# RCNN extension for SAM (segment anything model)

## Introduction:

Segment anything model has shown good capabilities in segmenting images with zero shot capability. We tried to use this segmentation as ROIs (regions of interests) for object detection in aerial imagery. To achieve this, the output of SAM will be passed through some convolutions and fully connected layers for classifying each ROI and regressing bounding box for it.

SAM comes with an automatic mask generator that creates a mesh grid of points and then for each point the segments will be calculated and then the whole result will be combined and post-processed using the NMS and some few other techniques.

## Data preparation:

To make the process of training and experimenting with different models, the SAM image encoder output, and automatic segmentation output have been pre computed for all the images and stored as numpy files for each image. The feature map has the dimension of 256 x 64 x 64 for each image and the segmentation result which is a list of dictionaries including the bounding box, confidence and some other specifications of each segment.

During the training only up-down flip, and right-left flip has been used.

Mosaic data augmentation is suggested to be added to the training process.

## Matching proposals to targets:

In order to calculate the loss for the extension modules that will be added to SAM, we need to first convert the ROIs to ground truth. For this purpose the following process is being done before the loss calculation (in build\_target method of compute\_loss class):

1. Each target will be matched with all of the ROIs corresponding to the target’s image and the IOU will be calculated between them and the ROI with the highest IOU greater than a threshold (e.g. 0.5) will be assigned to the target.
2. The xywh of the ROI will be changed to the xywh of the target.
3. The class of the target will be assigned to the ROI.
4. The processed ROIs will be passed for loss calculation.

## Loss function:

The loss funcation is combination of bounding box loss, classification loss, and objectness loss. However, the objectness loss has been removed during the last experiment.

Bounding box loss can be defined as 1 – IOU of the prediction and its corresponding ROI bounding box. However we only predict the offset to the xywh so we need to compute the predicted box from the offset using same formula used in Fast-rcnn. It is worthy to mention that the bounding box loss is only being calculated for positive ROIs and not for the others.

Objectness loss is being calculated based on the IOU which was computed for the bbox loss and the bojectness predictor should predict the IOU for positive and 0 for negative proposals.

Classification loss is also only computed for positive ROIs.

## Results:

The result for the models has been very poor, as low as 0.06 for MAP. The model cannot distinguish the objects from background good enough.

A graph of a graph

Description automatically generated

Ground truth:

A collage of a city

Description automatically generated

Predictions:

A collage of a city

Description automatically generated

## Challenges:

There can be many challenges causing the poor performance of the model and the most probable ones are listed below:

1. Lack of enough data augmentation. As the input is 256 channels feature maps it is difficult and complex to do data augmentation for it.
2. Loss function might have bugs. As the number of ROIs and predictions for each image is different and the number of negative ROIs is very high the model might just learn a trivial solution of predicting very low confidence for the objectness for the majority of the ROIs.
3. ROI pooling: as we do ROI pooling to convert the ROIS to a fixed size and the high variation in the size of ROIs might cause it to become a bottleneck and the model can not distinguish between background and foreground.
4. Weak model architecture. The model might be too weak to detect objects and maybe increasing the complexity of the model will solve the problem.
5. The quality of ROIs is not perfect and even with a low threshold of 0.5 IOU we still miss 20 – 30% of the labels.
6. Scale variance of the objects. Yolo and superyolo and many other object detection models relies heavily on the predefined anchors and anchor boxes. But in our case we do not have predefined anchors except the ROI bounding boxes.

## Potential enhancements:

1. Fist thing suggested is to change the loss function exactly as the loss function of the Fast-rcnn. Though I have tried with L2 loss for bounding box, removing the objectness loss and treating it as classification loss, and ignoring some of the predictions. Still they have not worked.
2. If the problems were not resolved by changing the loss function, we might need to introduce some kinds of dynamic anchors in various scales and aspect ratio same as other models to help with identifying the objects. The difference would be that we won’t have high number of anchors like YOLO or faster-rcnn covering the whole image but only a few anchors.