ONLINE PLACEBOS FOR CLASS-INCREMENTAL LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

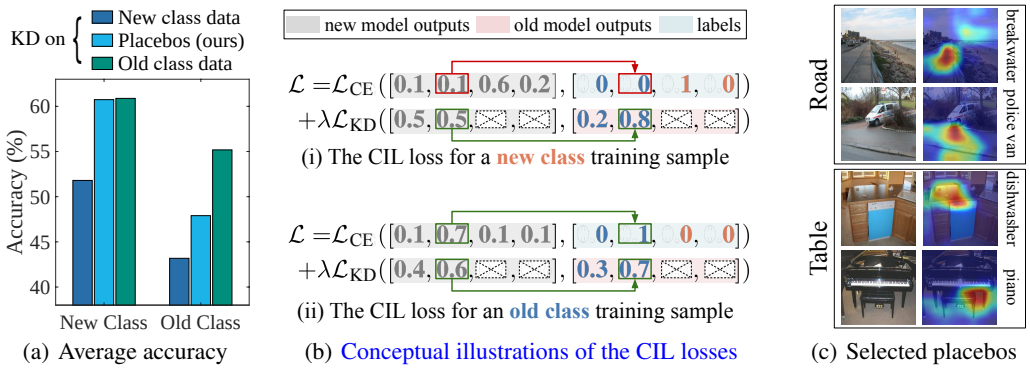
Not forgetting old class knowledge is a key challenge for class-incremental learning (CIL) when the model continuously adapts to new coming classes. A common technique to address this is knowledge distillation (KD) which penalizes prediction inconsistencies between old and new models. Such prediction is made with almost new class data, as old class data is extremely scarce due to the strict memory limitation in CIL. In this paper, we take a deep dive into KD losses and find that "using new class data for KD" not only hinders the model adaption (for learning new classes) but also results in low efficiency for preserving old class knowledge. We address this by "using the placebos of old classes for KD", where the placebos are chosen from a free image stream, such as Google Images, in an automatical and economical fashion. To this end, we train an online placebo selection policy to quickly evaluate the quality of streaming images (good or bad placebos) and use only good ones for one-time feed-forward computation of KD. We formulate the policy training process as an online Markov Decision Process (MDP), and introduce an online learning algorithm to solve this MDP problem without causing much computation costs. In experiments, we show that our method 1) is surprisingly effective even when there is no class overlap between placebos and original old class data, 2) does not require any additional supervision or memory budget, and 3) significantly outperforms a number of top-performing CIL methods, in particular when using lower memory budgets for old class exemplars, e.g., five exemplars per class. *The code is available in the supplementary.*

1 INTRODUCTION

AI learning systems are expected to learn new concepts while maintaining the ability to recognize old ones. In many practical scenarios, they cannot access the past data due to the limitations such as storage or data privacy but are expected to be able to recognize all seen classes. A pioneer work (Rebuffi et al., 2017) formulated this problem in the class-incremental learning (CIL) pipeline: training samples of different classes are loaded into the memory phase-by-phase, and the model keeps on re-training with new class data (while discarding old class data) and is evaluated on the testing data of both new and old classes. The key challenge is that re-training the model on the new class data tends to override the knowledge acquired from the old classes (McCloskey & Cohen, 1989; McRae & Hetherington, 1993; Ratcliff, 1990), and the problem is called "catastrophic forgetting". To alleviate this problem, most of CIL methods (Rebuffi et al., 2017; Hou et al., 2019; Douillard et al., 2020; Liu et al., 2020a; 2021a; Zhu et al., 2021) are equipped with knowledge distillation (KD) losses that penalize any feature and/or prediction inconsistencies between the models in adjacent phases.

The ideal KD losses should be computed on old class data since the teacher model (i.e., the model in the last phase) was trained on them. This is, however, impossible in the CIL setting, where almost all old class data are inaccessible in the new phase. Existing methods have to use new class data as a substitute to compute KD losses. We argue that this 1) hampers the learning of new classes as it distracts the model from fitting the ground truth labels of new classes, and 2) can not achieve the ideal result of KD, as the model can not generate the same soft labels (or features) on new class data as on old class data. We justify this from an empirical perspective as shown in Figure 1 (a):

We made the corresponding changes in the revised paper and colorized these changes in blue.



(a) Average accuracy (b) Conceptual illustrations of the CIL losses (c) Selected placebos

Figure 1: **(a) Average accuracy when computing KD loss on different data** using iCaRL (Rebuffi et al., 2017) on CIFAR-100. The KD loss (softmax KL divergence loss) is computed on new class data (**dark blue**), placebos (of old class data) selected by our method (**light blue**), and old class data **green**), i.e., the ideal case. **(b) Conceptual illustrations of the loss problem if using new class data for KD.** The **dark blue** and **orange** numbers denote the predictions of old and new classes, respectively. It is clear in (i) that the objectives are different when using a new class sample for KD (the oracle case is to have both "**ascent**"), e.g., the ground truth label for the second old class is 0, while the "KD label" at this position is 0.8. This is not an issue when using the old class sample, e.g., in (ii), its ground truth label and "KD label" have consistent magnitudes at the same position (1 and 0.7, respectively). **(c) Our selected placebos for two old classes ("road" and "table") and their activation maps** using GradCAM (Selvaraju et al., 2017) on CIFAR-100. The free image stream is ImageNet-1k that does not have class overlap with CIFAR-100. They are selected because their partial image regions contain similar visual cues with old classes.

the upper bound of KD is achieved when using "Old class data", and if compared to it, using "New class data" sees a clear performance drop for recognizing both old and new classes. In Figure 1 (b), we show the reason by diving into loss computation details: when using new class samples (as substitutes) to compute CE and KD losses simultaneously, these two losses are actually weakening each other, which does not happen in the ideal case of using old class samples.

Using unlabeled external data (called placebos in this paper) has been shown to be practical and effective in solving the above issue. Compared to existing works (Lee et al., 2019; Zhang et al., 2020), this paper aims to solve two open questions. **Q1**: How to adapt the placebo selection process in the non-stationary CIL pipeline. The ideal selection method needs to handle the dynamics of increasing classes in CIL, e.g., in a later incremental phase, it is expected to handle a more complex evaluation on the placebos of more old classes. **Q2**: How to control the computational and memory-related costs during the selection and utilization of placebos. It is not intuitive how to process external data without encroaching the memory allocated for new class data or breaking the strict assumption of memory budget in CIL.

We solve these questions by proposing a new method called PlacoboCIL that adjusts the policy of selecting placebos for each new incremental phase, in an online and automatic fashion and without needing extra memory. Specifically, to tackle **Q1**, we formulate the PlaceboCIL as an online Markov Decision Process (MDP) and introduce a novel online learning algorithm to learn a dynamic policy. In each new phase, this policy produces a phase-specific function to evaluate the quality of incoming placebos. The policy itself gets updated before the next phase. For **Q2**, we propose a mini-batch-based memory reusing strategy for PlaceboCIL. Given a free data stream, we sample a batch of unlabeled data, evaluate their quality by using our phase-specific evaluation function (generated by the learned policy), and keep only the high-quality placebos to compute KD losses. After this, we remove this batch totally from the memory before loading a new batch. In our implementation, this batch can be very small, e.g., 200 images. We randomly remove the same size of new class data for intaking this batch to keep the strict assumption of memory budget.

We evaluate PlaceboCIL by incorporating it into multiple strong baselines such as PODNet (Douillard et al., 2020), LUCIR (Hou et al., 2019), AANets (Liu et al., 2021a), and FOSTER (Wang

et al., 2022), and conducting a series of ablative studies. Our results on three popular CIL benchmarks show the clear and consistent superiority of PlaceboCIL, especially when using a low memory budget for old class exemplars. For example, our method boosts the last-phase accuracy by 6.9 percentage points on average when keeping only 5 exemplars per old class in the memory. In addition, it is worth mentioning that PlaceboCIL is surprisingly efficient even when there is no class overlap between placebos and original old class data. The reason is that PlaceboCIL can make use of the local visual cues in placebos, e.g., similar visual cues of "table" are found on the local regions of an "piano" (and "dishwasher") image as shown in Figure 1 (c).

Our contributions are three-fold. 1) A generic PlacoboCIL method that selects placebo images from a free image stream to solve the KD issue in existing methods. 2) A novel online learning algorithm for training a placebo selection policy and a mini-batch-based memory reusing strategy to avoid extra memory usage. 3) Extensive comparisons and visualizations on three CIL benchmarks, taking top-performing CIL models as baselines and with the same strict assumption on memory.

2 Related Work

Class-incremental learning (CIL) methods can be divided into three categories. ***Distillation-based*** methods introduce different knowledge distillation (KD) losses to consolidate previous knowledge. The key idea is to enforce prediction logits (Li & Hoiem, 2016; Rebuffi et al., 2017), feature maps (Douillard et al., 2020), or other essential information (Tao et al., 2020; Wang et al., 2022; Simon et al., 2021; Joseph et al., 2022) to be close to those of the pre-phase model. ***Memory-based*** methods use a small number of preserved old class data (called exemplars) (Rebuffi et al., 2017; Shin et al., 2017; Liu et al., 2020a; Prabhu et al., 2020) or augmented data (Zhu et al., 2021) to recall the old class knowledge. ***Network-architecture-based*** methods (Rusu et al., 2016; Xu & Zhu, 2018; Abati et al., 2020; Yan et al., 2021) design incremental network architectures by expanding the network capacity for new class data or freezing partial network parameters to keep the old class knowledge. Our method can be used to improve different *Distillation-based* CIL methods.

Some prior works used unlabeled external data for class-incremental learning. Lee et al. (2019) proposed a confidence-based sampling method to select unlabeled external data to compute a specially-designed global distillation loss. Zhang et al. (2020) randomly selected unlabeled samples and used them to compute KD loss for model consolidation. Liu et al. (2020b) used unlabeled data to maximize the classifier discrepancy when integrating an ensemble of auxiliary classifiers. Our method differs from them in two aspects. 1) Our method use the unlabeled data in a more generic way and can be applied to improve different distillation-based methods (Hou et al., 2019; Rebuffi et al., 2017; Wang et al., 2022), while the existing methods use unlabeled data to assist their specially-designed loss terms or components. 2) We train an online policy to select better unlabeled data to adapt to the non-stationary CIL pipeline while existing methods select unlabeled data by applying fixed (i.e., non-adaptive) rules in all incremental phases.

Online learning observes a stream of samples and makes a prediction for each element in the stream. There are mainly two settings in online learning: full feedback and bandit feedback. ***Full feedback*** means that the full reward function is given at each stage. It can be solved by Best-Expert algorithms (Even-Dar et al., 2005). ***Bandit feedback*** means that only the reward of the implemented decision is revealed. If the rewards are independently drawn from a fixed and unknown distribution, we may use e.g., Thompson sampling (Agrawal & Goyal, 2012) and UCB (Auer & Ortner, 2010) to solve it. If the rewards are generated in a non-stochastic version, we can solve it by e.g., Exp3 (Auer et al., 2002). ***Online MDP*** is an extension of online learning. Many studies (Even-Dar et al., 2009; Li et al., 2019) aim to solve it by converting it to online learning. In our case, we formulate the CIL as an online MDP and convert it into a classic online learning problem. The rewards in our MDP are non-stochastic because the training and validation data change in each phase. Therefore, we design our algorithm based on Exp3 (Auer et al., 2002).

3 Online Placebos for Class-incremental learning (PlaceboCIL)

CIL has multiple "training-testing" phases during which the number of classes gradually increases to the maximum. In the 0-th phase, data $\mathcal{D}_{1:c_0}$=$\{\mathcal{D}_1, ..., \mathcal{D}_{c_0}\}$, including the training samples of c_0 classes, are used to learn the model Θ_0. After this phase, only a small subset of $\mathcal{D}_{1:c_0}$ (i.e., exemplars

denoted as $\mathcal{E}_{1:c_0}$=$\{\mathcal{E}_1, ..., \mathcal{E}_{c_0}\}$) can be stored in the memory and used as replay samples in later phases. In the i-th phase, we use c_i to denote the number of classes we have observed so far. We get new class data $\mathcal{D}_{c_{i-1}+1:c_i}$=$\{\mathcal{D}_{c_{i-1}+1}, ..., \mathcal{D}_{c_i}\}$ of $(c_i - c_{i-1})$ classes and load exemplars $\mathcal{E}_{1:c_{i-1}}$ from the memory. Then, we initialize Θ_i with Θ_{i-1}, and train it using $\mathcal{T}_{1:c_i}$=$\mathcal{E}_{1:c_{i-1}} \cup \mathcal{D}_{c_{i-1}+1:c_i}$. The model Θ_i will be evaluated with a testing set $\mathcal{Q}_{1:c_i}$=$\{\mathcal{Q}_1, ..., \mathcal{Q}_{c_i}\}$ for all classes seen so far. Please note that in any phase of PlaceboCIL, we assume we can access a free image stream, where we can load unlabeled images and select placebos.

PlaceboCIL formulates the CIL task as an online MDP. In each phase, we update a policy, for which we sample a class-balanced subset from training data as the testing set, and use the updated policy to produce a phase-specific evaluation function. During model training, we sample unlabeled images, use the evaluation function to quickly judge the image quality (good or bad placebos), and select the good ones to compute KD loss. In this section, we introduce the formulation of online MDP in Section 3.1, show how to apply the policy to select placebos and compute KD loss in Section 3.2, and provide an online learning algorithm to update the policy in Section 3.3. *The pseudocode is given in Appendix A (Algorithms 1 and 2).*

3.1 Online MDP Formulation for CIL

The placebo selection process in CIL should be online inherently: training data (and classes) get updated in each phase, so the placebo selection policy should update accordingly. Thus, it is intuitive to formulate the CIL as an online MDP. In the following, we provide detailed formulations.

Stages. Each phase in the CIL task can be viewed as a stage in the online MDP.

States. The state should define the current situation of the intelligent agent. In CIL, we use the model Θ_i as the state of the i-th phase (i.e., stage). We use $\mathbb{S}$ to denote the state space.

Actions. We define the action as $\mathbf{a}_i$=(β_i, γ_i), consisting of the hyperparameters (β_i and γ_i) used to create an evaluation function. As β_i and γ_i vary in a continuous range, we *discretize* them to define a *finite* action space.[1] We will elaborate on how to take an action and deploy the hyperparameters in Section 3.2.

Policy π=$\{p(\mathbf{a}|\Theta_i)\}_{\mathbf{a}\in\mathbb{A}}$ is a probability distribution over the action space $\mathbb{A}$, given the current state Θ_i. We will elaborate how to update the policy using our proposed online learning algorithm in Section 3.3.

Environments. We take the training an testing data in each phase as the environment. In the i-th phase, the environment is $\mathcal{H}_i$=$(\mathcal{T}_{1:c_i}, \mathcal{Q}_{1:c_i})$, where $\mathcal{T}_{1:c_i}$ is the training data and $\mathcal{Q}_{1:c_i}$ is the testing data. The environment is time-varying because we observe different training data (and classes) in each new phase.

Rewards. CIL aims to train a model that is efficient in recognizing all classes seen so far. Therefore, it is intuitive to use the testing accuracy as the reward in each phase. We cannot observe any reward (i.e., testing accuracy) directly because the testing data is not accessible during training. We solve this by building a local testing set using a subset of training data (see details in Section 3.3). Our objective is to maximize a cumulative reward, i.e., $R = \sum_{i=1}^{N} r_{\mathcal{H}_i}(\Theta_i, \mathbf{a}_i)$, where $r_{\mathcal{H}_i}(\Theta_i, \mathbf{a}_i)$ denotes the i-th phase reward, The reward function $r_{\mathcal{H}_i}$ changes with $\mathcal{H}_i$, so it is time-varying.

3.2 Placebo Selection with Mini-batch-based Memory Reusing Strategy

In the following, we introduce how to build phase-specific evaluation functions using the policy, select high-quality placebos without breaking memory constraints, and compute KD loss with the selected placebos. The computation flow (in each phase) is illustrated in Figure 2.

Computing prototypes. Our placebo selection is based on the distance from placebo to the class prototype, i.e., the mean feature of each class (Snell et al., 2017). First, we compute the prototypes of all seen classes. We use exemplars to compute the prototypes of old classes, and use new class

[1] Though discretization suffers the curse of dimensionality, our experiments show that with a coarse grid, we already have significant improvements over pre-fixed hyperparameters.

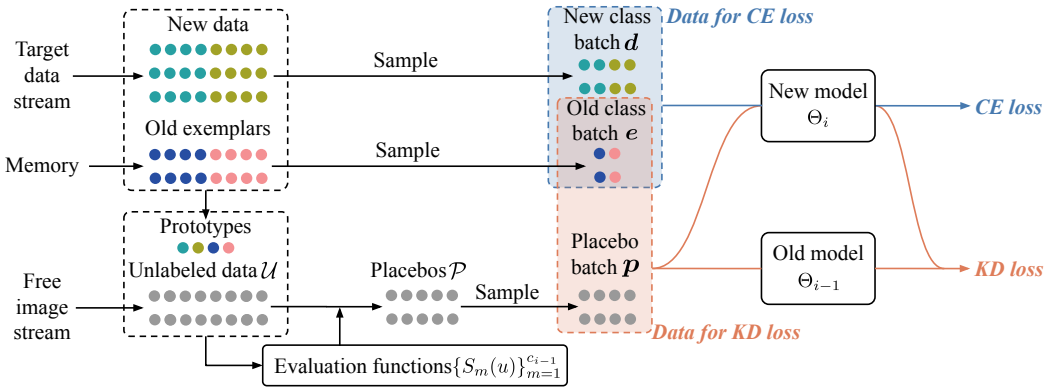


Figure 2: The computation flow of our PlaceboCIL in the i-th phase. At the beginning of this phase, we build phase-specific evaluation functions $\{S_m(u)\}_{m=1}^{c_{i-1}}$. During training, we select placebos as follows. 1) We load an unlabeled data batch $\mathcal{U}$ from the free image stream. 2) We compute scores using $\{S_m(u)\}_{m=1}^{c_{i-1}}$ for all samples in $\mathcal{U}$. 3) For each old class m, we select K placebos with the highest scores and add them to $\mathcal{P}$. 4) We delete used placebos from $\mathcal{P}$ after computing the loss. 5) When we use up the selected placebos in $\mathcal{P}$, we repeat the selection steps.

training data for new class prototypes, as follows,

$$\mathrm{Pro}(\mathcal{E}_n) = \frac{1}{|\mathcal{E}_n|} \sum_{z \in \mathcal{E}_n} \mathcal{F}_{\Theta_i}(z), \quad \mathrm{Pro}(\mathcal{D}_l) = \frac{1}{|\mathcal{D}_l|} \sum_{z \in \mathcal{D}_l} \mathcal{F}_{\Theta_i}(z), \tag{1}$$

where $\mathcal{F}_{\Theta_i}(\cdot)$ denotes the encoder (i.e., the feature extractor) of Θ_i. $\mathrm{Pro}(\mathcal{E}_n)$ and $\mathrm{Pro}(\mathcal{D}_l)$ denote the prototypes of the n-th old class and the l-th new class, respectively.

Building evaluation functions. We argue that high-quality placebos for the m-th old class should meet two requirements: (1) being close to the prototype of the m-th class in the feature space because they will be used to activate the related neurons of the m-th old class in the model; and (2) being far from the prototypes of all the other classes in the feature space so that they will not cause the KD issue (as shown in Figure 1). To achieve these, we design the following evaluation function $\mathcal{S}_m(x)$ for the m-th old class in the i-th phase:

$$\begin{aligned} \mathcal{S}_m(x) = &- \mathrm{Sim}\left(\mathcal{F}_{\Theta_i}(x), \mathrm{Pro}(\mathcal{E}_m)\right) + \beta_i \sum_{\substack{n=1 \\ n \neq m}}^{c_{i-1}} \frac{\mathrm{Sim}\left(\mathcal{F}_{\Theta_i}(x), \mathrm{Pro}(\mathcal{E}_n)\right)}{c_{i-1} - 1} \\ &+ \gamma_i \sum_{l=c_{i-1}+1}^{c_i} \frac{\mathrm{Sim}\left(\mathcal{F}_{\Theta_i}(x), \mathrm{Pro}(\mathcal{D}_l)\right)}{c_i - c_{i-1}}, \end{aligned} \tag{2}$$

where x denotes an unlabeled input image, and $\mathrm{Sim}(\cdot, \cdot)$ denotes cosine similarity. β_i and γ_i are two hyperparameters from the action $\mathbf{a}_i{=}(\beta_i, \gamma_i)$, sampled by the policy π.

Allocating mini-batch-based memory for placebos. We need to allocate a small amount of memory to store unlabeled images (before evaluating them). At the beginning of the i-th phase, we allocate memory buffers $\mathcal{U}$ and $\mathcal{P}$ respectively for the unlabeled image candidates and the selected placebos. *In order to not exceed the memory budget, we randomly remove the same number, i.e., $|\mathcal{U}+\mathcal{P}|$, of samples from the training data of new classes.* Our empirical results show this "remove" does not degrade the model performance on new classes. Please kindly refer to Appendix G for detailed results and comparisons.

Selecting placebos. Whenever the placebo buffer $\mathcal{P}$ is empty, we load a batch of unlabeled samples $\mathcal{U}$ from the free image stream, and choose K placebos for each old class to add into $\mathcal{P}$, as follows,

$$\mathcal{P} := \{x_k\}_{k=1}^{c_{i-1} \times K} = \underset{x_k \in \mathcal{U}}{\arg\max} \sum_{m=1}^{c_{i-1}} \sum_{k=1}^{K} \mathcal{S}_m(x_k). \tag{3}$$

Calculating loss with placebos. After selecting placebos, we sample a batch of new class data $\boldsymbol{d} \subset \mathcal{D}_{c_{i-1}+1:c_i}$, a batch of old class exemplars $\boldsymbol{e} \subset \mathcal{E}_{1:c_0}$, and a batch of placebos $\boldsymbol{p} \subset \mathcal{P}$. We calculate the overall loss as follows,

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}}(\Theta_i; \boldsymbol{d} \cup \boldsymbol{e}) + \lambda \mathcal{L}_{\mathrm{KD}}(\Theta_{i-1}, \Theta_i; \boldsymbol{p} \cup \boldsymbol{e}), \tag{4}$$

where $\mathcal{L}_{\mathrm{CE}}$ and $\mathcal{L}_{\mathrm{KD}}$ denote the CE loss and KD loss, respectively. λ is a hyperparameter to balance the two losses (Rebuffi et al., 2017). To control the memory usage, we delete $\boldsymbol{p}$ from $\mathcal{P}$ immediately after calculating the loss. When $\mathcal{P}$ is empty, we repeat the placebo selection operation.

3.3 Online Policy Learning Algorithm

A common approach to solving an online MDP is to approximate it as an online learning problem and solve it using online learning algorithms (Even-Dar et al., 2005; Agrawal & Goyal, 2012; Auer et al., 2002). We also follow this idea in PlaceboCIL, and our approximation follows the work by Even-Dar et al. (2009) that is theoretically proved to have the optimal regret. Specifically, Even-Dar et al. (2009) relax the Markovian assumption of the MDP by decoupling the cumulative reward function and letting it be time-dependent so that they can solve online MDP by standard online learning algorithms.

However, we cannot directly apply the algorithms proposed in (Even-Dar et al., 2009) to our problem. It is because they assume *full feedback* meaning that the model can observe the rewards of all actions in every learning phase (which is also why its online learning problem can be solved by Best Expert algorithms (Even-Dar et al., 2005)). While in CIL, we cannot observe any reward (i.e., the testing accuracies) because the testing data $Q_{1:c_i}$ are not accessible in any phase i. To address this problem, we split the training data we have in each phase into two subsets: one for training and another for validation. Once we have a validation set, we can solve our online learning problem based on Exp3 (Auer et al., 2002)—a simple and effective bandit algorithm. In the following, we elaborate how we do this data splitting in each local dataset (i.e., the entire data we have in each training phase of CIL), compute the decoupled cumulative reward, and learn the policy π with Exp3.

Rebuilding local datasets. To compute reward, we sample a class-balanced subset $\mathcal{B}_{1:c_i}$ from the training data $\mathcal{T}_{1:c_1}$. $\mathcal{B}_{1:c_i}$ contains the same number of samples for both the old and new classes. In this way, we rebuild the local training and validate sets, and update the environment from the oracle $\mathcal{H}_i{=}(\mathcal{T}_{1:c_i}, \mathcal{Q}_{1:c_i})$ (which is unavailable in CIL) to the local environment $h_i{=}(\mathcal{T}_{1:c_i} \setminus \mathcal{B}_{1:c_i}, \mathcal{B}_{1:c_i})$.

Decoupled cumulative reward. We create the decoupled cumulative reward function R based on the original cumulative reward function $\sum_{j=1}^{N} r_{\mathcal{H}_j}(\Theta_j, \mathbf{a}_j)$. In the i-th phase, we compute R as follows,

$$R(\mathbf{a}_i, h_i) = \sum_{j=i}^{i+n} r_{h_i}(\Theta_j, \mathbf{a}_i) + \text{costant}, \tag{5}$$

where the "constant" denotes the historical rewards from the 1-st phase to the $(i$-1)-th phase. It doesn't influence policy optimization. $R(\mathbf{a}_i, h_i)$ is the long-term reward of a time-invariant local MDP based on the local environment h_i. We use $R(\mathbf{a}_i, h_i)$ as an estimation of the final cumulative reward, following (Even-Dar et al., 2009). Because we don't know the total number of phases N during training, we assume there will be n phases in the future. Furthermore, we fix the action $\mathbf{a}_i$ to simplify the training process. $R(\mathbf{a}_i, h_i)$ is a function of $\mathbf{a}_i$ and h_i.

Training policy with Exp3. Exp3 (Auer et al., 2002) introduces an auxiliary variable $\mathbf{w} = \{w(\mathbf{a})\}_{\mathbf{a}\in\mathbb{A}}$. It is updated as follows. In the 1-st phase, we initialize $\mathbf{w}$ as $\{1, \ldots, 1\}$. In each phase i ($i{\geq}1$), we update $\mathbf{w}$ for T iterations. In the t-th iteration, we sample an action $\mathbf{a}_t{\sim}\pi$, apply the action $\mathbf{a}_t$ to the CIL system, and compute $R(\mathbf{a}_t, h_i)$ using Eq. 5. After that, we update $w(\mathbf{a}_t)$ in $\mathbf{w}$ as,

$$w(\mathbf{a}_t) \leftarrow w(\mathbf{a}_t) \exp(\xi R(\mathbf{a}_t, h_i)/p(\mathbf{a}_t|\Theta_i)), \tag{6}$$

where ξ can be regarded as the learning rate. After updating $\mathbf{w}$, we get the policy $\pi{=}\mathbf{w}/||\mathbf{w}||$. *The pseudocode is available in Appendix A (Algorithms 1 and 2).*

Method	20 exemplars/class		10 exemplars/class		5 exemplars/class	
	Average	Last	Average	Last	Average	Last
LwF (2016)	53.19	43.18	45.96	34.10	35.41	24.91
w/ ours	59.08 +5.89	49.15 +5.97	53.61 +7.65	38.36 +4.26	41.55 +6.14	28.68 +3.77
iCaRL (2017)	57.12	47.49	53.43	41.49	43.73	34.33
w/ ours	61.24 +4.12	51.47 +3.98	59.11 +5.68	46.42 +4.93	51.55 +7.82	39.35 +5.02
LUCIR (2019)	63.17	53.71	60.50	49.08	51.36	39.37
w/ ours	65.28 +2.11	56.23 +2.52	64.79 +4.29	55.44 +6.36	62.74 +11.35	53.25 +13.88
LUCIR-AANets (2021a)	66.12	55.27	61.12	48.83	53.81	42.93
w/ ours	67.65 +1.53	59.18 +3.91	64.30 +3.18	52.92 +4.09	60.27 +6.46	48.45 +5.52
FOSTER (2022)	70.62	62.97	62.03	52.23	56.80	43.11
w/ ours	71.97 +1.35	64.43 +1.46	65.12 +3.09	54.81 +2.48	62.78 +5.98	50.72 +7.61

Table 1: Evaluation results (%) on CIFAR-100 (N=5) using different baselines *w/* and *w/o* our PlaceboCIL. "Average" denotes the average accuracy over all phases. "Last" denotes the last phase (5-th phase) accuracy.

4 EXPERIMENTS

We evaluate our method on three CIL benchmarks and achieve consistent improvements over multiple baseline methods. Below we introduce datasets and implementation details, followed by results and analyses, including the comparison to the state-of-the-art, an ablation study, and the visualization of our placebos.

Datasets and free image streams. We use three datasets: CIFAR-100 (Krizhevsky et al., 2009), ImageNet-100 (Rebuffi et al., 2017), and ImageNet-1k (Russakovsky et al., 2015). ImageNet-100, which contains 100 classes, is sampled from ImageNet-1k. We use exactly the same classes or orders as the related works (Rebuffi et al., 2017; Hou et al., 2019). For CIFAR-100, we use ImageNet-1k as the free image stream. For ImageNet-100, we use a 900-class subset of ImageNet-1k, which is the complement of ImageNet-100 in ImageNet-1k. For ImageNet-1k, we use a $1,000$-class subset of ImageNet-21k (Deng et al., 2009) without any overlapping class (different super-classes from those in ImageNet-1k).

Implementation details. Following (Hou et al., 2019; Douillard et al., 2020; Liu et al., 2021a;b), we use a modified 32-layer ResNet for CIFAR-100 and an 18-layer ResNet for ImageNet datasets. The number of exemplars for each class is 20 in the default setting. The training batch size is 128. On CIFAR-100 (ImageNet-Subset/1k), we train it for 160 (90) epochs in each phase, and divide the learning rate by 10 after 80 (30) and then after 120 (60) epochs. If the baseline is POD-AANets (Liu et al., 2021a), we fine-tune the model for 20 epochs using only exemplars. We apply different forms of distillation losses on different baselines: (1) if the baselines are LwF and iCaRL, we use the softmax KL divergence loss; (2) if the baselines are LUCIR and AANets, we use the cosine embedding loss (Hou et al., 2019); and (3) if the baseline is POD-AANets, we use pooled outputs distillation loss (Douillard et al., 2020). For our PlaceboCIL, $|\mathcal{U}|$ and $|\mathcal{P}|$ are set as $1,000$ and 200, respectively. All experiments of our PlaceboCIL in the main paper use the "strict budget" setting, i.e., deleting $|\mathcal{U}+\mathcal{P}|$ samples from training data in order to not exceed the memory budget. An ablation study on the memory budget setting is provided in Appendix G. More implementation details are provided in Appendices C (benchmark protocol), D (network architecture), and E (training configurations).

Results on five baselines. Table 1 shows the average and last-phase accuracy for five baselines (i.e., LwF (Li & Hoiem, 2016), iCaRL (Rebuffi et al., 2017), LUCIR (Hou et al., 2019), AANets (Liu et al., 2021a), and FOSTER (Wang et al., 2022)). From the table, we make the following observations. 1) Using our PlaceboCIL boosts the performance of the baselines clearly and consistently in all settings, indicating that our method is generic and efficient. 2) When the number of exemplars decreases, the improvement brought by our method becomes more significant. For example, the last-phase accuracy improvement of LUCIR increases from 2.52 to 13.88 percentage points when the number of exemplars per class decreases from 20 to 5. This reveals that the superiority of our method is more obvious when the forgetting problem is more serious (with fewer exemplars) due to a tighter memory budget in CIL. 3) Our PlaceboCIL can boost the performance of all KD terms, i.e.,

Method	*CIFAR-100*			*ImageNet-100*			*ImageNet-1k*	
	N=5	10	25	5	10	25	5	10
LwF (2016)	53.19	46.98	45.51	53.62	47.64	44.32	44.35	38.90
iCaRL (2017)	57.12	52.66	48.22	65.44	59.88	52.97	51.50	46.89
TPCIL (2020)	65.34	63.58	–	76.27	74.81	–	64.89	62.88
DDE (2021)	65.34	63.58	–	76.27	74.81	–	64.89	62.88
GeoDL (2021)	65.14	65.03	63.12	76.63	75.40	71.43	65.23	64.46
DER (2021)	68.65	67.48	66.18	78.40	78.20	75.40	68.13	65.97
ELI (2022)	68.78	66.62	64.72	73.54	71.82	70.32	–	–
GD+ext (2019)	63.17±0.47	58.71±0.39	51.79±0.42	75.67±0.51	72.08±0.61	65.13±0.56	–	–
MUC-LwF (2020b)	59.03±0.35	53.27±0.47	49.06±0.49	72.31±0.53	68.92±0.60	62.93±0.62	–	–
POD-AANets (2021a)	66.12±0.41	64.11±0.32	62.12±0.51	76.63±0.47	75.40±0.36	71.43±0.32	67.60±0.39	64.79±0.42
w/ PlaceboCIL (ours)	67.65±0.45	65.78±0.40	64.95±0.46	78.24±0.52	77.14±0.47	**75.85**±0.42	68.55±0.34	65.49±0.38
FOSTER (2022)	70.62±0.58	68.43±0.45	63.83±0.62	80.21±0.67	77.63±0.73	69.27±0.50	69.32±0.47	66.07±0.61
w/ PlaceboCIL (ours)	**71.97**±0.49	**70.31**±0.59	**67.02**±0.65	**82.03**±0.49	**79.52**±0.60	72.79±0.45	**71.02** ±0.39	**68.82**±0.54

Table 2: Average accuracy (%) across all phases. The first block shows top-performing CIL methods. The second block shows CIL methods that use unlabeled data. The third block shows our method.

not only for logits-based KD (Rebuffi et al., 2017) but also for feature-based KD (Hou et al., 2019; Douillard et al., 2020).

Comparisons to the state-of-the-art. Table 2 (Blocks 1&3) shows the results of our best model (taking PlaceboCIL as a plug-in module in the top method (Wang et al., 2022)) and some recent top-performing methods. We can see that using our PlaceboCIL outperforms all previous methods. Intriguingly, we find that we can surpass others more when the number of phases is larger—where there are more serious forgetting problems. For example, when N=25, we improve POD-AANets by 4.4% on the ImageNet-100, while this number is only 1.6% when N=5 (which is an easier setting with more saturated results). This reflects the encouraging efficiency of our method for reducing the forgetting of old class knowledge in CIL models.

Comparisons to the CIL methods using unlabeled data. Table 2 (Blocks 2&3) shows the results of our best model and CIL methods using unlabeled data (GD+ext (Lee et al., 2019) and MUC-LwF (Liu et al., 2020b)). We can see that our method consistently performs better than others. For another related work, DMC (Zhang et al., 2020), we didn't find the public code. So, we compare ours with DMC using their paper's setting: iCaRL w/ ours achieves 62.3%, while the result of DMC is 59.1% (CIFAR-100, 10 phases, 10 classes/phase). Please kindly refer to Appendix B for more analyses.

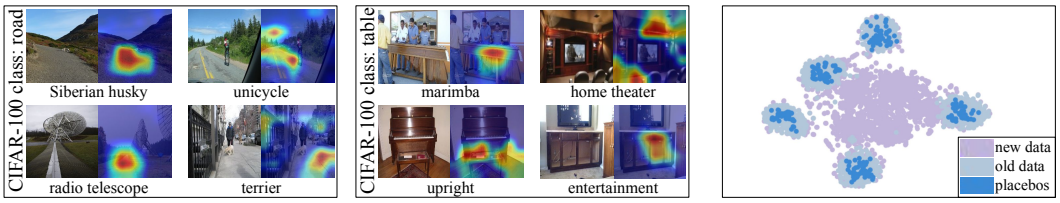
Ablation study. Tables 3 (and A1) show the ablation results for free data streams, policy learning methods, and placebo selection strategies.

1) First block. Row 1 shows the baselines. Row 2 shows our method.

2) Second block: different free data streams. Rows 3-6 show the ablation results for the following free data streams. (1) "Overlapping" means including samples from the overlapping classes between CIFAR-100 and ImageNet. (2) "Non-

No.	Setting	iCaRL		LUCIR-AANets	
		Average	Last	Average	Last
1	Baseline	57.12	47.49	66.72	57.77
2	PlaceboCIL	61.01	51.45	67.16	59.14
3	Overlapping	62.15	52.62	67.48	59.06
4	Non-overlapping	61.52	51.70	67.01	58.53
5	New data	57.70	47.51	66.69	57.33
6	Old data (oracle)	66.64	58.03	68.82	61.52
7	*w/o* Online learning	60.27	50.57	66.91	58.88
8	Offline RL	61.09	50.81	67.31	59.26
9	Higher confidence	60.43	49.36	66.97	58.12
10	Random placebos	56.27	46.64	66.23	57.22

Table 3: Ablation results (%) on CIFAR-100, N=5. (1) ***First block: baselines.*** Row 1 shows the baselines. Row 2 shows our method. All the following ablation settings (Rows 3-10) are based on Row 2. (2) ***Second block: different free data streams.*** Rows 3-6 show the ablation results for the following free data streams. (3) ***Third block: different policy learning methods.*** Row 7 is for using fixed evaluation functions (β_i=γ_i=1). Row 8 uses the offline RL (the REINFORCE algorithm) to train the selection policy. (4) ***Fourth block: different placebo selection strategies.*** Row 9 uses unlabeled data with higher confidence. Row 10 uses them randomly.



(a) Selected placebos (b) t-SNE visualization

Figure 3: (a) Selected placebos for two CIFAR-100 classes and their GradCAM activation maps. The free image stream is non-matching ImageNet-1k. (b) The t-SNE results on CIFAR-100 (N=5). For clear visualization, we randomly pick five new classes and five old classes. The **purple**, **light blue**, and **dark blue** points denote the new data, old data, and selected placebos, respectively.

overlapping" means using only the samples of non-overlapping classes between CIFAR-100 and ImageNet (more realistic than "Overlapping"). (3) "New data" means using only the current-phase new class data (i.e., without using any free data stream) as candidates to select placebos. (4) "Old data" means the original old class data are all accessible when computing KD loss (i.e., upper bound of KD effect). Please note that in (1) and (2), two classes are considered "overlapping" if their classes or super-classes overlap. For example, "n02640242 - sturgeon" in ImageNet-1k is regarded as an overlapping class of the "fish" in CIFAR-100, because they overlap at the level of super-class (i.e., "fish"). When comparing Row 4 with Row 2, we can find that our method is robust to the change of data streams: even if all overlapping classes are removed, our method can still achieve the same-level performance. Comparing Row 5 with Row 2, we can get a clear sense that using additional unlabeled data is definitely helpful. Comparing Row 6 with Row 2, we see that our method achieves comparable results to the upper bound.

3) Third block: different policy learning methods. Row 7 is for using fixed evaluation functions (β_i=γ_i=1). Row 8 uses the offline RL (the REINFORCE algorithm (Liu et al., 2021b)) to train the selection policy. Comparing Row 7 with Row 2 shows that using online learning successfully boosts the model performance. Comparing Row 8 with Row 2, we are happy to see that our online learning method achieves the same-level performance as the offline RL while the training time is much less. The training time of the baseline (without learning a policy) is 2.7 hours. It becomes around 650 hours if we solve the MDP by offline RL. In contrast, using our online method takes only 4.5 hours.

4) Fourth block: different placebo selection strategies. Row 9 uses unlabeled data with higher confidence following (Lee et al., 2019). Row 10 uses them randomly following (Zhang et al., 2020). Comparing these results with Row 2 shows our superiority. The "mini-batch-based memory reusing strategy" is applied in Rows 9 and 10.

Visualization results. Figure 3 (a) demonstrates the activation maps visualized by Grad-CAM for the placebos of two old classes on CIFAR-100 ("road" and "table"). ImageNet-1k is the free data stream. We can see that the selected placebos contain the parts of "road" and "table" even though their original labels (on ImageNet-1k) are totally different classes. While this is not always the case, our method seems to find sufficiently related images to old classes that activate the related neurons for old classes ("road" and "table"). To illustrate that, Figure 3 (b) shows t-SNE results for placebos, old class data (not visible during training), and new class data. We can see that the placebos are located near the old class data and far away from the new class data. This is why placebos can recall the old knowledge without harming the new class learning.

5 Conclusions

We proposed a novel method, PlaceboCIL, which selects high-quality placebo data from free data streams and uses them to improve the effect of KD in CIL. We designed an online learning method to make the selection of placebos more adaptive in different phases and a mini-batch-based memory reusing strategy to control memory usage. Extensive experiments show that our method is general and efficient.

ETHICS STATEMENT

Computational costs. Deep learning methods require intensive usage of computing resources, which is not climate-friendly. It calls for future research into proposing more effective training strategies that can speed up the training.

Privacy issues. Keeping old class exemplars has the issue of data privacy. This calls for future research that explicitly forgets or mitigates the identifiable feature of the data.

Licenses. We use the open-source code for the following papers: AANets (Liu et al., 2021a), iCaRL (Rebuffi et al., 2017), Mnemonics (Liu et al., 2020a), PODNet (Douillard et al., 2020), FOSTER (Wang et al., 2022). They are licensed under the MIT License.

Datasets. We use three datasets in our paper: CIFAR-100 (Krizhevsky et al., 2009), ImageNet-100 (Rebuffi et al., 2017) and ImageNet-1k (Russakovsky et al., 2015). The data for both datasets are downloaded from their official websites and allowed to use for non-commercial research and educational purposes.

REPRODUCIBILITY STATEMENT

We include the details of reproducing our experiments in Section 4 (main paper), Appendix C, Appendix D, and Appendix E. We also provide the code in the supplementary.

REFERENCES

Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *CVPR*, pp. 3931–3940, 2020. 3

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*, volume 23, pp. 39.1–39.26, 2012. 3, 6

Peter Auer and Ronald Ortner. Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. *Periodica Mathematica Hungarica*, 61(1-2):55–65, 2010. 3

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multi-armed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002. 3, 6

Charilaos Christopoulos, Athanassios Skodras, and Touradj Ebrahimi. The jpeg2000 still image coding system: an overview. *IEEE transactions on consumer electronics*, 46(4):1103–1127, 2000. 14

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pp. 248–255, 2009. 7

Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pp. 86–102, 2020. 1, 2, 3, 7, 8, 10, 14

Eyal Even-Dar, Sham M Kakade, and Yishay Mansour. Experts in a markov decision process. In *NIPS*, pp. 401–408, 2005. 3, 6

Eyal Even-Dar, Sham M Kakade, and Yishay Mansour. Online markov decision processes. *Mathematics of Operations Research*, 34(3):726–736, 2009. 3, 6

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016. 14

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pp. 831–839, 2019. 1, 2, 3, 7, 8, 14, 15

Xinting Hu, Kaihua Tang, Chunyan Miao, Xian-Sheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. In *CVPR*, 2021. 8

KJ Joseph, Salman Khan, Fahad Shahbaz Khan, Rao Muhammad Anwer, and Vineeth N Balasubramanian. Energy-based latent aligner for incremental learning. In *CVPR*, pp. 7452–7461, 2022. 3, 8

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 7, 10

Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *ICCV*, pp. 312–321, 2019. 2, 3, 8, 9, 13, 14

Yingying Li, Aoxiao Zhong, Guannan Qu, and Na Li. Online markov decision processes with time-varying transition probabilities and rewards. In *ICML workshop on Real-world Sequential Decision Making*, 2019. 3

Zhizhong Li and Derek Hoiem. Learning without forgetting. In *ECCV*, pp. 614–629, 2016. 3, 7, 8, 15

Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *CVPR*, pp. 12245–12254, 2020a. 1, 3, 10, 14

Yaoyao Liu, Bernt Schiele, and Qianru Sun. Adaptive aggregation networks for class-incremental learning. In *CVPR*, pp. 2544–2553, 2021a. 1, 2, 7, 8, 10, 14, 16

Yaoyao Liu, Bernt Schiele, and Qianru Sun. Rmm: Reinforced memory management for class-incremental learning. In *NeurIPS*, 2021b. 7, 9, 15

Yu Liu, Sarah Parisot, Gregory Slabaugh, Xu Jia, Ales Leonardis, and Tinne Tuytelaars. More classifiers, less forgetting: A generic multi-classifier paradigm for incremental learning. In *ECCV*, pp. 699–716, 2020b. 3, 8, 14

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. 1

K. McRae and P. Hetherington. Catastrophic interference is eliminated in pre-trained networks. In *CogSci*, 1993. 1

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *ECCV*, pp. 524–540, 2020. 3

R. Ratcliff. Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*, 97:285–308, 1990. 1

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental classifier and representation learning. In *CVPR*, pp. 5533–5542, 2017. 1, 2, 3, 6, 7, 8, 10, 14

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 7, 10

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv*, 1606.04671, 2016. 3

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *CVPR*, pp. 618–626, 2017. 2

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NIPS*, pp. 2990–2999, 2017. 3

Christian Simon, Piotr Koniusz, and Mehrtash Harandi. On learning the geodesic path for incremental learning. In *CVPR*, pp. 1591–1600, 2021. 3, 8

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, pp. 4077–4087, 2017. 4

Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *ECCV*, pp. 254–270, 2020. 3, 8

Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *ECCV*, 2022. 2, 3, 7, 8, 10

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, pp. 374–382, 2019. 14

Ju Xu and Zhanxing Zhu. Reinforced continual learning. In *NeurIPS*, pp. 899–908, 2018. 3

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *CVPR*, pp. 3014–3023, 2021. 3, 8

Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *WACV*, pp. 1131–1140, 2020. 2, 3, 8, 9, 13, 14

Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-lin Liu. Class-incremental learning via dual augmentation. *NeurIPS*, pp. 14306–14318, 2021. 1, 3

APPENDICES

A ALGORITHMS

Supplementary to Section 3. Algorithm 1 summarizes the overall training steps of the proposed method in the i-th phase ($i \geq 1$). Algorithm 2 presents the training with placebos for a specific action $\mathbf{a}$.

Algorithm 1: Our online learning algorithm in the i-th phase ($i \geq 1$)

Input : Old model Θ_{i-1}, training data $\mathcal{T}_{1:c_i}$, testing data $\mathcal{Q}_{1:c_i}$, learnable parameters $\mathbf{w}$, numbers of epochs M_1 and M_2.
Output: New model Θ_i, new exemplars $\mathcal{E}_{0:i}$, learnable parameters $\mathbf{w}$.

// Policy learning
if i=1 **then**
  Initialize $\mathbf{w} = \{1, \ldots, 1\}$;
for t in $i, ..., T$ **do**
  Randomly sample a class-balanced subset $\mathcal{B}_{1:c_i}$ from $\mathcal{T}_{1:c_i}$,;
  Create the local environment $h_i = ((\mathcal{T}_{1:c_i}) \setminus \mathcal{B}_{1:c_i}, \mathcal{B}_{1:c_i})$;
  Set the policy $\pi = \mathbf{w}/||\mathbf{w}||$;
  Sample an action $\mathbf{a}_t \sim \pi$;
  for j in $i, ..., i+n$ **do**
    Train Θ_j for M_1 epochs by **Algorithm 2** with inputs Θ_{j-1}, $\mathbf{a}_t$, h_i;
    Collect the reward $r_{h_i}(\Theta_j, \mathbf{a}_t)$;
  Compute the cumulative reward $\hat{R}(\mathbf{a}_t, h_i)$ by Eq. 5;
  Update $\mathbf{w}$ by Eq. 6;
// CIL training
Sample an action $\mathbf{a}_i \sim \pi$;
Train Θ_i for M_2 epochs by **Algorithm 2** with inputs Θ_{i-1}, $\mathbf{a}_i$, $\mathcal{H}_i = (\mathcal{T}_{1:c_i}, \mathcal{Q}_{1:c_i})$;
Select new exemplars $\mathcal{E}_{1:c_i}$ from $\mathcal{T}_{1:c_i}$.

Algorithm 2: Training with placebos for action $\mathbf{a}$

Input : Old model Θ_{old}, action $\mathbf{a} = \{\beta, \gamma\}$, environment $h = \{\mathcal{T}, \mathcal{Q}\}$.
Output: New model Θ, reward $r_h(\Theta, \mathbf{a})$ (i.e., the testing accuracy).

Initialize Θ with Θ_{old};
Create $\{S_m(x)\}_{m=1}^{c_{i-1}}$ based on $\mathbf{a} = \{\beta, \gamma\}$ using Eq. 2;
for *epochs* **do**
  Set $\mathcal{P} = \varnothing$;
  while $\mathcal{P} == \varnothing$ **do**
    Sample $\mathcal{U}$ from the free image stream;
    Select placebos $\mathcal{P} \subset \mathcal{U}$ using Eq. 3;
    for *iterations* **do**
      Sample mini-batches $\boldsymbol{p}$, $\boldsymbol{d}$, and $\boldsymbol{e}$;
      Compute the loss $\mathcal{L}$ by Eq. 4 and update Θ;
      Update placbo buffer $\mathcal{P} := \mathcal{P} \setminus \boldsymbol{p}$;
Compute the reward $r_h(\Theta, \mathbf{a})$ on $\mathcal{Q}$.

B COMPARISONS TO THE CIL METHODS USING UNLABELED DATA.

This is supplementary to Section 4 **"Comparisons to the CIL methods using unlabeled data."**

Part I: Comparing with GD+ext (Lee et al., 2019) and DMC (Zhang et al., 2020). Lee et al. (2019) use the unlabeled data to compute the distillation loss in CIL. Our work also uses unlabeled data but differs from (Lee et al., 2019) in three aspects:

- Our method is focused on the learnable strategies of sampling helpful unlabeled data for CIL. It is generic and can be plugged into different distillation-based frameworks, including both logit distillation (LwF and iCaRL) and feature distillation (LUCIR and PODNet). In Table 2, we also show the empirical results (comparison) to demonstrate our superiority.
- We propose an online learning algorithm. Its learned policy can adaptively produce phase-specific functions to evaluate the quality of the unlabeled samples. So our selection functions change accordingly when the number of phases increases. Lee et al. (2019)'s selection strategy remains unchanged when the number of phases increases. Zhang et al. (2020) use the unlabeled data without selection. In Table 3, we show our learnable selection functions generally perform better than (Lee et al., 2019) and (Zhang et al., 2020). Please kindly refer to Section 4 "**Ablation study**" for the results.
- Our employment of unlabeled data is dynamic and takes little memory out of the total memory budget. We maintain the memory budget strictly the same as the baselines (LUCIR (Hou et al., 2019), PODNet (Christopoulos et al., 2000), and Mnemonics (Liu et al., 2020a)), by loading fewer samples of new classes. In contrast, (Lee et al., 2019) requires double-size memory compared to the baselines (LUCIR (Hou et al., 2019), PODNet (Christopoulos et al., 2000)), and Mnemonics (Liu et al., 2020a)).

Part II: Comparing with MUC (Liu et al., 2020b). Our method and (Liu et al., 2020b) use unlabeled data for different purposes and in different ways. We use the unlabeled placebo data from a free image stream to improve the positive effect of KD in training CIL models. Liu et al. (2020b) integrate the ensemble of multiple FC classifiers to set regularization-based constraints. They use the unlabeled data to compute the classifier discrepancy loss by using the predictions of multiple classifiers.

C Benchmark Protocols

This is supplementary to Section 4 "**Implementation details**." We follow the benchmark protocol used in (Douillard et al., 2020; Hou et al., 2019; Liu et al., 2021a; 2020a). Given a dataset, the initial model (in the 0-th phase) is trained on the data of half of the classes. Then, it learns the remaining classes evenly in the subsequent N phases. For N, there are three options (5, 10, and 25), and the corresponding settings are called "N-Phase". The learned model in each phase is evaluated on the test set containing all seen classes. In the tables, we report average accuracy over all phases and the last-phase accuracy.

D Network Architecture Details

This is supplementary to Section 4 "**Implementation details**." Following (Hou et al., 2019; Liu et al., 2021a; Rebuffi et al., 2017; Wu et al., 2019), we use a modified 32-layer ResNet (Rebuffi et al., 2017) for CIFAR-100 and an 18-layer ResNet (He et al., 2016) for ImageNet-100 and ImageNet-1k. Please note that it is standard to use a shallower ResNet for ImageNet-100 and ImageNet-1k in CIL. The 32-layer ResNet consists of 1 initial convolution layer and 3 residual blocks (in a single branch). Each block has 10 convolution layers with 3×3 kernels. The number of filters starts from 16 and is doubled every next block. After these 3 blocks, there is an average-pooling layer to compress the output feature maps to a feature embedding. The 18-layer ResNet follows the standard settings in (He et al., 2016). We deploy AANets using the same parameters as its original paper (Liu et al., 2021a).

E More Training Configurations

This is supplementary to Section 4 "**Implementation details**." The training of the classification model Θ exactly follows the uniform setting in (Douillard et al., 2020; Hou et al., 2019; Liu et al., 2021a; 2020a).

F ABLATION STUDY ON MORE BASELINES

This is supplementary to Section 4 "**Ablation study**." In Table A1, we provide the ablation results on another two baselines, LwF Li & Hoiem (2016) and LUCIR Hou et al. (2019).

No.	Setting	LwF		LUCIR	
		Average	Last	Average	Last
1	Baseline	53.19	43.18	63.17	53.71
2	PlaceboCIL	59.06	48.56	65.42	56.18
3	Overlapping	58.95	48.71	65.73	57.26
4	Non-overlapping	58.59	49.28	65.48	56.99
5	New data	54.06	43.94	64.21	53.98
6	Old data (oracle)	61.41	51.16	67.02	58.98
7	*w/o* Online learning	57.06	46.12	63.83	55.33
8	Offline RL	59.01	48.18	65.14	56.95
9	Higher confidence	56.98	46.20	63.60	55.50
10	Random placebos	50.99	40.22	64.16	55.40

Table A1: Ablation results (%) on CIFAR-100, N=5.

G ABLATION STUDY FOR THE MEMORY BUDGET SETTINGS

This is supplementary to Section 4. In Table A2, Row 1 shows baselines. Rows 2 and 3 show the baselines with our PlaceboCIL as a plug-in module. Row 2 "budget" means the combined size of episodic memory and the placebo buffer is (strictly) equal to the episodic memory size in baseline methods. To this end, we randomly delete $|\mathcal{U}+\mathcal{P}|$ samples from the new class data $\mathcal{D}_{c_{i-1}+1:c_i}$ to obtain the buffer memory storing the unlabeled data batch $\mathcal{U}$ and $\mathcal{P}$. Row 3 "non-budget" means we are allowed to use a little additional memory to save the unlabelled data $\mathcal{U}$ and $\mathcal{P}$ without deleting other samples. Comparing Row 3 to Row 2, we can see that the performance of our PlaceboCIL only reduces 0.04 percentage points on average under the strict "budget" setting. The reason is the amount of new data is greatly larger than that of old samples, and removing a small batch of new data does not harm the performance much, as observed in (Liu et al., 2021b).

No.	Setting	LwF		iCaRL		LUCIR		AANets	
		Average	Last	Average	Last	Average	Last	Average	Last
1	Baseline	53.19	43.18	57.12	47.49	63.17	53.71	66.72	57.77
2	PlaceboCIL (budget)	59.06	48.56	61.01	51.45	65.42	56.18	67.16	59.14
3	PlaceboCIL (non-budget)	59.08	49.15	61.24	51.47	65.28	56.23	67.27	59.15

Table A2: Ablation results (%) on CIFAR-100, N=5."Average" and "Last" denote the average accuracy over all phases and the last-phase accuracy, respectively.

H EQUAL-SIZE SPLIT RESULTS

This is supplementary to Section 4 "**Ablation study**." In Table A3, we supplement the results using the equal-size split on CIFAR 100, 10-phase (10 classes/phase, 20 exemplars/class). We can observe that using placebos selected by heuristic evaluation functions (β_i=γ_i=1) consistently improves the results on three baselines. Using the online learning algorithm can further boost performance. The observations are similar to using the original setting in the main paper.

I ABLATION RESULTS FOR DIFFERENT UNLABELED DATA SOURCES

This is supplementary to Section 4 "**Ablation study**." We believe that the key to success is the design of our method instead of the choice of unlabeled data sources. To verify this, we provide

No.	Setting	LwF		iCaRL		LUCIR	
		Average	Last	Average	Last	Average	Last
1	Baseline	53.85	41.00	59.70	45.29	56.71	42.78
2	PlaceboCIL (ours)	57.68	42.44	62.32	46.52	57.89	44.08
3	PlaceboCIL (ours, w/o online learning)	56.31	41.76	61.05	46.02	57.01	43.13

Table A3: Supplementary to Table 3. Ablation results (%) using equal-size split on CIFAR-100, 10-phase (10 classes/phase).

the results for a new ablative setting: "using random unlabeled data" (to compute the distillation loss) in Table A4. We can observe that no matter what unlabeled data sources we use, our method consistently performs better than using random unlabeled data.

No.	Setting	LwF		iCaRL		LUCIR		AANets	
		Average	Last	Average	Last	Average	Last	Average	Last
1	Baseline	53.19	43.18	57.12	47.49	63.17	53.71	66.72	57.77
2	PlaceboCIL (ours, all)	59.29	49.64	61.17	50.96	65.48	56.77	67.33	59.32
3	PlaceboCIL (ours, overlapping)	58.95	48.71	62.15	52.62	65.73	57.26	67.48	59.06
4	Random unlabeled data (all)	50.99	40.22	56.27	46.64	64.16	55.45	66.23	57.22
4	Random unlabeled data (overlapping)	50.80	40.94	55.70	46.47	64.23	54.68	66.58	57.08

Table A4: Supplementary to Table 3. Ablation results (%) on CIFAR-100, N=5."Average" and "Last" denote the average accuracy over all phases and the last-phase accuracy, respectively. "All" denotes using all data from ImageNet, and "overlapping" means including samples from the overlapping classes between CIFAR-100 and ImageNet.

J REHEARSAL-FREE EXPERIMENTS

This is supplementary to Section 4 "**Ablation study**."

In Table A5, we provide the "rehearsal-free" results for two variants of our method: 1) PlacboCIL (original) w/o exemplars: we don't use any exemplars to compute the loss, only storing one exemplar each class to compute the evaluation function (Eq. 4); 2) PlacboCIL (higher confident) w/o exemplars: we don't use any exemplars, and select the unlabeled data with higher confidence as the placebos.

We can observe that using placebos is effective in the "rehearsal-free" setting. The original PlaceboCIL improves the average accuracy of the baseline by 6.30 percentage points. The "higher confident" version performs a little worse than the original, but it still improves the average accuracy of the baseline by 3.40 percentage points.

No.	Setting	Average	Last
1	Baseline	66.72	57.77
2	PlaceboCIL (original)	67.16	59.14
3	PlaceboCIL (higher confidence)	66.97	58.12
4	Baseline *w/o* exemplars	50.01	39.12
5	PlaceboCIL (original) *w/o* exemplars	56.31	44.02
6	PlaceboCIL (higher confidence) *w/o* exemplars	53.41	41.93

Table A5: Ablation results (%) on CIFAR-100, N=5. Baseline: LUCIR+AANets Liu et al. (2021a).