Continual Learning with Low Rank Adaptation

Martin Wistuba
Amazon Web Services

Prabhu Teja S
Amazon Web Services

Lukas Balles
Amazon Web Services

Giovanni Zappella
Amazon Web Services

Abstract

Recent work using pretrained transformers has shown impressive performance when fine-tuned with data from the downstream problem of interest. However, they struggle to retain that performance when the data characteristics changes. In this paper, we focus on continual learning, where a pre-trained transformer is updated to perform well on new data, while retaining its performance on data it was previously trained on. Earlier works have tackled this primarily through methods inspired from prompt tuning. We question this choice, and investigate the applicability of Low Rank Adaptation (LoRA) to continual learning. On a range of domain-incremental learning benchmarks, our LoRA-based solution, CoLoR, yields state-of-the-art performance, while still being as parameter efficient as the prompt tuning based methods.

1 Introduction

A primal feature of human cognitive abilities is to incrementally and continually update knowledge of a problem; a child can seamlessly learn to recognize newer breeds of dogs without forgetting previously learned ones. Modern machine learning systems, however, fail at this. When naïve methods for fine-tuning are used to update the weights, they perform well on the specific dataset it has been fine-tuned on, while losing performance on previous ones, a phenomenon called *catastrophic forgetting* [9, 20]. This issue, while not as drastic for modern pre-trained transformers [24], is still a major hindrance to the deployment of reliable systems. Continual learning [4, 21] deals with this problem of periodically updating a model with new data, while avoiding forgetting previous information.

In practice, data arrives as a sequence of datasets and we aim at performing well on the latest dataset while retaining performance on the previous ones. Several paradigms of continual learning are defined based on the differences between each dataset. In domain-incremental learning (DIL), the set of labels is fixed, whereas the data distribution can change arbitrarily. In class-incremental learning (CIL), the set of labels is growing with new datasets which poses the challenge of recognizing newly introduced classes. In task-incremental learning (TIL), we learn to solve different tasks and the number of tasks grows incrementally. At training and prediction time, we are aware of the task identity which is not the case in the other settings.

With transformer-based models becoming commonplace, several continual learning methods have been proposed that use specific architectural components of those models. These methods are heavily inspired by the parameter-efficient fine-tuning methods in NLP [27], primarily, prompt tuning [15]. Prompt tuning prepends a set of learnable parameters to the outputs of the input embedding layer and trains only those, while keeping the rest of the model frozen. Learning to Prompt (L2P) [37]

Workshop on Distribution Shifts, 37th Conference on Neural Information Processing Systems (NeurIPS 2023).

trains a set of input-dependent prompts that are shared across datasets, which encourages transfer. S-Prompts [30] instead learns a single prompt per dataset, and proposes a method to determine which prompt to use at inference. We discuss several other works in Appendix A. However, the choice of using prompt tuning is not justified sufficiently in these methods beyond parameter-efficiency, despite prior work [12, 29] demonstrating prompt tuning is slower to train and achieves lower test time performance than the full fine-tuning counterpart.

In this work we revisit this choice, in light of evidence from the NLP community that shows low rank update methods [12] perform better than prompt-based ones. We propose an adaptation of S-Prompts, the state-of-the-art for domain-incremental learning, called CoLoR for efficient continual training of vision transformers showing a significant improvement in predictive performance. With an empirical evaluation on three domain-incremental benchmarks, we show that CoLoR outperforms prompt-based methods such as L2P and S-Prompts in terms of average accuracy and forgetting. Furthermore, we show that these gains are achieved with approximately the same number of model parameters. We propose a simple extension to our method called CoLoR++ that yields state-of-the-art results on Split CIFAR-100.

2 Continual Low Rank Adaptation

We, discuss Low Rank Adaptation (LoRA), and then present our method Continual Low Rank Adaptation (CoLoR).

2.1 Low Rank Adaptation

We focus on vision transformers in this work, but this approach is sufficiently general to be used with other pre-trained transformers. A detailed description of vision transformers is provided in Appendix B. Traditional fine-tuning updates all the weights of a pre-trained transformer with the data of a downstream task. Low Rank Adaptation [12] constraints the update to a low rank one. An update to a parameter matrix $\boldsymbol{W} \in \mathbb{R}^{d\times k}$ of the form $\boldsymbol{W} \leftarrow \boldsymbol{W} + \Delta\boldsymbol{W}$ is constrained by parameterizing $\Delta\boldsymbol{W} = \boldsymbol{B}\boldsymbol{A}$ where $\boldsymbol{A} \in \mathbb{R}^{r\times k}$ and $\boldsymbol{B} \in \mathbb{R}^{d\times r}$. This restricts $\Delta\boldsymbol{W}$ to a rank r, and is also parameter-efficient; when $r \ll k$, the total number of parameters that are updated is $r(d + k)$ instead of kd as is in the case of full fine-tuning. In addition, LoRA is applied only to query and value embedding matrices ($\boldsymbol{W}_Q$ and $\boldsymbol{W}_V$) in all the layers of the network, thereby further reducing the number of trainable parameters compared to full fine-tuning. At inference, the added parameters can be merged with the old parameters, keeping the inference time unaffected.

2.2 CoLoR – Training and Inference

Training CoLoR leverages a pretrained model h and extends it using LoRA to train an expert model for each dataset D. Let us denote the expert model for dataset D with $f_D(\mathbf{x}) = g_D \circ h(\mathbf{x}; \Theta_D)$ where the parameters of h are frozen but it is extended by dataset-specific LoRA modules parameterized by Θ_D. g_D refers to the dataset-specific classification layer $g_D(\mathbf{x}) = \text{softmax}(\mathbf{w}_D^\intercal h(\mathbf{x}; \Theta_D) + b_D)$ which uses the [CLS] token of the vision transformer. The trainable parameters of the network are $\Theta_{D,l} = \{\boldsymbol{A}_Q^{t,l}, \boldsymbol{B}_Q^{D,l}, \boldsymbol{A}_V^{D,l}, \boldsymbol{B}_V^{D,l}\}$ corresponding to all LoRA components added to each layer l, and the parameters of the classifier $\mathbf{w}_D, b_D$. The overall network is trained with a loss appropriate for the downstream problem.

Inference As the dataset identifier D is not available at inference time, we use a simple unsupervised method [30] to infer it. We estimate k dataset prototype vectors for each dataset D at training time as follows. First, we embed each training instance using h (without LoRA modules), and run k-means on those feature embeddings. We store the k cluster centers which serve as representatives for dataset D. At inference time for an instance $\mathbf{x}$, we estimate the cluster center which is nearest to $h(\mathbf{x})$. Then, we use $f_{\hat{D}}$ to make the prediction for $\mathbf{x}$, where $\hat{D}$ is the dataset corresponding to the nearest cluster center.

3 Experiments

Experimental setup Our experiments closely mirror those of [30]. For domain incremental learning experiments, we show results on CORe50 [18] and DomainNet [22]. CORe50 is a benchmark for continual object recognition with 50 classes from 11 datasets with 8 of them acting as the training set, and the rest as the test set. DomainNet is a benchmark for image classification with 345 classes and 6 datasets. For class incremental experiments, we use Split CIFAR-100 [36] which splits the CIFAR-100 into 10 datasets of 10 contiguous classes each.

To facilitate a fair comparison of baselines, we use a ViT-B-16 model [6] pretrained on ImageNet21k from the timm library [33], and report average accuracy, *i.e.*, the fraction of correctly classified test instances up to the current dataset. Our code base is built on top of S-Prompts [30].

We provide a summary of our results here, and present detailed tables in Appendix D (Tables 2 to 4). We, primarily, focus on memory-free methods here and relegate a broader comparison with replay-based methods to the Appendix.

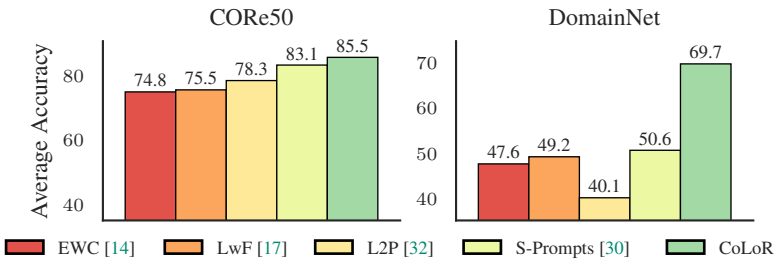

Figure 1: Results on two different datasets for domain-incremental learning. CoLoR improves by 2%-19% over the next best memory-free method.

CoLoR demonstrates new state-of-the-art results in domain-incremental learning. In Figure 1, CoLoR demonstrates superior performance compared to all other methods. It outperforms its closest competitor by 2% on CORe50, and 19% on DomainNet. Furthermore, CoLoR performs on par or better than replay-based methods (Appendix, Table 3).

LoRA is beneficial in class-incremental learning. Results on Split CIFAR-100 support our argument that LoRA is a better choice than prompt tuning, as CoLoR yields better results than S-Prompts (Figure 2). However, CoLoR lags behind L2P due to the quality of representations extracted by ViT ($h(\cdot)$) for the dataset identification method. To address this shortcoming, we propose the CoLoR++, which uses the representation extracted by the network after the first dataset update, *i.e.*, $h(x, \Theta_1)$. We believe that this feature extractor effectively represents the data as it has been trained on a portion of it, leading to improved results. A comparable enhancement is also noticed in domain-incremental learning, albeit to a lesser extent (Appendix, Table 3).

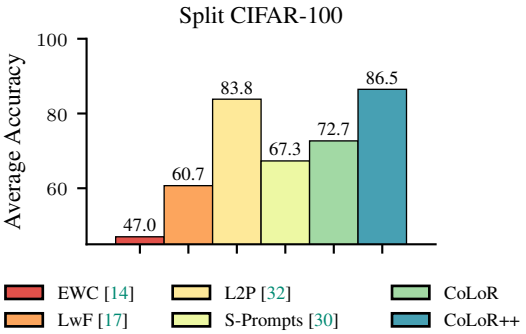

Figure 2: CoLoR improves by more than 5% on CIFAR-100 in the class-incremental scenario over S-Prompts.

CoLoR retains the parameter-efficiency of S-Prompts Table 2 summarizes the additional parameters required for CoLoR and its prompt-tuning competitors on a hypothetical two class problem. Since this efficiency holds only true for low ranks r, we report the additional accuracy results in Figure 3 and Tables 3 and 4 in the Appendix.

Table 1: Number of trainable parameters for each method. We report the parameters trained for DyTox, L2P, S-Prompts, and CoLoR ($r = 1$) for a hypothetical two class problem. † numbers are reproduced from [30].

	DyTox†	L2P†	S-Prompts†	CoLoR
Additional Parameters per Dataset (on average)	1.42M 1.65% ↑	18.43K 0.02% ↑	52.22K 0.06% ↑	38.40K 0.04% ↑

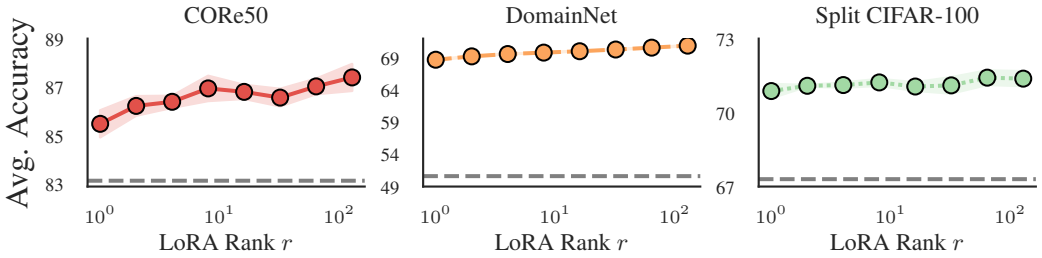

Figure 3: Increasing the rank by keeping all other settings fixed. Increasing the rank beyond 2-digit numbers yields only minor improvements in most cases. CoLoR outperforms its best competitor even with the smallest rank.

It is apparent, that for the same number of parameters, CoLoR still provides better results than its competitors. Furthermore, increasing the rank allows to trade parameter-efficiency for prediction performance.

CoLoR closes the gap between DIL and TIL. In previous experiments, we assume no access to the dataset identifier at inference, and use our dataset identification method to determine which LoRA module to use. In Table 2, we show the results for using an oracle dataset identification method. A substantial increase in accuracy is expected as the dataset identification is non-trivial; in particular, in CIL a wrongful dataset prediction leads to a mis-classification. However, for DIL this happens to a lesser degree and CoLoR closes the gap between TIL and DIL. Finally, TIL performance can be construed to be the upper bound of using LoRA-based modules for continual learning. Importantly, this upper bound is significantly higher than the one oftentimes attained by training a single model using all data (see Appendix, Table 4).

Table 2: Inferred dataset id vs known dataset id with CoLoR. We report the performance in the case where the dataset id is inferred as explained above and in the case there the correct dataset id is provided by an oracle. While the oracle-based setting is not realistic, this comparison is still useful to investigate the performance of the algorithm. This experiment is not applicable for CORe50.

	DomainNet	Split CIFAR-100
CoLoR (inferred dataset id)	69.67	71.42
CoLoR (correct dataset id)	73.68	98.67

4 Conclusions

In this work, we scrutinized the omnipresence of prompt tuning in recent continual learning methods in favor of other parameter-efficient fine-tuning (PEFT) methods. We did this by introducing CoLoR, a LoRA-based continual learning method. We empirically demonstrated that it outperforms its prompt tuning counterpart in domain- and class-incremental learning by a large margin and remains as parameter-efficient. Furthermore, we improved the unsupervised dataset identification strategy by using the representation of the fine-tuned model. This change resulted in new state-of-the-art results on Split CIFAR-100.

References

[1] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *NeurIPS*, 2020. 8, 9, 10

[2] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *ICCV*, 2021. 9, 10

[3] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019. 8, 9, 10

[4] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021. 1

[5] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *CVPR*, pages 5138–5146. Computer Vision Foundation / IEEE, 2019. 8

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3, 8

[7] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *CVPR*, 2022. 8, 9

[8] Beyza Ermis, Giovanni Zappella, Martin Wistuba, Aditya Rawal, and Cédric Archambeau. Memory efficient continual learning with transformers. In *NeurIPS*, 2022. 8

[9] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999. 1

[10] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, 2019. 8

[11] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR, 2019. 8

[12] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 2

[13] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. In *ICLR*, 2018. 8

[14] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. In *Proceedings of the national academy of sciences*, 2017. 3, 8, 9, 10

[15] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. 1, 8

[16] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics. 8

[17] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 3, 8, 9, 10

[18] Vincenzo Lomonaco and Davide Maltoni. CORe50: a new dataset and benchmark for continuous object recognition. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 17–26. PMLR, 13–15 Nov 2017. 3

[19] Francesco Marra, Cristiano Saltori, Giulia Boato, and Luisa Verdoliva. Incremental learning for the detection and classification of gan-generated images. In *WIFS*, 2019. 8

[20] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 1

[21] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019. 1

[22] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *ICCV*, 2019. 3

[23] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. GDumb: A simple approach that questions our progress in continual learning. In *ECCV*, 2020. 8, 9, 10

[24] Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. Effect of scale on catastrophic forgetting in neural networks. In *International Conference on Learning Representations*, 2022. 1

[25] Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models. In *ICLR*. OpenReview.net, 2023. 8

[26] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured laplace approximations for overcoming catastrophic forgetting. In *NeurIPS*, pages 3742–3752, 2018. 8

[27] Sebastian Ruder, Jonas Pfeiffer, and Ivan Vulić. Modular and parameter-efficient fine-tuning for NLP models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 23–29, Abu Dubai, UAE, December 2022. Association for Computational Linguistics. 1

[28] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogério Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *CVPR*, pages 11909–11919. IEEE, 2023. 8

[29] Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3949–3969, Seattle, United States, July 2022. Association for Computational Linguistics. 2

[30] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. In *NeurIPS*, 2022. 2, 3, 4, 8, 9, 10

[31] 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. pages 631–648, 2022. 8

[32] 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 *CVPR*, 2022. 3, 9, 10

[33] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 3

[34] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, 2019. 8, 9, 10

[35] Fei Ye and Adrian G. Bors. Learning latent representations across multiple data domains using lifelong VAEGAN. In *ECCV (20)*, volume 12365 of *Lecture Notes in Computer Science*, pages 777–795. Springer, 2020. 8

[36] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, 2017. 3

[37] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*, 2021. 1, 8

Appendix A Related Work

Continual learning methods can be broadly classified based on how they retain the information learned in previous datasets. *Replay-based* methods tackle catastrophic forgetting by using some additional data which is used when training on the new data [1–3, 10, 19, 23, 34]. These methods store a few data points from previous datasets in a memory of limited size and replay those data points during training. Memory-free approaches replace true data points with generated or auxiliary data, which is replayed [13, 35].

Regularization-based methods oftentimes require no memory and avoid forgetting by adding regularization terms to the loss function. These terms can either regularize the weights directly to avoid changing important weights [14, 26] or regularizing activation outputs [5, 17].

With the advent of large scale pre-trained transformers, memory-free continual learning based on prompt-tuning [15] for domain-incremental or class-incremental learning, or adapters [11] for task-incremental learning [8] have been proposed recently. Learning To Prompt (L2P) [37], based on prompt-tuning, learns a set of input-dependent prompts that are shared across datasets. DualPrompts [31] extends this by learning adding dataset-dependent and dataset-independent prompts at various points in the network. In addition to this idea, follow-up work proposes to learn components which are combined to prompts at inference time [28]. Works that simplify the problem by learning a per-dataset prompt that are combined for efficient forward transfer exist. However, this requires to assume a task-incremental setting where old prompts are not further updated [25] or access to old data [7]. S-Prompts overcomes this problem by training assuming a task-incremental setting and then solving the task identification problem at inference time using clustering [30]. The work discussed here for continual learning for transformers relies on variations of prompt-tuning or prefix-tuning [16]. Additionally, S-Prompts is primarily shown to work for domain-incremental scenarios. Our method, CoLoR, extends this line of work by using LoRA modules, retains the simplicity of S-Prompts, and is effective at both domain incremental and task incremental learning scenarios.

Appendix B Vision Transformer

In this section, we describe the Vision Transformer [6] (ViT) that we use in this paper. ViT ingests an image $I \in \mathbb{R}^{W\times H\times 3}$, and first extracts patches of size $P \times P$, totalling $\frac{W\times H}{P^2}$ patches per image. Each of these patches is flattened and embedded into a D dimensional space. To this a learned position encoding (E_{pos}) is added, and a special token called the classification ([CLS]) token is concatenated. We refer to this as $X_0 \in \mathbb{R}^{N\times D}$ where $N = \frac{W\times H}{P^2} + 1$. This operation can be represented as

$$X_0 = [\text{[CLS]}; I_p^1 E; \cdots I_p^{(N-1)} E] + E_{\text{pos}}. \tag{1}$$

This feature representation is processed through L layers of multi-head self attention layers.

$$\left.\begin{array}{ll} X_l^a & = \text{MHSA}(X_{l-1}) + X_{l-1} \\ X_l & = \text{FFN}(X_l^a) + X_l^a \end{array}\right\} \forall l = 1 \ldots L$$

The function MHSA consists of mutiple SA modules that function in parallel. Each SA module can be written as

$$\text{SA}(X_l) = \text{softmax}\left(\frac{X_l W_Q^l {W_K^l}^T X_l^T}{2\sqrt{d}}\right) X_l \boldsymbol{W}_V \tag{2}$$

and the FFN as

$$\text{FFN}(X_l) = \text{GeLU}(W_2^l \text{GeLU}(W_1^l X_l + b_1^l) + b_2^l). \tag{3}$$

The [CLS] token at X_L is fed into a linear layer $\mathbb{R}^D \to \mathbb{R}^C$ that outputs the logits for classification. The set of trainable parameters for fine-tuning is $\{W_*^l, b_*^l\}_{l=1}^L$.

Appendix C Training hyperparameters

We closely follow the protocol by earlier work to allow for fair comparison [30]. We adopt their data augmentation which consists of simple horizontal flips and random crops. We use a batch size of 128 and a weight decay of 0.0002. We set learning rates and epochs to minimize training budget. In most cases, we use 50 epochs with the exception of CORe50 where we use 20. As a default, we use a learning rate of 10^{-3}. For CIFAR-100, we use 0.01, for CORe50, 0.02. Cosine annealing is used to decay the learning rate over time. Unless otherwise stated, we use a LoRA rank of 64. We set the number of clusters to $k = 5$ as recommended for S-Prompts [30] in DIL. For CIL, we set the number of clusters to two times the number of new classes, *i.e.*, 20 for Split CIFAR-100. The choice of number of clusters and the rank is ablated in §3 and Appendix E.

Appendix D Results

In this section, we extend the results in Figures 1 and 2 by comparing CoLoR to replay-based methods in Tables 3 and 4.

For the domain incremental scenario presented in Table 3, we observe that CoLoR outperforms replay method with limited buffer sizes on most datasets. On DomainNet, performance of CoLoR is only matched by that of DyTox which uses a replay buffer.

In Table 4, we present detailed results for Split CIFAR-100. For fine-tuning, we fine-tune the entire ViT model and mask the outputs for classes not present in an update by setting those logits to $-\infty$. We find that this is important for L2P, without which its performance suffers drastically. Using "class-masking", fine-tuning results in Table 4 are substantially higher than the ones reported in literature as FT-seq and FT-seq-frozen. Furthermore, we report the results obtained when training the ViT on all data using LoRA, and fine-tuning the entire model as the upper bound.

Table 3: Average accuracy results on three domain-incremental benchmarks. CoLoR consistently outperforms alternative approaches even if these have access to previous data. This includes the self-reported upper bound for S-Prompts which has access to all data. Results marked with † from [32], and with ‡ from [30].

Method	Buffer Size	CORe50	DomainNet
S-Prompts (upper bound)	∞	84.01‡	63.22‡
LoRA ($r = 64$)	∞	96.15±0.07	73.62±0.02
DyTox [7]	50/class	79.21‡±0.10	62.94‡
ER [3]	50/class	80.10†±0.56	-
GDumb [23]	50/class	74.92†±0.25	-
BiC [34]	50/class	79.28†±0.30	-
DER++ [1]	50/class	79.70†±0.44	-
Co^{2}L [2]	50/class	79.75†±0.84	-
L2P [32]	50/class	81.07†±0.13	-
EWC [14]	0	74.82†±0.60	47.62‡
LwF [17]	0	75.45†±0.40	49.19‡
L2P [32]	0	78.33†±0.06	40.15‡
S-Prompts [30] ($k = 5$)	0	83.13‡±0.51	50.62‡
CoLoR ($r = 1,\ k = 5$)	0	84.88±0.10	67.71±0.08
CoLoR ($r = 8,\ k = 5$)	0	85.72±0.48	68.87±0.04
CoLoR ($r = 64,\ k = 5$)	0	85.52±0.42	69.67±0.04
CoLoR++ ($r = 64,\ k = 5$)	0	**86.75**±0.40	**70.06**±0.05

Table 4: Class-incremental learning on CIFAR-100. CoLoR outperforms S-Prompts and CoLoR++ all other continual learning methods. This includes the self-reported upper bound for L2P. Results marked with * are taken from [32]. We report new results for training on all data using LoRA and full fine-tuning.

Method	**Buffer size**	**Split CIFAR-100**	
		Average Acc (↑)	Forgetting (↓)
L2P (upper bound)	∞	90.85*±0.12	N/A
LoRA ($r = 64$)		92.49±0.07	N/A
Fine-Tuning		92.11±0.10	N/A
ER [3]	50/class	82.53*±0.17	16.46*±0.25
GDumb [23]		81.67*±0.02	N/A
BiC [34]		81.4*2±0.85	17.31*±1.02
DER++ [1]		83.94*±0.34	14.55*±0.73
Co^{2}L [2]		82.49*±0.89	17.48*±1.80
L2P-R [32]		86.31*±0.59	**5.83***±0.61
ER [3]	10/class	67.87*±0.57	33.33*±1.28
GDumb [23]		67.14*±0.37	N/A
BiC [34]		66.11*±1.76	35.24*±1.64
DER++ [1]		61.06*±0.87	39.87*±0.99
Co^{2}L [2]		72.15*±1.32	28.55*±1.56
L2P-R [32]		84.21*±0.53	7.72*±0.77
FT-seq-frozen	0	17.72*±0.34	59.09*±0.25
FT-seq		33.61*±0.85	86.87*±0.20
FT+class masking		67.02±4.20	24.37±3.76
EWC [14]		47.01*±0.29	33.27*±1.17
LwF [17]		60.69*±0.63	27.77*±2.17
L2P [32]		83.83*±0.04	7.63*±0.30
S-Prompts [30] ($k = 5$)		57.17±1.57	19.56±0.86
S-Prompts [30] ($k = 10$)		65.71±1.50	14.76±0.75
S-Prompts [30] ($k = 20$)		67.31±1.34	12.47±1.49
CoLoR ($r = 64,\ k = 5$)		59.98±0.04	18.69±0.41
CoLoR ($r = 64,\ k = 10$)		68.51±0.23	10.65±0.04
CoLoR ($r = 1,\ k = 20$)		70.87±0.23	10.16±0.19
CoLoR ($r = 8,\ k = 20$)		71.22±0.11	10.22±0.18
CoLoR ($r = 64,\ k = 20$)		71.42±0.24	10.27±0.39
CoLoR++ ($r = 1,\ k = 20$)		85.27±0.24	6.55±0.46
CoLoR++ ($r = 64,\ k = 20$)		**86.47**±0.07	6.25±0.34

Appendix E Ablations

In this section, we study the effect of the number of clusters k on the average accuracy. We vary k by fixing all other hyperparameters to the defaults described in Appendix C.

In Figure 4, we observe a similar behavior as that of increasing rank in Figure 3 for the number of clusters: more yields better results for CIL, where choosing a large enough number of clusters results in a substantial increase in performance. The advantages of increasing k further diminish very quickly. This is not surprising given that in this scenario, the clusters represent individual classes. Therefore, if k is smaller than the number of classes in an update (in this case 10), the centroids are not able to represent the dataset sufficiently causing dataset detection failures. This is clearly demonstrated by the saturation that we achieve once k reaches the number of new classes. We find that the choice of k is not too sensitive; above a certain small threshold, its choice has relatively little influence on the results. Optimizing it is relatively cheap as it does not require retraining the model.

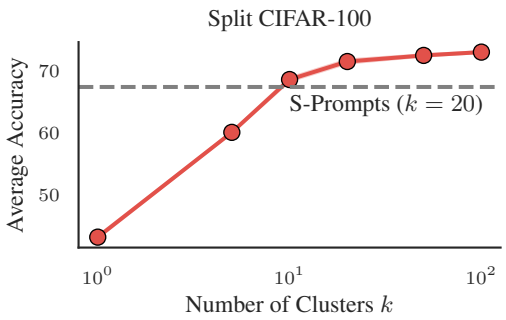

Figure 4: Increasing the number of clusters without changing any other setting. This significantly improves the performance in CIL (Split CIFAR-100) until k equals the number of new classes.