Supplementary material for "ScaIL: Classifier Weights Scaling for Class Incremental Learning"

Eden Belouadah, Adrian Popescu
Université Paris-Saclay, CEA, Département Intelligence Ambiante et Systèmes Interactifs,
91191 Gif-sur-Yvette, France
eden.belouadah,adrian.popescu@cea.fr

1. Introduction

In this supplementary material, we provide:

- a more detailed discussion of G_{IL}, the proposed aggregated evaluation score;
- results for fine tuning with $\mathcal{B} = 0$, i.e. without past exemplars memory;
- supplementary experiments related to the role of distillation in class incremental learning;
- algorithm implementation details.

2. Measuring the performance gap of IL algorithms

The proposal of aggregated measures is important for tasks which are evaluated in a large number of configurations [6, 8]. Building on previous work regarding such measures, the authors of [8] list eight criteria which should be met by global evaluation metrics when evaluating universal visual representations: (1) coherent aggregation, (2) significance, (3) merit bonus, (4) penalty malus, (5) penalty for damage, (6) independence to outliers, (7) independence to reference and (8) time consistency. They note that none of the global evaluation measures can fulfill all criteria simultaneously. However, their formulation which inspired us to propose G_{IL} fulfills the maximum number of criteria. While the IL context is different from that of universal representations, a majority of criteria from [8] are relevant here. The aggregation is easier in our work since the use of $Full$ as reference score is a natural upper bound for incremental learning algorithms. The aggregation of scores is natural in G_{IL} since all scores are compared to a single reference. The significance criterion, put forward in [6] is only implicitly modeled because configurations which give the largest gain contribute more to the global score. The merit bonus refers to the proportionality of the reward with respect to the reference method and is modeled through the denominator of Equation 3 of the paper. The penalty for damage and the penalty malus are not applicable since all methods penalize the performance compared to the upper bound. The independence to outlier methods has low effect in our case since it refers to the contributions of individual configurations. Since G_{IL} averages the contributions of a relatively large number of contributions, the risk related to outliers is rather reduced. Naturally, the more datasets and configurations are tested, the more robust the score will be. However, the computational resources needed for training in IL are large and we consider that the use of four datasets, with three memory sizes and three incremental learning splits gives a fair idea about the behavior of each algorithm. Time consistency is respected since methods are not compared to each other but only to a reference which is stable if the same deep model and data are used across time. A question remains whether datasets of different sizes should be given the same weights in the score but using weighting would further complicate the evaluation measure.

3. Fine tuning without memory

States	$\mathcal{Z} = 10$			
Dataset	ILSVRC	VGGFace2	Landmarks	CIFAR-100
LwF	43.80	48.30	46.34	79.49
FT^{noMem}	20.64	21.28	21.29	21.27
FT^{L2}	20.64	21.27	21.27	21.27
FT_{init}	60.95	90.90	68.77	55.05
FT_{init}^{L2}	51.57	76.84	61.42	47.48
$ScaIL$	21.96	23.06	22.31	33.49

Table 1: Top-5 accuracy of fine tuning without memory ($\mathcal{B} = 0$) for the four datasets with $\mathcal{Z} = 10$ states. For reference, we also present LwF [3], which is equivalent to $iCaRL$ [7] without memory.

Table 1 provides results obtained with fine tuning without memory for past classes ($\mathcal{B} = 0$) and $\mathcal{Z} = 10$ states. Trends are similar for the other $\mathcal{Z}$ values tested in the paper which are not presented here. The accuracy drops significantly for FT since the network cannot rehearse knowledge related to past classes. Catastrophic forgetting is more severe and past classes become unrecognizable in the current state. The accuracy of FT^{noMem} is mostly due to the recognition rate of new classes. When $\mathcal{Z} = 10$, they represent between a half and a tenth of the total number of classes for states $S = 1$ and $S = 9$, the first and the last incremental state respectively. The accuracy for past classes is close to random. Since $ScaIL$ depends heavily on the weights of past classes in the current state, its performance drops significantly. LwF [3] includes a distillation component which is clearly useful in absence of memory. It outperforms FT and $ScaIL$ for all datasets by a very large margin. This finding reinforces the conclusions of [7] regarding the positive role of distillation in incremental learning without memory.

4. Supplementary experiments related to distillation in IL

In Figure 1, we provide detailed top-5 accuracy per incremental state for FT, $FT^{distill}$ and $iCaRL$ for $\mathcal{B} = 0.5\%$ and $\mathcal{Z} = 50$ states. The largest value of Z from the paper was chosen in order to observe the behavior with and without distillation for a small number of classes per incremental state. For ILSVRC, VGGFace2 and Landmarks, the difference between FT and $FT^{distill}$ is small for initial incremental states, increases a lot afterwards and tends to decrease toward the end of the process but remains very large. This behavior is explained by the fact that, since past memory is only $\mathcal{B} = 0.5\%$, the number of exemplars per class becomes very small toward the end. For instance, $\mathcal{B}$ includes 5000 images for ILSVRC and there will be only 5 exemplars per class in the last states of the incremental process. It is noticeable that rehearsal in FT still works with such a small number of exemplars. These finding provides further support to the results reported in the paper regarding the negative role of distillation at large scale for imbalanced datasets when a memory of the past is available. Confirming the results from [7], distillation is indeed useful for CIFAR-100, where its performance is slightly better than that of FT. Also, the introduction of an external classifier in $iCaRL$ is clearly useful.

In Table 2 and Figure 2, we extend the analysis of top-1 types of errors presented in Table 2 and Figure 4 of the paper to the four datasets. The $e(p, p)$ errors related to the last incremental state are overrepresented for all four datasets compared. However, the errors toward the first incremental state are also better represented for VGGFace2 and even become dominant for Landmarks and CIFAR-100. This behavior is probably due to the fact that the initial state is stronger for easier tasks. In these cases, the model evolves to a lesser extent compared to ILSVRC, a more complex visual task.

		Incremental states								
		$\mathcal{S}^1$	$\mathcal{S}^2$	$\mathcal{S}^3$	$\mathcal{S}^4$	$\mathcal{S}^5$	$\mathcal{S}^6$	$\mathcal{S}^7$	$\mathcal{S}^8$	$\mathcal{S}^9$
ILSVRC										
FT	$c(p)$	2117	2995	3415	3875	3653	4451	4558	5003	3119
	$e(p,p)$	156	450	807	1363	1842	2710	2626	3932	2388
	$e(p,n)$	2727	6555	10778	14762	19505	22839	27816	31065	39493
	$c(n)$	4151	4322	4103	4141	4267	4304	4247	4378	4248
	$e(n,n)$	809	638	875	828	716	674	743	595	741
	$e(n,p)$	40	40	22	31	17	22	10	27	11
$FT^{distill}$	$c(p)$	850	1008	1355	1355	1195	1344	1419	1543	1562
	$e(p,p)$	472	1746	3700	4999	6904	8246	10771	13400	14556
	$e(p,n)$	3678	7246	9945	13646	16901	20410	22810	25057	28882
	$c(n)$	3645	3834	3597	3607	3744	3754	3605	3766	3662
	$e(n,n)$	1043	793	928	905	785	776	828	692	751
	$e(n,p)$	312	373	475	488	471	470	567	542	587
VGGFace2										
FT	$c(p)$	4168	7718	11062	14293	15953	19614	21075	24690	24196
	$e(p,p)$	89	282	611	947	1354	2170	3203	3827	4929
	$e(p,n)$	743	2000	3327	4760	7693	8216	10722	11483	15875
	$c(n)$	4825	4834	4866	4865	4881	4879	4887	4874	4883
	$e(n,n)$	155	143	118	119	108	102	101	108	108
	$e(n,p)$	20	23	16	16	11	19	12	18	9
$FT^{distill}$	$c(p)$	1729	2109	1886	1787	1520	1657	1412	1199	1131
	$e(p,p)$	242	1455	2553	3360	4056	5766	6248	6506	7838
	$e(p,n)$	3029	6436	10561	14853	19424	22577	27340	32295	36031
	$c(n)$	4620	4637	4694	4740	4747	4714	4693	4685	4728
	$e(n,n)$	299	239	236	203	212	224	218	248	216
	$e(n,p)$	81	124	70	57	41	62	89	67	56
Landmarks										
FT	$c(p)$	1670	3072	4476	5550	6564	7626	8081	9303	10309
	$e(p,p)$	38	131	318	616	879	1005	1340	1961	2237
	$e(p,n)$	292	797	1206	1834	2557	3369	4579	4736	5454
	$c(n)$	1945	1970	1959	1956	1973	1966	1975	1973	1971
	$e(n,n)$	51	27	35	37	24	27	25	23	27
	$e(n,p)$	4	3	6	7	3	7	0	4	2
$FT^{distill}$	$c(p)$	901	1011	859	815	788	769	622	533	419
	$e(p,p)$	159	831	1770	2617	3194	3880	4708	5889	6744
	$e(p,n)$	940	2158	3371	4568	6018	7351	8670	9578	10837
	$c(n)$	1893	1893	1902	1910	1937	1913	1949	1926	1936
	$e(n,n)$	66	53	58	61	37	53	36	52	38
	$e(n,p)$	41	54	40	29	26	34	15	22	26
CIFAR-100										
FT	$c(p)$	366	614	675	605	686	950	779	692	467
	$e(p,p)$	10	181	312	288	641	974	835	732	601
	$e(p,n)$	624	1205	2013	3107	3673	4076	5386	6576	7932
	$c(n)$	791	873	886	866	848	859	834	888	915
	$e(n,n)$	196	114	103	131	146	127	159	104	80
	$e(n,p)$	13	13	11	3	6	14	7	8	5
$FT^{distill}$	$c(p)$	719	1160	1507	1706	1988	2195	2349	2404	2251
	$e(p,p)$	91	457	847	1210	1800	2551	2929	3499	3743
	$e(p,n)$	190	383	646	1084	1212	1254	1722	2097	3006
	$c(n)$	694	742	735	752	723	767	708	786	814
	$e(n,n)$	78	62	40	53	48	35	57	38	28
	$e(n,p)$	228	196	225	195	229	198	235	176	158

Table 2: Top-1 correct and wrong classifications for vanilla fine tuning (FT) and fine tuning with distillation ($FT^{distill}$) for the four datasets with $\mathcal{Z} = 10$ and $\mathcal{B} = 0.5\%$.

5. Algorithm implementation details

We used the Github[1] public implementation from [7] to run $iCaRL$ on TensorFlow [1] with the same hyperparameters and training settings provided by the authors. Hyperparameters are as follows: $lr = 2.0$, $weight\ decay = 0.00001$, $momentum = 0.9$, $batch\ size = 128$. $iCaRL$ was run with a total of 60 epochs for the large datasets and for 70 epochs for CIFAR-100. The learning rate is divided by 5 at $epoch = \{20, 30, 40, 50\}$ for the large datasets and

[1] https://github.com/srebuffi/iCaRL

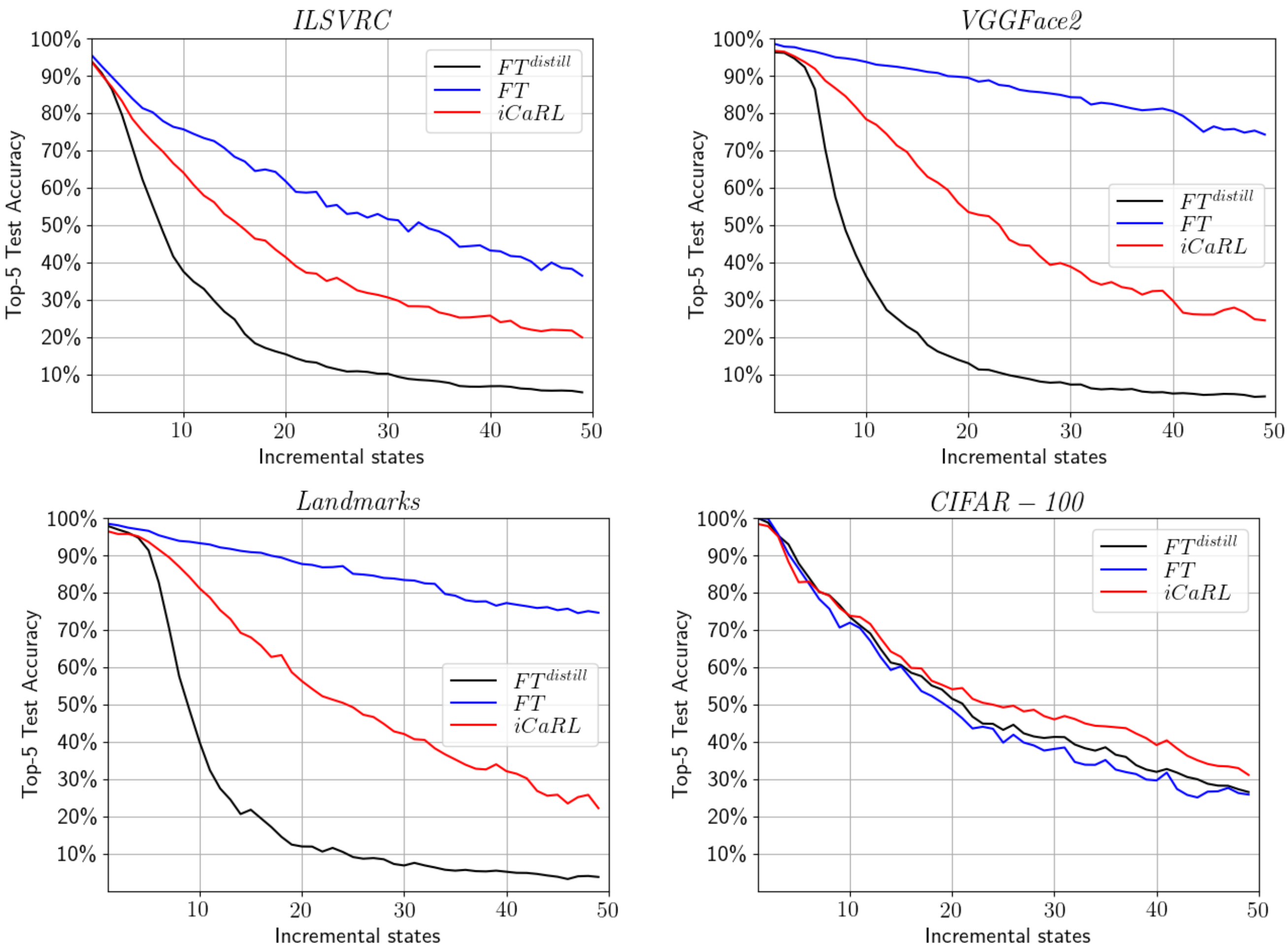

Figure 1: Detailed Top-5 Test accuracy for the four datasets with $\mathcal{Z} = 50$ and memory $\mathcal{B} = 0.5\%$. In this experiment, a comparison is done between FT, $FT^{distill}$ and $iCaRL$ to analyze the role of distillation.

at $epoch = \{49, 63\}$ for CIFAR-100. We tried to optimize the learning process by changing hyperparameters but couldn't improve the results presented by the original authors.

BiC [9] was also run using the public Github implementation[2] provided by the authors and the same hyperparameters.

All the other methods were implemented in Pytorch [4] with $batch\ size = 256$ (128 for CIFAR-100), $weight\ decay = 0.0001$ (0.0005 for CIFAR-100) and a $momentum = 0.9$. The first non-incremental state was trained for 100 epochs for large datasets and 300 epochs for CIFAR-100. The learning rate is set to 0.1 and divided by 10 when the error plateaus for 10 consecutive epochs (60 epochs for CIFAR-100). FT was run for 35 epochs (60 epochs for CIFAR-100). The only change compared to the standard training was to set initial learning rate per incremental state at $lr = \frac{0.1}{k+1}$, with $1 \leq k \leq \mathcal{Z} - 1$. This results in a gain of less than 1 top-5 accuracy point for ILSVRC with $\mathcal{Z} = 10$ and $\mathcal{B} = 0.5\%$. During training, the learning rate is divided by 10 when the error plateaus for 5 epochs (15 epochs for CIFAR-100).

The balanced fine tuning performed after FT in FT^{BAL} was run for 15 more epochs (30 epochs for CIFAR-100) and the learning rate is reinitialized to $lr = \frac{0.01}{k+1}$. We also tried to initialize the balanced fine tuning with $lr = \frac{0.1}{k+1}$ and continue from the last learning rate of the imbalanced fine tuning but results were lower. Equally important, training with more epochs did not provide any gain.

The fixed representation in $DeeSIL$ [2] is trained only with data from the first incremental batch. No external data was used to ensure that the method is comparable with the others. SVM training is done using the *scikit-learn* framework [5]. SVMs were optimized by dividing the IL training set to $\frac{90}{10}$ train/val subsets

[2]https://github.com/wuyuebupt/LargeScaleIncrementalLearning

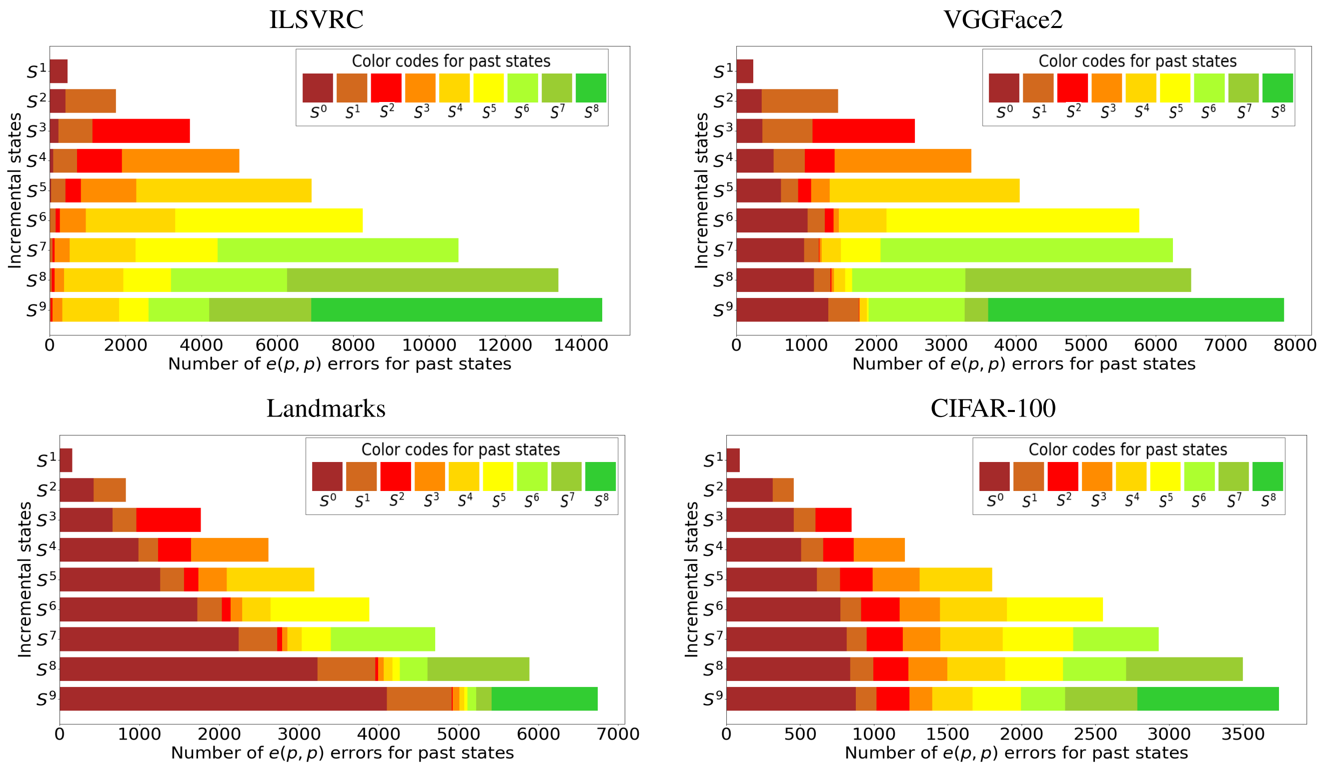

Figure 2: Detail of past-past errors $e(p,p)$ for individual states of $FT^{distill}$ on the four datasets with $\mathcal{Z} = 10$ and $\mathcal{B} = 0.5\%$. In each state, errors due to the latest past state are over-represented as a result of learning its associated state with an imbalanced training set. *Best viewed in color.*

and iterate through the values of the regularizer $C = \{0.0001, 0.001, 0.01, 1, 10, 100, 1000\}$. The optimal value was retained for each dataset configuration. SVMs are optimized only for the non-incremental state. The regularizer is then frozen and used for the subsequent incremental states. We used the default values of the other hyper-parameters provided in $sklearn$.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. A. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zhang. Tensorflow: A system for large-scale machine learning. *CoRR*, abs/1605.08695, 2016.

[2] E. Belouadah and A. Popescu. Deesil: Deep-shallow incremental learning. In *Computer Vision - ECCV 2018 Workshops - Munich, Germany, September 8-14, 2018, Proceedings, Part II*, pages 151–157, 2018.

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, CVPR, 2016.

[4] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *Advances in Neural Information Processing Systems Workshops*, NIPS-W, 2017.

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. VanderPlas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in python. *CoRR*, abs/1201.0490, 2012.

[6] S. Rebuffi, H. Bilen, and A. Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 506–516, 2017.

[7] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Conference on Computer Vision and Pattern Recognition*, CVPR, 2017.

[8] Y. Tamaazousti, H. Le Borgne, C. Hudelot, M. E. A. Seddik, and M. Tamaazousti. Learning more universal representations for transfer-learning. *arXiv:1712.09708*, 2017.

[9] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 374–382, 2019.