SS-IL: Separated Softmax for Incremental Learning

Hongjoon Ahn[1]*, Jihwan Kwak[4]*, Subin Lim[3], Hyeonsu Bang[1], Hyojun Kim[2] and Taesup Moon[4]†
[1] Department of Artificial Intelligence, [2] Department of Electronic and Electrical Engineering,
[3] Department of Computer Engineering, Sungkyunkwan University, Suwon, Korea
[4] Department of Electrical and Computer Engineering, Seoul National University, Seoul, Korea
{hong0805, tnqls985, bhs1996, leopard101}@skku.edu
{jihwan0508, tsmoon}@snu.ac.kr

Abstract

We consider class incremental learning (CIL) problem, in which a learning agent continuously learns new classes from incrementally arriving training data batches and aims to predict well on all the classes learned so far. The main challenge of the problem is the catastrophic forgetting, and for the exemplar-memory based CIL methods, it is generally known that the forgetting is commonly caused by the classification score bias that is injected due to the data imbalance between the new classes and the old classes (in the exemplar-memory). While several methods have been proposed to correct such score bias by some additional post-processing, e.g., score re-scaling or balanced fine-tuning, no systematic analysis on the root cause of such bias has been done. To that end, we analyze that computing the softmax probabilities by combining the output scores for all old and new classes could be the main cause of the bias. Then, we propose a new method, dubbed as Separated Softmax for Incremental Learning (SS-IL), that consists of separated softmax (SS) output layer combined with task-wise knowledge distillation (TKD) to resolve such bias. Throughout our extensive experimental results on several large-scale CIL benchmark datasets, we show our SS-IL achieves strong state-of-the-art accuracy through attaining much more balanced prediction scores across old and new classes, without any additional post-processing.

1. Introduction

Incremental or continual learning, in which the agent continues to learn incremental arrival of new training data, is one of the grand challenges in artificial intelligence and machine learning. Such setting, which does not assume the full availability of old training data, is recently gaining more attention particularly from the perspective of real-world applications. The reason is storing all the training data, which can easily become large-scale, in one batch often becomes unrealistic for memory- and computation-constrained applications, such as mobile phones or robots. Therefore, the continuous yet effective update of the learning agent without accessing the full data received so far is indispensable.

A viable candidate for such agent is the end-to-end learning based deep neural network (DNN) models. Following the recent success in many different applications [14, 3, 7], the DNN-based incremental learning methods have been also actively pursued in recent years. Despite some promising results, they also possess a critical limitation: the *catastrophic forgetting*, which refers to the problem that the generalization performance on the old data severely degrades after a naive fine-tuning of the model with the new data.

In this paper, we focus on the DNN-based *class* incremental learning (CIL), which we refer to learning a classifier to classify *new* object classes from every incremental training data and testing the classifier on all the classes learned so far. Among several different proposed approaches, the exemplar-memory based ones [28, 8, 31, 35, 4, 5], which allow to store small amount of training data from old classes in a separate memory, have been shown to be effective in mitigating the catastrophic forgetting.

The main challenge of using the exemplar-memory in CIL is to resolve the severe data imbalance between the training data points for the new classes and for the old classes (in the exemplar-memory). That is, the naive fine-tuning with such imbalanced data would heavily skew the prediction scores toward the newly learned classes, hence, the accuracy for the old classes would dramatically drop, resulting in a significant forgetting. Recently, several state-of-the-art methods [8, 31, 35, 4, 5] proposed to correct such score bias by some additional post-processing steps, *e.g.*, score re-scaling or balanced fine-tuning, after learning the classification models.

While above mentioned methods were effective to some extent in terms of improving the accuracy, we argue that they lack systematic analyses on the main reason of such bias and that some component of their schemes, *e.g.*, knowledge distillation (KD) [15], was naively used without any

*Equal contribution.
†Corresponding autho r.

proper justifications [31, 23, 35, 20]. To that regard, in this paper, we first analyze the root cause of such classification score bias, then propose a method that mitigates the cause in a sensible way. Namely, we argue that the bias is injected by the fact that the softmax probability used in the ordinary cross-entropy loss is always computed by combining the output scores of *all* classes, which forces the heavy penalization of the output scores for the old classes due to the data imbalance. Furthermore, we show that a naive use of the General KD (GKD) method, which also combines the output scores of *all* old classes to compute the soft target, may *preserve* the bias and even hurt the accuracy, if the prediction bias is already present in the model.

To resolve above issues, we propose Separated-Softmax for Incremental Learning (SS-IL), which consists of two main components. Firstly, we devise *separated softmax (SS)* output layer that mutually blocks the flow of the score gradients between the old and new classes, thus, mitigates the imbalanced penalization of the output probabilities for the old classes. Secondly, we show the *Task-wise KD (TKD)* [25], which also computes the soft target for the distillation in a task-separated manner, is particularly well-suited for our SS layer, since it attempts to preserve the task-wise knowledge without preserving the prediction bias that may remain across the tasks. In order to show the effectiveness of our approach, we carried out *extensive* experimental validations on several large-scale CIL benchmarks with various scenarios and fairly compared our SS-IL with recent strong baselines by reproducing all of them. As a result, we convincingly show our SS-IL achieves strong state-of-the-art accuracy via adequately balancing the prediction scores across old and new classes, without *any* additional post-processing.

In summary, our contribution is threefold:

- We propose a novel *separated softmax (SS)* layer, which prevents the old class scores from being overly penalized throughout the gradient steps.
- We show that using GKD in CIL may preserve the bias of the model, while TKD can bring synergy when particularly combined with SS that has the same intuition.
- We carry out extensive experimental validation of our SS-IL on several large-scale benchmarks with various CIL scenarios and fairly compared with recent, all-reproduced state-of-the-art baselines.

2. Related Work

Recently, there has been a plethora of work done to tackle the catastrophic forgetting problem in continual/incremental learning. For general continual learning, there has been three main approaches; 1) regularization-based [22, 34, 9, 2, 19], 2) dynamic architecture-based [29, 33, 24, 17], and 3) exemplar/replay-memory based [26, 10, 11, 30, 21, 32, 30, 21, 32] methods. For a more thorough survey, we refer the readers to [27].

For CIL, in particular, the exemplar-memory based method combined with knowledge distillation (KD) has been shown to be effective. We summarize the representative recent work in the following.

Using exemplar-memory and bias correction In CIL, early exemplar-memory based approaches, *e.g.*, iCaRL [28] and EEIL [8], have shown superior results. iCaRL classifies the examples using Nearest Mean of Exemplars (NME), and EEIL additionally exploits balanced fine-tuning, which further fine-tunes the network with a balanced training batches. Later, Javed et al. [18] points out that methods using exemplar-memory cause imbalanced dataset and consequently have been shown to suffer from the bias problem in the final FC layer. To tackle this imbalanced learning problem, several bias removal techniques have been proposed. Another balanced fine-tuning approach UW [23] utilizes gradient scaling by weighting the losses based on the statistics of the training data. BiC [31] corrects the bias of scores by additionally training a bias correction layer, and WA [35] corrects the biased weights in the FC layer based on the norm of each weight. Moreover, IL2M [4] rectifies the output softmax probability and ScaIL [5] scales the classifier weights, by utilizing statistical properties of the model outputs.

Knowledge distillation (KD) KD has been widely used in CIL as a popular technique to preserve and leverage the information learned from the old classes, so that the forgetting could be mitigated. However, as mentioned in the Introduction, several versions of KD have been confusingly used in different methods without proper justifications. Namely, for example, LwF [25], iCaRL [28], and EEIL [8] utilized the form of TKD, whereas BiC [31], UW [23], and WA [35] used the form of GKD. Each method simply made its choice, and no analysis or justification on the choice is given other than some intuitive arguments (*e.g.*, [23] justifies GKD, but, without proper comparison or evidence.)

Apart from above methods, LUCIR [16] and PODNet [13] considered a slightly different setting, in which an initial model is trained with large number of base classes to get useful feature representations, and they utilize the *feature* distillation to preserve those representations while learning the future classes. However, we believe their setting is more limited, and we compare their performances in the *pure* CIL setting to make a fair comparison with other baselines.

3. Preliminaries

3.1. Notations and problem setting

In CIL, we assume every incrementally-arrived training data, which is often called as the incremental *task*, consists of data for *new* m classes that have not been learned before. More formally, the training data for the incremental

task t is denoted by $\mathcal{D}_t = \{(\boldsymbol{x}_t^{(i)}, y_t^{(i)})\}_{i=1}^{n_t}$, in which $\boldsymbol{x}_t^{(i)}$, $y_t^{(i)}$, and n_t denote input data for task t, the corresponding (integer-valued) target label, and the number of training samples for the corresponding task, respectively. The total number of classes up to task t is denoted by $C_t = m \cdot t$, which leads to the labeling $y_t^{(i)} \in \{C_{t-1}+1, \ldots, C_t\} \triangleq \mathcal{C}_t$. During learning each incremental task, we assume a separate exemplar-memory $\mathcal{M}$ is allocated to store exemplar data for old classes. Namely, when learning the incremental task t, we store $\lfloor \frac{|\mathcal{M}|}{C_{t-1}} \rfloor$ data points from each class that are learnt until the incremental task $t-1$. Thus, as the incremental task grows, the number of exemplar data points stored for each class decreases linearly with t and we assume $|\mathcal{M}| \ll n_t$. The total number of incremental tasks is denoted by T.

Our classification model consists of a feature extractor, which has the deep convolutional neural network (CNN) architecture, and the classification layer, which is the final fully-connected (FC) layer with softmax output. We denote $\boldsymbol{\theta}$ as the parameters for our classification model. At incremental task t, the parameters of the model, $\boldsymbol{\theta}_t$, are learned using data points in $\mathcal{D}_t \cup \mathcal{M}$. After learning, the class prediction for a given sample $\boldsymbol{x}_{\text{test}}$ is obtained by

$$\hat{y}_{\text{test}} = \arg\max_{y \in \mathcal{C}_{1:t}} z_{ty}(\boldsymbol{x}_{\text{test}}, \boldsymbol{\theta}_t), \tag{1}$$

in which $z_{ty}(\boldsymbol{x}_{\text{test}}, \boldsymbol{\theta}_t)$ is the output score (before softmax) of the model $\boldsymbol{\theta}_t$ for class $y \in \{1, \ldots, C_t\} \triangleq \mathcal{C}_{1:t}$. Namely, at test time, the final FC layers are consolidated and the prediction among all classes in $\mathcal{C}_{1:t}$ is made as if by an ordinary multi-class classifier.

3.2. Knowledge distillation

As previously mentioned, the two main variations of KD used in CIL are General KD (GKD) and Task-wise KD (TKD), and the loss function defined for each method for learning task t is as follows: for an input data $\boldsymbol{x} \in \mathcal{D}_t \cup \mathcal{M}$,

$$\mathcal{L}_{\text{GKD},t}(\boldsymbol{x}, \boldsymbol{\theta}) \triangleq \mathcal{D}_{KL}(\boldsymbol{p}^{\tau}_{1:t-1}(\boldsymbol{x}, \boldsymbol{\theta}_{t-1}) \| \boldsymbol{p}^{\tau}_{1:t-1}(\boldsymbol{x}, \boldsymbol{\theta})) \tag{2}$$

$$\mathcal{L}_{\text{TKD},t}(\boldsymbol{x}, \boldsymbol{\theta}) \triangleq \sum_{s=1}^{t-1} \mathcal{D}_{KL}(\boldsymbol{p}^{\tau}_{s}(\boldsymbol{x}, \boldsymbol{\theta}_{t-1}) \| \boldsymbol{p}^{\tau}_{s}(\boldsymbol{x}, \boldsymbol{\theta})), \tag{3}$$

in which $\mathcal{D}_{KL}(\cdot\|\cdot)$ is the Kullback-Leibler divergence, τ is a temperature scaling parameter, $\boldsymbol{\theta}$ are the model parameters that are being learned for task t, and $\boldsymbol{\theta}_{t-1}$ are the model parameters *learned* up to task $t-1$. Furthermore, in (2) and (3), we define the c-th component of the probability vectors $\boldsymbol{p}^{\tau}_{s}(\boldsymbol{x}, \boldsymbol{\theta}) \in \Delta^{m}$ and $\boldsymbol{p}^{\tau}_{1:s}(\boldsymbol{x}, \boldsymbol{\theta}) \in \Delta^{C_s}$ as

$$p^{\tau}_{s,c}(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{e^{z_{sc}(\boldsymbol{x}, \boldsymbol{\theta})/\tau}}{\sum_{k \in \mathcal{C}_s} e^{z_{sk}(\boldsymbol{x}, \boldsymbol{\theta})/\tau}} \quad \text{and}$$

$$p^{\tau}_{1:s,c}(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{e^{z_{sc}(\boldsymbol{x}, \boldsymbol{\theta})/\tau}}{\sum_{k \in \mathcal{C}_{1:s}} e^{z_{sk}(\boldsymbol{x}, \boldsymbol{\theta})/\tau}},$$

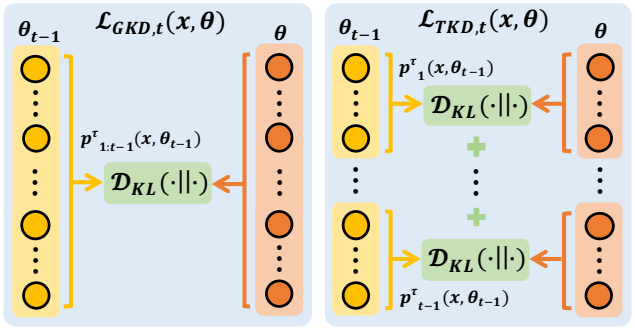

Figure 1. Illustration of $\mathcal{L}_{\text{GKD},t}(\boldsymbol{x}, \boldsymbol{\theta})$ (left) and $\mathcal{L}_{\text{TKD},t}(\boldsymbol{x}, \boldsymbol{\theta})$ (right)

respectively. Namely, $\boldsymbol{p}^{\tau}_{s}(\boldsymbol{x}, \boldsymbol{\theta})$ is the probability vector obtained by *only* using the output scores for task s when computing the softmax probability, and $\boldsymbol{p}^{\tau}_{1:s}(\boldsymbol{x}, \boldsymbol{\theta})$ is the probability vector obtained by using all the output scores for tasks from 1 to s when computing the softmax probability. Thus, minimizing (2) or (3) will both result in regularizing the past model $\boldsymbol{\theta}_{t-1}$, but (2) uses the global softmax probability across all past tasks, $\boldsymbol{p}^{\tau}_{1:t-1}(\boldsymbol{x}, \boldsymbol{\theta}_{t-1})$, while (3) uses the task-wise softmax probabilities, $\{\boldsymbol{p}^{\tau}_{s}(\boldsymbol{x}, \boldsymbol{\theta}_{t-1}))\}_{s=1}^{t-1}$, obtained separately for each task. In recent CIL baselines, (2) is used in [31, 23, 35], and (3) is used in [25, 8]. The difference between (2) and (3) is illustrated in Figure 1.

4. Motivation

As mentioned in the Introduction, several previous works [8, 23, 16, 31, 4, 35, 5] identified that the major challenge of the exemplar-memory based CIL is to resolve the classification score bias caused by the data imbalance. Here, we consider a simple example and give a convincing argument on why such score bias in injected as well as why a naive usage of GKD cannot fix the bias.

Namely, first note that the ordinary cross-entropy loss for learning task t used by the typical CIL methods can be expressed as

$$\mathcal{L}_{\text{CE},t}((\boldsymbol{x}, y), \boldsymbol{\theta}) = \mathcal{D}_{KL}(\boldsymbol{y}_{1:t} \| \boldsymbol{p}_{1:t}(\boldsymbol{x}, \boldsymbol{\theta})), \tag{4}$$

in which $\boldsymbol{y}_{1:t}$ is a one-hot vector in $\mathbb{R}^{C_t}$ that has value one at the y-th coordinate, and $\boldsymbol{p}_{1:t}(\boldsymbol{x}, \boldsymbol{\theta})$ is $\boldsymbol{p}^{\tau}_{1:t}(\boldsymbol{x}, \boldsymbol{\theta})$ with $\tau = 1$. Now, in order to systematically analyze the root cause of the prediction bias commonly presents in typical CIL methods, we carried out an experiment with a simple CIL method that uses the following loss

$$\mathcal{L}_{\text{CE},t}((\boldsymbol{x}, y), \boldsymbol{\theta}) + \mathcal{L}_{\text{GKD},t}(\boldsymbol{x}, \boldsymbol{\theta}) \tag{5}$$

with $(\boldsymbol{x}, y) \in \mathcal{D}_t \cup \mathcal{M}$ for learning task t. Namely, it learns the task t with the cross-entropy loss while trying to preserve past knowledge by $\mathcal{L}_{\text{GKD}}$. As shown in Figure 2, we experimented with the ImageNet dataset with $m = 100$ and $|\mathcal{M}| = 10k$, hence with total 10 tasks.

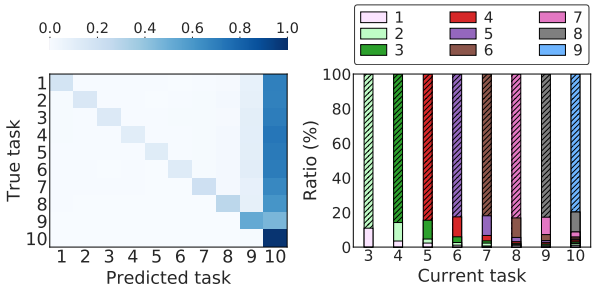

Figure 2. Left: The confusion matrix (across the tasks) based on the predictions of CIL model for test data. Right: The ratio of Top-1 predictions made by $\boldsymbol{\theta}_{t-1}$ on $\mathcal{D}_t$. Instead of denoting the predicted classes, we denote the tasks to which the predicted classes belong. Note that the dashed region in the right plot indicates the ratio of the latest old task, and it represents the bias on soft targets used for $\mathcal{L}_{\text{GKD}}$. All the results are on the ImageNet-1K dataset with $m = 100$ and $|\mathcal{M}| = 10k$.

4.1. Bias caused by ordinary cross-entropy

The left plot in Figure 2 shows the confusion matrix of test samples at the task level after learning all the tasks. It clearly shows the common prediction bias; namely, most of the prediction for past tasks are overly biased toward the most recent task (task 10). We argue that the root cause of this bias can be found in the gradient for the output score:

$$\frac{\partial \mathcal{L}_{\text{CE},t}((\boldsymbol{x},y),\boldsymbol{\theta})}{\partial z_{tc}} = p_{1:t,c}(\boldsymbol{x},\boldsymbol{\theta}) - \mathbb{1}_{\{c=y\}}, \tag{6}$$

in which $\mathbb{1}_{\{c=y\}}$ is the indicator for $c = y$. Note that since (6) is always positive for $c \neq y$, we can easily observe that when the model is being updated with data in $\mathcal{D}_t \cup \mathcal{M}$, the classification scores for the old classes will continue to decrease during the gradient descent steps done for the abundant samples for the new classes in $\mathcal{D}_t$. Thus, we believe that these imbalanced gradient descent steps for the classification scores of the old classes make the significant score bias toward the new classes. The toy illustration of gradient descent steps is illustrated in Figure 3.

4.2. Bias preserved by GKD

Now, as mentioned above, several previous works use GKD for the purpose of preserving the knowledge learned from past tasks. However, when the gradient from the cross-entropy loss causes a significant bias as mentioned in the previous section, we argue that using GKD would preserve such bias in the older model and even could hurt the performance. In $\mathcal{L}_{\text{GKD}}$ defined in (2), $\boldsymbol{p}^{\tau}_{1:t-1}(\boldsymbol{x},\boldsymbol{\theta}_{t-1})$ is the *soft* target computed from the *old* model $\boldsymbol{\theta}_{t-1}$, that is used for knowledge distillation. Now, Figure 2 (right) suggests that this soft target can be heavily skewed due to the bias caused by the cross-entropy learning. Namely, the figure shows the ratio of the *tasks* among $\{1,\ldots,t-1\}$, predicted by the *old* model $\boldsymbol{\theta}_{t-1}$ when the *new* task data points $\boldsymbol{x} \in \mathcal{D}_t$ were given as input, for each new task t (horizontal axis).

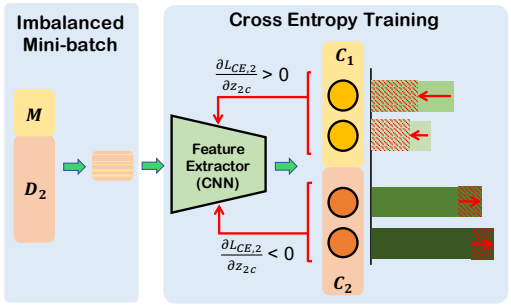

Figure 3. A toy illustration of gradient descent steps for $m = 2$ and $T = 2$ on imbalanced $\mathcal{D}_2 \cup \mathcal{M}$. The scores for class $c \in \mathcal{C}_1$ continue to decrease due to the imbalanced gradient descent steps.

We can observe that the predictions are overwhelmingly biased toward the most recent old task (*i.e.*, task $t-1$), which is due to the bias generated during learning task $t-1$ with the cross-entropy loss. This suggests that the soft target $\boldsymbol{p}^{\tau}_{1:t-1}(\boldsymbol{x},\boldsymbol{\theta}_{t-1})$ also would be heavily skewed toward the most recent old task (task $t-1$), hence, when it is used in GKD loss as (2), it will preserve such bias and could highly penalize the output probabilities of the *older* tasks. Hence, it could make the bias, or the *forgetting* of the older tasks, more severe.

Above two observations suggest that the main reason for the prediction bias could be to compute the softmax probability by combining the old and new tasks *altogether*. Motivated by this, we propose Separated-Softmax for Incremental Learning (SS-IL) in the next section.

5. Main Method

Our SS-IL mainly consists of two components, all motivated from the intuition built from the previous section: (1) Separated-Softmax (SS) output layer and (2) the Task-wise KD (TKD). We note using TKD for CIL was first proposed in LwF [25], but as we show in our experiment, TKD particularly becomes powerful when combined with our SS layer. For the sake of simple explanation, at the incremental task t, let $\mathcal{P}_t$ denote the classes of the previous tasks ($\mathcal{C}_{1:t-1}$) and $\mathcal{N}_t$ denote the classes of the new task ($\mathcal{C}_t$).

(1) Separated-Softmax (SS) layer: For $(\boldsymbol{x},y) \in \mathcal{D}_t \cup \mathcal{M}$, we define a separate softmax output layer by modifying the cross-entropy loss function as

$$\mathcal{L}_{\text{CE-SS},t}((\boldsymbol{x},y),\boldsymbol{\theta}) = \mathcal{L}_{\text{CE},t-1}((\boldsymbol{x},y),\boldsymbol{\theta}) \cdot \mathbb{1}\{y \in \mathcal{P}_t\} + \mathcal{D}_{KL}(\boldsymbol{y}_t \| \boldsymbol{p}_t(\boldsymbol{x},\boldsymbol{\theta})) \cdot \mathbb{1}\{y \in \mathcal{N}_t\}, \tag{7}$$

in which $\boldsymbol{y}_t$ stands for the one-hot vector in $\mathbb{R}^{|\mathcal{N}_t|}$ and $\boldsymbol{p}_t(\boldsymbol{x},\boldsymbol{\theta})$ is $\boldsymbol{p}^{\tau}_t(\boldsymbol{x},\boldsymbol{\theta})$ with $\tau = 1$. Namely, depending on whether $(\boldsymbol{x},y) \in \mathcal{M}$ or $(\boldsymbol{x},y) \in \mathcal{D}_t$, the softmax probability is computed separately by only using the output scores for $\mathcal{P}_t$ or $\mathcal{N}_t$, respectively, and the cross-entropy loss is

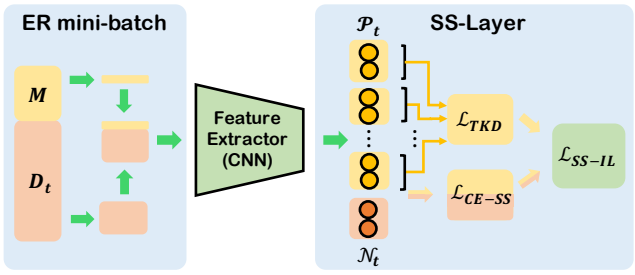

Figure 4. Illustration of SS-IL. The yellow regions represent the old classes, and red regions represent the new classes.

computed separately as well. While (7) is a simple modification of the ordinary cross-entropy (4), we can now observe that $\frac{\partial \mathcal{L}_{\text{CE-SS}}}{\partial z_{tc}} = 0$ for $c \in \mathcal{P}_t$ when $(\boldsymbol{x}, y) \in \mathcal{D}_t$. Therefore, the gradient from the new class samples in $\mathcal{N}_t$ will not have overly penalizing effect in the classification scores for the old classes in $\mathcal{P}_t$.

(2) Task-wise KD: In order to prevent preserving the bias present in GKD, we revisit the Task-wise distillation (TKD), used in LwF [25]. With the similar intuition as SS layer, we can easily see that it is natural to use TKD (3), which also uses the Separated-Softmax for each task, for the knowledge distillation. That is, in TKD, since the soft targets, $\{\boldsymbol{p}_s^{\tau}(\boldsymbol{x}, \boldsymbol{\theta}))\}_{s=1}^{t-1}$, are computed only within each task, TKD will not get affected by the task-wise bias that may remain in the old model $\boldsymbol{\theta}_{t-1}$, as opposed to the GKD shown in Section 4.2. Hence, we can expect that TKD is particularly well-suited for the SS layer, which will be shown in our experimental results.

Remark on the implementation detail: When random mini-batches from $\mathcal{D}_t \cup \mathcal{M}$ are naively used for the SGD updates of the model, it can also worsen the class ratio in a mini-batch towards the new classes. Such imbalance in mini-batches is expected to downplay the updates of the model for the old classes in our SS layer, since the gradient from the first part of (7) will be generated scarcely. Therefore, we additionally implement Experience Replay (ER) [11] technique that preserves the ratio of classes from old and new in a mini-batch to assure the minimum ratio of samples from $\mathcal{M}$. We empirically found that using ER gives more balanced prediction for SS-IL. Detailed analyses on using ER are in the Supplementary Materials.

Final loss function for SS-IL: By combining $\mathcal{L}_{\text{CE-SS},t}$ in (7) and $\mathcal{L}_{\text{TKD},t}$ in (3), the overall loss for SS-IL becomes:

$$\mathcal{L}_{\text{SS-IL},t}((\boldsymbol{x}, y), \boldsymbol{\theta}) = \mathcal{L}_{\text{CE-SS},t}((\boldsymbol{x}, y), \boldsymbol{\theta}) + \mathcal{L}_{\text{TKD},t}(\boldsymbol{x}, \boldsymbol{\theta}),$$

and the mini-batch SGD to minimize the loss is done with ER. Figure 4 illustrates our method, and the training algorithm is summarized in Supplementary Materials. We show in our experimental results that SS efficiently balances score between old and new classes, which, as a result, corrects the prediction bias. Finally, detailed results show that our SS-IL achieves the state-of-the-art accuracy for the various large scale benchmark datasets and in many different incremental scenarios.

6. Experiments

We believe the following two points are essential in evaluating a CIL model:

- Evaluation on the large-scale benchmark datasets.
- Evaluation on diverse incremental scenarios, *e.g.*, number of incremental tasks or size of the memory.

These points are related to the fundamentals of incremental learning considering real-world applications, which typically deal with large-scale data streams (both in data points and the number of classes) and various memory constraints. However, BiC [31] points out that many previously proposed CIL methods fail to scale up to large-scale datasets. Also, results in [4] and [6] show that CIL models are sensitive to incremental conditions such as the number of incremental tasks (T) and exemplar-memory size ($|\mathcal{M}|$). Therefore, we extensively compare our SS-IL with other state-of-the-art methods on two large-scale datasets (ImageNet ILSVRC 2012 [12] and Google Landmark Dataset v2 [1]) with various experimental scenarios.

For a fair comparison, we reproduced all the baselines covered in Table 1 and compared them in 15 different CIL scenarios. Evaluation on other CIL scenario proposed in [16, 13] that pre-trains the model on large base classes and uses fixed memory size per class was also covered in Supplementary Materials. Extensive analyses of our main contribution, the Separated-Softmax (SS) layer, are carried out to show the gradient blocking effect on balancing scores between old and new classes. Lastly, in detailed analyses about the distillation methods, we show the superiority of $\mathcal{L}_{\text{TKD}}$ over $\mathcal{L}_{\text{GKD}}$.

6.1. Datasets and evaluation protocol

ImageNet and Landmark-v2: As mentioned above, we used two large-scale benchmark datasets for our experiments, ImageNet and Google Landmark Dataset v2. ImageNet dataset consists of 1,000 classes, which has nearly 1,300 images per class. Google Landmark Dataset v2 consists of 203,094 classes, and each class has $1 \sim 10,247$ images. We sample 1,000 and 10,000 classes in the order of the largest number of samples per class to make two variations, and we denote each dataset as Landmark-v2-1K and Landmark-v2-10K, respectively. The number of images per class for Landmark ranges from 300 to 500 images, which inevitably provides model with imbalanced number of training data per class. By following the benchmark protocol in [28], we arrange the classes of each dataset in a fixed random order. We particularly stress that this is the first thorough CIL experiment on the large-scale Landmark-v2

Table 1. The results on various datasets and evaluation scenarios. The evaluation metrics are Average Top-1 and Top-5 accuracy. Accuracy is averaged over all the incremental tasks (*i.e.* including both initial task and incremental tasks)

T	$T = 10$			$\|\mathcal{M}\| = 10k$ (1K), $40k$ (10K)		
Dataset	ImageNet-1K	Land/mark-v2-1K	Landmark-v2-10K	ImageNet-1K	Landmark-v2-1K	Landmark-v2-10K
$\|\mathcal{M}\|$	$5k$ / $10k$ / $20k$	$5k$ / $10k$ / $20k$	$20k$ / $40k$ / $60k$	$T = 20$ / $T = 5$	$T = 20$ / $T = 5$	$T = 20$ / $T = 5$
Average Top-1 accuracy						
iCaRL [28]	47.0 / 50.5 / 53.1	37.4 / 41.1 / 44.0	23.1 / 27.2 / 29.2	44.8 / 56.2	38.1 / 45.1	23.8 / 31.2
FT [4]	38.1 / 45.8 / 53.5	41.9 / 48.9 / 55.8	33.6 / 40.3 / 44.5	44.5 / 46.9	45.0 / 53.6	38.3 / 43.1
IL2M [4]	41.9 / 48.4 / 55.3	42.4 / 49.2 / 55.9	34.2 / 40.7 / 44.8	45.6 / 52.4	45.2 / 54.2	38.4 / 44.0
EEIL [8]	57.8 / 59.4 / 60.9	52.1 / 55.5 / 58.2	43.5 / 46.1 / 48.0	53.5 / 63.8	50.5 / 59.1	41.5 / 49.8
BiC [31]	51.3 / 56.4 / 60.5	49.9 / 54.5 / 58.4	38.7 / 43.7 / 46.5	48.5 / 61.5	45.8 / 61.1	36.3 / 50.8
LUCIR [16]	51.0 / 53.6 / 56.5	50.5 / 53.7 / 57.3	46.2 / 49.1 / 50.9	46.8 / 61.3	48.5 / 61.0	**44.2** / 53.9
PODNet [13]	52.2 / 57.5 / 60.4	-	-	48.8 / 65.5	-	-
SS-IL (ours)	**63.5 / 64.5 / 65.2**	**57.7 / 59.0 / 59.9**	**50.1 / 51.4 / 51.9**	**58.8 / 68.2**	**51.4 / 64.3**	43.0 / **55.8**
Average Top-5 accuracy						
iCaRL [28]	71.0 / 75.1 / 77.4	56.9 / 62.0 / 65.2	35.6 / 41.9 / 44.8	69.7 / 79.7	58.6 / 65.7	37.8 / 46.8
FT [4]	66.7 / 73.3 / 78.8	62.1 / 68.5 / 74.0	49.4 / 56.7 / 60.6	71.3 / 73.0	64.6 / 72.5	54.6 / 58.9
IL2M [4]	70.6 / 75.3 / 79.7	62.4 / 68.5 / 73.9	49.7 / 56.7 / 60.6	71.8 / 78.7	64.4 / 73.1	54.3 / 59.8
EEIL [8]	81.2 / 82.0 / 83.0	72.6 / 74.9 / 76.7	60.4 / 62.6 / 64.1	77.0 / 85.3	70.3 / 77.7	57.8 / 66.0
BiC [31]	74.4 / 78.9 / 81.8	69.2 / 73.1 / 76.1	55.5 / 60.7 / 63.3	69.4 / 84.2	63.8 / 79.3	52.4 / 67.6
LUCIR [16]	72.4 / 75.6 / 78.7	68.7 / 72.2 / 75.3	62.2 / 65.2 / 66.7	69.2 / 82.7	67.7 / 78.0	60.7 / 69.3
PODNet [13]	73.6 / 79.4 / 82.1	-	-	71.1 / 85.8	-	-
SS-IL (ours)	**86.0 / 86.4 / 86.7**	**78.1 / 78.8 / 79.3**	**67.8 / 68.6 / 69.1**	**82.9 / 88.4**	**73.3 / 81.8**	**61.8 / 72.4**

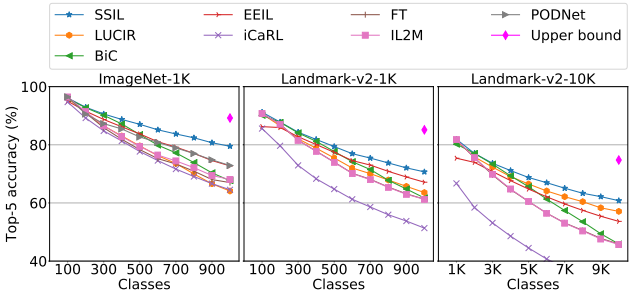

Figure 5. Top-5 accuracy results on ImageNet-1K, Landmark-v2-1K, and Landmark-v2-10K datasets for $T = 10$. The exemplar size is $|\mathcal{M}| = 20k$ in ImageNet-1K and Landmark-v2-1K datasets, and $|\mathcal{M}| = 60k$ in Landmark-v2-10K dataset.

dataset, with contains $10\times$ more classes (in Landmark-v2-10K) than previously reported results .

Evaluation protocol: To construct various training scenarios, we vary the total number of incremental tasks as $T = \{5, 10, 20\}$ which corresponds to $m = \{200, 100, 50\}$ in 1K datasets (ImageNet, Landmark-v2-1K) and $m = \{2000, 1000, 500\}$ in 10K dataset (Landmark-v2-10K), respectively. For the exemplar-memory size, we use $|\mathcal{M}| = \{5k, 10k, 20k\}$ for 1K datasets and $|\mathcal{M}| = \{20k, 40k, 60k\}$ for 10K dataset. We use the random selection used in [6] for constructing exemplar-memory. For the evaluation of CIL models, we use ImageNet validation set for ImageNet-1K, and we randomly select 50 and 10 images per class in Landmark-v2-1K and Landmark-v2-10K that are not in the training set, respectively. Additional explanations on dataset, evaluation protocol, and implementation detail are given in the Supplementary Materials.

6.2. Results

Table 1 shows the results on Average Top-1 and Top-5 accuracy. We compare our SS-IL with iCaRL [28], vanilla Fine-Tuning (FT) proposed in [4], IL2M [4], EEIL [8], BiC [31], LUCIR [16], and PODNet [13]. For each method, we used the hyperparameters reported in the original paper and run the experiments on all datasets. The left half of the table reports the results of fixed $T = 10$ with various exemplar-memory size $|\mathcal{M}|$, and the right half shows the results of fixed $|\mathcal{M}|$ with varying T. We could not run PODNet on Landmark-v2 datasets due to time and memory constraints.

In Table 1, we observe that there is no clear winner among the baselines; EEIL tends to excel for ImageNet-1K and Landmark-v2-1K, but for Landmark-v2-10K, LUCIR achieves the highest accuracy among baselines. Furthermore, the accuracy of all the baselines drastically drops when $|\mathcal{M}|$ gets smaller. The results presented here are consistent to the conclusion of [6] about the sensitiveness of recent CIL models to incremental scenarios and memory constraints. However, we observe that SS-IL dominates other baselines throughout almost every scenario. This indicates that SS-IL can be robustly applied to various conditions while other baselines are vulnerable to specific situation. Particularly, a notable characteristics of SS-IL is that the accuracy degradation of our model is minimal even when $|\mathcal{M}|$ becomes small, indicating the robustness with respect to the exemplar-memory size.

Figure 5 shows the overall Top-5 accuracy on each dataset with respect to the incremental task, when $|\mathcal{M}| = 20k$ and $T = 10$, and the tasks are denoted as classes. In this figure, we denote jointly trained approach as the Upper

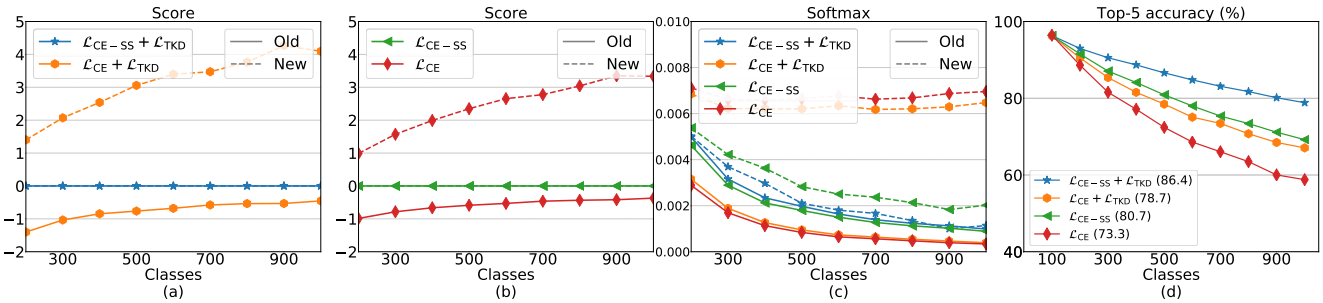

Figure 6. (a)/(b): Average classification scores for old & new classes. (c) Average softmax values for old & new classes. (d) Top-5 accuracy.

bound. Note that SS-IL again mostly dominates the baselines, and the performance gap over the baselines widens as the incremental task increases. Especially, in ImageNet-1K, compared with other baselines which have more performance degradation from the Upper bound, our SS-IL is less affected by catastrophic forgetting. Furthermore, we observe that iCaRL and EEIL achieve lower accuracy in the first incremental task. The Weak Nearest Exemplar Mean (NEM) classifier of iCaRL and the inefficient training schedule of EEIL could be the main reasons for such low accuracies.

6.3. Ablation study

In this section, we perform various detailed analyses on our SS layer to show its effect on balancing the prediction scores. To analyze its own strengths, we also carry out experiments that ablaties $\mathcal{L}_{\text{TKD}}$ in $\mathcal{L}_{\text{SS-IL}}$.

Figure 6 and 7 show the results of our analysis. In these figures, "$\mathcal{L}_{\text{CE}}$" stands for the model that does not have both TKD and SS layer, "$\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{TKD}}$" stands for the model that does not have SS layer, "$\mathcal{L}_{\text{CE-SS}}$", stands for the model that only has SS, and "$\mathcal{L}_{\text{CE-SS}} + \mathcal{L}_{\text{TKD}}$" stands for our SS-IL. We compare the four models on ImageNet-1K with $T = 10$ and $|\mathcal{M}| = 10k$ and analyze (a) the output scores and softmax probability values, (b) the prediction results with confusion matrix, and (c) the Top-5 accuracy.

The output score and softmax probability Figure 6 (a) and (b) show the average classification scores for the old and new classes, and Figure 6 (c) shows the average softmax probabilities, for the old and new classes, on test samples for each incremental task t. When computing the average scores and softmax probabilities, we first averaged the values over the old and new classes, respectively, then we averaged them over all the test samples. For the softmax probability, we first normalized the scores using all the seen classes, which include the new classes.

In Figure 6 (a) and (b), we can observe that for "$\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{TKD}}$" and "$\mathcal{L}_{\text{CE}}$", the score difference between the old and new classes is significant, confirming the existence of the score bias toward the new classes. Moreover, the gap widens as the incremental task increases. For "$\mathcal{L}_{\text{CE-SS}} + \mathcal{L}_{\text{TKD}}$" and "$\mathcal{L}_{\text{CE-SS}}$", however, the plots for old and new classes in Figure 6 (a) and (b) mostly overlap with each other, throughout the entire incremental learning stages. This result suggests that our SS successfully mitigates the classification score bias, without any necessary post-processing, and such balanced classification scores eventually lead to more balanced and accurate predictions.

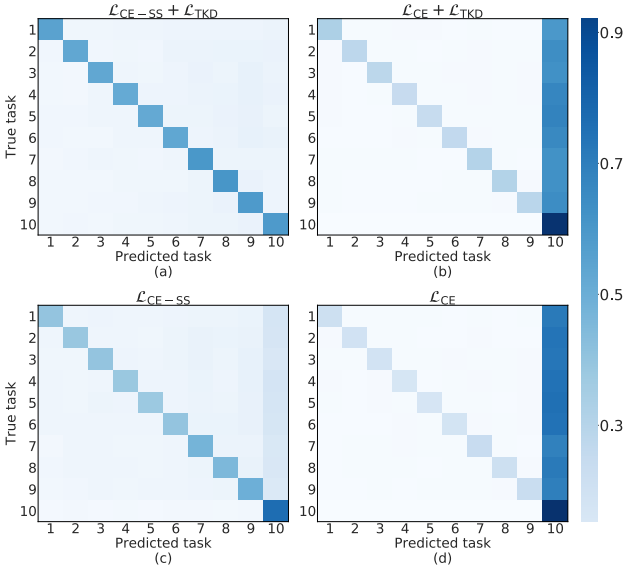

Figure 7. Confusion matrices based on the predictions of various models for test data. We denote the tasks to which the predicted class belongs.

Similarly, we also analyzed the average of the softmax probabilities normalized over all the seen classes in Figure 6 (c). Note that since computing softmax only considers the relative difference between scores, we can compare four models without considering the magnitude of the score. From the figure, we observe that the softmax values for both old and new classes for "$\mathcal{L}_{\text{CE-SS}} + \mathcal{L}_{\text{TKD}}$" and "$\mathcal{L}_{\text{CE-SS}}$" are almost the same, which is consistent with the results on the output score.

Confusion matrix Figure 7 shows the confusion matrices (across the tasks) of the class predictions for the four models. In Figure 7, we can observe that the predictions of "$\mathcal{L}_{\text{CE-SS}} + \mathcal{L}_{\text{TKD}}$" and "$\mathcal{L}_{\text{CE-SS}}$" are much more balanced compared to those of "$\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{TKD}}$" and "$\mathcal{L}_{\text{CE}}$", which have

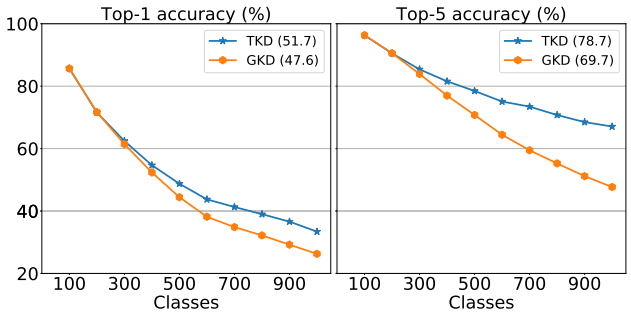

Figure 8. Top-1 accuracy (left) and Top-5 accuracy (right) for $\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{GKD}}$ and $\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{TKD}}$

highly biased predictions. We believe above results show that TKD alone is not enough for resolving the prediction bias, and SS is essential for achieving the balanced predictions across the incremental tasks.

Top-5 accuracy Figure 6(d) shows the Top-5 accuracy of the four models. In this figure, as we expected, "$\mathcal{L}_{\text{CE-SS}} + \mathcal{L}_{\text{TKD}}$" achieves the highest accuracy, and the models equipped with $\mathcal{L}_{\text{TKD}}$ outperform the models trained without it. Furthermore, we observe that "$\mathcal{L}_{\text{CE-SS}}$" outperforms "$\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{TKD}}$" and "$\mathcal{L}_{\text{CE}}$", which again confirms that using SS is essential in achieving high accuracy.

6.4. Analyses on KD

In this section, we carry out several experiments to highlight the difference of TKD and GKD. Firstly, we evaluate the performance of models trained with TKD and GKD, respectively. Secondly, for direct comparison, we also check the accuracy of the models when each KD loss is used for the distillation from the same biased soft target. That way, we can focus only on the effect of TKD and GKD. To clarify only the effect of KD, no bias correction scheme is used for training and the same training settings are used. Training details are explained in Supplementary Materials.

Comparison of $\mathcal{L}_{\text{TKD}}$ and $\mathcal{L}_{\text{GKD}}$ Figure 8 shows the Top-1 and Top-5 accuracy with respect to the varying KD loss. Here, model GKD denotes the model trained with loss (5) and model TKD stands for the one that replaces $\mathcal{L}_{\text{GKD}}$ in (5) with $\mathcal{L}_{\text{TKD}}$. They are trained on ImageNet-1K, $|\mathcal{M}| = 10k$, and $T = 10$. As shown in Figure 8, TKD achieves much higher accuracy than GKD, while the accuracy of GKD drastically declines as the task proceeds. As shown in Figure 2, model that does not use any bias correction method has an extreme bias and GKD may preserve such bias. On the other hand, Figure 8 shows that TKD is much less affected by the bias. [23] argues that using TKD may miss the knowledge about discrimination between the old tasks, hence, GKD should be preferred. However, as we observe from the figure, TKD is a clear winner in terms of accuracy, since the harm of preserving the prediction bias by GKD turns out to be much greater.

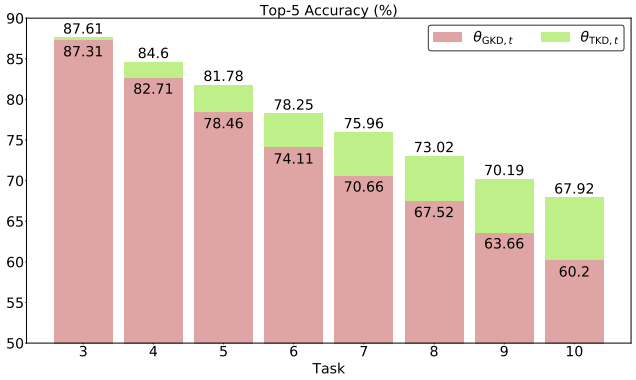

Figure 9. Top-5 accuracy for $\boldsymbol{\theta}_{\text{GKD},t}$ and $\boldsymbol{\theta}_{\text{TKD},t}$ which use $\mathcal{L}_{\text{GKD}}$ and $\mathcal{L}_{\text{TKD}}$ on same teacher model

Comparison of $\mathcal{L}_{\text{TKD}}$ and $\mathcal{L}_{\text{GKD}}$ on the same biased teacher In order to directly compare the behavior of $\mathcal{L}_{\text{TKD}}$ and $\mathcal{L}_{\text{GKD}}$ on the same biased soft target for distillation, we carry out another experiment with the following scenario:

1. Train with $\mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{GKD}}$ until task $t-1$ and obtain $\boldsymbol{\theta}_{t-1}$.
2. At task t, train $\boldsymbol{\theta}_{t-1}$ using two different KD losses, $\mathcal{L}_{\text{GKD}}$ and $\mathcal{L}_{\text{TKD}}$, and obtain two models, $\boldsymbol{\theta}_{\text{GKD},t}$ and $\boldsymbol{\theta}_{\text{TKD},t}$, respectively.

Figure 9 shows the Top-5 accuracy of $\boldsymbol{\theta}_{\text{GKD},t}$ and $\boldsymbol{\theta}_{\text{TKD},t}$ at tasks $t = 3, ..., 10$. We clearly observe that the accuracy of $\boldsymbol{\theta}_{\text{TKD},t}$ is always higher than that of $\boldsymbol{\theta}_{\text{GKD},t}$ at all tasks, which again shows that GKD preserving the score bias of $\boldsymbol{\theta}_{t-1}$ on $\boldsymbol{x} \in \mathcal{D}_t$ would be the main cause of such accuracy gap. In contrast, as in SS, TKD that utilizes the task specific separate softmax for distillation lessens the effects of the score bias. We believe this analysis is another evidence for justifying the TKD over GKD for CIL.

7. Concluding Remarks

We proposed SS-IL that addresses the classification score bias problem in the exemplar-memory based CIL. Our analysis suggests that ordinary softmax probability considering all the classes forces the heavy penalization of the output probabilities for the old classes, which is the main reason of the score bias. We also experimentally find that such bias is preserved by GKD and that TKD gets less affected by such bias. In our extensive experimental results, our SS-IL shows outstanding performance in most of the scenarios, and we verify that using SS can effectively balance the scores between old and new classes.

Acknowledgments

This work was supported in part by NRF Mid-Career Research Program [NRF-2021R1A2C2007884] and IITP grants [No.2019- 0-01396, Development of framework for analyzing, detecting, mitigating of bias in AI model and training data] [IITP-2021-2018-0-01798, ITRC Support Program], funded by the Korean government.

References

[1] Google landmarks dataset v2. 2019. 5

[2] Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. In *Advances in Neural Information Processing Systems*, pages 4394–4404, 2019. 2

[3] M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha. Deep learning algorithm for autonomous driving using googlenet. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 89–96, June 2017. 1

[4] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1, 2, 3, 5, 6

[5] Eden Belouadah and Adrian Popescu. Scail: Classifier weights scaling for class incremental learning. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1266–1275, 2020. 1, 2, 3

[6] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 2020. 5, 6

[7] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1721–1730, New York, NY, USA, 2015. ACM. 1

[8] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 233–248, 2018. 1, 2, 3, 6

[9] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018. 2

[10] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019. 2

[11] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*, 2019. 2, 5

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 2009. 5

[13] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, 2020. 2, 5, 6

[14] G. Hinton, Y. LeCun, and Y. Bengio. Deep learning. *Nature*, 521:436–444, 2015. 1

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 1

[16] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 2, 3, 5, 6

[17] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 2

[18] Khurram Javed and Faisal Shafait. Revisiting distillation and incremental classifier learning. In *Asian Conference on Computer Vision*, pages 3–17. Springer, 2018. 2

[19] Sangwon Jung, Hongjoon Ahn, Sungmin Cha, and Taesup Moon. Continual learning with node-importance based adaptive group sparse regularization. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3647–3658. Curran Associates, Inc., 2020. 2

[20] D. Kang, Y. Jo, Y. Nam, and J. Choi. Confidence calibration for incremental learning. *IEEE Access*, 8:126648–126660, 2020. 2

[21] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[22] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. 2

[23] Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 312–321, 2019. 2, 3, 8

[24] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2019. 2

[25] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017. 2, 3, 4, 5

[26] David Lopez-Paz and Marc Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing System (NIPS)*, pages 6467–6476. 2017. 2

[27] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with

neural networks: A review. *Neural Networks*, 113:54–71, 2019. 2

[28] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2001–2010, 2017. 1, 2, 5, 6

[29] 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. 2

[30] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing System (NIPS)*, pages 2990–2999. 2017. 2

[31] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 1, 2, 3, 5, 6

[32] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental learning using conditional adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6619–6628, 2019. 2

[33] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[34] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning (ICML)*, pages 3987–3995, 2017. 2

[35] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217, 2020. 1, 2, 3