Class Incremental Learning for Image Classification with Out-of-distribution Task Identification

Xusheng Cao, Haori Lu, Xialei Liu, *Member, IEEE,* Ming-Ming Cheng, *Senior Member, IEEE,*

***Abstract*—Class Incremental Learning (CIL) for image classification aims to address real-world scenarios by allowing a model to learn new categories while retaining the knowledge of old categories. It is more challenging than Task Incremental Learning (TIL) as task ID is not provided during testing. Therefore, transitioning from CIL to TIL is an intuitive approach to handling CIL problems for image classification. Currently, the main challenge of this approach lies in improving the accuracy of task identification. To address this issue, we propose to use a large-scale image-text pre-training model (i.e. CLIP) as the backbone, training and saving different classifiers for different tasks. Each classifier not only includes the classes of the current task, but also an Out-of-distribution (OOD) class corresponding to the classes encountered in all previous tasks. At test time, we iterate through classifiers from the last task to find the correct task ID of the test image, and perform classification in a TIL way. In addition, to tackle the issue of early-stop termination in iterative prediction due to model bias toward later tasks, we propose using CLIP zero-shot ability to assist learned OOD detection. Experiments show that our method achieves state-of-the-art performance on the traditional many-shot and the more challenging few-shot settings of CIFAR-100 and ImageNet-Subset datasets.**

***Index Terms*—Class Incremental Learning, Image Classification, Out-of-distribution Detection, Few-shot Learning, Image-text pretraining.**

I. Introduction

THE purpose of Incremental learning is to adapt deep models to real-world tasks. When encountering new categories in image classification, deep models should possess the ability to learn new knowledge while also retaining the previously acquired knowledge. The main challenge in Incremental learning for image classification is addressing catastrophic forgetting [1], which refers to the phenomenon of the model drifting away from its prior knowledge when learning new tasks, significantly reducing performance on old tasks.

There are three main scenarios [2] for Incremental learning: domain incremental learning (DIL), task incremental learning (TIL), and class incremental learning (CIL). DIL focuses on learning the same set of classes with different domain distributions, such as the real-world/animated versions of the same classes. The model can then learn invariant information across domains and make predictions despite different distributions. TIL and CIL differ in whether the task ID is provided during testing. CIL corresponds to the setting when task ID is not provided during testing, which is a more challenging form of incremental learning. Therefore, our study primarily focuses on addressing this problem.

The authors are with VCIP, CS, Nankai University, Tianjin 300350, X. Liu is the corresponding author(xialei@nankai.edu.cn).

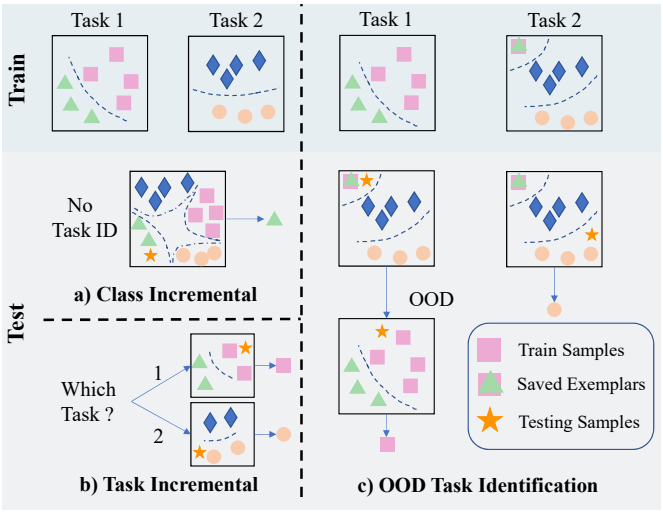


Fig. 1: a) and b). Illustration of CIL and TIL scenarios for image classification during both training and testing processes. c). Our methods. We identify the task ID through an OOD detection fashion, thus transitioning CIL to TIL, which is considerably easier.

Existing methods addressing CIL mainly fall into three categories. Replay-based methods [3]–[7] store a subset of previous data or generate some synthetic samples for replay and use it to train the model alongside new data to prevent forgetting. Regularization-based methods [8]–[12] apply constraints on the parameter space or activation space of the model during training to prevent catastrophic forgetting while learning new tasks. Architecture-based methods [13]–[18] modify the model's architecture to allow for adding more capacity, such as adding neurons, modules, or branches to improve performance on both new and old tasks.

An alternative intuitive idea to address CIL is to transition the learning objective from class incremental to task incremental by recognizing the task ID first. This involves using a specific method to determine the task ID of the test image and then testing in a task incremental way. OOD detection can be used as the method for identifying the task, as attempted in recent works. Kim et al. [19] decompose the continual learning problem into two sub-problems, within-task prediction (WP) and task-id prediction (TP), and use OOD detection to solve the TP problem. CLOM [20] trains each task as an OOD detection model which enables the model to not only classify the current task but also detect OOD samples that do not belong to the current task. MORE [21] is a multi-head

transformer-based model that trains each head to be a separate classifier for each task. ESN [22] proposes stage-isolated classifiers for incremental learning, selecting the best classifier using energy-based confidence scores. However, although their methods completed a transition from class incremental to task incremental, the performance of the OOD detection was still unsatisfactory, leading to low accuracy of the overall continual learning task.

Recently, Image-Text pre-training models have attracted a lot of research attention. Due to pre-training on a massive dataset of over 400M Image-Text pairs, CLIP [23] has excellent generalization ability and its image encoder has the potential ability to predict most real-world categories. In addition, CLIP has a very strong ability for zero-shot evaluation, reaching high performance on many downstream tasks. These characteristics of pre-trained models make them naturally strong for continual learning and OOD detection tasks. Fort et al. [24] show that pre-trained transformers can bring significant improvements on near-OOD benchmarks, where OOD samples have a similar distribution with ID samples. Ming et al. [25] add a separate softmax scaling to the similarity scores obtained by CLIP encoders, further improving the separability between ID and OOD data.

Hence, we present a novel methodology for CIL based on OOD task identification with the help of a pre-trained CLIP model. As illustrated in Fig. 1, during training, we fine-tune a classifier on the CLIP image encoder with $(N+1)$ heads (N heads for the first task) for task t $(t>1)$, where N is the number of classes in the current task (ID) and 1 represents all previously encountered classes (treated as the OOD class), and append it to a saved list. During testing, we iterate through the saved classifier lists from back to front to find the classifier that outputs an ID class for the test image. To further tackle the early-stop termination in the iterative testing process, we propose to use CLIP zero-shot to assist the OOD detection.

Our contributions can be summarized as:

- We apply Image-Text pre-training models to OOD task identification for CIL, leveraging CLIP's generalization to reduce model bias towards current tasks and mitigate forgetting.
- We propose using CLIP zero-shot to assist OOD detection, reducing the problem of early-stop during the OOD prediction iteration, and preventing a significant decrease in task recognition performance when the number of tasks increases.
- Our model has achieved state-of-the-art performance on the traditional many-shot and the more challenging few-shot settings of CIFAR-100 and ImageNet-Subset datasets.

II. Related work

A. Class incremental learning

CIL is a challenging setting in continual learning. Methods can be categorized into three types: regularization-based, replay-based and architecture-based methods [26].

Regularization-based methods add extra regularization terms between new and old tasks to ensure that old knowledge is kept while learning new tasks. EWC [9] penalizes the variation of each parameter based on their importance. SI [12] approximates parameters' importance by its contribution to the total loss variation. R-EWC [27] refines the implementation of the penalty and performs a factorized rotation of the parameter space. Another branch leverages knowledge distillation for continual learning, such as LWF [28], LwM [29] and EBLL [30]. However, as the number of tasks increases, mitigating forgetting through regularization terms becomes progressively challenging. Knowledge distillation is also popular in other applications, such as image segmentation [31], [32], and object detection [33], [34].

Replay-based methods estimate and recover old data distributions when training new tasks [35] by storing old knowledge in a memory buffer. Some methods simply retain a limited subset of past samples, with different selection strategies [6], [7]. iCaRL [6] and EEIL [3] perform KD on old and new training data, PODNet [36] introduces a spatial distillation loss and Co2L [4] performs a self-supervised distillation loss. Other methods store synthetic samples produced by generative models [37]–[39], or intermediate feature representation [10], [11] instead of real samples. We also employ a small memory to store previously classes, which together serve as an OOD class for training our OOD detector.

Architecture-based methods control which parameters belong to which tasks by training a mask or adding a new branch on the base model for each task. Piggyback [14], HAT [16], SupSup [17] and PackNet [40] adopt a fixed network architecture and optimize a binary mask to select specific parameters for each task. DyTox [41] incrementally trains task-specific adapter layers. DER [18] dynamically increases network branches for each task. Expert Gate [42] learns a modular expert network per task. PathNet [43] and RPSNet [44] build multiple parallel layer-wise network modules and select different paths for different tasks. In scenarios with long task sequences, this type of method results in excessively large models with a significant increase in the number of parameters, which makes it impractical and lacks applicability.

B. Few-shot Class Incremental learning

Unlike traditional CIL, Few-shot Class Incremental Learning(FSCIL) is dedicated to acquiring new knowledge with limited labeled data. Typically, the initial task (base session) is supported by abundant data, whereas in subsequent tasks (incremental sessions), each category is represented by merely 5-10 samples. The main challenge lies in preventing the model from over-fitting to new data. As a result, most approaches [45], [46] adopt a prototype-based method, training a feature extractor during the base session and subsequently either maintaining it unchanged or only updating a small portion of its modules. Although this strategy ensures the retention of old knowledge, it performs poorly on new tasks. Other approaches employ meta-learning techniques [47]–[49], aspiring for the model to learn how to discern patterns from a minimal number of samples. Recent approaches [50], [51] have intricately designed bi-level optimization techniques and synthetic feature generation methods to enhance the performance during

incremental sessions. Generally, continual learning methods designed for many-shot scenarios often struggle to achieve good results in few-shot setting. However, our OOD-based can be directly applied to this more challenging setting achieving SOTA results without any specialized modification.

C. OOD Detection

OOD detection detects test samples subject to distribution which is different from the training distribution. The approaches can be categorized into three groups. Classification-based, density-based and distance-based methods [52].

Classification-based methods find that OOD samples typically show lower maximum probabilities than ID samples, which can be used to conduct OOD detection [53]. ODIN [54] introduces temperature scaling and input perturbation to increase the separability between ID and OOD samples. G-ODIN [55] extends ODIN by adopting a specific training objective called DeConf-C.

Density-based methods model the probability density of data and regard samples with low density as OOD. [56] estimate Gaussian distributions of features and calculate the Mahalanobis distance for detecting OOD. Sometimes OOD data can also have high likelihood in the probability density model [57]. Therefore, some works propose to calculate other metrics to replace likelihood, such as likelihood ratio [58], likelihood regret [59] and SEM score [60].

Distance-based methods detect OOD by calculating the distance in feature space between test samples and prototypes of ID classes. [56] conduct OOD detection by calculating the minimum Mahalanobis distance between samples and all ID classes distributions. Without assuming that features follow a Gaussian distribution, [61] and [62] calculate the cosine distance between features and ID classes. [63] propose the deep nearest neighbor distance approach to directly detect OOD samples.

D. OOD applied to incremental learning

Recent works have started to apply OOD detection in continual learning. These methods mainly use OOD detection to determine whether a test sample belongs to the current task (ID) or other tasks (OOD). Kim et al. [19] decomposes the CIL problem into within-task prediction (WP) and task-id prediction (TP), adopting traditional OOD detection to solve TP problem. Due to the inferiority of traditional OOD detection methods, this approach is constrained by the low TP accuracy. CLOM [20] trains each task as an OOD detection model rather than a traditional classification model. While this approach can adapt TIL methods to CIL and achieve decent results, it falls short of achieving SOTA performance compared to the originally designed CIL approaches like DER [18]. MORE [21] builds a multi-head transformer-based model, n heads for n tasks, and an additional OOD head to classify previously encountered classes. We argue that separating the OOD detector and completely eliminating it during testing is redundant.

In addition to these OOD-based methods, ESN [22] also aggregates classifiers from previous stages or tasks, ensuring that old knowledge is retained while learning new tasks. However, unlike our approach, which detects whether a sample belongs to the current or a previous task using OOD detection, the energy-based method uses a temperature-controlled energy metric to normalize classifier outputs and ensure comparability across tasks.

III. Motivation

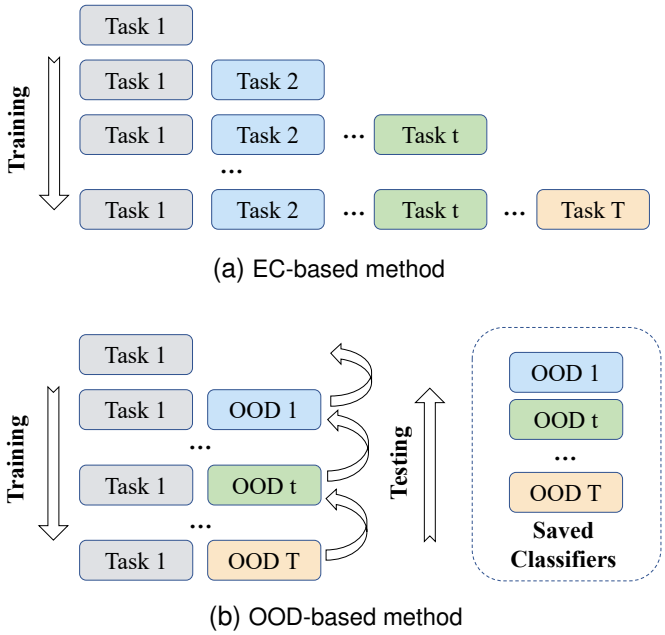


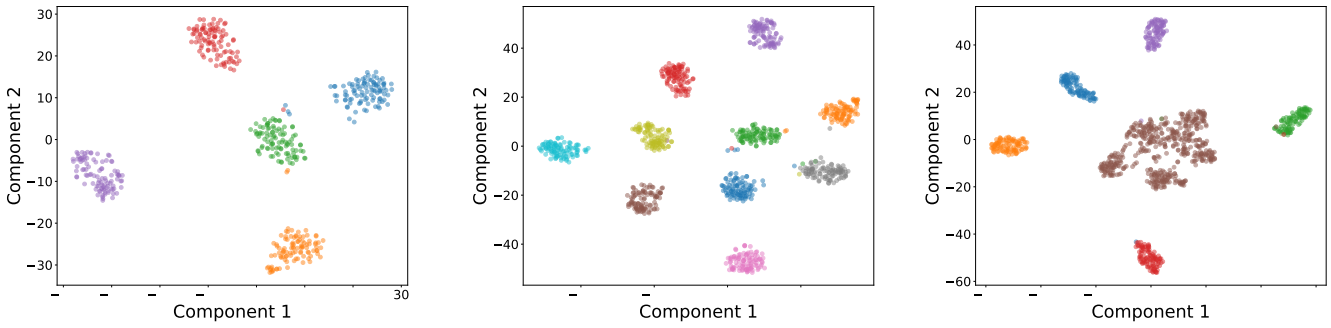
Fig. 2: Comparison of EC-based and our OOD-based continual learning methods. (a) The EC-based method adds new classification heads when every new task arrives. (b) The OOD-based method adds only one OOD head for all old classes. The detailed training and testing process of the OOD-based method can be found in following sections.

Here, we will first briefly introduce OOD based and Extended Classifier (EC) based continual learning methods. Then, we will use a toy example to illustrate why OOD-based methods outperform EC-based methods in continual learning.

The EC-based methods [64], [65] as in illustrated in Fig. 2 (a) add new class-specific classification heads when new tasks arrive. These new heads are trained alongside the existing ones to recognize all classes seen so far. During testing, this single classifier incorporating all heads is used for classification.

In contrast, our OOD-based method operates differently. As shown in Fig. 2 (b), when a new task is introduced, only one OOD classification head is added, which is used to recognize all old classes. During testing, samples are evaluated to determine whether they belong to the current task or previous tasks. If they belong to the previous tasks, they are tested using the classifier from the previous tasks until their task ID is identified.

Next, We will present a toy example with CIFAR100, B0-20 setting(dividing the 100 classes equally to 20 tasks) to



(a) first task (b) EC-based second task (c) OOD-based second task

Fig. 3: Visualization of the difference between EC-based and OOD based continual learning methods. We use the first two tasks in CIFAR100 B0-5 setting as the toy example. (a) is the t-SNE visualization after training the first 5 classes. (b) and (c) is the following 5 classes distribution using EC-based and OOD based methods respectively. Best viewed in color.

demonstrate the superiority of our OOD-based method. Firstly, in continual learning, traditional methods continuously expand classification heads as tasks increase to accommodate the increasing number of classes. Linear classifiers need to delineate new feature spaces for the incoming classes in the feature space, inevitably affecting the learned feature distribution and leading to catastrophic forgetting. In contrast, OOD-based continual learning methods maintain a constant number of classification heads for each task. The model focuses on learning the feature distribution of new classes in new tasks while treating all old classes as a single class. This significantly simplifies the variation in the feature space, thereby reducing forgetting. As shown in Fig. 3, EC-based methods need to incorporate the five new classes into the feature space learned from the previous five classes in subsequent tasks, making misclassification more likely. On the other hand, OOD-based methods only need to expand one OOD classification head (for all old classes). By sufficiently learning the feature distribution of new classes, old classes can still be recognized, achieving a balance between stability and plasticity.

Furthermore, due to device limitations and privacy concerns, storing (or only retaining a small portion of) old class samples in continual learning scenarios is generally not feasible. In EC-based methods, the imbalance between the large number of samples from new classes and the small number (or absence) of samples from old classes can easily lead to bias in the classifier, resulting in a tendency to classify samples as belonging to new classes and consequently forgetting old knowledge. In contrast, OOD-based methods classify all old classes as a single class (OOD class), significantly reducing the imbalance between new and old classes. This approach exhibits strong robustness to an increasing number of tasks (decreasing number of samples per old class) while mitigating the issue of catastrophic forgetting.

IV. Method

In this section, we first present the problem definition of CIL. Then, we introduce the method of applying OOD detection for task identification to implement CIL, followed by how to use the CLIP zero-shot classification to further improve OOD detection accuracy, thereby improving the performance of CIL. The whole pipeline of our proposed method can be found in Fig. 4.

A. Preliminaries

Class incremental learning for image classification The goal of CIL is to learn a sequence of tasks T so that the model can uniformly identify all classes contained in these tasks. Given a sequence of tasks $D = \{D_t\}_{t=1}^{T}$, where $D_t = \{(\mathbf{x}_i^t, y_i^t)\}_{i=1}^{N_t}$ is the dataset of task t, $\mathbf{x}_i^t \in \mathcal{X}_t$ is an input sample and $y_i^t \in \mathcal{Y}_t$ is its corresponding label, $|\mathcal{Y}_t| = N$ is the number of classes each task contains. The category spaces of all tasks are disjoint (*i.e.*, $\mathcal{Y}_t \cap \mathcal{Y}_{t'} = \emptyset, \forall t \neq t'$). During the training of task t, all training samples of the current task are available and only a few training samples of all previous tasks are stored in the exemplar set $\mathcal{V}$ for use.

From Task incremental to Class incremental for image classification In addition to CIL, TIL has been widely studied. Similar to CIL, TIL is also asked to identify test samples from a specific class in a task k after being trained on a sequence of tasks D. However, unlike CIL, which does not provide the task ID prior to test samples, TIL provides the ground-truth task ID for each test sample.

Inspired by this, some recent works [19]–[21] introduce a set of independent classifiers for each task, and employ an OOD detection mechanism to judge whether the testing sample belongs to a certain task without specifying a task ID. In order to achieve high-quality OOD detection, their OOD detection module either uses off-the-shelf OOD methods or uses exemplar samples to train the classification head of each task that adds an additional OOD output head. At the same time, in order to ensure the accuracy of intra-task classification for each task, they introduce some parameter isolation methods of TIL to overcome the problem of catastrophic forgetting. Although these methods have achieved improved results, the OOD detection accuracy and intra-task classification accuracy

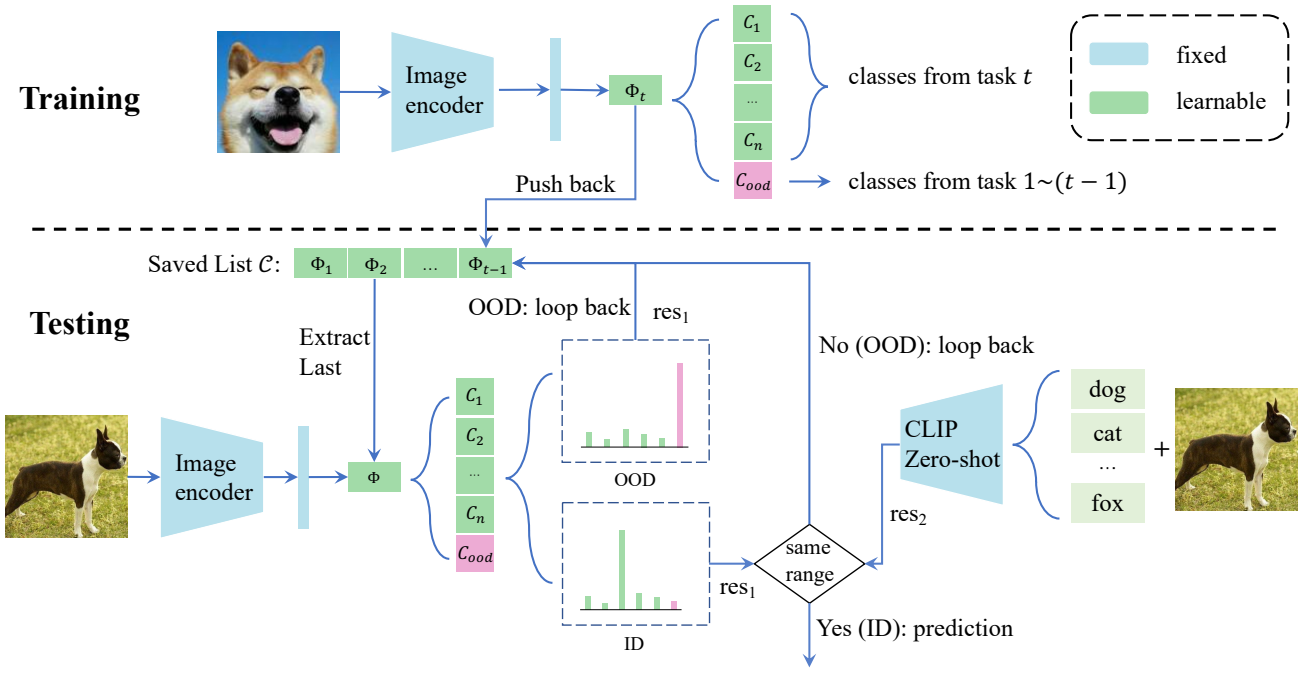


Fig. 4: During training task t $(t > 1)$, we fine-tune a classifier with $(N+1)$ heads, where N is the number of classes included in the current task (ID) and 1 represents all classes encountered by the model previously (treated as an OOD class), with all saved exemplars labeled as one class. After training, the classifier is fixed and appended to a classifier list. During testing, we extract the last classifier from the saved list and perform inference on the test image. If the result res_1 shows that it belongs to an OOD class (indicating that it does not belong to the task corresponding to the current classifier), we discard the current classifier and extract the next classifier from the saved list to continue inference until we encounter a classifier that outputs an ID class. At this point, we input the test image into the CLIP zero-shot classifier and obtain the result res_2. If res_1 and res_2 belong to the same task, we obtain the final classification result res_1 for the test image. Otherwise, we continue selecting the next classifier and repeat the above process.

are still difficult to guarantee under the incremental training of a moderate number of tasks from scratch.

B. OOD task identification

Pre-trained image-text backbone for CL For the purpose of improving OOD detection and intra-task classification performance, it is a good choice to introduce a large-scale Image-Text pre-trained model. CLIP [23] has demonstrated strong adaptability capability on many downstream tasks, thanks to the generalization representation obtained by training with the contrastive loss on up to 400M Image-Text pairs. Specifically, CLIP is a two-stream network consisting of two encoders (E^I, E^T) that map image and text data to multi-modal feature space respectively. Given a testing image, zero-shot image classification can be performed by computing cosine similarity between its visual feature embedding and text feature embedding of all the candidate classes.

Learning an OOD classifier for image classification with data In this section, we elaborate on how to build a CIL model with OOD detection by employing the visual encoder of a pre-trained CLIP. As mentioned above, a large-scale pre-trained CLIP already provides a good enough visual representation, so we use the frozen CLIP visual encoder as the backbone network for CIL training to avoid catastrophic forgetting. In order to implement OOD detection-based CIL, we learn a separate classifier for each task and store them in a classifier list $\mathcal{C}$ sequentially after training. Given a sequence of tasks D, for the first task of CIL, we fine-tune a linear classification layer $f(;\Phi_1)$ with standard cross-entropy loss based on the backbone network with all the training samples in $\mathcal{D}_1 = \{(x_m, y_m) : m \in [M]\}$:

$$\mathcal{L}_s = \frac{1}{M}\sum_{m=1}^{M}\mathcal{H}\left(y_m, f\left(z_m;\Phi_1\right)\right) \tag{1}$$

where M is the number of samples in D_1, $z_m \in \mathbb{R}^d$ is extracted feature embedding of x_m by using the frozen E^I, $\Phi_1 \in \mathbb{R}^{d\times N}$ is the parameters of first linear classification layer in $\mathcal{L}$. $d = 512$ denotes the dimension of the feature obtained by CLIP image encoder, while N is the number of classes in one task (we assume that each task has an equal number of classes, also known as B0 setting). $\mathcal{H}(\cdot,\cdot)$ is cross-entropy loss.

For the t^{th} $(t > 1)$ task, since we need to take all classes of samples other than the current task as OOD samples, we treat all the OOD samples as the same extra ODD class y^{ood}, then we fine-tune the linear classification layer $f(;\Phi_t)$ with the extra OOD class with following loss with all the training samples in $\mathcal{D}_t = \{(x_m, y_m) : m \in [M]\}$: and $\mathcal{V} = \{(x_v, y_v) : v \in [V]\}$:

Algorithm 1 Training algorithm

Input: X^t ▷ training examples in current task
Input: $x_v^1, \ldots, x_v^{t-1}$ ▷ exemplars from previous tasks

require: Φ_r ▷ random initiate new classifier
require: $\mathcal{C}_{t-1} = (\Phi_1, \ldots, \Phi_{t-1})$ ▷ saved classifier list

1: $X^{OOD} \leftarrow \text{RELABEL}(x_v^1, \ldots, x_v^{t-1})$
2: $\Phi_t \leftarrow \text{UPDATE}(X^{OOD}, X^t, \Phi_r)$ ▷ Eq.2
3: $x_v^t \leftarrow \text{HERDING}(X^t)$ ▷ exemplar selection
4: $\mathcal{C}_t = (\Phi_1, \ldots, \Phi_{t-1}, \Phi_t)$ ▷ append to list
5: **return** Φ

Algorithm 2 Testing Process

Input: x_{test} ▷ image to be tested

require: $\Phi = (\Phi_1, \ldots, \Phi_T)$ ▷ saved classifiers
require: E^I, E^T ▷ CLIP encoders
require: $\mathcal{W} = \{W_t\}_{t=1}^T$ ▷ category names

1: **for** t = T...2, 1 **do** ▷ iterate back to front
2: $\text{res}_1 \leftarrow f(E^I(x_{\text{test}}); \Phi_t)$ ▷ results from classifier
3: **if** $\text{res}_1 = \text{OOD}$ **then**
4: **continune** ▷ OOD: loop back
5: $\text{res}_2 \leftarrow \text{ZERO-SHOT}(x_{\text{test}}, \mathcal{W}, E^I, E^T)$
6: **if same-task**$(\text{res}_1, \text{res}_2)$ **then**
7: **return** res_1 ▷ ID: return prediction

$$\mathcal{L}_s = \frac{1}{M+V}\Big(\sum_{m=1}^{M} \mathcal{H}\left(y_m, f\left(z_m; \Phi_t\right)\right) + \sum_{v=1}^{V} \mathcal{H}\left(y^{ood}, f\left(z_v; \Phi_t\right)\right)\Big) \tag{2}$$

where M is the number of samples in D_t, V is the number of samples in the exemplar set $\mathcal{V}$, $z_v \in \mathbb{R}^d$ is extracted feature embedding of x_v by using the frozen E^I, $\Phi_t \in \mathbb{R}^{d\times(N+1)}$ is the parameters of t^{th} linear classification layer in $\mathcal{L}$. Algorithm 1 describes one training process of a specific task corresponding to OOD-based continual learning.

After incremental training for all the tasks, we can use TIL-like testing form for CIL testing and the task ID is automatically determined by the OOD detection module rather than manually specified: given a testing sample $\boldsymbol{x}_{\text{test}}$, we extract a classifier for each task from the classifier list $\mathcal{C}$ in the order of back to front and classify $\boldsymbol{x}_{\text{test}}$, if $\boldsymbol{x}_{\text{test}}$ is recognized as OOD in a classifier, it is considered that the sample does not belong to this task, and the next classifier is used for classification until the sample is recognized as one of the class in that task (ID) and this class is used as the final result.

Zero-shot OOD detector with image-text pertaining models Although it is a simple and efficient practice to add an additional channel to each classification head to detect OOD samples that do not belong to this task, the accuracy of OOD detection is still not satisfactory and eventually results in sub-optimal classification performance. To alleviate this problem, we add an additional OOD module that takes advantage of the powerful zero-shot recognition capability of CLIP.

Specifically, we design a two-level OOD sample detection mechanism based on fine-tuned visual classifier and CLIP Image-Text zero-shot classifier. During the testing stage, the testing sample $\boldsymbol{x}_{\text{test}}$ is firstly conducted classification / OOD detection by using visual classifiers popped up from $\mathcal{C}$ in the order of back to front. If $\boldsymbol{x}_{\text{test}}$ is recognized as OOD by a visual classifier, we will abandon the current classifier and continue to iteratively select the next classifier until we encounter a classifier that identifies the current test image as one of the ID class. Once $\boldsymbol{x}_{\text{test}}$ is recognized as ID, we will conduct further CLIP zero-shot classification: given the text label $\mathcal{W} = \{W_t\}_{t=1}^T$ of each class in $D = \{D_t\}_{t=1}^T$, since our CIL testing takes visual classifier from the classifier list in the order of back to front, for the current task t, we only provide text labels for all the classes that the model has seen while training this task $\left(\textit{i.e. } \mathcal{W} = \{W_t\}_{t=1}^t\right)$. This text label is prepended by a prompt to generate the input of text encoder E^T, prompt usually has the form of "a photo of a [CLASS]". The final output of E^T can be defined as $\{\boldsymbol{w}_i\}_{i=1}^{t\times N}$. The final zero-shot prediction probability can be computed as:

$$p(y = i \mid \boldsymbol{x_{test}}) = \frac{\exp\left(\cos\left(\boldsymbol{w_i}, \boldsymbol{z_{test}}\right)/\tau\right)}{\sum_{j=1}^{t\times N} \exp\left(\cos\left(\boldsymbol{w_j}, \boldsymbol{z_{test}}\right)/\tau\right)} \tag{3}$$

where $z_{test} \in \mathbb{R}^d$ is extracted feature embedding of x_{test} by using the frozen E^I and τ is a temperature scaling parameter used to control the sharpness of the output.

If the zero-shot classification result also indicates that the test image belongs to the task corresponding to the current classifier, we will use the classification result obtained by this visual classifier as the final prediction result. Otherwise, we continue to iterate and select the next classifier, and repeat the above operation.

Algorithm 2 gives a detailed description of the testing process of our proposed model. In line 2, we obtained the result res_1 of the current classifier. We proceed to the next line 3 only when we encounter a classifier that predicts the current sample as an ID class. Otherwise, we go back to line 1 and continue extracting the next classifier. In line 5 we make the CLIP zero-shot prediction res_2 with label names corresponding to the task ID predicted by the visual classifier. In line 6 we compare the two results obtained from two modules, if these two results both indicate the test sample belongs to current task, we return the final prediction res_1. Otherwise, we go back to lines 1-2 and select the previous classifier to continue the whole process again.

V. Experimental Results

In this section, we first describe the implementation details and baselines, and then we compare our method with the competing methods. An ablation study is followed to understand our method from different perspectives. All results are averaged among three experiments under different class orders, detailed results can be found in the supplementary material.

TABLE I: Results of our method and other state-of-the-art methods on CIFAR-100 under different settings. "# tasks" denotes the number of tasks. * means we replaced the ImageNet-21K pre-trained model with CLIP image encoder for fair comparison. In the methods column, italic bold text represents parameter-growing, while the others indicate parameter-static.

Method	Backbone	No. Para (M)	CIFAR-100 B0						CIFAR-100 B50			
			# tasks						# tasks			
			5		10		20		5		10	
			avg	last	avg	last	avg	last	avg	last	avg	last
iCaRL [6]	ResNet	11.2	71.1	59.6	65.3	50.9	61.2	44.9	65.1	56.0	58.6	49.5
LUCIR [66]			62.8	46.9	58.7	42.9	58.2	41.1	64.3	52.7	59.9	48.2
BiC [67]			73.1	61.5	68.8	53.6	66.5	47.8	66.6	55.1	60.3	48.7
WA [68]			72.8	60.3	69.5	53.7	67.3	48.2	64.0	52.8	57.9	48.1
PODNet [36]			66.7	51.5	58.0	40.7	54.0	35.8	67.3	56.0	64.0	51.7
RPSNet [44]			70.5	60.8	68.6	56.2	-	-	-	-	-	-
DER [18]			76.8	67.3	75.4	64.4	74.1	62.6	73.2	66.0	72.8	65.6
Dytox [41]	conViT	11.0	-	-	72.9	60.0	72.2	57.0	-	-	-	-
Continual-CLIP [69]	ViT-B/16	85.9	74.0	66.7	75.1	66.7	75.9	66.7	69.7	66.7	69.5	66.7
Linear Probe			78.8	72.9	80.2	72.4	78.9	69.4	78.9	71.2	77.6	72.3
L2P* [64]			78.1	65.7	79.2	67.5	79.2	68.4	77.2	68.3	76.5	69.2
DualPrompt* [65]			78.6	72.6	80.3	69.4	80.6	70.2	79.6	56.3	62.5	32.8
CODA-Prompt* [70]			68.3	33.8	71.1	42.9	69.3	39.6	76.6	51.3	64.3	32.5
Ours	ViT-B/16	86.1	81.1	75.5	84.3	74.2	84.0	73.9	82.8	76.3	**83.0**	75.6
Ours (order 1)			81.3	**77.2**	83.6	**77.6**	83.0	72.9	**84.2**	76.0	82.1	**75.7**
Ours (order 2)			**82.4**	75.1	**85.0**	73.4	82.3	75.0	82.5	75.6	81.8	73.5
Ours (order 3)			82.1	74.4	83.0	74.3	**84.1**	**75.8**	82.8	**78.5**	83.4	74.4

TABLE II: Results of our method and other state-of-the-art methods on ImageNet-Subset under different settings.

Method	Backbone	No. Para (M)	ImageNet-Subset B0						ImageNet-Subset B50			
			# tasks						#tasks			
			5		10		20		5		10	
			avg	last	avg	last	avg	last	avg	last	avg	last
iCaRL [6]	ResNet	11.2	78.1	65.2	74.1	58.5	69.0	50.9	60.7	44.7	57.3	44.4
End2End [3]			75.5	64.0	70.1	53.0	68.3	48.9	61.0	52.1	58.5	52.2
UCIR [66]			76.0	64.0	70.5	55.3	64.7	47.8	77.2	68.2	66.9	56.8
PODNet [36]			78.2	66.2	72.3	57.2	66.7	48.9	80.3	73.5	79.0	70.8
UCIR-DDE [66]			77.2	65.8	71.7	56.8	66.2	40.0	78.8	68.1	68.4	57.9
RM [71]			75.5	62.2	70.4	53.2	65.4	45.7	56.9	41.8	57.7	37.3
DER(w/o P) [18]			-	-	77.2	66.7	-	-	-	-	78.2	74.9
DyTox+ [41]	conViT	11.0	-	-	77.2	67.7	-	-	-	-	-	-
Continual-CLIP [69]	ViT-B/16	85.9	84.8	75.3	85.0	75.3	86.6	75.3	79.2	75.3	79.3	75.3
Linear Probe			79.4	64.5	81.8	67.4	84.0	72.0	81.4	67.2	83.1	72.0
L2P* [64]			81.7	76.5	80.3	75.2	80.1	75.7	74.9	74.2	72.3	72.6
DualPrompt* [65]			75.4	61.1	80.7	67.4	83.9	74.2	74.2	49.8	62.1	22.4
CODA-Prompt* [70]			51.6	24.9	64.1	34.8	69.8	44.0	65.1	28.8	57.3	20.0
Ours	ViT-B/16	86.1	84.9	77.6	85.3	77.4	85.6	77.2	85.2	**78.5**	**85.0**	78.5
Ours (order 1)			83.5	77.1	84.4	76.2	87.1	**79.0**	85.4	77.3	83.2	78.4
Ours (order 2)			85.7	76.9	85.1	75.6	86.6	75.8	84.4	77.5	84.3	77.5
Ours (order 3)			**85.8**	**78.5**	**86.6**	**78.5**	**87.8**	77.3	**86.6**	77.0	84.7	**79.3**

A. Experimental setup

Implementation details We conducted all experiments on a Linux cluster with 4 V100 GPUs. We tested the model on CIFAR-100, ImageNet-Subset [66] and 5-Datasets [72]. 5-Datasets is a special benchmark designed for cross datasets evaluation of continual learning. It contains five datasets consisting of CIFAR-10 [73], MNIST [74], Fashion-MNIST [75], SVHN [76] and notMNIST, with each dataset containing ten classes, treated as a single task, for a total of five tasks.

The B50 setting refers to the model first being trained on a larger dataset with 50 classes, followed by adding 5 or 10 new classes for each task. The other setting, B0, keeps the number of classes per task constant, dividing the 100 classes equally into 5, 10, or 20 tasks. We used the pre-trained CLIP image encoder and followed the original CLIP text prompt by using "a bad photo of [CLS]". The classifier consisted of an MLP with only one hidden layer. We used Adam [77] as the optimizer, and we trained each task equally for 10 epochs with a learning rate of 0.01 and weight decay of 0.0002. For all settings, we chose to fix the exemplar size at 2000 and set temperature scaling parameter τ to 2. In the CLIP zero-shot OOD detector module, we follow the approach of Continual CLIP [69], whereas we go deeper into the classifier list, the number of classes that zero-shot needs to distinguish decreases and the accuracy of zero-shot increases accordingly which helps us to better identify task IDs.

Competing methods We compare our method against 10 traditional baseline methods, iCaRL [6], LUCIR(LUCIR-DDE) [66], PODNet [36], RM [71], DER [18]. The baseline Linear Probe and our approach are based on the same pre-trained model, and a Linear Probe classifier head expands

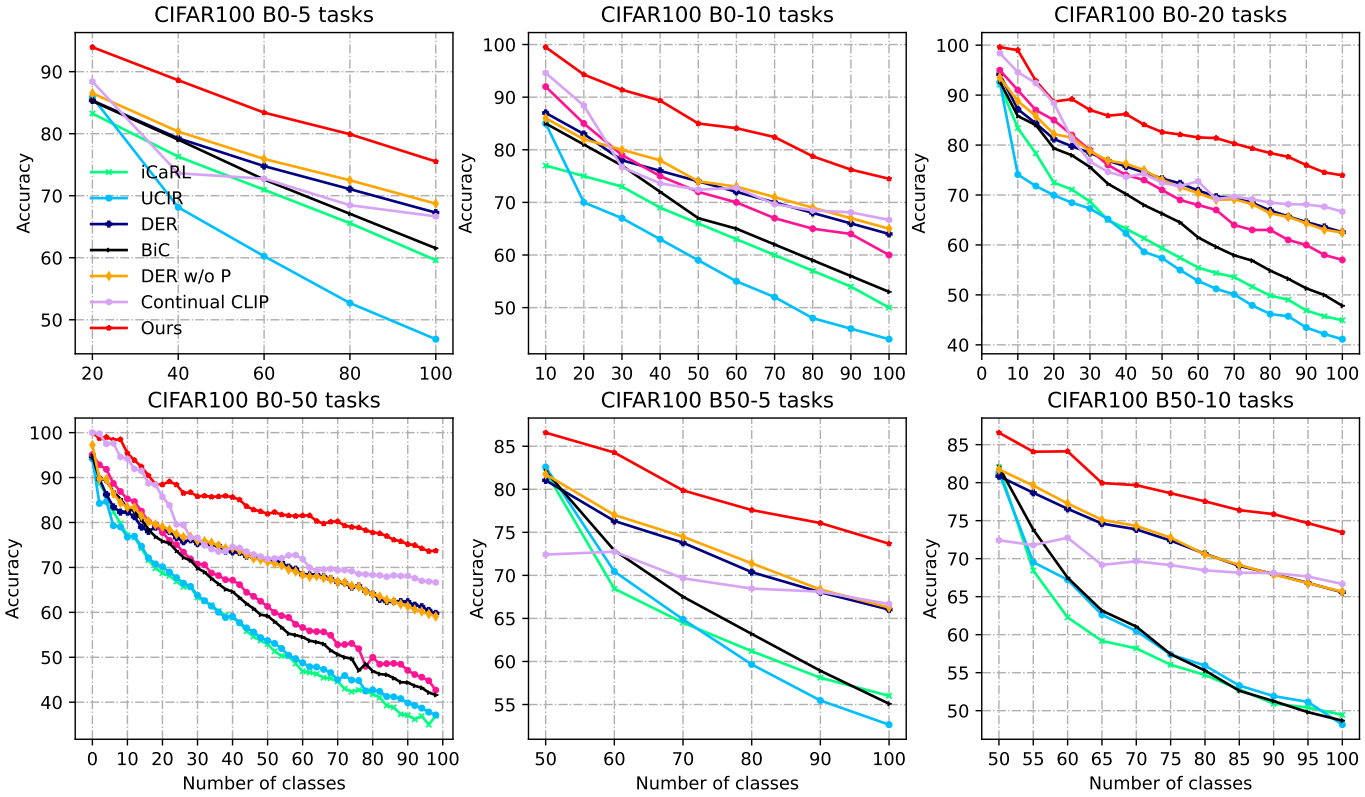


Fig. 5: Detailed comparison results of our method with the state-of-the-art on CIFAR-100.

with each task, and it directly uses the classifier head for classification. In addition, we also conducted a comparison with prompt-based methods L2P [64], DualPrompt [65] and CODA-Prompt [70]. These methods all employ a backbone pre-trained on ImageNet-21K and have been tested on CIFAR-100 and ImageNet-Subset. As mentioned in the literature [78], using NMC alone, without any additional training, the ImageNet-21k pre-trained ViT model achieves better results than L2P on all five datasets. Thus, we replaced the pre-trained model with CLIP image encoder for fair comparison for these three methods.

We divided the baseline into two parts: parameters-growing and parameters-static. DER [18] continuously replicates and expands the backbone throughout the learning process, while DualPrompt [65] assigns an expert prompt for each new task, which we classify as parameters-growing and indicate in the table with italicized bold text. Methods that only extend the classifier simply are considered parameters-static.

B. Comparison to state-of-the-art methods

Evaluation on CIFAR-100 As shown in Table I, our results consistently outperform the baselines in every setting. In the B0-5 tasks setting, we achieved about 8.2% improvement over DER at the last task. Note that the prompt-based methods perform much less than expected with the CLIP image encoder. We argue that most of the performance gains of these methods may come from the ImageNet pre-trained backbone instead of the design of the prompt pool. Additionally, we can see that in settings with a greater number of tasks (B0-20, B50-10), the extent of our model's improvement over SOTA is slightly reduced because the more tasks there are, the longer our OOD testing chain becomes, making errors more likely. However, due to our zero-shot OOD detector, which not only assists in identifying task ID but also guarantees the lower bound of our model, we can maintain remarkably high performance even when dealing with a large number of tasks.

TABLE III: Results of our method and other state-of-the-art methods on 5-Datasets.

Method	Backbone	Para	Acc
ER	ResNet	11.0	80.32
BiC		11.0	78.74
L2P	ViT-B/16	85.9	81.84
DualPrompt		85.9	77.91
CODA-Prompt		85.9	64.18
Ours	ViT-B/16	86.2	**87.31**

Evaluation on ImageNet-Subset Experiments on ImageNet-Subset (Table II) also demonstrate the effectiveness of our model. In each setting, our results are significantly higher than the SOTA. It is worth noting that our model maintains an average and last result of around 85 and 77 respectively across all five settings, which other models cannot match. This proves that although more tasks and a longer prediction chain can introduce more errors, the impact of these errors on performance is much smaller than the bias introduced to the

TABLE IV: Experiments of the Few-shot Class Incremental setting on CIFAR100.

Method	0	1	2	3	4	5	6	7	8	PD↓
iCaRL [6]	64.10	53.28	41.69	34.12	27.93	25.06	20.41	15.48	13.73	50.37
EEIL [3]	64.10	53.11	43.71	35.15	28.96	24.98	21.01	17.26	15.85	48.25
LUCIR [66]	64.10	53.05	43.96	36.97	31.61	26.73	21.23	16.78	13.54	50.56
TOPIC [79]	64.10	55.88	47.07	45.16	40.11	36.38	33.96	31.55	29.37	34.73
CEC [46]	73.07	68.88	65.26	61.19	58.09	55.57	53.22	51.34	49.14	23.93
F2M [45]	71.45	68.10	64.43	60.80	57.76	55.26	53.53	51.57	49.35	22.06
MetaFSCIL [50]	74.50	70.10	66.84	62.72	59.48	56.52	54.36	52.56	49.97	24.53
Entropy-reg [80]	74.40	70.20	66.54	62.51	59.71	56.58	54.52	52.39	50.14	24.26
L2P* [64]	91.22	88.35	82.80	70.34	68.66	64.34	60.78	58.32	54.89	36.33
DualPrompt* [65]	91.08	87.96	84.55	71.31	68.45	64.52	61.20	59.31	54.67	36.41
CODA-Prompt* [70]	**93.55**	**89.91**	**86.54**	76.65	71.91	67.12	64.52	62.89	59.32	34.23
Ours	88.93	85.86	83.14	**80.57**	**78.04**	**76.41**	**74.02**	**71.96**	**71.04**	**17.89**

classifier in previous methods as the number of tasks increases. This demonstrates the superiority of our OOD-based method.

Evaluation on multiple runs The last three rows of Table I and Table II present the results of our method under different class orders. Although the performance varies slightly across different orders, our method consistently shows significant improvements over the comparison methods. This indicates that our OOD-based approach is not dependent on a fixed learning sequence.

Evaluation on 5-Datasets Since each task in the 5-Datasets benchmark represents a different dataset, it is necessary to follow a unified training order for fair comparison with other baselines. We followed the settings of DualPrompt [65], using the sequence SVHN, MNIST, CIFAR-10, NotMNIST, and FashionMNIST. The last task accuracy was used as the evaluation metric. As shown in Table III, our model significantly outperforms other methods. This is primarily due to our OOD detection approach, which is better suited to scenarios involving different datasets, resulting in a substantial improvement in task recognition accuracy and achieving near-zero forgetting.

Experiments on few-shot settings We also conduct experiments in a few-shot setting on CIFAR100 (Table IV), where the first task includes all data from all 60 classes and the subsequent 8 tasks each contain 5 classes with only 5 samples per class. The competing methods are all EC-based or prototype-based methods including iCaRL [6], EEIL [3], LUCIR [66], TOPIC [79], CEC [46], F2M [45], MetaFSCIL [50], Entropy-seg [80] and prompt-based method L2P [64], DualPrompt [65] and CODA-Prompt [70]. It can be observed that our method may not initially outperform some EC-based methods, but as the number of seen classes increases (much-crowded feature spaces), our advantage becomes more prominent. In the last task, we achieved a performance of 71.04, at least 10 points higher than the EC-based methods.

C. Running Time analysis

By maintaining the CLIP image encoder as a fixed component, a single pass through the image encoder suffices for subsequent classifier determination. Furthermore, the subsequent classifier consists of just one hidden layer with minimal parameters, resulting in a negligible increase in computational overhead and memory consumption. In Table V, we present a comprehensive comparison of model parameters, inference latency, and accuracy in a B0-10 ImageNet-Subset setting. This includes the number of initial parameters, the final parameters after training, and the trainable parameters for various methods. The results unequivocally demonstrate that our approach imposes minimal additional latency when compared to recent methods. Remarkably, our method achieves significant performance improvements with only a modest increase in parameters.

D. Successful and failed cases

Here we discuss common failure cases in OOD classification and how integrating CLIP can improve overall accuracy. Using the CIFAR100 B0-20 setup as an example, the model, having learned three tasks (15 classes), encounters an "aquarium fish" during testing.

Without CLIP (left side of Fig. 6), an imbalance between in-distribution (ID) and OOD samples leads to misclassification, where the model incorrectly assigns images to the wrong task (ID). This happens due to limited exemplar memory in the first classifier, causing "early stops." With CLIP assistance (right side), both CLIP and the classifier must agree on whether a sample belongs to an ID class before final classification. If not, the sample moves to a previous classifier. CLIP's zero-shot classification, which uses class names, improves task identification. While CLIP helps with task assignments, the fine-grained classification is still handled by the classifier, leading to more accurate results. This combination significantly boosts overall model accuracy.

In Fig. 7 we present successful and failed cases under both the 5-Datasets and CIFAR100 B0-10 settings. In the 5-Datasets benchmark, since the final accuracy for SVHN is only 72%, many errors appear as misclassifications within the same task. It can be observed that in the incorrect predictions, the image content often resembles the misclassified result. In the correct predictions, the distribution gap between tasks is large, demonstrating that our method can accurately predict tasks. In CIFAR100, errors typically occur when the textual and visual

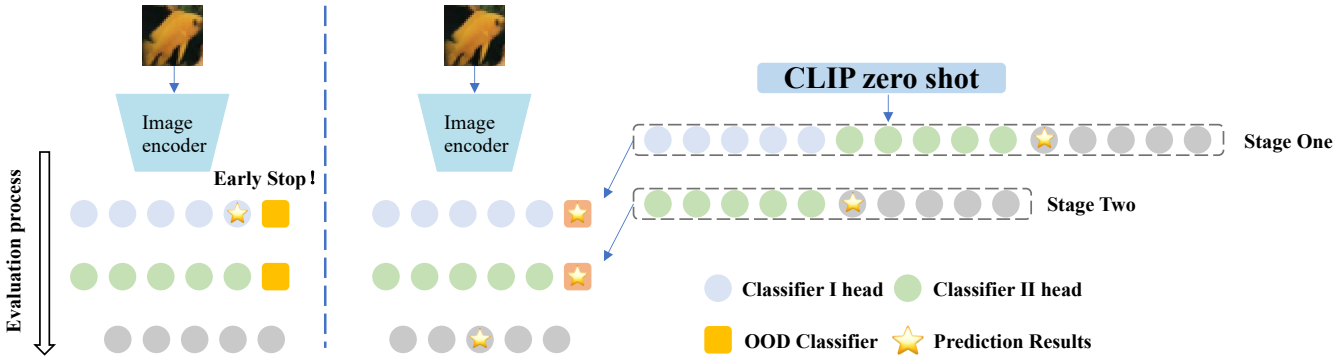


Fig. 6: Failed case for OOD incremental classifiers and successful case for our CLIP zero-shot guided method.

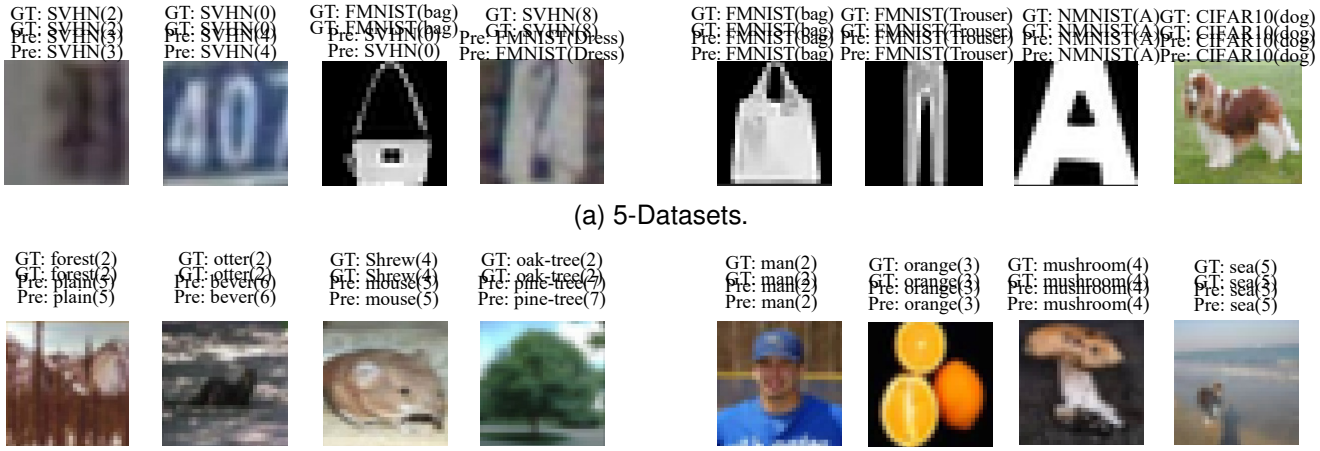


Fig. 7: Examples of successful and erroneous cases under the 5-Datasets and CIFAR100 benchmarks: (a) parentheses indicate the true labels, and (b) parentheses show the corresponding task ID, with a total of 10 tasks.

TABLE V: Comparison of the number of parameters and the inference latency. We conducted all experiments on a 3090 Ubuntu machine, using the official code for DER, DyTox, and L2P. We measured parameters in millions and latency in seconds. The latency was reported for a batch of 128 images with size (3, 224, 224).

Method	Initial para	Final para	Trainable para	Latency	Accuracy
DER	11.22	112.27	11.22	1.35	77.2
DyTox	11.01	11.02	11.02	1.83	77.2
L2P	85.9	85.9	0.12	0.63	80.3
Ours	86.19	88.89	0.27	1.86	85.3

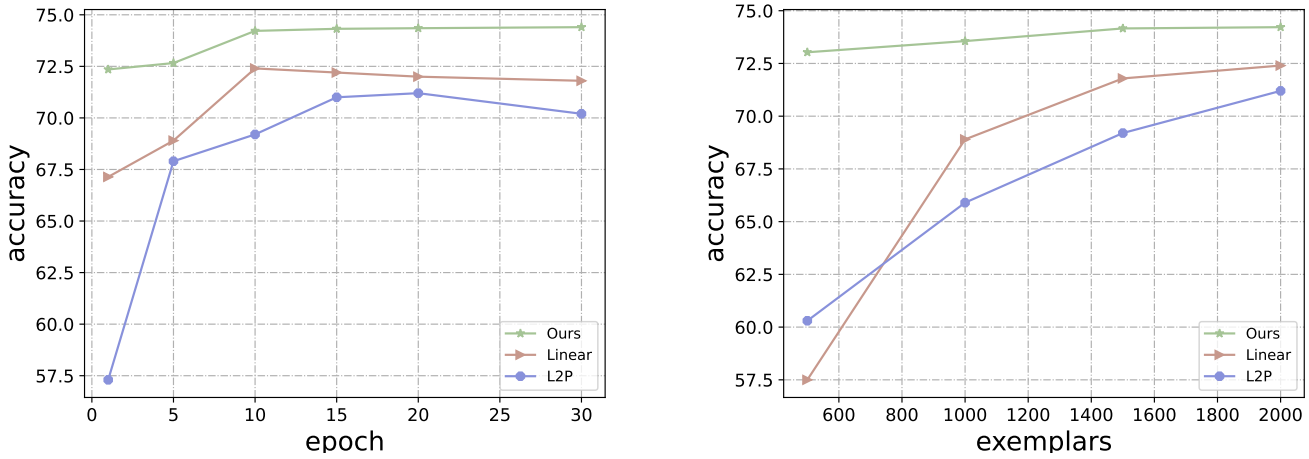
meanings overlap, such as between 'otter' and 'beaver,' or 'mouse' and 'shrew,' causing both the CLIP zero-shot and OOD incremental classifier to fail simultaneously.

E. Merits of the OOD-based method

As discussed in the Motivation section, Our OOD-based method does not require continuously expanding classification heads with the increase of tasks, therefore it needs to insert the features of new classes into the feature space already well learned from old classes. Therefore, 1) the model does not require too many adjustments thus the fine-tuning process is fast. 2) It is not sensitive to the number of exemplars because we treat all exemplars as belonging to the same class, greatly reducing the issue of class imbalance of traditional exemplar-based methods. We will now demonstrate these points through two experiments.

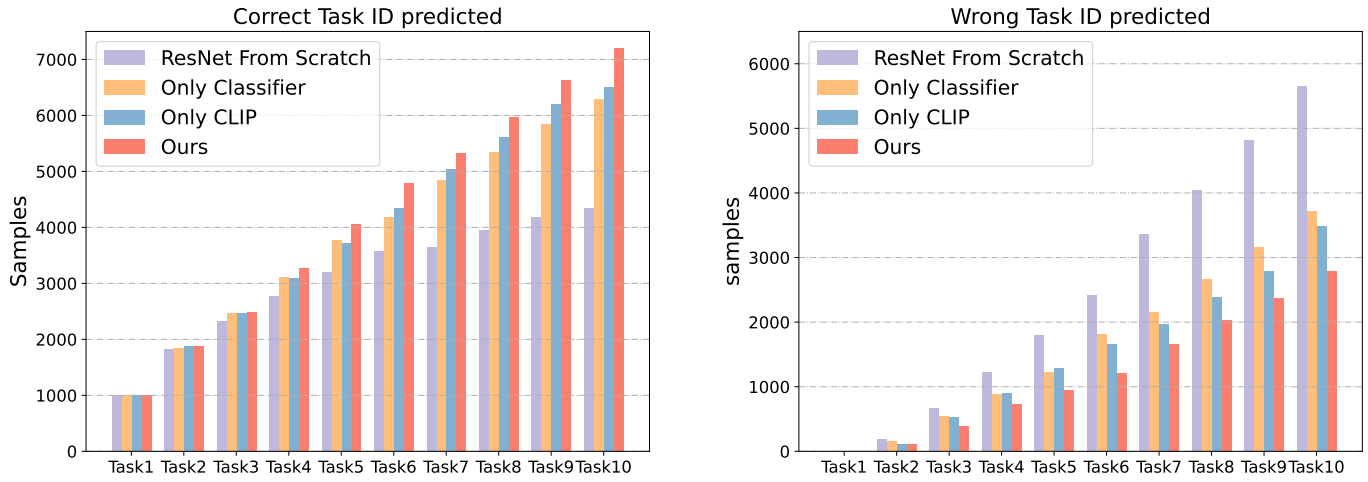
We conducted experiments on CIFAR100, with a B0-10 setting, comparing our OOD-based method with two Extension Classifier (EC) based methods: Linear Probe and L2P. Firstly, as shown in Fig. 8 (a), Our method achieves comparable performance even with just one epoch training, and its performance gradually improves with the increase of epochs. On the other hand, the EC-based methods show poor performance with a small number of epochs (under-fitting), and as the number of epochs increases, they tend to overfit to the classes of the new task due to class imbalance, leading to catastrophic forgetting and suboptimal performance.

Additionally, in Fig. 8 (b), it can be observed that the performance of EC-based methods significantly drops when the number of exemplars is low (500, 1000), indicating their classifiers heavily rely on the rehearsal memory. In contrast, our method exhibits strong robustness to changes in the number of exemplars, demonstrating that our performance



(a) Different numbers of training epochs (b) Different numbers of exemplars

Fig. 8: (a) illustrates the variation of model performance with the number of training epochs, while (b) demonstrates the variation of model performance with the number of exemplars. It can be observed that the performance of EC-based methods relies more on fine-tuning and a large number of exemplars, whereas our method exhibits a certain level of robustness to both the number of epochs and exemplars.



(a) Correct Task ID Predictions (b) Wrong Task ID Predictions

Fig. 9: The task prediction performance of our method and other ablation settings on CIFAR-100 B0-10. "Only CLIP" denotes we solely use CLIP for OOD detection. "Only Classifier" denotes we only use visual classifiers for OOD detection without CLIP as an assistant. We also trained a ResNet32 network from scratch for OOD detection, which is shown in "Resnet From Scratch".

stems from the proposed OOD-based classification method rather than exemplars.

F. Ablation study

Ablation study on two OOD task identification modules In this section, we discuss the effectiveness of each module we proposed. Firstly, we tested the simple classifier-based OOD detection without using CLIP as assistance (Only Classifier). In addition, we solely used CLIP for OOD detection. Specifically, We trained ten completely isolated classifiers for ten tasks, and during testing, we first used the zero-shot result to determine its task ID and then used the classifier saved from that task for classification (Only CLIP). As shown in the Figure 10, using Only Classifiers or Only CLIP for OOD detection is not effective enough. However, combining the two components results in at least 2% improvement on both the last task and average accuracy.

We argue that the pre-trained image encoder in CLIP is also a fundamental component for achieving high performance in our model. As validated by literature [24], CLIP has demonstrated its ability in detecting OOD samples on its own.

TABLE VI: The performance comparison on CIFAR-100 B0-10 setting between training a separate task recognizer [25] and our method.

task	2	3	4	5	6	7	8	9	10
Additional OOD	94.9	83.4	73.5	70.9	67.9	63.3	59.7	58.3	51.1
Ours	88.0	84.0	80.9	78.3	77.0	75.1	74.3	71.7	70.4

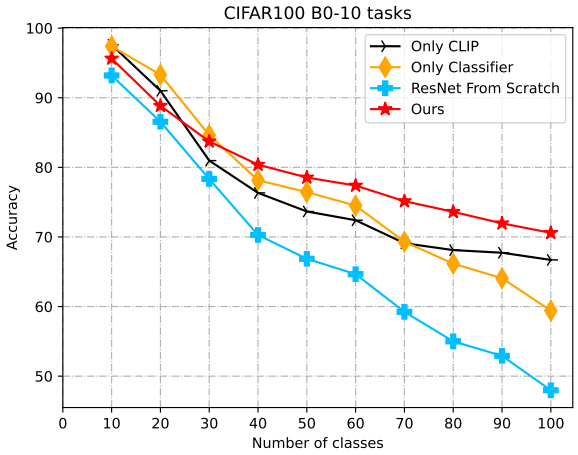


Fig. 10: The performance of our method and other ablation settings on CIFAR-100 B0-10.

Here, we tested training a ResNet32 network from scratch for OOD detection based continual learning, but found that the performance was much worse than finetuning a pre-trained model. As shown in Fig. 9 and Fig. 10, the method without pre-training performs very poorly in the later tasks, achieving only 47% on the last task.

Comparison to other OOD implementation We also explored other OOD detection-based continual learning methods [25]. We trained a task-specific classifier for each task and a task recognizer treating all classes within each task as the same class. During testing, we first used this recognizer to identify which task the test image belongs to, and then used the corresponding task-specific classifier to make the final classification result.

We report the accuracy of two OOD modules in Table VI. It can be observed that the classifier directly trained on task ID shows a rapid performance decline as the number of tasks increases. On the last task, the OOD accuracy is only 51.1%, which implies even lower accuracy for the model itself. In contrast, our method leverages the high performance of CLIP zero-shot and the outstanding generalization of the CLIP image encoder.

VI. Conclusion

In this work, we proposed an OOD detection-based approach to transform CIL into TIL, while utilizing the CLIP pre-training model to further improve the accuracy of task recognition. We found that during the continual learning process, the pre-trained backbone can prevent the model from drifting to the current task, and the zero-shot OOD detection can improve the performance of ID recognition while ensuring the lower limit of the overall model accuracy in settings with a large number of tasks. We believe that continual learning based on pre-trained models is of great research value, and the OOD-based method can maximize the use of the pre-trained model's characteristics. In future work, it is worth exploring how to use existing OOD methods to design more reasonable training and testing continual learning methods.

REFERENCES

[1] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *PNAS*, 2017.

[2] G. M. van de Ven and A. S. Tolias, "Three scenarios for continual learning," *NIPS Workshops*, 2019.

[3] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *ECCV*, 2018.

[4] H. Cha, J. Lee, and J. Shin, "Co2l: Contrastive continual learning," in *ICCV*, 2021.

[5] W. Li, B.-B. Gao, B. Xia, J. Wang, J. Liu, Y. Liu, C. Wang, and F. Zheng, "Cross-modal alternating learning with task-aware representations for continual learning," *IEEE Transactions on Multimedia*, pp. 1–14, 2023.

[6] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *CVPR*, 2017.

[7] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro, "Learning to learn without forgetting by maximizing transfer and minimizing interference," *ICLR*, 2019.

[8] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio, "Gradient based sample selection for online continual learning," *NIPS*, 2019.

[9] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *PNAS*, 2017.

[10] M. Toldo and M. Ozay, "Bring evanescent representations to life in lifelong class incremental learning," in *CVPR*, 2022.

[11] K. Wang, J. van de Weijer, and L. Herranz, "Acae-remind for online continual learning with compressed feature replay," *Pattern Recognition Letters*, 2021.

[12] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *ICML*, 2017.

[13] S. Thuseethan, S. Rajasegarar, and J. Yearwood, "Deep continual learning for emerging emotion recognition," *IEEE Transactions on Multimedia*, vol. 24, pp. 4367–4380, 2022.

[14] A. Mallya, D. Davis, and S. Lazebnik, "Piggyback: Adapting a single network to multiple tasks by learning to mask weights," in *ECCV*, 2018.

[15] K. Du, F. Lyu, L. Li, F. Hu, W. Feng, F. Xu, X. Xi, and H. Cheng, "Multi-label continual learning using augmented graph convolutional network," *IEEE Transactions on Multimedia*, vol. 26, pp. 2978–2992, 2024.

[16] J. Serra, D. Suris, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," in *ICML*, 2018.

[17] M. Wortsman, V. Ramanujan, R. Liu, A. Kembhavi, M. Rastegari, J. Yosinski, and A. Farhadi, "Supermasks in superposition," *NIPS*, 2020.

[18] S. Yan, J. Xie, and X. He, "Der: Dynamically expandable representation for class incremental learning," in *CVPR*, 2021.

[19] G. Kim, C. Xiao, T. Konishi, Z. Ke, and B. Liu, "A theoretical study on solving continual learning," *NIPS*, 2022.

[20] G. Kim, S. Esmaeilpour, C. Xiao, and B. Liu, "Continual learning based on ood detection and task masking," in *CVPR*, 2022.

[21] G. Kim, B. Liu, and Z. Ke, "A multi-head model for continual learning via out-of-distribution replay," in *Conference on Lifelong Learning Agents*, 2022.

[22] Y. Wang, Z. Ma, Z. Huang, Y. Wang, Z. Su, and X. Hong, "Isolation and impartial aggregation: A paradigm of incremental learning without interference," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 8, 2023, pp. 10 209–10 217.

[23] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *ICML*, 2021.

[24] S. Fort, J. Ren, and B. Lakshminarayanan, "Exploring the limits of out-of-distribution detection," *NIPS*, 2021.

[25] Y. Ming, Z. Cai, J. Gu, Y. Sun, W. Li, and Y. Li, "Delving into out-of-distribution detection with vision-language representations," *NIPS*, 2022.

[26] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *TPAMI*, 2021.

[27] X. Liu, M. Masana, L. Herranz, J. Van de Weijer, A. M. Lopez, and A. D. Bagdanov, "Rotate your networks: Better weight consolidation and less catastrophic forgetting," in *ICPR*, 2018.

[28] Z. Li and D. Hoiem, "Learning without forgetting," *TPAMI*, 2017.

[29] P. Dhar, R. V. Singh, K.-C. Peng, Z. Wu, and R. Chellappa, "Learning without memorizing," in *CVPR*, 2019.

[30] A. Rannen, R. Aljundi, M. B. Blaschko, and T. Tuytelaars, "Encoder based lifelong learning," in *ICCV*, 2017.

[31] G. Yang, E. Fini, D. Xu, P. Rota, M. Ding, T. Hao, X. Alameda-Pineda, and E. Ricci, "Continual attentive fusion for incremental learning in semantic segmentation," *IEEE Transactions on Multimedia*, 2022.

[32] A. Douillard, Y. Chen, A. Dapogny, and M. Cord, "Plop: Learning without forgetting for continual semantic segmentation," 2021.

[33] T. Feng, M. Wang, and H. Yuan, "Overcoming catastrophic forgetting in incremental object detection via elastic response distillation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9427–9436.

[34] K. Shmelkov, C. Schmid, and K. Alahari, "Incremental learning of object detectors without catastrophic forgetting," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3400–3409.

[35] L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: Theory, method and application," *TPAMI*, 2023.

[36] A. Douillard, M. Cord, C. Ollion, T. Robert, and E. Valle, "Podnet: Pooled outputs distillation for small-tasks incremental learning," in *ECCV*, 2020.

[37] Y. Cong, M. Zhao, J. Li, S. Wang, and L. Carin, "Gan memory with no forgetting," *NIPS*, 2020.

[38] X. Liu, C. Wu, M. Menta, L. Herranz, B. Raducanu, A. D. Bagdanov, S. Jui, and J. v. de Weijer, "Generative feature replay for class-incremental learning," in *CVPR Workshops*, 2020.

[39] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *NIPS*, 2017.

[40] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *CVPR*, 2018.

[41] A. Douillard, A. Ramé, G. Couairon, and M. Cord, "Dytox: Transformers for continual learning with dynamic token expansion," in *CVPR*, 2022.

[42] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *CVPR*, 2017.

[43] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, A. A. Rusu, A. Pritzel, and D. Wierstra, "Pathnet: Evolution channels gradient descent in super neural networks," 2017.

[44] J. Rajasegaran, M. Hayat, S. H. Khan, F. S. Khan, and L. Shao, "Random path selection for continual learning," *NIPS*, 2019.

[45] G. Shi, J. Chen, W. Zhang, L.-M. Zhan, and X.-M. Wu, "Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima," *Advances in neural information processing systems*, vol. 34, pp. 6747–6761, 2021.

[46] C. Zhang, N. Song, G. Lin, Y. Zheng, P. Pan, and Y. Xu, "Few-shot incremental learning with continually evolved classifiers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12 455–12 464.

[47] L. Bertinetto, J. F. Henriques, P. H. Torr, and A. Vedaldi, "Meta-learning with differentiable closed-form solvers," *arXiv preprint arXiv:1805.08136*, 2018.

[48] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," *arXiv preprint arXiv:1807.05960*, 2018.

[49] E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, U. Evci, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P.-A. Manzagol *et al.*, "Meta-dataset: A dataset of datasets for learning to learn from few examples," *arXiv preprint arXiv:1903.03096*, 2019.

[50] Z. Chi, L. Gu, H. Liu, Y. Wang, Y. Yu, and J. Tang, "Metafscil: A meta-learning approach for few-shot class incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 14 166–14 175.

[51] K. Zhu, Y. Cao, W. Zhai, J. Cheng, and Z.-J. Zha, "Self-promoted prototype refinement for few-shot class-incremental learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6801–6810.

[52] J. Yang, K. Zhou, Y. Li, and Z. Liu, "Generalized out-of-distribution detection: A survey," 2022.

[53] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *ICLR*, 2017.

[54] S. Liang and Y. Li, "Enhancing the reliability of out-of-distribution image detection in neural networks," *ICLR*, 2018.

[55] Y.-C. Hsu, Y. Shen, H. Jin, and Z. Kira, "Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data," in *CVPR*, 2020.

[56] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," *NIPS*, 2018.

[57] E. Nalisnick, A. Matsukawa, Y. W. Teh, D. Gorur, and B. Lakshminarayanan, "Do deep generative models know what they don't know?" *ICLR*, 2019.

[58] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan, "Likelihood ratios for out-of-distribution detection," *NIPS*, 2019.

[59] Z. Xiao, Q. Yan, and Y. Amit, "Likelihood regret: An out-of-distribution detection score for variational auto-encoder," *NIPS*, 2020.

[60] J. Yang, K. Zhou, and Z. Liu, "Full-spectrum out-of-distribution detection," *arXiv preprint arXiv:2204.05306*, 2022.

[61] E. Techapanurak, M. Suganuma, and T. Okatani, "Hyperparameter-free out-of-distribution detection using cosine similarity," in *ACCV*, 2020.

[62] X. Chen, X. Lan, F. Sun, and N. Zheng, "A boundary based out-of-distribution classifier for generalized zero-shot learning," in *ECCV*, 2020.

[63] Y. Sun, Y. Ming, X. Zhu, and Y. Li, "Out-of-distribution detection with deep nearest neighbors," in *ICML*, 2022.

[64] Z. Wang, Z. Zhang, C. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. G. Dy, and T. Pfister, "Learning to prompt for continual learning," *CVPR*, 2022.

[65] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C.-Y. Lee, X. Ren, G. Su, V. Perot, J. Dy *et al.*, "Dualprompt: Complementary prompting for rehearsal-free continual learning," *ECCV*, 2022.

[66] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *CVPR*, 2019.

[67] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, "Large scale incremental learning," in *CVPR*, 2019.

[68] B. Zhao, X. Xiao, G. Gan, B. Zhang, and S.-T. Xia, "Maintaining discrimination and fairness in class incremental learning," in *CVPR*, 2020.

[69] V. Thengane, S. Khan, M. Hayat, and F. Khan, "Clip model is an efficient continual learner," 2022.

[70] J. S. Smith, L. Karlinsky, V. Gutta, P. Cascante-Bonilla, D. Kim, A. Arbelle, R. Panda, R. Feris, and Z. 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*, 2023, pp. 11 909–11 919.

[71] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, and J. Choi, "Rainbow memory: Continual learning with a memory of diverse samples," in *CVPR*, 2021.

[72] S. Ebrahimi, F. Meier, R. Calandra, T. Darrell, and M. Rohrbach, "Adversarial continual learning," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*. Springer, 2020, pp. 386–402.

[73] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Toronto, ON, Canada*, 2009.

[74] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[75] H. Xiao, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[76] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng *et al.*, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, vol. 2011, no. 2. Granada, 2011, p. 4.

[77] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ICLR*, 2015.

[78] P. Janson, W. Zhang, R. Aljundi, and M. Elhoseiny, "A simple baseline that questions the use of pretrained-models in continual learning," 2023.

[79] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 183–12 192.

[80] H. Liu, L. Gu, Z. Chi, Y. Wang, Y. Yu, J. Chen, and J. Tang, "Few-shot class-incremental learning via entropy-regularized data-free replay," in *European Conference on Computer Vision*. Springer, 2022, pp. 146–162.