PCL: Proposal Cluster Learning for Weakly Supervised Object Detection

Peng Tang, Xinggang Wang, *Member, IEEE,* Song Bai, Wei Shen, Xiang Bai, *Senior Member, IEEE,* Wenyu Liu, *Senior Member, IEEE,* and Alan Yuille, *Fellow, IEEE*

Abstract—Weakly Supervised Object Detection (WSOD), using only image-level annotations to train object detectors, is of growing importance in object recognition. In this paper, we propose a novel deep network for WSOD. Unlike previous networks that transfer the object detection problem to an image classification problem using Multiple Instance Learning (MIL), our strategy generates proposal clusters to learn refined instance classifiers by an iterative process. The proposals in the same cluster are spatially adjacent and associated with the same object. This prevents the network from concentrating too much on parts of objects instead of whole objects. We first show that instances can be assigned object or background labels directly based on proposal clusters for instance classifier refinement, and then show that treating each cluster as a small new bag yields fewer ambiguities than the directly assigning label method. The iterative instance classifier refinement is implemented online using multiple streams in convolutional neural networks, where the first is an MIL network and the others are for instance classifier refinement supervised by the preceding one. Experiments are conducted on the PASCAL VOC, ImageNet detection, and MS-COCO benchmarks for WSOD. Results show that our method outperforms the previous state of the art significantly.

Index Terms—Object detection, weakly supervised learning, convolutional neural network, multiple instance learning, proposal cluster.

1 Introduction

Object detection is one of the most important problems in computer vision with many applications. Recently, due to the development of Convolutional Neural Network (CNN) [1], [2] and the availability of large scale datasets with detailed boundingbox-level annotations [3], [4], [5], there have been great leap forwards in object detection [6], [7], [8], [9], [10], [11]. However, it is very labor-intensive and time-consuming to collect detailed annotations, whereas acquiring images with only image-level annotations (*i.e.*, image tags) indicating whether an object class exists in an image or not is much easier. For example, we can use image search queries to search on the Internet (*e.g.*, Google and Flickr) to obtain a mass of images with such image-level annotations. This fact inspires us to explore methods for the Weakly Supervised Object Detection (WSOD) problem, *i.e.*, training object detectors with only image tag supervisions.

Many previous methods follow the Multiple Instance Learning (MIL) pipeline for WSOD [12], [13], [14], [15], [16], [17], [18], [19]. They treat images as bags and proposals as instances; then instance classifiers (object detectors) are trained under MIL constraints (*i.e.*, a positive bag contains at least one positive instance and all instances in negative bags are negative). In addition, inspired by the great success of CNN, recent efforts often combine MIL and CNN to obtain better WSOD performance. Some researches have shown that treating CNNs pre-trained on large scale datasets as off-the-shelf proposal feature extractors can obtain much better performance than traditional hand-designed features [12], [13], [14], [15]. Moreover, many recent works have achieved even better results for WSOD by an MIL network using standard end-to-end training [16], [18] or a variant of end-to-end training [17], [19]. See Section 2.3 for this variant of end-to-end and how it differs from the standard one. We use the same strategy of training a variant of end-to-end MIL network inspired by [17], [19].

Although some promising results have been obtained by MIL networks for WSOD, they do not perform as well as fully supervised ones [6], [7], [8]. As shown in Fig. 3 (a), previous MIL networks integrate the MIL constraints into the network training by transferring the instance classification (object detection) problem to a bag classification (image classification) problem, where the final image scores are the aggregation of the proposal scores. However, there is a big gap between image classification and object detection. For classification, even parts of objects can contribute to correct results (*e.g.*, the red boxes in Fig. 1), because important parts include many characteristics of the objects. Many proposals only cover parts of objects, and "seeing" proposals only of parts may be enough to roughly localize the objects. But this may not localize objects well enough considering the performance requirement of high Intersection-over-Union (IoU) between the resulting boxes and groundtruth bound-

- *P. Tang, X. Wang, X. Bai, and W. Liu are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, 430074 China. E-mail: {pengtang, xgwang, xbai, liuwy}@hust.edu.cn*
- *S. Bai is with the Department of Engineering Science, University of Oxford, Oxford, OX1 3PJ, UK. E-mail: songbai.site@gmail.com*
- *W. Shen is with the Key Laboratory of Specialty Fiber Optics and Optical Access Networks, Shanghai Institute for Advanced Communication and Data Science, School of Communication and Information Engineering, Shanghai University, Shanghai, 200444 China. E-mail: shenwei1231@gmail.com*
- *A. Yuille is with the Departments of Cognitive Science and Computer Science, Johns Hopkins University, Baltimore, MD 21218-2608 USA. E-mail: alan.l.yuille@gmail.com*

Fig. 1. Different proposals cover different parts of objects. All these proposals can be classified as "bird" but only the green boxes, which have enough IoU with groundtruth, contribute to correct detections.

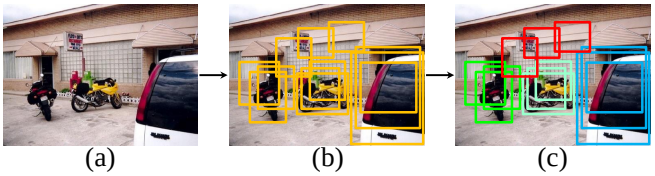


Fig. 2. The proposals (b), of an image (a), can be grouped into different proposal clusters (c). Proposals with the same color in (c) belong to the same cluster (red indicates background).

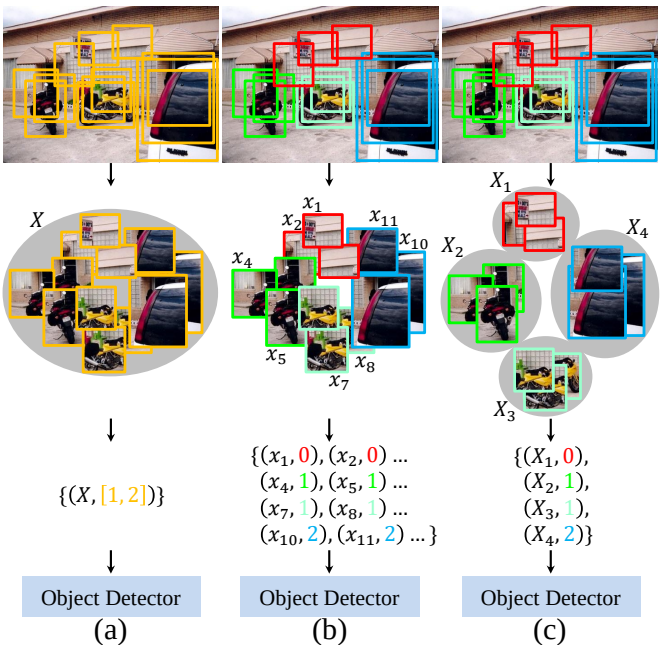


Fig. 3. (a) Conventional MIL networks transfer the instance classification (object detection) problem to a bag classification (image classification) problem. (b) We propose to generate proposal clusters and assign proposals the label of the corresponding object class for each cluster. (c) We propose to treat each proposal cluster as a small new bag. "0", "1", and "2" indicate the "background", "motorbike", and "car", respectively.

ingboxes: the top ranking proposals may only localize parts of objects instead of whole objects. Recall that for detection, the resulting boxes should not only give correct classification, but also localize objects and have enough overlap with groundtruth boundingboxes (*e.g.*, the green boxes in Fig. 1).

Before presenting our solution of the problem referred above, we first introduce the concept of proposal cluster. Object detection requires algorithms to generate multiple overlapping proposals closely surrounding objects to ensure high proposal recall (*e.g.*, for each object, there are tens of proposals on average from Selective Search [20] which have IoU>0.5 with the groundtruth boundingbox on the PASCAL VOC dataset). Object proposals in an image can be grouped into different spatial clusters. Except for one cluster for background proposals, each object cluster is associated with a single object and proposals in each cluster are spatially adjacent, as shown in Fig. 2. For fully supervised object detection (*i.e.*, training object detectors using boundingbox-level annotations), proposal clusters can be generated by treating the groundtruth boundingboxes as cluster centers. Then object detectors are trained according to the proposal clusters (*e.g.*, assigning all proposals the label of the corresponding object class for each cluster). This alleviates the problem that detectors may only focus on parts.

But in the weakly supervised scenario, it is difficult to generate proposal clusters because groundtruth boundingboxes that can be used as cluster centers are not provided. To cope with this difficulty, we suggest to find proposal clusters as follows. First we generate proposal cluster centers from those proposals which have high classification scores during training, because these top ranking proposals can always detect at least parts of objects. That is, for each image, after obtaining proposal scores, we select some proposals with high scores as cluster centers, and then proposal clusters are generated based on spatial overlaps with the cluster centers. Then the problem reduces to how to select proposals as centers, because many high scoring proposals may correspond to the same object. The most straightforward way is to choose the proposal with the highest score for each positive object class (*i.e.*, the object class exists in the image) as the center. But such a method ignores the fact that there may exist more than one object with the same object category in natural images (*e.g.*, the two motorbikes in Fig. 2). Therefore, we propose a graph-based method to find cluster centers. More specifically, we build a graph of top ranking proposals according to the spatial similarity for each positive object class. In the graph, two proposals are connected if they have enough spatial overlaps. Then we greedily and iteratively choose the proposals which have most connections with others to estimate the centers. Although a cluster center proposal may only capture an object partially, its adjacent proposals (*i.e.*, other proposals in the cluster) can cover the whole object, or at worst contain larger parts of the object.

Based on these proposal clusters, we propose two methods to refine instance classifiers (object detectors) during training. We first propose to assign proposals object labels directly. That is, for each cluster, we assign its proposals the label of its corresponding object class, as in Fig. 3 (b). Compared with the conventional MIL network in Fig. 3 (a), this strategy forces network to "see" larger parts of objects by assigning object labels to proposals that cover larger parts of objects directly, which fills the gap between classification and detection to some extent. While effective, this strategy still has potential ambiguities, because assigning the same object label to proposals that cover different parts of objects simultaneously may confuse the network and will hurt the discriminative power of the detector. To address this problem, we propose to treat each proposal cluster as a small new bag to train refined instance classifiers, as in Fig. 3 (c). Most of the proposals in these new bags

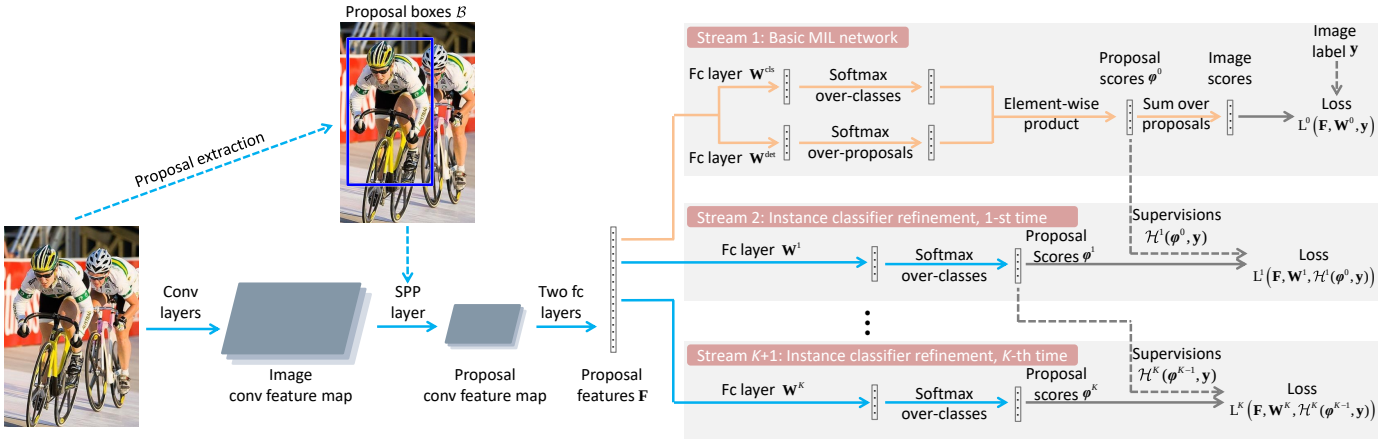


Fig. 4. The architecture of our method. All arrows are utilized during the forward process of training, only the solid ones have back-propagation computations, and only the blue ones are used during testing. During the forward process of training, an image and its proposal boxes are fed into the CNN which involves a series of convolutional layers, an SPP layer, and two fully connected layers to produce proposal features. These proposal features are branched into many streams: the first one for the basic MIL network and the other ones for iterative instance classifier refinement. Each stream outputs a set of proposal scores and generates proposal clusters consequently. Based on these proposal clusters, supervisions are generated to compute losses for the next stream. During the back-propagation process of training, proposal features and classifiers are trained according to the network losses. All streams share the same proposal features.

should have relatively high classification scores because the cluster centers covers at least parts of objects and proposals in the same cluster are spatially adjacent (except for the background cluster). In the same time, not all proposals in the bags should have high classification scores. Thus compared with the directly assigning label strategy, this strategy is more flexible and can reduce the ambiguities to some extent. We name our method Proposal Cluster Learning (PCL) because it learns refined instance classifiers based on proposal clusters.

To implement our idea effectively and efficiently, we further propose an online training approach. Our network has multiple output streams as in Fig. 4. The first stream is a basic MIL network which aggregates proposal scores into final image scores to train basic instance classifiers, and the other streams refine the instance classifiers iteratively. During the forward process of training, proposal classification scores are obtained and proposal clusters are generated consequently for each stream. Then based on these proposal clusters, supervisions are generated to compute losses for the next stream. According to the losses, these refined classifiers are trained during back-propagation. Except for the first stream that is supervised by image labels, the other streams are supervised by the image labels as well as outputs from their preceding streams. As our method forces the network to "see" larger parts of objects, the detector can discover the whole object instead of parts gradually by performing refinement multiple times (*i.e.*, multiple output streams). But at the start of training, all classifiers are almost untrained, which will result in very noisy proposal clusters, and so the training will deviate from the correct solutions a lot. Thus we design a weighted loss further by associating different proposals with different weights in different training iterations. After that, all training procedures can thus be integrated into a single end-to-end network. This can improve the performance benefiting from our PCL-based classifier refinement procedure. It is also very computational efficient in both training and testing. In addition, performance can be improved by sharing proposal features among different output streams.

We elaborately conduct many experiments on the challenging PASCAL VOC, ImageNet detection, and MS-COCO datasets to confirm the effectiveness of our method. Our method achieves 48.8% mAP and 66.6% CorLoc on VOC 2007 which is more than 5% absolute improvement compared with previous best performed methods.

This paper is an extended version of our previous work [21]. In particular, we give more analyses of our method and enrich literatures of most recent related works, making the manuscript more complete. In addition, we make two methodological improvements: the first one is to generate proposal clusters using graphs of top ranking proposals instead of using the highest scoring proposal, and the second one is to treat each proposal cluster as a small new bag. In addition, we provide more discussions of experimental results, and show the effectiveness of our method on the challenging ImageNet detection and MS-COCO datasets.

The rest of our paper is organized as follows. In Section 2, some related works are introduced. In Section 3, the details of our method are described. Elaborate experiments and analyses are conducted in Section 4. Finally, conclusions and future directions are presented in Section 5.

2 Related Work

2.1 Multiple instance learning

MIL, first proposed for drug activity prediction [22], is a classical weakly supervised learning problem. Many variants have been proposed for MIL [14], [23], [24], [25]. In MIL, a set of bags are given, and each bag is associated with a collection of instances. It is natural to treat WSOD as an MIL problem. Then the problem turns into finding instance classifiers only given bag labels. Our method also follows the MIL strategy and makes several improvements to WSOD. In particular, we learn refined instance classifiers

based on proposal clusters according to both instance scores and spatial relations in an online manner. [1]

MIL has many applications to computer vision, such as image classification [26], [27], weakly supervised semantic segmentation [28], [29], object detection [30], object tracking [31], *etc*. The strategy of treating proposal clusters as bags was partly inspired by [30], [31], where [30] proposes to train MIL for patches around groundtruth locations and [31] proposes to train MIL for patches around predicted object locations. However, they require groundtruth locations for either all training samples [30] or the beginning time frames [31], whereas WSOD does not have such annotations. Therefore, it is much harder to generate proposal clusters only guided by image-level supervisions for WSOD. In addition, we incorporate the strategy of treating proposal clusters as bags into the network training whereas [30], [31] do not. Oquab *et al*. [32] also train a CNN network using the max pooing MIL strategy to localize objects. But their methods can only coarsely localize objects regardless of their sizes and aspect ratios, whereas our method can detect objects more accurately.

2.2 Weakly supervised object detection

WSOD has attracted great interests nowadays because the amount of data with image-level annotations is much bigger and is growing much faster than that with boundingbox-level annotations. Many methods are emerging for the WSOD problem [13], [14], [33], [34], [35], [36], [37], [38], [39], [40]. For example, Chum and Zisserman [33] first initialize object locations by discriminative visual words and then introduce an exemplar model to measure similarity between image pairs for updating locations. Deselaers *et al*. [34] propose to initialize boxes by objectness [41] and use a CRF-based model to iteratively localize objects. Pandey and Lazebnik [35] train a DPM model [42] under weak supervisions for WSOD. Shi *et al*. [36] use Bayesian latent topic models to jointly model different object classes and background. Song *et al*. [38] develop a technology to discover frequent discriminative configurations of visual patterns for robust WSOD. Cinbis *et al*. [13] iteratively train a multi-fold MIL to avoid the detector being locked onto inaccurate local optima. Wang *et al*. [14] relax the MIL constraints into a derivable loss function to train detectors more efficient.

Recently, with the revolution of CNNs in computer vision, many works also try to combine the WSOD with CNNs. Early works treat CNN models pre-trained on ImageNet as off-the-shelf feature extractors [12], [13], [14], [15], [37], [38], [39], [40]. They extract CNN features for each candidate regions, and then train their own detectors on top of these features. These methods have shown that CNN descriptors can boost performance against traditional hand-designed features. More recent efforts tend to train end-to-end networks for WSOD [16], [17], [18], [19]. They integrate the MIL constraints into the network training by aggregating proposal classification scores into final image classification scores, and then image-level supervision can be directly added to image classification scores. For example, Tang *et al*. [16] propose to use max pooling for aggregation. Bilen and Vedaldi [17] develop a weighted sum pooing strategy. Building on [17], Kantorov *et al*. argue that context information can improve the performance. Diba *et al*. [19] show that weakly supervised segmentation map can be used as guidance to filter proposals, and jointly train the weakly supervised segmentation network and WSOD end-to-end. Our method is built on these networks and any of them can be chosen as our basic network. Our strategy proposes to learn refined instance classifiers based on proposal clusters, and propose a novel online approach to train our network effectively and efficiently. Experimental results show our strategies can boost the results significantly.

In addition to the weighted sum pooing, [17] also proposes a "spatial regulariser" that forces features of the highest scoring proposal and its spatially adjacent proposals to be the same. Unlike this, we show that finding proposal cluster centers using graph and treating proposal clusters as bags are more effective. The contemporary work [43] uses a graph model to generate seed proposals. Their network training has many steps: first, an MIL network [44] is trained; second, seed proposals are generated using the graph; third, based on these seed proposals, a Fast R-CNN [7] like detector is trained. Our method differs from [43] in many aspects: first, we propose to generate proposal clusters for each training iteration and thus our network is trained end-to-end instead of step-by-step, which is more efficient and can benefit from sharing proposal features among different streams; second, we propose to treat proposal clusters as bags for training better classifiers. As evidenced by experiments, our method obtains much better and more robust results.

2.3 End-to-end and its variants

In standard end-to-end training, the update requires optimizing losses w.r.t. all functions of network parameters. For example, the Fast R-CNN [7] optimizes their classification loss and boundingbox regression loss w.r.t. proposal classification and feature extraction for fully supervised object detection. The MIL networks in [16], [18] optimize their MIL loss w.r.t. proposal classification and feature extraction for WSOD.

Unlike the standard end-to-end training, there exists a variant of end-to-end training. The variant contains functions which depend on network parameters, but losses are not optimized w.r.t. all these functions [17], [19]. As we described in Section 2.2, the "spatial regulariser" in [17] forces features of the highest scoring proposal and its spatially adjacent proposals to be the same. They use a function of network parameters to compute the highest scoring proposal, and do not optimize their losses w.r.t. this function. Diba *et al*. [19] filter out background proposals using a function of network parameters and use these filtered proposals in their latter network computations. They also do not optimize their losses w.r.t. this function. Inspired by [17], [19], we use this variant of end-to-end training. More precisely, we do not optimize our losses w.r.t. the generated supervisions for instance classifier refinement.

2.4 Others

There are many other important related works that do not focus on weakly supervised learning but should be

1. "Instance" and "proposal" are used interchangeably in this paper.

discussed. Similar to other end-to-end MIL networks, our method is built on top of the Region of Interest (RoI) pooling layer [7] or Spatial Pyramid Pooling (SPP) layer [45] to share convolutional computations among different proposals for model acceleration. But both [7] and [45] require boundingbox-level annotations to train their detectors. The sharing proposal feature strategy in our network is similar to multi-task learning [46]. Unlike the multi-task learning that each output stream has their own relatively independent external supervisions for different tasks, in our method, all streams have the same task and supervisions of later streams depend on the outputs from their preceding streams.

3 METHOD

The overall architecture of our method is shown in Fig. 4. Given an image, about $2,000$ object proposals from Selective Search [20] or EdgeBox [47] are generated. During the forward process of training, the image and these proposals are fed into some convolutional (conv) layers with an SPP layer [45] to produce a fixed-size conv feature map per-proposal. After that, proposal feature maps are fed into two fully connected (fc) layers to produce proposal features. These features are branched into different streams: the first one is an MIL network to train basic instance classifiers and the others refine the classifiers iteratively. For each stream, proposal classification scores are obtained and proposal clusters are generated consequently. Then based on these proposal clusters, supervisions are generated to compute losses for the next stream. During the back-propagation process of training, the network losses are optimized to train proposal features and classifiers. As shown in the figure, supervisions of the 1-st refined classifier depend on the output from the basic classifier, and supervisions of k-th refined classifier depend on outputs from $\{k-1\}$-th refined classifier. In this section, we will introduce our method of learning refined instance classifiers based on proposal clusters in detail.

3.1 Notations

Before presenting our method, we first introduce some of the mostly used notations as follows. We have R proposals with boxes $\mathcal{B} = \{b_r\}_{r=1}^{R}$ for an given image and proposal features $\mathbf{F}$, where b_r is the r-th proposal box. The number of refined instance classifiers is K (*i.e.*, we refine instance classifier K times), and thus there are $K+1$ streams. The number of object classes is C. $\mathbf{W}^0$ and $\mathbf{W}^k, k \in \{1, ..., K\}$ are the parameters of the basic instance classifier and the k-th refined instance classifier, respectively. $\boldsymbol{\varphi}^0(\mathbf{F}, \mathbf{W}^0) \in \mathbb{R}^{C \times R}$ and $\boldsymbol{\varphi}^k(\mathbf{F}, \mathbf{W}^k) \in \mathbb{R}^{(C+1) \times R}, k \in \{1, ..., K\}$ are the predicted score matrices of the basic instance classifier and the k-th refined instance classifier, respectively, where $C+1$ indicates the C object classes and 1 background class. We use $\boldsymbol{\varphi}^k$ later for simplification, dropping the dependence on $\mathbf{F}, \mathbf{W}^k$. φ^k_{cr} is the predicted score of the r-th proposal for class c from the k-th instance classifier. $\mathbf{y} = [y_1, ..., y_C]^T$ is the image label vector, where $y_c = 1$ or 0 indicates the image with or without object class c. $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$ is the supervision of the k-th instance classifier, where $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y}), k = 0$ is the image label vector $\mathbf{y}$. $\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right)$ is the loss function to train the k-th instance classifier.

We compute N^k proposal cluster centers $\mathcal{S}^k = \{S^k_n\}_{n=1}^{N^k}$ for the k-th refinement. The n-th cluster center $S^k_n = (b^k_n, y^k_n, s^k_n)$ consists of a proposal box $b^k_n \in \mathcal{B}$, an object label y^k_n ($y^k_n = c, c \in \{1, ..., C\}$ indicates the c-th object class), and a confidence score s^k_i indicating the confidence that b^k_n covers at least part of an object of class y^k_n. We have $N^k + 1$ proposal clusters $\mathcal{C}^k = \{\mathcal{C}^k_n\}_{n=1}^{N^k+1}$ according to $\mathcal{S}^k$ ($\mathcal{C}^k_{N^k+1}$ for background and others for objects). For object clusters, the n-th cluster $\mathcal{C}^k_n = (\mathcal{B}^k_n, y^k_n, s^k_n), n \neq N^k + 1$ consists of M^k_n proposal boxes $\mathcal{B}^k_n = \{b^k_{nm}\}_{m=1}^{M^k_n} \subseteq \mathcal{B}$, an object label y^k_n that is the same as the cluster center label, and a confidence score s^k_n that is the same as the cluster center score, where s^k_n indicates the confidence that $\mathcal{C}^k_n$ corresponds to an object of class y^k_n. Unlike object clusters, the background cluster $\mathcal{C}^k_n = (\mathcal{P}^k_n, y^k_n), n = N^k + 1$ consists of M^k_n proposals $\mathcal{P}^k_n = \{P^k_{nm}\}_{m=1}^{M^k_n}$ and a label $y^k_n = C + 1$ indicating the background. The m-th proposal $P^k_{nm} = (b^k_{nm}, s^k_{nm})$ consists of a proposal box $b^k_{nm} \in \mathcal{B}$ and a confidence score s^k_{nm} indicating the confidence that b^k_{nm} is the background.

3.2 Basic MIL network

It is necessary to generate proposal scores and clusters to supervise refined instance classifiers. More specifically, the first refined classifier requires basic instance classifiers to generate proposal scores and clusters. Therefore, we first introduce our basic MIL network as the basic instance classifier. Our overall network is independent of the specific MIL methods, and thus any method that can be trained end-to-end could be used. There are many possible choices [16], [17], [18]. Here we choose the method by Bilen and Vedaldi [17] which proposes a weighted sum pooling strategy to obtain the instance classifier, because of its effectiveness and implementation convenience. To make our paper self-contained, we briefly introduce [17] as follows.

Given an input image and its proposal boxes $\mathcal{B} = \{b_r\}_{r=1}^{R}$, a set of proposal features $\mathbf{F}$ are first generated by the network. Then as shown in the "Basic MIL network" block of Fig. 4, there are two branches which process the proposal features to produce two matrices $\mathbf{X}^{\text{cls}}(\mathbf{F}, \mathbf{W}^{\text{cls}}), \mathbf{X}^{\text{det}}(\mathbf{F}, \mathbf{W}^{\text{det}}) \in \mathbb{R}^{C \times R}$ (we use $\mathbf{X}^{\text{cls}}, \mathbf{X}^{\text{det}}$ later for simplification, dropping the dependence on $\mathbf{F}, \mathbf{W}^{\text{cls}}, \mathbf{W}^{\text{det}}$) of an input image by two fc layers, where $\mathbf{W}^{\text{cls}}$ and $\mathbf{W}^{\text{det}}$ denote the parameters of the fc layer for $\mathbf{X}^{\text{cls}}$ and the parameters of the fc layer for $\mathbf{X}^{\text{det}}$, respectively. Then the two matrices are passed through two softmax layer along different directions: $[\boldsymbol{\sigma}(\mathbf{X}^{\text{cls}})]_{cr} = e^{x^{\text{cls}}_{cr}} / \sum_{c'=1}^{C} e^{x^{\text{cls}}_{c'r}}$ and $[\boldsymbol{\sigma}(\mathbf{X}^{\text{det}})]_{cr} = e^{x^{\text{det}}_{cr}} / \sum_{r'=1}^{R} e^{x^{\text{det}}_{cr'}}$. Let us denote $(\mathbf{W}^{\text{cls}}, \mathbf{W}^{\text{det}})$ by $\mathbf{W}^0$. The proposal scores are generated by element-wise product $\boldsymbol{\varphi}^0(\mathbf{F}, \mathbf{W}^0) = \boldsymbol{\sigma}(\mathbf{X}^{\text{cls}}) \odot \boldsymbol{\sigma}(\mathbf{X}^{\text{det}})$. Finally, the image score of the c-th class $[\boldsymbol{\phi}(\mathbf{F}, \mathbf{W}^0)]_c$ is obtained by the sum over all proposals: $[\boldsymbol{\phi}(\mathbf{F}, \mathbf{W}^0)]_c = \sum_{r=1}^{R} [\boldsymbol{\varphi}^0(\mathbf{F}, \mathbf{W}^0)]_{cr}$.

A simple interpretation of the two branches framework is as follows. $[\boldsymbol{\sigma}(\mathbf{X}^{\text{cls}})]_{cr}$ is the probability of the r-th proposal belonging to class c. $[\boldsymbol{\sigma}(\mathbf{X}^{\text{det}})]_{cr}$ is the normalized weight that indicates the contribution of the r-th proposal to image being classified to class i. So $[\boldsymbol{\phi}(\mathbf{F}, \mathbf{W}^0)]_c$ is obtained by weighted sum pooling and falls in the range of $(0, 1)$. Given the image label vector $\mathbf{y} = [y_1, ..., y_C]^T$. We train the

Algorithm 1 The overall training procedure (one iteration)

Input: An image, its proposal boxes $\mathcal{B}$, and its image label vector $\mathbf{y} = [y_1, ..., y_C]^T$; refinement times K.
Output: An updated network.

1: Feed the image and $\mathcal{B}$ into the network to produce proposal score matrices $\boldsymbol{\varphi}^k(\mathbf{F}, \mathbf{W}^k), k \in \{0, 1, ..., K\}$ (simplified as $\boldsymbol{\varphi}^k$ later).
2: Compute loss $\mathrm{L}^0\left(\mathbf{F}, \mathbf{W}^0, \mathbf{y}\right)$ by Eq. (1), see Section 3.2.
3: **for** $k = 1$ **to** K **do**
4: Generate supervisions $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$, see Section 3.4.
5: Compute loss $\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right)$ by Eq. (6)/(7)/(8), see Section 3.4.
6: Optimize $\sum_{k=0}^{K} \mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right)$, *i.e.*, Eq. (2), w.r.t. $\mathbf{F}, \mathbf{W}^k$ (not w.r.t. $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$).

basic instance classifier by optimizing the multi-class cross entropy loss Eq. (1) w.r.t. $\mathbf{F}, \mathbf{W}^0$.

$$\mathrm{L}^0\left(\mathbf{F}, \mathbf{W}^0, \mathbf{y}\right) = -\sum_{c=1}^{C}\{(1-y_c)\log(1-[\phi(\mathbf{F}, \mathbf{W}^0)]_c) + y_c \log[\phi(\mathbf{F}, \mathbf{W}^0)]_c\}. \quad (1)$$

3.3 The overall training strategy

To refine instance classifiers iteratively, we add multiple output streams in our network where each stream corresponds to a refined classifier, as shown in Fig. 4. We integrate the basic MIL network and the classifier refinement into an end-to-end network to learn the refined classifier online. Unlike the basic instance classifier, for an input image the output score matrix $\boldsymbol{\varphi}^k(\mathbf{F}, \mathbf{W}^k)$ of the k-th refined classifier is a $\{C+1\} \times R$ matrix and is obtained by passing the proposal features through a single fc layer (with parameters $\mathbf{W}^k$) as well as a softmax over-classes layer, *i.e.*, $\boldsymbol{\varphi}^k(\mathbf{F}, \mathbf{W}^k) \in \mathbb{R}^{(C+1)\times R}, k \in \{1, 2, ..., K\}$, as in the "Instance classifier refinement" blocks of Fig. 4. Notice that we use the same proposal features $\mathbf{F}$ for all classifiers. We use $\boldsymbol{\varphi}^k$ later for simplification, dropping the dependence on $\mathbf{F}, \mathbf{W}^k$.

As we stated before, supervisions to train the k-th instance classifier are generated based on proposal scores $\boldsymbol{\varphi}^{k-1}$ and image label $\mathbf{y}$. Thus we denote the supervisions by $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$. Then we train our overall network by optimizing the loss Eq. (2) w.r.t. $\mathbf{F}, \mathbf{W}^k$. We do not optimize the loss w.r.t. $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$, which means that the supervisions $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$ are only computed in the forward process and we do not compute their gradients to train our network.

$$\sum_{k=0}^{K} \mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right). \quad (2)$$

The loss $\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right), k > 0$ for the k-th refined instance classifier is defined in later Eq. (6)/(7)/(8) which are loss functions with supervisions provided by $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$. We will give details about how to get supervisions $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$ and loss functions $\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right)$ in Section 3.4.

During the forward process of each Stochastic Gradient Descent (SGD) training iteration, we obtain a set of proposal scores of an input image. Accordingly, we generate the supervisions $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$ for the iteration to compute the loss Eq. (2). During the back-propagation process of each SGD training iteration, we optimize the loss Eq. (2) w.r.t. proposal features $\mathbf{F}$ and classifiers $\mathbf{W}^k$. We summarize this procedure in Algorithm 1. Note that we do not use an alternating training strategy, *i.e.*, fixing supervisions and training a complete model, fixing the model and updating supervisions. The reasons are that: 1) it is very time-consuming because it requires training models multiple times; 2) training different models in different refinement steps separately may harm the performance because it hinders the process to benefit from the shared proposal features (*i.e.*, $\mathbf{F}$).

3.4 Proposal cluster learning

Here we will introduce our methods to learn refined instance classifiers based on proposal clusters (*i.e.*, proposal cluster learning).

Recall from Section 3.1 that we have a set of proposals with boxes $\mathcal{B} = \{b_r\}_{r=1}^{R}$. For the k-th refinement, our goal is to generate supervisions $\mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})$ for the loss functions $\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k(\boldsymbol{\varphi}^{k-1}, \mathbf{y})\right)$ using the proposal scores $\boldsymbol{\varphi}^{k-1}$ and image label $\mathbf{y}$ in each training iteration. We use $\mathcal{H}^k, \mathrm{L}^k$ later for simplification, dropping the dependence on $\boldsymbol{\varphi}^{k-1}, \mathbf{y}, \mathbf{F}, \mathbf{W}^k$.

We do this in three steps. 1) We find proposal cluster centers which are proposals corresponding to different objects. 2) We group the remaining proposals into different clusters, where each cluster is associated with a cluster center or corresponds to the background. 3) We generate the supervisions $\mathcal{H}^k$ for the loss functions L^k, enabling us to train the refined instance classifiers.

For the first step, we compute proposal cluster centers $\mathcal{S}^k = \{S_n^k\}_{n=1}^{N^k}$ based on $\boldsymbol{\varphi}^{k-1}$ and $\mathbf{y}$. The n-th cluster center $S_n^k = (b_n^k, y_n^k, s_n^k)$ is defined in Section 3.1. We propose two algorithms to find $\mathcal{S}^k$ in Section 3.4.1 (1) and (2) (also Algorithm 2 and Algorithm 3), where the first one was proposed in the conference version paper [21] and the second one is proposed in this paper.

For the second step, according to the proposal cluster centers $\mathcal{S}^k$, proposal clusters $\mathcal{C}^k = \{\mathcal{C}_n^k\}_{n=1}^{N^k+1}$ are generated ($\mathcal{C}_{N^k+1}^k$ for background and others for objects). The n-th object cluster $\mathcal{C}_n^k = (\mathcal{B}_n^k, y_n^k, s_n^k), n \neq N^k+1$ and the background cluster $\mathcal{C}_n^k = (\mathcal{P}_n^k, y_n^k), n = N^k+1$ are defined in Section 3.1. We use the different notation for the background cluster because background proposals are scattered in each image, and thus it is hard to determine a cluster center and accordingly a cluster score. The method to generate $\mathcal{C}^k$ was proposed in the conference version paper and is described in Section 3.4.2 (also Algorithm 4).

For the third step, supervisions $\mathcal{H}^k$ to train the k-th refined instance classifier are generated based on the proposal clusters. We use two strategies where $\mathcal{H}^k$ are either proposal-level labels indicating whether a proposal belongs to an object class, or cluster-level labels that treats each proposal cluster as a bag. Subsequently these are used to compute the loss functions L^k. We propose two approaches to do this as described in Section 3.4.3 (1) and (2), where the first one was proposed in the conference version paper and the second one is proposed in this paper.

Algorithm 2 Finding proposal cluster centers using the highest scoring proposal

Input: Proposal boxes $\mathcal{B} = \{b_1, ..., b_R\}$; image label vector $\mathbf{y} = [y_1, ..., y_C]^T$; proposal score matrix $\boldsymbol{\varphi}^{k-1}$.
Output: Proposal cluster centers $\mathcal{S}^k$.
1: Initialize $\mathcal{S}^k = \varnothing$.
2: **for** $c = 1$ **to** C **do**
3: **if** $y_c = 1$ **then**
4: Choose the r_c^k-th proposal by Eq. (3).
5: $\mathcal{S}^k$.append $\left((b_{r_c^k}, c, \varphi_{cr_c^k}^{k-1})\right)$.

Algorithm 3 Finding proposal cluster centers using graphs of top ranking proposals

Input: Proposal boxes $\mathcal{B} = \{b_1, ..., b_R\}$; image label vector $\mathbf{y} = [y_1, ..., y_C]^T$; proposal score matrix $\boldsymbol{\varphi}^{k-1}$.
Output: Proposal cluster centers $\mathcal{S}^k$.
1: Initialize $\mathcal{S}^k = \varnothing$.
2: **for** $c = 1$ **to** C **do**
3: **if** $y_c = 1$ **then**
4: Select top ranking proposals with indexes $\mathcal{R}_c^k$.
5: Build a graph G_c^k using the top ranking proposals.
6: **repeat**
7: Set $r_c^k = \text{argmax}_{r'} \sum_{r \in V_c^k} e_{crr'}^k$.
8: Set $s = \max_r \varphi_{cr}^{k-1}, r$ s.t. $e_{crr_c^k}^k = 1$ or $r = r_c^k$.
9: $\mathcal{S}^k$.append $\left((b_{r_c^k}, c, s)\right)$.
10: Remove the r-th proposal box from V_c^k, $\forall r$ s.t. $e_{crr_c^k}^k = 1$ or $r = r_c^k$.
11: **until** V_c^k is empty.

3.4.1 Finding proposal cluster centers

In the following we introduce two algorithms to find proposal cluster centers.

(1) Finding proposal cluster centers using the highest scoring proposal. A solution for finding proposal cluster centers is to choose the highest scoring proposal, as in our conference version paper [21]. As in Algorithm 2, suppose an image has object class label c (*i.e.*, $y_c = 1$). For the k-th refinement, we first select the r_c^k-th proposal which has the highest score by Eq. (3), where φ_{cr}^{k-1} is the predicted score of the r-th proposal, as defined in Section 3.1.

$$r_c^k = \underset{r}{\text{argmax}}\, \varphi_{cr}^{k-1}. \tag{3}$$

Then this proposal is chosen as the cluster center, *i.e.*, $S_n^k = (b_n^k, y_n^k, s_n^k) = (b_{r_c^k}, c, \varphi_{cr_c^k}^{k-1})$, where $b_{r_c^k}$ is the box of the r_c^k-th proposal. φ_{cr}^{k-1} is chosen as the confidence score that the r-th proposal covers at least part of an object of class c, because φ_{cr}^{k-1} is the predicted score of the r-th proposal been categorized to class c. Therefore, the highest scoring proposal can probably cover at least part of the object and thus be chosen as the cluster center.

There is a potential problem that one proposal may be chosen as the cluster centers for multiple object classes. To avoid this problem, if one proposal corresponds to the cluster centers for multiple object classes, this proposal would be chosen as the cluster center only by the class with the highest predicted score and we re-choose cluster centers for other classes.

(2) Finding proposal cluster centers using graphs of top ranking proposals. As stated in Section 1, although we can find good proposal cluster centers using the highest scoring proposal, this ignores that in natural images there are often more than one object for each category. Therefore, we propose a new method to find cluster centers using graphs of top ranking proposals.

More specifically, suppose an image has object class label c. We first select the top ranking proposals with indexes $\mathcal{R}_c^k = \{r_{c1}^k, ..., r_{cN_c^k}^k\}$ for the k-th refinement. Then we build an undirected unweighted graph $G_c^k = (V_c^k, E_c^k)$ of these proposals based on spatial similarity, where vertexes V_c^k correspond to these top ranking proposals, and edges $E_c^k = \{e_{crr'}^k\} = \{e(v_{cr}^k, v_{cr'}^k)\}, r, r' \in \mathcal{R}_c^k$ correspond to the connections between the vertexes. $e_{crr'}^k$ is determined according to the spatial similarity between two vertexes (*i.e.*, proposals) as in Eq. (4), where $I_{rr'}$ is the IoU between the r-th and r'-th proposals and I_t is a threshold (*e.g.*, 0.4).

$$e_{rr'} = \begin{cases} 1 & \text{if } I_{rr'} > I_t, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Therefore, two vertexes are connected if they are spatially adjacent. After that, we greedily generate some cluster centers for class c using this graph. That is, we iteratively select vertexes which have most connections to be the cluster centers, as in Algorithm 3. The number of cluster centers (*i.e.*, N^k) changes for each image in each training iteration because the top ranking proposals $\mathcal{R}_c^k$ change. See Section 4.2.9 for some typical values of N^k. We use the same method as in Section 3.4.1 (1) to avoid one proposal been chosen as the cluster centers for multiple object classes.

The reasons for this strategy are as follows. First, according to our observation, the top ranking proposals can always cover at least parts of objects, thus generating centers from these proposals encourages the selected centers to meet our requirements. Second, because these proposals cover objects well, better proposals (covering more parts of objects) should have more spatially overlapped proposals (*i.e.*, have more connections). Third, these centers are spatially far apart, and thus different centers can correspond to different objects. This method also has the attractive characteristic that it can generate adaptive number of proposals for each object class, which is desirable because in natural images there are arbitrary number of objects per-class. We set the score of the n-th proposal cluster center s_n^k by

$$s_n^k = \max_r \varphi_{cr}^{k-1}, r \text{ s.t. } e_{crr_c^k}^k = 1 \text{ or } r = r_c^k$$

(see the 8-th line in Algorithm 3) because if the adjacent proposals of a center proposal have high confidence to cover at least part of an object (*i.e.*, have high classification scores) the center proposal should also have such high confidence.

There is an important issue for the graph-based method: how to select the top ranking proposals? A simple method is to select proposals whose scores exceed a threshold. But in our case, proposal scores change in each training iteration, and thus it is hard to determine a threshold. Instead, for each

Algorithm 4 Generating proposal clusters

Input: Proposal boxes $\mathcal{B} = \{b_1, ..., b_R\}$; proposal cluster centers $\mathcal{S}^k = \{S_1^k, ..., S_{N^k}^k\}$.
Output: Proposal clusters $\mathcal{C}^k$.
1: Initialize $\mathcal{B}_n^k = \varnothing, \forall n \neq N^k + 1$.
2: Set y_n^k, s_n^k of $\mathcal{C}_n^k$ to y_n^k, s_n^k of $S_{n'}^k, \forall n \neq N^k + 1$.
3: Initialize $\mathcal{P}_{N^k+1}^k = \varnothing$ and set $y_{N^k+1}^k = C + 1$.
4: **for** $r = 1$ **to** R **do**
5: Compute IoUs $\{I_{r1}^k, ..., I_{rN^k}^k\}$.
6: Choose the most spatially adjacent center $S_{n_r^k}^k$.
7: **if** $I_{rn_r^k}^k > I_t'$ **then**
8: $\mathcal{B}_{n_r^k}^k$.append (b_r).
9: **else**
10: $\mathcal{P}_{N^k+1}^k$.append $\left((b_r, s_{n_r^k}^k)\right)$.

positive object class, we use the k-means [48] algorithm to divide proposal scores of an image into some clusters, and choose proposals in the cluster which has the highest score center to form the top ranking proposals. This method ensures that we can select the top ranking proposals although proposal scores change during training. Other choices are possible, but this method works well in experiments.

3.4.2 Generating proposal clusters

After the cluster centers are found, we generate the proposal clusters as in our conference version paper [21]. Except for the cluster for background, good proposal clusters require that proposals in the same cluster are associated with the same object, and thus proposals in the same cluster should be spatially adjacent. Specially, given the r-th proposal, we compute a set of IoUs $\{I_{r1}^k, ..., I_{rN^k}^k\}$, where I_{rn}^k is the IoU between the r-th proposal and the box b_n^k of the n-th cluster center. Then we assign the r-th proposal to the n_r^k-th object cluster if $I_{rn_r^k}^k$ is larger than a threshold I_t' (*e.g.*, 0.5) and to the background cluster otherwise, where n_r^k is the index of the most spatially adjacent cluster center as Eq. (5).

$$n_r^k = \underset{n}{\operatorname{argmax}}\, I_{rn}^k. \tag{5}$$

The overall procedures to generate proposal clusters are summarized in Algorithm 4. We set the proposal scores for the background cluster to the scores of their most spatially adjacent centers as the 10-the line in Algorithm 4, because if the cluster center S_n^k has confidence s_n^k that it covers an object, the proposal far away from S_n^k should have confidence s_n^k to be background.

3.4.3 Learning refined instance classifiers

To get supervisions $\mathcal{H}^k$ and loss functions L^k to learn the k-th refined instance classifier, we design two approaches as follows.

(1) Assigning proposals object labels. The most straightforward way to refine classifiers is to directly assign object labels to all proposals in object clusters because these proposals potentially correspond to whole objects, as in our conference version paper [21]. As the cluster centers covers at least parts of objects, their adjacent proposals (*i.e.*, proposals in the cluster) can contain larger parts of objects. Accordingly, we can assign the cluster label y_n^k to all proposals in the n-th cluster.

More specifically, the supervisions $\mathcal{H}^k$ are proposal-level labels, *i.e.*, $\mathcal{H}^k = \{\mathbf{y}_r^k\}_{r=1}^R$. $\mathbf{y}_r^k = [y_{1r}^k, ..., y_{(C+1)r}^k]^T \in \mathbb{R}^{(C+1)\times 1}$ is the label vector of the r-th proposal for the k-th refinement, where $y_{y_{n_r^k}r}^k = 1$ and $y_{cr}^k = 0, c \neq y_n^k$ if the r-th proposal belongs to the n-th clusters. Consequently, we use the standard softmax loss function to train the refined classifiers as in Eq. (6), where φ_{cr}^k is the predicted score of the r-th proposal as defined in Section 3.1.

$$\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k\right) = -\frac{1}{R}\sum_{r=1}^{R}\sum_{c=1}^{C+1} y_{cr}^k \log \varphi_{cr}^k. \tag{6}$$

Through iterative instance classifier refinement (*i.e.*, multiple times of refinement as k increase), the detector detects larger parts of objects gradually by forcing the network to "see" larger parts of objects.

Actually, the so learnt supervisions $\mathcal{H}^k$ are very noisy, especially in the beginning of training. This results in unstable solutions. To solve this problem, we change the loss in Eq. (6) to a weighted version, as in Eq. (7).

$$\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k\right) = -\frac{1}{R}\sum_{r=1}^{R}\sum_{c=1}^{C+1} \lambda_r^k y_{cr}^k \log \varphi_{cr}^k. \tag{7}$$

λ_r^k is the loss weight that is the same as the cluster confidence score s_n^k for object clusters or proposal confidence score s_{nm}^k for the background cluster if the r-th proposal belongs to the n-th cluster. From Algorithm 4, we can observe that λ_r^k is the same as the cluster center confidence score s_n^k. The reasons for this strategy are as follows. In the beginning of training, although we cannot obtain good proposal clusters, each s_n^k is small, hence each λ_r^k is small and the loss is also small. As a consequence, the performance of the network will not decrease a lot. During the training, the top ranking proposals will cover objects well, and thus we can generate good proposal clusters. Then we can train satisfactory instance classifiers.

(2) Treating clusters as bags. As we stressed before, although directly assigning proposals object labels can boost the results, it may confuse the network because we simultaneously assign the same label to different parts of objects. Focusing on this, we further propose to treat each proposal cluster as a small new bag and use the cluster label as the bag label. Thus the supervisions $\mathcal{H}^k$ for the k-th refinement are bag-level (cluster-level) labels, *i.e.*, $\mathcal{H}^k = \{y_n^k\}_{n=1}^{N^k+1}$. y_n^k is the label of the n-th bag, *i.e.*, the label of the n-th proposal cluster, as defined in Section 3.1.

Specially, for object clusters, we choose average MIL pooling, because these proposals should cover at least parts of objects and thus should have relatively high prediction scores. For the background cluster, we assign the background label to all proposals in the cluster according to the MIL constraints (all instances in negative bags are negative).

Then the loss function for refinement will be Eq. (8).

$$\mathrm{L}^k\left(\mathbf{F}, \mathbf{W}^k, \mathcal{H}^k\right) = -\frac{1}{R}\Big(\sum_{n=1}^{N^k} s_n^k M_n^k \log \frac{\sum\limits_{r \text{ s.t. } b_r \in \mathcal{B}_n^k} \varphi_{y_n^k r}^k}{M_n^k} + \sum_{r \in \mathcal{C}_{N^k+1}^k} \lambda_r^k \log \varphi_{(C+1)r}^k\Big). \tag{8}$$

s_n^k, M_n^k, and φ_{cr}^k are the cluster confidence score of the n-th object cluster, the number of proposals in the n-th cluster, and the predicted score of the r-th proposal, respectively, as defined in Section 3.1. $b_r \in \mathcal{B}_n^k$ and $r \in \mathcal{C}_{N^k+1}^k$ indicate that the r-th proposal belongs to the n-th object cluster and the background cluster respectively.

Compared with the directly assigning label approach, this method tolerates some proposals to have low scores, which can reduce the ambiguities to some extent.

3.5 Testing

During testing, the proposal scores of refined instance classifiers are used as the final detection scores, as the blue arrows in Fig. 4. Here the mean output of all refined classifiers is chosen. The Non-Maxima Suppression (NMS) is used to filter out redundant detections.

4 Experiments

In this section, we first introduce our experimental setup including datasets, evaluation metrics, and implementation details. Then we conduct elaborate experiments to discuss the influence of different settings. Next, we compare our results with others to show the effectiveness of our method. After that, we show some qualitative results for further analyses. Finally, we give some runtime analyses of our method. Codes for reproducing our results are available at https://github.com/ppengtang/oicr/tree/pcl.

4.1 Experimental setup

4.1.1 Datasets and evaluation metrics

We evaluate our method on four challenging datasets: the PASCAL VOC 2007 and 2012 datasets [3], the ImageNet detection dataset [4], and the MS-COCO dataset [5]. Only image-level annotations are used to train our models.

The PASCAL VOC 2007 and 2012 datasets have $9,962$ and $22,531$ images respectively for 20 object classes. These two datasets are divided into train, val, and test sets. Here we choose the trainval set ($5,011$ images for 2007 and $11,540$ images for 2012) to train our network. For testing, there are two metrics for evaluation: mAP and CorLoc. Following the standard PASCAL VOC protocol [3], Average Precision (AP) and the mean of AP (mAP) is the evaluation metric to test our model on the testing set. Correct Localization (CorLoc) is to test our model on the training set measuring the localization accuracy [34]. All these two metrics are based on the PASCAL criterion, *i.e.*, IoU>0.5 between groundtruth boundingboxes and predicted boxes.

The ImageNet detection dataset has hundreds of thousands of images with 200 object classes. It is also divided into train, val, and test sets. Following [6], we split the val set into val1 and val2, and randomly choose at most 1K images in the train set for each object class (we call it $\text{train}_{1\text{K}}$). We train our model on the mixture of $\text{train}_{1\text{K}}$ and val1 sets, and test it on the val2 set, which will lead to $160,651$ images for training and $9,916$ images for testing. We also use the mAP for evaluation on the ImageNet.

The MS-COCO dataset has 80 object classes and is divided into train, val, and test sets. Since the groundtruths on the test set are not released, we train our model on the MS-COCO 2014 train set (about 80K images) and test it on the val set (about 40K images). For evaluation, we use two metrics mAP@0.5 and mAP@[.5, .95] which are the standard PASCAL criterion (*i.e.*, IoU>0.5) and the standard MS-COCO criterion (*i.e.*, computing the average of mAP for IoU$\in$[0.5 : 0.05 : 0.95]) respectively.

4.1.2 Implementation details

Our method is built on two pre-trained ImageNet [4] networks VGG_M [49] and VGG16 [50], each of which has some conv layers with max-pooling layers and three fc layers. We replace the last max-pooling layer by the SPP layer, and the last fc layer as well as the softmax loss layer by the layers described in Section 3. To increase the feature map size from the last conv layer, we replace the penultimate max-pooling layer and its subsequent conv layers by the dilated conv layers [51], [52]. The newly added layers are initialized using Gaussian distributions with 0-mean and standard deviations 0.01. Biases are initialized to 0.

During training, the mini-batch size for SGD is set to be 2, 32, and 4 for PASCAL VOC, ImageNet, and MS-COCO, respectively. The learning rate is set to 0.001 for the first 40K, 60K, 15K, and 85K iterations for the PASCAL VOC 2007, PASCAL VOC 2012, ImageNet, and MS-COCO datasets, respectively. Then we decrease the learning rate to 0.0001 in the following 10K, 20K, 5K, and 20K iterations for the PASCAL VOC 2007, PASCAL VOC 2012, ImageNet, and MS-COCO datasets, respectively. The momentum and weight decay are set to be 0.9 and 0.0005 respectively.

Selective Search [20], EdgeBox [47], and MCG [53] are adopted to generate about $2,000$ proposals per-image for the PASCAL VOC, ImageNet, and MS-COCO datasets, respectively. For data augmentation, we use five image scales $\{480, 576, 688, 864, 1200\}$ (resize the shortest side to one of these scales) with horizontal flips for both training and testing. If not specified, the instance classifiers are refined three times, *i.e.*, $K = 3$ in Section 3.3, so there are four output streams; the IoU threshold I_t in Section 3.4.1 (2) (also Eq. (4)) is set to 0.4; the number of k-means clusters in the last paragraph of Section 3.4.1 (2) is set to 3; I_t' in Section 3.4.2 (also the 5-th line of Algorithm 4) is set to 0.5.

Similar to other works [19], [43], [54], we train a supervised object detector through choosing the top-scoring proposals given by our method as pseudo groundtruths to further improve our results. Here we train a Fast R-CNN (FRCNN) [7] using the VGG16 model and the same five image scales (horizontal flips only in training). The same proposals are chosen to train and test the FRCNN. NMS (with 30% IoU threshold) is applied to compute AP.

Our experiments are implemented based on the Caffe [55] deep learning framework, using Python and C++. The k-means algorithm to produce top ranking proposals is

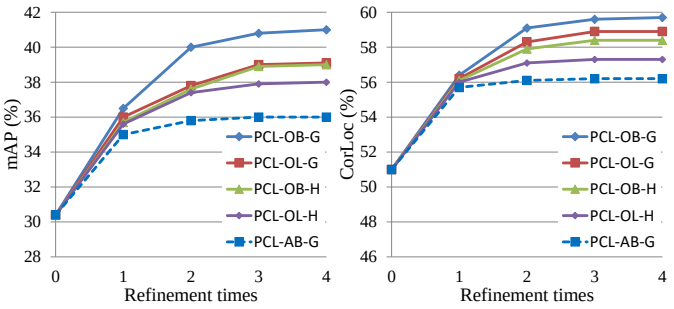


Fig. 5. Results on VOC 2007 for different refinement times and different training strategies, where "PCL-xx-H" and "PCL-xx-G" indicate the highest scoring proposal based method and the graph-based method to generate proposal clusters respectively, "PCL-OL-x" and "PCL-OB-x" indicate the directly assigning label method and the treating clusters as bags method to train the network online respectively, and "PCL-AB-x" indicates using the alternating training strategy.

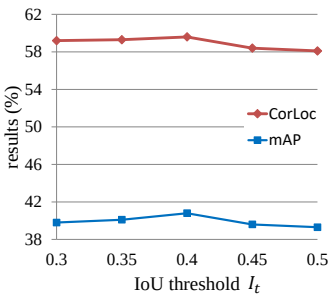


Fig. 6. Results on VOC 2007 for different IoU threshold I_t.

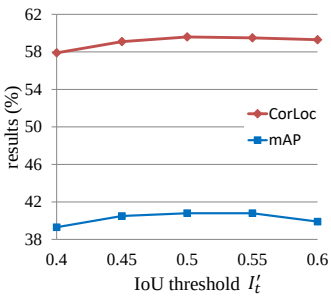


Fig. 7. Results on VOC 2007 for different IoU threshold I'_t.

implemented by scikit-learn [56]. All of our experiments are running on an NVIDIA GTX TitanX Pascal GPU and Intel(R) i7-6850K CPU (3.60GHz).

4.2 Discussions

We first conduct some experiments to discuss the influence of different components of our method (including instance classifier refinement, different proposal generation methods, different refinement strategies, and weighted loss) and different parameter settings (including the IoU threshold I_t defined in Section 3.4.1 (2), the number of k-means clusters described in Section 3.4.1 (2), the IoU threshold I'_t defined in Section 3.4.2, and multi-scale training and testing.) We also discuss the number of proposal cluster centers. Without loss of generality, we only perform experiments on the VOC 2007 dataset and use the VGG_M model.

4.2.1 *The influence of instance classifier refinement*

As the five curves in Fig. 5 show, we observe that compared with the basic MIL network, for both refinement methods, even refining instance classifier a single time boosts the performance a lot. This confirms the necessity of refinement. If we refine the classifier multiple times, the results are improved further. But when refinement is implemented too many times, the performance gets saturated (there are no obvious improvements from 3 times to 4 times). This is because the network tends to converge so that the supervision of the 4-th time is similar to the 3-rd time. In the rest of this paper we only refine classifiers 3 times. Notice that in Fig. 5, the "0 time" is similar to the WSDDN [17] using Selective Search as proposals.

4.2.2 *The influence of different proposal cluster generation methods*

We discuss the influence of different proposal cluster generation methods. As shown in the Fig. 5 (green and purple solid curves for the highest scoring proposal based method, blue and red solid curves for the graph-based method), for all refinement times, the graph-based method obtains better performance, because it can generate better cluster centers. Thus we choose the graph-based method in the rest of our paper.

4.2.3 *The influence of different refinement strategies*

We then show the influence of different refinement strategies. The directly assigning label method is replaced by treating clusters as bags (blue and green solid curves). From Fig. 5, it is obvious that the results by treating clusters as bags are better. In addition, compared with the alternating training strategy (blue dashed curve), our online training boosts the performance consistently and significantly, which confirms the necessity of sharing proposal features. Online training also reduces the training time a lot, because it only requires training a single model instead of training $K+1$ models for K times refinement in the alternating strategy. In the rest of our paper, we only report results by the "PCL-OB-G" method in Fig. 5 because it achieves the best performance.

4.2.4 *The influence of weighted loss*

We also study the influence of our weighted loss in Eq. (8). Note that Eq. (8) can be easily changed to the unweighted version by simply setting λ_r^k and s_n^k to be 1. Here we train a network using the unweighted loss. The results of the unweighted loss are mAP 33.6% and CorLoc 51.2%. We see that if we use the unweighted loss, the improvement from refinement is very scant and the performance is even worse than the alternating strategy. Using the weighted loss achieves much better performance (mAP 40.8% and CorLoc 59.6%), which confirms our theory in Section 3.4.3.

4.2.5 *The influence of the IoU threshold I_t*

Here we discuss the influence of the IoU threshold I_t defined in Section 3.4.1 (2) and Eq. (4). From Fig. 6, we see that setting I_t to 0.4 obtains the best performance. Therefore, we set I_t to 0.4 for the other experiments.

4.2.6 *The influence of the number of k-means clusters*

In previous experiments we set the number of k-means clusters described in the last paragraph of Section 3.4.1 (2) to be 3. Here we set it to other numbers to explore its influence. The results from other numbers of k-means clusters are mAP 40.2% and CorLoc 59.3% for 2 clusters, and mAP 40.7% and CorLoc 59.6% for 4 clusters, which are a little worse than the results from 3 cluster. Therefore, we set the number of k-means clusters to 3 for the other experiments.

TABLE 1
Results (AP in %) for different methods on the VOC 2007 test set. The upper part shows the results using a single model. The lower part shows the results of combing multiple models. See Section 4.3 for the definitions of the PCL-based methods.

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
WSDDN-VGG_F [17]	42.9	56.0	32.0	17.6	10.2	61.8	50.2	29.0	3.8	36.2	18.5	31.1	45.8	54.5	10.2	15.4	36.3	45.2	50.1	43.8	34.5
WSDDN-VGG_M [17]	43.6	50.4	32.2	**26.0**	9.8	58.5	50.4	30.9	7.9	36.1	18.2	31.7	41.4	52.6	8.8	14.0	37.8	46.9	53.4	47.9	34.9
WSDDN-VGG16 [17]	39.4	50.1	31.5	16.3	12.6	**64.5**	42.8	42.6	10.1	35.7	24.9	38.2	34.4	55.6	9.4	14.7	30.2	40.7	54.7	46.9	34.8
WSDDN+context [18]	**57.1**	52.0	31.5	7.6	11.5	55.0	53.1	34.1	1.7	33.1	**49.2**	**42.0**	47.3	56.6	15.3	12.8	24.8	48.9	44.4	47.8	36.3
PCL-OB-G-VGG_M	54.0	60.8	33.9	18.4	**15.8**	57.7	59.8	**52.0**	2.7	48.3	46.4	38.5	**47.9**	63.9	7.0	21.7	38.1	42.1	54.3	53.0	40.8
PCL-OB-G-VGG16	54.4	**69.0**	**39.3**	19.2	15.7	62.9	**64.4**	30.0	**25.1**	**52.5**	44.4	19.6	39.3	**67.7**	**17.8**	**22.9**	**46.6**	**57.5**	**58.6**	**63.0**	**43.5**
WSDDN-Ens. [17]	46.4	58.3	35.5	25.9	14.0	66.7	53.0	39.2	8.9	41.8	26.6	38.6	44.7	59.0	10.8	17.3	40.7	49.6	56.9	50.8	39.3
OM+MIL+FRCNN [54]	54.5	47.4	41.3	20.8	17.7	51.9	63.5	46.1	**21.8**	57.1	22.1	34.4	50.5	61.8	16.2	**29.9**	40.7	15.9	55.3	40.2	39.5
WCCN [19]	49.5	60.6	38.6	**29.2**	16.2	70.8	56.9	42.5	10.9	44.1	29.9	**42.2**	47.9	64.1	13.8	23.5	45.9	54.1	60.8	54.5	42.8
Jie *et al.* [43]	54.2	52.0	35.2	25.9	15.0	59.6	67.9	**58.7**	10.1	**67.4**	27.3	37.8	54.8	67.3	5.1	19.7	**52.6**	43.5	56.9	**62.5**	43.7
PCL-OB-G-Ens.	57.1	67.1	40.9	16.9	18.8	65.1	63.7	45.3	17.0	56.7	48.9	33.2	54.4	**68.3**	**16.8**	25.7	45.8	52.2	59.1	62.0	45.8
PCL-OB-G-Ens.+FRCNN	**63.2**	**69.9**	**47.9**	22.6	**27.3**	**71.0**	**69.1**	49.6	12.0	60.1	**51.5**	37.3	**63.3**	63.9	15.8	23.6	48.8	**55.3**	**61.2**	62.1	**48.8**

TABLE 2
Results (CorLoc in %) for different methods on the VOC 2007 trainval set. The upper part shows the results using a single model. The lower part shows the results of combing multiple models. See Section 4.3 for the definitions of the PCL-based methods.

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
WSDDN-VGG_F [17]	68.5	67.5	56.7	34.3	32.8	69.9	75.0	45.7	17.1	68.1	30.5	40.6	67.2	82.9	28.8	43.7	71.9	62.0	62.8	58.2	54.2
WSDDN-VGG_M [17]	65.1	63.4	59.7	45.9	38.5	69.4	77.0	50.7	30.1	68.8	34.0	37.3	61.0	82.9	25.1	42.9	**79.2**	59.4	68.2	64.1	56.1
WSDDN-VGG16 [17]	65.1	58.8	58.5	33.1	**39.8**	68.3	60.2	59.6	34.8	64.5	30.5	43.0	56.8	82.4	25.5	41.6	61.5	55.9	65.9	63.7	53.5
WSDDN+context [18]	**83.3**	68.6	54.7	23.4	18.3	73.6	74.1	54.1	8.6	65.1	47.1	**59.5**	67.0	83.5	**35.3**	39.9	67.0	49.7	63.5	65.2	55.1
PCL-OB-G-VGG_M	78.7	75.7	55.9	33.5	35.9	72.6	81.2	**69.5**	10.1	74.7	**52.5**	55.1	**73.1**	87.6	15.9	46.2	70.1	60.5	71.9	71.3	59.6
PCL-OB-G-VGG16	79.6	**85.5**	**62.2**	**47.9**	37.0	**83.8**	**83.4**	43.0	**38.3**	**80.1**	50.6	30.9	57.8	**90.8**	27.0	**58.2**	75.3	**68.5**	**75.7**	**78.9**	**62.7**
OM+MIL+FRCNN [54]	78.2	67.1	61.8	38.1	36.1	61.8	78.8	55.2	28.5	68.8	18.5	49.2	64.1	73.5	21.4	47.4	64.6	22.3	60.9	52.3	52.4
WSDDN-Ens. [17]	68.9	68.7	65.2	42.5	40.6	72.6	75.2	53.7	29.7	68.1	33.5	45.6	65.9	86.1	27.5	44.9	**76.0**	62.4	66.3	66.8	58.0
WCCN [19]	**83.9**	72.8	64.5	**44.1**	40.1	65.7	82.5	58.9	**33.7**	72.5	25.6	53.7	67.4	77.4	26.8	49.1	68.1	27.9	64.5	55.7	56.7
Jie *et al.* [43]	72.7	55.3	53.0	27.8	35.2	68.6	81.9	**60.7**	11.6	71.6	29.7	**54.3**	64.3	88.2	22.2	53.7	72.2	52.6	68.9	75.5	56.1
PCL-OB-G-Ens.	81.7	82.4	63.4	41.0	42.4	79.7	84.2	54.9	23.4	78.8	54.4	46.0	75.9	89.6	22.8	51.3	72.2	**66.1**	74.9	76.0	63.0
PCL-OB-G-Ens.+FRCNN	83.8	**85.1**	**65.5**	43.1	**50.8**	**83.2**	**85.3**	59.3	28.5	**82.2**	**57.4**	50.7	**85.0**	**92.0**	**27.9**	**54.2**	72.2	65.9	**77.6**	**82.1**	**66.6**

TABLE 3
Results (AP in %) for different methods on the VOC 2012 test set. See Section 4.3 for the definitions of the PCL-based methods.

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
WSDDN+context [18]	64.0	54.9	36.4	8.1	12.6	53.1	40.5	28.4	6.6	35.3	**34.4**	**49.1**	42.6	62.4	**19.8**	15.2	27.0	33.1	33.0	50.0	35.3
WCCN [19]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	37.9
Jie *et al.* [43]	60.8	54.2	34.1	14.9	13.1	54.3	53.4	**58.6**	3.7	53.1	8.3	43.4	49.8	69.2	4.1	17.5	43.8	25.6	**55.0**	50.1	38.3
PCL-OB-G-VGG_M	63.2	58.0	37.8	19.6	18.9	48.9	49.5	27.9	5.6	45.5	13.7	45.8	53.4	65.9	8.2	20.7	40.4	41.7	36.9	50.5	37.6
PCL-OB-G-VGG16	58.2	66.0	41.8	24.8	27.2	**55.7**	55.2	28.5	**16.6**	51.0	17.5	28.6	49.7	70.5	7.1	25.7	**47.5**	36.6	44.1	**59.2**	40.6
PCL-OB-G-Ens.	63.4	64.2	44.2	25.6	26.4	54.5	55.1	30.5	11.6	51.0	15.8	39.4	55.9	70.7	8.2	**26.3**	46.9	41.3	44.1	57.7	41.6
PCL-OB-G-Ens.+FRCNN	**69.0**	**71.3**	**56.1**	**30.3**	**27.3**	55.2	**57.6**	30.1	8.6	**56.6**	18.4	43.9	**64.6**	**71.8**	7.5	23.0	46.0	**44.1**	42.6	58.8	**44.2**

TABLE 4
Results (CorLoc in %) for different methods on the VOC 2012 trainval set. See Section 4.3 for the definitions of the PCL-based methods.

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mean
WSDDN+context [18]	78.3	70.8	52.5	34.7	36.6	80.0	58.7	38.6	27.7	71.2	32.3	48.7	76.2	77.4	16.0	48.4	69.9	47.5	66.9	62.9	54.8
Jie *et al.* [43]	82.4	68.1	54.5	38.9	35.9	**84.7**	73.1	**64.8**	17.1	78.3	22.5	57.0	70.8	86.6	18.7	49.7	80.7	45.3	70.1	77.3	58.8
PCL-OB-G-VGG_M	82.1	81.6	67.0	48.4	42.7	78.6	73.1	40.1	24.7	82.2	42.1	**61.6**	83.4	87.1	21.7	53.7	80.7	**64.9**	63.8	78.5	62.9
PCL-OB-G-VGG16	77.2	83.0	62.1	55.0	49.3	83.0	75.8	37.7	**43.2**	81.6	**46.8**	42.9	73.3	90.3	21.4	56.7	84.4	55.0	62.9	82.5	63.2
PCL-OB-G-Ens.	82.7	84.8	69.5	56.4	49.2	80.0	76.2	39.4	35.4	82.8	45.2	51.4	82.2	89.6	21.9	59.0	83.4	62.9	66.4	82.4	65.0
PCL-OB-G-Ens.+FRCNN	**86.7**	**86.7**	**74.8**	**56.8**	**53.8**	84.2	**80.1**	42.0	36.4	**86.7**	46.5	54.1	**87.0**	**92.7**	**24.6**	**62.0**	**86.2**	63.2	**70.9**	**84.2**	**68.0**

4.2.7 *The influence of the IoU threshold I'_t*

We also analyse the influence of I'_t defined in Section 3.4.2 and the 5-th line of Algorithm 4. As shown in Fig. 7, $I'_t = 0.5$ outperforms other choices. Therefore, we set I'_t to 0.5 for the other experiments.

4.2.8 *The influence of multi-scale training and testing*

Previously our experiments are conducted based on five image scales for training and testing. Here we show the influence of this multi-scale setting. We train and test our method using a single image scale 600 as the default scale setting of FRCNN [7]. The single-scale results are mAP 37.4% and CorLoc 55.5% which are much worse than our multi-scale results (mAP 40.8% and CorLoc 59.6%). Therefore, we use five image scales as many WSOD networks [17], [18], [19].

4.2.9 *The number of proposal cluster centers*

As we stated in Section 3.4.1 (2), the number of proposal cluster centers (*i.e.*, N^k) changes for each image in each training iteration. Here we give some typical values of N^k. In the beginning of training, the proposal scores are very noisy and thus the selected top ranking proposals to form graphs are scattered in images, which results in dozens of

TABLE 5
Results (mAP in %) for different methods on the ImageNet dataset. See Section 4.3 for the definitions of the PCL-based methods.

Method	Results
Ren *et al*. [12]	9.6
Li *et al*. [54]	10.8
WCCN [19]	16.3
PCL-OB-G-VGG_M	14.4
PCL-OB-G-VGG16	18.4
PCL-OB-G-Ens.	18.8
PCL-OB-G-Ens.+FRCNN	**19.6**

proposal cluster centers for each image. After some (about 3K) training iterations, the proposal scores are more reliable and our method finds 1∼3 proposal cluster centers for each positive object class. To make the training more stable in the beginning, for each positive object class we empirically select at most five proposal cluster centers which have higher scores, and the number of selected proposal cluster centers does not influence the performance much.

4.3 Comparison with other methods

Here we compare our best performed strategy PCL-OB-G, *i.e.*, using graph-based method and treating clusters as bags to train the network online, with other methods.

We first report our results for each class on VOC 2007 and 2012 in Table 1, Table 2, Table 3, and Table 4. It is obvious that our method outperforms other methods [17], [18] using single model VGG_M or VGG16 (PCL-OB-G+VGG_M and PCL-OB-G+VGG16 in tables.) Our single model results even better than others by combining multiple different models (*e.g.*, ensemble of models) [17], [19], [43], [54]. Specially, our method obtains much better results compared with other two methods also using the same basic MIL network [17], [18]. Importantly, [17] also equips the weighted sum pooling with objectness measure of EdgeBox [47] and the spatial regulariser, and [18] adds context information into the network, both of which are more complicated than our basic MIL network. We believe that our performance can be improved by choosing better basic MIL networks, like the complete network in [17] and using context information [18]. As reimplementing their method completely is non-trivial, here we only choose the simplest architecture in [17]. Even in this simplified case, our method achieves very promising results.

Our results can also be improved by combing multiple models. As shown in the tables, there are little improvements from the ensemble of the VGG_M and VGG16 models (PCL-OB-G-Ens. in tables). Here we do the ensemble by summing up the scores produced by the two models. Also, as mentioned in Section 4.1, similar to [19], [43], [54], we train a FRCNN detector using top-scoring proposals produced by PCL-OB-G-Ens. as groundtruths (PCL-OB-G-Ens.+FRCNN in tables). As we can see, the performance is improved further.

We then show results of our method on the large scale ImageNet detection dataset in Table 5. We observe similar phenomenon that our method outperforms other methods by a large margin.

TABLE 6
Results (mAP@0.5 and mAP@[.5, .95] in %) of different methods on the MS-COCO dataset. See Section 4.3 for the definitions of the PCL-based methods.

Method	mAP@0.5	mAP@[.5, .95]
Ge *et al*. [57]	19.3	8.9
PCL-OB-G-VGG_M	16.6	7.3
PCL-OB-G-VGG16	19.4	8.5
PCL-OB-G-Ens.	19.5	8.6
PCL-OB-G-Ens.+FRCNN	**19.6**	**9.2**

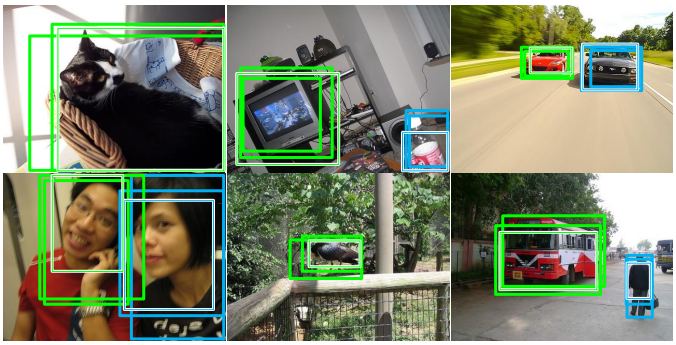


Fig. 8. Some learned proposal clusters. The proposals with white are cluster centers. Proposals with the same color belong to the same cluster. We omit the background cluster for simplification.

We finally report results of our method on MS-COCO in Table 6. Our method obtains better performance than the recent work [57]. In particular, Ge *et al*. [57] use the method proposed in our conference version paper [21] as a basic component. We can expect to obtain better detection performance through replacing our conference version method in [57] by our newly proposed method here, which we would like to explore in the future.

4.4 Qualitative results

We first show some proposal clusters generated by our method in Fig. 8. As we can see, the cluster centers contain at least parts of objects and are able to cover adaptive number of objects for each class.

We then show qualitative comparisons among the WSDDN [17], the WSDDN+context [18], and our PCL method, both of which use the same basic MIL network. As shown in Fig. 9, we can observe that for classes such as bike, car, cat, *etc*., our method tends to provide more accurate detections, whereas other two methods sometimes fails by producing boxes that are overlarge or only contain parts of objects (the first four rows in Fig. 9). But for some classes such as person, our method sometimes fails by only detecting parts of objects such as the head of person (the fifth row in Fig. 9). Exploiting context information sometimes help the detection (as in WSDDN+context [18]), we believe our method can be further improved by incorporating context information into our framework. All these three methods (actually almost all weakly supervised object detection methods) suffers from two problems: producing boxes that not only contain the target object but also include their adjacent similar objects, or only detecting parts of object for objects with deformation (the last row in Fig. 9).

Fig. 9. Some visualization comparisons among the WSDDN [17], the WSDDN+context [18], and our method (PCL) (in each image only the top-scoring box is shown). Green rectangle indicates success cases (IoU>0.5), red rectangle indicates failure cases (IoU<0.5), and yellow rectangle indicates groundtruths. The first four rows show examples that our method outperforms other two methods (with larger IoU). The fifth row shows examples that our method is worse than other two methods (with smaller IoU). The last row shows failure examples for both three methods.

We finally visualize some success and failure detection results on VOC 2007 trainval by PCL-Ens.+FRCNN, as in Fig. 10. We observe similar phenomena as in Fig. 9. Our method is robust to the size and aspect of objects, especially for rigid objects. The main failures for these rigid objects are always due to overlarge boxes that not only contain objects, but also include adjacent similar objects. For non-rigid objects like "cat", "dog", and "person", they often have great deformations, but their parts (*e.g.*, head of person) have much less deformation, so our detector is still inclined to find these parts. An ideal solution is yet wanted because there is still room for improvement.

4.5 Runtime

The runtime comparisons between our method and our basic MIL network [17] are shown in Table 7, where the runtime of proposal generation is not considered. As we can see, although our method has more components than our basic MIL network [17], our method takes almost the same testing time as it. This is because all our output streams share the same proposal feature computations. The small extra training computations of our method mainly come from the procedures to find proposal cluster centers and generate proposal clusters. Although with small extra training computations, our method obtains much better detection results than the basic MIL network.

TABLE 7
Runtime comparisons between our method ("PCL" in table) and our basic MIL network [17] ("Basic" in table).

	PCL		Basic	
	VGG_M	VGG16	VGG_M	VGG16
Training (second/iteration)	1.11	1.51	0.99	1.40
Testing (second/image)	0.71	1.22	0.71	1.21

5 Conclusion

In this paper, we propose to generate proposal clusters to learn refined instance classifiers for weakly supervised

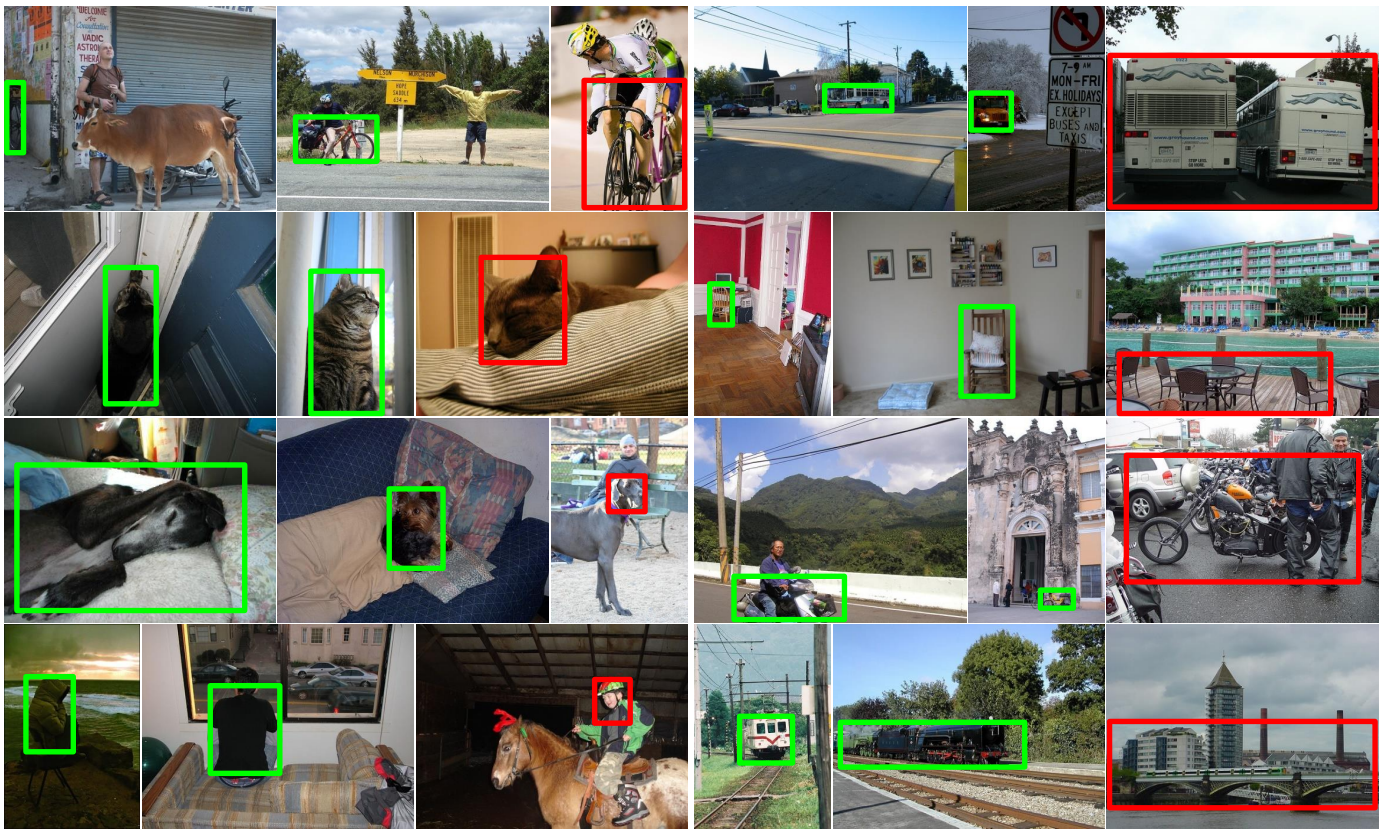


Fig. 10. Some detection results for class bicycle, bus, cat, chair, dog, motorbike, person, and train (in each image only the top-scoring box is shown). Green rectangle indicates success cases (IoU>0.5), and red rectangle indicates failure cases (IoU<0.5).

object detection. We propose two strategies for proposal cluster generation and classifier refinement, both of which can boost the performance significantly. The classifier refinement is implemented by multiple output streams corresponding to some instance classifiers in multiple instance learning networks. An online training algorithm is introduced to train the proposed network end-to-end for effectiveness and efficiency. Experiments show substantial and consistent improvements by our method. We observe that the most common failure cases of our algorithm are connected with the deformation of non-rigid objects. In the future, we will concentrate on this problem. In addition, we believe our learning algorithm has the potential to be applied in other weakly supervised visual learning tasks such as weakly supervised semantic segmentation. We will also explore how to apply our method to these related applications.

ACKNOWLEDGEMENTS

This work was supported by NSFC (No. 61733007, No. 61572207, No. 61876212, No. 61672336, No. 61573160), ONR with grant N00014-15-1-2356, Hubei Scientific and Technical Innovation Key Project, and the Program for HUST Academic Frontier Youth Team. The corresponding author of this paper is Xinggang Wang.

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[3] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2015.

[4] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[5] 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 *European Conference on Computer Vision*, 2014, pp. 740–755.

[6] 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.

[7] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

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

[10] 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*, 2016, pp. 21–37.

[11] Z. Zhang, S. Qiao, C. Xie, W. Shen, B. Wang, and A. L. Yuille, "Single-shot object detection with enriched semantics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5813–5821.

[12] W. Ren, K. Huang, D. Tao, and T. Tan, "Weakly supervised large scale object localization with multiple instance learning and bag splitting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 405–416, 2016.

[13] R. G. Cinbis, J. Verbeek, and C. Schmid, "Weakly supervised object localization with multi-fold multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 1, pp. 189–203, 2017.

[14] X. Wang, Z. Zhu, C. Yao, and X. Bai, "Relaxed multiple-instance SVM with application to object discovery," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1224–1232.

[15] M. Shi, H. Caesar, and V. Ferrari, "Weakly supervised object localization using things and stuff transfer," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3381–3390.

[16] P. Tang, X. Wang, Z. Huang, X. Bai, and W. Liu, "Deep patch learning for weakly supervised object classification and discovery," *Pattern Recognition*, vol. 71, pp. 446–459, 2017.

[17] H. Bilen and A. Vedaldi, "Weakly supervised deep detection networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2846–2854.

[18] V. Kantorov, M. Oquab, M. Cho, and I. Laptev, "ContextLocNet: Context-aware deep network models for weakly supervised localization," in *European Conference on Computer Vision*, 2016, pp. 350–365.

[19] A. Diba, V. Sharma, A. Pazandeh, H. Pirsiavash, and L. Van Gool, "Weakly supervised cascaded convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 914–922.

[20] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[21] P. Tang, X. Wang, X. Bai, and W. Liu, "Multiple instance detection network with online instance classifier refinement," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2843–2851.

[22] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, no. 1, pp. 31–71, 1997.

[23] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Advances in Neural Information Processing Systems*, 2003, pp. 577–584.

[24] Q. Zhang and S. A. Goldman, "EM-DD: An improved multiple-instance learning technique," in *Advances in Neural Information Processing Systems*, 2002, pp. 1073–1080.

[25] X. Wang, Y. Yan, P. Tang, X. Bai, and W. Liu, "Revisiting multiple instance neural networks," *Pattern Recognition*, vol. 74, pp. 15–24, 2018.

[26] Q. Li, J. Wu, and Z. Tu, "Harvesting mid-level visual concepts from large-scale internet images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 851–858.

[27] P. Tang, X. Wang, B. Feng, and W. Liu, "Learning multi-instance deep discriminative patterns for image classification," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3385–3396, 2017.

[28] D. Pathak, E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional multi-class multiple instance learning," in *International Conference on Learning Representations Workshop*, 2015.

[29] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1713–1721.

[30] C. Zhang, J. C. Platt, and P. A. Viola, "Multiple instance boosting for object detection," in *Advances in Neural Information Processing Systems*, 2006, pp. 1417–1424.

[31] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.

[32] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free?-weakly-supervised learning with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 685–694.

[33] O. Chum and A. Zisserman, "An exemplar model for learning object classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[34] T. Deselaers, B. Alexe, and V. Ferrari, "Weakly supervised localization and learning with generic knowledge," *International Journal of Computer Vision*, vol. 100, no. 3, pp. 275–293, 2012.

[35] M. Pandey and S. Lazebnik, "Scene recognition and weakly supervised object localization with deformable part-based models," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 1307–1314.

[36] Z. Shi, T. M. Hospedales, and T. Xiang, "Bayesian joint modelling for object localisation in weakly labelled images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 10, pp. 1959–1972, 2015.

[37] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darrell, "On learning to localize objects with minimal supervision," in *International Conference on Machine Learning*, vol. 32, 2014, pp. 1611–1619.

[38] H. O. Song, Y. J. Lee, S. Jegelka, and T. Darrell, "Weakly-supervised discovery of visual pattern configurations," in *Advances in Neural Information Processing Systems*, 2014, pp. 1637–1645.

[39] H. Bilen, M. Pedersoli, and T. Tuytelaars, "Weakly supervised object detection with convex clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1081–1089.

[40] C. Wang, W. Ren, K. Huang, and T. Tan, "Weakly supervised object localization with latent category learning," in *European Conference on Computer Vision*, 2014, pp. 431–445.

[41] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2189–2202, 2012.

[42] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[43] Z. Jie, Y. Wei, X. Jin, J. Feng, and W. Liu, "Deep self-taught learning for weakly supervised object localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1377–1385.

[44] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan, "HCP: A flexible CNN framework for multi-label image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1901–1907, 2016.

[45] 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.

[46] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[47] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *European Conference on Computer Vision*, 2014, pp. 391–405.

[48] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, 1967, pp. 281–297.

[49] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proceedings of the British Machine Vision Conference*, 2014.

[50] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

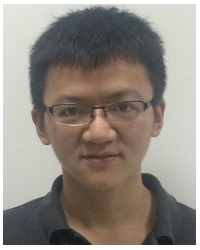
[51] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations*, 2016.

[52] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[53] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 1, pp. 128–140, 2017.

[54] D. Li, J.-B. Huang, Y. Li, S. Wang, and M.-H. Yang, "Weakly supervised object localization with progressive domain adaptation,"

in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3512–3520.
[55] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia*, 2014, pp. 675–678.
[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.
[57] W. Ge, S. Yang, and Y. Yu, "Multi-evidence filtering and fusion for multi-label classification, object detection and semantic segmentation based on weakly supervised learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1277–1286.



Peng Tang received the B.S. degree in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST) in 2014. He is currently pursuing the Ph.D. degree in the School of Electronic Information and Communications at HUST, and visiting the Department of Computer Science at Johns Hopkins University. He was an intern at Microsoft Research Asia in 2017. His research interests include image classification and object detection in images/videos.

Xinggang Wang is an assistant professor of School of Electronics Information and Communications of Huazhong University of Science and Technology (HUST). He received his Bachelor degree in communication and information system and Ph.D. degree in computer vision both from HUST. From May 2010 to July 2011, he was with the Department of Computer and Information Science, Temple University, Philadelphia, PA., as a visiting scholar. From February 2013 to September 2013, he was with the University of California, Los Angeles (UCLA), as a visiting graduate researcher. He is a reviewer of IEEE Trans on PAMI, IEEE Trans on Image Processing, IEEE Trans. on Cybernetics, Pattern Recognition, Computer Vision and Image Understanding, Neurocomputing, NIPS, ICML, CVPR, ICCV and ECCV etc. His research interests include computer vision and machine learning, especially object recognition.

Song Bai received the B.S. and Ph.D. degree in Electronics and Information Engineering from Huazhong University of Science and Technology (HUST), Wuhan, China in 2013 and 2018, respectively. He was with University of Texas at San Antonio (UTSA) and Johns Hopkins University (JHU) as a research scholar. His research interests include image retrieval and classification, 3D shape recognition, person re-identification, semantic segmentation and deep learning. More information can be found in his homepage: http://songbai.site/.

Wei Shen received his B.S. and Ph.D. degree both in Electronics and Information Engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2007 and in 2012. From April 2011 to November 2011, he worked in Microsoft Research Asia as an intern. In 2012, he joined School of Communication and Information Engineering, Shanghai University as an Assistant Professor. From 2017, he became an Associate Professor. He is currently visiting Department of Computer Science, Johns Hopkins University. His current research interests include random forests, deep learning, object detection and segmentation.

Xiang Bai received his B.S., M.S., and Ph.D. degrees from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003, 2005, and 2009, respectively, all in electronics and information engineering. He is currently a Professor with the School of Electronic Information and Communications, HUST. He is also the Vice-director of the National Center of Anti-Counterfeiting Technology, HUST. His research interests include object recognition, shape analysis, scene text recognition and intelligent systems. He serves as an associate editor for Pattern Recognition, Pattern Recognition Letters, Neurocomputing and Frontiers of Computer Science.

Wenyu Liu received the B.S. degree in Computer Science from Tsinghua University, Beijing, China, in 1986, and the M.S. and Ph.D. degrees, both in Electronics and Information Engineering, from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively. He is now a professor and associate dean of the School of Electronic Information and Communications, HUST. His current research areas include computer vision, multimedia, and machine learning. He is a senior member of IEEE.

Alan Yuille received the B.A. degree in mathematics from the University of Cambridge in 1976, and the Ph.D. degree in theoretical physics from Cambridge in 1980. He then held a post-doctoral position with the Physics Department, University of Texas, Austin, and the Institute for Theoretical Physics, Santa Barbara. He then became a Research Scientists with the Artificial Intelligence Laboratory, MIT, from 1982 to 1986, and followed this with a faculty position in the division of applied sciences, Harvard, from 1986 to 1995, rising to the position of an associate professor. From 1995 to 2002, he was a Senior Scientist with the Smith-Kettlewell Eye Research Institute in San Francisco. From 2002 to 2016, he was a Full Professor with the Department of Statistics, UCLA, with joint appointments in Psychology, Computer Science, and Psychiatry. In 2016, he became a Bloomberg Distinguished Professor of cognitive science and computer science with Johns Hopkins University. He received the Marr Prize and the Helmholtz Prize.