Single Shot Multi Box Detectors (SSD)

Friday, December 7, 2018 4:39 PM

What is SSD?
It is a machine learning algorithm used for the computer vision applications.

What is the purpose?
This algorithm is used to draw bounding boxes around the different objects of interest in an image with vector of values representing probabilities of different class of objects of interest in any particular box.

What is the advantage of SSD?
It is one of the real-time object detection algorithms. It can be run on embedded systems with decent computation capabilities in real time

What is the common pipeline of previous approaches?
Previous approaches involve the pipeline of hypothesizing the bounding boxes, resample pixels or features for each box, and apply the high quality classifier. (e.g. Faster R-CNN)

What is the drawback of the previous approaches that led to the approaches like SSD?
The previous approaches are too computationally intensive for embedded systems, even using high-end hardware, too slow for real-time applications. (e.g. Faster R-CNN operates only at 7 frames per second (FPS))

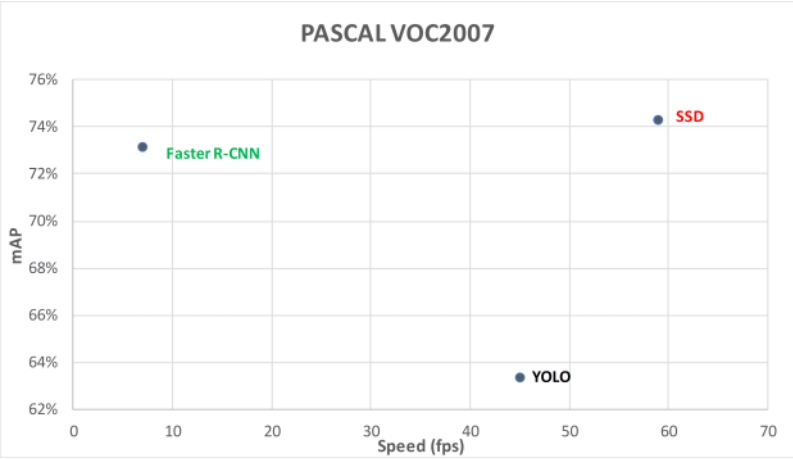
How SSD increases the prediction speed?
Improvement in speed comes from elimination of bounding box proposals and pixel or feature resampling. But this approach is also used by YOLO algorithm.

How it is different in the context of neural network architecture from YOLO ?
SSD discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature (from different depth of convolutional neural network) map location.

How SSD improves the prediction accuracy when compared to similar approaches?
SSD improves the prediction accuracy by using following tweaks

1. Uses small convolutional filter to predict object categories and offsets in bounding box locations, using separate predictors / filters for different aspect ratio detections.
2. Applying these filters to multiple feature maps from the later stages of a network in order to perform detections at multiple scales

What is the evidence that shows the performance of SSD is better than YOLO and Faster R-CNN?
All the three neural networks were tested on PASCAL VOC2007 dataset and the mean average precision (mAP) is plotted below. The algorithm which produces the point on the top right region (*High Processing Speed and High Accuracy*) is desirable and SSD lies in that region.



How the model of the SSD looks like?

SSD approach is based on a feed-forward CNN, which produces a collection of fixed-size bounding boxes and also scores for the presence of object class instances in those boxes, passed through a non-maximum suppression step to produce final detections.

What are the important layers of the SSD and what would they provide?

1. Base Layer:
   The early network layers is part of the standard architecture (truncated before the classification / fully connected layers) used for the high quality image classification. This truncated layer is called as the base layer.

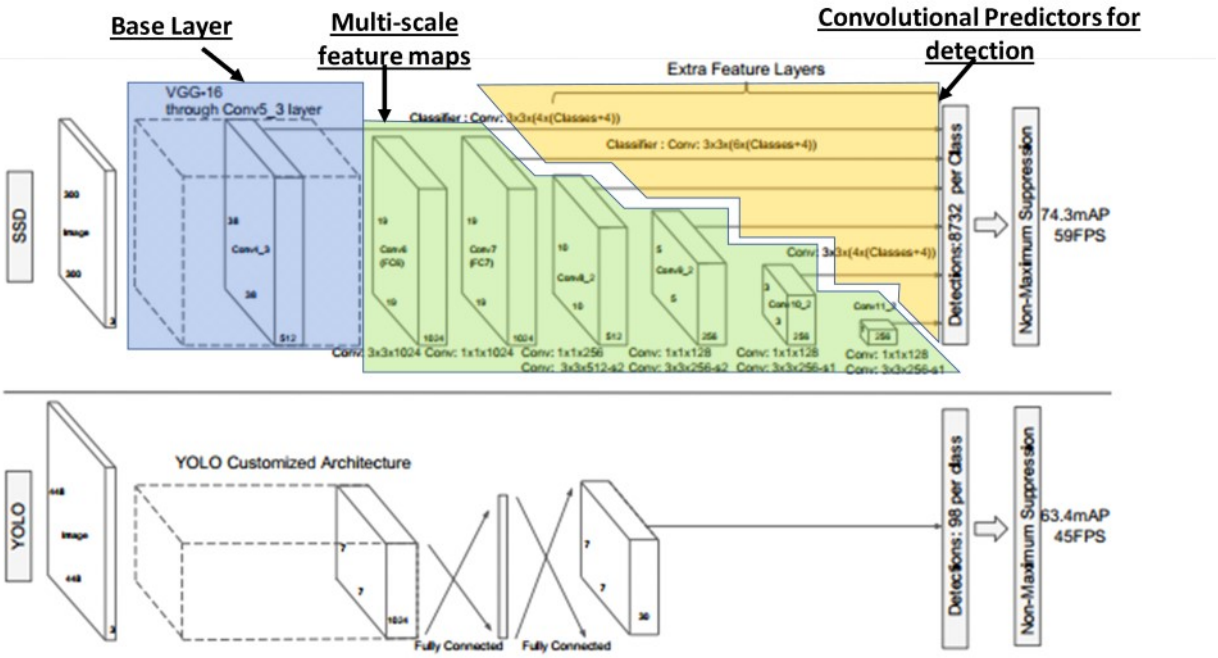
2. Multi-scale feature maps :
   Convolutional feature layers are added to the end of base (truncated) network. These layers decrease in size progressively to allow detections at multiple scales. The convolution model for predicting detections is different for each feature layer. (This approach is not in YOLO, YOLO operates in single scale feature map).

3. Convolutional Predictors for detection:
   Each feature layer of the multi-scale feature maps produce a set of detection predictions using a set of convolutional filters.
   Example: For a feature layer of size m x n with p channels, a small kernel of 3 x 3 x p is applied for each location to produce either a score for a category or a shape offset relative to the default box coordinates.
   Offset values are measured relative to each feature map location.
   (YOLO uses an intermediate fully connected layer instead of a convolutional filter for this step).

4. Default Boxes and aspect ratios:
   Each feature map cell of all feature maps (after the base layer) in the network is associated with a set of default bounding boxes. At each cell the offsets relative to the default box shapes in the cell and the per-class scores (indicate the presence of a class instance) of each default boxes will be predicted. For each box out of 'k' default boxes at a cell, 'c' class scores and the 4

offsets relative to default box shapes have to be computed. So a total of (c+4)k filters have to applied at each location to yield (c+4) x k x m x n outputs for a m x n feature map. This approach of creating different box shapes in several feature maps help to efficiently discretize the shape of possible output box shapes.

Please see the following figures for better understanding



How SSD differs from other approaches in training strategy ?

The ground truth information has to be assigned to specific outputs in the fixed set of detector outputs. This approach is also used in YOLO and also for the region proposal stage for Faster R-CNN and MultiBox.

Once this assignment is determined, loss function and back propagation are applied end-to-end.

1. <u>Matching Strategy:</u>
   It is necessary to determine which default boxes correspond to a ground truth detection. These default boxes vary over location, aspect ratio and scale. In MultiBox algorithm, ground truth box is matched with the default box with the best jaccard overlap. But, in SSD, the ground truth box is matched with default boxes that are having jaccard overlap higher than a threshold (0.5). This approach can make the network to predict high scores for multiple overlapping default boxes rather than requiring it to pick only the one with maximum overlap.

2. <u>Training Objective:</u>
   SSD used the same objective as of MultiBox algorithm but it is extended to handle multiple object categories also.
   Let $x^p_{ij} = \{1, 0\}$ be an indicator for matching the i-th default box to the j-th ground truth box of category p. In the matching strategy above, we can have $\sum_i x^p_{ij} \geq 1$.
   The overall objective loss function is a *weighted sum of the localization loss (loc) and the confidence loss* (conf):

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g))$$

where N is the number of matched default boxes. The weight term 'α' is set to 1 by cross validation. If N=0, then the loss will be set to 0.
The localization loss is a Smooth L1 loss between the predicted box (l) and the ground truth box (g) parameters. The aim is to regress to offsets for the center (cx; cy) of the default bounding box (d) and for its width (w) and height (h)

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$

$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right)$$

where d^{cx}_i , d^{cy}_i, d^{w}_i, d^{h}_i are the **parameters for i[th] default box** in a given cell location

l^{cx}_i , l^{cy}_i, l^{w}_i, l^{h}_i are the **predicted offset parameters for i[th] box** in a given cell location

g^{cx}_j , g^{cy}_j, g^{w}_j, g^{h}_j are the **parameters for the ground truth box** in a given cell location

The confidence loss is the SoftMax loss over multiple classes confidences (c).

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

What is Jaccard Overlap (or) Jaccard index?
Jaccard index, also known as Intersection over Union, is a statistic used for comparing the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

What is Smooth L1 loss?
Smooth L1-loss can be interpreted as a combination of L1-loss and L2-loss. It behaves as L1-loss when the absolute value of the argument is high, and it behaves like L2-loss when the absolute value of the argument is close to zero. The equation is:

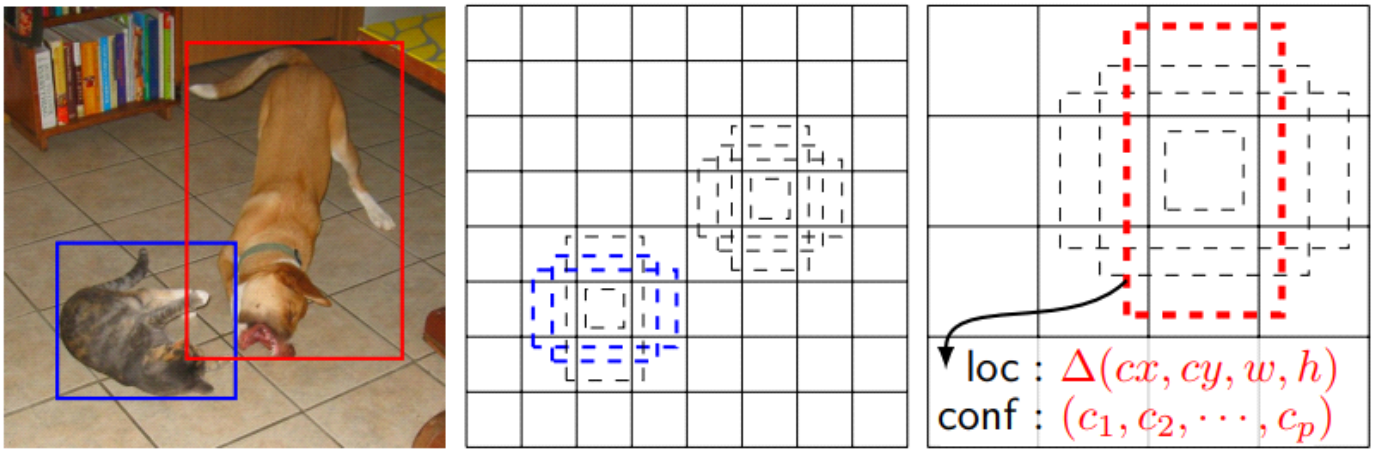
$$L_{1;smooth} = \begin{cases} |x| & \text{if } |x| > \alpha; \\ \frac{1}{|\alpha|}x^2 & \text{if } |x| \leq \alpha \end{cases}$$

α is a hyper-parameter here and is usually taken as 1. 1/α appears near x^2 term to make it

continuous.
Smooth L1-loss combines the advantages of L1-loss (steady gradients for large values of x) and L2-loss (less oscillations during updates when x is small).

How to choose scales and aspect ratios for default boxes in SSD?
In some approaches like Spatial Pyramid Pooling, the image is processed at different sizes to handle different object scales. Instead utilizing feature maps from different layers in a single network for prediction, the same effect can be utilized, with the advantage of sharing the parameters across all object scales. This approach is used for the application of sematic segmentation in approaches like Image Cascade Neural Network (IC-Net).
SSD used both the lower and upper feature maps for detection. In the example below, two feature maps (8 x 8 and 4 x 4) are used in the framework. With small computational overhead, we can use many more feature maps with different scales which ultimately have different receptive field sizes. In SSD, advantageously, there is no need to make the default boxes correspond to the actual receptive fields of each layer (the deviations from the default boxes are predicted).

Example:



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Scales:

1. Tiling of the default boxes are chosen such that the specific feature map is responsive to the particular scales of the object.
2. If we want to use m feature maps for prediction. Scale of the default box for each feature map is computed as:

$$s_k = s_{\text{min}} + \frac{s_{\text{max}} - s_{\text{min}}}{m - 1}(k - 1), \quad k \in [1, m]$$

where S_{min} is 0.2 and S_{max} is 0.9. So the lowest layer has the scale of 0.2 and the highest layer has the scale of 0.9 and all the layers in between are regularly spaced.

Aspect Ratios:

1. Different aspect ratios are assigned for the default boxes in each location at each feature layer

scale.

2. In SSD, the aspect ratios selected are denoted as
   $a_r \in \{1, 2, 3, 1/2, 1/3\}$
3. The width and height of the each box at specific scale s_k, aspect ratio a_r are calculated using the following formula

$$w_k^a = s_k \sqrt{a_r}$$

$$h_k^a = s_k / \sqrt{a_r}$$

4. For aspect ratio of 1, an additional default box of the following scale is selected

$$s'_k = \sqrt{s_k s_{k+1}}$$

   This results in the creation of six boxes for each feature map location.

5. The center of each default box is set to

$$\frac{i+0.5}{|f_k|}, \frac{j+0.5}{|f_k|}$$

   where $|f_k|$ is the size of the k-th square feature map.

In the example image shown above, the dog is matched to a default box in the 4 x 4 feature map, but not to any default boxes in the 8 x 8 feature map. Those boxes in the 8 x 8 feature map have different scales and do not match the dog box and these boxes are considered as negatives in training.
But if you take the cat image, it matches 2 default boxes in the 8 x 8 feature map.

How to match the imbalance between positive and negative examples?
When the number of possible default boxes is large, most of the default boxes are negatives, which creates a significant imbalance between the positive and negative training examples. So, in order to avoid this, the negative examples are sorted using the highest confidence loss for each default box and picked the top ones so that the ratio between negative and positive examples is at most 3:1.

How to make the SSD model more robust to various input object sizes and shapes?
As we already know the neural network can be made more robust by data augmentation.
<u>Data Augmentation:</u>
Each training image is randomly sampled by one of the following options:

1. Use entire original input image.
2. Sample a patch so that the minimum jaccard overlap with the objects is 0.1, 0.3, 0.5, 0.7 or 0.9.
3. Randomly sample a patch

The size of the sample image / patch from the above step can be from [0.1, 1] of the original image size and the aspect ratio between 1/2 and 2. The overlapped part of the ground truth box is kept if the center of it is in the sampled patch. After this step, each sampled patch is resized to fixed size and is horizontally flipped with a probability of 0.5 and some photo-metric distortions are applied.

Reference: https://arxiv.org/abs/1512.02325