ISSN: 2229-6956 (ONLINE)
DOI: 10.21917/ijsc.2022.0388

OBJECT DETECTION USING SEMI SUPERVISED LEARNING METHODS

Shymala Gowri Selvaganapathy[1], N. Hema Priya[2], P.D. Rathika[3] and K. Venkatachalam[4]
[1,2]Department of Information Technology, PSG College of Technology, India
[3]Department of Robotics and Automation Engineering, PSG College of Technology, India
[4]Depatment of Applied Cybernetics, University of Hradec Kralove, Czech Republic

Abstract

Object detection is used to identify objects in real time using some deep learning algorithms. In this work, wheat plant data set around the world is collected to study the wheat heads. Using global data, a common solution for measuring the amount and size of wheat heads is formulated. YOLO V3 (You Look Only Once Version 3) and Faster RCNN is a real time object detection algorithm which is used to identify objects in videos and images. The global wheat detection dataset is used for the prediction which contains 3000+ training images and few test images with csv files which have information about the ground box labels of the images. To build a data pipeline for the model Tensorflow data API or Keras Data Generators is used.

Keywords:

Deep Learning Algorithms, YOLO V3, Faster RCNN, Tensorflow data APIS, Keras Data Generators

1. INTRODUCTION

Object detection is one of the widely used applications of computer vision apart from recognizing images and segmentation. images and videos are analysed in many situations using these methods.

Object detection is also used to localize or locate an object in an image by drawing a rectangular box around it, called as the bounding box. This technology is far different from image recognition, but mostly misunderstood. Image recognition gives a label for an object but object detection has the capacity to locate the object too, while image segmentation is used to comprehend the constituents of a scene at pixel level.

For example, if there is an image with a dog in it, the label dog is created by image recognition. The same label will be created even if there are two dogs in the image. But object detection precisely draws a box around each dog and labels it dog. The precision of course is depending on the model used.

Xu [1] proposed two strategies in contrast to the already existing complex methods of many categories. This article presents the methods of Semi-supervised object detection approaches. One of them is a soft teacher mechanism where a classification score generated by the teacher network is used to weigh the classification loss of each unlabeled bounding box. Added to that is a method to select confident pseudo boxes for regression using a box jitter. An exploratory analysis on COCO dataset with an end-to-end training develops an accurate pseudo label.

Tang [2] proposed a semi-supervised approach with a dual model framework for contemporary object detectors. This method is featured with a dynamic measuring strategy for student-online rehabilitation and used multiple regional proposals and soft mock labels as purposes for student training. Integration of straightforward and lightweight data was highlighted so that the teacher can produce more reliable labels. The recent methodology STAC is compared. STAC uses hard pseudo samples by sparse selection and hard samples. The article states that the performance is improved by 0.64% AP. Made an analysis on MS- COCO dataset

Na Zhao [3] proposed a solution that points to cloud- based 3D object detection to estimate the object type and bounding box for the scene. This paper presents the Self-Ensemble Semi-Supervised 3D Object Detection approach. The highlights included three fixed losses to force the correlation between the two predictive sets of 3D object prediction, to facilitate structural reading and semantic dynamics of objects. The problem of requirement of a huge number of solid labels is eliminated. Exploratory data set analysis done using SUN RGB-D and ScanNet datasets

Sohn [4] proposed a Simple semi-supervised learning framework for object detection, an effective framework of SSL for virtual retrieval and data augmentation strategy. The proposed framework [4] is amenable to many variations, including usage of soft labels for classification loss, other detector frameworks than Faster RCNN, and other data augmentation strategies. Analyzed MS-COCO and VOC07 datasets in this framework.

Jeong [5] proposed a method that reiterated the importance of annotating the dataset precisely to improve the performance of object detection. A Semi-supervised acquisition (CSD)-based learning approach is a way to use compliance barriers as a tool to improve acquisition performance with an available non-labeled data. To guarantee the performance of a model, annotated dataset with more images is mandatory. Also, it is much more difficult, costly and time consuming to place bounding boxes for all objects. The model alleviates the time-consuming problem of putting bounding boxes on the unlabeled data. Background Elimination (BE) is also proposed to avoid the negative impact of the dominant domains on acquisition performance. The PASCAL VOC and MSCOCO dataset was explored in this article. CSD is evaluated in single-stage and two-stage detectors and the results highlight the advantages of the method.

Ponnusamy [6] proposed a method for YOLO Object Detection with OpenCV and Python which used a OpenCV DNN module with a pre-trained YOLO V6 model to perform object detection. Highlights include the removed weak detections in COCO dataset. This model can be used for custom object detection out of the COCO dataset i.e., other than the pre- trained ones. Also, the proposed method included more on object detection including SSD and Faster RCNN.

Humble Teachers Teach the Best Students to Get Internally Guided Items used a MS-COCO dataset. In this article Consistent with the findings in FixMatch, the combination of random add-ons really damages final prediction performance. Self-Ensemble Semi-Supervised 3D Object Detection used SUN RGB-D and

ScanNet datasets. The model requires a completely labeled dataset which is considered as gap. A Simple Semi-Supervised Learning Framework for Object Detection used MS-COCO and VOC07 datasets. In this paper STAC demonstrates an impressive performance gain already without taking confirmation bias issue into account, it could be problematic when using a detection framework with a stronger form of hard negative mining because noisy pseudo labels can be overly-used. Consistency-based Semi-Supervised Learning for Object Detection used PASCAL VOC and MSCOCO datasets. The distribution of unlabeled data and labeled data are equally considered.

2. DATASET AND PREPROCESSING

2.1 DATASET DESCRIPTION

A large dataset of images of wheat fields is collected from locations across the world. Most of the images will have a wheat head present on it. But to create a negative data, some images will not have a wheat head in it. Some images are annotated and have bounding boxes on them. A CSV file containing the range of the bounding box and width and height of the image is also created. The Image ID in the CSV file and the file name of the image is made to match. About 3000+ training images and 10 test samples are used with 25% of the data being used for visualization and refining bounding boxes.

2.2 FEATURE EXTRACTION

The proposed network model considers the complete image and all the objects in the scene. An attempt is made to draw a bounding box for all objects at one look, thanks to YOLO model used. As the first step, the image is divided into $S*S$ grids and each grid is analysed for presence of an object. If true, the centre of the object is identified. If the centre of the object falls inside a grid, then that grid is made responsible to detect the complete object.

The coordinates (x, y), height and width (w, h), and confidence for each bounding box have to be predicted. The centroid is calculated from the grid cell limits. Confidence is a factor of the ground truth of the box drawn, i.e., the intersection of union (IOU), between the projected box and any ground truth box. The conditional class probabilities based on the grid cell in which an object is located, $\Pr(Class_i|Object)$ is calculated as C. The value of C does not depend on the number of boxes B. Only one set of class probabilities per grid cell is anticipated. The conditional class probabilities are multiplied with individual box confidence predictions,

$$Pr(Class_i|Object)*Pr(Object)*IOU = Pr(Classi)*IOU$$

to get the class-specific confidence scores for each box. These ratings represent the likelihood of that class being in the box as well as the accuracy with which the projected box matches the object.

3. PROPOSED SYSTEM

3.1 OBJECT DETECTION

The Global detection datasets are taken and the images are converted into YOLO Shape and bounding boxes are cleaned and fed into the YOLO- based Convolutional Neural model.

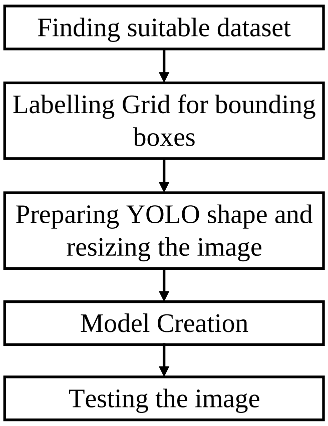
Finding suitable dataset
Labelling Grid for bounding boxes
Preparing YOLO shape and resizing the image
Model Creation
Testing the image

Fig.1. Block diagram

3.2 BOUNDING BOXES

A bounding box is an imaginary rectangle that serves as a reference point for object detection and generates a collision box for that object. These rectangles are drawn over images by data annotators, who define the X and Y coordinates of the object of interest within each image. This helps machine learning algorithms identify what they are looking for, determine collision pathways, and saves computational resources. In deep learning, bounding boxes are one of the most often used picture annotation approaches. This method can save money and improve annotation efficiency when compared to other image processing methods.

For object detection, the computer needs to know what an object is and where it is in order to detect it in an image. For example, self-driving cars. Other vehicles will be labelled and boundary boxes will be drawn around them by an annotator. This aids in the training of an algorithm to recognise different types of automobiles. Autonomous vehicles can safely navigate busy streets by annotating items such as vehicles, traffic lights, and pedestrians. To make this possible, self-driving automobile perception algorithm rely heavily on bounding boxes.

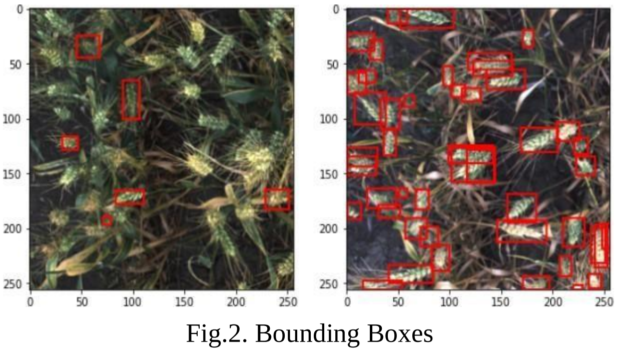

Fig.2. Bounding Boxes

4. DATA PIPELINE

In order to boost the confidence of postulated objects in still photos, the data pipeline is widely utilised. The first two modules illustrated in the Fig.2 are often constructed using time-consuming techniques. In most cases, the tracking module is the least computationally intensive. The two most computationally demanding modules and their properties, which can be parallelized to some extent, are discussed in the following sections.

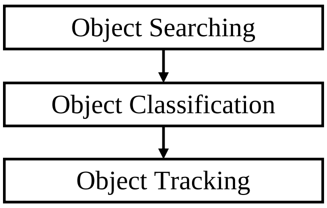

Fig.3. Data Pipeline

4.1 OBJECT SEARCHING

Finding objects in still photos is a difficult operation that is usually achieved using one of two methods: brute force or moving object segmentation. A normalised image window traverses the input frame at various scales and positions in the first category. The goal of this job is to extract features from each position of the normalised window and feed them into a trainable classifier that will determine whether or not the position includes an object. Concerning the effectiveness of finding an object, these techniques depend only on the performance of the classifier, which will be usually higher as more complex algorithms are used.

4.2 OBJECT CLASSIFICATION AND OBJECT TRACKING

Object classification can be accomplished by supervised or unsupervised approaches. Item tracking is a deep learning application in which the programs take a series of initial object detections and creates a unique identifier for each of them, then tracks the detected objects as they move around frames in a movie. Object tracking, in other terms, is the task of accurately recognising objects in a video and understanding them as a series of trajectories. Often, an indication surrounds the tracked object, such as a surrounding square that follows the object and shows the user where the object is on the screen.

4.3 FULLY CONNECTED NEURAL NETWORK LAYER

Fully Connected layers in neural organizations are those layers where every one of the contributions from one layer is associated with each enactment unit of the following layer. In most famous AI models, the last couple of layers are full associated layers that order the information extricated by past layers to shape the last yield. The fully connected layer consists of two to three layers of multilayer perceptron (MLPs). The multilayer perceptron map the activation volume from the different previous levels into a class probability distribution. In a standard MLP, the input layer is a vector. On the other hand, fully connected layers take an activation volume as the input. The fully connected layer for a layer l-1 is defined as:

4.4 BATCH NORMALIZATION

Normalization is a data pre-processing technique for converting numerical data to a common scale without changing the shape of the data. When we feed data into a machine learning or deep learning system, we usually modify the numbers to a balanced scale. Normalization is done in part to ensure that our model can generalise correctly.

Batch normalization is the process of adding more layers to a deep neural network to make it faster and more reliable. The standardising and normalising procedures are performed by the new layer on the input of a previous layer. A typical neural network is trained using batch data, which is a collection of input data. Similarly, with batch normalisation, the normalising procedure is done in batches rather than as a single input.

4.5 ACTIVATION FUNCTIONS

Each neuron in a neural network is connected to a large number of other neurons. This permits signals to move from the input layer to the output layers via the network. This contains a plethora of hidden layers in the space between the two. The signal onward propagation through this network is aided by the activation function. The input received by activation functions is transformed and the values are kept within a reasonable range. Initially, an activation function is assigned to a neuron or an entire layer of neurons. The weighted sum of input values is added up and the activation function is applied to the weighted sum of input values and transformation takes place.

These transformed values are the output to the next layer. Activation functions are of various types. The most common activation functions are the Sigmoid function, the TanH function, and the ReLU function. The sigmoid function transforms the values to the range between 0 and 1. The tanh function transforms the values between the range -1 and 1. It can be thought of as a scaled sigmoid function. The output values are centered around zero. The ReLU function takes the form of $f(x) = \max(0,x)$. Here, the transformation leads positive values to be 1, and negative values to be zero. It is shown to accelerate the convergence of gradient descent. ReLU has become the default activation function for hidden layers.

4.6 LEAKY RELU

The ReLU activation function has been improved with the Leaky ReLU function. In the case of the ReLU activation function, the gradient is 0 for all input values less than zero, deactivating the neurons in that region and perhaps causing the dying ReLUproblem.The term leaky ReLU was coined to describe a solution to this issue. We specify the ReLU activation function as an extremely small linear component of x instead of declaring it as 0 for negative values of inputs(x).

$$f(x)=\max(0.01*x,x)$$

If the input is positive, this method returns x, but if the input is negative, it returns a very little number, 0.01 times x. As a result, it also outputs negative values. The gradient of the left side of the graph now has a non-zero value as a result of this tiny change. As a result, there would be no more dead neurons in that area.

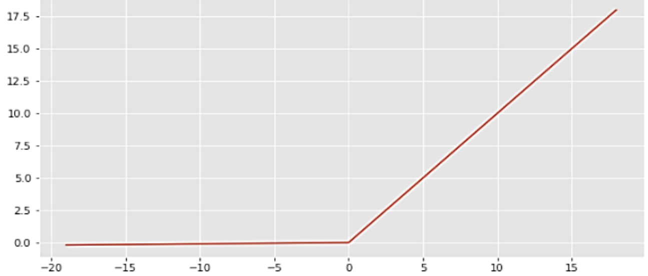

Fig.4. Leaky ReLu

5. MODELS

5.1 MODEL 1: YOLO

Multiple bounding boxes and class probabilities for those boxes are projected simultaneously by a single neural network. YOLO improves detection performance by training on entire photos. This unified model has various advantages over standard object detection approaches. YOLO is an acronym for you only live once. We don't require a complicated process because frame detection is a regression problem. To forecast detections, we simply execute our neural network on a new image at test time. On a Titan X GPU, our base network performs at 45 frames per second with no batch processing, and a faster version runs at more than 150 frames per second. This means we can process real-time streaming video with a latency of less than 25ms.

When making predictions, YOLO considers the image as a whole. Unlike sliding window and region proposal-based approaches, YOLO views the full image during training and testing, therefore it encodes contextual information about classes as well as their appearance implicitly. Because it cannot see the greater context, Fast R-CNN, a top detection method, misidentifies background patches in an image as objects. When compared to Fast R-CNN, YOLO makes less than half the number of background errors.

YOLO learns generalizable object representations. YOLO outperforms leading detection algorithms like DPM and R-CNN by a considerable margin when trained on natural photos and tested on artwork. YOLO is less likely to break down when applied to new domains or unexpected inputs because it is extremely generalizable.

The above table indicates the hyper parameters used for the model. Per grid cell, YOLO predicts numerous bounding boxes. We only want one bounding box predictor to be accountable for each object during training. Based on whose prediction has the highest current IOU with the ground truth, we assign one predictor to be responsible for predicting an object. As a result, the bounding box predictors become more specialised. Each predictor improves its ability to anticipate specific sizes, aspect ratios, or item classifications, increasing overall recall.

5.2 FASTER RCNN

The Fast R-CNN detector was published as an evolution of R-CNN. Fast R-CNN brings a design that simultaneously trains a classifier and regressor under the same network configurations. This achieved a speed over 200 times faster than R-CNN. It is one of the most popular object detection models and used in Madec et al. ResNet34 is used along with ResNet50 because it is less prone to overfitting and faster to train. We randomly sampled ten patches of size 1024×1024 pixels for each image in the training datasets and validation dataset. The test images were also predicted to be of size 1024×1024 pixels.

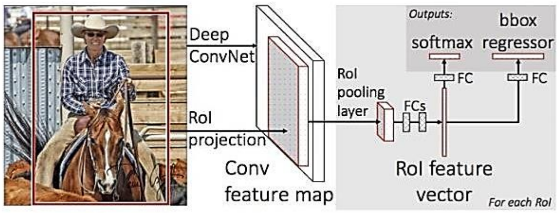

Fig.5. Architecture of fast RCNN

Table.1. Hyper parameter tuning

Block	Range	Input Layer	Kernel Size	Strides	Padding	Leaky RELU (alpha)	Batch Normalization
1	2	64	3	1	Same	0.1	Yes
		32	3	1	Same	0.1	Yes
		64	3	1	Same	0.1	Yes
2	2	128	3	1	Same	0.1	Yes
		64	3	1	Same	0.1	Yes
		128	3	1	Same	0.1	Yes
3	8	256	3	2	Same	0.1	Yes
		128	3	1	Same	0.1	Yes
		256	3	1	Same	0.1	Yes
4	8	512	3	2	Same	0.1	Yes
		256	3	1	Same	0.1	Yes
		512	3	1	Same	0.1	Yes
5	4	1024	3	2	Same	0.1	Yes
		512	3	1	Same	0.1	Yes
		1024	3	1	Same	0.1	Yes
Output Layer 1	-	512	3	1	Same	0.1	Yes
Output Layer 2	-	256	3	1	Same	0.1	Yes
Output Layer 3	-	128	3	1	Same	0.1	Yes
Prediction Layer	-	10	1	1	-	-	-

6. RESULTS

Even though the number of epochs is low, the model has predicted the test images well, missing some of the wheat objects only so if the number of epochs is set to 40 or 50 the model may predict all the wheat objects. When comparing the YOLO3 Model and Faster RCNN Model based on the training and validation loss, YOLO3 Model has lower loss. The model gives better training and validation loss when the epochs increase from 1 to 2.

Table.2. Results from YOLO3 and FasterRCNN

Model	Epochs	Final Loss	Validation Loss	Training Time
YOLO3	1	0.1342	0.20340	9 Hrs
YOLO3	2	0.08703	0.08599	18 Hrs
YOLO3	5	0.07660	-	45 Hrs
FasterRCNN	2	1.2280	1.15090	10 Hrs

Table.3. Comparison between YOLO3 AND Faster RCNN

Model	Epochs	Final Loss	Validation Loss
YOLO3	2	0.08703	0.08599
FasterRCNN	2	1.22800	1.15090

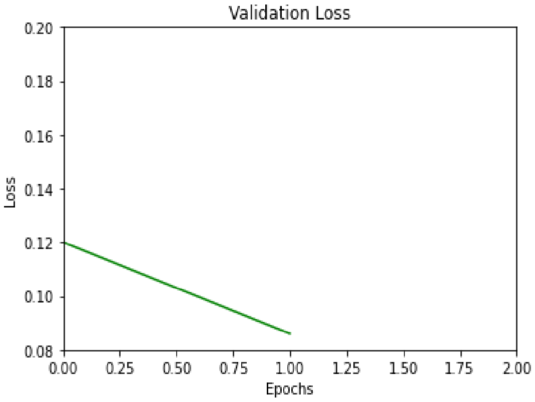

Fig.6. Validation Loss

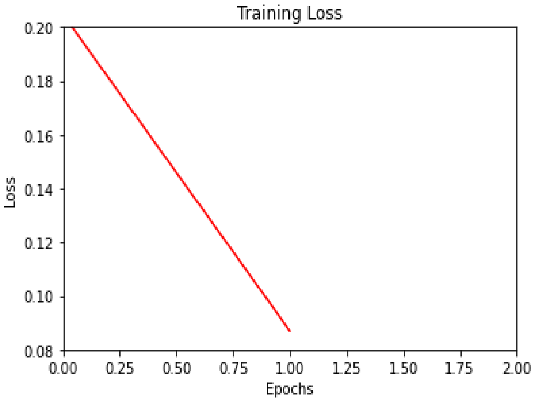

Fig.7. Training Loss

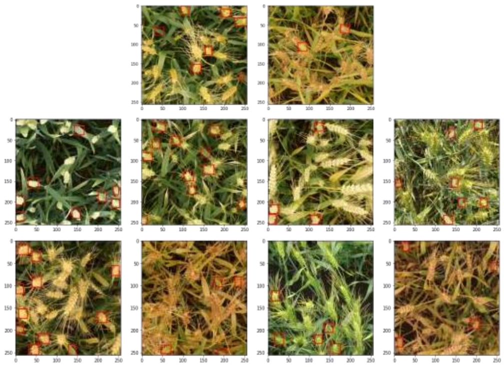

Fig.8. Prediction images for YOLO

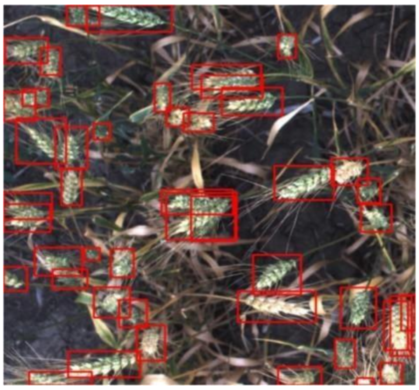

Fig.9. Predicted image for faster RCNN

7. CONCLUSION

For object detection yolo is one of best algorithms. To get large and accurate data about wheat fields wheat head detection is used. Wheat crop images are used to estimate the density and size of wheat heads in different varieties. Because of outdoor images, accurate wheat head detection is challenging. There is often overlap of dense wheat plants, and the wind can blur the photographs. Both make it difficult to identify single heads. Additionally, appearances vary due to maturity, color, genotype, and head orientation. So, the Yolo3 model is trained end to end for better results.

REFERENCES

[1] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei Xiang Bai and Zicheng Liu, "End-to-End Semi-Supervised Object Detection with Soft Teacher", Available at https://github.com/microsoft/SoftTeacher, Accessed at 2021.

[2] Yihe Tang, Weifeng Chen, Yijun Luo and Yuting Zhang, "Humble Teachers Teach Better Students for Semi-Supervised Object Detection", *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pp. 1-15, 2021.

[3] Na Zhao, Tat-Seng Chua and Gim Hee Lee, "Self-Ensembling Semi-Supervised 3D Object Detection", Master Thesis, Department of Computer Science, National University of Singapore, pp. 1-80, 2021.

[4] Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee and Tomas Pfister, "A Simple Semi-Supervised Learning Framework for Object Detection", *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pp. 1-12, 2020.

[5] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai and Zicheng Liu, "End-to-End Semi-Supervised Object Detection with Soft Teacher", Available at https://github.com/microsoft/SoftTeacher, Accessed at 2021.

[6] P.F. Felzenszwalb, R.B. Girshick, D. McAllester and D. Ramanan, "Object Detection with Discriminatively Trained Part based Models", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 9, pp. 1627-1645, 2010.

[7] S. Ren, K. He, R. Girshick and J. Sun, "Faster R- CNN: Towards 14 Real-Time Object Detection with Region Proposal Networks", *Proceedings of International Conference on Neural Information Processing Systems*, pp. 1-12, 2015.

[8] Nhu-Van Nguyen, Christophe Rigaud and Jean-Christophe Burie. "Semi-Supervised Object Detection with Unlabeled Data", *Proceedings of International Conference on Computer Vision Theory and Applications*, pp. 1-12, 2019.

[9] Chuck Rosenberg, Martial Hebert and Henry Schneiderman, "Semi-Supervised Self-Training of Object Detection Models", *Proceedings of International Conference on Computer Vision*, pp. 1-13, 2005.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", *Proceedings of International Conference on Computer Vision Theory and Pattern Recognition*, pp. 779-788, 2016.

[11] Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", *Proceedings of International Conference on Advances in Neural Information Processing Systems*, pp. 91-99, 2015.