DKT: Diverse Knowledge Transfer Transformer for Class Incremental Learning Supplementary materials

A. Appendix

A.1. Experiment details

A.1.1 Implementation details

We train our model on all datasets for 500 epochs. The optimizer is Adamw which sets weight-decay as $1e^{-6}$, decay-rate as 0.1 and cosine learning rate as $1e^{-3}$, including 5 epochs of warmup like other vision transformer's training. Also, the backbone is modified from RVT [6]. we applied the standard data augmentation of RVT [6]. We use RandAugment [1], random cropping, horizontal flip, normalization and cutout [2]. The cutout [2] is only used in CIFAR100 like FOSTER [8]. To alleviate the influence of unbalanced datasets, following LUCIR [4], DER [10], and Dytox [3], at the end of each task, we finetune our model with the balanced datasets selected in the memory buffer to alleviate the bias between the old categories and new categories. We choose to finetune our model with a learning rate $1e^{-4}$. In CIFAR100, For the 10 steps set, we finetune the model for one epoch before 3 steps and 10 epochs for other steps. For the 5 steps set, we finetune the model for one epoch before 2 steps and 10 epochs for other steps. For the 20 steps set, we finetune the model for one epoch before 4 steps and 10 epochs for other steps. In ImageNet100/1000, we finetune the model for 5 epochs before 3 steps and 10 epochs for other steps. During model training, the task-specific tokens and stability classifier in the duplex classifier are frozen to preserve the previous knowledge. Conversely, the plastic classifier in the duplex classifier and the task-general token are allowed to learn new knowledge and update the general knowledge in the tasks, respectively. During finetuning, we fix the feature extractor including SABs and GKAB. Corresponding to this, we open SKAB and duplex classifier to reduce the bias towards the new categories. We fix the task-specific tokens to maintain specific knowledge for previous tasks and set the task-general token trainable for GKAB to update the general knowledge. The order and sampler strategy are following Dytox [3]. Our batch size is 256 to suit our GPUs.

To balance the training time and accuracy, we set batch size 256 for two GPUs (RTX3090) on CIFAR100 and ImageNet100, and 256 for four GPUs on ImageNet1000 to accelerate training.

GPUs	Avg
2	75.83
4	75.61

Table 1. Performance on several GPUs in 10 steps on CIFAR100

A.1.2 Memory Buffer Setting

Differing from CNN, Vision Transformers usually use adam [5] optimizer and DistributedDataParallel instead of SGD and DataParallel. So the previous work [3] has made a mistake. The model selects N times anchors than the memory buffer, in which N is the number of GPUS, causing the accuracy of the paper much higher than true performance(73.3%→67.33% in 10 steps on CIFAR100). So in erratum distributed, It [3] reports the true performance and we compare our model with this performance. We solve this bug to maintain a suitable memory buffer. We suspend all but GPU0 and use GPU0 to select anchors using the way in iCaRL [7] like other continual learning methods [3, 7, 9–11]. We store the anchors as a new file and then push the anchors into other GPUs to guarantee that the memory buffer is suitable. Our experiment in Table 1 shows that our model can get similar performance in two or four GPUs just need to change the batch-size like 128 in four GPUs and 256 in two GPUs.

A.1.3 Computation FLOPs

Although FLOPs may not be a critical factor in incremental learning tasks, we posit that they can serve as a useful metric to gauge the superiority of a model. Thus, we have elected to include a comparison of FLOPs with respect to previous tasks in table 2. In Table 2, we have provided FLOPs of the proposed *DKT* and the competitive *DyTox global* and *DER w/o P* methods on CIFAR100 and ImageNet1000. We can see that, as DKT only needs to increase the computation of a task-specific token for each incremental task addition-

ally, instead of accumulatively computing the decoder like DyTox, it is almost computation cost-free in learning new tasks (less than 1% FLOPs increase after the final task). Instead, *DyTox global* and *DER w/o P* suffer about 30.8% and 899.7% FLOPs increase after the final learning task. This result demonstrates the efficiency of our novel design using cross-attention and task-specific tokens in learning new tasks.

	CIFAR100		ImageNet1000	
Methods	**FLOPs start**	**FLOPs final**	**FLOPs start**	**FLOPs final**
DER w/o P	0.372	3.716	1.827	18.265
DyTox global	0.606	0.793	2.010	2.552
Our DKT	0.615	0.618	2.064	2.066

Table 2. Computation complexity (*i.e.*, FLOPs (G)) on CIFAR100 and ImageNet1000. FLOPs start/final represent the FLOPs before/after 10 incremental learning tasks, respectively.

References

[1] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 1

[2] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 1

[3] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9285–9295, 2022. 1

[4] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 1

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

[6] Xiaofeng Mao, Gege Qi, Yuefeng Chen, Xiaodan Li, Ranjie Duan, Shaokai Ye, Yuan He, and Hui Xue. Towards robust vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12042–12051, 2022. 1

[7] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 1

[8] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. *arXiv preprint arXiv:2204.04662*, 2022. 1

[9] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. *arXiv preprint arXiv:1905.13260*, 2019. 1

[10] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2021. 1

[11] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217, 2020. 1