DLLAB WEEK 6 Pre-Report : You Only Look Once

Hyunsoo Cha[1], Kwanghoon Sohn[2]
[1]Yonsei University. [2]Yonsei University.

1 Abstract

The developer YOLO model also known as researchers implemented a new approach, YOLO, for object detection. Previously, the researchers solved the problem with multitask for object detection, but the researchers overridden it as a regression problem in YOLO. One neural network predicts class probability and bounding box by calculating the entire image only once. The bounding box is a rectangular box that tells the location of an object. The bounding box has the form of wrapping around the object it wants to detect. Class probability is the probability that the bounding box indicates which class an object belongs to. This probability value is presented as a conditional probability. YOLO is an end-to-end pipeline since it is constructed with one neural network. Finally, YOLO has faster processing speed compared to other neural networks. This speed is enough to process a video.

2 Introduction

I summarize the terms prior to the thesis summary. Detection model means to override the classifier and use it as a detector. Classification finds which class the object belongs to. For example, it is to find out whether a picture that humans think is a cat is actually a cat. In object detection, location information is required to determine. Existing models include DPM and R-CNN[4].

DPM is an abbreviation for Deformable Parts Models that detect objects in images by sliding window. R-CNN is a neural network for generating bounding boxes that enclose objects for a given whole image, using region proposal. Create a bounding box using region proposal and classify it by applying a classifier. Post-processing is applied to classified bounding box objects by adjusting, deduplication, and reassigning box scores[5]. Due to the complexity of the process, R-CNN neural networks have relatively slow computational rates. Furthermore, there exist difficulties in optimization. YOLO replaced the previous two neural networks with one regression problem. This means that the image pixels can be computed as a regression problem, the position of the bounding box, class probabilities.

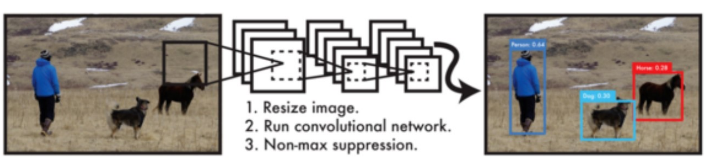

Figure 1: The YOLO described in figure 1 is as follows. The size of the input image is 448×448. This image enters a single convolutional network to find multiple bounding boxes at the same time and calculates the corresponding class probabilities.

Because it is a single neural network, detection performance optimization is easily achieved. The advantages of YOLO are as follows. First, YOLO has a fast computational speed. The speed of YOLO is fast because it has been changed to a regression problem. YOLO also does not need a complex pipeline. In the Titan X GPU written in the paper, 45 frames per second are processed without batch processing. This is the level at which video processing is possible. Second, YOLO simultaneously predicts multiple bounding boxes for the entire image. This is a differentiated approach from DPM[2] and R-CNN. Because it detects the entire image, it also learns the surrounding information. This is a way to reduce background errors. Finally, YOLO achieves high performance when testing pictorial images after natural image learning because it does the learning through generalization of objects. However, despite the three advantages, there are disadvantages. Instead of detecting it quickly, it is less accurate than other models. Accuracy is called mAP.

3 Single Network for Single Detection

For detection, the YOLO neural network truncates the input image to $S \times S$ grid. For confidence scores, grid cells predict from B bounding boxes. For their information, a score that indicates how accurate the bounding box is and exactly contains the object that they want to detect is called confidence score. The following definitions are defined: $Pr(\text{object}) * IOU_{\text{pred}}^{\text{truth}}$

IOU is the same one used in the previous experiment, meaning Intersection over Union. This refers to the intersection of the ground truth bounding box of an object and the predicted bounding box.

$$\text{IOU} = \frac{\text{Real Bounding Box} \cap \text{Predicted Bounding Box}}{\text{Real Bounding Box} \cup \text{Predicted Bounding Box}} \tag{1}$$

$Pr(\text{object}) = 0$ if there is no object in the grid cell. Therefore, confidence score = 0. If there is an object in the grid cell, $Pr(\text{obejct}) = 1$. It is ideal if the confidence score is equal to the IOU. The bounding box consists of x, y, w, h, and confidence. The relative position within the bounding box is represented by $0 \leq (x, y) \leq 1$. The width and height of the bounding box are called $0 \leq (w, h) \leq 1$ when the width and height of the input image are considered as 1. Use this parameter to predict the conditional class probabilities of the grid cell. The equation is as follows: $C(\text{conditional class probabilities}) = Pr(\text{Class}_i|\text{Object})$

Obtain 1 class probabilities per grid cell regardless of the bounding box in the grid cell. For reference, one grid cell predicts B bounding boxes. In the test, the equation (4) can obtain a class specific confidence score.

$$\begin{aligned} &\text{Class specific Confidence Score} \\ &= Pr(\text{Class}_i|\text{Object}) \cdot Pr(\text{Object}) \cdot IOU_{\text{pred}}^{\text{truth}} \\ &= Pr(\text{Class}_i) \cdot IOU_{\text{pred}}^{\text{truth}} \end{aligned} \tag{2}$$

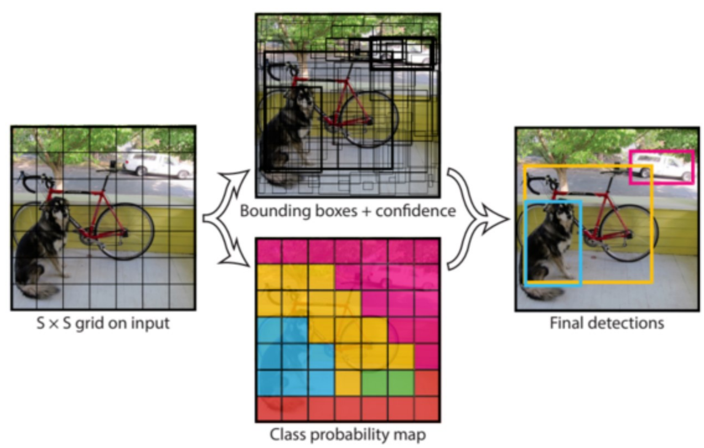

Figure 2: The researchers experimented with the PASCAL VOC dataset[1][1], which was set to $S = 7, B = 5$, with a total of 20 labeling classes present. That is, $C = 20$. Divide one image into seven grids and predict two bounding boxes from one grid cell. The size of the final tensor is $S \times S \times (B * 5 + C)$.

Network Design The layer in front of the YOLO model is the convolutional layer. The fully-connected layer is located behind the model. Perform feature vector of images through convolutional layers, and predict the coordinates of class probabilities and bounding boxes through fully-connected layers. This structure is a form of fine tuning on GoogLeNet[8], consisting of 24 convolutional layers and two fully-connected layers. The difference from GoogLeNet used 1×1 reduction layer and 3×3 convolutional layer[6]. the researchers can see that the final output is $7 \times 7 \times 30$.

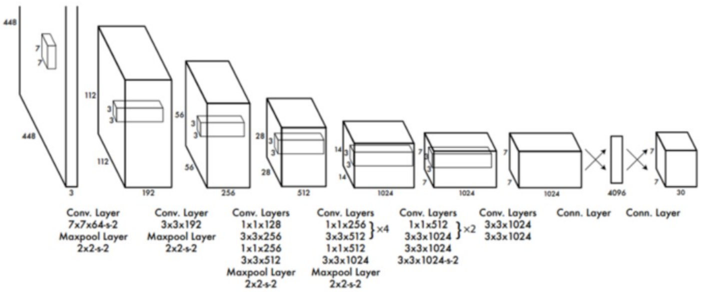

Figure 3: The brief architecture of YOLO model. the input layer should be 448×448. The only difference is the last layers, 1×1 convolutional layers

Training ImageNet dataset has 1000 classes[7] and pretrain the YOLO model's front convolutional layer. A total of 24 convolutional layers exist in the model, but only 20 are used for pretrain, and the rest are the same. This shows an accuracy of 88%. Training and reference were obtained using the Darknet framework. The Darknet framework is a neural network framework that can learn and train neural networks. It was trained using ImageNet for classification, four convolutional layers for object detection, and two fully connected layers. The weight of the layer was randomly initialized, and the resolution of the input image was quadrupled to 448×448.

Through neural networks, the final values are class probabilities and bounding box coordinates. The bounding box coordinates provide a value of x, y, w, h. As previously mentioned, these parameters have been normalized. The linear activation function was positioned at the bottom of the model and the leaky ReLU was applied. The expression of leaky ReLU is as follows.

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases} \tag{3}$$

it use sum-squared error to save loss and optimize this loss. The relationship between raising the mean accuracy, mAP, and optimizing the loss is not clear. The loss and classificaiton loss for the position of the bounding box are calculated separately and are not taught the same way. However, the method of optimizing SSE is calculated by treating the weights of the two losses equally. Another problem is that the confidence score is zero because no object exists within most grid cells. This results in an imbalance in the model. To overcome this, the researchers learned by increasing the weight of grid cells where objects exist, increasing the importance and in the opposite case reducing the weight. This means that the localization loss is greater weight than the classification loss, and the grid cell is also greater weight if there is an object. The parameters at this time are $\lambda_{\text{coord}} = 5, \lambda_{\text{noobj}} = 0.5$.

Other problems with SSE arise due to the size and weight of the bounding box. Regardless of the size of the bounding box, the same weight calculates the loss, but the smaller bounding box is more sensitive to small position changes. In other words, small bounding boxes are more likely to deviate from the object for location changes. To improve this, square root is applied to reduce the weight of the loss by reducing the growth rate. When predicting multiple bounding boxes per grid cell, one bounding box must be matched per object. In other words, it can select only one of several bounding boxes. Also, it can select the bounding box with the largest IOU with the ground-truth bounding box among several predicted bounding boxes. This means the bounding box that best wraps around the object. Training loss function is as follows.

$$\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}{}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} [(\sqrt{\omega_i} - \sqrt{\hat{\omega}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2 \\
&+ \sum_{i=0}^{S^2} I_{ij}^{\text{obj}} \sum_{C \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2
\end{aligned} \tag{4}$$

First, I_i^{obj} indicates whether or not an object exists in the grid cell. 1 if present in grid cell and 0 if not present. Instead of i, the term ij means the jth bounding box precursor of the grid cell. For each of the zero terms, the following is explained: First term: x, y loss calculation; second term: w, h loss calculation (which was square root for the big bounding box); third term: confidence score loss calculation; fourth term: confidence loss calculation when object does not exist; fifth term: conditional class probability loss calculation. λ was used as a balancing parameter. coord is the balance factor for coordinates, and noobj is the parameter of the bounding box according to the presence or absence of objects. The total epoch was set at 135, batch size at 64, momentum at 0.9, and decay at 0.0005. The learning rate rose from 0.001 to 0.01. It was implemented to increase the learning rate little by little and then decrease it again by epoch stage. To prevent overfitting, dropout, data augmentation was applied. The dropout ratio is 0.5. Data augmentation applied random scaling, random translation up to 20% of the original image.

Inference The inference phase also passes only one neural network, the same as the training phase. the researchers predict 98 bounding boxes per image and obtain class probabilities for each bounding box. One disadvantage is that multiple grid cells can detect 1 object simultaneously. This is called a multi-detection problem. This problem can be improved by non-maximal suppression. The mAP was improved by 2-3%.

4 Experiments

The dataset for experiments is PASCAL VOC 2007 and 2012[3]. the researchers compare the performance of YOLO and DPM. Fast YOLO has a mAP of 52.7%. This is twice as accurate as DPM. The result of mAP and FPS shows the relationship between mAP and FPS. Real-time detection in video is possible only when the FPS is 30 or FPS or higher. As a result, the researchers can see that the YOLO family is suitable for real-time detection. YOLO shows that the localization error is quite large, but the correct is small. Instead, it can be seen that YOLO shows a lower error in the background error.

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	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
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [28]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [29]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	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
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [33]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

Figure 4: PASCAL 2012 test results. Fast R-CNN with YOLO is the fourth height mAP. YOLO is the fastest model for real-time detector.

Fast R-CNN shows that the background error is large and YOLO is relatively small, so combining the two models is assumed to give better performance. In other words, the researchers need to identify the similarity of YOLO to the bounding box that R-CNN predicts. For PASCAL VOC 2007 dataset, the Fast R-CNN model recorded 71.8% mAP. The YOLO + Fast R-CNN is 75% mAP. However, this would result in slower computational speed than YOLO. In PASCAL VOC 2012 dataset, the researchers can see that YOLO achieved 57.9% mAP. This is a similar result to the R-CNN in VGG-16. In conclusion, the best model of computational speed is YOLO, and in accuracy, it is a combination of Fast R-CNN and YOLO. In addition, the researchers were able to confirm the superior general-purpose YOLO. In particular, YOLO confirmed its tendency to show the highest accuracy for artwork.

5 Conclusion

In conclusion, YOLO can quickly find objects with one neural network. In addition, it is a model that has outstanding performance in general purpose and is well worth using as a program.

6 Reference

[1] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[2] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[3] Shiry Ginosar, Daniel Haas, Timothy Brown, and Jitendra Malik. Detecting people in cubist art. In *European Conference on Computer Vision*, pages 101–116. Springer, 2014.

[4] RB Girshick. Fast r-cnn. corr, abs/1504.08083, 2015.

[5] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[6] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. corr abs/1312.4400 (2013). *arXiv preprint arXiv:1312.4400*, 2013.

[7] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[8] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. corr abs/1409.4842 (2014). *arXiv preprint arXiv:1409.4842*, 2014.