DyTox: Transformers for Continual Learning with DYnamic TOken eXpansion: Supplementary Materials

Arthur Douillard[1,2], Alexandre Ramé[1], Guillaume Couairon[1,3], Matthieu Cord[1,4]
[1]Sorbonne Université, [2]Heuritech, [3]Meta AI, [4]valeo.ai
arthur.douillard@heuritech.com, {alexandre.rame, matthieu.cord}@sorbonne-universite.fr, gcouairon@fb.com

Symbol	Meaning
(x_i^t, y_i^t)	Input sample & its label from the t^{th} task
C^t	Label set of the t^{th} task
$C^{1:t}$	All labels from all seen tasks
$\boldsymbol{\theta}_t$	Task token of the t^{th} task
Clf_t	Independent classifier of the t^{th} task
SAB_l	l^{th} Self-Attention Block
TAB	Task-Attention Block

Table 1: **Notations** used in the paper.

A. Appendix

Table 1 summarizes the notations used along this paper.

A.1. Experimental details

Datasets We use three datasets: CIFAR100 [15], ImageNet100, and ImageNet1000 [4]. CIFAR100 is made of 50,000 train RGB images and 10,000 test RGB images of size 32×32 for 100 classes. ImageNet1000 contains 1.2 million RGB train images and 50,000 validation RGB images of size 224×224 for 1000 classes. ImageNet100 is a subset of 100 classes from ImageNet1000. We follow PODNet [6] and DER [28] and use the same 100 classes they've used. Fine details about the datasets, like the class orders, can be found in the provided code in the options files (see readme).

Implementation For all datasets, we train the model for 500 epochs per task with Adam [13] with a learning rate of $5e^{-4}$, including 5 epochs of warmup. Following UCIR [11], PODNet [6], and DER [28], at the end of each task (except the first) we finetuned our model for 20 epochs with a learning rate of $5e^{-5}$ on a balanced dataset. In DyTox, we applied the standard data augmentation of DeiT [25] but we removed the pixel erasing [32], MixUp [30], and CutMix [29] augmentations for fair comparison. In contrast, in DyTox+ we used a MixUp [30] with beta distribution $\beta(0.8, 0.8)$. During all incremental tasks ($t > 1$), the old classifiers Clf_i, $i < t$ and the old task tokens $\boldsymbol{\theta}_i$, $i < t$ parameters are frozen. During the finetuning phase where classes are rebalanced [2, 11, 6, 28], these parameters are optimized, but the SABs are frozen.

Hyperparameter	Range	Chosen value
Learning rate	$1e^{-3}$, $5e^{-4}$, $1e^{-4}$	$5e^{-4}$
Epochs	300, 500, 700	500
λ	0.05, 0.1, 0.5	0.1
CIFAR's patch size	4, 8, 16	4
ImageNet's patch size	14, 16	16

Table 2: **Hyperparameters** that were tuned from the codebase of [25]. We ran a gridsearch on CIFAR100 10 steps on a validation set made of 10% of the training set, and kept fixed the chosen hyperparameters for all experiments (any number of steps and any datasets).

Hyperparameter tuning In contrast with previous works [6, 28], we wanted stable hyperparameters, tuned for a single setting and then applied on all experiments. This avoids optimizing for the number of tasks, which defeats the purpose of continual learning [7]. We tuned hyperparameters for DyTox using a validation subset made of 10% of the training dataset, and this only on the CIFAR100 experiment with 10 steps. We provide in Table 2 the chosen hyperparameters. Results in the main paper shows that those hyperparameters reach state-of-the-art on all other settings and notably on ImageNet.

Baselines E2E [2] and Simple-DER [18] results come from their respective papers. All other baseline results are taken from the DER paper [28]. We now further describe their contributions. iCaRL [23] uses a knowledge distillation loss [10] and at test-time predicts using a k-NN from its features space. E2E [2] learns a model with knowledge distillation and applies a finetuning after each step. UCIR [11]

TAB parameter sharing?	#P	Avg	Last
✗	97.59	72.20	56.00
✓	10.77	70.20	52.34

Table 3: **Investigation of the parameter sharing of TAB**. We report the "Avg" accuracy and the "Last" accuracy for the 50 steps setting on CIFAR100. The second row corresponds to DyTox.

uses cosine classifier and euclidean distance between the final flattened features as a distillation loss. BiC [27] uses a knowledge distillation loss and also re-calibrates [9] the logits of the new classes using a simple linear model trained on validation data. WA [31] uses a knowledge distillation loss and re-weights at each epoch the classifier weights associated to new classes so that they have the same average norm as the classifier weights of the old classes. PODNet [6] uses a cosine classifier and a specific distillation loss (POD) applied at multiple intermediary features of the ResNet backbone. RPSNet [21] uses knowledge distillation and manipulates subnetworks in its architecture, following the lottery ticket hypothesis [8]. DER [28] creates a new ResNet per task. All ResNets' embeddings are concatenated and fed to a unique classifier. ResNets are pruned using HAT [24] masking procedure. Note that DER pruning has multiple hyperparameters that are set differently according to the settings. Furthermore, the reported parameters count, after pruning, in [28] is an average of the count over all steps: the final parameters count (necessarily higher) wasn't available. Finally, Simple-DER [18] is similar to DER, with a simpler pruning method which doesn't require any hyperparameter tuning.

A.2. Parameter sharing of the TAB

Previous dynamic methods as DER [28] and Simple-DER [18] shared no parameters between tasks until the final classifier. We proposed instead with DyTox to share the encoder (SABs) and the decoder (TAB) parameters across tasks, leading to a minimal memory overhead while also sharing common information between tasks. In Table 3, we compare the impact of sharing the TAB per task — and only maintain different tokens per task. In the first row, a different TAB is created per task, while in the second row the same TAB is used — which is the DyTox strategy. A different TAB per task leads to better results (56% *v.s.* 52% in "Last" accuracy) because the network can be more diverse with each TAB specialized to its associated task. This increased diversity has a drawback: the memory overhead is too important (97M *v.s.* 10M parameters). We find in practice that DyTox strikes a good balance between memory overhead and continual performance.

A.3. Novel continual training procedure

DyTox++ We nicknamed DyTox+ our model when combined with a novel continual procedure based on MixUp [30]. We now refine DyTox+ into DyTox++ by adding a new component during the training: the Sharpness-Aware Minimizer (SAM) [16]. Indeed, **aiming for wider minima** is particularly important in continual learning [14, 26]. This is because sharp task-specific minima lead to overspecialization to a particular task and consequently to a forgetting of all other tasks. Weights constraints as EWC [14] or second-order optimization [17] have similar motivations. SAM estimates the worst closest parameters during a first forward/backward pass, and then optimizes the loss w.r.t. to them during a second forward/pass. In consequence, DyTox++ optimizes the loss not w.r.t. the current parameters but w.r.t. a region of possible parameters leading to wide local minima that span across multiple tasks. In practice, we used the Adaptive SAM (ASAM) [16], an extension of SAM that is more robust to hyperparameters.

DyTox+ and DyTox++ experiments The computational overhead of ASAM is lower than more complex second-order methods, but it still doubles the number of forward and backward passes. For this reason, we didn't include it in our main experiment but propose in Table 4 and Table 5 experiments on CIFAR100 and ImageNet100. The gain provided by MixUp then ASAM on our model (DyTox++) leads to a consistent improvement of +4.7% in "Avg" compared to the previous State-of-the-Art DER [28] on CIFAR100 50 steps (Table 4 and +4.6% on ImageNet100 10 steps (Table 5). Future works could consider the promising Look-SAM [19] to reduce the time overhead.

Training procedure introspection In Table 6, we compare DyTox+ and DyTox++ on CIFAR100 in a joint setting (no continual) and in a continual setting with 50 steps. In the joint setting, our model slightly benefits from both MixUp and ASAM: the gain is limited (+1.79 *p.p.*). On the other hand, those two methods greatly improve the extreme continual setting of 50 steps (+6.42 *p.p.*). This shows that the gain is not due to absolute improvements of the model performance. Moreover, using the Chaudrhy et al.'s forgetting [3] measure, we compare how much a model has forgotten relatively to its previous tasks. This metric is therefore agnostic to absolute performance improvements. DyTox had a forgetting of 33.15%, DyTox+ of 31.50%, and DyTox++ of 30.47%: a total reduction of 2.68 *p.p*. This validates our novel training procedures that are particularly efficient for continual learning.

Methods	10 steps #P	Avg	Last	20 steps #P	Avg	Last	50 steps #P	Avg	Last
ResNet18 Joint	11.22	-	80.41	11.22	-	81.49	11.22	-	81.74
Transf. Joint	10.72	-	76.12	10.72	-	76.12	10.72	-	76.12
WA [31]	11.22	69.46 $\pm$ 0.29	53.78	11.22	67.33 $\pm$ 0.15	47.31	11.22	64.32 $\pm$ 0.28	42.14
DER w/o P [28]	112.27	75.36 $\pm$ 0.36	**65.22**	224.55	74.09 $\pm$ 0.33	**62.48**	561.39	72.41 $\pm$ 0.36	**59.08**
DyTox	10.73	73.66 $\pm$ 0.02	60.67 $\pm$ 0.34	10.74	72.27 $\pm$ 0.18	56.32 $\pm$ 0.61	10.77	70.20 $\pm$ 0.16	52.34 $\pm$ 0.26
DyTox+	10.73	75.54 $\pm$ 0.10	62.06 $\pm$ 0.25	10.74	75.04 $\pm$ 0.11	60.03 $\pm$ 0.45	10.77	74.35 $\pm$ 0.05	57.09 $\pm$ 0.13
DyTox++	10.73	**77.10 $\pm$ 0.08**	64.53 $\pm$ 0.08	10.74	**76.57 $\pm$ 0.18**	**62.44 $\pm$ 0.22**	10.77	**75.45 $\pm$ 0.19**	58.76 $\pm$ 0.28

Table 4: **Results on CIFAR100** averaged over three different class orders. WA and DER w/o P results are reported from [28]. DyTox+ uses MixUp in addition of the DyTox strategy, DyTox++ further adds a sharpness-aware minimization [16].

Methods	#P	top-1 Avg	top-1 Last	top-5 Avg	top-5 Last
ResNet18 joint	11.22	-	-	-	95.1
Transf. joint	11.00	-	79.12	-	93.48
WA [31]	11.22	-	-	91.00	84.10
DER w/o P [28]	112.27	77.18	66.70	93.23	87.52
DyTox	11.01	77.15	69.10	92.04	87.98
DyTox+	11.01	79.22	69.06	93.72	88.82
DyTox++	11.01	**80.76**	**72.46**	**94.40**	**90.10**

Table 5: **Results on ImageNet-100** with 10 steps of 10 new classes each. WA and DER w/o P results are reported from [28]. DyTox+ uses MixUp in addition of the DyTox strategy, DyTox++ further adds a sharpness-aware minimizer.

Training	Joint (1 step) Last (↑)	50 steps Last (↑)	50 steps Forgetting (↓)
DyTox	76.12	52.34	33.15
DyTox+	77.51 +1.39	57.09 +4.75	31.50 -1.65
DyTox++	77.91 +0.40	58.76 +1.67	30.47 -1.03

Table 6: **"Last" accuracy and forgetting** [3] on CIFAR100 for the joint (1 step, no continual) and 50 steps settings.

A.4. Patch size effect on forgetting

Our model is the first application of transformers for continual computer vision. A key component of the transformer architecture is the patch tokenizer. The number of patch tokens in an image is determined by the patch size: a larger patch size means less tokens, and vice-versa. We wondered about the effect of the patch size on forgetting and tested three different kind of patch sizes in Table 7. Echoing results in vision transformers [5, 25], a smaller patch size (4 vs. 8 and 16) performs best in a joint training. However, the forgetting defined by Chaudhry et al. [3] is relatively similar, with 33.15% for a patch of size of 4, and 33.20% for a patch size of 16. Therefore, we argue that the transformer architecture is hardly sensitive to the patch resolution w.r.t its forgetting in continual learning.

Patch size	Joint (1 steps) Last (↑)	50 steps Last (↑)	50 steps Forgetting (↓)
4	76.12	52.34	33.15
8	67.65	43.93	35.44
16	50.15	31.49	33.20

Table 7: **Patch size effect on continual** for the joint (1 step, no continual) and 50 steps settings on CIFAR100. We choose a patch size of 4 for our main experiments: yet, it has only few impact on forgetting.

A.5. ResNet backbone

DyTox is made of two main components: the SABs and the unique TAB. The TAB structure, taking in input both patch tokens and a task token, is crucial to our strategy. Yet, the SAB could be of any kind of features extractor, based on convolutions or attentions. Following the hybrid network proposed in ablations by Dosovitskiy et al. [5], we tried to replace the collection of SABs by a ResNet18. The final features of the ResNet, before global pooling, of shape $(W \times H \times D)$ can be seen as $W \times H$ tokens of D dimension. We made a few modifications to this ResNet to boost its performance, namely removed the fourth and ultimate layer, and added a pointwise convolution with 504 output channels (so it can be divided by the 12 attention heads of the TAB), a batch normalization [12], and a ReLU activation. These simple modifications are sufficient for our proof of concept, and thus we also didn't tune deeply this model. We display in Table 8 the comparison of the two backbones on CIFAR100 50 steps: (1) with ResNet, and (2) with SABs (DyTox). Performances are slightly lower than DyTox with SABs, however, they are still significantly higher than previous state-of-the-art like WA [31], especially in "Last" accuracy. Moreover, the parameters count is comparable to DyTox. This experiment shows that our DyTox framework, while designed with a transformer backbone in mind, is also efficient on non-token-based architectures such as a ResNet.

Encoder	**#P**	**Avg**	**Last**
ResNet	10.68	68.53	50.05
SABs	10.77	70.20	52.34

Table 8: **Hybrid network** on CIFAR100 50 steps. While the features extractor is made of SABs in DyTox, here we instead use a modified ResNet18. Our framework still works well with a convolution-based approach.

	CIFAR100 Top-1		ImageNet100 Top-5	
Task decoder	Avg	Last	Avg	Last
Residual Adapters [22]	70.00	52.38	91.25	85.00
FiLM [20]	69.42	54.05	89.49	81.40
TAB *(ours)*	70.20	52.34	92.04	87.98

Table 9: **Alternative task conditioner** on CIFAR100 50 steps and ImageNet100 10 steps. While the simpler Residual Adapters and FiLM perform similarly to our TAB on CIFAR100, they forget significantly more on the complex ImageNet100.

A.6. Alternative task decoders

We investigate here other approaches for *conditioning* features to the different tasks. Residual Adapters [22] adds a different residual branch made of a pointwise convolution for each domain the model is learned (*e.g.* CIFAR then ImageNet then SVHN). This model needs the task/dataset/-domain identifier at test-time to determine which residual to use. For VQA task [1], FiLM [20] proposes to modify the visual features using the the textual query.

We adapt these two feature conditioning strategies for our transformer backbone architecture. We perform a global token pooling after the ultimate SAB, and apply for each learned task, a residual adapter or a FiLM. Residual adapter in our case is a MLP, and FiLM parameters are directly learned. As for DyTox, we forward each task-specific embedding to the respective task-specific classifier. We showcase the continual performance in Table 9 on CIFAR100 50 steps and ImageNet100 10 steps. On the former dataset, smaller and easier, the residual adapters and FiLM have similar performance as our TAB approach. On the other hand, as soon as the task complexity increases with the more detailed ImageNet100 dataset, FiLM and Residual adapter based conditioning strategies forget significantly more than our complete DyTox framework: TAB outperform the Residual Adapters by +2.98 *p.p* in "Last" top-5 accuracy and FiLM by +6.58 *p.p* .

References

[1] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015. (page 4).

[2] Francisco M. Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, 2018. (page 1).

[3] Arslan Chaudhry, Puneet Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, 2018. (pages 2, 3).

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. (page 1).

[5] 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 *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. (page 3).

[6] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, 2020. (pages 1, 2).

[7] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. In *International Conference on Machine Learning (ICML) Workshop*, 2018. (page 1).

[8] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019. (page 2).

[9] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, 2017. (page 2).

[10] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *Advances in Neural Information Processing Systems (NeurIPS) Workshop*, 2015. (page 1).

[11] 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 (CVPR)*, 2019. (page 1).

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015. (page 3).

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the Inter-*

national Conference on Learning Representations (ICLR)*, 2014. (page 1).

[14] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017. (page 2).

[15] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report*, 2009. (page 1).

[16] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning (ICML)*, 2021. (pages 2, 3).

[17] Janghyeon Lee, Hyeong Gwon Hong, Donggyu Joo, and Junmo Kim. Continual learning with extended kronecker-factored approximate curvature. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. (page 2).

[18] Zhuoyun Li, Changhong Zhong, Sijia Liu, Ruixuan Wang, and Wei-Shi Zheng. Preserving earlier knowledge in continual learning with the help of all previous feature extractors. In *arXiv preprint library*, 2021. (pages 1, 2).

[19] Yong Liu, Siqi Mai, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Sharpness-aware minimization in large-batch training: Training vision transformer in minutes. In *OpenReview*, 2021. (page 2).

[20] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018. (page 4).

[21] Jathushan Rajasegaran, Munawar Hayat, Salman Khan, Fahad Shahbaz Khan, Ling Shao, and Ming-Hsuan Yang. An adaptive random path selection approach for incremental learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. (page 2).

[22] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. (page 4).

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

[24] Joan Serrà, Dídac Surís, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning (ICML)*, 2018. (page 2).

[25] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers and distillation through attention. In *International Conference on Machine Learning (ICML)*, 2021. (pages 1, 3).

[26] Eli Verwimp, Matthias De Lange, and Tinne Tuytelaars. Rehearsal revealed: The limits and merits of revisiting samples in continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop*, 2021. (page 2).

[27] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. (page 2).

[28] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. (pages 1, 2, 3).

[29] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. (page 1).

[30] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018. (pages 1, 2).

[31] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shutao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. (pages 2, 3).

[32] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2017. (page 1).