with convex combinations of pairs of datapoints and their corresponding labels and was shown to have a favourable effect on generalization performance.

Different from vanilla mixup, we construct the mixup samples on a representation level by linearly interpolating between current and memory data representations, dubbed $\text{Mixup}_G(\mathcal{D}_t, \mathcal{M}_{t-1})$. Specifically,

$$\begin{aligned}(\mathbf{g}_{mix}, y_{mix}) &:= \text{Mixup}_G((\mathbf{x}_D, y_D), (\mathbf{x}_M, y_M)) \\ \mathbf{g}_{mix} &= \lambda G(\mathbf{x}_D) + (1-\lambda) G(\mathbf{x}_M) \\ y_{mix} &= \lambda y_D + (1-\lambda) y_M \end{aligned} \tag{6}$$

where $(\mathbf{x}_D, y_D) \sim \mathcal{D}_t$, $(\mathbf{x}_M, y_M) \sim \mathcal{M}_{t-1}$, $\lambda \sim Beta(\alpha_0, \alpha_0)$ and G is the feature extractor. We take $\alpha_0 = 3$ in the experiments. Notably, Eq. (6) can be viewed as a simplified variant of *manifold mixup* [25], where the mixup is performed in the hidden states at each layer of the network whereas ours is only in the representation space right before the classifier.

The classification loss computed on such mixup samples is defined as follows,

$$\mathcal{L}_{MM} = \mathbb{E}_{(\mathbf{x}_D, y_D) \sim \mathcal{D}_t, (\mathbf{x}_M, y_M) \sim \mathcal{M}_{t-1}} l(F(\mathbf{g}_{mix}), y_{mix}) \tag{7}$$

where $(\mathbf{g}_{mix}, y_{mix})$ is derived from Eq. (6) and F is the classifier of the proposed model. By embracing mixup samples between the memory set and current task, we expect smoother decision boundaries and benefits to classification performance when the undesirable test oscillation occurs. By performing mixup in the extracted feature space, we expect this mixup based classification loss can work together with the feature selection regularization above to bridge domain discrepancy and improve generalization performance.

Overall Learning Problem By integrating the knowledge distillation based classification loss in Eq. (2), the mixup based classification loss in Eq. (7), and the sparse regularization in Eq. (5) together, we obtain the following optimization

Table 1. The characteristics of datasets.

Datasets	#Tasks	#Classes/task	#Training/task	#Test/task
Split MNIST	5	2	12000	2000
Permuted MNIST	10	10	60000	10000
Split CIFAR-10	5	2	10000	2000
Split CIFAR-100	10	10	5000	1000

problem for the current task t:

$$\min_{\theta} \mathcal{L}_{\theta}(\mathcal{D}_t, \mathcal{M}_{t-1}) = \mathcal{L}_{CE} + \alpha \mathcal{L}_{KD} + \beta \mathcal{L}_{MM} + \gamma \mathcal{R}(\theta) \tag{8}$$

where α, β and γ are trade-off parameters. We solve it using a batch-wise gradient descent algorithm, which is summarized in Algorithm 1. After training for task t, the memory set $\mathcal{M}_{t-1}$ can be updated to $\mathcal{M}_t$ through task-wise random sampling before going to the next task.

4 Experiments

We conducted experiments on four benchmark continual learning datasets in the domain incremental learning scenario. In this section, we report the experimental setting and results.

4.1 Experimental Setting

Datasets We used four benchmark continual learning datasets: Split MNIST, Permuted MNIST, Split CIFAR-10, and Split CIFAR-100. Split MNIST [7] is generated by splitting the source MNIST dataset into five binary-class subsets in sequence (0/1, 2/3, 4/5, 6/7, 8/9). Permuted MNIST [7] is a variant of MNIST dataset, by applying a certain random pixel-level permutations to the entire MNIST dataset. We consider 10 tasks for this dataset. Split CIFAR-10 [16] is a sequential split of CIFAR-10 with five binary classification tasks. Split CIFAR-100 [16] has 10 ten-class classification tasks. The characteristics of all datasets are summarized in Table 1.

Comparison Methods We compared the proposed approach with nine comparison methods, two baselines and seven continual learning competitors: (1) *Joint* is a baseline that jointly trains on the data of all tasks. It serves as the upper bound for continual learning techniques. (2) *Finetune* is a baseline that simply fine-tunes the model from previous tasks on the current task dataset. (3) *EWC* is the regularization based continual learning method, Elastic Weight Consolidation [8]. (4) *SI* is another regularization based method, Synaptic Intelligence [26]. (5) *LwF* is a Learning without Forgetting method [9]. (6) *GEM*

Table 2. The average accuracy ± standard deviation (%) on test data of all tasks across 5 runs with different random seeds. The result of joint training, i.e. the upper bound, and the best accuracies of different buffer size are marked in bold. Note that '5t/10t' means after training 5/10 tasks.

Buffer	Model	MNIST		CIFAR-10	CIFAR-100	
		S-MNIST	P-MNIST	S-CIFAR10	S-CIFAR100-5t	S-CIFAR100-10t
	Joint	**98.59$_{\pm 0.15}$**	**97.90$_{\pm 0.09}$**	**90.67$_{\pm 0.22}$**	**51.30$_{\pm 0.42}$**	**43.80$_{\pm 0.83}$**
	Finetune	56.72$_{\pm 2.01}$	74.38$_{\pm 1.63}$	72.21$_{\pm 0.53}$	25.60$_{\pm 0.30}$	16.76$_{\pm 0.43}$
–	EWC	57.04$_{\pm 1.46}$	86.60$_{\pm 1.62}$	72.12$_{\pm 1.23}$	25.41$_{\pm 0.48}$	16.79$_{\pm 0.25}$
	SI	68.06$_{\pm 2.18}$	**93.59$_{\pm 0.77}$**	75.16$_{\pm 0.87}$	25.84$_{\pm 1.22}$	17.60$_{\pm 1.30}$
	LwF	**77.93$_{\pm 0.53}$**	53.25$_{\pm 1.84}$	**78.90$_{\pm 0.81}$**	**31.66$_{\pm 0.34}$**	**19.41$_{\pm 0.23}$**
200	GEM	91.08$_{\pm 0.70}$	86.67$_{\pm 0.48}$	79.26$_{\pm 0.31}$	28.29$_{\pm 1.60}$	18.72$_{\pm 0.37}$
	A-GEM	86.56$_{\pm 1.01}$	69.84$_{\pm 2.01}$	76.40$_{\pm 1.02}$	27.37$_{\pm 0.56}$	16.86$_{\pm 0.19}$
	ER-Res.	86.71$_{\pm 1.50}$	86.73$_{\pm 0.44}$	78.58$_{\pm 0.84}$	28.69$_{\pm 0.62}$	18.78$_{\pm 0.36}$
	FROMP	71.75$_{\pm 1.31}$	76.71$_{\pm 0.75}$	75.20$_{\pm 1.12}$	24.42$_{\pm 1.09}$	12.35$_{\pm 2.37}$
	Ours	**94.37$_{\pm 0.35}$**	**93.11$_{\pm 0.43}$**	**84.62$_{\pm 0.28}$**	**34.34$_{\pm 0.59}$**	**21.15$_{\pm 0.41}$**
500	GEM	94.67$_{\pm 0.32}$	92.89$_{\pm 0.86}$	81.83$_{\pm 0.61}$	32.04$_{\pm 0.77}$	20.88$_{\pm 0.31}$
	A-GEM	89.53$_{\pm 1.12}$	83.06$_{\pm 1.78}$	77.38$_{\pm 1.70}$	28.33$_{\pm 0.38}$	17.20$_{\pm 0.27}$
	ER-Res.	90.91$_{\pm 0.98}$	91.30$_{\pm 0.41}$	81.79$_{\pm 0.48}$	31.9$_{\pm 0.58}$	20.24$_{\pm 0.34}$
	FROMP	78.71$_{\pm 0.45}$	90.81$_{\pm 0.47}$	74.84$_{\pm 2.29}$	23.17$_{\pm 2.86}$	15.13$_{\pm 2.58}$
	Ours	**96.41$_{\pm 0.19}$**	**94.72$_{\pm 0.19}$**	**85.69$_{\pm 0.25}$**	**37.70$_{\pm 0.40}$**	**23.08$_{\pm 0.53}$**
1000	GEM	95.72$_{\pm 0.81}$	94.36$_{\pm 0.25}$	84.66$_{\pm 0.44}$	35.46$_{\pm 0.82}$	23.05$_{\pm 0.16}$
	A-GEM	95.31$_{\pm 1.48}$	88.34$_{\pm 0.41}$	79.12$_{\pm 0.47}$	28.21$_{\pm 0.39}$	17.37$_{\pm 0.27}$
	ER-Res.	95.09$_{\pm 0.22}$	93.00$_{\pm 0.11}$	83.77$_{\pm 0.37}$	34.06$_{\pm 0.54}$	22.23$_{\pm 0.23}$
	FROMP	88.34$_{\pm 1.11}$	93.09$_{\pm 0.09}$	74.75$_{\pm 2.85}$	24.04$_{\pm 3.36}$	16.62$_{\pm 2.88}$
	Ours	**97.35$_{\pm 0.25}$**	**95.60$_{\pm 0.18}$**	**86.72$_{\pm 0.30}$**	**40.20$_{\pm 0.52}$**	**25.62$_{\pm 0.22}$**

is the Gradient Episodic Memory method [10]. (7) *A-GEM* is a lightweight variant of GEM, Averaged GEM (A-GEM) [3]. (8) *ER-Reservoir* is an Experience Replay method based on reservoir sampling [4]. (9) *FROMP* is the Functional Regularization of Memorable Past method from [13] .

Architecture and Hyperparameter Selection As for model architectures, we employ a single-head multi-layer perceptron with two hidden layers following [23, 7, 24] for the MNIST datasets, and a CNN with 4 convolutional layers and 2 dense layers with dropouts following [26, 13] for CIFAR-10/100 datasets. For fair comparison, all approaches share the same model architectures. The hyperparameters are selected by a coarse grid search for all approaches and the best results are reported. For the proposed approach, we recommend $\alpha = 2$ and $\beta = 0.1$ for three Split datasets. For Permuted MNIST, we have $\alpha = 1$ and $\beta = 0.001$. For all datasets, $\gamma \in [10^{-4}, 10^{-1}]$. Generally, the greater buffer size results in smaller γ.

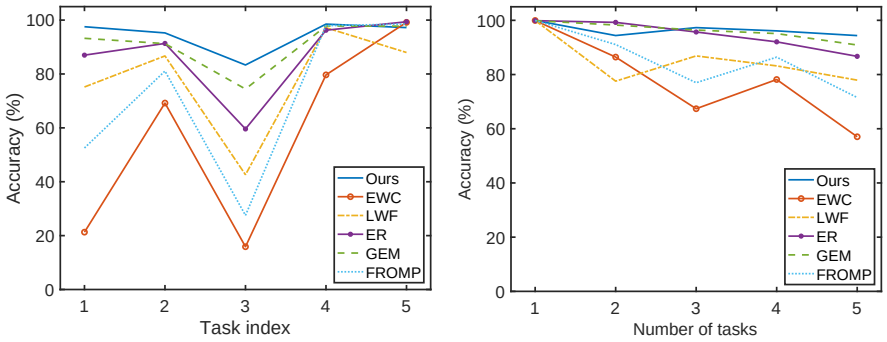

Fig. 1. Split MNIST dataset with buffer size of 200. **Left:** The test accuracy of each task after learning all tasks. **Right:** The evolution of average test accuracy of all seen tasks as more tasks are learned. All results are obtained via 5 runs with different random seeds.

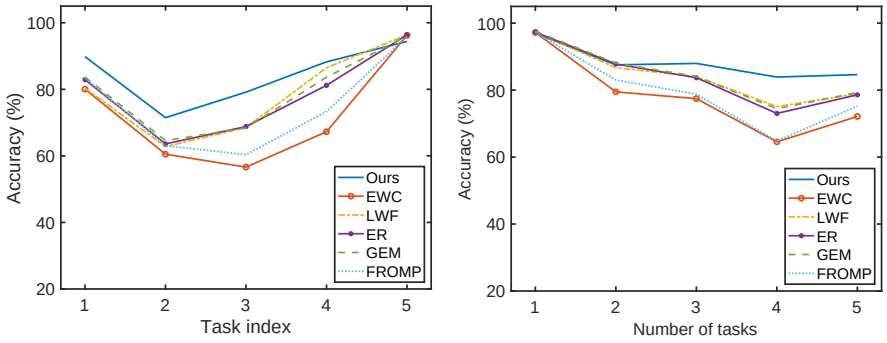

Fig. 2. Split CIFAR-10 dataset with buffer size of 200. **Left:** The test accuracy of each task after learning all tasks. **Right:** The evolution of average test accuracy of all seen tasks as more tasks are learned. All results are obtained via 5 runs with different random seeds.

4.2 Experimental Results

We experimented with three different buffer sizes, {200, 500, 1000}, for all benchmark datasets. Each experiment for each approach is repeated 5 runs with different random seeds. The random seeds work for the network initialization, training data shuffle and memories selection. The order of sequential tasks in Split MNIST and Split CIFAR-10/100 datasets remain unchanged throughout the experiments. For Permuted MNIST, once the random permutations for each task are generated, the task order is fixed across all runs. The results in terms of the average accuracies evaluated on test data of all tasks for the proposed approach and comparison methods are reported in Table 2. It is worth noting that our proposed approach achieves the best results compared to memory-based competitors across all dataset benchmarks over different buffer sizes and significantly outperforms these competitors when the buffer size is small, like 200. The pro-

Table 3. The average BWT ± standard deviation (%) across 5 runs with different random seeds. The best BWT of different buffer size are marked in bold. Note that the larger BWT means the better performance.

Buffer	Model	S-MNIST	S-CIFAR-10
–	EWC	-53.24 ± 1.81	-27.67 ± 1.53
	LwF	$\mathbf{-5.26 \pm 1.73}$	$\mathbf{-16.73 \pm 1.20}$
200	GEM	-9.49 ± 1.01	-18.71 ± 0.34
	ER-Res.	-16.15 ± 1.86	-19.42 ± 1.12
	FROMP	-34.72 ± 1.06	-22.63 ± 1.01
	Ours	$\mathbf{-2.29 \pm 0.61}$	$\mathbf{-8.08 \pm 0.29}$
500	GEM	-4.55 ± 0.43	-16.34 ± 0.71
	ER-Res.	-10.86 ± 1.25	-15.25 ± 0.37
	FROMP	-24.68 ± 1.33	-22.65 ± 2.02
	Ours	$\mathbf{-1.41 \pm 0.39}$	$\mathbf{-5.76 \pm 0.37}$

posed approach also outperforms the regularization-based methods in most cases except on the Permuted MNIST with buffer size 200, where the accuracy of the proposed approach is approximately 0.48% lower than *SI*. In the split datasets of either MNIST or CIFAR-10/100, *EWC* and *SI* perform poorly. This usually happens to most of parameter-regularization models in the domain incremental learning scenario. A memory set contrarily is favourable in this scenario. However, some memory-based models still produce poor performance when the buffer size is small, such as *FROMP* in most datasets and *A-GEM* in the permuted MNIST dataset with buffer size of 200. By contrast, our proposed approach demonstrate good performance across all datasets and scenarios.

Figure 1 and Figure 2 show the test accuracy results of the multiple tasks after and during continue learning on Split MNIST and Split CIFAR-10, respectively, when the buffer size is 200. Specifically, the figures on the left side of Figure 1 and Figure 2 report the test accuracy of each learned task after finishing the training on all tasks, while the figures on the right side report the evolution of average test accuracy of all seen tasks as new task arrives. It is clear to observe that after learning all tasks, the proposed approach still keep a competent memory of previously learned tasks compared to other competitors, where *EWC* and *FROMP* almost thoroughly forget task 1 and 3 of Split MNIST and also perform pretty poorly on task 3 of Split CIFAR-10. In the figures of evolution of average accuracy, the proposed approach maintain the best performance from task 3 of both datasets, which demonstrates the outstanding ability of our approach to overcome the catastrophic forgetting even with a small memory set.

Backward Transfer (BWT) In addition to test accuracy, another metric, called *Backward Transfer (BWT)* [10], has been used to measure the influence

Table 4. The effectiveness of knowledge distillation (KD), feature selection (FS) and manifold mixup (MM) on Split MNIST and Split CIFAR-10 datasets. CE represents the cross-entropy loss (Eq.1) over current and memory datasets. The average accuracy ± standard deviation (%) across 5 runs are reported.

S-MNIST				Buffer Size	
CE	KD	FS	MM	200	500
✓				88.71 ± 1.73	92.73 ± 0.51
✓	✓			93.03 ± 0.41	95.70 ± 0.33
✓	✓	✓		94.10 ± 0.35	96.27 ± 0.23
✓	✓	✓	✓	94.37 ± 0.35	96.41 ± 0.19

S-CIFAR-10				Buffer Size	
CE	KD	FS	MM	200	500
✓				76.05 ± 0.55	79.08 ± 0.53
✓	✓			83.06 ± 0.59	84.91 ± 0.29
✓	✓	✓		84.43 ± 0.28	85.44 ± 0.32
✓	✓	✓	✓	84.62 ± 0.28	85.69 ± 0.25

on previous tasks after learning the new task, which is defined as follows,

$$\text{BWT} = \frac{1}{T-1} \sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$$

where $R_{i,j}$ is the classification accuracy of the model on test data of task j after learning the training data from task i. The positive backward transfer means the improvement of performance on some previous tasks after learning new ones, whereas the negative backward transfer is known as forgetting. The larger BWT value indicates the better ability of the model to overcome forgetting. When two models have similar test accuracies, the one with larger BWT value would be preferable. Here we report our BWT results on the Split MNIST and Split CIFAR-10 datasets in Table 3. We can see that the proposed approach achieves much larger BWT values compared to other models on both datasets, which again validated its efficacy.

Discussion Following the literature work that uses memory set, we assume $m \geq T$ in this paper. What if the memory set is out of capacity with the increase of the number of tasks? This $m < T$ issue can be addressed by elaborately selecting memory examples or exploiting mixup to generate examples that are able to best stand for all past tasks.

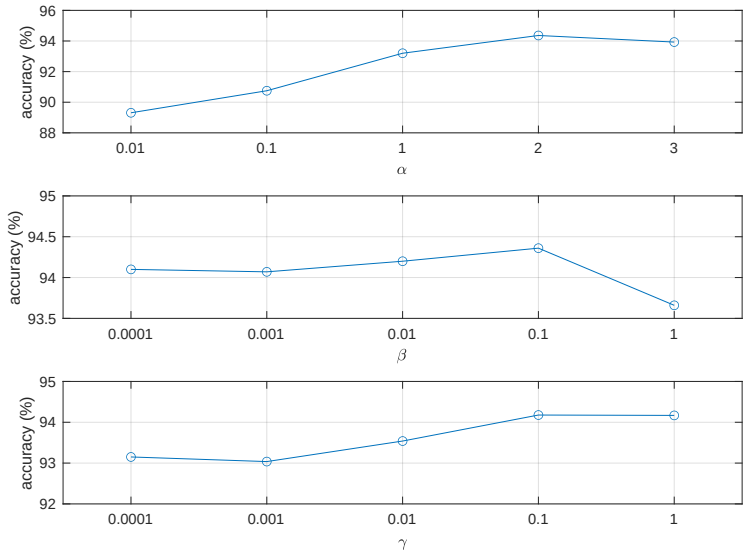

Fig. 3. Sensitivity to α, β and γ on Split MNIST dataset with the buffer size of 200.

4.3 Ablation Study

We further conducted an ablation study to verify the impact of each component in our proposed approach on the Split MNIST and Split CIFAR-10 datasets with the buffer size of 200 and 500. The results are presented in Table 4.

As demonstrated in [4], experience replay is a very simple and strong baseline, which here is treated as base model represented by 'CE' - the cross-entropy loss on current and memory datasets (Eq. (1)). We empirically observed that the experience replay with knowledge distillation (KD) is a much stronger method, which is also simple and effective and can beat the majority of elaborate CL techniques. Similar phenomenon was as well explored in [14]. As shown in Table 4, by adding the term of knowledge distillation, the performance obtains remarkable improvement. On the basis of knowledge distillation, we further implement the feature selection (FS) in the representation space by equipping model with a weighted l_1 regularization, so that only shared features across current and old tasks are utilized. Table 4 shows that by adding the feature regularization, the results get further improved. Finally, the simplified manifold mixup (MM) of current and memory data in the selected feature space is employed to further improve the generalization performance which is also verified by the slight but consistent improvement demonstrated in Table 4.

Hyperparameter Sensitivity Analysis We also conducted experiments on the Split MNIST dataset to evaluate the sensitivity of the proposed model to hyperparameters α, β and γ. When evaluating one hyperparameter, the values

of the other hyperparameters are fixed. The results with buffer size of 200 are presented in Figure 3. The parameter α represents the weight of knowledge distillation loss. The larger value of α means the greater degree of preserving previous knowledge. We investigated different α values from the set of $\{0.01, 0.1, 1, 2, 3\}$. We can see that with the increase of α value from 0.01, the performance increases due to the increasing contribution of knowledge distillation. However, when $\alpha = 3$ the performance degrades while $\alpha = 2$ yields the best result. This makes sense since that if the model focuses too much on precious tasks, it will lose the ability to adapt to a new task. Hence the value of α cannot be too large. The parameter β represents the weight of the classification loss from the mixup data. From Figure 3, we can observe that a reasonable value range for β is $[0.0001, 0.1]$ and the best result is produced when $\beta = 0.1$. The parameter γ is the weight of the feature selection regularization term. With buffer size as 200, the best result is achieved at $\gamma = 0.1$ in Figure 3. In general, a larger buffer size can result in a smaller suitable value of γ. This is due to the fact that with more previous samples from the memory set involved in the training loss, they will contribute more to the feature representation learning to naturally reduce the domain discrepancy.

5 Conclusions

In this paper, we proposed a simple but effective continual learning approach with representational and functional regularizations for the domain incremental learning scenario. To avoid catastrophic forgetting, we enforced the model predictions on both current and memory data to approach the previous ones by means of knowledge distillation. Additionally, to alleviate the domain discrepancy in the feature representation space between the old task domain and the current task domain, we further resorted to feature selection so as to minimize the domain gap in the selected representation space. Finally, mixup between current and memory data representations are incorporated to improve the generalization performance. We demonstrated the effectiveness of the proposed approach through extensive experiments on four benchmark continual learning datasets.

Acknowledgments. This research was supported in part by the NSERC Discovery Grant, the Canada Research Chairs Program, and the Canada CIFAR AI Chairs Program.

References

1. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: European Conf. on Comput. Vision (ECCV) (2018)
2. Chaudhry, A., Dokania, P.K., Ajanthan, T., Torr, P.H.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: European Conference on Computer Vision (ECCV) (2018)

3. Chaudhry, A., Ranzato, M., Rohrbach, M., Elhoseiny, M.: Efficient lifelong learning with a-gem. In: International Conference on Learning Representations (ICLR) (2019)
4. Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P.K., Torr, P.H., Ranzato, M.: Continual learning with tiny episodic memories. arXiv preprint arXiv:1902.10486 (2019)
5. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgeting in gradientbased neural networks. In: International Conference on Learning Representations (ICLR) (2014)
6. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning and Representation Learning Workshop (2015)
7. Hsu, Y.C., Liu, Y.C., Ramasamy, A., Kira, Z.: Re-evaluating continual learning scenarios: A categorization and case for strong baselines. In: NeurIPS Continual learning Workshop (2018)
8. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., Hadsell, R.: Overcoming catastrophic forgetting in neural networks. National Academy of Sciences (PNAS) **114**(13), 3521–3526 (2017)
9. Li, Z., Hoiem, D.: Learning without forgetting. In: European Conference on Computer Vision (ECCV) (2016)
10. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. In: Advances in Neural Information Processing Systems (NIPS) (2017)
11. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. Psychology of Learning and Motivation - Advances in Research and Theory **24**(C), 109 – 165 (1989)
12. Nguyen, C.V., Li, Y., Bui, T.D., Turner, R.E.: Variational continual learning. In: Int. Conf. on Learn. Repr. (ICLR) (2018)
13. Pan, P., Swaroop, S., Immer, A., Eschenhagen, R., Turner, R.E., Khan, M.E.: Continual deep learning by functional regularisation of memorable past. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
14. Pietro, B., Matteo, B., Angelo, P., Davide, A., Simone, C.: Dark experience for general continual learning: a strong, simple baseline. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)
15. Ratcliff, R.: Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. Psychological review **97 2**, 285–308 (1990)
16. Rebuffi, S., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
17. Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., Tesauro, G.: Learning to learn without forgetting by maximizing transfer and minimizing interference. In: Int. Conf. on Learning Representations (ICLR) (2019)
18. Ring, M.B.: Continual Learning in Reinforcement Environments. Ph.D. thesis, The University of Texas at Austin (1994)
19. Robins, A.: Catastrophic forgetting, rehearsal and pseudorehearsal. Connection Science **7**(2), 123–146 (1995)
20. Schwarz, J., Czarnecki, W., Luketina, J., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R., Hadsell, R.: Progress & compress: A scalable framework for continual learning. In: Int. Conf. on Machine Learn. (ICML) (2018)
21. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. In: Advances in Neural Information Processing Systems (NIPS) (2017)

22. Thrun, S.: A lifelong learning perspective for mobile robot control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (1994)
23. van de Ven, G.M., Tolias, A.S.: Generative replay with feedback connections as a general strategy for continual learning. arXiv preprint arXiv:1809.10635 (2018)
24. van de Ven, G.M., Tolias, A.S.: Three scenarios for continual learning. arXiv preprint arXiv:1904.07734 (2019)
25. Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Lopez-Paz, D., Bengio, Y.: Manifold mixup: Better representations by interpolating hidden states. In: International Conference on Machine Learning (ICML) (2019)
26. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. In: Int. Conf. on Machine Learn. (ICML) (2017)
27. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (ICLR) (2018)