УДК 004.932.4

OBJECT DETECTION ON IMAGES BASED ON YOLOX

Gao Haoxuan., студент гр.167011 магистрант

Белорусский государственный университет информатики и радиоэлектроники

г. Минск, Республика Беларусь

Шевчук О.Г. – доц, кан. тех. наук

Abstract. The article presents a method of image object detection based on YOLOX. It is shown that how to use the YOLOX model for inference to complete target detection, including image preprocessing, decode outputs of model and NMS algorithm. The test results show that the method can be applied to Android system and client-server scenario.

Keywords*:* object detection, image preprocessing, model inference, NMS algorithm.

1.Introduction

Object detection is a computer vision technique that allows us to identify and locate objects in an image or video. With this kind of identification and localization, object detection can be used to count objects in a scene and determine and track their precise locations, all while accurately labeling them.

Traditional object detection relies on sophisticated manual feature design and extraction, such as Histogram of Oriented Gradient, HOG. In 2012, AlexNet based on deep convolutional neural network (CNN) won the ImageNet competition with a significant advantage. Since then, deep learning has received widespread attention. In recent years, the rapid development of deep learning techniques has led to remarkable breakthroughs. Object detection has now been widely used in many real-world applications, such as autonomous driving, robot vision, video surveillance, etc.

YOLO is an advanced single-stage object detection algorithm, which has undergone the evolution of v1~v4, and has so far developed to YOLOX that does not rely on anchor boxes. In this article, the method of image object detection based on YOLOX is proposed, mainly including image preprocessing, how to decode, and the process of executing NMS. And the test result is shown.

2. Method of image object detection based on YOLOX

The method of image object detection based on YOLOX – image preprocessing, output decoding and the strategy for choosing prediction boxes – is offered. The method can be used for images of any size, and the final target detection result can be obtained.

The flowchart of the method of image object detection based on YOLOX is provided in figure 1.

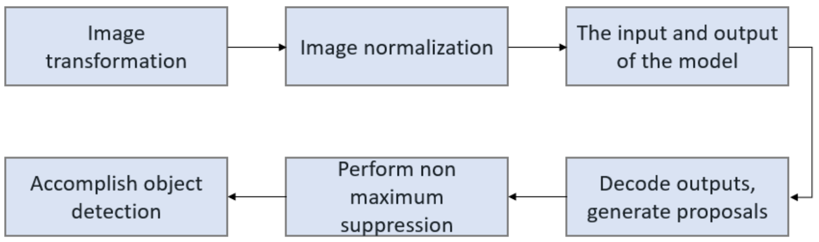


Image transformation
Image normalization
The input and output of the model
Decode outputs, generate proposals
Perform non maximum suppression
Accomplish object detection

Figure 1 Flowchart of the method of image object detection based on YOLOX

1) Image transformation.

For the images of any size images, they need to be transformed to the specified size. Because the size of the input image to the YOLOX model is 640×640 pixels. Apply the affine transformation to the original image to transform its size. According to the affine transformation theory, the scaling factor scale and the transformation matrix M can be obtained:

$$scale = \min(\frac{w'}{w}, \frac{h'}{h})$$

$$M = \begin{pmatrix} \text{scale} & 0 & -\frac{scale \cdot w}{2} + \frac{w'}{2} \\ 0 & \text{scale} & -\frac{scale \cdot h}{2} + \frac{h'}{2} \end{pmatrix}$$

where w and h are the width and height of the original image, w' and h' are the width and height of the target image. Meanwhile, the matrix M^{-1} can be obtained:

$$M^{-1} = \begin{pmatrix} \frac{1}{\text{scale}} & 0 & -\frac{b1}{scale} \\ 0 & \frac{1}{\text{scale}} & -\frac{b2}{scale} \end{pmatrix}$$

$$where\ b1 = -\frac{scale \cdot w}{2} + \frac{w'}{2}, b2 = -\frac{scale \cdot h}{2} + \frac{h'}{2}$$

Through the matrix M, the original image can be transformed easily. And through the matrix M^{-1}, the transformed image can be restored, showed in Figure 2.

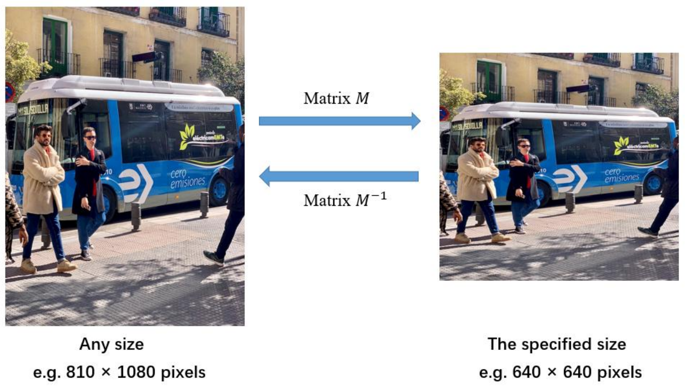


Figure 2 Image transformation

2) Image normalization.

Using the mean and std of Imagenet is a common practice. All models expect input images normalized in the same way, mini-batches of 3-channel RGB images of shape (3×H×W), where H and W are expected to be at least 224. The images have to be loaded in to a range of [0, 1] and then normalized using mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224, 0.225].

3) The input and output of the model.

The original image can be converted into the input image by image transformation and image normalization. The output data of the YOLOX model is the tensor of 8400 ×85 elements, in which 8400 represents 8400 predicted boxes and 85 represents 85 the elements of the same type contained in every predicted box, shown in Figure 3.

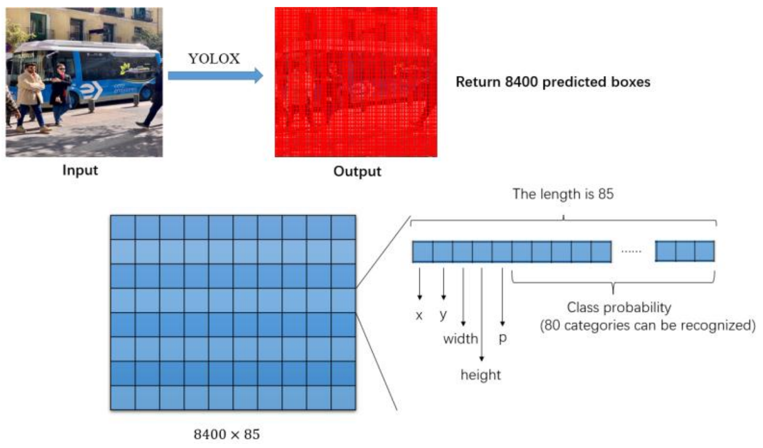


Figure 3 The output of the model

In Figure 3, the output of the model is the tensor of 8400 ×85 elements, meaning that the model predicts 8400 prediction boxes and every box has 85 properties. And x represents the abscissa of the center point of the prediction box, y represents the ordinate of the center point of the prediction box. Width represents the width of the prediction box. Height represents the height of the prediction box. P represents the probability that there is an object in the predicted box. Class probability represents the probability of being that class.

4) Decode outputs, generate proposals.

For an image, there are often only a few objects that need to be identified. However, there are 8400 boxes based on the output data. Obviously, the prediction boxes need to be further selected. In this step, it is necessary to manually set threshold to generate proposals with more accurate prediction boxes.

Before manually setting the threshold, it is necessary to know that the output 8400 prediction boxes are merged from three different sizes of anchors (shown in Figure 4). Because in the decoding process, the coordinates of the prediction boxes are related to the corresponding anchor.

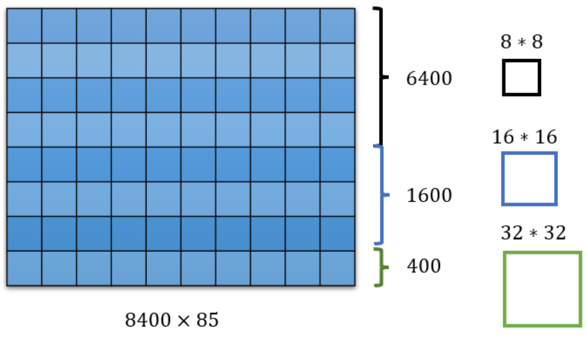


Figure 4 Three different sizes of anchors

In Figure 4, among the 8400 prediction boxes, the anchor size corresponding to 6400 prediction boxes is 8*8. This means that the original image of size 640*640 is divided into 80*80 anchors with stride of 8. Similarly, the anchor size corresponding to 1600 prediction boxes is 16*16, and the anchor size corresponding to 400 prediction boxes is 32*32.

Through the anchor, the coordinates of the prediction box can be obtained, and the probability threshold is further set manually. Here the probability threshold is set to 0.6. Then the confidence C that each prediction box has the corresponding category of objects can be given:

$$\mathrm{C} = p * \max(class\ \text{probability})$$

where p represents the probability that there is an object in the predicted box. Class probability represents the probability of being that class (shown in Figure 3). Then if the confidence C is greater than 0.6, save the proposal (the coordinates of the prediction box) and the corresponding the confidence

C. After generating the proposals, the possibly correct prediction boxes will be saved, shown in Figure 5.

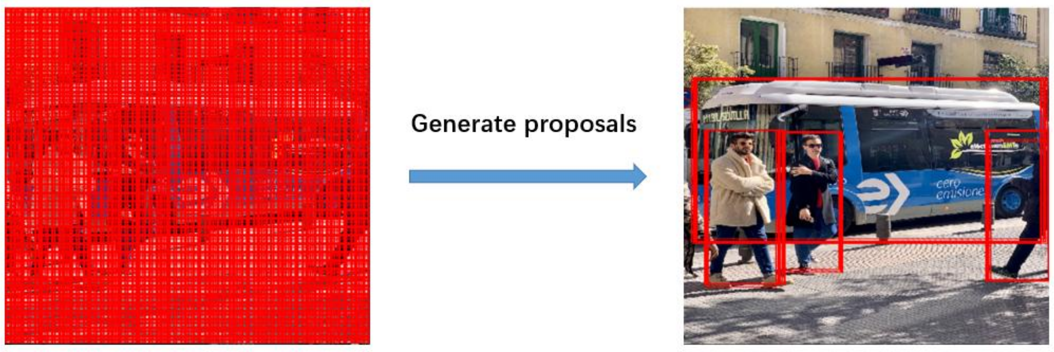


Figure 5 Generate proposals

5) Perform non maximum suppression.

The purpose of performing the algorithm is to eliminate redundant overlapping boxes with lower confidences. The algorithm steps are as follows:

Step 1: Start

Step 2: Given the proposal list, the confidence list, nms threshold that equals

0.6 and the empty list D.

Step 3: Select the proposal with highest confidence using the proposal list and the confidence list, remove it from the proposal list and add it to the list D.

Step 4: Compare this proposal with all the proposals in the proposal list, then calculate the IOU (Intersection over Union) of this proposal with every other proposal. If the IOU is greater than the nms threshold, remove that proposal from boxes.

Step 5: If there are elements in the proposal list

Go to Step 3

Step 6: End

In the end, the list D can be obtained, which contains the coordinate information of the final prediction boxes, which contains the objects to be detected, shown in Figure 6.

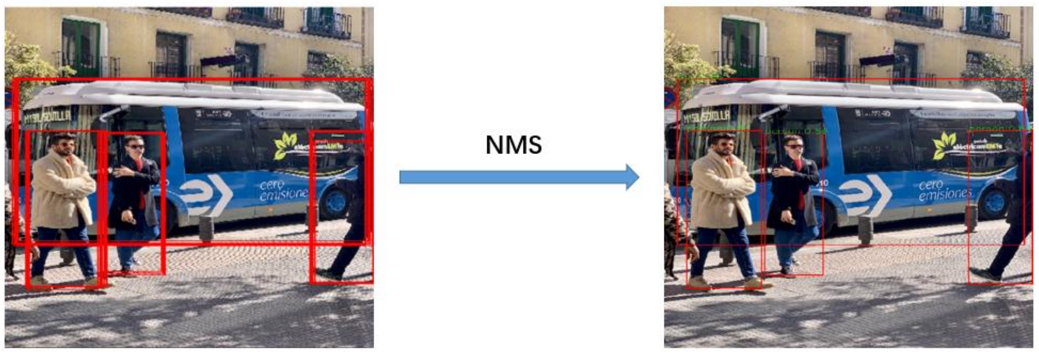


Figure 6 Perform NMS

6) Accomplish object detection

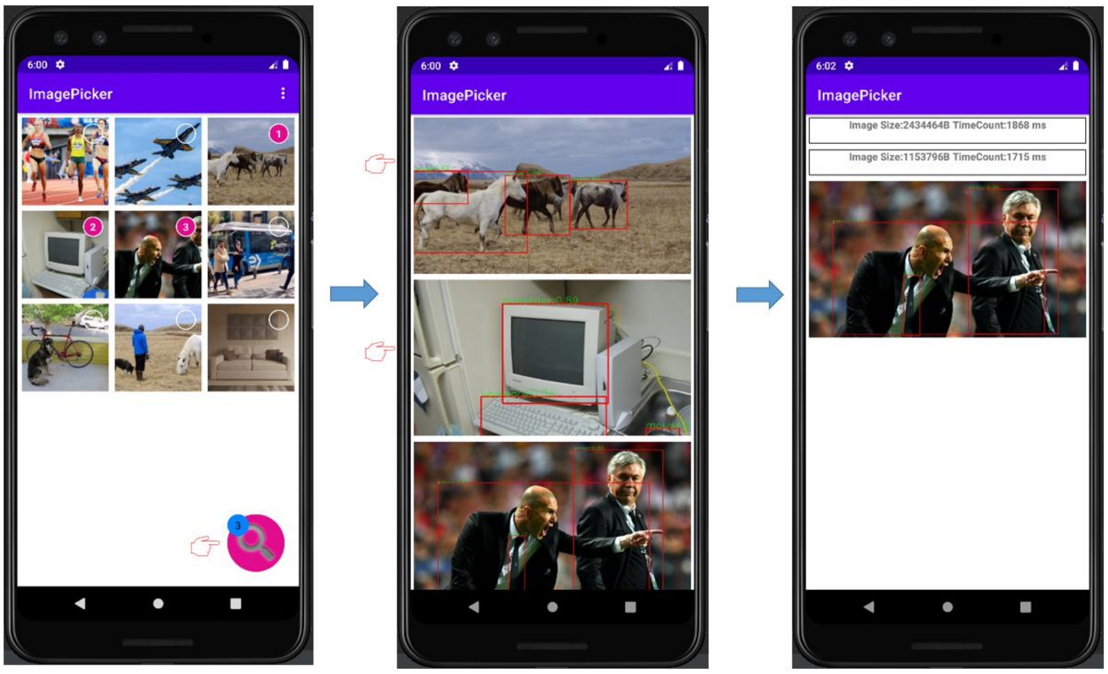
Based on the matrix M^{-1} from Image transformation (Figure 2), the coordinates of the final boxes can be converted to the coordinates in the original image. It accomplishes the object detection on the original image.

3. Test Result

The developed method is implemented in C ++ using library of OpenCV 4.5.4. The experiment was performed on a computer with the following specifications: Processor – Intel(R) Core (TM) i7-9750H CPU @ 2.60GHz 2.59 GHz; RAM – 16 GB; type of system - 64-bit operating system Windows 10.

Examples of test result are shown in Figure 7 and Figure 8.

In Figure 7, apply the method of image object detection to Android system, and the YOLOX model is also installed in Android system. Images in the file storage can be accessed and can be selected to preform object detection. And the size of image and the time of preforming object detection method are given.



```
Image Size:2434464B TimeCount:1868 ms
Image Size:1153796B TimeCount:1715 ms
```

Figure 7 Test Result in Android system

In Figure 8, the test results in server-side and client-side scenario are shown. Client can select multiple images locally and upload them to the server. The server processes the request and returns the image information, the time of preforming object detection method and the result of object detection.

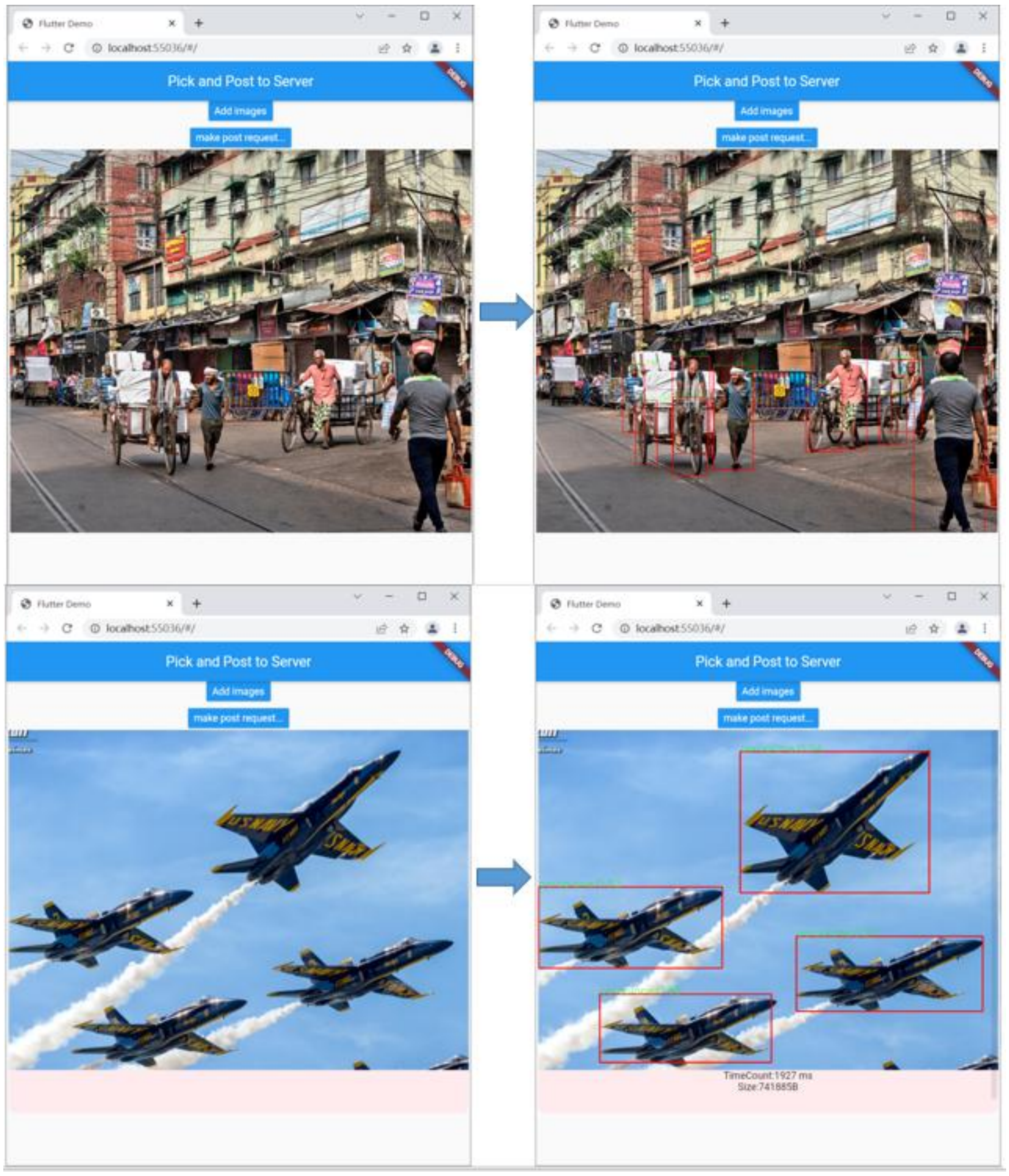


Figure 8 Test Result in C/S scenario

4. Conclusion

The proposed method mainly describes how to apply the YOLOX model for inference. For image preprocessing, affine transformation is applied, and image normalization is described. The problem of how to decode the output is analyzed, and the specific steps to perform NMS are given. Then the object detection on images is implemented. The test is carried out under the Android system and C/S scenarios, and the result is that the proposed method can effectively complete the object detection task.

References

1. Redmon J , Divvala S , Girshick R , et al. You Only Look Once: Unified, Real-Time Object Detection[J]. IEEE, 2016.
2. Redmon J , Farhadi A . YOLOv3: An Incremental Improvement[J]. arXiv e-prints, 2018.
3. Ge Z , Liu S , Wang F , et al. YOLOX: Exceeding YOLO Series in 2021[J]. 2021.
4. Neubeck A , Gool L . Efficient Non-Maximum Suppression[C]// International Conference on Pattern Recognition. IEEE Computer Society, 2006.
5. Stearns C C , Kannappan K . Method for 2-D affine transformation of images: US, US5475803 A[P]. 1995.