DeepID-Net: Deformable Deep Convolutional Neural Networks for Object Detection

Wanli Ouyang, Xiaogang Wang, Xingyu Zeng, Shi Qiu, Ping Luo, Yonglong Tian,
Hongsheng Li, Shuo Yang, Zhe Wang, Chen-Change Loy, Xiaoou Tang
The Chinese University of Hong Kong
wlouyang, xgwang@ee.cuhk.edu.hk

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 [14], 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 provide a global view for people to understand the deep learning object detection pipeline.

1. Introduction

Object detection is one of the fundamental challenges in computer vision. It has attracted a great deal of research interest [5, 39, 11, 19]. 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 object detection and recognition systems [44, 38, 19, 22]. 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 [44, 38, 22]. 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 targeting object detection task is a practical solution [10, 32, 57, 14]. 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 with

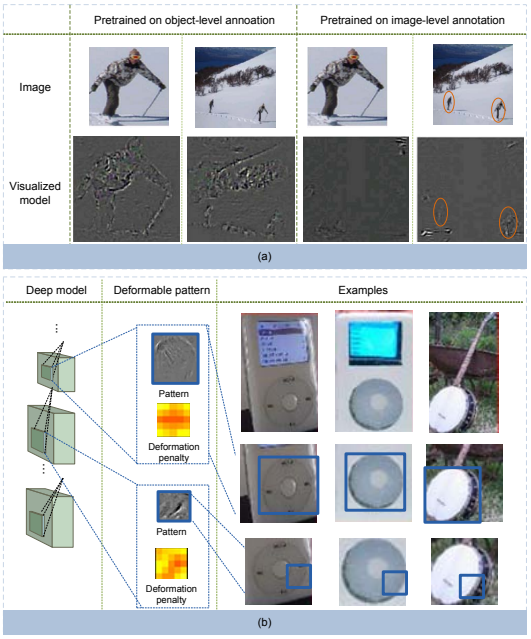

Figure 1. The motivation of this paper in new pretraining scheme (a) and jointly learning feature representation and deformable object parts shared by multiple object classes at different semantic levels (b). In (a), a model pretrained on image-level annotation is more robust to size and location change 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. *Best viewed in color.*

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 on 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 generic object detection [11, 59, 12, 52]. 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 the shared visual patterns and their deformation properties for multiple object classes at different semantic levels and composition levels.

The performance of deep learning object detection 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 devil of details [4, 14], this paper compares the performance of recent deep models, including AlexNet [21], ZF [54], Overfeat [36], and GoogleNet [44] under the same setting for different pretraining-finetuining schemes.

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 [14] framework. This paper also provides detailed component-wise experimental results on how our approach can improve the mean Averaged Precision (AP) obtained by RCNN [14] 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 [14, 44]. Then the deep model is fine-tuned on the ImageNet/PASCAL-VOC object detection dataset with 200/20 classes, which are the targeting 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.

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 [11, 59] for handling translational movement of parts. To handle more complex articulations, size change and rotation of parts were modeled in [12], and mixture of part appearance and articulation types were modeled in [3, 51]. A dictionary of shared deformable patterns was learned in [18]. In these approaches, features are manually designed.

Because of the power on learning feature representation, deep models have been widely used for object recognition, detection and other vision tasks [36, 54, 19, 37, 61, 17, 22, 14, 28, 30, 55, 56, 27, 23, 41, 43, 41, 42, 25, 26, 24, 58]. In

existing deep CNN models, max pooling and average pooling are useful in handling deformation but cannot learn the deformation penalty and geometric models of object parts. The deformation layer was first proposed in [29] for pedestrian detection. In this paper, we extend it to general object detection on ImageNet. In [29], the deformation layer was constrained to be placed after the last convolutional layer, while 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 [29], it was assumed that a pedestrian only has one instance of a body part, so each part filter only has 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.

Context gains attentions in object detection. The context information investigated in literature includes regions surrounding objects [5, 8, 13], object-scene interaction [9, 20], and the presence, location, orientation and size relationship among objects [2, 48, 49, 7, 31, 13, 40, 9, 53, 8, 50, 30, 6, 35, 45]. 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 are also other important components in the object detection pipeline, such as pretraining [14], network structures [36, 54, 21], refinement of bounding box locations [14], and model averaging [54, 21, 19]. 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.

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 [39] is used to propose candidate bounding boxes.

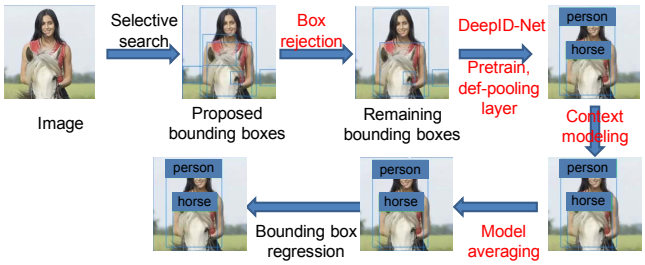

Figure 2. Overview of our approach. Find detailed description in the text of Section 3.1. Texts in red highlight the steps that are not present in RCNN [14].

2. An existing detector, RCNN [14] 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 [14] 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 [54] 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.

(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.

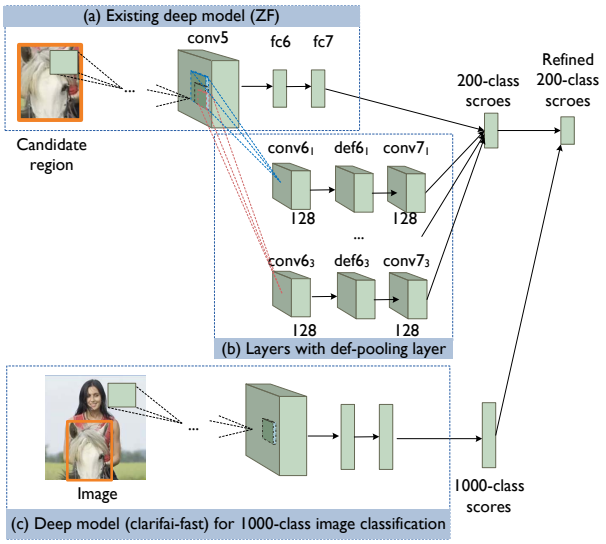

Figure 3. Architecture of DeepID-Net with three parts: (a) baseline deep model, which is ZF [54] in our best-performing single-model detector; (b) layers of part filters with variable sizes and def-pooling layers; (c) deep model to obtain 1000-class image classification scores. The 1000-class image classification scores are used to refine the 200-class bounding box classification scores.

3.3. New pretraining strategy

The widely used training scheme in deep learning based object detection [14, 57, 44] 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 pretain 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 targeting classes (instead of 1,000 classes) from the ImageNet Cls-Loc train data are selected for pre-training in Step (1), the mean AP on ImageNet Det val_2 drops by 5.7%.

3.4. Def-pooling layer

In the deformable part based model (DPM) [11] for object detection, part templates learned on HOG features are considered as part filters and they are convolved with input images. Similarly, we can consider the input of a convolutional layer in CNN as features and consider the filters of that convolutional layer as part filters. And the outputs of the convolutional layer are part detection maps.

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 part detection maps of size $W \times H$. Denote $\mathbf{M}_c$ as the cth part detection map. Denote the (i,j)th element of $\mathbf{M}_c$ by $m_c^{(i,j)}$. The def-pooling layer takes a small block with center $(s_x \cdot x, s_y \cdot y)$ and size $(2R+1) \times (2R+1)$ from the $\mathbf{M}_c$ and produce the element of the output as follows:

$$b_c^{(x,y)} = \max_{\delta_x,\delta_y \in \{-R,\cdots,R\}} \{m_c^{\mathbf{z}_{\delta_x,\delta_y}} - \sum_{n=1}^{N} a_{c,n} d_{c,n}^{\delta_x,\delta_y}\}, \quad \text{where } \mathbf{z}_{\delta_x,\delta_y} = (s_x \cdot x + \delta_x, s_y \cdot y + \delta_y). \tag{1}$$

- $b_c^{(x,y)}$ is the (x,y)th element of the output of the def-pooling layer. 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 max responses are allowed for each part filer.
- $m_c^{\mathbf{z}_{\delta_x,\delta_y}}$ is the visual score of placing the cth part at the deformed position $\mathbf{z}_{\delta_x,\delta_y}$.
- $a_{c,n}$ and $d_{c,n}^{\delta_x,\delta_y}$ are parameters learned by BP. $\sum_{n=1}^{N} a_{c,n} d_{c,n}^{\delta_x,\delta_y}$ is the penalty of placing the part from the assumed anchor position $(s_x \cdot x, s_y \cdot y)$ to the deformed position $\mathbf{z}_{\delta_x,\delta_y}$.

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 $|\delta_x|, |\delta_y| \leq k$ 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

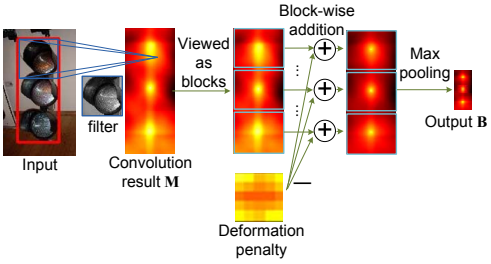

Figure 4. Def-pooling layer. The part detection map and the deformation penalty are summed up. Block-wise max pooling is then performed on the summed map to obtain the output $\mathbf{B}$ of size $\frac{H}{s_y} \times \frac{W}{s_x}$ (3×1 in this example).

provides the ability to learn the map that implicitly decides the kernel size for max-pooling.

Example 2. The deformation layer in [29] 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 $\text{conv}_{l-1,i}$ 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 1. To implement quadratic deformation penalty used in [11], 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) [11] and the DP-DPM [16].

Example 3. 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 assumed location $(s_x \cdot x, s_y \cdot y)$ 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 $(s_x \cdot x, s_y \cdot y)$, 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. 5 shows some learned deformation parameters $d_1^{\delta_x,\delta_y}$.

Take Example 2 as an example for BP learning. $a_{c,n}$ is the parameter in this layer and d_* is pre-defined constant. $\partial b_c^{(x,y)} / \partial a_{c,n} = -d_{c,n}^{(\delta_x,\delta_y)}$ for the position (δ_x, δ_y) with maximum value in (1). The gradients for the parameters in the layers before the def-pooling layer are back-propagated like max-pooling layer.

In our implementation, there are no fully connected layers after $\text{conv7}_{1,2,3}$ in Fig. 3 and Example 3 is used for def-pooling.

3.4.1 Analysis

A visual pattern has different spatial distributions in different object classes. For example, traffic lights and ipods have

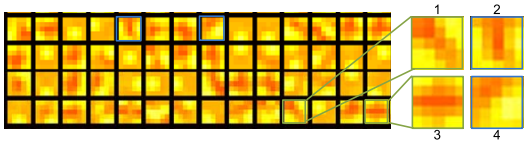

Figure 5. The learned deformation penalty for different visual patterns. The penalties in map 1 are low at diagonal positions. The penalties in map 2 and 3 are low at vertical and horizontal locations separately. The penalties in map 4 are high at the bottom right corner and low at the upper left corner.

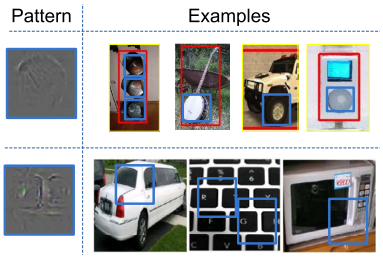

Figure 6. Repeated visual patterns in multiple object classes.

geometric constraints on the circular visual pattern in Fig. 6. 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 is modeled by conv7_1.

The def-pooling layer has the following advantages.

1. It can replace any convolutional 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. 4.

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. 6, the learned circular visual patterns are shared as different object parts in traffic lights, cars, and ipods.

The layers proposed in [29, 16] 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.

Take object detection for class volleyball as an example in Figure 7. 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. 7 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 [54, 21, 19], 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 [1]. They are different in net structures, pretraining schemes, loss functions for the deep model training, adding def-pooling layer or not, and doing bounding box rejection or not. Models generated in this way have higher diversity and are complementary to each other in improving the detection results.

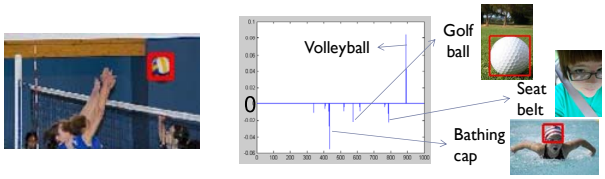

Figure 7. The SVM weights on image classification scores (a) for the object detection class volleyball (b).

The 4 models are automatically selected by greedy search on ImageNet Det val_2, and the mAP of model averaging is 50.3% on the test data of ILSVRC2014, while the mAP of the best single model is 48.2%.

4. Experimental results

Overall result on PASCAL VOC. For the VOC-2007 detection dataset, we follow the approach in [14] for splitting the training and testing data. Table 2 shows the experimental results on VOC-2007 testing data, which include approaches using hand-crafted features [15, 33, 47, 46, 11], deep CNN features [14, 19], and CNN features with deformation learning [16]. 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 ZF as the baseline net. Ours outperforms RCNN [14] and SPP [19] 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.

Experimental Setup on ImageNet. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 [34] 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 [14] 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 val_1/val_2 split is the same as that in [14].

Table 1. Detection mAP (%) on ILSVRC2014 for top ranked approaches with single model (sgl) and average model (avg).

approach	Flair [46]	RCNN[14]	Berkeley Vision	UvA-Euvision	DeepInsight	GoogLeNet[44]	ours
ImageNet val_2 (avg)	n/a	n/a	n/a	n/a	42	44.5	50.7
ImageNet val_2 (sgl)	n/a	31.0	33.4	n/a	40.1	38.8	48.2
ImageNet test (avg)	22.6	n/a	n/a	n/a	40.5	43.9	50.3
ImageNet test (sgl)	n/a	31.4	34.5	35.4	40.2	38.0	47.9

Table 2. Detection mAP (%) on PASCAL VOC-2007 test set.

HOG-DPM [15]	HSC-DPM [33]	Regionlet [47]	Flair [46]	DP-DPM [16]	RCNN[14]	SPP [19]	ours (single model)
33.7	34.3	41.7	33.3	45.2	58.5	59.2	64.1

Table 3. Study of bounding box (bbox) rejection and baseline deep model on ILSVRC2014 val_2 without context or def-pooling.

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 5. Investigation on baseline net structures with dep-pooling on ILSVRC2014 val_2. Use pretraining scheme 1 but no 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

Overall result on ImageNet Det. RCNN [14] is used as the state-of-the-art for comparison. The source code provided by the authors was used to and we were able to repeat their results. Table 1 summarizes the results from ILSVRC2014 object detection challenge. It includes the best results on the test data submitted to ILSVRC2014 from GoogLeNet [44], DeepInsignt, UvA-Euvision, and Berkeley Vision, 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.

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 3 shows the results for different baseline deep models and bounding box rejection choices. AlexNet in [21] is denoted as A-net, ZF in [54] is denoted as Z-net, and overfeat in [36] is denoted as O-net. Except for the two components investigated in Table 3, 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 3 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. Both with bounding box rejection, ZF [54] performs better than AlexNet [21], with 0.9% mAP improvement. Overfeat [36] performs better than ZF, with 4.8% mAP improvement. GoogleNet [44] performs better than Overfeat, with 1.2% mAP improvement.

4.1.2 Investigation on different pretraining schemes and baseline net structures

There are two different sets of data used for pretraining the baseline deep model: 1) the ImageNet Cls train data with 1000 classes and 2) the ImageNet Cls train data data with the same 200 classes as Det. There are two different annotation levels, image and object. Table 4 show the results for investigation on image class number, 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 improvment 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 [14].

Using object-level annotation, pretraining on 1000 classes performs better than pretraining on 200 classes by 5.7% mAP. Using the same 1000 classes, 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. Pretraining with more classes improves the generalization capability of the learned feature representations.

4.1.3 Investigation on def-pooling layer

Different deep model structures are investigated and results are shown in Table 5 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 moel, 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

Table 4. 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. Context is 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 6. 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

experiment shows the effectiveness of the def-pooling layer for generic object detection.

4.1.4 Investigation on the overall pipeline

Table 6 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 6. 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 [60], 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%.

5. Appedix A: Relationship between the deformation layer and the DPM

The quadratic deformation constraint in [11] can be represented as follows:

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

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 automatically learned. a_1 and a_2 (2) 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. (2) can be represented using Eq. (1) as follows:

$$\begin{aligned} &\tilde{m}^{(i,j)} = m^{(i,j)} - a_1 d_1^{(i,j)} - a_2 d_2^{(i,j)} - a_3 d_3^{(i,j)} - a_4 d_4^{(i,j)} - a_5, \\ &d_1^{(i,j)} = (i-b_1)^2, \ d_2^{(i,j)} = (j-b_2)^2, d_3^{(i,j)} = i - b_1, \\ &d_4^{(i,j)} = j - b_2, a_5 = {a_3}^2/(4a_1) + {a_4}^2/(4a_2). \end{aligned} \quad (3)$$

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:

$$b = \max_{(i,j)} \tilde{m}^{(i,j)}, \quad (4)$$

where $\tilde{m}^{(i,j)}$ is the (i,j)th element of the matrix $\tilde{\mathbf{M}}$ in (2).

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% (obtained by RCNN) to 50.3% on the ImgeNet 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.

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

References

[1] www.ee.cuhk.edu.hk/∼wlouyang/projects/imagenetdeepid. 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, 8

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 1, 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*, pages 647–655, 2014. 1

[11] 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, 6, 8

[12] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61:55–79, 2005. 2

[13] C. Galleguillos, B. McFee, S. Belongie, and G. Lanckriet. Multi-class object localization by combining local contextual interactions. In *CVPR*, pages 113–120. IEEE, 2010. 3

[14] 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, 6, 7

[15] R. Girshick, P. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://www.cs.berkeley.edu/rbg/latent-v5/. 6, 7

[16] 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

[17] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. *arXiv preprint arXiv:1403.1840*, 2014. 2

[18] B. Hariharan, C. L. Zitnick, and P. Dollár. Detecting objects using deformation dictionaries. In *CVPR*, 2014. 2

[19] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*. 2014. 1, 2, 3, 6, 7

[20] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *ECCV*, pages 30–43. 2008. 3

[21] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 3, 6, 7

[22] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. 1, 2

[23] P. Luo, Y. Tian, X. Wang, and X. Tang. Switchable deep network for pedestrian detection. In *CVPR*, 2014. 2

[24] P. Luo, X. Wang, and X. Tang. Hierarchical face parsing via deep learning. In *CVPR*, 2012. 2

[25] P. Luo, X. Wang, and X. Tang. A deep sum-product architecture for robust facial attributes analysis. In *ICCV*, 2013. 2

[26] P. Luo, X. Wang, and X. Tang. Pedestrian parsing via deep decompositional neural network. In *ICCV*, 2013. 2

[27] W. Ouyang, X. Chu, and X. Wang. Multi-source deep learning for human pose estimation. In *CVPR*, pages 2337–2344. IEEE, 2014. 2

[28] W. Ouyang and X. Wang. A discriminative deep model for pedestrian detection with occlusion handling. In *CVPR*, 2012. 2

[29] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *ICCV*, 2013. 3, 5, 6

[30] W. Ouyang, X. Zeng, and X. Wang. Modeling mutual visibility relationship in pedestrian detection. In *CVPR*, 2013. 2, 3

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

[32] 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

[33] X. Ren and D. Ramanan. Histograms of sparse codes for object detection. In *CVPR*, pages 3246–3253. IEEE, 2013. 6, 7

[34] 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. 6

[35] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, pages 1745–1752. IEEE, 2011. 3

[36] 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, 7

[37] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR*, 2014. 2

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

[39] A. Smeulders, T. Gevers, N. Sebe, and C. Snoek. Segmentation as selective search for object recognition. In *ICCV*, 2011. 1, 3

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

[41] Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *CVPR*, 2013. 2

[42] Y. Sun, X. Wang, and X. Tang. Hybrid deep learning for computing face similarities. In *ICCV*, 2013. 2

[43] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *CVPR*, 2014. 2

[44] 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, 7

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

[46] K. E. A. van de Sande, C. G. M. Snoek, and A. W. M. Smeulders. Fisher and vlad with flair. In *CVPR*, 2014. 6, 7

[47] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, pages 17–24. IEEE, 2013. 6, 7

[48] 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

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

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

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

[52] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Trans. PAMI*, 35(12):2878–2890, 2013. 2

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

[54] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional neural networks. *arXiv preprint arXiv:1311.2901*, 2013. 2, 3, 4, 6, 7

[55] X. Zeng, W. Ouyang, M. Wang, and X. Wang. Deep learning of scene-specific classifier for pedestrian detection. In *ECCV*, pages 472–487. 2014. 2

[56] X. Zeng, W. Ouyang, and X. Wang. Multi-stage contextual deep learning for pedestrian detection. In *ICCV*, 2013. 2

[57] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *ECCV*, pages 834–849. 2014. 1, 4

[58] R. Zhao, W. Ouyang, H. Li, and X. Wang. Saliency detection by multi-context deep learning. In *CVPR*, 2015. 2

[59] L. Zhu, Y. Chen, A. Yuille, and W. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010. 2

[60] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 8

[61] W. Y. Zou, X. Wang, M. Sun, and Y. Lin. Generic object detection with dense neural patterns and regionlets. *BMVC*, 2014. 2