arXiv:2211.12292v1 [cs.CV] 22 Nov 2022

Gated Class-Attention with Cascaded Feature Drift Compensation for Exemplar-free Continual Learning of Vision Transformers

Marco Cotogni[1] , Fei Yang[2], Claudio Cusano[1], Andrew D. Bagdanov[3] , Joost van de Weijer[2]
[1] Dept. of Electrical, Computer and Biomedical Engineering, University of Pavia, Pavia, Italy
[2]Computer Vision Center, Universitat Autonoma de Barcelona, Barcelona, Spain
[3]MICC, University of Florence Florence 50134, FI, Italy
marco.cotogni01@universitadipavia.it, {fyang, joost}@cvc.uab.es
claudio.cusano@unipv.it, andrew.bagdanov@unifi.it

Abstract

In this paper we propose a new method for exemplar-free class incremental training of ViTs. The main challenge of exemplar-free continual learning is maintaining plasticity of the learner without causing catastrophic forgetting of previously learned tasks. This is often achieved via exemplar replay which can help recalibrate previous task classifiers to the feature drift which occurs when learning new tasks. Exemplar replay, however, comes at the cost of retaining samples from previous tasks which for some applications may not be possible. To address the problem of continual ViT training, we first propose gated class-attention *to minimize the drift in the final ViT transformer block. This mask-based gating is applied to class-attention mechanism of the last transformer block and strongly regulates the weights crucial for previous tasks. Secondly, we propose a new method of* feature drift compensation *that accommodates feature drift in the backbone when learning new tasks. The combination of gated class-attention and cascaded feature drift compensation allows for plasticity towards new tasks while limiting forgetting of previous ones. Extensive experiments performed on CIFAR-100, Tiny-ImageNet and ImageNet100 demonstrate that our method outperforms existing exemplar-free state-of-the-art methods without the need to store any representative exemplars of past tasks. Code is available here: https://github.com/OcraM17/GCAB-CFDC*

1. Introduction

The initial excellent results of transformers for language tasks [49] have encouraged its application also for vision applications [12]. Vision Transformers (ViTs) currently achieve excellent results for many applications [6, 29, 44]. Most existing work on ViT training assumes that all training data is jointly available, an assumption which does not hold for real-world applications in which data arrives in a sequence of non-overlapping tasks. Continual learning considers learning from a non-IID stream of data. Applying a naive finetuning approach to such data results in a phenomenon called *catastrophic forgetting* which results in a drastic drop in performance on previous tasks [18]. The main goal of continual learning algorithms is to maximize the stability-plasticity trade-off [35], i.e. to mitigate forgetting of previously learned classes while maintaining the plasticity required to learn new ones.

One of the most successful approaches to preventing forgetting of previous tasks is *exemplar rehearsal* in which a subset of images from previous tasks is stored and then rehearsed when learning new ones [2, 4, 5, 9, 30, 38, 39]. Because of its success, the rehearsal technique has also been adopted by the initial works on continual learning for ViTs [14, 51]. However, for many applications the storage of previous task data might not be possible. This is especially true for applications with strict memory constraints and those where privacy or data use legislation prevents the long-term storage of data. In order to overcome this limitation, exemplar-free methods have been investigated [23, 27, 54, 55]. These methods do not store any data from previous tasks, however their application to continual learning of ViTs has not been fully explored.

In this paper we propose one of the first exemplar-free ViT-based methods for class-incremental learning (CIL). One of the challenges of exemplar-free continual learning is that these models tend to forget previously learned features while they are learning features for new tasks. The architecture we propose is based on a **gated class-attention** mechanism applied to the ViT decoder in order to mitigate the forgetting of learned features. As proposed by Serra et al. [42], mask-based gating mechanisms are usually associated with the task-incremental scenario (i.e. scenarios in which a task

id is known at inference time). We propose a solution to overcome this limitation by applying the masking mechanism only on the transformer decoder via multiple forward passes. This solution allows us to use mask-based gating in a class-incremental setup.

Mask-based gating prevents the drift of weights in the decoder, however it does not mitigate the drift in the transformer encoder. In fact, learning a stable backbone in the exemplar-free scenario is very difficult due to the drift in encoder weights that occurs during the learning of new tasks. To address this, we propose a method for *backbone regularization in combination with a feature drift compensation* mechanism that uses a cascade of projection networks that map the current backbone features to those of the previous backbone. This allows increased plasticity while maintaining stability across tasks, incurring a small computational cost due to the feature projection cascade. We also show, however, that knowledge distillation can be used to alleviate the computational burden of the projection cascade and the need for multiple forward passes at inference time.

The main contribution of this work are:

- a gated class-attention mechanism inspired by [42], called GCAB, that mitigates weight drift in the transformer decoder while also overcoming the need for a task id at inference time;
- a novel method for backbone regularization and feature drift compensation that increases plasticity towards new classes while maintaining the stability of previously learned ones;
- a method for GCAB distillation that reduces the computational overhead due to multiple forward passes and the memory overhead of storing projection networks; and
- experiments on multiple benchmarks demonstrate that our exemplar-free approach achieves state-of-the-art performance compared to other exemplar-free methods and outperforms recent continual learning methods developed for ViT by a large margin even when these are equipped with a small memory of exemplars.

2. Related Work

Continual Learning. Continual learning algorithms can be grouped in three categories [33]: regularization approaches, parameter based regularization [23, 26, 28, 56], and data based regularization [7, 11, 21, 22, 27, 53, 58]; rehearsal approaches, which store [8, 38] or generate exemplars [50, 57]; and bias-correction approaches [7, 21, 53].

Continual Learning with VITs. Visual Transformers recently outperforms convolutional neural networks and in particular resnet [19] in several tasks like classification [12] or segmentation [59]. Although ViTs are considered state-of-the-art models, their application in continual learning has not been fully explored. Douillard et al. [14] proposed a transformer-based architecture, called DyTox, with a dynamic task-token expansion for mitigating catastrophic forgetting. Wang et al. [51] proposed an inter-task attention mechanism for ViTs. Wang et al. [52] described a prompting method for continually learning a classifier using a pretrained, frozen ViT backbone. Even if the performances showed in these work are very remarkable, the challenge of continually learning the parameters of a ViT without storing exemplars or using a pretrained model, is still open. Differently from previous work, we propose an exemplar-free ViT approach to class-incremental learning. Our work is inspired by DyToX [14], which applies a task conditioned class-attention block. However, DyToX shares the class-attention block parameters between tasks which leads to forgetting, and it therefore requires exemplars to counter this. Instead, we replace the task token with a task-specific gating function that prevents forgetting and does not require exemplars. In addition, we introduce feature drift compensation, which allows for more plasticity in the backbone.

Parameter Isolation in Continual Learning. In this family of algorithms, learnable masks are applied to the weights of the model in order to reduce forgetting. Mallaya et al. [31] proposed Piggyback, a masked-based method able to learn the weight masks while training a backbone. The same group proposed Packnet [32] which, via iterative pruning and sequential re-training, is able to add multiple tasks to a single network. Serra et al. [42] proposed to apply masks to layer activations in order to limit the update of the parameters more relevant to a specific task. Masana et al. [34] proposed a system of ternary-masks applied on to layer activations for preventing catastrophic forgetting and backward transfer. Yan et al. [54] proposed a Dynamical Expandable Representation (DER) for continual learning. In this work, channel-level masks are used for pruning the feature extractor. Rajasegaran et al. [37] proposed Random Path Selection (RPS). This approach uses a parameter isolation mechanism, distillation, and a replay-buffer to learn different paths for each task without the need for a task id during inference.

Exemplar-Free Continual Learning. This is one of the most challenging scenario in continual learning. In this paradigm, it is not possible to store any exemplars from the previously observed classes. Li et al. [27] proposed an exemplar-free data regularization approach to mitigate forgetting. This method distills knowledge of the previous model into the new one in order to prevent weight drift while learning the new task data. Kirkpatrick et al. [23] described an exemplar-free weight regularization approach called Elastic Weight Consolidation (EWC) for preventing weight drift. Similarly, Aljundi et al. [1] proposed a method

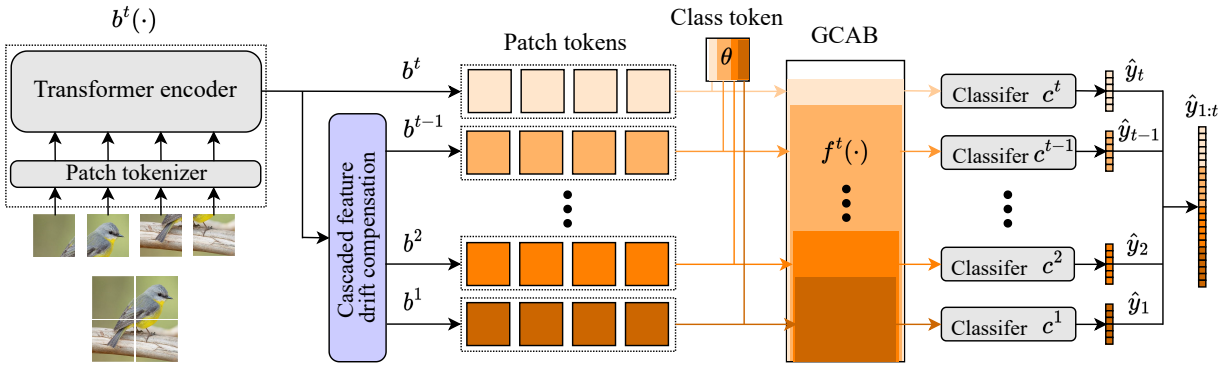

Figure 1. Overview of the architecture. Our main contributions are the *gated class-attention block (GCAB)* to prevent forgetting in the final ViT block (Section 3.2) and the *cascaded feature drift compensation* to compensate for feature drift of the backbone network (Section 3.5).

that accumulates the importance of each model parameter by analyzing the effect of their change to the predicted output. Yu et al. [55] proposed a semantic drift compensation mechanism to compensate for feature drift in previous tasks by approximating it with the drift estimated with current task data. Toldo et al. [45] presented a framework for modeling the semantic drift of model weights of and estimating feature drift in the representation of previously learned classes. Pelosin et al. [36] proposed an attention distillation mechanism for exemplar-free visual transformer in task-incremental learning.

3. Method

In this section, we propose our approach to exemplar-free ViT-based class-incremental learning. We begin by introducing the problem setup for class-incremental learning and the transformer architecture used in subsequent sections to perform exemplar-free class-incremental learning with Vision Transformers.

3.1. Problem setup

Here we define the class-incremental learning setup and the specific Vision Transformer architecture we use.

Class-incremental learning setup. In class-incremental learning the model must learn a sequence of T tasks, where each task t introduces a number of new classes C^t. The data D^t of task t contains samples (x_i, y_i), where x_i is input data labeled by $y_i \in C^t$. Note that we consider the case in which that there is no overlap between different task label sets: $C^i \cap C^j = \emptyset$ if $i \neq j$, as is commonly assumed [33]. The model is evaluated on all previously seen classes $C^{\leq t} = \cup_{t' \leq t} C^{t'}$. Class-incremental learning differs from task-incremental learning in that it has no access to the task label t at inference time, and is therefore considered a more challenging setting [10, 47]. Furthermore, in this paper, we consider the more restrictive setup of exemplar-free class-incremental learning in which no data from previous tasks is saved.

Transformer architecture. We use a vision transformer based on the one proposed by [12] and the recent improvements of [46]. It consists of transformer encoder and decoder, each built with several multi-head attention blocks. In Figure 1 we give a schematic diagram of our architecture. Formally, the input image $x \in \mathbb{R}^{H \times W \times C}$ is passed through a patch tokenizer that splits x into N patches and projects them using a 2D convolutional layer to obtain a set of N patch tokens $x_0 \in \mathbb{R}^{N \times D}$. A learnable position embedding is added to the patch tokens as in [16]. The patch tokens x_0 are passed as input to a sequence of M transformer encoder blocks, each yielding tensors of the same dimensions. Each block is composed of a multi-head self-attention (SA) mechanism [49], layer normalization and a Multi-layer Perceptron (MLP), each with residual connections:

$$\begin{aligned} x'_l &= x_l + \mathrm{SA}(x_l) \\ x_{l+1} &= x'_l + \mathrm{MLP}(x'_l) \end{aligned} \tag{1}$$

We follow the design of CaiT [46], and only insert a class token in combination with a class-attention layer in the last block of the decoder. In our solution, the transformer decoder is composed of one single block. To distinguish the various parts of the transformer network, we define the image output prediction $\hat{y} = c(f(b(x; \Psi)))$, where the backbone features $b(x; \Psi) \in \mathbb{R}^{(N+1) \times D}$ parameterized by Ψ are the output of the self-attention blocks, and $f(b(x; \Psi))$ refers to the feature output of the decoder before classifier c.

3.2. Gated class-attention

Parameter isolation methods work by isolating a limited set of parameters after learning each task [10, 32, 41]. We build upon the work of *Hard Attention to the Task* (HAT) [42] in which attention masks are learned for each task. The masks operate on the activations of the network. Parameters of the network used by previous tasks can be exploited by new tasks, thereby allowing for forward transfer,

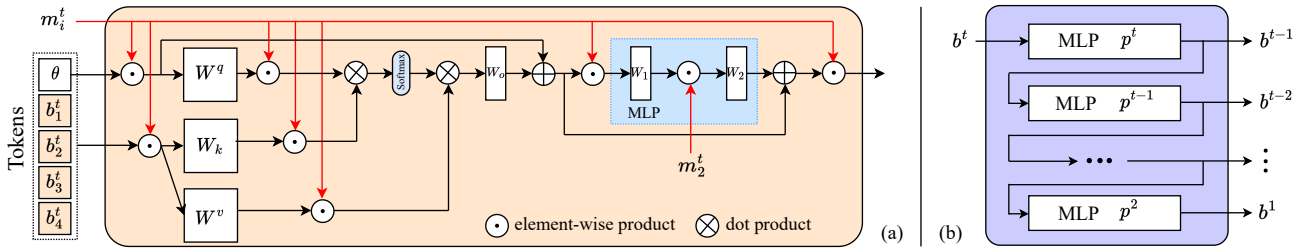

Figure 2. (a) GCAB applies masks to the activations of the class-attention block. This allows prevention of forgetting of previous tasks. (b) The cascade of feature projection networks maps the current backbone features to previous task features to compensate for feature drift.

but their update is restricted to prevent forgetting. The main strengths of this approach are the good forward transfer with little or no forgetting of previous tasks, together with the ability to automatically learn which neurons to dedicate to each task within the capacity limit of the neural network.

The forward pass of parameter isolation methods is usually conditioned on the task, and is therefore restricted to task-incremental learning. A possible way to extend these methods to class-incremental learning would be to run one forward pass for each task and then combine the task predictions (e.g., by concatenation). However, this would increase the run time linearly in the number of tasks. To limit computational overhead, we propose to only apply attention masks to the last block of the ViT. In Section 3.5, we investigate distillation to further reduce computational overhead.

Mask-based class-attention gating. We apply the attention-gating in the last transformer block, i.e. the decoder, which contains class-attention [46]. This block combines the patch tokens from previous blocks with a learnable class token θ. In Figure 2 (left) we give a schematic diagram of the gated class-attention block. We define a number of learnable masks m for task t as:

$$m^t = \sigma(sAt^{\mathrm{T}}), \tag{2}$$

where, slightly abusing notation, t represents both a task *index* and a *one-hot vector* identifying the current task, A is a learnable embedding matrix, σ is the sigmoid activation function, and s is a positive scaling parameter. Here m^t refers to the neurons that have been selected for task t.

We can now define the gated forward pass through the final class-attention block by introducing a mask for all the activations contributing to the final class token output. These masks correspond to: the input tokens (m_i^t), queries and keys (m_{QK}^t), values (m_V^t), the MLP (m_1 and m_2), and the class-attention output (m_o). We can then compute the query (Q), key (K), value (V), attention (A) and self attention output (O) given the block token inputs O^t and these masks:

$$\begin{aligned}
&p = [\theta, b] \in \mathbb{R}^{(N+1)\times D} \\
&Q^t = W_q(\theta \odot m_i^t)\ ,\ K^t = W_k(p \odot m_i^t)\ ,\ V^t = W_v(p \odot m_i^t) \\
&A^t = \text{softmax}\left\{\left(\left(Q^t \odot m_{QK}^t\right)\left(K^t \odot m_{QK}^t\right)^T\right)/\sqrt{d/h}\right\} \\
&O^t = W_o A^t (V^t \odot m_V^t).
\end{aligned} \tag{3}$$

The gating mechanism is also applied to the MLP (see Eq. 1) according to:

$$\begin{aligned}
b'^t &= O^t + \theta \\
u^t = W_1(b'^t \odot m_1^t),\ \ v^t &= W_2(u^t \odot m_2^t) \\
f^t &= v^t + O^t
\end{aligned} \tag{4}$$

We call this the Gated Class-Attention Block (GCAB). The masks are learned during the training of task t and their role is twofold: they select those activations that are used to compute the task-conditioned output and they restrict the backpropagation of future tasks, preventing changes to weights that used by previous tasks. Given that both m_i^t and m_1^t operate on the class token embedding θ, we set $m_1^t = m_i^t$. In the experimental section, we show that good results are obtained when sharing the weights and setting $m_{QK} = m_V = m_1 = m_o = m_i$, resulting in only two learnable masks m_i and m_2. In Figure 2 (left) we show the interaction between the masks m_i and m_2 and the class-attention block.

For each task a dedicated classifier c^t is added which produces predictions $\hat{y}^t = W_{clf}(f^t \odot m_o^t)$ using the vector f^t output from the GCAB. We perform multiple forward passes of patches extracted from the backbone b^t through the decoder f^t (i.e. we pass it t times). For each pass s, the obtained vectors l^s are then passed to the corresponding classifier c^s. The classifier outputs are then concatenated $C = [c^1, \cdots, c^t]$ and the binary cross entropy loss $\mathcal{L}_{\text{BCE}}$ is computed using the target vector.

Training. During the training of the current task t the scaling parameter s from equation 2, is scaled with the batch index: $s = \frac{1}{s_{max}} + (s_{max} - \frac{1}{s_{max}})\frac{i-1}{I-1}$, where i current batch index and I is the total number of batches in an epoch. This was found to be beneficial in [42].

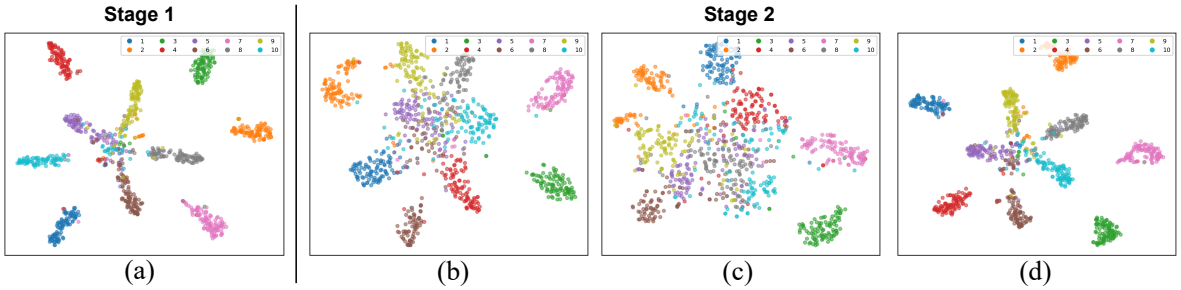

Figure 3. t-SNE visualization of the output of the backbone $f(\cdot)$ after the first task (stage 1) and second task (stage 2). These visualizations are based on the task-agnostic embeddings in the 10 task scenario on CIFAR-100. (a) Results after Stage 1. Results after Stage 2 (b) without PFR during training, (c) with PFR during training but without feature drift compensation, and (d) with PFR during training and with feature drift compensation. More t-SNE results are provided in the Supplementary Material.

At the end of the current task, the learned masks are accumulated as $m_*^{<t} = \max(m_*^{t-1}, m_*^{<t-1})$, where '$*$' stands for any of the specific mask subscripts introduced above. Accumulated masks $m_*^{<t}$ are used during backpropagation to prevent updating weights considered important for the tasks observed so far. The masks are learned by minimizing the following loss function:

$$\mathcal{L}_{\text{GCAB}} = \lambda_{\text{GCAB}} \frac{\sum_x m_x^t (1 - m_x^{<t})}{\sum_x 1 - m_x^{<t}}, \tag{5}$$

where m_x^t is the mask learned at the current task t for component x of the GCAB, x ranges over the mask subscripts described above, and $m_x^{<t}$ is the cumulative mask. λ_{GCAB} is a tunable hyperparameter controlling the capacity of the masks learned during the tasks. This equation encourages new task mask m_x^t to be sparse, however it permits use of activations already used by previous tasks $m_x^{<t}$ at no cost.

The cumulative masks also play a pivotal role during the training of new tasks. Consider, for example, weights W_q that map from the input tokens (masked by m_i^t) to the queries (masked by m_{QK}^t). We then define the elements of the weight mask according to $M_{q,kl}^{<t} = 1 - \min(m_{i,k}^{<t}, m_{QK,l}^{<t})$ where $m_{i,k}^{<t}$ refers to the k-th element of $m_i^{<t}$. The update rule for the backpropagation of the gradient is then: $W_q = W_q - \lambda M_q^{<t} \odot \frac{\partial L}{\partial W_q}$. This update rule prevents the updating of part of the weights learned for previous tasks. The input mask also influeces the updating of the class token embedding which is given by $W_\theta = W_\theta - \lambda(1 - m_i^{<t}) \odot \frac{\partial L}{\partial W_\theta}$. In the Supplementary Material we show the update rules for all weight matrices in the class-attention block.

3.3. Backbone Regularization and Cascaded Feature Drift Compensation

In the previous section we applied gating only to the last transformer block to limit computational overhead. Gating ensures that only minimal changes occur to the weights relevant to previous tasks, however the network can still suffer from forgetting due to backbone feature drift. To mitigate forgetting in the backbone, we apply regularization to these features.

A straightforward way to prevent forgetting in the backbone network via regularization is feature distillation [21]. Feature distillation encourages backbone features at task t to remain close to those at task t-1, however, this was found to limit plasticity [13]. To ensure stability without sacrificing plasticity, some recent works in continual learning of self-supervised representations have proposed to learn a projector between feature extractors [15, 17]. This approach, called Projected Functional Regularization (PFR), introduces a projection network p^t that maps the current backbone features to those of the previous backbone. PFR allows the new backbone to learn new features without imposing a high regularization penalty as long as the new features can still be projected back to those of the previous backbone. The PFR loss function is:

$$\mathcal{L}_{\text{pfr}} = \lambda_{\text{pfr}} \mathbb{E}_{x \sim \mathcal{D}^t} \left[\mathcal{S}\left(p^t\left(b\left(x; \Psi^t\right)\right), b\left(x; \Psi^{t-1}\right)\right)\right] \tag{6}$$

where $\mathcal{S}$ is the cosine distance, λ_{pfr} is a trade-off parameter, and Ψ^t refers to the parameters of the backbone after learning task t.

The gained plasticity induced by PFR leads to a misalignment of the current backbone with previous class-attention layers and classifiers. Consider the predictions of the system after training t tasks on data of task $s \leq t$: $\hat{y} = c^t(f^t(b^t(x)))$. The gating mechanism described above ensures that $c^t(f^t(\cdot)) \approx c^s(f^s(\cdot))$, however $b^t \neq b^s$. That is, the backbone feature representation at task t has *drifted* away from the representation at previous task s. To address this we perform *cascaded feature drift compensation (FDC)* (see Fig. 2(b)) and exploit the learned projection networks p^t in a *cascade* to align the current backbone with the learned class-attention block at any previous task s:

$$\hat{y}_s = c^t(f^t(p^{s+1}(p^{\cdots}(p^{t-1}(p^t(b^t(x))))))) \tag{7}$$

All projection networks p^t must be saved in this formulation in order to compute the projection cascade that compensates

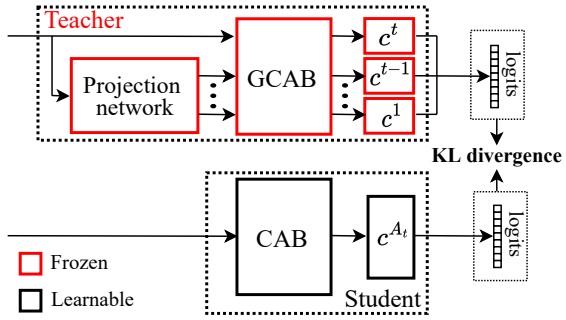

Figure 4. Illustration of GCAB distillation.

for backbone feature drift. In Section 3.5 we show how knowledge distillation can be used to eliminate the need for projection networks p^t at inference time.

To illustrate the effectiveness of cascaded feature drift compensation, we analyzed the embeddings produced by the GCAB. In Figure 3 we visualize the embedding produced by the GCAB using t-SNE [48]. The embedding of the images from the test set of the first task are shown after the training of the first task (Stage 1) and the second task (Stage 2). Only applying PFR during training results in a latent space where classes no longer align with their location after Stage 1 (see Figure 3(c)). This is problematic since the previously trained classifier of task 1 no longer aligns with them and will therefore suffer a significant drop in performance (as verified in our ablation study). After applying the cascaded feature drift compensation, the class distributions are mapped back to their original locations and align again with the classifier head (see Figure 3(d)). In conclusion, this illustration shows that cascaded feature drift compensation can be a strong tool to recover from feature drift in the backbone. This is important since it allows for high plasticity during the continual learning process.

3.4. Training Objective and Inference

The final objective we use for incremental training of the model is the sum of the three loss functions:

$$\mathcal{L} = \mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{pfr}} + \mathcal{L}_{\text{GCAB}} \tag{8}$$

After training the current task, an evaluation phase is performed over all classes seen so far. At task t, the model is tested for the tasks $t' \leq t$. All images are passed through the backbone to obtain tokens b. The tokens are passed t times through the Gated Class-Attention Block. Based on the task index of the forward pass t', the composition of the previously stored projection networks is used as in Equation 7 to align the current backbone features to those of previous task t'. In Figure 2 we give a schematic diagram of the Feature Drift Compensation mechanism during inference. During inference the parameter s of Equation 2 is equal to s_{max}.

3.5. GCAB Distillation

To overcome the increased computational cost due to multiple forward passes for all tasks and the cascaded projection layers, we perform knowledge distillation [20] to transfer the class-conditioned GCAB (the *teacher*) into a single class-attention block (CAB) with the same architecture as the GCAB but *without* masks, and an aggregated classifier c^{A_t} (*student*). This reduces the number of parameters, since the task projection networks are no longer needed, and eliminates the need for multiple forward passes. As shown in Fig. 4, the CAB and c^{A_t} trained by minimizing the Kullback–Leibler (KL) divergence between the logits output by the teacher and student models. Note that this distillation is conducted only with the data from task t, while the transformer backbone b and teacher hyper-classifier are frozen during training. We also investigate the use of static masks in the CAB to leave unused capacity in the student network in order to accommodate potential future tasks (see Section 4.3 for details).

4. Experimental Results

For our experiments, we consider three datasets: CIFAR-100 [24], Tiny-ImageNet [25] and ImageNet100 [40] (for details see the Supplementary Material). We set the number of transformer encoder blocks to $M = 5$, each one with $H = 12$ heads for the multi-head self-attention mechanism. The dimension of the embeddings is set to $D = 384$. We train each task for 500 epochs using Adam with $lr = 1e^{-4}$ and a batch size of 128. At inference time, we deactivate layer normalization in the classifier for the experiments with a larger first task, since it induces a task bias. We set hyper-parameters $\lambda_{\text{pfr}} = 0.001$, $\lambda_{\text{GCAB}} = 0.05$ and $s_{max} = 800$.

4.1. Comparison with the State-of-the-art

Equal task split scenarios. We consider two different CIL scenarios: 5 tasks and 10 tasks equally split among all classes. For CIFAR-100, tasks contain 20 classes for the 5 task scenario and 10 for the 10 task scenario. For Tiny-ImageNet and ImageNet100 we consider only the 10-task scenario, with 20 and 10 classes in each task, respectively. For CIFAR-100 and Tiny-ImageNet the images are split in $N = 64$ patches. For ImageNet100 the number of patches is $N = 196$. We report the task-agnostic top-1 accuracy over all the classes of the dataset after training the last task: $ACC_{TAG} = \frac{1}{N}\sum_{t=1}^{N} a_t$, where N is the total number of classes in the dataset. For the task-aware scenario we report the mean accuracy over all the tasks after training the last task: $ACC_{TAW} = \frac{1}{T}\sum_{t=1}^{T} a_t$, where T is the total number of tasks. We compare our approach with several methods: ER [39], AGEM [9], iCarl [38], FDR [3], DER++ [4], ERT [5], RM [2], LVT [51], DyToX [14], and A-D [36]. The memory buffer for the exemplar-based methods is lim-

Method	# Params	Exemplar-Free	CIFAR-100 5 Tasks		CIFAR-100 10 Tasks		Tiny-ImageNet		ImageNet100	
			Class-IL	Task-IL	Class-IL	Task-IL	Class-IL	Task-IL	Class-IL	Task-IL
ER [39]	11.2	✗	21.94	62.41	14.23	67.57	8.79	39.16	9.58	36.24
AGEM [9]	11.2	✗	17.97	53.55	9.44	55.04	8.28	23.79	9.27	25.20
iCaRL [38]	11.2	✗	30.12	55.70	22.38	60.81	8.64	28.41	12.59	33.75
FDR [3]	11.2	✗	22.84	63.57	14.85	65.88	8.77	40.15	10.08	37.80
DER++ [4]	11.2	✗	27.46	62.55	21.76	59.54	11.16	40.91	11.92	31.96
ERT [5]	11.2	✗	21.61	54.75	12.91	58.49	10.85	39.54	13.51	36.94
RM [2]	11.2	✗	32.23	62.05	22.71	66.28	13.58	41.96	16.76	35.18
LVT† [51]	8.9	✗	39.68	66.92	35.41	72.80	17.34	46.15	19.46	41.78
DyToX† [14]	10.7	✗	36.52	-	25.94	-	13.14	-	24.82	-
A-D† [36]	5.6	✓	15.61	42.82	16.77	55.53	-	-	10.92	39.13
Ours†	12.4	✓	**49.36**	**81.01**	**35.90**	**82.08**	**26.82**	**65.92**	**40.10**	**81.82**
Ours (dis-80)†	10.7	✓	48.85	79.75	35.42	81.97	26.44	65.02	36.22	80.28

Table 1. Comparison on CIFAR-100, Tiny-ImageNet, and ImageNet100. All methods except ours and A-D use a memory buffer of 200 exemplars. The accuracies reported here are the ACC_{TAG} and ACC_{TAW} computed after the training the last task. The methods marked with † are based on ViT. *Ours (dis-80)* indicates using GCAB distillation with 80% capacity usage in the distilled CAB.

ited to 200 images, which is the setting proposed in [51].

From Table 1, we see that our method outperforms all methods in both class-incremental and task-incremental scenarios. Note that our approach outperforms the three other ViT-based methods, Dytox, LVT, and A-D. Our method doubles the class-IL results obtained by A-D, the only other exemplar-free ViT method. For Tiny-ImageNet we observed that our method was able to outperform all the compared methods by a large margin. In particular, we obtain a significant improvement of almost 10% with respect to LVT even though LVT uses exemplars and we do not. Also on the larger ImageNet100 dataset, we outperform all the other competitors by a large margin. Especially notable are the improved results with respect to DyToX which is based on the same architecture as ours but uses exemplars.

To further compare our performance with DyToX, we vary the number of exemplars in DyToX and compare this to our exemplar-free results (see Figure 5(b)). We see that, when few exemplars are considered (less than 400), DyToX achieves lower task-agnostic accuracy on CIFAR-100 in the 10-task scenario. The performance of our method is higher with respect to DyToX using up to 500 exemplars. Increasing the buffer further allows DyToX to obtain significantly higher accuracy at the cost of storing more exemplars.

Larger first task scenarios. In this setting the number of classes involved in the training of the architecture during the first task is 40 for CIFAR-100 and 100 for TinyImagenet. The remaining classes (60 and 100, respectively) are then split into 5 tasks. This scenario is less challenging than the *equal task split scenario* because it allows training of a high-quality backbone on the first task. This setup is often used by exemplar-free methods which during continual learning can then rely on the backbone trained during the first task. We only compare to other exemplar-free methods in this experiment.

We compare our model with several state-of-the-art methods. For this scenario we report the average incremental accuracy defined as: $ACC_{AVG} = \frac{1}{T}\sum_{t=1}^{T} a_t$, where a_t is the task-agnostic accuracy computed over the classes observed up to task t (as used by [45] for this experiment). In Table 2, we observe that our method obtains the best performance on CIFAR-100 and TinyImageNet. The gain on the CIFAR-100 5-task scenario is especially notable, where we outperform the state-of-the-art by over 5%.

Method	CIFAR-100	Tiny-ImageNet
EWC [23]	26.26	14.63
LWF [27]	39.51	40.62
LWM [11]	40.49	28.39
PASS [60]	56.53	47.00
SDC [55]	57.62	47.89
Evanescent [45]	59.37	48.56
Ours	**65.05**	**50.01**

Table 2. Comparison of methods on CIFAR-100 and Tiny-ImageNet on the larger first task scenario for 5 tasks. The accuracies reported here are the average incremental accuracies ACC_{AVG}.

Gated Class Attention	Backbone Regularization	Feature Drift Compensation	$\mathbf{ACC_{TAG}}$
			11.90
✓			31.35
✓	FD		30.50
✓	PFR (2)		7.96
✓	PFR (1)	✓	31.82
✓	PFR (2)	✓	35.90

Table 3. Ablation study on the components of our architecture. Results are on the 10-task scenario on CIFAR-100. We report the average incremental accuracy ACC_{TAG}. In parentheses are the number of layers of the MLP used for PFR.

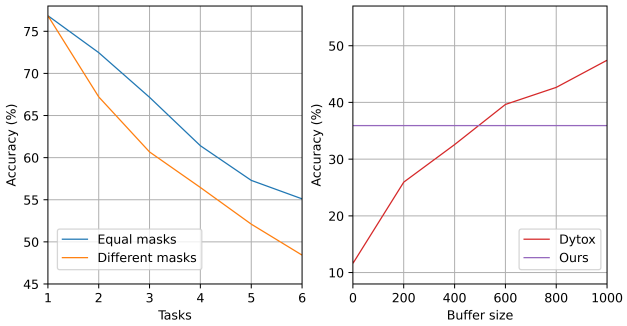

Figure 5. (a) Ablation on CIFAR-100 in the larger first task scenario. The two curves show different choices for the GCAB. (b) Comparison between our method and DyToX. Our **exemplar-free method** is competitive with this state-of-the-art exemplar-based method until the memory buffer reaches around 500.

4.2. Ablation Study

We ablate on the importance of the different components of our approach. In Table 3 we report six possible configurations on the 10-task split of CIFAR-100. First, we consider fine-tuning our architecture without applying any continual learning strategy to the base architecture (11.90%). Then we apply the gated class-attention mechanism to the transformer decoder. This solution increases the average accuracy by 20%, showing the importance of preventing forgetting in the final block. As explained in Section 3.2, we use only two masks for gating the transformer decoder. This does not prevent backbone weight drift when passing from one task to the next. Therefore, we apply a simple backbone regularization based on feature distillation [21] which does not increase overall performance (30.50%). When we replace feature distillation with the projected function regularization in combination with feature drift compensation, we obtain better performance – notably, a more than 5% increase when using a 2-layer MLP. These results confirm the importance of projecting the learned backbone features to the previous features space.

We additionally analyze the design of the masks considered in the Gated Class-Attention Block. We consider the larger first task scenario, 40 classes in the first task, and the remaining are split equally split in 5 tasks. As reported in the left part og Figure 5, we plot the results obtained with two different choices for the masks: 1) For the orange curve $m_{QK} \neq m_V \neq m_1 \neq m_o \neq m_i \neq m_2$. 2) For the blue curve $m_{QK} = m_V = m_1 = m_o = m_i$. From the figure we see that using the two masks for the gated class-attention block provides the best results. In the Supplementary Material we report the percentage of masked capacity used in relation to the increase in the number of classes observed on CIFAR-100 for both 5- and 10-task scenarios.

	Time (ms)	GFLOPs
ViT	~ 21	0.589
GCAB	~ 30	1.648
Distilled GCAB	$\sim$ **21**	**0.606**

Table 4. Runtime and computational cost for a single forward pass with the base ViT, and our method with and without GCAB distillation after the last task on the CIFAR-100 10 task scenario.

4.3. GCAB Distillation

We conduct the experiments with GCAB distillation in four scenarios: CIFAR-100 with 5 and 10 tasks, and Tiny-ImageNet with 5 and 10 tasks. We freeze the transformer encoder, projection network, GCAB, and classifiers after the last task and train the student CAB (as shown in Figure 4) with an Adam optimizer and a learning rate $5e^{-3}$ for 200 epochs with only data from the last task. When performing GCAB distillation we use static binary masks at the same position as the masks in GCAB (see Fig.2 (right)) to control the capacity usage in the student CAB (this potentially allows us to continue training on further tasks).

From the results in Figure 6 we see that GCAB distillation obtains almost the same performance as the model before distillation when the capacity usage is higher than 80%, and only a small performance drop at 60% capacity usage. As shown in Table 4, GCAB distillation can overcome the increased computational cost from multiple forward passes for all tasks during inference.

5. Conclusion

We presented an exemplar-free approach to class-incremental Visual Transformer training. Our method, through the Gated Class-Attention Mechanism, achieves low forgetting by learning and masking important neurons for each task. High plasticity is ensured via backbone regularization and Feature Drift Compensation using a cascade of feature projection networks. Through classifier distillation, we are able to overcome the limitations due to the multiple forward passes and computational overhead required by multiple forward passes through a task-conditioned network. Ours is one of the first effective approaches to exemplar-free, class-incremental training of ViTs.

Acknowledgements

We acknowledge the Spanish Government funded project PID2019-104174GB-I00/AEI/10.13039/501100011033 and the European Commission funded Horizon 2020 project, grant number 951911 (AI4Media).

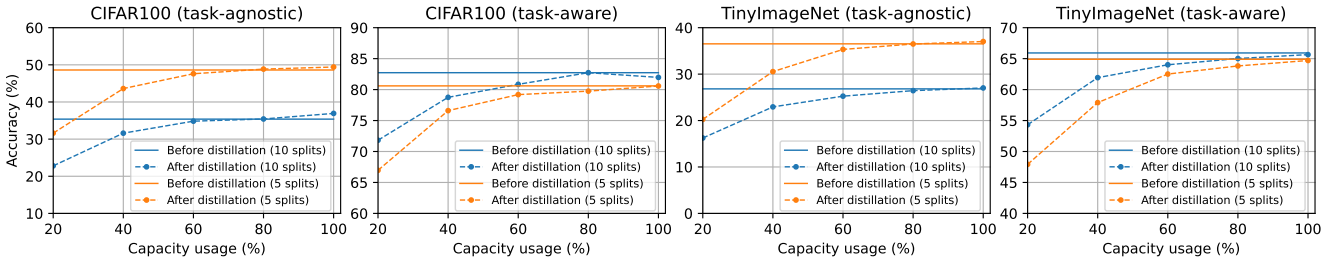

Figure 6. Average accuracy after the last task with and without GCAB distillation.

A. Supplementary Material

A.1. Datasets

The CIFAR-100 dataset is composed of 60,000 images, each 32×32 pixels and divided into 100 classes. Each class has 500 training and 100 test images. The Tiny-ImageNet is a reduced version of the original ImageNet dataset with 200 classes. The classes are split into 500 for training, 50 for validation, and 50 for test (for a total number of 120,000 images). The images are 64×64 pixels. The ImageNet100 dataset is a selection of 100 classes from the larger ImageNet dataset (composed of 1000 classes). The images in this reduced version are 120,000 for training and 5000 for test. For our purposes, we resized the images of this third dataset to 224×224.

A.2. Update Rules for Gated Class-Attention

The backpropagation update rules are the following:

$$
\begin{aligned}
W_Q &= W_Q - \lambda M_q^{<t} \odot \frac{\partial L}{\partial W_q} & W_k &= W_k - \lambda M_k^{<t} \odot \frac{\partial L}{\partial W_k} \\
W_v &= W_v - \lambda M_v^{<t} \odot \frac{\partial L}{\partial W_v} & W_\theta &= W_\theta - \lambda (1 - m_{i,k}^{<t}) \odot \frac{\partial L}{\partial W_\theta} \\
W_o &= W_o - \lambda M_o^{<t} \odot \frac{\partial L}{\partial W_o} & W_1 &= W_1 - \lambda M_1^{<t} \odot \frac{\partial L}{\partial W_1} \\
W_2 &= W_2 - \lambda M_2^{<t} \odot \frac{\partial L}{\partial W_2} & W_{\text{clf}} &= W_{\text{clf}} - \lambda (1 - m_o^{<t}) \odot \frac{\partial \text{clf}}{\partial W_{\text{clf}}}
\end{aligned}
$$

and the weight masks are defined as:

$$
\begin{aligned}
M_{q,kl}^{<t} &= 1 - \min(m_{i,k}^{<t}, m_{QK,l}^{<t}) & M_{k,kl}^{<t} &= 1 - \min(m_{i,k}^{<t}, m_{QK,l}^{<t}) \\
M_{v,kl}^{<t} &= 1 - \min(m_{i,k}^{<t}, m_{V,l}^{<t}) & M_{o,kl}^{<t} &= 1 - \min(m_{V,k}^{<t}, m_{1,l}^{<t}) \\
M_{1,kl}^{<t} &= 1 - \min(m_{1,i}^{<t}, m_{2,l}^{<t}) & M_{2,kl}^{<t} &= 1 - \min(m_{2,i}^{<t}, m_{o,l}^{<t})
\end{aligned}
$$

A.3. Additional results

Accuracy Matrices. In Figures 7 and 8, we show the task aware and task agnostic accuracy matrices for the 5-task scenario on CIFAR-100. We observe that, through our gated class-attention block, we achieve stable accuracy on previous tasks in the task-aware scenario. In Figure 8 the task-agnostic matrix shows a drop in performance when learning new tasks. However, this behaviour is expected since the problem becomes increasingly complex when adding new tasks.

Cumulative Metrics. To verify the accuracy and forgetting of our method, we compute the Cumulative Accuracy and Cumulative Forgetting as defined by [43]. We see from Figure 9 that the cumulative accuracy of our model in the 10-task scenario on CIFAR-100 is stable without any drop when increasing the number of observed tasks. We report in Figure 10 the cumulative forgetting for each task. Even if there is a slight increase for early tasks, the cumulative forgetting does not increase above 4%.

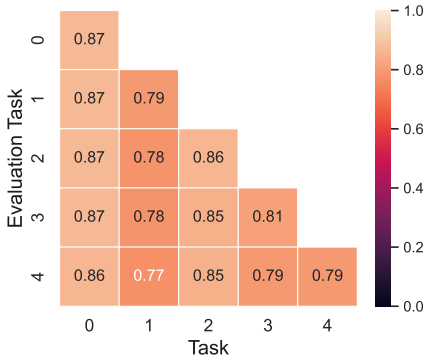

Figure 7. Task-Aware on CIFAR-100, 5 tasks.

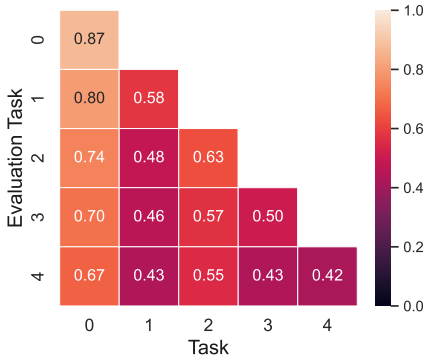

Figure 8. Task-Agnostic on CIFAR-100, 5 tasks.

Gating Capacity. In Figure 11, we show the percentage of used masked capacity as tasks are added. The experiments were performed on CIFAR-100 dataset for 5- and 10-task scenarios. We observe that most most of the available capacity is used during the firsts tasks. For subsequent ones the percentage of occupied capacity is significantly lower compared to the first ones. From the 5-task scenario

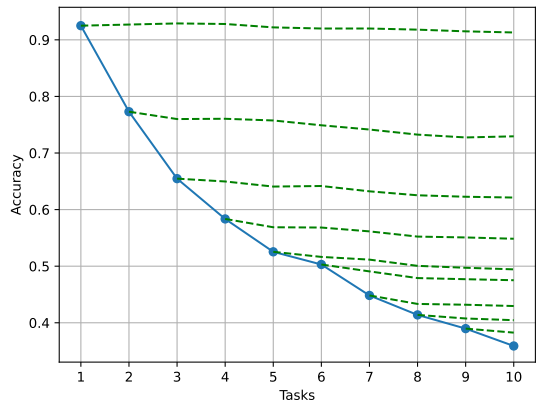

Figure 9. Cumulative Accuracy on the 10-task scenario of CIFAR-100.

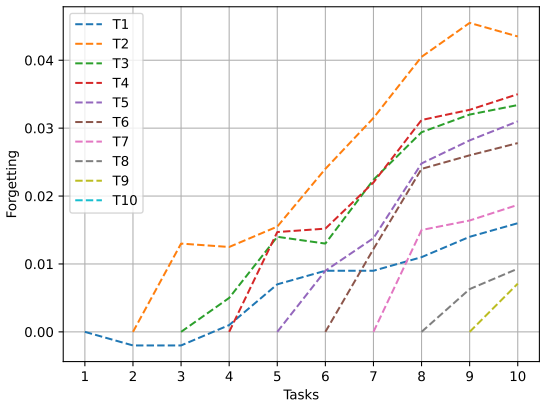

Figure 10. Cumulative Forgetting on the 10-task scenario of CIFAR-100.

curve we that there is almost 10% of the capacity available for possible new incoming tasks. For the 10-task scenario, however, after completing 60% of the tasks the capacity is almost full and the last tasks are using less than 2% of capacity each.

Equal task split scenarios: Exemplar-free comparison. In Table 5 we compare our method with other exemplar-free approaches from the state-of-the-art on the equal task split scenario. We observe that our method outperforms PASS [60] (that uses self-supervision for guiding incremental learning) and SDC [55]. The results show a significant drop SDC performance compared to the results in Table 2. This is because SDC is very sensitive to the size of the initial task since it requires a strong backbone. The original paper only includes results on larger first tasks. The good results of PASS show the strength of self-supervised representations which generalize well to new tasks. We think

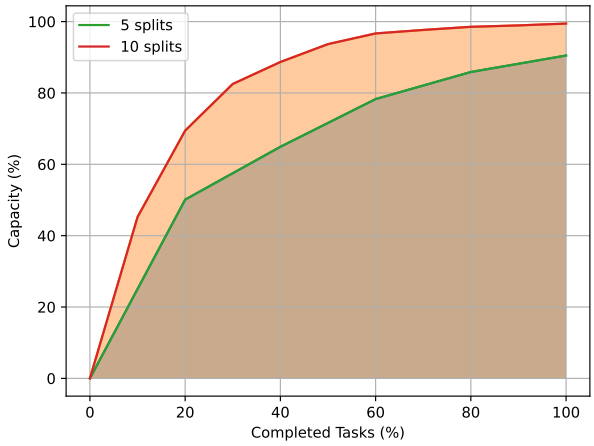

Figure 11. Gated Class-Attention Block capacity on CIFAR-100 in the 5- and 10-task scenarios.

	CIFAR-100		Tiny-ImageNet	ImageNet100
	5 Splits	10 Splits	10 Splits	10 Splits
Ours	49.36	35.90	26.82	40.10
PASS [60]	48.28	33.76	24.23	25.22
SDC [55]	6.65	7.41	3.94	11.52

Table 5. Exemplar-free methods comparison on Equal task split scenarios.

that this observation is complementary to our method and that they could also be combined in future work.

A.4. Hyperparameter Analysis

We analyze the importance and the robustness of our method with respect to the hyperparameters described in Section 3. In Figure 13 we show the behavior of our method in terms of ACC_{TAG} on the CIFAR-100 5-task scenario under changing hyperparameters. In particular, in the upper part of the figure the accuracy as a function of λ_{GCAB} is shown. In this plot the other hyperparameter λ_{pfr} is kept fixed at 0.001 (as described in Section 4 of the main text). The accuracy is stable, confirming the robustness of our method over a wide range of values of λ_{GCAB}. Only for extremely large values of λ_{GCAB} does the accuracy significantly drop.

In the lower part of Figure 13 we show the variation of the ACC_{TAG} as a function of λ_{pfr} when $\lambda_{GCAB} = 0.05$. For higher values of λ_{pfr}, the value of $\mathcal{L}_{\text{pfr}}$ strongly overpowers $\mathcal{L}_{\text{BCE}}$ and $\mathcal{L}_{\text{GCAB}}$, pushing the model to not correctly learn useful features for the classification and the mask. For smaller values of λ_{pfr} the magnitude of this term of the loss function is comparable with the others, which emphasizes the importance of correctly learning a projector net-

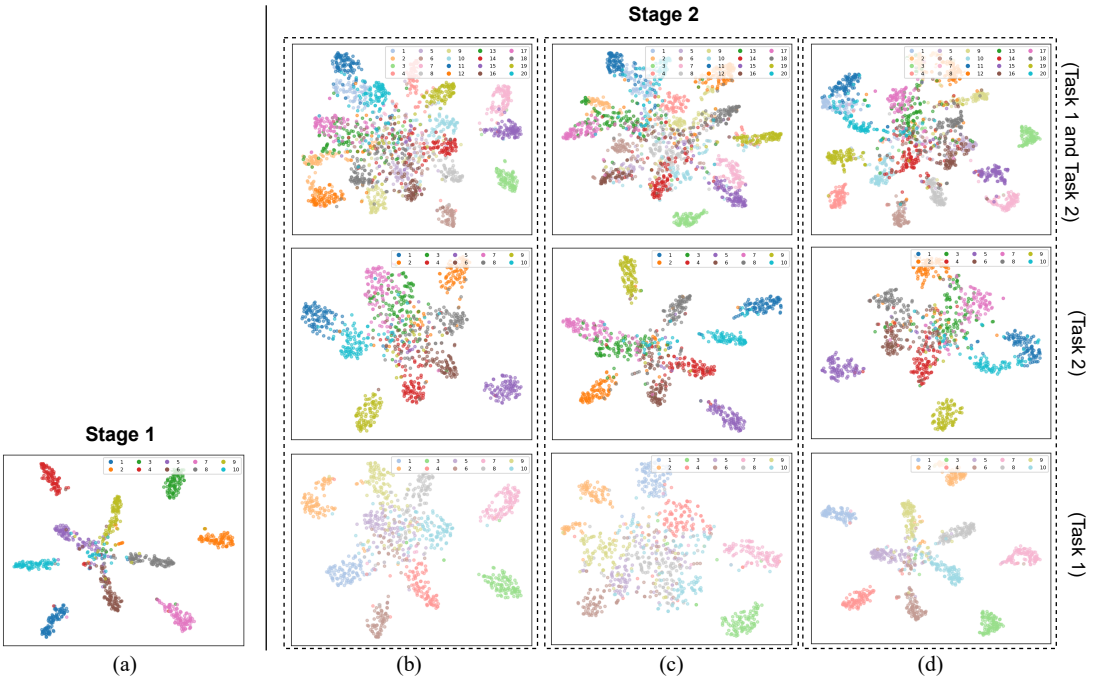

Figure 12. t-SNE visualization of embedding space (output of $f(\cdot)$) at the first two tasks (task-agnostic). (a) Results after Stage 1. Results after Stage 2 (b) without PFR during training, (c) with PFR during training but without feature drift compensation, and (d) with PFR during training and with feature drift compensation

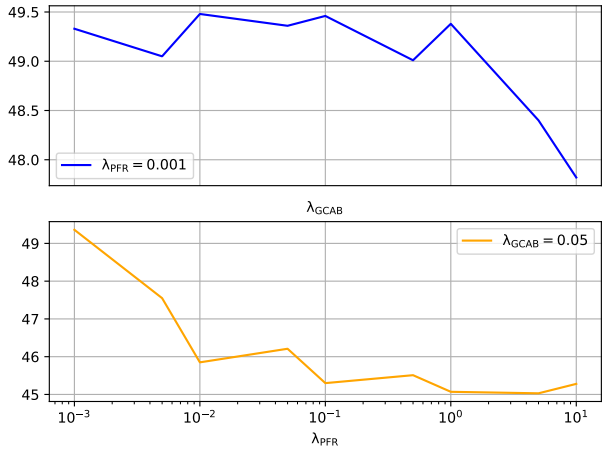

Figure 13. $ACC - TAG$ on CIFAR-100 for the 5-task scenario under varying hyperparameters. **Top:** for fixed λ_{pfr}, the ACC_{TAG} as a function of λ_{GCAB}. **Bottom:** for fixed λ_{GCAB}, the ACC_{TAG} as a function of λ_{pfr}.

work that will be used during inference.

A.5. Additional Embedding Visualizations

We report here the complete visualization of the embeddings produced with and without the projection network cascade. In column (a) of the Figure 12 we show the embedding obtained after the training of the first task. In column (b) we report the result of the embedding without PFR during training. In column (c) and (d) we show the embeddings obtained using the PFR during training. The difference between (c) and (d) is the use of the Feature Drift Compensation mechanism during inference. The different rows of the figure show the embeddings produced by the network for the different tasks.

The projection network learned during the training of the second task is used to align the embeddings produced by the current backbone to the ones produced by the previous one. We see that during inference on first task data after the training of the second task (i.e. Stage 2 in Figure 12) the Feature Drift Compensation better preserves first-task clusters in embedding space: using the projection network, the embeddings of first task classes are clearly clustered. We also analyze the test set embedding features of the second task (second row), after the training of the second task. In Figure 14 the task-aware feature embeddings with and without the use of the Feature Drift Compensation during the inference are reported.

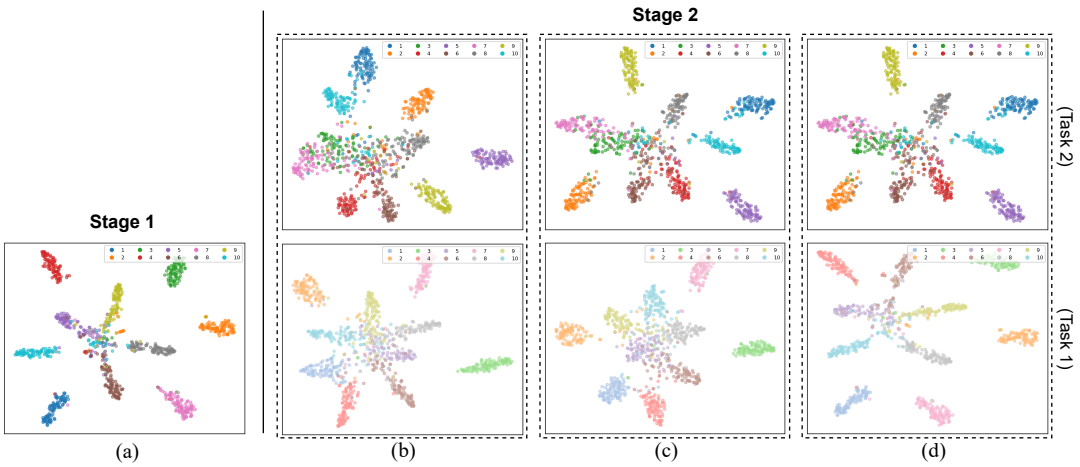

Figure 14. t-SNE visualization of embedding space (output of $f(\cdot)$) at the first two tasks (task-aware). (a) Results after Stage 1. Results after Stage 2 (b) without PFR during training, (c) with PFR during training but without feature drift compensation, and (d) with PFR during training and with feature drift compensation

A.6. Comparison of Our Contribution with Other Methods

To clarify our contribution in this paper, we summarize the difference compared with recent continual learning methods in Table 6 and Table 7. As shown in Table 6, our method is the first exemplar-free class incremental learning method based on the transformer. [36] also propose a transformer-based CL method with attention distillation, but for task-incremental learning. The method of [52] requires a pre-trained backbone and dynamically prompts it to a sequence of tasks. [14] propose a transformer-based method with dynamic token expansion for continual learning. This method achieves very good performances on CIL, but requires exemplars for mitigating the forgetting. Finally [51] propose a lifelong vision transformer without any pretraining but requiring exemplars.

Method	Architecture	Exemplars-Free	Non-Pretrained
Ours	ViT	✓	✓
Dytox [14]	ViT	✗	✓
LVT [51]	ViT	✗	✓
A-D [36]	ViT	✓	✓
Prompt [52]	ViT	✓	✗

Table 6. Comparison with published state-of-the-art approaches.

In Table 7 we report recent parameter-isolation continual learning methods. To the best of our knowledge, ours is the first exemplar-free class-incremental learning method trained completely from scratch. Piggyback [31] uses task-specific binary masking for adapting the backbone network to new tasks. Although it does not use exemplar rehearsal, it is based on a pretrained backbone. Similarly, [32] describe PackNet which is based on task-aware iterative pruning and re-training to mitigate forgetting and increase plasticity. These methods apply the learned masks directly to the weights of the network. Another category of parameter-isolation methods instead apply the masks to the layer activations. [42] describe a task-aware method for preventing updates to weights that are most useful for previous tasks. Similarly, [34] propose a ternary-mask method without the need for pretraining or exemplars. Despite their effectiveness, these activation masking methods are task-aware. Our instead is the first parameter isolation, task-agnostic and exemplar-free approach trained entirely from scratch.

Method	Weights/Activations	Class-incremental	Scratch	Exemplar-free
Piggyback [31]	Weights	✗	✗	✓
PackNet [32]	Weights	✗	✓	✓
HAT [42]	Activations	✗	✓	✓
Ternary [34]	Activations	✗	✓	✓
Ours	Activations	✓	✓	✓

Table 7. Comparison with different parameter-isolation approaches from the state-of-the-art.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018. 2

[2] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8218–8227, 2021. 1, 6, 7

[3] Ari S Benjamin, David Rolnick, and Konrad Kording. Measuring and regularizing networks in function space. *arXiv preprint arXiv:1805.08289*, 2018. 6, 7

[4] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020. 1, 6, 7

[5] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and Simone Calderara. Rethinking experience replay: a bag of tricks for continual learning. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2180–2187. IEEE, 2021. 1, 6, 7

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021. 1

[7] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018. 2

[8] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018. 2

[9] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*, 2018. 1, 6, 7

[10] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 3

[11] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5138–5146, 2019. 2, 7

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 3

[13] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 86–102. Springer, 2020. 5

[14] 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, 2, 6, 7, 12

[15] Enrico Fini, Victor G Turrisi da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2022. 5

[16] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International conference on machine learning*, pages 1243–1252. PMLR, 2017. 3

[17] Alex Gomez-Villa, Bartlomiej Twardowski, Lu Yu, Andrew D Bagdanov, and Joost van de Weijer. Continually learning self-supervised representations with projected functional regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3867–3877, 2022. 5

[18] Ian J. Goodfellow, Mehdi Mirza, Xia Da, Aaron C. Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgeting in gradient-based neural networks. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. 1

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[20] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 6

[21] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839, 2019. 2, 5, 8

[22] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016. 2

[23] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 1, 2, 7

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[25] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 6

[26] Janghyeon Lee, Hyeong Gwon Hong, Donggyu Joo, and Junmo Kim. Continual learning with extended kronecker-factored approximate curvature. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9001–9010, 2020. 2

[27] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1, 2, 7

[28] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your

networks: Better weight consolidation and less catastrophic forgetting. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 2262–2268. IEEE, 2018. 2

[29] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 1

[30] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017. 1

[31] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82, 2018. 2, 12

[32] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018. 2, 3, 12

[33] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. *arXiv preprint arXiv:2010.15277*, 2020. 2, 3

[34] Marc Masana, Tinne Tuytelaars, and Joost Van de Weijer. Ternary feature masks: zero-forgetting for task-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3570–3579, 2021. 2, 12

[35] Martial Mermillod, Aurélia Bugaiska, and Patrick Bonin. The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects, 2013. 1

[36] Francesco Pelosin, Saurav Jha, Andrea Torsello, Bogdan Raducanu, and Joost van de Weijer. Towards exemplar-free continual learning in vision transformers: an account of attention, functional and weight regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3820–3829, 2022. 3, 6, 7, 12

[37] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[38] 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*, pages 2001–2010, 2017. 1, 2, 6, 7

[39] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018. 1, 6, 7

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 6

[41] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv*, 2016. 3

[42] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018. 1, 2, 3, 4, 12

[43] Albin Soutif-Cormerais, Marc Masana, Joost Van de Weijer, and Bartłomiej Twardowski. On the importance of cross-task features for class-incremental learning. *CoRR*, 2021. 9

[44] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7262–7272, 2021. 1

[45] Marco Toldo and Mete Ozay. Bring evanescent representations to life in lifelong class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16732–16741, 2022. 3, 7

[46] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 32–42, 2021. 3, 4

[47] Gido M van de Ven and Andreas S Tolias. Three scenarios for continual learning. In *NeurIPS Continual Learning Workshop*, 2018. 3

[48] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 6

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1, 3

[50] Liyuan Wang, Kuo Yang, Chongxuan Li, Lanqing Hong, Zhenguo Li, and Jun Zhu. Ordisco: Effective and efficient usage of incremental unlabeled data for semi-supervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5383–5392, 2021. 2

[51] Zhen Wang, Liu Liu, Yiqun Duan, Yajing Kong, and Dacheng Tao. Continual learning with lifelong vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 171–181, 2022. 1, 2, 6, 7, 12

[52] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149, 2022. 2, 12

[53] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019. 2

[54] 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, 2

[55] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6982–6991, 2020. 1, 3, 7, 10

[56] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017. 2

[57] Mengyao Zhai, Lei Chen, and Greg Mori. Hyper-lifelonggan: scalable lifelong learning for image conditioned generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2246–2255, 2021. 2

[58] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020. 2

[59] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021. 2

[60] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5871–5880, 2021. 7, 10