

# LVP-CLIP: Revisiting CLIP for Continual Learning with Label Vector Pool

Yue Ma, Huantao Ren, Boyu Wang, Jingang Jin, Senem Velipasalar, Qinru Qiu  
Syracuse University

{yma183, hren11, bwang30, jjin24, svelipas, qiqiu}@syr.edu

## Abstract

Continual learning aims to update a model so that it can sequentially learn new tasks without forgetting previously acquired knowledge. Recent continual learning approaches often leverage the vision-language model CLIP for its high-dimensional feature space and cross-modality feature matching. Traditional CLIP-based classification methods identify the most similar text label for a test image by comparing their embeddings. However, these methods are sensitive to the quality of text phrases and less effective for classes lacking meaningful text labels. In this work, we rethink CLIP-based continual learning and introduce the concept of Label Vector Pool (LVP). LVP replaces text labels with training images as similarity references, eliminating the need for ideal text descriptions. We present three variations of LVP and evaluate their performance on class- and domain-incremental learning tasks. Leveraging CLIP's high dimensional feature space, LVP learning algorithms are task-order invariant. The new knowledge does not modify the old knowledge, hence, there is minimum forgetting. Different tasks can be learned independently and in parallel with low computational and memory demands. Experimental results show that proposed LVP-based methods outperform the current state-of-the-art baseline by a significant margin of 40.7%.

## 1. Introduction

Deep neural networks trained by using supervised learning have achieved remarkable accuracy in classification tasks. Their effectiveness relies on the assumption that the training data distribution fully and accurately represents the testing data. However, in real-world scenarios, data samples are often not available all at once. New classes of knowledge are discovered sequentially and corresponding training data arrives in stages. Continual learning addresses this by incrementally training a model to effectively learn new tasks without catastrophic forgetting [21] of previously ac-

This project was partially funded by AFRL's FA8750-21-C-1511 contract and NSF IUCRC ASIC Center (CNS-1822165).

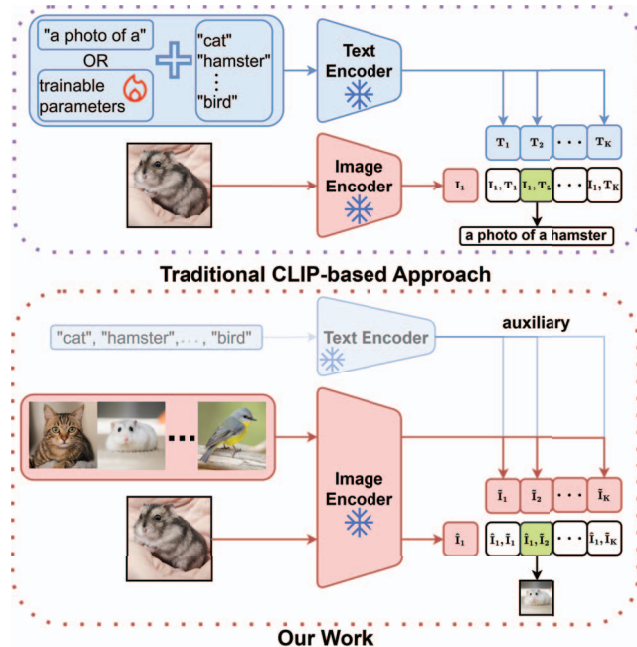


Figure 1. Comparison with traditional CLIP-based approaches. While traditional methods compare similarity between the encoded test image and text labels, our approach evaluates similarity between image embeddings directly and makes the text encoder play an auxiliary role when possible.

quired knowledge. Recent continual learning approaches frequently utilize the vision-language model CLIP (Contrastive Language-Image Pretraining) [23] due to its high-dimensional feature space and ability to match features across different modalities.

CLIP bridges the gap between language and vision through contrastive learning, enabling zero-shot classification by matching a list of text embeddings to the test image embedding. The primary challenge of CLIP lies in the vast search space for the optimal text, since different text inputs can yield highly different results. Recent research works have focused on improving the text embedding quality. For example, PointClip [41] leverages human knowledge and experience to design text descriptions for point cloud data. PointClipV2 [46] uses language models, like GPT-3 [4], to

generate enhanced text. Methods like CoOp [45] and Co-CoOp [44] further improve performance by incorporating trainable parameters into the text encoder. Yet, all these approaches still classify the input image by searching for the best matching text embedding.

L2P [36] first introduces the prompt-pool on pre-trained ViT [8] for continual learning, leveraging trainable prompts inside a pool to retain knowledge from different tasks. Following works [28, 30, 33, 35] improve L2P by either re-designing the prompts and prompt pools or refining the matching procedure between the prompts and embeddings. A challenge for prompt-pool-based methods is the uncertainty in the matching procedure between the test embedding and the prompt key. Although various optimization objectives are designed to improve matching, there is no guarantee that the optimal prompt will be selected for a given test image. If an incorrect prompt key is chosen, leading to the selection of wrong prompts, the result is more likely to be inaccurate, since the prompt may introduce biased information instead of providing helpful knowledge. Furthermore, as the same set of prompts is updated across different tasks, forgetting is unavoidable.

In this work, we revisit image classification and continual learning by introducing a novel concept, referred to as the Label Vector Pool (LVP). As shown in Fig. 1, instead of searching for the optimal text label phrases and relying only on their embeddings for similarity comparison, we utilize the image embeddings generated from training images as references. The proposed approach, utilizing LVP, enables CLIP-based incremental learning models to reduce if not remove their dependency on the quality of label phrases. LVP allows CLIP to adapt to a wide range of datasets, particularly those with classes difficult to describe in text or class names that have no semantic meaning, such as “ZIL103” or “ZSU234”, etc. Furthermore, feature vectors from the same modality tend to cluster more closely than those from different modalities, leading to improved classification accuracy.

Since we use CLIP as the foundation model and leverage its embedding capabilities, we refer to our method as LVP-CLIP, which is a completely different approach compared to prompt-pool-based methods. Instead of constructing a prompt pool that provides additional features for the classifier, we consolidate the knowledge of each class into a single vector and store it directly in the pool. This single vector can be formed from training image embeddings or as a combination of image and text embeddings. Given a test image, we simply calculate the similarity between its embedding and each vector in LVP, then select the class with the highest similarity.

Most class-incremental and domain-incremental learning algorithms assume a known upper bound on class count, as they rely on MLP-based classifiers with fixed output dimensions. In contrast, our proposed LVP-CLIP imposes

no restriction on the number of classes. Its memory cost grows linearly at a very low rate as the number of classes increases. In fact, we demonstrate its scalability by applying it to a cross-dataset, cross-domain incremental learning task, Cross-Task Incremental Learning (CTIL), which includes 595 classes, as discussed in detail in Sec. 5. LVP-CLIP avoids performance degradation as the number of tasks increases, since learning new information does not modify previously stored knowledge. In other words, it treats incremental learning and batch learning equivalently. As a similarity-based approach, it simply searches for the embedding vector in the feature space that is most similar to the input vector. The high dimensionality of its feature space provides substantial memory capacity. The main contributions of this work include the following:

- We propose the concept of Label Vector Pool (LVP) by revisiting the classification procedure of CLIP. As a similarity based approach like CLIP, the LVP uses image embedding or a mixture of image and text embeddings as references, hence is more flexible and less biased to any one modality compared to CLIP.
- We present three variations of LVP, namely LVP-CLIP-I, LVP-CLIP-IT and LVP-CLIP-C. We evaluate our methods in class-incremental, domain-incremental and cross-task incremental settings and outperform SOTA methods.
- The proposed LVP-based incremental learning has orders of magnitude lower computational complexity in learning and 2x lower inference complexity compared to baselines.
- The performance of LVP-based learning is scalable to large number of tasks. We demonstrate that LVP has outstanding memory capacity and learning capability using an experimental setting that consists of 4 commonly used incremental learning datasets with 595 classes.

## 2. Related Work

**Conventional solutions for continual learning.** Existing continual learning algorithms can be classified into three categories, namely regularization-based, architecture-based, and rehearsal-based methods. Regularization-based methods [1, 11, 12, 17, 39, 40] set constraints on the trainable parameters by limiting the learning rate of the important parameters for old tasks. Although these methods do not require additional memory for replay buffers and additional model parameters, they have limited performance on complex datasets. Architecture-based methods [15, 20, 26, 34, 38, 42, 43] create separate parameters for each task to bypass catastrophic forgetting. Rehearsal-based methods [2, 3, 5, 6, 25, 37] maintain a buffer of data from past tasks. During the optimization for new tasks, the model is also trained on buffered data from previous tasks to mitigate forgetting. The performance of these methods is limited by the buffer size, and as buffer size decreases

the performance declines sharply. Additionally, these methods are challenging to apply when data is private or cannot be stored [27]. Our LVP-CLIP addresses continual learning challenge by extracting class knowledge directly from CLIP without relying on a rehearsal buffer.

**Prompt-based continual learning.** Almost all recent works that achieved noteworthy performance in continual learning use a prompt-based architecture. After L2P [36] first proposed visual prompting, which constructs a prompt pool for continual learning tasks, prompt-pool-based methods have become the main track for its great performance. DualPrompt [35] improves the pool design with task-invariant prompts (G-Prompt) for general knowledge and task-specific prompts (E-Prompt) for expert knowledge. S-Prompts [33] designs domain-specific prompts and utilizes a K-NN operation as a domain identifier for inference. AttriClip [32] adapts the prompt pool for CLIP by maintaining a prompt pool for the text encoder. The prompt-pool-based methods may suffer from the mismatching challenge, since there is a potential to select the wrong prompt during inference. In addition, prompts add additional computation complexity in training and inference. These challenges do not apply to LVP. All of these recent continual learning works rely on the pre-trained models, such as ViT [8] and CLIP [23]. Hence, we adopt the same foundational model for feature extraction and embedding.

### 3. Preliminaries

For clarity, we use  $\sim$  and  $\wedge$  to differentiate the symbols for training and testing sets, respectively, in the rest of the paper. We use the superscripts to denote the index of classes or tasks and the subscripts to denote the index of training/testing instances.

#### 3.1. Continual Learning

We consider a sequence of tasks  $S = \{S^1, \dots, S^M\}$ , where  $M$  is the total number of tasks. Each task is a set,  $S^t = \{(x_1^t, y_1^t), \dots, (x_{n^t}^t, y_{n^t}^t)\}$ ,  $t \in [1, \dots, M]$ , where  $(x_i^t, y_i^t)$ ,  $i \in [1, \dots, n^t]$ , is a pair containing the input  $x_i^t \in X$  and its corresponding label  $y_i^t \in Y$  and  $n^t$  is the total numbers of samples. The goal of continual learning is to train a model  $f(\theta) : X \rightarrow Y$  continuously over time on a set of tasks, arriving sequentially, such that it learns the new tasks without forgetting the old tasks.

Task-, Domain- and Class-Incremental Learning (TIL, DIL, CIL) are the three main scenarios of continual learning. Domain-incremental learning assumes that the number and labels of classes remain consistent across tasks, and the only difference among tasks is the distribution of the input data. Both Task- and Class-incremental learning address the scenario, where each task introduces a distinct set of new classes to be learned. Task-incremental learning assumes a known task identity at inference time, whereas the

class-incremental learning does not make such assumption.

### 3.2. CLIP

CLIP is a vision-language model trained on text-image pairs. It consists of a text encoder  $g_{\theta_t}(\cdot)$  and an image encoder  $f_{\theta_t}(\cdot)$ . Given a sentence  $txt$  and an image  $img \in \mathbb{R}^{H \times W \times C}$ , the text and image embeddings are  $g_{\theta_t}(txt) = T$  and  $f_{\theta_t}(img) = I$  respectively, where  $T, I \in \mathbb{R}^D$  with  $D$  denoting the dimension of the embeddings. Given a dataset containing  $K$  classes, the  $txt$  is a phrase like “a photo of a  $[y]$ ”, where  $y \in [1, \dots, K]$  is the index of the label, and  $[y]$  denotes the class name of the label. The probability of labeling the test image  $img$  with the class  $y$  is computed as:

$$p(y|img) = \frac{\exp(\langle T^y, I \rangle)}{\sum_{k=1}^K \exp(\langle T^k, I \rangle)}, \quad (1)$$

where  $\langle \cdot, \cdot \rangle$  denotes the similarity function. Three commonly used similarity functions are L1 norm, L2 norm and cosine similarity, denoted by  $\langle \cdot, \cdot \rangle_{L1}$ ,  $\langle \cdot, \cdot \rangle_{L2}$ ,  $\langle \cdot, \cdot \rangle_{Cos}$ , respectively. To classify an image, the class with the highest probability is chosen, i.e.

$$\hat{y} = \underset{k}{\operatorname{argmax}} p(k|img), \quad k \in [1, \dots, K]. \quad (2)$$

## 4. Methodology

In this section, we first introduce a superset method, LVP, by rethinking how CLIP can be used for classification. Then, we design three continual learning methods utilizing LVP, namely LVP-I, LVP-IT and LVP-C. Finally, we describe the loss functions to train these methods.

### 4.1. Label Vector Pool(LVP)

As discussed in Sec. 3.2, CLIP classifies an input image by comparing the distance between the image embedding and a list of text embeddings that represent class labels. The query vector with unknown-ID is compared with a group of key vectors with known IDs, and the query vector is assigned the ID of the key vector with the highest similarity. Following this concept, we define label/labeled vector  $L \in \mathbb{R}^D$  as a vector with known ID, such as the class or domain name, or task ID, etc. A LVP for class  $k$  is a set of label vectors with ID  $k$ , denoted as  $L^k = \{L_1^k, L_2^k, \dots, L_{P^k}^k\}$ , where  $L_i^k \in \mathbb{R}^D$ ,  $i \in [1, \dots, P^k]$ ,  $k \in [1, \dots, K]$ , and  $P$  is the pool size. The similarity between an image embedding  $I$  and  $L^k$  is computed as the maximum similarity between  $I$  and all instances in  $L^k$ :

$$\langle L^k, I \rangle = \max(\langle L_1^k, I \rangle, \langle L_2^k, I \rangle, \dots, \langle L_{P^k}^k, I \rangle). \quad (3)$$

The probability that a testing image  $img$  belongs to class  $y$  can be computed as:

$$p(y|img) = \frac{\exp(\langle L^y, I \rangle)}{\sum_{k=1}^K \exp(\langle L^k, I \rangle)}. \quad (4)$$

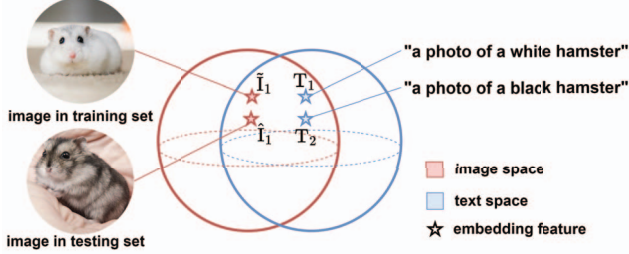


Figure 2. The hypothesis is that embeddings in the same modality should be more similar to each other. The training image embedding  $\tilde{I}_1$  is expected to be more similar to the test image embedding  $\hat{I}_1$  than the text embeddings  $T_1$  and  $T_2$ , i.e.,  $\langle \tilde{I}_1, \hat{I}_1 \rangle > \langle \tilde{I}_1, T_1 \rangle, \langle \tilde{I}_1, \hat{I}_1 \rangle > \langle \tilde{I}_1, T_2 \rangle$ .

As can be seen, Eq. (1) is a special case of Eq. (4), where the LVP contains only one labeled vector, which is the text embedding of the class name, i.e.,  $L^k = \{T^k\}$ , and  $P^k = 1$ . We denote the computational complexity for classifying one image as  $O$ , and represent it using the number of times the similarity function is calculated, i.e.,  $O = \sum_{k=1}^K P^k$ . If all classes have the same pool size  $P$ , then the complexity simplifies to  $O = \sum_{k=1}^K P^k = P \times K$ .

Our LVP-based approach is a general framework for similarity-based classification and extends CLIP in two ways: (1) the test image is no longer restricted to comparison with text embeddings, it can be compared with any labeled vectors with matching dimensions; (2) each class can be represented by one or multiple labeled vectors, which may be obtained from different modalities.

## 4.2. Motivation for Using Image LVP

The training of CLIP ensures that matching images and text phrases will be mapped to nearby locations in the feature space. Intuitively, we expect that embeddings of inputs from the same modality (e.g., image-to-image or text-to-text) will be closer to each other than those from different modalities, as illustrated in Fig. 2. As discussed in Sec. 4.1, the task of classification is to identify the labeled vector most similar to the query. This raises an interesting question: can we use image embeddings by themselves in the LVP instead of text embeddings?

We verified this idea on CIFAR100 [14] dataset by using the embeddings of the training images as the LVP. The experimental results are shown in Tab. 1. When 30% of the training data is used as the LVP, the classification accuracy already surpasses that achieved by using the text labels as LVP. With the entire set of training data used as the LVP, testing accuracy improves by almost 5% compared to using text labels. Since we used CLIP as the foundation model and leveraged its powerful embedding capability, we refer to our approach as LVP-CLIP. The drawback of LVP-CLIP is that the computational and memory complexity increases

	10%	30%	50%	70%	100%	text
P	50	150	250	350	500	1
O	5000	15000	25000	35000	50000	100
Acc.	71.0	74.8	76.1	76.9	78.2	73.3

Table 1. Testing accuracy using embeddings of the training images (columns 2-6) and label text (column 7) as the LVP on CIFAR100. The percentage values in the first row represent the proportion of the training set included in the LVP. The second (P) and third (O) rows show the corresponding LVP size and the computational complexity of testing a single image.

as the pool size  $P$  grows, which will be addressed next.

## 4.3. Designing More Efficient LVP

To address the computational and memory complexity while maintaining performance, we propose three different designs using LVP: (i) **LVP-CLIP-I** uses only image embeddings in the LVP, and reduces the pool size  $P$  to 1, so that its inference complexity is the same as the zero-shot learning while providing better accuracy; (ii) **LVP-CLIP-IT** extends the LVP by combining text embeddings with image embeddings; (iii) Unlike LVP-CLIP-I and LVP-CLIP-IT, **LVP-CLIP-C** does not use the similarity function to compare the test image with LVP. It trains a classifier using information stored in LVP and applies it for classification.

The overall framework for the three variations of LVP-CLIP is shown in Fig. 3. The traditional CLIP-based classifier compares the encoded test image with the text-embedding of the class label. This can be considered as a special case of LVP, where the label vector is the text embedding. Therefore, we also refer to it as LVP-CLIP-T.

**Reducing LVP Size (LVP-CLIP-I).** As noted in Sec. 3, using the whole training set as the LVP is effective, yet expensive in terms of computational and memory requirements. If we can use only one vector in the LVP, which one would be the best representative of the whole training set for each class? Each image embedding  $I^k$  is a set of image features,  $I^k = \{E_1^k, E_2^k, \dots, E_D^k\}$ , where  $E^k \in \mathbb{R}$ . We plot the distributions of those features in the same class and compare such distributions across different example classes in Fig. 4. In the figure, the superscript of the features indicates class index, and subscripts represent the feature index. As can be seen, within a class, each feature roughly follows a Gaussian distribution. Across classes, features have different combinations of means. Therefore, we use their *mean* value as the representative label vector for each class,

$$L^k = \{\tilde{I}^k\}, \text{ and } \tilde{I}^k = \frac{1}{\tilde{N}^k} \sum_{i=1}^{\tilde{N}^k} \tilde{I}_i^k, \quad (5)$$

where  $\tilde{I}_i^k \in \mathbb{R}^D$  is a training image embedding of class  $k \in [1, K]$  and the total number of training images from

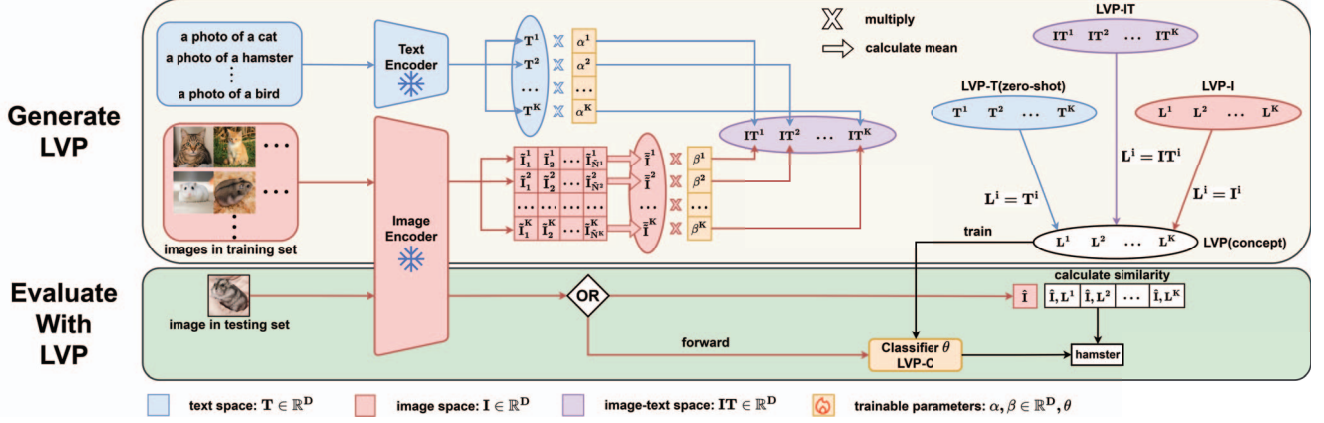


Figure 3. Framework of LVP-CLIP. Firstly, the concept of LVP is demonstrated. Secondly, three realizations of LVP is shown as LVP-T, LVP-I and LVP-IT. LVP-T known as zero-shot is LVP generated form the text encoder. Our proposed LVP-I is the mean of image embeddings of each class in the training set. LVP-IT can be obtained as a combination of LVP-T and LVP-I with the task-specific trainable paremeters  $\alpha, \beta$  of each class. In addition, LVP-C is a classifier optimized on LVP.

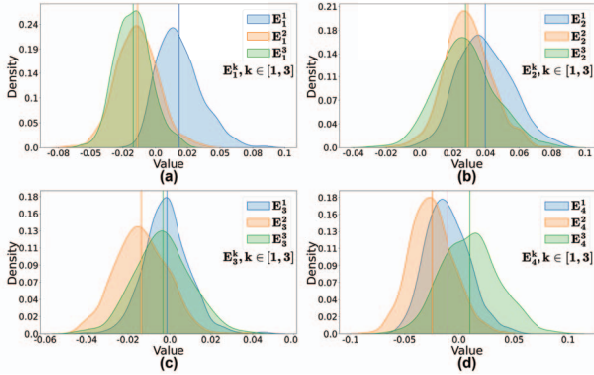


Figure 4. Distributions of the same feature across different classes in CIFAR100 training set. We examine the first 4 feature distributions for 3 classes, denoted as  $E_i^k, i \in [1, 4], k \in [1, 3]$ . Panels (a) to (d) show the distributions of  $E_1^k$  to  $E_4^k$  for these three classes. The vertical lines represent the mean values of each distribution. As shown, all features approximately follow a Gaussian distribution, with different combinations of means across different classes.

	text	100%	LVP-I	LVP-IT	LVP-C
P	1	500	1	1	1
O	100	50000	100	100	-
Acc.	73.3	78.2	80.1	82.0	81.0

Table 2. Testing accuracy of different LVP-CLIP methods on CIFAR100. (Cosine similarity is used here.)

class  $k$  is  $\tilde{N}^k$ . As shown in Tab. 2, by using the average embedding of the whole training data in each class as the LVP, we not only reduce the inference complexity but also improve the performance by about 2% with LVP-CLIP-I.

**Enrich Image Embedding Representation with Text (LVP-CLIP-IT).** The success of CLIP indicates that text

embeddings and image embeddings follow similar distributions and have very high correspondence. In cases, where the distribution of training images is not similar to that of the testing images, information from another modality, i.e., text, may serve as an unbiased prompt. Therefore, with LVP-CLIP-IT, we design the LVP as a weighted combination of the text embedding and the average embedding  $\tilde{I}^k$  of the training images as follows:

$$L^k = \{IT^k\}, \text{ and } IT^k = \alpha^k \times T^k + \beta^k \times \tilde{I}^k, \quad (6)$$

where  $\alpha^k, \beta^k \in \mathbb{R}^D$ , are trainable vectors to balance the text embedding and image embedding, respectively. Experimental results in Tab. 2 show that combining image embedding and text embedding improves the performance by another 2% over LVP-I.

#### 4.4. LVP with a Classifier (LVP-CLIP-C)

With LVP-CLIP-I and LVP-CLIP-IT, the embedding vectors belonging to difference classes are stored, and classification is performed by comparing the distance between the embedding of a test image against each stored labeled vector using a similarity function. This process treats all feature dimensions as equally important. To handle cases, where some features should have higher importance than others in the classification process, we build a simple linear classifier that maps an embedding vector to a class prediction,  $f_\theta(\cdot) : \mathbb{R}^D \rightarrow [0, 1]^K$  and train it using data only from the LVP. For a test image  $\hat{I}$ , the prediction is:

$$\text{class} = \text{argmax}(f_\theta(\hat{I})). \quad (7)$$

#### 4.5. Optimization Objective

Among the three variations of LVP, LVP-CLIP-I has no trainable parameters. It solely relies on the averaging of the

embeddings of training images. LVP-CLIP-IT and LVP-CLIP-C, on the other hand, have a small set of parameters that are trainable.

For LVP-CLIP-IT, we set  $\alpha, \beta$  as task-specific parameters to avoid forgetting, and use the entropy loss to train them. Given task  $t \in [1, M]$  and class  $k^t \in [1, K^t]$ , where  $M$  and  $K^t$  are the total number of tasks and the number of classes in task  $t$ , respectively, the loss function used to train  $\alpha$  and  $\beta$  is:

$$\mathcal{L} = \mathbb{E}\left[-\log \frac{\exp(\langle IT^y, I \rangle)}{\sum_{k^t=1}^{K^t} \exp(\langle IT^{k^t}, I \rangle)}\right], t \in [1, \dots, M]. \quad (8)$$

For each new task, a new set of  $\alpha$  and  $\beta$  will be optimized.

The parameter  $\theta$  in LVP-CLIP-C is shared among all tasks. Since it is trained using labeled vectors stored in LVP, every time a new task is added, it will be trained again with the current LVP. The process is similar to experience replay. In this way, it also does not suffer from the forgetting. Again, entropy loss is used for the training. Given the LVP  $L^k, k \in [1, K]$  of each class, the loss function can be written as:

$$\mathcal{L} = \mathbb{E}\left[-\log \frac{\exp(L^y)}{\sum_{k=1}^K \exp(L^k)}\right]. \quad (9)$$

#### 4.6. Discussion on Complexity and Performance

For all three LVP variants, the size of the LVP grows as the new tasks are learned, albeit at a very slow rate. For each class, we only need one or a few labeled vectors based on the difficulty and distribution of each dataset. Taking ImageNet100 as an example, each labeled vector  $L \in \mathbb{R}^D$  is of size  $D = 768$ , which is only 0.5% of the size of an image ( $3 \times 224 \times 224$ ). Even for 100 classes, the overall size of LVP is 76800, equivalent to the size of 0.5 images. The same amount of memory is required to store the text embeddings for classes, if CLIP is used to classify input images.

Furthermore, LVP-CLIP is task-order invariant, since each LVP is generated independently. As a result, its performance is not affected by task order. Additionally, since new label vectors do not modify the existing ones, LVP-CLIP exhibits minimal forgetting. The independence of the label vectors allows for parallel learning of different classes and making it simple to merge multiple classes—a distinct advantage over existing approaches. Finally, the LVP pool is not a rehearsal buffer, since it does not store raw images, making it less vulnerable to user privacy leakage.

### 5. Experiments

We evaluate LVP-CLIP in three experiment settings: (i) class-incremental learning (CIL), (ii) domain-incremental learning (DIL), and (iii) cross-tasks incremental learning (CTIL). We compare LVP-CLIP with the SOTA methods

across various categories under commensurate experimental settings. Additionally, we perform extensive ablation studies to gain deeper insights into our approach.

**Implementation details.** We use frozen text and image encoders throughout the experiments. For the image encoder, ViT-L/14 [23] is used as the backbone in all experiments. While most CLIP-based works use cosine similarity to calculate the distance between two embeddings, we have found that L1 distance works better for image-image embeddings and is much simpler. Therefore, we use L1 similarity for LVP-CLIP-I, and cosine similarity for LVP-CLIP-IT. The parameter  $\alpha$  in LVP-CLIP-IT is initialized to 0.5 for all datasets while  $\beta$  is consistently initialized to 1. We use SGD as the optimizer to train  $\alpha$  and  $\beta$ , with a learning rate of 0.0001.

For LVP-CLIP-C, the classifier is trained using labeled vectors from LVP-IT, except for datasets Core50 where no semantically unique labels are provided, and thus, LVP-I vectors are used. The classifier is trained with the ADAM optimizer, using a learning rate of 0.01. Training is stopped when the loss reaches approximately 0.1-0.05. None of the three proposed variants of LVP-CLIP requires knowledge of the total number of classes in advance.

For the CIL experiments, we generate one label vector for each class, therefore,  $P^k = 1, k \in [1, \dots, K]$ . For the DIL experiments, we generate a label vector for each domain in a class. Thus  $P^k$  equals to the number of domains.

An *upper-bound* of classification accuracy is obtained for each experiment by assuming that all training data were available upfront and a classifier is trained based on the complete dataset by using features extracted by the CLIP image encoder.

We use *average testing accuracy* (higher is better) as our metric [19]. After all tasks are learned, the overall accuracy is calculated by averaging the accuracy of each task. In all the experiments except CTIL, the testing data for different tasks is of equal size, ensuring that the average test accuracy is not biased toward any specific task.

#### 5.1. Class Incremental Learning

We first evaluate our method on two popular 2D image datasets namely CIFAR100 [14] and ImageNet100 [7]. Following the setup in [32], we select 100 classes from the original ImageNet. Both CIFAR100 and ImageNet100 are divided into 10 tasks, with each task consisting of 10 classes.

We compare our methods with two rehearsal-based methods, iCaRL [24] and ARI [31], and three CLIP-based methods, CoOp [45], Continual-CLIP [29] and AttriCLIP [32]. For a fair comparison, all methods are implemented using ViT-L [8], except for iCaRL and ARI, which are implemented with ResNet [10].

In Tab. 3, the best and 2nd-best performances are shown in **bold** and with underline, respectively. For ImageNet100,

all variants of LVP-CLIP outperform other CLIP-based and experience-replay methods with significant margins. Specifically, LVP-CLIP-IT and LVP-CLIP-C provide 9.2% and 9.3% improvement, respectively, over the best performing CLIP-based method AttriCLIP. On CIFAR100, LVP-CLIP-IT has 0.6% higher accuracy than AttriCLIP.

Method	Buffer size	CIFAR100 [14]	ImageNet100 [7]
iCaRL [24]	20/class	49.5*	59.5*
ARI [31]	20/class	80.9*	79.3*
CoOp [45]	10/class	67.6*	79.3*
Cont.-CLIP [29]	0	66.7*	75.4*
AttriCLIP [32]	0	<u>81.4*</u>	83.3*
LVP-CLIP-I	0	80.2	91.8
LVP-CLIP-IT	0	<b>82.0</b>	<u>92.5</u>
LVP-CLIP-C	0	81.1	<b>92.6</b>
Upper-bound	-	86.5	96.0

Table 3. Testing accuracy on CIFAR100 and ImageNet100. Data with \* is obtained from [32].

## 5.2. Domain Incremental Learning

For these experiments, we use two popular public datasets, namely DomainNet [22] and CORE50 [18]. DomainNet includes objects from 345 classes. Each object is represented by images spanning six domains: clipart, infographic, painting, quickdraw, real-world and sketch. Each domain has its own dedicated training and testing datasets. Therefore, it has 6 training tasks and 6 testing tasks.

The CORE50 dataset, on the other hand, consists of 50 classes across 11 domains. Of these, 8 domains are used for training data, presented sequentially one at a time as 8 tasks, while 3 domains are reserved for testing. Importantly, the testing domains are not part of the training process, making CORE50 also suitable as a domain adaptation dataset.

The performance is measured as the average accuracy over all testing domains. For every class, we generate a label vector for each trained domain, resulting an LVP of size 6 for DomainNet and 8 for CORE50.

We compare our method with two regularization-based approaches, EWC [13] and LwF [16], a rehearsal-based method, ER [9], and two prompt-pool-based methods, L2P [36] and S-Prompts [33]. For fair comparison, all methods are implemented with ViT-L [8]. As seen in Tab. 4, for both DomainNet and CORE50 datasets, the variants of LVP-CLIP provide the top and second-best performances outperforming all the baselines.

## 5.3. Cross-Task Incremental Learning

We present the Cross-Task Incremental Learning (CTIL) experimental setting, which uses an ID-UNKNOWN multi-dataset for task-incremental learning. Here, a task refers to the unit for continual learning, encompassing both class-incremental and domain-incremental learning across dataset. Our motivation is to show how well a method

Method	Buffer size	DomainNet [22]	CORE50 [18]
EWC [13]	0	60.0	75.8
LwF [16]	0	61.4	77.6
ER [9]	50/class	64.3	79.5
L2P [36]	0	67.6	79.7
S-Prompts [33]	0	69.7	85.2
LVP-CLIP-I	0	<u>70.1</u>	<u>86.1</u>
LVP-CLIP-IT	0	<b>70.9</b>	-
LVP-CLIP-C	0	68.6	<b>89.6</b>
Upper-bound	-	75.9	99.0

Table 4. Testing accuracy on DomainNet and CORE50 datasets. There is not LVP-CLIP-IT result for CORE50 because the dataset does not have differentiable text labels for different classes.

can perform in a more realistic setting to continuously learn new tasks. CTIL presents a significant challenge in continual learning, as it requires a model to perform both CIL and DIL. We conduct experiments on a four-dataset CTIL benchmark, which combines CIFAR100, ImageNet100, DomainNet and Core50, resulting in a total of 595 (100+100+345+50) distinct object classes, divided into 34 (10+10+6+8) training tasks and 29 (10+10+6+3) testing tasks. The 34 training tasks are processed sequentially in a random order.

Since this setting is too challenging for most of the existing incremental learning frameworks, we chose only L2P [36] and DualPrompt [35] as the baseline methods for comparison. We also present the performance of our methods and the baselines on each individual dataset. In the second-to-last column of Tab. 5, we show the “Ideal” performance of each technique in the combined dataset. This is calculated as the weighted average of their accuracies on individual datasets, adjusted by the number of test samples in each. For example, the “Ideal” score for LVP-CLIP-I is calculated as  $(80.2*10+91.8*10+70.1*6+86.1*3)/(10+10+6+3) = 82.7$ . The *Difference* between the *Ideal* and the actual accuracy is shown in the last column. As a reference, the upper-bound of the classification accuracy is also provided, which is obtained by training a classifier using all the training data with a ViT-L backbone.

It is not surprising that the actual accuracy is consistently lower than the ideal accuracy. This is due to two main reasons: (i) as the number of classes grows, their separation in the feature space diminishes, making it harder to distinguish them; and (ii) as tasks are trained sequentially, earlier tasks are increasingly susceptible to forgetting. Since LVP-CLIP-I does not modify previously learned knowledge, it largely avoids forgetting. The 1.8% drop in accuracy is mainly due to more congested feature distributions. LVP-CLIP-C shows a slightly higher degradation (4.0%), likely because its simple linear classifier does not work well in a congested feature space. Overall, the LVP-CLIP-based approaches closely approximate the ideal performance, indicating that the object classes in these four datasets remain

Method	CIFAR100	ImageNet100	DomainNet	CORe50	CF100 + IN100 + DN + CR50	Ideal	Difference
Tasks train/test	10/10	10/10	6/6	8/3	34/29	-	-
L2P [36]	<b>88.3</b>	82.3	67.6	79.7	37.4	81.1	-43.7
DualPrompt [35]	<u>86.5</u>	85.4	<b>71.8</b>	84.3	40.3	82.9	-42.6
LVP-CLIP-I	80.2	91.8	70.1	<u>86.1</u>	<u>80.9</u>	82.7	-1.8
LVP-CLIP-IT	82.0	<u>92.5</u>	<u>70.9</u>	-	<b>81.0</b>	83.7	-2.7
LVP-CLIP-C	81.1	<b>92.6</b>	68.6	<b>89.6</b>	79.4	83.4	-4.0
Upper-bound	86.5	96.0	75.9	99.0	87.1	88.9	-1.8

Table 5. Testing accuracy on CIFAR100 + ImageNet100 + DomainNet + CORe50. **Bold** is the best and underline is the second best.

separable in the high-dimensional feature space. This also indicates that the significant 42% accuracy drop observed in L2P and DualPrompt is primarily attributed to forgetting.

Although prompt-pool-based approaches show slightly better performance on certain datasets (e.g., CIFAR100 and DomainNet), they greatly suffer from forgetting as the number of classes increases. Moreover, DualPrompt and L2P require roughly twice as much computation at inference compared to LVP-CLIP. In addition, LVP-CLIP-based learning only requires forward propagation, whereas L2P and DualPrompt rely on backpropagation, making LVP-CLIP far more efficient in terms of learning complexity.

## 5.4. Ablation Studies

Ablation studies are conducted to assess the effect of various design variables on performance. All studies were performed on CIFAR100.

**Text Prompt Quality.** The impact of different text prompts is shown in Tab. 6. Here, we use LVP-CLIP-T to represent the traditional CLIP-based classification, where the text (T) embedding serves as the label vector for similarity. Two slightly different text phrases are used for LVP-CLIP-IT and LVP-CLIP-T and their performances are compared. As can be seen, traditional CLIP is highly sensitive to text prompt quality, making it essential to optimize the prompt. However, the proposed LVP-CLIP-IT is less susceptible to the text quality, since it uses text as supplementary information alongside image features.

Text	“[cls]”	“a photo of a [cls]”
LVP-CLIP-T	65.9	73.3
LVP-CLIP-IT	<b>81.3</b>	<b>82.0</b>

Table 6. Ablation study on the impact of text prompt qualities.

**Initialization of  $\alpha$ .** The performance of LVP-CLIP-IT is sensitive to the initial value of  $\alpha$  as shown in Tab. 7. It is a good practice to set  $\alpha$  to 0.5 when text quality is good, and decrease it as text quality declines.

Initialization of $\alpha$	0.1	0.3	0.5	0.7	1.0
LVP-CLIP-IT	80.9	81.6	<b>82.0</b>	81.9	81.7

Table 7. Ablation study on the impact of  $\alpha$ 's initial value.

**Training dataset size.** The effect of training dataset size on LVP-CLIP is shown in Tab. 8. Since the labeled vector is the average embedding of the training images, a larger training set generally improves the label vector's approximation of the distribution mean, provided the data is randomly sampled. However, it appears that around 150 training images sufficient to obtain a relatively good estimation of the mean. As expected, with a small number of training images, the LVP-CLIP-IT significantly outperforms LVP-CLIP-I and LVP-CLIP-C, due to the additional information provided by the text embedding. For more experimental results and analysis, please refer to the Suppl. file.

# of training set	1%	3%	5%	10%	30%	50%	100%
train data/class	5	15	25	50	150	250	500
LVP-CLIP-I	66.2	74.9	77.3	78.8	80.0	79.9	<b>80.2</b>
LVP-CLIP-IT	74.2	79.3	80.3	81.0	81.7	81.8	<b>82.0</b>
LVP-CLIP-C	67.3	75.5	77.8	79.6	80.8	81.1	<b>81.1</b>

Table 8. Ablation result of the # of training images on CIFAR100.

## 6. Limitations

As our proposed LVP-CLIP does not introduce additional parameters into the feature extraction process, the quality of LVP depends on the performance of the pre-trained models. We plan to address this limitation in future work by developing methods or algorithms that optimize pre-trained models through our proposed LVP, maintaining their robustness while also adapting them to newly learned classes.

## 7. Conclusion

In this paper, we have introduced a novel concept, the Label Vector Pool (LVP), which enables incremental learning without forgetting by harnessing the powerful feature extraction and encoding capabilities of CLIP. LVP blurs the distinction between batch learning and incremental learning, since it does not modify the learned model to accommodate new knowledge. This approach offers a highly cost-effective solution for incremental learning. It provides several times the memory capacity of traditional methods and superb scalability to large dataset. It significantly outperforms the SOTA approach in cross-dataset mixed class- and domain-incremental learning settings with 595 classes.

## 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] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 2
- [3] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 2
- [4] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1
- [5] 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
- [6] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9516–9525, 2021. 2
- [7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6, 7
- [8] 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, 2021. 2, 3, 6, 7
- [9] Tyler L. Hayes, Nathan D. Cahill, and Christopher Kanan. Memory efficient experience replay for streaming learning. In *2019 International Conference on Robotics and Automation (ICRA)*, page 9769–9776. IEEE Press, 2019. 7
- [10] 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. 6
- [11] Zixuan Ke, Bing Liu, and Xingchang Huang. Continual learning of a mixed sequence of similar and dissimilar tasks. In *Advances in Neural Information Processing Systems*, pages 18493–18504. Curran Associates, Inc., 2020. 2
- [12] 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
- [13] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 7
- [14] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 4, 6, 7
- [15] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *Proceedings of the 36th International Conference on Machine Learning*, pages 3925–3934. PMLR, 2019. 2
- [16] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 7
- [17] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. 2
- [18] Vincenzo Lomonaco and Davide Maltoni. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on robot learning*, pages 17–26. PMLR, 2017. 7
- [19] 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
- [20] 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 (CVPR)*, 2018. 2
- [21] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem, 1989. 1
- [22] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019. 7
- [23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1, 3, 6
- [24] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 6, 7
- [25] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. 2
- [26] 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

- [27] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, page 1310–1321, New York, NY, USA, 2015. Association for Computing Machinery. 3
- [28] 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11909–11919, 2023. 2
- [29] Vishal Thengane, Salman Khan, Munawar Hayat, and Fahad Khan. Clip model is an efficient continual learner. *arXiv preprint arXiv:2210.03114*, 2022. 6, 7
- [30] Boyu Wang, Yue Ma, and Qinru Qiu. Prompt-based domain incremental learning with modular classification layer. In *ECAI 2024*. IOS Press, 2024. 2
- [31] Runqi Wang, Yuxiang Bao, Baochang Zhang, Jianzhuang Liu, Wentao Zhu, and Guodong Guo. Anti-retroactive interference for lifelong learning. In *European Conference on Computer Vision*, pages 163–178. Springer, 2022. 6, 7
- [32] Runqi Wang, Xiaoyue Duan, Guoliang Kang, Jianzhuang Liu, Shaohui Lin, Songcen Xu, Jinhu Lü, and Baochang Zhang. Attriclclip: A non-incremental learner for incremental knowledge learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3654–3663, 2023. 3, 6, 7
- [33] 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, 3, 7
- [34] Zifeng Wang, Tong Jian, Kaushik Chowdhury, Yanzhi Wang, Jennifer Dy, and Stratis Ioannidis. Learn-prune-share for lifelong learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 641–650, 2020. 2
- [35] 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: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, page 631–648, Berlin, Heidelberg, 2022. Springer-Verlag. 2, 3, 7, 8
- [36] 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, 3, 7, 8
- [37] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [38] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3014–3023, 2021. 2
- [39] Hongxu Yin, Pavlo Molchanov, Jose M. Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K. Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deep-inversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2
- [40] 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
- [41] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8552–8562, 2022. 1
- [42] Tingting Zhao, Zifeng Wang, Aria Masoomi, and Jennifer Dy. Deep bayesian unsupervised lifelong learning. *Neural Networks*, 149:95–106, 2022. 2
- [43] Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning, 2023. 2
- [44] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16816–16825, 2022. 2
- [45] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 2, 6, 7
- [46] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyu Guo, Ziyao Zeng, Zipeng Qin, Shanghang Zhang, and Peng Gao. Pointclip v2: Prompting clip and gpt for powerful 3d open-world learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2639–2650, 2023. 1