A Study on the Effect of Target Object Size in Object Detection

Jacob Cassar Ellis
Author

Matthew Montebello
Supervisor

Dylan Seychell
Co-Supervisor

ABSTRACT

Recent years have seen an impressive increase in the efficiency and accuracy of object detection models through the use of Region-based Convolutional Neural Networks (R-CNN). Studies have been carried out to improve object detection models. However, the detection of small objects still poses numerous challenges for the said models. Small object detection is considered as one of the biggest challenges in object detection for several reasons. One in particular is due to the resizing of the feature maps within the pooling stage resulting in the loss of the small target object's features. Moreover, accuracy diminishes as these networks struggle to distinguish between foreground and complex backgrounds such as rough terrain.

An experiment was designed to investigate a selection of deep neural networks and analyse the effects that the distance between the capturing device and the target object. The set up requires a custom dataset, containing object instances that are identifiable by the selected models, each with an object instance at a varying distance. The COCO dataset was selected as the training and benchmarking dataset in this experiment. Following the selection of a benchmarking dataset, the custom dataset used to test for the effect of target object size was obtained. The custom dataset was prepared in a controlled environment set up with pre-established distance measurements and multiple small objects placed in a non-complex background for detection. By varying the range of distances, size of the object instance decreases, producing a lower amount of features.

1. INTRODUCTION

Object detection is a task that deals with detecting instances of objects of a specific class. Over the recent years, many researchers have invested their time into developing solutions and improving upon them continuously to provide the most accurate and efficient model [6, 5, 10, 7]. Recently, through the rapid development in deep learning, there have been numerous breakthroughs in large scale object detection and image classification. Deep Learning has brought about models such as region-based Convolutional Neural Networks that can perform object detection in fast and efficient manner [3].

These models have proven to work extremely well for detecting large objects like planes or cars but suffer when attempting to handle small objects [14]. There are several reasons for this, which include occlusion, lack of light, and the information present in the image itself. As the size of the object decreases, the spatial resolution between the pixels is reduced, thus resulting in a lack of information to produce a rich representation of the target object [2]. Without an efficient representation, the model is unable to detect the object, or could produce false positives by suggesting the target object is elsewhere. The limits of each model can be tested by varying the distance between the capture device and the target object as it mimics the object becoming smaller. Through this experiment, we will demonstrate how it is possible to determine the maximum distance limit for each model to correctly detect an object and improve upon it.

All modern solutions for object detection are heavily based on Convolutional Neural Networks. These powerful networks operate by extracting low level abstractions from images to extract small features like edges. These lower level features are combined to extract more complex and abstract features. This occurs between the hidden layers that are present within the CNN, through combinations of convolutions and pooling. Moreover, the recent developments of YOLO[12, 11], R-CNN [2] and Faster R-CNN's [13] have proven to provide very accurate results in a short amount of time for detecting large objects.

However, the low-level abstractions are the reason that detecting small objects poses such an issue. Current models reduce image resolution between the convolutional layers in order to extract the features of an object. These features are passed into the following layer and eventually combined to create a rich representation of the object. Small objects, however, only take up a small percentage of the image's spatial resolution. Thus, when the resolution is decreased, the small objects located in the image have too little information for the CNN to provide any form of features. This results in the loss of the small object completely.

The information provided through this experimental could provide new insights to help develop more accurate systems. Moreover, the results could also be used to pinpoint the limitations of current state-of-the-art models.

2. AIMS AND OBJECTIVES

The aim of this project is to investigate the impact of spatial resolution in small object detection by varying the distance between target object and capture device. In order to achieve this goal, the following objectives have been identified:

- Research and evaluate current state-of-the-art object detection models.
- Design and build a dataset for small object detection at varying distances based on the COCO dataset.
- Investigate the accuracy of the chosen models over varying distances from the object instance.

3. LITERATURE REVIEW

The following section focuses on the first objective, by delving deeper into the history of CNNs, whilst also exploring the branches that have been developed since AlexNet [5]. The additional branches discussed heavily focuses on object detection and will be split into two main sub-sections. The first will focus on Region-Based Frameworks such as R-CNN [2] , while the second will focus on Regression and Classification Based Frameworks like YOLO [12] that eventually resulted into YOLOv3 [11]. This will be followed by a brief description on the problem of small object detection. Furthermore, will highlight how certain models have been specialized for small object detection.

A Region-based Convolutional Neural Network (R-CNN) [2] which makes use of a selective search (SS) algorithm to generate around 2000 region proposals for the inputted image was proposed by Gershick in 2014. The feature extraction methodology applied in R-CNN makes use of a warping function cropping each region of interest to a fixed resolution. The AlexNet [5] architecture was modified for the CNN module of the network to provide 4096 dimensional features. Once the features are extracted, class-specified linear Support Vector Machines (SVM) are used to classify and locate the objects within the image. Gershick et al modified R-CNN [2] by applying a simple multi-task loss function to both classification and bounding-box regression in [1] called *Fast R-CNN*. Moreover, applied a spatial pyramid pooling (SPP) [4] layer to process the whole image using a CNN to produce a convolutional feature map. Using this feature map, region proposals are extracted and applied to the RoI pooling layer. This layer is a special case of SPP as it is a one level pyramid [17] which is used to transform the feature vectors into a fixed size to be fed to the fully connected layer. Once the feature vectors have passed through the fully connected layers, they are branched into two sibling output layers, the softmax and bounding box regressor branches. The softmax branch is used to generate a probability distribution over $k+1$ classes, where k refers to object classes with the addition of a background class. The bounding box regressor branch refines the position of each box using a set of four real-valued numbers for each of the K object classes [1].

Ren et al proposed a Region Proposal Network (RPN) which allows for cost-free region proposals creating Faster R-CNN [13]. The first stage is the RPN, or deep FCN that is used to propose regions. The second stage makes use of the previously explained *Fast R-CNN* design. RPN allows for an image input of varying size to produce a set of object proposals each with their respective score. The set of object proposals is created by sliding an $n \times n$ window over the convolutional feature map returned by the final convolutional layer. By applying this sliding window, a low dimensional feature vector is obtained for each sliding window which is then passed through ReLu to introduce non-linearity. Here, Ren et al. introduces translation invariant anchors which allows the RPN to handle multiple scales and ratios through a pyramid of anchors as it is more cost efficient [13].

Redmond et al. sought to produce an object detection model that is crafted on the foundation of human perception whilst also improving on models such as RCNN. The first YOLO framework [12] was developed in 2015, enabling end-to-end training whilst maintaining high performance in real-time by simultaneously predicting multiple bounding boxes and their respective class probabilities. The input image is taken as a whole and then split into a $S \times S$ grid. A grid cell contains the center of an object then said grid cell is responsible for detecting that object [12]. The grid cells produce a set of B bounding box predictions with confidence scores which provide an insight as to how certain the model is that there exists an object in that cell and how accurate the predicted bounding box is around said object.

CNNs are able to create low-level abstractions such as lines or edges and combine them to produce a classification. However, this is why they struggle when attempting to detect a target object with a relatively small size. This is due to the pooling layers found within the hidden layers of a CNN. These pooling layers are also used in the network architectures of Faster R-CNN and YOLO which reduce size of the images drastically. By doing this, the features of the small target object that are extracted from the first convolutional layers 'disappear' at a point deeper into the network. Thus, the features do not reach the detection and classification stages of the model.

A recent study on litter detection was carried out by Schembri et al [14] which made use of high variable backgrounds. Schembri states that tuning the sensitivity between foreground and background increases performances which can be done using dataset engineering and background set sampling. In addition to this, certain models such as [15] are specialized for small objection as they use Feature Pyramid Networks [8] (FPNs). FPNs merge two existing ideologies, single feature maps and pyramidal feature hierarchy to handle scale invariance to enrich lower lever layers with semantically stronger features obtained from the later convolutional layers [8].

4. METHODOLOGY

In this section, the implementation of the necessary steps taken to test for the impact of target object size in object detection will be explained in further detail. Specifically, how each model was implemented, trained on the COCO dataset and evaluated using a custom test dataset.

The first step was to select a dataset containing a large selection of objects that could be used to test on. It is crucial that one selects the correct dataset for the task at hand as it provides the ability to measure the performance of the object detector being tested. For this experiment, the COCO dataset [9] was used as it provides the highest categories and instances per image. The COCO dataset contains over 80 object instances of common objects which allows us to select from a variety of different objects to test for the effect of target size in object detection. Furthermore, provides a set

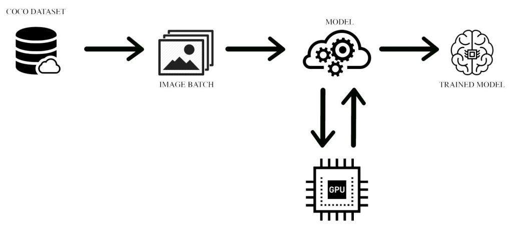

Figure 1: High-level architecture for Training a Model

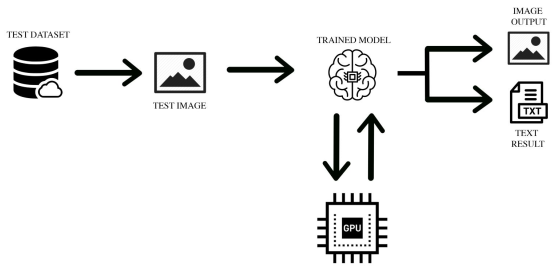

Figure 2: High-level architecture for Testing a Model

of annotated images with bounding boxes to test for object detection. In addition, each bounding box annotation also provides a respective label, used to confirm if the model being tested has correctly identified the target object instance. The COCO dataset is also used as a bench-marking tool to provide the Average Precision and Average Recall of a model across several scales. These scales, small, medium and large focus on the size of the object instance, allowing each model to be evaluated with respect to target object size [11]. A sample of the YOLOv3 Baseline results can be seen in Figure 3.

The next step was to obtain a selection of object detection models pre-trained on the COCO dataset. For this task, three different object detectors were selected. Firstly, YOLOv3 [11] is the third model developed by Redmond et al. This version builds upon the existing work of presented in section 3. YOLOv3 predicts bounding boxes by making four predictions relating to co-ordinates of the box (t_x, t_y, t_w, t_h). Moreover, an objectness score is introduced through the use of logistic regression for each bounding box dependent on the amount of overlap between the predicted box and the ground truth. This allows the model to achieve a *Average Precision* of 57.9. Secondly, Detectron2 [16] is a system that implements state-of-the-art object detection algorithms and was selected as provides the ability to train a model using different backbones. The model chosen is a Faster R-CNN using a ResNeXt-101-FPN backbone. The FPN provides several feature maps at different scales while still maintaining the semantic information of the features themselves in a fully convolutional manner. The Detectron2 implementation can achieve a *Average Precision* of 43. Finally, EfficientDet was selected as it has recently been released making use of a new EfficientNet backbone, which was inspired from one stage detectors. Moreover, uses a BiFPN for feature extraction and a shared class prediction network for classification. The model selected, EfficientDet-d7 can achieve an *Average Precision* of 52.2 [15]. A high level architecture detailing the process of training a selected model from scratch can be seen in Figure 1.

The final step to evaluate the effect of target object size in object detection was to obtain a custom dataset containing images of identifiable objects at varying distances. To do this, a high level architecture describing the processing of testing a model using a custom dataset can be seen in Figure 2. The test set was created by measuring out a maximum distance of twelve meters, marked at one meter intervals. The camera is then placed at these one meter intervals , capturing at image at each interval. The process is then repeated for the remaining object classes selected. By increasing the distance from the camera to the object, the resultant image reduces the target object size thus allowing us to evaluate each model for each instance. For this experiment, four common objects were selected to be used to compile the test set from the 80 different object classes presented in the COCO dataset. The objects that were selected are as follows:

- chair
- sports ball
- clock
- bottle

For each object selected, a set of baseline images were obtained to provide a ground truth to evaluate from. These baselines are to ensure that the objects presented are identifiable by the object detection model being tested at the time. Once the baselines were obtained, the next step was to obtain the set of images containing the object instances at varying distances for each of the mentioned object instances.

5. EVALUATION AND RESULTS

To evaluate the model, the baseline results will be converted into a confusion matrix following the template presented in Table 1. Each object instance will contain exactly one *True Positive* (T_p), that is the object expected to be classified. Any additional classifications that are not based on the object instance is measured as a *False Positive* (F_p). The *False Negative* (F_n) refers to the instance where the model incorrectly classifies the object instance.

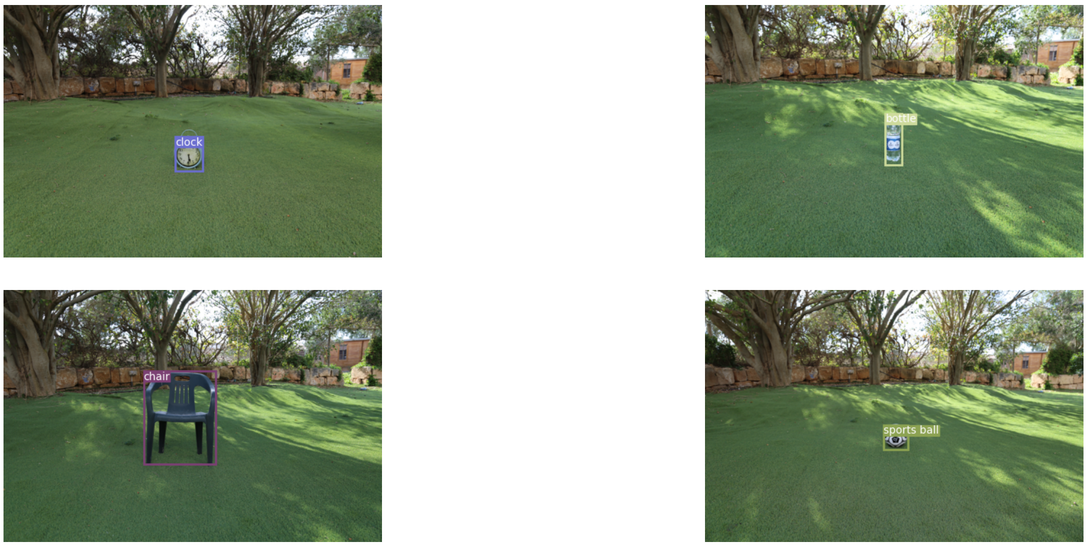

Figure 3: YOLOv3 Baseline Results produced during experiment

		Test Result	
		Positive	Negative
Classified	Positive	T_p	F_p
	Negative	F_n	T_n

Table 1: Confusion matrix

Model	Clock	Sports Ball	Chair	Bottle
YOLOv3	1	0.99	0.97	1
Detectron2	1	0.99	0.98	1
Efficientdet	0.76	0.95	0.78	0.83

Table 2: Confidence Score for Baselines

Once the confusion matrix has been obtained, the recall can be obtained using *Equation 1.* By obtaining the recall for each object instance, each model can be tested for its sensitivity to target object size by alternating the distance. Additionally, we can calculate the precision of these models using *Equation 2.*

$$precision = \frac{T_p}{T_p + F_p} \tag{1}$$

$$recall = \frac{T_p}{T_p + F_n} \tag{2}$$

By combining these two results using *Equation 3* we obtain the F-Measure. However, for this experiment a bias β will be introduced as seen in *Equation 4.* By implementing this bias we can give more importance to the recall of each instance.

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3}$$

$$F_\beta = (1+\beta)^2 \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{4}$$

The first step was to evaluate the baselines in order to obtain a ground truth for each object instance. To do this, each of the implemented models had their threshold limits for confidence score to 0.1, the remaining setting were left unaltered. By using a low threshold limit, we can ensure that the model is not producing a significant amount false positives. Moreover, ensures that the object instance being tested is not classified more than once using a different label. Each model successfully managed to identify and classify each selected object correctly. Table 2 shows the confidence score each model obtained for the respective object baseline. This gives a general insight to how the object detectors are expected to perform. It is interesting to note that EfficientDet-d7 produced the lowest results across all baselines, despite it being one of the current best performing models.

Once the ground truths were obtained, the custom dataset was applied to test for the effect of target object size. To maintain consistency, each model was implemented using a threshold limit of 0.1. The remaining settings were unaltered to ensure the best performance in terms of accuracy. It is important to note that YOLO compresses the image size drastically for classification which could affect the overall accuracy of the model. To test using this custom dataset, each object class was tested separately. The twelve images provided for each object were classified and the results were stored in the same directory. Each result consists of the classified image and the respective bounding box, label and confidence score. These results were then used to compile a confusion matrix for each object instance.

Each of the selected object detection models were successfully tested on the custom dataset. Although it was not the main objective, it was noted that YOLO and Efficientdet-d7 took less time than the Detectron2 model to process and classify each set of object images. This was expected as both YOLO and Efficientdet-d7 are a form of one staged detector. Upon first glance, it was found that the EfficientDet-

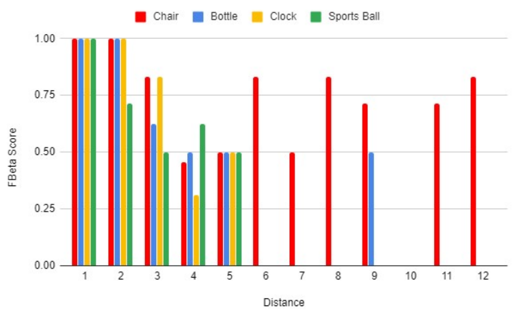

Figure 4: Detectron2 FBeta Scores against Distance

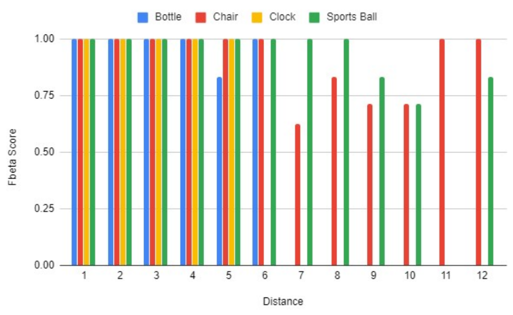

Figure 5: EfficientDet FBeta Scores against Distance

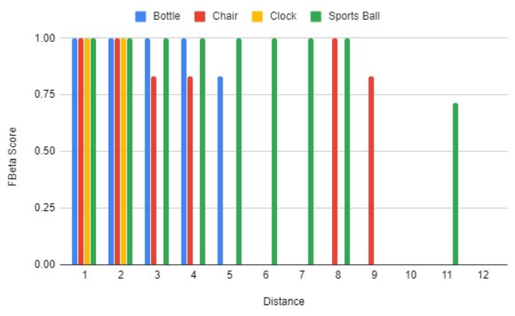

Figure 6: YOLOv3 FBeta Scores against Distance

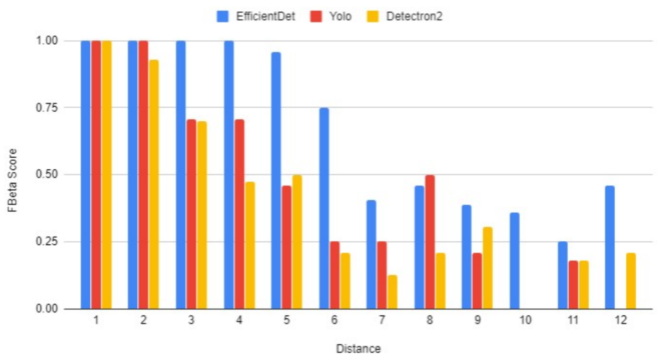

Figure 7: Average FBeta Scores against Distance

D7 model was the most successful at identifying the selected objects over distances. With a total true positive count of 34 from a maximum of 48. However, one can note that the Faster R-CNN Detectron2 implementation was also relatively successful, achieving a true positive count of 31 from 48. YOLO was the least successful with a total true positive count of 22. Although, as previously explained, could be due to the resizing of each image.

However, by using the results obtained we are able to obtain the Precision and Recall values for each of the models using Equation 1 and Equation 2. The results can be seen in Table 3. Although YOLO provided the lowest total positive count we can see that the model is still very precise in a score of 56%. Efficientdet-d7 provided the highest precision overall with a score of 61% whilst the Faster R-CNN achieved a score of 20%. However, these results are biased due to the implementation of a lower threshold as more of an emphasis was placed on Recall. As mentioned previously, YOLO was the least successful at classifying and correctly labelling each object instance with an overall recall score of 46%. The second best performing model is the Faster R-CNN model which although had a very low precision, produced a recall score of 65%. Finally, Efficientdet-d7 was the best performing, achieving a recall score of 71%.

A reflection of the results presented in Table 3 can be seen in Figure 4, Figure 5 and Figure 6 for each of the selected object instances. In these figures, the average FBeta score for each object instance against distance can be seen. It was interesting to note how the model begin to decrease in accuracy once the object exceed the five meter mark. The better visualization of this can be seen in *Figure 7* which provides a comparison of the Average FBeta scores obtained by each model at the distanced labelled on the x-axis. It is clear to see how the introduction of FPNs increased the accuracy for small object detection. The YOLOv3 and Detectron2 exhibited similar behaviour, reaching scores of under 50% once the target object is at five meters or more. However, it was found that the EfficientDet model provided a minimal decrease to its previously perfect results upon reaching five meters. At six meters, the model struggled to detect the bottle and clock instances once as seen in Figure 6. This was interesting to find as object sizes for these two instances are similar to that of the sports ball which was identified in all but one instance This could possibly show that small object detection requires distinctive features, object with similar features could potentially reduce an object detectors performance.

Table 3: Precision, Recall and F_β Results

Model	**Precision**	**Recall**	F_β
YOLO	0.564	0.458	0.465
Efficientdet	0.607	0.708	0.672
Detectron2	0.199	0.646	0.448

6. CONCLUSION

6.1 Future Work

The results seen in *Figure 7* could be used to provide a different point of view to tackle the issue of small object detection. A possible improvement to the current experiment would be to produce a similar dataset to that presented in this dissertation to train each model. By training specifically for objects over varying distances, each model could possibly be better equipped at carrying out the two experiments presented in the previous section. Additionally, the experiment itself could be improved by creating the test dataset using a plain background reducing the overall amount of false positives in each image as this affects accuracy. Moreover, one could test additional object detection models such as SSD, CBNET or SNIPER which could provide further insight to the effect of target size in object detection. Finally, the results obtained can be used as a basis for applying the selected models for applications that are better suited for that particular model. A typical application, as presented in [14] would be to apply the EfficientDet model to a drone using aerial object detection for litter detection.

6.2 Conclusion

In conclusion, this paper provided an analysis on the effect of target object size in object detection. To do this, three objectives were presented in Section 2 which provided a firm plan for this project. The results shown in *Figure 7* provides a overview of the results obtained from the experiments in a single chart. Each model can be compared to each other directly at the distance labelled on the x-axis.

The first objective sought to provide an in depth analysis on how current state of the art models tackle the process of object detection. This objective was completed through Section 3 using current state-of-the-art models that were identified using their submission to the object detection task. Once these models were identified, an extensive insight to how these object detect models were inspired and how each model classifies an image differently using several different architectures is provided. Once this objective was complete, the remaining objectives could be achieved.

To complete the second objective , a custom dataset was obtained that could be used to test the effect of target object size. To do this, the design of the set-up explained in Section 4 introduced the idea of applying one meter intervals between the object and the capturing device , thus reducing the area of the object captured in the image as the distance increases. The procedures described in the Methodology section provide a guide on how to set up the environment to obtain the custom dataset. Moreover, provides additional details regarding the selection of the object instances and how the design of the custom dataset allows for the study of the effect of target object size in object detection.

Once all the previous objectives were completed, each model was tested using the custom dataset explained in Evaluation and Results section. In order to test for overall accuracy, several evaluation metrics were presented in Section 5 that are commonly used for object detection. Futhermore, the results obtained in the first step successfully produces a set of baselines that were used as a ground truth for the object instance. The second step tested for the effect of target object size through the variation of distance. By successfully evaluating each of the selected models in terms of accuracy, the final objective was successfully completed. The results obtained found that the EfficientDet-d7 model was the most accurate reaching an average FBeta measure of 60.7%, with the remaining two models being fairly close in terms of accuracy at 46.5% for YOLOv3 and 44.8% for Detectron2.

7. REFERENCES

[1] R. Girshick. Fast r-cnn. *ICCV*, 2015.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurace object detection and semantic segmentation. *CVPR*, 2014.

[3] S. Haykin. *Neural Networks and Learning Machines.* Pearson, Hamilton, Ontario, Canada, 3 edition, 2010.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE TPAMI*, 2014.

[5] A. Krizhevky, I. Sutskever, and G. E.Hinton. Imagenet classification with deep convolutional neural networks. *Proc. 27th International Conference on Machine Learning*, pages 1097–1105, 2012.

[6] Y. LeCun, P. Haffner, L. Bottou, , and Y. Bengio. Object recognition with gradient-based learning. *Shape, contour and grouping in computer vision*, 38:319–345, 1999.

[7] M. Lin, Q. Chen, and S. Yan. Network in network. *International Conference on Learning Representations*, 2013.

[8] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. *CVPR*, 2016.

[9] T.-Y. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. *CoRR*, abs/1405.0312, 2014.

[10] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. *Proceedings of the 25th International Conference on Neural Information Processing Systems*, 2010.

[11] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. 2018.

[12] J. Redmond, S. Divvila, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *IEEE conference on CVPR*, pages 779–788, 2015.

[13] S. Ren, K. He, R. Girshick, and J. Su. Faster r-cnn: Towards real-time object detection with region proposal networks. *CVPR*, 2016.

[14] M. Schembri and D. Seychell. Identification of small objects within high variability backgrounds. *ISPA*, 2019.

[15] M. Tan, R. Pang, and Q. Le. Efficientdet: Scalable and efficient object detection. 11 2019.

[16] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[17] Z.-Q. Zhao, P. Zheng, S. tao Xu, and X. Wu. Object detection with deep learning: A review. *TNNLS*, 2019.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Declaration

Plagiarism is defined as “the unacknowledged use, as one’s own work, of work of another person, whether or not such work has been published” (Regulations Governing Conduct at Examinations, 1997, Regulation 1 (viii), University of Malta).

I / We*, the undersigned, declare that the [assignment / Assigned Practical Task report / Final Year Project report] submitted is my / our* work, except where acknowledged and referenced.

I / We* understand that the penalties for making a false declaration may include, but are not limited to, loss of marks; cancellation of examination results; enforced suspension of studies; or expulsion from the degree programme.

Work submitted without this signed declaration will not be corrected, and will be given zero marks.

* Delete as appropriate.

(N.B. If the assignment is meant to be submitted anonymously, please sign this form and submit it to the Departmental Officer separately from the assignment).

Jacob Cassar Ellis	[illegible]
Student Name	Signature
Student Name	Signature
Student Name	Signature
Student Name	Signature

ICT3909-YR-A-1920	A Study on the Effect of Target Object Size in Object Detection
Course Code	Title of work submitted

04/06/2020

Date