A self-supervised learning approach for astronomical images

Ana Carolina Martinazzo

Dissertation text submitted to
the
Institute of Mathematics and Statistics
of the
University of São Paulo
in order to
obtain the degree of
Master of Science

Master Program: Computer Science

Advisor: Prof. Dr. Nina Hirata

Author has received financial support from FAPESP

São Paulo, 2021

A self-supervised learning approach for astronomical images

Esta versão da dissertação contém as correções e alterações sugeridas pela Comissão Julgadora durante a defesa da versão original do trabalho, realizada em 25/10/2021. Uma cópia da versão original está disponível no Instituto de Matemática e Estatística da Universidade de São Paulo.

Comissão Julgadora:

Profª. Drª. Nina Sumiko Tomita Hirata (IME-USP)

Prof. Dr. Laerte Sodre Junior (IAG-USP)

Prof. Dr. Jurandy Gomes de Almeida Junior (UNIFESP)

Acknowledgments

First of all, I would like to thank Nina for being available from the first time we met, for being such an inspiring advisor and researcher, and also for being understanding about me going back to work in industry. I am very happy that I had the chance to meet you and work with you during these years.

I would also like to thank my family, specially Patricia and Jimi, for bearing with me from the moment I decided to pursue graduate studies until the moment I finally obtained the degree of Master of Science. I know it has been a long journey and am forever grateful for your support.

Besides, I would like to thank colleagues from the Institute of Astronomy, Geophysics and Atmospheric Sciences (IAG), specially Claudia and Lilianne, who introduced me to many S-PLUS contributors and were always available to help me understand astronomical concepts.

Finally, I would like to thank colleagues from our Image Understanding research group, specially Mateus, with whom I had the opportunity to work in a couple of fun projects which turned out to be published. I already miss our group meetings where we would discuss all sorts of technical and non-technical stuff while sipping our coffees.

Last but not least, I would like to thank São Paulo Research Foundation (FAPESP), processes 2017/25835-9, 2018/25671-9, for financially supporting this research work.

Abstract

MARTINAZZO, A. C. **A self-supervised learning approach for astronomical images**. 2021. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2021.

Modern astronomical sky surveys are providing us with a flood of images with unusual characteristics, such as numerous channels, saturated signals, faint signals, uncertainties, and varying signal-to-noise ratios. The complexity and diversity of these images make them an adequate use case for deep convolutional neural networks. Moreover, they yield millions of detected objects whose classes are mostly unknown. Given this context, the main objective of this work is to investigate deep representation learning approaches for multichannel astronomical images, focusing on finding reasonable representations that do not require labeled data and that make use of some domain knowledge. A reasonable representation may be thought of as one that contains enough discriminative information, that can be later used for higher-level tasks such as object classification, outlier detection and clustering. We propose a self-supervised learning approach that makes use of astronomical properties (more specifically, magnitudes) of the objects in order to pretrain deep neural networks with unlabeled data. We choose the task of classifying galaxies, stars and quasars as a baseline for quantifying the quality of the learned representations, and empirically demonstrate that our approach yields results that are better than – or at least comparable to – a benchmark RGB model pretrained on ImageNet.

Keywords: convolutional neural networks, astronomical image processing, self-supervised learning.

Resumo

MARTINAZZO, A. C. **Uma abordagem de aprendizagem auto-supervisionada para imagens astronômicas**. 2021. Dissertação (Mestrado) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2021.

Programas de varredura do céu contemporâneos têm nos fornecido um grande volume de imagens com características pouco usuais, tais como grande quantidade de canais, sinais saturados, sinais fracos, incertezas, e razões sinal-ruído variáveis. A complexidade e diversidade dessas imagens faz com que elas sejam dados bastante adequados e interessantes para uso de redes neurais convolucionais profundas. Dado este contexto, o principal objetivo deste trabalho é investigar abordagens de aprendizagem de representações para imagens astronômicas usando redes neurais, com foco em encontrar representações satisfatórias que não necessitem de dados rotulados, e que incorporem um pouco de conhecimento específico da Astronomia. Uma representação satisfatória pode ser definida como uma representação que contenha informação discriminativa suficiente para que possa ser utilizada em tarefas de mais alto nível, tais como classificação de objetos, detecção de anomalias, e agrupamento. É proposta uma abordagem de aprendizagem auto-supervisionada que utiliza propriedades astronômicas (mais especificamente, magnitudes) dos objetos a fim de possibilitar o pré-treinamento de redes neurais profundas com dados não rotulados. A tarefa de classificar galáxias, estrelas e quasares é escolhida como uma base comparativa para quantificar a qualidade das representações aprendidas. Demonstramos empiricamente que nossa abordagem produz resultados que são melhores do que – ou comparáveis a – um modelo de referência pré-treinado no ImageNet.

Keywords: redes neurais convolucionais, processamento de imagens astronômicas, aprendizagem auto-supervisionada.

Publications

The following articles have been published over the course of this research project:

- A. Martinazzo and N. S. T Hirata. Multiband image classification of astronomical objects. Workshop of Works in Progress at the Conference on Graphics, Patterns and Images, 2019. [MH19]
- M. Espadoto, A. Martinazzo and N. S. T Hirata. Deep Learning for Astronomical Object Classification: A Case Study. International Conference on Computer Vision Theory and Applications, 2020. [EMH20]
- A. Martinazzo, M. Espadoto and N. S. T. Hirata. Self-supervised Learning for Astronomical Image Classification. International Conference on Pattern Recognition, 2020. [MEH20]

Contents

List of Abbreviations

AI	Artificial Intelligence
CNN	Convolutional Neural Network
DL	Deep Learning
DR	Data Release
FITS	Flexible Image Transport System
FWHM	Full Width at Half Maximum
GPU	Graphical Processing Unit
ML	Machine Learning
S-PLUS	Southen Photometric Local Universe Survey
SDSS	Sloan Digital Sky Survey
WISE	Wide-field Infrared Survey

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Unlike other natural sciences, such as Physics and Biology, in which various kinds of experiments can be designed to validate theories, Astronomy relies almost entirely on images. Each photon captured by the mirrors of a telescope encodes information about its time of arrival, its spatial position and its energy content. Those three pieces of information should provide us with all there is that we can learn about the universe.

The development of Observational Astronomy as a modern science at first naturally followed advances in photography. The first documented attempt at taking a picture of an object at the sky (the Moon) happened in the late 1830s [Nor38]. Later on, astronomers transitioned to using telescopes, which become larger and capable of capturing more photons at better spatial and temporal resolutions as technology evolves. This results in increasingly large amounts of images to process and analyze. Even nowadays, it is common practice for experts to pick a few dozens or hundreds of objects and then analyze their images by eye with the aid of specific image processing tools. However, with the newest generation of telescopes starting to scan our sky at rates that can reach the order of terabytes per night [Tys02], this kind of manual work is becoming unfeasible.

Much like Observational Astronomy, Computer Vision is also transitioning from rather small-scale, task-specific to large-scale, generalist techniques. Deep neural networks automatically learn high-level information without the need for employing manually designed feature descriptors. Because deep learning models are often much larger and more complex than traditional models, they also require a lot more data. In our use case, this should not be a problem, since last-generation sky surveys are already providing us with a flood of rich, diverse data.

Astronomical images present a set of unique characteristics that make them attractive for trying out deep learning methods: numerous channels, saturated signals, faint signals, uncertainties, and signal-to-noise ratios that vary with the filters used and the atmospheric conditions. Data also become noisier and more heterogeneous as astronomical surveys go

deeper, capturing signals from fainter, farther away sources.

Some researchers believe that the next step towards more data-driven science would be to detect patterns in tabular data extracted from images by fitting mathematical models to point sources. Indeed, many recent works that propose data-driven methods use tabular data for tasks such as classification [CDSM+19]. Another possible approach is to apply machine learning methods to spectroscopic data, which can be roughly interpreted as a count of captured photons as a function of wavelength, as proposed in [LZD+18]. A major drawback of both tabular and spectroscopic data is that neither of them provide detailed morphological information.

We believe that it is possible to learn more if we let deep learning models process the data in a more raw format – that is, in images. Raw astronomical images come in multiple channels, whereas digital images are usually represented using the RGB color model, which consists of only three channels. In the RGB color model, red, green and blue light are mixed to reproduce a variety of colors. This choice of primary colors is related to the physiology of the human eye. The human eye contains three different kinds of photoreceptor cells, called cones, each responding differently to light of different wavelengths. The RGB model aims to maximize the difference between the responses of the cone cells, which yields a broader range of possible colors to be perceived by human vision.

It makes perfect sense to try and convert raw astronomical images, which usually come in FITS file format, into informative, aesthetically pleasing color images that can be interpreted by human vision. Indeed, composite RGB images can be generated and are largely used for scientific communication in Astronomy. However, it is not clear whether this sort of pre-processing actually helps machine vision, or not. In order to acquire some understanding of how deep learning models deal with raw FITS images as opposed to RGB images generated with specific domain knowledge, we conduct a comparative study between models trained with either type of image. Recent works have proposed models using either RGB images [GMH18] or FITS images [KB16]. However, to the best of our knowledge, a comparative analysis of RGB and FITS models is novel.

There is yet another challenge that we expect to tackle in this work. Large sky surveys yield millions – sometimes billions – of detected objects. A tiny minority of those detected objects have already been observed by other more detailed surveys, such as spectroscopic surveys, and then thoroughly studied by experts. Only the classes of those objects are truly known. This challenge is even more prominent when studying the Southern sky, because detailed surveys here in the Southern hemisphere are more scarce. Highly qualified domain experts still spend a significant amount of time looking at tons of images of objects and handpicking the most relevant ones for further analysis. It is desireable to develop reliable ways to automatically classify all objects from a mostly unlabeled dataset, or to automatically find the most relevant ones.

Drawing on self-supervised learning, we devise a feature extraction approach that, by exploiting known quantitative attributes of the astronomical objects, is able to make use of

both labeled and unlabeled data. We conjecture that this approach is capable of generating encodings that encompass both morphological information from images and astrophysical information from quantitative attributes, making these encodings more powerful than either images or numerical attributes alone. To the best of our knowledge, combining astronomical images of objects and their quantitative attributes into a single representation is also a novel proposal.

1.2 Objectives

The main proposal of this work is to find reasonable representations for astronomical images, learned by deep convolutional neural networks. We consider a representation reasonable if it is able to encode enough discriminative information that can be used, for instance, in a classification task. Given this broad context, we define three questions that can be thought of as incremental steps of our research:

(i) Can astronomical properties be learned from images?

(ii) Can FITS images yield more information than RGB images?

(iii) Can reasonable representations that do not depend on labels be learned?

The first question arises from a curiosity about whether it would be feasible to map images directly to their astronomical properties using deep learning in an end-to-end fashion, without any intermediate steps that would likely require domain knowledge. The second question is aimed at verifying whether deep learning models can extract more information from raw FITS images than from RGB composite images which were generated with domain knowledge. Finally, the third question tackles the problem of how to include unlabeled data in the representation learning scheme. In order to evaluate and compare the performance of the models, we choose the task of classifying galaxies, stars and quasars.

The main contributions of this work are also threefold: (i) an open-source toolbox for preparing raw astronomical images for use in deep learning models, (ii) a comparative study between models trained with FITS and RGB images, and (iii) a deep representation learning approach that does not rely on class labels.

1.3 Work organization

This work is organized as follows. In Chapter 2, an overview of convolutional neural networks and self-supervised learning is given. In Chapter 3, Observational Astronomy concepts related to this work are introduced, and characteristics of the data survey used in this work are explained. In Chapter 4, the method proposed for pre-processing and learning representations from unlabeled astronomical images is presented. In Chapter 5, experimental

results are shown. And, finally, in Chapter 6, conclusions are drawn and possibilities for future work are elucidated.

Chapter 2

Fundamentals

In this chapter, theoretical concepts that will be used throughout this work are presented. In Section 2.1, we give an overview of convolutional neural networks (CNNs), starting by explaining how they work and how they can be trained, and then presenting important related concepts, such as: types of convolutional layers, hyperparameters, activation functions, optimization algorithms, and regularization techniques. In Section 2.2, we present a method known as self-supervised learning, which enables the use of unlabeled data to train deep neural networks, and give examples of some self-supervised learning approaches for images. Finally, in Section 2.3, we give an overview of techniques that allow visualization of the features learned by CNNs, presenting both methods that are focused on visualizing the inner workings of CNNs, and methods that are focused on reducing the dimensionality of the feature vectors extracted from neural networks so that they can be projected in 2D.

2.1 Convolutional Neural Networks

Neural networks are a family of powerful, expressive machine learning models whose origin can be traced back to the Perceptron [Ros58], proposed in the 1960s. They consist of sets of units that are connected to each other according to some connectivity rule, and that are organized in layers. Each unit in a layer receives inputs from units in the previous layer, generates an encoding that is a linear combination of those inputs based on a set of weights W, passes this enconding through a non-linear activation function, and then sends the output to some other units. The sets of weights related to each unit are learned, that is, they are fitted to a set of data points by optimizing a predefined cost function. In mathematical terms, each unit computes

$$y = \sigma(\sum_{i=1}^{N} w_i \cdot x_i + b) \tag{2.1}$$

where N is the number of inputs (that is, the number of units in the previous layer) and $\sigma(\cdot)$ is an activation function.

The layers that are neither the first input layer nor the final output layer are called hidden layers. The universal approximation theorem states that feed-foward networks (that is, networks whose connections do not form cycles) with a single hidden layer containing a finite, sufficiently large number of units can approximate any continous mapping from input to output, as shown in [Hor91]. In other words, it states that a simple neural network is capable of representing a great variety of functions when fitted to adequate sets of weights. Although this theorem is a very important one from a theoretical standpoint, it does not hint at the learnability of those weights.

The backpropagation algorithm [RHW86] allows these weights to be learned through gradient-based optimization. It provides an efficient way of computing the gradient of the cost function by making use of the chain rule. The key idea behind it is that data is passed through the network in a feed-forward manner, and then error vectors are computed backwards. Automatic differentiation can be used for numerically evaluating the derivatives; it is indeed used by deep learning frameworks, such as TensorFlow [ABC+16] and PyTorch [PGM+19]. These frameworks build neural networks as computation graphs, from which derivatives can be defined. After computing the gradient of a predefined error function, it must be passed to an actual optimization algorithm, such as the gradient descent. The optimization algorithm updates weights in such a way that the error function is minimized until stopping criteria are met.

In practice, it has been empirically shown that adding more layers to a neural network yields better performances. As the number of layers increases, however, fully-connected architectures (where all units between two consecutive layers are connected) become prohibitive. For certain types of data, it is possible to exploit different connectivity rules. For instance, connections based on convolutions.

Convolution is a mathematical operation that receives two functions and returns a third function computed by integrating the product of the two functions, after reflecting one of them over the y-axis. It expresses how one function changes the shape of the other. It is closely related to cross-correlation, an operation that describes how similar two functions are. The main difference in the definitions of convolution and cross-correlation is that, in cross-correlation, none of the functions are reflected. It is worth noting that, in most CNNs, neither the convolution filter nor the image are reflected, so strictly speaking, they use cross-correlation operations.

Convolutions satisfy a variety of nice properties, such as commutativity, associativity and the convolution theorem, which make them suitable for image processing. In discrete space, convolution is computed by

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m] \tag{2.2}$$

where, in the context of image processing, f is a convolution filter and g is the image itself (or vice-versa, since the operation is commutative). Note that both functions can be

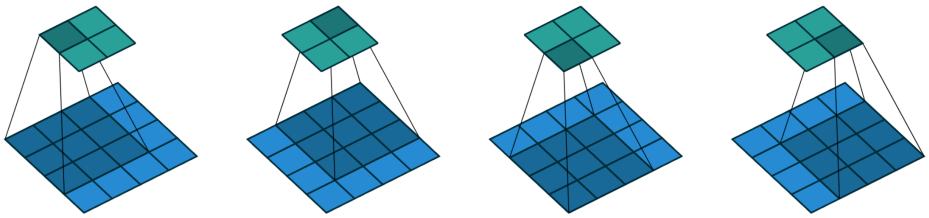


Figure 2.1: A convolution operation with `padding=0` and `stride=1`. Image from [DV16].

defined in two-dimensional (if images are composed of a single channel, such as greyscale images) or three-dimensional space (if images are composed of multiple channels, such as RGB images and FITS images), and that, in theory, the convolutions would be computed over the entire image.

In practical applications, a finite number of neighbours must be chosen for computing the convolution at each point. A well stablished approach is using squared (or cubic) convolution filters (also called kernels) of small size (as little as 3) that slide over the image, as represented in Figure 2.1. In traditional computer vision, filters are designed for describing features and performing specific tasks such as detecting edges and corners. CNNs arose as a disruptive approach wherein designing filters is no longer necessary – the weights of the filters are learned – and quickly became the state-of-the-art in many computer vision tasks.

Convolutional layers hold two important properties that make them more efficient than fully-connected layers:

(i) they use shared weights: a single set of small filters is slided over the entire image;

(ii) they use sparse connections: the convolution operation at a point depends on a small set of its neighbours.

Those two properties allowed convolutional architectures to go deeper, sometimes having hundreds of layers, and achieve unprecedented performance while keeping a somewhat feasible number of weights. During the 2010s, when the first large-scale CNN image classifiers came out, architectures have been proposed first to try and achieve peak performance strictly by going deeper [SZ14, SLJ+15], and then to try and mitigate efficiency issues [HZC+17, MZZS18, FRKK18]. CNNs still have a lot of redundancies that can be further pruned [FC19].

Some researchers believe that neural networks learn increasingly higher-level, higher-quality representations as data passes through deeper layers, whereas others believe that the networks learn a multi-step program that may or may not perform better as more layers are added. The former interpretation is more widely acepted, but the latter is supported by experiments with deep networks with stochastic depth [HSL+16].

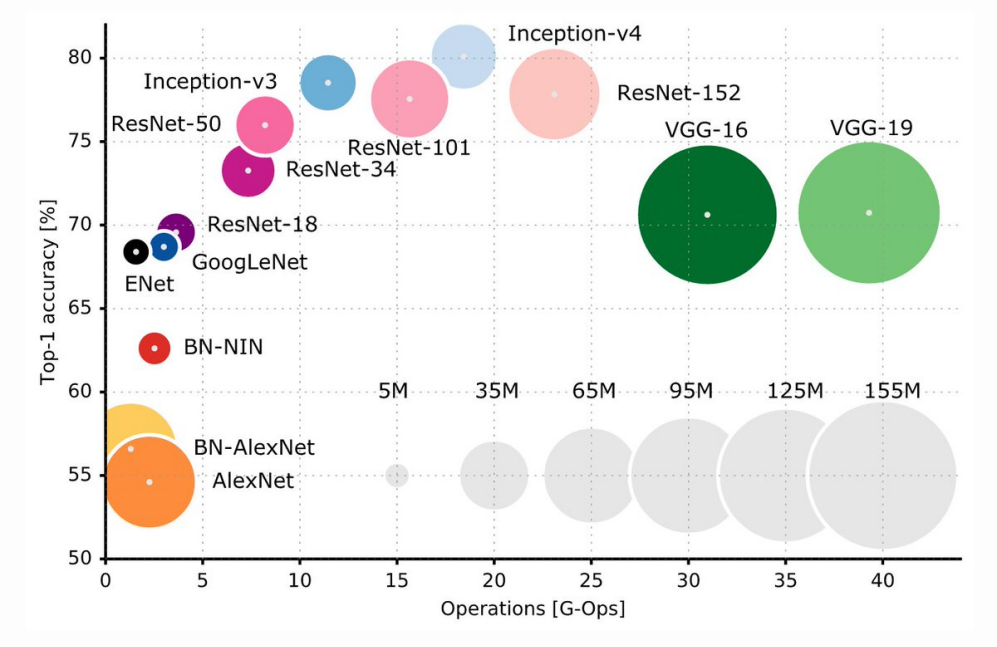


Figure 2.2: Top-1 accuracy of models trained on ImageNet versus amount of operations required for a single forward pass. Image from [CPC16].

Types of convolutional layers

Convolutional layers are the core of convolutional neural networks, thus it is expected that they will be the first feature of the networks to be explored and tweaked when searching for more performative architectures, both for minimizing errors and for maximizing efficiency. In this subsection, the most used types of convolution are listed, along with the architectures in which they were first adopted. Figure 2.2 illustrates how different models (that use different convolutional blocks) can present a wide range of sizes, accuracies, and amount of operations per single forward pass.

Standard convolution A standard convolution applied over multi-channel images is a three-dimensional operation. A standard convolution using cubic filters of side 3 over a 3-channel image yields $3^2 \cdot 3 = 27$ multiplication operations for computing each point.

Strided convolution Stride refers to the distance between spatial positions where a convolution filter is applied. Applying a strided convolution is equivalent to using a step larger than one when sliding the convolution filter over the image. Default convolutions have stride=1. Strides larger than one can be used for downsampling an image, for instance.

Dilated convolution Dilation refers to the distance between spatial positions while computing a convolution. Applying a dilated convolution is equivalent to using a step larger than one in the summation shown in equation 2.2. Dilation increases the receptive field (the region in input space that a given filter is "looking at") of the convolution. It was proposed in [YK16].

1x1 convolution A 1x1 convolution is a pointwise convolution along the depth direction. It is a way of performing feature reduction and of introducing more non-linearity to the model (it is usually followed by a non-linear activation function). It can also be seen as feature pooling. It was first proposed in the Network-in-Network architecture [LCY13], then popularized by the Inception model [SLJ+15] and largely adopted in newer models.

Spatially separable convolution Spatial separation decomposes a two-dimensional squared convolution filter of size s into two one-dimensional convolution filters of size s. It can be done when the columns or rows of the filter are linearly dependent, and reduces the amount of multiplication operations. Considering a squared filter of size 5, for instance, instead of $5^2 = 25$ operations, only $2 \cdot 5 = 10$ operations will be computed. This type of separation does not seem to be widely used in CNNs yet, but it is seen in the EffNet [FRKK18].

Depth-wise separable convolution Depth-wise separation decomposes a costly three-dimensional convolution into two-dimensional convolutions along the spatial dimensions (one per channel) followed by 1x1 convolutions along the depth dimension (one per pixel). Given a squared filter of size s and an input image of c channels, a three-dimensional convolution would yield $s^2 \cdot c$ multiplications per point, whereas depth-wise separable convolutions would yield $s^2 + c$ multiplications per point. It is used in Xception [Cho17] and MobileNet [HZC+17].

Grouped convolution Grouped convolution consists of splitting input channels into multiple groups and performing convolution in each of them independently. It was introduced by AlexNet [KSH12] to split a network into two GPUs, but they noted that each group of convolutions ended up consistently learning specialized features, which is an indication that this type of convolution may be used to extract higher-quality representations from images. Recent models such as ResNeXt [XGD+17] and ShuffleNet [MZZS18] make use of grouped convolutions. In the ResNeXt model, in particular, the number of groups can be controlled through a new architecture hyperparameter called cardinality.

Transposed convolution Transposed convolution is used in image-to-image tasks, such as image segmentation or image generation. It receives a small feature map and returns a larger map, using either padding or stride. It is also known as fractionally-strided convolution. Examples of models that use transposed convolution are Deep Convolutional Generative Adversarial Networks [RMC15] and Fully Convolutional Networks [LSD15].

Hyperparameters

Besides the complexity of the neural networks themselves, there is the additional complexity of setting several hyperparameters. Hyperparameters are parameters that, unlike weights, are not optimized during training. Their values must be set prior to training, in order to fully define the architecture and the training procedure. Hyperparameters can be split into two broad groups, one related to the architecture of the model, and the other related to the optimization algorithm. Examples of architecture hyperparameters are depth, width, filter size, number of filters per layer. Examples of optimization hyperparameters are batch size, learning rate, regularization norms, initial weights. The best choice of hyperparameters can also vary with data.

It is possible to carry out hyperparameter tuning in order to find optimal values for a given combination of dataset and architecture. The two most widely used techniques for searching hyperparameters are Grid Search and Random Search. As suggested by its name, Grid Search exaustively generates possible sets of hyperparameters from a discrete set of pre-defined values, and evaluates the performance of the model for each generated set. Random Search, on the other hand, samples sets of hyperparameters from a pre-defined distribution of possible values, and evaluates the performance of the model for each sampled set. Random Search tends to outperform Grid Search, specially for continuous spaces.

Both techniques are very well suited for models whose training costs are not prohibitive, such as Random Forests or Support Vector Machines. This, however, is not the case for neural networks. For finding optimal training hyperparameters for neural networks, Grid Seach may be performed over small sets of hyperparameters. Still, there is extensive research aimed at finding somewhat universal optimal values for some hyperparameters. For instance, it has been found that a batch size of 32 examples yields the best performance for a variety of settings [ML18]. As for optimal neural network architectures, there is a whole field of research concerned with finding them, known as Neural architecture search (NAS). NAS techniques can be based on reinforcement learning [ZL17] or on evolutionary algorithms [RMS+17].

Multinomial classification

Multinomial classification refers to a classification task with more than two classes, as opposed to binary classification, where instances are classified into two classes. Traditional models such as Support Vector Machines and Logistic Regression are binary by definition, and must be adapted in order to be used in multinomial classification problems. Such adaptation can be done either by transforming a single multinomial classification problem into multiple binary classification problems, or by extending existing binary classification frameworks to tackle multinomial problems.

Considering a classification task with k classes, the strategy of reducing one multinomial problem into multiple binary problems can be applied in two ways:

One-vs-one (OVO) A classifier is trained for each possible pair of classes, resulting in $k(k-1)/2$ different classifiers. When predicting the class of a new example, each classifier votes for a class, and the class with the most votes is assigned to the new example.

One-vs-all (OVA) A classifier is trained for each class, considering samples in such class as positive samples and all other classes as negative samples, resulting in k different classifiers. This technique requires that classifiers output a continuous score rather than a binary response, and that all those scores are defined in a single range (e.g., [0, 1]), so that all scores can be compared. When predicting the class of a new example, each classifier outputs a score and the class whose score was the highest is assigned to the new example. A drawback of this approach is that it yields highly imbalanced training sets. On the other hand, it is simple to add new classes to an existing multinomial classification frawework: one may simply train an additional classifier corresponding to the new class.

Using this reduction strategy, the number of classifiers required for a single multinomial classification task grows either linearly (for OVA) or quadratically (for OVO) with the number of classes, making it unfeasible for a large number of classes, as in the ImageNet dataset, which has a thousand classes.

The strategy of extending a binary framework is more efficient and, fortunately, quite straightforward to apply to neural networks: instead of adding a single unit to the output layer, k units may be added to the output layer, each corresponding to the score of a class. These scores may be normalized by applying a normalized exponential function (more commonly known as softmax function) on the output layer:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{2.3}$$

where i corresponds to a unit in the output layer.

This strategy of having k output units, all normalized by a softmax activation function, is similar to the OVO strategy in that both yield *relative* scores, or proportions, that add up to one. Those scores may be interpreted as probabilities, since they satisfy all three axioms of a probability:

(i) the probability of an event is a non-negative real number;

(ii) the probability of the entire sample space is one, that is, $P(S) = 1$;

(iii) if events A and B are mutually exclusive, then $P(A \cup B) = P(A) + P(B)$.

The use of the softmax function makes neural networks well suited for problems with a large number of classes, since only one classifier must be trained and then run on inference

time, regardless of the number of classes. Neural networks are trained using one-hot encoded vectors of class labels (i.e. $y = [1, 0, 0, 0]$ refers to class 1, $y = [0, 1, 0, 0]$ refers to class 2, and so on). Then, at inference time, the predicted class is given by $argmax(\hat{y})$.

Activation functions

Recall that each neural unit receives an input, generates a linear combination of this input, and then passes it through an activation function. Activation functions are a key feature of neural networks, because it is them that enable networks to learn non-linear mappings. An important property that they must satisfy is that of being differentiable, because their derivatives are required for computing the cost gradient.

One of the most well-known activation functions is the sigmoid, or logistic function. It maps real-valued numbers to [0,1], and therefore its outputs can be interpreted as probabilities. The softmax function can be regarded as a generalization of the sigmoid; it takes an input vector of size n and outputs n normalized probabilities proportional to the exponentials of the input numbers. Both the sigmoid and the softmax are usually added to the last layer of the network, in order to produce scores in [0, 1]. The former is used for binary classification and the latter for multiclass classification.

Note that the logistic function "squishes" a large input space into the output range [0,1]. This means that large changes in the input can result in small changes in the output; in other words, small derivatives. The vanishing gradient problem is the term given to the situation where the cost gradient becomes so small that the network gets stuck in a state and cannot be further trained. This usually does not happen in shallow networks, but it becomes a significant problem as more layers are added, since in the backpropagation algorithm, the derivatives of each layer must be multiplied by each other, from last to first, in order to find the derivatives of the first layers.

Many activation functions have been proposed in order to mitigate limitations such as the vanishing gradient. One is the hyperbolic tangent ($tanh$), which has a shape that is very similar to that of the sigmoid, but maps outputs to [-1,1], making derivatives a bit larger. Another one is the rectified linear unit ($ReLU$), which has no upper bound, making derivatives even larger. Besides avoiding the vanishing gradient, the $ReLU$ function is also cheap to compute, since it consists of two linear functions. The $ReLU$ function, and some of its variants, are widely used in hidden layers nowadays. Figure 2.3 shows a comparison between sigmoid, $tanh$ and $ReLU$.

Optimization and regularization techniques

The cost function to be optimized is generally non-convex, meaning that optimization algorithms can get stuck in local minima. Many gradient-based optimization algorithms for machine learning have been proposed. The most traditional one is the gradient descent, in

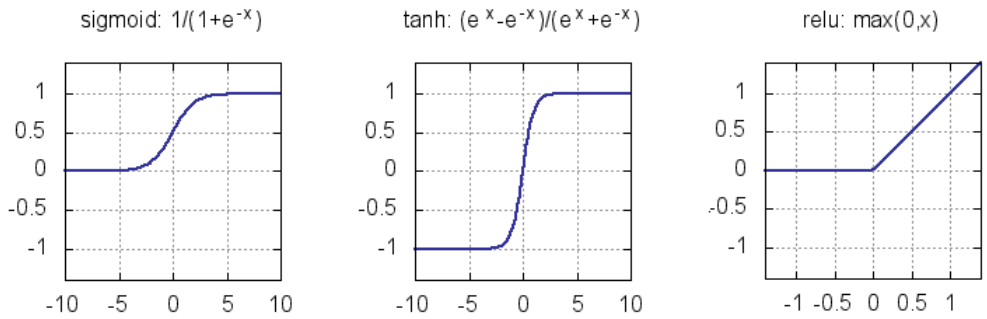


Figure 2.3: Sigmoid, tanh and ReLU activation functions. Image adapted from [Yur].

which the weights are updated after each epoch[1]. This results in an unstable, slow training procedure. In practice, variants such as stochastic gradient descent (SGD) and mini-batch gradient descent are more largely used. In the former, the weights are updated after each example is passed through the network, and in the latter, they are updated after a *mini-batch* of examples is passed through the network. Using mini-batches improves training speed while maintaining stability. Small batch sizes, between two and 32, are more recommended [ML18]. Another optimization algorithm that is based on SGD but tends to converge faster is the Adaptive Moment Estimation (Adam) [KB15].

To avoid getting stuck in local minima while keeping a decent convergence speed, a learning rate must be set and adequately updated during training. The learning rate is a hyperparameter that controls how much the weights are adjusted to the cost gradient computed at each iteration. Appropriate learning rate values are highly dependent on the optimization algorithm, and also a bit dependent on the architecture of the network. It is common practice to start training with a relatively high learning rate (roughly between 10^{-2} and 10^{-4}), to accelerate convergence at a more "rough" level, and to employ a learning rate schedule that gradually decreases the learning rate, to avoid jumping over minima when the optimization process is at a more fine level. Sophisticated algorithms such as Adam compute adaptive learning rates, ditching the need to manually define a learning rate schedule. Even in those cases, it is still common practice to employ additional strategies to control the learning rate, such as reducing it when apparent plateaus are found over the cost surface.

There is one more thing that must be paid attention to when doing machine learning: overffiting. It happens when the model adjusts too well to training data and fails at generalizing to new data. To avoid this, regularization techniques are often employed. They are specially necessary when dealing with large, complex models such as neural networks. A well-known technique that was borrowed from traditional regression models is the L2 regularization, sometimes also called weight decay, that basically consists of reducing computed weights by a small factor. Two other popular regularization techniques specific to neural networks are Dropout [SHK+14] and Batch Normalization [IS15]. Dropout consists of randomly "erasing" a portion of the neural units at each training iteration, in order to avoid neighbouring units

[1]An epoch refers to one iteration that passes every training example through the model.

from co-adapting, that is, to prevent neighbouring units from learning to capture the same patterns. Although Dropout often yields more robust models, it also indicates that models are very redundant. Batch Normalization, which has also been shown to improve convergence speed and stability, consists of normalizing layer inputs for each training mini-batch. It has been observed that Dropout and Batch Normalization often do not mix well [LCHY18]. Usually, architecture designers pick whichever techniques they believe work better in their specific contexts.

2.2 Self-supervised Learning

Self-supervised learning is a strategy devised to leverage unlabeled data for training deep neural networks. It has shown promising results when applied to images [KZB19] and to text [LCG+20]. It consists of producing attributes that can be cheaply and automatically computed from unlabeled data, and use those attributes as labels to train models on a pretext task, with the objective of learning representations that can be used in higher-level tasks, such as classification, clustering and outlier detection. Some methods of producing pretext tasks for image data are:

Clustering [CBJD18] Features extracted from a (randomly initialized) model are clustered and their cluster assignments are used as pseudo-labels for iteratively training the model. The method alternates between clustering features in order to generate pseudo-labels and using such pseudo-labels as targets for updating the model.

Image Rotation [GSK18] Four copies of images are generated by rotating them by 0°, 90°, 180° and 270°, and a model is trained to predict which rotation was applied to each image.

Relative Patch Location [DGE15] Pairs of patches from images are generated, and a model is trained to predict the position of the second patch relative to the first (e.g. north, east, northeast, and so on).

In Astronomy, most data is unlabeled, but useful properties of the objects, such as brightness and size, can be readily computed from analytical models, without supervision. This yields an ideal setting for using self-supervision, where computed properties are used as targets for regression tasks. In this way, unlabeled data are included in the learning process, leading to representations that are closer to the distribution of the observed objects in the universe at large.

2.3 Feature Visualization

Features extracted by deep neural networks are notably hard to analyze and interpret. Promising branches of study for evolving in the direction of more interpretable neural networks

Figure 2.4: Feature visualization produced by optimizing a channel objective, that is, finding out which kinds of input would produce a certain response in a channel [OMS17].

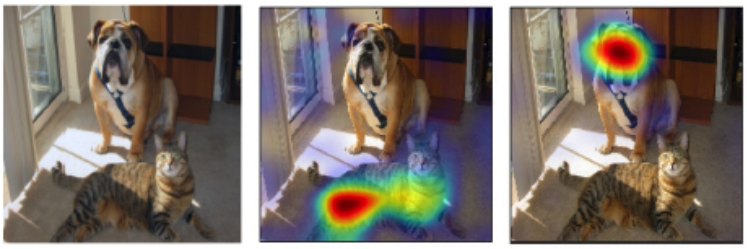


Figure 2.5: Activation maps for cat and dog classes produced by using the gradient flowing into the last layer of a CNN to assign importance values for each unit (neuron) in that layer [SCD+16].

for computer vision tasks are the visualization of the features learned by units (neurons) and layers [MOT15, OMS17], and the visualization of activations [SCD+16]. Feature visualizations are used for checking which visual patterns the model learned for distinguishing between classes, whereas activation visualization are used for checking at which regions of an image a model "looks" when predicting its class. Both can be really useful when analyzing results in so-called natural perception tasks. Because human performance is usually very good in those tasks, we humans can assess whether the patterns learned by the model make sense. For instance, if we train a model for recognizing dogs and cats, we can use feature visualization (as shown in Figure 2.4) and activation visualization (as shown in Figure 2.5) to try and understand some of its inner workings.

Computer vision tasks are usually considered natural perception tasks. However, some patterns, such as those hidden in financial or astronomical data, are very unlikely to be grasped by human perception. As an example, Figure 2.6 shows astronomical objects that are quite difficult to distinguish, even to the trained eye, but that belong to three different classes: stars, galaxies, and quasars. For such tasks, human performance cannot be used as a baseline for assessing machine performance.

In this scenario, tools such as feature maps and activation maps do not provide useful insights on the performance of the model, and more sophisticated techniques are required. A popular technique for analyzing complex, high-dimensional features, such as the ones extracted by a deep neural network, is dimensionality reduction. Two of the most widely used

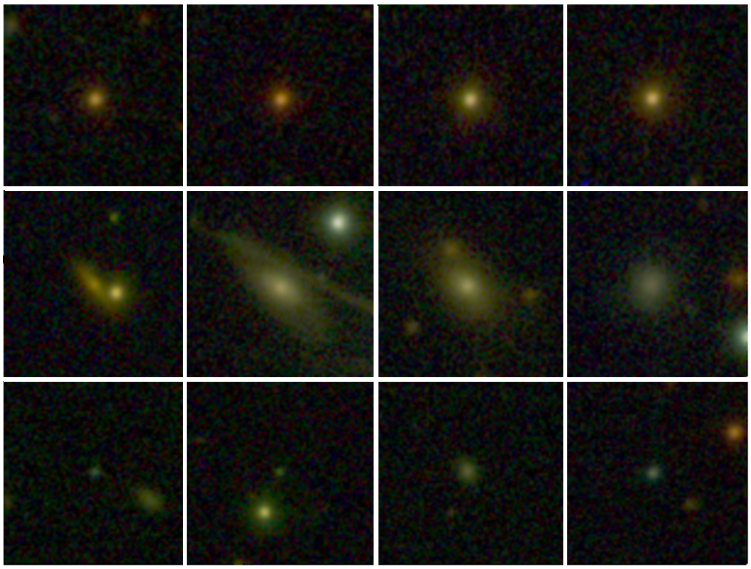


Figure 2.6: Sample images of stars (top row), galaxies (middle row) and quasars (bottom row) from the S-PLUS [ORS+19]. Since sometimes such objects look nearly the same to the human eye, the interpretation of a machine learning model that classifies them cannot rely solely on methods such as feature maps and localization maps. More sophisticated techniques, such as dimensionality reduction, should also be adopted.

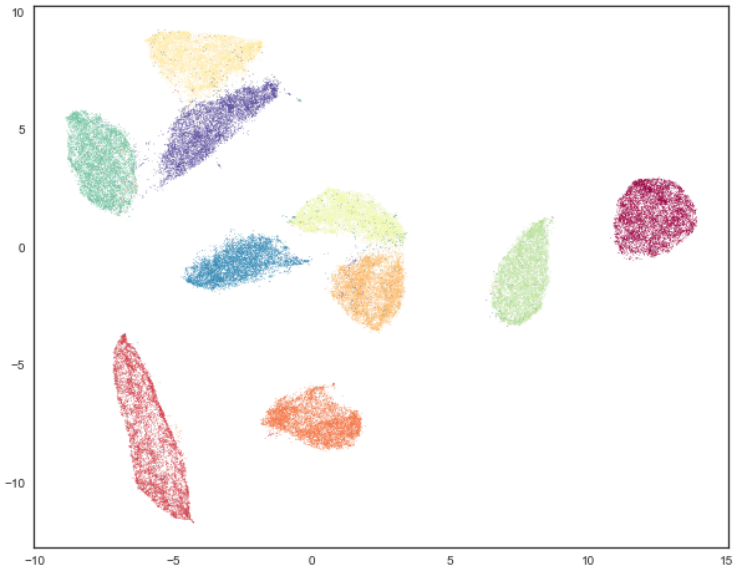


Figure 2.7: Raw 784-dimensional vectors from the MNIST dataset [LC10] projected into two dimensions using the dimensionality reduction algorithm UMAP [MHM18]. Such projections can be used to aid with interpretation of high-dimensional features extracted from complex data.

dimensionality reduction algorithms are t-SNE [LvdM08] and UMAP [MHM18]. Figure 2.7 shows two-dimensional projections of the well-known MNIST digits dataset [LC10], produced by UMAP. Such projected features may be used to find clusters and outliers that can be more carefully analyzed afterwards.

Chapter 3

Astronomical Data

In this chapter, an overview of concepts related to astronomical data is given. In Section 3.1, we present a brief primer on Observational Astronomy, where we detail the objects that we will identify in this work, and explain how object magnitudes and sky coordinates are defined. Magnitudes and sky coordinates are important concepts because they are used in Chapters 4 and 5 in order to build datasets and analyze results. Then, in Section 3.2, we present the Southern Photometric Local Universe Survey (S-PLUS), the sky survey that generated the data used in this work, and explain how such data were generated. Finally, in Section 3.3, we explain the three data formats used in this work: FITS images, composite RGB images, and catalogs.

3.1 A Very Brief Primer on Observational Astronomy

Observational Astronomy [Bur18] is the field of astronomy concerned with recording data about the observable universe, as opposed to Theoretical Astronomy, which is concerned with describing the observable universe through analytical models.

Astronomical data comes in two main formats: spectra and images. Spectra consist of nearly continuous measures of energy fluxes emitted or absorbed by an object as a function of wavelength. Their fine-grained resolutions allow detection of narrow peaks and valleys that are reliably used to categorize objects and determine many of their properties, such as which chemical elements are present in them and what are their distances from us. These information, in turn, are used to validate theories of how and when a given type of object formed, how it evolved, what are the physical processes involved in its formation and evolution. In spite of providing such rich and fine-grained information, spectra are hardly scalable: a spectrograph requires hours of exposure in order to collect enough photons across all wavelengths. Spectral data, thus, is expensive and scarce. There are a few well-known databases of spectra, such as the ones from the Sloan Digital Sky Survey (SDSS)[1], that have become significantly large. Still, there is an inherent bias in how objects were chosen for spectroscopic analysis.

[1] https://sdss.org/dr15/spectro

To learn about the universe at large while minimizing risk of overfitting theoretical models to biased datasets, images are needed. Images can be understood as a discretized version of spectra, in which energy flux measures are grouped in broader ranges of wavelength. The number of passbands (also called filters) may vary from five to a few tens. Photometric surveys yield information about numbers of objects that are orders of magnitude larger than spectroscopic ones. Even though these information usually come at the cost of larger uncertainties when estimating properties of the objects, they are vital for defining the distribution of objects at large scales, for capturing transient phenomena (such as asteroids and supernova explosions), and also for detecting unusual phenomena (the so-called outliers) that may be more carefully studied with a spectrograph afterwards. Besides, images provide rich morphological information that is usually absent in spectra, and that can be used for studying formation and evolution of objects in different ways.

Types of objects

Examples of objects that can be seen in the deep night sky are stars, planets, comets, asteroids, galaxies [Moc09]. Sky surveys are tailored for studying predefined sets of objects. In this work, we focus on identifying three types of objects:

Stars Stars are the majority of objects that are visible to the naked eye. They look like point objects, except for the Sun, which is very close to our planet and looks more like what it really is: a spheroid of plasma (ionized gas) bound together by its own gravity. The earliest stellar classification system divides stars into seven spectral classes based on their surface temperatures (which also determines their colors). Stars can also be divided into three populations based on their chemical composition and age. Population I stars are young stars with high metallic content[2]. Population II stars are intermediate stars with relatively little metallic content. Population III stars are very old stars that contain no metals. The existence of Population III stars was inferred from theory and they are thought to be extinct by now. The search for these ancient stars is of great interest because they can yield information on how the very first stars formed.

Galaxies Galaxies are extense objects composed of stars, star remnants, gas, dust and dark matter. The galaxy in which our Sun in located is the Milky Way, and all stars that can be individually resolved in images are in Local Group galaxies. The Milky Way itself cannot be completely observed because we are inside it. It appears as hazy, curved spread of stars and dust in the night sky – hence the name Milky Way. The observation of the Milky Way, even if incomplete, enables the study of details that may be generalized to other galaxies, and the observation of other galaxies enables the validation of a general theory of how galaxies form and evolve. Galaxies are classified

[2]In Astronomy, all elements other than hydrogen and helium, the two lowest-mass elements, are called metals.

based on their morphology as elliptical, spiral or irregular, and each of those classes also contain subclasses [Hub26].

Quasars Quasar is a contraction word for quasi-stellar object. This type of object was first identified in the 1960s [Sch63]. They are extremely bright objects located at the center of some galaxies, composed of supermassive black holes surrounded by a gaseous structure. Black holes are very dense, compact objects that yield a gravitational field so strong that not even light can escape it. Even though the black holes themselves are invisible, when gas falls towards them, enourmous amounts of energy are released. This radiation is seen in images as star-like structures, even though it comes from a completely different type of object. Because they are so bright, they are the farthest objects that can be detected in surveys, which can yield information about the frontiers of the observable universe.

Magnitudes

Magnitude is a dimensionless measure of the brightness of an object in a given passband. The magnitude system was first introduced by Hipparchus, a Greek astronomer, back in the second century BC. It is based on a logarithmic scale, defined such that a change of one in magnitude corresponds to a change of $100^{1/5}$ in brightness. Lower values of magnitude correspond to brighter objects, whereas higher values of magnitude correspond to fainter objects. For instance, a star of magnitude 1 is 100 times brighter than a star of magnitude 6.

There are two kinds of magnitude: the apparent magnitude and the absolute magnitude. The absolute magnitude is related to the luminosity emitted by an object, and the apparent magnitude is related to the fraction of this total flux that can be measured from Earth. The apparent magnitude is computed through a comparison between a given object and a reference object, which is usually a well-studied, nearby star. Only the apparent magnitude can be directly measured from an image. The absolute magnitude is computed from the apparent magnitude and the distance of the object.

Sky coordinates

A celestial coordinate system is a system for determining the positions of astronomical objects in the sky [Bur18]. A robust, universal coordinate system is essential for the production and consumption of astronomical data. It allows professional astronomers to build catalogs of objects using a standard protocol across various surveys, and to cross-match observations of the same object across multiple surveys. It also allows amateur astronomers to contribute; for instance, by reporting unusual observed phenomena (such as supernova explosions).

The equatorial coordinate system is the standard in sky surveys. It is based on the abstraction of the celestial sphere, a sphere that is concentric to Earth, that has arbitrarily large radius, and to which all astronomical objects are “glued” regardless of their true distance

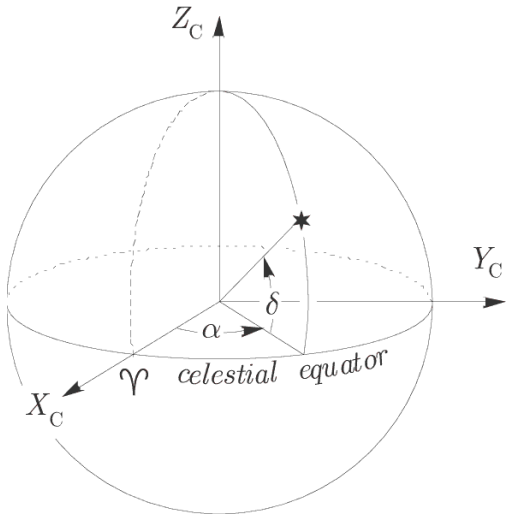


Figure 3.1: Equatorial coordinate system, defined by the right ascension α and the declination δ. Image adapted from [Dea07].

from Earth. This enables the determination of the apparent position of an object when its distance is unknown. Adopting the Earth's equator as a referencial plane, a pair of spherical coordinates can be defined: declination and right ascension.

The declination δ is analogous to the terrestrial latitude. It measures the angular distance of an object perperdicular to the equator, starting from celestial equator (the projection of the equator line onto the celestial sphere). It grows positive in the North direction and negative in the South direction, ranging from $-90°$ (the South Pole) to $90°$ (the North Pole). The right ascension α, likewise, is equivalent to the terrestrial longitude. It measures the angular distance of an object in the eastly direction along the celestial equator, starting from the March equinox Υ[3] to the meridian passing through the object. Right ascension ranges from 0 to 24h (or 360°). Figure 3.1 illustrates how the declination and the right ascension of an object are computed.

In many fields of research, right ascension and declination are enough information on the position of an object. However, in order to derive three-dimensional distributions, an additional coordinate is required. In these situations, astronomers frequently adopt redshift as a proxy for quantifying how far an object is. Redshift is the term used to define a displacement towards redder, lower-frequency wavelengths that is observed in spectra. This displacement happens when an object is receding from the observer and the waves emitted by this object are received by the observer at a lower frequency, a phenomena known as Doppler effect. The opposite effect (that is, waves being received by the observer at a bluer, higher frequency when the object is moving towards her/him) also happens, albeit rarely, and is called blueshift. Both the redshift and the blueshift are instances of the Doppler effect.

The redshift is a dimensionless quantity that is computed by the ratio

[3]The March equinox corresponds to the point at which the Sun crosses the celestial equator from the South to the North.

$$z = \frac{\lambda_{observed} - \lambda_{emitted}}{\lambda_{emitted}} \tag{3.1}$$

where λ is wavelength.

Without information on distances, the properties of the objects are said to be apparent. For instance, if a nearby star is compared to a distant galaxy simply by looking at their images, without taking into consideration their distances, it does look like the star is brighter than the galaxy, even though a galaxy is made up of tons of stars. Apparent properties, such as apparent brightness, are useful for a lot of studies. On the other hand, sometimes it is necessary to derive intrinsic properties, such as absolute brightness, size, mass. Those are all dependent on distance. Thus, while two-dimensional coordinates may be well enough for a variety of studies (including the object identification problem explored in this work), having precise three-dimensional coordinates is required to carry out other kinds of studies (for instance, the problem of validating theoretical models of object formation and evolution).

3.2 The Southern Photometric Local Universe Survey

The images analyzed in this work were collected by the Southern Photometric Local Universe Survey (S-PLUS) [ORS+19], an ongoing sky survey that was launched in 2016 and will provide the scientific community with detailed image maps of our Southern skies. Among the scientific goals of the S-PLUS project are the search for very old stars, the search for distant quasars, and the study of star formation in various galactic environments. The study of old, distant objects and of how they formed is aimed at a greater understanding of the early Universe.

The S-PLUS is carried out with the T80-South telescope, located at the Cerro Tololo Inter-american Observatory, in Chile. The T80-South is the Southern twin of the Javalambre Auxiliary Survey Telescope (T80/JAST), installed at the *Observatorio Astrofisico de Javalambre*, in Spain. Both telescopes use the same camera and the same filter set. The T80 is currently carrying out the Javalambre Photometric Local Universe Survey (J-PLUS) [CMCH+19], which is similar to S-PLUS but focused on the Northern sky.

In order to accurately capture signals that allow the observation of fine-grained characteristics of objects, light is collected by the S-PLUS using twelve filters. Each of these filters lets through ranges of wavelengths that correspond to different physical phenomena. Figure 3.2 illustrates how much light each filter lets through as a function of wavelength.

The five broad filters correspond to the widely used ugriz photometric system, composed of ultraviolet, green, red, near infrared and infrared filters, which are suitable for constraining the spectral curve of an object. The other seven filters – F378, F395, F410, F430, F515, F660 and F861 – are narrow and were specially designed by the T80/JAST team with the purpose of approximating spectral lines of chemical elements that are important for characterizing objects.

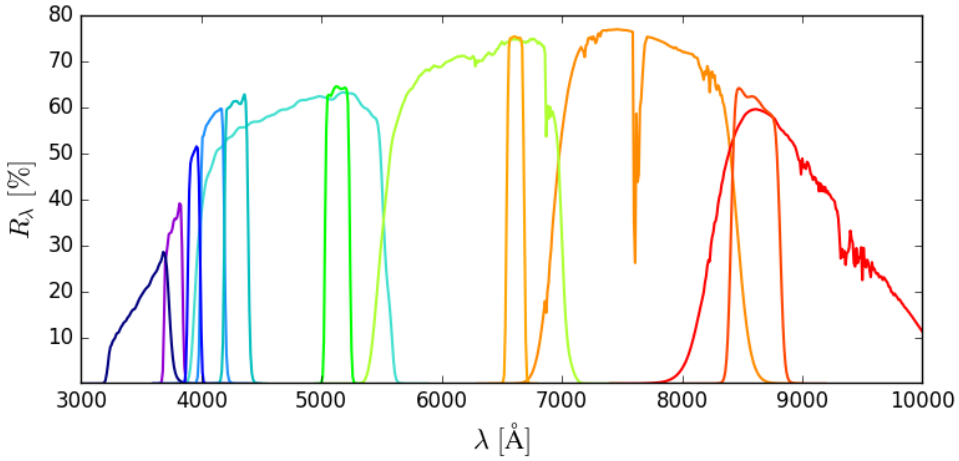


Figure 3.2: Transmission of the S-PLUS filters as a function of wavelength [ORS+19].

The First Data Release of the S-PLUS, which has been made public in July 2019, covers a region of the sky known as Stripe82. It is a stripe of the sky around the Equator, whose declinations range from $-1.26°$ to $1.26°$ and right ascensions range from $300°$ to $60°$. Being a very well studied region, that has been imaged by multiple surveys both from the Northern and the Southern Hemispheres, it is ideal for trying out new image processing approaches that may later be applied to other regions.

The full Stripe82 dataset consists of 170 fields imaged by twelve different filters. Each of these fields yields images of 9232x9216 px. The resolution of the images is such that 1 pixel corresponds to 0.55 arc seconds. For comparison, the angular diameter of the Moon is roughly 31 arc minutes – if the Moon were to be imaged by this survey, its diameter would be around 3380px.

The S-PLUS features an internal pre-processing pipeline that yields what are called science-ready images and object catalogs. The raw imaging data collected by the telescope goes through a low-level image processing technique termed coadding, which consists of registering and stacking images. This is common practice in astronomical image processing: the same region of the sky is imaged multiple times in order to enhance faint signals (that would otherwise be missed or interpreted as noise) and reduce uncertainties. Having the same region imaged many times is also important for sky surveys focused on identifying time-variant objects, such as comets, asteroids (whose positions vary), variable stars and supernovae (whose fluxes vary). It is worth noting that, in practice, regions are not perfectly uniformly sweeped. For instance, when two neighbouring regions of the sky are imaged, it is usual to add some overlap between them. This overlap is necessary to ensure homogeneous flux calibration between the fields (in other words, to avoid discontinuities between fields), but it may also generate artifacts, such as those areas near the borders having more detected objects because those areas were imaged more times.

After going through low-level processing, images are segmented, objects are detected, and an object catalog is generated using the Source Extractor (SExtractor) software [BA96].

Although alternative approaches for detection of astronomical objects have been proposed [MFLP12], SExtractor is still the most widely used by the community. SExtractor passes the images through a pipeline that includes background subtraction, object segmentation and property estimation. The background subtraction phase consists of estimating global and local mean background values, combining them, and subtracting them from the images. The object segmentation phase is based on adaptive thresholding techniques, which work sufficiently well for the problem of separating bright point sources from a dark background. Finally, the property estimation phase is carried out by fitting each object to mathematical models that yield measurements of properties such as magnitudes and full-width half-maximum[4]. The resulting S-PLUS catalog contains more than 3 million objects, and roughly a third of them have missing values because their signals are too weak in some bands. SExtractor does not generate "universal" descriptors that can be used to reliably classify all the detected objects, thus it is of great interest to develop robust methods that work well for a variety of signals.

3.3 Data formats

Astronomical data used in this work comes in three different formats: FITS images, composite RGB images, and catalogs. FITS images are collected by photometric sky surveys and can be used to generate RGB composite images and photometric catalogs. In this section, we further detail each of those formats.

FITS images

The FITS images are stored in FITS (Flexible Image Transport System), a file format widely used in Astronomy to store and transport multidimensional arrays and tables. To manipulate these files, the Astropy package [CPWS+18] is used. Those images are the result of the pre-processing pipeline mentioned in Section 3.2 that performs low-level processing tasks – as registration, stacking and background subtraction – over the raw images. Knowing that each pixel value corresponds to a count of photons that arrived at that position over a long exposure time, it is expected that the pixel values are defined over larger ranges when compared to standard unsigned 8-bit images. Moreover, the background subtraction task yields artifacts such as negative and real-valued pixels. In order to take in these peculiarities, the image arrays from the S-PLUS were stored as 32-bit floating-point numbers.

In the context of preparing the images for usage in deep convolutional networks, this incurs in a problem: how to normalize these images in a reliable, robust manner? Unsigned 8-bit images can readily be mapped to the [0, 1] range by dividing their values by 255. Unsigned 16-bit images, such as the science-ready ones from the SDSS, can also be mapped to [0, 1] by dividing their values by 65535. 32-bit floats, however, can store values in the

[4]The full-width half-maximum (FWHM) refers to the width of a Gaussian at half of its maximum value; it is used to approximate the diameter of point sources that do not have sharp edges.

Figure 3.3: 1000x1000px image patch of the S-PLUS survey. From left to right: r-filter image in linear scale; r-filter image in asinh scale; composite RGB in asing scale.

range $[-3.4 \cdot 10^{38}, +3.4 \cdot 10^{38}]$.

In Section 3.1, we presented the magnitude system, which was devised to categorize objects according to their brightnesses. The magnitude system is based on a logarithmic scale. It was developed at first by looking at stars with naked eye, and it is now known that human vision perceives changes in brightness in a logarithmic scale. Non-linear transformations are often used by astronomers in order to enhance subtle structures, such as diffuse dust around the center of an object, which would not be visible (for humans) in a linear scale. The *log* transform is the most used one, but it is not defined for negative values. The inverse hyperbolic sine (*asinh*) function is proposed as an alternative to the logarithm when images have negative values [LGS99]. The *asinh* function behaves like a linear function for small absolute values, and becomes similar to the logarithm for large absolute values. Figure 3.3 shows a 1000x1000 px patch of a S-PLUS image in linear scale and in asinh scale. Note how many objects become visible and more detailed after non-linear transformation.

Another approach for preparing FITS images for use with neural networks is to calibrate them. The idea of calibration is to transform photon counts (i.e. raw pixel values) into surface brightnesses, which are not dependent on the photon detection system and thus can be used to compare results from multiple surveys [PD01]. Calibration is considered a mandadory preprocessing step when studying images in the context of astronomical research. In the context of applied machine learning, it can be understood as a way of stardardizing pixel values, which helps the neural network converge faster and to a lower minimum.

In the context of the S-PLUS DR1, the calibration transform function can be defined as

$$x_{calib} = \frac{10^{5-0.4zp}}{ps^2} x \tag{3.2}$$

This calibration function takes the pixel scale of the detector ps (in units of angle per pixel, e.g. arcsec/pixel), a raw image x, and a vector of zero points[5] zp (one per channel) as inputs, and returns a calibrated image x_{calib} as output. It is worth noting that the pixel scale

[5]The zero point is defined as the magnitude of an object that yields one count per second on the detector.

is a constant for all images (more specifically, $ps = 0.55$ for the S-PLUS system), and thus the only varying inputs are the zero point value and the image itself. Zero points are constant for each combination of channel and field, and different zero points represent variations in sky conditions (e.g. clear or foggy sky; new or full moon) and photon detectors (e.g. detectors for photons of different wavelengths).

The calibration procedure works by applying a different constant for each combination of channel and field, which adjusts for different image capture conditions and yields standardized images. The results presented in this work were obtained using calibrated FITS images with no additional preprocessing.

RGB images

Composite RGB images are built by combining at least three color channels from FITS images. Images can be close to their "natural" color (the color we would perceive if we were out there in space actually looking at an object) if the FITS image contains channels that are similar enough to red, green and blue. For science communication, it does not really matter whether images are natural. Representative images can be built out of any combination of channels, even channels that extrapolate the limits of visible light (e.g., X-ray radiation), with the purpose of visually revealing information about an object. In general, channels are assigned in chromatic order (e.g., the channel with lowest wavelength should be assigned to blue, and the channel with highest wavelength should be assigned to red). Sometimes, the chromatic order is not followed either for purely aesthetic reasons or for enhancing specific phenomena.

The S-PLUS pre-processing pipeline generates composite RGB images using the Trilogy software [Coe15]. It applies a non-linear transformation (usually $asinh$) over the FITS images and then combines all twelve channels by adding multiple channels into each of the R, G and B channels. Figure 3.3 shows a patch of a S-PLUS image in colors.

It is worth noting that, even though composite RGB images can encode information from multiple channels, they are rarely used by themselves to carry out research. In general, each channel is analyzed separately in greyscale.

Object catalogs

Spectroscopic object catalogs are built by extracting attributes directly from spectra. Photometric object catalogs, on the other hand, are built by performing image segmentation on the FITS images and then estimating attributes of the segmented objects through mathematical model fitting. In this work, the following attributes from the S-PLUS DR1 catalog[6] are used: right ascension, declination, x pixel coordinate, y pixel coordinate, full-width half-maximum and twelve magnitudes (one per filter).

[6] https://datalab.noao.edu/splus

There are numerous ways to compute magnitudes from images [Ber17]. In the S-PLUS catalog, three magnitude estimates per filter are provided: `aper, petro, auto`. `aper` refer to integrating light using a fixed circular aperture, which is best for comparing values between different datasets and surveys. `petro` is the Petrosian magnitude, in which light is integrated using a diameter that varies as a function of the local light profile in the neighbourhood of a given object; this method integrates most light but yields lower signal-to-noise ratios, and is best for estimating physical properties of the objects. `auto` also refers to a method that integrates light using a variable diameter, but this diameter estimation is more restricted, which yield higher signal-to-noise ratios and thus lower uncertainties. Throughout this work, only `auto` magnitudes are used.

Chapter 4

Method

In this chapter, we present our method for learning representations from unlabeled astronomical images. In Section 4.1, we present our self-supervised learning approach, which is composed of a pretext task that makes use of astronomical properties and a downstream supervised task. Then, in Section 4.2, we present our data preparation pipeline, which is composed of two steps: (i) selection of a subset of objects from S-PLUS and (ii) image preprocessing. Finally, in Section 4.3, we present implementation details, including neural network architecture choice, training procedure, and resource usage. We also share the repository where code written to run experiments is available.

4.1 Our Self-supervised Learning Approach

The emergence of open-source tools for processing and analyzing astronomical data, such as Astropy [CRT+13, CPWS+18], Astroquery [GSB+19] and AstroML [VCIG14], combined with the daily generation of terabytes of astronomical data by modern sky surveys, has fostered ML-driven astronomical research. In a comprehensive survey of the reach of ML and AI in Astronomy [FJ19], authors found that subfields related to the study of stars and galaxies are among the most mature in the adoption of ML tools. One of the most common application of ML in such subfields is object classification.

In such object classification tasks, astronomical properties alone are commonly structured in tabular format and used as input features for training traditional ML models (i.e. not DL). Examples of supervised tasks that use structured data include: classification of RR Lyrae stars[1] [SHM+16] using XGBoost [CG16]; star-galaxy [CDSM+19] and star-galaxy-quasar [NdOH+21] classification using RandomForest [CG01].

For tackling problems with unstructured data, such as images, DL-based techniques have begun to be used. Examples of supervised tasks that use images include: galaxy morphology classification [SHCB+17], galaxy detection in deep field images [GMH18], quasar classification [PIP18], and redshift estimation [Hoy16, PIP18]. For learning representations

[1] A RR Lyrae star is a type of star whose brightnesses, as seen from Earth, vary periodically.

from unlabeled data, various autoencoder-based approaches have been proposed [GFHL14, AC19], which are sometimes referred to as self-supervised. Besides that, contrastive learning has also been proposed in recent works [HSH+20, HHS+21]. Contrastive learning consists of learning representations in such a way that representations of similar samples (e.g. an image crop and augmented versions of it) should remain close, and it is considered a self-supervised learning approach as well.

The appearance of such works indicate that, indeed, self-supervised learning is a promising direction for learning representations of astronomical objects. This is because although deep neural networks are capable of performing automatic feature extraction for unstructured data, such as images, the quality of the generated encodings are heavily dependent on the specific task for which the model was trained. In other words, an encoder that is trained to classify a set of stars, galaxies and quasars is capable of generating encodings that include information that is relevant to differentiate those three classes, but such encoder will likely not be well suited for generating informative encodings from other types of objects.

The core idea behind the adoption of self-supervised learning approaches is that including unlabeled data in the training process yields more representative features. In order to leverage both unlabeled data and domain knowledge, we propose a self-supervised learning approach composed of a pretext task and a downstream task as follows:

Pretext task Train a neural network using unlabeled images as inputs, and astronomical properties as outputs.

Downstream task Fine-tune this neural network using labeled images as inputs, and classes as outputs.

Note that the pretext task does not use any information about classes. It uses information about astronomical properties that can be easily and reliably computed from unlabeled images, such as magnitudes. Since these properties are usually represented by continuous values, the pretext task is more likely to be a regression task. Still, properties could be quantized in order to transform the regression task into a classification task. Also note that the idea behind the pretext task is somehow generating signals that can be used as targets for training a model in a supervised manner, which indeed corresponds to a self-supervised learning setting, as detailed in Section 2.2. After the pretext task, the pretrained model may be fine-tuned for various downstream classification tasks.

Table 4.1: Number of samples in each split of datasets X_u (unlabeled) and X (labeled).

	X_u	X	%
train	104871	90697	90
validation	5520	4774	5
test	5811	5025	5

4.2 Dataset preparation

Selection of a subset of objects from S-PLUS

A subset of the objects catalogued in S-PLUS DR1 [ORS+19] was selected for this work. This subset was obtained by filtering out all objects with warning flags[2] or with missing magnitude values. Such missing values can happen when not enough photons are collected by a given photometric filter, so the magnitude value relative to that filter cannot be computed. We are aware that our filtering approach may generate a biased subset, with distributions that are different from the ones found in the universe at large. We did consider employing simple inputation methods, such as setting the mean or the median of a column for all rows which have that column missing. However, this would also change data distributions, and moreover, it could hurt the quality of the representations we want to learn.

After filtering out all objects with missing magnitudes or warning flags, the remaining objects were then matched to the spectroscopic catalog from SDSS DR15 [AAA+18] by searching for objects which have the same coordinates (right ascension and declination) within a tolerance of 1 arc second ($\sim 2 \cdot 10^{-4}$ degrees) using the Astropy package [CPWS+18]. Since both the S-PLUS and the SDSS coordinates have a precision of 10^{-4} degree, a tolerance of $2 \cdot 10^{-4}$ degree was considered reasonable. Larger tolerances were tested, but because some regions of the sky are very cluttered, they yielded duplicated, unreliable matchings. This catalog matching process yields two disjoint datasets: X_u, composed of objects that are unlabeled, and X, composed of objects that are labeled with the spectroscopic classes (star, galaxy, quasar) found in SDSS. Afterwards, the unlabeled dataset is more heavily filtered by removing all objects with magnitude uncertainties higher than 0.05, since high uncertainties would yield noisy targets for the pretext task (magnitude regression). We did experiment with datasets including objects with larger uncertainties, but noted that model convergence was harder, and model performance was poorer.

Both X_u and X are split into train, validation and test sets. Table 4.1 shows the number of samples of each set. X_u splits are stratified by r-magnitude intervals (of 1 unit), and X splits are stratified both by class and by r-magnitude intervals. This guarantees that magnitude and class distributions are similar across splits in each dataset. Figure 4.1 shows the r-magnitude

[2]The S-PLUS catalog comes with a `photoflag` column whose values are bit flags corresponding to problems that occured during the process of computing attributes for an object. For instance, an object whose attributes are likely to be biased by neighbouring objects (flag 1) and that has at least one saturated pixel (flag 4) would yield `photoflag=5`. Such problem codes are generated by SExtractor [BA96].

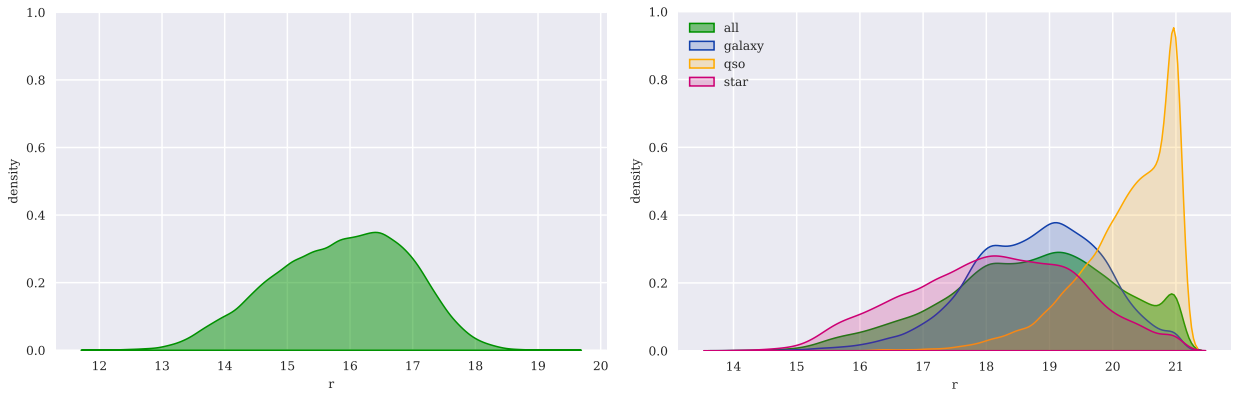


Figure 4.1: Normalized distributions of r-magnitude values of the X_u (left) and X (right) datasets.

Table 4.2: Number of samples per class in labeled dataset X.

	# samples	%
star	43881	43.7
galaxy	42158	41.9
quasar	14457	14.4

distribution in X_u and X. Note that r-magnitude distributions do vary between X_u and X: the mean value of r-magnitude is 15.9 in X_u and 18.7 in X. This difference occurs due to the fact that we adopted more conservative data quality criteria for X_u (i.e. filtering out all objects with magnitude uncertainties higher than 0.05). Also, note that quasars do tend to have higher magnitudes (meaning that they tend to be less bright), so the quasar class is expected to be harder to distinguish from others.

Moreover, as shown in Table 4.2, classes in X are imbalanced: the quasar class has significantly less samples than star and galaxy classes. To handle this imbalance, we modify the loss function of the classifiers, adding class weights which are inversely proportional to each class prevalence.

Image preprocessing

After having defined the subset of objects from S-PLUS DR1 to be used in experiments, separate images for each object must be cropped from 9232x9216 px sky fields. The full-width half-maximum (FWHM), which approximates the sizes of the objects, and the pixel coordinates available in the catalogs were used to generate such images. Based on conversations with astronomers, visual inspection, and analysis of the distribution of FWHM values, we concluded that squared crops of side equal to $3FWHM$ were adequate for capturing relevant information about the object and its context while avoiding cluttering. Figure 4.2 shows the distribution of FWHM values in X_u and X. 97.5% of all objects in X_u and X combined have $FWHM \leq 10px$, meaning that squared crops of 30px would be sufficient for the majority of the samples. We rounded this value up to the closest power of two.

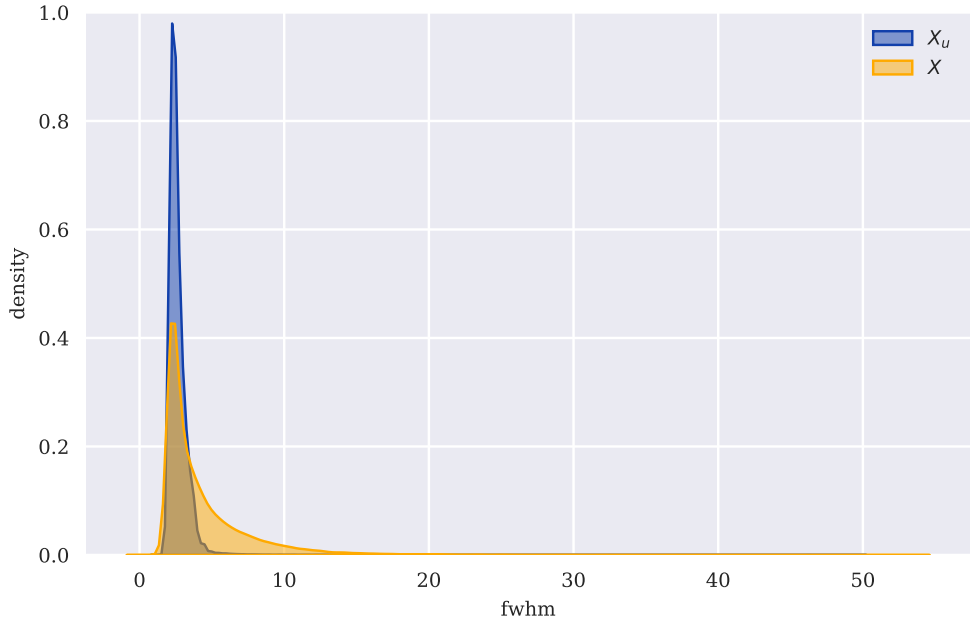
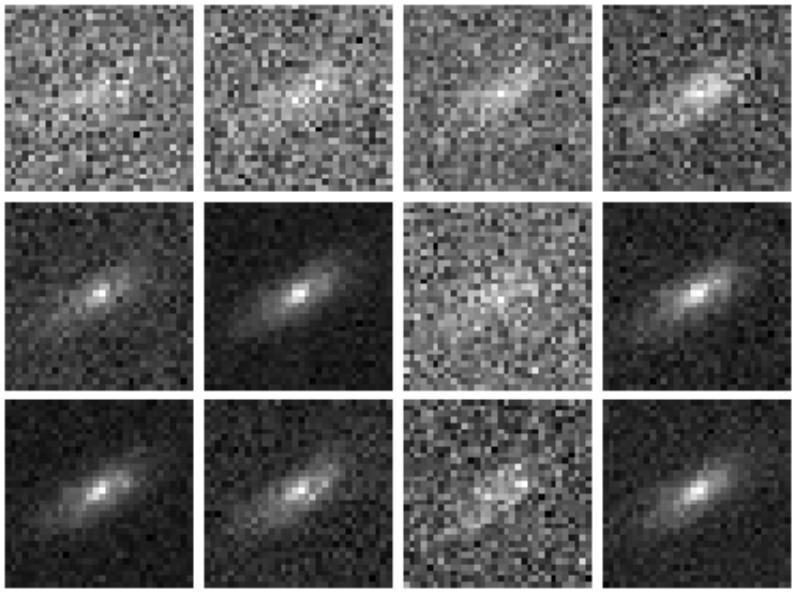
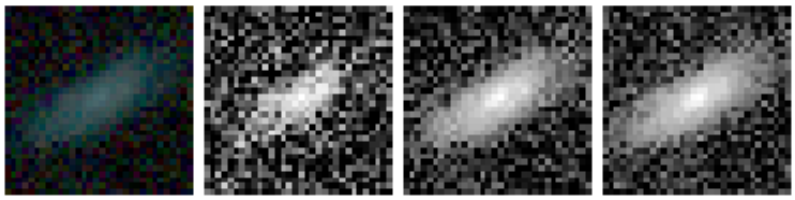


Figure 4.2: Normalized distributions of FWHM values (in pixels) of the X_u and X datasets. FWHM is used as a proxy for the size of the objects. In order to capture enough contextual information around objects, which are mostly point-like, $3FWHM$ is adopted as an estimate for their diameter.



(a) In twelve bands. From top-left to right-bottom: U, F378, F395, F410, F430, G, F515, R, F660, I, F861, Z.



(b) In RGB. From left to right: stacked RGB, R, G and B.

Figure 4.3: Image crop of a galaxy.

Our cropping procedure applies the following rule for each object: if $3FWHM <= 32$, generate a crop of 32 x 32 px around the center of the object, else generate a crop of side $3FWHM$ and resize it to 32 x 32px. This procedure yields standardized 32px squared crops. Note that only 3.5% of the objects are required to be downscaled, and those tend to be large objects with high signal-to-noise ratios (that is, easier examples). Also note that, for smaller objects, we give preference to generating fixed-size crops that keep more contextual information, instead of performing upscaling or padding, as the latter approaches would introduce unnecessary artifacts. We generate crops both from composite RGB images and from FITS images. Figure 4.3 shows an image crop of a galaxy in 12 bands and RGB, respectively. Note how, for the FITS image, the broad bands usually contain more signal. Also, note how the RGB image appears to be noisier, since it combines noisy signals from multiple bands into three bands.

For FITS images, we apply the calibration procedure defined by Equation 3.2. The objective of this calibration is to tranform raw pixel values, which correspond to counts of photons captured for that pixel position, into energy fluxes that have astrophysical meaning. Besides, this calibration procedure also yields values that are inside a more well-defined range compared to the original unbounded values, which contributes to more efficient model training. After calibration, the range of pixel values are [-0.79, 13.63] in X_u and [-1.13, 1.59] in X. X_u ended up having a broader value range, whose maximum value is one order of magnitude larger than the maximum value in X. This was expected due to the fact that X_u correponds to unknown objects of all sorts and thus tends to be more heterogeneous, whereas X corresponds to a select set of objects for which detailed spectroscopic information is available and thus tends to be more homogeneous. We could have filtered out objects with higher pixel values from X_u, which may be outliers, but we chose not to add this extra filtering step in our selection approach. In our experiments, we did not run into issues due to the differences in pixel ranges between X_u and X.

4.3 Implementation details

Neural network architecture

VGG [SZ14] was chosen as the CNN backbone for this work. Even though it might be perceived as a somewhat dated architecture, we chose to stick to VGG for three main reasons. First, in our paper [EMH20], we carried out an extensive search using various astronomical image datasets and CNN architectures, including state-of-the-art architectures such as Inception [SLJ+15], ResNeXt [XGD+17] and DenseNet [HLvdMW16], and found out that VGG yielded the best performance (measured by accuracy) in most cases. Second, VGG has a simple feed-forward backbone composed only of convolution and pooling operations, which results in faster training, allowing us to test a lot of scenarios quickly and iterate faster. It is worth noting that we used only the backbone of the VGG architecture, removing its

two dense layers (which are composed of 4096 units each), which indeed make the original architecture very heavy. We replaced this set of dense layers with a pooling layer followed by a 1024-unit dense layer and a final output layer. Third, since there are already a lot of scenarios that we want to analyze, we tried and kept the architecture and its hyperparameters fixed. We did experiment a bit with ResNeXt [XGD+17] and EfficientNet [TL19], which have interesting characteristics that we initially wanted to take advantage of. However, both backbones were considerably slower and harder to train on our proposed pretext task, which is a regression with up to 12 outputs. Thus, we chose to stick to VGG and focus on varying other "hyperparameters" in our problem, such as number of input channels and number of training samples.

Training procedure

Besides picking a CNN backbone, it is also important to define and stick to a consistent training procedure. There are tons of variations that can be tested when training, such as different optimizers, learning rate schedules, batch sizes, and data augmentation techniques. We did experiment with classical optimizers such as Stochastic Gradient Descent as well as state-of-the-art optimizers such as Adam [KB15] and RAdam [LJH+19]. We also experimented with learning rate decay, early stopping, and image augmentation via flipping and rotation with the Albumentations library [BPK+18]. We empirically found that the optimal training setup was heavily dependent on each scenario (e.g. stopping training when a monitored metric has not improved after n epochs would be adequate for one scenario, but would result in underfitting for other scenarios). If we chose to search for the optimal hyperparameters for each scenario, it would be quite difficult to keep track of all those moving parts, and also to compare results. Thus, after some exploration, we chose to keep things simple and fixed.

We found that Adam was better at handling varying scenarios, thus in all experiments, we use Adam with an initial learning rate of 10^{-4}, except for finetuning tasks, where we start with a slightly lower learning rate of 10^{-5}. We train all models for a large number of epochs – 100 for scenarios with large training sets (100k samples), and 300 for scenarios with small training sets ($\leq$ 2k samples) – and report results from the epoch whose validation error was the smallest. For scenarios that include finetuning, before training the full neural network, the top layers are trained for 10 epochs while keeping the backbone layers frozen. Each scenario is trained three times, to account for fluctuations in the optimization process itself, and also in the resources used.

Resource usage

Experiments were run in a single machine of the Computer Vision Research Group at IME-USP. This machine is equipped with four GeForce RTX 2070 GPUs, which were used to train different scenarios in parallel. The RTX 2070 comes with 7982 MiB of RAM, which

is not a lot. Because of that, and also because we are fond of efficient usage of resources, we implemented a few optimizations in our code.

First, we load our full train-validation set in one four-dimensional NumPy array of shape `(n_samples, im_side, im_side, n_channels)`. Having the full dataset in one large file instead of multiple small files reduces metadata size and I/O use, besides making it easier to version it. Second, we cast this four-dimensional array as single-precision floating points (float32), and also make sure TensorFlow is configured to use float32 precision. This halves memory usage when compared to double-precision floating points (float64), and also makes calculations faster, at the cost of a (probably negligible) loss in precision. Since we are not dealing with highly precise data that could benefit from double-precision representations, sticking to float32 is quite reasonable. If needed, precision could be cut down even further to float16, but this is usually not recommended due to numerical instability. Instead, TensorFlow documentation recommends mixed (float32 + float16) precision. Third, and last, we configure TensorFlow to allocate GPU memory dynamically, instead of allocating all memory from all avaiable GPUs for a single process. It can be done either via the TensorFlow Python API or via environment variables. This enables running one process per GPU, or even running multiple processes in one GPU, and is specially useful when working with shared resources, which was our case.

Code

Code written to run experiments is openly available at https://github.com/amartinazzo/label-the-sky and may be used to reproduce results. The code is structured in loosely coupled modules for preprocessing, training, and postprocessing, each containing their own classes and helper functions, and scripts that use such modules for building and running preprocessing and training pipelines. With this structure, code may be easily extended for new scenarios and use cases, some of which listed in Chapter 6.

Chapter 5

Experimental Results

In this chapter, experimental results obtained from 3, 5 and 12-channel models are presented. First, in Section 5.1, we present results for the pretraining task, wherein models receive 3, 5 or 12-channel images of astronomical objects as inputs, and predict magnitude values of such objects. Then, in Section 5.2, we present results for the downstream task, which is classification of stars, galaxies and quasars. Various scenarios are considered for training those classifiers: with or without finetuning, using big or small datasets, using weights from ImageNet models (only for RGB classifiers) or from magnitude models which were obtained in the pretraining task. Besides considering all those scenarios, results are also compared and analyzed considering the effect of the number of channels, and of the magnitude and size of the objects. Finally, in Section 5.3, results are discussed.

5.1 Pretraining

Our pretraining task consists of learning magnitudes from images. For 12-channel images, which are raw images, and 3-channel images, which are RGB composites generated from raw images, twelve magnitudes (one per photometric filter) are used as targets. For 5-channel images, whose channels correspond to broad band filters, only the corresponding five magnitudes are used as targets. In order to ease training, magnitude values are reescaled by dividing them by 35, which is slightly larger than the largest magnitude value found in X_u. Thus, after rescaling, all magnitude values are within the [0, 1] range. Then, when running predictions, magnitudes are scaled back to their original values by multiplying the outputs of the model by 35.

Figure 5.1 shows how validation errors decreased during training. It is noticeable that the 12-channel model converged faster and achieved the lowest error. Table 5.1 shows the values of such validation errors. Note that the 12-channel model yields the best result and achieves a MAE of 0.049, 25% lower than the 3-channel model, which is the second best. Also note that MAE achieved by the 5-channel model is nearly 3 times the MAE of the 12-channel model. This large difference between the best and worse results corroborates the idea that

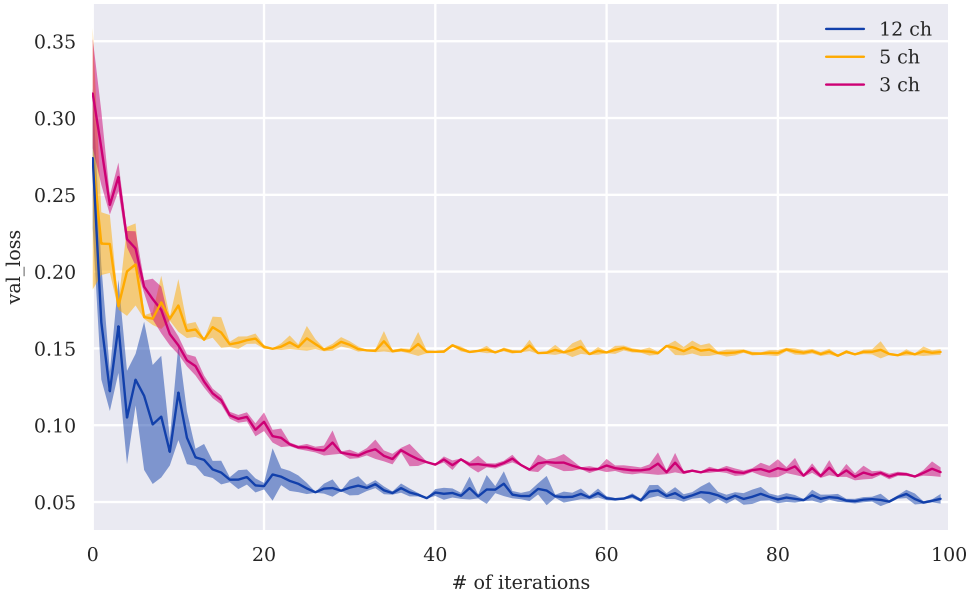


Figure 5.1: Validation loss curves for 12, 5 and 3-channel regression models. The 12-channel model converged faster and achieved the lowest loss.

Table 5.1: Validation errors for 12, 5 and 3-channel regression models.

# channels	mean absolute error
03	0.0655 ± 0.0014
05	0.1449 ± 0.0002
12	0.0490 ± 0.0003

using all twelve channels available helps the model learn better mappings from images to magnitudes, even if it makes the model slightly more complex (that is, increases the number of parameters of the model).

In Chapter 3, we explained that photometric filters of different wavelength intervals (e.g. ultraviolet, green, red, and so on) have various sensitivities, that is, they differ in how capable and effective they are in collecting photons. We also showed that each combination of object and photometric band yields a different uncertainty measure for its corresponding magnitude value. For that reason, besides evaluating the overall mean absolute error, it is worth evaluating how the error varies with the photometric band. Table 5.2 contains a comparison between the mean absolute error and the mean uncertainty for each photometric band, computed from the test set of X_u. The mean uncertainty can be understood as a measure of how noisy the targets we want to learn in this task are, and thus can be used as a proxy for the lowest error that can be achieved. Note how the three bands with largest uncertainties (in descending order: f395, u, f378) also yielded the largest mean absolute errors.

We also make use of t-SNE projections as a way to visually assess the features learned by the regression models. Figure 5.2 shows projections of features extracted from the test set of X_u colored by r-magnitude. We did experiment with various hyperparameters (e.g. perplexity, number of neighbours) and similar patterns emerged in all cases. 12 and 3-channel

Table 5.2: Test errors for each photometric filter of the 12-channel regression model, compared to the mean uncertainty for each filter. Filters with larger uncertainties yielded larger errors.

channel	MAE	mean uncertainty
u	0.0694	0.0239
f378	0.0716	0.0263
f395	0.0577	0.0309
f410	0.0478	0.0221
f430	0.0495	0.0212
g	0.0477	0.0200
f515	0.0478	0.0200
r	0.0432	0.0200
f660	0.0436	0.0200
i	0.0403	0.0200
f861	0.0496	0.0200
z	0.0459	0.0200

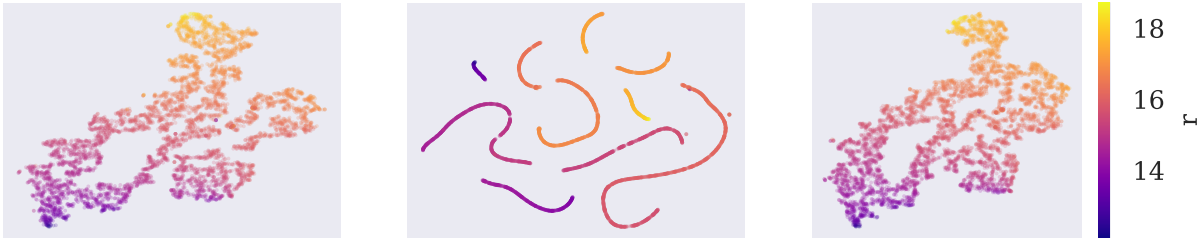


Figure 5.2: Projections of features extracted from the test set of X_u using 12, 5 and 3-channel models, colored by r-magnitude. They were generated using t-SNE with perplexity=50.

feature projections present similar well-spread structures, whereas the 5-channel presents worm-like structures. This may be an indication that, for the 5-channel model, t-SNE was not able to capture global structure, and instead focused on local structure, which is more likely to be noise (i.e. not statistically relevant). This is in agreement with the fact that the 5-channel regression model yielded a validation MAE that is a order of magnitude larger than the other models.

5.2 Classification

In order to assess the quality of the representations learned by the pretrained models presented in Section 5.1, classifiers were built employing such models as backbones. Our classification task consists of distinguishing stars, galaxies and quasars. We consider the following training scenarios:

- pretrained models can be used to train classifiers either with finetuning (backbone weights are updated) or without finetuning (backbone weights are frozen; pretrained model is used as a feature extractor)
- classifiers can be trained using either the entire dataset X (big data scenario) or small subsets of X (small data scenario)
- RGB classifiers can use either magnitude weights or ImageNet [RDS+14] weights

Thus, in total, we train and compare 16 different scenarios (4 for 12-channel models, 4 for 5-channel models, and 8 for 3-channel models). In scenarios in which the models are finetuned, training is carried out as detailed in Section 4.3: first, backbone layers are kept frozen and top layers are trained for 10 epochs, and then all layers are unfrozen and trained for 100 epochs. On the other hand, in scenarios in which models are not finetuned, the backbone is kept fixed and only the top layers are trained.

Besides feature extraction versus finetuning experiments, we also run small data experiments. The intuit of such experiments is to verify whether it would be feasible to use small datasets curated by experts in order to adapt pretrained models to various supervised tasks. We sample subsets of increasing size (where each subset is contained in the following, larger subset), starting at a subset of 100 samples and ending at a subset of 2k samples, using an incrementing step of 100 samples. Thus, in total, 20 subsets are used, and models are trained 20 times for each scenario. Random seeds are fixed to guarantee that the same subsets are used across different scenarios. With data from these 20 runs, curves of accuracy (or any other collected metric) as a function of the size of the training set can be built.

We separate this section in three subsections as follows. In the first subsection, we evaluate how 3-channel models that use magnitude weights compare to models that use ImageNet weights. In the second subsection, we evaluate how model performance changes with the

Table 5.3: Validation accuracies for 3-channel classifiers, pretrained either on ImageNet or magnitudes, with or without finetuning.

	without finetuning	with finetuning
ImageNet	0.8668 ± 0.0012	0.9156 ± 0.0010
magnitudes	0.7011 ± 0.0010	0.9138 ± 0.0010

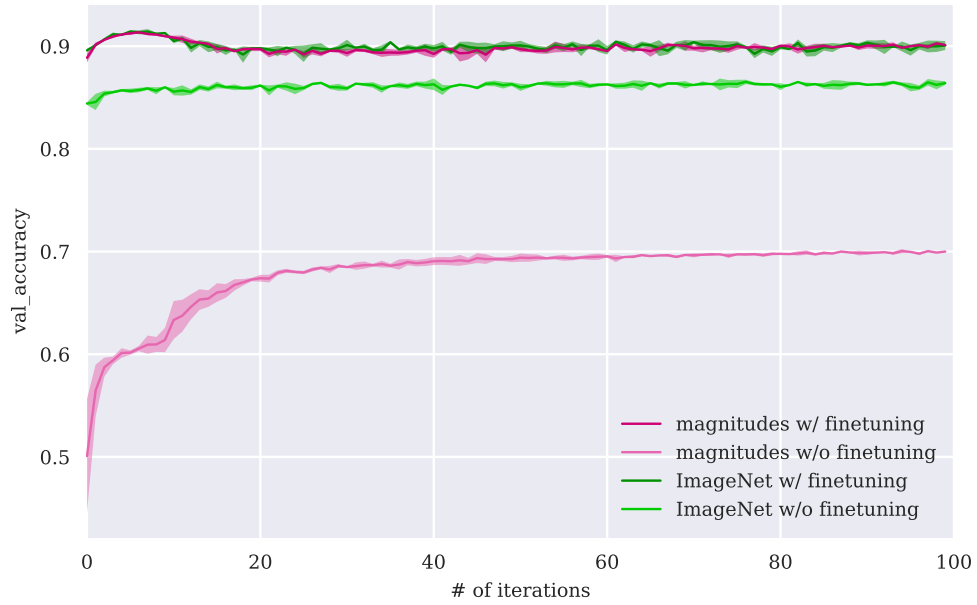


Figure 5.3: Validation accuracy curves for 3-channel classifiers, pre-trained either on ImageNet or magnitudes, with or without finetuning. Magnitudes and ImageNet classifiers achieve similar accuracies when trained with finetuning.

number of channels of the inputs images. Finally, in the third subsection, we evaluate how model performance changes with the magnitude and the size of the objects. In each subsection, big data and small data scenarios are tackled.

Comparison to benchmark RGB model

In this set of experiments, we use ImageNet [RDS+14] models as a benchmark, and compare 3-channel models pretrained either on ImageNet or magnitudes. The objective of such experiments is to verify that our proposal of pretraining models using magnitudes yield performances that are at least comparable to a benchmark model. As shown in Table 5.3, when finetuning is not used (that is, when pretrained models are used as feature extractors), the ImageNet classifier yields a significantly higher accuracy, indicating that features learned by the magnitude regression model are not as universal as features learned by the ImageNet model. On the other hand, when finetuning is adopted, ImageNet and magnitudes models yield equivalent validation accuracies. Figure 5.3 also shows that ImageNet and magnitude models trained with finetuning yield equivalent validation accuracies.

Besides, when using small training sets, the ImageNet classifier also yields higher validation accuracies, as shown in Figure 5.4. As the size of the training set approaches 2k samples,

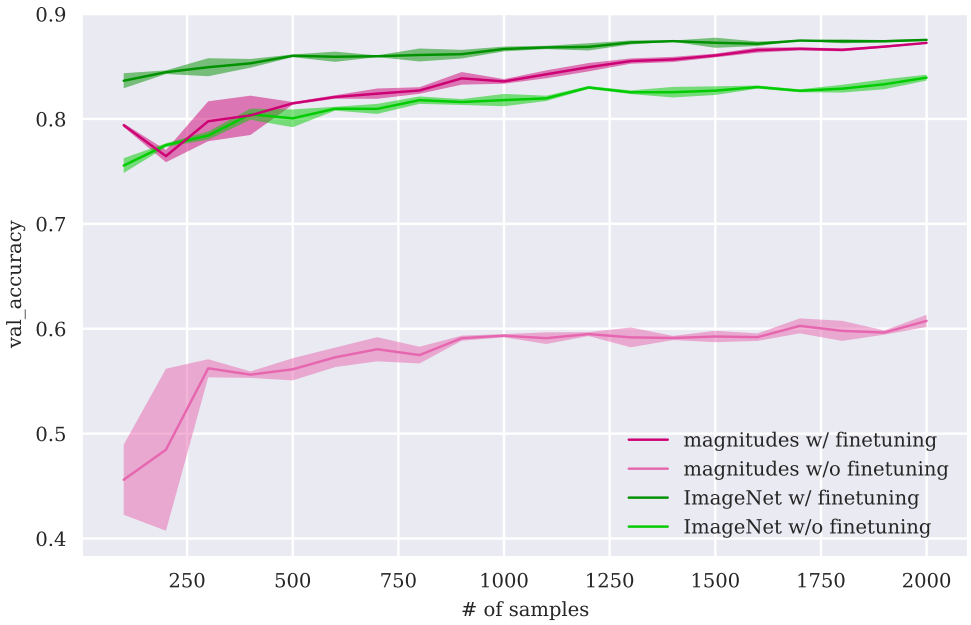


Figure 5.4: Validation accuracy as a function of the size of the training set for 3-channel classifiers, pre-trained either on ImageNet or magnitudes, with or without finetuning. The ImageNet classifiers seem to achieve better accuracies for very small training sets, but the magnitude model with finetuning quickly catches up as the size of the training set increases.

the difference in accuracy between ImageNet and magnitude classifiers quickly vanishes and approaches the performance of the classifiers trained on the entire dataset. This indicates that it would indeed be feasible to achieve good performance with small datasets for star-galaxy-quasar classification.

Effect of the number of channels

In this set of experiments, we compare 12, 5 and 3-channel models, all pretrained on magnitudes. The objective of such experiments is to analyze how the number of channels affects performance on the star-galaxy-quasar classification task. Table 5.4 shows validation accuracies for such scenarios. It can be seen that, when training without finetuning (that is, using the pretrained model as a feature extractor), the 3-channel model yields the best accuracy, followed by the 12-channel model. This may be due to two reasons: first, the 3-channel input images have less complex pixel values (they are standard RGB images, which are represented as uint8 values), and second, the 3-channel model is slightly less complex (i.e. has fewer trainable parameters). Thus, it makes sense to assume that the mapping from features to classes in the 3-channel model is also less complex, and easier to learn. On the other hand, when models are trained with finetuning, the 12-channel model yields the best accuracy, and the 5 and 3-channel models yield equivalent accuracies. This can also be seen in Figure 5.5.

When training with small subsets of X, similar patterns emerge, as shown in Figure 5.6. It is interesting to note that smaller training sets seem to have larger standard deviations

Table 5.4: Validation accuracies for 12, 5 and 3-channel classifiers, pretrained on magnitudes, with or without finetuning.

	without finetuning	with finetuning
3 channels	0.7011 ± 0.0010	0.9138 ± 0.0010
5 channels	0.5436 ± 0.0008	0.9130 ± 0.0010
12 channels	0.6859 ± 0.0002	0.9378 ± 0.0018

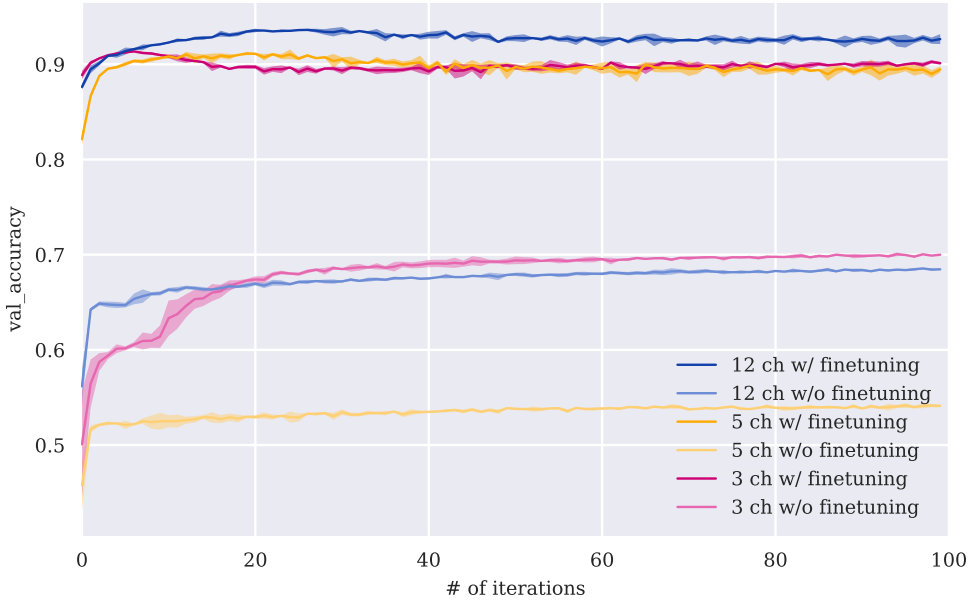


Figure 5.5: Validation accuracy curves for 12, 5 and 3-channel classifiers, pre-trained on magnitudes, with or without finetuning. The 12-channel classifier converged faster and also achieved the highest accuracy.

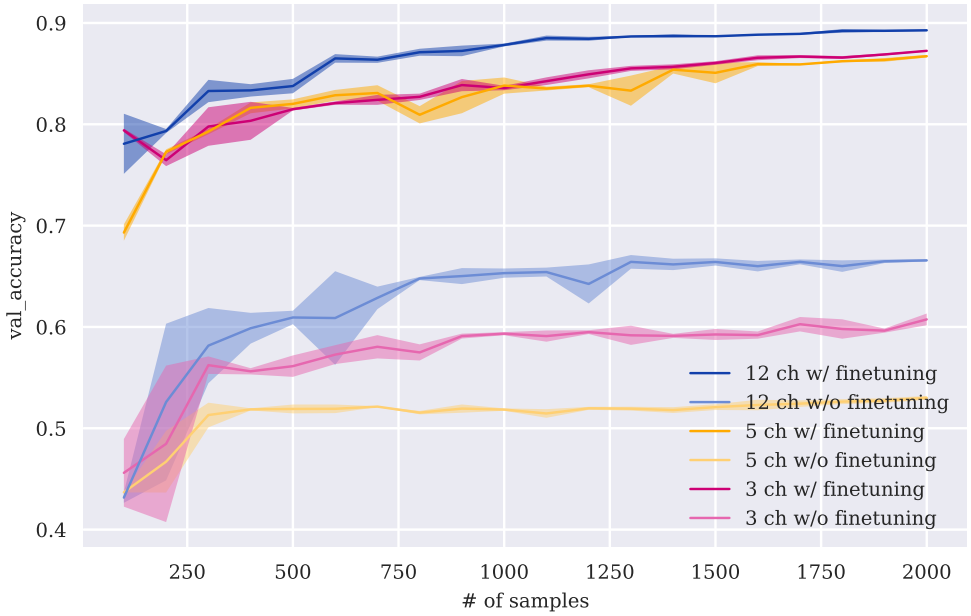


Figure 5.6: Validation accuracy as a function of the size of the training set for 12, 5 and 3-channel classifiers, pre-trained on magnitudes, with or without finetuning.

Table 5.5: Test accuracies for 3 and 12-channel classifiers, pretrained on magnitudes, with finetuning.

	accuracy
3 channels	0.9095
12 channels	0.9367

Table 5.6: Test metrics for each class of the 12-channel (3-channel) classifier, pretrained on magnitudes, with finetuning.

	precision	recall
galaxy	0.9647 (0.9531)	0.9341 (0.9545)
star	0.9496 (0.9230)	0.9540 (0.9120)
quasar	0.8269 (0.7466)	0.8921 (0.7704)

across training runs (representated by the shaded area around each curve).

The 5-channel models yielded the worst results in all cases, including the pretraining task. This indicates that the presence of the narrow-band filters is indeed important for generating better representations of the astronomical objects.

Table 5.6 shows more fine-grained performance metrics for 12 and 3-channel classifiers with finetuning. It can be seen that performance for the quasar class is significantly better when using 12-channel images: precision is 8.03pp higher, and recall is 12.16pp higher. Table 5.7 further details performance differentes between 12 and 3-channel classifiers. It shows that 17% of the quasar samples in the test set were incorrectly classified as stars by the 3-channel model, whereas only 6% were incorrectly classified as stars by the 12-channel model. This indicates that using raw images instead of RGB composite images helps the model encode more information about the harder and less prevalent class, which is the quasar class.

Besides that, it is worth noting that, because accuracies in Table 5.5 are computed using the test set, values are slightly lower than the values reported in Table 5.4, which is computed using the validation set. Test accuracy is 0.4pp lower for the 3-channel model and 0.1pp lower for the 12-channel model when compared to validation accuracy. This may be an indication that the 12-channel model generalizes better.

We also make use of t-SNE projections in order to visualize features extracted from finetuned models. Figure 5.7 shows such projections, extracted from the test set of X and colored by class. It can be seen that 12 and 3-channel models yielded more nicely separated

Table 5.7: Normalized confusion matrix for 12-channel (3-channel) classifier, pretrained on magnitudes, with finetuning. Metrics were computed on the test set.

		predicted class		
		galaxy	star	quasar
true class	galaxy	0.93 (0.95)	0.03 (0.02)	0.03 (0.02)
	star	0.02 (0.03)	0.95 (0.91)	0.03 (0.06)
	quasar	0.05 (0.06)	0.06 (0.17)	0.89 (0.77)

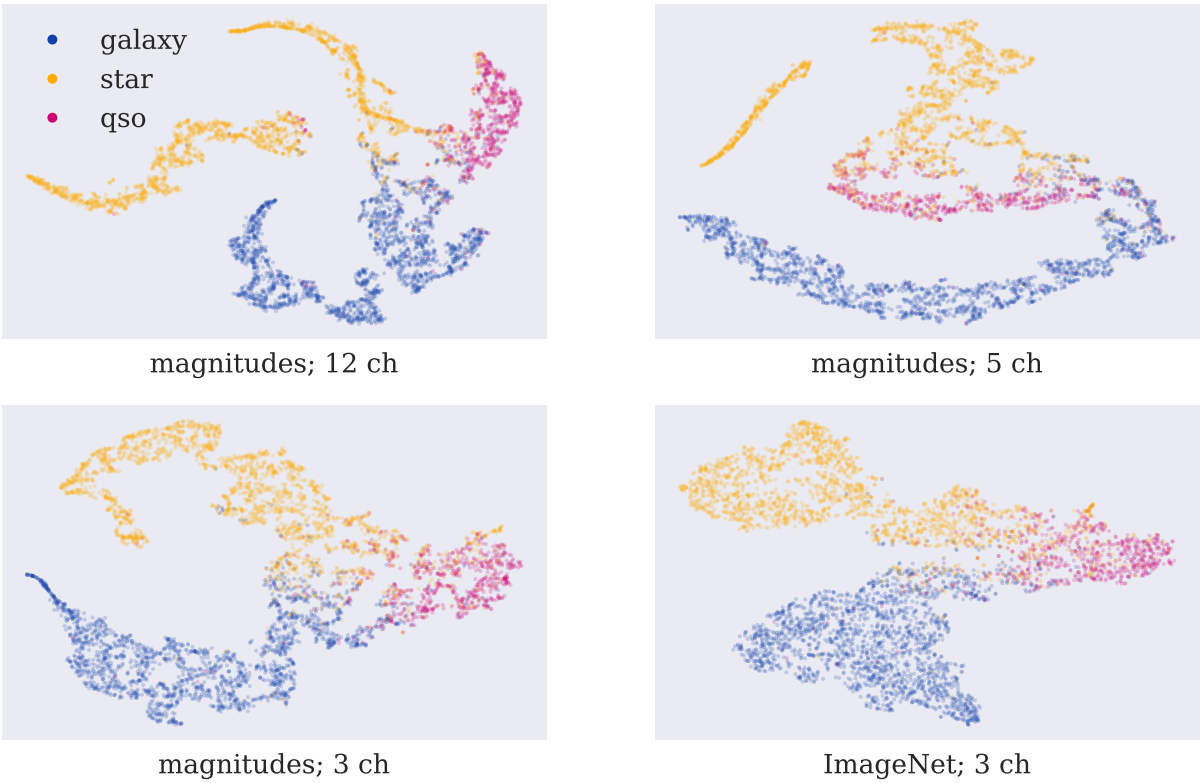


Figure 5.7: Projections of features extracted from the test set of X using 12, 5 and 3-channel models finetuned either from ImageNet or from magnitudes. Projections were generated using t-SNE with perplexity=50. The 12-channel model does seem to generate more well-separated clusters.

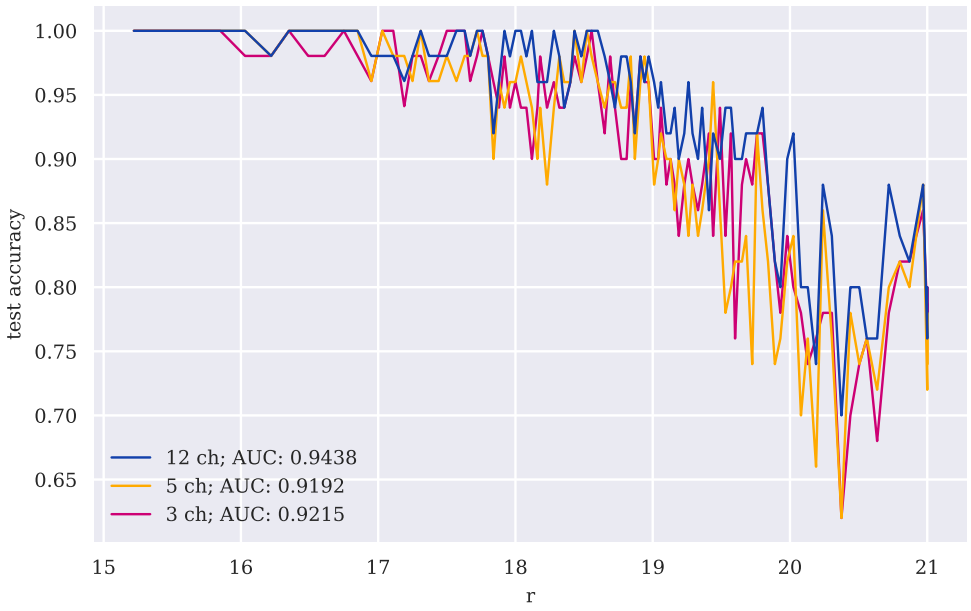


Figure 5.8: Test accuracy as a function of r-magnitude for 12, 5 and 3-channel classifiers, pre-trained on magnitudes, with finetuning.

groups of objects.

Effect of the magnitude and size of the objects

In this set of experiments, we evaluate how model performance changes with the brightness (magnitude) and size (FWHM) of the objects. In order to do that, we take the test set of X, order it by the attribute we want to evaluate (r-magnitude or FWHM) and slice it into evenly sized subsets of 100 objects each. Then, accuracies (or any other metrics) for each subset are computed. We chose to use subsets of equal size so that metrics computed from them could be compared in a fairer manner. With such data, curves of accuracy as a function of a given attribute can be generated. As a way to quantitatively compare such curves, we also compute the area under the curves (AUC) using the trapezoidal rule. We normalize AUC values so that they fall within the [0, 1] range.

Figure 5.8 shows how test accuracies vary with the brightness of the objects, represented by r-magnitude. It can be seen that accuracy is higher for objects with lower r-magnitude (which are brighter), and drops as r-magnitude increases. Also, it is noticeable that the 12-channel model yield the best accuracies across r-magnitude values. This is reinforced by the fact that the AUC is also larger for the 12-channel model, followed by the 3-channel and then the 5-channel models. Such results are in agreement with results shown in the previous subsection.

Figure 5.9 shows how test accuracies vary with the size of the objects, represented by FWHM. Since our procedure for slicing the dataset is to slice it into evenly sized subsets (instead of subsets whose attribute intervals are evenly spaced), curves look more cluttered for lower FWHM values, which are more prevalent. It can be seen that larger FWHM values

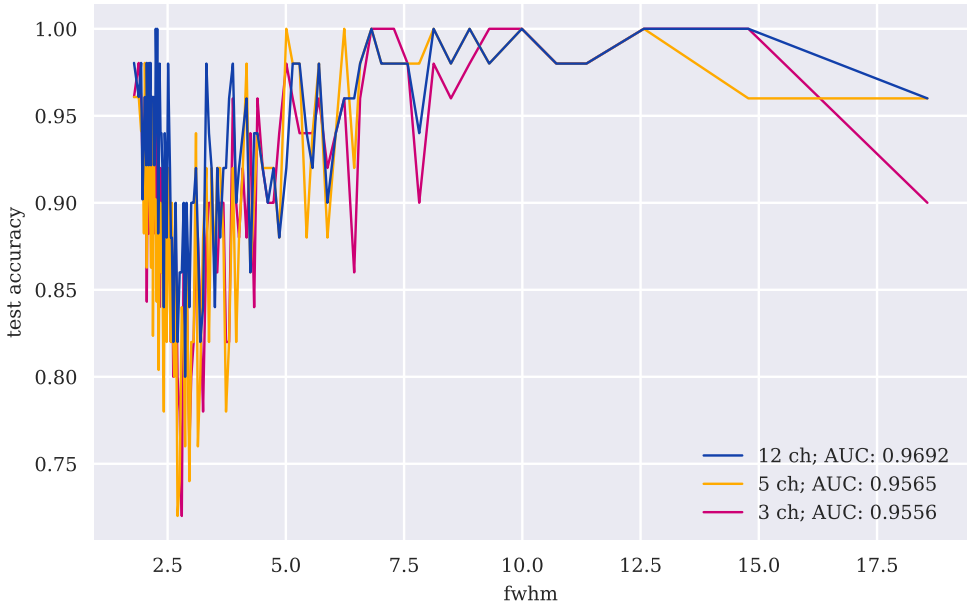


Figure 5.9: Test accuracy as a function of FWHM for 12, 5 and 3-channel classifiers, pretrained on magnitudes, with finetuning.

yield better accuracies. It can also be seen that the 12-channel model again yields the best accuracies and the best AUC. 5-channel and 3-channel have almost equivalent AUC values. This may be explained by the fact that the 3-channel images combine multiple channels into one and thus tend to look noisier, as shown in Figure 4.3.

5.3 Discussion

In this chapter, experimental results for the pretraining and classification tasks proposed in Section 4.1 were presented. In Section 5.1, we demonstrated that it is possible to train CNNs to predict magnitudes with a reasonably low error. These pretrained CNNs can be used for estimating magnitudes of new astronomical objects, but they can also be used for extracting features. It is worth noting that, even though magnitude distributions were different for the pretraining dataset X_u and the classification dataset X (due to reasons which were detailed in Section 4.2), our proposed self-supervised approach worked well. This is shown in Section 5.2, where we trained classifiers in various scenarios: without/with finetuning, using small/big datasets.

We found that pretrained models only need a few hundreds of samples in order to achieve performances comparable to accuracies of the models trained on the complete dataset. Given that context, it is quite feasible to select a small, representative sample of objects to be labeled by domain experts in order to get a finetuned model that would then enable object classification at scale. Moreover, we showed that the performance of the classifiers vary with the brightness and size of the objects (small, faint objects are harder to classify), and that 12-channel classifiers are more robust to changes in brightness and size. Overall, 12-channel

Table 5.8: Comparison of metrics from our 12-channel DL classifier (based on FITS images) and a tree-based classifier (based on tabular data) proposed by Nakazono et el [NdOH+21]. Metrics from the tree-based classifier are in parenthesis. Both classifiers were trained and evaluated using only S-PLUS data as inputs, but datasets are different: our DL classifier uses DR1, and the tree-based classifier uses DR2. Moreover, different filtering and split criteria were used in each case. More specifically, Nakazono et al did not need to filter out objects with missing magnitude values.

	precision	recall
galaxy	0.9647 (0.8139)	0.9341 (0.8537)
star	0.9496 (0.9817)	0.9540 (0.7856)
quasar	0.8269 (0.5247)	0.8921 (0.9224)

models yielded the best results, both for the pretext and the classification tasks.

Besides leveraging unlabeled data for learning representations from astronomical images, another motivation for this work was to find out whether using astronomical data in an unstructured, raw format (i.e. FITS images instead of tables that are computed from FITS images using domain knowledge) in ML models could yield better performance. In order to answer this question, in Table 5.8, we put metrics from our best classifier alongside metrics reported by Nakazono et al [NdOH+21][1]. It is worth noting that results reported by Nakazono et al are not directly comparable to ours, since different datasets and different filtering criteria were used, and are included mainly as a reference for discussion.

One of the objectives of [NdOH+21] was to produce a reliable catalog of galaxies, stars, and quasars in S-PLUS DR2 [AFSH+21]. Their approach in order to achieve that was to train tree-based classifiers using the twelve magnitudes available in the S-PLUS catalog as features, plus four morphological features (such as FWHM) that are also available in the catalog, and SDSS spectroscopic classes as labels. It can be seen in Table 5.8 that our approach yields better results in most of the cases, sometimes by a very large margin: precision of the quasar class is 30.22pp higher, and recall of the star class is 16.84pp higher. These large differences corroborate our idea that letting the models learn from data in a more raw format likely yields better performance. On the other hand, the tree-based model yielded better results in two cases (i.e. precision of the star class is 3.21pp higher, and recall of the quasar class is 3.03pp higher). This shows that a less complex model, which requires less data and compute resources, is usually a good baseline, which is in agreement with Occam's razor[2].

In order to boost overall performance of the tree-based classifier, authors proposed including additional features by matching S-PLUS to a third sky survey, WISE [WEM+10], which contains information that is important for distinguishing quasars. A drawback of this proposal is that it adds another external data dependence and results in a smaller dataset, given by the overlap of objects available in S-PLUS, WISE and SDSS. Given that context, our DL approach gives researchers more flexibility to explore each sky survey independently,

[1]Nakazono et al [NdOH+21] reported completeness and purity values, which are commonly used as classification metrics in Astronomy. Completeness corresponds to recall, and purity corresponds to precision.

[2]Occam's razor suggests picking simpler ML models, since they are expected to generalize better.

since it depends on only one external data source, which are spectroscopic labels from SDSS.

Chapter 6

Conclusions

In this work, we explored approaches for extracting representations from multichannel astronomical images using deep convolutional neural networks. We began by experimenting with training CNNs using 12-channel images in [MH19], then we went on to carry out a comparative study of CNN models applied to various astronomical object classification tasks in [EMH20], and finally proposed a self-supervised learning approach for pretraining CNNs with unlabeled data in [MEH20].

We sumarize below how we succeeded in answering the three research questions defined in Section 1.2:

(i) Can astronomical properties be learned from images?

We proposed a training procedure where models receive images as inputs, and learn to predict numerical astronomical properties. In Section 5.1, we show that this training procedure yields reasonably low validation errors and can be used as a pretraining step.

(ii) Can FITS images yield more information than RGB images?

We trained models using both FITS and RGB images as inputs, and compared them. In Section 5.1, we show that 12-channel models yield better results for regression of magnitudes. Likewise, in Section 5.2, we show that 12-channel models also yield better results for star-galaxy-quasar classification.

(iii) Can reasonable representations that do not depend on labels be learned?

As mentioned in question (i), we proposed a training procedure that uses images as inputs and astronomical properties as outputs. This training procedure produces pretrained models that can be fine-tuned for various tasks, but that can also be used as feature extractors. In Section 5.1, we show that pretrained models generate representations that can be useful for exploratory analyzes and unsupervised tasks.

As outlined in questions (i) and (iii), the core contribution of this work is the proposal of a self-supervised learning approach that leverages astronomical properties for training

deep convolutional neural networks with unlabeled astronomical images. Another important contribution is a preprocessing method that produces DL-ready image crops. We demonstrated that our end-to-end method, including preprocessing and training pipelines, works well by applying them to a dataset from the S-PLUS DR1 [ORS+19]. Besides that, we also demonstrated that training object classifiers from astronomical images instead of astronomical properties in table format (which are computed from images) yields better performance.

We believe that self-supervised learning is a powerful approach for tackling problems not only in Astronomy, but also in other fields of study where domain knowledge may inspire novel self-supervised approaches. We expect that our work contributes to the advancement of discussions on self-supervised learning in the ML community.

This research work opened up a various possibilities for future work. We outline some possible directions below.

Different astronomical properties as targets. A straightforward extension of this work would be a comparative study of using different astronomical properties as targets for the pretext task. More specifically, knowing that astronomical objects that are close to each other frequently share similar histories and characteristics, it would certainly be enriching to consider information about the location of the object in space, given by latitude, longitude and redshift.

More diverse downstream tasks. The task of classifying stars, galaxies and galaxies was chosen as the downstream task for evaluating the learned representations. A possibility for extending this work would be evaluating the learned representations on other downstream tasks of varying degrees of difficulty, including unsupervised tasks such as clustering and anomaly detection. We believe that such unsupervised approaches could be very useful for discovering rare objects.

Larger pretraining dataset. Low quality data, such as samples with missing values or very noisy targets, are filtered out from the pretraining dataset in this work, which produces differences between the distributions of the pretraining and the downstream datasets. An interesting direction for future work would be to investigate techniques that can be employed in order to treat such low quality samples and keep them in the dataset while avoiding performance decay. This would yield a pretraining dataset whose distribution would likely be closer to the distributions of the downstream datasets.

Uncertainty estimation. Yet another interesting extension of this work would be somehow including the uncertainties associated with the astronomical properties in the regression model. Such uncertainties could be included when training, but also when running inferences. Uncertainty in deep learning [GTA+21, RAH21] is a nascent but promising research field, and it would certainly be important to provide uncertainty estimates along with property estimates, specially if such regression models are to be used and trusted by domain experts.

Bibliography

[AAA+18] D. S. Aguado, R. Ahumada, A. Almeida, S. F. Anderson, B. H. Andrews, B. Anguiano, E. A. Ortiz, A. Aragon-Salamanca, M. Argudo-Fernandez, M. Aubert, V. Avila-Reese, C. Badenes, S. B. Rembold, K. Barger, J. Barrera-Ballesteros, D. Bates, J. Bautista, R. L. Beaton, T. C. Beers, F. Belfiore, M. Bernardi, M. Bershady, F. Beutler, J. Bird, D. Bizyaev, G. A. Blanc, M. R. Blanton, M. Blomqvist, A. S. Bolton, M. Boquien, J. Borissova, J. Bovy, W. N. Brandt, J. Brinkmann, J. R. Brownstein, K. Bundy, A. Burgasser, N. Byler, M. C. Diaz, M. Cappellari, R. Carrera, B. C. Sodi, Y. Chen, B. Cherinka, P. D. Choi, H. Chung, D. Coffey, J. M. Comerford, J. Comparat, K. Covey, G. da Silva Ilha, L. da Costa, Y. S. Dai, G. Damke, J. Darling, R. Davies, K. Dawson, V. de Sainte Agathe, A. D. Machado, A. D. Moro, N. D. Lee, A. M. Diamond-Stanic, H. D. Sanchez, J. Donor, N. Drory, H. du Mas des Bourboux, C. Duckworth, T. Dwelly, G. Ebelke, E. Emsellem, S. Escoffier, J. G. Fernandez-Trincado, D. Feuillet, J.-L. Fischer, S. W. Fleming, A. Fraser-McKelvie, G. Freischlad, P. M. Frinchaboy, H. Fu, L. Galbany, R. Garcia-Dias, D. A. Garcia-Hernandez, L. A. G. Oehmichen, M. A. G. Maia, H. Gil-Marin, K. Grabowski, M. Gu, H. Guo, J. Ha, E. Harrington, S. Hasselquist, C. R. Hayes, F. Hearty, H. H. Toledo, H. Hicks, D. W. Hogg, K. Holley-Bockelmann, J. A. Holtzman, B.-C. Hsieh, J. A. S. Hunt, H. S. Hwang, H. J. Ibarra-Medel, C. E. J. Angel, J. Johnson, A. Jones, H. Jonsson, K. Kinemuchi, J. Kollmeier, C. Krawczyk, K. Kreckel, S. Kruk, I. Lacerna, T.-W. Lan, R. R. Lane, D. R. Law, Y.-B. Lee, C. Li, J. Lian, L. Lin, Y.-T. Lin, C. Lintott, D. Long, P. Longa-Pena, J. T. Mackereth, A. de la Macorra, S. R. Majewski, O. Malanushenko, A. Manchado, C. Maraston, V. Mariappan, M. Marinelli, R. Marques-Chaves, T. Masseron, K. L. Masters, R. M. McDermid, N. M. Pena, S. Meneses-Goytia, A. Merloni, M. Merrifield, S. Meszaros, D. Minniti, R. Minsley, D. Muna, A. D. Myers, P. Nair, J. C. do Nascimento, J. A. Newman, C. Nitschelm, M. D. Olmstead, A. Oravetz, D. Oravetz, R. A. O. Minakata, Z. Pace, N. Padilla, P. A. Palicio, K. Pan, H.-A. Pan, T. Parikh, J. P. III, S. Peirani, S. Penny, W. J. Percival, I. Perez-Fournon, T. Peterken, M. Pinsonneault, A. Prakash, J. Raddick, A. Raichoor, R. A. Riffel, R. Riffel, H.-W. Rix, A. C. Robin, A. Roman-Lopes, B. Rose, A. J. Ross, G. Rossi, K. Rowlands, K. H. R. Rubin, S. F. Sanchez, J. R. Sanchez-Gallego, C. Sayres, A. Schaefer, R. P. Schiavon, J. S. Schimoia, E. Schlafly, D. Schlegel, D. Schneider, M. Schultheis, H.-J. Seo, S. J. Shamsi, Z. Shao, S. Shen, S. Shetty, G. Simonian, R. Smethurst, J. Sobeck, B. J. Souter, A. Spindler, D. V. Stark, K. G. Stassun, M. Steinmetz, T. Storchi-Bergmann, G. S. Stringfellow, G. Suarez, J. Sun, M. Taghizadeh-Popp, M. S. Talbot, J. Tayar, A. R. Thakar, D. Thomas, P. Tissera, R. Tojeiro, N. W. Troup, E. Unda-Sanzana, O. Valenzuela, M. V.-M. na, J. A. V. Mata, D. Wake, B. A. Weaver, A.-M. Weijmans, K. B. Westfall, V. Wild, J. Wilson, E. Woods,

R. Yan, M. Yang, O. Zamora, G. Zasowski, K. Zhang, Z. Zheng, Z. Zheng, G. Zhu, J. C. Zinn, and H. Zou. The fifteenth data release of the sloan digital sky surveys: First release of manga derived quantities, data visualization tools and stellar library, 2018, arXiv:1812.02759. doi:10.3847/1538-4365/aaf651.

[ABC+16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: A system for large-scale machine learning, 2016, arXiv:1605.08695.

[AC19] M. A. Aragon-Calvo. Self-supervised Learning with Physics-aware Neural Networks I: Galaxy Model Fitting, 2019, arXiv:1907.03957. doi:10.1093/mnras/staa2228.

[AFSH+21] F. Almeida-Fernandes, L. Sampedro, F. R. Herpich, A. Molino, C. E. Barbosa, M. L. Buzzo, R. A. Overzier, E. V. R. de Lima, L. M. I. Nakazono, G. B. O. Schwarz, H. D. Perottoni, G. F. Bolutavicius, L. A. Gutiérrez-Soto, T. Santos-Silva, A. Z. Vitorelli, A. Werle, D. D. Whitten, M. V. C. Duarte, C. R. Bom, P. Coelho, L. S. Jr., V. M. Placco, G. S. M. Teixeira, J. Alonso-García, T. C. Beers, A. Kanaan, T. Ribeiro, W. Schoenell, and C. M. de Oliveira. Data Release 2 of S-PLUS: accurate template-fitting based photometry covering ∼1000 square degrees in 12 optical filters, 2021, arXiv:2104.00020.

[BA96] E. Bertin and S. Arnouts. SExtractor: Software for source extraction. *Astronomy and Astrophysics Supplement Series*, 117:393–404, 1996. doi:10.1051/aas:1996164.

[Ber17] E. Bertin. SExtractor Documentation, 2017. URL: https://sextractor.readthedocs.io/_/downloads/en/des_dr1/pdf/.

[BPK+18] A. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin. Albumentations: fast and flexible image augmentations, 2018, arXiv:1809.06839. doi:10.3390/info11020125.

[Bur18] M. S. Burns. A Practical Guide to Observational Astronomy, 2018. URL: https://faculty1.coloradocollege.edu/~sburns/courses/18-19/pc362/PracticalObsAstro.pdf.

[CBJD18] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep Clustering for Unsupervised Learning of Visual Features. In *European Conference on Computer Vision*, 2018, arXiv:1807.05520.

[CDSM+19] M. V. Costa-Duarte, L. Sampedro, A. Molino, H. S. Xavier, F. R. Herpich, A. L. Chies-Santos, C. E. Barbosa, A. Cortesi, W. Schoenell, A. Kanaan, T. Ribeiro, C. M. de Oliveira, S. Akras, A. Alvarez-Candal, C. L. Barbosa, J. L. N. Castellón, P. Coelho, M. L. L. Dantas, R. Dupke, A. Ederoclite, A. Galarza, T. S. Gonçalves, J. A. Hernandez-Jimenez, Y. Jiménez-Teja, A. Lopes, P. A. A. Lopes, R. L. de Oliveira, J. L. M. de Azevedo, L. M. Nakazono, H. D. Perottoni, C. Queiroz, K. Saha, L. S. Jr., E. Telles, and R. C. T. de Souza. The s-plus: a star/galaxy classification based on a machine learning approach, 2019, arXiv:1909.08626.

[CG01] T. Chen and C. Guestrin. Random forests. *Machine Learning*, 45:5–32, 2001. doi:10.1023/A:1010933404324.

[CG16] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system, 2016, arXiv:1603.02754. doi:10.1145/2939672.2939785.

[Cho17] F. Chollet. Xception: Deep Learning with Depthwise Separable Convolutions. *Conference on Computer Vision and Pattern Recognition*, 2017. doi:10.1109/CVPR.2017.195.

[CMCH+19] A. J. Cenarro, M. Moles, D. Cristóbal-Hornillos, A. Marín-Franch, A. Ederoclite, J. Varela, C. López-Sanjuan, C. Hernández-Monteagudo, R. E. Angulo, H. V. Ramió, K. Viironen, S. Bonoli, A. A. Orsi, G. Hurier, I. S. Roman, N. Greisel, G. Vilella-Rojo, L. A. Díaz-García, R. Logroño-García, S. Gurung-López, D. Spinoso, D. Izquierdo-Villalba, J. A. L. Aguerri, C. A. Prieto, C. Bonatto, J. M. Carvano, A. L. Chies-Santos, S. Daflon, R. A. Dupke, J. Falcón-Barroso, D. R. Gonçalves, Y. Jiménez-Teja, A. Molino, V. M. Placco, E. Solano, D. D. Whitten, J. Abril, J. L. Antón, R. Bello, S. B. de Toledo, J. Castillo-Ramírez, S. Chueca, T. Civera, M. C. Díaz-Martín, M. Domínguez-Martínez, J. Garzarán-Calderaro, J. Hernández-Fuertes, R. Iglesias-Marzoa, C. Iñiguez, J. M. J. Ruiz, K. Kruuse, J. L. Lamadrid, N. Lasso-Cabrera, G. López-Alegre, A. López-Sainz, N. Maícas, A. Moreno-Signes, D. J. Muniesa, S. Rodríguez-Llano, F. Rueda-Teruel, S. Rueda-Teruel, I. Soriano-Laguía, V. Tilve, L. Valdivielso, A. Yanes-Díaz, J. S. Alcaniz, C. M. de Oliveira, L. Sodré, P. Coelho, R. L. de Oliveira, A. Tamm, H. S. Xavier, L. R. Abramo, S. Akras, E. J. Alfaro, A. Alvarez-Candal, B. Ascaso, M. A. Beasley, T. C. Beers, M. B. Fernandes, G. R. Bruzual, M. L. Buzzo, J. M. Carrasco, J. Cepa, A. Cortesi, M. V. Costa-Duarte, M. D. Prá, G. Favole, A. Galarza, L. Galbany, K. Garcia, R. M. G. Delgado, J. I. González-Serrano, L. A. Gutiérrez-Soto, J. A. Hernandez-Jimenez, A. Kanaan, H. Kuncarayakti, R. C. G. Landim, J. Laur, J. Licandro, G. B. L. Neto, J. D. Lyman, J. M. Apellániz, J. Miralda-Escudé, D. Morate, J. P. Nogueira-Cavalcante, P. M. Novais, M. Oncins, I. Oteo, R. A. Overzier, C. B. Pereira, A. Rebassa-Mansergas, R. R. R. Reis, F. Roig, M. Sako, N. Salvador-Rusiñol, L. Sampedro, P. Sánchez-Blázquez, W. A. Santos, L. Schmidtobreick, B. B. Siffert, and E. Telles. J-PLUS: The Javalambre Photometric Local Universe Survey. *Astronomy and Astrophysics*, 622:A176, 2019. doi:10.1051/0004-6361/201833036.

[Coe15] D. Coe. Trilogy: FITS image conversion software, 2015. URL: https://stsci.edu/~dcoe/trilogy.

[CPC16] A. Canziani, A. Paszke, and E. Culurciello. An Analysis of Deep Neural Network Models for Practical Applications, 2016, arXiv:1605.07678.

[CPWS+18] T. A. Collaboration, A. M. Price-Whelan, B. M. Sipocz, H. M. Günther, P. L. Lim, S. M. Crawford, S. Conseil, D. L. Shupe, M. W. Craig, N. Dencheva, A. Ginsburg, J. T. VanderPlas, L. D. Bradley, D. Pérez-Suárez, M. de Val-Borro, T. L. Aldcroft, K. L. Cruz, T. P. Robitaille, E. J. Tollerud, C. Ardelean, T. Babej, M. Bachetti, A. V. Bakanov, S. P. Bamford, G. Barentsen, P. Barmby, A. Baumbach, K. L. Berry, F. Biscani, M. Boquien, K. A. Bostroem, L. G. Bouma, G. B. Brammer, E. M. Bray, H. Breytenbach, H. Buddelmeijer,

D. J. Burke, G. Calderone, J. L. C. Rodríguez, M. Cara, J. V. M. Cardoso, S. Cheedella, Y. Copin, D. Crichton, D. DÁvella, C. Deil, E. Depagne, J. P. Dietrich, A. Donath, M. Droettboom, N. Earl, T. Erben, S. Fabbro, L. A. Ferreira, T. Finethy, R. T. Fox, L. H. Garrison, S. L. J. Gibbons, D. A. Goldstein, R. Gommers, J. P. Greco, P. Greenfield, A. M. Groener, F. Grollier, A. Hagen, P. Hirst, D. Homeier, A. J. Horton, G. Hosseinzadeh, L. Hu, J. S. Hunkeler, Z. Ivezic, A. Jain, T. Jenness, G. Kanarek, S. Kendrew, N. S. Kern, W. E. Kerzendorf, A. Khvalko, J. King, D. Kirkby, A. M. Kulkarni, A. Kumar, A. Lee, D. Lenz, S. P. Littlefair, Z. Ma, D. M. Macleod, M. Mastropietro, C. McCully, S. Montagnac, B. M. Morris, M. Mueller, S. J. Mumford, D. Muna, N. A. Murphy, S. Nelson, G. H. Nguyen, J. P. Ninan, M. Nöthe, S. Ogaz, S. Oh, J. K. Parejko, N. Parley, S. Pascual, R. Patil, A. A. Patil, A. L. Plunkett, J. X. Prochaska, T. Rastogi, V. R. Janga, J. Sabater, P. Sakurikar, M. Seifert, L. E. Sherbert, H. Sherwood-Taylor, A. Y. Shih, J. Sick, M. T. Silbiger, S. Singanamalla, L. P. Singer, P. H. Sladen, K. A. Sooley, S. Sornarajah, O. Streicher, P. Teuben, S. W. Thomas, G. R. Tremblay, J. E. H. Turner, V. Terrón, M. H. van Kerkwijk, A. de la Vega, L. L. Watkins, B. A. Weaver, J. B. Whitmore, J. Woillez, and V. Zabalza. The Astropy Project: Building an inclusive, open-science project and status of the v2.0 core package, 2018, arXiv:1801.02634. doi:10.3847/1538-3881/aabc4f.

[CRT+13] T. A. Collaboration, 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, A. Conley, N. Crighton, K. Barbary, D. Muna, H. Ferguson, F. Grollier, M. M. Parikh, P. H. Nair, H. M. Günther, C. Deil, J. Woillez, S. Conseil, R. Kramer, J. E. H. Turner, L. Singer, R. Fox, B. A. Weaver, V. Zabalza, Z. I. Edwards, K. A. Bostroem, D. J. Burke, A. R. Casey, S. M. Crawford, N. Dencheva, J. Ely, T. Jenness, K. Labrie, P. L. Lim, F. Pierfederici, A. Pontzen, A. Ptak, B. Refsdal, M. Servillat, and O. Streicher. Astropy: A community python package for astronomy, 2013, arXiv:1307.6212. doi:10.1051/0004-6361/201322068.

[Dea07] R. Deakin. Satellite Orbits, 2007. URL: https://researchgate.net/publication/228860752_SATELLITE_ORBITS.

[DGE15] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised Visual Representation Learning by Context Prediction, 2015, arXiv:1505.05192.

[DV16] V. Dumoulin and F. Visin. A guide to convolution arithmetic for deep learning, 2016, arXiv:1603.07285.

[EMH20] M. Espadoto, A. Martinazzo, and N. S. T. Hirata. Deep Learning for Astronomical Object Classification: A Case Study. *International Conference on Computer Vision Theory and Applications*, 2020. doi:10.5220/0008939800870095.

[FC19] J. Frankle and M. Carbin. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. *International Conference on Learning Representations*, 2019. arXiv:1803.03635.

[FJ19] C. J. Fluke and C. Jacobs. Surveying the reach and maturity of machine learning and artificial intelligence in astronomy, 2019, arXiv:1912.02934. doi:10.1002/widm.1349.

[FRKK18] I. Freeman, L. Roese-Koerner, and A. Kummert. EffNet: An Efficient Structure for Convolutional Neural Networks. *International Conference on Image Processing*, 2018. doi:10.1109/ICIP.2018.8451339.

[GFHL14] P. Graff, F. Feroz, M. P. Hobson, and A. Lasenby. SkyNet: an efficient and robust neural network training tool for machine learning in astronomy. *Monthly Notices of the Royal Astronomical Society*, 441:1741–1759, 2014. doi:10.1093/mnras/stu642.

[GMH18] R. E. González, R. P. Muñoz, and C. A. Hernández. Galaxy detection and identification using deep learning and data augmentation. *Astronomy and Computing*, 2018. doi:10.1016/j.ascom.2018.09.004.

[GSB+19] A. Ginsburg, B. M. Sipőcz, C. E. Brasseur, P. S. Cowperthwaite, M. W. Craig, C. Deil, J. Guillochon, G. Guzman, S. Liedtke, P. L. Lim, K. E. Lockhart, M. Mommert, B. M. Morris, H. Norman, M. Parikh, M. V. Persson, T. P. Robitaille, J.-C. Segovia, L. P. Singer, E. J. Tollerud, M. de Val-Borro, I. Valtchanov, J. Woillez, and the Astroquery collaboration. astroquery: An astronomical web-querying package in python, 2019, arXiv:1901.04520. doi:10.3847/1538-3881/aafc33.

[GSK18] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised Representation Learning by Predicting Image Rotations. In *International Conference on Learning Representations*, 2018, arXiv:1803.07728.

[GTA+21] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, M. Shahzad, W. Yang, R. Bamler, and X. X. Zhu. A Survey of Uncertainty in Deep Neural Networks, 2021, arXiv:2107.03342.

[HHS+21] M. A. Hayat, P. Harrington, G. Stein, Z. Lukić, and M. Mustafa. Estimating galactic distances from images using self-supervised representation learning, 2021, arXiv:2101.04293.

[HLvdMW16] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely Connected Convolutional Networks, 2016, arXiv:1608.06993.

[Hor91] K. Hornik. Approximation Capabilities of Multilayer Feedforward Networks. *Neural Networks*, 4:251–257, 1991. doi:10.1016/0893-6080(91)90009-T.

[Hoy16] B. Hoyle. Measuring photometric redshifts using galaxy images and deep neural networks, 2016, arXiv:1504.07255. doi:10.1016/j.ascom.2016.03.006.

[HSH+20] M. A. Hayat, G. Stein, P. Harrington, Z. Lukić, and M. Mustafa. Self-supervised representation learning for astronomical images, 2020, arXiv:2012.13083.

[HSL+16] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger. Deep Networks with Stochastic Depth. *European Conference on Computer Vision*, 2016. doi:10.1007/978-3-319-46493-0_39.

[Hub26] E. Hubble. Extra-galactic nebulae. *Astrophysical Journal*, 64:321–369, 1926. doi:10.1086/143018.

[HZC+17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, 2017, arXiv:1704.04861.

[IS15] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *International Conference on Machine Learning*, 37:448–456, 2015. arXiv:1502.03167.

[KB15] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, 2015. arXiv:1412.6980.

[KB16] E. J. Kim and R. J. Brunner. Star–galaxy classification using deep convolutional neural networks. *Monthly Notices of the Royal Astronomical Society*, 464:4463–4475, 2016. doi:10.1093/mnras/stw2672.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Conference on Neural Information Processing Systems*, 2012. doi:10.1145/3065386.

[KZB19] A. Kolesnikov, X. Zhai, and L. Beyer. Revisiting Self-Supervised Visual Representation Learning. In *Conference on Computer Vision and Pattern Recognition*, 2019, arXiv:1901.09005.

[LC10] Y. LeCun and C. Cortes. MNIST handwritten digit database, 2010. URL: http://yann.lecun.com/exdb/mnist/.

[LCG+20] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*, 2020, arXiv:1909.11942.

[LCHY18] X. Li, S. Chen, X. Hu, and J. Yang. Understanding the Disharmony between Dropout and Batch Normalization by Variance Shift, 2018, arXiv:1801.05134.

[LCY13] M. Lin, Q. Chen, and S. Yan. Network In Network, 2013, arXiv:1312.4400.

[LGS99] R. H. Lupton, J. E. Gunn, and A. S. Szalay. A Modified Magnitude System that Produces Well-Behaved Magnitudes, Colors, and Errors Even for Low Signal-to-Noise Ratio Measurements. *The Astronomical Journal*, 118:1406–1410, 1999. doi:10.1086/301004.

[LJH+19] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. On the variance of the adaptive learning rate and beyond, 2019, arXiv:1908.03265.

[LSD15] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation, 2015, arXiv:1411.4038.

[LvdM08] G. H. L. van der Maaten. Visualizing data using t-SNE. *The Journal of Machine Learning Research*, 2008.

[LZD+18] W. Liu, M. Zhu, C. Dai, D. Y. He, J. Yao, H. F. Tian, B. Y. Wang, K. Wu, Y. Zhan, B.-Q. Chen, A.-L. Luo, R. Wang, Y. Cao, and X. C. Yu. Classification of large-scale stellar spectra based on deep convolutional neural network. *Monthly Notices of the Royal Astronomical Society*, 483:4774–4783, 2018. doi:10.1093/mnras/sty3020.

[MEH20] A. Martinazzo, M. Espadoto, and N. S. T. Hirata. Self-supervised Learning for Astronomical Image Classification. *International Conference on Pattern Recognition*, 2020. doi:10.1109/ICPR48806.2021.9412911.

[MFLP12] M. Masias, J. Freixenet, X. Lladó, and M. Peracaula. A review of source detection approaches in astronomical images. *Monthly Notices of the Royal Astronomical Society*, 422:1674–1689, 2012. doi:10.1111/j.1365-2966.2012.20742.x.

[MH19] A. Martinazzo and N. S. T. Hirata. Multiband image classification of astronomical objects. *Workshop of Works in Progress at the Conference on Graphics, Patterns and Images*, 2019. doi:10.5753/sibgrapi.est.2019.8314.

[MHM18] L. McInnes, J. Healy, and J. Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, 2018, arXiv:1802.03426.

[ML18] D. Masters and C. Luschi. Revisiting Small Batch Training for Deep Neural Networks, 2018, arXiv:1804.07612.

[Moc09] D. L. Moche. Astronomy: A Self-Teaching Guide, 2009.

[MOT15] A. Mordvintsev, C. Olah, and M. Tyka. Inceptionism: Going deeper into neural networks, 2015. URL: https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html.

[MZZS18] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun. ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design. *European Conference on Computer Vision*, 2018. doi:10.1007/978-3-030-01264-9_8.

[NdOH+21] L. Nakazono, C. M. de Oliveira, N. S. T. Hirata, S. Jeram, C. Queiroz, S. S. Eikenberry, A. H. Gonzalez, R. Abramo, R. Overzier, M. Espadoto, A. Martinazzo, L. Sampedro, F. R. Herpich, F. Almeida-Fernandes, A. Werle, C. E. Barbosa, L. S. Jr., E. V. Lima, M. L. Buzzo, A. Cortesi, K. Menéndez-Delmestre, S. Akras, A. Alvarez-Candal, A. R. Lopes, E. Telles, W. Schoenell, A. Kanaan, and T. Ribeiro. On the discovery of stars, quasars, and galaxies in the Southern Hemisphere with S-PLUS DR2. *Monthly Notices of the Royal Astronomical Society*, 2021. arXiv:2106.11986. doi:10.1093/mnras/stab1835.

[Nor38] D. Norman. The Development of Astronomical Photography. *Osiris 5*, pages 560–594, 1938. doi:10.1086/368498.

[OMS17] C. Olah, A. Mordvintsev, and L. Schubert. Feature visualization. *Distill Working Group*, 2(11), November 2017. doi:10.23915/distill.00007.

[ORS+19] C. M. Oliveira, T. Ribeiro, W. Schoenell, A. Kanaan, R. A. Overzier, A. Molino, L. Sampedro, P. Coelho, C. E. Barbosa, A. Cortesi, M. V. Costa-Duarte, F. R. Herpich, J. A. Hernandez-Jimenez, V. M. Placco, H. S. Xavier, L. R. Abramo, R. K. Saito, A. L. Chies-Santos, A. Ederoclite, R. L. de Oliveira, D. R. Gonçalves, S. Akras, L. A. Almeida, F. Almeida-Fernandes, T. C. Beers, C. Bonatto, S. Bonoli, E. S. Cypriano, E. V. R. de Lima, R. S. de Souza, G. F. de Souza, F. Ferrari, T. S. Gonçalves, A. H. Gonzalez, L. A. Gutiérrez-Soto, E. A. Hartmann, Y. Jaffe, L. O. Kerber, C. Lima-Dias, P. A. A. Lopes, K. Menendez-Delmestre, L. M. I. Nakazono, P. M. Novais, R. A. Ortega-Minakata, E. S. Pereira, H. D. Perottoni, C. Queiroz, R. R. R. Reis, W. A.

Santos, T. Santos-Silva, R. M. Santucci, C. L. Barbosa, B. B. Siffert, L. S. Jr., S. Torres-Flores, P. Westera, D. D. Whitten, J. S. Alcaniz, J. Alonso-García, S. Alencar, A. Alvarez-Candal, P. Amram, L. Azanha, R. H. Barbá, P. H. Bernardinelli, M. B. Fernandes, V. Branco, D. Brito-Silva, M. L. Buzzo, J. Caffer, A. Campillay, Z. Cano, J. M. Carvano, M. Castejon, R. C. Fernandes, M. L. L. Dantas, S. Daflon, G. Damke, L. J. de Melo de Azevedo, D. F. D. Paula, K. G. Diem, R. Donnerstein, O. L. Dors, R. Dupke, S. Eikenberry, C. G. Escudero, F. R. Faifer, H. Farías, B. Fernandes, C. Fernandes, S. Fontes, A. Galarza, N. S. T. Hirata, L. Katena, J. Gregorio-Hetem, J. D. Hernández-Fernández, L. Izzo, M. J. Arancibia, V. Jatenco-Pereira, Y. Jiménez-Teja, D. A. Kann, A. C. Krabbe, C. Labayru, D. Lazzaro, G. B. L. Neto, A. R. Lopes, R. Magalhães, M. Makler, R. de Menezes, J. Miralda-Escudé, R. Monteiro-Oliveira, A. D. Montero-Dorta, N. Muñoz-Elgueta, R. S. Nemmen, J. L. N. Castellón, A. S. Oliveira, D. Ortíz, E. Pattaro, C. B. Pereira, R. de la Reza, B. Quint, L. Riguccini, H. J. R. Pinto, I. Rodrigues, F. Roig, S. Rossi, K. Saha, R. Santos, A. S. Müller, L. A. Sesto, R. Silva, A. V. S. Castelli, R. Teixeira, E. Telles, R. C. T. de Souza, C. Thöne, A. de Ugarte Postigo, F. Urrutia-Viscarra, C. H. Veiga, M. Vika, A. Z. Vitorelli, A. Werle, S. V. Werner, and D. Zaritsky. The Southern Photometric Local Universe Survey (S-PLUS): improved SEDs, morphologies and redshifts with 12 optical filters. *Monthly Notices of the Royal Astronomical Society*, 489:241–267, 2019. doi:10.1093/mnras/stz1985.

[PD01] J. Palmer and A. C. Davenhall. The CCD Photometric Calibration Cookbook, 2001. URL: https://faculty1.coloradocollege.edu/~sburns/courses/18-19/pc362/PracticalObsAstro.pdf.

[PGM+19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Conference on Neural Information Processing Systems*, 2019. arXiv:1912.01703.

[PIP18] J. Pasquet-Itam and J. Pasquet. Deep learning approach for classifying, detecting and predicting photometric redshifts of quasars in the sloan digital sky survey stripe 82. *Astronomy & Astrophysics*, 611, 2018. arXiv:1712.02777. doi:10.1051/0004-6361/201731106.

[RAH21] N. V. N. Rodrigues, L. R. Abramo, and N. S. Hirata. The information of attribute uncertainties: what convolutional neural networks can learn about errors in input data, 2021, arXiv:2108.04742.

[RDS+14] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2014, arXiv:1409.0575.

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986. doi:10.1038/323533a0.

[RMC15] A. Radford, L. Metz, and S. Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks, 2015, arXiv:1511.06434.

[RMS+17] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. Le, and A. Kurakin. Large-Scale Evolution of Image Classifiers. *International Conference on Machine Learning*, 2017. arXiv:1703.01041.

[Ros58] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958. doi:10.1037/h0042519.

[SCD+16] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, 2016, arXiv:1610.02391. doi:10.1007/s11263-019-01228-7.

[Sch63] M. Schmidt. 3C 273 : A Star-Like Object with Large Red-Shift. *Nature*, 197:1040, 1963. doi:10.1038/1971040a0.

[SHCB+17] H. D. Sánchez, M. Huertas-Company, M. Bernardi, D. Tuccillo, and J. L. Fischer. Improving galaxy morphologies for sdss with deep learning, 2017, arXiv:1711.05744. doi:10.1093/mnras/sty338.

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. URL: https://jmlr.org/papers/v15/srivastava14a.html.

[SHM+16] B. Sesar, N. Hernitschek, S. Mitrović, Željko Ivezić, H.-W. Rix, J. G. Cohen, E. J. Bernard, E. K. Grebel, N. F. Martin, E. F. Schlafly, W. S. Burgett, P. W. Draper, H. Flewelling, N. Kaiser, R. P. Kudritzki, E. A. Magnier, N. Metcalfe, J. L. Tonry, and C. Waters. Machine-learned identification of rr lyrae stars from sparse, multi-band data: the ps1 sample. *The Astronomical Journal*, 2016. arXiv:1611.08596. doi:10.3847/1538-3881/aa661b.

[SLJ+15] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. *Conference on Computer Vision and Pattern Recognition*, 2015. doi:10.1109/CVPR.2015.7298594.

[SZ14] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014, arXiv:1409.1556.

[TL19] M. Tan and Q. V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *International Conference on Machine Learning*, 2019. arXiv:1905.11946.

[Tys02] J. A. Tyson. Large Synoptic Survey Telescope: Overview. *SPIE - The International Society for Optical Engineering*, 4836, 2002. doi:10.1117/12.456772.

[VCIG14] J. T. VanderPlas, A. J. Connolly, Z. Ivezic, and A. Gray. Introduction to astroml: Machine learning for astrophysics, 2014, arXiv:1411.5039. doi:10.1109/CIDU.2012.6382200.

[WEM+10] E. L. Wright, P. R. M. Eisenhardt, A. Mainzer, M. E. Ressler, R. M. Cutri, T. Jarrett, J. D. Kirkpatrick, D. Padgett, R. S. McMillan, M. Skrutskie, S. A. Stanford, M. Cohen, R. G. Walker, J. C. Mather, D. Leisawitz, T. N. G.

III, I. McLean, D. Benford, C. J. Lonsdale, A. Blain, B. Mendez, W. R. Irace, V. Duval, F. Liu, D. Royer, I. Heinrichsen, J. Howard, M. Shannon, M. Kendall, A. L. Walsh, M. Larsen, J. G. Cardon, S. Schick, M. Schwalm, M. Abid, B. Fabinsky, L. Naes, and C.-W. Tsai. The Wide-field Infrared Survey Explorer (WISE): Mission Description and Initial On-orbit Performance. *The Astronomical Journal*, 140, 2010. arXiv:1008.0031. doi:10.1088/0004-6256/140/6/1868.

[XGD+17] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated Residual Transformations for Deep Neural Networks. *Conference on Computer Vision and Pattern Recognition*, 2017. doi:10.1109/CVPR.2017.634.

[YK16] F. Yu and V. Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. *International Conference on Learning Representations*, 2016. arXiv:1511.07122.

[Yur] D. Yuret. Knet.jl: Multilayer perceptrons. URL: https://knet.readthedocs.io/en/latest/mlp.html.

[ZL17] B. Zoph and Q. V. Le. Neural Architecture Search with Reinforcement Learning, 2017, arXiv:1611.01578.