Applied Intelligence
https://doi.org/10.1007/s10489-022-03509-0

RT-Net: replay-and-transfer network for class incremental object detection

Bo Cui[1,2] · Guyue Hu[1,2,3] · Shan Yu[1,4,5]

Accepted: 12 March 2022

Abstract

Despite the remarkable performance achieved by DNN-based object detectors, class incremental object detection (CIOD) remains a challenge, in which the network has to learn to detect novel classes sequentially. Catastrophic forgetting is the main problem underlying this difficulty, as neural networks tend to detect new classes only when training samples for old classes are absent. In this paper, we propose the Replay-and-Transfer Network (RT-Net) to address this issue and accomplish CIOD. We develop a generative replay model to replay features of old classes during learning of new ones for the RoI (Region of Interest) head, using the stored latent feature distributions. To overcome the drastic changes of the RoI feature space, guided feature distillation and feature translation are introduced to facilitate knowledge transfer from the old model to the new one. In addition, we propose holistic ranking transfer, which transfers ranking orders of proposals to the new model, to enable the region proposal network to identify high quality proposals for old classes. Importantly, this framework provides a general solution for CIOD, which can be successfully applied to two task settings: set-overlapped, in which the old and new training sets are overlapped, and set-disjoint, in which the old and new tasks have unique samples. Extensive experiments on standard benchmark datasets including PASCAL VOC and COCO show that RT-Net can achieve state-of-the-art performance for CIOD.

Keywords Incremental learning · Object detection · Generative replay · Knowledge transfer

1 Introduction

Object detection is a fundamental and challenging problem in computer vision. Various detection methods [9, 10, 14, 16, 25, 28, 29, 36, 37] have been proposed based on deep neural networks [24] and have broad potential applications [6, 17, 42, 43, 50]. Despite the marked improvement in accuracy with datasets such as PASCAL VOC [8] and COCO [27], most of these models can only detect classes that are fully supervised during training. A challenge for real world applications is learning object detectors incrementally, where new classes are added in multiple training stages, while samples or labels for old classes are missing. Catastrophic forgetting [30] is a critical problem with class incremental learning that results in performance degradation on old classes as new classes are incrementally added. Past studies of incremental learning can be roughly divided into three primary families: regularization-based [21, 49], distillation-based [3, 26, 35, 46] and replay-based methods [20, 40]. However, there are relatively few methods for incremental object detection.

Class incremental learning is one of the most important parts of continual learning/lifelong learning [1, 7, 31]. Intelligent agents shall be designed to learn continuously (i.e., learn consecutive tasks without performance degeneration on previously learned tasks) to achieve general artificial intelligence. Therefore, algorithms developed for CIOD

✉ Bo Cui
bo.cui@nlpr.ia.ac.cn

[1] Brainnetome Center and National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China

[3] Present address: School of Computing, National University of Singapore, Singapore, 117417, Singapore

[4] CAS Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Beijing 100190, China

[5] School of Future Technology, University of Chinese Academy of Sciences, Beijing 100049, China

Published online: 03 August 2022

must localize and classify instances of classes sequentially exposed to the model. Specifically, we let $\mathcal{C}$ denote the set of classes that are incrementally introduced to the object detector and assume that there are a total of s sequential stages in the learning process. In stage t, the newly added classes $\mathcal{C}_t$ are a subset of $\mathcal{C}$: $\mathcal{C}_t \subset \mathcal{C}$ (such that $\mathcal{C}_i \cap \mathcal{C}_j = \emptyset$, for any $i, j \leq s$). We let $\mathcal{X}_t = \{\mathbf{x}_i\}_{i=1}^{N_t}$ denote the images containing annotated objects of classes $\mathcal{C}_t$, and $\mathcal{Y}_t = \{Y_i, Y_i = [\mathbf{y}_1 ..., \mathbf{y}_a] \in \mathbb{R}^{a \times 5}\}_{i=1}^{N_t}$ denote the annotations, where N_t is the number of available images for stage t, Y_i is the set of labels for image $\mathbf{x}_i$, and a is the number of annotated instances for image $\mathbf{x}_i$ respectively. Each label $\mathbf{y}$ is a 5-dim vector: $\mathbf{y} = [\mathbf{y}_{box}, y]$, where the 4 elements of $\mathbf{y}_{box}$, $[x_{min}, y_{min}, x_{max}, y_{max}]$, are the bounding box coordinates for one instance, while the last element y is the class label ($y \in \mathcal{C}_t$). In stage t, an object detector D_t must detect instances of accumulated classes $\mathcal{C}_1 \cup \mathcal{C}_2 ... \cup \mathcal{C}_t$, although only the newest images $\mathcal{X}_t$ and annotations $\mathcal{Y}_t$ are available.

Recently, many algorithms have been proposed for class incremental image recognition (CIIR). However, CIOD shows more theoretical significance than CIIR because CIOD requires localizing and classifying objects simultaneously and incrementally. The practical significance of CIOD is that CIOD has broad applications. For example, online services (such as online shopping and instance searching) that are equipped with object detection models must cover the varying interests of users, making incremental learning a critical feature for a personalized and robust object detection system. CIOD may be more useful on edge devices such as mobile robots, self-driving cars and smartphones, which rely on object detection for many important applications, including vision-based grasping, autopilot and augmented reality. These intelligent agents also have to detect objects incrementally to respond to a series of changing circumstances once deployed.

The first modern convolutional neural network [12, 22, 23] based incremental object detector, ILWCF [41], formulates the incremental detection problem as a classification with localization task on precomputed region proposals with Fast R-CNN [9]. Using knowledge distillation [26], the classification and box regression outputs of the old model on the data of the new stage are preserved while learning new classes. This method is further improved by RKT [34], where a relation distillation loss function that aims to preserve the relations of selected proposals is proposed. These methods produce regional proposals using EdgeBoxes [51] or MCG [33], which are slow and produce proposals of lower quality than those generated with a modern region proposal network (RPN). CIFRCN [13] is the first to apply the knowledge-distillation technique to RPN, yielding an end-to-end class-incremental object detector based on Faster R-CNN. However, the experimental results of CIFRCN are reported under a different experimental setting from that used with ILWCF. Faster ILOD [32] is also based on Faster R-CNN, with knowledge distillation applied to RPN and Fast R-CNN heads and backbone features. Thus, incrementally learning object detectors without catastrophic forgetting remains challenging.

In this study, we propose an effective class incremental object detector, RT-Net, based on Faster R-CNN to solve the class-incremental object detection problem. As shown in Figs. 1 and 2, there are two problem settings for class-incremental object detection, depending on how to build the incremental datasets. The first setup, which we denote as set-overlapped, follows [41]: each training stage contains all the images that have at least one object of a novel class and with only the newest classes annotated. Following [13], we also consider the set-disjoint setup: each learning stage contains a unique set of images, whose objects only belong to the novel classes in the current stage. Three techniques, generative feature replay (GFR), guided feature distillation (GFD) and holistic ranking transfer (HRT), are the primary components of the method. Generative replay has been used in incremental classification tasks. However, how to effectively extend it to incremental detection is still unclear. Faster R-CNN consists of three parts: a backbone for feature extraction, an RPN for proposal generation and an FRCN (RoI head) for proposal classification and bounding box regression. As shown in Fig. 3, guided feature distillation is used on the output of the backbone network to prevent forgetting. Then, we can use generative feature replay to replay the features of old classes for FRCN. We propose using center loss with softmax loss for classification to make the final feature vectors easily modeled by Gaussians for replay. In addition, we use holistic ranking transfer to enable the new model to distinguish high-quality proposals from lower ones for old classes for the set-overlapped setting.

The contributions of this paper are as follows: (1) We identify that the primary problem of incremental object detection is a lack of old data, and design a generative feature replay framework to address the problem of incremental object detection. (2) We propose a guided feature distillation method for the backbone network and a holistic ranking transfer method for the RPN, to effectively transfer the knowledge of the old model to the new model. (3) We perform a thorough investigation of the components of the proposed method and various alternatives with experimental comparisons. (4) We experimentally demonstrate that RT-Net can achieve state-of-the-art performance on standard class-incremental object detection tasks under two different experimental settings.

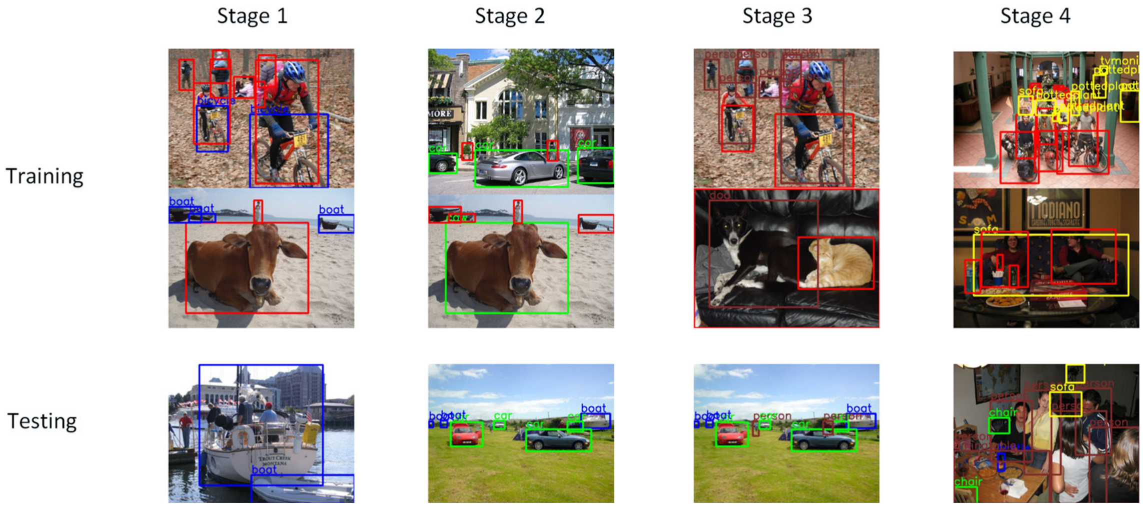

Fig. 1 Set-overlapped setting for class incremental object detection. In stage t the images in the training split that have any instances of a novel class will be used for training, in which some images may have been used in earlier stages. So the training images may contain instances from the old classes, which are not annotated. During testing, evaluation is based on all observed classes so far, with testing images that have any instances of an old or new class. Blue/green/brown/yellow boxes are annotations for new classes in stages 1-4, respectively. Red boxes: instances of old or future classes with no annotations

2 Related work

This study considers two major research topics: object detection and class incremental learning.

2.1 Object detection

State-of-the-art object detection models can be divided into two-stage and one-stage methods. Two-stage detectors first extract class-agnostic region proposals of the potential objects and then conduct classification and box regression. Fast R-CNN [9] uses precomputed proposals, which is time-consuming. Faster R-CNN [37] introduces a region proposal network (RPN) that shares features with the detection network and thus enables nearly cost-free proposal generation. Cascade R-CNN [2] uses multiple detection heads with increasing IoU thresholds to iteratively refine the detection results. In contrast, the one-stage detectors integrate the two stages into one unified process. YOLO [36] can predict bounding boxes and class probabilities directly from the full images. SSD [29] extends YOLO with multiscale feature maps and adopts diverse default boxes for various object shapes. RetinaNet proposes focal loss [28] to deal with dense detections on multiscale feature maps.

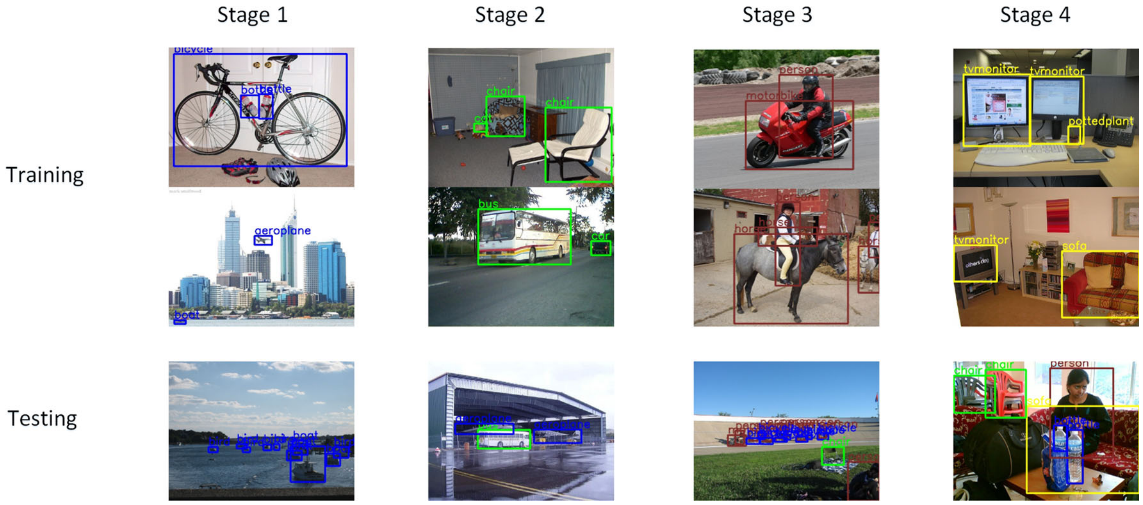

Fig. 2 Set-disjoint setting for class incremental object detection. Each learning stage contains a unique set of training images, whose objects only belong to the novel classes in the current stage. Testing is also different from the set-overlapped setting in which images containing instances of unseen classes are excluded

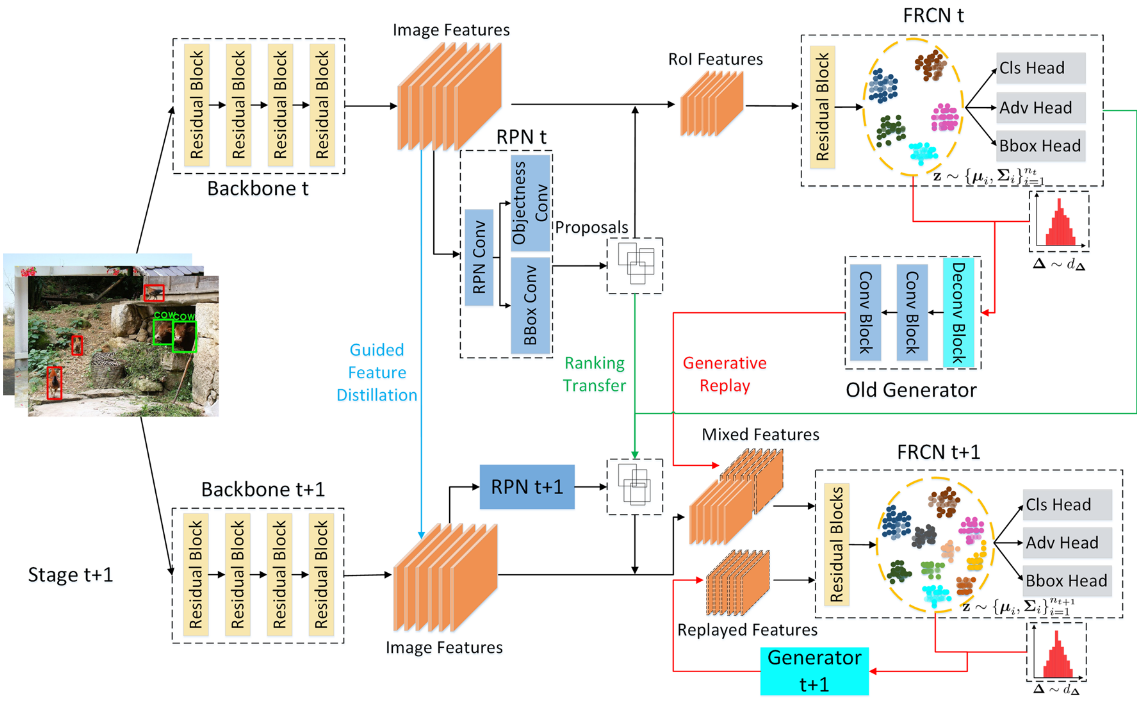

Fig. 3 System overview. The green line shows the ranking transfer process; the red line shows the generative feature replay; and the blue line shows the feature distillation

General object detection has been thoroughly studied, while incremental learning of object detectors still remains an open problem.

2.2 Class incremental learning

Catastrophic forgetting [30] is a core problem with class-incremental learning. Existing studies of incremental learning can be roughly divided into three primary groups: regularization-based, distillation-based and replay-based methods. As the pioneer work of regularization-based methods, elastic weight consolidation (EWC) uses the Fisher information guided regularization technique to protect the most important weights for past tasks [21]. Distillation-based methods, such as LwF [26], use probability distillation to make the predictions of the new model similar to those of the old network. Finally, replay-based methods use a generative model to sample synthetic data from previously learned distributions [20, 40]. However, there are relatively few methods for incremental object detection learning. To our knowledge, ILWCF [41], RKT [34], Faster ILOD [32] and CIFRCN [13] are the most relevant methods that have been investigated in prior studies for incremental object detection.

3 Methodology

3.1 Problem formulation and system overview

The proposed RT-Net is designed based on the Faster R-CNN framework, which consists of three parts: a backbone B, an RPN R and an FRCN F. We first review the forward process of Faster R-CNN.

Faster R-CNN An image $\mathbf{x}$ will first be forwarded through the backbone, and the image feature maps, denoted as $\mathbf{f}$, can then be determined. Based on $\mathbf{f} = B(\mathbf{x})$ and a set of anchor boxes $\mathcal{A}$, RPN then performs several additional convolutions to determine the initial proposals, which are l bounding boxes $\mathcal{P}^a = \{r_1^a, r_2^a ..., r_l^a\}$, with foreground/background scores $\{\mathbf{s}_1^r, \mathbf{s}_2^r ..., \mathbf{s}_l^r\}$ for the softmax function or objectness scores $\{s_1^r, s_2^r ..., s_l^r\}$ for the sigmoid function. After non-maximum suppression (NMS) and eliminating low-confidence boxes, the final proposal set $\mathcal{P}$ contains m proposals $\{r_1, r_2 ..., r_m\}$. With such proposals, the image feature will be fed to the RoI-pooling layer to obtain the corresponding RoI features $\{\mathbf{u}_1, \mathbf{u}_2 ..., \mathbf{u}_m\}$. Then, such feature maps are fed to FRCN, which produces the final feature vectors $\{\mathbf{v}_1, \mathbf{v}_2 ..., \mathbf{v}_m\}$. The final outputs

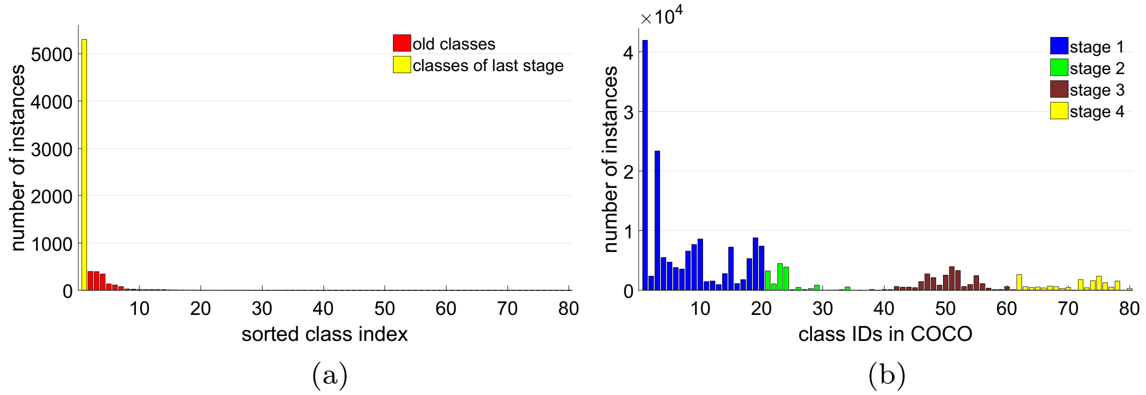

Fig. 4 Instance distributions with the COCO dataset. **a** Instance distribution of the last learning stage for a 75+1+1+1+1+1 set-overlapped incremental detection setting. Class indexes are sorted by available instance numbers. **b** Instance distribution of 4 stages for a 20+20+20+20 set-disjoint setting. Class indexes are sorted by COCO IDs

of FRCN are m bounding boxes $\{bb_1^p, bb_2^p..., bb_m^p\}$ with multiclass scores $\{\mathbf{s}_1^{bb}, \mathbf{s}_2^{bb}..., \mathbf{s}_m^{bb}\}$, respectively. Then, the system applies class-wise NMS after each proposal is assigned to a class with the largest score and obtains final q boxes $\mathcal{P}^{bb} = \{bb_1, bb_2..., bb_q\}$ with corresponding assigned class scores $\mathcal{S}^{bb} = \{s_1^{bb}, s_2^{bb}..., s_q^{bb}\}$.

RT-Net for Incremental Object Detection For each new stage $t+1$, n_{t+1}^{new} new classes are added for learning. The detector must localize all instances of $n_{t+1} = n_t + n_{t+1}^{new}$ classes seen thus far. We let $\mathcal{G}^{new}$ denote the set of ground truth instances for new classes in image $\mathbf{x}$. For the set-disjoint setting, all old images are excluded from the training set. However, for the set-overlapped setting, some old images are available. We also let $\mathcal{G}^{old}$ denote the set of pseudo ground truths for old classes in $\mathbf{x}$, which serve as the substitute for true annotations. These pseudo ground truths are calculated by thresholding the detection results of the training set of stage t, with thresholds computed and saved from stage t. The method to compute such a threshold for every class after training of stage t is introduced in Algorithm 1. The data distributions of old/new classes are different for the set-overlapped and set-disjoint settings. As shown in Fig. 4a, the numbers of instances for each class that can be used during the last training stage are plotted. Despite some classes with large instance numbers, most old classes have only a few samples available. This problem is even worse for the set-disjoint setting because images containing instances of old classes are all excluded. Figure 4b shows that the data distributions along the learning stages are imbalanced. Based on such observations, we thus propose using generative replay to mitigate the lack of data and data imbalances during incremental object detection. Specifically, a generator G will be added to the Faster R-CNN framework to replay the RoI features. Thus, the RT-Net $\mathcal{M}$ consists of four components $\mathcal{M} = \{B, R, F, G\}$.

Algorithm 1 Threshold computing for classes in stage t.

```
Input: Training set $\{\mathcal{X}_t, \mathcal{Y}_t\}$ and trained detector $D_t$ for stage $t$.
Output: Threshold set $\mathcal{T}_t = \{th_c\}_{c=n_{t-1}+1}^{n_t}$.
$\{N_c = 0\}_{c=n_{t-1}+1}^{n_t}$, $\{\mathcal{S}^c = \{\}\}_{c=n_{t-1}+1}^{n_t}$.
for each label set $Y \in \mathcal{Y}_t$ do
    for each class $c \in \mathcal{C}_t$ do
        Calculate $n_c$, the number of ground truth boxes from $Y$ for class $c$.
        $N_c = N_c + n_c$.
    end for
end for
for each image $\mathbf{x} \in \mathcal{X}_t$ do
    Feed $\mathbf{x}$ to $D_t$ to obtain final scores and boxes after NMS: $\mathcal{S}^{bb}, \mathcal{P}^{bb} = D_t(\mathbf{x})$.
    for each class $c \in \mathcal{C}_t$ do
        $\mathcal{S}^c = \mathcal{S}^c \cup \mathcal{S}_c^{bb}$.
        Sort $\mathcal{S}^c = \{s_i^c\}_{i=1}^{len}$ in descending order, and remove any element $s_i^c$ with $i > 2N_c$.
    end for
end for
for each class $c \in \mathcal{C}_t$ do
    $th_c := s_{2N_c}^c$.
end for
```

In stage $t = 1$, the Faster R-CNN $\{B_1, R_1, F_1\}$ is first trained with images $\mathcal{X}_1$. After regular training, the generator G_1 is added, and a new adversarial head is added to FRCN to serve as the discriminator, as in generative adversarial nets (GANs) [11]. The generator is then trained with Faster R-CNN for several epochs. Then, the RoI features for classes n_1 can be replayed by the generator G_1. In stage $t = 2$, we have a new set of images $\mathcal{X}_2$ with instances belonging to new classes or a mixture of new and old classes (with no annotations). The new network $\mathcal{M}_2$ is initialized from $\mathcal{M}_1$. To make RoI features consistent with the last stage, guided feature distillation (GFD) is deployed on the backbone network. High-quality proposals are important to detect instances of old classes; thus, we also design effective knowledge transfer methods for the RPN. $\{B_2, R_2, F_2\}$ can then be trained effectively. After training of $\{B_2, R_2, F_2\}$, although knowledge transfer techniques are applied to the backbone, this component in $\mathcal{M}_1$ and $\mathcal{M}_2$ are different; thus, the two sets of RoI features computed using old and new models, $\mathcal{U}_1$ and $\mathcal{U}_2$, may lie in different feature spaces and are not compatible with each other. Inspired by [18], we propose transforming features $\mathcal{U}_1$ to the same feature space as $\mathcal{U}_2$. We train a feature adaptation network T_2 to map $\mathcal{U}_1$ to the same space as $\mathcal{U}_2$ and retrain the FRCN part with the generator. Then, we determine the newest trained model $\mathcal{M}_2 = \{B_2, R_2, F_2, G_2\}$. We follow the same procedure for subsequent stages; the difference is that initial training for F_{t+1} also uses the RoI features generated by G_t. The training process for stage $t+1$ is shown in Fig. 3.

3.2 Generative feature replay for FRCN

We use feature replay with the RoI features to prevent forgetting in the FRCN and to mitigate data imbalance between the new and old classes. A straightforward choice is using conditional GAN (cGAN) [5], which consists of a class-conditional generator $G(\mathbf{z}, c)$ that is associated with a class-conditional discriminator $D(\mathbf{u}, c)$. Typically, c is the class label and sampled from the categorical distribution P_d, and $\mathbf{z}$ is random noise sampled from a normal distribution. The generator and discriminator are trained to optimize the following adversarial objective:

$$\mathcal{L}_{GAN}(G, D) = \mathop{\mathbb{E}}_{c \sim P_d}\Bigg[\mathop{\mathbb{E}}_{\mathbf{u} \sim d_c}[log D(\mathbf{u}, c)] + \mathop{\mathbb{E}}_{\mathbf{z} \sim \mathcal{N}(0, I)} [log(1 - D(G(\mathbf{z}, c), c))]\Bigg]. \tag{1}$$

The primary drawback of using standard cGAN is that the sampled normal distribution and class-conditional signals have no connection with the true final feature distributions. Sampling randomly makes the quality of certain replayed RoI features uncontrollable. To solve this problem, we propose using Gaussian Mixture (GM) models to unify the distributions for sampling vectors and posterior distributions of final features.

Given a training set with $n_t + 1$ classes including a background class, for N final feature vectors of proposals with class labels $\{(\mathbf{v}_j; y_j), 1 \leq j \leq N\}$, the Softmax loss is defined as:

$$\mathcal{L}_{softmax} = -\frac{1}{N}\sum_{j=1}^{N} log \frac{exp(\mathbf{w}_{y_j}^T \mathbf{v}_j + b_{y_j})}{\sum_{i=1}^{n_t+1} exp(\mathbf{w}_i^T \mathbf{v}_j + b_i)}. \tag{2}$$

Different from the softmax loss, we may further assume that the final feature vector $\mathbf{v}$ follows Gaussian Mixture distributions. Under such an assumption, the classification loss can be computed as the cross-entropy between the posterior probability distribution and the one-hot class label as:

$$\begin{aligned}\mathcal{L}_{ce+gm} &= -\frac{1}{N}\sum_{j=1}^{N}\sum_{i=1}^{n_t+1} \mathbb{1}(y_j = i) \log p(i|\mathbf{v}_j) \\ &= -\frac{1}{N}\sum_{j=1}^{N} \log \frac{\mathcal{N}(\mathbf{v}_j; \boldsymbol{\mu}_{y_j}, \boldsymbol{\Sigma}_{y_j}) p(y_j)}{\sum_{i=1}^{n_t+1} \mathcal{N}(\mathbf{v}_j; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) p(i)}\end{aligned} \tag{3}$$

where $\mathbb{1}$ is the indicator function, which equals 1 if y_j equals i, or 0 otherwise; $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are the mean and covariance of class i in the feature space; and $p(i)$ is the prior probability of class i. Typically, optimizing the classification loss only cannot directly drive the final feature vectors toward the expected Gaussian mixture distributions. To solve this problem, a likelihood regularization term for measuring how well the training samples fit the assumed distributions is used [44]. However, we cannot assume that the background class in object detection also follows a Gaussian distribution. Therefore, we only consider positive samples in this case:

$$\mathcal{L}_{lkd} = -\sum_{i=1}^{N} \mathbb{1}(y_i \leq n_t) \log \mathcal{N}(\mathbf{v}_i; \boldsymbol{\mu}_{y_i}, \boldsymbol{\Sigma}_{y_i}). \tag{4}$$

Finally the GM loss for object detection $\mathcal{L}_{gm}$ is defined as follows:

$$\mathcal{L}_{cls\_gm} = \mathcal{L}_{ce+gm} + \lambda \mathcal{L}_{lkd}, \tag{5}$$

where λ is a nonnegative weighting hyperparameter that is set to 0.1. Although GM loss can connect the distributions of final feature vectors and sampling prior for cGAN, we propose using a much simpler method to achieve similar results, center loss [45] for positive samples, which is defined as follows:

$$\mathcal{L}_{center} = \frac{1}{2N_{pos}} \sum_{j=1}^{N_{pos}} \|\mathbf{v}_j^p - \boldsymbol{\mu}_{y_j}\|_2^2. \tag{6}$$

Therefore, the classification loss is defined as:

$$\mathcal{L}_{cls\_cen} = \mathcal{L}_{softmax} + \mathcal{L}_{center}. \tag{7}$$

We let $\mathcal{L}_{cls}$ denote the general classification loss referring to $\mathcal{L}_{cls\_gm}$ or $\mathcal{L}_{cls\_cen}$. We use $\mathcal{L}_{cls\_cen}$ in the experiments. With the learned classification model, the class centers $\{\boldsymbol{\mu}_i\}_{i=1}^{n_t}$ and covariance matrices $\{\boldsymbol{\Sigma}_i\}_{i=1}^{n_t}$ can be calculated, and the final features can be modeled with Gaussian mixture distributions. To achieve replay, we propose training a generator G and discriminator D that share layers with FRCN except an independent head to effectively replay the features.

Another problem of replay for object detection is how to set the box regression targets for the replayed features. To solve this problem, we propose to sample normalized coordinate shifts Δ as the second conditional signal. For every item g in ground truth set $\mathcal{G}$ of image $\mathbf{x}$, coordinate shifts that achieve IoU above 0.5 with g will be sampled. Thus, the sampled coordinate shifts are also the box regression targets. Negative samples can then be generated by shifting positive samples until the IoUs are below 0.5 and applying simple feature-padding. The FRCN and generator can be trained using the following loss:

$$\mathcal{L}_{adv} = \mathop{\mathbb{E}}_{c\sim P_d}\left[\mathop{\mathbb{E}}_{\mathbf{u}^p\sim d_c}[logD(\mathbf{u}^p)] + \mathop{\mathbb{E}}_{\mathbf{z}\sim\mathcal{N}_c,\Delta\sim d_\Delta} [log(1 - D(G(\mathbf{z}, \Delta)))]\right], \tag{8}$$

where $\mathbf{z}$ is sampled from the Gaussian distribution of class c. We let $\mathbf{u}_g$ and $\mathbf{u}_{gen}$ denote the feature maps for the overlapped region between the ground truth features and the generated RoI features. $\mathcal{L}_{over}$ is defined as the L2 loss between $\mathbf{u}_g$ and $\mathbf{u}_{gen}$. We also let $\mathcal{L}_{rec}$ denote the reconstruction loss (L2 loss) for real RoI features $\mathbf{u}$ and $G(F(\mathbf{u}))$. The full loss for FRCN F and generator G is:

$$\mathcal{L}_{full} = \mathcal{L}_{adv} + \mathcal{L}_{cls} + \mathcal{L}_{bbox} + \mathcal{L}_{over} + \mathcal{L}_{rec}, \tag{9}$$

where $\mathcal{L}_{bbox}$ is the smooth L1 loss [9] for real and generated RoI features. Classification loss $\mathcal{L}_{cls}$ is also computed using both real and generated RoI features. For instances of old classes, the class labels and targets for bounding box regression are computed using the pseudo ground truths because there are no annotations for them. The output heads of FRCN are reset before adversarial learning to keep the joint training of generator and FRCN stable.

Determining the Ratio of Replayed Samples The criterion for determining the ratio of replayed samples is simple. As the numbers of annotated instances and the average numbers of positive samples associated with each instance for classes in each stage can be calculated, these numbers are stored and used to compute the ratios of replayed positive samples for old classes in the new learning stage. Specifically, in stage t, for any old class c with n^a annotated instances and n^p positive samples per instance, $max(n^a * n^p, n^a_{min} * n^p)$ positive samples will be replayed in each epoch. The parameter n^a_{min} is set to 200 to deal with insufficient training data. The positive samples within a minibatch are randomly picked from the set of $\mathcal{C}_1 \cup \mathcal{C}_2 ... \cup \mathcal{C}_t$ according to the sample ratios.

3.3 Guided feature distillation for backbone network

To make feature replay feasible in incremental object detection, feature distillation should be deployed on the backbone network to guarantee that the RoI features are consistent with previous stages when learning new classes. We propose using guided feature distillation (GFD), to compute spatial attention weights from feature maps and $\mathcal{G}^{old}$. Formally, the GFD loss is:

$$\mathcal{L}_{gfd} = \frac{1}{2N_f}\sum_{i=1}^{W}\sum_{j=1}^{H} w_{ij}\sum_{c=1}^{C}(f^{new}_{ijc} - f^{old}_{ijc})^2, \tag{10}$$

where $N_f = C\sum_{i=1}^{W}\sum_{j=1}^{H} w_{ij}$, W, H, and C are the width, height, and number of channels of the feature maps, respectively. As shown in Fig. 5, the spatial attention weights are computed first using summation of all channels of the feature maps: $w_{ij} = \sum_{c=1}^{C}|f^{old}_{ijc}|^2$, and then normalization: $w_{ij} = w_{ij}/w_{max}$. Here w_{max} is the maximum value of the weights. We then use Gaussian-like kernels to reweight foreground regions of old classes to protect the informative feature regions from drastic changes. Specifically, for a pseudo ground truth from $\mathcal{G}^{old}$ with coordinates $[x_{min}, y_{min}, x_{max}, y_{max}]$ on the feature maps and class score s, the weights for this region can be rewritten as:

$$w_{ij} = s * exp^{-(\frac{(i-c_x)^2}{2\sigma_x^2} + \frac{(j-c_y)^2}{2\sigma_y^2})}, \tag{11}$$

where $\sigma_x = (x_{max} - x_{min})/2$, $\sigma_y = (y_{max} - y_{min})/2$, $c_x = (x_{max} + x_{min})/2$ and $c_y = (y_{max} + y_{min})/2$. Because a pixel may be located in the overlapped region of multiple RoIs, only the highest weight computed according to any single RoI is used. Using different distillation weights for old RoIs and other regions, the knowledge of the old model can thus be effectively transferred to the new model without harming the learning of the instances of new classes.

3.4 Feature transformation

Feature transformation is applied after training Faster R-CNN in stage $t > 1$. With network $\mathcal{M}_t$ initialized from $\mathcal{M}_{t-1}$, the backbone, RPN and FRCN are trained with regular losses for instances of new classes and with corresponding guided feature distillation, holistic ranking transfer and feature replay methods for old classes. Although feature distillation is used for backbone B_t, its parameters may be different from the old one B_{t-1}. Thus, the two sets of RoI features, $\mathcal{U}_t$ and $\mathcal{U}_{t-1}$, lie in different feature spaces and are not compatible with each other. Therefore, we train a feature adaptation network T_t to map $\mathcal{U}_{t-1}$ to the same space as $\mathcal{U}_t$. Once the feature transformation network is trained, we create a new feature set $\mathcal{U}^{trans}_t$ by transforming the existing features generated from the generator G_{t-1} to the same feature space as $\mathcal{U}_t$. Then, the FRCN can be trained using the combined features $\mathcal{U}^{trans}_t$ and $\mathcal{U}_t$. When training for FRCN is completed, we train the conditional generator together with FRCN using

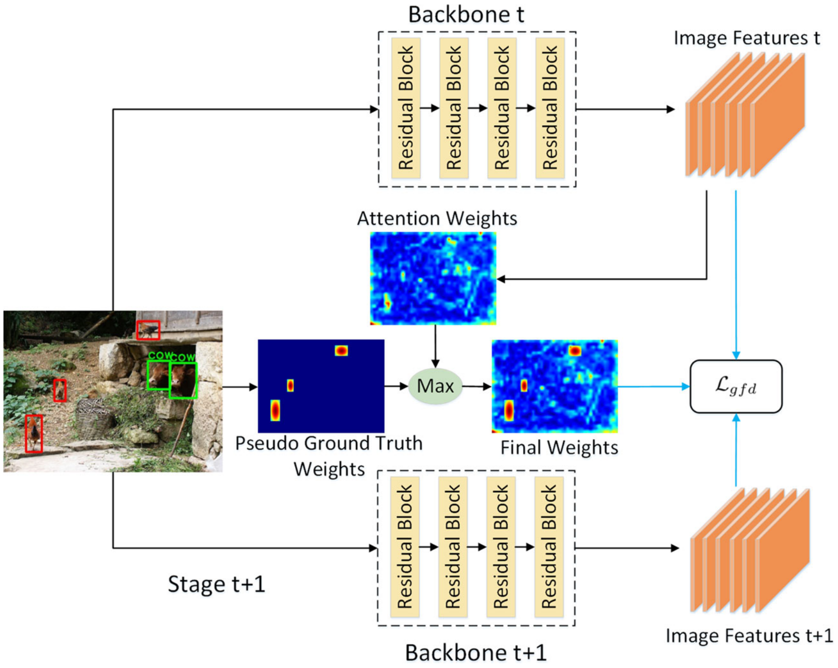

Fig. 5 Guided Feature Distillation. Green boxes are annotations for new classes. Red boxes are pseudo ground truths detected by the old model. The spatial attention map is computed using the output feature maps of the backbone network, demonstrating diverse importance weights of spatial regions for the old model. Another weight map is computed to highlight the old class regions. The final weight map is a combination of these 2 maps and is used to protect important features from drastic changes during learning new classes

$\mathcal{U}_t^{trans}$ and $\mathcal{U}_t$ for another several epochs, yielding the new generator G_t.

The feature transformation is achieved by learning a transformation function $T_t : \mathbb{R}^d \rightarrow \mathbb{R}^d$, which maps the output of the previous RoI pooling layer to the current RoI feature space using the current task images $\mathcal{X}_t$. We let $\mathcal{U}^{pair}$ denote the set of feature pairs $(\bar{\mathbf{u}}, \mathbf{u}) \in \mathcal{U}^{pair}$. Given an image $\mathbf{x} \in \mathcal{X}_t$ and region r, $\bar{\mathbf{u}}$ corresponds to the RoI feature extracted with $RoIpooling(B_{t-1}(\mathbf{x}), r)$, while. Conversely, $\mathbf{u}$ corresponds to the feature representation extracted with the model in the current stage (i.e., $RoIpooling(B_t(\mathbf{x}), r)$. The RoI features used belong to the positive samples w.r.t. new ground truths G^{new}, and pseudo ground truths of old classes G^{old} in the set-overlapped setting.

When training the feature transformation network T, we use the following loss function:

$$\mathcal{L}_{\text{ft}}(\bar{\mathbf{u}}, \mathbf{u}) = \mathcal{L}_{\text{sim}}(\mathbf{u}, T(\bar{\mathbf{u}})) + \mathcal{L}_{\text{cls}}(T(\bar{\mathbf{u}}), y) + \mathcal{L}_{\text{box}}(T(\bar{\mathbf{u}}), \mathbf{y}_{box}), \quad (12)$$

where y and $\mathbf{y}_{box}$ are the corresponding class label and box regression targets for $\mathbf{u}$. The first term $L_{\text{sim}}(\mathbf{u}, T(\bar{\mathbf{u}}))$ is the L1 loss, which encourages the adapted feature descriptor $T(\bar{\mathbf{u}})$ to be similar to $\mathbf{u}$, which is its counterpart that is extracted from the updated network. The purpose of this method is to transform features between different feature spaces, while feature distillation can prevent features from drifting markedly in the feature space. The second loss term $\mathcal{L}_{\text{cls}}(T(\bar{\mathbf{u}}), y)$ is the cross-entropy loss computed by feed the transformed features to the FRCN. This term encourages transformed features to belong to the correct class y. Additionally, $\mathcal{L}_{\text{box}}(T(\bar{\mathbf{u}}), \mathbf{y}_{box})$ is the smooth L1 loss, which encourages transformed feature descriptors to achieve similar results for regression of box coordinates.

Algorithm 2 Sampling for ranking transfer.

```
1: Input: Ground truth set of new classes $\mathcal{G}^{new}$, proposal set $\mathcal{P}$, final box set $\mathcal{P}^{bb}$, and threshold set $\mathcal{T} = \{th_i\}_{i=1}^{n_t}$ for the old classes.
2: Output: Grouped proposal set $\{\mathcal{P}_i^g\}_{i=1}^{n_g}$ for ranking transfer.
3: Select final boxes from $\mathcal{P}^{bb}$ with class scores above corresponding thresholds and form pseudo ground truth set $\mathcal{G}^{old} = \{g_i\}_{i=1}^{n_g}$.
4: Compute IoUs between $\mathcal{P}$ and $\mathcal{G}^{old}$.
5: for each pseudo ground truth $g \in \mathcal{G}^{old}$ do
6:     Select proposals that are with the largest IoU with $g$ rather than the other elements in $\mathcal{G}^{old}$ or $\mathcal{G}^{new}$, and form set $\mathcal{P}^g$.
7:     Add proposals merged (as a result of NMS) to each element $r \in \mathcal{P}^g$ to $\mathcal{P}^g$, and assign the same $s^{bb}$ to them. Split $\mathcal{P}^g$ to 5 groups $\mathcal{P}_1$-$\mathcal{P}_5$, with $s_{old}^r$ ranges from (0.9,1], (0.8,0.9], (0.7,0.8], (0.6,0.7] and (0.1,0.6].
8: end for
```

3.5 Holistic ranking transfer for RPN

In the set-overlapped setting of incremental detection, some old images may be reused for training during new stages. It is appealing to transfer the knowledge of old RPN to the new one. Instead of knowledge distillation, we use listwise ranking loss [4, 47] to achieve knowledge transfer for RPN; specifically, this method is used to transfer the ranking orders of proposals generated by old models to the newest model.

We use π to denote a permutation of the list (with length k) indexes. Formally, we denote the candidate region proposals as R. Then, the probability of a specific permutation π is given as:

$$P(\pi|R) = \prod_{i=1}^{k} \frac{exp\left[S(r_{\pi_i})\right]}{\sum_{j=i}^{k} exp\left[S(r_{\pi_j})\right]}, \tag{13}$$

where $S(r)$ is a score function based on the similarity between proposal r and pseudo ground truth g:

$$S(r) = -\alpha\|\mathbf{s}^r - \mathbf{s}_g^r\|_2^\beta, \tag{14}$$

α and β are sharpening parameters to make the margin between samples large sufficient for optimization. In this study, we use weighted average scores from the RPN and FRCN of the old model to compute the transfer targets for any proposal r (including g): $\mathbf{s}_{old}^r = \mathbf{s}^r * 0.7 + (s^{bb}, 1 - s^{bb}) * 0.3$, while for the new model, the output of Softmax is used directly: $\mathbf{s}_{new}^r = \mathbf{s}^r$. If using the sigmoid function as the objectness score function, for the old model, the weighted average score from the RPN and FRCN is used: $s_{old}^r = s^r * 0.7 + s^{bb} * 0.3$; and we directly use the objectness score: $s_{new}^r = s^r$ for the new model. Cross entropy loss for permutations Π is used as the loss function:

$$\mathcal{L}_{hrt}(R_{new}, R_{old}) = -\sum_{\pi \in \Pi} P(\pi|R_{old}) log P(\pi|R_{new}). \tag{15}$$

As shown in Fig. 6, this method is named as holistic ranking transfer because it uses the outputs from both the RPN and FRCN of the old model. For 2-stage detectors, the output of the second stage provides a more accurate evaluation on the quality of the bounding boxes. The sampling method for ranking transfer in stage $t+1$ is summarized in Algorithm 2. When training, randomly selected $N_l = 8$ lists with $k = 8$ proposals (1 each from $\mathcal{P}_1$-$\mathcal{P}_4$, and another 4 from $\mathcal{P}_5$) each are used. The full loss for training RPN is the summation of holistic ranking transfer loss for old classes, regular objectness loss for new classes, and box regression loss for all classes:

$$\mathcal{L}_{RPN} = \mathcal{L}_{hrt}^{old} + \mathcal{L}_{objectness}^{new} + \mathcal{L}_{bbox}. \tag{16}$$

As shown in Fig. 7, the proposed method can effectively transfer the ranking knowledge of the old model to the new one, where high-quality proposals are scored higher than low-quality proposals.

4 Experiments

4.1 Datasets

We evaluate the proposed method using two detection benchmarks: PASCAL VOC 2007 and COCO 2014. VOC 2007 contains 5K images in the trainval split and 5K images in the test split for 20 object classes. Conversely, COCO has 80K images in the training set and 40K images in the validation set across 80 object classes. We use the standard mean average precision (mAP) of IoU = 0.5 for VOC 2007 and mAP weighted across different IoUs from 0.5 to 0.95 for evaluation with COCO. Evaluation with VOC 2007 is performed on the test split (i.e., the train and val splits are used for training), while for COCO, 5K images in the minival subset from the validation set are used for testing (i.e., the train split and remaining images in the val split are used for training).

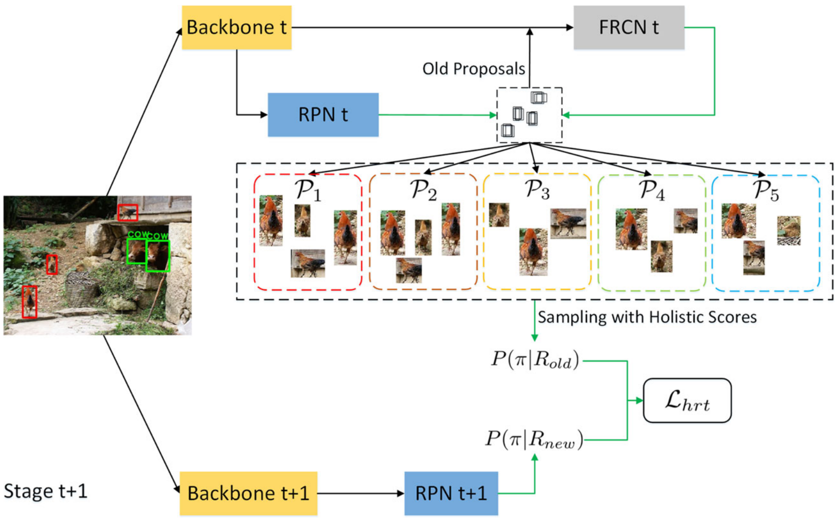

Fig. 6 Holistic ranking transfer. The red boxes are pseudo ground truths detected by the old model. Proposals are grouped and lists of proposals are sampled for knowledge transfer from the old model

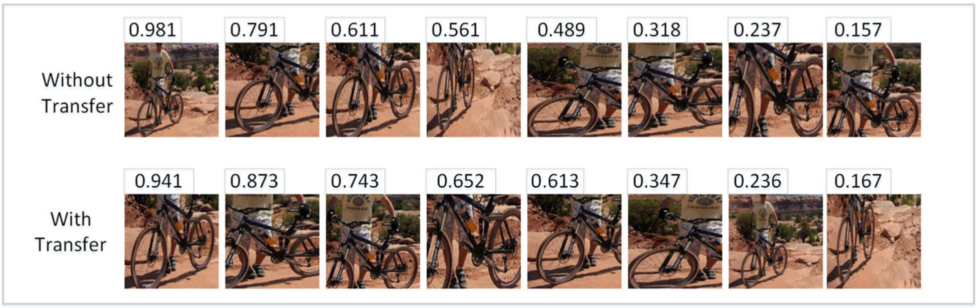

Fig. 7 Results of holistic ranking transfer. All proposals resized to the same size for visualization. When testing, our ranking transfer method can preserve the ranking orders of old models for high quality proposal generation

4.2 Implementation details

We use the standard implementation for the ResNet-based Faster R-CNN network [38]. ResNet-50 and ResNet-101 [15] pretrained on ImageNet [39] are used as the backbones in different experimental settings, following [13, 34]. The architecture of the generator is similar to [48], with 1 deconvolution block and 2 convolution blocks (using ResNet blocks). Additionally, the transformation network is a simple stack of 3 convolution blocks. We set the training epochs and learning rates of different learning stages following [32]. The parameters α and β in Eq. 14 are both set to 3.

4.3 Comparison with state-of-the-arts

The set-overlapped setting is the most popular experimental setting for incremental object detection. To compare fairly with the state-of-the-art methods [13, 34], we use their settings and perform incremental object detection experiments in 4 stages. We split the 20 classes in the VOC dataset into 4 groups with 5 classes added every stage. Similarly, with the COCO dataset, 20 new classes are added every stage. For the set-overlapped setting, we compare the proposed method with ILWCF [41], RKT [34], ILWCF Faster [32] and Faster ILOD [32]. We use the reported results from [32, 34]. As shown in Table 1, compared with ILWCF and RKT, the proposed method performs much better in terms of mAP. However, these two methods do not use modern RPN to generate proposals. Therefore, for fair comparison, we take ILWCF Faster and Faster ILOD as the baselines. The final mAP improvements over ILWCF Faster and Faster ILOD are 4.6% and 4.2%, respectively. We also combine baseline methods with components of the proposed method: generative feature replay (GFR), holistic ranking transfer (HRT) and guided feature distillation (GFD). In this study, feature transformation is tied to generative feature replay if not specified. With knowledge distillation denoted as KD, we implement GFD + Faster ILOD, GFD + KD-RPN + GFR-FRCN, and GFD + HRT-RPN + KD-FRCN. We also implement EdgeBoxes + GFD + GFR-FRCN. The experimental results with the VOC dataset in Table 1 show that the proposed method outperforms its counterparts for each stage. The proposed GFD can improve Faster ILOD by 0.9% final mAP, indicating that GFD outperforms FD. Compared with ILWCF and RKT, EdgeBoxes + GFD + GFR-FRCN achieve much better performance. GFD + KD-RPN + GFR-FRCN also outperforms GFD + Faster ILOD, Faster ILOD and ILWCF Faster by 2.6%, 3.5% and 3.9%, respectively, indicating that the proposed GFR method is effective for FRCN. GFD + HRT-RPN + KD-FRCN also outperforms GFD + Faster ILOD, which verifies that the proposed HRT method is effective for RPN.

Table 1 Results (mAP %) on VOC dataset with 5 new classes in every stage, under the set-overlapped setting. The best results are highlighted in bold

	5	10	15	20
ILWCF [41]	57.6	55.9	53.7	47.0
RKT [34]	57.6	56.8	56.9	52.9
EdgeBoxes + GFD + GFR-FRCN	57.8	57.4	57.1	53.4
ILWCF Faster [32]	69.6	58.7	51.5	49.3
Faster ILOD [32]	69.6	58.5	53.8	49.7
GFD + Faster ILOD	69.6	58.8	54.9	50.6
GFD + HRT-RPN + KD-FRCN	69.6	58.9	55.4	51.8
GFD + KD-RPN + GFR-FRCN	69.7	59.1	56.6	53.2
Ours (RT-Net)	69.7	**59.4**	**57.4**	**53.9**

We also perform experiments with the COCO dataset, and the results are shown in Table 2. A marked improvement in mAP is achieved compared to ILWCF and Faster ILOD during the learning stage. The final improvements over ILWCF and Faster ILOD are 5.9% and 2.6%, respectively. Using Faster ILOD as a reference, the three components of the proposed method, GFD, HRT and GFR, contribute 0.6%, 0.7% and 1.3% mAP gains, respectively. The initial mAP gains of the proposed method are small, which verifies that the proposed method is effective for class incremental object detection, not for traditional fully supervised object detection.

Qualitative results on COCO are shown in Fig. 8, which indicates that the proposed method can effectively detect newly learned classes, such as persons in stage 3 and

Table 2 Results (mAP %) on COCO dataset with 20 new classes in every stage, under the set-overlapped setting. The best results are highlighted in bold

	20	40	60	80
ILWCF [41]	22.6	20.3	18.6	16.4
Faster ILOD [32]	29.2	26.3	22.5	19.7
GFD + Faster ILOD	29.2	26.5	22.9	20.3
GFD + HRT-RPN + KD-FRCN	29.2	26.7	23.9	21.0
GFD + KD-RPN + GFR-FRCN	29.3	26.9	24.2	21.7
Ours (RRT)	29.3	27.1	24.6	22.0
Ours (w/o FT)	29.3	27.3	24.3	21.4
Ours (Gaussian)	29.3	26.9	24.5	21.6
Ours (Vanilla)	29.2	26.7	24.2	20.5
Ours (RT-Net)	29.3	**27.8**	**25.3**	**22.3**

Table 3 Results (mAP %) on VOC dataset with 10 new classes in every stage, under the set-overlapped setting. The best results are highlighted in bold

	10	20
ILWCF [41]	65.8	62.4
RKT [34]	65.8	63.1
ILWCF Faster [32]	73.9	63.1
Faster ILOD [32]	73.9	63.2
OWOD [19]	73.9	64.6
Ours (RT-Net)	74.0	**67.9**

umbrella in stage 4, without forgetting old tasks of detecting boats or chairs. We also perform experiments on 2-stage incremental learning because it is the primary task setting in OWOD [19]. As shown in Table 3, the proposed method also outperforms state-of-the-art methods by a margin.

We perform experiments to evaluate the proposed method on extreme imbalanced incremental object detection, which is a 6-stage learning process with a base model trained on most of the classes in stage 1 and 1 new class added each time for the following stages. Experiments are performed on the VOC dataset with a 15+1+1+1+1+1

Fig. 8 Results of the 4-stage set-overlapped class incremental object detection on COCO

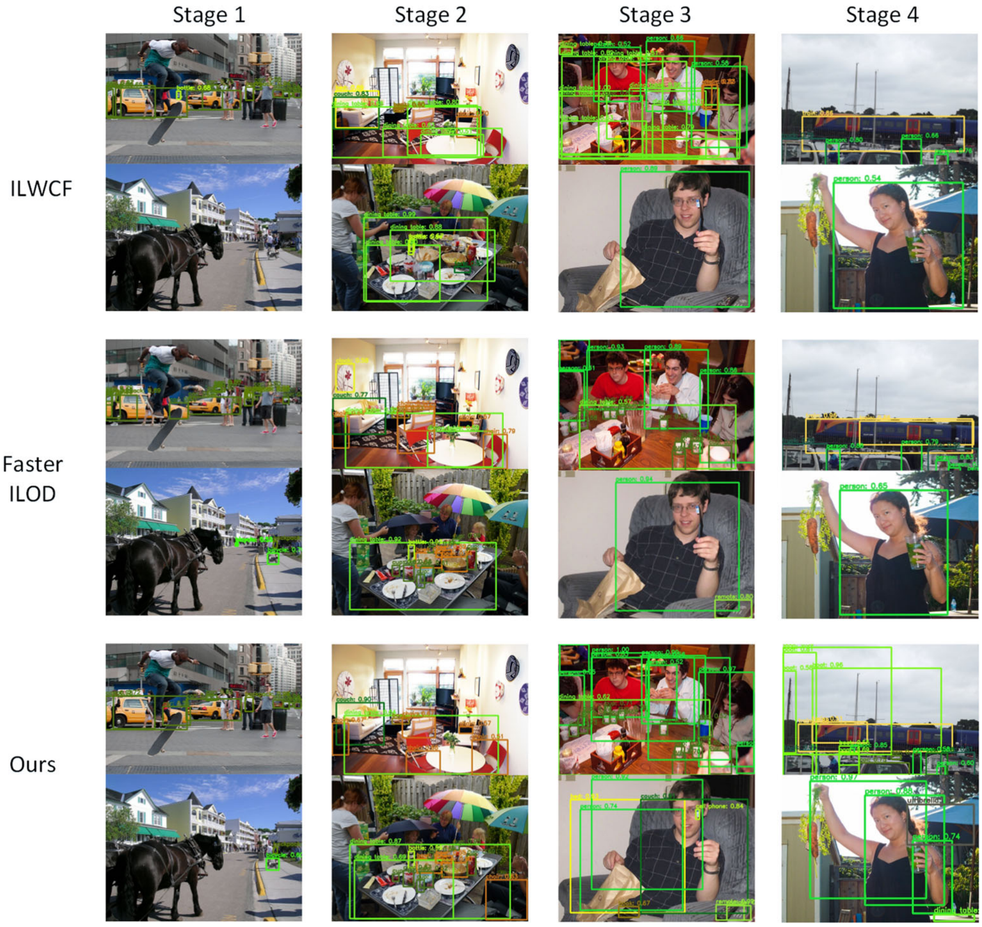

Table 4 Results (mAP %) on VOC dataset with 1 new class in every stage, under the set-overlapped setting. The best results are highlighted in bold

	15	16	17	18	19	20
ILWCF [41]	70.0	66.7	63.0	61.2	60.7	60.6
ILWCF Faster [32]	73.1	68.7	67.7	64.1	60.1	58.1
Faster ILOD [32]	73.1	70.1	68.3	65.9	63.7	61.3
Ours (RT-Net)	73.2	**72.3**	**70.4**	**68.9**	**66.4**	**64.3**

Table 5 Results (mAP %) on COCO dataset with 1 new class in every stage, under the set-overlapped setting. The best results are highlighted in bold

	75	76	77	78	79	80
ILWCF [41]	21.1	19.2	17.2	15.1	13.6	12.4
ILWCF Faster [32]	22.5	19.3	16.9	16.0	14.1	13.0
Faster ILOD [32]	22.5	21.4	19.6	17.1	15.1	13.9
Ours (RT-Net)	22.7	**21.7**	**20.5**	**18.9**	**17.7**	**15.8**

Table 6 Accuracies (AP %) of old and new classes of COCO dataset with 1 new class in every stage, under the set-overlapped setting. The best results are highlighted in bold

Class/method	mAP_{1-75}	76	77	78	79	80	$\mathrm{mAP}_{\mathrm{all}}$
C_{1-75}/Faster ILOD	22.5	-	-	-	-	-	22.5
C_{1-75}/Ours (RT-Net)	22.7	-	-	-	-	-	22.7
$C_{(1-75)+76}$/Faster ILOD	21.2	40.3	-	-	-	-	21.4
$C_{(1-75)+76}$/Ours (RT-Net)	**21.5**	**43.6**	-	-	-	-	**21.7**
$C_{(1-75)+...+77}$/Faster ILOD	19.4	37.2	21.1	-	-	-	19.6
$C_{(1-75)+...+77}$/Ours (RT-Net)	**20.3**	**41.4**	**23.2**	-	-	-	**20.5**
$C_{(1-75)+...+78}$/Faster ILOD	16.9	34.5	17.9	22.8	-	-	17.1
$C_{(1-75)+...+78}$/Ours (RT-Net)	**18.6**	**38.9**	**20.8**	**25.6**	-	-	**18.9**
$C_{(1-75)+...+79}$/Faster ILOD	14.8	32.8	16.5	19.4	21.9	-	15.1
$C_{(1-75)+...+79}$/Ours (RT-Net)	**17.4**	**37.2**	**19.8**	**22.3**	**25.6**	-	**17.7**
$C_{(1-75)+...+80}$/Faster ILOD	13.4	31.5	15.2	17.7	18.8	42.1	13.9
$C_{(1-75)+...+80}$/Ours (RT-Net)	**15.3**	**36.0**	**18.4**	**21.1**	**23.5**	**45.8**	**15.8**

Table 7 Results (mAP %) on VOC dataset with 5 new classes in every stage, under the set-disjoint setting. The best results are highlighted in bold

	5	10	15	20
ILWCF [41]	66.3	52.0	47.0	39.3
CIFRCN [13]	63.9	57.5	50.9	48.5
FD-RPN + GFR-FRCN	64.2	60.2	52.8	49.9
GFD + FD-RPN + KD-FRCN	64.0	59.1	51.7	49.4
Ours (RT-Net)	64.2	**61.7**	**54.4**	**51.8**

Table 8 Results (mAP %) on COCO dataset with 20 new classes in every stage, under the set-disjoint setting. The best results are highlighted in bold

	20	40	60	80
ILWCF [41]	49.2	30.2	23.2	20.9
CIFRCN [13]	58.1	31.8	26.0	22.9
FD-RPN + GFR-FRCN	58.3	32.6	27.4	23.7
GFD + FD-RPN + KD-FRCN	58.4	32.7	27.2	23.8
Ours (w/o FT)	58.5	32.9	27.5	23.8
Ours (Gaussian)	58.5	33.4	28.2	24.5
Ours (Vanilla)	58.4	33.1	27.4	24.1
Ours (RT-Net)	58.5	**33.8**	**28.9**	**24.9**

setting and on the COCO dataset with a 75+1+1+1+1+1 setting. For such experimental settings, after training in stage 1, we keep the backbone fixed. Results are shown in Tables 4 and 5. Specifically, the proposed method outperforms Faster ILOD by 3.0% mAP after learning all 6 stages on VOC. With the COCO dataset, the proposed method achieves a 1.9% mAP gain over Faster ILOD.

Importantly, as shown in Table 6, the proposed method performs better than Faster ILOD not only on the old classes, but also on the newly added classes. Although the overall accuracies are mostly determined by the old classes as a result of the large ratio of old classes, our method demonstrated impressive incremental learning capability. In this experimental setting, large amount of training data for the base classes is enough to train a powerful backbone network. So the backbone network is fixed for faster training in the incremental stages. The proposed RT-Net can achieve similar results with the backbone network finetuned using guided feature distillation, e.g. 15.9% final mAP after learning all the classes. With generative feature replay, the FRCN can be trained in a regular manner for both the old classes and the newly added ones. In addition, guided feature distillation and holistic ranking transfer can transfer the knowledge of old classes to the new model without harming the learning of new classes. On the contrary, vanilla distillation based methods suffer from performance tradeoffs between the old classes and the new ones.

For the set-disjoint setting, we compare the proposed method with ILWCF and CIFRCN. With feature distillation denoted as FD as in [13], we also implement GFD + FD-RPN + KD-FRCN and FD-RPN + GFR-FRCN. We show the experimental results with VOC and COCO in Tables 7 and 8, respectively. Compared with CIFRCN, the proposed method achieves much better performance in terms of average mAP. The mAPs in the final stage are 3.3% and 2.0% higher than those of CIFRCN for VOC and COCO, respectively. For this setting, we cannot use ranking transfer for RPN because we cannot access any past images used in earlier stages. Therefore, the proposed method in this scenario refers to GFD + FD-RPN + GFR-FRCN. Results also show that by combining GFR or GFD with CIFRCN, we can achieve large mAP improvement over the baseline method, which verifies the effectiveness of the two components. Qualitative results with COCO are shown in Fig. 9. Because old images are never used for training in the set-disjoint setting, the baseline methods suffer from catastrophic forgetting. However, the proposed method can effectively detect old classes, such as sheep and boats in stage 2, birds in stage 3 and chairs in stage 4.

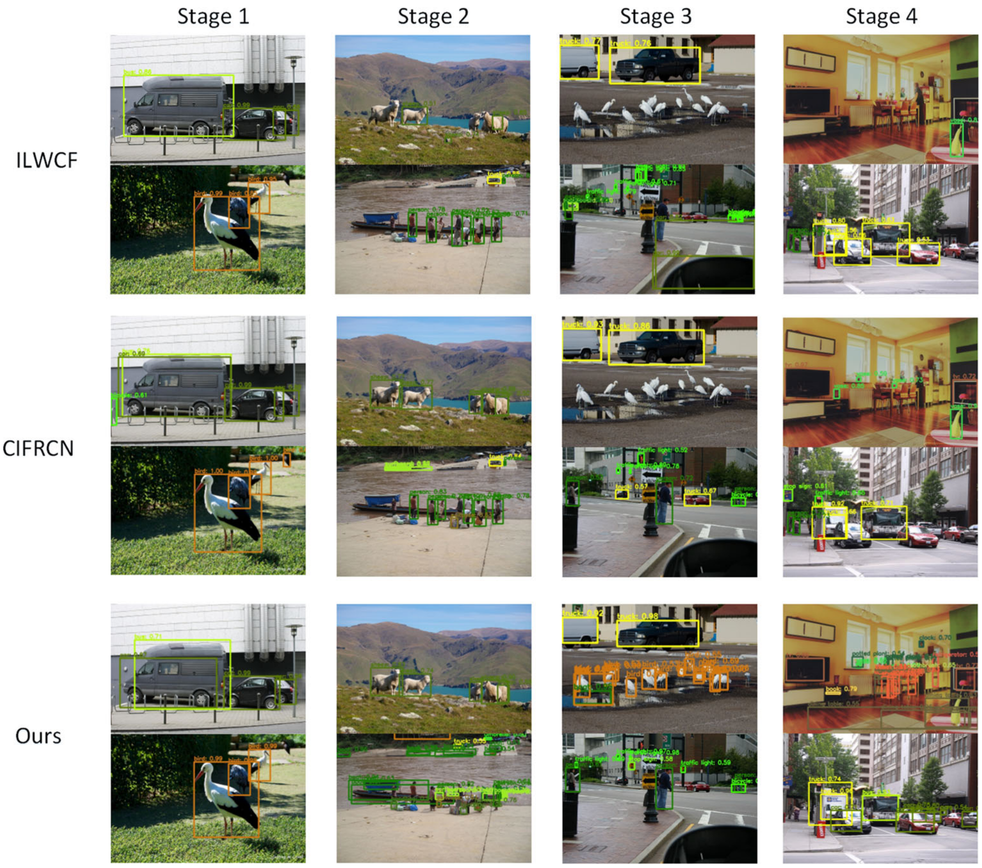

Fig. 9 Results of the 4-stage set-disjoint class incremental object detection on COCO

4.4 Ablation Study

Loss Functions for Feature Embedding The primary contribution of this study is the generative replay framework. The proposed generative replay method samples from the distributions of final feature vectors under the assumption of Gaussian mixture distributions. We thus perform ablation experiments about the choice of loss functions for feature embedding. Specifically, we compare the proposed center loss-based classification loss with Gaussian mixture loss and vanilla softmax loss. Results are shown in Tables 2 and 8, where 0.7% and 1.8% final mAP gain can be achieved by the proposed method compared with Gaussian loss and vanilla softmax loss for the 4-stage set-overlapped incremental detection. These results are consistent for the set-disjoint setting, where 0.4% and 0.8% absolute improvements on mAP are achieved. These results verify the effectiveness of the proposed design of loss functions for final feature embedding.

Feature Transformation Feature transformation is an important component in the proposed framework. As shown in Tables 2 and 8, the mAPs for the set-overlapped setting and set-disjoint setting at the end of learning all 4 stages degrade by 0.9% and 1.1% without feature transformation, respectively. Evolution of the backbone network results in a change in the RoI feature space, although feature distillation is deployed on the backbone network. Therefore, feature transformation is an important component that benefits generative feature replay.

Knowledge Transfer Methods for RPN We compare the proposed holistic information-guided ranking transfer (HRT) with RPN-guided ranking transfer (RRT) and knowledge distillation (KD). As shown in Table 2, compared with the proposed method using HRT, the mAPs of using RRT and GFD + KD-RPN + GFR-FRCN drop 0.3% and 0.6%, respectively, indicating that both ranking transfer and holistic guidance contribute to the overall performance, while holistic guidance is more important.

5 Conclusion

In this study, we developed RT-Net, an effective class-incremental object detector. The key contribution of this study is the design of a novel generative feature replay method that can mitigate the lack of old training data for a faster R-CNN-based incremental object detector. Because feature replay works well when the knowledge stored in the backbone can be preserved, we used guided feature distillation to achieve this. An effective knowledge transfer method for RPN, holistic ranking transfer, was also developed to allow the RPN to distinguish high-quality proposals from lower proposals for old classes. Experiments performed on two incremental object detection benchmarks demonstrate the effectiveness of the proposed framework.

Acknowledgements This work was supported by the National Key Research and Development Program of China (2017YFA0105203), the Strategic Priority Research Program of the Chinese Academy of Sciences (XDB32040200) and Beijing Academy of Artificial Intelligence.

Declarations

Conflict of Interests The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Belouadah E, Popescu A, Kanellos I (2021) A comprehensive study of class incremental learning algorithms for visual tasks. Neural Netw 135:38–54
2. Cai Z, Vasconcelos N (2018) Cascade r-cnn: Delving into high quality object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6154–6162
3. Castro FM, Marín-Jiménez MJ, Guil N, Schmid C, Alahari K (2018) End-to-end incremental learning. In: Proceedings of the European conference on computer vision (ECCV), pp 233–248
4. Chen Y, Wang N, Zhang Z (2018) Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In: Proceedings of the AAAI conference on artificial intelligence, pp 2852–2859
5. Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath AA (2018) Generative adversarial networks: an overview. IEEE Signal Process Mag 35(1):53–65
6. Dai X, Yuan X, Wei X (2021) Tirnet: Object detection in thermal infrared images for autonomous driving. Appl Intell 51(3):1244–1261
7. Delange M, Aljundi R, Masana M, Parisot S, Jia X, Leonardis A, Slabaugh G, Tuytelaars T (2021) A continual learning survey: Defying forgetting in classification tasks. IEEE Trans Pattern Anal Mach Intell
8. Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A (2010) The pascal visual object classes (voc) challenge. Int J Comput Vis 88(2):303–338
9. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp 1440–1448
10. Girshick R, Donahue J, Darrell T, Malik J (2015) Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans Pattern Anal Mach Intell 38(1):142–158
11. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Proceeding of the advances in neural information processing, pp 2672–2680
12. Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J et al (2018) Recent advances in convolutional neural networks. Pattern Recognit 77:354–377
13. Hao Y, Fu Y, Jiang YG, Tian Q (2019) An end-to-end architecture for class-incremental object detection with knowledge

distillation. In: Proceedings of the IEEE international conference on multimedia & expo (ICME), pp 1–6

14. He K, Zhang X, Ren S, Sun J (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans Pattern Anal Mach Intell 37(9):1904–1916
15. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
16. He K, Gkioxari G, Dollár P, Girshick R (2018) Mask r-cnn. IEEE Trans Pattern Anal Mach Intell 42(2):386–397
17. He Z, Ren Z, Yang X, Yang Y, Zhang W (2021) Mead: a mask-guided anchor-free detector for oriented aerial object detection. Appl Intell:1–16
18. Iscen A, Zhang J, Lazebnik S, Schmid C (2020) Memory-efficient incremental learning through feature adaptation. In: Proceedings of the European conference on computer vision (ECCV), pp 699–715
19. Joseph K, Khan S, Khan FS, Balasubramanian VN (2021) Towards open world object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5830–5840
20. Kemker R, Kanan C (2018) Fearnet: Brain-inspired model for incremental learning. In: Proceedings of the international conference on learning representations
21. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A et al (2017) Overcoming catastrophic forgetting in neural networks. Proc Nat Acad Sci USA 114(13):3521–3526
22. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Proceeding of the advances in neural information processing, pp 1097–1105
23. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324
24. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
25. Leng J, Liu Y (2021) Context augmentation for object detection. Appl Intell:1–13
26. Li Z, Hoiem D (2017) Learning without forgetting. IEEE Trans Pattern Anal Mach Intell 40(12):2935–2947
27. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014) Microsoft coco: Common objects in context. In: Proceedings of the European conference on computer vision (ECCV), pp 740–755
28. Lin TY, Goyal P, Girshick R, He K, Dollár P (2018) Focal loss for dense object detection. IEEE Trans Pattern Anal Mach Intell 42(2):318–327
29. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) Ssd: Single shot multibox detector. In: Proceedings of the European conference on computer vision (ECCV), pp 21–37
30. McCloskey M, Cohen NJ (1989) Catastrophic interference in connectionist networks: The sequential learning problem. In: Psychol learn motivat, vol 24, pp 109–165
31. Parisi GI, Kemker R, Part JL, Kanan C, Wermter S (2019) Continual lifelong learning with neural networks: a review. Neural Netw 113:54–71
32. Peng C, Zhao K, Lovell BC (2020) Faster ilod: Incremental learning for object detectors based on faster rcnn. Pattern Recognit Lett 140:109–115
33. Pont-Tuset J, Arbelaez P, Barron JT, Marques F, Malik J (2016) Multiscale combinatorial grouping for image segmentation and object proposal generation. IEEE Trans Pattern Anal Mach Intell 39(1):128–140
34. Ramakrishnan K, Panda R, Fan Q, Henning J, Oliva A, Feris R (2020) Relationship matters: Relation guided knowledge transfer for incremental learning of object detectors. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops
35. Rebuffi SA, Kolesnikov A, Sperl G, Lampert CH (2017) icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2001–2010
36. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
37. Ren S, He K, Girshick R, Sun J (2016a) Faster r-cnn: towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach Intell 39(6):1137–1149
38. Ren S, He K, Girshick R, Zhang X, Sun J (2016b) Object detection networks on convolutional feature maps. IEEE Trans Pattern Anal Mach Intell 39(7):1476–1481
39. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis 115(3):211–252
40. Shin H, Lee JK, Kim J, Kim J (2017) Continual learning with deep generative replay. In: Proceeding of the advances in neural information processing
41. Shmelkov K, Schmid C, Alahari K (2017) Incremental learning of object detectors without catastrophic forgetting. In: Proceedings of the IEEE international conference on computer vision, pp 3400–3409
42. Sun W, Dai L, Zhang X, Chang P, He X (2021) Rsod: Real-time small object detection algorithm in uav-based traffic monitoring. Appl Intell:1–16
43. Tian R, Shi H, Guo B, Zhu L (2021) Multi-scale object detection for high-speed railway clearance intrusion. Appl Intell:1–16
44. Wan W, Zhong Y, Li T, Chen J (2018) Rethinking feature distribution for loss functions in image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 9117–9126
45. Wen Y, Zhang K, Li Z, Qiao Y (2019) A comprehensive study on center loss for deep face recognition. Int J Comput Vis 127(6-7):668–683
46. Wu Y, Chen Y, Wang L, Ye Y, Liu Z, Guo Y, Fu Y (2019) Large scale incremental learning. In: Proceedings of the IEEE Conference on computer vision and pattern recognition, pp 374–382
47. Xia F, Liu TY, Wang J, Zhang W, Li H (2008) Listwise approach to learning to rank: theory and algorithm. In: Proceedings of the international conference on machine learning, pp 1192–1199
48. Xiang Y, Fu Y, Ji P, Huang H (2019) Incremental learning using conditional adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp 6619–6628
49. Zeng G, Chen Y, Cui B, Yu S (2019) Continual learning of context-dependent processing in neural networks. Nat Mach Intell 1(8):364–372
50. Zhu D, Xia S, Zhao J, Zhou Y, Niu Q, Yao R, Chen Y (2021) Spatial hierarchy perception and hard samples metric learning for high-resolution remote sensing image object detection. Appl Intell:1–16
51. Zitnick CL, Dollár P (2014) Edge boxes: Locating object proposals from edges. In: Proceedings of the European conference on computer vision (ECCV), pp 391–405