Small-Object Detection

CSE523: Machine Learning & CSE541: Computer Vision

Authors: Jay Patel AU2040014 -- Dhanya Mehta AU2040021 -- Spandan Shah AU2040265

Abstract

Problem Definition: Evaluate performance of various object detection techniques (in case of small objects) on AU Drone dataset.

Object detection is a fundamental task in computer vision, but detecting small objects in high-resolution images is challenging due to the limited spatial resolution of convolutional neural networks (CNNs). EfficientDet, that achieve state-of-the-art performance with significantly fewer parameters and FLOPs than previous methods. It uses a compound scaling method that scales the depth, width, and resolution of the model to achieve better accuracy and efficiency. A new focal loss that is optimized for object detection tasks is also introduced. The models are evaluated on popular object detection benchmark, COCO and VisDrone, and it shows that they achieve state-of-the-art performance with much smaller models than previous methods.

Keywords—EfficientDet, QueryDet, MetrixNet, AP, MPA, IOU

I. Introduction

Small object detection is basically a computer vision problem where you aim to accurately identify objects that are small in a video feed or image. Major problem is to detect instances of visual objects of certain classes (for example, humans, animals, cars, or buildings) in digital images. Through the project, we aim to achieve a concise understanding of the algorithms available to solve the problem mentioned. The object itself does not necessarily need to be small. For instance, small object detection is crucial in aerial computer vision, where you need to be able to accurately identify objects even though each individual object will be small relative to the photo size. To train the model, the VisDrone dataset has been used. The EfficientDet and QueryDet are the main algorithms to solve the problem.

II. Literature Survey

A. EfficientDet

EfficientDET is a new model that combines the strengths of one-stage and two-stage object detection models, so we can consider it as light weight and low computational model. It uses a single convolutional neural network (CNN) to produce both object class and bounding box predictions. This model introduce a new method for scaling the network where the resolution depth and width can be scaled up to our requirements , which enhances the depth, width, and resolution of the neural network. This scaling method leads to significant improvement in rescaling the image due to which small object of low resolution which are converted to higher resolution in D7 model architecture which helps in identifying small object significantly. The best part for this model is it uses BiFPN Network which extract features in neural network by the Top-down and Bottom-up approach by which every small information can be extracted to the model by repeated blocks of iteration of layers. Other than this there are seven layers of feature extraction from D0 to D7. Baseline model is D0 where backbone used is ReteinaNet-50 and YOLO-V3 and can be experimented with other backbone for different results

The authors test the performance of their model on several commonly used object detection datasets such as COCO, Pascal VOC, and Open Images. They demonstrate that EfficientDet outperforms previous methods and is more computationally efficient.

How features are getting extracted, and the model is scaled:

For the given list of multi-scale features $\vec{P}^{in} = (P^{in}_{l1}, P^{in}_{l1,} ...)$, where P^{in}_{li} represents the feature at level l_i, our goal is to find a transformation f that can effectively aggregate different features and output a list of new features: $\vec{P}^{out} = f(\vec{P}^{in})$.
The conventional top-down FPN takes level 3-7 input features $= \vec{P}^{in} = (P^{in}_3,\ldots, P^{in}_7)$
where P^{in}_3, represents a feature level with resolution of $1/2^i$ of the input images. For instance, if input resolution is 640x640, then P^{in}_3 represents feature level 3 $(640/2^3 = 80)$ with resolution 80x80, while P^{in}_7 represents feature level 7 with a resolution 5x5. The conventional FPN aggregates multi-scale features in a top-down manner:
$P^{out}_7 = Conv(P^{in}_7)$
$P^{out}_6 = Conv(P^{in}_6 + Resize(P^{in}_7))$
.
.
$P^{out}_3 = Conv(P^{in}_3 + Resize(P^{in}_4))$
Resize is usually an upsampling or downsampling operation for resolution matching, and Conv is a convolutional operation for feature processing.[1]

B. QueryDet

This model incorporates object detection and feature extraction, including YOLOv3, FPN, and Sparse R-CNN as backbone. YOLOv3 is a model that identifies objects using a single CNN for predicting both the object class and bounding box. FPN is a

feature pyramid network that enhances the multi-scale representation of features in an image. Sparse R-CNN is an object detection model that utilizes a sparse convolutional operator to decrease the computation and memory requirements of the CNN.

QueryDet is a new object detection model that combines the advantages of the previous methods to tackle the challenge of identifying small objects in high-resolution images. It utilizes a cascaded sparse query that capitalizes on the spatial sparsity of small objects to speed up the object detection process. The authors introduce a novel sparse sampling technique to decrease the memory requirements of the CNN.

The model is trained on well-known dataset including COCO and PASCAL VOC and show that QueryDet outperforms prior methods and achieves state-of-the-art results while also being much more computationally efficient.

C. Performance Metrices Formulations:

AP (Average Precision):

To calculate the AP score, a precision-recall curve using the model's predictions and ground-truth labels has to be generated. Assuming that the precision-recall curve is shown in the following table.

Recall	Precision
0.0	1.0
0.1	0.9
0.2	0.8
0.3	0.7
0.4	0.6
0.5	0.5
0.6	0.4
0.7	0.3
0.8	0.2
0.9	0.1
1.0	0

Then, the precision-recall curve has to be divided into several segments, and calculated the average precision for each segment. The AP for each segment and the weighted average of these AP values were calculated as follows:

Segment 1: Recall values between 0.0 and 0.3

- AP1 = (0.9 + 0.8 + 0.7) / 3 = 0.8

Segment 2: Recall values between 0.3 and 0.5

- AP2 = (0.6 + 0.5) / 2 = 0.55

Segment 3: Recall values between 0.5 and 1.0

- AP3 = (0.4 + 0.3 + 0.2 + 0.1) / 4 = 0.25

Weighted average of AP values:

- AP = (AP1 * (0.3 - 0.0)) + (AP2 * (0.5 - 0.3)) + (AP3 * (1.0 - 0.5)) = 0.46

Therefore, model achieved an AP score of 0.46 on this dataset.

AR (Average Recall):

To calculate the AR score, the recall values for different levels of confidence thresholds has to be computed. The recall values were computed as the number of true positive detections divided by the total number of ground-truth objects. The recall values for different levels of confidence thresholds are shown in the following table:

Confidence Threshold	Recall
0.5	0.80
0.6	0.75
0.7	0.70
0.8	0.65
0.9	0.60

Then the Average Recall (AR) score was computed as the mean of the recall values at different levels of confidence thresholds:

AR = (0.80 + 0.75 + 0.70 + 0.65 + 0.60) / 5 = 0.70

Therefore, the model achieved an Average Recall (AR) score of 0.70 on this dataset.

IoU (Intersection over union):

To calculate the IoU for each object in the test set, the predicted bounding boxes will be generated first using the model's outputs. The predicted bounding boxes are to be compared with the ground-truth bounding boxes then, and the IoU values are to be computed using the following formula:

IoU = (Intersection Area) / (Union Area)

where the intersection area is the area of the overlap between the predicted bounding box and the ground-truth bounding box, and the union area is the area of the union between the two bounding boxes.

Average IoU = (Sum of all IoU values) / (Number of objects) Which will give the final IoU score for the model.

III. Implementation

We transformed the dataset for EfficientDet to Tfrecord. From the COCO format, which is used to train the dataset in EfficientDet. Then we made Infer model to be used for EfficientDet. We tested that model on two images and understood the parameters for the same. Then, implementation of EfficientDet was done by us as per the GitHub repository provided on git. Then we tested dataset on that trained model. We have also train and test the QueryDet model on visDrone dataset. For this we first converted dataset to a coco formate.

IV. Results

We have used to infer model of EfficientDet D0 and EfficientDet D7 on the same image and we found the bounding box of D7 to be more accurate and it also detects more number of smaller objects. In D0 it is detecting a 5 object and D7 detecting a 19 object from that image.

We have trained and tested the model of the EfficientDet-d0 and QueryDet on the visdrone dataset. And found the AP, AP50, AP75, AR for both of it. The Ap of small object is very small in it add object like car and van has high AP.

	AP	AP .50	AP .75	AR
EfficientDet D0	38.26	54.64	37.58	43.62
QueryDet	27.33	49.56	26.55	35.67

V. Conclusion

For the EfficientDet model, the accuracy is higher than QueryDet. It gives lower accuracy then coco dataset due to small object that makes harder to Detect it. Most of detection happened near .50 present IoU that why there are big difference in AP.50 and AP .75. It happened due to small objects. EfficientD7 detect more object in same image then other. Scaling is an important part to detect the small object.

VI. References

[1] EfficientDet - Tan, M., Pang, R., & Le, Q. V. (2020, July 27). EfficientDet: Scalable and efficient object detection. arXiv.org. Retrieved February 20, 2023, from https://doi.org/10.48550/arXiv.1911.09070

[2] QueryDet - Yang, C., Huang, Z., & Wang, N. (2022, March 24). Querydet: Cascaded sparse query for accelerating high-resolution small object detection. arXiv.org. Retrieved February 22, 2023, from https://arxiv.org/abs/2103.09136

[3] MetrixNet - Rashwan, A., Agarwal, R., Kalra, A., & Poupart, P. (2020, January 9). Matrixnets: A new scale and aspect ratio aware architecture for object detection. arXiv.org. Retrieved March 1, 2023, from https://doi.org/10.48550/arXiv.2001.03194