GenDet: Meta Learning to Generate Detectors from Few Shots

Liyang Liu, Bochao Wang, Zhanghui Kuang, Jing-Hao Xue, Member, IEEE,
Yimin Chen, Wenming Yang, Member, IEEE, Qingmin Liao, Member, IEEE, and Wayne Zhang

***Abstract*—Object detection has made enormous progress and has been widely used in many applications. However, it performs poorly when only limited training data is available for novel classes which the model has never seen before. Most existing approaches solve few-shot detection tasks implicitly without directly modeling the detectors for novel classes. In this paper, we propose GenDet, a new meta-learning based framework which can effectively generate object detectors for novel classes from few shots, and thus conducts few-shot detection tasks explicitly. The detector generator is trained by numerous few-shot detection tasks sampled from base classes each with sufficient samples, and thus it is expected to generalize well on novel classes. An adaptive pooling module is further introduced to suppress distracting samples and aggregate the detectors generated from multiple shots. Moreover, we propose to train a reference detector for each base class in the conventional way, with which to guide the training of the detector generator. The reference detectors and the detector generator can be trained simultaneously. Finally, the generated detectors of different classes are encouraged to be orthogonal to each other for better generalization. The proposed approach is extensively evaluated on the ImageNet, VOC and COCO datasets under various few-shot detection settings, and it achieves new state-of-the-art results.**

***Index Terms*—Meta Learning, Few-shot Learning, Object Detection, Weight Generation**

I. Introduction

THANKS to the resurgence of deep networks [1] and the construction of large scale datasets [2], [3], [4], object detection has witnessed consistent improvements in the last five years [5], [6], [7], [8], [9]. Typically, learning to detect novel classes, which the model has never seen before, needs to collect and train on abundant images of those classes, which is prohibited in many real-world applications. Therefore, it is desired to obtain object detectors in the few-shot setting. Namely, a model is trained on the base classes each of which with sufficient training samples, and then can be quickly adapted to novel classes with little data, leveraging the past knowledge learned from base classes.

There have been many few-shot classification methods developed so far [10], [11], [12], [13], [14]. However, few-shot detection, which aims at bounding box regression and region classification simultaneously, is much more complicated but under-explored. Trivially finetuning the model (trained on base classes) with little novel class data easily leads to over-fitting because of data scarcity. Pioneers of few-shot detection have explored background depression [15], metric learning [16] and feature reweighting [17], [18]. Most of these previous works carry out few-shot detection *implicitly*, either by manipulating the feature maps to highlight regions [15] or channels [17], [18] related to novel classes, or by constraining different classes to be separable in a learned embedding space [16]. Besides, [19] proposes a simple finetuning baseline for few-shot detection, introducing the instance-level feature normalization when finetuning on novel classes.

Recently, MetaDet [20] introduces model regression to few-shot detection, in the same spirit with [21]. It learns a class-agnostic transformation $\mathcal{T}_\phi$ which regresses from *models learned with few samples* to *models learned with many samples* in the model parameter space. MetaDet optimizes class-specific weights $\boldsymbol{W}$ of the detectors for novel classes via regularized finetuning as shown in Fig. 1 (a). In contrast, in this work, we propose an approach termed GenDet for few-shot detection, which can directly and effectively generate detectors for novel classes as shown in Fig. 1 (b). GenDet learns a class-agnostic detector generator $\boldsymbol{f}_\psi$ which predicts the detectors from the support examples, and thus it conducts few-shot detection *explicitly*.

GenDet is trained via sampling N shots from each of K sampled base classes and simulating the few-shot detection testing scenario. Furthermore, in previous few shot detection methods [16], [17], [18], only the naïve average strategy is employed, where the detectors from few shots are simply averaged regardless of their importances. This strategy leads to poor quality of the aggregated detectors because there exist noisy samples. We are the first to discuss this problem in few-shot detection, and accordingly we propose a learnable adaptive pooling module to filter out the noisy samples in the few-shot training set. In adaptive pooling the detectors are weighted by their importances, which are implemented as the similarities with the baseline mean detector. We empirically validate the effectiveness of our adaptive pooling module in obtaining better aggregated detectors for novel classes.

This work was partly supported by the Natural Science Foundation of Guangdong Province (No. 2020A1515010711) and the Special Foundation for the Development of Strategic Emerging Industries of Shenzhen (No. JCYJ20200109143035495 and No. JCYJ20200109143010272).

Liyang Liu, Wenming Yang and Qingmin Liao are the Shenzhen Key Lab. of Information Sci&Tech/Shenzhen Engineering Lab. of IS&DCP, Shenzhen International Graduate School/Department of Electronic Engineering, Tsinghua University, Shenzhen, 518055, China (e-mail: liu-ly14@mails.tsinghua.edu.cn; yang.wenming@sz.tsinghua.edu.cn; liaoqm@tsinghua.edu.cn).

Bochao Wang, Zhanghui Kuang, Yimin Chen and Wayne Zhang are with SenseTime Research (e-mail: sergeywong@gmail.com; kuangzhanghui@sensetime.com; chenyimin@sensetime.com; wayne.zhang@sensetime.com).

Jing-Hao Xue is with the Department of Statistical Science, University College London, London WC1E 6BT, U.K. (e-mail: jinghao.xue@ucl.ac.uk).

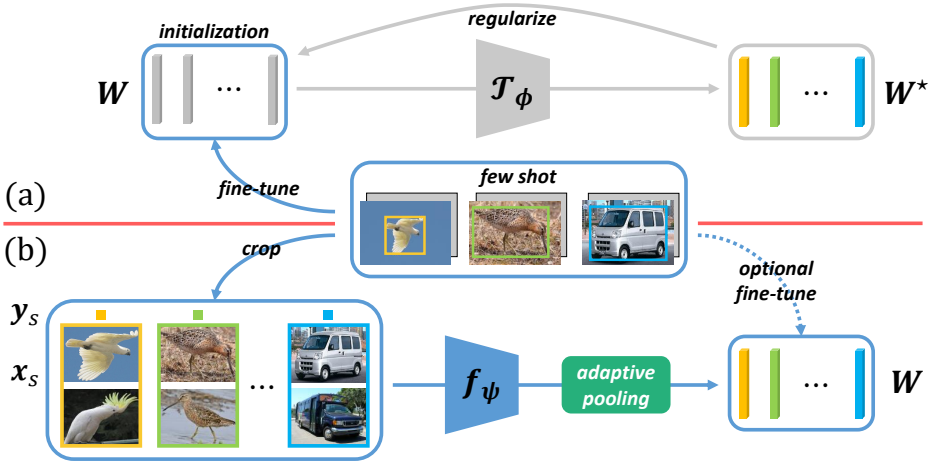

Fig. 1. Comparison of the meta testing procedure between MetaDet [20] and our GenDet. (a) MetaDet optimizes *randomly initialized* novel class detectors $\boldsymbol{W}$ via regularized finetuning. The intermediate detectors $\boldsymbol{W}^\star = \mathcal{T}_\phi(\boldsymbol{W})$ transformed from regressor $\mathcal{T}_\phi$ act as finetuning regularization only, the novel class detectors $\boldsymbol{W}$ finetuned with few-shot data are ultimately used for detecting novel classes; (b) GenDet directly generates detectors $\boldsymbol{W}$ for novel classes from their few cropped regions via the meta model $\boldsymbol{f}_\psi$. The generated detectors $\boldsymbol{W}$ can be directly used to detect novel classes or further finetuned to improve the performance. So MetaDet has to adopt finetuning while GenDet can be applied both with and without finetuning on novel-class samples. In GenDet an adaptive pooling module is also proposed to aggregate the detectors generated from multiple instances.

Besides, we train reference detectors for all the base classes with all samples in the conventional way. Then the generated detector for a class is constrained to approximate its corresponding reference detector, which is more discriminative as it is optimized by the whole dataset other than just the sampled few-shot detection tasks. In this way we use the reference detectors as training guidance to learn a detector generator that can generate detectors with stronger discriminative ability. Surprisingly, the reference detectors and the detector generator can be end-to-end learned simultaneously without stage-wise training as in MetaDet [20]. Finally, to further increase the discriminative ability, the generated detectors for novel classes are encouraged to be orthogonal to each other for better generalization. Our proposed GenDet can be efficiently applied for novel classes without finetuning, and it can be further improved with finetuning as demonstrated in our experiments. In summary, our contributions are as follows:

1) We learn to generate detectors for novel classes by episodic training on sampled base classes. We propose an adaptive pooling module to better aggregate detectors generated from multiple shots, especially when there exist noisy samples.
2) We propose to train reference detectors for all the base classes via conventional (many-shot) training, with which to guide the training of the detector generator, so that it can generate detectors with stronger discriminative ability. The reference detectors and the detector generator can be end-to-end trained in a single stage.
3) Our generated detectors can be deployed both with and without finetuning. We introduce the orthogonality prior as a regularization during few-shot finetuning on novel classes to better distinguish visually similar classes, and thus increasing the generalization ability.
4) Our method is shown to be effective on multiple few-shot detection benchmarks, *i.e.*, ImageNet [16], VOC [17] and COCO [17], where it outperforms previous state-of-the-arts [16], [17], [18], [19], [20] by a large margin, showing its superiority.

II. Related Work

A. Few-shot learning

Few-shot learning aims at rapidly generalizing to new tasks with limited samples, leveraging the prior knowledge learned from large-scale base dataset. Plenty of methods have been developed for few-shot learning, especially few-shot image classification [22]. **Embedding learning** methods [14] embed samples of classes to a semantic space, then classification is done by measuring the similarity of the inputs. Specifically, Siamese Networks [23] embeds pairs of inputs with the same model, and it learns to predict similarity between inputs in order to generalize to novel classes. Matching Networks [10] adopt the attention mechanism on the embeddings of the labeled samples to predict the classes of the unlabeled ones. Prototypical Networks [13] extend Matching Networks by using a prototype, the centroid of each class, as the classifier. [24] additionally learns a category-agnostic mapping to transform the mean-sample representation to its class-prototype representation, considering the prototypes in Prototypical Networks [13] may be inaccurate. Furthermore, finetuning with geometric constraints is applied in [25] to force intra-class similarity and inter-class disparity. [26] transfers knowledge from base classes to novel classes by embedding image features as class adapting principal directions, instead of the vector representation adopted by previous works.

Weight generation methods [11], [27] predict the parameters of novel classes from support images. Model Recommendation [28] generates novel-task parameters by collaborative filtering. Weight imprinting [29] adds new weight vectors to the classification layer of a model without requiring back propagation. [30] achieves few-shot fine-grained recognition by learning the piecewise exemplar-to-classifier mapping function to avoid overfitting caused by the high dimensionality.

Learning to finetune methods [12] achieve fast convergence on novel tasks by learning good parameter initializations which are common for all tasks. **Learning with external memory** methods [31], [32], [33] directly stores the needed prior knowledge in the external memory to be retrieved or updated, and thus enables fast generalization. **Generative modeling** methods [34], [35] hallucinate samples of novel classes to accommodate for the lack of data. Others learn prior knowledge by parts and relations [36], [37], super classes [38] or latent variables.

Our work also relates to robust few-shot learning [39], where representation outliers and label outliers [40] of the novel dataset are analyzed, and outlier suppression is achieved by the robust attentive profile networks. The adaptive pooling module proposed is our effort to deal with outliers of the support set. Compared with classification, it is much more complicated to annotate images for detection, so it is appealing to learn detectors of novel classes from few samples, which is an under-explored problem we devoted ourselves to.

B. Generic object detection

Great achievements have been made in generic object detection with deep learning models [5], [8], [41], [42]. According to whether there are region proposals to be used, these methods can be divided into two categories: proposal-based and proposal-free. Proposal-based detectors, pioneered by R-CNN series [5], [8], [41], extract class-agnostic region proposals of the potential objects at first, then proposal-level classification and box regression are conducted. In contrast, proposal-free detectors attempt to predict bounding boxes and get detection confidences of each category directly. YOLO [7] divides the image into grids, then predicts several bounding boxes and class probabilities for each grid cell. SSD [6] extends YOLO with feature pyramid for multi-scale detection and it adopts default/anchor boxes with different sizes to detect objects with various shapes. RetinaNet [43] proposes a focal loss to deal with the imbalance between easy and hard examples in dense detection on multi-scale feature maps extracted by FPN [44], [45]. Recently, anchor-free methods [46], [47] are proposed to eliminate the pre-defined anchor boxes in the detection framework. FCOS [47] makes full use of all points in a ground truth box and suppresses low-quality detected bounding boxes by a centerness branch. CornerNet [48] detects pairs of corners and groups them to form the final detected bounding boxes. These generic object detection methods focus on many-shot scenarios while we develop GenDet for few-shot cases.

C. Few-shot detection

Most existing few-shot learning methods focus on classification, while few-shot detection is an under-explored problem and there are only few attempts. MSPLD [49] leverages a large number of unlabeled images for representation learning in the context of few-shot detection. LSTD [15] designs a deep architecture to boost the recall rate for detection, using SSD [6] as the RPN module of Faster R-CNN [8]. The authors propose to transfer the pretrained detector to the few-shot scenario via regularized finetuning, where background depression [15] is adopted. Kang *et al.* [17] propose a feature reweighting method where a model learns to predict reweighting vectors for each novel class from few shots, and the reweighting vector is applied to the feature map of the image to obtain class-specific features. Meta R-CNN [18], by reweighting the features of RoIs, extends the work of [17] with Faster [8]/Mask R-CNN [50] in detection and segmentation, respectively. LSTD [15] and Feature Reweighting [17]/Meta R-CNN [18] can be considered to highlight features related to novel classes, either in the spatial or channel dimension, and then class-agnostic detection head is applied on the reweighted feature maps to detect novel classes. Our approach differs from theirs in that we generate class-specific detectors explicitly from support instances while feature reweighting methods [17], [18] predict a class attentive vector, which is used to reweight the feature maps and thus acts as an implicit detector. Also, instead of the average pooling used in [17], [18], where the attentive vector for each class is computed by simply averaging over few shots, we propose a learnable adaptive pooling (AdaPool) module to effectively aggregate information from multiple shots, suppressing the influence of noisy samples. RepMet [16] exploits a distance metric learning (DML) module to model multi-modal distribution of each category. It learns a generalizable metric via triplet loss [51] on base classes and transfers it to novel classes. RepMet conducts detection by computing the distance of RoI features in the embedding space, and thus it can be considered as a non-parametric model. It does not consider the K-way, N-shot testing scenario, while our model is trained as it is deployed, and thus takes the advantage of "training as testing" [10]. TFA [19] trains the class-agnostic backbone on the base classes and then learns the class-specific heads (box classifier and regressor) on novel-class data, where instance-level feature normalization is adopted in the scaled cosine similarity for better performance.

MetaDet [20] regresses from detector parameters learned with few samples to those learned with many samples, while our GenDet directly transforms few-shot samples of novel classes to detector parameters. MetaDet employs a two-stage meta-training procedure. In the first stage, the class-agnostic backbone and the base-class detectors are trained on large-sample images. Then in the second stage, the detector transformer $\boldsymbol{\mathcal{T}}_{\boldsymbol{\phi}}$ is learned by sampling few-shot tasks from base classes, where it is responsible for predicting the detectors learned with many shots from those learned with few shots. This two-stage training paradigm may lead to suboptimal optimization. However, our detector generator can be learned simultaneously with the reference detectors in a single stage, and thus is benefited from end-to-end learning. Also, MetaDet can only be adopted with finetuning on novel classes. During finetuning, as shown in Fig. 1 (a), *approximated* many-shot detectors $\boldsymbol{W}^{\star} = \boldsymbol{\mathcal{T}}_{\boldsymbol{\phi}}(\boldsymbol{W})$ are used to regularize the optimization of few-shot detectors $\boldsymbol{W}$ for novel classes. Namely, the target novel class detectors $\boldsymbol{W}$ are randomly initialized and learned with few shots from novel classes, and at the same time $\boldsymbol{W}$ are regularized to match the transformed ones $\boldsymbol{W}^{\star}$. In contrast, our generated detectors are already applicable ones and can be directly used to detect novel classes. They can also be regarded as good initialization and further finetuned,

TABLE I
COMPARISON BETWEEN GENDET AND PREVIOUS STATE-OF-THE-ARTS.

method	RepMet [16]	Reweight [17]	Meta R-CNN [18]	TFA [19]	MetaDet [20]	GenDet (Ours)
parametric model	✗	✓	✓	✓	✓	✓
class-specific head	✓	✗	✗	✓	✓	✓
adaptive pooling	✗	✗	✗	✗	✗	✓
many-shot guided	✗	✗	✗	✗	✓	✓
end-to-end training	✓	✓	✓	✓	✗	✓
episodic training	✗	✓	✓	✗	✓	✓
optional finetuning	✓	✗	✗	✗	✗	✓
orthogonality prior	✗	✗	✗	✗	✗	✓

opening up the opportunity for both high efficiency and high performance. Tab. I gives a clear comparison between GenDet and previous methods for few-shot detection.

D. Siamese tracking

GenDet also relates to Siamese-network-based tracking methods [52], [53], which can be formulated as one shot detection in a local region other than the full image. In these methods, RoI on the template frame acts as the target object and a network extracts features from it. The same network extracts feature maps of subsequent frames, where the one shot detection happens through a correlation operator between the feature of RoI and the feature maps of upcoming frames. Locality prior ensures that the location of the detected target object does not depart too much away from its initial position. Tracking could be regarded as a special case of few-shot detection but is less ambiguous, as the appearances in adjacent frames do not vary much; whereas in detection, instances of the same class can look very different while instances of different classes may appear similar. So it requires the detection model to have much stronger discriminative ability than tracking.

III. METHODOLOGY

A. Tasks and motivation

We first review the task of few-shot detection. In K-way, N-shot detection, we are given a support dataset $\mathcal{D}_s = \{(\boldsymbol{x}_s, \boldsymbol{y}_s)_i\}$ for K novel classes, each with N support instances (so $1 \leq i \leq KN$). $\boldsymbol{x}_s$ denotes the object region in support images and $\boldsymbol{y}_s = \left(\boldsymbol{y}_s^{\text{cls}}, \boldsymbol{y}_s^{\text{loc}}\right)$ includes the *class* and *location* label of $\boldsymbol{x}_s$. We are supposed to obtain detectors $\boldsymbol{W}$ by using the provided training samples $\mathcal{D}_s$. Then the detectors $\boldsymbol{W}$ are used to predict detection results on the query dataset $\mathcal{D}_q = \left\{\left(\boldsymbol{x}_q, \boldsymbol{y}_q\right)_j\right\}$. $\boldsymbol{x}_q$ represents an RoI induced by a region proposal (or a local region induced by the receptive field of a network), and its ground truth label $\boldsymbol{y}_q = \left(\boldsymbol{y}_q^{\text{cls}}, \boldsymbol{y}_q^{\text{loc}}\right)$ is used to evaluate the performance of the detectors $\boldsymbol{W}$.

As we know, directly training $\boldsymbol{W}$ on the few-shot dataset $\mathcal{D}_s$ easily leads to over-fitting and thus weak generalization ability on $\mathcal{D}_q$. So we propose to train a meta model $\boldsymbol{f}_{\boldsymbol{\psi}}$ to generate detectors on C base classes, each with abundant training samples, and then apply the learned meta model $\boldsymbol{f}_{\boldsymbol{\psi}}$ on K novel classes. The procedure is divided into two phases. In *meta-training*, to simulate the testing scenario, we randomly sample K-way, N-shot detection tasks from C base classes to train the meta model $\boldsymbol{f}_{\boldsymbol{\psi}}$. The sampled tasks are also known as "episodes" and this training strategy is termed as "episodic training". As the meta model can be trained on as many sampled tasks as desired, the common knowledge of generating detectors can be acquired and later generalized to K novel classes. Then in *meta-testing*, the trained meta model is applied to generate detectors for K novel classes. By learning to generate detectors using the meta model $\boldsymbol{f}_{\boldsymbol{\psi}}$, we effectively avoid the over-fitting issue caused by directly training the detectors $\boldsymbol{W}$ with few training samples.

B. Detector generation

Specifically, we learn to generate detectors directly from few support instances with the meta model $\boldsymbol{f}_{\boldsymbol{\psi}}$ as shown in Fig. 2. At the top of the figure, $\boldsymbol{f}_{\boldsymbol{\psi}}$ is parameterized by $\boldsymbol{\psi}$ and takes the cropped object regions $\boldsymbol{x}_s$ of support images as input. For the 1-shot setting, it outputs the predicted detector of the corresponding class $\boldsymbol{y}_s^{\text{cls}}$ as follows:

$$\widehat{\boldsymbol{W}}_k^{(1)} = \boldsymbol{f}_{\boldsymbol{\psi}}\left(\boldsymbol{x}_s\right) \in \mathbb{R}^D,\ k = \boldsymbol{y}_s^{\text{cls}}, \tag{1}$$

where D is the dimension of vectorized detector parameters. In general, the generated detectors may contain both the region classification and box regression weights. In the middle, a region $\boldsymbol{x}_q$ on query images is transformed to $\boldsymbol{g}_{\boldsymbol{\phi}}\left(\boldsymbol{x}_q\right)$ on the feature map through a parameterized feature extractor $\boldsymbol{g}_{\boldsymbol{\phi}}$. The generated detectors of K classes $\widehat{\boldsymbol{W}} \in \mathbb{R}^{K \times D}$ are convolved with the extracted feature $\boldsymbol{g}_{\boldsymbol{\phi}}\left(\boldsymbol{x}_q\right)$ to predict the query label $\boldsymbol{y}_q$. In meta training, we train the detector generator $\boldsymbol{f}_{\boldsymbol{\psi}}$ and feature extractor $\boldsymbol{g}_{\boldsymbol{\phi}}$ simultaneously via sampling episodes from base classes, where parameters $\{\boldsymbol{\psi}, \boldsymbol{\phi}\}$ are optimized through loss minimization. The loss can be any conventional detection loss for region classification and location regression. Here we term it as the **generated detector loss**:

$$\mathcal{L}_{\text{d}} = \text{loss}\left(\boldsymbol{y}_q; \boldsymbol{g}_{\boldsymbol{\phi}}\left(\boldsymbol{x}_q\right), \widehat{\boldsymbol{W}}\right). \tag{2}$$

The detector generator $\boldsymbol{f}_{\boldsymbol{\psi}}$ is learned with a large number of few-shot detection tasks sampled from base classes, so it can transfer its detector generation ability to novel classes. Then in meta testing, detectors for novel classes can be generated by using few instances of novel classes. Later, the generated detectors could be used to detect other instances of these novel classes on testing images.

In our GenDet different classes share the bounding box regression parameters, which is commonly employed in object detection, especially in one-stage methods. So $\boldsymbol{f}_{\boldsymbol{\psi}}$ is supposed

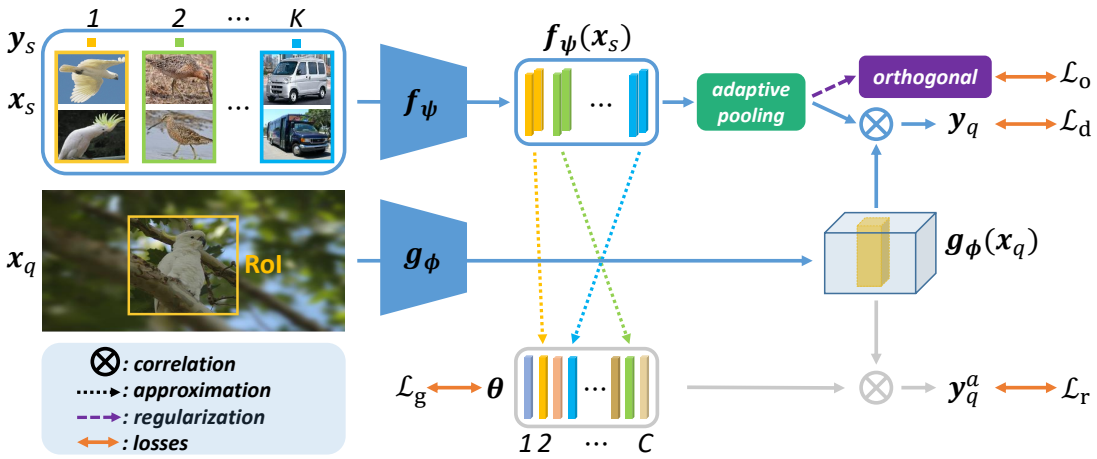

Fig. 2. Schematic illustration of our framework GenDet. **Top:** detector generation branch. The detector generator $\boldsymbol{f}_\psi$ generates the detectors from support instances $(\boldsymbol{x}_s, \boldsymbol{y}_s)$ of different classes (indicated by different colors). The adaptive pooling module is proposed to aggregate the detectors generated by multiple shots of each class. The generated detectors are also regularized to be orthogonal to each other for better discriminative ability. **Middle:** representation learning branch. The feature extractor $\boldsymbol{g}_\phi$ transforms each region $\boldsymbol{x}_q$ (yellow rectangle) in query images to $\boldsymbol{g}_\phi(\boldsymbol{x}_q)$ (yellow tensor) on the feature map. The extracted feature $\boldsymbol{g}_\phi(\boldsymbol{x}_q)$ of an RoI is correlated/convolved with the generated detector $\boldsymbol{f}_\psi(\boldsymbol{x}_s)$ to predict the query label $\boldsymbol{y}_q$. **Bottom:** reference detector learning branch. The reference detectors $\boldsymbol{\theta}$ are introduced to guide the training of detector generation. Reference detectors $\boldsymbol{\theta}$ are implemented as randomly initialized model parameters for all the base classes, like in the many-shot setting. It plays a similar role as $\boldsymbol{f}_\psi(\boldsymbol{x}_s)$ and is correlated with the extracted RoI feature $\boldsymbol{g}_\phi(\boldsymbol{x}_q)$ to predict the *auxiliary* query label $\boldsymbol{y}_q^a$. The dashed lines indicate that the generated detectors should try to approximate the reference detectors. The main difference between the two branches (top and bottom) lies in *what detectors* an RoI is convolved with to predict the corresponding target label. In the top branch, the detectors are generated by the meta-model $\boldsymbol{f}_\psi$ from few instances of the sampled K base classes. While in the bottom branch, the detectors are class-specific parameters for all of the C base classes. More details about the losses can be found in the text.

to extract features discriminative enough to distinguish different classes, it takes the object regions cropped from support images as input and thus the location information of support images is only used for cropping objects. The query image in each episode is the input of the feature extractor $\boldsymbol{g}_\phi$ to produce the feature maps, which the generated detectors are convolved with to produce the detection results. During training, we sample an image from one of the sampled base classes as the query image to ensure that there is at least one instance from the sampled classes. The query image acts as a positive example for classes that appear in it and a negative example for classes that do not. During testing, the query image may either contain instances of the K novel classes or not. The few shot detection method is supposed to predict correct results if there exist instances from the K classes, otherwise it should predict the query image as all-background.

We make the detector generator $\boldsymbol{f}_\psi$ and the feature extractor $\boldsymbol{g}_\phi$ not to share parameters because of their different roles: 1) $\boldsymbol{f}_\psi$ focuses on the classification-related features while $\boldsymbol{g}_\phi$ extracts features that are responsible for both the classification and regression task; 2) $\boldsymbol{f}_\psi$ is applied on the cropped region of objects while $\boldsymbol{g}_\phi$ is applied on the whole image, where specific domain shifts exist. In fact we have tried to share the backbone but it leads to slightly worse performance (~1% mAP as in Tab. III) on the 5-way 5-shot VOC benchmark, which may result from less model capacity. We also notice that sharing the parameters between $\boldsymbol{f}_\psi$ and $\boldsymbol{g}_\phi$ will not reduce computation and thus cannot speed up the inference procedure.

C. Adaptive pooling

In the multi-shot ($N > 1$) case, the aggregated generated detector for novel class k can be obtained via simply averaging the generated detector parameters from each support instance n of class k:

$$\overline{\boldsymbol{W}}_k = \frac{1}{N}\sum_{n=1}^{N} \widehat{\boldsymbol{W}}_k^{(n)}, \tag{3}$$

$$\widehat{\boldsymbol{W}}_k^{(n)} = \boldsymbol{f}_\psi\left(\boldsymbol{x}_s^{(n)}\right),\ k = \boldsymbol{y}_s^{\mathrm{cls}(n)}, \tag{4}$$

where (n) indicates the n-th example. However, as shown in Fig. 3, some support instances for the novel class can be noisy, which easily leads the averaged parameters to be distracted and results in less effective detectors. Inspired by the set-based face recognition [54], we propose to overcome this issue via adaptively pooling the information from multiple shots, highlighting the commonality and suppressing the noise. To be specific, the adaptively pooled detector is obtained by a weighted average, with the weight computed by the similarity between the single-shot detector and the mean detector:

$$\widehat{\boldsymbol{W}}_k = \sum_{n=1}^{N} \alpha_n \widehat{\boldsymbol{W}}_k^{(n)}, \tag{5}$$

$$\alpha_n = \frac{\exp\left(s_n\right)}{\sum_{m=1}^{N}\exp\left(s_m\right)},\ s_n = \widehat{\boldsymbol{W}}_k^{(n)} \cdot \mathrm{FC}\left(\overline{\boldsymbol{W}}_k\right), \tag{6}$$

where FC means a fully connected layer and $\cdot$ is the inner product. Adaptive pooling can be seen as a learnable generalization of average pooling, and in this way we are able to aggregate the information from multiple shots and suppress the noise. Fig. 3 visualizes the learned weights, as shown, the undesired support instances are effectively down-weighted.

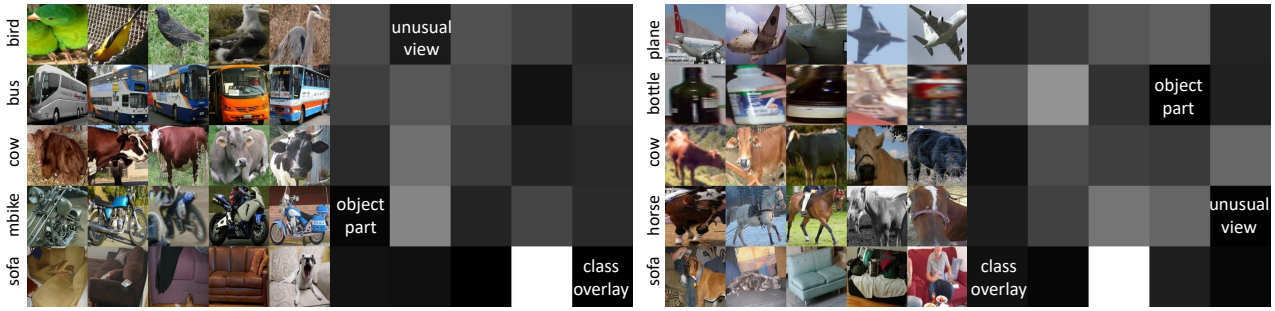

Fig. 3. Certain object instances for the novel classes can be noisy, *e.g.*, "class overlay", "object part" and "unusual view", and thus distract the averaged generated detectors. Adaptive pooling is proposed to adaptively weight the detectors generated by these instances, according to their importances, which is computed by the similarity with the average detector. The grayscale images beside the natural images visualize the learned importance of each support instance, and darker means less important. As shown noisy instances are effectively suppressed.

D. Reference guided training

If we only employ episodic training to obtain the learned parameters $\{\psi, \phi\}$ of the detector generator $\boldsymbol{f}_{\psi}$ and feature extractor $\boldsymbol{g}_{\phi}$, each training iteration is restricted to one episode where only a limited number of classes appear. This may result in weak discriminative ability of the extracted feature $\boldsymbol{g}_{\phi}(\boldsymbol{x}_q)$, as it distinguishes the sampled classes only. Moreover, in the absence of abundant training data, the generated detectors $\widehat{\boldsymbol{W}}$ can hardly match the detectors learned with large-sample images. So we propose to train reference detectors via large-sample training, with which to guide the training of the detector generator $\boldsymbol{f}_{\psi}$ and the feature extractor $\boldsymbol{g}_{\phi}$. Specifically, apart from the generated detectors $\widehat{\boldsymbol{W}}$, we introduce a reference detector $\boldsymbol{\theta}_c \in \mathbb{R}^D$ (bottom of Fig. 2) for each base class c which is randomly initialized and trained simultaneously with $\boldsymbol{f}_{\psi}$ and $\boldsymbol{g}_{\phi}$. This leads to the following two advantages. **Firstly**, with the help of the reference detectors $\boldsymbol{\theta} \in \mathbb{R}^{C\times D}$, the extracted feature $\boldsymbol{g}_{\phi}(\boldsymbol{x}_q)$ is forced to distinguish all base classes at each iteration. So $\boldsymbol{g}_{\phi}$ can produce much more discriminative representations than the episodic-only case. In implementation, besides the original class label $\boldsymbol{y}_q^{\text{cls}} \in [1, K]$ corresponding to the index in the sampled K classes of each episode, for each query region $\boldsymbol{x}_q$ we introduce another auxiliary class label $\boldsymbol{y}_{\text{all}} \in [1, C]$ corresponding to the index in the total C base classes. $\boldsymbol{y}_{\text{all}}$ exists because the sampled K classes all come from the C base classes. Note that the location label is shared in both cases, regardless of K sampled classes or C overall classes. Then $\boldsymbol{g}_{\phi}(\boldsymbol{x}_q)$ is convolved with $\boldsymbol{\theta}$ to predict the newly introduced auxiliary label $\boldsymbol{y}_q^a = \left(\boldsymbol{y}_{\text{all}}, \boldsymbol{y}_q^{\text{loc}}\right)$, where we introduce the **reference detector loss**:

$$\mathcal{L}_{\text{r}} = \text{loss}\left(\boldsymbol{y}_q^a; \boldsymbol{g}_{\phi}(\boldsymbol{x}_q), \boldsymbol{\theta}\right). \tag{7}$$

Secondly, via constraining the generated detector to be close to its corresponding reference detector, we can generate better detectors $\widehat{\boldsymbol{W}}$ for sampled classes, which in turn leads to better optimization for detector generator $\boldsymbol{f}_{\psi}$. We specify the **guided generation loss** as L_1 discrepancy:

$$\mathcal{L}_{\text{g}} = \sum_{c=1}^{C} \mathbb{I}\left(\boldsymbol{y}_s^{\text{cls}} = c\right) \|\boldsymbol{f}_{\psi}(\boldsymbol{x}_s) - \boldsymbol{\theta}_c\|_1, \tag{8}$$

where $\mathbb{I}(\cdot)$ is an indicator function which equals 1 if the condition in the bracket is true and 0 otherwise, and $\boldsymbol{\theta}_c$ means the reference detector for class c. Moreover, with the trained reference detectors for base classes, we endow our model the ability to detect both base and novel classes. We can use $\boldsymbol{\theta}$ as base class detectors and employ $\boldsymbol{f}_{\psi}$ to generate detectors for novel classes in meta testing.

We stress that in our GenDet, training the reference detectors $\boldsymbol{\theta}$ of base classes do not violate the meta-learning concepts. We are not trying to fit class-specific losses of base classes, in contrast we use $\boldsymbol{\theta}$ to guide the training of the detector generator $\boldsymbol{f}_{\psi}$ via losses $\mathcal{L}_{\text{r}}$ and $\mathcal{L}_{\text{g}}$. By *episodic training* the generated detectors only need to distinguish the sampled K classes in each episode, while by *conventional training* the reference detectors $\boldsymbol{\theta}$ are trained to distinguish all $C > K$ base classes. So theoretically $\boldsymbol{\theta}$ is more discriminative than the generated detectors. By constraining the generated detectors to approximate $\boldsymbol{\theta}$, eventually $\boldsymbol{f}_{\psi}$ is better trained to generalize on novel classes in the K-way N-shot setting, such that it can generate novel class detectors with stronger discriminative ability.

E. Orthogonality regularization

Our meta-model $\boldsymbol{f}_{\psi}$ after meta-training can be deployed both without and with finetuning on the test dataset. If without finetuning, the detectors $\widehat{\boldsymbol{W}}$ generated by applying $\boldsymbol{f}_{\psi}$ on novel classes can be immediately used. Otherwise if with finetuning, $\widehat{\boldsymbol{W}}$ is used to initialize the target novel class detectors $\boldsymbol{W}$, which is then optimized by loss minimization with few samples. We can regard our generated detectors as good initialization for the novel-class detectors, and the optimized detectors can be obtained in just few epochs before over-fitting. However, as some novel classes may share similar appearance but have not been used to train the detector generator $\boldsymbol{f}_{\psi}$, so the generated detectors for them can be ambiguous.

In few-shot finetuning, from the perspective of Bayesian parameter estimation, an appropriate prior distribution of parameters can provide useful bias when data lacks. It motivates us to use the natural orthogonality prior for regularizing $\boldsymbol{W} \in \mathbb{R}^{K\times D}$ of novel classes, adding discriminative ability to the model. The above constraint can be formulated as the **orthogonal regularization loss** as follows:

$$\mathcal{L}_{\mathrm{o}}\left(\boldsymbol{W}\right)=\|\widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{W}}^{\top}-\boldsymbol{I}_{K}\|_{1}, \tag{9}$$

where $\widetilde{\boldsymbol{W}}$ is the row normalized version of $\boldsymbol{W}$, $\|\cdot\|_1$ denotes the entry-wise matrix norm which is the sum of all entries' absolute values in the matrix and $\boldsymbol{I}_K \in \mathbb{R}^{K\times K}$ is the identity matrix. This prior is based on the intuition that detectors among different classes should vary from each other so as to maximize the discriminative power. Eventually, we cast finetuning on the K-way N-shot dataset as maximum a posteriori. When lack of data we could resort to appropriate prior. We argue that the orthogonality prior is suitable for our classification parameters since we are trying to distinguish the K classes, especially when they are visually similar. The eventual loss is composed of both the *likelihood* of few-shot data and the *prior* from regularization. The orthogonality term only acts as a weighted regularization, not a hard constraint.

As for implementation, we make different class detectors share the box regression parameters. The classification parameters for all K novel classes can be stacked as $\boldsymbol{W} \in \mathbb{R}^{K\times D}$ during finetuning, for FCOS $D = 256$ and for Faster R-CNN $D = 2048$. To apply the orthogonality constraint, we first normalize $\boldsymbol{W}$ to obtain $\widetilde{\boldsymbol{W}} \in \mathbb{R}^{K\times D}$ so that each row in $\widetilde{\boldsymbol{W}}$ is a unit-norm vector. Then we compute the cosine similarity between each pair of rows in $\widetilde{\boldsymbol{W}}$ by $\boldsymbol{S} = \widetilde{\boldsymbol{W}}\widetilde{\boldsymbol{W}}^{\top} \in \mathbb{R}^{K\times K}$. Next we constrain the similarity matrix $\boldsymbol{S}$ to be close to the identity matrix $\boldsymbol{I}_K \in \mathbb{R}^{K\times K}$ via L_1 loss, such that the similarity between any pair of different classes is minimized, or equivalently the discriminative ability of classification parameters is maximized.

F. Model training and testing

During meta-training, we sample episodic dataset $\mathcal{D}^t = \left\{\left(\boldsymbol{x}_s, \boldsymbol{y}_s\right)_i^t, \left(\boldsymbol{x}_q, \boldsymbol{y}_q\right)_j^t\right\}$ for each task t from base classes and learn the model parameters $\{\boldsymbol{\psi}, \boldsymbol{\phi}, \boldsymbol{\theta}\}$ for detector generator $\boldsymbol{f}_{\boldsymbol{\psi}}$, feature extractor $\boldsymbol{g}_{\boldsymbol{\phi}}$ and reference detectors $\boldsymbol{\theta}$, via minimizing the losses discussed above:

$$\mathcal{L} = \mathcal{L}_{\mathrm{d}} + \mathcal{L}_{\mathrm{r}} + \alpha\mathcal{L}_{\mathrm{g}}, \tag{10}$$

where α is the hyper-parameter for balancing the guided generation loss. One may expect that the detector generator $\boldsymbol{f}_{\boldsymbol{\psi}}$ and the reference detectors $\boldsymbol{\theta}$ are to be trained in two separate stages as in MetaDet [20], but we emphasize that the model can be trained end-to-end in a single stage. Specifically, in the implementation, $\boldsymbol{\theta}$ is randomly initialized model weights and represents detector parameters for all C base classes, and $\boldsymbol{f}_{\boldsymbol{\psi}}$ is trained by sampling episodes from C base classes to simulate testing. An episode includes few shots $\boldsymbol{x}_s$ from the K sampled base classes, which is a subset of the C base classes. In a training step, we sample two images from the first of the K sampled classes as queries. Because the query images $\boldsymbol{x}_q$ belong to one class in C base classes, so they can also act as mini-batch samples of the conventional training. That is why they can also be used to train the reference detectors $\boldsymbol{\theta}$, and thus simultaneous training is achieved.

In meta-testing, we sample K-way, N-shot tasks on novel classes to evaluate the performance of our learned model. As in Fig. 1 (b), $\boldsymbol{f}_{\boldsymbol{\psi}}$ is fixed and applied to novel classes. It uses few shots from novel classes as inputs and predicts detector parameters for them. If with finetuning, the predicted parameters are further finetuned for several epochs with these samples to improve performance. Beside the traditional detection loss, the orthogonality regularization loss is included to increase the discriminative ability of the finetuned detectors:

$$\mathcal{L}_{\mathrm{ft}} = \mathcal{L}_{\mathrm{d}} + \beta\mathcal{L}_{\mathrm{o}}, \tag{11}$$

where β is introduced to weight the regularization term. We have also tried to add orthogonality during meta-training, but it leads to degraded performance. We think both the reference guidance during meta-training and the orthogonality term during meta-testing act as regularization for the generated detectors, excessive constraints in meta-training may cause optimization difficulty.

Specifically, during *training*, we adopt the 5-way 5-shot setting, where the detector generator $\boldsymbol{f}_{\boldsymbol{\psi}}$ predicts a detector for each support instance (cropped object). Then the adaptive pooling module uses these detectors to obtain an aggregated one, which is convolved with the query image feature map to produce the detection results. During *one-shot testing* where only a single support instance is provided, the generated detector from this instance is directly convolved with the query image, since our adaptive pooling module is generally applicable to any-shot and naturally degrades to have no effect when there is only one shot. In *few-shot testing* where few shots are available, the inference procedure is the same as few-shot training. The adaptive pooling module uses the generated detectors from multiple support instances to form the aggregated one. Note that we generate only classification parameters for each class and make different classes share the regression parameters. The novel classes share the regression parameters with base classes that are learned via conventional training, and the detector generator only generates classification parameters for novel classes during few-shot testing.

Our detector generator $\boldsymbol{f}_{\boldsymbol{\psi}}$ and feature extractor $\boldsymbol{g}_{\boldsymbol{\phi}}$ are *class-agnostic*, they produce discriminative detector parameters and activation maps for *all* classes, respectively. The generated detectors are able to distinguish the novel classes in each episode during testing because we have simulated the few-shot tasks by using the samples from base classes during training. We have sampled numerous few-shot tasks from base classes to train $\boldsymbol{f}_{\boldsymbol{\psi}}$ and $\boldsymbol{g}_{\boldsymbol{\phi}}$, so we expect them to *generalize* well on novel classes. If *without finetuning*, the generated detectors are directly convolved with the query image feature map to predict the detection results as done in the training phase. Intuitively, the support instances of the novel class contain the class information, so the detectors generated from them are also class-aware. Even though the detectors without finetune may not be that discriminative, they are still reasonable enough because $\boldsymbol{f}_{\boldsymbol{\psi}}$ transfers a certain amount of knowledge from base classes to novel classes. If *with finetuning*, we abandon $\boldsymbol{f}_{\boldsymbol{\psi}}$ once the detectors for novel classes are generated. The generated detectors act as good initializations, and then we finetune on novel class few-shot

data with orthogonality regularization to optimize the detectors for better performance.

IV. Experiments

In this section, we first illustrate the details we use to implement GenDet and the dataset we adopt. Then ablation studies are conducted to show the effect of different components in GenDet. Finally, we compare our GenDet with the state-of-the-art methods on three benchmarks.

A. Implementation Details

GenDet is a general few-shot detection method and thus can be adopted with both proposal-free and proposal-based detection models. We first use FCOS [47] because of its higher training speed and memory efficiency when many classes are involved, which is common in few-shot detection. In this case, the detector generator $\boldsymbol{f}_{\psi}$ in Fig. 2 is a ResNet-50 [55] with the last fully connected layer discarded. To obtain the generated detector $\boldsymbol{f}_{\psi}(\boldsymbol{x}_s)$, a support instance $\boldsymbol{x}_s$ goes through all the convolutional layers of ResNet and global average pooling. At last, another 1×1 convolution layer transforms $\boldsymbol{f}_{\psi}(\boldsymbol{x}_s)$ to dimension $D = 256$. The feature extractor $\boldsymbol{g}_{\phi}$ includes a ResNet-50 backbone and FPN [44], the same as the original FCOS. The extracted feature $\boldsymbol{g}_{\phi}(\boldsymbol{x}_q)$ is a $1 \times 1 \times D$ vector on the feature pyramid, responsible for detecting objects of different sizes. Following FCOS, we implement the detector as 3 branches, namely region classification (one binary cross-entropy loss for each class), box regression (IoU loss) and centerness prediction (binary cross-entropy loss).

For comparison with previous state-of-the-arts, we also adapt GenDet on proposal-based Faster R-CNN [8], which most of the existing few-shot detection methods use. In the case of Faster R-CNN, $\boldsymbol{f}_{\psi}$ is a complete ResNet-50 (w/o the last fully connected layer). $\boldsymbol{g}_{\phi}(\boldsymbol{x}_q)$ is obtained via RoI pooling on the feature map (FPN is not used) and region-based CNN. The RoIs are produced by the RPN [8] and the R-CNN is the last 3 residual blocks of ResNet. The detectors are composed of region classification (cross-entropy loss) and box regression (smooth L_1 loss) branch. Also, `background` detectors are jointly learned with the reference detectors $\boldsymbol{\theta}$ to distinguish between object and non-object, as in many-shot Faster R-CNN. In both cases of FCOS and Faster R-CNN, box regression/centerness prediction parameters are shared among classes, we only generate region classification parameters for different classes. This follows the intuition that different classes own distinguished appearances, but their location regression have common characteristics. Also, the orthogonality constraint in Eq. (9) is only applied to classification parameters, as the constraint makes no sense for location regression parameters.

Specifically, for Faster R-CNN based GenDet, the detector generator $\boldsymbol{f}_{\psi}$ is ResNet-50, which is composed of the first conv followed by 4 stages of convs, a 7×7 average pooling and a flatten operation. It takes the cropped object (resized to 224×224) as input and outputs the detector's classification parameters $\widehat{\boldsymbol{W}}_k^{(n)} \in \mathbb{R}^{2048}$ for the n-th shot of the k-th class. The mean detector for the k-th class is the average of $\widehat{\boldsymbol{W}}_k^{(n)}$, $n \in [1, N]$ and can be represented as $\overline{\boldsymbol{W}}_k = \sum_{n=1}^{N} \widehat{\boldsymbol{W}}_k^{(n)}$. Then the fully-connected layer $\mathrm{FC} : \mathbb{R}^{2048} \to \mathbb{R}^{2048}$ maps the mean detector $\overline{\boldsymbol{W}}_k \in \mathbb{R}^{2048}$ to the transformed domain $\overline{\boldsymbol{W}}_k^{\mathrm{fc}} \in \mathbb{R}^{2048}$. The unnormalized importance s_n of the one-shot detector is computed as the inner-product between $\widehat{\boldsymbol{W}}_k^{(n)}$ and $\overline{\boldsymbol{W}}_k^{\mathrm{fc}}$: $s_n = \widehat{\boldsymbol{W}}_k^{(n)} \cdot \overline{\boldsymbol{W}}_k^{\mathrm{fc}}$. Then $\{s_n\}$ are normalized by softmax operator to compute the weights $\{\alpha_n\}$, which are used to obtain the adaptively pooled detector $\widehat{\boldsymbol{W}}_k$ for the k-th class as Eq. (5). The detector is composed of the region classifier and the box regressor. The aggregated classification parameters $\widehat{\boldsymbol{W}} \in \mathbb{R}^{(K+1)\times 2048}$ are stacked by $\widehat{\boldsymbol{W}}_k$ from K classes and task-shared background parameters $\boldsymbol{W}_{\mathrm{bg}} \in \mathbb{R}^{2048}$. We take $\widehat{\boldsymbol{W}}$ as the parameters of 1×1 convolutional layers and convolve them with RoI features to predict the classification scores. Box regression parameters $\boldsymbol{W}_{\mathrm{box}} \in \mathbb{R}^{2048\times 4}$ are shared among classes and act as the parameters of 1×1 convolutional layers which are convolved with RoI features to output the regressed boxes. For FCOS based GenDet, the detector generator $\boldsymbol{f}_{\psi}$ is ResNet-50 similar to that in the Faster R-CNN case, except that after the flatten operation we add a dimension reduction layer to reduce the 2048-D outputs to 256-D to comply with the channel dimension of FPN adopted by FCOS. Accordingly, the fully connected layer in Eq. (6) should also be modified as $\mathrm{FC} : \mathbb{R}^{256} \to \mathbb{R}^{256}$. In FCOS the detector is composed of classification, regression and centerness branches. The classification branch is constituted by K binary classifiers, one for each class. The classification scores are the outputs of 1×1 convolutional layers whose parameters are given by $\widehat{\boldsymbol{W}}_k$ and the bias $b \in \mathbb{R}$ which are shared among classes. The parameters of the box regression branch and centerness prediction branch are $\boldsymbol{W}_{\mathrm{box}} \in \mathbb{R}^{256\times 4}$ and $\boldsymbol{W}_{\mathrm{ctr}} \in \mathbb{R}^{256}$, respectively, and they are shared among classes.

If not specified, we adopt 5-way, 5-shot training episodes, *i.e.*, each training episode consists of 5 randomly sampled classes and 5 support images for each class. 2 images from the first sampled class, which simultaneously constitute a mini-batch for training reference detectors $\boldsymbol{\theta}$ in the conventional way, serve as query images for training $\boldsymbol{f}_{\psi}$ in the episodic way. Object regions from support images are cropped and resized to 224 pixels per side. Query images are resized to 600 pixels for the shorter side and no more than 1000 pixels for the longer side. The model is optimized using stochastic gradient descent (SGD) with 8 GPUs. The weight decay is 10^{-4} and the momentum is set to 0.9. We train the model on base classes for 13 epochs with an initial learning rate of 0.02, which is multiplied by 0.1 at 8 and 11 epochs. As for finetuning, the model trained on the base classes is finetuned for 5 epochs on the novel classes. Hyper-parameters for guided generation loss and orthogonality regularization loss are set to $\alpha = 10^{-2}$ (FCOS, or $\alpha = 10^{-3}$ for Faster R-CNN) and $\beta = 1$ via cross validation. As $\mathcal{L}_{\mathrm{g}}$ in Eq. (8) relates to L_1-norm of large-dim vectors (256-D in FCOS and 2048-D in Faster R-CNN), so small α is to balance the large dimension.

For few-shot detection, we think the meta model should be meta trained on a base dataset which contains a large enough number of classes, so it can learn prior knowledge

TABLE II
EFFECT OF DIFFERENT COMPONENTS ON MAP (%) AT IOU=0.5 WITH IMAGENET BENCHMARK. THE STUDIED MODEL IS GENDET + FCOS. "ADAPOOL" IS SHORT FOR THE ADAPTIVE POOLING MODULE. "$\mathcal{L}_{\mathrm{d}}$" IS THE GENERATED DETECTOR LOSS, "$\mathcal{L}_{\mathrm{r}}$" MEANS THE REFERENCE DETECTOR LOSS, "$\mathcal{L}_{\mathrm{g}}$" DENOTES THE GUIDED GENERATION LOSS AND "$\mathcal{L}_{\mathrm{o}}$" IS THE ORTHOGONALITY REGULARIZATION LOSS.

variants	AdaPool	$\mathcal{L}_{\mathrm{d}}$ in Eq. (2)	$\mathcal{L}_{\mathrm{r}}$ in Eq. (7)	$\mathcal{L}_{\mathrm{g}}$ in Eq. (8)	$\mathcal{L}_{\mathrm{o}}$ in Eq. (9)	finetune w/o	finetune w/
w/o adaptive pooling	✗	✓	✓	✓	✓	65.9	76.5
w/o reference detectors	✓	✓	✗	✗	✓	63.5	72.7
w/o guided training	✓	✓	✓	✗	✓	65.6	76.6
w/o episodic training	✗	✗	✓	✗	✓	51.2	68.5
w/o orthogonality term	✓	✓	✓	✓	✗	67.3	75.8
our full model	✓	✓	✓	✓	✓	**67.3**	**77.8**

TABLE III
EFFECT OF DIFFERENT COMPONENTS ON MAP (%) AT IOU=0.5. THE STUDIED MODEL IS GENDET + FASTER R-CNN. RESULTS OF 5-WAY, 5-SHOT ON THE FIRST CLASS SPLIT OF VOC ARE SHOWN.

variants	mAP
share backbone	56.6
w/o adaptive pooling	55.4
w/o guided training	55.3
w/o episodic training	48.5
w/o orthogonality	54.3
stage-wise training	36.3
naïve finetuning	34.2
our full model	**57.7**

that generalizes well to novel classes. ImageNet [2], which is a large-scale benchmark for image classification and object detection with thousands of classes and millions of images, is an ideal testbed for evaluating few-shot detection methods. RepMet [16] proposed a few-shot detection benchmark based on ImageNet, where the first 101 classes (mostly animals and birds species) from ImageNet's 1000 classes are used as base classes for meta training, and 214 classes from the same concept domain are used as novel classes for meta testing. In each few-shot detection task, N randomly sampled support instances and 10 query images for each of the 5 classes are used for evaluation. Three different shot settings $N \in \{1, 5, 10\}$ are included in this benchmark. For ImageNet, the backbone is pretrained on COCO [4] as done in [16]. We use this benchmark for our ablation studies with $N = 5$ and we carry out 100 few-shot detection tasks in our testing procedure. We follow the standard VOC metric and report mean average precision (mAP) with intersection-over-union (IoU) at 0.5. The mAP metric is computed after aggregating the detection results from all 100 tasks.

B. Ablation studies

In Tab. II we verify the effectiveness of different components we proposed above for few-shot detection. Comparing the first and last row, we note that replacing average pooling with the adaptive pooling effectively suppresses noise of the support instances, and thus leads to better performance.

If "w/o reference detectors" $\boldsymbol{\theta}$, the whole framework degrades to a few-shot detection model only using episodic training (or meta-learning). When *w/o finetuning* on novel classes, due to the absence of training guidance for both $\boldsymbol{g}_{\phi}$ and $\boldsymbol{f}_{\psi}$, the feature extractor $\boldsymbol{g}_{\phi}$ cannot learn representations that are discriminative enough, also the detector generator $\boldsymbol{f}_{\psi}$ cannot generate detectors that match the reference detectors trained in the large-sample setting. So transferring $\boldsymbol{g}_{\phi}$ and $\boldsymbol{f}_{\psi}$ to novel classes will not be as effective as training with guidance from reference detectors $\boldsymbol{\theta}$. As for *w/ finetuning* case, because the generated detectors $\widehat{\boldsymbol{W}}$ are not strong enough, which are used to initialize the detectors $\boldsymbol{W}$ for novel classes, the performance is also lower than "our full model".

For "w/o guided training", we do use the reference detectors $\boldsymbol{\theta}$. However, we only use it to guide the training of $\boldsymbol{g}_{\phi}$ but not that of $\boldsymbol{f}_{\psi}$ (the same as setting $\alpha = 0$). Then the discriminative ability of feature extractor $\boldsymbol{g}_{\phi}$ is largely boosted and thus leads to higher mAP. However, as the guidance is not applied for training the detector generator $\boldsymbol{f}_{\psi}$, the generated detectors are less effective so the mAP is still inferior to "our full model".

In the case of "w/o episodic training", during meta training the detector generator is not optimized directly through the generated detector loss $\mathcal{L}_{\mathrm{d}}$. Although we can still learn a reasonable detector generator by matching the generated detectors to the reference detectors, the performance drops dramatically especially without finetuning due to not simulating the testing scenario, which verifies the necessity of episodic training.

The variant "w/o orthogonality term" (the same as setting $\beta = 0$) is also inferior w.r.t. *with finetuning* mAP. As we only employ the orthogonality in meta testing with finetuning, so it does not influence the performance of the w/o finetuning case. During finetuning, novel class detectors $\boldsymbol{W}$ are initialized by the generated detectors $\widehat{\boldsymbol{W}}$, and the number of training samples is small. As the generated detectors for novel classes may be similar because different classes (such as two species of dogs) may have similar appearances, it's essential to add the orthogonality constraint to detectors $\boldsymbol{W}$. In Fig. 4 we visualize the correlation between detectors of novel classes during finetuning at the beginning of each epoch (iter), demonstrating that the regularization loss effectively increases orthogonality and thus discriminative ability.

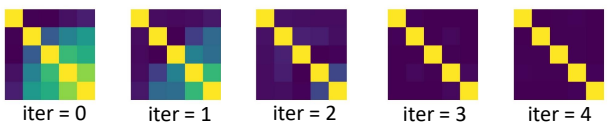

Fig. 4. Visualization of the correlation between detectors of different classes when finetuning. Darker color denotes lower correlation and vice versa.

To be consistent with previous works, we also conduct ablation studies on the VOC [3] dataset. VOC 2007 and 2012

TABLE IV
COMPARISONS ON IMAGENET WITH MAP (%) AT IOU=0.5 UNDER VOC'S METRIC.

method	model	backbone	FPN [44]	DCN [56]	w/o finetune			w/ finetune		
					N=1	5	10	1	5	10
RepMet [16]	Faster R-CNN	ResNet-101	✓	✓	56.9	68.8	71.5	59.2	73.9	79.2
GenDet (Ours)	FCOS	ResNet-50	✓	✗	55.1	62.4	64.4	59.9	75.0	81.2
	Faster R-CNN	ResNet-50	✗	✗	**64.8**	**75.9**	**77.6**	**65.7**	**81.2**	**85.4**

contain 20 categories and tens of thousands of images. Kang *et al.* [17] randomly select 5 categories from VOC as the novel classes and leave the remaining 15 classes as the base classes. The evaluation is done on three different random class splits they propose, and each class split is evaluated with different shot settings $N \in \{1, 2, 3, 5, 10\}$. The results shown in Tab. III (5-way 5-shot on the first class split) draw similar conclusions as those of ImageNet. We also tried "stage-wise training" similar as in MetaDet [20], *i.e.*, first train the feature extractor $\boldsymbol{g}_{\phi}$ and reference detectors $\boldsymbol{\theta}$ with many shots, then fix them and use sampled few-shot tasks to train $\boldsymbol{f}_{\psi}$ with reference guidance in the second stage. This leads to much worse performance than "our full model". We think our end-to-end training can better overcome the domain-shift problem from many-shot to few-shot because of joint optimization of $\boldsymbol{f}_{\psi}$, $\boldsymbol{g}_{\phi}$ and $\boldsymbol{\theta}$. Moreover, we provide the result of "naïve finetuning", which finetunes the model trained on base classes with few shots from novel classes. It leads to poor performance as expected resulting from over-fitting.

TABLE V
COMPARE GENDET+FASTER R-CNN WITH STATE-OF-THE-ART ON VOC BY MAP (%) AT IOU=0.5 UNDER VOC'S METRIC. THE SUPPORT DATA IS THE SAME AS THAT PROPOSED BY [17] AND THE RESULTS ARE FOR THE FIRST CLASS SPLIT.

method	backbone	N=1	2	3	5	10
Reweight [17]	DarkNet-19	14.8	15.5	26.7	33.9	47.2
Meta R-CNN [18]	ResNet-101	19.9	25.5	35.0	45.7	51.5
GenDet (Ours)	ResNet-50	**40.7**	**48.1**	**52.5**	**57.5**	**62.4**

TABLE VI
COMPARE GENDET+FASTER R-CNN WITH STATE-OF-THE-ART ON VOC BY MAP (%) AT IOU=0.5 UNDER VOC'S METRIC. RESULTS FOR BOTH BASE AND NOVEL CLASSES ARE REPORTED.

method	backbone	N=3		10	
		base	novel	base	novel
Reweight [17]	DarkNet-19	64.8	26.7	63.6	47.2
Meta R-CNN [18]	ResNet-101	64.8	35.0	67.9	51.5
GenDet (Ours)	ResNet-50	**68.4**	**52.5**	**69.3**	**62.4**

C. Compare with state-of-the-art

Results on ImageNet. To compare fairly and avoid testing instability, we conduct exactly the same 500 few-shot detection tasks used by RepMet [16] and the final performance (mAP@0.5) is computed with the results aggregated from all the 500 episodes. Tab. IV summarizes the comparison between GenDet and RepMet. GenDet based on FCOS is inferior to RepMet in the w/o finetuning case, but is superior to RepMet remarkably when w/ finetuning. Note that RepMet deploys a much more powerful backbone ResNet-101 with deformable convolutional networks [56]. Moreover, GenDet based on Faster R-CNN outperforms RepMet significantly both w/ and w/o finetuning. We think that GenDet outperforms RepMet mainly because of episodic training, showing the necessity of "training as testing" in few-shot object detection.

Results on VOC. We note that previous works on the VOC benchmark use different detection models and backbones, *i.e.*, Feature Reweighting [17] uses YOLOv2 [7] with DarkNet-19, Meta R-CNN [18] uses Faster R-CNN with ResNet-50/101, TFA [19] uses Faster R-CNN with ResNet-101 and MetaDet [20] uses Faster R-CNN with DarkNet-19/VGG-16. For fair comparisons, we implement GenDet based on Faster R-CNN with VGG-16/ResNet-50/101 as the backbone. Consistent with previous methods [17], [18], [19], [20], we use the combination of VOC2007 and VOC2012 trainval set for training on base classes. The support set is chosen from VOC0712 trainval set and the query set is composed of all the images from VOC07 test set. As previous methods on VOC all adopt the w/ finetuning setting, we also report the results of w/ finetuning. First, we use exactly the same support training data as [17], [18] for fair comparison and the results are shown in Tab. V. We further provide the results of both base and novel classes in Tab. VI. As shown, our method significantly outperforms [17], [18] in terms of mAP for both base and novel classes. [17], [18] adopt a class-agnostic detection head on the class attentive feature maps, while GenDet generates class-specific detection heads, which give the model more capability to detect novel classes.

As noted by [19], [20], different support data usually leads to fluctuated results. So to ensure the testing stability, we conduct 20 trials for each class split and report the average mAP. The results are in Tab. VII where the backbone is pretrained on ImageNet. The same three base/novel class splits as [17] are used for fair comparisons. Our GenDet significantly outperforms the other few-shot object detection methods under various backbones and class splits, showing its effectiveness. Compared with [19], [20] which learn novel class detectors from randomly initialized weights, our method finetunes the generated detectors which are good initializations. The better starting point makes it more likely to converge to optimal in a few epochs before overfitting.

Results on COCO. COCO [4] has 80k training images, 40k validation images and 20k testing images from 80 classes, covering all categories in VOC. In general, 5k images from validation set are used for evaluation and the left images in the training and validation set are combined for training, namely trainval35k. Kang *et al.* [17] choose the 20 categories from COCO that also appear in VOC as the novel classes and leave the remaining 60 classes to be the base classes. Compared with VOC, COCO is more challenging for detection as there exists

TABLE VII
COMPARE GENDET+FASTER R-CNN WITH STATE-OF-THE-ART ON VOC BY MAP (%) AT IOU=0.5 UNDER VOC'S METRIC, THE SAME THREE CLASS SPLITS ARE USED IN DIFFERENT METHODS. ONLY TFA MODELS USE MULTI-SCALE TRAINING AND FEATURE PYRAMID NETWORKS.

method	backbone	split 1					split 2					split 3				
		N=1	2	3	5	10	1	2	3	5	10	1	2	3	5	10
Reweight [17]	DarkNet-19	14.8	15.5	26.7	33.9	47.2	15.7	15.3	22.7	30.1	39.2	19.2	21.7	25.7	40.6	41.3
Meta R-CNN [18]	ResNet-101	19.9	25.5	35.0	45.7	51.5	10.4	19.4	29.6	34.8	45.4	14.3	18.2	27.5	41.2	48.1
TFA w/ fc [19]	ResNet-101	36.8	29.1	43.6	55.7	57.0	18.2	29.0	33.4	35.5	39.0	27.7	33.6	42.5	48.7	50.2
TFA w/ cos [19]	ResNet-101	**39.8**	36.1	44.7	55.7	56.0	23.5	26.9	34.1	35.1	39.1	30.8	34.8	42.8	49.5	49.8
MetaDet [20]	DarkNet-19	17.1	19.1	28.9	35.0	48.8	18.2	20.6	25.9	30.6	41.5	20.1	22.3	27.9	41.9	42.9
MetaDet [20]	VGG-16	18.9	20.6	30.2	36.8	49.6	21.8	23.1	27.8	31.7	43.0	20.6	23.9	29.4	43.9	44.1
GenDet (Ours)	VGG-16	30.8	37.2	43.0	46.6	54.3	22.5	23.5	29.1	33.2	43.6	27.1	33.1	37.8	44.3	49.7
GenDet (Ours)	ResNet-50	38.5	**47.1**	**52.2**	**57.7**	**63.5**	**26.8**	**34.0**	**37.3**	**42.8**	**48.3**	**33.4**	**40.0**	**44.3**	**51.2**	**56.5**

TABLE VIII
COMPARE GENDET+FASTER R-CNN WITH STATE-OF-THE-ART ON COCO BY MAP (%) AT DIFFERENT IOU THRESHOLDS AND OBJECT AREAS UNDER COCO'S METRIC. ONLY TFA MODELS USE MULTI-SCALE TRAINING AND FEATURE PYRAMID NETWORKS.

method	backbone	N=10						30					
		AP@IoU			AP@area			AP@IoU			AP@area		
		0.5:0.95	0.5	0.75	S	M	L	0.5:0.95	0.5	0.75	S	M	L
Reweight [17]	DarkNet-19	5.6	12.3	4.6	0.9	3.5	10.5	9.1	19.0	7.6	0.8	4.9	16.8
Meta R-CNN [18]	ResNet-50	8.7	**19.1**	6.6	2.3	7.7	14.0	12.4	25.3	10.8	2.8	11.6	19.0
TFA w/ fc [19]	ResNet-101	10.0	-	9.2	-	-	-	13.4	-	13.2	-	-	-
TFA w/ cos [19]	ResNet-101	**10.0**	-	9.3	-	-	-	13.7	-	13.4	-	-	-
MetaDet [20]	VGG-16	7.1	14.6	6.1	1.0	4.1	12.2	11.3	21.7	8.1	1.1	6.2	17.3
GenDet (Ours)	VGG-16	7.3	14.9	6.3	1.4	5.9	13.0	11.4	22.7	10.0	3.2	9.4	19.2
GenDet (Ours)	ResNet-50	9.2	17.7	8.8	3.3	7.7	14.6	14.0	26.7	13.2	4.4	12.1	23.3
GenDet (Ours)	ResNet-101	9.9	18.8	**9.6**	**3.6**	**8.4**	**15.4**	**14.3**	**27.5**	**13.8**	**4.8**	**13.0**	**24.2**

TABLE IX
COMPARE GENDET+FASTER R-CNN WITH STATE-OF-THE-ART ON COCO $\rightarrow$ VOC WITH MAP (%) AT IOU=0.5 UNDER VOC'S METRIC.

method	backbone	mAP
Reweight [17]	DarkNet-19	32.3
Meta R-CNN [18]	ResNet-50	37.4
MetaDet [20]	VGG-16	34.0
GenDet (Ours)	VGG-16	35.4
GenDet (Ours)	ResNet-50	**39.4**

more small and occluded objects. For COCO benchmark, we follow the standard evaluation metric on COCO and report the mAP averaged over different IoU thresholds from 0.5 to 0.95, also the mAP over different object areas. The few-shot detection tasks are constructed as 20 way, $N \in \{10, 30\}$ shot tasks. The same as on VOC, GenDet is also based on Faster R-CNN with different backbones. Tab. VIII shows our GenDet achieves the best performance, demonstrating it also generalizes well to the challenging dataset.

Results of Cross Dataset. Considering in practice the base classes and novel classes may come from different domains, we also compare the performance with previous works in the COCO $\rightarrow$ VOC cross dataset setting with $N = 10$. The meta model is trained with COCO images of the 60 COCO classes which are absent in VOC, while to be tested on the VOC images of the 20 VOC classes. Tab. IX shows that GenDet generalizes best in the cross dataset setting.

V. CONCLUSIONS

In this work we propose GenDet, a detector generation model for few-shot object detection. We train the detector generator by sampling few-shot detection tasks simulating the testing scenario. An adaptive pooling module is proposed to aggregate information from multiple shots by suppressing the noisy instances. The reference detectors which are trained in the conventional way are introduced to guide the training of the detector generator. Our detector generator and reference detectors can be trained simultaneously in a single stage, benefited from end-to-end training. Moreover, the generated detectors are constrained to be orthogonal to each other for better generalization. With extensive experiments on different benchmarks, we show the superiority of our proposed method over previous state-of-the-arts.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.

[2] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[3] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[4] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014, pp. 740–755.

[5] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*. Cham: Springer International Publishing, 2016, pp. 21–37.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[9] L. Liu, Z. Kuang, Y. Chen, J. Xue, W. Yang, and W. Zhang, "Incdet: In defense of elastic weight consolidation for incremental object detection," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2020, DOI: 10.1109/TNNLS.2020.3002583.

[10] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, pp. 3630–3638.

[11] L. Bertinetto, J. a. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," in *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016, pp. 523–531.

[12] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 1126–1135.

[13] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017, pp. 4077–4087.

[14] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1199–1208.

[15] H. Chen, Y. Wang, G. Wang, and Y. Qiao, "Lstd: A low-shot transfer detector for object detection," in *AAAI Conference on Artificial Intelligence*, 2018.

[16] L. Karlinsky, J. Shtok, S. Harary, E. Schwartz, A. Aides, R. Feris, R. Giryes, and A. M. Bronstein, "Repmet: Representative-based metric learning for classification and few-shot object detection," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5192–5201.

[17] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, "Few-shot object detection via feature reweighting," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 8419–8428.

[18] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin, "Meta r-cnn: Towards general solver for instance-level low-shot learning," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9576–9585.

[19] X. Wang, T. E. Huang, T. Darrell, J. E. Gonzalez, and F. Yu, "Frustratingly simple few-shot object detection," in *International Conference on Machine Learning (ICML)*, 2020.

[20] Y. Wang, D. Ramanan, and M. Hebert, "Meta-learning to detect rare objects," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9924–9933.

[21] Y.-X. Wang and M. Hebert, "Learning to learn: Model regression networks for easy small sample learning," in *Computer Vision – ECCV 2016*. Cham: Springer International Publishing, 2016, pp. 616–634.

[22] X. Li, Z. Sun, J.-H. Xue, and Z. Ma, "A concise review of recent few-shot meta-learning methods," *Neurocomputing*, 2020, DOI: https://doi.org/10.1016/j.neucom.2020.05.114.

[23] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2, 2015.

[24] D. Das and C. G. Lee, "A two-stage approach to few-shot learning for image recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 3336–3350, 2019.

[25] H.-G. Jung and S.-W. Lee, "Few-shot learning with geometric constraints," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[26] S. Rahman, S. Khan, and F. Porikli, "A unified approach for conventional zero-shot, generalized zero-shot, and few-shot learning," *IEEE Transactions on Image Processing*, vol. 27, no. 11, pp. 5652–5667, 2018.

[27] S. Gidaris and N. Komodakis, "Dynamic few-shot visual learning without forgetting," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4367–4375.

[28] Y. Wang and M. Hebert, "Model recommendation: Generating object detectors from few samples," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1619–1628.

[29] N. Passalis, A. Iosifidis, M. Gabbouj, and A. Tefas, "Hypersphere-based weight imprinting for few-shot learning on embedded devices," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[30] X.-S. Wei, P. Wang, L. Liu, C. Shen, and J. Wu, "Piecewise classifier mappings: Learning fine-grained learners for novel categories with few examples," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 6116–6125, 2019.

[31] Q. Cai, Y. Pan, T. Yao, C. Yan, and T. Mei, "Memory matching networks for one-shot image recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4080–4088.

[32] T. Munkhdalai and H. Yu, "Meta networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. International Convention Centre, Sydney, Australia: PMLR, 06–11 Aug 2017, pp. 2554–2563.

[33] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning*. JMLR.org, 2016, pp. 1842–1850.

[34] B. Hariharan and R. Girshick, "Low-shot visual recognition by shrinking and hallucinating features," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 3037–3046.

[35] Y. Wang, R. Girshick, M. Hebert, and B. Hariharan, "Low-shot learning from imaginary data," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7278–7286.

[36] Li Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 4, pp. 594–611, 2006.

[37] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[38] R. Salakhutdinov, J. Tenenbaum, and A. Torralba, "One-shot learning with a hierarchical nonparametric bayesian model," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, ser. Proceedings of Machine Learning Research, vol. 27. Bellevue, Washington, USA: PMLR, 02 Jul 2012, pp. 195–206.

[39] J. Lu, S. Jin, J. Liang, and C. Zhang, "Robust few-shot learning for user-provided data," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[40] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 845–869, 2013.

[41] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[43] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2999–3007.

[44] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 936–944.

[45] X. Wang, S. Zhang, Z. Yu, L. Feng, and W. Zhang, "Scale-equalizing pyramid convolution for object detection," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 13 356–13 365.

[46] C. Zhu, Y. He, and M. Savvides, "Feature selective anchor-free module for single-shot object detection," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 840–849.

[47] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9626–9635.

[48] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," *International Journal of Computer Vision*, vol. 128, no. 3, pp. 642–656, 2020.

[49] X. Dong, L. Zheng, F. Ma, Y. Yang, and D. Meng, "Few-example object detection with model communication," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1641–1654, 2018.

[50] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 2020.

[51] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 815–823.

[52] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European Conference on Computer Vision*. Cham: Springer, 2016, pp. 850–865.

[53] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *2018 IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, 2018, pp. 8971–8980.

[54] J. Yang, P. Ren, D. Zhang, D. Chen, F. Wen, H. Li, and G. Hua, "Neural aggregation network for video face recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5216–5225.

[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[56] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 764–773.

Liyang Liu received the B.Eng. degree in electronic engineering from Tsinghua University, Beijing, China, in 2014, where he is currently pursuing the Ph.D. degree with the department of electronic engineering. His research interests include computer vision and object detection.

Bochao Wang received his M.S. and B.S. degrees from Sun Yat-sen University, China. This work was finished when he was in SenseTime. His research interests include object detection, image generation and automated machine learning.

Zhanghui Kuang received his B.S. degree from Sun Yat-Sen University, Guangzhou, China, in 2009, and Ph.D. degree from The University of Hong Kong in 2014. He is currently a Research Director in SenseTime Group Limited. His research interests include deep learning and computer vision.

Jing-Hao Xue received his Dr.Eng. degree in signal and information processing from Tsinghua University in 1998 and Ph.D. degree in statistics from the University of Glasgow in 2008. He is an Associate Professor in the Department of Statistical Science, University College London. His research interests include statistical machine learning, high-dimensional data analysis, pattern recognition and image analysis.

Yimin Chen received his bachelor and master degrees in biomedical engineering from Huazhong University of Science and Technology in 2009 and 2012 respectively, his Ph.D. degree in electronic engineering from City University of Hong Kong in 2016. He is currently a senior researcher in SenseTime. His research focuses on computer vision and pattern recognition.

Wenming Yang received his Ph.D. degree in information and communication engineering from Zhejiang University in 2006. He is an Associate Professor in the International Graduate School at Shenzhen/Department of Electronic Engineering, Tsinghua University. His research interests include image processing, pattern recognition, computer vision and artificial intelligence in medicine.

Qingmin Liao received his Ph.D. degree in signal processing and telecommunications from the University of Rennes 1, France, in 1994. He is a Professor in the Department of Electronic Engineering of Tsinghua University, in 2002. Since 2005, he has been the Director of the Laboratory of Visual Information Processing (VIP Lab) in the Graduate School at Shenzhen, Tsinghua University. His research interests include image/video processing, transmission, analysis, biometrics and their applications.

Wayne Zhang received the B.Eng. degree in electronic engineering from the Tsinghua University, Beijing, China, in 2007, the M.Phil. degree in 2009, and Ph.D. degree in 2012, both in information engineering from the Chinese University of Hong Kong. He is currently a Senior Research Director in SenseTime Group Limited. He served as an EXCO member of AI Specialist Group of Hong Kong Computer Society from 2018 to 2019. He was ranked 6th data scientist by Kaggle, one of the largest data science community in the world, in 2012. His research interests include deep learning and computer vision.