In Defense of the Learning Without Forgetting for Task Incremental Learning

Supplementary

A ResNets architectures

In section 4.1 of the main paper, we offered to use various ResNet architectures for LwF: RN-20, RN-32, RN-62, WRN-20-W2, and WRN-20-W5. All these networks share a common structure but differ in width or depth. This structure starts with a single convolutional layer of 16 filters with a kernel size of 3x3 and stride 1, followed by 3 groups of "blocks". Each group is parameterized by the number of blocks, width, and stride for the first block in the group. The baseline width (width factor equals 1) of each group is 16, 32, and 64, and strides 1, 2, and 2 respectively.

To implement the blocks, the class of BasicBlock from the PyTorch framework is employed. Each block contains 2 convolutional layers with a kernel size of 3x3 and a skip connection. The structure ends with an adaptive average pooling of size 1x1. Moreover, each convolutional layer is followed by a batch normalization layer and a ReLU activation function.
The parameters of the architectures in our work:

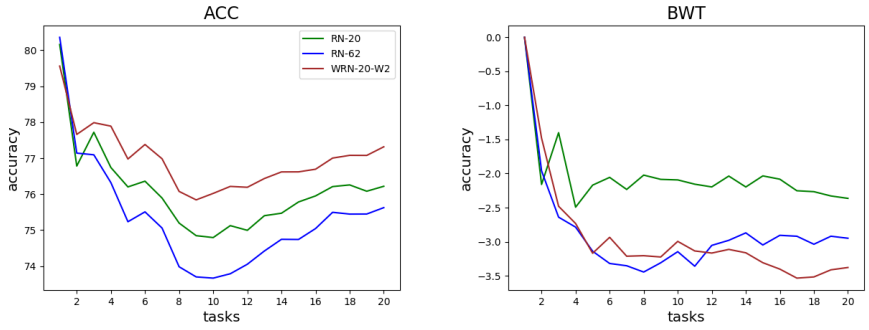
- **RN-20** a width factor of 1 and 3 blocks in each group.
- **RN-32** a width factor of 1 and 5 blocks in each group.
- **RN-62** a width factor of 1 and 10 blocks in each group.
- **WRN-20-W2** a width factor of 2 and 3 blocks in each group.
- **WRN-20-W5** a width factor of 5 and 3 blocks in each group.

B LwF with AlexNet and data augmentations

In the main text the best architecture is tested for LwF with data augmentations, namely WRN-20-W5. In this section we provide results for AlexNet-like architectures with augmentations as well, the results are provided in Tab. I. We observe that the data augmentations does not provide recovery from the harmful Dropout component in AlexNet-D. However, it does provide performance boost for AlexNet-ND, as expected.

		CIFAR 5-Split		CIFAR 10-Split		CIFAR 20-Split		Tiny-ImageNet 40-Split	
Arch.	Aug.	BWT	ACC	BWT	ACC	BWT	ACC	BWT	ACC
AlexNet-D		-39.9 ± 1.4	36.6 ± 1.5	-52.9 ± 1.2	28.1 ± 1.3	-54.4 ± 1.1	31.3 ± 0.8	-50.5 ± 1.0	25.0 ± 0.4
AlexNet-D	✓	-46.2 ± 1.8	38.0 ± 1.7	-56.9 ± 0.8	30.1 ± 0.7	-58.0 ± 0.5	31.6 ± 0.3	52.6 ± 0.8	25.9 ± 0.5
AlexNet-ND		-1.8 ± 0.6	56.6 ± 1.1	-2.9 ± 0.2	67.0 ± 1.0	-3.1 ± 0.3	75.5 ± 0.6	-2.8 ± 0.3	66.9 ± 0.8
AlexNet-ND	✓	-0.5 ± 0.4	69.5 ± 1.1	-0.7 ± 0.3	76.7 ± 0.9	-0.9 ± 0.2	83.5 ± 0.5	-1.4 ± 0.3	73.2 ± 0.7

Table I: LwF results with AlexNet-like architecture with data augmentations. all results are produced by us and are averaged over five runs with standard deviations. D=Dropout, ND=No Dropout.



(a) (b)

Figure I: The evolution in time of the accuracy and the forgetting for CIFAR 20-Split with LwF and different width and depth architectures, average over 5 random seeds. No augmentation used in these experiments. (a) ACC (Eq. 1) after learning task t as a function of t. (b) BWT (Eq. 2) after learning task t as function of t.

C Width vs. depth for LwF

In Fig. I we offer another view on the effect of different depth and width for LwF. The results are provided for the baseline ResNet architecture, RN-20, and two comparable capacity architectures. One with greater depth, RN-62, and another with greater width, WRN-20-W2. The results show that although RN-62 and WRN-20-W2 share a similar amount of forgetting, from task 2 onward RN-62 under-performs with respect to ACC.

This suggests that LwF with a deeper ResNet network is struggling to acquire new knowledge while keeping the previous one. Comparing RN-62 with RN-20 highlights a more severe problem where LwF is struggling to utilize deeper networks both in terms of ACC and BWT. However, increased width has a positive effect on performance over time, even at the price of increased forgetting. Fortunately, we were able to mitigate this increased forgetting with data augmentations, which not only reduced forgetting substantially but also increased ACC.

D EWC and IMM with WRN-20-W5

In our experiments we found EWC and IMM (both MEAN and MODE variants) to perform poorly with ResNet architectures and specifically with WRN-20-W5. The results, for this architecture, can be found in Tab. II. As can be seen, using WRN-20-W5 the methods are not competitive and perform lower than when using the AlexNet-like architecture, as quoted in the main paper. This performance gap suggests that the methods require modifications in order to enjoy more modern architecture, like ResNet. We attribute this to the challenge imposed by the batch normalization layers.

		CIFAR 5-Split		CIFAR 10-Split		CIFAR 20-Split		Tiny-ImageNet 40-Split	
Method	Aug.	BWT	ACC	BWT	ACC	BWT	ACC	BWT	ACC
EWC		-11.0 ± 2.4	46.8 ± 2.1	-24.8 ± 3.6	39.8 ± 2.6	-33.5 ± 5.5	40.9 ± 5.3	-31.4 ± 2.0	34.8 ± 1.6
EWC	✓	-11.6 ± 3.9	60.1 ± 4.4	-31.9 ± 2.6	46.8 ± 2.4	-45.7 ± 4.1	38.2 ± 3.4	-45.1 ± 3.1	31.1 ± 3.5
IMM-MEAN		-12.3 ± 8.5	24.6 ± 8.7	-3.5 ± 5.6	27.3 ± 4.4	-2.9 ± 1.3	33.3 ± 2.0	$+0.2 \pm 1.5$	28.1 ± 1.3
IMM-MEAN	✓	-16.9 ± 4.7	29.3 ± 3.2	-4.9 ± 2.5	29.4 ± 3.1	-3.3 ± 2.1	30.9 ± 1.3	-1.6 ± 4.0	26.8 ± 3.0
IMM-MODE		-22.7 ± 6.3	39.4 ± 3.9	-34.8 ± 4.0	34.5 ± 3.1	-47.3 ± 4.0	30.3 ± 3.3	-42.5 ± 2.1	27.5 ± 1.4
IMM-MODE	✓	-39.8 ± 2.1	44.0 ± 2.1	-52.0 ± 3.3	35.2 ± 2.7	-58.8 ± 5.4	30.2 ± 5.2	-52.4 ± 2.7	26.4 ± 2.5

Table II: EWC and IMM results with WRN-20-W5. all results are produced by us and are averaged over five runs with standard deviations.

E ACC and BWT over time

In Fig. II we provide the BWT and ACC scores after learning each task for CIFAR-100 with 5 and 10 splits. These results were omitted from the main text for brevity and provided here as complementary results.

Similarly to the results shown in the paper (main text Fig. 2), the advantage of LwF over the baseline methods is evident. LwF can learn new tasks with a similar level of performance to the previous ones while maintaining the knowledge from the previous tasks. In contrast, both EWC and IMM fail to do so. For HAT, the difference in performance between different CIFAR-100 splits, where the performance is more stable for a short sequence of tasks, could point to an insufficient per task capacity problem. However, since LwF can both learn new tasks and maintain old ones with similar capacity, this points to an under-utilization of the network capacity. Thus, we suspect that HAT is not scalable for long task sequences even with larger networks. Although HyperCL seems to have very competitive results for these splits, its shortcoming is revealed in the main paper, looking at a longer sequence of tasks, such as Tiny-ImageNet.

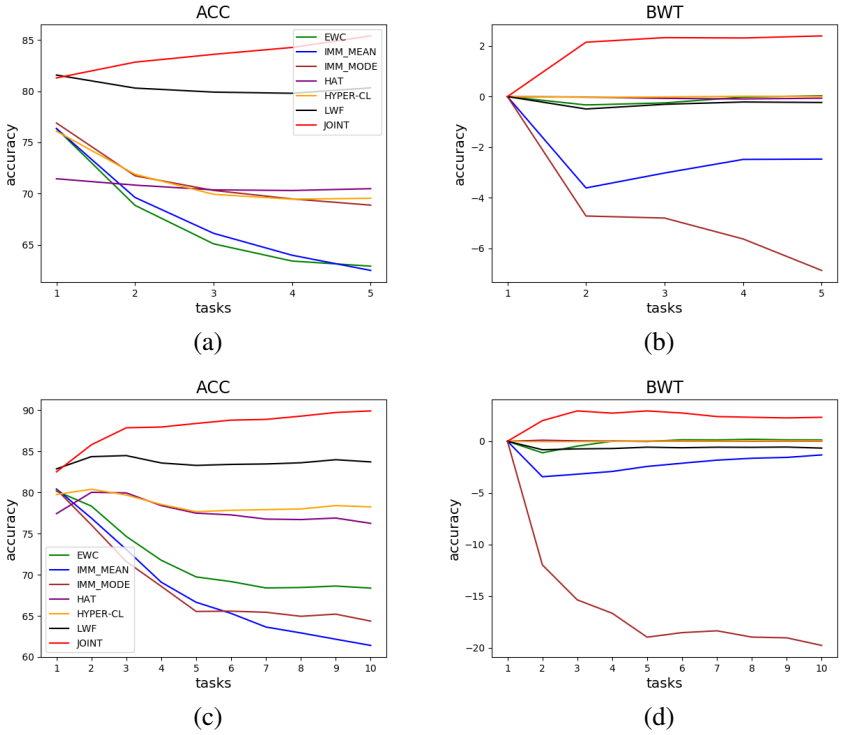


Figure II: The evolution in time of the accuracy and the forgetting, for the best performing setting of each method average over 5 random seeds. ACC (Eq. 1) after learning task t as a function of t. BWT (Eq. 2) after learning task t as function of t. (a) & (b) results over time for CIFAR 5-Split and (c) & (d) results over time for CIFAR 10-Split.