Modular Embedding Recomposition for Incremental Learning

Aniello Panariello
aniello.panariello@unimore.it
Emanuele Frascaroli
emanuele.frascaroli@unimore.it
Pietro Buzzega
pietro.buzzega@unimore.it
Lorenzo Bonicelli
lorenzo.bonicelli@unimore.it
Angelo Porrello
angelo.porrello@unimore.it
Simone Calderara
simone.calderara@unimore.it

AImageLab
University of Modena and Reggio Emilia
Italy

Abstract

The advent of pre-trained Vision-Language Models (VLMs) has significantly transformed Continual Learning (CL), mainly due to their zero-shot classification abilities. Such proficiency makes VLMs well-suited for real-world applications, enabling robust performance on novel unseen classes without requiring adaptation. However, fine-tuning remains essential when downstream tasks deviate significantly from the pre-training domain. Prior CL approaches primarily focus on preserving the zero-shot capabilities of VLMs during incremental fine-tuning on a downstream task. We take a step further by devising an approach that transforms preservation into enhancement of the zero-shot capabilities of VLMs. Our approach, named MoDular Embedding Recomposition (MoDER), introduces a modular framework that trains multiple textual experts, each specialized in a single seen class, and stores them in a foundational hub. At inference time, for each unseen class, we query the hub and compose the retrieved experts to synthesize a refined prototype that improves classification. We show the effectiveness of our method across two popular zero-shot incremental protocols, Class-IL and MTIL, comprising a total of 14 datasets. The codebase is available at https://github.com/aimagelab/mammoth.

1 Introduction

Vision-Language Models (VLMs) like CLIP [29] have gained significant attention for their flexibility and robust zero-shot capabilities. These characteristics make VLMs particularly appealing in applications demanding high adaptability, where data arrives incrementally, and frequent retraining is not feasible due to time or computational constraints. However, incrementally fine-tuning such models often causes degraded performance in recognizing previously seen classes (*catastrophic forgetting*) [23]. Even worse, incrementally fine-tuning VLMs gradually erodes zero-shot capabilities for unseen domains [43]. This form of forgetting diminishes the potential to generalize to classes the model has never seen, limiting its adaptability "out of the box".

To mitigate this issue, one could resort to Continual Learning (CL), with approaches that generally divide into three categories [33]. Namely, given a sequence of tasks of disjoint classes, *regularization techniques* [12, 17] penalize changes to the most significant weights. *Replay methods* [1, 2, 3, 25, 26, 30] interleave past examples, either stored in memory or generated by a model, with new data to reinforce prior knowledge. *Architectural approaches* [21, 31] add task-specific parameters to preserve learned representations. While these approaches could be applied to large pre-trained models, the high memory footprint of fine-tuning them makes it impractical. Recent CL methodologies [5, 32, 35, 36, 37] incorporate Parameter-Efficient Fine-Tuning (PEFT) techniques [10, 45] to reduce such burden.

In addition to forgetting previously acquired knowledge, fine-tuning a VLM significantly degrades its zero-shot accuracy, as new training data overwrites the broader pre-trained knowledge [39, 43]. The authors of ZSCL [43] introduce this problem under the name of Zero-Shot Continual Learning. However, methods from this field [5, 39, 43] primarily focus on preventing the deterioration of zero-shot performance and are not designed to actively improve it.

Our approach, **MoDular Embedding Recomposition** (**MoDER**), goes further by aiming to *enhance* zero-shot capabilities proactively. To generalize to unseen classes, we train experts incrementally and use them to forge new tailored models on the fly. Specifically, we leverage *model compositionality* [22, 27] and devise textual expert modules that are both **memory-efficient** and **linearly composable** in parameter space. Thanks to these experts, stored in a repository called the **foundational hub**, we can forge textual prototypes for unseen classes by reusing models learned across tasks. This provides the basis for building a modular learning system that prevents forgetting and easily adapts to unseen classes and domains.

We validate MoDER on two recognized settings – Class-Incremental Learning (Class-IL) [33] and Multi-Domain Task Incremental Learning (MTIL) [39, 43] – employing 14 diverse datasets that present various challenges, including unrelated domains, numerous fine-grained classes, and substantial shifts from the broad pre-training knowledge. To ensure that our proposal improves transfer on unseen classes without sacrificing performance on seen classes, we measure performance as both Class Incremental Transfer [5] and Final Average Accuracy. Our main contributions are as follows:

- We introduce MoDular Embedding Recomposition, an incremental approach that retains prior knowledge and actively reuses and recombines it to improve zero-shot performance.
- We introduce a hub of textual specialized experts, allowing for the creation of new textual prototypes that enhance classification for unseen classes.
- We conduct experiments on various datasets and benchmarks showcasing state-of-the-art performance for incremental fine-tuning VLMs.

2 Related Works

Class-Incremental Learning and Prompt Tuning. Learning to Prompt (L2P) [37] pioneers the use of prompts for Class-IL by introducing a shared pool of prompts, with a query function that selects the most relevant prompts for each input image. DualPrompt [36] extends L2P by introducing prefix-tuning [16] and a hierarchy of general task-specific prompts. Similarly, CODA-Prompt [32] applies prefix-tuning and a differentiable, soft mechanism for prompt selection. Among CLIP-based techniques, AttriCLIP [35] employs a pool of learnable textual prompts for the CLIP text encoder. Conversely, CGIL [5] optimizes a single textual prompt per class and incorporates generated visual features to alleviate forgetting.

Zero-Shot Continual Learning. This framework, recently introduced in [43], addresses the challenge of incrementally fine-tuning Vision-Language Models (VLMs) without compromising their zero-shot performance. To address this, the authors utilize a teacher-student strategy on a reference dataset to preserve pre-training knowledge. Their approach, termed ZSCL, requires substantial GPU memory due to the simultaneous loading of the two CLIP instances. MoE-Adapters [39] introduces multiple LoRA experts per task to mitigate these limitations. However, this solution necessitates task identity knowledge during inference, adding complexity via a learnable router module that still depends on a reference dataset. CGIL [5] employs learnable textual prompts trained on synthetic visual features to prevent forgetting past knowledge; however, its approach for unseen classes is conservative, as it still relies on hand-crafted input templates. Our approach circumvents these constraints by leveraging the composition of class-specific experts from a foundational hub.

3 Preliminaries

Setting. In Class-Incremental (Class-IL) Continual Learning, a deep model $f(\cdot;\theta)$, parameterized by θ, is trained on a sequence of T tasks. In each one, the model is presented with a training dataset $\mathcal{D}_t = \{(x^{(n)}, y^{(n)})\}_{n=1}^{N_t}$, with N_t examples and labels $y^{(n)}$ drawn from a partition of classes $\mathcal{Y}_t$. In the following, we will refer to the current task with t_c and denote $\mathcal{Y}^{[1:t_c]} = \bigcup_{t=1,\dots,t_c} \mathcal{Y}_t$ as the set of **seen** classes, namely the union of the classes learned during training, up to and including the current task t_c. Importantly, the classes are non-overlapping across tasks, meaning that $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ for $i \neq j$.

The goal in CL is to minimize the cumulative empirical risk across all tasks $\mathcal{L}_{\text{CL}} = \sum_{t=1}^{T} \frac{1}{N_t} \sum_{(x,y)\in\mathcal{D}_t} [\mathcal{L}(f(x;\theta), y)]$, where $\mathcal{L}$ commonly refers to the cross-entropy loss. However, since the model processes tasks sequentially and only has access to the training data of the current task, directly optimizing $\mathcal{L}_{\text{CL}}$ is impractical.

Zero-shot classification with CLIP. CLIP [29] is composed of two models – a visual encoder, $E^{\text{vis}}(\cdot)$, and a text encoder, $E^{\text{txt}}(\cdot)$ – trained to produce aligned representations of paired image-text data. These encoders enable *zero-shot classification*, *i.e.*, categorizing an image x into categories that may not have been seen during training. Specifically, for each candidate class within a set $\mathcal{Y}$, a text prompt is generated by embedding the class name "`[CLS]`" within a hand-crafted template such as "`a photo of a [CLS]`". These class-specific prompts are processed through the text encoder, yielding a textual embedding z_i^{txt} for the i-th class. To determine class logits, cosine similarity $\langle\cdot,\cdot\rangle$ is computed between the visual embedding $z^{\text{vis}} = E^{\text{vis}}(x)$ of the input image x and each class-level textual embedding.

Low-Rank Adaption (LoRA). Fine-tuning methods like LoRA [10] serve to adapt a pre-trained weight matrix $\theta_0{}^{(l)} \in \mathbf{R}^{d\times k}$ of a layer l in a parameter-efficient manner. In short, given

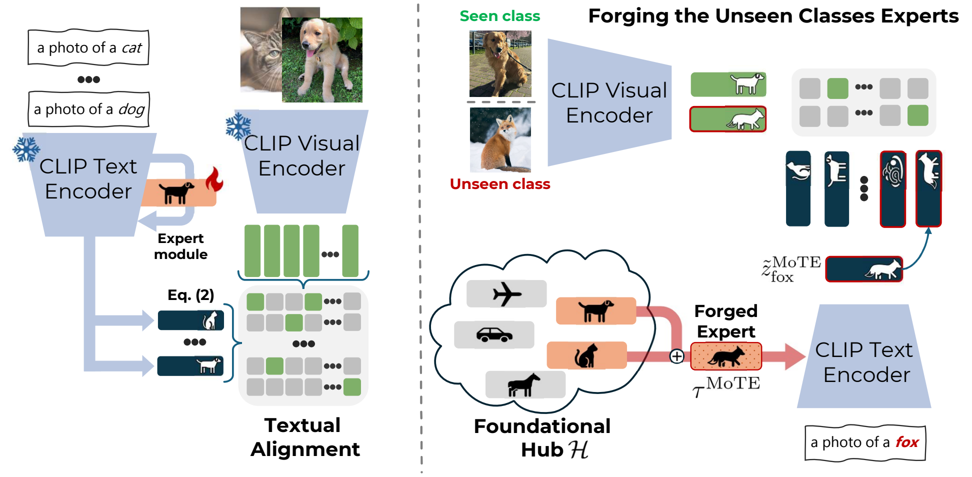

Figure 1: An overview of our approach MoDER. The left side depicts the generative modeling and Textual Alignment (TA) phases. The right side represents the forging of the embedding for the unseen classes.

a model with fine-tuned weights θ^{FT}, LoRA assumes that shifts w.r.t. pre-training weights $\tau = \theta^{\mathrm{FT}} - \theta_0$ lie on a low-dimensional manifold. Hence, the displacement τ is parametrized as low-rank matrix $\tau = BA$, where $B \in \mathbf{R}^{d \times r}$, $A \in \mathbf{R}^{r \times k}$, and $r \ll \min(d,k)$. Lowering the rank r can significantly reduce the number of learnable parameters without compromising performance.

4 MoDular Embedding Recomposition

The name of our method – **MoDular Embedding Recomposition** (**MoDER**) – builds on the concept of enhancing capabilities for unseen classes by recomposing fragments of knowledge accumulated across previous tasks. Our training framework consists of specialized experts in the form of lightweight PEFT modules, which fine-tune the CLIP text encoder. We then introduce **Textual Alignment** (**TA**, see Section 4.1), namely a training strategy that promotes the composition of different learnable modules.

During inference, we deploy a dual strategy to compute textual embeddings. Specifically, for classes from the seen set $\mathcal{Y}^{[1:t_c]}$, we use the output of the respective LoRA experts. For the unseen set, we present an approach called **Mixture of Textual Experts** (**MoTE**, see Section 4.3) that leverages compositionality to create experts on the fly. Specifically, given a **foundational hub** $\mathcal{H}$, comprising all trained experts, we merge the K experts most relevant to the target unseen class.

4.1 Textual Parameter-Efficient Specialized Experts

Given all images x in the current dataset $\mathcal{D}_{t_c}$, we **freeze** the CLIP visual encoder and extract the relative visual embeddings z^{vis}. These embeddings are used to learn **aligned** textual prototypes – one per class in the current task – ensuring that image-to-text cosine similarity allows for accurate classification. To do this, while existing methods mainly rely on prompt-tuning [44, 45], we explore a different approach and devise a distinct LoRA [10] expert module τ_i for each class. The textual embedding for the class i fine-tuned through the expert is obtained as:

$$\tilde{z}_i^{\text{txt}} = E^{\text{txt}}(p_i; \theta_0 + \tau_i) \quad \text{s.t. } \tau_i = B_i A_i \text{ , } i \in \mathcal{Y}^{[1:t_c]} \tag{1}$$

where $p_i =$ "`a photo of a [CLS]`" is a hand-crafted textual prompt for the i-th class within the seen set. Hence, an expert specializes the CLIP text encoder on input textual prompts related to the same reference class. The displacement τ_i can be understood as **task vector** [11], that is, the direction in parameter space along which the capabilities of the pre-trained model rapidly improve for the target class i. In a sense, we aim to capture unique aspects of the class i with the corresponding task vector τ_i.

It is noted that, to achieve specialization, each expert in our method builds upon their own set of weights. While this may raise concerns in memory-constrained settings, our formulation is flexible. It can be adapted to define task-specific experts (as shown in Table 2), requiring only one additional set of parameters per task. Furthermore, we can leverage highly efficient adaptation techniques, such as VeRA [14]. Results for both LoRA- and VeRA-based expert modules are reported in Table 3.

4.2 Textual Alignment (TA)

While training the textual experts for the newly introduced task, we must consider two aspects: *i)* the new textual prototypes should not interfere with the existing ones, which were trained for previous classes, in order to prevent *catastrophic forgetting* during prediction; *ii)* the experts should be trained to be composable, such that their linear combination yields meaningful outputs. In the following, we will discuss how our training strategy, called Textual Alignment (TA), manages to achieve both of the aforementioned objectives.

To avoid interference between experts and the resulting forgetting issues, we augment the training data for the current task by including highlights from the past, akin to generative replay [5]. Namely, we train the experts not only on examples from the current task's data but also on **synthetic visual embeddings** generated by lightweight generative models previously trained on past tasks. Specifically, at the onset of the current task, we train a lightweight **class-conditioned diffusion model** [9], denoted by $g_{t_c}(\cdot)$, on the features produced by the visual encoder. Notably, due to the low-dimensional nature of these features, the generator can be implemented as a lightweight Multi-Layer Perceptron (MLP), with a negligible computational cost. In particular, each diffusion model $g_{t_c}(\cdot)$ consists of an eight-layer MLP with 256 hidden channels per layer and SELU [13] as the activation function.

Thanks to the generators trained on previous tasks (one per task), we can compensate for the absence of corresponding data in later stages and create balanced training sets for each expert. In practice, after learning the generator for the current task, we construct a synthetic dataset $\mathcal{D}_{\text{SYN}}$ by combining the features generated from all the diffusion models. We then train the experts using only data from the $\mathcal{D}_{\text{SYN}}$, as discussed in the following.

Loss function. To train the PEFT experts, we aim to align the text embeddings in Equation (1) with the visual embeddings in the synthetic dataset $\mathcal{D}_{\text{SYN}}$. While a standard cross-entropy loss could be used for multi-modal alignment (as in [29, 45]), we treat each class independently by framing the problem as multiple binary classification problems. Specifically, we train the experts with the **sigmoid loss** $\mathcal{L}_\sigma$, similarly to [41]. Formally, given a data-point (z^{vis}, j) of class j, the per-sample loss becomes:

$$s_i = \langle z^{\text{vis}}, \tilde{z}_i^{\text{txt}} \rangle, \text{ where } \tilde{z}_i^{\text{txt}} \text{is from Equation (1)},$$
$$\mathcal{L}_\sigma(z^{\text{vis}}, j; \tau_i) = \textstyle\sum_{i \in \mathcal{Y}^{[1:t_c]}} \log(1 + e^{-s_i \cdot \mathbf{1}_{\{i=j\}}}), \tag{2}$$

where $\mathbf{1}_{\{i=j\}}$ is an indicator function that returns 1 if the candidate class i equals the ground truth class j of the synthetic example z^{vis}, and -1 otherwise.

We employ the sigmoid loss over cross-entropy primarily to enhance training efficiency. While cross-entropy requires a joint forward/backward pass through all experts, with the sigmoid loss, we can reformulate the problem into multiple independent binary tasks. This allows us to split each update step into manageable batches of experts, which can be distributed across distinct nodes in a multi-GPU or multi-node setup. Finally, as further discussed in the experimental section, we observe that the sigmoid loss yields a consistent yet surprising beneficial effect in terms of transfer to unseen categories.

After their training, the experts are stored in the **foundational hub** $\mathcal{H}$, which grows incrementally with each task to accommodate new knowledge. Such a hub serves as a deep module library, enabling reuse and composition of expert modules for unseen tasks.

4.3 Expert Forging via Mixture of Experts

During evaluation, we compare the visual embeddings extracted from the test images with the textual embeddings from the experts. For a class i of the seen set, we follow Equation (1) and prompt the associated expert $E^{\text{txt}}(p_i; \theta_0 + \tau_i)$.

Differently, considering a class j from the *unseen set* (*e.g.*, those present in future tasks), we forge a new textual prototype by leveraging the experience accumulated up to the current task and stored in the foundational hub $\mathcal{H}$. Specifically, we first identify the K experts in $\mathcal{H}$ most relevant to the unseen class j, then combine their **weights** to form a new expert. We term this procedure the **Mixture of Textual Experts** (**MoTE**); it proceeds as follows::

$$\tilde{z}_j^{\text{MoTE}} = E^{\text{txt}}\left(p_j; \theta_0 + \boldsymbol{\tau}_j^{\text{MoTE}}\right). \text{ where } \boldsymbol{\tau}_j^{\text{MoTE}} = \textstyle\sum_{i \in \text{top}_\text{K}(p_j)} w_{i,j} \tau_i, \tag{3}$$

Here, for an unseen class j, p_j denotes its synthetized textual prompt, while $w_{i,j}$ represents an affinity score between classes i and j. In addition, $\text{top}_\text{K}(p_j)$ returns the K experts from the seen set $\mathcal{Y}^{[1:t_c]}$ that maximize the similarity $\texttt{sim}(i,j)$. We resort to text-to-text similarity in the original CLIP space, such that $\texttt{sim}(i,j) = \langle z_i^{\text{txt}}, z_j^{\text{txt}} \rangle$. We normalize these scores across the K experts with softmax, thus obtaining the $w_{i,j}$ in Equation (3).

Improving expert capabilities. Following Equation (1), each expert is fed with the same hand-crafted prompt during training. However, this approach will likely result in over-fitting issues, with poor generalization when varying the input prompt. This would be particularly detrimental when forging the textual embedding of an unseen class j. Indeed, following Equation (3), the corresponding prompt p_j provided to each expert entails a massive domain shift for the expert i, which was trained solely on p_i. Therefore, we need a tailored strategy to enhance the robustness of the experts to domain shifts and improve their out-of-distribution capabilities, also shown in [28] to be beneficial for model compositionality.

In this respect, we combine two simple yet effective strategies. The first one is ***template augmentation***: in analogy with the concept of data augmentation, we modify the training process by randomly sampling a textual template to construct the input prompt — *e.g.*, one from the 80 templates commonly used for ImageNet zero-shot tests [29].

Algorithm 1 MoDER — Training Phase

for each dataset $\mathcal{D}_{t_c},\ t_c \in 1,\dots,T$ **do**
  Fit g_{t_c} on $\{E^{\mathrm{vis}}(x) \mid x \in \mathcal{D}_{t_c}\}$
  $\mathcal{D}_{\mathrm{SYN}} = \{z^{\mathrm{vis}} \sim g_i, \forall\, i \in 1,\dots,t_c\}$
  # Textual Alignment
  for `it` $:=1,\dots$ **do**
    Sample random prompt $p_i\ \forall\, i \in \mathcal{Y}$
    $\tilde{z}_i^{\mathrm{txt}} = E^{\mathrm{txt}}(p_i;\theta_0+\tau_i)$
    Sample a batch from $\mathcal{D}_{\mathrm{SYN}}$
    Update expert τ_i with $\mathcal{L}_\sigma$ (Equation (2))

Algorithm 2 MoDER — Forging

for each class i in $\mathcal{Y}_{\mathrm{seen}}$ **do**
  $\mathcal{Z}[i] \leftarrow E^{\mathrm{txt}}(p_i;\theta_0+\alpha\tau_i)$
 # Mixture of Textual Experts
for each class j in $\mathcal{Y}_{\mathrm{unseen}}$ **do**
  $\mathrm{top}_{\mathrm{K}}(p_j) \leftarrow$ top-K experts from $\mathcal{H}$
  $\mathcal{Z}[j] \leftarrow$ Combine their weights
  $\tilde{z}_j^{\alpha\text{-MoTE}} \leftarrow E^{\mathrm{txt}}(p_j;\theta_0+\alpha\boldsymbol{\tau}_j^{\mathrm{MoTE}})$
$z^{\mathrm{vis}} \leftarrow E^{\mathrm{vis}}(x_{\mathrm{new}})$ ▷ ***# classification***
$y_{\mathrm{pred}} \leftarrow \arg\max_{y\in\mathcal{Y}} \langle z^{\mathrm{vis}}, \mathcal{Z}_y\rangle$

Secondly, inspired by [38], we enhance OOD skills by ensembling the weights $\theta_i = \theta + \tau_i$ of each expert with those of the original zero-shot model, represented by θ_0:

$$\theta_i = (1-\alpha)\theta_0 + \alpha\theta_i = \theta_0 + \alpha\tau_i, \tag{4}$$

where $\alpha \in [0,1]$. We refer to this step as **α-smoothing** and denote by $\tilde{z}_j^{\alpha\text{-MoTE}}$ the embedding from the *smoothed* experts. With this, the final textual embedding $\tilde{z}_j^{\alpha\text{-MoTE}}$ can be generated with a single forward pass with weights $\theta + \alpha\boldsymbol{\tau}_j^{\mathrm{MoTE}}$, as follows:

$$\tilde{z}_j^{\alpha\text{-MoTE}} = E^{\mathrm{txt}}(p_j;\theta_0+\alpha\boldsymbol{\tau}_j^{\mathrm{MoTE}}). \tag{5}$$

Training and inference algorithms are provided in Algorithms 1 and 2 respectively.

5 Experiments

We evaluate MoDER on Class-Incremental Learning (Class-IL) [33] and Multi-domain Task Incremental Learning (MTIL) [39, 43]. Both involve a sequence of image classification tasks, assessing forgetting on the old tasks and generalization to unseen ones. MTIL evaluates transfer capabilities across distinct domains; in Class-IL, the unseen set comes from the same image domain observed during training (*e.g.*, satellite imagery).

Class-Incremental Learning (Class-IL). We employ five datasets with varying degrees of alignment with the pre-training. *Split Imagenet-R* [8] (200 classes split into 10 tasks, as in [32, 37, 42]) includes a subset of ImageNet classes, but with a significant domain shift. We also consider two fine-grained datasets, *Split Cars-196* [15] and *Split CUB-200* [34] (10 tasks each), with only a few samples per class. Additionally, we evaluate on two out-of-distribution datasets: *Split EuroSAT* [6, 7] (RGB satellite images for land cover classification split into 5 binary tasks), and *Split ISIC* [4] (6 skin disease classes, split into 3 binary tasks).

Multi-domain Task Incremental Learning. MTIL comprises 11 consecutive tasks, each learning on a distinct dataset: *i.e.*, Aircraft, Caltech101, CIFAR100, DTD, EuroSAT, Flowers, Food, MNIST, OxfordPet, StanfordCars, and SUN397. The datasets are evaluated under two task orders proposed by the original authors (Order-I and Order-II, with our results for

Model	IN-R	Cars	CUB	ESAT	ISIC	Avg.
CLIP [29]	82.1	66.2	50.9	55.0	22.4	55.3
AttriCLIP [35]	85.7	74.0	54.1	59.7	24.1	61.0
MoE Ad. [40]	88.2	75.8	61.7	55.8	21.1	60.5
ZSCL [43]	85.3	72.5	62.8	69.7	25.3	63.1
CGIL [5]	86.7	78.8	66.3	71.5	48.2	70.3
MoDER (LoRA)	**89.7**	**87.0**	**73.2**	74.0	**52.8**	**75.3**
MoDER (VeRA)	89.6	86.9	72.9	**74.7**	51.7	75.2

Table 1: The Class Incremental Transfer on the tested benchmarks.

Model	Transf.	Δ	Avg.	Δ	Last	Δ
CLIP	69.4	0.0	65.3	0.0	65.3	0.0
Cont. FT	44.6	-24.8	55.9	-9.4	77.3	+12.0
LwF	58.9	-10.5	64.7	-0.6	74.6	+9.3
Wise-FT	52.3	-17.1	60.7	-4.6	77.7	+12.4
ZSCL	68.1	-1.3	75.4	+10.1	83.6	+18.3
MoE Ad.	68.9	-0.5	76.7	+11.4	85.0	+19.7
MoDER	**69.7**	+0.3	**76.9**	+11.6	**85.8**	+20.5

Table 2: Accuracy of various approaches in the MTIL setting (Order I).

Model	IN-R	Cars	CUB	ESAT	ISIC	Avg.
CLIP [29]	81.9	65.0	50.5	53.3	26.6	55.5
CODA-P. [32]	78.9	45.2	72.2	63.7	47.4	61.5
AttriCLIP [35]	87.4	75.6	58.3	72.3	28.3	64.4
SLCA [42]	85.5	73.5	**87.8**	93.6	63.8	80.8
TMC [19]	63.2	39.9	63.4	65.0	48.9	56.1
Inf-LoRA [18]	84.4	58.0	80.4	79.7	56.1	71.7
MoE Ad. [40]	**90.7**	77.8	65.0	80.6	34.5	69.7
STAR-P. [24]	89.2	86.5	85.2	94.2	67.4	84.5
ZSCL [43]	89.1	77.7	62.4	79.1	34.1	68.5
CGIL [5]	89.4	89.3	83.1	96.2	73.0	86.2
MoDER (LoRA)	89.7	**90.1**	83.7	**96.4**	76.3	**87.2**
MoDER (VeRA)	89.4	89.6	82.7	96.3	**76.4**	86.9

Table 3: The Final Avg. Accuracy on the tested benchmarks.

Order-II included in the supplementary material). Unlike Class-IL, MTIL relaxes the constraint of unknown task identities at test time. This adjustment simplifies the challenge posed by the 1201 classes spread across diverse domains, making evaluation more manageable.

5.1 Implementation details

For a fair comparison, all methods use the same backbone. Specifically, in the Class-IL and CI-Transfer setting, we follow [5] and employ OpenAI's CLIP with ViT-L/14 (we also resport results using a ViT-B/16 in the supplementary material). In contrast, in the MTIL setting, we follow [39, 43] and use OpenAI's CLIP with ViT-B/16.

Each diffusion model is trained from scratch for 30K iterations using the AdamW optimizer [20] with a learning rate of 1×10^{-3} and weight decay of 1×10^{-2}. To create the synthetic dataset $\mathcal{D}_{\text{SYN}}$, for each class, we sample approximately 15K embeddings batched into sizes of 512. During the TA phase, we train the LoRA using Adam as optimizer with a learning rate of 1×10^{-4} for Split Imagenet-R, Split Cars-196 and Split CUB-200, 1×10^{-3} for Split EuroSAT and Split ISIC. The LoRA rank is fixed at 16. The experiments with VeRA adopt the same hyperparameters, except the learning rates: 1×10^{-3} for Split Imagenet-R, Split Cars-196 and Split CUB-200, and 1×10^{-2} for Split EuroSAT and Split ISIC.

5.2 Evaluation metrics

In the Class-Incremental Learning (Class-IL) setting, the task identities remain unknown during inference. To assess performance on unseen classes, we use the *Class Incremantal*

	IN-R	Cars	CUB	ESAT	ISIC	Avg.
MoDER	**89.7**	**87.0**	**73.2**	**74.0**	**52.8**	**75.3**
CE loss	80.9	71.2	66.8	65.7	50.8	67.1
No TAug	89.7	87.0	72.0	73.6	49.1	74.3
$\alpha = 0$	88.0	86.0	72.3	55.0	34.7	67.2

Table 4: Ablation analysis on CI-Transfer using LoRA-based experts.

Model	Train. Params	GPU (MiB)
LWF [17]	149.6 M ($\times$7.1)	32172 ($\times$6.7)
ZSCL [43]	149.6 M ($\times$7.1)	26290 ($\times$5.5)
MoE Ad. [39]	59.8 M ($\times$2.9)	22358 ($\times$4.7)
Textual Alignment	16.2 M ($\times$0.8)	4748 ($\times$0.99)
generator	4.7 M ($\times$0.2)	45 ($\times$0.01)
MoDER	20.9 M ($\times$1)	4793 ($\times$1)

Table 5: Trainable parameters and GPU memory usage for models for MTIL.

Transfer (CI-Transfer) metric [5], which measures accuracy on tasks not yet encountered by the model (see Table 1). Additionally, we report results on the MTIL benchmark [39, 43] in Table 2. For this benchmark, we report the three standard evaluation metrics: *Transfer*, which captures changes in zero-shot accuracy; *Average*, which tracks average accuracy throughout the incremental training; and *Last*, representing accuracy on the final task. Finally, in Table 3, we present results in terms of *Final Average Accuracy* scores to assess the impact of the different methods on the seen classes. Results are averaged over three different runs, with standard deviations and further details in the supplementary material.

5.3 Comparison with the State of the Art

Competing Methods. We compare MoDER against several established CL methods from recent literature. We always include a baseline showing the zero-shot performance of CLIP. Following the previous literature, in the MTIL setting we include the Continual FT baseline, which fine-tunes CLIP without any mechanisms for preventing forgetting.

Results. Our method, MoDER, demonstrates strong and consistent performance across all benchmarks. In particular, we achieve a clear and substantial lead in terms of CI-Transfer, outperforming all CLIP-based methods (Table 1). Furthermore, Figure 2 illustrates the *CI-Transfer* progress throughout training, revealing a steeper increase in zero-shot accuracy compared to other methods, highlighting the effectiveness of our transfer technique.

In the MTIL benchmark, MoDER also excels, surpassing existing approaches across the *Average* and *Last* metrics (Table 2). Notably, it not only prevents zero-shot degradation but improves zero-shot performance, outperforming even CLIP's original zero-shot accuracy, as shown by the *Transfer* metric. Finally, in the standard Class-IL setting (Table 3), MoDER achieves the best average performance, albeit by a narrower margin, consistently enhancing both zero-shot generalization to new classes and retention on previously seen ones.

5.4 Ablative studies

We present in Table 4 the ablation studies on the Class Incremental Transfer metric.

Sigmoid loss *vs* cross entropy loss. Besides the advantage of reducing the computational and memory overhead, we find that the use of the sigmoid loss **significantly contributes** to the performance of MoDER. This gain is unique to the CI-Transfer metric, as the contribution on the Final Average Accuracy is instead marginal. We conclude that the sigmoid loss, which models each class label independently, provides a more suitable learning objective for experts intended to be composed. Since each class is learned as an independent function,

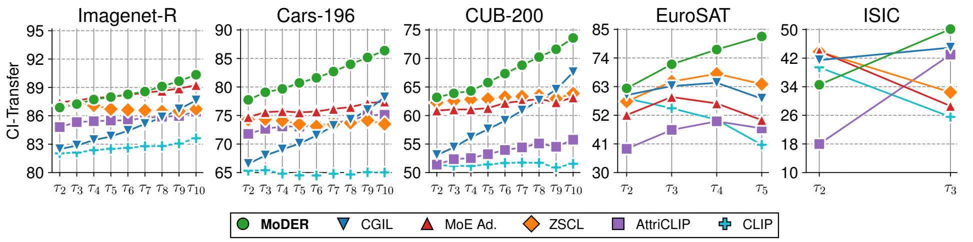

Figure 2: For various benchmarks, the accuracy trend in Class-Incremental transfer indicates the model's effectiveness in transferring to unseen classes in future tasks. A higher trend reflects greater effectiveness in adapting to unseen classes.

with the sigmoid loss experts can specialize in distinct concepts and be reused modularly in novel combinations. Similarly, Template Augmentation improves performance in CI-Transfer, albeit to a lesser extent than the sigmoid loss.

Preservation of OOD performance. The results indicate that α-smoothing significantly contribute to the performance on unseen classes. Indeed, when no smoothing is applied ($\alpha = 0$), the model achieves inferior performance, especially on out-of-domain datasets.

On the computational cost of MoDER. As shown in Table 5, MoDER achieves strong performance with substantially fewer trainable parameters and lower GPU memory usage than other baselines. This efficiency highlights its scalability and suitability for large-scale class-incremental scenarios. All results are measured on a single NVIDIA 3060 GPU with 12GB of memory. During **inference**, MoDER incurs the same overhead as the base CLIP model. Unlike other models such as STAR-Prompt [24], MoE-Adapters [40], and AttriCLIP [35], which require two forward passes, MoDER requires only a single forward pass on the visual encoder, as the text embeddings can be computed once and cached for future reuse.

6 Conclusion

This work builds on the concept of a foundational hub: a virtual, expandable space for incrementally learned models that support the re-use and creation of new expert models. The results in Section 5.3 demonstrate the effectiveness in mitigating forgetting of previous concepts and facilitating the creation of new ones. Beyond performance, our approach does not incur privacy concerns, is suitable for online scenarios, and is compatible with distributed computing environments since the experts can be trained independently.

Acknowledgements

We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support. This work was supported by the Fortissimo Plus (FFplus) project Grant Agreement No. 101163317), under the European High-Performance Computing Joint Undertaking (EuroHPC JU) and the Digital Europe Programme. The authors gratefully acknowledge access to compute resources enabled by FFplus. Funded by the European Union.

References

[1] Giovanni Bellitto, Federica Proietto Salanitri, Matteo Pennisi, Matteo Boschini, Lorenzo Bonicelli, Angelo Porrello, Simone Calderara, Simone Palazzo, and Concetto Spampinato. Saliency-driven experience replay for continual learning. *Advances in Neural Information Processing Systems*, 2024.

[2] 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*, 2020.

[3] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New Insights on Reducing Abrupt Representation Change in Online Continual Learning. In *International Conference on Learning Representations*, 2022.

[4] Noel CF Codella, David Gutman, M Emre Celebi, Brian Helba, Michael A Marchetti, Stephen W Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, et al. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In *IEEE International Symposium on Biomedical Imaging*, 2018.

[5] Emanuele Frascaroli, Aniello Panariello, Pietro Buzzega, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Clip with generative latent replay: a strong baseline for incremental learning. In *Proceedings of the British Machine Vision Conference*, 2024.

[6] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Introducing eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, 2018.

[7] 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*, 2019.

[8] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, 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. *IEEE International Conference on Computer Vision*, 2021.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.

[10] 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. *International Conference on Learning Representations*, 2021.

[11] Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations*, 2022.

[12] 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*, 2017.

[13] G. Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-normalizing neural networks. *Advances in Neural Information Processing Systems*, 2017.

[14] Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *International Conference on Learning Representations*, 2024.

[15] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2013.

[16] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[17] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[18] Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2024.

[19] Tian Yu Liu and Stefano Soatto. Tangent model composition for ensembling and continual fine-tuning. In *IEEE International Conference on Computer Vision*, 2023.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[21] 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*, 2018.

[22] Gianluca Mancusi, Mattia Bernardi, Aniello Panariello, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Is multiple object tracking a matter of specialization? In *Advances in Neural Information Processing Systems*, 2024.

[23] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, 1989.

[24] Martin Menabue, Emanuele Frascaroli, Matteo Boschini, Enver Sangineto, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Semantic residual prompts for continual learning. In *Proceedings of the European Conference on Computer Vision*, 2024.

[25] Monica Millunzi, Lorenzo Bonicelli, Angelo Porrello, Jacopo Credi, Petter N Kolm, and Simone Calderara. May the forgetting be with you: Alternate replay for learning with noisy labels. In *Proceedings of the British Machine Vision Conference*, 2024.

[26] Matteo Mosconi, Andriy Sorokin, Aniello Panariello, Angelo Porrello, Jacopo Bonato, Marco Cotogni, Luigi Sabetta, Simone Calderara, and Rita Cucchiara. Mask and compress: Efficient skeleton-based action recognition in continual learning. In *International Conference on Pattern Recognition*, 2024.

[27] Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Advances in Neural Information Processing Systems*, 2023.

[28] Angelo Porrello, Lorenzo Bonicelli, Pietro Buzzega, Monica Millunzi, Simone Calderara, and Rita Cucchiara. A second-order perspective on model compositionality and incremental learning. *International Conference on Learning Representations*, 2025.

[29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

[30] 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*, 2017.

[31] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[32] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2023.

[33] Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 2022.

[34] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[35] Runqi Wang, Xiaoyue Duan, Guoliang Kang, Jianzhuang Liu, Shaohui Lin, Songcen Xu, Jinhu Lü, and Baochang Zhang. Attriclip: A non-incremental learner for incremental knowledge learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2023.

[36] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European Conference on Computer Vision*, 2022.

[37] 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 conference on Computer Vision and Pattern Recognition*, 2022.

[38] Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022.

[39] Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2024.

[40] Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2024.

[41] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *IEEE International Conference on Computer Vision*, 2023.

[42] 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 *IEEE International Conference on Computer Vision*, 2023.

[43] Zangwei Zheng, Mingyu Ma, Kai Wang, Ziheng Qin, Xiangyu Yue, and Yang You. Preventing zero-shot transfer degradation in continual learning of vision-language models. *IEEE International Conference on Computer Vision*, 2023.

[44] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2022.

[45] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 2022.