Final Year Project

Interim Report

Fast and Efficient Model for Real-Time Tiger Detection In The Wild

Ong Chong Jun, Gregory (U2021112E)

Under mentorship of Associate Professor Deepu Rajan

School Of Computer Science and Engineering

Overview Of Problem

For species biodiversity to be maintained, wildlife conservation is essential. When endangered species are not protected, ecosystems may become unbalanced and the environment may suffer. Accurate monitoring of the geographic distribution and population health of these endangered species is increasingly important to this aim, especially in light of poaching and habitat destruction. Traditional approaches to transmitter placement on animals are prone to sensor failure, have trouble scaling to huge populations, and make it impossible to evaluate how the wildlife interacts with its surroundings. With the use of unmanned aerial vehicles or camera traps to gather visual data, computer vision techniques are a potential approach to monitoring wildlife.

To get precise population counts and track the movement of wildlife, detection is a crucial vision technique. However, a number of obstacles prevent the deployment of such systems.

To begin with, the resource limitations on the edge camera necessitate low power and precise tiger recognition to initiate the image capture and prevent massively unrelated image recording from using storage card space and battery life. Significant progress has been made in the field of object detection as a result of recent deep learning innovations. However, the majority of cutting-edge techniques involve a multi-stage methodology or complex deep learning network topologies that need large amounts of computational power.

Current Implementation

For the dataset, I am using 4,434 images of high resolution (1920x1080) of Tigers with 9,496 bounding boxes. To reduce storage, power, and networking consumption the data contains only a subset of frames with at least one Amur Tiger present. The annotations are provided in PASCAL VOC format with bounding boxes assigned to each frame.

For the model, I have explored and implemented the lightweight model that is proposed in the report here[1].

[1] Fast and Efficient Model for Real-Time Tiger Detection In the Wild
https://drive.google.com/file/d/1FUqstTv3lJTgigxOXhClqSrBG-4OnKLR/view?usp=drive_link

For a brief summary, the design of the model is built upon the Feature Pyramid Network(FPN) and heavily rely on Depthwise Separable Convolutions in addition to the lightweight FD-MobileNet backbone(**Table 1**) to increase efficiency.

To further elaborate, FPN improves object identification and recognition at different scales in an image. In order to recognise objects of various sizes, it builds a pyramid of multi-scale feature maps. Through a top-down and bottom-up approach, FPN integrates high-level, semantically rich information with low-level, high-resolution features. By providing a thorough representation of a picture through this integration, FPN is better able to do tasks like object detection, segmentation, and other tasks where objects may appear at various scales. Convolutional neural networks can successfully comprehend and interpret images containing objects of various sizes and settings thanks in large part to the FPN architecture.

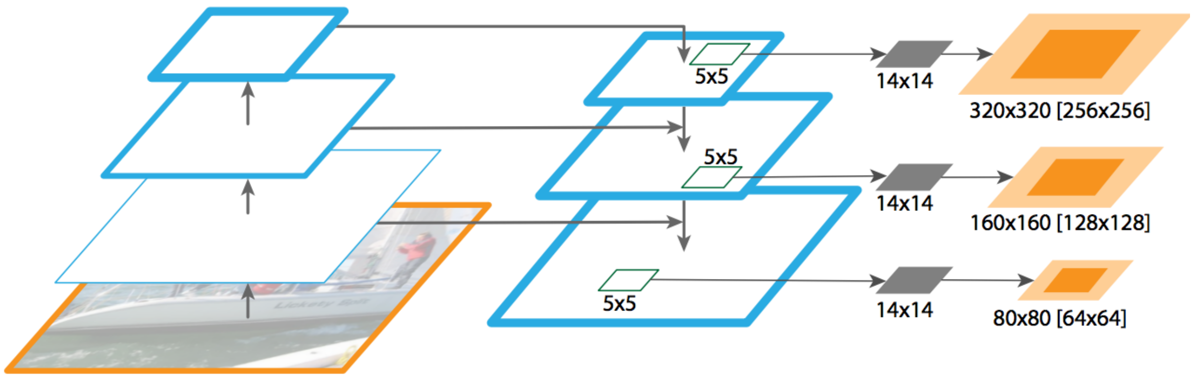
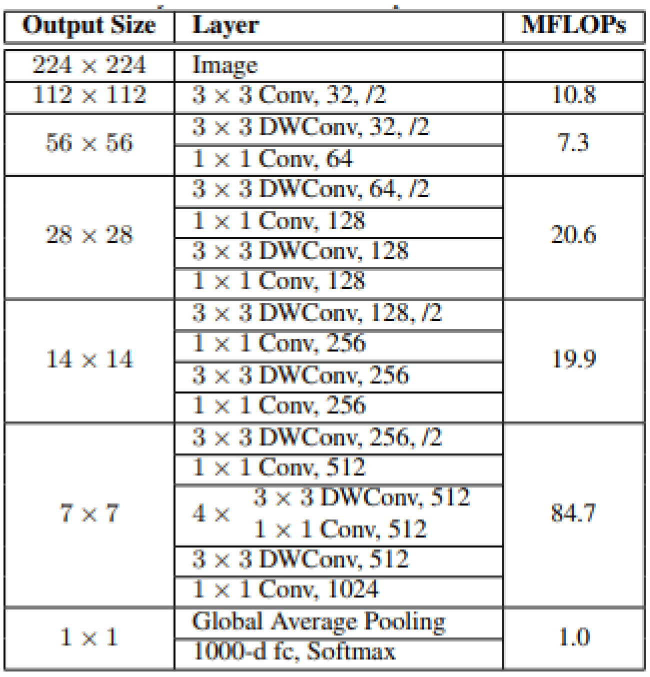


Figure 1:Generalised diagram to show how FPN roughly works



Output Size	Layer	MFLOPs
224 × 224	Image	
112 × 112	3 × 3 Conv, 32, /2	10.8
56 × 56	3 × 3 DWConv, 32, /2	7.3
	1 × 1 Conv, 64	
28 × 28	3 × 3 DWConv, 64, /2	20.6
	1 × 1 Conv, 128	
	3 × 3 DWConv, 128	
	1 × 1 Conv, 128	
14 × 14	3 × 3 DWConv, 128, /2	19.9
	1 × 1 Conv, 256	
	3 × 3 DWConv, 256	
	1 × 1 Conv, 256	
7 × 7	3 × 3 DWConv, 256, /2	84.7
	1 × 1 Conv, 512	
	$4 \times \begin{matrix} 3 \times 3 \text{ DWConv}, 512 \\ 1 \times 1 \text{ Conv}, 512 \end{matrix}$	
	3 × 3 DWConv, 512	
	1 × 1 Conv, 1024	
1 × 1	Global Average Pooling	1.0
	1000-d fc, Softmax	

Table 1: FD-MobileNet Architecture. "/2" indicates the stride of the layer is 2. DWConv: depthwise convolution

Due to their efficiency gains, fast downsampling and depthwise convolution are employed by FD-MobileNet. Fast downsampling decreases the spatial dimensions of feature maps quickly using procedures like pooling and strided convolution. This benefits object detection in particular by enabling faster computation, lower memory consumption, and greater localization accuracy. Similarly, depthwise convolution separates the convolution process into two parts, thereby significantly lowering the number of parameters. As a result, models become smaller, quicker, and more memory-efficient. It is especially helpful in circumstances with limited resources since it efficiently collects geographical data and reduces computing complexity. These 2 methods improve the speed and resource utilisation of neural networks, making them suitable for real-time applications and low-power devices.

In order to create larger feature maps in the final layers, we also modify the RetinaNet architecture from its initial design. This change effectively lowers processing requirements by allowing us to train the network effectively even on smaller image dimensions.

I implemented the model using PyTorch. 70% of images are randomly selected as the training set and the remaining 30% as the validation set to prevent the model overfitting. Training was done using my laptop's GPU (GTX 1650 with Max-Q Design) , with Adam optimizer and the learning rate of 10^-3. Early Stopping criterion with mAP on Validation set for 15 epochs was used and the learning rate is reduced by a factor of 2 each time the validation mAP stopped increasing for 5 epochs.

Using the above configuration, I was able to achieve a validation_mAP of **0.8074** after 50 epochs. Testing it on random images as test data, the model is able to detect the tigers with almost perfect accuracy with a confidence score of about 0.99 (**Figure 2**).

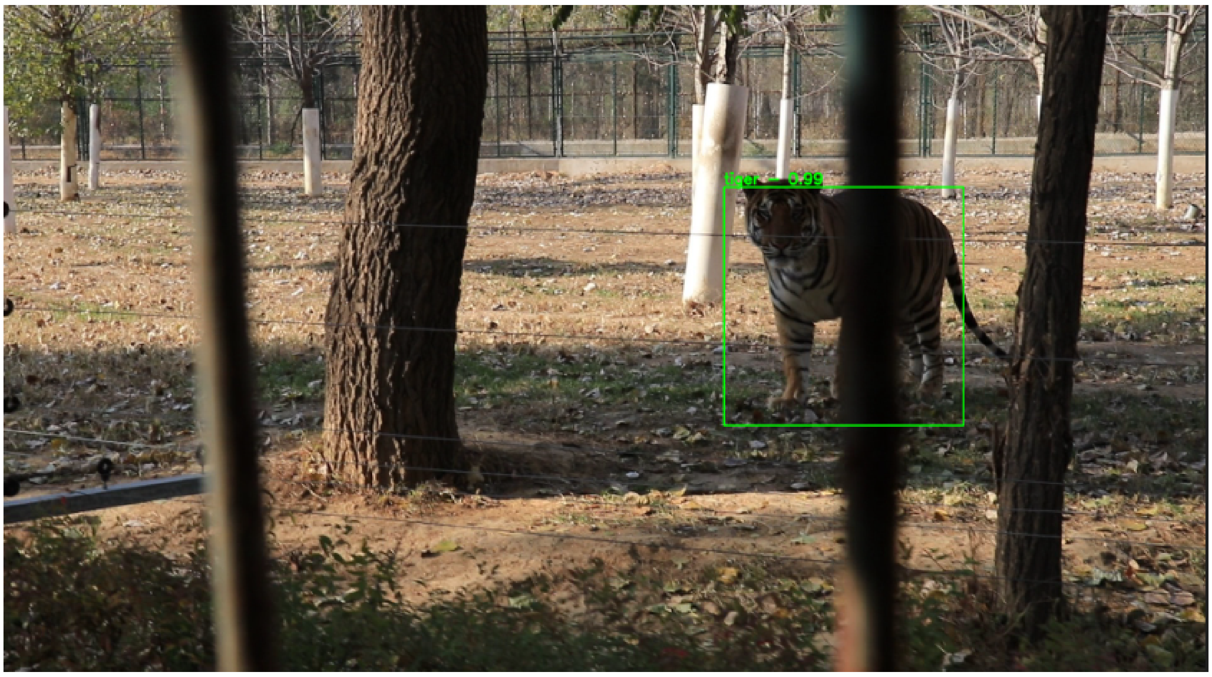


Figure 2: Bounding box with confidence score plotted by the model

This shows that the model is very accurate in detecting tigers. In the wild, both accuracy and time taken to detect tigers are both important. That is because animals may not linger around the camera but instead may be moving past the frame of the camera that is set up in the wild. Hence, the model that we use should be not only accurate but also fast in detection.

Ongoing Work

This brings about the point on measuring the timing it took for detection for the model. I am currently trying to measure the time it took for detection and also the time taken for different parts of the architecture so that I can investigate parts which are taking longer. I am also currently implementing another lightweight model, EfficientNet-lite and am looking to measure the accuracy and the time taken for detection. Hopefully, by comparing the accuracy and time taken for detection for both models, we can draw up some conclusions about what EfficientNet-Lite is doing well in and potentially, suggest changes to improve the architecture of the original FD-mobileNet.

Conclusion

In the wild, I believe that the time taken for detection is very important. I believe that it may even be a good idea to sacrifice some accuracy for faster detection if the accuracy is sufficiently high enough. Since the original model has relatively high accuracy and confidence score, I believe we should try to improve the detection speed of the model and by looking into EfficientNet-Lite, it may provide us with some insights on how we can do so.