TÉCNICO LISBOA

# Deep Learning Algorithms for Sunspot Detection

**André Mourato Martins**

Thesis to obtain the Master of Science Degree in

## Aerospace Engineering

Supervisors: Prof. Rodrigo Martins de Matos Ventura
Dr. João Pedro de Sousa Faria

## Examination Committee

Chairperson: Prof. Afzal Suleman
Supervisor: Prof. Rodrigo Martins de Matos Ventura
Member of the Committee: Prof. Arlindo Manuel Limede de Oliveira

**December 2021**

This thesis is dedicated to my Father, who died as a result of a shipwreck while I was walking the path to this graduation. Your legacy will not be forgotten.

# Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

# Acknowledgments

Firstly, I would like to thank my supervisors, Professor Rodrigo Ventura and Dr João Faria, for their support and guidance throughout this dissertation. Your contribution was most certainly appreciated.

I cannot begin to express my thanks to certain professors whom I had the honour and good fortune of meeting during my journey in this institution. Each in their own way, inspired me and nurtured me an eagerness to learn further and passion for knowledge. Among them are, Professor Ana Maria Rego, Professor Carlos Cardoso Fernandes, Professor Luiz Braga Campos, Professor Mario Graça, Professor Pedro Borges Dinis and Professor Sofia Nique. Each one of you surely marked this passage, from where I can proudly say I had the privilege to know you and learn from you.

I would also like to extend my sincere gratitude to Professor Joaquim Guerreiro Marques, for the support, encouragement and enlightenment in a difficult time, which later led to the transition to my current academic field.

From the many acquaintances made during this journey, I had the opportunity to meet two great individuals, with whom I shared a whole spectrum of moments, to whom I am grateful for their friendship and for making this voyage so special. To you Francisco and João, thank you.

To my sister, Carolina, for her affection and love, and for taking care of our Mother when I was not there. Thank you.

To my wonderful and dear Mother, thank you for your unconditional love and for your constant support. Thank you for all the sacrifices you made throughout my life for providing me and my sister with the best. This achievement is as much mine as yours. Thank you.

My sincere thank you to all those who I did not mention and contribute to this enriching journey.

Lastly, I thank you God, for everything.

# Resumo

A influência solar no clima espacial e ambiente terrestre é inteligível. Forte atividade de tempestades geomagnéticas podem afetar significativamente astronautas em órbita, os sistemas de comunicação e de GPS, e ainda causar disrupção nas redes de distribuição de energia na Terra; tornando vital a contínua monitorização e previsão de atividade solar. Manchas solares são perturbações magnéticas na fotosfera caracterizadas pela sua aparência escura no disco solar, estando diretamente relacionadas com os fenómenos que contribuem para estas fortes tempestades, nomeadamente erupções solares e ejeções de massa coronal. Esta tese encontra-se na intersecção entre a vigilância solar e visão computacional, pela aplicação de algoritmos de estado da arte de aprendizagem profunda na deteção automática de manchas e grupos de manchas solares. Com base em duas abordagens, segmentação semântica e segmentação instancial, três algoritmos foram implementados, entre eles a U-Net, U-Net 3+ e a Mask R-CNN. A abordagem semântica apresentou resultados superiores a dois algoritmos de estado da arte de deteção de manchas solares com 74.2% IoU, deixando sólidas promessas de superar o melhor algoritmo comparado, com um futuro aumento de capacidade da rede. A abordagem instancial, ainda um desafio no campo de visão computacional, alcançou 51.7 AP de caixa delimitadora e precisão de 78.6% na previsão do número de grupos de manchas solares entre 2010 e 2014 no conjunto de teste. Ambos resultados são promissores, abrindo caminho para mais investigação e desenvolvimento apontando para a efetivação de um algoritmo autónomo e eficaz para o efeito.

**Palavras-chave:** Sol, aprendizagem profunda, redes neurais, deteção, segmentação, manchas solares

# Abstract

The solar influence on the space weather and terrestrial environment is intelligible. Strong geomagnetic storm activity can significantly affect astronauts in orbit, communications and GPS systems and disrupt Earth's power distribution networks, making continuous monitoring and forecasting of solar activity vital. Sunspots are magnetic disturbances in the photosphere characterized by their dark appearance in the solar disk, being directly related to phenomena that contribute to these intense storms, namely solar flares and coronal mass ejections. This thesis lies at the intersection between solar surveillance and computer vision by applying state-of-the-art deep learning algorithms in the automatic detection of sunspots and sunspot groups. Three algorithms were implemented based on two approaches, semantic segmentation and instance segmentation, among them U-Net, U-Net 3+ and Mask R-CNN. The semantic approach presented superior results to two state-of-the-art sunspot detection algorithms with 74.2% IoU, leaving solid promises of outperforming the best algorithm compared with, increasing the network capacity. An improved Mask R-CNN achieved 51.7 AP of bounding box and 78.6% in predicting the number of sunspot groups between 2010 and 2014 in the test set. Both results are promising, paving the path for further research and development, aiming at the execution of an autonomous and efficient algorithm for the purpose.

**Keywords:** Sun, deep learning, neural networks, detection, segmentation, sunspots

# Contents

Acknowledgments . . . . . vii

Resumo . . . . . ix

Abstract . . . . . xi

List of Tables . . . . . xv

List of Figures . . . . . xvii

**1 Introduction 1**

1.1 Motivation . . . . . 1

1.2 Thesis Outline . . . . . 3

**2 Background and State-of-the-Art 5**

2.1 The Sun . . . . . 5

2.2 Automatic Sunspot Detection . . . . . 11

2.3 Deep Learning . . . . . 14

2.3.1 Neural Networks . . . . . 14

2.3.2 Convolutional Neural Networks . . . . . 16

2.4 Semantic Segmentation . . . . . 17

2.4.1 U-Net . . . . . 18

2.4.2 U-Net 3+ . . . . . 19

2.5 Instance Segmentation . . . . . 21

2.5.1 Mask R-CNN . . . . . 22

**3 Methodology 33**

3.1 Dataset . . . . . 33

3.2 Evaluation Metrics . . . . . 41

3.2.1 COCO Detection Evaluation Metrics . . . . . 42

3.3 U-Net and U-Net 3+ . . . . . 43

3.4 Mask R-CNN . . . . . 46

3.4.1 Mask R-CNN fine-tune . . . . . 46

**4 Results 55**

4.1 U-Net and U-Net 3+ . . . . . 55

4.2 Mask R-CNN . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 57

**5 Conclusions 63**

5.1 Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 64

**Bibliography 67**

**A Sunspot Groups McIntosh Classification Example Images 75**

**B Ground-truth Data Examples 77**

**C Inference Samples 79**

# List of Tables

3.1 DHO available sunspot revised catalogues summary. . . . . . . . . . . . . . . . . . . . . . 33

3.2 Mask R-CNN bounding box and instance segmentation mask results on test set for randomly initialized standard baseline. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49

3.3 Region Network Proposal main hyperparameters. . . . . . . . . . . . . . . . . . . . . . . . 50

3.4 Sunspot group training instances in COCO dimensions with SDO images rescale to $1024^2$ pixels. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 52

3.5 RoI Heads main hyperparameters. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53

4.1 Comparison between the implemented U-Net model and three state-of-the-art sunspot detection algorithms. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

4.2 Comparison between U-Net and U-Net 3+ results with an effective network input of $512^2$ pixels. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

4.3 Best experiments' modification setups with respect to the standard baseline. . . . . . . . 58

4.4 Mask R-CNN (FPN) bounding box and instance segmentation mask results on test set, for both pre-trained and random initialization of standard baseline together with v1 setup comparison. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58

4.5 Mask R-CNN (FPN) bounding box and instance segmentation mask results on test set for each setup. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 58

4.6 Comparison between the number of ground-truth groups of the test set and the number of groups predicted by the v2 setup. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 62

# List of Figures

2.1 Structural anatomy visualisation of the Sun. . . . . . . . . . . . . . . . . . . . . . . . . 6

2.2 Solar differential rotation. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

2.3 Butterfly diagram showing paired position and area of sunspots. . . . . . . . . . . . . . . 7

2.4 International yearly mean sunspot number (black) up to 1749 and monthly 13-month smoothed sunspot number (blue) from 1749 to 2021. . . . . . . . . . . . . . . . . . . . . . 8

2.5 Solar rotational angle with respect to the celestial equator. . . . . . . . . . . . . . . . . . 10

2.6 Solar coordinate systems. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

2.7 Example of SMART and STARA detections. . . . . . . . . . . . . . . . . . . . . . . . . 12

2.8 Example of ASAP detections. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

2.9 Feedforward Fully-Connected Neural Network. . . . . . . . . . . . . . . . . . . . . . . . 15

2.10 Kernel movement through a convolutional unit. . . . . . . . . . . . . . . . . . . . . . . . 17

2.11 Example of a semantic segmented image. . . . . . . . . . . . . . . . . . . . . . . . . . 17

2.12 U-Net architecture. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

2.13 Comparison between U-Net, U-Net++ and U-Net 3+ architectures. . . . . . . . . . . . . . 19

2.14 Construction illustration of the full-scale aggregated feature map of the third decoder layer. 20

2.15 Example of an instance segmented image. . . . . . . . . . . . . . . . . . . . . . . . . . 21

2.16 Mask R-CNN architecture. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

2.17 Residual block of ResNet and ResNext. . . . . . . . . . . . . . . . . . . . . . . . . . . 23

2.18 Left: ResNet-50. Right: ResNeXt-50 with a 32×4d structure. Inside each bracket are the shape of a residual block, and outside the bracket is the number of stacked blocks on a stage. $C$ corresponds to the cardinality. . . . . . . . . . . . . . . . . . . . . . . . . . . 24

2.19 Feature Pyramid Network architecture. . . . . . . . . . . . . . . . . . . . . . . . . . . 25

2.20 Region Proposal Network Structure. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

2.21 Region Proposal Network Head and anchor box placement. . . . . . . . . . . . . . . . . 26

2.22 Re-sampling procedure. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

2.23 RoIAlign operation. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

2.24 RoI Heads . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 31

3.1 Full disk white-ligh observation from the Mitaka Observatory and its visualization of the annotated mask provided by DPD . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34

3.2 Example of contours anomalies detected in the ground-truth masks generation from ground-based catalogues. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37
3.3 Limb darkening intensity profile detection and subsequent correction of a record from 22/1/2011 with SLDTk. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
3.4 Example of a semantic segmentation mask generation. . . . . . . . . . . . . . . . . . . 39
3.5 Example of an instance segmentation mask loaded in Detectron2. . . . . . . . . . . . . . 40
3.6 Poly . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 44
3.7 Training behaviour comparison between of random initialized model and with ImageNet pre-training. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 48

4.1 Comparison between the ground-truth mask and the predicted mask by the best U-Net model in a disk zoomed area. Better visualization in Appendix C. . . . . . . . . . . . . . . 57
4.2 Comparison between the ground-truth mask and the predicted mask by the U-Net model in a disk zoomed area. Better visualization in Appendix C. . . . . . . . . . . . . . . . . . 57
4.3 Training behaviour evolution of Region Network Proposal. . . . . . . . . . . . . . . . . . 59
4.4 Training behaviour evolution of Box Head. . . . . . . . . . . . . . . . . . . . . . . . . 59
4.5 Training behaviour evolution of Mask Head and Mask R-CNN with ResNeXt-101-FPN-v2 total loss. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 59
4.6 Comparison between an instance segmentation ground-truth sample and its Mask R-CNN w/ ResNeXt-FPN-v3 prediction. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 61
4.7 Comparison between an zoomed area of a instance segmentation ground-truth sample and its Mask R-CNN (ResNeXt-v3) prediction. Better visualization in Appendix C. . . . . . 62

A.1 Sunspot group class example images taken from SDO for each McIntosh class. . . . . . . 76

B.1 Annotated mask visualization provided by HMIDD from 21/11/2013. . . . . . . . . . . . . 77
B.2 Superimposed ground-truth generated semantic mask. . . . . . . . . . . . . . . . . . . . 78

C.1 Comparison between the ground-truth mask and the predicted mask by the best U-Net model in a disk zoomed area. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 79
C.2 Comparison between the ground-truth mask and the predicted mask by the best U-Net model in a disk zoomed area. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 80
C.3 Comparison between an instance segmentation ground-truth sample and its Mask R-CNN w/ ResNeXt-FPN-v3 prediction. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 81
C.4 Comparison between an instance segmentation ground-truth sample and its Mask R-CNN w/ ResNeXt-FPN-v3 prediction. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 82

# Chapter 1

# Introduction

## 1.1 Motivation

The Sun, a middle-aged yellow dwarf star, has been observed for millennia. Throughout the evolutionary trajectory of the Homo Sapiens, the Sun has gone from an object of mystery and worship to nowadays being an active subject of scientific research, carefully followed and analysed. Among various aspects that justified this surveillance, arguably the most important is the impact that solar activity can have on life on Earth. This thesis aims to introduce a new paradigm in the field of solar surveillance, specifically in the automatic detection of sunspots, by applying state-of-the-art deep learning algorithms.

The existence of features on the solar surface was known even before Galileo invented the telescope in the 17th century. Sunspots are manifestations of the solar magnetic field, visually spotted by their dark patch appearance on the solar surface. The first-ever written record of solar activity and sunspots goes back to 800 B.C. by Chinese astronomers [1]; however, the earliest known actual drawing of sunspots is dated just in 1128, belonging to an English monk named John of Worcester. Galileo and Christoph Scheiner were among the first to perform telescopic observations of sunspots. Until the 19th century, there was a rather small improvement in solar imaging, the reason why the quality of the first drawings resultant from telescopic observations was fairly similar to those two centuries later [2]. Although the quality was reasonable enough for a proper observation, consistent monitoring was still missing.

With the beginning of the space age in the last century and the establishment of new observatories, an entirely new dimension of solar studies began. Not only continuous observations started but also a considerable increase of imaging resolution was reached, which is especially important in the observation of massive dimensions bodies as the Sun.

More than ten probes have been launched so far to study the Sun specifically. The Solar Dynamics Observatory (SDO), Parker Solar Probe and the Solar Orbiter are currently operational. SDO was launched in 2010 by NASA and is studying how solar activity is created and how ”space weather” originates from that activity. This is accomplished by determining how the Sun’s magnetic field is generated and structured and how this magnetic energy is converted and released into space in the form of solar wind, energetic particles and variations in the solar irradiance.

The current core elements of the study of the Sun are mainly related to the solar wind, coronal mass ejections (CMEs) and solar flares. The large increase in the solar particles flux originated by solar flares, and coronal mass ejections collide with Earth's magnetosphere by a factor up to 10000 times the average flux. These abrupt bursts in the solar wind stream reach the Earth in between 30 to 70 hours and cause geomagnetic storms involving distortions of the Earth's magnetic field, which shields us from most of the charged-particle emission from the Sun. These phenomena are of significant importance to space weather and the Earth environment. Solar flares and CMEs are directly associated with sunspots. They occur near sunspots, commonly at the dividing line between areas of oppositely directed magnetic fields. An increase of sunspots numbers will result in more solar flares, hence originating an increase in geomagnetic storm activity towards Earth [3].

A tremendous amount of research has been conducted to understand how much the solar output affects Earth's climate. The conclusions are clear; it can have significant impacts. Evidence shows that Earth's climate is sensitive to quite feeble changes in the energy coming from the Sun. Periods of massive sunspot activity are associated with increases in the Sun's energy output, characterised by dramatic enlargements of ultraviolet radiation, which have a considerable effect on Earth's atmosphere. Not surprisingly, the reverse is true during minimum sunspot activity. Historically, the period from 1500 to 1850 was distinguished by regional cold conditions, later denominated by ”The Little Ice Age”. Interestingly enough, this period coincided with the so-called Maunder Minimum, a known span of near-zero sunspot activity from 1645 to 1715 [4].

There are practical needs for predicting space weather on time windows ranging from a few hours up to a complete sunspot cycle or even longer. Perturbations in the ionosphere caused by solar-induced turbulence give rise to phase and amplitude fluctuations in the signals coming from and to satellites, disrupting communications. An example of that occurs with Global Positioning Satellite (GPS) receivers that can temporarily lose their signals' lock and sometimes even return wrong information. In large-scale electrical power networks, the induced currents can disrupt the distribution of electricity by causing failures in the transformers that subsequently deregulate the voltage throughout the grid [3, 5].

In the era of artificial satellites, space weather has become progressively important. Increases in the solar EUV radiation cause increased heating of the upper atmosphere, making it expand and causing drag increments on satellites in low orbit. Consequently, lifetime estimations of satellites in every mission must take these factors into account to have accurate predictions, and these forecasts take place several years ahead. With the start of continuous human presence in the International Space Station and promises of human space endeavours in the coming decades, space weather has taken on significantly greater importance. The energised particles emitted by solar flares and CMEs, as well as galactic cosmic rays, constitute a real threat for astronauts in space, with evidence showing that cancer and degenerative diseases are to be expected from these exposures[1].

The importance of solar studies and continuous surveillance is graspable. Currently, SDO sends over one terabyte of data daily to Earth, and methods for fast and reliable analysis are more critical than ever. Over the past two decades, the research on automated detection of solar features has increased dra-

[1] https://www.nasa.gov/analogs/nsrl/why-space-radiation-matters

matically following the volume increase of data availability, with several algorithms and image processing techniques being developed. Although the accomplishments so far have proven to be successful, many aspects can be improved. Most of the softwares developed are based on threshold techniques, border methods (edge detection), mathematical morphology or Bayesian pattern recognition, that rely immensely on properties of the data, image resolution and average intensity, which tend to be difficult to generalise to unseen data without supervision, they are not precisely autonomous in their extent and present far from real-time performance [6].

The last decade brought numerous advances in the field of machine learning, more precisely in deep learning, resultant from the significant leaps achieved in the hardware industry, making it possible for a vast amount of theories and algorithms postulated in the past, like deep convolutional neural networks and recurrent neural networks, to be to tested and empirically studied. With computers being equipped with more computational capacity, faster and cheaper processing, and mainly graphics processing units (GPUs), the computational power (measured in FLOPS) increased by 170 times over the past 10-year span[2].

The algorithms proposed in this dissertation combine state-of-the-art computer vision algorithms based on deep learning techniques for automatic detection of sunspots and sunspots groups. The goal is not only to correctly detect these solar features but also to tackle the limitations already mentioned presented by the current methods. These algorithms take as input continuum-intensity images from the Helioseismic and Magnetic Imager (HMI), one of the three instruments aboard the SDO, and are trained, validated and tested using sunspots catalogues provided by the Debrecen Heliophysical Observatory. The results are promising, with one approach surpassing two state-of-art algorithms and in range to reach the best directly comparable algorithm, and the other capable of detecting and clustering sunspot groups pretty well, thus paving the way for more research to be done in this matter, towards scientific level performance. The main scientific contributions of this thesis can be summarized as follows:

- Design and construction of a method to generate binary sunspot masks from the Debrecen Heliophysical Observatory sunspot catalogues to build the ground-truth datasets for semantic and instance segmentation.

- Application of U-Net and U-Net 3+ to sunspot segmentation, outperforming the results presented by a state-of-the-art algorithm.

- Introduction and development of a novelty approach to sunspot group detection and segmentation, based on instance segmentation, implemented using Mask R-CNN.

## 1.2 Thesis Outline

This document is comprised of five chapters. Chapter 2 compacts a theoretical background of the essential topics that are developed later. It starts with an overview of the solar physics relevant to the total understanding of this work, followed by an overview of the state-of-the-art automatic sunspot detection

[2] https://ourworldindata.org/technological-progress

methods. In addition, a brief description of deep learning and artificial neural networks is presented. Note that the purpose here is not to make an extensive description of all related theory and procedures associated with every detail that later is employed, thus it is assumed that the reader has a complete understanding of deep learning to its full extent. Lastly, the state-of-the-art architectures used in both approaches are described in conjunction with the deep learning techniques on which they were based.

Chapter 3 encapsulate the aspects related to the methodology and implementation of this work. Extending from a detailed description of the data to the dataset construction, were the challenges that the whole data preparation process implied from the data acquisition to the final accurate ground-truths are described. Additionally, the evaluation metrics employed in both approaches and a detailed description of the experiments conducted for the three implemented models is conducted.

Chapter 4 presents the final quantitative results for each algorithm, description of implementation aspects and data loading details, examples of inference samples from each network, and analysis of the results.

Chapter 5 presents the final conclusions and description of future works, and how the procedures developed here and the results can be refined in further applications.

Appendix A provides sunspot groups McIntosh classification examples with images from the SDO. Appendix B provides some ground-truth data examples, and Appendix C presents some inference samples from the final models in the test set.

# Chapter 2

# Background and State-of-the-Art

## 2.1 The Sun

The Sun is at the heart of our solar system. More than 99% of the solar system's mass is in our star, and its gravity keeps the planets, satellites, asteroids, and the minor debris of its surroundings, in their orbits. Although it is a fairly typical star, the Sun is a stellar body of the highest importance in astronomy, as the only star that we can observe in great detail. In order to understand the object of study of this work – sunspots – first, we have to understand certain properties and structural behaviour of the Sun.

From our star and all others, the only information we can directly obtain is its radiation and position. Every other aspect related to the physics of stellar structure, birth and evolution are derived from certain quantities such as mass, radius, composition, rotation, and magnetic field, which are also inferred. In astrophysics, the quantity which can be measured with the greatest accuracy is time; as a result, each of these quantities derives from it [7].

The Standard Solar Model (SSM) [8] is an important mathematical formulation of the Sun treated as a symmetric plasma sphere that constitutes a reference to every stellar evolution calculation. This model is used to predict the internal observables[1] such as neutrino fluxes and oscillation frequencies, through the modulation of classical stellar equations and the knowledge of fundamental physics, such as photon interaction and nuclear reaction rates, which is consequently used to validate its assumptions for its generalization to other stars [9].

The anatomical structure of the Sun is illustrated in Figure 2.1 and can be stratified into six different layers: the core, the radiative zone, the convective zone, the visible surface called the photosphere, the chromosphere, and the outermost region, the corona. The core generates energy through thermonuclear reactions, a process where two lighter atoms fuse to form a heavier one whose mass is smaller than the mass of its constituents; the temperature in the Sun's core is estimated to be around 15 million degrees Celsius. The energy produced in the core moves slowly outward by radiation, taking more than 170,000 years travelling through the radiative zone up to the convective zone. The convection zone starts at a third of the solar radius below the solar surface, where energy is transported outwards by

[1] physical quantities that can be measured.

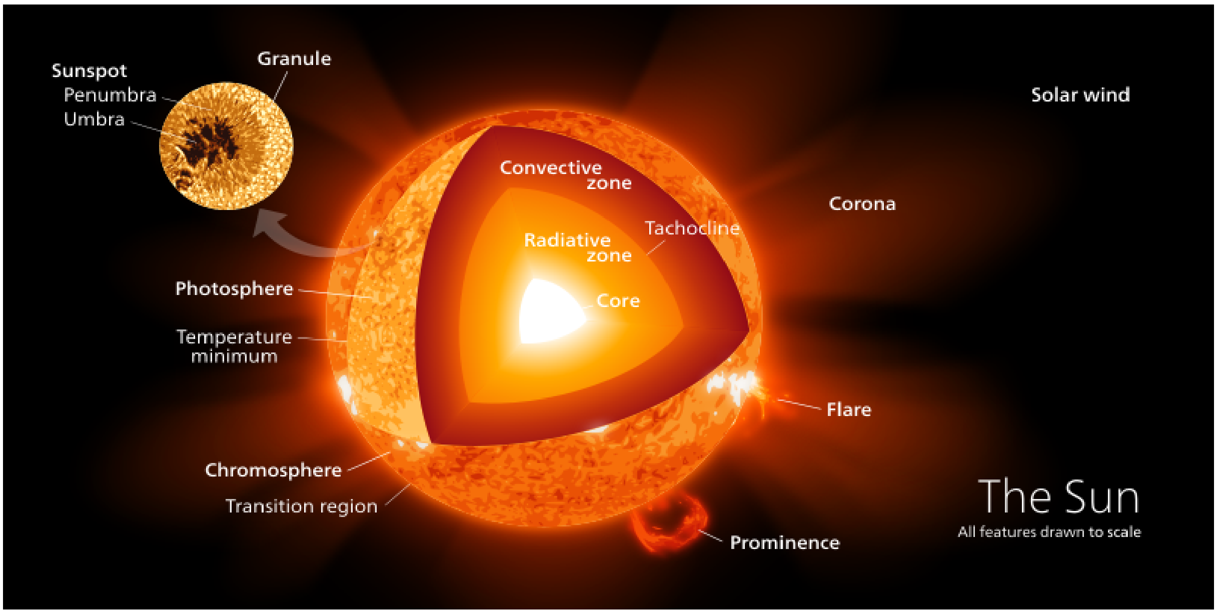


Figure 2.1: Structural anatomy visualisation of the Sun[2].

convective motions. The temperature drops below 2 million degrees Celsius in the convection zone, where large bubbles of hot plasma move upwards. The visible surface, called the photosphere, is a 500 km gassy region from which light is radiated. The temperature here is around 5500º C, and it is surprisingly much colder than the Sun's atmosphere above it. Because the upper part of the photosphere is colder than the lower part, an image of the Sun appears brighter in the centre than on the edge or limb of the solar disk, in a phenomenon known as limb darkening [7]. Above the photosphere are the tenuous chromosphere and the corona, which make up the thin solar atmosphere. The temperatures here vary roughly from 6000ºC to $2\times10^6$ ºC, in a phenomenon that is still not understood completely, decreasing in the outer corona, which expands and flows outward into space as the solar wind. Generally, we cannot see the solar atmosphere, the brightness of the underlying photosphere drowns out its light. It is only during solar eclipses that typically we can see it or through the use of telescopes.

Sunspots, the most conspicuous manifestation of the Sun's magnetic fields, are regions on the photosphere that appear darker than the surrounding areas; this happens due to a reduced surface temperature originated by strong magnetic fields that partially inhibit the regular transport of energy by convection [3]. A typical sunspot comprises a dark centre called the umbra and a lighter ring called the penumbra. They come in a wide range of sizes, from several times larger than Earth to so considerably small that telescopic observation is difficult, sometimes being almost negligible to the human eye. Although single spots appear, they tend to occur in pairs or groups, with the pair constituents being of opposite magnetic polarity. By observing the movement of sunspots, the well-known English astronomer Riccard Carrington found out that the Sun does not rotate as a solid body but rather differentially, faster at the equatorial zones and slower at higher solar latitudes as shown in Figure 2.15. This behaviour leads to a non-constant rotational period, increasing from 24.47 sidereal days at the solar equator to approximately

[2] https://commons.wikimedia.org/wiki/File:Sun_poster.svg

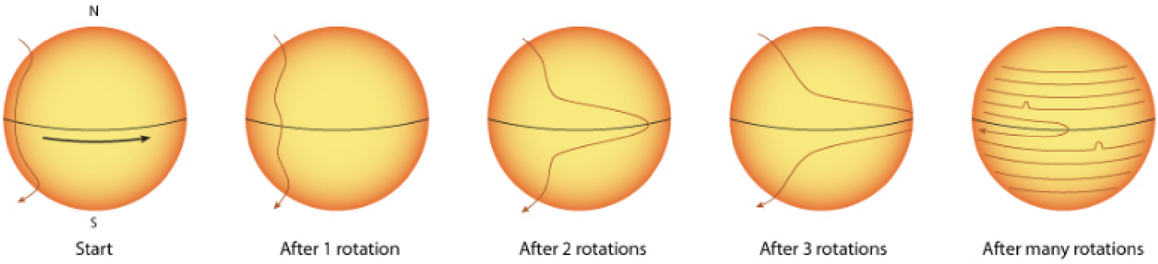


Figure 2.2: Solar differential rotation. [10]

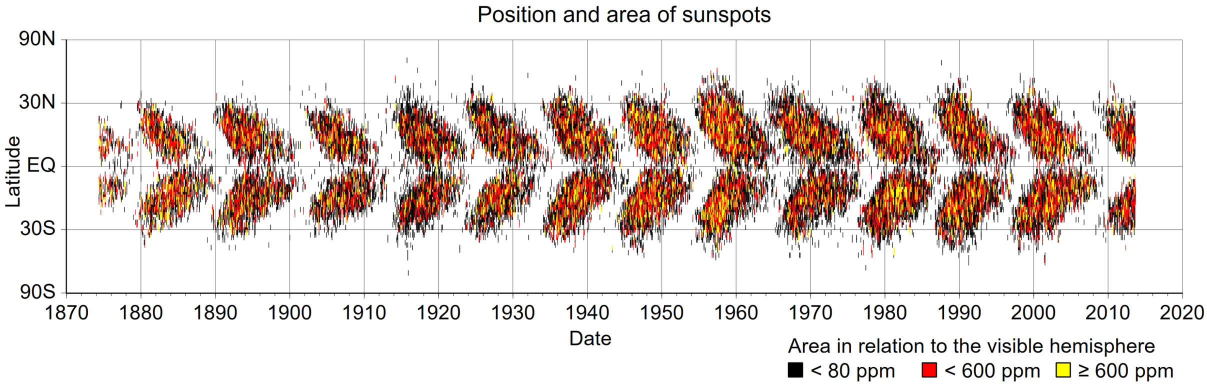


Figure 2.3: Butterfly diagram showing paired position and area of sunspots[4].

38 sidereal days at the poles [5]. The differential rate of rotation between the radiative and convection zones along with turbulent convective motions creates the so-called solar dynamo, a phenomenon that is responsible for the generation of electric currents and consequently magnetic fields.

Variability in the sunspots number with time was first noticed by an amateur German astronomer, Samuel Schwabe, and later found to lead to the solar cycle's formulation, a nearly periodic 11-year change in the Sun's activity. During a solar cycle, the sunspot activity increases until it reaches a maximum, a time characterized by high rates of solar material ejection, number of sunspots (and size), which can reach several hundred in a day and a flip of the magnetic poles. Along with the number of sunspots, the location of sunspots also varies throughout the sunspot cycle. At solar minimum, sunspots tend to form around latitudes of 30º to 45º in both hemispheres (North and South), and as the cycle progresses through the solar maximum, sunspots tend to concentrate closer to the equator, around a latitude of 15º, forming even closer towards the end of a cycle. This recurrent positional behaviour of sunspots predicted by Sporer's law [3] gave later rise to the butterfly diagrams carried out by Edward Maunder, illustrated in Figure 2.3.

The main solar activity index is the sunspot number; however, this number is not trivial to infer and can fluctuate significantly between different observers, methodologies and criteria. To accommodate this issue, Rudolf Wolf in 1848 established a relative sunspot number, known as the Wolf [7] number. This

[3] Although given its name, this behaviour was actually first noticed by R. Carrington [11].

[4] `https://solarscience.msfc.nasa.gov/images/bfly.pdf`

[6] `http://www.sidc.be/silso/ssngraphics`

[7] also called Zurich sunspot number.

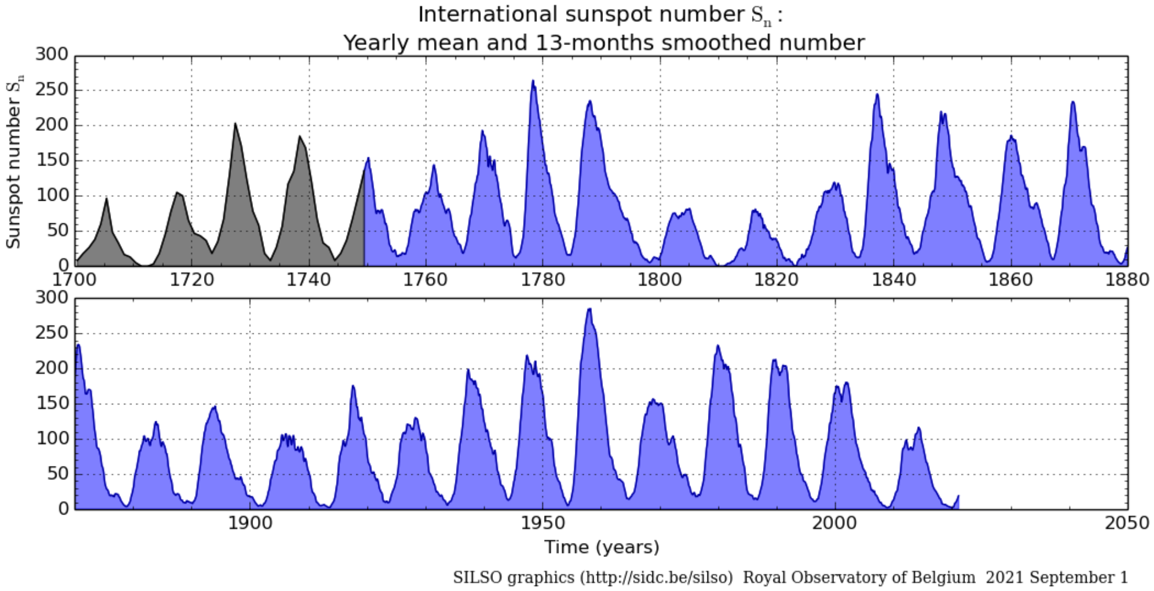


Figure 2.4: International yearly mean sunspot number (black) up to 1749 and monthly 13-month smoothed sunspot number (blue) from 1749 to 2021[6].

daily measurement is given by:

$$R = K \cdot (10 \cdot g + s) \tag{2.1}$$

where $g$ is the number of groups of sunspots, $s$ is the number of individual spots, and $K$ is a factor that attempts to account for the other variables in the measurement, such as location and instrumentation, also known as the personal reduction coefficient. As previously mentioned, sunspots most commonly appear in groups, therefore it should be understood how can these features be separated and classified into different groups.

The modified Zurich McIntosh sunspot classification system is the standard method followed to the stated purposed and is based on the general form $Zpc$, where $Z$ is the Modified Zurich Class (Z), $p$ describes the penumbra of the principal spot and $c$ describes the distribution of spots in the interior of the group. Although there are 60 possible combinations of valid McIntosh classifications[8] the $Z$ component is in fact the one which partitions the groups into the following different classes [12]:

- **A**: a small single unipolar sunspot. Representing either the formative or final stage of evolution;
- **B**: a bipolar sunspot group with no penumbra on any of the spots;
- **C**: a bipolar sunspot group. One sunspot must have penumbra;
- **D**: a bipolar sunspot group with penumbra on both ends of the group. Longitudinal extent does not exceed 10°;
- **E**: a bipolar sunspot group with penumbra on both ends. Longitudinal extent exceeds 10° but not 15°;

[8] 7 in $Z$, 6 in $p$, 4 in $c$.

- **F**: an elongated bipolar sunspot group with penumbra on both ends. Longitudinal extent of penumbra exceeds 15°;
- **H**: a unipolar sunspot group with penumbra.

Sunspots groups appear in active regions, areas in the solar surface where the magnetic field is particularly strong and complex, although not all active regions produce sunspots. These regions are easily spotted in magnetograms and ultraviolet spectrum records. Currently, the Sunspot Index and Long-term Solar Observations (SILSO) and the Space Weather Prediction Center (SWPC) are the most highly regarded institutions in the study, surveillance and forecast, of sunspots and active regions respectively. SILSO have the longest record of sunspots number and sunspot groups, going from 1700 to the current days, using only data resultant from ground observation. Although there was already a significant amount of research and development of automatic methods, as it is possible to observe now the Sun in more sophisticated ways, SILSO still performs sunspots counting by the human eye. The reason for this stands with the ambition of not creating a discontinuity in the methodology and keeping the results from different epochs strictly comparable. By their simplicity, human observation presents some advantages that remain valid today such as, low-cost observations that can be performed by many observers allowing a more robust time series; the multiplicity existence of stations that allows extensive cross-validation, thus preventing unrecognized biases that may affect single observatories, whatever their level of sophistication. On the other hand, this traditional method presents some disadvantages since ground observation has its limitations. Despite being stable in the long term, it cannot be considered entirely reliable in short periods. The quality of the observation is fundamentally dependent on the atmospheric conditions, being the analysis impossible to perform in days of strong atmospheric water vapour content and turbulence. Furthermore, human observation also presents some degree of subjectivity inevitably. The main sources of subjectivity are:

- the threshold for the smallest sunspot: the subjectivity range is bounded by the existence of a lower limit for the smallest sunspots, namely 2500km, corresponding to the average size of the solar photospheric granules.
- how groups are split: This factor dominates mainly during high solar activity since groups can be rather complex and densely packed on the solar disk.

To overcome these limitations, SILSO derives the daily sunspot number by averaging the outputs from many observatories around the globe. In this way, it mitigates the subjectivity factors and solves the possible problems of missing data. The daily random disparity among observers behaves similarly to a noise detector, with the systematic bias of an observer (i.e. a tendency to over/under-count, or over/under split) being accounted in the personal reduction coefficient K previously mentioned. The result of this global statistic constitute the International Sunspot Number ($S_n$) [13].

The physical behaviour of the Sun is complex, and there are still numerous exciting aspects that could be here described; however, they are not in the scope of this work. Before concluding this brief heliophysics description, some notions regarding the solar coordinate systems should be introduced, as

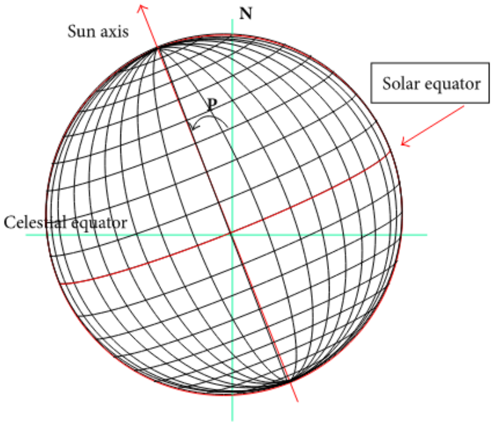


Figure 2.5: Solar rotational angle with respect to the celestial equator. [14]

they constitute a fundamental foundation regarding solar observation and analysis. Defining a single coordinate system that can encapsulate every aspect of the Earth-Sun orbital dance in the most adjustable way is difficult. The Sun is a gaseous body; therefore, there are no fixed points of reference, considering then its differential rotation only make it worse. Furthermore, the Sun's rotation axis is tilted w.r.t the Earth's ecliptic; therefore, the solar north pole appears displaced as seen from Earth, and this displacement is not constant but rather periodic, with the rotational angle varying between +/- 7º as the Earth orbits. Moreover, in order for a solar coordinate system to be considered complete, it must include time. The fact that Earth rotates and revolves raises the need to take these variations into account. The three main categories of solar coordinate system are divided into [15]:

- **Heliographic coordinates**: expresses the latitude and longitude of a feature on the solar surface and can be extended to three dimensions by adding the radial distance from the centre of the Sun. This system has two variations based on the offset difference in the definition of longitude, Stonyhurst and Carrington Heliographic coordinates.

- **Heliocentric coordinates**: expresses the true spatial position of a feature in physical units from the centre of the Sun. There are several well-established systems, being the Heliocentric-Cartesian and the Heliocentric-Radial the ones most used.

- **Projected coordinates**: this class of coordinate systems, also known as helioprojective, mimics the heliocentric coordinate system replacing physical distances with angles. Although there is a one-to-one correspondence between the heliocentric and the helioprojective parameters, the latter is an observer-centric system. For example, if the observer is on Earth, this can be described as a geocentric coordinate system. These systems are celestial spherical coordinates, and Helioprojective-Cartesian and Helioprojective-Radial coordinates are the ones most used.

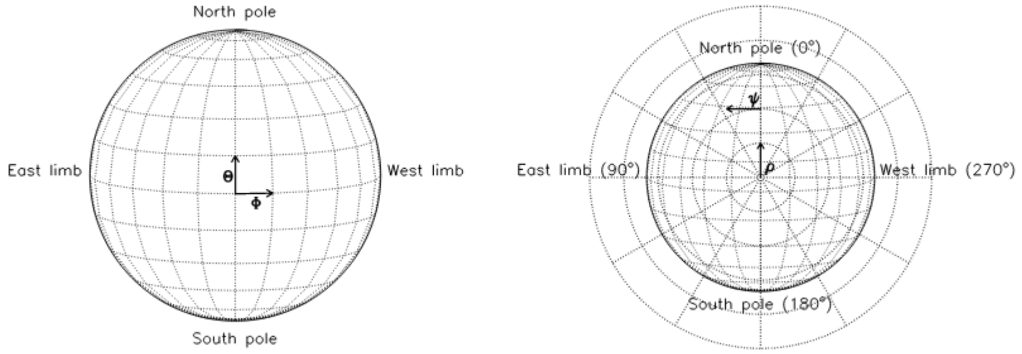


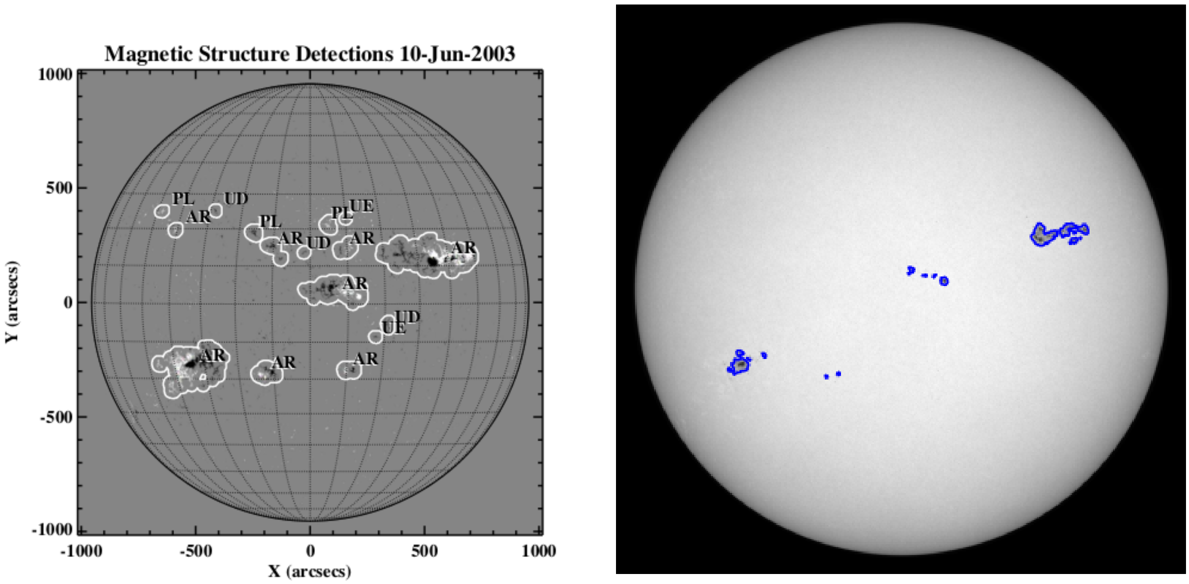
Figure 2.6: **Left**: Heliographic Stonyhurst system. **Right**: Heliocentric-Radial system

## 2.2 Automatic Sunspot Detection

The launch of SOHO in 1995 provided the solar physics community with an unprecedented view of the Sun until then. Before that period, constant surveillance of our star was carried out by ground-based observatories, and space-borne data was therefore a tremendous step regarding the quality of observation. The Solar Dynamics Observatory launched in 2010 came as an improvement of SOHO and with that, an enormous increase of data sent back to Earth, from 250MB daily stream per day from SOHO to 1 TB of data per day from SDO [6]. With SDO returning an equivalent of a 4K image every second, human analysis of the data would require a large team working 24 hours a day, the reason why automated analysis procedures began to be essential. Regarding sunspots analysis, several advances were made to fully automate the procedures for sunspot detection, although nowadays, the field seems to have reached a plateau.

For an inexperienced individual, it seems relatively trivial to detect sunspots and perform handmade drawings of their dark contours, although centuries of controversy in this matter among the solar physics community has proven quite the opposite. The intensity variability nature (and shape) of sunspots give rise to certain degrees of subjectivity, mentioned in section 2.1, making the task of automatic methods even more difficult.

The first approaches were mainly based in **threshold techniques** (Zharkov et al. (2005), Jewalikar and Singh (2010), Dasgupta et al.(2011)), **edge detection** (Zharkov et al. (2005)), **region growing** (Zharkov et al.(2005)) and **mathematical morphology transforms** (Zharkov et al.(2005), Curto et al.(2008)). Also, Nguyen et al.(2005) combined some image processing techniques and clustering methods in white-light images for the recognition and classification of sunspots according to the modi-



(a) SMART

(b) STARA

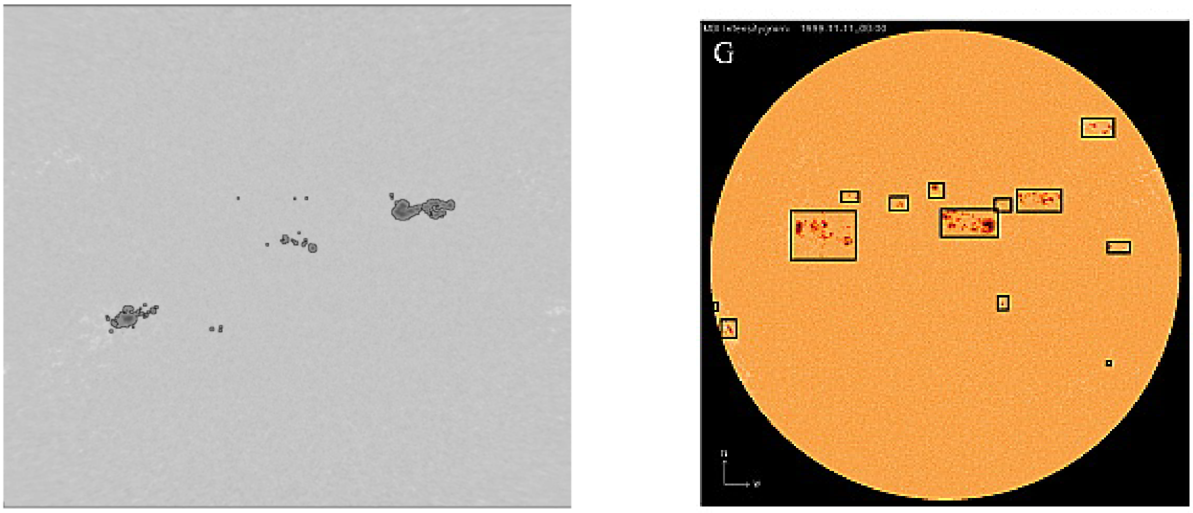
Figure 2.7: Example of SMART and STARA detections. [6]

fied Zurich class of the McIntosh system. Hybrid methods, that include different approaches, have also been developed and can be found in Qahwaji and Colak (2006), Manish et al. (2014) and Shahamatnia et al. (2016). Djafer et al. (2012) adopted wavelet analysis to detect sunspots and the solar limb, performing threshold filtering and Gaussian filtering masks. Another example of a hybrid is the work of Yu et al. (2014), which combines mathematical morphology and region growing techniques to automatic detect sunspots and to differentiate the umbra and penumbra.

The performance of the first approaches was subject of analysis. Zharkova et al. (2005) compared manual procedures results with the automatic methods developed until then and proved the efficiency of the latter. The effectiveness and robustness of these techniques led to the development of automatic algorithms that were able not only to detect sunspots but to extract other important characteristics, such as area and coordinates. The arguably three algorithms that had the most impact were the **Solar Monitor Active Region Tracker** (SMART) Higgins et al. (2010), the **Automated Solar Activity Prediction** (ASAP) Colak and Qahwaji (2009) and the **Sunspot Tracking And Recognition Algorithm** (STARA) Watson et al. (2009), later improved into new versions.

SMART is an algorithm that extracts, classifies and tracks active regions from magnetograms using image processing techniques to determine the boundary of an active region. It uses two consecutive line-of-sight magnetograms that are further smoothed and thresholded to identify potential features. Both detections are overlaid to inspect and remove which features are not present in both magnetograms. Next, the remaining boundaries detection are dilated to create the final one. An example set of SMART detections is shown in Figure 2.7. This algorithm was initially developed to work on data from SOHO-MDI and later adapted for use with SDO-HMI magnetograms, with several new physical property modules being added in the new version.

The STARA algorithm is mainly based on morphological transformations, and it was originally de-



(a) ASAP

(b) ASAP

Figure 2.8: Example of ASAP detections. [6]

veloped for use with SOHO-MDI data but later extended to work with data from SDO and ground-observatories. First, the data is fed to the algorithm, and the image colour space is inverted so that the sunspots appear as bright areas and the disk dark. A morphological top-hat transform is then applied to record the intensity fluctuations on the solar disk to keep sunspots and remove the background. An example of a typical set of detections of STARA is given in Figure 2.7.

The ASAP algorithm is composed of a set of methods for sunspot and active region detection and solar flare prediction based on the work of Colak and Qahwaji. Unlike the other algorithms, ASAP uses quick look (in GIF or JPEG format) images for its calculations. Essentially, the algorithm starts by ingesting continuum-intensity images, applying then image pre-processing techniques such as solar disk boundary detection and limb darkening removal. Next, the heliocentric coordinates are converted to Carrington heliographic coordinates since it provides more accurate detections in the latter coordinate system. Then, morphological transformations and intensity thresholding methods are applied to the newly create images for the final sunspot detection. ASAP was also extended with a tracking algorithm, making use of intersection between objects (e.g. sunspots) in two consecutive records. Although the constant coordinates conversion for each iteration (i.e. new input image) is computationally expensive, the algorithm performs accurate detections even in very small sunspots. Figure 2.8 shows an example of detections performed by ASAP.

In [6] these three algorithms were compared, and it was concluded that they present similar performance as measured in number and area of sunspots, providing also a good agreement with the NOAA numbers and areas, as well as the daily international sunspot number. A common aspect between them is the constant need to incorporate pre-processing techniques before the central processing and feature extraction task, aiming at homogenizing the solar images in terms of dimensions, size and limb darkening removal.

Carvalho et al. (2016) developed a study that aimed to evaluate the capability of some methods in the automatic detection of sunspots when compared to reference detections provided by a solar observer expert. This analysis compared the ASAP algorithm and two original methods developed for the study. One of the methods applied filters to eliminate low frequencies and outliers and increase the contrast of the images. Two thresholds are subsequently applied to segment the umbra and penumbra, followed by an exclusion criteria w.r.t the size and intensity. The other method is based on morphology operators, namely top-hat transform, area-opening and thinning, and it was the approach that presented the best results among the three.

More approaches were made in recent years, Zhao et al. (2016) developed an automatic recognition of sunspots based on ground-based observations, accomplished with a gaussian pre-processing algorithm followed by morphological Bot-hat operation and Otsu threshold. Also, Carvalho et al. (2020) developed recently new approaches to this problem following the previous work of [31], an improvement of a method based on mathematical morphology and a pixel intensity approach. Detailed pre-processing steps were explained, followed by the methodology of each method, and a final comparison based on defined metrics was exposed, having the pixel intensity-based approach a slightly better performance.

The first public deep learning approach based on convolutional neural networks for sunspot detection was developed in [34] (2019). In this work, semantic segmentation for detection followed by clustering techniques was developed for sunspot counting purposes. The results showed that the algorithm could catch the tendency of sunspot numbers over a four-year window quite well, producing an error of 18% compared to the 8% error for the average station in the ISN network. No detection evaluation metrics were reported, although a predicted segmented mask by the developed algorithm presented an excellent approximation to the provided ground truth.

## 2.3 Deep Learning

Deep learning is a sub-field of machine learning re-branded from Artificial Neural Networks (ANNs) that make use of stacks of computational layers to process, extract and learn information from a specific input. Unlike machine learning handcrafted methods, these algorithms are capable of learning complex and hierarchical representations from raw data without the feature engineering steps. This distinctive learning approach is currently state-of-the-art in spatial recognition of patterns, presenting superior performances when compared to traditional methods in tasks such as computer vision. The algorithm developed in this work is based on a specific type of neural networks, namely Convolutional Neural Networks (CNN); thus, here will briefly be described how these networks operate.

### 2.3.1 Neural Networks

Biologically inspired efforts to mimic the way our brain works, these networks are composed of mathematical units called artificial neurons, which receive specific inputs and process them to produce an output, just like a biological neuron. A human nervous system is composed of approximately 86 billion neurons; each of them receives input signals (synapses) from its dendrites and produces an output signal along its axon. Each artificial neuron on the other hand, is essentially a mathematical function with

$m$ inputs that produces a certain output that is sent to the next network layers, which in the simplest form computes:

$$y = a\left(\sum_{j=0}^{m} w_j x_j + b\right) \tag{2.2}$$

where $a$ is an activation function, $w$ is the weight vector for every input $x$, and $b$ is the bias, a parameter that allows the activation function to shift to fit the data better.

**Feedforward neural networks**, also known as **multilayer perceptrons** (MLPs), are the simplest type of ANNs. The goal of a neural network is to approximate some function $f*$, a functions that maps an input $x$ to an output $y$ ($y = f*(x)$), while learning the value of the parameters $\theta$ that result in the best function approximation, simply described in the form $y = f(x;\theta)$. Essentially, a deep neural network model can be thought of as a universal approximator provided with methods to estimate its parameters [35].

The capability of NNs to approximate any complex functions is directly related to the role of the non-linear activation functions. Real-world data is irregular and complex; inserting non-linear units into every neuron provides these networks with a degree of elasticity that enables a highly flexible learning power. Three of the most popular activation functions are the sigmoid, hyperbolic tangent, and the rectified linear unit, which is currently the most used. Such networks typically have three types of layers: the input layer, the hidden layers and the output layer, as shown in Figure 2.9.

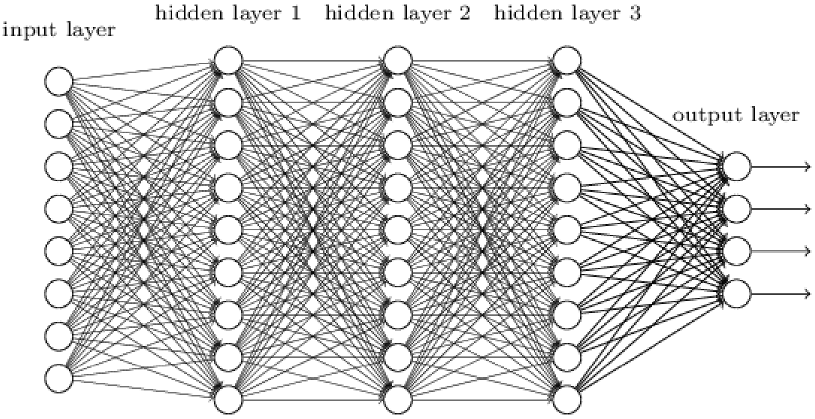


Figure 2.9: Feedforward Fully-Connected Neural Network[9].

The learning procedure of feedforward neural networks considering supervised learning (the learning type employed in this work) can be briefly described as, given input data and the corresponding desired outputs (ground truth), the initial values for all the weights (randomly initialized or following a particular initialization method); the process starts by feeding the model with input data (start of the forward pass), the data is processed by each neuron as it travels through the network, and a measure of the error is calculated in the output layer by comparing the resulting outputs to the ground truth following a defined metric. The goal here is to minimize the error; thus optimization of the network parameters (weights) must be performed. After completing a forward pass and consequently computing the error, the change

[9] `http://neuralnetworksanddeeplearning.com/chap6.html`

in the error following a unit change in weights (i.e., the gradient of the error w.r.t the parameters) is computed using the chain rule for the derivative of a composite function – the loss function (backward pass):

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j} \frac{\partial y_j}{\partial x_i} \tag{2.3}$$

The value of the weights is then updated in the direction that points to a decrease in the output error by means of an optimization algorithm called gradient descent, sliding through the loss surface following:

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial L}{\partial \theta_j} \tag{2.4}$$

where $\alpha$ is the learning rate, $L$ the loss and $\theta$ a weight vector.

The forward pass followed by the backward pass (backpropagation - short for ”backward propagation of the error”) is repeated until the error, evaluated on unseen data, falls below an acceptable value. Following the best practices, a neural network model is trained, validated, and tested in three independent sets: the training set, validation set, and the test set respectively.

### 2.3.2 Convolutional Neural Networks

Convolutional neural networks, also known as CNNs or ConvNets, are a class of neural networks inspired by the human visual cortex, where its architectures make the explicit assumption that the inputs are images, encoding the input information in a more sensible way. These networks differ from simple MLPs by employing spatial convolution operations in at least one of their layers and have neurons arranged in 3 dimensions: width, height, depth.

The convolution operation allows the network to gain spatial knowledge of the input rather than treat an input image as a flat one-dimensional vector, which on the contrary of MLPs, provide locality (i.e., local operations on neighbourhoods) and translation invariance.

The formulation of the convolution operation in machine learning differs from the definition of pure mathematics, corresponding instead to the **cross-correlation** function, which is the same as the classical convolution but without kernel flipping, given by:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n). \tag{2.5}$$

where $I$ is a two-dimensional input image and $K$ is a two-dimensional kernel (also called filter). Essentially, in a convolutional unit, a kernel of a defined size composed of weights slides through an input image, computing the internal product between the two entities in every defined discrete step, as illustrated in Figure 2.10.

[10] https://machinethink.net/blog/convolutional-neural-networks-on-the-iphone-with-vggnet/

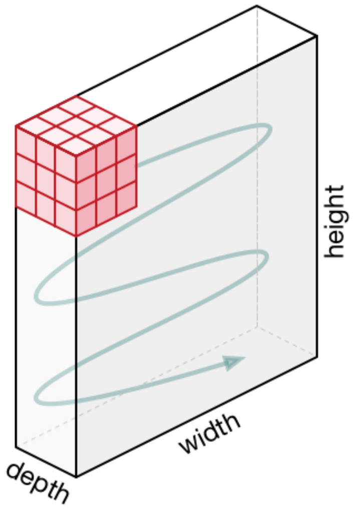


Figure 2.10: Kernel movement through a convolutional unit[10].

## 2.4 Semantic Segmentation

Image segmentation is a process whereby a digital image is subdivided into sets of pixels to be further grouped into similar regions or segments. This task can be performed via different techniques; amongst the most common classical computer vision approaches are thresholding, clustering, histogram-based, and graph partitioning methods. Image segmentation can be generally divided into two categories: semantic segmentation and instance segmentation.

Semantic segmentation is a computer vision task where the goal is to label each pixel in an image accordingly to the class object it represents, resulting in an image that is segmented by classes as illustrated in Figure 2.11. There is no distinction between pixels of the same class; thus, different and separated objects belonging to the same class are treated equally, that is, with no ID attribute.

Despite the popularity of the traditional methods, deep learning came to revolutionize many computer vision problems, being image segmentation one of them, tackling it using deep CNNs.

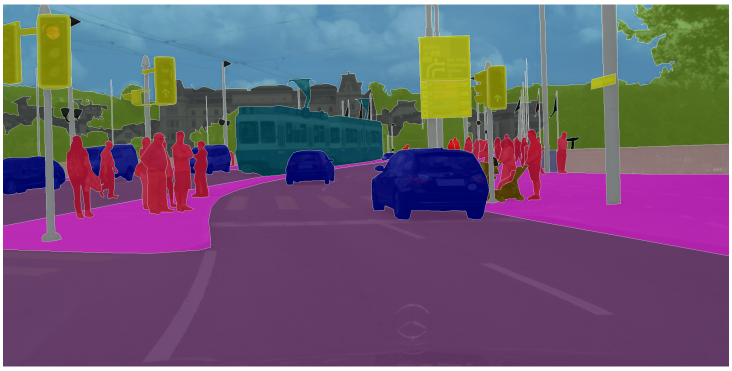


Figure 2.11: Example of a semantic segmented image. [36]

In the context of automatic sunspot detection, as we shall see in the next chapter, the application of CNNs is still to be properly explored, and significant improvements concerning classical methods can certainly be achieved. The CNN architectures introduced here to tackle automatic sunspot detection based on semantic segmentation are the U-Net and U-Net 3+.

### 2.4.1 U-Net

U-Net [37] is an architecture initially developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg. The **U** term derives from the U-shaped architecture of the network, which consists of a symmetric encoder-decoder structure, as shown in Figure 2.12.

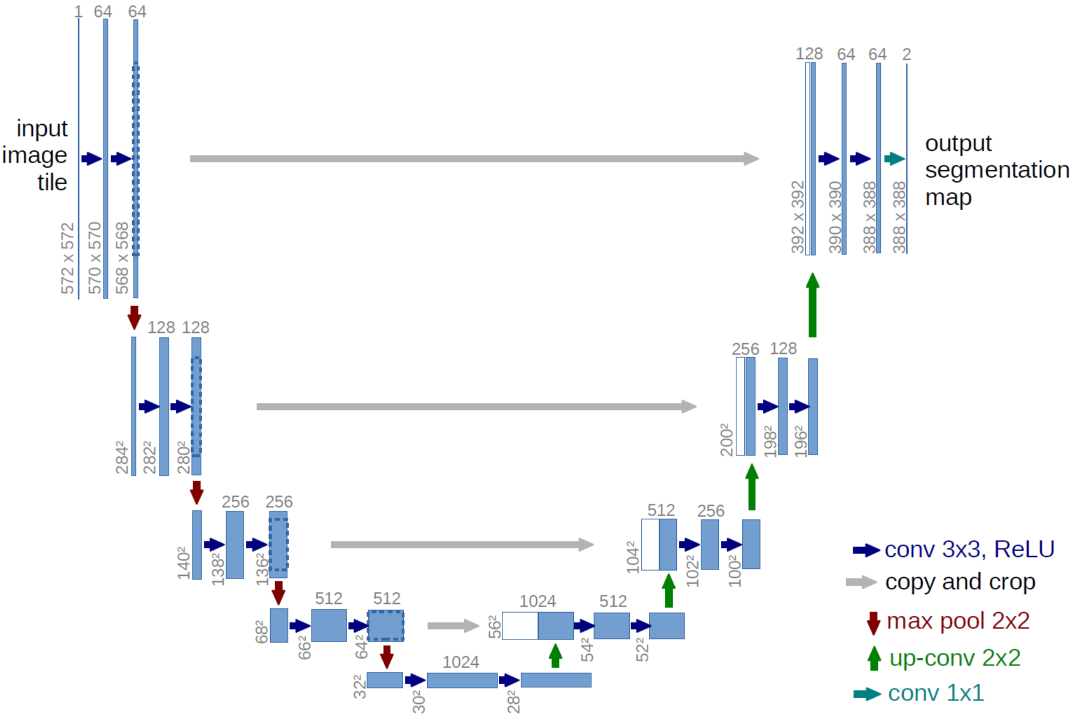


Figure 2.12: U-Net architecture. [37]

Essentially, the encoder is responsible for extracting information from the image, down-sampling it similarly to a standard CNN, and the decoder restores the image dimensions with precise locations for final pixel-wise classification to create a fully segmented image.

The encoding path comprises four blocks; each of them consists of two 3×3 convolution layers followed by ReLU activation with batch normalization and one 2×2 max-pooling layer. As it can be seen in Figure 2.12, padded convolutions are not employed, resulting in a loss of four pixels in every convolution operation. Also, after each pooling, the number of channels are doubled in the next convolutional layer throughout the contracting path.

The significant contribution of U-Net is related to the up-sampling path, where a large number of feature channels are created by up-sampling each feature map using transposed convolutions (also known as deconvolution or up-convolution) with stride 2, concatenating the corresponding cropped feature map from the down-sampling path to get better precise locations. This operation is followed by 3×3 convolution layer and ReLU activation with batch normalization through all four decoding blocks. Finally, in the last operation of the fourth block, a 1×1 convolution layer is applied to reduce the feature channels to the number of classes, and the pixel-wise classification is performed applying a softmax activation $f(s)$

following cross-entropy loss[11]:

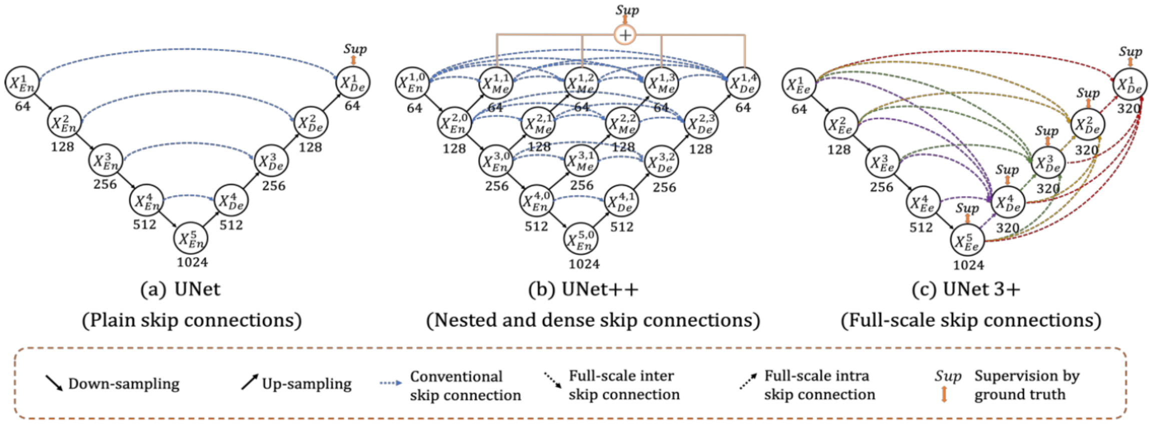
$$L = -\sum_{i}^{B} w_i \log(f(s)_i) \quad , \quad f(s)_i = \frac{e^{s_i}}{\sum_j^B e^{s_j}} \tag{2.6}$$

The use of skip connections to combine the high-level semantic feature maps from the decoder and the corresponding low-level detailed feature maps from the encoder proved to be a great add-on. Since its release in 2015, U-Net has seen a gigantic burst in medical-imaging usage and many computer vision applications, with several new architectures still being developed deriving from it. Among them, U-Net 3+ was chosen to be studied and implemented in this application to sunspot detection.

### 2.4.2 U-Net 3+

Following several U-Net improved networks, U-Net++ [38] came as a modified version where the skip connections were redesigned in a nested and dense manner. Additionally, a deep supervision [39] module was added, enabling the network to operate in two modes, an accurate mode where the output from all resolution segmentation branches are averaged, and a fast mode where only one of the segmentation branches is selected.

Following the latter approach, U-Net 3+ (2020) arises as an improvement by redesigning the inter-connections (skip connections) between the encoder and the decoder as well as the intra-connections between the decoders to capture fine-grained details from the different scale levels, but with fewer parameters to improve the computation efficiency. Additionally, to tackle the over-segmentation (false positives) in non-organ images, a classification-guided module is proposed by jointly training with an image-level classification.


Sup
$X^1_{En}$ 64
$X^2_{En}$ 128
$X^3_{En}$ 256
$X^4_{En}$ 512
$X^5_{En}$ 1024
$X^1_{De}$ 64
$X^2_{De}$ 128
$X^3_{De}$ 256
$X^4_{De}$ 512
$X^{1,0}_{En}$ $X^{1,1}_{Me}$ $X^{1,2}_{Me}$ $X^{1,3}_{Me}$ $X^{1,4}_{De}$
$X^{2,0}_{En}$ $X^{2,1}_{Me}$ $X^{2,2}_{Me}$ $X^{2,3}_{De}$
$X^{3,0}_{En}$ $X^{3,1}_{Me}$ $X^{3,2}_{De}$
$X^{4,0}_{En}$ $X^{4,1}_{De}$
$X^{5,0}_{En}$
$X^1_{Ee}$ $X^2_{Ee}$ $X^3_{Ee}$ $X^4_{Ee}$ $X^5_{Ee}$
$X^1_{De}$ $X^2_{De}$ $X^3_{De}$ $X^4_{De}$ 320
Down-sampling
Up-sampling
Conventional skip connection
Full-scale inter skip connection
Full-scale intra skip connection
Sup Supervision by ground truth

(a) UNet
(Plain skip connections)

(b) UNet++
(Nested and dense skip connections)

(c) UNet 3+
(Full-scale skip connections)

Figure 2.13: Comparison between U-Net, U-Net++ and U-Net 3+ architectures. [40]

[11] This probability calculation and loss function follow the official implementation.

In this work, the deep supervision and classification-guided module were not employed by limited computational resources; therefore, their description will be here omitted.

#### 2.4.2.1 Full Scale Skip Connections

Both U-Net with direct connections and U-Net++ with nested and dense connections do not explore sufficient information from full scales. To tackle this insufficiency, each decoder layer in U-Net 3+ incorporates both same-scale and smaller-scale feature maps from the encoder and larger-scale feature maps from the decoder, capturing fine-grained details and coarse-grained semantics in all scales. To better illustrate how a specific feature map is constructed, Figure 2.14 shows how the third encoder layer ($X^3_{De}$) is built.

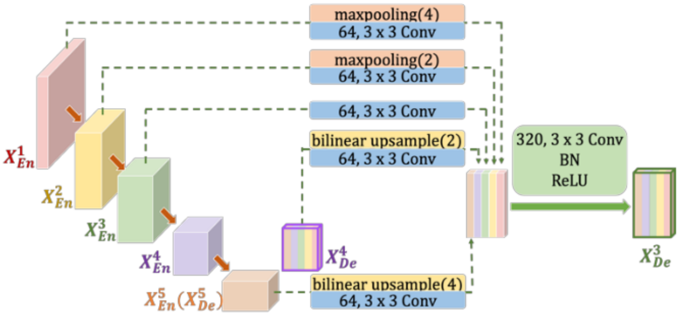


Figure 2.14: Construction illustration of the full-scale aggregated feature map of the third decoder layer. [40]

Similarly to U-Net, the feature map from the same level (scale) from the encoder layer $X^3_{En}$ is received following a 3×3 convolution. Although now, the skip connections from the smaller-scale encoder layers (inter-connections), $X^1_{En}$ and $X^2_{En}$, are also aggregated, delivering the low-level semantic information, by applying non-overlapping max-pooling operation[12]. In addition, the intra decoder skip connections from the higher semantic levels, $X^4_{En}$ and $X^5_{En}$, are concatenated by bi-linear up-sampling. A feature aggregation mechanism is performed on the concatenated feature maps, consisting of a 3×3 convolution with batch normalization and 320 channels following ReLU activation function, to merge all the corresponding semantic information from the five levels. Formally, this operation is postulated as follows, with $i$ being the indexes of the down-sampling path along with the encoder, $N$ the encoder total number of levels, the stack of feature maps represented by $X^i_{De}$ is given by as:

$$X^i_{De} = \begin{cases} X^i_{En}, & i = N \\ H\left(\left[\underbrace{\mathcal{C}(\mathcal{D}(X^k_{En}))^{i-1}_{k=1}, \mathcal{C}(X^i_{En})}_{\text{Scales:}1^{th}\sim i^{th}}, \underbrace{\mathcal{C}(\mathcal{U}(X^k_{De}))^{N}_{k=i+1}}_{\text{Scales:}(i+1)^{th}\sim N^{th}}\right]\right), & i = 1, \ldots, N-1 \end{cases} \tag{2.7}$$

where $H(\cdot)$ is the feature aggregation mechanism that computes the convolution followed by a batch normalization and ReLU activation, $\mathcal{C}(\cdot)$ denotes a convolution operation, $\mathcal{D}(\cdot)$ and $\mathcal{U}(\cdot)$ indicate up and down-sampling operation respective, $[\cdot]$ corresponds to the concatenation.

[12] Note that the first level should go through a bigger max-pooling factor (4) than the second because naturally, the feature map from that level have larger dimensions.

U-Net 3+ without deep supervision and classification-guided module (i.e., only with the full-scale skip connections improvement), presented in [40] around 3.0 dice averaged points over the standard U-Net with the same backbone using focal loss function (and 3.30 points over with), the reason why it was chosen to be also implemented here.

## 2.5 Instance Segmentation

Instance segmentation is a computer vision task for detecting and localizing an object in an image. It comprises two different tasks in a single approach, namely object detection and semantic segmentation. Instance segmentation is still a difficult challenge in computer vision, and over the years, several different techniques have been developed to its effective automation. While in semantic segmentation there is no distinction between objects of the same class, in instance segmentation each object in an image receives one individual ID.

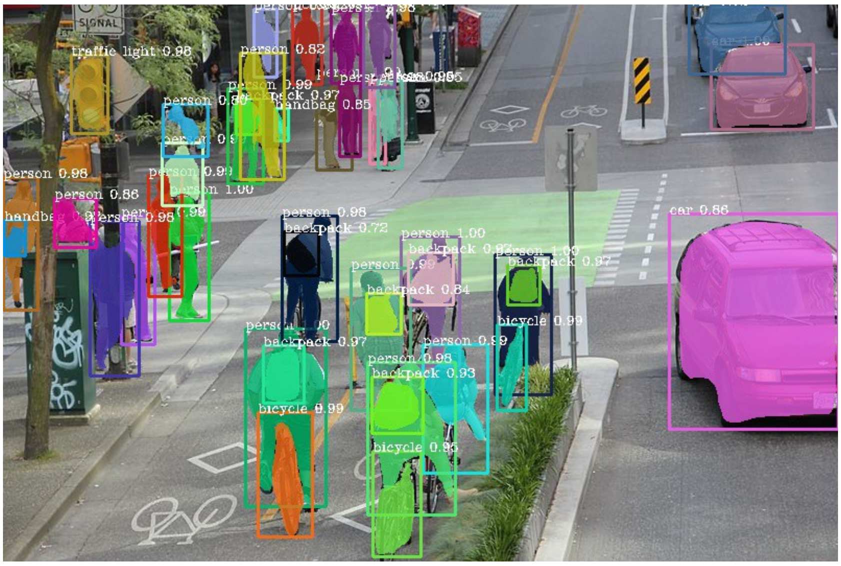


Figure 2.15: Example of an instance segmented image[13].

There are essentially three main approaches to this problem: pixel labeling (semantic segmentation) followed by clustering [41–43] , dense sliding window methods [44–47] and detection followed by segmentation [48–51] – the most popular approach. The algorithm introduced in this work to tackle automatic sunspot group detection based on instance segmentation techniques belongs to the latter category, described in depth next.

[13] https://github.com/ayoolaolafenwa/PixelLib

### 2.5.1 Mask R-CNN

Mask R-CNN [48] became state-of-the-art in object instance segmentation in 2017, winning the best paper award at the International Conference on Computer Vision (ICCV) [52], and it is still one of the most powerful object detectors algorithms. It extends Faster R-CNN [53] by adding a branch for predicting segmentation masks on top of each region-of-interest (RoI), in parallel with the existing branch for classification and bounding box regression.

There are three major modifications from Faster R-CNN to Mask R-CNN: the employment of Feature Pyramid Network (FPN); the replacement of RoIPool with RoIAlign; the introduction of an additional branch to predict segmentation masks - Mask Head. Figure 2.16 depicts the architecture of the network, and it is composed of three main components that can be understood to perform the following tasks:

1. **Backbone Network**: it is responsible for extracting features from the input image at different scales. It uses a feature pyramid network architecture composed of a deep residual network (ResNet) as backbone.

2. **Region Proposal Network**: it generates proposals of possible object bounding boxes from the multi-scale features maps passed by the backbone network.

3. **RoI Heads**: composed of a Box Head and a Mask Head. The former is responsible for object class prediction and bounding box regression from specific features maps of the backbone network in conjunction with proposals boxes from Region Proposal Network. The Mask Head takes the same information as the input and outputs segmentation mask of each RoI.

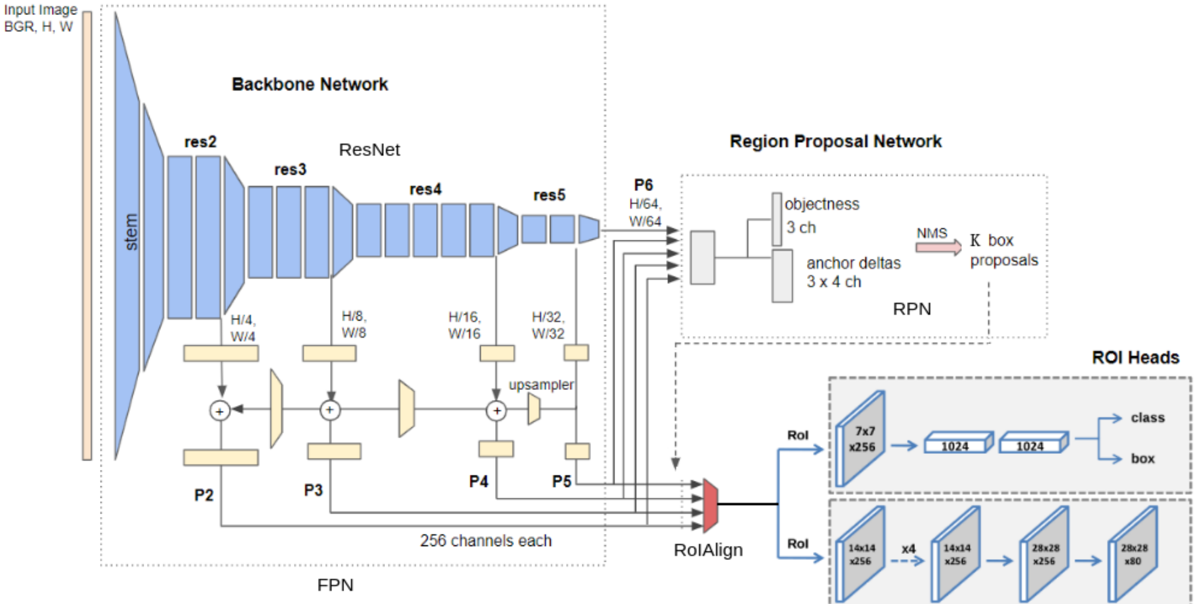


Figure 2.16: Mask R-CNN architecture[14].

[14] Modified from `https://medium.com/@hirotoschwert/digging-into-detectron-2-part-4-3d1436f91266` and [54].

To latter unfold the adjustment procedure to an accurate sunspot group detection, a detailed explanation of the fundamental processes of Mask R-CNN is extended next, as the total understanding of this network is essential for its fine-tune.

**Backbone Network**

The Mask R-CNN backbone network is composed of an FPN architecture with a ResNet backbone to encode features from the input. FPN uses a top-down architecture with lateral connections to build an internal network feature pyramid with several levels from a single scale input. The main advantage of featurizing each level of an image pyramid is that the model is capable of learning multi-scale features representations, therefore leveraging the semantics from low to high levels.

The construction of the pyramid involves a bottom-up and a top-down pathway with lateral connections. The **bottom-up pathway** is the forward pass of the ResNet backbone that is composed with five convolution blocks (in Fig.2.16: stem, res2-res5), henceforward denoted as stages[15], whereby the output of the last layer of each stage is the considered feature map of that stage[16].

ResNet [55] is a widely used encoder that achieved excellent performance results in the ImageNet Large Scale Visual Recognition Challenge 2015 (ILSVRC2015). The main exploit of ResNets is the solving of the famous vanishing gradient problem[17] by introducing a residual block - an identity connection that maps the input of a block to the output making the gradients during backpropagation flow directly through these connections from the later layers to the entrance of the residual block. The original version of ResNet presents several variants, differing in the number of layers (e.g.: ResNet-34, ResNet-101). In [56] (2017) ResNeXt was released, presenting a new architecture constructed by repeating a building block (residual block) that aggregates a set of transformations with the same topology. This strategy employs a new dimension, named cardinality (the size of the set of transformations), which proved to be more effective than going deeper or wider when increasing the network capacity.

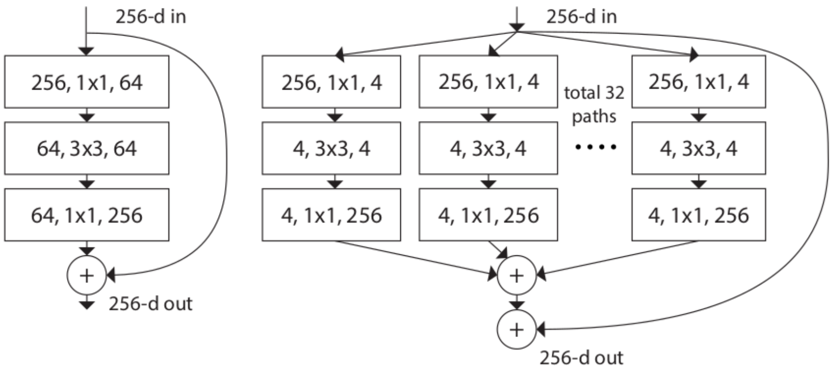


Figure 2.17: **Left**: A ResNet residual block. **Right**: A residual block of ResNeXt with cardinality of 32. [56]

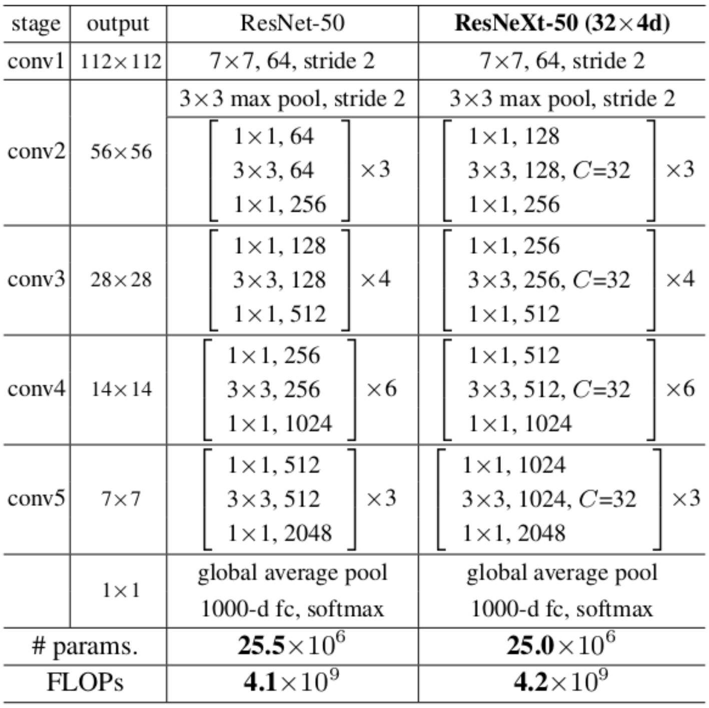
[15] For clarity, stages correspond to the backbone convolutional blocks which are themselves composed by several residual blocks, which number depends on its depth level.

[16] This choice is natural since the deepest layer of each stage should have the strongest features.

[17] Briefly, as deeper a neural network goes, the gradients of the loss functions approaches zero during backpropagation, making the network hard to train.

Figure 2.17 shows a residual block comparison between the original ResNet and a ResNeXt of the same stage. Note that instead of having a single convolutional path with 3 operations, there are 32 paths (cardinality dimension) where the number of features channels is just 4 instead of 64 in each one of them (denoted as **32**×**4d**), having the same kernel size. These 32 paths are then concatenated after the last operation of each residual block (note that each residual block is represented as a blue rectangle in Figure 2.16.)

In Figure 2.18 is shown the ResNet-50 architecture alongside a ResNeXt version of the same depth. With, **conv2-conv5** representing the **res2-res5** of the Mask R-CNN backbone[18]. Note that the indicated output dimensions do not correspond to this application case of Mask R-CNN. Illustrated is an example case to a $224^2$ input, hence a 112×112 conv1 output.



| stage | output | ResNet-50 | **ResNeXt-50 (32×4d)** |
|---|---|---|---|
| conv1 | 112×112 | 7×7, 64, stride 2 | 7×7, 64, stride 2 |
| conv2 | 56×56 | 3×3 max pool, stride 2 | 3×3 max pool, stride 2 |
| | | $\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 128\\3\times3, 128, C{=}32\\1\times1, 256\end{bmatrix}\times3$ |
| conv3 | 28×28 | $\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$ | $\begin{bmatrix}1\times1, 256\\3\times3, 256, C{=}32\\1\times1, 512\end{bmatrix}\times4$ |
| conv4 | 14×14 | $\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$ | $\begin{bmatrix}1\times1, 512\\3\times3, 512, C{=}32\\1\times1, 1024\end{bmatrix}\times6$ |
| conv5 | 7×7 | $\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$ | $\begin{bmatrix}1\times1, 1024\\3\times3, 1024, C{=}32\\1\times1, 2048\end{bmatrix}\times3$ |
| | 1×1 | global average pool<br>1000-d fc, softmax | global average pool<br>1000-d fc, softmax |
| # params. | | $\mathbf{25.5}\times10^6$ | $\mathbf{25.0}\times10^6$ |
| FLOPs | | $\mathbf{4.1}\times10^9$ | $\mathbf{4.2}\times10^9$ |

Figure 2.18: **Left**: ResNet-50. **Right**: ResNeXt-50 with a 32×4d structure. Inside each bracket are the shape of a residual block, and outside the bracket is the number of stacked blocks on a stage. $C$ corresponds to the cardinality. [56]

Each residual block from Figure 2.17, have their number of parameters equal to:

$$C \cdot (256 \cdot d + 3 \cdot 3 \cdot d \cdot d + d \cdot 256), \tag{2.8}$$

yielding therefore a similar number of parameters for the original version block and ResNeXt (**32**×**4d**).

The FPN **top-down pathway** starts from the last stage (which contains the smallest feature maps) downwards, merging the larger feature maps by upscale operations with a factor of 2 along with the lateral connections. This up-sample is achieved by first applying 1×1 convolutions to bring down the

[18] Note that the last stage is naturally not included in Mask R-CNN, which represents the classification layer; and stem correspond to conv1.

number of channels to 256, up-sampling then using the nearest neighbour method. Then, the feature map is merged with the corresponding bottom-up map (which also was previously subjected to a 1×1 convolutional layer) by element-wise addition as illustrated in Figure 2.19. Finally, each stage merged feature map undergoes a 3×3 convolution to essentially reduce the aliasing effect[19] of up-sampling, to produce the final set of features maps called $\{P_2, P_3, P_4, P_5, P_6\}$.

It should be noted that although the top-down pathway possesses strong semantic properties of the input, the positions of that features are not sufficiently accurate since they have been subjected to several down-sampling and up-sampling operations. Thus, the major role of the lateral connections is to directly pass more precise locations of features from the finer levels of the bottom-up maps to the top-down maps.

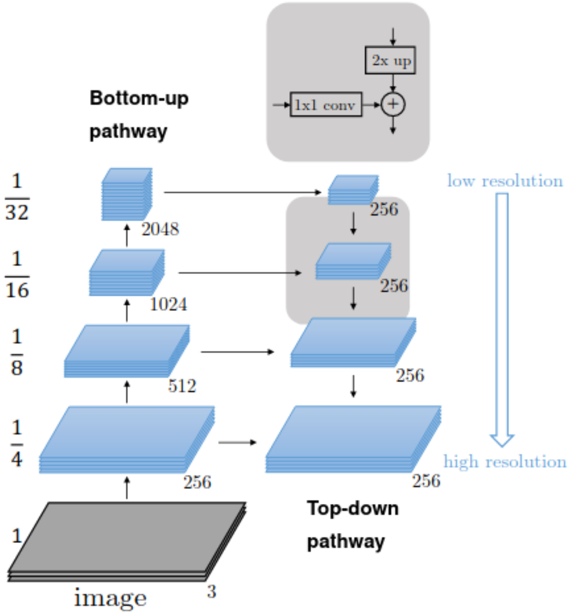


Figure 2.19: Feature Pyramid Network architecture. The gray patch represents the lateral connections[21].

### Region Proposal Network

The second module of Mask R-CNN is the Region Proposal Network (RPN). RPN can be thought of as a sliding-window class-agnostic object detector. The general idea of this module is that a small sub-network is evaluated on 3×3 sliding windows on top of each FPN feature map, performing an object/non-object classification and bounding box regression in parallel. Both tasks are computed w.r.t to a set of anchor boxes generated at each sliding window location. The most promising proposals are then sent to the RoI Heads. The RPN structure is illustrated in Figure 2.20.

This network was first proposed in [53], in which a pyramid of anchors varying in size and aspect ratio is generated at each slide through the last shared convolutional feature map from the backbone. In [57] this approach is modified to leverage from the multi-scale feature maps from FPN, assigning thus one

[19] Appearance of jagged edges.
[21] Modified from `http://presentations.cocodataset.org/COCO17-Stuff-FAIR.pdf`
[22] `https://medium.com/@hirotoschwert/digging-into-detectron-2-part-4-3d1436f91266`

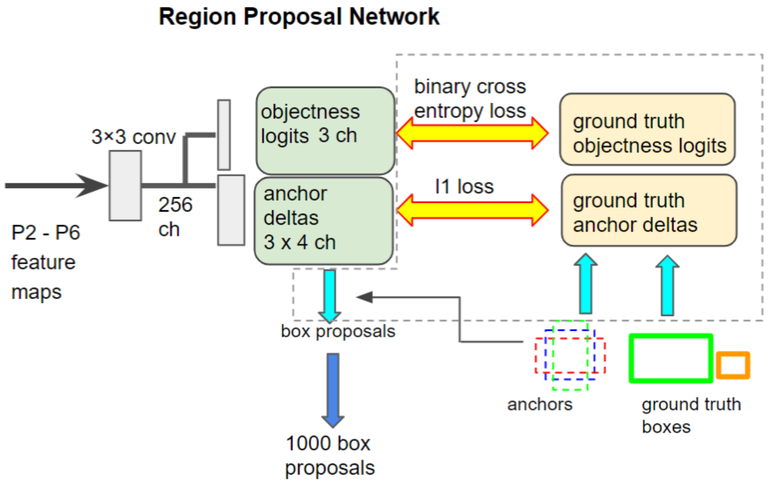


Figure 2.20: Region Proposal Network Structure[22].

specific anchor box size for each level respecting the same aspect ratio strategy {1:2, 1:1, 2:1}. In each sliding window location, are placed a total of 15 anchors boxes (3 per level), which undergoes a 3×3 convolutional layer with 256 channels followed by two siblings 1×1 convolution branches that compute for classification and regression, denoted henceforward as **RPN Head**, depicted in Figure 2.21.

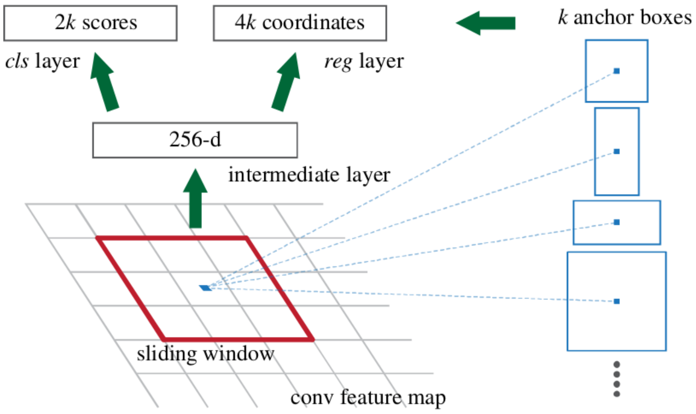


Figure 2.21: Region Proposal Network Head and anchor box placement. [53]

The **anchors generator** insert anchor boxes on the five FPN feature maps independently, and these locations correspond to grid points in the input image which are naturally dependent of the stride ({4,8,16,32,64}) from which the feature map resulted, and from $P_2$ to $P_6$ are originally assigned the anchor sizes {$32^2$,$64^2$,$128^2$,$256^2$,$512^2$} respectively.

After processing RPN Head, it is time to compute and evaluate the proposals. Thus, the ground-truth boxes are loaded, and training labels are assigned to each anchor based on their Intersection-over-Union (IoU) ratio with ground-truth bounding boxes. Formally, an anchor is considered a foreground box if it has the highest IoU for a given ground-truth box or an IoU over 0.7 with any ground-truth box, considered background if it has an IoU lower than 0.3 for all ground-truth boxes, and ignored otherwise.

It should be stressed that the ground-truth boxes are not explicitly assigned to levels considering their dimension but rather associated with anchors boxes, which have been assigned to pyramid levels.

Next, the regression coefficients are computed, adopting the parameterization of the 4 coordinates following:

$$\begin{aligned} t_x &= (x - x_a)/w_a, & t_y &= (y - y_a)/h_a, \\ t_w &= log(w/w_a), & t_h &= log(h/h_a), \\ t_x^* &= (x^* - x_a)/w_a, & t_y^* &= (y^* - y_a)/h_a, \\ t_w^* &= log(w^*/w_a), & t_h^* &= log(h^*/h_a) \end{aligned} \tag{2.9}$$

where $x$, $y$, $w$ and $h$ represent the boxes centre coordinates and its width and height, respectively. The variable $x$ belongs to the predicted box, $x_a$ the anchor box, and $x^*$ the ground-truth box (same for $y$,$w$,$h$). This can be thought of as bounding-box regression from an anchor box to a nearby ground-truth box, and these regression coefficients translate to deltas between boxes that the network should learn.

The anchor boxes are then re-sampled accordingly to the training strategy before ongoing loss calculations. The reason for this is because the number of background boxes are dominant, which would result in significant imbalances further that would complicate the learning of foreground cases.

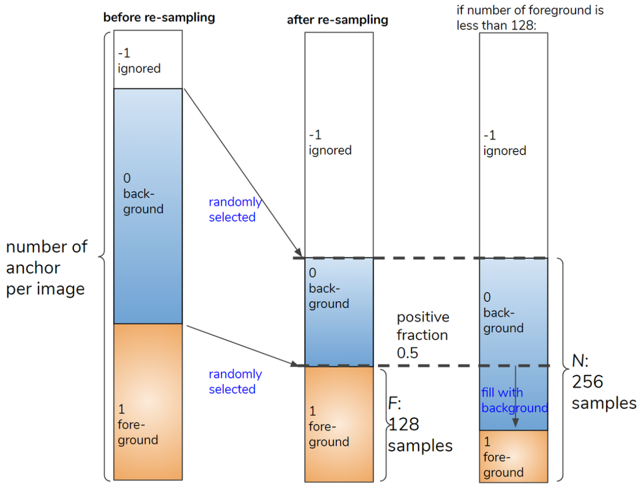


Figure 2.22: Re-sampling procedure[23].

This procedure starts by randomly selecting background and foreground proposals up to defined quantity, respecting a particular portion of desired positive boxes (0.5). Then, if the number of foreground boxes does not reach the desired respective samples, the missing amount is filled with background boxes, as illustrated in Figure 2.22.

Finally, the RPN loss is computed, defined as the sum of both objectness loss and localization (regression) loss:

$$L_{RPN} = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{loc}} \sum_i p_i^* L_{loc}(t_i, t_i^*), \tag{2.10}$$

where $i$ is the index of an anchor, and $p_i$ is the predicted probability of anchor $i$ being an object (in

[23] https://medium.com/@hirotoschwert/digging-into-detectron-2-part-4-3d1436f91266

this case – sunspot group). The ground-truth label $p_i^*$ is 1 if the anchor is positive, and 0 otherwise. $t_i$ represents the vector of the parameterized coordinates of the predicted bounding box (from RPN Head), and $t_i^*$ is that of the ground-truth box associated with a positive anchor.

The classification loss is computed as the cross-entropy loss over two classes (object $vs.$ not object), defined as:

$$L_{cls} = -\sum_{i=1}^{N} y_i \, \log(p(y_i)) \tag{2.11}$$

For the localization loss is used the smooth L1 loss, which is defined over a tuple of regression coefficient of the ground-truth bounding boxes $t^* = (t_x^*, t_y^*, t_w^*, t_h^*)$ and the predicted boxes $t = (t_x, t_y, t_w, t_h)$:

$$L_{reg}(t_i^*, t_i) = \sum_{i \,\in\, x,y,w,h} \text{smooth}_{L1}(t_i^* - t_i), \tag{2.12}$$

in which

$$\text{smooth}_{L1} = \begin{cases} 0.5\,x \ , & \text{if } |x| < 1 \\ |x| - 0.5 \ , & \text{otherwise} \end{cases} \tag{2.13}$$

The term $p_i^* L_{loc}$ means the regression loss is activated only for positive anchors $(p_i^* = 1)$ and disabled otherwise. The outputs of each contribution ($cls$ and $loc$) consist of $\{p_i\}$ and $\{t_i\}$ respectively. It should be noted that the localization loss is only defined to foreground anchors and the classification loss to either foreground anchors and background anchors, being the others neglected from training purposes as previously mentioned, which naturally lead to a different contribution of both tasks, being normalized on $N_{loc}$ and $N_{cls}$ and weighted by a balancing parameter $\lambda$. In [53] the ablation experiments show that the results are slightly insensitive to a wide range of $\lambda$, which is by default set to 10 to approximately equalize both contributions.

Next, the last stage of RPN is reached, where the most promising proposal boxes are selected to be sent to the RoI Heads. Firstly, the computed regression coefficients are then applied to the generated anchor boxes to prune them. This operation decodes the predictions by ”unparameterizing” and offsetting them to the image. Anchor boxes that cross image boundaries are removed during training to avoid large error terms that may prevent the objective function from converging. During inference, these boxes are just clipped to the image boundary.

Then, the predicted boxes are sorted by their predicted objectness scores at each stage (FPN levels), and the top-$K$ (denoted as pre-NMS top-$K$) scored boxes (2000 by default) are chosen from each FPN level. Naturally, there is a significantly greater amount in P2 than the rest of the levels; if some of them have resulted in a number of anchors (which respected the standards previously defined) below $K$, all the boxes of that individual level are selected.

Lastly, Non-Maximum Suppression (NMS) is applied at each level independently (with a threshold of $0.7$ by default) to remove the usual significant amount of overlapping boxes, and a new top-$K$ (post-NMS top-$K$) proposals is selected from all the levels together ($1000$ as default). This final selection represents the number of proposals that are processed in the last Mask R-CNN components.

### RoI Heads

The RoI Heads is where the object class prediction, final bounding box regression and binary segmentation mask predicted are computed for each region of interest. This last module takes as input the features maps from FPN, the proposal boxes from RPN and the ground truth boxes, as depicted in Figure 2.16.

**1) Proposal Box Sampling**

This first step is applied only during training, similar to what is proceeded in the previous module. From RPN reaches the top-$K$ proposals boxes and the ground-truth boxes are added as input as well to the predicted proposals to accelerate training convergence[24]. The foreground and background proposals boxes are firstly re-sampled to balance the training objective. The proposals that have higher IoU than a newly defined threshold (0.5) with the ground-truth are counted as foreground, otherwise as background, here there are no ignored boxes, in contrast to what happens in RPN, and a new positive ratio between the two is used to balance them accordingly to the training objective considering a specific batch size also defined[25]. If the foreground boxes do not reach the desired amount, background boxes are filled in with the remaining quantity.

**2) FPN-RoI Mapping**

As shown in Figure 2.16, the features maps from each FPN level will have a particular size w.r.t the input image, differing following their deepness level. To the RoI Heads only the $P2$ to $P5$ levels are fed as input, leaving therefore $P6$ out of this module.

This step aims to associate the appropriate FPN feature map to a particular RoI based on its area. For example, the $P5$ level possesses semantic features more appropriate for larger objects than the $P2$ level that can detect smaller objects more accurately. This mapping is proposed on [57], which associate proposal boxes to the appropriate feature map, following:

$$k = \left\lfloor k_0 + log_2(\sqrt{wh}/224) \right\rfloor \tag{2.14}$$

where $k$ indicates the level of assignment, 224 is the canonical ImageNet pre-training size (where our backbone was pre-trained on), and $k_0$ the canonical level - the feature map level index where a canonically-sized box with $wh$ equal to $224^2$ should be mapped into, which is equal to 4 in this case. If the level of assignment is lower or greater than the minimum or maximum level ($P2, P5$), the value is clamped to the closer limit. For example, a box size of $512^2$ is assigned to $P5$ level.

**3) RoI Align**

As previously mentioned, one of the significant changes of Mask R-CNN architecture from Faster R-CNN is the replacement of the RoIPool layer with RoIAlign. RoIPool is an operation to extract feature maps from the proposed RoIs of non-uniform sizes (7×7 originally for box head). Briefly, RoIPool starts by quantizing a RoI, which is then divided into spatial bins, which are themselves quantized, and lastly,

[24] For example, if the input image presents three sunspots groups (instances) and $K$ is 1000, the total number of proposals will be 1003.

[25] For instance, using a batch size of 300 proposals from the top-1000 with a positive fraction of 0.25, would result in 75 positive boxes and 225 negative boxes.

feature values covered by each bin are aggregated by max-pooling. The problem associated with these operations in an instance segmentation application is that quantizations introduce misalignments between the RoI and the extracted features. While this may not significantly affect classification, which is robust to small translations, it has a sufficient negative impact on predicting pixel-accurate masks.

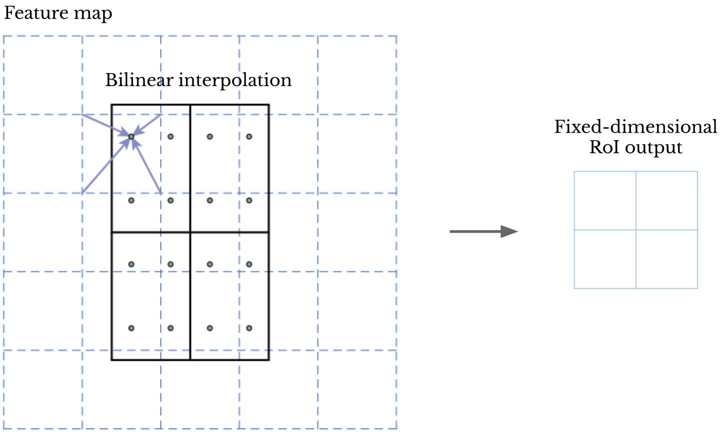


Figure 2.23: RoIAlign operation[26].

The RoIAlign layer was originally proposed to address this issue, which withdraws the harsh quantization effect of RoIPool by adequately aligning the extracted features with the input. It uses bilinear interpolation to compute the exact values of the input features at four equally sampled locations in each RoI bin as illustrated in Figure 2.23.

Note that this procedure happens independently for each head, thus creating two feature maps for each RoI to be fed to the box and mask heads. Once it is completed, the computations for each head take place.

**Box Head**

Finally, we reach one of the final branches of Mask R-CNN. Here, each positive RoI is classified into a respective class, and its box position and shape is pruned. The architecture of this head is essentially similar to what was postulated in [53, 58], illustrated in Figure 2.24 in the upper diagram. Each RoI is first flattened to be passed through two fully connected layers yielding two output tensors, a classification score tensor ($[B, N_{classes}+1]$)[27] and a bounding box prediction tensor ($[B, N_{classes} \times 4]$)[28], being $B$ the batch size and $N_{classes}$ the number of classes.

During training, the losses w.r.t the final output tensors are computed similarly to what was defined in section 2.5.1, the only difference is related to the classification loss where now the probability classification is w.r.t to each class instead of object/not-object.

During inference, the regression coefficients are applied to the proposal boxes to specify how to deform each box, the boxes that cross the input image are cropped to the respective boundary, and NMS is applied to remove the usual high amount of overlapping boxes over a defined threshold, done

[26] Modified from [48].
[27] +1 counts for the background.
[28] 4 corresponds to the bounding box variables.

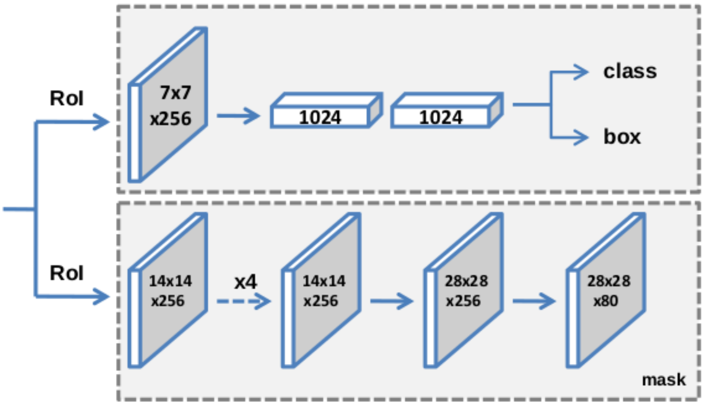


Figure 2.24: RoI Heads. In the upper and bottom branch are the Box Head and Mask Head, respectively.

separately for each class. Then the final boxes are filtered accordingly to a defined number of maximum detection per image.

**Mask Head**

Parallel to the box head computations, the mask head is also processed. The architecture of this branch is composed of an FCN with four 3×3 convolutional layers maintaining the number of channels and dimensions, followed by an up-sample layer (transposed convolution) that is further subjected to a 1×1 convolution layer in order to bring down the number of channels to the number of classes to make the predictions.

The masks branch has a final $Km^2$ dimensional output for each positive RoI, which encodes $K$ binary masks resolutions of $m \times m$, one for each class. It should be noted that $L_{mask}$ is defined for each positive RoI similarly to Box Head with a defined threshold of 0.5, although here on pixel level, if over the threshold the pixel is assumed to belong to the object otherwise not. Thus finally, is applied a per-pixel sigmoid activation to compute then the average binary cross-entropy loss, defined as:

$$L_{mask} = -\frac{1}{K} \sum_{i=i}^{K} \sum_{j=1}^{m \times m} (\log P_{i,j}^{M}). \tag{2.15}$$

where $P_{i,j}^{M}$ is the $j$-th pixel of the $i$-th generated mask. The generated mask is then submitted to post-processing. The mask tensor is padded to avoid boundary effects caused by the earlier up-sample operation, and the bounding box coordinates (w.r.t the input image) are re-scaled[29] according to the new mask and converted to the nearest integer. The mask is also re-scaled according to the input image size using bilinear interpolation and finally superimposed in the image.

Concluding, the multi-task loss of RoI Head is then defined as:

$$L_{RoI} = L_{cls} + L_{box} + L_{mask} \tag{2.16}$$

with the Mask R-CNN total loss given by the sum between $L_{RPN}$ and $L_{RoI}$.

[29] Note that the box regression output tensor is w.r.t the feature map dimension.

## Chapter 3

# Methodology

### 3.1 Dataset

The colossal quantity of astronomical data recorded during the last 100 years is the perfect ingredient to machine learning algorithms, as better they perform as more data they have to ingest. The variety of applications in astronomy is immense, and to further push the boundaries of our capabilities to analyse and extract insightful information from this vast amount of data in a faster and reliable way, these algorithms come to our aid.

There are different sources and types of solar data; for graphical images, it can be essentially divided into ground-based and space-borne data, being the latter which provides clearer recordings as they are not challenged by atmospheric adversities. The Debrecen Heliophysical Observatory (DHO) provides the most consolidated sunspot datasets, being currently the Debrecen Photoheliographic Data (DPD) and the Helioseismic and Magnetic Imager Debrecen Data (HMIDD) the most detailed ground-based and space-borne sunspots catalogues respectively [59]. Both provide area and position data for each observable sunspot, performed by a software developed to the purpose that has been improved over the years named Sunspot Automatic Measurement (SAM) [60, 61], which previsions are later revised and corrected by solar experts.

Table 3.1: DHO available sunspot revised catalogues summary.

| | DPD | HMIDD |
|---|---|---|
| Data Range | 1974-2015 | 2010-2014 |
| Cadence | 1 day | 1 hour |
| Type | ground-base | space-borne |

The DPD sunspot catalogue is compiled as a continuation of Greenwich Photoheliographic Results (GPR)[62] and have available ground-based revised records from 1974 to 2015 [59]. HMIDD, on the other hand, provides hourly data from 2010 to 2014, with 2015 to 2017 (daily data) still in a preliminary state, all recorded by HMI aboard SDO. DPD dataset is constituted mainly with full-disc observations taken at DHO and its Gyula Observing Station, however, in days of bad atmospheric conditions, the best observation recorded among the global observatories network is chosen. This aspect, unfortunately, leads to a non-homogeneous structure of the image data, since depending on the observatory, they might differ in resolution (from $4096^2$ to $8192^2$ pixels), sometimes with limb darkening not corrected and the disk not centred in the plate, which oblige an additional pre-processing. An example image from the Mitaka Observatory (Tokyo), together with its sunspot annotations provided by DPD in visual format, is shown in Figure 3.1.

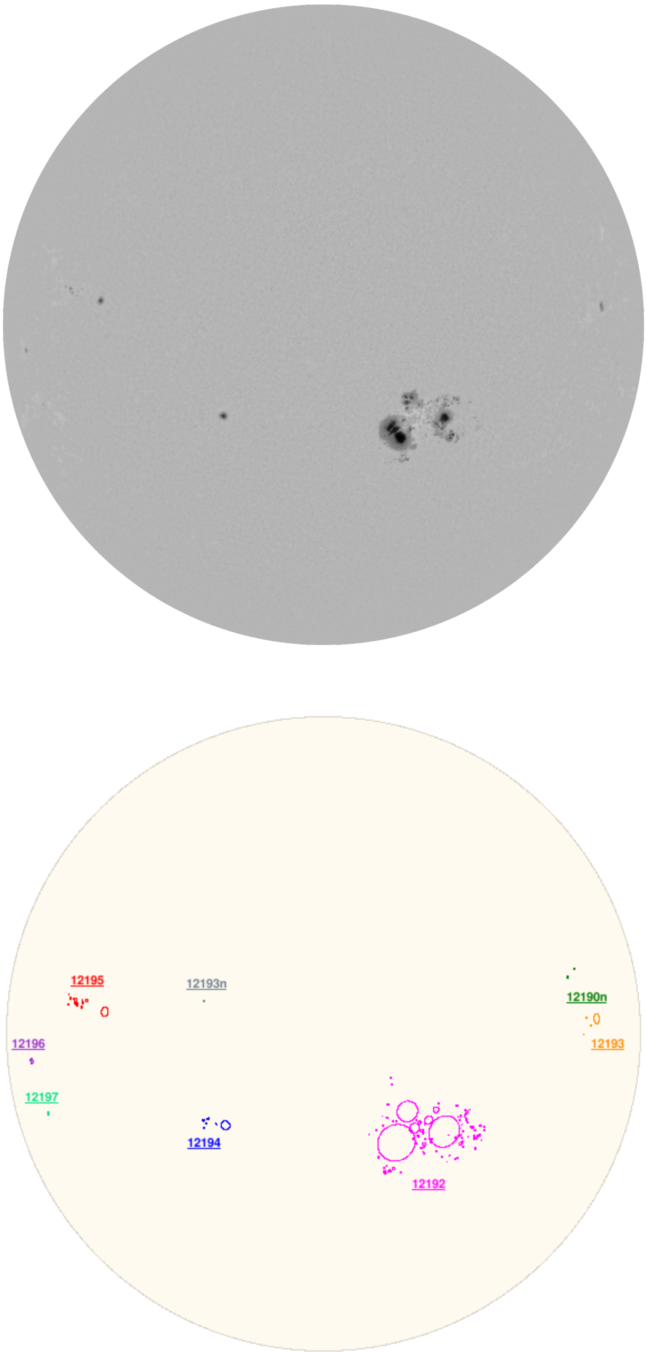


Figure 3.1: **Top**: Full disk white-light observation from the Mitaka Observatory (25/10/2014).
**Bottom**: Annotated mask of the white-light observation provided by DPD with NOAA group numbers.

Although DPD presents a visual representation of their catalogues, the sunspots annotations are provided in text format, indicating its coordinates, area, and other fields for each spot individually, at a specific time. Thus, in order to have our ground-truth data in the desired state when fed to the algorithms, they must be first pre-processed. For that, binary segmentation masks need to be generated from these sunspots catalogues. As one goal was to build a larger enough dataset for this implementation, the first idea was to leverage from the ground-based catalogues conjugated with space-borne images in the available years, but the necessary procedure raises some critical issues.

Considering the Earth-Sun distance and the SDO orbit, the average angle SDO$\sphericalangle$Sun (from the ecliptic) and a visible observatory (VO)$\sphericalangle$Sun can be assumed to be the same by approximation; hence the annotations made from ground-based data at a specific time instant should be valid in the space-borne image at the same instant. So first, the DPD catalogues of each available SDO year (2010-2018) were downloaded, concatenated, and properly handled. Then, since the catalogues are made at specific instants not necessarily equal to the SDO available continuum-intensity images (white-light) that have a time cadence of 720 or 45 seconds depending on the available series (with or without limb darkening removal, respectively), the nearest image w.r.t the catalogue time was downloaded. The series with limb darkening removal was selected, and the download was conducted using a Python library called Sunpy [63] that connects to the Joint Science Operations Center (JSOC)[1] API by a JSOC's Client that query data and make export requests from it. It was selected only the nearest image for each daily record to perform the crucial next step with the most accuracy possible, the coordinate system rotation. The idea here is that since the differential rotational period of the Sun is known, it is possible to rotate one coordinate at a particular instant to the respective location at another instant. This is achieved by first converting the Carrington Heliographic coordinates of each sunspot into the Helioprojective coordinate system, and using SunPy, a map is built from the FITS file (image format provided), making it possible to leverage from the built-in functions of that class object, being one of them the calculation of where an Helioprojective coordinate maps to after a time period delta, taking into account the differential rotation profile of the Sun.

After the rotated coordinates match the time instant of the respective SDO image, the segmentation masks can be generated. A recursive flood fill algorithm conducts this procedure. First, a seed is placed at the centre of a sunspot; then since we are in possession of the area of the specific spot, it is possible to find the rest of the pixels that belong to the spot by checking their intensity, the darker surrounding pixels are selected orderly until the area of the sunspot is reached. This procedure is described in Algorithm 1 and works quite well; the only problem associated with it is related to the previous step.

By superimposing the resultant mask in the original image, it was found that there were small misalignments in the more minor spots, where the seed was not inserted inside the spot, and therefore the flood fill algorithm was starting its expansion outside the same, which led to visible critical issues in the generated masks, as shown in Figure 3.2. Due to the inertia of sunspots and the unpredictable magnetic tubes perturbances of the photosphere, some spots tend to deviate slightly from the theoretical differential rotational profile of the Sun's surface, hence its predicted location can be slightly shifted.

[1] Official data source of the Solar Dynamics Observatories data products.

**Algorithm 1** Ground-truth masks generation procedure

```
Input: sunspot pixel coordinates ($ss\_coord$), sunspot area ($wsa$), full-disk image ($image$), number of pixel in solar disk ($n\_dpx$), disk mask set ($disk\_mask$), zeros mask array ($mask$)
Output: binary segmentation mask ($mask$)
procedure GENERATE BINARY MASK(input)
    $ws\_mask$ = $\{\}$    ▷ whole spot mask set
    for each sunspot in $image$ do
        $centre$ = get_disk_centre($image$)    ▷ coordinates of disk's centre
        $distance$ = get_distance($centre$, $ss\_coord$)    ▷ pixel euclidean distance
        $ss\_num\_px$ = convert_area_to_pixel($centre$, $distance$, $wsa$, $n\_dpx$)    ▷ convert spot area in millionths of solar hemisphere to pixel
        $o$ = 3    ▷ half pixel-window size
        $window$ = get_pixel_window_around_sunspot($image$, $ss\_coord$, $o$)    ▷ create a pixel window centered in spot coordinate expanding in both directions by $o$ amount
        $low$ = get_window_lowest_pixel_index($window$)    ▷ get darker pixel index
        $new$ = get_low_coordinate($ss\_coord$, $o$, $low$)    ▷ get darker pixel coordinate
        $ws$ = [ ] , $candidates$ = [ ]    ▷ whole spot area; coordinate candidates
        $exp\_rate$ = 1    ▷ pixel expansion rate
        while len($ws$) $<$ len($ss\_num\_px$) do
            expand = create_expansion_window_around_new($new$)    ▷ create a 3×3 window for each new incoming $new$
            for i in set($expand - ws$) do
                $candidates \leftarrow i$
            end for
            $new$ = priority_queue_get_smallest($candidates$, $exp\_rate$, $image$)    ▷ retrieve the coordinates from the candidate that presents the smallest pixel value
            $ws \leftarrow new$
            for n in $new$ do
                remove_n_from_candidates($n$, $candidates$)
            end for
        end while
        $ws\_mask \leftarrow ws$
        for c in set.intersection($ws\_mask$, $disk\_mask$) do
            $mask[c] = 1$    ▷ to every sunspot pixel assign a pixel value of 1 (white)
        end for
    end for
    return $mask$
end procedure
```

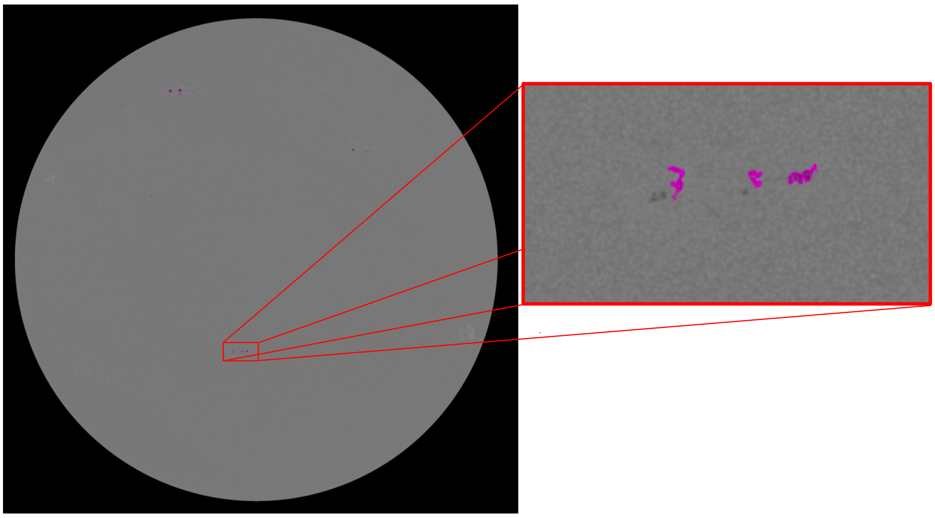


Figure 3.2: Example of contours anomalies detected in the ground-truth masks generation from ground-based catalogues.

To minimise this issue, the interval between the image recording and the catalogue time must be as shorter as possible. The SDO series with the smallest cadence (45 seconds) were retrieved from JSOC[2] to address this problem, but now it is necessary one intermediate step before jumping into the masks generation, the limb darkening removal. As described in section 2.1, this event creates a visual effect whereby the centre looks brighter than the limb and to make the spot areas expansion near the limb, it should be first corrected, otherwise leads to consistent anomalies every time a spot is on the edge of the solar disk. To correct this phenomena, SLDTk[3] tool was used. The procedure starts by detecting the solar disk, then the pixel intensity profile of the disk is inferred by creating circles of increasing radius expanding until the borders storing the mean values of each circle perimeter. Next, a second-degree polynomial function is fitted to the intensity profile, which is by last straightened, correcting each pixel intensity using a linear fitting model, as shown in Figure 3.3. Using the 45s series, it was visible that most of the occurrences illustrated in Figure 3.2 were gone, although in tiny spots it could still be detected, even within intervals of 10 seconds. As a matter of fact, this is not a concern in the small to medium sunspots (neither in the larger ones naturally) since the seed even if it is not placed in the exact supposed centre, the expansion will start inside the spot, and the result will be the same; however, in the minimal sunspots it can be catastrophic. These anomalies, although small they seem, can be quite harmful to the algorithm. Having a constant source of irregularities in the ground-truth data can prevent the model from converging to the global minima, and therefore this should be rectified.

To overcome this obstacle it was necessary to take a step back and rely only on the HMIDD catalogues to build the dataset, with the drawback of having only annotations from mid-2010 to 2014. As they are made specifically in observations from the SDO, it simplifies the pre-processing steps necessary for the masks generation since the coordinate system does not need to be rotated, and the series with limb

[2] The first tests were conducted using the 720s series.

[3] `https://github.com/LandingEllipse/SLDTk`

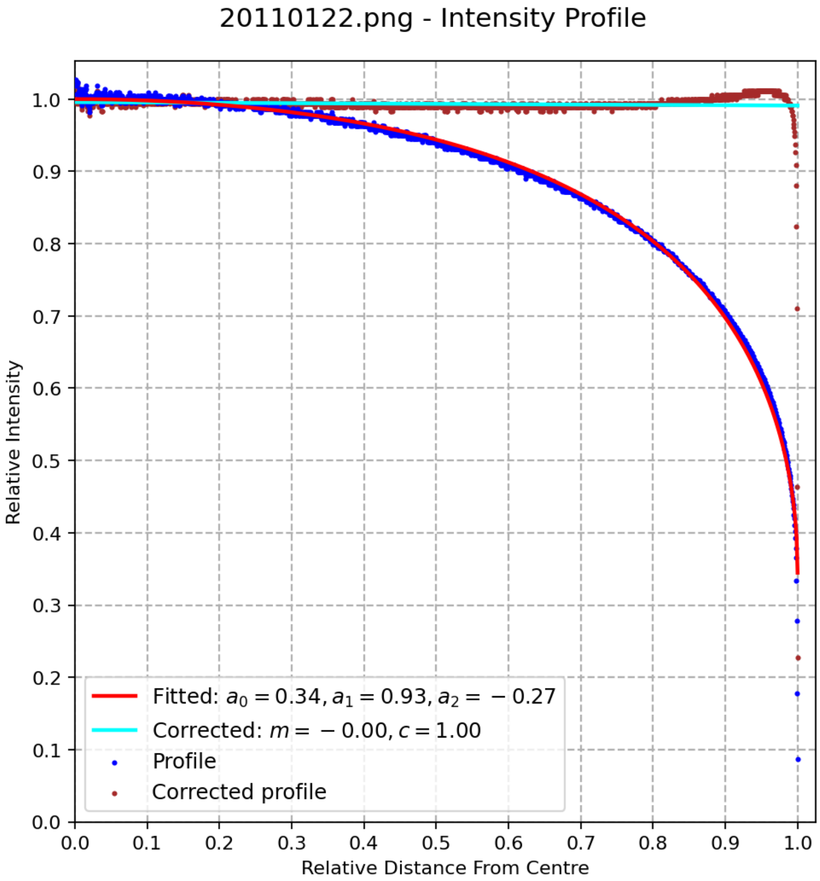


Figure 3.3: Limb darkening intensity profile detection and subsequent correction of a record from 22/1/2011 with SLDTk.

darkening correction could be selected sparing its processing as well. Although the use of HMIDD catalogues implies a shorter solar cycle sampling, it provides hourly data, and that benefits the dataset size. Thus, in order to make the most of the daily distribution, four records equally spaced in 24 hours period were downloaded from JSOC by the same procedure described previously. The final superimposed masks examples showed extremely accurate results not presenting critical anomalies visibly present, as far as seen from a non-expert solar observer, which indicates that the datasets could be finally safely generated. Quite small filaments in the borders of very few spots, which might indicate that the real spot area is slightly smaller than the annotation states, and extremely small misalignments in very few tiny spots can still be found, but naturally, these final masks are as good as the annotations from the algorithm used by DHO are, which limits the effectiveness of this procedure. An example sample of a mask generation procedure is shown in Figure 3.4, and a superimposed ground-truth semantic masks in Figure B.2.

Considering the two approaches developed in this work, semantic and instance segmentation, two separated mask generation processes should be conducted. The semantic masks are group agnostic; the spots individual IDs are not necessary for the model, therefore for each solar image, we have a single binary segmentation mask where the same pixel value is assigned to different groups of spots. On the other hand, for instance segmentation, each group will be treated as an instance (object), therefore for

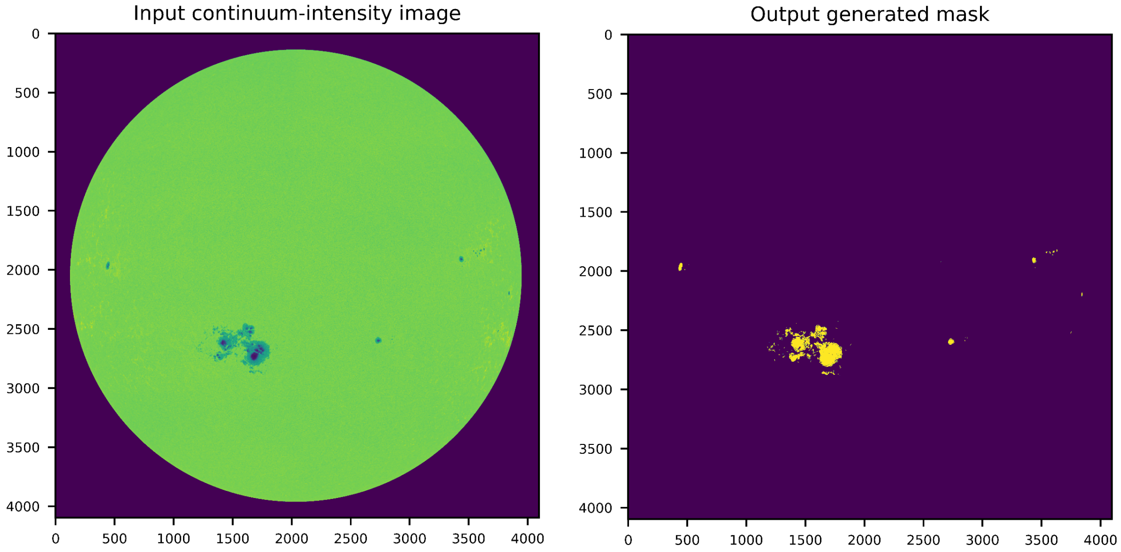
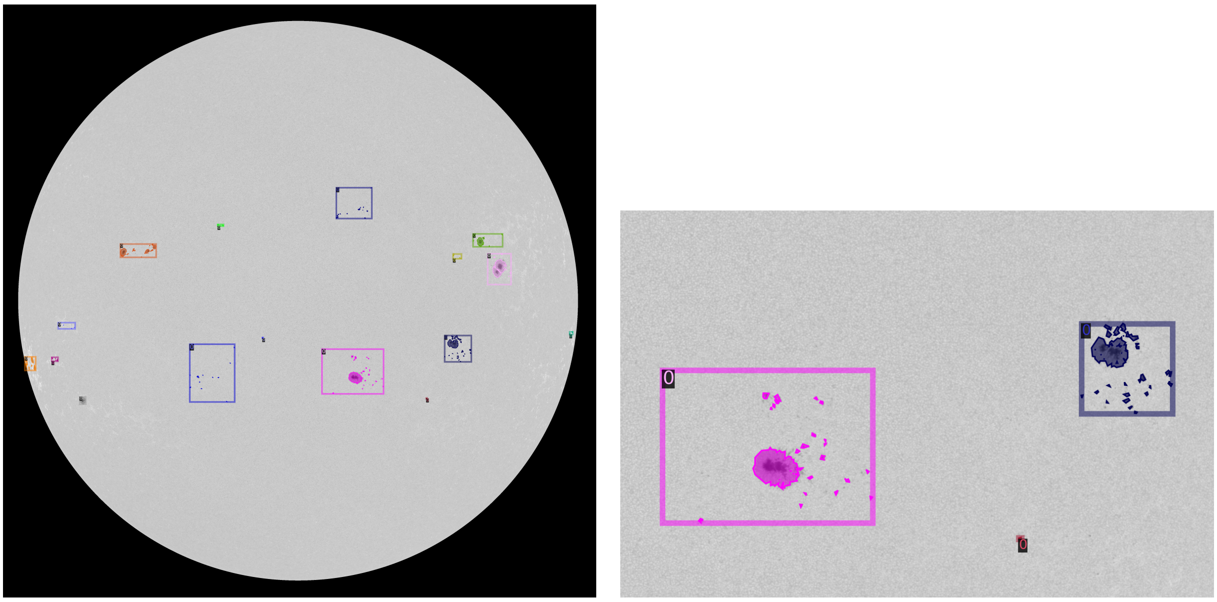


Figure 3.4: Example of a semantic segmentation mask generation.

each solar image it was created several segmentation masks equal to the number of groups present in the image for further processing[4]. Additionally, bounding boxes for each group are also needed, the most external spot coordinates define them for the different boundaries in each group. Furthermore, in order to load the dataset into the framework used, it is needed to convert it to COCO format - a specific JSON structure that dictates how labels and metadata are saved for an image. This format stores all the pixels for each instance segmented, coordinates to every bounding box and the correspondence from each solar image to its respective information block for every single image in the specific dataset (e.g.: training, validation and test sets). Thus, for each dataset, we will have a JSON file containing all the annotations necessary for every image, making the whole process of the dataloaders more efficient. A script was therefore written to convert the instance masks to COCO format. An example of a final instance mask loaded in Detectron2 is shown in Figure 3.5.

[4] Note that is one of the possible ways, for example, having a single mask with scaled pixels values according to the number of groups, (i.e., 0: background, 1: group X, 2: group Y, ...) can also do the trick. In this case, it was used several black and white masks not normalized (0: background, 255: group) for a single record just for the ease of visualization.



(a) Full-disk instance segmentation mask containing sixteen groups

(b) Zoomed area of the full-disk containing three groups

Figure 3.5: Example of an instance segmentation mask loaded in Detectron2.

In regard to the final dataset split into training, validation and test set, some aspects should be noted. Randomly assigning samples accordingly to some distribution into the different sets in the context of solar imaging surveillance is not the best practice due to temporal dependencies of the global dataset. For example, in one day we have a total of 4 samples (ideally), the differences in the intrinsic information between those samples are small; thus, if three of them are assigned to the three different sets, the model will see the nearly same data in all sets, which is not desirable for generalisation purposes. To accommodate this matter, the division of the sets was made with the following distribution:

- **training set**: every sample from January to September of each year.
- **validation set**: every sample of October of each year.
- **test set**: every sample from November to December of each year.

In conclusion, the final dataset is composed of:

- **6468** 4K[5] images captured by SDO[6].
- **59679** group instances.
- approximately **884252** individual spots[7].

[5] $4096^2$ pixels.

[6] Due to availability issues, around 350 images were not retrieved from JSOC, for that reason this amount is smaller than supposed.

[7] For the same reason mentioned previously, this value was estimated from the declared 929641 individual spots in HMIDD.

## 3.2 Evaluation Metrics

In the context of each approach, more adequate metrics exist depending on the application and its significance w.r.t to the data itself. Considering that ideally, the goal is to establish a comparison with automatic sunspot detection algorithms, the most representative metrics were used.

The instance segmentation approach, unfortunately, cannot establish a direct comparison with the common metrics employed by the current algorithms for sunspot detection indicated in section 2.2. Since it encapsulates two problems, the sunspot group (object) detection and its segmentation, the standard evaluation metrics for this type of algorithms are distinct from those mentioned. Thus, only the mask quality can be visually compared, however with the drawback of being performed in a multi-task regime, that is, being computed and optimised paired with RPN's two tasks and the Box's Head two tasks. For that reason, the semantic approach is the one that in fact can establish this comparison on a fairer ground. Nevertheless, the bounding box evaluation w.r.t the HMIDD dataset following the COCO metrics is computed, as well as the precision of the number of groups predictions along the analysed years.

To evaluate the semantic segmentation models, **Intersection-over-Union (IoU)** was computed, which is also calculated in [31](2016) where three methods for automatic sunspot detection were compared, one of them being the ASAP presented in section 2.2, and the remaining methods were originally developed to the study. However, due to unknown reasons, in [31] the IoU metric was given the name "**Quality Index (Q)**", presenting the same formulation as the former. Considering a ground-truth $(Y)$ set and a predicted set $(\hat{Y})$:

$$IoU(Y, \hat{Y}) = \frac{|Y \cap \hat{Y}|}{|Y \cup \hat{Y}|} \tag{3.1}$$

in other words, IoU is defined as the area of intersection of the sets over the area of the union, which can also be written as:

$$IoU = \frac{TP}{TP + FN + FP} = Q, \tag{3.2}$$

Note that other metrics could the used in this matter which are also commonly used, for instance, accuracy and dice coefficient, although the former should not be employed in the context of highly imbalanced class data since is not representative, which is the case (sunspots may represent less than 1% of the pixels in a sample), and the latter has not been used in the context of sunspot detection algorithms, therefore not considered here.

In regard to instance segmentation, COCO metrics were employed since they are widely used, computing as the primary metric the **mean average precision (mAP)** along with different values of IoU and instance scales, which follow a particular procedure. In object detection, a prediction is considered to be a **true positive (TP)** if satisfies three conditions: confidence score greater than a given threshold; the predicted class matches the class of a ground-truth; and the predicted box has an IoU greater than a given threshold with the ground-truth. If either of the latter two conditions is violated, a prediction is considered a **false positive (FP)**[8]. When a confidence score of a detection that is supposed to be positive is lower than the threshold, the detection counts as **false negative (FN)**. On the other hand, when

[8] It is worth mentioning that depending on a particular application/challenge other conditions may need to be respected, for instance the PASCAL VOC Challenge includes some additional rules to define true/false positives.

the confidence score of a detection that should be a negative is lower than the defined threshold, the detection counts as a **true negative (TN)**.

**Precision** is defined as the percentage of items (e.g., pixels) classified as positive which are actually positive. **Recall** is the true positive rate, defined as the percentage of positives that are classified as positive.

$$precision = \frac{TP}{TP + FP} \tag{3.3}$$

$$recall = \frac{TP}{TP + FN} \tag{3.4}$$

To have a more suitable numerical metric for comparison between different detectors, since the precision-recall curve is not ideal, **average precision (AP)** comes into play. Essentially, AP is the precision averaged across all unique recalls levels and can be defined as the area under the interpolated precision-recall curve, which can be calculated as:

$$AP = \sum_{i=1}^{n-1} (r_{i+1} - r_i) p_{interp}(r_{i+1}) \tag{3.5}$$

where $p_{interp}$ is the interpolated precision at a certain recall level $r$, traditionally equally space (e.g: $r$ = 0.1, 0.2, ...), in ascending order at which the precision was first interpolated. When the application involves more than one class (which is not the case in this application), the mAP is calculated by averaging AP across the all classes. Thus here, AP and mAP are equivalent.

### 3.2.1 COCO Detection Evaluation Metrics

COCO challenge defines several AP[9] metrics using different thresholds, including:

- $AP^{IoU=.5:.05:.95}$, AP averaged over 10 IoU thresholds (primary challenge metric).
- $AP^{IoU=.50}$, AP at IoU=0.5.
- $AP^{IoU=.75}$, AP at IoU=0.75.

Additionally, AP is calculated across different object scales, averaged over 10 thresholds:

- $AP^{small}$, AP for small objects that have less than 32$^2$ pixels in their segmentation mask.
- $AP^{medium}$, AP for medium objects that have greater than 32$^2$ pixels in their segmentation mask but less then 96$^2$ pixels.
- $AP^{large}$, AP for large objects that have more than 96$^2$ pixels in their segmentation mask.

[9] mAP will be denoted as AP henceforward.

## 3.3 U-Net and U-Net 3+

The semantic segmentation approach to sunspot detection establishes a direct comparison to the state-of-the-art detection algorithms described in section 2.2. While instance segmentation is a novel approach to this problem that merges the sunspot detection with its segmentation, semantic segmentation only tackles the pixel-by-pixel classification, which is essentially similar to what the algorithms based on mathematical morphology and pixel intensity do in distinct ways.

The implementation of U-Net and U-Net 3+ was based on PaddleSeg [64] (2021), a high-efficient development toolkit for image segmentation of PaddlePaddle [65], an open-source deep learning framework developed by Baidu. This toolkit provides numerous high-quality segmentation models following the official implementation releases, and it aims for ease and speed of research and development, the reasons why it was chosen.

The application of both networks follows the standard released architecture, and the same pre-processing pipeline is applied to both. Here, binary segmentation masks in image format can be directly loaded as annotations.

Both networks were submitted to different training setups, varying input sizes, augmentation strategies and optimisers. Due to the considerably small size of numerous sunspots, the loss of information originated by a high resize factor hugely harm the performance of the models. This event was in fact detected when establishing a direct comparison between U-Net to U-Net 3+. The capacity restrictions of the GPUs used during this implementation only allowed U-Net 3+ to have a network input of $512^2$ pixels, against the $1024^2$ pixels size allowed for U-Net, thus for their fair comparison, U-Net was also trained and tested with an input size of $512^2$. The effects of this re-scaling are shown in the next chapter.

Concerning data augmentation, beyond the generalization purposes, with the intention to make the model robust to different source images that might present slightly different properties, some noise was introduced. This aspect is more common in ground-based observations; as a matter of fact, in SDO images this is not a serious problem since it usually[10] present the same image properties, but these appearance changes can be problematic to several algorithms that rely on pixel intensity for features detection, like the ones described in section 2.2. Thus, in addition to the common horizontal/vertical flip and normalization, the following transforms were added:

- Random Distortion: distort an image following the specified configuration with a probability of 0.5 each.
  - Brightness Range: 0.2
  - Contrast Range: 0.2
  - Saturation Range: 0.1

SGD and Adam were the optimizers used, Adam provided a faster convergence but SGD presented as expected better results (around two IoU points). Polynomial decay was chosen for learning rate

[10] There is an observable appearance change in the converted grey-scale images recorded during 2011 for example. Before that period, the images appeared darker than the current ones stored. Sporadic occurrences can also be detected after that year.

schedule as its use was set in [66, 67] for semantic segmentation tasks further yielding higher final accuracy, and is given by:

$$lr = lr_0 \cdot (1 - \frac{i}{T_i})^{power} \tag{3.6}$$

where $lr_0$ is the initial learning rate, $i$ denotes the iteration number and $T_i$ is the total number of iterations. The $power$ term controls the shape of learning rate decay as shown in Figure 3.6, presenting better results when set to 0.9.

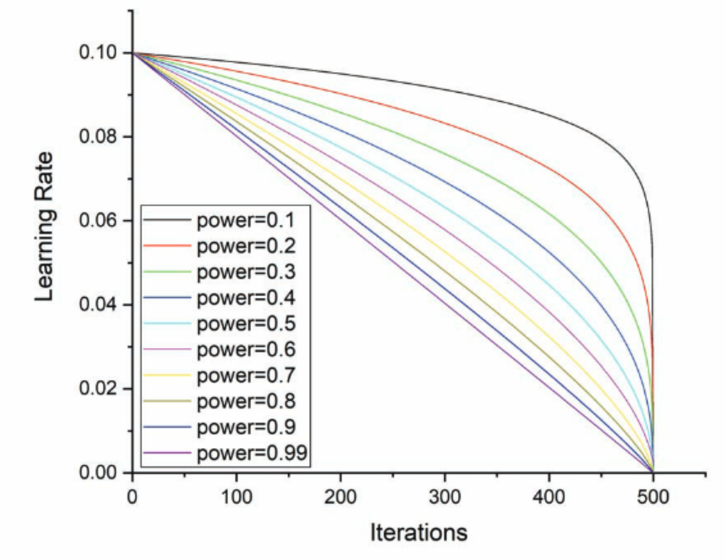


Figure 3.6: Poly

The learning rate was submitted to several rounds of testing in the set {0.1; 0.01; 0.001; 0.0001}, presenting better model training behaviour for 0.01 for both models with a batch size of 1.

Regarding the loss function, three of the most widely used losses to segmentation, namely, binary cross-entropy, focal loss and dice loss, were applied to inspect which one yielded better evaluation performance.

Binary cross-entropy (BCE), which is defined as a measure of the difference between two probability distributions for a given random variable or set of events, and it is given by:

$$L_{BCE}(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \tag{3.7}$$

where $y$ is the numerical ground-truth label and $\hat{y}$ is the model's predicted probability for the event.

Focal loss (FL) can be seen as a variation of BCE and is very used in highly imbalanced class scenarios (which is the case). It down-weights the contribution of easy cases and enables the model to focus more on learning the hard ones. It is defined as:

$$FL(p_t) = -\alpha_t (1 - p_t)^{\gamma} \log(p_t)\,, \quad p_t = \begin{cases} p\,, & \text{if } y = 1 \\ 1 - p\,, & \text{otherwise} \end{cases} \tag{3.8}$$

where $p_t$ is the probability of a class, $\gamma > 0$, and when $\gamma = 1$ FL works like cross-entropy loss function. Similarly, $\alpha$ generally range from [0,1], it can be set by inverting class frequency (the case used) or treated as a hyperparameter.

Dice coefficient is a widely used metric in image segmentation that calculates the similarity between two images, later it was also adapted as a loss function known as Dice loss (DL) [68], and it's defined as:

$$DL(y, \hat{p}) = 1 - \frac{2y\hat{p} + 1}{y + \hat{p} + 1} \tag{3.9}$$

The results of each experiment are described in the next chapter.

## 3.4 Mask R-CNN

The deep learning model introduced to tackle sunspot group detection is based on the Mask R-CNN architecture. The official Mask R-CNN implementation developed by Facebook AI Research (FAIR) presented in [48] was released in Detectron (2018), a FAIR's open-source software platform that implements state-of-the-art object detection algorithms. A year later Detectron2 (2019) came as substitute, rewriting Detectron which is implemented in Caffe2 (deep learning framework) now in Pytorch [71]. Detectron2 presents several improvements in regard to the previous version, it has more modular and extensible design, by moving the entire training pipeline to GPU became faster and more scalable, and it presents a variety of improvements in several sub-optimal low-level design choices making the platform more efficient. All these considerations were taken into account, making Detectron2 the natural choice for this thesis Mask R-CNN implementation, which not only provides a more efficient environment for training and testing purposes but also guarantees a straight and secure comparison between the official model release and the modified model for our task in hands.

As described in section 2.5.1, Mask R-CNN presents several details and design decision in the different three components that resulted from previous state-of-the-art detectors, baseline theories and empirical studies. The released model was designed to tackle the standard natural image competitions, trained in datasets such as the Common Object in Context (COCO)[72], which present challenges far different from space-borne telescopic observations of the Sun.

Several adaptations of Mask R-CNN to broadly distinct problems have been conducted recently, for example, nuclei segmentation (2019), multi-organ segmentation (2020), optical nerve and disk detection (2020), remote sensing (2021), aircraft detection (2021), titanium dioxide particles detection (2021), and bacterial colony counting (2021). The adjustment strategies of these approaches were analysed, and several insights were taken for this application to sunspot group detection.

### 3.4.1 Mask R-CNN fine-tune

The direct use of Mask R-CNN original implementation model (i.e: the one trained on a vast natural scene images with coarsely-annotated objects) is naturally not sufficient. As a matter of fact, if applied for inference or even trained without any parameter adjustment to our application on solar images, the results are essentially poor. The main reason behind this failure is associated with the completely distinct nature of the tasks, leading further to sub-optimal network loose hyperparameters. In order to properly adjust the network to our purpose, first the architecture needs to meticulously studied and understood. The section 2.5.1 reviews thoroughly each component and process of Mask R-CNN following the official implementation by Facebook AI Research (FAIR), here it will be exposed the adjustments made and the aspects related to their implementation.

A sunspot group can have several spots islands, making therefore this problem fundamentally different as an instance is composed by a non-uniform number of objects, in contrast to an instance being a single object. This brings great challenges to bounding box regression particularly in periods of increased solar activity. Moreover, the considerable number of very small sunspots (being a significant portion of them almost untraceable to the human eye) is detected with difficulty by the backbone which complicates not only the masks prediction but also the bounding box delimitation since the boundaries of sunspots groups are generally constituted by small sunspots. Additionally, the size of sunspots groups differ greatly from natural objects, and the anchor box generation must be in accordance with the network input dimensions since the groups sizes naturally depend on it.

Mask R-CNN is comprised of dozens of hyperparameters from their three main components. Some of these hyperparameters affect the quality of the model and the ability to infer correct results, others more associated with time and memory cost of running the algorithm, which also affects the training behaviour. There are two approaches to adjusting hyperparameters: **manual hyperparameter tuning** and **automatic hyperparameter tuning**. Choosing the hyperparameters manually requires full understanding of what each hyperparameter do, how their different range of values affect the network, and how deep learning models achieve good generalization. Automatic hyperparameter selection algorithms significantly reduce the need of understanding these concepts and how the network fully operate, but they can be extremely computationally expensive, Goodfellow et al. [80].

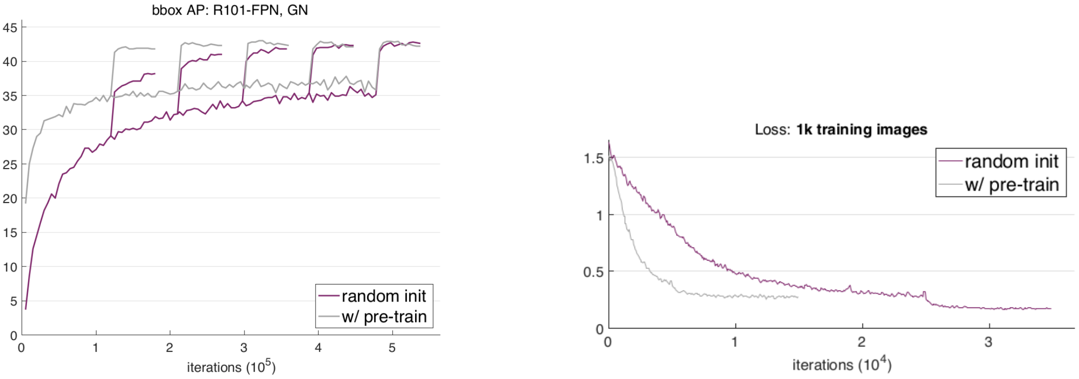
### Rethinking Transfer Learning

Transfer learning is a process of domain adaption widely use in deep learning, which consists on transferring the domain knowledge of a specific model trained for a specific tasks and reuse it in another related task. The benefits of this technique are faster training convergence, and in some cases, a performance increase.

Deep supervised learning models require a large amount of annotated data to properly train these networks aiming the best possible results. This aspect is even more pivotal in large architectures with millions of ”learnable” parameters such as Mask R-CNN. Training these networks in small datasets from scratch tend more easily to overfit, since its huge model capacity contribute strongly to approximate the objective function in a way that is hardly generalized in unseen data. Naturally, the attribution of “small” or “large” denominations are centrally relative to some reference point, although in the field of machine-learning this boundaries are not strictly defined and can vary greatly depending on the model, task and domain. For object detection for instance, state-of-the-art models are trained nowadays on over 300k annotated images, a usual large dataset size. The dataset for this implementation contains 6k images, which can be understood as small. This detail is especially important when applying transfer learning since there are different strategies depending on the dataset size and similarity of the pre-trained model and target model datasets.

In [81](2018) by FAIR, an extensive analysis is done to investigate the effects of the use of transfer learning of pre-trained models on ImageNet in comparison with training from scratch, specifically on Mask R-CNN. The comparison was made for different conditions, such as different datasets sizes, different augmentation setup (very important since it can help considerably the scenarios of small dataset sizes and its strong use highly affects the learning behaviour of both approaches), different normalizations (BN,SyncBN, GN) and training strategies.

The setup behaviour more related, and therefore most relevant, to the implementation of this work is associated when training with 1k images in analysis. The random initialization was able to catch up with the pre-trained setup training loss as depicted in Figure 3.7 b), but when it did, it presented a significantly lower validation accuracy (3.4 AP $vs.$ 9.9 AP), which shows a considerable lack of generalization capability.



(a) Learning curves of bounding box AP on COCO val2017 set using Mask R-CNN with R101-FPN and group normalization (GN). [81]

(b) Training loss when training with 1k COCO images on a Mask R-CNN with R50-FPN and GN. Using the same hyperparameters, the randomly initialized model can catch up for training loss, but when it does, presents a lower validation accuracy of 6.5 AP difference. [81]

Figure 3.7: Training behaviour comparison between of random initialized model and with ImageNet pre-training. [81]

The main observations to add from this study is that training from scratch requires longer learning schedules to catch up the ImageNet pre-train, however both approaches tend to converge to similar results when the dataset is large enough[11], with random initialization presenting a slightly better improvement. Also, ImageNet pre-training does not necessarily help reduce overfitting unless we enter in a very small data regime, which is the case here.

Training with high-resolution images raises the problem of long training schedules. As a matter of fact, resizing SDO images to 1K[12] and using ResNet-101 backbone with the unmodified Mask R-CNN model, took around 2 days in a NVIDIA GeForce GTX 1070 for just 100k iterations, which for hyperparameter tuning is far from ideal. For this reason, and taking into account the important generalization aspect mentioned above, the use of a pre-trained model seemed a reasonable choice, as for the model fine tuning as well as for comparison with the randomly initialized setup, to actually infer the ImageNet pre-training effect on sunspot detection.

As mentioned, there are different strategies when approaching transfer learning, essentially falling into four main cases depending on the dataset: large dataset with data similar to the original dataset; small dataset with data similar to the original dataset; large dataset with different data from the original dataset; and small dataset with different data from the original dataset. The dataset for this implementation belongs to the latter case, therefore its best practices should be pursued.

One can look into CNN layers as knowledge hierarchy levels, deeper they are the more complex features of the input they are able detect, being the first layers the ones that have more low-level feature

[11] With 10k images the model with pre-training reached 26.0 AP with 60k iterations and the counterpart model trained from scratch had 25.9 AP at 220k iterations, which is comparably accurate.

[12] 1024×1024.

knowledge, such as shapes and edges. Although an ImageNet pre-trained backbone contains a lot of gained information that is useless to our application, it possess already the sufficient knowledge in early stages that are relevant to sunspot detection. In this cases the common practice is to freeze the first layers (first here is not strictly defined), randomly initialize the weights of the rest of the network, replace the existing heads last layers matching the number of classes for the specific tasks and finally train the network only updating the unfreezed layers. In this implementation, when applied transfer learning, it was frozen down the first two ResNet stages (stem, res2), and the rest of the backbone, the RPN and RoI Heads were randomly initialized for training. This decision is supported in [82], which demonstrates that the first convolutional block is generic and task independent, and in [58] shows that allowing it, or not, to train (conv_1 can be understood as the equivalent of res2 here) has no meaningful effect in the final average precision.

### Observations from the standard baseline

Before any adjustment to the network, its behaviour with the standard settings was analysed. As described in section 2.5.1, Mask R-CNN is composed of five individual losses, and it is based on each of those that during training can be spotted the components with worst performances.

Thus, a ResNet-50 was selected for analysis with random initialization, training with the learning schedule set to 100k iterations (enough for analysis), learning rate of 0.01, batch size of 1 and network input of $1024^2$ pixels. Since early training, the RPN was the component that in proportion most contribute to the general loss, mainly its $L_{loc}$ term, fluctuating around 50% of the total loss ($\approx$40% $L_{loc}$, 10% $L_{cls}$). Regarding the RoI Heads, classification had the least contribution (around 10%) which is expected, being the box regression the most unstable task, fluctuating between 17% and 27% of the total loss during the last 40k iterations, with the mask loss steadily converging.

Concerning the COCO detection metrics, the results of this direct application are poor as shown in Table 3.2, but from these dimensional separation performances reported by the evaluator important insights could be taken. Clearly, the biggest challenge of the model is to detect accurately the small sunspot groups, presenting metrics for bounding box (which was expected from the training losses) and segmentation for smaller instances very low.

Considering the network data flow, RPN should be first analysed prior to the RoI Heads. The backbone is not the main concern, at least for an initial approach. If the proposals from RPN are unsatisfactory, the post-processing in RoIAlign and further task branches will perform poorly as well, therefore its tune must take place first. The results comparison between different backbones with random and pre-trained initialization with the standard baseline are described in the section 4.2.

Table 3.2: Mask R-CNN bounding box and instance segmentation mask results on test set for randomly initialized standard baseline.

| | BBox | | | | | | Mask | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Mask R-CNN w/** | AP | AP50 | AP75 | APs | APm | APl | AP | AP50 | AP75 | APs | APm | APl |
| ResNet-50-FPN | 10.4 | 20.5 | 16.7 | 7.2 | 36.2 | 12.5 | 5.9 | 17.0 | 3.9 | 0.6 | 31.0 | 27.6 |

### RPN Experiments

In order to understand the response of RPN sub-components to the information flow, its engine is submitted to test. The main hyperparameters of this network are shown in Table 3.3, with: $RPN_{IN-F}$ being the input features maps from FPN; $RPN_{BT}$ the pixel boundary threshold for boxes that cross the image (-1 stands for them complete removal); $RPN_{IoU-T}$ the lower and upper intersection-over-union threshold; $RPN_{BS}$ the regions/anchors (batch size) per image used to train RPN; $RPN_{PF}$ the target positive fraction; $RPN_{SL1-\beta}$ the transition point from L1 regression loss to L2 loss (set to 0.0 to make simply L1); $RPN_{PR-NMS-K-TR}$ and $RPN_{PR-NMS-K-TS}$ the number of top scoring RPN proposals to keep before applying NMS (this value is per FPN level) for training and testing respectively; $RPN_{PS-NMS-K-TR}$ and $RPN_{PS-NMS-K-TS}$ the number of final top scoring RPN proposals to keep after applying NMS (this value is the union of the proposals of all levels) for training and testing respectively; $RPN_{NMS-T}$ the NMS threshold; $RPN_{CONV-D}$ the channel dimensions of RPN (set to -1 to use the same number of output channels as input channels); $RPN_{AG-S}$ the anchor sizes (i.e: root of area) in absolute pixels w.r.t to the network input for each FPN feature map level respectively and $RPN_{AG-AP}$ the aspect ratios for each generated anchor box. Note that more hyperparameters are related to this component, although not particularly relevant to fine-tune analysis.

Table 3.3: Region Network Proposal main hyperparameters.

| Hyperparameters | |
|---|---|
| $RPN_{IN-F}$ | $[P2,P3,P4,P5,P6]$ |
| $RPN_{BT}$ | -1 |
| $RPN_{IoU-T}$ | [0.3,0.7] |
| $RPN_{BS}$ | 256 |
| $RPN_{PF}$ | 0.5 |
| $RPN_{SL1-\beta}$ | 0.0 |
| $RPN_{PR-NMS-K-TR}$ | 2000 |
| $RPN_{PR-NMS-K-TS}$ | 1000 |
| $RPN_{PS-NMS-K-TR}$ | 1000 |
| $RPN_{PS-NMS-K-TS}$ | 1000 |
| $RPN_{NMS-T}$ | 0.7 |
| $RPN_{CONV-D}$ | -1 |
| $RPN_{AG-S}$ | [32, 64, 128, 256, 512] |
| $RPN_{AG-AP}$ | [1:2, 1:1, 2:1] |

To better understand the influence of each parameter, the ablation experiments in [48, 53, 57, 58] were carefully analysed, where the same parameters have been already subjected to different ranges of values and configurations for training and testing purposes. $RPN_{BS}$, $RPN_{AG-S}$, $RPN_{PS-NMS-K-TR}$ and $RPN_{PS-NMS-K-TS}$ were the parameters most evaluated in those experiments in regard to RPN.

Here, $RPN_{BS}$ and $RPN_{PS-NMS-K-TR}$ were submitted to a different range of values in accordance with the analysed approaches. As expected, both reflect significant effects on the convergence of the total loss, making it significantly faster with an increase of both, however making each iteration more expensive. It is found that an increase of $RPN_{BS}$ leads to a significantly faster convergence, presenting a greater influence on both regression losses than on both classification losses, as tested with the following values: $\{256(\text{default}), 512, 1024, 2048, 3000, 4096\}$.

These tests were conducted for 100k iterations with pre-trained weights provided by Detectron2 with a ResNet-50 backbone, as it is the least computational expensive (therefore faster), and the modification results are valid to the other ResNet backbones naturally, with a network batch size of 1. Mask R-CNN presented lower values for the total loss with an increase up to 4096 proposals, although the difference from 3000 to 4096 proposals was residual, and since it increases the computation time by some hours (as it sees sixteen times more proposals than the default value), was not considered further. The default value caught the training losses of the 3000 setup only after 30k iterations. To evaluate the effective detection results, the weights of both models that were being stored every 5k iterations (checkpoints) which presented similar training losses were sent to inference in the validation set, and it was verified an increase around 0.5 AP in bounding box for a $RPN_{BS}$ of 3000, which together with the significant convergence factor validate the change.

Regarding $RPN_{PS-NMS-K-TR}$, its study was conjugated with different $RPN_{BS}$ values (256,512). The double-up value of the default $RPN_{PS-NMS-K-TR}$ reflected poorly in the Box Head loss for both batch sizes, which indicates that the network does not actually leverage in seeing more than 1000 proposals in the RoI Heads, but the reverse. This event validates what is shown in [58][13], which illustrates the mAP and AR evolution for a different number of proposals, while AR continuously increases due to more proposals mAP starts to decrease at a certain point, not presenting a correlated behaviour with AR; therefore this parameter must be set carefully as more proposals are not always better. To understand the reason for this event in this application, the number of positive boxes (foreground) that reach the RoI Heads was extracted and averaged, presenting a mean value around 190 boxes, which indicates that increasing $RPN_{PS-NMS-K-TR}$ only leads to more negative proposals (and with lower scores) and that may cause the performance decrease. Thus, this test proceeded in the opposite direction, decreasing $RPN_{PS-NMS-K-TR}$ down to 500 proposals; however, it was verified a massive effect on the regression loss, taking much more time to converge, therefore the default value was maintained.

Concerning $RPN_{AG-S}$, its tune is pivotal in this model application to sunspot detection since sunspot groups dimensions can differ significantly from natural objects. From the results of the standard baseline, it was clear that the small instances category needed special attention, and for that, the dataset should be analysed. In Table 3.4 is shown the statistics of the training set w.r.t to the COCO evaluation dimensions. The original 4K images are rescaled for training to $1024^2$ pixels to fit on the GPU, therefore these presented values are accordingly to the resize. Count shows the number of instances for each dimension, followed by the minimum lateral size of a box, the maximum lateral size of a box, mean box lateral size and box standard deviation lateral size. It is clear that there is a high inconsistency between the sunspot pixels amount of a group (i.e., pixels in the mask) and its box size, which is naturally expected, however it makes the training extremely difficult because the Box Head might have to implicitly learn the configurations of different sunspot groups class in an unsupervised way. $RPN_{AG-S}$ was the

[13] The analysis made in [58] was performed for selective search (SS) rather than RPN (the more recent approach), although the effect on the loss evolution on the Box head can be assumed to be similar since we are evaluating the number of proposals rather than the quality of them.

parameter that most was subjected to testing as it should be in tune with the resize factor of the original input, thus not only the five box values for each level was tested but also different input sizes to seek the best match. Note that there is no theoretical approach that might approximate the size of the anchor boxes, only empirically it is possible to find the best configuration.

Table 3.4: Sunspot group training instances in COCO dimensions with SDO images rescale to 1024$^2$ pixels.

| Dimensions | Count | Min. Size | Max. Size | Mean | Std |
|---|---|---|---|---|---|
| Small | 31892 | 0.25 | 324.75 | 29.53 | 25.15 |
| Medium | 11160 | 4.75 | 534.5 | 80.12 | 27.22 |
| Large | 1133 | 29.25 | 534.75 | 133.27 | 11.49 |

Small instances dominate the distribution, and its box size variation as shown in Table 3.4 is a concern. The reader should take in mind that the anchor boxes are not static, as described in section 2.5.1, the bounding box prune computation will be responsible for learning how to deform each box with respect to the RoI, therefore here the goal is to find the best halfway configuration that allows this pruning most efficient. Setting the smaller and large anchor boxes to the minimum and maximum lateral size is sub-optimal since the mean boxes size concentration are far from those limits. Thus, several tests were conducted to infer the minimum and maximum anchor box size that could better approximate the size limits shown, particularly the lower bound since there is a significant concentration of very small boxes. Regarding the middle levels, the idea was to set the anchor boxes inside a mean-std dimensional interval, although several configurations might lead to similar results here. After several series of tests with a 50k iter regime, input size of $\{$800$^2$, 1024$^2$, 1280$^2$, 1568$^2\}$, and comparing the different setups only at the end of each training schedule, the best setup found was $\{$Input size = 1024$^2$ pixels; $RPN_{AG-S}$ = $\{$18, 46, 92, 160, 360$\}\}$.

The nature of sunspot group detection justifies a reconsideration of the IoU threshold. Since an instance is not composed of a unified body, like in natural images, but rather several separated bodies, this parameter must be tested to different configurations as well. In cases of great sunspot group density, the network continuously presents many false positives since the distance between groups can be very small, as shown in Figure B.1, and their delimitation is extremely difficult to predict. Following that thought, it seems logical to force the network to see more negative cases with higher overlap (IoU) and to see more accurate positive proposals. This can be accomplished by increasing both the lower threshold and positive threshold. The training behaviour showed to be very sensitive, particularly to larger modifications, taking more time to converge. The tests were conducted similarly as for $RPN_{BS}$, comparing model setups loaded with the weights from which the training loss was similar, between all the combinations of threshold limits in $\{$0.3; 0.35; 0.4; 0.45$\}$ (lower bound) and $\{$0.7; 0.75; 0.8$\}$ (upper bound), with a training schedule of 100k iterations since the convergence time is quite affected. The best setup was found with $RPN_{IoU}$: $\{$0.4, 0.75$\}$, presenting a surprising three points increase in bounding box AP comparatively with the initial setup, which validates successfully this conceptual strategy.

Lastly, $RPN_{NMS-T}$ was also submitted to tests with all the last mentioned modifications employed since it happens *a posteriori* in the network process. The increase was naturally tested to see if sending less but more accurate proposals was better, with intervals of 5% from the default value using the same comparison strategy from previous experiments. It was noted a slower convergence by each increase, which is expected since the RoI Heads see less positive samples, and a slightly better box AP (around one point) in the validation set with $RPN_{NMS-T}$ of 0.75, interestingly being in tune with $RPN_{IoU}$.

Regarding the bounding box loss function, smooth L1 loss with no transition to L2 ($\beta = 0$) is the most used in several detection models as it is known to be the most robust to outliers [58], therefore the default configuration was maintained. $RPN_{BT}$ was also maintained since there is no ground-truth cases of bounding boxes that might touch the boundaries of the image, therefore their removal is adequate.

The best configuration of hyperparameters is further described in section 4.2, together with the final results.

### RoI Heads Experiments

Following the best outcome configuration of RPN, the RoI Heads were analysed. The modifications made until this point already presented great improvements in the detection metrics, and by now $L_{reg}$ and $L_{mask}$ represented together 65% of the total loss.

Maintaining the original architecture of both heads, the capacity of both was submitted to testing. During the tuning of RPN, it was not detected significant effects on the loss metrics of $L_{cls}$ and $L_{mask}$, only in $L_{reg}$, which mean that the tasks less affected by the quality of the regions proposals were the classification and mask prediction. The main hyperparameters of RoI Heads are presented in Table 3.5:

Table 3.5: RoI Heads main hyperparameters.

| Hyperparameters | |
|---|---|
| $RoI_{BS}$ | 512 |
| $RoI_{PF}$ | 0.25 |
| $RoI_{BB-SL1-\beta}$ | 0.0 |
| $RoI_{BB-PoolerR}$ | 7 |
| $RoI_{BB-PoolerT}$ | ROIAlign |
| $RoI_{BB-N-FC}$ | 2 |
| $RoI_{BB-FCDim}$ | 1024 |
| $RoI_{M-PoolerR}$ | 14 |
| $RoI_{M-PoolerT}$ | ROIAlign |
| $RoI_{M-N-CONV}$ | 4 |
| $RoI_{M-CONV-Dim}$ | 256 |

where $RoI_{BS}$ is the number of regions (batch size) used to train the RoI Heads, $RoI_{PF}$ is the RoI positive fraction, $RoI_{BB-SL1-\beta}$ is the transition point from L1 to L2 (beta), $RoI_{BB-PoolerR}$ is the output resolution of the Box Head RoIAlign operation, $RoI_{BB-PoolerT}$ is the pooler type of Box Head, $RoI_{BB-N-FC}$ is the number of fully connected layers in Box Head, $RoI_{BB-FCDim}$ is the dimension of the fully connected layers in the Box Head, $RoI_{M-PoolerR}$ is the output resolution of the Mask Head RoIAlign operation, $RoI_{M-PoolerT}$ is the pooler type of Mask Head, $RoI_{M-N-CONV}$ is the number of convolutional layers in the FCN, $RoI_{M-CONV-Dim}$ is the channels dimension in each layer of the FCN.

In regard to the Box Head regression, smooth L1 loss with no transition to L2 was maintained for the same reason as before. $RoI_{BS}$ followed the same tuning strategy of $RPN_{BS}$, however with no gains of performance when increasing the default value. The capacity increase was explored by increasing the dimensions of the fully connected layer and the number of layers. Similarly to the RPN testing, these experiments were conducted in training regimes from 50k to 100k iterations range (depending on the modification) for temporal reasons[14]. It was found a significant improvement (around three AP points) with an increase of $RoI_{BB-FC-Dim}$ to 2048, maintaining the number of layers. The capacity was stretched out further by increasing the number of layers and increasing $RoI_{BB-FC-Dim}$ to 4096; although the training loss decreased, the validation results were similar to 2048, showing signs thus of overfitting, for that reason not considered further[15].

Detectron2 provide an improvement of RoIAlign, described in section 3.4, namely RoIAlignV2. To simply explain the modification from the original version presented in [48], consider a continuous coordinate $c$. Its two neighboring pixels indices are computed by $floor(c - 0.5)$ and $ceil(c - 0.5)$. If $c = 1.3$, the pixel neighbours with discrete indices are [0] and [1], which are sampled from the interpolation method at 0.5 and 1.5. However, the original RoIAlign does not subtract the 0.5 when computing the neighbouring pixel indexes, and therefore it uses pixels with a slightly incorrect alignment when performing bilinear interpolation. RoIAlignV2 rectifies it by subtracting the half-pixel (0.5) from the RoI coordinates to compute it more accurately, and for that reason also applied here.

Concerning the Mask Head, its simple structure was tested by increasing the number of convolutional layers and feature maps. The idea was not only to increase the capacity but also to increase the receptive field of the FCN to perhaps be able to output more accurate segmentation of the smaller spots, although this proved to be quite inefficient, presenting overfitting just within a 50k regime.

The pooler resolution, on the other hand, proved to have a preponderant role in the quality of the masks, presenting improvements around three AP points with the double-up value of $RoI_{M-PoolerR}$. This increase largely raises the computational costs, and for that reason, it could not be more stretched out in the available hardware for memory grounds. These experiments results are described in the next chapter.

[14] For example, 50k iterations with a ResNet-50 backbone take around 20 hours with the available hardware (with validation loops every 6k iterations) when increasing the dimension of the FC layer to the double.

[15] Although this was the decision taken, there is a possibility that with a longer training schedule this further capacity increase might have led to better results later, but as mentioned, these experiments were strictly limited by the computational resources and time.

# Chapter 4

# Results

In this chapter, the results of both approaches are presented and discussed. As mentioned previously, the semantic segmentation approach was implemented in PaddlePaddle framework and the instance segmentation model in Pytorch, using Detectron2 library. Both were trained, validated and tested with the following hardware setup:

- NVIDIA GeForce GTX 1070 8GB
- Intel(R) Core(TM) i7-5820K CPU @ 3.30GHz

## 4.1 U-Net and U-Net 3+

Following the experiments' procedures described in section 3.3, Table 4.1 presents the U-Net IoU results of the best models for each loss function used, setting a resize factor of 2 (i.e., rescaling each training and validation/test image to $2048^2$ pixels) followed by a random crop of $1024^2$, full augmentation mentioned in section 3.3, batch size of 1 and learning rate of 0.01 for 80k iterations (around 1 day for U-Net, and 1 day and 8h for U-Net 3+) with random initialization. Each were submitted to three runs with the indicated setup, presented are the best performances from both. All the U-Net models presented better performance than the ASAP algorithm, and Method 2 of [31], both described in section 2.2; however, around 10.0 IoU points behind Method 3. The experiments of [31] used a total of 45 images of October and November of 2014 with 1200×1000 pixels, acquired in the Geophysical and Astronomical Observatory of the University of Coimbra (OGAUC), labelled by a solar observer expert to build the ground-truth dataset.

In Table 4.2 is shown the best results achieved for U-Net 3+ where due to computational limitations the network could only be trained with an effective input image of $512^2$ pixels, comparing to the corresponding U-Net with the same settings. The best results were achieved with a training pipeline that consisted of a first target resize to $1024^2$ followed by a random crop of $512^2$ pixels. Other configurations were tested to study the best setup, for example, a direct resize to $512^2$, and a resize to $2048^2$ followed by a random crop of $512^2$ pixels, however presenting worst performances. The best setup was trained three times for both models, presented are the best scores achieved.

Table 4.1: Comparison between the implemented U-Net model and three state-of-the-art sunspot detection algorithms.

| Model | IoU |
|---|---|
| ASAP [31] | 0.6930 |
| Method 2 [31] | 0.6435 |
| **Method 3** [31] | **0.8465** |
| U-Net (BCE loss) | 0.7166 |
| U-Net (FL loss) | 0.7238 |
| **U-Net (Dice loss)** | **0.7424** |

Table 4.2: Comparison between U-Net and U-Net 3+ results with an effective network input of $512^2$ pixels.

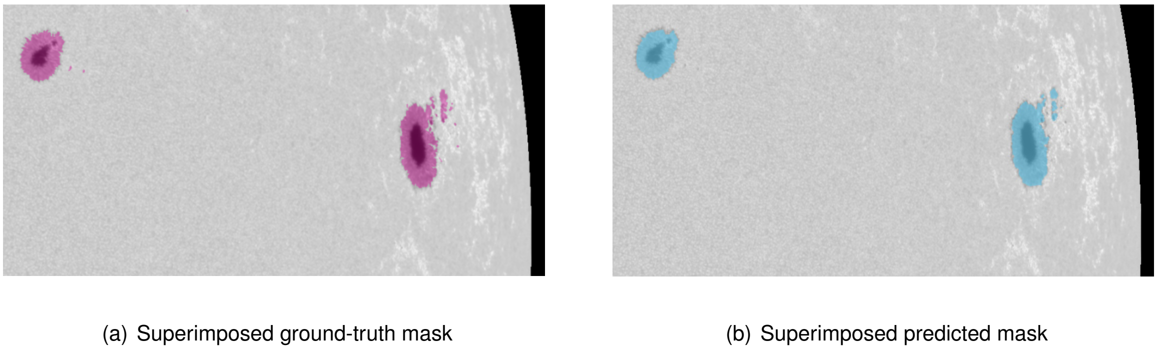
| Model | Params | IoU |
|---|---|---|
| U-Net (Dice loss) | 13.40M | 0.6206 |
| **U-Net3+ (Dice loss)** | 26.97M | **0.6496** |

The same U-Net model trained with both setups presented a decrease of around 12% with a resize factor of 4, which illustrate as expected the performance harm inducted by the huge information loss created by the rescaling operation. This aspect is critical in sunspot detection due to the massive number of extremelly small spots, making its detection with large rescales even harder.

Nevertheless, the results are auspicious, showing that with an increase of capacity of U-Net (or U-Net 3+), for instance, by employing a ResNet-101 backbone[1], this approach can at least approximate significantly the results of Method 3. The same applies to U-Net 3+. Furthermore, with a more robust backbone in both architectures and optimizing the training strategy by not applying any rescale factor but just feeding the network with fixed-size patches of rich information in each batch, that is, not sending the huge blocks of background in each image but only patches that contain spots, may in fact surpass the results presented by Method 3 while limiting the computational price.

Below is shown comparisons between the U-Net (Dice loss) prediction masks and the superimposed ground-truth masks. It is clear that the detection of larger spots is quite accurate. However, the model has difficulties in behaving on the boundaries of the spots and mainly, predicting the smaller sunspots accurately, as depicted in Figure 4.2.

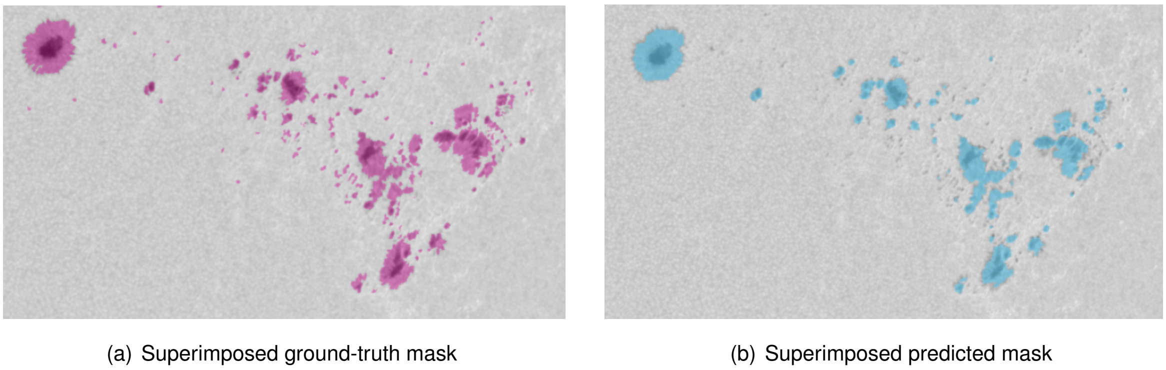
[1] U-Net with a ResNet-101 backbone presents around 55.90M parameters, four times the capacity of the standard U-Net used in this work.



(a) Superimposed ground-truth mask

(b) Superimposed predicted mask

Figure 4.1: Comparison between the ground-truth mask and the predicted mask by the best U-Net model in a disk zoomed area. Better visualization in Appendix C.



(a) Superimposed ground-truth mask

(b) Superimposed predicted mask

Figure 4.2: Comparison between the ground-truth mask and the predicted mask by the U-Net model in a disk zoomed area. Better visualization in Appendix C.

## 4.2 Mask R-CNN

Following the experiments description of section 3.4, the best configurations from the extensive analysis are presented on Table 4.3. Indicated are the parameters that ultimately were modified w.r.t the standard baseline as described previously, although numerous additional versions were subjected to experimentation.

The experiments related to the standard baseline with random initialisation in comparison to the v1 setup with pre-trained weights provided by Detectron2 are shown in Table 4.4. In these tests, all models were trained for 100k iterations, learning rate of 0.01 with linear warm-up and batch size of 1, with $1024^2$ input images with a simple random horizontal flip as augmentation. The effect of ImageNet pre-training (p-t) is clear, showing a similar increase of bounding box AP for ResNet-101 and ResNeXt-101-v1, around 6 points in both models. This aspect proves the evidences discussed in section 3.4, and to spare computational time and resources, all final models were initialised with pre-trained weights.

The improvements of the modified setup are clear, with the v1 setup presenting an increase around 15 points in bounding box AP and 4 points in segmentation mask from the standard baseline, showing thus the full effect of the fine-tune strategy conducted in section 3.4.

Table 4.3: Best experiments' modification setups with respect to the standard baseline.

| Hyperparameters | Standard baseline | v1 | v2 | v3 |
|---|---|---|---|---|
| $RPN_{IoU-T}$ | [0.3,0.7] | [0.4,0.75] | [0.4,0.75] | [0.4,0.75] |
| $RPN_{BS}$ | 256 | 3000 | 3000 | 3000 |
| $RPN_{NMS-T}$ | 0.7 | 0.75 | 0.75 | 0.75 |
| $RPN_{AG-S}$ | [32, 64, 128, 256, 512] | [18, 46, 92, 160, 360] | [18, 46, 92, 160, 360] | $==$ |
| $RoI_{BB-PoolerT}$ | ROIAlign | ROIAlignV2 | ROIAlignV2 | ROIAlignV2 |
| $RoI_{BB-FCDim}$ | 1024 | 1024 | 2048 | 2048 |
| $RoI_{M-PoolerR}$ | 14 | 14 | 14 | 28 |
| $RoI_{M-PoolerT}$ | ROIAlign | ROIAlignV2 | ROIAlignV2 | ROIAlignV2 |
| $RoI_{M-N-CONV}$ | 4 | 4 | 6 | 4 |

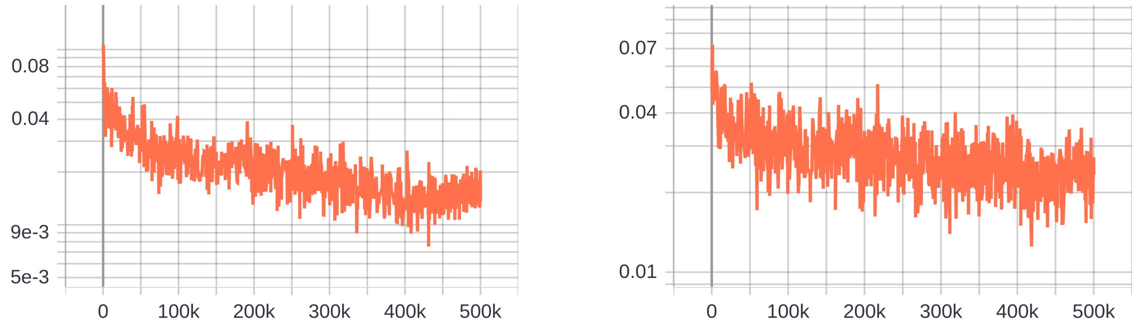
Table 4.4: Mask R-CNN (FPN) bounding box and instance segmentation mask results on test set, for both pre-trained and random initialization of standard baseline together with v1 setup comparison.

| | **BBox** | | | | | | **Mask** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Mask R-CNN w/** | AP | AP50 | AP75 | APs | APm | APl | AP | AP50 | AP75 | APs | APm | APl |
| ResNet-50 | 10.4 | 20.5 | 16.7 | 7.2 | 36.2 | 12.5 | 5.9 | 17.0 | 3.9 | 0.6 | 31.0 | 27.6 |
| ResNet-101 | 11.0 | 22.1 | 10.8 | 14.4 | 33.8 | 6.1 | 5.7 | 15.7 | 3.0 | 0.5 | 28.1 | 27.9 |
| ResNet-101(p-t) | 17.9 | 31.9 | 18.4 | 18.0 | 44.0 | 15.2 | 10.5 | 27.9 | 5.8 | 2.4 | 39.1 | 40.1 |
| ResNeXt-101 | 15.4 | 28.3 | 15.6 | 17.7 | 46.4 | 15.4 | 7.3 | 20.4 | 3.3 | 1.0 | 31.9 | 36.0 |
| ResNeXt-101-v1 | 31.8 | 54.5 | 33.5 | 24.3 | 56.9 | 51.6 | 11.4 | 33.2 | 5.3 | 2.7 | 37.1 | 31.3 |
| **ResNeXt-101-v1(p-t)** | **37.9** | 57.6 | 41.1 | 26.9 | 69.0 | 66.5 | **12.2** | 33.5 | 6.1 | 2.6 | 39.1 | 31.9 |

Table 4.5: Mask R-CNN (FPN) bounding box and instance segmentation mask results on test set for each setup.

| | **BBox** | | | | | | **Mask** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Mask R-CNN w/** | AP | AP50 | AP75 | APs | APm | APl | AP | AP50 | AP75 | APs | APm | APl |
| ResNet-101-v1 | 41.4 | 59.1 | 46.4 | 27.9 | 75.7 | 76.5 | 11.5 | 31.3 | 5.7 | 2.1 | 37.1 | 32.2 |
| ResNeXt-101-v1 | 47.3 | 65.4 | 53.5 | 35.3 | 79.3 | 78.4 | 12.5 | 33.2 | 6.9 | 3.1 | 39.4 | 33.4 |
| **ResNeXt-101-v2** | **51.7** | 68.9 | 57.6 | 39.7 | 83.2 | 82.3 | 12.2 | 32.4 | 6.7 | 3.2 | 38.4 | 32.6 |
| **ResNeXt-101-v3** | 51.3 | 64.9 | 57.2 | 35.3 | 84.5 | 85.5 | **15.1** | 37.5 | 9.4 | 4.0 | 46.5 | 42.0 |

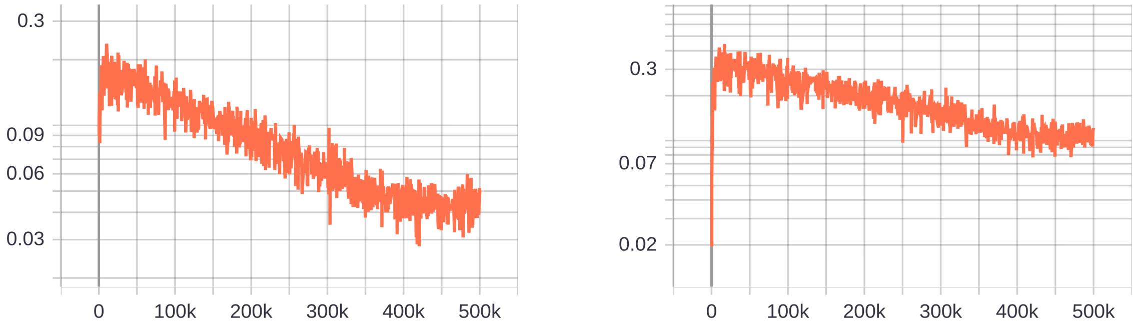
The final model results are shown in Table 4.5. Each network was initialised with pre-trained weights to accelerate convergence due to the long training schedules necessary, and trained up to 300k iterations (around 6 days and 18h) with validations loops of 12k iterations, presenting best validation scores from 180k (ResNet-101-v1) to 240k iterations (ResNeXt-101-v2) with the same training conditions mentioned previously. The latter was further subjected to a longer push of more 200k (around 5 days and 6h) to study the minimum locality; although presenting overfit behaviour since the best checkpoint (240k iter), with the validation scores decreasing steadily up to 500k. The training behaviour of the Mask R-CNN w/ ResNeXt-101-FPN-v2 is shown in the following plots.



(a) RPN classification loss

(b) RPN localization loss

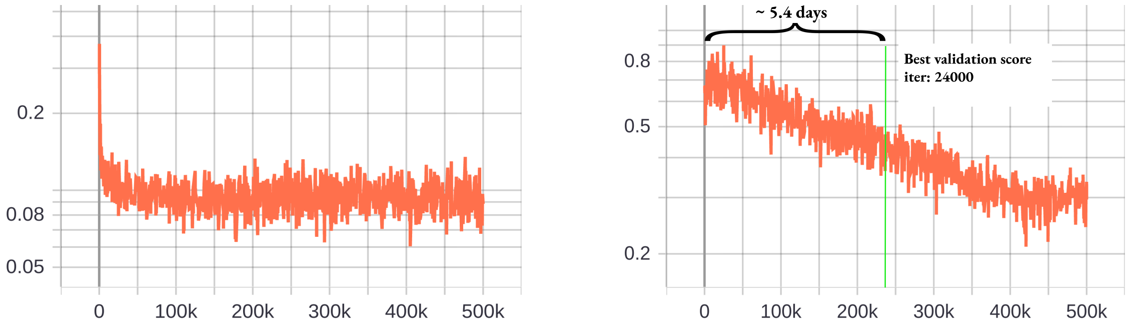
Figure 4.3: Training behaviour evolution of Region Network Proposal.



(a) Box Head classification loss

(b) Box Head regression loss

Figure 4.4: Training behaviour evolution of Box Head.



(a) Mask Head classification loss

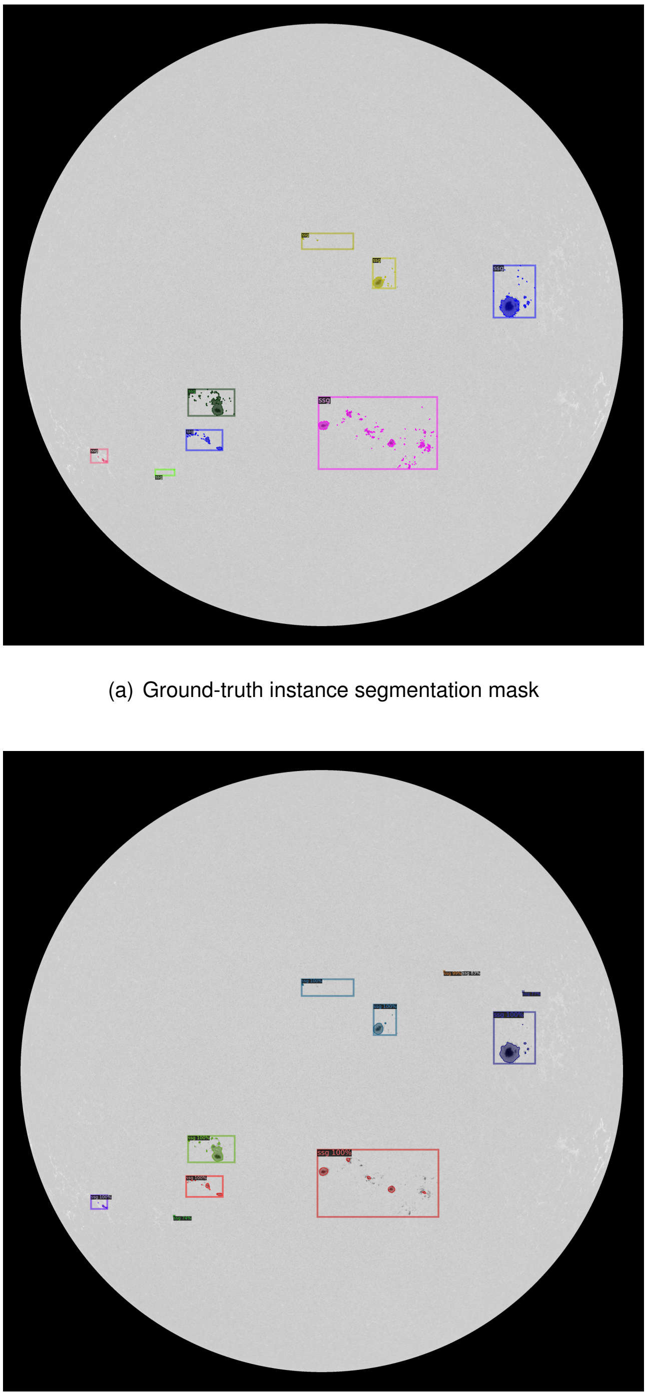
(b) Mask R-CNN total loss

Figure 4.5: Training behaviour evolution of Mask Head and Mask R-CNN with ResNeXt-101-FPN-v2 total loss.

Considering the difficult task of the bounding box regression, the results are quite satisfactory, presenting the best AP score of 51.7 with the v2 setup. The increase of capacity of the fully connected layers (1024 to 2048) showed an improvement of 4.4 points over the v1 setup. The AP difference between v2 and v3 setups can be neglected since they have the same box head configuration. In addition, ResNeXt-101 presents a surprisingly 5.9 points improvement over ResNet-101, which is interesting considering that both have the same capacity as shown in section 2.5.1. Although the results are quite good for medium to large scales, the algorithm presents limitations in accurately predicting smaller groups' bounding boxes. From Figures 4.3 and 4.4, it can be seen that the Region Proposal Network presents much lower losses of box localization than the Box Head regression losses and both look at the same features maps (excluding P6) and are optimised with the same loss function, which indicates that the incapability of the Box Head to predict the small groups bounding boxes accurately might be related to the RoIAlignV2 (or Pool/Align) operation. As a matter of fact, this operation forcibly implies a considerable information loss with a small resolution output (7×7), which turn out to be critical to the extremely small spots that end up being lost after the operation. The configuration of sunspot groups tend to have a dispersion of smaller spots towards the boundaries of their limitation, and since this task massively depends on the instances delimitation structure, the predictions based on features maps that do not possess accurate information of smaller spots are hugely harmed. Although the increase of RoIAlignV2 output resolution for the Box Head was not tested, it is safe to claim that better performances would result, as seen in Mask Head, increasing however the computational price. Figure C.3 shows an example of a bounding box group prediction with a ResNeXt-101-FPN-v3.

Regarding the mask prediction task, from the results it is clear that the algorithm has severe limitations to predict the segmentation mask of sunspot groups accurately. This poor performance is mainly related to the feeble mask head architecture and the RoIAlign operation. From all single components of Mask R-CNN, the Mask Head is by far the one that could be most improved. Its straightforward FCN architecture, which in any case is the foundation of modern semantic segmentation approaches, nowadays is far from the state-of-the-art models' performances [83, 84] and have huge limitations in cases that are needed a high low-level semantic precision, therefore these results are not a surprise.

Similarly to the Box Head, the other aspect has to do with the region of interest alignment operation, which enforces a significant information loss in instances not composed by solid unified bodies, like a common natural object. This aspect can be detected in the AP difference between the v2 and v3 setup, where a double-up in the output resolution of RoIAlignV2 showed an improvement of around 3.0 AP points. In fact, this operation works reasonable for natural objects since they present a singular object with a solid unification, being robust to the interpolation operations in RoIAlign. However, being a sunspot group composed of several spots islands, these operations turn out to be catastrophic, as depicted in Figure 4.7 b). In the green prediction, for instance, it is clear that the model has the tendency to predict unified bodies as it creates a sort of bridge between the two poles of the central spot, which is resultant from the interpolation operations into a small resolution output map. It is visible also that small singular spots are hardly segmented, as shown in Figure 4.7. Furthermore, Table 4.4 shows a ResNeXt-101-v1(p-t) mask AP similar to ResNeXt-101-v1 in Table 4.5, which is expected from the training behaviour presented in Figure 4.5 a), converging quite early and not showing real improvements in a longer training schedule. In addition, Table 4.5 shows that the increase of the number of convolutional layers in the v2 setup presented in reality a decrease in performance in the test set, which also indicates that the poor instance mask performance is not related to the capacity of FCN. More examples of inference samples can be found in Appendix C.



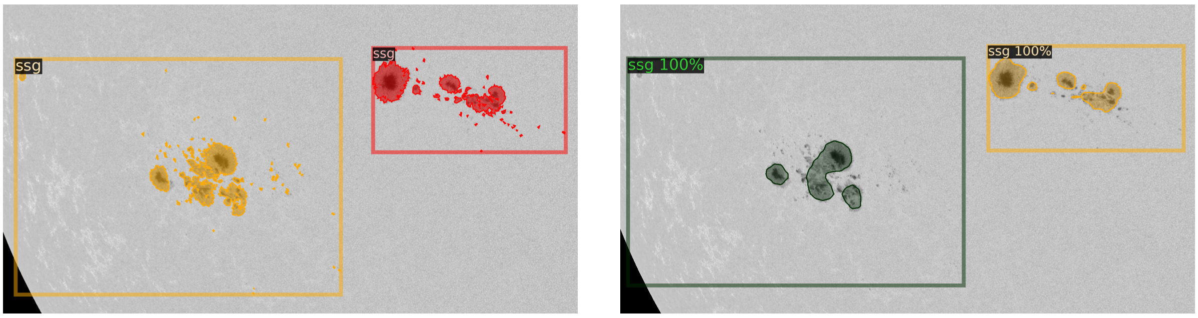
(a) Ground-truth instance segmentation mask

(b) Predicted instance segmentation mask

Figure 4.6: Comparison between an instance segmentation ground-truth sample and its Mask R-CNN w/ ResNeXt-FPN-v3 prediction.

The comparison between the number of groups predictions and the number of ground-truth groups of November and December from 2010 to 2014 (test set) was also conducted using the v2 setup, presented in Table 4.6[2]. The model is able to catch the trend pretty well, although its performance decreases significantly with an increase of solar activity, as expected. In fact, 2013 was a year characterised by

[2] Note that the number of ground-truth groups presented does not represent the monthly international value since the test set contains several daily samples (up to 4 varying).



(a) Ground-truth mask

(b) Predicted mask

Figure 4.7: Comparison between an zoomed area of a instance segmentation ground-truth sample and its Mask R-CNN (ResNeXt-v3) prediction. Better visualization in Appendix C.

Table 4.6: Comparison between the number of ground-truth groups of the test set and the number of groups predicted by the v2 setup.

| | Nº groups | Nº predicted groups | Accuracy |
|---|---|---|---|
| 2010 | 723 | 681 | 94.2% |
| 2011 | 2450 | 2145 | 87.6% |
| 2012 | 1840 | 1458 | 79.2% |
| 2013 | 3872 | 1994 | 51.5% |
| 2014 | 1912 | 2212 | 84.3% |
| Total | 10797 | 8490 | 78.6% |

a massive increase in sunspot numbers and huge groups density, as illustrated in Figure B.1. With a severe decrease in distance between groups, it is also noted a considerable decrease in bounding box regression, naturally because the delimitation between groups is very difficult to infer. This could be improved by providing the algorithm with McIntosh classification for each group; although it would make the Box Head classification task much more demanding, the model would learn the configuration of different groups in a supervised way explicitly, and that should help it to predict groups boundaries better, hence increasing the performance of the regression task.

# Chapter 5

# Conclusions

The presented thesis aimed for the development of an automatic sunspot detection deep learning algorithm. Based on two approaches, semantic and instance segmentation, three algorithms were implemented for sunspot and sunspot group detection purposes, and promising results were achieved.

Additionally, an automated data pipeline procedure for generating the ground-truth dataset based on sunspots catalogues provided by DHO was designed and built to serve as a foundation for both approaches. The importance of this procedure should not be taken lightly as it represents a critical point in both the development and performance of these algorithms. A deep learning model performance can be as good as the data it is fed with; thus, the ground-truth generation needed in this application represents a crucial role.

The implemented U-Net model surpasses two state-of-the-art methods and presents an additional important counterpoint, namely the total removal of calibration procedure needs, characteristic of the algorithms based on pixel intensity. In fact, ground-based and space-borne observations can present differences in image properties that are originated from distinct sources, which sometimes leads to brighter or darker solar records. All three models developed proved to be robust to these events, which is an essential step to a fully autonomous sunspot detection algorithm.

A novelty approach to sunspot group detection was developed by the application of Mask R-CNN to the purpose. It comprises object detection followed by semantic segmentation in a unified model, which is still a difficult challenge in computer vision. Although far from scientific use, the results showed to be auspicious, paving the path for more research and development. The central aspect responsible for that is the region-of-interest alignment operation employed in Mask R-CNN, which implies a considerable input information loss critical for bounding box regression and segmentation tasks. The low-level semantic properties of sunspots require extreme precision for the accurate detection of tiny spots; if this aspect is not reasonably performed, the bounding box regression task will have severe limitations to make accurate predictions since the boundaries of sunspot groups are typically composed of smaller spots. Furthermore, when mapped into a fixed-size small feature map, this operation creates inevitable losses more significant to the small spots islands spread around the group, making them hardly segmented by the Mask Head. Even with these limitations, the best fine-tuned Mask R-CNN setup version could achieve good precision in the prediction of number of sunspot group in the five years span.

## 5.1 Future Work

The purpose of this thesis was successfully accomplished. From the ground-truth generation to the sunspot detection, every integral step was employed, and the final goal was reached. However, much more must still be done to make these algorithms entirely reliable for scientific use.

The first crucial aspect to look at it is the data itself. The foundation of this work had to be cemented in a somewhat uneven ground, where the ground-truth annotations were in the first place created by an algorithm developed by the DHO, therefore some imprecision might be associated with them. In addition, the adopted procedure to generate the ground-truth mask might itself present some irregularities. The area expansion was performed based on pixel intensity level in the neighbourhood of the supposed centre coordinate. While this procedure may be highly accurate most of the time, the delimitation of sunspot boundaries and particularly minor spots may present some inaccuracies. In addition, a natural improvement would be to increase the dataset size as far as possible.

In regard to the semantic segmentation approach, several experiments can be further developed. U-Net can be built with a more robust backbone, for example with ResNeXt-101 or EfficientNet (2019). Additionally, the input rescale factors should be studied to evaluate the impact on this models' performance. It was found here that the best U-Net model performance was achieved with an input resize to $2048^2$ pixels followed by a random crop of $1024^2$, presenting a gap of 12% IoU points to the best U-Net model of an effective network input of $512^2$ pixels resultant from a resize to $1024^2$ followed by a random crop. Ideally, no resize factor should be employed, although naturally, the computational price will ghastly increase. To address this problem, the training pipeline can be optimised by taking the most from each iteration. Without resizing applied, the input can be divided into several fixed-size patches, and only the patches that contain spots, or that contain a number of pixels that belong to sunspots greater than a specific threshold, are further sent to the network for training. This approach would overcome the GPU memory limitations and make each iteration more efficient since the greatest portion of the input is background. These considerations are naturally valid into U-Net 3+ as well, and in fact, it is the architecture from the two that will result in better performances as shown. U-Net 3+ implemented architecture in this work followed the official release, the same backbones mentioned previously for U-Net will certainly lead to better performances if applied on U-Net 3+ as well. In addition, deep supervision with the classification-guided module along with the hybrid loss also presented in [40] can be applied here to reach better results.

Concerning the instance segmentation approach, other alternatives to Mask R-CNN ought to be pursued. The consequences of RoIAlign(V2/Pool) are critical to accurate sunspot group detection. Even with a replacement of Mask Head with a more robust architecture that would naturally lead to a significant improvement of the segmentation mask results, ultimately, the performances will not be as good as desirable. The operation is not suited for this application, where low-level precision is vital to detect tiny spots. The interpolations occur in the proposed regions where the background is dominant, and since the pixel intensity of sunspots only composed by penumbra is not sufficiently high enough, they are simply lost during the process.

In any case, from the different techniques for instance segmentation mentioned in section 2.5, detection followed by segmentation should still be the one more adequate to this application – if a unified model is desired. As a matter of fact, hybrid approaches can be very effective. For example, an encoder-decoder architecture (as U-Net) to perform pixel-wise classification followed by an improved Region Proposal Network to bounding box regression could be an interesting approach to this problem, where the two models would be used in sequence, and RPN would leverage not only from the input image but also from the segmentation mask output from the first model. Another similar option, would be to replace RPN with a traditional clustering method, for example, DBSCAN adjusted to the problem, leveraging as well from the already predicted segmentation masks.

A natural extension to the semantic approach is to perform the umbra and penumbra detection instead of treating a sunspot as a whole, but for that, the importance of an extremely accurate ground-truth dataset is even greater. Moreover, the McIntosh group classification can also be employed in the instance segmentation approach. For that, this information should be extracted from the NOAA/SWPC, which provides daily solar region summaries (SRS), and from the NOAA active regions numbers, the bridge to DPD/HMIDD can be established to each sunspot group, making it therefore a multi-class instance segmentation problem.

There is a solid conviction that a fully autonomous deep learning algorithm for sunspot detection that surpasses the current state-of-the-art approaches performances will rise in the near future. Complete automation of such matters should already be accomplished by now. The benefits are clear, either for scientific use and educational purposes. What has been accomplished so far by the DHO is notable, but several aspects can still be improved in regard to the sunspot catalogues and the algorithms that generate them. Being an initial deep learning approach to this problem, this work hopes to push the existing boundaries further and serve as a foundation for future research.

# Bibliography

[1] H. Hayakawa, K. Iwahashi, H. Tamazawa, Y. Ebihara, A. Kawamura, H. Isobe, K. Namiki, and K. Shibata. *Records of auroral candidates and sunspots in Rikkokushi, chronicles of ancient Japan from early 7th century to 887.*, volume 69. Publications of the Astronomical Society of Japan, 2017. https://doi.org/10.1093/pasj/psx087.

[2] J. M. Vaquero. Historical sunspot observations: a review. *Advances in Space Research*, 40(7): 929–941, 2007.

[3] J. Thomas and N. Weiss. *Sunspots and Starspots*. Cambridge University Press, 2008. ISBN 978-0-521-86003-1.

[4] L. Golub and J. M. Pasachoff. *Nearest Star - The Surprising Science of Our Sun*. Cambridge University Press, 2014. ISBN 978-1-107-67264-2.

[5] D. T. Oyedokun and P. J. Cilliers. Chapter 16 - geomagnetically induced currents: A threat to modern power systems. In A. F. Zobaa, S. H. Abdel Aleem, and A. Y. Abdelaziz, editors, *Classical and Recent Aspects of Power System Optimization*, pages 421–462. Academic Press, 2018. ISBN 978-0-12-812441-3. doi: https://doi.org/10.1016/B978-0-12-812441-3.00016-1. URL `https://www.sciencedirect.com/science/article/pii/B9780128124413000161`.

[6] C. Verbeeck, P. A. Higgins, T. Colak, F. T. Watson, V. Delouille, B. Mampaey, and R. Qahwaji. A multi-wavelength analysis of active regions and sunspots by comparison of automatic detection algorithms. *Solar Physics*, 283(1):67–95, 2013.

[7] D. J. Mullan. *Physics of the Sun - A First Course*. CRC Press, 2010. ISBN 978-1-4200-8308-8.

[8] Guenther, D. B., Demarque, P., Kim, Y.-C., . Pinsonneault, and M. H. *Standard solar model.* The American Astronomical Society, 1992.

[9] S. Turck-Chièze. The standard solar model and beyond. *Journal of Physics: Conference Series*, 665, 01 2016. doi:10.1088/1742-6596/665/1/012078.

[10] E. S. A. CESAR. Sun's differential rotation student's guide – intermediate level cesar's science case. `https://cesar.esa.int/upload/201710/suns_differential_rotation_students_guide_intermediate_level_008.pdf`.

[11] R. C. Carrington. *Observations of the spots on the sun [microform] : from November 9, 1853, to March 24, 1861, made at Redhill / by Richard Christopher Carrington.* Williams and Norgate, London, 1863.

[12] P. S. McIntosh. The classification of sunspot groups. *Solar Physics*, 125(2):251–267, 1990.

[13] W. D. C. SILSO. The international sunspot number. *International Sunspot Number Monthly Bulletin and online catalogue*. URL `http://www.sidc.be/silso/`.

[14] T. Chen, J. Lih, T. Chang, C. Yang, M. Lu, C.J.Liu, and M. Ho. Using sidereal rotation period expressions to calculate the sun's rotation period through observation of sunspots. *Mathematical Problems in Engineering*, 2015, 2015. doi: 10.1155/2015/153407.

[15] W. Thompson. Coordinate systems for solar image data. *Astronomy & Astrophysics*, 449(2):791–803, 2006.

[16] S. Zharkov, V. Zharkova, S. Ipson, and A. Benkhalil. Technique for automated recognition of sunspots on full-disk solar images. *Journal on Advances in Signal Processing*, 318462, 2005. doi: https://doi.org/10.1155/ASP.2005.2573.

[17] V. Jewalikar and S. Singh. Automated sunspot extraction, analysis and classification. *International Conference on Image and Video Processing and Computer Vision (IVPCV-10)*, 2010.

[18] U. Dasgupta, S. Singh, and V. Jewalikar. Sunspot number calculation using clustering. *Third National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics*, pages 171–174, 2011.

[19] S. Zharkov, V. Zharkova, S. Ipson, and A. Benkhalil. Statistical properties of sunspots in 1996-2004. *Solar Physics*, 228:377–397, 2005.

[20] J. Curto, M. Blanca, and E. Martínez. Automatic sunspots detection on full-disk solar images using mathematical morphology. *Solar Physics*, 250(2):411–429, 2008.

[21] S. H. Nguyen, T. T. Nguyen, and H. S. Nguyen. Rough set approach to sunspot classification problem. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, volume 3642, pages 263–272. Springer, 2005.

[22] R. Qahwaji and T. Colak. Hybrid imaging and neural networks techniques for processing solar images. *International journal of computers and their applications*, 13:9–16, 2006.

[23] T. I. Manish, D. Murugan, and T. G. Kumar. Automatic detection of sunspot activities using advanced detection model. *IOSR Journal of Computer Engineering*, 16:83–87, 2014.

[24] E. Shahamatnia, I. Dorotovič, J. M. Fonseca, and R. A. Ribeiro. An evolutionary computation based algorithm for calculating solar differential rotation by automatic tracking of coronal bright points. *Journal of Space Weather and Space Climate*, 6:A16, Mar. 2016. doi: 10.1051/swsc/2016010.

[25] D. Djafer, A. Irbah, and M. Meftah. Identification of sunspots on full-disk solar images using wavelet analysis. *Solar Physics*, 281:863–875, 2012. doi: 10.1007/s11207-012-0109-3.

[26] L. Yu, L. Deng, and S. Feng. Automated sunspot detection using morphological reconstruction and adaptive region growing techniques. In *Proceedings of the 33rd Chinese Control Conference*, pages 7168–7172, 2014. doi: 10.1109/ChiCC.2014.6896184.

[27] V. Zharkova, S. IPSON, A. Benkhalil, and S. Zharkov. Feature recognition in solar images. *Artificial Intelligence Review*, 23:209–266, 2005. https://doi.org/10.1007/s10462-004-4104-4.

[28] P. A. Higgins, P. T. Gallagher, R. J. McAteer, and D. S. Bloomfield. Solar magnetic feature detection and tracking for space weather monitoring. *Advances in Space Research*, 47(12):2105–2117, 2010.

[29] T. Colak and R. Qahwaji. Automated solar activity prediction: a hybrid computer platform using machine learning and solar imaging for automated prediction of solar flares. *Space Weather*, 7(6), 2009. doi: https://doi.org/10.1029/2008SW000401.

[30] F. Watson, L. Fletcher, S. Dalla, and S. Marshall. Modelling the longitudinal asymmetry in sunspot emergence: the role of the wilson depression. *Solar Physics*, 260(1):5–19, 2009.

[31] S. Carvalho, P. Pina, T. Barata, R. Gafeira, and A. Garcia. Ground-based Observations of Sunspots from the Observatory of Coimbra: Evaluation of Different Automated Approaches to Analyse its Datasets. In I. Dorotovic, C. E. Fischer, and M. Temmer, editors, *Coimbra Solar Physics Meeting: Ground-based Solar Observations in the Space Instrumentation Era*, volume 504 of *Astronomical Society of the Pacific Conference Series*, page 125, Apr. 2016.

[32] C. Zhao, G. Lin, Y. Deng, and X. Yang. Automatic recognition of sunspots in hsos full-disk solar images. *Publications of the Astronomical Society of Australia*, 33:e018, 2016. doi: 10.1017/pasa.2016.17.

[33] S. Carvalho, S. Gomes, T. Barata, A. Lourenço, and N. Peixinho. Comparison of automatic methods to detect sunspots in the coimbra observatory spectroheliograms. *Astronomy and Computing*, 32:100385, 2020. ISSN 2213-1337. doi: https://doi.org/10.1016/j.ascom.2020.100385. URL `https://www.sciencedirect.com/science/article/pii/S2213133720300391`.

[34] E.Fini. A deep learning approach to sunspot detection and counting. Master's thesis, Politecnico di Milano, 2019.

[35] E. Stevens, L. Antiga, and T. Viehmann. *Deep Learning with PyTorch*. 2020. ISBN 9781617295263. URL `https://pytorch.org/assets/deep-learning/Deep-Learning-with-PyTorch.pdf`.

[36] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding, 2016.

[37] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[38] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. Unet++: A nested u-net architecture for medical image segmentation. *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support : 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, held in conjunction with MICCAI 2018, Granada, Spain, S...*, 11045: 3–11, 2018.

[39] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-Supervised Nets. In G. Lebanon and S. V. N. Vishwanathan, editors, *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pages 562–570, San Diego, California, USA, 09–12 May 2015. PMLR. URL `https://proceedings.mlr.press/v38/lee15a.html`.

[40] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu. Unet 3+: A full-scale connected unet for medical image segmentation. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1055–1059, 2020.

[41] M. Bai and R. Urtasun. Deep watershed transform for instance segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2858–2866, 2017.

[42] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother. Instancecut: From edges to instances with multicut. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7322–7331, 2017.

[43] A. Arnab and P. H. S. Torr. Pixelwise instance segmentation with a dynamically instantiated network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 879–888, 2017.

[44] P. H. O. Pinheiro, R. Collobert, and P. Dollár. Learning to segment object candidates. In *NIPS*, 2015.

[45] P. H. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollár. Learning to refine object segments. In *ECCV*, 2016.

[46] J. Dai, K. He, Y. Li, S. Ren, and J. Sun. Instance-sensitive fully convolutional networks. In *ECCV*, 2016.

[47] X. Chen, R. B. Girshick, K. He, and P. Dollár. Tensormask: A foundation for dense object segmentation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2061–2069, 2019.

[48] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[49] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3150–3158, 2016.

[50] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. H. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár. A multipath network for object detection. *ArXiv*, abs/1604.02135, 2016.

[51] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4438–4446, 2017.

[52] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, Los Alamitos, CA, USA, oct 2017. IEEE Computer Society. doi: 10.1109/ICCV.2017.322. URL `https://doi.ieeecomputersociety.org/10.1109/ICCV.2017.322`.

[53] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39: 1137–1149, 2015.

[54] A. M. Hafiz and G. M. Bhat. A survey on instance segmentation: state of the art. *International Journal of Multimedia Information Retrieval*, 9(3):171–189, Jul 2020. ISSN 2192-662X. doi: 10.1007/s13735-020-00195-x. URL `http://dx.doi.org/10.1007/s13735-020-00195-x`.

[55] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[56] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[57] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944, 2017.

[58] R. B. Girshick. Fast r-cnn. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[59] T. Baranyi, L. Győri, and A. Ludmány. On-line tools for solar data compiled at the debrecen observatory and their extensions with the greenwich sunspot data. *Solar Physics*, 291(9-10):3081–3102, Aug 2016. ISSN 1573-093X. doi: 10.1007/s11207-016-0930-1. URL `http://dx.doi.org/10.1007/s11207-016-0930-1`.

[60] L. Győri. Automation of area measurement of sunspots. *Solar Physics*, 180(109), 1998. doi: 10.1023/A:1005081621268.

[61] L. Győri. Automated determination of the alignment of solar images. *Hvar Obs*, 180(299), 2005.

[62] R. C. f. A. Debrecen Heliophysical Observatory and E. Sciences. Revised version of greenwich photoheliographic results (gpr) sunspot database. `http://fenyi.solarobs.csfk.mta.hu/en/databases/GPR/`.

[63] S. J. Mumford, S. Christe, D. Pérez-Suárez, J. Ireland, A. Y. Shih, A. R. Inglis, S. Liedtke, R. J. Hewett, F. Mayer, K. Hughitt, et al. Sunpy - python for solar physics. *Computational Science & Discovery*, 8(1), 2015.

[64] Y. Liu, L. Chu, G. Chen, Z. Wu, Z. Chen, B. Lai, and Y. Hao. Paddleseg: A high-efficient development toolkit for image segmentation, 2021.

[65] Y. Ma, D. Yu, T. Wu, and H. Wang. Paddlepaddle: An open-source deep learning platform from industrial practice. *Frontiers of Data and Domputing*, 1(1):105, 2019. doi: 10.11871/jfdc.issn.2096.742X.2019.01.011. URL `http://www.jfdc.cnic.cn/EN/abstract/article_2.shtml`.

[66] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. *ArXiv*, abs/1506.04579, 2015.

[67] P. Mishra and K. Sarawadekar. Polynomial learning rate policy with warm restart for deep neural network. In *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, pages 2087–2092, 2019. doi: 10.1109/TENCON.2019.8929465.

[68] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. *Lecture Notes in Computer Science*, page 240–248, 2017. ISSN 1611-3349. doi: 10.1007/978-3-319-67558-9_28. URL `http://dx.doi.org/10.1007/978-3-319-67558-9_28`.

[69] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. `https://github.com/facebookresearch/detectron`, 2018.

[70] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[71] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[72] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2015.

[73] H. Jung, B. Lodhi, and J. Kang. An automatic nuclei segmentation method based on deep convolutional neural networks for histopathology images. *BMC Biomedical Engineering*, 1(24), 2019. ISSN 2524-4426.

[74] J. Shu, F. Nian, M. Yu, and X. Li. An improved mask r-cnn model for multiorgan segmentation. *Mathematical Problems in Engineering*, 2020. doi: 10.1155/2020/8351725. URL `https://doi.org/10.1155/2020/8351725`.

[75] H. Almubarak, Y. Bazi, and N. Alajlan. Two-stage mask-rcnn approach for detecting and segmenting the optic nerve head, optic disc, and optic cup in fundus images. *Applied Sciences*, 10(11), 2020. ISSN 2076-3417. doi: 10.3390/app10113833. URL `https://www.mdpi.com/2076-3417/10/11/3833`.

[76] O. L. F. d. Carvalho, O. A. de Carvalho Júnior, A. O. d. Albuquerque, P. P. d. Bem, C. R. Silva, P. H. G. Ferreira, R. d. S. d. Moura, R. A. T. Gomes, R. F. Guimarães, and D. L. Borges. Instance segmentation for large, multi-channel remote sensing imagery using mask-rcnn and a mosaicking approach. *Remote Sensing*, 13(1), 2021. ISSN 2072-4292. doi: 10.3390/rs13010039. URL `https://www.mdpi.com/2072-4292/13/1/39`.

[77] Q. Wu, D. Feng, C. Cao, X. Zeng, Z. Feng, J. Wu, and Z. Huang. Improved mask r-cnn for aircraft detection in remote sensing images. *Sensors*, 21(8), 2021. ISSN 1424-8220. doi: 10.3390/s21082618. URL `https://www.mdpi.com/1424-8220/21/8/2618`.

[78] P. Monchot, L. Coquelin, K. Guerroudj, N. Feltin, A. Delvallée, L. Crouzier, and N. Fischer. Deep learning based instance segmentation of titanium dioxide particles in the form of agglomerates in scanning electron microscopy. *Nanomaterials (Basel)*, 12(4), 2021. doi: 10.3390/nano11040968.

[79] T. Naets, M. Huijsmans, P. Smyth, L. Sorber, and G. de Lannoy. A mask r-cnn approach to counting bacterial colony forming units in pharmaceutical development, 2021.

[80] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[81] K. He, R. B. Girshick, and P. Dollár. Rethinking imagenet pre-training. *CoRR*, abs/1811.08883, 2018. URL `http://arxiv.org/abs/1811.08883`.

[82] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[83] B. Li, Y. Shi, Z. Qi, and Z. Chen. A survey on semantic segmentation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1233–1240, Los Alamitos, CA, USA, nov 2018. IEEE Computer Society. doi: 10.1109/ICDMW.2018.00176. URL `https://doi.ieeecomputersociety.org/10.1109/ICDMW.2018.00176`.

[84] S. Minaee, Y. Boykov, F. M. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2021.

[85] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ArXiv*, abs/1905.11946, 2019.

[86] T. Baranyi, L. Győri, and A. Ludmány. On-line tools for solar data compiled at the debrecen observatory and their extensions with the greenwich sunspot data. *Solar Physics*, 291(9-10):3081–3102, Aug 2016. ISSN 1573-093X. doi: 10.1007/s11207-016-0930-1. URL `http://dx.doi.org/10.1007/s11207-016-0930-1`.

# Appendix A

# Sunspot Groups McIntosh Classification Example Images

Here is presented an illustration of the McIntosh Classification of sunspot groups from SDO images.

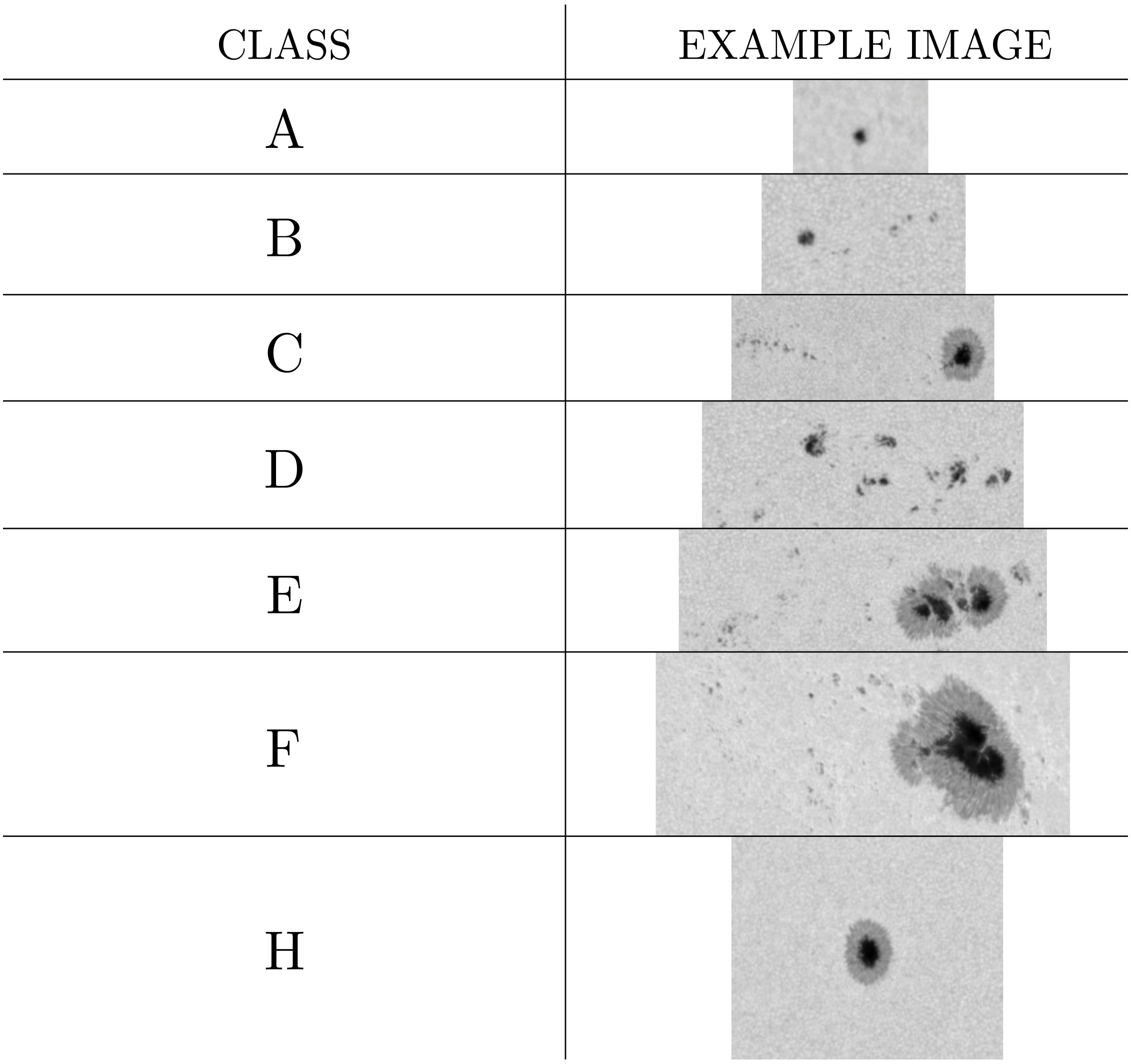


Figure A.1: Sunspot group class example images taken from SDO for each McIntosh class. [34]

# Appendix B

# Ground-truth Data Examples

Here are presented two examples of ground-truth data. An annotated mask visualization provided by HMIDD of a record that possesses a high sunspot groups density, and a full-disk superimposed ground-truth mask sample.

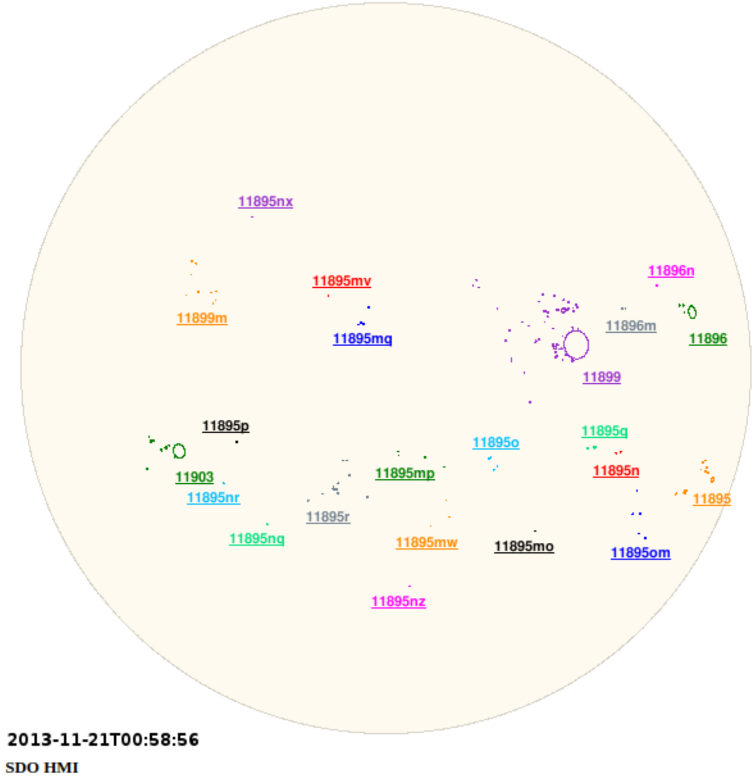


Figure B.1: Annotated mask visualization provided by HMIDD from 21/11/2013. [86]

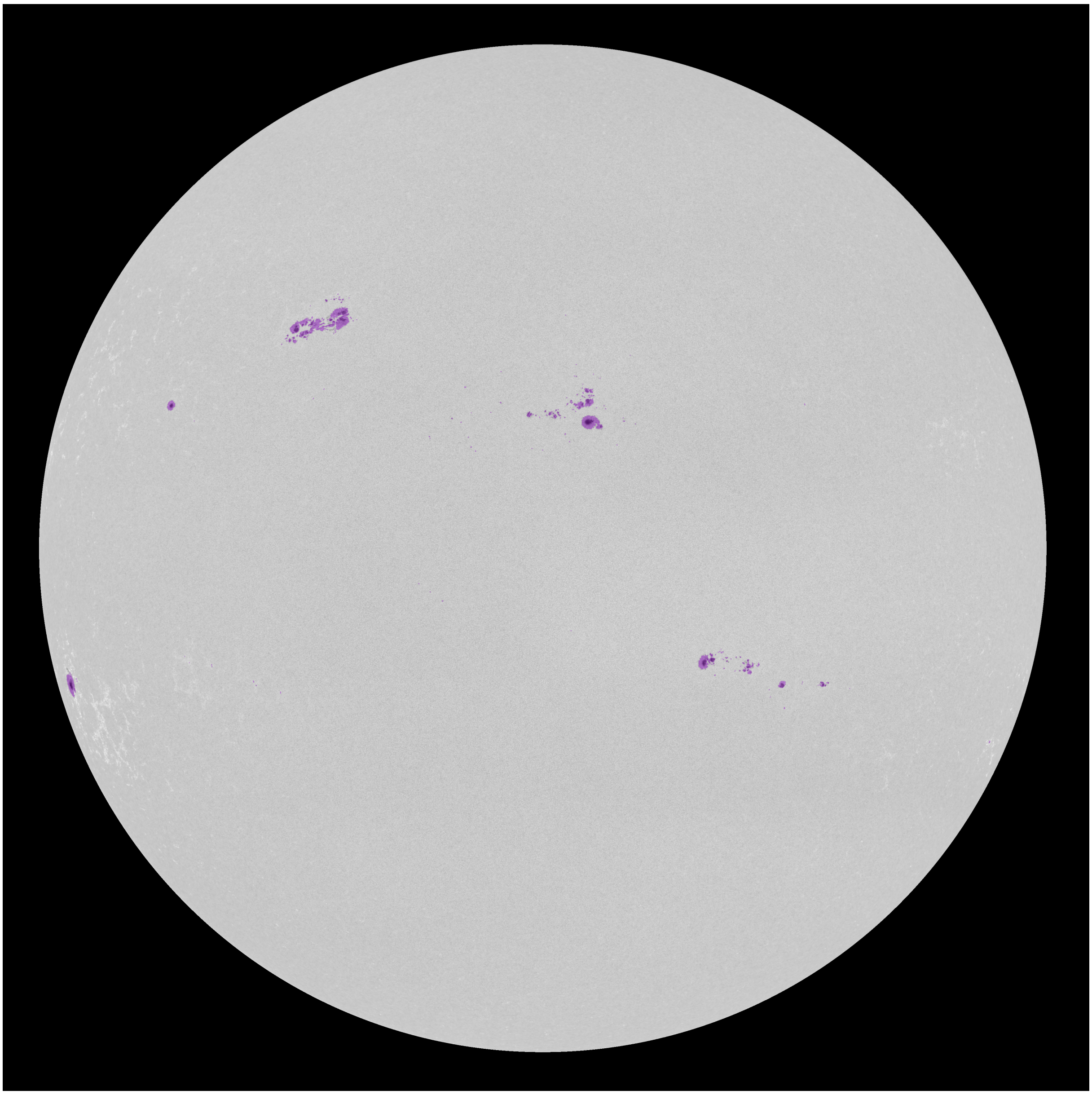
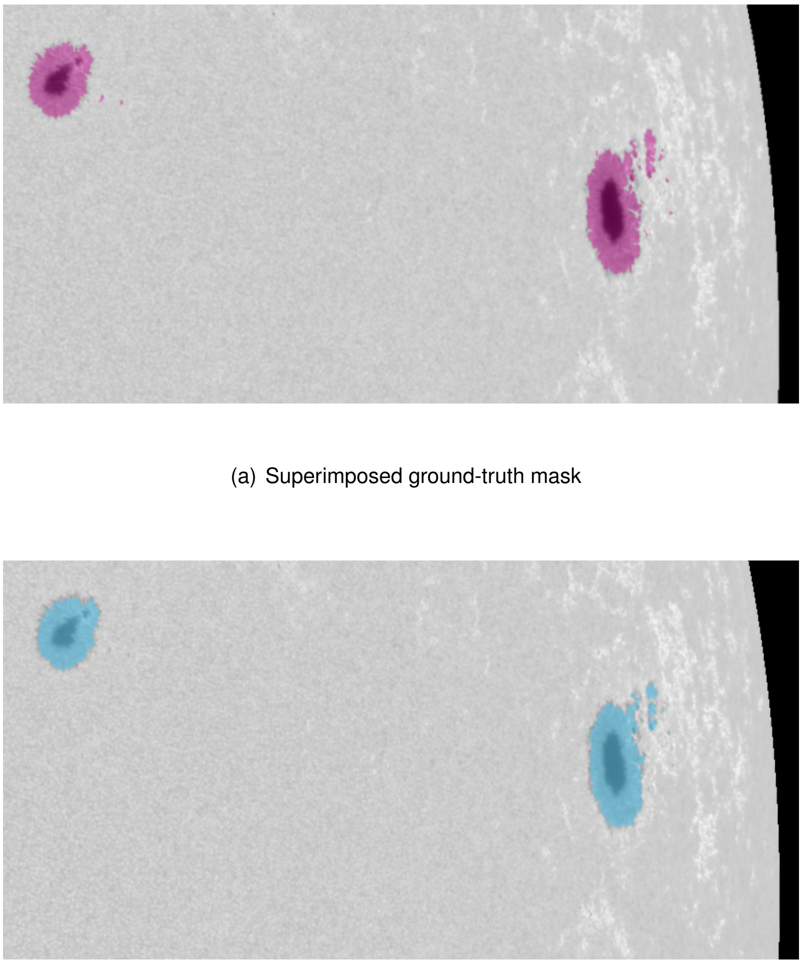


Figure B.2: Superimposed ground-truth generated semantic mask.

# Appendix C

# Inference Samples

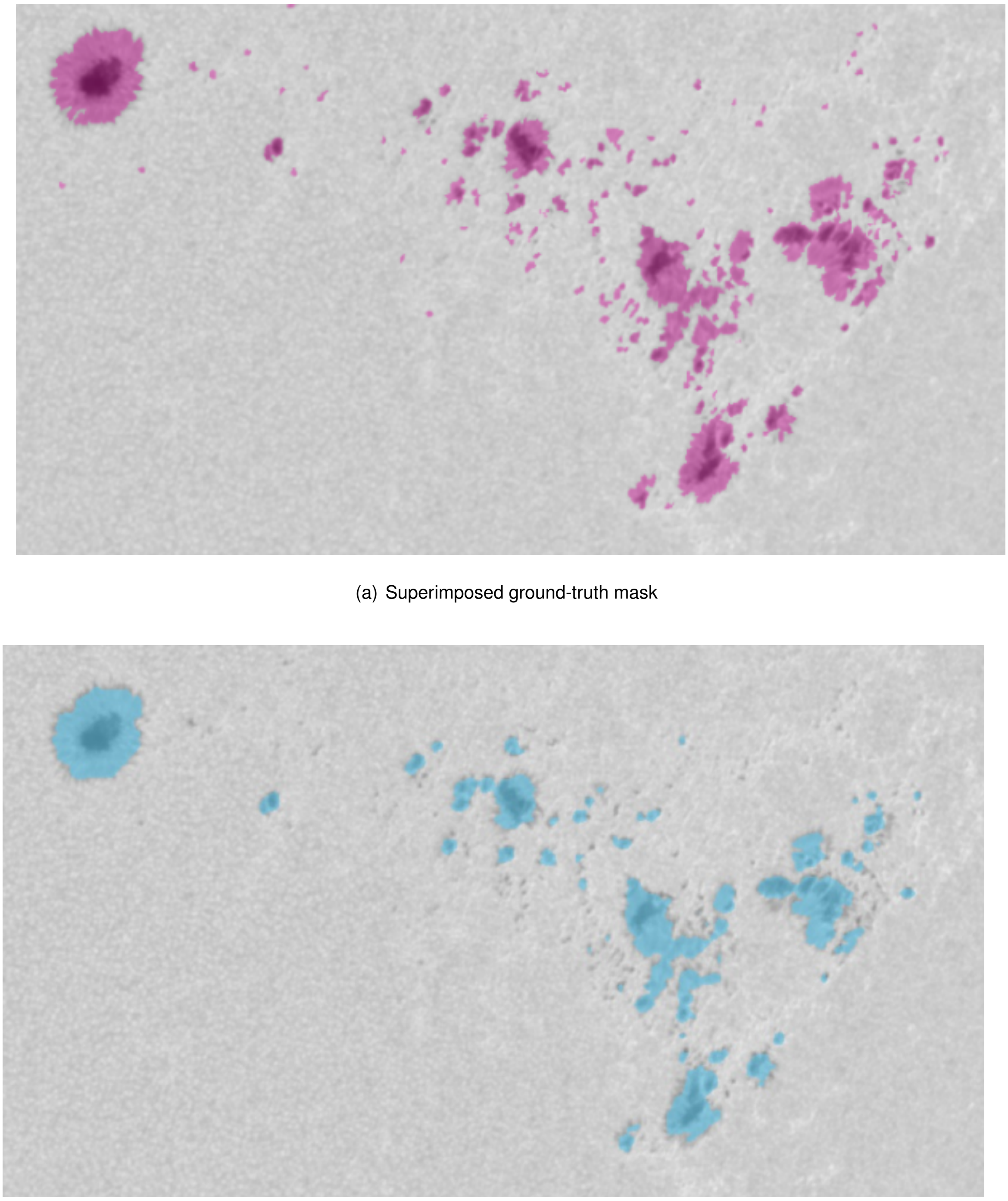
Here are presented some examples of predictions on the test set of U-Net and Mask R-CNN w/ ResNeXt-101-v3.



(a) Superimposed ground-truth mask

(b) Superimposed predicted mask

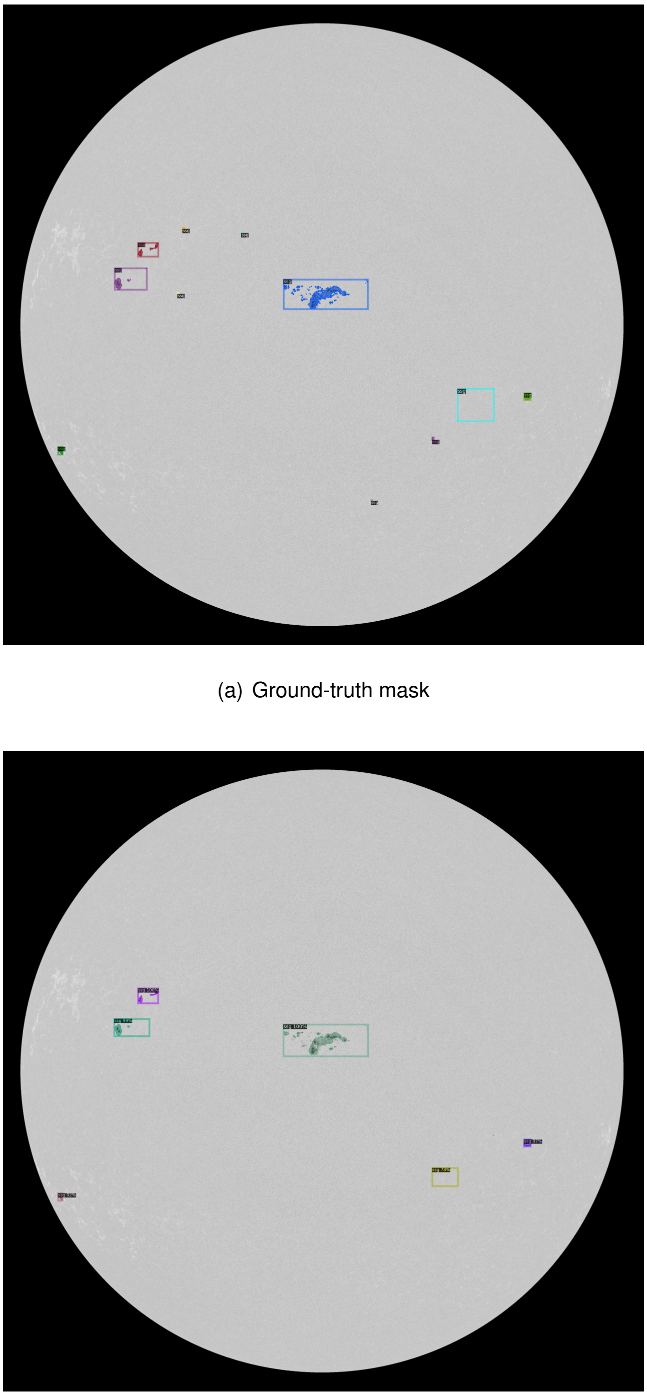
Figure C.1: Comparison between the ground-truth mask and the predicted mask by the best U-Net model in a disk zoomed area.



(a) Superimposed ground-truth mask

(b) Superimposed predicted mask

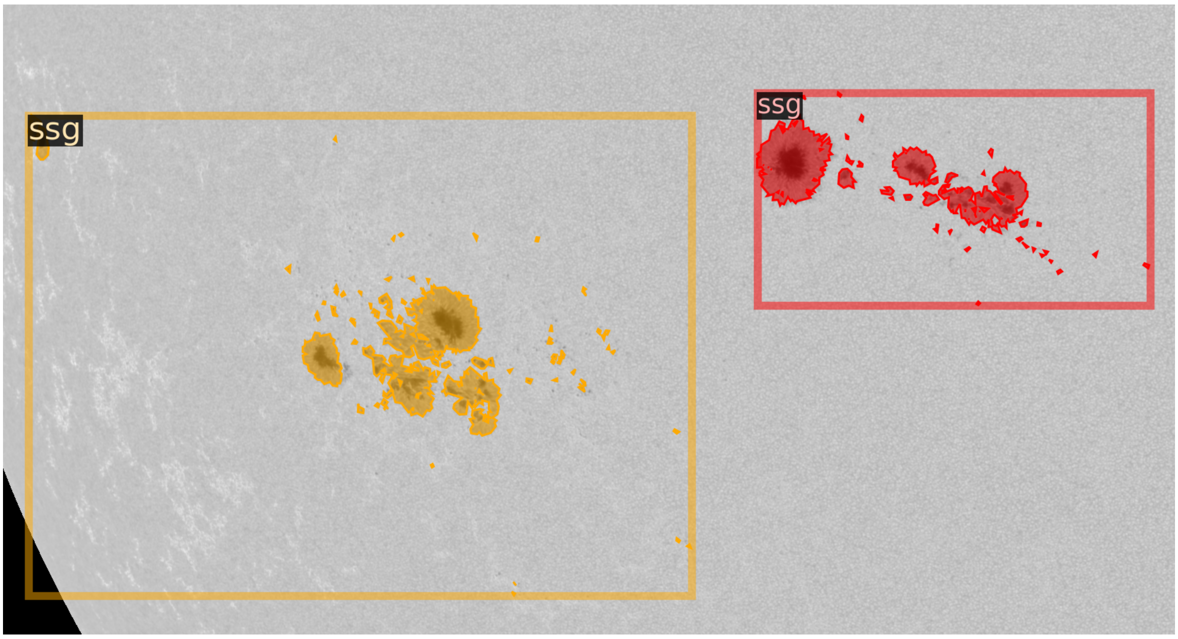
Figure C.2: Comparison between the ground-truth mask and the predicted mask by the best U-Net model in a disk zoomed area.



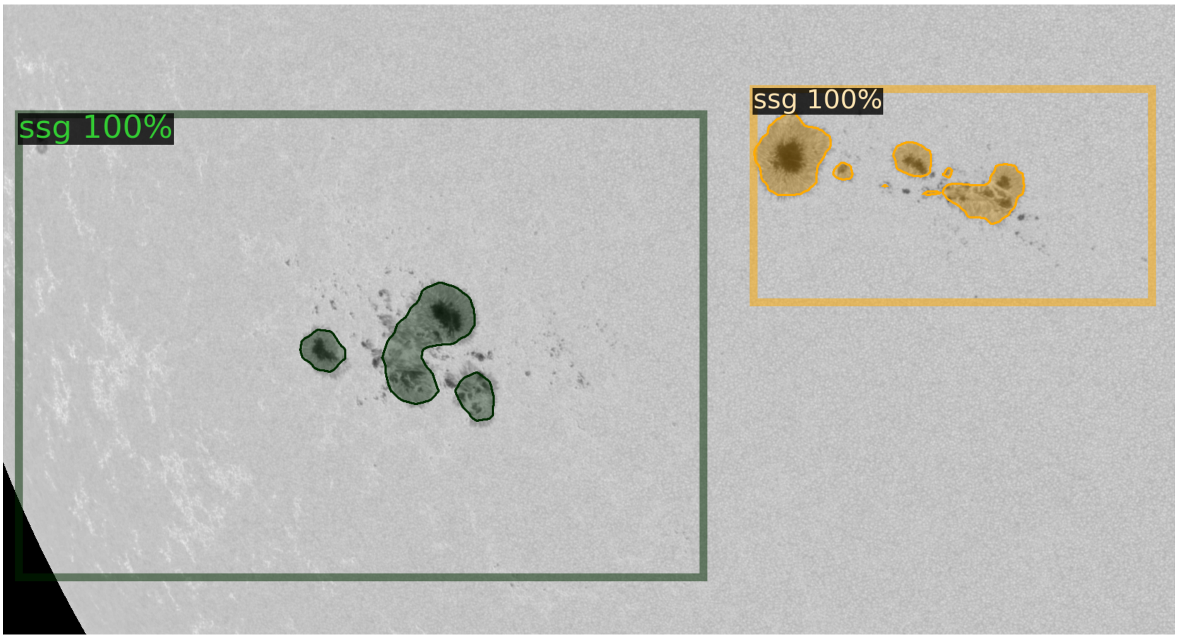
(a) Ground-truth mask

(b) Predicted mask

Figure C.3: Comparison between an instance segmentation ground-truth sample and its Mask R-CNN w/ ResNeXt-FPN-v3 prediction.



(a) Ground-truth instance segmentation mask



(b) Predicted instance segmentation mask

Figure C.4: Comparison between an instance segmentation ground-truth sample and its Mask R-CNN w/ ResNeXt-FPN-v3 prediction.