Small-Object Detection in Remote Sensing Images and Video

by

Stamatios Orfanos

Submitted
in partial fulfilment of the requirements for the degree of

Master of Artificial Intelligence
at the
UNIVERSITY OF PIRAEUS
July 2024

Author... **Stamatios Orfanos**
II-MSc “Artificial Intelligence”
, 2024

Certified by.. **Ilias Maglogiannis**
Professor
Thesis Supervisor

Certified by.. **Theodoros Giannakopoylos**
Researcher
Member of Examination Committee

Certified by.. **Michael Filippakis**
Professor
Member of Examination Committee

Small-Object Detection in Remote Sensing Images and Video

By

Stamatios Orfanos

Submitted to the II-MSc “Artificial Intelligence” on XX XX, 2024, in
partial fulfilment of the
requirements for the MSc degree

Abstract

Object detection in remote sensing images has been a challenging problem for the computer vision research community because the objects in such images have very few pixels (10-20 pixels). There have been many improvements in the mean Average Precision (mAP) of the models using different techniques, but all these improvements come at a cost. The detection models are becoming bigger, which can cause a problem especially when a detection model is intended for use in a satellite or an Unmanned Aerial Vehicle, since their computation capabilities are limited. The thesis introduces a novel approach that has achieved a significant reduction in computational complexity, specifically a 32.67% decrease in Giga Floating Point Operations Per Second (GFLOPs) for the Transformer Prediction Head YOLOv5 (TPH-YOLO) model. Remarkably, on the Aerial Image Tiny Object Detection (AI-TOD) dataset, this optimization also achieves an increase of 6.3% mAP at 50% IoU threshold and 2.4% at the average mAP across IoU thresholds from 50% to 95%. The results demonstrate the effectiveness of the proposed method in balancing computational efficiency with detection performance for the utilized datasets.

Thesis Supervisor: Ilias Maglogiannis
Title: Professor

Contents

List of Figures

List of Tables

1 Introduction

Remote sensing imaging is a process used to gather information about objects or areas from a distance, typically using aircraft or satellites. This process is essential in various fields due to its ability to detect and monitor the physical characteristics of an area by measuring its reflected and emitted radiation.

Remote sensing imaging starts with data acquisition through sensors, that are mounted on platforms like satellites or aircraft, capture electromagnetic radiation and can range from simple cameras to complex radar systems. After capturing this data, it is transmitted to ground stations for processing. The processing stage often involves correcting any image distortions, enhancing details, and converting the raw data into usable formats.

Remote sensing imaging has applications across a broad spectrum of fields. Starting with environmental monitoring is one of the primary uses, enabling the observation and analysis of environmental changes like deforestation and the health of aquatic ecosystems. In agriculture, it helps monitor crop health and soil conditions, aiding in the efficient management of resources. The technique is also crucial in disaster management, where it is used to assess damage from natural disasters and plan effective responses. Urban planning benefits from remote sensing by providing data for the development and monitoring of infrastructure and urban growth. In the military and intelligence sectors, remote sensing is key for surveillance and reconnaissance, providing critical information for national security.

Provided the numerous applications of remote sensing images, the computer vision research community is continually pushing to develop object detection models that can effectively parse and interpret the vast amount of data captured by remote sensors. In remote sensing images the objects are small fraction of the pixels of the image, qualifying this process as Small Object Detection.

Even though impressive results have been achieved on large and medium objects in large-scale detection benchmarks, the performance on small or tiny objects is far from satisfactory. Compared with large and medium objects, the small objects are more difficult to detect accurately, because of four main difficulties. Firstly, small objects have low resolution and insufficient features. Secondly, the span of object-scale is large and multiple scales coexist. Thirdly, the examples of small objects are scarce and lastly the categories for small objects are imbalanced for the majority of datasets.The concept of small or tiny objects seeks to elucidate the scale of these objects or the proportion of pixels they occupy within the entire image. There are two main ways to define small objects.

The first way is the use relative size. According to the definition of Society of Photo-Optical Instrumentation Engineers [SPIE], if the object size is less than 0.12% of the original image, it is regarded as a small object. Following the same principle Krishna and Jawahar [1] showed that an object is considered small if it occupies only a tiny portion of the image, which is less than 1% of the image area. Namely, the bounding box of a small object should cover less than 1% of the original image. The second way of defining a small object by using the absolute size, where a small object has size less than 32x32 pixels defined in MS-COCO dataset or 16x16 pixels defined in USC-GRAD- STDdb [2].

There have been some improvements of the models using different techniques, but all these improvements come at a increased complexity and size of the model. This complexity can be prohibitive for applications in a satellite or an Unmanned Aerial Vehicle since their computation capabilities are limited. Driven by the need for more precise object detection models, this thesis proposes a novel methodology to reduce the computation cost of the detection model for utilization in such cases.

This thesis aims to explore the combination of two successful models from two different approaches in object detection, while maintaining a smaller model size. The evaluation process utilizes datasets that have been parts of employed in the original research papers of these models. Furthermore, this thesis extends its scope to the field of Remote Sensing Images (RSI). Both the original and modified versions of the models will be evaluated using a common RSI dataset. This will facilitate a comprehensive comparison of all model results within a consistent dataset, thereby providing valuable insights into their performance in real-world scenarios.

The remainder of the thesis is organized as follows:

Chapter 2 contains related work surrounding the scope of the thesis. It starts with an explanation of the architecture of Recurrent Convolutional Neural Networks (R-CNNs) alongside the Feature Pyramid Networks and analyzes the distinct role and functionality of each component. It follows with the explanation of the architecture of the Vision Transformers that were used as a basis for the detector of our model. It continues by explaining the difficulties of detecting small objects in remote sensing images.

Chapter 3 introduces the architecture of the suggested model and is presented, providing an extensive description of its design and functionality. It also highlights the significant publications and research that have been a major help in advancing and improving the model's design.

Chapter 4 offers an in-depth overview of the datasets used to evaluate the proposed model. It states the specific parameters used throughout the training phase to ensure a thorough knowledge of the model's learning process. The experimental findings are presented at the end of the chapter, providing a concrete indicator of the model's effectiveness.

Chapter 5 analyzes the experimental findings and discuss the implications of the differences that were observed across different models and datasets. This aims to unravel the underlying implications of these differences, thereby enhancing the understanding of the models' performance across different datasets.

Chapter 6 provides an overview of the future work which aims at investigating performance differences, enhancing the model's performance and testing the method's generalizability across various datasets and domains.

2 Related Work

This chapter highlights the foundational theories and recent advancements in the fields of remote sensing and computer vision, particularly in small object detection. By examining previous research that addresses similar challenges, this section not only underscores the technological progress achieved but also identifies the gaps that the current model aims to bridge. In the field of computer vision, Convolutional Neural Networks (CNNs) served as the initial models for image analysis, primarily focused on image classification where the entire image is labeled as a single object category. While CNNs had great performance in these tasks, their application to object detection in complex images revealed significant limitations.

This chapter begins with the foundational R-CNN model, which introduced the use of convolutional networks for robust object detection. The following sections dive into Fast R-CNN and Faster R-CNN, which iteratively refined the integration of region proposal mechanisms with deep learning, significantly enhancing detection efficiency and speed. The exploration continues with Mask R-CNN and Feature Pyramid Networks, which extended capabilities to instance segmentation and improved feature representation at multiple scales, respectively.

Further advancements are discussed through the Extended Feature Pyramid Network, which brought additional refinements in multi-scale feature integration. The latter sections of the chapter explore cutting-edge developments like the Vision Transformer and Masked-Attention Mask Transformer, which incorporate transformer architectures to push the boundaries of object detection and segmentation.

2.1 Region-based Convolution Neural Networks

The R-CNN family of models represents a fundamental shift in object detection, introducing deep learning to generate high-quality region proposals that are then classified by a convolutional neural network. This evolutionary path not only streamlined the detection pipeline but also improved the scalability and applicability of these models in real-world scenarios, where speed and accuracy are crucial. The successive refinements from R-CNN through Mask R-CNN highlight a trajectory of continuous improvement, with each iteration bringing more sophisticated integration of features and functionalities.

2.1.1 Region-based Convolution Neural Networks

The need for more sophisticated solutions that could accurately identify and locate multiple objects within images led to the development of Region-based Convolutional Networks[3] (R-CNNs). Starting with the base Region-based Convolution Neural Network This approach combines region proposal algorithms with the feature extraction capabilities of CNNs. R-CNNs begin by generating potential object-bound regions in an image, a process known as region proposal. Each region is then cropped and resized to a fixed size before being fed into a pre-trained convolutional neural network.

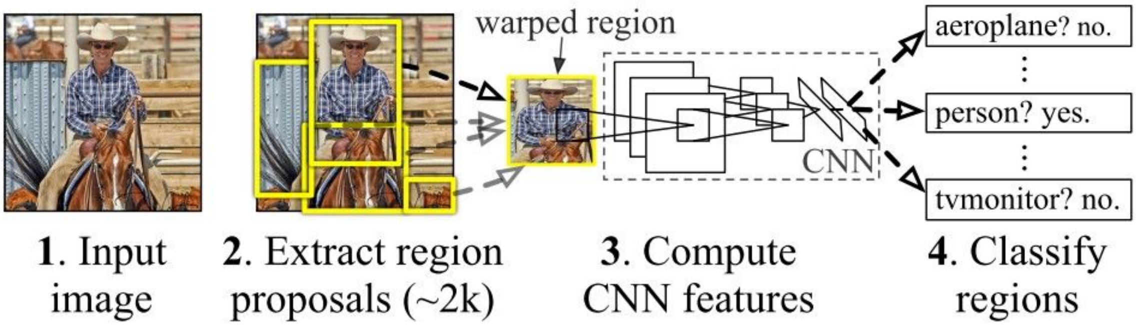

Figure 2.1: Region-based Convolution Neural Network Architecture

As presented in the 2.1 the R-CNN consists of 3 main modules. The first module generates 2,000 region proposals using the Selective Search algorithm. After being resized to a fixed pre-defined size, the second module extracts a feature vector of length 4,096 from each region proposal. The third module uses a pre-trained SVM algorithm to classify the region proposal to either the background or one of the object classes.

Some the limitations of the R-CNN model are the facts that it is a multi-stage model, where each stage is an independent component, thus, it cannot be trained end-to-end. Also the R-CNN depends on the Selective Search algorithm for generating region proposals, which takes a lot of time and cannot be customized to the detection problem. Lastly each region proposal is fed independently to the CNN for feature extraction, which makes it impossible to run R-CNN in real-time.

2.1.2 Fast Region-based Convolution Neural Networks

Fast R-CNN improved upon the original R-CNN's efficiency, where instead of cropping and resizing each region separately, the entire image is passed through the CNN to extract features. Regions of interest (ROIs) are then selected from the feature map using the proposed bounding boxes from the selective search. These ROIs are then pooled into a fixed-size feature map and passed through fully connected layers for classification and bounding box regression in the figure 2.2.

In this model a proposed a new layer called ROI Pooling that extracts equal-length feature vectors from all proposals in the same image, where compared to R-CNN, which has multiple stages, Faster R-CNN builds a network that has only a single stage.

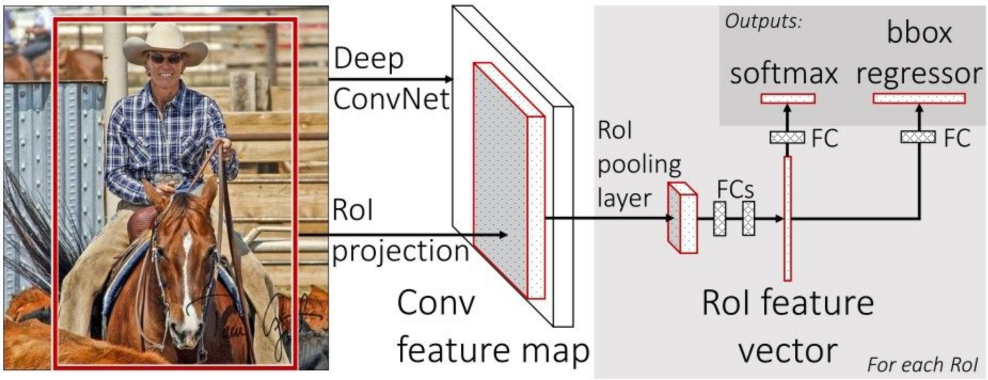

Figure 2.2: Fast Region-based Convolution Neural Network Architecture

The RoI pooling layer uses max pooling to convert the features inside any valid region of interest into a small feature map with a fixed spatial extent of $H \times W$, where H and W are layer hyper-parameters that are independent of any particular RoI. In this paper, an RoI is a rectangular window into a convolution feature map. Each RoI is defined by a four-tuple (r, c, h, w) that specifies its top-left corner (r, c) and its height and width (h, w). Also one of the great inclusions of this model was the implementation of multi-task loss:

$$L(p, u, t', v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t', v)z$$

,where the classification loss $L_{cls}(p, u)$ is defined as the negative log likelihood of the true class u, expressed as:

$$L_{cls}(p, u) = -\log p_u$$

The localization loss L_{loc} is defined over the predicted bounding box parameters $t' = (t'_x, t'_y, t'_w, t'_h)$ and the ground truth bounding box parameters $v = (v_x, v_y, v_w, v_h)$ for class u. The Iverson bracket $[u \geq 1]$ is used as an indicator function that evaluates to 1 when u is 1 or more, and 0 otherwise. This function helps in applying the localization loss only when there is a foreground class detected, effectively ignoring the background.

The overall loss $L(p, u, t', v)$ is then a combination of classification and localization losses, modulated by a parameter λ, representing the trade-off between these two tasks:

$$L(p, u, t', v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t', v)$$

2.1.3 Faster Region-based Convolution Neural Networks

While Fast R-CNN improved upon its predecessors in terms of both speed and accuracy, the Faster R-CNN[4] architecture emerged as an even more refined version. Fast R-CNN effectively addressed the inefficiencies of previous models by integrating a region of interest (RoI) pooling layer to connect convolutional feature extraction and region proposal tasks. However, it still relied on external region proposal algorithms, which remained a bottleneck in terms of computational efficiency and speed.

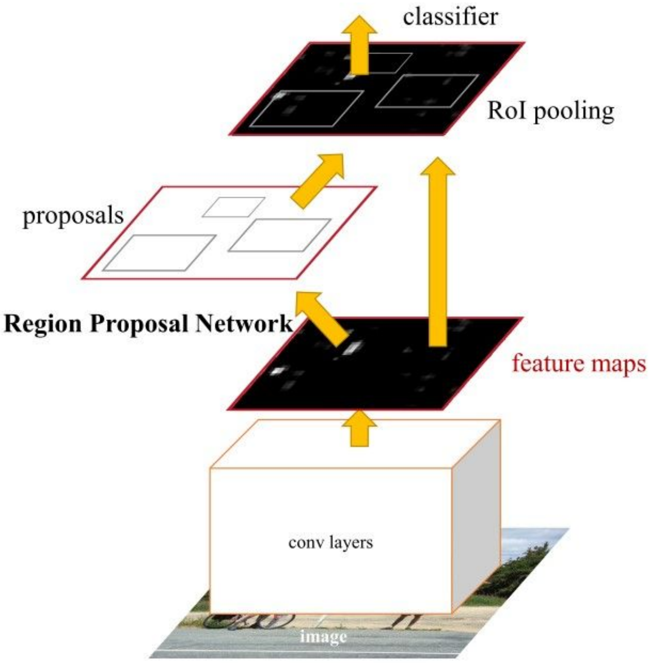

Figure 2.3: Faster Region-based Convolution Neural Network Architecture

Faster R-CNN resolved this by introducing a novel component, the Region Proposal Network (RPN)[5]. A Region Proposal Network takes an image (of any size) as input and outputs a set of rectangular object proposals, each with an objectness score. This process is modeled with a fully convolutional network, because the ultimate goal is to share its computation with a Fast R-CNN object detection network, as it is assumed that both nets share a common set of convolutional layers, as seen in the Figure 2.3.

To generate region proposals, a small network slides over the convolutional feature map output by the last shared convolutional layer. This small network takes as input an $n \times n$ spatial window of the input convolutional feature map. Each sliding window is mapped to a lower-dimensional feature. This feature is fed into two sibling fully-connected layers—a box-regression layer (reg layer) and a box-classification layer (cls layer). This mini-network is illustrated at a single position in the Figure 2.4. Also since the mini-network operates in a sliding-window fashion, the fully-connected layers are shared across all spatial locations.

This architecture is naturally implemented with an n×n convolutional layer followed by two sibling 1×1 convolutional layers.

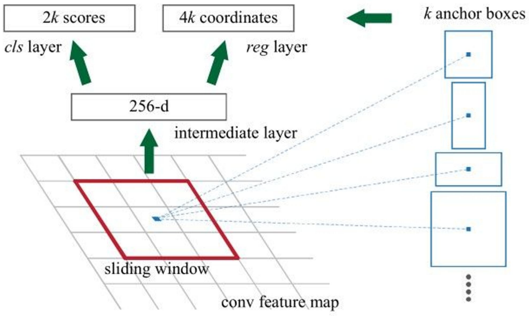

Figure 2.4: Region Proposal Network

At each sliding-window location, there is simultaneously a prediction of multiple region proposals, where the number of maximum possible proposals for each location is denoted as k. The k proposals are parameterized relative to k reference boxes, which are called anchors. An anchor is centered at the sliding window in question, and is associated with a scale and aspect ratio as seen again in Figure 2.4. By default 3 scales and 3 aspect ratios are used, yielding k = 9 anchors at each sliding position. For a convolutional feature map of a size $W \times H$, there are $W \times H \times k$ anchors in total.

2.1.4 Masked Region-based Convolution Neural Networks

Mask R-CNN builds on the ideas and successes of the Faster R-CNN model, which predicts both a class label and a bounding-box offset for each candidate object. To these, Mask R-CNN adds a third branch specifically designed to output the object mask, providing a straightforward and logical extension to the existing framework. This addition allows Mask R-CNN to capture the precise spatial layout of objects, a task that necessitates extracting significantly finer detail than what is required for classifying objects or predicting bounding boxes alone.

Mask R-CNN adopts the same two-stage procedure as Faster R-CNN, with an identical first stage the RPN. In the second stage, in parallel to predicting the class and box offset, Mask R-CNN also outputs a binary mask for each RoI. This is in contrast to most recent systems, where classification depends on mask predictions. This approach follows the spirit of Fast R-CNN that applies bounding-box classification and regression in parallel.

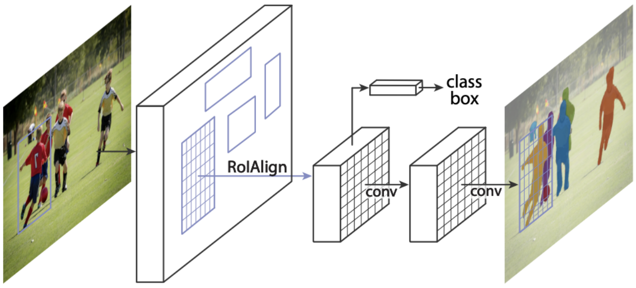

Figure 2.5: Mask Region Convolution Neural Network Architecture

A mask encodes an input object's spatial layout. Thus, unlike class labels or box offsets that are inevitably collapsed into short output vectors by fully-connected layers, extracting the spatial structure of masks can be addressed naturally by the pixel-to-pixel correspondence provided by convolutions. Specifically, an $m \times m$ mask is predicted from each RoI using an FCN. This allows each layer in the mask branch to maintain the explicit $m \times m$ object spatial layout without collapsing it into a vector representation that lacks spatial dimensions. Unlike previous methods that resort to fully-connected layers for mask prediction, this fully convolutional representation requires fewer parameters, and is more accurate as demonstrated by experiments.

This pixel-to-pixel behavior requires our RoI features, which themselves are small feature maps, to be well aligned to faithfully preserve the explicit per-pixel spatial correspondence. This motivated us to develop the following RoIAlign layer that plays a key role in mask prediction.

RoIPool is a standard operation for extracting a small feature map like 7×7 from each RoI. RoIPool first quantizes a floating-number RoI to the discrete granularity of the feature map, this quantized RoI is then subdivided into spatial bins which are themselves quantized, and finally feature values covered by each bin are aggregated usually by max pooling.

Quantization, such as that performed on a continuous coordinate x by calculating $\frac{x}{16}$, where 16 represents the feature map stride accompanied by the rounding. Similarly, this quantization process occurs when coordinates are divided into discrete bins, for example, into a $7x7$ grid. However, these quantization steps can lead to slight misalignments between the Region of Interest (RoI) and the features extracted from it, potentially affecting the accuracy of the model.

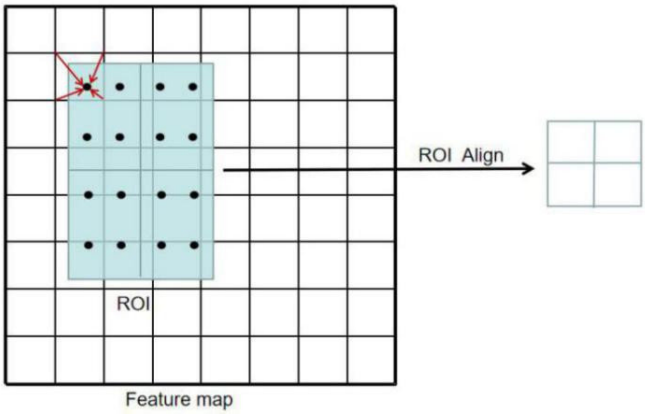

Figure 2.6: Mask Region Of Interest

While this may not impact classification, which is robust to small translations, it has a large negative effect on predicting pixel-accurate masks. To address this, a RoIAlign layer is proposed that removes the harsh quantization of RoIPool, properly aligning the extracted features with the input. A bilinear interpolation is used to compute the exact values of the input features at four regularly sampled locations in each RoI bin, and aggregate the result.

Furthermore once again this model utilizes a multi-task loss in order to take into consideration all the outputs of the model.

$$L = L_{cls} + L_{box} + L_{mask}$$

The mask branch has a $K \times m^2$-dimensional output for each RoI, which encodes K binary masks of resolution m × m, one for each of the K classes. To this a per-pixel sigmoid is applied, and define Lmask as the average binary cross-entropy loss. For an RoI associated with ground-truth class k, L_{mask} is only defined on the k-th mask. This definition of L_{mask} allows the network to generate masks for every class without competition among classes, since on the dedicated classification branch to predict the class label used to select the output mask. This decouples mask and class prediction. This is different from common practice when applying FCNs to semantic segmentation, which typically uses a per-pixel softmax and a multinomial cross-entropy loss. In that case, masks across classes compete; in our case, with a per-pixel sigmoid and a binary loss, they do not.

2.2 Feature Pyramid Networks

2.2.1 Feature Pyramid Network

2.2.2 Extended Feature Pyramid Network

2.3 Vision Transformers

2.3.1 Vision Transformers

2.3.2 Masked-Attention Mask Vision Transformer

3 Methodology

3.1 Architecture

In this chapter, we demonstrate the overall design that is based on the two-stage model PANet, the framework is illustrated in Figure 18. In our design, we add a Split Stage between the backbone and the “top-down” path of the neck. The purpose of this stage is to split the feature maps of the equivalent levels of the backbone. This is achieved with a 1x1 Convolution layer which reduces the channel dimension of each level to 256-d, which means that we will have 2 × 128-d feature maps for each level. ...

4 Experiments

4.1 Datasets

The datasets that were used for the experiment are the Microsoft Common Object in COntext (MS COCO), Aerial Images Tiny Object Detection (AI-TOD), and VisDrone. The MS COCO and VisDrone datasets were used to evaluate the performance of the proposed architecture by comparing its results with the results of the original models (PANet and TPH-YOLO). The dataset AI-TOD was chosen because it consists of only remote sensing images and the objects in the images only have very few pixels, something that makes it a challenging dataset. ...

5 Discussion

The implementation and evaluation of the proposed method on the PANet and TPH-YOLOv5 models produced two different results on the model performances. It was extended that while the implementation of the proposed method on TPH-YOLOv5 significantly reduced the computational cost of the model with little loss in performance or even improvement in performance on a dataset with remote sensing images, the implementation of the method on PANet did not have as much improvement in computational cost and the loss in performance was higher. ...

6 Conclusion

In this thesis, we introduced a novel network-level gradient path design for object detection models, incorporating both ‘top-down’ and ‘bottom-up’ pathways in the network’s ‘neck’. This method is versatile, with applicability to both single-stage and multi-stage models. Upon evaluating the performance of the original models and their modified counterparts using our proposed method, we arrived at the following conclusions: ...

A Appendix A

Details of the modified TPH-YOLOv5 architecture and additional experimental results.