# **Comparative Study of Models for Facial Mask Detection**

##### Atharva Unmesh Raut, Rashi Haresh Rupchandani, Dhanur Jeetendra Sharma, Atharva Hiwarekar, Anuraag Kulkarni, Priyanshu Shrivastava

Department of Computer Science

Rutgers - The State University of New Jersey

***Abstract* -** This document outlines the report for our final project for CS536: Machine Learning. We’ve collated the descriptions and results of our comparison between 3 models as listed.

1. YOLOv7

2. GRCNN **(**Detectron2)

3. RetinaNet

I. Problem Statement

The problem we aim to address is the issue of detecting whether individuals in a large crowd are wearing masks or not. This is important as wearing masks is a crucial precautionary measure to prevent the spread of infectious diseases like COVID-19. The traditional manual method of checking compliance with mask-wearing can be time-consuming and requires a large workforce. Therefore, automating this process using state-of-the-art models like YOLOv7 can help detect non-compliant individuals in real time with high accuracy. By benchmarking YOLOv7 against other leading models like Detectron2 and RetinaNet, the study aims to compare the performance of different models for mask detection and identify the most effective one. Overall, this solution can help in ensuring public safety and minimizing the risk of future pandemics by enforcing compliance with mask-wearing protocols.

II. Methodology

In this project, we'll put many models into practice to address the facial mask recognition issue and evaluate their effectiveness using the Facial Mask Dataset.

These models would be benchmarked based on their accuracy and performance. Our goal is to determine the most efficient and accurate model for facial mask detection, which could potentially be used in real-world scenarios to help enforce the wearing of masks in public places.

III. Dataset

The dataset we are working with contains 853 RGB images of individuals and multiple groups of people belonging to 3 classes:

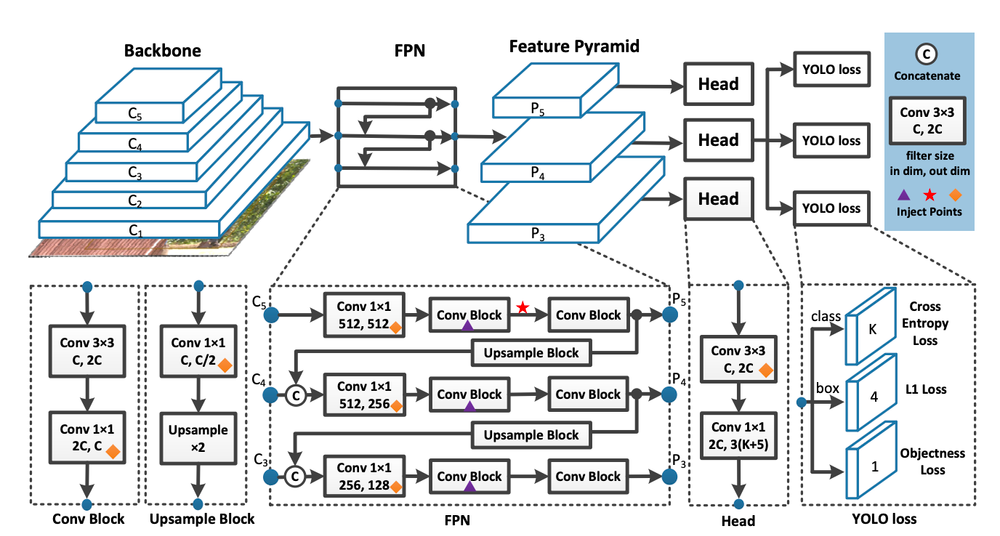
1. Wearing a mask

2. Not wearing a mask, and

3. Wearing mask incorrectly.

IV. Implementation

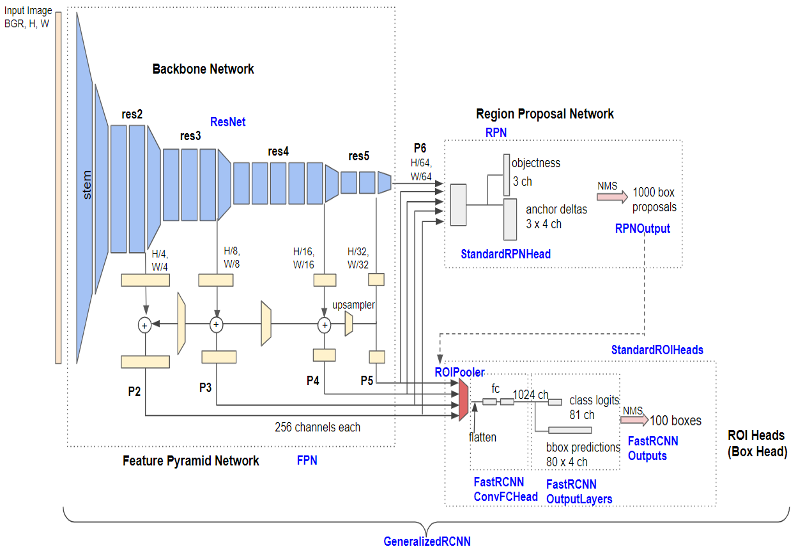
**YOLOv7**



YOLO stands for “You Only Look Once”, it is a popular family of real-time object detection algorithms. The original YOLO object detector was first released in 2016. It was created by Joseph Redmon, Ali Farhadi, and Santosh Divvala. At release, this architecture was much faster than other object detectors and became state-of-the-art for real-time computer vision applications. Since then, different versions and variants of YOLO have been proposed, each providing a significant increase in performance and efficiency. YOLOv7 is the latest official YOLO version created by the original authors of the YOLO architecture. It provides a greatly improved real-time object detection accuracy without increasing the inference costs. As previously shown in the benchmarks, compared to other known object detectors, YOLOv7 can effectively reduce about 40% of parameters and 50% computation of state-of-the-art real-time object detections, and achieve faster inference speed and higher detection accuracy. In general, it provides a faster and stronger network architecture that provides a more effective feature integration method, more accurate object detection performance, a more robust loss function, and an increased label assignment and model training efficiency. As a result, YOLOv7 requires several times cheaper computing hardware than other deep learning models. It can be trained much faster on small datasets without any pre-trained weights.

**GRCNN (Detectron2):**

The Generalized Region-based Convolutional Neural Network (GRCNN) algorithm using the Detectron2 framework developed by Facebook is a state of the art 2 stage object detection model that helps isolate and identify objects in an image with high accuracy. (Illustrated). This is a type of object detection algorithm that can detect and classify objects within images. The architecture comprises three main components: the backbone, the Region Proposal Network (RPN) and Region of Interest (RoI) Heads.



Backbone:

The backbone of the model is a ResNet network with a Feature Pyramid Network (FPN) attached to it. The ResNet network serves as a feature extractor, extracting meaningful features from the input image. The FPN network is used to combine features from multiple levels of the ResNet network, to allow the model to detect objects at different scales.

The FPN network has five stages (fpn\_lateral2, fpn\_output2, fpn\_lateral3, fpn\_output3, fpn\_lateral4, fpn\_output4, fpn\_lateral5, fpn\_output5). Each stage consists of a lateral convolutional layer and an output convolutional layer. The lateral convolutional layer takes the feature map from the previous stage and reduces its number of channels to 256. The output convolutional layer takes the lateral feature map and applies a 3x3 convolution to it, which preserves the spatial resolution but reduces the number of channels. The output feature map from each stage is then upsampled and added to the input feature map of the next stage to produce the final feature pyramid.

Region Proposal Network

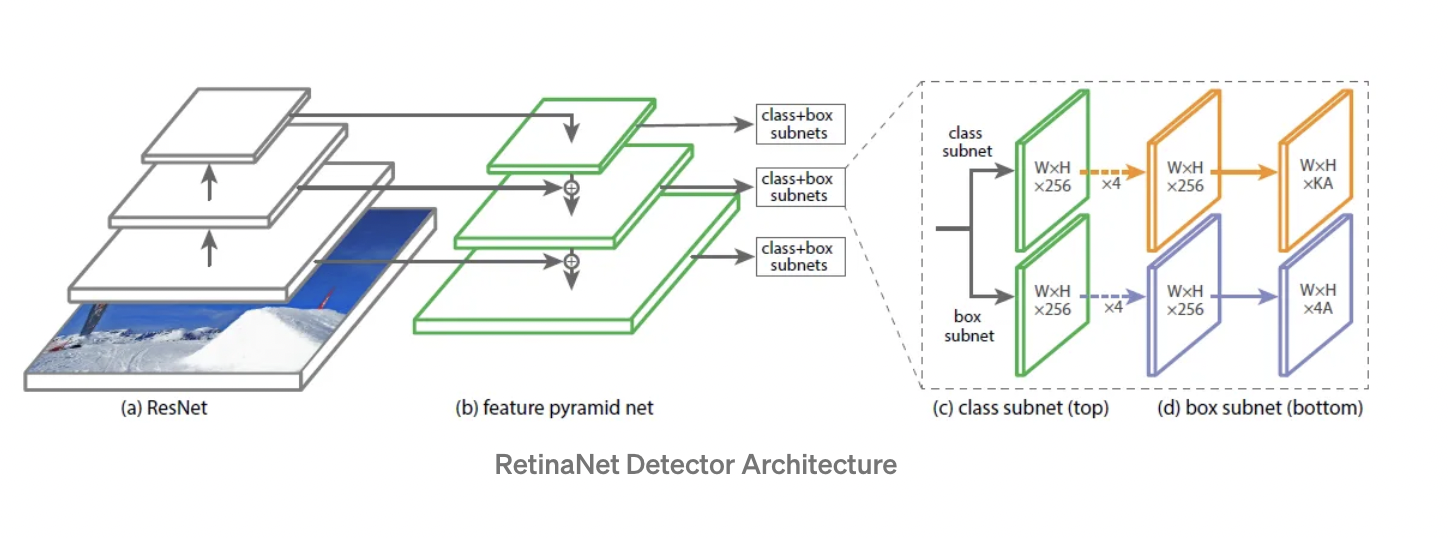
The RPN component of the model takes the feature pyramid produced by the backbone and generates region proposals. These region proposals are areas of the input image that the model believes may contain an object. The RPN is composed of two fully connected layers that produce a set of objectness scores and bounding box regression offsets for each anchor box at each spatial location in the feature pyramid. The anchor boxes are predetermined boxes of different sizes and aspect ratios that cover the input image at different scales. The RPN uses these anchor boxes to generate region proposals.

Region of Interest (RoI) Heads:

Overall, this architecture allows the model to detect objects within images by extracting features from the input image using a ResNet backbone and a Feature Pyramid Network, generating region proposals using an RPN, and classifying and refining the proposed regions using a region-based CNN.

**RetinaNet**

RetinaNet is one of the best one-stage object detection models that has proven to work well with dense and small-scale objects. RetinaNet was created by improving two existing single-stage object detection models: Feature Pyramid Networks (FPN) and Focal Loss.



RetinaNet Architecture: RetinaNet can be divided into three components:

1.Backbone Network (e.g., ResNet + FPN)

2. Object Classification Sub-Network

3. Object Regression Sub-Network

The backbone Network: -

Bottom up pathway - For feature extraction, a bottom up pathway (e.g. ResNet) is employed. As a result, regardless of the size of the input image, it computes feature maps at various scales.

Top down pathway with lateral connections- The top down pathway samples the spatially coarser feature maps from higher pyramid levels, and the lateral connections combine top-down and bottom-up layers of the same spatial size.

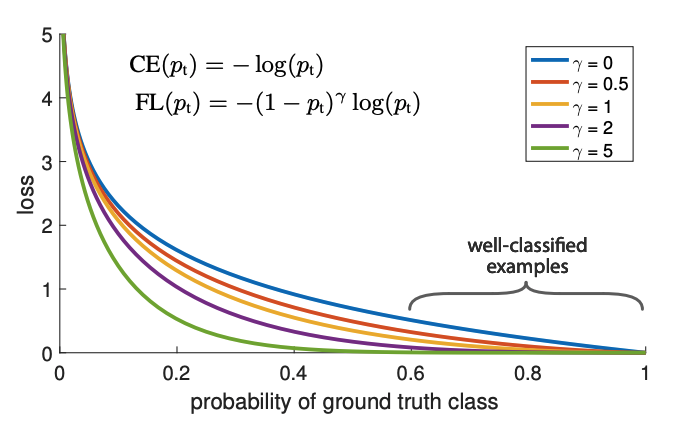
Higher level feature maps tend to have low resolution, although being semantically stronger, and are thus better suited to detecting larger things; on the other hand, grid cells from lower level feature maps have high resolution and are thus better suited to detecting smaller items. As a result, this architecture is scale-invariant and can deliver superior performance in terms of both speed and accuracy.

Classification subnetwork - For each anchor box and item class, it estimates the likelihood of an object being present at each spatial location.

Regression subnetwork - For each ground-truth object, it regresses the offset for the bounding boxes from the anchor boxes.

**Focal Loss:**

One of RetinaNet's significant contributions is the use of a focal loss function, which is an improvement over the commonly used Cross-Entropy Loss (CE) and is introduced to deal with the class imbalance problem with single-stage object detection models.



The loss function is modified to reduce the weight of easy examples, focusing training on hard negatives. A modulating factor (1-pt)γ is added to the cross entropy loss.

V. Results

| **Parameter** | **Detectron2** | **RetinaNet** | **YoloV7** |
| --- | --- | --- | --- |
| Precision | 0.64 | 0.75 | 0.895 |
| Recall | 0.67 | 0.69 | 0.451 |
| Loss | 0.44 | 0.23 | 0.051 |

The comparative study of the 3 models on the dataset shows that Detectron2 performs worse than RetinaNet and YoloV7 in terms of both precision and Recall. YoloV7 has the highest precision of the three models. RetinaNet has the highest Recall.

VI. Conclusion

From the above results we can conclude that Yolov7 is ideal for this use case as it has high precision and fast inference speed. This also makes it suitable for real-time scenarios.

VII. References

1. Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement.

2. Chien-Yao Wang, Alexey Bochkovskiy, and Hong- Yuan Mark Liao, Institute of Information Science, Academia Sinica, Taiwan. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.

3. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision (ECCV), 21-37

4. Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal loss for dense object detection. IEEE International Conference on Computer Vision (ICCV), 2980-2988

5. Wu, Y., Kirillov, A., Massa, F., Lo, W. Y., & Girshick, R. (2021). Detectron2