Deformable Convolution Networks

Arka Sadhu

IIT Bombay

October 18, 2017

Outline

Outline I

Limitations of Convolutional Networks

- CNNs cannot model large unknown transformations because of fixed geometric structures of CNN modules.
- Convolution samples features at fixed locations.
- Region of Interest (RoI) use fixed spatial bins.
- Example : Receptive fields of a convolution layer is the same at all places. This is not desirable at higher layers which encode semantic features rather than spatial features.
- Instead of bounding boxes we would rather want exact boundaries.
- Hence we move on to Deformable Convolutional Networks.

Two New Modules

- Deformable Convolutions : basic idea is to add 2d offset to enable a deformed sampling grid. These offset are also learnt simultaneously along with the convolutional layers.
- Deformable RoI : similar idea. Adds offset to each bin position in the regular bin partitioning.
- Combined to get Deformable Convolutional Networks.
- Authors claim that this can directly replace existing CNN architecture.

Outline I

Simple Convolution to Deformable Convolutions

- Let R denote the set of points which are to be considered for the convolution. In usual convolution of size 3 this R will have $(-1, -1)$ to $(1, 1)$.
- Let input feature map be denoted by x and output feature map denoted by y, and w be in the weights of the convolution filter. For a particular point p_0,

$$y(p_0) = \sum_{p_n \in R} w(p_n) x(p_0 + p_n)$$

.

- For the case of deformable convolutions the new equation will be

$$y(p_0) = \sum_{p_n \in R} w(p_n) x(p_0 + p_n + \Delta p_n)$$

.

Simple Convolution to Deformable Convolutions (Contd.)

- Note: Δp_n can be fractional. To get the value of $x(p_0 + p_n + \Delta p_n)$ bilinear interpolation is used.
- Let G(., .) be the bilinear interpolation kernet. Then for any point p (could be fractional as well)

$$x(p) = \sum_q G(p, q)x(q)$$

- Authors claim that this is easy to compute since G will be non-zero at very small number of qs.

Deformable Convolutions Example

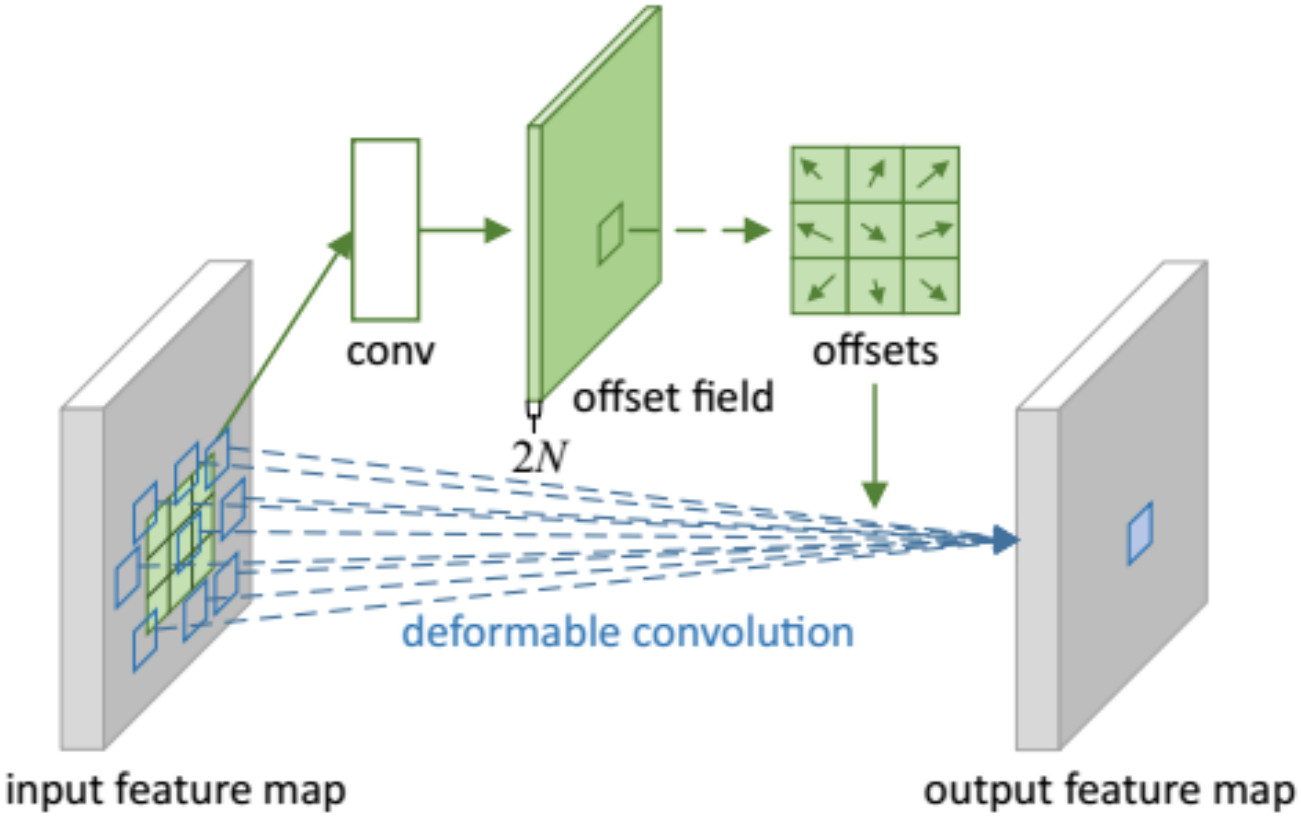


Figure 2: Illustration of 3×3 deformable convolution.

Figure: Deformable Convolution example

Deformable Convolutions Example (contd.)

- As can be seen in 1 offsets are computed by applying a convolutional layer over the same input feature map.
- During training both offsets and convolution kernel are learnt simultaneously.
- The backprop for offsets is given as :

$$\frac{\partial y(p_0)}{\partial \Delta p_n} = \sum_{p_n \in R} w(p_n) \sum_q \frac{\partial G(q, p_0 + p_n + \Delta p_n)}{\partial \Delta p_n} x(q)$$

- The partial derivative of the bilinear interpolation kernel can be calculated from its 1-D version.

Outline I

What is RoI and RoI pooling

- RoI is region of interest. The best example would be a bounding box for an object in an image.
- We would like to work even when this bounding box is not be constrained to rectangular.
- RoI pooling divides the RoI into k by k bins and outputs a feature map y of size k-by-k. This could be max or average pooling or any other kind of pooling. For say (i, j)-th bin with n_{ij} pixels we can have:

$$y(i,j) = \sum_{p \in bin(i,j)} x(p_0 + p)/n_{ij}$$

RoI pooling to Deformable RoI pooling

- For the deformable RoI pooling case we will instead have:

$$y(i,j) = \sum_{p \in bin(i,j)} x(p_0 + p + \Delta p_{ij)})/n_{ij}$$

- Again Δp_{ij} could be fractional and we would use bilinear interpolation.
- The paper introduces the idea of normalized offsets $\Delta \hat{p}_{ij}$ and actual offset is calculated using $\Delta p_{ij} = \gamma * \Delta \hat{p}_{ij} \cdot (w, h)$. This is intuitively required to account for the different k used in the RoI pooling. Emperically γ is set to 0.1
- Extra : Position Sensitive RoI.

Deformable RoI Pooling Example

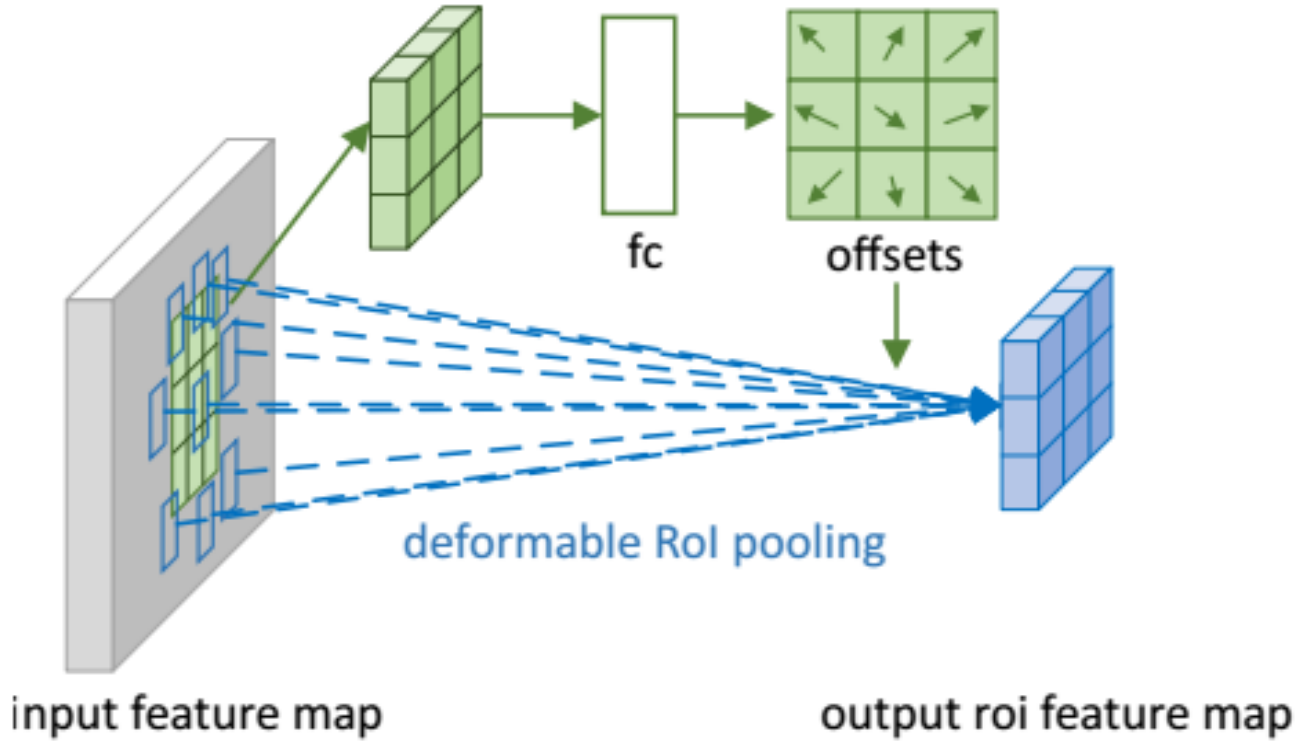


Figure 3: Illustration of 3×3 deformable RoI pooling.

Figure: Deformable RoI pooling Example

Deformable RoI Pooling Example (Contd.)

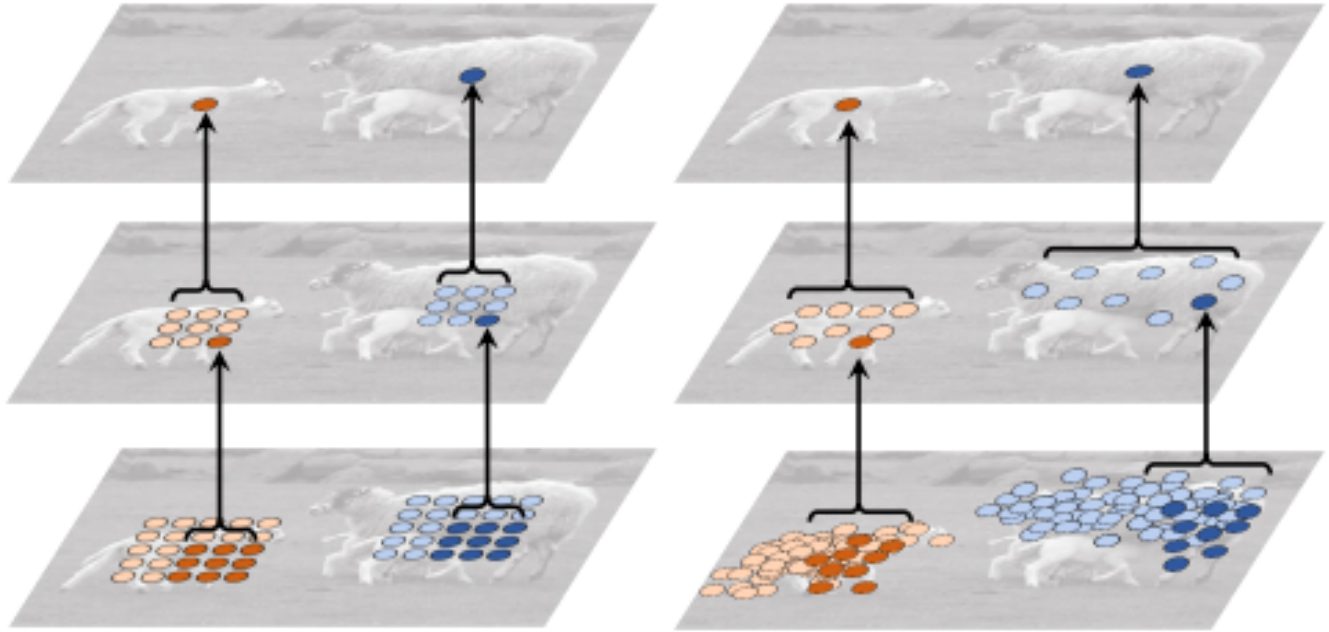
- RoI pooling generates the pooled feature maps which is passed onto a fc layer which generates the normalized offsets which are further converted to the actual offsets.
- Similar backprop works.

Outline I

Deformable Convolutional Networks

- Since both deformable convolution and the deformale roi pooling have same input output structure as that of their counterparts in vanilla CNN, they can readily replace them without affecting the overall network.
- To integrate Deformable conv nets to state of the art CNN architectures two stages are involved.
- First, the cnn generates feature maps over the whole image.
- Second, a task specific network uses the generated feature maps for a specific target.

Deformable Convolutional Network Example Image



(a) standard convolution (b) deformable convolution

Figure 5: Illustration of the fixed receptive field in standard convolution (a) and the adaptive receptive field in deformable convolution (b), using two layers. Top: two activation units on the top feature map, on two objects of different scales and shapes. The activation is from a 3×3 filter. Middle: the sampling locations of the 3×3 filter on the preceding feature map. Another two activation units are

Deformable Convolution Network Motivation

- When the deformable convolution layers are stacked on top of each other, the effect of composited deformation is profound.
- The receptive field and the sampling locations are adaptively adjusted according to the objects scale and shape in deformation. The localization is aspect is enhanced in non-rigid objects specially.
- The paper suggests that using deformable networks gives performance boost in many cases like semantic segmentation, object detection and in general gives baseline improvements to Faster-RCNN as well.