PODNet: Pooled Outputs Distillation for Small-Tasks Incremental Learning: Supplementary Material

Arthur Douillard[1,2], Matthieu Cord[2,3], Charles Ollion[1], Thomas Robert[1], and Eduardo Valle[4]

[1] Heuritech, Paris, France
{arthur.douillard,thomas.robert,charles.ollion}@heuritech.com
[2] Sorbonne University, Paris, France
matthieu.cord@sorbonne-universite.fr
[3] valeo.ai, Paris, France
[4] University of Campinas, Campinas, Brazil
dovalle@dca.fee.unicamp.br

1 Spatial-based distillation without POD-flat

In Table 3.b of the main paper, we compared distillation loss alternatives to our final POD-spatial. In this table, the spatial-based losses were evaluated with POD-flat. In Table 1, we evaluate those same losses without POD-flat. "*None*" is using only our LSC classifier without any distillation losses. Notice that POD-spatial —and its sub-components POD-width and POD-height– are the only losses barely affected by POD-flat's absence. Note that all alternative losses were tuned on the validation set to get the best performance, including those from external papers. Still, our proposed loss, POD-spatial, outperforms all, both with and without POD-flat.

Table 1. Comparison of distillation losses based on intermediary features. All losses evaluated without POD-flat.

Loss	NME	CNN
None	41.56	40.76
POD-pixels	42.21	40.81
POD-channels	55.91	50.34
POD-gap	57.25	53.87
POD-width	61.25	57.51
POD-height	61.24	57.50
POD-spatial	**61.42**	**57.64**
GradCam [1]	41.89	42.07
Perceptual Style [5]	41.74	40.80

Table 2. Effect of the initial task size and the M_{total} on the models performance. We report the average incremental accuracy

(a) Evaluation of an easier memory constraint ($M_{\text{total}} = 2000$)

	Nb. steps	
Loss	50	10
iCaRL [8]	42.34	56.52
BiC [9]	48.44	55.03
UCIR (NME) [3]	54.08	62.89
UCIR (CNN) [3]	55.20	63.62
PODNet (NME)	**62.47**	**64.60**
PODNet (CNN)	**61.87**	**64.68**

(b) Varying initial task size, with $M_{\text{per}} = 20$, and followed by 50 to 90 tasks made of a single class

	Initial task size				
Loss	10	20	30	40	**50**
iCaRL [8]	40.97	41.28	43.38	44.35	44.20
BiC [9]	41.58	40.95	42.27	45.18	47.09
UCIR (NME) [3]	42.33	40.81	46.80	46.71	48.57
UCIR (CNN) [3]	43.25	41.69	47.85	47.51	49.30
PODNet (NME)	**45.09**	**49.03**	**55.30**	**57.89**	**61.40**
PODNet (CNN)	**44.95**	**47.68**	**52.88**	**55.42**	**57.98**

2 Robustness of our model

Refer to tables 2a and 2b for the experiments with respectively a more flexible memory ($M_{\text{total}} = 2000$) and various initial task size (from 10 to 50 classes).

3 Implementation details

For all datasets, images are augmented with random crops and flips. For CIFAR100, we additionally change image intensity by a random value in the range [-63, 63]. We train our model for 160 epochs for CIFAR100, and 90 epochs for both ImageNet100 and ImageNet100, with a SGD optimizer with momentum of 0.9. For all datasets, we start with a learning rate of 0.1, a batch size of 128, and cosine annealing scheduling. The weight decay is $5 \cdot 10^{-4}$ for CIFAR100, and $1 \cdot 10^{-4}$ for ImageNet100 and ImageNet1000. For CIFAR100 we set model hyperparameters $\lambda_c = 3$ and $\lambda_f = 1$, while for ImageNet100 and 1000 we set $\lambda_c = 8$ and $\lambda_f = 10$. Our model uses POD-spatial and POD-flat except when explicitly stated otherwise. Following Hou et al. [3], we multiply both losses by the adaptive scaling factor: $\lambda = \sqrt{N/T}$ with N being the number of seen classes and T the number of classes in the current task.

For all POD losses (e.g. POD-spatial, POD-gap, etc.), before sum-pooling, following Zagoruyko and Komodakis [10], we take the features to the power of 2 element-wise. The vector resulting from the pooling is then L2 normalized.

4 Number of proxies per class

While our model's expressiveness increases with more proxies in $\mathcal{L}_{\text{LSC}}$, it remains fairly stable for values between 5 and 15, thus, for simplicity, we kept it fixed to 10 in all experiments.

In initial experiments, we had the following pairs for the number of clusters (k) and average incremental accuracy (acc): k=1, acc=56.80%; k=2, 57.14%; k=4, acc=57.40%; k=6, acc=57.46%; k=8, acc=57.95%, and k=10, acc=57.98% — i.e., a 1.18 p.p. improvement moving from k=1 to k=10. On ImageNet100, with 10 steps/tasks (increments of give classes per task), moving from k=1 to k=10 improved 1.51 p.p. on acc.

5 Reproducibility

Code Dependencies The Python version is 3.7.6. We used the PyTorch [7] (version 1.2.0) deep learning framework and the libraries Torchvision (version 0.4.0), NumPy [6] (version 1.17.2), Pillow (version 6.2.1), and Matplotlib [4] (version 3.1.0). The CUDA version is 10.2. Initial experiments were done with the data loaders library Continuum [2]. PODNet's full code is released at: `github.com/arthurdouillard/incremental_learning.pytorch`.
We provide all configuration files necessary to reproduce results, including seeds and class ordering.

References

1. Dhar, P., Singh, R.V., Peng, K.C., Wu, Z., Chellappa, R.: Learning without memorizing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 1
2. Douillard, A., Lesort, T.: Continuum, data loaders for continual learning. https://github.com/Continvvm/continuum (2020). https://doi.org/10.5281/zenodo.3759673 3
3. Hou, S., Pan, X., Change Loy, C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 2
4. Hunter, J.D.: Matplotlib: A 2d graphics environment. Computing in Science & Engineering (2007) 3
5. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV) (2016) 1
6. Oliphant, T.E.: A guide to NumPy. Trelgol Publishing USA (2006) 3
7. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: Advances in Neural Information Processing Systems (NeurIPS), Autodiff Workshop (2017) 3
8. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 2
9. Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y.: Large scale incremental learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 2
10. Zagoruyko, S., Komodakis, N.: Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. Proceedings of the International Conference on Learning Representations (ICLR) (2016) 2