Supplementary Materials for 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

This material specifies data configuration in CIL scenarios and model implementation details on SS-IL and other baselines. It also includes additional experimental results on ER mini-batch ablation study and other CIL scenarios. Additional CIL setting [8, 5] consists of large initial base classes and limited memory usage per class. Lastly, model performance with respect to the incremental task is reported to show the overall behavior.

For brevity, we use the term *large base* for scenarios that gives 50% of total classes as initial task and *base* for those which consider fixed number of classes across all tasks. Also, *memory per class* is used for exemplar-memory constraint allowing only constant number of samples per classes and *fixed memory* for alleviated memory usage to fully store samples from seen classes. For example, Table 1 (Manuscript) corresponds to *base* and *fixed memory* setting. As a result, we show comprehensive results of the models in 4 different settings and 15 different conditions within each setting, which results in 60 scenarios in total.

1. Datasets and evaluation protocol

ImageNet: ILSVRC 2012 dataset consists of 1,000 classes, which has nearly 1,300 images per class. By following the benchmark protocol in [11], we arrange the classes of each dataset in a fixed random order. For the evaluation of CIL models, we use ILSVRC 2012 validation set for testing.

In Table 1 (Manuscript), we experiment with varied total number of incremental tasks, $T = \{5, 10, 20\}$, which corresponds to $m = \{200, 100, 50\}$ per task, and for the exemplar-memory size, we use $|\mathcal{M}| = \{5k, 10k, 20k\}$. When constructing exemplar-memory, we use Random selection [2] for fixed memory setting, which simply samples random data from old classes. As exemplars from the new classes are randomly selected, it is required to delete exemplars from the old classes. In order to maintain balanced number of exemplars across all the old classes, classes that have more exemplars are selected and exemplars for the corresponding class become more likely to be removed. By doing so, difference of the number of samples across the classes is at most 1.

In growing memory setting, we use Ring buffer approach proposed in [4]. We stored a constant number of samples per old class, which we denote $|\mathcal{M}_{per}| = \{5, 10, 20\}$. Thus, the number of samples stored in the memory grows as new tasks sequentially arrive. In large base setting, we first train the model with 50% of total classes and incrementally learn additional classes per task which corresponds to $m = \{100, 50, 25\}$. Here, we also compare the experimental results of two exemplar-memory managing approach : growing memory and fixed memory.

Landmark-v2: Google Landmark Dataset v2 consists of 203,094 classes, and each class has $1 \sim 10,247$ images. Since the dataset is highly imbalanced, we sample 1,000 and 10,000 classes in the order of largest number of samples per class. We denote Landmark-v2 dataset with 1,000 and 10,000 classes as Landmark-v2-1K and Landmark-v2-10K, respectively. After sampling the classes, we arrange the classes in a fixed random order. For evaluation, we randomly select 50 and 10 images per each class in Landmark-v2-1K and Landmark-v2-10K that are not in the training set for testing.

Since Landmark-v2-1K consists of same number of classes with ImageNet, all the figures regarding memory size ($|\mathcal{M}|$ or $|\mathcal{M}_{per}|$) and task numbers(T) are similar with ImageNet. However, in Landmark-v2-10K which is composed of $10,000$ classes, the number of classes in each task is changed to $\{2000, 1000, 500\}$ when we set $T = \{5, 10, 20\}$ in base setting and to $\{1000, 500, 250\}$ in large base setting. For exemplar-memory size, we use $|\mathcal{M}| = \{20k, 40k, 60k\}$ in fixed memory and $|\mathcal{M}_{per}| = \{2, 4, 6\}$. in growing memory setting.

*Equal contribution.
†Corresponding author.

Table 1. Hyper-parameters for all methods. Details of post-processing implementations on each method are in Section 2.1

	Training			Post-processing					
Methods / Hyper-parameters	Epochs	LR	LR schedule	Epochs	LR	LR schedule	LR decay rate	Batch size	Other hyper-parameters
iCaRL	60	0.1	20 30 40 50	-	-	-	0.2	128	-
FT	100	0.1	40 80	-	-	-	0.1	128	-
IL2M	100	0.1	40 80	-	-	-	0.1	128	-
EEIL	40	0.1	10 20 30	30	0.01	10 20	0.1	128	-
BiC	100	0.1	30 60 90	200	0.001	60 120 180	0.1	256	-
LUCIR	90	0.1	30 60	-	-	-	0.1	128	$m = 0.5, K = 2, \lambda_{base} = 10$
PODNet	90	0.05	-	20	0.01	-	0.1	64	$\lambda_c = 8, \lambda_f = 10$
SS-IL	100	0.1	40 80	-	-	-	0.1	128	See details

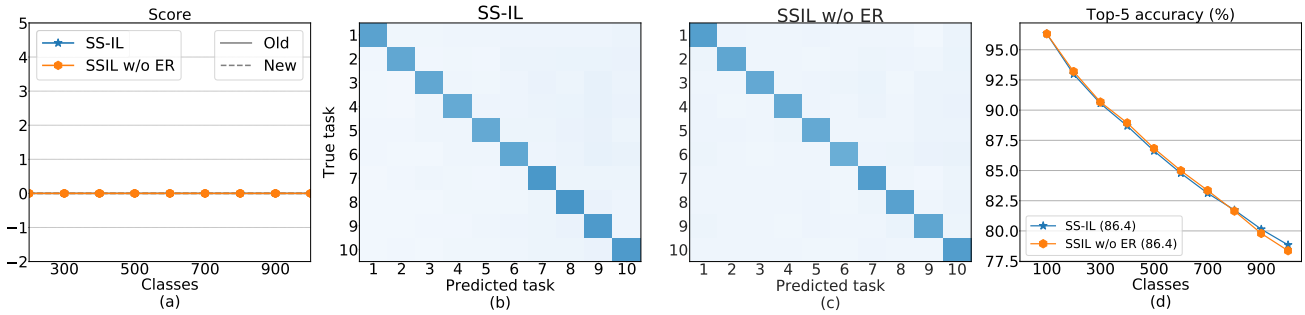


Figure 1. (a) Average classification score for old and new classes. (b) and (c) confusion matrix, (d) Top-5 accuracy.

2. Implementation details

2.1. Baselines and SS-IL

All the baselines use the Resnet-18 [7] architecture and are implemented using PyTorch framework [10]. The weight decay is set to 0.0001 and stochastic gradient descent (SGD) is used with momentum 0.9. Softmax scaling parameter(τ) used for distillation in Eq.(2) and Eq.(3) (Manuscript) is set to 2. Details of the hyper-parameters are summarized in Table 1 and additional explanations on each baseline model are written in this section.

We planned to consider WA [13] as one of our baselines for comparison. However, it was unable to compare our method with WA, since it did not publish its official code and reproducing its algorithm was unfeasible. Including SS-IL and all the other baselines, the code implementations will be publicly available.

iCaRL [11]: Considering the implementations proposed in [9], instead of binary cross entropy, multi-class cross entropy loss is used for both classification loss and KD loss.

FT and IL2M [1]: After initial task, the training epochs and the learning rate decay schedule are divided by 4, *i.e.* 25 epochs and learning rate decay at 10 and 20 epoch.

EEIL [3]: During post-processing, balanced fine-tuning is applied to both the feature extractor and the classifier. Same with iCarL, muti-class cross entropy is used for both classification loss and KD loss.

BiC [12]: After training, BiC additionally trains bias correction layer using auxiliary validation set. We used 9:1 split for train:val split, which is reported to be the best choice in the original paper.

LUCIR [8]: LUCIR indicates the "CNN" based method which shows better performance in large scale dataset compared with "NEM" based method. Also, additional balanced fine-tuning for post-processing was not implemented, since its effect is insignificant according to [8].

PODNet [6]: For the classifier, 10 proxies are used, and for faster convergence, NCA loss is used by setting margin and scale to 0.6 and 1. For post-processing, balanced fine-tuning is performed on the output classifier.

SS-IL (ours) The batch size used for $\mathcal{D}_t$, *i.e.* $N_{\mathcal{D}_t}$, is 128, and we use different replay batch size, $N_{\mathcal{M}}$, depending on the number of different incremental tasks; *i.e.*, $N_{\mathcal{M}} = 16/32/64$ for $T = 20/10/5$, respectively. Thus, the ratio of $N_{\mathcal{D}_t}$ over $N_{\mathcal{M}}$ is $8/4/2$, respectively.

2.2. Details on KD analysis models

For a fair comparison, two models used in Section 6.4 (Manuscript) are implemented on the same experimental conditions and differ in the KD loss term ($\mathcal{L}_{\text{TKD}}$ and $\mathcal{L}_{\text{GKD}}$). We use the Resnet-18 [7] architecture and the stochastic gradient descent (SGD) with momentum 0.9. The number of epochs for training incremental task is 100. The learning rate starts at 0.1 and is divided by 10 at 40 and 80 epochs. The weight decay is set to 0.0001 and the batch size is 128. Softmax scaling parameter(τ) used for distillation is set to 2. It was trained on base and fixed memory setting which chooses Random selection for exemplar-memory construction.

3. Algorithm

Algorithm 1 shows the overall training mechanism of our SS-IL.

Algorithm 1 Separated Softmax for Incremental Learning (SS-IL)

Require: $\{\mathcal{D}_t\}_{t=1}^{T}$: Training dataset
Require: $\mathcal{M} \leftarrow \{\}$: Memory buffer
Require: E: The number of epochs per task.
Require: $N_{\mathcal{D}_t}, N_{\mathcal{M}}$: Training & replay batch sizes
Require: α: Learning rate
Require: $\boldsymbol{\theta}$: Network parameters
  # Start class incremental learning
  Randomly initialize $\boldsymbol{\theta}$
  for $t = 1, ..., T$ **do**
    for $e = 1, ..., E$ **do**
      # Sample a mini-batch of size $N_{\mathcal{D}_t}$
      for $B_{\mathcal{D}_t} \sim \mathcal{D}_t$ **do**
        # Sample a mini-batch of size $N_{\mathcal{M}}$
        $B_{\mathcal{M}} \sim \mathcal{M}$
        $\mathcal{L}_t(\boldsymbol{\theta}) = \sum_{(\boldsymbol{x},\boldsymbol{y}) \in B_{\mathcal{D}_t} \cup B_{\mathcal{M}}} \mathcal{L}_{\text{SS-IL},t}((\boldsymbol{x},\boldsymbol{y}),\boldsymbol{\theta})$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \frac{\alpha}{N_{\mathcal{D}_t}+N_{\mathcal{M}}} \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}_t(\boldsymbol{\theta})$
      end for
    end for
    $\mathcal{M} \leftarrow$ UpdateMemory($\mathcal{D}_t, \mathcal{M}$)
  end for

4. Analysis on ER mini-batch

Table 2. Results on ImageNet-1K with varying $N_{\mathcal{M}}$ and T.

T/$N_{\mathcal{M}}$	16 / 32 / 64	16 / 32 / 64
	Average Top-1 accuracy	Average Top-5 accuracy
20	58.8 / 59.0 / 58.9	82.9 / 82.6 / 82.4
10	54.3 / 64.5 / 68.2	86.6 / 86.4 / 86.0
5	68.4 / 68.4 / 68.2	88.8 / 88.6 / 88.4

In this section, we carry out analyses on ER mini-batch for ImageNet-1K with $T = 10$ and $|\mathcal{M}| = 10k$. Figure 1 shows the ablation study results on ER mini-batch. Note that "SS-IL w/o ER" stands for SS-IL without ER mini-batch. In Figure 1 (a), similarly as the results shown in Manuscript, due to the effectiveness of SS, "SS-IL w/o ER" also has balanced output scores. By comparing Figure 1 (b) and (c), "SS-IL" shows little more balanced predictions, and as a result, "SS-IL" shows minute increase in the final task. Though the effect of using ER mini-batch is marginal, to get more balanced prediction and well performing results, we use it as an additional technique.

Table 2 shows the results on Average Top-1 and Top-5 accuracy with respect to varying ER mini-batch size, $N_{\mathcal{M}}$, and the total number of incremental tasks, T. From the table, we observe that no matter what $N_{\mathcal{M}}$ is being used, the accuracy differences are negligible. This indicates that using ER mini-batch is effective regardless of the ratio between old and new class samples in the mini-batch, if the old class examples are guaranteed to some extent.

5. Additional results

5.1. Additional CIL scenarios

Table 3, 4, and 5 report Average Top-1 and Top-5 accuracy in ImageNet-1K and Landmark-v2-1K. Each table represents different CIL setting depending on additional memory constraint and base class quantity. Namely, CIL scenarios in Table 4 and Table 5 assume recently proposed large base setting. Also, Table 3 and Table 5 assume growing memory setting which has more strict memory constraint. Baseline models that show comparable results among those in Table 1(Manuscript) are selected for evaluation. Due to time and memory limitations, we are currently unable to report all the results for Landmark-v2-10K dataset, particularly for PODNet [6]. We will make sure to update the remaining results as soon as possible in an arXiv version.

In Table 3, we clearly observe that our SS-IL is superior to other methods for the hard memory constraint setting. Results in $T = 10$ and $|\mathcal{M}_{per}| = \{5, 10, 20\}$ show that SS-IL with $|\mathcal{M}_{per}| = 5$ even excel other models that use two or four times more images per class. Also, compared to results in Table 1 (Manuscript), SS-IL shows much small Top-1 accuracy drop caused by most of the cases, *e.g*, SS-IL(1.3%↓), EEIL(8.9%↓), BIC(5.5%↓), LUCIR(3.8%↓), PODNet(3.7%↓) at ImageNet-1K $T = 10$ and $|\mathcal{M}| = 20$K, which corresponds to ImageNet-1K $T = 10$ and $|\mathcal{M}_{per}| = 20$. These results show SS-IL's strong robustness in memory conditions which in turn leads to highest performance in most CIL scenarios.

In Table 4 and 5, we clearly observe that SS-IL outperforms strong baselines in large base setting (LUCIR, PODNet) as well on many scenarios. Note that unlike LUCIR and PODNet that utilize large base setting tailored algorithm, SS-IL does not assume any CIL scenario in learning objective and achieves as much or even better performance. SS-IL also shows no significant difference in accuracy when growing memory setting is adapted in Table 5.

5.2. Overall Top-1 and Top-5 accuracy

We report overall Top-1 and Top-5 accuracy on each dataset with respect to the incremental task. By referring these figures below, we can compare each methods in a more class incremental view. Figure 2 and 3 show the detailed results used to generate (Table 1 (Manuscript). Similarly, Figure 4, Figure 5, and Figure 6 show the overall results in Table 3, Table 4, and Table 5. In summary, SS-IL achieves much higher accuracy than other baselines for most of the scenarios.

Table 3. The results on *base* setting combined with *growing memory* setting. $|\mathcal{M}_{per|}$ denotes the number of stored samples per old class. The evaluation metrics are the Average Top-1 and Top-5 accuracy. Accuracy is averaged over all the incremental tasks (i.e. including both initial task and incremental tasks)

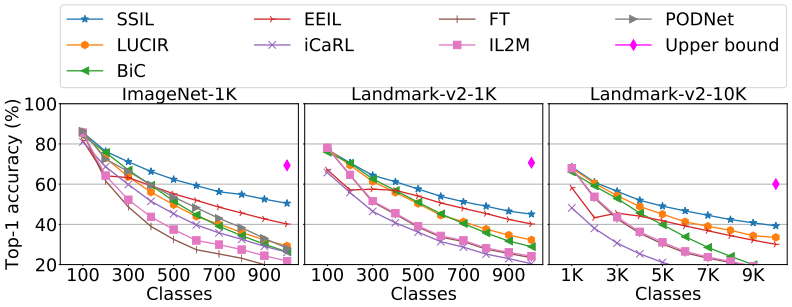
	$T = 10$		$\|\mathcal{M}_{per}\| = 10$ (1K), 4 (10K)	
Dataset	ImageNet-1K	Landmark-v2-1K	ImageNet-1K	Landmark-v2-1K
$\|\mathcal{M}_{per}\|$	5 / 10 / 20	5 / 10 / 20	$T = 20$ / $T = 5$	$T = 20$ / $T = 5$
Average Top-1 accuracy				
EEIL [3]	38.5 / 46.2 / 52.0	38.8 / 44.5 / 50.0	42.2 / 51.2	38.8 / 49.5
BiC [12]	38.5 / 43.0 / 55.0	39.1 / 47.8 / 53.8	38.5 / 59.1	36.4 / 58.1
LUCIR [8]	47.0 / 49.7 / 52.7	46.1 / 49.3 / 52.6	39.0 / 59.5	41.1 / 58.0
PODNet [6]	44.2 / 57.3 / 56.7	-	40.5 / 63.6	-
SS-IL (ours)	**62.3 / 63.4 / 63.9**	**56.0 / 57.0 / 58.1**	**57.0 / 67.4**	**49.9 / 62.7**
Average Top-5 accuracy				
EEIL [3]	65.3 / 72.3 / 76.9	58.5 / 64.3 / 69.5	68.1 / 76.0	58.6 / 69.0
BiC [12]	57.5 / 67.1 / 78.2	57.0 / 67.4 / 72.8	59.8 / 79.8	54.8 / 76.9
LUCIR [8]	67.8 / 71.3 / 74.8	63.7 / 70.9 / 67.5	59.7 / 81.0	59.2 /75.3
PODNet [6]	64.8 / 79.2 / 78.9	-	62.7 / 84.2	-
SS-IL (ours)	**85.3 / 85.9 / 86.0**	**76.8 / 77.3 / 77.7**	**81.6 / 88.2**	**72.2 / 80.9**

Table 4. The results on *large base* setting combined with *fixed memory* setting. $|\mathcal{M}|$ denotes the number of stored samples during training.

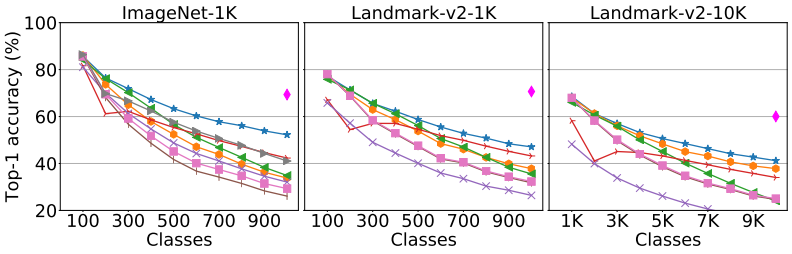
	$T = 10$		$\|\mathcal{M}\| = 10k$ (1K), $40k$ (10K)	
Dataset	ImageNet-1K	Landmark-v2-1K	ImageNet-1K	Landmark-v2-1K
$\|\mathcal{M}\|$	$5k$ / $10k$ / $20k$	$5k$ / $10k$ / $20k$	$T = 20$ / $T = 5$	$T = 20$ / $T = 5$
Average Top-1 accuracy				
EEIL [3]	40.6 / 46.2 / 50.7	41.0 / 46.6 / 51.6	39.3 / 52.7	41.3 / 51.8
BiC [12]	41.4 / 46.4 / 50.7	41.1 / 45.8 / 49.8	36.2 / 55.4	36.5 / 55.3
LUCIR [8]	54.7 / 57.6 / 60.6	54.9 / 58.3 / 61.7	54.8 / 60.3	**55.2** / 61.4
PODNet [6]	47.9 / 58.4 / **64.2**	-	51.0 / 65.3	-
SS-IL (ours)	**59.9 / 61.9** / 63.4	**57.5 / 59.8 / 61.9**	**57.1 / 65.7**	55.0/ **64.0**
Average Top-5 accuracy				
EEIL [3]	66.7 / 71.8 / 75.6	61.1 / 66.1 / 70.1	64.6 / 77.8	60.6 / 71.1
BiC [12]	64.5 / 69.8 / 74.2	59.5 / 64.4 / 68.4	58.0 / 78.9	53.6 / 74.0
LUCIR [8]	77.5 / 80.5 / 83.2	73.3 / 76.3 / 78.9	78.4 / 82.6	74.0 / 78.4
PODNet [6]	68.9 / 79.9 / 84.7	-	73.0 / 85.7	-
SS-IL (ours)	**84.7 / 85.4 / 86.2**	**78.3 / 79.4 / 80.6**	**82.3 / 87.6**	**76.0 / 82.2**

Table 5. The results on *large base* setting combined with *growing memory* setting.

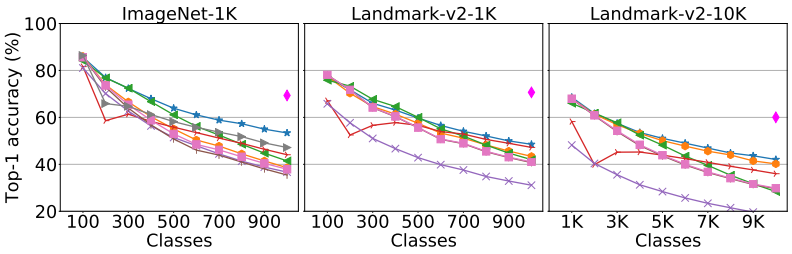
	$T = 10$		$\|\mathcal{M}_{per}\| = 10$ (1K), 4 (10K)	
Dataset	ImageNet-1K	Landmark-v2-1K	ImageNet-1K	Landmark-v2-1K
$\|\mathcal{M}_{per}\|$	5 / 10 / 20	5 / 10 / 20	$T = 20$ / $T = 5$	$T = 20$ / $T = 5$
Average Top-1 accuracy				
EEIL [3]	31.6 / 38.1 / 44.0	33.8 / 40.1 / 46.4	32.8 / 43.1	36.6 / 44.7
BiC [12]	38.0 / 44.4 / -	39.4 / 44.6 / 49.0	33.9 / 53.6	35.6 / 54.1
LUCIR [8]	53.3 / 55.9 / 59.0	52.5 / 56.5 / 59.8	52.6 / 59.2	**52.9** / 59.8
PODNet [6]	41.9 / 53.5 / 61.9	-	45.7 / 63.3	-
SS-IL (ours)	**58.3 / 60.5 / 62.3**	**55.7 / 58.0 / 60.2**	**55.4 / 64.9**	**52.9 / 62.8**
Average Top-5 accuracy				
EEIL [3]	56.9 / 64.1 / 69.6	52.6 / 59.1 / 65.0	57.3 / 69.0	55.2 / 63.8
BiC [12]	61.2 / 68.0 / -	57.6 / 63.2 / 67.4	55.4 / 77.6	52.6 / 73.0
LUCIR [8]	75.7 / 78.8 / 81.8	70.1 / 75.0 / 77.6	76.4 / 81.5	72.0 / 77.3
PODNet [6]	62.0 / 75.3 / 83.0	-	67.6 / 84.4	-
SS-IL (ours)	**84.1 / 84.8 / 85.7**	**77.4 / 78.6 / 79.7**	**81.2 / 87.5**	**74.5 / 81.8**



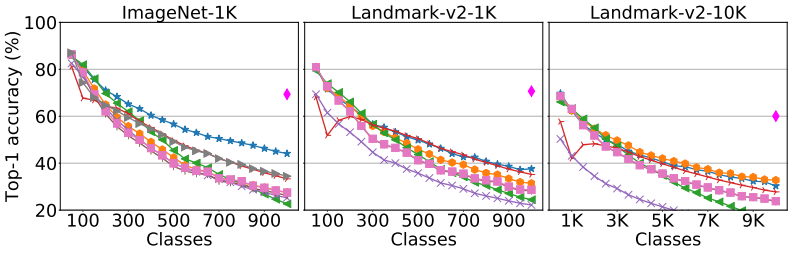
(a) $T = 10$ and $|\mathcal{M}| = 5k(1\text{K}), 20k(10\text{K})$



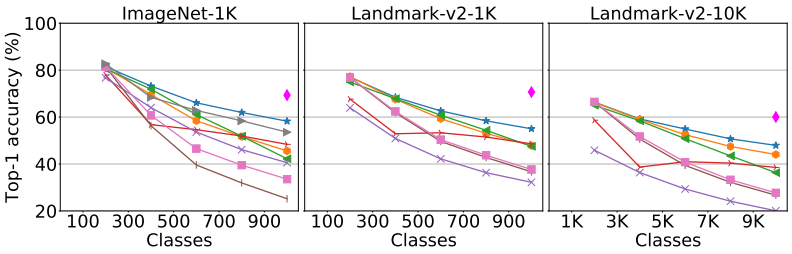
(b) $T = 10$ and $|\mathcal{M}| = 10k(1\text{K}), 40k(10\text{K})$



(c) $T = 10$ and $|\mathcal{M}| = 20k(1\text{K}), 60k(10\text{K})$

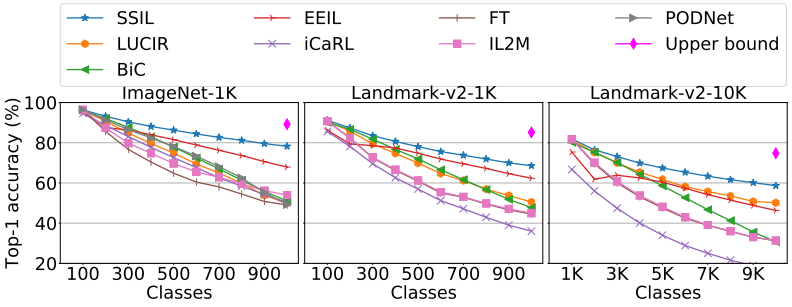


(d) $T = 20$ and $|\mathcal{M}| = 10k(1\text{K}), 40k(10\text{K})$

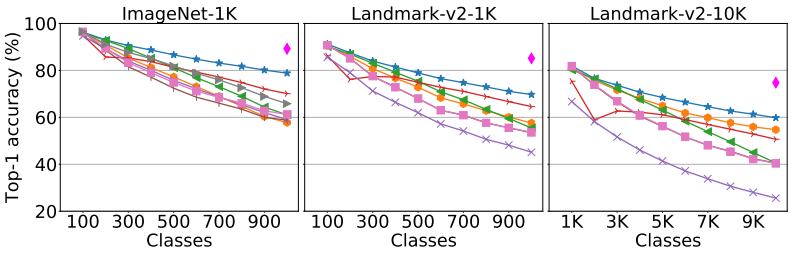


(e) $T = 5$ and $|\mathcal{M}| = 10k(1\text{K}), 40k(10\text{K})$

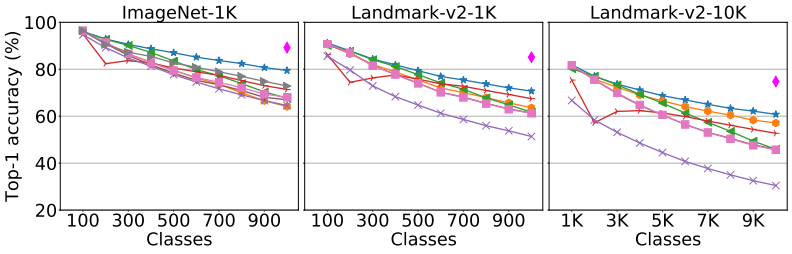
Figure 2. Top-1 accuracy on ImageNet-1K, Landmark-1K and, Landmark-10K.



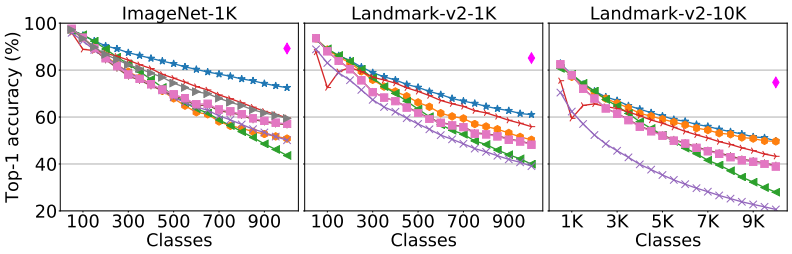
(a) $T = 10$ and $|\mathcal{M}| = 5k$(1K), $20k$(10K)



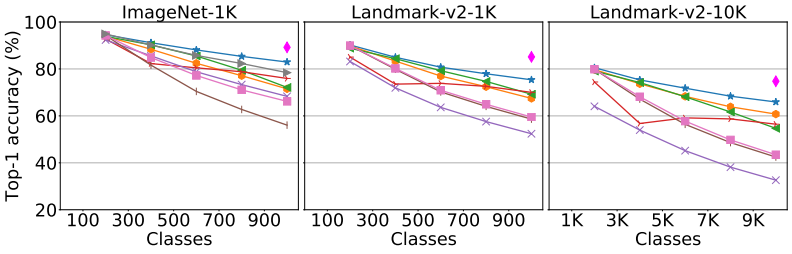
(b) $T = 10$ and $|\mathcal{M}| = 10k$(1K), $40k$(10K)



(c) $T = 10$ and $|\mathcal{M}| = 20k$(1K), $60k$(10K)

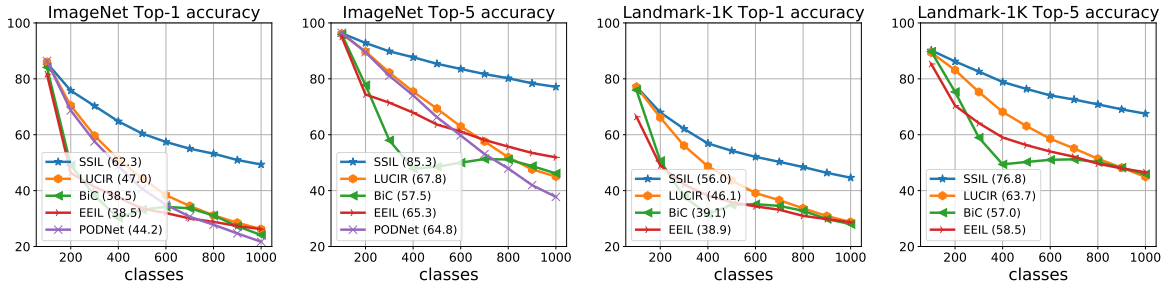


(d) $T = 20$ and $|\mathcal{M}| = 10k$(1K), $40k$(10K)

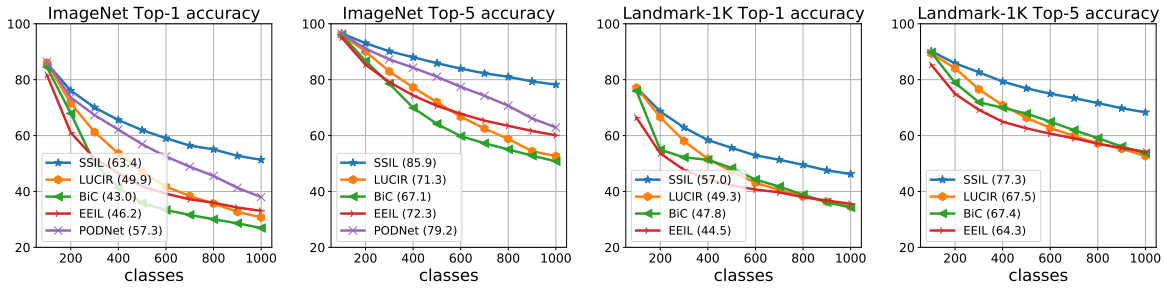


(e) $T = 5$ and $|\mathcal{M}| = 10k$(1K), $40k$(10K)

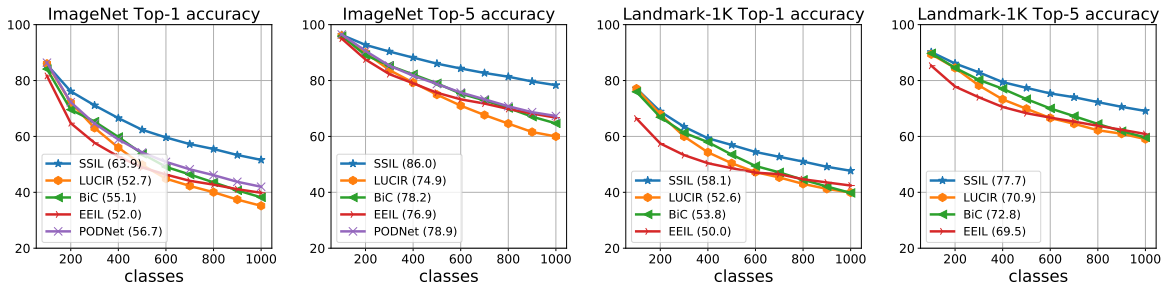
Figure 3. Top-5 accuracy on ImageNet-1K, Landmark-1K and, Landmark-10K.



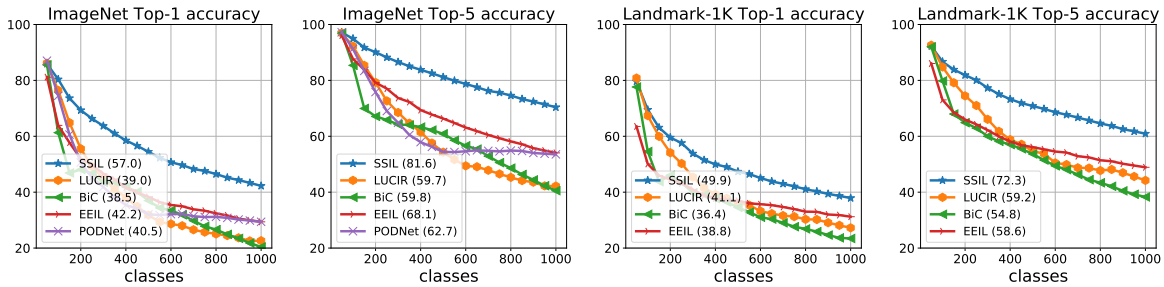
(a) $T = 10$ and $|\mathcal{M}_{per}| = 5$



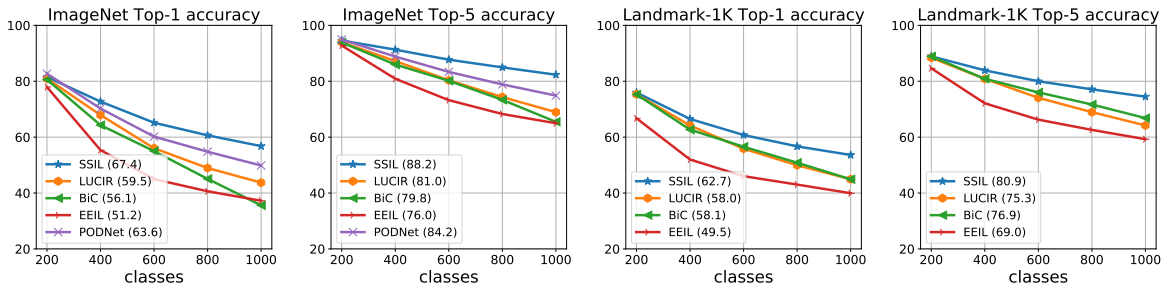
(b) $T = 10$ and $|\mathcal{M}_{per}| = 10$



(c) $T = 10$ and $|\mathcal{M}_{per}| = 20$

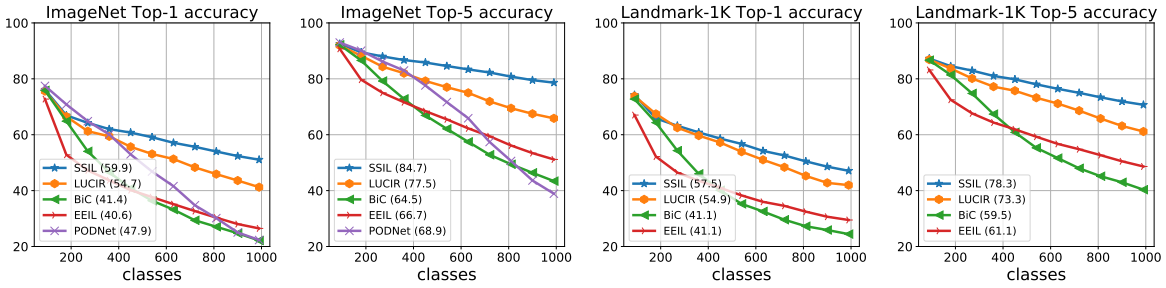


(d) $T = 20$ and $|\mathcal{M}_{per}| = 10$

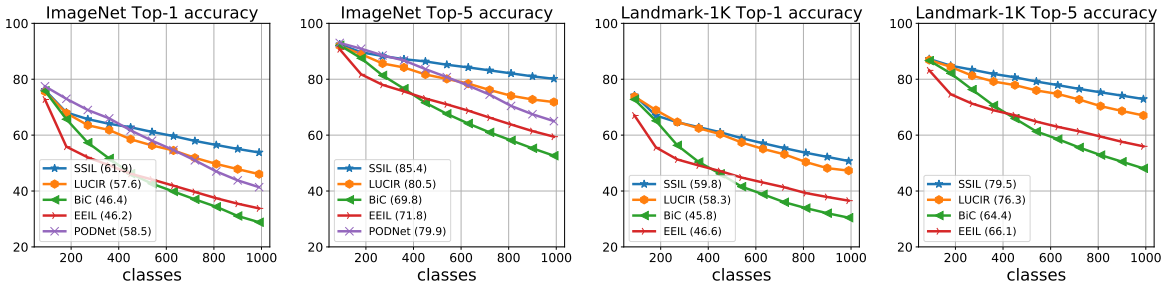


(e) $T = 5$ and $|\mathcal{M}_{per}| = 10$

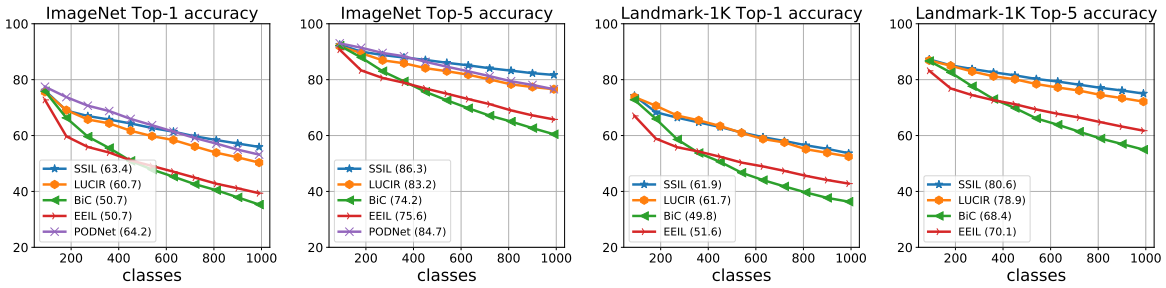
Figure 4. Top-1 and Top-5 accuracy on base and growing memory setting at ImageNet-1K and Landmark-1K



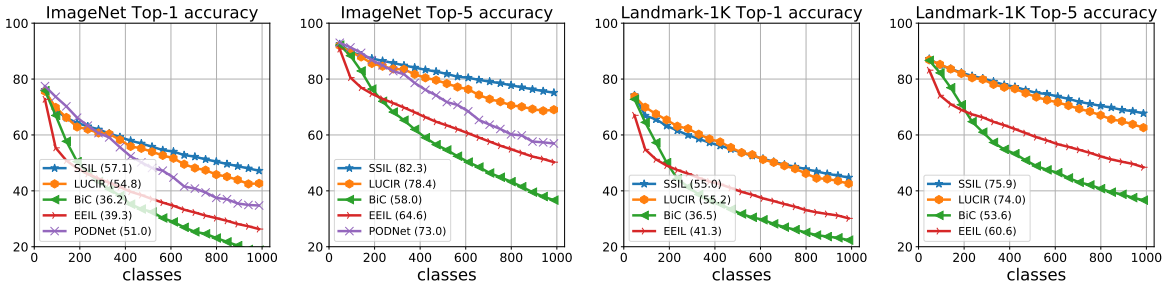
(a) $T = 10$ and $|\mathcal{M}| = 5$



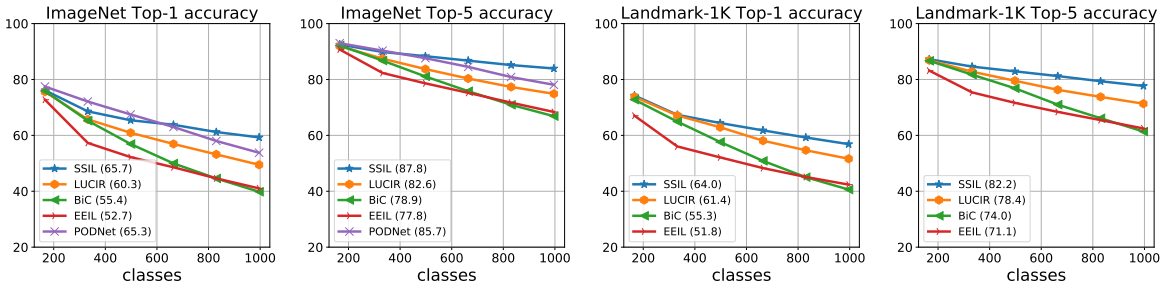
(b) $T = 10$ and $|\mathcal{M}| = 10k$



(c) $T = 10$ and $|\mathcal{M}| = 20k$

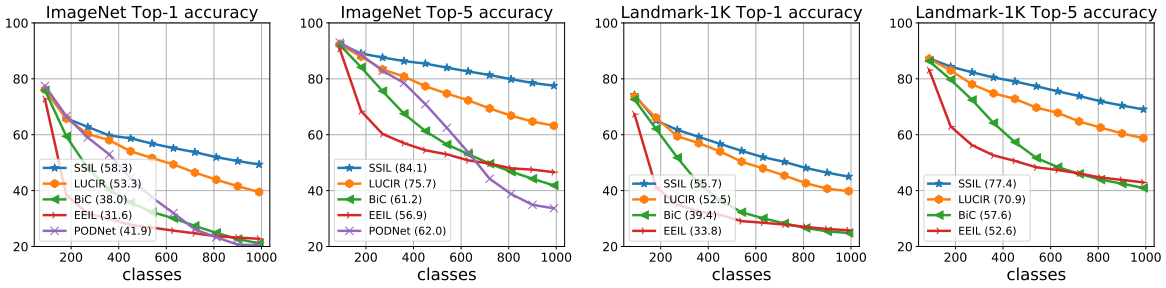


(d) $T = 20$ and $|\mathcal{M}| = 10k$

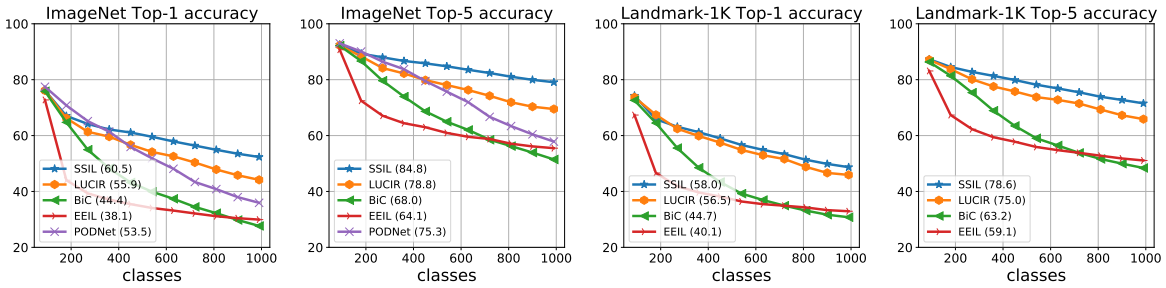


(e) $T = 5$ and $|\mathcal{M}| = 10k$

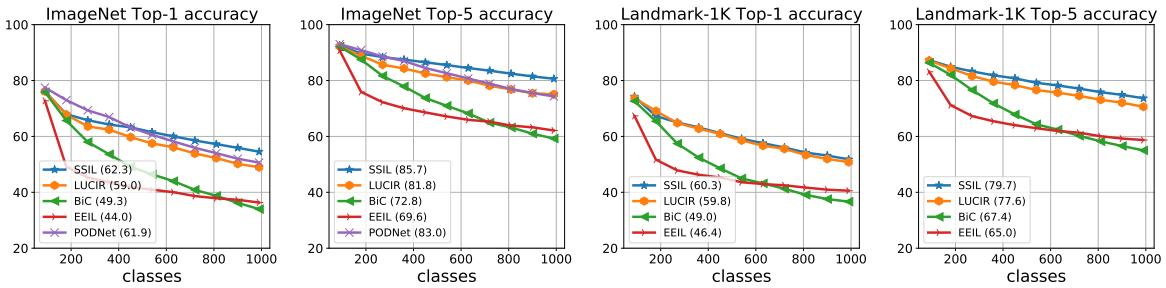
Figure 5. Top-1 and Top-5 accuracy on large base and fixed memory setting at ImageNet-1K and Landmark-1K



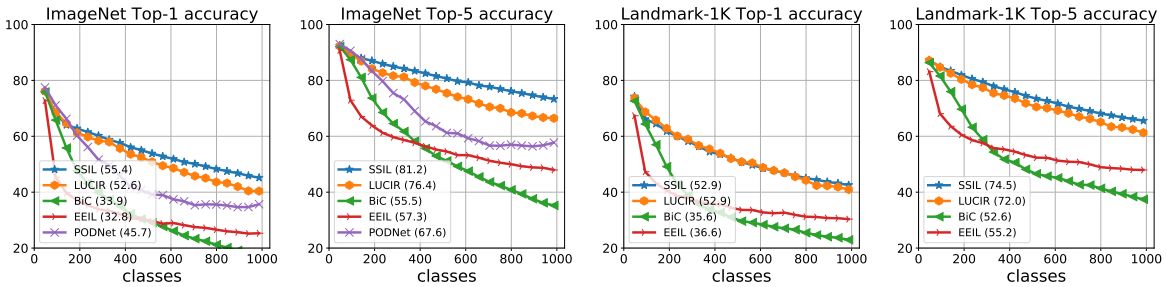
(a) $T = 10$ and $|\mathcal{M}_{per}| = 5k$



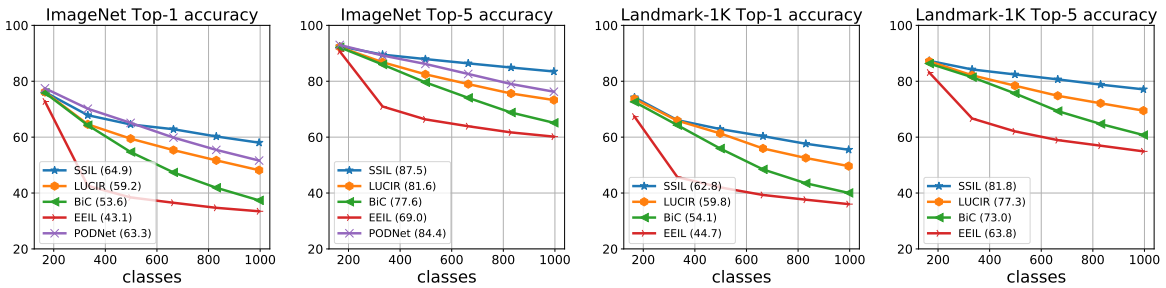
(b) $T = 10$ and $|\mathcal{M}_{per}| = 10$



(c) $T = 10$ and $|\mathcal{M}_{per}| = 20$



(d) $T = 20$ and $|\mathcal{M}_{per}| = 10$



(e) $T = 5$ and $|\mathcal{M}_{per}| = 10$

Figure 6. Top-1 and Top-5 accuracy on large base and growing memory setting at ImageNet-1K and Landmark-1K

References

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

[2] Eden Belouadah, Adrian Popescu, and Ioannis Kanellos. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 2020. 1

[3] 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. 2, 4

[4] 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. 1

[5] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer vision-ECCV 2020-16th European conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XX*, volume 12365, pages 86–102. Springer, 2020. 1

[6] 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, 3, 4

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[8] 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. 1, 2, 4

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

[10] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 2

[11] 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

[12] 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. 2, 4

[13] 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. 2