A Fast Orientation Invariant Detector Based on the One-stage Method

Jun Yin, Huadong Pan, Hui Su, Zhonggeng Liu and Zhirong Peng

advanced research institute of Zhejiang Dahua Technology Co.Ltd, China

Abstract. We propose an object detection method that predicts the orientation bounding boxes (OBB) to estimate objects locations, scales and orientations based on YOLO (You Only Look Once), which is one of the top detection algorithms performing well both in accuracy and speed. Horizontal bounding boxes(HBB), which are not robust to orientation variances, are used in the existing object detection methods to detect targets. The proposed orientation invariant YOLO (OIYOLO) detector can effectively deal with the bird's eye viewpoint images where the orientation angles of the objects are arbitrary. In order to estimate the rotated angle of objects, we design a new angle loss function. Therefore, the training of OIYOLO forces the network to learn the annotated orientation angle of objects, making OIYOLO orientation invariances. The proposed approach that predicts OBB can be applied in other detection frameworks. In additional, to evaluate the proposed OIYOLO detector, we create an UAV-DAHUA datasets that annotated with objects locations, scales and orientation angles accurately. Extensive experiments conducted on UAV-DAHUA and DOTA datasets demonstrate that OIYOLO achieves state-of-the-art detection performance with high efficiency comparing with the baseline YOLO algorithms.

1 Introduction

Object detection is an important and challenging technology related to computer vision. In general, object detection can be divided into two components: object recognition and object localization. Object recognition, which is also called object identification, involves not only distinguishing foreground targets from background but also assigning these objects the proper object class labels. Object localization is always achieved by regressing positions and scales of targets.

Recently, deep learning has fundamentally changed how computers perform object detection. The current deep learning detectors of state-of-the-art can be divided into two categories: one-stage approaches, including single shot multibox detector(SSD)[1], YOLO[2,3,4], and two-stage methods, including region based convolutional neural network method(R-CNN)[5], Fast R-CNN[6], Faster R-CNN[7], Mask R-CNN[8], region-based fully convolutional network(R-FCN)[9]. In two-stage approaches, proposals are firstly generated by selective search[5,6,10,11] or region proposal network[7], and then classification and regression are operated on them. Although the two-stage methods have been achieving top performances on several objects detection challenging benchmarks, including PASCAL VOC[12] and MSCOCO[13], they are often too slow for real-time applications even if on a high computation capability hardware. In contrast, considering the high efficiency, the one-stage approach attracts much more attention recently.

The past decades have witnessed inspiring progress in general object detection in natural scenes. Most existing object detection methods predict the HBB to locate objects. HBBs, rectangles parameterized by the center point position and scale, have shown great power on general object localization. However, when the viewpoint of the camera moves to the top of the object, predicting HBBs are relatively difficult because of huge variation in orientation of objects. In bird's view images, using HBBs to locate objects have three major defects: Firstly, as shown in Fig.1 (a), there are three viewpoints for one target with different orientations. When they are annotated by HBB, the same targets have different sizes and aspect ratios because of different orientations. Secondly, the discrimination between targets and background becomes difficult, as shown in Fig.1 (b). The given annotated car with the HBB has more background information which is harmful to the trained detector which may have more false positives. Finally, as shown in Fig.1 (c), the rotated objects are distributed densely. On one hand, dense objects annotated with HBBs cause severe occlusions, which is hard to train a more accurate detector. On the other hand, a detector which predicts HBB is hard to locate and count targets. To this end, an orientation invariance object detection method based on YOLOv3[4] (OIYOLO) is proposed, which aims to predict the OBB to locate objects.

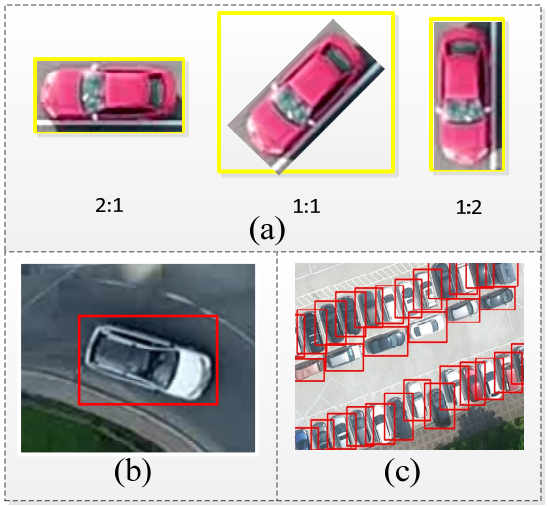

Figure 1. Three major defects caused by using HBBs to predict objects. (a) The same objects with different scales and aspect radios. (b)The background information is introduced in training detectors. (c)Dense objects are difficult to detection and count.

The main contributions of this work are summarized as follows: (1) Based on the one-stage YOLOv3 method, we propose an OIYOLO method which predicts OBB to estimate object locations. (2) We create a UAV-DAHUA dataset using an unmanned aerial vehicle to take 2989 images, including a total of 10224 object instances, which is properly annotated by OBB. This dataset is appropriate for object counting and object detection. (3) In addition, we build a running high-quality convolutional neural network model which runs at 50FPS on a NVIDIA GeForce 1080Ti. (4) Extensive experiments on UAV-DAHUA and DOTA[14] demonstrate that OIYOLO achieves state-of-the-art detection accuracy.

2 Related Work

2.1 Traditional Object Detectors

Early classical object detectors are based on the sliding-window paradigm, which apply the hand-crafted features and classifiers on each sliding widows. Viola and Jones[15] design the Haar feature and employ the AdaBoost algorithm to train a series of cascaded classifiers for face detection. Discriminatively Trained Part-Based Models (DPM)[16] leverage mixtures of scale deformable part models to represent object appearances, which maintaining top results on PASCAL VOC object detection benchmark.

2.2 Two-stage Object Detectors

The two-stage approaches always consist of two parts. The first part generates object proposals, and the second one achieves classification and regression to achieve object detection and location. Girshick[5] employ selective search to generate proposals. Then they apply CNN like Alexnet[17], VGGnet[18], ZFnet[19], Resnet[20]to extract the deep features of object proposals. Finally, they use the support vector machine algorithm to classify objects and apply a repressor to estimate object locations and scales. In [7], He et al. apply region proposal network (RPN) to generate proposals and employ fully connected layers to perform classification and localization, which achieve end-to-end training.

2.3 One-stage Object Detectors

The one-stage approach attracts much more attentions due to high efficiency. In YOLOv1[2], Redmon proposes a YOLO detector that greatly improved the speed by dividing a single image into multiple grids and simultaneously performing localization and classification in each grid. YOLOv2[3] removes the fully connect layers and uses anchor boxes to perform classification and localization. The enhanced version of YOLOv2, which is denoted as YOLOv3[4], employs multiple up-sampling layers to detect objects, which significantly improved both the speed and the accuracy.

Due to high efficiency and accuracy of yolov3 detector, we propose a detection approach based on YOLOv3, which is orientation invariant. The next section explains the OIYOLO algorithm specifically. In section 3 we discuss the experiment details and section 4 shows the experiment results.

3 OIYOLO

3.1 OBB

Although most existing object detectors employ HBBs to perform objects scales and localizations estimation, there are three major defects caused by using HBBs to predict objects as described in section one. To this end, we propose an object detection method that predicts the orientation bounding box (OBB) to estimate objects locations, scales and orientations based on YOLO (You Only Look Once). OBBs have object orientation angles information besides object scales and localizations. As shown in Fig.2(a), the same targets annotated by OBBs have same scales and aspect radios, which is helpful in training object detectors. In Fig.2(b), since OBBs contain less background information than HBBs, our detector trained by OBBs has less false positives comparing with our baseline algorithms YOLO. As shown in Fig2.(c), OBBs can efficiently separate dense objects with no overlapped areas.

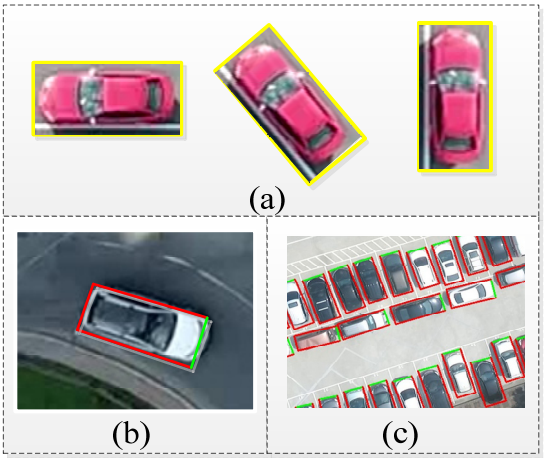

Figure 2. Three major superiorities caused by using OBBs to predict objects. (a) The same objects with same scales and aspect radios. (b)The background information is not introduced in training detectors. (c)Dense objects are easy to detection and count.

3.2 OIYOLO Layer

OIYOLO generates detection results with a convolutional structure, as show in Figure 3. Input images go through several convolutional layers to extract features. The last prediction layer, called OIYOLO layer, realizes object classification and location regression with features having extracted. OIYOLO layer predicts four coordinates, one objectness score and one angle offset information in each anchor box. The number of anchor boxes OIYOLO layer outputs depends on both scales and angle interval we set. If we select anchor boxes with s scales and n classes, setting angle interval σ between 0 degree and 360 degree, total number of anchor boxes is $s*(360/\sigma)$ and total dimension of OIYOLO layer output $s*(360/\sigma)*(4+1+1+n)$.

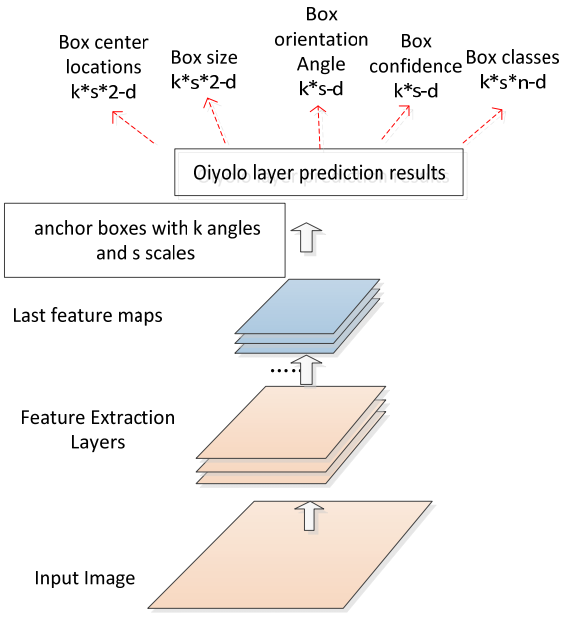

Figure 3. The network structure of OIYOLO

Four coordinates are x, y, w, h coordinate, which are denoted t_x, t_y, t_w, t_h respectively. An input image is divided into a $S * S$ grid in our system. The x and y coordinates represent center offsets of a particular grid cell. We predict t_x and t_y with logistic regression, constraining their number in the range of 0 and 1. The w and h coordinates represent comparison between sizes of predicted boxes and anchor boxes. Angle offset information, denoted t_{angle}, reflects offsets to angle value in particular anchor boxes. Four coordinates and angle offset information for predicted boxes and ground truth boxes are as shown in Eqn. (1) and Eqn. (2) .

By default we set minimum angle value 0, maximum angle value 360, angle interval σ smaller than 45. Then angle value of the n-th anchor box is $(n\text{-}1)*\sigma$. Angle value of particular anchor box will be called anchor angle for short then. We must let difference between predicted angle and anchor angle in the range of $[-\sigma, \sigma]$ to ensure network convergence. So we predict t_{angle} using logistic regression firstly to bound their value fall between 0 and 1 and readjusting range of results $[-\tan(\sigma), \tan(\sigma)]$.

$$
\begin{aligned}
&t_x = (p_x - a_x), t_y = (p_y - a_y) \\
&t_w = \log(\frac{p_w}{a_w}), t_h = \log(\frac{p_h}{a_h}) \\
&t_{angle} = \tan(p_{angle} - a_{angle})
\end{aligned} \tag{1}
$$

$$
\begin{aligned}
&t_x^* = (g_x - a_x), t_y^* = (g_y - a_y) \\
&t_w^* = \log(\frac{g_w}{a_w}), t_h^* = \log(\frac{g_h}{a_h}) \\
&t_{angle}^* = \tan(g_{angle} - a_{angle})
\end{aligned} \tag{2}
$$

In Eqn. (1) and Eqn. (2), p_x, p_y, p_w, p_h, p_{angle} denote center position, width, height and angle of predicted boxes in grid cell, g_x, g_y, g_w, g_h, g_{angle} denote center position, width, height and angle of ground truth bounding boxes in grid cell, and a_x, a_y, a_w, a_h, denote center position, width, height and angle of anchor boxes in grid cell.

Objectness score, denoted t_o, reflects probability one anchor box containing an object on the grid cell. In the methods using HBB, such as Faster-RCNN and YOLO, objectness score value is set according to Intersection-over-Union(IOU) between anchor boxes and predicted boxes during training. IOU between two boxes defines as Eqn. (3).

$$
IOU(B_1, B_2) = \frac{B_1 \cap B_2}{B_1 \cup B_2} \tag{3}
$$

Where $\cap$ means intersection operation of box B_1 and B_2, $\cup$ means union operation of box B_1 and B_2.

However, IOU calculation is more complex when angle of two boxes are different. Because intersection and union of two boxes can be any polygon less than eight sides. So we define new IOU calculation equation as shown in Eqn. (4) which is composed of original IOU function and *delta* function. The IOU function chooses the anchor box whose center position and size close to ground truth box most. The *delta* function chooses the anchor box whose angle close to ground truth box most.

$$delta(\theta_1,\theta_2)=1-\frac{abs(\theta_1-\theta_2)}{\sigma}$$
$$OverLap(B_1,B_2)=IOU(B_1',B_2)*delta(\theta_1,\theta_2) \quad (4)$$

Where $OverLap(B_1, B_2)$ reflects the overlap ratio of box B_1 and B_2, B_1' represents a box whose center position and size the same to B_1 and angle the same to B_2. θ_1 and θ_2 represent the angle of box B_1 and B_2 respectively. σ represents angle interval we set.

3.3 Loss Function

The objective loss function of OIYOLO L is shown in Eqn. (5). We use sum-squared error to optimize because it can equally weights error in larger and smaller boxes and is easy to optimize. The classification loss L_{class} and objectness loss $L_{objectness}$, as shown Eqn. (6), are similar to yolov2. OIYOLO penalizes classification error and regression error only for anchor boxes which are "responsible" for ground truth box. "Responsible" means this anchor box has smallest center distance and largest overlap with one ground truth box. In OBB regression loss function L_{oibox}, we calculate sum-squared error loss of t_x, t_y, t_w, t_h and t_{angle} between predicted OBB and ground truth bounding boxes, as shown in Eqn. (7).

$$L=\alpha L_{class}+\beta L_{objectness}+\gamma L_{oibox} \quad (5)$$

$$L_{objectness}=\frac{1}{2}\sum_{i\in Pos}(t_o^{*i}-t_o^i)^2+\frac{1}{2}\sum_{i\in Neg}(t_o^{*i}-t_o^i)^2 \quad (6)$$

$$L_{oibox}=\sum_{i\in Pos}\sum_j\sum_{m\in\{x,y,w,h,angle\}}\frac{1}{2}(2-g_{w\_r}^j*g_{h\_r}^j)(t_m^i-t_m^{*j})^2 \quad (7)$$

Where $g_{w\_r}^j$ and $g_{h\_r}^j$ denotes relative width and relative height of j-th ground truth box, t_o^*, t_o denotes objectness score of ground truth boxes and predicted boxes respectively. Coefficient $(2-g_{w\_r}^j * g_{h\_r}^j)$ acts as weight to equally weights error in large boxes and small boxes. Coefficients α, β and γ are weights to balance classification, objectness and OBB regression losses.

If *OverLap* function between the anchor box and a ground truth object is larger than any other anchor boxes when training, we set this particular anchor box belong to positive examples, objectness score of this anchor box should be 1. If *OverLap* function between the anchor box and a ground truth object is not the largest but more than some threshold, we ignore this objectness score prediction. If *OverLap* function between the anchor box and any ground truth object is smaller than threshold, we set this particular anchor box belong to negative examples, objectness score of this anchor box is zero. By default we set the threshold 0.5. As shown as Eqn. (8).

$$t_o^*=\begin{cases}1, & if \quad i\in Pos\\ 0, & if \quad i\in Neg\end{cases} \quad (8)$$

4 Experiments and results

4.1 Dataset

Experiments are conducted on UAV-DAHUA dataset and DOTA dataset. UAV-DAHUA dataset has 2989 images from different sensors and platforms. The original size of images in the dataset ranges from about 672*377 to about 1372*941. Each image contains objects exhibiting a wide variety of scales and orientations. The fully annotated UAV-DAHUA images contain 10224 object instances, each of which is labeled by the location of object center, width, height and angle. The range of angle ranges from 0 degree to 360 degrees which demonstrates the orientation of objects. DOTA dataset contains 2806 aerial images and the size of images ranges from about 800*800 to about 4000*4000. The DOTA dataset contains 188282 instances, each of which is labeled by an arbitrary quadrilateral using 15 common object categories.

The method we annotate UAV-DAHUA dataset is θ-based oriented bounding boxes, namely (x_c, y_c, w, h, θ), where (x_c, y_c) represents center location, w, h mean width and height of the oriented bounding boxes, θ denotes angle from horizontal direction of standard HBB. The annotation way DOTA dataset[14] adopted is arbitrary quadrilateral bounding boxes, denoted as $\{(x_i, y_i), i = 1,2,3,4\}$, representing position of OBB's four vertices. It is difficult for us to transform arbitrary quadrilateral bounding boxes into θ-based oriented bounding boxes if the arbitrary quadrilaterals have much difference from rectangles. So we only select large vehicle, small vehicle and plane, whose arbitrary quadrilateral bounding boxes are similar to rectangles , to detect in 15 categories of DOTA dataset.

4.2 Complement details

OIYOLO uses a changed resnet-18 for detection. The first max pooling layer is removed and the last connection layers are replaced by route layers and OIYOLO layers like YOLO does. The network input size is set 832 × 480 in UAV-DAHUA dataset and 1500 × 1500 in DOTA dataset. The feature map of the first OIYOLO layer is down-sampled by 4 times of input size and 3 times for the second OIYOLO layer.

OIYOLO is trained for vehicle detection in UAV-DAHUA dataset and trained for detecting large vehicles, small vehicles and planes in DOTA dataset. We get the size of anchor boxes in OIYOLO layer running k-means clustering and set angles ranging from 0 degree to 315 degree with angle interval 45 degree. We set 3 anchor boxes in UAV-DAHUA dataset and 6 anchor boxes in DOTA dataset because size of objects in DOTA dataset is much different.

4.3 Detection Results

In this section, we compare the performance between OIYOLO and YOLO in bird's eye views. The same convolution architecture and training data argumentation strategies are used in all the detectors.

Figure 4 shows detection results on UAV-DAHUA dataset by using OIYOLO and YOLO. In each detection

box, the green line represents the head of the target. OIYOLO can not only output the locations of the vehicles, but also the orientation of each vechicle. Besides, the method using OIYOLO can successfully detect most of the vehicles and generate less false dismissals than using YOLO.

Table 1 shows precision and recall of vehicle detection results on UVA-DAHUA dataset in the thresholds varies from 0.3 to 0.6, that Table 1(a) and Table 1(b) represent precision and recall of vehicle detection results with YOLO and OIYOLO respectively. Precision and recall of OIYOLO are always larger than that of YOLO in each threshold. OIYOLO has better performance than YOLO in this test.

In Figure 5, we compare detection results between experiments of YOLO and OIYOLO on DOTA dataset. Obviously, location precision of densely packed objects in experiments of YOLO is lower than experiments in OIYOLO. And some of objects are not detected in areas densely crowded with object instances.

Table 1. Precision and recall of vehicle detection results in different thresholds on UAV-DAHUA dataset. Table 1(a) presents precision and recall of vehicle detection results using YOLO. Table 1(b) presents precision and recall of vehicle detection results using OIYOLO.

Threshold	Precision	Recall
0.3	79.23%	76.06%
0.4	84.84%	73.53%
0.5	88.79%	70.56%
0.6	91.07%	68.77%

(a)

Threshold	Precision	Recall
0.3	82.11%	80.15%
0.4	86.28%	76.13%
0.5	89.63%	74.72%
0.6	93.23%	71.32%

(b)

5 Conclusions

Most existing detection algorithms employ the horizontal bounding box (HBB) to perform objects scales and orientations estimation, which is not robust to rotations. In this work, we propose an object detection method that predicts the orientation bounding boxes (OBB) to estimate objects locations, scales and orientations based on YOLO, which is rotation invariant due to its ability of estimating the orientation angles of objects. Our detector enforces the network to learn orientation information of the targets. In addition, the proposed approach that predicts OBB can be applied in other detection frameworks.

Figure 4. Detection results of OIYOLO using OBB and YOLO using HBB on UAV-DAHUA dataset. The first row shows the results of YOLO using HBB. Results of OIYOLO are shown in the second row.

Figure 5. Detection results of OIYOLO using OBB on DOTA dataset. The first row shows the ground truth of DOTA images and the second row shows the results of OIYOLO.

References

1. W. Liu, D. Anguelov, D. Erhan, C.Szegedy, S. Reed, C.Y. Fu, A.C. Berg. Single Shot MultiBox Detector. *European Conference on Computer Vision*, 21-37 (2016)
2. J. Redmon, S. Divvala, R. Girshick, A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 779-788 (2016)
3. J. Redmon, A. Farhadi. YOLO9000: Better, Faster, Stronger. *IEEE Conference on Computer Vision and Pattern Recognition*, 6517-6525 (2016)
4. J. Redmon, A. Farhadi. YOLOv3: An Incremental Improvement. *IEEE Conference on Computer Vision and Pattern Recognition* (2018)
5. R. Girshick, J. Donahue, T. Darrell, J.Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 580-587 (2014)
6. R. Girshick. Fast R_CNN. *Computer science*, 1440-1448(2015)
7. S. Ren, K. He, R. Girshick, J. sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **39**,6(2017)
8. K. He, G. Gkioxari, P. Dollar, R. Girshick. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PP**, 99(2017)
9. J. Dai, Y. li, K. He, J.Sun. Object Detection via Region-based Fully Convolutional Networks. *Neural information processing systems*, 379-387(2016)
10. K. He, X, Zhang, S. Ren, J. Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**,9(2015)
11. K. Sande, J. Uijlings, T. Gevers, A. Smeulders. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE International Journal of Computer Vision*, **104**,2(2013)
12. M. Everingham, LV. Gool, CKI. Williams, J. Winn, A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of computer vision*, 88,2(2010)
13. T. Lin, M. Maire, S. Belongie, J.Hays, P. Perona. Microsoft COCO: Common Objects in Context. *European conference on computer vision*, 740-755(2014)
14. G. Xia, X. Bai, J. Ding, Z. Zhu, S. Belongie, et al. DOTA: A Large-scale Dataset for Object Detection in Aerial Images(2018)
15. P. Viola, M. Jones. Rapid object detection using a boosted cascade of simple features. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*,**1**,2(2001)
16. P. Felzenszwalb , R. Girshick , D. Mcallester. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**,9(2010)
17. A. Krizhevsky, I. Sutskever, G. Hinton, et al. ImageNet Classification with Deep Convolutional Neural Networks. *Neural information processing systems*, 1097-1105(2012)
18. K. Simonyan, A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International conference on learning representations*(2015)

19. M. Zeile, R. Fergus. Visualizing and Understanding Convolutional Networks. *European conference on computer vision*, 818-833(2014)
20. K. He, X. Zhang, S. Ren, J. Sun. Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 770-778(2016)