可借鉴的点：1.在已有模型的基础上增添新class的能力 2.Faster RCNN是基本网络结构 3.可以应用于使用了RPN方法的网络中

Faster ILOD: Incremental Learning for Object Detectors based on Faster RCNN

Can Peng, Kun Zhao, and Brian C. Lovell

School of ITEE, The University of Queensland, Brisbane, QLD, Australia

Abstract

The human vision and perception system is inherently incremental where new knowledge is continually learned over time whilst existing knowledge is retained. On the other hand, deep learning networks are ill-equipped for incremental learning. When a well-trained network is adapted to new categories, its performance on the old categories will dramatically degrade. To address this problem, incremental learning methods have been explored which preserve the old knowledge of deep learning models. However, the state-of-the-art incremental object detector employs an external fixed region proposal method that increases overall computation time and reduces accuracy comparing to Region Proposal Network (RPN) based object detectors such as Faster RCNN. The purpose of this paper is to design an efficient end-to-end incremental object detector using knowledge distillation. We first evaluate and analyze the performance of the RPN-based detector with classic distillation on incremental detection tasks. Then, we introduce multi-network adaptive distillation that properly retains knowledge from the old categories when fine-tuning the model for new task. Experiments on the benchmark datasets, PASCAL VOC and COCO, demonstrate that the proposed incremental detector based on Faster RCNN is more accurate as well as being 13 times faster than the baseline detector.

1 Introduction

问题，当旧模型针对新事物调整参数后，此模型对于原来所对应的事物检测能力下降

Benefiting from the rapid development of deep learning models, the performance of object detectors has increased dramatically over the years. However, the gap between state-of-the-art performance and the human visual system is still huge. One of the main obstacles is incrementally learning new tasks in the dynamic real-world where new categories of interest can emerge over time. For example, in the pathology area, new sub-types of disease patterns are identified over time due to the continued growth in our knowledge and understanding. An ideal disease pattern detection system should be able to learn a new sub-type of disease from the pathology images without losing the ability to detect old disease sub-types. Humans can learn to recognize new categories without forgetting previously learned knowledge. However, when state-of-the-art object detectors are fine-tuned for new tasks, they often fail on the previously trained tasks — a problem called catastrophic forgetting (Goodfellow et al., 2014; McCloskey and Cohen, 1989). Figure 1 shows an example of this problem on the PASCAL VOC dataset (Everingham et al., 2010). The normal training shown in Figure 1 is the conventional way to make a detector work well on all tasks — this requires the model to be trained on labeled data from both the old and new tasks. Unfortunately, this retraining procedure is both time-consuming and computationally expensive. This method also requires access to all of the data for all tasks which is quite impractical in many real-life applications due to a variety of reasons. The old training data may be inaccessible as it may have been lost or corrupted, or perhaps it is simply too large, or there may be licensing or distribution issues.

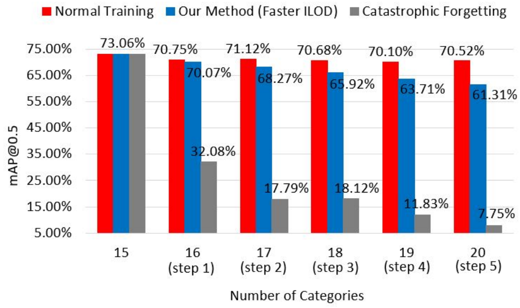


Figure 1: An example of incremental object detection. The model is first trained on 15 categories of the PASCAL VOC dataset followed by the addition of 1 new class in each of 5 steps. During training, only annotations for the current learning classes are provided and all other class objects are ignored. Normal training retrains the model from scratch with all the available data (old and new) which gives the best possible performance. Catastrophic forgetting is what happens when fine-tuning the old class model with just the new class data. Our Faster ILOD method dramatically alleviates the forgetting problem when training on the new class data only.

normal training:trainging with labels both from old and new tasks

In contrast to image classification where the input image only contains one class of objects, an image for object detection can contain multiple classes of objects. Under incremental object detection scenarios, the classes of objects in one image can come from both the new task and the old tasks. In practice, in incremental training of a previously trained detector for a specific new task, only annotations for the new object are provided and other

incremental training: only labels for new task are provided

确认一下missing annotations:是说对于task2，如果含有马的里边也有人，那么他们两个的注释在训练集中就都不会给出？

objects are not annotated which may lead to missing annotations for the previously learned objects. Even if all of the old data is available for normal training, these data need to be re-annotated to contain labels about all classes for all incremental learning tasks learned so far. Figure 2 shows an example of the missing annotation problem. Therefore, compared to incremental classification, incremental detection is a more challenging task, since it not only needs to solve catastrophic forgetting, but also the missing annotations for the old classes in the new data.

To bridge the performance gap between catastrophic forgetting and normal full dataset training, Shmelkov et al. (2017) proposed an incremental object detector using knowledge distillation (Hinton et al., 2015). Their method is based on the largely superseded Fast RCNN (Girshick, 2015) detector which uses an external fixed proposal generator rather than a CNN, so training is not end-to-end. To avoid missing annotation for region proposals, Shmelkov et al. (2017) deliberately chose the external fixed proposal generator of Fast RCNN, so that the proposals would be agnostic to the object categories. The more recent Faster RCNN (Ren et al., 2015) uses a trainable Region Proposal Network (RPN) to boost both accuracy and speed. The RPN-based methods are expected to be fragile to incremental learning, because the unlabeled old class objects are treated as background during retraining of the RPN detector which may adversely affect the RPN proposals on the old classes. In other words, if the retrained RPN is no longer able to detect the old classes by outputting proposals, there is simply no possibility of classifying them.

To address this challenge of incremental learning for RPN-based detectors, we first analyze the capability of RPN on the missing annotation problem for incremental detection. Then, we propose an incremental framework, Faster ILOD, using multi-network adaptive distillation to improve the performance.

The contributions of this paper are as follows:

- We find that due to its unique anchor selection scheme, in the incremental detection scenario, RPN has the capability to tolerate missing annotations for old class objects to some extent.
- Multi-network adaptive distillation is designed to further boost the accuracy of the proposed incremental object detector.
- Using Faster RCNN (Ren et al., 2015) as the fundamental network, we demonstrate the superior performance of our model on both the PASCAL VOC (Everingham et al., 2010) and COCO (Lin et al., 2014) datasets under several incremental detection settings.
- Our framework is generic and can be applied to any object detectors using the RPN.

2 Problem Formulation

Incremental learning for object detection consists of S incremental steps. In each incremental step, only the batch of training data for the new classes (C_n) is accessible. Given an object detection model that is previously trained using images from certain old classes (C_o), incremental object detection is the task of retraining the model to maintain the detection of the old classes (C_o) whilst detecting the new classes (C_n). We refer to the original model as the old model (teacher model) and the retrained model as the new model (student model). In multi-step incremental detection scenarios ($S > 1$), for each step, all categories that were trained during any previous steps are called the old classes. In this paper, we follow the protocol used by Shmelkov et al. (2017). More specifically, in this paper, we target the challenging real-life incremental detection scenarios as follows:

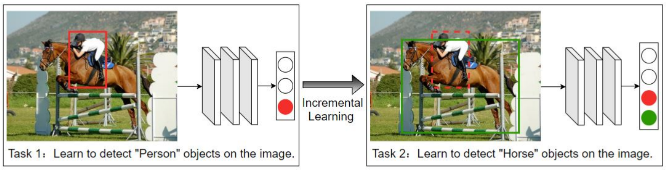

Incremental Learning
Task 1: Learn to detect "Person" objects on the image.
Task 2: Learn to detect "Horse" objects on the image.

Figure 2: An example of the missing annotation problem for incremental object detection (Shmelkov et al., 2017). The model is trained to incrementally learn two tasks: detect human objects and horse objects. First, we label all human objects in the training images for task 1 and use them to train the model. Then, we label all horse objects in the training images for task 2 and use them to train the model. If the training images for task 2 also contain human objects, the labels for them are not provided.

关于新旧训练任务的说明

- In each incremental training step, only training data for the new classes is available; no representative data exemplars of the old classes from previous incremental steps are available.
- Objects from the old classes may appear in the training images of the new detection tasks; however, the annotations for these old class objects are not provided.
- The retrained detector should have the capability to detect objects from both the new classes and the old classes trained in all previous incremental steps.

3 Related Work

Our work focuses on applying Knowledge Distillation (KD) to RPN-based object detectors to improve both speed and accuracy in incremental scenarios. In this section, we introduce the background of KD followed by a discussion about its application to incremental learning scenarios.

3.1 Knowledge Distillation

KD was first introduced by Hinton et al. (2015) for classification model compression. Model compression transfers the knowledge learned from a high performance cumbersome source model to a small target model. The intuition behind KD is that the relative probabilities of incorrect answers can reveal the potential relations between different categories. For example, in handwritten digit recognition, 7 is more likely to be confused with 1 than 8. Thus, during model compression, it is advantageous to train the target model by outputs from the source model instead of ground truth labels. Romero et al. (2015) proposed hint learning to improve the performance of model compression which distills information from feature maps of the source model. Chen et al. (2017) adopted both the distillation method of Hinton et al. (2015) and hint learning of Romero et al. (2015) to detection model compression. Heo et al. (2019) proposed a pre-ReLU feature distillation method to improve the distillation quality for model compression.

3.2 KD based Incremental Learning Method

As the KD method has the capability to transfer the knowledge of one model to another model, it has become one of the most commonly used tools for incremental learning. When applying KD to incremental learning, the old model output for new data is combined with its ground truth information to train the new model. We first discuss related methods for incremental classification followed by methods for incremental detection.

Li and Hoiem (2017) first applied KD to incremental learning and built an incremental classifier called LwF. The LwF method does not require any old data to be stored and uses KD as an additional regularization term on the loss function to force the new model to follow the behavior of the old model on old tasks. Zhou et al. (2019) proposed a multi-model distillation method called M2KD which directly matches the category outputs of the current classification model with those of the corresponding old models. Mask based pruning is used to compress the old models in M2KD. Rebuffi et al. (2017) introduced a KD-based incremental classification method called iCaRL. iCaRL stores some old data by selecting representative exemplars from each of the old classes based on herding. The stored old exemplars and new data are combined to train the new model. However, as only limited exemplars are stored, there is a prediction bias towards the new classes due to data size imbalance between the old and new classes. Castro et al. (2018) kept all final classification layers during incremental learning for distillation to alleviate this data imbalance. Wu et al. (2019a) proposed using a few balanced old and new data batches to train additional two-parameter offset for the model output to remove the bias.

In contrast to classification, research on incremental object detection is quite limited in the literature. Shmelkov et al. (2017) adapted the incremental classification method LwF of (Li and Hoiem, 2017), and proposed an incremental object detector where no old data is available. In this paper, we call their method Incremental Learning Object Detector (ILOD). As ILOD is based on the Fast RCNN (Girshick, 2015) detector, it uses an external proposal generator such as EdgeBox (Zitnick and Dollár, 2014) or MCG (Arbeláez et al., 2014) to generate region proposals. Shmelkov et al. (2017) deliberately chose the external fixed proposal generator of Fast RCNN to ensure that proposals would be agnostic to object categories. In our experiments, we show that our proposed method can perform well on more the efficient RPN-based detectors such as Faster RCNN (Ren et al., 2015).

Hao et al. (2019) proposed an end-to-end incremental object detector. In their experiments, they divided the data classes into multiple class groups and trained their model to incrementally learn the class groups. For both training and testing of each class group, they specifically ignored all images that contain objects from multiple class groups. This process artificially ensures that the training images for the new classes do not contain any old objects and thus avoids the missing annotation problem. However, in real-life applications, there is a high likelihood that the input image may contain objects from both the old and new classes.

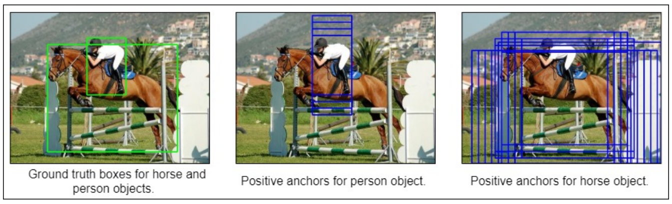
Chen et al. (2019) also had a similar motivation and proposed a distillation method for incremental object detection. However, their proposed method is only evaluated on the VOC dataset over three settings without comparison to the state-of-the-art method. Moreover, the experimental results are not clearly presented with specific accuracy in two settings. Li et al. (2019) proposed a one-stage incremental object detector based on RetinaNet (Lin et al., 2017). In their experiments, they did not mention how they handle the annotations for old classes on new data and they only performed one-step incremental detection on the VOC dataset.

In contrast to the previous works, we target designing a high performance incremental object detector for real-life applications where new task images may also contain objects from the old classes, but the annotations for the old classes are not present. In addition, we perform experiments for both one-step and multi-step incremental learning on two detection benchmark datasets — PASCAL VOC (Everingham et al., 2010) and COCO (Lin et al., 2014).

4 Evaluation of Robustness of RPN to Incremental Object Detection

For supervised learning, correct annotation is very important to guarantee the performance of the trained model. Thus, some previous literature (Hao et al., 2019; Shmelkov et al., 2017) assumed that during incremental training, the missing annotations for the old class objects would adversely affect RPN performance. Based on this assumption, researchers have adopted several methods to avoid this problem. Shmelkov et al. (2017) used the fixed external proposal generator of Fast RCNN to acquire proposals agnostic to categories. Hao et al. (2019) carefully chose the training data to avoid old class objects appearing in new class data.

Before designing our own framework, we first evaluate how the RPN network behaves during incremental learning. To that end, we applied the KD method in ILOD (Shmelkov et al., 2017) to the Faster RCNN (Ren et al.,



Ground truth boxes for horse and person objects.

Positive anchors for person object.

Positive anchors for horse object.

Figure 3: An example of the RPN network working principle.

2015) detector and followed the same training strategy to train the model. The training strategy is described in detail in Section 6. In all of our experiments, ILOD applied to Faster RCNN outperforms the original ILOD method due largely to the underlying superiority of Faster RCNN. More specifically, according to Table 1, for one-step incremental experiments on the VOC dataset, under adding 1, 5 and 10 classes protocol, ILOD applied to Faster RCNN outperforms the original ILOD by 0.45%, 3.63% and 0.11%, respectively.

According to Table 2, for one-step incremental experiments on the COCO dataset, under adding 5, 10 and 40 classes protocol, ILOD applied to Faster RCNN outperforms original ILOD by 4.1%, 3.3% and 3.0%, respectively at 0.5 IoU. According to Figure 5, for multi-step incremental experiments on the VOC dataset, under each time adding 1, 2 and 5 classes protocol, ILOD applied to Faster RCNN outperforms original ILOD by an average 1.32%, 4.11% and 0.79%, respectively. According to Figure 6, for multi-step incremental experiments on the COCO dataset, under each time adding 1 class protocol, ILOD applied to Faster RCNN outperforms the original ILOD by an average 3.2% at 0.5 IoU. In summary, unlike what was assumed by Shmelkov et al. (2017) and Hao et al. (2019), we see that, the performance of Faster RCNN is not greatly affected in the case of incremental learning where the annotations of old classes are not provided in the new data.

Surprisingly, the RPN network is relatively robust towards missing annotations. This behavior has also been observed by Wu et al. (2019b). In their experiments, after dropping 30% of the annotations, the performance of Faster RCNN only decreases by 5% (Wu et al., 2019b). The task of RPN is to take the features of an image as input and output a set of category agnostic possible object proposals. To that end, the RPN network randomly samples 256 anchors in an image to train a binary classifier, where the sampled positive and negative anchors have a ratio of up to 1:1 (Ren et al., 2015). An anchor is labeled as positive if it has the highest IoU overlap with a ground truth bounding box or the IoU overlap with a ground truth box is higher than a threshold (e.g. 0.7). An anchor is labeled as negative if the IoU overlap with any ground truth box is lower than a threshold (e.g. 0.3). Anchors that are neither positive nor negative do not contribute to RPN training. During incremental learning, the ground truth information for the old class objects are not provided. However, during RPN training, the method randomly samples 256 anchors from many thousands of anchors. This means that the risk of a selected negative anchor bounding a well localized old category object is quite low. Note further that a large proportion of the new class images may not contain any old class objects, so their annotations will be perfectly correct. It appears that the main problem of missing annotation is that the number of positive samples used to train the RPN is reduced, since the positive anchors for the old classes are missing.

Figure 3 shows an example to demonstrate the working principle for the RPN network. Three ratios (0.5, 1, 2) and five scales (2, 4, 8, 16, 32) are used for anchor generation. The anchor stride is set to 16. The example image has image size of 480 × 364 pixels, so 10350 anchors are generated. After filtering repeated and excessively small anchors, 8874 anchors are available to be used. With annotations for both the horse and person objects being provided, there are 26 positive anchors and 7307 negative anchors. Assume the task is to incrementally learn to detect horse (new class), and the person category is regarded as the old class. With only ground truth annotation for the horse object (new class) being provided, there are 22 positive (horse) anchors and 7476 negative anchors. Within these 7476 negative anchors, 4 positive anchors containing the person object (old class) are wrongly regarded as negative anchors. The false negative rate is just 0.05%. On the other hand, assume the task is to incrementally learn to detect a person (new class), and the horse category is regarded as the old class. With only ground truth annotation for the person object (new class) being provided, there are 4 positive (person) anchors and 8705 negative anchors. Within these 8705 negative anchors, 22 positive anchors containing the horse object (old class) are wrongly regarded as negative anchors. The false negative rate is now 0.25%.

Offsetting the adverse effects of missing annotation in the ILOD method, although distillation is only applied at the final outputs, the gradients of the distillation loss back-propagate through the entire network and will tend to encourage both the RPN and feature extractor to recognize old classes. This helps explain why RPN training is not catastrophically affected by the missing annotations — at least not over the range of our experiments, such as one and several-step incremental settings.

5 Faster ILOD for Robust Incremental Object Detection

There remains an accuracy gap between the ILOD method applied to RPN-based detectors and full data training. For example, according to Figure 5, for the VOC dataset, the full data training result is 69.50% but under multi-step incremental learning, all the results for ILOD applied to Faster RCNN are less than 60% accuracy. According to Figure 6, for the COCO dataset, the full data training result is 42.71% at 0.5 IoU but under multi-step incremental setting, the result for ILOD applied to Faster RCNN is only 24% at 0.5 IoU. In this section, we propose a novel multi-network adaptive distillation method

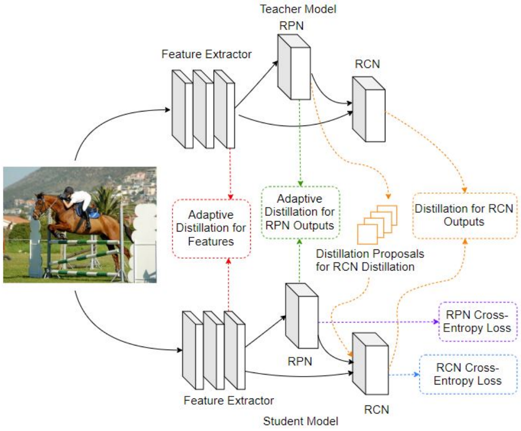

Teacher Model
RPN
Feature Extractor
RCN
Adaptive Distillation for Features
Adaptive Distillation for RPN Outputs
Distillation for RCN Outputs
Distillation Proposals for RCN Distillation
RPN Cross-Entropy Loss
RCN Cross-Entropy Loss
RPN
Feature Extractor
RCN
Student Model

Figure 4: Framework of our proposed method to perform incremental object detection on Faster RCNN (Ren et al., 2015): Adaptive distillation on the feature maps and RPN outputs works together with a conventional distillation component on the final outputs to preserve previous knowledge and alleviate missing annotation problem for the old classes on new data.

to further narrow the gap. We first discuss the backbone network used for our model and then discuss each component of our proposed method.

5.1 Object Detection Network

Our proposed method for incremental object detection is illustrated by Figure 4. It comprises two models: a teacher model (N_{te}) and a student model (N_{st}). The teacher model is a frozen copy of the original detector which detects objects from the old categories ($C_{te} = C_o$). The student model is the adapted model that needs to be trained to detect objects from both the old and new categories ($C_{st} = C_o \cup C_n$). It is also initially a copy of the original detector but the number of outputs in the last layer is increased to predict for the additional new classes. We use Faster RCNN (Ren et al., 2015) as our backbone network. Faster RCNN is a two-stage end-to-end object detector which consists of three parts: (1) A Convolutional Neural Network (CNN) based feature extractor to provide features; (2) a Region Proposal Network (RPN) to produce regions of interest (RoIs); (3) a class-level classification and bounding box regression network (RCN) to generate the final prediction for each of the proposals from RPN (Chen et al., 2017). In order to create a high performance incremental object detector, it is important to properly account for all three components.

5.2 Multi-Network Adaptive Distillation

To make a model remember what it learned before, similar to ILOD (Shmelkov et al., 2017), we adapt knowledge distillation. But unlike ILOD which only performs one-step distillation at the final outputs, we perform multi-network distillation on the feature maps, RPN and RCN outputs. In addition, knowledge distillation is originally developed for model compression which only requires the preservation of learned knowledge. Incremental learning requires not only maintaining learned knowledge, but also learning new knowledge from the new classes. Thus, directly applying distillation loss to force the student model to follow the behavior of the teacher model will simply prevent new data learning. To solve this problem, we propose adaptive distillation which uses the teacher model outputs as a lower bound to adaptively distill old knowledge.

Feature Distillation: The desired feature extractor needs to provide features that are effective for both old and new categories. To build the desired feature extractor, we utilize normalized adaptive distillation with a $\mathcal{L}_1$ loss. Specifically, all feature maps are normalized to obtain the corresponding zero-mean feature maps, $\tilde{F}_{te}$ and $\tilde{F}_{st}$, from the teacher model and student model, respectively. For each activation in the feature map, we then check its value from the student model ($\tilde{f}_{st} \in \tilde{F}_{st}$) with the corresponding value from the teacher model ($\tilde{f}_{te} \in \tilde{F}_{te}$). If the teacher's activation, $\tilde{f}_{te}$, has a higher value, a loss is generated to force the student model to increase its value for this input, since this activation is important for the old classes. On the other hand, if the student's activation, $\tilde{f}_{st}$, has a higher value, the loss is zero since this activation is likely important for the new classes. The feature distillation loss is:

$$\mathcal{L}_{F\_Dist} = \frac{1}{\mathcal{M}} \sum \begin{cases} \left\| \tilde{f}_{te} - \tilde{f}_{st} \right\|_1 , & \text{if } \tilde{f}_{te} > \tilde{f}_{st} \\ 0 , & \text{otherwise} \end{cases} \tag{1}$$

where te and st subscripts refer to teacher and student networks respectively and $\mathcal{M}$ is the total number of activation values in the feature map.

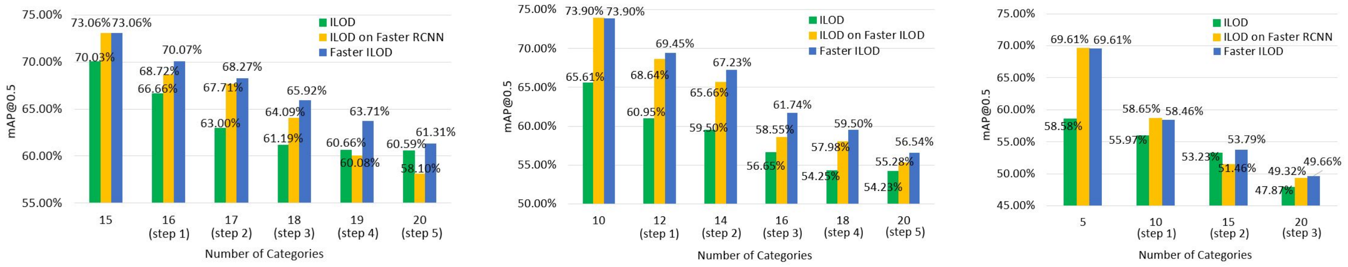
RPN Distillation: The desired RPN needs to provide proposals for objects from both new and old classes. Similar to feature distillation loss, we use the teacher model RPN output as a lower bound to force the student model to choose anchors according to both the training data from the new classes and the teacher model RPN output. In addition, the bounding box regression can provide incorrect values since the real valued regression output is unbounded and partially trained only at positive anchors. Inspired by the distillation method used for detection model compression by Chen et al. (2017), we use a threshold, $\mathcal{T}$, to control regression. In our experiments, empirically we set $\mathcal{T} = 0.1$. For RPN distillation, $\mathcal{L}_2$ loss is used. Suppose $\mathcal{N}$ is the total number of anchors, q is the RPN classification output, and r is the RPN bounding box regression output. The RPN distillation loss is:

$$\mathcal{L}_{RPN\_Dist} = \frac{1}{\mathcal{N}} \sum \begin{cases} \left\| q_{te} - q_{st} \right\|_2^2 + \beta \left\| r_{te} - r_{st} \right\|_2^2 , & \text{if } q_{te} > q_{st} \\ 0 , & \text{otherwise} \end{cases} \tag{2}$$

where

$$\beta = \begin{cases} 1 , & \text{if } q_{te} > (q_{st} + \mathcal{T}) \\ 0 , & \text{otherwise.} \end{cases}$$

RCN Distillation: The desired RCN needs to predict each RoI for both old and new classes in an unbiased manner. We follow the method in ILOD (Shmelkov et al.,

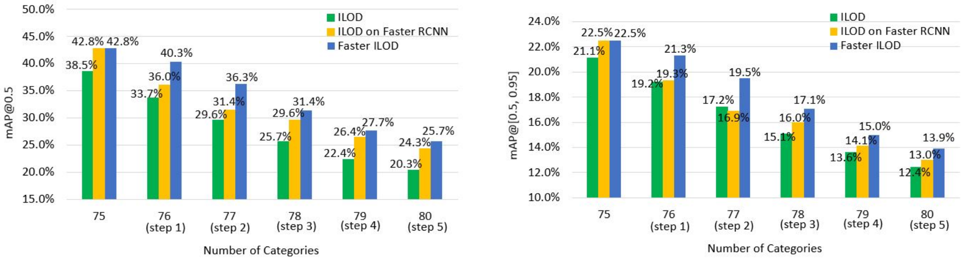


(a) Under the add one new class at a time protocol.

(b) Under the add two new classes at a time protocol.

(c) Under the add five new classes at a time protocol.

Figure 5: Overall mAP accuracy performance for ILOD, ILOD applied to Faster RCNN and Faster ILOD on the VOC dataset under different multi-step incremental scenarios.



(a) mAP at 0.5 IoU.

(b) mAP weighted across different IoU from 0.5 to 0.95.

Figure 6: Overall mAP accuracy performance for ILOD, ILOD applied to Faster RCNN and Faster ILOD on the COCO dataset under the add one new class at a time protocol.

2017) to perform RCN distillation. More specifically, for each image, we randomly choose 64 out of 128 RoIs with the smallest background score according to the RPN output from the teacher model. Then these proposals are fed into the RCN of the student model and the final outputs of the teacher model are used as targets for the old classes. The outputs of student model on the new classes are not included in the RCN distillation. For each RoI classification output, p, we subtract the mean over the class dimension to get the zero-mean classification result, $\tilde{p}$. We use $\mathcal{L}_2$ loss for the distillation. Let $\mathcal{K}$ be the total number of sampled RoIs, C_o be the number of old classes including background, and t be the bounding box regression result. The RCN distillation loss is then written as:

$$\mathcal{L}_{RCN\_Dist} = \frac{1}{\mathcal{K} \times C_o} \sum \left[\left\| \tilde{p}_{te} - \tilde{p}_{st} \right\|_2^2 + \left\| t_{te} - t_{st} \right\|_2^2 \right] \quad (3)$$

Total Loss Function: The overall loss ($\mathcal{L}_{total}$) will be the weighted summation of the standard Faster R-CNN loss (Ren et al., 2015), feature distillation loss (1), RPN distillation loss (2), and RCN distillation loss (3). Hyperparameters λ_1, λ_2 and λ_3 help to balance each loss term, and are empirically set to 1.

$$\mathcal{L}_{total} = \mathcal{L}_{RCNN} + \lambda_1 \mathcal{L}_{F\_Dist} + \lambda_2 \mathcal{L}_{RPN\_Dist} + \lambda_3 \mathcal{L}_{RCN\_Dist} \quad (4)$$

6 Experiments

We call our proposed method Faster ILOD as it is designed to work with Faster RCNN. In this section, we compared our Faster ILOD method with the original ILOD method as well as ILOD applied to Faster RCNN.

6.1 Dataset and Evaluation Metric

In our experiments, two detection benchmark datasets are used, PASCAL VOC 2007 (Everingham et al., 2010) and COCO 2014 (Lin et al., 2014). VOC 2007 comprises 10k images of 20 object categories — 5k for training and 5k for testing. COCO 2014 comprises 164k images of 80 object categories — 83k for training, 40k for validation and 41k for testing. For the evaluation metric, we use mean average precision (mAP) at 0.5 Intersection over Union (IoU) for both datasets and also use mAP weighted across different IoU from 0.5 to 0.95 for COCO. To validate our method, we have investigated several incremental settings for these two datasets, such as one-step and multi-step addition. The sequence of categories is arranged according to the category names in alphabetical order.

6.2 Implementation Details

The results for ILOD (Shmelkov et al., 2017) are generated using their public implementation. Edge-Boxes (Zitnick and Dollár, 2014) is used to generate the external pro-

Table 1: Average mAP for one-step incremental experiments on the VOC dataset. N_{te}(a-b) is the first-step normal training for categories a to b. $+N_{st}$(c) is the incremental learning result after adding c new categories.

	N_{te}(1-19)	$+N_{st}$(1)	N_{te}(1-15)	$+N_{st}$(5)	N_{te}(1-10)	$+N_{st}$(10)	N_{te}(1-20)
ILOD (Shmelkov et al., 2017)	68.60%	67.27%	70.03%	62.72%	65.61%	61.03%	69.50%
ILOD applied to Faster RCNN	70.10%	67.72%	73.06%	66.35%	73.90%	61.14%	70.52%
Faster ILOD w/o feature distillation		68.32%		67.73%		61.78%	
Faster ILOD		**68.56%**		**67.94%**		**62.16%**	

Table 2: Average mAP for one-step incremental experiments on the COCO dataset. The detection accuracy is evaluated by mAP at 0.5 IoU (mAP@.5) and mAP weighted across different IoU from 0.5 to 0.95 (mAP@[0.5, 0.95]).

	N_{te}(1-75)	$+N_{st}$(5)	N_{te}(1-70)	$+N_{st}$(10)	N_{te}(1-40)	$+N_{st}$(40)	N_{te}(1-80)
ILOD (Shmelkov et al., 2017) (mAP@.5)	38.52%	33.80%	38.39%	34.26%	40.01%	36.11%	38.22%
ILOD applied to Faster RCNN (mAP@.5)	42.78%	37.87%	42.95%	37.59%	46.98%	39.10%	42.71%
Faster ILOD (mAP@.5)		**39.60%**		**39.87%**		**40.10%**	
ILOD (Shmelkov et al., 2017) (mAP@[.5, .95])	21.10%	19.19%	21.73%	19.52%	22.69%	19.83%	21.21%
ILOD applied to Faster RCNN (mAP@[.5, .95])	22.45%	20.02%	22.90%	19.90%	24.43%	20.21%	22.52%
Faster ILOD (mAP@[.5, .95])		**21.03%**		**21.28%**		**20.64%**	

Table 3: Average mAP for one-step incremental experiments on the VOC dataset for adding different categories. N_{te}(1-19) is the first-step normal training for 19 categories. $+N_{st}$(c) is the incremental learning result for category c.

	N_{te}(1-19)	$+N_{st}$(TV)	N_{te}(1-19)	$+N_{st}$(aeroplane)	N_{te}(1-19)	$+N_{st}$(person)	N_{te}(1-20)
Amount of training data	4931	256	4806	238	4485	2008	5011
Number of old categories occurring in the training data	-	9	-	6	-	19	-
Percentage of missing annotation	-	60.40%	-	22.22%	-	40.87%	-
ILOD applied to Faster RCNN	70.10%	67.72%	70.32%	67.62%	70.08%	65.79%	70.52%
Faster ILOD		**68.56%**		**68.92%**		**69.57%**	

posals. Faster ILOD and ILOD applied to Faster RCNN are implemented using PyTorch. For a fair comparison of our approach with ILOD (Shmelkov et al., 2017), we use the same backbone network (ResNet-50 (He et al., 2016)) and similar training strategy mentioned in their paper. In the first step of training, we set the learning rate to 0.001, decaying to 0.0001 after 30k iterations, and momentum is set to 0.9. The network is trained using 40k iterations for VOC and 400k iterations for COCO. In the following incremental steps, learning rate is set to 0.0001. The network is trained using 5k-10k iterations when only one class is added and the same number of iterations as the first step if multiple classes are added at once.

6.3 Experiments on VOC Dataset

Table 1 shows the results for one-step incremental settings when the number of new classes equals 1, 5 and 10, respectively. In all three settings, Faster ILOD is more accurate than both ILOD and ILOD applied to Faster RCNN. Under these settings, we also performed experiments on Faster ILOD without adaptive feature distillation to prove the efficiency of each part of our method. Compared to the experimental results for multi-step increments, we see the improvement is not significant in the one-step settings. This is likely because one-step increments require a small amount of fine-tuning on the old model and the catastrophic forgetting and missing annotation problems might not be significant, which provides little room for improvement. When we retrain the old model in the multiple incremental steps, the build-up of detection errors due to catastrophic forgetting and missing annotation are approximately exponential on the old classes. So this is a more difficult scenario.

We have also investigated the results under multi-step incremental scenarios. Figure 5(a) shows the performance of Faster ILOD, ILOD and ILOD applied to Faster RCNN, when first training with 15 classes followed by the addition of 1 class for 5 steps. Observing from Figure 5(a), under the add one new class at a time protocol, Faster ILOD outperforms ILOD and ILOD applied to Faster RCNN in each incremental step and the average performance gain is 3.44% and 2.12% respectively. Figure 5(b) shows the performance of three models under the condition of first training with 10 classes followed by addition of 2 classes for 5 times. Under this incremental setting, Faster ILOD also performs best for all five incremental steps and the average accuracy improvement is 5.78% towards ILOD and 1.67% towards ILOD applied to Faster RCNN. Figure 5(c) shows the performance of three models under the condition of first training with 5 classes followed by addition of 5 classes for 3 times. Under this incremental scenario, Faster ILOD outperforms ILOD applied to Faster RCNN except the first step and always has better accuracy than ILOD. The average accuracy increase is 1.61% towards ILOD and 0.83% towards ILOD applied to Faster RCNN.

6.4 Experiments on COCO Dataset

For our experiments on the COCO dataset, we use train set and valminusminival set as our training data and minival set as our testing data. Table 2 shows the results under one-step incremental settings, where the number of new classes is 5, 10 and 40, respectively. Figure 6 shows the results for multi-step incremental detection under the add one new class at a time protocol. In both scenarios, Faster ILOD provides the best detection accuracy. In particular, under multi-step incremental detection shown in Figure 6, Faster ILOD outperforms ILOD and ILOD applied to Faster RCNN in all steps and has an average gain of 5.92% and 2.74% (1.88% and 1.53%) respectively at 0.5 IoU (weighted across different IoU from 0.5 to 0.95).

6.5 Discussions

To explore how learning different categories affects the model performance on the same incremental setting, under the one-step incremental learning scenario, we performed the experiments on adding different categories on the VOC dataset. These experiments indicate that our method can always achieve improvements in different learning sequence. Table 3 shows the results for incrementally learning the new class (TV and aeroplane). It also shows the results for incrementally learning the more challenging person category. Comparing with other categories, incrementally learning person category is more challenging since 'person' objects often appear with many other category objects in one image. In VOC training data, there are 2008 out of 5011 images containing person objects. Within the 2008 training images, there are 3641 other objects (40.87% missing annotation) which belong to 19 old categories. As for person category, the amount of training data is large, the model is trained for 40,000 instead of 10,000 iterations for convergence. For all three different categories, our proposed Faster ILOD outperforms ILOD applied to Faster RCNN. According to Table 3, the improvement of our adaptive distillation method for person category is higher than aeroplane and TV categories which have fewer missing annotations (55 and 408, respectively) and fewer training data (238 and 256, respectively). Although the missing annotation percentage for TV category is high (60.40%), it only comes from 9 old categories.

In summary, for different categories, they will face different difficulty levels for incremental learning. We conjecture that the difficulty levels are related to three items: (1) the amount of training data; (2) the number of old categories co-occurring in the new training data; (3) the percentage of missing annotations in the training data. Therefore, the challenge of incremental learning is different under different incremental learning scenarios which leads to different accuracy improvements of our method.

6.6 Evaluations on Detection Speed

As the original ILOD code is built in Tensorflow, to fairly compare the detection speeds for ILOD and Faster ILOD, we rebuilt ILOD on Pytorch. All experiments were performed on an NVIDIA Tesla V100 GPU. Average detection time per image of ILOD and Faster ILOD with ResNet-50 (He et al., 2016) on the VOC dataset is 1396.66 ms and 109.52 ms respectively. As ILOD relies on an external proposal generator to acquire proposals, the inference speed of ILOD is about 13 times slower than Faster ILOD due to the fixed proposal network.

7 Conclusion

In this paper, we find that the RPN network is relatively robust towards missing annotations for old classes on incremental object detection and then propose a novel end-to-end framework, Faster ILOD. By adaptively distilling the old information in multi-networks, the proposed method aims to preserve the capabilities of the detector on old classes with limited affect towards the learning on new classes. Our method shows superior results on the PASCAL VOC and COCO datasets and outperforms the state-of-the-art incremental detector (Shmelkov et al., 2017) by a large margin in most cases.

Acknowledgments

This research was funded by the Australian Government through the Australian Research Council and Sullivan Nicolaides Pathology under Linkage Project LP160101797.

References

Arbeláez, P., Pont-Tuset, J., Barron, J.T., Marques, F., Malik, J., 2014. Multiscale combinatorial grouping, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 328–335.

Castro, F.M., Marín-Jiménez, M.J., Guil, N., Schmid, C., Alahari, K., 2018. End-to-end incremental learning, in: Proceedings of European Conference on Computer Vision, pp. 233–248.

Chen, G., Choi, W., Yu, X., Han, T., Chandraker, M., 2017. Learning efficient object detection models with knowledge distillation, in: Advances in Neural Information Processing Systems, pp. 742–751.

Chen, L., Yu, C., Chen, L., 2019. A new knowledge distillation for incremental object detection, in: 2019 International Joint Conference on Neural Networks (IJCNN), IEEE. pp. 1–7.

Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A., 2010. The pascal visual object classes (voc) challenge. International Journal of Computer Vision 88, 303–338.

Girshick, R., 2015. Fast r-cnn, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448.

Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y., 2014. An empirical investigation of catastrophic forgetting in gradient-based neural networks, in: Proceedings of International Conference on Learning Representations.

Hao, Y., Fu, Y., Jiang, Y.G., Tian, Q., 2019. An end-to-end architecture for class-incremental object detection with knowledge distillation. 2019 IEEE International Conference on Multimedia and Expo (ICME) , 1–6.

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.

Heo, B., Kim, J., Yun, S., Park, H., Kwak, N., Choi, J.Y., 2019. A comprehensive overhaul of feature distillation, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 1921–1930.

Hinton, G., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network, in: NIPS Deep Learning and Representation Learning Workshop. URL: `http://arxiv.org/abs/1503.02531`.

Li, D., Tasci, S., Ghosh, S., Zhu, J., Zhang, J., Heck, L., 2019. Rilod: near real-time incremental learning for object detection at the edge, in: Proceedings of the 4th ACM/IEEE Symposium on Edge Computing, pp. 113–126.

Li, Z., Hoiem, D., 2017. Learning without forgetting. IEEE transactions on Pattern Analysis and Machine Intelligence 40, 2935–2947.

Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2017. Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, pp. 2980–2988.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context, in: European Conference on Computer Vision, Springer. pp. 740–755.

McCloskey, M., Cohen, N.J., 1989. Catastrophic interference in connectionist networks: The sequential learning problem, in: Psychology of Learning and Motivation. Elsevier. volume 24, pp. 109–165.

Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H., 2017. icarl: Incremental classifier and representation learning, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 2001–2010.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, pp. 91–99.

Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y., 2015. Fitnets: Hints for thin deep nets. International Conference on Learning Representations , 1–13.

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.

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y., 2019a. Large scale incremental learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 374–382.

Wu, Z., Bodla, N., Singh, B., Najibi, M., Chellappa, R., Davis, L.S., 2019b. Soft sampling for robust object detection. The British Machine Vision Conference (BMVC) .

Zhou, P., Mai, L., Zhang, J., Xu, N., Wu, Z., Davis, L.S., 2019. M2kd: Multi-model and multi-level knowledge distillation for incremental learning .

Zitnick, C.L., Dollár, P., 2014. Edge boxes: Locating object proposals from edges, in: Proceedings of European Conference on Computer Vision, Springer. pp. 391–405.