CLIP with Generative Latent Replay: a Strong Baseline for Incremental Learning

Emanuele Frascaroli
emanuele.frascaroli@unimore.it
Aniello Panariello
aniello.panariello@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

With the emergence of Transformers and Vision-Language Models (VLMs) such as CLIP, fine-tuning large pre-trained models has recently become a prevalent strategy in Continual Learning. This has led to the development of numerous prompting strategies to adapt transformer-based models without incurring catastrophic forgetting. However, these strategies often compromise the original zero-shot capabilities of the pre-trained CLIP model and struggle to adapt to domains that significantly deviate from the pre-training data. In this work, we propose **Continual Generative training for Incremental prompt-Learning**, a simple and novel approach to mitigate forgetting while adapting CLIP. Briefly, we employ Variational Autoencoders (VAEs) to learn class-conditioned distributions within the embedding space of the visual encoder. We then exploit these distributions to sample new synthetic visual embeddings and train the corresponding class-specific textual prompts during subsequent tasks. Through extensive experiments on different domains, we show that such a generative replay approach can adapt to new tasks while improving zero-shot capabilities, evaluated using a novel metric tailored for CL scenarios. Notably, further analysis reveals that our approach can bridge the gap with joint prompt tuning. The codebase is available at https://github.com/aimagelab/mammoth.

1 Introduction

In real-world applications, data is rarely presented all at once; instead, it typically arrives incrementally and in a sequential order. The primary challenge in developing a neural network capable of incremental learning is the *catastrophic forgetting* [29] phenomenon; it describes

the tendency of these models to replace previously acquired knowledge with knowledge from new data, making them less proficient in tasks they have previously encountered.

Recently, Continual Learning has been influenced by the advent of Vision Transformers (ViTs) [10] and Large Vision-Language Models (VLMs) such as CLIP [28, 35]. In particular, several CL approaches [20, 39, 44, 45] take inspiration from parameter-efficient fine-tuning (PEFT) techniques [13, 46] and frequently employ prompt learning [19, 51, 52], where the model is adapted using a few learnable vectors termed *soft prompts*. For instance, CoOp [52] has been a significant source of inspiration: it learns a context prompt, which is concatenated to the textual name of the class and fed to the CLIP text encoder. Several extensions to CoOp have been proposed, such as CoCoOp [51], which injects information extracted from the visual encoder into the textual prompts.

In addition to their rich features, pre-trained Vision-Language models like CLIP provide strong zero-shot capabilities. This allows them to achieve remarkable continual learning performance without any fine-tuning [40], thereby avoiding forgetting by design. However, for tasks that deviate from CLIP's pre-training (*e.g.*, satellite and medical domains), adaptation is essential and must be considered. It is noted that incremental fine-tuning of CLIP models presents non-trivial challenges due to their scale and complexity (*e.g.*, number of parameters, image-text alignment, careful tuning of hyper-parameters). Due to catastrophic forgetting, incremental fine-tuning may lead to performance that is even worse than that of the frozen zero-shot model. Moreover, it has been shown [50] that fine-tuning CLIP can hinder its original zero-shot capabilities across other tasks and domains. Finally, only a few approaches [43, 48] offer an adaptation strategy compatible with open-vocabulary classification tests. In contrast, most of them [31] rely on a new classification head to model the posterior of new classes, limiting their applicability to the closed-vocabulary setting.

In this work, we propose a novel and simple approach that addresses these shortcomings during the incremental fine-tuning of CLIP. Inspired by CoOp [52], we freeze both the visual and text encoders and learn class-specific prompts to feed into the text encoder. This approach allows the model to maintain sufficient plasticity to adapt to new domains while remaining stable enough to preserve its original zero-shot capabilities. Moreover, as tasks progress and the model learns new classes, we use the corresponding learned prompts for the already observed classes. For the new ones, instead, we leverage hand-crafted prompts (*e.g.*, `"a photo of a <CLS>"`), resulting in a hybrid prompting approach.

Since prompt learning alone is not enough to overcome the challenges of a CL scenario (see Sec. 5), we bridge the gap with joint training through **generative replay**. Unlike standard rehearsal methods [5, 6, 36], we do not rely on a buffer of real examples but employ multiple lightweight generative models to learn the underlying data distribution. This approach offers two significant advantages: *i)* leveraging potentially infinite (synthetic) samples rather than relying solely on a subset of the dataset, and *ii)* ensuring data anonymity, thereby meeting privacy constraints.

Furthermore, our approach distinguishes itself from existing generative replay methods [12, 38], which generally focus on generating images in the input space. Instead, we model the data distribution in the latent space, thereby mitigating the *curse of dimensionality*. Specifically, for each new class, we train a lightweight Variational Autoencoder (VAE) [23] to model the distribution of the CLIP visual embeddings. By operating in the lower-dimensional latent space, we significantly reduce the complexity of our generative models, enabling their training to be completed in just a few minutes on standard GPUs. As discussed in Sec. 6, we evaluate our approach by comparing various generative and prompt-learning techniques, providing comprehensive validation of our choices.

We evaluate our proposed methodology, called **Continual Generative training for Incremental prompt-Learning** (CGIL), on various standard *class-incremental* CL benchmarks, showing state-of-the-art performance even on domains where zero-shot CLIP fails. Indeed, it overcomes the previous best performer by a wide margin (**+11%** on average). Inspired by [48, 50], we devise an additional metric to assess the zero-shot capabilities on future tasks. We evaluate in this setting all competitors with zero-shot capabilities (*i.e.*, the ones relying on a VLM), showing the superiority of our prompting strategy. We remark on the following contributions:

- We propose CGIL, a simple yet effective approach for incremental fine-tuning of CLIP models. It combines prompt-learning techniques with latent generative replay.
- We introduce a new metric to assess zero-shot performance on future tasks.
- Through extensive experiments, we demonstrate the validity of our approach, achieving state-of-the-art performance in widely adopted class-incremental benchmarks.

2 Related works

Continual Learning (CL) methods are designed to tackle the issue of *catastrophic forgetting* [29], which prevents the continuous transfer of previously acquired knowledge when data comes as a stream. Traditional approaches can be broadly classified into three main categories: *i) regularization techniques* [24, 26] prevent the most important parameters from drifting too far from the optimum; *ii) architectural-based methods* allocate specific sets of parameters for each incremental task [28, 37]; *iii)* rehearsal-based methods adopt a small memory buffer to store past exemplars that are used for later replay [2, 3, 5, 6, 32, 34, 36]. At the cost of bending the rules of continual learning, the latter models have been established as the state of the art when continuously training from scratch.

Recently, the advent of (large) pre-trained models based on the Vision Transformer (ViT) architecture [10, 55] has changed this paradigm [4], fostering the emergence of buffer-free alternatives [30, 39, 44, 45, 49] that achieve minimal forgetting without compromising privacy. These approaches are designed for *class-incremental* learning [41], whose goal is continuously expanding the set of recognizable classes with each incoming task. While class-incremental learning is typically regarded as the most challenging scenario for CL [1, 11, 41], its standard evaluation overlooks the potential loss of zero-shot capabilities in Vision-Language models like CLIP. In standard offline settings where all data is available at once, recent works have shown that it is possible to fine-tune CLIP models while maintaining – or even improving – their zero-shot capabilities for both a single task [7, 21, 47] or multiple tasks [48, 50]. Similarly to the latter, we aim to encourage the incremental fine-tuning of CLIP in an incremental scenario.

3 Preliminaries

Contrastive Language-Image Pre-Training (CLIP). CLIP [35] consists of a visual encoder $E_{vis}(\cdot)$ (which can be either a ViT or a CNN) and a text encoder $E_{txt}(\cdot)$ (typically a transformer). They are trained with a contrastive objective on image-text pairs to obtain aligned latent embeddings. Once trained, CLIP can be used for **zero-shot classification**. To do so, an image x is fed to the visual encoder to compute the embedding $z_{vis} = E_{vis}(x)$.

In parallel, for each candidate class, a text prompt is created by embedding the class label into a template like "`a photo of a <CLS>`". The resulting class-level prompts are tokenized and fed to the text encoder, producing a textual representation z^i_{txt} for each class y^i. The posterior probabilities are computed as the cosine similarity (noted as $\langle\cdot,\cdot\rangle$) between visual and class-level textual representations:

$$p(y^i|x) = \frac{exp(\langle z^i_{txt}, z_{vis}\rangle/\tau)}{\sum_{j=1}^{|\mathcal{Y}|} exp(\langle z^j_{txt}, z_{vis}\rangle/\tau)}, \tag{1}$$

where τ is a temperature parameter and $\mathcal{Y}$ represents the set of classes.

Prompt-learning for the CLIP model. Prompt learning techniques allow the efficient fine-tuning of large pre-trained models. Among these methods, CoOp [52] stands out as particularly effective. In a nutshell, CoOp does not rely on hand-crafted prompts to generate the input for the CLIP text encoder but rather on learnable context vectors V. These context vectors are concatenated with the label token $[CLS]$ and learned through gradient descent, using the similarity scores from Eq. (1) as logits of the cross-entropy loss function.

4 Method

Problem setting. In class-incremental CL, a deep model $f(\cdot;\theta)$ parametrized by θ is presented with a sequence of tasks $\mathcal{T}_i$ with $i := \{1,\dots,T\}$, where T denotes the number of tasks. The t-th task provides N_t examples that form the dataset $\mathcal{D}_t := \{x^{(n)}, y^{(n)}\}_{n=1}^{N_t}$ with label $y^{(n)} \in \mathcal{Y}_t$. Importantly, each task relies on a set of classes disjoint from others such that $\mathcal{Y}_i \cap \mathcal{Y}_j = \emptyset$ if $i \neq j$. The objective of CL is to minimize the empirical risk on all tasks:

$$\mathcal{L}_{\text{CL}} = \sum_{i=1}^{T} \mathbb{E}_{(x,y)\sim\mathcal{T}_i} \left[\mathcal{L}(f(x;\theta), y)\right], \tag{2}$$

where $\mathcal{L}$ is the loss function (*e.g.*, the cross entropy for classification). Since the model observes one task at a time, only the examples of the current task are available during training, making it unfeasible to directly optimize Eq. (2). Therefore, tailored strategies are required to prevent catastrophic forgetting.

4.1 CGIL: generative replay meets prompt learning

Fig. 1 depicts our approach termed **Continual Generative training for Incremental prompt-Learning** (**CGIL**). From a high-level perspective, CGIL comprises two main phases.

- **Phase 1 (*generative modelling*).** Using all images from the current task, we extract the corresponding image embeddings through the CLIP visual encoder. These embeddings are then grouped by their respective classes and used to train multiple independent Variational Autoencoders (VAEs), with one VAE dedicated to each class.
- **Phase 2 (*prompt alignment*).** We learn the context vectors for the text encoder (as in CoOp). However, instead of computing image embeddings from real images, we sample synthetic embeddings from all VAEs, encompassing both past and current classes.

The two phases are repeated at each task to refine previously learned prompts with knowledge from subsequent tasks. We refer the reader to Algorithm 1 for a procedural overview.

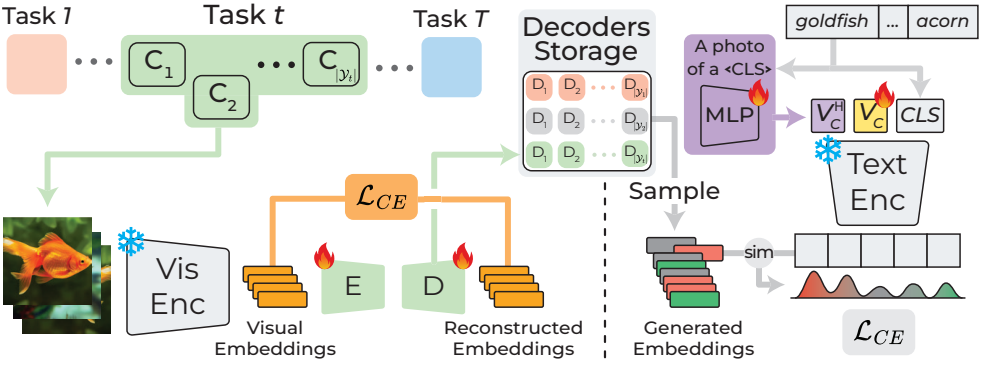

Figure 1: Training of the generative models (**left**) and prompt alignment (**right**). For each class C_i of task t, we train a class-specific generative model (*i.e.*, a VAE). Afterwards, only the decoders are retained for later generative replay, while the encoders can be discarded. In the second phase, we perform prompt alignment by matching the features sampled from all stored decoders up to task t with the text features generated using the learnable prompts.

Generative modelling. Our initial goal is to learn, for each class $y \in \mathcal{Y}_t$ of the current task t, the distribution within the CLIP latent space. We do so by extracting all the visual features $z_{vis} = E_{vis}(x)$ for each image $x \in \mathcal{D}_t$ from the current task. Subsequently, we employ the standard Evidence Lower Bound (ELBO) objective to independently train one VAE for each class, resulting in the training of $|\mathcal{Y}_t|$ encoders and decoders. The VAE encoder and decoder are lightweight, consisting of only three fully connected layers interleaved with LeakyReLU activations. Once the VAEs are trained, we discard the encoders and retain the decoders in a memory buffer. Indeed, the decoders will be used to sample new data points from the respective priors during the subsequent alignment phases.

Prompts alignment. In the second phase, we build a synthetic dataset by collecting visual embeddings sampled from all the stored decoders and perform prompt tuning by aligning the synthetic visual embeddings with the corresponding class-level text embeddings. These text embeddings are generated by the text encoder, which is provided with learnable prompts. In detail, the prompt construction involves generating two distinct tokens: *i)* a learnable class-specific token V_c to capture fine-grained details, and *ii)* a hyper-token V_c^H that models cross-domain knowledge. The hyper-token is generated by a shared, learnable Multi-Layer Perceptron (MLP), fed with the textual embedding z_{txt} obtained through the standard hand-crafted prompt. To sum up, the prompt $\mathbf{t}_c$ for the class c fed to the text encoder E_{txt} is:

$$\mathbf{t}_c = [V_c^H]\ [V_c]\ [CLS], \tag{3}$$

$$\text{where} \quad V_c^H = \text{MLP}(E_{txt}(\texttt{"a photo of a <CLS>"})). \tag{4}$$

The posterior probability of the class c is obtained as in Eq. (1), using the text embeddings obtained with our prompts $z_{txt}^c = E_{txt}(t_c)$. The training of each V_c and the MLP is performed via gradient descent, leveraging synthetic data from all previously encountered tasks. It is worth noting that the training process is really fast: because the visual embeddings are sampled from the VAEs rather than through the CLIP visual encoder, we can avoid the costly forward passes through the deep visual encoder.

Algorithm 1 Incremental learning of CLIP with CGIL

Require: datasets $D_t,\ t \in 1,\dots,T$

$\mathcal{M} \leftarrow \{\}$ ▷ initialize a memory buffer for storing the VAE decoders
for each dataset $D_t,\ t \in 1,\dots,T$ **do**
  # *Learning class-specific generative models in latent space*
  for each class $y \in \mathcal{Y}_t$ **do**
    $\mathcal{D}_t^{(y)} := \{x_i \mid (x^{(n)}, y^{(n)}) \in \mathcal{D}_t,\ y_i = y\}$ ▷ filter examples with label y
    $\mathcal{V}_t^{(y)} := \{E_{vis}(x_i) \mid x_i \in \mathcal{D}_t^{(y)}\}$ ▷ compute visual embeddings with CLIP
    Instantiate a new VAE with an encoder $E_y(\cdot)$ and a decoder $D_y(\cdot)$
    Train $E_y(\cdot)$ and $D_y(\cdot)$ on $\mathcal{V}_t^{(y)}$ ▷ minimize the ELBO
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{D_y(\cdot)\}$ ▷ store the decoder into the memory buffer
  end for
  # *Prompt alignment through generative replay*
  $\mathcal{S} = \{(\hat{z}_{vis}, c) \mid \hat{z}_{vis} \sim D_c(\cdot), \forall\, c \in \mathcal{Y}_1,\dots,\mathcal{Y}_t\}$ ▷ build the synthetic joint dataset
  for *it* $:= 1,\dots$ **do**
    Construct prompts $\mathbf{t}_c\ \forall\, c \in \mathcal{Y}_1,\dots,\mathcal{Y}_t$ ▷ see Eq. (3)
    $\mathcal{L}_{\text{GR}} \leftarrow$ sample a batch of pairs from $\mathcal{S}$ and use Eq. (1)
    $\mathbf{t}_c \leftarrow \mathbf{t}_c - lr \cdot \nabla_{\mathbf{t}_c} \mathcal{L}_{\text{GR}}$ ▷ apply Gradient Descent
  end for
end for

Thus, previously learned contexts are further fine-tuned, incorporating knowledge from subsequent tasks without incurring forgetting. During inference, we feed the image through the visual encoder and compute the posterior probability for each class.

Zero-shot inference. We adopt a **hybrid** approach to deal with both seen and unseen classes. For classes the model has encountered in previous tasks (*seen*), we employ the corresponding learned prompts. For classes it has not yet encountered, we feed the text encoder with the original hand-crafted prompts (*e.g.*, `"a photo of a <CLS>"`). Such a straightforward approach allows us to further preserve the zero-shot capabilities of CLIP while adapting to novel classes that arrive sequentially.

5 Experiments

Datasets. We evaluate our approach across a wide range of datasets with different levels of similarity *w.r.t.* the ImageNet pre-train [9, 33]. In particular, we test on:

- *Split Imagenet-R* [16], is a general-knowledge dataset frequently adopted in recent CL benchmarks [39, 44, 45, 49], with 200 classes split across 10 tasks.
- *Split Cars-196* [25] and *Split CUB-200* [42], are fine-grained datasets regarding car models and bird species, respectively. Both scenarios involve 10 subsequent tasks.
- *Split EuroSAT* [14, 15], which features RGB satellite images and defines a land cover classification problem consisting of 5 binary tasks.
- *Split ISIC* [8], including images with 6 skin diseases equally split into 3 tasks.

Model	Img-R	Cars-196	CUB-200	EuroSAT	ISIC	*Avg.*
Zero-shot CLIP [35]	81.95	64.99	50.52	53.32	26.59	55.47
LwF † [26]	19.09	23.24	16.73	25.13	33.06	23.45
GDumb † [34]	44.28	28.74	61.34	90.99	61.64	57.40
DER++ † [5]	56.66	53.66	74.62	93.08	65.68	68.74
L2P [45]	66.49	38.18	62.21	46.34	47.13	52.07
DualPrompt [44]	68.50	40.14	66.00	71.39	49.99	59.20
CODA-Prompt [39]	75.45	31.99	67.30	63.12	44.87	56.55
AttriCLIP [43]	87.39	75.63	58.28	72.33	28.26	64.38
SLCA † [49]	77.00	67.73	**84.71**	88.69	59.19	75.46
ZSCL [50]	89.14	77.66	62.43	79.13	34.14	68.50
MoE Adapters [48]	**90.67**	77.76	64.98	80.56	34.52	69.70
CGIL	89.42	**89.27**	83.12	**96.17**	**73.03**	**86.20**

Table 1: The Final Avg. Accuracy on the tested benchmarks. † denotes methods that fine-tune the whole model, while other methods apply parameter-efficient techniques.

Metrics. We adopt the more challenging evaluation setting of class-incremental learning [41] (CIL), where the task to which the data belongs is unknown during inference. To assess performance in this setting, we employ the average accuracy of each task computed at the end of the last training phase. This metric is referred to as **Final Average Accuracy** (FAA) or *Last Accuracy*. Additionally, we assess zero-shot performance on future (unseen) tasks by adapting the **Transfer** metric from [48, 50], originally introduced to evaluate zero-shot capabilities across different domains. Specifically, let A_t^i be the CIL accuracy on the i-th task after being trained until task t, the **Class Incremental Transfer** is defined as:

$$\text{CI-Transfer} = \frac{1}{T-1}\sum_{t=1}^{T-1}\left(\frac{1}{T-t}\sum_{i=t+1}^{T} A_t^i\right). \tag{5}$$

Implementation details. We train each VAE for 500 epochs, employing the Adam optimizer [22] with a learning rate of 0.0002. The hidden and latent sizes are 512 and 256, respectively. The synthetic embeddings, approximately $15K$ per class, are shuffled and divided into batches of 128. During the prompt-learning phase, we use Adam with a learning rate of 0.03. We employ CLIP with the ViT-L/14 backbone for each model. Finally, all results are averaged across 3 different seeds, impacting the composition of tasks. The standard deviations are reported in the Supplementary Material, along with additional details.

Comparison methods. We benchmark our model against several state-of-the-art prompt-tuning methods, including L2P [45], DualPrompt [44], CODA-Prompt [39], AttriCLIP [43], and ZSCL [50]. Additionally, we assess models that fine-tune the entire architecture, namely LwF [26], GDumb [34], DER++ [5], and SLCA [49]. In addition to such methods, we integrate MoE Adapters [48] into our framework, a parameter-efficient approach designed to prevent zero-shot accuracy degradation across datasets. To ensure a fair comparison, we train all competing models, tuning their hyperparameters for optimal performance. We include the performance of zero-shot frozen CLIP as a baseline to assess the efficacy of prompt tuning methods. Additionally, AttriCLIP, ZSCL, Moe Adapters, and our CGIL are also evaluated on future tasks, measuring how their zero-shot capabilities are affected by incremental training.

CI-Transfer	Img-R	Cars-196	CUB-200	EuroSAT	ISIC	*Avg.*
Zero-shot CLIP [35]	82.14	66.16	50.86	55.00	22.42	55.32
AttriCLIP [43]	85.75	73.98	54.07	59.69	24.14	59.53
ZSCL [50]	85.30	72.49	62.76	69.74	25.31	63.12
MoE Adapters [48]	**88.25**	75.82	61.73	55.77	21.06	60.53
CGIL	86.71	**78.80**	**66.34**	**71.52**	**48.18**	**70.31**

Table 2: The Class Incremental Transfer on the tested benchmarks. Only methods with zero-shot capabilities (*i.e.*, with CLIP as a backbone) could be tested.

5.1 Comparison with the State of the Art

Tab. 1 reports the CIL performance for all evaluated competitors and benchmarks. The last column shows the average accuracy of each method across all benchmarks. Despite the impressive results of zero-shot CLIP on Split Imagenet-R and Split Cars-196, it fails in other domains, particularly in the medical field. Consequently, competitors that rely on CLIP are heavily affected by this limitation and exhibit a similar drop. On the other hand, CGIL successfully addresses CLIP-related issues, delivering top-tier performance in all scenarios. Considering average performance, our method achieves a substantial lead (**+11**) over the best competitor, namely SLCA [49], while other prompt-based techniques fall behind.

Zero-shot performance. Tab. 2 displays the average CIL accuracies on unseen tasks, as measured by the CI-Transfer metric (Eq. (5)). Such a metric targets zero-shot capabilities; thus, only those approaches featuring CLIP or similar VLM models are evaluated. Nevertheless, similar trends emerge, with CLIP and other competitors excelling in Split Imagenet-R and Split Cars-196 but struggling with other datasets. CGIL exhibits remarkable ability in leveraging both the zero-shot expertise of CLIP and its knowledge from previously learned tasks, achieving superior performance in nearly all benchmarks.

6 Model analysis

To better validate the effectiveness of CGIL and its architectural design, we report additional experiments in Tab. 3.

Detailed comparison with CoOp . We evaluate vanilla CoOp under two distinct benchmarks: one trained jointly, *i.e.* without partitioning the dataset into tasks (*Joint*), and the other trained in the conventional CIL scenario (*Fine-tune*). For the latter, we make two minor adjustments to accommodate the incremental scenario: *i*) we allocate a class-specific learnable prompt to each class, rather than relying on a single global prompt, and *ii)* during subsequent tasks, the previously learned prompts are kept frozen. This strategy, also employed in [39], helps prevent forgetting: conversely, training all contexts in subsequent tasks could overwrite previous knowledge by altering learned prompts.

The insights derived from the results of these two approaches – see first rows of Tab. 3 – highlight the proficiency of CGIL in bridging the gap between fine-tuning and joint training when leveraging prompt learning. Indeed, our method matches the performance of the *joint* approach. As other ablation studies indicate, this success can be primarily ascribed to the effectiveness of our generative replay strategy.

	Img-R	Cars-196	CUB-200	EuroSAT	ISIC	*Avg.*
CoOp (*Joint*)	89.24	89.89	82.52	96.25	73.57	86.29
CoOp (*Fine-tune*)	84.94	68.61	59.84	79.27	37.08	65.95
CGIL	**89.42**	**89.27**	**83.12**	**96.17**	**73.03**	**86.20**
Different generative approaches						
Multinomial Gaussian	83.89	82.59	80.06	85.70	51.89	76.83
Mixture of Gaussians	88.54	88.82	82.10	93.04	62.42	82.98
Diffusion Models	89.28	90.14	83.48	95.73	68.99	85.52
VAEs (CGIL)	89.42	89.27	83.12	96.17	73.03	86.20
Different techniques to create the context prompt						
Class-specific token	89.09	88.96	83.06	95.59	72.21	85.78
MLP-generated token	89.41	88.91	82.82	95.69	70.79	85.52
Multiple shared tokens	89.02	88.08	81.50	95.47	70.18	84.85
CGIL	89.42	89.27	83.12	96.17	73.03	86.20

Table 3: Ablative studies on CGIL. Results are expressed as Final Average Accuracy.

Different generative models. Along with Variational Autoencoders, we evaluate various families of generative models to determine which best complements our method. The most straightforward approach involves fitting a multivariate Gaussian distribution for each class [49]. As indicated in Tab. 3, this approach alone achieves state-of-the-art results (it achieves an average of 76.83, compared to 75.46 for SLCA). However, exploiting more powerful generative models like VAEe considerably improves the effectiveness of the alignment procedure. This suggests that the quality and variety of the generated data are crucial.

We also evaluate Mixture of Gaussians (MoGs) models, which combine multiple Gaussian components, allowing for greater flexibility in representing the variability within each class. Moreover, we investigate the application of Denoising Diffusion Probabilistic Models (DDPMs) [17], both saturating the required performance in a generation. We train the DDPMs with the same hyper-parameters as VAEs, described in Sec. 5. Among the two, we stick to VAEs due to their faster training and reduced number of parameters *w.r.t.* DDPMs.

Different prompting techniques. We recall that our context consists of a class-specific token and a generated token (see Eq. (3)). Hence, at the bottom of Tab. 3, we present the results with different choices. Specifically, we evaluate: *i*) using a single class-specific context, as in *CoOp (Fine-tune)*; *ii*) utilizing only the hyper-token generated by the MLP; and *iii*) adopting a method similar to the original CoOp, where multiple tokens are learned and shared across classes. For the first two ablative variants, we increase the number of contextual tokens in our preliminary experiments. However, while the third variant, which uses a shared context across classes, benefited from this modification, the first two strategies showed no improvement.. Therefore, we report only the results with a single context token.

The results of these alternatives fall shortly behind CGIL, indicating that the main contributors are the generative rehearsal and the alignment phase. This becomes evident when considering the gap in performance between CoOp *(Fine-tune)* and *Class-specific Context*: They share the same prompting mechanism, but the latter is enhanced with generative replay.

6.1 Discussion and limitations

On the memory and computational costs. Our VAEs decoders are relatively lightweight in terms of memory storage, requiring only half a million parameters each. Nevertheless, CGIL may face limitations as the number of classes increases, which could restrict its applicability in certain scenarios. Compared with a standard rehearsal approach, the memory requirement of one decoder is comparable to a buffer of 14 RGB images of size 224×224 (*i.e.*, ~2MB). However, we note that these decoders are only needed for the training phase, while during inference, only the CLIP visual encoder and the embeddings z^c_{txt} of our prompts are required. Thus, the computational cost for inference is equivalent to a single pass through the visual encoder plus a matrix multiplication to obtain the similarity scores. This represents a significant advancement over other CL-prompting methods. Indeed, L2P, DualPrompt, and CODA-Prompt execute the forward pass on the image twice, while AttriCLIP computes both visual and textual embeddings during test time.

When considering the computational costs of training, learning the generative models in latent space allows our VAEs to remain notably lightweight, potentially eliminating the need for a GPU. Instead, the alignment phase presents a higher level of complexity, as it involves backpropagating gradients through the CLIP text encoder. Nonetheless, the duration of this phase can be controlled by adjusting the size of the synthetic dataset generated.

Online CL setting. We highlight that CGIL may be classified under the category of online CL methods [1, 27], as training images are fed only once to the visual encoder. Although this requires temporarily storing the visual embeddings of all samples from the current task, the memory burden is negligible due to the low memory footprint required for storing the latent embeddings. These are employed to train our generative models and subsequently discarded.

7 Conclusions

We introduce a novel framework, **Continual Generative training for Incremental prompt-Learning** (**CGIL**), that allows the incremental fine-tuning of CLIP models. Variational Autoencoders are employed to learn the latent distributions of input images, enabling the generation of synthetic latent embeddings. Such data is exploited in subsequent tasks to fine-tune CLIP through prompt learning. Our approach significantly outperforms state-of-the-art CL methods when tested across a broad spectrum of benchmarks and domains. By introducing a new metric, the **Class Incremental Transfer**, we evaluate zero-shot performance on future tasks during training, demonstrating that our framework is the most effective at leveraging past knowledge to predict unseen classes. Further analysis validates our architectural choices and shows that CGIL bridges the gap with the performance of prompts learned jointly.

Acknowledgements

We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support. This paper has been supported from Italian Ministerial grant PRIN 2020 “LEGO.AI: LEarning the Geometry of knOwledge in AI systems”, n. 2020TA3K9N. Additionally, the research activities of Angelo Porrello have been partially supported by the Department of Engineering “Enzo Ferrari” through the program FAR_2023_DIP – CUP E93C23000280005.

References

[1] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient Based Sample Selection for Online Continual Learning. In *Advances in Neural Information Processing Systems*, 2019.

[2] Giovanni Bellitto, Federica Proietto Salanitri, Matteo Pennisi, Matteo Boschini, Angelo Porrello, Simone Calderara, Simone Palazzo, and Concetto Spampinato. Selective attention-based modulation for continual learning. *arXiv preprint arXiv:2403.20086*, 2024.

[3] Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[4] Matteo Boschini, Lorenzo Bonicelli, Angelo Porrello, Giovanni Bellitto, Matteo Pennisi, Simone Palazzo, Concetto Spampinato, and Simone Calderara. Transfer without forgetting. In *Proceedings of the European Conference on Computer Vision*, 2022.

[5] 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.

[6] 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.

[7] Sanjoy Chowdhury, Sayan Nag, and Dinesh Manocha. APoLLo : Unified adapter and prompt learning for vision language models. In *Empirical Methods in Natural Language Processing*, 2023.

[8] 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.

[9] Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[11] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. In *arXiv preprint arXiv: 1805.09733*, 2018.

[12] Rui Gao and Weiwei Liu. Ddgr: Continual learning with deep diffusion-based generative replay. In *International Conference on Machine Learning*, 2023.

[13] Zeyu Han, Chao Gao, Jinyang Liu, Sai Qian Zhang, et al. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*, 2024.

[14] 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.

[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*, 2019.

[16] 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.

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

[18] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, 2021.

[19] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *Proceedings of the European Conference on Computer Vision*, 2022.

[20] Dahuin Jung, Dongyoon Han, Jihwan Bang, and Hwanjun Song. Generating instance-level prompts for rehearsal-free continual learning. In *IEEE International Conference on Computer Vision*, 2023.

[21] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2023.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[23] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014.

[24] 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.

[25] 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.

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

[27] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems*, 2017.

[28] 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.

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

[30] Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. *Advances in Neural Information Processing Systems*, 2024.

[31] 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.

[32] 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.

[33] Jaehoon Oh, Sungnyun Kim, Namgyu Ho, Jin-Hwa Kim, Hwanjun Song, and Se-Young Yun. Understanding cross-domain few-shot learning based on domain similarity and few-shot difficulty. *Advances in Neural Information Processing Systems*, 2022.

[34] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Proceedings of the European Conference on Computer Vision*, 2020.

[35] 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.

[36] 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.

[37] 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.

[38] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in Neural Information Processing Systems*, 2017.

[39] 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.

[40] Vishal Thengane, Salman Khan, Munawar Hayat, and Fahad Khan. Clip model is an efficient continual learner. *arXiv preprint arXiv:2210.03114*, 2022.

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

[42] 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.

[43] 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.

[44] 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.

[45] 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.

[46] Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arXiv preprint arXiv:2312.12148*, 2023.

[47] Hantao Yao, Rui Zhang, and Changsheng Xu. Visual-language prompt tuning with knowledge-guided context optimization. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2023.

[48] 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.

[49] 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.

[50] 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.

[51] 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.

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

A Implementation Details

We provide further details of our experimental setup as follows.

Image size. For all our benchmarks, we rescale input images (RGB) to a resolution of 224×224.

Data augmentation. For methods based on CLIP as their backbone, we employ the standard CLIP preprocessing, which solely involves RGB normalization. For all other methods, the training phase incorporates random cropping and horizontal flipping.

Reproducibility. We conduct each experiment thrice, using the fixed seeds of 1992, 1996, and 1997. Each seed determines a unique class order for each dataset, thus influencing how data are partitioned into tasks.

Split Cars-196. In this benchmark, data is split into 9 tasks of 20 classes each, and a final task with the remaining 16 classes.

Split ISIC. From the original dataset [8], we removed the most frequent class "*Melanocytic nevus*".

B Standard Deviations

The standard deviations for our primary experiments are presented in Tab. D, which correspond to the results in Tab. 1. It's important to note that the order of classes, and consequently the composition of tasks, can vary due to different seeds. This variation can lead to significant discrepancies in the results of some methods, highlighting their sensitivity to this factor with high variances.

Model	Img-R	Cars-196	CUB-200	EuroSAT	ISIC
LwF † [26]	±5.72	±1.88	±4.16	±2.78	±1.98
GDumb † [34]	±0.51	±0.47	±0.46	±1.49	±3.64
DER++ † [5]	±0.97	±1.51	±0.73	±1.62	±2.16
L2P [45]	±0.40	±2.33	±1.92	±7.86	±3.84
DualPrompt [44]	±0.52	±2.36	±0.57	±4.94	±1.07
CODA-Prompt [39]	±0.56	±3.39	±3.19	±6.30	±3.50
AttriCLIP [43]	±0.41	±0.06	±1.21	±2.09	±1.07
SLCA † [49]	±0.33	±0.85	±0.40	±0.48	±3.83
MoE Adapters [48]	±0.15	±1.02	±0.29	±0.53	±8.25
CGIL	±0.12	±0.14	±0.10	±0.10	±1.75

Table D: The standard deviations on the tested benchmarks (results in Tab. 1). † denotes methods that fine-tune the whole model, while other methods apply parameter-efficient techniques.