This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

DeepID-Net: Object Detection with Deformable Part Based Convolutional Neural Networks

Wanli Ouyang*, *Member, IEEE,* Xingyu Zeng*, *Student Member, IEEE*, Xiaogang Wang, *Member, IEEE* ,Shi Qiu *Member, IEEE*, Ping Luo, *Member, IEEE*, Yonglong Tian *Student Member, IEEE*, Hongsheng Li, *Member, IEEE*, Shuo Yang *Student Member, IEEE*, Zhe Wang, *Student Member, IEEE*, Hongyang Li, Kun Wang, Junjie Yan, Chen-Change Loy, *Member, IEEE*, Xiaoou Tang, *Fellow, IEEE*

Abstract—In this paper, we propose deformable deep convolutional neural networks for generic object detection. This new deep learning object detection framework has innovations in multiple aspects. In the proposed new deep architecture, a new deformation constrained pooling (def-pooling) layer models the deformation of object parts with geometric constraint and penalty. A new pre-training strategy is proposed to learn feature representations more suitable for the object detection task and with good generalization capability. By changing the net structures, training strategies, adding and removing some key components in the detection pipeline, a set of models with large diversity are obtained, which significantly improves the effectiveness of model averaging. The proposed approach improves the mean averaged precision obtained by RCNN [16], which was the state-of-the-art, from 31% to 50.3% on the ILSVRC2014 detection test set. It also outperforms the winner of ILSVRC2014, GoogLeNet, by 6.1%. Detailed component-wise analysis is also provided through extensive experimental evaluation, which provides a global view for people to understand the deep learning object detection pipeline.

Index Terms—CNN, convolutional neural networks, object detection, deep learning, deep model

1 Introduction

Object detection is one of the fundamental challenges in computer vision and has attracted a great deal of research interest [11], [50]. Intra-class variation in appearance and deformation are among the main challenges of this task.

Because of its power in learning features, the convolutional neural network (CNN) is being widely used in recent large-scale detection and recognition systems [60], [54], [21], [27], [82], [81]. Since training deep models is a non-convex optimization problem with millions of parameters, the choice of a good initial point is a crucial but unsolved problem, especially when deep CNN goes deeper [60], [54], [27]. It is also easy to overfit to a small training set. Researchers find that supervised pretraining on large-scale image classification data and then finetuning for the target object detection task is a practical solution [10], [45], [78], [16]. However, we observe that there is still a gap between the pretraining task and the finetuning task that makes pretraining less effective. The problem of the training scheme is the mismatch between pretraining for the image classification task and fine-tuning for the object detection task. For image classification, the input is a whole image and the task is to recognize the object within this image. Therefore, learned feature representations have robustness to scale and location change of objects in images. Taking Fig. 1(a) as an example, no matter how large and where a person is in the image, the image should be classified as person. However, robustness to object size and location is not required for object detection. For object detection, candidate regions are cropped and warped before they are used as input of the deep model. Therefore, the positive candidate regions for the object class person should have their locations aligned and their sizes normalized. On the contrary, the deep model is expected to be sensitive to the change in position and size in order to accurately localize objects. An example to illustrate the mismatch is shown in Fig. 1 (a). Because of such mismatch, the image classification task is not an ideal choice to pretrain the deep model for object detection. Therefore, a new pretraining scheme is proposed to train the deep model for object detection more effectively.

Part deformation handling is a key factor for the recent progress in object detection [12], [83], [13], [73], [37]. Our new CNN layer is motivated by three observations. First, deformable visual patterns are shared by objects of different categories. For example, the circular visual pattern is shared by both banjo and ipod as shown in Fig. 1(b). Second, the regularity on deformation exists for visual patterns at different semantic levels. For example, human upper bodies, human heads, and human mouths are parts at different semantic levels with different deformation properties. Third, a deformable part at a higher level is composed of deformable parts at a lower level. For example, a human upper body is composed of a head and other body parts. With these observations, we design a new deformation-constrained pooling (def-pooling) layer to learn

- *Wanli Ouyang (equal contribution), Xingyu Zeng (equal contribution), Xiaogang Wang, Hongsheng Li, Zhe Wang, and Hongyang Li are with the Department of Electronic Engineering at the Chinese University of Hong Kong, Hong Kong. Shi Qiu, Ping Luo, Yonglong Tian, Shuo Yang, Chen-Change Loy, and Xiaoou Tang are with the Department of Electronic Engineering at the Chinese University of Hong Kong, Hong Kong. Wanli Ouyang and Xiaogang Wang are corresponding authors.*
  E-mail: wlouyang, xgwang@ee.cuhk.edu.hk.

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

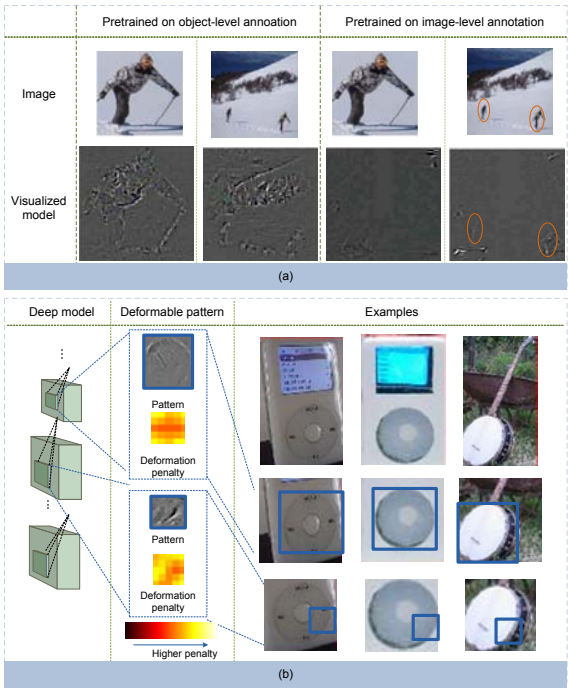


Figure 1. The motivation of this paper in new pretraining scheme (a) and jointly learning feature representation and deformable object parts shared by multiple object classes at different semantic levels (b). In (a), a model pretrained on image-level annotation is more robust to size and location change while a model pretrained on object-level annotation is better in representing objects with tight bounding boxes. In (b), when ipod rotates, its circular pattern moves horizontally at the bottom of the bounding box. Therefore, the circular patterns have smaller penalty moving horizontally but higher penalty moving vertically. The curvature part of the circular pattern are often at the bottom right positions of the circular pattern. Magnitudes of deformation penalty are normalized to make them comparable across the two examples in (a) for visualization. *Best viewed in color*.

the shared visual patterns and their deformation properties for multiple object classes at different semantic levels and composition levels.

The performance of deep learning systems depends significantly on implementation details [4]. However, an evaluation of the performance of the recent deep architectures on the common ground for large-scale object detection is missing. As a respect to the investigation on details in deep learning [4], [16], this paper compares the performance of recent deep models, including AlexNet [25], ZF [75], Overfeat [52], and GoogLeNet [60] under the same setting for different pretraining-finetuning schemes. We also provide experimental analysis on the properties that cause the accuracy variation in different object classes.

In this paper, we propose a deformable deep convolutional neural network for object detection; named as DeepID-Net. In DeepID-Net, we jointly learn the feature representation and part deformation for a large number of object categories. We also investigate many aspects in effectively and efficiently training and aggregating the deep models, including bounding box rejection, training schemes, context modeling, and model averaging. The proposed new framework significantly advances the state-of-the-art for deep learning based generic object detection, such as the well known RCNN [16] framework. This paper also provides detailed component-wise experimental results on how our approach can improve the mean Averaged Precision (AP) obtained by RCNN [16] from 31.0% to mean AP 50.3% step-by-step on the ImageNet Large Scale Visual Recognition Challenge 2014 (ILSVRC2014) object detection task.

The contributions of this paper are as follows:

1) A new deep learning framework for object detection. It effectively integrates feature representation learning, part deformation learning, context modeling, model averaging, and bounding box location refinement into the detection system. Detailed component-wise analysis is provided through extensive experimental evaluation. This paper is also the first to investigate the influence of CNN structures for the large-scale object detection task under the same setting. By changing the configuration of this framework, multiple detectors with large diversity are generated, which leads to more effective model averaging.
2) A new scheme for pretraining the deep CNN model. We propose to pretrain the deep model on the ImageNet image classification and localization dataset with 1000-class object-level annotations instead of with image-level annotations, which are commonly used in existing deep learning object detection [16], [60]. Then the deep model is fine-tuned on the ImageNet/PASCAL-VOC object detection dataset with 200/20 classes, which are the target object classes in the two datasets.
3) A new deformation constrained pooling (def-pooling) layer, which enriches the deep model by learning the deformation of object parts at any information abstraction levels. The def-pooling layer can be used for replacing the max-pooling layer and learning the deformation properties of parts.
4) Analysis on the object properties that influence the variation in object detection accuracy for different classes.

Preliminary version of this paper is published in [38]. This paper include more analysis on the proposed approach and add experimental investigation on the properties that influence the accuracy in detecting objects.

The models pretrained by both image-level annotation and object-level annotation for AlexNet [25], ZF [75], overfeat [52] and GoogLeNet [60] and the models after fine-tuning on ILSVRC2014 are provided online [1].

2 Related Work

Since many objects have non-rigid deformation, the ability to handle deformation improves detection performance. Deformable part-based models were used in [12], [83], [41], [70], [65] for handling translational movement of parts. To handle more complex articulations, size change and rotation of parts were modeled in [13], and mixture of part appearance and articulation types were modeled in [3], [72]. A dictionary of shared deformable patterns was learned in [20]. In these approaches, features were manually designed.

1. www.ee.cuhk.edu.hk/~wlouyang/projects/ImageNet

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

Because of the power on learning feature representation, deep models have been widely used for object recognition, detection and other vision tasks [25], [52], [75], [21], [53], [85], [19], [27], [16], [35], [40], [76], [77], [33], [29], [57], [59], [57], [58], [31], [32], [30], [80], [79], [68], [69], [15], [47], [26], [46], [34], [39], [42], [43]. Krizhevsky *et al.* proposed a neural network with 60 million parameters and 650,000 neurons [25]. This neural network was the first to show the power of deep CNN in large-scale computer vision task. Later on, Razavian *et al.* [45] showed that the OverFeat network [52] trained for object classification on ILSVRC13 was a good feature representation for the diverse range of recognition tasks of object image classification, scene recognition, fine grained recognition, attribute detection and image retrieval applied to a diverse set of datasets. As a further step of using the model trained for ImageNet object classification, Girshick *et al.* found finetuning the CNN pretrained on the ImageNet object classification to be effective on various object detection benchmark datasets [16]. In existing deep CNN models, max pooling and average pooling were useful in handling deformation but could not learn the deformation penalty and geometric models of object parts. The deformation layer was first proposed in [36] for pedestrian detection. In this paper, we extend it to general object detection on ImageNet. In [36], the deformation layer was constrained to be placed after the last convolutional layer. In this work the def-pooling layer can be placed after all the convolutional layers to capture geometric deformation at all the information abstraction levels. In [36], it was assumed that a pedestrian only had one instance of a body part, so each part filter only had one optimal response in a detection window. In this work, it is assumed that an object has multiple instances of a part (e.g. a car has many wheels), so each part filter is allowed to have multiple response peaks in a detection window. Moreover, we allow multiple object categories to share deformable parts and jointly learn them with a single network. This new model is more suitable for general object detection.

The use of context has gained attention in recent works on object detection. The context information investigated in literature includes regions surrounding objects [5], [8], [14], object-scene interaction [9], [22], and the presence, location, orientation and size relationship among objects [2], [64], [66], [7], [44], [14], [56], [9], [74], [8], [71], [40], [6], [51], [61]. In this paper, we use whole-image classification scores over a large number of classes from a deep model as global contextual information to refine detection scores.

Besides feature learning, deformation modeling, and context modeling, there were also other important components in the object detection pipeline, such as pretraining [16], network structures [52], [75], [25], refinement of bounding box locations [16], and model averaging [75], [25], [21]. While these components were studies individually in different works, we integrate them into a complete pipeline and take a global view of them with component-wise analysis under the same experimental setting. It is an important step to understand and advance deep learning based object detection.

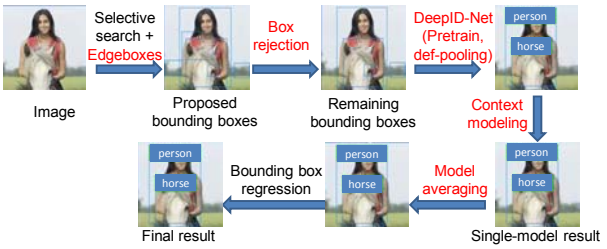


Figure 2. Overview of our approach. Detailed description is given in Section 3.1. Texts in red highlight the steps that are not present in RCNN [16].

3 METHOD

3.1 Overview of our approach

An overview of our proposed approach is shown in Fig. 2. We take the ImageNet object detection task as an example. The ImageNet image classification and localization dataset with 1,000 classes is chosen to pretrain the deep model. Its object detection dataset has 200 object classes. In the experimental section, the approach is also applied to the PASCAL VOC. The pretraining data keeps the same, while the detection dataset only has 20 object classes. The steps of our approach are summarized as follows.

1) Selective search proposed in [55] and edgeboxes proposed in [84] are used to propose candidate bounding boxes.
2) An existing detector, RCNN [16] in our experiment, is used to reject bounding boxes that are most likely to be background.
3) An image region in a bounding box is cropped and fed into the DeepID-Net to obtain 200 detection scores. Each detection score measures the confidence on the cropped image containing one specific object class. Details are given in Section 3.2.
4) The 1000-class whole-image classification scores of a deep model are used as contextual information to refine the detection scores of each candidate bounding box. Details are given in Section 3.6.
5) Average of multiple deep model outputs is used to improve the detection accuracy. Details are given in Section 3.7.
6) Bounding box regression proposed in RCNN [16] is used to reduce localization errors.

3.2 Architecture of DeepID-Net

DeepID-Net in Fig. 3 has three parts:

(a) The baseline deep model. The ZF model proposed in [75] is used as the default baseline deep model when it is not specified.
(b) Branches with def-pooling layers. The input of these layers is the conv5, the last convolutional layer of the baseline model. The output of conv5 is convolved with part filters of variable sizes and the proposed def-pooling layers in Section 3.4 are used to learn the deformation constraint of these part filters. Parts (a)-(b) output 200-class object detection scores. For the cropped image region that contains a horse as shown in Fig. 3(a), its ideal output should have a high score for the object class horse but low scores for other classes.

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

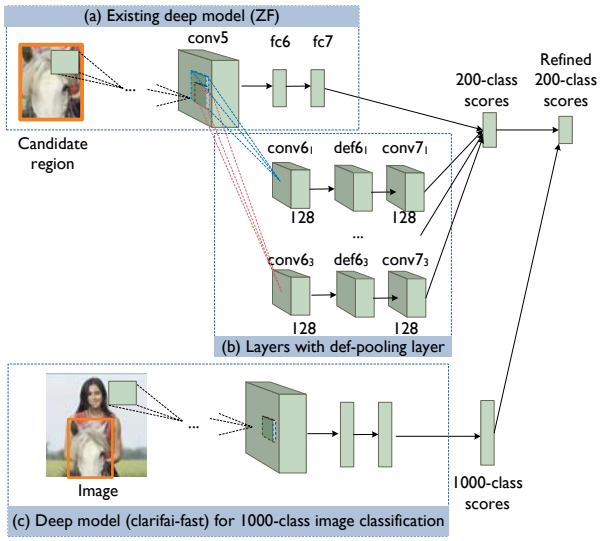


Figure 3. Architecture of DeepID-Net with three parts: (a) baseline deep model, which is ZF [75] in our single-model detector; (b) layers of part filters with variable sizes and def-pooling layers; (c) deep model to obtain 1000-class image classification scores. The 1000-class image classification scores are used to refine the 200-class bounding box classification scores.

(c) The deep model (ZF) to obtain image classification scores of 1000 classes. Its input is the whole image, as shown in Fig. 3(c). The image classification scores are used as contextual information to refine the classification scores of bounding boxes. Detail are given in Section 3.6.

3.3 New pretraining strategy

The widely used training scheme in deep learning based object detection [16], [78], [60] including RCNN is denoted by Scheme 0 and described as follows:

1) Pretrain deep models by using the image classification task, i.e. using image-level annotations from the ImageNet image classification and localization training data.
2) Fine-tune deep models for the object detection task, i.e. using object-level annotations from the object detection training data. The parameters learned in Step (1) are used as initialization.

The deep model structures at the pretraining and fine-tuning stages are only different in the last fully connected layer for predicting labels ($1,000$ classes for the ImageNet classification task vs. 200 classes for the ImageNet detection task). Except for the last fully connected layers for classification, the parameters learned at the pretraining stage are directly used as initial values for the fine-tuning stage.

We propose to pretrain the deep model on a large auxiliary object detection training data instead of the image classification data. Since the ImageNet Cls-Loc data provides object-level bounding boxes for 1000 classes, more diverse in content than the ImageNet Det data with 200 classes, we use the image regions cropped by these bounding boxes to pretrain the baseline deep model in Fig. 3(a). The proposed pretraining strategy is denoted as Scheme 1 and bridges the image- vs. object-level annotation gap in RCNN.

1) Pretrain the deep model with object-level annotations of $1,000$ classes from ImageNet Cls-Loc train data.
2) Fine-tune the deep model for the 200-class object detection task, i.e. using object-level annotations of 200 classes from ImageNet Det train and val_1 (validation set 1) data. Use the parameters in Step (1) as initialization.

Compared with the training scheme of RCNN, experimental results show that the proposed scheme improves mean AP by 4.5% on ImageNet Det val_2 (validation set 2). If only the 200 target classes (instead of 1,000 classes) from the ImageNet Cls-Loc train data are selected for pretraining in Step (1), the mean AP on ImageNet Det val_2 drops by 5.7%.

Another potential mismatch between pretraining and fine-tuning comes from the fact that the ImageNet classification and localization (Cls-Loc) data has $1,000$ classes, while the ImageNet detection (Det) data only targets on 200 classes, most of which are within the $1,000$ classes. In many practical applications, the number of object classes to be detected is small. People question the usefulness of auxiliary training data outside the target object classes. Our experimental study shows that feature representations pretrained with $1,000$ classes have better generalization capability, which leads to better detection accuracy than pretraining with a subset of the Cls-Loc data only belonging to the 200 target classes in detection.

3.4 Def-pooling layer

3.4.1 DPM and its relationship with CNN

In the deformable part based model (DPM) [12] for object detection, the following steps are used at the testing stage:

1) Extract HOG feature maps from the input image.
2) Obtain the part detection score maps by filtering the HOG feature maps with the learned part filters/detectors. The part filters are learned by latent SVM.
3) Obtain deformable part score maps by subtracting deformation penalty from part detection score maps.
4) Sum up the deformable part score maps from multiple parts to obtain the final object detection score map.

Denote the convolutional layer at the lth layer by $conv_l$. Denote the output maps of $conv_l$ by $\mathbf{M}^l$. The steps above for DPM have the following relationship for CNN:

1) The HOG feature map in DPM corresponds to the output of a convolutional layer. Consecutive convolutional layers and pooling layers can be considered as extracting feature maps from input image.
2) The part detection maps in DPM correspond to the output response maps of the convolutional layer. For example, the output of $conv_{l-1}$ is the feature maps $\mathbf{M}^{l-1}$, which are treated as input feature maps of $conv_l$. Filtering on HOG feature maps using part filters in DPM is similar to filtering on the feature maps $\mathbf{M}^{l-1}$ using the filters of $conv_l$ in CNN. Each output channel of $conv_l$ corresponds to a part detection map in DPM. The filter of $conv_l$ for an output channel corresponds to a part filter in DPM. The response map in CNN is called part detection map in the following of this paper.
3) The deformation penalty in DPM for each part corresponds to the deformation penalty in our proposed def-pooling layer for CNN. Details are given in Section 3.4.2.

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

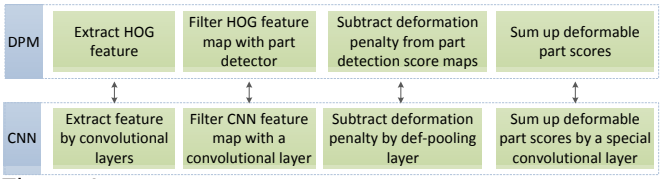


Figure 4. The relationship between the operations in the DPM and the CNN.

4) Summing up the deformable part score maps for multiple parts in DPM is similar to summing up multiple def-pooling layer output maps with a special case of convolutional layer. Def-pooling layer output maps can be summed up by using a convolutional layer after the def-pooling layer with filter size 1×1 and filter coefficients being constant 1 for the convolutional layer.

In summary, HOG feature extraction, part detector filtering, deformation penalty subtraction and part detection scores aggregation in DPM [12] have their corresponding operations in the CNN as shown in Fig. 4.

3.4.2 Definition of the def-pooling layer

Similar to max-pooling and average-pooling, the input of a def-pooling layer is the output of a convolutional layer. The convolutional layer produces C maps of size $W \times H$. Denote $\mathbf{M}_c$ as the cth map. Denote the (i,j)th element of $\mathbf{M}_c$ by $m_c^{(i,j)}$, $i = 1,\dots,W, j = 1,\dots,H$. The def-pooling layer takes a small block with center (x,y) and size $(2R+1) \times (2R+1)$ from the $\mathbf{M}_c$ and produce the element of the output as follows:

$$\tilde{b}_c^{(x,y)} = \max_{\delta_x,\delta_y \in \{-R,\cdots,R\}} \tilde{m}_c(\mathbf{z}_{\delta_x,\delta_y}, \delta_x, \delta_y), \quad (1)$$

$$\text{where } x = 1,\dots,W, y = 1,\dots,H, \quad (2)$$

$$\tilde{m}_c(\mathbf{z}_{\delta_x,\delta_y}, \delta_x, \delta_y) = m_c^{\mathbf{z}_{\delta_x,\delta_y}} - \phi(\delta_x, \delta_y) \quad (3)$$

$$\mathbf{z}_{\delta_x,\delta_y} = [x,y]^{\mathrm{T}} + [\delta_x, \delta_y]^{\mathrm{T}}, \quad (4)$$

$$\phi(\delta_x, \delta_y) = \sum_{n=1}^{N} a_{c,n} d_{c,n}^{\delta_x,\delta_y}. \quad (5)$$

- (x,y) denotes the assumed anchor location of object part.
- (δ_x, δ_y) denotes the translation/displacement of object part from the anchor position.
- $\mathbf{z}_{\delta_x,\delta_y}$ as defined in (4) is the deformed location from the assumed anchor position.
- $m_c^{\mathbf{z}_{\delta_x,\delta_y}}$ in (3) is the element in $\mathbf{M}_c$ at the location $\mathbf{z}_{\delta_x,\delta_y}$. It is considered as the score of matching the cth part filter with the features at the deformed location $\mathbf{z}_{\delta_x,\delta_y}$.
- $\phi(\delta_x, \delta_y)$ in (3) and (5) is the deformation penalty of placing the part from the assumed anchor position (x,y) to the deformed location $\mathbf{z}_{\delta_x,\delta_y}$. $a_{c,n}$ and $d_{c,n}^{\delta_x,\delta_y}$ in (5) are parameters of deformation that can be pre-defined or learned by back-propagation (BP). N denotes the number of parameters $a_{c,n}$ and $d_{c,n}^{\delta_x,\delta_y}$.
- $\tilde{m}_c(\mathbf{z}_{\delta_x,\delta_y}, \delta_x, \delta_y)$ in (1) and (3) is the deformable part score. It is obtained by subtracting the deformation penalty $\phi(\delta_x, \delta_y)$ from the visual matching score $m_c^{\mathbf{z}_{\delta_x,\delta_y}}$.
- $\tilde{b}_c^{(x,y)}$ is the (x,y)th element of the output of the def-pooling layer. For the anchor location (x,y), $\tilde{b}_c^{(x,y)}$ is obtained by taking the maximum deformable part score $\tilde{m}_c(\mathbf{z}_{\delta_x,\delta_y}, \delta_x, \delta_y)$ within the displacement range R, i.e. $\delta_x, \delta_y \in \{-R,\cdots,R\}$.

The def-pooling layer can be better understood through the following examples.

Example 1. If $N = 1$, $a_n = 1$, $d_1^{\delta_x,\delta_y} = 0$ for $|\delta_x|, |\delta_y| \le k$ and $d_1^{\delta_x,\delta_y} = \infty$ for $|\delta_x|, |\delta_y| > k$, then this corresponds to max-pooling with kernel size k. It shows that the max-pooling layer is a special case of the def-pooling layer. Penalty becomes very large when deformation reaches certain range. Since the use of different kernel sizes in max-pooling corresponds to different maps of deformation penalty that can be learned by BP, def-pooling provides the ability to learn the map that implicitly decides the kernel size for max-pooling.

Example 2. If $N = 1$ and $a_n = 1$, then $d_1^{\delta_x,\delta_y}$ is the deformation parameter/penalty of moving a part from the anchor location (x,y) by (δ_x, δ_y). If the part is not allowed to move, we have $d_1^{0,0} = 0$ and $d_1^{\delta_x,\delta_y} = \infty$ for $(\delta_x, \delta_y) \neq (0,0)$. If the part has penalty 1 when it is not at the assumed location (x,y), then we have $d_1^{0,0} = 0$ and $d_1^{\delta_x,\delta_y} = 1$ for $(\delta_x, \delta_y) \neq (0,0)$. It allows to assign different penalty to displacement in different directions. If the part has penalty 2 moving leftward and penalty 1 moving rightward, then we have $d_1^{\delta_x,\delta_y} = 1$ for $\delta_x < 0$ and $d_1^{\delta_x,\delta_y} = 2$ for $\delta_x > 0$. Fig. 6 shows some learned deformation parameters $d_1^{\delta_x,\delta_y}$. Fig. 7 shows some visualized parts.

Example 3. The deformation layer in [36] is a special case of the def-pooling layer by enforcing that $\mathbf{z}_{\delta_x,\delta_y}$ in (1) covers all the locations in $\mathrm{conv}_{l-1,i}$ and only one output with a pre-defined location is allowed for the def-pooling layer (i.e. $R = \infty$, $s_x = W$, and $s_y = H$). The proof can be found in Appendix A. To implement quadratic deformation penalty used in [12], we can predefine $\{d_{c,n}^{\delta_x,\delta_y}\}_{n=1,2,3,4} = \{\delta_x, \delta_y, (\delta_x)^2, (\delta_y)^2\}$ and learn parameters a_n. As shown in Appendix A, the def-pooling layer under this setting can represent deformation constraint in the deformable part based model (DPM) [12] and the DP-DPM [18].

Take Example 2 as an example for BP learning. $a_{c,n}$ is the parameter in this layer and d_* is pre-defined constant. Then we have:

$$\frac{\partial b_c^{(x,y)}}{\partial a_{c,n}} = -d_{c,n}^{(\Delta_x,\Delta_y)}, \qquad (\Delta_x, \Delta_y) = \mathrm{argmax}_{\delta_x,\delta_y \in \{-R,\cdots,R\}} \{m_c^{\mathbf{z}_{\delta_x,\delta_y}} - \phi(\delta_x, \delta_y)\}. \quad (6)$$

where (Δ_x, Δ_y) is the position with the maximum value in (1). The gradients for the parameters in the layers before the def-pooling layer are back-propagated like max-pooling layer.

Similar to max-pooling and average pooling, subsampling can be done as follows:

$$b_c^{(x,y)} = \tilde{b}_c^{(s_x \cdot x, s_y \cdot y)} \quad (7)$$

For $\mathbf{M}_c$ of size $W \times H$, the subsampled output has size $\frac{W}{s_x} \times \frac{H}{s_y}$. Therefore, multiple instances of an object part at multiple anchor locations are allowed for each part filer.

In our implementation, $N = 1$, $R = 2$ and Example 2 is used for def-pooling, there are no fully connected layers after $\mathrm{conv}7_1, 2, 3$ in Fig. 3. We did not find improvement on ImageNet by further increasing N and R. Further study on N and R could be done on other datasets and particular categories in the future work.

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

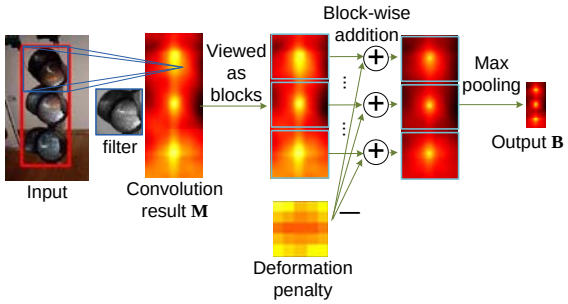


Figure 5. Def-pooling layer. The part detection map and the deformation penalty are summed up. Block-wise max pooling is then performed on the summed map to obtain the output $\mathbf{B}$ of size $\frac{H}{s_y} \times \frac{W}{s_x}$ (3×1 in this example).

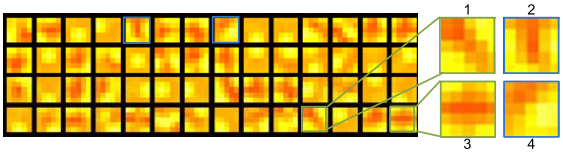


Figure 6. The learned deformation penalty for different visual patterns. The penalties in map 1 are low at diagonal positions. The penalties in map 2 and 3 are low at vertical and horizontal locations separately. The penalties in map 4 are high at the bottom right corner and low at the upper left corner.

3.4.3 Analysis

A visual pattern has different spatial distributions in different object classes. For example, traffic lights and ipods have geometric constraints on the circular visual pattern in Fig. 8. The weights connecting the convolutional layers conv7_1 - conv7_3 in Fig. 3 and classification scores are determined by the spatial distributions of visual patterns for different classes. For example, the car class will have large positive weights in the bottom region but negative weights in the upper region for the circular pattern. On the other hand, the traffic light class will have positive weights in the upper region for the circular pattern.

A single output of the convolutional layer conv7_1 in Fig. 3 is from multiple part scores in def6_1. The relationship between parts of the same layer def6_1 is modeled by conv7_1.

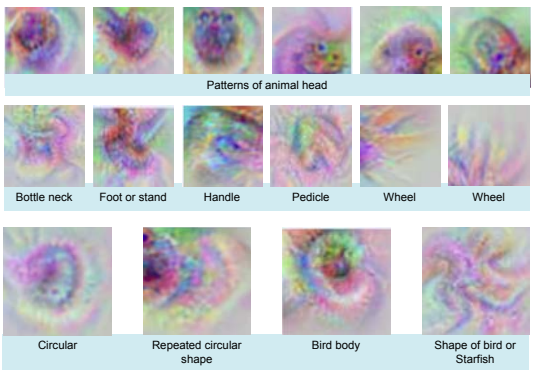


Figure 7. The learned part filters visualized using deepdraw [1].

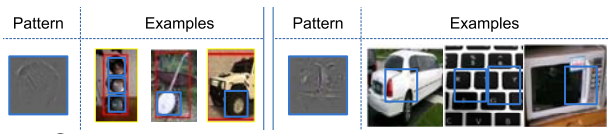


Figure 8. Repeated visual patterns in multiple object classes.

The def-pooling layer has the following advantages.

1) It can replace any pooling layer, and learn deformation of parts with different sizes and semantic meanings. For example, at a higher level, visual patterns can be large parts, e.g. human upper bodies, and the def-pooling layer can capture the deformation constraint of human upper parts. At a middle level, the visual patterns can be smaller parts, e.g. heads. At the lowest level, the visual patterns can be very small, e.g. mouths. A human upper part is composed of a deformable head and other parts. The human head is composed of a deformable mouth and other parts. Object parts at different semantic abstraction levels with different deformation constraints are captured by def-pooling layers at different levels. The composition of object parts is naturally implemented by CNN with hierarchical layers.
2) The def-pooling layer allows for multiple deformable parts with the same visual cue, i.e. multiple response peaks are allowed for one filter. This design is from our observation that an object may have multiple object parts with the same visual pattern. For example, three light bulbs co-exist in a traffic light in Fig. 5.
3) As shown in Fig. 3, the def-pooling layer is a shared representation for multiple classes and therefore the learned visual patterns in the def-pooling layer can be shared among these classes. As examples in Fig. 8, the learned circular visual patterns are shared as different object parts in traffic lights, cars, and ipods.

The layers proposed in [36], [18] does not have these advantages, because they can only be placed after the final convolutional layer, assume one instance per object part, and does not share visual patterns among classes.

3.5 Fine-tuning the deep model with hinge-loss

In RCNN, feature representation is first learned with the softmax loss in the deep model after fine-tuning. Then in a separate step, the learned feature representation is input to a linear binary SVM classifier for detection of each class. In our approach, the softmax loss is replaced by the 200 binary hinge losses when fine-tuning the deep model. Thus the deep model fine-tuning and SVM learning steps in RCNN are merged into one step. The extra training time required for extracting features ($\sim$ 2.4 days with one Titan GPU) is saved.

3.6 Contextual modeling

The deep model learned for the image classification task (Fig. 3 (c)) takes scene information into consideration while the deep model for object detection (Fig. 3 (a) and (b)) focuses on local bounding boxes. The 1000-class image classification scores are used as contextual features, and concatenated with the 200-class object detection scores to form a 1200 dimensional feature vector, based on which a linear SVM is learned to refine the 200-class detection scores. For a specific object class, not all object classes from the image classification model are useful. We learn a two-stage SVM to remove most of the classes. In the first stage, all scores from the 1000 classes are used for learning a linear SVM. At the second stage, the 10 classes with the largest magnitude in the linear SVM

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

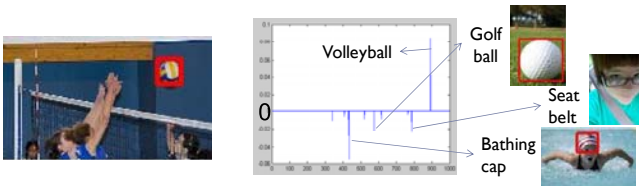


Figure 9. The SVM weights on image classification scores (a) for the object detection class volleyball (b).

weights learned in the first stage are selected as features and then a linear SVM is learned for a given object class to be detected. Therefore, only the classification scores of 10 classes from the image classification deep model are used for each class to be detected. The SVM is explicitly trained but not within the network framework. If 5, 20, 50, 100 or 1000 classes are used, the mAP drops by 0, 0.2%, 0.8%, 0.9% and 4.5% respectively when compared with the result of using 10 classes. This result shows that only a few number of classes are helpful for detection. The heuristic selection of 10 classes helps to remove the effect from uncorrelated classes.

Take object detection for class volleyball as an example in Figure 9. If only considering local regions cropped from bounding boxes, volleyballs are easy to be confused with bathing caps and golf balls. In this case, the contextual information from the whole-image classification scores is helpful, since bathing caps appear in scenes of beach and swimming pools, golf balls appear in grass fields, and volleyballs appear in stadiums. The whole images of the three classes can be better distinguished because of the global scenery information. Fig. 9 plots the learned linear SVM weights on the 1000-class image classification scores. It is observed that image classes bathing cap and golf ball suppress the existence of volleyball in the refinement of detection scores with negative weights, while the image class volleyball enhances the detection score of volleyball.

3.7 Combining models with high diversity

Model averaging has been widely used in object detection. In existing works [75], [25], [21], the same deep architecture was used. Models were different in cropping images at different locations or using different learned parameters. In our model averaging scheme, we learn models under multiple settings. The settings of the models used for model averaging are shown in Table 4. They are different in net structures, pretraining schemes, loss functions for the deep model training, adding def-pooling layer or not. The motivation is that models generated in this way have higher diversity and are complementary to each other in improving the detection results after model averaging. For example, although model no. 4 has low mAP, it is found by greedy search because its pretraining scheme is different from other models and provides complementary scores for the averaged scores.

The 6 models as shown in Table 4 are automatically selected by greedy search on ImageNet Det val_2 from 10 models, and the mAP of model averaging is 50.3% on the test data of ILSVRC2014, while the mAP of the best single model is 47.9%.

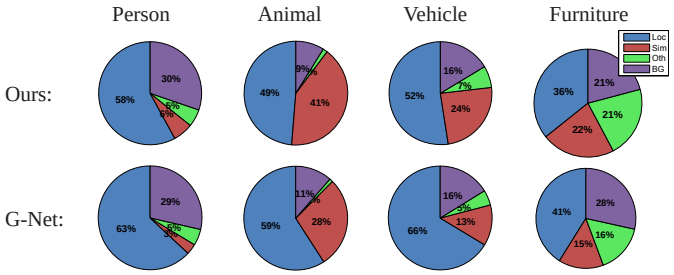


Figure 10. Fraction of high-scored false positives on VOC-2007 that are due to poor localization (Loc), confusion with similar objects (Sim), confusion with other VOC objects (Oth), or confusion with background or unlabeled objects (BG).

4 Experimental Results

Our experimental results are implemented based on the Caffe [24]. Only selective search is used for proposing regions if not specified.

Overall result on PASCAL VOC. For the VOC-2007 detection dataset [11], we follow the approach in [16] for splitting the training and testing data. Table 1 shows the experimental results on VOC-2007 testing data, which include approaches using hand-crafted features [17], [48], [63], [62], [12], deep CNN features [16], [21], [15], [47], [46], [68], and CNN features with deformation learning [18]. Since all the state-of-the-art works reported single-model results on this dataset, we also report the single-model result only. Our model was pretrained on bounding box annotation, with deformation, without context, and with GoogLeNet as the baseline net. Ours outperforms RCNN [16] and SPP [21] by about 5% in mAP. RCNN, SPN and our model are all pre-trained on the ImageNet Cls-Loc training data and fine-tuned on the VOC-2007 training data. Table 2 shows the per-class mAPs for our approach with G-Ntt and RCNN with VGG and GoogleNet [16]. Fig. 10 shows the analysis on false positives using the approach in [23].

Overall result on MS-COCO. Without using context, our single model has mAP 25.6% on the MS-COCO Test-dev dataset [28].

Experimental Setup on ImageNet. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 [49] contains two different datasets: 1) the classification and localization (Cls-Loc) dataset and 2) the detection (Det) dataset. The training data of Cls-Loc contains 1.2 million images with labels of $1,000$ categories. It is used to pretrain deep models. The same split of train and validation data from the Cls-Loc is used for image-level annotation and object-level annotation pretraining. The Det contains 200 object categories and is split into three subsets, train, validation (val), and test data. We follow RCNN [16] in splitting the val data into val_1 and val_2. Val_1 is used to train models, val_2 is used to evaluate separate components, and test is used to evaluating the overall performance. The $\text{val}_1/\text{val}_2$ split is the same as that in [16].

Overall result on ImageNet Det. RCNN [16] is used as the state-of-the-art for comparison. The source code provided by the authors was used and we were able to repeat their results. Table 3 summarizes the results from ILSVRC2014 object detection challenge. It includes the best

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

Table 1
Detection mAP (%) on **VOC-2007 test**. All approaches are trained on VOC-2007 data. Bounding box regression is used in DPM, SPP, RCNN, RCNN-V5, fRCN, and our approach. Only a single model is used for all approaches.

approach	DPM [17]	HSC-DPM [48]	Regionlet [63]	Flair [62]	DP-DPM [18]	SPP [21]	RCNN [16]	RCNN-v5 [16]	fRCN [15]	RPN [47]	YOLO [46]	Superpixel Label [68]	ours
	33.7	34.3	41.7	33.3	45.2	58.5	63.1	66.0	66.9	67.6	59.1	61.4	69.0

Table 2
VOC-2007 test detection average precision (%) for RCNN using VGG and our approach.

	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
RCNN+VGG	73.4	77.0	63.4	45.4	44.6	75.1	78.1	79.8	40.5	73.7	62.2	79.4	78.1	73.1	64.2	35.6	66.8	67.2	70.4	71.1	66.0
RCNN+G-Net	73.7	72.4	65.4	47.2	44.6	71.2	77.4	74.2	42.6	71.1	57.5	72.2	72.7	74.9	62.5	37.8	67.9	66.4	65.3	70.9	64.4
Ours+G-Net	77.1	76.8	75.6	54.5	51.9	76.1	79.5	77.7	48.0	78.2	61.1	82.1	78.1	76.1	65.9	35.4	75.3	67.2	71.7	71.9	69.0

Table 3
Detection mAP (%) on ILSVRC2014 for top ranked approaches with single model (sgl) and average model (avg).

approach	Flair [62]	RCNN [16]	Berkeley Vision [16]	UvA-Euvision [55]	DeepInsight [67]	GoogLeNet [60]	Superpixel Label [68]	ours
ImageNet val_2 (avg)	n/a	n/a	n/a	n/a	42	44.5	45.4	50.7
ImageNet val_2 (sgl)	n/a	31.0	33.4	n/a	40.1	38.8	42.8	48.2
ImageNet test (avg)	22.6	n/a	n/a	n/a	40.5	43.9	45.0	50.3
ImageNet test (sgl)	n/a	31.4	34.5	35.4	40.2	38.0	42.5	47.9

Table 4
Models used for model averaging. The result of mAP is on val_2 without bounding box regression and context. For net design, D-Def(O) denotes our DeepID-Net that uses def-pooling layers using Overfeat as baseline structure, D-Def(G) denotes DeepID-Net that uses def-pooling layers using GoogLeNet as baseline structure, G-net denotes GoogLeNet. For pretraining, 0 denotes the pretraining scheme of RCNN [16], 1 denotes the Scheme 1 in Section 3.3. For loss of net, h denotes hinge loss, s denotes softmax loss. Bounding box rejection is used for all models. Selective search and edgeboxes are used for proposing regions.

model number	1	2	3	4	5	6
net design	D-Def(O)	D-Def(G)	G-net	G-net	D-Def(G)	D-Def(G)
Pretrain	1	1	1	0	1	1
loss of net	h	h	s	s	h	h
mAP (%)	43.3	47.3	45.5	42.1	47.3	44.9

results on the test data submitted to ILSVRC2014 from GoogLeNet [60], DeepInsight[67], UvA-Euvision[55], Berkeley Vision[16], which ranked top among all the teams participating in the challenge. In terms of single-model and model averaging performance, we achieve the highest mAP. It outperforms the winner of ILSVRC2014, GoogleNet, by 6.1% on mAP. Table 4 shows the 6 models we used in model averaging.

4.1 Ablation study

The ImageNet Det is used for ablation study. Bounding box regression is not used if not specified.

4.1.1 Baseline deep model and bounding box rejection

As shown in Fig. 3, a baseline deep model is used in our DeepID-Net. Table 5 shows the results for different baseline deep models and bounding box rejection choices. AlexNet in [25] is denoted as A-net, ZF in [75] is denoted as Z-net, and Overfeat in [52] is denoted as O-net. GoogLeNet in [60] is denoted as G-net. Except for the two components investigated in Table 5, other components are the same as RCNN, while the new training schemes and the new components introduced in Section 3.2 are not included. The configuration in the second column of Table 5 is the same as RCNN (mean mAP 29.9%). Based on RCNN, applying bounding box rejection improves mAP by 1%. Therefore, bounding box rejection not only saves the time for training and validating new models, which is critical for future research, but also improves detection accuracy. Bounding box rejection is not constrained to particular detectors such as RCNN or fast RCNN. The time required to process one image is around 3.5 seconds per image using RCNN and around 0.2 seconds using fast RCNN. Both with bounding box rejection, ZF [75] performs better than AlexNet [25], with 0.9% mAP improvement. Overfeat [52] performs better than ZF, with 4.8% mAP improvement. GoogLeNet [60] performs better than Overfeat, with 1.2% mAP improvement.

Experimental results in Table 5 show the further investigation on the influence of bounding box rejection scheme in training and testing stage. Experimental results on two different CNN architectures, i.e. A-net and Z-net, show that the mAP is similar whether the rejection scheme in the testing stage is used or not. And the rejection scheme in the training stage is the main factor in improving the mAP. If there is concern that the rejection scheme results in lower recall of the candidate windows at the testing stage, the rejection scheme at the testing stage can be skipped. If not specified, bounding box rejection is used in both training and testing stages.

4.1.2 Investigation on the number of object classes at the pretraining stage

In order to investigate the influence from the number of object classes at the pretraining stage, we use the AlextNet and train on the ImageNet classification data without using the bounding box labels. Table 7 shows the experimental results. As pointed out in [50], the 200 classes in ImageNet detection corresponds to 494 classes in ImageNet classification.

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

Table 5
Study of bounding box (bbox) rejection and baseline deep model on ILSVRC2014 val$_2$. Pretrained without bounding box labels. Def-pooling, context and bounding box regression are not used.

bbox rejection?	n	y	y	y	y
deep model	A-net	A-net	Z-net	O-net	G-net
mAP (%)	29.9	30.9	31.8	36.6	37.8
meadian AP (%)	28.9	29.4	30.5	36.7	37

Table 6
Study of bounding box (bbox) rejection at the training and testing stage without context or def-pooling. Pretrained without bounding box labels. Def-pooling, context and bounding box regression are not used.

bbox rejection train?	n	y	y	y	y
bbox rejection test?	n	y	n	y	n
deep model	A-net	A-net	A-net	Z-net	Z-net
mAP (%)	29.9	30.9	30.8	31.8	31.5
meadian AP (%)	28.9	29.4	29.3	30.5	30.4

Therefore, we investigate three pretraining settings: 1) use the corresponding 494-class samples in ImageNet classification training data but train the deep model as a 200-class classification problem; 2) use the corresponding 494-class samples in ImageNet classification training data and train the deep model as a 494-class classification problem; 3) use the 1000-class samples in ImageNet classification training data and train the deep model as a 1000-class classification problem. The same fine-tuning configuration is used for these three pretraining settings. Experimental results show that 494-class pretraining performs better than 200-class pretraining by 3% mAP. 1000-class pretraining performs better than 494-class pretraining by 4.3% mAP. 3000-class pretraining further improves the mAP by 2.4% compared with 1000-class pretraining. For 3000-class pretraining, each sample carries much more information: for an apple image, the 3000-class pretraining provides further information that it is not the other 2999 classes. And the use of more classes makes the training task challenging and not easy to overfit.

4.1.3 Investigation on def-pooling layer

Different deep model structures are investigated and results are shown in Table 8 using the new pretraining scheme in Section 3.3. Our DeepID-Net that uses def-pooling layers as shown in Fig. 3 is denoted as D-Def. Using the Z-net as baseline deep model, the DeepID-Net that uses def-pooling layer in Fig. 3 improves mAP by 2.5%. Def-pooling layer improves mAP by 2.3% for both O-net and G-net. This experiment shows the effectiveness of the def-pooling layer for generic object detection. In our implementation of def-pooling for G-net, we only replace max-pooling by def-pooling but did not add an additional feature maps like that in Fig. 3(b). 2.3% mAP improvement is still observed on G-net by replacing the max-pooling with def-pooling.

Table 7
Study of number of classes used for pretraining. AlexNet is used. Pretrained without bounding box labels. Def-pooling, context and bounding box regression are not used.

number of classes	200	494	1000	3000
mAP (%)	22.6	25.6	29.9	32.3
meadian AP (%)	19.8	23.0	28.9	31.7

Table 8
Investigation on def-pooling for different baseline net structures on ILSVRC2014 val$_2$. Use pretraining scheme 1 but no bounding box regression or context.

net structure	Z-net	D-Def(Z)	O-net	D-Def(O)	G-net	D-Def(G)
mAP (%)	36.0	38.5	39.1	41.4	40.4	42.7
meadian (%)	34.9	37.4	37.9	41.9	39.3	42.3

4.1.4 Investigation on different pretraining schemes and baseline net structures

There are two different annotation levels, image and object. Table 9 shows the results for investigation on annotation levels and net structures. When producing these results, other new components introduced in Section 3.4-3.6 are not included. For pretraining, we drop the learning rate by 10 when the classification accuracy of validation data reaches plateau, until no improvement is found on the validation data. For fine-tuning, we use the same initial learning rate (0.001) and the same number of iterations (20,000) for dropping the learning rate by 10 for all net structures, which is the same setting in RCNN [16].

Pretraining on object-level-annotation performs better than pretraining on image-level annotation by 4.4% mAP for A-net and 4.2% for Z-net. This experiment shows that object-level annotation is better than image-level annotation in pretraining deep model.

4.1.5 Investigation on the overall pipeline

Table 10 summarizes how performance gets improved by adding each component step-by-step into our pipeline. RCNN has mAP 29.9%. With bounding box rejection, mAP is improved by about 1%, denoted by $+1\%$ in Table 10. Based on that, changing A-net to Z-net improves mAP by 0.9%. Changing Z-net to O-net improves mAP by 4.8%. O-net to G-net improves mAP by 1.2%. Replacing image-level annotation by object-level annotation in pretraining, mAP is increased by 2.6%. By combining candidates from selective search and edgeboxes [84], mAP is increased by 2.3%. The def-pooling layer further improves mAP by 2.2%. Pretraining the object-level annotation with multiple scales [4] improves mAP by 2.2%. After adding the contextual information from image classification scores, mAP is increased by 0.5%. Bounding box regression improves mAP by 0.4%. With model averaging, the final result is 50.7%.

4.2 Per-Class Accuracy as a Function of Object Properties on ILSVRC14 Object Detection Data

Inspired by the analysis in [50], we perform analysis on the object properties that influence the variation in object detection accuracy for different classes in this section. Our result with 50.7% mAP on the val2 data is used for analysis.

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

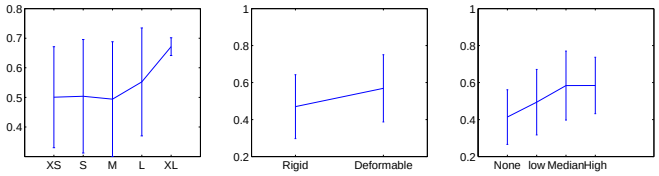


Figure 11. The mean average precision of our best-performing model in the y axis as a function of real-word size (left), deformability (middle), and texture (right) in the x axis.

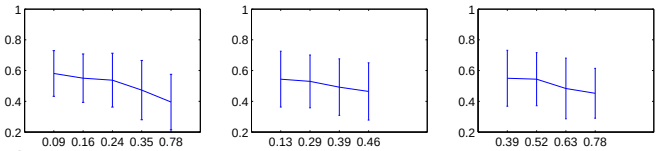


Figure 12. The mean average precision of our best-performing model in the y axis as a function of the variance in aspect ratio (left), part existence (middle) and in-plane, out-plane rotation(right). The x axis denotes the average variance of each group.

For real-world size, deformability, and amount of texture, the following conclusion on the detection accuracy as a function of these object properties can be drawn from the experimental results in Fig. 11:

Real-world size, XS for extra small (e.g. nail), small (e.g. fox), medium (e.g. bookshelf), large (e.g. car) or XL for extra large (e.g. airplane). The object detection model performs similar on extra small, small or medium ones, which is different from the optimistic model in [50]. It performs better on extra large and large objects, with extra large objects having the highest mean AP (close to 70%).

Deformability within instance, Rigid (e.g., mug) or deformable (e.g., snake). Similar to [50], we also find that deformable objects have higher accuracy than rigid objects.

Amount of texture, none (e.g. punching bag), low (e.g. horse), medium (e.g. sheep) or high (e.g. honeycomb). The model is better on objects with at least median level of texture compared to untextured or low textured objects.

The three properties above are investigated in [50] using optimistic model, i.e. directly compare all the entries in the past 3 years to obtain the most optimistic measurement of state-of-the-art accuracy on each category. Using our best performing model, similar conclusion can be drawn.

In the following, we investigate new properties that we found influential to object detection accuracy. Object classes are sorted in ascending order using these properties and then uniformly grouped, i.e. all groups have the number of classes.

Variance in aspect ratio. Aspect ratio is measured by the width of the object bounding box divided by the height of the bounding box. Objects with large variance in aspect ratio, e.g. band aid and nail, are more likely to be slim and have large variation in rotation, which result in the drastic appearance change of the visual information within the object bounding box. Therefore, as shown in Fig. 12, objects with lower variance in aspect ratio performs better.

Variance in part existence. Many objects have some of their parts not existing in the bounding box because of occlusion or tight-shot. The variation in part existence causes the appearance variation for object of the same class. For example, a backpack with only its belt in the bounding box is very different in appearance from a backpack with its bag in the bounding box. We labeled the object parts and their existences for all the 200 classes on the val1 data and use them for obtaining the variance in part existence. As shown in Fig. 12, objects with lower variance in part existence performs better.

Variance in rotation. In-plane and out-of-plane rotation are factors that influence the within-class appearance variation. An ax with frontal view is very different in appearance from an ax with side view. An upright ax is very different in appearance from a horizontal ax. We labeled the rotation of objects for all the 200 classes on the val1 data and use them for obtaining the variance in rotation. As shown in Fig. 12, objects with lower variance in rotation performs better.

Number of objects per image. The number of object per image for the cth object class, denoted by N_c, is obtained as follows:

$$N_c = \frac{1}{P_c} \sum_{p_c=1}^{P_c} (n_{p_c}), \tag{8}$$

where n_{p_c} is the number of objects within the image of the p_cth sample for class c, $p_c = 1 \ldots, P_c$. N_c is obtained from the val1 data. When there are large number objects within an image, they may occlude each other and appear as background for the ground truth bounding box of each other, resulting in the added complexity of object appearance and background clutter. As shown in Fig. 13, some small objects, bee and butterfly, have less than 2 objects per image on average. And they have very high AP, 90.6% for butterfly and 76.9% for bee. We find that the images in val1 with these samples are mostly captured by tight shot, and they have relatively simple background. As shown in Fig. 13, the model performs better when the number of objects per image is smaller.

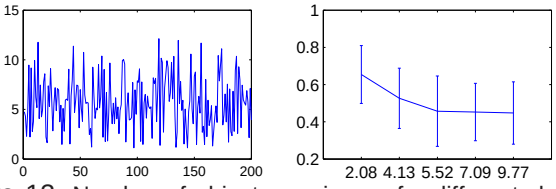


Figure 13. Number of objects per image for different classes (left) and the detection accuracy as a function of the average number of objects per image.

Mean area per bounding box. We measure the size of the bounding box by the area (width multiplied by height) of this bounding box. We did not use the bounding box size over image size in [50] because the image size may have influence on image classification and object localization but should have small influence on object detection, in which bounding box is independently evaluated. As shown in Fig. 14, the average area for different objects varies a lot. Sofa has the largest mean area. Although butterfly and bee are extra small in real-world size, they are large in average areas, 36.8k for bee and 57.7k for butterfly. As shown in Fig. 13, the average AP is higher when the mean area per bounding box is larger.

Recall from region proposal. In our model, selective search and edgeboxes are used for proposing the regions. After

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

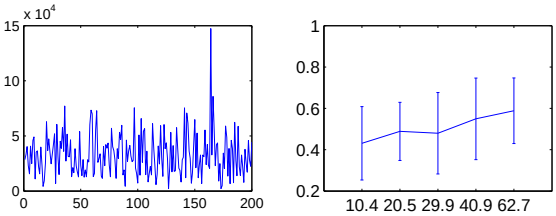


Figure 14. The average size of the bounding box for different classes (left) and the detection accuracy as a function of the average number of objects per image.

bounding box rejection, 302 boxes per image are obtained on val1. The average recall is 89.19% for overlap greater than 0.5 and 78.98% for overlap greater than 0.7.

Fig. 15 shows the 5 classes with lowest and highest average precision and their corresponding factors. The 5 object classes with the lowest accuracy are mostly none-deformable, having low texture, small bounding box size, large number of objects per image, large variation in aspect ratio, part existence and rotation.

We also tried other properties, like variation in bounding box size, average aspect ratio, number of positive samples, but did not find them to have strong correlation to the detection accuracy.

Fig. 16 shows the object classes with large mAP improvement and mAP drop when the def-pooling is used. 140 of the 200 classes have their mAP improved. Def-pooling brings large mAP gains for mammals like squirrel with deformation and instruments like banjo with rotation. However, man-made objects such as waffle iron, digital clock, cocktail shaker and vacuum have inconsistent existence of object parts, large variation in rotation and part appearance. Therefore, the mAP gains are negative for these man-made objects.

Fig. 17 shows the detection accuracy for object classes grouped at different WordNet hierarchical levels. It can be seen that vertebrates that are neither mammal nor fish, i.e. bird, frog, lizard, snake, and turtle, have the largest mAP. Mammals also have large mAP because mammals share similar appearance, have rich texture and have many object classes that help each other in learning their feature representations. Generally, artifacts have lower mAP because they have low texture and large variation in shape. Texts in dashed boxes of Fig. 17 show the absolute mAP gain obtained by bounding box rejection and def-pooling for each group. It can be seen that def-pooling has 9.5% mAP gain for detecting person. Def-pooling has higher mAP gain (4.6%) for mammals with regular deformation and part appearance than substances (0.4%) with irregular deformation and part appearance.

5 Appendix A: Relationship between the deformation layer and the DPM

The quadratic deformation constraint in [12] can be represented as follows:

$$\tilde{m}^{(i,j)} = m^{(i,j)} - a_1(i - b_1 + \frac{a_3}{2a_1})^2 - a_2(j - b_2 + \frac{a_4}{2a_2})^2, \quad (9)$$

where $m^{(i,j)}$ is the (i,j)th element of the part detection map $\mathbf{M}$, (b_1, b_2) is the predefined anchor location of the pth part. They are adjusted by $a_3/2a_1$ and $a_4/2a_2$, which are

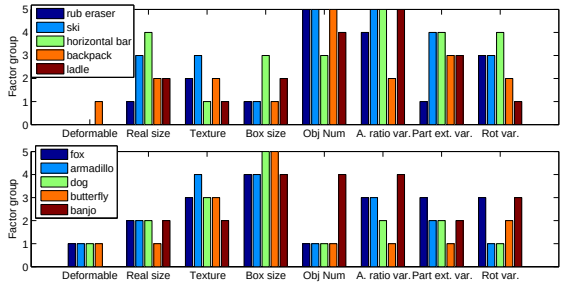


Figure 15. The factors for the 5 object classes with lowest AP (top) and highest AP (bottom). The y axis denotes the group index g for the factors in Fig. 11-14, e.g. deformation ($g = \{0, 1\}$), real-world size ($g = \{1, \ldots, 5\}$), texture ($g = \{1, \ldots, 4\}$), box size ($g = \{1, \ldots, 5\}$). Larger g denotes higher deformation, real size, texture etc. The x axis corresponds to different object classes, e.g. ski, ladle, with different factors, e.g. deformation, real size, texture. Legends denote different object classes.

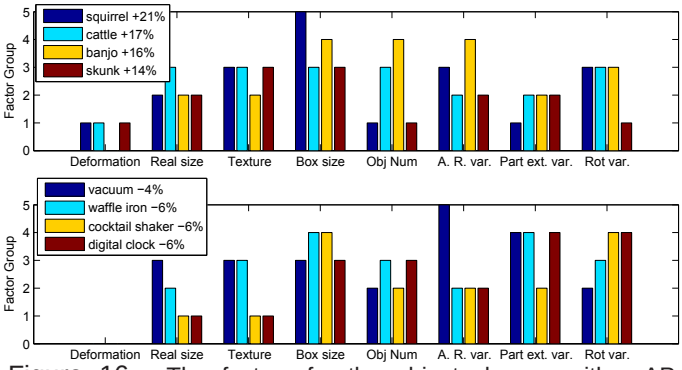


Figure 16. The factors for the object classes with mAP improvement (top) and mAP drop (bottom) introduced by the def-pooling. The meaning of x and y axes are the same as 15. Legends denote different object classes and their mAP change caused by def-pooling.

automatically learned. a_1 and a_2 (9) decide the deformation cost. There is no deformation cost if $a_1 = a_2 = 0$. Parts are not allowed to move if $a_1 = a_2 = \infty$. (b_1, b_2) and $(\frac{a_3}{2a_1}, \frac{a_4}{2a_2})$ jointly decide the center of the part. The quadratic constraint in Eq. (9) can be represented using Eq. (1) as follows:

$$\begin{aligned} \tilde{m}^{(i,j)} &= m^{(i,j)} - a_1 d_1^{(i,j)} - a_2 d_2^{(i,j)} - a_3 d_3^{(i,j)} - a_4 d_4^{(i,j)} - a_5, \\ d_1^{(i,j)} &= (i - b_1)^2, \quad d_2^{(i,j)} = (j - b_2)^2, d_3^{(i,j)} = i - b_1, \\ d_4^{(i,j)} &= j - b_2, a_5 = {a_3}^2/(4a_1) + {a_4}^2/(4a_2). \end{aligned} \quad (10)$$

In this case, a_1, a_2, a_3 and a_4 are parameters to be learned and $d_n^{(i,j)}$ for $n = 1, 2, 3, 4$ are predefined. a_5 is the same in all locations and need not be learned. The final output is:

$$b = \max_{(i,j)} \tilde{m}^{(i,j)}, \quad (11)$$

where $\tilde{m}^{(i,j)}$ is the (i,j)th element of the matrix $\tilde{\mathbf{M}}$ in (9).

6 Conclusion

This paper proposes a deep learning based object detection pipeline, which integrates the key components of bounding box reject, pretraining, deformation handling, context modeling, bounding box regression and model averaging. It significantly advances the state-of-the-art from mAP 31.0%

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

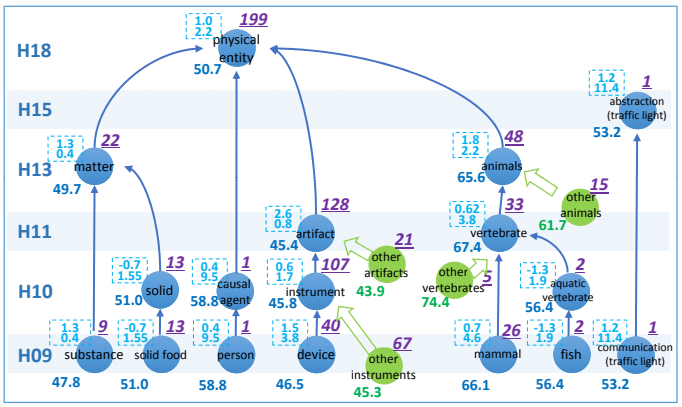


Figure 17. The detection accuracy for object classes grouped at different WordNet hierarchical levels. Tilted text at the upper right of the circle denotes the number of classes within the 200 object classes of the ILSVRC14 detection task for this WordNet synonym set (synset). Texts at the upper left of the circle denote the absolute mAP gain obtained by bounding box rejection and def-pooling. Un-tilted text below the circle denote the mAP in percentage for this WordNet synset. For example, the WordNet synset 'matter' has height 13, 22 object classes and mAP 49.7%, bounding box rejection has mAP gain of 1.34% and the def-pooling has mAP gain of 0.4%. The 'other vertebrates' denotes the object classes that are vertebrates but not mammal or aquatic vertebrate, similarly for 'other artifacts' and 'other instruments'.

(obtained by RCNN) to 50.3% on the ImageNet object task. Its single model and model averaging performances are the best in ILSVC2014. A global view and detailed component-wise experimental analysis under the same setting are provided to help researchers understand the pipeline of deep learning based object detection.

We enrich the deep model by introducing the def-pooling layer, which has great flexibility to incorporate various deformation handling approaches and deep architectures. Motivated by our insights on how to learn feature representations more suitable for the object detection task and with good generalization capability, a pretraining scheme is proposed. By changing the configurations of the proposed detection pipeline, multiple detectors with large diversity are obtained, which leads to more effective model averaging. This work shows the important modules in an object detection pipeline, although each has its own parameter setting set in an ad hoc way. In the future, we will design an end-to-end system that jointly learns these modules.

Acknowledgment: This work is supported by the General Research Fund sponsored by the Research Grants Council of Hong Kong (Project Nos. CUHK14206114, CUHK14205615 and CUHK14207814, CUHK417011,CUHK14203015).

REFERENCES

[1] Deepdraw. DeepDraw on github.com/auduno/deepdraw. 6
[2] O. Barinova, V. Lempitsky, and P. Kohli. On detection of multiple object instances using hough transforms. In *CVPR*, 2010. 3
[3] L. Bourdev and J. Malik. Poselets: body part detectors trained using 3D human pose annotations. In *ICCV*, 2009. 2
[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014. 2, 9
[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 3
[6] C. Desai and D. Ramanan. Detecting actions, poses, and objects with relational phraselets. In *ECCV*, 2012. 3
[7] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, 2009. 3
[8] Y. Ding and J. Xiao. Contextual boost for pedestrian detection. In *CVPR*, 2012. 3
[9] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. In *CVPR*, 2009. 3
[10] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014. 1
[11] M. Everingham, L. V. Gool, C. K. I.Williams, J.Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010. 1, 7
[12] P. Felzenszwalb, R. B. Grishick, D.McAllister, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Trans. PAMI*, 32:1627–1645, 2010. 1, 2, 4, 5, 7, 11
[13] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61:55–79, 2005. 1, 2
[14] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet. Multi-class object localization by combining local contextual interactions. In *CVPR*, 2010. 3
[15] R. Girshick. Fast r-cnn. *ICCV*, 2015. 3, 7, 8
[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1, 2, 3, 4, 7, 8, 9
[17] R. Girshick, P. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://www.cs.berkeley.edu/rbg/latent-v5/. 7, 8
[18] R. Girshick, F. Iandola, T. Darrell, and J. Malik. Deformable part models are convolutional neural networks. *arXiv preprint arXiv:1409.5403*, 2014. 5, 6, 7, 8
[19] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. *arXiv preprint arXiv:1403.1840*, 2014. 3
[20] B. Hariharan, C. L. Zitnick, and P. Dollár. Detecting objects using deformation dictionaries. In *CVPR*, 2014. 2
[21] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*. 2014. 1, 3, 7, 8
[22] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*. 2008. 3
[23] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *Computer Vision–ECCV 2012*, pages 340–353. Springer, 2012. 7
[24] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 7
[25] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 3, 7, 8
[26] K. Lenc and A. Vedaldi. R-cnn minus r. *arXiv preprint arXiv:1506.06981*, 2015. 3
[27] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 1, 3
[28] 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 *ECCV*, pages 740–755. 2014. 7
[29] P. Luo, Y. Tian, X. Wang, and X. Tang. Switchable deep network for pedestrian detection. In *CVPR*, 2014. 3
[30] P. Luo, X. Wang, and X. Tang. Hierarchical face parsing via deep learning. In *CVPR*, 2012. 3
[31] P. Luo, X. Wang, and X. Tang. A deep sum-product architecture for robust facial attributes analysis. In *ICCV*, 2013. 3
[32] P. Luo, X. Wang, and X. Tang. Pedestrian parsing via deep decompositional neural network. In *ICCV*, 2013. 3
[33] W. Ouyang, X. Chu, and X. Wang. Multi-source deep learning for human pose estimation. In *CVPR*, 2014. 3
[34] W. Ouyang, H. Li, X. Zeng, and X. Wang. Learning deep representation with large-scale attributes. In *ICCV*, 2015. 3
[35] W. Ouyang and X. Wang. A discriminative deep model for pedestrian detection with occlusion handling. In *CVPR*, 2012. 3
[36] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *ICCV*, 2013. 3, 5, 6
[37] W. Ouyang and X. Wang. Single-pedestrian detection aided by multi-pedestrian detection. In *CVPR*, 2013. 1

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

Table 9

Ablation study of the two pretraining schemes in Section 3.3 for different net structures on ILSVRC2014 val$_2$. Scheme 0 only uses image-level annotation for pretraining. Scheme 1 uses object-level annotation for pretraining. Def-pooling bounding box regression and context are not used.

net structure	A-net	A-net	A-net	Z-net	Z-net	Z-net	Z-net	O-net	O-net	G-net	G-net
class number	1000	1000	1000	1000	200	1000	1000	1000	1000	1000	1000
bbox rejection	n	n	y	y	n	n	y	y	y	y	y
pretrain scheme	0	1	1	0	1	1	1	0	1	0	1
mAP (%)	29.9	34.3	34.9	31.8	29.9	35.6	36.0	36.6	39.1	37.8	40.4
meadian AP (%)	28.9	33.4	34.4	30.5	29.7	34.0	34.9	36.7	37.9	37.0	39.3

Table 10

Ablation study of the overall pipeline for single model on ILSVRC2014 val2. It shows the mean AP after adding each key component step-by-step.

detection pipeline	RCNN	+bbox rejection	A-net to Z-net	Z-net to O-net	O-net to G-net	image to bbox pretrain	+edgbox candidate	+Def pooling	+multi-scale pretrain	+context	+bbox regression
mAP (%)	29.9	30.9	31.8	36.6	37.8	40.4	42.7	44.9	47.3	47.8	48.2
meadian AP (%)	28.9	29.4	30.5	36.7	37.0	39.3	42.3	45.2	47.8	48.1	49.8
mAP improvement (%)		+1	+0.9	+4.8	+1.2	+2.6	+2.3	+2.2	+2.4	+0.5	+0.4

[38] W. Ouyang, X. Wang, X. Zeng, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, C.-C. Loy, et al. Deepid-net: Deformable deep convolutional neural networks for object detection. In *CVPR*, 2015. 2

[39] W. Ouyang, X. Wang, C. Zhang, and X. Yang. Factors in finetuning deep model for object detection. In *CVPR*, 2016. 3

[40] W. Ouyang, X. Zeng, and X. Wang. Modeling mutual visibility relationship in pedestrian detection. In *CVPR*, 2013. 3

[41] W. Ouyang, X. Zeng, and X. Wang. Single-pedestrian detection aided by 2-pedestrian detection. *IEEE Trans. PAMI*, 2015. 2

[42] W. Ouyang, X. Zeng, and X. Wang. Learning mutual visibility relationship for pedestrian detection with a deep model. *IJCV*, pages 1–14, 2016. 3

[43] W. Ouyang, X. Zeng, and X. Wang. Partial occlusion handling in pedestrian detection with a deep model. *IEEE Trans. Circuits Syst. Video Technol.*, 2016. 3

[44] D. Park, D. Ramanan, and C. Fowlkes. Multiresolution models for object detection. In *ECCV*, 2010. 3

[45] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. *arXiv preprint arXiv:1403.6382*, 2014. 1, 3

[46] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2015. 3, 7, 8

[47] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015. 3, 7, 8

[48] X. Ren and D. Ramanan. Histograms of sparse codes for object detection. In *CVPR*, 2013. 7, 8

[49] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2014. 7

[50] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 1, 8, 9, 10

[51] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, 2011. 3

[52] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 2, 3, 8

[53] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR*, 2014. 3

[54] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[55] A. Smeulders, T. Gevers, N. Sebe, and C. Snoek. Segmentation as selective search for object recognition. In *ICCV*, 2011. 3, 8

[56] Z. Song, Q. Chen, Z. Huang, Y. Hua, and S. Yan. Contextualizing object detection and classification. In *CVPR*, 2011. 3

[57] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *CVPR*, 2013. 3

[58] Y. Sun, X. Wang, and X. Tang. Hybrid deep learning for computing face similarities. In *ICCV*, 2013. 3

[59] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, 2014. 3

[60] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 1, 2, 4, 8

[61] S. Tang, M. Andriluka, A. Milan, K. Schindler, S. Roth, and B. Schiele. Learning people detectors for tracking in crowded scenes. *ICCV*, 2013. 3

[62] K. E. A. van de Sande, C. G. M. Snoek, and A. W. M. Smeulders. Fisher and vlad with flair. In *CVPR*, 2014. 7, 8

[63] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013. 7, 8

[64] B. Wu and R. Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *IJCV*, 75(2):247–266, 2007. 3

[65] J. Yan, Z. Lei, L. Wen, and S. Z. Li. The fastest deformable part model for object detection. In *CVPR*, 2014. 2

[66] J. Yan, Z. Lei, D. Yi, and S. Z. Li. Multi-pedestrian detection in crowded scenes: A global view. In *CVPR*, 2012. 3

[67] J. Yan, N. Wang, Y. Yu, S. Li, and D.-Y. Yeung. Deeper vision and deep insight solutions. In *ECCV workshop on ILSVRC2014*, 2014. 8

[68] J. Yan, Y. Yu, X. Zhu, Z. Lei, and S. Z. Li. Object detection by labeling superpixels. In *CVPR*, 2015. 3, 7, 8

[69] B. Yang, J. Yan, Z. Lei, and S. Z. Li. Convolutional channel features. In *ICCV*, pages 82–90, 2015. 3

[70] W. Yang, W. Ouyang, H. Li, and X. Wang. End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. CVPR, 2016. 2

[71] Y. Yang, S. Baker, A. Kannan, and D. Ramanan. Recognizing proxemics in personal photos. In *CVPR*, 2012. 3

[72] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011. 2

[73] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Trans. PAMI*, 35(12):2878–2890, 2013. 1

[74] B. Yao and L. Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *CVPR*, 2010. 3

[75] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional neural networks. *arXiv preprint arXiv:1311.2901*, 2013. 2, 3, 4, 7, 8

[76] X. Zeng, W. Ouyang, M. Wang, and X. Wang. Deep learning of scene-specific classifier for pedestrian detection. In *ECCV*. 2014. 3

[77] X. Zeng, W. Ouyang, and X. Wang. Multi-stage contextual deep learning for pedestrian detection. In *ICCV*, 2013. 3

[78] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *ECCV*. 2014. 1, 4

[79] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee. Improving object detection with deep convolutional networks via bayesian optimization and structured prediction. In *CVPR*, 2015. 3

[80] R. Zhao, W. Ouyang, H. Li, and X. Wang. Saliency detection by multi-context deep learning. In *CVPR*, 2015. 3

[81] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. *arXiv:1412.6856*, abs/1412.6856, 2014. 1

[82] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*,

0162-8828 (c) 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2016.2587642, IEEE Transactions on Pattern Analysis and Machine Intelligence

2014. 1
[83] L. Zhu, Y. Chen, A. Yuille, and W. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010. 1, 2
[84] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 3, 9
[85] W. Y. Zou, X. Wang, M. Sun, and Y. Lin. Generic object detection with dense neural patterns and regionlets. *BMVC*, 2014. 3

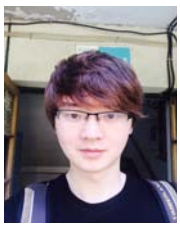
Wanli Ouyang received the PhD degree in the Department of Electronic Engineering, The Chinese University of Hong Kong, where he is now a Research Assistant Professor. His research interests include image processing, computer vision and pattern recognition.

Xingyu Zeng received the B.S. degree in Electronic Engineering and Information Science from University of Science and Technology of China in 2011. He is currently a PHD student in the Department of Electronic Engineering at the Chinese University of Hong Kong. His research interests include computer vision and deep learning.

Xiaogang Wang received the PhD degree from the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology in 2009. He is currently an Assistant Professor in the Department of Electronic Engineering at The Chinese University of Hong Kong. His research interests include computer vision and machine learning.

Shi Qiu received the BS and PhD degree from Tsinghua University (2009) and the Chinese University of Hong Kong (2014), respectively. He is currently a research scientist at SenseTime Group Limited. His research interests include object recognition, object detection, and image search. He worked as a postdoctoral fellow at the Chinese University of Hong Kong from 2014 to 2015.

Ping Luo is a postdoctoral researcher at the Department of Information Engineering, The Chinese University of Hong Kong, where he received his Ph.D. in 2014. His research interests include deep learning, computer vision, and computer graphics, focusing on face analysis, pedestrian analysis, and large-scale object recognition and detection.



Yonglong Tian is currently an M.Phil student at Department of Information Eingeering in The Chinese University of Hong Kong. His research interests include computer vision and machine learning, and object detection.

Hongsheng Li received the masters and doctorate degrees in computer science from Lehigh University, Pennsylvania, in 2010 and 2012, respectively. He is an associate professor in the School of Electronic Engineering at University of Electronic Science and Technology of China. His research interests include computer vision, medical image analysis, and machine learning.

Shuo Yang received the B.E. degree in Software Engineering from Wuhan University, China, in 2013. He is currently working toward the PhD degree in the Department of Information Engineering at the Chinese University of Hong Kong. His research interests include computer vision and machine learning, in particular, face detection and object recognition.

Zhe Wang received the BEng in Optical Engineering from Zhejiang University, China, in 2012. He is currently working toward a PhD degree in the Department of Electronic Engineering, the Chinese University of Hong Kong. His research interest is focused on compressed sensing, magnetic resonance imaging, image segmentation and object detection.

Hongyang Li received his B.E. degree in major Electronic and Information Engineering at Dalian University of Technology (DUT), 2014. From 2013 to 2014, he was a research assistant in the computer vision group at DUT. He is currently a Ph.D. candidate under supervision of Prof. Xiaogang Wang at The Chinese University of Hong Kong. His research interests focus on deep learning and object recognition.

Junjie Yan received his PhD degree in 2015 from National Laboratory of Pattern Recognition, Chinese Academy of Sciences. Since 2016, he is the principle scientist in SenseTime Group Limited and leads the research in object detection and tracking, and applications in video surveillance.

Kun Wang received the B.S. degree in Software Engineering from Beijing Jiaotong University in 2014. He is currently a research assistant in the Department of Electronic Engineering at the Chinese University of Hong Kong. His research interests include crowd analysis and object detection with deep learning.

Chen Change Loy received the PhD degree in Computer Science from the Queen Mary University of London in 2010. He is currently a Research Assistant Professor in the Department of Information Engineering, Chinese University of Hong Kong. Previously he was a postdoctoral researcher at Vision Semantics Ltd. His research interests include computer vision and pattern recognition, with focus on face analysis, deep learning, and visual surveillance.

Xiaoou Tang received the B.S. degree from the University of Science and Technology of China, Hefei, in 1990, and the M.S. degree from the University of Rochester, Rochester, NY, in 1991. He received the Ph.D. degree from the Massachusetts Institute of Technology, Cambridge, in 1996. He is a Professor in the Department of Information Engineering and Associate Dean (Research) of the Faculty of Engineering of the Chinese University of Hong Kong. He worked as the group manager of the Visual Computing Group at the Microsoft Research Asia from 2005 to 2008. His research interests include computer vision, pattern recognition, and video processing. Dr. Tang received the Best Paper Award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2009. He is a program chair of the IEEE International Conference on Computer Vision (ICCV) 2009 and has served as an Associate Editor of IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) and International Journal of Computer Vision (IJCV). He is a Fellow of IEEE.

0162-8828 (c) [redacted] is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.