Online-LoRA: Task-free Online Continual Learning via Low Rank Adaptation

Xiwen Wei
The University of Texas at Austin
xiwenwei@utexas.edu

Guihong Li
AMD
liguihong1995@gmail.com

Radu Marculescu
The University of Texas at Austin
radum@utexas.edu

Abstract

Catastrophic forgetting is a significant challenge in online continual learning (OCL), especially for non-stationary data streams that do not have well-defined task boundaries. This challenge is exacerbated by the memory constraints and privacy concerns inherent in rehearsal buffers. To tackle catastrophic forgetting, in this paper, we introduce *Online-LoRA*, a novel framework for task-free OCL. Online-LoRA allows to finetune pre-trained Vision Transformer (ViT) models in real-time to address the limitations of rehearsal buffers and leverage pre-trained models' performance benefits. As the main contribution, our approach features a novel online weight regularization strategy to identify and consolidate important model parameters. Moreover, Online-LoRA leverages the training dynamics of loss values to enable the automatic recognition of the data distribution shifts. Extensive experiments across many task-free OCL scenarios and benchmark datasets demonstrate that Online-LoRA can be robustly adapted to various ViT architectures, while achieving better performance compared to SOTA methods.

1 Introduction

Continual learning (CL) enables machine learning systems to learn new concepts while retaining previously learned knowledge, a crucial requirement for real-time applications such as robotics, healthcare, and autonomous driving (50; 27). However, a significant challenge in CL is *catastrophic forgetting*, where new learning disrupts previously acquired knowledge. Existing CL methods are categorized as either task-based or task-free, depending on the knowledge of task boundaries, and as either online or offline, depending on whether data is processed in a single pass or multiple iterations (44; 59; 19; 5; 53; 17). Offline task-based CL assumes stationary data distributions and known task boundaries, but these assumptions rarely align with real-world scenarios where data flows continuously and lacks clear task boundaries (29; 13; 58; 37; 3).

To address these limitations, we focus on *task-free online CL (task-free OCL)*, where data is seen only once, and task identities and boundaries are unknown during training and inference (28; 11; 20; 57). Pre-trained Vision Transformers (ViTs) have shown strong performance across various tasks and offer robust transfer learning capabilities, particularly in data-scarce environments (12; 60; 54; 15; 16; 61). Recent methods using parameter-efficient fine-tuning (PEFT), such as prompt tuning and Low-Rank Adaptation (LoRA), have shown promise for offline task-based CL (30; 23; 10; 55; 48), but their potential in task-free OCL remains unexplored.

We propose **Online-LoRA**, a novel approach that integrates pre-trained ViTs and LoRA for task-free OCL. Online-LoRA detects shifts in data distribution by monitoring loss plateaus and introduces new LoRA parameters to adapt incrementally. Additionally, we introduce an online regularization mechanism that mitigates forgetting by dynamically estimating the importance of LoRA parameters with minimal memory overhead, reducing computational demands compared to EWC++ (6).

We summarize our main contributions as follows:

1. We propose Online-LoRA, an innovative approach that enables efficient learning from changing data in an online, task-free manner by detecting data distribution shifts through loss plateaus and incorporating an online weight regularization mechanism.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

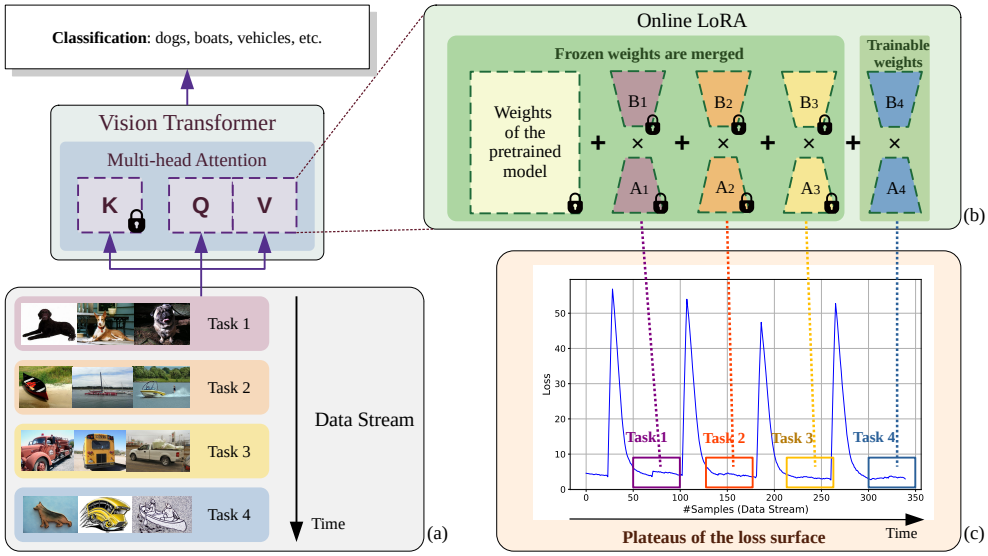


Figure 1: The overview of Online-LoRA. As the data is continuously streamed (a), a new pair of trainable LoRA parameters (A_4, B_4) is added (b) every time the loss surface encounters a plateau (c). Subsequently, the previous LoRA parameters $(A_1, B_1; A_2, B_2; A_3, B_3)$ are frozen (the lock sign in (b)) and merged to the weights of the pre-trained ViT model.

2. Our evaluations across various ViT architectures and multiple task-free OCL benchmarks, including class and domain incremental learning settings, show that Online-LoRA consistently outperforms existing SOTA methods and maintains robust performance across diverse scenarios.

2 Related work

2.1 Continual learning

Many existing CL methods, including architecture-based, regularization-based, rehearsal-based, and prompt-based approaches, face challenges when adapted to task-free OCL. While some methods, like rehearsal-based approaches (8; 2; 42) and instance-wise prompt-based methods such as L2P (54) and MVP (40), can be transferred to task-free OCL, others, like architecture-based methods (46; 45; 14; 43) and certain regularization-based methods (24; 58), require task identity and are not suitable. Although L2P (54) employs a trick that masks out irrelevant classes during training, which contradicts the task-free OCL setting), we ensure a fair comparison by evaluating against L2P (54) and MVP (40) under the Stochastic Incremental Blurry (Si-blurry) scenario proposed by (40), where task boundaries are not explicitly defined.

2.2 Parameter efficient fine-tuning

PEFT is a transfer learning approach that fine-tunes specific sub-modules within a pre-trained network, reducing computation while maintaining performance comparable to full fine-tuning (22). LoRA (23), a notable example, has been used in CL but is limited to task-based offline settings due to its reliance on explicit task boundaries (9; 31; 47; 33). To our knowledge, Online-LoRA is the first to extend LoRA to a task-free OCL scenario for transformer-based vision models.

3 Online-LoRA

3.1 Problem formulation

We define a data stream of unknown distributions $D = \{D_1, ..., D_N\}$ over $X \times Y$, where X and Y are input and output space respectively (38). At each time step s, the system receives a batch of non i.i.d samples x_k^t, y_k^t from the current distribution D_t of task t; the system sees this batch only once. Moreover, at any moment s, the distribution D_t can itself experience sudden or gradual changes from D_t to D_{t+1}. The system is unaware of when and how these distribution changes happen.

For simplicity, we assume that D_t is the data distribution of task t, and any shift from D_t to D_{t+1} is sudden. Of note, this remains a task-free setting, since gradual transitions from D_t to D_{t+1} can still be modeled by adding intermediate tasks and making these distributions increasingly similar, thus

effectively blurring the explicit boundaries between tasks. Our Online-LoRA does not assume any task boundaries at any time.

3.2 Loss-guided model adaptation

In LoRA, for a pre-trained weight matrix $W_{init} \in R^{d\times k}$, the update ΔW in $W \leftarrow W_{init} + \Delta W$ is reformulated as a low-rank decomposition: $\Delta W = BA$, where $A \in R^{r\times k}$ and $B \in R^{d\times r}$, and the rank $r \ll min(d, k)$. W_{init} remains fixed during training and does not receive gradient updates, while A and B contain trainable parameters. In existing LoRA-based CL methods (47; 55), new LoRA parameters are added for each new task t', resulting in a set of LoRA parameters denoted as $\{A_{t'}, B_{t'}\}$, where $A_{t'} \in R^{d\times r}$, $B_{t'} \in R^{r\times k}$, d and k are the input and output dimensions of the attention layer, and rank $r \ll min(d, k)$. When learning task t, if the initial ViT weights are denoted as W_{init}, then for an input sample X, the model output Y becomes:

$$Y = (W_{init} + \sum_{t'=0}^{t} B_{t'} A_{t'})X \tag{1}$$

This *incremental model* mitigates catastrophic forgetting by minimizing interference between old and new tasks (see Figure 1) and applies LoRA only to the query and value projection matrices in attention layers. As data from previous tasks is unavailable when training on future tasks, old LoRA parameters are frozen and merged with pre-trained weights to reduce memory overhead. However, existing LoRA-based methods require explicit task boundaries to initialize new parameters, which is not feasible in task-free OCL with continuous data flow.

To address this, we use the concept of the *loss surface* (3). A decreasing loss indicates effective learning, while an increasing loss suggests a shift in data distribution. We assume convergence before shifts, with *loss plateaus* indicating stable phases (see Appendix D). At these plateaus, we consolidate learning by freezing current LoRA weights and initializing new ones, merging frozen weights with pre-trained attention weights to prevent parameter accumulation.

3.3 Online parameter importance estimation

Many studies have shown the effectiveness of weight regularization in reducing catastrophic forgetting by estimating parameter importance (1; 24; 6). However, in an online scenario with shifting data distributions, parameter importance varies dynamically, making static estimation unsuitable. Additionally, with pre-trained ViT models, traditional methods like calculating the Fisher information matrix for importance estimation (24) are computationally inefficient.

We draw on Bayesian principles similar to EWC (24), treating model parameters as random variables and updating prior knowledge using Bayes' rule. Given data D:

$$\log p(\theta|D) = \log p(D|\theta) + \log p(\theta) - \log p(D) \tag{2}$$

Splitting D into current sample x and past data D_{prev}, this becomes:

$$\log p(\theta|D) = \log p(x|\theta) + \log p(\theta|D_{prev}) - \log p(x) \tag{3}$$

We approximate the posterior as a Gaussian centered at the MAP solution θ_{MAP} using a Laplace approximation, with the empirical Fisher information matrix for covariance. Treating the LoRA adapter $\sum_{t'} B_{t'} A_{t'} X$ as two separate linear layers (56), we use two smaller importance weight matrices, $\Omega^{A,l} \in \mathbb{R}^{d\times r}$ and $\Omega^{B,l} \in \mathbb{R}^{r\times k}$, for each attention layer, allowing efficient online updates (see Appendix L.1 for details).

To estimate parameter importance efficiently, we focus on the sensitivity of loss relative to LoRA parameters. The *importance weight matrices* match the dimensions of LoRA parameters:

$$\Omega^{A,l}_{ij} = \frac{1}{N}\sum_{k=1}^{N} \nabla_{W^{A,l}_{ij}} \log p(x_k|\theta) \circ \nabla_{W^{A,l}_{ij}} \log p(x_k|\theta) \tag{4}$$

$$\Omega^{B,l}_{ij} = \frac{1}{N}\sum_{k=1}^{N} \nabla_{W^{B,l}_{ij}} \log p(x_k|\theta) \circ \nabla_{W^{B,l}_{ij}} \log p(x_k|\theta) \tag{5}$$

Here, $W^{A,l}$ and $W^{B,l}$ are new trainable LoRA parameters added to the l^{th} attention layer, and x_k are samples from a small *hard buffer* containing high-loss samples. After updating the importance

		Split-CIFAR-100		Split-ImageNet-R		Split-ImageNet-S		Split-CUB-200	
		A_{Final} (↑)	*Forgetting* (↓)	A_{Final} (↑)	*Forgetting* (↓)	A_{Final} (↑)	*Forgetting* (↓)	A_{Final} (↑)	*Forgetting* (↓)
ViT-B/16	AGEM (7)	12.67±1.87	82.51±2.27	5.60±2.74	53.97±1.97	0.16±0.04	9.42±0.17	10.84±1.57	47.79±0.04
	ER (8)	44.85±1.83	44.67±4.29	40.99±3.96	32.38±0.89	30.21±0.70	37.14±1.83	31.66±0.83	14.23±0.07
	EWC++ (6)	10.61±0.74	84.10±1.11	3.86±2.02	56.95±1.46	0.32±0.28	22.46±4.69	26.14±3.46	47.69±0.07
	MIR (2)	48.36±3.11	43.41±1.02	41.51±2.99	31.32±5.17	30.33±3.81	35.92±1.75	31.64±2.97	23.43±0.05
	GDumb (42)	41.00±19.97	-	8.87±1.36	-	1.65±0.22	-	9.09±1.03	-
	PCR (34)	48.48±0.15	46.23±1.29	46.11±3.03	25.50±0.41	38.75±0.22	35.01±2.12	41.11±1.43	29.64±1.20
	DER++ (4)	36.64±6.11	56.94±7.55	30.90±8.04	24.26±4.14	6.47±0.06	15.34±0.15	26.61±1.27	32.16±0.55
	LODE (DER++) (32)	44.29±1.48	45.54±3.32	42.20±6.46	31.83±1.05	9.97±2.29	**8.48±1.24**	39.20±4.25	41.64±3.59
	EMA (DER++) (49)	42.28±4.36	55.59±1.48	41.75±1.98	32.65±1.55	16.88±2.23	36.28±1.09	35.26±3.31	25.55±3.35
	EMA (RAR) (49)	47.10±0.82	50.01±0.35	30.04±0.33	39.36±0.04	14.06±0.37	36.28±1.09	33.34±1.11	28.68±0.56
	Ours	**49.40±1.36**	**41.74±2.58**	**48.18±0.63**	**23.85±1.48**	**47.06±0.24**	28.09±3.25	**41.46±0.31**	**13.64±0.68**
	UB	89.50±0.04	-	76.78±0.44	-	63.82±0.02	-	82.81±1.07	-
ViT-S/16	AGEM (7)	7.43±2.15	82.45±5.46	2.35±0.87	48.01±0.05	2.75±2.86	**18.81±0.44**	1.40±0.17	27.06±1.39
	ER (8)	31.91±2.06	52.21±6.41	32.73±0.20	45.37±1.72	19.53±1.44	45.10±0.48	21.81±3.02	24.52±2.98
	EWC++ (6)	6.80±2.13	81.59±7.43	1.32±0.83	53.54±0.19	4.08±3.24	21.28±0.46	2.07±0.54	28.44±0.83
	MIR (2)	29.08±1.14	39.42±1.60	**34.73±0.29**	48.66±0.69	13.96±1.95	42.61±0.08	22.95±1.12	32.54±0.88
	GDumb (42)	10.87±4.94	-	5.33±1.09	-	2.09±0.32	-	3.28±0.99	-
	PCR (34)	32.89±1.47	39.90±2.51	21.96±0.27	45.12±0.08	14.37±0.95	43.96±0.48	22.28±2.73	29.87±0.04
	DER++ (4)	17.67±4.04	51.65±3.67	22.17±4.27	54.79±0.89	18.15±0.66	46.22±0.95	29.53±2.37	21.49±1.16
	LODE (DER++) (32)	28.65±3.06	40.42±1.58	31.65±0.72	43.72±0.09	17.59±0.84	47.85±0.23	26.81±0.89	21.86±2.30
	EMA (DER++) (49)	12.76±0.65	41.17±1.75	20.89±3.05	48.03±1.79	12.93±0.13	22.59±0.16	35.79±5.27	24.85±4.20
	EMA (RAR) (49)	19.21±2.16	41.99±1.73	16.11±0.35	50.58±0.83	14.50±2.71	23.79±2.91	34.53±1.04	30.19±0.36
	Ours	**32.16±0.24**	**38.64±0.65**	33.21±0.50	**42.76±0.18**	**22.45±0.43**	44.56±0.24	**37.41±0.16**	**20.78±2.54**
	UB	86.55±0.01	-	69.94±0.34	-	59.28±0.11	-	73.91±1.16	-

Table 1: Results of disjoint class-incremental learning. '↑' means higher is better and '↓' means lower is better. Regularization-based methods (EWC++, AGEM, and LODE) yield low accuracy and low forgetting on Split ImageNet-S. This is because their overly rigid constraints on model updates hinder effective learning. The best results are noted by **bold**. *UB* is the upper-bound performance.

weights, the model penalizes changes to important parameters, with θ_{MAP} being the model weights at the last loss plateau. Our final objective is:

$$\min_{W^A, W^B} \mathcal{L}(F(X;\theta), Y) + \mathcal{L}(F(X_B;\theta), Y_B) + L_{\text{LoRA}}(W^A, W^B) \tag{6}$$

$$L_{\text{LoRA}}(W^A, W^B) = \frac{\lambda}{2} \sum_{l \in |Attn|} ((\Omega^{A,l} \circ (W^{A,l}) \circ (W^{A,l})) + (\Omega^{B,l} \circ (W^{B,l}) \circ (W^{B,l}))) \tag{7}$$

where $Attn$ is the set of attention layers, $\mathcal{L}(F(X;\theta), Y)$ is the loss of current samples, and $\mathcal{L}(F(X_B;\theta), Y_B)$ is the loss of hard buffer samples.

4 Experiments

4.1 Experimental Setup

Benchmarks: Datasets (CIFAR-100, Imagenet-R, Imagenet-S, CUB-200, and CORe50) under three scenarios (disjoint class-incremental, Si-Blurry class-incremental, and domain-incremental). See Appendix B for details. **Baselines**: the Upper-bound (*UB*) baseline refers to supervised fine-tuning on the entire dataset of i.i.d. data, representing the optimal performance. See Appendix J for details of other SOTA methods. **Metrics**: A_{AUC}, A_{Final} (higher values indicate better accuracy), and *Forgetting* (lower values indicate lower forgetting). See Appendix A for the detailed definitions. **Hyper-parameters**: For Online-LoRA, see Appendix C; for baselines, see Appendix J; and for results with other buffer sizes, see Appendix K. **Training epoch**: Given our focus on online CL, the training epoch is set to 1 for all experiments.

4.2 Experimental Results

Table 1 summarizes the results on the disjoint class-incremental benchmarks Split CIFAR-100, Split ImageNet-R, Split ImageNet-S, and Split CUB-200. Our Online-LoRA, outperforms all other compared methods consistently across the ViT-B/16 and ViT-S/16. On Split ImageNet-S, Online-LoRA exhibits standout performance, significantly outperforming all other methods, and notably reducing the gap to the upper bound. As shown in Table 1, Online-LoRA consistently performs well across various dataset sizes, while GDumb (42) struggles with smaller datasets like Split CIFAR-100 and performs poorly on larger datasets like Split-ImageNet-R and Split ImageNet-S. GDumb's reliance on a replay buffer leads to class imbalance with larger datasets due to limited representation. In contrast, Online-LoRA uses a small, targeted 'hard buffer' of high-loss samples, effectively addressing these challenges without depending heavily on memory size.

For results on additional scenarios (Si-Blurry class-incremental in Appendix F and domain-incremental in Appendix E), different task sequence lengths (Appendix H), and the ablation study (Appendix I), please see the Appendix.

5 Conclusion

In this paper, we introduced Online-LoRA, a novel method for task-free online CL that adapts to changing data distributions by dynamically analyzing the loss surface and using online weight regularization to prevent catastrophic forgetting. Our experiments demonstrate that Online-LoRA outperforms state-of-the-art methods, particularly in scenarios with long task sequences, and approaches the upper bound in domain-incremental settings. With the increasing use of pre-trained models in CL, Online-LoRA provides a solid foundation for practical task-free online CL systems.

References

[1] Aljundi, R. et al.: Memory aware synapses: Learning what (not) to forget. In: Proceedings of the European conference on computer vision (ECCV). pp. 139–154 (2018)

[2] Aljundi, R. et al.: Online continual learning with maximal interfered retrieval. Advances in neural information processing systems **32** (2019)

[3] Aljundi, R. et al.: Task-free continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11254–11263 (2019)

[4] Buzzega, P. et al.: Dark experience for general continual learning: a strong, simple baseline. Advances in neural information processing systems **33**, 15920–15930 (2020)

[5] Cai, Z. et al.: Online continual learning with natural distribution shifts: An empirical study with visual data. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 8281–8290 (2021)

[6] Chaudhry, A. et al.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: Proceedings of the European conference on computer vision (ECCV). pp. 532–547 (2018)

[7] Chaudhry, A. et al.: Efficient lifelong learning with a-gem. arXiv preprint arXiv:1812.00420 (2018)

[8] Chaudhry, A. et al.: Continual learning with tiny episodic memories. In: Workshop on Multi-Task and Lifelong Reinforcement Learning (2019)

[9] Chen, S. et al.: Adaptformer: Adapting vision transformers for scalable visual recognition. arXiv preprint arXiv:2205.13535 (2022)

[10] Chitale, R. et al.: Task arithmetic with lora for continual learning. arXiv preprint arXiv:2311.02428 (2023)

[11] De Lange, M. et al.: Continual prototype evolution: Learning online from non-stationary data streams. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8250–8259 (October 2021)

[12] Dosovitskiy, A. et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)

[13] Farquhar, S. et al.: Towards robust evaluations of continual learning. arXiv preprint arXiv:1805.09733 (2018)

[14] Fernando, C. et al.: Pathnet: Evolution channels gradient descent in super neural networks. arXiv preprint arXiv:1701.08734 (2017)

[15] Gao, Q. et al.: A unified continual learning framework with general parameter-efficient tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11483–11493 (2023)

[16] Han, X. et al.: Pre-trained models: Past, present and future. AI Open **2**, 225–250 (2021)

[17] He, J. et al.: Online continual learning for visual food classification. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 2337–2346 (2021)

[18] He, K. et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[19] He, X. et al.: Task agnostic continual learning via meta learning. arXiv preprint arXiv:1906.05201 (2019)

[20] He, Y. et al.: Dyson: Dynamic feature space self-organization for online task-free class incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2024)

[21] Hendrycks, D. et al.: The many faces of robustness: A critical analysis of out-of-distribution generalization. ICCV (2021)

[22] Houlsby, N. et al.: Parameter-efficient transfer learning for nlp. In: International conference on machine learning. pp. 2790–2799. PMLR (2019)

[23] Hu, E.J. et al.: Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)

[24] Kirkpatrick, J. et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)

[25] woo Koh, H. et al.: Online continual learning on class incremental blurry task configuration with anytime inference. ArXiv **abs/2110.10031** (2021), `https://api.semanticscholar.org/CorpusID:239024453`

[26] Krizhevsky, A. et al.: Learning multiple layers of features from tiny images (2009)

[27] Lee, C.S. et al.: Clinical applications of continual learning machine learning. The Lancet Digital Health **2**(6), e279–e281 (2020)

[28] Lee, S. et al.: A neural dirichlet process mixture model for task-free continual learning. arXiv preprint arXiv:2001.00689 (2020)

[29] Lesort, T. et al.: Understanding continual learning settings with data distribution drift analysis. arXiv preprint arXiv:2104.01678 (2021)

[30] Lester, B. et al.: The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691 (2021)

[31] Lian, D. et al.: Scaling & shifting your features: A new baseline for efficient model tuning. In: Advances in Neural Information Processing Systems (NeurIPS) (2022)

[32] Liang, Y.S. et al.: Loss decoupling for task-agnostic continual learning. Advances in Neural Information Processing Systems **36** (2023)

[33] Liang, Y.S. et al.: Inflora: Interference-free low-rank adaptation for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23638–23647 (2024)

[34] Lin, H. et al.: Pcr: Proxy-based contrastive replay for online class-incremental continual learning. In: CVPR. pp. 24246–24255 (2023)

[35] Liu, Z. et al.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV. pp. 10012–10022 (2021)

[36] Lomonaco, V. et al.: Core50: a new dataset and benchmark for continuous object recognition. In: Proceedings of the 1st Annual Conference on Robot Learning. vol. 78, pp. 17–26 (2017)

[37] Lopez-Paz, D. et al.: Gradient episodic memory for continual learning. Advances in neural information processing systems **30** (2017)

[38] Mai, Z. et al.: Online continual learning in image classification: An empirical survey. Neurocomputing **469**, 28–51 (2022)

[39] Microsoft: Deepspeed: A deep learning optimization library. `https://github.com/microsoft/DeepSpeed` (2024), accessed: 2024-09-05

[40] Moon, J.Y. et al.: Online class incremental learning on stochastic blurry task boundary via mask and visual prompt tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2023)

[41] Moon, J.Y. et al.: Online class incremental learning on stochastic blurry task boundary via mask and visual prompt tuning. In: ICCV (2023)

[42] Prabhu, A. et al.: Gdumb: A simple approach that questions our progress in continual learning. In: The European Conference on Computer Vision (ECCV) (August 2020)

[43] Rajasegaran, J. et al.: Random path selection for continual learning. Advances in Neural Information Processing Systems **32** (2019)

[44] Rao, D. et al.: Continual unsupervised representation learning. Advances in neural information processing systems **32** (2019)

[45] Rusu, A.A. et al.: Progressive neural networks. ArXiv **abs/1606.04671** (2016), https://api.semanticscholar.org/CorpusID:15350923

[46] Shanahan, M. et al.: Encoders and ensembles for task-free continual learning. arXiv preprint arXiv:2105.13327 (2021)

[47] Smith, J. et al.: Continual diffusion: Continual customization of text-to-image diffusion with c-lora. ArXiv **abs/2304.06027** (2023), https://api.semanticscholar.org/CorpusID:258078844

[48] Smith, J.S. et al.: Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11909–11919 (2023)

[49] Soutif-Cormerais, A. et al.: Improving online continual learning performance and stability with temporal ensembles. In: Conference on Lifelong Learning Agents. pp. 828–845. PMLR (2023)

[50] Verwimp, E. et al.: Continual learning: Applications and the road forward. arXiv preprint arXiv:2311.11908 (2023)

[51] Wah, C. et al.: Caltech-ucsd birds 200. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)

[52] Wang, H. et al.: Learning robust global representations by penalizing local predictive power. In: Advances in Neural Information Processing Systems. pp. 10506–10518 (2019)

[53] Wang, Z. et al.: Online continual learning with contrastive vision transformer. In: European Conference on Computer Vision. pp. 631–650. Springer (2022)

[54] Wang, Z. et al.: Learning to prompt for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 139–149 (2022)

[55] Wistuba, M. et al.: Continual learning with low rank adaptation. In: NeurIPS 2023 Workshop on Distribution Shifts (DistShifts) (2023), https://www.amazon.science/publications/continual-learning-with-low-rank-adaptation

[56] Yang, A.X. et al.: Bayesian low-rank adaptation for large language models. ArXiv **abs/2308.13111** (2023), https://api.semanticscholar.org/CorpusID:261214713

[57] Ye, F. et al.: Online task-free continual generative and discriminative learning via dynamic cluster memory. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 26202–26212 (2024)

[58] Zenke, F. et al.: Continual learning through synaptic intelligence. In: International conference on machine learning. pp. 3987–3995. PMLR (2017)

[59] Zeno, C. et al.: Task agnostic continual learning using online variational bayes. arXiv preprint arXiv:1803.10123 (2018)

[60] Zhang, G. et al.: Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19148–19158 (2023)

[61] Zhou, D.W. et al.: Deep class-incremental learning: A survey. arXiv preprint arXiv:2302.03648 (2023)

A Evaluation Metrics

In this section, we present the definitions of the three evaluation metrics we used in our experiments, supplementing Section **??** in the main paper.

Let $a_{i,j}$ be the testing accuracy on the i^{th} task after training on j^{th} task. The total number of tasks is denoted by T.

Final Accuracy The final accuracy A_{Final} is calculated as the average accuracy across all tasks after training on the final task:

$$A_{\text{Final}} = \frac{1}{T} \sum_{i=1}^{T} a_{i,T} \tag{8}$$

Area Under the Curve of Accuracy The A_{AUC} (Area Under the Curve of Accuracy) is defined as the area under the curve (AUC) of the accuracy-to-# of samples curve (25). To construct the curve, the accuracy is measure after each sample is observed. A_{AUC} measures the any time inference accuracy of the model:

$$A_{\text{AUC}} = \sum_{i=1}^{k} f(i \cdot \Delta n) \cdot \Delta n, \tag{9}$$

where the step size Δn is defined as $\Delta n = 1$, representing the number of samples observed between inference queries, and $f(\cdot)$ denotes the curve in the accuracy-to-{number of samples} plot. A high A_{AUC} indicates that the method consistently maintains high accuracy throughout training.

Forgetting Forgetting is defined as the averaged differences between the historical maximum accuracy of task k and the accuracy of task k after all tasks finish training:

$$\text{Forgetting} = \frac{1}{T-1} \sum_{k=1}^{T-1} \max_{t=1,2,\ldots,T-1} (a_{k,t} - a_{k,T}) \tag{10}$$

The last task T is excluded because the forgetting of the last task is always 0.

B Evaluation Benchmarks

We evaluate our approach under three different scenarios: disjoint class-incremental, Si-Blurry class-incremental, and domain-incremental.

Disjoint class-incremental setting is when the datasets are split into disjoint tasks, each consisting of a unique set of classes. We conduct experiments with three datasets under this setting: Split-CIFAR-100 splits the CIFAR-100 dataset (26) into 10 tasks with 10 classes per task. Split-ImageNet-R splits the ImageNet-R dataset (21) into 10 tasks with 20 classes per task. Split-ImageNet-S splits the ImageNet-Sketch dataset (52) randomly into 10 tasks with 100 classes per task or into 20 tasks with 50 classes per task. Split-CUB-200 splits the CUB-200-2011 dataset (51) into 5 tasks with 40 classes per task.

Stochastic incremental-Blurry (Si-Blurry) (41) class-incremental setting is when the class distributions change in a stochastic manner, with classes overlapping across tasks and the task boundaries being dynamic and not clearly defined. We randomly select 50% of the entire classes to be "disjoint classes" (newly encountered classes that never appeared before), and 10% to be "blurry classes" (classes that do not belong to a fixed task and may appear in multiple learning tasks over time).

Domain-incremental setting is when the input distribution shifts over time, but the classes remain consistent. We use the CORe50 dataset (36) for this setting; it has 11 distinct domains (8 for training, 3 for testing). The samples from the training domains arrive sequentially.

C Experimental Details

In this section, we provide details of the experiments we reported in the paper, supplementing Section 4 in the main paper. All experiments are run on a single NVIDIA A-100 GPU.

Data preprocessing Because we focus on the ViT architectures ViT-B/16 and ViT-S/16, all input images are resized to 224×224 and normalized to [0, 1].

Hyperparameters For tuning the threshold values for each dataset (CIFAR-100 (26), ImageNet-R (21), ImageNet-S (52), CUB-200 (51), and CORe50 (36)), we conducted a grid search following the protocol in (38). The threshold grid is shown in Table 2. Table 3 shows the threshold values we used

Threshold	CIFAR-100	ImageNet-R	ImageNet-S	CUB-200	CORe50
Mean	[2.2, 2.6, 2.8, 3.0]	[5.2, 5.4, 5.6, 5.8, 6.0]			[18.0, 24.0, 30.0]
Variance	[0.02, 0.03, 0.04, 0.06, 0.08, 0.1]				[0.6, 0.8, 1.0, 1.2]

Table 2: Hyperparameter grid for the mean and variance threshold values of the loss window in our Online-LoRA.

in our experiments. For CIFAR-100, ImageNet-R, and ImageNet-S, these threshold values remain consistent in both disjoint and Si-blurry class-incremental scenarios.

We set the regularization factor λ=2000.0 (see Equation 7 in the main paper) for all experiments.

Threshold	CIFAR-100	ImageNet-R	ImageNet-S	CORe50	CUB-200
Mean	2.6	5.2	5.6	6.0	24.0
Variance	0.03	0.02	0.06	0.1	1.0

Table 3: Mean and variance thresholds of the loss window for different datasets.

D Loss Surface

Figure 2 shows more qualitative examples of how the loss surface recognizes data distribution shifts, supplementing Section 3.2 in the main paper. MAS (3) introduces the *loss surface* to derive information about incoming streaming data in the task-free scenario. As shown in Figure 2, the peaks on the loss surface indicate shifts in the input data distribution. And the stable regions, namely plateaus, signal the convergence of the model. For instance, the Split CIFAR-100 dataset has 10 distinct tasks, with the data distribution remaining constant within each task. As a result, during the learning process of Split CIFAR-100, there are 9 shifts in data distribution, corresponding to 9 peaks in the loss surface, as illustrated in Figure 2.

To identify plateaus on the loss surface, we employ a *loss window*, which is a sliding window that moves across consecutive training losses. Within this window, we closely observe both the mean and variance of the losses. A plateau is identified when both metrics fall below a predefined threshold (see Appendix C for details). Upon detecting a plateau, we proceed to introduce new LoRA parameters and update the estimation of the model parameter importance. Our goal in identifying plateaus is to mark periods of stable prediction following shifts in data distribution. Therefore, we only classify a phase as a plateau if it follows a peak. A peak is recognized when the loss window's mean increases by an amount exceeding the standard deviation of the window within a single batch.

E Results of Domain-incremental Learning

Table 4 summarizes the results on the domain-incremental setting. Our proposed method, Online-LoRA, not only significantly outperforms other SOTA methods, but also closes a substantial portion of the gap with the upper-bound (UB) performance.

To summarize, the Online-LoRA consistently achieves superior performance under various setups. These results indicate its robustness and adaptability, not only in different ViT setups, but also for dynamically evolving data. In addition to effectively mitigating forgetting, Online-LoRA shows good plasticity.

F Results of Si-blurry Class-incremental Learning

Results on Si-blurry class-incremental setting. Table 5 summarizes the results on the Si-blurry class-incremental benchmarks with datasets CIFAR-100, ImageNet-R, and ImageNet-S. In the Si-blurry scenario, Online-LoRA consistently outperforms all the considered methods by significant margins across both metrics, A_{AUC} and A_{Final}. The superior performance in anytime inference can be largely attributed to Online-LoRA strategic utilization of loss surface plateaus, which consolidates the

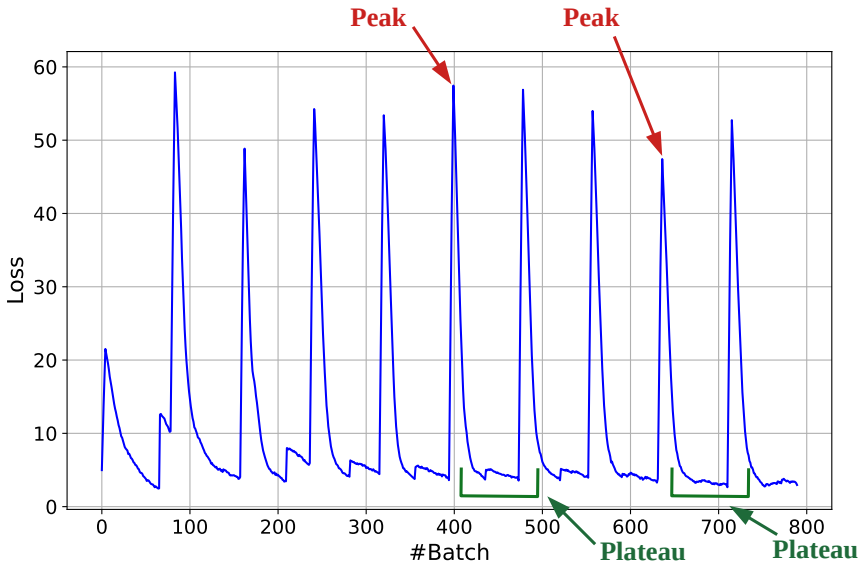


Figure 2: Loss surface of Online-LoRA on Split CIFAR-100 using ViT-B/16 model. Note that other peaks and plateaus exist but are not marked.

	ViT-B/16		ViT-S/16	
	A_{Final} (↑)	*Forgetting* (↓)	A_{Final} (↑)	*Forgetting* (↓)
AGEM (7)	80.15±2.97	2.23±0.81	78.22±3.51	3.19±0.09
ER (8)	85.85±1.35	0.72±0.03	78.99±3.85	5.04±0.10
EWC++ (6)	78.65±6.51	2.31±0.17	79.03±4.54	4.80±0.69
MIR (2)	74.35±4.07	11.01±1.05	86.49±0.81	2.53±0.84
GDumb (42)	77.20±3.49	-	75.64±2.92	-
PCR (34)	87.16±0.73	0.78±0.03	75.20±1.48	0.61±0.02
DER++ (4)	81.88±7.06	10.13±7.00	89.33±0.62	0.42±0.57
LODE (DER++) (32)	77.02±2.22	17.30±2.82	83.48±5.84	24.54±0.94
L2P (54)	87.97±0.37	**0.00±0.00**	86.47±0.23	**0.00±0.00**
MVP (40)	84.82±0.54	**0.00±0.00**	79.85±0.33	3.55±0.39
Ours	**93.71±0.01**	**0.00±0.00**	**90.96±0.02**	**0.00±0.00**
Upper Bound (*UB*)	95.6±0.01	-	93.56±0.01	-

Table 4: Results of domain-incremental learning on CORe50 (36). '↑' means higher is better and '↓' means lower is better. Online-LoRA not only achieves the highest final accuracy but also demonstrates the lowest forgetting.

knowledge precisely when needed. Online-LoRA is also more flexible than EWC (24) which does so only at specific discrete moments; Online-LoRA also avoids the excessive frequency of updates that introduce noise as seen in EWC++ (6).

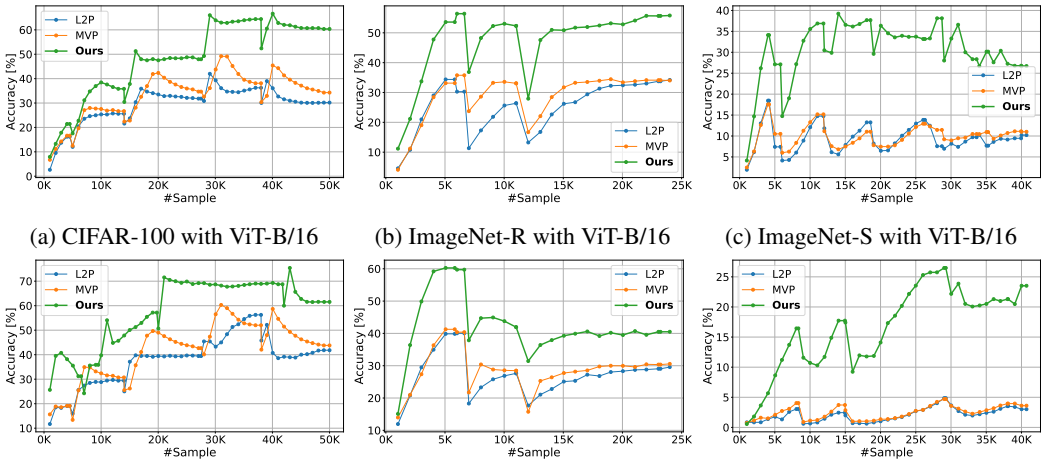
Figure 3 displays the trend of accuracy as more samples are provided, highlighting the consistent performance of Online-LoRA across two different ViT architectures. Compared to other methods, Online-LoRA effectively learns new knowledge from incoming samples, which leads to an increase in accuracy.

G Results of Swin Transformer

In this section, we present the results for the disjoint class-incremental and domain-incremental settings (for details on these settings, see Section **??** in the main paper) using the Swin Transformer architecture (35). For a fair comparison, the hyperparameters for the baseline methods are set

		CIFAR-100 (26)		ImageNet-R (21)		ImageNet-S (52)	
		A_{AUC} (↑)	A_{Final} (↑)	A_{AUC} (↑)	A_{Final} (↑)	A_{AUC} (↑)	A_{Final} (↑)
ViT-B/16	L2P	43.01±9.37	39.86±2.28	22.71±1.86	27.08±2.49	10.02±0.42	13.58±4.04
	MVP	47.52±9.74	44.49±0.93	27.79±0.62	31.64±1.77	10.68±0.45	13.99±1.73
	Ours	**60.12**±5.79	**61.70**±6.29	**45.05**±1.59	**48.00**±6.01	**30.81**±2.09	**30.22**±4.36
	UB	89.50±0.04		76.78±0.44		63.82±0.02	
ViT-S/16	L2P	37.82±12.19	30.88±1.39	24.31±1.83	21.83±2.13	2.00±0.12	3.61±1.08
	MVP	40.31±9.52	35.55±2.11	27.04±1.09	26.67±3.70	2.27±0.14	3.72±0.77
	Ours	**52.84**±7.97	**58.72**±1.44	**39.47**±1.93	**36.61**±4.63	**15.35**±0.92	**20.18**±1.84
	UB	86.55±0.01		69.94±0.34		59.28±0.11	

Table 5: Results of Si-blurry class-incremental learning. '↑' means higher is better and '↓' means lower is better. All datasets are split into 5 blurry tasks. To ensure a fair comparison with L2P (54) and MVP (40), we exclude the loss from hard buffer samples in Online-LoRA. The best results are noted by **bold**.



(a) CIFAR-100 with ViT-B/16 (b) ImageNet-R with ViT-B/16 (c) ImageNet-S with ViT-B/16

(d) CIFAR-100 with ViT-S/16 (e) ImageNet-R with ViT-S/16 (f) ImageNet-S with ViT-S/16

Figure 3: Average accuracy versus number of samples for Si-Blurry CIFAR-100, ImageNet-R, and ImageNet-S scenarios. As shown, the Online-LoRA consistently outperforms competing methods, maintaining high accuracy throughout.

according to the descriptions in Appendix J.3. For our method, we use a learning rate of 0.0003 for the Swin Transformer.

As shown in Table 6, our approach consistently outperforms other baseline methods in both disjoint class-incremental and domain-incremental learning settings. This demonstrates that our method remains effective across various ViT architectures, extending beyond the ViT-B/16 and ViT-S/16 models reported in Section 4.2 of the main paper.

H Exploration with length of task sequence

Table 7 summarizes the results on Split ImageNet-S dataset across varying task sequence lengths; Table 8 summarizes the results on Si-blurry ImageNet-S. As the task sequence is longer, all methods experience a decline in performance. However, Online-LoRA exhibits the smallest reduction in performance, showcasing its robustness against longer task sequences. This can be attributed to its utilization of loss surface plateaus, which effectively captures and adapts to shifts in data distribution at instance level.

In contrast, for prompt-based learning methods such as L2P, longer task sequences challenge the capacity of prompt pool as more task-specific information needs to be encoded. Similarly, for replay-based methods, the strategy of selecting informative samples from the buffer is prone to biases

Method	Split-ImageNet-S		CORe50	
	A_{Final} (↑)	*Forgetting* (↓)	A_{Final} (↑)	*Forgetting* (↓)
AGEM (7)	31.67±0.96	50.12±0.27	90.15±1.31	1.16±0.05
ER (8)	42.60±0.75	38.68±0.26	88.93±2.99	4.16±0.09
EWC++ (6)	29.57±1.57	51.87±0.04	90.91±1.28	0.04±0.02
MIR (2)	42.90±0.19	38.49±0.15	87.47±0.65	5.67±0.14
GDumb (42)	14.76±1.13	-	79.52±3.00	-
Ours	**53.75**±0.29	**32.86**±0.89	**95.29**±0.06	**0.00**±0.00
UB	71.98±0.23	-	97.56±0.02	-

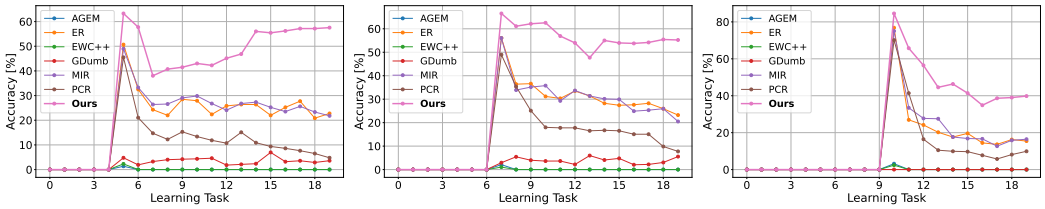
Table 6: Results of disjoint class-incremental learning and domain-incremental learning using Swin Transformer. '↑' means higher is better and '↓' means lower is better. The best results are noted by **bold**. *UB* is the upper-bound performance. With Swin Transformer, our Online-LoRA method consistently outperforms other baseline methods across various settings, demonstrating its adaptability and effectiveness across different ViT architectures.

Method	10 tasks		20 tasks	
	A_{Final} (↑)	*Forgetting* (↓)	A_{Final} (↑)	*Forgetting* (↓)
AGEM (7)	0.16±0.04	9.42±0.17	0.11±0.05	7.96±0.10
ER (8)	30.21±0.70	37.14±1.83	22.81±0.30	43.61±0.16
EWC++ (6)	0.32±0.28	22.46±4.69	0.11±0.05	**5.26**±0.45
MIR (2)	30.33±3.81	35.92±1.75	22.04±0.41	39.17±0.13
GDumb (42)	1.65±0.22	-	1.97±0.79	-
PCR (34)	38.75±0.22	35.01±2.12	17.87±2.18	45.46±0.07
DER++ (4)	6.47±0.06	15.34±0.15	2.29±0.23	23.14±0.06
LODE (DER++) (32)	9.97±2.29	**8.48**±1.24	13.47±0.66	35.89±1.63
EMA (DER++) (49)	16.88±2.23	36.28±1.09	11.55±0.66	38.56±0.22
EMA (RAR) (49)	14.06±0.37	36.28±1.09	9.05±0.60	29.77±1.70
Ours	**47.06**±0.24	**28.09**±3.25	**44.19**±2.09	**28.48**±0.24
Upper Bound (*UB*)	63.82±0.02			

Table 7: Comparison with other methods on Split ImageNet-S for different lengths of task sequences. '↑' means higher is better and '↓' means lower is better. ViT-B/16 model is used.

in longer task sequences. This bias may result in an inadequate representation of earlier tasks or an overemphasis on more recent tasks, hurting the methods overall performance.

Furthermore, Figure 4 shows the accuracy on the validation set for four tasks at the time they are first encountered and after each subsequent task is learned (see Appendix M for results of other tasks). As shown in Figure 4, Online-LoRA consistently outperforms the other SOTA methods in terms of



(a) Task #5 (b) Task #7 (c) Task #10

Figure 4: Test accuracy of three tasks versus the number of learning tasks. ViT-B/16 model is used on Split ImageNet-S with 20 tasks. The accuracy for each task prior to the model being trained on it is recorded as zero, since no measurements are taken at that stage, as the model has not yet been exposed to the corresponding task.

Method	Task sequence	ImageNet-S	
		A_{AUC} (↑)	A_{Final} (↑)
L2P (54)	5 tasks	10.02±0.42	13.58±4.04
MVP (40)		10.68±0.45	13.99±1.73
Ours		**30.81**±**2.09**	**30.22**±**4.36**
L2P (54)	10 tasks	9.06±0.43	12.49±3.39
MVP (40)		9.50±0.29	12.24±2.16
Ours		**30.69**±**0.59**	**31.44**±**4.39**
L2P (54)	20 tasks	6.57±0.54	7.13±0.89
MVP (40)		7.87±0.24	8.98±1.49
Ours		**26.91**±**0.25**	**25.73**±**6.15**

Table 8: Comparison with prompt-based methods on Si-blurry ImageNet-S at different length of task sequence. ViT-B/16 is used.

Incremental LoRA	Hard loss	A_{Final} (↑)	$Forgetting$ (↓)
-	-	28.68±0.13	53.45±0.04
✓	-	34.74±0.31	34.37±1.15
-	✓	36.08±0.19	35.75±0.33
✓	✓	**48.23**±**0.74**	**23.85**±**1.08**

Table 9: Ablation results of ViT-B/16 model on Split ImageNet-R dataset. '↑' means higher is better and '↓' means lower is better. "Incremental LoRA": introducing new, trainable LoRA at each loss plateau with the model parameter regularization in Equation 7. . "Hard loss": including $\mathcal{L}(F(X_B;\theta), Y_B)$ (the loss from hard buffer samples) in the final learning objective in Equation 6. ✓ indicates the presence of the component, − indicates its absence.

preserving the performance of previously learned tasks, which underscores the effectiveness of our online parameter regularization in mitigating catastrophic forgetting.

I Ablation Study

I.1 Ablation study on Imagenet-R dataset

Table 9 shows the ablation study on the effectiveness of each component ("incremental LoRA" and "hard loss") of Online-LoRA on Split ImageNet-R (10 tasks). The results demonstrate the crucial role of each component of Online-LoRA in overall performance. More results in Appendix I.

Simply fine-tuning a single set of LoRA parameters (i.e. without incorporating any components of Online-LoRA) results in significantly worse performance compared to our approach, with a 20% drop in accuracy (from 48.23% to 28.68%). Additionally, excluding the loss from hard buffer samples within the Online-LoRA framework leads to a substantial performance decline from 48.23% to 34.74% (a 13.5% decrease). This emphasizes the crucial role of maintaining a minimal buffer with only the four most challenging samples in mitigating forgetting.

Furthermore, the absence of new LoRA initialized at plateaus of the loss surface and model parameter regularization results in a significant performance decline of 12%, from 48.23% to 36.08%. This highlights the importance of continuously adding new LoRA parameters to minimize task interference and implementing online weight regularization to prevent catastrophic forgetting.

I.2 Ablation Study on Imagenet-S Dataset

In addition to the ablation results on Split Imagenet-R presented in Section **??** of the main paper, this section provides further ablation results on the Split Imagenet-Sketch dataset with varying task lengths. As shown in Table 10, our Online-LoRA consistently outperforms other variants that lack certain components. These results demonstrate that both the hard buffer loss and incremental LoRA, along with online parameter regularization, are crucial for the performance of our approach.

Incremental LoRA	Hard loss	10 tasks		20 tasks	
		A_{Final} (↑)	$Forgetting$ (↓)	A_{Final} (↑)	$Forgetting$ (↓)
-	-	30.66±0.25	38.70±0.40	24.49±2.61	39.29±2.57
✓	-	31.11±2.60	34.62±2.98	32.47±0.29	33.14±1.39
-	✓	36.26±0.12	39.29±2.57	35.43±4.99	32.56±2.72
✓	✓	**47.06±0.24**	**28.09±3.25**	**44.19±2.09**	**28.48±0.24**

Table 10: Ablation results of ViT-B/16 model on Split ImageNet-Sketch dataset. '↑' means higher is better and '↓' means lower is better. "Incremental LoRA": introducing new, trainable LoRA at each loss plateau with the model parameter regularization in Equation 7 in paper. "Hard loss": including $\mathcal{L}(F(X_B;\theta), Y_B)$ (the loss from hard buffer samples) in the final learning objective in Equation 6 in paper. A check mark (✓) indicates the presence of the component, while a dash (—) indicates its absence.

The baseline involves continuous fine-tuning of a single set of LoRA parameters. In contrast, Online-LoRA introduces an incremental LoRA architecture coupled with parameter importance-based regularization, and preserves a hard buffer along with its loss computations. Individually, each component improves performance and reduces forgetting. However, integrating both components into the baseline achieves the optimal performance, demonstrating the efficacy of our complete approach.

I.3 Impact of Pre-trained Weights

In this section, we demonstrate that our experimental settings do not provide any unfair advantage to our Online-LoRA approach through the use of pre-trained ViT models.

First, it is important to note that all baseline methods in our experiments utilize the same pre-trained ViT models as their backbones, just like Online-LoRA. Consequently, all methods benefit from the pre-training to varying extents, particularly those originally implemented with ResNet18 backbones (Table 12). For detailed information on the backbones used by each baseline, please refer to Appendix J.2.

Second, we show that simply using pre-trained models without applying any CL methods or strategies fails to yield competitive performance. To illustrate this, we introduce three simple baselines:

- **Frozen FT**: This baseline freezes the pre-trained backbone (feature extractor). Only the classification head (the final layer used for classification) is continuously fine-tuned on the data stream. Given that the model is pre-trained on the ImageNet-21K dataset, if any unfair advantage exists due to data leakage or other factors, it should be evident here by showing strong performance.
- **Continual FT**: This baseline fully fine-tunes the pre-trained model, including both the backbone and the classification head, on each new data batch. This is consistent with our OCL setting where the model encounters each data batch only once. If the pre-trained weights alone brings any unfair advantage, this baseline should perform competitively, similar to methods specifically designed for CL.
- **Random Head**: This baseline uses the pre-trained model's backbone with a newly initialized classifier head and performs only inference without any fine-tuning. Since the classifier head is randomly initialized, it should provide a clear lower bound for performance, demonstrating that without any adaptation or learning, the model's performance is essentially at chance level.

As shown in Table 11, **Random Head** baseline achieves near-zero accuracy, confirming that merely using pre-trained weights without adaptation to the test dataset does not have an advantage. Although the **Frozen FT** and **Continual FT** baselines outperform some CL methods (which also use the same pre-trained models), they still suffer from severe forgetting and exhibit a significant performance gap compared to other methods, particularly our Online-LoRA, with nearly a 20% difference in final accuracy and a 30% difference in forgetting.

These results demonstrate that the performance advantages of our Online-LoRA method over the baseline CL methods are not simply due to the use of pre-trained models. Instead, they arise from the

Method	Accuracy (↑)	Forgetting (↓)
Random Head	0.08±0.00	-
Frozen FT	27.98±0.29	55.12±0.43
Continual FT	28.49±0.21	53.49±0.07
AGEM (7)	5.60±2.74	53.97±1.97
ER (8)	40.99±3.96	32.38±0.89
EWC++ (6)	3.86±2.02	56.95±1.46
MIR (2)	41.51±2.99	31.32±5.17
GDumb (42)	1.65±0.22	-
PCR (34)	46.11±3.03	25.50±0.41
DER++ (4)	30.90±8.04	24.26±4.14
LODE (DER++) (32)	42.20±6.46	31.83±1.05
EMA (DER++) (49)	41.75±1.98	32.65±1.55
EMA (RAR) (49)	30.04±0.33	39.36±0.04
Online-LoRA (ours)	**48.18±0.63**	**23.85±1.48**
UB	63.82±0.02	-

Table 11: Performance comparison between pre-trained models without CL strategies and pre-trained models with CL strategies on Split ImageNet-R (online class-incremental learning setting). ViT-B/16 backbone is used. While some methods do not outperform simple fine-tuning on a continuous data stream, other CL methods provide significant performance improvements to the pre-trained model. This demonstrates that the advantages of CL methods, including Online-LoRA, are not solely due to the use of pre-trained weights but also stem from the effectiveness of the methods themselves. UB is the upper-bound baseline trained on the i.i.d. data of the datasets. The best results are noted by **bold**.

effectiveness of our approach. The pre-trained weights provide a common foundation for all methods, but it is our approach that leads to superior performance.

J Baseline Settings

In this section, we provide the experimental settings for the baseline methods used in our experiments[1].

J.1 Overview of Baselines

- **AGEM** (7): Averaged Gradient Episodic Memory, utilizes samples in the memory buffer to constrain parameter updates.
- **ER** (8): Experience replay, a rehearsal-based method with random sampling in memory retrieval and reservoir sampling in memory update.
- **EWC++** (6): An online version of EWC (24), a regularization method that limits the update of parameters crucial to past tasks.
- **MIR** (2): Maximally Interfered Retrieval, a rehearsal-based method that retrieves memory samples with loss increases given the estimated parameter update based on the current batch.
- **GDumb** (42): Greedy Sampler and Dumb Learner, a strong baseline that greedily updates the memory buffer from the data stream with the constraint to keep a balanced class distribution.
- **PCR** (34): Proxy-based contrastive replay, a rehearsal-based method that replaces the samples for anchor with proxies in a contrastive-based loss.
- **DER++** (4): Dark Experience Replay++, a rehearsal-based method using knowledge distillation from past experiences.
- **LODE** (32): Loss Decoupling, a rehearsal-based method that decouples the learning objectives of old and new tasks to minimize interference.

[1]Codebases used: https://github.com/AlbinSou/online_ema.git, https://github.com/liangyanshuo/Loss-Decoupling-for-Task-Agnostic-Continual-Learning.git, https://github.com/FelixHuiweiLin/PCR.git, https://github.com/RaptorMai/online-continual-learning.git

Method	Acc. w/ ResNet18	Acc. w/ ViT-B/16	Performance Gain (%)
AGEM (7)	5.4±0.6	12.67±1.87	134.63
ER (8)	14.5±0.8	44.85±1.83	209.31
EWC++ (6)	4.8±0.2	10.61±0.74	121.04
MIR (2)	14.8±0.7	48.36±3.11	226.76
GDumb (42)	24.8±0.7	41.00±19.97	65.32
PCR (34)	21.8±0.9	48.48±0.15	122.39
DER++ (4)	15.5±1.0	36.64±6.11	136.39
LODE (DER++) (32)	37.8±1.1	44.29±1.48	17.17
EMA (DER++) (49)	23.2±1.2	42.28±4.36	82.24
EMA (RAR) (49)	35.4±1.2	47.10±0.82	33.05

Table 12: Performance comparison on CIFAR-100 between ResNet18 and pre-trained ViT-B/16 in an online class-incremental learning scenario. Acc. stands for Accuracy. All rehearsal-based methods use a buffer size of 500 for fair comparison. The results demonstrate that there is no unfair comparison in our experiments, as all methods benefit from the pre-trained ViT-B/16 model. The performance gain is computed as the percentage increase from the ResNet18 accuracy to the ViT-B/16 accuracy for each method.

- **EMA** (49): Exponential Moving Average, a model ensemble method that combines models from various training tasks.
- **L2P** (54): Learning to Prompt, a prompt-based method that prepends learnable prompts selected from a prompt pool to the embeddings of a pre-trained transformer.
- **MVP** (40): Mask and Visual Prompt tuning, a prompt-based method that uses instance-wise feature space masking.

J.2 Backbone

Among the baseline methods we compare, L2P (54) and MVP (40) originally reported results using a ViT-B/16 model (12) pre-trained on ImageNet21k, while the other baselines (AGEM (7), ER (8), EWC++ (6), MIR (2), GDumb (42), DER++ (4), PCR (34), LODE (32), EMA (49)) reported results using a ResNet18 (18) architecture.

To ensure a fair comparison, we standardized our experimental setup by evaluating all baselines using the same pre-trained ViT model (ViT-B/16 and ViT-S/16). For methods originally implemented with ResNet18, we reimplemented them with ViT to match the experimental conditions of L2P and MVP. As shown in Table 12, all methods perform better with the pre-trained ViT-B/16 than with ResNet18, supporting our argument that using a pre-trained ViT provides a more consistent and stronger baseline for performance comparisons.

J.3 Training Settings

The following settings are shared by the baseline methods (and our Online-LoRA) in the experiments:

- Buffer Size: 500. Methods using a buffer include AGEM (7), ER (8), MIR (2), GDumb (42), PCR (34), DER++ (4), LODE (32), and EMA (49).
- Optimizer: Adam.
- Batch Size: 64.

In Table 13, we summarize the hyperparameters used for all baseline methods in our experiments. To ensure a fair comparison, we adopted these hyperparameters from their original codebases. However, because the baseline methods used different backbones and batch sizes in their original experiments, we adjusted the learning rates for some baselines to standardize the comparison across all methods. For tuning the learning rates, we followed the protocol outlined in (38) and conducted a grid search on a small cross-validation set. The hyperparameter grid for the baselines is detailed in Table 14.

K Exploration with Buffer Size

Table 15 we show more results of the impact of buffer sizes on the performance of replay-based methods (AGEM (7), ER (8), GDumb (42), MIR (2)).

Method	CIFAR-100	ImageNet-R	ImageNet-S	CUB-200	CORe50
AGEM (7)	LR=0.0001, WD=0.0001				
ER (8)	LR=0.0001, WD=0.0001, Episode memory per batch=10				
EWC++ (6)	LR=0.0001, WD=0.0001, λ=100, α=0.9 Number of training batches after which the Fisher information will be updated: 50				
MIR (2)	LR=0.0001, WD=0.0001, Number of subsample=50				
GDumb (42)	LR=0.001, WD=0.0001, Minimal learning rate: 0.0005, Gradient clipping=10, Epochs to train for memory=30				
PCR [34]	LR=0.0001, WD=0.0001, Episode memory per batch=10, Temperature=0.09, Warmup of buffer before retrieve=4				
DER++ (4)	LR=0.0003, α=0.2, β=0.5				
LODE (32)	LR=0.0003, C=1.0, ρ=0.1				
EMA (49)	LR=0.0002, λ for warm-up: 0.9, λ=0.99				
L2P (54)	LR=0.003, Size of the prompt pool=10, Length of a single prompt=10, Number of prepended prompt=4				
MVP (40)	LR=0.005, γ=2.0, m=0.5, α=0.5				

Table 13: Hyperparameters for the baseline methods on ViT-B/16. LR: learning rate. WD: weight decay.

Method	CIFAR-100	ImageNet-R	ImageNet-S	CUB-200	CORe50
AGEM (7)	LR: [0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1] WD: [0.0001, 0.001, 0.01, 0.1]				
ER (8)	LR: [0.0001, 0.0003, 0.001, 0.003] WD: [0.0001, 0.001, 0.01, 0.1]				
EWC++ (6)	LR: [0.0001, 0.001, 0.01, 0.1] WD: [0.0001, 0.001]				
MIR (2)	LR: [0.0001, 0.001, 0.01, 0.1] WD: [0.0001, 0.001]				
GDumb (42)	LR: [0.001, 0.01, 0.1 WD: [0.0001, 0.000001]				
PCR [34]	LR: [0.0001, 0.001, 0.01, 0.1] WD: [0.0001, 0.001]				
DER++ (4)	LR: [0.0003, 0.003, 0.03]				
LODE (32)	LR: [0.0003, 0.003, 0.03]				
EMA (49)	LR: [0.0001, 0.0002, 0.0003, 0.0004, 0.0005]				

Table 14: Hyperparameter grid for the baseline methods using the ViT-B/16 backbone. LR: learning rate; WD: weight decay. Since L2P (54) and MVP (40) use the same backbone and batch size as in our experiments, their learning rates were not adjusted.

As shown in Table 15, when the buffer size increases, all replay-based methods see improvements in their performance across the benchmarks. Notably, when the buffer size hits 5000 (a large capacity; 20% of the ImageNet-R training set, 12.5% of the ImageNet-S training set), the difference in performance between GDumb and other replay-based methods narrows. This suggests that the sophisticated memory retrieval strategies employed by these other methods do not significantly outperform GDumb's simple approach of training directly on the buffered data. Moreover, the performance of rehearsal-based methods drops when the buffer size shrinks. This highlights the

Buffer size	Method	Split-ImageNet-R	Split-ImageNet-S	Core50
500	AGEM (7)	5.60±2.74	0.16±0.04	80.15±2.97
	ER (8)	40.99±3.96	30.21±0.70	85.85±1.35
	MIR (2)	41.51±2.99	30.33±3.81	74.35±4.07
	GDumb (42)	8.87±1.36	1.65±0.22	77.20±3.49
1000	AGEM (7)	7.16±1.56	0.23±0.04	78.73±3.87
	ER (8)	44.71±2.63	34.32±0.53	84.27±4.11
	MIR (2)	46.65±5.63	33.99±1.72	82.64±1.12
	GDumb (42)	19.19±1.36	2.71±0.12	78.09±3.75
5000	AGEM (7)	7.21±0.34	0.12±0.02	77.57±3.56
	ER (8)	47.23±2.71	37.65±0.23	81.32±2.19
	MIR (2)	**49.33±3.49**	35.90±2.35	81.18±3.20
	GDumb (42)	46.08±0.64	9.68±0.28	69.42±1.06
	Ours	48.18±0.63	**47.06±0.24**	**93.71±0.01**
	UB	76.78±0.44	63.82±0.02	95.60±0.01

Table 15: Results of replay-based methods with different buffer size. A_{Final} metric and ViT-B/16 model is used. Each dataset has 10 disjoint tasks. *UB* is the upper-bound baseline trained on the i.i.d. data of the datasets. The best results are noted by **bold**.

Method	#params (M)	FLOPs ($\times 10^{15}$)	Training time (s)
AGEM (7)	85.88	140.52	828.39
ER (8)	85.88	140.05	849.43
EWC++ (6)	85.88	214.36	1076.53
GDumb (42)	85.88	18.44	2078.59
MIR (2)	85.88	161.04	1069.29
Ours	86.47	151.20	864.60

Table 16: Computational statistics for Online-LoRA and baseline methods on CIFAR-100 in the online class-incremental learning scenario using the ViT-B/16 backbone. FLOPs are measured as 'forward FLOPs per GPU' using the DeepSpeed FLOPS Profiler (39). All experiments are conducted on a single NVIDIA A100 GPU.

efficiency of our Online-LoRA, which achieves high performance using just a minimal buffer size of 4.

L Computation Analysis

L.1 Efficiency via Separate LoRA Adapters

In this section, we explain how our online parameter importance estimation achieves greater efficiency compared to EWC (24) and EWC++ (6) by treating the LoRA adapter $\sum_{t'} B_{t'} A_{t'} X$ as two separate linear layers.

In EWC (24), the size of the importance weight matrix equals to the number of parameters squared. For instance, to employ EWC in ViT-B/16, the model needs to store and update a 86.6M×86.6M matrix, representing a significant memory an computational overhead. By handling the LoRA adapter as two distinct layers, our Online-LoRA approach employs two smaller importance weight matrices, $\Omega^{A,l} \in R^{d \times r}$ and $\Omega^{B,l} \in R^{r \times k}$, for each attention layer. The combined size of these matrices is proportional to the total number of LoRA parameters, calculated as follows: #attention heads × 2 (for Q and V projection matrices) × input size × rank × 2. For a ViT-B/16 model with a LoRA rank of 4, this equates to a total of: 12 heads × 2 × 768 input size × 4 rank × 2 = 147,456. This additional memory footprint is negligible (∼0.17% of the total parameters of the ViT-B/16 model), which enables the *online* updates of the importance weights.

L.2 Training Statistics

In this section, we present the model parameter size, training FLOPs, and training time for our Online-LoRA and the baseline methods.

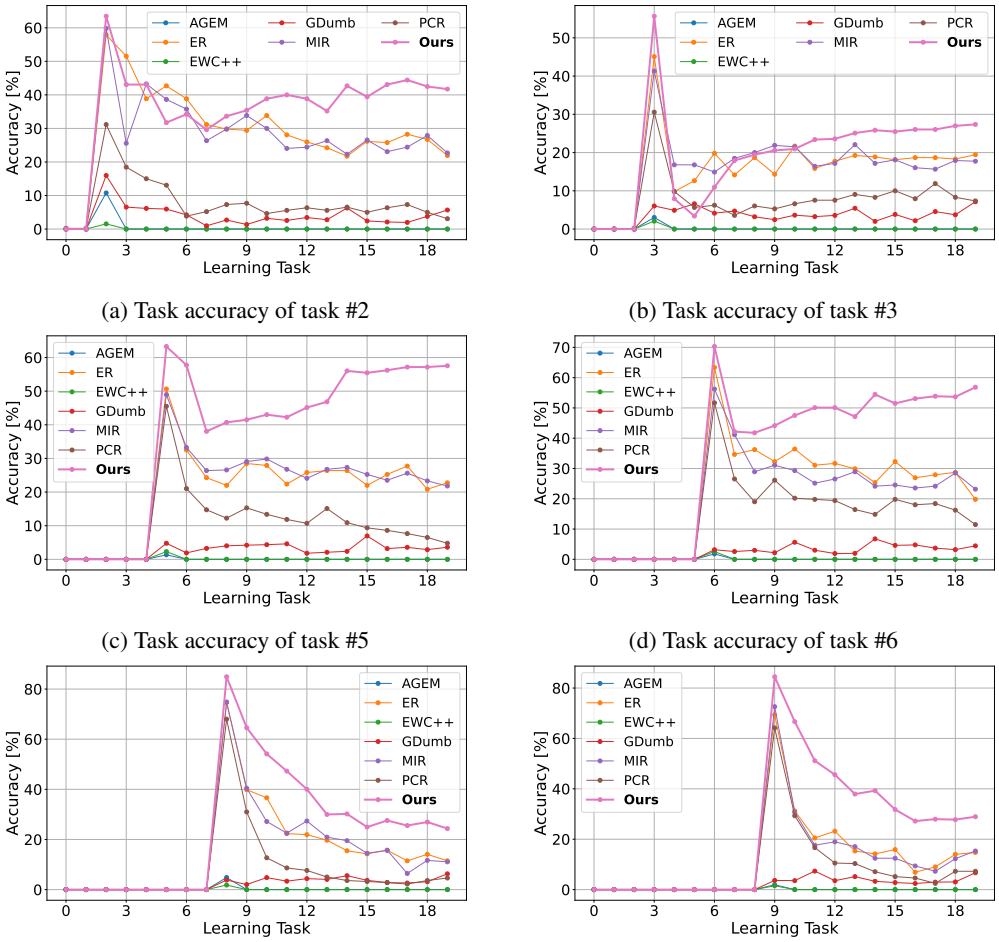
As shown in Table L.2, our Online-LoRA model introduces approximately 0.6M additional parameters due to the inclusion of LoRA parameters, which represents a negligible increase (0.69%) compared to the original size of the ViT-B/16 model. Notably, our memory buffer contains only 4 data samples, whereas other baselines (except EWC++) require at least 500 samples in their buffers to achieve comparable performance (see Appendix K for more details). Regarding computational consumption measured by FLOPs during training, Online-LoRA demonstrates advantages over EWC++ (6), thanks to our efficient computation of the importance weight matrix, as explained in Section 3.3 of the main paper. The extremely low FLOPs of GDumb (42) can be attributed to its design, which involves greedily updating the memory buffer without employing additional strategies. However, its training time is relatively high because retraining is triggered frequently to maintain a balanced memory buffer, which adds overhead despite the low FLOPs.

M Task Accuracy

In this section, Figure 5 and Figure 6 show task accuracy as a function of the number of learning tasks as described in Section H in the main paper. The ViT-B/16 model is employed on the Split ImageNet-S dataset with 20 tasks. These results demonstrate that our Online-LoRA consistently outperforms the other methods in mitigating the forgetting of previously learned tasks.

Figure 5a shows that AGEM (7) begins with an initial accuracy of $\sim$10%. However, this accuracy drastically decreases for subsequent tasks, eventually dropping to zero. Given that the Split ImageNet-S dataset consists of 20 tasks with 500 classes per task, AGEM's performance is no better than that of a random model, which would have an expected accuracy of 0.2%. This dramatic decline is primarily due to the increasingly restrictive constraints placed on gradient updates as the number of tasks increases. Such constraints significantly hurt the model's ability to learn from new tasks, showing a fundamental weakness of AGEM in handling long sequences of diverse tasks. A similar issue was observed with EWC++ (6), another regularization-based approach.

In contrast, our Online-LoRA model does not encounter this problem even though an online weight regularization is used. This is because our model is continuously expanded by adding new LoRA parameters (see Section 3.2 in the main paper). This strategy allows the model to adapt to new information more flexibly, bypassing the learning limitations encountered by traditional regularization methods like AGEM and EWC++.



(a) Task accuracy of task #2

(b) Task accuracy of task #3

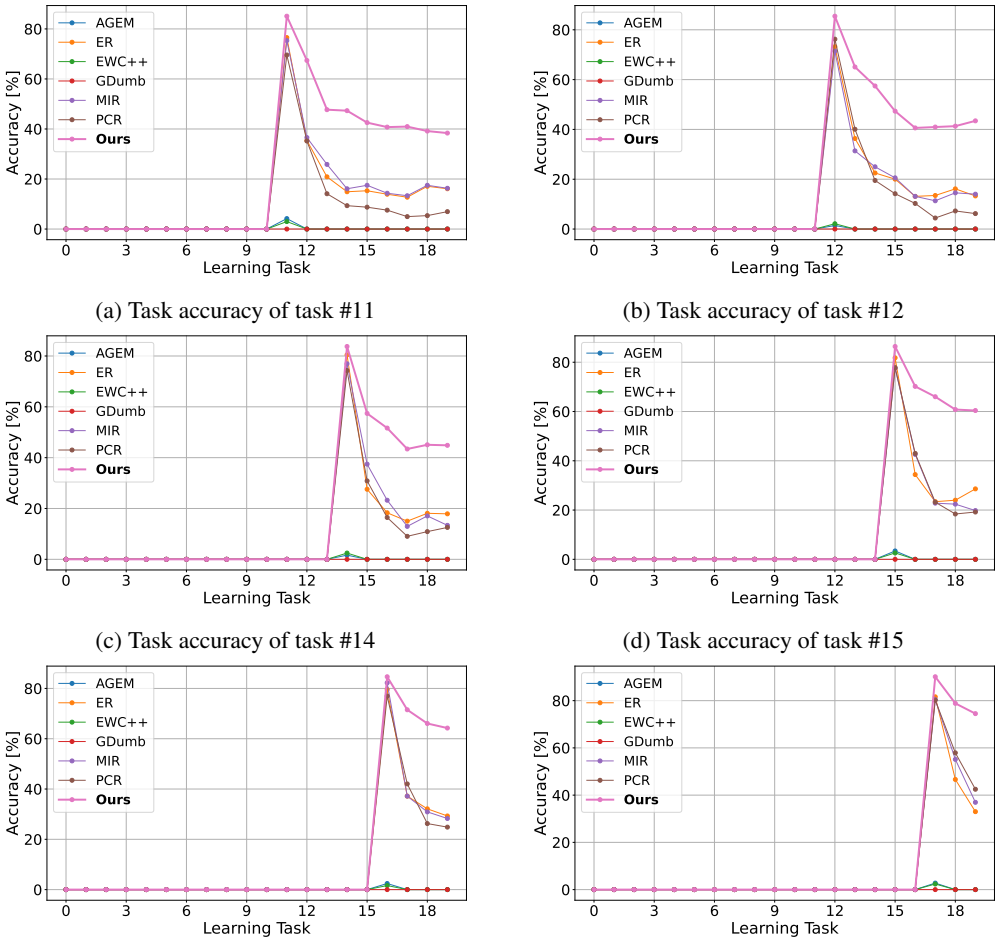
(c) Task accuracy of task #5

(d) Task accuracy of task #6

(e) Task accuracy of task #8

(f) Task accuracy of task #9

Figure 5: Task accuracy versus the number of learning tasks of task #2 to task #9. Our Online-LoRA consistently outperforms all the other methods in maintaining accuracy on previously learned tasks. Note that the recorded accuracy for initial tasks is zero, not due to poor model performance, but because our evaluation prioritizes mitigating forgetting in tasks the model has already encountered.



(a) Task accuracy of task #11

(b) Task accuracy of task #12

(c) Task accuracy of task #14

(d) Task accuracy of task #15

(e) Task accuracy of task #16

(f) Task accuracy of task #17

Figure 6: Task accuracy versus the number of learning tasks of task #11 to task #17. Compared to the results of task #2 to task #9 in Figure 5, our Online-LoRA has greater advantages over the other methods for these newer tasks #11 to task #17. Zero accuracy for initial tasks results from not measuring them at the time the specific task had not been learned yet.