ICICLE: Interpretable Class Incremental Continual Learning – Supplementary Materials

Dawid Rymarczyk[1,2,3,*] Joost van de Weijer[4,5] Bartosz Zieliński[1,3,6]
Bartłomiej Twardowski[4,5,6]
[1] Faculty of Mathematics and Computer Science, Jagiellonian University
[2] Doctoral School of Exact and Life Sciences, Jagiellonian University [3] Ardigen SA
[4] Autonomous University of Barcelona [5] Computer Vision Center [6] IDEAS NCBR
[*] dawid.rymarczyk@doctoral.uj.edu.pl

Additional experimental setup details

Here we present additional details on the experimental setup. We performed a hyperparameter search for λ_{dist} ($\lambda_{dist} \in \{10.0, 5.0, 0.0, 0.5, 0.1, 0.05, 0.01, 0.005, 0.001, 0.0005, 0.0001\}$). We use Adam optimizer with a learning rate of 0.001 and parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set the batch size to 75 and use input images of resolution $224 \times 224 \times 3$. The weights of the network are initialized with Xavier's normal initializer.

We perform a warmup training where the weights of f are frozen for 10 epochs, and then we train the model until it converges with 12 epochs early stopping. We use the learning schema presented in Table 1. Depending on the number of tasks, we perform warm-up training with $\{5, 5, 4\}$ epochs and joint training phase for $\{21, 15, 11\}$, longer with fewer tasks. Similarly, we perform prototype projection every $\{10, 7, 5\}$ epoch. So with more tasks, we perform fewer training epochs (Table 1).

Results on Stanford Cars

Table 2 depicts how the ICICLE method works on the Stanford Cars dataset compared to other baseline methods. Results are consistent with the ones on CUB-200-2011 and show that ICICLE outperforms all standard CL learning methods adapted to ProtoPNet architecture.

Comparison to GDumb

Additionally, we compared our approach with GDumb [1], a baseline method in CL learning, in scenarios involving 3, 5, and 10 images per class with 4 tasks learning achieving 20.3, 34.2, 57.6, and 13.0, 26.7, 48.8 for task-aware and task-agnostic respectively. ICICLE outperformed GDumb with a small number of examples, and task-aware for GDumb-10 was the only exception where GDumb achieved a higher accuracy score.

Distant initialization

In Table 5, we showed that proximity-based initialization is most beneficial. However, here in Figure 1, we show how initialization of prototypical parts at a distance from already existing once generates concepts that are too general or carry information about background.

Task-recency bias compensation

Here, in Table 3 we show what is the influence of task-recency bias compensation on each task accuracy for task agnostic scenario. After compensation, the accuracy on task 1 increased the most, but at the same time accuracy on all other tasks was sacrificed. However, the average accuracy after the compensation is increased.

Detailed results after learning each task

In Table 4, and Table 5 we show detailed accuracies of the ICICLE after learning each task for CUB-200-2011 on a single run with the same seed in four and ten tasks learning scenarios.

Analysis of the hyperparameters for baselines

In Table 6, Table 7, and Table 8 we show the influence of the hyperparameters for each of the baseline methods, EWC, LWF, and LWM, respectively. Based on that, the parameters of these methods were chosen for comparison with ICICLE.

Phase	Model layers	Learning rate	Scheduler	Weight decay	Duration
Warm-up	add-on 1×1 convolution	$1 \cdot 10^{-3}$	None	None	5, 5, 4 epochs
	prototypical layer	$1 \cdot 10^{-3}$			
Joint	convolutions f	$1 \cdot 10^{-4}$	by half every 5 epochs	10^{-4}	21, 15, 10 epochs early stopping
	add-on 1×1 convolution	$1 \cdot 10^{-3}$			
	prototypical layer	$1 \cdot 10^{-3}$			

Table 1. Learning schema for the ICICLE method.

Method	Avg. inc. task-aware accuracy			Avg. inc. task-task agnostic accuracy		
	4 tasks	7 tasks	14 tasks	4 tasks	7 tasks	14 tasks
Freezing	0.572 ± 0.031	0.518 ± 0.041	0.486 ± 0.026	0.309 ± 0.012	0.155 ± 0.031	0.092 ± 0.014
Finetuning	0.216 ± 0.009	0.167 ± 0.011	0.149 ± 0.012	0.182 ± 0.006	0.124 ± 0.013	0.057 ± 0.001
EWC	0.456 ± 0.021	0.315 ± 0.037	0.287 ± 0.041	0.258 ± 0.019	0.152 ± 0.022	0.011 ± 0.009
LWM	0.459 ± 0.072	0.416 ± 0.048	0.305 ± 0.022	0.233 ± 0.026	0.171 ± 0.016	0.080 ± 0.008
LWF	0.375 ± 0.021	0.356 ± 0.024	0.250 ± 0.020	0.230 ± 0.011	0.171 ± 0.005	0.092 ± 0.008
ICICLE	**0.654 ± 0.014**	**0.645 ± 0.003**	**0.583 ± 0.048**	**0.335 ± 0.005**	**0.203 ± 0.010**	**0.116 ± 0.018**

Table 2. Average incremental accuracy comparison for different numbers of tasks on Stanford Cars, demonstrating the negative impact of the high number of tasks to be learned on models' performance. Despite this trend, ICICLE outperforms the baseline methods across all task numbers.

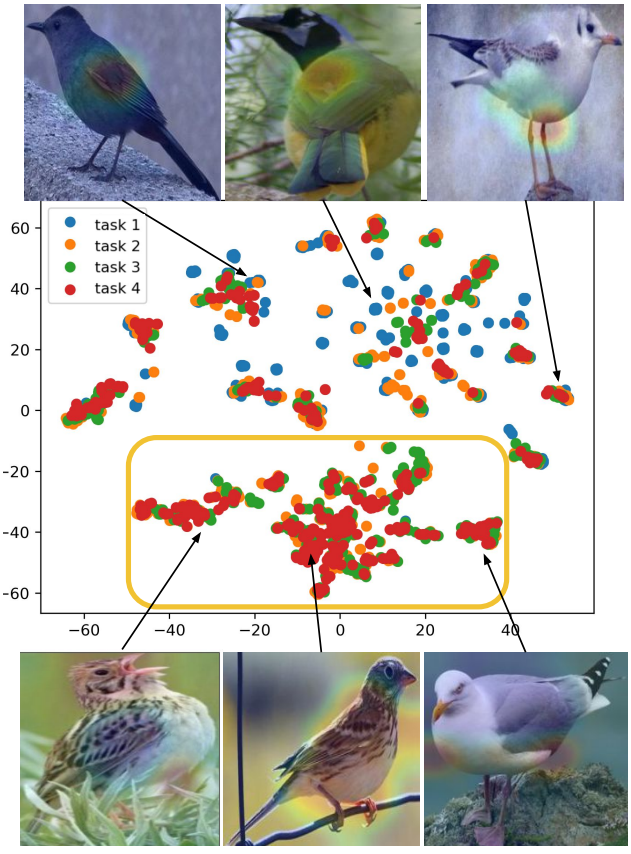

Figure 1. Image depicts the 2D TSNE projection of prototypes. One can observe that there is a cluster of prototypes from all tasks except the first one (yellow box). This is observable with distant initialization and prototype visualizations show that those prototypical parts are representing the background or vague concepts.

	Task 1	Task 2	Task 3	Task 4	Avg
Task aware	0.514	0.717	0.725	0.698	0.663
before	0.028	0.301	**0.434**	**0.575**	0.335
after	**0.233**	**0.365**	0.314	0.486	**0.350**

Table 3. Task-recency bias compensation influence of a single run. Results show that our compensation method balance more the results per each task in task agnostic scenario.

After	Task-aware accuracy					Task-task agnostic accuracy				
	Task 1	Task 2	Task 3	Task 4	Avg	Task 1	Task 2	Task 3	Task 4	Avg
Task 1	0.806	NA	NA	NA	0.806	0.806	NA	NA	NA	0.806
Task 2	0.740	0.759	NA	NA	0.750	0.089	0.747	NA	NA	0.418
Task 3	0.622	0.736	0.759	NA	0.706	0.033	0.633	0.549	NA	0.404
Task 4	0.514	0.717	0.725	0.698	0.663	0.028	0.484	0.378	0.505	0.349

Table 4. Results of the ICICLE method before task-recency compensation for four task learning scenario after each learning episode.

After	Task-aware accuracy										
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9	Task 10	Avg
Task 1	0.920	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.920
Task 2	0.666	0.869	NA	NA	NA	NA	NA	NA	NA	NA	0.767
Task 3	0.462	0.818	0.858	NA	NA	NA	NA	NA	NA	NA	0.713
Task 4	0.420	0.751	0.774	0.774	NA	NA	NA	NA	NA	NA	0.680
Task 5	0.314	0.625	0.680	0.672	0.784	NA	NA	NA	NA	NA	0.615
Task 6	0.268	0.538	0.627	0.617	0.760	0.747	NA	NA	NA	NA	0.593
Task 7	0.265	0.476	0.584	0.617	0.713	0.706	0.769	NA	NA	NA	0.590
Task 8	0.258	0.413	0.551	0.555	0.667	0.634	0.741	0.764	NA	NA	0.573
Task 9	0.253	0.398	0.494	0.492	0.598	0.587	0.701	0.745	0.852	NA	0.569
Task 10	0.244	0.371	0.462	0.441	0.573	0.560	0.667	0.729	0.816	0.803	0.567
	Task-agnostic accuracy										
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9	Task 10	Avg
Task 1	0.920	NA	NA	NA	NA	NA	NA	NA	NA	NA	0.920
Task 2	0.010	0.869	NA	NA	NA	NA	NA	NA	NA	NA	0.439
Task 3	0.0	0.060	0.854	NA	NA	NA	NA	NA	NA	NA	0.305
Task 4	0.0	0.0	0.339	0.751	NA	NA	NA	NA	NA	NA	0.273
Task 5	0.0	0.0	0.030	0.323	0.746	NA	NA	NA	NA	NA	0.220
Task 6	0.0	0.0	0.004	0.090	0.451	0.684	NA	NA	NA	NA	0.205
Task 7	0.0	0.0	0.0	0.020	0.193	0.432	0.712	NA	NA	NA	0.194
Task 8	0.0	0.0	0.0	0.020	0.073	0.233	0.497	0.643	NA	NA	0.181
Task 9	0.0	0.0	0.0	0.0	0.035	0.138	0.338	0.435	0.676	NA	0.180
Task 10	0.0	0.0	0.0	0.0	0.016	0.070	0.214	0.285	0.484	0.643	0.171

Table 5. Results of the ICICLE method before task-recency compensation for ten task learning scenario after each learning episode.

α	Task-aware accuracy					Task-task agnostic accuracy				
	0.01	0.1	1.0	5.0	10.0	0.01	0.1	1.0	5.0	10.0
	0.185	0.329	0.441	0.197	0.167	0.170	0.185	0.213	0.168	0.144

Table 6. Influence of the $alpha$ parameter in EWC on the accuracy of ProtoPNet architecture in four task learning scenario.

γ	Task-aware accuracy					Task-task agnostic accuracy				
	0.001	0.01	0.1	1.0	10.0	0.001	0.01	0.1	1.0	10.0
	0.240	0.240	0.431	0.355	0.231	0.209	0.209	0.212	0.209	0.209

Table 7. Influence of the γ parameter in LWM on the accuracy of ProtoPNet architecture in four task learning scenario.

λ	Task-aware accuracy					Task-task agnostic accuracy				
	0.001	0.01	0.1	1.0	10.0	0.001	0.01	0.1	1.0	10.0
	0.232	0.232	0.238	0.359	0.249	0.207	0.210	0.210	0.231	0.221

Table 8. Influence of the λ parameter in LWF on the accuracy of ProtoPNet architecture in four task learning scenario.

References

[1] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 524–540. Springer, 2020. 1