Online Continual Learning under Extreme Memory Constraints - Supplementary Material

Enrico Fini[1,2], Stéphane Lathuilière[3], Enver Sangineto[1], Moin Nabi[2], and Elisa Ricci[1,4]

[1] University of Trento, Trento, Italy
[2] SAP AI Research, Berlin, Germany
[3] LTCI, Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France
[4] Fondazione Bruno Kessler, Trento, Italy
enrico.fini@unitn.it

1 Longer Task Sequences

It is desirable for a CL agent to be capable of learning long sequences of tasks, or large sets of classes. In fact, in some practical scenarios, it becomes necessary to keep information about old tasks even after learning a large amount of new tasks.

To this end, as a supplement to the experiments shown in the main paper, we test our method on Stanford 40 Actions [4], a dataset for understanding human actions in still images. The dataset contains 40 classes; we divided them into 10 tasks, composed by 4 classes each. The bounding boxes, provided with the data, are used to crop the images which are then resized as in [2] to 224x224 pixels to fit them into a ResNet18 [1]. We used the network architecture that is provided with PyTorch [3] and was used in [1] for ImageNet. Apart from these slight modifications, we used the same experimental protocol as described in section 4 of the main paper, as well as the same baselines for Memory-Constrained Online Continual Learning (MC-OCL).

Table 1. Final test accuracy on Stanford 40 Actions with 10 tasks

Method	**T0**	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**AVG**
Finetune	34.6	33.8	26.5	33.4	29.7	35.1	37.2	38.2	40.6	52.2	36.1
L2	34.6	39.8	36.1	34.3	40.1	37.1	38.0	38.6	38.1	44.9	38.2
BLD	41.7	39.8	43.9	44.2	45.5	44.7	49.7	49.1	48.5	50.9	**45.8**

Table 1 shows that *Batch-level Distillation (BLD)* significantly outperforms the other MC-OCL baselines on most of the tasks. The only exception is the last task, in which our method performs substantially on par with *finetuning*. On average *BLD* is approximately 10% more accurate at classifying the correct action with respect to *finetuning*. This difference in performance is even more pronounced if we put the results into perspective. In fact, since each subset of the dataset contains 4 classes, chance level is set to 25% accuracy. This means that *finetune* looses almost all the information it had accumulated on old tasks. Conversely, given the online assumption, single-task performance is bounded to the accuracy that the model can achieve in one pass over the data. We found that, on average, this value lays around 52%. It follows that the average performance of *BLD* (45.8%) is only approximately 6% lower than the upper bound.

For what concerns the naïve *L2* baseline (refer to section 4.3 for further detail), we notice that it performs poorly throughout all the tasks, only slightly counteracting catastrophic forgetting (tasks 0 to 8), but also causing underfitting on new tasks (task 9). This shows that imposing constraints on parameters is significantly less effective than using knowledge distillation. In addition, we remind the reader that the amount of memory used by *L2* is several orders of magnitude larger than the overhead needed by *BLD*.

In conclusion these additional experiments show that *BLD* is able to effectively prevent forgetting even for longer task sequences in the context of MC-OCL, making it a good candidate for real world applications.

2 Constant Probability Bank Size

In the paper we mainly focused on minimizing the amount of memory that is passed between batches (*inter-batch*). Instead, in this section, we analyze more closely the memory overhead we need to allocate during a learning step (*intra-batch*).

While by definition our method (*BLD*) does not require any inter-batch information transfer, it still needs to save the probability bank during the *warm-up* stage, in order to perform distillation during the *joint* stage. The size of the probability bank $|\hat{Y}|$, expressed as the number of *Float32* that are stored during task t, can be calculated with the following formula:

$$|\hat{Y}| = (t-1)\, N_A\, N_C\, |\mathcal{B}|, \tag{1}$$

where N_A and N_C are the number of augmentations and classes per task respectively, and $|\mathcal{B}|$ is the size (number of images) of a generic batch drawn from the current task data D_t. Equation 1 shows that the size of the probability bank increases linearly with the number of tasks. Although this memory overhead is negligible if compared, for instance, to the memory expansion due to the gradients, we want to show that it is possible to reduce it to constant size, with just a minor loss in performance. In fact, it is enough to allocate a fixed size storage P for the probability bank, and then partition it into equal size subsets as new tasks are added. Hence, the amount of memory dedicated to a generic old task $t' < t$ decreases as t increases, in formulas: $|\hat{Y}^{t'}| = P\,/\,(t-1)$. Since t, N_C and $|\mathcal{B}|$ are fixed, we reduce the number of augmentations N_A to fit the probabilities in the shrinked partition.

In table 2 we test our modified model (*BLD Const.*) according to the new definition of probability bank on CIFAR10 with 5 tasks. The results for the other baselines are lifted from the main paper and reported here for easier comparison.

Table 2. Final test accuracy on CIFAR10. *Intra-batch M.O.* stands for *Intra-batch Memory Overhead*; *BLD Const.* represents *BLD* with constant probability bank size.

Method	**T0**	**T1**	**T2**	**T3**	**T4**	**AVG**	**Intra-batch M.O.**
Finetune	59.6	58.2	66.8	80.2	97.0	72.3	-
L2	75.5	65.3	73.5	81.3	96.8	78.5	44.8MB
BLD	83.4	83.2	79.5	88.1	97.0	**86.2**	32kB
BLD Const.	75.8	77.4	70.5	82.4	97.0	80.6	**8kB**

The experimental protocol is exactly the same as for previous experiments on CIFAR10.

The size of the memory bank P for *BLD Const.* is set to 2000 *Float32*, or equivalently 8kB of data. This value is equal to the size of the probability bank in the case of normal *BLD* during task 1. The difference here is that for the subsequent tasks, the memory bank keeps growing linearly for *BLD* while it remains constant for *BLD Const.*. The maximum value of the Intra-batch Memory Overhead is reported in table 2.

The results show, once again, a clear advantage for our distillation-based methods on all tasks. For what concerns *BLD Const.*, we notice that despite it uses significantly less memory compared to any other method, it still outperforms the MC-OCL baselines. With respect to the original version of *BLD*, our experiments show an evident loss of performance, though justified by the 4x shrinkage of the probability bank.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
2. Masana, M., Tuytelaars, T., van de Weijer, J.: Ternary feature masks: continual learning without any forgetting. arXiv preprint arXiv:2001.08714 (2020)
3. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
4. Yao, B., Jiang, X., Khosla, A., Lin, A.L., Guibas, L., Fei-Fei, L.: Human action recognition by learning bases of action attributes and parts. In: 2011 International Conference on Computer Vision. pp. 1331–1338. IEEE (2011)