

# Overcoming Catastrophic Forgetting with Unlabeled Data in the Wild

Kibok Lee\* Kimin Lee† Jinwoo Shin† Honglak Lee\*  
\*University of Michigan †KAIST

\*{kibok, honglak}@umich.edu †{kiminlee, jinwoos}@kaist.ac.kr

## Abstract

Lifelong learning with deep neural networks is well-known to suffer from catastrophic forgetting: the performance on previous tasks drastically degrades when learning a new task. To alleviate this effect, we propose to leverage a large stream of unlabeled data easily obtainable in the wild. In particular, we design a novel class-incremental learning scheme with (a) a new distillation loss, termed global distillation, (b) a learning strategy to avoid overfitting to the most recent task, and (c) a confidence-based sampling method to effectively leverage unlabeled external data. Our experimental results on various datasets, including CIFAR and ImageNet, demonstrate the superiority of the proposed methods over prior methods, particularly when a stream of unlabeled data is accessible: our method shows up to 15.8% higher accuracy and 46.5% less forgetting compared to the state-of-the-art method. The code is available at <https://github.com/kibok90/iccv2019-inc>.

## 1. Introduction

Deep neural networks (DNNs) have achieved remarkable success in many machine learning applications, e.g., classification [10], generation [29], object detection [9], and reinforcement learning [39]. However, in the real world where the number of tasks continues to grow, the entire tasks cannot be given at once; rather, it may be given as a sequence of tasks. The goal of class-incremental learning [33] is to enrich the ability of a model dealing with such a case, by aiming to perform both previous and new tasks well.<sup>1</sup> In particular, it has gained much attention recently as DNNs tend to forget previous tasks easily when learning new tasks, which is a phenomenon called catastrophic forgetting [7, 28].

The primary reason of catastrophic forgetting is the limited resources for scalability: all training data of previous tasks cannot be stored in a limited size of memory as the number of tasks increases. Prior works in class-incremental learning focused on learning in a closed environment, i.e., a

<sup>1</sup>In class-incremental learning, a set of classes is given in each task. In evaluation, it aims to classify data in any class learned so far without task boundaries.

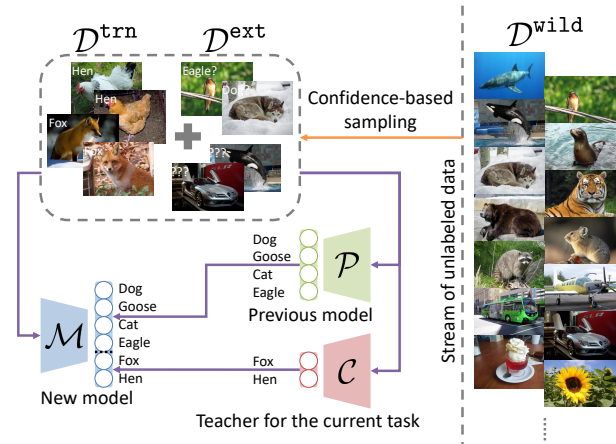


Figure 1. We propose to leverage a large stream of unlabeled data in the wild for class-incremental learning. At each stage, a confidence-based sampling strategy is applied to build an external dataset. Specifically, some of unlabeled data are sampled based on the prediction of the model learned in the previous stage  $\mathcal{P}$  for alleviating catastrophic forgetting, and some of them are randomly sampled for confidence calibration. Under the combination of the labeled training dataset and the unlabeled external dataset, a teacher model  $\mathcal{C}$  first learns the current task, and then the new model  $\mathcal{M}$  learns both the previous and current tasks by distilling the knowledge of  $\mathcal{P}$ ,  $\mathcal{C}$ , and their ensemble  $\mathcal{Q}$ .

model can only see the given labeled training dataset during training [3, 12, 23, 24, 33]. However, in the real world, we live with a continuous and large stream of unlabeled data easily obtainable on the fly or transiently, e.g., by data mining on social media [26] and web data [17]. Motivated by this, we propose to leverage such a large stream of unlabeled external data for overcoming catastrophic forgetting. We remark that our setup on unlabeled data is similar to self-taught learning [31] rather than semi-supervised learning, because we do not assume any correlation between unlabeled data and the labeled training dataset.

**Contribution.** Under the new class-incremental setup, our contribution is three-fold (see Figure 1 for an overview):

- A. We propose a new learning objective, termed global distillation, which utilizes data to distill the knowledge of reference models effectively.

- B. We design a 3-step learning scheme to improve the effectiveness of global distillation: (i) training a teacher specialized for the current task, (ii) training a model by distilling the knowledge of the previous model, the teacher learned in (i), and their ensemble, and (iii) fine-tuning to avoid overfitting to the current task.
- C. We propose a confidence-based sampling method to effectively leverage a large stream of unlabeled data.

In the contribution  $\mathcal{A}$ , global distillation encourages the model to learn knowledge over all previous tasks, while prior works only applied a task-wise local distillation [3, 12, 24, 33]. In particular, the proposed global distillation distills the knowledge of how to distinguish classes across different tasks, while local distillation does not. We show that the performance gain due to global distillation is particularly significant if some unlabeled external data are available.

In the contribution  $\mathcal{B}$ , the first two steps (i), (ii) of the proposed learning scheme are designed to keep the knowledge of the previous tasks, as well as to learn the current task. On the other hand, the purpose of the last step (iii) is to avoid overfitting to the current task: due to the scalability issue, only a small portion of data in the previous tasks are kept and replayed during training [3, 30, 33]. This inevitably incurs bias in the prediction of the learned model, being favorable for the current task. To mitigate the issue of imbalanced training, we fine-tune the model based on the statistics of data in the previous and current tasks.

Finally, the contribution  $\mathcal{C}$  is motivated from the intuition that as the data distribution of unlabeled data is more similar to that of the previous tasks, it prevents the model from catastrophic forgetting more. Since unlabeled data in the wild is not necessarily related to the previous tasks, it is far from being clear whether they contain an useful information for alleviating catastrophic forgetting. Therefore, we propose to sample an external dataset by a principled sampling strategy. To sample an effective external dataset from a large stream of unlabeled data, we propose to train a confidence-calibrated model [19, 20] by utilizing irrelevant data as out-of-distribution (OOD)<sup>2</sup> samples. We show that unlabeled data from OOD should also be sampled for maintaining the model to be more confidence-calibrated.

Our experimental results demonstrate the superiority of the proposed methods over prior methods. In particular, we show that the performance gain in the proposed methods is more significant when unlabeled external data are available. For example, under our experiment setup on ImageNet [6], our method with an external dataset achieves 15.8% higher accuracy and 46.5% less forgetting compared to the state-of-the-art method (E2E) [3] (4.8% higher accuracy and 6.0% less forgetting without an external dataset).

<sup>2</sup>Out-of-distribution refers to the data distribution being far from those of the tasks learned so far.

## 2. Approach

In this section, we propose a new learning method for class-incremental learning. In Section 2.1, we further describe the scenario and learning objectives. In Section 2.2, we propose a novel learning objective, termed global distillation. In Section 2.3, we propose a confidence-based sampling strategy to build an external dataset from a large stream of unlabeled data.

### 2.1. Preliminaries: Class-Incremental Learning

Formally, let  $(x, y) \in \mathcal{D}$  be a data  $x$  and its label  $y$  in a dataset  $\mathcal{D}$ , and let  $\mathcal{T}$  be a supervised task mapping  $x$  to  $y$ . We denote  $y \in \mathcal{T}$  if  $y$  is in the range of  $\mathcal{T}$  such that  $|\mathcal{T}|$  is the number of class labels in  $\mathcal{T}$ . For the  $t$ -th task  $\mathcal{T}_t$ , let  $\mathcal{D}_t$  be the corresponding training dataset, and  $\mathcal{D}_{t-1}^{\text{cor}} \subseteq \mathcal{D}_{t-1} \cup \mathcal{D}_{t-2}^{\text{cor}}$  be a coreset<sup>3</sup> containing representative data of previous tasks  $\mathcal{T}_{1:(t-1)} = \{\mathcal{T}_1, \dots, \mathcal{T}_{t-1}\}$ , such that  $\mathcal{D}_t^{\text{trn}} = \mathcal{D}_t \cup \mathcal{D}_{t-1}^{\text{cor}}$  is the entire labeled training dataset available at the  $t$ -th stage. Let  $\mathcal{M}_t = \{\theta, \phi_{1:t}\}$  be the set of learnable parameters of a model, where  $\theta$  and  $\phi_{1:t} = \{\phi_1, \dots, \phi_t\}$  indicate shared and task-specific parameters, respectively.<sup>4</sup>

The goal at the  $t$ -th stage is to train a model  $\mathcal{M}_t$  to perform the current task  $\mathcal{T}_t$  as well as the previous tasks  $\mathcal{T}_{1:(t-1)}$  without task boundaries, i.e., all class labels in  $\mathcal{T}_{1:t}$  are candidates at test time. To this end, a small coreset  $\mathcal{D}_{t-1}^{\text{cor}}$  and the previous model  $\mathcal{M}_{t-1}$  are transferred from the previous stage. We also assume that a large stream of unlabeled data is accessible, and an essential external dataset  $\mathcal{D}_t^{\text{ext}}$  is sampled, where the sampling method is described in Section 2.3. Note that we do not assume any correlation between the stream of unlabeled data and the tasks. The outcome at the  $t$ -th stage is the model  $\mathcal{M}_t$  that can perform all observed tasks  $\mathcal{T}_{1:t}$ , and the coreset  $\mathcal{D}_t^{\text{cor}}$  for learning in subsequent stages.

**Learning objectives.** When a dataset  $\mathcal{D}$  is labeled, the standard way of training a classification model  $\mathcal{M} = \{\theta, \phi\}$  is to optimize the cross-entropy loss:

$$\mathcal{L}_{\text{cls}}(\theta, \phi; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(x,y) \in \mathcal{D}} [-\log p(y|x; \theta, \phi)].$$

On the other hand, if we have a reference model  $\mathcal{R} = \{\theta^{\mathcal{R}}, \phi^{\mathcal{R}}\}$ , the dataset  $\mathcal{D}$  does not require any label because the target label is given by  $\mathcal{R}$ :

$$\begin{aligned} \mathcal{L}_{\text{dst}}(\theta, \phi; \mathcal{R}, \mathcal{D}) \\ = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \sum_{y \in \mathcal{T}} [-p(y|x; \theta^{\mathcal{R}}, \phi^{\mathcal{R}}) \log p(y|x; \theta, \phi)], \end{aligned}$$

<sup>3</sup>Coreset is a small dataset kept in a limited amount of memory used to replay previous tasks. Initially,  $\mathcal{D}_0^{\text{cor}} = \emptyset$ .

<sup>4</sup>If multiple task-specific parameters are given, then logits of all classes are concatenated for prediction without task boundaries. Note that tasks do not have to be disjoint, such that a class can appear in multiple tasks.

where the probabilities can be smoothed for better distillation (see [11] or the supplementary material).

**Previous approaches.** At the  $t$ -th stage, the standard approach to train a model  $\mathcal{M}_t$  is to minimize the following classification loss:

$$\mathcal{L}_{\text{cls}}(\theta, \phi_{1:t}; \mathcal{D}_t^{\text{trn}}). \quad (1)$$

However, in class-incremental learning, the limited size of the coreset makes the learned model suffer from catastrophic forgetting. To overcome this, the previous model  $\mathcal{P}_t = \{\theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}}\} \triangleq \mathcal{M}_{t-1}$  has been utilized to generate soft labels, which is the knowledge of  $\mathcal{P}_t$  about the given data [3, 12, 24, 33]:

$$\sum_{s=1}^{t-1} \mathcal{L}_{\text{dst}}(\theta, \phi_s; \mathcal{P}_t, \mathcal{D}_t^{\text{trn}}), \quad (2)$$

where this objective is jointly optimized with Eq. (1). We call this task-wise knowledge distillation as local distillation (LD), which transfers the knowledge within each of the tasks. However, because they are defined in a task-wise manner, this objective misses the knowledge about discrimination between classes in different tasks.

## 2.2. Global Distillation

Motivated by the limitation of LD, we propose to distill the knowledge of reference models *globally*. With the reference model  $\mathcal{P}_t$ , the knowledge can be globally distilled by minimizing the following loss:

$$\mathcal{L}_{\text{dst}}(\theta, \phi_{1:(t-1)}; \mathcal{P}_t, \mathcal{D}_t^{\text{trn}} \cup \mathcal{D}_t^{\text{ext}}). \quad (3)$$

However, learning by minimizing Eq. (3) would cause a bias: since  $\mathcal{P}_t$  did not learn to perform the current task  $\mathcal{T}_t$ , the knowledge about the current task would not be properly learned when only Eq. (1)+(3) are minimized, i.e., the performance on the current task would be unnecessarily sacrificed. To compensate for this, we introduce another teacher model  $\mathcal{C}_t = \{\theta^{\mathcal{C}}, \phi_t^{\mathcal{C}}\}$  specialized for the current task  $\mathcal{T}_t$ :

$$\mathcal{L}_{\text{dst}}(\theta, \phi_t; \mathcal{C}_t, \mathcal{D}_t^{\text{trn}} \cup \mathcal{D}_t^{\text{ext}}). \quad (4)$$

This model can be trained by minimizing the standard cross-entropy loss:

$$\mathcal{L}_{\text{cls}}(\theta^{\mathcal{C}}, \phi_t^{\mathcal{C}}; \mathcal{D}_t). \quad (5)$$

Note that only the dataset of the current task  $\mathcal{D}_t$  is used, because  $\mathcal{C}_t$  is specialized for the current task only. We revise this loss in Section 2.3 for better external data sampling.

However, as  $\mathcal{P}_t$  and  $\mathcal{C}_t$  learned to perform only  $\mathcal{T}_{1:(t-1)}$  and  $\mathcal{T}_t$ , respectively, discrimination between  $\mathcal{T}_{1:(t-1)}$  and  $\mathcal{T}_t$  is not possible with the knowledge distilled from these two

reference models. To complete the missing knowledge, we define  $\mathcal{Q}_t$  as an ensemble of  $\mathcal{P}_t$  and  $\mathcal{C}_t$ : let

$$p_{\max} = \max_y p(y|x, \theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}}),$$

$$y_{\max} = \arg \max_y p(y|x, \theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}}).$$

Then, the output of  $\mathcal{Q}_t$  can be defined as:

$$p(y|x, \theta^{\mathcal{Q}}, \phi_{1:t}^{\mathcal{Q}}) = \begin{cases} p_{\max} & \text{if } y = y_{\max}, \\ \frac{1-p_{\max}-\varepsilon}{1-p_{\max}} p(y|x, \theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}}) & \text{elif } y \in \mathcal{T}_{1:(t-1)}, \\ \varepsilon p(y|x, \theta^{\mathcal{C}}, \phi_t^{\mathcal{C}}) & \text{elif } y \in \mathcal{T}_t, \end{cases} \quad (6)$$

such that  $\sum_y p(y|x, \theta^{\mathcal{Q}}, \phi_{1:t}^{\mathcal{Q}}) = 1$ . Here,  $\varepsilon$  adjusts the confidence about whether the given data is in  $\mathcal{T}_{1:(t-1)}$  or  $\mathcal{T}_t$ . This information is basically missing, however, can be computed with an assumption that the expected predicted probability is the same over all negative classes  $\forall y \neq y_{\max}$ , i.e.,  $\mathbb{E}_y [p_\varepsilon(y|x, \theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}})] = \mathbb{E}_{y \neq y_{\max}} [p_\varepsilon(y|x, \theta^{\mathcal{C}}, \phi_t^{\mathcal{C}})]$ :

$$\varepsilon = \frac{(1-p_{\max})|\mathcal{T}_t|}{|\mathcal{T}_{1:t}|-1}. \quad (7)$$

Since the ensemble model  $\mathcal{Q}_t$  is able to perform all tasks, all parameters can be updated:

$$\mathcal{L}_{\text{dst}}(\theta, \phi_{1:t}; \mathcal{Q}_t, \mathcal{D}_t^{\text{ext}}). \quad (8)$$

Note that the labeled dataset  $\mathcal{D}_t^{\text{trn}}$  is not used, because it is already used in Eq. (1) for the same range of classes.

Finally, our global distillation (GD) model learns by optimizing Eq. (1)+(3)+(4)+(8):

$$\begin{aligned} & \mathcal{L}_{\text{cls}}(\theta, \phi_{1:t}; \mathcal{D}_t^{\text{trn}}) + \mathcal{L}_{\text{dst}}(\theta, \phi_{1:(t-1)}; \mathcal{P}_t, \mathcal{D}_t^{\text{trn}} \cup \mathcal{D}_t^{\text{ext}}) \\ & + \mathcal{L}_{\text{dst}}(\theta, \phi_t; \mathcal{C}_t, \mathcal{D}_t^{\text{trn}} \cup \mathcal{D}_t^{\text{ext}}) \\ & + \mathcal{L}_{\text{dst}}(\theta, \phi_{1:t}; \mathcal{Q}_t, \mathcal{D}_t^{\text{ext}}). \end{aligned} \quad (9)$$

We study the contribution of each term in Table 2.

**Balanced fine-tuning.** The statistics of class labels in the training dataset is also an information learned during training. Since the number of data from the previous tasks is much smaller than that of the current task, the prediction of the model is biased to the current task. To remove the bias, we further fine-tune the model after training with the same learning objectives. When fine-tuning, for each loss with  $\mathcal{D}$  and  $\mathcal{T}$ , we scale the gradient computed from a data with label  $k \in \mathcal{T}$  by the following:

$$w_{\mathcal{D}}^{(k)} \propto \frac{1}{|\{(x, y) \in \mathcal{D} | y = k\}|}. \quad (10)$$

Since scaling a gradient is equivalent to feeding the same data multiple times, we call this method *data weighting*.

---

**Algorithm 1** 3-step learning with GD.

---

```

1:  $t = 1$ 
2: while true do
3:   Input: previous model  $\mathcal{P}_t = \mathcal{M}_{t-1}$ , coreset  $\mathcal{D}_{t-1}^{\text{cor}}$ ,
      training dataset  $\mathcal{D}_t$ , unlabeled data stream  $\mathcal{D}_t^{\text{wild}}$ 
4:   Output: new coreset  $\mathcal{D}_t^{\text{cor}}$ , model  $\mathcal{M}_t = \{\theta, \phi_{1:t}\}$ 
5:    $\mathcal{D}_t^{\text{trn}} = \mathcal{D}_t \cup \mathcal{D}_{t-1}^{\text{cor}}$ 
6:    $N_C = |\mathcal{D}_{t-1}^{\text{cor}}|$ ,  $N_D = |\mathcal{D}_t^{\text{trn}}|$ 
7:   Sample  $\mathcal{D}_t^{\text{ext}}$  from  $\mathcal{D}_t^{\text{wild}}$  using Algorithm 2
8:   Train  $\mathcal{C}_t$  by minimizing Eq. (12)
9:   if  $t > 1$  then
10:    Train  $\mathcal{M}_t$  by minimizing Eq. (9)
11:    Fine-tune  $\phi_{1:t}$  by minimizing Eq. (9),
      with data weighting in Eq. (10)
12:   else
13:     $\mathcal{M}_t = \mathcal{C}_t$ 
14:   end if
15:   Randomly sample  $\mathcal{D}_t^{\text{cor}} \subseteq \mathcal{D}_t^{\text{trn}}$  such that
       $|\{(x, y) \in \mathcal{D}_t^{\text{cor}} | y = k\}| = N_C / |\mathcal{T}_{1:t}|$  for  $k \in \mathcal{T}_{1:t}$ 
16:    $t = t + 1$ 
17: end while

```

---

We also normalize the weights by multiplying them with  $|\mathcal{D}|/|\mathcal{T}|$ , such that they are all one if  $\mathcal{D}$  is balanced.

We only fine-tune the task-specific parameters  $\phi_{1:t}$  with data weighting, because all training data would be equally useful for representation learning, i.e., shared parameters  $\theta$ , while the bias in the data distribution of the training dataset should be removed when training a classifier, i.e.,  $\phi_{1:t}$ . The effect of balanced fine-tuning can be found in Table 4.

**Loss weight.** We balance the contribution of each loss by the relative size of each task learned in the loss: for each loss for learning  $\mathcal{T}$ , the loss weight at the  $t$ -th stage is

$$w^L = \frac{|\mathcal{T}|}{|\mathcal{T}_{1:t}|}. \quad (11)$$

We note that the loss weight can be tuned as a hyperparameter, but we find that this loss weight performs better than other values in general, as it follows the statistics of the test dataset: all classes are equally likely to be appeared.

**3-step learning algorithm.** In summary, our learning strategy has three steps: training  $\mathcal{C}_t$  specialized for the current task  $\mathcal{T}_t$ , training  $\mathcal{M}_t$  by distilling the knowledge of the reference models  $\mathcal{P}_t$ ,  $\mathcal{C}_t$ , and  $\mathcal{Q}_t$ , and fine-tuning the task-specific parameters  $\phi_{1:t}$  with data weighting. Algorithm 1 describes the 3-step learning scheme.

For coreset management, we build a balanced coreset by randomly selecting data for each class. We note that other more sophisticated selection algorithms like herding [33] do not perform significantly better than random selection, which is also reported in prior works [3, 42].

---

**Algorithm 2** Sampling external dataset.

---

```

1: Input: previous model  $\mathcal{P}_t = \{\theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}}\}$ ,
      unlabeled data stream  $\mathcal{D}_t^{\text{wild}}$ , sample size  $N_D$ ,
      number of unlabeled data to be retrieved  $N_{\text{max}}$ 
2: Output: sampled external dataset  $\mathcal{D}_t^{\text{ext}}$ 
3:  $\mathcal{D}^{\text{prev}} = \emptyset$ ,  $\mathcal{D}^{\text{OOD}} = \emptyset$ 
4:  $N_{\text{prev}} = 0.3N_D$ ,  $N_{\text{OOD}} = 0.7N_D$ 
5:  $N(k) \triangleq |\{(x, y, p) \in \mathcal{D}^{\text{prev}} | y = k\}|$ 
6: while  $|\mathcal{D}^{\text{OOD}}| < N_{\text{OOD}}$  do
7:   Get  $x \in \mathcal{D}_t^{\text{wild}}$  and update  $\mathcal{D}^{\text{OOD}} = \mathcal{D}^{\text{OOD}} \cup \{x\}$ 
8: end while
9:  $N_{\text{ret}} = N_{\text{OOD}}$ 
10: while  $N_{\text{ret}} < N_{\text{max}}$  do
11:   Get  $x \in \mathcal{D}_t^{\text{wild}}$  and compute the prediction of  $\mathcal{P}$ :
       $\hat{p} = \max_y p(y|x; \theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}})$ ,
       $\hat{y} = \arg \max_y p(y|x; \theta^{\mathcal{P}}, \phi_{1:(t-1)}^{\mathcal{P}})$ 
12:   if  $N(\hat{y}) < N_{\text{prev}}/|\mathcal{T}_{1:(t-1)}|$  then
13:      $\mathcal{D}^{\text{prev}} = \mathcal{D}^{\text{prev}} \cup \{(x, \hat{y}, \hat{p})\}$ 
14:   else
15:     Replace the least probable data in class  $\hat{y}$ :
       $(x', \hat{y}, p') = \arg \min_{(x, y, p) \in \mathcal{D}^{\text{prev}} | y = \hat{y}} p$ 
16:     if  $p' < \hat{p}$  then
17:        $\mathcal{D}^{\text{prev}} = (\mathcal{D}^{\text{prev}} \setminus \{(x', \hat{y}, p')\}) \cup \{(x, \hat{y}, \hat{p})\}$ 
18:     end if
19:   end if
20:    $N_{\text{ret}} = N_{\text{ret}} + 1$ 
21: end while
22: Return  $\mathcal{D}_t^{\text{ext}} = \mathcal{D}^{\text{OOD}} \cup \{x | (x, y, p) \in \mathcal{D}^{\text{prev}}\}$ 

```

---

### 2.3. Sampling External Dataset

Although a large amount of unlabeled data would be easily obtainable, there are two issues when using them for knowledge distillation: (a) training on a large-scale external dataset is expensive, and (b) most of the data would not be helpful, because they would be irrelevant to the tasks the model learns. To overcome these issues, we propose to sample an external dataset useful for knowledge distillation from a large stream of unlabeled data. Note that the sampled external dataset does not require an additional permanent memory; it is discarded after learning.

**Sampling for confidence calibration.** In order to alleviate catastrophic forgetting caused by the imbalanced training dataset, sampling external data that are expected to be in the previous tasks is desirable. Since the previous model  $\mathcal{P}$  is expected to produce an output with high confidence if the data is likely to be in the previous tasks, the output of  $\mathcal{P}$  can be used for sampling. However, modern DNNs are highly overconfident [8, 19], thus a model learned with a discriminative loss would produce a prediction with high confidence even if the data is not from any of the previous tasks. Since most of the unlabeled data would not be rele-

vant to any of the previous tasks, i.e., they are considered to be from out-of-distribution (OOD), it is important to avoid overconfident prediction on such irrelevant data. To achieve this, the model should learn to be confidence-calibrated by learning with a certain amount of OOD data as well as data of the previous tasks [19, 20]. When sampling OOD data, we propose to randomly sample data rather than relying on the confidence of the previous model, as OOD is widely distributed over the data space. The effect of this sampling strategy can be found in Table 5. Algorithm 2 describes our sampling strategy. The ratio of OOD data ( $N_{\text{prev}} : N_{\text{OOD}}$ ) is determined by validation; for more details, see the supplementary material. This sampling algorithm can take a long time, but we limit the number of retrieved unlabeled data in our experiment by 1M, i.e.,  $N_{\text{max}} = 1\text{M}$ .

**Confidence calibration for sampling.** For confidence calibration, we consider the following confidence loss  $\mathcal{L}_{\text{cnf}}$  to make the model produce confidence-calibrated outputs for data which are not relevant to the tasks the model learns:

$$\mathcal{L}_{\text{cnf}}(\theta, \phi; \mathcal{D}) = \frac{1}{|\mathcal{D}| |\mathcal{T}|} \sum_{x \in \mathcal{D}} \sum_{y \in \mathcal{T}} [-\log p(y|x; \theta, \phi)].$$

During the 3-step learning, only the first step for training  $\mathcal{C}_t$  has no reference model, so it should learn with the confidence loss. For  $\mathcal{C}_t$ ,  $(x, y)$  is from OOD if  $y \notin \mathcal{T}_t$ . Namely, by optimizing the confidence loss under the coreset of the previous tasks  $\mathcal{D}_{t-1}^{\text{cor}}$  and the external dataset  $\mathcal{D}_t^{\text{ext}}$ , the model learns to produce a prediction with low confidence for OOD data, i.e., uniformly distributed probabilities over class labels. Thus,  $\mathcal{C}_t$  learns by optimizing the following:

$$\mathcal{L}_{\text{cls}}(\theta^{\mathcal{C}}, \phi^{\mathcal{C}}; \mathcal{D}_t) + \mathcal{L}_{\text{cnf}}(\theta^{\mathcal{C}}, \phi^{\mathcal{C}}; \mathcal{D}_{t-1}^{\text{cor}} \cup \mathcal{D}_t^{\text{ext}}). \quad (12)$$

Note that the model  $\mathcal{M}_t$  does not require an additional confidence calibration, because the previous model  $\mathcal{P}_t$  is expected to be confidence-calibrated in the previous stage. Therefore, the confidence-calibrated outputs of the reference models are distilled to the model  $\mathcal{M}_t$ . The effect of confidence loss can be found in Table 3.

### 3. Related Work

**Continual lifelong learning.** Many recent works have addressed catastrophic forgetting with different assumptions. Broadly speaking, there are three different types of works [41]: one is class-incremental learning [3, 33, 42], where the number of class labels keeps growing. Another is task-incremental learning [12, 24], where the boundaries among tasks are assumed to be clear and the information about the task under test is given.<sup>5</sup> The last can be seen as data-incremental learning, which is the case when the set of class labels or actions are the same for all tasks [16, 35, 36].

<sup>5</sup>The main difference between class- and task-incremental learning is that the model has single- and multi-head output layer, respectively.

These works can be summarized as continual learning, and recent works on continual learning have studied two types of approaches to overcome catastrophic forgetting: model-based and data-based. Model-based approaches [1, 4, 14, 16, 21, 25, 27, 30, 34, 35, 36, 37, 43, 45] keep the knowledge of previous tasks by penalizing the change of parameters crucial for previous tasks, i.e., the updated parameters are constrained to be around the original values, and the update is scaled down by the importance of parameters on previous tasks. However, since DNNs have many local optima, there would be better local optima for both the previous and new tasks, which cannot be found by model-based approaches.

On the other hand, data-based approaches [3, 12, 13, 24, 33] keep the knowledge of the previous tasks by knowledge distillation [11], which minimizes the distance between the manifold of the latent space in the previous and new models. In contrast to model-based approaches, they require to feed data to get features on the latent space. Therefore, the amount of knowledge kept by knowledge distillation depends on the degree of similarity between the data distribution used to learn the previous tasks in the previous stages and the one used to distill the knowledge in the later stages. To guarantee to have a certain amount of similar data, some prior works [3, 30, 33] reserved a small amount of memory to keep a coreset, and others [22, 32, 38, 41, 42] trained a generative model and replay the generated data when training a new model. Note that the model-based and data-based approaches are orthogonal in most cases, thus they can be combined for better performance [15].

**Knowledge distillation in prior works.** Our proposed method is a data-based approach, but it is different from prior works [3, 12, 24, 33], because their model commonly learns with the task-wise local distillation loss in Eq. (2). We emphasize that local distillation only preserves the knowledge within each of the previous tasks, while global distillation does the knowledge over all tasks.

Similar to our 3-step learning, [36] and [12] utilized the idea of learning with two teachers. However, their strategy to keep the knowledge of the previous tasks is different: [36] applied a model-based approach, and [12] distilled the task-wise knowledge for task-incremental learning.

On the other hand, [3] had a similar fine-tuning, but they built a balanced dataset by discarding most of the data of the current task and updated the whole networks. However, such undersampling sacrifices the diversity of the frequent classes, which decreases the performance. Oversampling may solve the issue, but it makes the training not scalable: the size of the oversampled dataset increases proportional to the number of tasks learned so far. Instead, we propose to apply data weighting.

**Scalability.** Early works on continual learning were not scalable since they kept all previous models [2, 16, 24, 35,

43]. However, recent works considered the scalability by minimizing the amount of task-specific parameters [33, 36]. In addition, data-based methods require to keep either a coreset or a generative model to replay previous tasks. Our method is a data-based approach, but it does not suffer from the scalability issue since we utilize an external dataset sampled from a large stream of unlabeled data. We note that unlike coreset, our external dataset does not require a permanent memory; it is discarded after learning.

## 4. Experiments

### 4.1. Experimental Setup

**Compared algorithms.** To provide an upper bound of the performance, we compare an *oracle* method, which learns by optimizing Eq. (1) while storing all training data of previous tasks and replaying them during training. Also, as a *baseline*, we provide the performance of a model learned without knowledge distillation. Among prior works, three state-of-the-art methods are compared: *learning without forgetting (LwF)* [24], *distillation and retrospection (DR)* [12], and *end-to-end incremental learning (E2E)* [3]. For fair comparison, we adapt LwF and DR for class-incremental setting, which are originally evaluated in task-incremental learning setting: specifically, we extend the range of the classification loss, i.e., we optimize Eq. (1)+(2) and Eq. (1)+(2)+(4) for replication of them.

We do not compare model-based methods, because data-based methods are known to outperform them in class-incremental learning [22, 41], and they are orthogonal to data-based methods, such that they can potentially be combined with our approaches for better performance [15].

**Datasets.** We evaluate the compared methods on CIFAR-100 [18] and ImageNet ILSVRC 2012 [6], where all images are downsampled to  $32 \times 32$  [5]. For CIFAR-100, similar to prior works [3, 33], we shuffle the classes uniformly at random and split the classes to build a sequence of tasks. For ImageNet, we first sample 500 images per 100 randomly chosen classes for each trial, and then split the classes. To evaluate the compared methods under the environment with a large stream of unlabeled data, we take two large datasets: the TinyImages dataset [40] with 80M images and the entire ImageNet 2011 dataset with 14M images. The classes appeared in CIFAR-100 and ILSVRC 2012 are excluded to avoid any potential advantage from them. At each stage, our sampling algorithm gets unlabeled data from them uniformly at random to form an external dataset, until the number of retrieved samples is 1M.

Following the prior works, we divide the classes into splits of 5, 10, and 20 classes, such that there are 20, 10, and 5 tasks, respectively. For each task size, we evaluate the compared methods ten times with different class orders (different set of classes in the case of ImageNet) and report

the mean and standard deviation of the performance.

**Evaluation metric.** We report the performance of the compared methods in two metrics: the average incremental accuracy (ACC) and the average forgetting (FGT). For simplicity, we assume that the number of test data is the same over all classes. For a test data from the  $r$ -th task  $(x, y) \in \mathcal{D}_r^{\text{test}}$ , let  $\hat{y}(x; \mathcal{M}_s)$  be the label predicted by the  $s$ -th model, such that

$$A_{r,s} = \frac{1}{|\mathcal{D}_r^{\text{test}}|} \sum_{(x,y) \in \mathcal{D}_r^{\text{test}}} \mathbb{I}(\hat{y}(x; \mathcal{M}_s) = y)$$

measures the accuracy of the  $s$ -th model at the  $r$ -th task, where  $s \geq r$ . Note that prediction is done without task boundaries: for example, at the  $t$ -th stage, the expected accuracy of random guess is  $1/|\mathcal{T}_{1:t}|$ , not  $1/|\mathcal{T}_r|$ . At the  $t$ -th stage, ACC is defined as:

$$\text{ACC} = \frac{1}{t-1} \sum_{s=2}^t \sum_{r=1}^s \frac{|\mathcal{T}_r|}{|\mathcal{T}_{1:s}|} A_{r,s}.$$

Note that the performance of the first stage is not considered, as it is not class-incremental learning. While ACC measures the overall performance directly, FGT measures the amount of catastrophic forgetting, by averaging the performance decay:

$$\text{FGT} = \frac{1}{t-1} \sum_{s=2}^t \sum_{r=1}^{s-1} \frac{|\mathcal{T}_r|}{|\mathcal{T}_{1:s}|} (A_{r,r} - A_{r,s}),$$

which is essentially the negative of the backward transfer [25]. Note that smaller FGT is better, which implies that the model less-forgets about the previous tasks.

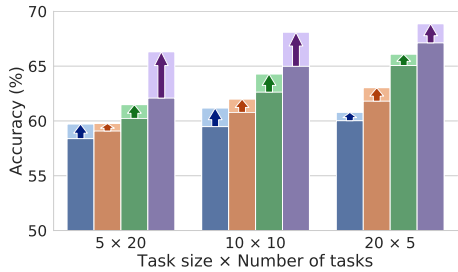
**Hyperparameters.** The backbone of all compared models is wide residual networks [44] with 16 layers, a widen factor of 2 (WRN-16-2), and a dropout rate of 0.3. Note that this has a comparable performance with ResNet-32 [10]. The last fully connected layer is considered to be a task-specific layer, and whenever a task with new classes comes in, the layer is extended to produce a prediction for the new classes. The number of parameters in the task-specific layer is small compared to those in shared layers (about 2% in maximum in WRN-16-2). All methods use the same size of coreset, which is 2000. For scalability, the size of the sampled external dataset is set to the size of the labeled dataset, i.e.,  $N_D = |\mathcal{D}_t^{\text{trn}}|$  in Algorithm 2. For validation, one split of ImageNet is used, which is exclusive to the other nine trials. The temperature for smoothing softmax probabilities [11] is set to 2 for distillation from  $\mathcal{P}$  and  $\mathcal{C}$ , and 1 for  $\mathcal{Q}$ . For more details, see the supplementary material.

### 4.2. Evaluation

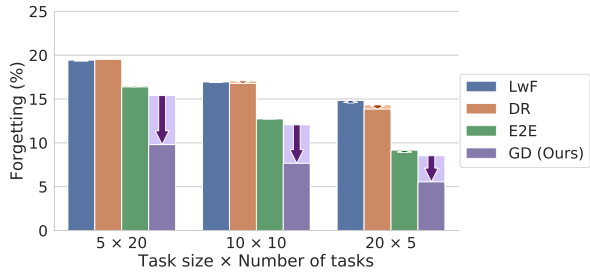
**Comparison of methods.** Table 1 and Figure 2 compare our proposed methods with the state-of-the-art methods. First, even when unlabeled data are not accessible,

Table 1. Comparison of methods on CIFAR-100 and ImageNet. We report the mean and standard deviation of ten trials for CIFAR-100 and nine trials for ImageNet with different random seeds in %.  $\uparrow$  ( $\downarrow$ ) indicates that the higher (lower) number is the better.

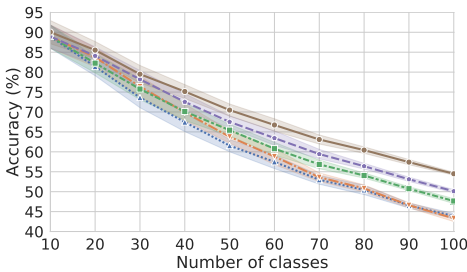
Dataset	CIFAR-100						ImageNet					
	5		10		20		5		10		20	
Task size	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )
Oracle	78.6 $\pm$ 0.9	3.3 $\pm$ 0.2	77.6 $\pm$ 0.8	3.1 $\pm$ 0.2	75.7 $\pm$ 0.7	2.8 $\pm$ 0.2	68.0 $\pm$ 1.7	3.3 $\pm$ 0.2	66.9 $\pm$ 1.6	3.1 $\pm$ 0.3	65.1 $\pm$ 1.2	2.7 $\pm$ 0.2
Baseline	57.4 $\pm$ 1.2	21.0 $\pm$ 0.5	56.8 $\pm$ 1.1	19.7 $\pm$ 0.4	56.0 $\pm$ 1.0	18.0 $\pm$ 0.3	44.2 $\pm$ 1.7	23.6 $\pm$ 0.4	44.1 $\pm$ 1.6	21.5 $\pm$ 0.5	44.7 $\pm$ 1.2	18.4 $\pm$ 0.5
LwF [24]	58.4 $\pm$ 1.3	19.3 $\pm$ 0.5	59.5 $\pm$ 1.2	16.9 $\pm$ 0.4	60.0 $\pm$ 1.0	14.5 $\pm$ 0.4	45.6 $\pm$ 1.9	21.5 $\pm$ 0.4	47.3 $\pm$ 1.5	18.5 $\pm$ 0.5	48.6 $\pm$ 1.2	15.3 $\pm$ 0.6
DR [12]	59.1 $\pm$ 1.4	19.6 $\pm$ 0.5	60.8 $\pm$ 1.2	17.1 $\pm$ 0.4	61.8 $\pm$ 0.9	14.3 $\pm$ 0.4	46.5 $\pm$ 1.6	22.0 $\pm$ 0.5	48.7 $\pm$ 1.6	18.8 $\pm$ 0.5	50.7 $\pm$ 1.2	15.1 $\pm$ 0.5
E2E [3]	60.2 $\pm$ 1.3	16.5 $\pm$ 0.5	62.6 $\pm$ 1.1	12.8 $\pm$ 0.4	65.1 $\pm$ 0.8	8.9 $\pm$ 0.2	47.7 $\pm$ 1.9	17.9 $\pm$ 0.4	50.8 $\pm$ 1.5	13.4 $\pm$ 0.4	53.9 $\pm$ 1.2	8.8 $\pm$ 0.3
GD (Ours)	62.1 $\pm$ 1.2	15.4 $\pm$ 0.4	65.0 $\pm$ 1.1	12.1 $\pm$ 0.3	67.1 $\pm$ 0.9	8.5 $\pm$ 0.3	50.0 $\pm$ 1.7	16.8 $\pm$ 0.4	53.7 $\pm$ 1.5	12.8 $\pm$ 0.5	56.5 $\pm$ 1.2	8.4 $\pm$ 0.4
+ ext	<b>66.3 <math>\pm</math> 1.2</b>	<b>9.8 <math>\pm</math> 0.3</b>	<b>68.1 <math>\pm</math> 1.1</b>	<b>7.7 <math>\pm</math> 0.3</b>	<b>68.9 <math>\pm</math> 1.0</b>	<b>5.5 <math>\pm</math> 0.4</b>	<b>55.2 <math>\pm</math> 1.8</b>	<b>9.6 <math>\pm</math> 0.4</b>	<b>57.7 <math>\pm</math> 1.6</b>	<b>7.4 <math>\pm</math> 0.3</b>	<b>58.7 <math>\pm</math> 1.2</b>	<b>5.4 <math>\pm</math> 0.3</b>



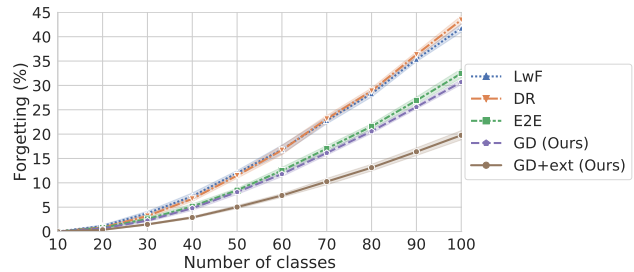
(a) ACC improvement by learning with external data



(b) FGT improvement by learning with external data



(c) ACC with respect to the number of trained classes



(d) FGT with respect to the number of trained classes

Figure 2. Experimental results on CIFAR-100. (a,b) Arrows show the performance gain in the average incremental accuracy (ACC) and average forgetting (FGT) by learning with unlabeled data, respectively. (c,d) Curves show ACC and FGT with respect to the number of trained classes when the task size is 10. We report the average performance of ten trials.

our method outperforms the state-of-the-art methods, which shows the effectiveness of the proposed 3-step learning scheme. Specifically, in addition to the difference in the loss function, DR does not have balanced fine-tuning, E2E lacks the teacher for the current task  $\mathcal{C}_t$  and fine-tunes the whole networks with a small dataset, and LwF has neither  $\mathcal{C}_t$  nor fine-tuning. Compared to E2E, which is the best state-of-the-art method, our method improves ACC by 4.8% and FGT by 6.0% on ImageNet with a task size of 5.

On the other hand, as shown in Figure 2(a)–2(b), learning with an unlabeled external dataset improves the performance of compared methods consistently, but the improvement is more significant in GD. For example, in the case of ImageNet with a task size of 5, by learning with the external dataset, E2E improves ACC by 3.2%, while GD does by 10.5%. Also, the relative performance gain in terms of FGT is more significant: E2E forgets 1.1% less while GD does 43.1%. Overall, with our proposed learning scheme and knowledge distillation with the external dataset, GD improves its ACC by 15.8% and FGT by 46.5% over E2E.

Table 2. Comparison of models learned with different reference models on CIFAR-100 when the task size is 10. “ $\mathcal{P}$ ,” “ $\mathcal{C}$ ,” and “ $\mathcal{Q}$ ” stand for the previous model, the teacher for the current task, and their ensemble model, respectively.

$\mathcal{P}$	$\mathcal{C}$	$\mathcal{Q}$	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )
$\checkmark$			62.9 $\pm$ 1.2	14.7 $\pm$ 0.4
$\checkmark$	$\checkmark$		67.0 $\pm$ 0.9	10.7 $\pm$ 0.3
		$\checkmark$	65.7 $\pm$ 0.9	11.2 $\pm$ 0.2
$\checkmark$	$\checkmark$	$\checkmark$	<b>68.1 <math>\pm</math> 1.1</b>	<b>7.7 <math>\pm</math> 0.3</b>

**Effect of the reference models.** Table 2 shows an ablation study with different set of reference models. As discussed in Section 2.2, because the previous model  $\mathcal{P}$  does not know about the current task, the compensation by introducing  $\mathcal{C}$  improves the overall performance. On the other hand,  $\mathcal{Q}$  does not show better ACC than the combination of  $\mathcal{P}$  and  $\mathcal{C}$ . This would be because, when building the output of  $\mathcal{Q}$ , the ensemble of the output of  $\mathcal{P}$  and  $\mathcal{C}$  is made with an assumption, which would not always be true. However, the

Table 3. Comparison of models learned with a different teacher for the current task  $\mathcal{C}$  on CIFAR-100 when the task size is 10. For “cls,”  $\mathcal{C}$  is not trained but the model learns by optimizing the learning objective of  $\mathcal{C}$  directly. The model learns with the proposed 3-step learning for “dst.” The confidence loss is added to the learning objective for  $\mathcal{C}$  for “cnf.” We do not utilize  $\mathcal{Q}$  for this experiment, because “cls” has no explicit  $\mathcal{C}$ .

$\mathcal{C}$	Confidence	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )
$\times$		$62.9 \pm 1.2$	$14.7 \pm 0.4$
cls		$62.9 \pm 1.3$	$14.5 \pm 0.5$
cls	cnf	$65.3 \pm 1.0$	$11.7 \pm 0.3$
dst		$66.2 \pm 1.0$	$11.2 \pm 0.3$
dst	cnf	<b><math>67.0 \pm 0.9</math></b>	<b><math>10.7 \pm 0.3</math></b>

Table 4. Comparison of different balanced learning strategies on CIFAR-100 when the task size is 10. “DW,” “FT-DSet,” and “FT-DW” stand for training with data weighting in Eq. (10) for the entire training, fine-tuning with a training dataset balanced by removing data of the current task, and fine-tuning with data weighting, respectively.

Balancing	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )
$\times$	$67.1 \pm 0.9$	$11.5 \pm 0.3$
DW	$67.9 \pm 0.9$	$9.6 \pm 0.2$
FT-DSet	$67.2 \pm 1.1$	$8.4 \pm 0.2$
FT-DW	<b><math>68.1 \pm 1.1</math></b>	<b><math>7.7 \pm 0.3</math></b>

knowledge from  $\mathcal{Q}$  is useful, such that the combination of all three reference models shows the best performance.

**Effect of the teacher for the current task  $\mathcal{C}$ .** Table 3 compares the models learned with a different teacher for the current task  $\mathcal{C}_t$ . In addition to the baseline without  $\mathcal{C}_t$ , we also compare the model directly optimizes the learning objective of  $\mathcal{C}_t$  in Eq. (5) or (12), i.e., the model learns with hard labels rather than soft labels when optimizing that loss. Note that introducing a separate model  $\mathcal{C}$  for distillation is beneficial, because  $\mathcal{C}$  learns better knowledge about the current task without interference from other classification tasks. Learning by optimizing the confidence loss improves the performance, because the confidence-calibrated model samples better external data as discussed in Section 2.3.

**Effect of balanced fine-tuning.** Table 4 shows the effect of balanced learning. First, balanced learning strategies improve FGT in general. If fine-tuning in 3-step learning is skipped but data weighting in Eq. (10) is applied in the main training (DW), the model shows higher FGT than having balanced fine-tuning on task-specific parameters (FT-DW), as discussed in Section 2.2. Note that data weighting (FT-DW) is better than removing the data of the current task to construct a small balanced dataset (FT-DSet) proposed in [3], because all training data are useful.

**Effect of external data sampling.** Table 5 compares different external data sampling strategies. Unlabeled data are beneficial in all cases, but the performance gain is differ-

Table 5. Comparison of different external data sampling strategies on CIFAR-100 when the task size is 10. “Prev” and “OOD” columns describe the sampling method for data of previous tasks and out-of-distribution data, where “Pred” and “Random” stand for sampling based on the prediction of the previous model  $\mathcal{P}$  and random sampling, respectively. In particular, for when sampling OOD by “Pred,” we sample data minimizing the confidence loss  $\mathcal{L}_{\text{cnf}}$ . When only Prev or OOD is sampled, the number of sampled data is matched for fair comparison.

Prev	OOD	ACC ( $\uparrow$ )	FGT ( $\downarrow$ )
$\times$	$\times$	$65.0 \pm 1.1$	$12.1 \pm 0.3$
$\times$	Random	$67.6 \pm 0.9$	$9.0 \pm 0.3$
Pred	$\times$	$66.0 \pm 1.2$	$7.8 \pm 0.3$
Pred	Pred	$65.7 \pm 1.1$	$10.2 \pm 0.2$
Pred	Random	<b><math>68.1 \pm 1.1</math></b>	<b><math>7.7 \pm 0.3</math></b>

ent over sampling strategies. First, observe that randomly sampled data are useful, because their predictive distribution would be diverse such that it helps to learn the diverse knowledge of the reference models, which makes the model confidence-calibrated. However, while the random sampling strategy has higher ACC than sampling based on the prediction of the previous model  $\mathcal{P}$ , it also shows high FGT. This implies that the unlabeled data sampled based on the prediction of  $\mathcal{P}$  prevents the model from catastrophic forgetting more. As discussed in Section 2.3, our proposed sampling strategy, the combination of the above two strategies shows the best performance. Finally, sampling OOD data based on the prediction of  $\mathcal{P}$  is not beneficial, because “data most likely to be from OOD” would not be useful. OOD data sampled based on the prediction of  $\mathcal{P}$  have almost uniform predictive distribution, which would be locally distributed. However, the concept of OOD is a kind of complement set of the data distribution the model learns. Thus, to learn to discriminate OOD well in our case, the model should learn with data widely distributed outside of the data distribution of the previous tasks.

## 5. Conclusion

We propose to leverage a large stream of unlabeled data in the wild for class-incremental learning. The proposed global distillation aims to keep the knowledge of the reference models without task boundaries, leading better knowledge distillation. Our 3-step learning scheme effectively leverages the external dataset sampled by the confidence-based sampling strategy from the stream of unlabeled data.

## Acknowledgements

This work was supported in part by Kwanjeong Educational Foundation Scholarship, NSF CAREER IIS-1453651, and Sloan Research Fellowship. We also thank Lajanugen Logeswaran, Jongwook Choi, Yijie Guo, Wilka Carvalho, and Yunseok Jang for helpful discussions.



## References

- [1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, 2018.
- [2] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *CVPR*, 2017.
- [3] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, 2018.
- [4] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, 2018.
- [5] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.
- [6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
- [7] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.
- [8] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *ICML*, 2017.
- [9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.
- [10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
- [12] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *ECCV*, 2018.
- [13] Khurram Javed and Faisal Shafait. Revisiting distillation and incremental classifier learning. In *ACCV*, 2018.
- [14] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetful learning for domain expansion in deep neural networks. In *AAAI*, 2018.
- [15] Hyo-Eun Kim, Seungwook Kim, and Jaehwan Lee. Keep and learn: Continual learning by constraining the latent space for knowledge preservation in neural networks. In *MICCAI*, 2018.
- [16] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 2017.
- [17] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *ECCV*, 2016.
- [18] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.
- [19] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018.
- [20] Kibok Lee, Kimin Lee, Kyle Min, Yuting Zhang, Jinwoo Shin, and Honglak Lee. Hierarchical novelty detection for visual object recognition. In *CVPR*, 2018.
- [21] Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. In *NeurIPS*, 2017.
- [22] Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Andrei Stoian, and David Filliat. Generative models from the perspective of continual learning. *arXiv preprint arXiv:1812.09111*, 2018.
- [23] Yu Li, Zhongxiao Li, Lizhong Ding, Peng Yang, Yuhui Hu, Wei Chen, and Xin Gao. Supportnet: solving catastrophic forgetting in class incremental learning with support data. *arXiv preprint arXiv:1806.02942*, 2018.
- [24] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *ECCV*, 2016.
- [25] David Lopez-Paz et al. Gradient episodic memory for continual learning. In *NeurIPS*, 2017.
- [26] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *ECCV*, 2018.
- [27] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *ECCV*, 2018.
- [28] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*, 24:109–165, 1989.
- [29] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.
- [30] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *ICLR*, 2018.
- [31] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *ICML*, 2007.
- [32] Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *ICCV*, 2017.
- [33] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017.
- [34] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *NIPS*, 2018.
- [35] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.
- [36] Jonathan Schwarz, Jelena Luketina, Wojciech M Czarnecki, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *ICML*, 2018.

- [37] Joan Serra, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, 2018.
- [38] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NeurIPS*, 2017.
- [39] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 2017.
- [40] Antonio Torralba, Rob Fergus, and William T Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *PAMI*, 30(11):1958–1970, 2008.
- [41] Gido M van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635*, 2018.
- [42] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, Zhengyou Zhang, and Yun Fu. Incremental classifier learning with generative adversarial networks. *arXiv preprint arXiv:1802.00853*, 2018.
- [43] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *ICLR*, 2018.
- [44] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.
- [45] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, 2017.