Few-shot Object Detection via Feature Reweighting

Bingyi Kang[1]*, Zhuang Liu[2]*, Xin Wang[2], Fisher Yu[2], Jiashi Feng[1], Trevor Darrell[2]
[1]National University of Singapore [2]University of California, Berkeley

Abstract

Conventional training of a deep CNN based object detector demands a large number of bounding box annotations, which may be unavailable for rare categories. In this work we develop a few-shot *object detector that can learn to detect novel objects from only a few annotated examples. Our proposed model leverages fully labeled base classes and quickly adapts to novel classes, using a meta feature learner and a reweighting module within a one-stage detection architecture. The feature learner extracts meta features that are generalizable to detect novel object classes, using training data from base classes with sufficient samples. The reweighting module transforms a few support examples from the novel classes to a global vector that indicates the importance or relevance of meta features for detecting the corresponding objects. These two modules, together with a detection prediction module, are trained end-to-end based on an episodic few-shot learning scheme and a carefully designed loss function. Through extensive experiments we demonstrate that our model outperforms well-established baselines by a large margin for few-shot object detection, on multiple datasets and settings. We also present analysis on various aspects of our proposed model, aiming to provide some inspiration for future few-shot detection works.*

1. Introduction

The recent success of deep convolutional neural networks (CNNs) in object detection [32, 15, 30, 31] relies heavily on a huge amount of training data with accurate bounding box annotations. When the labeled data are scarce, CNNs can severely overfit and fail to generalize. In contrast, humans exhibit strong performance in such tasks: children can learn to detect a novel object quickly from very few given examples. Such ability of learning to detect from few examples is also desired for computer vision systems, since some object categories naturally have scarce examples or their annotations are hard to obtain, e.g., California firetrucks, endangered animals or certain medical data [33].

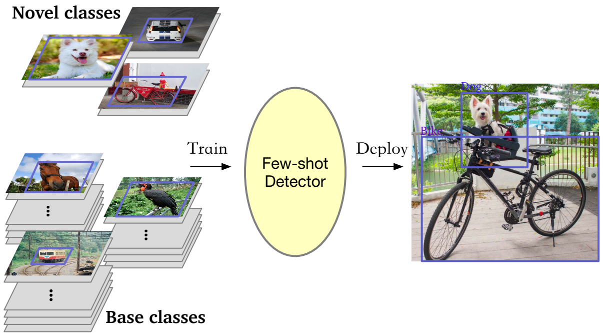


Figure 1: We aim to obtain a few-shot detection model by training on the base classes with sufficient examples, such that the model can learn from a few annotated examples to detect novel objects on testing images.

In this work, we target at the challenging *few-shot object detection* problem, as shown in Fig. 1. Specifically, given some base classes with sufficient examples and some novel classes with only a few samples, we aim to obtain a model that can detect both base and novel objects at test time. Obtaining such a few-shot detection model would be useful for many applications. Yet, effective methods are still absent. Recently, meta learning [39, 35, 12] offers promising solutions to a similar problem, *i.e.*, few-shot classification. However, object detection is by nature much more difficult as it involves not only class predictions but also localization of the objects, thus off-the-shelf few-shot classification methods cannot be directly applied on the few-shot detection problem. Taking Matching Networks [39] and Prototypical Networks [35] as examples, it is unclear how to build object prototypes for matching and localization, because there may be distracting objects of irrelevant classes within the image or no targeted objects at all.

We propose a novel detection model that offers few-shot learning ability through fully exploiting detection training data from some base classes and quickly adapting the detection prediction network to predict novel classes according to a few support examples. The proposed model first learns meta features from base classes that are generalizable to the detection of different object classes. Then it effectively utilizes a few support examples to identify the meta

*Equal contribution.

features that are important and discriminative for detecting novel classes, and adapts accordingly to transfer detection knowledge from the base classes to the novel ones.

Our proposed model thus introduces a novel detection framework containing two modules, *i.e*., a meta feature learner and a light-weight feature reweighting module. Given a query image and a few support images for novel classes, the feature learner extracts meta features from the query image. The reweighting module learns to capture global features of the support images and embeds them into reweighting coefficients to modulate the query image meta features. As such, the query meta features effectively receive the support information and are adapted to be suitable for novel object detection. Then the adapted meta features are fed into a detection prediction module to predict classes and bounding boxes for novel objects in the query (Fig. 2). In particular, if there are N novel classes to detect, the reweighting module would take in N classes of support examples and transform them into N reweighting vectors, each responsible for detecting novel objects from the corresponding class. With such class-specific reweighting vectors, some important and discriminative meta features for a novel class would be identified and contribute more to the detection decision, and the whole detection framework can learn to detect novel classes efficiently.

The meta feature learner and the reweighting module are trained together with the detection prediction module end-to-end. To ensure few-shot generalization ability, the whole few-shot detection model is trained using an two-phase learning scheme: first learn meta features and good reweighting module from base classes; then fine-tune the detection model to adapt to novel classes. For handling difficulties in detection learning (e.g., existence of distracting objects), it introduces a carefully designed loss function.

Our proposed few-shot detector outperforms competitive baseline methods on multiple datasets and in various settings. Besides, it also demonstrates good transferability from one dataset to another different one. Our contributions can be summarized as follows:

- We are among the first to study the problem of few-shot object detection, which is of great practical values but a less explored task than image classification in the few-shot learning literature.
- We design a novel few-shot detection model that 1) learns generalizable meta features; and 2) automatically reweights the features for novel class detection by producing class-specific activating coefficients from a few support samples.
- We experimentally show that our model outperforms baseline methods by a large margin, especially when the number of labels is extremely low. Our model adapts to novel classes significantly faster.

2. Related Work

General object detection. Deep CNN based object detectors can be divided into two categories: proposal-based and proposal-free. RCNN series [15, 14, 32] detectors fall into the first category. RCNN [15] uses pre-trained CNNs to classify the region proposals generated by selective search [38]. SPP-Net [17] and Fast-RCNN [14] improve RCNN with an RoI pooling layer to extract regional features from the convolutional feature maps directly. Faster-RCNN [32] introduces a region-proposal-network (RPN) to improve the efficiency of generating proposals. In contrast, YOLO [29] provides a proposal-free framework, which uses a single convolutional network to directly perform class and bounding box predictions. SSD [22] improves YOLO by using default boxes (anchors) to adjust to various object shapes. YOLOv2 [30] improves YOLO with a series of techniques, e.g., multi-scale training, new network architecture (DarkNet-19). Compared with proposal-based methods, proposal-free methods do not require a per-region classifier, thus are conceptually simpler and significantly faster. Our few-shot detector is built on the YOLOv2 architecture.

Few-shot learning. Few-shot learning refers to learning from just a few training examples per class. Li *et al*. [20] use Bayesian inference to generalize knowledge from a pre-trained model to perform one-shot learning. Lake *et al*. [19] propose a Hierarchical Bayesian one-shot learning system that exploits compositionality and causality. Luo *et al*. [23] consider the problem of adapting to novel classes in a new domain. Douze *et al*. [9] assume abundant unlabeled images and adopts label propagation in a semi-supervised setting.

An increasingly popular solution for few-shot learning is meta-learning, which can further be divided into three categories: a) Metric learning based [18, 37, 39, 35]. In particular, Matching Networks [39] learn the task of finding the most similar class for the target image among a small set of labeled images. Prototypical Networks [35] extend Matching Networks by producing a linear classifier instead of weighted nearest neighbor for each class. Relation Networks [37] learn a distance metric to compare the target image to a few labeled images. b) Optimization for fast adaptation. Ravi and Larochelle [28] propose an LSTM meta-learner that is trained to quickly converge a learner classifier in new few-shot tasks. Model-Agnostic Meta-Learning (MAML) [12] optimizes a task-agnostic network so that a few gradient updates on its parameters would lead to good performance on new few-shot tasks. c) Parameter prediction. Learnet [2] dynamically learns the parameters of factorized weight layers based on a single example of each class to realize one-shot learning.

Above methods are developed to recognize novel images only, there are some other works tried to learn a model that

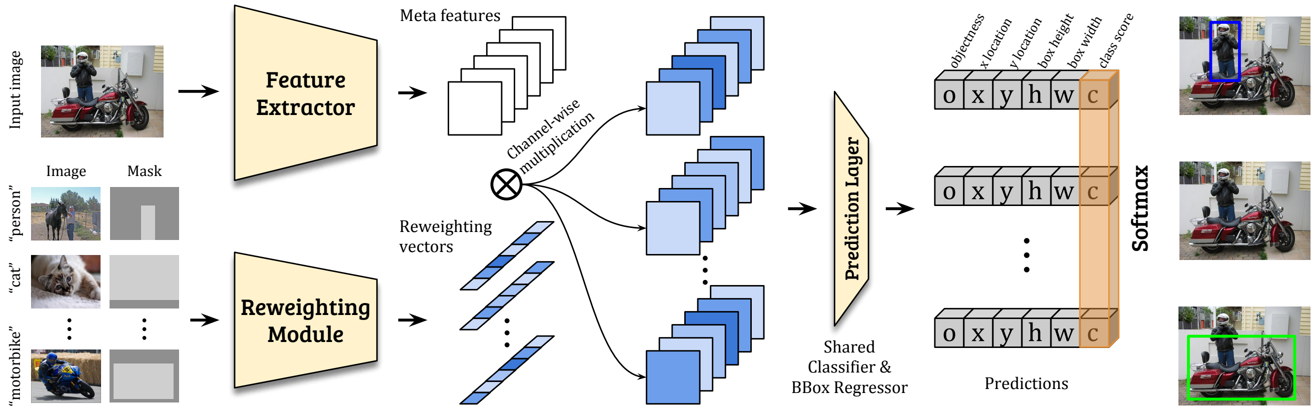


Figure 2: The architecture of our proposed few-shot detection model. It consists of a meta feature extractor and a reweighting module. The feature extractor follows the one-stage detector architecture and directly regresses the objectness score (o), bounding box location (x, y, h, w) and classification score (c). The reweighting module is trained to map support samples of N classes to N reweighting vectors, each responsible for modulating the meta features to detect the objects from the corresponding class. A softmax based classification score normalization is imposed on the final output.

can classify both base and novel images. Recent works by Hariharan *et al.* [16, 40] introduce image hallucination techniques to augment the novel training data such that novel classes and base classes are balanced to some extend. Weight imprinting [26] sets weights for a new category using a scaled embedding of labeled examples. DynamicNet [13] learns a weight generator to classification weights for a specific category given the corresponding labeled images. These previous works only tackle image classification task, while our work focuses on object detection.

Object detection with limited labels. There are a number of prior works on detection focusing on settings with limited labels. The weakly-supervised setting [3, 7, 36] considers the problem of training object detectors with only image-level labels, but without bounding box annotations, which are more expensive to obtain. Few example object detection [25, 41, 8] assumes only a few labeled bounding boxes per class, but relies on abundant unlabeled images to generate trustworthy pseudo annotations for training. Zero-shot object detection [1, 27, 42] aims to detect previously unseen object categories, thus usually requires external information such as relations between classes. Different from these settings, our few-shot detector uses very few bounding box annotations (1-10) for each novel class, without the need for unlabeled images or external knowledge. Chen *et al.* [4] study a similar setting but only in a transfer learning context, where the target domain images only contains novel classes without base classes.

3. Approach

In this work, we define a novel and realistic setting for few-shot object detection, in which there are two kinds of data available for training, *i.e.*, the *base classes* and the *novel classes*. For the base classes, abundant annotated data are available, while only a few labeled samples are given to the novel classes [16]. We aim to obtain a few-shot detection model that can learn to detect novel object when there are both base and novel classes in testing by leveraging knowledge from the base classes.

This setting is worth exploring since it aligns well with a practical situation—one may expect to deploy a pre-trained detector for new classes with only a few labeled samples. More specifically, large-scale object detection datasets (e.g., PSACAL VOC, MSCOCO) are available to pre-train a detection model. However, the number of object categories therein is quite limited, especially compared to the vast object categories in real world. Thus, solving this few-shot object detection problem is heavily desired.

3.1. Feature Reweighting for Detection

Our proposed few-shot detection model introduces a meta feature learner $\mathcal{D}$ and a reweighting module $\mathcal{M}$ into a one-stage detection framework. In this work, we adopt the proposal-free detection framework YOLOv2 [30]. It directly regresses features for each anchor to detection relevant outputs including classification score and object bounding box coordinates through a detection prediction module $\mathcal{P}$. As shown in Fig. 2, we adopt the backbone of YOLOv2 (*i.e.*, DarkNet-19) to implement the meta feature extractor $\mathcal{D}$, and follow the same anchor setting as YOLOv2. As for the reweighting module $\mathcal{M}$, we carefully design it to be a light-weight CNN for both enhancing efficiency and easing its learning. Its architecture details are deferred to the supplementary due to space limit.

The meta feature learner $\mathcal{D}$ learns how to extract meta features for the input query images to detect their novel objects. The reweighting module $\mathcal{M}$, taking the support examples as input, learns to embed support information into reweighting vectors and adjust contribution of each meta feature of the query image accordingly for following detection prediction module $\mathcal{P}$. With the reweighting module , some meta features informative for detecting novel objects would be excited and thus assist detection prediction.

Formally, let I denote an input query image. Its corresponding meta features $F \in \mathbb{R}^{w \times h \times m}$ are generated by $\mathcal{D}$: $F = \mathcal{D}(I)$. The produced meta feature has m feature maps. We denote the support images and their associated bounding box annotation, indicating the target class to detect, as I_i and M_i respectively, for class $i, i = 1, \ldots, N$. The reweighting module $\mathcal{M}$ takes one support image (I_i, M_i) as input and embed it into a class-specific representation $w_i \in \mathbb{R}^m$ with $w_i = \mathcal{M}(I_i, M_i)$. Such embedding captures global representation of the target object w.r.t. the m meta features. It will be responsible for reweighting the meta features and highlighting more important and relevant ones to detect the target object from class i. More specifically, after obtaining the class-specific reweighting coefficients w_i, our model applies it to obtain the class-specific feature F_i for novel class i by:

$$F_i = F \otimes w_i, \quad i = 1, \ldots, N, \tag{1}$$

where $\otimes$ denotes channel-wise multiplication. We implement it through 1×1 depth-wise convolution.

After acquiring class-specific features F_i, we feed them into the prediction module $\mathcal{P}$ to regress the objectness score o, bounding box location offsets (x, y, h, w), and classification score c_i for each of a set of predefined anchors:

$$\{o_i, x_i, y_i, h_i, w_i, c_i\} = \mathcal{P}(F_i), \quad i = 1, \ldots, N, \tag{2}$$

where c_i is one-versus-all classification score indicating the probability of the corresponding object belongs to class i.

3.2. Learning Scheme

It is not straightforward to learn a good meta feature learner $\mathcal{D}$ and reweighting module $\mathcal{M}$ from the base classes such that they can produce generalizable meta features and rweighting coefficients. To ensure the model generalization performance from few examples, we develop a new two-phase learning scheme that is different from the conventional ones for detection model training.

We reorganize the training images with annotations from the base classes into multiple few-shot detection learning tasks $\mathcal{T}_j$. Each task $\mathcal{T}_j = \mathcal{S}_j \cup \mathcal{Q}_j = \{(I_1^j, M_1^j), \ldots, (I_N^j, M_N^j)\} \cup \{(I_j^q, M_j^q)\}$ contains a support set S_j (consisting of N support images each of which is from a different base class) and a query set $\mathcal{Q}_j$ (offering query images with annotations for performance evaluation).

Let θ_D, θ_M and θ_P denote the parameters of meta feature learner $\mathcal{D}$, the reweighting module $\mathcal{M}$ and prediction module $\mathcal{P}$ respectively. We optimize them jointly through minimizing the following loss:

$$\begin{aligned}\min_{\theta_D, \theta_M, \theta_P} \mathcal{L} &:= \sum_j \mathcal{L}(\mathcal{T}_j) \\ &= \sum_j \mathcal{L}_{\text{det}}(\mathcal{P}_{\theta_P}(\mathcal{D}_{\theta_D}(I_q^j) \otimes \mathcal{M}_{\theta_M}(\mathcal{S}_j)), M_j^q).\end{aligned}$$

Here $\mathcal{L}_{\text{det}}$ is the detection loss function and we explain its details later. The above optimization ensures the model to learn good meta features for the query and reweighting coefficients for the support.

The overall learning procedure consists of two phases. The first phase is the *base training* phase. In this phase, despite abundant labels are available for each base class, we still jointly train the feature learner, detection prediction together with the reweighting module . This is to make them coordinate in a desired way: the model needs to learn to detect objects of interest by referring to a good reweighting vector. The second phase is *few-shot fine-tuning*. In this phase, we train the model on both base and novel classes. As only k labeled bounding boxes are available for the novel classes, to balance between samples from the base and novel classes, we also include k boxes for each base class. The training procedure is the same as the first phase, except that it takes significantly fewer iterations for the model to converge.

In both training phases, the reweighting coefficients depend on the input pairs of (support image, bounding box) that are randomly sampled from the available data per iteration. After few-shot fine-tuning, we would like to obtain a detection model that can directly perform detection without requiring any support input. This is achieved by setting the reweighting vector for a target class to the average one predicted by the model after taking the k-shot samples as input. After this, the reweighting module can be completely removed during inference. Therefore, our model adds negligible extra model parameters to the original detector

Detection loss function. To train the few-shot detection model, we need to carefully choose the loss functions in particular for the class prediction branch, as the sample number is very few. Given that the predictions are made classwisely, it seems natural to use binary cross-entropy loss, regressing 1 if the object is the target class and 0 otherwise. However, we found using this loss function gave a model prone to outputting redundant detection results (e.g., detecting a train as a bus and a car). This is due to that for a specific region of interest, only one out of N classes is truly positive. However, the binary loss strives to produce balanced positive and negative predictions. Non-maximum suppression could not help remove such false positives as it only operates on predictions within each class.

To resolve this issue, our proposed model adopts a softmax layer for calibrating the classification scores among different classes, and adaptively lower detection scores for the wrong classes. Therefore, the actual classification score for the i-th class is given by $\hat{c}_i = \frac{e^{c_i}}{\sum_{j=1}^{N} e^{c_j}}$. Then to better align training procedure and few-shot detection, the cross-entropy loss over the calibrated scores $\hat{c}_i$ is adopted:

$$\mathcal{L}_c = -\sum_{i=1}^{N} \mathbb{1}(\cdot, i) \log(\hat{c}_i), \tag{3}$$

where $\mathbb{1}(\cdot, i)$ is an indicator function for whether current anchor box really belongs to class i or not. After introducing softmax, the summation of classification scores for a specific anchor is equal to 1, and less probable class predictions will be suppressed. This softmax loss will be shown to be superior to binary loss in the following experiments. For bounding box and objectiveness regression, we adopt the similar loss function $\mathcal{L}_{bbx}$ and $\mathcal{L}_{obj}$ as YOLOv2 [30] but we balance the positive and negative by not computing some loss from negatives samples for the objectiveness scores. Thus, the overall detection loss function is $\mathcal{L}_{\text{det}} = \mathcal{L}_c + \mathcal{L}_{bbx} + \mathcal{L}_{obj}$.

Reweighting module input. The input of the reweighting module should be the object of interest. However, in object detection task, one image may contain multiple objects from different classes. To let the reweighting module know what the target class is, in additional to three RGB channels, we include an additional "mask" channel (M_i) that has only binary values: on the position within the bounding box of an object of interest, the value is 1, otherwise it is 0 (see left-bottom of Fig. 2). If multiple target objects are present on the image, only one object is used. This additional mask channel gives the reweighting module the knowledge of what part of the image's information it should use, and what part should be considered as "background". Combining mask and image as input not only provides class information of the object of interest but also the location information (indicated by the mask) useful for detection. In the experiments, we also investigate other input forms.

4. Experiments

In this section, we evaluate our model and compare it with various baselines, to show our model can learn to detect novel objects significantly faster and more accurately. We use YOLOv2 [30] as the base detector. Due to space limit, we defer all the model architecture and implementation details to the supplementary material. The code to reproduce the results will be released at https://github.com/bingykang/Fewshot_Detection.

4.1. Experimental Setup

Datasets. We evaluate our model for few-shot detection on the widely-used object detection benchmarks, i.e., VOC 2007 [11], VOC 2012 [10], and MS-COCO [21]. We follow the common practice [30, 32, 34, 6] and use VOC 07 test set for testing while use VOC 07 and 12 train/val sets for training. Out of its 20 object categories, we randomly select 5 classes as the novel ones, while keep the remaining 15 ones as the base. We evaluate with 3 different base/novel splits. During base training, only annotations of the base classes are given. For few-shot fine-tuning, we use a very small set of training images to ensure that each class of objects only has k annotated bounding boxes, where k equals 1, 2, 3, 5 and 10. Similarly, on the MS-COCO dataset, we use 5000 images from the validation set for evaluation, and the rest images in train/val sets for training. Out of its 80 object classes, we select 20 classes overlapped with VOC as novel classes, and the remaining 60 classes as the base classes. We also consider learning the model on the 60 base classes from COCO and applying it to detect the 20 novel objects in PASCAL VOC. This setting features a cross-dataset learning problem that we denote as *COCO to PASCAL*.

Note the testing images may contain distracting base classes (which are not targeted classes to detect) and some images do not contain objects of the targeted novel class. This makes the few-shot detection further challenging.

Baselines. We compare our model with five competitive baselines. Three of them are built upon the vanilla YOLOv2 detector with straightforward few-shot learning strategies. The first one is to train the detector on images from the base and novel classes together. In this way, it can learn good features from the base classes that are applicable for detecting novel classes. We term this baseline as *YOLO-joint*. We train this baseline model with the same total iterations as ours. The other two YOLO-based baselines also use two training phases as ours. In particular, they train the original YOLOv2 model with the same base training phase as ours; for the few-shot fine-tuning phase, one fine-tunes the model with the same iterations as ours, giving the *YOLO-ft* baseline; and one trains the model to fully converge, giving *YOLO-ft-full*. Comparing with these baselines can help understand the few-shot learning advantage of our models brought by the proposed feature reweighting method. The last two baselines are from a recent few-shot detection method, i.e., Low-Shot Transfer Detector (LSTD) [4]. LSTD relies on background depression (BD) and transfer knowledge (TK) to obtain a few-shot detection model on the novel classes. For fair comparison, we re-implement BD and TK based on YOLOV2, train it for the same iterations as ours, obtaining *LSTD(YOLO)*; and train it to convergence to obtain the last baseline, *LSTD(YOLO)-full*.

4.2. Comparison with Baselines

PASCAL VOC. We present our main results on novel classes in Table 1. First we note that our model significantly outperforms the baselines, especially when the la-

	Novel Set 1					Novel Set 2					Novel Set 3				
Method / Shot	1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
YOLO-joint	0.0	0.0	1.8	1.8	1.8	0.0	0.1	0.0	1.8	0.0	1.8	1.8	1.8	3.6	3.9
YOLO-ft	3.2	6.5	6.4	7.5	12.3	8.2	3.8	3.5	3.5	7.8	8.1	7.4	7.6	9.5	10.5
YOLO-ft-full	6.6	10.7	12.5	24.8	38.6	12.5	4.2	11.6	16.1	33.9	13.0	15.9	15.0	32.2	38.4
LSTD(YOLO)	6.9	9.2	7.4	12.2	11.6	9.9	5.4	3.3	5.7	19.2	10.9	7.6	9.5	15.3	16.9
LSTD(YOLO)-full	8.2	11.0	12.4	29.1	38.5	11.4	3.8	5.0	15.7	31.0	12.6	8.5	15.0	27.3	36.3
Ours	**14.8**	**15.5**	**26.7**	**33.9**	**47.2**	**15.7**	**15.3**	**22.7**	**30.1**	**40.5**	**21.3**	**25.6**	**28.4**	**42.8**	**45.9**

Table 1: Few-shot detection performance (mAP) on the PASCAL VOC dataset. We evaluate the performance on three different sets of novel categories. Our model consistently outperforms baseline methods.

		Average Precision						Average Recall					
# Shots		0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
10	YOLO-ft	0.4	1.1	0.1	0.3	0.7	0.6	5.8	8.0	8.0	0.6	5.1	15.5
	YOLO-ft-full	3.1	7.9	1.7	0.7	2.0	6.3	7.8	10.5	10.5	1.1	5.5	20
	LSTD(YOLO)	0.4	1.1	0.2	0.2	0.7	0.6	5.8	7.9	7.9	0.6	5.0	15.3
	LSTD(YOLO)-full	3.2	8.1	2.1	**0.9**	2.0	6.5	7.8	10.4	10.4	1.1	5.6	19.6
	Ours	**5.6**	**12.3**	**4.6**	**0.9**	**3.5**	**10.5**	**10.1**	**14.3**	**14.4**	**1.5**	**8.4**	**28.2**
30	YOLO-ft	0.6	1.5	0.3	0.2	0.7	1.0	7.4	9.4	9.4	0.4	3.9	19.3
	YOLO-ft-full	7.7	16.7	6.4	0.4	3.3	14.4	11.7	15.3	15.3	1.0	7.7	29.2
	LSTD(YOLO)	0.6	1.4	0.3	0.2	0.8	1.0	7.1	9.1	9.2	0.4	3.9	18.7
	LSTD(YOLO)-full	6.7	15.8	5.1	0.4	2.9	12.3	10.9	14.3	14.3	0.9	7.1	27.0
	Ours	**9.1**	**19.0**	**7.6**	**0.8**	**4.9**	**16.8**	**13.2**	**17.7**	**17.8**	**1.5**	**10.4**	**33.5**

Table 2: Few-shot detection performance for the novel categories on the COCO dataset. We evaluate the performance for different numbers of training shots for the novel categories.

bels are extremely scarce (1-3 shot). The improvements are also consistent for different base/novel class splits and number of shots. In contrast, LSTD(YOLO) can boost performance in some cases, but might harm the detection in other cases. Take 5-shot detection as an example, LSTD(YOLO)-full brings 4.3 mAP improvement compared to YOLO-ft-full on novel set 1, but it is worse than YOLO-ft-full by 5.1 mAP on novel set 2. Second, we note that YOLO-ft/YOLO-ft-full also performs significantly better than YOLO-joint. This demonstrates the necessity of the two training phases employed in our model: it is better to first train a good knowledge representation on base classes and then fine-tune with few-shot data, otherwise joint training with let the detector bias towards base classes and learn nearly nothing about novel classes. More detailed results about each class is available at supplementary material.

COCO. The results for COCO dataset is shown in Table 2. We evaluate for $k = 10$ and $k = 30$ shots per class. In both cases, our model outperforms all the baselines. In particular, when the YOLO baseline is trained with same iterations with our model, it achieves an AP of less than 1%. We also observe that there is much room to improve the results obtained in the few-shot scenario. This is possibly due to the complexity and large amount of data in COCO so that few-shot detection over it is quite challenging.

COCO to PASCAL. We evaluate our model using 10-shot image per class from PASCAL. The mAP of YOLO-ft, YOLO-ft-full, LSTD(YOLO), LSTD(YOLO)-full are 11.24%, 28.29%, 10.99% 28.95% respectively, while our method achieves 32.29%. The performance on PASCAL novel classes is worse than that when we use base classes in PASCAL dataset (which has mAP around 40%). This might be explained by the different numbers of novel classes, *i.e.*, 20 v.s. 5.

4.3. Performance Analysis

Learning speed. Here we analyze learning speed of our models. The results show that despite the fact that our few-shot detection model does not consider adaptation speed explicitly in the optimization process, it still exhibits surprisingly fast adaptation ability. Note that in experiments of Table 1, YOLO-ft-full and LSTD(YOLO)-full requires 25,000 iterations for it to fully converge, while our model only require 1200 iterations to converge to a higher accuracy. When the baseline YOLO-ft and LSTD(YOLO) are trained for the same iterations as ours, their performance is far worse. In this section, we compare the full convergence behavior of YOLO-joint, YOLO-ft-full and our method in Fig. 3. The AP value are normalized by the maximum value during the training of our method and the baseline together. This experiment is conducted on PASCAL VOC base/novel split 1, with 10-shot bounding box labels on novel classes.

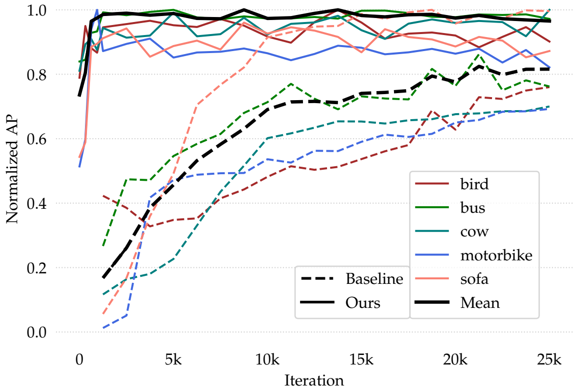


Figure 3: Learning speed comparison between our proposed few-shot detection model and the YOLO-ft-full baseline. We plot the AP (normalized by the converged value) against number of training iterations. Our model shows much faster adaption speed.

From Fig. 3, our method (solid lines) converges significantly faster than the baseline YOLO detector (dashed lines), for each novel class as well as on average. For the class Sofa (orange line), despite the baseline YOLO detector eventually slightly outperforms our method, it takes a great amount of training iterations to catch up with the latter. This behavior makes our model a good few-shot detector in practice, where scarcely labeled novel classes may come in any time and short adaptation time is desired to put the system in real usage fast. This also opens up our model's potential in a life-long learning setting [5], where the model accumulates the knowledge learned from past and uses/adapts it for future prediction. We also observe similar convergence advantage of our model over YOLO-ft-full and LSTD(YOLO)-full.

Learned reweighting coefficients. The reweighting coefficient is important for the meta-feature usage and detection performance. To see this, we first plot the 1024-d reweighting vectors for each class in Fig. 4a. In the figure, each row corresponds to a class and each column corresponds to a feature. The features are ranked by variance among 20 classes from left to right. We observe that roughly half of the features (columns) have notable variance among different classes (multiple colors in a column), while the other half are insensitive to classes (roughly the same color in a column). This suggests that indeed only a portion of features are used differently when detecting different classes, while the remaining ones are shared across different classes.

We further visualize the reweighting vectors by t-SNE [24] in Fig. 4b learned from 10 shots/class on base/novel split 1. In this figure, we plot the reweighting vector generated by each support input, along with their average for each class. We observe that not only vectors of the same classes tend to form clusters, the ones of visually similar classes also tend to be close. For instance, the classes Cow, Horse, Sheep, Cat and Dog are all around the right-bottom corner, and they are all animals. Classes of transportation tools are at the top of the figure. Person and Bird are more visually different from the mentioned animals, but are still closer to them than the transportation tools.

Learned meta features. Here we analyze the learned meta features from the base classes in the first training stage. Ideally, a desirable few-shot detection model should preferably perform as well when data are abundant. We compare the mAP on base classes for models obtained after the first-stage base training, between our model and the vanilla YOLO detector (used in latter two baselines). The results are shown in Table 3. Despite our detector is designed for a few-shot scenario, it also has strong representation power and offers good meta features to reach comparable performance with the original YOLOv2 detector trained on a lot of samples. This lays a basis for solving the few-shot object detection problem.

	Base Set 1	Base Set 2	Base Set 3
YOLO Baseline	**70.3**	**72.2**	70.6
Our model	69.7	72.0	**70.8**

Table 3: Detection performance (mAP) on base categories. We evaluate the vanilla YOLO detector and our proposed detection model on three different sets of base categories.

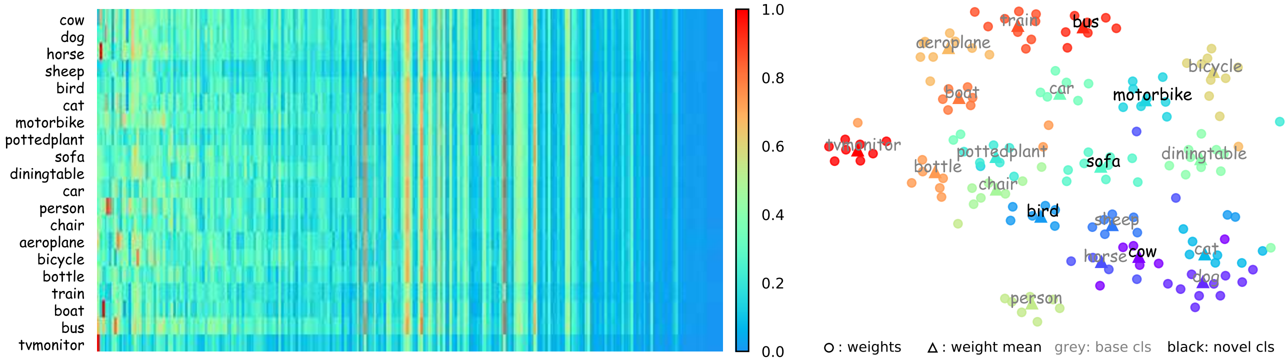
4.4. Ablation Studies

We analyze the effects of various components in our system, by comparing the performance on both base classes and novel classes. The experiments are on PASCAL VOC base/novel split 1, using 10-shot data on novel classes.

Which layer output features to reweight. In our experiments, we apply the reweighting module to moderate the output of the second last layer (layer 21). This is the highest level of intermediate features we could use. However, other options could be considered as well. We experiment with applying the reweighting vectors to feature maps output from layer 20 and 13, while also considering only half of features in layer 21. The results are shown in Table 4. We can see that the it is more suitable to implement feature reweighting

at deeper layers, as using earlier layers gives worse performance. Moreover, moderating only half of the features does not hurt the performance much, which demonstrates that a significant portion of features can be shared among classes, as we analyzed in Sec. 4.3.

Loss functions. As we mentioned in Sec. 3.2, there are several options for defining the classification loss. Among them the binary loss is the most straightforward one: if the inputs to the reweighting module and the detector are from the same class, the model predicts 1 and otherwise



(a) **(b)**

Figure 4: (a) Visualization of the reweighting coefficients (in row vectors) from the reweighting module for each class. Columns correspond to meta feature maps, ranked by variance among classes. Due to space limit, we only plot randomly sampled 256 features. (b) t-SNE [24] visualization of the reweighting coefficients. More visually similar classes tend to have closer coefficients.

	Layer 13	Layer 20	Layer 21	Layer 21(half)
Base	69.6	69.2	**69.7**	69.2
Novel	40.7	43.6	**47.2**	46.9

Table 4: Performance comparison for the detection models trained with reweighting applied on different layers.

	Single-binary	Multi-binary	Softmax
Base	49.1	64.1	**69.7**
Novel	14.8	41.6	**47.2**

Table 5: Performance comparison for the detection models trained with different loss functions.

0. This binary loss can be defined in following two ways. The single-binary loss refers to that in each iteration the reweighting module only takes one class of input, and the detector regresses 0 or 1; and the multi-binary loss refers to that per iteration the reweighting module takes N examples from N classes, and compute N binary loss in total. Prior works on Siamese Network [18] and Learnet [2] use the single-binary loss. Instead, our model uses the softmax loss for calibrating the classification scores of N classes. To investigate the effects of using different loss functions, we compare model performance trained with the single-binary, multi-binary loss and with our softmax loss in Table 5. We observe that using softmax loss significantly outperforms binary loss. This is likely due to its effect in suppressing redundant detection results.

Input form of reweighting module. In our experiments, we use an image of the target class with a binary mask channel indicating position of the object as input to the meta-model. We examine the case where we only feed the image. From Table 6 we see that this gives lower performance especially on novel classes. An apparently reasonable alternative is to feed the cropped target object together with the image. From Table 6, this solution is also slightly worse. The necessity of the mask may lie in that it provides the precise information about the object location and its context.

We also analyze the input sampling scheme for testing and effect of sharing weights between feature extractor and reweighting module. See supplementary material.

Image	Mask	Object	Base	Novel
✓			69.5	43.3
✓	✓		**69.7**	**47.2**
✓		✓	69.2	45.8
✓	✓	✓	69.4	46.8

Table 6: Performance comparison for different support input forms. The shadowed line is the one we use in main experiments.

5. Conclusion

This work is among the first to explore the practical and challenging few-shot detection problems. It introduced a new model to learn to fast adjust contributions of the basic features to detect novel classes with a few example. Experiments on realistic benchmark datasets clearly demonstrate its effectiveness. This work also compared the model learning speed, analyzed predicted reweighting vectors and contributions of each design component, providing in-depth understanding of the proposed model. Few-shot detection is a challenging problem and we will further explore how to improve its performance for more complex scenes.

Acknolwedgement

Jiashi Feng was partially supported by NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112. This work was in part supported by the US DoD, the Berkeley Deep Drive (BDD) Center, and the Berkeley Artificial Intelligence Research (BAIR) Lab.

References

[1] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. *arXiv preprint arXiv:1804.04340*, 2018. 3

[2] Luca Bertinetto, João F Henriques, Jack Valmadre, Philip Torr, and Andrea Vedaldi. Learning feed-forward one-shot learners. In *Advances in Neural Information Processing Systems*, pages 523–531, 2016. 2, 8

[3] Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2846–2854, 2016. 3

[4] Hao Chen, Yali Wang, Guoyou Wang, and Yu Qiao. Lstd: A low-shot transfer detector for object detection. *arXiv preprint arXiv:1803.01529*, 2018. 3, 5

[5] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018. 7

[6] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016. 5

[7] Ali Diba, Vivek Sharma, Ali Mohammad Pazandeh, Hamed Pirsiavash, and Luc Van Gool. Weakly supervised cascaded convolutional networks. In *CVPR*, 2017. 3

[8] Xuanyi Dong, Liang Zheng, Fan Ma, Yi Yang, and Deyu Meng. Few-example object detection with model communication. *arXiv preprint arXiv:1706.08249*, 2017. 3

[9] Matthijs Douze, Arthur Szlam, Bharath Hariharan, and Hervé Jégou. Low-shot learning with large-scale diffusion. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[10] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015. 5

[11] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[12] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *ICML*, 2017. 1, 2

[13] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375, 2018. 3

[14] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2

[15] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 1, 2

[16] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3037–3046. IEEE, 2017. 3

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014. 2

[18] Gregory Koch. Siamese neural networks for one-shot image recognition. In *ICML Workshop*, 2015. 2, 8

[19] Brenden M Lake, Ruslan R Salakhutdinov, and Josh Tenenbaum. One-shot learning by inverting a compositional causal process. In *Advances in neural information processing systems*, pages 2526–2534, 2013. 2

[20] Fei-Fei Li, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006. 2

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[22] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2

[23] Zelun Luo, Yuliang Zou, Judy Hoffman, and Li F Fei-Fei. Label efficient learning of transferable representations acrosss domains and tasks. In *Advances in Neural Information Processing Systems*, pages 165–177, 2017. 2

[24] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 7, 8

[25] Ishan Misra, Abhinav Shrivastava, and Martial Hebert. Watch and learn: Semi-supervised learning for object detectors from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3593–3602, 2015. 3

[26] Hang Qi, Matthew Brown, and David G Lowe. Low-shot learning with imprinted weights. *arXiv preprint arXiv:1712.07136*, 2017. 3

[27] Shafin Rahman, Salman Khan, and Fatih Porikli. Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts. *arXiv preprint arXiv:1803.06049*, 2018. 3

[28] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 2

[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2

[30] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525. IEEE, 2017. 1, 2, 3, 5

[31] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region

proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1, 2, 5

[33] Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual review of biomedical engineering*, 19:221–248, 2017. 1

[34] Zhiqiang Shen, Zhuang Liu, Jianguo Li, Yu-Gang Jiang, Yurong Chen, and Xiangyang Xue. Dsod: Learning deeply supervised object detectors from scratch. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1937–1945. IEEE, 2017. 5

[35] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017. 1, 2

[36] Hyun Oh Song, Yong Jae Lee, Stefanie Jegelka, and Trevor Darrell. Weakly-supervised discovery of visual pattern configurations. In *Advances in Neural Information Processing Systems*, pages 1637–1645, 2014. 3

[37] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. *CVPR*, 2018. 2

[38] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013. 2

[39] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016. 1, 2

[40] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *CVPR*, 2018. 3

[41] Yu-Xiong Wang and Martial Hebert. Model recommendation: Generating object detectors from few samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1619–1628, 2015. 3

[42] Pengkai Zhu, Hanxiao Wang, Tolga Bolukbasi, and Venkatesh Saligrama. Zero-shot detection. *arXiv preprint arXiv:1803.07113*, 2018. 3