

Read Between the Layers: Leveraging Intra-Layer Representations for Rehearsal-Free Continual Learning with Pre-Trained Models

Kyra Ahrens* Hans Hergen Lehmann* Jae Hee Lee Stefan Wermter
University of Hamburg

{kyra.ahrens, jae.hee.lee, stefan.wermter}@uni-hamburg.de
hergen.lehmann@studium.uni-hamburg.de

Abstract

We address the Continual Learning (CL) problem, where a model has to learn a sequence of tasks from non-stationary distributions while preserving prior knowledge as it encounters new experiences. With the advancement of foundation models, CL research has shifted focus from the initial learning-from-scratch paradigm to the use of generic features from large-scale pre-training. However, existing approaches to CL with pre-trained models only focus on separating the class-specific features from the final representation layer and neglect the power of intermediate representations that capture low- and mid-level features naturally more invariant to domain shifts. In this work, we propose LayUP, a new class-prototype-based approach to continual learning that leverages second-order feature statistics from multiple intermediate layers of a pre-trained network. Our method is conceptually simple, does not require any replay buffer, and works out of the box with any foundation model. LayUP improves over the state-of-the-art on four of the seven class-incremental learning settings at a considerably reduced memory and computational footprint compared with the next best baseline. Our results demonstrate that fully exhausting the representational capacities of pre-trained models in CL goes far beyond their final embeddings. The code will be made publicly available upon acceptance.

1. Introduction

Continual Learning (CL) is a subfield of machine learning that focuses on enabling models to learn from a stream of data and adapt to novel concepts without forgetting previously acquired knowledge [6, 45]. While traditional works on CL primarily focus on the learning-from-scratch paradigm, the introduction of large foundation models has initiated a growing interest in developing CL methods upon the powerful representations resulting from large-scale pre-

*Equal contribution.

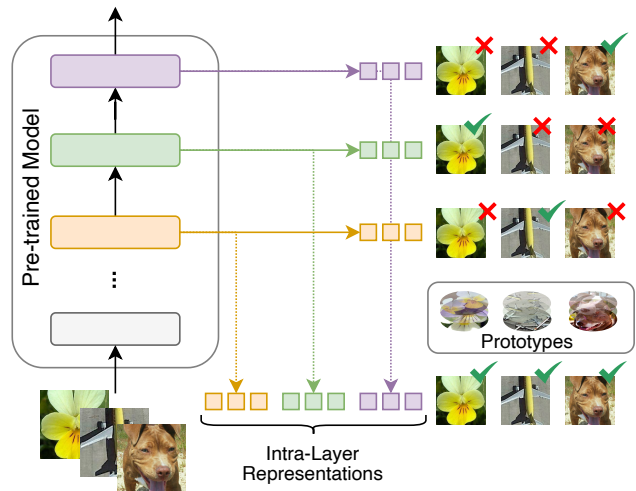


Figure 1. Overview of LayUP: Enriching last layer representations of pre-trained models with features from intermediate layers leads to better calibrated similarity measures for prototype-based classification and more robustness to domain gaps in the continual learning setting.

training. Continual learning with pre-trained models builds on the assumption that a large all-purpose feature extractor provides strong knowledge transfer capabilities and great robustness to catastrophic forgetting [39] during incremental adaptation to downstream tasks.

Such methods for downstream adaptation can be categorized into two general strategies [40, 61]: (i) Carefully fine-tuning the pre-trained representations (*full body adaptation*), or (ii) keeping the pre-trained backbone fixed while learning additional parameters (such as prompts [22, 31, 35], adapters [5, 7, 19, 20, 50], or other methods [25, 34, 38]). Both strategies have in common that they do not rely on rehearsal from past data to perform well, unlike many of the top-performing CL strategies used for models trained from scratch. Despite their merits, most approaches to CL with pre-trained

models that fall under one of those strategies update a fully connected classification head (*linear probe*) via iterative gradient methods during training and inference, which is subject to task-recency bias [37, 54, 62] and catastrophic forgetting [48, 69]. Class-prototype methods [21, 40, 72] have been established as a viable alternative, as they average extracted features directly, which is not only computationally efficient but also superior to CL methods applied to the trainable linear probe.

Despite their effectiveness, contemporary class-prototype methods for CL use only the features obtained by the last layer of the backbone for class-prototype construction. However, as the discrepancy between the pre-training and fine-tuning domain increases, the high-level last-layer features might not generalize well enough to provide a good class separability in the target domain anymore. The challenge is therefore to find a prototype-based classifier that leverages the representations of a pre-trained feature extractor holistically, ideally at comparably low memory and computational cost.

In this work, we hypothesize that representations are most effectively leveraged from multiple layers to construct a classifier based on first-order (class-prototypes) and second-order (Gram matrix) feature statistics. Such a strategy is inspired by works on Neural Style Transfer [13, 23], which involve applying the artistic style of one image to the content of another image, a problem that can be viewed through the lens of domain adaptation [32]. To achieve neural style transfer, a model has to disentangle image information into content and style (e.g., textures, patterns, colors). While content information is expressed in terms of activations of features at different layers (which translates to calculating intra-layer class-prototypes in our case), style information is expressed as correlations between features of different layers (which corresponds to us calculating intra-layer Gram matrices).

We thus propose **Intra-Layer Universal Prototypes (LayUP)**, a class-prototype method for CL that is based on the aforementioned strategy to disentangle style and content information of images at multiple layers of a pre-trained network. An overview of our method is given in Fig. 1. LayUP obtains rich and fine-grained information about images to perform a more informed classification that is less sensitive to domain shifts. We additionally experiment with different parameter-efficient fine-tuning strategies to further refine the intra-layer representations obtained by LayUP. Across several CL benchmarks and learning settings, our method provides improvements and reduces the gap to upper bound performance by up to 67% (Stanford Cars-196 in the CIL setting) compared with the next best baseline while reducing its memory and compute requirements by 81% and up to 90%, respectively.

Our contributions are threefold: (1) We present and ex-

amine in detail the CL strategy with pre-trained models and show why it benefits from using intra-layer representations for classification. We further demonstrate the advantages of leveraging cross-correlations between intra-layer and inter-layer features to decorrelate class-prototypes. (2) Building on our insights, we propose LayUP, a novel class-prototype approach to CL that leverages second-order statistics of features from multiple layers of a pre-trained model. Inspired by prior works [40, 71], we experiment with different methods for parameter-efficient model tuning and extend our method towards first session adaptation. Our final approach is conceptually simple, light in memory and compute, and works out-of-the-box with any transformer model. (3) We report performance improvements with a pre-trained ViT-B/16 [12] backbone on several benchmarks both in the Class-Incremental Learning (CIL) and in the more challenging Online Continual Learning (OCL) setting and show that LayUP is especially effective under large distributional shifts and in the low-data regime. Our results highlight the importance of taking a deeper look at the intermediate layers of pre-trained models to better leverage their powerful representations, thus identifying promising directions for CL in the era of large foundation models.

2. Related Work

2.1. Continual learning

Traditional approaches to CL consider training a model from scratch on a sequence of tasks, while striking a balance between adapting to the currently seen task and maintaining high performance on previous tasks. They can be broadly categorized into *regularization* [1, 11, 27, 33, 67], which selectively restrict parameter updates, *(pseudo-)rehearsal* [2, 4, 47, 51], which recover data either from a memory buffer or from a generative model, and *dynamic architectures* [53, 55, 66], which allocate disjoint parameter spaces for each task.

2.2. Continual learning with pre-trained models

Leveraging the powerful representations from large foundation models for downstream continual learning has shown to not only facilitate knowledge transfer, but also to increase robustness against forgetting [43, 49]. L2P [65] and Dual-Prompt [64] outperform traditional CL baselines by training a pool of learnable parameters (i.e., prompts) that are used as queries to guide a Vision Transformer (ViT) [12] towards adapting to downstream tasks. They serve as the basis for several more recent prompt learning methods such as S-Prompt [63], DAP [24], and CODA-Prompt [56], which further increase performance.

Contrary to prompt learning methods that keep the backbone parameters fixed while updating a small set of additional parameters, some parallel works explore methods for

making careful adjustments to the backbone parameters. SLCA [69] uses a small learning rate for updates to the ViT backbone, while training a linear probing layer with a larger learning rate. L2 [57] applies regularization to the self-attention parameters of a ViT during continual fine-tuning. First Session Adaptation (FSA) [44] makes adjustments to the parameters of the backbone only during first task training.

As an alternative to the methods above that adjust the feature space via cross-entropy loss and backpropagation, constructing prototypes directly from the pre-trained features can avoid catastrophic forgetting during CL without storing data. Prior works [21, 72] find that a conceptually simple Nearest Mean Classifier (NMC) outperforms various prompt learning methods. ADAM [72] combines NMC with FSA and concatenates the features from the initial pre-trained ViT and the adapted parameters. Other methods leverage second-order feature statistics, i.e., covariance information, for class-prototype accumulation: FSA+LDA [44] applies an incremental version of linear discriminant analysis to a pre-trained ResNet encoder [14]. RanPAC [40] uses high-dimensional random feature projections to decorrelate class-prototypes of a pre-trained ViT backbone.

All the aforementioned approaches have in common that they consider only the last layer representations (or, features) for classification. We provide a different perspective to the prototype-based approach to CL and show that intra-layer representations can add valuable information to the linear transformation of class prototypes (cf. Eq. (5)), which leads to better class separability and increased robustness to large distributional shifts.

3. Preliminaries

3.1. Continual learning problem setting

We consider a feature extractor $\phi(\cdot)$ composed of L consecutive layers and a sequence of training datasets $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_T\}$, where the t^{th} task $\mathcal{D}_t = \{(\mathbf{x}_{t,n}, y_{t,n})\}_{n=1}^{N_t}$ contains pairs of input samples $\mathbf{x}_{t,n} \in \mathcal{X}_t$ and their ground-truth labels $y_{t,n} \in \mathcal{Y}_t$. Let \mathcal{Y}_t denote the label space of the t^{th} task and $\mathcal{Y} = \bigcup_{t=1}^T \mathcal{Y}_t$. The total number of classes seen in \mathcal{D} (i.e., the number of unique elements in \mathcal{Y}) is denoted with C . We represent the d_l -dimensional encoded features from the l^{th} layer of model ϕ from some test image $\mathbf{x} \in \mathcal{D}_{\text{test}}$ with unknown label during inference as $\phi_l(\mathbf{x}) \in \mathbb{R}^{d_l}$. We consider rehearsal-free CL, where historical data can not be fetched for replay. Experiments in Sec. 5 are conducted under *Class-Incremental Learning* (CIL) and *Online Continual Learning* (OCL) settings which prohibit the access of the model to task identifiers during testing (notably, our method does not need any task-specific information during training, either). OCL is a more tightened definition of CIL towards each datum in the stream of tasks being observed only once.

3.2. Class-prototype methods for CL with pre-trained models

To leverage the powerful representations from large-scale pre-training while overcoming the limitations of full body adaptation and linear probing, *class-prototype* methods construct representatives for each class from the last layer features of a pre-trained model. The most straightforward class-prototype method is the Nearest Mean Classifier (NMC), which aggregates per-class training sample features via averaging (denoted as $\bar{\mathbf{c}}_y$ for each class y) and, during inference, selects for each test sample’s feature vector the class with the smallest Euclidean distance [21] or highest cosine similarity [72] with its class-prototype. Considering the cosine similarity measure and some $\mathbf{x} \in \mathcal{D}_{\text{test}}$, the predicted class label is obtained by:

$$\hat{y} = \arg \max_{y \in \{1, \dots, C\}} s_y, \quad s_y := \frac{\phi_L(\mathbf{x})^T \bar{\mathbf{c}}_y}{\|\phi_L(\mathbf{x})\| \cdot \|\bar{\mathbf{c}}_y\|}, \quad (1)$$

where $\|\cdot\|$ denotes the L^2 norm. However, it was found that the assumption of an isotropic covariance of features (i.e., features are mutually uncorrelated) that is made under Eq. (1) does not hold for pre-trained models. To account for correlations between features as a means to better “calibrate” similarity measures, class-prototype methods that leverage second-order statistics were proposed [40, 44]. One such method is based on the closed-form ordinary least square solution [41] to ridge regression [18], which is very effective in decorrelating class prototypes in CL [40]. It takes Gram matrix \mathbf{G} and class-prototypes (or, regressands) \mathbf{c}_y that are aggregated via summation (instead of averaging to obtain $\bar{\mathbf{c}}_y$) to yield:

$$\hat{y} = \arg \max_{y \in \{1, \dots, C\}} s_y, \quad s_y := \phi_L(\mathbf{x})^T (\mathbf{G} + \lambda \mathbf{I})^{-1} \mathbf{c}_y \quad (2)$$

for a regression parameter $\lambda \geq 0$, d_L -dimensional identity matrix \mathbf{I} , and \mathbf{G} expressed as summation over outer products as

$$\mathbf{G} = \sum_{t=1}^T \sum_{n=1}^{N_t} \phi_L(\mathbf{x}_{t,n}) \otimes \phi_L(\mathbf{x}_{t,n}) \quad (3)$$

Although Eq. (1) and Eq. (2) are defined with respect to the maximum number of classes C after observing all T tasks, they can be applied after seeing any task $t \leq T$ or any sample $n \leq N_t$. When denoting the extracted features of some input datum $\mathbf{x}_{t,n} \in \mathcal{D}_t$ as $\mathbf{f}_{t,n} = \phi_L(\mathbf{x}_{t,n})$ and considering $\mathbf{F} \in \mathbb{R}^{N \times d_L}$ as concatenated row-vector features $\mathbf{f}_{t,n}$ of all N training samples, Eq. (3) will be reduced to $\mathbf{G} = \mathbf{F}^T \mathbf{F}$, which corresponds to the original definition of a Gram matrix as used in the closed-form ridge estimator [18].

4. LayUP: Intra-Layer Universal Prototypes

4.1. Motivation

We argue that a combination of (i) enriching the last layer representations with hierarchical intra-layer features and (ii) decorrelating intra-layer features, which represent image properties such as content and style, via Gram matrix transformation increases robustness to domain shifts and thus improves generalizability to downstream continual tasks. To this end, we propose to integrate embeddings from multiple layers via concatenation into a ridge regression estimator as defined in Eq. (2). Let

$$\Phi_{-k}(\mathbf{x}) = (\phi_{L-k+1}(\mathbf{x}), \dots, \phi_{L-1}(\mathbf{x}), \phi_L(\mathbf{x})) \quad (4)$$

denote the concatenated input features of some input sample $\mathbf{x} \in \mathcal{D}_{\text{test}}$, extracted from the last k layers of the pre-trained model $\phi(\cdot)$. Such features can be, e.g., class embeddings ([CLS]) of a transformer [59] encoder or flattened feature maps of a ResNet [14] encoder. This yields a modified estimator

$$s_y := \Phi_{-k}(\mathbf{x})^T (\mathbf{G} + \lambda \mathbf{I})^{-1} \mathbf{c}_y \quad (5)$$

with $(d_{L-k+1} + \dots + d_{L-1} + d_L)$ -dimensional identity matrix \mathbf{I} and

$$\mathbf{G} = \sum_{t=1}^T \sum_{n=1}^{N_t} \Phi_{-k}(\mathbf{x}_{t,n}) \otimes \Phi_{-k}(\mathbf{x}_{t,n}) \quad (6)$$

We first motivate our approach by showing that intra-layer representations capture expressive class statistics for prototype-based classification. For the split CIFAR-100 ($C = 100$) and ImageNet-R ($C = 200$) datasets, we construct one prototype-based classifier using Eq. (2) per layer $l \in \{1, \dots, L\}$ and count the total number of classes that layer l predicts better than any other layer $l' \neq l$. Results are provided in Fig. 2a. We observe that especially the last four intermediate layers (8–11) account for a top performance of between 25% and 54% of all classes, indicating that the representations obtained by intermediate layers can generally add meaningful knowledge to better separate classes at different hierarchical levels.

There are two intuitive ways to integrate intra-layer representations into the class-prototype classifier: Averaging over k separate classifiers per Eq. (2) or concatenating representations from the last k layers to obtain shared class prototypes per Eq. (5). Fig. 2b shows average accuracies for two split datasets and different values of k . For each k , using shared class-specific information as per Eq. (5) is superior to averaging over separate classifiers, showing that the correlations across layers, which are captured in the shared Gram matrix (cf. Eq. (6)), add meaningful information to enhance class separability and thus improve performance. Although this

leads us to choose higher memory demands compared with a last layer classifier and averaging over layer-wise classifiers, we will show in Sec. 4.3 that our method is still considerably lighter in memory and computation than other competitive class-prototype methods for CL.

Interestingly, the performance of the last layer classifier is roughly the same compared with averaging over separate classifiers for all choices of k . This observation refutes the argument that class embeddings from multiple consecutive layers that are individually strong at predicting specific classes might, when combined, introduce noise that impairs performance, which is clearly not the case in Fig. 2b.

4.2. Combination with parameter-efficient model adaptation

A benefit of class-prototype methods for CL is that they can be orthogonally combined with adaptation techniques to refine the pre-trained representations. As accumulated class prototypes are subject to discrepancy upon distributional shifts during supervised fine-tuning, any adaptation strategies to the latent representations should be carefully tailored to the class-prototype method used. Previous works [44, 72] have found adaptation to downstream continual tasks during First Session Adaptation (FSA) on \mathcal{D}_1 as sufficient to bridge the domain gap while maintaining full compatibility with CL.

To maintain the powerful generic features of the pre-trained backbone, we choose to apply FSA to additional parameter-efficient transfer learning (PETL) parameters while keeping the backbone frozen throughout. To update PETL parameters during FSA, we train a linear classification head with N_1 outputs via Adam optimizer and cross-entropy loss and discard the classification head afterward. During the CL phase, we keep all model parameters frozen and update \mathbf{G} and \mathbf{c}_y only.

We follow prior works [40, 72] and experiment with Visual Prompt Tuning (VPT) [22], Scaling and Shifting of Features (SSF) [34], and AdaptFormer [5] as PETL methods and refer to the cited papers for details. The pseudocode of the LayUP algorithm for the CIL setting is depicted in Alg. 1. For the OCL setting, the FSA stage is omitted and a fixed λ is used, as a dynamic λ search requires multiple passes over the input data.

4.3. Memory and runtime complexity comparisons

We compare the memory and runtime requirements of LayUP with the three competitive CL methods ADAM [72], SLCA [69], and RanPAC [40] for a typical class count of $C = 200$. We neglect the memory cost of the ViT backbone, the fully connected classification layer, and the class prototypes, as they are similar to all baselines. LayUP stores $\sim 1\text{M}$ PETL parameters (AdaptFormer [5]) and an additional intra-layer Gram matrix in memory. The size of this layer depends

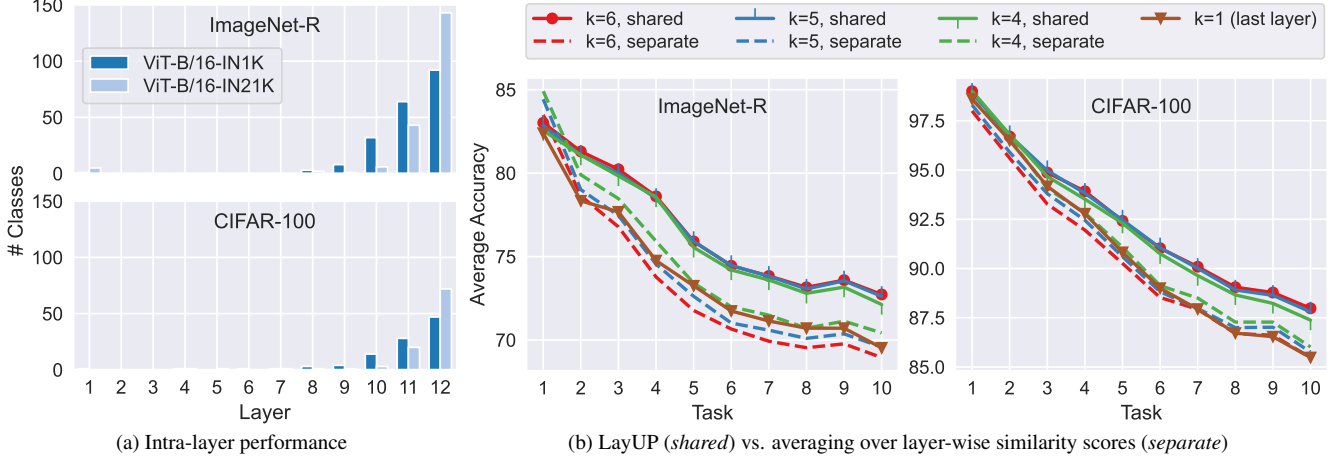


Figure 2. (a) Comparison of classification performance of layers $(1, \dots, L)$ for two different pre-training paradigms of a ViT-B/16 [12] backbone and split ImageNet-R and CIFAR-100 datasets. Each bar indicates the absolute count of classes for which a classifier constructed using Eq. (2) for the respective layer yields highest classification accuracy compared to all other layers.

(b) Average accuracy (%) over CL training on split ImageNet-R and CIFAR-100 datasets following phase B of Alg. 1: Comparison of LayUP implementations for different values of k using Eq. (5) versus averaging over separate similarity scores for each layer using Eq. (2).

Algorithm 1 LayUP Training

Require: pre-trained network $\phi(\cdot)$

Require: PETL parameters

Require: data $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_T\}$

Require: k

Phase A: First Session Adaptation with PETL

for every sample $(\mathbf{x}, y) \in \mathcal{D}_1$ **do**

 Collect $\phi_L(\mathbf{x})$

 Update PETL parameters via Adam [26]

Phase B: Continual Learning with LayUP

for task $t = 1, \dots, T$ **do**

for every sample $(\mathbf{x}_t, y_t) \in \mathcal{D}_t$ **do**

 Collect $\Phi_{-k}(\mathbf{x}_t)$ (Eq. (4))

$\mathbf{G} \leftarrow \mathbf{G} + \Phi_{-k}(\mathbf{x}_t) \otimes \Phi_{-k}(\mathbf{x}_t)$

$\mathbf{c}_{y_t} \leftarrow \mathbf{c}_{y_t} + \Phi_{-k}(\mathbf{x}_t)$

Cf. Appendix A.1

 Optimize λ to compute $(\mathbf{G} + \lambda \mathbf{I})^{-1}$

on the choice of k and has $(d_{L-k+1} + \dots + d_{L-1} + d_L)^2$ entries (note that for ViT-B/16 [12] model, every layer outputs tokens of the same dimensionality, such that $d_l = 768$ for $l \in \{1, \dots, L\}$). For $k = 6$, which we use in our final experiments, the matrix stores $\sim 21\text{M}$ values. RanPAC updates a Gram matrix of size 100M and requires an additional $\sim 11\text{M}$ parameters for the random projection layer and PETL parameters. SLCA stores C additional covariance matrices of size d_L^2 with a total of $\sim 118\text{M}$ entries. Finally, ADAM stores a second copy of the feature extractor, necessitating 84M additional parameters. Assuming a comparable memory cost for model parameters and matrix entries, our method reduces

additional memory requirements by 81%, 82%, and 75% compared with RanPAC, SLCA, and ADAM, respectively.

Considering runtime complexity during training, ADAM performs fine-tuning of a full ViT-B/16 backbone during first-task adaptation. SLCA requires updates to a full ViT-B/16 backbone during slow learning and additionally performs Cholesky decomposition [9] on the class-wise covariance matrices for pseudo-rehearsal, which requires $C \cdot d_L^3 \approx 10^{11}$ operations. RanPAC and LayUP only make updates to PETL parameters during training and perform matrix inversion during inference, albeit with differently constructed and sized Gram matrices. For $k = 6$, the matrix inversion needs $(d_{L-k+1} + \dots + d_{L-1} + d_L)^3 \approx 10^{11}$ operations for LayUP and $10\,000^3 = 10^{12}$ for RanPAC, thus LayUP reduces RanPAC’s runtime complexity during inference by up to 90%.

5. Experiments

In what follows, we conduct a series of experiments to investigate the performance of our approach under different CL settings. First, we provide an overview of the datasets and implementation details in Sec. 5.1. Second, we perform empirical comparisons of our method with strong recent baselines under the CIL and the more challenging OCL settings in Sec. 5.2 and Sec. 5.3, respectively. Third, we conduct ablation studies in Sec. 5.4. We conclude our empirical analysis by investigating the contribution of different choices of k last layers to the classifier performance in Sec. 5.5.

5.1. Datasets and implementation details

Following prior works [40, 64, 65, 69, 72], we experiment with two ViT-B/16 [12] models, the former with

Method	CIFAR	IN-R	IN-A	CUB	OB	VTAB	Cars
Joint full FT	93.6	86.6	71.0	91.1	80.3	95.5	83.7
Joint linear probe	87.9	71.2	56.4	89.1	78.8	90.4	66.4
L2P [65]	84.6	72.5	42.5	65.2	64.7	77.1	38.2
DualPrompt [64]	81.3	71.0	45.4	68.5	65.5	81.2	40.1
CODA-P [56]	86.3	75.5	-	-	-	-	-
ADAM [72]	87.6	72.3	52.6	87.1	74.3	84.3	41.4
SLCA [69]	91.5	77.0	59.8*	84.7	73.1*	89.2*	67.7
RanPAC [40]	92.2	78.1	61.8	90.3	79.9	92.6	77.7
LayUP	91.5	82.1	63.1	88.1	77.6	94.6	82.0
Ablations							
w/o FSA	88.7	73.4	51.6	86.9	77.0	92.9	77.5
$k = 1$	90.4	79.0	61.5	86.0	74.3	93.1	75.5
$k = 1$, w/o FSA	86.5	69.2	55.6	85.4	72.8	92.2	69.2
LayNMC	88.1	75.2	56.2	85.5	71.8	87.5	57.6

Table 1. Average accuracy (%) after training: Comparison of prompting, backbone fine-tuning, and class-prototype methods for the CIL setting. Results are taken from [40] except results for SLCA and Coda-P, which are taken from the respective papers. Results for SLCA marked with (*) were reproduced using the officially released code. The remaining results for CODA-P could not be reproduced due to excessive GPU memory requirements for training (>24GB).

self-supervised pre-training on ImageNet-21K (**ViT-B/16-IN21K**) [52], the latter with additional supervised fine-tuning on ImageNet-1K (**ViT-B/16-IN1K**) [30]. During adaptation in the CIL setting, PETL parameters are trained using a batch size of 48 for 20 epochs and Adam [26] with momentum for optimization. We employ learning rate scheduling via cosine annealing, starting with 0.03. We train five prompt tokens for VPT and use a bottleneck dimension of 16 for AdaptFormer. All baselines use the same aforementioned ViT-B/16 backbones and are trained on seven representative split datasets CIFAR-100 (**CIFAR**) [29], ImageNet-R (**IN-R**) [16], ImageNet-A (**IN-A**) [17], CUB-200 (**CUB**) [60], OmniBenchmark (**OB**) [70], Visual Task Adaptation Benchmark (**VTAB**) [68], and Stanford Cars-196 (**Cars**) [28]. None of the methods compared in Tab. 1 and Tab. 2 require a data buffer for rehearsal. We use $T = 10$ for all datasets, except $T = 5$ for VTAB, which is a benchmark composed of five different datasets to be learned consecutively. (We report additional results for $T = 5$ and $T = 20$ in Appendix C.4.) VTAB can be considered as a hybrid class- and domain-incremental learning benchmark according to the definition in [58], as label spaces of each task are partially overlapping. Dataset statistics can be found in Appendix B. For comparison, we use the average accuracy [36] metric $A_t = \frac{1}{t} \sum_{i=1}^t R_{t,i}$, with $R_{t,i}$ denoting the classification accuracy on the i^{th} task after training on the

Method	CIFAR	IN-R	IN-A	CUB	OB	VTAB	Cars
Sequential FT	12.8	5.3	6.9	5.1	3.3	7.3	10.9
NMC [72]	83.4	61.2	49.3	86.7	73.2	88.4	37.9
RanPAC $_{\lambda=0}$ [40]	88.7	70.6	1.4	0.9	76.9	9.3	0.6
RanPAC $_{\lambda=1}$ [40]	88.7	70.6	0.9	0.7	76.9	9.0	0.6
LayUP$_{\lambda=0}$	88.7	73.4	40.9	86.4	77.0	10.4	66.3
LayUP$_{\lambda=1}$	88.7	73.4	51.6	86.9	77.0	92.9	77.5
Ablations							
$k = 1, \lambda = 0$	86.5	69.6	55.6	85.4	72.8	92.2	69.2
$k = 1, \lambda = 1$	86.5	69.6	55.6	85.4	76.3	92.3	69.2
LayNMC	80.8	59.7	50.4	82.9	69.3	85.9	40.0

Table 2. Average accuracy (%) after training: Comparison of class-prototype methods for the OCL setting. Results for RanPAC and NMC are reproduced according to the officially released code in [40].

t^{th} task. All experiments are conducted with $k = 6$ unless stated otherwise. Final accuracy A_T after learning the last task is reported for random seed 1993 and the best combination of PETL method and ViT backbone (similar to [40, 72]) in the main paper. We refer to Appendix C.1 for analysis of each A_t , forgetting measures, and variability across random initialization, and to Appendix C.3 for average accuracy over training for different PETL methods and pre-trained backbones in the CIL setting.

5.2. Performance in the CIL setting

We first compare LayUP with several prompt learning, fine-tuning, and class-prototype methods for CL. We additionally report results for joint training of a randomly initialized linear probe and joint full fine-tuning (FT) of the ViT-B/16 backbone. Final accuracy scores are shown in Tab. 1.

LayUP surpasses all baselines for four of the seven split datasets. Interestingly, these four datasets (IN-R, IN-A, VTAB, Cars) have two distinctive characteristics: First, they have the highest domain gap (indicated by the cosine distance to the pre-trained *mini*ImageNet domain for the offline setting and 50 shots, cf. [44]) among all datasets. Second, they have significantly less training data than CIFAR, OB, and CUB (cf. Appendix B). The former confirms our initial hypothesis that intra-layer representations are more domain-invariant and consequently more robust to large distributional shifts. The latter indicates that especially in the low-data regime, in contrast to other methods that tend to overfit to the target domain, LayUP constructs class prototypes and decision boundaries that generalize well even if the amount of training data is scarce compared with the data used for ViT pre-training. It is further noteworthy that RanPAC, which is the only baseline that LayUP does not consistently outperform, is considerably more expensive regarding both mem-

ory and computation (cf. Sec. 4.3).

5.3. Performance in the OCL setting

We are interested in assessing the performance of our method in the challenging OCL setting, where only a single pass over the continual data stream is allowed. All baselines are class-prototype methods for continual learning on a frozen embedding network (thus we omit FSA stages for RanPAC, and our approach), except for sequential fine-tuning, where we update the parameters of the pre-trained ViT via cross-entropy loss and Adam [26] optimizer during a single epoch. As the choice of ridge regression [18] parameter as used in RanPAC and our method requires prior knowledge about the downstream continual data—which we do not have in a realistic streaming learning setting—we compare with two simplified versions of the Gram matrix inversion $(\mathbf{G} + \lambda \mathbf{I})^{-1}$ (cf. Eq. (5) and Eq. (2)): In the first variant, we omit λ -based regularization completely, such that $\lambda = 0$ and the Gram matrix is inverted without regularization (i.e., \mathbf{G}^{-1}). In the second variant, e.g., as used in [44], we compare with $\lambda = 1$, such that we linearly transform identity-regularized class prototypes (i.e., $(\mathbf{G} + \mathbf{I})^{-1}$).

As indicated in Tab. 2, unrestricted fine-tuning of the backbone is detrimental to the generalizability of the pre-trained embeddings and leads to forgetting and low performance. Consequently, the sequential FT baseline is outperformed by class-prototype methods in the OCL setting for most benchmarks. LayUP is largely robust to missing regularization ($\lambda = 0$), with the exception of a drop in performance on VTAB. Moreover, it maintains a high performance across all benchmarks for regularization with $\lambda = 1$. On the contrary, RanPAC shows high variability in performance across benchmarks under the absence of regularization and for regularization with $\lambda = 1$, partly resulting in near-zero accuracy. Such results can be explained by the fact that the high-dimensional Gram matrix obtained after random projections in RanPAC overfits the training data and thus struggles to generalize, especially under a complete lack of regularization. The higher overall robustness to different OCL settings makes LayUP a favorable approach in continual learning scenarios where the length of the input stream and the nature of the data might not be known in advance.

5.4. Ablation Study

An extensive ablation study is provided in the bottom parts of Tab. 1 and Tab. 2. We particularly analyze changes in performance among three different dimensions: (i) Using intra-layer representations, (ii) leveraging second-order feature statistics via Gram matrix inversion, and (iii) applying first session adaptation to bridge the domain gap to downstream CL tasks. To address (i), we experiment with LayUP $_{k=1}$ as a baseline, which takes the last layer representations for classification only. We further construct LayNMC as a baseline

k	CIFAR	IN-R	IN-A	CUB	OB	VTAB	Cars
1	89.0	79.4	61.6	85.3	71.6	93.1	75.5
2	89.5	80.5	63.1	86.3	72.8	93.8	78.2
3	90.1	80.8	61.2	87.5	75.1	94.1	80.7
4	90.2	81.5	62.6	87.4	75.6	94.5	81.3
5	90.7	81.4	63.2	87.5	77.1	93.5	81.7
6	91.0	81.5	62.7	87.2	77.0	93.4	82.0
7	91.0	81.5	62.4	87.5	77.1	93.5	82.7
8	91.1	81.3	62.6	87.2	77.7	93.6	81.7
9	91.1	82.0	62.6	87.5	78.0	94.7	82.4
10	90.7	81.5	62.8	87.6	77.7	94.7	81.8
11	90.8	82.2	63.9	87.1	70.3	94.6	81.6
12	90.7	82.0	63.9	87.1	77.5	93.6	81.3

Table 3. Average accuracy (%) after training: LayUP performance for different values of k for a pre-trained ViT-B/16-IN1K and FSA with AdaptFormer. The 1st, 2nd, and 3rd highest scores and the 1st, 2nd, and 3rd lowest scores are highlighted.

to analyze (ii), which extends the cosine similarity classifier in Eq. (1) towards intra-layer presentations. Finally, for (iii), the FSA stage (phase A in Alg. 1) is omitted.

LayUP consistently benefits from intra-layer representations for class-prototype construction, with the sole exception of training on IN-A in the OCL setting (Tab. 2). It further benefits from leveraging second-order statistics for the decorrelation of class prototypes and first session training to bridge the domain gap to downstream CL tasks. Finally, it is the combination of (i), (ii), and (iii) that yields the highest performance and thus makes LayUP a strong and competitive class-prototype method for CL with pre-trained models.

5.5. Analysis of intra-layer representation depth

To provide more insights into the impact of the intra-layer representation depth on LayUP performance, we plot in Tab. 3 the average accuracy after training for different choices of k using a ViT-B/16-IN1K backbone and AdaptFormer [5] as PETL method.

We observe that LayUP configurations with $k \geq 4$ consistently surpass the final layer classifier (i.e., $k = 1$), which shows that class-prototype methods for CL can generally benefit from intra-layer representations. Although the optimal choice of k differs for each dataset, the results are indicative of a choice of larger k . As a larger value of k directly corresponds to increased memory and runtime complexity during inference, its choice should be carefully weighed up, taking into account demands and restrictions at different dimensions.

CIFAR	IN-R	IN-A	CUB	OB	VTAB	Cars
94%	79%	81%	89%	80%	94%	83%

Table 4. Percentage of classes per dataset that are better or equally classified with LayUP for $k = 6$ compared with the last layer Gram-inverted classifier (Eq. (2)).

5.6. How universal are intra-layer prototypes?

We are interested in determining how universal the intra-layer prototypes are for classification, i.e., how broad the range of classes is that benefit from LayUP. For this purpose, we run phase B of Alg. 1 and provide the fraction of classes for each dataset that has an increased or equal final accuracy score with LayUP for $k = 6$ compared with LayUP for $k = 1$ (last layer Gram-inverted classifier, cf. Eq. (2)). Tab. 4 shows that between 80% and 94% of the total classes in each dataset are better or equally classified upon introducing intermediate representations to class-prototype construction, indicating the universal applicability of intra-layer prototypes to a broad range of tasks, classes, or domains.

6. Conclusion

In this paper, we propose LayUP, a simple yet effective rehearsal-free class-prototype method for continual learning with pre-trained models. LayUP leverages intra-layer representations of a pre-trained feature extractor to increase robustness and generalizability, especially under large domain gaps and in low-data regimes. It further computes second-order feature statistics to decorrelate class prototypes, combined with parameter-efficient first-task adaptation. Extensive experiments across a range of image classification datasets and continual learning settings demonstrate that LayUP performs strongly against competitive baselines at a small fraction of their memory and compute cost. In future work, we will further investigate integrating class-prototype methods with continual adaptation of pre-trained models beyond first-task adaptation, which is particularly beneficial for learning under multiple distributional shifts.

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)*, 2018. 2
- [2] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark Experience for General Continual Learning: a Strong, Simple Baseline. In *Advances in Neural Information Processing Systems*, pages 15920–15930. Curran Associates, Inc., 2020. 2
- [3] Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 1
- [4] Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient Lifelong Learning with A-GEM. In *International Conference on Learning Representations*, 2019. 2
- [5] Shoufa Chen, Chongjian GE, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. AdaptFormer: Adapting Vision Transformers for Scalable Visual Recognition. In *Advances in Neural Information Processing Systems*, pages 16664–16678. Curran Associates, Inc., 2022. 1, 4, 7, 5
- [6] Zhiyuan Chen and Bing Liu. Lifelong Machine Learning, Second Edition. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3), 2018. 1
- [7] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision Transformer Adapter for Dense Predictions. In *The Eleventh International Conference on Learning Representations*, 2023. 1
- [8] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote Sensing Image Scene Classification: Benchmark and State of the Art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017. 1
- [9] André-Louis Cholesky. Note sur une méthode de résolution des équations normales provenant de l’application de la méthode des moindres carrés à un système d’équations linéaires en nombre inférieur à celui des inconnues. Application de la méthode à la résolution d’un système défini d’équations linéaires. *Bulletin Géodésique*, 2(1):67–77, 1924. 5
- [10] Mircea Cimpoi, Subhansu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing Textures in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1
- [11] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyang Wu, and Rama Chellappa. Learning Without Memorizing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [12] 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 *International Conference on Learning Representations*, 2021. 2, 5
- [13] Leon Gatys, Alexander Ecker, and Matthias Bethge. A Neural Algorithm of Artistic Style. *Journal of Vision*, 16(12):326, 2016. 2
- [14] 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 (CVPR)*, 2016. 3, 4
- [15] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019. 1
- [16] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadam, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu,

- Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8340–8349, 2021. [6](#), [1](#)
- [17] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural Adversarial Examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15262–15271, 2021. [6](#), [1](#)
- [18] Arthur E. Hoerl and Robert W. Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67, 1970. [3](#), [7](#)
- [19] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-Efficient Transfer Learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. [1](#)
- [20] Edward J. Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022. [1](#)
- [21] Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. A Simple Baseline that Questions the Use of Pretrained-Models in Continual Learning. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2022. [2](#), [3](#)
- [22] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual Prompt Tuning. In *Computer Vision – ECCV 2022*, pages 709–727, Cham, 2022. Springer Nature Switzerland. [1](#), [4](#), [5](#)
- [23] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural Style Transfer: A Review. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3365–3385, 2020. [2](#)
- [24] Dahuin Jung, Dongyoon Han, Jihwan Bang, and Hwanjun Song. Generating Instance-level Prompts for Rehearsal-free Continual Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11847–11857, 2023. [2](#)
- [25] Zixuan Ke, Hu Xu, and Bing Liu. Adapting BERT for Continual Learning of a Sequence of Aspect Sentiment Classification Tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4746–4755, Online, 2021. Association for Computational Linguistics. [1](#)
- [26] Diederik P Kingma and Jimmy Lei Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015. [5](#), [6](#), [7](#)
- [27] 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*, 114(13):3521–3526, 2017. [2](#)
- [28] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D Object Representations for Fine-Grained Categorization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, 2013. [6](#), [1](#)
- [29] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. *Master’s thesis, University of Toronto*, 2009. [6](#), [1](#)
- [30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012. [6](#)
- [31] Brian Lester, Rami Al-Rfou, and Noah Constant. The Power of Scale for Parameter-Efficient Prompt Tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. [1](#)
- [32] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying Neural Style Transfer. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2230–2236. AAAI Press, 2017. Place: Melbourne, Australia. [2](#)
- [33] Zhizhong Li and Derek Hoiem. Learning without Forgetting. *Transactions on Pattern Analysis and Machine Intelligence*, 40(12), 2018. [2](#)
- [34] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & Shifting Your Features: A New Baseline for Efficient Model Tuning. In *Advances in Neural Information Processing Systems*, pages 109–123. Curran Associates, Inc., 2022. [1](#), [4](#), [5](#)
- [35] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Computing Surveys*, 55(9): 1–35, 2023. [1](#)
- [36] David Lopez-Paz and Marc’ Aurelio Ranzato. Gradient Episodic Memory for Continual Learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. [6](#), [1](#)
- [37] Zheda Mai, Ruiwen Li, Hyunwoo Kim, and Scott Sanner. Supervised Contrastive Replay: Revisiting the Nearest Class Mean Classifier in Online Class-Incremental Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3589–3599, 2021. [2](#)
- [38] Simone Marullo, Matteo Tiezzi, Marco Gori, Stefano Melacci, and Tinne Tuytelaars. Continual Learning with Pretrained Backbones by Tuning in the Input Space, 2023. arXiv:2306.02947 [cs]. [1](#)
- [39] Michael McCloskey and Neal J. Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *Psychology of Learning and Motivation - Advances in Research and Theory*, 24(C), 1989. [1](#)
- [40] Mark D. McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. RanPAC: Random Projections and Pre-trained Models for Continual Learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2023. [1](#), [2](#), [3](#), [4](#), [5](#), [6](#)

- [41] Kevin P. Murphy. *Machine learning: a probabilistic perspective*. MIT Press, Cambridge, MA, 2012. 3
- [42] M.-E. Nilsback and A. Zisserman. A Visual Vocabulary for Flower Classification. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06)*, pages 1447–1454, New York, NY, USA, 2006. IEEE. 1
- [43] Oleksiy Ostapenko, Timothee Lesort, Pau Rodriguez, Md Rifat Arefin, Arthur Douillard, Irina Rish, and Laurent Charlin. Continual Learning with Foundation Models: An Empirical Study of Latent Replay. In *Proceedings of The 1st Conference on Lifelong Learning Agents*, pages 60–91. PMLR, 2022. 2
- [44] Aristeidis Panos, Yuriko Kobe, Daniel Olmeda Reino, Rahaf Aljundi, and Richard E. Turner. First Session Adaptation: A Strong Replay-Free Baseline for Class-Incremental Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18820–18830, 2023. 3, 4, 6, 7, 2
- [45] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019. 1
- [46] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar. Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3498–3505, Providence, RI, 2012. IEEE. 1
- [47] Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. GDumb: A Simple Approach that Questions Our Progress in Continual Learning. In *Computer Vision – ECCV 2020*, pages 524–540, Cham, 2020. Springer International Publishing. 2
- [48] Vinay Venkatesh Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of Catastrophic Forgetting: Hidden Representations and Task Semantics. In *International Conference on Learning Representations*, 2021. 2
- [49] Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. Effect of scale on catastrophic forgetting in neural networks. In *International Conference on Learning Representations*, 2022. 2
- [50] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 1
- [51] 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. 2, 1
- [52] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lih Zelnik-Manor. ImageNet-21K Pretraining for the Masses. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. 6
- [53] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive Neural Networks. *arXiv:1606.04671 [cs]*, 2016. arXiv: 1606.04671. 2
- [54] Dawid Rymarczyk, Joost van de Weijer, Bartosz Zieliński, and Bartłomiej Twardowski. ICICLE: Interpretable Class Incremental Continual Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1887–1898, 2023. 2
- [55] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming Catastrophic Forgetting with Hard Attention to the Task. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018. 2
- [56] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zolt Kira. CODA-Prompt: Continual Decomposed Attention-Based Prompting for Rehearsal-Free Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11909–11919, 2023. 2, 6
- [57] James Seale Smith, Junjiao Tian, Shaunak Halbe, Yen-Chang Hsu, and Zolt Kira. A Closer Look at Rehearsal-Free Continual Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2409–2419, 2023. 3
- [58] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning, 2019. arXiv:1904.07734 [cs, stat]. 6
- [59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 4
- [60] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, California Institute of Technology, 2011. Publisher: California Institute of Technology. 6, 1
- [61] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A Comprehensive Survey of Continual Learning: Theory, Method and Application, 2023. arXiv:2302.00487 [cs]. 1
- [62] Quanzhang Wang, Renzhen Wang, Yichen Wu, Xixi Jia, and Deyu Meng. CBA: Improving Online Continual Learning via Continual Bias Adaptor. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19082–19092, 2023. 2
- [63] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-Prompts Learning with Pre-trained Transformers: An Occam’s Razor for Domain Incremental Learning. In *Advances in Neural Information Processing Systems*, pages 5682–5695. Curran Associates, Inc., 2022. 2
- [64] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. DualPrompt: Complementary Prompting for Rehearsal-Free Continual Learning. In *Computer Vision – ECCV 2022*, pages 631–648, Cham, 2022. Springer Nature Switzerland. 2, 5, 6, 1
- [65] 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 (CVPR)*, pages 139–149, 2022. 2, 5, 6
- [66] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong Learning with Dynamically Expandable

- Networks. In *International Conference on Learning Representations*, 2018. [2](#)
- [67] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual Learning Through Synaptic Intelligence. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017. [2](#)
- [68] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruysen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A Large-scale Study of Representation Learning with the Visual Task Adaptation Benchmark, 2020. arXiv:1910.04867 [cs, stat]. [6](#), [1](#)
- [69] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. SLCA: Slow Learner with Classifier Alignment for Continual Learning on a Pre-trained Model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19148–19158, 2023. [2](#), [3](#), [4](#), [5](#), [6](#), [1](#)
- [70] Yuanhan Zhang, Zhenfei Yin, Jing Shao, and Ziwei Liu. Benchmarking Omni-Vision Representation Through the Lens of Visual Realms. In *Computer Vision – ECCV 2022*, pages 594–611, Cham, 2022. Springer Nature Switzerland. [6](#), [1](#)
- [71] Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A Model or 603 Exemplars: Towards Memory-Efficient Class-Incremental Learning. In *The Eleventh International Conference on Learning Representations*, 2023. [2](#)
- [72] Da-Wei Zhou, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting Class-Incremental Learning with Pre-Trained Models: Generalizability and Adaptivity are All You Need, 2023. arXiv:2303.07338 [cs]. [2](#), [3](#), [4](#), [5](#), [6](#), [1](#)

Read Between the Layers: Leveraging Intra-Layer Representations for Rehearsal-Free Continual Learning with Pre-Trained Models

Supplementary Material

Appendices

A. Training and Implementation Details

A.1. Optimization of ridge regression parameter λ

We perform the optimization over the regression parameter λ (cf. Alg. 1) as follows: For every task t , we perform a random 80:20 split of the training data stratified by ground truth labels. We then update \mathbf{G} and $\mathbf{c}_y \forall y \in \mathcal{Y}_t$ (which were already updated during task $1, \dots, t-1$ training) for the first 80% of training data of task t and then choose the value of $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2, 10^3\}$ that yields highest per-sample accuracy on the remaining 20% of training data. Finally, we update \mathbf{G} and $\mathbf{c}_y \forall y \in \mathcal{Y}_t$ for those 20% of training data of task t and repeat the same process with $t+1$.

A.2. Data augmentation

During the training process, data augmentation was applied to all datasets, incorporating random cropping of the images to varying sizes, ranging from 70% to 100% of their original dimensions, while maintaining an aspect ratio between 3:4 and 4:3. After the resizing, images were randomly flipped horizontally and brightness, contrast, saturation, and hue were varied randomly in a 10% range. Finally, the images were center-cropped to 224x224 pixels for all datasets, except for CIFAR-100, where images were directly resized from the original 32x32 to 224x224 pixels. During inference, images of all datasets were resized to 224x224 pixels without further modification.

A.3. Compute resources

All experiments in this work were conducted on an Ubuntu system version 20.04.6 with a single NVIDIA GeForce RTX 3080 Ti (12GB memory) GPU.

B. Datasets

We provide a summary of the datasets compared in the main experiments in Tab. 5. CIFAR-100 (**CIFAR**) contains 100 classes of natural images of different domains and topics and can be considered relatively in-distribution with the pre-train domain of ImageNet-1K and ImageNet-21K. ImageNet-R (**IN-R**) contains image categories overlapping with ImageNet-1K, but is a selection of out-of-distribution samples for the pre-train dataset that are either hard examples

or newly collected data of different styles. ImageNet-A (**IN-A**) likewise has overlapped categories with ImageNet-1K, but comprises real-world adversarially filtered images that fool existing ImageNet pre-trained classifiers. The Caltech-UCSD Birds-200-2011 (**CUB**) dataset is a specialized collection of labeled images of 200 bird species, encompassing a diverse range of poses and backgrounds. OmniBenchmark (**OB**) serves as a compact benchmark designed to assess the generalization capabilities of pre-trained models across semantic super-concepts or realms, encompassing images of 300 categories that represent distinct concepts. The Visual Task Adaptation Benchmark (**VTAB**) as used in our work is a composition of the five datasets Resisc45 [8], DTD [10], Pets [46], EuroSAT [15], and Flowers [42], that are learned consecutively to emulate the emergence of different domains. The Stanford Cars-196 (**Cars**) dataset comprises images of cars belonging to one of 196 unique combinations of model and make.

	Original	CL	T	N_{train}	N_{val}	C
CIFAR	[29]	[51]	10	50 000	10 000	100
IN-R	[16]	[64]	10	24 000	6 000	200
IN-A	[17]	[72]	10	6 056	1 419	200
CUB	[60]	[72]	10	9 465	2 323	200
OB	[70]	[72]	10	89 668	5 983	300
VTAB	[68]	[72]	5	1 796	8 619	50
Cars	[28]	[69]	10	8 144	8 041	196

Table 5. Overview of datasets: Original publication, formulation for the CL setting, number of tasks (T) in the main experiments in Sec. 5, number of training samples (N_{train}), number of validation samples (N_{val}), and number of classes (C).

C. Additional Results

We use the following two metrics in our work for experimental evaluation: *Average accuracy* A_t (following the definition of [36]) and *average forgetting* F_t (following the definition of [3]). Average accuracy is defined as

$$A_t = \frac{1}{t} \sum_{i=1}^t R_{t,i}, \quad (7)$$

where $R_{t,i}$ denotes the classification accuracy on task i after training on task t . Using the same notion of $R_{t,i}$, average

forgetting is defined as

$$F_t = \frac{1}{t-1} \sum_{i=1}^{t-1} \arg \max_{t' \in \{1, \dots, t-1\}} R_{t',i} - R_{t,i} \quad (8)$$

As the number of different classes to choose from in a CIL and an OCL setting grows as new tasks are being introduced, A_t tends to decrease and F_t tends to rise over the course of training. While we only report on average accuracy metric in the main paper, the experimental results in Appendix C.1, Appendix C.2, and Appendix C.3 are reported both with respect to average accuracy and average forgetting.

C.1. Variability across seeds and performance over time

LayUP accuracy and forgetting for all seven datasets over the course of class-incremental training (Phase B in Alg. 1) is depicted in Fig. 3. There are almost no differences between final results after the last task across seeds, which indicates LayUP being insensitive to different task orders and initialization of PETL parameters. The highest variability can be observed for the VTAB benchmark, where each task represents a completely new domain. Therefore, the task order—and consequently the domain that the representations are adjusted to during first session adaptation—has a greater influence on the generalization capabilities of the model during CL. As expected, the average forgetting increases as more tasks are being introduced, as the model has more classes to tell apart during inference.

C.2. Experiments with different layer choice k

To analyze the learning behavior over time in the CIL setting among different choices of maximum representation depth k for prototype construction, we plot average accuracy and forgetting for $k = 1$ (last layer only), $k = 6$, and $k = 12$ (all layers) in Fig. 4. Clearly, the classification performance based on last layer representations only is inferior to intra-layer representations as used in LayUP, as it leads to both a lower accuracy and a higher forgetting rate. At the same time, there is no difference in performance between $k = 6$ and $k = 12$ evident, indicating that early layers of the model do not add meaningful knowledge to the classification. Given that the choice of k is subject to a trade-off between performance gain and memory and computational cost, the results confirm $k = 6$ as used in the main experiments in Sec. 5 to be a reasonable choice.

C.3. Experiments with different backbones and PETL methods

Following prior works [40, 72], we experiment with different PETL methods and ViT-B/16 models for LayUP, as the additional benefit from additional fine-tuning of the representations differs depending on the characteristics of the downstream domain [44]. Results are shown in Fig. 5.

In all but one dataset, we found adapter methods (AdaptFormer) to be superior to be superior to prompt learning (VPT) or feature modulation (SSF). However, there is no combination of PETL method and pre-trained model that generally outperforms all other variants. Such results confirm the findings of [44] and underline the importance of considering different pre-training schemes and strategies for additional fine-tuning.

C.4. Experiments with different task counts T

To show whether a benefit of intra-layer representations can be observed for different task counts, we compare average accuracy scores after training for six different datasets, three different task counts ($T \in \{5, 10, 20\}$) and three different choices of k last layers for class-prototype generation. $k = 1$ corresponds to classification only based on the last layer (i.e., final) representations of the backbone, as it is done in prior work. $k = 6$ means to concatenate layer-wise features from the latter half of the network layers. Finally, $k = 12$ uses concatenated features of all layers of the pre-trained ViT for classification. Results are presented in Tab. 6.

Performance scores across task counts and datasets are consistently higher for intra-layer representations ($k = 6$ and $k = 12$) compared with last-layer-only representations ($k = 1$). However, differences in performance between $k = 6$ and $k = 12$ are not apparent (with the exception of a $T = 20$ split on the CUB dataset, which can be attributed to some sensitivity to initialization of adapter parameters). Considering that a higher k corresponds to an increased computational and memory demand as G and c_y increase in dimension, this confirms $k = 6$ as reasonable choice for maximum representation depth.

k	T	CIFAR	IN-R	IN-A	CUB	OB	Cars
1	5	89.9	80.6	62.5	85.3	73.2	75.6
	10	89.1	79.1	61.4	85.5	71.6	75.6
	20	86.3	77.8	59.5	83.5	72.2	75.5
6	5	91.3	83.0	64.5	87.6	78.0	82.2
	10	90.8	81.9	62.5	87.7	76.6	81.8
	20	90.3	80.4	61.6	69.5	77.8	81.8
12	5	91.6	82.7	62.7	87.1	77.6	81.8
	10	91.1	82.1	62.7	87.5	73.8	81.7
	20	90.5	80.3	56.8	86.0	78.3	81.9

Table 6. Average accuracy (%) after training: Comparison of different task counts T for $k = 1$ (prototype construction from last layer only), $k = 6$ (as used in Sec. 5), and $k = 12$ (prototype construction from all network layers). Scores listed are for AdaptFormer and ViT-B/16-IN1K. VTAB has a fixed number of datasets that are treated as tasks (thus $T = 5$) and is therefore omitted in the comparison.

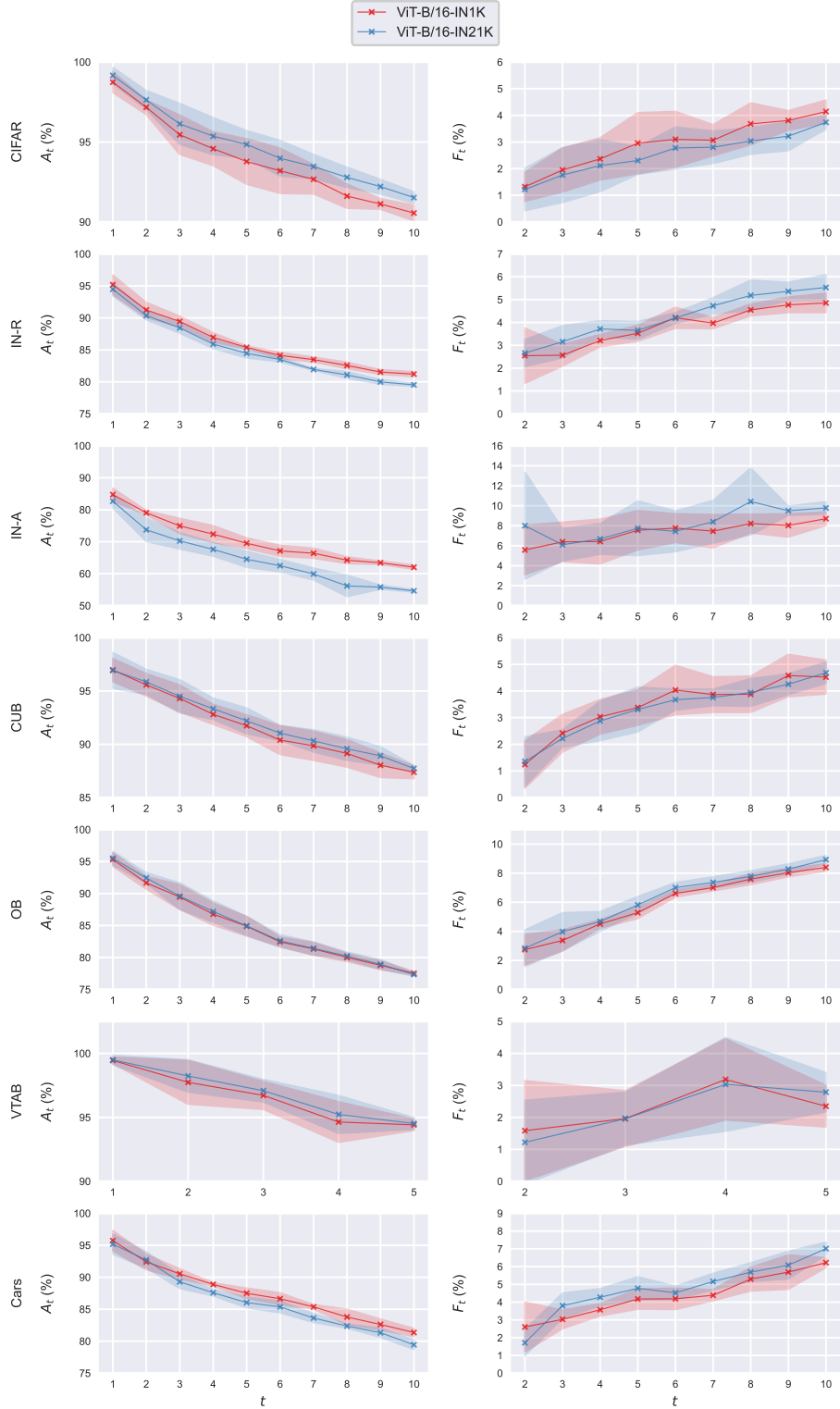


Figure 3. Average accuracy (*left*) and average forgetting (*right*) after training on each task t in the CIL setting: Variability across random seeds for each ViT-B/16 models after first session training with AdaptFormer as PETL method. Results are reported for seeds 1993-1997 to ensure reproducibility with the resulting standard error indicated by shaded area.

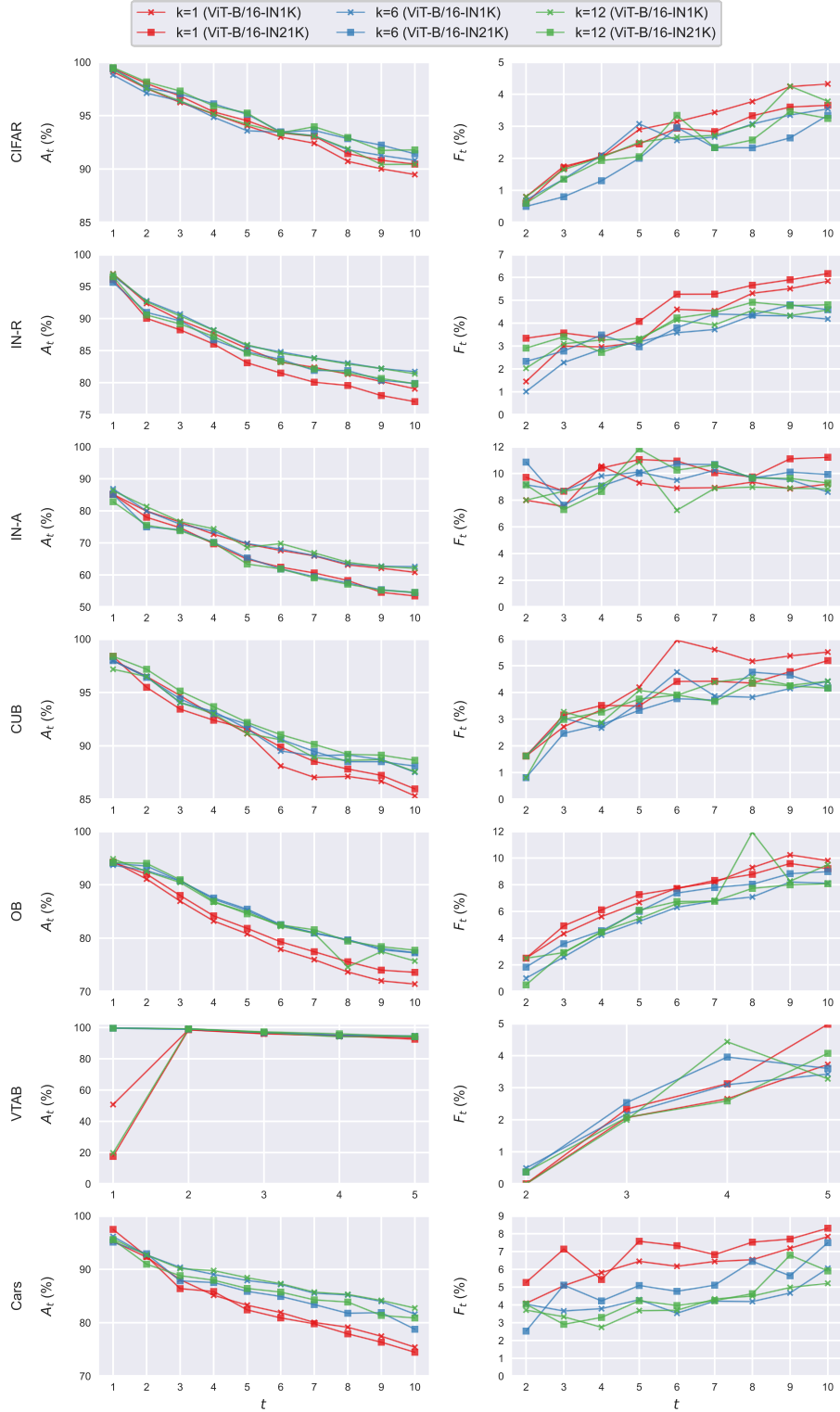


Figure 4. Average accuracy (*left*) and average forgetting (*right*) after training on each task t in the CIL setting: Comparison of different choices of k last layers for prototype construction ($k \in \{1, 6, 12\}$) and ViT-B/16 models after first session training with AdaptFormer as PETL method.

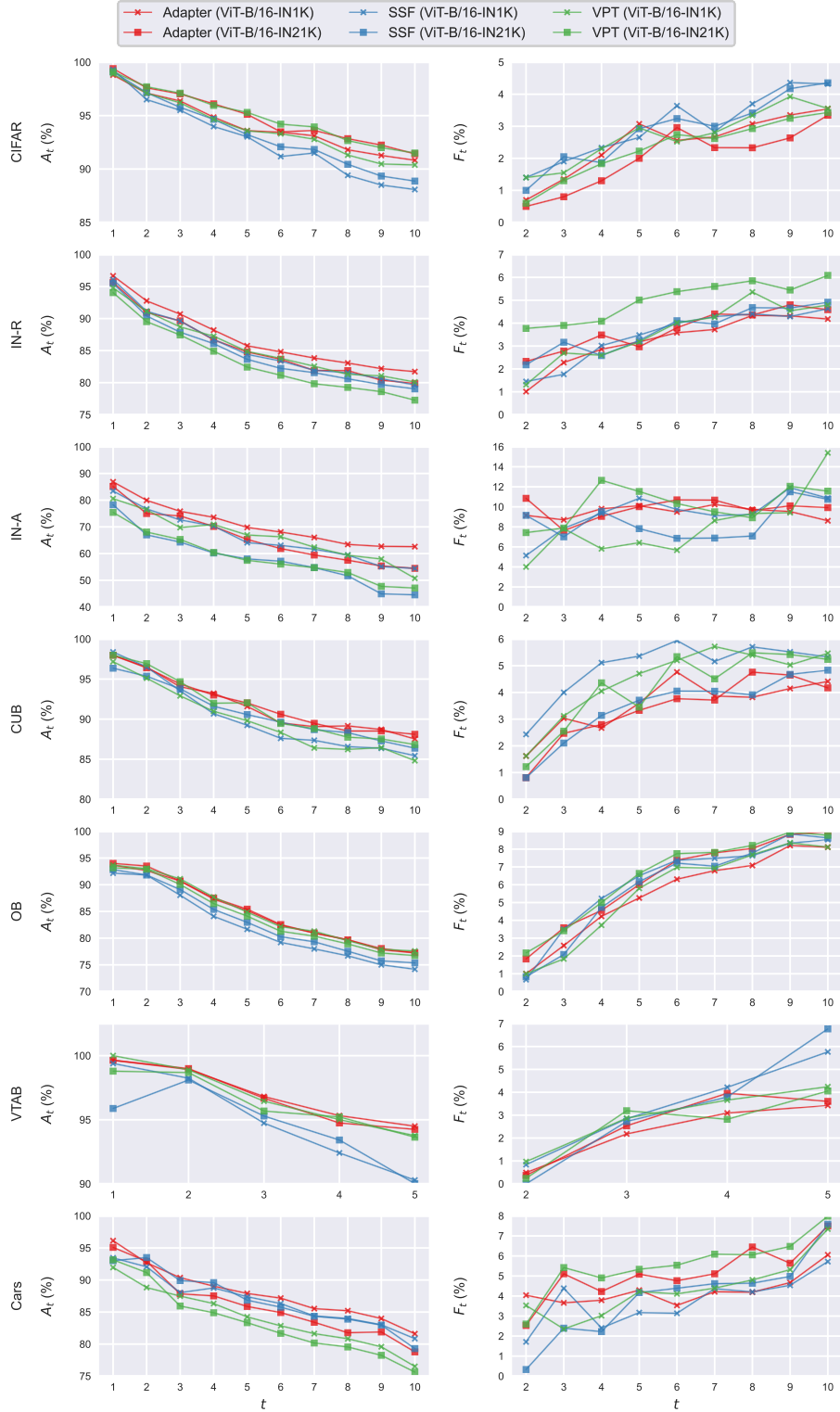


Figure 5. Average accuracy (*left*) and average forgetting (*right*) after training on each task t in the CIL setting: Comparison of different PETL methods (AdaptFormer [5], SSF [34], and VPT [22]) and ViT-B/16 models.