Small object detection using various detection model

Jay Patel
AU2040014
Ahmedabad University
jay.p@ahduni.edu.in

Dhanya Mehta
AU2040021
Ahmedabad University
dhanya.m@ahduni.edu.in

Spandan Shah
AU2040265
Ahmedabad University
spandan.s@ahduni.edu.in

***Abstract*—Problem Definition: Evaluate performance of various object detection techniques (in case of small objects) on VisDrone dataset. The purpose of this report is to evaluate the performance of the various object detection using different models like EfficientDEt, QueryDet,SyNet, End-to-End DEtection TRansformer,YOLO-V5 and identify the small objects which is still a major challenge in the domain of computer vision.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). The above models have performed well for the object detection and the reason for selecting them is the same as they might perform well for small object detection as well.**

***Index Terms*—Keywords—EfficientDet,QueryDet,SyNet, end-to-end Detection Transformer,YOLO-V5,mAP,AR, 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.VisDrone dataset has been used for our project purpose One of the main challenges in small object detection is the imbalance between the size of the object and the size of the image. As the object size becomes smaller, the object occupies a smaller portion of the image, making it more challenging for traditional object detection methods to accurately detect and localize the object. Furthermore, small objects are often occluded by other objects in the scene or affected by complex background clutter, making it more challenging to identify and localize them. For current implementation,we will implement the above given models and evaluate the performances and score of the model with different baseline models.

II. Background/Literature Review

As per the research paper referenced, the models we observed need to have lesser FLOPs in order to meet our computational requirements as resources are limited. So model we choose like EfficientDet , QueryDet, YOLO-v5 were the one which were having fewer FLOPS and relatively good accuracy, the reason was the feature extraction was so robust in these model which may help in furthur identification of the object for complex image. Since the model perform poor in compare with the smaller object detection as the feature extraction was not that effective compare to other models.

A. *EfficientDet*

Efficient Detection was one of the light weight model developed by the Google as model can be scaled with higher resolution with different model from D0 to D7 and as BiFPN as feature pyramid network for extracting the features and baseline model as retinaNET-R50.

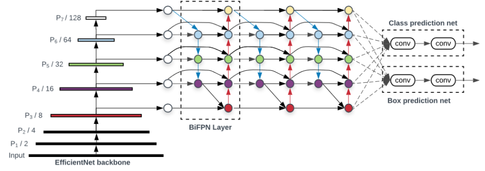


Fig. 1. Arch for EfficientDet

B. *QueryDet*

QueryDet is built on top of the popular Faster R-CNN framework, which consists of a backbone network that extracts features from the input image. The feature extractor details in QueryDet include a ResNet-based backbone network with four stages, each of which outputs feature maps at a different spatial resolution. The feature maps are then fed into a set of query-based attention layers, which perform feature aggregation and filtering based on learned query vectors. Finally, the output feature maps are passed to the RPN and region-based network for object detection as per figure

C. *SyNet(FasterRCNN)*

The architecture of SyNET consists of a backbone network, which extracts features from the input image, followed by a set of symmetric convolutional layers and objectness-aware modules. The output of these layers is then passed to a region

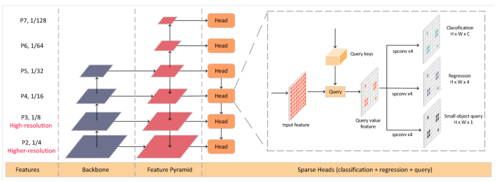


Fig. 2. Arch for YOLO V5

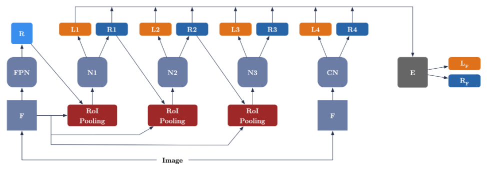


Fig. 3. Arch for SyNet

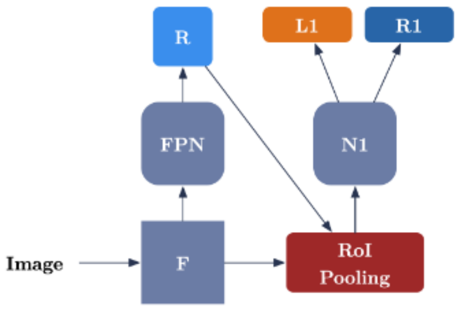


Fig. 4. Arch for Faster RCNN

proposal network (RPN) and a region-based network for object detection.

D. End to End DEtection TRansformer

This model was developed by the facebook developer team,DETR uses a transformer-based architecture.In DETR, the input image is first passed through a backbone network that extracts features from the image. The feature maps are then passed through a set of transformer layers that perform object detection and classification. Instead of using anchor boxes to generate candidate object locations, DETR generates a fixed set of object queries that are used to attend to different regions of the feature maps.

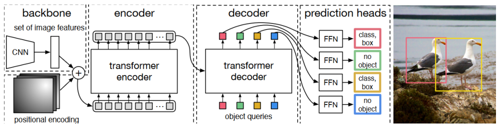


Fig. 5. Arch for End to End DEtection TRansformer

E. YOLO V5

YOLO-v5 uses a single neural network to perform object detection, which is trained end-to-end on a large dataset of annotated images. The network consists of a backbone, neck, and head, where the backbone extracts features from the input image, the neck processes the features to enhance object localization, and the head performs object detection and classification.

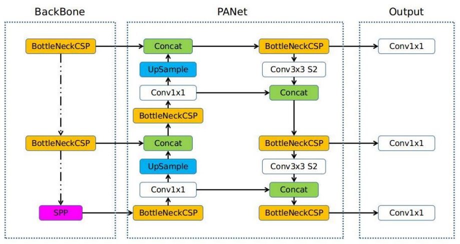


Fig. 6. Arch for YOLO V5

III. Implementation

A. Dataset

VisDrone dataset has been used for our research analysis. The VisDrone dataset has been widely used as a benchmark dataset for evaluating the performance of object detection and tracking algorithms in aerial images.The dataset consists of more than 6,000 images and 10 hours of video footage captured by drones, covering various scenes and scenarios, such as urban areas, forests, and crowded events but we have considered only aerial images for our project.

B. Data Conversion

The models that have been implemented were originally developed and trained for COCO dataset for most of the model like , but as it needed to be developed and trained on the drone dataset, the model has been developed and trained on the VisDrone dataset. The VisDrone dataset that has been used to train the models by the team is not in the same format as the COCO dataset, so it had to be converted to the dataset of VisDrone into the COCO dataset format. For this, the code for the conversion of data of VisDrone to COCO format has been provided by QueryDet-PyTorch's author. Here, one thing to be noticed is, YOLO V5 is trained in the VisDrone dataset by default.

C. Algorithms

We have implemented following Object Detection algorithms on our dataset for small object detection,

1) EfficientDet D0
2) Efficient D5
3) QueryDet
4) SyNet (Faster RCNN)
5) HoughNet
6) End to End DEtection TRansformer
7) YOLO V5

In available resources, the models that were successfully implemented are QueryDet, EfficientDet, SyNet, and YOLO V5.

D. Analysis

1) EfficientDet: For given Fig.7, The graph provided is about the behavior of EfficientDet over time after each training. One can see that the precision gradually gets better.

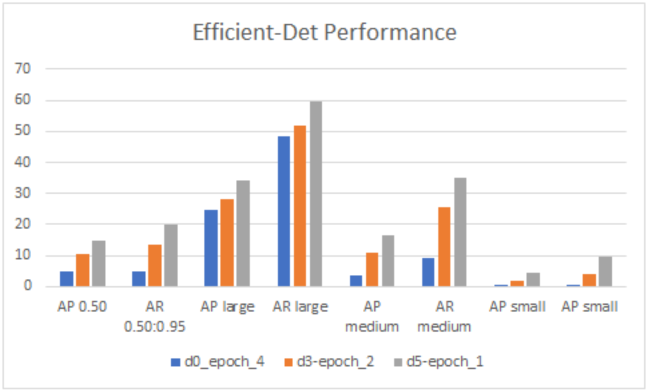


Fig. 7. Precision for EfficientDet

2) QueryDet: For given Fig.8, The graph provided is about the behavior of QueryDet over time after each training. One can see that the precision gradually gets better.

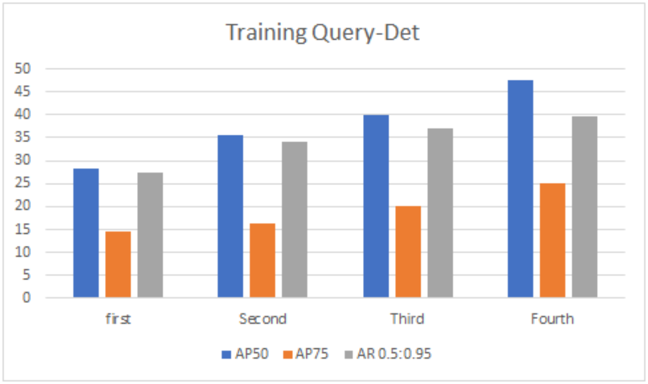


Fig. 8. Precision of QueryDet

For given Fig.9, now for the classes of QueryDet, the graph provided is providing information regarding the increase in class after each training.

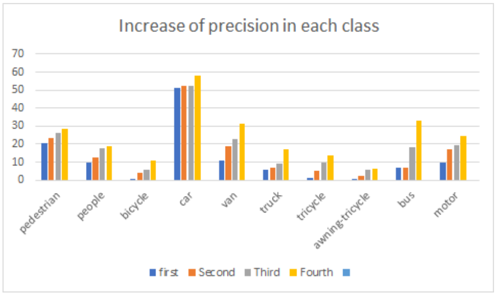


Fig. 9. Precision in classes of QueryDet

3) SyNet: For given Fig.10, The graph provided is about the behavior of SyNet over time after each training. One can see that the precision gradually gets better.

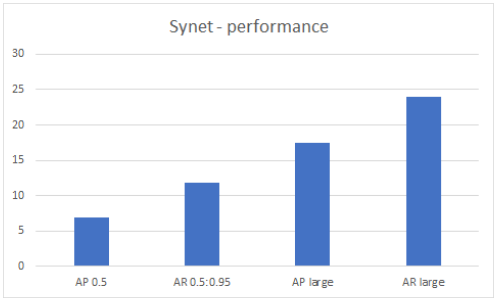


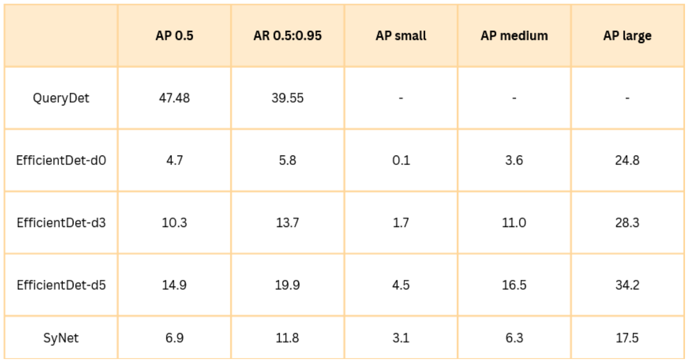
Fig. 10. Precision of SyNet

From the graphs, one can see how the precision value gets higher with training the model again and again. (Because of limitations of resources, else training model for sufficient time.) There would be an exponential rise in precision over time.

RESULTS

A. Accuracy

Models	AP_{small}	AP(0.5)	AR
QueryDet	-	47.48	39.55
EfficientDet-D0 (RetinaNet-R50)	0.1	4.7	5.8
EfficientDet-D3 (RetinaNet-R50+NAS-FPN)	1.7	10.3	13.7
EfficientDet-D5	4.5	14.9	19.9
SyNet (Faster RCNN)	3.1	6.9	11.8



	AP 0.5	AR 0.5:0.95	AP small	AP medium	AP large
QueryDet	47.48	39.55	-	-	-
EfficientDet-d0	4.7	5.8	0.1	3.6	24.8
EfficientDet-d3	10.3	13.7	1.7	11.0	28.3
EfficientDet-d5	14.9	19.9	4.5	16.5	34.2
SyNet	6.9	11.8	3.1	6.3	17.5

Fig. 11.

Conclusion

In summary, the study evaluated the performance of the small object detection algorithms - Query Detection, EfficientDet, SyNet using Faster RCNN as the backbone, End to End DEtection TRansformer and YOLO V5. For smaller object higher training computation is required and light weight models were not that effective since a higher level of detail and accuracy is required in detection.As per our analysis, the observed mAP of light weight model lies around 0.3 to 0.10 which is too less for a threshold of IOU 0.5 and larger object like car was able to predict the label correctly around 0.9 accuracy for faster models and the problem we faced was the lack of GPU resource due to which model was not able to cross the threshold of IOU 0.5 , for further future work, one can try to reduce other computational cost and can play with threshold for further analysis but problem of false-negative may arise which has to be taken care of.Due to multiple bounding box formed in one image and as image resolution is lower detection model are quite hard to detect as image in Visdrone are captured from a certain height and certain orientation.

Reference

[1] Tan, M., Pang, R., Le, Q. V. (2020, July 27). EfficientDet: Scalable and efficient object detection. arXiv.org. Retrieved February 25, 2023, from https://doi.org/10.48550/arXiv.1911.09070.

[2] Google. (n.d.). Automl/efficientdet at master · google/automl. GitHub. Retrieved February 25, 2023, from https://github.com/google/automl/tree/master/efficientdet

[3] Yang, C., Huang, Z., Wang, N. (2022, March 24). Querydet: Cascaded sparse query for accelerating high-resolution small object detection. arXiv.org. Retrieved March 1, 2023, from https://arxiv.org/abs/2103.09136.

[4] Yang, C. (n.d.). Chenhongyiyang/Querydet-Pytorch: [CVPR 2022 oral] Querydet: Cascaded sparse query for accelerating high-resolution small object detection. GitHub. Retrieved March 1, 2023, from https://github.com/ChenhongyiYang/QueryDet-PyTorch

[5] Albaba, B. M., Ozer, S. (2020, December 23). SyNet: An ensemble network for object detection in UAV images. arXiv.org. Retrieved March 24, 2023, from https://doi.org/10.48550/arXiv.2012.12991.

[6] Mertalbaba. (n.d.). Mertalbaba/SyNet: Object Detection with Ensemble Networks. GitHub. Retrieved March 24, 2023, from https://github.com/mertalbaba/SyNet

[7] Samet, N., Hicsonmez, S., Akbas, E. (2020, July 24). Houghnet: Integrating near and long-range evidence for bottom-up object detection. arXiv.org. Retrieved April 2, 2023, from https://doi.org/10.48550/arXiv.2007.02355.

[8] Nerminsamet. (n.d.). Nerminsamet/houghnet: [ECCV-20] official pytorch implementation of HoughNet, a voting-based object detector. GitHub. Retrieved April 2, 2023, from https://github.com/nerminsamet/houghnet

[9] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S. (2020, May 28). End-to-end object detection with Transformers. arXiv.org. Retrieved March 27, 2023, from https://doi.org/10.48550/arXiv.2005.12872.

[10] Facebookresearch. (n.d.). Facebookresearch/detr: End-to-end object detection with Transformers. GitHub. Retrieved March 27, 2023, from https://github.com/facebookresearch/detr

[11] jocher, glenn. (n.d.). Ultralytics/yolov5: Yolov5 in PyTorch ONNX CoreML TFLite. GitHub. Retrieved April 10, 2023, from https://github.com/ultralytics/yolov5.