Noname manuscript No.
(will be inserted by the editor)

End-to-End Learning of Latent Deformable Part-based Representations for Object Detection

Taylor Mordan · Nicolas Thome · Gilles Henaff · Matthieu Cord

Received: date / Accepted: date

Abstract Object detection methods usually represent objects through rectangular bounding boxes from which they extract features, regardless of their actual shapes. In this paper, we apply deformations to regions in order to learn representations better fitted to objects. We introduce DP-FCN, a deep model implementing this idea by learning to align parts to discriminative elements of objects in a latent way, *i.e.* without part annotation. This approach has two main assets: it builds invariance to local transformations, thus improving recognition, and brings geometric information to describe objects more finely, leading to a more accurate localization. We further develop both features in a new model named DP-FCN2.0 by explicitly learning interactions between parts. Alignment is done with an in-network joint optimization of all parts based on a CRF with custom potentials, and deformations are influencing localization through a bilinear product. We validate our models on PASCAL VOC and MS COCO datasets and show significant gains. DP-FCN2.0 achieves state-of-the-art results of 83.3% and 81.2% on VOC 2007 and 2012 with VOC data only.

Keywords Object Detection · Fully Convolutional Network · Deep Learning · Part-based Representation · End-to-End Latent Part Learning

Taylor Mordan · Matthieu Cord
Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6
F-75005 Paris, France
E-mail: taylor.mordan@lip6.fr, matthieu.cord@lip6.fr

Taylor Mordan · Gilles Henaff
Thales Land and Air Systems
2 Avenue Gay-Lussac, 78990 Élancourt, France
E-mail: gilles.henaff@fr.thalesgroup.com

Nicolas Thome
CEDRIC, Conservatoire National des Arts et Métiers
292 Rue St Martin, 75003 Paris, France
E-mail: nicolas.thome@cnam.fr

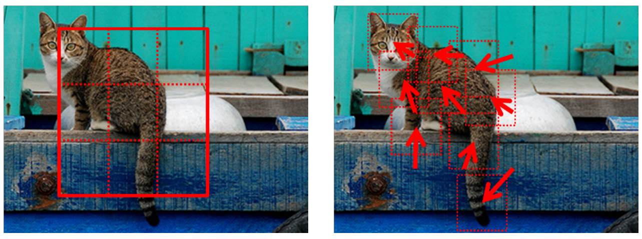
(a) Original region (b) Deformed region

Fig. 1: **Illustration of deformations.** Regions are divided into regular grids (a) and all cells are moved from their initial positions to adapt to the shape of the object and better describe it (b), improving both recognition and localization.

1 Introduction

Recent years have witnessed a great success of Deep Learning with deep Convolutional Networks (ConvNets) (LeCun et al, 1989; Krizhevsky et al, 2012) in several visual tasks. Originally mainly used for image classification (Krizhevsky et al, 2012; Simonyan and Zisserman, 2015; He et al, 2016), they are now widely used for others tasks such as object detection (Girshick et al, 2014; Girshick, 2015; Dai et al, 2016b; Zagoruyko et al, 2016; Lin et al, 2017a) or semantic segmentation (Long et al, 2015; Chen et al, 2015; Li et al, 2017). In particular for detection, region-based deep ConvNets (Girshick et al, 2014; Girshick, 2015; Dai et al, 2016b) are currently the leading methods. They exploit region proposals (Ren et al, 2015; Pinheiro et al, 2016; Gidaris and Komodakis, 2016a) as a first step to focus on interesting areas within images, and then classify and finely relocalize these regions at the same time.

Although they yield excellent results, region-based deep ConvNets still present a few issues that need to be solved. Networks are usually initialized with models pre-trained on

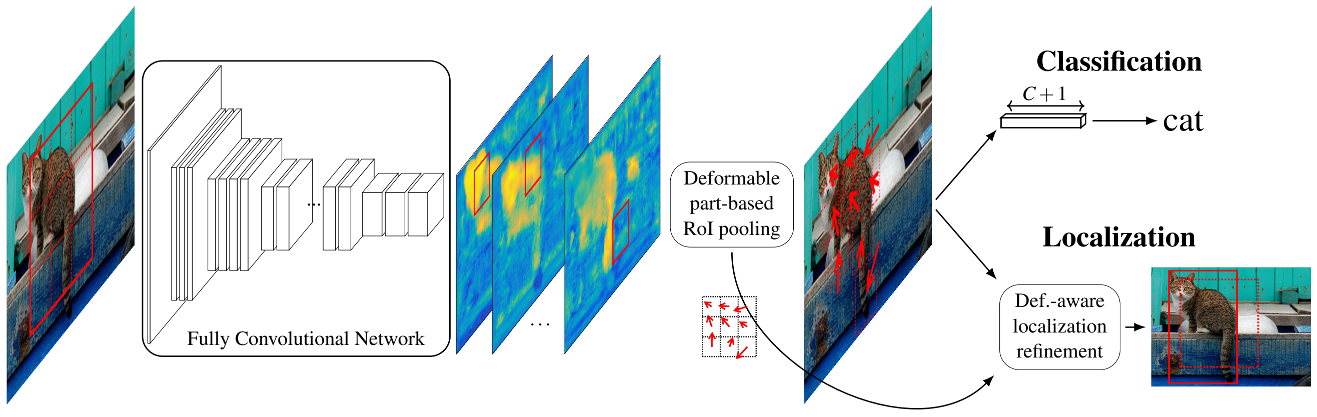
Fully Convolutional Network
Deformable part-based RoI pooling
Classification
$C+1$
cat
Localization
Def.-aware localization refinement

Fig. 2: **Architecture of DP-FCN.** It is composed of a FCN to extract dense feature maps with high spatial resolution (Section 3.1), a deformable part-based RoI pooling layer to compute a representation aligning parts (Section 3.2) and two sibling classification and localization prediction branches (Section 3.3). Initial rectangular region is deformed to focus on discriminative elements of object. Alignment of parts brings invariance for classification and geometric information refining localization *via* a deformation-aware localization module.

ImageNet dataset (Russakovsky et al, 2015) and are therefore prone to suffer from mismatches between classification and detection tasks. As an example, pooling layers bring invariance to local transformations and help learning more robust features for classification, but they also reduce the spatial resolution of feature maps and make the network less sensitive to the positions of objects within regions (Dai et al, 2016b), both of which are bad for accurate localization. Furthermore, the use of rectangular bounding boxes limits the representation of objects, in the way that boxes may contain a significant fraction of background, especially for non-rectangular objects.

Before the introduction of Deep Learning into object detection by Girshick et al (2014), the state of the art was led by approaches exploiting Deformable Part-based Models (DPMs) (Felzenszwalb et al, 2010). These methods are in contrast with region-based deep ConvNets: while the latter relies on strong features learned directly from pixels and exploit region proposals to focus on interesting areas of images, DPM explicitly takes into account geometry of objects by optimizing a graph-based representation and is usually applied in a sliding window fashion over images. Both approaches exploit different hypotheses and seem therefore complementary.

In this paper, we propose Deformable Part-based Fully Convolutional Network (DP-FCN) and its improved successor DP-FCN2.0, two end-to-end models integrating ideas from DPM into region-based deep ConvNets for object detection, as an answer to the aforementioned issues. They learn part-based representations of objects and align these parts to enhance both classification and localization (see Figure 1). Training is done with box-level supervision only, *i.e.* without part annotations. They improve upon existing object detectors with two key contributions.

The first one is the introduction of a new deformable part-based RoI pooling layer, which explicitly selects discriminative elements of objects around region proposals by simultaneously optimizing latent displacements of all parts (middle of Figure 2). Indeed, using a fixed box geometry must be sub-optimal, especially when objects are not rigid and parts can move relative to each other. Through alignment of parts, deformable part-based RoI pooling increases the limited invariance to local transformations brought by pooling, which is beneficial for classification.

In addition, aligning parts gives access to their configurations (*i.e.* their positions relative to each other), which brings important geometric information about objects, *e.g.* their shapes, poses or points of view. The second improvement is the design of a deformation-aware localization module (right of Figure 2), a specific module exploiting configuration information to refine localization. It improves bounding boxes regression by explicitly modeling displacements of parts in the localization branch, in order to tightly fit boxes around objects.

We integrate the previous ideas into Fully Convolutional Networks (FCNs) (He et al, 2016; Dai et al, 2016b) (left of Figure 2) and show that those architectures are amenable to an efficient computation of parts and their deformations. They also offer natural solutions to keep spatial resolution, which is beneficial since the application of deformable part-based approaches is severely dependent on the availability of rather fine feature maps (Savalle et al, 2014; Girshick et al, 2015; Wan et al, 2015).

This paper is a two-fold extension of our previous work (Mordan et al, 2017) already introducing DP-FCN. We first improve it here with DP-FCN2.0, which has better designs for both key modules of the model: a better part alignment in the deformable part-based RoI pooling layer (detailed in Section 3.2.2) and a more accurate description of shapes in the deformation-aware localization refinement module (detailed in Section 3.3.2). With these improvements, the network is now able to express more relations between all parts by explicitly taking their relative interactions into account, and so shapes of objects are described more finely. Our second main contribution is experimental. We present a more detailed ablation study, with additional visualizations of the models and their outputs. DP-FCN2.0 also obtains state-of-the-art results on standard PASCAL VOC 2007 and 2012 datasets (Everingham et al, 2015) with VOC data only, and especially show better results than Mordan et al (2017) in all common object detection metrics, *i.e.* both in recognition and localization. We finally experimentally validate the effectiveness of deformations on the more challenging and larger-scale MS COCO dataset (Lin et al, 2014).

2 Related work

Region-based object detectors. The leading approaches in object detection are currently region-based deep ConvNets. Since the seminal works of R-CNN (Girshick et al, 2014) and Fast R-CNN (Girshick, 2015), most object detectors exploit existing region proposals or directly learn to generate them (Ren et al, 2015; Gidaris and Komodakis, 2016a; Pinheiro et al, 2016), and then use RoI pooling layers to locally pool features within those regions. Compared to sliding window approach, the use of region proposals allows the model to focus the computation on interesting areas of images and to balance positive and negative examples to ease learning. Other improvements are now commonly used, *e.g.* using intermediate high-resolution layers to refine coarse deep feature maps (Bell et al, 2016; Kong et al, 2016; Zagoruyko et al, 2016; Lin et al, 2017a) in order to have a finer accuracy in locating objects, or selecting interesting regions for building mini-batches (Shrivastava et al, 2016; Dai et al, 2016b).

We note that there is a second kind of object detectors, not based on region proposals, *e.g.* YOLO (Redmon et al, 2016; Redmon and Farhadi, 2017), SSD (Liu et al, 2016). While their performances have long trailed behind those of region-based detectors, RetinaNet (Lin et al, 2017b) has now closed the gap between the two kinds of approaches.

Deformable Part-based Models. The core idea behind DPM (Felzenszwalb et al, 2010) is to represent each class by a root filter describing whole appearances of objects and a set of part filters to finely model local parts. Each part filter is assigned to an anchor point, defined relative from the root, and move around during detection to model deformations of objects and best fit them. A regularization is further introduced in the form of a deformation cost penalizing large displacements. Each part is then optimizing a trade-off between maximizing detection score and minimizing deformation cost. Final detection output combines scores from root and all parts. Accurate localization is done with a post-processing step.

Several extensions have been proposed to DPM, *e.g.* using a second hierarchical level of parts to finely describe objects (Zhu et al, 2010), sharing part models between classes (Ott and Everingham, 2011), learning from strongly supervised annotations (*i.e.* at the part level) to get a better model (Azizpour and Laptev, 2012), exploiting segmentation clues to improve detection (Fidler et al, 2013).

CRF optimization. Joint optimization of multiple variables is often performed to bring spatial coherence in tasks with structured predictions, such as semantic segmentation, *e.g.* Krähenbühl and Koltun (2011); Chen et al (2015); Zheng et al (2015). For this application, this yields improved results compared to independently classifying each pixel, by filtering out spatially isolated labels or taking more context into account. The optimization problem often being challenging, it is in most cases cast as an inference over a Conditional Random Field (CRF) tailored to the problem, for which there exist several algorithms. Krähenbühl and Koltun (2011) propose an efficient inference algorithm for fully connected CRFs relying on Mean Field approximation, and apply it to semantic segmentation task. They show improvements with joint optimization of all pixels with respect to independent prediction at each location, while keeping computational requirements low. The same algorithm has then been used in a number of following works in semantic segmentation, including Chen et al (2015); Zheng et al (2015). In particular, Zheng et al (2015) integrate it as layers within networks so that models are learned in an end-to-end way with CRFs. Those can then influence training, as they are not relegated to post-processing anymore. Chandra et al (2017) generalize this approach by learning deep embeddings, allowing exact inference over fully connected CRFs, and by applying it to other tasks than semantic segmentation, such as saliency estimation and human part segmentation. In this paper, we propose to cast the computation of deformations of regions as a CRF optimization, so that all parts are optimized jointly and their interactions are expressed in the model.

Part-based deep ConvNets. The first attempts trying to use deformable parts with deep ConvNets simply exploited deep features learned by an AlexNet (Krizhevsky et al, 2012) to use them with DPMs (Savalle et al, 2014; Girshick et al, 2015; Wan et al, 2015), but without region proposals. However, tasks implying spatial predictions (*e.g.* detection, segmentation) require fine feature maps in order to have accurate localization (Lin et al, 2017a). The fully connected layers were therefore discarded to keep enough spatial resolution, lowering results. We solve this issue by using a FCN, well suited for these kinds of applications as it naturally keeps spatial resolution. Thanks to several tricks easily integrable into FCNs (*e.g.* dilated convolutions (Chen et al, 2015; Long et al, 2015; Yu and Koltun, 2016) or skip pooling (Bell et al, 2016; Kong et al, 2016; Zagoruyko et al, 2016)), FCNs have recently been successful in various tasks, *e.g.* image classification (He et al, 2016; Zagoruyko and Komodakis, 2016; Xie et al, 2017), object detection (Dai et al, 2016b), semantic segmentation (Li et al, 2017), weakly supervised learning (Durand et al, 2017).

Zhang et al (2014) introduce parts for detection by learning part models and combining them with geometric constraints for scoring. It is learned in a strongly supervised way, *i.e.* with part annotations. Although manually defining parts can be more interpretable, it is likely sub-optimal for detection as they might not correspond to most discriminative elements.

Parts are often used for fine-grained recognition. Lin et al (2015) propose a module for localizing and aligning parts with respect to templates before classifying them, Simon and Rodner (2015) find part proposals from activation maps and learn a graphical model to recognize objects, Zhang et al (2016) use two sub-networks for detection and classification of parts, Sicre et al (2017) consider parts as a vocabulary of latent discriminative features decoupled from the task and learn them in an unsupervised way. Usage of parts is also common in semantic segmentation, *e.g.* Wang et al (2015); Dai et al (2016a); Li et al (2017).

The work closest to ours is Deformable ConvNet (Dai et al, 2017), a concurrent model which also exploits deformations to adapt to shapes of objects. While the ideas behind it are similar to ours, deformations are computed in a different way. Dai et al (2017) obtain deformations by using convolutional layers to estimate them, whereas we cast it as an optimization problem and solve it. While their approach is more general, in that it can be applied to convolutional layers in addition to RoI pooling layers, the solutions we propose in this paper are more controllable and can be tuned to specific purposes.

Our work is based on R-FCN (Dai et al, 2016b), which also uses a FCN to achieve a great efficiency. Compared to previous Fast R-CNN model (Girshick, 2015), the subnetworks after RoI pooling are here reduced at minimum to have very light per-region computation. Classification and localization for each region is then achieved by encoding information into several feature maps, processed by a position-sensitive RoI pooling layer, rather than in the following corresponding subnetworks. We improve upon it by learning more flexible representations than with fixed box geometry. It allows our model to align parts of objects to bring invariance into classification and to exploit geometric information from positions of parts to refine localization.

A previous version of this work was presented by Mordan et al (2017), which we extend here with new contributions. Our new model, named DP-FCN2.0, improves upon the first version by explicitly modeling interactions between parts, in both the part alignment and localization refinement stages. It is then able to learn more accurate representations of objects. Inspired by DPM (Felzenszwalb et al, 2010), deformable part-based RoI pooling from DP-FCN (Mordan et al, 2017) uses a star graphical model to move parts: displacements of parts only depend on the global region proposals, *i.e.* they are conditionally independent from each other given the positions of the region proposals. In contrast, DP-FCN2.0 uses a fully connected graph, *i.e.* where all parts relate to each other. By relaxing the conditional independence assumption, deformations for all parts are optimized jointly, and the model can exploit correlations between displacements to improve part alignment and recognition. The joint optimization is performed with a CRF integrated within the network, and its inference is carried out at each forward pass, allowing end-to-end learning similarly to what is done by Zheng et al (2015). The other major contribution deals with refining localization predictions with computed deformations. Again, DP-FCN2.0 outperforms its predecessor by encoding richer information. While DP-FCN only refines global predictions with features computed from deformations, DP-FCN2.0 lets predictions and displacements of all parts interact with each other through bilinear products to yield final predictions. By learning interactions between parts, the localization is much more effective in leveraging deformations of regions to identify shapes of objects.

3 Deformable Part-based Fully Convolutional Networks

In this section, we present our model Deformable Part-based Fully Convolutional Network (DP-FCN), a deep network for object detection. It represents regions with several parts it aligns by explicitly optimizing their positions. This alignment improves both classification and localization: the part-based representations are more invariant to local transformations and the configurations of parts give important information about the geometry of objects. This idea can be inserted into most of state-of-the-art network architectures. The model is end-to-end trainable without part annotation and adds a small computational overhead only.

The complete architecture is depicted in Figure 2 and is composed of three main modules: (i) a Fully Convolutional

Network (FCN) applied on whole images, (ii) a deformable part-based RoI pooling layer, and (iii) two sibling prediction layers for classification and localization. We now describe all three parts of our model in more details.

3.1 Fully convolutional feature extractor

Our model relies on a FCN (*e.g.* He et al, 2016; Zagoruyko and Komodakis, 2016; Xie et al, 2017) as backbone architecture, as this kind of network enjoys several practical advantages, leading to several successful models, *e.g.* Dai et al (2016b); Li et al (2017); Durand et al (2017). First, it allows to share most computation on whole images and to reduce per-RoI layers, as noted in R-FCN (Dai et al, 2016b). Second and most important to our work, it directly provides feature maps linked to the task at hand (*e.g.* detection heatmaps, as illustrated in the middle of Figure 2 and on the left of Figure 3) from which final predictions are simply pooled, as done by Dai et al (2016b); Durand et al (2017). Within DP-FCN, inferring the positions of parts for a region is done with a particular kind of RoI pooling that we describe in Section 3.2.

The fully convolutional structure is therefore suitable for computing responses of all parts for all classes as a single map for each of them. A corresponding structure is used for localization. The complete representation for a whole image (classification and localization maps for each part of each class) is obtained with a single forward pass and is shared between all regions of the same image, which is very efficient.

Since relocalization of parts is done within feature maps, the resolution of these maps is of practical importance. FCNs contain only spatial layers and are therefore well suited for preserving spatial resolution, as opposed to networks ending with fully connected layers, *e.g.* Krizhevsky et al (2012); Simonyan and Zisserman (2015). Specifically, if the stride is too large, deformations of parts might be too coarse to describe objects correctly. We reduce it by using dilated convolutions (Chen et al, 2015; Long et al, 2015; Yu and Koltun, 2016) on the last convolution block and skip pooling (Bell et al, 2016; Kong et al, 2016; Zagoruyko et al, 2016) to combine the last three.

3.2 Deformable part-based RoI pooling

The aim of this layer is to divide region proposals into several parts and to locally relocalize these to best match shapes of objects (as illustrated in Figure 1). Each part then models a discriminative local element and is to be aligned at the corresponding location within the image. This deformable part-based representation is more invariant to transformations of objects because the parts are positioned accordingly and their local appearances are stable (Felzenszwalb et al, 2010). This is especially useful for non-rigid objects, where a box-based representation must be sub-optimal.

The separation of a region R into parts is done with a regular grid of fixed size $I \times J$ fitted to it (Girshick, 2015; Dai et al, 2016b). Each cell (i, j) is then interpreted as a distinct part $R_{i,j}$. This strategy is simple yet effective (Zhu et al, 2010; Wan et al, 2015). Since the number of parts (*i.e.* IJ) is fixed as a hyper-parameter, it is easy to have a complete detection heatmap $z_{i,j,c}$ already computed for each part (i, j) of each class c (left of Figure 3). Part locations then only need to be optimized within corresponding maps.

The deformation of parts allows them to slightly move around their reference positions (partitions of the initial regions), selects the optimal latent displacements, and pools values from selected locations. During training, deformations are optimized without part-level annotations, *i.e.* only box-level annotations are needed, just as in the traditional object detection task. Displacements computed during the forward pass are stored and used to backpropagate gradients at the same locations. We further note that the deformations are computed for all parts and classes independently. However, no deformation is computed for the *background* class: they would not bring any relevant information as there is no discriminative elements for this class. The same displacements of parts are used to pool values from the localization maps.

We present two different strategies for computing deformations in the next sections. The first one, already introduced in (Mordan et al, 2017), considers each part independently from others. While this is highly efficient, it might miss complex relations between parts. In contrast, the second method performs a joint optimization on all parts simultaneously and takes interactions between parts into account by leveraging a CRF formulation. It is then able to model object geometries more finely.

3.2.1 Independent deformations of parts

This first approach (Mordan et al, 2017) draws ideas from the original DPM (Felzenszwalb et al, 2010) and is applied separately to all parts. For a part (i, j) of a region R and a class c, the set $\Delta_{i,j}^{R}$ of possible displacements $\delta = (\delta x, \delta y)$ is such that the part $R_{i,j}$ still stays within the feature map z after moving by δ. We then define the score $S_{i,j,c}^{R}(\delta)$ of these part and class for a displacement $\delta \in \Delta_{i,j}^{R}$ as the value pooled at the new location ($R_{i,j}$ offset by δ) and penalized by the magnitude of the displacement:

$$S_{i,j,c}^{R}(\delta) = \underset{(x,y)\in R_{i,j}}{\text{Pool}} \; z_{i,j,c}(x+\delta x, y+\delta y) - \lambda^{def} \|\delta\|_2^2 \qquad (1)$$

where λ^{def} represents the strength of the regularization (bias toward small deformations), and Pool is an average pooling as in (Dai et al, 2016b), but any pooling function could

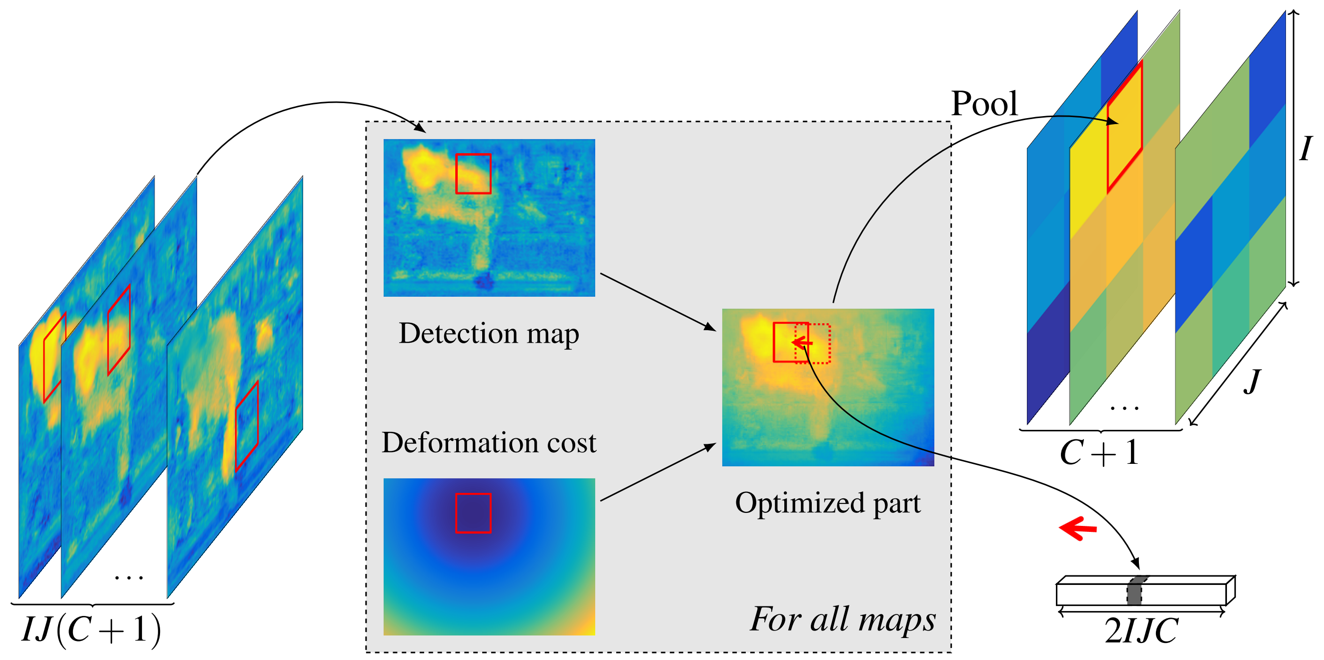

Fig. 3: **Deformable part-based RoI pooling with independent deformations.** Each input feature map corresponds to a part of a class (or *background*). Positions of parts are optimized separately within detection maps with deformation costs as regularization, and values are pooled within parts at the new locations. Output includes a map for each class and the computed displacements of parts, to be used for localization.

be used instead. Here, the deformation cost is the squared distance of the displacement on the feature map, but other functions could be used equally. The deformable part-based RoI layer consists in maximizing this quantity with respect to the displacement, and therefore optimizes a trade-off between maximizing the score on the corresponding feature map and minimizing the displacement from the reference position (see Figure 3). Its output $p_c^R(i,j)$ then writes:

$$p_c^R(i,j) = \max_{\delta \in \Delta_{i,j}^R} \left[S_{i,j,c}^R(\delta) \right] \tag{2}$$

$$= \max_{\delta \in \Delta_{i,j}^R} \left[\operatorname*{Pool}_{(x,y)\in R_{i,j}} z_{i,j,c}(x+\delta x, y+\delta y) - \lambda^{def} \|\delta\|_2^2 \right]. \tag{3}$$

While Equation (3) is used to compute the output of the layer for part (i,j) of region R and class c, it also gives the displacement $d_c^R(i,j) = \left(dx_c^R(i,j), dy_c^R(i,j)\right)$ for that part: it is the argmax of Equation (3), *i.e.* the $\delta = (\delta x, \delta y)$ maximizing it. Those displacements are extracted from the layer to be used for localization thereafter (see Section 3.3). We emphasize that this formulation does not require any annotations about positions of parts, and can therefore be used in the standard object detection setup (*i.e.* with bounding boxes only).

λ^{def} is directly linked to the magnitudes of the displacements of parts, and therefore to the deformations of RoIs too, by controlling the squared distance regularization (*i.e.* preference for small deformations). Increasing it puts a higher weight on regularization and effectively reduces displacements of parts, but setting it too high prevents parts from moving and removes the benefits of our approach. It is noticeable this deformable part-based RoI pooling is a generalization of the position-sensitive RoI pooling from Dai et al (2016b). Setting $\lambda^{def} = +\infty$ clamps all displacements $d_c^R(i,j)$ to $(0,0)$, leading to the formulation of position-sensitive RoI pooling:

$$p_c^R(i,j) = \operatorname*{Pool}_{(x,y)\in R_{i,j}} z_{i,j,c}(x,y). \tag{4}$$

On the other hand, setting $\lambda^{def} = 0$ removes regularization and parts are then free to move. With λ^{def} too low, the results decrease, indicating that regularization is practically important. However, the results appeared to be stable within a large range of values of λ^{def}. Additionally, optimization of δ is performed by brute force in a limited range and not the whole image, *i.e.* the sets $\Delta_{i,j}^R$ are restricted to their intersections with a centered ball of small radius. With λ^{def} not too small, the regularization effectively restricts displacements to lower values, leaving the results of pooling unchanged. In all experiments, we use $\lambda^{def} = 0.3$.

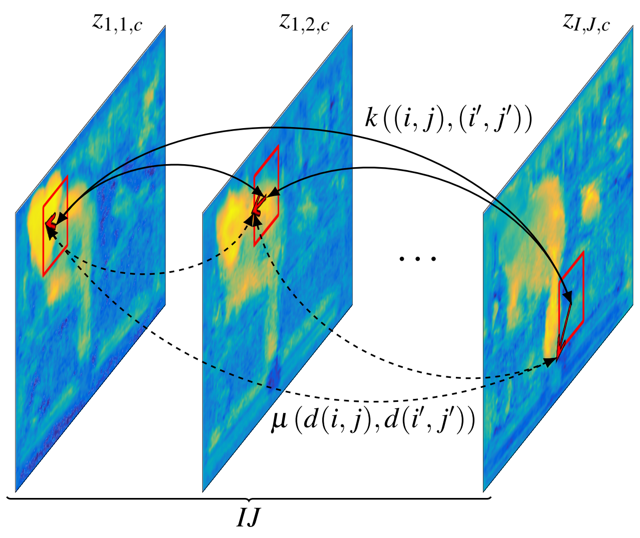
$z_{1,1,c}$
$z_{1,2,c}$
$z_{I,J,c}$
$k\left((i,j),(i',j')\right)$
$\mu\left(d(i,j),d(i',j')\right)$
IJ

Fig. 4: **Visualization of pairwise potentials of CRF between parts** for a region of a class c. Interactions between all IJ parts are taken into account through pairwise potentials ϕ_p. These are composed of two main terms: a kernel k controlling the strength of the interactions according to the distances between parts, and a compatibility function μ encouraging similarity of displacements.

We further normalize the displacements dx_c^R and dy_c^R by the heights and widths of parts respectively to make the layer invariant to the scales of the images and regions. Indeed, the parts should move to the same positions relative to the objects, regardless of the scales at which they appear in the images and irrespective of any scaling factor applied to the images. We also normalize the classification feature maps before forwarding them to deformable part-based RoI pooling layer to ensure classification and regularization terms are comparable. We do this by L_2-normalizing at each spatial location the block of $C+1$ maps for each part separately, *i.e.* replacing z from Equation (3) with

$$\bar{z}_{i,j,c}(x,y) = \frac{z_{i,j,c}(x,y)}{\sqrt{\sum_{c'} z_{i,j,c'}(x,y)^2}}. \tag{5}$$

3.2.2 CRF-based joint deformations of parts

The second strategy to compute deformations jointly considers all parts in a single optimization problem. All displacements are inferred simultaneously, so that it is possible to model dependencies between them and enforce consistency. We then have a fully connected graphical model, *i.e.* displacement of a given part is influenced by those of all other parts. This is in contrast with the independent deformations from Section 3.2.1 which uses a star model, *i.e.* parts are conditionally independent from each other given the whole region, like the original DPM (Felzenszwalb et al, 2010).

We do this by casting the optimization problem into a Conditional Random Field (CRF) inference over displacements of parts within regions. We define original unary and pairwise potentials by hand so that the CRFs act as a regularization and lead to a more robust part alignment stage. By integrating the CRF inference algorithm within the deformable part-based RoI pooling layer, *i.e.* the inference is carried out for all regions at each forward pass, we are still able to perform end-to-end training on GPU with a moderate overhead.

A different CRF is instantiated for each region R and class c (but for the *background* class as no deformations are computed), and they are all optimized in parallel during forward passes. There are $I \times J$ variables $D_c^R(i,j)$ considered here, each associated with a given part (i,j) and indicating its displacement $d_c^R(i,j)$. The Gibbs probability distribution of the CRF conditioned on an image I is then

$$P\left(D_c^R = d_c^R | I\right) = \frac{1}{Z_c^R(I)} \exp\left(-E_c^R(d_c^R|I)\right) \tag{6}$$

with Z_c^R the partition function and E_c^R the corresponding Gibbs energy (Lafferty et al, 2001). From now on, we drop the R and c notations as well as the conditioning on image I for convenience.

We use the fully connected CRF formulation of Krähenbühl and Koltun (2011) to model dependencies between all pairs of parts. The Gibbs energy E for displacements d then takes the form

$$E(d) = \sum_{i,j} \phi_u\left(d(i,j)\right) + \sum_{(i,j)<(i',j')} \phi_p\left(d(i,j),d(i',j')\right) \tag{7}$$

where ϕ_u and ϕ_p are the unary and pairwise potentials.

The unary potential ϕ_u is computed independently for each part, and is based on the visual features (*i.e.* the feature maps z) only. It does not consider any relations between parts nor produce consistency between their displacements. For each part (i,j), it gives a negative log-probability distribution over possible displacements for that part. We use the score function $S_{i,j}$ from the independent deformation model (defined in Equation (1) from Section 3.2.1) as unnormalized probability distribution and apply a $SoftMax$ function to it to obtain a valid distribution, yielding

$$\phi_u\left(d(i,j)\right) = -LogSoftMax\left[S_{i,j}\right]\left(d(i,j)\right). \tag{8}$$

The main purpose of using a CRF is to use a pairwise potential ϕ_p to relate pairs of displacements in order to enforce consistency between them (see Figure 4). We use it here to smooth the deformation field over the region by introducing the constraint that nearby parts should have similar displacements, through the design of a specific form for the potential ϕ_p. Doing so, it increases the robustness of the part alignment stage. Following Krähenbühl and Koltun (2011), we use a potential of the form

$$\phi_p\left(d(i,j),d(i',j')\right) = w_0\, k\left((i,j),(i',j')\right) \times \mu\left(d(i,j),d(i',j')\right) \quad (9)$$

where w_0 is the weight of the pairwise component, k is a gaussian kernel and μ is a compatibility function between displacements.

We define dedicated functions k and μ suited to our particular problem of computing deformations of a region. The kernel k controls the weights of the pairwise links according to how far apart the parts are, and has the following expression:

$$k\left((i,j),(i',j')\right) = \exp\left(-\frac{|i-i'|^2+|j-j'|^2}{2\sigma^2}\right) \quad (10)$$

with σ giving the width of the kernel. The compatibility function μ gives the penalty assigned to a pair of displacements, and we choose it so that the deformation field over the region tends to be smoother, then acting as a regularization:

$$\mu\left(d(i,j),d(i',j')\right) = \frac{|dx(i,j)-dx(i',j')|^2}{\sigma_d} + \frac{|dy(i,j)-dy(i',j')|^2}{\sigma_d} \quad (11)$$

with σ_d controlling the strength of the penalty according to how similar the displacements are. Other norms can also be used in μ (*i.e.* changing the exponent of the power), but they experimentally do not yield any improvement. In summary, the pairwise potential ϕ_p takes the form

$$\phi_p\left(d(i,j),d(i',j')\right) = w_p \exp\left(-\frac{|i-i'|^2+|j-j'|^2}{2\sigma^2}\right) \times \left(|dx(i,j)-dx(i',j')|^2+|dy(i,j)-dy(i',j')|^2\right) \quad (12)$$

where $w_p = \frac{w_0}{\sigma_d}$.

We run T iterations of a Mean Field algorithm to perform approximate inference on the CRF, and use an efficient gaussian filtering in order to speed it up (Krähenbühl and Koltun, 2011). This is done simultaneously for all classes c and all regions R at each forward pass, *i.e.* all the CRFs are optimized in parallel, in order to obtain all the deformations d_c^R. These are then used to backpropagate gradients at selected locations, as done with independent deformations. While there are multiple CRFs to optimize at the same time, they are all rather small since the number of variables (*i.e.* the number of parts IJ) is limited. Therefore, this only adds a moderate overhead compared to having independent deformations. In all experiments, we use $w_p = 0.3$, $\sigma = 1.3$ and we perform a single Mean Field iteration (*i.e.* $T = 1$), as doing more iterations does not lead to significant improvement.

We note that this CRF-based formulation of deformable part-based RoI pooling is a generalization of the independent deformation formulation of Mordan et al (2017) (Section 3.2.1). Indeed, setting the pairwise weight $w_p = 0$ or doing no iteration of Mean Field inference (*i.e.* $T = 0$) results in maximizing $S_{i,j}$, which is exactly Equation (2).

3.3 Classification and localization predictions with deformable parts

Predictions are performed with two sibling branches, for classification and relocalization of region proposals, as is common practice (Girshick, 2015). The classification branch is simply composed of an average pooling followed by a SoftMax layer. This is the strategy employed in R-FCN (Dai et al, 2016b), but the deformations introduced before (with deformable part-based RoI pooling) bring more invariance to transformations of objects and boost classification.

Regarding localization, the same approach is used by R-FCN, *i.e.* a simple average of pooled localization values. However, this is not adapted to DP-FCN as it is for classification, due to the presence of deformations. Indeed, while the positions and dimensions of input bounding boxes are implied by the pooling regions (*i.e.* parts) in R-FCN, it is no longer the case when those are moved by a deformable part-based RoI pooling layer. With the same strategy as R-FCN, the network would not keep track of the displacements of parts (which are never made explicit in this architecture) and would therefore be unaware of the exact input bounding box to be relocalized, leading to approximate localization.

To solve that issue, we introduce a deformation-aware localization module, explicitly taking deformations of parts into account. Since we want bounding boxes to tightly enclose objects, localization should not be invariant to local transformations but adapt accordingly. The configuration of parts (*i.e.* their positions relative to each other) is obtained as a by-product of the alignment of parts performed before, and can then be exploited to refine naive localization predictions obtained from pooling at deformed locations, so that exact geometries of bounding boxes are recovered. It also gives rich geometric information about the appearances of objects, *e.g.* their shapes or poses, that can be used to further enhance localization accuracy.

In the following sections, we introduce two versions of localization refinement module. The first approach computes naive, deformation-unaware predictions, then uses displacements of parts to improve them; it is already presented in (Mordan et al, 2017). Rather than considering global predictions only, the second method exploits partial predictions made by all parts individually, and directly combines them with displacements of parts to yield final predictions. That way, interactions between both positions and outputs of all

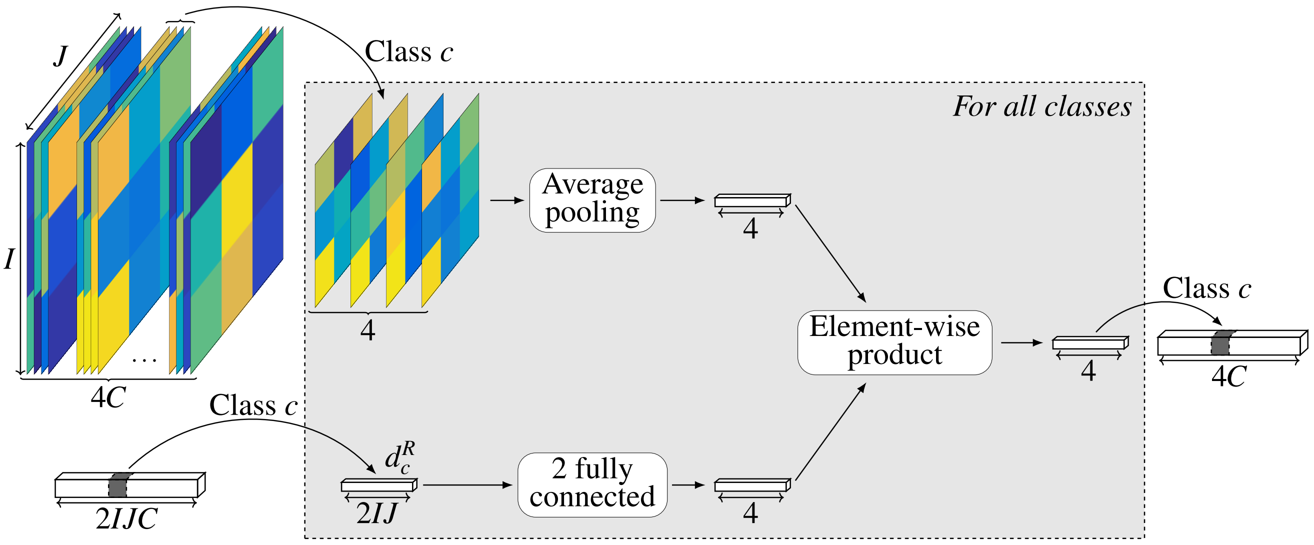

Fig. 5: **Deformation-aware global localization refinement.** Relocalizations of bounding boxes obtained by averaging pooled values from localization maps (upper path) do not benefit from deformable parts. To do so, displacements of parts are forwarded through two fully connected layers (lower path) and are element-wise multiplied with the previous output to refine it, separately for each class. Localization is done with 4 values per class, following Girshick et al (2014); Girshick (2015).

parts can be expressed, resulting in a more accurate localization.

For both modules, the refinement is mainly geometric rather than semantic, *i.e.* it depends only on the displacements of parts and not on the classes of objects. Therefore, the same configuration of parts should give the same refinement. For this reason, the localization is applied for each class separately and parameters are shared between classes. Additionally, sharing parameters can act as a regularization for classes with fewer examples.

3.3.1 Global localization refinement

This localization module (Mordan et al, 2017) separately processes outputs and displacements of parts, for a class c and a region R, before merging them with a simple operation (see Figure 5). It exploits the strategy of R-FCN, *i.e.* an average pooling of partial predictions from parts, to compute a first deformation-unaware prediction (upper path in Figure 5). This output is based on visual features only, without considering deformations, as noted before.

For that reason, we extract the feature vector d_c^R of normalized displacements (dx_c^R, dy_c^R) of all parts, computed by the deformable part-based RoI pooling layer (as shown in the bottom right corner of Figure 3), and use it to refine previous naive prediction. d_c^R, of size $2IJ$ (*i.e.* a 2D displacement for each part), is forwarded through a simple sub-network (lower path in Figure 5) to yield a feature vector of size 4 (the same as the prediction, following Girshick et al (2014); Girshick (2015)) encoding the positions of parts. The sub-network is composed of two fully connected layers with a ReLU between them. The size of the first layer is set to 256 in all our experiments. The result is then element-wise multiplied with the first prediction to adjust it accordingly to the exact locations where it was computed, yielding the final localization output.

3.3.2 Bilinear localization refinement

While the previous method computes a prediction and only globally refines it with deformations, this second approach to localization refinement jointly considers all partial predictions and displacements of parts in a single operation. That way, it expresses interactions between parts more effectively and at a finer level.

To do this we use a bilinear product between predictions and displacements, that directly outputs the final localization (see Figure 6), which is of size 4 as before. With that operation, all pairs of prediction and displacement, even from different parts, contribute to the output. It can therefore model richer and more complex shapes than the global relocalization, and the final detections are more accurate.

To reduce computation here, we use a Tucker decomposition (Tucker, 1966): we compute two feature vectors u_c^R and v_c^R of lower size s for both partial predictions and displacements, with a simple fully connected layer applied to each input, and only feed these two vectors into the bilinear

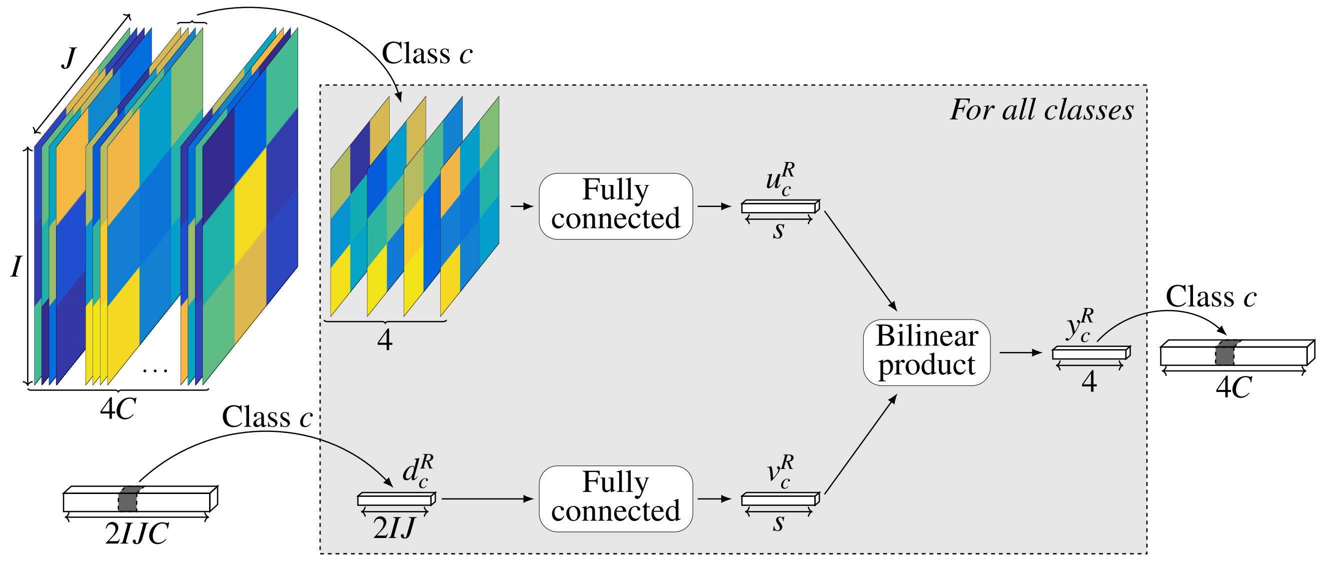

Fig. 6: **Deformation-aware bilinear localization refinement.** For each region and class, both predictions and displacements from all parts are separately embedded into lower dimensional features before feeding a bilinear product layer (*i.e.* a Tucker decomposition) to yield final localization prediction of size 4, following Girshick et al (2014); Girshick (2015). This kind of refinement naturally learns relations between pairs of parts, and so describes shapes of objects more finely.

layer. Each of the four localization output values y_c^R is then obtained with

$$y_c^R(l) = \sum_{m=1}^{s}\sum_{n=1}^{s} u_c^R(m)\mathrm{T}(m,n,l)v_c^R(n) + b(l) \tag{13}$$

where T is a tensor of size $s \times s \times 4$ and b is a bias of size 4, both learned within the layer and shared between classes. In all experiments, we use a reduced size of $s = 32$, which keeps memory and computation requirements low. While having bigger features yields slightly better results, we think this is a good trade-off between performance and computation. More complex combination operations could be used instead of the Tucker decomposition to further improve performance, *e.g.* MUTAN (Ben-Younes et al, 2017).

4 Experiments

4.1 Main results

Experimental setup. We perform this analysis with the fully convolutional backbone architecture ResNet-50 (He et al, 2016) whose model, pre-trained on ImageNet (Russakovsky et al, 2015), is freely available. The network is trained with SGD for 60,000 iterations with a learning rate of $5 \cdot 10^{-4}$ and for 20,000 further iterations with $5 \cdot 10^{-5}$. The momentum parameter is set to 0.9 and the weight decay to 10^{-4}. Each mini-batch is composed of 64 regions from a single image at the scale of 600px, selected according to Fast R-CNN (Girshick, 2015). Horizontal flipping of images with probability 0.5 is used as data augmentation. We exploit the region proposals computed by AttractioNet (Gidaris and Komodakis, 2016b,a) released by the authors. The top 2,000 regions are used for learning and the top 300 are evaluated during inference. We use $I \times J = 7 \times 7$ parts, as advised by the authors of R-FCN (Dai et al, 2016b). As is common practice, detections are post-processed with NMS with the standard threshold of 0.3.

All experiments in this section are conducted on PASCAL VOC 07+12 dataset (Everingham et al, 2015): training is done on the union of the 2007 and 2012 trainval sets and testing on the 2007 test set. In addition to the standard mAP@0.5 (*i.e.* PASCAL VOC style) metric, results are also reported with the mAP@0.75 and mAP@[0.5:0.95] (*i.e.* MS COCO style) metrics to thoroughly evaluate the effects of proposed improvements.

Performances of models. Performance of our implementation of R-FCN (Dai et al, 2016b) with the given setup is shown in the first row of Table 1. Using independent deformations and global localization refinement, DP-FCN (second row of Table 1) outperforms R-FCN in all three metrics with large margins. In particular, it gains 2.0 points in mAP@0.5 over R-FCN. Then, with the improved joint deformations and bilinear localization refinement, DP-FCN2.0 (last row of Table 1) has better results, with an significant improvement of 4.4 points in mAP@0.75 with respect to DP-FCN. These results validate the effectiveness of deformations within networks to enhance detection, and also that

Model	Independent deformations	Joint deformations	Global localization refinement	Bilinear localization refinement	mAP@ 0.5	mAP@ 0.75	mAP@ [0.5:0.95]
R-FCN (Dai et al, 2016b)					74.1	39.4	40.0
DP-FCN (Mordan et al, 2017)	✓		✓		76.1	40.9	41.3
DP-FCN2.0 (ours)		✓		✓	**76.5**	**45.3**	**43.2**

Table 1: **Main results of DP-FCN2.0** on PASCAL VOC 2007 test in average precision (%). Without deformable part-based RoI pooling nor localization refinement module, it is equivalent to R-FCN (the reported results are those of our implementation with the given setup).

$\left(\begin{array}{c}\textbf{mAP@}\\\textbf{0.5}\end{array}\right)$	No localization refinement	Global localization refinement	Bilinear localization refinement
No deformation	R-FCN 74.1	–	–
Independent deformations	75.8 (+1.7)	DP-FCN 76.1 (+2.0)	76.4 (+2.3)
Joint deformations	–	76.4 (+2.3)	DP-FCN2.0 76.5 (+2.4)

Table 2: **Ablation study of DP-FCN2.0 in mAP@0.5** on PASCAL VOC 2007 test in average precision (%). Results are given with absolute performances, with improvements with respect to R-FCN between parenthesis.

$\left(\begin{array}{c}\textbf{mAP@}\\\textbf{0.75}\end{array}\right)$	No localization refinement	Global localization refinement	Bilinear localization refinement
No deformation	R-FCN 39.4	–	–
Independent deformations	38.8 (-0.6)	DP-FCN 40.9 (+1.5)	45.0 (+5.6)
Joint deformations	–	40.5 (+1.1)	DP-FCN2.0 45.3 (+5.9)

Table 3: **Ablation study of DP-FCN2.0 in mAP@0.75** on PASCAL VOC 2007 test in average precision (%). Results are given with absolute performances, with improvements with respect to R-FCN between parenthesis.

$\left(\begin{array}{c}\textbf{mAP@}\\\textbf{[0.5:0.95]}\end{array}\right)$	No localization refinement	Global localization refinement	Bilinear localization refinement
No deformation	R-FCN 40.0	–	–
Independent deformations	40.4 (+0.4)	DP-FCN 41.3 (+1.3)	42.9 (+2.9)
Joint deformations	–	41.6 (+1.6)	DP-FCN2.0 43.2 (+3.2)

Table 4: **Ablation study of DP-FCN2.0 in mAP@[0.5:0.95]** on PASCAL VOC 2007 test in average precision (%). Results are given with absolute performances, with improvements with respect to R-FCN between parenthesis.

richer models of deformations (*i.e.* with interactions between parts) lead to better performance.

4.2 Ablation study

Experimental setup. For this ablation study, we use the same experimental setup as before (Section 4.1) so that results are directly comparable.

Analysis of models. We present a detailed analysis of results for each new module in Table 2, Table 3 and Table 4 for the three metrics mAP@0.5, mAP@0.75 and mAP@[0.5:0.95] respectively. In each table, R-FCN is shown in the top left corner as the baseline. Adding the deformable part-based RoI pooling with independent deformations to R-FCN (second rows of tables) improves mAP@0.5 by 1.7 points. Indeed, this metric is rather permissive so the localization does not need to be very accurate. On the other hand, we see a negative effect on mAP@0.75. That is due to the uncertainty in the positions of parts, leading to an imprecise localization as already noted in Section 3.3. Overall, this is still beneficial, with a gain of 0.4 points in mAP@[0.5:0.95]. The improvements are therefore mainly due to a better recognition, thus validating the role of deformable parts. With the global localization refinement module (second columns of tables), the mAP@0.5 has only a small improvement, because localization accuracy is not a issue. However, it further improves mAP@0.75 by 2.1 points (*i.e.* 1.5 points with respect to R-FCN) and mAP@[0.5:0.95] by 0.9 points, validating the need for such a module. This confirms that it solves the previous issue of approximate localization and that aligning parts brings geometric information useful for localization.

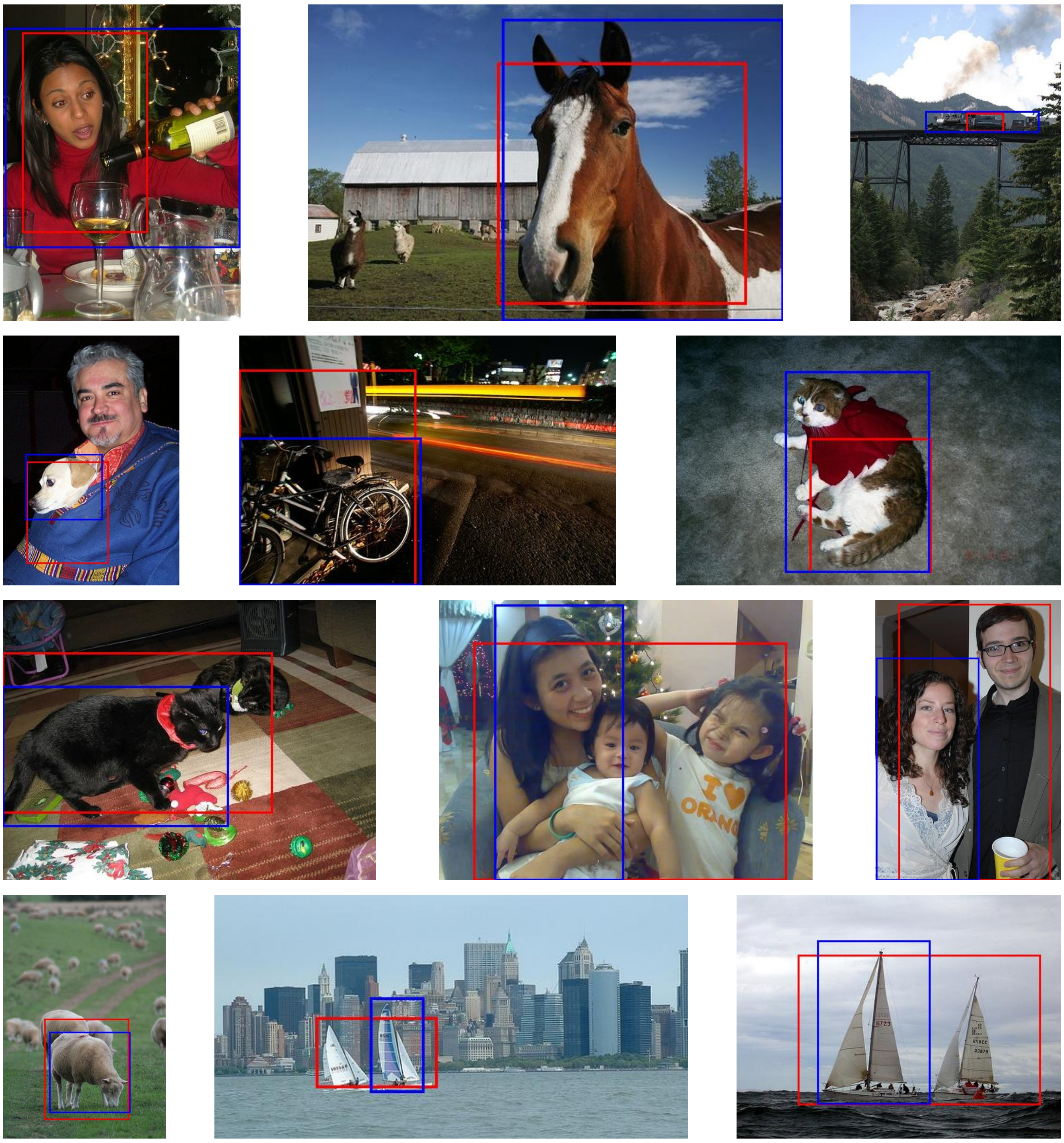

Fig. 7: **Comparison of detections from R-FCN (red) and DP-FCN (blue).** DP-FCN tightly fits objects (first two rows) and separates close instances (last two rows) better than R-FCN.

We then change the independent deformations to use the joint CRF-based ones (last rows of tables), which brings an additional improvement of 0.3 points for both mAP@0.5 and mAP[0.5:0.95] metrics with respect to Mordan et al (2017). This therefore confirms that deformations play an important role in recognition, as already noted. When using the bilinear localization refinement (last columns of tables) in place of the global one, it yields great improvements of 4.1 and 1.6 points in mAP@0.75 and mAP@[0.5:0.95] respectively, while it is smaller in mAP@0.5. This again confirms that this module is mainly dealing with the accuracy of the localization, but not with the recognition of the object

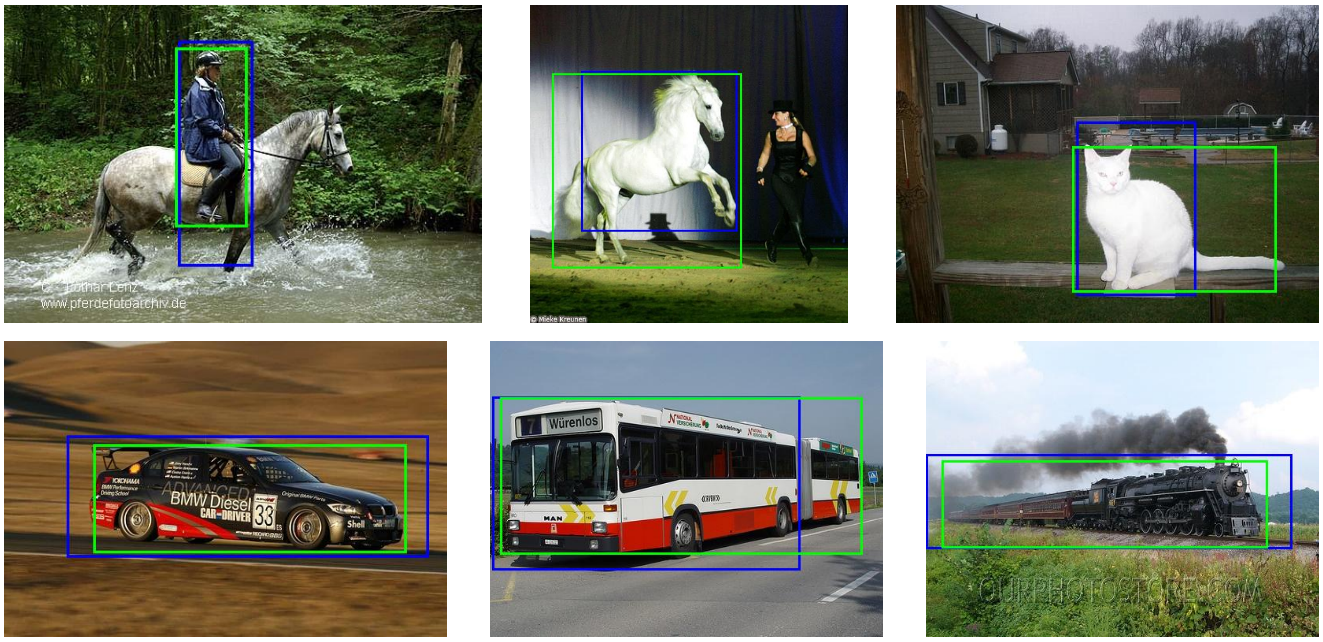

Fig. 8: **Comparison of detections from DP-FCN (blue) and DP-FCN2.0 (green).** Predictions of DP-FCN2.0 are better localized in general.

Model	Number of parameters	Number of FLOPs	Forward time (s)
R-FCN (Dai et al, 2016b)	32.26 M	133.6 G	0.167
DP-FCN (Mordan et al, 2017)	32.28 M	134.3 G	0.299
DP-FCN2.0 (ours)	32.27 M	152.5 G	0.492

Table 5: **Runtime analysis of DP-FCN2.0.** Values reported are computed with ResNet-50 on images at scale of 600px, and averaged over PASCAL VOC 2007 test.

categories. By combining both improved modules (bottom right corners of tables), DP-FCN2.0 has additional gains in all three metrics, showing that the two contributions are complementary, and validates the importance of taking interactions of parts into account for accurate predictions.

4.3 Further analysis

Comparison with R-FCN. Some examples of detection outputs are illustrated in Figure 7 to visually compare R-FCN and DP-FCN, and evaluate proposed improvements. It appears that R-FCN can more easily miss extremal parts of objects (see first two rows, *e.g.* the woman's left arm or the ears of the horse), and that DP-FCN is better at separating close instances (see last two rows, *e.g.* people or boats next to each other), thanks to deformable parts. While detections from DP-FCN and DP-FCN2.0 are often rather similar, the latter generally fits objects more tightly. We show some examples of that in Figure 8.

Runtime analysis. We present some statistics about R-FCN, DP-FCN and DP-FCN2.0 in Table 5. The first column shows that all models have roughly the same number of parameters, *i.e.* our approaches do not bring many additional parameters and so should not need significantly more examples to be learned. The average number of FLOPs (multiply-adds) and times of network forward passes are displayed in the following two columns. It is noticeable that DP-FCN yields a moderate overhead compared to R-FCN, while the more computational intensive inference carried out by DP-FCN2.0, because of the CRFs introduced, leads to a heavier model.

Interpretation of parts. As in the original DPM (Felzenszwalb et al, 2010), the semantics of parts is not explicit in our model. Part positions are instead automatically learned to optimize detection performance, in a weakly supervised manner. Therefore the interpretation in terms of semantic parts is not systematic, especially because our division of regions into parts is finer than in DPM, leading to smaller part areas. Some deformed parts are displayed on Figure 9 for DP-FCN and Figure 10 for DP-FCN2.0, with a 3×3

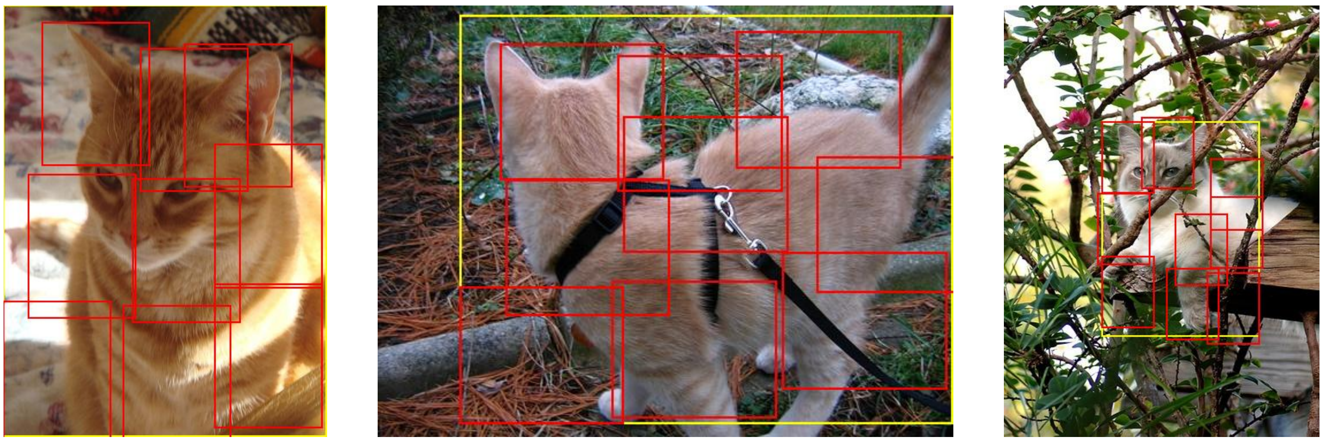

Fig. 9: **Examples of deformations of parts from DP-FCN.** Initial region proposals are shown in yellow and deformed parts in red. Only 3 $\times$ 3 parts are displayed for clarity.

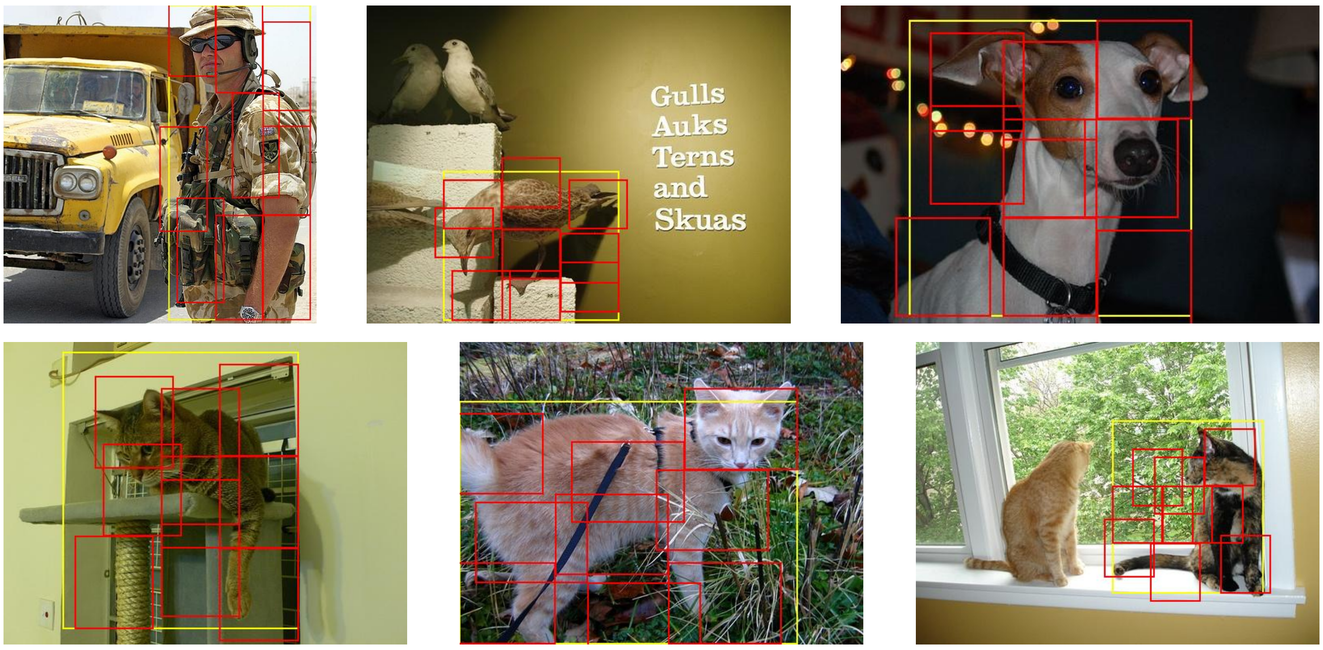

Fig. 10: **Examples of deformations of parts from DP-FCN2.0.** Initial region proposals are shown in yellow and deformed parts in red. Only 3 $\times$ 3 parts are displayed for clarity.

part division for easier visualization. It is noticeable that the models are able to better fit to objects with deformable parts than with simple bounding boxes.

Network architecture. We compare DP-FCN with several FCN backbone architectures in Table 6, in particular the 50- and 101-layer versions of ResNet (He et al, 2016), Wide ResNet (Zagoruyko and Komodakis, 2016) and ResNeXt (Xie et al, 2017). We see that the detection mAP of DP-FCN can be significantly increased by using better networks. ResNeXt-101 (64x4d) gives the best results among the tested ones, with large improvements in all metrics, despite not using dilated convolutions. We expect DP-FCN2.0 to behave similarly, in particular to give the best results with ResNeXt-101 (64x4d) too.

4.4 Comparison with state of the art

Experimental setup. In order to achieve the best results possible, we bring the following improvements to the setup of Section 4.2: we first replace ResNet-50 by ResNeXt-101 (64x4d) (Xie et al, 2017) and increase the number of iterations to 120,000 and 40,000 on PASCAL VOC datasets, and to 480,000 and 160,000 on MS COCO dataset, with the same learning rates, using 2 images per mini-batch with

FCN architecture for DP-FCN (Mordan et al, 2017)	mAP@0.5	mAP@0.75	mAP@[0.5:0.95]
ResNet-50 (He et al, 2016)	76.1	40.9	41.3
ResNeXt-50 (32x4d) (Xie et al, 2017)*	76.3	40.8	41.4
Wide ResNet-50-2 (Zagoruyko and Komodakis, 2016)	77.9	43.3	42.9
ResNet-101 (He et al, 2016)	78.1	44.2	43.6
ResNeXt-101 (32x4d) (Xie et al, 2017)*	78.6	45.2	44.4
ResNeXt-101 (64x4d) (Xie et al, 2017)*	**79.5**	**47.8**	**45.7**

Table 6: **Comparison of different FCN architectures used with DP-FCN** (Mordan et al, 2017) on PASCAL VOC 2007 test in average precision (%). Entries marked with * do not use dilated convolutions.

Method	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
FRCN (Girshick, 2015)	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
HyperNet (Kong et al, 2016)	76.3	77.4	83.3	75.0	69.1	62.4	83.1	87.4	87.4	57.1	79.8	71.4	85.1	85.1	80.0	79.1	51.2	79.1	75.7	80.9	76.5
Faster R-CNN (Ren et al, 2015)	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
SSD (Liu et al, 2016)	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
MR-CNN (Gidaris and Komodakis, 2015)	78.2	80.3	84.1	78.5	70.8	68.5	88.0	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85.0	76.4	48.5	76.3	75.5	85.0	81.0
LocNet (Gidaris and Komodakis, 2016b)	78.4	80.4	85.5	77.6	72.9	62.2	86.8	87.5	88.6	61.3	86.0	73.9	86.1	87.0	82.6	79.1	51.7	79.4	75.2	86.6	77.7
FRCN OHEM (Shrivastava et al, 2016)	78.9	80.6	85.7	79.8	69.9	60.8	88.3	87.9	89.6	59.7	85.1	**76.5**	87.1	87.3	82.4	78.8	53.7	80.5	78.7	84.5	80.7
ION (Bell et al, 2016)	79.4	82.5	86.2	79.9	71.3	67.2	88.6	87.5	88.7	60.8	84.7	72.3	87.6	87.7	83.6	82.1	53.8	81.9	74.9	85.8	81.2
R-FCN (Dai et al, 2016b)	80.5	79.9	87.2	81.5	72.0	69.8	86.8	88.5	89.8	**67.0**	**88.1**	74.5	89.8	90.6	79.9	81.2	53.7	81.8	**81.5**	85.9	79.9
Deformable ConvNet (Dai et al, 2017)	82.6																				
DP-FCN (Mordan et al, 2017)	83.1	89.8	**88.6**	**85.2**	73.9	**74.7**	**92.1**	90.4	**94.4**	58.3	84.9	75.2	93.4	**93.1**	**87.4**	**85.9**	53.9	85.3	80.0	90.4	85.9
DP-FCN2.0 (ours)	**83.3**	**92.0**	**88.6**	83.9	**75.9**	72.8	89.9	**91.5**	93.1	57.4	85.5	75.4	**94.1**	92.7	87.0	85.0	**55.6**	**85.6**	80.6	**92.7**	**86.2**

Table 7: **Detailed detection results on PASCAL VOC 2007 test** in average precision (%). For fair comparisons, the table only includes methods trained on PASCAL VOC 07+12.

Method	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
FRCN (Girshick, 2015)	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
HyperNet (Kong et al, 2016)	71.4	84.2	78.5	73.6	55.6	53.7	78.7	79.8	87.7	49.6	74.9	52.1	86.0	81.7	83.3	81.8	48.6	73.5	59.4	79.9	65.7
Faster R-CNN (Ren et al, 2015)	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
SSD (Liu et al, 2016)	74.9	87.4	82.3	75.8	59.0	52.6	81.7	81.5	90.0	55.4	79.0	59.8	88.4	84.3	84.7	83.3	50.2	78.0	66.3	86.3	72.0
FRCN OHEM (Shrivastava et al, 2016)	76.3	86.3	85.0	77.0	60.9	59.3	81.9	81.1	91.9	55.8	80.6	**63.0**	90.8	85.1	85.3	80.7	54.9	78.3	70.8	82.8	74.9
ION (Bell et al, 2016)	76.4	88.0	84.6	77.7	63.7	63.6	80.8	80.8	90.9	55.5	81.9	60.9	89.1	84.9	84.2	83.9	53.2	79.8	67.4	84.4	72.9
R-FCN (Dai et al, 2016b)	77.6	86.9	83.4	81.5	63.8	62.4	81.6	81.1	93.1	58.0	83.8	60.8	92.7	86.0	84.6	84.4	59.0	80.8	68.6	86.1	72.9
DP-FCN (Mordan et al, 2017)[1]	80.9	89.3	84.2	**85.4**	**74.4**	70.0	84.0	**86.2**	93.9	**62.9**	85.1	62.7	92.7	87.4	86.0	86.8	**61.3**	85.1	**74.8**	**88.2**	**78.5**
DP-FCN2.0 (ours)[2]	**81.2**	**89.8**	**85.6**	84.7	74.3	**70.8**	**85.1**	85.4	**94.3**	62.6	**86.5**	62.0	**92.8**	**88.4**	**88.0**	**87.4**	61.0	**85.4**	73.7	88.0	78.3

Table 8: **Detailed detection results on PASCAL VOC 2012 test** in average precision (%). For fair comparisons, the table only includes methods trained on PASCAL VOC 07++12.

the same number of regions per image. We include common tricks: color data augmentations (Krizhevsky et al, 2012), bounding box voting (Gidaris and Komodakis, 2015) with a threshold of 0.5 on PASCAL VOC and 0.75 on MS COCO, and averaging of detections between original and flipped images (Bell et al, 2016; Zagoruyko et al, 2016). We set the relative weight of the multi-task (classification/localization) loss (Girshick, 2015) to 7 and enlarge input boxes by a factor 1.3 to include some context.

PASCAL VOC 2007 and 2012. Results of DP-FCN and DP-FCN2.0, along with those of recent methods, are reported in Table 7 for VOC 2007 and in Table 8 for VOC 2012. For fair comparisons we only report results of methods trained on VOC 07+12 and VOC 07++12 respectively, but using additional data, *e.g.* MS COCO images, usually improves results (He et al, 2016; Dai et al, 2016b). DP-FCN achieves 83.1% and 80.9% on these two datasets, yielding large gaps with all competing methods. In particular, DP-FCN outperforms R-FCN (Dai et al, 2016b), the work closest to ours, by significant margins (2.6 and 3.3 points respectively). DP-FCN2.0 yields 83.3% and 81.2% on VOC 2007 and 2012 respectively, which are small additional improvements of 0.2 and 0.3 points with respect to Mordan et al (2017). As studied in Section 4.2, the main improvement of this model lies in the accuracy of localization, which is not reflected here with the official PASCAL VOC metric, *i.e.* mAP@0.5. We note that these results could be further improved with additional common enhancements, *e.g.* multi-scale training and testing (He et al, 2015) or OHEM (Shrivastava et al, 2016).

MS COCO. In order to validate the effectiveness of deformations for object detection, we present the results of DP-FCN, DP-FCN2.0 and other concurrent methods on the chal-

[1] `http://host.robots.ox.ac.uk:8080/anonymous/QNUYVS.html`

[2] `http://host.robots.ox.ac.uk:8080/anonymous/07DMTQ.html`

Method	mAP@ [0.5:0.95]	mAP@ 0.5	mAP@ 0.75	mAP@ Small	mAP@ Medium	mAP@ Large
MultiPath (Zagoruyko et al, 2016) (on val)	31.5	49.6				
R-FCN (Dai et al, 2016b)	31.5	53.2		14.3	35.5	44.2
ION (Bell et al, 2016)	33.1	55.7	34.6	14.5	35.2	47.2
DP-FCN (Mordan et al, 2017)	34.0	54.7	37.2	15.9	36.4	47.5
DP-FCN2.0 (ours)	34.8	54.8	38.4	15.8	37.2	49.0
FPN (Lin et al, 2017a)	36.2	**59.1**		18.2	39.0	48.2
Deformable ConvNet (Dai et al, 2017)	37.5	58.0		19.4	40.1	**52.5**
RetinaNet (Lin et al, 2017b)	**39.1**	**59.1**	**42.3**	**21.8**	**42.7**	50.2

Table 9: **Detection results on MS COCO test-dev** in average precision (%). All methods are trained on the bounding box detection trainval set (except MultiPath which is trained on the 115k train set) and are single model.

lenging and large-scale MS COCO dataset (Lin et al, 2014) in Table 9. While more recent approaches, *e.g.* Feature Pyramid Network (FPN) (Lin et al, 2017a), RetinaNet (Lin et al, 2017b), have better results, we see that DP-FCN is still competitive with the state of the art, showing the generality of our approach. It notably outperforms R-FCN again on this dataset. Again, DP-FCN2.0 yields better results than Mordan et al (2017), with improvements of 0.8 and 1.2 points in the official and mAP@0.75 metrics, which are strict in localization. However, training on this dataset is rather computational expensive, and all the leading methods use heavy GPU resources for that. It allows them to be parameterized directly on MS COCO, while we do it on PASCAL VOC and then transfer selected values, which might be suboptimal. By training longer, tuning hyper-parameters more carefully or by integrating our ideas into newer architectures, *e.g.* FPN (Lin et al, 2017a), we expect higher results.

4.5 Examples of detections

Some example detections of the final DP-FCN model trained on VOC 07+12 data (Section 4.4) on unseen VOC 2007 test images are shown in Figure 11 and Figure 12. We note that DP-FCN can successfully detect objects under simple as well as challenging conditions. The last row of Figure 12 shows some failure cases where some objects are misclassified, although they are accurately localized. Example detections are illustrated in the same way for DP-FCN2.0 in Figure 13 and Figure 14.

5 Conclusion

In this paper, we propose DP-FCN2.0, an extension of our previous work DP-FCN (Mordan et al, 2017). These two models for object detection learn latent deformable part-based representations thanks to two new modules: a deformation part-based RoI pooling layer aligning parts with discriminative elements of objects, thus increasing invariance to local transformations, and a localization refinement module exploiting configurations of parts to accurately identify shapes of objects. These contributions are then naturally integrated within FCNs for high efficiency. In this extension, we further make interactions between parts explicit, so that they are learned by our model. This yields finer representations of objects, and both recognition and localization are improved. This is done by casting alignment as a CRF inference with custom potentials, optimizing all parts jointly, and by using a bilinear deformation-based refinement for localization. Deformations make our models more flexible than traditional region-based detectors, restricted to extract features from generic bounding boxes only. Moreover, this is done without part annotations during training and the joint CRF-based optimization is wrapped within the deformable part-based RoI pooling layer in order to enable end-to-end learning, which makes deformations easy to integrate into any region-based architecture. Finally, experimental validation shows significant gains on the standard PASCAL VOC datasets with several common metrics, and especially with the ones more strict on localization. Our models also achieve state-of-the-art results with VOC data only. However, using deformations with recent state-of-the-art network architectures should boost performance even more.

References

Azizpour H, Laptev I (2012) Object detection using strongly-supervised deformable part models. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV), pp 836–849 3

Bell S, Zitnick L, Bala K, Girshick R (2016) Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3, 4, 5, 15, 16

Ben-Younes H, Cadène R, Thome N, Cord M (2017) MUTAN: Multimodal tucker fusion for visual question an-

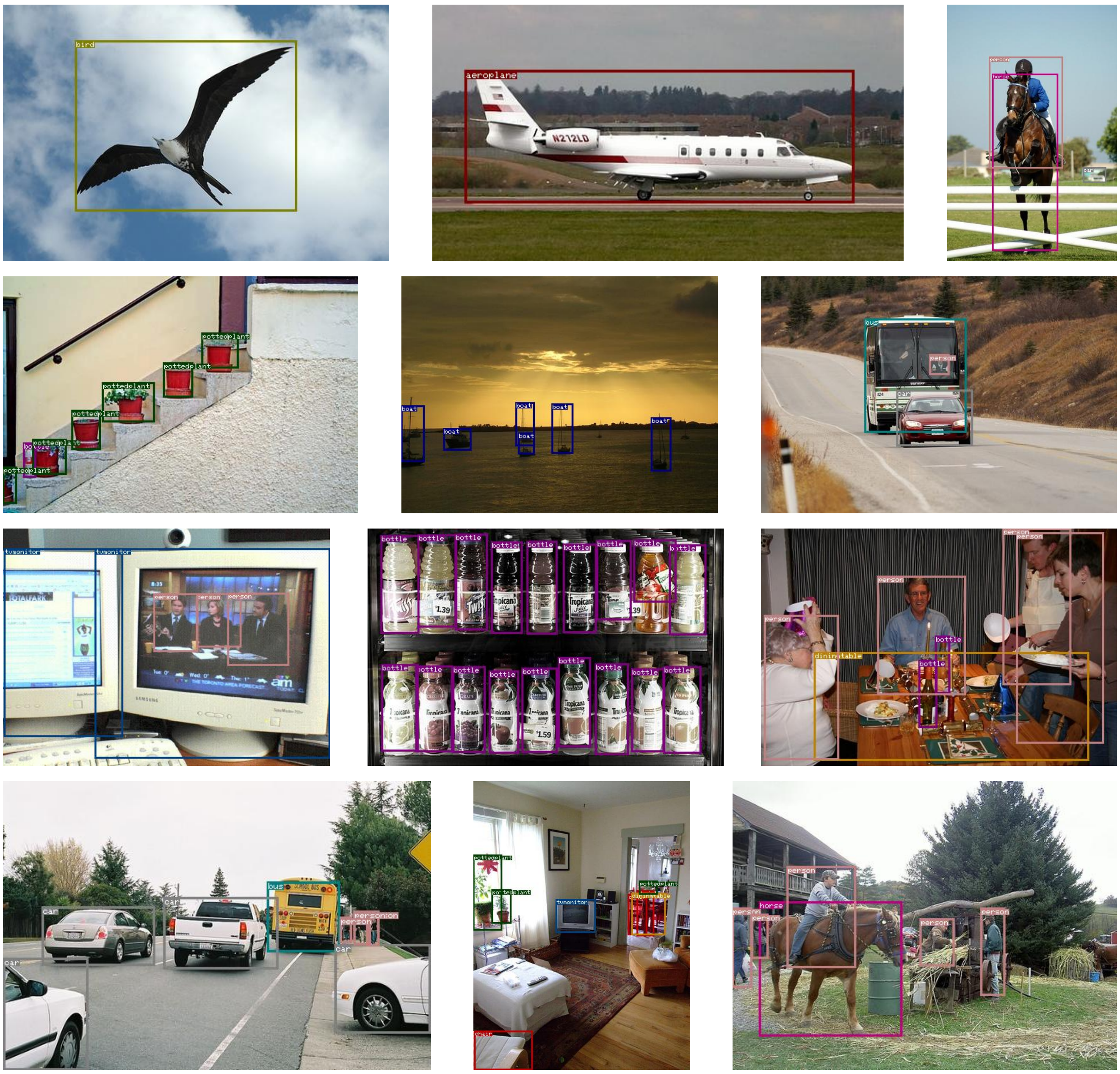

Fig. 11: **Example detections of DP-FCN** trained on VOC 07+12 data (Section 4.4) on unseen VOC 2007 test images, using VOC color code for classes. All detections with score above 0.6 are shown.

swering. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) 10

Chandra S, Usunier N, Kokkinos I (2017) Dense and low-rank gaussian CRFs using deep embeddings. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) 3

Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille A (2015) Semantic image segmentation with deep convolutional nets and fully connected CRFs. In: Proceedings of the International Conference on Learning Representations (ICLR) 1, 3, 4, 5

Dai J, He K, Li Y, Ren S, Sun J (2016a) Instance-sensitive fully convolutional networks. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV), pp 534–549 4

Dai J, Li Y, He K, Sun J (2016b) R-FCN: Object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems (NIPS) 1, 2, 3, 4, 5, 6, 8, 10, 11, 13, 15, 16

Dai J, Qi H, Xiong Y, Li Y, Zhang G, Hu H, Wei Y (2017) Deformable convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision

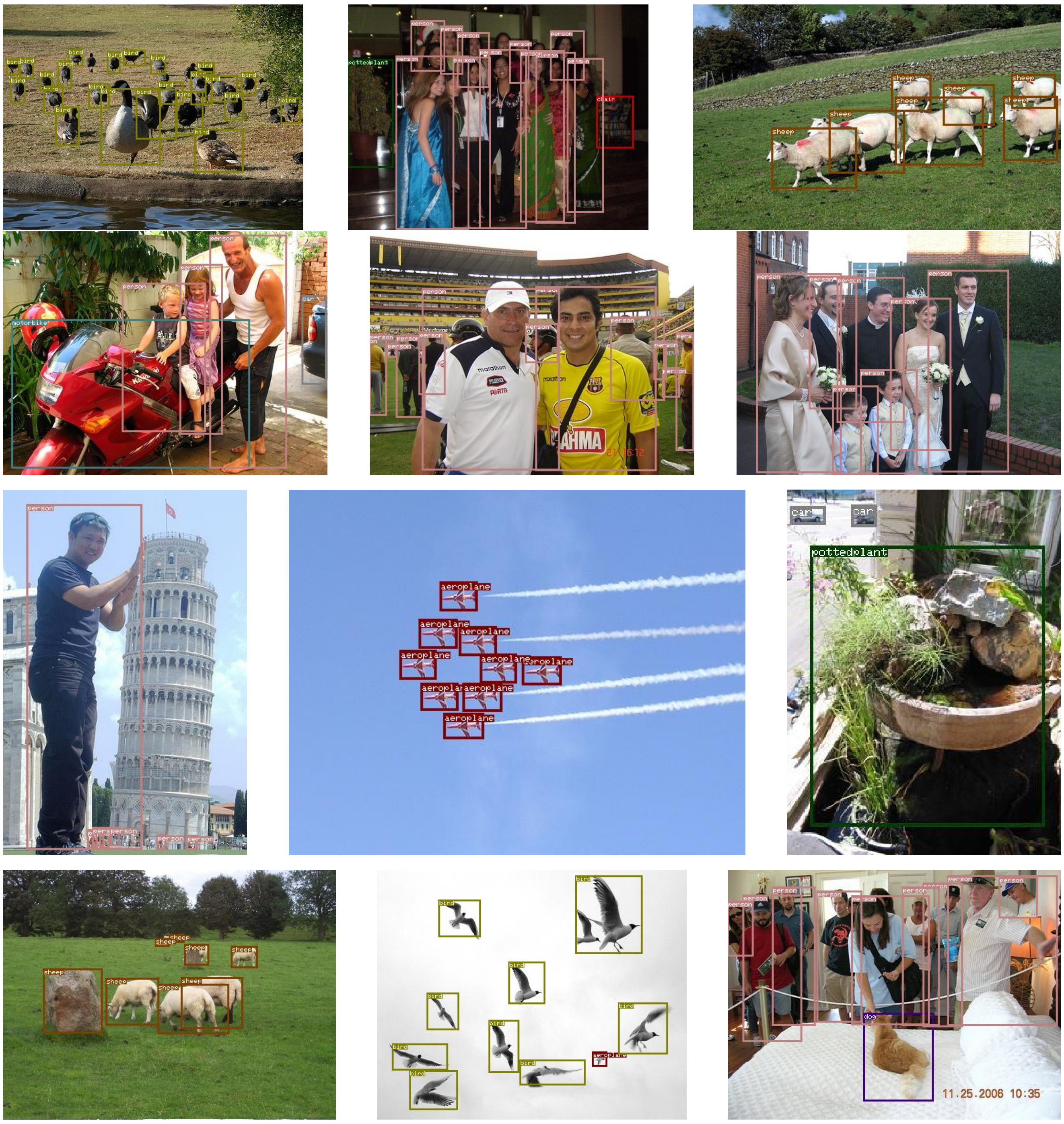

Fig. 12: **Example detections of DP-FCN** trained on VOC 07+12 data (Section 4.4) on unseen VOC 2007 test images, using VOC color code for classes. Last row shows some failure cases. All detections with score above 0.6 are shown.

(ICCV) 4, 15, 16

Durand T, Mordan T, Thome N, Cord M (2017) WILDCAT: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4, 5

Everingham M, Eslami A, Van Gool L, Williams C, Winn J, Zisserman A (2015) The PASCAL visual object classes challenge: A retrospective. International Journal of Computer Vision (IJCV) 111(1):98–136 3, 10

Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained

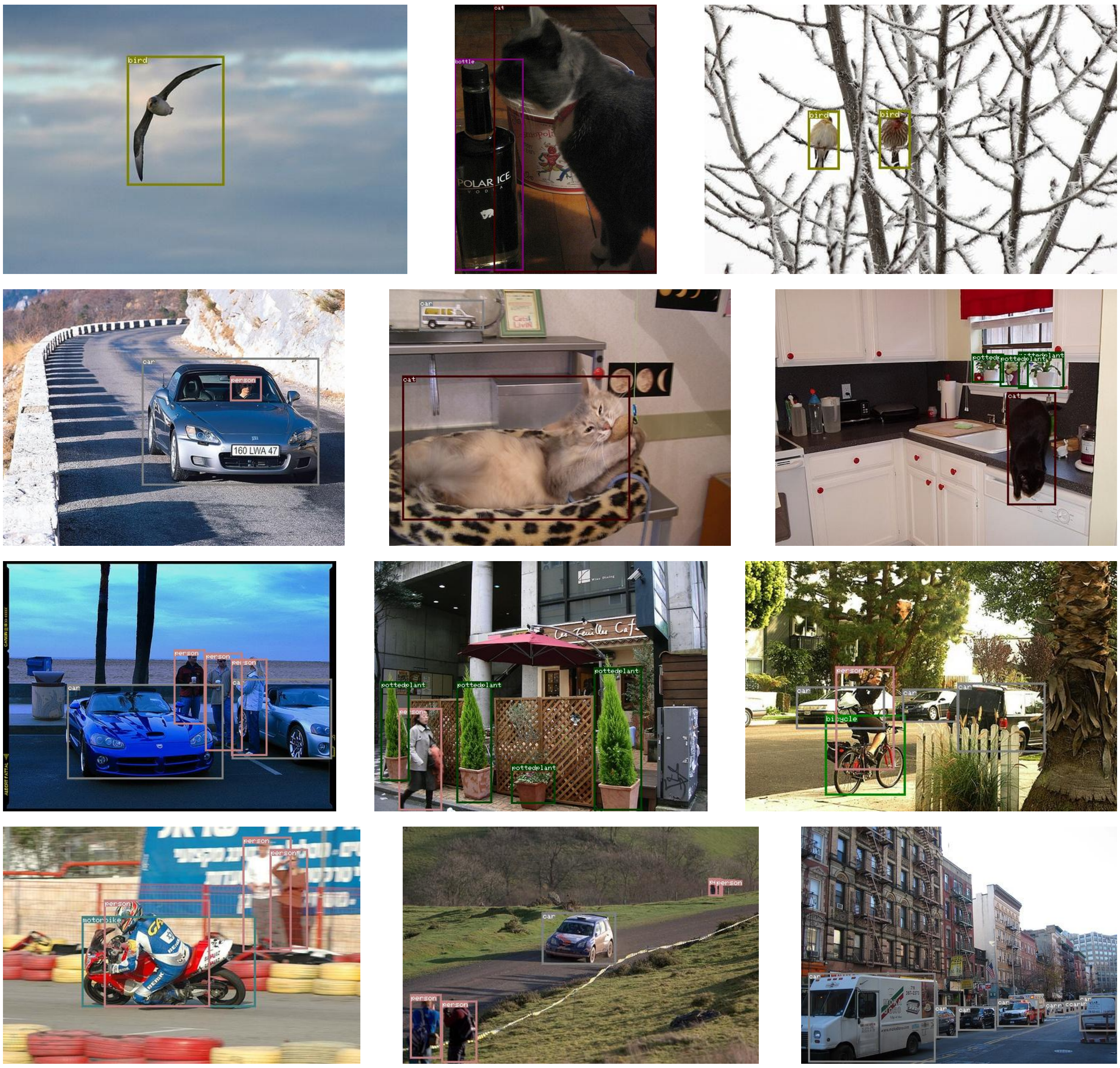

Fig. 13: **Example detections of DP-FCN2.0** trained on VOC 07+12 data (Section 4.4) on unseen VOC 2007 test images, using VOC color code for classes. All detections with score above 0.6 are shown.

part-based models. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 32(9):1627–1645 2, 3, 4, 5, 7, 13

Fidler S, Mottaghi R, Yuille A, Urtasun R (2013) Bottom-up segmentation for top-down detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 3294–3301 3

Gidaris S, Komodakis N (2015) Object detection via a multi-region and semantic segmentation-aware CNN model. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1134–1142 15

Gidaris S, Komodakis N (2016a) Attend refine repeat: Active box proposal generation via in-out localization. In: Proceedings of the British Machine Vision Conference (BMVC) 1, 3, 10

Gidaris S, Komodakis N (2016b) LocNet: Improving localization accuracy for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 10, 15

Girshick R (2015) Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1440–1448 1, 3, 4, 5, 8, 9, 10, 15

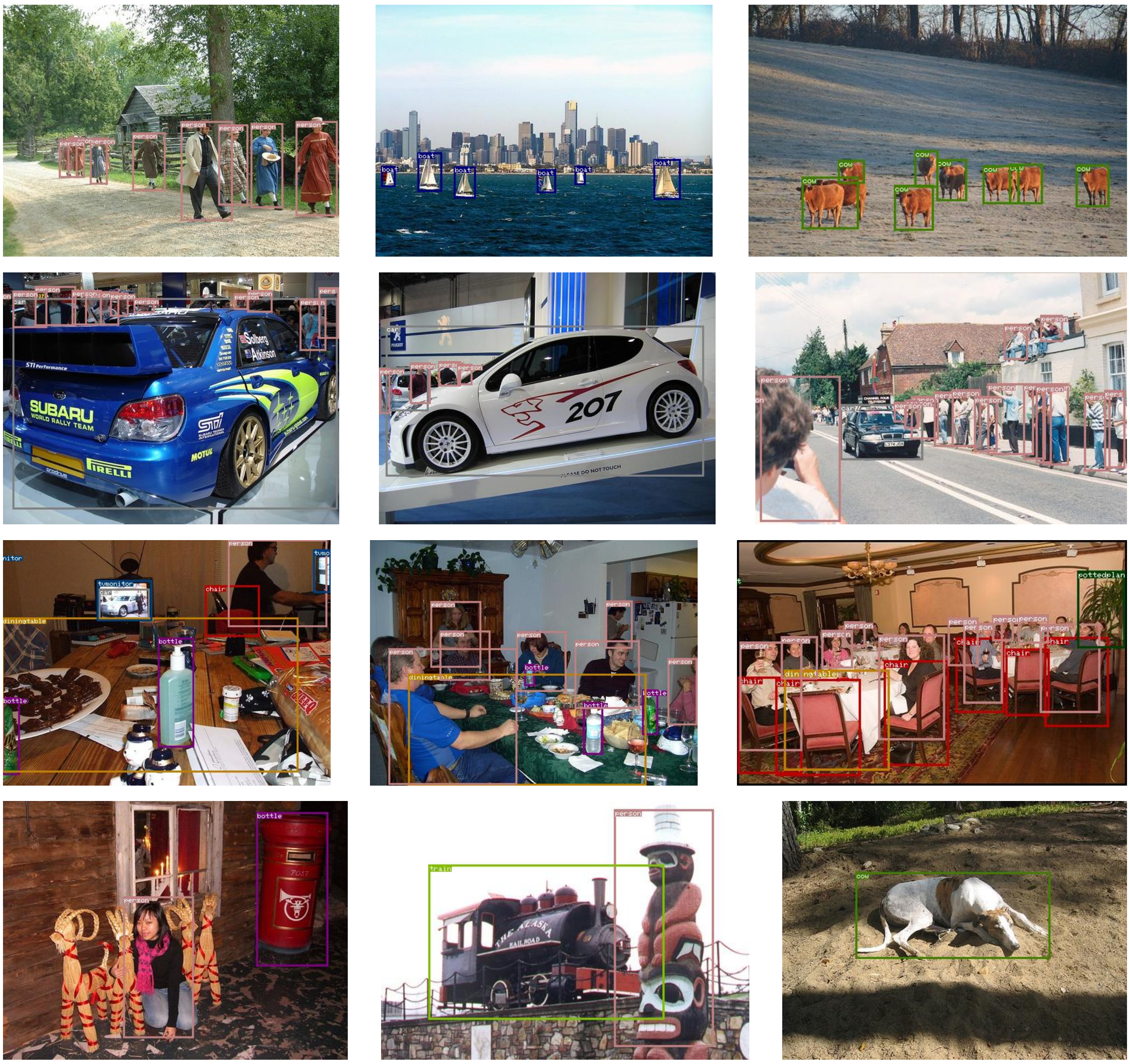

Fig. 14: **Example detections of DP-FCN2.0** trained on VOC 07+12 data (Section 4.4) on unseen VOC 2007 test images, using VOC color code for classes. Last row shows some failure cases. All detections with score above 0.6 are shown.

Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 580–587 1, 2, 3, 9, 10

Girshick R, Iandola F, Darrell T, Malik J (2015) Deformable part models are convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 437–446 3, 4

He K, Zhang X, Ren S, Sun J (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 37(9):1904–1916 15

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1, 2, 4, 5, 10, 14, 15

Kong T, Yao A, Chen Y, Sun F (2016) HyperNet: Towards accurate region proposal generation and joint object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3, 4, 5, 15

Krähenbühl P, Koltun V (2011) Efficient inference in fully connected CRFs with gaussian ddge potentials. In: Advances in Neural Information Processing Systems (NIPS), pp 109–117 3, 7, 8

Krizhevsky A, Sutskever I, Hinton G (2012) ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS), pp 1097–1105 1, 4, 5, 15

Lafferty J, McCallum A, Pereira F (2001) Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proceedings of the International Conference on Machine Learning (ICML) 7

LeCun Y, Boser B, Denker J, Henderson D, Howard R, Hubbard W, Jackel L (1989) Backpropagation applied to handwritten zip code recognition. Neural computation 1(4):541–551 1

Li Y, Qi H, Dai J, Ji X, Wei Y (2017) Fully convolutional instance-aware semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1, 4, 5

Lin D, Shen X, Lu C, Jia J (2015) Deep LAC: Deep localization, alignment and classification for fine-grained recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1666–1674 4

Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick L (2014) Microsoft COCO: Common objects in context. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV), pp 740–755 3, 16

Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017a) Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1, 3, 4, 16

Lin TY, Goyal P, Girshick R, He K, Dollár P (2017b) Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) 3, 16

Liu W, Anguelov D, Erhan D, Szegedy C, Reed S (2016) SSD: Single shot multibox detector. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV) 3, 15

Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 3431–3440 1, 4, 5

Mordan T, Thome N, Cord M, Henaff G (2017) Deformable part-based fully convolutional network for object detection. In: Proceedings of the British Machine Vision Conference (BMVC) 3, 4, 5, 8, 9, 11, 12, 13, 15, 16

Ott P, Everingham M (2011) Shared parts for deformable part-based models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1513–1520 3

Pinheiro P, Lin TY, Collobert R, Dollár P (2016) Learning to refine object segments. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV), pp 75–91 1, 3

Redmon J, Farhadi A (2017) YOLO9000: Better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3

Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3

Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NIPS), pp 91–99 1, 3, 15

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg A, Fei-Fei L (2015) ImageNet large scale visual recognition challenge. International Journal of Computer Vision (IJCV) 115(3):211–252 2, 10

Savalle PA, Tsogkas S, Papandreou G, Kokkinos I (2014) Deformable part models with CNN features. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV), Parts and Attributes Workshop 3, 4

Shrivastava A, Gupta A, Girshick R (2016) Training region-based object detectors with online hard example mining. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 3, 15

Sicre R, Avrithis Y, Kijak E, Jurie F (2017) Unsupervised part learning for visual recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4

Simon M, Rodner E (2015) Neural activation constellations: Unsupervised part model discovery with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1143–1151 4

Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: Proceedings of the International Conference on Learning Representations (ICLR) 1, 5

Tucker L (1966) Some mathematical notes on three-mode factor analysis. Psychometrika 31(3):279–311 9

Wan L, Eigen D, Fergus R (2015) End-to-end integration of a convolution network, deformable parts model and non-maximum suppression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 851–859 3, 4, 5

Wang P, Shen X, Lin Z, Cohen S, Price B, Yuille AL (2015) Joint object and part segmentation using deep learned potentials. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1573–1581 4

Xie S, Girshick R, Dollár P, Tu Z, He K (2017) Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 4, 5, 14, 15

Yu F, Koltun V (2016) Multi-scale context aggregation by dilated convolutions. In: Proceedings of the International Conference on Learning Representations (ICLR) 4, 5

Zagoruyko S, Komodakis N (2016) Wide residual networks. In: Proceedings of the British Machine Vision Conference (BMVC) 4, 5, 14, 15

Zagoruyko S, Lerer A, Lin TY, Pinheiro P, Gross S, Chintala S, Dollar P (2016) A MultiPath network for object detection. In: Proceedings of the British Machine Vision Conference (BMVC) 1, 3, 4, 5, 15, 16

Zhang H, Xu T, Elhoseiny M, Huang X, Zhang S, Elgammal A, Metaxas D (2016) SPDA-CNN: Unifying semantic part detection and abstraction for fine-grained recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1143–1152 4

Zhang N, Donahue J, Girshick R, Darrell T (2014) Part-based R-CNNs for fine-grained category detection. In: Proceedings of the IEEE European Conference on Computer Vision (ECCV), pp 834–849 4

Zheng S, Jayasumana S, Romera-Paredes B, Vineet V, Su Z, Du D, Huang C, Torr P (2015) Conditional random fields as recurrent neural networks. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1529–1537 3, 4

Zhu L, Chen Y, Yuille A, Freeman W (2010) Latent hierarchical structural learning for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1062–1069 3, 5