Knowledge Restore and Transfer for Multi-Label Class-Incremental Learning

Songlin Dong[1 #], Haoyu Luo[1 #], Yuhang He[1 *], Xing Wei[2], Jie Cheng[3], Yihong Gong[2]
[1]College of Artificial Intelligence, Xi'an Jiaotong University
[2]School of Software Engineering, Xi'an Jiaotong University
[3]ACS Lab, Huawei Technologies, Shenzhen, China
{dsl972731417,luohaoyu,hyh1379478}@stu.xjtu.edu.cn
chengjie8@huawei.com, {weixing,ygong}@mail.xjtu.edu.cn

Abstract

Current class-incremental learning research mainly focuses on single-label classification tasks while multi-label class-incremental learning (MLCIL) with more practical application scenarios is rarely studied. Although there have been many anti-forgetting methods to solve the problem of catastrophic forgetting in single-label class-incremental learning, these methods have difficulty in solving the MLCIL problem due to label absence and information dilution problems. To solve these problems, we propose a Knowledge Restore and Transfer (KRT) framework containing two key components. First, a dynamic pseudo-label (DPL) module is proposed to solve the label absence problem by restoring the knowledge of old classes to the new data. Second, an incremental cross-attention (ICA) module is designed to maintain and transfer the old knowledge to solve the information dilution problem. Comprehensive experimental results on MS-COCO and PASCAL VOC datasets demonstrate the effectiveness of our method for improving recognition performance and mitigating forgetting on multi-label class-incremental learning tasks. The source code is available at https://github.com/witdsl/KRT-MLCIL.

1. Introduction

Class-Incremental Learning (CIL) [5, 12, 17, 38, 49] aims to continuously learn new classes as well as maintain the performance of old classes. When applied to a classification task, most existing CIL methods [52, 13, 5, 56] generally first assume each image only contains a single object, and then develop anti-forgetting mechanisms to learn new classes without forgetting the old ones, *i.e.*, the single-label CIL problem. In real-world applications, however, an image usually contains multiple objects (*e.g.*, a **man** riding his **bicycle**) and the provided labels are often *category-incomplete*

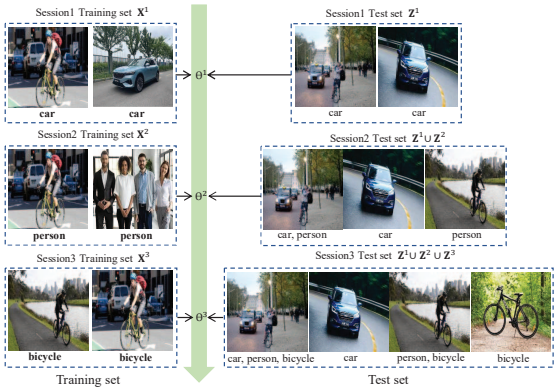


Figure 1: The illustration of the multi-label class-incremental learning task. Supposed there are three categories in total: car, person, and bicycle, which are incrementally learned in three sessions. (Θ^1,Θ^2,Θ^3 are the models that are continuously trained)

due to incrementally defined classes. For example, as shown in Figure 1, only a class label 'car' is provided in session 1. Then, newly defined 'person' and 'bicycle' are annotated on previously and newly collected images in sessions 2 and 3, respectively. The classification models are expected to recognize all the newly and previously encountered categories (*e.g.*, recognize 'car' in session 1 and recognize 'car', 'person' and 'bicycle' in session 3). Taking category-incomplete labels as inputs in different sessions and having the capacity of recognizing all the encountered categories, we term this ability as *multi-label class-incremental learning* (MLCIL). This is a more challenging but practical problem for real-world applications.

A simple solution of the MLCIL problem is to fine-tune a multi-label classification (MLC) model using the training samples of each new session. However, this method leads to catastrophic forgetting [39], where classification accuracy on old classes deteriorate drastically. Another feasible solution

*Yuhang He is the corresponding author; # Songlin Dong and Haoyu Luo are co-first authors

is introducing single-label CIL methods [40, 13, 5, 52] to the MLCIL problem by adopting ML classification head [41]. Most existing CIL methods [40, 13, 5, 51] solve the catastrophic forgetting problem through replaying a portion of representative old exemplars (ER) and designing different knowledge distillation (KD) losses [40, 5, 13] to transfer knowledge from old sessions to new sessions.

However, adapting these methods to the MLCIL problem is faced with two major challenges: 1) the *label absence* of old classes. At each session, images are only annotated with new classes even if they contain old class objects. The absence of old class labels makes these images negative samples of the old classes, thus leading to more serious catastrophic forgetting. For example, in the training session 3 in Figure 1, the right image contains a **person** riding his **bicycle** in front of a **car** is labeled as '**bicycle**'. Training this image with the single label '**bicycle**' makes it a negative sample of **car** and **person**, leading to catastrophic forgetting of these two old classes. (2) The *information dilution* during knowledge transfer. To alleviate forgetting, most existing CIL methods retain an old-sample buffer and transfer the knowledge of old samples to new sessions by knowledge distillation. However, retaining data is often not allowed in practice due to privacy and safety concerns. Even worse, the widely used KD techniques will omit detailed information [61]. This makes the MLC classifier have less ability to recognize the challenging (*e.g.*, small or occluded) objects especially when there are multiple objects and dramatically decreases the multi-label classification performance.

To address the above challenges, in this paper, we propose a *knowledge restore and transfer* (KRT) framework for the MLCIL problem. The KRT framework contains two major modules: 1) a dynamic pseudo-label (DPL) module to restore the knowledge of old classes and 2) an incremental cross-attention (ICA) module to transfer knowledge across different sessions. More specifically, in the DPL module, we feed new data to the old model and generate pseudo labels of the old classes according to dynamic thresholds. The pseudo labels restore the knowledge of old classes and are combined with the current labels (of new classes) to jointly train the new model. In the ICA module, a unified **knowledge transfer** (KT) token for all the sessions and multiple **knowledge retention** (KR) tokens for different sessions, respectively, are designed to maintain and transfer the old knowledge. The KT token aims to learn knowledge-transfer-related information and is continuously trained across all the sessions. The KR token aims to learn category-related knowledge of the current session and is only trained on the current session. By incorporating the KT token with an old session KR token, the ICA outputs a session-specific embedding for the input image, which transfers the knowledge of old session to the current session. During the training and inference process of the t-th session, the ICA outputs a total number of t session-specific embeddings using the current and $t-1$ previously obtained KR tokens, and leverage these embeddings for multi-label classification. Preserving the old session knowledge to the KR token and transferring them to the current session using the KT token, the ICA module can effectively preserve and transfer knowledge for incremental learning and solve the problem of information dilution caused by KD. On this basis, a token loss is designed to optimize the ICA module to transfer knowledge and prevent the forgetting of old knowledge.

For extensive evaluation, we construct the MLCIL baselines by adapting the latest multi-label methods [41, 26, 31] and state-of-the-art CIL methods [30, 45, 40, 13, 43, 59, 51, 5, 56] to this new problem and comparing our KRT with them. We conduct comprehensive experiments on popular MLC datasets, including MS-COCO [33], and PASCAL VOC [16]. To summarize, our main contributions include:

- We propose a knowledge restore and transfer (KRT) framework, which is one of the first attempts, to address the multi-label class-incremental learning (MLCIL) problem.
- We design a dynamic pseudo-label (DPL) module to solve the label absence problem by restoring the knowledge of old classes to the new data.
- We develop an incremental cross-attention (ICA) module with session-specific KR tokens storing knowledge and a unified KT token transferring knowledge to solve the information dilution problem.
- Extensive experiments on MS-COCO and PASCAL VOC demonstrate that the proposed method achieves state-of-the-art performance on the MLCIL task.

2. Related Work

2.1. Single-label Incremental Learning

Regularization-based methods introduce a regularization term in the loss function so that the updated parameter retains old knowledge. 1) Parameter regularization: reduce the variation of parameters related to old tasks [28, 63, 2, 45]. EWC [28] uses a fisher matrix to preserve the important parameters of the historical tasks. Then oEWC [45] and other methods [63, 35, 2] are constantly improving the parameter importance calculation. 2) Data regularization: consolidate the old knowledge by using previous models as soft teachers while learning the new data [30, 11]. For example, LWF [30] exploits knowledge distillation to mitigate forgetting.

Rehearsal-based methods store a set of exemplars as representative of the old data to train with new data from the current task. Most of the rehearsal-based methods are used to solve class-incremental learning [40, 51, 7, 13, 5] problem. Early rehearsal method ER [43] simply constructs a

memory buffer to save samples from old tasks to retrain with new data. On this basis, DER++ [5] proposes knowledge distillation penalties on data stored in the memory buffer. Moreover, the iCaRL [40] and its variants [6, 22, 13, 24] prevent forgetting by selecting exemplars by using the herding [57] technique and designing different distillation losses. BIC [59] and other methods [64, 4] perform an additional bias correction process to modify the classification layer. TPCIL [51] constructs an EHG to model the feature space and propose a topology-preserving loss to maintain the feature space topology. Recent methods [60, 47, 3] propose adaptive aggregation networks or mimic the feature space distribution of oracle to improve the above rehearsal-based methods.

Architectural-based methods provide independent parameters for each task to prevent possible forgetting. Most of the architectural methods require additional task oracle and are restricted to the multi-head setup (Task-IL scenario). Abati et al. [37, 46, 1] propose different strategies to isolate the old and new task parameters and Rusu et al. [44] replicates a new network for each task to transfer prior knowledge through lateral connections to new tasks. The latest architectural methods [61, 14, 54] combined with the rehearsal methods achieve a better anti-forgetting effect. These methods dynamically expand or prune the network parameters to accommodate the new data at the expense of limited scalability. Moreover, L2P [56] exploits dependent prompting methods based on a pre-trained ViT model for continual learning, which achieves state-of-the-art results on multiple single-label incremental learning tasks.

2.2. Multi-label Incremental Learning

Multi-label classification aims to gain a comprehensive understanding of objects and concepts in an image and the proposed methods can be categorized into two main directions: label dependency [19, 55, 8, 29, 34] and loss function [58, 32, 41]. In this paper, we adopt asymmetric loss (ASL) [41] as the classification loss to achieve our KRT and all other compared methods.

Multi-label online incremental learning. Online incremental learning [36] involves organizing tasks into a non-stationary data stream, where the agent can only receive a mini-batch of task samples from the data stream and traverse the data of each task only once. Currently, not only has there been a large amount of works [62, 48, 50, 20, 21] on single-label OIL, but multi-label online incremental learning has also gradually received widespread attention. For example, Du et al. [15] construct the relationship between labels and design a graph convolutional network to learn them. PRS [26] proposes sample-in/sample-out mechanisms to balance the class distribution in memory. Furthermore, OCDM [31] proposes a greedy algorithm to control the class distribution in memory fast and efficiently when the data stream consists of multi-label samples.

3. Method

3.1. Problem Formulation

Assuming that there are a total number of T incremental sessions $\{\mathbf{D}^1, \mathbf{D}^2, ..., \mathbf{D}^T\}$, where $\mathbf{D}^t = \{\mathbf{X}^t, \mathbf{Z}^t\}$ is consisted of a training set $\mathbf{X}^t$ and a test set $\mathbf{Z}^t$. Each training set is defined as $\mathbf{X}^t = \{(\mathbf{x}_i^t, \mathbf{y}_i^t)\}$, where $\mathbf{x}_i^t$ is the i-th training sample and $\mathbf{y}_i^t \subseteq \mathbf{C}^t$ is a label set with $1 \leq |\mathbf{y}_i^t| \leq |\mathbf{C}^t|$. $\mathbf{C}^t$ denotes the class collection at the t-th session and $\forall m, n\ (m \neq n),\ \mathbf{C}^m \cap \mathbf{C}^n = \varnothing$. With the ML-CIL setting, a *unified* **multi-label classification** model will be incrementally trained across the T sessions. At each session t, only $\mathbf{X}^t$ is available during training and the model is evaluated on a combination of test sets $\mathbf{Z}^{1\sim t} = \mathbf{Z}^1 \cup \cdots \cup \mathbf{Z}^t$ and is expected to recognize all the encountered classes $\mathbf{C}^{1\sim t} = \mathbf{C}^1 \cup \cdots \cup \mathbf{C}^t$.

3.2. Framework

Given a multi-label classifier composing of a feature extractor $f(;\theta)$ and a classification head $\varphi(;\phi)$. We use $\Theta = \{\theta, \phi\}$ to denote the total parameters. First, we train a base model Θ^1 using $\mathbf{X}^1$ with the ASL loss [41]. Then, we incrementally fine-tune the base model using $\mathbf{X}^2, ..., \mathbf{X}^T$, and get $\Theta^2, ..., \Theta^T$. At the session t $(t > 1)$, the classification head is expanded for new classes by adding $N^t = |\mathbf{C}^t|$ output neurons.

Figure 2 illustrates the framework of the proposed KRT, which composed of two core designs: DPL and ICA modules. At each session t, we first feed the training set $\mathbf{X}^t$ to the DPL module to generate pseudo labels of the old classes, which combine with the current labels as the new input $\hat{\mathbf{X}}^t$ to jointly train the new model Θ^t. Second, the patch tokens X_P are fed to ICA module to transfer knowledge across different sessions. The x_T (blue) is knowledge transfer (KT) token which is trained across all the sessions. The x_R^t (yellow) and $x_R^{1\sim t-1}$ (green) are knowledge retention (KR) tokens of current and old sessions, respectively. By incorporating the KT token with current and old KR tokens, the ICA output a total number of t session-specific embeddings ($e^{1\sim t}$) for current session multi-label classification. Finally, a token loss $\mathbf{L}_{TL}$ is designed to jointly optimize the ICA module. The following part of this section provides detailed descriptions of these two components.

3.3. Dynamic Pseudo-Label Module

To prevent the catastrophic forgetting caused by the *label absence* problem, we propose a dynamic pseudo-label (DPL) module. Concretely, in the session t, given an input image $\mathbf{x}_i^t$ and a previous model Θ^{t-1}, where Θ^{t-1} has already learned the knowledge of K classes. We utilize the model Θ^{t-1} to perform an inference on $\mathbf{x}_i^t$ and get the classification probabilities $\mathbf{p_i} = \{p_1, p_2, ..., p_K\}$ of K classes, where $p_k \in (0, 1)$ denotes the probability of class k. If $p_k \geq \eta$, the image is very likely to contain the k-th category object, and $p_k < \eta$

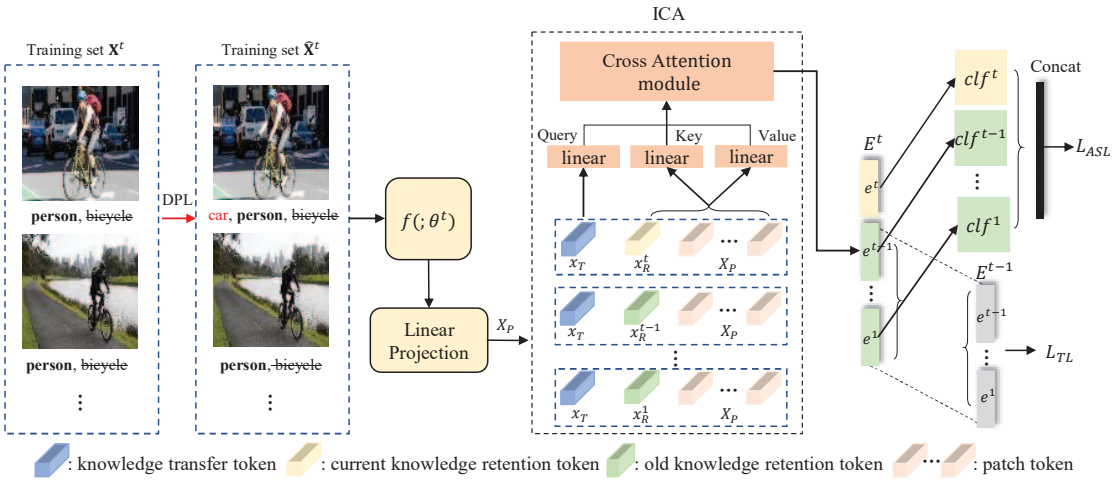


Figure 2: The framework of our proposed KRT for MLCIL problem. The image is first to restore old knowledge by dynamic pseudo-label (DPL) module and obtains the pseudo label 'car' ('~~bicycle~~' indicates that the class has not been defined yet). Then, we feed the restored image into the feature extractor $f(;\theta^t)$ and the linear projection to obtain patch token X_P. Finally we fed the X_P into incremental cross-attention (ICA) module to acquire final output logits (E^{t-1} and E^t are outputs of previous and current ICA module).

denotes the opposite. Here, $\eta \in (0, 1)$ is the initial threshold. Finally, we compose pseudo-label set $\mathbf{S}^t$ of all pseudo labels generated by training samples in the current session.

In general, we employ model Θ^{t-1} to infer the training set $\mathbf{X}^t$ in this session and merge the obtained old pseudo-label set $\mathbf{S}^t$ with the label set $\mathbf{Y}^t$ of this session as new ground truth $\hat{\mathbf{Y}}^t$. Then we use the updated training set $\hat{\mathbf{X}}^t$ to train the model Θ^t. However, as the number of learning sessions increases, the abundance of pseudo labels can impede the ability to acquire knowledge about new classes. Additionally, the model inevitably forgets the knowledge of old classes, particularly those learned early on, resulting in the generation of inaccurate pseudo labels. To address these issues, we propose a straightforward yet efficient method, called dynamic threshold adjustment, which reduces the incidence of falsely generated pseudo labels during the incremental process. Specifically, before each incremental session begins, we dynamically adjust the threshold η^t based on β^t and μ^t. The $\beta^t = \frac{|\mathbf{S}^t|}{M^t}$ is the number of average pseudo labels per image at the current session, where $|\mathbf{S}^t|$ and M^t are the number of generated pseudo labels and training samples at the current session, respectively. The $\mu^t = (\frac{|\mathbf{C}_o^t|}{|\mathbf{C}_a|} \times \mu)$ is the target value, where $|\mathbf{C}_o^t|$ is the number of old classes that have been learned, $|\mathbf{C}_a|$ is the total number of classes in the dataset and the μ is a hyper-parameter. The detailed of DPL algorithm is written in **Appendix**. The ablation study 4.4 have proven that the DPL module adapts well to the multi-label incremental learning task, effectively restores old knowledge, and solves the catastrophic forgetting problem.

3.4. Incremental Cross-Attention Module

3.4.1 Build the ICA Module

In this section, we introduce the construction of the incremental cross-attention module. Concretely, for each input image, the output of the feature extractor $f(;\theta)$ is $\mathcal{F} \in \mathbb{R}^{h\times w\times c}$, where h, w denote the height and width of feature map respectively, and c represents the dimension. After that, we add a linear projection layer to project the features from dimension c to d to match the incremental cross-attention module and reshape the projected features to be patch token $X_P \in \mathbb{R}^{L\times d}$, where $L = hw$. In order to learn new classes while preserve the model performance on the old classes, two learnable tokens are concatenated with the sequence of patch token X_P including a knowledge transfer (KT) token $x_T \in \mathbb{R}^d$ and a knowledge retention (KR) token $x_R \in \mathbb{R}^d$. Making the X_P, x_T and x_R as the input of the cross-attention module:

$$
\begin{aligned}
Q &= W_q[x_T], \\
K &= W_k[x_R, X_P], \\
V &= W_v[x_R, X_P], \\
z &= W_o \text{softmax}\left(\frac{QK^T}{\sqrt{l/h}}\right) V + b_o, \qquad (1)
\end{aligned}
$$

where l is the embedding dimension, and h is the number of attention heads. Our incremental cross-attention (ICA) module defines the KT token (x_T) as query(Q). And the concatenation of KR token x_R and patch token X_P (i.e.

$[x_R, X_P]$) as key(K) and value(V). These tokens are fed into the cross-attention (CA) module:

$$e_1 = x_T + \mathrm{CA}(\mathrm{Norm}(x_T, x_R, X_P)), \quad (2)$$

$$e = e_1 + \mathrm{MLP}(\mathrm{Norm}(e_1)), \quad (3)$$

where CA($\cdot$) and Norm($\cdot$) denote the cross-attention and layer normalization in [53], respectively, and MLP is a multi-layer perception with a single hidden layer. The output embedding $e \in \mathbb{R}^d$ keeps the same dimension as Q (i.e. x_T). Then we feed the embedding e into a classification head φ and obtain the output logits $o \in \mathbb{R}^N$.

3.4.2 ICA Based MLCIL

In the first session, we add the unified KT token x_T which is continuously trained across all the sessions and the KR token x_R^1 which is only trained on current session. At the session t, we expand our ICA module by creating a new KR token x_R^t while keeping the old KR tokens $x_R^{1\sim t-1}$. Therefore, we have one unified KT x_T and t KR tokens $x_R^{1\sim t}$ (The old KR tokens $x_R^{1\sim t-1}$ preserve the knowledge of old classes of the corresponding sessions and are frozen at the current session t). For each input image, we feed it into the feature extractor $f(;\theta^t)$ and linear projection layer to acquire the patch token X_P. By incorporating the KT token with an old session KR token, the ICA outputs a session-specific embedding for X_P, which transfers the knowledge of old session to the current session. In order to acquire all the potential object categories of the image, the ICA outputs a total number of t session-specific embeddings $\{e^1, ..., e^t\}$ using the current and $t-1$ previous KR tokens, and leverage these embeddings for current session multi-label classification.

Finally, each embedding $e^{1\sim t}$ is fed to the corresponding classification heads $\varphi^{1\sim t}$ with parameters $\phi^{1\sim t}$ to obtain the output logits $o^{1\sim t}$:

$$\begin{aligned} o^1 &= \varphi^1(\mathrm{ICA}((x_T, x_R^1, X_P); \phi^1)), \\ o^2 &= \varphi^2(\mathrm{ICA}((x_T, x_R^2, X_P); \phi^2)), \\ &\cdots \\ o^t &= \varphi^t(\mathrm{ICA}((x_T, x_R^t, X_P); \phi^t)), \end{aligned} \quad (4)$$

where $o^t \in R^{N^t}$. Then we concatenate all output logits as $O^t = [o^1, o^2, ..., o^t]$ to compute the classification loss $\mathbf{L}_{ASL}$. On this basis, to balance stability and plasticity, we concatenate these session-specific embeddings as the output of the ICA module, denoted as $E^t = [e^1, ..., e^t]$, to compute the token loss $\mathbf{L}_{TL}$ (see in 3.5).

3.5. Loss Function

Our model is trained on two losses: (1) the classification loss $\mathbf{L}_{ASL}$: asymmetric loss [41], and (2) the token loss $\mathbf{L}_{TL}$ applied on the ICA module. In summary, the total loss in the incremental learning (IL) sessions is:

$$\mathbf{L}_{IL} = \mathbf{L}_{ASL} + \lambda \mathbf{L}_{TL}, \quad (5)$$

where λ is hyper-parameter.

Asymmetric loss: We adopt an asymmetric loss [41] for classification. We can predict category probabilities of each image $\mathbf{p} = [p_1, ..., p_N] \in \mathbb{R}^N$:

$$L_{ASL} = \frac{1}{N} \sum_{n=1}^{N} \begin{cases} (1-p_n)^{\gamma+} log(p_n), & y_n = 1, \\ p_n^{\gamma-} log(1-p_n), & y_n = 0, \end{cases} \quad (6)$$

where $\mathbf{y}_n$ is the binary label to indicate if image has label n. $\gamma+$ and $\gamma-$ are the positive and negative focusing parameters, respectively.

Token loss: To optimize the ICA module and prevent the forgetting of old knowledge, we propose a token loss to penalize the changes of old session-specific embeddings. The $\mathbf{L}_{TL}$ can be written as:

$$\mathbf{L}_{TL} = 1 - < E^{t-1}, E^t[: e^{t-1}] >, \quad (7)$$

where E^{t-1} and E^t are the previous and current output of the ICA module, and $<, >$ represents cosine similarity.

4. Experiment

4.1. Datasets and Experimental Details

Datasets and Benchmark. We use MS-COCO 2014 [33] and PASCAL VOC 2007 [16] datasets to evaluate the effectiveness of our method in MLCIL task. MS-COCO is a widely-used, large-scale dataset for evaluating multi-label classification. It comprises $122,218$ images and covers 80 object classes. The training set contains 80K images, the validation set contains 40K images, and on average, each image has 2.9 labels. PASCAL VOC dataset consists of $9,963$ images across 20 object classes with 5K images for training and 5K images for testing. The average number of labels per image is 2.4.

Followed by CIL works [13, 5], we evaluate our methods on MS-COCO dataset with two protocols including 1)*COCO-B0*: we train all 80 classes in several splits including 4 and 8 incremental sessions. 2)*COCO-B40*: we first train a base model on 40 classes and the remaining 40 classes are divided into splits of 4 and 8 sessions. In addition, we evaluate our methods on VOC with two protocols that are 1)*VOC-B0*: this trains the model in batches of 4 classes from scratch. 2)*VOC-B10*: this starts from a model trained on 10 classes, and the remaining 10 classes come in 5 sessions. Inspired by the IOD task [49, 17], the order of incremental learning is the lexicographical order of category names.

Evaluation Metrics. For settings with MLCIL task, we adopt two metrics, average accuracy and last accuracy, which are widely used in CIL works [13, 5]. Following the MLC works [41, 34], we adopt the mean average precision (mAP) to evaluate all the categories that have been learned in each session and report the average mAP (the average of the mAP of all sessions) and the last mAP (final session mAP). To

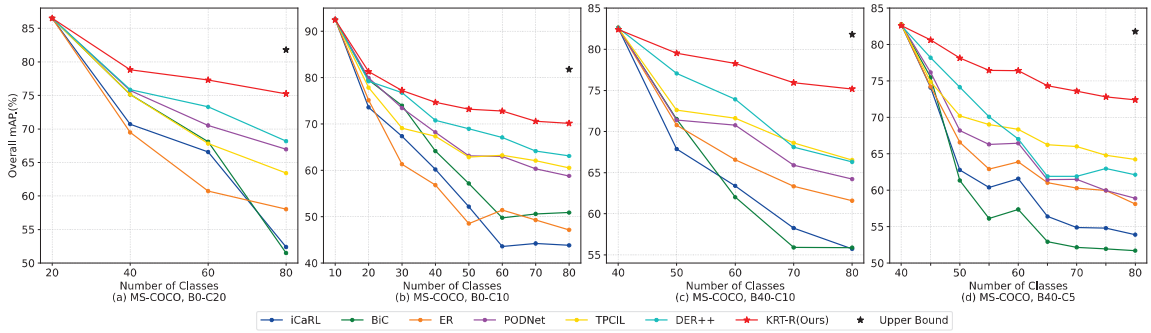


Figure 3: Comparison results (mAP%) on MS-COCO dataset under different protocols against rehearsal-based methods.

provide a more comprehensive evaluation of the performance after training on all incremental tasks, we also report the per-class F1 measure (CF1) and overall F1-measure (OF1) alongside the last accuracy.

Implementation Details. Followed by ASL [41], we adopt ImageNet-21k pre-trained TResNetM [42] as our backbone (All compared methods also use ImageNet-21k pre-trained TResNetM or ViT-B/16 as the backbone). We train the model for 20 epochs (4 warm-up epochs) using Adam [27] optimizer and OneCycleLR scheduler with a weight decay of 1e-4. The batch size is set to 64. To train the base model, we set the learning rate to 4e-5. During the incremental session, we set the learning rate to 1e-4 for COCO and 4e-5 for VOC. The data augmentation techniques include rand augmentation [9] as well as cutout [10]. Furthermore, we conduct experiments three times and reported the average results. More implementation details are provided in the **Appendix**.

4.2. Comparison Methods

For comparative experiments, we run several baselines and state-of-the-art single-label continual learning methods in our MLCIL setting. We select widely recognized and best-performing methods based on several recent CIL works [5, 13, 56]. To provide a comprehensive analysis, we also include latest state-of-the-art multi-label online incremental learning methods [26, 31]. In addition, we use $\mathbf{L}_{ASL}$ instead of cross-entropy loss as classification loss and rely on the original code base for implementation and hyper-parameter selection to ensure optimal performance.[2]

Baseline Methods. FT method fine-tunes the model without any anti-forgetting constraints. Upper-bound is the supervised training on the data of all tasks, which is usually regarded as the upper-bound performance a IL method can achieve.

Class-incremental Methods. We select nine representative CIL works to our MLCIL setting, including oEWC [45], LWF [30], iCARL [40], BiC [59], ER [43], TPCIL [51], Der++ [5], PODNet [13], and L2P [56]. oEWC [45] and LwF [30] are representative regularization-based works. ICARL, BiC and ER are classical rehearsal-based methods. Der++, PODNet, and TPCIL are best-performing rehearsal-based methods. L2P [56] is the latest SOTA CIL method based on ViT-B/16, we compare the relative performance to the corresponding upper-bound performance for fairness.

ML Online-incremental Methods. We select two latest SOTA ML online incremental learning methods in our ML-CIL setting to compare, including PRS [26] and OCDM [31].

Our Methods. KRT is our proposed method without rehearsal buffer. KRT-R is KRT equipped with a rehearsal buffer for a fair comparison with SOTA methods.

4.3. Comparison Results

Results on MS-COCO. Table 1 shows the results on MS-COCO B0-C10 and B40-C10 benchmarks. KRT outperforms all comparing methods consistently, in terms of both average accuracy (mAP) and last accuracy (CF1, OF1 and mAP). Specifically, when the buffer size is large (20/class), our method achieves the best final accuracy of **70.2%** and **75.2%** on two benchmarks, which outperforms the latest SOTA rehearsal-based methods by **7.1%** and **8.7%**, respectively. When the buffer size gets smaller (5/class), KRT-R achieves even greater performance gains compared to other continual learning methods. It is worth noting that when the buffer size is set to 0, rehearsal-based CIL methods become ineffective. For example, the final mAP of the PODNet dropped sharply by at least **33.2%**. However, KRT still maintains superior performance by outperforming the regularization-based methods and other rehearsal-based methods even they have large rehearsal buffer.

Figure 3 shows the comparison curves on four challenge benchmarks with larger buffer size. It is observed that KRT-R(Ours) consistently outperforms all other CIL methods at every session regardless of the incremental settings and is the closest to the Upper Bound. As the number of sessions

[2]Task-incremental learning [37, 46, 25, 23] and singe-label online incremental learning [62, 48, 50, 21, 18] methods are not included as they are not applicable to different class-incremental setting.

Method	Source Task	Buffer size	MS-COCO B0-C10				MS-COCO B40-C10			
			Avg. Acc	Last Acc			Avg. Acc	Last Acc		
			mAP (%)	CF1	OF1	mAP (%)	mAP (%)	CF1	OF1	mAP (%)
Upper-bound	Baseline	-	-	76.4	79.4	81.8	-	76.4	79.4	81.8
FT [41]	Baseline	0	38.3	6.1	13.4	16.9 (↓ 49.0)	35.1	6.0	13.6	17.0 (↓ 57.0)
PODNet [13]	CIL		43.7	7.2	14.1	25.6 (↓ 40.3)	44.3	6.8	13.9	24.7 (↓ 49.3)
oEWC [45]	CIL		46.9	6.7	13.4	24.3 (↓ 41.6)	44.8	11.1	16.5	27.3 (↓ 46.7)
LWF [30]	CIL		47.9	9.0	15.1	28.9 (↓ 37.0)	48.6	9.5	15.8	29.9 (↓ 44.1)
KRT(Ours)	MLCIL		**74.6**	**55.6**	**56.5**	**65.9** (↓ 0.0)	**77.8**	**64.4**	**63.4**	**74.0** (↓ 0.0)
TPCIL [51]	CIL	5/class	63.8	20.1	21.6	50.8 (↓ 17.5)	63.1	25.3	25.1	53.1 (↓ 21.2)
PODNet [13]	CIL		65.7	13.6	17.3	53.4 (↓ 14.9)	65.4	24.2	23.4	57.8 (↓ 16.5)
DER++ [5]	CIL		68.1	33.3	36.7	54.6 (↓ 13.7)	69.6	41.9	43.7	59.0 (↓ 15.3)
KRT-R(Ours)	MLCIL		**75.8**	**60.0**	**61.0**	**68.3** (↓ 0.0)	**78.0**	**66.0**	**65.9**	**74.3** (↓ 0.0)
iCaRL [40]	CIL	20/class	59.7	19.3	22.8	43.8 (↓ 26.4)	65.6	22.1	25.5	55.7 (↓ 19.5)
BiC [59]	CIL		65.0	31.0	38.1	51.1 (↓ 19.1)	65.5	38.1	40.7	55.9 (↓ 19.3)
ER [43]	CIL		60.3	40.6	43.6	47.2 (↓ 23.0)	68.9	58.6	61.1	61.6 (↓ 13.6)
TPCIL [51]	CIL		69.4	51.7	52.8	60.6 (↓ 9.6)	72.4	60.4	62.6	66.5 (↓ 8.7)
PODNet [13]	CIL		70.0	45.2	48.7	58.8 (↓ 11.4)	71.0	46.6	42.1	64.2 (↓ 11.0)
DER++ [5]	CIL		72.7	45.2	48.7	63.1 (↓ 7.1)	73.6	51.5	53.5	66.3 (↓ 8.9)
KRT-R(Ours)	MLCIL		**76.5**	**63.9**	**64.7**	**70.2** (↓ 0.0)	**78.3**	**67.9**	**68.9**	**75.2** (↓ 0.0)
PRS [26]	MLOIL	1000	48.8	8.5	14.7	27.9 (↓ 41.4)	50.8	9.3	15.1	33.2 (↓ 41.9)
OCDM [31]	MLOIL		49.5	8.6	14.9	28.5 (↓ 40.8)	51.3	9.5	15.5	34.0 (↓ 41.1)
KRT-R(Ours)	MLCIL		**75.7**	**61.6**	**63.6**	**69.3** (↓ 0.0)	**78.3**	**67.5**	**68.5**	**75.1** (↓ 0.0)

Table 1: Class-incremental results on MS-COCO dataset. Compared methods are grouped based on different source tasks. Buffer size 0 means no rehearsal is required, where most SOTA CIL methods are not applicable anymore.

Method	Backbone	Param.	Avg. mAP%	Last mAP%
Upper-bound	ViT-B/16	86.0M	-	83.16
L2P [56]			73.07	70.42 (▽ 12.74)
L2P-R [56]			73.64	71.68 (▽ 11.48)
Upper-bound	TResNetM	29.4M	-	81.80
KRT(Ours)			**77.83**	**74.02** (▽ **7.78**)
KRT-R(Ours)			**78.34**	**75.18** (▽ **6.62**)

Table 2: Class-Incremental results on MS-COCO dataset under the B40-C10 setting against prompt-based CIL method. ▽ indicates the gap towards the Upper Bound of corresponding backbone.

Method	Buffer Size	VOC B0-C4		VOC B10-C2	
		Avg. Acc	Last Acc	Avg. Acc	Last Acc
Upper bound	-	-	93.6	-	93.6
FT [41]		82.1	62.9	70.1	43.0
iCarL [40]	2/class	87.2	72.4 (↓ 11.0)	79.0	66.7 (↓ 13.8)
BIC [59]		86.8	72.2 (↓ 11.2)	81.7	69.7 (↓ 10.8)
ER [43]		86.1	71.5 (↓ 11.9)	81.5	68.6 (↓ 11.9)
TPCIL [51]		87.6	77.3 (↓ 6.1)	80.7	70.8 (↓ 9.7)
PODNet [13]		88.1	76.6 (↓ 6.8)	81.2	71.4 (↓ 9.1)
DER++ [5]		87.9	76.1 (↓ 7.3)	82.3	70.6 (↓ 9.9)
KRT-R(Ours)		**90.7**	**83.4** (↓ 0.0)	**87.7**	**80.5**(↓ 0.0)

Table 3: Comparison results on PASCAL VOC dataset. All metric are in mAP%

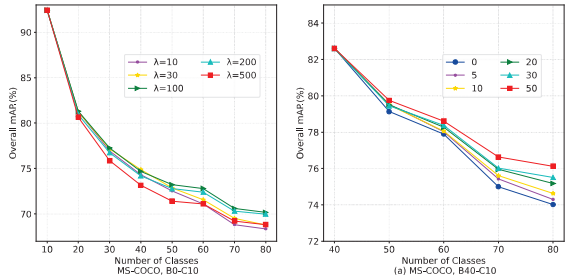
increases, we observe a widening gap between the performance of the KRT method and other methods. This suggests that our method is better suited for long-term incremental learning scenarios.

Table 1 also presents a comparison between KRT-R and MLOIL methods. We observe that the online learning methods do not perform well on the MLCIL task. Our KRT outperforms both PRS and OCDM by a large margin.

Table 2 shows the comparison between KRT and prompt-based methods. Since the L2P method is based on the pre-trained ViT, we use the towards to the upper bound (▽) to measure the performance of each method given a specific backbone. We can observe that KRT relatively outperforms L2P by at least **4.86%** with or without rehearsal buffer.

Results on PASCAL VOC. Table 3 summarizes the experimental results on PASCAL VOC dataset. We observe a similar conclusion to those on MS-COCO dataset. Concretely, KRT consistently surpasses other methods by a considerable margin on two benchmarks. In the comparison results with the incremental data split into 5 sessions, KRT achieves the best last mAP value of **83.4**% and outperforms the other methods by **6.1%(77.3%→83.4%)**. Moreover, on the B10-C2 benchmark, our method outperforms second best method from **71.4%** to **80.5%(9.1%)** at the last session.

The outstanding performance of KRT over all compared methods on two MLC datasets indicates that the effectiveness of our methods for improving recognition performance and mitigating forgetting for MLCIL task even without a rehearsal buffer.



(a) (b)

Figure 4: **Robustness Testing.** (a) Sensitive study of hyper-parameter λ. (b) The influence of buffer size.

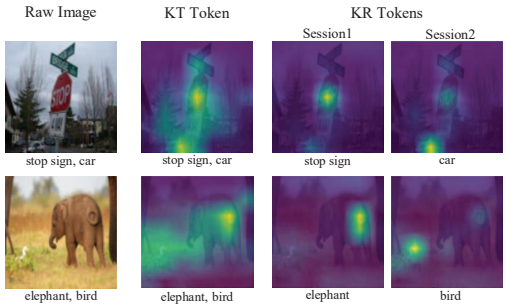


Figure 5: Visualization of ICA module.

4.4. Ablation Study

The Effectiveness of Each Component. Table 4 demonstrates the results of our ablative experiments on COCO B40-C10 setting with large buffer size. We use a distillation loss $\mathbf{L}_{KD}$ [22] applied on the globally pooled feature as the **baseline** method and generate three additional variants of KRT. (a) KRT w/o DPL: We optimize using only the ICA module. (b) KRT w/o ICA: We optimize using only the DPL module. (c) KRT w/ KD: we add extra the loss $\mathbf{L}_{KD}$ to our KRT method.

Model	KD	ICA	DPL	Avg. Acc	Last Acc
Baseline	✓			65.93	58.02 (↑ 0.0)
(a) w/o DPL		✓		77.06	71.97 (↑ 13.95)
(b) w/o ICA			✓	77.14	73.12 (↑ 15.10)
(c) w/ KD	✓	✓	✓	78.14	74.77 (↑ 16.75)
KRT		✓	✓	**78.34**	**75.18** (↑ 17.16)

Table 4: The contribution of each component.

As shown in Table 4, the baseline model produces the lowest last mAP of **58.02%**. Using ICA or DPL modules separately both bring a significant improvement (rows(a,b)). Only using ICA module (row a) improves the last mAP by **13.95%** and with DPL module used separately (row b), we observe a **15.10%** relative improvement. Applying the KD loss degrades the performance (row c). Though it is popularly used by CIL methods [22, 13], it may be not so effective for MLCIL. These results strongly prove that the ICA and DPL module are very effective to prevent forgetting and improve performance for the MLCIL tasks.

Sensitive Study of Hyper-parameter λ. To verify the robustness of KRT , we conduct experiments on MS-C0C0 B0-C10 with different hyper-parameters λ. More specifically, we test $\lambda = 10, 30, 100, 200, 500$ respectively. The comparison results are shown in Figure 4a. We can see that our KRT get best performance when $\lambda = 100$ and the performance changes are minimal under different λ.

The Influence of Buffer Size. We gradually increase the buffer size from 0 per class to 50 per class and report the performance of the our KRT on MS-COCO B40-C10 in Figure 4b. The final mAP only increases from **74.02%** to **76.13%** as the buffer size change from 0 to 50. Form the results. we can see that our KRT is more effective and robust and it can overcome forgetting even without buffer.

Visualization of ICA Module. To further demonstrate the effectiveness of ICA module, we illustrate several attention map examples of the KR and KT tokens in Figure 5. The KR tokens of different sessions only maintains the category knowledge of the current session, and the continuously trained KT token learns the knowledge of all sessions.

More detailed experimental results and more visualization images are provided in the **Appendix**.

5. Conclusion

In this paper, we focus on a challenging but more practical problem named multi-label class incremental learning (MLCIL). Compared to the vanilla CIL problem, MLCIL are faced with two major challenges: the *label absence* of old classes and the *information dilution* during knowledge transfer. To solve these challenges, we propose a knowledge restore and transfer (KRT) framework containing two key components, *i.e.*, a dynamic pseudo-label (DPL) module to solve the label absence problem by restoring the knowledge of old classes to the new data and an incremental cross-attention (ICA) module with session-specific KR tokens storing knowledge and a unified KT token transferring knowledge to solve the information dilution problem. Extensive experimental results on MS-COCO and PASCAL VOC datasets show that our method significantly outperforms existing state-of-the-art methods and demonstrate the superiority of the proposed method.

Acknowledgments

This work was funded by the National Key Research and Development Project of China under Grant No. 2020AAA0105600, and by the National Natural Science Foundation of China under Grant No. U21B2048 and No. 62006183. Thanks to Huawei's support.

References

[1] Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *CVPR*, pages 3931–3940, 2020.

[2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and et al. Memory aware synapses: Learning what (not) to forget. In *ECCV*, pages 139–154, 2018.

[3] Arjun Ashok, KJ Joseph, and et al. Class-incremental learning with cross-space clustering and controlled transfer. In *ECCV*, pages 105–122. Springer, 2022.

[4] Eden Belouadah and et al. Il2m: Class incremental learning with dual memory. In *ICCV*, pages 583–592, 2019.

[5] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and et al. Dark experience for general continual learning: a strong, simple baseline. *NIPS*, 33:15920–15930, 2020.

[6] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, pages 233–248, 2018.

[7] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *ICCV*, pages 9516–9525, 2021.

[8] Zhaomin Chen, Xiu-Shen Wei, Peng Wang, and Yanwen Guo. Learning graph convolutional networks for multi-label recognition and applications. *In TPAMI*, 2021.

[9] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR Workshops*, pages 702–703, 2020.

[10] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[11] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *CVPR*, pages 5138–5146, 2019.

[12] Songlin Dong, Xiaopeng Hong, Xiaoyu Tao, Xinyuan Chang, Xing Wei, and Yihong Gong. Few-shot class-incremental learning via relation knowledge distillation. In *AAAI*, volume 35, pages 1255–1263, 2021.

[13] Arthur Douillard, Matthieu Cord, and et al. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pages 86–102. Springer, 2020.

[14] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *CVPR*, pages 9285–9295, 2022.

[15] Kaile Du, Fan Lyu, Fuyuan Hu, Linyan Li, Wei Feng, Fenglei Xu, and Qiming Fu. Agcn: augmented graph convolutional network for lifelong multi-label image recognition. In *ICME*, pages 01–06. IEEE, 2022.

[16] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[17] Tao Feng, Mang Wang, and et al. Overcoming catastrophic forgetting in incremental object detection via elastic response distillation. In *CVPR*, pages 9427–9436, 2022.

[18] Enrico Fini, Stéphane Lathuiliere, Enver Sangineto, Moin Nabi, and Elisa Ricci. Online continual learning under extreme memory constraints. In *ECCV*, pages 720–735. Springer, 2020.

[19] Yunchao Gong, Yangqing Jia, Thomas Leung, Alexander Toshev, and Sergey Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013.

[20] Qiushan Guo, Xinjiang Wang, Yichao Wu, Zhipeng Yu, Ding Liang, Xiaolin Hu, and Ping Luo. Online knowledge distillation via collaborative learning. In *CVPR*, pages 11020–11029, 2020.

[21] Ya-nan Han and Jian-wei Liu. Online continual learning via the knowledge invariant and spread-out properties. *Expert Systems with Applications*, 213:119004, 2023.

[22] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pages 831–839, 2019.

[23] Jian Jiang and Oya Celiktutan. Neural weight search for scalable task incremental learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1390–1399, 2023.

[24] Minsoo Kang, Jaeyoo Park, and Bohyung Han. Class-incremental learning by knowledge distillation with adaptive feature consolidation. In *CVPR*, pages 16071–16080, 2022.

[25] Zixuan Ke, Bing Liu, and Xingchang Huang. Continual learning of a mixed sequence of similar and dissimilar tasks. *NIPS*, 33:18493–18504, 2020.

[26] Chris Dongjoo Kim and et al. Imbalanced continual learning with partitioning reservoir sampling. In *ECCV*, 2020.

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] 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. *National Academy of Sciences*, 114(13):3521–3526, 2017.

[29] Jack Lanchantin, Tianlu Wang, Vicente Ordonez, and Yanjun Qi. General multi-label image classification with transformers. In *CVPR*, pages 16478–16488, 2021.

[30] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[31] Yan-Shuo Liang and Wu-Jun Li. Optimizing class distribution in memory for multi-label online continual learning. *arXiv preprint arXiv:2209.11469*, 2022.

[32] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017.

[33] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[34] Shilong Liu, Lei Zhang, Xiao Yang, Hang Su, and Jun Zhu. Query2label: A simple transformer way to multi-label classification. *arXiv preprint arXiv:2107.10834*, 2021.

[35] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *International Conference on Pattern Recognition*, pages 2262–2268. IEEE, 2018.

[36] David Lopez-Paz et al. Gradient episodic memory for continual learning. In *NIPS*, pages 6467–6476, 2017.

[37] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *CVPR*, pages 7765–7773, 2018.

[38] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. Recall: Replay-based continual learning in semantic segmentation. In *ICCV*, pages 7026–7035, 2021.

[39] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pages 109–165. Elsevier, 1989.

[40] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017.

[41] Tal Ridnik, Emanuel Ben-Baruch, and et al. Asymmetric loss for multi-label classification. In *ICCV*, pages 82–91, 2021.

[42] Tal Ridnik, Hussam Lawen, Asaf Noy, Emanuel Ben Baruch, Gilad Sharir, and Itamar Friedman. Tresnet: High performance gpu-dedicated architecture. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1400–1409, 2021.

[43] Matthew Riemer, Ignacio Cases, and et al. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.

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

[45] Jonathan Schwarz, Wojciech Czarnecki, and et al. Progress & compress: A scalable framework for continual learning. In *ICML*, pages 4528–4537. PMLR, 2018.

[46] Joan Serra, Didac Suris, Marius Miron, and et al. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, pages 4548–4557. PMLR, 2018.

[47] Yujun Shi, Kuangqi Zhou, Jian Liang, Zihang Jiang, Jiashi Feng, Philip HS Torr, Song Bai, and Vincent YF Tan. Mimicking the oracle: An initial phase decorrelation approach for class incremental learning. In *CVPR*, pages 16722–16731, 2022.

[48] Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class-incremental continual learning with adversarial shapley value. In *AAAI*, volume 35, pages 9630–9638, 2021.

[49] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. Incremental learning of object detectors without catastrophic forgetting. In *ICCV*, pages 3400–3409, 2017.

[50] Shengyang Sun, Daniele Calandriello, Huiyi Hu, Ang Li, and Michalis Titsias. Information-theoretic online memory selection for continual learning. *arXiv preprint arXiv:2204.04763*, 2022.

[51] Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *ECCV*, pages 254–270. Springer, 2020.

[52] Xiaoyu Tao, Xiaopeng Hong, Xinyuan Chang, Songlin Dong, Xing Wei, and Yihong Gong. Few-shot class-incremental learning. In *CVPR*, pages 12183–12192, 2020.

[53] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, pages 32–42, 2021.

[54] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *ECCV*, pages 398–414. Springer, 2022.

[55] Zhouxia Wang, Tianshui Chen, Guanbin Li, Ruijia Xu, and et al. Multi-label image recognition by recurrently discovering attentional regions. In *ICCV*, pages 464–472, 2017.

[56] Zifeng Wang, Zizhao Zhang, and et al. Learning to prompt for continual learning. In *CVPR*, pages 139–149, 2022.

[57] Max Welling. Herding dynamical weights to learn. In *ICML*, pages 1121–1128, 2009.

[58] Tong Wu, Qingqiu Huang, Ziwei Liu, Yu Wang, and Dahua Lin. Distribution-balanced loss for multi-label classification in long-tailed datasets. In *ECCV*, pages 162–178. Springer, 2020.

[59] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, pages 374–382, 2019.

[60] Haofei Xu and Juyong Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *CVPR*, pages 1959–1968, 2020.

[61] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *CVPR*, pages 3014–3023, 2021.

[62] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. *arXiv preprint arXiv:2106.01085*, 2021.

[63] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, pages 3987–3995. JMLR. org, 2017.

[64] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *CVPR*, pages 13208–13217, 2020.