Galaxy Classification in the Perseus Cluster With a Convolutional Neural Network

A Project

Presented to

The Faculty of the Department of Physics and Astronomy at

San José State University, CA, USA

In Partial Fulfillment

of the Requirements for the Degree

Masters of Science

by

Jason Pruitt

May, 2024

The Designated Research Project Committee Approves the Project Titled

Galaxy Classification in the Perseus Cluster With a Convolutional Neural Network

by

Jason Pruitt

APPROVED FOR THE DEPARTMENT OF PHYSICS AND ASTRONOMY

SAN JOSÉ STATE UNIVERSITY

May 2024

Dr. Aaron Romanowsky	Department of Physics and Astronomy
Dr. Ehsan Khatami	Department of Physics and Astronomy
Dr. Thomas Madura	Department of Physics and Astronomy

Abstract

Studying galaxy features affords astronomers valuable insight to understanding physics and matter on all scales, investigating both how galaxies' constituent parts behave and were formed. While the red sequence provides a good way to determine galaxy cluster membership, it is difficult to apply to galaxies that either do not fall on the sequence or are incredibly faint.

In this project, we apply the deep learning architecture ResNet-50 trained on webscraped Sloan Digital Sky Survey image thumbnails labeled from a combination of bright ($r < 19.4$) tagged Perseus Cluster data and existing spectroscopic data ($0.01 < z < 0.033$) to classify galaxies based off of cluster membership. On an independent test set we find that our method achieves promising results on relevant classification metrics.

Contents

1 List of Acronyms and Abbreviations

- Perseus Cluster Catalogue (PCC)
- Early Type Galaxies (ETG)
- Late Type Galaxies (LTG)
- Dwarf Elliptical Galaxies (dE/ETG)
- Lambda Cold Dark Matter (ΛCDM)
- Red-Green-Blue color model (RGB)
- Machine Learning (ML)
- Artificial Neural Network (ANN)
- Convolutional Neural Network (CNN)
- High Performance Cluster (HPC)
- Rectified Linear Unit (ReLU)
- Modified National Institute of Standards and Technology (MNIST)
- Graphical Processing Unit (GPU)
- Central Processing Unit (CPU)
- European Space Agency (ESA)
- Sloan Digital Sky Survey (SDSS)
- Dwarf Elliptical/Elliptical Type Galaxy (dE/ETG)
- Late Type Galaxy (LTG)
- Right Ascension (RA)
- Declination (DEC)
- True Positive (TP)
- True Negative (TN)
- False Positive (FP)
- False Negative (FN)

2 Introduction

2.1 Astronomy

Galaxy classification lends insight to the formation and evolution of galaxies throughout points in their lifetimes, as well as their elemental makeup. Traditionally, classification requires features to be manually extracted to be tagged - ending in both a time-consuming and potentially subjective endeavor. Given the ever-increasing volume of data gathered and generated from sky surveys and telescopes, the field is interested in a solution that can extract underlying features that human eyes and judgment may overlook, as well as those that humans can't see.

Previous efforts in classifying galaxies sorted the interstellar bodies into groups based on their shapes. This system is called galaxy morphological classification, or classifying galaxies by their morphology. Galaxy morphology allows astronomers insight into how galaxies were formed, how they interacted with their surroundings, and how they acted internally over time (Abraham and van den Bergh, 2001). One of the most common morphological systems referenced by astronomers is the Hubble Tuning Fork, the model proposed by Edwin Hubble which separated galaxies into classes of shapes based off of their age and star content (Hubble, 1927).

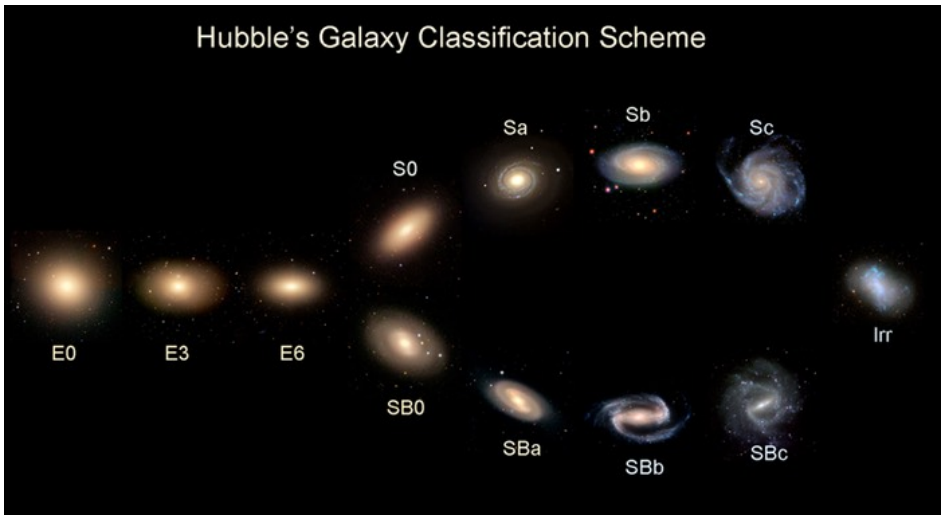


Figure 1: Hubble Tuning fork - Edwin Hubble's scheme for galaxy classification with three separate categories: Elliptical, Spiral, and Irregular. The two branches of spiral galaxies differentiate between galaxies with or without a bar. Source: `https://itu.physics.uiowa.edu/labs/advanced/classifying-galaxies/part-1-hubbles-tuning-fork`

The tuning fork, displayed in Figure 1, separates galaxies into elliptical, spiral and irreg-

ular, as well as a further split to delineate the difference between the types of spiral galaxies based off of the presence of a central bar.

Subsequent iterations of the model (de Vaucouleurs, 1958) would introduce subtypes such as the late ellipticals, very late spirals, magellanic spirals, and magellanic irregulars, as well as Sandage (1961) to introduce inner ring and pure spirals subtypes. Additionally, Hubble called called elliptical galaxies "early type" and spiral galaxies "late type" to denote the progression through the tuning fork from left to right. This unfortunately led to misconceptions, as Hubble did not mean for this to detail the progression through a galaxy's lifespan (as if the elliptical galaxies would evolve into spirals and irregulars) but rather that it was to denote the position in the sequence (Hubble, 1927). Despite this, we still use the "early" and "late" identifiers to differentiate between elliptical and spiral/irregular galaxies, respectively.

While we now know that Hubble's model does not detail the lifespan of a galaxy, since galaxy age can be better ascertained from the types of their stars, we still note that the Hubble model provides a useful way to classify and categorize galaxies.

It is important to note as well that in the tuning fork, the spiral type galaxies appear bluer, while the elliptical galaxies appear redder. But also included in Figure 1 is an irregular type galaxy, one which differs in structure from the spiral type galaxies and would be smaller and fainter. This leads us to consider small galaxies, otherwise known as dwarfs. Galaxies this small are particularly difficult to classify since they are faint. However, dwarf galaxies have been called interesting "laboratories" for understanding galaxy and star formation models such as the generally accepted model of ΛCDM (Lambda Cold Dark Matter)(Sales et al., 2022). Since dwarfs are so small, they have fewer stars than their larger counterparts, and are easier to simulate as a result. This also means galaxy formation processes are more pronounced and apparent for dwarfs.

In this project we create a method that considers both morphological and photometric features to classify galaxy cluster members. With a machine learning approach we train a model to "learn" the morphological and photometric features that determine cluster membership from only the image cutout from a given sky survey, removing the need to extract the data from the objects and cutting down on time to classify immensely. In order to understand how this can be done, it is important to first understand the use of color and color bands in astronomy.

2.1.1 Color Bands, and Photometry vs Spectrometry

We consider two techniques common to astronomy in this project: photometry and spectroscopy. Photometry is the measurement of the intensity/brightness of light from an object

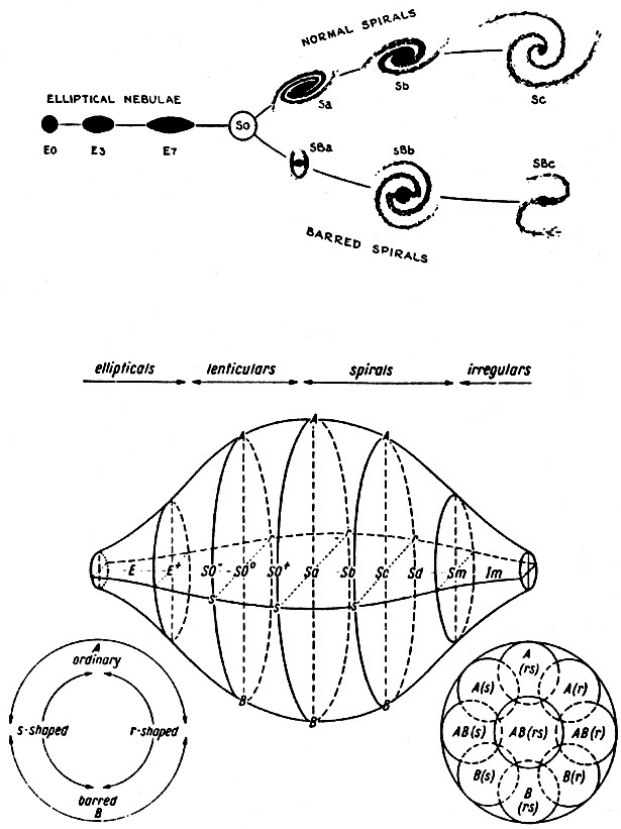


Figure 2: Hubble tuning fork shown above, with de Vaucouleurs' revisions(de Vaucouleurs, 1958) below, transforming the model from 2 dimensional to 3 with the inclusion of late ellipticals (E+), very late spirals (Sd), magellanic spirals (Sm), and magellanic irregulars (Im), also including the Sandage additions of the inner ring (r) and pure spiral subtypes (s).

across different bands or filters, while spectroscopy is the dispersive measurement of intensity of light over different wavelengths. While the two are different approaches to obtaining information about stellar objects, neither describes every relevant feature of an object, and so they are often used in conjunction.

Since light acts as a wave, the light from a star is characterized by the distance between two successive peaks in that wave, called wavelength λ. Within the range that human eyes can see, we know shorter wavelengths to appear more blue, and longer wavelengths to appear more red. Inversely related to a light wave's wavelength is its frequency f. This means that higher frequencies of light appear more red and lower frequencies of light appear more blue. This treatment of light describes all regions of the electromagnetic spectrum, where wavelengths between around 380 to 700 nanometers are visible to the human eye. Spectrometry is an astronomical technique that separates the constituent wavelengths (spectrum) of an object and measures the intensity of light at each wavelength.

In photometry, telescopes and other imaging systems are constructed to filter out different "bands" or regions of the electromagnetic spectrum so that the constituent parts of the light emitted from an interstellar body can be analyzed separately. Commonly, the magnitude of a star measures how bright it is, with lower numbers corresponding to brighter objects and higher numbers corresponding to fainter objects. When referencing a particular magnitude, one must indicate which color the magnitude corresponds to.

The Sloan Digital Sky Survey (SDSS) (Blanton et al., 2017) is a project aimed at mapping the night sky in the hopes to provide a catalogue of important objects in space. The photometric system used by SDSS (Fukugita et al., 1996) is comprised of ultraviolet (u), green (g), red (r), and two infrared wavelengths (i and z). Astronomers commonly use the subtraction of these magnitudes to symbolize color, such as $u - g$, $g - i$, and so on for all combinations. Larger color values, or the differences in magnitudes, correspond to redder objects and smaller color values correspond to bluer objects.

In addition to temperature, the colors of a star or galaxy give insight to quantities such as chemical composition, morphology, and most importantly - cluster membership. Galaxies that follow the color-based relationship called the "red sequence" were instrumental for us to train on so that our model learned relevant features.

2.1.2 The Red Sequence Method for Cluster Galaxy Identification

An incredibly common diagram to produce in astronomy is a Color-Magnitude diagram, since galaxies will fall into distinct categories based off of their evolution. The two regions of interest for this project are the red sequence, and the blue cloud, with Figure 3. The red sequence is a well known color-magnitude relationship between galaxy cluster members and background galaxies (Gladders and Yee, 2005), where astronomers are able to mathematically determine whether or not a galaxy is likely to be inside a cluster. This means that the early type elliptical or lenticular galaxies at a certain redness in the cluster will most likely have an expected brightness. Despite being early type galaxies, these red sequence galaxies are actually older. This means that the stars in the galaxy have become redder through age, as well as the galaxy having ceased to produce new stars from galaxy quenching (Graves et al., 2007). Quenching is the reduction or halting of a galaxy's production of new stars, and can happen from external processes such as such as nearby galaxies tidally stripping stars or internal processes such as gas and heat dispersion from cooling. Conversely, the active star production in the arms of spiral galaxies are the cause for their luminosity, and the galaxies are brighter than their redder elliptical counterparts. As a result, cluster members often appear in the red sequence and cluster background objects often appear in the blue cloud.

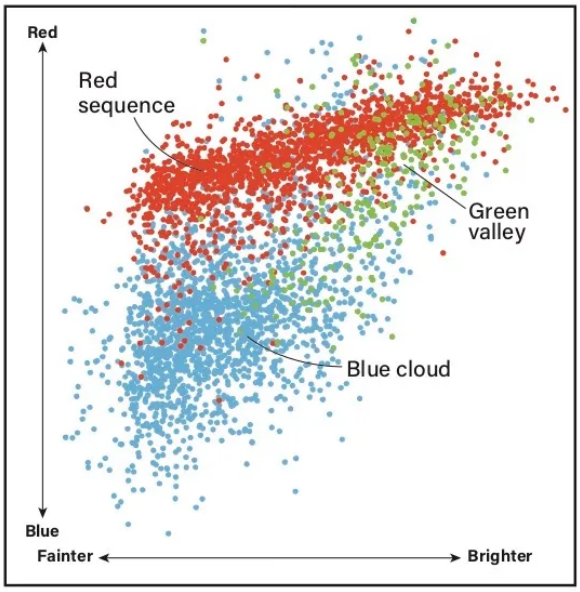


Figure 3: Color-Magnitude diagram showing the red Sequence, blue cloud, and green valley (Bakich, 2021), modified from Gavazzi et al. (2010).

However, this categorization does not perfectly predict galaxy cluster membership, especially given the presence of cluster galaxies that do not fall on the red sequence. This leads us to believe that more than just the red sequence is to be considered for these edge cases, since it would misclassify galaxies that are faint yet cluster members. Additionally, one can extract the spectroscopic redshift data from bright galaxies' spectra, but this becomes much more difficult and expensive for the fainter dwarf galaxies. It is also important to consider a red sequence galaxy may appear both red and faint, where it would be difficult to determine whether the object was in fact a cluster member, or if it was a giant galaxy in the background.

We applied a machine learning method to introduce a different approach to cluster galaxy classification that does not sort based purely on magnitudes and colors, as well as built to handle the red sequence edge cases mentioned previously. In order to understand how a machine learning model might be able to accomplish such a task requires us to explain how machine learning works.

2.2 Machine Learning

Machine learning is the field of study that is interested in enabling a computer to perform tasks without explicit instruction. Since present-day experiments in astronomy produce

output data in the form of images, and other fields of study have successfully used machine learning to classify images, such as applying machine learning for handwriting recognition (Guo et al., 2017), astronomers have sought to employ the same techniques for galaxy image classification. While machine learning is not a new method, the presence of large enough data sets to justify its widespread application is. This increase in data availability in tandem with heavy improvements to computer hardware positions machine learning to be an increasingly attractive avenue for analyzing galaxy image data as the fields of both data science and astronomy advance.

The basic categories of machine learning are supervised, unsupervised, semi-supervised, and reinforced learning, with a helpful diagram displayed in Figure 4. Supervised learning is where a model is given sample data with correct labels which are used to verify the model's performance. Unsupervised learning is where a model is trained on unlabeled input data, learning hidden patterns in the data. Semi-supervised learning is a mixture of the two, where the model interacts with both a low amount of labels and unlabeled data. Reinforcement learning is where analysis of positive and negative outcomes are fed back into the model to tailor its behavior. Since this project seeks to classify new galaxy data after processing data that have been correctly labeled, the project falls under the category of supervised learning.

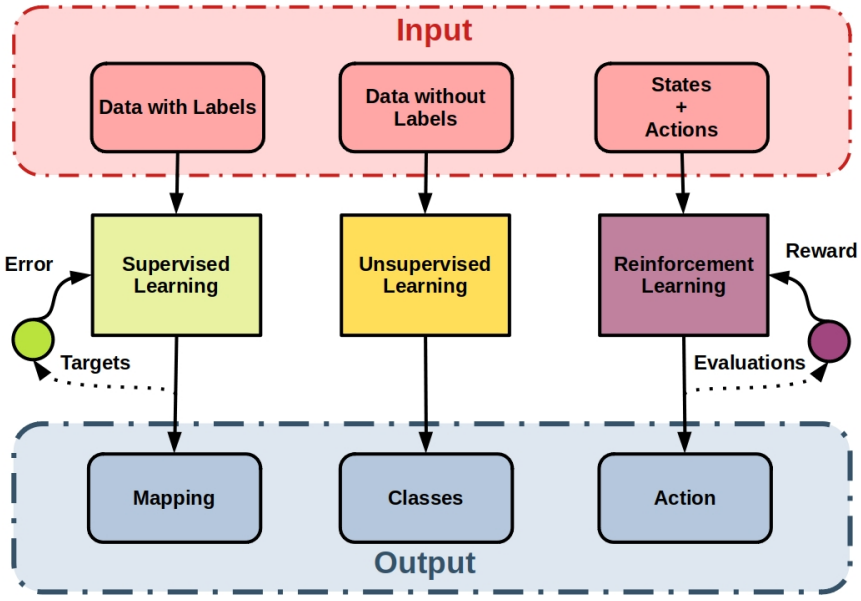


Figure 4: Subdivision of Machine Learning by supervision
https://starship-knowledge.com/supervised-vs-unsupervised-vs-reinforcement

Since the dataset that our model learns on is large (after preprocessing, over 1000s of

objects), we look to a subset of machine learning called deep learning. The field of deep learning leverages many-layered neural networks to learn to solve problems similarly to human learning processes. To understand this, as well as our efforts in this project, it is important to first understand neural networks.

2.3 Neural Networks

Neural networks are mathematical models constructed to mimic the way mammalian brains learn by using multiply connected "neurons" organized in layers, where each neuron processes information from the previous layer, allowing for complex pattern recognition and decision-making tasks.

Neural networks are constructed to accept an input vector the size of the number of features for given problem, then to process the input through any number of intermediate layers (often referred to as "hidden layers"), and to then output a vector the size of the number of possible predictions. A diagram of a simple neural network is shown in Figure 5. One pass through a neural network is commonly called an "epoch". Through successive passes, one wishes to reduce the "loss" or "cost" function, which describes the distance between the desired outputs of the neural network and the actual outputs. There are many different loss functions each made for different tasks.

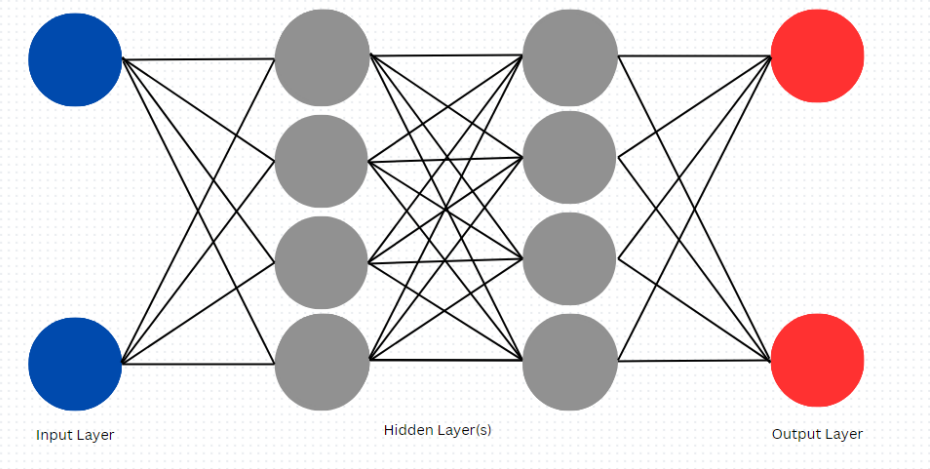


Figure 5: A simple neural network

2.4 Convolution and Convolutional Neural Networks

Pattern recognition is a task that scientists hope to train computers to outperform humans. While this is currently an area where human vision often proves to perform better, some

CNNs have outperformed even expert classification (Buetti-Dinh et al., 2019). Besting human performance is of interest since the eye can recognize and extract impressive amounts of information from sensory input. For example, a human can easily distinguish between digits such as those shown in Figure 6, even with variations such as handwriting or shifts.

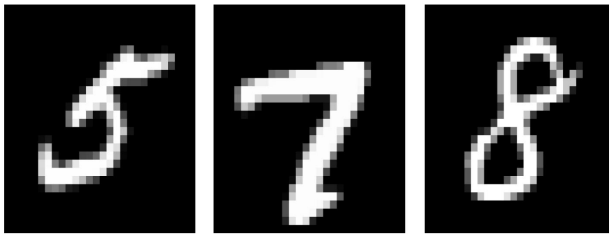


Figure 6: MNIST Handwriting digits, a classic example detailing images the human eye easily can recognize and categorize. Source: http://yann.lecun.com/exdb/mnist/

However, for a computer, the inclusion of small variations in handwriting or styles causes problems. In order for a computer to achieve the same result, researchers have combined the mathematical tool of convolution and neural networks for image classification in the form of the Convolutional Neural Network (CNN). CNNs have already yielded impressive results outside of astronomy for endeavors such as facial recognition (Jiang and Learned-Miller, 2017). In order to best understand the reasons for why CNNs are able to approximate human subconscious feature extraction we digress into the procedure of convolution from mathematics. In mathematics, the convolution of two functions f and g is defined as

$$f(t) * g(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau, \tag{1}$$

(also denoted as $(f * g)(t)$). This operation outputs a new function that shows where the two input functions are similar, shown in Figure 7. Function convolution has found applications in various fields, such as signal processing (Smith, 1997), numerical methods (Press, 1989), and even in scattering in an atomic lattice (Read, 2009).

In the regime of neural networks, convolution occurs as one of the network's layers. The convolutional layer receives the image in the form of a matrix. Here, the layer treats the matrix as one of the two functions to convolve, with the other function of a smaller square "kernel" matrix that is made to scan across the image grid. In the same way that two convolved functions activate when similar (Figure 7), the kernel matrix in this method is constructed to activate when a given feature is recognized (Figure 8) so the different regions of the image are described and associated to different features. This step constructs a feature map, also called an activation map. The extracted image features are then associated to the

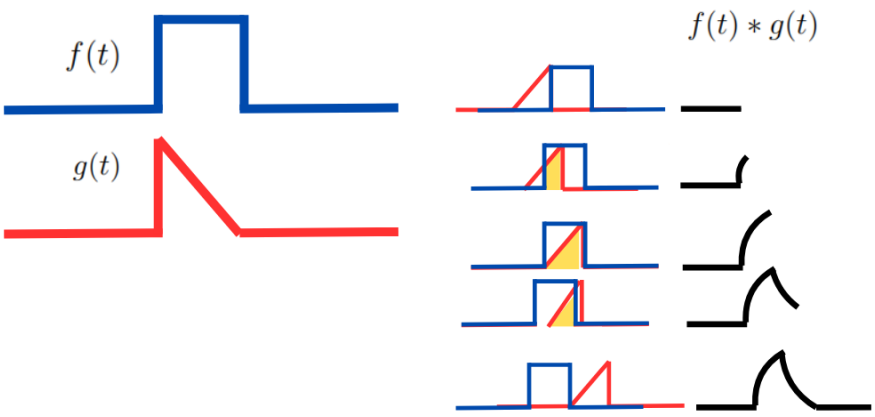


Figure 7: A visual representation of the convolution of two functions, here where function $f(t)$ is a square pulse and $g(t)$ is a triangle pulse.

features of the classification targets so that the network can attempt to classify input images. It is at this step where an activation function is applied so that the network recognizes the features from the activation map. There are many commonly applied activation functions, but for the purposes of CNN, the rectified Linear Unit (ReLU) activation function is most often chosen. ReLU is defined as

$$f(x) = max(0, x), \tag{2}$$

and has been found to perform better than other hidden layer activation functions for deep learning techniques (Brownlee, 2020). One reason why ReLU is chosen specifically for CNNs is that it helps preserve performance between many layers, which CNNs commonly hold. Another advantage of the ReLU function is that it allows for faster training than the alternatives, for example the $tanh(x)$ and sigmoid functions, without large reductions in accuracy.

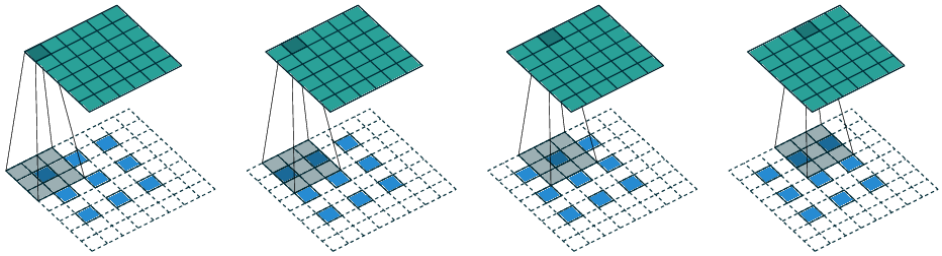


Figure 8: Step by step visual representation of 2D convolution, here where the 6×6 kernel matrix scans over the 8×8 pixel image (Shi et al., 2019)

While CNNs do well with image processing, the method holds a particular tendency to

overfit. Overfitting is where a model performs well on training data but poorly on new data. A common approach to reduce overfitting is the removal of extraneous connections between network neurons every iteration, a method dubbed "dropout". However, Khalifa et al. (2017) describe a CNN architecture that was able to classify galaxies to an accuracy of 97.272 percent, avoiding the problem of overfitting without the inclusion of a dropout layer. Given the diversity of approaches that implement CNNs, Becker et al. (2021) compare different CNN architectures and evaluate their performance with different metrics. Most interestingly, Dieleman et al. (2015) was able to achieve more than 99% accuracy on the Galaxy Zoo (Lintott et al., 2008; Willett et al., 2017) challenge on Kaggle in 2014. Dieleman fed image data to a CNN model to classify galaxies based off of morphology, utilizing CNNs' strength in extracting images' constituent features. The approach was particularly successful because CNN methods are invariant to rotations, which we will also make use of. To further understand how this works, as well as understand the model we employ for this project, we discuss deep neural networks in deep learning.

2.5 Deep Learning and Residual Learning

When models process an incredibly large amount of data, the model must be built to correctly characterize the multitude of features the dataset possesses. To account for a large amount of features to learn, researchers will increase the depth/number of layers in their models. With more layers, models can extract finer and finer details, such as the importance of the outline in an image, like in Figure 9.

Problems arise such as vanishing/exploding gradients, where successive passes through the model will cause the changes in the model weights to tend to infinity or zero. Both cases result in the model never converging to an optimum solution. This is another reason for the choice of ReLU as the activation function, as Sigmoid and Tanh contain a high variance between their inputs and outputs. It can easily be seen that successive derivatives of such functions will cause the weights' gradients to vanish, as shown in Figure 10.

2.5.1 ResNets

The model architecture that we chose to use for training comes from the work done to develop residual networks for image classification by He et al. (He et al., 2016). In this paper, the authors developed a technique to avoid accuracy degradation as model depth increases. This problem is different from the vanishing/exploding gradients problem, where convergence is made difficult at the outset of training. The degradation mentioned is an initial accuracy increase, and then a rapid fall. To combat this, the authors change the mapping of the

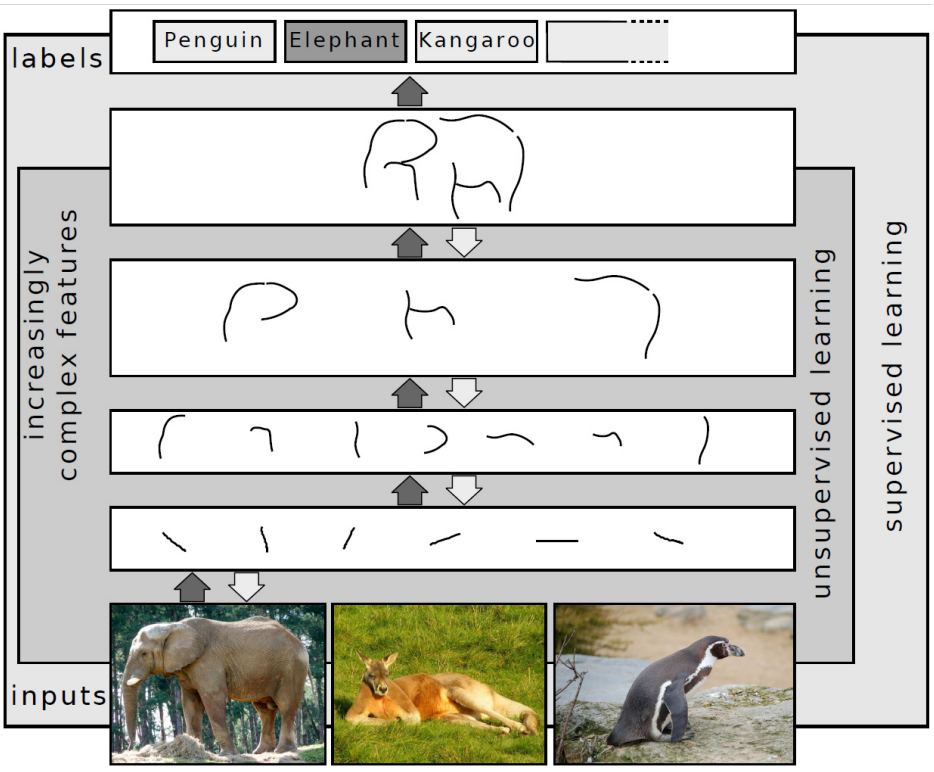


Figure 9: A visual example of how the inclusion of more layers aids in processing increasingly complex features(Arnold et al., 2011)

problem. If some target classification $\mathcal{H}(x)$ is desired given an input x, then considering that some residual $\mathcal{F}(x)$ can be defined as

$$\mathcal{F}(x) = \mathcal{H}(x) - x, \tag{3}$$

or the difference between the desired output and input. He et al. then solve to find

$$\mathcal{F}(x) + x = \mathcal{H}(x). \tag{4}$$

This meant that the network need only train to classify the residual $\mathcal{F}(x)$ and then recombine with the input x in order to find the desired classification function. With this reformulation of the problem, He et al. develop "skip" or "shortcut" connections to find $\mathcal{F}(x) + x$ through a mapping that adds no additional parameters since the connections simply map the identity to the end. Displayed in Figure 11 is a residual block, which is a building block for high depth ResNet neural networks that the paper pioneered. These networks perform better than some of their higher parameter, higher computational complexity counterparts such as VGG (Visual Geometry Group), despite ResNets possessing greater model depth - a trait that He et al. noted to usually led to higher training error for other

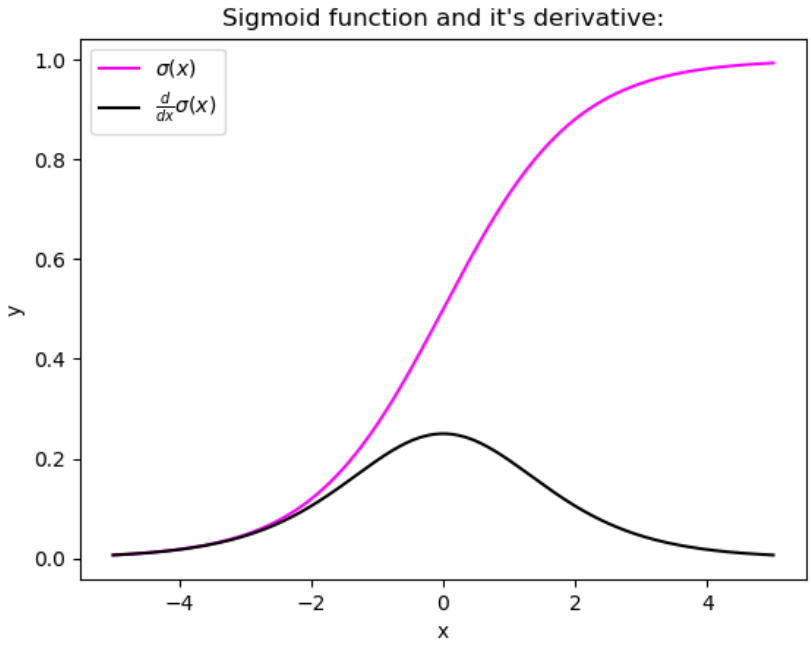


Figure 10: Visual Representation of vanishing gradient - since the derivative of sigmoid is close to zero, successive applications of the derivative will cause the change in the model's weights to go to zero.

models.

Our project in particular employs ResNet-50, which includes further improvements on the residual blocks. Figure 12 displays the schematic for a "bottleneck" building block. The bottleneck layers still make use of the identity shortcut technique, as well as the 1×1 convolutional layer initially reducing and subsequently increasing the dimensions, the 3×3 acting as the bottleneck before fully reforming to make the residual.

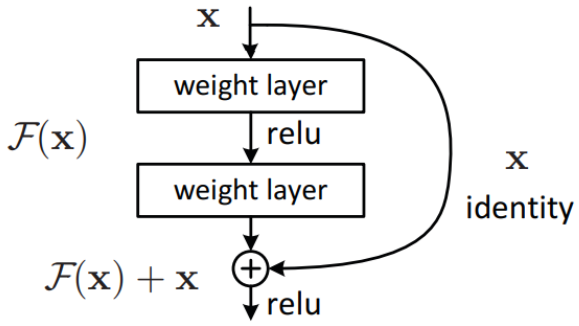


Figure 11: The residual block detailed in He et al.

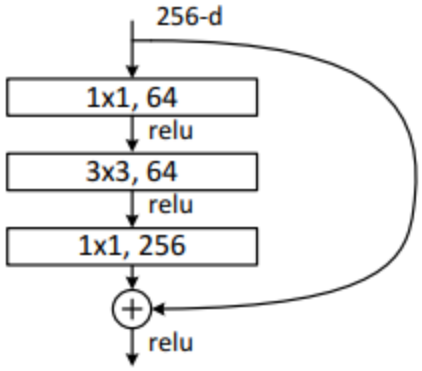


Figure 12: The schematic for the "bottleneck" building block included in ResNet-50, the project's model of choice.

2.5.2 Adapting ResNet-50

For this project, we construct ResNet-50 and modify the input layer to accept $200 \times 200 \times 3$ image tensors. The output layer was also changed to size 2 to represent our classes, with the softmax activation function. Normally sigmoid is the activation function of choice for the output of binary classification, but since issues arose the problem was treated as a multiclass classification with two output classes and therefore necessitated the softmax function. Otherwise, the ResNet-50 architecture is unchanged, with Figure 13 displaying the architecture described from He et al. (2016).

2.6 Hardware

It is notable that we require Graphical Processing Units (GPUs) instead of Central Processing Units (CPUs) due to their advantage in computing a higher volume of low level tasks, as well as their ability to distribute tasks to compute simultaneously. While CPUs can certainly run the level of operations for the given task, the greater number of cores in GPUs that these operations can be distributed over leads to faster performance.

And while it is possible to train our models on computers equipped with modern GPUs, we leverage San José State's High Performance "Spartan" Cluster to train our model given the availability of multiple compute nodes and the ability to remotely train. At the time of writing, the Spartan HPC GPU compute nodes provide up to 68.6 TFLOPS at peak performance from 17 NVIDIA Tesla P100 12 GB GPUs.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 13: Table from He et al. (2016) detailing different ResNet architectures. We employ the 50-layer ResNet-50 with modifications to the input and output layers.

2.7 Problem to Solve

Making use of the William Herschel Telescope (WHT) covering 0.7×0.7 Mpc^2 of the Perseus Cluster central region, Wittmann et al. (2017) created an deeper and wider survey than SDSS that was then later catalogued by Wittmann et al. (2019). We call this the Perseus Cluster Catalogue (PCC) and it served as the starting training set for our model.

In this project, we apply a Convolutional Neural Network (CNN) method to classify galaxy cluster members trained on tagged Perseus Cluster Catalogue objects from the work of Wittmann et al. (2019) and an additional bolstering set of objects with labels from their spectroscopic data. The training of this model will both automate the task of classifying cluster galaxies and create a method that can learn implicit features from the image thumbnails that the human eye cannot resolve. The training will also hopefully afford insight on ultra diffuse galaxies (Gannon et al., 2021; Wittmann et al., 2017), a subset of dwarf galaxies, since they are pivotal to understanding and testing early galaxy and star formation models.

3 Training Methods

A step-by-step overview of the training process is:

1. Collapse Classes (Cluster Member/Background non-member)

2. Bright Cut ($r < 19.4$)

3. Overly Red Image Filter

4. Balanced Random Sample

5. Rotate Images

3.1 Data Acquisition and Brightness Cut

We webscraped 200×200 pixel image cutouts at 0.1 arcsec/pixel scale from SDSS Data Release 16 (Ahumada et al., 2020), at an object's RA (Right Ascension) and DEC (Declination) and feed these to our model as the input. A variety of examples of image cutouts are displayed in Figure 14.

3.2 Collapsing the Classes from Wittmann

We take the 5437 tagged objects from Wittmann, which were initially split between multiple classes. These were:

1. Cluster or background Late Type Galaxy (LTG) (384 objects)

2. Likely background Early Type Galaxy (ETG) or unresolved source (3008 objects)

3. Likely cluster or background edge-on disk galaxy (1049 objects)

4. Likely dE/ETGcluster candidate (398 objects)

5. Likely merging system (23 objects)

6. Possible dE/ETGcluster candidate (98 objects)

7. Background galaxy with possibly weak substructure (477 objects)

where dE/ETG is dwarf elliptical/early type galaxy. Given the similarities between the classes, we recognized that many of them could be grouped into two classes - cluster candidate or cluster background galaxies. Classes 1,2,3, and 7 were combined for the background classes, and classes 4 and 6 were combined for the member classes. We also exclude class 5 entirely. With this combination, the model now serves to determine cluster membership of a given galaxy.

(a) PCC-3848, background galaxy (class 0), notably a bright background object.

(b) PCC-313, cluster member galaxy (class 1), notably a bright member object and red sequence galaxy

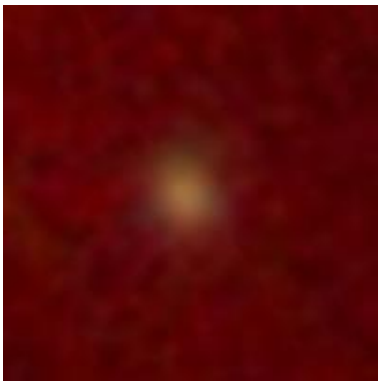
(c) PCC-825, background galaxy (class 0), notably a faint background object.

(d) PCC-2039, cluster member galaxy (class 1), notably a faint member galaxy object in the blue cloud.

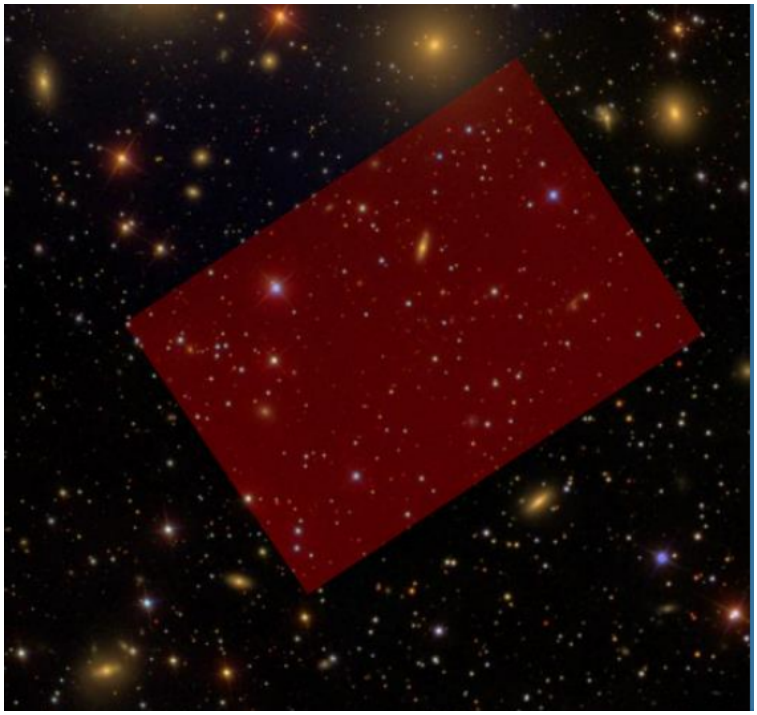
Figure 14: A variety of SDSS image cutouts in the training set, showcasing bright and faint objects in both member and background classes.

3.3 Feature Engineering - Filtering and Random Sampling

An initial training run produced a model that heavily overclassified background galaxies. We found this to be because most images that were in the original 5437 training set were extremely faint, and determined that there were very little features in these faint images for the model to learn. By choosing to look at images brighter than $r < 19.4$, we allow our model to experience more apparent features while still preserving the ability to classify fainter galaxies. After the discriminating on the brightness threshold, we are left with 272 objects. Furthermore, we note that a small portion of the remaining images seem to have



(a) Example of an overly-red image in our dataset



(b) The overly-red region in the Perseus Cluster

erroneously been colored red, shown in Figure 15a and Figure 15b. This is the result of an imperfect sky subtraction, where the i-band of the the image is under-subtracted, leaving too much red light in the image. We credit Dr. Aniruddha Thakar at SDSS putting us in contact with an unnamed expert for this explanation. These objects are then dropped from the training set by using the OpenCV python package to extract the amount of red pixels from the image, and removed if the amount exceeds 50% of the total image pixels. This left 230 total images in the training set, with 114 member objects and 116 background objects.

3.4 Data Augmentation - Rotations

This significant reduction in the data set from 5437 images to 230 means that the model has much less data to train on, and so we sought to apply data augmentation techniques to bolster the training set.

There are many data augmentation techniques to apply to increase the size of a training set of images, the most notable of which are:

1. Flipping
2. Cropping

3. Translating

4. Rotating

5. Zoom or Scaling

6. Brightness or color changing

7. Noise addition

We chose to only apply rotations to our images as the other methods would not preserve the physics of what the model is expected to classify. For example, the color of a galaxy is intrinsically related to its physical properties, so a color or brightness change would not be physically reasonable. Similarly, zooming or scaling would imply the galaxy was a different size than actually observed.

A train/test split of 75/25 is then applied and the images are augmented by rotating on the common unit circle angles, shown in Figure 16. Since 0 and 360 degrees would produce the same image, only the first image is kept. This bolstered the training set by producing 15 times as many images (3450). Special care is taken during the train/test split so that the augmented images do not repeat objects between the training and test sets (e.g. Object 45 will not appear rotated 45 degrees in the training set and then 90 degrees in the test set). In addition to the 75/25 train/test split, a further 75/25 validation split is applied to the training data. The validation set is important to include so that the model's performance can be checked during training, where we notice that the large or increasing gaps between the training loss and validation loss are indicators of overfitting.

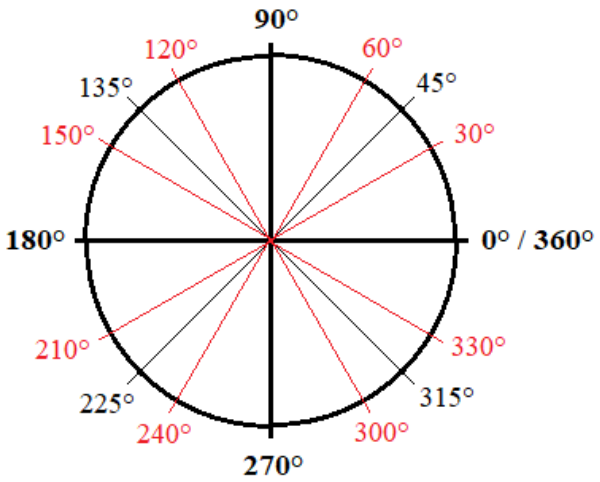


Figure 16: The angles chosen for the images to be rotated - the common unit circle angles

3.5 Hyperparameter Tuning

Hyperparameters are parameters set before training a model and are therefore not learned by or during the process of training. The inclusion of a validation set lends insight to which hyperparameters are better than others. In general, when the training and validation losses are close together, a model avoids the problem of overfitting. If the losses between the two diverge, the model will tend to overfit, since this signifies that the model is much better at only classifying the data it is trained on in comparison to new data it has not experienced. For this project, to find an optimal set, the hyperparameters that were varied between different models were number of epochs, learning rate, and batch size. Other relevant but unvaried hyperparameters were the choice of optimizer (Stochastic Gradient Descent) and the train/validation/test split ratio. We also held the train and validation batch size at 20, choosing to vary learning rate instead of batch size taking the recommendations from a hyperparameter tuning-focused paper (Smith, 2018).

Hyperparameters specifically relevant to CNNs include modifications to the kernel matrix. While these were not varied per model, they are important to consider for the given model architecture, ResNet-50. For instance, different sizes of kernel matrices to scans over the image during the convolution steps vary how the convolutional layer places emphasis on the features it recognizes. Another relevant hyperparameter to vary is the stride, which is analogous to stepsize as the kernel scans over the input image. Additionally, ResNet-50 includes padding, which has the kernel matrix scan over extraneous grid points surrounding the image. An example of padding is displayed in Figure 17.

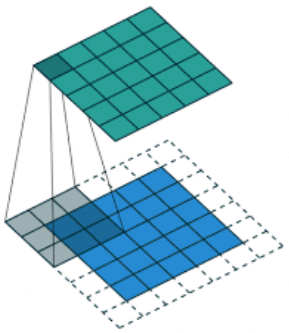


Figure 17: An example of 2D convolution, here using a 3×3 kernel matrix and the inclusion of padding (Pröve, 2017)

The perfect mixture of hyperparameters is difficult to find as it varies from project to project based off of a problem's formulation and data considered. Hyperparameter tuning is an endeavor that balances between the possibility of over or underfitting, with consideration

to costs such as hardware capability as well. For example, without even considering its effects on the learning rate, one cannot increase epoch number without bound. Such an increase would require more computing time on increasingly more powerful hardware. The best set of hyperparameters are those that provide the best learning rate while avoiding over and underfitting in reasonable time with consideration to hardware limitation.

Our optimal set of hyperparameters ended up at 200 epochs, a training and validation batch size of 20 images, an exponentially decaying learning rate starting at 0.08, decaying at a rate of 0.9, displayed in Figure 18. Our models were able to train within 2 hours.

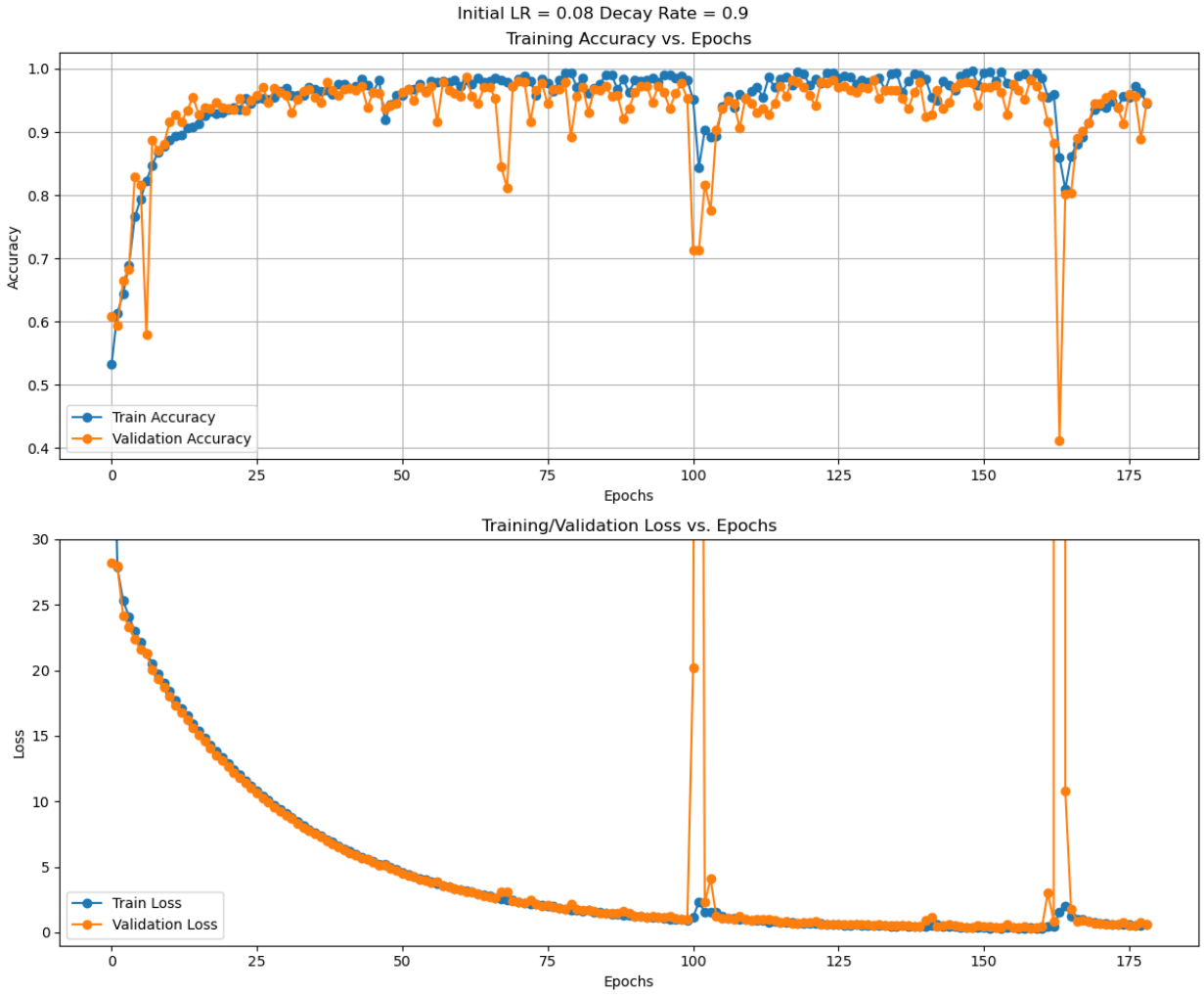


Figure 18: An example training run of the model, attempting to find optimal hyperparameters. Initial LR refers to the initial learning rate at the outset of a schedule that exponentially decayed at a rate of 0.9

4 Model Evaluation (Metrics)

In order to understand the performance of our model, we look to other metrics besides accuracy, as accuracy can potentially overstate the positive performance of a method on unbalanced data. In other words, if a model was given an imbalanced set of 98 cats and 2 dogs, one could still report a 98 percent accuracy if the model was correct on all cats but wrong for every dog. To circumvent this, we construct a confusion matrix, where the information of when the model classifies a False Positive (FP), False Negative (FN), True Positive (TP), and True Negative (TN) are all included. With access to these metrics, we are also able to calculate the model's Precision, Recall, and F1 Scores (defined below) on each class label.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1 - Score = 2\frac{Precision * Recall}{Precision + Recall} \tag{7}$$

An example confusion matrix is shown in Figure 19 for our initial evaluation of the PCC Object trained model.

5 Initial Results on Wittmann Catalogue Objects

The initial results from training on the chosen objects from Wittmann's Perseus Cluster Catalogue are shown in Figure 19, displaying a 97% accuracy on the balanced test set from the train/test split. This means that the model was able to learn the features that correspond to cluster membership for objects in the PCC.

To ensure the model took color related features into account, training was run with greyscale versions of the images (Figure 20), as well as the full color images. Since the accuracy (on the balanced test set) dropped by more than 5%, we concluded that the model's training took color features into account.

We also extract the filters from the residual blocks, shown in Figure 21. With this evaluation, it was time to see how the model performed on data outside of the PCC.

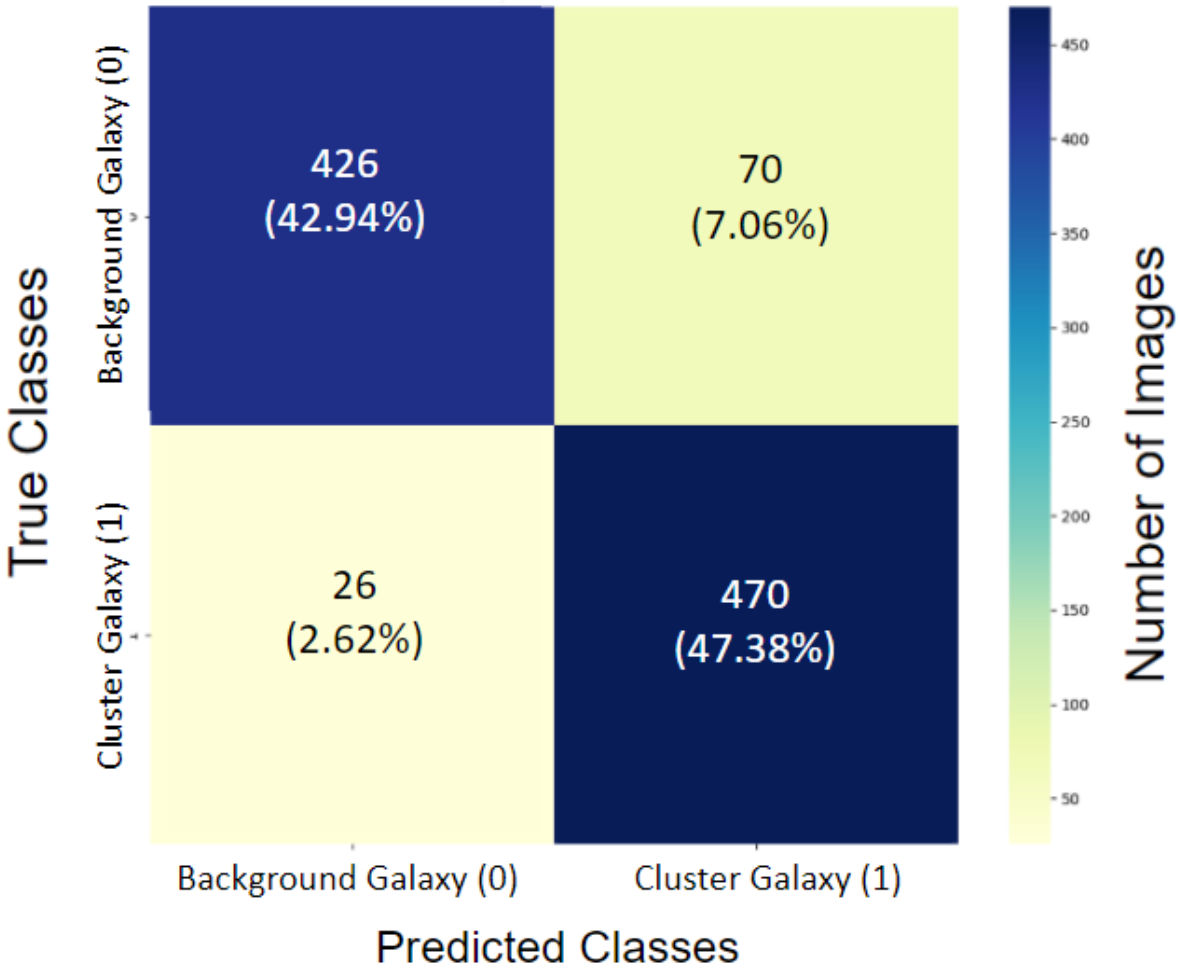


Figure 19: A confusion matrix detailing the classification report of the model run on full color images of PCC bright objects, after the dataset was expanded with rotations. Note that the member class is 1 and background class is 0.

6 Finding New Testing Data

After training on the PCC Objects, we wished to evaluate the model's performance on objects that were completely new, i.e. outside of the Wittmann's catalogue. While searching radially outward from the center of the Perseus Cluster would give us new objects, we would have no way to know if the model's prediction as correct. This meant that we needed to find an relationship between extracted data parameters to discriminate between background and cluster member galaxies. To do so, we created color-color plots of all of the different color bands to see if there were clearly defined selection regions to separate between cluster members and background galaxies. From a pair plot of all combinations of common pho-

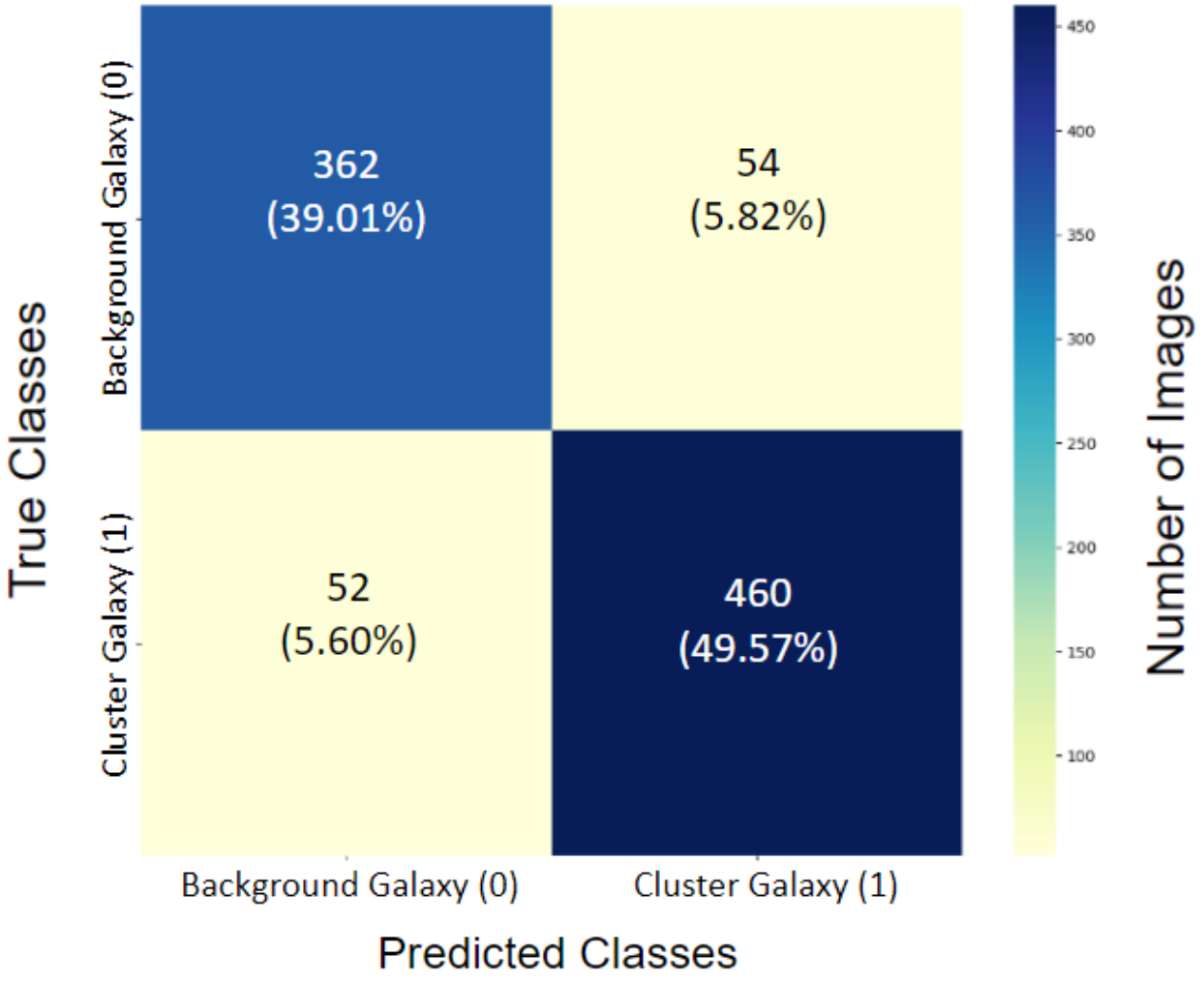


Figure 20: A confusion matrix detailing the classification report of the model run on greyscale images of PCC bright objects, after the dataset was expanded with rotations. We noted the performance decrease (relative to 19) and conclude the model did in fact learn RGB color based features from the dataset. Note that the member class is 1 and background class is 0.

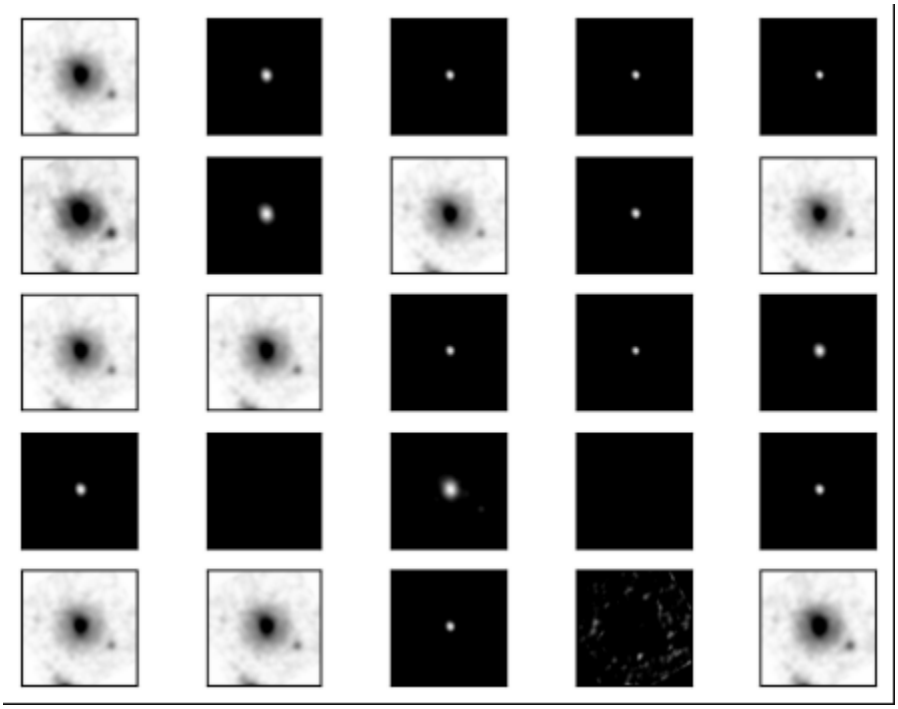


Figure 21: A subset of the filters from a residual block

tometric bands (Figure 23, we noticed that the most prominent selection regions could be made in the relationship between the r-band magnitude, r, and the $g - z$ color band, shown in Figure 22. This relationship is the red sequence.

Using SQL queries to SDSS, we acquired a new test set of 144 images to evaluate the model, starting from the center of the Perseus Cluster ($RA = 49.9467, DEC = 41.5131$) in a radial search up to 45 arcminutes outward. This SQL query is given in Appendix B.

However, the model that trained on bright ($r < 19.4$) PCC Objects was unable to properly classify the background class in these new objects, with the confusion matrix's off diagonal elements comprising of 21% of the total images. This poor performance is shown in Figure 24. We initially attributed this to some objects in the training set being labeled as galaxies under photometric data but stars in spectroscopic data. This mismatch motivated an investigation into a way to remove these stars erroneously labeled as galaxies, from which the model would learn undesirable features. One thought was to examine the SDSS image flags, which give insight to potential issues with the image. We extracted the flags from these objects and attempted to find common flags between stars and common flags between galaxies. Displayed in Table 1 are examples of some of the flags from these objects and what each mean.

Unfortunately, it appeared as if no discerning set existed; i.e. no flag was particularly unique to stars or galaxies alone. We found there to be 10 stars labeled as galaxies in the

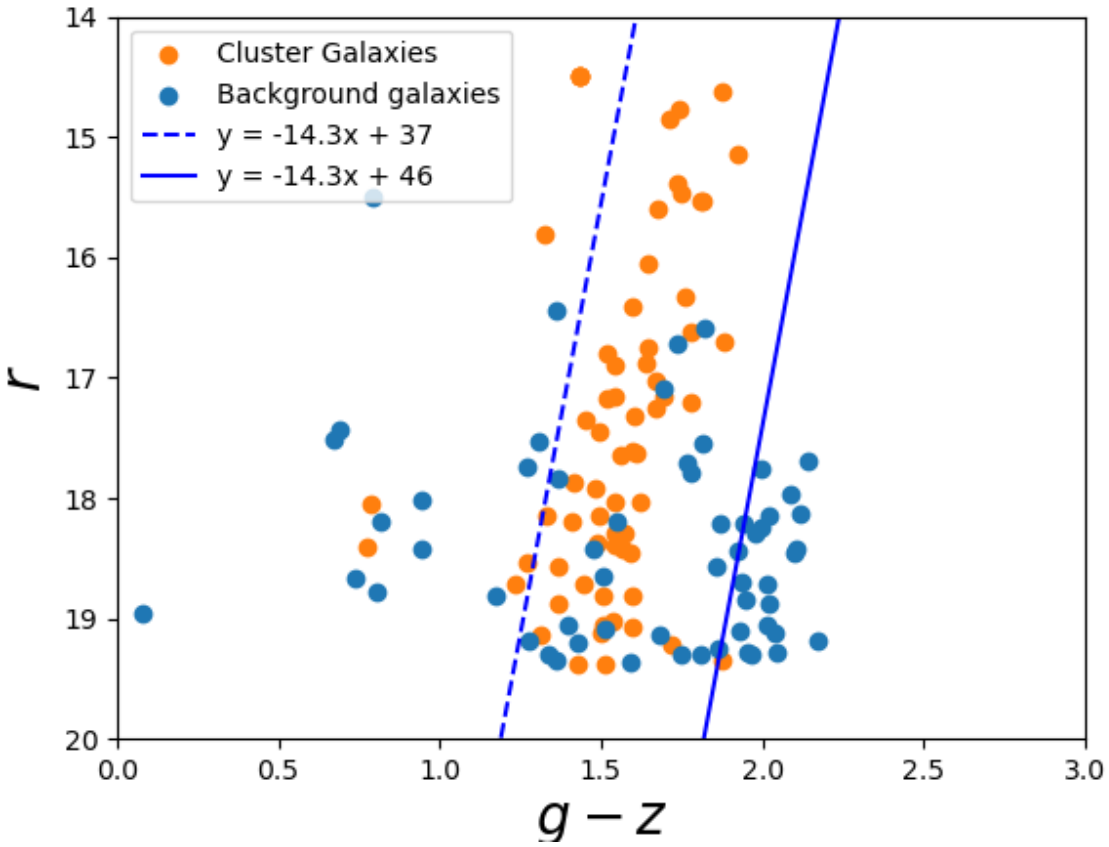


Figure 22: Color-magnitude plot that gave us the selection regions to pick out red sequence galaxies in photometry. The diagonal lines outline the red sequence.

PCC bright objects. Despite this, we were able to devise a method that performed even with the presence of stars in the training set by adding more objects in a radial search outward from the center of the cluster.

Since the model trained on the PCC objects initially performed poorly on data further from the center of the cluster (Figure 24), we determined that the objects to add to the training set would come from the spectroscopic search from before. Since spectroscopic redshift would give us a parameter related to distance, we found it to be a good indicator of cluster membership, as shown in Figure 25. This showed that cluster members appeared in the spectroscopic redshift region $0.01 < z < 0.033$, allowing us to assign ground-truth labels to data as we searched radially from the center of the cluster.

Inclusion of this data meant that our training set would not only be on the tagged objects from Wittmann's catalogue, but also added more objects that exist inside and outside the Perseus Cluster. This added 233 objects to our training set, meaning that post-rotations

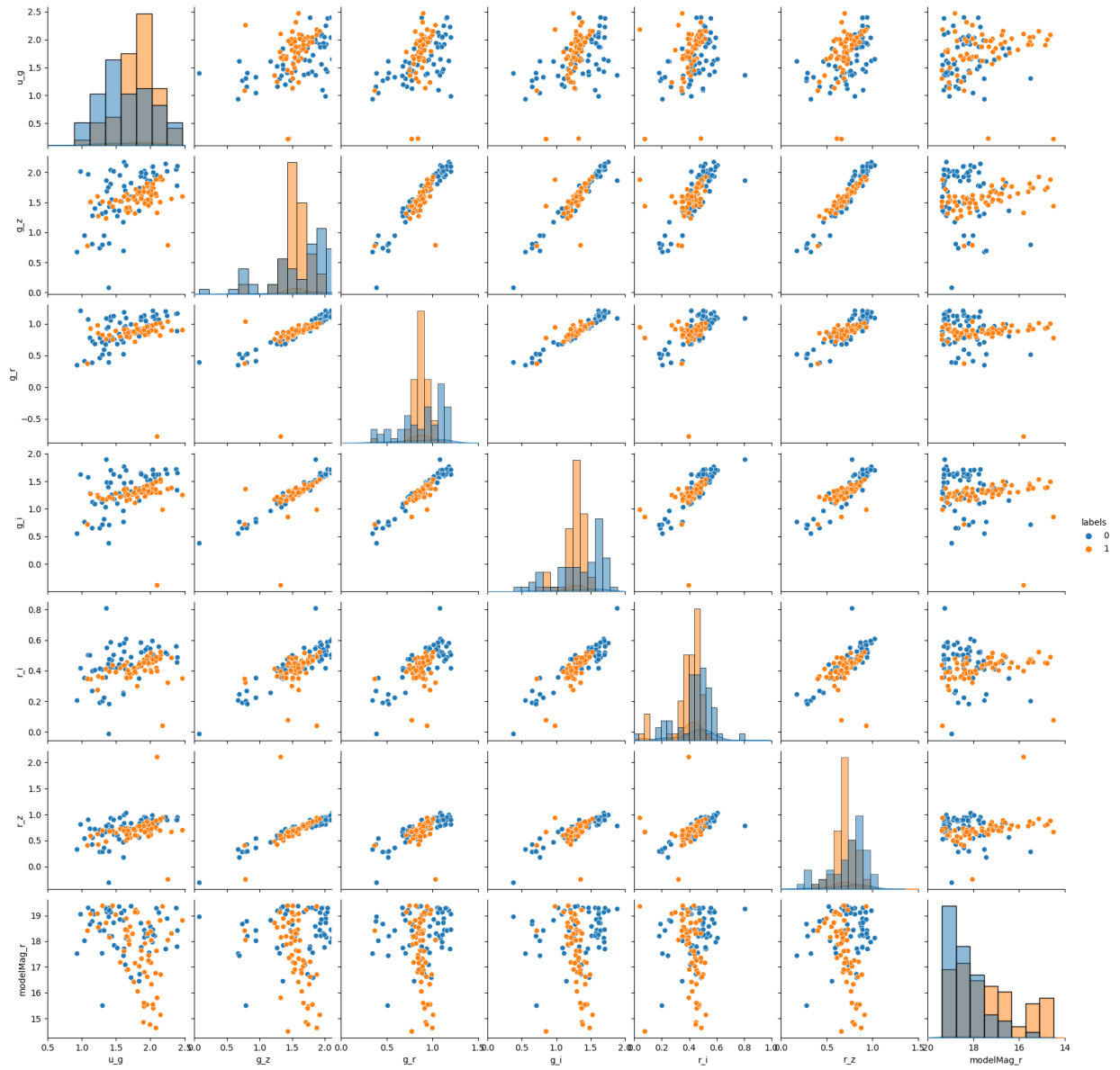


Figure 23: Pairplot of color band relationships from PCC Bright Objects. Member class (1) labeled in orange and background class (0) labeled in blue.

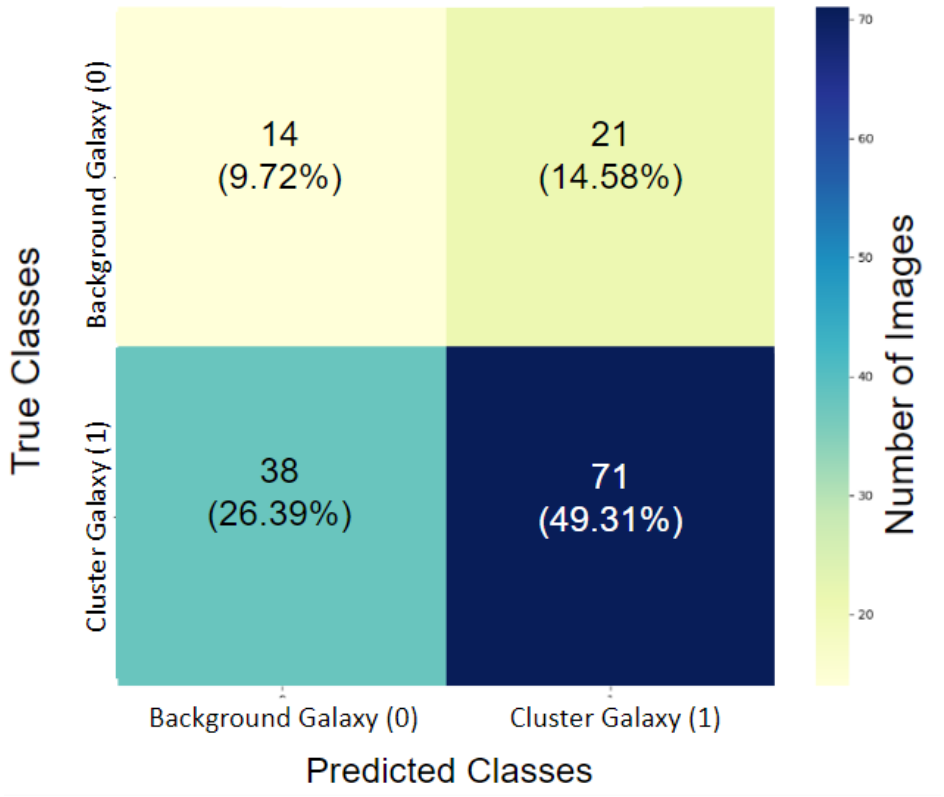


Figure 24: Model Performance trained on PCC brights on new test data, overclassification of member class necessitated bolstering of training set.

Flag	Explanation
NODEBLEND	Object recognized as composite and was unable to be separated (Deblended).
CHILD	Children of a deblended object.
SUBTRACTED	Marked when the extended wings around a bright star object are subtracted out.
SATURATED	Questionable photometry, i.e. one of the bands was found to have saturated pixels.
CR	Image in question contains a cosmic ray that was interpolated over.
NOPETRO	The object's Petrosian radius was unable to be determined.

Table 1: Examples of flags common to SDSS objects in our training set. Information from `https://live-sdss4org-dr16.pantheonsite.io/algorithms/flags_detail/`

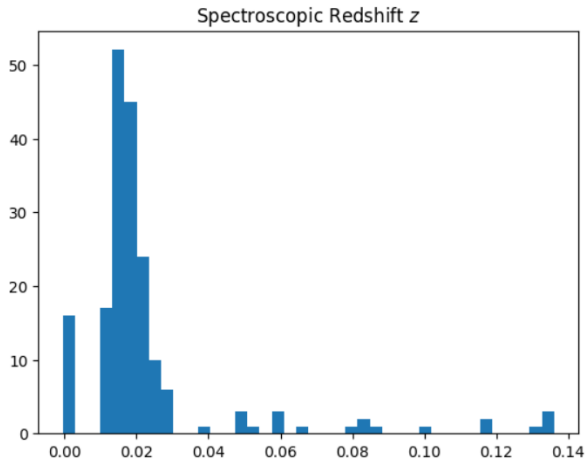


Figure 25: A histogram of the spectroscopic redshift (z parameter) of galaxies in a search up to 45 arcminutes radially outward from the center of the Perseus Cluster. This search was to find new objects to add to the training set. We note that the cluster members appear in the region $0.01 < z < 0.033$.

the model would train on 6945 images (pre train/test split). The color-magnitude plot of all objects in the training set is displayed in Figure 27, as well as the comparison between the two sets separately in Figure 28 (with PCC objects only) and Figure 29 (with Spectroscopic search only).

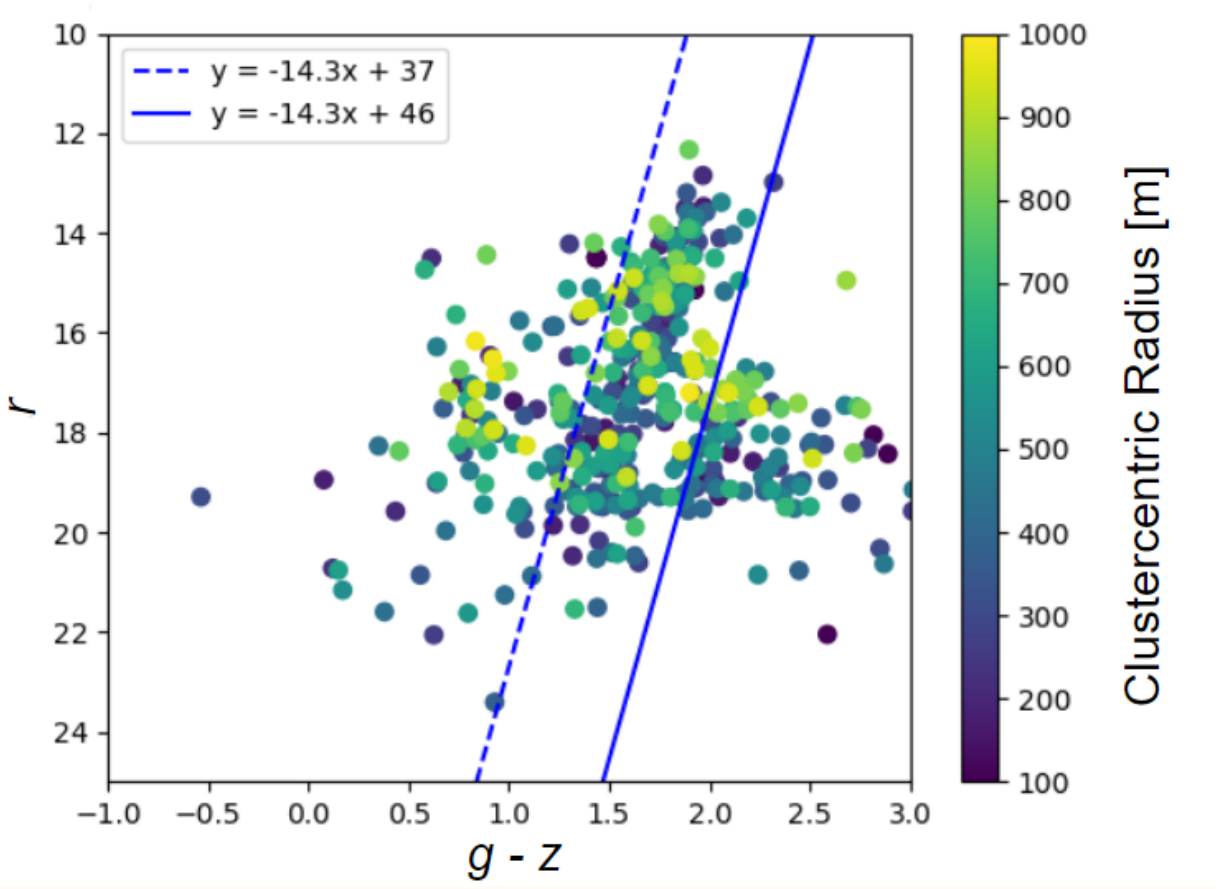


Figure 26: Color magnitude plot of all objects in the training set after combination of PCC brights and spectroscopic search. This set was also filtered for the overly-red objects as well. Colorbar is of clustercentric radius or the distance of a galaxy from the center of the cluster (in meters). The two lines dividing the data outline the red sequence.

7 Results from Spectroscopic Addition

After the addition of 233 objects from the spectroscopic search, we find that the model performs favorably on the initial test set from the train/test split, yielding the metrics displayed in Table 2

The accompanying confusion matrix is displayed in Figure 30 and training report in Figure 31. In order to independently verify the model, we employ another radial search up to 90 arcminutes outward from the center of the Perseus Cluster, but subtract out the common objects from the training set to ensure this search had no common objects with the training set. The SQL query to obtain this independent test set is also given in Appendix B. The performance of the model via confusion matrix is displayed in Figure 32 and the classification report in Table 3

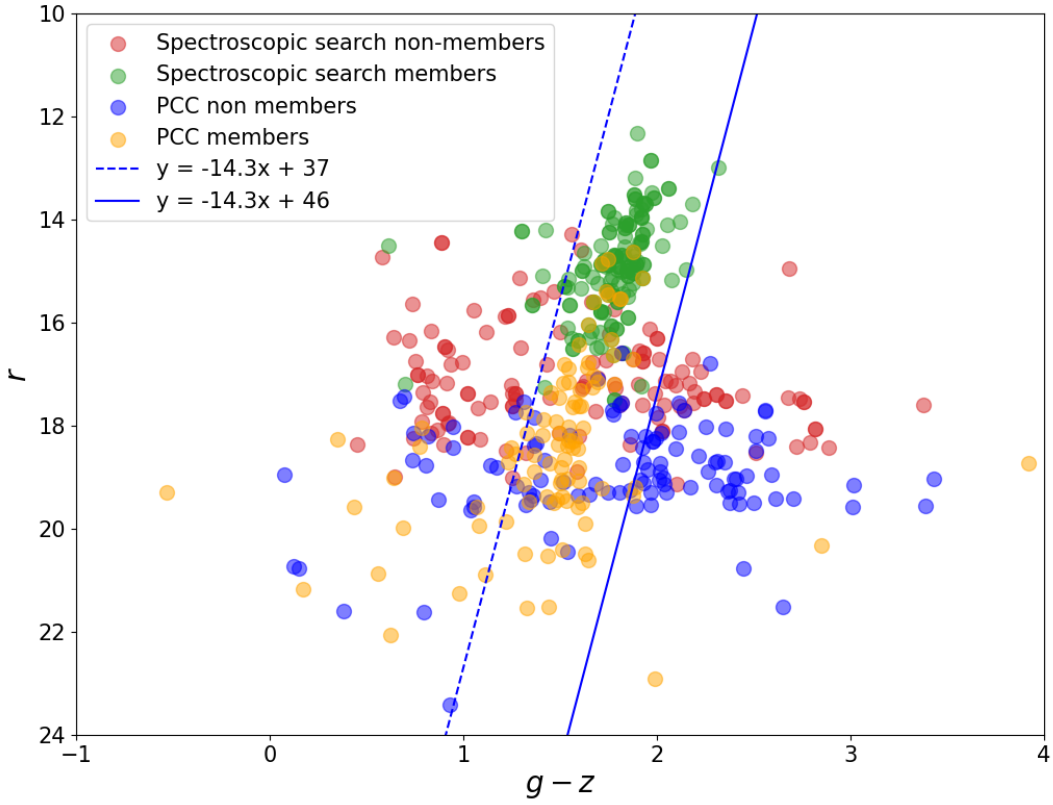


Figure 27: Color magnitude plot of all objects in training set after spectroscopic addition. The objects in red and green are from the spectroscopic search, with the members in red and the background galaxies in green. The objects in blue and yellow are from the Perseus Cluster Catalogue, with the members in yellow and the background galaxies in blue. Also included are the lines that estimate the red sequence boundaries.

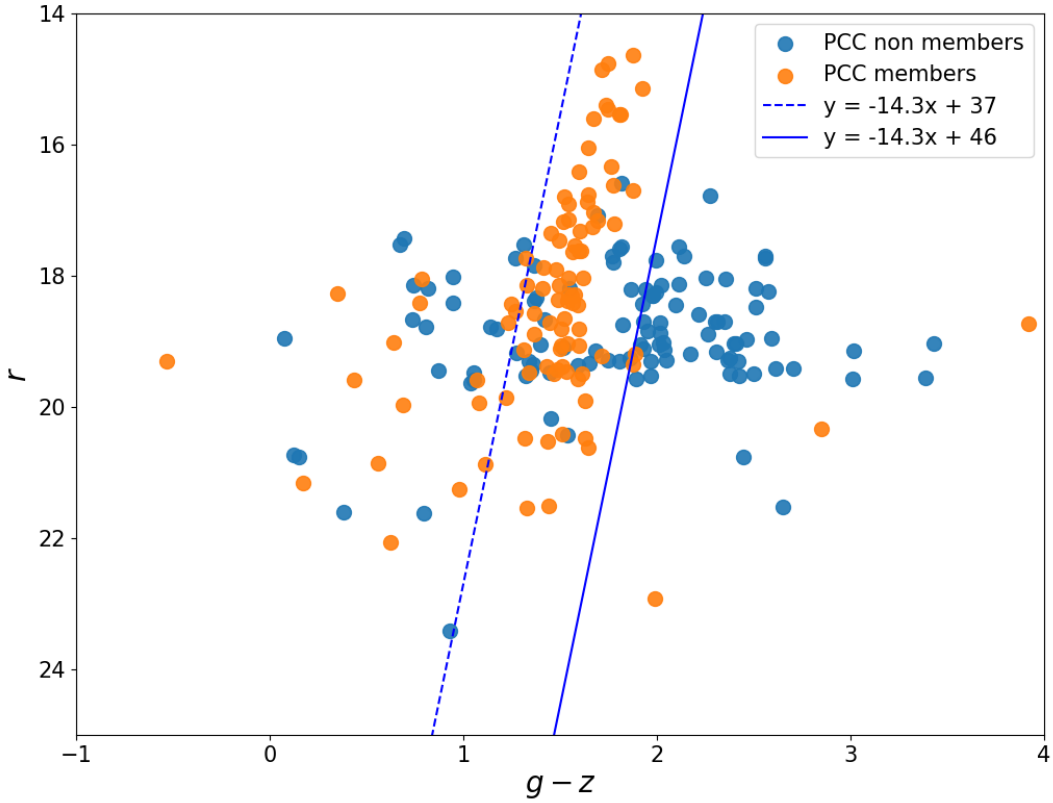


Figure 28: Color magnitude diagram of the training objects from the PCC with member objects in orange and background objects in blue.

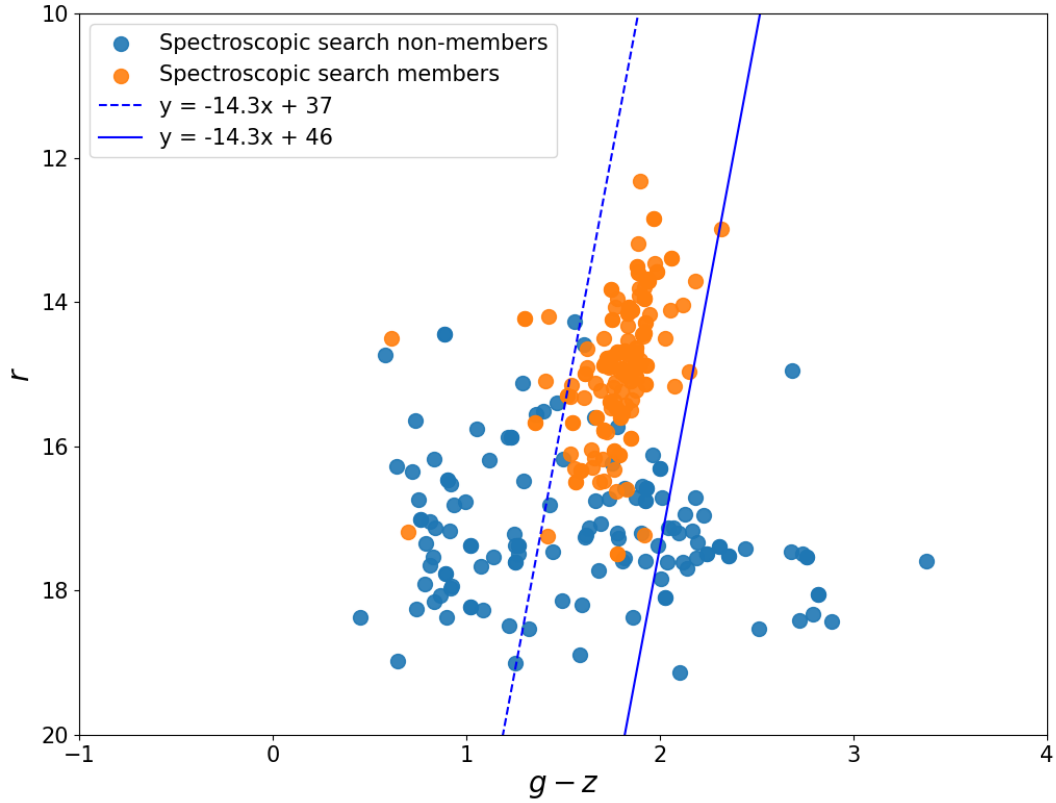


Figure 29: Color magnitude diagram of the training objects from the spectroscopic addition with member objects in orange and background objects in blue.

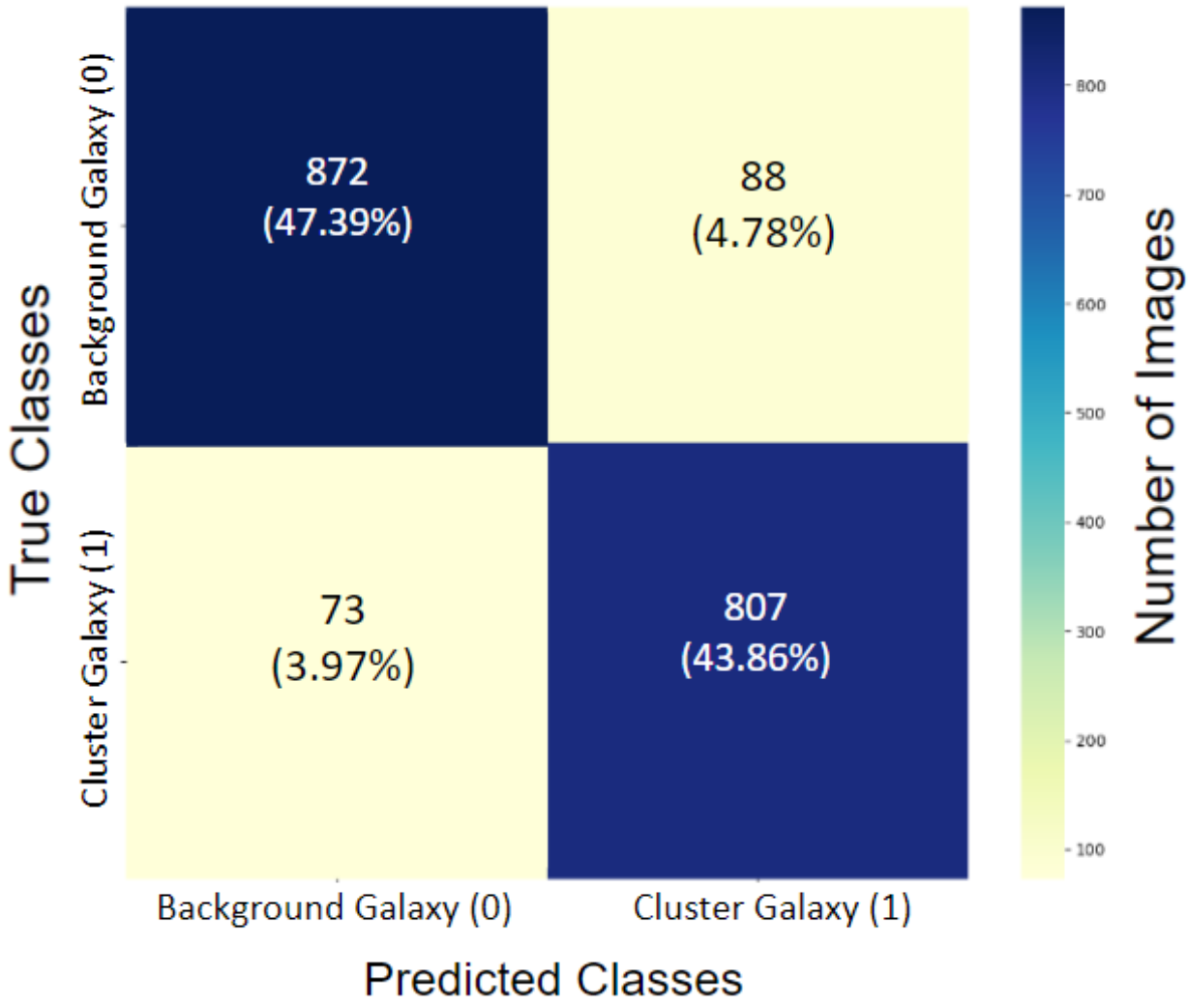


Figure 30: Confusion matrix of the model performance on the train/test test set, trained on both PCC and Spectroscopic objects

	Precision	Recall	F1-score	Support
Background (0)	0.96	0.88	0.92	960
Cluster Member (1)	0.88	0.96	0.92	880
Accuracy	-	-	0.92	1840
Macro Average	0.92	0.92	0.92	1840
Weighted Average	0.92	0.92	0.92	1840

Table 2: Classification metrics from the model on the initial test set from the train/test split

	Precision	Recall	F1-score	Support
Background (0)	0.94	0.94	0.94	247
Cluster Member (1)	0.84	0.84	0.84	88
Accuracy	-	-	0.92	335
Macro Average	0.89	0.89	0.89	335
Weighted Average	0.92	0.92	0.92	335

Table 3: Classification metrics from the model on the independent test set

To further explore the model's performance, we looked at the model's behavior in the training set split between the blue cloud, red sequence, and a region we called the "red shoulder", where galaxies would appear redder but not within the estimated boundaries of the red sequence. We note that while the model is able to achieve the metrics displayed in Table 3

8 Application to New Data and Future Work

Given the good performance of our model on data trained on not only Perseus Cluster galaxies but galaxies that fall on or near the red sequence, it is possible the model will perform well on new higher resolution data. The method is also planned to be applicable on much higher resolution imaging data from the Subaru telescope. The new imaging from the European Sky Agency (ESA) covers the Perseus Cluster, and when their higher resolution images are able to be extracted/downloaded our model should be tested against them.

Furthermore, we hope our model will be able to correctly classify Ultra-Diffuse Galaxies (UDGs), as well as dwarf galaxies. Since these are incredibly low-luminosity galaxies, they are difficult to classify by conventional methods, which would be an excellent application of our model. Deep Subaru / Hyper Suprime-Cam imaging of these objects exists and would

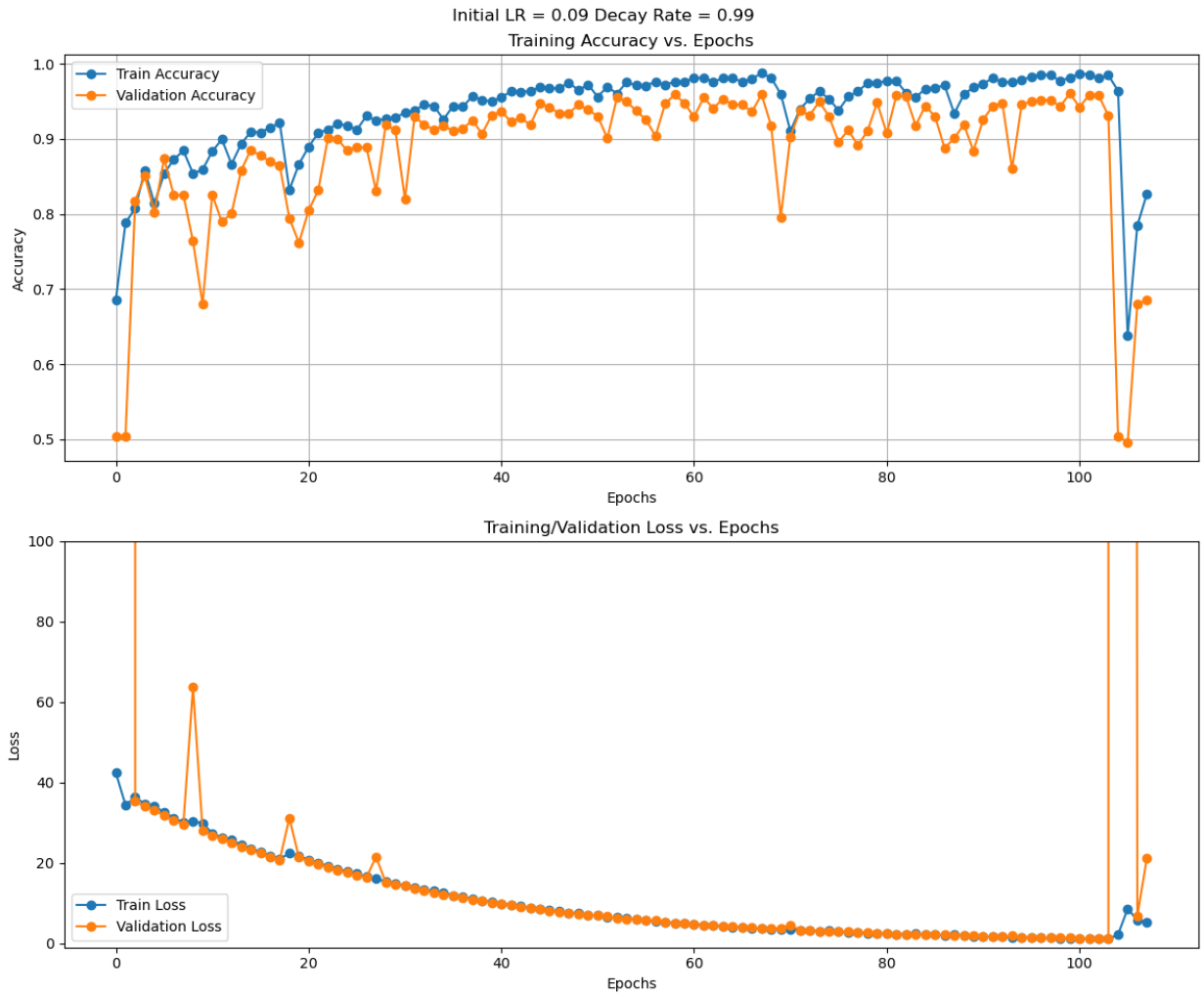


Figure 31: Training report of the model performance on the PCC and Spectroscopic objects

provide a test of model robustness.

9 Conclusion

We conclude that the application of a deep learning model, namely ResNet-50 provides good results for determining galaxy cluster membership. This should eliminate the need for costly endeavors such as finding the spectroscopic redshift data for every object, and instead requiring the much easier task of running our model on a given image of the object. We note that the model learns the relevant features from RGB color, as well as not only the color-magnitude relationship of the red sequence, but the features of edge cases such as galaxies that are cluster members but off of the red sequence.

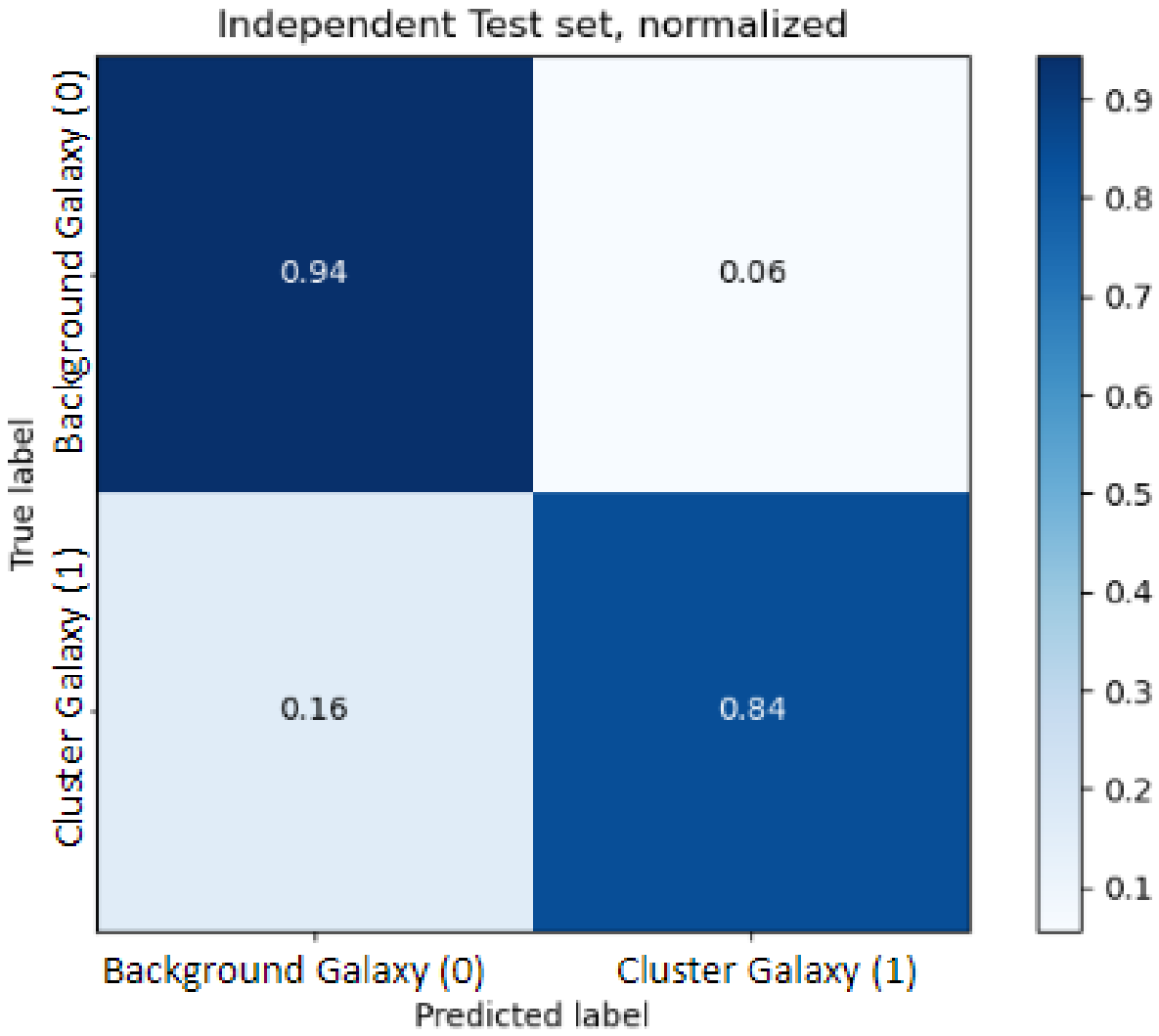


Figure 32: Normalized confusion matrix of the model performance on the independent test set

Blue Cloud	Precision	Recall	F1-score	Support
Background (0)	1.00	0.99	0.99	89
Cluster Member (1)	0.50	1.00	0.67	1
Accuracy	-	-	0.99	90
Macro Average	0.75	0.99	0.83	90
Weighted Average	0.99	0.99	0.99	90
Red Sequence	Precision	Recall	F1-score	Support
Background (0)	0.85	0.88	0.87	78
Cluster Member (1)	0.87	0.83	0.85	71
Accuracy	-	-	0.86	149
Macro Average	0.86	0.86	0.86	149
Weighted Average	0.86	0.86	0.86	149
Red Shoulder	Precision	Recall	F1-score	Support
Background (0)	0.97	0.95	0.96	80
Cluster Member (1)	0.78	0.88	0.82	16
Accuracy	-	-	0.94	96
Macro Average	0.88	0.91	0.89	96
Weighted Average	0.94	0.94	0.94	96

Table 4: Classification metrics from the model on the independent test set in each color region.

A Code and Data Availability

The software is uploaded to Github at `https://github.com/Json-To-String/JPAstro`. The image data and code required to run (and produce new runs) are contained in the above repository as well.

B SQL Queries

```
%% Query for searching out 45 arcmins to acquire red sequence objects
SELECT TOP 100
    p.objID, p.ra, p.dec,
    p.modelMag_r,
    p.modelMag_u - p.modelMag_g as u_g,
    p.modelMag_g - p.modelMag_z as g_z,
    p.modelMag_g - p.modelMag_r as g_r,
    p.modelMag_g - p.modelMag_i as g_i,
    p.modelMag_r - p.modelMag_i as r_i,
    p.modelMag_r - p.modelMag_z as r_z,
    p.petroRad_r, p.flags, dbo.fPhotoFlagsN(p.flags) as flag_text,
    s.specObjID, s.z, s.zErr, s.zWarning, s.class, s.subClass
FROM
    photoObj as p
JOIN SpecObjAll s ON p.objID = s.bestObjID
JOIN dbo.fGetNearbyObjEq(49.9467, 41.5131, 45) as N ON N.objID = p.objID
WHERE
    p.modelMag_r > -14.3*(p.modelMag_g - p.modelMag_z) + 37
    and p.modelMag_r < -14.3*(p.modelMag_g - p.modelMag_z) + 46
    and p.type = 3

%% Query for searching out 90 arcmins to acquire final independent test set
SELECT TOP 1000
    p.objID, p.ra, p.dec,
    p.modelMag_r as R_mag,
    p.modelMag_r - p.extinction_r as r0,
    p.modelMag_g - p.extinction_g - p.modelMag_z + p.extinction_z as g_z0,
    p.modelMag_u - p.modelMag_g as u_g,
```

```
    p.modelMag_g - p.modelMag_z as g_z,
    p.modelMag_g - p.modelMag_r as g_r,
    p.modelMag_g - p.modelMag_i as g_i,
    p.modelMag_r - p.modelMag_i as r_i,
    p.modelMag_r - p.modelMag_z as r_z,
    p.petroRad_r,
    s.specObjID, s.z, s.zErr, s.zWarning,
    N.distance
FROM
    photoObj as p
JOIN SpecObjAll s ON p.objID = s.bestObjID
JOIN dbo.fGetNearbyObjEq(49.9467, 41.5131, 90) as N ON N.objID = p.objID
WHERE
    p.type = 3
ORDER BY distance
```

References

Abraham, R. G. and van den Bergh, S.: 2001, *Science* **293(5533)**, 1273

Ahumada, R. et al.: 2020, *The Astrophysical Journal Supplement Series* **249(1)**, 3

Arnold, L., Rebecchi, S., Chevallier, S., and Paugam-Moisy, H.: 2011, in *The European Symposium on Artificial Neural Networks*

Bakich, M. E.: 2021, *How to Build a Galaxy*, `https://www.astronomy.com/science/the-beginning-to-the-end-of-the-universe-how-to-build-a-galaxy/`

Becker, B., Vaccari, M., Prescott, M., and Grobler, T.: 2021, *Monthly Notices of the Royal Astronomical Society* **503(2)**, 1828

Blanton, M. R. et al.: 2017, *The Astronomical Journal* **154(1)**, 28

Brownlee, J.: 2020, *A Gentle Introduction to the Rectified Linear Unit (ReLU)*, `https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/`

Buetti-Dinh, A. et al.: 2019, *Biotechnology Reports* **22**, e00321

de Vaucouleurs, G.: 1958, *Astrophysical Journal* **128**, 465

Dieleman, S., Willett, K. W., and Dambre, J.: 2015, *Monthly Notices of the Royal Astronomical Society* **450(2)**, 1441–1459

Fukugita, M. et al.: 1996, *The Astrophysical Journal* **111**, 1748

Gannon, J. et al.: 2021, *Monthly Notices of the Royal Astronomical Society* **510(1)**, 946–958

Gavazzi, G., Fumagalli, M., Cucciati, O., and Boselli, A.: 2010, *Astronomy and Astrophysics* **517**, A73

Gladders, M. D. and Yee, H. K. C.: 2005, *The Astrophysical Journal Supplement Series* **157(1)**, 1–29

Graves, G. J., Faber, S. M., Schiavon, R. P., and Yan, R.: 2007, *The Astrophysical Journal* **671(1)**, 243–271

Guo, T., Dong, J., Li, H., and Gao, Y.: 2017, in *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pp 721–724

He et al.: 2016, in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp 770–778

Hubble, E. P.: 1927, *The Observatory* **50**, 276

Jiang, H. and Learned-Miller, E.: 2017, in *2017 12th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2017)*, pp 650–657

Khalifa, N. E. M. et al.: 2017, *CoRR* abs/1709.02245

Lintott, C. J. et al.: 2008, *Monthly Notices of the Royal Astronomical Society* **389(3)**, 1179

Press, W. H.: 1989, *Numerical recipes in Pascal: The Art of Scientific Computing*, https://archive.org/details/numericalrecipes0000unse/page/450/mode/2up

Pröve, P.-L.: 2017, *An Introduction to Different Types of Convolutions in Deep Learning*, https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d

Read, R. J.: 2009, *Applications of the convolution theorem*, https://www-structmed.cimr.cam.ac.uk/Course/Convolution/convolution.html#apps

Sales, L. V., Wetzel, A., and Fattahi, A.: 2022, *Nature Astronomy* **6**, 897

Sandage, A.: 1961, *The Hubble Atlas of Galaxies*, Vol. 681, Carnegie Institution of Washington Publication

Shi, F., Xu, Z., Yuan, T., and Zhu, S.-C.: 2019, *HUGE2: a Highly Untangled Generative-model Engine for Edge-computing*

Smith, L. N.: 2018, *A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay*

Smith, S. W.: 1997, *The Scientist and Engineer's Guide to Digital Signal Processing*, http://www.dspguide.com/ch13/2.htm

Willett, K. W. et al.: 2017, *Mon. Not. Roy. Astron. Soc.* **464(4)**, 4176

Wittmann, C. et al.: 2017, *Monthly Notices of the Royal Astronomical Society* **470(2)**, 1512

Wittmann, C. et al.: 2019, *The Astrophysical Journal Supplement Series* **245(1)**, 10