Doubly Perturbed Task Free Continual Learning

Byung Hyun Lee[1], **Min-hwan Oh**[2], **Se Young Chun**[1,3,†]

[1]Department of Electrical and Computer Engineering, Seoul National University
[2]Graduate School of Data Science, Seoul National University
[3]INMC & IPAI, Seoul National University
ldlqudgus756@snu.ac.kr, minoh@snu.ac.kr, sychun@snu.ac.kr

Abstract

Task Free online Continual Learning (TF-CL) is a challenging problem where the model incrementally learns tasks without explicit task information. Although training with entire data from the past, present as well as *future* is considered as the gold standard, naive approaches in TF-CL with the current samples may be conflicted with learning with samples in the future, leading to catastrophic forgetting and poor plasticity. Thus, a proactive consideration of an unseen future sample in TF-CL becomes imperative. Motivated by this intuition, we propose a novel TF-CL framework considering future samples and show that injecting adversarial perturbations on both input data and decision-making is effective. Then, we propose a novel method named Doubly Perturbed Continual Learning (DPCL) to efficiently implement these input and decision-making perturbations. Specifically, for input perturbation, we propose an approximate perturbation method that injects noise into the input data as well as the feature vector and then interpolates the two perturbed samples. For decision-making process perturbation, we devise multiple stochastic classifiers. We also investigate a memory management scheme and learning rate scheduling reflecting our proposed double perturbations. We demonstrate that our proposed method outperforms the state-of-the-art baseline methods by large margins on various TF-CL benchmarks.

Introduction

Continual learning (CL) addresses the challenge of effectively learning tasks when training data arrives sequentially. A notorious drawback of deep neural networks in a continual learning is *catastrophic forgetting* (McCloskey and Cohen 1989). As these networks learn new tasks, they often forget previously learned tasks, causing a decline in performance on those earlier tasks. If, on the other hand, we restrict the update in the network parameters to counteract the catastrophic forgetting, the learning capacity for newer tasks can be hindered. This dichotomy gives rise to what is known as the *stability-plasticity dilemma* (Carpenter and Grossberg 1987; Mermillod, Bugaiska, and Bonin 2013). The solutions to overcome this challenge fall into three main strategies: regularization-based methods (Kirkpatrick et al. 2017; Jung et al. 2020; Wang et al. 2021), rehearsal-based methods (Lopez-Paz and Ranzato 2017; Shin et al. 2017; Shmelkov, Schmid, and Alahari 2017; Chaudhry et al. 2018b, 2021), and architecture-based methods (Mallya and Lazebnik 2018; Serra et al. 2018).

In Task Free CL (TF-CL) (Aljundi, Kelchtermans, and Tuytelaars 2019), the model incrementally learns classes in an online manner agnostic to the task shift, which is considered more realistic and practical, but more challenging setup than offline CL (Koh et al. 2022; Zhang et al. 2022). The dominant approach to relieve forgetting in TF-CL is memory-based approaches (Aljundi, Kelchtermans, and Tuytelaars 2019; Pourcel, Vu, and French 2022). They employ a small memory buffer to preserve a few past samples and replay them when training on a new task, but the restrictions on the available memory capacity highly degenerate performance on past tasks.

Recently, several works suggested evolving the data distribution in memory by perturbing memory samples (Wang et al. 2022; Jin et al. 2021). Meanwhile, flattening the weight loss landscape has also shown benefits in CL setups (Cha et al. 2020; Deng et al. 2021). However, most of the prior CL works primarily concentrated on past samples, often overlooking future samples. Note that many CL studies use "i.i.d. offline" as the oracle method of the best possible performance, not only for past and present data but also for future data. Therefore, incorporating unknown future samples in the CL model could be helpful in reducing forgetting and enhancing learning when training with real future samples.

In this work, we first demonstrate an upper bound for the TF-CL problem with unknown future samples, considering both adversarial input and weight perturbation, which has not been fully explained yet. Based on the observation, we propose a method, doubly perturbed continual learning (DPCL), addressing adversarial input perturbation with perturbed function interpolation and weight perturbation, specifically for classifier weights, through branched stochastic classifiers. Furthermore, we design a simple memory management strategy and adaptive learning rate scheduling induced by the perturbation. In experiments, our method significantly outperforms the existing rehearsal-based methods on various CL setups and benchmarks. Here is the summary of our contributions.

- We propose an optimization framework for TF-CL and

† Corresponding author.

show that it has an upper bound which considers the adversarial input and weight perturbations.

- Our proposed method, DPCL, uses perturbed function interpolation and branching stochastic classifiers for input and weight changes with perturbation-based memory management and adaptive learning rate.
- The proposed method outperforms baselines on various CL benchmarks and can be adapted to existing algorithms, consistently improving their performance.

Related Works

Continual Learning (CL)

CL seeks to retain prior knowledge while learning from sequential tasks exhibiting data distribution shifts. Most existing CL methods (Lopez-Paz and Ranzato 2017; Kirkpatrick et al. 2017; Chaudhry et al. 2018a; Zenke, Poole, and Ganguli 2017; Rolnick et al. 2019; Yoon et al. 2018; Mallya, Davis, and Lazebnik 2018; Hung et al. 2019) primarily focus on the offline setting, where the learner can repeatedly access task samples during training without time constraints under the distinct task definitions separating task sequences.

Task Free Continual Learning (TF-CL)

TF-CL (Aljundi, Kelchtermans, and Tuytelaars 2019; Jung et al. 2023; Pourcel, Vu, and French 2022) addresses more general scenario where the model incrementally learns classes in an online manner and the data distribution change arbitrarily without explicit task information. The majority of existing TF-CL approaches fall under rehearsal-based approaches (Aljundi, Kelchtermans, and Tuytelaars 2019; He et al. 2020; Wang et al. 2022). They store a small number of samples from previous data stream and later replay them alongside new mini-batch data. Thus, we focus on rehearsal-based methods due to their simplicity and effectiveness.

Recently, DRO (Wang et al. 2022) proposed to edit memory samples by adversarial input perturbation, making it gradually harder to be memorized. Raghavan and Balaprakash (2021) showed that the CL problem has an upper bound whose objective is to minimize with adversarial input perturbation, but it didn't fully consider the TF-CL setup. Meanwhile, Deng et al. (2021) demonstrated the effectiveness of applying adversarial weight perturbation on training and memory management for CL. To our best knowledge, it has not been investigated yet considering both input and weight perturbation simultaneously in TF-CL, and our work will propose a method that takes both into account.

Input and Weight Perturbations

Injecting input and weight perturbations into a standard training scheme is known to be effective for robustness and generalization by flattening the input and weight loss landscape. It is well known that flat input loss landscape is correlated to the robustness of performance of a network to input perturbations. In order to enhance robustness, adversarial training (AT) intentionally smooths out the input loss landscape by training on adversarially perturbed inputs. There are alternative approaches to flatten the loss landscape, through gradient regularization (Lyu, Huang, and Liang 2015; Ross and Doshi-Velez 2018), curvature regularization (Moosavi-Dezfooli et al. 2019), and local linearity regularization (Qin et al. 2019). Meanwhile, multiple studies have demonstrated the correlation between the flat weight loss landscape and the standard generalization gap (Keskar et al. 2017; Neyshabur et al. 2017). Especially, adversarial weight perturbation (Wu, Xia, and Wang 2020) effectively improved both standard and robust generalization by combining it with AT or other variants.

Problem Formulation

Revisiting Conventional TF-CL

We denote a sample $(x, y) \in X \times Y$, where $X \subseteq \mathbb{R}^d$ is the input space (or image space), $Y \subseteq \mathbb{R}^C$ is the label space, and C is the number of classes. A deep neural network to predict a label from an image can be defined as a function $h : X \to Y$, parameterized with θ and this learnable parameter θ can be trained by minimizing sample-wise loss $\ell(h(x;\theta), y)$. TF-CL is challenging due to varying data distribution $\mathcal{P}_t$ over iteration t, so the learner encounters a stream of data distribution $\{\mathcal{P}_t\}_{t=0}^{T}$ via a stream of samples $\{(x^t, y^t)\}_{t=1}^{T}$ where $(x^t, y^t) \sim \mathcal{P}_t$. Let us denote the sample-wise loss for (x^t, y^t) by $\mathcal{L}_t(\theta) = \ell(h(x^t;\theta), y^t)$. Then, TF-CL trains the network h in an online manner (Aljundi et al. 2019):

$$\theta^t \in \arg\min_{\theta} \mathcal{L}_t(\theta) \tag{1}$$

$$\text{subject to } \mathcal{L}_\tau(\theta) \leq \mathcal{L}_\tau(\theta^{t-1}); \forall \tau \in [0, \ldots, t-1].$$

Novel TF-CL Considering a Future Sample

Many CL studies regard the "i.i.d. offline" training as the oracle due to its consistently low loss not only for past and present, but also for future data. Thus, considering the future samples could enhance the performance in TF-CL setup. We first relax the constraints in (1) as a single constraint:

$$\frac{1}{t}\sum_{\tau=0}^{t-1} \mathcal{L}_\tau(\theta) \leq \frac{1}{t}\sum_{\tau=0}^{t-1} \mathcal{L}_\tau(\theta^{t-1}). \tag{2}$$

Secondly, we introduce an additional constraint with a nuisance parameter θ' considering a future sample,

$$\mathcal{L}_{t+1}(\theta') \leq \mathcal{L}_{t+1}(\theta). \tag{3}$$

Then, using Lagrangian multipliers, the TF-CL with the minimization in (1) with new constraints (2) and (3) will be

$$\theta^t \in \arg\min_{\theta} \; \mathcal{L}_t(\theta) + \frac{\lambda}{t}\sum_{\tau=0}^{t-1}(\mathcal{L}_\tau(\theta) - \mathcal{L}_\tau(\theta^{t-1})) + \rho(\mathcal{L}_{t+1}(\theta') - \mathcal{L}_{t+1}(\theta)) \tag{4}$$

where $\lambda > 0$ and $\rho > 0$ are Lagrangian multipliers.

Doubly Perturbed Task Free Continual Learning

Unfortunately, the future sample and most past samples are not available in TF-CL. Instead of minimizing the loss (4) directly, we minimize its surrogate independent of past and future samples. For this, we utilized the observation from Wu et al. (2019) and Ahn et al. (2021) that change of parameter in classifier is more significant than change in encoder.

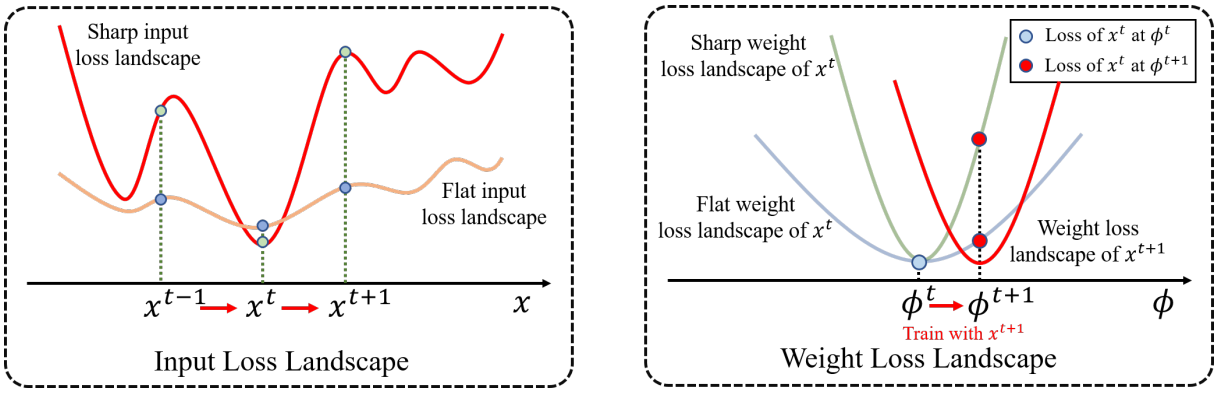


Figure 1: (Left) Input loss landscape of TF-CL when the weight θ^t has been determined for sample x^t. We desire $\ell(h(x;\theta^t),y)$ to be flat about x^t so that the loss for $x^\tau, \tau \in [1,\cdots,t-1,t+1]$ do not fluctuate significantly from x^t. (Right) Weight loss landscape of TF-CL where ϕ gets shifted from ϕ^t to ϕ^{t+1} by training for new sample x^{t+1}. We desire $\ell(h(x^t;[\theta_e;\phi]),y^t)$ to be flat about ϕ^t so that the loss for x^t doesn't increase dramatically when ϕ shifts from ϕ^t to ϕ^{t+1}.

Let us consider the network $h = g \circ f$ that consists of the encoder f and the classifier g, with the parametrization $h(\cdot;\theta) = g(f(\cdot;\theta_e);\phi)$ where $\theta = [\theta_e;\phi]$. Suppose that the new parameter $\theta' \approx [\theta_e;\phi']$ has almost no change in the encoder with the future sample (x^{t+1},y^{t+1}) while may have substantial change in the classifier. We also define $\eta_1^t := \max_\tau \|x^t - x^\tau\| < \infty, \tau = 0,\ldots,t-1,t+1$ and $\eta_2^t := \max_{\phi'} \|\phi' - \phi^t\|$. Then, we have a surrogate of (4).

Proposition 1. *Assume that $\mathcal{L}_t(\theta)$ is Lipschitz continuous for all t and ϕ' is updated with finite gradient steps from ϕ^t, so that ϕ' is a bounded random variable and $\eta_2^t < \infty$ with high probability. Then, the upper-bound for the loss (4) is*

$$\mathcal{L}_t(\theta) + \lambda \max_{\|\Delta x\| \le \eta_1^t} \mathcal{L}_{t,\Delta}(\theta) + \rho \max_{\|\Delta\phi\| \le \eta_2^t} \max_{\|\Delta x\| \le \eta_1^t} \mathcal{L}_{t,\Delta}([\theta_e;\phi^t+\Delta\phi]), \quad (5)$$

where $\mathcal{L}_{t,\Delta}(\theta) = \ell(h(x^t+\Delta x;\theta),y^t)$.

Proposition 1 suggests that injecting adversarial perturbation on input and classifier's weight could help to minimize the TF-CL loss (4). Note that both the second and third term have stably improved robustness and generalization of training (Ross and Doshi-Velez 2018; Wu, Xia, and Wang 2020). Here, η_1^t handles the data distribution shift. For example, more intense perturbation is introduced for better robustness with large η_1^t when crossing task boundaries.

Intuitively, such perturbations are known to find flat input and weight loss landscape (Madry et al. 2017; Foret et al. 2020). For the input, it is desirable to achieve low losses for both past and future samples with the current network weights. From Figure 1, a flatter input landscape is more conducive to achieving this goal. Moreover, if the loss of x^t is flat about weights, then one would expect only a minor increase in loss compared to a sharper weight landscape when the weights shift by training with new samples. Since directly computing the adversarial perturbations is inefficient due to additional gradient steps, we approximately minimize this *doubly perturbed* upper-bound (5) in an efficient way.

Efficient Optimization for Doubly Perturbed Task Free Continual Learning

In this section, we propose a novel CL method, called **D**oubly **P**erturbed **C**ontinual **L**earning (**DPCL**), which is inexpensive but very effective to handle the loss (5) with efficient input and weight perturbation schemes. We also design a simple memory management and adaptive learning rate scheme induced by these perturbation schemes.

Perturbed Function Interpolation

Minimizing the second term in (5) requires gradient for input, which is heavy computation for online learning. From Lim et al. (2022), we design a **P**erturbed **F**unction **I**nterpolation (**PFI**), a surrogate of the second term in (5). Let the encoder f consist of L-layered networks, denoted by $f = f_{(l+1):L} \circ f_{0:l}$, where $f_{0:l}$ maps an input to the hidden representation at lth layer (denoted by f^l) and $f_{(l+1):L}$ maps f^l to the feature space of the encoder f. We define the average loss for samples whose label is y as $\bar{\ell}_y = \sum_{\tau:y^\tau=y} \ell(h(x^\tau;\theta^\tau),y^\tau)/|\tau : y_\tau = y|$. Then, for a randomly selected lth layer of f, the hidden representation of a sample is perturbed by noise considering its label:

$$\check{f}^l = (\mathbf{1} + \mu_m \xi_m) \odot f^l + \mu_a \xi_a,\ \xi_m, \xi_a \sim \mathcal{N}(0,I), \quad (6)$$

where $\mathbf{1}$ denotes the one vector, $\odot$ is the Hadamard product, I is the identity matrix, $\mu_m = \sigma_m \tan^{-1}(\bar{\ell}_y)$, $\mu_a = \sigma_a \tan^{-1}(\bar{\ell}_y)$, and σ_m, σ_a are hyper-parameters. When label y is first encountered, we set $\mu_m = \sigma_m$, $\mu_a = \sigma_a$. Instead of computing true $\bar{\ell}_y$, it is updated by exponential moving average whenever a sample of label y is encountered.

As the main step, the function interpolation can be implemented for the two perturbed feature representations $\check{f}_i^l$ and $\check{f}_j^l$ with their labels y^i and y^j, respectively, as follows:

$$(\tilde{f}^l, \tilde{y}) = (\zeta \check{f}_i^l + (1-\zeta)\check{f}_j^l, \zeta y_i + (1-\zeta) y_j), \quad (7)$$

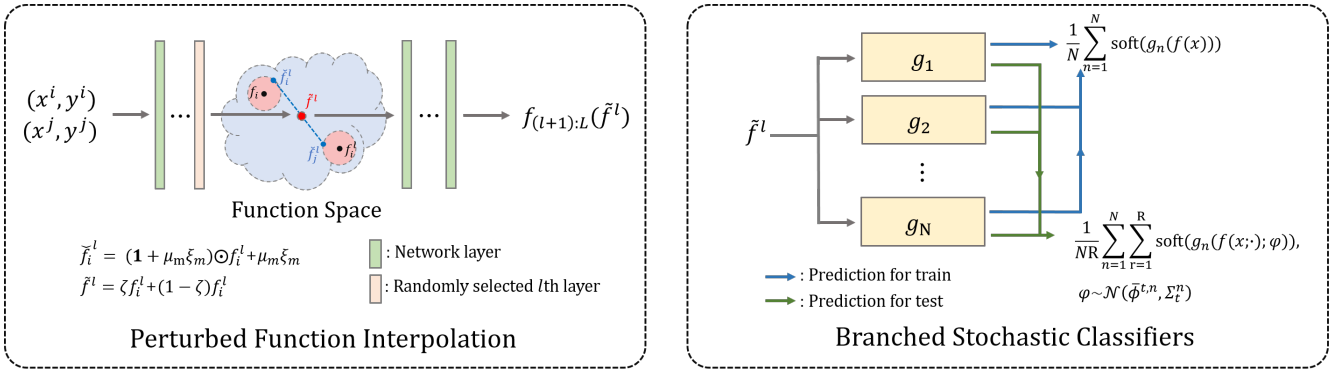


Figure 2: Illustration of Perturbed Function Interpolation (PFI) and Branched Stochastic Classifiers (BSC). PFI randomly perturbs the input, which makes the input loss landscape smooth. For weight perturbation, branched stochastic classifier utilizes weight average along the training trajectory, introduces multiple classifiers, and conduct variational inference during test.

where $\zeta \sim Beta(\alpha, \beta)$, $Beta(\cdot, \cdot)$ is the beta distribution. Finally, the output is computed by $f_{(l+1):L}(\tilde{f}^l)$ and we denote this as $\tilde{f}(\cdot)$. Since PFI only requires element-wise multiplication and addition with samplings from simple distributions, its computational burden is negligible.

Lim et al. (2022) has shown that using perturbation and Mixup in function space can be interpreted as the upper bound of adversarial loss for the input under simplifying assumptions including that the task is binary classification. We extend it to multi-class classification setup assuming the classifier is linear for each class node and its node is trained in terms of binary classification. Let $\tilde{\mathcal{L}}_\tau(\theta) = \ell(g(\tilde{f}(x^\tau)), y^\tau)$ is the loss computed by PFI.

Proposition 2. *Assume that $\mathcal{L}_\tau(\theta)$ is computed by binary classifications for multi-classes. We also suppose $\|\nabla_\theta h(x^\tau;\theta)\| > 0$, $d_1 \le \|f_\tau^l\|_2 \le d_2$ for some $0 < d_1 \le d_2$. With more regularity assumptions in Lim et al.(2022),*

$$\tilde{\mathcal{L}}_\tau(\theta) \ge \max_{\|\delta\|\le\epsilon} \ell(h(x^\tau+\delta;\theta), y^\tau) + \mathcal{L}_\tau^{\text{reg}} + \epsilon^2\psi_1(\epsilon), \quad (8)$$

where $\psi_1(\epsilon) \to 0$ and $\mathcal{L}_\tau^{\text{reg}} \to 0$ as $\epsilon \to 0$, and ϵ is assumed to be small and determined by each sample.

In Proposition 2, both $\mathcal{L}_\tau^{reg}$ and $\epsilon^2\psi_1(\epsilon)$ are negligible for small ϵ and the adversarial loss term becomes dominant in the right side of (8), which validates the use of PFI. See the supplementary materials for the details on Proposition 2.

Branched Stochastic Classifiers

Minimizing the third term in (5) requires additional gradient steps. We bypass this inspired by ideas from Izmailov et al. (2018), Maddox et al. (2019), and Wilson and Izmailov (2020), updating multiple models by averaging their weights along training trajectories and performing variational inference for decisions of these models. Cha et al. (2021) confirmed that such stochastic weight averaging effectively flattens the weight loss landscape compared to the adversarial perturbation. In this work, we use this solely for the classifier, termed **B**ranched **S**tochastic **C**lassifiers (**BSC**).

We first consider a single linear classifier g, represented as $g = [g_1, \dots g_C]$ where g_c is the sub-classifier connected to the node for the cth class parameterized by ϕ_c. Considering the setup of TF-CL, we independently apply the weight perturbation to g_c. Assume that g_c at iteration t follows a multivariate Gaussian distribution with mean $\bar{\phi}_c^t$ and covariance Σ_t^c. With the iteration T_c when the model first encounters the class c, $\bar{\phi}_c^t$ and Σ_t^c are updated every P iterations:

$$\bar{\phi}_c^t = \frac{k_c\bar{\phi}_c^{(t-P)} + \phi_c^t}{k_c+1}, \Sigma_t^c = \frac{1}{2}\Sigma_{diag,t}^c + \frac{D_{t,c}^T D_{t,c}}{2(A-1)}, \quad (9)$$

where $k_c = \lfloor (t-T_c)/P \rfloor$ with floor function $\lfloor\cdot\rfloor$, $\Sigma_{diag,t}^c = diag(\overline{(\phi_c^t)^{\cdot 2}} - (\bar{\phi}_c^t)^{\cdot 2})$, $diag(v)$ is the diagonal matrix with diagonal v, and $(\cdot)^{\cdot 2}$ is the element-wise square. For (9), $\overline{(\phi_c^t)^{\cdot 2}}$ and $D_{t,c} \in \mathbb{R}^{q\times A}$ are updated as:

$$\overline{(\phi_c^t)^{\cdot 2}} = \frac{k_c\overline{(\phi_c^{t-P})^{\cdot 2}} + (\phi_c^t)^{\cdot 2}}{k_c+1},$$
$$D_{t,c} = [D_{t-P,c}[2:A]\ (\phi_c^t - \bar{\phi}_c^t)], \quad (10)$$

where $D[2:A] \in \mathbb{R}^{q\times(A-1)}$ is the submatrix of D obtained by removing the first column of D. For inference, we predict the class probability p^t using the variational inference with $\bar{\phi}^t = [\bar{\phi}_1^t \cdots \bar{\phi}_C^t]$ and $\Sigma_t = diag([\Sigma_{t,1} \cdots \Sigma_{t,C}])$ by

$$p^t = \frac{1}{R}\sum_{r=1}^{R} \text{soft}(g(f(x;\theta_e^t);\varphi)), \quad \varphi \sim \mathcal{N}(\bar{\phi}^t, \Sigma_t), \quad (11)$$

where $\text{soft}(\cdot)$ is the softmax function. We can view Σ_t as the low-rank measures on deviation of the classifier parameters and the samplings as weight perturbation.

Wilson and Izmailov (2020) verified that both deep ensembles and weight moving average can effectively improve the generalization performance. Since training multiple networks is time-consuming, we instead introduce multiple linear classifiers. With the decisions of the classifiers, the final decision for an input is determined by $\bar{p}^t = \frac{1}{N}\Sigma_{n=1}^N p_n^t$, where N is the number of classifiers. In this case, each classifier can be viewed as an instance with perturbed weights.

Methods	CIFAR100 (M=2K)		CIFAR100-SC (M=2K)		ImageNet-100 (M=2K)	
	ACC↑	FM↓	ACC↑	FM↓	ACC↑	FM↓
ER (Rolnick et al. 2019)	36.87±1.53	44.98±0.91	40.09±0.62	30.30±0.60	22.35±0.29	51.87±0.24
EWC++ (Chaudhry et al. 2018a)	36.35±1.62	44.23±1.21	39.87±0.93	29.84±1.04	22.28±0.45	51.50±1.42
DER++ (Buzzega et al. 2020)	39.34±1.01	40.97±2.37	41.54±1.79	29.82±2.26	25.20±2.06	52.16±3.26
BiC (Wu et al. 2019)	36.64±1.73	44.46±1.24	38.63±1.32	29.96±1.54	22.41±1.23	50.94±1.34
MIR (Aljundi et al. 2019a)	35.13±1.35	45.97±0.85	37.84±0.86	31.55±1.00	22.75±1.03	52.65±0.85
CLIB (Koh et al. 2022)	37.48±1.27	42.66±0.69	37.27±1.63	30.04±1.85	23.85±1.36	49.96±1.69
ER-CPR (Cha et al. 2020)	40.98±0.12	44.47±0.45	41.93±0.42	30.67±0.39	27.08±3.26	49.93±1.06
FS-DGPM (Deng et al. 2021)	38.03±0.58	39.90±0.39	37.03±0.57	31.05±1.63	25.73±1.68	49.32±2.03
DRO (Ye and Bors 2022)	39.23±0.74	41.57±0.25	39.86±0.95	27.76±0.77	27.68±1.23	39.96±0.87
ODDL (Wang et al. 2022)	41.49±1.38	40.01±0.52	40.82±1.16	29.06±1.87	27.54±0.63	41.23±1.06
DPCL	**45.27±1.32**	**37.66±1.18**	**45.39±1.34**	**26.57±1.63**	**30.92±1.17**	**37.33±1.53**

Table 1: Results on CIFAR100, CIFAR100-SC, and ImageNet-100. The tasks are distinguished by disjoint sets of classes. For all datasets, we measured averaged accuracy (ACC) and forgetting measure(FM) (%) averaged by 5 different random seeds.

Perturbation-Induced Memory Management and Adaptive Learning Rate

Several studies showed the benefits of advanced memory management (Chrysakis and Moens 2020) and adaptive learning rate (Koh et al. 2022) in TF-CL. To take the same advantage, we propose **P**erturbation-**I**nduced **M**emory management and **A**daptive learning rate (**PIMA**). For memory management, we basically balance the number of samples for each class in the memory $\mathcal{M}_t$ and compute the sample-wise mutual information empirically for a sample (x, y) as

$$\mathbb{I}(x; \phi^t) = \mathbb{H}(\bar{p}^t) - \frac{1}{N}\sum_{n=1}^{N} \mathbb{H}(p_n^t), \tag{12}$$

where $\mathbb{H}(\cdot)$ is the entropy for class distribution. To manage $\mathcal{M}_t$, we introduce a history H_t which stores the mutual information for memory samples. Let $H_t(x, y)$ be the accumulated mutual information for a memory sample (x, y) at t. If (x, y) is selected for training, $H_t(x, y)$ is updated by

$$H_t(x, y) = (1 - \gamma)H_{t-1}(x, y) + \gamma\mathbb{I}(x; \phi^t), \tag{13}$$

where $\gamma \in (0, 1)$. Otherwise, $H_t(x, y) = H_{t-1}(x, y)$. To update the samples in the memory, we first identify the class $\hat{y}$ that occupies the most in $\mathcal{M}_t$. We then compare the values in $\{H_t(x, y)|(x, y) \in \mathcal{M}_t, y = \hat{y}\}$ with $\mathbb{I}(x^t; \phi^t)$ for the current stream sample (x^t, y^t). If $\mathbb{I}(x^t; \phi^t)$ is the smallest, we skip updating the memory. Otherwise, we remove the memory sample of the lowest accumulated mutual information.

We also propose a heuristic but effective adaptive learning rate induced by H_t. Whenever $\bar{\phi}_c^t$ is updated, we scale the learning rate η_t by a factor $\omega > 1$ if $\mathbb{E}_{(x,y)\sim\mathcal{M}_t}[H_t(x, y)] > \mathbb{E}_{(x,y)\sim\mathcal{M}_t}[H_{t-1}(x, y)]$ or $\frac{1}{\omega} < 1$ otherwise. The algorithm for the memory management and adaptive learning rate are explained in the supplementary materials.

Experiments

Experimental Setups

Benchmark datasets. We evaluate on three CL benchmark datasets. **CIFAR100** (Rebuffi et al. 2017) and **CIFAR100-SC** (Yoon et al. 2019) contains 50,000 samples and 10,000 samples for training and test. **ImageNet-100** (Douillard et al. 2020) is a subset of ILSVRC2012 with 100 randomly selected classes which consists of about 130K samples for training and 5000 samples for test. For both CIFAR100 and CIFAR100-SC, we split 100 classes into 5 tasks by randomly selecting 20 classes for each task (Rebuffi et al. 2017) and we considered the semantic similarity for CIFAR100-SC (Yoon et al. 2019). For Imagenet-100, we split 100 classes into 10 tasks by randomly selecting 10 classes for each task (Douillard et al. 2020).

Task configurations. We also considered various setups: disjoint, blurry (Bang et al. 2021), and i-blurry (Koh et al. 2022) setups. **Disjoint** task configuration is the conventional CL setup where any two tasks don't share common classes. As a more general configuration, the **blurry** setup involves learning the same classes for all tasks while having different class distributions per task. Meanwhile, each task in **i-blurry** setup consists of both shared and disjoint classes, which is more realistic than the disjoint and blurry setups.

Baselines. Since most of TF-CL methods are rehearsal-based methods, we compared our DPCL with **ER** (Rolnick et al. 2019), **EWC++** (Chaudhry et al. 2018a), **BiC** (Wu et al. 2019), **DER++** (Buzzega et al. 2020), and **MIR** (Aljundi et al. 2019a). By EWC++, we combined ER and the work of Chaudhry et al. (2018a). We compared DPCL with **CLIB** (Koh et al. 2022), which was proposed for i-blurry CL setup. We also experimented **DRO** (Wang et al. 2022), which proposed to perturb the memory samples via distributionally robust optimization. Lastly, we experimented **FS-DGPM** (Deng et al. 2021), CPR (Cha et al. 2020) by combining it with ER (**ER-CPR**), and **ODDL** (Ye and Bors 2022).

Evaluation metrics. We employ two primary evaluation metrics: averaged accuracy **(ACC)** and forgetting measure **(FM)**. ACC is a commonly used metric for evaluating CL methods (Chaudhry et al. 2018a; Han et al. 2018; van de Ven and Tolias 2018). FM is used to measure how much the average accuracy has dropped from the maximum value for a task (Yin, Li et al. 2021; Lin et al. 2022). The details for the metric is explained in the supplementary materials.

Implementation details. The overall experiment setting is

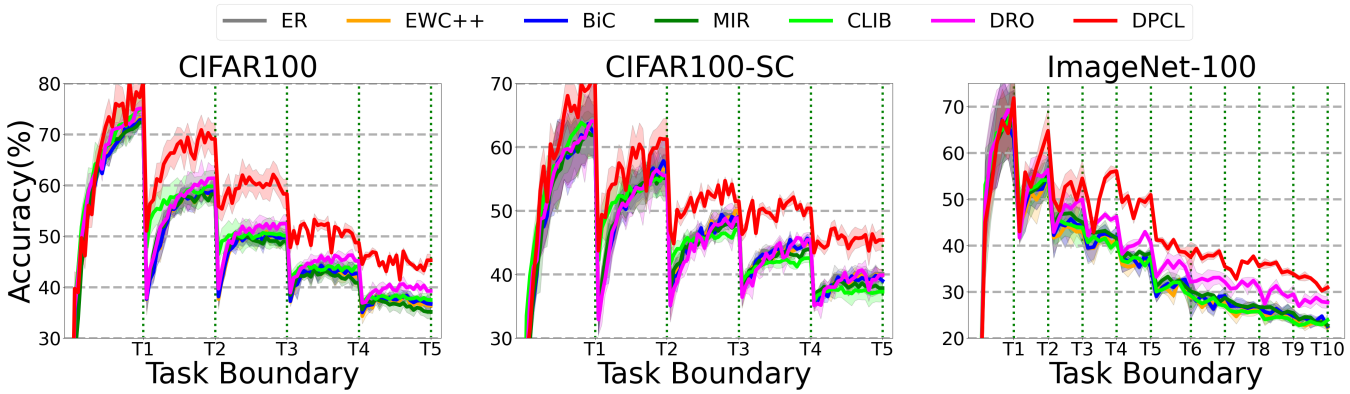


Figure 3: Any-time inference results on CIFAR100, CIFAR100-SC, and ImageNet-100. Each point represents average accuracy over 5 different random seeds and the shaded area represents the standard deviation(±) around the average accuracy.

Methods	Blurry (M=2K)		i-Blurry (M=2K)	
	ACC↑	FM↓	ACC↑	FM↓
ER	24.24±1.30	20.64±2.50	39.43±1.09	15.45±1.48
EWC++	23.84±1.57	20.67±3.34	38.55±0.79	15.57±2.36
DER++	24.50±3.03	17.35±4.24	44.34±0.67	13.14±4.64
BiC	24.96±1.82	20.12±3.78	39.57±0.90	14.23±2.19
MIR	25.15±0.08	15.49±2.07	38.26±0.63	15.12±2.69
CLIB	38.13±0.73	**4.69±0.99**	47.04±0.89	11.69±2.12
ER-CPR	28.72±1.67	18.67±1.23	42.59±0.66	18.01±2.68
FS-DGPM	29.72±0.22	14.51±2.82	41.99±0.65	11.81±0.12
DRO	20.86±2.45	17.11±2.47	41.78±0.42	11.97±2.79
ODDL	33.35±1.09	15.12±1.98	39.71±1.32	16.12±1.65
DPCL	**47.58±2.75**	11.44±2.64	**50.22±0.39**	**11.49±2.54**

Table 2: Results on various setups on CIFAR100. We measured averaged accuracy (ACC) and forgetting measure (FM) (%) averaged by 5 different seeds.

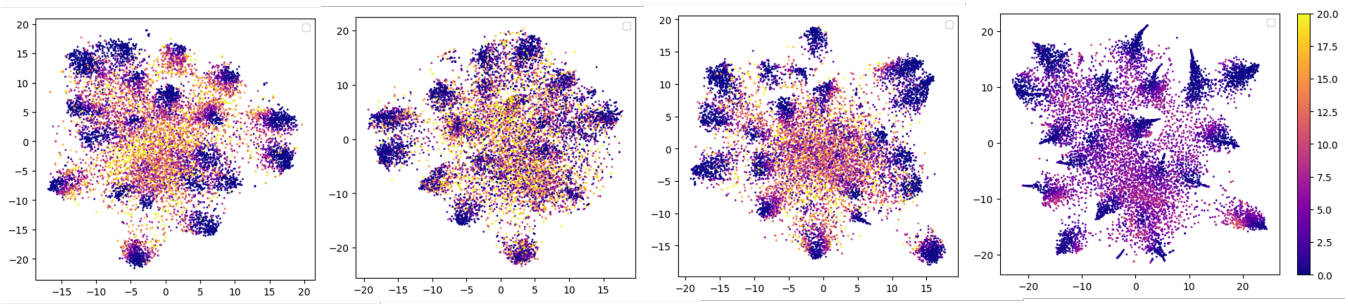
Methods	One-step (s)	Tr. Time (s)	Model Size	GPU Mem.
ER	0.012	7126	1.00	1.00
EWC++	0.027	16402	2.00	1.23
DER++	0.019	11384	1.00	1.53
BiC	0.015	10643	1.01	1.02
MIR	0.029	18408	1.00	1.96
CLIB	0.061	32519	1.00	4.32
ER-CPR	0.015	8082	1.00	1.00
DRO	0.038	23389	1.00	3.46
ODDL	0.032	22905	2.14	3.31
DPCL	0.017	10925	1.03	1.06

Table 3: Results of runtime/parametric complexity. One-step (s): one-step throughput in second. Tr. Time (s): total training time in second. Model Size: normalized model size. GPU Mem.: normalized training GPU memory.

based on Koh et al. (2022). We used ResNet34 (He et al. 2016) as the base feature encoder for all datasets. We used a batch size of 16 and 3 updates per sample for CIFAR100 and CIFAR100-SC and batch size of 64 and 0.25 updates per sample for ImageNet-100. We used a memory size of 2000 for all datasets. We utilized the Adam optimizer (Kingma and Ba 2015) with an initial learning rate of 0.0003 and applied an exponential learning rate scheduler except CLIB and the optimization configurations reported in the original papers were used for CLIB. We applied AutoAugment (Cubuk et al. 2019) and CutMix (Yun et al. 2019). For DRO, we conducted the CutMix separately for the samples from stream buffer and memory since it conflicts with the perturbation for the memory samples. Since both utilizing CutMix and our PFI is ambiguous, we didn't apply CutMix for our method. More information for the implementation details can be found in the supplementary materials.

Main Results

Results on various benchmark datasets We first conducted experiments on CIFAR100, CIFAR100-SC, and ImageNet-100 in a disjoint setup. As shown in Table 1, DPCL significantly improves the performance on all three datasets. The extent of improvement of EWC++, BiC, and MIR was marginal compared to ER, as already observed in other studies (Raghavan and Balaprakash 2021; Ye and Bors 2022). ER-CPR, FS-DGPM, DRO, and ODDL are perturbation-based methods and achieved high performance on all datasets among baselines. Interestingly, we observed that for CIFAR100-SC, ER outperformed several baselines. CIFAR100-SC is divided into tasks based on their parent classes, and it seems that some existing advanced methods have limitations to improve upon ER's performance for the dataset in the challenging TF-CL scenario. On the other hand, our proposed method significantly outperformed ER for all datasets. Recently, (Koh et al. 2022) argued that inference should be conducted at any-time for TF-CL since the model is agnostic to the task boundaries in practice. Based on this argument, we present the results of our method in terms of any-time inference in Fig. 3, where the vertical grids indicate task boundaries. From Fig. 3, our method consistently exhibits the best performance regardless of the iterations during training.



(a) ER (b) CLIB (c) DRO (d) DPCL

Figure 4: t-SNE on the features at the end of the encoder with CIFAR100. We computed the features and losses for samples in first task after training the last 5th task. The color represents the loss of a sample (yellow for high loss and purple for low loss). We can see that our DPCL has overall low loss for all regions, especially near the class boundaries.

Results on various task configurations. We evaluated our method on various task configurations. For this purpose, we evaluated the performance on CIFAR100 under the blurry (Bang et al. 2021) and i-blurry setup (Koh et al. 2022). From Table 2, we can observe that ER, EWC, BiC, and MiR show similar performance. CLIB, which is designed for the i-blurry setup, achieved the highest performance among the baselines. In contrast, DRO showed low performance in the blurry setup. On the other hand, our method consistently outperformed the baseline methods by a significant margin in both the blurry and i-blurry setups. Since there exists class imbalance in blurry or i-blurry settings, it empirically shows the robustness of the proposed method to class imbalance.

Runtime/Parametric complexity analysis. In Table 3, we evaluated runtime and parametric complexity of the baselines and DPCL. One-step throughput (One-step) and total training time (Training Time) were measured in second for CIFAR100. We also measured model size (Model Size) and training GPU memory (GPU Mem.) on ImageNet-100 after normalizing their values for ER as 1.0. We didn't consider the memory consumption for replay memory since it is constant for all methods. From Table 3, we can see that our proposed method introduces mild increase on runtime, model size, and training GPU memory compared to other CL methods. Note that DRO, the gradient-based perturbation method, has significantly increased both the training time and memory consumption.

Qualitative analysis. Fig. 4 shows the t-SNE results for samples within the first task after training the last task under the disjoint setup on CIFAR100. On the t-SNE map, we marked each samples with shade that represents the magnitude of the loss values for the corresponding features. From Fig. 4, we can observe that our method produces much smoother loss landscape for features compared to baselines. It verifies that our method indeed flattens loss landscape in function space. For more experiments for the loss landscape, please refer to the supplementary materials.

Ablation study. To understand the effect of each component in our DPCL, we conducted an ablation study. We measured the performance of the proposed method by removing each component from DPCL. As shown in Table 4, we observed the obvious performance drop when removing each component, indicating the efficacy of each component of our method. Furthermore, to demonstrate the orthogonality of PFI and BSC to baselines, we applied them to other baselines such as BiC and CLIB. For this, we excluded the previously applied CutMix. Table 4 confirms that the two components of our method can easily be combined with other methods to enhance performance.

Methods	CIFAR100 (Disjoint)	
	ACC↑	FM↓
DPCL w/o PFI	40.85±1.68	42.29±2.32
DPCL w/o BSC	41.75±1.43	40.56±1.94
DPCL w/o PIMA	42.94±1.23	39.79±2.23
DPCL	45.27±1.32	37.66±1.18
BiC	36.64±1.73	44.46±1.24
BiC w/ PFI and BSC	43.63±0.74	38.57±1.67
CLIB	37.48±1.27	42.66±0.69
CLIB w/ PFI and BSC	44.97±0.97	37.78±1.24

Table 4: Ablation studies on CIFAR100. PFI and BSC were applied to BiC and CLIB. PIMA was excluded since it conflicts with the baseline methods.

Conclusion

In this work, we proposed a novel optimization framework for Task Free CL (TF-CL) and showed that it has an upper-bound which addresses the input and weight perturbations. Based on the framework, we proposed a method, Doubly Perturbed Continual Learning (DPCL), which employs perturbed function interpolation and incorporates branched stochastic classifiers for weight perturbations, with an upper-bound analysis considering adversarial perturbations. By additionally proposing simple scheme of memory management and adaptive learning rate, we could effectively improve the baseline methods on TF-CL. Experimental results validated the superiority of DPCL over existing methods across various CL benchmarks and setups.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. NRF-2022R1A4A1030579, 2022R1C1C100685912, NRF-2022M3C1A309202211), by Creative-Pioneering Researchers Program through Seoul National University, and by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)].

References

Ahn, H.; Kwak, J.; Lim, S.; Bang, H.; Kim, H.; and Moon, T. 2021. Ss-il: Separated softmax for incremental learning. *ICCV*, 844–853.

Aljundi, R.; Caccia, L.; Belilovsky, E.; Caccia, M.; Lin, M.; Charlin, L.; and Tuytelaars, T. 2019a. Online Continual Learning with Maximally Interfered Retrieval. *NeurIPS*, 11849–11860.

Aljundi, R.; Kelchtermans, K.; and Tuytelaars, T. 2019. Task-free continual learning. *CVPR*, 11254–11263.

Aljundi, R.; Lin, M.; Goujaud, B.; and Bengio, Y. 2019. Gradient based sample selection for online continual learning. *NeurIPS*.

Bang, J.; Kim, H.; Yoo, Y.; Ha, J.-W.; and Choi, J. 2021. Rainbow memory: Continual learning with a memory of diverse samples. *CVPR*, 8218–8227.

Buzzega, P.; Boschini, M.; Porrello, A.; Abati, D.; and Calderara, S. 2020. Dark experience for general continual learning: a strong, simple baseline. *NeurIPS*, 33: 15920–15930.

Carpenter, G. A.; and Grossberg, S. 1987. ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied optics*, 26(23): 4919–4930.

Cha, J.; Chun, S.; Lee, K.; Cho, H.-C.; Park, S.; Lee, Y.; and Park, S. 2021. Swad: Domain generalization by seeking flat minima. *NeurIPS*, 22405–22418.

Cha, S.; Hsu, H.; Hwang, T.; Calmon, F.; and Moon, T. 2020. CPR: Classifier-Projection Regularization for Continual Learning. *ICLR*.

Chaudhry, A.; Dokania, P. K.; Ajanthan, T.; and Torr, P. H. 2018a. Riemannian walk for incremental learning: Understanding forgetting and intransigence. *ECCV*, 532–547.

Chaudhry, A.; Gordo, A.; Dokania, P.; Torr, P.; and Lopez-Paz, D. 2021. Using hindsight to anchor past knowledge in continual learning. *AAAI*, 35: 6993–7001.

Chaudhry, A.; Ranzato, M.; Rohrbach, M.; and Elhoseiny, M. 2018b. Efficient Lifelong Learning with A-GEM. *ICLR*.

Chrysakis, A.; and Moens, M.-F. 2020. Online continual learning from imbalanced data. *ICML*, 1952–1961.

Cubuk, E. D.; Zoph, B.; Mane, D.; Vasudevan, V.; and Le, Q. V. 2019. Autoaugment: Learning augmentation strategies from data. *CVPR*, 113–123.

Deng, D.; Chen, G.; Hao, J.; Wang, Q.; and Heng, P.-A. 2021. Flattening sharpness for dynamic gradient projection memory benefits continual learning. *NeurIPS*, 34: 18710–18721.

Douillard, A.; Cord, M.; Ollion, C.; Robert, T.; and Valle, E. 2020. Podnet: Pooled outputs distillation for small-tasks incremental learning. *ECCV*, 86–102.

Foret, P.; Kleiner, A.; Mobahi, H.; and Neyshabur, B. 2020. Sharpness-aware Minimization for Efficiently Improving Generalization. *ICLR*.

Han, B.; Yao, Q.; Yu, X.; Niu, G.; Xu, M.; Hu, W.; Tsang, I.; and Sugiyama, M. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *NeurIPS*, 31.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. *CVPR*, 770–778.

He, X.; Sygnowski, J.; Galashov, A.; Rusu, A. A.; Teh, Y. W.; and Pascanu, R. 2020. Task Agnostic Continual Learning via Meta Learning. *4th Lifelong Machine Learning Workshop at ICML*.

Hung, C.-Y.; Tu, C.-H.; Wu, C.-E.; Chen, C.-H.; Chan, Y.-M.; and Chen, C.-S. 2019. Compacting, picking and growing for unforgetting continual learning. *NeurIPS*, 32.

Izmailov, P.; Wilson, A.; Podoprikhin, D.; Vetrov, D.; and Garipov, T. 2018. Averaging weights leads to wider optima and better generalization. *UAI*, 876–885.

Jin, X.; Sadhu, A.; Du, J.; and Ren, X. 2021. Gradient-based editing of memory examples for online task-free continual learning. *NeurIPS*, 34: 29193–29205.

Jung, D.; Lee, D.; Hong, S.; Jang, H.; Bae, H.; and Yoon, S. 2023. New Insights for the Stability-Plasticity Dilemma in Online Continual Learning. *ICLR*.

Jung, S.; Ahn, H.; Cha, S.; and Moon, T. 2020. Continual learning with node-importance based adaptive group sparse regularization. *NeurIPS*, 3647–3658.

Keskar, N. S.; Mudigere, D.; Nocedal, J.; Smelyanskiy, M.; and Tang, P. T. P. 2017. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *ICLR*.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. *ICLR*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13): 3521–3526.

Koh, H.; Kim, D.; Ha, J.-W.; and Choi, J. 2022. Online Continual Learning on Class Incremental Blurry Task Configuration with Anytime Inference. *ICLR*.

Lim, S. H.; Erichson, N. B.; Utrera, F.; Xu, W.; and Mahoney, M. W. 2022. Noisy Feature Mixup. *ICLR*.

Lin, S.; Yang, L.; Fan, D.; and Zhang, J. 2022. Beyond not-forgetting: Continual learning with backward knowledge transfer. *NeurIPS*, 35: 16165–16177.

Lopez-Paz, D.; and Ranzato, M. 2017. Gradient episodic memory for continual learning. *NeurIPS*, 30.

Lyu, C.; Huang, K.; and Liang, H.-N. 2015. A unified gradient regularization family for adversarial examples. *ICDM*, 301–309.

Maddox, W. J.; Izmailov, P.; Garipov, T.; Vetrov, D. P.; and Wilson, A. G. 2019. A simple baseline for bayesian uncertainty in deep learning. *NeurIPS*, 32.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv:1706.06083*.

Mallya, A.; Davis, D.; and Lazebnik, S. 2018. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. *ECCV*, 67–82.

Mallya, A.; and Lazebnik, S. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. *CVPR*, 7765–7773.

McCloskey, M.; and Cohen, N. J. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*, 24: 109–165.

Mermillod, M.; Bugaiska, A.; and Bonin, P. 2013. The stability-plasticity dilemma: investigating the continuum from catastrophic forgetting to age-limited learning effects. *Frontiers in psychology*, 4: 504.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Uesato, J.; and Frossard, P. 2019. Robustness via curvature regularization, and vice versa. *CVPR*, 9078–9086.

Neyshabur, B.; Bhojanapalli, S.; McAllester, D.; and Srebro, N. 2017. Exploring generalization in deep learning. *NeurIPS*, 30.

Pourcel, J.; Vu, N.-S.; and French, R. M. 2022. Online Task-free Continual Learning with Dynamic Sparse Distributed Memory. *ECCV*, 739–756.

Qin, C.; Martens, J.; Gowal, S.; Krishnan, D.; Dvijotham, K.; Fawzi, A.; De, S.; Stanforth, R.; and Kohli, P. 2019. Adversarial robustness through local linearization. *NeurIPS*, 32.

Raghavan, K.; and Balaprakash, P. 2021. Formalizing the generalization-forgetting trade-off in continual learning. *NeurIPS*, 34: 17284–17297.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. *CVPR*, 2001–2010.

Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.; and Wayne, G. 2019. Experience replay for continual learning. *NeurIPS*, 32.

Ross, A.; and Doshi-Velez, F. 2018. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. *AAAI*, 32.

Serra, J.; Suris, D.; Miron, M.; and Karatzoglou, A. 2018. Overcoming catastrophic forgetting with hard attention to the task. *ICLR*, 4548–4557.

Shin, H.; Lee, J. K.; Kim, J.; and Kim, J. 2017. Continual learning with deep generative replay. *NeurIPS*, 30.

Shmelkov, K.; Schmid, C.; and Alahari, K. 2017. Incremental learning of object detectors without catastrophic forgetting. *ICCV*, 3400–3409.

van de Ven, G. M.; and Tolias, A. S. 2018. Three continual learning scenarios and a case for generative replay. *arXiv preprint arXiv:1904.07734*.

Wang, L.; Zhang, M.; Jia, Z.; Li, Q.; Bao, C.; Ma, K.; Zhu, J.; and Zhong, Y. 2021. Afec: Active forgetting of negative transfer in continual learning. *NeurIPS*, 22379–22391.

Wang, Z.; Shen, L.; Fang, L.; Suo, Q.; Duan, T.; and Gao, M. 2022. Improving task-free continual learning by distributionally robust memory evolution. *ICML*, 22985–22998.

Wilson, A. G.; and Izmailov, P. 2020. Bayesian deep learning and a probabilistic perspective of generalization. *NeurIPS*, 33: 4697–4708.

Wu, D.; Xia, S.-T.; and Wang, Y. 2020. Adversarial weight perturbation helps robust generalization. *NeurIPS*, 33: 2958–2969.

Wu, Y.; Chen, Y.; Wang, L.; Ye, Y.; Liu, Z.; Guo, Y.; and Fu, Y. 2019. Large scale incremental learning. *CVPR*, 374–382.

Ye, F.; and Bors, A. G. 2022. Task-Free Continual Learning via Online Discrepancy Distance Learning. *NeurIPS*.

Yin, H.; Li, P.; et al. 2021. Mitigating forgetting in online continual learning with neuron calibration. *NeurIPS*, 34: 10260–10272.

Yoon, J.; Kim, S.; Yang, E.; and Hwang, S. J. 2019. Scalable and Order-robust Continual Learning with Additive Parameter Decomposition. *ICLR*.

Yoon, J.; Yang, E.; Lee, J.; and Hwang, S. J. 2018. Lifelong Learning with Dynamically Expandable Networks. *ICLR*.

Yun, S.; Han, D.; Oh, S. J.; Chun, S.; Choe, J.; and Yoo, Y. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. *ICCV*, 6023–6032.

Zenke, F.; Poole, B.; and Ganguli, S. 2017. Continual learning through synaptic intelligence. *ICML*, 3987–3995.

Zhang, Y.; Pfahringer, B.; Frank, E.; Bifet, A.; Lim, N. J. S.; and Jia, A. 2022. A simple but strong baseline for online continual learning: Repeated Augmented Rehearsal. *NeurIPS*.