Machine Learning Approaches for Astrophysics and Cosmology

by

Xan Morice-Atkinson

This thesis is submitted in partial fulfilment of
the requirements for the award of the degree of
Doctor of Philosophy of the University of Portsmouth.

March 30, 2019

For my mum, who passed away a week before I started postgraduate study.

Abstract

This thesis focuses on applying machine learning methods to both astronomical and cosmological problems. Regarding the application to astronomy, I use data analysis techniques new to astronomy to detect strong correlations in observed data to perform feature pre-selection, machine learning techniques (four tree-based methods including Random Forests) to classify astronomical objects, and novel software packages to interpret a machine learning model in an attempt to understand how it is correctly classifying objects. I showcase these techniques by applying them to the problem of star-galaxy separation using data from the Sloan Digital Sky Survey (hereafter SDSS) and the results show that the rate of misclassifications can be reduced by up to $\approx 33\%$ over the standard SDSS `frames` approach.

In reference to the application to cosmology, I seek to answer the question: 'can we distinguish between cosmological/gravitational models using machine learning, and if so, what features are useful discriminants?'. To approach this, I use an image classification machine learning method called Convolutional Neural Networks (CNNs) to classify dark matter particle simulations created with different theories of gravity. The results show that these simulations can be classified to a high degree of accuracy. I then investigate the model, using generated datasets with known parameters to probe the decision boundaries of the CNN and determine where the model breaks down. I also manipulate the CNN into creating representations of dark matter particle simulations to understand which features of the simulations it has been able to learn about - showing that CNNs do not have to simply focus on matter density variance and can learn about higher order statistics such as isodensity curvature. All of these methods are new to the analysis of different theories of gravity in cosmology.

Table of Contents

List of Tables

List of Figures

Declaration

The work of this PhD thesis was carried out at the Institute of Cosmology and Gravitation, University of Portsmouth, United Kingdom, under the supervision of Dr. David Bacon and Professor Kazuya Koyama.

Whilst registered as a candidate for the above degree, I have not been registered for any other research award. The results and conclusions embodied in this thesis are the work of the named candidate and have not been submitted for any other academic award.

The work in this thesis is my own unless otherwise stated.

Several codes were provided by Dr. Ben Hoyle and Dr. Cullan Howlett. The use of these resources is stated throughout.

Chapters 2 and 3 contain work from Morice-Atkinson et al. (2018).

Word count: 29155 words.

Acknowledgements

I would like to thank the Institute of Cosmology and Gravitation (ICG), the University of Portsmouth, and SEPNet for the opportunity to undertake postgraduate study.

I would like to extend my sincere gratitude and thanks to my supervisor, David Bacon, for his guidance, patience, and kindness throughout my PhD. His support over the years has gotten me to the other side.

Thank you to my main collaborator, Ben Hoyle, who endured many video calls with me. He introduced me to machine learning and inspired me to learn about it. I hope one day I will be able to thank him in person.

I would also like to thank Gary Burton and Edd Edmondson for their help when using the Sciama HPC cluster, Kazuya Koyama and Hans Winther for their guidance and teaching me about the DGP dark matter particle simulations, and Cullan Howlett for his explanations and help with measuring the power spectrum.

I would like to express particular gratitude to my friends at the ICG; Tom, Matt W, Hedda, Ben T, Jimmy, Jen, Diego, David, James, Cullan, Karen, Coleman, Nic, and Sam. Thank you to my friends outside of the ICG, Owen, Callum, Jake, Les, and Emily C who have always been there for me.

Claire Le Cras, thank you for your understanding, patience, love, and support over these years.

Finally, I would like to thank my family; Dad and Irene, Elke, James, and Cash. You are the ones that allowed me to get to this point with your enduring love, understanding, and support. Thank you to my Mum, who helped me through every day.

DB is supported by Science and Technology Facilities Council (STFC) consolidated grant ST/N000668/1.

Numerical computations were done on the Sciama High Performance Compute (HPC) cluster which is supported by the ICG, SEPNet and the University of Portsmouth.

Funding for Sloan Digital Sky Survey III has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, and the U.S. Department of Energy Office of Science. The SDSS-III web site is http://www.sdss3.org/.

SDSS-III is managed by the Astrophysical Research Consortium for the Participating Institutions of the SDSS-III Collaboration including the University of Arizona, the Brazilian Participation Group, Brookhaven National Laboratory, Carnegie Mellon University, University of Florida, the French Participation Group, the German Participation Group, Harvard University, the Instituto de Astrofisica de Canarias, the Michigan State/Notre Dame/JINA Participation Group, Johns Hopkins University, Lawrence Berkeley National Laboratory, Max Planck Institute for Astrophysics, Max Planck Institute for Extraterrestrial Physics, New Mexico State University, New York University, Ohio State University, Pennsylvania State University, University of Portsmouth, Princeton University, the Spanish Participation Group, University of Tokyo, University of Utah, Vanderbilt University, University of

Virginia, University of Washington, and Yale University.

Acronyms and Abbreviations

ADA Adaboost/Adaptive Boosting.

Adam Adaptive Moment Estimation.

AM Activation Maximisation.

ANN Artificial Neural Networks.

BAO Baryonic Acoustic Oscillations.

BOSS Baryon Oscillation Spectroscopic Survey.

CDM Cold Dark Matter.

CMB Cosmic Microwave Background.

CNNs Convolutional Neural Networks.

DES Dark Energy Survey.

eBOSS Extended Baryon Oscillation Spectroscopic Survey.

EFEs Einstien Field Equations.

EMNIST Extended MNIST.

EXT Extra-trees.

FFT Fast Fourier Transform.

GANs	Generative Adversarial Networks.
GBT	Gradient Boosted Trees.
L-PICOLA	A Lightcone-enabled Parallel Implementation of the COLA (COmoving Lagrangian Acceleration) method.
LRGs	Luminous Red Galaxies.
MG-PICOLA	Modified Gravity version of L-PICOLA.
MINT	Mutual Information based Transductive Feature Selection method.
MNIST	Modified National Institute of Standards and Technology database.
NIST	National Institute of Standards and Technology database.
NN	Neural Network.
QSO	Quasar.
RF	Random Forest.
SDSS	Sloan Digital Sky Survey.
SGD	Stochastic Gradient Descent.
SNe	Supernova.
SVM	Support Vector Machine.

Chapter 1

Introduction

The current model of our Universe indicates that everything we can observe originated from the expansion of a very high density and high temperature region - a model we call the Big Bang. During the last century, there has been great progress in the understanding of this model and the nature of our Universe. This great progress encompasses the solutions to Einstein's General Relativity (GR; Einstein, 1916), the discovery that the expansion of the Universe is accelerating (Riess et al., 1998; Perlmutter et al., 1999), and precise measurements of the fraction of energy and mass components the Universe is comprised of (Komatsu et al., 2011).

The standard cosmological model (or ΛCDM model) goes some way to explaining these observations. It has a Cold Dark Matter (CDM) component to account for the matter that we cannot observe directly, and a dark energy (Λ) component to account for the negative pressure that causes the accelerated expansion we observe. While the model is very strange it fits the observed data well, but there are still alternative theories that have yet to be ruled out.

In the late-time Universe we have observed in modern galaxy surveys (Alam et al., 2015; Abbott et al., 2018) that matter has condensed to form stars and billions of these stars have become gravitationally bound to form galaxies.

Through the process of analysing this observed data, we can infer constraints on cosmological models. It is for this reason that the models must be well understood, and the observed data be devoid of erroneous samples. It is therefore of great interest to investigate new techniques, such as machine learning, to provide insight into cosmological models and to correctly classify objects in the Universe.

In this chapter I will begin with a discussion of GR and an introduction to theoretical cosmology. I will then introduce some of the observational evidence supporting the theory and discuss galaxies and stars in the late-time Universe. I will then discuss modifications to gravity, particularly DGP, a theory that utilises more dimensions to explain the acceleration of the expansion of the Universe. These topics will set the scene for my investigation into using machine learning techniques in cosmology.

1.1 Cosmological Model

The Copernican principle asserts that there are no preferred observers in the Universe, arising from Copernicus' model that the retrograde motion of the planets was caused by the Earth's motion around the sun. This was a huge divergence from the earlier thinking that the Earth was at the center of the universe. A natural and more modern progression to this principle is that of the Cosmological Principle. In a similar way, the Cosmological Principle asserts that there are no preferred directions or regions (on large scales) in the universe, making it both statistically isotropic and homogeneous. More specifically, this would mean that on large scales of the universe, every direction would look the same, and the average density of matter would be the same in all places of the universe. It is often assumed (and this can be tested) that the laws of physics would be constant throughout the Universe in space and time - allowing our investigation into the nature of the Universe to be vastly less complicated. Finally, it is also established that all of space was causally connected at some point, with space and time starting with the

Big Bang - a high density, high temperature, causally connected state (see Section 1.3.1).

In this section I briefly outline some equations for our cosmological model from General Relativity (GR), a general metric in terms of GR that conforms with the cosmological principal - the FLRW metric, the equation governing the expansion of space - the Friedmann equation, and other related phenomena such as redshift, the magnitude system, and cosmological distance measures.

1.1.1 General Relativity

The theory of General Relativity describes gravity as a geometric property of spacetime, where the curvature of spacetime is directly influenced by the energy and momentum of matter and radiation (see Carroll (2004) for further discussion in connection to this section).

The theory is formulated with three key principles, summarised as:

- The principle of equivalence: This states that the inertial mass of an object is equal to its gravitational mass. The acceleration of a body due to gravity is independent of that body's mass.
- The principle of general covariance: The form of the laws of physics is the same in all inertial and accelerating frames. This is to say that physical phenomena are independent of the coordinate system used to describe them.
- The principle of consistency: This requires a new scientific theory to be able to reproduce the successful results from old theories it is designed to replace. For example, given the correct conditions, General Relativity should reduce to the laws of Newtonian mechanics and (with the omission of gravity) the formulations of special relativity.

With these principles in mind, Einstein formulated the theory that describes how the curvature of spacetime changes due to its matter content, and how in turn, the curvature causes that matter to gravitate. In the presence of curvature, geometry can no longer be described by normal Euclidean rules. For example, the length of a path is no longer given by $ds^2 = dx^2$ where x is a Euclidean coordinate; instead, the interval between two infinitesimally close events in spacetime is defined as

$$ds^2 = g_{\mu\nu} dx^\mu dx^\nu, \tag{1.1}$$

where $g_{\mu\nu}$ is the spacetime metric tensor and x are some coordinates. This metric is used to calculate the curvature of a Riemannian manifold from the Riemann tensor, defined as

$$R^\rho{}_{\lambda\mu\nu} = \Gamma^\rho{}_{\nu\lambda,\mu} - \Gamma^\rho{}_{\mu\lambda,\nu} + \Gamma^\rho{}_{\mu\alpha}\Gamma^\alpha{}_{\nu\lambda} - \Gamma^\rho{}_{\nu\alpha}\Gamma^\alpha{}_{\mu\lambda}, \tag{1.2}$$

where a comma denotes a derivative. The Christoffel symbols, $\Gamma_{\lambda\mu\nu}$, describe an affine connection. This allows vectors to be transported around a curved spacetime while staying parallel to the original connection. They can be described with the metric as

$$\Gamma_{\lambda\mu\nu} = \frac{1}{2}(g_{\lambda\mu,\nu} + g_{\lambda\nu,\mu} - g_{\mu\nu,\lambda}). \tag{1.3}$$

If spacetime is flat, then $g_{\mu\nu} = diag(1, -1, -1, -1)$ and $\Gamma_{\lambda\mu\nu} = 0$.

The curvature of a manifold can be defined by the magnitude of the transformation of a vector after a parallel transport around a closed loop. This is what the Riemann curvature tensor measures. It follows that if the curvature was flat, then the magnitude of the transformation would be equal to 0, therefore $R^\rho{}_{\lambda\mu\nu} = 0$.

The Riemann tensor can be contracted over to produce the Ricci tensor $R_{\mu\nu} = g^{\rho\lambda} R_{\rho\lambda\mu\nu}$, which can be contracted over to produce the Ricci scalar $R = g^{\mu\nu} R_{\mu\nu}$. These are both used in conjunction with the metric to formulate the Einstein curvature tensor $G_{\mu\nu}$, which measures the curvature of the Universe in GR.

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R. \tag{1.4}$$

Now that the mathematics related to the curvature of the universe in GR has been defined, we can now outline the equations related to the matter content of the universe. The energy-momentum tensor, T^{μ}_{ν}, describes the energy and momentum of material in the Universe.

In the particular cosmological case that we're interested in, we can use the cosmological principal, that the Universe is statistically homogeneous and isotropic, T^{μ}_{ν} must be a perfect fluid (meaning it is characterized only by its rest frame mass density and isotropic pressure, and all other possible properties are neglected). As such, it can be defined as

$$T^{\mu}_{\nu} = (\rho + P)U^{\mu}U_{\nu} - P\delta^{\mu}_{\nu} \tag{1.5}$$

where ρ is mass density, P is pressure, and U is the four velocity of the fluid. Energy-momentum conservation is described by

$$T^{\mu}_{\nu;\mu} = 0. \tag{1.6}$$

Now that we have equations governing the curvature of spacetime and the energy-momentum of the matter residing within it, these can be combined to form the Einstein Field Equations,

$$G_{\mu\nu} = \frac{8\pi G}{c^4}T_{\mu\nu} \tag{1.7}$$

The constant G, on the right hand side of the equation comes from the principle of consistency. This means that in the presence of weak gravitational fields, this equation will reproduce dynamics of Newtonian gravity.

1.1.2 The Friedmann-Lemaître-Robertson-Walker Metric

Shortly after Einstein published his theory of GR, Friedmann (1924), Lemaître (1927), Robertson (1935), and Walker (1935) all (independently) answered the question, in terms of GR, what is the most general metric that conforms to homogeneity and isotropy in the universe? The metric's most common form is

$$ds^2 = dt^2 - a(t)^2 \left[\frac{dr^2}{1 - kr^2} + r^2 d\theta^2 + sin^2(\theta) d\phi^2 \right] \tag{1.8}$$

where t is proper time - a coordinate-independent time measurement (the time measured by an observer along a trajectory), and r, θ, and ϕ are spatial coordinates. $a(t)$ is the scale factor, which measures the size of the Universe as a function of time and is normally set equal to 1 for the present day scale factor. k is the curvature parameter, which can be set to one of three different geometries, -1 for an open universe, 0 for a flat universe, and 1 for a closed universe.

For a flat FLRW space in cartesian coordinates, by calculating the Christoffel symbols (Equation 1.3) and using them with the Ricci tensor $R_{\mu\nu}$, the only relevant non-zero components of the Ricci tensor are

$$R_{tt} = -3\frac{\ddot{a}}{a}, \qquad R_{xx} = R_{yy} = R_{zz} = (a\ddot{a} + 2\dot{a}^2) \tag{1.9}$$

and the Ricci scalar is

$$R = 6\left(\frac{\ddot{a}(t)}{a(t)} + \frac{\dot{a}^2(t)}{a^2(t)}\right) \tag{1.10}$$

1.1.3 The Friedmann Equation

The Friedmann equation (Friedmann, 1922) is derived from using Equations 1.9 and 1.10 with the EFEs (Equation 1.7) and takes the form

$$H^2 = \left(\frac{\dot{a}}{a}\right)^2 = \frac{8\pi G}{3}\rho - \frac{kc^2}{a^2}. \tag{1.11}$$

This relates the density of the universe ρ with the rate of change of the scale factor with time ($H(t)$, the Hubble parameter). When the derivative of this equation is taken, the Friedmann acceleration equation is acquired

$$\dot{H} + H^2 = \frac{\ddot{a}}{a} = -\frac{4\pi G}{3}(\rho + 3P) \tag{1.12}$$

with P arising from the conservation of the energy-momentum tensor (Equation 1.6) where $\dot{\rho} = -3H(\rho + P)$. In Section 1.4 there will be a discussion of the DGP model - a departure from the ΛCDM model, in which I describe how the Friedmann equation is modified.

Hubble's law describes the rate of cosmic expansion, and was first confirmed by Edwin Hubble (1929). It explains the observation that objects in deep space seemingly have a 'redshift' - a relative velocity away from Earth causing the light we observe to be doppler shifted. The law is written locally as

$$v = H_0 d \tag{1.13}$$

where v is a galaxy's velocity, d is the proper distance to that galaxy, and H_0 is the Hubble parameter at the present redshift ($z = 0$). The motion of objects in the Universe that is solely due to this expansion is called the Hubble flow. The unit of H_0 is most commonly known as km s^{-1} Mpc^{-1}, but it is often expressed by the dimensionless parameter h_0, where $h_0 = \frac{H_0}{100}$km s^{-1}Mpc^{-1}. The most up to date measurement of h_0 from the Planck mission is 0.674 ± 0.005 (Planck Collaboration et al., 2018a), very different from Hubble's first measurement of 5 (Hubble, 1929). The current value of h_0 is still contested to this day with values calculated to be as much as 0.7 (Betoule et al., 2014).

Assessing the density of the Universe can be done with the Friedmann equation. ρ_c is the critical density for a flat universe where $k = 0$, therefore Equation 1.11 can be written as

$$\rho_c = \frac{3H^2}{8\pi G} \tag{1.14}$$

The density parameter Ω is defined as the ratio of the observed density parameter ρ to the critical density ρ_c:

$$\Omega \equiv \frac{\rho}{\rho_c} = \frac{8\pi G\rho}{3H^2}. \tag{1.15}$$

This means that when this ratio is equal larger that 1, space is geometrically closed, and the universe will stop expanding and eventually collapse. If Ω is measured to be smaller than 1, space is geometrically open, and the universe will continue expanding forever. The Planck Collaboration et al. (2018a) results indicate that $\Omega = 1$ and there is no contribution from the curvature parameter Ω_k, with it being calculated to be equal to 0.001 ± 0.002. This leads to being able to define the overall density of the universe as in the ΛCDM model, a sum of its density components

$$\Omega \equiv \Omega_m + \Omega_r + \Omega_\Lambda = 1 \tag{1.16}$$

where m is matter, r is radiation, and Λ is the cosmological constant.

Matter (Ω_m) makes up around 30% of the total density of the Universe and is comprised of two components, baryonic matter and dark matter. Baryonic matter (protons and neutrons) is the matter we are able to observe, and makes up all atoms, planets, stars, galaxies in the universe. Dark matter, we are unable to directly observe and is thought to interact only with gravitational forces, and can only be detected by observing its effect on astronomical objects using gravitational lensing or by studying galaxy rotation curves. The other 70% of the density of the Universe is thought to be 'dark energy', a possible explanation for the accelerating expansion of the universe. Carroll (2001) provides a review of a proposed dark energy model that uses a Cosmological constant Λ - which assumes that dark energy is constant through space and time. The accelerated expansion of the Universe can be characterised by the equation of state of dark energy, w, which is the ratio of its pressure (p) to its energy density (ρ). The final component of the content of the Universe is that of radiation, r, which was dominant in the early stages of the universe, but now makes up a negligible fraction of the total density.

1.1.4 Redshift, Magnitudes, and Distance Measures

As said in Section 1.1.3, redshift (z) is measured from light emitted from an object in deep space that has been doppler-shifted. This ratio of the change in wavelength is defined as

$$\frac{\lambda_r}{\lambda_e} = 1 + z \tag{1.17}$$

where λ_r and λ_e are wavelength received and emitted respectively. To calculate the total time for the light to travel the distance $r = 0$ to $r = R$, Equation 1.17 is related to the scale factor by integrating the null geodesic equation, $ds^2 = 0$, for the FLRW metric, giving

$$\chi = \int_{t_e}^{t_r} \frac{dt}{a(t)} = \int_0^R \frac{dr}{1 - kr^2} \tag{1.18}$$

where t_e and t_r are the time a photon was emitted and received respectively, and t is the present time. The object in deep space that emitted this light is stationary in comoving coordinates r (its perceived velocity is due to the scale factor $a(t)$). If light was to be emitted and received a short time later ($t_e + \Delta t_e$ and $t_r + \Delta t_r$), it will have travelled the same comoving distance (χ). This means

$$\int_{t_e}^{t_r} \frac{dt}{a(t)} = \int_{t_e+\Delta t_e}^{t_r+\Delta t_r} \frac{dt}{a(t)}. \tag{1.19}$$

If the limits of the integral are rearranged and $\Delta t_{e,r}$ are assumed to be negligible,

$$\frac{dt_e}{a(t_e)} = \frac{dt_r}{a(t_r)}. \tag{1.20}$$

If Equation (1.20) is instead considered for two successive peaks of emitted light waves, the time elapsed between the peaks of the waves is proportional to the wavelength λ

$$\frac{\lambda_r}{\lambda_e} = \frac{a(t_r)}{a(t_e)} = 1 + z \tag{1.21}$$

For present day, where $a(t_r) = 1$, this equation can be written as $a(t_e) = \frac{1}{1+z}$. The wavelength emitted by these photons in rest frame has been accurately measured in spectroscopic laboratory experiments, and consequently, the redshift is calculated from the shifted wavelengths we observe.

A useful distance measure is that of angular diameter distance, D_A. It is defined as the ratio between an object's proper size (l) and apparent angular size (θ) as measured on the sky: $D_A = \frac{l}{\theta}$.

In a spatially flat universe, D_A can be related to the comoving distance or luminosity distance (a measurement of an object's luminosity at cosmological distances) and redshift such that

$$D_A = \frac{r}{1+z} = \frac{D_L}{(1+z)^2}, \tag{1.22}$$

where the redshift term accounts for the evolution of the scale factor between the object's light emission and the present.

The luminosity distance can also be related to an object's absolute and apparent magnitude, with the absolute magnitude (M) being a measure of an object's intrinsic luminosity, and apparent magnitude (m) being a measure of the object's magnitude if it were viewed from a distance of 10 parsecs. This relation is

$$M = m - 5(log_{10}(D_L) - 1) \tag{1.23}$$

The bolometric magnitude is different from the absolute magnitude as it accounts for radiation at all wavelengths, and is defined by the ratio of luminosity to that of the Sun ($L_\odot$) in the following way

$$M_{bol,\star} - M_{bol,\odot} = -2.5log_{10}\left(\frac{L_\star}{L_\odot}\right). \tag{1.24}$$

We will use these magnitudes extensively in Chapter 3.

1.2 The Visible Universe

In this section I briefly discuss the objects we can directly observe in our Universe, stars and galaxies. I will discuss why they are important to classify correctly, and how these classification methods have become more automated in recent years. This sets the scene for my machine learning classification work in Chapter 3.

1.2.1 Stars and Galaxies

One method of probing the properties of the Universe is through the processes of stars and galaxies. For instance, the deflection of light around our star due to GR was first measured by Dyson and Eddington (Dyson et al. (1920)). In more recent times, the most distant galaxy ever discovered (named GN-z11), has a spectroscopically confirmed redshift of $z = 11.09$ (Oesch et al., 2016) and existed approximately 400Myr after the Big Bang. This type of age dating of galaxies provides an independent check on the timescales provided from cosmological models. Many different physical galaxy properties such as age, type, metallicity, and stellar mass, can be obtained by observing the light that comes from them using recent advances in stellar population modelling techniques. Hubble was the first to classify or group galaxies by type, in a sequence or 'tuning-fork' (Hubble, 1926), with the main classifications being large, red, early-type elliptical shaped galaxies, and bluer, late-type spiral or barred-spiral galaxies. Methods analagous to this are still used to classify galaxies today as shown in Masters et al. (2011), where they show a colour cut (where the magnitude of light in one filter is compared with another) of g-i $>$ 2.35 is able to split the galaxies into morphologies comparable to early-type Luminous Red Galaxies (LRGs) and late-type passive spiral galaxies at lower redshift. This is useful, as being able to accurately select targets is important for precise measurements of the cosmic distance scale or Hubble parameter - for example, the use of quasars (QSOs; highly luminous and distant galaxies) is discussed further in Section 1.3.4.

1.2.2 The Automation of Classification

In recent years, surveys of the sky such as the Sloan Digital Sky Survey (SDSS, Alam et al., 2015) and the Dark Energy Survey (DES, DES Collaboration et al., 2017) have incorporated data processing/reduction pipelines to automate the processing of vast amounts of data, helping to standardise and accelerate science calculations. These pipelines convert images seen by the telescope into managable and useable data, with the type of data depending on the instruments used to observe the objects. A full spectrum (a detailed measure of flux between two wavelengths) of the object may be available if it has been observed with a spectrograph, or photometric magnitudes (a coarse measurement in a small number of filters) calculated from the images will be available. Various programs are incorporated in different pipelines to automatically output results. For example, in the particular example of determining properties of galaxies, the SDSS pipeline (Bolton et al., 2012) includes various template fitting methods: a program called HyperZ (Bolzonella et al., 2000) in its pipeline that compares photometric data to redshifted stellar population models, and a full spectral fitting method called Firefly (Wilkinson et al., 2017). In addition to template fitting methods, there is a number of training methods included such as ANNz (Collister and Lahav, 2004) or TPZ (Carrasco Kind and Brunner, 2013), which use machine learning techniques to learn relations between the data and target results, such as redshift or galaxy properties. These newer type of machine learning techniques have yet to be included in these data processing pipelines.

Further discussion, analysis, and results of old classification methods included in the SDSS pipeline and newer classification methods are presented in Chapter 3.

1.3 Cosmological Probes

Beyond these questions of classification, we want to test the nature of the cosmological model, posing the question - 'Is the Universe as we would expect from ΛCDM, or do we need a further model?'. I approach these questions in a machine learning context in Chapters 4 and 5.

In this section, I describe four main methods of probing the nature of our universe: analysing the Cosmic Microwave Background (CMB) to determine what happened at the very start of the Universe, the use of the light from Type 1a Supernova to calculate the expansion rate of the Universe, the use of Weak Gravitational Lensing to shed light on the nature of dark matter, and the analysis of the largest structures in our universe to further constrain the behaviour of the cosmological model.

1.3.1 CMB

The very early universe, when it was approximately one millionth of its current volume and 3,000,000 K in temperature, was filled with a ‘cosmic soup’ of coupled protons, electrons, and photons (Liddle, 2003). Due to the high temperature, the photons had an energy higher than the energy required to ionize hydrogen. As a consequence, any electron trying to bind to a proton was knocked away by a photon. This interaction (Thompson scattering) caused the mean free path of the photons to be very short, forming an opaque ionized plasma. This plasma had a very high sound speed due to the density and pressure being provided mainly by the photons, and propagated as a spherical sound wave. As the Universe expanded further and cooled, the photons lost their high energy, and electrons could bind with the protons. The photons could no longer interact with the particles, and were able to free stream through the universe. This process is known as decoupling, and marks the time known as recombination when the universe was no longer opaque, but transparent.
This radiation (named the Cosmic Microwave Background (CMB)) is the earliest radiation we can observe, has the form of Black Body radiation, and has a measured temperature of $2.72548 \pm 0.00057K$ (Fixsen, 2009). When measured at different angular distances on the sky, the CMB contains small irregularities or anisotropies. These conform to what is expected from initial quantum fluctuations in the Universe, magnified to a cosmic scale (Parker, 1968, 1969). These fluctuations are thought to be the initial seed for the large-scale structure we observe in the Universe at present day. The exis-

tence of the CMB was first predicted in 1948 (Gamow, 1948b,a; Alpher and Herman, 1948b,a), and has been measured many times through the years, with the first result from Arno Penzias and Robert Wilson in the 1960's and the latest result from the Planck satellite in 2018 (Planck Collaboration et al., 2018b).
Figure 1.1 shows the the Fourier transform of the two-point correlation function or the power spectrum of the CMB, a measurement taken by the Planck satellite (Planck Collaboration et al., 2016). Large scales are at low l and small scales at high l. The first acoustic peak is at $l \sim 240$ and is sensitive to matter content of the universe after decoupling. Its position is consistent with the Universe being spatially flat; if its position was further to the left or right, it would indicate a closed universe or open universe respectively (Doroshkevich et al., 1978; Kamionkowski et al., 1994). The amplitude of the acoustic peaks are dependent on the matter density of the Universe. The second peak is lower than the first, which is due to the temperature of the baryonic matter causing pressure resulting in a damping of the amplitude. Having a third peak that is more enhanced than the second indicates a dark matter dominated matter density before recombination (Hu et al., 1996). At high l (after 1000) the peaks are quashed by Silk Damping (Silk, 1968), where photons after decoupling have travelled a finite distance before they are scattered causing the diffusion or damping on the smallest scales.

Modelling the shape of the CMB power spectrum can give good constraints on cosmological parameters of the Universe, as these peaks in the power spectrum were frozen in at the time of recombination. This can be done numerically with codes such as CAMB (Lewis et al., 2000) or CLASS (Lesgourgues, 2011), which solve the full set of Einstein Field and Boltzmann equations for a chosen cosmological model to replicate the expected temperature fluctuations on the sky given some initial perturbations. These are compared with measurements taken by satellites like Planck, with the latest observational results of H_0 being 67.4 ± 0.5km s^{-1}Mpc^{-1}.

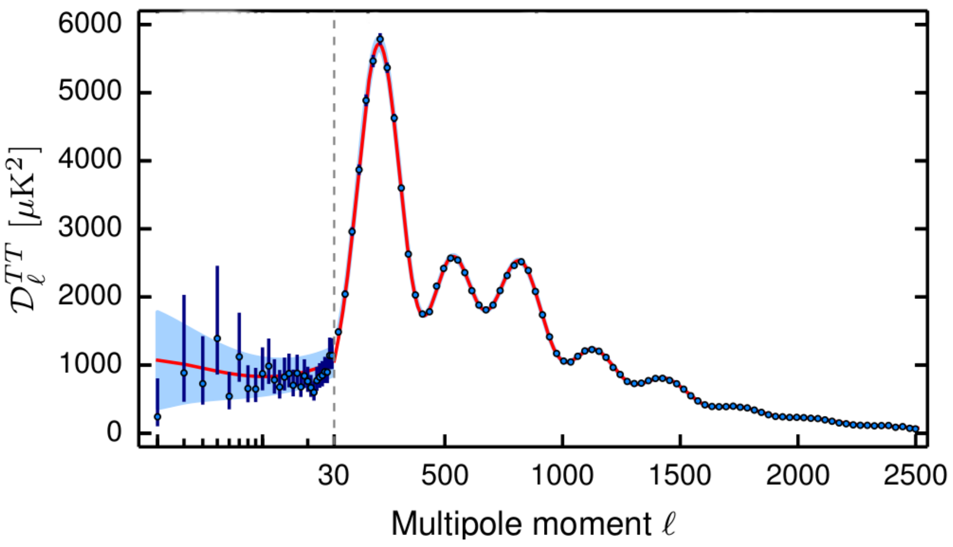


Figure 1.1: Power Spectrum of CMB measured in Planck Collaboration et al. (2016). This plot shows the measured amplitude of the temperature fluctuations in the CMB as a function of Multipole moment l (angular scale). The blue points are the measured data points, with the red curve being a prediction from theory.

1.3.2 Type 1a SNe

A Type 1a Supernova (SNe; da Silva, 1993) occurs when a carbon-oxygen white dwarf in a binary system accretes mass from a companion star and exceeds the Chandrasekhar limit of $\sim 1.44 M_{\odot}$, at which point it can no longer support its own mass due to pressure from electron degeneracy (Lieb and Yau, 1987). The star ignites due to the increase in temperature, fusion begins, and a large portion of the carbon and oxygen is fused into heavier elements. The energy released from this fusion causes the temperature to increase rapidly and the star undergoes a violent explosion, causing a massive shockwave and increase in luminosity to a typical absolute magnitude of M_S = -19.3 (Hillebrandt and Niemeyer, 2000). This typical magnitude is reached due to Type 1a SNe having similar masses, hence these objects can be used as standard candles - accurate distance measures of the universe.

The measurement of the peak brightness in Type 1a SNe by observing luminosity over time led to the first evidence of the accelerating expansion of the Universe (Riess et al., 1998). The brightness was converted to a luminosity distance (D_L) and spectral lines were used to calculate SNe's redshift (z). When this was done, Riess et al. (1998) discovered that the calculated distance to the high-redshift SNe were on average 10% to 15% higher than expected in a universe where there is no cosmological constant and $\Omega_m = 0.2$.

Figure 1.2 (Betoule et al., 2014) shows a more recent result using SNe from multiple data sources (Riess et al., 2007; Conley et al., 2011) called the Joint Lightcurve Analysis (JLA) . As in the Riess et al. (1998) paper, distance modulus (here, μ) is used along with redshift to show that $H_0 =$ 70km s^{-1}Mpc^{-1}.

The tension between Planck and Type 1a SNe measurements of H_0 has been previously said to be due to systematic uncertainties in either methods (Dhawan et al., 2018). The latest result from the Riess et al. (2018) paper calculates $H_0 = 73.48 \pm 1.66$km s^{-1}Mpc^{-1} after using a new sample that addresses two outstanding systematic uncertainties affecting prior comparisons of Milky Way and extragalactic Cepheids (stars that vary in brightness periodically) used to calibrate H_0: their dissimilarity of periods and photometric systems. This result has further increased the tension with the Planck + ΛCDM result to 3.7σ, and the source of this discrepancy remains undetermined.

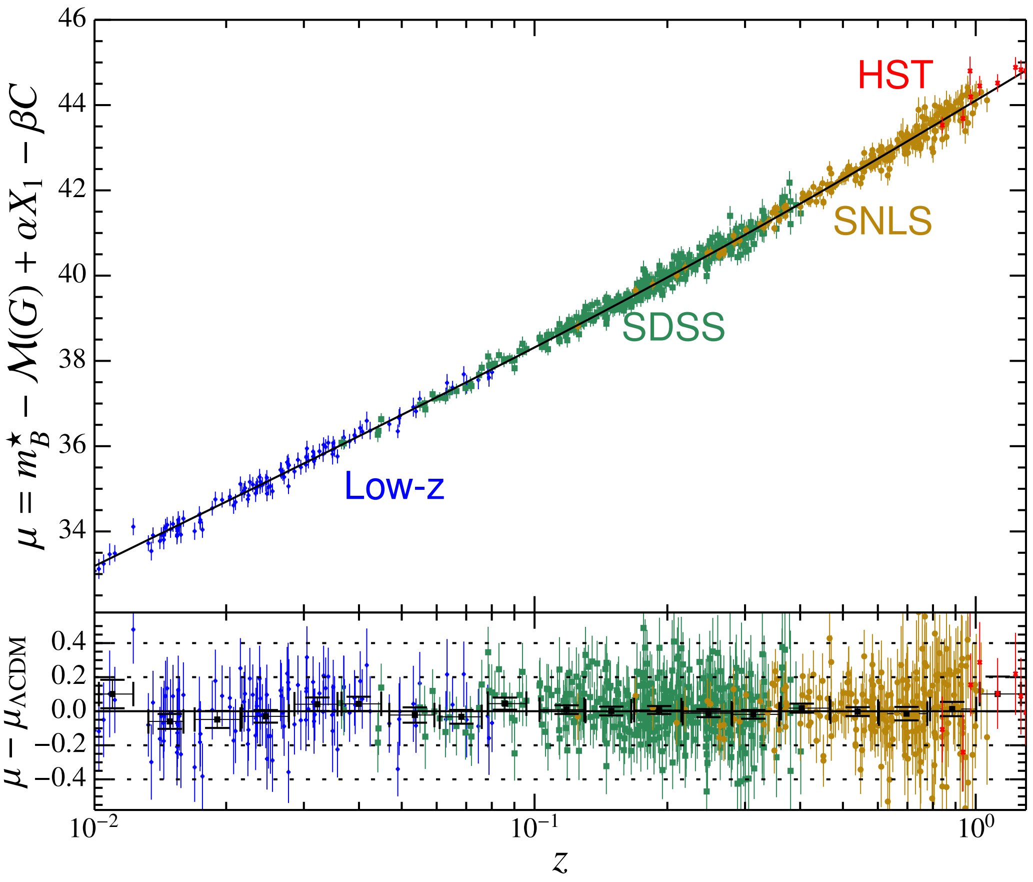


Figure 1.2: Hubble diagram of supernovae from various samples. The top panel shows the distance modulus (μ) versus redshift (z) of the best-fit ΛCDM cosmology for a fixed value of $H_0 = 70\text{km s}^{-1}\text{Mpc}^{-1}$ (black line). The bottom panel shows the residuals as a function of redshift. The weighted average of the residuals in logarithmic redshift bins of width $\Delta/z \sim 0.24$ (black dots in bottom panel) (Betoule et al., 2014).

1.3.3 Weak Gravitational Lensing

Cosmological constraints can also be determined from the assessment of weak gravitational lensing, measuring the distribution of matter in the universe (Hu and Tegmark, 1999). Unlike strong gravitational lensing, where the path of light (from a background object) is distorted around an object (in the foreground) with a mass density greater than a critical density, weak lensing takes effect along most lines of sight through the universe. This is to say, a foreground mass can be detected through the alignment of background objects - measurement of their ellipticities or shear.

There are several ways in which weak gravitational lensing can occur: for example, clusters of galaxies can cause strong lensing effects on aligned background sources, but also a more general weak lensing on the surrounding background sources. These types of lensing measurements are very important because they are independent from theories of star formation or cluster dynamics. These standalone lensing mass maps can be correlated with optical measurements to reveal the connection between dark matter mass and stellar mass (Clowe et al., 2004).

A more subtle type of weak lensing comes in the form of galaxy-galaxy lensing. This occurs when a galaxy causes the lensing, instead of a cluster of galaxies. The detection of this signal is somewhat more difficult as the galaxy shears need to be stacked or combined to make the signal significant. An even weaker signal than that of galaxy-galaxy lensing is cosmic shear. This is when the large-scale structure of the universe produces a measurable shear in background sources, usually found by calculating a two-point shear correlation function.

This type of weak lensing analysis has been performed by DES (DES Collaboration et al., 2017), using a combination of three different two-point functions: the cosmic shear correlation function of 26 million source galaxies, the galaxy angular autocorrelation function of 650,000 luminous red galaxies, and the galaxy-shear cross-correlation of luminous red galaxy positions and source galaxy shears. The constraints on S_8 (σ_8, the amplitude of the (linear) power spectrum on the scale of 8 h^{-1}Mpc) and Ω_m DES calculated from their analysis, compared to, and combined with Planck results are shown in Figure 1.3. There is clearly a tension between DES and Planck with both the S_8 and Ω_m parameters.

Another interesting result relevant to this work is that of the current largest curved-sky galaxy weak lensing mass map from the DES first-year (DES Y1) data (Chang et al., 2018). The map (shown in Figure 1.4) made from gravitational lensing measurements of 26 million galaxies in the DES Y1 data. It covers over 1300 deg^2 and covers redshifts $0.2 < z < 1.3$. Red regions have more dark matter than average, blue regions less dark matter.

1.3.4 Large-Scale Structure

Near the beginning of the Universe, as the CMB shows, matter was almost uniformly distributed across space. As time progressed, the dark matter collapsed into a cosmic web as shown in Figure 1.4; baryons fell into these gravitational potentials so that galaxies clustered together, tracing out the cosmic web in a biased fashion. The specifics of the web pattern tell us about gravity, matter density, the matter power spectrum, and dark energy. This cosmic web pattern is used to discriminate between theories of gravity in Chapters 4 and 5.

One particular aspect of the web patters is the feature known as the Baryon Acoustic Oscillations (BAO). After recombination, when the photons had free streamed away, an excess of the baryons were left behind in a spherical shell around the initial excess density of dark matter. This spherical shell expanded along with the Universe, leaving a signature of the overdensity of baryons after recombination - this is known as a 'peak' in the BAO. Even though the overdensities in dark matter pulled the baryons into the potential wells, the signature of the baryon overdensity is still present and measurable as a peak in the two-point correlation function of galaxy distributions. Measuring this BAO signal was first achieved by Eisenstein et al. (2005) and allows us to further constrain the cosmological parameters of the

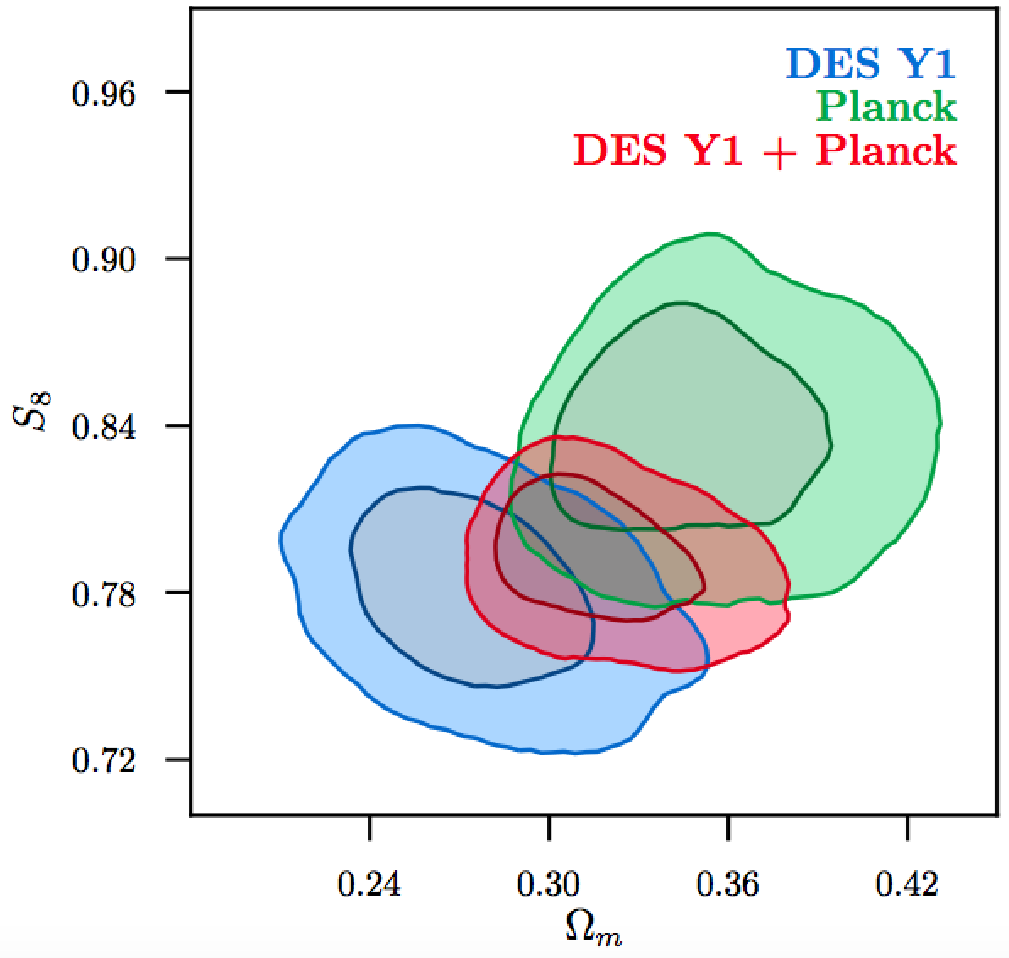


Figure 1.3: This plot shows the 68% and 95% confidence levels for constraints on the values of S_8 and Ω_m using weak lensing from DES Y1 data. The blue contour is solely using DES Y1 data, the green contour is using Planck data, and the red contour is using a combination of both results (DES Collaboration et al., 2017).

Universe by comparing the observed result to predictions. For example, the BAO peak would be observed to be at a different scale, or intensity, if the flatness or matter content of the Universe were different from the currently accepted model.

The reason this result has only been achieved in recent years is because the BAO signal is very weak, and a large survey such as SDSS-III's BOSS (Dawson et al., 2013) is required for the signal to be significant. The BAO measurement is calculated through measuring the two-point correlation function of galaxy positions in different redshift bins.

An example of a recent detection of the peak in the BAO at ~ 105 Mpc on the x-axis can be seen in Figure 1.5. This peak is seen in 3 redshift bins (shifted in ζ on the y-axis for clarity), where the red curve is $0.2 < z < 0.5$, black is $0.4 < z < 0.6$, and blue is $0.5 < z < 0.75$. It was calculated using a combined galaxy sample of 1.2 million galaxies over 9329 deg^2, and a volume of 18.7 Gpc^3. They calculate H_0 to be 67.6 ± 0.5km s^{-1}Mpc^{-1} assuming flat ΛCDM cosmology, consistent with Planck Collaboration et al. (2018a).

This analysis has been also been performed with the SDSS-IV eBOSS sample (Dawson et al., 2016), including the first measurement of the BAO between redshift 0.8 and 2.2 (Ata et al., 2018). This analysis was performed using a sample of 147,000 QSOs over 2044 deg^2. Figure 1.6 shows the BAO peak in configuration space, and Figure 1.7 shows their constraints on ΛCDM. The errors on this plot are larger due to the use of the quasar sample, which is more uncertain data than the sample used to calculate the BAO peak seen in Figure 1.5.

The different methods that I have described can be combined to give more precise constraints on cosmological parameters, as shown in Figure **??** from the Alam et al. (2017) paper. The figure shows constraints (contours representing confidence intervals) on w and Ω_K for a model with varying spacial curvature and a constant equation of state of dark energy. It uses data from Planck 2015 results, SDSS Data Release 12 for the BAO measurements, and the Joint Lightcurve Analysis for the SNe data. It is seen here that when combining results gained from multiple methods, the errors on the measurements of cosmological parameters are greatly reduced.

1.4 Modified Gravity

While all the examples that have been given so far point towards the universe fitting a ΛCDM model, there are still anomalies that cannot be explained. The most serious problem is known as the 'cosmological constant problem', where the observed value of Λ is around 120 orders of magnitude smaller than the theoretically expected value (cut off at the Planck scale). Another problem is called the 'coincidence problem', and poses the question 'if the Universe is accelerating in its expansion as we observe, why is it happening now?'. Another way of thinking about this problem is to ask the question 'why is Ω_Λ and Ω_m in the same order of magnitude?'.

Clifton et al. (2012) has a comprehensive review of alternative theories to ΛCDM, of which one will be considered further in this work. The number of theories that can be constructed solely using the metric tensor is limited, as given by Lovelock's theorem (Lovelock, 1971, 1972). This means that to construct a theory of gravity with field equations different from those in GR, one (or more) of the following must be done:

- Locality be abandoned.
- Fields considered other than that of the metric tensor.
- Spatial dimensions considered that are not four dimensional.
- Abandon rank (2,0) tensor field equations, symmetry of the field equations when exchanging indices, or divergence-free field equations.

DGP (Dvali et al., 2000) is useful and well-developed modified gravity theory to consider as a competitor to GR, for the purpose of developing machine learning approaches that distinguish between laws of gravity. DGP utilises a higher number of spacial dimensions than four, and is used extensively throughout this work in the form of dark matter particle simulations. A basic overview of DGP is given in Section 1.4.1 and a description of the dark matter particle simulations is given in Section 4.

To arrive at the theory of DGP gravity the EFEs mentioned in Section 1.1.1 must be rewritten in the form of the Einstein-Hilbert action,

$$S = \frac{1}{2}\int R\sqrt{-g}d^4x \tag{1.25}$$

where g is the determinant of $g_{\mu\nu}$ and R is the Ricci scalar. This equation results in the EFEs when the action is varied with respect to the metric and $\delta S = 0$. This means that if one wants to determine the form of the field equations for a different theory of gravity, we must simply define an action.

1.4.1 DGP Gravity

The DGP gravity model was first theorised by Dvali et al. (2000), and stipulates that matter is confined to a four-dimensional brane, which is embedded in a five-dimensional bulk space-time, making it an example of a *braneworld* model. The DGP action is defined as

$$\begin{aligned} S = &\int_{brane} d^4x\sqrt{-g}\left(\frac{R}{16\pi G}\right) + \int d^5x\sqrt{-g^{(5)}}\left(\frac{R^{(5)}}{16\pi G^{(5)}}\right) \\ &+ S_m(g_{\mu\nu}, \psi_i), \end{aligned} \tag{1.26}$$

where g^5 is the determinant of the five-dimensional metric of the bulk $g_{\mu\nu}$, with $R^{(5)}$ being its Ricci scalar. Similarly, g is the determinant of the metric of the brane ($g_{\mu\nu}$), with R being its Ricci scalar. G and $G^{(5)}$ are the four- and five-dimensional gravitational constants. ψ_i describes the matter fields and their action S_m, which are confined to the four-dimensional part of the model. The crossover scale, r_c, is shown in Equation 1.27, and is the ratio of $G^{(5)}$ to G,

$$r_c = \frac{1}{2}\frac{G^{(5)}}{G}. \tag{1.27}$$

Below the crossover scale r_c gravity looks four dimensional, and above it, characteristics of five dimensions become apparent. The explanation of the acceleration of the expansion of the universe is approached by two methods in the DGP model, the normal branch of DGP (hereafter nDGP), or the self-accelerating branch (sDGP). In the latter, the model does not require a dark energy field, but numerous publications have pointed towards this model being ruled out. This is due to it being in tension with CMB and supernovae data (Fang et al., 2008), but also issues with the propagation of ghosts (resultant additional degrees of freedom) (Koyama, 2007). In contrast, the nDGP model does require a dark energy term in the four dimensional part of the action to explain the accelerated expansion (Schmidt, 2009). Koyama and Maartens (2006); Koyama (2007) showed that the expansion rate of the nDGP model can be written as,

$$H(a) = H_0\sqrt{\Omega_{m0}a^{-3} + \rho_{DE}(a)/\rho_{c0} + \Omega_{rc}} - \sqrt{\Omega_{rc}}, \tag{1.28}$$

where a subscript 0 means the value at present day, a is the scale factor, H_0 is the Hubble expansion rate, $\Omega_{m0} = \bar{\rho}_{m0}8\pi G/(3H_0^2)$, and is fractional matter density where $\bar{\rho}_m$ is the background value of the matter density, and $\Omega_{rc} = 1/(4H_0^2 r_c^2)$. The dark energy term is denoted as ρ_{DE}, and ρ_c is the critical density as in Equation 1.14. This dark energy term can be tuned so that the expansion rate exactly matches that seen in ΛCDM (Schmidt, 2009).

In the past, there have been several methodologies used to parameterise models of gravity such as: presenting predictions for weak lensing correlation functions given modified gravity models and calculating how weak lensing statistics would change as a function of scale and redshift (Beynon et al., 2010), or using cosmic shear - either through a tomographic approach where correlations between the lensing signal in different redshift bins are used to recover redshift information, or via a 3D approach, where the full redshift information is carried through the whole analysis. Dark matter particle simulations have also been created using other modified gravity models (f(R) gravity) to assess how they affect the clustering of dark matter and halos and fill non-linear scales (Zhao et al., 2011). In this work, I use dark matter particle simulations created with the nDGP model of gravity, along with ones created with ΛCDM gravity in order to examine whether a machine learning approach is able to distinguish between the resulting cosmic webs.

1.5 Outline of Thesis

We have surveyed the state of cosmology, noting the importance of categorization of objects, and selection of cosmological models. In this thesis, I will examine both of these crucial tasks from a machine learning perspective.

As discussed in Section 1.3, observed data needs to be well categorised and clean when processed to be able to calculate reliable results from it. In Chapter 2 I describe four tree-based machine learning techniques that can be used to classify astronomical objects. I then outline a data analysis technique called MINT (developed to aid in genome trait prediction problems) that detects strong correlations in observed data and selects the most important features that relate to particular classification. I also describe how a novel software package called *treeinterpreter* can interpret a tree-based model in an attempt to understand not only which features are most important to a particular classification, but what values of the feature contribute most to a classification. The results from these methods will form the majority of my work in Chapter 3.

Turning to cosmological/gravitational models, it can be seen from the theory that the nDGP and ΛCDM models are very similar, especially on small scales. For these reasons it can be hard to tell models of gravity apart, especially when the two point statistics of these models can be so close. Conventional methods of measuring data have worked up to this point, but machine learning methods could help break degeneracies in these models. In Chapter 2 I describe a machine learning method called Convolutional Neural Networks (CNNs) that can classify image data. I use CNNs to classify dark matter particle simulations as either nDGP or ΛCDM gravity (Chapter 4 and try to unravel the CNNs' inner workings (Chapter 5) using two methods. In the first method, I create my own dataset with parameters I have defined to extract decision boundaries from the model, and in the second method I use a pre-trained CNN (that can already correctly classify models of gravity) to transform a random field into what it has learned a nDGP or ΛCDM dark matter particle simulation looks like. In Chapter 6 I will describe further work I have carried out and outline my conclusions.

Throughout this work, I adopt the standard Λ cold dark matter model with the best-fit cosmological parameters from Jarosik et al. (2011) (WMAP-Yr74), $\Omega_m = 0.267$, $\Omega_\Lambda = 0.734$, and $H0 = 71$ kms^{-1}Mpc^{-1}.

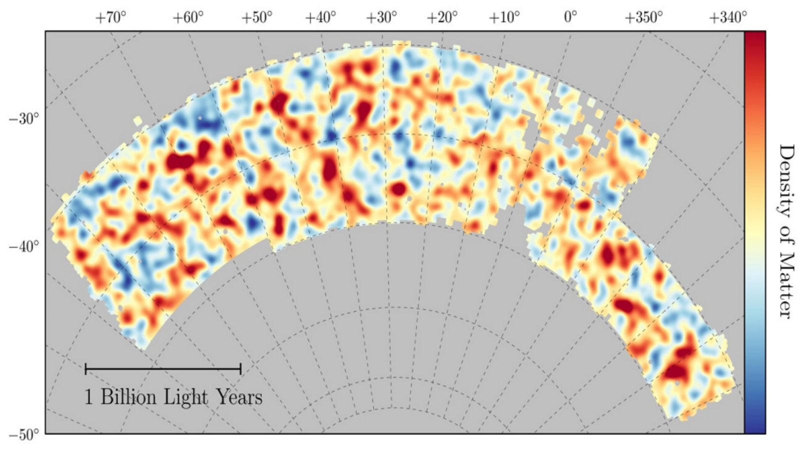


Figure 1.4: Map of dark matter made from gravitational lensing measurements of 26 million galaxies in the DES Y1 data. It covers over 1300 deg^2 and covers redshifts $0.2 < z < 1.3$. Red regions have more dark matter than average, blue regions less dark matter. (Chang et al., 2018).

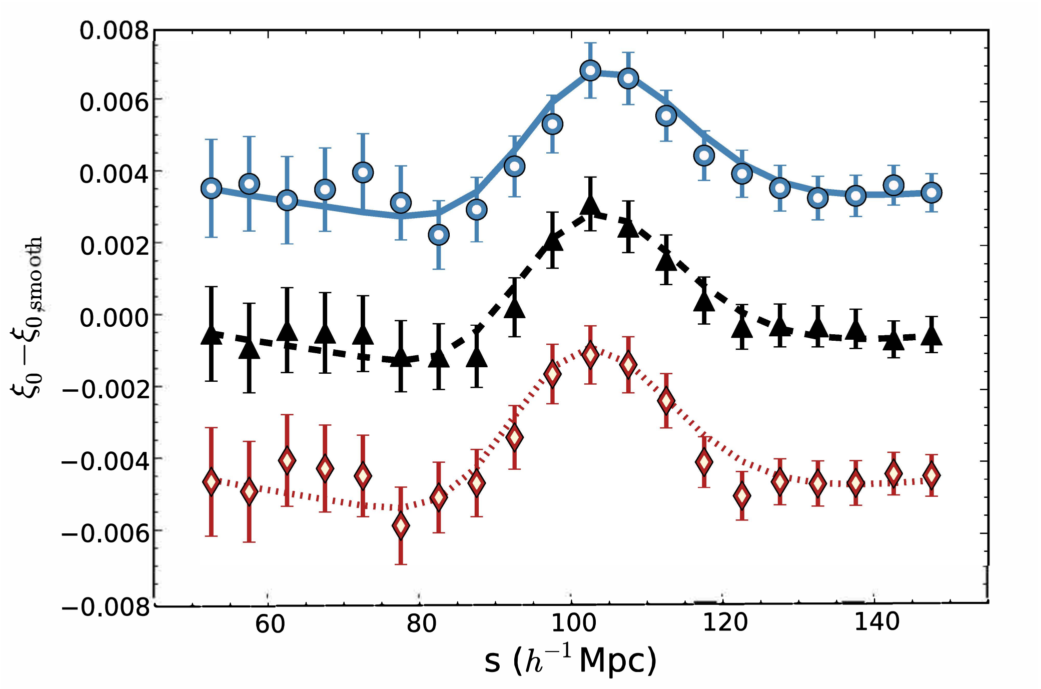


Figure 1.5: BAO from SDSS-III BOSS with configuration space correlation function ($\xi(s)$) on the y-axis and separation (s) on the x-axis. The colours represent redshift bins, where the red curve is $0.2 < z < 0.5$, black is 0.4 $< z <$ 0.6, and blue is $0.5 < z < 0.75$. The blue and red curves are shifted by an arbitrary value of $\pm$ 0.4 for clarity. (Alam et al., 2017).

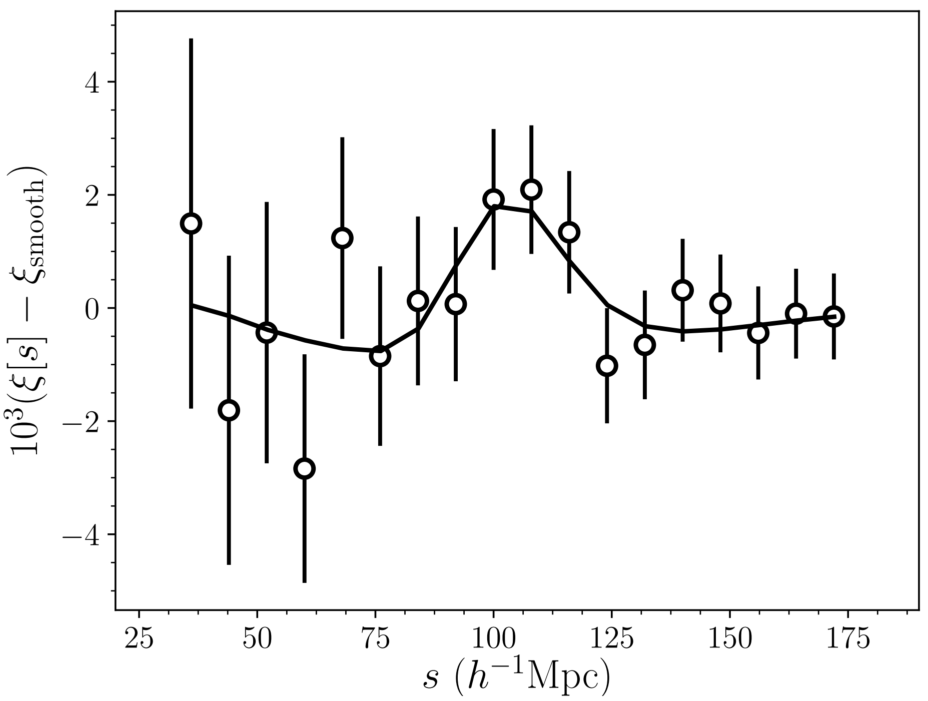


Figure 1.6: The eBOSS DR14 quasar spherically-averaged BAO signal, with configuration space ($\xi(s)$) on the y-axis and separation (s) on the x-axis. The smooth component of the best-fit model has been subtracted from the best-fit model and the measurements in order to isolate the BAO feature (Ata et al., 2018).

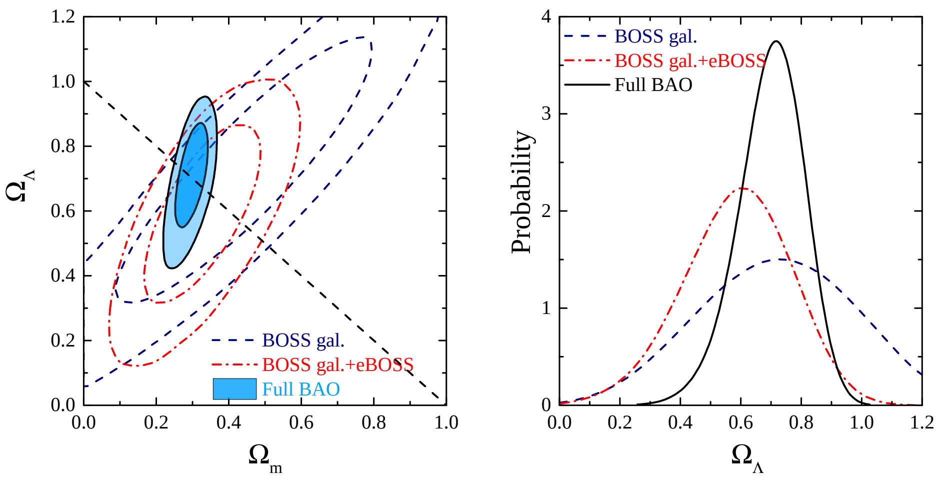


Figure 1.7: The left panel shows constraints on Ω_m and Ω_Λ using the 68% and 95% confidence levels for three sets of BAO data. The blue contour is using only BOSS sample, red is when combining it with the DR14 sample, and the filled blue contour is when using all available BAO data (including BOSS Lyα). The right panel shows the one dimensional probability distribution function for the value of Ω_Λ for each of the three sets of BAO data. (Ata et al., 2018).

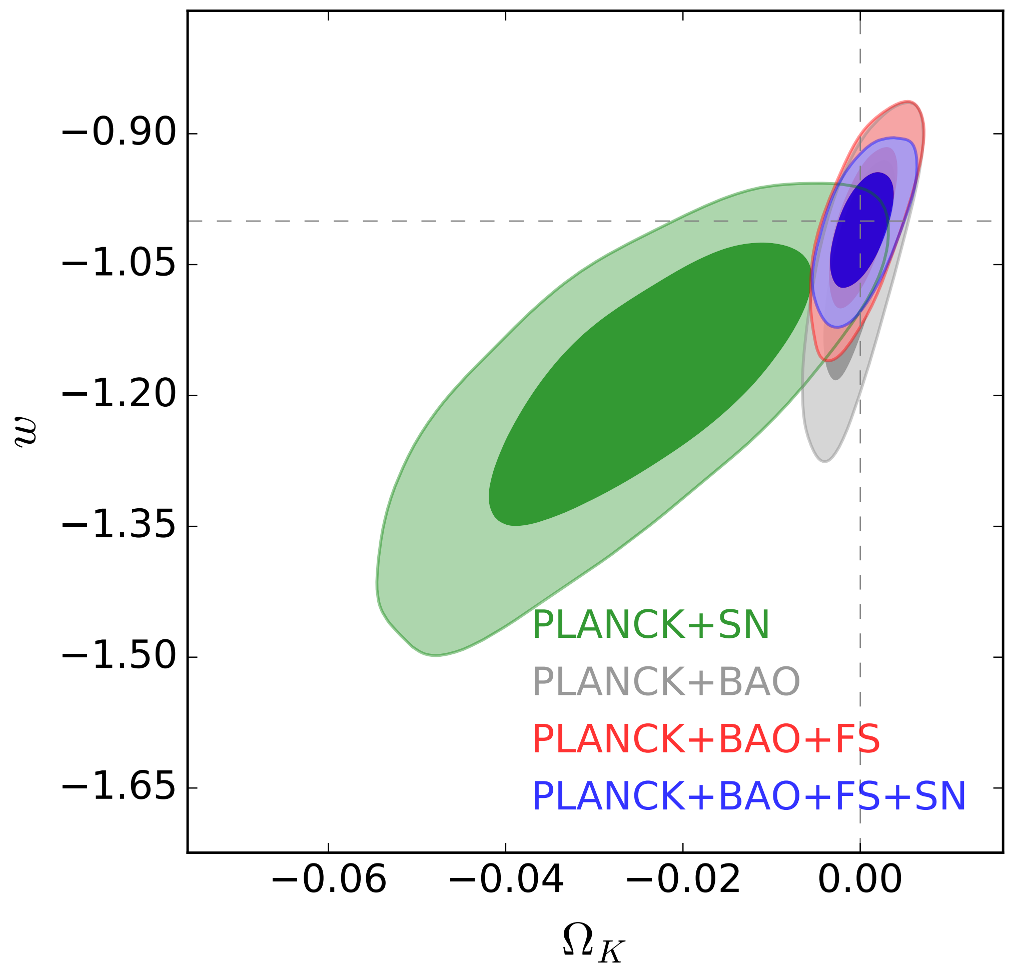


Figure 1.8: Contours showing constraints on w and Ω_K uses data from Planck 2015 results, SDSS Data Release 12 for the BAO measurements, and the Joint Lightcurve Analysis for the SNe data (shown on the plot as SN). FS relates to the use of the full shape of the galaxy power spectrum.

Chapter 2

Machine Learning Methods

Machine learning is a technique that allows computers to progressively improve on a given task without being programmed to do so. The term was first used by Arthur Samuel (1959) where he used a method called Artificial Neural Networks (ANNs, commonly shortened to neural networks) to teach a computer how to play checkers. The basic premise of machine learning theory is that algorithms built in a specific way can learn from trends in data and subsequently make predictions about new and unseen data.

Machine learning methods can either be 'supervised' or 'unsupervised' depending on the information available in the data. In supervised methods, each training sample has a corresponding label or target value as to what the sample represents. For example, in a machine learning application where a user was teaching a computer about what spiral or elliptical galaxies look like, a picture of a spiral galaxy could have the corresponding label of 'spiral'. Conversely, in unsupervised methods, no labels are assigned to training samples and the algorithm finds trends in the data by itself.

The most widely used applications of supervised learning are with classification or regression problems. A classification problem would be like the one mentioned previously, an image recognition problem where images are classified as containing either a spiral or elliptical galaxy (discrete labels), whereas a regression problem involves attempting to predict a continuous variable, such as redshift or magnitude.

In order for a machine learning algorithm to learn about trends in the data, the input variables must be considered. These are commonly knows as 'features' - individual measurable properties. Examples of these would be measures of object magnitudes, shape, or colour, and for image recognition techniques whole images can be used. It is important that the features that are chosen are discriminative and independent to ensure efficient learning.

There are many different types of algorithms that can be used to do machine learning; some popular ones include:

- Tree-based methods use a flowchart-like model to make decisions about a sample's target value (Breiman et al., 1984).
- Neural networks use an interconnected group of artificial neurons which model data in a non-linear fashion (Rosenblatt, 1957).
- Support vector machines (SVM) map samples in feature space separating categories as much as possible (Cortes and Vapnik, 1995).
- Bayesian networks use probability distributions of variables with conditional dependencies to separate data (Heckerman, 1995).

In this work, I will be considering the application of four *tree-based methods* to perform star-galaxy classification, and a deep learning method called *Convolutional Neural Networks (CNNs)* to perform classification of n-body simulations of either ΛCDM or nDGP dark matter particle simulations. In this chapter, I will outline these two methods using well known datasets: the Iris Flower Dataset for tree-based methods, and the Modified National Institute of Standards and Technology (MNIST) database of handwritten digits for CNNs. Section 2.1 will discuss each of the four tree-based methods used, the method behind how features are chosen for the star-galaxy separation work, and a method to analyse and interpret tree-based models. Section 2.2 will introduce the theory behind ANNs and CNNs, describing each component of the model, and possible ways of analysing and interpreting CNN models in order to understand what they have learned.

2.1 Tree-based Methods

This section introduces the machine learning algorithms used in this work, including the methodologies behind the Mutual Information based Transductive Feature Selection (MINT) feature selection algorithm (He et al., 2013) and a method to simplify ensemble methods based on decision trees called *treeinterpreter* (Saabas, 2015).

The birth of tree-based methods lies with the classification tree model for the Iris Flower Dataset that Fisher (1936) used to introduce linear discriminant analysis (LDA).

In Fisher's 1936 paper, he introduced a multi-variate dataset commonly known as the Iris Flower Dataset. Included in this dataset are 50 samples from three species of Iris flower, with each sample including four features: petal width and length, and sepal width and length. Much like the MNIST dataset described in Section 2.2.7, the Iris flower dataset has become a standard benchmark for testing new machine learning algorithms. In this work, I use this dataset to calculate a ranking of features in importance relating to correct classifications (a function included in scikit-learn python package called `feature_importance`), which can help in the process of interpreting the inner workings of a machine learning model. This is further explored in Section 2.1.4, where I show an example of the `feature_importance` calculation and also describe a newer method to better determine the importance of each feature called *treeinterpreter*.

As well as the Iris Flower Dataset, Fisher brought forward the idea of LDA, which finds a linear combination of features to separate data into two or more classes (e.g. spiral or elliptical galaxy). It does this by using the condition that the probability distribution functions that relate features to classifications and log of the likelihood ratios be larger than a set threshold, which would indicate the features belong to a particular classification. Whilst this isn't specifically the first example of a decision tree, Fisher did bring forward the dataset that is still used today and the idea of regression classifying, which is the end result of tree-based classification methods.

The first specific example of a regression tree algorithm was published by Morgan and Sonquist (1963) in their Automatic Interation Detection (AID) method. In the AID method, at the start of the tree (the root node) data is recursively split (or grouped) into two child nodes. The quality of the splits are measured and splitting stops when the quality surpasses a certain threshold (the modern day application of this process is more thoroughly explained in Section 2.1.2). The result which the AID method predicted was the end node sample mean, a continuous variable, therefore making AID a regression method. The THeta Automatic Interaction Detection method (THAID) used the ideas from AID to solve classification problems, choosing to split data as it progresses through the tree to maximise the number of observations in each category (Messenger and Mandell, 1972).

There was not much interest in these methods, as it was quickly shown that the AID method could severely overfit the data (Einhorn, 1972). This meant the model was able to predict the training data with high accuracy, but did not generalise well and was inadequate when attempting to predict on unseen data. It was also shown that there was a problem with feature selection in the trees - if there were two features that were highly correlated, only one would appear in the tree (Doyle, 1973). This made any kind of measure of the importance of features relative to each other unreliable.

These issues prevented tree-based methods from being adopted until the publication by Breiman et al. (1984), titled “Classification And Regression Trees” (CART). In this paper, a new method was employed to address the generalisation issues of AID and THAID. In CART, the trees would not use the AID/THAID method to determine when to stop splitting, but would rather grow a large tree and remove splits (pruning) to reduce the size of the tree to find a minimum cross-validation estimate of error. Also, to account for the unreliability in measuring feature importance in the AID/THAID method, CART had the ability to produce ’surrogate splits’ in the tree. This is when a series of splits are performed using alternate features, as a substitute for the original desired split. This has two main advantages: it can deal with missing data values in the original desired split, but also compare feature splits against each other (providing a score) providing a more reliable method of determining feature importance.

From CART, interest has developed in utilising ensembles of classifiers to make predictions on data. In section 2.1.1, I discuss the various ensemble methods used for the results presented in Chapter 3.

There is a more comprehensive review of the history of tree-based methods in Loh (2014).

2.1.1 Object Classification Using Tree Based Machine Learning Methods

In this section, various tree-based machine learning methods used in this work are described in more detail. Four tree-based machine learning methods are used in this work: Random Forest (RF, Breiman, 2001), Adaboost (ADA, Freund and Schapire, 1997; Zou et al., 2009), Extra Randomised Trees (EXT, Geurts et al., 2006) and Gradient Boosted Trees (GBT, Friedman, 1999, 2001; Hastie et al., 2009). I use the implementations of these algorithms from within the `scikit-learn` python package (Pedregosa et al., 2011) - tools for data mining and data analysis. All of these methods are able to draw a decision boundary in multidimensional parameter spaces which distinguishes between classifications. I describe these algorithms briefly below.

A decision tree is a flowchart-like model that makes ever finer partitions of the input features (for instance, photometric properties) of the training data. Each partition is represented by a branch of the tree. The input feature and feature value used to generate the partitions are chosen to maximise the success rate of the target values (for instance, point source or galaxy classifications) which reside on each branch. This process ends at leaf nodes, upon which one or more of the data sit. A new object is queried down the tree and lands on a final leaf node. It is assigned a predicted target value from the true target values of the training data on the leaf node. A single decision tree is very prone to over fitting training data.

Random Forests (RF, Breiman, 2001) train by generating a large number of decision trees, with each tree using a bootstrap re-sample of the training data and a random sample of the input features. During classification of new data the majority vote across all trees is taken. By building a model that takes a vote from many decision trees, the problem of over fitting the training set is overcome, allowing better generalisation to unseen data.

Extra Randomized Trees (EXT, Geurts et al., 2006) is a similar algorithm to Random Forests, but splits in the generated decision trees are decided at random instead of calculating a metric. This makes model training faster and can further improve generalisation.

Adaboost (ADA, Freund and Schapire, 1997; Zou et al., 2009) and Gradient Boosted Trees (GBT, Friedman, 1999, 2001; Hastie et al., 2009) are both examples of boosted algorithms, which convert so-called decision stumps into strong learners. Decision stumps are shallow decision trees (trees with a low depth) that result in predictions close to a random guess. The data is processed through these trees multiple times with the algorithm weighting the model based on performance. Adaboost changes the model between iterations by re-weighting the data of objects that were misclassified at a rate governed by the **learning_rate** parameter. This minimises model error by focusing the subsequent tree on those misclassified objects. Gradient Boosted Trees changes the model by iteratively adding decision stumps according to the minimisation of a differentiable loss function (which tracks misclassification) using gradient descent. The model will start with an ensemble of decision stumps and the loss will be assessed. Between each iteration the algorithm adds decision stumps that reduce the loss of the model, stopping when loss can no longer be reduced (when the gradient of reducing loss flattens).

2.1.2 Assessing Node Splits (Metrics)

A decision tree progresses from the whole dataset being split in a node at the top of the tree to small subgroups of the data being split at the bottom of the tree, and does this by choosing a variable at each node that best splits the data. The quality of this split is assessed using a metric, which generally assess the homogeneity of the target variable during the progression. In this work, I test and use two metrics when creating tree-based models, gini impurity (Light and Margolin, 1971), and information gain (or cross-entropy impurity) (Quinlan, 1986).

As said previously, at each node an assessment takes place before the data is split again. Given that there are training vectors x_i and target or label vectors y_i for a given dataset. If values 0, 1, ... K-1 are the target classification for node m in the decision tree, representing a region R_m with N_m observations, the proportion of classification k observations in node m is given as p_{mk}, where

$$p_{mk} = 1/N_m \sum_{x_i \in R_m} I(y_i = k). \tag{2.1}$$

Gini Impurity is then a measure of how often an element that is chosen randomly from the data residing on that node X_m would be incorrectly labelled if it was to be given a label at random. Its formulation is,

$$G(X_m) = \sum_k p_{mk}(1 - p_{mk}), \tag{2.2}$$

where G would be the measure of Gini Impurity.

Information Gain (Cross-Entropy Impurity) considers that given a sample reached this node that is being assessed, what is the expected amount of information required to specified whether a new example should be labelled in a particular way. It is formulated as,

$$IG(X_m) = -\sum_k p_{mk}\log(p_{mk}), \tag{2.3}$$

where IG would be the measure of Information Gain (Cross-Entropy Impurity). These metrics are quite similar, except that using information gain is slightly more computationally taxing due to the logarithmic part of the calculation. In this work, both of these metrics are tested when tuning the tree-based methods.

2.1.3 Feature selection using MINT

To aid in the interpretation of the results of tree-based ensemble methods, it would be advantageous to select only a small number of features, but chosen wisely such that they are minimally correlated with each other, and have strong classifying power (Andrew. Hall, 2000). Various feature selection methods have been explored in recent years in relation to object classification problems, such as the Fisher discriminant (Fisher, 1936; Soumagnac et al., 2015), or the previously mentioned `feature_importance` function provided in the scikit-learn package (Pedregosa et al., 2011; Hoyle et al., 2015). This work shows the first known application of the feature selection method known as MINT (He et al., 2013) to astronomical data.

To explain how MINT works, I start with an explanation of 'Maximum Relevance and Minimum Redundancy' (mRMR), another method of feature selection which can help to find a small number of relevant input features without relinquishing classification power. This has been proven to work in multiple datasets involving e.g. handwritten digits, arrhythmia, NCI cancer cell lines, and lymphoma tissues (Ding and Peng, 2005; Peng et al., 2005).

mRMR first calculates the maximum relevance, a feature selection method based on the measurement of mutual dependence (correlation) between the features. Maximum relevance measures the mean of all of the mutual information values (a measure of correlation) between unique pairs of individual features x_i, and classes c, with the aim of finding a set of features most correlated with a specific classification. The maximum relevance calculation is given by maximising D for the selected features S and class c where

$$D = \frac{1}{|S|} \sum_{x_i \in S} 1(x_i; c), \tag{2.4}$$

with I being the mutual information, and $|S|$ is the cardinality of the feature set S. Selecting features that are maximally relevant to the classification causes the set of returned features to be highly correlated with one another. To compensate for this, features that are highly correlated with other features are removed using minimum redundancy. This is calculated by minimising R for the selected features S where

$$R = \frac{1}{|S|^2} \sum_{x_i, x_j \in S} I(x_i, x_j), \tag{2.5}$$

with $I(x_i,x_j)$ representing the mutual information between features x_i and x_j.

We would like to maximise D (Equation 2.4) while minimising R (Equation 2.5). This can be simplified, completing the mRMR calculation by combining these requirements in one equation, and maximising Φ where

$$\Phi = D - R. \tag{2.6}$$

This ensures the returned set of selected features is highly correlated with the classification, but are mutually exclusive from other features in the set.

This work uses an extension of mRMR called Mutual Information based Transductive Feature Selection (MINT) (He et al., 2013), a method designed to help with the 'curse of dimensionality' in genome trait prediction. This arises due to the issue of having many more features than samples in the dataset. MINT assesses the mutual information between the training sample's features and classification and, setting it apart from mRMR, between individual features in both the training and test sample.

This means that MINT can effectively combine Equations 2.4 and 2.5 into Equation 2.6, the same as mRMR, but is able to exploit a much larger amount of data due to the assessment of the correlation between features for the entire sample, not just the training sample.

I will now consider an expanded version of Equation 2.6 with the MINT modifications included. The incremental search for features using MINT works in the following way; I assume we have a set of X total features, and

S_{m-1} as a subset of those features containing $m-1$ features. The m-th feature is selected from the remaining feature set, $X - S_{m-1}$, by maximising Φ in the same way as in Equation 2.6, as follows:

$$max_{x_j \in X - S_{m-1}} \\ [I(x_j^{\texttt{Tr}}; c^{\texttt{Tr}}) - \frac{1}{m-1} \sum_{x_i \in S_{m-1}} I(x_j^{\texttt{Tr + Test}}; x_i^{\texttt{Tr + Test}})]. \tag{2.7}$$

The modifications are made clear by the indication of which sample set is being used in the mutual information calculations, either only the training (`Tr`), or both training and test (`Test`) samples.

I follow He et al. (2013) and explore the high dimensional feature space using the greedy algorithm (Vince, 2002). In the case of MINT, greedy means that parts of the calculation are performed dynamically - utilising previously calculated values in the MINT algorithm for future MINT calculations - making the feature selection process vastly quicker.

A user defined number of features is selected using the MINT algorithm, thus reducing the amount of input data (by reducing the number of features) required to make a robust prediction for the test sample. In this work, I have incorporated python code kindly provided by Ben Hoyle to calculate MINT feature selections in Chapter 3.

2.1.4 Interpreting Models of Tree Based Methods

I use tree-based machine learning methods because they are robust, difficult to overfit, and have methods available to aid in interpreting them (Hastie et al., 2009). By examining the decision trees created by the algorithm, the inner workings of the model can be understood. However, when the data are vast and complex and an ensemble of trees is used, the scope of the model deepens to such a degree that interpretation becomes near impossible. It is for this reason that new methods of model interpretation must be investigated.

The possible structure of an example decision tree from a Random Forest with no limit on the hyperparameter 'maximum depth' is seen in Figure 2.1.

Starting from the top and working down the tree, an object would advance through the tree in one direction or another towards the leaves (predicted class) at the bottom. It is clear from the complexity of the tree that it is unfeasible to easily gain information relating to the inner workings of the model by simply looking through the trees. This is especially the case since each tree will have drawn different decision boundaries relating to specific types of objects. For example, one tree may be very good at classifying red point sources, while another may excel at classifying blue galaxies.

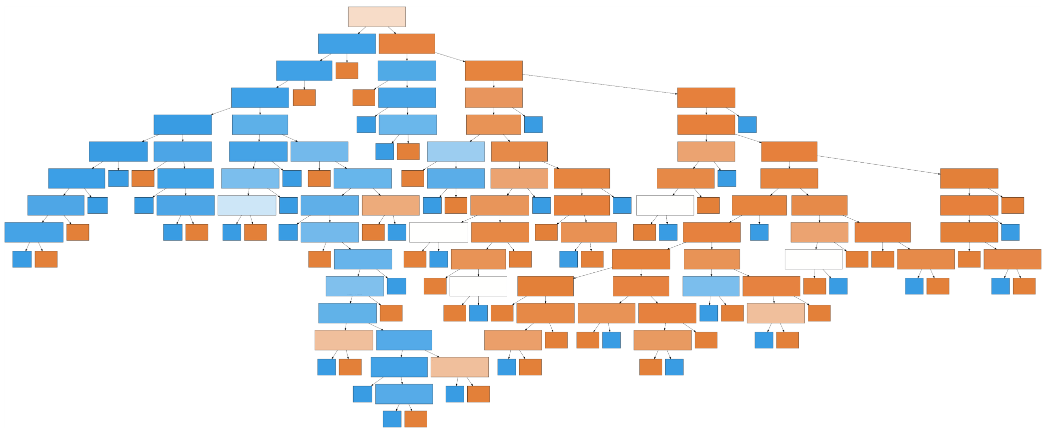


Figure 2.1: Example of a single decision tree from a Random Forest comprised of 256 trees with unrestricted maximum depth. Blue colours indicate one classification, while orange colours indicate a different classification. Opacity of colour represents probability of classification with more solid colours denoting higher probabilities.

There are methods for determining which features are important to the machine learning model, such as the `feature_importance` function provided in the scikit-learn package (Pedregosa et al., 2011). This is sometimes referred to as the 'mean decrease impurity', which is the total decrease in node impurity, an assessment of how well the model is splitting the data, averaged over all of the trees in the ensemble (Breiman et al., 1984). This is to say that the features in the model are assessed, and if they consistently contribute to making classifications, their importance increases. This is useful, but somewhat ambiguous as it does not give much insight into the individual decisions the trees make, such as where it is most efficient to draw a boundary in parameter space. An example of the result of the `feature_importance` function performed using default settings for a Random Forest on the Iris Flower dataset is shown in Figure 2.2. When looking at this figure, it is clear that petal width is the most important feature when classifying the types of Iris flower, however, no insight is given about the importance of any specific width.

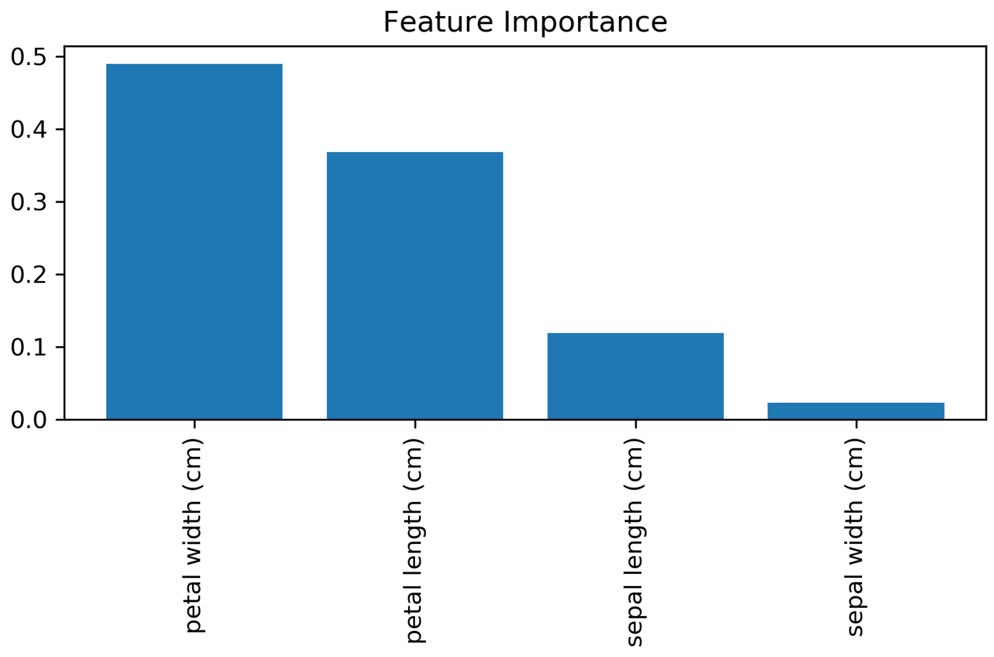


Figure 2.2: Feature Importance using a Random Forest with default settings on the Iris Flower dataset.

Instead, a python package called *treeinterpreter*[1] (Saabas, 2015) can be used in an effort to decipher this information. For each object, *treeinterpreter* follows the path through the tree, taking note of the value of the feature in question every time it contributes or detracts from an object being given a particular classification. This means that one can investigate how much the

[1] `https://github.com/andosa/treeinterpreter`

value of a particular feature contributes to the probability of a certain classification. This is the first known application of *treeinterpreter* to astronomical data (on the suggestion of Ben Hoyle).

To learn how *treeinterpreter* works, I start with the mathematical description for a prediction given by a single tree. The probability of a particular object being a member of class c is given by the prediction function $f(x)$, where x is the feature vector for the object in question. In the case where $f(x)$ is obtained from a single tree, I have

$$f(x) = c_{full} + \sum_{k=1}^{K} contrib(x, k), \tag{2.8}$$

where c_{full} is the initial classification bias due to the class distribution in the sample for the class c, and $contrib(x, k)$ is the contribution from feature k in the feature vector x to the probability of being classified as class c. An object traversing through the tree from the root to the leaves follows this path: at the root node, the probability of class c is the classification bias c_{full} and if there were no further splits in the tree, the probability that any object in the test sample was of class c would remain at the initial classification bias c_{full}. I.e. my sample could be 63% galaxies, so any object starting in a single tree would have a 63% probability of being a galaxy. As an object's feature is questioned by the node (e.g. `FIBERMAG_G` < 22), the probability of it being classified as a galaxy deviates from that 63% by a small amount; defined by the percentage of the training sample of class c that satisfied the node criteria. It is this contribution, $contrib(x, k)$, and those from subsequent nodes that are summed to give the overall prediction for the object.

Extending this to an ensemble of trees is fairly straightforward; the overall prediction function $F(x)$ from a Random Forest is the average of those of its trees $f_j(x)$ (Breiman et al., 1984),

$$F(x) = \frac{1}{J} \sum_{j=1}^{J} f_j(x), \tag{2.9}$$

where the number of trees is given as J.

There is one last consideration to account for in the *treeinterpreter* calculation; if each decision tree has been built using a bootstrap of the whole sample, the initial bias of the tree, c_{full}, will be different for each tree. It is for this reason that the bias terms of each tree are averaged and added to the average contribution of each feature. This makes the full equation in *treeinterpreter* (Saabas, 2015) for the prediction function

$$F(x) = \frac{1}{J}\sum_{j=1}^{J} c_{j_{full}} + \sum_{k=1}^{K}(\frac{1}{J}\sum_{j=1}^{J} contrib_j(x,k)). \tag{2.10}$$

This not only presents which features are important to a particular classification in the model overall, but also which features were important for the individual classification of each object. As the value of the feature for each object in the data is known, where in parameter space the model is succeeding or failing can be determined.

2.2 Artificial Neural Networks

An Artificial Neural Network is made from interconnected layers of artificial neurons. To understand modern ANNs, the understanding of the 'perceptron' (Rosenblatt, 1957) is the natural starting point. A perceptron is an algorithm that takes multiple inputs, and produces a binary classification, above or below a set threshold. A simple diagram of this is shown in Figure 2.3, where $x_{1,2,3}$ are inputs and the circle is the perceptron.

Each input x has a corresponding weight w and the perceptron is assigned a bias b, with the weight being a value that corresponds with the input's importance to the output, and the bias changing how easy it is to pass the threshold. The output to the perceptron can be defined as the weighted sum of the weights and inputs $w \cdot x = \sum_j w_j x_j$ plus the bias,

$$\text{output} = \begin{cases} 0, & \text{if } w \cdot x + b \leq 0 \\ 1, & \text{if } w \cdot x + b > 0 \end{cases} \tag{2.11}$$

where the cases 0 or 1 are whether the perceptron was activated or not. This equality condition, in this case a step function, applied to the summed weights and biases is called an 'activation function'. This current model of a neuron acts the same as a NAND logic gate and when arranged in layers (a network), these perceptrons can compute any standard logic function. Learning algorithms could automatically choose the optimum weights and biases for our network by changing them a small amount, and then assessing the subsequently changed output. The quantification of this change is called a loss or cost function, which will be better defined in Section 2.2.2. The only problem with this idea is that with perceptrons, because of the binary nature of them, changing the weights and biases a small amount could cause the output to change a considerable amount. This would occur when a changed output from 0 to 1 propagates through the layers, changing subsequent layers in the network in a complicated way.

A solution to this problem is to change the perceptrons to 'sigmoid neurons', changing the activation function at the end to a sigmoid function. This would allow the inputs and outputs to these neurons to be continuous values from 0 to 1. This now means that a small change in the weights and biases in the neuron results in a small change in the output, and allows the network to be tuned for better results.

Since the application of the sigmoid function, research into activation functions has progressed and newer ones such as ReLU (Hahnloser et al., 2000, 2003) or Leaky ReLU (Maas, 2013) have proved to be more successful. The name ReLU means 'Rectified Linear Unit', where a linear unit would be an activation of $x = y$, and it is rectified in the respect that it only considers positive values. A visual representation of what these activation functions look like is in Figure 2.4.

2.2.1 Backpropagation

The idea of backpropagation was first theorised by Rumelhart et al. (1988) and works by repeatedly tuning the weights in the network, then assessing the

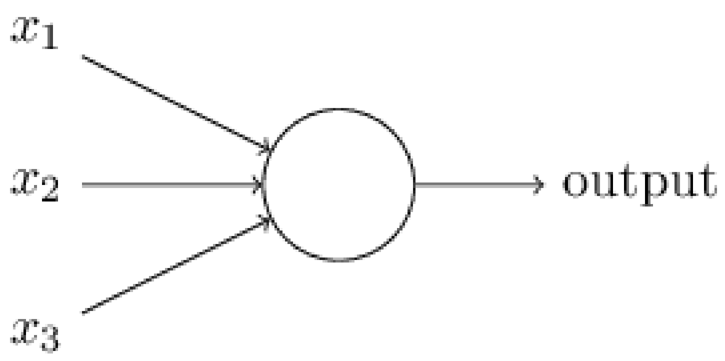


Figure 2.3: Perceptron diagram (Nielsen, 2015)

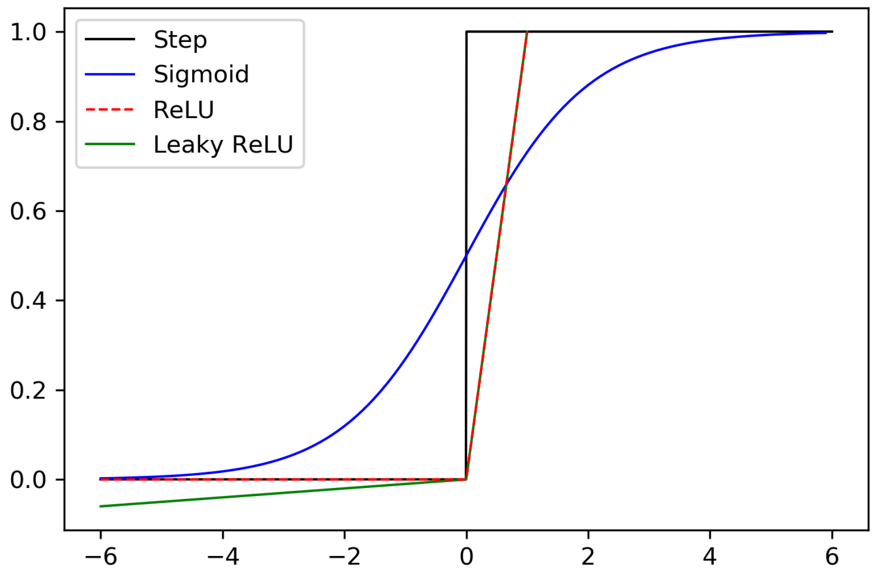


Figure 2.4: Activation functions

difference between the output and the desired output (the loss or cost functions defined in Section 2.2.2). This is the process that allows the network to 'learn' about the inputs, and adapt the weights and biases to more accurately predict desired outputs. This allowed neural networks that contained hidden layers - layers of neurons in between the input and output layers which allow for much more complex functions to be modelled in ANNs. An example of an ANN architecture with one single hidden layer is shown in Figure 2.5.

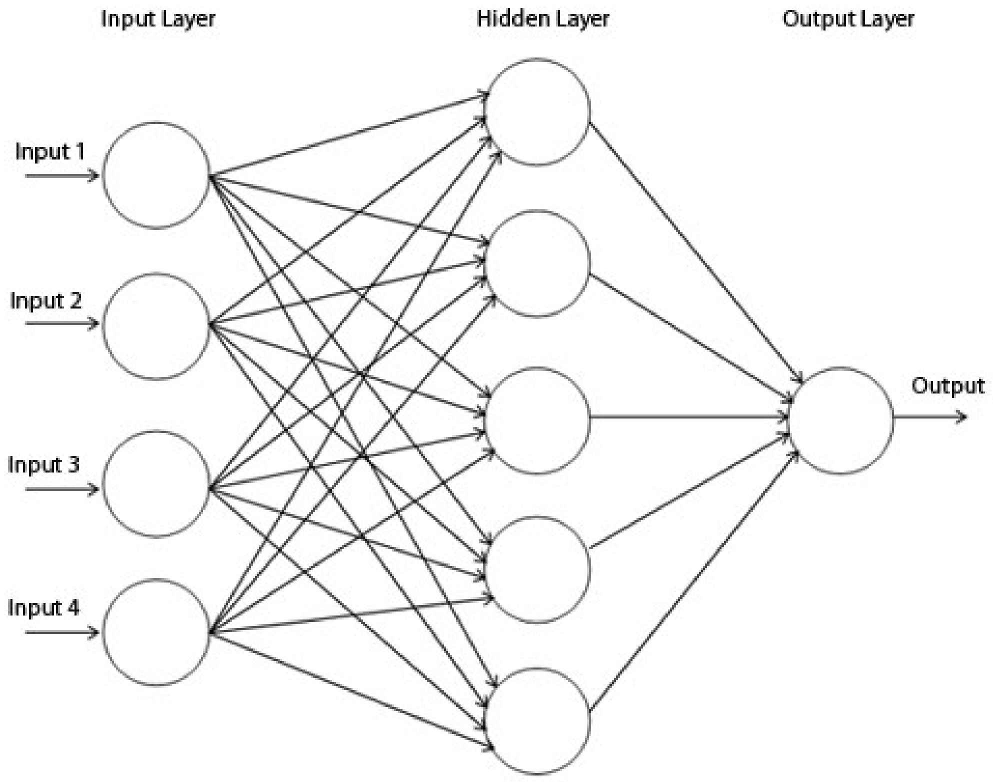


Figure 2.5: ANN with single hidden layer (Image taken from https://www.nicolamanzini.com/single-hidden-layer-neural-network/)

2.2.2 Loss functions

The way that I assess predictions of a classification model is to use cross-entropy loss (de Boer et al., 2005) or log-loss, which gives a probability value between 0 and 1. The loss increases as shown in Figure 2.6 as the predicted class moves further away from the true class. In the context of this figure, this means if the true class was assigned the label 1, and the model predicted the label of 1 with a probability of 0.2, the loss would be the high value of 2. The model is highly penalised for incorrect predictions due to the exponential nature of the loss at the lowest probabilities. In theory, a perfect model would have a loss value of 0.

If the classification is binary,

$$\text{cross entropy} = -(y\log(p) + (1-y)\log(1-p)) \tag{2.12}$$

where y is the true classification and p is the predicted classification. If there is more than two possible classifications (multiclass), a separate loss is calculated for each class c per sample s and the result is summed over all classed M:

$$-\sum_{c=1}^{M} y_{s,c}\log(p_{s,c}) \tag{2.13}$$

This is the main loss function used in Chapters 4 and 5 of this work.

2.2.3 Gradient Descent Optimisation Algorithms

The way neural networks are tuned is by using a technique called gradient descent (Robbins and Sutton, 1951; Kiefer and Wolfowitz, 1952). Gradient descent is an optimisation algorithm that iteratively tweaks some parameters of a function to minimise the loss or cost. This is analogous to a ball rolling down a hill and finding the minimum point in a valley. The general process is as follows:

- Parameters are initialised and corresponding loss is calculated,

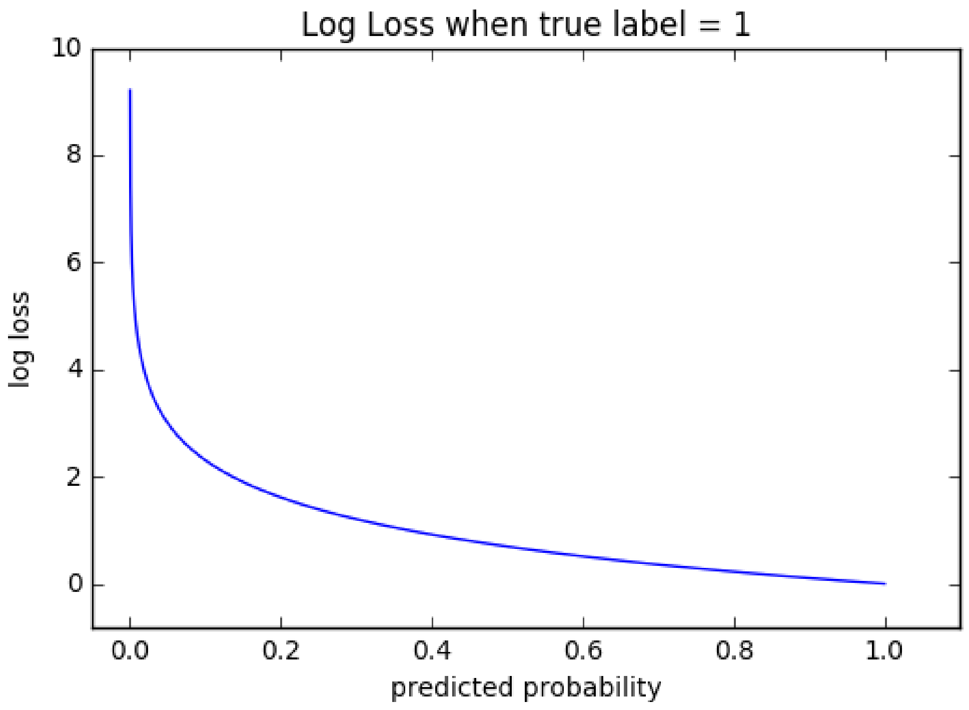


Figure 2.6: Cross-entropy loss (Image taken from https://github.com/bfortuner/ml-cheatsheet/blob/master/docs/loss_functions.rst)

- Parameters are changed by a small amount (governed by the learning rate) and the loss is recalculated,
- Gradient of the change in loss is calculated,
- If the loss is reducing, the parameters continue to be changed in the same way and the process is repeated until convergence is reached.

If the learning rate is set too high the optimal parameters may be overlooked and the loss will oscillate; if the learning rate is too low, the optimal parameters may never be achieved.

Gradient descent can be performed in a number of ways regarding the number of training examples considered per parameter update. Either the gradient is calculated after the whole dataset has been processed (batch gradient descent), after each training example (stochastic gradient descent; SGD), or somewhere in between (mini-batch gradient descent). Each method has advantages and disadvantages; for example, batch gradient descent can be slow and cumbersome due to memory limitations with large datasets but will always converge to the global or local minimum depending on the error surface. SGD can cause the loss to fluctuate wildly as it does not consider many similar training examples simultaneously, but this allows it to find more optimal local minima when complex error surfaces occur. In this work, I utilise mini-batch gradient descent with a batch size set at 25.

It can be seen that as the idea of gradient descent is developed, there are many methods one can employ to arrive at an optimal set of parameters, spawning many different optimisation algorithms. One popular idea throughout these algorithms is to vary the learning rate through a parameter called momentum (Rumelhart et al., 1988) or something similar, helping to achieve convergence faster and reducing oscillations. A momentum term increases the parameter updates when the loss is reducing, and decreases the parameter updates when the loss is increasing.

The gradient descent optimisation algorithm used through Chapters 4 and 5 of this work is Adaptive Moment Estimation (Adam) (Kingma and

Ba, 2014). Adam computes adaptive learning rates for each parameter and stores an exponentially decaying average of past gradients, which is similar to a momentum term but gives Adam a more complex form of hysteresis. The authors of the algorithm show that it works well empirically and compares well to other forms of adaptive learning methods.

2.2.4 Data Augmentation

A commonly used technique in deep learning is that of 'data augmentation' (Yaeger et al., 1996), a preprocessing step where the dataset is used to create new samples that the machine learning algorithm considers as new data. This means it does not identify the modified sample as the same as one it has previously seen, as this would cause problems in the bias of the dataset. Using data augmentation can improve regularisation and help reduce overfitting of the training set due to the increased number of samples, and is also useful if there is limited data or storage space available. In terms of an image dataset, there are several well tested ways of performing data augmentation, such as cropping, rotating, or flipping the images. The method of data augmentation used in this work is detailed in Section 4.

2.2.5 The Neural Network Learning Process

Every time the network looks over a set number of examples in a training set, this is called one epoch. After one epoch has occurred, the network is validated using a set number of examples from the test data (the performance of the network is assessed) until either the user stops the training manually, or the training is set to be stopped automatically when a certain point is reached. This point could be when the loss stops falling after a few epochs or the networks achieves a certain level of performance.

Now that the ideas of loss functions, gradient descent, batches, and epochs have been established, exactly when the network learns can be better established. During each epoch the neural network looks at the training data in batches and after each batch is seen by the network, backpropagation occurs. This is when the weights and biases are tuned in the network through monitoring the loss function. This is a relentless process, with weights and biases being updated often (more than) thousands of times per epoch. It is through this method that the error function surface (the local and global maxima and minima of loss) is explored, and a viable set of weights and biases are discovered in a neural network.

2.2.6 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks are a category of deep learning artificial neural networks and have been commonly applied to visual based machine learning tasks such as recognition or classification. Examples of this include being able to recognise animals or objects in images, different characters or digits, or sentiment in sentences. One of the first CNNs that became popular was called LeNet5 by Lecun et al. (1998), where he identified that using individual pixels as features do not take advantage of the fact that features in images are highly spacially correlated. The CNN architecture he used was very complex, subsampling the training images (called MNIST and described in Section 2.2.7) through many layers to select important features from the images and eventually passing those features to a multi-layered neural network to classify them. Many modern CNN architectures, including the ones employed by this work, follow the same principles.

CNNs can analyse images of colour or greyscale; when using a coloured image, each of the red blue and green components of the image are split into channels that the CNN can analyse and identify features in. In greyscale images such as the MNIST dataset, only one channel is required.

In this work, I utilise a python package called *Keras* (Chollet et al., 2015) (which is built upon Tensorflow (Abadi et al., 2015)) to create CNN models.

2.2.7 MNIST Database of Handwritten Digits

The MNIST database (Lecun et al., 1998) is a large collection of digitised handwritten digits, created from a combination of two National Institute of Standards and Technology (NIST) databases. These databases were originally sourced from high school students (Special Database 1) and employees of the United States Census Bureau (Special Database 3). The database contains 60,000 training samples and 10,000 test samples, each normalised to produce an image of 28x28 pixels, with pixel intensities ranging from 0 (white) to 255 (black).

Much like the Iris Flower dataset described in Section 2.1, the MNIST database has been adopted by the machine learning community as a standard benchmark for machine learning algorithms, with a number of scientific papers looking to achieve the lowest error rate. The creators of the database obtained an error rate of 0.8% using SVMs (Lecun et al., 1998), but this has since been improved upon using CNNs, achieving an error rate of 0.23% (Cireşan et al., 2012), or to 0.21% using new regularisation techniques (Wan et al., 2013). A new and extended version of the database has since been released as of 2017 called Extended MNIST (EMNIST). Derived from NIST Special Database 19, it contains 240,000 training images and 40,000 test images. The MNIST database has been used to perform a number of tests relating to the model sensitivity analysis methods detailed in Section 2.2.9.

2.2.8 Layers in CNNs

Figure 2.7 shows the layers used in the LeNet5 CNN architecture. From left to right, the layers are: the input layer with example MNIST training sample, convolutional layers and pooling (subsampling) layers interlaced, then they are passed through full connections to the multi-layered neural network and the classification output layer. Each of these layers will be described in the following subsections.

Convolutional Layers

Consider the input image to a convolutional layer as a matrix of pixel values. For example, in the case of the MNIST dataset, a 28x28 matrix of pixel values ranging from 0 (white) to 255 (black). The steps to computing a convolutional layer are:

- Choose a ‘filter’ or ‘kernel’ - a separate smaller matrix with randomly initialised values,
- Move the filter across the input image starting at the top left, this is called a ‘stride’,
- Between each stride an element wise matrix multiplication is computed and the sum of the matrix is taken.

The resulting matrix is smaller and is commonly known as a ‘convolved feature’ or ‘feature map’, as its purpose is to detect or map features. This means the filter essentially becomes a template matching operation in the CNN. Having multiple filters with different properties that are learned during the training process allows the CNN to pick out many different features in the same layer. At this point in the CNN, after the convolution operation, it is normal to include an activation layer to introduce non linearity into the network, which is required to ensure the a network with multiple layers can learn complex functions. Without introducing non-linearity to the network, regardless of the number of convolutions or hidden layers, the network would compute a linear transformation of the input values.

Pooling Layers

Pooling layers are included in CNNs to reduce dimensionality while retaining important information about the image. There are different ways of pooling information such as maximum, average and sum. Figure 2.8 shows an example of a max pooling operation with the stride parameter set at 2 on a 4x4 feature map on the left with the resultant 2x2 filter on the right. The maximum value is taken from each coloured 2x2 stride, creating the new pooled filter. In the LeNet5 architecture and in this work, pooling is performed after each convolution (and applied activation).

Pooling is useful for this work for two main reasons: reducing the number of parameters in the model helps to prevent overfitting and the model becomes more generalised, as small changes in inputted samples do not change outputs from pooling (Lecun et al., 1998).

Other Layers

There are various other layers or operations commonly used in finalising the architecture of a CNN. These layers include,

- Batch Normalisation layer: Performing a normalisation operation between each layer of a neural network has been shown to improve performance and stability (Ioffe and Szegedy, 2015). The inputs passed to each layer after batch normalisation are of zero mean and unit variance, achieved by scaling the activations.

- Flatten layer: This is an operation where all of the feature maps and dimensions of the feature maps are organised into a 1D vector, ready to pass to a fully connected layer of neurons. It is equivalent to performing a dimension transformation in common programming languages and is also required (for instance) to save a 3D vector to a standard text file.

- Softmax layer (Bridle, 1990): This is the final layer in a neural network classifier, and provides a probability distribution over K different classifications. It is defined as $\sigma(z)_j = e^{z_j} / \sum_{k=1}^{K} e^{z_k}$ for $j = 1$ to K, where a K dimensional vector or arbitrary real values (as would be outputted by the CNN) are normalised into a K dimensional vector $\sigma(z)$ where the components sum to 1.

A table of the CNN architecture used in this work can be found in Table 4.1.

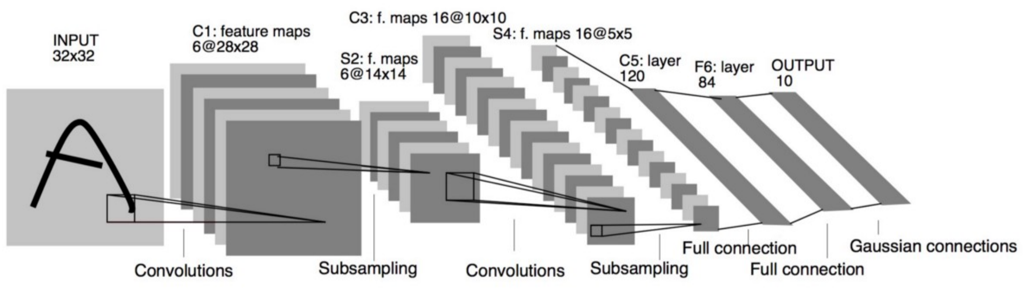


Figure 2.7: LeNet5 CNN architecture (Lecun et al., 1998)

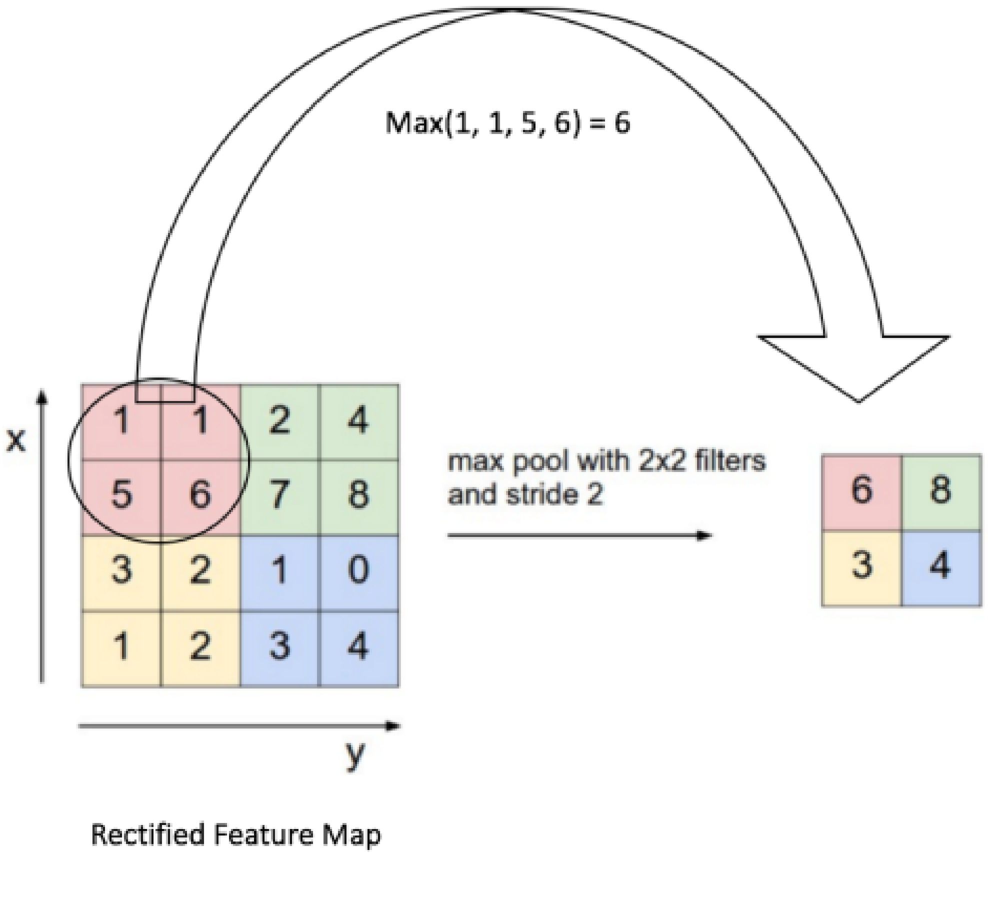


Figure 2.8: Max pooling example (Image taken from https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/)

2.2.9 Interpreting CNNs

While it is useful that CNNs can achieve such great accuracies in image classification tasks, there are many reasons why understanding how models make their decisions is important to study. For instance, if CNNs were to be used to drive medical decisions such as classification of malignant tumours, or safety decisions such as in an accident involving self driving vehicles, why the model is behaving a certain way becomes very important to justify the use of them. While this reason for model interpretation is important, it has been shown in previous work that model interpretation can be used in astronomy to verify current techniques, or enlighten the community to previously overlooked information (Morice-Atkinson et al., 2018). In this section, I outline the methods I have used to interpret why the CNN models I create in Section 4.2 are successful when classifying dark matter particle simulations of different models of gravity. The results from these methods are shown in Section 5.1.

Model Sensitivity Testing Methods

Generated Gaussian Random Fields

One way of inferring which features a trained CNN uses to classify dark matter particles as either ΛCDM or nDGP gravity would be to perform classification on a dataset with a subset of the features known to be present in the training dataset. The reason a subset would be used is because it would reduce the number of free parameters in the dataset. The method I use to do this is as follows:

- Train a CNN to classify ΛCDM and nDGP dark matter particle simulations to a high accuracy.
- Measure the matter power spectrum of the dark matter particle simulations in the training dataset, allowing the two point correlation function to be parameterised for ΛCDM and nDGP gravity.

- Create a set of power laws that are very similar to the matter power spectrum measured in the ΛCDM and nDGP simulations. Some power laws with different amplitudes, some with different gradients.
- Use the created power laws to generate numerous gaussian random fields with the same spacial dimensions as the training set.
- Allow the trained CNN to attempt to classify these generated gaussian random fields as if they were real dark matter particle simulations.
- Identify which power laws classified as ΛCDM or nDGP gravity.

In doing this, I will be able to determine which power laws the CNN prefers to classify as one model of gravity or the other. Perhaps one model of gravity will prefer high amplitudes on large scales and small amplitudes on small scales, and the other model will prefer a more equal amplitude at all scales. This investigation will be performed for CNN models trained using: ΛCDM and nDGP simulations that are very different when comparing their power spectra, ΛCDM and nDGP simulations with more similar power spectra, and ΛCDM and nDGP simulations where the amplitude in their power has been removed (i.e. they have been normalised).

Activation Maximisation

In this work I attempt to interpret the CNN model using a technique called 'Activation Maximisation' (AM) contained in a python package called *Keras-vis* (Kotikalapudi and contributors, 2017). The idea behind AM is to create an input image of a chosen classification using the CNN model by monitoring a new loss function, AM loss. This is done through gradient descent iterations where $\frac{\delta \text{AM Loss}}{\delta \text{input}}$ is monitored - i.e. the input is changed as the AM loss reduces.

The basic process is as follows:

- A CNN model is trained using a training set, and validated as one that performs well,

- A new input image is created with the same dimensions as the dataset, with pixel values drawn from a gaussian distribution with set mean and variance. In the case of the MNIST dataset, it would be a 28x28 array of pixels,
- A desired classification is chosen,
- This new input image is processed through the CNN model, and the AM loss is assessed.
- The input is then changed and the AM loss is assessed again, and the gradient of the loss is computed. If the AM loss is reducing, the input data sample has been changed in the correct manner and will continue being modified in the same way.
- The AM image is output, and resembles what the CNN would classify as the desired/chosen classification with high accuracy

Through this process, the new input image will slowly change from Gaussian distributed values to values that resemble the desired classification, as the activation of the parameters in the CNN model is maximised. When the AM loss no longer reduces, assessing the resulting new input image created from the CNN model should yield insights into how the model makes correct classifications.

Figure 2.9 shows what a CNN trained on MNIST data (to a classification accuracy of $> 99\%$) produces when it is tasked to change a grid of Gaussian distributed values to what the CNN would classify as a number 8 with a high accuracy. It can be seen that the resultant AM image does resemble a number 8, but it looks quite different to the MNIST training images.

There are various parameters which can be changed when performing activation maximisation. They are briefly described as: the output limits of the created AM image (set to a minimum of -1 and a maximum of 1 in my case), and tolerances for AM loss, and two regularisation parameters called 'LPNormalisation' (Riesz, 1910) and 'Total Variation' (Mahendran and

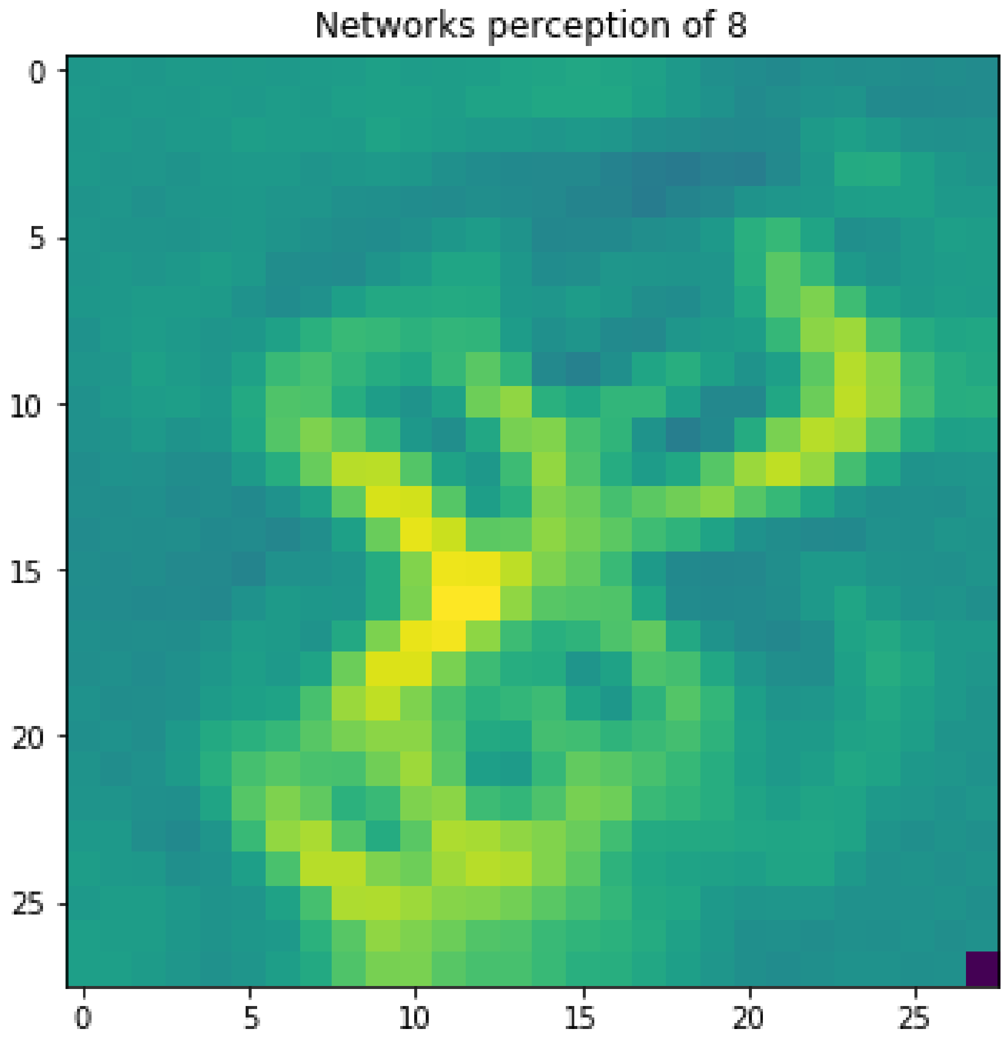


Figure 2.9: Activation Maximisation image example for MNIST (the number 8). (Image generated from https://github.com/raghakot/keras-vis/blob/master/examples/mnist/activation$_{m}aximization.ipynb$)

Vedaldi, 2015). The LPNormalisation regularisation in this context helps to control the intensities seen in the AM image, and the Total Variation regularisation encourages more coherent image structures (Kotikalapudi and contributors, 2017). These loss parameters are kept to their default values throughout this work. In Section 5.2 I use the CNN models to create AM images of simulations (hereafter AM simulations) for each ΛCDM and nDGP gravity and I then analyse the outputs. I use this analysis to compare the AM simulations to the original simulations I used to create the CNN models to gain insight into what characteristics the model has been able to learn about. This is the first known application of AM methods to cosmological data.

Chapter 3

Source Separation Using Machine Learning

An important and long-standing problem in astronomy is that of object classification; for example, whether an object in a photographic plate is a nearby star or a distant galaxy. Independent of the data-sample under investigation, the process of building a source catalog will require object classification. There are multiple ways of determining the classification of astronomical objects, each with their own advantages and disadvantages. For example, template fitting methods applied to photometric (Baum, 1962; Puschell et al., 1982) or spectroscopic data (Cappellari and Emsellem, 2004; Sarzi et al., 2006) can be accurate but are dependent on the choice of templates; whereas classifying objects by radial profile (Le Fevre et al., 1986) can be quick, but of limited accuracy due to the small amount of information used for each object. For instance, radial profile data alone cannot be used to distinguish between point sources, such as stars and QSOs.

There are successful complex point source separation methods in use to identify astronomical objects, such as likelihood functions (Kirkpatrick et al., 2011), where an object is classified as a QSO based on the summed Gaussian distance to every object in a set of known QSOs and stars in colour space. There are also complex machine learning methods for object classification that exist, such as Artificial Neural Networks that use photometry to isolate high redshift QSOs (Yèche et al., 2010), or objects at fainter magnitudes (Soumagnac et al., 2015). A recent investigation has also been performed into source separation using multi narrow-band data with Convolutional Neural Networks (Cabayol et al., 2018). A comparison of many of these methods applied to Dark Energy Survey Y1 data can be found in Sevilla-Noarbe et al. (2018).

This work aims to introduce a new combination of machine learning data analysis methods to astronomy[1], specifically with the use case of object classification, although it is noted that these methods can be readily applied to other problems. The goal is to use machine learning to improve the precision/purity of object classification from photometric data, while simultaneously analysing the generated machine learning models in an effort to understand the decision making processes involved. The object classification method I aim to improve on is the classification parameter stored in the SDSS catalogue as `frames` (explained further in Section 3.1.

I achieve this by selecting data properties relevant to the classification problem, then using those data with a range of machine learning algorithms to classify astronomical objects. During object classification, information behind the decision-making process that is usually internal to the machine learning algorithm will be gathered, output, and visualised to achieve a deeper understanding of how the machine learning algorithm succeeds in classifying individual objects.

The Chapter is laid out as follows. Section 3.1 describes the SDSS data and standard classification method behind assigning the `frames` parameter, Section 3.2 includes a comparison of algorithm performance, and methods to interpret the decision making processes in one of the tree-based algorithms. Section 3.3 details the results obtained from these methods in terms of purity and completeness.

[1]My code is hosted at https://github.com/xangma/ML_RF

3.1 Data

In this Section I introduce the observational data used in this Chaper, which is drawn from the Sloan Digital Sky Survey (hereafter SDSS Gunn et al., 1998). I briefly review the standard photometric star/galaxy classification criterion given by the `frames` method which is obtained through the query of the `objc_type` parameter (Stoughton et al., 2002) in the CasJobs SkyServer (Szalay et al., 2002).

3.1.1 Observational data

The data in this work is drawn from SDSS Data Release 12 (DR12, Alam et al., 2015). The SDSS uses a 2.5 meter telescope at Apache Point Observatory in New Mexico and has CCD wide field photometry in 5 bands (u, g, r, i, z York et al., 2000; Smith et al., 2002; Gunn et al., 2006; Doi et al., 2010), including an expansive spectroscopic follow-up program (Eisenstein et al., 2011; Dawson et al., 2013; Smee et al., 2013) covering 14,555 square degrees of the northern and equatorial sky. The SDSS collaboration has obtained more than 3 million spectra of astronomical objects using dual fiber-fed spectrographs. An automated photometric pipeline performs object classification to an r band magnitude of $r \approx 22$ and measures photometric properties of more than 100 million galaxies. The complete data sample, and many derived catalogs including galaxy photometric properties, are publicly available through the `CasJobs` server (Li and Thakar, 2008)[2].

As I will draw large random samples from the SDSS DR12 data, the full relevant dataset must first be obtained. I obtain object IDs, magnitudes and errors as measured in different apertures in each band, radial profiles, both photometric and spectroscopic type classifications, and photometry quality "flags" using the query submitted to CasJobs shown in Appendix A.1. Flags are useful indicators of the status of each object in the catalogue, warn of possible problems with the object images, or possible problems with the var-

[2]skyserver.sdss3.org/CasJobs

ious measurements related to the object[3]. The resulting catalog is similar to that used in Hoyle et al. (2015), but redshift information is omitted. A range of standard colors (e.g., `PSFMAG_U-PSFMAG_G`) and non-standard colors (e.g. `PSFMAG_U-CMODELMAG_G`) are generated for each object. The final catalog contains 215 input quantities, or 'features'. The magnitudes used in this Chapter are `PSFMAG`, `CMODELMAG`, `DERED`, and `FIBERMAG`. `DERED` magnitudes are `MODELMAG` magnitudes that are corrected for galaxy extinction. SDSS states `MODELMAG` magnitudes are calculated by using 'the model (exponential or de Vaucouleurs) of higher likelihood in the r filter, and applying that model (i.e., allowing only the amplitude to vary) in the other bands after convolving with the appropriate PSF in each band'. `FIBERMAG` represents the flux contained within the aperture of a spectroscopic fiber in each band. In Section I describe the magnitudes used in the `frames` method, `PSFMAG` and `CMODELMAG`. Objects that have a clean spectroscopic classification are filtered by selecting objects with a `Zwarning` flag in the catalogue that is equal to 0. This selection removes $\approx 11\%$ of the sample.

The final catalog contains 3,751,496 objects. It is noted that approximately 66% of these objects are spectroscopically classified as galaxies with the remaining objects classified as point sources. Two random samples from the final catalogue are selected: the first is a training sample of 10,000 objects and the second is a test sample comprised of 1.5 million objects. The small training sample allows a large exploration of model space to be completed in a tractable time scale.

3.1.2 Existing SDSS Classification Schemes: Spectral Fitting and Photometric Selection

The SDSS provides both a spectroscopic and a photometric classification for each object which both attempt to infer if the object is a galaxy or a point source, including both stars and QSOs. I briefly review both techniques

[3]see https://www.sdss.org/dr12/algorithms/photo_flags_recommend/

below.

The spectroscopic classification is stored in a catalogue parameter called `CLASS`, which is assigned by comparing spectral templates and the observed spectra using a χ^2 cost function (Bolton et al., 2012). During this process galaxy templates are restricted between redshifts, $0 < z < 2$ and QSO templates are restricted to $z < 7$. It is noted that the observed spectra are masked outside the wavelength range of 3600Åto 10400Å. In this work I assume that this analysis produces the true object classification due to the fact that this method directly determines the differences between single stellar spectra and compound galaxy spectra of many stars, and I will use it to compare different photometric classification predictions.

A second empirical method using photometric data is called `frames` (stored as the `objc_type` parameter in the CasJobs SkyServer), and uses the combination of following photometric magnitude measurements `PSFMAG-X - CMODELMAG-X`. The `PSFMAG` magnitude is calculated by fitting a point spread function model to the object which is then aperture corrected, as appropriate for isolated stars and point sources (see Stoughton et al., 2002). The `CMODELMAG` magnitude is a composite measurement generated by a linear combination of the best fit exponential and de Vaucouleurs light profile fits in each band. The resulting `CMODELMAG` magnitude has excellent agreement with Petrosian magnitudes for galaxies, and PSF magnitudes of stars (Abazajian et al., 2004). Therefore the condition `PSFMAG-X - CMODELMAG-X` is a reasonable discriminator between galaxies and point sources.

In detail the composite feature `PSFMAG-X - CMODELMAG-X` is divided into two bins for each of the X=5 SDSS bands, and the separating condition used to determine the object class is the same for each band and given by

$$\mathtt{PSFMAG} - \mathtt{CMODELMAG} > 0.145. \tag{3.1}$$

The SDSS pipeline provides the `frames` classification for each object in each photometric band, as well as an overall classification calculated by summing the fluxes in all bands and applying the same criterion as in Equation

3.1. This latter summation is used as the base line SDSS photometric classification scheme in this work. It is my understanding that this threshold of 0.145 was chosen through experimentation, as discussed in Section 4.4.6.1 of Stoughton et al. (2002).

The distribution in `PSFMAG` vs. `CMODELMAG` is shown for the training sample in Figure 3.1, with the condition given in Equation 3.1 as the dashed black line, and the colours denoting spectroscopic classification.

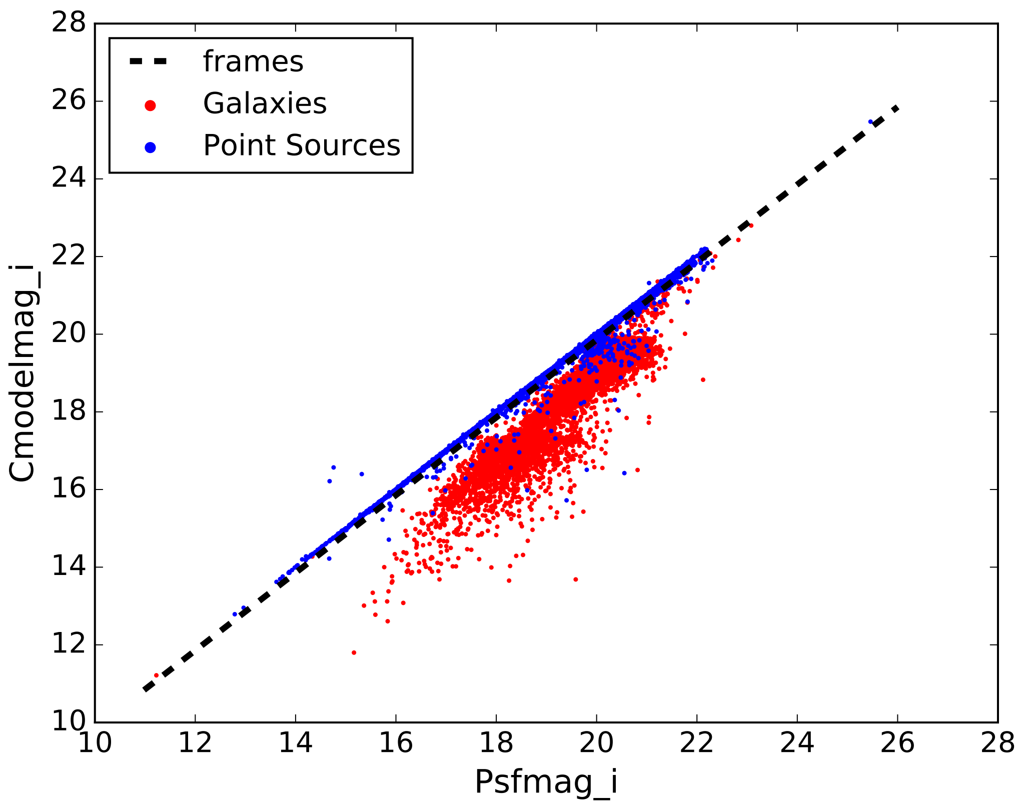


Figure 3.1: Object classification using the `frames` method. Here we show the relevant difference between two magnitude estimates in the I band, with the discriminating dashed black line drawn according to Equation 3.1.

In this work I investigate if a new photometric classification can improve the accuracy of the `frames` methods, and if by understanding how some machine learning systems work, I can motivate changes to these base-line photometric classification schemes. The authors of the `frames` method state that it accurately classifies objects at the 95% confidence level to r =21, and that the method becomes unreliable at fainter magnitudes (Stoughton et al., 2002).

3.1.3 Data Preparation

For the main body of this work, I only select data with good photometry and spectra. In particular I select objects in the catalogue where their `clean` flag is equal to 1. This removes objects which are duplicates, or with deblending issues, interpolation issues, or have suspicious detections, or are stars close to the edge of the survey.

How this may bias the results is explored, and a standalone test is performed in Section 3.3.1 with and without the `clean` flag selection to determine what effect this has on the classification accuracy.

3.2 Methods

In this work, I perform object classification using the four tree-based algorithms described in Section 2.1 (Random Forest, Extra Randomised Trees, Gradient Boosted Trees, and Adaboost. These methods are used with each of the following three subsets of photometric features:

- the five features that the SDSS pipeline uses in the `frames` method (i.e., `PSFMAG - CMODELMAG` for each filter);
- five features selected using a feature selection method, MINT as discussed in Section 2.1.3;
- all 215 features available in the sample.

Each test is performed with 10000 objects in the training sample, predicting on a test sample of 1.5 million objects. I will show the results for accuracy of classification in Section 3.3.2 for each algorithm operating on the different

	True Galaxies	True Point Sources
Objects classified as galaxies	T_g	F_{ps}
Objects classified as point sources	F_g	T_{ps}

Table 3.1: Variables used for defining purity and completeness.

subsets of photometric features. In this work, accuracy of classification is 100% when the classification provided from the tree-based algorithms or the `frames` method is equal to the classification provided by the `CLASS` parameter.

Each classification method is assessed using the standard metric of purity and completeness. I adopt the same definition as in Soumagnac et al. (2015) where purity refers to the fraction of retrieved instances that are relevant; completeness is the fraction of relevant instances that are retrieved. These measures are defined for galaxies in Equations 3.2 and 3.3 using the variables in Table 3.1, with the equations for point sources being similar. In relation to this work, purity would be a measure of how many galaxy classifications ($T_g + F_{ps}$) correctly identified galaxies (T_g), and completeness would be a measure of how many galaxies (T_g) were correctly identified out of the total amount of galaxies ($T_g + F_g$).

$$Purity = \frac{T_g}{T_g + F_{ps}} \tag{3.2}$$

$$Completeness = \frac{T_g}{T_g + F_g} \tag{3.3}$$

Figure 3.2 shows an example of the decision boundaries created from a Random Forest run using only two features, a simplified version of the first test in the list above. The area where the algorithm classifies objects as galaxies is shown in red, with classifications of stars shown in blue. The areas where classifications are more distinct have bolder colours, with the area around the horizontal boundary showing more uncertainty in object classi-

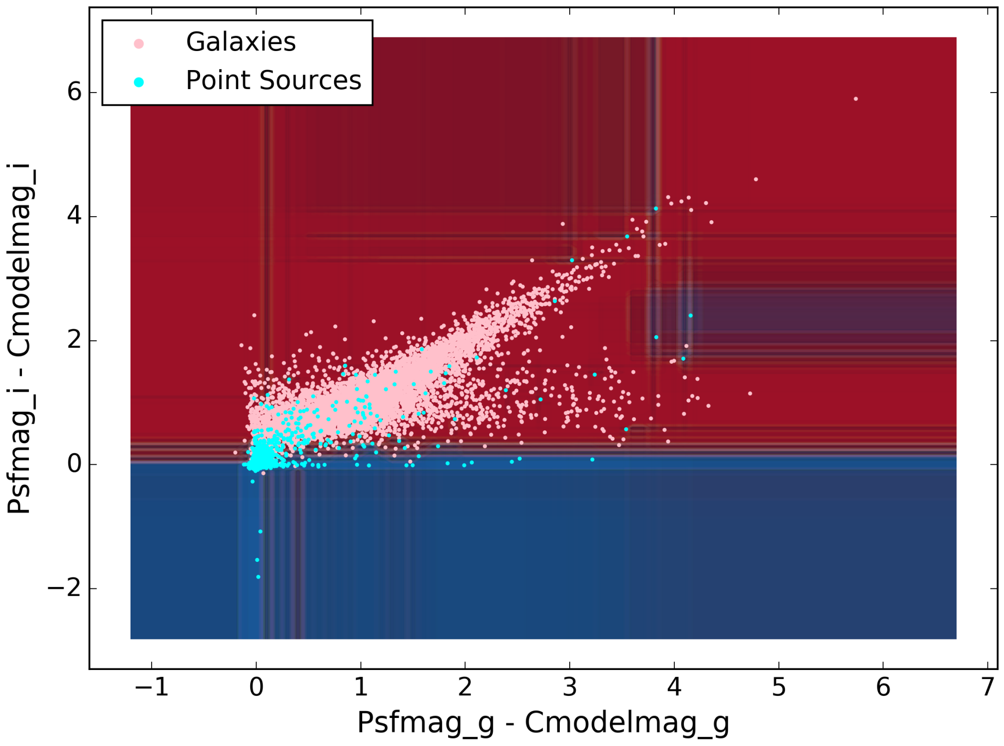


Figure 3.2: Training data (pink and cyan points for galaxies and point sources) plotted over the decision boundaries (red and blue background for galaxies and point sources), generated by an example Random Forest run using `frames` features in g and i band. The colour of the training data denotes spectroscopic classification.

fication. The plotted points show all 10000 objects of the training sample, colour-coded by their spectroscopic type. It should be noted that the Random Forest draws boundaries very similar to the ones in the SDSS pipeline paper, though not as linear. However, it can be seen that some objects are misclassified using both the `frames` method and this particular Random Forest run. Using more than two features, such as in the tests listed above, allows the machine learning methods to utilise more dimensions in parameter space and consequently achieve a higher accuracy of classification.

3.2.1 Results from feature selection using MINT

The SDSS pipeline measures and calculates a rich abundance of features from the photometric images. Rather than just focusing on those features employed in the `frames` algorithm, one may also choose other available features to pass to the machine learning algorithms. In this work, I reduce the number of features from 215 to 5 using MINT. This is to mirror the number of features the `frames` method uses and to test whether accurate predictions can be made with severely reduced data per object. Another major reason for utilising MINT over other feature selection methods such as mRMR is because it is able to utilise photometric data from the 1.5 million objects in the test sample. This provides more confidence that the selected features will be those which are correlated least with one another, thus giving us the best chance of accurate object classification.

Table 3.2 shows the results of the MINT feature selection method for 5 or 10 total selected features. It can be seen that there are features in common between these two sets; these have clearly been identified as robust and distinct features for classification.

The effect of changing the number of MINT selected features on the classification accuracy has been investigated in a test Random Forest run (with 256 trees and no set maximum depth). This can be seen in Figure 3.3. The accuracy of the results only increases slightly ($\approx 0.2\%$) as the number

Number of selected MINT features (using 10000 training objects and 1.5 million test objects)

5	10
PSFMAG_G - CMODELMAG_R	DERED_Z - FIBERMAG_R
PSFMAG_I - FIBERMAG_I	PSFMAG_I - CMODELMAG_I
DERED_G - FIBERMAG_G	PSFMAG_I - FIBERMAG_I
PSFMAG_I - CMODELMAG_I	DERED_G - FIBERMAG_G
PSFMAG_R - FIBERMAG_Z	PSFMAG_G - CMODELMAG_R
	PSFMAG_Z - FIBERMAG_Z
	PSFMAG_G - CMODELMAG_G
	PSFMAG_R - FIBERMAG_Z
	DERED_R - PSFMAG_R
	PSFMAG_R - FIBERMAG_R

Table 3.2: The features selected by MINT when setting the total number of features to five, or ten. `PSFMAG`, `DERED`, `FIBERMAG`, and `CMODELMAG` are all different estimates of magnitude in the 5 possible SDSS bands of u,g,r,i, and z.

of MINT selected features increases. Also shown is the effect of changing the number of objects in the training sample. Again, the accuracy does not change significantly (<1%).

3.2.2 Results from Interpreting Models of Tree Based Methods using *treeinterpreter*

The output data when using *treeinterpreter* is visualised in Figure 3.4, where I present results for a particular example feature `FIBERMAG_G - CMODELMAG_R`; this feature's results exemplify several notable behaviours. These figures were produced by creating a Random Forest model, and classifying each object in the test sample with it while outputting the contributions to the probability of classification for each feature. A density plot was then created with the object's colour (`FIBERMAG_G - CMODELMAG_R`) on the x-axis, and the contribution to the probability of being classified as a galaxy on the y-axis. The density shown by the colour bars in each figure represent: the number of actual spectroscopically confirmed galaxies in Figure 3.4a, the number of galaxies the model predicted correctly in Figure 3.4b, model purity for objects the model predicted as galaxies in Figure 3.4c, and model completeness for actual spectroscopically confirmed galaxies in Figure 3.4d. Purity refers to the fraction of retrieved instances that are relevant; completeness is the fraction of relevant instances that are retrieved.

Figure 3.4a shows the contribution to the probability of galaxy classification from `FIBERMAG_G - CMODELMAG_R`, for all of the galaxies in the test sample, given a Random Forest model trained on 10000 objects (using 256 trees and all 215 features in our catalogue). The colours show the number of objects with white showing the absence of data. Most of the galaxies fall into a small line of assigned probability of 0.002 at a `FIBERMAG_G - CMODELMAG_R` value of approximately 2.3, the mean of the sample, with the remaining galaxies scattered around the plot making up the blue colour. For this particular feature, `FIBERMAG_G - CMODELMAG_R`, some objects in the sample are given a reduced probability of being galaxies (i.e. they receive a negative contri-

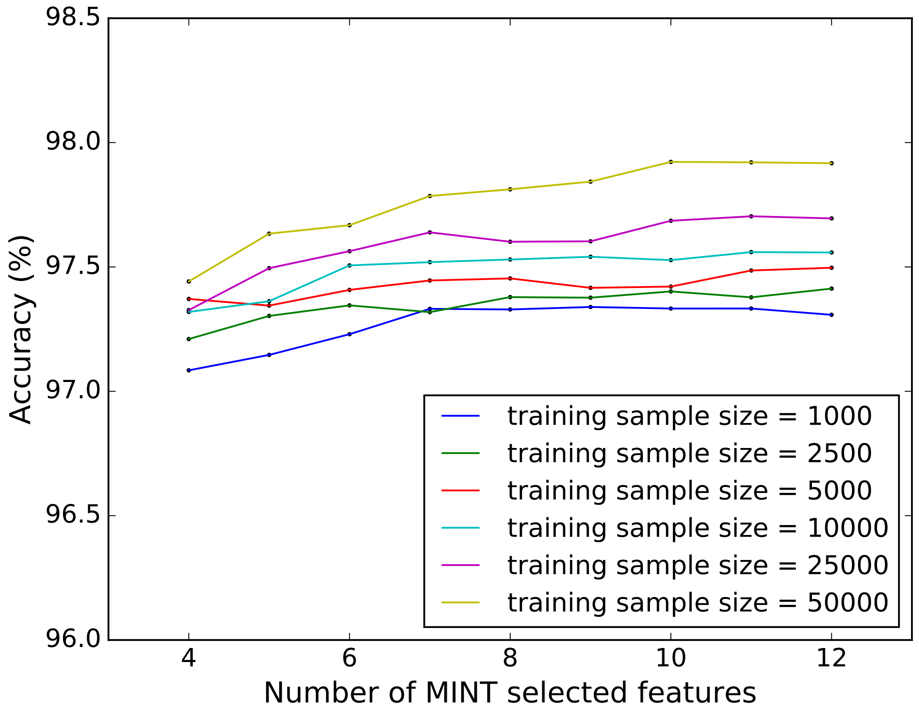
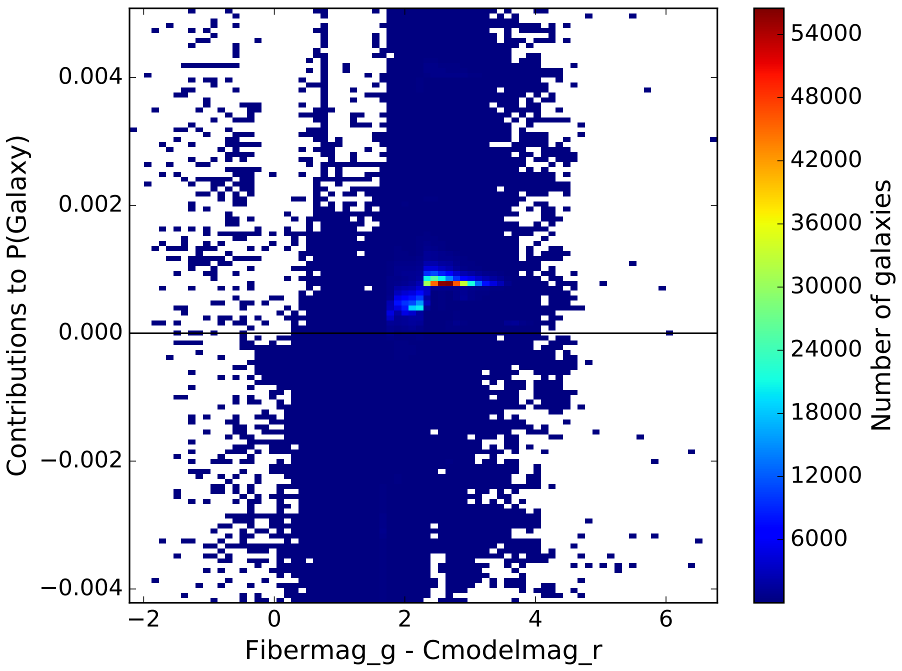
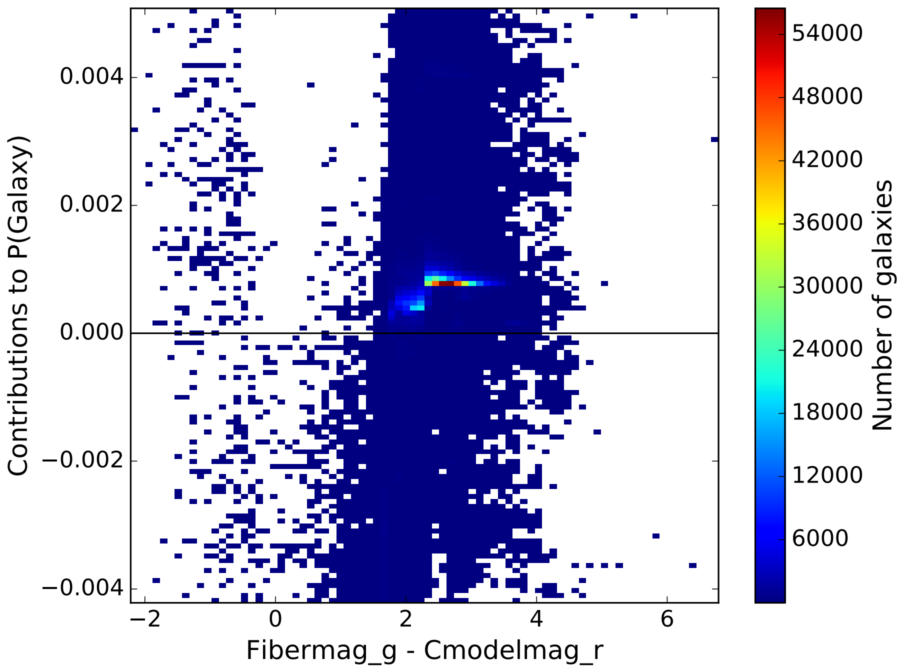


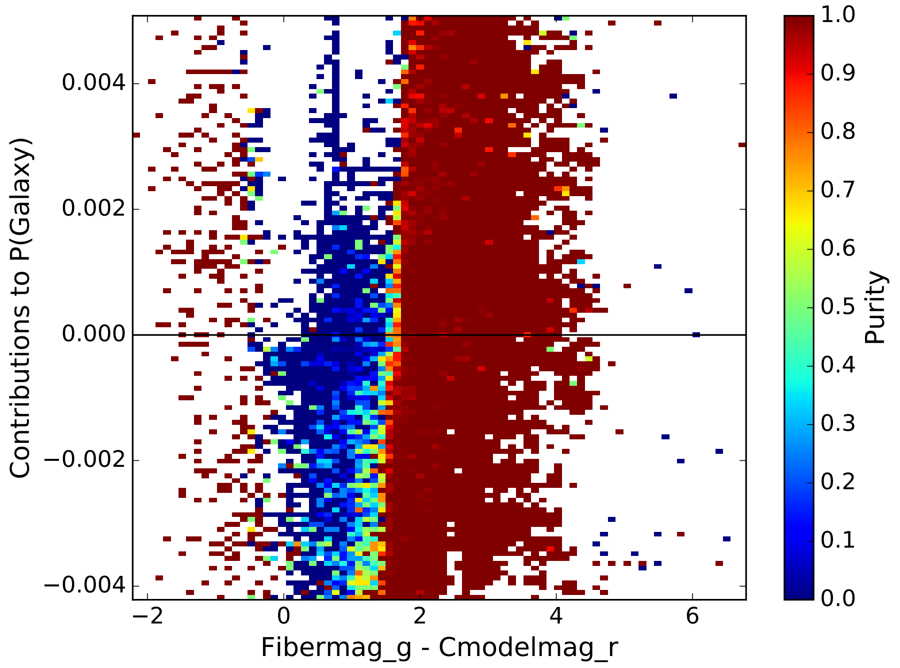
Figure 3.3: Effect of number of MINT selected features on predictive accuracy. Coloured lines denote the number of objects used in the training sample.



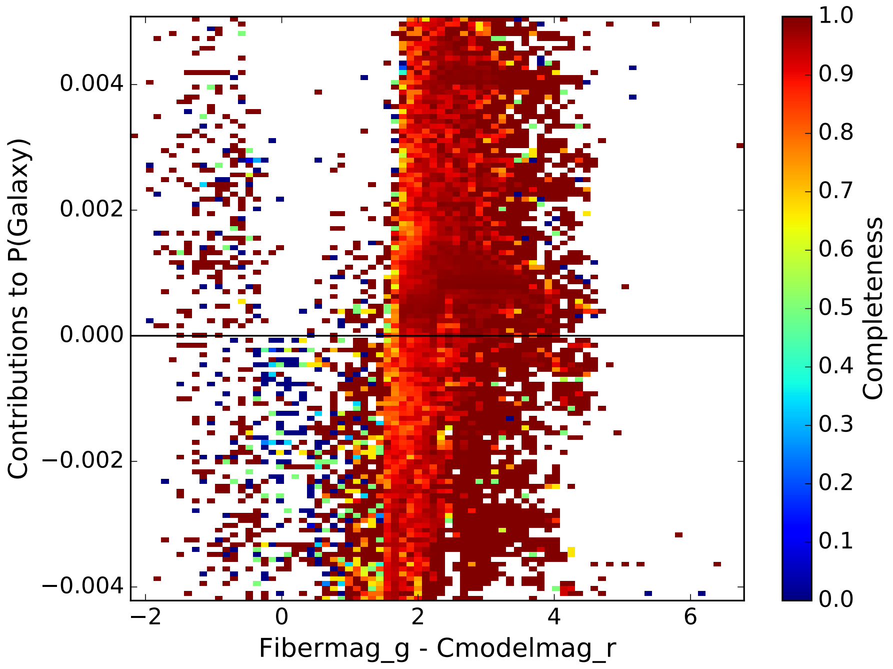
(a) The contribution to the probability of being predicted a galaxy by `FIBERMAG_G - CMODELMAG_R` of all spectroscopically confirmed galaxies in sample. Colour represents number of galaxies.



(b) The contribution to the probability of being predicted a galaxy by `FIBERMAG_G - CMODELMAG_R` for galaxies that have been correctly classified. Colour represents number of galaxies.



(c) The contribution to the probability of being predicted a galaxy by `FIBERMAG_G - CMODELMAG_R` for all objects classified as galaxies where the colour represents model purity.



(d) The contribution to the probability of being predicted a galaxy by `FIBERMAG_G - CMODELMAG_R` for all spectroscopically confirmed galaxies where the colour represents model completeness.

Figure 3.4: Density plot of contributions to the probability of a galaxy classification by `PSFMAG_G - CMODELMAG_I` for spectroscopically confirmed galaxies. Purity refers to the fraction of retrieved instances that are relevant; completeness is the fraction of relevant instances that are retrieved. In relation to this work, purity would be a measure of how many galaxy classifications correctly identified galaxies, and completeness would be a measure of how many galaxies were correctly identified out of the total amount of galaxies. This example was created with a Random Forest comprising of 256 trees with no maximum depth, using all 215 available features.

bution to probability); these are the data points below the black line. The model does not necessarily incorrectly classify these galaxies due to this one feature; there may be other features that are more important to the model than this one for classifying these particular galaxies.

Figure 3.4b shows the same as 3.4a, but for all the galaxies in the test sample that were correctly classified as galaxies. The colouring is the same as in Figure 3.4a. There are a number of galaxies with a `FIBERMAG_G - CMODELMAG_R` value of 0 to 2 that were incorrectly classified as point sources by the model, as they are missing when comparing to Figure 3.4a.

The colour of Figure 3.4c shows the purity of the galaxy classification, the fraction of retrieved instances that are relevant. Here it can be seen that the model has failed to correctly classify bluer galaxies, where `FIBERMAG_G - CMODELMAG_R` is closer to 0. This is because that region of parameter space is being used to classify point sources, see Figure 3.5.

The colour of Figure 3.4d shows the completeness of the galaxy classification; this can be interpreted as the probability that the object will be a galaxy given the model. Around values of `FIBERMAG_G - CMODELMAG_R` = 0, it can be seen that the model begins to fail at classifying galaxies correctly.

Visual analysis of this kind provides insight into how the model is drawing boundaries in parameter space, and information about where the limitations of the classifications arise.

3.2.3 Performance of Algorithms

Each machine learning method used in this work was tuned to optimise classification performance. This is achieved by varying the hyperparameters for each algorithm (such as number of trees and tree depth) and assessing the performance of the model using k-fold cross-validation (Mosteller and Tukey, 1968). The scikit-learn implementation of this method is called `GridsearchCV` [4]. In the K-fold cross-validation method, the best hyperpa-

[4]`http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html`

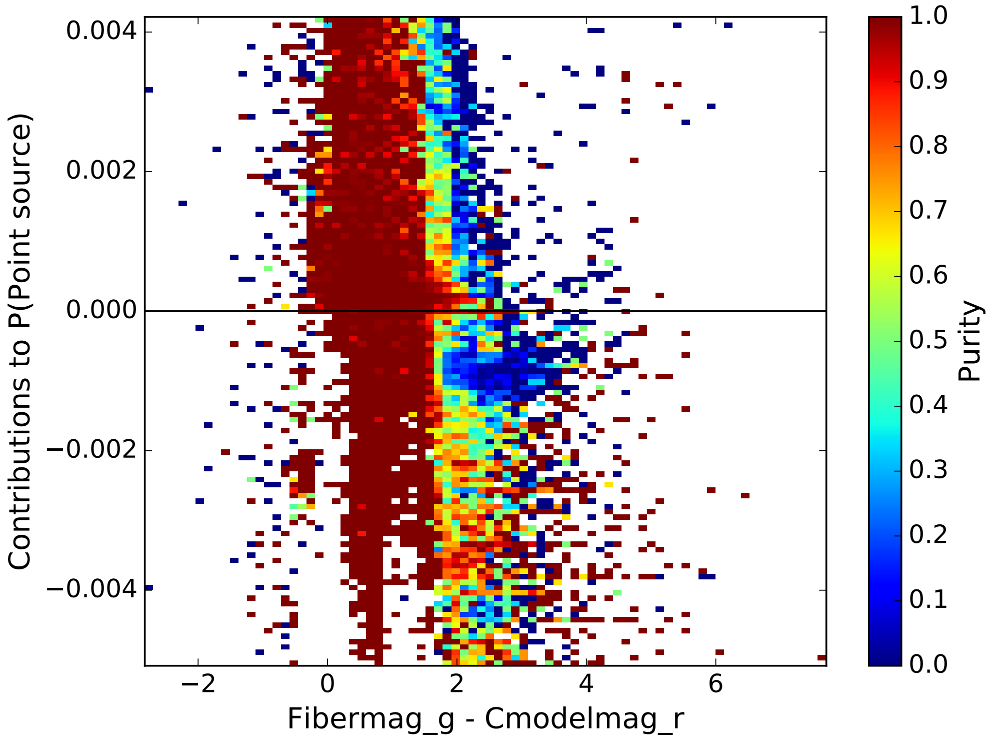


Figure 3.5: The contribution to the probability of being classified as a point source by `FIBERMAG_G - CMODELMAG_R` where the colour represents purity. The correctly classified point sources here are occupying the parameter space of the incorrectly classified galaxies in Figure 3.4c. 3.4c.

rameters are determined by splitting the training data up into a user defined number of groups (ten for example), training the model on nine of the groups, and testing the model on the last remaining group. The groups are then rotated until each group has been tested and the results of the tests are averaged. This process is performed for each set of hyperparameters, the results from the averaged tests are compared, and the set of hyperparameters with the best results is chosen.
The explored hyperparameters are: **n_estimators**: the number of trees, **max_features**: the number of features to consider when looking for the best split within a tree, **min_samples_leaf**: the minimum number of objects required to be at a leaf node, **criterion**: the function that measures the quality of the split, **min_samples_split**: the minimum number of samples required to make a split, **max_depth**: limits the maximum depth of the trees, and **learning_rate**: (used only in the boosted model building methods of ADA and GBT) shrinks the contribution of each classifier by the set value.

The most efficient hyperparameters are listed in Tables 3.4, 3.5, and 3.6 for the `frames` features test, the MINT selected features test, and the all features test respectively. The full grids can be seen in Table 3.3.

In most cases, 64 trees is an adequate number of estimators for all of the tested machine learning algorithms. However, it can be seen that the preferred trees are shallower when using five MINT selected features, yet the mean validation scores match or exceed that of the tests when using the `frames` set of features. This shows that MINT selected features do not degrade the predictive power, while reducing the number of computations.

3.2.4 Using Random Forests as a motivation for improving `frames`

Machine learning algorithms can also be used to optimise or check preexisting decision boundaries such as the ones provided by the `frames` method in Equation 3.1. It is possible that a line very similar to the black dashed line in Figure 3.1 would be more accurate in classifying these objects. To check

Hyperparameter Grid

n_estimators	64, 128, 256, 512
max_features	1, 3, None
min_samples_leaf	1, 3, 10
criterion	gini, entropy
min_samples_split	2, 3, 10
max_depth	3, 6, 9, None
learning_rate	0.001, 0.01, 0.1, 0.5, 1.0

Table 3.3: Hyperparameters for each machine learning algorithm (where applicable) which we explored during the gridsearch cross-validation. **n_estimators** is the number of trees, **max_features** is the number of features to consider when looking for the best split within a tree, **min_samples_leaf** is the minimum number of objects required to be at a leaf node, **criterion** is the function that measures the quality of the split, **min_samples_split** is the minimum number of samples required to make a split, **max_depth** limits the maximum depth of the trees, and **learning_rate** (used only in the boosted model building methods of ADA and GBT) shrinks the contribution of each classifier by the set value.

Hyperparameter Optimization Results (using `frames` features)

	RF	ADA	EXT	GBT
n_estimators	64	512	64	64
max_features	3	1	1	1
min_samples_leaf	3	1	1	3
criterion	gini	entropy	entropy	-
min_samples_split	3	2	2	3
max_depth	None	6	None	9
learning_rate	-	1.0	-	0.1
Mean Validation Score	0.974	0.975	0.974	0.974
Standard Deviation	0.004	0.003	0.004	0.002

Table 3.4: The most efficient variables for each machine learning method when only using the `frames` set of features. The **Mean Validation Score** is the accuracy which the best parameters achieved. Rows are as in Table 3.3.

if this is the case, I generated a Random Forest model on the training set, using only PSFMAG_I and CMODELMAG_I as input features (the same features as in the `frames` method for I-band). After performing a hyperparameter search (excluding `max_features` as I only have 2 features), I then generated a fine grid of x and y coordinates spanning our training set magnitude limits and used the model to classify each of those points, which then outputs the decision boundary. I fit a straight line to the main trend of the decision boundary, and use this line instead of the one provided by the `frames` method of classification to classify objects, and determine if the Random Forest model can improve on it. I present the results of this test in Section 3.3.3.

3.3 Results of object classification using machine learning

Presented in this section are the results from the tests described in previous sections. In particular I show results for the investigation into whether the `clean` flag generates artificial bias in the sample and model (Section 3.1.3). I then compare the `frames` classification method with machine learning methods as introduced in Section 2.1.1. I examine the use of Random Forests to improve the `frames` classification as discussed in Section 3.2.4, and finally present an example of multiclass classification where I classify objects as galaxies, stars, or QSOs.

3.3.1 Clean flag test

As described in Section 3.1.3, I perform a Random Forest test without the pre-selection of objects labeled as clean in the CasJobs database, to assess how this affects accuracy. Using the `frames` features defined in Section 3.1.2, with optimised Random Forest settings (after performing a new hyperparameter search because applying this flag changes the objects in the sample), the results from this test reach a total accuracy of 97.2%. This is 0.2% below the achievable rate when applying the `clean` flag.

As this work is essentially a proof of concept and not a comparison of machine learning models, I have chosen to utilise the `clean` flag in our tests to ensure the machine learning algorithm can build a model from reliable objects. This reduces noise in the model that could have influence on the placement of decision boundaries, which would cloud interpretability.

3.3.2 Comparing `frames` and Machine Learning Methods

In this section, the main comparison is made between object classification using the SDSS `frames` criteria and the machine learning methods described in Section 2.1.1.

Object classification is assessed using the `frames` criteria (Equation 3.1). Table 3.7 shows the results from the `frames` method of object classification in all filters separately, as well as combined. It is seen here by using all filters in combination that 97.1% of object classifications match the classification given by spectroscopy. This result shows that the `frames` method performs object classification above the 95% confidence level while remaining simple and monotonic. The next tests will use machine learning methods to attempt to improve on this.

Table 3.8 shows the results of the different machine learning algorithms using the same set of features as the `frames` classification method. In all cases, the accuracy is slightly higher than that achieved by the `frames` method, with the average accuracy increase being 0.3%, and the highest accuracy being 97.4%.

Table 3.9 shows the results from the machine learning runs with 5 MINT selected features (see Table 3.2 and Section 3.2.1). The highest accuracy seen in this set of runs is also 97.4%, showing that the MINT selected features are only as useful for classification accuracy as those selected for `frames` (except in the case of the Adaboost algorithm which shows a slight improvement of 0.1%). It is of interest that there is only one feature in common between `frames` and MINT, and yet they succeed equally well under machine learning.

While using a low number of features (specially selected or not) in combination with machine learning methods yields good results, accuracy can be further improved by using as much data as possible. Table 3.10 shows the results when using all available features in our catalogue, for each machine learning algorithm. It is seen here that the ExtraTrees and Gradient Boosted Trees method achieves the highest accuracies, correctly classifying 98.1% of the objects in the test sample. This improves on the `frames` object classification accuracy by 1.0%, which is $\approx$ 33% improvement in the rate of misclassification.

3.3.3 Using Random Forests as a motivation for improving `frames`

In Section 3.2.4, I discussed how Random Forests could be used to check or optimise a method like `frames`. Figure 3.6 shows that by fitting a line to the main trend of the decision boundary used by the Random Forest model, we obtain a slightly shallower line than the one given by the `frames` method, with the equation being $y = 0.993x + -0.218$. Using this new line to classify the test data, I improve the accuracy of object classification in the I band by $\approx$0.8%, and discover that objects are more likely to be point sources when `CMODELMAG_I` is lower than `PSFMAG_I` at fainter magnitudes (though this effect decreases as brightness of the object increases).

3.3.4 Multiclass Classification

The SDSS pipeline outputs both a classification type and subtype from the template fitting of spectra (e.g. type = point source, sub type = star or QSO). Therefore, it is possible to test machine learning algorithms with the more complex task of deciding between more classifications than just galaxy or point source.

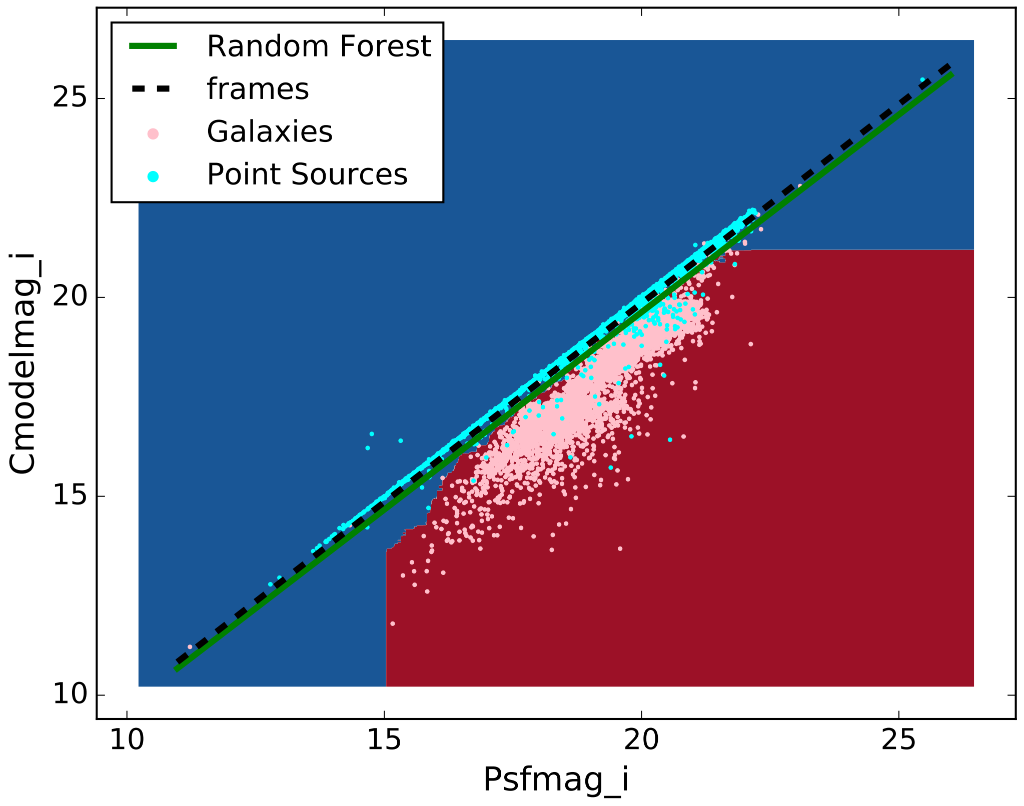


Figure 3.6: The decision boundaries generated by a Random Forest run using `PSFMAG_I` and `CMODELMAG_I` as features. The training data (pink and cyan points for spectroscopically confirmed galaxies and point sources) has been plotted over the decision boundaries (red and blue background for galaxies and point sources). The original frames method of classification is shown by the black dashed line, and the Random Forest motivated method of classification is shown by the green line.

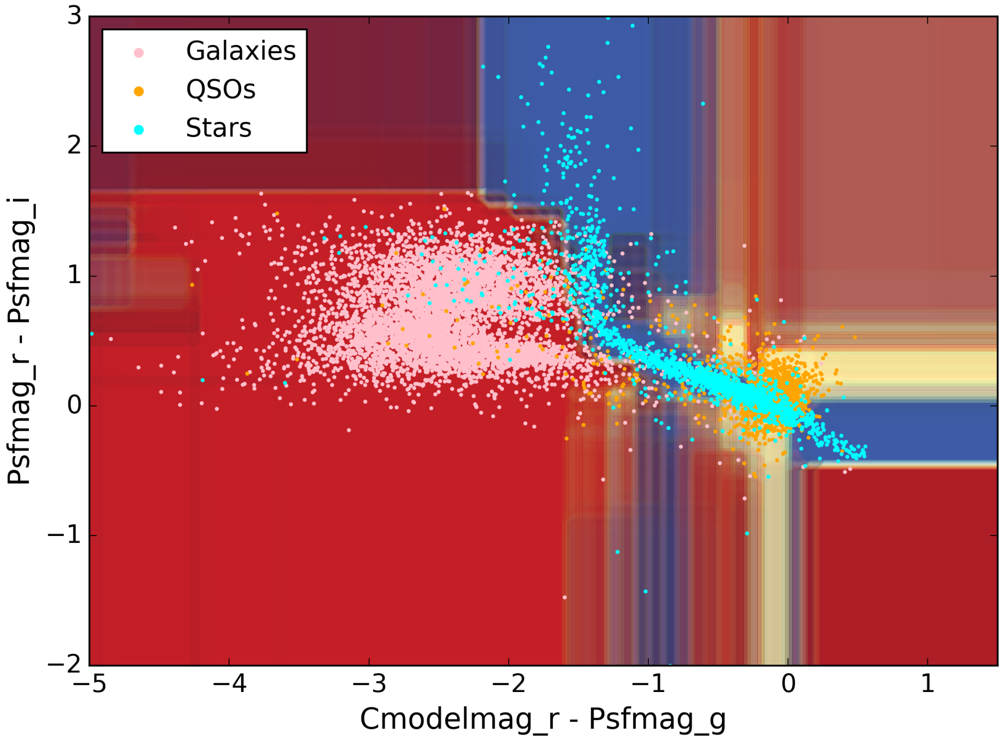


Figure 3.7: The decision boundaries generated by an example Random Forest run on a multiclass problem using two photometric colours as features. The training data (pink, cyan, and orange points for spectroscopically confirmed galaxies, point sources, and QSOs) has been plotted over the decision boundaries (red, blue, and orange background for galaxies, point sources, and QSOs).

Figure 3.7 shows the decision boundaries from a Random Forest run using two photometric colours where the algorithm was asked to decide if an object was a star, galaxy, or QSO - a multiclass problem. The two colours were chosen as features for this example because they better disperse the data than using two `frames` features. The training process is the same as for a binary classification problem except that here the decision trees in the forest will have a fraction of leaves which identify QSOs. After a fresh hyperparameter search, the Random Forest achieves an object classification accuracy of 89.6%. This accuracy is lower than in previous tests due to the model's inability to accurately distinguish between stars and QSOs; this may be due to their inherent similarities as point sources. Nevertheless, this example points towards the potential of ML methods for more extensive multiclassification problems.

The work in this chapter has showcased tree-based machine learning methods by revisiting the long standing object classification method used in the SDSS pipeline, `frames`, with the aim of increasing object classification accuracy using photometric data. I have developed a pipeline that offers in-depth analysis of machine learning models using *treeinterpreter*, which has the ability to select the most important and relevant features specific to the input data using MINT. In practice, the pipeline improves on the `frames` object classification accuracy by 1.0%, which is $\approx 33\%$ improvement in the rate of misclassification (object classification error improved from $\approx 3\%$ to $\approx 2\%$).

Hyperparameter Optimization Results (using 5 MINT features)

	RF	ADA	EXT	GBT
n_estimators	64	512	64	256
max_features	1	1	3	1
min_samples_leaf	1	3	3	10
criterion	entropy	entropy	gini	-
min_samples_split	10	3	3	10
max_depth	3	4	None	9
learning_rate	-	0.01	-	0.01
Mean Validation Score	0.974	0.974	0.973	0.974
Standard Deviation	0.006	0.006	0.005	0.006

Table 3.5: The most efficient variables for each machine learning method when using 5 MINT selected features. The **Mean Validation Score** is the accuracy which the best parameters achieved. Rows are as in Table 3.3.

Hyperparameter Optimization Results (using all features)

	RF	ADA	EXT	GBT
n_estimators	256	512	512	512
max_features	None	None	None	None
min_samples_leaf	1	1	1	10
criterion	entropy	entropy	entropy	-
min_samples_split	2	10	3	2
max_depth	None	3	None	6
learning_rate	-	0.1	-	0.1
Mean Validation Score	0.979	0.980	0.981	0.981
Standard Deviation	0.003	0.004	0.004	0.004

Table 3.6: The most efficient variables for each machine learning method when using all available features in the sample. The **Mean Validation Score** is the accuracy which the best parameters achieved. Rows are as in Table 3.3.

`frames` method results (objc_type vs template type using 1.5 million objects from the test sample.)

	Completeness		Purity		F1 Score		Accuracy
	Galaxies	Point Sources	Galaxies	Point Sources	Galaxies	Point Sources	
u	0.814	0.773	0.854	0.719	0.834	0.745	0.799
g	0.957	0.937	0.961	0.930	0.959	0.933	0.949
r	0.990	0.932	0.959	0.983	0.974	0.957	0.968
i	0.991	0.911	0.948	0.985	0.969	0.947	0.961
z	0.985	0.813	0.896	0.971	0.938	0.885	0.920
ALL	0.986	0.943	0.966	0.980	0.977	0.961	0.971

Table 3.7: Results of classification for both galaxies and point sources using the `frames` method (Equation 3.1) in separate photometric filters, and using all filters. F1 score is the harmonic mean of the purity and completeness, and accuracy is the fraction of objects predicted correctly when comparing with the classification from fitted spectra. It is seen here that the **`r`** band filter gives the highest accuracy of classification, but when using a summation of the fluxes from all of the photometric bands available, accuracy is increased.

Machine Learning algorithm results (`frames` features)

	Completeness		Purity		F1 Score		Accuracy
	Galaxies	Point Sources	Galaxies	Point Sources	Galaxies	Point Sources	
Random Forest	0.986	0.955	0.973	0.976	0.979	0.966	0.974
Adaboost	0.985	0.954	0.972	0.975	0.979	0.965	0.973
ExtraTrees	0.986	0.953	0.972	0.977	0.979	0.965	0.974
Gradient Boosted Trees	0.985	0.955	0.973	0.976	0.979	0.965	0.974

Table 3.8: Results of classification with four machine learning methods using the same features as in the `frames` method. Columns are as in Table 3.7.

Machine Learning algorithm results (5 MINT selected features)

	Completeness		Purity		F1 Score		Accuracy
	Galaxies	Point Sources	Galaxies	Point Sources	Galaxies	Point Sources	
Random Forest	0.986	0.954	0.972	0.977	0.979	0.965	0.974
Adaboost	0.986	0.953	0.971	0.977	0.979	0.965	0.974
ExtraTrees	0.986	0.956	0.973	0.976	0.979	0.966	0.974
Gradient Boosted Trees	0.986	0.954	0.972	0.977	0.979	0.965	0.974

Table 3.9: Results of classification with four machine learning methods using 5 MINT selected features listed in Table 3.2. Columns are as in Table 3.7.

Machine Learning algorithm results (all features)

	Completeness		Purity		F1 Score		Accuracy
	Galaxies	Point Sources	Galaxies	Point Sources	Galaxies	Point Sources	
Random Forest	0.990	0.964	0.978	0.983	0.984	0.973	0.980
Adaboost	0.989	0.964	0.978	0.982	0.984	0.973	0.980
ExtraTrees	0.990	0.966	0.979	0.983	0.985	0.975	0.981
Gradient Boosted Trees	0.989	0.968	0.980	0.982	0.985	0.975	0.981

Table 3.10: Results of classification for four machine learning methods using all available features in the catalogue. Columns are as in Table 3.7.

Chapter 4

Testing Models of Gravity Using Machine Learning - Classification

As shown in Chapter 3, machine learning algorithms are able to create models that accurately classify astronomical objects as either stars or galaxies using photometric data. I have also been able to interrogate the model, parameterising where the decision boundaries are drawn, and where the model's classifications succeed or fail. With the formation of stars and galaxies being a consequence of the laws of gravity, it would be interesting to pose the question - 'Can current machine learning algorithms help in determining which theory of gravity our Universe follows?', and if they can 'how are these machine learning algorithms determining this?'. To answer these questions I use dark matter particle n-body simulations created using two types of gravity, ΛCDM and nDGP, which result in differing particle distributions. This work is the first known application of machine learning algorithms to classify different models of gravity using n-body simulations in cosmology.

In this chapter I describe the code used to generate the ΛCDM and nDGP simulations and the parameters used to generate them. I perform some basic statistical analysis on the simulations, including measuring the power spectrum, and form the simulations into a labelled training and test set ready for my CNN to learn from (Section 4.1). The goal is to create a CNN model that can classify each simulation as either ΛCDM or nDGP gravity models. I describe the architecture of the CNN in Section 4.2, and perform tests monitoring classification accuracy of a CNN model using a constant set of ΛCDM simulations and nDGP simulations of varying $r_c H_0$ values (Section 4.3). In Section 4.5 I describe how this work can be extended, and introduce new types of machine learning models that could apply to this type of investigation, including some preliminary tests. In Chapters 4 and 5 of this work, accuracy of classification is 100% when the classification provided from the CNN model is equal to the model of gravity used to generate the simulation.

All of the simulations used in the calculations in Chapters 4 and 5 were generated using the local supercomputer to the Institute of Cosmology and Gravitation, Sciama 2. This was parallelised using a script written in UNIX, running on up to 1000 cores at a time. All of the CNN calculations were done either using a Nvidia K80 graphics processing unit (GPU) provided by Google Cloud Services, or my own Nvidia GTX 1080 and 1050 GPUs.

4.1 Particle Simulations

Whilst not specifically a dark matter particle simulation, the first experiment that could be considered a type of particle simulation was performed by Holmberg (1941), where he constructed an array of lights to mimic gravitational force as both light and gravity intensity follow a r^{-2} dependency. The aim of his experiment was to demonstrate the tidal interactions between two galaxies.

From these initial demonstrations, the field developed through the 1960's with the idea that galaxy clusters form through gravitational instabilities within an expanding universe (van Albada, 1961). The first numerical calculations of galaxy cluster stability then followed using only 10-100 particles (Aarseth, 1963; von Hoerner, 1963), and then hundreds of particles (Hénon, 1964; Peebles, 1970). In the 1980's, physical processes included in cosmological simulations became much more detailed, with models where the mass of the universe was dominated by collisionless particles. The growth of density fluctuations in the expanding universe and the temperature of the particles had an impact on the linear power spectrum of the density fluctuations (eventually leading to the CDM model (Peebles, 1983)).

When larger simulations were able to be produced the study of dark matter halos commenced, with Frenk et al. (1985) and Quinn et al. (1986) analysing the structure of halos containing a few thousand particles per object. It was through the 1990's that simulations containing 10^5 particles were achievable and as such, the beginnings of the research into the inner structure and accretion history of dark matter halos (Tormen et al., 1997). Due to hardware advances, n-body simulation codes became more complex in the 2000's, with the creating of a popular code called GADGET-2 (GAlaxies with Dark matter and Gas intEracT) (Springel, 2005). GADGET-2 could be run on hardware in parallel, and could cope with hundreds of millions of particles, with the size of the simulation limited to the amount of available physical memory. The code also employed various techniques for calculating long and short range gravitational forces felt between the particles.

A new code has been released in the last few years called L-PICOLA (Howlett et al., 2015), which vastly reduces the computational time required to create accurate n-body simulations.

4.1.1 L-PICOLA and MG-PICOLA

To create dark matter particle simulations I utilise a modified version of the L-PICOLA (Howlett et al., 2015) code called MG-PICOLA (Winther et al., 2017), which includes the option of nDGP gravity (as well as $F(R)$ gravity discussed in Section 1.4). L-PICOLA is used to generate and evolve a set of initial conditions (based on second-order Lagrangian perturbation theory; 2LPT) into a dark matter field at $z = 0$ in speeds that are 3 orders of magnitude faster than full non-linear N-Body simulations. In order to create these simulations, one must consider how cold dark matter particles evolve over cosmic time - they follow the equation of motion as described in Scoccimarro (1998)

$$\frac{d^2\Psi}{d\tau^2} + \mathcal{H}(\tau)\frac{d\Phi}{d\tau} + \nabla\Phi = 0, \tag{4.1}$$

where, Φ is the gravitational potential, $\mathcal{H}(\tau) = \frac{dlna}{d\tau}$ is the conformal Hubble parameter with a being the scale factor. Ψ is the displacement vector of the particle and relates the particle's Eulerian position to its initial Lagrangian position. To provide an accurate solution to Equation 4.1, L-PICOLA uses the COLA method (Tassev et al., 2013). The COLA method works by using the first and second-order lagrangian displacements, which provide exact solutions at large, quasi-linear scales, and solving for the non-linear component. Howlett et al. (2015) states that the reason this method is so useful is because the Lagrangian displacements only have to be calculated once, at $z = 0$, and scaled by the appropriate derivatives of the growth factors.

L-PICOLA uses the 'Kick-Drift-Kick' (or 'leapfrog') method (Quinn et al., 1997) to update the velocities and positions of each particle in each timestep, using a combination of the gravitational potential Φ and the stored 2LPT displacements. In this method, after an initial 'kick', particle velocities are calculated from the displacements and updated to the nearest half-integer timestep. Once the particles have drifted for a half-integer timestep, the particle positions are calculated from the previous velocities. This results in (excluding the beginning and end of the evolution) the particle positions and velocities never being calculated at the same point in time, with each calculation leapfrogging the other. The reason this method is used is because it is time-reversible, guaranteeing that errors in the total energy of the system do not increase with time, and that the total energy of the dynamical system is conserved.

For my investigation, the velocity information for each dark matter particle simulation is not used for model discrimination, however, in Section 5.3 I discuss how this could be used to improve this result.

The gradient of Φ is calculated using the Particle-Mesh method (an overview is in Hockney and Eastwood (1988)). In this method, a mesh is placed over the dark matter particles, and the gravitational forces are calculated at each mesh point. Then the gravitational potential for each particle is calculated by interpolation using the coordinates of the mesh points and particles. This gravitational potential is used to update the velocity and resultant displacement in the timestep. This method is efficient because for N_m^3 mesh points in the simulation and N^3 particles, the maximum number of force calculations required in each timestep is N_m^3 where $N = N_m$, which is much quicker than calculating the contribution to the gravitational force at each mesh point from each particle.

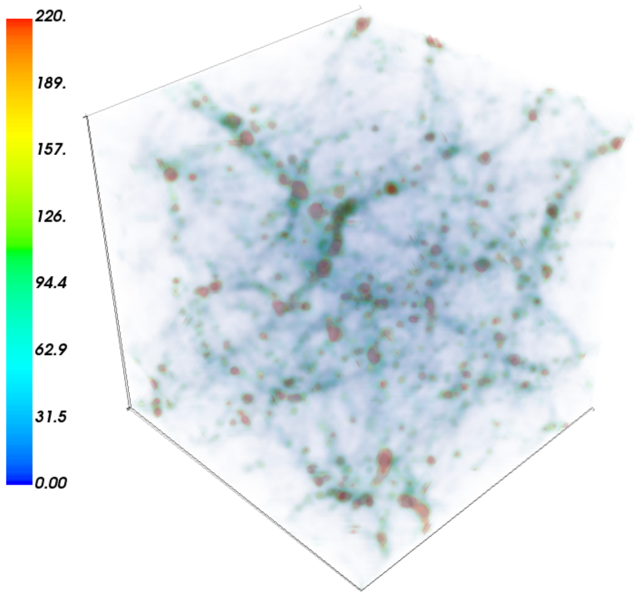
4.1.2 Simulation Parameters and Dataset Specifics

The settings used to create the simulations used in Chapters 4 and 5 are detailed in this section. I created each simulation using MG-PICOLA (v0.9) on a single core of the local supercomputer. Each simulation is 128 Mpc^3 in volume binned to 64^3 (meaning each voxel is 2 Mpc^3) and contains 128^3 dark matter particles of a constant mass of 2*10^{10}M$_\odot$. The nDGP simulation sets use r_cH_0 values of 0.5, 0.75, 1.00, 1.50, 2.50, 5.00. In all cases, simulations are converted from particle number counts (n) in each voxel to overdensities ($\delta = \frac{n-\bar{n}}{\bar{n}}$). The python code I wrote creates simulation sets comprising of 500 simulations for each model of gravity, i.e. 500 ΛCDM simulations, 500 nDGP with an r_cH_0 value of 0.5, 500 nDGP with an r_cH_0 value of 0.75 etc. For each test, the ΛCDM simulation set and one nDGP simulation set of

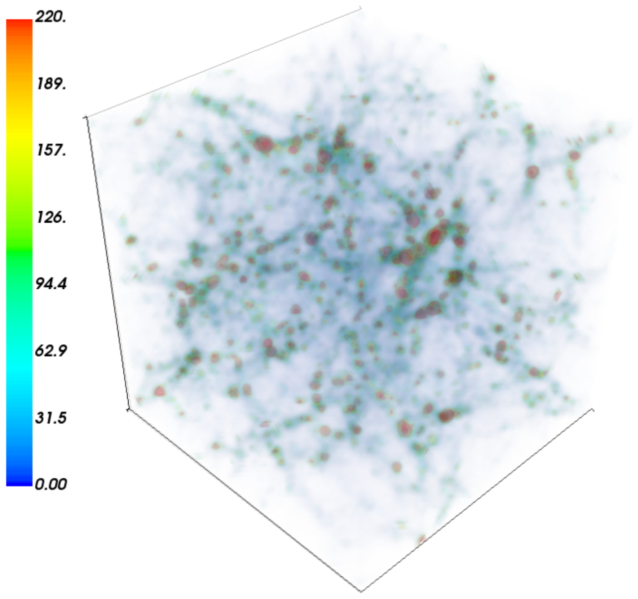
a particular $r_c H_0$ value is chosen. These sets are combined and randomly divided in equal portions (50/50) into a training and test set. This shuffle and split is performed before any data augmentation techniques so there is no crossover of rotated simulations between the training and test set. Each of these simulations is a cube with 6 faces, which can each be rotated 4 times, extending the total dataset size to an effective 12000 training and 12000 test samples. If this data augmentation technique was not used, creating 24000 unique simulations would take 24 times longer than creating 1000 unique simulations and rotating them. Instead of saving all of the rotated simulations in a large file, to save disk space I wrote a python generator (a type of function in the python programming language) that performs this rotational data augmentation as the CNN is training.
When training a CNN (using the set of ΛCDM simulations and a set of nDGP simulations with a particular $r_c H_0$ value), 10000 simulations are randomly drawn from the training sample, and when testing, 5000 simulations are drawn from the testing sample. This process represents one 'epoch' of training, and in this investigation, training occurs for 15 epochs per test.

Figure 4.1 shows both a ΛCDM and nDGP simulation ($r_c H_0 = 0.5$) randomly drawn from the training set.

Figure 4.2 shows 25 averaged histograms of the overdensities of both ΛCDM and nDGP simulations. The x-axis is displayed logarithmically and the errors are the red lines and are the standard deviation of the 25 histograms of simulations. It is seen here that ΛCDM gravity produces on average a more sparse distribution of particles throughout the simulations than is seen in the nDGP gravity simulations. Figure 4.3 shows the same as Figure 4.2, except the y-axis is also displayed logarithmically. This shows that when comparing the densest regions of the simulations, nDGP gravity produces more intense dense regions on average.



(a) ΛCDM input simulation



(b) nDGP input simulation

Figure 4.1: Example of input simulations generated by MG-PICOLA. The colour bar represents number of particles. The plotted data includes the bottom 5% (approximately) of the data in the ranges of the simulations so that structure can be seen.

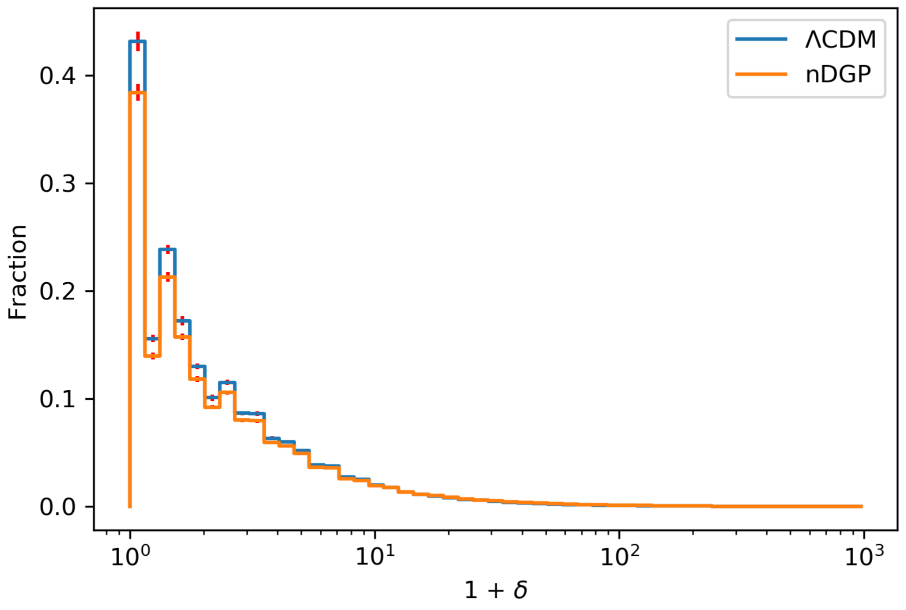


Figure 4.2: Histogram of overdensities of input simulations generated by MG-PICOLA. Blue and orange lines are the averaged histograms of 25 simulations randomly drawn from the dataset where $r_c H_0 = 0.5$ for the nDGP simulations. Errors are the red lines and are the standard deviation of the 25 histograms of simulations.

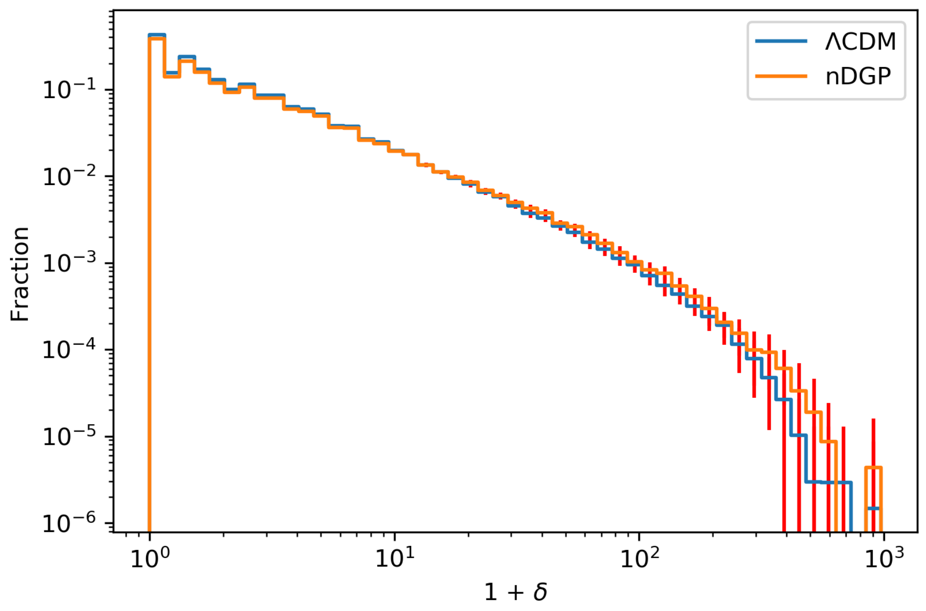


Figure 4.3: Histogram of overdensities of input simulations generated by MG-PICOLA. Blue and orange lines are the averaged histograms of 25 simulations randomly drawn from the dataset where $r_c H_0 = 0.5$ for the nDGP simulations. Errors are the red lines and are the standard deviation of the 25 histograms of simulations.

4.1.3 Power Spectrum

To quantify one of the main differences in the models of gravity, I measure their power spectrum using the Fourier transform of the two-point correlation function of the particle densities. The power is measured in each spherical shell in Fourier space, or each k bin or k mode. This is done using Equation 4.2 (Howlett, 2016),

$$P(\underline{k}) = \sum_{\underline{k}} \left[(\widetilde{N}_c(\underline{k})^2 - N_p) g^2 \right] \frac{V^2}{N_p^2 V}, \tag{4.2}$$

in which $N_c = N - \bar{N}$ and $g = \frac{sin(k*ksize)}{k*ksize}$, N_p is the total number of particles in the simulation, V is the simulation volume, N_c is the fourier transformed overdensities, g is a corrective factor required due to the fact that we've binned the simulation to a grid, and $P(\underline{k})$ is the power in that particular k bin. The code to perform this measurement was kindly provided by Cullan Howlett.

Figure 4.4 shows the measured and averaged power spectrum of 25 ΛCDM and nDGP simulations randomly drawn from the training set with the nDGP simulations having differing r_cH_0 values. The error bars are the standard deviation of the randomly drawn 25 simulations for each model of gravity. It is seen here that using an r_cH_0 value of 0.50 (the blue line) causes a $\sim$10% enchancement in power across all k modes. This should enable the CNN to discriminate between the models of gravity used to produce the simulations quite well. As this r_cH_0 value increases, the power enhancement over ΛCDM diminishes and the CNN should find it increasingly more difficult to correctly classify the simulations as being created using a particular model of gravity. This effect is shown more clearly in Figure 4.5, where only a portion of Figure 4.4 is shown.

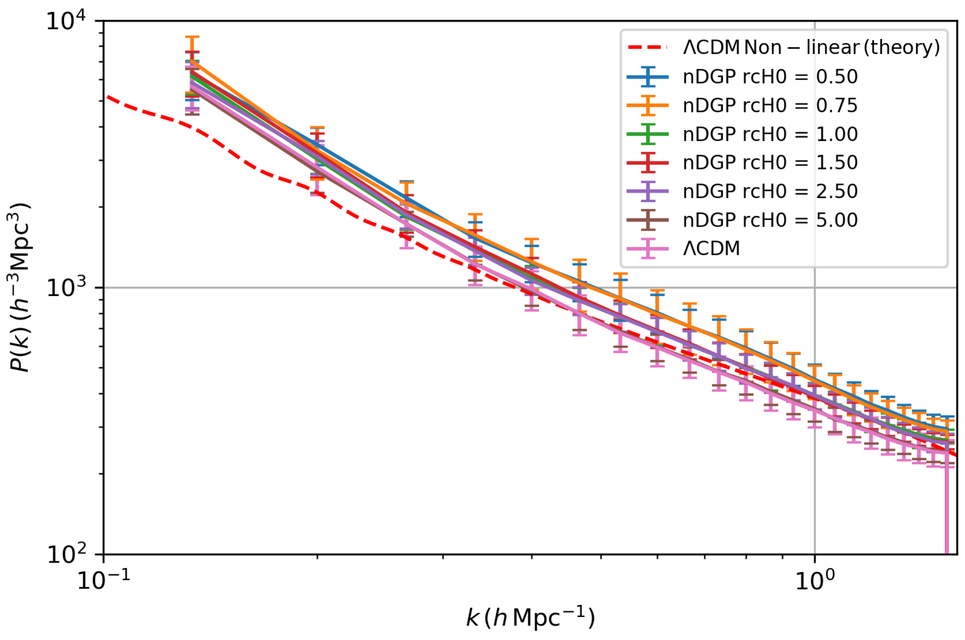


Figure 4.4: Power Spectra of input simulations generated by MG-PICOLA. The coloured lines are averages of 25 simulations randomly drawn from the dataset for each chosen r_cH_0 value. The error bars are the standard deviation of the randomly drawn 25 simulations.

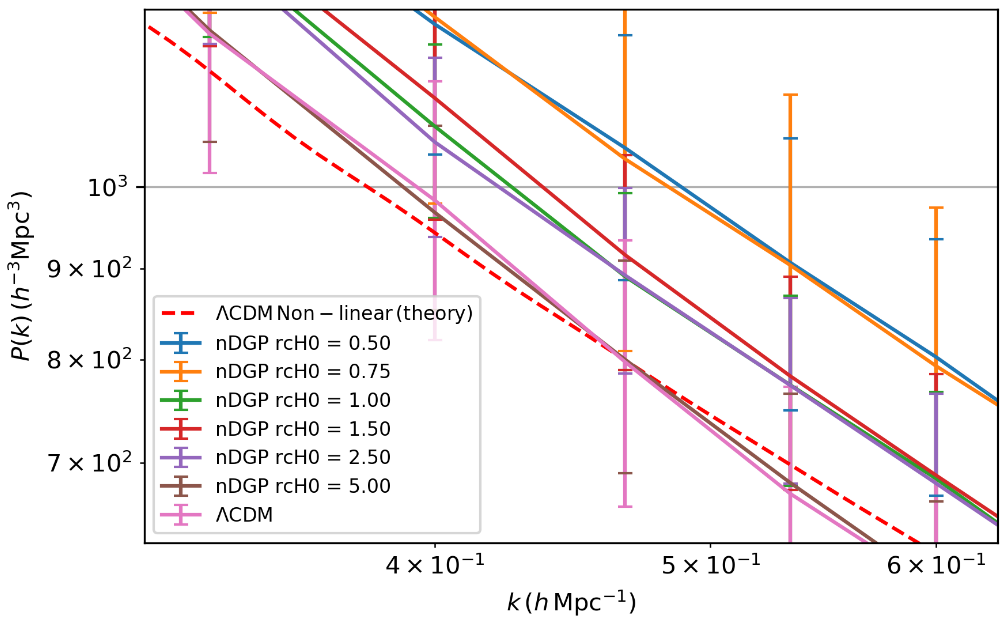


Figure 4.5: Power Spectra of input simulations generated by MG-PICOLA (zoomed). The coloured lines are averages of 25 simulations randomly drawn from the dataset for each chosen $r_c H_0$ value. The error bars are the standard deviation of the randomly drawn 25 simulations.

4.1.4 Minkowski Functionals

Minkowski Functionals are useful when characterising the large-scale structure seen in cosmological simulations (Schmalzing et al., 1996). They provide this information in the form of four geometric parameters for 3D images: volume, surface area, mean curvature, and Euler Number (χ). These are commonly referred to as V_0, V_1, V_2, and V_3 respectively.

As a basic explanation, these parameters are calculated for a range of 'threshold' values (ν) (see Weinberg et al. (1987) and Melott (1990)). For a simulation with constant dark matter particle masses, this threshold value is a parameter that changes the properties of an isodensity surface. The left panel in Figure 4.6 shows a dark matter particle simulation with spheres of small radii decorating the particle positions, corresponding to a small threshold value. The right panel shows what would occur when the threshold value is increased. If the collection of spheres together is considered to be an isodensity surface, the Minkowski Functionals would be the computation of the total volume (V_0), surface area (V_1), mean curvature (V_2), and Euler Number (V_3) of the isodensity surface at varying threshold parameters. Euler Number in this instance would be a measure of the connectivity of the spheres, or how much each sphere intersects with another. When spheres (components) intersect, they create tunnels or cavities; where a tunnel would be a hole through the isodensity surface that connects one side to the other and a cavity would be a hole inside the isodensity surface that does not connect to the outside. Due to this phenomena, measuring the Euler Number gives the ability to characterise filamentary structures. The Euler number is defined as

$$\chi = \text{number of components} - \text{number of tunnels} + \text{number of cavities}. \quad (4.3)$$

In this work, I have adapted code included in two github repositories: garrelt/Size-Analysis-C2-Ray for the code to calculate the Minkowski Functionals in C, and jeremycclo/msci_reionisation to use the code from within python. Both of these repositories use this code in analyzing the size distribution of HII regions (interstellar atomic ionized hydrogen).

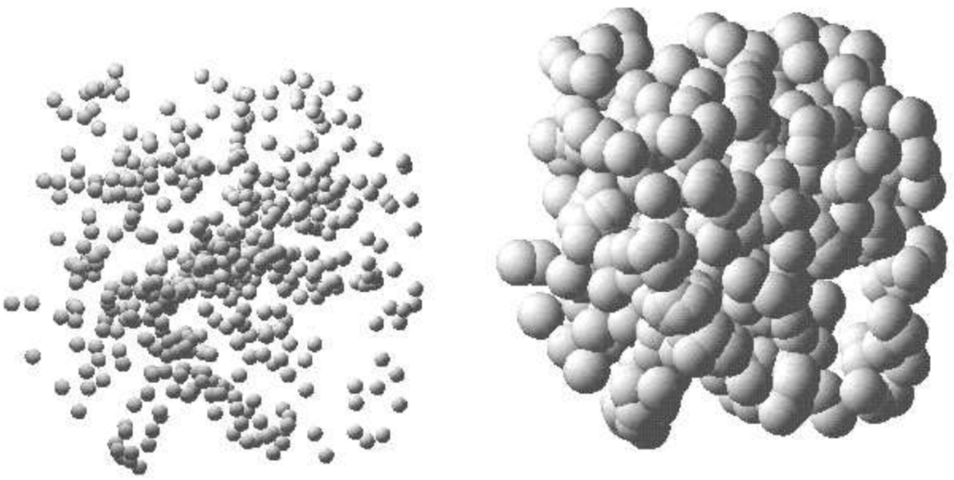


Figure 4.6: Simulation particles with spheres of different radii placed around them representing a low threshold value (left) or high threshold value (right). Figure from Schmalzing et al. (1996).

To aid in understanding, I have calculated the average of ten Minkowski Functionals for Gaussian random fields in a 3D cube and plotted the results in Figure 4.7. The top left panel (V_0) shows that at a low threshold value, the total volume of the isodensity surface is high, and as the threshold value increases the volume decreases. The top right panel (V_1) shows that at a low threshold value, the surface area of the isodensity surface is low, then as the threshold value increases there is a peak in surface area. At high threshold value, the isodensity surface starts smoothing out (imagine the spheres in the right panel of Figure 4.6 increase in radius) and the surface area decreases. In the bottom right panel (V_2), as the threshold value starts increasing, the mean curvature of the isodensity surface decreases, until there is a crossover point and the mean curvature begins to increase until the isodensity surface starts smoothing out at a high threshold value. Finally, in the bottom right panel (V_3), it is seen that there is a rapid depression in connectivity within the isodensity surface at the middle threshold value.

Figure 4.8 shows the average of ten Minkowski Functionals for the ΛCDM and nDGP (r_cH_0 = 0.50) dark matter particle simulations and Figure 4.9 shows the percentage difference between these Minkowski Functionals. Errors are omitted on this plot because they are negligible. It should be noted that in this work, the density threshold parameter ν is a function of the overdensity (δ) in the n-body simulations. The Minkowski Functional in each case is therefore 0 below ν = -1.

When examining these Minkowski Functionals, it is clear that they are all quite similar. The biggest difference between the models of gravity are shown in V_3, the Euler number, with nDGP simulations having a higher peak at $\sim \nu$ = -1. This means that between the simulations, the nDGP model provides more filametary structures than the ΛCDM model. In Chapter 5, I will examine if the CNN model is attempting to use this characteristic to classify simulations by using the activation maximisation method on the CNN models with the density amplitude removed. This will allow the CNN model to produce simulations that it believes looks most like ΛCDM or nDGP gravity, and I will measure the Minkowski Functionals of these outputted simulations.

4.2 Classification of Different Models of Gravity Using CNNs

The aim of this investigation is to determine to what accuracy a CNN can tell the difference between dark matter particle simulations created using ΛCDM and nDGP models of gravity, and how the chosen r_cH_0 value for nDGP simulations affects those accuracies. The CNN model architecture used in this investigation is shown in Table 4.1, with each convolutional layer labelled with numbers 1 to 6. This CNN architecture mimics that of Ravanbakhsh et al. (2017) due to their successes in using this architecture to predict values of σ_8 and Ω_m. The model will be trained with a constant ΛCDM dataset and an nDGP dataset each with a different r_cH_0 value, and the results will infer

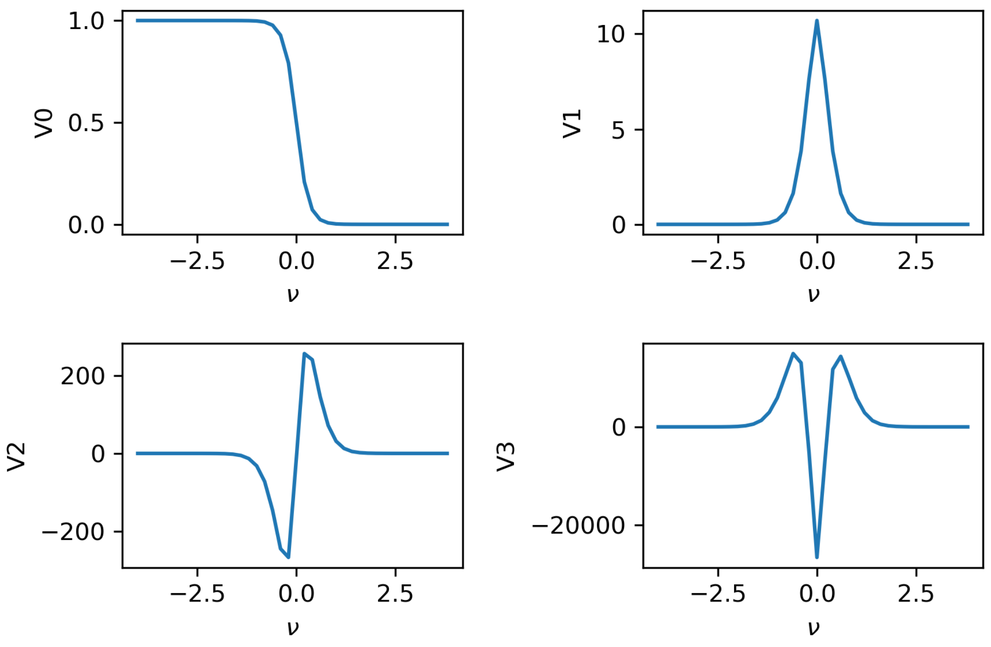


Figure 4.7: Minkowski functionals for a Gaussian distribution.

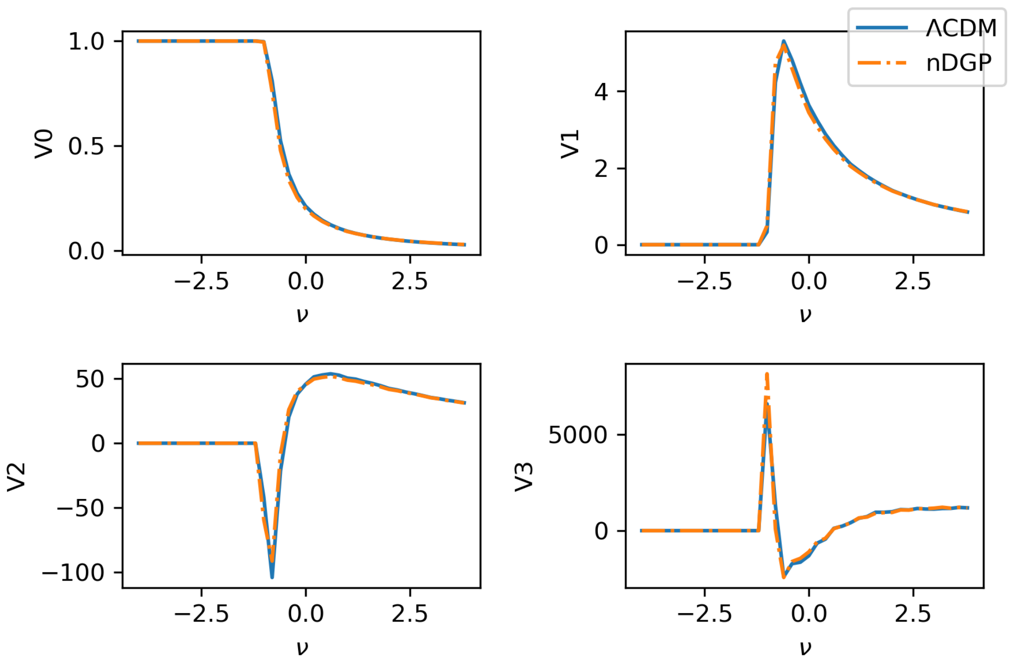


Figure 4.8: Minkowski functionals for the input simulations.

at what r_cH_0 value the CNN model no longer has the ability to tell apart the different models of gravity.

The *Output Shape* column in Table 4.1 shows the dimensions of the simulations at each layer (shown in the *Layer (type)* column). In the top half of the network this is in the format (x, y, z, n_filt), with the x,y,z, variables being the dimensions of the simulation and n_filt being the number of filters in that layer. In the bottom half of the network after the *Flatten* layer, the *Output Shape* takes the form of (units), with the units being the number of neurons in that layer. For consistency, batch size is kept to a size of 25 for each test.

The *Param #* column is the number of parameters in the *Filter Shape* that can be modified during the training process and is calculated from (x * y * z * n_channels + 1) * n_filt. In this calculation, n_channels is the number of filters from the previous layer. This means that the number of parameters in Layer 2 is (4 * 4 * 4 * 2 + 1) * 12 = 1548.

It can be seen here that the dimensions of the filters (input shapes) get progressively smaller as the number of filters get progressively larger. This can be interpreted as a compression of the most relevant information from the largest cosmic scales available in the simulation.

4.3 CNN r_cH_0 Investigation

In this section I present the results from the investigation into the binary classification of ΛCDM and nDGP simulations with differing r_cH_0 values. For each choice of r_cH_0 value, the maximum accuracy of classification associated with the lowest loss value within 25 epochs of training is taken, using a batch size of 25. The resulting accuracies are shown in Table 4.2.

The results in Table 4.2 clearly show that the CNN model can correctly

Layer (no)	Layer (type)	Filter Shape	Output Shape	Param #
1	Conv3D	(3, 3, 3, 2)	(62, 62, 62, 2)	56
	Batch Norm		(62, 62, 62, 2)	0
	LeakyReLU		(62, 62, 62, 2)	0
	AveragePooling		(31, 31, 31, 2)	0
2	Conv3D	(4, 4, 4, 12)	(28, 28, 28, 12)	1548
	Batch Norm		(28, 28, 28, 12)	0
	LeakyReLU		(28, 28, 28, 12)	0
	AveragePooling		(14, 14, 14, 12)	0
3	Conv3D	(9, 9, 9, 64)	(6, 6, 6, 64)	559936
	Batch Norm		(6, 6, 6, 64)	0
	LeakyReLU		(6, 6, 6, 64)	0
4	Conv3D	(3, 3, 3, 64)	(4, 4, 4, 64)	110656
	Batch Norm		(4, 4, 4, 64)	0
	LeakyReLU		(4, 4, 4, 64)	0
5	Conv3D	(2, 2, 2, 128)	(3, 3, 3, 128)	65664
	Batch Norm		(3, 3, 3, 128)	0
	LeakyReLU		(3, 3, 3, 128)	0
6	Conv3D	(2, 2, 2, 128)	(2, 2, 2, 128)	131200
	Batch Norm		(2, 2, 2, 128)	0
	LeakyReLU		(2, 2, 2, 128)	0
7	Flatten		(1024)	0
8	Dense		(1024)	1049600
	LeakyReLU		(1024)	0
	Dropout		(1024)	0
9	Dense		(256)	262400
	LeakyReLU		(256)	0
	Dropout		(256)	0
10	Dense		(2)	514
	Softmax		(2)	0

Table 4.1: The CNN architecture used.

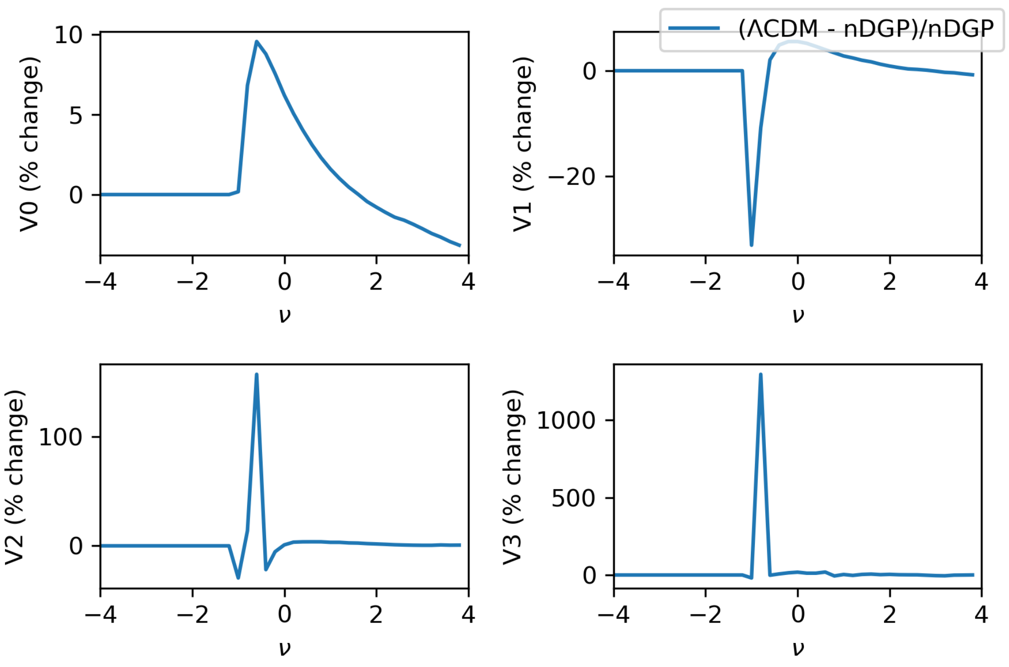


Figure 4.9: Differences in Minkowski functionals for the input simulations.

$r_c H_0$	Highest Accuracy	Accuracy achieved on epoch number
0.50	1.0000	13
0.75	0.9998	25
1.00	0.9944	4
1.50	0.9652	23
2.50	0.5440	1
5.00	0.5470	1

Table 4.2: CNN model classification results.

discriminate between ΛCDM simulations and nDGP simulations, especially when a low r_cH_0 value is used in the nDGP simulations. The fact that the model can classify $r_cH_0 = 0.50$ simulations with an error less than 1 in 5000 is not surprising, given that there is a ∼10% enhancement in the power spectrum across all k-modes when comparing to ΛCDM simulations. Given that the densest regions in the nDGP simulations are higher in average and absolute terms than those in the ΛCDM simulations, the CNN could simply be learning to classify using the maximum densities found in the dataset. In Chapter 5, I investigate whether this is likely to be what is happening.

What is particularly interesting about the results in Table 4.2 is the fact that the CNN model can still discriminate between the models of gravity to an accuracy above 96% when the nDGP simulations have an r_cH_0 value of 5.00, a ∼5% enhancement in the power spectrum across all k-modes. With simulations of differing models of gravity and two-point statistics so close, it is possible that there are many ΛCDM simulations in the dataset with denser regions than nDGP simulations, suggesting the CNN model is using other features than the average and absolute values of the densest regions to discriminate between models of gravity. In Chapter 5, I attempt to test the CNN model to determine how it is able to classify these simulations.

Figure 4.11 shows accuracy of classification, and Figure 4.10 shows the loss (defined in Section 2.2.2) during the training of the CNN using nDGP simulations with a r_cH_0 value of 0.50. It can be seen that loss gradually decreases as the epoch and accuracy increases until the fourth epoch, where the loss starts dramatically increasing. This means the gradient descent algorithm had found a local minimum and then suddenly abandoned it. This can be due to a number of reasons and each solution must be tested to determine the cause. One reason the loss may have started to increase is if the learning rate is too large and not adaptive. To counter this in future investigations into this topic, the learning rate could be reduced when the loss plateaus, further refining the loss in a given local minimum instead of stepping out of it. Another reason may be if the chosen gradient descent

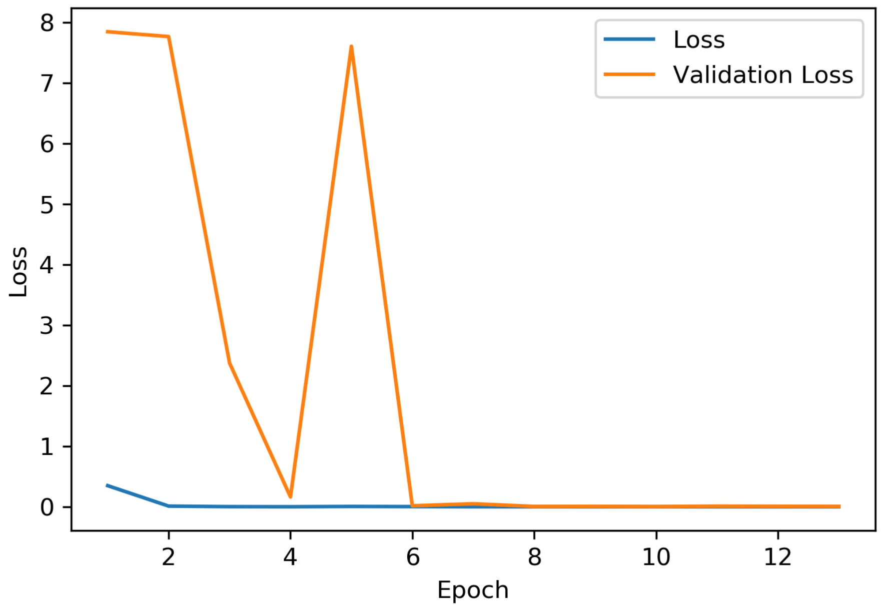


Figure 4.10: Loss during training over 13 epochs, where the blue line is training loss, and the orange line is validation loss. The r_cH_0 value in the nDGP simulations is set to 0.50 for this test.

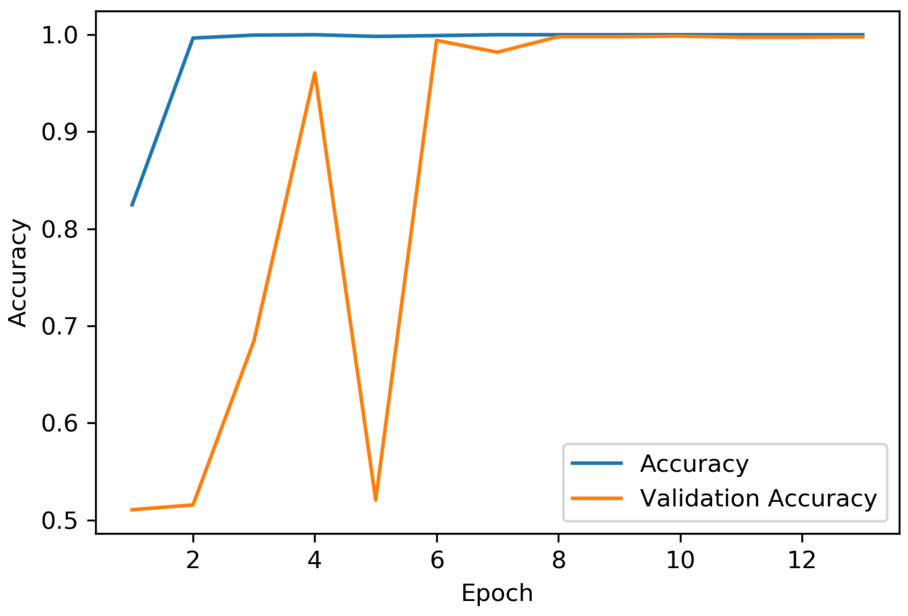


Figure 4.11: Accuracy during training over 13 epochs, where the blue line is training accuracy, and the orange line is validation accuracy. The r_cH_0 value in the nDGP simulations is set to 0.50 for this test.

algorithm incorporates an element of momentum. This would mean when the loss initially decreases in the early epochs, the algorithm gains momentum and steps out of the local minimum in later epochs. One other common reason the loss may have started to increase is because the CNN model begins to overfit the training data in the later epochs, which can be countered by the stronger application of regularisation techniques such as dropout. Given that this particular trained CNN model classifies the different models of gravity with an error of less than 1 in 5000, the reason for this fluctuation in loss will not be further investigated.

4.4 CNN r_cH_0 Investigation (removed amplitude)

In this section, I perform the same tests as in Section 4.3 with one key difference - I divide each simulation (δ field) by its standard deviation (hereafter referred to as simulations with removed amplitude information). This gives each simulation a standard deviation of one, and normalises their power spectrum amplitudes. The reason this is done is to ensure the CNN cannot simply use the power spectrum amplitudes to classify each simulation as one model of gravity or the other; it must be classifying using a different feature unique to each simulation, such as the distribution of particles (Figure 4.2) or the mean isodensity curvature as shown in Section 4.1.4. How these models are classifying simulations will be also investigated in Chapter 5.

The resulting accuracies of this test are shown in Table 4.3. When comparing the results to that in Table 4.2 it is seen that the accuracies are not as high and there is not a meaningful classification result when r_cH_0 was more than 1.00. Also, the model did not improve after training on one epoch of data (10,000 simulations), indicating that the signal in the data was not prominent enough to develop a well defined way to classify the simulations. However, this result shows when removing amplitude information from the dark matter particle simulations, CNNs can still classify different models of

$r_c H_0$	Highest Accuracy	Accuracy achieved on epoch number
0.50	0.8826	12
0.75	0.6336	1
1.00	0.6552	1
1.50	0.5556	2
2.50	0.5212	1
5.00	0.4940	1

Table 4.3: CNN model classification results (removed amplitude).

gravity correctly. This implies that the CNN is utilising the differences of the mean isodensity curvature of the simulations as shown in Section 4.1.4. This result is particularly interesting, considering the typical use cases of CNNs (as described in Section 2.2 are to identify shapes and patterns, which may be partly reflected in complex phenomena such as the mean isodensity curvature.

4.5 New Types of Models

So far throughout this chapter, I have shown that CNNs with a standard architecture (Table 4.1) comparable to that seen in the original LeNet paper (Lecun et al., 1998) are able to classify dark matter particle simulations by their theory of gravity. In this section, I present the results from the same analysis using Siamese Networks, and outline new and upcoming methods that will be useful for analysing and generating cosmological simulations in the future. This is the first known application of CNNs of the "One-shot learning" variety to cosmological data, and I initiated and carried out all aspected of this investigation.

4.5.1 Siamese Networks

Siamese networks (Bromley et al., 1993; Chopra et al., 2005; Koch, 2015) are a sub-category in a machine learning area called "One-shot learning" (Fei-Fei et al., 2006; Lake et al., 2011)- a method in computer vision problems where the number of training examples required is greatly reduced. Siamese networks do this by having two identical networks in parallel that share weights between them and a loss function that focuses on the differences between the two images. This means that instead of having one network learning about both models of gravity, there can be two networks learning about the differences between the two. Consequently, the objective changes from "which model of gravity is this?" as investigated in Section 4.2 to "is this model ΛCDM or not?". As with all machine learning methods, they have their own strengths and weaknesses. Siamese networks are limited due to the fact that using them to investigate multiclass problems can be memory intensive, especially with 3d datasets, as multiple training examples need to be fed to the parallel networks simultaneously. However, they are beneficial because sharing the weights across the parallel networks means there are fewer parameters in total, meaning less training data is required and consequently there is less tendency to overfit. Also, if the inputs are of similar nature and because each network learns about each class, the parallel networks are somewhat simpler to compare.

The loss function that was previously used in Section 4.2 (binary cross entropy) is not appropriate for Siamese networks and instead, a contrastive loss function (Hadsell et al., 2006) which focuses on the difference between classes is a better choice. This is done by focusing on the euclidean distance, D_w between the outputs of the networks,

$$D_w = \sqrt{\{G_W(X_1) - G_W(X_2)\}} \tag{4.4}$$

where G_W is the output of one side of the parallel networks, and X_1 and X_2 is the input data pair. This distance is seen in the Siamese network architecture shown in Figure 4.12. In my case, Layer 10 in Table 4.1 is removed from the

Layers	Accuracy	Accuracy achieved on epoch number
0.50	0.9258	24
0.75	0.8814	12
1.00	0.9716	23
1.50	0.7458	21
2.50	0.5078	10
5.00	0.5016	10

Table 4.4: Siamese CNN model classification results.

standard CNN architecture and the contrastive loss function is used in its place, assessing the output from two identical copies of the layers before it. The contrastive loss function is defined as:

$$C_{loss} = (1 - Y)\frac{1}{2}(D_w)^2 + (Y)\frac{1}{2}\{max(0, m - D_w)\}^2 \tag{4.5}$$

where C_{loss} is the contrastive loss and Y is where the input pairs match or are different. For instance, if the input pair is both ΛCDM simulations, Y would be 0, or if one simulation is ΛCDM and one is nDGP, Y would be 1. max is a function specifying the bigger value between 0 and $m - D_w$, and m is a margin value that ensures disparate pairs beyond this margin do not contribute to the loss. This causes the network to optimise based on pairs that it evaluates as similar, but are actually disparate.

I have performed the gravity model classification with Siamese networks with code adapted from examples on the Keras team's Github repository. The results are shown in Table 4.4.

Table 4.4 shows that it is clear that Siamese networks are able to classify ΛCDM and nDGP simulations correctly, but are in fact slower than using a standard CNN when using the same batch size. Each epoch took approximately twice the time to compute (the hardware is dealing with two simula-

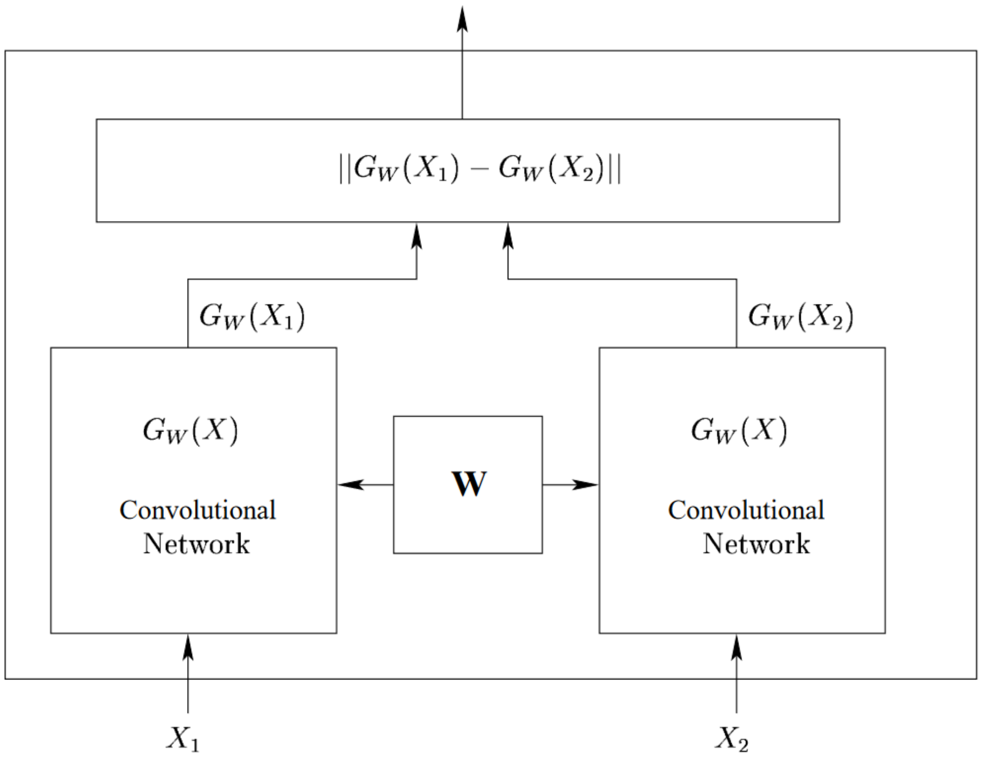


Figure 4.12: Siamese CNN Architecture. Figure taken from Chopra et al. (2005).

tions simultaneously), and the accuracies reached are not as high (accuracies were > 0.99 for $r_c H_0 <= 1.00$), and accuracies were reached in higher epochs than with standard CNNs.

It should be noted that when training the Siamese networks, the loss reduced in a much more stable fashion than using standard CNNs. This will be due to the fact that a contrastive loss had to be used for training Siamese networks, and not the standard binary cross entropy loss. Taking this into consideration, future work could focus on leaving Siamese networks to train for longer to determine whether the change in loss function and architecture could lead to overall higher accuracies for the nDGP simulations with higher $r_c H_0$ values.

4.6 Further Discussion of Results

While this investigation demonstrates that dark matter particle simulations of different models of gravity can be classified using both standard CNNs and Siamese networks to a high degree of accuracy depending on the $r_c H_0$ value, the method and results could be improved upon for all of the tests presented in this chapter.
Regarding the method, in the future with access to more hardware, the results could be made more robust by performing the training multiple times and averaging the accuracies per $r_c H_0$ value. This would reveal whether there is a meaningful result when trying to classify nDGP sims with an $r_c H_0$ value of 2.50 or higher. This could be done in conjunction with a gridsearch, exploring various learning rates, batch sizes, loss functions, and CNN architectures to maximise classification accuracy.

When considering the performance of the Siamese networks, it should be noted that just as the standard CNN test was adapted from the classification of handwritten digits in the MNIST dataset, the generally accepted type of problem to solve with Siamese networks is usually facial recognition/classification tasks. These datasets contain more than one person (a multiclass problem), with images taken from all different angles (samples from the same class). This is different to the dataset used in this investigation, as I am using only two classes; however, due to the rotation of the simulations and simulations generated from different initial conditions, my dataset does contain many different examples of the same class, mimicking the faces from different angles. It is for this reason that I believe that better quality results could be obtained by making this a multiclass problem. This could be done by either having a constant ΛCDM dataset with a nDGP dataset containing all different $r_c H_0$ values, or making the nDGP dataset a modified gravity dataset that contains more than one model of modified gravity (such as the inclusion of simulations created with $f(R)$ gravity).

4.6.1 Generative Adversarial Networks (GANs)

A more recent development in machine learning is that of Generative adversarial networks (GANs), first introduced by Goodfellow et al. (2014). GANs work by training two separate networks simultaneously, with one network producing fake training examples (generative), and one network discriminating between fake and real training examples (adversarial). These networks train in a competitive manner in order to better each other, meaning the adversarial side will teach the generative side to create better fake training examples, and in turn the generative side will teach the adversarial side to better discriminate between the real and fake training examples. This is done until a loss function is minimised, and either the networks cannot further progress each other, or the adversarial network can not discriminate between real and fake training examples.

Methods such as GANs are particularly interesting for the interpretation of machine learning algorithms. If one were to create a GAN that could generate unique dark matter particle simulations that were indistinguishable from simulations created in the traditional manner (with programs such as L-PICOLA or GADGET), they could be analysed and forensically compared to each other to help determine the successes and failures of the machine learning model. This would shed light on how machine learning algorithms best model our universe, and vastly reduce the computational resources required for traditional methods. Early work into GANs that can produce dark matter particle simulations has already started, with 500 and 100Mpc 2d simulations producing results that are qualitatively and quantitatively very similar to the original training data as shown by Rodriguez et al. (2018). Due to time constraints, I have left tests using GANs to future work.

Chapter 5

Testing Models of Gravity Using Machine Learning - Interpretation

In this Chapter, I expand upon various methods described in Chapter 2 and present results from them using the CNN models created in Section 4.2. This chapter deals with two methods to investigate how the CNN models are correctly classifying simulations as one model of gravity or the other. Results from the Model Sensitivity Analysis method are presented in Section 5.1, along with the caveats of the method. The results from the Activation Maximisation method are presented in Section 5.2, and the caveats of this method are also discussed. The chapter concludes with ways to address these caveats and improve these techniques, and a discussion about work that could be undertaken in the future to expand the research into this area.

5.1 Model Sensitivity Analysis

In this section, I explain the specifics behind the method described in Section 2.2.9. Firstly, the testing dataset must be created.

Power Spectrum

To infer whether the CNN model is using two-point statistics alone to classify the dark matter particle simulations, I first create a set of power laws similar to the measured power spectra of the CNN training datasets as shown in Figure 4.4. These power laws are created using a set of amplitudes (A) and powers (n) over a set range of k bins, with the equation taking the form Ak^n. In this test, 30 amplitudes each with 30 power values were chosen, totalling 900 power laws. The amplitudes are logarithmically spaced between $5.4 \cdot 10^1 < A < 8.6 \cdot 10^1$, and the power values are linearly spaced between $-2.5 < n < -0.1$. These values are chosen by an iterative search, where it has been tested that the CNN model will classify Gaussian random fields generated using different power laws as either ΛCDM gravity or nDGP gravity (and not simply classify every generated Gaussian field as one model of gravity). The power laws are kept constant when testing each CNN model.

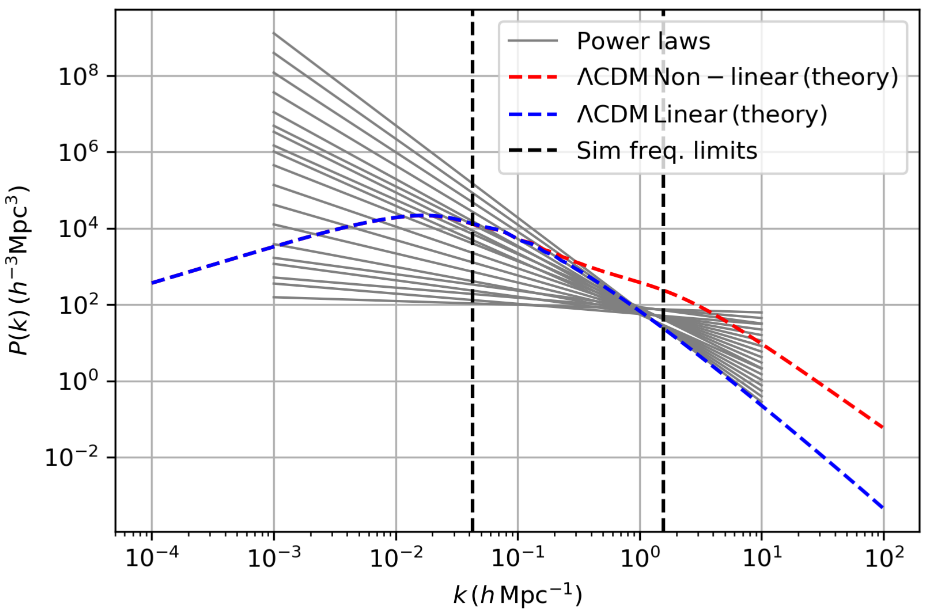


Figure 5.1: Power laws used in Model Sensitivity Testing.

It is clear when looking at the chosen power laws in Figure 5.1 that they do not lie near the amplitude of the non-linear part of the power spectrum (red dashed line), especially at small scales. This appears to be due to the how the CNN treats the differences between the simulation overdensities compared to the generated Gaussian random fields. A major way these differ is in the magnitudes, with the overdensities being in the hundreds, and the generated fields being in the tens. This may be a contributing factor as to why the CNN is only able to classify data using the amplitude of the large scales.

5.1.1 Generated Gaussian Random Fields

For this part of the investigation, the pre-trained CNN model using nDGP simulations with $r_c H_0$ values of 0.50, 0.75, 1.00 and 1.50 are used, as they are the only CNN models that could classify the simulations to an accuracy higher than that of 54%. Results of the CNN models with an $r_c H_0$ value above 1.50 are not presented. Each generated Gaussian distribution drawn from the power laws shown in Figure 5.1 is classified by each CNN model a total of 10 times and averaged to ensure the result is robust.

5.1.2 Model Sensitivity Analysis Results

Figure 5.4 shows the results when using the trained CNN in Section 4.3 (with nDGP simulations with $r_c H_0 = 0.50$) to classify generated Gaussian random fields from power laws shown in Figure 5.1. The amplitude increases on the x-axis and the gradient decreases on the y axis (with -2.5 being the steepest and -0.1 being the shallowest). Yellow colour in the plot indicates where the CNN classifies a generated Gaussian random field as ΛCDM gravity, and blue indicates a nDGP classification. If there were no gradient from yellow to blue, Figure 5.4 would show a definitive boundary in classification between fields created using different amplitudes and slope of power laws. In this result it is seen that nearly any distribution drawn from a power law with an amplitude

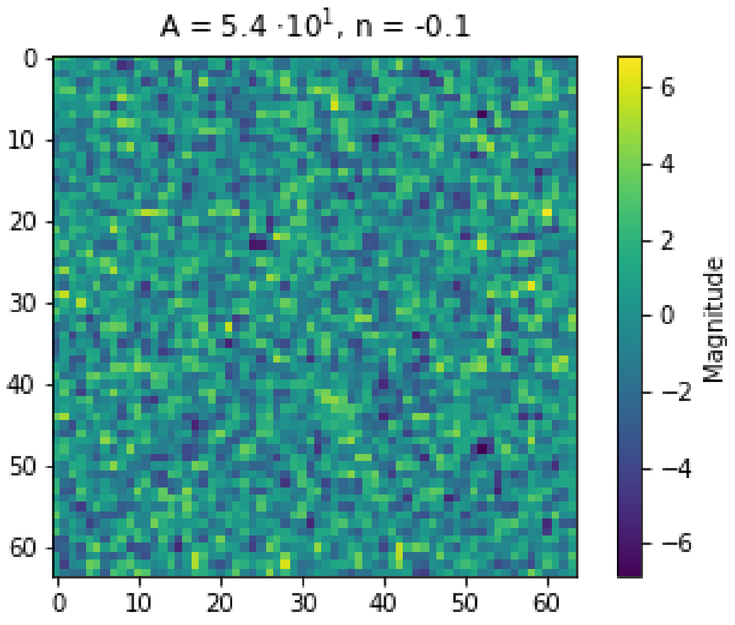


Figure 5.2: 2D Slice through 3D generated random Gaussian field created for Model Sensitivity Testing (low amplitude shallow gradient).

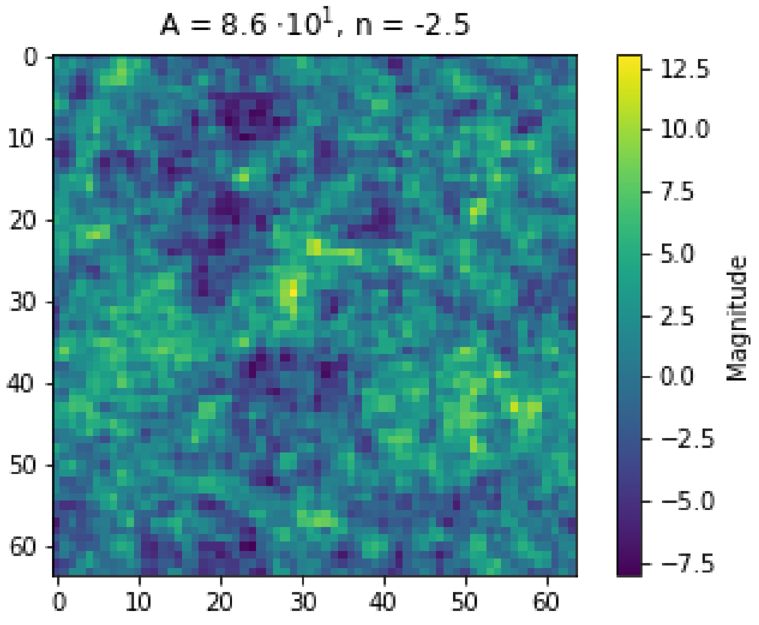


Figure 5.3: 2D Slice through 3D generated random Gaussian field created for Model Sensitivity Testing (high amplitude steep gradient).

lower than $6.3 \cdot 10^1$ is classified by the CNN as ΛCDM. The CNN model clearly has trouble classifying the distributions with a gradient steeper than -2.0, especially at high amplitudes. Figure 5.4 also shows that distributions with shallow power laws and high amplitudes are typically classified as nDGP gravity.

Given Gaussian random fields created using the same parameters, Figure 5.5 shows that when testing with the CNN model trained with simulations where $r_c H_0 = 0.75$ (power spectrum tending closer to that of ΛCDM as $r_c H_0$ increases), the CNN model classifies any Gaussian random field drawn from a shallow power law as nDGP, regardless of amplitude. The boundary to define fields as ΛCDM for this model is less well defined than in Figure Figure 5.4, with the CNN model continuing to struggle to definitively classify steeper power laws as one model of gravity or the other.

Figure 5.6 shows a similar classification boundary to that seen in Figure 5.4, however for the fields drawn from the steepest amplitudes (n < -2.0), the CNN has trouble making any meaningful classifications regardless of amplitude.

Figure 5.7 shows a clearer classification boundary in amplitude for the CNN model using nDGP simulations with an $r_c H_0 = 1.50$ than any of the other CNN models. This is interesting because this is the first time a strict boundary has been drawn using amplitudes, implying that as the $r_c H_0$ value increases as the nDGP simulations start to increasingly resemble the ΛCDM simulations, the CNN model seems to have changed the way it classifies simulations.

5.1.3 Model Sensitivity Analysis Results (removed amplitude)

Upon performing the same model sensitivity analysis as in Section 5.1.2, the CNN models from Section 4.4 classified every generated Gaussian random field as ΛCDM gravity. Also, when varying the amplitudes of the generated Gaussian random fields through and iterative search, the model continued

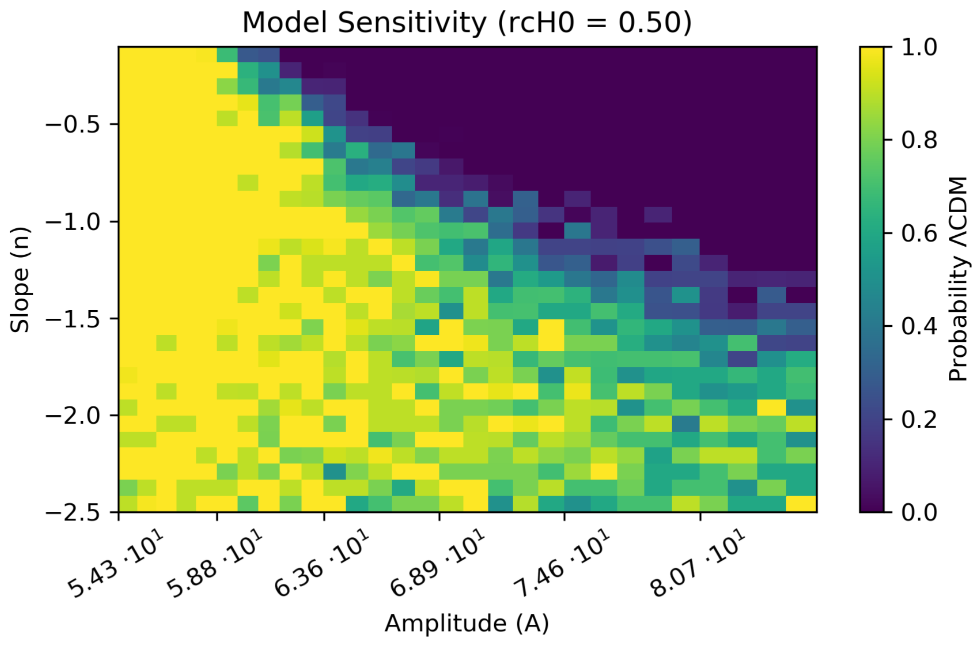


Figure 5.4: Results from tested power laws using generated Gaussian random fields where $r_c H_0 = 0.50$.

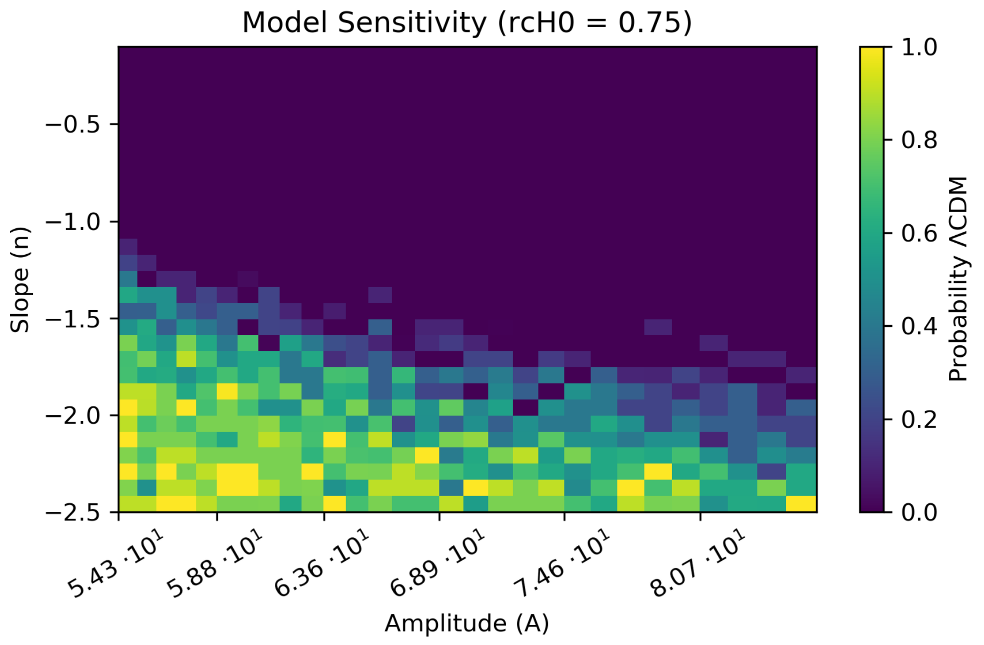


Figure 5.5: Results from tested power laws using generated Gaussian random fields where $r_c H_0 = 0.75$.

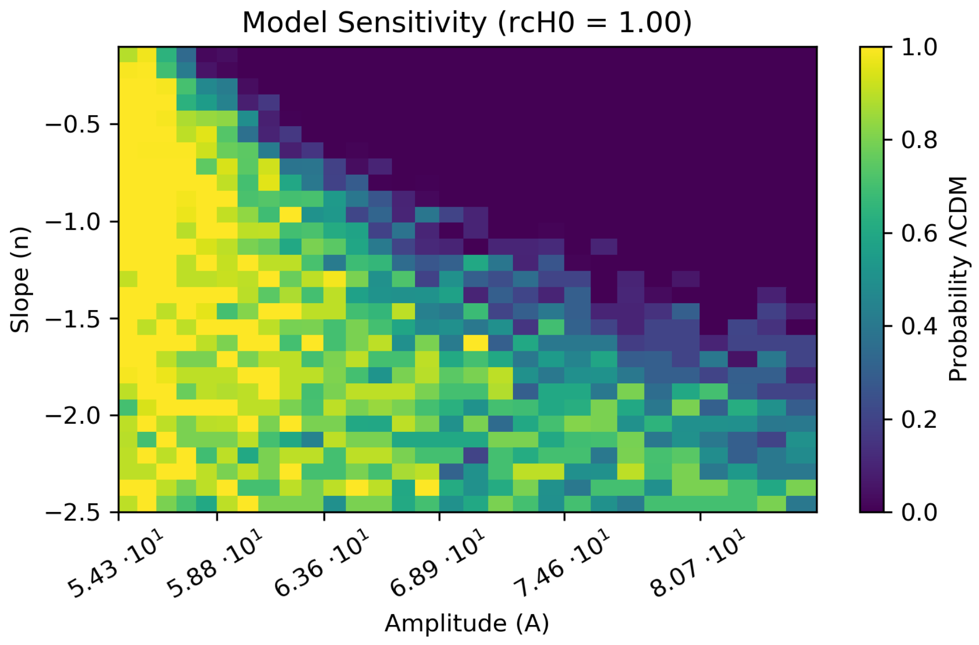


Figure 5.6: Results from tested power laws using generated Gaussian random fields where $r_c H_0 = 1.00$.

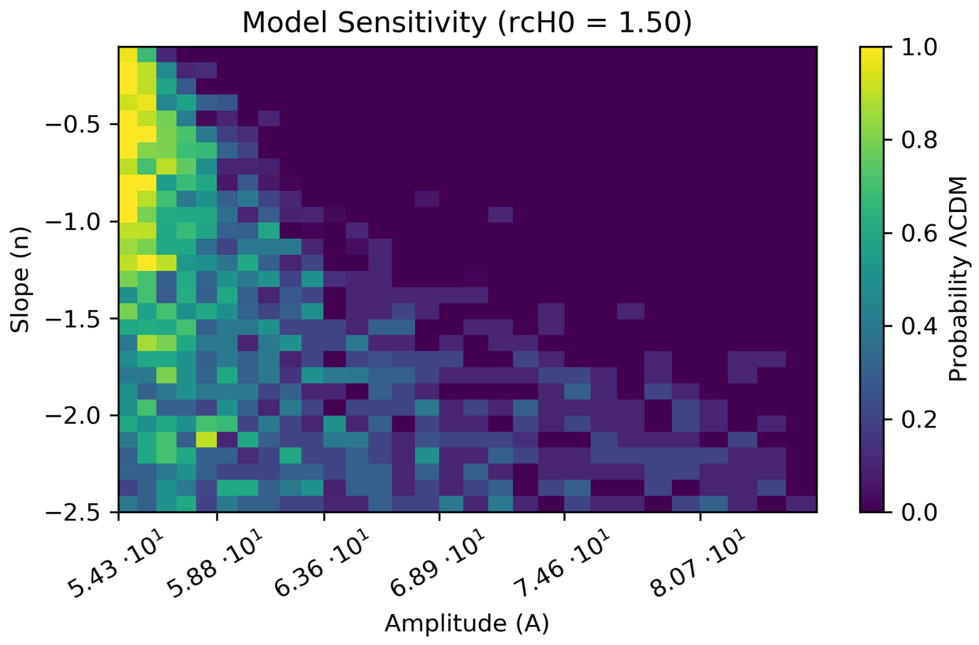


Figure 5.7: Results from tested power laws using generated Gaussian random fields where $r_cH_0 = 1.50$.

to classify every field as ΛCDM gravity. It is for this reason that Figures similar to 5.4 - 5.7 could not be produced for this section. It seems likely that this is because the CNN did not learn about the amplitude of matter density (or differences in this) for any of the types of simulations provided and therefore classified every simulation as one model of gravity to achieve the highest accuracy it could - 50%.

5.2 Activation Maximisation

In Section 2.2.9 I outlined the Activation Maximisation (AM) method, which activates the CNN in such a way that it could change a distribution of randomly initialised values to values that the CNN is most likely to classify as a certain model of gravity. This essentially attempts to create the input image that the CNN model would classify as a ΛCDM or nDGP simulation as certainty. For the remainder of this work I will call this output the 'AM simulation', with one being produced for each model of gravity. For this investigation I used the CNN model where the nDGP simulations had an $r_c H_0$ value of 0.50, and I outputted AM simulations for the CNN model with normal density amplitudes and the one with the amplitudes removed. The reason I have chosen the model where $r_c H_0 = 0.50$ for the nDGP simulations is because it had the highest classification accuracy compared to models with other $r_c H_0$ values. This was also the case when retaining amplitude information or removing it, inferring that the model has been able to learn about features other than the matter density amplitudes. The settings I used for the AM simulation production are the default ones set in the keras-vis package (discussed in Section 2.2.9) and are as follows: 200 iterations (backpropagation of loss cycles), LP Normalisation weight of 10, Total Variation weight of 10, and AM loss weight of 1.

It is assumed that a valid method for determining whether the AM method is working is to attempt to classify the AM simulation with the CNN that produced it. It should classify the AM simulation as the model of gravity it represents with absolute certainty. However, the method fails to pass this test. The fact that the limits of the AM simulation have been set to (the default values of) between -1 and 1 results in a similar normalisation problem as described in Section 5.1, where the CNN is expecting inputs of simulation overdensities, not scaled values between -1 and 1. This results in the CNN models not being able to classify the AM simulations correctly. This is not the case when applying the AM technique to the MNIST dataset, as is in the example on the keras-vis package's repository. The explanation for this is that the MNIST dataset's training samples (the pixelated images of the handwritten digits) always have a constant greyscale pixel value ranging from 0 to 255, and therefore, the generated AM image of a desired class can also have set limits between 0 and 255. This leads to a correct classification of the generated AM image from the MNIST data, as the CNN is expecting values between 0 and 255, and the AM image adheres to that requirement, thus verifying the AM method for that example. When attempting to use the AM technique with my simulation datasets however, each ΛCDM or nDGP simulation does not have set limits of what the maximum number of particles in a voxel can be, leading to each simulation having its own maximum magnitude of matter density. This causes the verification method of correctly classifying AM simulations with the CNN model that generated them to be inappropriate. It should be noted that when AM images are generated for the MNIST dataset, they do classify as the number they are meant to represent.

Acknowledging these issues, there is no problems with investigating the fact that the AM simulations between the models of gravity are quite different. As such, I have measured the histogram of overdensities, power spectra, and Minkowski Functionals of outputted AM simulations created using CNNs with amplitude information included, and with it removed. I have then averaged these measurements to investigate the particle distributions, density amplitudes, and topologies of the AM simulations. These measurements will provide insight about what the CNN has been able to identify as features in the training sets, giving insight into how important the amplitude information is and if CNNs can utilise other information such as isodensity curvature.

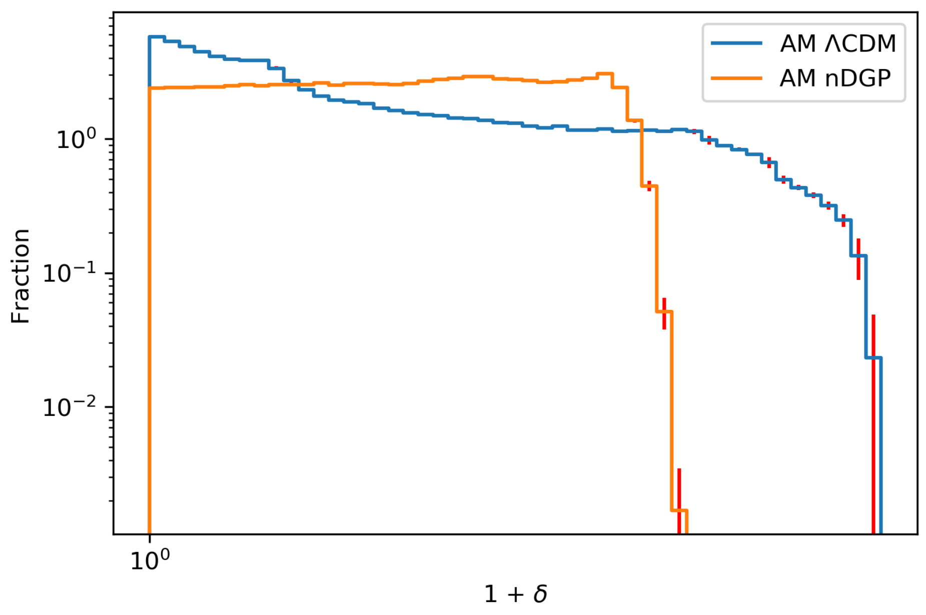


Figure 5.8: Histogram of overdensities of AM simulations generated by a CNN model. Blue and orange lines are the averaged histograms of 25 AM simulations. Errors are the red lines and are the standard deviation of the 25 histograms of the AM simulations. The errors do extend beyond the visible range of the y-axis and are not shown.

Figure 5.8 shows 25 averaged histograms of AM simulations for both ΛCDM and nDGP models. The axes have been displayed logarithmically in order to enhance the differences between the simulations. It is seen here that the distributions in each AM simulation are vastly different, with the ΛCDM AM distribution more closely resembling the distribution seen in the ΛCDM input simulation (Figure 4.3). Towards the middle of the x-axis, the orange line is higher than the blue line, indication that the nDGP AM simulation has denser regions than the ΛCDM AM simulation. This is also seen (albeit more subtly) in the histograms of the input data when the y-axis is displayed logarithmically (Figure 4.3). This result shows that the ANN was able to learn a more realistic particle distribution in the case of ΛCDM, but also learned that nDGP has denser regions.

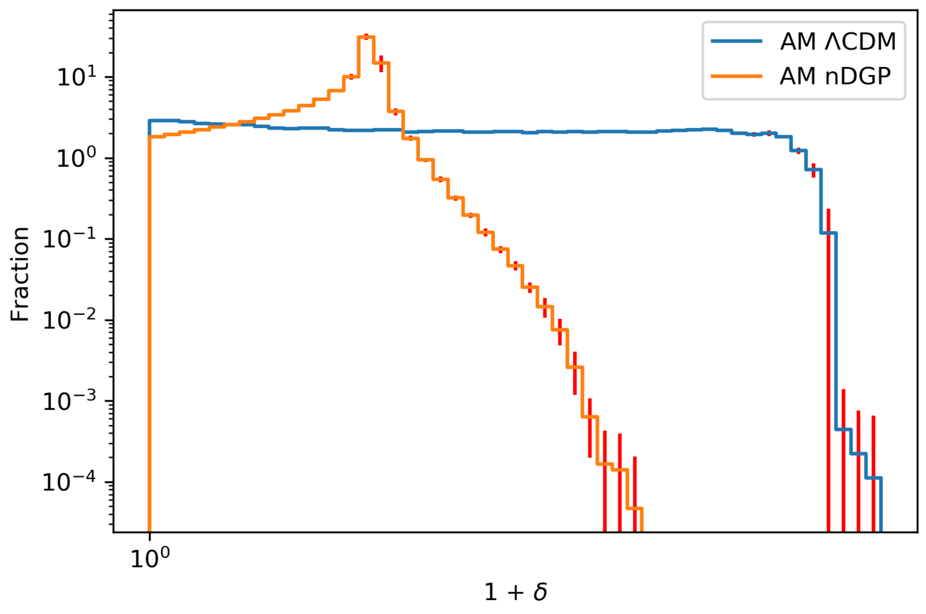


Figure 5.9: Histogram of overdensities of AM simulations generated by a CNN model (removed amplitude). Blue and orange lines are the averaged histograms of 25 AM simulations. Errors are the red lines and are the standard deviation of the 25 histograms of the AM simulations. The errors do extend beyond the visible range of the y-axis and are not shown.

Figure 5.9 shows 25 averaged histograms of AM simulations for both ΛCDM and nDGP models, but created using a CNN trained on simulations with removed amplitude information. Towards the center of the plot is a large peak in the orange line (nDGP AM simulation). It seems likely that this was created because the CNN which trained on simulations with amplitude information removed had to enhance the density in that region in order to accentuate the differences between the models of gravity.

5.2.1 Power Spectra of Activation Maximisation Simulations

Figures 5.10 and 5.11 are calculated from 25 AM simulations using CNN models with and without density amplitudes. For both models ($r_c H_0 = 0.50$ in both models, and one has amplitude information, the other does not), I calculated the power spectra and averaged the results. The error bar shown is the standard deviation of the averaged power.

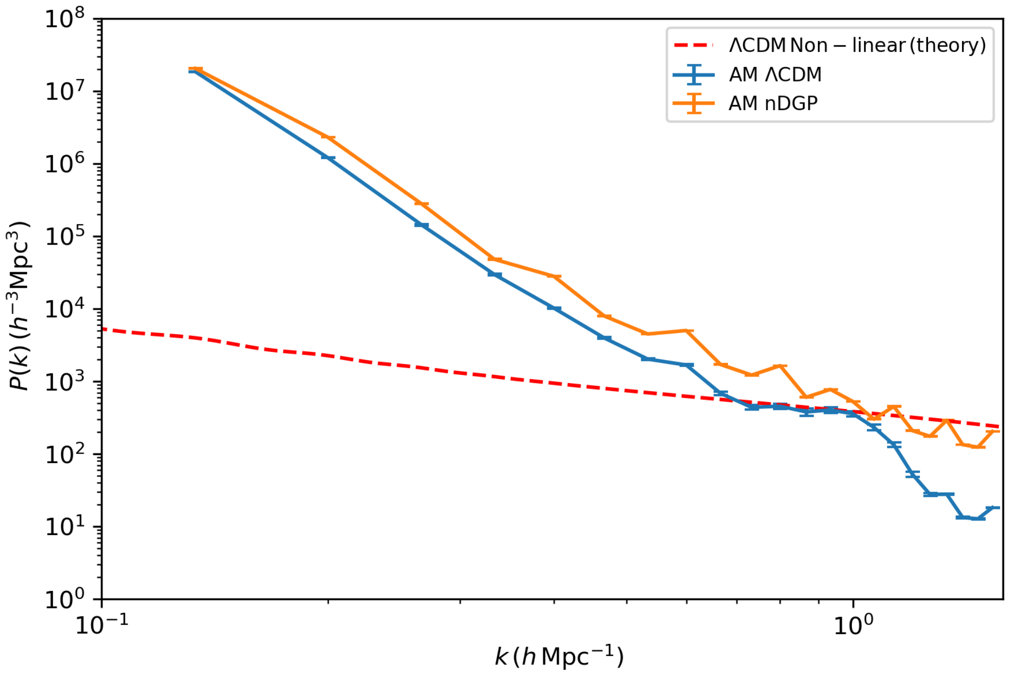


Figure 5.10: Power spectrum of Activation Maximisation simulation using a CNN model that includes matter density amplitude information.

As discussed earlier, it is possible that the normalisation issues faced in this investigation contribute to the power spectra shown in both Figures 5.10 and 5.11 being offset from the prediction from theory (red dashed line). Also, the parameters used to create the AM simulations (LP-normalisation regularisation, Total Variation regularisation, number of iterations etc.) could have contributed to the seen overall deviation in the power spectrum from

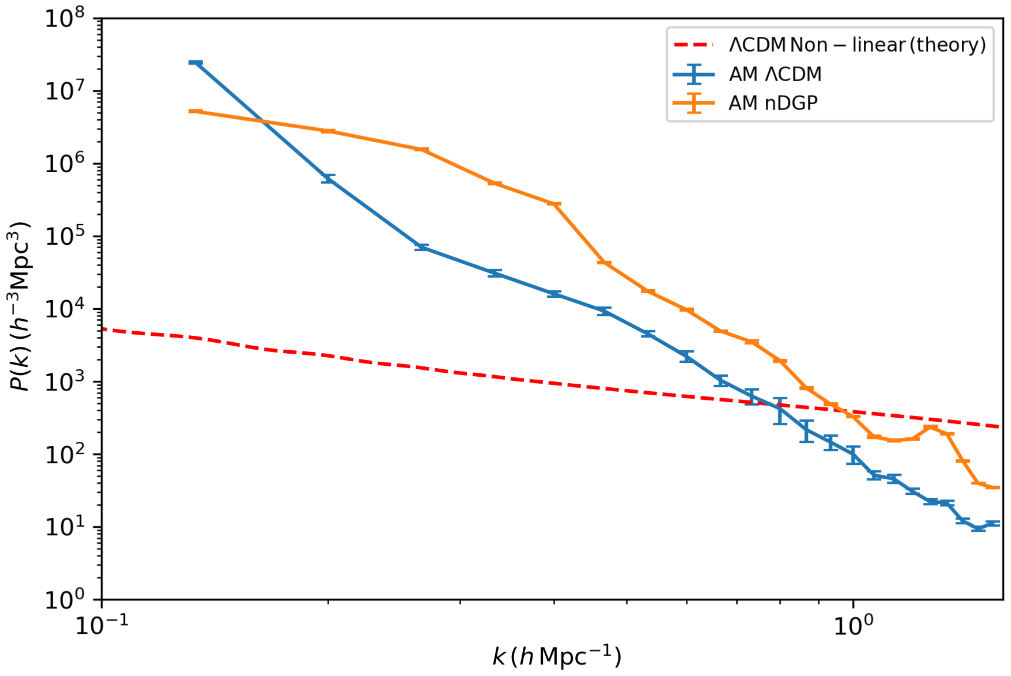


Figure 5.11: Power spectrum of Activation Maximisation simulation using a CNN model that does not include matter density amplitude information.

the theory. Disregarding the fact that the power spectrum of the AM simulations shows that there are some issues with the result, it is interesting to note that in both cases (AM simulations created using CNNs trained on simulations with and without amplitude information), the nDGP AM simulations has higher power across all k-modes. This was the case earlier when examining the nDGP simulations in the training set (Figures 4.4 and 4.5) at low $r_c H_0$ values. This would be expected when the model has been made aware of the differences in the amplitude of the matter density (Figure 5.10). However, the CNN model that trained on simulations where the amplitudes were normalised between ΛCDM and nDGP simulations also produces AM simulations with a significant difference in amplitude of the matter density. In fact, the difference in power across all k-modes is more pronounced. This is a somewhat surprising result, considering the CNN was never trained with any simulations (of either ΛCDM or nDGP) with amplitude information included. This is also seen in the particle distributions of the AM simulations created using CNNs trained on simulations with amplitude information removed (Figure5.9. To be able to reproduce the denser regions seen in the nDGP simulations, the amplitudes in the AM simulations have been enchanced across all k-modes.

Hopefully more information about these AM simulations can be gained by investigating their Minkowski Functionals.

5.2.2 Minkowski Functionals of Activation Maximisation Simulations

Figures 5.12 and 5.14 show the averaged Minkowski Functionals from 10 AM simulations (both ΛCDM and nDGP) created using CNN models trained on data and without the inclusion of matter density amplitude information. The error bars shown are the standard deviation of the averaged Minkowski Functionals. Figures 5.13 and 5.15 show the percent differences between them.

Figure 5.12 shows that the result for V_1, the surface area of the isodensity

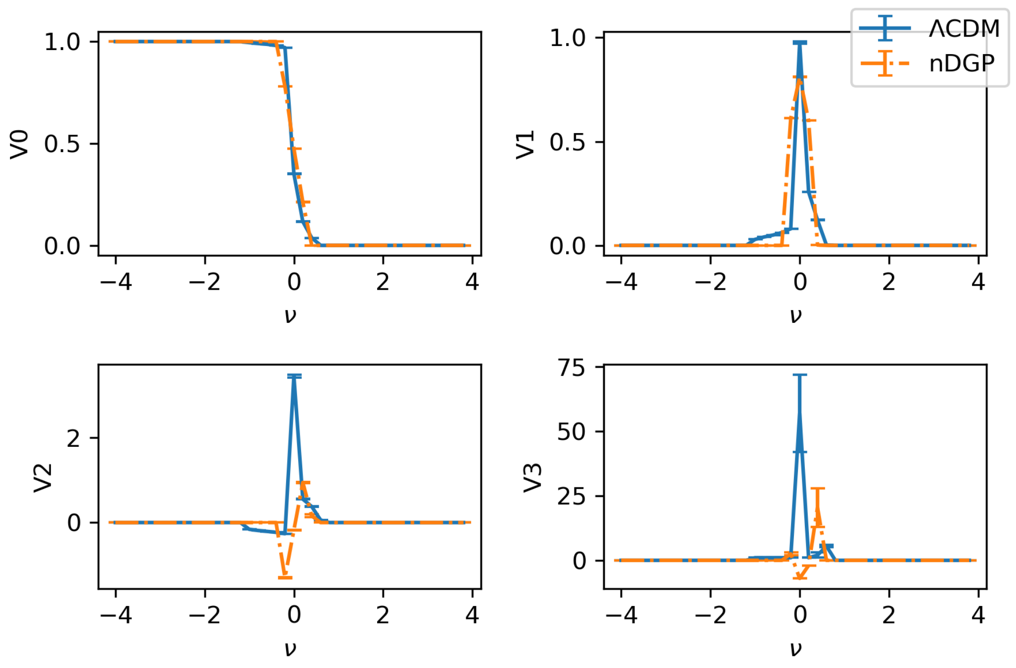


Figure 5.12: Minkowski Functionals of Activation Maximisation simulation using a CNN model that includes matter density amplitude information.

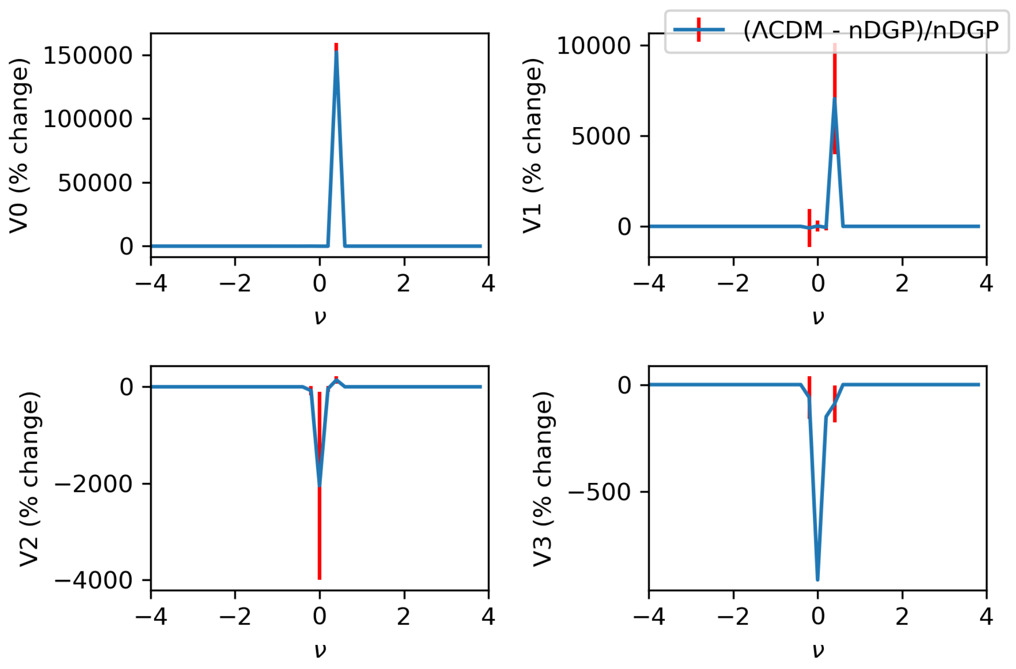


Figure 5.13: Difference in Minkowski Functionals of Activation Maximisation simulation using a CNN model that includes matter density amplitude information.

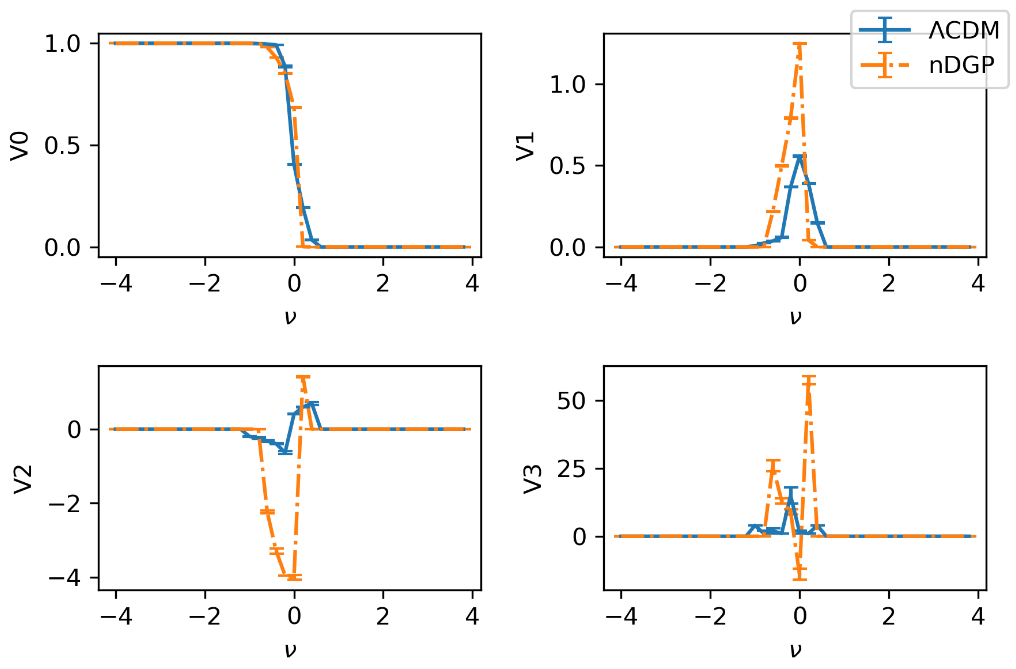


Figure 5.14: Minkowski Functionals of Activation Maximisation simulation using a CNN model that does not include matter density amplitude information.

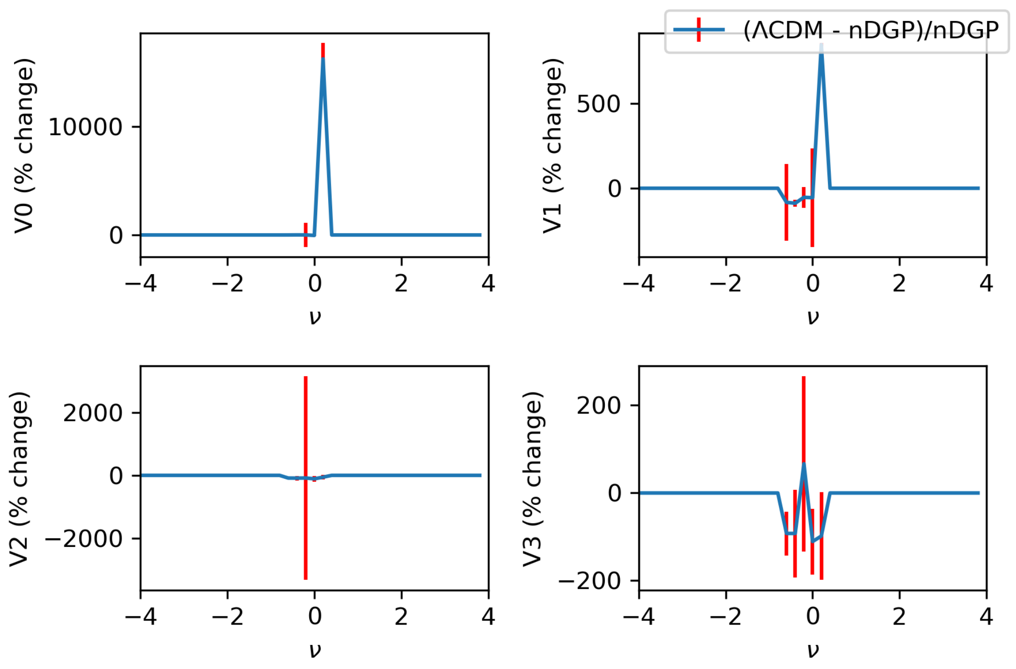


Figure 5.15: Difference in Minkowski Functionals of Activation Maximisation simulation using a CNN model that does not include matter density amplitude information.

surface, behaves similarly between the ΛCDM and nDGP AM simulations when amplitude information is included in the training. This is not the case when examining Figure 5.14, where the V_1 result between the AM simulations is quite different. The most interesting result in this investigation of the AM simulations appears when comparing the V_2 and V_3 Minkowski Functionals. It is seen in both cases (with and without amplitude information included when training), that the AM simulations that have been produced for ΛCDM and nDGP have vastly different mean curvature and Euler number characteristics. This shows that the CNN has managed to identify differences between the models of gravity with or without the inclusion of information about the amplitude of the matter density. This result indicates that the CNN is utilising differences in mean curvature or Euler number characteristics to classify the simulations.

Comparing this result to the Minkowski Functionals calculated for the input simulations the CNN used to train with provides some interesting insight. In the V_2 panel (mean curvature) in Figure 5.12, both of the AM simulations (created with CNNs trained using simulations with amplitude information) show a small depression before the threshold ν is 0, then a peak after 0. This phenomena is also seen in the V_2 input simulation Minkowski Functionals seen in Figure 4.8, and is even more prominent in the V_2 panels in Figures 5.14 where the AM simulations were created with CNNs trained using simulations without amplitude information. This indicates that the CNN had to focus more on the differences in isodensity curvature between the models of gravity. There is a clear indication, therefore, that the CNN is harnessing higher order information encapsulated in the Minkowski Functionals.

In the next section, I will discuss what can be done to improve this work and continue this area of research, both by extending methods already discussed in this work and utilising completely new ones.

5.3 Discussion

The results in this chapter show that it is possible to define decision boundaries for CNNs using techniques such as model sensitivity analysis, and even gain insight into how CNNs can correctly classify different models of gravity by investigating outputs from the Activation Maximisation technique. Regarding the model sensivity analysis, I have shown that the trained CNN is sensitive to generated Gaussian random fields drawn from power spectra with differing amplitudes and slopes. I have also been able to determine (for differing r_cH_0 values) where the CNN is no longer able to classify generated fields and how the model breaks down. In reference to the AM technique, I have been able to interrogate the model and the results indicate that even if the amplitude differences between the models of gravity are removed, the CNN is able to detect differences in features such as mean isodensity curvature (V_2 Minkowski Functional). With this being the first time either of these techniques have been used on cosmological data, there are inevitably some improvements that could be made.

5.3.1 Improvements to the training of CNNs (Chapter 4)

In Section 4.2, the accuracy of classification in all of the CNN models could be increased by spending a longer time training. This work allowed 25 epochs of training using a sample size of 10,000 and batch size of 25, meaning each CNN model viewed each simulation at least 25 times and tuned the weights and biases in the CNN 10,000 times through backpropagation. Using these settings did result in some of the CNN models being able to discriminate between the models of gravity, but the training was not exhaustive. Instead of setting the limit at 25 epochs, in the future the training could be stopped upon reaching a higher limit. Keras, the python package used to create and train the CNNs in this work does include a function (called *early stopping*) that can stop the training upon reaching a plateau in loss reduction or a plateau in accuracy maximisation. For instance, this means that when the loss is reducing during training, if the model had reached a minimum point on the error surface and could not improve, training will stop automatically before that minimum is abandoned. This would prevent that good model from being lost. There is a caveat that should be considered when using this method: if the tolerance is set too high and training is stopped too early, there may have been a better model with lower loss that the model never had the change of achieving. In real world applications, this is why machine learning models are constantly being retrained, as there are so many parameters that can change the chances of finding good models.

In conjunction with the early stopping function, it is also good practice to use the previously mentioned function to lower the learning rate when the reducing loss plateaus. This ensures that as the CNN finds a minimum on the error surface, instead of the loss bouncing around that minimum and possibly losing that model entirely, the loss can instead gradually reduce and refine the minimum. Once the minimum has been refined, the loss would plateau and the early stopping function would stop the training.

Another way to improve the models created in Section 4.2 would be to perform a hyperparameter search, which has become more possible and as such, more prevalent in recent years. Newly created functions have been published on github by deep learning enthusiasts (Autonomio, 2018) and research teams (Bergstra et al., 2013) and could be used in the future in application to this work. This could help determine which combination of kernel sizes, number of hidden layers, number of neurons in those hidden layers, loss function, and learning rate are best to create a CNN model with the lowest possible loss or highest possible accuracy.

Departing from the core changes that could made to possibly improve the CNN model, it could be argued that the results in Section 4.4 could be improved by making the Siamese network a multiclass problem. The goal of a CNN is to reduce the loss between the current result and the desired result; this leads to the probability that the CNN will learn about the most prominent feature in the data to accurately classify it. Due to the fact that I am only using two classes (ΛCDM and nDGP simulations) for the Siamese network architecture, it is possible that the network would only have to learn about one model of gravity to be able to classify both of them correctly. If the simulation it is assessing is not the model of gravity it has learned about, it must be the other. This could be rectified by making this a multi-class problem, and including other models of gravity for the CNN to learn about. Then the CNN cannot simply learn about one model of gravity to be able to classify them all correctly. This change would also help greatly in the interpretation of the model when using Siamese networks. This work did not investigate methods such as using activation maximisation in conjunction with Siamese network architectures because the code is still in development, though in the future, Siamese networks that provide AM simulations could be very interesting to analyse. Due to the fact that Siamese networks have their own branch of kernels that would only learn about one model of gravity, it could be argued that the AM simulations produced by them could be much more representative of the training simulations. This could give more accurate insights into what features the CNN has been able to learn about or model correctly.

5.3.2 Improvements to the interpretation of CNNs (Chapter 5)

Regarding the results in Section 5.1.2, the normalisation issue was a major issue in analysing the CNN models for both the Model Sensitivity Analysis and Activation Maximisation techniques. I believe this issue could be addressed by rescaling the AM simulations after they have been produced to the average mean and standard deviation of the simulations in the training set; however, there are still possible problems with this idea. For example, if the AM simulation that is supposed to represent a ΛCDM simulation was rescaled to represent the training set, would it be more or less appropriate to rescale the values to the mean and standard deviation of the whole training set, or just the ΛCDM simulations contained in the training set? If it was rescaled to the statistics of just the ΛCDM simulations, it could be argued that there is information leakage/contamination from the training set to the AM simulation, as the statistics have been gained due to the fact that the classification is already known in the training set. To try to mitigate this, each AM simulation could be rescaled to the mean and standard deviation of the whole training set (both ΛCDM simulations and nDGP simulations). While this seems like the solution to the problem, the AM simulations will still all have a condition imposed of a maximum number of particles that can be in any one voxel, a condition that does not exist in the training dataset. Even though these problems still exist, it is clear that AM can still be a useful tool when trying to interrogate CNN models and determine how they are making decisions.

In this work, I have used AM to investigate the CNNs by creating a dataset by sampling various power laws. To be able to truly investigate whether isodensity curvature is being used as a classification feature in the CNNs, one idea would be to generate data with different curvatures. This could be done by creating a set of desired curvatures in the Minkowski Functional, generating Gaussian random fields, iteratively changing the field and measuring the curvature, and using a χ^2 minimisation function to guide the fields closer to the desired curvature. This newly generated set of data with user defined curvatures could be used to probe the decision boundaries of the CNN model using curvature in conjunction with the user defined power laws, giving further insights into which features the CNN is able to identify.

5.3.3 Further improvements

In recent years, investigations into using Fast Fourier Transforms (FFTs) in conjunction with CNNs have become more popular. The most popular research relating to this union of methods is to speed up the method and reduce the memory required to perform the convolutions (Abtahi et al., 2017). This would allow CNNs to be trained faster on high-quality hardware, and allow low-quality hardware to use them at all. In relation to my investigation, I believe FFTs could be used to improve the results gained from the AM technique described in Section 5.2. When considering why the AM images are classified correctly by the CNN that produced them when applied to the MNIST example, it should be noted that the current AM algorithm creates the image by changing values in a random distribution, which is done in real space, not Fourier space. This is acceptable when considering the MNIST dataset, as two pixels side by side would have a high probability of being linked causally. For instance, when considering an image of the number 0, it is probable that there will be a high distribution of pixels that are side by side that are white (or the pixel value is a 0), and a high probability that there will be a distribution of pixels close together that have a pixel value closer to 255. This is the same when considering a distribution of dark matter particles. A much more appropriate method of producing an AM simulation when considering dark matter particle distributions would be to change the values of a random distribution in Fourier space, and then converting back to real space at the end of the AM loss minimisation process. I briefly investigated this issue when producing results for this work, and I can report that training a CNN using a training set of Fourier transformed dark matter particle simulations was possible (achieving classification accuracies over 70% using nDGP simulations with an $r_c H_0$ value of 0.50), but I have left modifying of the keras-vis AM code to be able to produce AM simulations in Fourier space for future work.

There are many possible ways to extend this work further as methods using deep learning to model and investigate our Universe mature. One such idea would be to include the particle velocity data when training the CNN, instead of simply using particle distributions as I have done. Besides the hardware requirements, this would be quite easy to implement at the technical level. In current deep learning coding frameworks, training on color images is achieved by splitting the red, green, and blue colors up into their own separate 'channels' and are provided to the CNN simultaneously for each training sample. The equivalent could be done with these dark matter particle simulations, having one channel for particle positions, and one channel for particle velocities. There would need to be tests done about whether using the same kernels and having the data put through different channels, or a different CNN architecture all together, would be the more appropriate method of approaching this problem. The different CNN architecture could be one that is somewhat similar to the Siamese network, except not providing each branch of the CNN with simulations of different classification, but with different data of the same simulation. In this architecture the Euclidean distance between the branches would not be compared as in Siamese networks, as they are of the same class, but instead the kernels would be flattened and concatenated before being passed to hidden layers.

This discussion has shown that there is a lot of work that can be done on this subject in the future; however, the results presented in this work provide a basis to improve upon. In Chapters 4 and 5 I have been able to show that a CNN with no hyperparameter optimisation can discriminate between different models of gravity to a high level of accuracy (depending on the $r_c H_0$ value in the nDGP simulations). I have been able to show that when removing matter density amplitude information from the simulation sets, arguably their strongest identifying feature, the CNN can still classify them correctly. I have also been able to show that one shot learning methods such as Siamese networks are also able to classify these simulations correctly, and should be further investigated to improve their effectiveness, as they could be useful tools for making the CNN training process quicker.

I have then gone on to introduce a new method of extracting decision boundaries from CNNs using generated data in the form of Gaussian random fields, and also utilised and analysed the outputs from new techniques (Activation Maximisation) in an effort to determine if the CNN models can learn about the amplitude of matter density, or higher order statistics such as the isodensity curvature in the models. The results indicate that the CNN was able to learn about features other than the amplitude of the matter density.

Chapter 6

Further Work and Conclusions

In this final chapter, I describe further work I have carried out during my studies, and draw together my conclusions about the thesis.

6.1 HST Proposal

Outside of the machine learning methods I have used thus far, I have had the opportunity to use more conventional methods to characterise astronomical objects. I was approached by a colleague, Tom Collett, to perform a redshift estimation using DES photometric data for a strong lensing candidate. The reason for this was to create a proposal to acquire Hubble Space Telescope (HST) imaging data to perform more accurate mass calculations than were possible with the photometric DES data.

To do this I used a program called HyperZ (Bolzonella et al., 2000), a template fitting code that works with photometric data to calculate the redshift and physical galaxy properties (such as galaxy age, mass, metallicity, and star-formation history). It does this by iteratively fitting the observed photometric data to a set of reference (templates) or synthetic spectra, and calculating the χ^2 value (a measure of the difference between the observed fluxes and template fluxes, then the best χ^2 is chosen.

I used two template sets for this investigation, the first were the star-forming templates as described in Maraston et al. (2006). These include 32 sets of various star-formation histories, four different metallicities, 221 ages in the range of 1Myr - t(z) (restricted or unrestricted age), and dust in various amounts. The second set were the purely passive templates described in Maraston et al. (2009), where there was single-burst of star-forming, and a spread of metallicities. After fitting the data, the redshift for the candidate with the best χ^2 was z = 2.6 $\pm$ 0.4.

The redshift for the lensed galaxy was calculated to be at z = 2.39, within the errors of my initial calculation, from re-reduced spectral data (shown in Figure 6.1) before Collett et al. (2017) was submitted. The mass enclosed within the 14 arc second Einstein ring is calculated to be $10^{14.2}$ solar masses. A full light profile reconstruction of the lensed images was performed to infer the parameters of the mass distribution. This lensed system is of interest because it requires either a very shallow dark matter profile, or the presence of two merging dark matter components. The initial redshift that I calculated helped to confirm the system and constrain the model. I am an author on the paper and the proposal to acquire the HST data for the candidate was accepted.

6.2 Conclusions

Research into the area of machine learning has become prevalent in recent years, and it is important that research fields such as astronomy and cosmology rapidly benefit from new modelling methods. Considering the successes machine learning has had in other disciplines such as medicine, it is right that these new methods are investigated and implemented along with current well-established methods.

In relation to the results presented in Chapter 3, it can be seen that while previous methods of classification perform very well, machine learning methods (especially feature driven and tuned models) can outperform them. In this work I have been able to develop a pipeline that offers in-depth analysis of

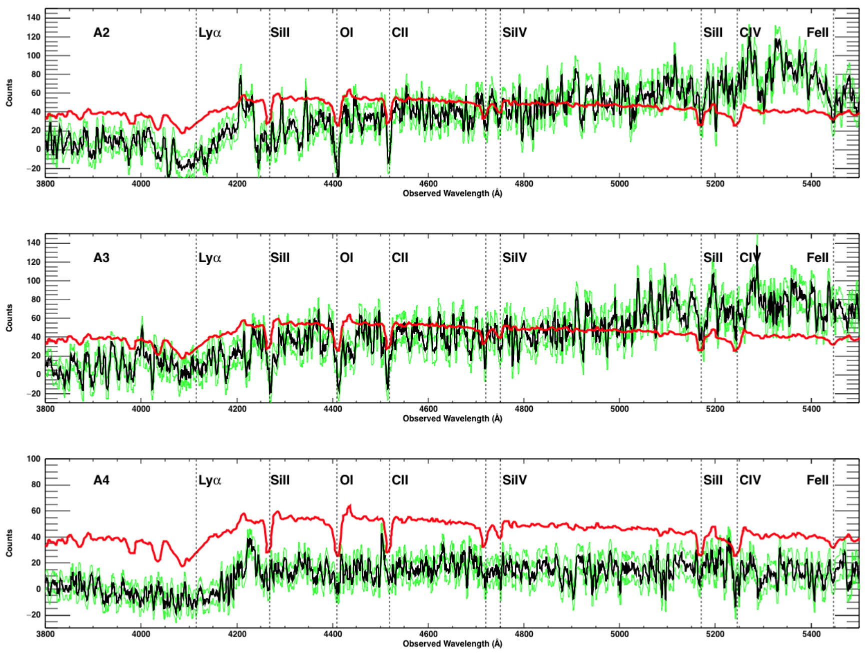


Figure 6.1: Spectra of gravitationally lensed galaxy from Collett et al. (2017). The absorption features from the galaxy at z = 2.39 are indicated by the vertical black dotted lines. The green overlaid spectra shows the $\pm1\sigma$ errors from spectral extraction. The red overlaid spectra is a template of Shapley et al. (2003) shifted by the measured redshift.

machine learning models using *treeinterpreter*. This package allows features used in a random forest to not only be ranked against one another, but shows what particular value of the features were most important to the classification. I have also applied the MINT feature selection method, which utilises all of the data (training and test) to select a user defined number of features that are the least correlated with one another, and simultaneously correlated most with the classifications. The results show that the pipeline improves on the `frames` object classification accuracy by 1.0%, which is $\approx 33\%$ improvement in the rate of misclassification. Indeed, there are several reasons for considering methods such as those outlined in Chapter 2.

Firstly, it has been shown that tree-based methods offer at least some level of interpretability. Machine learning models and feature selection methods such as MINT may choose to use features that do not seem to be obvious, so figuring out how and why the model is working has been difficult. With new codes such as *treeinterpreter*, I have shown that the models can be analysed in such a way as to provide insight into which features are important to the problem and why. Using such methods, it is possible for the machine to pick out relations/correlations that have been previously missed.

Secondly, a higher degree of classification accuracy can be achieved - one closer to that obtained by fitting spectra. The machine learning algorithms also output probabilities for each classification, allowing users to single out objects which are a problem for the machine learning model.

Thirdly, the machine learning method of classification is computationally almost as quick as the `frames` method. For future surveys, speed of data processing will become a very important problem. Our method could be included in the pipeline of a new survey, where a standard training set is created and given to the pipeline (from science verification data for example), and the model could be continuously improved as new data is observed.

This work is an example of how new methods like *treeinterpreter* and MINT are useful in understanding the relationship between data and the performance of machine learning models. This analysis would have to be repeated for new datasets from different astronomical surveys because the results presented here are not trivially transferable. In the future, as well as being incorporated into survey data processing pipelines, these methods could be applied to other problems in astronomy such as predicting redshifts or the physical properties of galaxies, and offer new insights into how and why machine learning algorithms make their decisions.

In relation to the results presented in Chapters 4 and 5, I have developed an approach to discriminating between cosmologies and gravities by producing dark matter particle simulations with different cosmological models, compiling them into augmented datasets, and feeding them to machine learning methods for classifying image data such as CNNs. I have shown that these CNNs are able to distinguish between dark matter particles simulations created using different theories of gravity to a high degree of accuracy. I have also demonstrated that this is the case even when removing the most obvious discriminator by normalising the power spectrum amplitudes. Further to this, I have demonstrated that one-shot learning methods (CNNs of a different architecture) are able to do the same. Here I used Siamese networks in cosmology for the first time.

Following on from these novel results, I have been able to probe the model and understand how it works, demystifying how CNNs are able to tell the difference between these simulations. This has been done by extracting decision boundaries and parameterising where the model breaks down using generated datasets with known features (Gaussian random fields). Finally, I have been able to manipulate the CNN into producing its own representations of different theories of gravity in order to study and determine what it has been able to learn, showing that CNNs are able to learn about higher order statistics present in the simulations such as isodensity curvature. I have discussed how to improve the results for the existing methods, and outlined new directions that would be beneficial to the field such as the further development of GANs. This would drastically reduce the required hardware and time to produce accurate dark matter particle simulations.

This work lays the foundation of using CNNs to study theories of gravity in two major ways. Firstly, given that CNNs are able to learn about the different theories, we can study successful CNN models to discover features in dark matter particle simulations that may have been previously overlooked - analogous to a fresh set of eyes looking at a problem. Secondly, if we consider our simulations to be true representations of the universe, we could project the simulations to create convergence maps and train CNNs with them. We could then compare the generated convergence maps with those like the one created from weak lensing using DES Y1 data (Figure 1.4) and parameterise the differences. Finally, we could input the real convergence map to the trained CNN and produce predictions about our Universe. The quality of the predictions would increase as the simulations become larger and more complex, for instance, with the inclusion of baryonic matter that would affect predictions on small scales. Using extensions to the ideas presented in this work it may be possible in the future for CNNs to answer the question - 'Does our Universe follow Einstein's theory of gravity, or a different one?'.

Appendix A

First Appendix

A.1 Casjobs SQL Query

This is the SQL Query submitted to Casjobs to obtain all the values required to calculate the whole sample used in this work.

```
select s.specObjID, s.class as spec_class, q.objid,
q.dered_u,q.dered_g,q.dered_r,q.dered_i, q.dered_z,
q.modelMagErr_u,q.modelMagErr_g, q.modelMagErr_r,
q.modelMagErr_i,q.modelMagErr_z,
q.extinction_u,q.extinction_g,q.extinction_r,
q.extinction_i,q.extinction_z,
q.cModelMag_u,q.cModelMagErr_u, q.cModelMag_g,q.cModelMagErr_g,
q.cModelMag_r,q.cModelMagErr_r, q.cModelMag_i,q.cModelMagErr_i,
q.cModelMag_z,q.cModelMagErr_z,
q.psfMag_u,q.psfMagErr_u, q.psfMag_g,q.psfMagErr_g,
q.psfMag_r,q.psfMagErr_r, q.psfMag_i,q.psfMagErr_i,
q.psfMag_z,q.psfMagErr_z,
q.fiberMag_u,q.fiberMagErr_u, q.fiberMag_g,q.fiberMagErr_g,
q.fiberMag_r,q.fiberMagErr_r, q.fiberMag_i,q.fiberMagErr_i,
q.fiberMag_z,q.fiberMagErr_z,
q.expRad_u, q.expRad_g, q.expRad_r, q.expRad_i, q.expRad_z,
q.clean, s.zWarning

into mydb.specPhotoDR12 from SpecObjAll as s
join photoObjAll as q on s.bestobjid=q.objid
left outer join Photoz as p on s.bestobjid=p.objid
```

References

Aarseth, S. J. (1963). Dynamical evolution of clusters of galaxies, I. *MNRAS*, 126:223.

Abadi, M., Agarwal, A., Barham, P., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abazajian, K., Adelman-McCarthy, J. K., Agüeros, M. A., et al. (2004). The Second Data Release of the Sloan Digital Sky Survey. *AJ*, 128:502–512, astro-ph/0403325.

Abbott, T. M. C., Abdalla, F. B., Allam, S., et al. (2018). The Dark Energy Survey Data Release 1. *ArXiv e-prints*, 1801.03181.

Abtahi, T., Kulkarni, A., and Mohsenin, T. (2017). Accelerating convolutional neural network with fft on tiny cores. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4.

Alam, S., Albareti, F. D., Allende Prieto, C., et al. (2015). The Eleventh and Twelfth Data Releases of the Sloan Digital Sky Survey: Final Data from SDSS-III. *ApJS*, 219:12, 1501.00963.

Alam, S., Ata, M., Bailey, S., et al. (2017). The clustering of galaxies in the completed SDSS-III Baryon Oscillation Spectroscopic Survey: cosmological analysis of the DR12 galaxy sample. *MNRAS*, 470:2617–2652, 1607.03155.

Alpher, R. A. and Herman, R. (1948a). Evolution of the Universe. *Nature*, 162:774–775.

Alpher, R. A. and Herman, R. C. (1948b). On the Relative Abundance of the Elements. *Physical Review*, 74:1737–1742.

Andrew. Hall, M. (2000). Correlation-based feature selection for machine learning. 19.

Ata, M., Baumgarten, F., Bautista, J., et al. (2018). The clustering of the SDSS-IV extended Baryon Oscillation Spectroscopic Survey DR14 quasar sample: first measurement of baryon acoustic oscillations between redshift 0.8 and 2.2. *MNRAS*, 473:4773–4794, 1705.06373.

Autonomio (2018). Hyperparameter optimization for keras models. `https://github.com/autonomio/talos`.

Baum, W. A. (1962). Photoelectric Magnitudes and Red-Shifts. In McVittie, G. C., editor, *Problems of Extra-Galactic Research*, volume 15 of *IAU Symposium*, page 390.

Bergstra, J., Yamins, D., and Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In Dasgupta, S. and McAllester, D., editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 115–123, Atlanta, Georgia, USA. PMLR.

Betoule, M., Kessler, R., Guy, J., et al. (2014). Improved cosmological constraints from a joint analysis of the SDSS-II and SNLS supernova samples. *A&A*, 568:A22, 1401.4064.

Beynon, E., Bacon, D. J., and Koyama, K. (2010). Weak lensing predictions for modified gravities at non-linear scales. *MNRAS*, 403:353–362, 0910.1480.

Bolton, A. S., Schlegel, D. J., Aubourg, É., et al. (2012). Spectral Classification and Redshift Measurement for the SDSS-III Baryon Oscillation Spectroscopic Survey. *AJ*, 144:144, 1207.7326.

Bolzonella, M., Miralles, J.-M., and Pelló, R. (2000). Photometric redshifts based on standard SED fitting procedures. *A&A*, 363:476–492, astro-ph/0003380.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. (1984). *Classification and Regression Trees (Wadsworth Statistics/Probability)*. Chapman and Hall/CRC.

Bridle, J. S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In Soulié, F. F. and Hérault, J., editors, *Neurocomputing*, pages 227–236, Berlin, Heidelberg. Springer Berlin Heidelberg.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. (1993). Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*, NIPS'93, pages 737–744, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Cabayol, L., Sevilla-Noarbe, I., Fernández, E., et al. (2018). The PAU Survey: star-galaxy classification with multi narrow-band data. *ArXiv e-prints*, 1806.08545.

Cappellari, M. and Emsellem, E. (2004). Parametric Recovery of Line-of-Sight Velocity Distributions from Absorption-Line Spectra of Galaxies via Penalized Likelihood. *PASP*, 116:138–147, astro-ph/0312201.

Carrasco Kind, M. and Brunner, R. J. (2013). TPZ: photometric redshift PDFs and ancillary information by using prediction trees and random forests. *MNRAS*, 432:1483–1501, 1303.7269.

Carroll, S. M. (2001). The Cosmological Constant. *Living Reviews in Relativity*, 4:1, astro-ph/0004075.

Carroll, S. M. (2004). *Spacetime and geometry: An introduction to general relativity.*

Chang, C., Pujol, A., Mawdsley, B., et al. (2018). Dark Energy Survey Year 1 results: curved-sky weak lensing mass map. *MNRAS*, 475:3165–3190, 1708.01535.

Chollet, F. et al. (2015). Keras. `https://github.com/fchollet/keras`.

Chopra, S., Hadsell, R., and LeCun, Y. (2005). Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 539–546, Washington, DC, USA. IEEE Computer Society.

Cireşan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column Deep Neural Networks for Image Classification. *ArXiv e-prints*, 1202.2745.

Clifton, T., Ferreira, P. G., Padilla, A., and Skordis, C. (2012). Modified gravity and cosmology. *Phys. Rep.*, 513:1–189, 1106.2476.

Clowe, D., Gonzalez, A., and Markevitch, M. (2004). Weak-Lensing Mass Reconstruction of the Interacting Cluster 1E 0657-558: Direct Evidence for the Existence of Dark Matter. *ApJ*, 604:596–603, astro-ph/0312273.

Collett, T. E., Buckley-Geer, E., Lin, H., et al. (2017). Core or Cusps: The Central Dark Matter Profile of a Strong Lensing Cluster with a Bright Central Image at Redshift 1. *ApJ*, 843:148, 1703.08410.

Collister, A. A. and Lahav, O. (2004). ANNz: Estimating Photometric Redshifts Using Artificial Neural Networks. *PASP*, 116:345–351, astro-ph/0311058.

Conley, A., Guy, J., Sullivan, M., et al. (2011). Supernova Constraints and Systematic Uncertainties from the First Three Years of the Supernova Legacy Survey. *ApJS*, 192:1, 1104.1443.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

da Silva, L. A. L. (1993). The classification of supernovae. *Ap&SS*, 202:215–236.

Dawson, K. S., Kneib, J.-P., Percival, W. J., et al. (2016). The SDSS-IV Extended Baryon Oscillation Spectroscopic Survey: Overview and Early Data. *AJ*, 151:44, 1508.04473.

Dawson, K. S., Schlegel, D. J., Ahn, C. P., et al. (2013). The Baryon Oscillation Spectroscopic Survey of SDSS-III. *AJ*, 145:10, 1208.0022.

de Boer, P.-T., Kroese, D., Mannor, S., and Rubinstein, R. (2005). A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67. Imported from research group DACS (ID number 277).

DES Collaboration, Abbott, T. M. C., Abdalla, F. B., et al. (2017). Dark Energy Survey Year 1 Results: Cosmological Constraints from Galaxy Clustering and Weak Lensing. *ArXiv e-prints*, 1708.01530.

Dhawan, S., Jha, S. W., and Leibundgut, B. (2018). Measuring the Hubble constant with Type Ia supernovae as near-infrared standard candles. *A&A*, 609:A72, 1707.00715.

Ding, C. and Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3(2):185–205.

Doi, M., Tanaka, M., Fukugita, M., et al. (2010). Photometric Response Functions of the Sloan Digital Sky Survey Imager. *AJ*, 139:1628–1648, 1002.3701.

Doroshkevich, A. G., Zel'dovich, Y. B., and Syunyaev, R. A. (1978). Fluctuations of the microwave background radiation in the adiabatic and entropic theories of galaxy formation. *Soviet Ast.*, 22:523–528.

Doyle, P. (1973). The use of automatic interaction detector and similar search procedures. *Operational Research Quarterly (1970-1977)*, 24(3):465–467.

Dvali, G., Gabadadze, G., and Porrati, M. (2000). 4D gravity on a brane in 5D Minkowski space. *Physics Letters B*, 485:208–214.

Dyson, F. W., Eddington, A. S., and Davidson, C. (1920). A Determination of the Deflection of Light by the Sun's Gravitational Field, from Observations Made at the Total Eclipse of May 29, 1919. *Philosophical Transactions of the Royal Society of London Series A*, 220:291–333.

Einhorn, H. J. (1972). Alchemy in the behavioral sciences. *The Public Opinion Quarterly*, 36(3):367–378.

Einstein, A. (1916). The Foundation of the General Theory of Relativity. *Annalen Phys.*, 49(7):769–822. [,65(1916)].

Eisenstein, D. J., Weinberg, D. H., Agol, E., et al. (2011). SDSS-III: Massive Spectroscopic Surveys of the Distant Universe, the Milky Way, and Extra-Solar Planetary Systems. *AJ*, 142:72, 1101.1529.

Eisenstein, D. J., Zehavi, I., Hogg, D. W., et al. (2005). Detection of the Baryon Acoustic Peak in the Large-Scale Correlation Function of SDSS Luminous Red Galaxies. *ApJ*, 633:560–574, astro-ph/0501171.

Fang, W., Wang, S., Hu, W., et al. (2008). Challenges to the DGP model from horizon-scale growth and geometry. *Phys. Rev. D*, 78(10):103509, 0808.2208.

Fei-Fei, L., Fergus, R., and Perona, P. (2006). One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1469-1809.1936.tb02137.x.

Fixsen, D. J. (2009). The Temperature of the Cosmic Microwave Background. *ApJ*, 707:916–920, 0911.1955.

Frenk, C. S., White, S. D. M., Efstathiou, G., and Davis, M. (1985). Cold dark matter, the structure of galactic haloes and the origin of the Hubble sequence. *Nature*, 317:595–597.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139.

Friedman, J. H. (1999). Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38:367–378.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Ann. Statist.*, 29(5):1189–1232.

Friedmann, A. (1922). Über die Krümmung des Raumes. *Zeitschrift fur Physik*, 10:377–386.

Friedmann, A. (1924). Über die Möglichkeit einer Welt mit konstanter negativer Krümmung des Raumes. *Zeitschrift fur Physik*, 21:326–332.

Gamow, G. (1948a). The Evolution of the Universe. *Nature*, 162:680–682.

Gamow, G. (1948b). The Origin of Elements and the Separation of Galaxies. *Physical Review*, 74:505–506.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1):3–42.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., et al. (2014). Generative Adversarial Networks. *ArXiv e-prints*, 1406.2661.

Gunn, J. E., Carr, M., Rockosi, C., et al. (1998). The Sloan Digital Sky Survey Photometric Camera. *AJ*, 116:3040–3081, astro-ph/9809085.

Gunn, J. E., Siegmund, W. A., Mannery, E. J., et al. (2006). The 2.5 m Telescope of the Sloan Digital Sky Survey. *AJ*, 131:2332–2359, astro-ph/0602326.

Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

Hahnloser, R. H. R., Sarpeshkar, R., Mahowald, M. A., Douglas, R. J., and Seung, H. S. (2000). Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, 405:947–951.

Hahnloser, R. H. R., Seung, H. S., and Slotine, J.-J. (2003). Permitted and forbidden sets in symmetric threshold-linear networks. *Neural Comput.*, 15(3):621–638.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*. Springer.

He, D., Rish, I., Haws, D., et al. (2013). MINT: Mutual Information based Transductive Feature Selection for Genetic Trait Prediction. *ArXiv e-prints*, 1310.1659.

Heckerman, D. (1995). A tutorial on learning bayesian networks.

Hénon, M. (1964). L'évolution initiale d'un amas sphérique. *Annales d'Astrophysique*, 27:83.

Hillebrandt, W. and Niemeyer, J. C. (2000). Type IA Supernova Explosion Models. *ARA&A*, 38:191–230.

Hockney, R. W. and Eastwood, J. W. (1988). *Computer simulation using particles.*

Holmberg, E. (1941). On the Clustering Tendencies among the Nebulae. II. a Study of Encounters Between Laboratory Models of Stellar Systems by a New Integration Procedure. *ApJ*, 94:385.

Howlett, C. (2016). Modelling and measuring cosmological structure growth.

Howlett, C., Manera, M., and Percival, W. J. (2015). L-PICOLA: A parallel code for fast dark matter simulation. *Astronomy and Computing*, 12:109–126, 1506.03737.

Hoyle, B., Rau, M. M., Zitlau, R., Seitz, S., and Weller, J. (2015). Feature importance for machine learning redshifts applied to sdss galaxies. *Monthly Notices of the Royal Astronomical Society*, 449(2):1275–1283, http://mnras.oxfordjournals.org/content/449/2/1275.full.pdf+html.

Hu, W., Sugiyama, N., and Silk, J. (1996). The Physics of Microwave Background Anisotropies. *ArXiv Astrophysics e-prints*, astro-ph/9604166.

Hu, W. and Tegmark, M. (1999). Weak Lensing: Prospects for Measuring Cosmological Parameters. *ApJ*, 514:L65–L68, astro-ph/9811168.

Hubble, E. (1929). A Relation between Distance and Radial Velocity among Extra-Galactic Nebulae. *Proceedings of the National Academy of Science*, 15:168–173.

Hubble, E. P. (1926). Extragalactic nebulae. *ApJ*, 64.

Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ArXiv e-prints*, 1502.03167.

Jarosik, N., Bennett, C. L., Dunkley, J., et al. (2011). Seven-year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Sky Maps, Systematic Errors, and Basic Results. *ApJS*, 192:14, 1001.4744.

Kamionkowski, M., Spergel, D. N., and Sugiyama, N. (1994). Small-scale cosmic microwave background anisotropies as probe of the geometry of the universe. *ApJ*, 426:57–60, astro-ph/9401003.

Kiefer, J. and Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. *Ann. Math. Statist.*, 23(3):462–466.

Kingma, D. P. and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *ArXiv e-prints*, 1412.6980.

Kirkpatrick, J. A., Schlegel, D. J., Ross, N. P., et al. (2011). A Simple Likelihood Method for Quasar Target Selection. *ApJ*, 743:125, 1104.4995.

Koch, G. R. (2015). Siamese neural networks for one-shot image recognition.

Komatsu, E., Smith, K. M., Dunkley, J., et al. (2011). Seven-year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Cosmological Interpretation. *ApJS*, 192:18, 1001.4538.

Kotikalapudi, R. and contributors (2017). keras-vis. `https://github.com/raghakot/keras-vis`.

Koyama, K. (2007). TOPICAL REVIEW: Ghosts in the self-accelerating universe. *Classical and Quantum Gravity*, 24:R231–R253, 0709.2399.

Koyama, K. and Maartens, R. (2006). Structure formation in the Dvali Gabadadze Porrati cosmological model. *J. Cosmology Astropart. Phys.*, 1:016, astro-ph/0511634.

Lake, B., Salakhutdinov, R., Gross, J., and Tenenbaum, J. (2011). One shot learning of simple visual concepts.

Le Fevre, O., Bijaoui, A., Mathez, G., Picat, J. P., and Lelievre, G. (1986). Electronographic BV photometry of three distant clusters of galaxies. I - Observations and reduction techniques. *A&A*, 154:92–99.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. 86:2278 – 2324.

Lemaître, G. (1927). Un Univers homogène de masse constante et de rayon croissant rendant compte de la vitesse radiale des nébuleuses extra-galactiques. *Annales de la Société Scientifique de Bruxelles*, 47:49–59.

Lesgourgues, J. (2011). The Cosmic Linear Anisotropy Solving System (CLASS) I: Overview. *ArXiv e-prints*, 1104.2932.

Lewis, A., Challinor, A., and Lasenby, A. (2000). Efficient Computation of Cosmic Microwave Background Anisotropies in Closed Friedmann-Robertson-Walker Models. *ApJ*, 538:473–476, astro-ph/9911177.

Li, N. and Thakar, A. R. (2008). CasJobs and MyDB: A Batch Query Workbench. *Computing in Science and Engineering*, 10:18–29.

Liddle, A. (2003). *An introduction to modern cosmology; 2nd ed.* Wiley, Chichester.

Lieb, E. H. and Yau, H.-T. (1987). A rigorous examination of the Chandrasekhar theory of stellar collapse. *ApJ*, 323:140–144.

Light, R. J. and Margolin, B. H. (1971). An analysis of variance for categorical data. *Journal of the American Statistical Association*, 66(335):534–544.

Loh, W.-Y. (2014). Fifty years of classification and regression trees. *International Statistical Review*, 82(3):329–348, https://onlinelibrary.wiley.com/doi/pdf/10.1111/insr.12016.

Lovelock, D. (1971). The Einstein Tensor and Its Generalizations. *Journal of Mathematical Physics*, 12:498–501.

Lovelock, D. (1972). The Four-Dimensionality of Space and the Einstein Tensor. *Journal of Mathematical Physics*, 13:874–876.

Maas, A. L. (2013). Rectifier nonlinearities improve neural network acoustic models.

Mahendran, A. and Vedaldi, A. (2015). Visualizing Deep Convolutional Neural Networks Using Natural Pre-Images. *ArXiv e-prints*, 1512.02017.

Maraston, C., Daddi, E., Renzini, A., et al. (2006). Evidence for TP-AGB Stars in High-Redshift Galaxies, and Their Effect on Deriving Stellar Population Parameters. *ApJ*, 652:85–96, astro-ph/0604530.

Maraston, C., Strömbäck, G., Thomas, D., Wake, D. A., and Nichol, R. C. (2009). Modelling the colour evolution of luminous red galaxies - improvements with empirical stellar spectra. *MNRAS*, 394:L107–L111, 0809.1867.

Masters, K. L., Maraston, C., Nichol, R. C., et al. (2011). The morphology of galaxies in the Baryon Oscillation Spectroscopic Survey. *MNRAS*, 418:1055–1070, 1106.3331.

Melott, A. L. (1990). The topology of large-scale structure in the universe. *Phys. Rep.*, 193:1–39.

Messenger, R. and Mandell, L. (1972). A modal search technique for predictibe nominal scale multivariate analys. 67.

Morgan, J. N. and Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American Statistical Association*, 58(302):415–434, https://amstat.tandfonline.com/doi/pdf/10.1080/01621459.1963.10500855.

Morice-Atkinson, X., Hoyle, B., and Bacon, D. (2018). Learning from the machine: interpreting machine learning algorithms for point- and extended-source classification. *MNRAS*, 1712.03970.

Mosteller, F. and Tukey, J. W. (1968). Data analysis including statistics. In Lindzey, G. and Aronson, E., editors, *The Handbook of Social Psychology*, volume 2, chapter 10, pages 80–203. Second edition. Five volumes.

Nielsen, M. A. (2015). Neural networks and deep learning.

Oesch, P. A., Brammer, G., van Dokkum, P. G., et al. (2016). A Remarkably Luminous Galaxy at z=11.1 Measured with Hubble Space Telescope Grism Spectroscopy. *ApJ*, 819:129, 1603.00461.

Parker, L. (1968). Particle creation in expanding universes. *Phys. Rev. Lett.*, 21:562–564.

Parker, L. (1969). Quantized fields and particle creation in expanding universes. i. *Phys. Rev.*, 183:1057–1068.

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peebles, P. J. E. (1970). Structure of the Coma Cluster of Galaxies. *AJ*, 75:13.

Peebles, P. J. E. (1983). The sequence of cosmogony and the nature of primeval departures from homogeneity. *ApJ*, 274:1–6.

Peng, H., Long, F., and Ding, C. (2005). Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238.

Perlmutter, S., Aldering, G., Goldhaber, G., et al. (1999). Measurements of Ω and Λ from 42 High-Redshift Supernovae. *ApJ*, 517:565–586, astro-ph/9812133.

Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. (2016). Planck 2015 results. XX. Constraints on inflation. *A&A*, 594:A20, 1502.02114.

Planck Collaboration, Aghanim, N., Akrami, Y., et al. (2018a). Planck 2018 results. VI. Cosmological parameters. *ArXiv e-prints*, 1807.06209.

Planck Collaboration, Akrami, Y., Arroja, F., et al. (2018b). Planck 2018 results. X. Constraints on inflation. *ArXiv e-prints*, 1807.06211.

Puschell, J. J., Owen, F. N., and Laing, R. A. (1982). Near-infrared photometry of distant radio galaxies - Spectral flux distributions and redshift estimates. *ApJ*, 257:L57–L61.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Quinn, P. J., Salmon, J. K., and Zurek, W. H. (1986). Primordial density fluctuations and the structure of galactic haloes. *Nature*, 322:329–335.

Quinn, T., Katz, N., Stadel, J., and Lake, G. (1997). Time stepping N-body simulations. *ArXiv Astrophysics e-prints*, astro-ph/9710043.

Ravanbakhsh, S., Oliva, J., Fromenteau, S., et al. (2017). Estimating Cosmological Parameters from the Dark Matter Distribution. *ArXiv e-prints*, 1711.02033.

Riess, A. G., Casertano, S., Yuan, W., et al. (2018). New Parallaxes of Galactic Cepheids from Spatially Scanning the Hubble Space Telescope: Implications for the Hubble Constant. *ApJ*, 855:136, 1801.01120.

Riess, A. G., Filippenko, A. V., Challis, P., et al. (1998). Observational Evidence from Supernovae for an Accelerating Universe and a Cosmological Constant. *AJ*, 116:1009–1038, astro-ph/9805201.

Riess, A. G., Strolger, L.-G., Casertano, S., et al. (2007). New Hubble Space Telescope Discoveries of Type Ia Supernovae at z = 1: Narrowing Constraints on the Early Behavior of Dark Energy. *ApJ*, 659:98–121, astro-ph/0611572.

Riesz, F. (1910). Untersuchungen über systeme integrierbarer funktionen. *Mathematische Annalen*, 69(4):449–497.

Robbins, H. and Sutton, M. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407.

Robertson, H. P. (1935). Kinematics and World-Structure. *ApJ*, 82:284.

Rodriguez, A. C., Kacprzak, T., Lucchi, A., et al. (2018). Fast Cosmic Web Simulations with Generative Adversarial Networks. *ArXiv e-prints*, 1801.09070.

Rosenblatt, F. (1957). The perceptron: A perceiving and recognizing automaton. Report 85-460-1, Project PARA, Cornell Aeronautical Laboratory, Ithaca, New York. Rosenblatt, F.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1988). Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, pages 696–699. MIT Press, Cambridge, MA, USA.

Saabas, A. (2015). treeinterpreter. `https://github.com/andosa/treeinterpreter`.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM J. Res. Dev.*, 3(3):210–229.

Sarzi, M., Falcón-Barroso, J., Davies, R. L., et al. (2006). The SAURON project - V. Integral-field emission-line kinematics of 48 elliptical and lenticular galaxies. *MNRAS*, 366:1151–1200, astro-ph/0511307.

Schmalzing, J., Kerscher, M., and Buchert, T. (1996). Minkowski Functionals in Cosmology. In Bonometto, S., Primack, J. R., and Provenzale, A., editors, *Dark Matter in the Universe*, page 281. astro-ph/9508154.

Schmidt, F. (2009). Cosmological simulations of normal-branch braneworld gravity. *Phys. Rev. D*, 80(12):123003, 0910.0235.

Scoccimarro, R. (1998). Transients from initial conditions: a perturbative analysis. *MNRAS*, 299:1097–1118, astro-ph/9711187.

Sevilla-Noarbe, I., Hoyle, B., Marchã, M. J., et al. (2018). Star-galaxy classification in the Dark Energy Survey Y1 dataset. *ArXiv e-prints*, 1805.02427.

Shapley, A. E., Steidel, C. C., Pettini, M., and Adelberger, K. L. (2003). Rest-Frame Ultraviolet Spectra of z˜3 Lyman Break Galaxies. *ApJ*, 588:65–89, astro-ph/0301230.

Silk, J. (1968). Cosmic Black-Body Radiation and Galaxy Formation. *ApJ*, 151:459.

Smee, S. A., Gunn, J. E., Uomoto, A., et al. (2013). The Multi-object, Fiber-fed Spectrographs for the Sloan Digital Sky Survey and the Baryon Oscillation Spectroscopic Survey. *AJ*, 146:32, 1208.2233.

Smith, J. A., Tucker, D. L., Kent, S., et al. (2002). The u'g'r'i'z' Standard-Star System. *AJ*, 123:2121–2144, astro-ph/0201143.

Soumagnac, M. T., Abdalla, F. B., Lahav, O., et al. (2015). Star/galaxy separation at faint magnitudes: application to a simulated Dark Energy Survey. *MNRAS*, 450:666–680.

Springel, V. (2005). The cosmological simulation code GADGET-2. *MNRAS*, 364:1105–1134, astro-ph/0505010.

Stoughton, C., Lupton, R. H., Bernardi, M., et al. (2002). Sloan Digital Sky Survey: Early Data Release. *AJ*, 123:485–548.

Szalay, A. S., Gray, J., Thakar, A. R., et al. (2002). The SDSS SkyServer: Public Access to the Sloan Digital Sky Server Data. *eprint arXiv:cs/0202013*, cs/0202013.

Tassev, S., Zaldarriaga, M., and Eisenstein, D. J. (2013). Solving large scale structure in ten easy steps with COLA. *J. Cosmology Astropart. Phys.*, 6:036, 1301.0322.

Tormen, G., Bouchet, F. R., and White, S. D. M. (1997). The structure and dynamical evolution of dark matter haloes. *MNRAS*, 286:865–884, astro-ph/9603132.

van Albada, G. B. (1961). Evolution of clusters of galaxies under gravitational forces. *AJ*, 66:590.

Vince, A. (2002). A framework for the greedy algorithm. *Discrete Appl. Math.*, 121(1-3):247–260.

von Hoerner, S. (1963). Die numerische Integration des n-Körper-Problems für Sternhaufen, II. *Z. Astrophys.*, 57.

Walker, A. G. (1935). On the formal comparison of Milne's kinematical system with the systems of general relativity. *MNRAS*, 95:263–269.

Wan, L., Zeiler, M., Zhang, S., Cun, Y. L., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In Dasgupta, S. and McAllester, D., editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1058–1066, Atlanta, Georgia, USA. PMLR.

Weinberg, D. H., Gott, III, J. R., and Melott, A. L. (1987). The topology of large-scale structure. I - Topology and the random phase hypothesis. *ApJ*, 321:2–27.

Wilkinson, D. M., Maraston, C., Goddard, D., Thomas, D., and Parikh, T. (2017). FIREFLY (Fitting IteRativEly For Likelihood analYsis): a full spectral fitting code. *MNRAS*, 472:4297–4326, 1711.00865.

Winther, H., Koyama, K., Manera, M., Wright, B., and Zhao, G.-B. (2017). Cola with scale-dependent growth: applications to screened modified gravity models. *Journal of Cosmology and Astroparticle Physics*. 31 pages, 11 figures. The code can be found at https://github.com/HAWinther/MG-PICOLA-PUBLIC. 12 months embargo.

Yaeger, L., Lyon, R., and Webb, B. (1996). Effective training of a neural network character classifier for word recognition. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS'96, pages 807–813, Cambridge, MA, USA. MIT Press.

Yèche, C., Petitjean, P., Rich, J., et al. (2010). Artificial neural networks for quasar selection and photometric redshift determination. *A&A*, 523.

York, D. G., Adelman, J., Anderson, Jr., J. E., et al. (2000). The Sloan Digital Sky Survey: Technical Summary. *AJ*, 120:1579–1587, astro-ph/0006396.

Zhao, G.-B., Li, B., and Koyama, K. (2011). N-body simulations for f(R) gravity using a self-adaptive particle-mesh code. *Phys. Rev. D*, 83:044007, 1011.1257.

Zou, H., Zhu, J., Rosset, S., and Hastie, T. (2009). Multi-class adaboost. *Statistics and its Interface*, 2:349–360.