CODA-Prompt: COntinual Decomposed Attention-based Prompting for Rehearsal-Free Continual Learning -Supplementary Materials (Appendix)-

James Seale Smith[1,2] **Leonid Karlinsky**[2,4] **Vyshnavi Gutta**[1] **Paola Cascante-Bonilla**[2,3]
Donghyun Kim[2,4] **Assaf Arbelle**[4] **Rameswar Panda**[2,4] **Rogerio Feris**[2,4] **Zsolt Kira**[1]
[1]Georgia Institute of Technology [2]MIT-IBM Watson AI Lab [3]Rice University [4]IBM Research

A. Additional Implementation Details

For all methods, we use the Adam [4] optimizer with $\beta_1 = 0.9$ and $\beta_1 = 0.999$, and a batch size of 128 images. We resize all images to 224x224 and normalize them to [0,1]. We train CIFAR-100 and DomainNet for 20 epochs, and ImageNet-R for 50 epochs (chosen to ensure models converge fully for each task). As discussed in the main text, we use the same prompting lengths and locations for L2P [10] and DualPrompt [9] as recommended by the more recent DualPrompt paper. Specifically, for Dualprompt, we use a length 5 prompt in layers 1-2 (referred to as *general* prompts) and length 20 prompts in layers 3-5 (referred to as *task-expert*). For L2P, we use a prompt pool of size 20, total prompt length of size 20, and choose 5 prompts from the pool to use during inference.

As done in DualPrompt [9], we tuned all additional hyperparameters using 20% of the training data as validation data. This resulted in using a learning rate of $1e^{-3}$ for all prompting methods (as opposed to $5e^{-3}$ as reported in DualPrompt), and a learning rate of $1e^{-4}$ for all methods which fully fine-tune the model. We searched for learning rates in the values of $\{1e^{-6}, 5e^{-6}, 1e^{-5}, 5e^{-5}, 1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}, 1e^{-2}\}$. We also found that cosine-decaying learning rate outperforms a constant learning rate (which was used in the original DualPrompt implementation). We conjecture that the reduced and decaying learning rate explain the performance boost we obtained on the 10-task ImageNet-R benchmark using our implementations.

For our method, we use a prompt length of 8 and 100 prompt components (and prompt at the same locations as DualPrompt), which were chosen with a hyperparameter sweep on validation data to have the best trade-off between performance and parameter efficiency. We searched for prompt lengths in the range of [4,40] and prompt components in the range of [5,500]. We use $\lambda = 0.1$ to weight the orthogonality regularization loss, chosen from sweeping across decade values from $1e^{-6}$ up to $1e^{2}$. As shown in Section 5.3 of our main text, we see that increasing the prompt length has little effect on our method, whereas increasing the prompt component size has strong returns all the way up to 200 components.

Finally, when implementing classification loss for fine-tuning, L2P, DualPrompt, and CODA-P, we re-use a technique from the official GitHub repo for the DualPrompt and L2P papers [9, 10] and replace the predictions from past-task logits with negative infinity when training a new task. This results in a softmax prediction of "0" for these past task classes and prevents gradients from flowing to the linear heads of past task classes. While not discussed in these papers, this technique is *crucial* for performance of these methods, as we confirmed during reproduction. Essentially, the linear layer is highly biased towards new tasks in class-incremental learning in the absence of rehearsal, so this technique prevents the linear head from learning a bias towards new classes over past classes. We note that this bias is a well-known issue [1, 11].

B. Additional Results

We report extended results, including standard deviations and additional parameters trained, for all benchmarks in Tables A (5-task ImageNet-R [3, 9]), B (10-task ImageNet-R), C (20-task Imagenet-R), D (10-task CIFAR-100 [5]), E (5-task DomainNet [8]), and A (Dual-Shift ImageNet-R).

We evaluate methods using (1) average accuracy A_N, or the test accuracy averaged over all N tasks, and (2) average forgetting [2, 6, 7] F_N, or the drop in task performance averaged over N tasks. The reader is referred to Appendix C of Wang *et al.* [9] for the formal metric definitions. We emphasize that A_N is the more important metric and encompasses both method plasticity *and* forgetting, whereas F_N provides additional context subject to the model's plasticity (i.e., a lower F_N value *and* a lower A_N value would indicate that the model's lower forgetting results from its reduced adaptivity to new tasks, which is an undesirable trait).

Table A. **Results (%) on 5-task ImageNet-R (40 classes per task)**. A_N gives the accuracy averaged over tasks, F_N gives the average forgetting, and N_{param} gives the % of trainable parameters and final parameters w.r.t. the base ViT pre-trained model. We report the mean and standard deviation over 5 trials.

Method	A_N (↑)	F_N (↓)	N_{param} (↓) Train/Final
Upper-Bound	77.13	-	100/100
ER (5000)	71.72 ± 0.71	13.70 ± 0.26	100/100
FT	18.74 ± 0.44	41.49 ± 0.52	100/100
FT++	60.42 ± 0.87	14.66 ± 0.24	100/100
LwF.MC	74.56 ± 0.59	4.98 ± 0.37	100/100
L2P++	70.83 ± 0.58	3.36 ± 0.18	0.7/100.7
Deep L2P++	73.93 ± 0.37	2.69 ± 0.10	9.6/109.6
DualPrompt	73.05 ± 0.50	$\mathbf{2.64 \pm 0.17}$	0.5/100.5
CODA-P-S	75.19 ± 0.47	2.65 ± 0.15	0.7/100.7
CODA-P	$\mathbf{76.51 \pm 0.38}$	2.99 ± 0.19	4.6/104.6

Table B. **Results (%) on 10-task ImageNet-R (20 classes per task)**. A_N gives the accuracy averaged over tasks, F_N gives the average forgetting, and N_{param} gives the % of trainable parameters and final parameters w.r.t. the base ViT pre-trained model. We report the mean and standard deviation over 5 trials.

Method	A_N (↑)	F_N (↓)	N_{param} (↓) Train/Final
Upper-Bound	77.13	-	100/100
ER (5000)	64.43 ± 1.16	10.30 ± 0.05	100/100
FT	10.12 ± 0.51	25.69 ± 0.23	100/100
FT++	48.93 ± 1.15	9.81 ± 0.31	100/100
LwF.MC	66.73 ± 1.25	3.52 ± 0.39	100/100
L2P++	69.29 ± 0.73	2.03 ± 0.19	0.7/100.7
Deep L2P++	71.66 ± 0.64	1.78 ± 0.16	9.6/109.6
DualPrompt	71.32 ± 0.62	1.71 ± 0.24	0.8/100.8
CODA-P-S	73.93 ± 0.49	$\mathbf{1.60 \pm 0.20}$	0.7/100.7
CODA-P	$\mathbf{75.45 \pm 0.56}$	1.64 ± 0.10	4.6/104.6

Table C. **Results (%) on 20-task ImageNet-R (10 classes per task)**. A_N gives the accuracy averaged over tasks, F_N gives the average forgetting, and N_{param} gives the % of trainable parameters and final parameters w.r.t. the base ViT pre-trained model. We report the mean and standard deviation over 5 trials.

Method	A_N (↑)	F_N (↓)	N_{param} (↓) Train/Final
Upper-Bound	77.13	-	100/100
ER (5000)	52.43 ± 0.87	7.70 ± 0.13	100/100
FT	4.75 ± 0.40	16.34 ± 0.19	100/100
FT++	35.98 ± 1.38	6.63 ± 0.11	100/100
LwF.MC	54.05 ± 2.66	2.86 ± 0.26	100/100
L2P++	65.89 ± 1.30	1.24 ± 0.14	0.7/100.7
Deep L2P++	68.42 ± 1.20	1.12 ± 0.13	9.6/109.6
DualPrompt	67.87 ± 1.39	1.07 ± 0.14	1.3/101.3
CODA-P-S	70.53 ± 1.24	1.00 ± 0.15	0.7/100.7
CODA-P	$\mathbf{72.37 \pm 1.19}$	$\mathbf{0.96 \pm 0.15}$	4.6/104.6

Table D. **Results (%) on 10-task CIFAR-100 (10 classes per task)**. A_N gives the accuracy averaged over tasks, F_N gives the average forgetting, and N_{param} gives the % of trainable parameters and final parameters w.r.t. the base ViT pre-trained model. We report the mean and standard deviation over 5 trials.

Method	A_N (↑)	F_N (↓)	N_{param} (↓) Train/Final
Upper-Bound	89.30	-	100/100
ER (5000)	76.20 ± 1.04	8.50 ± 0.37	100/100
FT	9.92 ± 0.27	29.21 ± 0.18	100/100
FT++	49.91 ± 0.42	12.30 ± 0.23	100/100
LwF.MC	64.83 ± 1.03	5.27 ± 0.39	100/100
L2P++	82.50 ± 1.10	1.75 ± 0.42	0.7/100.7
Deep L2P++	84.30 ± 1.03	$\mathbf{1.53 \pm 0.40}$	9.5/109.5
DualPrompt	83.05 ± 1.16	1.72 ± 0.40	0.7/100.7
CODA-P-S	84.59 ± 0.87	1.76 ± 0.28	0.6/100.6
CODA-P	$\mathbf{86.25 \pm 0.74}$	1.67 ± 0.26	4.6/104.6

Table E. **Results (%) on 5-task DomainNet (69 classes per task)**. A_N gives the accuracy averaged over tasks, F_N gives the average forgetting, and N_{param} gives the % of trainable parameters and final parameters w.r.t. the base ViT pre-trained model. We report the mean and standard deviation over 3 trials.

Method	A_N (↑)	F_N (↓)	N_{param} (↓) Train/Final
Upper-Bound	79.65	-	100/100
ER (5000)	58.32 ± 0.47	26.25 ± 0.24	100/100
FT	18.00 ± 0.26	43.55 ± 0.27	100/100
FT++	39.28 ± 0.21	44.39 ± 0.31	100/100
LwF.MC	$\mathbf{74.78 \pm 0.43}$	5.01 ± 0.14	100/100
L2P++	69.58 ± 0.39	2.25 ± 0.08	0.9/100.9
Deep L2P++	70.54 ± 0.51	2.05 ± 0.07	9.7/109.7
DualPrompt	70.73 ± 0.49	$\mathbf{2.03 \pm 0.22}$	0.6/100.6
CODA-P-S	71.80 ± 0.57	2.54 ± 0.10	0.6/100.6
CODA-P	73.24 ± 0.59	3.46 ± 0.09	4.8/104.8

Table F. **Results (%) on ImageNet-R *with covariate domain shifts***. Results are included for 5 tasks (40 classes per task). We simulate domain shifts by randomly removing 50% of the dataset's domains (e.g., clipart, paintings, and cartoon) for the training data of each task (see SM for more details). A_N gives the accuracy averaged over tasks, F_N gives the average forgetting, and N_{param} gives the % of trainable parameters and final parameters w.r.t. the base ViT pre-trained model. We report the mean and standard deviation over 5 trials.

Method	A_N (↑)	F_N (↓)	N_{param} (↓) Train/Final
Upper-Bound	77.13	-	100/100
ER (5000)	67.39 ± 0.37	11.94 ± 0.17	100/100
FT	17.93 ± 0.27	37.49 ± 0.28	100/100
FT++	54.51 ± 0.68	14.41 ± 0.33	100/100
LwF.MC	64.02 ± 1.55	7.05 ± 0.27	100/100
L2P++	65.08 ± 0.29	2.79 ± 0.32	0.7/100.7
Deep L2P++	65.74 ± 0.12	2.48 ± 0.30	9.6/109.6
DualPrompt	66.98 ± 0.08	$\mathbf{2.21 \pm 0.28}$	0.8/100.8
CODA-P-S	69.73 ± 0.18	2.35 ± 0.19	0.7/100.7
CODA-P	$\mathbf{71.35 \pm 0.08}$	2.56 ± 0.26	4.6/104.6

For each result, we calculate the mean and standard deviation over separate runs. Each run contains different shuffles of the class order; specifically, we shuffle the classes using a random seed that is set for each "trial run" - and form the tasks using this class shuffle. *Importantly, the class order and all randomized seeds, including model initialization, are consistent between different methods in the same "trial run".*

We additionally report the number of parameters *trained* (i.e., unlocked during training a task) as well as the *total* number of parameters in the final model. These are reported in % of the backbone model for easy comparison. Importantly, we design CODA-P-S to have fewer parameters than DualPrompt in the 10-task ImageNet-R setting (our main experiment setting). As we change the number of tasks in ImageNet-R, the number of total parameters for DualPrompt changes because the pool size is set as equal to the number of total tasks by definition (unlike ours, which is set as a hyper-parameter, allowing us to increase or decrease the number of trainable parameters to accommodate the underlying complexity of the task.)

C. ImageNet-R Dual-Shift Benchmark

Our motivation for the challenging *dual-shift* ImageNet-R [3, 9] benchmark is to show robustness to two different types of continual distribution shifts: semantic *and* covariate. Specifically, there are 15 image types in the ImageNet-R dataset: *'art', 'cartoon', 'deviantart', 'embroidery', 'graffiti', 'graphic', 'misc', 'origami', 'painting', 'sculpture', 'sketch', 'sticker', 'tattoo', 'toy', 'videogame'*. We divide the dataset into 5 tasks of 40 classes each, and within each task we randomly remove 8 of the domain types from the training data. The task becomes more challenging because we now have image type domain shifts injected into the continual learning task sequence, in addition to the already-present class-incremental shifts.

References

[1] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 844–853, October 2021. 1

[2] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-GEM. In *International Conference on Learning Representations*, 2019. 1

[3] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021. 1, 4

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[5] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Tech Report*, 2009. 1

[6] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6470–6479, USA, 2017. Curran Associates Inc. 1

[7] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022. 1

[8] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019. 1

[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. *arXiv preprint arXiv:2204.04799*, 2022. 1, 4

[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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149, 2022. 1

[11] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 1