SCHOOL OF ENGINEERING AND DESIGN

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis

Automatic Detection of Rooftop Solar Photovoltaics from Satellite Imagery using Deep Learning

Mohammad Azeem Khan

Matriculation Number: 03743182

SCHOOL OF ENGINEERING AND DESIGN

CHAIR OF LAND MANAGEMENT

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis

Automatic Detection of Rooftop Solar Photovoltaics from Satellite Imagery using Deep Learning

Land Management & Geospatial Science (LMGS)

Author:	Mohammad Azeem Khan
Supervisor:	Prof. Walter Timo De Vries
Co-Supervisor:	Mr. Tobias Bendzko
Advisor:	Mr. Markus Biberacher
Submission Date:	21 June 2023

Acknowledgments

I want to express my sincere gratitude to the team at Research Studio Austria, particularly Mr. Markus Biberacher, for their invaluable support and guidance throughout the duration of my thesis. Their expertise, resources, and encouragement greatly contributed to the successful completion of this research. I am also thankful to my supervisor from the university, Mr. Tobias Bendzko, for his valuable insights and constructive feedback that helped shape the direction of my study. Additionally, I would like to extend my appreciation to my friends Mr. Shahbaz Khan (M.Sc), Muhammad Arhab (M.Sc) for their assistance and support during the course of my thesis. Their willingness to share knowledge and provide valuable input significantly enriched the quality of my work. Furthermore, I would like to thank all the participants who willingly shared their insights and experiences, as their contributions were essential for the success of this study. Finally, I am grateful to my family and friends for their unwavering support and understanding throughout this academic journey. Their encouragement and belief in my abilities were constant sources of motivation. I am truly privileged to have had the opportunity to work with such remarkable individuals and organizations who played a significant role in the realization of this thesis. Their contributions have been invaluable, and I am sincerely thankful for their support.

Abstract

In the past decade, there has been a substantial increase in the installation of solar panels in residential and commercial settings. This is primarily driven by cost reductions and the growing recognition of the urgency to address climate change. However, effectively monitoring these installations presents a challenge due to their widespread and distributed nature. To address this challenge, this thesis project focuses on automating the identification of rooftop solar photovoltaic (PV) systems using deep learning techniques and satellite imagery. The study specifically targets the Salzburg region of Austria, aiming to support the region's renewable energy goals by leveraging the advantages of satellite imagery and deep learning, such as improved efficiency, accuracy, and safety compared to manual inspections. The objectives of the study include the collection of a dataset consisting of satellite images and the annotation of these images with ground-truth labels. Subsequently, a deep learning model is developed and trained to automatically detect and segment PV systems. The research also addresses various questions, such as the benefits of utilizing satellite imagery in deep learning, the selection of an appropriate deep learning model for object detection, the potential applications of the obtained information, and the technical challenges involved in implementing Mask R-CNN for PV detection. Through a comparative evaluation of multiple models from the literature, Mask R-CNN emerges as the most effective approach for object detection and instance segmentation. The model is trained on a custom dataset and fine-tuned using different hyperparameters. The experimental results demonstrate the model's effectiveness, achieving a bounding box mAP score of 0.803 for accurate detection and localization of PV systems and the segmentation mAP score of 0.781 indicates the model's ability to accurately segment individual instances within the detected bounding boxes. In conclusion, this research contributes to the development of an automated method for detecting and mapping rooftop solar PV systems using deep learning and satellite imagery. The findings have significant implications for renewable energy planning, policy-making, and monitoring in the Salzburg region, while also providing a foundation for similar research in other geographical areas.

Keywords: Mask R-CNN, rooftop solar PVs, object detection, satellite imagery, Deep Learning.

Contents

Chapter 1

Introduction

The goal of this thesis project was to explore the potential of deep learning techniques to automate the identification of rooftop solar photovoltaic (PVs) systems using satellite images. The project was conducted at the iSpace department of Research Studio Austria Forschungsgesellschaft, located in Salzburg [1]. This chapter serves as an introduction to the problem addressed in the thesis, as well as the project's scope and limitations.

1.1 Background

Coal, oil, and natural gas are still major contributers in electricity generation worldwide. However, the carbon dioxide emissions from these fuels are significant contributors to climate change. Coal remained the world's primary source of energy in 2019, which accounts for 37% of all electricity produced, according to the International Energy Agency's Global Energy (IEA Review 2021) [2]. Despite the significant progress made by renewable energy sources such as hydropower, wind, and solar power, they still accounted for only 26% of the world's electricity production in 2019 [3]. To mitigate the emissions of carbon dioxide and combat climate change, it is crucial to enhance the contribution of renewable energy sources. In addition, renewable energy can be an effective means to solve urban energy requirements. Currently, urban areas are home to more than half (55%) of the world's population. This trend of urbanization is expected to continue, and projections indicate that the urban population will reach 68% by the year 2050 [4]. Energy demand is increasing due to rapid urbanization, as more people move into cities and engage in energy-intensive activities such as heating and cooling buildings, and using various appliances and devices. The existing energy infrastructure and supplies are under intense pressure as a result of this growth in energy demand.

Solar energy is typically the first type of renewable energy that comes to mind when thinking about how to increase energy demand while decreasing our reliance on non-renewable sources. Solar panels, being a renewable and clean energy source, which produces electricity from sunlight, can significantly reduce carbon dioxide emissions and also lessen the demand for new power plants, which can be expensive and have an adverse effect on the environment.

Rooftop solar photovoltaics (PVs) are actually a smaller component of a larger electricity generation portfolio using solar photovoltaics that offers a quick installation time and low levelized cost [5]. They grant a decentralized system that can be installed by both small-scale industrial and commercial complexes and individual homeowners, adding a local source of energy that can supplement the overall energy supply [6, 7]. By empowering citizens and communities to become "prosumers" who create and use their own electricity as needed, rather than just relying on centralized grid infrastructure, they can aid in tackling the problem of access to energy for any usage domain.

There is an increasing need for managing rooftop solar photovoltaic array power generation as their popularity continues to rise. This is especially crucial in urban areas where there is rapid deployment of decentralized power generation through solar photovoltaic. In order to manage the electric grid and make sure that the supply of electricity keeps up with demand, smart cities, third-party companies, government agencies, and utilities, are all involved [8]. One of the challenges faced in this process is the unpredictability of power generation from distributed rooftop solar PV arrays. Factors like weather conditions and other variables can cause fluctuations in the generation of solar power. As the use of rooftop solar PV arrays increases, it becomes more difficult for utilities to accurately predict the net load, resulting in potential financial losses. Although homeowners benefit from using solar energy during the day, utilities still need to maintain the sufficient energy-generating capacity to meet their electricity demands when solar PV arrays are not producing electricity [8]. As a result, there is mounting demand for these stakeholders to devise strategies for controlling this fluctuation and ensuring the resilience of the electric grid. However, obtaining such detailed information can be challenging as most of the available databases are collected from utility interconnection fillings and surveys which are limited in their spatial resolution and completeness [9]. Furthermore, collecting information for a large number of household PV arrays can be both time-consuming and expensive, which can lead to outdated information in some cases.

This study explores an automated method for gathering data on the location and size of existing installations on solar photovoltaic (PV) systems, which has various advantages for the development of the solar energy industry. Automated systems are capable of identifying solar panels more rapidly and accurately than manual inspections, resulting in increased efficiency, accuracy, and safety. They can also aid in cost-cutting and encourage the use of renewable energy sources.

The proposed approach offers significant advantages compared to existing methods of collecting PV data. Firstly, it allows for information collection of PVs at a remarkably high level of geospatial resolution. This high-resolution data can greatly assist in energy infrastructure decision-making and planning at various levels, including local, regional, and state levels. By integrating detailed spatial maps of PV data, such as the existing PV capacity per km^2, into geographic information systems (GIS) or Energy-GIS systems, it becomes possible to address this challenge effectively [10]. This integration enhances the understanding of PV deployment patterns and enables more informed and targeted energy planning and policy implementation [11, 12, 13]. Overall, the proposed approach empowers stakeholders with valuable insights and data for the effective management of solar energy resources. These systems are being used more frequently to assess and plan energy infrastructure.

The algorithm can be applied relatively cheaply, i.e., by executing a computer program, making it economical to employ frequently as new images are available. This is especially useful for tracking the expansion of PV arrays in urban areas, where it is preferred that the algorithm run on a frequent basis, such as once a month.

1.2 Motivation

The ongoing increase in installed PV capacity indicates that more people are realizing the potential of solar energy to contribute to a sustainable and low-carbon future [14]. The long-term objective of creating a renewable energy system that lowers greenhouse gas emissions and guarantees a clean and sustainable power and heat supply globally depends on meeting the expanding demand for solar energy. As the demand continues to increase, it becomes crucial to closely monitor PV installations. In this study, I have chosen Salzburg, Austria as the focus area to monitor photovoltaic (PV) installations as Austria has ambitious goals to increase its renewable energy use and reduce carbon emissions by achieving 45%-50% of the total energy consumption from renewable sources and 100% of the total electricity consumption from renewables by 2030, with a target solar PV capacity of 9.7 GW [15]. In addition, Salzburg also has set targets to generate 100% electricity from renewable sources, increase the share of renewable energy up to 65%, reduce per capita energy consumption by 30% from 2010 to 2050, and aims to increase the proportion of locally produced renewable energy to 32.3% by 2050, compared to 8.8% in 2010 [16]. These targets highlight Salzburg's commitment to transitioning towards cleaner and more sustainable energy sources. So, monitoring and evaluating the progress of rooftop solar PV installations is essential to achieving these targets, but it poses a challenge due to the country's vast geographic area and a large number of installations. Deep learning techniques combined with high-resolution satellite imagery, can offer an efficient and accurate solution. This thesis explores the potential of deep learning techniques to extract rooftop solar PVs from satellite imagery in the Salzburg region and contribute to the country's efforts to increase its renewable energy capacity and reduce its carbon footprint.

1.3 Aim & Objectives

The aim of this thesis is to develop and evaluate an automated method utilizing a deep learning algorithm and satellite imagery to detect rooftop solar photovoltaic (PV) systems, along with their size and location, in the Salzburg region of Austria. The specific objectives of this study are as follows:

1. To collect and preprocess a high-quality dataset of satellite images covering the Salzburg region.

2. To annotate the dataset with ground-truth labels of rooftop solar PV systems.

3. To design, train, and evaluate a deep learning model for automatic detection and segmentation of rooftop solar PV systems that is scalable to

identify PV systems across different countries and can process different resolutions of satellite images.

By achieving these objectives, this study aims to contribute to the development of an efficient and accurate method for detecting and mapping rooftop solar PV systems, providing valuable insights for renewable energy planning and policy-making in Salzburg, Austria.

1.4 Research questions

The purpose of this work is to explore the feasibility and effectiveness of using satellite images and deep learning techniques, to precisely map and monitor rooftop solar photovoltaic (PV) systems in Salzburg, Austria. In order to direct the investigation, I have created a series of research questions.

1. What are the advantages of using satellite imagery in deep learning for detecting rooftop solar PVs?
2. Which deep learning model demonstrates the highest performance and suitability for object detection in satellite imagery?
3. What are the potential applications and beneficiaries of the information obtained from this study regarding distributed solar panels??
4. What are the key technical and operational challenges in implementing Mask R-CNN for the automatic detection of rooftop solar PVs?

1.5 Thesis outline

This section provides a brief overview of each chapter in the study, highlighting the key topics and objectives covered in each chapter.

- Chapter 1: Introduction: Provides an overview of the background, motivation, objectives, and research questions of the study, emphasizing the significance of efficient data collection for solar photovoltaic arrays.
- Chapter 2: Related Works: Presents a comprehensive review of object detection and instance segmentation methods, including traditional and

deep learning-based approaches. Evolution of algorithm from CNN to Mask-RCNN.

- Chapter 3: Methodology: Describes the study area, conducts a literature review, performs a comparative analysis of deep learning algorithms, and presents a case study on instance segmentation of rooftop solar PV installations using Mask R-CNN.
- Chapter 4: Advantages of Satellite Imagery in Deep Learning: Explores the advantages of using satellite imagery in deep learning applications, particularly for object detection and instance segmentation.
- Chapter 5: Best deep learning model: Comparative analysis and insights: Compares and evaluates state-of-the-art deep learning models for object detection and instance segmentation in satellite imagery.
- Chapter 6: Experimental Setup: Describes the dataset, hardware, software, and performance metrics used in the experiments.
- Chapter 7: Experiment and Results: Presents the experimental design, training procedure, quantitative and qualitative results, and discusses the applications and implications of the findings.
- Chapter 8: Conclusion and Future Work: Summarizes the main findings, contributions, and limitations of the study, and suggests potential areas for future research and improvement in the field.

Chapter 2

Related Works

This chapter gives an overview of the literature in the area while concentrating on recent developments and methodologies used in both conventional and deep learning approaches for object detection. The first section of the chapter covers object detection techniques, which are crucial for precisely identifying and localizing items in a picture. The traditional methods, which include the Histogram of Oriented Gradients (HOG) and the Scale-Invariant Feature Transform (SIFT) algorithm, among others, are taken into account. Although these methods have been employed extensively in the past, their accuracy and effectiveness may be constrained. The chapter then delves into deep learning-based approaches, such as Convolutional neural networks (CNNs) and Region-based convolutional neural network (R-CNN) which have shown remarkable performance in object detection tasks. The chapter also examines the concept of instance segmentation, which goes beyond object recognition by classifying objects at the pixel level in addition to detecting them. This chapter lays the groundwork for the other chapters of this study by reviewing the existing literature on object-detecting methods, including conventional and deep learning approaches as well as instance segmentation techniques.

2.1 Object detection

Object detection is a crucial computer vision task that involves identifying and locating objects of a specific class, such as humans, animals, or cars, within digital images. The objective of the task is to accurately determine the location and boundaries of objects within an image and assign them to respective categories [17]. Once these regions are identified, algorithms are used to extract features from the image data and classify the objects within them. It has significant uses in commercial sectors and scientific research, including text detection, face detection, pedestrian detection, vehicle detection, and medical picture detection [18].

Figure 6.2 depicts the evolution of object detection over the years. Tradi-

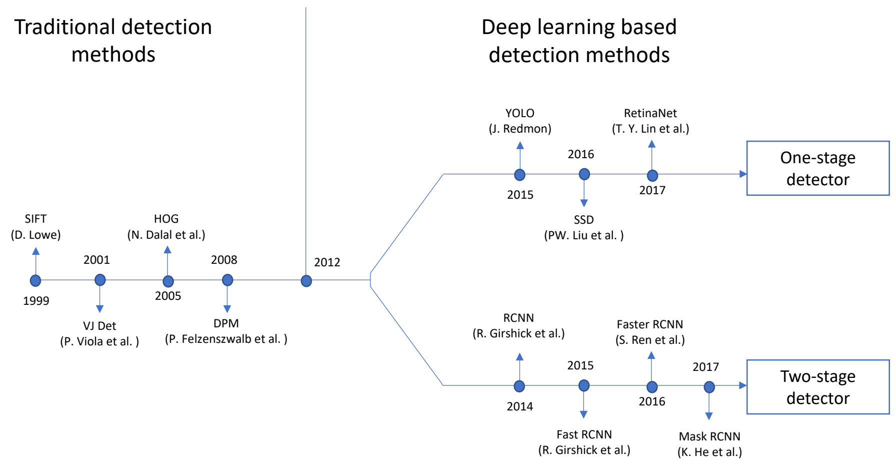

Traditional detection methods
Deep learning based detection methods
SIFT (D. Lowe)
1999
2001
VJ Det (P. Viola et al.)
HOG (N. Dalal et al.)
2005
2008
DPM (P. Felzenszwalb et al.)
2012
YOLO (J. Redmon)
2015
2016
SSD (PW. Liu et al.)
RetinaNet (T. Y. Lin et al.)
2017
One-stage detector
RCNN (R. Girshick et al.)
2014
2015
Fast RCNN (R. Girshick et al.)
Faster RCNN (S. Ren et al.)
2016
2017
Mask RCNN (K. He et al.)
Two-stage detector

Figure 2.1: The milestones of object detection evolution [19].

tional machine learning techniques were employed for object detection prior to the development of convolutional neural networks (CNNs), however, the accuracy was low, and a complex detection process. In 2012, a significant shift occurred in the field of object detection methods. This pivotal year marked the transition from conventional machine learning techniques to the emergence of deep learning methods. The introduction and subsequent success of deep learning techniques gradually replaced the traditional approaches used in object detection [20].

2.1.1 Traditional object detectors

The initial approach used in the early phases of creating object detection algorithms was to rely on handcrafted features. The main cause of this was the absence of feasible alternatives for image representation that could successfully represent the complex features of objects. Therefore, to increase the speed of image processing, the developers had to rely on sophisticated feature representations and a variety of methodologies.

In 1999, a new method was introduced by David Lowe to detect objects in images using the Scale-Invariant Feature Transform (SIFT) algorithm. SIFT is a computer vision algorithm that can identify and extract small, distinct features in an image, even if they are at different scales or orientations [21]. However, SIFT can be very computationally demanding, implying that processing large amounts of data can be time-consuming and resource-intensive., high-resolution images. This makes it difficult to use SIFT for real-time applications if only one computer processor is being used.

In 2001, two researchers named M. Jones and P. Viola developed a face detection technique that could detect human faces in real-time without any limitations based on skin color [22]. Their detector was much faster than other algorithms of the time and could achieve similar accuracy. The Viola-Jones (VJ) detector worked by using a sliding window approach, where it scanned through all possible locations and scales in an image to find a face. Although this method seems simple, it required a lot of computing power at the time. To improve the speed of their algorithm, Viola and Jones used three important techniques which are “integral image”, “feature selection”, and “detection cascades” [23]. Overall, the Viola-Jones face detection algorithm was a significant advancement in the field of computer vision, and its techniques have been widely used in subsequent face detection systems.

The Histogram of Oriented Gradients (HOG) feature detector was first proposed by N. Dalal and B. Triggs in 2005 as a way to improve upon existing feature descriptors such as the scale-invariant feature transform and shape contexts [24, 21, 25]. The HOG descriptor was designed to balance feature invariance, including translation, scale, and illumination, with nonlinearity. HOG was primarily inspired by the issue of pedestrian detection, even though it may be used to identify a number of object classes. To recognize, the HOG detector repeatedly rescales the input image to account for objects of various sizes while maintaining the same detection window size. For many years, a number of object detectors and a wide range of computer vision applications have used the HOG detector as a crucial building block [26, 27, 28].

In 2008, P. Felzenszwalb et al. introduced the Deformable Part-based Model (DPM), which is a framework for object detection using machine learning [26]. The DPM approach involves sliding a fixed-sized window across an image and assessing whether an object is present within that window. For instance, in order to identify a bicycle, the object is first broken down into its component parts, such as the handlebar, body, and wheel, and each part is identified separately. The model is trained on sets of positive and negative examples, where the former includes images containing the object of interest, and the latter consists of images that do not contain the object.

The DPM model has been successfully used to a variety of tasks, including vehicle, face, and pedestrian recognition. To obtain high levels of accuracy, it might be computationally demanding and require a lot of training data.

2.1.2 Deep learning-based object detectors

Deep learning-based object detectors refer to computer vision models that leverage convolutional neural networks (CNNs) to detect and classify objects within digital images or videos. CNNs are highly effective in learning feature representations that capture the relevant information within images, making them a popular choice for image recognition tasks.

Object detection can be broadly categorized into two groups: one-stage detection and two-stage detection. One-stage detection methods, such as YOLO , SSD, and RetinaNet, directly predict the bounding boxes and class probabilities of objects in a single pass [29, 30, 31]. Two-stage detection methods, such as R-CNN, Fast R-CNN, Faster R-CNN, and Mask R-CNN, first generate region proposals using a separate network and then refine them to produce accurate object detections [32, 33, 34, 35].

Both one-stage and two-stage detection techniques have advantages and disadvantages. One-stage detectors are frequently quicker and easier to use, which makes them better suited for real-time applications. They might, however, trade up certain accuracy at the expense of speed. On the other hand, two-stage detectors are typically more accurate but can be slower and more complicated. The particular application and its needs ultimately determine the object detection method to be used. The choice of object detection method ultimately depends on the specific application and its requirements.

Two-stage object detection architectures

RCNN

In the field of object detection, Girshick and others introduced the R-CNN (Region-based Convolutional Neural Network) detection method in 2013 [32]. Before R-CNN was created, the majority of object detection algorithms were based on sliding window approaches, which were computationally expensive and usually inaccurate in identifying objects of different sizes and aspect ratios.

R-CNN, on the other hand, proposed a region-based method that first creates a set of region proposals, or areas in the image that are likely to contain an object, and then uses a convolutional neural network (CNN) to each region proposal to classify and localize the objects in the region. The accuracy and effectiveness of object detection were greatly enhanced by the use of region suggestions and a CNN, and R-CNN attained state-of-the-art performance on numerous benchmark datasets at the time of its introduction.

Overall, the development of the R-CNN algorithm, which performed 30% better than traditional methods, was a significant milestone in the field of object detection and opened the door for future developments [36, 37]. Although R-CNN was a major breakthrough in object detection, it also had some significant drawbacks. Some of the drawbacks of R-CNN are:

1. High computational cost: R-CNN is computationally expensive, as it requires running a CNN on each region proposal. This makes it unsuitable for real-time applications and limits its scalability.

2. Inefficient region proposal generation: R-CNN uses a selective search algorithm to generate region proposals, which is slow and not very accurate. This can lead to missed detections or false positives.

3. Fixed region size: R-CNN uses fixed-sized regions for classification and localization, which may not be optimal for objects of different sizes and aspect ratios.

In order to address the limitations of R-CNN mentioned above, a new algorithm called SPPNet (Spatial Pyramid Pooling Network) was proposed in the same year (2014) [38].

Fast RCNN

Fast R-CNN is an object detection algorithm proposed by Ross Girshick in 2015 as an improvement over the original R-CNN algorithm [33]. Fast R-CNN introduced a major breakthrough by utilizing a single neural network for region proposal and object detection, leading to a significant improvement in the algorithm's speed and efficiency. Additionally, Fast R-CNN uses RoI pooling to extract a fixed-sized feature map from each region proposal, and max pooling is used to reshape them into a fixed size. These regions are then fed into a set of fully connected layers for classification and bounding box regression.

Fast R-CNN improves on earlier work in a number of ways, including training and testing speed and detection accuracy. For instance, it achieves a higher mean average precision (mAP) on PASCAL VOC 2012 and trains the very deep VGG16 network nine times faster than R-CNN while being 213 times faster at test time. Fast R-CNN trains VGG16 three times faster, performs tests ten times faster, and is more accurate than SPPnet [39].

Overall, Fast R-CNN represents a significant improvement over previous object detection algorithms, and it achieved state-of-the-art performance on several benchmark datasets.

Faster RCNN

In 2015, S. Ren et al. proposed the Faster RCNN detector, an extension of the Fast R-CNN algorithm, which aims to improve the speed and accuracy of object detection even further [34]. The main innovation in Faster R-CNN is the Region Proposal Network (RPN), which replaces the selective search method used in earlier object detection algorithms. The RPN is a fully convolutional neural network that generates region suggestions directly from the CNN's feature map, significantly reducing the computational work required to do so.

Faster R-CNN extracts fixed-sized feature maps from each area proposal using a set of fully connected layers for classification, bounding box regression, and RoI pooling. On the other hand, faster R-CNN is more accurate and effective than its predecessors because of the RPN. In terms of accuracy and speed, faster R-CNN performed significantly better than earlier state-of-the-art object detection algorithms.

By utilizing a set of fully connected layers, faster R-CNN extracts fixed-sized feature maps from each area proposal for classification, bounding box regression, and ROI pooling. It is more precise than its predecessors because it utilizes the RPN.

Due to its speed and accuracy, Faster R-CNN has emerged as a preferred option for object detection in real-world applications and represents an overall considerable advance over earlier object detection algorithms.

The development of the aforementioned algorithms increased the efficiency and precision of object detection, but reliable extraction of the location and boundary of the object remained a challenge because of the limitations of RoI pooling and the absence of pixel-level segmentation.

Mask RCNN

Mask R-CNN introduced by He et al. in 2017 extends Faster R-CNN by adding a mask branch that runs in parallel with the existing branch for bounding box recognition [35]. This allows Mask R-CNN to simultaneously predict both the object mask and bounding box for each RoI proposal. Despite the additional branch, Mask R-CNN runs at 5 frames per second, adds only a little overhead to Faster R-CNN, and is easily trainable. Additionally, Mask R-CNN is simple to apply to various tasks, such as allowing for the estimation of human poses within the same framework.

Mask R-CNN has exhibited exceptional performance on several benchmark datasets, including the COCO task suite, surpassing previous state-of-the-art results. In fact, without any additional improvements, Mask R-CNN has surpassed all currently available single-model entries on every COCO suite test, including instance segmentation, bounding-box object detection, and person keypoint detection, without any additional enhancements [35].

Single stage object detection architectures

YOLO

R. Joseph et al. first proposed YOLO in 2015 [29]. It was the first one-stage detector in the deep learning era. By simultaneously predicting object classes and bounding boxes in the same neural network, the You Only Look Once (YOLO) object detection approach was developed in order to simplify the detection process and eliminate the requirement for a separate region proposal phase. YOLO is considered a strong, fast, and simple algorithm for real-time object detection, achieving frame rates of up to 45 frames per second on a GPU and competitive accuracy on standard benchmarks such as the PASCAL VOC and COCO datasets. The YOLO network operates on the entire image and

divides it into regions to predict bounding boxes and class probabilities for each region. The original version of YOLO achieved a mean average precision (mAP) of 52.7% at 155 frames per second on the VOC-2007 dataset, while an improved version achieved an mAP of 63.4% at 45 frames per second [40]. However, YOLO has some drawbacks, such as difficulty in detecting small objects and lower localization accuracy compared to two-stage detectors.
To address the issue of detecting small objects in images, subsequent versions of YOLO and the SSD algorithm have focused on improving this aspect of object detection [41, 42, 43, 30]. More recently, the YOLOv4 team proposed YOLOv7, which introduces optimized structures such as dynamic label assignment and model structure reparameterization, resulting in improved speed and accuracy compared to most existing object detectors [44]. YOLOv7 achieves a range of 5 FPS to 160 FPS for real-time object detection tasks.

SSD

The single-shot multi-box detector (SSD), developed by W. Liu et al, was designed for real-time object detection using a single shot to detect multiple objects in an image [30]. To increase the real-time speed detection accuracy of Fast RCNN, it does away with the requirement for a region proposal network (RPN). Multi-reference and multi-resolution detection techniques were added to SSD in order to increase its accuracy in detecting small objects. However, the accuracy of SSD increases with the number of default boundary boxes, which can reduce its speed. It has been demonstrated that the SSD300 model can operate at a frame rate of 59 frames per second (FPS) and still achieve a mean Average Precision (mAP) of 74.3%. The SSD500 model, on the other hand, has a slower frame rate of 22 FPS but a higher mAP of 76.9%. According to these findings, SSD500 performs better than both Faster R-CNN, which achieves a lower mAP of 73.2% at a slower frame rate of 7 FPS, and YOLOv1, which achieves a lower mAP of 63.4% but at a faster frame rate of 45 FPS [45]. The accuracy and speed trade-offs between various object detection models are revealed by these performance metrics, it is important to note.

One key distinction between SSD (Single Shot MultiBox Detector) and previous object detectors is their approach to handling objects of different scales. Unlike previous detectors that only perform object detection on the top layers of the network, SSD detects objects at various scales by leveraging multiple layers throughout the network [46]. SSD, despite its advantages in localization accuracy compared to RCNN for similar categories, tends to exhibit higher classification errors [47].

RetinaNet

RetinaNet, introduced by T.-Y. Lin et al. in 2017, is a one-stage object detector designed to address the accuracy issues faced by previous one-stage detectors when compared to two-stage detectors [31]. The class imbalance between foreground and background objects during training was the cause of this issue, as the RetinaNet developers found out. They solved this problem by developing a new loss function called "focal loss," which modifies the traditional cross-entropy loss to emphasize difficult, misclassified examples more during training. This enabled RetinaNet to achieve comparable accuracy to two-stage detectors, with a COCO mAP@.5 of 59.1%, while maintaining high detection speed [48].

2.2 Instance segmentation

Instance segmentation is a difficult computer vision task that involves detecting and localizing objects in an image and predicting a mask for each object instance. This task requires advanced deep learning models due to its complexity and the large amount of data processing involved. The primary goal of instance segmentation is to differentiate between individual objects in an image and assign a unique label to each instance, while also segmenting each object at the pixel level. This is different from semantic segmentation, which assigns a label to each pixel without distinguishing between instances [49].

Overall, instance segmentation is an essential tool in computer vision with numerous real-world applications, including object detection, robotics, and autonomous vehicles. Due to the complexity of object detection and the amount of computational power needed to carry it out correctly, it is still a difficult task.

Figure 2.2 presents an annotated image sample from the COCO dataset, which demonstrates the distinction between different types of annotations, including image-level annotations, object-level annotations, and segmentations at the class/semantic- or instance-level. The image provides a visual represen-

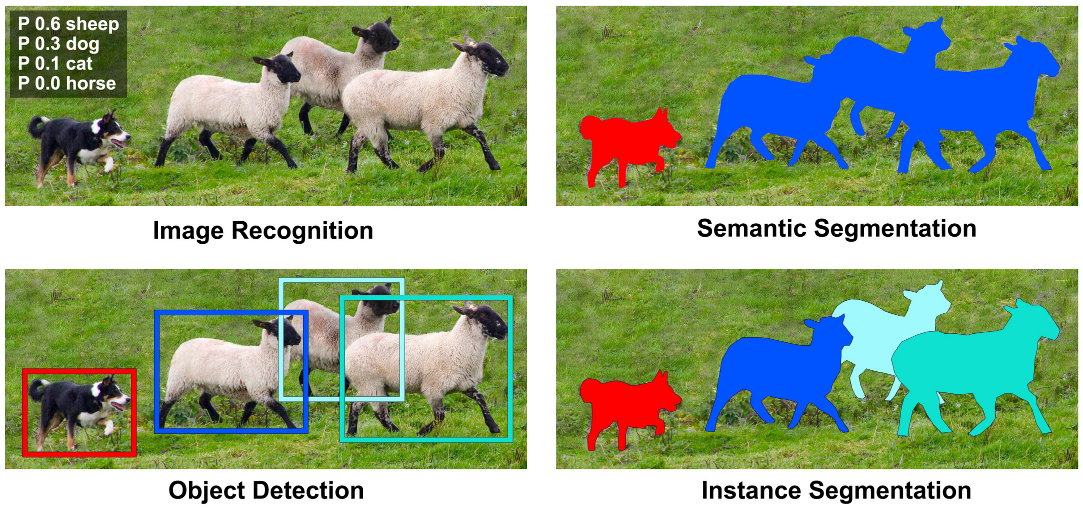


Figure 2.2: This image demonstrates the differences between image-level annotations, object-level annotations, and segmentations at the semantic- or instance-level [50].

tation of how these types of annotations differ and how they can be utilized in computer vision tasks such as object detection and instance segmentation.

Instance segmentation approaches can be broadly categorized into two main types: segmentation-based methods and detection-based methods. Segmentation-based methods involve predicting pixel-level categories and then combining the same class of pixels to generate the final instance segmentation results. On the other hand, detection-based methods detect object instances and then generate pixel-level segmentation for each instance [51]. Bai et al. employed a method for instance segmentation in which they predicted energy values at the pixel level and then used watershed algorithms to group the pixels together [52]. Kirillov et al. proposed a method for improving the accuracy of instance segmentation models by incorporating boundary detection information into the clustering procedure [53]. They achieved this by adding an additional step to the clustering process, which considers the boundary characteristics of the segmented regions, resulting in more precise and accurate instance segmentation.

Detection-based methods for instance segmentation can be traced back to the DeepMask approach proposed by Pinheiro et al. [54]. DeepMask generates instance masks by utilizing sliding windows and predicting masks using detectors such as R-FCN [55]. The performance of DeepMask in instance segmentation is constrained by its redundant feature representation and inadequate local coherency, preventing it from attaining superior results. Dai et al. proposed a method for instance segmentation using instance-sensitive fully convolutional networks (FCNs) [56]. Their approach involves generating position-sensitive maps and assembling them to obtain the final masks, resulting in improved accuracy and efficiency in instance segmentation. Mask R-CNN is a simple and effective method for instance segmentation that generates a binary mask for each class independently [35]. This approach involves extending the Faster R-CNN object detection framework with an additional mask branch, which predicts a binary mask for each detected object instance.

2.3 Evolution of object detection architectures: From CNN to Mask R-CNN

A fundamental task in computer vision called object detection includes locating and identifying objects in an image. Object detection architectures have significantly improved over time, resulting in higher accuracy and performance. From the earliest Convolutional Neural Networks (CNNs) to the most advanced Mask R-CNN, this section addresses the development of object detection architectures.

2.3.1 Convolutional Neural Network (CNN)

When an image is processed by a computer, it is represented as an array of numbers, as shown in Figure 2.3. Convolutional neural networks (CNNs) are specifically designed to extract local and abstract features from these arrays and use these features to predict the class of the image [57]. CNNs are extremely effective in tasks like image classification, object detection, video processing, natural language processing, and speech recognition due to their strong learning capabilities, which are primarily caused by the use of multiple feature extraction stages that can automatically learn representations from the data [58].

A Convolutional Neural Network (CNN) is made up of three primary types of neural layers: convolutional layers, pooling layers, and fully connected layers [57]. Each of these layers has a distinct function within the network.

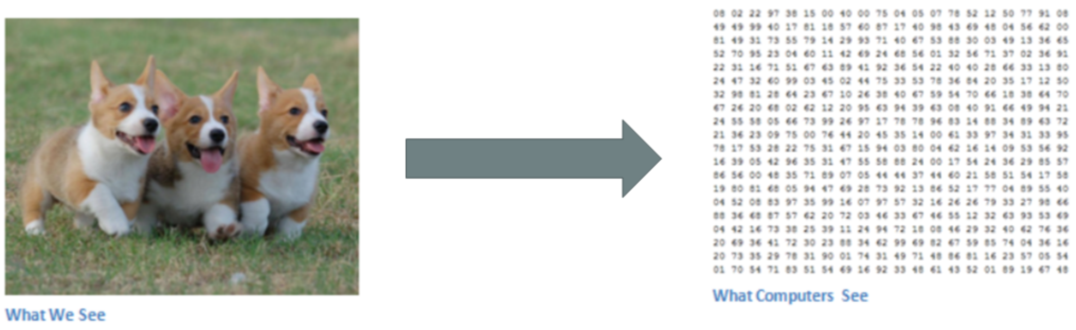


Figure 2.3: What we see vs What computers see. [59]

Convolutional Layer

One of the fundamental components of a Convolutional Neural Network (CNN) is a convolutional layer. Using a collection of trainable filters or kernels, it applies the mathematical action known as convolution to the input image or feature map [60]. The size of the filters is usually smaller than the actual image. In order to create a feature map that highlights the existence of various features in the input image, the convolution process involves sliding the filter over the input image, computing the dot product between the filter and one small portion of the image at each place, and then continuing repeatedly. [61].

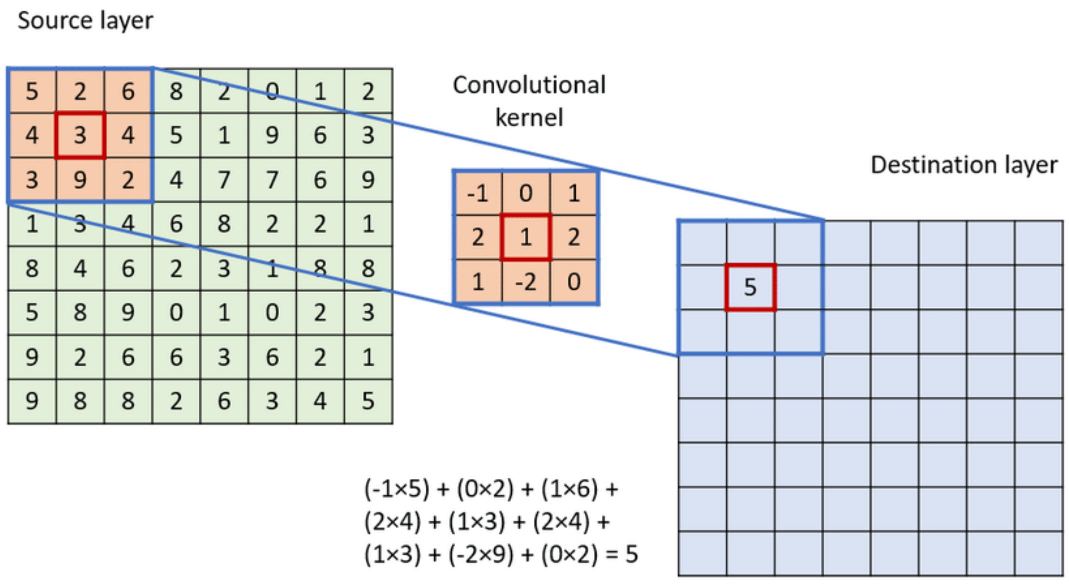


Figure 2.4: Schematic illustration of a convolutional operation. [62]

Figure 2.4 shows a representation that explains the convolutional process. The convolutional kernel fills the pixels in the destination layer as it passes over the source layer. By changing its weights during training, each filter in a convolutional layer can learn to recognize a different feature in the input image, such as edges, corners, or textures. A single convolutional layer can have many filters to learn a variety of features [63].

The results of a linear operation, such as convolution, are then fed through a nonlinear activation function, like ReLU (Rectified Linear Unit), to provide the network nonlinearity and allow it to learn more complicated features [64].

In order to develop a deeper, more complex neural network that is capable of carrying out more challenging image processing tasks, like object detection and image segmentation, convolutional layer output is frequently fed into a pooling layer, fully connected layer, or another convolutional layer [60].

Activation Layer

The activation function plays a critical role in convolutional neural networks (CNNs). It is responsible for introducing nonlinearity to the calculated features at each stage of the network, namely convolution, sub-sampling, and full connection. By using a nonlinear activation function, the network can effectively address the issue of limited expressiveness that may arise from linear operations alone [65].

Some common activation layers used in deep learning include:

1. *ReLU Layer*: The Rectified Linear Unit (ReLU) layer converts all negative input values to zero while maintaining positive values. This speeds up training and lessens the vanishing gradient issue [64].

2. *Sigmoid Layer*: The input is subjected to the sigmoid layer's application of the sigmoid function, which transforms the input to a value between 0 and 1. In tasks requiring binary classification, this is frequently utilized [66].

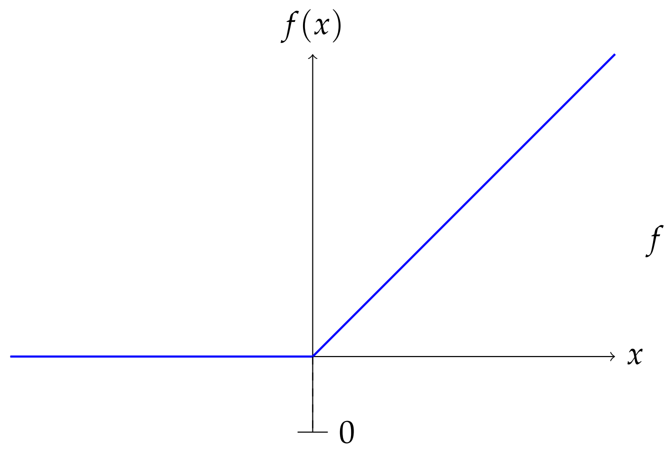
3. *Tanh Layer*: The input is subjected to the tanh function, which transforms the input to a value between -1 and 1, as part of the hyperbolic tangent (tanh) layer. This is frequently applied while performing multi-class classification tasks [66].

4. *Softmax Layer*: The softmax layer normalizes a vector of inputs so that their sum is 1. This is frequently used to transform the neural network's output

into a probability distribution over the classes in multi-class classification jobs [67].

In a neural network, activation layers are frequently combined with other layers such as convolutional layers, fully connected layers, and recurrent layers. The selection of the proper activation function can be a crucial step in the model design process because it can have a major impact on the network's performance.

ReLU is the most commonly used activation function in CNNs. It is a non-linear activation function that introduces non-linearity into the output of each neuron in a neural network [68].
The ReLU function is defined as follows:



$$f(x) = max(0, x) \tag{2.1}$$

where x is the input to the activation function. In other words, the output of the activation function is equal to the input if the input is greater than zero, if input is equal or less than to zero the output of the activation function is zero.

It sets all negative values in the output of the neuron to zero and leaves positive values unchanged. This makes the ReLU function computationally efficient and helps to mitigate the vanishing gradient problem.
The Softmax function is another frequently used activation function in CNNs, especially in the output layer for multiclass classification tasks. However, the mathematical computation of this function is not covered within the scope of this thesis.

Pooling Layer

A pooling layer is a sort of layer that is often added to a convolutional neural network (CNN) after a group of convolutional layers. The goal of a pooling

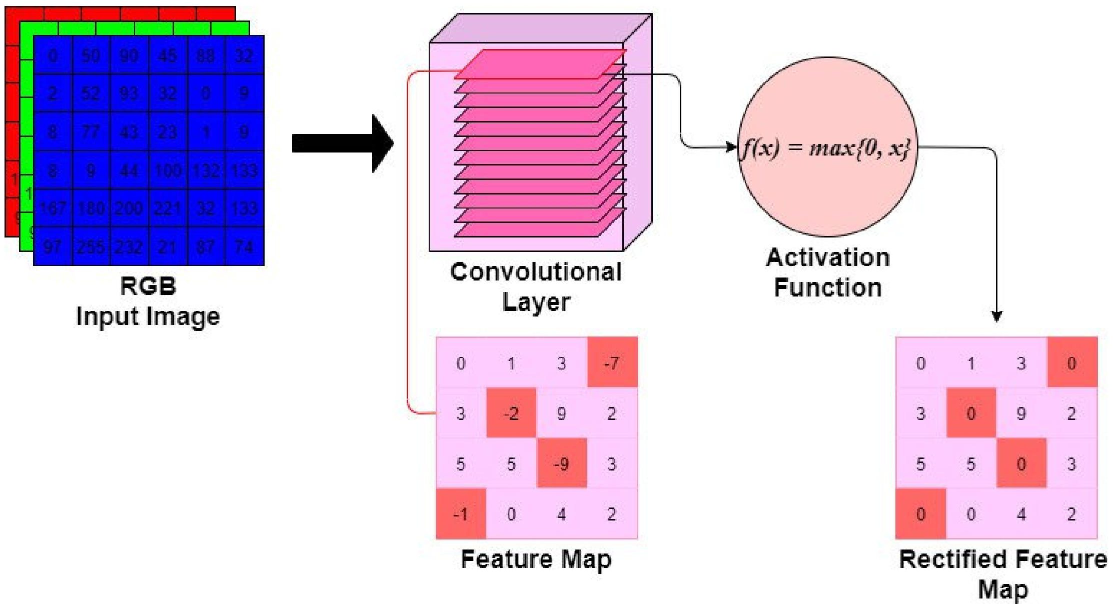


Figure 2.5: After passing through an activation function, the nodes in the feature map are subjected to a threshold such that any negative values are set to 0 before being outputted [69].

layer is to decrease the input's spatial dimension (height and width) while maintaining the key characteristics [70].

There are different types of pooling layers, including average pooling and max pooling.

Max pooling: Max pooling is the most popular kind of pooling layer, in which the largest value found within a rectangular region (referred to as the pool size) is chosen and utilized as the output value. This successfully decreases the input volume's dimensionality while retaining the most important data [71]. The two main parameters of a pooling layer in a convolutional neural network (CNN) are the kernel size and stride.

Kernel size: The size of the filter that is applied to the input feature map is determined by the kernel size. The kernel size determines the height and width of the filter, which is typically square in shape. For instance, a 2x2 kernel size indicates that the filter will be a square of that size.

Stride: For each pooling operation, the stride indicates how far the filter is shifted over the input feature map. For each pooling operation, the filter will be moved, for instance, by two locations with a stride of 2.

Figure 2.6 depicts how a convolutional neural network (CNN) can utilize max pooling to reduce the dimensionality of feature maps while still preserving the spatial arrangement of features.

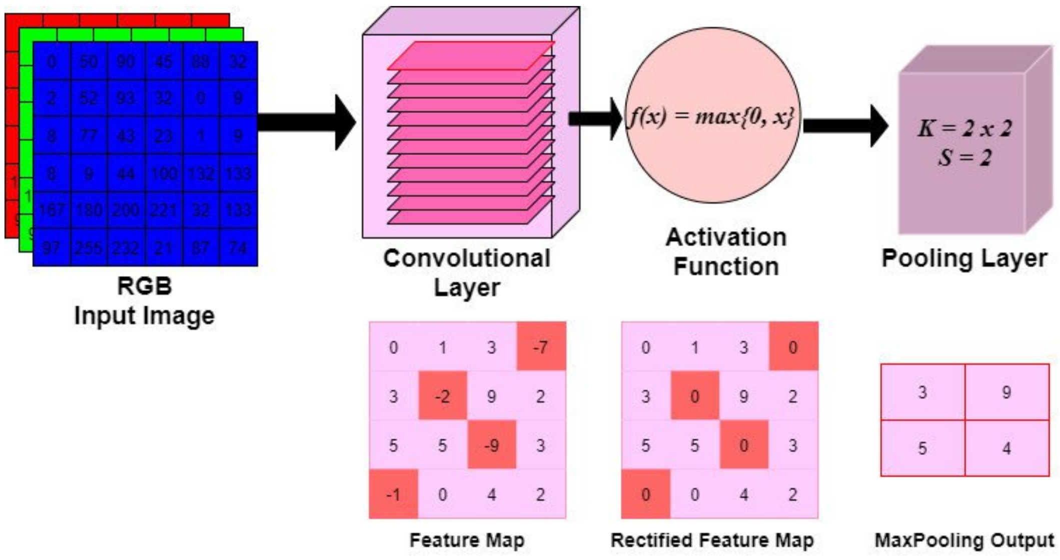


Figure 2.6: Representation of pooling layer with kernel size and stride of 2 and 2 respectively [69].

Fully Connected Layer

Fully connected layers are frequently utilized at the network's end to carry out the network's final classification or regression task in convolutional neural networks (CNNs). The final output is generated by these layers using the flattened vector of output from the previous convolutional and pooling layers. For classification tasks, the fully connected layer typically has the same number of neurons as classes in the dataset [72]. For example, if the dataset has 10 classes, the final fully connected layer would have 10 neurons.

The output from the fully connected layer is often passed through a softmax activation function to produce a probability distribution over the classes in the dataset. During training, the weights of the fully connected layer are learned using an optimization algorithm such as backpropagation [73].

If the model is not properly regularized, the fully connected layers in CNNs can have a lot of parameters, which can result in overfitting. In order to avoid overfitting and improve the model's performance, techniques including dropout, weight decay, and early stopping can be implemented [74].

The above figure 2.7 describes the architecture of a convolutional neural network (CNN) with convolutional, activation, and pooling layers. The CNN processes an image's input by running it through several convolutional layers, each of which is followed by an activation function. To make the feature maps

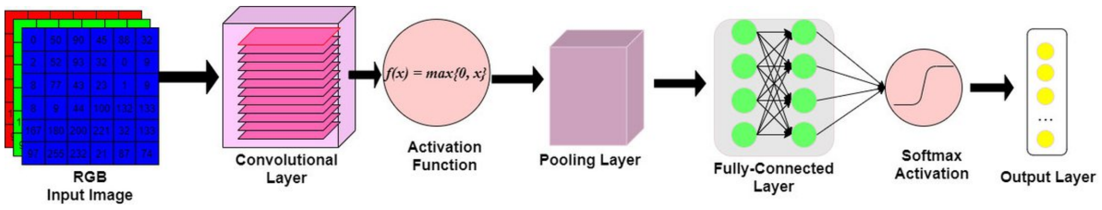


Figure 2.7: Complete architecture of CNN [69].

less dimensional, the output of the final convolutional layer is passed through several pooling layers. The final output of the network is created by feeding the flattened output from the last pooling layer into one or more fully connected layers.

2.3.2 R-CNN Algorithm

A deep learning architecture RCNN (Region-based Convolutional Neural Network) was introduced in 2014 by Girshick et al. to deal with the challenge of traditional object detection in images [32]. RCNN is designed to accomplish this by first producing a set of area proposals inside an image and then classifying those regions as either containing or not containing an object, as shown in figure 2.8. The goal of object detection is to precisely identify and determine the precise position of objects present in an image.

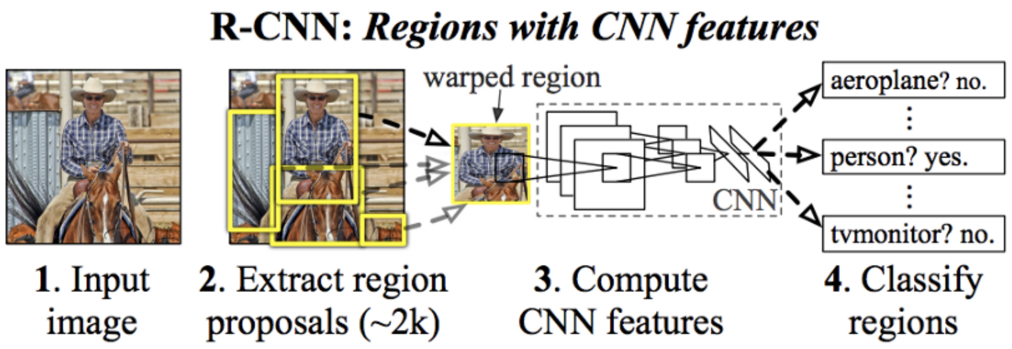


Figure 2.8: Complete architecture of R-CNN [32].

Architecture consists of three modules:

Region Proposal: The RCNN method starts by generating a list of candidate bounding boxes that might contain an object of interest. The selective search algorithm is used to achieve this, and it generates sub-segments of the image depending on color, texture, size, and shape [75]. These sub-segmentations are then combined iteratively to form objects using a bottom-up approach, resulting in a set of around 2000 region proposals or bounding box candidates, as depicted in figure 2.9. The RCNN algorithm uses these proposals to identify the regions of the image that require further analysis by the neural network.

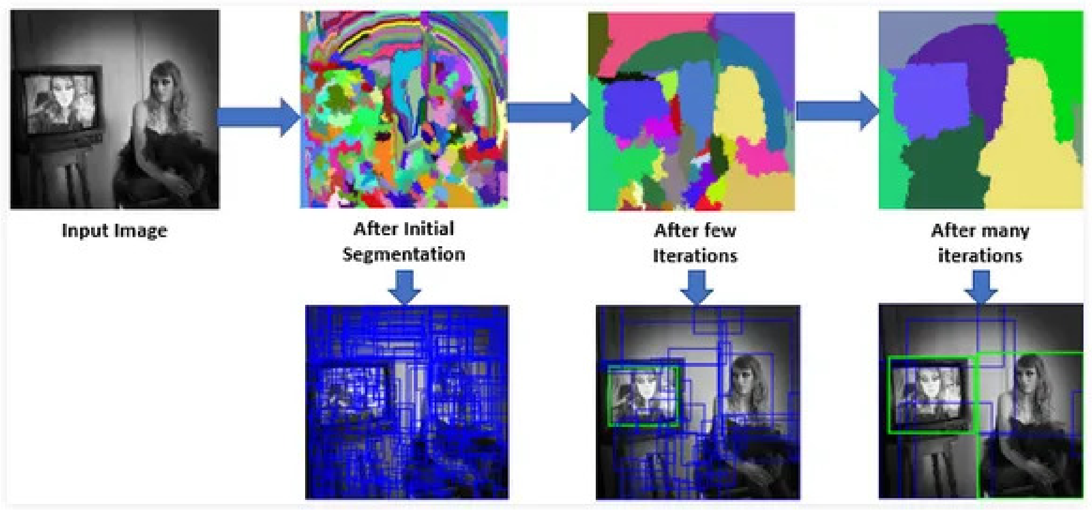


Figure 2.9: Demonstrates the working principle of the selective search algorithm for generating object proposals. [76].

Feature Extraction: In the second step, RCNN involves taking an image region proposal, resizing it to a fixed size (227x227), and passing it through the pre-trained AlexNet network to obtain a feature vector. The object in the region proposal is then classified using this feature vector as input to a Support Vector Machine (SVM) classifier. The SVM is trained on a set of labeled examples to learn the features that are most relevant for each object class [77].

Object detection and Localization: The feature vector is input into a group of fully connected layers in the last step when object detection and localization are accomplished. These layers produce a class label for the object in the region proposal as well as a well-defined bounding box that closely encloses the object. A bounding box regression technique is used to further refine the bounding box in order to increase localization precision [32].

Despite the improvement in accuracy, R-CNN algorithm still has some challenges such as:

1. R-CNN requires training multiple models separately, the training process can be very slow and computationally expensive.

2. R-CNN uses a fixed-size pooling operation to extract features from each

region proposal, which can result in a loss of spatial resolution.

3. R-CNN requires a lot of disk memory during training because it needs to save the feature maps of all the region proposals.

2.3.3 Fast R-CNN Algorithm

Fast R-CNN is an advanced version of R-CNN, introduced by R. Girshick in 2015 to address a few drawbacks of the original R-CNN algorithm. Fast R-CNN is a single-stage object detection algorithm that uses a convolutional neural network to classify objects and predict their bounding boxes in a single pass [33].

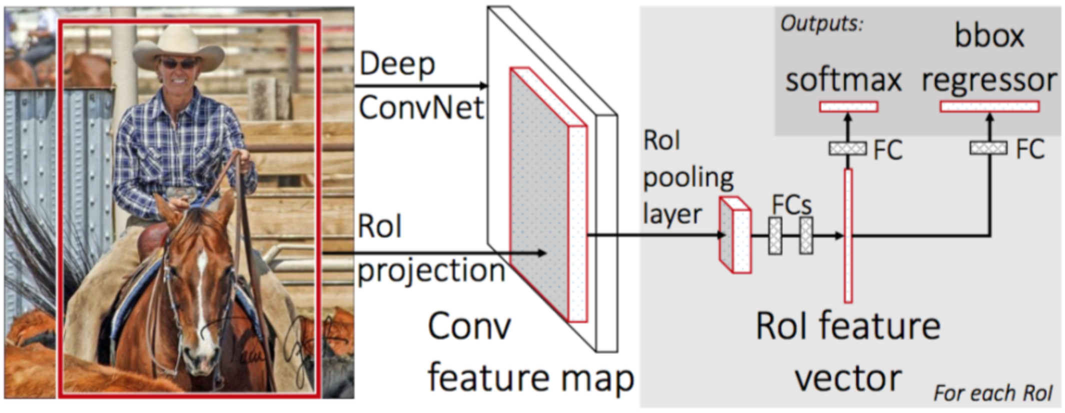


Figure 2.10: Fast R-CNN architecture. [33].

Fast R-CNN has several key innovations that differentiate it from R-CNN:

Feature Extraction: Instead of extracting features for each image patch from scratch, fast R-CNN first uses a CNN to extract features for the entire image.

Region Proposal: External region proposal methods like selective search are used to generate region proposals or regions of interest (ROIs), which are combined with corresponding feature maps to form patches for object detection.

ROI Pooling: Fast R-CNN introduces ROI pooling to extract features of each region proposal. This operation involves combining the feature maps of varying sizes from the fully connected layer and the preceding convolution layer into a unified, fixed-size representation.

Multi-Task Loss Function: Fast R-CNN trains classification and regression tasks in parallel using a multi-task loss function to accelerate model convergence.

In the previous model (R-CNN), the training process was complex and slow, requiring separate training of the CNN, SVM classifier, and bounding box regressor. As seen in Figure 2.10, the Fast R-CNN model uses a single network to simultaneously train the classification and bounding box regression tasks. The separate SVM classifier is replaced by a Softmax layer. This streamlines training and boosts efficiency and accuracy. These advancements speed up processing by 9 times, maintain spatial information, occupy less disk space, and increase training speed and detection accuracy.

Overall, Fast R-CNN is an advancement up from earlier object detection techniques, although there is still potential for development. Fast R-CNN relies on time-consuming and sometimes inaccurate external region proposal techniques, such as selective search, which affects detection accuracy [78].

2.3.4 Faster R-CNN Algorithm

Faster R-CNN is a popular object detection algorithm developed by S Ren & the Microsoft research team in 2015 [34].

By introducing a Region Proposal Network (RPN), the Faster R-CNN model addresses the bottleneck problem in Fast R-CNN. As a fully convolutional network, it does not require an additional mechanism for region proposal, such as selective search, and predicts object proposals directly from the convolutional feature maps of the input image. Figure 2.12 shows how the RPN centers a set of potential bounding boxes termed "anchors" at each sliding window location in the feature map. The RPN uses these anchors to predict item suggestions. They come in a variety of aspect ratios and scales to accommodate a wide range of object sizes and shapes. As the single network for object detection in Faster R-CNN, the RPN module acts as the "attention" of the network.

The use of a single convolutional neural network (CNN) for both region proposal and object categorization is one of the Faster R-CNN model's key achievements. This unified architecture, as can be seen in figure 2.11 improves the algorithm's overall speed and accuracy by minimizing the number of computing steps.

The Faster R-CNN algorithm uses a two-stage approach for object detection. In the first stage, the RPN generates region proposals by scanning the feature

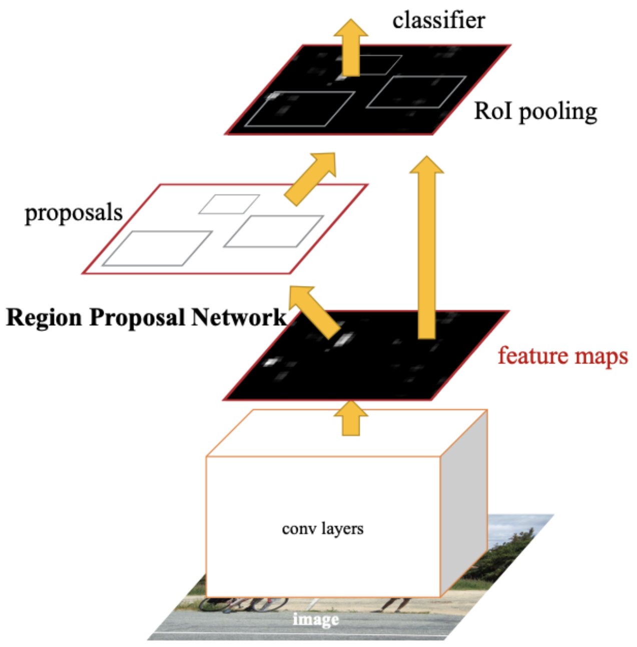


Figure 2.11: Faster R-CNN flow diagram [34].

map of the input image with a small convolutional network. These proposals are then refined using a bounding box regression network [34]. More details are discussed in section 3.4.2.

In the second stage, the region proposals are fed into a Fast R-CNN network for object classification and refinement of the bounding boxes. The Fast R-CNN network uses RoI pooling to extract a fixed-length feature vector from each region proposal and passes it through fully connected layers for classification and regression.

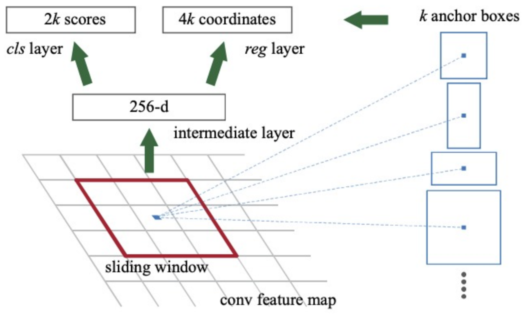


Figure 2.12: Region Proposal Network (RPN)

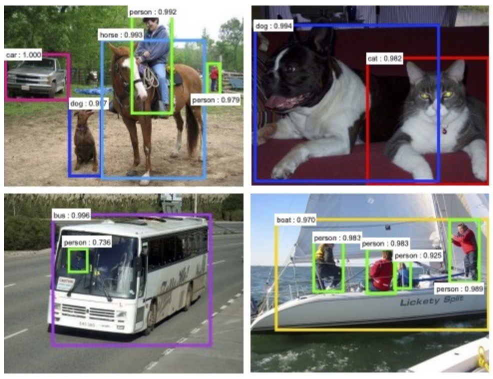


Figure 2.13: sample detections achieved through RPN proposals on PASCAL VOC 2007 test.

On a number of object detection benchmarks, including PASCAL VOC and

MS COCO, Faster R-CNN has attained cutting-edge performance. Because of its excellent precision and real-time performance, it has numerous applications in areas such as robotics, surveillance, and self-driving cars [34].

As a result, Faster R-CNN has greatly outperformed earlier models as an accurate and effective object detection system. Real-time performance without sacrificing accuracy has been made possible by using a single CNN for region proposal and identification of objects along with the RPN. It is a valuable addition to the field of computer vision and has numerous applications in real-world scenarios as shown in figure 2.13.

The architecture and working of Mask R-CNN will be explained later in section 3.4.2.

Chapter 3

Methodology

This section outlines the research approach and methods used to address the research questions. This chapter presents the research methodology employed in this study, serving as a roadmap to explain the research objectives and the suitable methodology chosen to achieve those objectives. The methodology provides a clear and systematic approach to direct the research process, ensuring that the objectives are effectively discussed and the research questions are answered. It offers a comprehensive overview of the research area, data collection methods, data analysis techniques, and any tools or software utilized in the research process.

3.1 Study area

In this study, **Salzburg City** has been selected as the research area. This city is located in Austria near the border with Germany, along the Salzach River, and is surrounded by the Eastern Alps. Since 1996, UNESCO has designated Salzburg, known for its rich cultural heritage, as a World Heritage Site [79]. In addition to its cultural significance, Salzburg City has shown great dedication to sustainability and renewable energy, aligning with Austria's ambitious targets of achieving carbon neutrality by 2040 and generating 100% electricity from renewable sources by 2030 [80]. Salzburg is recognized as one of Austria's greenest cities, and a role model for sustainable energy practices because 70% of its total energy usage is derived from renewable sources. It is renowned for its environmentally friendly public transportation system, which includes trolleybuses that are powered by electricity and have been in service since 1940 [81]. Salzburg's commitment to developing and enhancing its sustainable transportation system is consistent with the more general objective of reducing emissions and making the city more environmentally friendly. To achieve the target of 100% renewable electricity by 2030, Salzburg is planning to increase the adoption of rooftop solar photovoltaics installation. Monitoring the quantity of existing rooftop solar PVs poses several challenges. This study aims to collect valuable information about the size, location, and distribution of solar panels in the city. By detecting and analyzing these solar panels, policymakers, and urban planners can develop efficient regulations, allocate resources effectively, and optimize the utilization of solar energy in the city. It can also aid in identifying areas with high solar potential that can benefit from targeted interventions and further promote the use of renewable energy. Ultimately, the study aims to promote sustainable energy practices and the creation of a more environmentally friendly and energy-efficient infrastructure. In conclusion, Salzburg city's high solar energy potential, dedication to renewable energy initiatives, potential for policy-making and resource allocation, and focus on sustainable development make it a suitable research area to identify solar panels using satellite images.

3.2 Literature review

The main objective was to explore the advantages of using satellite imagery in deep learning for rooftop solar PV detection through this literature review. To achieve this, a scoping review methodology was utilized, which involves mapping the existing literature in a specific research area [82].

By conducting the scoping review, the study seek to gather both existing knowledge and new insights regarding the utilization of deep learning methods and satellite data for the identification of rooftop solar PV installations. This comprehensive review aimed to provide a good understanding of the current state of research, identify key trends, methodologies, and challenges, and identify potential areas for further investigation and improvement in this domain.

During the literature review, a thorough search for relevant scholarly articles, conference papers, and technical reports was conducted using the snowball searching technique [83]. This involved utilizing popular databases such as IEEE Xplore, ResearchGate, ScienceDirect, and Google Scholar to access a diverse range of publications in the field. Specific search terms, including "Rooftop solar PV detection," "deep learning," "satellite imagery," "Object detection," and related terms, were used to ensure a comprehensive search.

The gathered literature provided insightful information on a number of topics, including the advantages and challenges of integrating satellite imagery in deep learning-based approaches, the methodology and techniques used, and the performance evaluation measures employed in comparable studies. By synthesizing and analyzing the findings from the literature review, a solid understanding of the current knowledge and gaps in the field was obtained.

3.3 Comparative analysis of deep learning algorithms for object detection and segmentation

The primary objective of this comparison is to identify the most effective deep-learning model for instance segmentation in satellite imagery. An extensive literature review was done to look at previous work on instance segmentation in satellite imagery in order to accomplish the aforementioned objective. The review aimed to identify and compare various deep-learning models that have been applied to object detection in satellite imagery. The intention of this comparative analysis was to assess the effectiveness of these models and determine their suitability for specific tasks.

The literature search followed the same methodology as previously described, ensuring a comprehensive collection of relevant studies for this research. In line with the methodology outlined by S. Shrivastava et al. [84]. The comparative analysis in this study incorporated multiple factors to evaluate the performance of deep-learning models for instance segmentation in satellite imagery.

These factors were included in widely accepted performance metrics (Moree details are provided in Section 6.4):

1. Mean Average Precision (mAP): mAP is a widely used performance metric in object detection and instance segmentation tasks. It measures the precision-recall trade-off by calculating the average precision at different recall levels. It takes into account the precision of predicted instances and their overlap with ground truth instances. Higher mAP indicates better model performance in accurately detecting and localizing objects.Mean Average Precision (mAP): mAP is a widely used performance metric in object detection and instance segmentation tasks. It measures the precision-recall trade-off by calculating the average precision at different recall levels. It takes into account the precision of predicted instances and their overlap with ground truth instances. Higher

mAP indicates better model performance in accurately detecting and localizing objects [85].

2. Intersection over Union (IoU): Intersection over Union, is a performance metric commonly used to measure the degree of overlap between two bounding boxes or segmentation regions. In the task such as object detection and segmentation, IoU evaluates how well the predicted region aligns with the ground truth region. The resulting value ranges from 0 to 1, where 0 indicates no overlap and 1 indicates a perfect match [86].

3. F1 score: The F1 score is a performance metric that combines precision and recall into a single value, providing a balanced assessment of the model's performance. Precision measures the model's ability to correctly identify positive instances out of all predicted positive instances, while recall measures the model's ability to correctly identify positive instances out of all actual positive instances. The F1 score considers both precision and recall by calculating their harmonic mean. The value of F1 ranges from 0 to 1, with 1 indicating the best possible performance [87].

4. The comparison also focused on the model's ability to detect small objects and accurately segment objects in satellite imagery. To ensure uniformity and fairness in the evaluation process, the comparison was carried out using the same dataset.

This analysis aims to give a thorough evaluation of the performance of deep-learning models in terms of their accuracy, efficiency, and overall performance in instance segmentation tasks. By incorporating the aforementioned metrics, this research aims to identify the best deep-learning model for achieving precise and efficient instance segmentation in satellite imagery.

3.4 Case study: Instance segmentation of rooftop solar PVs installations in satellite imagery using Mask R-CNN

The case study aims to leverage the Mask R-CNN (Region-based Convolutional Neural Network) model for the detection of solar panels in satellite imagery for the Salzburg area.

3.4.1 Data collection

One of the crucial steps in research study is data collection, and this case study is no exception. The objective was to collect satellite imagery data that would be suitable for training and evaluating the Mask R-CNN model for the detection of solar panels in satellite imagery. In this study, a total of 1000 high-resolution images were gathered from Land Salzburg [88], each having a resolution of 0.30m and a size of 6250x5000 pixels. These images were in the commonly used geotiff format for storing geospatial data. The Coordinate Reference System (CRS) employed for these images was EPSG:31285, which corresponds to the MGI / Austria M31 CRS.

During the data collection procedure, ethical considerations were taken into account. This included ensuring compliance with data privacy regulations and obtaining the necessary usage rights for the satellite imagery data. The study took into account the privacy and rights of any individuals or organizations linked to the satellite imagery while also preserving the accuracy and confidentiality of the information gathered.

3.4.2 Mask R-CNN Algorithm

Mask-RCNN (Mask-Region Convolutional Neural Network) is a state-of-the-art algorithm for instance segmentation, which evolved from the previous versions of RCNN, Fast-RCNN, and Faster-RCNN. The first three versions were developed for object detection, while Mask-RCNN is specifically designed for instance segmentation [35].

Mask-RCNN differs from Faster-RCNN in a number of ways, including the use of RoIAlign rather than RoIPool and the use of the Fully Convolutional Network (FCN) for mask prediction. Mask-RCNN generates a mask independently for each class [89].

Mask-RCNN has two main stages. In the first stage, it uses a Region Proposal Network (RPN) to propose candidate object bounding boxes. In the second stage, it classifies the candidate boxes, refines the boxes, and predicts masks for each instance. Overall, Mask-RCNN achieves state-of-the-art results in instance segmentation and object detection tasks. So the overall structure can be illustrated by the following figure 3.1.

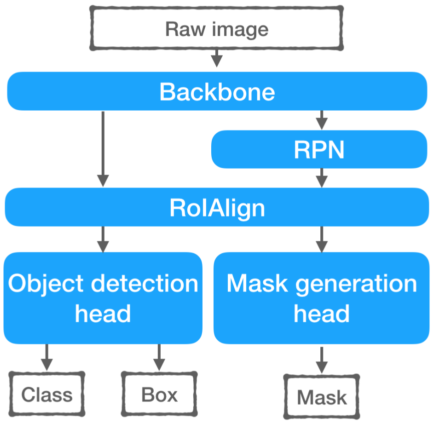


Figure 3.1: MaskR-CNN framework for instance segmentation [90].

A thorough explanation of Mask R-CNN and its various components, including a backbone, a Region Proposal Network (RPN), a Region of Interest (RoI) alignment layer, a bounding-box object detection head, and a mask generation head is provided in the following sections. The first four components make up the Faster R-CNN model, which is extended by adding the mask generation head.

Backbone

The backbone is a crucial component of Mask R-CNN responsible for feature extraction from the image. It acts as a feature extractor by applying convolutional operations to the input image to generate a feature map. Commonly used backbones in Mask R-CNN are Residual Networks (ResNets) with or without a Feature Pyramid Network (FPN). ResNet is a popular choice for backbone as it has demonstrated excellent performance in various computer vision tasks [91].

In a ResNet backbone, the input image goes through multiple residual bottleneck blocks to create a feature map. The residual blocks are designed to improve the training of very deep neural networks by addressing the vanishing gradient problem [92]. The input to each block is added to its output, allowing the network to learn a residual function that predicts the difference between the input and output features. This approach enables the network to learn more meaningful features without adding additional layers.

Figure 3.2 illustrates how a ResNet backbone is constructed using multiple residual bottleneck blocks, where each block has different channel d/d′ configurations. The ResNet backbone is composed of several residual blocks, and each block includes two paths for input. The first path consists of multiple convolutional layers, and the second path is a shortcut connection. The outputs from both paths are added together element-wise, making it easier for gradients to propagate through the blocks [90].

This allows each block to learn an identity function more efficiently.

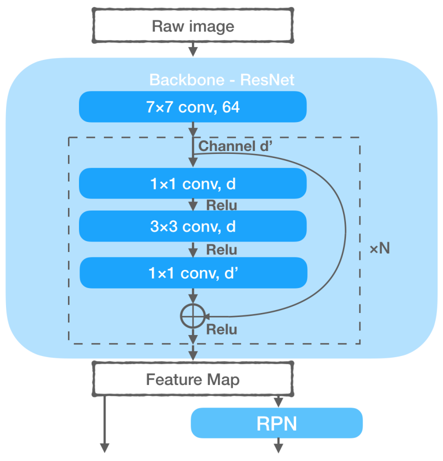


Figure 3.2: ResNet backbone with residual bottleneck blocks [90]

The FPN is an extension of the ResNet backbone that uses a top-down architecture with lateral connections to construct an in-network feature pyramid from a single-scale input, allowing for features to be extracted at multiple scales [93]. This approach improves the detection of objects at different scales, which can be particularly useful for detecting small or large objects in an image. Additionally, FPN reduces the amount of computation required to extract features from an image, making the overall process faster and more efficient.

The final layer of the convolutional neural network backbone generates a feature map that encodes high-level information about the input image, including the presence of various object instances, their categories, and spatial properties. This feature map is then utilized by the Region Proposal Network (RPN) to generate a set of region proposals.

In summary, the backbone in Mask R-CNN is responsible for generating a feature map from the input image using convolutional operations. ResNet and FPN are commonly used as backbones, and they help in improving the performance of the network by enabling it to learn more meaningful features and capture features at multiple scales.

Region Proposal Network (RPN)

The Region Proposal Network (RPN) is a crucial component in object detection networks like Mask R-CNN. Its primary function is to propose regions that may contain objects within an input image.

The diagram in Figure 3.4 shows the working of RPN where it can be seen that the RPN takes a feature map as input, which is obtained by applying convolutional layers to the image. It slides a small window (typically 3x3) over each position in the feature map and for each position, it generates a set of anchor boxes with different scales and aspect ratios that are placed evenly across the image to cover it completely which can be seen in figure 2.12. These boxes overlap with each other to fill up the image as much as possible. For each anchor box, the RPN calculates the object's bounding box with the highest overlap, which is measured by the Intersection Over Union (IOU) ratio [94].

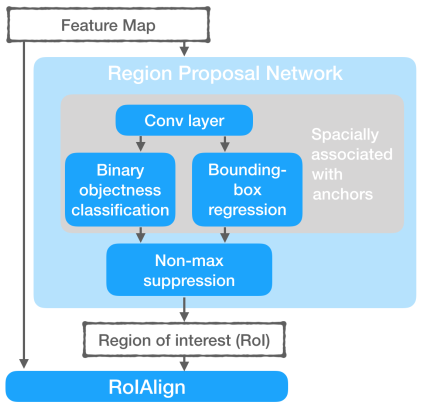


Figure 3.3: Region Proposal Network (RPN) in the Mask R-CNN architecture [90]

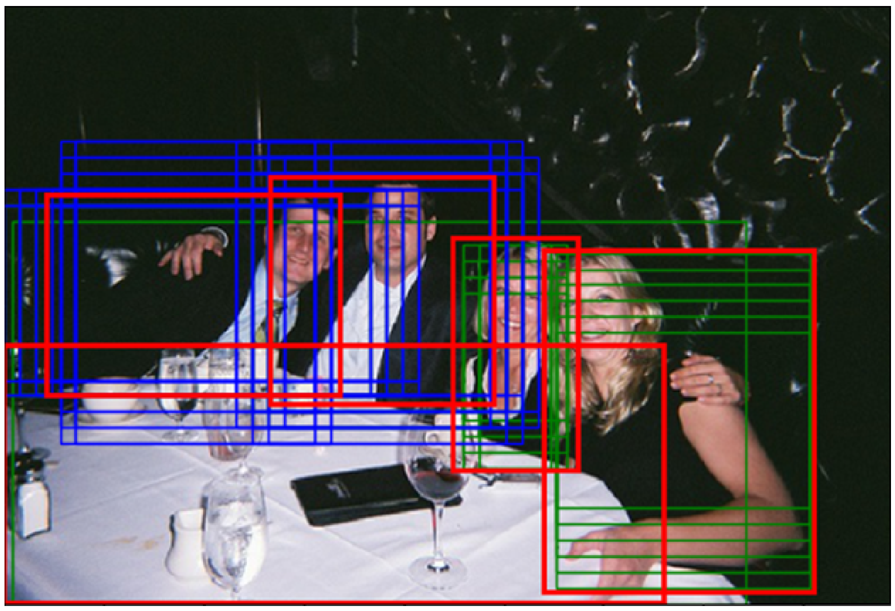


Figure 3.4: An illustration of how anchor boxes are used in object detection networks [94]

1. *Ground truth boxes are marked in red.*
2. *For each ground truth box, find the anchor box with the highest overlap and mark it in green.*
3. *Mark all anchor boxes whose overlap with any ground truth box exceeds a certain threshold in blue.*

If the highest IOU ratio is greater than 50%, the anchor box determines the object. If the IOU ratio is greater than 40% but less than 50%, the detection is ambiguous, and if it is less than 40%, the RPN predicts no object in that anchor box.

The RPN also includes a classifier that gives the probability of a proposal containing the target object and a regression that regresses the coordinates of the proposals. For each anchor box, the RPN outputs two values: a binary classification score indicating whether the anchor box contains an object or not, and a set of four offsets that can be used to modify the dimensions and placement of the anchor box to better fit the object [95].

After generating the set of proposed regions, non-maximum suppression (NMS) is applied to remove any overlapping proposals that may suggest the same object. This step ensures that only the most confident proposals are considered for further processing.

The RPN (Region Proposal Network) generates a set of proposed regions, known as RoIs (Region of Interest), which are then passed to the subsequent stage of the object detection network. In this stage, the RoIs undergo further processing, including feature extraction and classification, to identify and classify the objects present in the image.

ROI Align

In the Mask R-CNN, the proposed regions (RoIs) generated by the RPN are aligned with their corresponding areas in the feature map through the RoI Align layer. This step is crucial to address the problem of location misalignment caused by quantization in the RoI pooling. To achieve proper alignment, RoIAlign avoids hash quantization and instead uses non-integer values such as $x/16$. Bilinear interpolation is then employed to accurately compute the floating-point location values in the input [35]. This is important since RoIs come in different locations, scales, and aspect ratios, whereas the feature map has a fixed shape.

The ROI Align layer in Mask R-CNN performs the following steps, which can also be seen in Figure 3.5:

1. Input: The layer takes as input the feature map obtained from the backbone network and the proposals generated by the RPN.
2. Dividing RoIs into grids: Each proposal (RoI) is divided into a fixed number of equal-sized bins (e.g., 7x7 or 14x14), depending on the specific implementation.
3. Sampling: Within each bin, several (usually four) points (or dots) are uniformly sampled to obtain the corresponding features from the feature map.
4. Bilinear interpolation: For each sampled point, the layer uses bilinear interpolation to compute the exact feature value at the sub-pixel location. This interpolation is done between the four nearest neighboring feature map points, which allows for sub-pixel accuracy.
5. Pooling: The feature vectors obtained from the sampled points within each bin are then pooled together (e.g., by taking their average or max value) to obtain a single, fixed-sized feature map for each RoI.
6. Output: The output of the ROI Align layer is a set of fixed-size feature maps (one for each RoI).

After the ROI Align layer aligns the features from the RoIs with their corresponding areas in the feature map, the aligned features are fed into fully connected layers for classification and bounding box regression. Additionally, a separate fully convolutional network is used to predict a binary mask for each object instance within the RoI. RoiAlign improved Mask R-CNN mask accuracy by relative 10 % to 50 % [35]. By using fixed-size feature maps for each RoI, the model can process all RoIs efficiently, regardless of their size and aspect ratio.

Loss function

The loss function serves as a measure of the discrepancy between the predicted output and the actual label. A more reliable model is one with a lower loss value. The main objective of the training is to reduce the loss function.

In the Mask R-CNN architecture, the total loss function comprises three key components: the candidate box classification loss, the bounding box regression loss, and the mask segmentation loss [35].

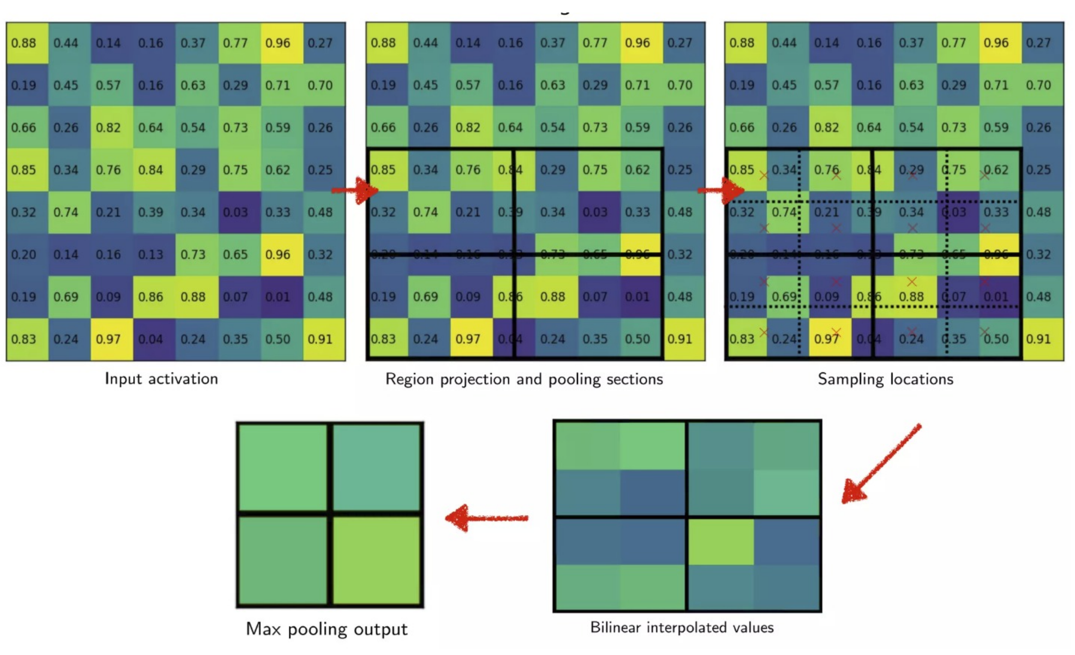

Input activation
Region projection and pooling sections
Sampling locations
Max pooling output
Bilinear interpolated values

Figure 3.5: Steps followed in ROI Align layer [96]

$$L_{total} = L_{cls} + L_{box} + L_{mask} \tag{3.1}$$

Classification loss The classification loss (L_{cls}) is used to measure the difference between predicted class scores and the ground-truth class labels for each proposal (RoI). The goal is to correctly classify each RoI as either foreground (object) or background (non-object). It helps to exclude boxes whose predicted category is background during the candidate box generation process, improving the model's accuracy in predicting candidate boxes. It is typically calculated using cross-entropy loss, which is given by the following formula:

$$L_{cls} = \frac{1}{N_{cls}} \sum_{i} -\log[p_i^* p_i + (1 - p_i^*)(1 - p_i)] \tag{3.2}$$

where N_{cls} represents the normalization parameter, p_i^* is the ground-truth label (either 0 or 1) for the i-th RoI, and p_i is the predicted probability for the i-th RoI [97].

Bounding box loss The Bounding box loss (L_{box}) function measures the difference between the predicted bounding box coordinates and the actual bounding box coordinates labeled in the training data. Its purpose is to train the model to accurately predict the coordinates of the bounding box surrounding an object in an image.

The bounding box loss is calculated only for positive proposals, which are the proposals that have an Intersection over Union (IoU) with a ground truth box greater than a certain threshold. For negative proposals (those that do not overlap with any ground truth box), the bounding box regression is not performed.

The formula for the bounding box loss can be written as:

$$L_{bbox} = \frac{1}{N_{bbox}} \sum_{i \in Pos} \text{smooth}_{L1}(t_o - v_o) \tag{3.3}$$

where N_{bbox} is the number of Region of Interests (RoIs), Pos is the set of RoIs that have an Intersection over Union (IoU) greater than a certain threshold with a ground truth bounding box, t_o is the predicted offset parameter, v_o is the actual offset parameter, and smooth L1 is a loss function defined as:

$$\text{smooth}_{L1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1 \\ |x| - 0.5, & \text{otherwise} \end{cases} \tag{3.4}$$

The offset parameters, t_o and v_o, are used to compute the predicted and actual bounding box coordinates, respectively. Specifically, t_o represents the predicted offset from the proposal box to the ground truth box, and v_o represents the actual offset from the proposal box to the ground truth box. These offsets are defined as follows:

$t_o = (t_x, t_y, t_w, t_h)$ & $v_o = (v_x, v_y, v_w, v_h)$ where t_x, t_y, t_w, and t_h are the predicted offset values for the X-coordinate, Y-coordinate, width, and height of the bounding box, respectively, and v_x, v_y, v_w, and v_h are the corresponding ground truth offset values [97].

Mask segmentation loss The mask segmentation loss (L_{mask}) is an important component in training the Mask R-CNN model for instance segmentation. It measures the difference between the predicted binary mask and the ground truth mask for each instance in the image.

The loss function can be defined as the average binary cross-entropy loss between the predicted mask and the ground truth mask for each instance in the image. This is expressed mathematically as:

$$L_{mask} = -\frac{1}{m^2} \sum_{i=1}^{N} [y_i \log(y_i^k) + (1 - y_i) \log(1 - y_i^k)] \tag{3.5}$$

Here, N is the number of RoIs, k is the category index for the RoI, m is the size of the predicted mask, and y_i and y_i^k are the ground truth and predicted binary masks for the i-th RoI and k-th category, respectively [98].

The loss function penalizes incorrect predictions of the binary mask by assigning higher loss values for larger deviations from the ground truth mask. By minimizing this loss function during training, the model is able to accurately segment objects in an image at the pixel level.

3.4.3 Experimental duration

The experimental duration for this case study spanned from December 2022 to May 2023. During this period, various stages of the experiment were conducted, including data acquisition, pre-processing, model training, validation, and performance evaluation. Throughout the entire experiment, it was made sure to document every step taken. This included details about how the experiment was set up, where I obtained the data, and the methodologies used. It was important to maintain transparency and ensure that this work could be reproduced by others.

The detailed findings and results obtained from this case study, including the performance metrics and evaluation of the deep-learning model, will be presented and discussed in subsequent chapters.

Overall, the six-month experimental duration allowed for a comprehensive and rigorous analysis of the proposed methodology and its effectiveness in addressing the challenges of instance segmentation in satellite imagery.

Chapter 4

Advantages of satellite imagery in deep learning

Satellite data plays a crucial role in efficiently mapping and monitoring Earth's resources, and events. Satellite images are valuable for a variety of applications, including spatial planning, environmental monitoring, and natural resource management, because they contain a wealth of data [99].

In conjunction with satellite data, deep learning has rapidly grown and gained widespread recognition as one of the groundbreaking technologies in the field of big data analysis [26]. The combination of satellite data and deep learning techniques has revolutionized various fields, It has led to advancements in Earth observation and environmental monitoring, agriculture and crop monitoring, urban planning and infrastructure development, climate change and environmental modeling, disaster management and emergency response, transportation and navigation, and natural resource management [100, 101, 102, 103, 104]. These technologies provide valuable insights, improve decision-making, and support sustainable practices across these domains. Satellite imagery offers the advantage of providing high-resolution images that capture fine details of the Earth's surface. However, extracting meaningful insights from such unstructured data on a large scale can be challenging, even with intensive manual analysis. Recent developments in deep learning methods have demonstrated promising results in addressing this challenge, leading to significant improvements in fundamental computer vision tasks like object detection and classification [105]. By analyzing high-resolution satellite imagery and applying object detection algorithms, it is now possible to identify and track vehicles on the ground [106]. In the realm of transportation and logistics, satellite-based vehicle detection and tracking can provide valuable information for traffic management, route optimization, and infrastructure planning.

Satellite imagery offers the advantage of eliminating double counting and reducing the chances of incorrect counts. This is because of the extensive coverage offered by satellite images, which capture large areas in a single acquisition. Unlike other surveying methods that may require human presence, satellite remote sensing avoids disturbing the species or ecosystems being observed [107].

Ishii et al. conducted an experiment to compare the performance of convolutional Neural Networks (CNN) and support vector machines (SVM) in object recognition using Landsat 8 images. The results of their study demonstrated that CNN outperformed SVM in this particular task. This finding underscores the potential of deep learning techniques, specifically CNN, for achieving superior performance in image recognition tasks involving Landsat 8 imagery [108].

Satellite imagery has provided a vast amount of valuable information to researchers, planners, and stakeholders in both public and private sectors. It enables efficient policy-making and decision-making processes by facilitating tasks such as mapping the aftermath of severe weather events like cyclones and earthquakes or studying the gradual changes in urban development within cities over extended periods [109].

Feng et al. conducted a study in which the authors focused on the automated detection and category recognition of ships from high-resolution aerial images using deep learning models. The research aimed to develop a robust and efficient method for ship identification and classification in order to enhance maritime surveillance and monitoring [110].

Remote sensing data is characterized by its inherent geospatial nature, meaning that it is naturally linked to specific locations in the Earth's geographic space. This geospatial property enables the integration and fusion of pixel information from remote sensing imagery with other types of data sources, such as geographic information system (GIS) layers, geo-tagged images obtained from social media platforms, or data collected from different sensors [111]. By combining and analyzing these diverse datasets, a more comprehensive understanding of the Earth's features and phenomena can be obtained, facilitating applications in various fields such as environmental monitoring, urban planning, agriculture, and disaster management [112].

The Copernicus program ensures the ongoing acquisition of data over extended periods of time, providing a consistent and reliable source of information. As an example, the Sentinel-1 satellite captures images of the entire Earth every six days [113]. This frequent coverage allows for up-to-date monitoring and analysis of various phenomena on a global scale.

Existing techniques for calculating rooftop solar energy, like traditional surveying or using costly tools like LIDAR, have drawbacks in terms of speed as well as economically. Some techniques also rely on specific data formats or high-resolution imagery, which adds to the complexity and demands on resources. Although these methods can result in accurate estimates, they are time-consuming, expensive, and labor-intensive.

However, utilizing deep learning advancements, particularly in image segmentation, offers the potential for resolution. It is now possible to precisely identify rooftops in satellite imagery by using deep learning techniques [114]. Deep learning algorithms can effectively analyze satellite images, extract relevant features, and classify rooftop regions with high precision [115].

Deep learning applied to Earth observation (EO) images is a vibrant and rapidly evolving research field. It has shown remarkable success in various tasks, including road detection, classification, and dense labeling of EO data [116].

Segmentation of satellite images offers a convenient and affordable method for identifying solar arrays installed on rooftops and on the ground across a specific area [117].

Overall, integration of deep learning with satellite imagery provides numerous benefits, such as the capacity to analyze high-resolution images, process extensive coverage areas, leverage multi-spectral and hyperspectral data, conduct temporal analysis, automate object detection, achieve precise object segmentation and classification, and improve decision-making across various domains. This integration facilitates thorough and efficient analysis of satellite imagery, resulting in valuable insights and well-informed decision-making.

Chapter 5

Best deep learning model for object detection and instance segmentation: Comparative analysis and insights

Instance segmentation is vital for precise object identification in satellite imagery, and deep learning models have revolutionized this field. Mask R-CNN has emerged as a leading approach, excelling in object detection and segmentation. Several studies have compared Mask R-CNN, a widespread object detection and instance segmentation model, with other state-of-the-art models. These comparative studies aim to determine the best model for these tasks and assess the performance of different approaches. These comparisons shed light on model performance across datasets and object categories. This chapter provides an in-depth analysis of the literature, aiming to identify the most accurate, robust, and efficient model for satellite imagery analysis.

In the study conducted by Mahmoud et al., Mask R-CNN was compared to several object detection methods, namely Faster-RCNN, YOLO1, YOLO2, SSD, and R-FCN [118]. The comparison focused on evaluating the performance of these techniques in terms of average precision (AP). The mean AP values obtained for each method on the NWPU dataset are as follows: Faster RCNN (FRCN) achieved an AP of 76.4, YOLO1 obtained 66.7, YOLO2 achieved 79.7, SSD obtained 89.4, R-FCN achieved 92.8, and the proposed method achieved the highest AP of 95. This indicates that the proposed method outperformed the baseline techniques in terms of accuracy.

In a comparative study between Mask R-CNN and U-Net for instance segmentation, it was observed that Mask R-CNN outperformed U-Net with a slightly higher mean average precision (mAP) score. Mask R-CNN achieved an mAP of 0.519, while U-Net achieved an mAP of 0.515 [119]. These results indicate that Mask R-CNN demonstrated slightly better performance in accurately segmenting and detecting instances within the images compared to U-Net.

According to He et al. in their paper, Mask R-CNN outperforms previous state-of-the-art single-model results on the COCO instance segmentation task. They mention that all instantiations of their model surpass baseline variants of previous state-of-the-art models, including Multi-Task Network Cascades (MNC) and Fully Convolutional Instance-Aware Semantic Segmentation (FCIS), which were the winners of the COCO 2015 and 2016 segmentation challenges, respectively. The mean Average Precision (mAP) values achieved by these models are as follows: MNC achieves an mAP of 44.3, FCIS achieves an mAP of 49.5, and Mask R-CNN achieves an mAP of 60.0 [120, 121]. This suggests that Mask R-CNN demonstrates superior performance compared to these previous models in terms of instance segmentation accuracy on the COCO dataset.

Sanchez et al. claim that when segmenting objects in images, Mask RCNN can locate objects with high precision. Although YOLO also performs precise object recognition, it has difficulties accurately detecting small objects [122]. This suggests that Mask RCNN is particularly capable of providing accurate object detection, which can be especially useful when dealing with small objects in satellite imagery, like solar panels.

In a study conducted by Kureshi et al., the performance of Mask RCNN was compared to other state-of-the-art methods using the COCO 2014 and Pascal VOC 2012 datasets. The results revealed that Mask RCNN performed competitively in terms of mean average precision (mAP) across different object categories when compared to methods like Faster R-CNN, SSD, and SSOD (Semi-supervised object detection) [123]. Mask RCNN offers the advantage of object segmentation, which is not available in YOLO and Faster RCNN. This additional capability allows Mask RCNN to outperform these models, despite being a two-stage model. In comparison to Faster RCNN specifically, Mask RCNN demonstrates superior performance [123].

In earlier research on object detection with multichannel imagery, segmentation-first methods utilizing object-based convolutional neural networks (CNNs) were primarily used. This method has drawbacks because it gives every instance of an object the same semantic information. Mask-RCNN, in contrast, offers distinct object-level information that enables it to handle overlapping objects and produce promising outcomes [124]. As a result, the predominant architecture for instance segmentation in remote sensing data is the Mask-RCNN/Faster-RCNN approach.

Zhao et al. conducted an experiment where they applied the Mask-RCNN algorithm with ResNet101-FPN as the backbone structure for building extraction. They introduced a boundary regularization technique and compared their approach to traditional Mask-RCNN models using the F1 score metric. The results showed that the Mask-RCNN model achieved a higher F1 score of 0.717, surpassing the performance of their proposed algorithm, which achieved an F1 score of 0.713 [125].

In scenarios where computational resources are not a constraint, two-stage detectors generally achieve higher detection accuracies compared to one-stage detectors. Support for this observation comes from the fact that many approaches that succeed in well-known detection challenges primarily make use of two-stage frameworks. The flexibility and suitability of two-stage frameworks for region-based classification contribute to their superior performance. The most commonly employed frameworks in this category include Faster RCNN, RFCN, and Mask RCNN. On the other hand, one-stage frameworks such as YOLO and SSD typically exhibit lower performance when detecting small objects but can be competitive in detecting large objects, as demonstrated by previous studies [126, 127].

Comparing different models and determining the best one for a specific application can indeed be a challenging task. Ultimately, the selection of the best model involves carefully considering the trade-off between speed and accuracy based on the specific needs of the application and the available computational resources.

The literature analysis highlights the effectiveness of Mask R-CNN for object detection and instance segmentation in satellite imagery. The studies compared Mask R-CNN with other state-of-the-art models, including Faster-RCNN, YOLO, SSD, R-FCN, MNC, and FCIS, across different datasets and object categories. Overall, the literature supports the conclusion that Mask R-CNN is the most accurate, robust, and efficient model for object detection and instance segmentation in satellite imagery, outperforming other state-of-the-art models in various performance metrics.

Chapter 6

Experimental Setup

This chapter presents the experimental setup employed in the object detection research project. The success of any object detection model relies on the quality and diversity of the training dataset. Therefore, the first section of this chapter, "Dataset," describes the process of data preparation, including dataset collection, annotation, and formatting. The dataset was carefully curated to include a diverse range of solar panel images, ensuring the model's ability to detect objects in various scenarios and conditions. The subsequent section, "Hardware," details the hardware configuration utilized for the research project, encompassing the processor, GPU, RAM, and CUDA version. The following section, "Software," discusses the software tools and libraries employed, such as Python, PyTorch, Anaconda, and Visual Studio Code, which provided a robust environment for implementing and analyzing the deep learning models. Finally, the "Performance Metrics" section introduces the key metrics used to evaluate the model's performance, including mean average precision (mAP), precision, recall, and intersection over union (IoU). The chapter concludes by laying the foundation for the subsequent chapters, providing a comprehensive understanding of the experimental setup employed throughout the research project.

6.1 Dataset

Preparing the data is a crucial step in object detection as it enables the model to learn and accurately identify the characteristics of objects. Object detection models rely on extensive and diverse datasets during the training phase to capture a wide range of object features and variations.

During the data preparation process, several tasks are typically performed. These tasks may include collecting and curating the dataset, annotating the objects of interest, and ensuring data consistency and quality. Collecting a diverse dataset ensures that the model can learn to detect objects in various scenarios, such as different lighting conditions, viewpoints, and object sizes.

By preparing a comprehensive dataset, the object detection model can learn from a wide variety of examples, enhancing its ability to accurately detect and localize objects in different settings. This data preparation step sets the foundation for training a robust and effective object detection model.

6.1.1 Data preprocessing

A Python script was created to aid in the data preparation process for training. The script utilized a technique known as tiling, which involved dividing the large images os size 6250*5000 into smaller sizes of 500x500 and 250x500 pixels. This method was employed to make the dataset more manageable for training the object detection model. Through the tiling process, the initial set of 1000 large images was transformed into 130,000 small images, with each small image representing a unique section of the original large image. This approach allowed for more targeted analysis and training.

Figure 6.1 illustrates three examples of the tiled images generated by the Python script, showcasing the resulting smaller image sizes. These tiled images served as the foundation for subsequent stages of the experiment, such as annotation and model training.

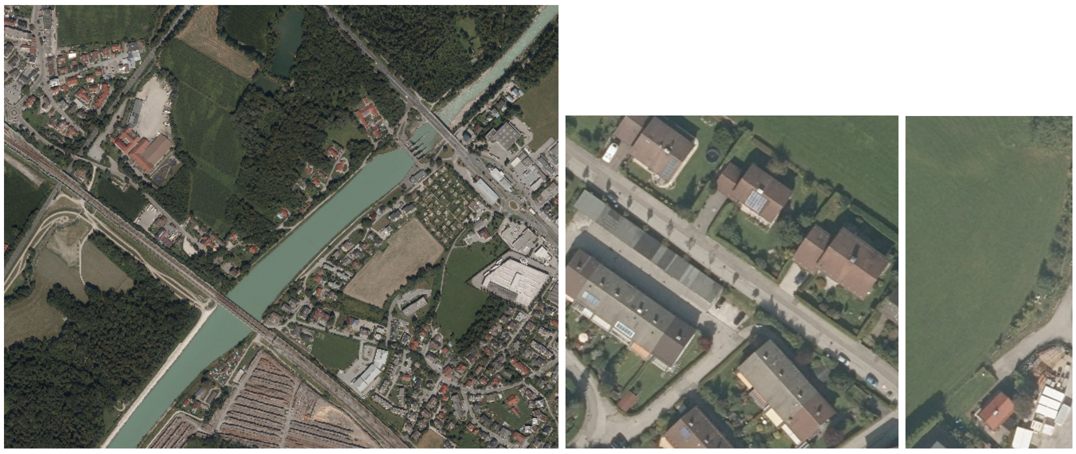


Figure 6.1: Example of images of 3 different sizes.

6.1.2 Data annotation

An object detection model's effectiveness is greatly influenced by the quality of its training data, which should be correctly classified and annotated. Annotation plays a vital role in data preparation, as it involves labeling the objects within the images or providing bounding boxes around them. This annotation process helps the model understand the spatial extent and location of each object within the image. During the training process, the model adjusts a large number of parameters to effectively identify the features of the objects being detected. However, without a carefully annotated dataset, this becomes a challenging task, and the model's accuracy may be compromised. Thus, having a finely annotated dataset is crucial for training a reliable object detection model that includes a diverse set of object classes with precise and consistent annotations. The annotations should cover essential attributes such as accurate object boundaries, class labels, and occlusions.

After collecting the satellite imagery data, the next step was to annotate the data for training the instance segmentation model. For this purpose, the LabelMe tool was used, which is a graphical image annotation tool written in Python [128]. The tool allowed to annotate the small images with class *solar_panel*. Figure 6.2 shows the working of labelme tool to annotate images with class solar_panel.

The annotations were saved in JSON (JavaScript Object Notation) format, which is a simple and lightweight data-interchange format that can be easily read and written by humans, as well as parsed and generated by machines [129]. The annotations contained information about the object's category, location, shape, and size, as depicted in Figure 6.3.

6.1.3 Data formatting

To use the annotated data for object detection, it needs to be converted into a standard format that can be understood by the object detection framework. Popular formats include COCO, PASCAL VOC, and YOLO [130]. In this case, after the annotations were completed, they were converted into the COCO (Common Objects in Context) dataset format as it is used by Mask R-CNN for both training and evaluation.

The COCO dataset format is designed for large-scale object detection, captioning, and segmentation tasks. It includes annotations for object categories, bounding boxes, and segmentation masks. Additionally, the format provides information on image size, file name, and image ID, as shown in figure 6.3 &

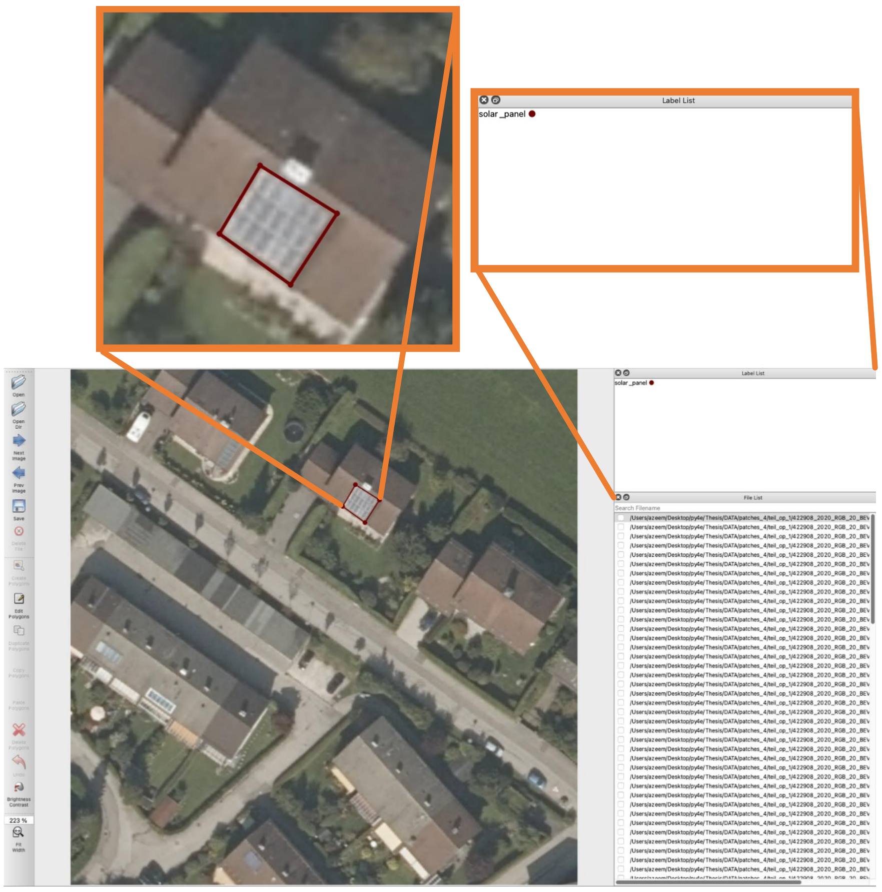


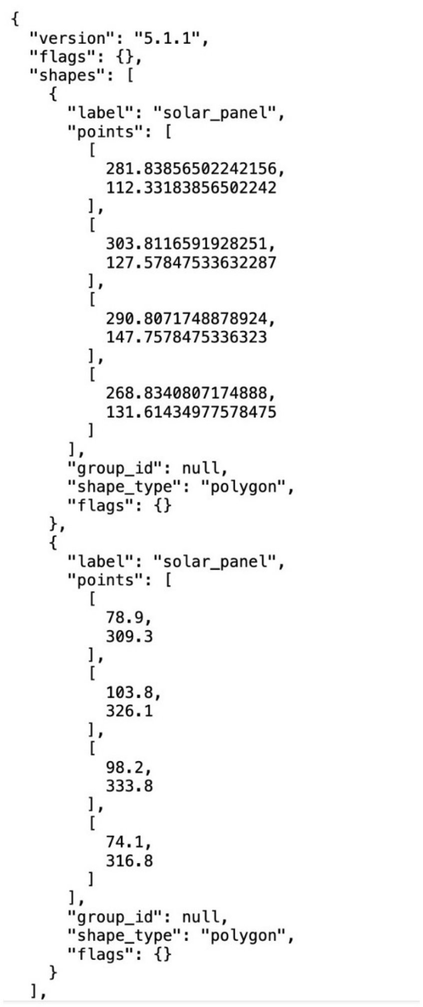
Figure 6.2: Data annotation using labelme tool.

6.4 [131]. The annotated data in the COCO dataset format was then used to train the instance segmentation model.

6.1.4 Data splitting

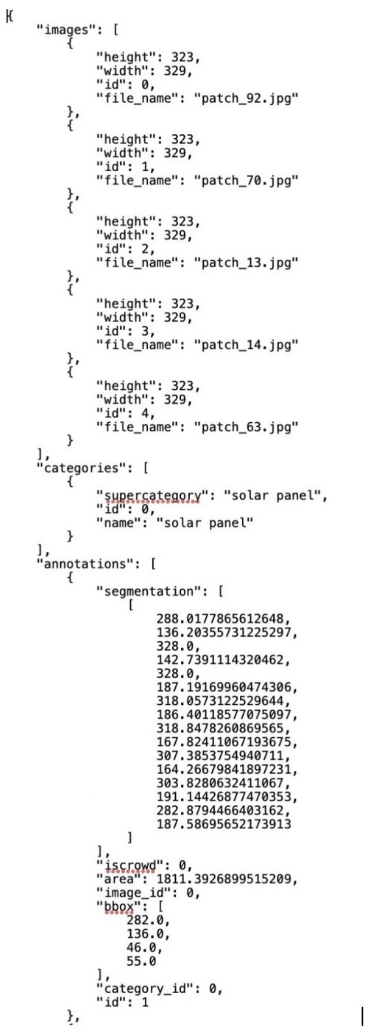
The division of data into distinct sets for training and testing an object detection model is a critical step in dataset preparation. To split the available data into three distinct sets for training, validation, and testing is the main goal of data splitting.

For this project, the data was divided into three distinct sets: training, validation, and testing. The commonly used convention of allocating 70% of the data for training, 15% for validation, and 15% for testing was followed. The model underwent training using the provided training set, enabling it to learn from the data and improve its performance through optimization. The validation set served as a means to fine-tune the model and select the best hyperparameters. Finally, the testing set provided an independent evaluation



```
{
  "version": "5.1.1",
  "flags": {},
  "shapes": [
    {
      "label": "solar_panel",
      "points": [
        [
          281.83856502242156,
          112.33183856502242
        ],
        [
          303.8116591928251,
          127.57847533632287
        ],
        [
          290.8071748878924,
          147.7578475336323
        ],
        [
          268.8340807174888,
          131.61434977578475
        ]
      ],
      "group_id": null,
      "shape_type": "polygon",
      "flags": {}
    },
    {
      "label": "solar_panel",
      "points": [
        [
          78.9,
          309.3
        ],
        [
          103.8,
          326.1
        ],
        [
          98.2,
          333.8
        ],
        [
          74.1,
          316.8
        ]
      ],
      "group_id": null,
      "shape_type": "polygon",
      "flags": {}
    }
  ],
```

Figure 6.3: JSON format



```
{
    "images": [
        {
            "height": 323,
            "width": 329,
            "id": 0,
            "file_name": "patch_92.jpg"
        },
        {
            "height": 323,
            "width": 329,
            "id": 1,
            "file_name": "patch_70.jpg"
        },
        {
            "height": 323,
            "width": 329,
            "id": 2,
            "file_name": "patch_13.jpg"
        },
        {
            "height": 323,
            "width": 329,
            "id": 3,
            "file_name": "patch_14.jpg"
        },
        {
            "height": 323,
            "width": 329,
            "id": 4,
            "file_name": "patch_63.jpg"
        }
    ],
    "categories": [
        {
            "supercategory": "solar panel",
            "id": 0,
            "name": "solar panel"
        }
    ],
    "annotations": [
        {
            "segmentation": [
                [
                    288.0177865612648,
                    136.20355731225297,
                    328.0,
                    142.7391114320462,
                    328.0,
                    187.19169960474306,
                    318.0573122529644,
                    186.40118577075097,
                    318.8478260869565,
                    167.82411067193675,
                    307.3853754940711,
                    164.26679841897231,
                    303.8280632411067,
                    191.14426877470353,
                    282.8794466403162,
                    187.58695652173913
                ]
            ],
            "iscrowd": 0,
            "area": 1811.3926899515209,
            "image_id": 0,
            "bbox": [
                282.0,
                136.0,
                46.0,
                55.0
            ],
            "category_id": 0,
            "id": 1
        },
```

Figure 6.4: COCO format

of the trained model's performance, enabling an unbiased assessment of its effectiveness.

It was ensured that the images in each set contained a diverse range of solar panel instances in various orientations, sizes, and lighting conditions. I have randomly sampled the images for each set to ensure that each set had a representative sample of the entire dataset.

6.2 Hardware

The research project was conducted using the following hardware configuration:

- **Processor**: AMD Ryzen 7 1800X Eight-Core. The AMD Ryzen 7 1800X is a high-performance processor with eight cores, providing ample processing power for running deep learning algorithms and data-intensive tasks.

- **GPU**: NVIDIA GeForce RTX 2080 SUPER with 8 GB VRAM. The NVIDIA GeForce RTX 2080 SUPER is a powerful graphics processing unit that is well-suited for deep learning tasks. Its 8 GB of VRAM allows for efficient training and inference of large neural network models.
- **RAM**: 32 GB. The system includes 32 GB of random-access memory (RAM), providing ample memory for data processing and model training. Sufficient RAM is crucial for handling large datasets and training deep learning models effectively.
- **CUDA**: The research project made use of CUDA version 11.8, which is a parallel computing platform and API model developed by NVIDIA. This was done to fully leverage the capabilities of the NVIDIA GPU for accelerated deep learning computations.

The hardware configuration for this research project was comprised of the AMD Ryzen 7 1800X Eight-Core processor, NVIDIA GeForce RTX 2080 SUPER GPU, 32 GB of RAM, and CUDA version 11.8. This powerful setup was utilized to enable efficient training, inference, and experimentation for the deep learning model in the project.

6.3 Software

The implementation of the research project involved the use of several software tools and libraries. The following software components were utilized:

- **Python**: Python is a widely-used programming language known for its simplicity and readability. Version 3.9.15 was employed in this research to develop and execute the necessary scripts and code.
- **PyTorch**: PyTorch, a popular deep learning framework, was utilized for building and training the deep neural network models. Version 1.13.0+cu117 of PyTorch was specifically used in this project to take advantage of its powerful tensor computation capabilities and automatic differentiation.

- **Anaconda**: Anaconda is a platform that provides a convenient environment for managing and organizing Python packages, especially for scientific computing and data analysis. The specific version 22.11.1 of Anaconda was utilized to create a consistent and reproducible software environment for the project.

- **Visual Studio Code**: VS code, a versatile and lightweight code editor, was used as the primary integrated development environment (IDE) for writing, editing, and debugging the project code. Its user-friendly interface and extensive plugin ecosystem facilitated efficient development workflows.

Additionally, the following libraries were employed:

- **mmdet**: The mmdetection library, version 2.26.0, was used extensively in this research. It provided the necessary tools and functions for implementing various object detection and instance segmentation models, including the Mask R-CNN. The library offered pre-defined network architectures, loss functions, and training pipelines to streamline the development process.

- **TensorBoard**: TensorBoard, a visualization toolkit provided by TensorFlow, was employed for visualizing and analyzing the training progress and performance of the deep learning models. It allowed for the inspection of metrics, model graphs, and other relevant information, aiding in the interpretation and optimization of the training process.

The integration of these software components and libraries provided a robust and efficient environment for conducting the experiments, implementing the Mask R-CNN model, and analyzing the results.

6.4 Performance Metrics

The section aims to describe the performance metric utilized to assess the model's effectiveness, which is mean average precision (mAP). The use of mAP has become common in evaluating segmentation models, and it is widely recognized in object detection and segmentation competitions like PASCAL VOC, COCO, and ImageNet challenges [132].

Before discussing mAP, it's crucial to understand the key metrics that form its foundation. These metrics are fundamental in object detection and segmentation and have specific roles in the calculation of mAP.

- True Positive (TP): In object detection and segmentation, TP refers to the correct detection or segmentation of an object that matches the ground truth. It represents the cases where the model correctly predicts the positive class or identifies an object correctly.
- False Positive (FP): In object detection and segmentation, FP occurs when the model incorrectly predicts the positive class or identifies an object that is not present in the ground truth. It represents an object that was mistakenly detected or segmented when there was no corresponding object in the ground truth.
- True Negative (TN): TN represents the cases where the model truly predicts the negative class or correctly identifies the absence of an object. In object detection and segmentation, it refers to correctly identifying areas without objects as negatives.
- False Negative (FN): In object detection and segmentation, FN occurs when the model fails to predict the positive class or misses an object that is present in the ground truth. It represents the failure to detect or segment an object that exists in the ground truth.

These terms are fundamental in assessing the performance of object detection and segmentation models. They are essential in computing metrics like precision, recall, and F1 score, which help in evaluating the model's ability to accurately detect and segment objects. By analyzing values such as TP, FP, TN, and FN, it is possible to assess the model's strengths and weaknesses in distinguishing between positive and negative instances.

6.4.1 Precision and Recall

In the context of object detection and instance segmentation, precision and recall are fundamental metrics. Precision measures the accuracy of the model's predictions, specifically the proportion of correct positive predictions out of all positive predictions. It is computed as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{6.1}$$

Recall, on the other hand, measures the model's ability to detect relevant instances within the dataset. It is calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{6.2}$$

A high precision value close to 1 indicates that the majority of the predictions made by the model are correct. It reflects the ability of the model to accurately identify and classify objects. On the other hand, a high recall value close to 1 indicates that the model has successfully detected a large portion of the actual objects present in the images. It demonstrates the model's capability to capture the majority of the objects of interest. In summary, a high precision value signifies accurate predictions, while a high recall value signifies comprehensive object detection.

6.4.2 Intersection over Union (IoU)

IoU is a measure that determines the overlap between the predicted bounding box or segmentation mask and the ground truth annotation. It is calculated as the intersection area between the predicted and ground truth regions to the union area of both regions, as shown in figure 6.5. IoU values are between 0 and 1, where 0 denotes a lack of overlap and 1 denotes complete overlap.

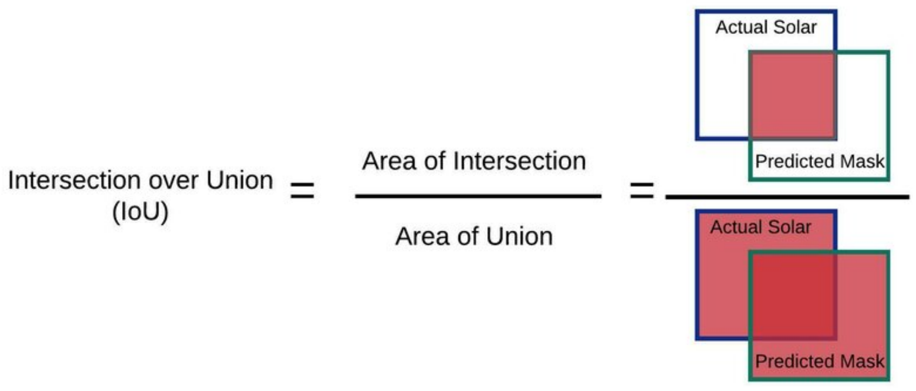


Figure 6.5: Intersection over Union (IoU) [133].

The IoU threshold value allows us to determine whether a prediction falls into the categories of True Positive, False Positive, or False Negative. It serves as a criterion for evaluating the overlap between predicted and ground truth regions.

The threshold used in mAP (mean Average Precision) and AP (Average Precision) calculations is determined by the intersection over union (IoU) of a detection. The threshold is typically applied to the intersection over union (IoU) measurement between the predicted bounding box/segmentation and the ground truth bounding box/segmentation.

After obtaining the TP, FP, and FN values for a specific intersection over union (IoU) threshold, it is possible to compute the precision-recall curve.

6.4.3 Average Precision (AP)

Average Precision (AP) is a metric that provides a comprehensive evaluation of a model's performance in object detection and instance segmentation tasks. The precision and recall values obtained from a precision-recall curve are used to derive it.

The equation for Average Precision (AP) is defined as:

$$AP = \int_0^1 p(r)\, dr \tag{6.3}$$

where: p represents precision, and r represents recall.
The integral is performed over all points on the precision-recall curve. The equation calculates the area under the precision-recall curve, which reflects the average precision across different recall values.

6.4.4 Mean Average Precision (mAP)

The mAP (mean Average Precision) is obtained by calculating the average precision for different classes and then taking the average across them. A higher mAP indicates better performance, with a value of 1 representing perfect accuracy [134].

It's important to note that while the interpretation of AP and mAP may vary in different contexts, in the COCO challenge, AP and mAP refer to the same metric for evaluating object detection models [85].

In summary, mAP (Mean Average Precision) is a performance metric used to evaluate instance segmentation models. The choice of IoU threshold significantly affects precision and recall values, making it important to select an appropriate threshold based on the specific application. mAP is a robust and widely accepted metric that allows for meaningful comparisons and provides insights into a model's performance across different object classes. It calculates the average precision for each class and provides a consolidated evaluation of the model's accuracy in object detection and spatial localization tasks. By considering a range of IoU thresholds, mAP provides a comprehensive assessment that accounts for varying levels of object overlap, resulting in a more reliable evaluation of the model's capabilities.

Chapter 7

Experiment and Results

This chapter presents the training procedure and the evaluation of the Mask R-CNN model for solar panel detection. The chapter begins by describing the training procedure, including the backbone architecture, optimizer settings, learning rate scheduler, and loss function. Following the training procedure, the chapter presents the results obtained from various experiments with different hyperparameters and learning rate schedules. The evaluation metrics, including bbox_mAP_50 and segm_mAP_50, are used to assess the model's performance in object detection and segmentation. The results are summarized in a table, providing an overview of the performance achieved in each experiment.

Overall, this chapter provides a detailed account of the training process and the corresponding results, highlighting the effectiveness of the trained Mask R-CNN model for solar panel detection.

7.1 Training procedure

The Mask R-CNN model trained on the custom dataset (discussed in section 6.1) utilized ResNet50 as the backbone architecture. As the backbone of the Mask R-CNN model, ResNet50 played a crucial role in extracting features from input images. Its deep layers enabled the model to learn hierarchical representations of visual patterns, capturing both low-level and high-level features By focusing on a single class, "solar_panel," the model was optimized to perform accurate object detection and segmentation for solar panels.

Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.02, momentum of 0.9, and weight decay of 0.0001 was utilized. Learning rate was adjusted using a step scheduler, where specific learning rates were applied at different epochs to optimize the model's performance. Learning rate adjustments were made at predefined training steps, and the learning rate was reduced by a factor of 10. For instance, initially, the learning rate was set to 0.01, and after 8 epochs, it was reduced to 0.001. Then, after 11 epochs, it was further reduced to 0.0001. This process of gradually decreasing the learning rate helps the model converge and find the optimal parameter values. The training procedure was performed with different initial learning rates, namely 0.02, 0.01, 0.001, and 0.0001. By exploring different learning rates, the model can effectively navigate the training process and adjust its parameters accordingly to achieve better performance on the custom dataset.

During the training process, the CrossEntropyLoss loss function was utilized to optimize the Mask R-CNN model and enhance its performance. The CrossEntropyLoss loss function measured the disparity between the model's predicted outputs and the ground truth annotations. By optimizing this loss function, the model was able to learn and capture object relationships, locations, and semantic information more effectively, ultimately leading to improved object detection and segmentation capabilities.

The trained Mask R-CNN model was evaluated on the validation dataset at an interval of 1 epoch. Two key metrics, 'bbox' (bounding box) and 'segm' (segmentation), were employed to assess the model's performance. The 'bbox' metric evaluated the accuracy of object detection, while the 'segm' metric measured the model's ability to accurately segment objects. By considering both metrics, a comprehensive evaluation of the model's capabilities in object localization and segmentation was obtained.

The model can be used for inference on new data after being trained. The model processes the input image through the network, performs region proposals, classifies objects, refines bounding boxes, and generates pixel-wise segmentation masks to provide accurate object detection and segmentation results.

In summary, by customizing the Mask R-CNN model to focus on the solar panel class and training it with the custom dataset, the model was optimized for accurate solar panel detection and segmentation. The training process involved adjusting the learning rate, utilizing the CrossEntropyLoss loss function, and evaluating the model's performance. The resulting model could then be used for precise inference on new images, providing valuable insights for solar panel detection applications. In the next section, the results are discussed.

7.2 Results

In this section, the results obtained from the experimental analysis of the designed, trained, and evaluated Mask R-CNN model on the custom dataset for rooftop solar PV system detection and segmentation in satellite imagery are presented. The primary objective was to assess the model's performance in accurately identifying and segmenting rooftop solar PV systems. To evaluate the model, mean average precision (mAP) metrics were employed for both bounding box detection and instance segmentation tasks.

7.2.1 Quantitative results

Specific details of the quantitative results are presented in Table 7.1. This table provides a comprehensive overview of the different training stages employed in the experimental analysis. It includes the information like number of iterations, epochs, initial learning rates, and learning rate scheduler for each training stage. The evaluation metrics, bbox_mAP_50 and segm_mAP_50, provide insights into the model's performance for bounding box detection and instance segmentation, respectively.

Number of iterations	**Epochs**	**Initial LR**	**LR Schedule**	**bbox_mAP_50**	**segm_mAP_50**
1	25	0.02	Linear 8, 11	0.769	0.767
2	10	0.01	Linear 8, 11	0.799	0.793
3	10	0.001	Linear 8, 11	0.792	0.774
4	5	0.001	Linear 400, 500	**0.803**	**0.781**
5	44	0.0001	Linear 400, 500	0.785	0.766
6	9	0.001	CosineAnnealing	0.795	0.761
7	12	0.01	CosineAnnealing	0.796	0.798
8	23	0.02	CosineAnnealing	0.792	0.794

Table 7.1: Training stages and evaluation metrics

Among the various training experiment evaluated, the best result in terms of bounding box detection precision was obtained after 4 iterations with an initial learning rate of 0.001 and 5 epochs. This outcome highlights the success of the Mask R-CNN model in accurately identifying regions of interest and precisely locating rooftop solar PV systems in satellite imagery. The achieved bbox_mAP_50 score of 0.803 indicates the model's effectiveness in detecting the exact boundaries of the PV systems, which is crucial for subsequent analysis and decision-making.

However, the corresponding segm_mAP_50 score for instance segmentation in the same stage was 0.781. Although this score is slightly lower than the bbox_mAP_50 score, it still indicates a reasonably good performance in segmenting the individual PV systems. Instance segmentation is a more challenging task as it involves accurately delineating the boundaries of each instance within an image.

The following graphs, obtained from experiment number 4, provide visual representations of the training process and performance of the Mask R-CNN model on the custom dataset. The two curves shown in Figure 7.1 provide insights into the model's performance in terms of bounding box detection and instance segmentation, respectively. The bbox_mAP_50 curve represents the mean average precision (mAP) for bounding box detection at a threshold of 50%. It evaluates the accuracy of the model in locating objects within bounding boxes. A higher value on this curve indicates better performance in precisely detecting object boundaries. On the other hand, the segm_mAP_50 curve measures the mAP for instance segmentation at a 50% threshold. This curve assesses the model's ability to accurately segment individual instances within the detected bounding boxes. A higher value on this curve indicates superior performance in accurately outlining the boundaries of each object.

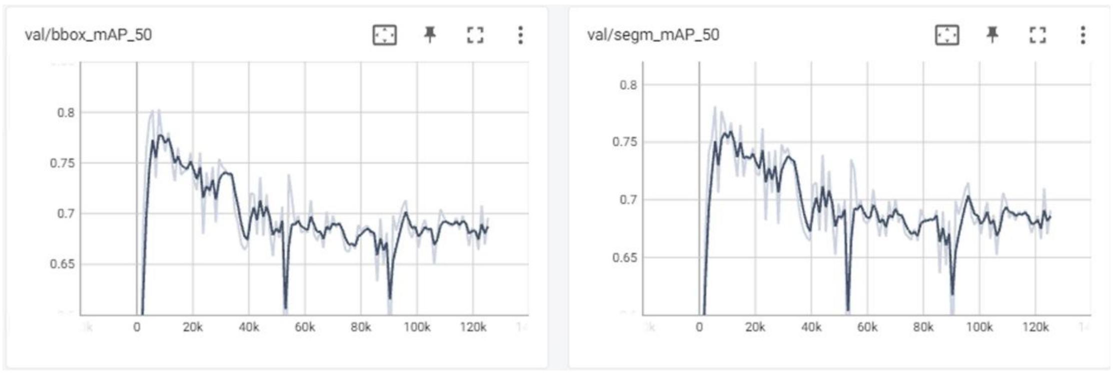


Figure 7.1: Bounding box detection mAP (bbox_mAP_50) and Instance segmentation mAP Curve at 50% threshold (segm_mAP_50).

Figure 7.2 represents the loss curves in the Mask R-CNN training process, including overall loss, bounding box detection loss, classification loss, and mask loss. These curves provide valuable insights into the training process of the Mask R-CNN model. It consists of four distinct curves that represent different aspects of the model’s performance.

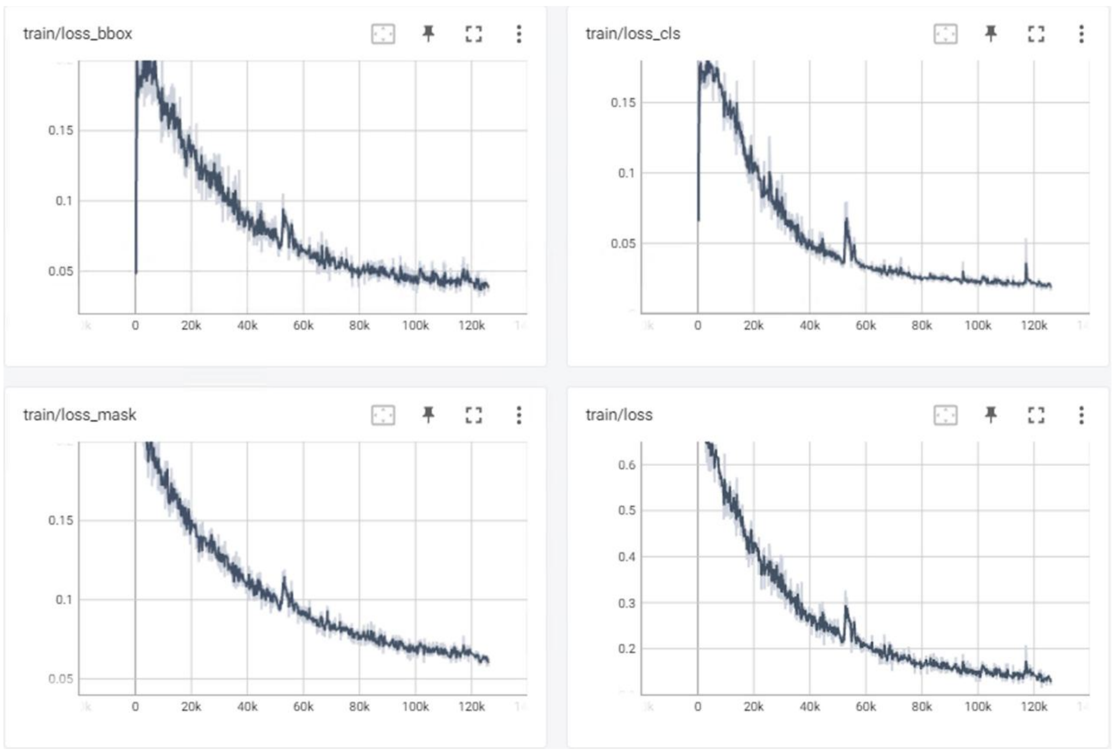


Figure 7.2: Loss curves in the Mask R-CNN training process, including bounding box detection Loss (loss_bbox), classification loss (loss_cls), mask loss (loss_mask), and overall loss (loss).

The first curve represents the bounding box loss, which measures the discrepancy between the predicted bounding box coordinates and the ground truth annotations. This curve provides information about the model’s ability to accurately localize objects and estimate their spatial boundaries. A decreasing bounding box loss curve indicates that the model is learning to precisely predict object locations.

The second curve represents the classification loss, which measures the discrepancy between the predicted class labels and the ground truth annotations. This curve reflects the model’s performance in accurately classifying objects into different categories. A decreasing classification loss curve indicates that the model is improving its object recognition capabilities.

The third curve represents the mask loss, which measures the discrepancy between the predicted instance segmentation masks and the ground truth masks. This curve evaluates the model’s ability to accurately segment and outline objects within an image. A decreasing mask loss curve signifies that the model is learning to generate precise instance segmentation masks.

The fourth curve represents the overall loss, which encompasses all components of the loss function. It reflects the cumulative loss value calculated during each training iteration. A decreasing trend in the overall loss curve indicates that the model is gradually improving its predictions.

7.2.2 Qualitative results

The qualitative analysis of the Mask R-CNN model provides valuable insights into its performance, in addition to the quantitative evaluations. Figure 7.3 showcases two examples of input images, accompanied by their respective ground truth annotations and predicted outputs obtained using Mask R-CNN. The purpose of these examples is to visually assess the model's effectiveness in detecting and segmenting rooftop solar PV systems. By comparing the ground truth annotations with the predicted outputs, one can observe the model's ability to accurately identify and delineate the regions of interest. The predicted bounding boxes and segmented masks provide a visual representation of the model's understanding of the rooftop solar PV systems' spatial boundaries and locations within the satellite imagery. Additionally, the inference time for processing a full-size image with dimensions of 6250x5000 pixels was measured to be 120.079 seconds. This information provides insights into the computational efficiency of the model and its ability to handle large-scale satellite imagery in real-world scenarios. This qualitative analysis complements the quantitative evaluation by providing a more intuitive and visual perspective on the model's performance. Overall, the qualitative assessment aids in gaining a comprehensive understanding of the Mask R-CNN model's performance on the custom dataset.

In addition to generating segmented masks and bounding boxes, an additional enhancement has been made to the code. It now generates a CSV file containing the coordinates of the bounding boxes and segmented mask area (in m^2) for each detected PV system in the input images as shown in Figure 7.4. These coordinates are referenced in the EPSG:31258 CRS system, specifically the MGI / Austria M31 projection. This modification provides a convenient and structured way to store and utilize the bounding box information for further analysis or downstream tasks. The CSV file includes precise coordinates of the top-left corner, bottom-right corner, and center of each bounding box and the

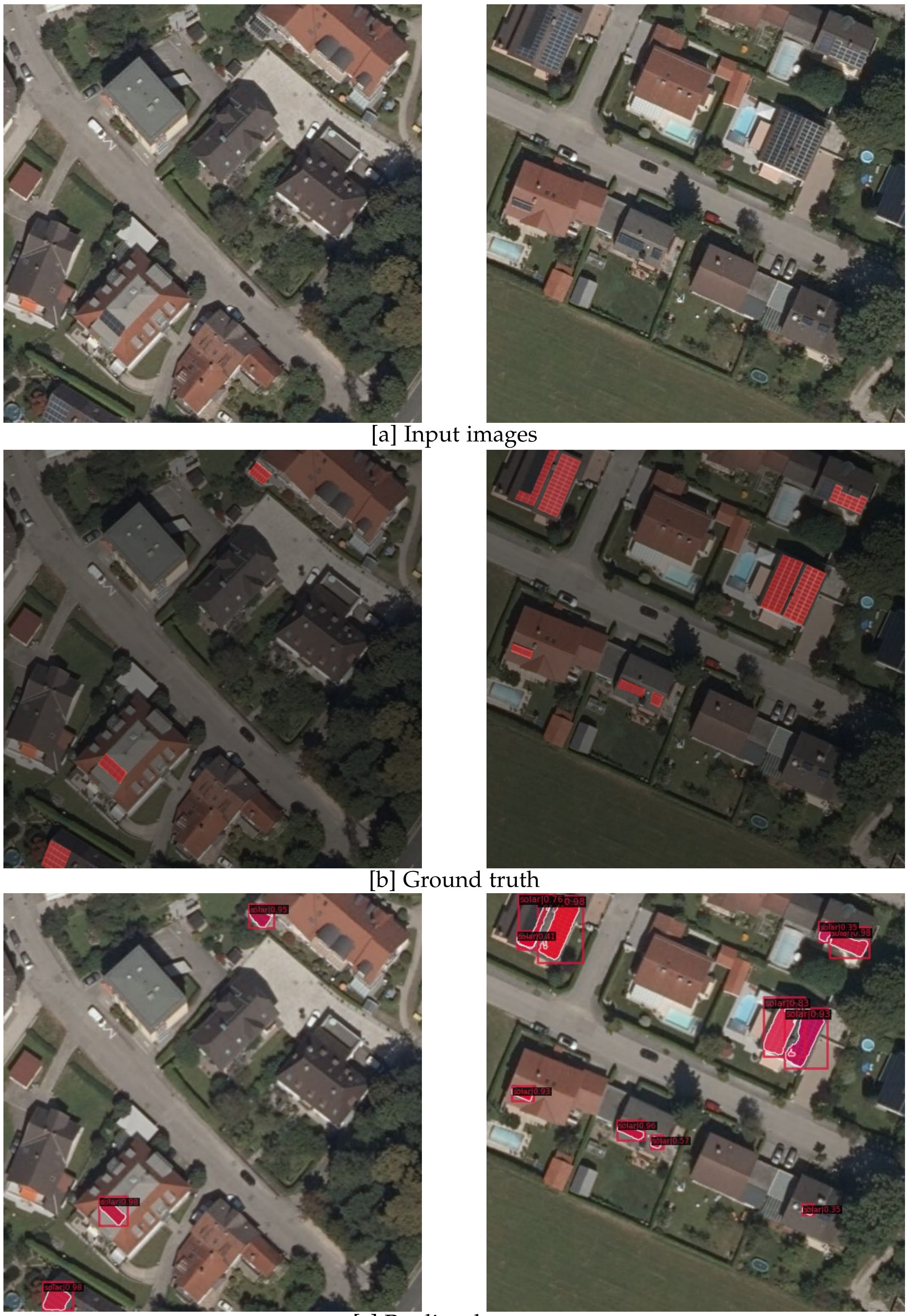
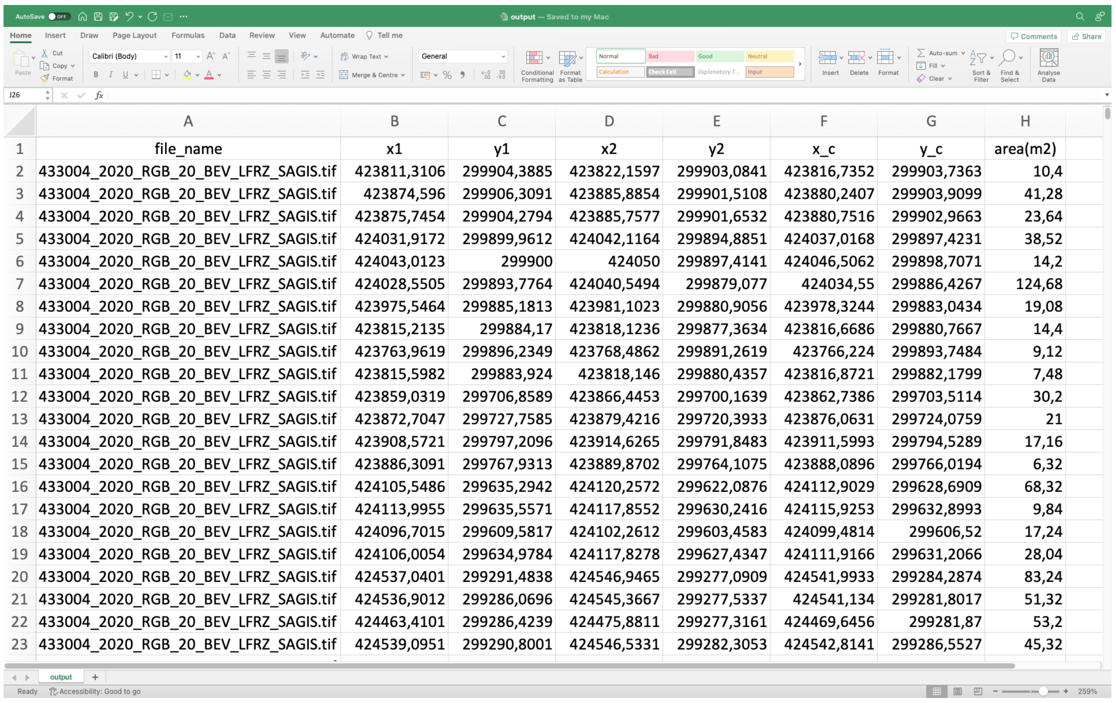


Figure 7.3: Two examples of input images [a] with their ground truth labels [b] and the predicted output [c] of the experiment.



	A	B	C	D	E	F	G	H
1	file_name	x1	y1	x2	y2	x_c	y_c	area(m2)
2	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423811,3106	299904,3885	423822,1597	299903,0841	423816,7352	299903,7363	10,4
3	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423874,596	299906,3091	423885,8854	299901,5108	423880,2407	299903,9099	41,28
4	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423875,7454	299904,2794	423885,7577	299901,6532	423880,7516	299902,9663	23,64
5	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424031,9172	299899,9612	424042,1164	299894,8851	424037,0168	299897,4231	38,52
6	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424043,0123	299900	424050	299897,4141	424046,5062	299898,7071	14,2
7	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424028,5505	299893,7764	424040,5494	299879,077	424034,55	299886,4267	124,68
8	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423975,5464	299885,1813	423981,1023	299880,9056	423978,3244	299883,0434	19,08
9	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423815,2135	299884,17	423818,1236	299877,3634	423816,6686	299880,7667	14,4
10	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423763,9619	299896,2349	423768,4862	299891,2619	423766,224	299893,7484	9,12
11	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423815,5982	299883,924	423818,146	299880,4357	423816,8721	299882,1799	7,48
12	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423859,0319	299706,8589	423866,4453	299700,1639	423862,7386	299703,5114	30,2
13	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423872,7047	299727,7585	423879,4216	299720,3933	423876,0631	299724,0759	21
14	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423908,5721	299797,2096	423914,6265	299791,8483	423911,5993	299794,5289	17,16
15	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	423886,3091	299767,9313	423889,8702	299764,1075	423888,0896	299766,0194	6,32
16	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424105,5486	299635,2942	424120,2572	299622,0876	424112,9029	299628,6909	68,32
17	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424113,9955	299635,5571	424117,8552	299630,2416	424115,9253	299632,8993	9,84
18	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424096,7015	299609,5817	424102,2612	299603,4583	424099,4814	299606,52	17,24
19	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424106,0054	299634,9784	424117,8278	299627,4347	424111,9166	299631,2066	28,04
20	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424537,0401	299291,4838	424546,9465	299277,0909	424541,9933	299284,2874	83,24
21	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424536,9012	299286,0696	424545,3667	299277,5337	424541,134	299281,8017	51,32
22	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424463,4101	299286,4239	424475,8811	299277,3161	424469,6456	299281,87	53,2
23	433004_2020_RGB_20_BEV_LFRZ_SAGIS.tif	424539,0951	299290,8001	424546,5331	299282,3053	424542,8141	299286,5527	45,32

Figure 7.4: Example visualization of the generated CSV file containing bounding box coordinates and segmented mask area.

area of each mask.

This information can be easily accessed, processed, and integrated into various applications or workflows, enabling efficient object localization and subsequent actions based on the detected objects. The inclusion of the CSV output expands the utility and versatility of the Mask R-CNN model by providing detailed bounding box coordinates in a readily accessible format.

7.2.3 Challenges

Implementing Mask R-CNN for automatic detection of rooftop solar PVs involves several challenges, including:

1. Data annotation: Annotating a large dataset of 130,000 images for labeling rooftop solar panels posed a significant challenge due to variations in appearance, texture, size among the panels, and collusion caused by surrounding objects and shadows, difficulty in detecting small panels due to low resolution of satellite imagery. In satellite imagery, certain objects may appear similar or have low visual contrast, making it difficult to accurately label them. This task required substantial time, effort, and expertise.

2. Model complexity and resource requirements: Implementing and training the Mask R-CNN model on large-scale datasets was computationally expensive and time-consuming. It required substantial computational resources, such as GPUs or TPUs, to handle the model's complexity and training demands.

3. Fine-tuning for best results: Achieving optimal performance in bounding box detection and mask segmentation for rooftop solar PVs required fine-tuning the Mask R-CNN model. Finding the right hyperparameters and optimizing the model's configuration for accurate detection and segmentation of solar panels was a challenge.

7.2.4 Applications

The potential applications of this information can be summarized as follows:

- **Strategic planning and design**: The knowledge of spatial distribution and density of rooftop solar PVs can assist utilities and grid operators in strategically planning and designing distribution networks, optimizing power flow, and enhancing overall system resilience.

- **Load forecasting and demand management**: Accurate detection of rooftop solar PV systems enables utilities to estimate solar energy generation and its impact on the distribution grid. This information can be leveraged for proactive load balancing, peak shaving, and demand response programs, ensuring efficient energy supply and demand management.

- **Spatial planning**: The information about existing rooftop solar PV installations can inform spatial planning efforts, guiding the identification of suitable areas for future solar installations and maximizing renewable energy potential.

- **Tailored subsidy schemes**: The knowledge of rooftop solar PV installations can assist in designing targeted subsidy schemes, allocating financial incentives or grants to areas with low solar PV penetration, and encouraging wider adoption of solar power.

- **Research institutions**: Research institutions can utilize the information for various studies and analyses related to renewable energy, urban planning, climate change, and sustainability. It provides valuable data for research on energy transition, solar energy efficiency, and the impact of solar installations on the environment and society.

- **Environmental organizations**: Environmental organizations can utilize the information to assess the environmental benefits of solar energy and advocate for renewable energy policies. It helps them monitor the growth of solar installations, track progress toward climate goals, and raise awareness about the importance of clean energy sources.

The beneficiaries of this information include government, utilities, energy planners, policymakers, communities, institutions, and individuals interested in promoting renewable energy and achieving sustainability goals. By leveraging the insights gained from the study, these stakeholders can make informed decisions, implement effective strategies, and drive the transition to a cleaner and more resilient energy system. The applications derived from the data on distributed solar installations, including spatial planning, energy community planning, smart grid operation, and tailored subsidy schemes, are instrumental in helping Salzburg and Austria achieve their goals of transitioning to a sustainable and renewable energy system. Through spatial planning and technology, the optimal placement of solar installations can be determined, the potential of renewable energy can be maximized and it can support the objective of Salzburg to increase the share of locally produced energy from renewables. Energy community planning, facilitated by the data on distributed solar installations aligns with Austria's goal of fostering energy autonomy and increasing the adoption of renewables. Integrating distributed solar power systems into the smart grid infrastructure enhances grid efficiency, balancing the supply and demand of renewable energy and contributing to Salzburg's commitment to generating all electricity from renewable sources by 2030. Lastly, tailored subsidy schemes based on the data on rooftop solar installations incentivize wider adoption of solar power, especially in areas with low solar penetration, furthering Salzburg's renewable energy targets. By leveraging these applications, Salzburg and Austria can make significant strides toward a sustainable and resilient energy future.

Chapter 8

Conclusion and Future Works

8.1 Conclusion

In conclusion, this project successfully achieved its objectives of detecting and segmenting rooftop solar PVs in satellite imagery using the Mask R-CNN model. The mean average precision (mAP) score for detection achieved a value of 0.803, indicating the model's capability to accurately identify and localize rooftop solar PV systems in satellite imagery. Additionally, the mAP score for segmentation reached 0.781, demonstrating the model's effectiveness in accurately delineating the boundaries of individual solar panels within the detected regions. These high mAP scores validate the model's performance and its potential for automated detection and segmentation tasks.

The methodology employed in this project involved a thorough literature review and comparative analysis to identify the most suitable deep learning model for rooftop solar PV detection and segmentation in satellite imagery. The selected model, Mask R-CNN, was then trained on a custom dataset to automate the process. This methodology proved successful in achieving higher accuracy and demonstrated the promise of satellite imagery and deep learning in accurately analyzing rooftop solar installations.

The information obtained from this project on distributed solar installations holds great significance for various stakeholders including utilities, grid operators, energy planners, policymakers, communities, and individuals interested in promoting renewable energy and sustainability. The insights gained from this study enable informed decision-making, effective strategy implementation, and progress toward a cleaner and more resilient energy system.

Moreover, the applications derived from this research could be used to speed up the transition to a sustainable and resilient energy future in Salzburg, Austria, and beyond. By leveraging these applications, significant strides can be made in achieving sustainable energy systems, enhancing grid resilience, and promoting the adoption of solar power.

In summary, this project contributes to the field of deep learning in satellite imagery analysis, specifically focusing on rooftop solar PV detection and segmentation. The findings demonstrate the effectiveness of the Mask R-CNN model and its potential applications in various domains. By addressing research questions, providing valuable insights, and offering practical solutions, this project paves the way for further advancements in automated detection and analysis of solar panel installations using satellite imagery.

8.2 Future work

In future work, there are several areas that can be explored in a sequential manner to enhance the performance and applicability of solar panel detection using Mask R-CNN in satellite imagery:

1. Analysis: To analyze the spatial distribution, size, and geolocation of the detected rooftop solar PV systems.
2. Optimizer selection: Experiment with different optimizers, such as Adam, to evaluate their impact on the model's training dynamics and detection accuracy.
3. Backbone architecture variation: Compare different backbone architectures, such as ResNet101, to identify the most suitable one for solar panel detection in terms of accuracy and speed.
4. Whole image analysis: Investigate methods that allow the model to operate on the entire image without tiling, improving detection accuracy for objects spanning multiple tiles.
5. Generalization to other cities: Expand the dataset to include satellite imagery from various cities or regions to enhance the model's ability to detect solar panels in diverse contexts.
6. Adverse weather conditions: Collect and integrate satellite imagery captured under different weather conditions, such as cloudy or overcast skies, to improve the model's robustness in adverse weather scenarios.

7. Hybrid approaches: Explore hybrid approaches that combine Mask R-CNN with other computer vision techniques or data sources, such as LiDAR data, to improve the accuracy and reliability of solar panel detection.

By following this sequential approach, the performance, efficiency, generalization capabilities, and real-world applicability of solar panel detection using Mask R-CNN can be systematically improved. These advancements contribute to the field of renewable energy monitoring and facilitate the wider adoption of solar power technology.

List of Figures

List of Tables

Bibliography

[1] *RSA Research Studio Austria*. https://www.researchstudio.at/. Accessed March 4, 2023.

[2] IEA. *World Energy Balances: Overview*. https://www.iea.org/reports/world-energy-balances-overview. License: CC BY 4.0. 2021.

[3] H. Ritchie, M. Roser, and P. Rosado. "Energy". In: *Our World in Data* (2022). https://ourworldindata.org/e

[4] U. Nations. *2018 Revision of World Urbanization Prospects*. https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html. Accessed on May 2, 2023. 2018.

[5] M. Victoria, N. Haegel, I. M. Peters, R. Sinton, A. Jäger-Waldau, C. Cañ, C. Breyer, M. Stocks, A. Blakers, I. Kaizuka, K. Komoto, and A. Smets. "Solar photovoltaics is ready to power a sustainable future". In: *Joule* 5 (Mar. 2021). DOI: 10.1016/j.joule.2021.03.005.

[6] R. Hiremath, S. Shikha, and N. Ravindranath. "Decentralized energy planning; modeling and application-a review". In: *Renewable and Sustainable Energy Reviews* 11 (2007), pp. 729–752. DOI: 10.1016/j.rser.2005.07.005.

[7] M. Wolsink. "Distributed energy systems as common goods: Socio-political acceptance of renewables in intelligent microgrids". In: *Renewable and Sustainable Energy Reviews* 127 (July 2020), p. 109841. DOI: 10.1016/j.rser.2020.109841.

[8] Q. Li, Y. Feng, Y. Leng, and D. Chen. "SolarFinder: Automatic Detection of Solar Photovoltaic Arrays". In: *2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 2020, pp. 193–204. DOI: 10.1109/IPSN48710.2020.00024.

[9] K. Bradbury, R. Saboo, J. Malof, T. Johnson, A. Devarajan, W. Zhang, L. Collins, and R. Newell. "Distributed Solar Photovoltaic Array Location and Extent Data Set for Remote Sensing Object Identification". In: (May 2016). DOI: 10.6084/m9.figshare.3385780.v1. URL: https://figshare.com/articles/dataset/Distributed_Solar_Photovoltaic_Array_Location_and_Extent_Data_Set_for_Remote_Sensing_Object_Identification/3385780.

[10] J. M. Malof, K. Bradbury, L. M. Collins, and R. G. Newell. "Automatic detection of solar photovoltaic arrays in high-resolution aerial imagery". In: *Applied Energy* 183 (Dec. 2016), pp. 229–240. DOI: 10.1016/j.apenergy.2016.08.191. URL: https://doi.org/10.1016%2Fj.apenergy.2016.08.191.

[11] A. Bazmi and G. Zahedi. "Sustainable energy systems: role of optimization modeling techniques in power generation and supply - a review". In: *Renewable and Sustainable Energy Reviews* 15 (2011), pp. 3480–3500. DOI: 10.1016/j.rser.2011.05.003.

[12] A. Mirakyan and R. De Guio. “Integrated energy planning in cities and territories: a review of methods and tools”. In: *Renewable and Sustainable Energy Reviews* 22 (2013), pp. 289–297. DOI: 10.1016/j.rser.2013.01.033.

[13] Z. Huang, H. Yu, Z. Peng, and M. Zhao. “Methods and tools for community energy planning: a review”. In: *Renewable and Sustainable Energy Reviews* 42 (2015), pp. 1335–1348. DOI: 10.1016/j.rser.2014.11.042.

[14] IEA. *Energy Technology Perspectives 2020 - Special Report on Clean Energy Innovation*. Paris: OECD Publishing, 2020. DOI: 10.1787/ab43a9a5-en.

[15] I. R. E. Agency. *Renewable energy targets in 2022: A guide to design*. Abu Dhabi, 2022. URL: https://mc-cd8320d4-36a1-40ac-83cc-3389-cdn-endpoint.azureedge.net/-/media/Files/IRENA/Agency/Publication/2022/Nov/IRENA_RE_targets_2022.

[16] *HOW IS SALZBURG USING RENEWABLE ENERGY TO CHANGE THE FUTURE?* Website. Mayors of Europe, Mar. 2022. URL: https://mayorsofeurope.eu/top-stories/how-is-salzburg-using-renewable-energy-to-change-the-future/.

[17] J. Deng, X. Xuan, W. Wang, Z. Li, H. Yao, and Z. Wang. “A review of research on object detection based on deep learning”. In: *Journal of Physics: Conference Series* 1684.1 (2020), p. 012028. DOI: 10.1088/1742-6596/1684/1/012028.

[18] R. Sharma. “Object Detection in Dense Volume Data”. In: (2022).

[19] C. Murthy, M. Hashmi, N. Bokde, and Z. Geem. “Investigations of Object Detection in Images/Videos Using Various Deep Learning Techniques and Embedded Platforms—A Comprehensive Review”. In: *Applied Sciences* 10.9 (2020), p. 3280. DOI: 10.3390/app10093280.

[20] J. Heaton. “Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning”. In: *Genetic Programming and Evolvable Machines* 19.1 (2017), pp. 1–3.

[21] D. G. Lowe. “Object Recognition from Local Scale-Invariant Features”. In: *The Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2. 1999, pp. 1150–1157. DOI: 10.1109/ICCV.1999.790410.

[22] P. Viola and M. J. Jones. “Robust real-time face detection”. In: *International Journal of Computer Vision* 57.2 (2004), pp. 137–154. DOI: 10.1023/B:VISI.0000013087.49260.fb.

[23] P. Viola and M. Jones. “Rapid object detection using a boosted cascade of simple features”. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 1. 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517.

[24] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[25] G. LoweDavid. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* (2004).

[26] P. Felzenszwalb, D. Mcallester, and D. Ramanan. “A Discriminatively Trained, Multiscale, Deformable Part Model”. In: vol. 8: June 2008. DOI: 10.1109/CVPR.2008.4587597.

[27] P. F. Felzenszwalb, R. B. Girshick, and D. A. McAllester. “Cascade object detection with deformable part models”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), pp. 2241–2248.

[28] T. Malisiewicz, A. Gupta, and A. A. Efros. “Ensemble of exemplar-SVMs for object detection and beyond”. In: *2011 International Conference on Computer Vision*. 2011, pp. 89–96. DOI: 10.1109/ICCV.2011.6126229.

[29] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 779–788.

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. “SSD: Single Shot MultiBox Detector”. In: To appear. 2016. URL: http://arxiv.org/abs/1512.02325.

[31] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. “Focal Loss for Dense Object Detection”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[32] R. Girshick, J. Donahue, T. Darrell, and J. Malik. “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81.

[33] R. Girshick. “Fast R-CNN”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[34] S. Ren, K. He, R. Girshick, and J. Sun. “Faster rcnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[35] K. He, G. Gkioxari, P. Dollár, and R. Girshick. “Mask R-CNN”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

[36] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. “Object Detection with Discriminatively Trained Part-Based Models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1627–1645. DOI: 10.1109/TPAMI.2009.167.

[37] X. Ren and D. Ramanan. “Histograms of Sparse Codes for Object Detection”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2013.

[38] K. He, X. Zhang, S. Ren, and J. Sun. “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1904–1916. DOI: 10.1109/TPAMI.2015.2389824.

[39] R. B. Girshick. "Fast R-CNN". In: *CoRR* abs/1504.08083 (2015). arXiv: 1504.08083. URL: http://arxiv.org/abs/1504.08083.

[40] D. S. Central. *Improving Real-Time Object Detection with YOLO*. https://www.datasciencecentral.com/improving-real-time-object-detection-with-yolo/. Posted by LubaBelokon. Oct. 2017.

[41] J. Redmon and A. Farhadi. "YOLOv3: An Incremental Improvement". In: *CoRR* abs/1804.02767 (2018). arXiv: 1804.02767. URL: http://arxiv.org/abs/1804.02767.

[42] A. Bochkovskiy, C. Wang, and H. M. Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: *CoRR* abs/2004.10934 (2020). arXiv: 2004.10934. URL: https://arxiv.org/abs/2004.10934.

[43] J. Redmon and A. Farhadi. "YOLO9000: Better, Faster, Stronger". In: *CoRR* abs/1612.08242 (2016). arXiv: 1612.08242. URL: http://arxiv.org/abs/1612.08242.

[44] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao. *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. 2022. arXiv: 2207.02696 [cs.CV].

[45] S.-H. Tsang. *Review: SSD (Single Shot Detector) — Object Detection*. https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11. Nov. 2018.

[46] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye. *Object Detection in 20 Years: A Survey*. 2023. arXiv: 1905.05055 [cs.CV].

[47] C. B. Murthy, M. F. Hashmi, N. D. Bokde, and Z. W. Geem. "Investigations of Object Detection in Images/Videos Using Various Deep Learning Techniques and Embedded Platforms—A Comprehensive Review". In: *Applied Sciences* 10.9 (May 2020), p. 3280. ISSN: 2076-3417. DOI: 10.3390/app10093280. URL: http://dx.doi.org/10.3390/app10093280.

[48] S. Humbarwadi. *Object Detection with RetinaNet*. https://keras.io/examples/vision/retinanet/. Last modified on July 14, 2020. May 2020.

[49] A. Z. Zhu, V. Casser, R. Mahjourian, H. Kretzschmar, and S. Pirk. *Instance Segmentation with Cross-Modal Consistency*. 2022. arXiv: 2210.08113 [cs.CV].

[50] R. Tedrake. *Robotic Manipulation. Perception, Planning, and Control*. 2022. URL: http://manipulation.mit.edu.

[51] W. Gaihua, L. Jinheng, C. Lei, D. Yingying, and Z. Tianlun. "Instance segmentation convolutional neural network based on multi-scale attention mechanism". In: *PLoS ONE* 17.1 (2022), e0263134. DOI: 10.1371/journal.pone.0263134.

[52] M. Bai and R. Urtasun. "Deep Watershed Transform for Instance Segmentation". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 2858–2866.

[53] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. “Instancecut: from edges to instances with multicut”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5008–5017.

[54] P. H. O. Pinheiro, R. Collobert, and P. Dollár. “Learning to Segment Object Candidates”. In: *CoRR* abs/1506.06204 (2015). arXiv: 1506.06204. URL: http://arxiv.org/abs/1506.06204.

[55] J. Dai, Y. Li, K. He, and J. Sun. “R-fcn: Object detection via region-based fully convolutional networks”. In: *Advances in neural information processing systems* 29 (2016).

[56] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. “Instance-Sensitive Fully Convolutional Networks”. In: *European Conference on Computer Vision*. 2016.

[57] D. Bhatt, C. Patel, H. Talsania, J. Patel, R. Vaghela, S. Pandya, K. Modi, and H. Ghayvat. “CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope”. In: *Electronics* 10.20 (Oct. 2021), p. 2470. ISSN: 2079-9292. DOI: 10.3390/electronics10202470. URL: http://dx.doi.org/10.3390/electronics10202470.

[58] A. Khan, A. Sohail, U. Zahoora, and et al. “A survey of the recent architectures of deep convolutional neural networks”. In: *Artificial Intelligence Review* 53 (2020), pp. 5455–5516. DOI: 10.1007/s10462-020-09825-6.

[59] C. C. Chatterjee. “Basics of the Classic CNN: How a classic CNN (Convolutional Neural Network) work?” In: *Towards Data Science* (July 2019).

[60] M. Mishra. *Convolutional Neural Networks, Explained*. Towards Data Science. Aug. 2020. URL: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.

[61] S. Albawi, T. A. Mohammed, and S. Al-Zawi. “Understanding of a convolutional neural network”. In: *2017 International Conference on Engineering and Technology (ICET)*. Antalya, Turkey, Dec. 2017, pp. 1–6. DOI: 10.1109/ICEngTechnol.2017.8308186.

[62] D. Podareanu, V. Codreanu, S. Aigner, C. Leeuwen, and V. Weinberg. *Best Practice Guide - Deep Learning*. Feb. 2019. DOI: 10.13140/RG.2.2.31564.05769.

[63] T. Guo, J. Dong, H. Li, and Y. Gao. “Simple convolutional neural network on image classification”. In: *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*. 2017, pp. 721–724. DOI: 10.1109/ICBDA.2017.8078730.

[64] R. Yamashita, M. Nishio, R. Do, and et al. “Convolutional neural networks: an overview and application in radiology”. In: *Insights into Imaging* 9 (2018), pp. 611–629. DOI: 10.1007/s13244-018-0639-9.

[65] G. Lin and W. Shen. “Research on convolutional neural network based on improved Relu piecewise activation function”. In: *Procedia Computer Science* 131 (Jan. 2018), pp. 977–984. DOI: 10.1016/j.procs.2018.04.239.

[66] N. Chintalapudi, G. Battineni, M. A. Hossain, and F. Amenta. "Cascaded Deep Learning Frameworks in Contribution to the Detection of Parkinson's Disease". In: *Bioengineering* 9.3 (2022), p. 116.

[67] N.-u. Rehman, M. S. Zia, T. Meraj, H. T. Rauf, R. Damaševičius, A. M. El-Sherbeeny, and M. A. El-Meligy. "A self-activated cnn approach for multi-class chest-related COVID-19 detection". In: *Applied Sciences* 11.19 (2021), p. 9023.

[68] D. Liu. *A Practical Guide to ReLU*. Blog post. 2017. URL: `https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7` (visited on 12/17/2019).

[69] J. McDermott. *Convolutional Neural Networks — Image Classification w. Keras*. Accessed Date: March 31, 2023. URL: `https://www.learndatasci.com/tutorials/convolutional-neural-networks-image-classification/`.

[70] T. Guo, J. Dong, H. Li, and Y. Gao. "Simple convolutional neural network on image classification". In: *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*. 2017, pp. 721–724. DOI: `10.1109/ICBDA.2017.8078730`.

[71] R. Nirthika, S. Manivannan, A. Ramanan, and R. Wang. "Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study". In: *Neural Computing and Applications* 34.7 (2022). Epub ahead of print, Feb 1, 2022, pp. 5321–5347. DOI: `10.1007/s00521-022-06953-8`.

[72] W. Ma and J. Lu. *An Equivalence of Fully Connected Layer and Convolutional Layer*. 2017. arXiv: `1712.01252 [cs.LG]`.

[73] M. B. Bora, D. Daimary, K. Amitab, and D. Kandar. "Handwritten character recognition from images using CNN-ECOC". In: *Procedia Computer Science* 167 (2020), pp. 2403–2409.

[74] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting". In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[75] J. Uijlings, K. Sande, T. Gevers, and A. Smeulders. "Selective Search for Object Recognition". In: *International Journal of Computer Vision* 104 (Sept. 2013), pp. 154–171. DOI: `10.1007/s11263-013-0620-5`.

[76] A. Mohan. *Review On RCNN*. Accessed Date: March 31, 2023. URL: `https://medium.datadriveninvestor.com/review-on-rcnn-c079fc269a7d`.

[77] S. Ananth. "R-CNN for object detection: A technical paper summary". In: *Towards Data Science* (Apr. 2019).

[78] R. Gandhi. "R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms: Understanding object detection algorithms". In: *Towards Data Science* (July 2018).

[79] UNESCO. *UNESCO World Heritage Centre - Historic Centre of Salzburg*. `https://whc.unesco.org/en/list/784/`. 1996.

[80] R. Geyer, S. Knöttner, C. Diendorfer, G. Drexler-Schmid, and V. Alton. "100% Renewable Energy for Austria's Industry: Scenarios, Energy Carriers and Infrastructure Requirements". In: *Applied Sciences* 11.4 (Feb. 2021), p. 1819. ISSN: 2076-3417. DOI: 10.3390/app11041819. URL: http://dx.doi.org/10.3390/app11041819.

[81] Mayors of Europe. *HOW IS SALZBURG USING RENEWABLE ENERGY TO CHANGE THE FUTURE?* https://mayorsofeurope.eu/top-stories/how-is-salzburg-using-renewable-energy-to-change-the-future/. Mar. 2022.

[82] M. T. Pham, A. Rajić, J. D. Greig, J. M. Sargeant, A. Papadopoulos, and S. A. McEwen. "A scoping review of scoping reviews: advancing the approach and enhancing the consistency". In: *Research synthesis methods* 5.4 (2014), pp. 371–385.

[83] C. Wohlin. "Guidelines for snowballing in systematic literature studies and a replication in software engineering". In: *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering (EASE '14)*. Association for Computing Machinery. New York, NY, USA, 2014, Article 38, 1–10. DOI: 10.1145/2601248.2601268.

[84] S. Srivastava, A. Divekar, C. Anilkumar, and et al. "Comparative analysis of deep learning image detection algorithms". In: *Journal of Big Data* 8.66 (2021). Received: 12 December 2020, Accepted: 22 February 2021, Published: 10 May 2021. DOI: 10.1186/s40537-021-00434-w.

[85] Deval Shah. *Mean Average Precision (mAP) Explained: Everything You Need to Know*. Website. V7, Mar. 2022. URL: https://www.v7labs.com/blog/mean-average-precision.

[86] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. "Generalized intersection over union: A metric and a loss for bounding box regression". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 658–666.

[87] S. Awwad, B. Igried, M. Wedyan, and M. Alshira'H. "Hybrid features for object detection in RGB-D scenes". In: *Indonesian Journal of Electrical Engineering and Computer Science* 23 (Aug. 2021), pp. 1073–1083. DOI: 10.11591/ijeecs.v23.i2.pp1073-1083.

[88] *Salzburg Government*. https://www.salzburg.gv.at/. Accessed April 4, 2023.

[89] P. Dave, M. Diwan, and M. Raval. "Two Stage Traffic Sign Detection and Recognition based on K-Means and Mask RCNN Algorithm". In: (2021).

[90] X. Zhang. *Understanding Mask R-CNN Basic Architecture*. https://www.shuffleai.blog/blog/Understanding_Mask_R-CNN_Basic_Architecture.html. Accessed April 12, 2023. 2021.

[91] K. He, X. Zhang, S. Ren, and J. Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[92] Y. Huang, L. Lin, P. Cheng, J. Lyu, R. Tam, and X. Tang. “Identifying the Key Components in ResNet-50 for Diabetic Retinopathy Grading from Fundus Images: A Systematic Investigation”. In: *Diagnostics* 13.10 (May 2023), p. 1664. ISSN: 2075-4418. DOI: 10.3390/diagnostics13101664. URL: http://dx.doi.org/10.3390/diagnostics13101664.

[93] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. *Feature Pyramid Networks for Object Detection*. 2017. arXiv: 1612.03144 [cs.CV].

[94] Ankur. *Object Detection and Classification using R-CNNs*. https://www.telesens.co/2018/03/11/object-detection-and-classification-using-r-cnns/#ResNet_50_Network_Architecture. Accessed April 14, 2023. 2018.

[95] T. Karmarkar. *Region Proposal Network (RPN) — Backbone of Faster R-CNN*. Medium. Aug. 2018. URL: https://medium.com/egen/region-proposal-network-rpn-backbone-of-faster-r-cnn-4a744a38d7f9.

[96] C. Lim. *Mask R-CNN*. https://www.slideshare.net/windmdk/mask-rcnn. Accessed April 17, 2023. 2017.

[97] B. Montrucchio, A. C. Marceddu, and Y. Chang. “Waste Detection Based On Mask R-CNN”. In: (2022).

[98] T. Wang, K. Zhang, W. Zhang, R. Wang, S. Wan, Y. Rao, Z. Jiang, and L. Gu. “Tea picking point detection and location based on Mask-RCNN”. In: *Information Processing in Agriculture* (2021).

[99] T. Blaschke. “Object based image analysis for remote sensing”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 65.1 (2010), pp. 2–16.

[100] M. M. D. Oghaz, M. Razaak, H. Kerdegari, V. Argyriou, and P. Remagnino. “Scene and environment monitoring using aerial imagery and deep learning”. In: *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE. 2019, pp. 362–369.

[101] T. T. Nguyen, T. D. Hoang, M. T. Pham, T. T. Vu, T. H. Nguyen, Q.-T. Huynh, and J. Jo. “Monitoring agriculture areas with satellite images and deep learning”. In: *Applied Soft Computing* 95 (2020), p. 106565.

[102] P. Zhang, Y. Ke, Z. Zhang, M. Wang, P. Li, and S. Zhang. “Urban land use and land cover classification using novel deep learning models based on high spatial resolution satellite imagery”. In: *Sensors* 18.11 (2018), p. 3717.

[103] D. Rolnick, P. L. Donti, L. H. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. S. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown, et al. “Tackling climate change with machine learning”. In: *ACM Computing Surveys (CSUR)* 55.2 (2022), pp. 1–96.

[104] K. Chaurasia, R. Nandy, O. Pawar, R. R. Singh, and M. Ahire. “Semantic segmentation of high-resolution satellite images using deep learning”. In: *Earth Science Informatics* 14.4 (2021), pp. 2161–2170.

[105] N. Jean, M. Burke, M. Xie, W. M. Davis, D. B. Lobell, and S. Ermon. "Combining satellite imagery and machine learning to predict poverty". In: *Science* 353.6301 (2016), pp. 790–794. DOI: 10.1126/science.aaf7894.

[106] P. Golej, J. Horak, P. Kukuliac, and L. Orlikova. "Vehicle detection using panchromatic high-resolution satellite images as a support for urban planning. Case study of Prague's centre". In: *GeoScape* 16.2 (2022), pp. 108–119.

[107] I. Duporge, O. Isupova, S. Reece, D. W. Macdonald, and T. Wang. "Using very-high-resolution satellite imagery and deep learning to detect and count African elephants in heterogeneous landscapes". In: *Remote Sensing in Ecology and Conservation* 7.3 (2021), pp. 369–381.

[108] T. Ishii, R. Nakamura, H. Nakada, Y. Mochizuki, and H. Ishikawa. "Surface object recognition with CNN and SVM in Landsat 8 images". In: *2015 14th IAPR International Conference on Machine Vision Applications (MVA)*. 2015, pp. 341–344. DOI: 10.1109/MVA.2015.7153200.

[109] Dr. Prasun Ranjan. *Satellite Image Processing: Applications and Possibilities*. Website. Oct. 2020. URL: https://aabsys.com/satellite-image-processing-applications-and-possibilities/.

[110] Y. Feng, W. Diao, X. Sun, M. Yan, and X. Gao. "Towards Automated Ship Detection and Category Recognition from High-Resolution Aerial Images". In: *Remote Sensing* 11.16 (Aug. 2019), p. 1901. ISSN: 2072-4292. DOI: 10.3390/rs11161901. URL: http://dx.doi.org/10.3390/rs11161901.

[111] X. X. Zhu, D. Tuia, L. Mou, G.-S. Xia, L. Zhang, F. Xu, and F. Fraundorfer. "Deep Learning in Remote Sensing: A Comprehensive Review and List of Resources". In: *IEEE Geoscience and Remote Sensing Magazine* 5.4 (2017), pp. 8–36. DOI: 10.1109/MGRS.2017.2762307.

[112] G. Jialeng, S. Suárez de la Fuente, and T. R. Smith. "BoatNet: Automated Small Boat Composition Detection using Deep Learning on Satellite Imagery". In: *UCL Open: Environment Preprint* (2022).

[113] *Copernicus: Sentinel-1 - 2021*. Website. Aug. 2022. URL: https://www.eoportal.org/satellite-missions/copernicus-sentinel-1-2021#spacecraft.

[114] A. Sampath, P. Bijapur, A. Karanam, V. Umadevi, and M. Parathodiyil. "Estimation of rooftop solar energy generation using Satellite Image Segmentation". In: *2019 IEEE 9th International Conference on Advanced Computing (IACC)*. 2019, pp. 38–44. DOI: 10.1109/IACC48062.2019.8971578.

[115] T. Mujtaba and M. A. Wani. "Automatic solar panel detection from high-resolution orthoimagery using deep learning segmentation networks". In: *Deep Learning Applications, Volume 2* (2021), pp. 101–122.

[116] V. Mnih and G. E. Hinton. "Learning to detect roads in high-resolution aerial images". In: *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part VI 11*. Springer. 2010, pp. 210–223.

[117] M. A. Wani and T. Mujtaba. "Segmentation of Satellite Images of Solar Panels Using Fast Deep Learning Model". In: *International Journal of Renewable Energy Research (IJRER)* 11.1 (2021), pp. 31–45.

[118] A. Mahmoud, S. Mohamed, R. El-Khoribi, and H. AbdelSalam. "Object Detection Using Adaptive Mask RCNN in Optical Remote Sensing Images". In: *International Journal of Intelligent Engineering and Systems* 13 (Feb. 2020), pp. 65–76. DOI: `10.22266/ijies2020.0229.07`.

[119] A. O. Vuola, S. U. Akram, and J. Kannala. "Mask-RCNN and U-Net Ensembled for Nuclei Segmentation". In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. 2019, pp. 208–212. DOI: `10.1109/ISBI.2019.8759574`.

[120] J. Dai, K. He, and J. Sun. "Instance-aware semantic segmentation via multi-task network cascades". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3150–3158.

[121] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. "Fully convolutional instance-aware semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2359–2367.

[122] S. Sánchez Hernández, H. Romero, and A. Morales. "A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework". In: *IOP Conference Series: Materials Science and Engineering* 844 (June 2020), p. 012024. DOI: `10.1088/1757-899X/844/1/012024`.

[123] S. Rajjak and A. Kureshi. "Multiple-Object Detection and Segmentation Based on Deep Learning in High-Resolution Video Using Mask-RCNN". In: *International Journal of Pattern Recognition and Artificial Intelligence* 35 (Oct. 2021). DOI: `10.1142/S0218001421500385`.

[124] J. Ruiz-Santaquiteria, G. Bueno, O. Deniz, N. Vallez, and G. Cristobal. "Semantic versus instance segmentation in microscopic algae detection". In: *Engineering Applications of Artificial Intelligence* 87 (2020), p. 103271.

[125] K. Zhao, J. Kang, J. Jung, and G. Sohn. "Building extraction from satellite images using mask R-CNN with building boundary regularization". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2018, pp. 247–251.

[126] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen. "Deep learning for generic object detection: A survey". In: *International journal of computer vision* 128 (2020), pp. 261–318.

[127] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. "Speed/accuracy trade-offs for modern convolutional object detectors". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7310–7311.

[128] C. Zhang. *How to create custom COCO data set for instance segmentation*. `https://www.dlology.com/blog/how-to-create-custom-coco-data-set-for-instance-segmentation/`. Accessed April 4, 2023. 2019.

[129] F. Pezoa, J. L. Reutter, F. Suarez, M. Ugarte, and D. Vrgoč. "Foundations of JSON schema". In: *Proceedings of the 25th international conference on World Wide Web*. 2016, pp. 263–273.

[130] Edge AI and Vision Alliance. *Exploring Data Labeling and the 6 Different Types of Image Annotation*. Blog post. Apr. 2022. URL: `https://www.edge-ai-vision.com/2022/04/exploring-data-labeling-and-the-6-different-types-of-image-annotation/`.

[131] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. "Microsoft coco: Common objects in context". In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 740–755.

[132] R. Padilla, S. L. Netto, and E. A. Da Silva. "A survey on performance metrics for object-detection algorithms". In: *2020 international conference on systems, signals and image processing (IWSSIP)*. IEEE. 2020, pp. 237–242.

[133] P. Parhar, R. Sawasaki, A. Todeschini, C. Reed, H. Vahabi, N. Nusaputra, and F. Vergara. *HyperionSolarNet: Solar Panel Detection from Aerial Images*. Jan. 2022.

[134] Jonathan Hui. *mAP (mean Average Precision) for Object Detection*. Medium. Mar. 2018. URL: `https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173`.