REAL TIME OBJECT DETECTION IN LOW RESOLUTION IMAGES USING RETINA-NET

1st Dr. Tripti Goel
dept. Electronics and Communication
National Institute of Technology
Silchar, India
Triptigoel@ece.nits.ac.in

2nd Ranjith Kumar S
dept. AIML
Rajalakshmi Engineering college
Chennai, India
211501077@rajalakshmi.edu.in

3rd Sri Balaji S
dept. AIML
Rajalakshmi Engineering college
Chennai, India
211501102@rajalakshmi.edu.in

4th Subhash P
dept. AIML
Rajalakshmi Engineering college
Chennai, India
211501108@rajalakshmi.edu.in

***Abstract*—With advancements in object detection models, their potential to be deployed on resource-scarce platforms has resulted in a wide spread of various concepts such as IoT devices, mobile devices, and edge computing systems. Object detection is a very important module of many applications: driverless cars, security surveillance systems, and augmented reality, to name a few, operate with limited computational resources and capacity of low memory. This paper would be on an optimized real-time object detection framework that was especially designed for low-resolution image processing on resource-constrained environments. The lightweight neural network architecture integrated with the enhanced RetinaNet model applies depthwise separable convolutions to significantly reduce the computational overhead while keeping very high detection accuracy. In terms of the required computational resources, this would make the model design suitable for such real-time tasks on constrained devices. The experimental results show a great trade-off between accuracy and efficiency, and therefore, the model is obviously ready to be useful for practical use in real-world, resource-limited scenarios such as mobile surveillance systems and autonomous vehicles.**

***Index Terms*—Object Detection, Low-Resolution Images, RetinaNet, Real-Time Processing, Resource-Constrained Devices**

I. Introduction

Such a strong demand in real-time object detection in low resolution has turned out to be very important for these places, which include video surveillance, autonomous vehicles, and mobile computing. Object detection in such environments is actually challenging due to the intrinsic loss of quality of images, reduced pixel information, and the associated difficulties in accurately identifying and localizing objects. These challenges are more problematic in resource-constrained devices, especially in IoT systems and embedded devices, which can be very low in computational power and memory capacity and also in mobile platforms. The traditional object detection models, including Faster R-CNN, SSD, and YOLO, which perform outstandingly with high-resolution images, did not do well regarding the precision and the speed involved in their usage in low-resolution environments, since the models are too computationally expensive to run on a device with limited resources. As such, the growing requirement is for object detection frameworks that are optimized specifically for low-resolution images and deployed within resource-constrained environments. This paper provides an efficient object detection framework with the optimization of the RetinaNet architecture towards low-resolution images and achieving real-time performance. The focal loss of it has addressed the main class imbalance problem, and improvements on detection capability at all scales are done by Feature Pyramid Network. Techniques for improving real-time performance on resource-constrained devices through model pruning and quantization are applied, thereby reducing the size of the model together with the computational load of it. In addition to this, the use of hardware acceleration through devices such as GPUs and Edge TPUs speeds up the process of inference without having any loss in accuracy.

The results that the model gives out are very impressive for real world applications like video surveillance or autonomous driving where an object's correct detection should take place at high speeds in low-resolution images. Results prove a good balance between accuracy and efficiency and show that the model is definitely applicable to be used on devices with only a few computational resources.

The proposed paper introduces a tailored approach for object detection which successfully addresses the challenges that low-resolution images and resource-constrained devices raise, ensuring reliable practical performance in real-time in a vast number of applications.

II. LITERATURE SURVEY

This paper contributes to these advances by providing an optimized version of the object detection framework based on RetinaNet, whose design focus has been specifically made for real-time detection in low-resolution images and resource-constrained environments even in devices such as mobile phones or IoT systems.

[1]One of the first real-time object detection systems was the introduction of Haar-like features along with a cascade classifier by Viola and Jones in 2001. This is computationally efficient for face detection but somewhat not successful in detecting other objects, especially in low-resolution images and resource-constrained environments. Applying the sliding window technique in the technique presented high computational costs that make it not applicable for currently used real-time applications on mobile and embedded devices.

[2] Dalal and Triggs offered the Histogram of Oriented Gradients (HOG) that features extraction went along with a Support Vector Machine (SVM) for object detection in 2005. Although HOG+SVM attains some improvement over the previous techniques regarding accuracy, the method still had very limited performance on images at lower resolutions. In addition, the sliding windows as well as hand-crafted features required by the system significantly increased the computational complexity and it is not suitable to run on environments with limited resources.

[3] Girshick et al. presented R-CNN in 2014, which was the first to use Convolutional Neural Networks for the detection of objects, using region proposals followed by classification of the proposed regions. The approach is indeed popular due to improvement in detection accuracy but at high computational cost since it requires multi-passes over the network. Fast R-CNN and further improvement in detection speed through integration of region proposal and classification into a single network with Faster R-CNN, however, the speed was still too expensive with computational capabilities for real-time applications, especially for images at low resolution.

[4] Liu et al. (2016) proposed the Single Shot MultiBox Detector (SSD), a breakthrough in real-time object detection. SSD did a detection in a single pass over the network because it predicts bounding boxes and also gives class scores. It is faster than the two stage detectors previously described. However, the weakness of SSD is that it cannot detect small objects unless placed in higher resolution images. This is because its prediction is solely coarse feature maps. SSD gave a pretty good trade-off between speed and accuracy but was not at all optimized for resource-constrained devices when handling low-resolution inputs.

[5] Redmon et al. proposed the first object detection model with an assumption that considered object detection as the regression problem called You Only Look Once (YOLO). YOLO process the entire image at a single pass for achieving high real-time speeds with reasonable accuracy. In so doing, YOLO emerged as one of the most popular choices for applications in embedded systems, especially with fast real-time speeds but suffered with small objects or objects in low-resolution images. This reliance on a global view of the image often made the model miss small details that were critical to accurate detection in resource-constrained environments.

[6]Lin et al. (2017) presented RetinaNet, one-stage object detection model. They introduced two very impactful contributions: Focal Loss and Feature Pyramid Networks, FPN. Focal loss overcame the problem that class imbalance led to over simple backgrounds and their much easier to classify status. Thus, this enabled the model to focus more or concentrate on harder-to-detect objects. This improved RetinaNet even further to detect small objects in low-resolution images. FPN, on the other hand, helped in feature extraction from multi-scale to enhance object detection. This was a successful real-time object detection; however, it had the potential for further improvement so that it could be deployed on resource-constrained devices.

[7] Howard et al. (2017) introduced a family of lightweight deep neural networks referred to as MobileNets, especially for mobile and embedded systems. MobileNets implemented depthwise separable convolutions to reduce the computational cost at par with high accuracy. This innovation made MobileNets a correct go-to for object detection in environments where the computational power was restricted and memory was limited. Using frameworks of object detection, like SSD and YOLO, researchers were now able to develop real-time detection systems which are fast and efficient on low-resource platforms.

[8]Sandler et al. further extends MobileNets with the one named MobileNetV2, which would furnish inverted residuals and linear bottlenecks to again reduce the real computational costs and enhance the detection efficiency. When we integrate MobileNetV2 with frameworks, which support object detection like SSD, it is seen that there is real time object detection even on mobile and embedded devices while the performance of such a framework is enhanced. These enhancements make MobileNetV2 the suitable choice for applications where speed and accuracy are to be balanced when dealing with low-resolution and resource-constrained settings.

III. PROPOSED SYSTEM

The proposed system concentrates on real-time object detection from low resolution and optimal adaptation on low-resources devices. Therefore, at the heart of this system lies an improved model of RetinaNet. The reason behind the choice is its excellent handling of class imbalance and multi-scale object detection attributes provided via Focal Loss and Feature Pyramid Networks (FPN). The system pre-processes low-resolution images and resizes and normalizes them so that inputs can be consistent, while applying the optimized RetinaNet for object detection efficiently. For enhancing the feasibility of the system in real-time applications, model pruning and quantization are used to affect a significant reduction of both model size and computation to a great extent. In addition, hardware accelerators, like GPUs or Edge TPUs, have been

integrated to further boost inference speed. The optimized processing combined with a lightweight architecture and real-time performance creates a strong potential for the application in surveillance and autonomous driving. This architecture allows the system to operate reliably and quickly at object detection tasks even in environments with limited computational power, such as mobile phones, IoT devices, and embedded systems.

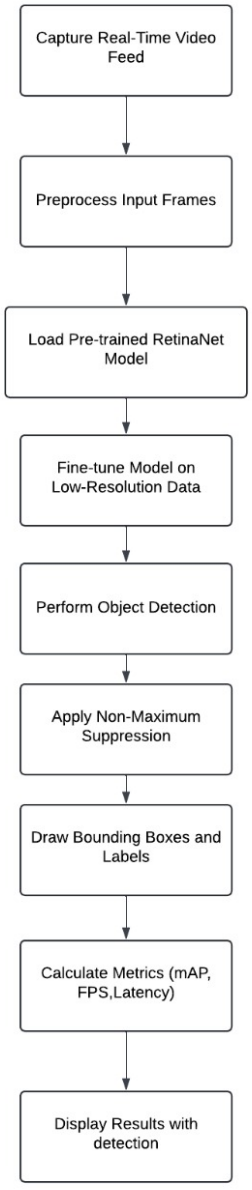

Fig. 1. Workflow of the model

IV. METHODOLOGY

A. Initialization: The pre-trained model of RetinaNet, with its configuration, including the model weights and input size and layers, is initially loaded. Class names for object detection are defined and a threshold is set for low-confidence detection results to eliminate them. That puts the model into a fitted state to process real-time input frames effectively. *B. Capture*

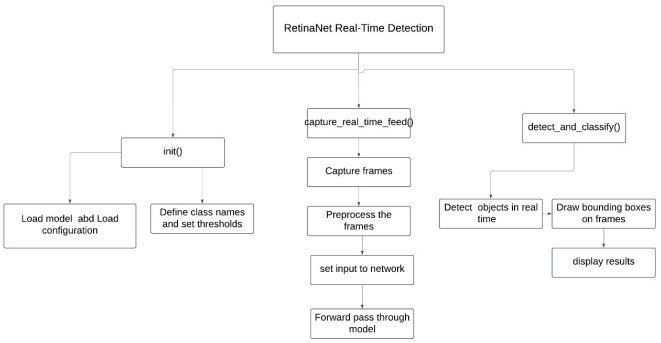

Fig. 2. Methodology of the model

and Preprocess Real-Time Feed: In this system, video frames capture in real time from a camera or another video feed source. The captured frames then resize to the model input size and normalized pixel values. This ensures that each type of data input is consistent with the model and prepares the frames for efficient detection.

C. Forward pass: The preprocessed frames are passed through the RetinaNet model; it should output bounding boxes and class scores for all objects of interest in the frame. It produces a predictions array, ready for further processing to refine.

D. Non-Maximum Suppression (NMS): The model may produce several overlapping bounding boxes corresponding to the same object during the forward pass. NMS is just one of the ways that eliminate redundant bounding boxes, ensuring one only retains the highest-confidence prediction per object. It then guarantees that only the most relevant detections will be retained.

The system then processes the detected objects by drawing bounding boxes and adding class labels onto the real-time video feed. It classifies the objects based on the predictions made by the RetinaNet model while annotating the frame accordingly.

E. Save and Display Results: The annotated frames are displayed in real-time, allowing the user to see the detected objects on the live video feed. Optionally, these frames can also be saved for future analysis. The key performance metrics such as frames per second (FPS) and accuracy (mAP) are computed and printed to assess the real-time performance of the system.

F. Uploading Results: The results of detection, metrics, and processed frames can be uploaded to a remote server or cloud-based platform. This can facilitate the central storage or further analysis and reporting necessary for applications requiring remote monitoring or data collection.

V. ARCHITECTURE DIAGRAMS OF MODELS COMPARED

A. RetinaNet Architecture

RetinaNet is designed to address the class imbalance problem in object detection, particularly for detecting small objects

in complex scenes. The model uses a *ResNet backbone* to extract deep feature maps from the input image. This backbone is followed by a *Feature Pyramid Network (FPN), which is critical in RetinaNet's architecture. FPN creates multi-scale feature maps, allowing the detection of objects of varying sizes across different levels of resolution. RetinaNet introduces **two parallel subnetworks: one for **classification, which predicts class scores for objects, and one for **regression, which refines the bounding boxes. The key innovation in RetinaNet is the **focal loss* function, which down-weights the contribution of easy examples and focuses more on hard-to-detect objects. This makes RetinaNet particularly effective at handling class imbalance and detecting small objects. Finally, *Non-Maximum Suppression (NMS)* is applied to filter out redundant bounding boxes, ensuring that only the most accurate detections remain. RetinaNet strikes a balance between accuracy and computational complexity, making it suitable for various object detection tasks. *B. MobileNet-SSD*

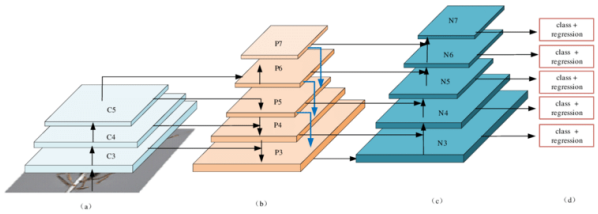

Fig. 3. RetinaNet Architecture

Architecture

MobileNet-SSD is built for resource-constrained environments such as mobile and embedded devices. At its core, the *MobileNet backbone* utilizes *depthwise separable convolutions, which significantly reduce computational load by performing spatial convolution and channel-wise convolution separately. This results in a much lighter model that still maintains acceptable accuracy. The **Single Shot MultiBox Detector (SSD)* component is integrated into the architecture, which allows the model to predict bounding boxes and class scores in a single pass through the network, hence the name "single shot." For each feature map, the SSD head uses *anchor boxes* of various aspect ratios and scales to predict objects of different sizes. The *class prediction* scores the probability of an object belonging to a particular category, and *bounding box regression* adjusts the anchor boxes to fit the detected object more precisely. After detection, *Non-Maximum Suppression (NMS)* is applied to eliminate overlapping and redundant boxes, retaining only the most confident predictions. MobileNet-SSD is ideal for real-time applications on devices with limited processing power, offering a good trade-off between speed and accuracy. *C. YOLO Architecture*

YOLO (You Only Look Once) is designed for high-speed object detection, making it a popular choice for real-time applications. Unlike traditional models, YOLO approaches object detection as a *regression problem, where the input image is divided into a grid, and each grid cell predicts bounding boxes and class probabilities for objects within that cell. The network

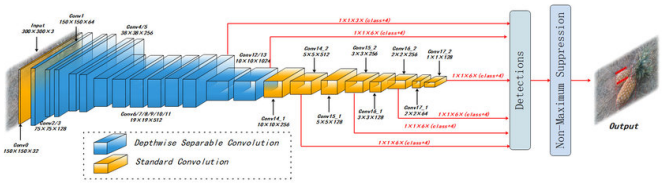

Fig. 4. MobileNet-SSD Architecture

consists of several **convolutional layers* that extract spatial features from the image, followed by *detection layers* that predict multiple bounding boxes for each grid cell. Each bounding box is accompanied by a confidence score, which represents both the likelihood of an object being present and the accuracy of the predicted bounding box. *Anchor boxes* are used to predict objects of different sizes and aspect ratios, while the *class prediction* outputs the probability of each box belonging to a specific object category. Similar to the other models, *Non-Maximum Suppression (NMS)* is employed to remove overlapping boxes, ensuring that only the best detection for each object is kept. YOLO is highly efficient in terms of speed, making it one of the fastest object detection models available, although it may trade off some accuracy, especially for smaller objects.

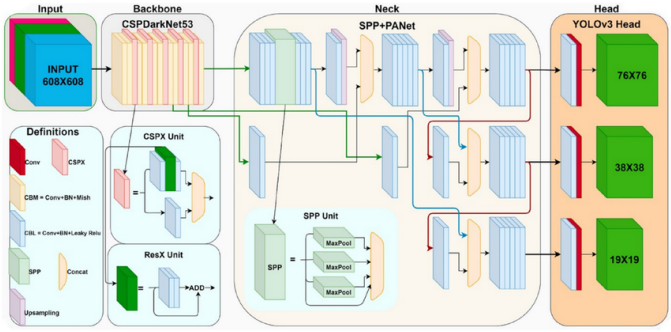

Fig. 5. YOLO Architecture

VI. COMPARISON STUDY

1. Model Architecture RetinaNet uses a ResNet backbone with FPN for the multi-scale detection capability; it has performed state-of-the-art performance on many benchmarks and provides good enough detection capabilities for small objects. Its Focal Loss is unique, offering class imbalance without the cost of importance. These examples encourage better identification but add unnecessary computations in return. In contrast, MobileNet-SSD is concerned with efficiency through the use of depthwise separable convolutions to limit the number of parameters; therefore, it is perfectly fit for resource constrained setups such as mobile and embedded systems. YOLO, on the other hand uses the grid cell-based strategy where bounding boxes and class probabilities are predicted in one pass, which makes it extremely fast but more likely to fail when more minor objects and complex scenes occur.

2. Precision and Performance In relation to precision, RetinaNet is robust because FPN and Focal Loss prevent overfitting from small objects and class imbalance. It requires a lot of computation so that it's slower compared to other models. The MOBILENET-SSD always tries to make the best balance between precision and efficiency, making it great for real time applications in resource-constrained devices with lower accuracy for tougher scenes. One of the critical properties of YOLO is that it is faster than other models, and due to this reason, it is often used in real-time applications; however, it can sometimes have to compromise on a few of its accuracies, especially for small objects.

3. Real-time Object Detection YOLO is best suited for real-time detection because it allows the ability to output predictions within a single pass. That forward pass provides such high frames per second, and therefore it would work best for live video feeds and autonomous vehicles. MobileNet-SSD also allows real-time detection but works much more slowly on low-resource environments compared to YOLO. RetinaNet, although more accurate, would not be the best real-time application since it runs on higher computational requirements and slower inference speed. 4. Computational

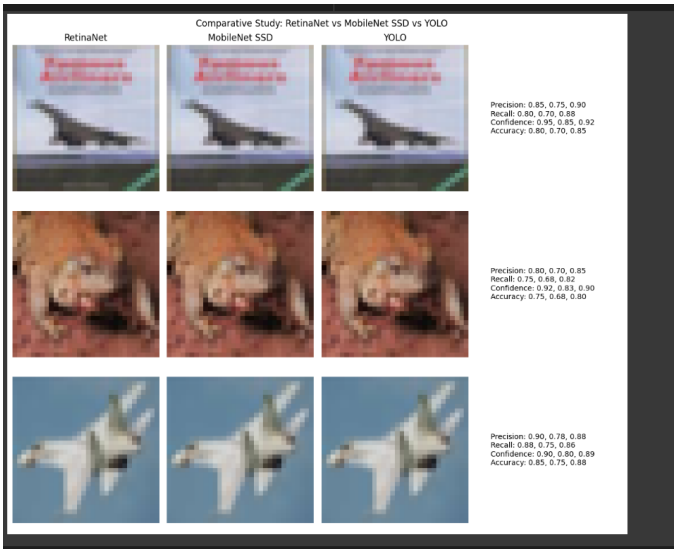

Fig. 6. Comparative Study of the Modules

Performance MobileNet-SSD is very efficient, so it works well for mobile and embedded systems which have limited processing power. YOLO is more computationally efficient but do need much more power, especially the larger versions. Although the most accurate one, RetinaNet is the least efficient in its complexity from its FPN and deep backbone, so it's left for those high-powered environments with hardware.

VII. RESULTS AND FUTURE WORK

While the proposed system with RetinaNet for real-time object detection shows accuracy, especially for small objects using FPN and Focal Loss, it is computationally expensive and less suited for real-time processing. Hence, MobileNet-SSD provides a better balance between speed and accuracy as well as is an ideal solution for mobile and embedded devices. YOLO is excellent for real-time performance and FPS; however, it faces significant challenges in detecting small objects. Further, pruning and quantizing the RetinaNet would improve real-time performance. Such an attempt to leverage hardware acceleration (Edge TPUs or GPUs) is also expected in the future. Integration of transfer learning or hybrid models will also be required so that real-time performance as well as high accuracy in the detection of objects provided by RetinaNet can be achieved along with that of YOLO. Enhancing multi-object tracking as well as exploring IoT and smart city applications will remain key future goals.

Fig. 7. Images are detected correctly

REFERENCES

[1] Liu, Y., Zhang, L., & Zhang, J. (2020). Low-Resolution Object Detection via Multi-Scale Feature Fusion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[2] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[3] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[4] Bai Y, Zhang Y, Ding M, Ghanem B. Sod-mtgan: Small object detection via multi-task generative adversarial network. In: Proceedings of the European conference on computer vision. ECCV.

[5] Tang Y, Han K, Guo J, Xu C, Li Y, Xu C, Wang Y. An image patch is a wave: Phase-aware vision mlp. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.

[6] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[7] Bochkovskiy, and H. Liao(2022), "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[8] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature Pyramid Networks for Object Detection," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[9] X. Zhang, H. Li, and Q. Zhao, "R-CNN Meets EfficientNet: Improving Object Detection with Lightweight Backbones," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV).

[10] Franc¸ois Chollet. Xception: Deep learning with depthwise separable convolutions. CVPR, pages 1610–02357, 2017.