Supplementary Materials: Resurrecting Old Classes with New Data for Exemplar-Free Continual Learning

Dipam Goswami[1,2] Albin Soutif–Cormerais[1,2] Yuyang Liu[3,4] Sandesh Kamath[1,2]
Bartłomiej Twardowski[1,2,5] Joost van de Weijer[1,2]

[1]Department of Computer Science, Universitat Autònoma de Barcelona
[2]Computer Vision Center, Barcelona [3]University of Chinese Academy of Sciences
[4]Shenyang Institute of Automation, Chinese Academy of Sciences [5]IDEAS-NCBR
{dgoswami, albin, skamath, btwardowski, joost}@cvc.uab.es, liuyuyang@sia.cn

1. Training settings and hyperparameters

Since, the current approaches are not designed and optimized for the *small-start* settings used in our work, we adapt relevant methods and optimize them for these settings and achieve comparable baselines to our approach. We list the exact experimental settings to enable reproducibility.

Augmentations: As implemented in PyCIL [12], for CIFAR-100, we use the same augmentation policy which consists of small random transformations like contrast or brightness changes. Similarly, for the other datasets, we use the default set of augmentations which include random crop and random horizontal flip. For a fair evaluation, we use the same set augmentations for all the methods.

LwF: In CIFAR-100, TinyImageNet and ImageNet-Subset datasets for the first task, similar to PyCIL [12], we use a starting learning rate of 0.1, momentum of 0.9, batch size of 128, weight decay of 5e-4 and trained for 200 epochs, with the learning rate reduced by a factor of 10 after 60, 120, and 160 epochs, respectively. For subsequent tasks, we used an initial learning rate of 0.05 for CIFAR-100 and ImageNet-Subset and 0.001 for TinyImageNet. The learning rate is reduced by a factor of 10 after 45 and 90 epochs and trained for a total of 100 epochs. We set the the temperature to 2 and the regularization strength to 10 for CIFAR-100 and TinyImageNet and 5 for ImageNet-Subset. For the fine-grained datasets, we use a learning rate of 0.01 for the first task and a learning rate of 0.005 for subsequent tasks. The regularization strength is set to 20.

For the NCM classifier, SDC and our proposed method ADC, we use the same training settings as LwF.

SDC: For SDC [11], we set the hyperparameters $\sigma = 0.3$ for CIFAR-100, TinyImagenet and for the fine-grained datasets. For Imagenet-Subset, we set $\sigma = 1.0$.

PASS: We follow the implementation of PASS [13] from PyCIL [12] and set $\lambda_{fkd} = 10$ and $\lambda_{proto} = 10$.

SSRE: We follow the implementation of SSRE [14] from PyCIL [12] and set $\lambda_{fkd} = 10$ and $\lambda_{proto} = 10$.

FeTrIL: For first task, we use a learning rate of 0.1 for CIFAR-100, TinyImageNet and ImageNet-Subset and follow the exact same settings as the original implementation [5]. For the fine-grained datasets, we use a learning rate of 0.01 for the first task.

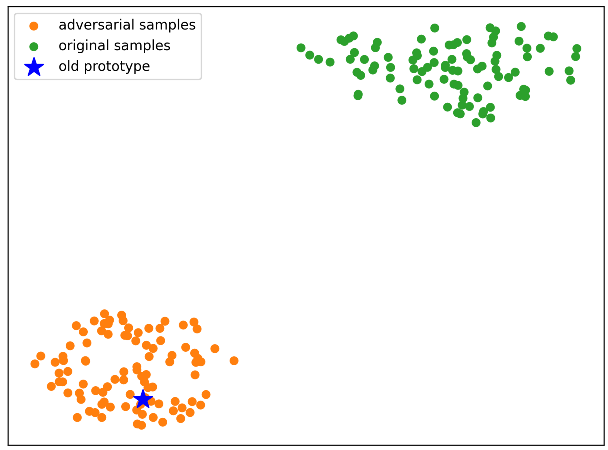


Figure 1. The t-SNE plot demonstrates that the adversarial samples generated using our proposed method lie close to the target old class prototype compared to the closest current task samples (in green) and can thus be reliably used for drift estimation.

FeCAM: We use the same training setting as LwF for the first task training. FeCAM [1] requires no training after the first task and stores the prototypes and covariance matrices from all the classes. Similar to the original implementation, we use the covariance shrinkage hyperparameters of (1,1) and the Tukey's normalization value of 0.5.

ADC: We use a α value of 25, iterations $i = 3$ and number of closest samples $m = 100$ for all the datasets.

2. Robustness to different class orders

In CIL, the order of classes can influence the performance and thus we shuffle the class orders and observe how ADC and the existing methods like LwF, NCM, SDC, FeTrIL and FeCAM perform. While we used the seed 1993 following previous works [4–6, 11] for the results reported in the main paper, here we use four different seeds 0, 1, 2, 3 and report

Method	T = 5		T=10	
	A_{last}	A_{inc}	A_{last}	A_{inc}
LwF [3]	45.67 ± 1.37	62.12 ± 1.08	26.59 ± 2.44	45.60 ± 2.52
NCM	$\underline{52.68 \pm 0.57}$	$\underline{65.91 \pm 0.4}$	40.53 ± 2.74	$\underline{56.21 \pm 3.55}$
SDC [11]	52.26 ± 3.0	63.88 ± 1.23	$\underline{40.65 \pm 1.82}$	56.18 ± 3.0
FeTrIL [5]	45.52 ± 0.33	60.84 ± 0.46	37.0 ± 0.57	52.08 ± 0.51
FeCAM [1]	46.93 ± 0.34	61.49 ± 0.55	33.13 ± 0.93	48.10 ± 1.27
ADC (Ours)	$\mathbf{58.12 \pm 1.42}$	$\mathbf{69.29 \pm 1.17}$	$\mathbf{45.43 \pm 3.03}$	$\mathbf{59.59 \pm 4.11}$

Table 1. Evaluation of EFCIL methods with mean and standard deviation using 5 random seeds for CIFAR-100. Best results in **bold** and second best results are underlined.

Method	T = 5		T=10	
	A_{last}	A_{inc}	A_{last}	A_{inc}
LwF [3]	39.03 ± 0.43	50.96 ± 0.93	27.75 ± 0.51	40.04 ± 0.96
NCM	38.76 ± 0.18	51.74 ± 0.8	28.07 ± 0.97	$\underline{42.86 \pm 0.98}$
SDC [11]	$\underline{40.28 \pm 0.37}$	$\underline{52.21 \pm 0.89}$	$\underline{28.15 \pm 0.67}$	42.09 ± 1.03
FeTrIL [5]	29.94 ± 0.83	45.08 ± 0.98	23.6 ± 0.42	37.41 ± 0.63
FeCAM [1]	26.03 ± 0.49	41.11 ± 0.93	23.78 ± 0.48	37.30 ± 1.23
ADC (Ours)	$\mathbf{41.29 \pm 0.47}$	$\mathbf{52.36 \pm 0.95}$	$\mathbf{32.68 \pm 0.43}$	$\mathbf{44.89 \pm 1.03}$

Table 2. Evaluation of EFCIL methods with mean and standard deviation using 5 random seeds for TinyImageNet. Best results in **bold** and second best results are underlined.

Method	T = 5		T=10	
	A_{last}	A_{inc}	A_{last}	A_{inc}
LwF [3]	49.16 ± 1.34	68.88 ± 0.74	34.18 ± 3.69	57.96 ± 3.64
NCM	56.80 ± 1.90	72.45 ± 0.87	$\underline{44.04 \pm 1.93}$	64.43 ± 0.43
SDC [11]	$\underline{59.62 \pm 1.56}$	$\underline{74.44 \pm 0.76}$	42.68 ± 1.88	$\underline{65.26 \pm 0.59}$
FeTrIL [5]	50.52 ± 1.06	65.64 ± 0.95	40.74 ± 0.50	56.34 ± 0.76
FeCAM [1]	53.83 ± 0.46	67.88 ± 0.67	42.46 ± 0.89	57.93 ± 1.45
ADC (Ours)	$\mathbf{61.62 \pm 0.93}$	$\mathbf{75.29 \pm 0.61}$	$\mathbf{47.62 \pm 1.55}$	$\mathbf{67.03 \pm 0.46}$

Table 3. Evaluation of EFCIL methods with mean and standard deviation using 5 random seeds for ImageNet-Subset. Best results in **bold** and second best results are underlined.

the mean and standard deviation using these 5 seeds for both the last task accuracy A_{last} and the average incremental accuracy A_{inc} in Tabs. 1 to 3. The proposed method ADC outperforms SDC and NCM consistently across all settings on CIFAR-100, TinyImageNet and ImageNet-Subset. This demonstrates the robustness of ADC which improves over the existing methods irrespective of the class order.

3. Perturbation guarantee

We specifically select the closest samples to each old prototype, one at a time (see Algorithm 1) to ensure we generate adversarial samples for all the old classes. On CIFAR100 (T=10), we get an average of 59 samples out of 100, which are successfully perturbed for all old classes after the last task. While performing 5 iterations (instead of 3) generates an average of 69 successful perturbations for old classes, this does not lead to a significant accuracy change (Tab. 3a). Therefore, we have used 3 iterations in our implementation.

We analyze the position of the closest current task samples and the generated adversarial samples with respect to a target old class prototype in the old feature space using a t-SNE plot in Fig. 1. We observe that the adversarial samples lie close to the prototype, while the original samples are distant from the prototype. This validates the effectiveness of the adversarial attack in the old feature space and shows how new samples obtained using targeted adversarial attacks can be used to represent old classes. These adversarial samples behave as pseudo-exemplars and can now be used to estimate the drift of prototypes from the old to the new feature space.

4. Prompt-based Methods

Prompt-based methods [7, 9, 10] aim to learn prompt parameters that can be used with frozen pre-trained models without updating the parameters of the model. A recent work, HiDe-Prompt [8] also freezes the pre-trained ViT backbones and proposes an ensemble strategy for using prompts. Different from them, our objective is to continually learn new representations and update the backbone at every task. These methods have static features due to the frozen backbone and avoid the feature drift problem we are tackling. We think it is unfair to compare the performance of frozen pre-trained models with our method (training from scratch and updating the backbone). While freezing pre-trained models works well for mainstream datasets, it is crucial to update the backbone and learn new representations for training domain-specific models for data that are not commonly seen in pre-trained data, and thus it is necessary to develop drift compensation methods. Janson et al. [2] show that while pre-trained models with a simple NCM baseline work similar to L2P on Cifar100, they struggle on ImageNet-R with data of different styles like cartoon, graffiti, and origami.

5. Visualization of adversarial images

We visualize the original and adversarially perturbed images and the corresponding perturbations for CIFAR-100 and TinyImageNet in Fig. 2 and Fig. 3. We observe that the perturbations are perceptible in most of the adversarial images generated from low-resolution images of CIFAR-100 and TinyImageNet while for ImageNet-Subset, CUB-200 and Stanford Cars having high-resolution images of 224x224, the perturbations are not perceptible.

References

[1] Dipam Goswami, Yuyang Liu, Bartłomiej Twardowski, and Joost van de Weijer. FeCAM: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 1, 2

[2] Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A simple baseline that questions the use of pretrained-models in continual learning. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022. 3

[3] Zhizhong Li and Derek Hoiem. Learning without forgetting. *Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2017. 2

[4] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2022. 1

[5] Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. Fetril: Feature translation for exemplar-free class-incremental learning. In *Winter Conference on Applications of Computer Vision (WACV)*, 2023. 1, 2

[6] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

[7] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[8] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured suboptimality. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[9] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision (ECCV)*, 2022. 3

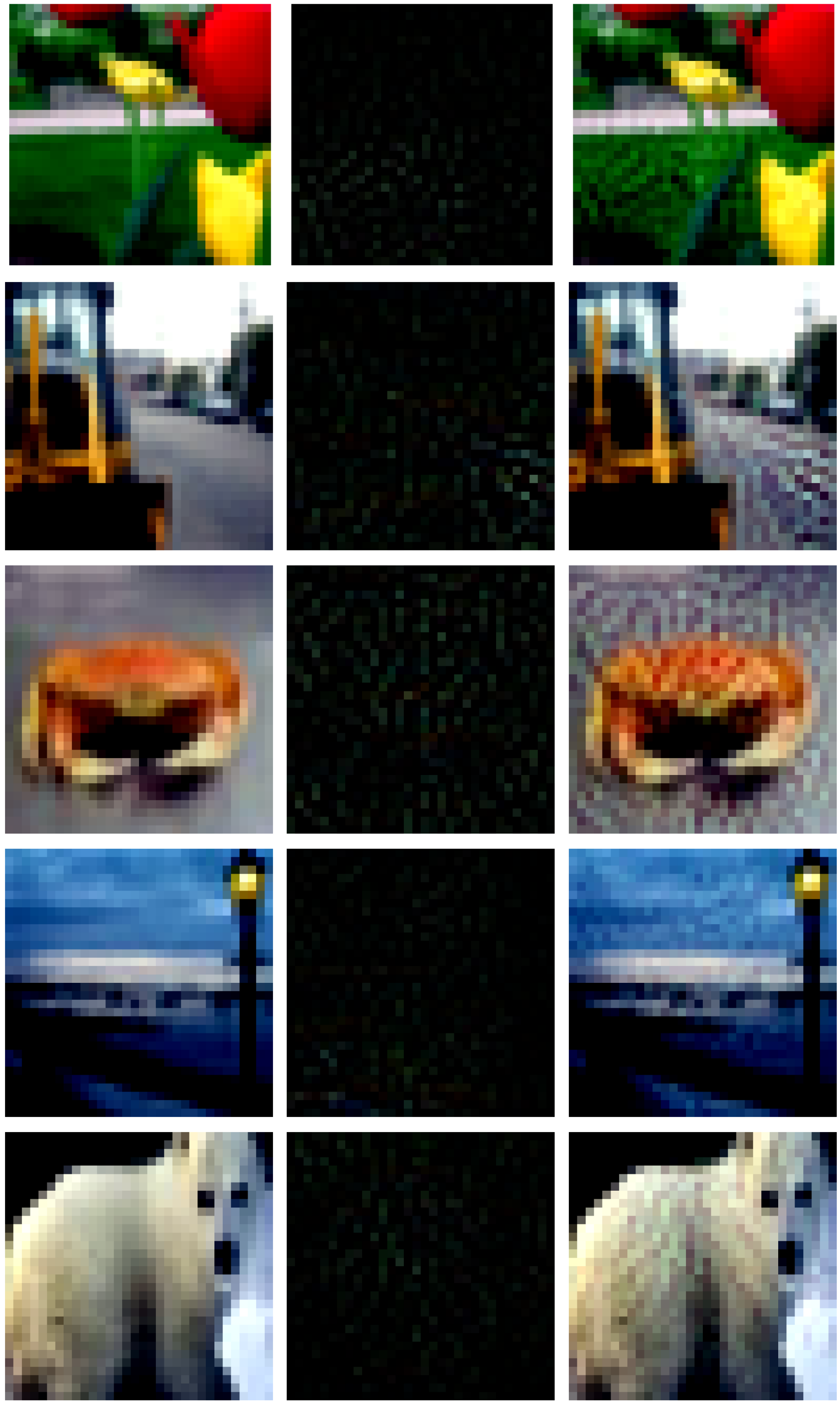
[10] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[11] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2

[12] Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, and De-Chuan Zhan. Pycil: a python toolbox for class-incremental learning. *SCIENCE CHINA Information Sciences*, 2023. 1

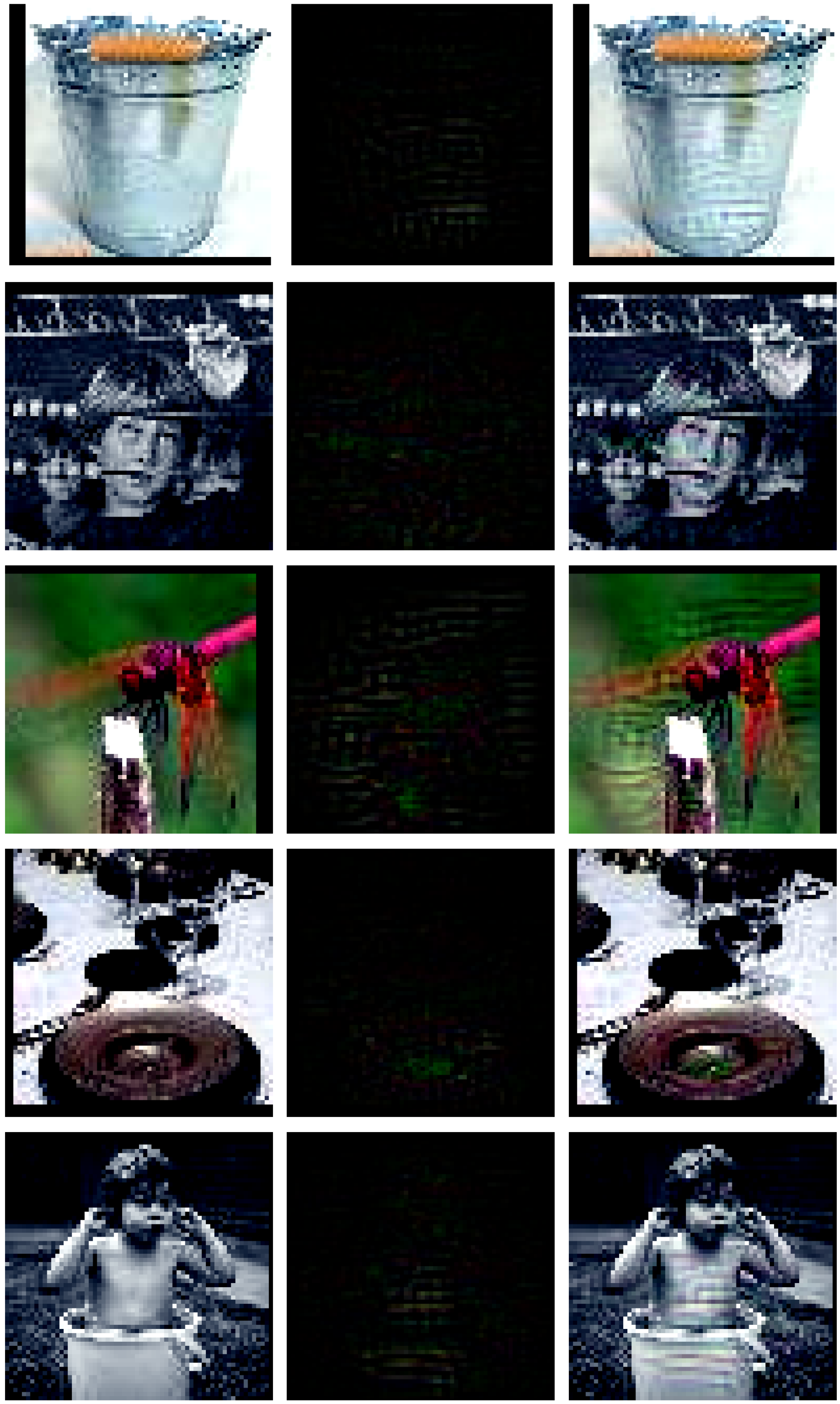
[13] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1

[14] Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1



(a) Original image (b) Adversarial perturbation (c) Adversarial image

Figure 2. Visualization of image, perturbation and the corresponding adversarial image for some samples from CIFAR-100.



(a) Original image (b) Adversarial perturbation (c) Adversarial image

Figure 3. Visualization of image, perturbation and the corresponding adversarial image for some samples from TinyImageNet.