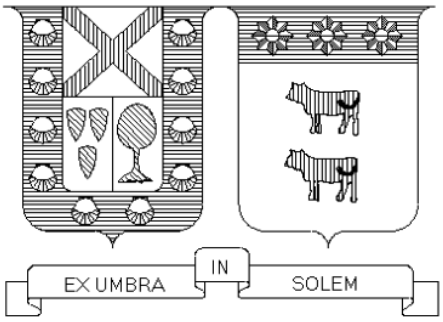
UNIVERSIDAD TECNICA FEDERICO SANTA MARIA
DEPARTAMENTO DE INFORMATICA



TENSOR MASK R-CNN: A TENSOR METHODS AND DEEP LEARNING APPROACH TO MULTI-BAND MORPHOLOGICAL SEGMENTATION AND CLASSIFICATION OF GALAXIES

HUMBERTO ANDRES FARIAS AROCA

Tesis para optar la grado de
Doctor en Ingeniería Informática

Valparaíso, Chile

Abril, 2023

TITULO DE LA TESIS:

TENSOR MASK R-CNN: A TENSOR METHODS AND DEEP LEARNING APPROACH TO MULTI-BAND MORPHOLOGICAL SEGMENTATION AND CLASSIFICATION OF GALAXIES

AUTOR:

HUMBERTO ANDRES FARIAS AROCA

TRABAJO DE GRADO, presentado en cumplimiento parcial de los requisitos para el Grado de Doctor en Ingeniería Informática de la Universidad Técnica Federico Santa María.

Prof. Mauricio Solar, Dr
Universidad Federal de Rio de Janeiro (UFRJ, Brasil). _____
Profesor Guia

Prof. Mauricio Araya, Dr
Université de Lorraine, Nancy, France. _____
Correferente interno

Prof. Diego Mardones, Ph.D
U. de Harvard, EEUU _____
Correferente Externo Nacional

Prof. Bruno Merin, Ph.D
Universidad Autónoma de Madrid, España _____
Correferente Externo Internacional

Prof. Hernán Astudillo, Ph.D
Georgia Institute of Techonolgy, EEUU _____
Presidente comisión

Valparaíso, Chile

Abril, 2023

A mi Familia por darme esas palabras de aliento cuando fue necesario, y al Dr. Solar por su guía sin la cual no hubiera podido recorrer este camino.

ABSTRACT

In order to understand the formation and evolution of galaxies, it is essential to classify them according to their morphology. The growing volume of astronomical data has motivated the application of advanced computer vision techniques for this classification. It is particularly noteworthy that Deep Learning models have demonstrated outstanding results in the automation of this classification. Despite the satisfactory results, the architectures focused on specific tasks rather than a single pipeline. One of the challenges of this pipeline is the fact that it must address the real complexity of the problem, which on the one hand is the fact that current models are based on RGB images instead of multiband photometric observations which present a richer set of information. There is a need to model latent structures not only at the level of individual bands but also at the level of information distributed in the third dimension, such as wavelets or relationships between bands. In order to develop a single pipeline that addresses the tasks of classification, location, and morphological segmentation, the speed of these models must also be considered. In addition to its processing speed, it is also important to consider its computational efficiency. In order to achieve this objective, it is essential to design architectures that take computational efficiency into account in their implementation. This challenge is addressed by Tensor Mask R-CNN, a pipeline for regional convolutional neural-based galaxy morphology classification, localization, and segmentation, as well as the application of tensor algebra methods on the mask R-CNN backbone's convolutional layers to accelerate this deep learning architecture.

RESUMEN

La clasificación de galaxias en función de su morfología es fundamental para la comprensión de la formación y evolución de estas. Esta clasificación en el contexto del creciente volumen de datos astronómicos ha motivado la aplicación de técnicas avanzadas de visión computacional. Destacan especialmente los resultados de modelos de Deep Learning que han mostrado resultados sobresalientes en la automatización de esta clasificación. Pero a pesar de los resultados satisfactorios, las arquitecturas utilizadas se han centrado en tareas específicas y no en un pipeline único. Este pipeline debe abordar la complejidad real del problema, por un lado está el hecho que los actuales modelos trabajan sobre imágenes (RGB) y no han modelado la riqueza de la información presente en las observaciones fotométricas multibanda. Por lo que es necesario enfoques que modelan las estructuras latentes no sólo a nivel de bandas individuales, si no también en la información que está distribuida en la tercera dimensión que puede ser wavelets o las relaciones entre bandas. El otro aspecto que se debe considerar en la implementación de un único pipeline que aborde las tareas de clasificación, localización y segmentación morfológica es el speed de estos modelos que son parte de él. Esto no solo considerando la velocidad de su procesamiento si no también su eficiencia computacional. Disponer de arquitecturas que consideren en su implementación la eficiencia computacional dado el volumen y dimensionalidad de los datos astronómicos es clave para el logro del objetivo. Para lograr estos desafío, proponemos Tensor Mask R-CNN un pipeline único de clasificación, localización y segmentación morfología de galaxias basado neuronales convolucionales regionales y la aplicación de métodos de álgebra tensorial sobre capas convolucionales del backbone de la red Mask R-CNN con el fin de acelerar esta arquitectura de Deep Learning.

INDICE

INDICE DE TABLAS

INDICE DE FIGURAS

1 Introduction

1.1 The problem of astronomical big data

This decade will see a series of astronomical mega-projects coming into operation, which will produce complex data whose dimensionality and volume will exceed any current scale. This requires the application of a new generation of machine learning models more robust and faster than those currently applied. These projects include the Rubin Observatory[Ivezic et al., 2011] and Space-based observatories as Wide Field Infrared Survey Telescope (WFIRST)[Spergel et al., 2015] and James Webb Space Telescope (JWST)[Gardner et al., 2006]. The Rubin observatory will generate about 36 terabytes of information per night, which implies that data will be available that will have a sequential relationship in images from extended and point sources. This will require the application of models that also incorporate sequential relationships in their architecture. The speed of classification is relevant too, as the Rubin observatory data stream will deliver a 15-GB image every 15 seconds, even more if we consider that one of the scientific objectives of the Rubin observatory is to generate alerts for transients in less than 1 minute. It is expected that the Rubin observatory will generate 10,000,000 alerts every night, thus prompt classification is a characteristic that must be present in the new models that are proposed.

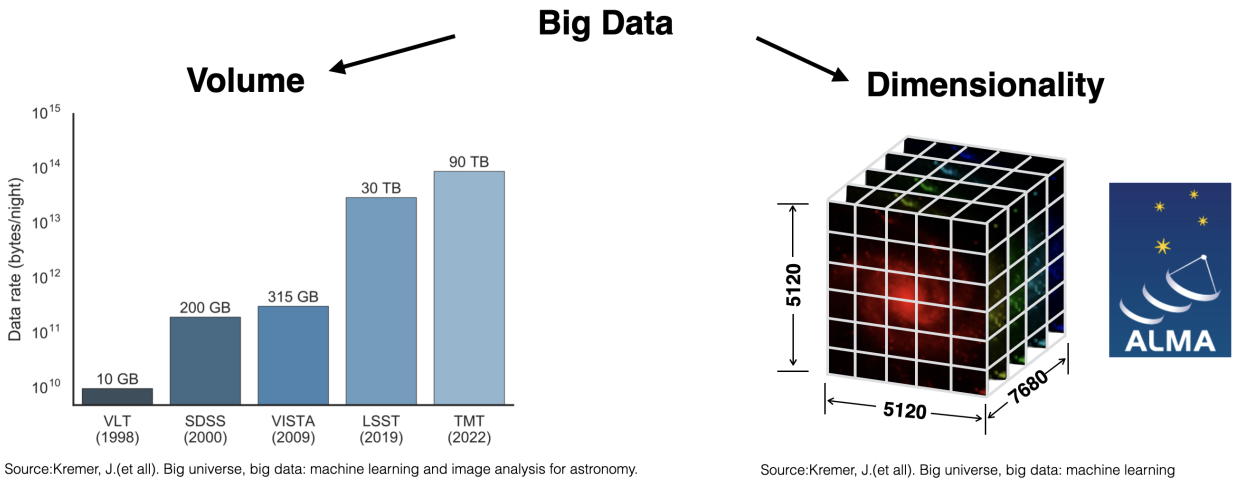


Figure 1.1: On many occasions, the bigdata problem in astronomy is reduced to the volume of data generated by mega-observatories. Due to the dimensionality of some types of observations, such as those of ALMA, there are also complexities associated with their processing.

There is already evidence to indicate that modern astronomy is a paradigm shift not only for this science in itself, also for other areas of knowledge. The challenges at the engineering level in the new scientific megaprojects such as the Rubin Observatory, SKA or E-ELT have motivated the need for new techniques and models, an open challenge even for computer science. It is projected that from 2020 astronomy will begin to generate data in a PB-Scale (≈60 Petabyte) [Berriman and Groom, 2011]. This need has given a boost to Astroinformatics, which as a first definition corresponds to the disciplinary crossing between astronomy and computing. This definition opens up a wide space for development, contributions in areas such as telescope operation software [Broekema et al., 2020], the application of machine learning [Barchi et al., 2020], optimization of observation plans [Macktoobian et al., 2020] and appearance directly related to the reduction pipeline of data [Jenness and Economou, 2015] to name some applications. Astroinformatics is now more a disciplinary crossing and can be considered as a field in itself. The present proposal, which is part of astroinformatics, aims to incorporate new Machine Learning and Computer Vision(CV) techniques to solve astronomy problems in a BigData context.

Integral field spectrographs (IFS) is one of the most powerful observation techniques to obtain spatially resolved spectroscopy of extended objects. IFS combining imaging and spectroscopy provides a powerful technique to provide a sharper and deeper 3D understanding of galaxies. The final product of IFS is a data-cube, with two spatial dimension (Ra/Dec) and a one spectral dimension (wavelength/velocity). The high density of information in a datacube, especially in cubes created using the techniques of Fibers+lenslets and Image slicer allow spatially resolved kinematics and emission lines in distant galaxies. Working with cubes of astronomical data is complex. On the one hand, we have the problem of data size that has been extensively studied in recent times Araya et al. [2016] Law et al. [2016] Hassan et al. [2013] Hassan et al. [2011]; but there is another equally relevant problem that has not had the same scientific attention,

as the dimensionality of these cubes. This problem in computer science is known as the curse of dimensionality, a term coined by Bellman Bellman [1961]. It essentially indicates that the number of samples needed to estimate an arbitrary function with a given level of precision grows exponentially with the number of dimensions. To contextualize the problem in astronomy let consider the cubes of ALMA. Figure 1.1 shows some cubes whose dimensions are 5120 on the physical axis. Due to ALMA's capabilities, data cubes can have up to 7680 channels of frequencies, corresponding to the velocity axis, resulting in almost 80 million elements in a single data cube. Deep Learning Akeret et al. [2017] Ma et al. [2017] and advanced models Kremer et al. [2017] Polsterer et al. [2015] of search and classification in astronomy, is a growing field, given the outstanding results in many fields of knowledge. The use of these spectra's cubes as input for these models imply a superlative computational complexity, so one of the objectives of the present work is to prepare these cubes for the use of machine learning.

Throughout the development of this work, the different elements that will support the hypothesis will be constructed iteratively. A detailed description of the problem and the relevance of the contributions made in this work will be presented in chapter 1. As well as a review of the theoretical framework that fits within the scope of the proposal. Chapter 2 will provide a detailed description of the astronomical data, emphasizing how they should be utilized as inputs for a deep learning model. Our next chapter will examine how it was necessary to work on the representation of astronomical observations as multidimensional data structures. It will be discussed in chapter 4 how the non-accelerated version of the pipeline was developed and how the mathematical framework, specifically the tensor algebra, plays an important role in the hypothesis. A final element that must be discussed before addressing the proposal of this work is the current state of the art in terms of existing methods to accelerate deep learning models, which will be addressed in chapter 5. Furthermore, the proposal is also innovative in this field. In the last chapter of this thesis, I synthesize all of the elements that have been presented in the previous chapters and present the main result that has come out of this thesis and its contribution to the field.

1.2 Morphology classification of extended sources

Galaxy morphology is arguably the major indicator of the physical processes that drive the evolution of galaxies. Hence, morphological classification of galaxies is essential for the understanding of the formation and evolution of galaxies in the universe. The foundations of morphological classification were proposed by [Hubble, 1926] as a purely descriptive system. Traditionally, the classification has been performed either visually by experts or through the automated extraction of features, this is, the measurement of proxies such as concentration index, surface brightness profile, color, etc., which correlate with different morphological characteristics [Conselice, 2003]. However, both methods have pros and cons. For instance, experts can reach high accuracy in recognizing structures and shapes but can only inspect a very limited amount of data. On the other hand, features can be obtained massively by computers, but results are not always satisfactory. Nevertheless, the advent of the CCD and increase of computational power gave rise to the use of computational techniques for the analysis of astronomical data. For over two decades, the identification and segmentation of individual and extended sources in astronomical images has relied on semi-automated software, mainly SExtractor [Bertin and Arnouts, 1996]. The advantages of this approach are the capacity to classify galaxies in large volumes of data, although at the risk of using proxies of morphology that not necessarily correlate with morphological types. SExtractor is a powerful tool that is the standard in the extended sources segmentation process. But it is a software that requires expert knowledge in its use given the multiple configurations that must be performed. In works [Boucaud et al., 2019] around deep learning models applied to galaxy debblening, that is, the separation of galaxies that have Overlapping. Results have been obtained indicating that these models outperform by a factor of 4 compared to the classical SExtractor approach.

Recent advances in Machine Learning and Computer Vision(CV) techniques [Baron, 2019] have demonstrated satisfactory results for the identification and automated classification of astronomical objects in images. Narrowing the problem to the morphology classification of extended sources, such as galaxies, the approaches that have shown the most outstanding results are those based on the latest advances in computational vision techniques. Deeplearning is presenting promising results in astronomy. Especially Object recognition methods that apply computer vision models to identify objects in digital images. These techniques are grouped into four categories: Image Classification, Object Localization, Object Detection and Instance segmentation. This categorization is based on the results that each method can achieve. In figure 2.2 we can visualize the result of each of these categories. The first category addresses one of the central problems in computer vision, it focuses on predicting the class of an object in an image. In Image Classification are the firsts works [Odewahn et al., 1992], [Dieleman et al., 2015], [Banerji et al., 2010] [Sandler et al., 1991] applied in astronomy. The Object Localization methods allow you to indicate if a certain object is in an image in addition to indicating its location with a bounding box.

In the case of the last and powerful two categories Object Detection and Instance segmentation represent the state of the art of Object recognition methods. The first category in addition to indicating the class of an object as

Image Classification, also indicating their location with a bounding box. In this category we find two main families of architectures: Region Proposals (R-CNN Girshick et al. [2014] , Fast R-CNN [Girshick, 2015] , Faster R-CNN [Ren et al., 2015]) and You Only Look Once (YOLO) [Redmon et al., 2016] [Redmon and Farhadi, 2017] . In AstroCV[González et al., 2018] an architecture based on YOLO was used was applied for the morphological classification of galaxies. Emphasizing speed when performing this classification.

The most powerful category is Instance segmentation. The methods in this category combine the characteristics of Object Detection with those of semantic segmentation. This is an intermediate category that classifies each pixel of image into the given classes. The objective of instance segmentation is to segment or separate each "instance" of a class in an image. These methods allow reach identifying object at the pixel level. The last difference with Object Detection is that the bounding box used for highlighting the object is replaced. This bounding box is changed to highlighting the specific pixels belonging to the identified object using for example a mask. It is important to point out, for the objectives of the morphological classification of galaxy, that this characteristic is relevant. Since it allows to extract each object separately without any background, this is not possible for example in methods based on semantic segmentation. There are two especially outstanding architectures in this category: U-Net [Ronneberger et al., 2015] and Mask R-CNN [He et al., 2017] . U-net is a fully convolutional network architecture. In other words, the network does not have a fully-connected layer. It owes its name to the way its layers are organized, this form is because the network consists of a contracting path and an expansive path. An architecture based on U-Net was used for mitigating radio frequency interference (RFI) signals in radio data [Akeret et al., 2017]. The results in terms of their precision – recall curve are up to the standard method used by radio astronomers.

Mask R-CNN extends Faster R-CNN by adding the functionality of to pixel-level image segmentation. Is based on two stages: first, it scans the image to generate proposals; which are areas with a high likelihood to contain an object. Is based on two stages: first, it scans the image to generate proposals. This process seeks to find areas with a high likelihood to contain an object. Later it classifies these areas (proposals). Finally, bounding boxes and masks are created on the original image. Mask R-CNN is the state of the art in Instance segmentation. A specific reason for opting for Mask R-CNN instead of a U-NET-based architecture is whether or not the scientific goals of the problem needs the polygon mask that Mask R-CNN generates. Put another way is if it is necessary to highlight each feature for positional information.

1.3 Segmentation morphology of galaxies

Semantic object segmentation is a common task in astronomy, for example using sextractor [Bertin and Arnouts, 1996]. The separation of sources or debblending for example galaxies versus stars is a process that requires reaching the pixel level. In other words, indicate whether or not a particular pixel belongs to a particular source. This is complementary to the classification of sources; In the case of galaxies, the morphology of these must also be incorporated into the model. Therefore, a machine learning model is necessary to achieve both objectives in a single process autonomously. In addition to considering the necessary constraints associated with the complexities of the volume and dimensionality of the data currently available and on the horizon of the Rubin Observatory.

Given the current volumes of information, it is necessary to seek the implementation of a unique pipeline of classification, location and segmentation of galaxies according to their morphology. In a previous proposal called Mask Galaxy: Morphological segmentation of Galaxies. A pipeline based on a Mask R-CNN network that allows you to incorporate pixel-level segmentation of an image already classified and also located by the model was implemented. In figure 1.2 we can visualize the architecture of the Mask R-CNN model.

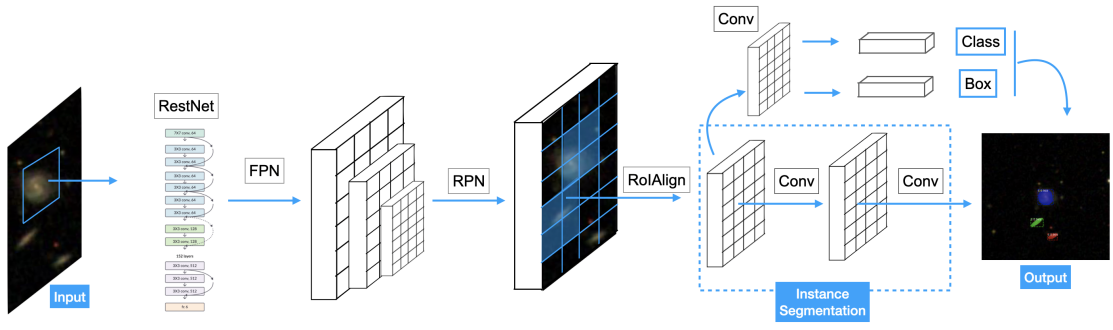


Figure 1.2: Mask R-CNN model architecture with two stages: object-based approach and image segmentation.

It is the state of the art of CNN Regions-based networks (R-CNN). The objective of the R-CNN is to solve the problem of object detection. For this purpose in general terms the process is divided into: Generate the Region of Interest (ROI) Proposals and classify these ROIs. The calculation process to find the ROI has a high computational complexity. In order to deal with this complexity and solve the problem, a series of architectures based on R-CNN have been presented. These architectures are Fast R-CNN, Faster R-CNN and finally Mask R-CNN. The main difference between R-CNN, Fast R-CNN and Faster R-CNN is in the substantial increase in the speed of: training and inference. This increase in the case of Faster R-CNN is due to the incorporation of a Region Proposal Network (RPN) after the last convolutional layer. This implies that the R-CNN ROI proposal method is replaced by a deep network. This will ensure that ROIs are inferred from the network generated feature map. This network are called backbone. The backbone is a convolutional neural network responsible for generating the space of features of the model. In this work, we used a Residual Deep Neural Network (Resnet) He et al. [2016] with 101 layers with MS-COCOLin et al. [2014] dataset pre-trained weights as the backbone. Taking advantage of this characteristic of the model, a training strategy based on transfer Learning and Differential learning rates was applied.

One of the premises of the proposal was the implementation of a machine learning pipeline for morphological classification of galaxies. The above under the context that classifying whether or not a galaxy has a certain morphology does not address the entire complexity of the problem. To classify a galaxy it must first be located, this must be the first stage of the pipeline. The problem of localization becomes more complex in scenarios such as the 3.2 gigapixels of the Rubin Observatory or other mega scientific projects such as the E-ELT. The location as indicated can be addressed with computer vision techniques such as object detection and localization. The proposed model includes the generation of a bounding box around the galaxies. Automatically locating these on the plane.

The results achieved in accuracy and other metrics, the state of the art is achieved. Results in terms of classification and object detection applied to the morphological classification of galaxies. Finally to achieve an automatic pipeline, another task must be included: Segmentation. The proposal includes automatic segmentation of galaxies tailored to their individual morphological. The result is presented visually as a mask and the model returns this segmentation as a set of coordinates that can be automatically incorporated into a catalog. These results position the proposal as the first to accomplish these three tasks (classification, location and segmentation).

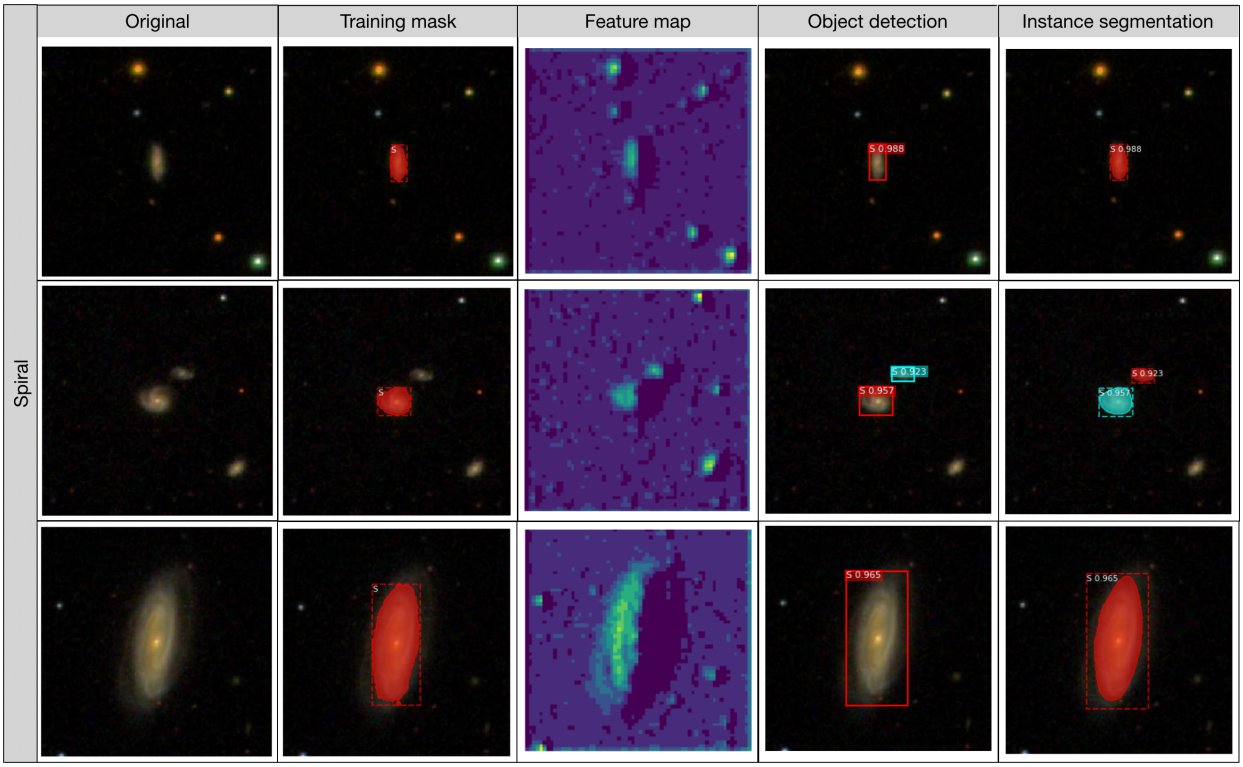


Figure 1.3: Automatic classification, location and segmentation pipeline applied to spiral-type galaxies

In figure1.3 it is possible to verify that the model is an automatic pipeline for the classification and morphological segmentation of galaxies. The first image corresponds to a galaxy to be identified. The second shows the input of the

model for training (tag and mask). The third corresponds to the feature map generated by the backbone(ResNet) of the network. The third corresponds to the result of classification and location of the model. Finally the model incorporates the segmentation of the galaxy.

1.4 Thesis proposal

With Tensor Mask R-CNN, a more compact and faster morphological classification (and segmentation) model of galaxies is pointed out. Based on using not only RGB images as input of the network without real astronomical observations. These multi-band observations will be first processed using tensor methods seeking to extract the most significant characteristics. On the basis that this data is naturally multidimensional and therefore the relevant information is distributed throughout this feature space. In addition to the incorporation of tensor methods in the CNN layers of the Mask R-CNN network, it will be possible to reduce the number of parameters to be learned in the convolutional layers of the backbone. Formalizing a model that respects and perseveres the multidimensional structures present in the multi-band observations of the current surveys and aims to support the survey of the petascale era in astronomy.

Hypothesis :
Incorporating multidimensional algebra factorization methods into an instance segmentation model derived from Mask R-CNN, with astronomical photometric observations as input, will enable the development of an automated pipeline to categorize, locate, and segment galaxies efficiently and accurately. These tasks will be supported on a scale comparable to current astronomical data.
Objectives:

- Implement a multi-band astronomical data loading interface. In the form of a tensor of order 3 as input to the model.
- Implement a dimensionality reduction model based on tensor methods techniques. They preserve the multidimensional structures present in multi-band observations.
- Design and implement a model based on Mask R-CNN that has the ability to work with multiband astronomical data.
- Train the T-Mask backbone. Replacing the 2D convolutional layers of the model with tensor layers to speed up the model.
- Implement computational support on Pytorch (GPU) to process these tensors.

To be able to support the proposal, there are three axes that need to be validated:

- The application of tensor methods to astronomical data, specifically multi-band observational data, in order to test the validity of a dimensionality reduction technique that does not impact the multilinear manifold present in such observations. A detailed discussion of this topic can be found in chapter 3.
- The use of Mask R-CNN as the basis for an automatic pipeline of galaxies morphological segmentation is validated to be functional to the scientific objectives of these observations. It is important to validate the use of an instant segmentation architecture on astronomical observations, given the particular characteristics of these observations, before applying the innovations proposed in this thesis. In chapter 4, we will focus on the design of the pipeline architecture, the creation of the dataset, and mainly the scientific conclusion that validates this model as a basis for the aforementioned proposal.

2 Astronomical Data

2.1 Astronomical scientific products

For the purpose of contextualization there are four main categories of scientific products generated from different observation techniques:

1. Light curves.
2. Images.
3. Data cubes.
4. Survey.

Images: Within the scientific products generated from astronomical observations the images have had a important place. The first images were generated in an analogous way, but with the incorporation of chip charge-coupled device (CCD) they made a qualitative leap in quality and precision. The images are generated mainly from photometry, using instruments such as: NASA/ESA Hubble Space Telescope. Its importance is evident as they will be one of the main scientific products of the Rubin Observatory. Also the images are the source of other types of scientific products such as light curves and spectral information.

Data Cube: These cubes are the scientific result of the spectroscopy observations and radio observations. Aims to obtain greater precision and depth. However, this sacrifices field observation, unlike photometry that aims to cover more field, but less depth. The data cubes are formed by two physical coordinates, Right ascension(Ra)/Declination(Dec), and a third coordinate that in the case of ALMA, is wavelength or velocity. In addition to ALMA, there are other instruments such as The Multi Unit Spectroscopic Explorer (MUSE) in the Very Large Telescope (VLT) that also generates an Integral Field Unit (IFU) cube in the visible wavelength range.

Survey: Among the various types of data generated in astronomy, large area surveys are especially useful for the application and testing of such machine learning models, partly because the observations generated by a survey exhibit a certain level of homogeneity in their characteristics and in the technical configurations of the instruments from which they were generated. Some good examples of this are the Sloan Digital Sky Survey (SDSS)[Adelman-McCarthy et al., 2008] and the Gaia mission [Brown et al., 2016]. The SDSS each night it produces approximately 200 gb of data. Its catalog reaches around 170 TB in the Data Release 15. The SDSS is a Multi-Band Survey that uses five photometric broad-band filters u, g, r, i, and z. Using the SDSS, close to 200 million galaxies have been observed. Their brightness has been captured using each of these five bands. The new surveys will move us to the petascale astronomy. Therefore, in addition to the scientific characteristics of the problem, the computational complexity of working with a high dimensional input must be considered. So the models must incorporate the speed of processing as a key factor in their formulation. A solution to this problem is to reduce this dimensionality and weight, but retained the multidimensional structures and relationships generated from multi-band observations.

2.2 Astronomical data in the computer science scope (data structures)

Section 2.1 different that modern astronomical generates scientific products were presented. These observations can be modeled according to their dimensionality in different data structures. Figure 2.1 shows that light curves are one-dimensional data that can be represented as 1D arrays. The images from the perceptive of computer science are 2D arrays. The case of images also includes combined images, these are representations of multidimensional observations (eg. photometric observation) converted to RGB. This is the dominant input in deeplening models applied to astronomical problems. The most relevant case for this proposal are the cube data, represented as 3D arrays. This category includes scientific products in varied sections of the radio spectrum. There are the radio astronomy results as ALMA. Photometric Observations can also be considered as low resolution data cubes. For example, the five filters of the SDSS of a certain observation. Integral field spectrographs (IFS) stands out among the visible spectrum. IFS is one of the most powerful observation techniques to obtain spatially resolved spectroscopy of extended objects. IFS combining imaging and spectroscopy provides a powerful technique to provide a sharper and deeper 3D understanding of galaxies. The high density of information in a IFS datacube, especially in cubes created using the techniques of Fibers+lenslets and Image slicer allow spatially resolved kinematics and emission lines in distant galaxies.

The entry for the proposal is a photometric Observation, represented as low resolution data cube. This data cube has two physical axes (Right ascension(Ra)/Declination(Dec)), and a third axis (Wavelet, Time, Intensity). Tensor from the computational perspective is a multidimensional array, in our implementation is modeled as data structure in Pytorch. This is due to its native support for multidimensional data structures, also known as tensors. Pytoch takes special care

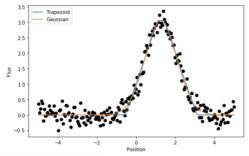
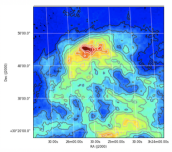
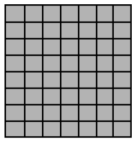
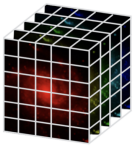
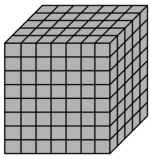
	Astronomy	Computer Science	Mathematical
Light curve		 Array 1D	**Vector** (1-Way Tensor)
Image		 Array 2D	**Matrix** (2-Way Tensor) **Linear Methods**
Data Cube		 Array 3D	**Tensor** (3-Way Tensor) **Tensor Methods**

Figure 2.1: Representation of astronomical observations from a mathematical perspective and their representation as data structures.

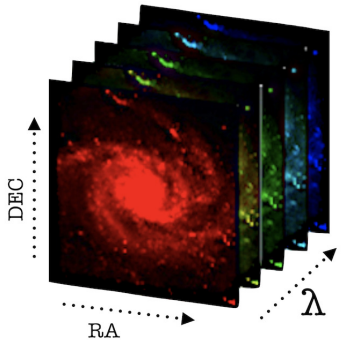


Figure 2.2: The representation of what would be an astronomical data cube. As in the case of SDSS with five filters.

in its implementation to ensure that these representations are fast, as it includes low-level methods that store the tensor information in a neighboring manner.

This classification is based on the models applicable to these scientific products, on the data structures on which they can be represented, which is the mathematical framework that allows them to operate on them and finally which the computational support necessary to deal with their complexity and volume. The present proposal raises the need to have the computational and mathematical support to address the considered astronomical data: Data quantity and Data Complexity.

Data Complexity - The astronomical data are inherently multidimensional. Either when considering the metadata associated with its observation or in observations from spectroscopy by way of exemplification. The exploitation of this multidimensionality through robust models that allow to manipulate and find latent information in spaces of greater parameters an open problem. As in the case of data cubes and the relationships that we can find in the velocity axis, they are an open problem.

Data quantity - For this proposal, in addition to the size of the disk data, it is proposed that the astronomical data can be divided based on the amount of information in the final scientific products of the reduction pipelines of each observatory or instrument. Astronomical data can have a high level of information or dense data, such as data cubes. On the other hand, there are light curves with a low level of information or sparce data. The midpoint are the images that we will denote as semi-sparse data.

3 Tensor representation, constrain and processing of multidimensional astronomical

A version of this chapter has been published in the journal Astronomy and Computing. In general, the content is the same, since it is part of the hypothesis validation process. Several chapters were validated in pairs, thus validating the use of tensor methods to represent and reduce the dimensionality of astronomical data. The original article DOI is https://doi.org/10.1016/j.ascom.2018.10.007

The astronomical data are mostly multidimensional, this is naturally understood in the results of spectroscopic observations, but also the case of photometry where we can make a stack of for example different filters of a given observation. Considering the foregoing, we can indicate that many of the problems in astronomy can be represented as matrices or bi-dimensional arrays, either from the perspective of linear algebra or from the computer sciences as well as bi-dimensional data structures. For example, we can consider a cube of ALMA data as a tensor, or a stack of matrices, or its zero moment as a two-dimensional array. Understanding the multidimensional nature of astronomical data, associated with its superlative increase in size and considering the growing interest in models Rabanser et al. [2017] such as machine learning, which archive a certain degree of automation in the search for the latent multilinear manifold in these data is that it is necessary to explore dimensionality reduction techniques that allow to create more compact representations of these data as input for these new models.

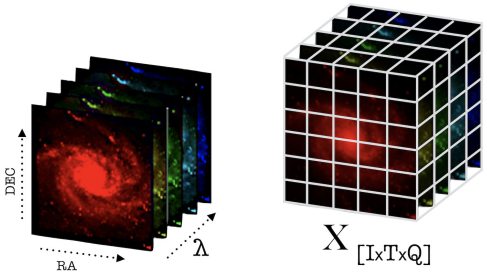


Figure 3.1: In the form of a tensor of order 3, the structure of a multiband observation is represented.

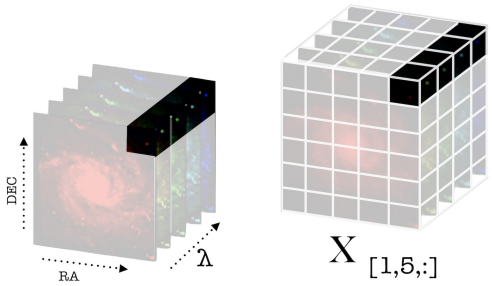


Figure 3.2: Spaxel represented as Mode-3 (tube) fibers, $\boldsymbol{X}_{1,5,:}$

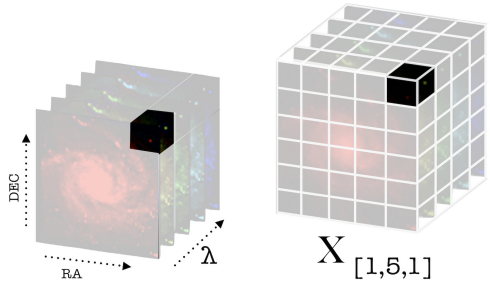


Figure 3.3: Voxel represented as sub-tensor, $\boldsymbol{X}_{1,5,1}$ cu

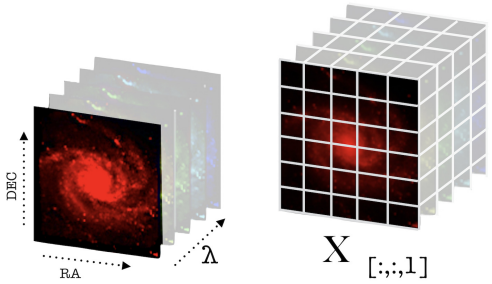


Figure 3.4: Tensor matrix (band) selection, $\boldsymbol{X}_{:,:,1}$

3.1 Dimensionality reduction

Before we delve into dimensionality reduction techniques, and assuming that we can represent astronomical data in a matrix form, our problem is to find the theoretical support needed to extract correlations and structures of lower dimensionality from these two-dimensional data, that is to say we are faced with a problem of matrix factorization. For this purpose there are widely studied linear modelsGoyal et al. [2013] and used successfully in a wide number of fields of knowledge including astronomyBrunt and Heyer [2013]. We will see an instruction from them as a basis for the approach of our multilinear proposal.

3.2 Principal component analysis (PCA)

PCA is a powerful linear and non-parametric technique for dimensionality reduction, matrix factorization and also it has been used to find the most representative elements of two-dimensional array. Basically, its operation consists of finding the eigenvectors that define a certain set of data (represented by matrix). PCA is a linear operator applied to multidimensional data that transports this data to spaces of lower dimensionality, seeking to maximize the variance in this new space.

Its definition is the following:

$$\min_{U,V} \left\| X - UV^T \right\|_F^2 \tag{1}$$

$$\text{subject to } U^T U = V^T V = I \tag{2}$$

Where U is , V is , X is ,F is , and I is .

Although PCA is a powerful technique, its results are exposed to a complex problem called the rotational problem, which we can see in (5), where in general terms between $\hat{X} = UV^t$ you can insert infinite possible matrices that will give us infinite solutions. $\hat{X}$ is defined as .

3.3 The rotational problem

By converting a spectral observation to a matrix form, in the path to apply PCA over this representation we will have to deal with the rotational problem of matrix factorization Rabanser et al. [2017]. That in broad terms implied, that we will don't have theoretical guarantees about the uniqueness of the matrix decomposition that we are applying. A formal definition of the problem is:

$$\hat{X} = UV^t \tag{3}$$

$$\hat{X} = UV^t = UF^{-1}FV^t \tag{4}$$

$$\hat{X} = UV^t = UF^{-1}FV^t = U^{'}V^{'t} \tag{5}$$

To avoid or rather to decrease the rotational problem of PCA we can apply certain constraints to the model based on the physical nature of the astronomical data. We already know the data present in a spectral cube are nonnegative, applying this constrains we can use Non-negative matrix factorization (NMF) doi [2009]. The definition of NMF is as follows:

$$\min_{U,V} \left\| X - UV^T \right\|_F^2 \tag{6}$$

$$\text{subject to } U \geq 0, V \geq 0 \tag{7}$$

NMF has been widely studied and presents outstanding results in 2D data, in the case of astronomy, Ivezić et al. [2014]. Finally, another linear method applied to the astronomical data is PCA Tomography Murray [2015]. This method is an extension of PCA, designed to exploit the fact that in the radio astronomical data the number of sources is lower than the cases visible to the data, where PCA has been used. PCA Tomography has presented outstanding results for multi-source data, but is also designed to work over a matrix representation of the data. That is, it is to convert multidimensional data into a flat version of these represented in a matrix form.

As we presented in the previous section, both PCA and its derivatives are powerful methods for matrix factorization, specially NMF. Since we can use the nature of the astronomical data in our favor (non-negativity) we can add these constraints to our model. Despite this there are worksZhou et al. [2013] that directly state that matrification of a multidimensional array would destroy the inherent spatial structures of the way in which these data were captured, this is complex if take care the cost of an spectroscopy observation.

3.4 Tensor methods

As a technique for reducing dimensionality based on matrix factorization, PCA has been widely used and validated throughout multiple studies. It can also achieve these results under very strict conditions such as the orthogonality

constraints (in the case of SVD) of the data, and the uniqueness of its results is subject to the well-known problem of rotation without these restrictions. As a result of the foregoing, we can infer that it requires methods of analysis that are able to capture and preserve the multidimensional relationships of the information contained within the cubes of astronomical data. As a result, it is necessary to have a computational and mathematical framework that supports this multidimensional approach in order to be successful. For several reasons, tensor decompositions in specific Tucker can be viewed as a generalization of PCA for spaces of greater dimensionality. However, the advantage of tensor decompositions in general Tucker is that under certain conditions, the application is less restrictive than PCA, and the result is highly distinctive.

In algebraic terms a tensor is a generalization of the matrix, for example a matrix is a tensor of degree 2 (2-way tensor), as well as the vectors are tensors of degree 1 (1-way tensor). The most important case for our work are the tensors with a higher dimensionality (nth order, with $n3$), they are called high order tensor or n-way tensor. Having already contextualized why we have opted in our proposal to use tensor decompositions, we will make an overview of the main theoretical elements that will be used in the implementation of the solution, but for a more accurate definition see Kolda and Bader [2009] and doi [2009]

3.4.1 Kronecker Product

Given two matrices, $A \in \Re^{m \times n}$, and $B \in \Re^{p \times q}$, the Kronecker product of A and B is denoted as $A \otimes B$, matrix with shape $mp \times nq$.

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & \cdots & a_{mn}B \end{bmatrix} \tag{8}$$

Algorithm 1: Kronecker product

Data: Matrices A and B
Result: Kronecker product matrix
$C_{m \times n}(\Re) =$ zeros matrix;
for *j=1 to mp step 1* **do**
  for *i=1 to nq step 1* **do**
    $C_{j,i} = A_{\lfloor \frac{j}{p} \rfloor, \lfloor \frac{i}{q} \rfloor} \cdot B_{j \ mod \ p \ , \ i \ mod \ q}$
  end
end

3.4.2 Unfolding

Given a tensor $X \in \Re^{I_1 \times I_2 \times \cdots \times I_N}$, the unfolding process of *mode-n* of X, is denoted by X_n, a matrix in $\Re^{I_n \times I_M}$, where:

$$M = \prod_{k=1 \wedge k \neq n}^{N} I_k \quad , \tag{9}$$

and describe by the mapping from element $(i_1, i_2, \cdots, i_N)$ to (i_n, j), with:

$$j = \sum_{K=1 \wedge k \neq n}^{N} i_k \times \prod_{m=k+1}^{N} I_m. \tag{10}$$

3.4.3 Tensor-Matrix Product

Given a matrix $A \in \Re^{F \times I_n}$, denoted the Tensor-Matrix Product for the *mode-n* as $X \times_n A$, and return a tensor with dimension $\Re^{I_1 \times \cdots \times I_{n-1} \times F \times I_{n+1} \times \cdots \times I_N}$, define as:

$$(X \times_n A)(i_1, \cdots, i_{n-1}, f, i_{n+1}, \cdots, i_N) = \sum_{i_{n=1}}^{I_N} X(i_1, \cdots, i_N) A(f, i_n). \tag{11}$$

3.5 Tensor decomposition

Since we have specified our representation of a spectral cube as a tensor of order 3, in addition to establishing the operational theoretical framework to work on this tensor, it is time to present Tensor Decomposition (TD) that we can begin by defining as generalizations of the SVD for data structures of higher order, but this analogy will cease to be such when we incorporate restrictions to our proposal as the non-negativeness of astronomical data. Within the state of the art we can find several TD models, but historically stand out two: Canonical polyadic decomposition (CPD)Harshman [1970] and Tucker decompositionTucker [1964]. Both share the characteristic of being decomposed bases in outer product but each of them has advantages and disadvantages, especially associated with astronomical data, which we will detail.

Canonical polyadic decomposition (CPD): It is also known as Tensor Rank Decomposition, since in essence it can be defined as the search for a compact representation of a tensor in the form of the sum of rank-1 tensors. CPD unlike PCA does not establish an orthogonality constrains to ensure its uniqueness, but it comes from the fact that CPD decompose a tensor into a sum of rank-1 tensors.

Let $X \in \Re^{I \times J \times K}$ be a third-order tensor. The goal is to compute a CP decomposition with *R* competents that best approximates X,i.e., to find

$$\min_{\hat{X}} \left\| X - \hat{X} \right\| \text{ with } X = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\lambda; \mathbf{A}, \mathbf{B}, \mathbf{C}] \tag{12}$$

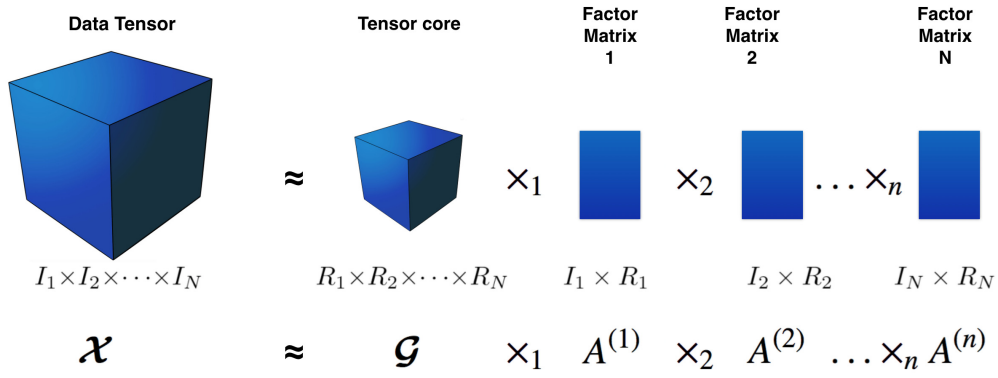


Figure 3.5: Illustration of a Tucker decomposition for a 3-way tensor X.

Tucker decomposition: Specifically Tucker3 can be presented as a generalization of PCA for high-order tensor. Tucker as we can see in Figure 3.5, decomposes the full format tensor X into a tensor core G and a series of matrices $A^{(n)}$ called factors, where n is equal to the order or dimensionality of the tensor X.

$$X = G \times_1 A^{(1)} \times_2 A^{(2)} \times_3 A^{(3)} \ldots \times_n A^{(n)} \tag{13}$$

The three-way case where $X \in \Re^{I \times J \times K}$, we have:

$$\min_{\hat{X}} \left\| X - \hat{X} \right\| \text{ with } X = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} g_{pqr} \mathbf{a}_p \circ \mathbf{b}_q \circ \mathbf{c}_r = G \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C} = [G; \mathbf{A}, \mathbf{B}, \mathbf{C}]. \tag{14}$$

For the implementation of Tucker3 there are two dominant algorithms in the literature: Higher-Order Singular Value Decomposition (HOSVD)De Lathauwer et al. [2000b] and Higher Order Orthogonal Iteration (HOOI)De Lathauwer et al. [2000a] . The first one can be considered as an SVD in higher-order tensors. A more detailed definition is:

Higher-Order Singular Value Decomposition (HOSVD): As the name implies, the relationship of HOSVD with its bi-dimensional counterpart (SVD) is that the first is in general terms the application of the second. Its fundamental idea is to iteratively find the components with the greatest variation in each of the modes in a unique way per mode.

Higher Order Orthogonal Iteration (HOOI): It is an algorithm based fundamentally on ALS and aims to find the n-ranks iteratively, that is, the HOSVD result on the tensor and use this result to start the factor matrices Rabanser et al. [2017] .

Algorithm 2: Tucker Decomposition with HOOI

Data: X, ϵ
$\left(G, \left\{U^{(u)}\right\}\right) = ST - HOSVD(X, \epsilon)$
repeat
  for $n = 1, \cdots, N$ **do**
    $Y \leftarrow X \times \left\{\mathbf{U}^{(m)T}\right\}_{m \neq n}$
    $\mathbf{S}^{(n)} \leftarrow \mathbf{Y}_{(n)}\mathbf{Y}_{(n)}^{T}$
    $\mathbf{U}^{(n)} \leftarrow$ leadding $R_n$eigenvectors of $\mathbf{S}^{(n)}$
  end
  $G \leftarrow Y \times_N \mathbf{U}^{(n)T}$
until *the quantity* $\left(\|X\|^2 - \|G\|^2\right)$ *ceases to decrease*;
return $\left(G, \left\{U^{(n)}\right\}\right)$

3.6 Tensor representation, constrain (storage) and processing of multidimensional astronomical data over intense computing support

There is no question that the big data problem in astronomy is a well known one, but in most cases it is limited by the amount of data being collected. Additionally, we address another aspect of the problem: the dimensionality problem in the context of multidimensional data, particularly Astronomical data cubes. Using tensor decompositions, we achieve two goals: first, we achieve super compression rates by using Nonnegative Tucker Factorization (NTD) that allows us to save up to 91% of disk space and network traffic. The loss of information is parameterized, and second, we use a CANDECOMP/PARAFAC or Canonical Decomposition (CP) to determine a latent multilinear manifold for these astronomical cubes. Since this is a Big Data problem, we applied three implementation methods for our library TensorFit: an intense GPU approach supported by PyTorch and a traditional HPC approach using MPI. It is natural to question why we limit ourselves to using two-dimensional models, such as PCA, to reduce the dimensionality of spectral cubes when we are dealing with multidimensional data (astronomical cubes). Consequently, multidimensional approaches that preserve the multidimensional relationships in these data are at a disadvantage. For this reason, it has been proposed to switch from linear algebra to multilinear algebra using tensor theory.

Despite the fact that linear methods have demonstrated good results for certain problems, there remains an open problem in capturing the richness of multidimensional astronomical data as well as their characteristics (non-negativity) and guaranteeing that the solutions found are unique. Here is where the value of our proposal comes into play, as we are moving from linear decomposition to multilinear decomposition, which will use the theory of tensors as a framework in order to reduce dimensionality as well as find the latent multilinear manifold present in these astronomical observations.

As the basis of our proposal, we have the following premises to justify the application of multilinear models:

- Spectroscopic observations contain relevant information along all axes of the cube.
- The data of a spectral (astronomical) cube are positive Ivezić et al. [2014] .
- A tensor decomposition under certain conditions is unique. In particular as they are tensors of order 3 Zhou et al. [2013] .

The entry for our proposal is a cube of astronomical data or spectral cube. This cube has two physical axes (ra, dec) and a third axis (walevelett, time, intency), so that a representation in a tensor of order 3 is almost natural. In our case, we extract this data from a FITS, read the content with the Spectral-Cube library, finally converting it into a multidimensional array on Pytorch (GPU and HPC).

Tensorfit has four main goals, convert cubes of data (or greater dimensionality) into tensors from the computational and algebraic perspective. Also have the algebraic support (Tensor decomposition, Kronecker, Tensor dot matrix, unfolding, etc.) to operate on these tensors and provide high-performance computing on GPUs and CPUs to process these tensors with the methods of the algebraic support defined above. Finally coupled to the current processing pipelines, the input is a cube that can be processed by the current library, which can be over-compressed or processed with tensor techniques and return a cube that can continue to be processed by current libraries.

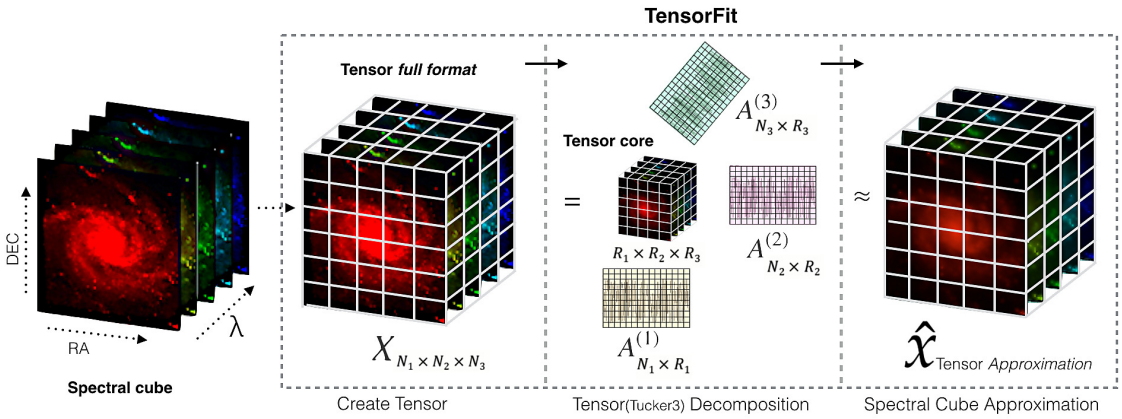


Figure 3.6: Pipeline of Tensor Decomposition (Tucker) in TensorFit

Our implementation is based on the Spectra-Cube Robitaille et al. [2013] library to read these cubes and then our functions convert these cubes into multidimensional arrays in Pytorch,because it offers torch, a general purpose array library similar to Numpy but with native python support for GPU, already at this point we have a representation of the astronomical data in a format that could be used even as input a Deep Neural Network also in Pytorch. With the representation constructed we used the Tensorly Kossaifi et al. [2016] library to operate on the tensor, using the Pytorch backend, unfortunately the complete GPU support is still under development which is evident in the results that will be seen in the conclusion section.

In our proposal, one of the most important statement is to use tensor decomposition as a technique for dimensionality reduction in specific Nonnegative tensor factorization (NTF) Kim and Choi [2007] , can achieve rates of super-compression while preserving the multi-linear latent manifold into cubes spectral data. To validate our proposal we use our TensorFit library that allows you to apply decomposition tensor to different types of spectral cubes, parameterizing the HOOI algorithm according to the levels of information loss that we are willing to accept.

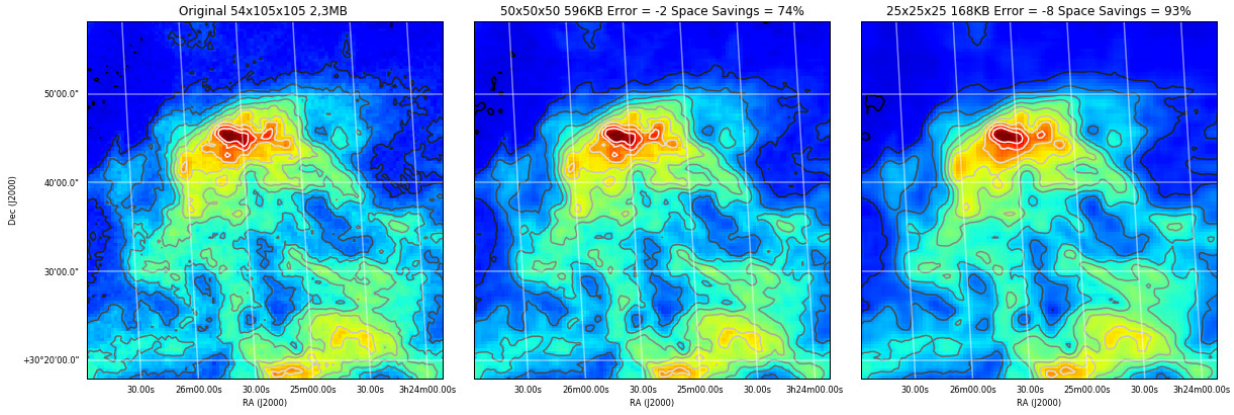


Figure 3.7: Contour curves over the moment 0 of three cubes, the first the original two seconds compressed with Tucker.

For our first experience we used an observation of L1448, a star-forming region in Perseus, which has been widely used in the validation of astronomical software such as astropy Kim and Choi [2007] . This cube of data has a weight of 2.3 MB and its dimensions are 105x105x54 (Ra, Dec, Velocity) to which we apply tensor decomposition with different parameters, finally on the approximate and original cubes we apply the moment zero for comparison reasons. As shown in Figure 3 to validate that the geometric structure has not been lost in the compression process, we apply contour curves that clearly show that the most representative structures remain even in cases where we reach a compression of 25x25x25 and a final weight of 168 KB which implies 93% savings in disk space, but maintaining 85% accuracy.

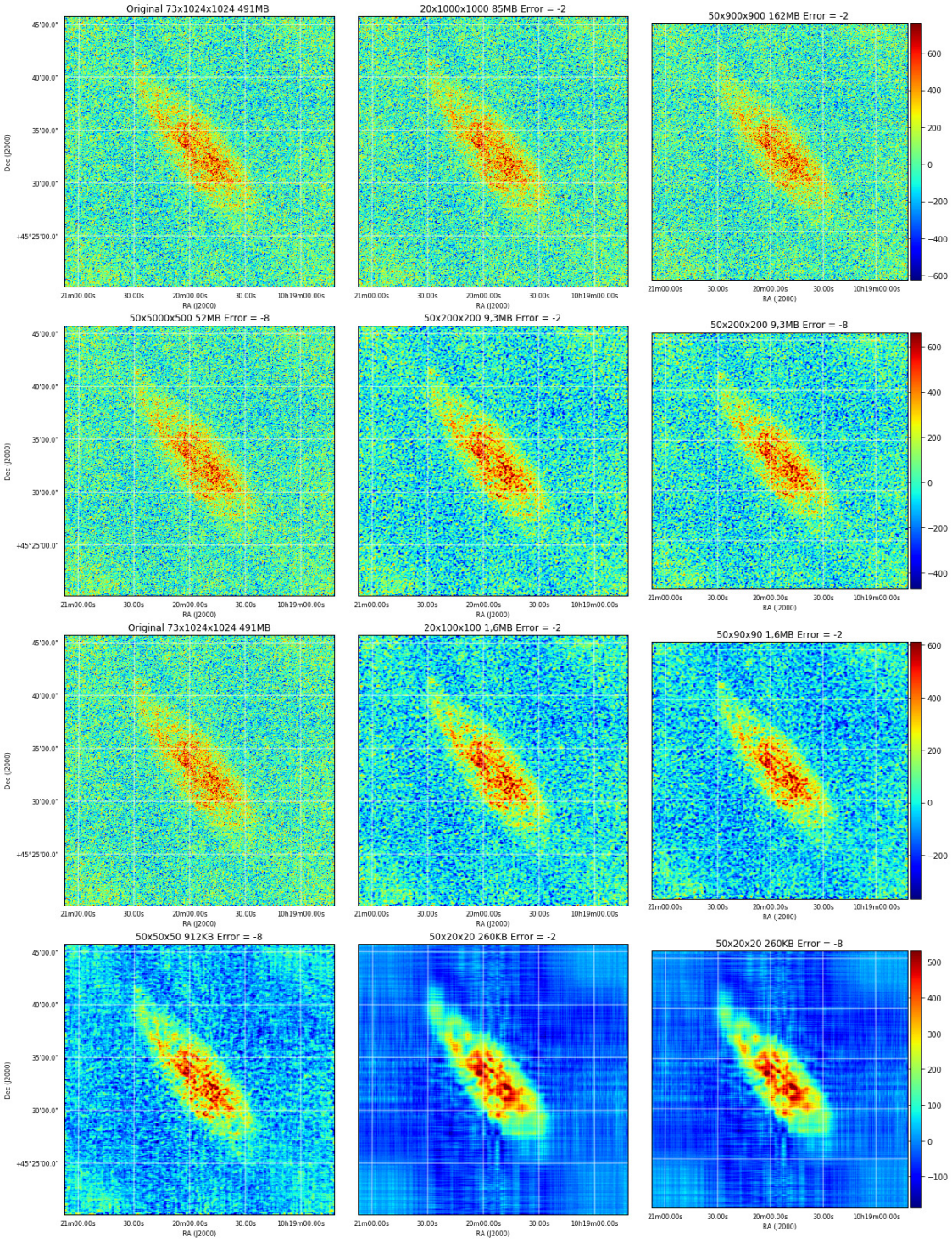


Figure 3.8: Results of moment 0 for twelve spectral cubes, the first two rows are cubes with Low level of decomposition and the last two rows are cubes with High level of decomposition. The first graph of each group corresponds to the moment 0 of the original cube.

Once we have validated our premise about the conservation of the structures present in the data cubes, we will concentrate on the levels of compression that we can achieve. In Figure 3.7 we can see two interesting results according to the level of decomposition:

Table 3.1: Experimental results with different configurations of Tucker.

Shape	Error	RMSE	Time	Size [MB]	Space Savings[%]
20x1000x1000	-2	0.0002991346118506044	2min 23s	85	71
20x1000x1000	-8	0.0002991346118506044	3min 48s	85	71
50x900x900	-2	0.0001753489632392302	2min 44s	162	44
50x900x900	-8	0.0001753489632392302	3min 32s	162	44
50x500x500	-4	0.0001939747307915240	1min 53s	52	82
50x500x500	-8	0.0001939747307915240	1min 57s	52	82
50x500x500	-2	0.0001939747307915240	1min 52s	52	97
25x250x250	-8	0.0003312000480946153	1min 8s	8	97
25x250x250	-2	0.0003311944019515067	3min 58s	8	97
50x200x200	-8	0.0003150611591991037	1min 7s	9.3	96

Low level of decomposition - In the first two rows we parameterize tucker with values of reduction of high dimensionality, that is to say where the size of the core tensor is close to that of the original tensor.In this case we can see that much information is preserved in reconstruction, and view in Figure 3.8 time 0 of reconstruction so we can corroborate with accuracy levels in Table 1. The above achieved space savings rates ranging from 44% to 97%, but always maintaining the multi-linear manifold as we can see.

High level of decomposition - In this case we bet on a size of the tensor core significantly lower than the original tensor. As we can see in the last two rows of Figure 3.8 see the level of information loss is greater and we also see that the amount of space saved also. In addition, although initially we could also indicate that these results are similar to a denoising process.

Our initial motivation was to recognize the multidimensionality of the data cube, and to seek computational and theoretical support that would enable us to develop models that respect the physical characteristics of these observations, as well as take into account all relationships that occur across all the dimensions (axes) of the data cube. Towards accomplishing the above, we proposed the use of tensor theory as a theoretical basis, and as a test case, we assert that applying a dimensionality reduction technique based on tensor decomposition would reduce the dimensionality and also reveal the latent multi-linear manifold present in the astronomical data. Through experimental validation, we have demonstrated our implementation can work with the dimensionality and volume of the astronomical cubes in base tensor decomposition in order to achieve extreme compression rates in addition to maintaining the multilinear manifold of the astronomical data cubes.

The purpose of this chapter was to validate the ability to model astronomical data as tensors at the data structure level. Have mathematical support for them, particularly in terms of reducing their dimensionality or generating a compact version of them. To provide support for part of the hypothesis, this validation will be used in the following chapters to apply these to models that use these astronomical data. This opens up several research lines surrounding the use of compressed data cubes in machine learning models, as their structures are maintained, but their dimensionality is reduced. In addition to the above, there are also models of tensor regression that are used, among other things, for the characterization of bright features, the morphological classification of galaxies, or the improvement of the resolution of some observations. Furthermore, canonical polyadic decomposition can be applied to detect spectral lines or transients, for example. While our library for the GPU version has enabled us to work with cubes with representative volumes, we do not have the ability to handle, for example, SKA cubes, so we will need to continue iterating in new versions that do not require Numpy to read Spectralcube cubes and transform them into n-arrays in Pytorch.

4 Analyzing astronomical observations using deep learning

A version of this chapter has been published in the journal Astronomy and Computing. In general, the content is the same, since it is part of the hypothesis validation process. It was the objective of this study to validate the use of an instant segmentation model as part of an automatic pipeline. A pipeline was developed to morphologically segment galaxies, which was successfully achieved and validated by peers. The original article DOI is https://doi.org/10.1016/j.ascom.2020.100420

The astronomical data are generally multidimensional. This is naturally understood in the results of spectroscopic observations, but also in the case of photometry. On this latter point, it can make, for example, a stack of different filters of a given observation. Considering the foregoing, it can be indicated that many of the problems in astronomy can be represented as matrices or two-dimensional (2D) arrays, either from the perspective of linear algebra or from the computer sciences, as well as 2D data structures. For example, a data cube of ALMA could be seen as a tensor, or a stack of matrices, or its zero moment as a 2D array. Understanding the multidimensional nature of astronomical data, associated with their superlative increase in size, and considering the growing interest in models [Rabanser et al., 2017] such as machine learning, which achieves a certain degree of automation in the search for the latent multi-linear manifold, in these data is necessary to explore dimensionality reduction techniques that could allow to create more compact representations of these data, as an input for these new models.

As indicated in section 1.3, the architecture of the networks used so far have based their operation on three-channel images (RGB). Not taking into account the richness of the multidimensional information present in the multi-band observations. This responds to a domain transfer of models that have shown outstanding results in other scientific fields that work with this data format. Working with multiband photometric astronomical observations as the input to a deep learning model involves two options:

Recent advances in Machine Learning and Computer Vision techniques have demonstrated satisfactory results for the identification and automated classification of astronomical objects in images [for a summary, see Baron, 2019]. Among the various types of data generated in astronomy, large area surveys are especially useful for the application and testing of such machine learning models, partly because the observations generated by a survey exhibit a certain level of homogeneity in their characteristics and in the technical configurations of the instruments from which they were generated. Some good examples of this are the Sloan Digital Sky Survey [SDSS hereafter; **?**] and the Gaia mission [Brown et al., 2016].

Applications of Deep Learning are delivering promising results in astronomy, especially in object recognition methods that apply computer vision models to identify objects in digital images. These techniques are grouped into four categories: Image Classification, Object Localization, Object Detection and Instance Segmentation. This categorization is based on the results that each method can achieve. In Figure 4.1, we illustrate the result of each of these categories. The first category addresses one of the central problems in computer vision: predicting the class or type of an object in an image. The first applications of computer vision in astronomy corresponds to Image Classification [e.g., Sandler et al., 1991; Odewahn et al., 1992; Dieleman et al., 2015; Banerji et al., 2010]. The methods in the second category (Object Localization) answer the question if a certain object is in an image, indicating its location with a bounding box additionally.

The other two categories, Object Detection and Instance segmentation, represent the state of the art of Object in image recognition methods. Object Detection methods also indicate the class of an object as Image Classification methods do, showing its location with a bounding box as well. Within Object Detection we find two main families of architectures: Region Proposals [R-CNN; Girshick et al., 2014] , Fast R-CNN [Girshick, 2015] , Faster R-CNN [Ren et al., 2015]) and You Only Look Once [YOLO; Redmon et al., 2016; Redmon and Farhadi, 2017]. For instance, AstroCV is an implementation based on YOLO architecture for morphological classification of galaxies which emphasizes speed when performing this task.

The most powerful category is the Instance Segmentation. The methods in this category combine the characteristics of Object Detection with those of Semantic Segmentation. This is an intermediate category that determines what pixels in the image belong to a given class. The objective of the instance segmentation is to segment or separate each "instance" of a class in an image at pixel level. A key difference between Object Detection and Instance Segmentation is that the bounding box used for defining an object in the former is complemented with a segmentation mask in the latter. This last characteristic is a relevant feature for the morphological classification of galaxies, because it makes possible to extract each object separately without any background. This is not possible, for example, in semantic segmentation-based methods.

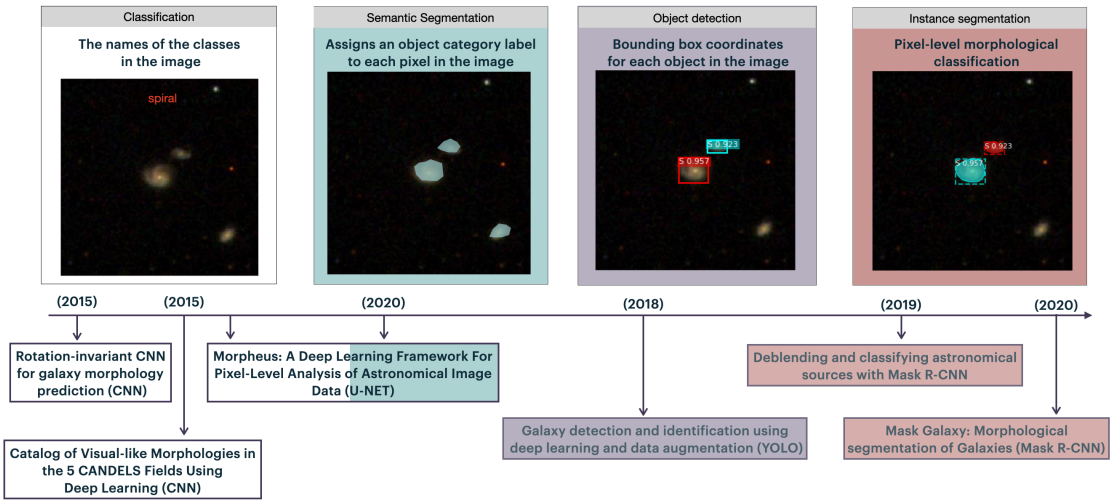


Figure 4.1: Examples of the output of four different Computer Vision models applied to an astronomical image of galaxies.

There are two especially outstanding architectures in Instance Segmentation: U-Net [Ronneberger et al., 2015] and Mask R-CNN [He et al., 2017]. U-net is fully-convolutional network architecture. In other words, the network does not have a fully-connected layer. It owes its name to the way its layers are organized. This form is the result of network consistency, that is, a contracting path and an expansive path. For example, an architecture based on U-Net was used to mitigate radio frequency interference (RFI) signals in radio data [Akeret et al., 2017]. The results, in terms of its precision —recall curve, are up to the standard method used by radio astronomers.

Mask R-CNN extends Faster R-CNN by adding the functionality to pixel-level image segmentation. It is based on two stages: Firstly, it scans the image to generate proposals. This process seeks to find areas with a high likelihood of containing an object; then it classifies these areas (proposals). Secondly, bounding boxes and masks are overlaid on the original image. Mask R-CNN is the state of the art in Instance Segmentation. A major reason to choose Mask R-CNN instead of U-NET-based architecture is whether or not the scientific goals of the problem need the polygonal mask that Mask R-CNN generates. To put it another way, whether or not it is necessary to identify each pixel (and its position) belonging to a feature. Arguably, the standard tool used for segmentation of extended sources is SExtrator. Although, this software requires expert knowledge for its use, given the multiple parameters that must be set for successful results. Moreover, recent development in deep learning models are providing better results than SExtractor. For instance, Boucaud et al. [2019] applied a model to solve galaxy deblending, that is, the separation of overlapping galaxies. Their results indicate that the proposed model outperforms SExtractor by a factor of 4.

4.1 Mask R-CNN

The identification of sources, for example of galaxies and stars, is a process that requires to reach the pixel level. This is a complementary process to the classification of sources. In the case galaxy classification, their morphology must also be incorporated into the model. Therefore, a machine learning model needs to achieve both objectives in a single process, autonomously. In addition, it should consider the necessary constraints associated with the complexities of the volume and dimensionality of the data currently available.

The Mask R-CNN is the state of the art of the Region-based Convolutional Neural Networks (R-CNN). The goal of the R-CNN family architecture is to solve the problem of object detection. To achieve this purpose, broadly speaking, the process is divided into: Generate the Region of Interest (RoI); proposals; and, classify these RoIs. In particular, finding the RoIs presents high computational complexity. In order to deal with this complexity and solve the problem, a series of architectures based on R-CNN have been presented.

These architectures are Fast R-CNN, Faster R-CNN and finally Mask R-CNN. The main difference among them is the substantial increase in the speed of training and inference in the newer networks. Moreover, Mask R-CNN is the evolution of Faster R-CNN that adds the segmentation capability to the model. Thus, in addition classification and localization, the network adds instance segmentation as a third output to the model, as shown in Figure 4.2. The first part of the Mask R-CNN architecture is a pre-trained Convolutional Neural Network on image classification tasks. This

network is called the Backbone and it is responsible of generating the space of features (hereinafter feature map) of the model. We use a Residual Deep Neural Network (Resnet) [He et al., 2016] with 101 layers and the MS-COCO dataset [Lin et al., 2014] pre-trained weights as the backbone. The feature map acts as an input for the following stages of the model.

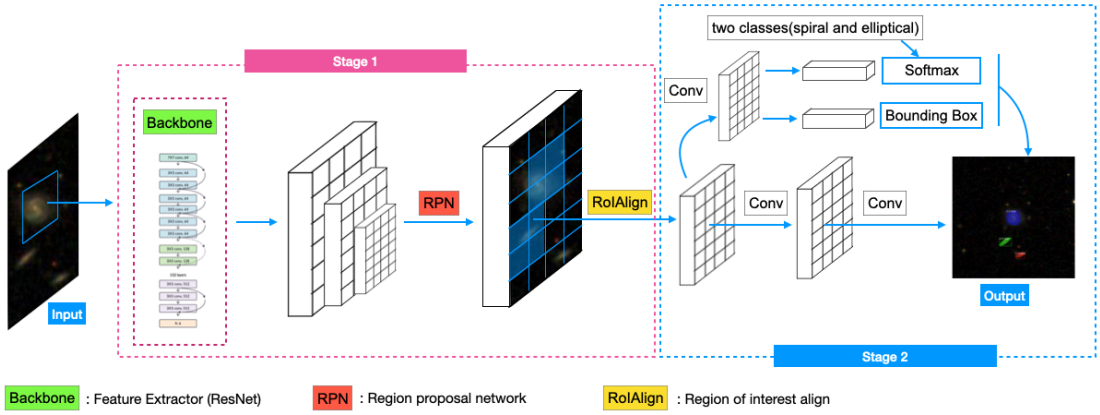


Figure 4.2: Mask R-CNN model architecture with two stages: First, it scans the image to generate proposals.Later it classifies these areas (proposals). Finally, bounding boxes and masks are created on the original image. Diagram based on He et al. [2017].

The next stage in the model is the Region Proposal Network (RPN). A RPN is a lightweight fully-convolutional network that operates as a binary classifier. The process consists of running a small $n \times n$ spatial window on the feature map of each image to generate a set of potential regions called *proposals*. These proposed regions or proposals have different sizes because they depend on the size of the spatial sliding window used to generate them. The size of such a sliding window defines an anchor box. A network has many anchor boxes that have predefined locations and scales relative to images. The size of the anchor boxes is closely related to the nature of the problem. In particular, we utilize anchors of sizes {4,8,16,32,128} pixels. We present the results of applying different anchor sizes to an image in the left panel of Figure 4.3. The three galaxies are clearly identified in this panel.

Any model that aims to classify galaxies should consider their different sizes. For instance, a Milky Way-like galaxy is one hundred thousand light years in diameter approximately, whereas dwarf galaxies are a few percent of that size. On the other hand, galaxies such as M100 are approximately 52.5 million light years diameter. Furthermore, galaxies will exhibit a particular angular size according to their distances. Following this rationale, the size of the anchors that we use considers the size of the input images and the variety of sizes of the galaxies that could be found in such images. In particular, we choose small anchors considering the small size of the objects (galaxies) in our sample, the pixel scale of the data plus the small size(256×256 pixels) of the training images as well.

The RPN generated anchors at three different scales and three different aspect ratios. Each of these anchors has an objectiveness base score of two classes: object/no-object. Finally, non-maximum suppression (NMS) is applied to anchors in order to remove boxes that have not overlapped. In practice, a value is calculated using an Intersection over Union (IoU) metric and boxes that do not reach a predefined threshold are removed. IoU measures the overlap between 2 boundaries, as can be seen in Figure 4.4. IoU is the relationship between the box that represents the correct position of an object (ground truth), and another box that represents the prediction of the model. IoU is defined as a fraction represented in Equation (15) and represented as boxes B1 and B2 in Figure 4.4. A threshold of 0.8 IoU was used in the model to filter RPN proposals.

$$IoU = \frac{B1 \cap B2}{B1 \cup B2} \tag{15}$$

Next, the RPN proposals that pass the binary classification process (object/no-object) go to the next stage with the name of ROI. The left panel of Figure 4.3 shows the RPNs contributing to the training (the solid line) and those that do not contribute (the dotted lines). After applying Non-max Suppression, there are fewer anchors than the number of candidates. These filtered anchors are renamed as Regions of Interest (RoIs). These RoIs go through two processes called RoI Classifier and Bounding Box Regressor. The result of these processes generate two outputs for each RoI: The Class of the object and the Bounding Box Refinement. The Classifier, unlike the RPN, incorporates the real class

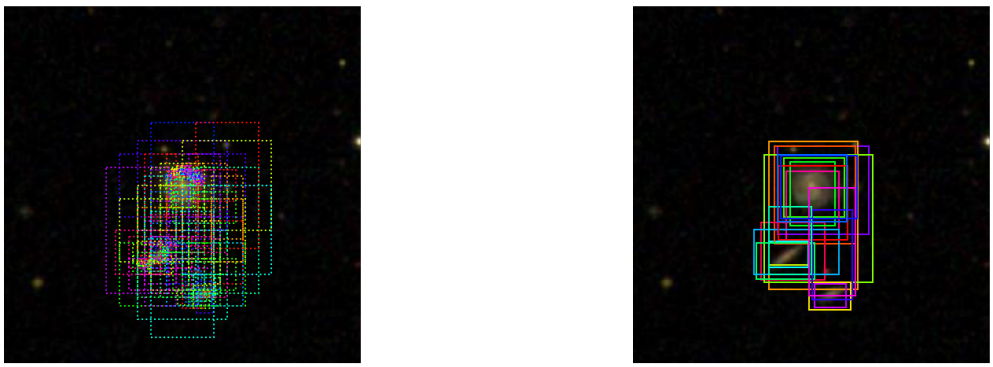


Figure 4.3: RPN refinement process: This stage of the model transforms the boxes proposed by the RPN into a RoI. Left: Anchors before refinement. Right: Refined anchors after non-max suppression result in RoIs.

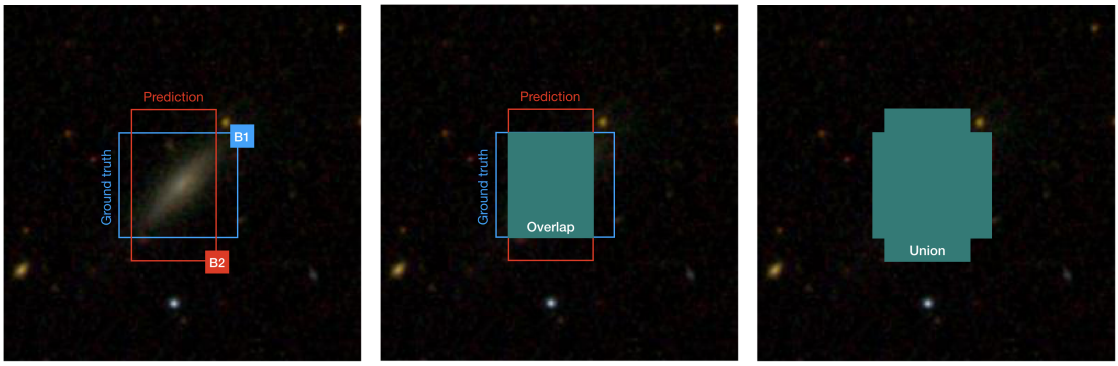


Figure 4.4: Intersection over Union (IoU) visual representation.

that the object belongs to; in this case, whether it is an elliptical or a spiral galaxy. In the case of the Bounding Box Regressor, it is necessary to make a size adjustment because of the different sizes that the boxes generated by RPN have. Also, the pixel-level segmentation stage requires much more fine-grained alignment with the bounding box. Therefore, it is necessary to obtain fixed-size boxes first. For this purpose, the model incorporates the stage called The RoIAlign layer. The RoIAlign mechanism is based on cropping a part of an original feature map selected in a bounding box and resizing it to a fixed-size. Table 4.1 lists the two techniques used for this size change: square and crop.

Finally, as indicated before, a unique element of the Mask R-CNN architecture for the objective of this work is the ability to segment the model. The mask is generated by a fully convolutional network receiving as input the RoI selected by the RoI Classifier. In other words, all the regions that contain objects of the classes that exceed the minimum parameterization accuracy and with bounding boxes of the defined size. An important aspect to consider for this model is the particularity that astronomical sources do not exhibit sharp edges. In this sense, the masks achieve the pixel level and is represented as floating numbers, achieving thus more precision to whether or not a pixel belongs to the classified object. This is a relevant aspect for the automatic segmentation of astronomical sources generated by the model. Figure 4.5 shows the step-by-step procedure followed by the entire model being applied to an SDSS image. The final result is verified for the classification, location and segmentation of the three galaxies present in the image. All this process occurs within a single automatic pipeline based on machine learning.

The Mask R-CNN architecture, as already explained, enables the classification, location, and segmentation implemented as single automated process. Therefore, the error function of this model includes these three tasks, as defined in Equation (16). This loss function is applied to each sampled RoI, where $\mathcal{L}_{\text{cls}}$ is the log loss function over two classes (Elliptical and Spiral). The $\mathcal{L}_{\text{box}}$ term captures the error generated by the bounding box regression.

$$\mathcal{L} = \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{box}} + \mathcal{L}_{\text{mask}} \tag{16}$$

These errors expressed in the ($\mathcal{L}_{\text{cls}}$,$\mathcal{L}_{\text{box}}$) terms are the ones that evaluate the classification and location tasks. Regarding the $\mathcal{L}_{\text{mask}}$ term, it provides a gauge of the segmentation error. Equation (17) defines the expression to calculate this error, which is the average binary cross-entropy loss. This error is only calculated in RoIs where the class of the identified object corresponds to the ground truth class of that object.

$$\mathcal{L}_{\text{mask}} = -\frac{1}{m^2} \sum_{1 \leq i,j \leq m} \left[y_{ij} \log \hat{y}_{ij}^{k} + (1 - y_{ij}) \log(1 - \hat{y}_{ij}^{k}) \right] \tag{17}$$

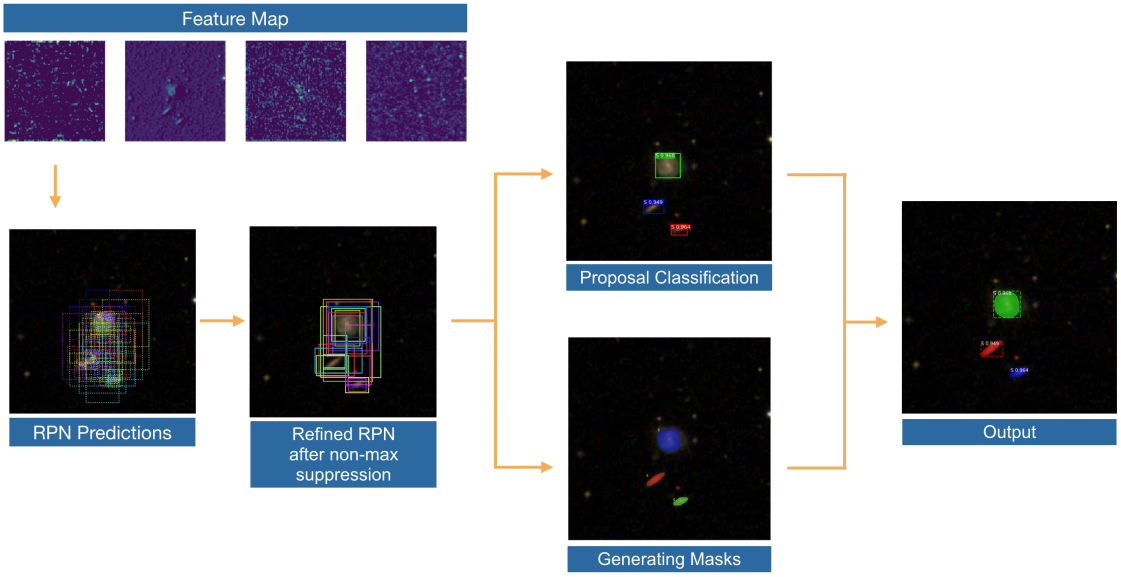


Figure 4.5: The two-stage Mask R-CNN model is summarized in four steps.

4.2 Instance segmentation applied to Segmentation morphology of galaxies

The backbone is a key element of the Mask R-CNN architecture, because it makes possible to reuse an already highly specialized architecture in the detection of features in images. In this work, we used a Residual Deep Neural Network (Resnet) to utilize the transfer learning technique in the first stage, from MS-COCO Lin et al. [2014], and in a second iteration, as will be seen in this chapter, from a network trained by the authors in the recognition of galaxies (not including morphology). Taking advantage of this characteristic of the model, it was applied a training strategy based on transfer Learning and Differential learning rates.

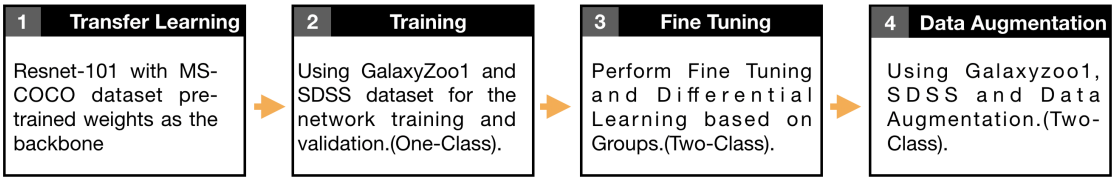


Figure 4.6: Training strategy involves four steps.

4.3 Dataset labels: The Galaxy Zoo datasets and morphological classes

We utilize the catalog of morphological classifications from the Galaxy Zoo 1 (GZ1, hereafter), and the Galaxy Zoo 2 (hereafter GZ2) datasets for the network training and validation. These datasets have been largely used in the astronomical community because they fulfill the requisites of volume and labeled data required for training [e.g., Barchi et al., 2020]. Both GZ1 and GZ2 were Citizen Science projects, where approximately 10^5 and 8×10^4 volunteers performed morphological classification of around 9×10^5 and 3×10^5 galaxies, for GZ1 and GZ2, respectively, using

Table 4.1: Learning configurations of the training process.

Configuration	Two class network
Number of classes(*)	3
Pool size	7
Backbone	resnet101
Backbone strides	{4, 8, 16, 32, 64}
Activation	Relu
Detection max instances	400
Image resize mode	crop
Image shape	256x256
Learning momentum	0.9
Learning rate	1e-03
Max gt instances	200
Mini mask	true
ROI positive ratio	0.33
RPN anchor scales	[8, 16, 32, 64, 128]
RPN nms threshold	0.8
Train ROIs per image	128
Steps per epoch	1000
Use RPN ROIs	true
(*) Number of classes including background	

RGB images presented through a web portal[1]. The astronomical images came from the SDSS, which carried out a photometric campaign of a large portion of the northern sky in five photometric bands. The color images presented in GZ1 and GZ2 were composed using the (g, r, i) photometric bands, mapped as the JPEG (R,G,B) channels, respectively, following the prescription described in Lupton et al. [2004].

The GZ1 morphological classification scheme includes Spiral (clockwise, counter-clockwise and edge-on), Elliptical, Merger, and Star or Don't Know (e.g., artifact) labels. A second more complex classification scheme is presented in the GZ2 dataset. Because the latter scheme is a more detailed classification, we intersect the GZ1 and GZ2 catalogs of galaxies with spectroscopic redshifts, using the *dr7objid* label. The merged catalog contains 239,639 objects. For training and validation, we utilize only the labels from the GZ1 catalog. However, we assess the network performance using galaxies assigned to the same class (Spiral or Elliptical) in both catalogs.

It is important to note that the Galaxy Zoo samples only include the nearest, brightest, and largest systems. As a result, we will not train the network to detect star-like objects. Both Galaxy Zoo projects applied several cuts to ensure that the sub sample of SDSS only contains fine morphological features that can be resolved and classified by volunteers. Specifically, the sample of galaxies was pre-filtered, using the following parameters from the SDSS Data Release 7: a Petrosian half-light magnitude brighter than 17.0 in the r-band after Galactic extinction correction was applied; the Petrosian radius lies between (petroR90) > 3[arcsec], and redshift between $0.0005 < z < 0.25$ [Willett et al., 2013].

To minimize uncertainties in the classification, we generate an intermediate catalog that only includes objects classified as either Spiral (S) or Elliptical (E), and whose debiased fraction of votes is greater than 80%. This results in a catalog with $33,809$ and $81,406$ objects, classes S and E, respectively. The remaining $124,424$ objects are labeled as Uncertain (U), following the GZ1 classification. The final data set was created by randomly selecting $12,000$ objects from this intermediate data set. This dataset is made up of $6,000$ spiral-type galaxies and $6,000$ elliptical-type galaxies, for the purpose of having a balanced dataset. Galaxies labeled as Uncertain (U) were not included in the dataset because they do not constitute a morphological class, but they correspond to galaxies that did not obtain 80% of the votes to be assigned as elliptical or spiral. The galaxies marked as U will be used in the Section 4.5 to validate the capacity of the model to classify and compared against the ability of a human classifier.

The $12,000$ objects were divided as follows: Training set (8,000 objects), Validation set (2,000 objects) and Test set (2,000 objects). These three subsets are composed evenly by the two classes (spiral and elliptical).

[1]Currently at https://www.zooniverse.org/projects/zookeeper/galaxy-zoo/

4.4 Dataset images: The Sloan Digital Sky Survey

After the creation of a balanced data set of labels from the Galaxy Zoo catalogs, we generate the respective images that correspond to each labeled object. We utilize images from the SDSS, accessed through the SkyServer [Szalay et al., 2002] *SDSS DR15 Finding Chart Tool* web service. For each labeled object, we download a JPEG image centered at the object coordinates. We set the scale to match the native pixel scale of the SDSS imager ($0.396127\ [arcsec/pix]$), and the image width and height to 256 pixels, which equals 101.4 arcseconds. The images provided by the web service are created by the SDSS through the `ImgCutout` web service [Nieto-Santisteban et al., 2004], which pre-processes the SDSS data to produce 24-bit JPEG images on demand.

4.5 Learning strategy

Table 4.2: Training configurations of the training process.

Stage	Layer Trained	Epoch	Learning rate	Train steps	Train steps
2	heads	10	1e-03	100	20
2	5+	20	1e-04	100	20
2	3+	30	1e-05	100	20
2	all	40	1e-06	100	20
3	heads	10	1e-03	1000	200
3	5+	20	1e-04	1000	200
3	3+	30	1e-05	1000	200
3	all	40	1e-06	1000	200
4	heads	15	1e-04	1000	200
4	5+	25	1e-05	1000	200
4	all	35	1e-06	1000	200
4	all	50	1e-06	1000	200

Traditionally, the training of CNNs requires a dataset of thousands of images with a label that indicates their respective class; for example, in the case of galaxies, its morphological type. In the case of Mask R-CNN, this need can be mitigated by the backbone, which is part of the Mask R-CNN architecture. The backbone is a convolutional neural network responsible for generating the space of features of the model. Consequently, the training strategy is based on transfer learning and differential learning rates and is divided into four stages as shown in Figure 6.11. The first stage (transfer learning) is focused on performing a domain adaptation of Resnet-10, moving from weights based on MS-COCO to those of the dataset created from Galaxy Zoo 1 (labels) and SDSS (images). The second stage is focused on executing a deep adjustment of the weights of the selected backbone, enabling this network to identify and segment galaxies. As a third step, a strategy of deep adjustment of the backbone weights is maintained. In addition, a Data Augmentation pipeline was incorporated into the training process. At this stage, the morphology of the galaxies was incorporated into the parameter space that the network had to learn. Finally, a new dataset of $12,000$ images and a fine-tuning learning strategy based on differential learning rates were selected from the knowledge base. A detail of the steps is described below.

Stage one – A Resnet with 101 layers was used for the backbone of the network with the MS-COCO dataset pre-trained weights. This dataset was selected intending to harness the power of generalization of this model. In this case, the dataset had $12,000$ randomly selected images, but it kept the representation of the balanced classes (elliptical and spiral). In the space of parameters that the network had to learn, it was not included the morphological information of the galaxies, the only indication was that it was a galaxy. The objective of this stage is to make a domain adaptation of Resnet-101 trained to identify 81 objects that are part of MS-COCO.

Stage two – An intense training strategy was put into effect to achieve a model of the recognition of galaxies between various astronomical sources. Table 4.2 shows the training strategy focused on using softer learning rates in the first layers and more intense rates in the intermediate layers, leaving the highest learning rates for the final layers. This strategy has been applied previously in other works, for instance Yosinski et al. [2014]. They suggest that the first layers are specialized in the detection of shapes and edges. Hence, at this level the adjustment needed is not deep, because the visible galaxies are also composed of shapes and edges. The most complex filters reside at the medium level and are specialized in the detection of parts of objects. Finally, the outermost layers of Resnet are specialized in recognizing complete objects from different shapes and sizes. This is where it is more important for the backbone weights to be adjusted so that the model can recognize galaxies, among other astronomical sources.

Stage three – This stage had a fine-tuning-based approach as a transfer learning strategy in complement to Differential Learning Rates. The learning rate was reduced by $10\times$ per layer, as seen in the table 4.2, following a strategy of Differential Learning Rates based on groups. This is because at this stage the morphology is incorporated into the parameter space that the network must learn. It is necessary to remove fully connected layers to be able to re-train the parts of the network responsible for creating the feature maps of the model.

This new network was initialized with the weights calculated in stage two. In the training, there was no adjustment to the weights of the first layers, but adjustments with a fine rate are permitted in the intermediate layers. Only in the case of the output layers the model was adjusted with greater intensity in order to achieve the morphological segmentation of galaxies according to the information incorporated into the model.

The difference with stage one is that it is not necessary to make a domain adaptation because it was already based on a model capable of recognizing galaxies.

Stage four – In stage four, the training also followed the strategy of Differential Learning Rates. In this case, there is an increase in the number of epochs with softer rates, as seen in the Table 4.2. At this stage, a data augmentation pipeline was incorporated. This is a common technique in Deep Learning used to obtain a better generalization of the model. Basically, it consists of increasing the amount of training data using images from the same dataset. This pipeline was constrained to the physical characteristics of the problem, which will be presented in detail next.

we present the relevant findings of this work. First, we evaluate the performance of the model using the classical machine learning metrics for classification tasks. Next, we evaluate the results obtained by Instance Segmentation applied to JPEG SDSS images. After, we test the application of the network to perform astrometric measurements on a random subset of Galaxy Zoo galaxies as an example of an application of the unique segmentation capabilities offered by Mask R-CNN.

Classification and location – The metrics used to assess the performance in the classification of a machine learning model are precision, recall, F1-score, accuracy and error rate. To calculate these metrics, it is necessary to count the results of the model on the validation sub-sample. Figure 4.7 shows confusion matrices, which is a standardized way to summarize the prediction results on classification. There are four possible outcomes, which are defined as:

- TN / True Negative: Case when the entry datum is negative and the model predicts negative.
- TP / True Positive: Case when the entry datum is positive and the model predics positive.
- FN / False Negative: Case when the entry datum is positive and the model predicts negative.
- FP / False Positive: Case when the entry datum is negative and the model predicts positive.

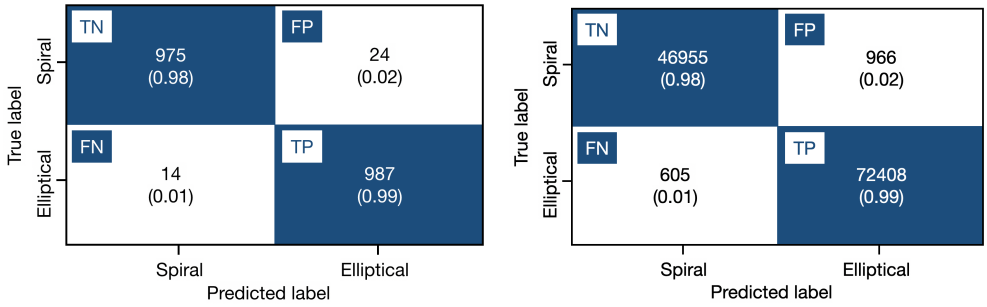


Figure 4.7: Confusion matrices of the prediction results over image classification. Left: Metrics of the model for the validation subsample and the GZ1 classification labels. This sample required a minimum score of 0.8. Right: Same as Left panel, but for the remaining $120,934$ galaxies that correspond to the galaxies in common between the GZ1 and GZ2 samples.

The classification accuracy is given by Equation(18). Following this definition, the model achieves a value of 98%. This, in addition to the fact that the classes in the training dataset were balanced, enables us to estimate the predictive capacity of the model.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{18}$$

As shown in Table 4.3, TN reaches a total of 975 cases for spiral galaxies, and 987 cases for elliptical galaxies. This describes a layered model of correct discerning, in terms of the individual entries.

Table 4.3: Statistics of the model

Metric	Spiral galaxies (1)	Elliptical galaxies (2)
TP (True positive)	975	987
TN (True negative)	986	975
FP (False positive)	14	24
FN (False negative)	24	14
ACC (Accuracy)	0.98	0.98
ERR (Error rate)	0.019	0.019
F1	0.98	0.981

However, accuracy by itself is not sufficient to assert whether the model is a precise morphological classifier of galaxies. Precision and recall metrics are also necessary for this purpose. In global terms: Precision, as defined in Equation (19), allows one to evaluate the quality of the model rating. The precision reaches 97% —this is the percentage of cases where the model is successful when in the correct galaxy class. Then we have the recall, which indicates what percentage of the positive class the model has been able to identify correctly. Recall is defined in Equation (20). Our model reaches 99%. These values were achieved by setting a classification threshold of IOU = 0.8. Thus, both metrics are usually in tension. Since a binary classifier seeks recall and precision, being both equally important, the metric F1 was calculated. This metric reaches a value of 98%. With these results, the classification power of our model is validated.

$$precision(p) = \frac{TP}{TP + FP} \tag{19}$$

$$recall(r) = \frac{TP}{TP + FN} \tag{20}$$

In the case of the evaluation of an object detection model, the element used is the Intersection over Union (IoU) metric. IoU applies the model as an evaluation metric. In addition to the classification indicators, it is necessary to verify the localization power of the model since it is difficult for the ground truth and prediction to match exactly. Having a metric that measures this relationship is important, for it is essential to set an IoU threshold to indicate whether the prediction is a true positive or a false positive. The IoU threshold is normally set at 0.5. IoU, and it does not consider the class to which the located object belongs. As the proposed model is a morphological classifier, it is necessary to incorporate this information into the evaluation. This is why the metric used is Mean Average Precision (mAP). The network achieved a value of 0.93 of mAP with all validation dataset. This result validates the power of the Architecture, based on Mask R-CNN for the automatic processes of automatic classification, location and segmentation of galaxies in a single pipeline based on deep learning.

This chapter aims to validate the use of an instant segmentation model as an alternative to sextractor and other applications that require manual configuration. Due to the volume of data available, it is not possible to rely on this supervised approach. Combined with what was analyzed in chapter 3, this provides the basis for validating two of the three premises that support the thesis' hypothesis. Although the 93% mAP achieved is a metric that validates the model at the deep learning level, it is necessary to compare the model with expert-level work. As a result, the model's inferences will be compared with the classifications made by Galaxy Zoo's experts.

Comparison of Network and Human classification (GZ1 and GZ2): Having validated the classification capabilities of the model according to the usual metrics, next we compare the classification capabilities of our model against those of human classifiers. As we have indicated before, the motivation behind the present work is to generate a single pipeline for automatic classification and segmentation of galaxies in the context of the current volume of data generated in astronomy.

Figures 4.8 and 4.9 summarize the accuracy of our model against the GZ1 scores for galaxies classified as Spiral or Elliptical, respectively. In particular, Figure 4.8 compares the accuracy values of the model against the GZ1 scores for Spirals, with the requirement that the GZ2 class also is S. For the cases where the model agrees with the GZ classifications (shown in blue color), we divided the sample between $(GZ1_{score} > 0.8)$ —which usually is referred as the *clean* sample in GZ1 (upper left panel)— and $GZ1_{score} > 0.8$ (lower left panel). The model returns a high accuracy for the clean sample, with median values over 97% and a low 1-σ dispersion. The network performs well

even for the second sample with lower GZ1 scores. In both cases, it is clear how accuracy increases monotonically as the GZ1 score increases. We interpret this as an indication of good agreement between the metric given by network accuracy and the scores of human classifiers. On the contrary, the upper right panel displays the values where the model is in disagreement with the GZ classification. However, it is important to note that there are two relevant differences compared to the previous panels. First, the median value of the accuracy, for GZ scores within the clean sample, is much lower than the accuracy for galaxies in agreement (upper panels). Second, the 1-σ values are also much broadly distributed around the mean, and lower. After a visual inspection of many misclassified galaxies, we conclude that the model actually fails in most of the cases; nevertheless, there are cases where the model is correct and GZ is not —one case is shown in the lower right panel.

Figure 4.9 presents results in a similar fashion as Figure 4.8, but for galaxies classified as Elliptical in both GZ1 and GZ2. In this case, we again see the correspondence and monotonical increase of accuracy with GZ1 score. However, the median value of accuracy is lower, for any GZ1 score, compared to the S class in Figure 4.8. Furthermore, the dispersion is larger. In the case of misclassified galaxies (upper right panel) and after visual inspection of many galaxies, we find that the network fails in most cases; however, the fraction of cases where the network is right is larger compared the equivalent case shown in Figure 4.8. An example of this is shown in the images within the lower right panel. Anyway, the accuracy reported by the network for misclassified galaxies is still lower than the accuracy for the correctly classified ones (upper left panel). This behavior was also observed in the previous figure.

Some additional insight is provided by the distributions shown in the histograms of both figures. The distributions of accuracy values for objects classified correctly are skewed and narrowly peaked at values larger than 0.90. On the contrary, the accuracy distribution for misclassified galaxies resembles a uniform distribution, approximately between 0.6 and 0.95, with no peak. For both morphological classes, the misclassified galaxies are small compared to the ones classified correctly, as it was shown in Figure 4.7. In general, we find that misclassified galaxies had a bright star near or over the galaxy.

Taking into consideration the performance of the model, shown above, we consider that the network provides clean samples for accuracy values over 0.9 for both classes, which would add around 48% more galaxies to the clean sample of the GZ1.

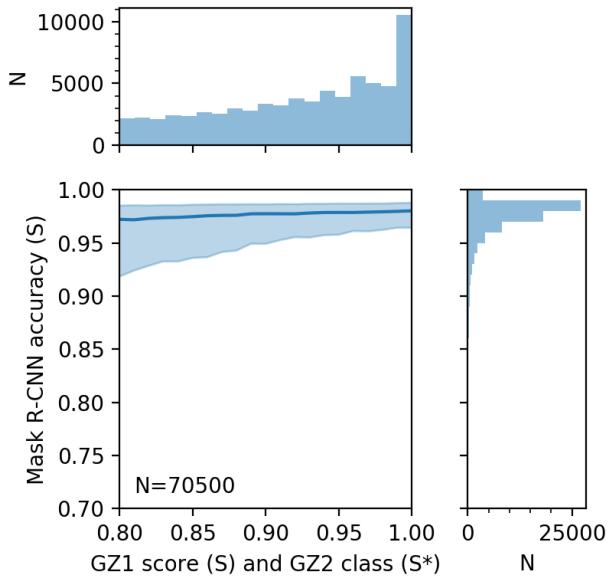
Automatic classification, location and segmentation pipeline.: One of the premises of the proposal was the implementation of a machine learning pipeline for morphological classification of galaxies. This, under the context that classifying whether or not a galaxy has a certain morphology does not address the entire complexity of the problem. Furthermore, it is the location of this galaxy within a flat as the new scientific mega-projects such as the Rubin Observatory, E-ELT, are captured.

Additionally, another problem is the location of the galaxy, in highly complex scenarios such as the 3.2 gigapixels of the Rubin Observatory or other mega scientific projects such as the E-ELT. The location, as indicated, can be addressed with computer vision techniques, such as object detection and localization. The proposed model includes the generation of a bounding box around the galaxies, automatically locating these on the plane. With the results achieved in accuracy and other metrics presented in the table 4.3, the state of the art is achieved in terms of classification and object detection applied to the morphological classification of galaxies.

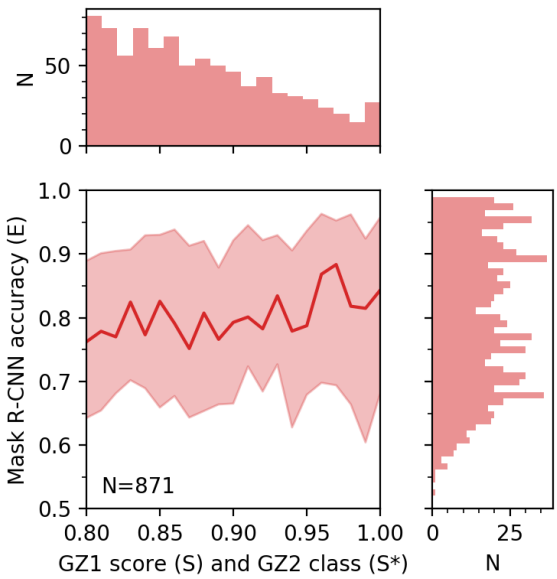
To achieve an automatic pipeline, another task must be included: Segmentation. The proposal includes automatic segmentation of galaxies tailored to their individual morphology. The result is visually presented as a mask, and the model returns this segmentation as a set of coordinates that can be automatically incorporated into a catalog. These results position the proposal as the first to achieve these three tasks (classification, location and segmentation). In Figure 4.10 it can be appreciated that the model is an automatic pipeline of classification and morphological segmentation of galaxies. The first image corresponds to a galaxy to be identified. The second shows the input of the model for training (tag and mask). The third corresponds to the feature map generated by the backbone (Resnet) of the network. The third corresponds to the result of classification and the location of the model. Finally, the model incorporates the segmentation of the galaxy.

As we mentioned in Section 6.1, no color correction nor level adjustments were applied to the images of the dataset. This is a relevant factor because it impacts in the ability to generalize the model. Figure 4.11 shows the response of the network to an example image with marginal background calibration (first panel). Furthermore, we test the performance of the model to increasing chroma noise drawn from a Gaussian distribution with scales {12.75, 25.5,51.0,76.5}, for panels two to four}. Even under a Gaussian noise with scale 76.5, the model is able to locate, segment and classify the object as a spiral galaxy with an accuracy of 0.92.

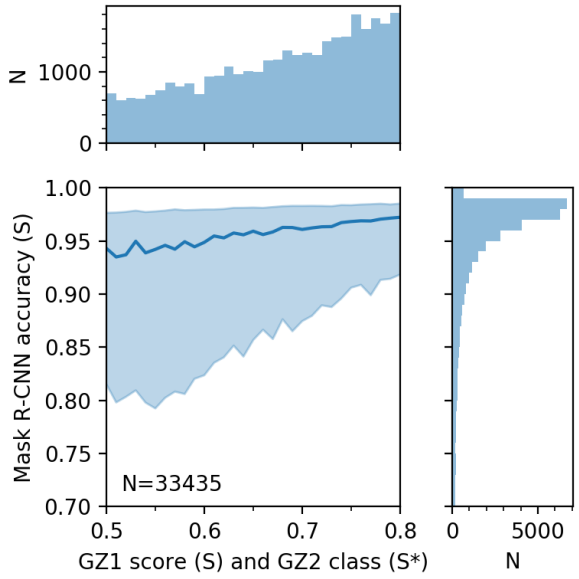
Additionally, we validate the ability to generalize the model in terms of not being limited to the size of the dataset images. We downloaded a 1000×1000 JPEG image from the SDSS SkyServer centered at $(\alpha, \delta) = (309.42896\,\mathrm{deg}, -1.1696352\,\mathrm{deg})$ with a pixel scale of $0.360\ [arcsec/pixel]$. The results obtained were 82 galaxies



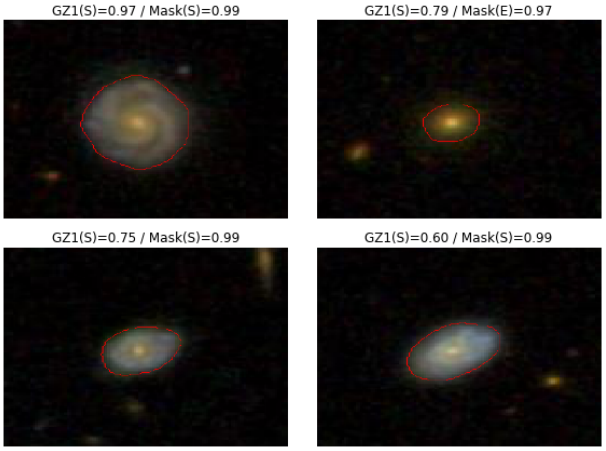
(a) Network results for galaxies classified correctly. The galaxies included comprise the *clean* GZ1 sample (i.e., $score > 0.8$) and were classified as S in GZ2.



(b) Mask R-CNN accuracy for galaxies misclassified as Ellipticals.



(c) Same as panel 4.8a, but for a GZ1 sample within $(0.5 < score < 0.8)$.

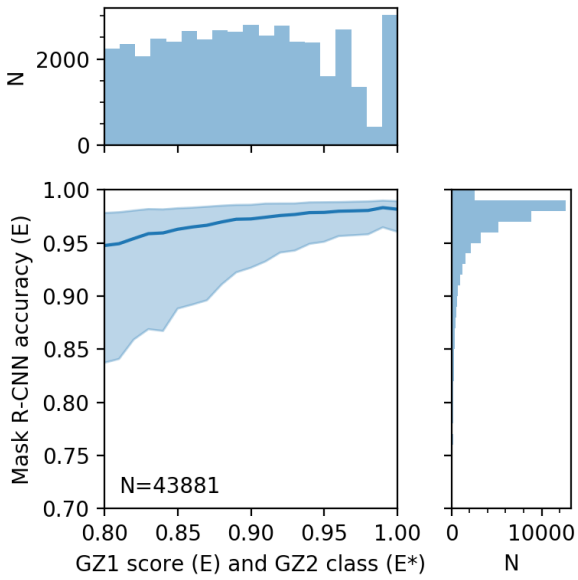


(d) Randomly selected galaxies to illustrate the cases in the other three panels. Each image shows the respective GZ1 score and Mask R-CNN accuracy on top. The red line corresponds to the segmentation mask edge.

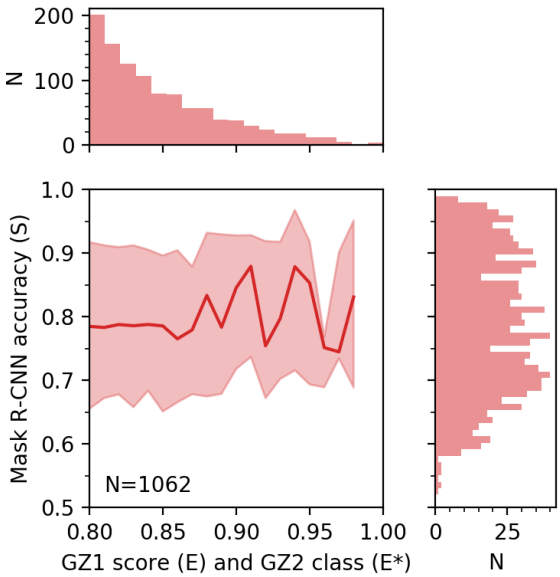
Figure 4.8: Mask R-CNN accuracy performance compared to galaxies classified as Spiral in both GZ1 and GZ2. In each plot, the thick line represents the median Mask R-CNN score for 1% bins in the GZ1 score for the S class. The shaded region displays the $1 - \sigma$ percentiles. The histograms show the distribution of the sample included within each plot, whose total counts are given by the value of N.

detected, segmented and classified, as shown in Figure 4.12. It is noteworthy to indicate that the model provides the location of the galaxies found and that this information can be incorporated into a catalog.

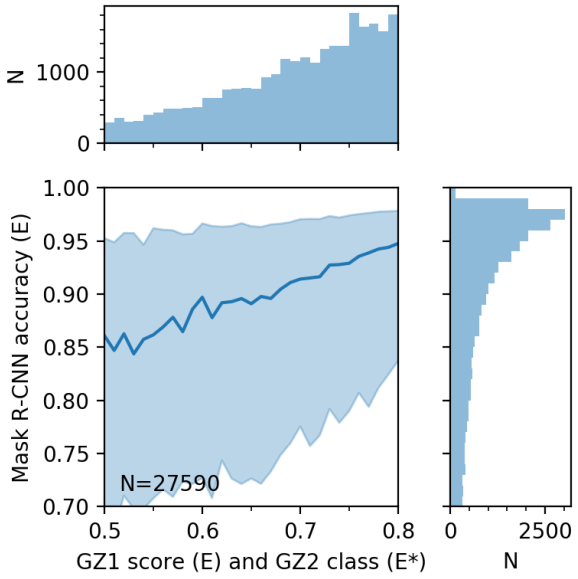
Semantic object segmentation and morphological classification are common tasks in astronomy. Typically, object segmentation is achieved with tools such like SExtractor. However, morphological classification requires the imple-



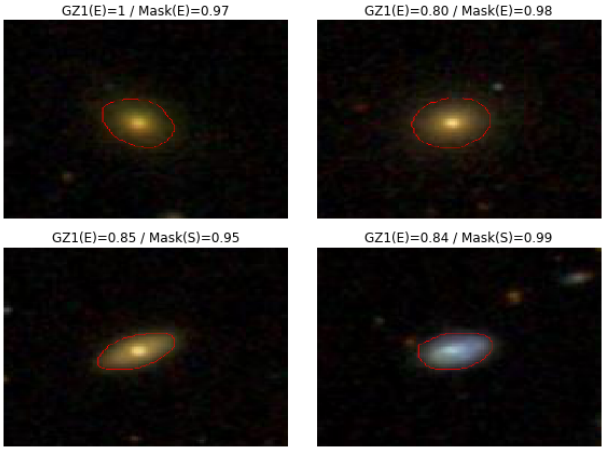
(a) Network results for galaxies classified correctly. The galaxies included comprise the *clean* GZ1 sample (i.e., $score > 0.8$) and were classified as E in GZ2.



(b) Mask R-CNN accuracy for galaxies misclassified as Spirals.



(c) Same as panel 4.9a, but for a GZ1 sample within $(0.5 < score < 0.8)$.



(d) Randomly selected galaxies to illustrate the cases in the other three panels. Each image shows the respective GZ1 score and Mask R-CNN accuracy on top. The red line corresponds to the segmentation mask edge

Figure 4.9: Mask R-CNN accuracy performance compared to galaxies classified as Elliptical in GZ1 *and* GZ2. In each plot, the thick line represents the median Mask R-CNN score for a 1% bin in the GZ1 score for E class. The shaded region displays the $1-\sigma$ percentiles. The histograms show the distribution of the sample included within each plot, whose total counts are given by the value of N.

mentation of a different suit of tools, which also depend on the approach (using descriptive quantities as proxies of morphology or CNNs over images). In this work, we have shown that both stages can be performed by this model through an automatic process using an architecture based on the state of the art of instance segmentation, associated with a training strategy based on transfer learning and differential learning rates. Considering the data scale of the upcoming surveys, such as the Rubin Observatory, it becomes necessary to advance both in accuracy and inference

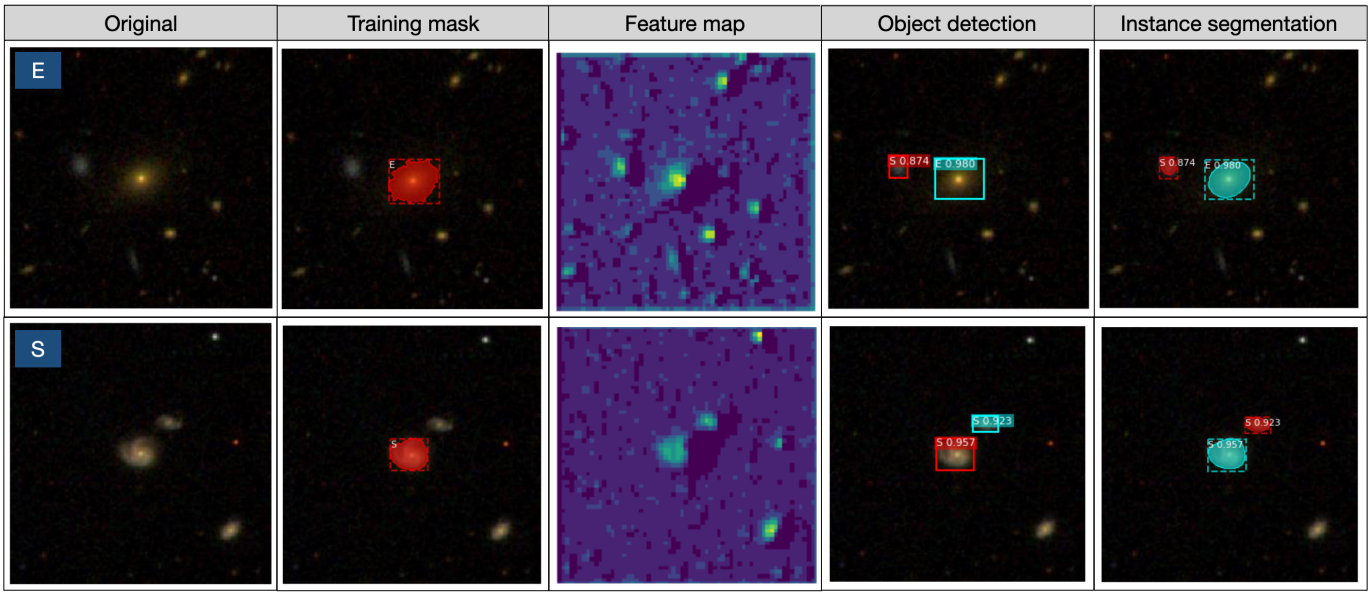


Figure 4.10: Automatic classification, location and segmentation pipeline applied to elliptical-type and spiral-type of galaxies

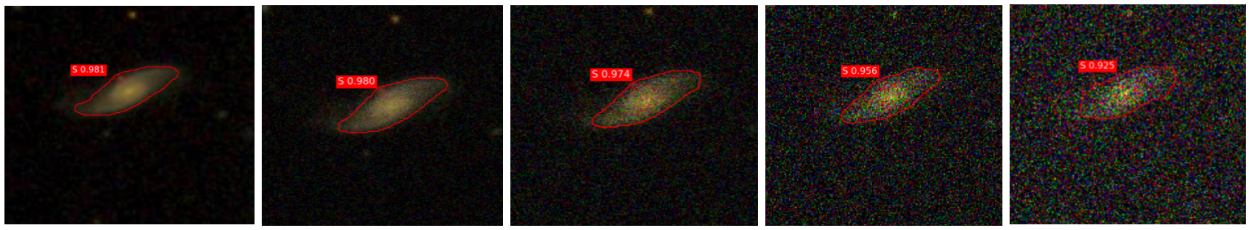


Figure 4.11: Response of the Mask R-CNN to chroma noise in the background. The leftmost panel shows an image included in the validation dataset. The other three panels correspond to the same image after adding chroma Gaussian noise with scales {12.75, 25.5, 51.0 and 76.5}, respectively, from left to right.

speed. In the case of accuracy, effort should be made in passing from e-channel RGB images to an input based on the multi-band photometric observations currently available. However, this increases the computational complexity due to the increase in dimensionality, so additional work should be made to advance inference speed. In this sense, the development of new techniques on dimensionality reduction while preserving the multidimensional structures and relationships present in these observations is required.

The global model reaches the accuracy of 93% of mAP. It is clear that there are many iterations to increase this percentage, especially when dealing with problems such as separation of galaxies with overlapping, galaxies of a very large size, among other problems. Implementing a single pipeline for the detection, segmentation, and morphological classification of galaxies has been validated, as well as peer validation of the second premise of this thesis. Our next chapter will discuss the last premise that supports the hypothesis, i.e. the creation of a compact version of the model that achieves an mAP that allows support of the model's scientific (astronomical) objectives. Moreover, this version of the pipeline provides results for analysis in the form of images with segmented sources, masks, and tabular data. It is possible to join these data with existing catalogs. It took approximately 2.5 hours to apply the pipeline to the entire Galaxy Zoo 2 catalog (243,480 SDSS images). This process was run on a P2.8XLARGEinstance provided by Amazon Web Services (AWS). In this instance, there are eight NVIDIA K80 GPUs and 488 GB of RAM.

Figure 4.12: Result of the application of the model to a query to the Skyserver. No color correction was applied. The result was 82 galaxies, correctly morphologically segmented.

5 Speeding-up convolutional neural networks

In the preceding chapter, we looked at how computer vision models are widely used in a wide range of image processing, video processing, speech processing, and natural language processing applications. However, a Deep Learning architecture for computer vision requires a significant amount of computing power, memory, and computing power [Inference and Learning, 2015]. In fact, this requirement is directly related to the design of the model architectures, since deep learning architectures are known to function in a state where the number of parameters substantially exceeds the number of training examples. A design like this is called over-parametrization, which means a model has more parameters than needed, in general, over-parametrization will lead to overfitting. It does not happen in deep learning architectures, there are several studies [Brown et al., 2020] [Sze et al., 2017] that address this intriguing issue, which is partly responsible for deep neural networks' success. Theoretically, there are enough parameters in the model for 100% accuracy to be achieved, but it is obvious that many of the hyperparameters determine this accuracy. The over-parametrization of neural networks provides mathematical support for stochastic gradient descent (SGD) to find a global minimum, but how to do so without running into an overfitting problem has been focused on by several authors [Zhou, 2021][Sankararaman et al., 2020]; however, it remains to be a field of study.

In general, the size of deep learning models is following a trend that continues to increase, because there is a correlation between their size and their ability to fit a wide variety of functions [Goodfellow et al., 2017]. In order for a neural network model to have a certain capacity, it must contain a certain number of nodes (with) and have a certain number of layers (depth). The relationship between the width of neural networks and gradient confusion has been studied in many ways. For example, in [Sankararaman et al., 2020], the authors concluded that increased neural network widths lead to lower gradient confusion, which results in faster model training. It is stated in [Goodfellow et al., 2017] that deeper models of neural networks can represent functions that are becoming increasingly complex as their depth increases.

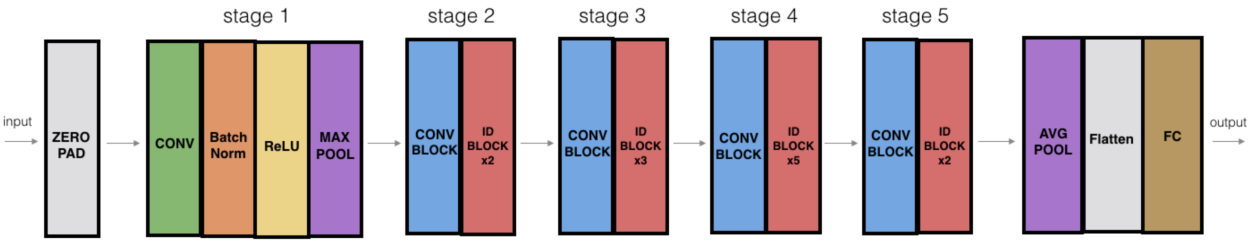


Figure 5.1: Resnet-50 model architecture.

In the field of neural networks, a model with more layers and more hidden units per layer is extremely powerful. The processing of this has a high cost in several aspects. We need to pay attention first to memory, because the limited memory bandwidth of today's DRAM devices is one of the biggest challenges in deep learning. The importance of this is particularly obvious in the process of training the input data, weight parameters, and activations as an input propagate through the network must be accounted for in the forward pass as it propagates through the network. In the back-propagation pass, these parameters are used to calculate the error gradients. For instance, in the case of the Resnet-50 [Xuelei et al., 2017], which is a powerful backbone model used in many computer vision tasks as in the architecture proposed here. Figure 5.1 shows that ResNet-50 has over 26 million weight trainable parameters. The parameters are further divided into five stages, each with its own convolution and identity block. In each convolution block, there are 3 convolution layers, and in each identity block, there are also 3 convolution layers. In order to train this model, it is necessary to compute ~ 16 million activations in the forward pass.

In a traditional approach, these weights would be represented as 32-bit floating-point data in order to store the weight as well as the activation in memory. Due to the fact that these parameters need to be kept in memory during backpropagation, as well as a step cache if the optimization is being performed (Momentum, RMSProp or another optimization technique). Additionally, we must consider other computational considerations. For example, modern GPUs have a memory limit of 6, 8 or 12 GB. There are cards with higher memory capacities, however they are prohibitively expensive. When using a batch size of 128, ResNet-50, con 224 x 224 RGB image, has a memory footprint of 13 GB. As this is a scenario of fine-tuning the model, it illustrates the need to explore alternatives to reduce the memory footprint generated when training these models. The model inference process, on the other hand, does not evade these Computational Considerations [Brown et al., 2020]. When deploying a deep network into a production environment, one of the most important aspects is the network latency. This could potentially have a negative impact on the capacity of the model to perform correctly, as it could disrupt its execution. The memory footprint of inference is the sum of parameters and activations stored in memory, but unlike training, activations are not accumulated. A focus on accelerating neural networks is motivated by their computational resource consumption, not taking into account

other aspects. A common input size now is 224x224 pixels, which forces many use cases to use this size for reasons of architecture reuse, as described in [Zhuang et al., 2021]. Due to the relatively low resolution of inference activations, it is not surprising that memory consumption has not been regarded as a bottleneck when it comes to inference activations. A preprocessing step is required before the input images can be included in the network, so the size of this process is an important consideration in real scientific scenarios. For the purpose of this project, we must also consider the aspects related to the inference of the model during its optimization based on the dimensionality and size of the astronomical data presented in the chapter 2.

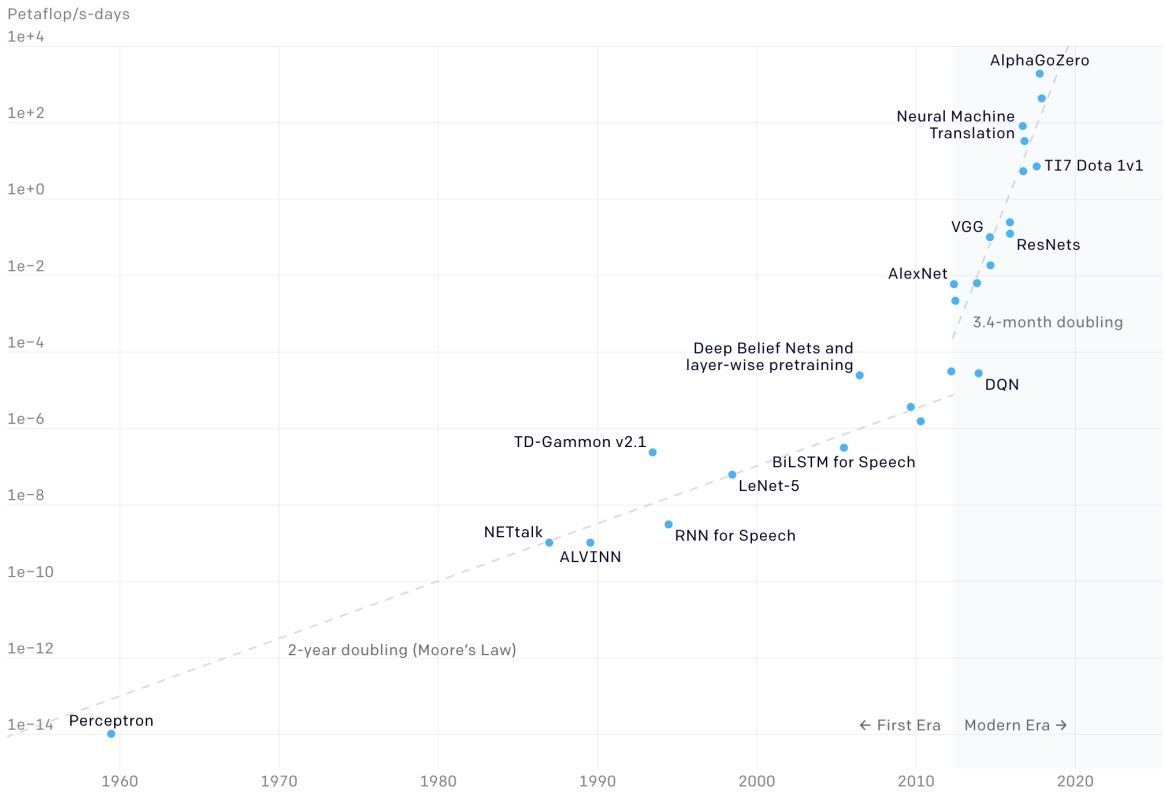


Figure 5.2: A recent analysis by OpenAI showed that the amount of compute used in the largest AI training runs has had a doubling period of 3.5 months since 2012 (net increase of 300,000x)

Finally, one of the emerging aspects of deep learning models that need to be considered is their computational power. As shown in figure 5.2, the energy consumption of models is also increasing, which has motivated a new line of research focused on developing hardware-friendly models and techniques. GPUs of today are capable of operating in a number of different power states, which are determined, for example, by the power demands of the models they are used to run. Various works [Hoefler et al., 2021]have been undertaken to reduce the size of neural networks through different approaches, in order to achieve higher scalability, performance, and energy efficiency. Currently, energy efficiency is a major concern for researchers developing deep learning models for astrophysical applications. An example of this is the fact that one of the key themes for the Astronomical Data Analysis Software and Systems Conference Series in its latest edition is the reduction of the environmental footprint of astronomy. In this conference, artificial intelligence guidelines for astronomical applications are defined.

According to the Computational Considerations presented in this paper, the model should consider methods designed to reduce the computational complexity and memory requirements of the proposed solution for automatic segmentation and classification of galaxies. Nevertheless, there is always a trade-off between reducing computational power, memory, and energy consumption while maintaining model performance. A variety of techniques will be discussed below, including results showing that the use of methods offer a powerful model that is more effective than the current SOTA in terms of applying an instant segmentation model to morphological classification of galaxies.

5.1 Model Compression

Thus, it is necessary to reduce the cost of processing these models while maintaining their performance. Using two approaches, we can make models that are faster and lighter. The over-parameterization of CNNs, i.e. the redundancy of neurons, is an inherent characteristic of CNNs. There is already a consensus within the SOTA of Deep Neural Network Acceleration methods that this redundancy can be mitigated without implying a significant or almost no loss of accumulation loss. Model parameters can be compressed or reduced in size by a variety of methods. Based on the results of their application in the model architecture, neural networks compression methods can be categorized. A division into three categories is presented by [Sze et al., 2020]: 1) Network structure, 2) Network optimization, and 3) Hardware design. A first approach focuses on optimizing the internal processes of the model in order to achieve model compression. It is noted in [Elsken et al., 2019] that many models suffer from design or conceptual problems during their development, since they are created manually. Following an approach based on automated neural architecture search, it should be possible to obtain models that are better suited to the nature of the problem and more efficient in their processing. As part of this approach, we also find techniques such as Knowledge Distillation[Ruffy and Chahal, 2019], which assumes that a training process from a master network to a student network will result in the latter emulating the master network but with a smaller model. Network Optimization [Lebedev and Lempitsky, 2018] seeks to modify the architecture of the model, optimizing the type of operations that must be performed during inference. We will find in this category approaches that reduce redundancy by working on the parameters of the model, such as Network Pruning[Reed, 1993] . Another approach in this domain is to reduce the precision of weights, biases, and activations at the level of their representation in memory. This family of methods is known as Network quantization [Krishnamoorthi, 2018], and it involves replacing datatypes with datatypes of reduced width.

There are a number of approaches to reduce the parameters of the network, but one of the most recent approaches is parameter factorization [Panagakis et al., 2021]. With the help of tensor methods, this family of techniques is aimed at reducing the parameters of the network. This can be applied either in the convolutional layer or in the convolutional layer decomposes higher-order tensors into lower-order tensors simplifying memory access and compressing model size while maintaining the multidimensional relationships. The final category of models or techniques for speeding up models involves the hardware used to process them. In this category, which is called neural network accelerators [Li et al., 2017], research is being done to optimize CPUs [Cornea, 2015] for model inference. There has also been considerable interest in the community with regard to optimizing them for training. In [Daghaghi et al., 2021] , back-propagation times based on sparse hash tables are compared to those achieved with GPUs. It is primarily due to the growing demand for GPUs, which has resulted in a shortage of GPUs available for sale. Several other models make use of memory in order to avoid long distance communication between memory and computation units. While this speeds up the processing of models, it has the disadvantage that it can lead to stability problems when reading the results. There are techniques described in [Wang et al., 2021] that optimize the reading process by offering a framework that, according to the authors, allows full recovery of most models' state-of-the-art accuracy, as well as achieving at least an order of magnitude greater energy efficiency.

The purpose of this work is to integrate techniques that belong to the category of Network Optimization. In the next section, we will discuss each of the approaches that dominate the literature in this field.

5.2 Network Pruning

Pruning is the most dominant method of compression, according to the literature. According to this approach, neural networks have redundant parameters, and it is thought that these parameters have a negligible value and can be ignored since they are redundant. It is interesting to note that this method is inspired by the process of synaptic pruning in the human brain. Pruning involves identifying neurons that contribute less to the model task through a series of stages. The objective is to remove these neurons without affecting the performance of the network. There is an extensive literature around an idea that seems simple, however it has been demonstrated that the correct elimination of the components of the model does not necessarily depend on the extent to which they contribute to the inference. There are three axes that form the core of the entire domain:

Where pruning should be applied - We are speaking at the structural level of the model. This technique can be applied at the weight level (Fine Pruning) in order to make the weights (matrices) sparse. In practice, this means replacing the values of the weights to be eliminated with zeros, resulting in the pruning of a connection on the network. Due to the absence of restrictions, this approach is often referred to as unstructured pruning. Although this method has the advantage of being rapid in application, it has a greater disadvantage. It should be noted that the mechanisms for speeding up operations on sparse arrays are not necessarily included in the implementation of the models. Due to the fact that each weight with a 0 value occupies as much space as before, matrix multiplication is not faster.

Conversely, we have methods that avoid working with sparse matrices, they focus on the complete structure of the model. In particular, the works that focused on pruning entire convolutional filters are noteworthy. Using this approach results in sparse layers, which further implies the elimination of feature maps that are outputs of said filters. In contrast to the weight-based approach, we will be removing rows and columns from a weight matrix in this instance. They are referred to as structured pruning methods since they are designed to prune larger structures.

Pruning: How to do it - As Upon identifying the target structure to be pruned, criteria must be established to determine which structures should be preserved. As a result of ranking the importance of the parameters or filters, the first approximation decreases the weights below a certain threshold. Therefore, an approach based on the magnitude of the weight can be eliminated, provided that the magnitude is associated with the fact that the weights do not significantly contribute to the model's objective. Using this approach in unstructured pruning is quite straightforward, but it becomes more complex in full structure pruning. Several approaches have been developed to address structured pruning based on the weight magnitude criterion. A learnable multiplicative parameter or gate is added to the model after the layer that we wish to prune, using the L1 norm of the weights of each convolutional filter. This parameter is incorporated in a pruning iteration process during training. Each iteration, all gates are ranked, then the filters with the lowest ranking are pruned. Additional criteria are beyond the approach based on the magnitude of the weights. According to, the authors aim to increase the gradient magnitude in order to measure the effectiveness of pruning.

In this approach, gradients are accumulated over a minibatch of training data rather than weights. By multiplying the weights with this gradient, we are able to rank them.

Pruning should be applied when - After the pruning structure and criteria have been defined, the method for pruning the neural model must be selected. In traditional pruning methods [Micikevicius et al., 2017] three stages are involved: training, pruning and fine tuning. This approach is known as static pruning and involves pruning the network once the training has been completed. However, this approach involves additional computation and training time, which is at the expense of speeding up the network. In the case of the so-called dynamic pruning, this process is carried out during training time, aiming to remove the whole fine-tuning process. This approach has been followed by [Cornea, 2015], [He et al., 2018], [Renda et al., 2020] with various approaches. [Renda et al., 2020] call their method Learning-Rate Rewinding, because it involves fully retraining a network once it has been pruned.

5.3 Network Quantization and binarization

The training of neural networks has traditionally[Sze et al., 2020] been performed using a floating point format called single precision, or FP32. This format utilizes 32 bits in computer memory and offers 7 to 8 valid digits of precision. The deep learning models can, however, function with smaller representations, such as FP16 or even FP8, if this level of precision is not always required. An analysis of mixed precision training was conducted on the NVIDIA technical blog in [Micikevicius et al., 2017] as part of the discussion of their Volta GPU series' half-precision support. According to this study, the majority of weights and gradients can be represented accurately within the 16-bit range of representability, whereas scaling up was sufficient to achieve convergence for gradients that did not. According to the authors, mixed-precision training reduces the amount of resources required by using lower-precision arithmetic, which is beneficial in two important ways. The switch from FP32 to FP8 would, for example, result in a fourfold reduction in model size, resulting in a reduction in memory requirements. Additionally, there is the option of reducing training or inference time by speeding up computations on the GPU (reducing the arithmetic bandwidth) and (in the distributed case) by reducing the network bandwidth. This reduction, however, entails risks since the representation may not be able to capture, for example, the changes in gradients during training. Due to the fact that the representations in memory required to perform complex functions, such as those required by deep learning, are designed to avoid problems such as underflow, where the weights of our network are so small that they clamp to zero, this would significantly affect the model's convergence. As a result of very high weight values, when changing from FP32 to FP8, for example, we may experience an overflow problem where the weight becomes NaN, not a number, when going from FP32 to FP8.

A large amount of research has been conducted on the subject of quantization, a method of reducing memory representation. In order to maintain accuracy of the models, these methods try to reduce computation demand and increase power efficiency. As outlined in [Wang et al., 2021], the quantization approach relies on working on the representation of network parameters in memory. Quantization can be applied in two ways, depending on whether training is included in the process. In the first family of methods[Banner et al., 2019], the reduction is applied after the training phase, i.e., after the weights and inputs have been trained in FP32. A quantized model can be selected for the network by utilizing quantization techniques and re-optimizing it. Due to the fact that the neural network is frozen, it cannot be updated, which makes it extremely easy to apply. Nevertheless, of the two approaches, this one poses the greatest risk of higher accuracy loss because all quantization-related errors occur after training is completed and cannot be corrected.

Second, research focuses on developing methods that take into account the training phase, hence the name Quantization-aware training (QAT). In addition to the quantization process applied to the weights, gradients are also calculated for the quantized weights. These methods may include fine-tuning a stable full precision model or retraining a quantized model.

5.4 Parameter factorization

The use of tensor methods in machine learning models seeks to extend the matrix approach of these models to higher dimensional spaces. These methods have been applied in a variety of areas within machine learning research. In [Rahimi and Homayounpour, 2020], the authors use an unsupervised method based on Tensor factorization for extracting a multi-view document embedding. Among the applications of tensor methods in NLP is [Rahimi and Homayounpour, 2021] which uses a tensor-based method to analyze sentiments by integrating word cooccurrence and word polarity information. This proposal focuses on accelerating deep learning models as a result of the application of these methods to computer vision tasks. The use of tensor decomposition in this area falls under the category of parameter factorization, as it is one of several techniques we have presented to speed up deeplearning models. As a result, deeplearning models will be less overparameterized but also feature a distinctive feature. Chapter 3 demonstrated that tensor decomposition can preserve relationships or multidimensional structures in multidimensional data, which were validated. The same feature is incorporated into the reduction of model parameters. As can be seen in figure 5.3, the left side shows the feature map of layer without factorization. In the image on the right where the Tucker decomposition has been applied, we can see that the present structures are preserved.

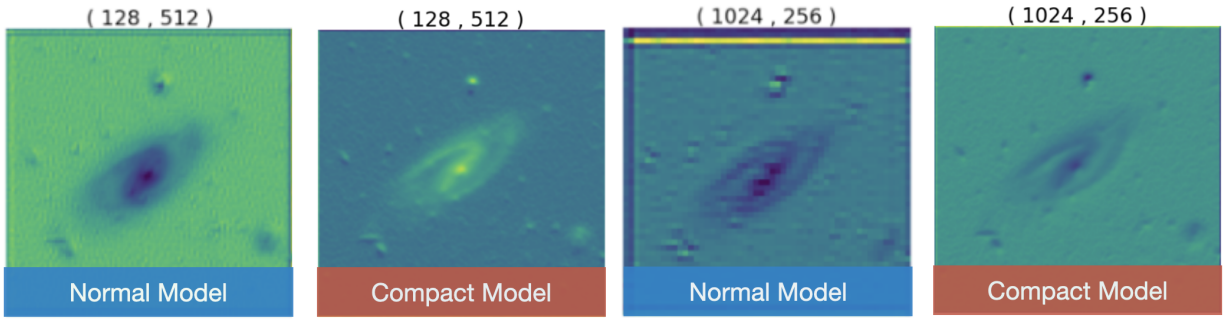


Figure 5.3: In both the compact and normal versions of the model demonstrates how the application of the proposed technique preserves the most relevant information more effectively.

Furthermore, other studies have suggested that tensor methods have other advantages, in addition to the advantages of reducing parameters in the architecture and accelerating models. [Panagakis et al., 2021] argues that tensor methods produce more parsimonious models when applied to models when applied to parameter calculations. Furthermore, the resulting models share the same advantages as the other approaches in terms of both disk space and memory usage. Additionally, these decompositions result in more robust noise models, as the authors emphasize in their report. Since it was necessary to verify the ability to represent astronomical data using tensors, this feature was also validated in chapter 3. An interesting feature that will be developed in the next chapter is that models optimized with tensor methods present better results in domain changes. When it comes to astroinformatics, this is important because, in many deep learning models used for astronomical data, part of the architecture was trained using different types of data, such as the COCO dataset.

The concepts and methods of tensors in mathematics and computer science were discussed in detail in Chapter 3. The Tucker decomposition is a generalization of the SVD factorization in higher dimension spaces, which falls under the category of parameter factorization as a method for accelerating deep learning models. Based on the layers of already trained models, this approach seeks to generate compact versions of these models from these layers. The process focuses on 2D convolutional layers in order to generate low-rank versions of these structures. The reason for this is that these layers typically consume 50% to 90% of the total processing time for the architecture, depending on the model. The substantive difference with the pruning techniques is that tensor decomposition methods use only the weights of a layer, not including the forward pass to backward pass processes. Therefore, the multilinear structure of the input data to the model is preserved as a result. The substantive difference with the pruning techniques is that tensor decomposition methods use only the weights of a layer, not including the forward pass to backward pass processes. Therefore, the multilinear structure of the input data to the model is preserved as a result.

In [Kim et al., 2015], the authors propose generating a compact representation using Tucker decomposition on a VGG that can be displayed on mobile devices with limited computational power. As a fundamental principle of this approach, a 2D convolutional layer is considered a tensor with four dimensions: cols x rows x input_channels x output_channels. According to chapter 3, when applying the Tucker decomposition we obtain two types of products, a core tensor and, in this case, four matrix factors. In this case, a 2D convolutional layer would then be divided into several smaller convolutional layers. Under the assumption that each layer is overparameterized and can be represented by a tensor of lower rank, this method works layer by layer. The most remarkable thing about this approach is that although the result is a greater number of layers, the authors have validated that the total number of parameters will be smaller as a result of this approach. Further, the number of floating point operations is reduced, which is a combination of results that cannot be obtained from other accelerators. Therefore, this was the primary reason for selecting this method for this proposal.

Table 5.1: tucker no VBMF

Layer	in_channels	out_channels	kernel_size
Original(Conv2d)	64	64	(1,1)
Tucker(Conv2d_0)	64	15	(1,1)
Tucker(Conv2d_1)	15	15	(1,1)
Tucker(Conv2d_2)	15	64	(1,1)

Tucker decomposition involves three stages: (1) rank selection, (2) low-rank tensor decomposition, and (3) fine-tuning. A critical element that conditions the stability of the accuracy after the application of Tucker is the correct selection of the rank with which this decomposition is applied. The rank of the convolutional layer in the future corresponds to the size of the core tensor. However, the authors in [Hillar and Lim, 2013] have shown that finding the ideal rank decomposition, like most tensor problems, is NP-Hard. This represents one of the main drawbacks associated with Tucker's method when applied to convolutional layers. Experimentation of different sizes of the core tensor is one strategy, while monitoring the accuracy of the model in its compact form at the same time. As shown in table 5.1 , Tucker was applied with ranks of [15, 15], corresponding to the size of the core tensor and the size of the future convolutional layer to the first convolutional of Resnet-50. Tucker was applied using ranks of [15, 15]. A new convolutional layer is created by replacing the original layer with the decomposition product (tensor core and factor matrix).

Table 5.2: Tucker VBMF

Layer	in_channels	out_channels	kernel_size
Original(Conv2d)	128	128	(3,3)
Tucker(Conv2d_0)	128	50	(1,1)
Tucker(Conv2d_1)	50	53	(1,1)
Tucker(Conv2d_2)	53	128	(1,1)

It can be observed in this case that the core tensor has symmetrical dimensions, but that is not a constraint. To provide stability to model accumulation, it is necessary to find a compact representation which preserves the most relevant data. All convolutional layers can be replaced by compact representations as a result of this process. As a result of the size of the models being evaluated for the backbone, this manual testing process by layers is not feasible. A ResNet50 has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer, for which mathematical support is necessary to find an automatic approximation to the best possible kernel rank. The SOTA in this area is the application of variational Bayesian matrix factorization [Nakajima et al., 2013] as a method for estimating rank. Table 5.2 shows the result of applying Tucker with VBMF estimated ranks of [53, 50]. This was an automatic process, which provided the mathematical support for applying this approach to the entire convolutional network, which is the approach used in this proposal.

6 Tensor Mask

The model proposed is based on Mask R-CNN. In the architecture of this network is the Backbone which is a standard pre-train convolutional neural network (CNN) that serves as a feature extractor. As is known in the implementation of deep neural networks, one of the factors that restricts the depth of these models is the computational cost of processing these architectures. One of the reasons for this computational cost is the amount of parameters that need to be processed in the full connected layers. To solve these problems, are propose incorporate tensor-train[Novikov et al., 2015] layers into the architecture of CNN networks. These tensor-train layers point to more economical parameterizations for fully connected layers. Such compressed parameter spaces lead to a reduction in computational costs.

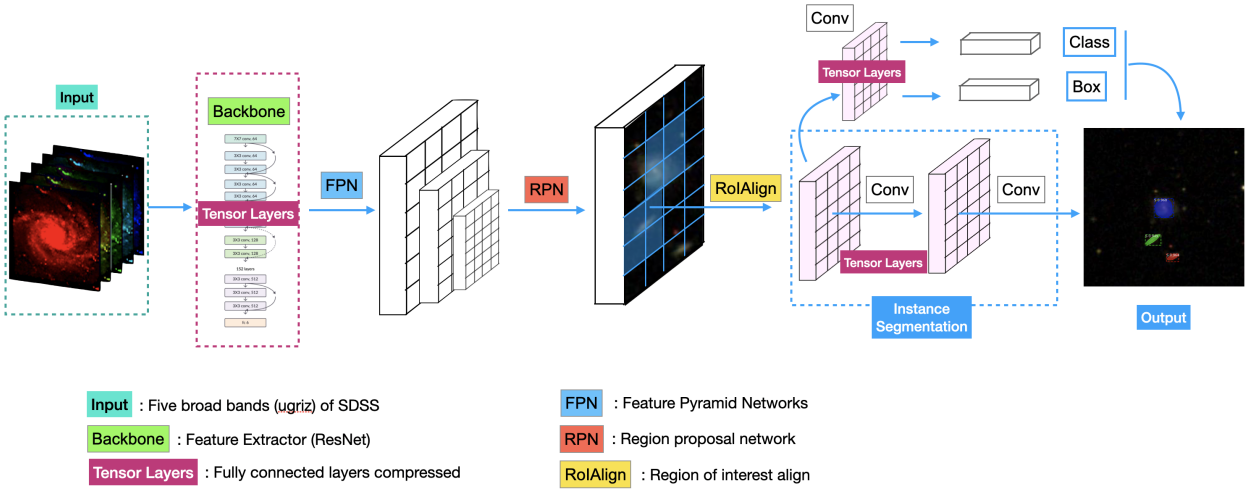


Figure 6.1: Tensor Mask R-CNN model architecture.

Figure 6.1 illustrates the differentiated elements of the current state of the art in this type of instance segmentation model. This work contributes to the state of the art by replacing the backbone of the mask R-CNN model with a compact version that is more efficient in the use of computational resources and therefore faster. In this case, we are using the Tucker method, which falls within the category of factorization techniques designed to accelerate neural models. As of the time of writing this thesis, it is the first to address the problem of accelerating instance segmentation models using this aproach. The same applies to the segmentation of galaxies based on their morphological characteristics. Each of the necessary stages for the construction of this automatic pipeline of morphological segmentation of galaxies will be discussed in greater detail in the following sections of this chapter.

6.1 Dataset creation

One of the considerations for the use of convolutional neural network (CNN), in general, is the need of a training data set of images able to satisfy the following three requirements. First, the objects in the dataset have to contain a label indicating the class to which each object belongs to. Second, CNNs require all classes to be equally represented in the dataset; and third, the need to split the dataset into training and validation sub-sets. This last requirement implies that the dataset needs to contain many images —usually several thousands. In the case of Mask R-CNN, it is also necessary to incorporate an additional element: the mask. This last element is a key factor for the architecture, because it provides the segmentation shape (and implicitly the location) of the objects.

In the context of classification of astronomical sources, there is no validated, standardized and prepared dataset for Deep Learning training. To overcome this limitation, we need to create a dataset of images, labels and masks. Below we describe the steps for the creation of the training set. In this section, we describe the dataset used in the present work, which consists of images of galaxies with their respective morphological classification.

6.2 The GalaxyZoo DECaLS dataset

The Galaxy Zoo project [Lintott et al., 2008] was the first astrophysical application of a “crowdsourced” solution to the problem of classifying large datasets of images, which became basically an impossible task for professional astronomers due to the immense amount of data available. In practice, Galaxy Zoo classifications were made by volunteers through a web portal that presented a simple morphological classification scheme of galaxies, implemented as a decision tree

based on questions, along with an image of a galaxy from the Sloan Digital Sky Survey (SDSS, [York et al., 2000]) . The project received more than 4×10^7 classifications from around 10^5 participants, and produced several catalogs — the largest one comprising over 300,000 galaxies reliably classified. Besides these large and precise datasets of classified galaxies, which can be seen as a rich dataset of labeled images for Computer Science applications, this initial Galaxy Zoo initiative provided proof that the general public can classify large datasets with a precision similar to that of professional astronomers.

In a more recent reincarnation, the Galaxy Zoo DECaLS [Walmsley et al., 2022] is a similar citizen-science initiative that utilizes the images from the Dark Energy Camera Legacy Survey (DECaLS). This dataset contains deeper astronomical images (r-band magnitude of 23.6 instead of r-band magnitude equal to 22.2 for SDSS images), which translates to images of galaxies that exhibit more morphological features. As a consequence, the decision tree presented to the human classifiers has been adapted to include the variety of features present in the data. This decision tree contains five levels of branches (see Figure 4 of cita3), with paths that depend on the answers. The data products of Galaxy Zoo DECaLS include image stamps in JPG formats, along with catalogs with the fraction of votes received for each classification along the decision tree. These votes and images are next used in the following subsection.

6.3 Creation of a balanced subset

We created a dataset for training, validation and testing of our CNN based on the catalogs (classifications) and images provided by the Galaxy Zoo DECaLS. We paid particular attention to creating a balanced dataset per class while keeping a number of unique objects per class reasonably large. This is important because not all data-augmentation techniques can be applied to all classes of galaxies. For instance, while rotations or reflections may work well for galaxy mergers or spiral galaxies, the symmetrical and featureless aspect of elliptical galaxies make the use of this technique in particular irrelevant. Because the classification for each object is performed by multiple participants independently as a measure to minimize errors, each galaxy has a distribution or fraction of votes between the options for a given question in the decision tree. In particular, we constructed our dataset to include four major classes: Elliptical, Spiral, Barred Spiral, and Merger. Elliptical galaxies were selected by requiring smooth-or-featured-smooth-debiased > 0.6 and smooth-or-featured-featured-or-disk-debiased < 0.4. Spiral galaxies were selected as having smooth-or-featured-smooth-debiased < 0.4 and smooth-or-featured-featured-or-disk-debiased > 0.6 and bar-no-debiased > 0.7. This criteria will include edge-on galaxies too. On the other hand, Barred Spirals were selected as having smooth-or-featured-smooth-debiased < 0.4 and smooth-or-featured-featured-or-disk-debiased > 0.6 and disk-edge-on-no-debiased > 0.7 and bar-strong-debiased > 0.3. Finally, Mergers were selected by requiring merging-merger-debiased > 0.4. Finally, we selected 6,000 objects for each class, except for Merger as it only contained 2,229 objects.

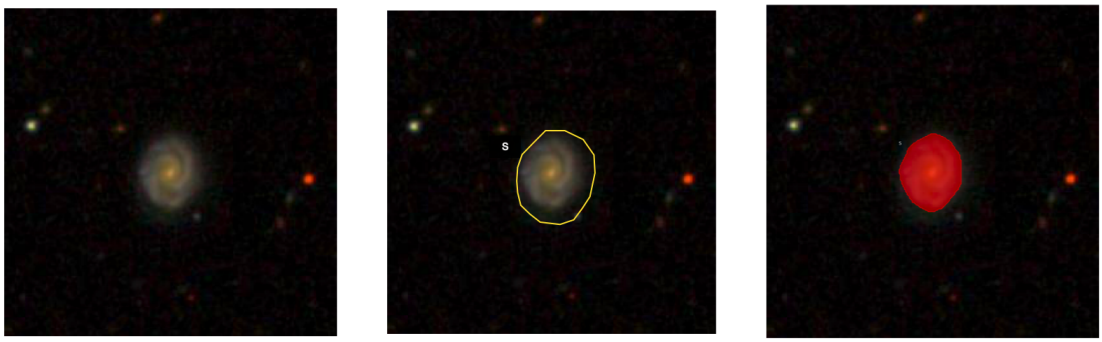


Figure 6.2: Creation of a training mask over an image. Left: Original image obtained from the GalaxyZoo DECaLS dataset, centered at the Equatorial coordinates of the object listed by Galaxy Zoo. Center: Mask created following a manual process using the VGG Image Annotator tool. Red dots are marked by the user, while green lines are linearly interpolated between the dots by the tool. Right: Mask created by an automatic process (red) overlaid on the input image. The mask was calculated using image manipulation libraries in the Python programming language, for the central object in the image.

6.4 Dataset masks: Automated data annotation

The final step to build the dataset is the creation of the masks. The mask serves the purpose of delineating the labeled objects in an image that will be used after for training and validation. The mask creation can be either a manual or an automated process. Several works use manual annotation tools, such as VGG Image Annotator [Dutta and Zisserman, 2019], LabelBox [lab] , LabelMe [Torralba et al., 2010] , COCO UI [coc] or RectLabel [Rec]. These tools allow the

user to create the mask for each labeled object in an image manually. This process is illustrated in the middle panel of Figure 6.2, where a galaxy is marked manually over a SDSS image using the VGG tool.

Figure 6.2 presents the results of the automated mask creation method, using the center of the image as a reference. The masks were created using an automated annotation process, based on image processing algorithms from the Scikit-Image package in Python. The object, located in the center of the image, was identified using the Otsu method. This method is used in computer vision to perform segmentation by dividing an image into significant regions or objects, and then delivering a binary image as a result which is interpreted as a boolean mask. The basic idea behind the Otsu method is to select an optimal gray level threshold value that will separate the background and objects of interest in an image based on their gray level distribution. In addition to the method described above, we used the imantics library [ima] to perform a semantic analysis of the image and to have greater precision in the segmentation of the object.

6.5 Data Augmentation

The training and validation datasets are able to satisfy the needs of the problem in terms of volumes and a balanced representation of examples for each class. As indicated, the use of transfer learning based on the backbone of the network and the training strategy also provide a base knowledge to train the model. As indicated by some works [e.g., Mikołajczyk and Grochowski, 2018], the use of Data Augmentation helps to improve the ability to generalize the model. In the Data Augmentation process, we used the imgaug [Jung et al., 2019] library with 122 possible image augmenters and augmentation techniques. These Data Augmentation techniques include Affine transformations, perspective transformations, contrast changes, Gaussian noise, dropout of regions, hue/saturation changes, cropping/padding, blurring, among others. But, as it can be seen in Figure 6.3, not all augmentation techniques have scientific relevance for the characteristic of astronomical images.

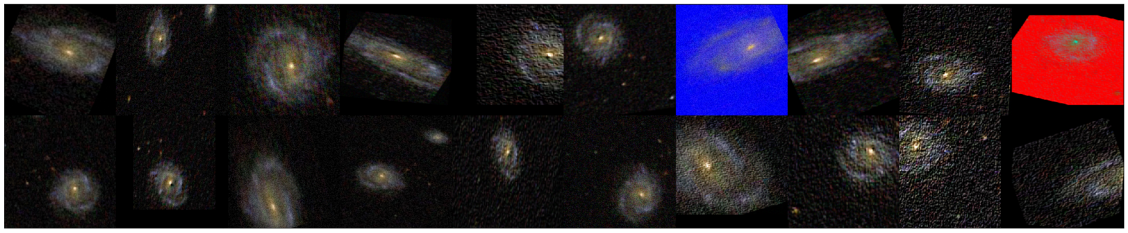


Figure 6.3: Examples of different data augmentation techniques included in the imgaug library and applied to the same image of a spiral galaxy. As mentioned in Section 6.5, only techniques that are possible in an astronomical context were used for this work.

Specifically, Figure 6.3 makes clear that some techniques, such as contrast changes, are not applicable because they affect the chromatic characteristics of the data in an unrealistic shape in the context of astronomical data. Thus, these type of techniques do not add a value to the dataset and were not used. On the other hand, there are geometric transformations, such as Affine, which enable rotations that can be performed given the scientific nature of the problem. There are also techniques in this category, such as PiecewiseAffine, that work on the basis of fixing anchor points and then randomly moving the neighboring pixels around this point. This leads to morphological distortions of the galaxies in the images.

Figure 6.4 displays the Data Augmentation Pipeline applied to the dataset. This pipeline consists of a sequence of four techniques: Fliplr, Flipud, OneOf (rotates in different grades) and AddElementWise. The first of these techniques is Fliplr. It consists of performing a horizontal mirror of the image, in this case it was applied randomly to 50% of all input images. In the case of Flipud, the probability parameterization is the same, it only differs in the rotation, which is vertical. The OneOf augmenter was incorporated to increase the geometric heterogeneity of the images. It is intended to always perform exactly one of the techniques that groups this block. The OneOf includes also three possible configurations of the rotation technique (90, 180 or 270 degrees) of the images. Finally, we have the arithmetic technique called AddElementwise, which aims to create new images by adding slightly different pixels to its neighbors. With the incorporation of this Data augmentation Pipeline, the dataset was considerably increased, allowing it to go from a mAP of 0.81 to one of mAP of 0.93.

6.6 Model Implementation and Training

An instance segmentation architecture based on Mask R-NN is presented in section 4.1, which evaluates the effectiveness of such architectures in morphologically classifying galaxies. Although the objective was achieved, the conclusions of this iteration of the proposal indicated the need to modularize the architecture components, thus creating a pipeline that

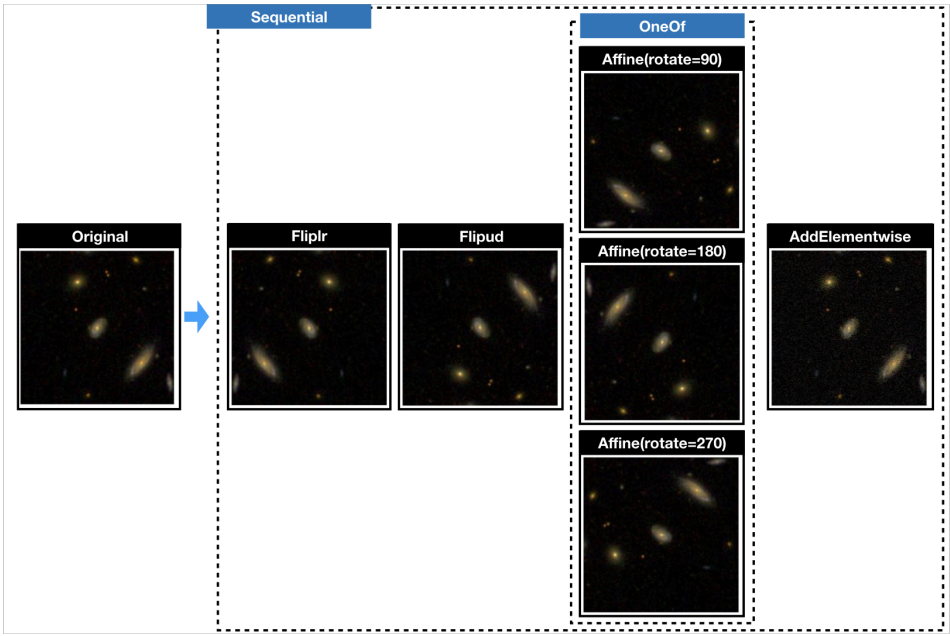


Figure 6.4: Data Augmentation pipeline.

can optimize each component in accordance with the MLOPS approach. This aimed to optimize each of the models that are part of the pipeline. Although the backbone of the architecture was configured in chapter 4, its training was integrated into the training of the Mask R-CNN. Transfer learning was thus initiated from a network trained on the 81 MS-COCO Lin et al. [2014] classes, not from a classifier from the same problem domain. The main issues with the original pipeline were classification errors in and sources larger or smaller than the average size of the dataset objects, as I indicated earlier. It was corrected by training a classifier independently in the same data domain as the final segmenter to correct the latter. Figure 6.5 illustrates Backbone training, which yields a classification model of four classes of galaxies based on their morphology. This model can be changed and re-trained as many times as estimated. Once the model satisfies the scientific requirements of the problem, it is incorporated into the second pipeline preparation stage. In this stage, compression methods based on tensor algebra will be utilized to generate a compact representation of the model, with the intention of speeding up the pipeline and the backbone. This compression process is not mandatory for the pipeline to work because, as will be seen in the following sections, the pipeline results were compared with the compressed and uncompressed backbone in order to validate this thesis' hypothesis. As soon as a compact representation of the backbone is generated, it is necessary to fine-tune the model in order to increase its accuracy. The main contribution of this approach lies in the design of a pipeline with a low level of code dependency between its components. Applied to galaxy morphology classification, this model is an improvement on the SOTA of a Mask R-CNN based instance segmentation model.

In the following sections, we will discuss each stage of pipeline preparation. This chapter will conclude with the design and implementation of the final pipeline.

6.7 Training the backbone

According to section 4.1, the backbone is an important component of object detection and instance segmentation models. Additionally known as the Feature Extractor or Encoder, it is responsible for creating the feature space of the model (hereinafter the feature map). In Figure 6.6 we show the feature maps of the backbone(Resnet-101). It can be seen how different layers are sensitive, after training with our dataset, to astronomical objects at different spatial scales.

CNNs are currently the dominant backbone architectures. The main networks used for this are AlexNet Krizhevsky [2014], VGGNet Krizhevsky [2014], ResNet He et al. [2016], and ResNeXt Xie et al. [2017]. In the original architecture of Mask R-cnn Abdulla [2017], both with configurations of 50 and 101 layers were evaluated for the Resnet backbone. Additionally, Lin et al. [2017] have proposed a hybrid architecture called a Feature Pyramid Network (FPN) that can be incorporated. This model is composed of a bottom-up and a top-down pathway for that takes an image (single-scale) of an arbitrary size as input and outputs proportionally sized feature maps at multiple scales. Mask R-CNN originally

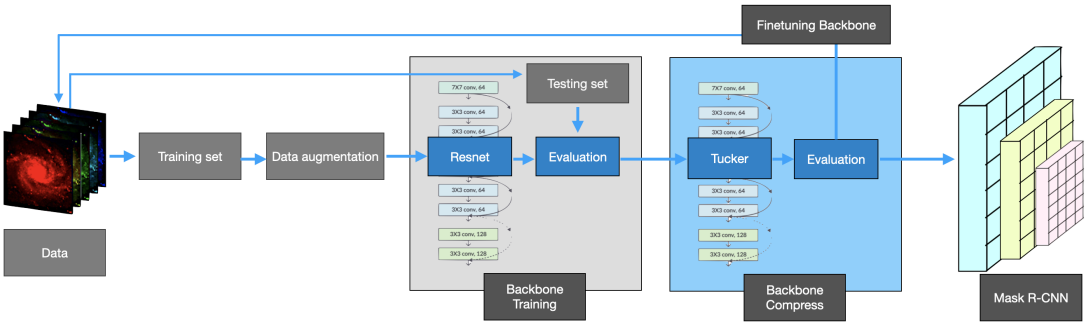


Figure 6.5: The process of training the model, as well as the compression of its backbone.

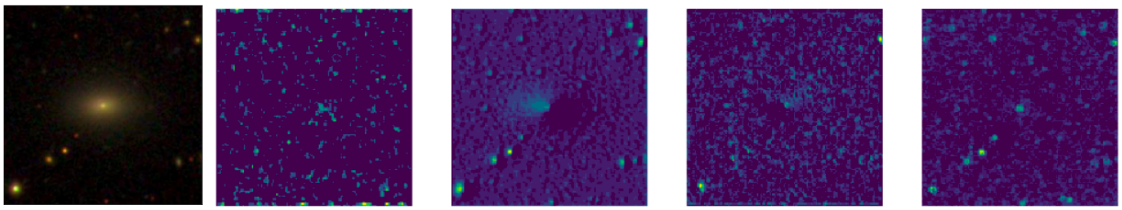


Figure 6.6: Backbone Feature Map. The leftmost panel shows an input image passed to the network. The other four panels show different feature maps produced by the backbone in block 2 of a resnet-101. In the second map (center panel) the three galaxies are clearly visible, whereas the smallest sources are clearly highlighted in the rightmost panel.

Table 6.1: In the class weight column, the distribution of examples for each class in the dataset is the result of the algorithm that assigns a factor for classes with the lowest representation.

Target class	Samples	class weight
Elliptical (E)	6000	0.63
Merger (M)	2229	0.86
Spirals (S)	6000	0.63
Barred Spirals (SB)	6000	0.63

had a ResNet-101 backbone architecture with FPN, which outperformed SOTA for instance segmentation at that time. Backbone design and implementation play a crucial role in the final quality of the model Tan et al. [2020].

Table 6.2: The hyperparameters that were used in the training process of the instant segmentation model are configured as follows.

Learning rate	Momentum	Optimizer
0.001	0.9	Stochastic Gradient Descent

Four architectures were trained for selection of the backbone: Resnet18, Resnet 50, and Resnet 101. Based on the recommendations from the official Python implementation for the hyperparameters of these models, all of these models were trained with the configuration presented in table 6.2. Two training processes were conducted for each model, one with 20 and the other with 40 epochs. Each of these training sessions was conducted using the data argumentation technique discussed in section 7.4. It was also necessary to address the problem of class imbalance as shown in table 6.1. To mitigate this problem there are various approximations. From the more traditional ones like *undersampling*, which consists of incorporating fewer samples from the majority class. This implies loss of information and therefore increases the risk of overfitting. On the other hand is *Oversampling*, which consists of creating samples copied from minority classes. A specialization of this concept is the Synthetic Minority Over-sampling Technique (SMOTE). This method relies on generate synthetic samples from the dataset to increase the minority classes impact on the model. SMOTE is a powerful technique, but it does have some issues that limit its use. For example, it does not behave well in datasets of high dimensionality or cardinality. Given the characteristics of the prepared dataset, there is another method for handling imbalanced data. This method is called *Class Weight*, and has particularly potent in models deeplearning characteristics. The previous techniques work on the inputs or samples of the model, in the case of class

weight, it works on hyperprameters of the model. Work on network weights modify the loss function directly during training. That is why a network where this technique has been applied to address the imbalanced data problem is called Cost-Sensitive neural network or Weighted Neural Network. In general terms class weight is a function that modified to weight misclassification errors given more weightage to the minority's classes in the cost function. There are several approaches to calculate the class weight factors. In this paper a *balanced* approach is followed. Where the distribution of the class weight factors is inversely proportional to the distribution of the classes. In table 6.1 we can see the weighting that affected the samples of the different classes during training.

Table 6.3: As a result of the training of six versions of the classifier (Resnet), the standard for this model has been reached. As an important point to note, the backbone of the instant segmentation model will be trained in the same data space as those to which it will be applied. Additionally, despite having approximately half of the parameters of the resnet-101, the resnet-50 achieves an outstanding result.

Model	Epoch	Acc	Training time
Resnet-18	20	0.78	50m 3s
Resnet-18	40	0.78	92m 25s
Resnet-50	20	0.78	88m 39s
Resnet-50	40	0.79	177m 20s
Resnet-101	20	0.79	145m 21s
Resnet-101	40	0.79	292m 40s

A detailed analysis of the results of the 6 training sessions can be found in table 6.3. In the case of Resnet-101, the top-1 accuracy reported in He et al. [2016] for Image Classification on ImageNet is exceeded. However, the same does not apply to Resnet-50, where the top-1 accuracy is 80.1% in the work of Shen and Xing [2022] , and the trained model is found to reach 79% after 40 epochs. It is also important to take into account the training time, which will be compared to the inference time for compressed models in the following section.

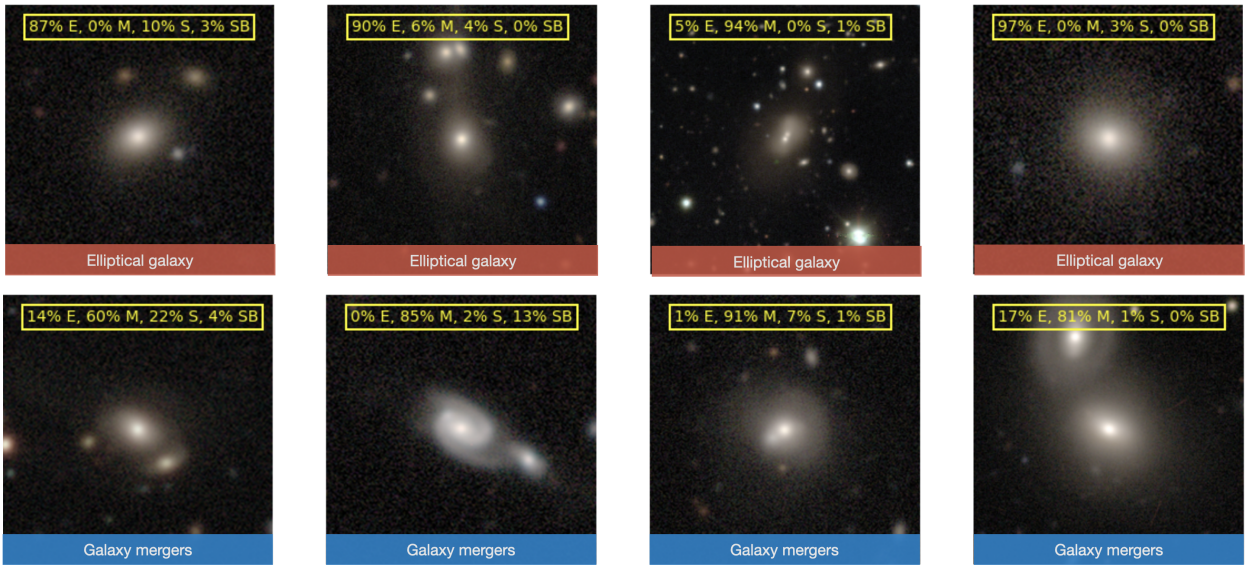


Figure 6.7: Training results for a classifier (Resnet) for galaxies of the Merger and Elliptical types.

The figure 6.7 and 6.14 illustrates the results of applying the classifier to the four classes for which it has been trained: spiral galaxies, elliptical galaxies, galaxy mergers, and barred spiral galaxies. The first row of figure 6.7 illustrates the results obtained in Elliptical galaxies. It is interesting to observe how the classifier obtains outstanding results in the image with clearly defined sources, as others have a more diffuse galaxy shape. It was previously classified as an elliptical galaxy in the catalog, but it has been reclassified as a merger by both the classifier and an expert. In the second row of the 6.7 image, this type of galaxy can be found; however, the classifier is able to recognize the interaction regardless of the distance between the nuclei. Another interesting aspect is what happens in the first image, in which the classifier indicates that it is a merger with 60% accuracy. However, as we can see, the galaxies that are interacting are also correctly classified in the lower percentages, but the interaction is recognized as well. As a result of the backbone objective of highlighting the Regions of Interest for the other parts of the model, the latter becomes relevant. Figure

6.14 illustrates the correct distinction between spiral and barred spiral galaxies. The figure was ordered so that evidence could be obtained as in galaxies of similar structures or definitions, the classifier is able to distinguish spiral galaxies according to the presence of barral structures.

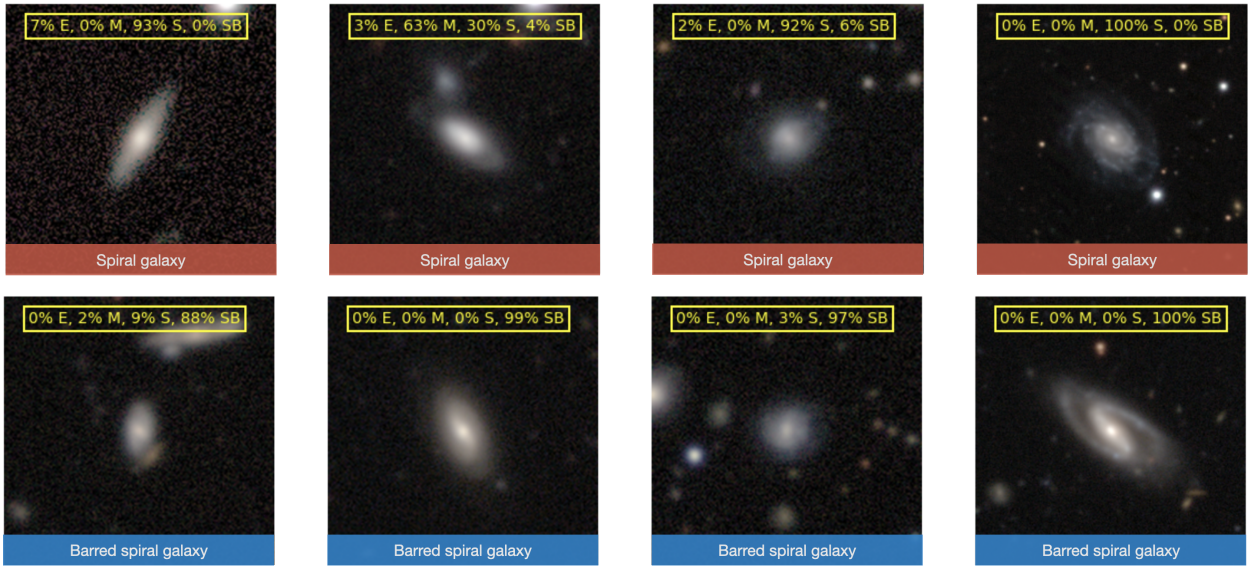


Figure 6.8: Training results for a classifier (Resnet) for galaxies of the Spiral and Barred Spiral.

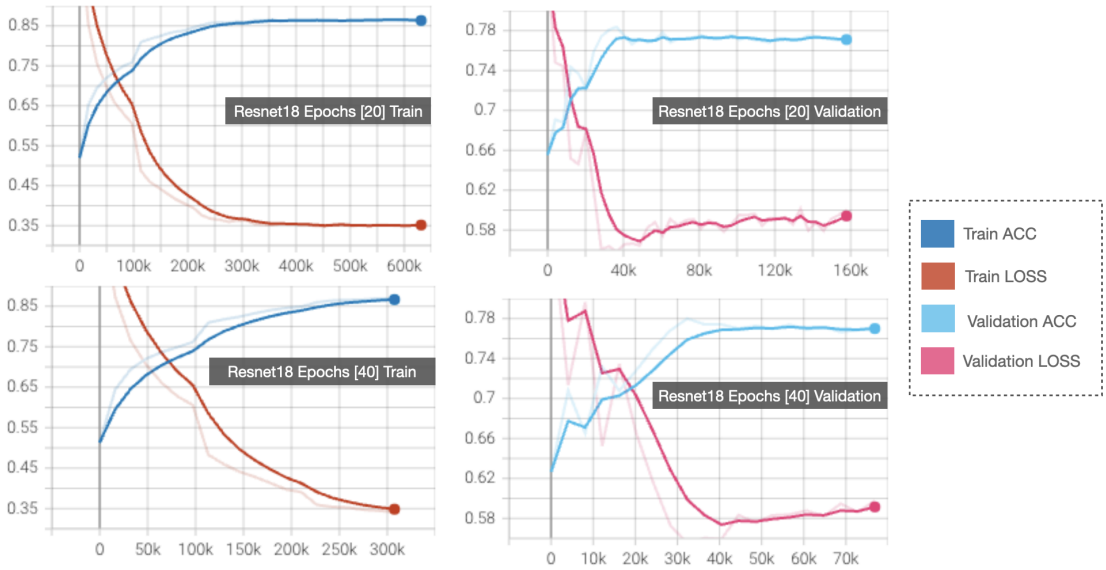


Figure 6.9: A plot illustrating the training metrics (ACC and LOSS) for the Resnet-18 network as well as the validation metrics. There is no evidence of overfitting in the behavior of the model.

6.8 Tensor methods over the backbone

The previous section deals in detail with the model's backbone training process. Table 6.3 indicates that after this first stage, six trained models were available for compression. Chapter 3 discussed compression and acceleration techniques for deep learning models that utilize tensor methods. This section applies these techniques to the six already trained backons. A version of each architecture without training with DECam data would also be included in order to evaluate how significant a backbone trained with the same data is for segmentation training.

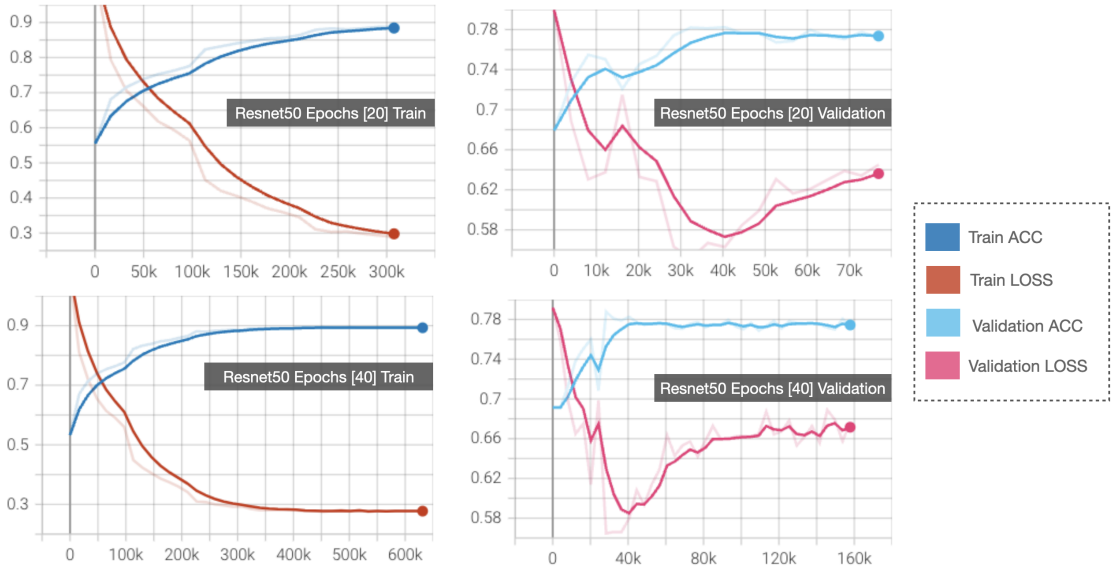


Figure 6.10: A plot illustrating the training metrics (ACC and LOSS) for the Resnet-50 network as well as the validation metrics. There is no evidence of overfitting in the behavior of the model

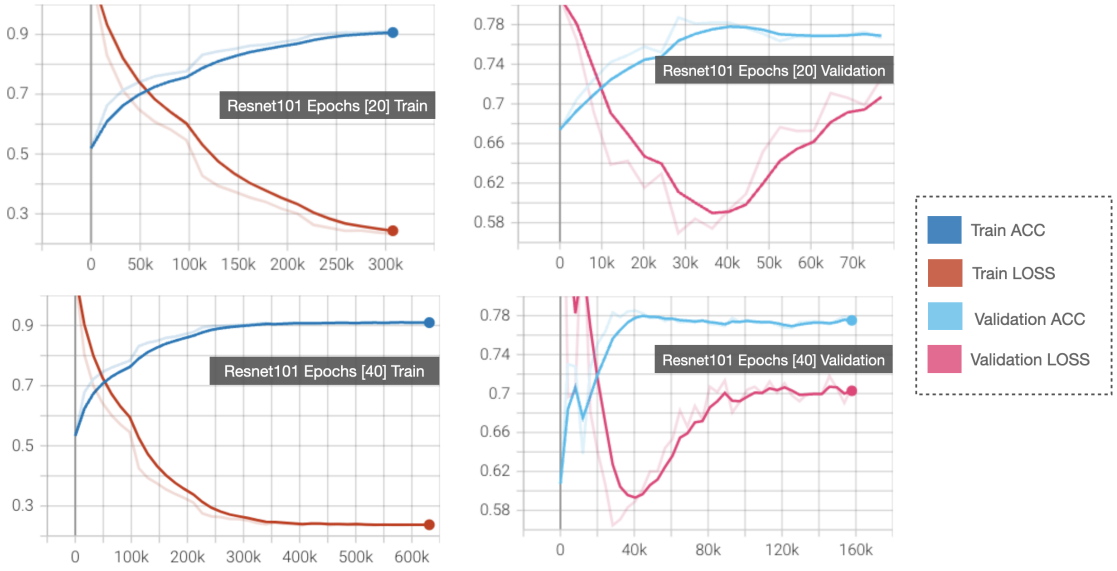


Figure 6.11: A plot illustrating the training metrics (ACC and LOSS) for the Resnet-101 network as well as the validation metrics. There is no evidence of overfitting in the behavior of the model.

Table 6.4: The computational efficiency of a model can be measured by evaluating a set of performance metrics. This table confirms that the technique proposed in this thesis achieves outstanding results in reducing energy consumption, memory usage, and the total number of parameters that must be processed at inference time.

	NORMAL				COMPACT			
Model	TNP(M)	MAdd(G)	FLOPs(G)	MemRW(MB)	TNP(M)	MAdd(G)	FLOPs(G)	MemRW(MB)
Resnet18	11.2	3.64	1.82	95.09	6.3	2.12	1.06	80.37
Resnet50	23.5	8.22	4.12	309.64	14.9	5.54	2.78	287.5
Resnet101	42.9	15.66	7.84	486.22	25.9	9.91	4.97	439.06

The purpose of applying tensor decomposition to the backbone is to increase the computational efficiency of the model, among other factors. However, the model must not fall below a certain level of accuracy that will allow the architecture to meet its scientific objectives. Based on the following metrics, the computational efficiency of the various architectures used as backbone was evaluated:Total number of network parameters (TNP), Theoretical amount of floating point arithmetics (FLOPs), Theoretical amount of multiply-adds (MAdd) and Total MemRead+MemWrite(MemRW).

As a result of these variables, it is possible to consider both the computational efficiency of the model and the number of parameters to be processed. This is in contrast to other works in the area of accelerating neural network architectures, which focus on the number of parameters to be processed as an indicator. We can examine the results of applying the Tucker decomposition method to the same variable in table 6.4, and see that the mean decrease of these values, in the case of Resnet-18, is 44%. This same phenomenon is evident in the Resnet-50 with 37% and in the Resnet-101 with 39%. This indicator alone does not have greater relevance unless it is contrasted with the other metrics that evaluate the performance of the model at hand. Hardware level, as well as metrics that assess model accuracy.

The FLOPS are one of the standard metrics used to assess the computational efficiency of a model or system, which corresponds to the total number of floating point operations necessary for a single forward pass in Deep Laeraning. The Resnet-50 has achieved a remarkable result with this metric, since of the 4.12GFlops that correspond to the operation standard calculated by various benchmarks, 2.78GFlops are achieved. This is significant since it almost achieves the 2.3 GFlops that the SOTA acceleration of a Resnet-50 achieved by Li et al. [2016] using a prunning technique. Resnet-18 experienced the same improvement scenario with 45% fewer operations required, and Resnet-101 experienced a 36% reduction in operations.

For this proposal is especially relevant to the case of the MAdd metric. Tucker method was used to modify the architecture of the network and therefore the type and number of operations carried out in the network by modifying the convolutional fields. Using this metric, we can empirically demonstrate a significant reduction in the number of operations necessary and a consequent increase in model inference speed. The Resnet-18 model was optimized from 3.64GMAdd to 2.12GMAdd, and this optimization has been replicated with Resnet-50, which has a 33% optimization when evaluating the number of operations required. Resnet-101 shows a greater reduction in the number of operations, with 37% fewer of these operations. This also opens up new research opportunities as the number of layers increases, resulting in a greater degree of overparameterization.

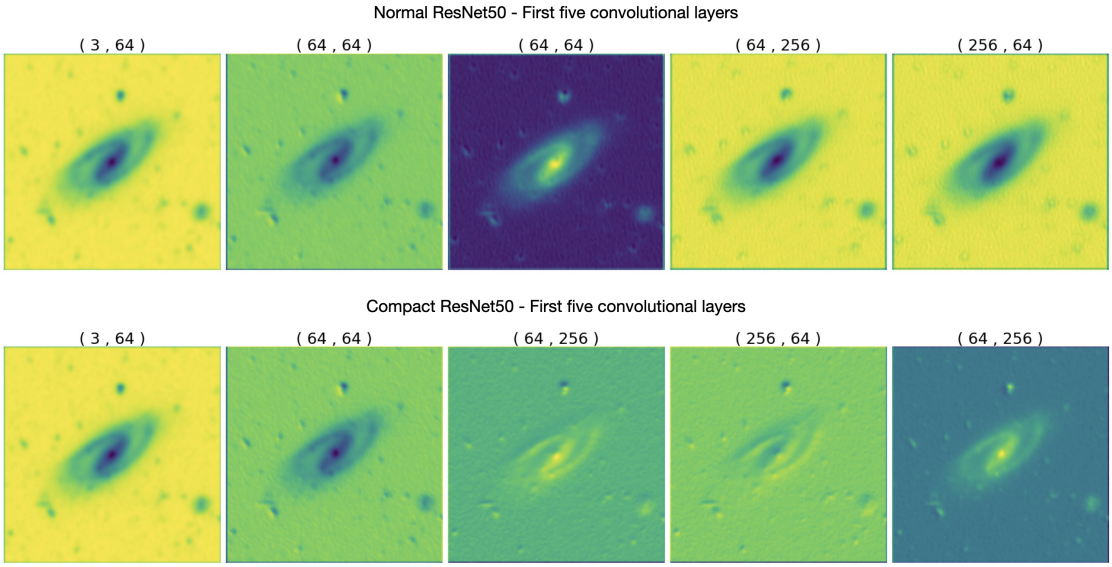


Figure 6.12: Using the approach proposed in this thesis, the feature maps of equivalent layers are compared between a normal Resnet-50 and its compact version. As a result of Tucker's equivalent to PCA, an interesting behavior is evident when considering the future calculation of the RoI. Thus, the resulting feature maps are more defined than their equivalents without decomposition.

A common consideration when implementing applications in the field of big data is the amount of write and read access to memory, which is known as MemRW. In the deep learning community that is seeking to bring these models to production environments, memory footprints are of greater concern. However, this is also relevant to the present proposal, as stated at the beginning of the chapter. As can be seen from table 6.4, the proposal presented in this

thesis allows the memory footprint to be reduced. For example, for the Resnet-50, the memory footprint is reduced to 287.5MB from 309.64MB. In this metric, data is measured not by how fast the model is processed, but rather by the amount of data that is written and read when the model is processed. The latter will depend on the characteristics of the hardware used to run the model.

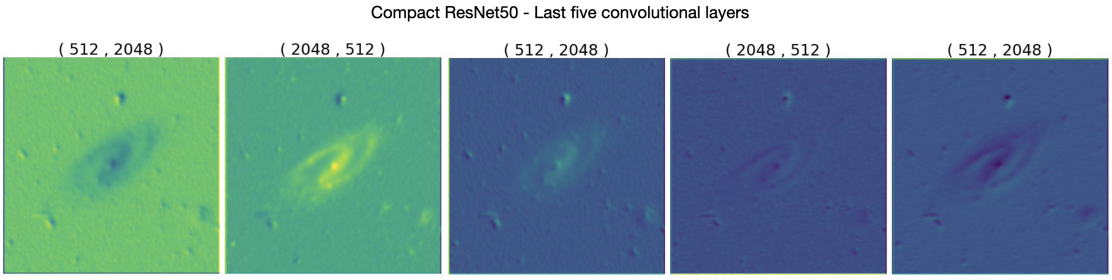


Figure 6.13: The last convolutional layer of a compact version of a Resnet-50 network serves as the backbone of a pipeline.

It is relevant to reaffirm that the training of this stage aims to generate a backbone for the next stage of the pipeline. As mentioned in section 4.1, this backbone is responsible for generating the feature map required for segmenting the data. Based on the method proposed in this paper, figure 6.12 displays feature maps of the first five convolutional layers of Resnet-50 and its compact version. With 49 convolution layers in the original Resnet-50 model, Tucker decomposition has been applied to generate the compact version. This compact version has 33 convolution layers. As can be seen, the filters change after the decompression, but the feature maps do not show any significant differences. Finally, in figure 6.13 we can visualize the five feature maps of the last convolutional layers of the model, where it is evident that despite having 37% fewer parameters and 16 fewer layers, the feature maps allow us to project that the compact versions will provide a solid base. For training the next stage of the pipeline. The validity of this claim will be examined in the following sections.

6.9 Training the Instance Segmentation model

Chapter 4 examined the general characteristics and each component of the Mask r-CNN architecture in the context of its application to astronomical data. The Backbone stands out among these components since it is the main point of optimization according to the method presented in section 5.4. In chapter 4, the main accomplishment is validating the pipeline for galaxy morphological segmentation, achieving results comparable or even superior to the results of human classification. Furthermore, we achieved a 0.93 mAP in addition to other outstanding metrics described in this section. Based on these results and the publication of this work, the foundation has been laid for optimizing this automatic pipeline, with a focus on accelerating its performance.

Table 6.5: An overview of the training results for the instance segmentation model, both in its normal and compact versions. As can be seen from the final implementation, the hypothesis has been validated as a compact version with mAP equal to or greater than the initial work presented in chapter 4 was produced.

Model	Compressed	Epoch	mAP	Training time
Resnet-50	Si	3	0.87	1h 21min
Resnet-50	No	3	0.89	1h 23min
Resnet-50	Si	10	0.90	4h 31min
Resnet-50	No	10	0.92	4h 35min
Resnet-50	Si	20	0.93	9h 7min
Resnet-50	No	20	0.94	9h 18min

Since the backbone is the component with the greatest number of parameters, and where a greater number of operations are performed on these parameters, it plays a key role. Consequently, this proposal incorporates, at the code level, mechanisms for changing the backbone independently of the other components of the architecture. This chapter contains sections 6.6 and 6.8 that deal with training and optimizing this backbone. A diagram of this process is shown in figure 6.5. Upon completion of the training, we have 3 models (resnet-18, resnet-50, and resnet-101) with two versions each (20 and 40 epochs). In section 6.6, you can review the results of this training in more detail. To generate a compact version of the model, the tensor methods proposed in section 5.4 were applied, specifically the Tucker decomposition. The parameters of this model have been significantly reduced. Table 6.4 of section 6.8 summarizes the results of this

compression, showing the degree to which various metrics measuring the computational performance of the models have been optimized. As a result of all these elements, the final phase of building the pipeline involves training a new version of mask r-cnn based on these new backbones. As a result of that, this section will be dedicated to discussing that issue.

The table 6.5 shows the results of training a mask r-cnn with a resnet-50(40 epoch) as the backbone. The training strategy was the same as that described in chapter 4 for the non-optimized pipeline version. This is an aspect that was not initially considered in the proposal, using a backbone trained with data from the same domain as the instance segmentation model. The result of this is a fast convergence of the model, even when simpler architectures are used as the backbone of the model.

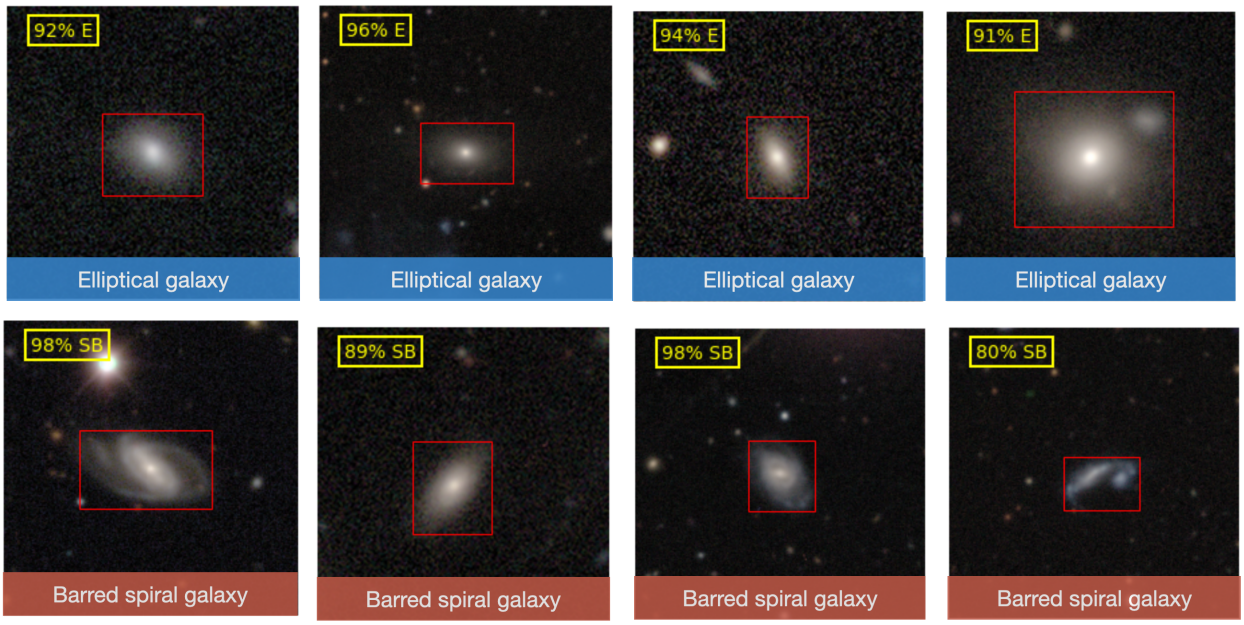


Figure 6.14: Here are examples of results obtained by applying the pipeline to Elliptical and Barred Spiral galaxies. Focusing on visualizing how the ability to locate sources in various scenarios can be visualized.

Analyzing the results of the model's capability for detecting and segmenting objects is also necessary in order to validate an instance segmentation model. Despite the mAP metric's focus on this goal, it is also important to analyze the results visually. Figures 6.14 and 6.15 illustrate the results of the model using a compact version of resnet-50 as a backbone and the representative scenarios for the four pipeline classes. It is clear from figure 6.14 that the model behaves well both when sources are well defined as well as when they are more blurred for elliptical galaxies. However, as in the previous case, it must confuse those with 91% indicating the galaxy is an elliptical galaxy, while including another galaxy in the bounding box at the same time. Considering that this case is the merger of two galaxies, this may be a mistaken initial classification. When used on galaxies of the barred spiral galaxy type, the model provides prominent results in terms of classification, as well as the ability to locate the bounding box correctly. Figure 6.15 shows cases of mergers and spiral galaxies, where the compact model is capable of correctly identifying and locating interacting galaxies based on various scenarios. As indicated in chapter 4, the classification accuracy follows that of a human expert. Finally, in the case of galaxies in Spiral, the model exhibits a similar behavior, correctly classifying galaxies of this type in different orientations. In the compact version of the pipeline, there is also tolerance for images with noise or inconsistency, as in the first example. This was validated in chapter 4 with the systematic integration of Gaussian noise.

6.10 Interpretability of models

Interpretability was included in the analysis of the results obtained by the compact model. It is possible to visually analyze the results of classifiers based on deep learning in several different ways, but for the purposes of this paper, we will categorize them into two categories. Methods based on gradients such as GradCAM, GradCAM++, LayerCAM, among others. Additionally, there are methods referred to as "gradient free": AblationCAM, ScoreCAM, and EigenCAM. The division, as indicated, is based on the type of model for which interpretability is required. GradCAM presents complications in the case of instance segmentation and object location models. This is due to the fact that the process of generating the output of this type of model requires a number of operations and elements, such as the creation of

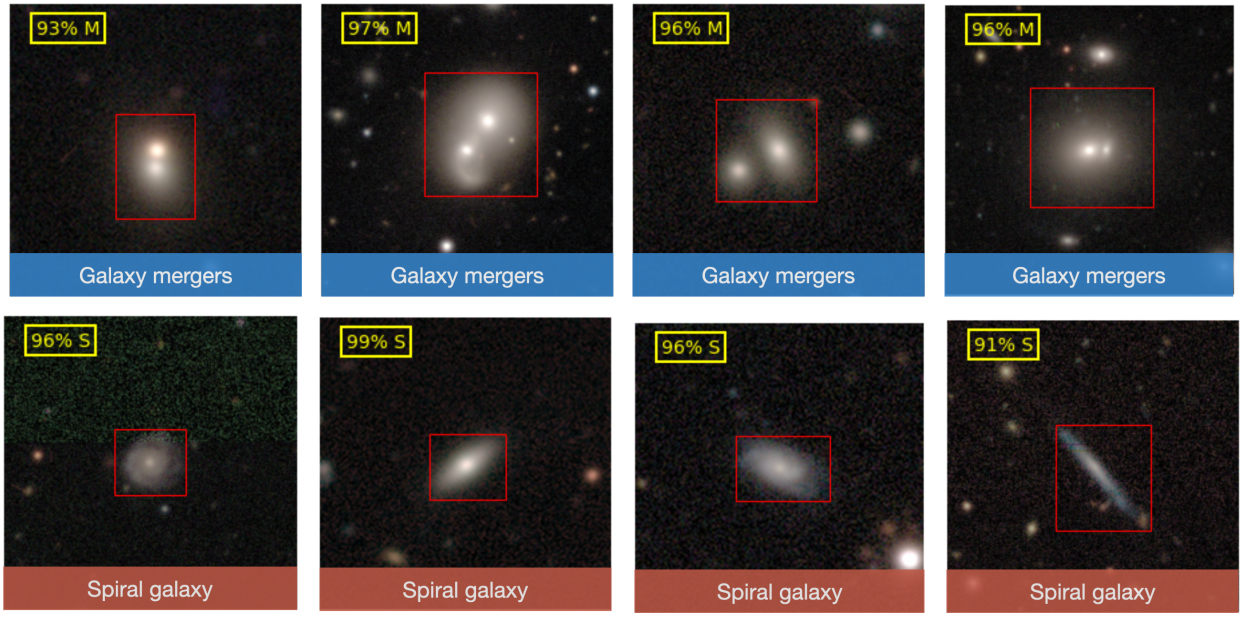


Figure 6.15: According to figure 6.14, the model is able to localize objects as well as classify them correctly. Mergers and spiral galaxies, in this case.

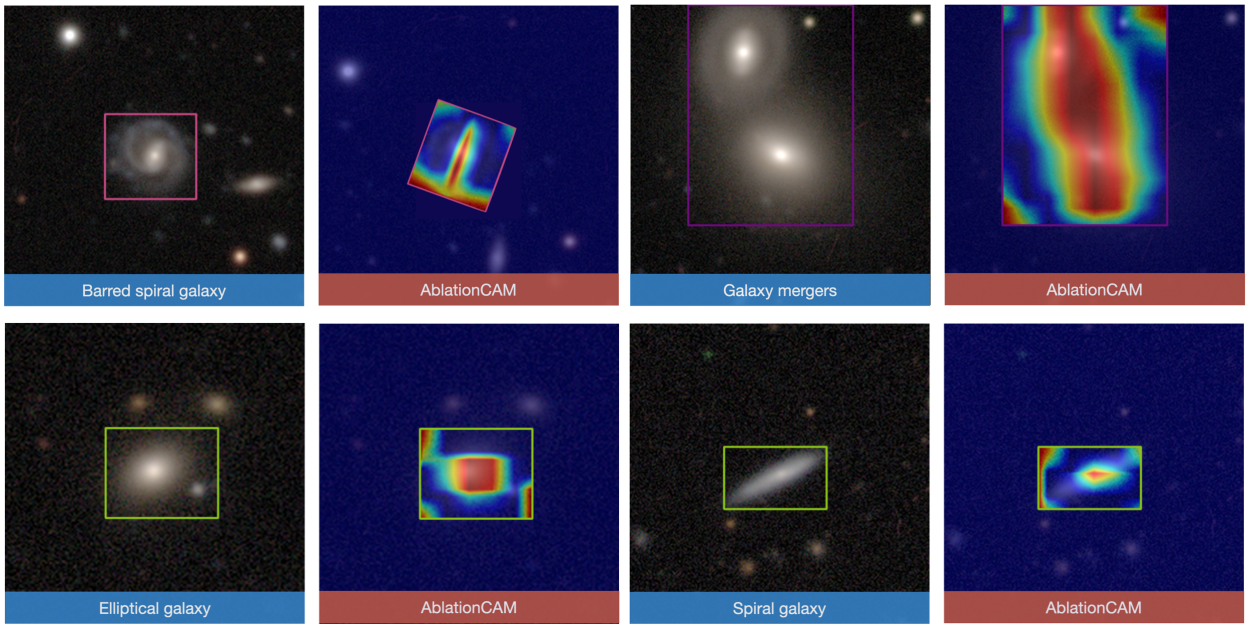


Figure 6.16: The interpretation of results using a method based on AblationCAM, the scientific coherence of the model's generalization, and its qualification and segmentation capabilities are validated.

dictionaries containing bounding boxes, masks, and labels, among others. In order to apply a method based on gradient calculation, we would have to go to the source code level in PyTorch and incorporate it into the low-level inference. This type of model has equally, or even more powerful alternatives, such as AblationCAM, the current state of the art in free gradient modeling. In terms of computational resources, this method has the major disadvantage of being time consuming. A faster alternative to AblationCAM is EigenCAM, which is significantly faster, but has difficulty distinguishing between classes. Consequently, the latter is excluded from the scope of this tesis, since in images such as figure 6.16, in which there are hundreds of thousands of sources, discernibility is of the utmost importance for assessing the accuracy of the compact model.

Figure 6.16 illustrates the four classes and their interpretability maps in one image instead of by band, which makes it easier to understand the reasons for the classification. It can be observed that the compact version of the model perfectly captures or identifies the interaction between galaxies as the key element for classifying them as merger-type galaxies. It is difficult to visualize the element that achieves the correct classification for the simplest galaxies, such as the spiral type, but, as in all cases, the ability to locate and segment the model is evident. It is interesting to examine the case of barred spiral galaxies since, as can be seen, the feature that distinguishes these galaxies corresponds to a bar-shaped structure situated centrally.

6.11 Acceleration of the model

In table 6.6, you can see that the mask r-cnn model, trained on a compact version of the resnet-50 as the backbone, achieved a speedup of 2x in this experiment. This conclusive result validates the premise of the hypothesis that tensor methods can be applied to the backbone of the model, which allows us to obtain faster models as a result of applying tensor methods. As a result of incorporating the backbone training in the same types of data as the inference of the segmentation model, it was possible to reach the same mAP as in the previous work, which is considered the reference for this type of model in astroinformatics.

Table 6.6: Result of applying two Mask R-CNN models to 1000 random images from the validation data set.

Backbone	compressed	epoch	mAP	Inference time
Resnet-50 (40 epochs)	Si	20	0.93	1min 2s
Resnet-50 (40 epochs)	No	20	0.94	2min 14s

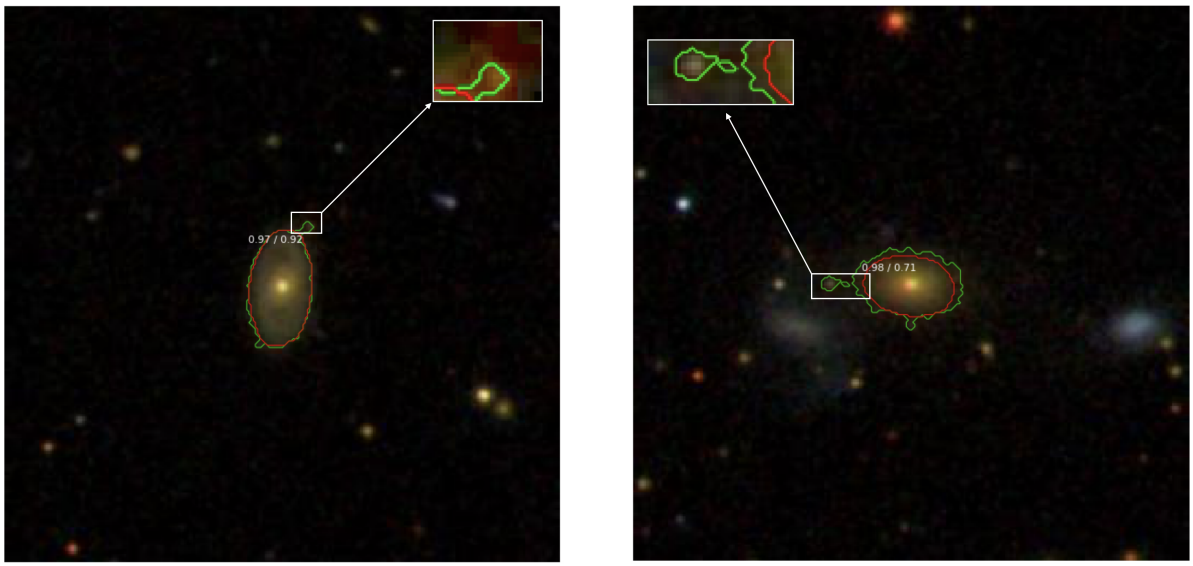


Figure 6.17: Image Segmentation. Both panels show Ground Truth (GT) and detections. The green contour corresponds to the GT, whereas the red contour corresponds to the output made by the network. Numbers in white font display Accuracy/IoU, respectively, measured by the model. Left panel: Example for a S galaxy. Right panel: Example of an E galaxy.

7 Conclusions

The methodology for validating the hypothesis will be divided into three stages. At the data structure level, we were able to model astronomical data that is naturally multidimensional, especially in the z-axis (time, filters, frequencies, etc.) as tensors of order 3. Upon achieving this goal, have the mathematical framework available for generating compact versions of these models or reducing their dimensionality. The objective was fully achieved, and its correctness was validated by peers in the journal Astronomy and Computing. Furthermore, we use these tensor methods to generate compact versions of convolutional networks, specifically Resnet. In spite of this, it can clearly be concluded with the application of Tucker decomposition that a more parsimonious backbone was achieved in terms of parameters, more data-efficient, and more robust to various types of noise, as well as domain shifts. We can see that using a resnet-50 as a backbone exceeds the mAP of the original proposal, which used a resnet-101. However, the most noteworthy result is the compact version of the resnet-50, which has 37% fewer parameters than the original. Using this compact version, the mask r-cnn reaches its original mAP of 0.93. In terms of the acceleration of the model, a test was conducted with 1000 randomly selected images to determine if it had the ability to accelerate.

A second requirement for the hypothesis was the ability to build an automated pipeline for the morphological segmentation of galaxies. In chapter 4, the model that achieved a mAP of 93% was described. Additionally, the results of the pipeline were compared to the classifications made by humans. Considering the size of astronomical data in volume, the automatic pipeline is relevant and necessary for the astroinformatics community. It has been peer-validated at work Farias et al. [2020]. The compact model also reflected this when evaluating its capacity. Figure 6.17 illustrates the powerful results of the compact model by automatically segmenting the shape of galaxies. The green contour corresponds to the mask entered as input to the model, whereas the red contour corresponds to the prediction made by the network. In addition, accuracy and IoU are displayed. The latter corresponds to how much the predicted boundary overlaps with the real object boundary. The left panel of Figure 6.17 shows how the proposed mask, created using the imantics and Scikit-Image, includes an unresolved source that is not part of the spiral galaxy analyzed. On the other hand, using the automatic pipeline proposed, the classification accuracy achieved is 97%. The segmentation generated by the model is superior, it fits almost exactly to the contour and shape of the galaxy, achieving 97% in the object localization indicator. The right panel of Figure 6.17 shows a better case of the quality of the segmentation. The masking process through Scikit-Image generated three isolated masks, but they are considered part of the same galaxy, but the model correctly segments only the analyzed elliptical galaxy.

As a result of validating the hypothesis on all 3 axes and achieving a final speed of 2X, we can conclude that it is correct. The use of tensor methods for generating a compact version of a model's backbone preserves the information required for the model to be energy efficient and to maintain its segmentation capability. Combining tensor methods and complementary techniques, such as pruning, opens up several research avenues. In the case of a library, the interpretability can be visualized by bands correctly, however, both cases fall outside the scope of this thesis.

References

RectLabel an image annotation tool to label images for bounding box object detection and segmentation. https://rectlabel.com/. Accessed: 2020-01-24.

coco-ui front end user interfaces that are used to annotate coco dataset. https://github.com/tylin/coco-ui. Accessed: 2020-01-24.

imantics image semantics. https://github.com/jsbroks/imantics. Accessed: 2020-01-24.

labelbox annotate data to build and ship artificial intelligence applications. https://labelbox.com/. Accessed: 2020-01-24.

Multiplicative Iterative Algorithms for NMF with Sparsity Constraints, chapter 3, pages 131–202. Wiley-Blackwell, 2009. ISBN 9780470747278. doi: 10.1002/9780470747278.ch3. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470747278.ch3.

W. Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. *GitHub repository*, 2017.

J. K. Adelman-McCarthy, M. A. Agüeros, S. S. Allam, C. A. Prieto, K. S. Anderson, S. F. Anderson, J. Annis, N. A. Bahcall, C. Bailer-Jones, I. K. Baldry, et al. The sixth data release of the sloan digital sky survey. *The Astrophysical Journal Supplement Series*, 175(2):297, 2008.

J. Akeret, C. Chang, A. Lucchi, and A. Refregier. Radio frequency interference mitigation using deep convolutional neural networks. *Astronomy and computing*, 18:35–39, 2017.

M. Araya, G. Candia, R. Gregorio, M. Mendoza, and M. Solar. Indexing data cubes for content-based searches in radio astronomy. *Astronomy and Computing*, 14(Supplement C):23 – 34, 2016. ISSN 2213-1337. doi: https://doi.org/10.1016/j.ascom.2016.01.002. URL http://www.sciencedirect.com/science/article/pii/S2213133716000032.

M. Banerji, O. Lahav, C. J. Lintott, F. B. Abdalla, K. Schawinski, S. P. Bamford, D. Andreescu, P. Murray, M. J. Raddick, A. Slosar, et al. Galaxy zoo: reproducing galaxy morphologies via machine learning. *Monthly Notices of the Royal Astronomical Society*, 406(1):342–353, 2010.

R. Banner, Y. Nahshan, and D. Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. *Advances in Neural Information Processing Systems*, 32, 2019.

P. Barchi, R. de Carvalho, R. Rosa, R. Sautter, M. Soares-Santos, B. Marques, E. Clua, T. Gonçalves, C. de Sá-Freitas, and T. Moura. Machine and deep learning applied to galaxy morphology-a comparative study. *Astronomy and Computing*, 30:100334, 2020.

P. H. Barchi, R. R. de Carvalho, R. R. Rosa, R. A. Sautter, M. Soares-Santos, B. A. D. Marques, E. Clua, T. S. Gonçalves, C. de Sá-Freitas, and T. C. Moura. Machine and Deep Learning applied to galaxy morphology - A comparative study. *Astronomy and Computing*, 30:100334, Jan 2020. doi: 10.1016/j.ascom.2019.100334.

D. Baron. Machine learning in astronomy: a practical overview. *arXiv preprint arXiv:1904.07248*, 2019.

R. Bellman. Adaptative control processes, 1961.

G. B. Berriman and S. L. Groom. How will astronomy archives survive the data tsunami? *Commun. ACM*, 54(12): 52–56, Dec. 2011. ISSN 0001-0782. doi: 10.1145/2043174.2043190. URL http://doi.acm.org/10.1145/2043174.2043190.

E. Bertin and S. Arnouts. Sextractor: Software for source extraction. *Astronomy and Astrophysics Supplement Series*, 117(2):393–404, 1996.

A. Boucaud, M. Huertas-Company, C. Heneka, E. E. O. Ishida, N. Sedaghat, R. S. de Souza, B. Moews, H. Dole, M. Castellano, E. Merlin, V. Roscani, A. Tramacere, M. Killedar, and A. M. M. Trindade. Photometry of high-redshift blended galaxies using deep learning. *Monthly Notices of the Royal Astronomical Society*, 12 2019. ISSN 0035-8711. doi: 10.1093/mnras/stz3056. URL https://doi.org/10.1093/mnras/stz3056. stz3056.

P. C. Broekema, V. Allan, R. V. van Nieuwpoort, and H. E. Bal. On optimising cost and value in compute systems for radio astronomy. *Astronomy and Computing*, 30:100337, 2020.

A. G. Brown, A. Vallenari, T. Prusti, J. De Bruijne, F. Mignard, R. Drimmel, C. Babusiaux, C. Bailer-Jones, U. Bastian, M. Biermann, et al. Gaia data release 1-summary of the astrometric, photometric, and survey properties. *Astronomy & Astrophysics*, 595:A2, 2016.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

C. M. Brunt and M. H. Heyer. Principal component analysis of spectral line data: analytic formulation. *Monthly Notices of the Royal Astronomical Society*, 433(1):117–126, 2013. doi: 10.1093/mnras/stt707. URL http://dx.doi.org/10.1093/mnras/stt707.

C. J. Conselice. The relationship between stellar light distributions of galaxies and their formation histories. *The Astrophysical Journal Supplement Series*, 147(1):1, 2003.

M. Cornea. Intel avx-512 instructions and their use in the implementation of math functions. *Intel Corporation*, pages 1–20, 2015.

S. Daghaghi, N. Meisburger, M. Zhao, and A. Shrivastava. Accelerating slide deep learning on modern cpus: Vectorization, quantizations, memory optimizations, and more. *Proceedings of Machine Learning and Systems*, 3:156–166, 2021.

L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000a.

L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000b.

S. Dieleman, K. W. Willett, and J. Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly notices of the royal astronomical society*, 450(2):1441–1459, 2015.

A. Dutta and A. Zisserman. The vgg image annotator (via). *arXiv preprint arXiv:1904.10699*, 2019.

T. Elsken, J. H. Metzen, and F. Hutter. Neural architecture search: A survey. *The Journal of Machine Learning Research*, 20(1):1997–2017, 2019.

H. Farias, D. Ortiz, G. Damke, M. J. Arancibia, and M. Solar. Mask galaxy: Morphological segmentation of galaxies. *Astronomy and Computing*, 33:100420, 2020.

J. P. Gardner, J. C. Mather, M. Clampin, R. Doyon, M. A. Greenhouse, H. B. Hammel, J. B. Hutchings, P. Jakobsen, S. J. Lilly, K. S. Long, et al. The james webb space telescope. *Space Science Reviews*, 123(4):485–606, 2006.

R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

R. E. González, R. P. Muñoz, and C. A. Hernández. Astrocv: Astronomy computer vision library. *Astrophysics Source Code Library*, 2018.

I. Goodfellow, Y. Bengio, and A. Courville. Deep learning (adaptive computation and machine learning series). *Cambridge Massachusetts*, pages 321–359, 2017.

N. Goyal, S. Vempala, and Y. Xiao. Fourier PCA and Robust Tensor Decomposition. *ArXiv e-prints*, June 2013.

R. A. Harshman. Foundations of the parafac procedure: models and conditions for an" explanatory" multimodal factor analysis. 1970.

A. H. Hassan, C. J. Fluke, and D. G. Barnes. Interactive visualization of the largest radioastronomy cubes. *New Astronomy*, 16(2):100–109, 2011.

A. H. Hassan, C. J. Fluke, D. G. Barnes, and V. Kilborn. Tera-scale astronomical data analysis and visualization. *Monthly Notices of the Royal Astronomical Society*, 429(3):2442–2455, 2013.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

Y. He, G. Kang, X. Dong, Y. Fu, and Y. Yang. Soft filter pruning for accelerating deep convolutional neural networks. *arXiv preprint arXiv:1808.06866*, 2018.

C. J. Hillar and L.-H. Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):1–39, 2013.

T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden, and A. Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. *The Journal of Machine Learning Research*, 22(1):10882–11005, 2021.

E. Hubble. No. 324. Extra-galactic nebulae. *Contributions from the Mount Wilson Observatory / Carnegie Institution of Washington*, 324:1–49, Jan 1926.

G.-B. D. L. Inference and B. D. Learning. A performance and power analysis. *NVidia Whitepaper, Nov*, 2015.

Z. Ivezic, T. Axelrod, W. Brandt, D. Burke, C. Claver, A. Connolly, K. Cook, P. Gee, D. Gilmore, S. Jacoby, et al. Large synoptic survey telescope: From science drivers to reference design. Technical report, SLAC National Accelerator Lab., Menlo Park, CA (United States), 2011.

Ž. Ivezić, A. Connolly, J. Vanderplas, and A. Gray. *Statistics, Data Mining and Machine Learning in Astronomy*. Princeton University Press, 2014.

T. Jenness and F. Economou. Orac-dr: A generic data reduction pipeline infrastructure. *Astronomy and Computing*, 9:40 – 48, 2015. ISSN 2213-1337. doi: https://doi.org/10.1016/j.ascom.2014.10.005. URL http://www.sciencedirect.com/science/article/pii/S2213133714000614.

A. B. Jung, K. Wada, J. Crall, S. Tanaka, J. Graving, S. Yadav, J. Banerjee, G. Vecsei, A. Kraft, J. Borovec, C. Vallentin, S. Zhydenko, K. Pfeiffer, B. Cook, I. Fernández, W. Chi-Hung, A. Ayala-Acevedo, R. Meudec, M. Laporte, et al. imgaug. https://github.com/aleju/imgaug, 2019. Online; accessed 25-Sept-2019.

Y. D. Kim and S. Choi. Nonnegative tucker decomposition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007. doi: 10.1109/CVPR.2007.383405.

Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv preprint arXiv:1511.06530*, 2015.

T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, Aug. 2009. ISSN 0036-1445. doi: 10.1137/07070111X. URL http://dx.doi.org/10.1137/07070111X.

J. Kossaifi, Y. Panagakis, and M. Pantic. Tensorly: Tensor learning in python. *arXiv preprint arXiv:1610.09555*, 2016.

J. Kremer, K. Stensbo-Smidt, F. Gieseke, K. S. Pedersen, and C. Igel. Big universe, big data: Machine learning and image analysis for astronomy. *IEEE Intelligent Systems*, 32(2):16–22, Mar.-Apr. 2017. ISSN 1541-1672. doi: 10.1109/MIS.2017.40. URL doi.ieeecomputersociety.org/10.1109/MIS.2017.40.

R. Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

A. Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

D. R. Law, B. Cherinka, R. Yan, B. H. Andrews, M. A. Bershady, D. Bizyaev, G. A. Blanc, M. R. Blanton, A. S. Bolton, J. R. Brownstein, et al. The data reduction pipeline for the sdss-iv manga ifu galaxy survey. *The Astronomical Journal*, 152(4):83, 2016.

V. Lebedev and V. Lempitsky. Speeding-up convolutional neural networks: A survey. *Bulletin of the Polish Academy of Sciences. Technical Sciences*, 66(6):799–811, 2018.

H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016.

Z. Li, Y. Wang, T. Zhi, and T. Chen. A survey of neural network accelerators. *Frontiers of Computer Science*, 11: 746–761, 2017.

T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

C. J. Lintott, K. Schawinski, A. Slosar, K. Land, S. Bamford, D. Thomas, M. J. Raddick, R. C. Nichol, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. , 389(3):1179–1189, Sept. 2008. doi: 10.1111/j.1365-2966.2008.13689.x.

R. Lupton, M. R. Blanton, G. Fekete, D. W. Hogg, W. O'Mullane, A. Szalay, and N. Wherry. Preparing red-green-blue images from ccd data. *Publications of the Astronomical Society of the Pacific*, 116(816):133, 2004.

L. Ma, Z. Chen, L. Xu, and Y. Yan. Multimodal deep learning for solar radio burst classification. *Pattern Recognition*, 61:573–582, 2017.

M. Macktoobian, D. Gillet, and J.-P. Kneib. Optimal target assignment for massive spectroscopic surveys. *Astronomy and Computing*, 30:100364, 2020. ISSN 2213-1337. doi: https://doi.org/10.1016/j.ascom.2020.100364. URL http://www.sciencedirect.com/science/article/pii/S2213133719301465.

P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

A. Mikołajczyk and M. Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, May 2018. doi: 10.1109/IIPHDW.2018.8388338.

S. Murray. Pca for radio astronomy data cubes, 05 2015.

S. Nakajima, M. Sugiyama, S. D. Babacan, and R. Tomioka. Global analytic solution of fully-observed variational bayesian matrix factorization. *The Journal of Machine Learning Research*, 14(1):1–37, 2013.

M. A. Nieto-Santisteban, A. S. Szalay, and J. Gray. *ImgCutout, an Engine of Instantaneous Astronomical Discovery*, volume 314 of *Astronomical Society of the Pacific Conference Series*, page 666. 2004.

A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov. Tensorizing neural networks. In *Advances in neural information processing systems*, pages 442–450, 2015.

S. C. Odewahn, E. Stockwell, R. Pennington, R. M. Humphreys, and W. Zumach. Automated star/galaxy discrimination with neural networks. In *Digitised Optical Sky Surveys*, pages 215–224. Springer, 1992.

Y. Panagakis, J. Kossaifi, G. G. Chrysos, J. Oldfield, M. A. Nicolaou, A. Anandkumar, and S. Zafeiriou. Tensor methods in computer vision and deep learning. *Proceedings of the IEEE*, 109(5):863–890, 2021.

K. L. Polsterer, F. Gieseke, and C. Igel. Automatic galaxy classification via machine learning techniques: Parallelized rotation/flipping invariant kohonen maps (pink). In *Astronomical Data Analysis Software an Systems XXIV (ADASS XXIV)*, volume 495, page 81. Citeseer, 2015.

S. Rabanser, O. Shchur, and S. Günnemann. Introduction to Tensor Decompositions and their Applications in Machine Learning. *ArXiv e-prints*, Nov. 2017.

Z. Rahimi and M. M. Homayounpour. Tens-embedding: a tensor-based document embedding method. *Expert Systems with Applications*, 162:113770, 2020.

Z. Rahimi and M. M. Homayounpour. Tenssent: a tensor based sentimental word embedding method. *Applied Intelligence*, 51(8):6056–6071, 2021.

J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

R. Reed. Pruning algorithms-a survey. *IEEE transactions on Neural Networks*, 4(5):740–747, 1993.

S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

A. Renda, J. Frankle, and M. Carbin. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389*, 2020.

T. P. Robitaille, E. J. Tollerud, P. Greenfield, M. Droettboom, E. Bray, T. Aldcroft, M. Davis, A. Ginsburg, A. M. Price-Whelan, W. E. Kerzendorf, et al. Astropy: A community python package for astronomy. *Astronomy & Astrophysics*, 558:A33, 2013.

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.

F. Ruffy and K. Chahal. The state of knowledge distillation for classification. *arXiv preprint arXiv:1912.10850*, 2019.

D. Sandler, T. Barrett, D. Palmer, R. Fugate, and W. Wild. Use of a neural network to control an adaptive optics system for an astronomical telescope. *Nature*, 351(6324):300–302, 1991.

K. A. Sankararaman, S. De, Z. Xu, W. R. Huang, and T. Goldstein. The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In *International conference on machine learning*, pages 8469–8479. PMLR, 2020.

Z. Shen and E. Xing. A fast knowledge distillation framework for visual recognition. In *European Conference on Computer Vision*, pages 673–690. Springer, 2022.

D. Spergel, N. Gehrels, C. Baltay, D. Bennett, J. Breckinridge, M. Donahue, A. Dressler, B. Gaudi, T. Greene, O. Guyon, et al. Wide-field infrarred survey telescope-astrophysics focused telescope assets wfirst-afta 2015 report. *arXiv preprint arXiv:1503.03757*, 2015.

A. S. Szalay, J. Gray, A. R. Thakar, P. Z. Kunszt, T. Malik, J. Raddick, C. Stoughton, and J. vandenBerg. The SDSS SkyServer: Public Access to the Sloan Digital Sky Server Data. *arXiv e-prints*, art. cs/0202013, Feb 2002.

V. Sze, Y. Chen, T. Yang, and J. Emer. Efficient processing of deep neural networks: A tutorial and survey. arxiv 2017. *arXiv preprint arXiv:1703.09039*, 2017.

V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer. Efficient processing of deep neural networks. *Synthesis Lectures on Computer Architecture*, 15(2):1–341, 2020.

M. Tan, R. Pang, and Q. V. Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

A. Torralba, B. C. Russell, and J. Yuen. Labelme: Online image annotation and applications. *Proceedings of the IEEE*, 98(8):1467–1484, 2010.

L. R. Tucker. The extension of factor analysis to three-dimensional matrices. *Contributions to mathematical psychology*, 110119, 1964.

M. Walmsley, C. Lintott, T. Géron, S. Kruk, C. Krawczyk, K. W. Willett, S. Bamford, L. S. Kelvin, L. Fortson, Y. Gal, W. Keel, K. L. Masters, V. Mehta, B. D. Simmons, R. Smethurst, L. Smith, E. M. Baeten, and C. Macmillan. Galaxy Zoo DECaLS: Detailed visual morphology measurements from volunteers and deep learning for 314 000 galaxies. , 509(3):3966–3988, Jan. 2022. doi: 10.1093/mnras/stab2093.

Z. Wang, T. Luo, R. S. M. Goh, W. Zhang, and W.-F. Wong. Optimizing for in-memory deep learning with emerging memory technology. *arXiv preprint arXiv:2112.00324*, 2021.

K. W. Willett, C. J. Lintott, S. P. Bamford, K. L. Masters, B. D. Simmons, K. R. Casteels, E. M. Edmondson, L. F. Fortson, S. Kaviraj, W. C. Keel, et al. Galaxy zoo 2: detailed morphological classifications for 304 122 galaxies from the sloan digital sky survey. *Monthly Notices of the Royal Astronomical Society*, 435(4):2835–2860, 2013.

S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

L. Xuelei, D. Liangkui, W. Li, and C. Fang. Fpga accelerates deep residual learning for image recognition. 2017.

D. G. York, J. Adelman, J. Anderson, John E., S. F. Anderson, J. Annis, N. A. Bahcall, J. A. Bakken, R. Barkhouser, S. Bastian, E. Berman, W. N. Boroski, S. Bracker, C. Briegel, J. W. Briggs, J. Brinkmann, R. Brunner, S. Burles, L. Carey, M. A. Carr, F. J. Castander, B. Chen, P. L. Colestock, A. J. Connolly, J. H. Crocker, I. Csabai, P. C. Czarapata, J. E. Davis, M. Doi, T. Dombeck, D. Eisenstein, N. Ellman, B. R. Elms, M. L. Evans, X. Fan, G. R. Federwitz, L. Fiscelli, S. Friedman, J. A. Frieman, M. Fukugita, B. Gillespie, J. E. Gunn, V. K. Gurbani, E. de Haas, M. Haldeman, F. H. Harris, J. Hayes, T. M. Heckman, G. S. Hennessy, R. B. Hindsley, S. Holm, D. J. Holmgren, C.-h. Huang, C. Hull, D. Husby, S.-I. Ichikawa, T. Ichikawa, Ž. Ivezić, S. Kent, R. S. J. Kim, E. Kinney, M. Klaene, A. N. Kleinman, S. Kleinman, G. R. Knapp, J. Korienek, R. G. Kron, P. Z. Kunszt, D. Q. Lamb, B. Lee, R. F. Leger, S. Limmongkol, C. Lindenmeyer, D. C. Long, C. Loomis, J. Loveday, R. Lucinio, R. H. Lupton, B. MacKinnon, E. J. Mannery, P. M. Mantsch, B. Margon, P. McGehee, T. A. McKay, A. Meiksin, A. Merelli, D. G. Monet, J. A. Munn, V. K. Narayanan, T. Nash, E. Neilsen, R. Neswold, H. J. Newberg, R. C. Nichol, T. Nicinski, M. Nonino, N. Okada, S. Okamura, J. P. Ostriker, R. Owen, A. G. Pauls, J. Peoples, R. L. Peterson, D. Petravick, J. R. Pier, A. Pope, R. Pordes, A. Prosapio, R. Rechenmacher, T. R. Quinn, G. T. Richards, M. W. Richmond, C. H. Rivetta, C. M. Rockosi, K. Ruthmansdorfer, D. Sandford, D. J. Schlegel, D. P. Schneider, M. Sekiguchi, G. Sergey, K. Shimasaku, W. A. Siegmund, S. Smee, J. A. Smith, S. Snedden, R. Stone, C. Stoughton, M. A. Strauss, C. Stubbs, M. SubbaRao, A. S. Szalay, I. Szapudi, G. P. Szokoly, A. R. Thakar, C. Tremonti, D. L. Tucker, A. Uomoto, D. Vanden Berk, M. S. Vogeley, P. Waddell, S.-i. Wang, M. Watanabe, D. H. Weinberg, B. Yanny, N. Yasuda, and SDSS Collaboration. The Sloan Digital Sky Survey: Technical Summary. , 120(3):1579–1587, Sept. 2000. doi: 10.1086/301513.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

H. Zhou, L. Li, and H. Zhu. Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association*, 108(502):540–552, 2013. ISSN 0162-1459. doi: 10.1080/01621459.2013.776499.

Z.-H. Zhou. Why over-parameterization of deep neural networks does not overfit? *Science China Information Sciences*, 64(1):1–3, 2021.

W. Zhuang, T. Hascoet, X. Chen, R. Takashima, T. Takiguchi, Y. Ariki, et al. Convolutional neural networks inference memory optimization with receptive field-based input tiling. *APSIPA Transactions on Signal and Information Processing*, 12(1), 2021.