Using deep learning to predict the properties of galaxy major mergers in EAGLE simulations

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
in
ASTRONOMY AND DATA SCIENCE

Author :	Malavika Vijayendra Vasist
Student ID :	s2033046
Supervisor 1:	Maxwell Cai
Supervisor 2:	Simon Portegies Zwart

Leiden, The Netherlands, June 2019

Using deep learning to predict the properties of galaxy major mergers in EAGLE simulations

Malavika Vijayendra Vasist

Leiden Observatory, Leiden University
P.O. Box 9500, 2300 RA Leiden, The Netherlands

June 2019

Abstract

Galaxy mergers impact the evolution of galaxies by contributing to their mass growth and change in morphology thus motivating us to study them. Observational images capture mergers at a single instant in time making it hard to interpret their properties. Hence, we must resort to indirect means of assessing them by comparison with simulations. Simulations provide an all round quantitative understanding of galaxy mergers, their properties and impact on evolution. The idea is to utilize simulation data to infer observed galaxy merger properties. In this thesis, we train a Deep Neural Network model on galaxy merger images generated from EAGLE simulations with their corresponding properties namely size and mass ratio. We successfully generate two image sets of data for galaxy mergers, at z=0 and 20>z>0 separately, using two zooming techniques namely- the EAGLE package and a self written zooming algorithm. The training results in an accuracy of 85% and 80% on the datasets z=0 and 20>z>0 for mass ratio and 90% and 70% for size ratio respectively. Considering that similar accuracies are achieved, we imply that the visualization techniques aren't crucial to the training, suggesting that the model is robust. We also imply from the high accuracies achieved that Deep Neural Networks are an effective tool in studying galaxy mergers. Further, we conclude that with higher accuracies achieved by increasing the resolution of the images, this technique can be used to study observational images of galaxy mergers.

Contents

List of Equations

List of Figures

List of Tables

Chapter 1

Introduction

The event of collision of two galaxies is called a galaxy merger. This is illustrated in the Fig. 1.1. Galaxy mergers influence the properties of the galaxies involved in the merger either through strong gravitational interactions or by friction caused by the collision of gas and dust between them. The effects of mergers on galactic properties like mass, morphology and angular momentum have been explored in literature by means of both observations and simulations.

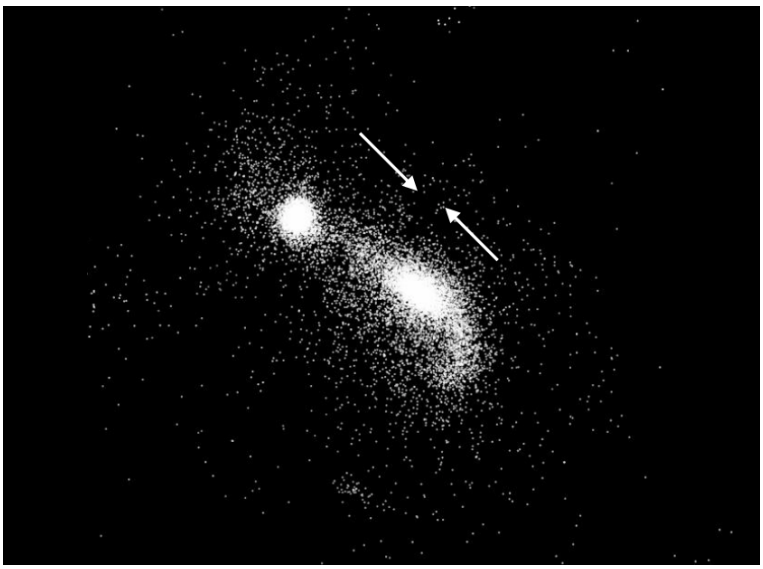


Figure 1.1: *The above image is a visualization of a galaxy merger in EAGLE simulations.*

1.1 Effect of galaxy mergers

Analysis of central galaxies in EAGLE simulations have shown that major mergers (where galaxy mass ratio μ = 1/4)are responsible for most mass growth in massive galaxies ($11 < \log_{10}M\ [M_{\odot}] < 12$) due to ex-situ stars (stars formed outside the halo of the galaxy) (Qu et al. 2016). It is seen that this accretion of ex-situ stars causes a spheroidal build up in the galaxy leading to a morphological transformation from disk-like morphology to a spheroid-like morphology in more massive ($\log_{10}M\ [M_{\odot}] > 10.5$) galaxies (Clauwens et al. 2018). This morphological transformation is mostly seen in gas-poor mergers (mergers of galaxies with low gas-to-stellar mass ratio), hence causing a reduction in its angular momentum. In gas-rich mergers it is seen that in-situ star formation is triggered that contributes towards disk retention (Rodriguez-Gomez et al. 2017- Illustris) hence making mergers inefficient in destroying disks (Kannan et al. 2015- Horizon-AGN) and increasing their angular momentum (Lagos et al. 2017- EAGLE simulations).

MUSE (Deep Multi Unit Spectroscopic Explorer- Ventou et al. 2017) uses observations in the Hubble Ultra Deep Field (HUDF) and Hubble Deep Field South(HDF-S) to evaluate the significance of mergers by mapping the evolution of the fraction of pairs with respect to the galaxy population per redshift (as in Qu et al. 2016). It is seen that mergers increase till redshift three and then reduce. This might imply that mergers are linked to the cosmic star formation rate evolution where most morphological transformations occur (eg. MadauDickinson. 2014). All the above stated simulations and the observation suggest that mergers are important in shaping the evolution of galaxies. Hence understanding them might help us understand galaxy evolution.

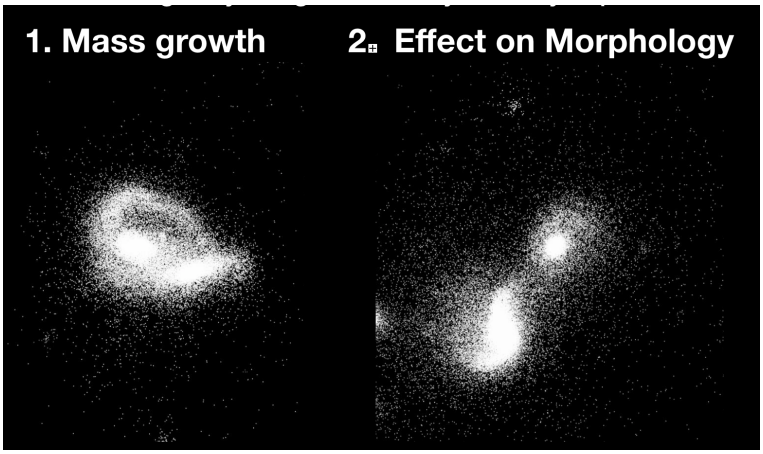


Figure 1.2: *The above image shows the two most prominent effect of mergers on galaxy evolution, namely mass growth due to accretion and change in morphology due to accretion and impact of collision, as visualized in EAGLE simulations.*

1.2 Detection of galaxy mergers

Galaxy mergers have been detected by close pairs method (Woods and Geller 2007) where galaxy mergers are implied from algorithmically identifying luminosity peaks. Although fairly reliable, this technique fails when the spectrum entails imprecise radial distance measurements due to peculiar velocities. Also the algorithm has a fair chance of missing out on mergers which don't exhibit two distinct luminosity peaks.

Another technique used to identify mergers is using feature detectors on imaging data (Conselice 2003 and Lotz et al. 2004). This technique encounters a difficulty in capturing the whole diversity of mergers as per appearance, making it error prone.

By far the most reliable technique for galaxy merger classification has been crowd sourcing, where the general public manually inspected and classified the morphology of nearly one million galaxies via the internet (Lintott et al. 2008,2010). Although this proves to be highly accurate, it is human resource intensive. To reduce this dependency, one other method that has proven useful is the usage of Neural Networks that mimics the human brain.

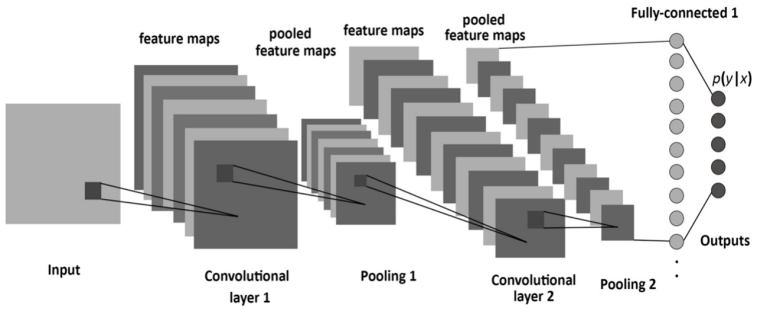


Figure 1.3: *The above image represents a typical Deep Neural Network using Convolutional Neural Networks. Source: https://www.mdpi.com/1099-4300/19/6/242*

1.3 Deep Neural Networks

A Deep Neural Network is an architecture of Deep Learning. It is a class of machine learning algorithm with multiple layers. The deeper the network gets, the more intricate details it is able to extract from the input images. In our Deep Neural Network we use Convolutional Neural Networks and exploit its feature detection abilities. For a long time this has been explored in the computer vision and the machine learning learning community.

Convolutional Neural Networks imitate the connectivity of neurons in the human brain (Hubel and Wiesel 1962, Fukushima et al. 1983). The neurons connected to each other in a hierarchical fashion are capable of pattern recognition. Deeper the layers, the more complex features it can recognize. Convolutional

layers are organized in 3 dimensions namely width-breadth- color. The image is cross-correlated with several filters that is swept across the whole image. Each filter is designed to pick up on a certain feature. Higher the value of cross-correlation, greater is the probability of finding the feature at that location. The layers aren't fully connected and the final layer gives a single dimensional vector entailing the probabilities of it belonging to each class. A visualization of the architecture of a Convolutional Neural Network is shown in Fig. 1.3.

The feature learning aspect of CNNs has been explored in galaxy morphology classification in Dieleman et al. 2015. Here a Deep Neural Network is built using 4 Convolutional Neural Networks and 3 Dense layers. The paper exploits translational and rotational symmetry of galaxies to classify galaxies into one of the six morphological classes namely loose winding arms, edge-on disks, irregulars, disturbed, other, and tight winding arms, based on the annotated images from the Galaxy Zoo project with a near perfect accuracy (> 99 %).

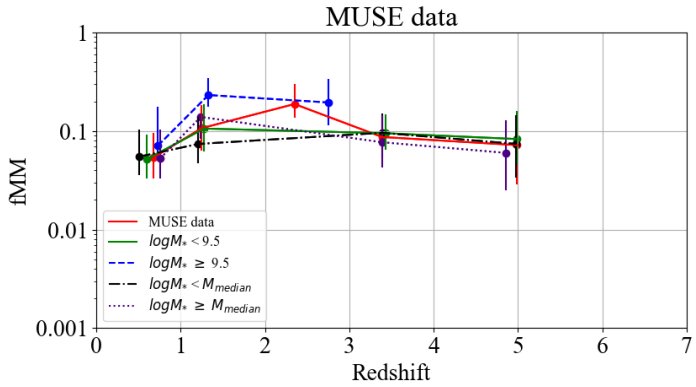


Figure 1.4: *The above figure shows data plotted from MUSE observations. It plots the fraction of major mergers as it varies through redshifts. We see that the fraction of galaxies involved in a merger is less than 20% hence making mergers a rare occurrence.*

1.4 Simulations

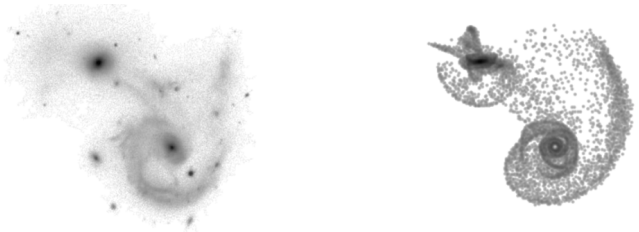


Figure 1.5: *The above two images represent a galaxy merger. On the left is an observational image taken from the Galaxy Zoo mergers project. On the right lies its simulated rendering.*

Simulations are an important tool to study galaxy mergers. Mergers are a rare occurence in nature. This is supported by the Fig. 1.4, which plots the fraction of major mergers as a function of redshift. Fraction of major mergers, as the name suggests, is the fraction of galaxies involved in a major merger per redshift, as gathered from MUSE observations (E.Ventou et al. 2017). We see that the highest fraction achieved is less than 0.3 in log space, suggesting that mergers are a rare occurrence in nature consequently reducing the sample space that can be studied. Thus, by simulating galaxy mergers we can expand the sample space.

Observational images capture mergers at a single instant in time making it hard to interpret their properties. Hence, we resort to indirect means to assess them by comparison with simulations. Simulations provide an all round quantitative understanding of galaxy mergers, their properties and impact on evolution. We utilize simulation data to infer observed galaxy merger properties. Fig. 1.5 shows two images of a galaxy merger. On the left lies the galaxy merger captured by observations in the Galaxy zoo mergers project, and on its right lies the simulated rendering of the same merger (see link in GalaxyZoo). By understanding the properties of the simulated merger, we can safely associate those properties to the actual merger on the left. In this thesis, we use EAGLE simulations (McAlpine et al. 2016 and The EAGLE team 2017) to generate the images of galaxy mergers and extract their corresponding properties .

1.4.1 The EAGLE simulations*

EAGLE stands for Evolution and Assembly of Galaxies and their Environments. This simulates the evolution of dark matter, gas, stars from redshift z=127 to present day at z=0. It is based on the LCDM cosmology with parameters derived from the data of Planck Collaboration (2014). The code used to carry this out is a modified version of the parallel N-body smoothed particle hydrodynamics (SPH) GADGET 3 code (Springel et al. 2008; Springel 2005). The modifications done to the SPH solver that EAGLE uses is termed as Anarchy. EAGLE uses several physics subgrid modules in an attempt to capture the physics realistically, namely- radiative cooling and photo-heating (Wiersma, Schaye and Smith 2009), star formation (Schaye and Dalla Vecchia 2008), stellar evolution and enrichment (Wiersma et al 2009), blackhole growth (Springel et al. 2005; Rosas- Guevara et al. 2015) and Stellar and AGN feedback (Dalla Vecchia Schaye 2012). The parameters for these modules are calibrated with values that fit with observations. The simulation starts with a fixed number of Dark Matter(DM) particles. An algorithm called the 'Friends-of-Friends' (FoF) algorithm (Davis et al 1985) is used to identify bound structures of DM particles. This algorithm considers all the DM particles closer than 0.2 times the mean DM distance to be in the same clump called a 'halo'. Once DM clups are identifed, all baryonic structures that are gravitationally bound are identified within the halo through an algorithm called SUBFIND algorithm (Springel et al 2001; Dolag et al 2009). These structures are called 'subhaloes'. Most subhaloes

*The description of this section is adapted from the Thesis - The relation between galaxy morphology and merger history in the EAGLE simulations, by MVV of Leiden University, 2018

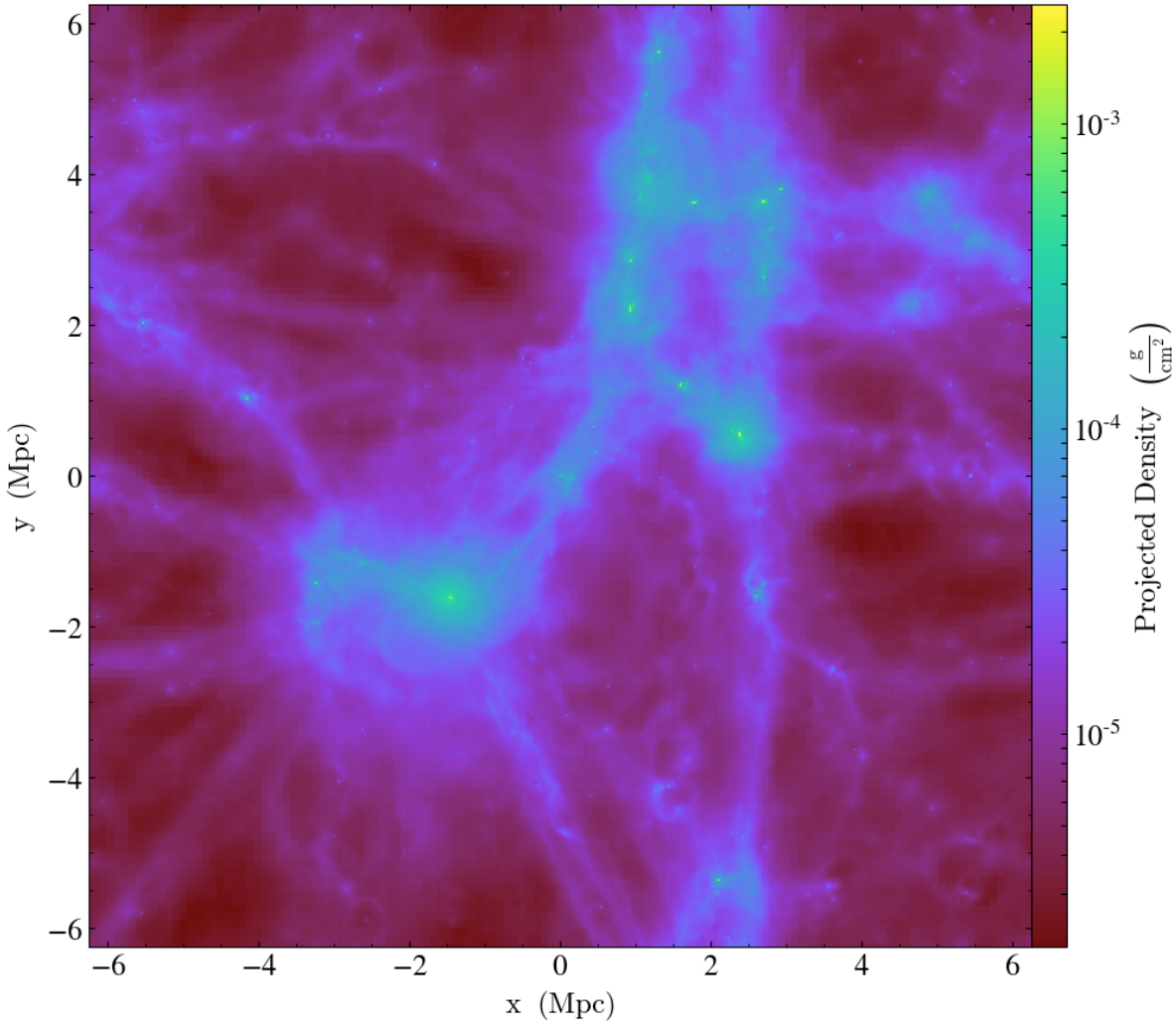


Figure 1.6: *What we see is the visualization of EAGLE. A density plot of all the particles in the simulation L0012N0188 within a box of length 12 Mpc at redshift zero is plotted here. Image Courtesy: Dr Camila Correa, Leiden University.*

in EAGLE contain galaxies. The centre of each of the subhalo is defined by the local minima in the gravitational potential field. The subhalo with the lowest minimum compared to all other subhaloes in the halo is considered the main subhalo. The galaxy in the main subhalo is defined as the 'central galaxy'. The rest of the galaxies in the halo are classified as 'satellites'. Of the several simulations of various cosmological volumes and resolutions run by EAGLE, this thesis uses the reference model, Ref-L100N1504 run in a co-moving box of size 100 cMpc comprising of an equal number (1504^3) of gas and DM particles in the beginning of the simulation where the gas and DM particles have a mass of 1.81×10^6 $M_\odot$ and 9.7×10^6 M$\odot$ each respectively. The simulation also assumes a softening length of 0.7 proper kpc. As the simulation progresses, the state of the system is stored at 29 time instants (snapshots) from z=20 to z=0. This thesis is aimed at exploring what the higher resolution of the simulation can reveal about galaxy mergers in all epochs. The table below shows the parameters of the simulation used here.

Parameter	units	value
box length	cMpc	100
# of particles		2×1504^3
gas/star particle mass	M$\odot$	1.81×10^6
DM particle mass	M$\odot$	9.7×10^6

This composite image is produced using projection in the z direction. 'friends of friend' (FoF) algorithm is used to simulate the dark matter clusters, and a 'subfind' algorithm is used to fill them up with baryonic particles based on which DM particles (within a cluster) they are gravitational bound to. The color scale shows the variation of density in the plot. The bright green spots seen here are the most dense regions of the simulation. This plot is centered at the maximum potential.

1.5 This Thesis

In this thesis, we want to see if we can push the potential of Deep Neural Networks to learn beyond morphological classification of galaxy mergers and extract some its properties from the images of galaxy mergers derived from the EAGLE simulations.

In chapter 2, we define major mergers for a chosen distance and mass criteria and identify the galaxies involved in major mergers by their Galaxy IDs throughout all redshifts in EAGLE simulations. We extract their corresponding particle data using the a) Eagle package (z=0) and b) a self written code (20>z>0) , and plot them in Python using the Matplotlib package hence visualizing the major mergers. We successfully visualize 320 mergers (7680 merger images). We also discuss how the images can be improved in the future for a more realistic rendering of galaxy mergers. Once we have the images, we explore the properties that can be extracted from these images to be used as labels. We justify why the mass and size ratio were picked and discuss how they were extracted from the simulation data. We conclude with suggestions to improve the labeling of the images.

Version of June 2019– Created June 30, 2019 - 16:21

In chapter 3, we discuss the implementation of the Deep Neural Network used to predict the properties (namely mass and size ratio) from the images of galaxy mergers that we have generated from EAGLE simulations. We introduce the software TensorFlow and walk through the process of building a low-level and a high-level TensorFlow based data-pipeline. We discuss various data augmentation techniques that were performed on the generated images. We visualize the model and discuss its architecture.

In chapter 4, we elaborate on the training procedure and the distribution of the data into training, testing and validation on both datasets (z=0 and 20>z> 0). We discuss the measure of evaluation used. We analyze the technical aspects of the network and justify the choice of the batch size, accuracy measure, optimizer, learning rate and momentum. We highlight the computing resources and libraries that were used in building and training the Deep Neural Network.

In chapter 5, we discuss the results obtained from training the network on both the datasets (z=0 and 20>z> 0). This is done by plotting the progress of training and validation accuracy with the increasing number of epochs along with the evolution of the training and validation loss.

In Chapter 6, we discuss some insights gathered from the results obtained. This is followed by Conclusion in Chapter 7. Chapter 8 is devoted to future work that can be pursued.

Chapter 2

Data pre-processing

In this chapter, we discuss how the galaxy merger images were generated from EAGLE simulations. We define major mergers and identify the galaxies involved in a major merger by their Galaxy ID in the EAGLE simulations. We extract the stellar particle data of the corresponding galaxies using the 1) EAGLE package (for z=0) and 2) self written code (for $20>z>0$), and plot them using the python's matplotlib package. We also explore the properties of the galaxy mergers that can be implied from the merger images by the neural network. Lastly, we suggest improvements to the image generation process and labelling to ensure a more realistic rendering of galaxy mergers in the future.

2.1 Visualizing galaxy mergers

The galaxy mergers are visualized from EAGLE simulations to be fed into the Deep Neural Network. Only stellar particles are considered for visualization as it is more visually apparent in observational images of galaxy mergers. Fig. 2.1 gives an example of the visualized galaxy mergers from the EAGLE simulations.

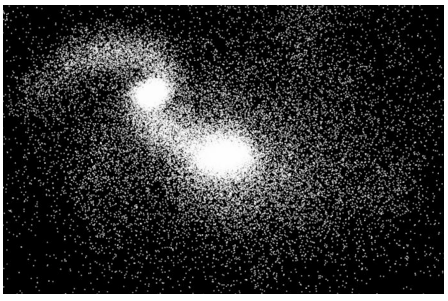


Figure 2.1: *Visualizing galaxy mergers using EAGLE simulations*

2.1.1 Defining Major Mergers

We define mergers by applying the technique of 'close pairs'. We believe that spatially close galaxies are gravitationally bound to each other and are eventually destined to collide. This makes close pairs a good estimate of mergers. We arrive at an appropriate proximity threshold, the distance criteria, to identify close pairs. We quantify the size of a galaxy by its half mass radius (HMR), defined as the distance from its centre of potential encompassing half the stellar mass of the galaxy. In an effort to choose an appropriate distance criteria, we start by mapping the distribution of sizes of all the galaxies in the simulation versus its stellar mass in Fig. 2.3. We see that the half mass radius for the galaxy distribution in EAGLE simulations is between 5 to 40kpc. As the sizes of the galaxies follow a wide range, we define the distance criteria to depend on its size. Thus, we choose our distance criteria to be $10\times$ HMR. The number 10 is incidental. This makes our distance criteria vary between 50 to 400kpc. The figure is plotted with error bars indicating the 25th and 75th percentiles.

Once close pairs are identified, they are considered as mergers and classified as 'major' or 'minor' depending on the ratio of the stellar masses of the galaxies constituting the pair. If the galaxy under consideration (M_1) is in close pairing with a neighbouring galaxy (M_2) and their stellar mass ratio μ (Mu) = M_2/M_1 where $M_1 > M_2$, is greater than a predefined mass criteria of $1/6$, it is considered to be a major merger. $1/6$ is chosen to match observations (E.Ventou et al. 2017). It also provides a suitable aspect ratio to the galaxy merger images making it easier for the network to learn from.

Hence, any two galaxies that lie within the distance criteria of $10\times$ HMR and has a mass ratio greater than the mass criteria of $1/6$, is considered be to a major merger. The assumptions are illustrated in the Fig. 2.2 below.

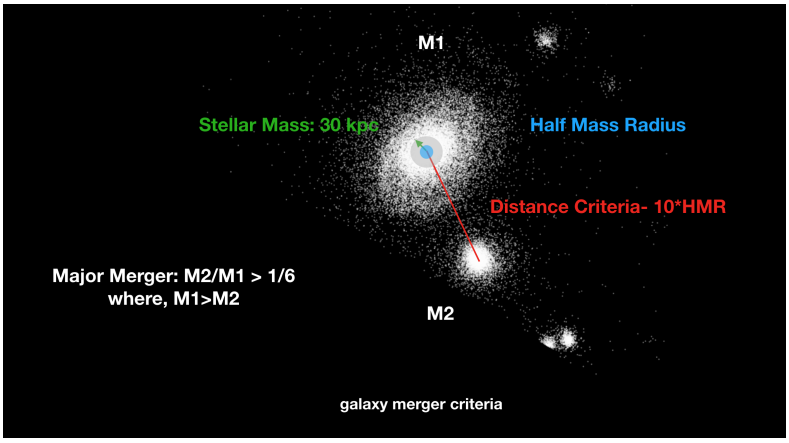


Figure 2.2: *The above figure shows the assumptions used to visualize galaxy major mergers in EAGLE simulations.*

Version of June 2019– Created June 30, 2019 - 16:21

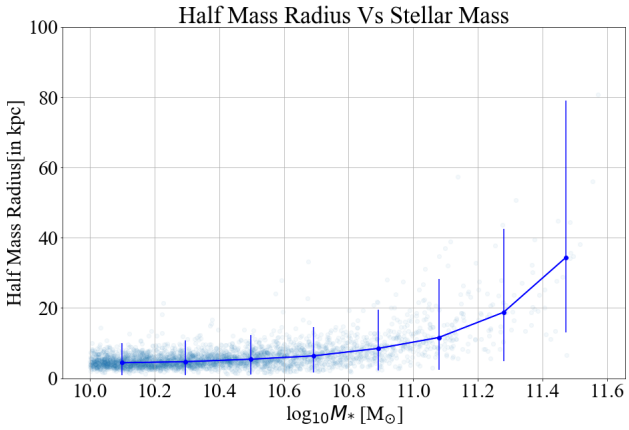


Figure 2.3: *Half Mass Radius is plotted against Stellar Mass for all the galaxies of the simulation. The error bars signify 25 and 75 percentiles. It can be seen that much of the galaxies have a HMR less than 40kpc hence making it a good proximity threshold to identify close pairs.*

2.1.2 Visualization

We download the main galaxy information namely the Galaxy ID, Group Number, Stellar Mass, Half Mass Radius and the x,y and z co-ordinates of the Centre of Potential from the tables Ref-L100N1504_Subhalo and Ref-L100N1504_Aperture for galaxies that have a stellar mass of 10^9 $M_\odot$ (mass threshold) or higher, measured with an aperture size of 30pkpc or greater, from the EAGLE database (EAGLE database) per redshift. As our aim is to realistically visualize galaxies, a high number of particles per galaxy is desirable. With the resolution chosen (Ref100N1504), each star particle weighs 1.81×10^6 $M_\odot$ implying that for galaxies possessing a stellar mass of 10^9 $M_\odot$ or higher, we obtain 1000 star particles or more. Since we see that this is a reasonable number of particles to visualize galaxies realistically, we enforce this mass threshold.

The Galaxy ID is a unique identifier of each galaxy in the simulation, the Group Number is an integer number used to identify FoF(Friends of Friends) halo per snapshot(redshift), the Stellar Mass is the combined mass of all the star particles in each galaxy in solar mass units, the Half Mass Radius is the physical radius enclosing half of the stellar mass in pkpc, and the Centre of Potential co-ordinates in 3-D imply the co-moving position of the minimum of the gravitational potential defined by the position of the most bound particle in the subhalo.

Once the above mentioned information is downloaded from the database, galaxies involved in a major merger are identified based on the previously mentioned distance criteria of $10\times$HMR and mass criteria of 1/6. We find 1472 mergers throughout all the redshifts(z=0: 126 and $20>z>0$: 1346). For each merger, the corresponding halo encompassing them is identified by its Group Number. Here we call them merger haloes. Once the stellar particles belong-

ing to all the haloes are downloaded from the particle database of EAGLE simulations (EAGLE particle database) per redshift, only those belonging to the merger haloes are extracted and stored separately in csv files for further plotting.

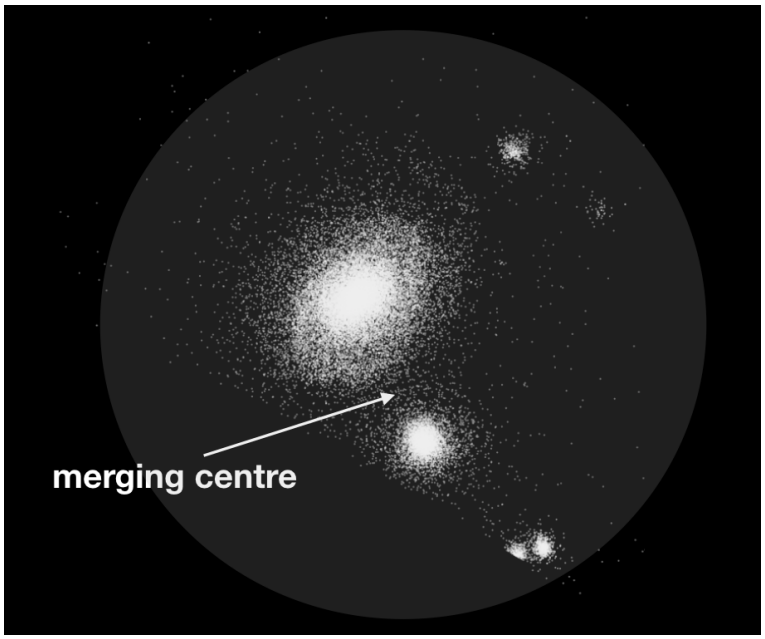


Figure 2.4: *The above image shows the galactic merger centre of two galaxies enclosed within a dark matter halo visualized in the EAGLE simulations.*

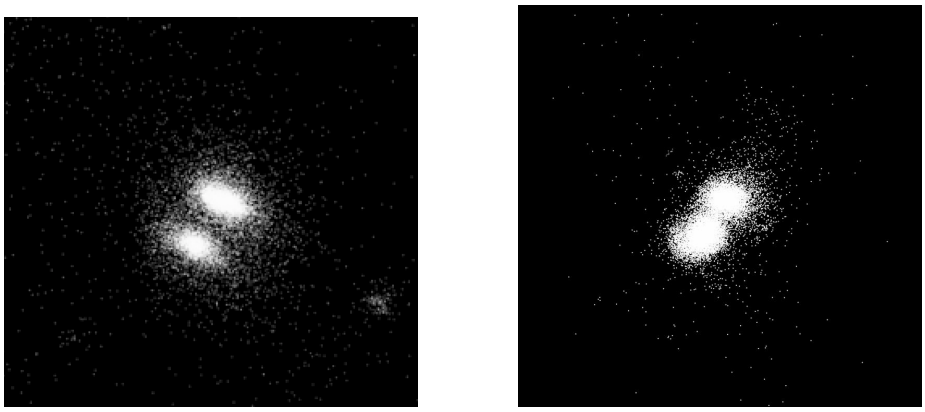
Each merger halo contains stellar particles pertaining to all the galaxies in the halo including the satellites. In order to image only those pertaining to galaxies involved in a major merger(ignoring the satellites), we estimate the mid point between the merging galaxies using kernel density estimation and zoom into it. Kernel density estimation is a way to estimate the probability density function (PDF) of a random variable in a non-parametric way. Using gaussian kernels, we estimate the two most dense points in the halo, which mostly pertains to the two centres of the galaxies involved in a merger, and calculate their centre. This mid-point is assumed to be the galaxy merger centre. This is illustrated in the Fig. 2.4.

In this thesis, we use two techniques to zoom into the image of galaxy mergers that result in two datasets of images. The first technique involves setting the x and y limit on the axes and the second technique involves using the EAGLE package. In total we generate 320 merger images (39: z=0, 281: 20>z>0). We use two different techniques because we find that when zoomed in at the calculated apparent merging centre, the EAGLE package doesn't capture the whole diversity of galaxy mergers. Nevertheless, we also find that the EAGLE package extracts particle data in a much time and memory efficient way, thus inspiring us to accommodate both types of data visualization in the thesis. Eventually we will show that the visualization technique plays little role in training as both datasets achieve similar accuracies.

2.1.3 Zooming by setting a limit on the axes, 20>z>0

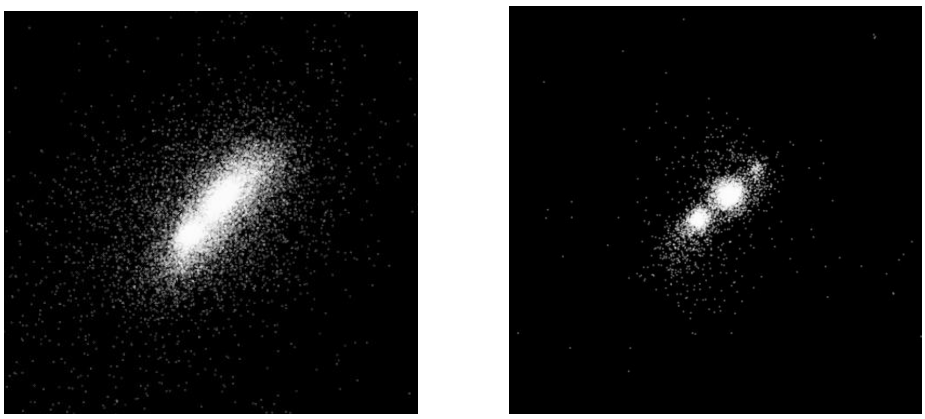
This technique is adopted for galaxy mergers with a redshift 20>z>0. The stellar particle data belonging to the merger halo is plotted using the scatter plot function in matplotlib.pyplot package of python. The 3-dimensional cube is projected onto 2 dimensions at a 0 degree elevation and azimuth angle. The merging galaxies are captured by zooming into the merger halo at the calculated merger centre by setting the x and y limit of the plot. From Fig. 2.3, we see that the half mass radius for all galaxies in EAGLE simulations is within 40kpc. We realize that to obtain galaxy merger images with a high aspect ratio, we must zoom into a window of size less than 40kpc×40kpc. Thus, we choose a window size of 35kpc×35kpc by trial and error. We see that this generalizes well to give us 281 mergers for 20>z>0.

Only mergers with stellar particle count less than 200,000 particles are plotted. This not only quickens the process of plotting but also saves significant memory space.



(a) *xy-limit* ***(b)*** *read eagle package*

Figure 2.5: *Example 1- The above figures represent a galaxy merger visualized by the a) xy-limit and the b) read eagle package.*



(a) *xy-limit* ***(b)*** *read eagle package*

Figure 2.6: *Example 2- The above figures represent a galaxy merger visualized by the a) xy-limit and the b) read eagle package.*

2.1.4 Zooming via the EAGLE package, z=0

Galaxies are known to grow in size as they evolve. Consequently more recent redshifts have a higher occurrence of more massive galaxies comprising a high number of star particles making their imaging- memory and time intensive. To tackle this issue, we image galaxy mergers of redshift 0 (snapshot 28), using the read eagle package. The Read EAGLE package (see Read EAGLE package) scans only through the required spatial regions of the simulations without having to read all the data. This proves to be time efficient. It works by splitting the simulation volume into a grid. Particle data belonging to a particular grid cell are stored consecutively in the snapshot files. Some extra datasets are added to the snapshot files which specify which grid cells are stored in which files and the location of the start of each cell in the file.

We feed snapshot 28 into the package (using the function `EagleSnapshot`) and select a region in the EAGLE simulation space, centering on the calculated merger centre (using the function `select_region`) capturing a window of 25kpc$\times$25kpc. We read the stellar particle dataset from this region. The window size was chosen incidentally by trial and error. It is 10kpc smaller than the previous case since the diversity we see in one redshift(here z=0) is lesser. Once the particles are downloaded, we plot them using the scatter plot function in matplotlib.pyplot package of python similarly as above. Mergers that the eagle package fails to capture are captured by the previous technique of setting a limit on the axes as explained above. All the images generated are black and white.

Fig. 2.5 and Fig. 2.6 show a comparison of two galaxy mergers each visualized from the above two mentioned zooming techniques.

2.2 Improvements in visualization

We suggest a few improvements to the visualization procedure that could improve the quality of the images generated from the EAGLE simulations.

- The current images are plotted using a default resolution of 72 DPI and a matplotlib figure size of (10,10), resulting in images with size 770×775 pixels. The galaxy merger images with a higher DPI can be saved to obtain a higher resolution on them.
- All the stellar particles are plotted with equal brightness. Setting a brightness gradient might render more realistic images as star particles towards the galactic centre are usually brighter.
- The zooming can be customized to the sizes(HMR) of the galaxies involved in the merger.
- The projection from 3D to 2D in generating the images is carried out at 0 degrees azimuth and elevation in the python package matplotlib.pyplot. Instead, the projection angle can be customized to orient along the combined angular momentum of the two galaxies, hence capturing a face-on alignment of the mergers enclosing more information about them.
- The EAGLE package is used only for the redshift 0 (snapshot 28). Using it for all the redshifts will make the process picky but quick.
- Here we generate only black and white images. Simulating colored images might improve the clarity of mergers and hence generate more realistic images.

2.3 Progenitor properties

Galaxy mergers play a crucial role in galaxy evolution. Observationally we see only a snapshot and donot see how the whole merger enfold in front of our eyes, thus we try to understand how the mergers affect the fate of the galaxies by indirect means. One way is to understand the properties of the progenitor galaxies involved in the merger. Once we generate the galaxy merger images in EAGLE simulations, we inspect them to explore the properties of the progenitors we can deduce from it.

On first glance we notice that the relative sizes of the galaxies involved in the merger is the most visually apparent feature, hence we define that as the size ratio. As the size of a galaxy is implied from its half mass radius, for every merger involving two galaxies with sizes HMR_1 and HMR_2 such that $HMR_1 > HMR_2$, the size ratio is defined as HMR_1/HMR_2. The size ratio is a number between [0,1].

From Fig. 2.3, we see that the size of the galaxies implied by its half mass radius is an increasing function with stellar mass, making the size and its stellar mass directly proportional to each other. Thus we conclude that the stellar mass ratio is also visually deducible like size ratio. Observationally, the estimation of the mass of a galaxy corresponds to its stellar mass at visible wavelengths. This inturn is used to characterize the overall mass of the galaxy. Hence, observationally we can characterize the mass ratio from the stellar mass ratio. Accordingly, we calculate the mass ratio defined in this thesis as the ratio of

the stellar masses of the two galaxies involved in a merger. It ranges between [0,1]. For every merger involving two galaxies with stellar masses M_1 and M_2 such that $M_1 > M_2$, the mass ratio is defined as M_1/M_2. The mass ratio is also a number between [0,1].

Cases where galaxy mergers are in an advanced stage of merging, the friends of friends (FoF) algorithm fails to recognize the merging galaxies independently in the simulations and assigns a combined half mass radius and mass, resulting in a faulty calculated size and mass ratios for the merger. To ensure accurate labeling in this scenario, we trace the progenitors of the identified merging galaxies in the previous snapshot and calculate their size and mass ratio. These values are more accurate as they represent the galaxies before they merged indistinguishably.

Fig. 2.8 shows the distribution of galaxy mergers across mass and size ratio for z=0 and 20>z>0. We see that the labels are balanced across all bins for both properties across all redshifts.

We calculate the size and mass ratios of every galaxy merger and store them as labels to their corresponding merger images. We then feed the merger images and their labels into the neural network and allow it to learn to predict these properties. We hope that in the future, with sufficient accuracy, these properties can be deduced from real galaxy merger images obtained observationally.

2.4 Improvements in labelling

We suggest a few improvements to the labelling of the generated images of galaxy mergers.

- The distance criteria can be explored.
- As seen in the Fig. 2.8 the labels are more or less uniform. They can be further transformed into a uniform distribution using the quantile transformer feature of sklearn.

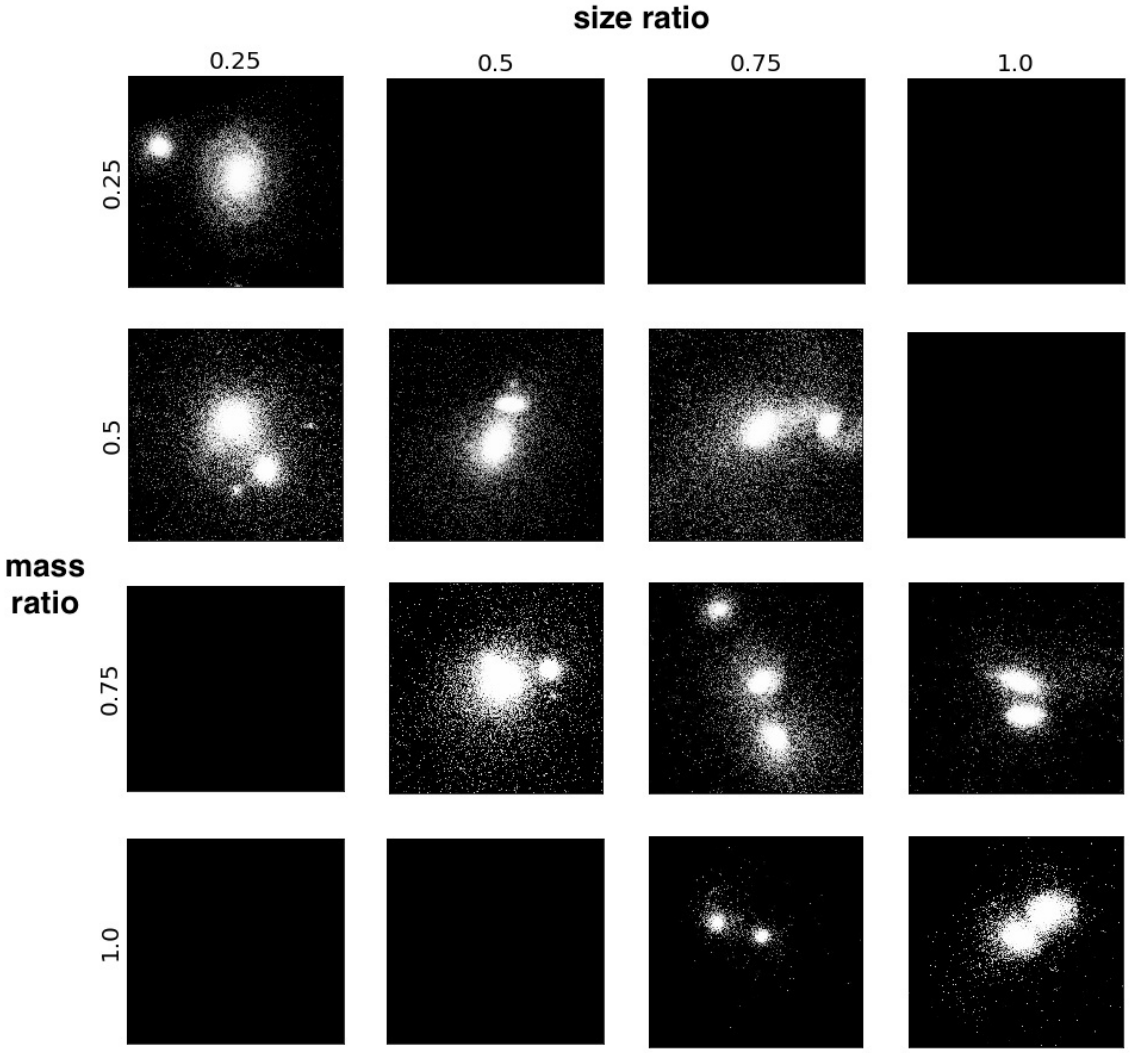


Figure 2.7: *The above figure visualizes galaxy mergers with the different mass and size ratios as indicated.*

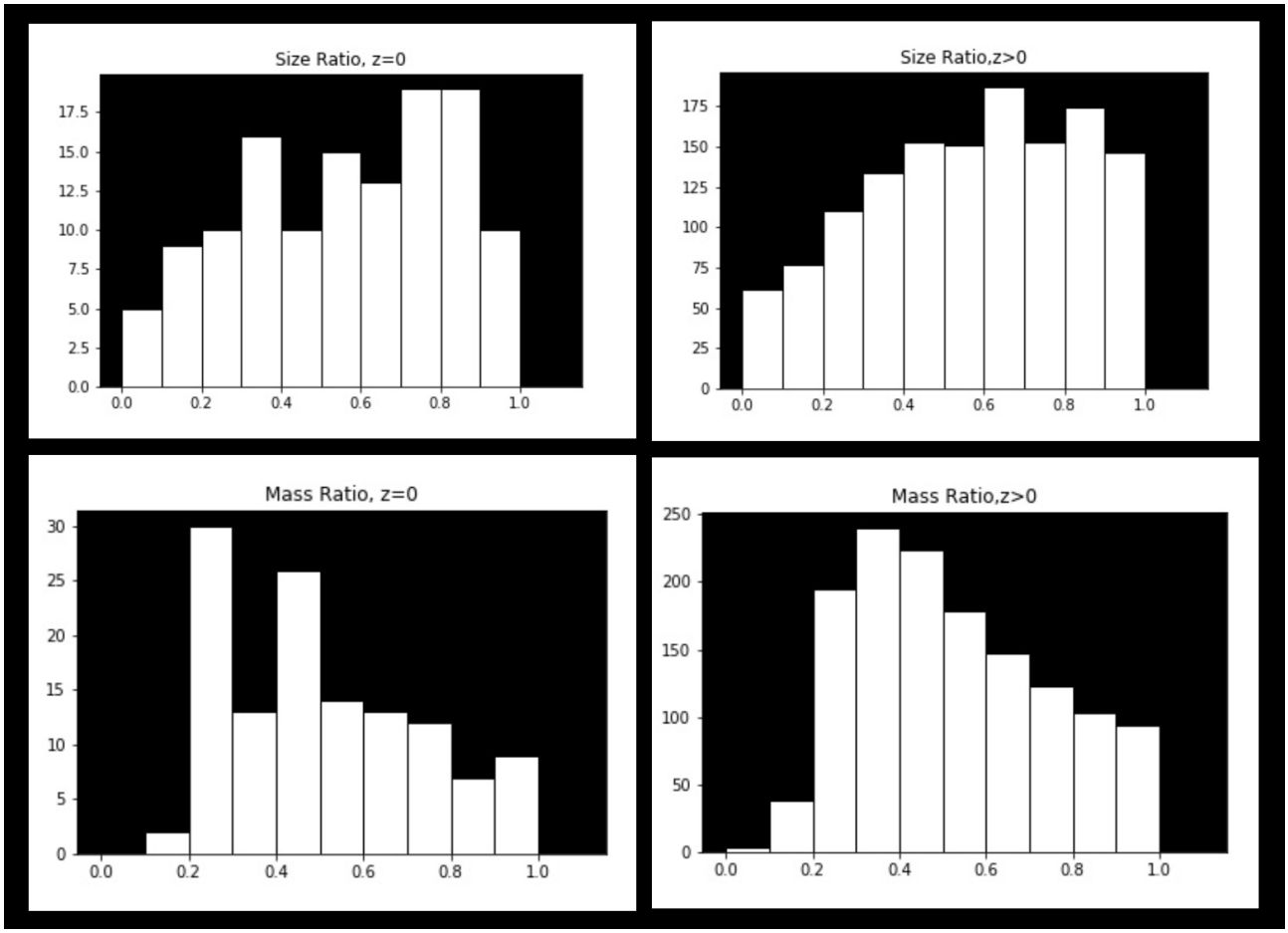


Figure 2.8: *The above figure plots the distribution of galaxy mergers in mass and size ratio as in EAGLE simulations.*

Chapter 3

Implementation

In this chapter, we discuss how the software TensorFlow is used to construct the data pipeline. We discuss its low level and high level APIs, namely TensorFlow core and tf.keras and how they are used to build a dataset to feed into the model. We mention the versions of the packages used. Once we have built the datasets, we discuss the concept of deep learning and its effectiveness. We examine a Deep Neural Network built by keras that was designed for the MNIST dataset and its implementation in this thesis.

3.1 TensorFlow

TensorFlow is a software library by Google used for machine learning applications like neural networks (see TensorFlow). Much of the tasks in TensorFlow are performed using tensors. We have a few thousand images which deems the training process expensive processing-wise. To ensure a high performance, we generate an input piepline that can continuosly feed input data into the model. TensorFlow provides a low-level API (see Application Programming Interface) called TensorFlow core, and high-level API called tf.keras. In this thesis, we will use both the APIs to build our input data pipeline to feed into the neural network.

We use the GPU version of the TensorFlow 1.10.0 for parallel processing and version 2.2.4 of Keras. The python that is compatible to run these packages is version 3.6.8.

3.2 TensorFlow core pipeline

Here we develop a low level TensorFlow core pipeline for the galaxy merger images and its size and mass ratio labels. The key feature of low level API of

TensorFlow is that the operations are defined independently from their execution. The operations are defined in a construct called a graph and the execution is initiated by running a session for that graph.

To build our image dataset, we start by constructing a graph defining the preprocessing operations that we intend to perform on the images, that are input one by one via a placeholder. We run the graph in a session, where the images are processed and returned as tensors. Example messages are used to map the tensors to their labels and feed into a .tfrecord format file. TFRecords is a format for storing a sequence of binary records. They are designed to be used with TensorFlow (see TFRecords). Storing in the binary format is memory efficient and time efficient hence reducing the overall training time of the model (see TFRecords conversion and Prasad.P).

Separate TFRecords are created for training, test and validation set of images. 70% of the images are used for training, the rest 15% each are used for validation and testing. This split in images is conventional. The labels are distributed among 4 classes- 0.25,0.5,0.75 and 1.0. Each set (training, testing and validation) contains 4 TFRecords (one for each class). Each TFRecord holds images and their corresponding labels belonging to that class. The stored labels are the indexes of the class to which they belong.

TFRecords are read by parsing through the `Example` and extracting the image tensors and the labels separately into a dataset. The contents of the dataset are shuffled, repeated and divided into batches. This dataset is iterated over and and fed into the model. (The procedure is based off the code from TFRecords github).

3.3 TensorFlow's Keras

`tf.keras` is a high level API by TensorFlow used for machine learning applications. It is TensorFlow's implementation of the Keras API specification.It is user friendly, modular and composable and easy to extend (see Keras). Here we use `tf.keras` to train the model. The data pipeline is much straightforward as the images and their corresponding labels are stored in numpy arrays. The labels are converted into one-hot encoders before they are stored. The set of images are split into train and test datasets using the sklearn's `train_test_fit` function. Following conventions, 66% of the images are stored in the training set, and the rest are stored in the test set. These arrays are directly fed into the model.

3.4 Data Augmentation

We successfully generate 320 merger images from all the redshifts. As a larger dataset ensures better exposure to the network consequently achieving a larger accuracy on training, we perform certain image augmentation techniques on the merger images generated.

We rotate each merger image by an angle of 15 degrees, until it does a full 360 degrees circle. This is achieved by axis translation using a translation matrix about the centre of the image. This gives us 24 merger images per merger. Further, we horizontally flip each of these images using the horizontal flip parameter in `ImageDataGenerator` function of TensorFlow before fitting it into the model. This function is discussed further in the next section. In total, we are able to generate 15,360 images.

3.5 Deep Neural Network (DNN)

The model used is a Deep Neural Network. Deep Neural Networks are a class of Machine Learning algorithm with multiple layers. Deeper the network, the more intricate details it learns. It resembles the complex neural network in the brain and imitates it in its functionality from driving cars, locating defective cancer cells, in home assistance, translation to image classification.

Image classification has been at the forefront of deep learning applications for a long time. One of the first attempts of image classification was on a large database called the MNIST (Modified National Institute of Standards and Technology) database (see MNIST database). The MNIST database consists of 60,000 training images and 10,000 testing images of handwritten digits. Over the years, several networks have been developed to train these images and accurately predict them. One such Deep Neural Network developed by keras is used in this thesis.

3.6 Model

Keras has built several model examples one can start training their datasets with. In this thesis, we use a deep neural network given by the keras example model. This model is used because it has proven to be very robust since it gives a 99.25% test accuracy just after 12 epochs on the MNIST dataset (see keras github). The choice of the model is incidental.

Our model implementation is shown in Fig. 3.1. It has two 2-dimensional Convolutional Neural Networks with a kernel size of 3×3 that picks up features from the input images. 3 Dropout layers with a constant 0.5 that check over-fitting, a Maxpooling layer of pool size 2×2 to improve computational performance and restrict over-fitting, and a Flatten layer to feed it into a Dense layer for spitting out the class-wise prediction of probabilities. All the activation functions are ReLu except the last one which is softmax. This is so because the softmax activation function gives the probabilities of the image belonging to each class, hence all the probabilities add up to 1. This is a convenient way of classification. The input images of size (224,224,3) are divided into 4 classes namely -0.25,0.5,0.75 and 1.0. Although the images generated are black and white, all the 3 dimensions are considered in order to generalize the images well on colored images for future training. This does not affect the results in

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
conv2d (Conv2D)	(None, 222, 222, 32)	896
dropout (Dropout)	(None, 222, 222, 32)	0
conv2d_1 (Conv2D)	(None, 220, 220, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 110, 110, 32)	0
dropout_1 (Dropout)	(None, 110, 110, 32)	0
flatten (Flatten)	(None, 387200)	0
dense (Dense)	(None, 128)	49561728
dropout_2 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 4)	516

Total params: 49,572,388
Trainable params: 49,572,388
Non-trainable params: 0

Figure 3.1: *The above figure summarizes all the layers and their corresponding array sizes in our deep neural network model.*

any way. The default learning rate is used and no weight decay is considered. The figure also indicates the number of trainable parameters.

Chapter 4

DNN training

In this chapter we discuss how the constructed keras data pipeline is fed into the model using the `model.fit_generator` function of TensorFlow. We examine the training procedure, the choice of the optimizer (see SGD), and the accuracy measure (see categorical crossentropy) used. We discuss ways of saving the model and their weights. We explore how the training and test accuracies and losses can be visualized using Tensoboard and Model Checkpoints. We examine the data augmentation options and justify the hyperparameters used to train the model.

4.1 Compiling

The deep neural network model is compiled using `model.compile` function of TensorFlow. This defines the model with a loss function, optimizer and metrics. Because the labels are one-hot encoded as mentioned above, the loss function used is categorical cross-entropy (see Loss). This is given by the formula:

$$L(true, pred) = -\sum_{x} true(x) log(pred(x)) \tag{4.1}$$

where true(x) represents the one-hot label encoding of the true labels and pred(x) represents the output of the softmax activation function. The average value of the loss per batch is calculated and displayed during training.

The optimizer used is Stochastic Gradient Descent (see SGD and Optimizer). It finds the minima of the loss function and iteratively tries to achieve it. This is carried out by calculating the gradient for the loss function for one sample in the dataset and updating the model parameters in every iteration. This makes it a quick optimizer. The iterations are performed until the optimum model parameters (weights and biases) are achieved and the model leads to convergence

(zero loss). This is called backpropagation. The parameter update is given by the formula:

$$\theta = \theta - \eta.\Delta H(\theta, true(x), pred(x)) \tag{4.2}$$

where, θ is the parameter, η indicates the constant learning rate and ΔL is the gradient of the loss function L. Here we use a learning rate of 0.01. Because of the frequent updates, the parameter updates have a high variance hence causing a lot of fluctuations in the Loss function values. Although this helps to explore better local minima, convergence might not be achieved and it can result in overshooting. This decreases when a low decay rate is introduced to the learning rate. Hence we use a decay of 1e-06. To achieve a faster convergence and reduce fluctuations, we use the momentum method which updates the parameters in a direction of constant gradient. Here we use a high momentum of 0.9. Nesterov momentum is our choice of the momentum because it offers a stronger convergence for convex functions like the loss function considered.

Accuracy measure is taken to be categorical accuracy (see Accuracy). This is given by the formula:

$$Accuracy = mean((max(true) == max(pred))) \tag{4.3}$$

This calculates the rate of mean accuracy across all predictions for multi-class classification problems.

4.2 Training and Testing

Once the model is compiled, we call the `model.fit_generator` (see Fit generator) to train the model. This accepts the dataset batchwise, performs backpropagation and updates the model parameters until the desired number of epochs is reached.

We supply it with the generator function, `ImageDataGenerator`, of TensorFlow that allows for the images to be fed into the model in batches in an infinite loop. To assess the completion of one epoch, the quantity `steps_per_epoch` is defined. Skimming through all the images in the dataset in batches once, marks the completion of an epoch. Hence `steps_per_epoch` is given as $train(test)\_images/batch\_size$ (here 10). The training and testing arrays along with their one-hot labels are input into the function. We choose the batch size as 64 and 200 as the number of epochs. Both these values are chosen incidentally depending on the size of the dataset that constitutes 660 training and 330 testing images.

We use Tensorboard visualisation tool to visualise the progress of the training via monitoring the training accuracy and loss, and validation accuracy and loss. These results are shown in the next chapter. We store the updated model

parameters in a hdf file after each epoch using the Model Checkpoints tool of TensorFlow. Tensorboard and ModelCheckpoints are input to Callbacks parameter of the `model.fit_generator` function.

4.3 Computing resources

We use the following resources to generate images from EAGLE simulations and train the model. We used 3 systems and the super computer called the Little Green Machine 2. Little Green Machine 2 has 4 nodes with 4 16 GB GPUs each. It uses the OpenPower architecture developed by IBM. The other 3 machines used were titan2, pisces and gerelingsplas each having a x86_64 architecture and RAM memories of 125, 62 and 5 GB respectively.

4.4 Software

The whole project is coded in python 3.6.8. All the packages are downloaded using Anaconda 3. The table shows summarizes the packages used for carrying out the operations in this thesis:

Package	Version
TensorFlow GPU	1.10.0
Keras	2.2.4
Sklearn	0.19.1

Table 4.1: *Packages used and their versions*

Chapter 5

Results

In this chapter, we discuss the results obtained from training on the keras data pipeline. In each case we train the model to predict over 4 classes namely 0.25,0.5,0.75 and 1.0. To ascertain how well the model is learning, we plot the training accuracy and loss, and validation accuracy and loss as a function of the number of epochs. Accuracy is calculated using the equation in 4.3 and the loss is calculated using the equation in 4.1. Below we summarize the hyper-parameters of the model:

Compiler parameter	type
Accuracy	categorical accuracy
Loss	categorical crossentropy
Optimizer	Stochastic Gradient Descent

Table 5.1: *Compiler parameters*

Optimizer Parameter	value
Learning rate	0.01
Decay	1e-06
Momentum	0.9
Nesterov	True

Table 5.2: *Optimizer parameters*

Model fit parameter	value
batch size	64
steps per epoch	10
epochs	200
callbacks	Tensorboard & ModelCheckpoints

Table 5.3: *Model fit parameters*

5.1 Size Ratio

As mentioned in section 2.3, the half mass radius can be used to quantitatively characterize the size of a galaxy in EAGLE simulations. We define the half mass radius as the radius enclosing half the stellar mass of the galaxy. If the galaxies involved in a merger have sizes inferred by HMR_1 and HMR_2 such that $HMR_1 > HMR_2$, we define the size ratio as HMR_1/HMR_2. The size ratio is a number between [0,1]. In this section, we discuss the training results obtained when the galaxy merger images are trained on their size ratio.

a) z=0, EAGLE package images

There are a total of 936 images in this set of mergers out of which 628 images (66%) are used for training and the rest of the 308 images(33%) are used for validation. This training is carried out for 200 epochs.

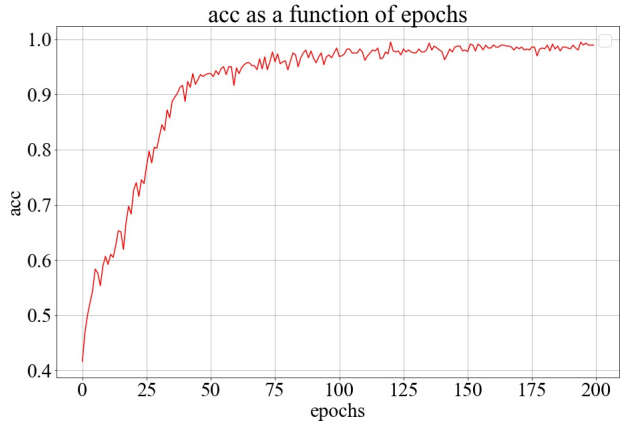


Figure 5.1: *The above figure represents the training accuracy as a function of the number of epochs for images generated via the EAGLE package trained over their size ratio.*

From Fig. 5.1 we see that the training accuracy reaches around 98%. This signifies that the network has almost successfully learned to identify the size ratio from among the training images. The accuracy increases steeply between the 0th and the 50th epoch where it hits a value of 92%, implying that much of the training happens within these 50 epochs. We see that there might be room for more improvement, but keeping the loss in mind, we train upto 200 epochs.

The loss is plotted in Fig 5.2. We see that the loss continuously decreases with the number of epochs. The loss hits almost 0 by the end of 200 epochs. This implies that the network predicts accurately on the training data after training.

Similarly, the validation accuracy is plotted in Fig.5.3. After 50 epochs, the accuracy saturates at 90%. The validation loss is plotted in 5.4. We see that the loss decreases steeply from epochs 0 to 50, and stabilizes till the 75th epoch and

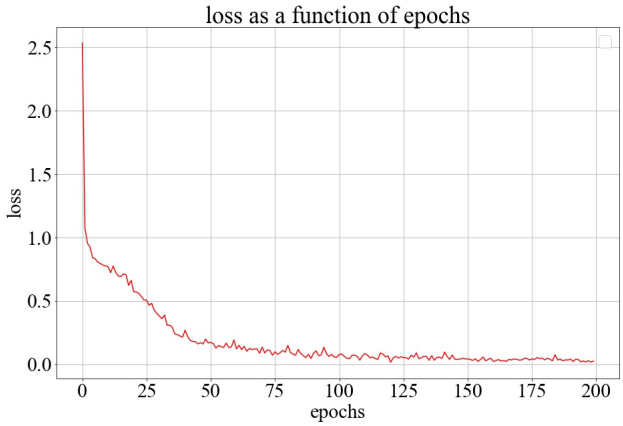


Figure 5.2: *The above figure represents the training loss as a function of the number of epochs for images generated via the EAGLE package trained over their size ratio.*

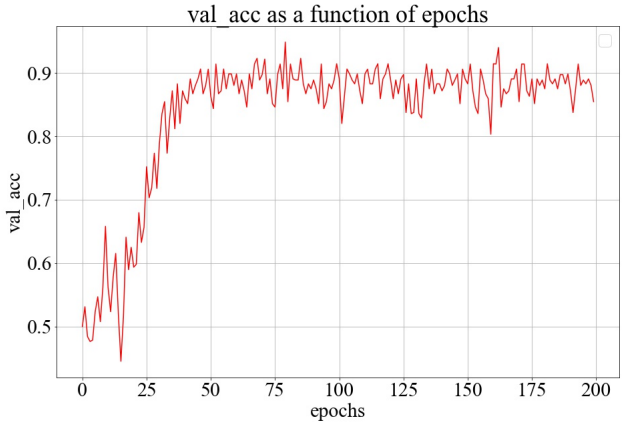


Figure 5.3: *The above figure represents the validation accuracy as a function of the number of epochs for images generated via the EAGLE package trained over their size ratio.*

increases slightly with the increase in the number of epochs until 200. We think that this increasing trend might be due to overfitting.

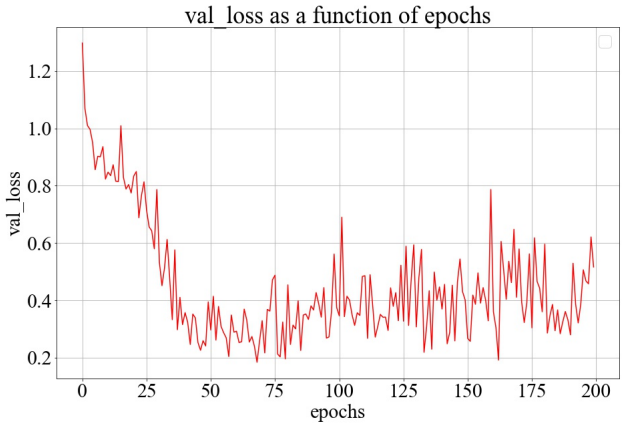


Figure 5.4: *The above figure represents the validation loss as a function of the number of epochs for images generated via the EAGLE package trained over their size ratio.*

b) 20>z>0, zoom images

There are a total of 1000 images in this set of mergers out of which 670 images (66%) are used for training and the rest of the 330 images(33%) are used for validation. This training is carried out for 200 epochs.

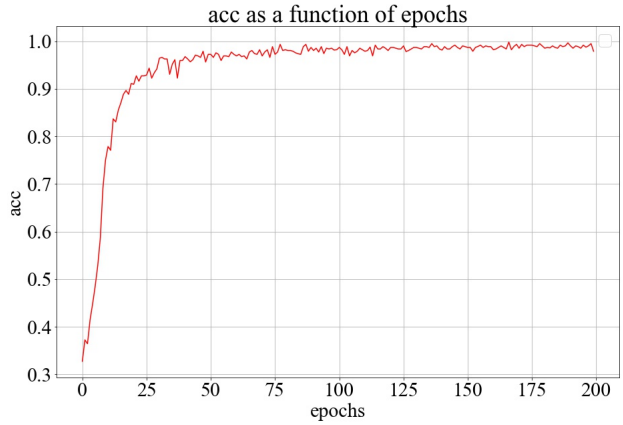


Figure 5.5: *The above figure represents the training accuracy as a function of the number of epochs for images generated via zooming, trained over their size ratio.*

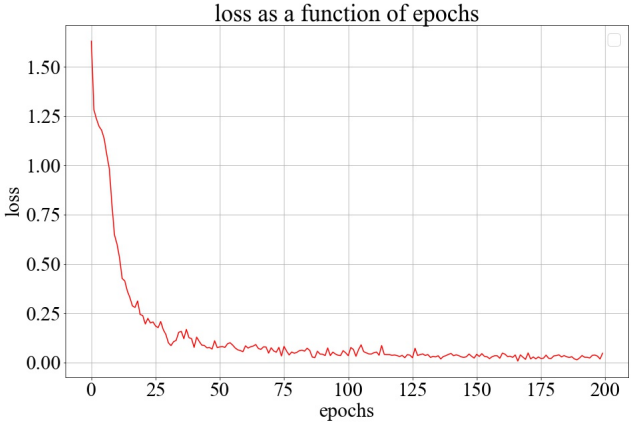


Figure 5.6: *The above figure represents the training loss as a function of the number of epochs for images generated via zooming, trained over their size ratio.*

From Fig. 5.5 we see that the training accuracy reaches around 99%. This signifies that the network has almost successfully learned to identify the size ratio from among the training images. The accuracy increases steeply between the 0th and the 25th epoch where it hits a value of 92%, implying that much of the training happens within these 25 epochs.

The loss is plotted in Fig 5.6. We see that the loss continuously decreases with

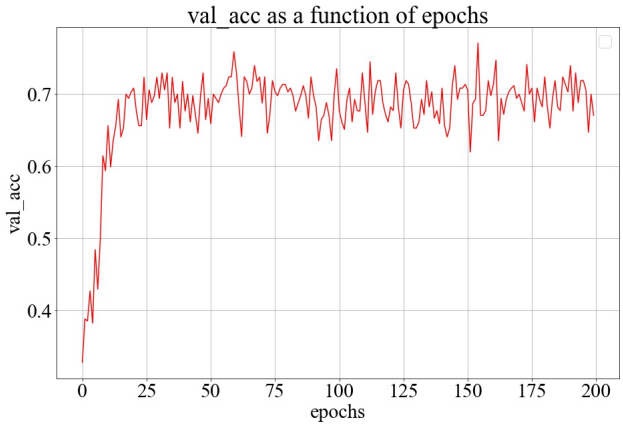


Figure 5.7: *The above figure represents the validation accuracy as a function of the number of epochs for images generated via zooming, trained over their size ratio.*

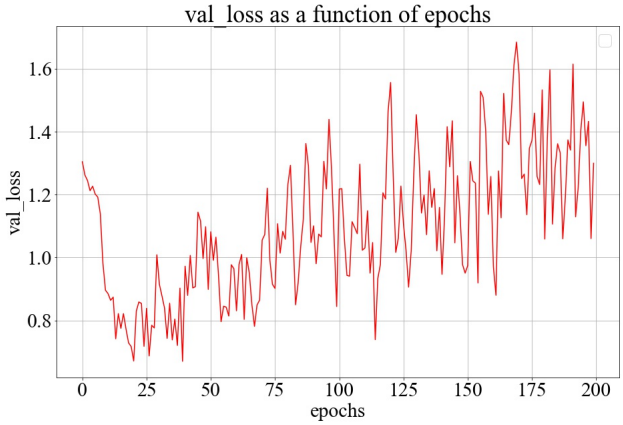


Figure 5.8: *The above figure represents the validation loss as a function of the number of epochs for images generated via zooming, trained over their size ratio.*

the number of epochs. The loss hits almost 0 by the end of 200 epochs. This implies that the network predicts accurately on the training data after training.

Similarly, the validation accuracy is plotted in Fig.5.7. After 25 epochs, the accuracy saturates at 70%. The validation loss is plotted in 5.4. We see that the loss decreases steeply from epochs 0 to 25 and starts increasing linearly until the 200th epoch. We think that this increasing trend might be due to overfitting.

5.2 Mass Ratio

Observationally the estimation of the mass of a galaxy corresponds to its stellar mass at visible wavelengths. This inturn is used to characterize the overall mass of the galaxy. Hence in this thesis, we use the stellar mass to define the mass ratio of mergers. Stellar mass is defined as the mass in the star particles enclosed within a distance of 30kpc from the centre of potential of a galaxy. Therefore mass ratio was calculated by taking the ratios of the stellar masses of the two galaxies involved in a merger. It ranges between [0,1].

a) z=0, EAGLE package images

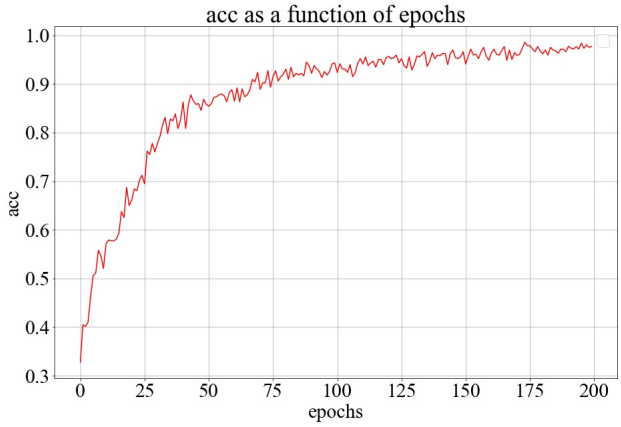


Figure 5.9: *The above figure represents the training accuracy as a function of the number of epochs for images generated via the eagle package trained over their mass ratio.*

From Fig. 5.9 we see that the training accuracy reaches around 97%. This signifies that the network has almost successfully learned to identify the mass ratio from among the training images. The accuracy increases steeply between the 0th and the 75th epoch where it hits a value of 90%, implying that much of the training happens within these 75 epochs.

The loss is plotted in Fig 5.10. We see that the loss continuously decreases with the number of epochs. The loss hits almost 0 by the end of 200 epochs. This implies that the network predicts accurately on the training data after training.

Similarly, the validation accuracy is plotted in Fig.5.11. After 50 epochs, the accuracy saturates at 85%. The validation loss is plotted in 5.12. We see that the loss decreases steeply from epochs 0 to 50 and starts slightly increases until the 200th epoch.

Figure 5.10: *The above figure represents the training loss as a function of the number of epochs for images generated via the eagle package trained over their mass ratio.*

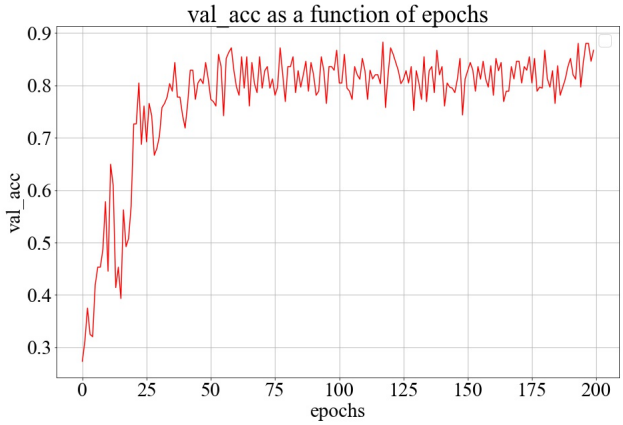


Figure 5.11: *The above figure represents the validation accuracy as a function of the number of epochs for images generated via the eagle package trained over their mass ratio.*

b) 20>z>0, zoom images

There are a total of 2000 images in this set of mergers out of which 1340 images (66%) are used for training and the rest of the 660 images(33%) are used for validation. The batch size used here is 100. This training is carried out for 200 epochs.

From Fig. 5.13 we see that the training accuracy reaches 100%. This signifies that the network has successfully learned to identify the mass ratio from among the training images. The accuracy increases steeply between the 0th and the 25th epoch where it hits a value of 98%, implying that much of the

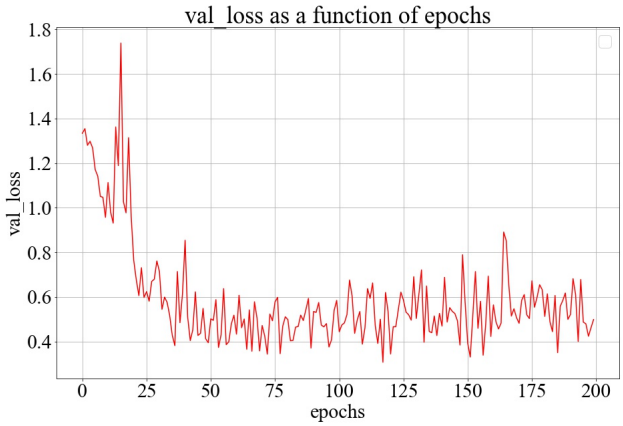


Figure 5.12: *The above figure represents the training validation as a function of the number of epochs for images generated via the eagle package trained over their mass ratio.*

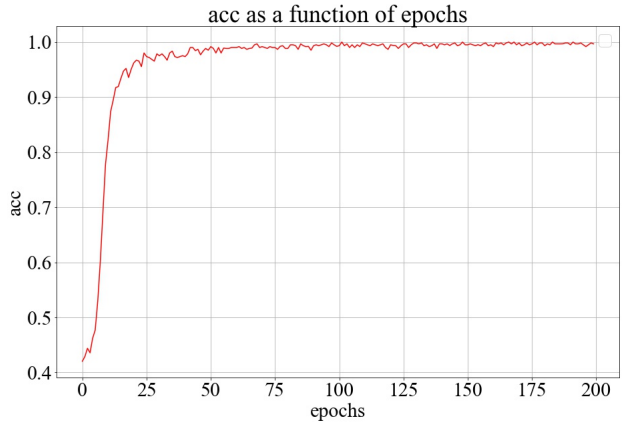


Figure 5.13: *The above figure represents the training accuracy as a function of the number of epochs for images generated via zooming, trained over their mass ratio.*

training happens within these 25 epochs.

The loss is plotted in Fig 5.14. We see that the loss continuously decreases with the number of epochs. The loss hits almost 0 by the end of 200 epochs. This implies that the network predicts accurately on the training data after training.

Similarly, the validation accuracy is plotted in Fig.5.15. After 25 epochs, the accuracy saturates at 80%. The validation loss is plotted in 5.12. We see that the loss decreases steeply from epochs 0 to 25 and starts increasing linearly until the 200th epoch.

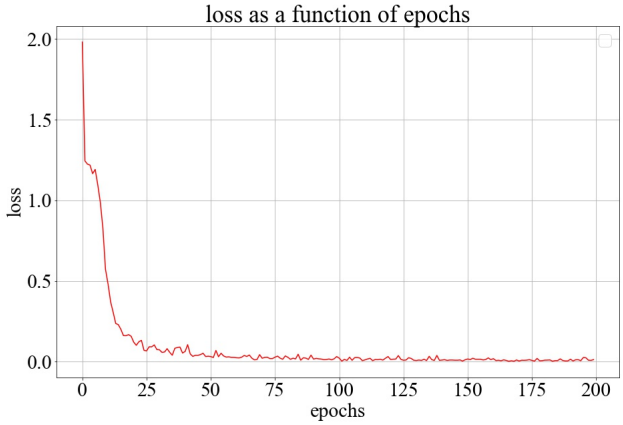


Figure 5.14: *The above figure represents the training loss as a function of the number of epochs for images generated via zooming, trained over their mass ratio.*

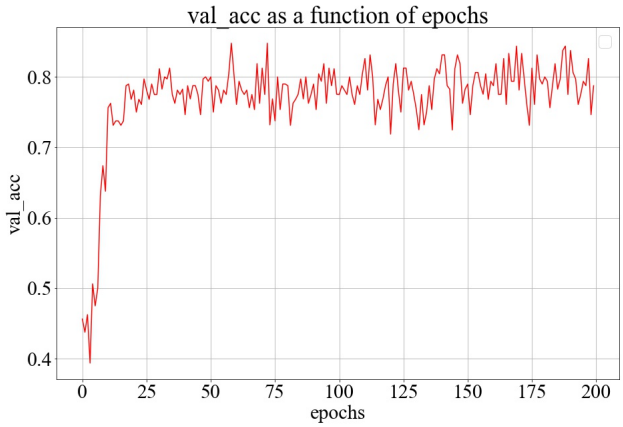


Figure 5.15: *The above figure represents the validation accuracy as a function of the number of epochs for images generated via zooming, trained over their mass ratio.*

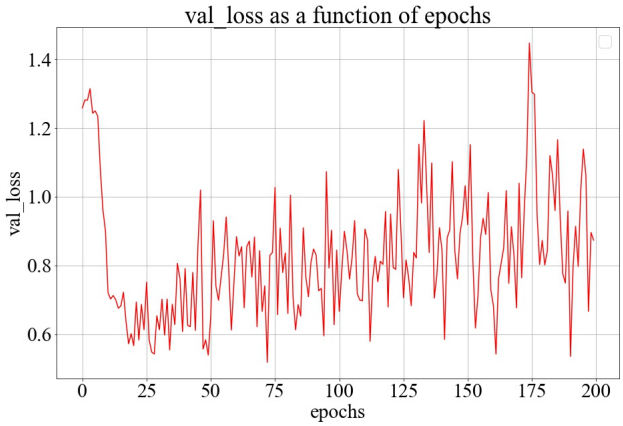


Figure 5.16: *The above figure represents the validation loss as a function of the number of epochs for images generated via zooming, trained over their mass ratio.*

Chapter 6

Discussion

In this chapter we discuss the results obtained as plotted in the previous chapter.

We see that there is a gap between the training and validation accuracies in each case (10-20%). We think that this is because of lack of variety of data. The network seems to learn all that it has to from the training set of images, but doesn't generalize well on the new set of validation data, hence causing a slight drop in validation accuracy. If trained further, this can cause the network to memorize the training images consequently performing worse on the new validation dataset. This is called overfitting. We believe that with an increase in the number of images, the validation accuracy can be improved.

We also notice that despite the differences in the visualization techniques for generating galaxy merger images, the accuracies donot differ much. This implies that the network is robust and independent of the visualization.

In Fig. 5.4, 5.8, 5.12 and 5.15 plotted above, we see constant fluctuations in the value of the loss beyond the point of most learning. We think this might be due to the optimizer used. Stochastic Gradient Descent updates the model parameters every iteration due to which, the parameter updates have a high variance which causes fluctuations in the loss. To tackle this we use a high momentum of 0.9. We also notice in Fig. 5.8 and 5.16, where the validation losses for zoomed images is plotted, there is an increasing trend in the loss. We think that this might be due to overfitting.

Chapter 7

Conclusion

Galaxy mergers are important to understand galaxy evolution as they contribute to the mass growth and change in morphology of galaxies. It is hard to determine galaxy merger properties from observations. Since we know everything about our simulation, we aim to infer observed galaxy merger properties with simulation data. We have chosen EAGLE simulations to generate galaxy merger images. We explore the properties that can be extracted from the images and choose visually apparent properties like mass and size ratio. We train a Deep Neural Network on the generated images of galaxy mergers against these properties. The network is adopted from the keras example model that was initially successfully trained on the MNIST dataset with an accuracy of 99.25% test accuracy. The results show that, the mass ratio at z=0 and $20>z>0$ achieves a validation accuracy of 85% and 80% each and size ratio achieves 90% and 70% respectively. The high accuracies achieved indicates that Deep Neural Network can be used to quantitatively deduce properties from galaxy merger images. With further improvement in the accuracy, the model can be used to predict on observational images.

Chapter 8

Future work

Here we mention a few ways in which we can further explore the project.

- Implementing the improvements mentioned in section 2.2 and 2.4 and create a more realistic rendering of galaxy mergers.
- Using more augmentation techniques like scaling, cropping, translation, blurring etc to expand the sample space.
- Considering the initial parameters from simulations and upsampling the mergers to achieve images of a higher resolution.
- Tweaking the hyper-parameters to achieve a higher accuracy.
- Deducing properties like relative velocities and collision angles of merging galaxies.
- Using Transfer Learning (see Appendix A.3)
- On achieving higher accuracies from training a neural network on simulations, we can deduce properties of galaxy mergers obtained observationally and understand them quantitatively.

Appendix A

Appendix

A.1 Assumptions

Distance Criteria - The proximity threshold between two galaxies to be regarded as neighbours.

Major Mergers - Galaxy pairs that lie within a distance of 30kpc from each others' centre of potential and have a mass ratio ($\mu = M_2/M_1$ where $M_1 > M_2$) greater than $1/6$.

Mass Criteria - The ratio of the masses of the two galaxies involved in a merger to be regarded as a major merger.

Stellar Mass - The total mass of all the stars in a galaxy lying within a 30kpc radius from its centre of potential in the EAGLE simulations.

Half Mass Radius- The radius enclosing half the stellar mass of the galaxy from its Centre of Potential.

A.2 Redshifts in EAGLE simulations

EAGLE simulation provides insight into the evolution of gas, star, dark matter and Black Hole particles within a simulation box of 100Mcpc (Ref-L1001504), as they evolve through time from redshift 20 to redshift 0 captured at 29 instants of time called snapshots. We quantitatively understand the simulation in entirety in these snapshots. The snapshots and their corresponding redshifts, lookback time and expansion factors are shown in the table below.

SnapNum	Redshift	Lookback time	Expansion factor
28	0.00	0.00	1.000
27	0.10	1.34	0.909
26	0.18	2.29	0.846
25	0.27	3.23	0.787
24	0.37	4.16	0.732
23	0.50	5.19	0.665
22	0.62	6.01	0.619
21	0.74	6.71	0.576
20	0.87	7.37	0.536
19	1.00	7.93	0.499
18	1.26	8.86	0.443
17	1.49	9.49	0.402
16	1.74	10.05	0.365
15	2.01	10.53	0.332
14	2.24	10.86	0.309
13	2.48	11.16	0.287
12	3.02	11.66	0.249
11	3.53	12.01	0.221
10	3.98	12.25	0.201
9	4.49	12.46	0.182
8	5.04	12.63	0.166
7	5.49	12.75	0.154
6	5.97	12.86	0.143
5	7.05	13.04	0.124
4	8.07	13.16	0.110
3	8.99	13.25	0.100
2	9.99	13.32	0.091
1	15.13	13.53	0.062
0	20.00	13.62	0.047

Figure A.1: *List of snapshot numbers and their corresponding redshifts in EAGLE simulations.*

Version of June 2019– Created June 30, 2019 - 16:21

A.3 Transfer learning

Training a Convolutional Neural Network is essentially tuning the free parameters/weights of the network based on an optimization algorithm. As the network becomes deeper and hence more complex, the training set must also expand. In case of a limited training set, although the network learns with high accuracy to identify the properties of the set, it doesn't perform well on new data. This makes it less accommodating. This problem is called overfitting. To tackle overfitting and hence generalize the network, various regularization techniques can be used. One of the regularization techniques is to freeze the pre-trained weights of the CNN layers and train the network over only the last few dense layers to achieve a higher accuracy than training the network from scratch. This technique is called transfer learning (Ackermann et al. 2018). In this paper, the detection of galaxy mergers (from the Darg et al. 2010) is carried out by using the pre-trained weights of the CNN trained on millions of images of everyday things from the IMAGENET dataset (Deng et al. 2009) as a starting point. It is seen that this improves the performance of the network significantly.

The sample space in Ackermann et al. 2018 constitutes of 3003 merger galaxies (Darg et al. 2010) and 10000 non-interacting galaxies(Sloan Digital Sky Survey data release 7) labeled from the Galaxy Zoo project. This represents a small fraction of galaxy mergers from among millions of galaxies within the redshift range $0.005 < z < 0.1$. Galaxy mergers thus, are a rare occurrence and aren't labeled in observations. As neural networks get more complex, in order to understand galaxy mergers beyond their morphology and aim at detecting more subtle properties like mass ratio, size ratio, relative velocities and inclination, regularization techniques are necessary to overcome the problem of overfitting. Thus Transfer Learning is a useful tool to tackle it.

Version of June 2019– Created June 30, 2019 - 16:21

Acknowledgements

I would like to thank my supervisor Dr. Maxwell Cai for the opportunity to work on this project and his constant support. I would also like to thank Dr. Camila Correa, Nastasha Wijers and Dr James Trayford, for their valuable help regarding EAGLE simulations. I would like to thank the second reader for taking the time to read my thesis and giving valuable comments on it. Lastly, I would like to express my gratitude to my parents, boyfriend, my ex, friends and classmates for their support through this journey.

Bibliography

[1] Accuracy
2016. Accuracy. = https://github.com/keras-team/keras/blob/c2e36f369b411ad1d0a40ac096fe35f73b9dffd3/keras/metrics.py.

[2] Ackermann, S., K. Schawinski, C. Zhang, A. K. Weigel, and M. D. Turp
2018. Using transfer learning to detect galaxy mergers. *Monthly Notices of the Royal Astronomical Society*, 479(1):415–425.

[3] Adrian, R.
2018. Fit generator. = https://www.pyimagesearch.com/2018/12/24/how-to-use-keras-fit-and-fit_generator-a-hands-on-tutorial/.

[4] Alves, B. A. F. S.
2018. Galaxy mergers. `https://github.com/b-fontana/GalaxyMergers`.

[5] Clauwens, B., J. Schaye, M. Franx, and R. G. Bower
2018. The three phases of galaxy formation. *Monthly Notices of the Royal Astronomical Society*, 478(3):3994–4009.

[6] Conselice, C. J.
2003. The Relationship between Stellar Light Distributions of Galaxies and Their Formation Histories. , 147:1–28.

[7] Crain, R. A., J. Schaye, R. G. Bower, M. Furlong, M. Schaller, T. Theuns, C. Dalla Vecchia, C. S. Frenk, I. G. McCarthy, J. C. Helly, et al.
2015. The eagle simulations of galaxy formation: calibration of subgrid physics and model variations. *Monthly Notices of the Royal Astronomical Society*, 450(2):1937–1961.

[8] Darg, D. W., S. Kaviraj, C. J. Lintott, K. Schawinski, M. Sarzi, S. Bamford, J. Silk, R. Proctor, D. Andreescu, P. Murray, R. C. Nichol, M. J. Raddick, A. Slosar, A. S. Szalay, D. Thomas, and J. Vandenberg
2010. Galaxy Zoo: the fraction of merging galaxies in the SDSS and their morphologies. *Monthly Notices of the Royal Astronomical Society*, 401(2):1043–1056.

[9] Davis, M., G. Efstathiou, C. S. Frenk, and S. D. White
1985. The evolution of large-scale structure in a universe dominated by cold dark matter. *The Astrophysical Journal*, 292:371–394.

[10] Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei
2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, Pp. 248–255. Ieee.

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

[12] Dolag, K., S. Borgani, G. Murante, and V. Springel
2009. Substructures in hydrodynamical cluster simulations. *Monthly Notices of the Royal Astronomical Society*, 399(2):497–514.

[13] Duffy, A. R., J. Schaye, S. T. Kay, and C. Dalla Vecchia
2008. Dark matter halo concentrations in the wilkinson microwave anisotropy probe year 5 cosmology. *Monthly Notices of the Royal Astronomical Society: Letters*, 390(1):L64–L68.

[14] EAGLE
2017a. Eagle database. = http://galaxy-catalogue.dur.ac.uk:8080/Eagle/.

[15] EAGLE
2017b. Eagle particle database. = http://data.cosma.dur.ac.uk:8080/eagle-snapshots/.

[16] Fukushima, K., S. Miyake, and T. Ito
1983. Neocognitron: A neural network model for a mechanism of visual pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):826–834.

[17] GalaxyZoo
. Galaxyzoo. = https:https://data.galaxyzoo.org/galaxy- zoo-mergers/targets/index.html.

[18] Hamed, M.
2015. Tfrecords stack overflow. = https:https:https://stackoverflow.com/questions/33849617/how-do-i-convert-a-directory-of-jpeg-images-to-tfrecords-file-in-tensorflow.

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

[20] Helly, J.
2017. Read eagle package. = https://gitlab.cosma.dur.ac.uk/jch/Read_Eagle.

[21] Hopkins, P. F., L. Hernquist, P. Martini, T. J. Cox, B. Robertson, T. Di Matteo, and V. Springel
2005. A physical model for the origin of quasar lifetimes. *The Astrophysical Journal Letters*, 625(2):L71.

[22] Hubel, D. H. and T. N. Wiesel
1962. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154.

[23] Kannan, R., A. V. Macció, F. Fontanot, B. P. Moster, W. Karman, and R. S. Somerville
2015. From discs to bulges: effect of mergers on the morphology of galaxies. *Monthly Notices of the Royal Astronomical Society*, 452(4):4347–4360.

[24] keras team
2018. keras github. = https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py.

[25] Lintott, C. J., K. Schawinski, A. Slosar, K. Land, S. Bamford, D. Thomas, M. J. Raddick, R. C. Nichol, A. Szalay, D. Andreescu, P. Murray, and J. Vandenberg
2008. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. , 389:1179–1189.

[26] Loss
. Loss. = http://deeplearning.net/software/theano/library/tensor/nnet/nnet.html#theano.tensor.nnet.nnet.categorical_crossentropy.

[27] Lotz, J. M., J. Primack, and P. Madau
2004. A New Nonparametric Approach to Galaxy Morphological Classification. , 128:163–182.

[28] Madau, P. and M. Dickinson
2014. Cosmic star-formation history. *Annual Review of Astronomy and Astrophysics*, 52(1):415–486.

[29] McAlpine, S., J. C. Helly, M. Schaller, J. W. Trayford, Y. Qu, M. Furlong, R. G. Bower, R. A. Crain, J. Schaye, T. Theuns, et al.
2016. The eagle simulations of galaxy formation: public release of halo and galaxy catalogues. *Astronomy and Computing*, 15:72–89.

[30] Optimizer
. Optimizer. = https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f.

[31] Prasad, P.
. Tfrecords advantages. = https:https://medium.com/ymedialabs-innovation/how-to-use-tfrecord-with-datasets-and-iterators-in-tensorflow-with-code-samples-ffee57d298af.

[32] Qu, Y., J. C. Helly, R. G. Bower, T. Theuns, R. A. Crain, C. S. Frenk, M. Furlong, S. McAlpine, M. Schaller, J. Schaye, and S. D. M. White
2017. A chronicle of galaxy mass assembly in the eagle simulation. *Monthly Notices of the Royal Astronomical Society*, 464(2):1659–1675.

[33] Rodriguez-Gomez, V., L. V. Sales, S. Genel, A. Pillepich, J. Zjupa, D. Nelson, B. Griffen, P. Torrey, G. F. Snyder, M. Vogelsberger, V. Springel, C.-P. Ma,

and L. Hernquist
2017. The role of mergers and halo spin in shaping galaxy morphology. *Monthly Notices of the Royal Astronomical Society*, 467(3):3083–3098.

[34] Scannapieco, C. e. a., M. Wadepuhl, O. Parry, J. Navarro, A. Jenkins, V. Springel, R. Teyssier, E. Carlson, H. Couchman, R. Crain, et al.
2012. The aquila comparison project: the effects of feedback and numerical methods on simulations of galaxy formation. *Monthly Notices of the Royal Astronomical Society*, 423(2):1726–1749.

[35] Schaye, J., R. A. Crain, R. G. Bower, M. Furlong, M. Schaller, T. Theuns, C. Dalla Vecchia, C. S. Frenk, I. G. McCarthy, J. C. Helly, A. Jenkins, Y. M. Rosas-Guevara, S. D. M. White, M. Baes, C. M. Booth, P. Camps, J. F. Navarro, Y. Qu, A. Rahmati, T. Sawala, P. A. Thomas, and J. Trayford
2015. The EAGLE project: simulating the evolution and assembly of galaxies and their environments. , 446:521–554.

[36] Springel, V.
2005. The cosmological simulation code gadget-2. *Monthly notices of the royal astronomical society*, 364(4):1105–1134.

[37] Springel, V., J. Wang, M. Vogelsberger, A. Ludlow, A. Jenkins, A. Helmi, J. F. Navarro, C. S. Frenk, and S. D. White
2008. The aquarius project: the subhaloes of galactic haloes. *Monthly Notices of the Royal Astronomical Society*, 391(4):1685–1711.

[38] Springel, V., S. D. White, G. Tormen, and G. Kauffmann
2001. Populating a cluster of galaxies–i. results at z= 0. *Monthly Notices of the Royal Astronomical Society*, 328(3):726–750.

[39] Tensorflow
.a. Keras. = https:https:https://www.tensorflow.org/guide/keras.

[40] Tensorflow
.b. Tensorboard. = https://www.tensorflow.org/guide/summaries_and_tensorboard.

[41] Tensorflow
. Tfrecords. = https://www.tensorflow.org/tutorials/load_data/tf_records.

[42] Tensorflow
. Tfrecords conversion. = https://medium.com/mostly-ai/tensorflow-records-what-they-are-and-how-to-use-them-c46bc4bbb564.

[43] Tensorflow
2017. Tfrecords github. = https://github.com/tensorflow/models/blob/f87a58cd96d45de73c9a8330a06b2ab56749a7fa/research/inception/inception/data/build_image_data.py.

[44] The EAGLE team
2017. The EAGLE simulations of galaxy formation: Public release of particle data. *ArXiv e-prints*.

[45] Ventou, E., T. Contini, N. Bouché, B. Epinat, J. Brinchmann, R. Bacon, H. Inami, D. Lam, A. Drake, T. Garel, et al.
2017. The muse hubble ultra deep field survey-ix. evolution of galaxy merger fraction since zâ 6. *Astronomy & Astrophysics*, 608:A9.

[46] Ventou, E., T. Contini, N. Bouché, B. Epinat, J. Brinchmann, R. Bacon, H. Inami, D. Lam, A. Drake, T. Garel, L. Michel-Dansac, R. Pello, M. Steinmetz, P. M. Weilbacher, L. Wisotzki, and M. Carollo
2017. The MUSE Hubble Ultra Deep Field Survey. IX. Evolution of galaxy merger fraction since z 6. , 608:A9.

[47] Wiersma, R. P., J. Schaye, and B. D. Smith
2009a. The effect of photoionization on the cooling rates of enriched, astrophysical plasmas. *Monthly Notices of the Royal Astronomical Society*, 393(1):99–107.

[48] Wiersma, R. P., J. Schaye, T. Theuns, C. Dalla Vecchia, and L. Tornatore
2009b. Chemical enrichment in cosmological, smoothed particle hydrodynamics simulations. *Monthly Notices of the Royal Astronomical Society*, 399(2):574–600.

[49] Wikipedia contributors
2019a. Application programming interface — Wikipedia, the free encyclopedia. [Online; accessed 18-June-2019].

[50] Wikipedia contributors
2019b. Deep learning — Wikipedia, the free encyclopedia. [Online; accessed 20-June-2019].

[51] Wikipedia contributors
2019c. Mnist database — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=MNIST_database&oldid=899811825`. [Online; accessed 20-June-2019].

[52] Wikipedia contributors
2019d. Tensorflow — Wikipedia, the free encyclopedia. [Online; accessed 18-June-2019].

[53] Woods, D. F. and M. J. Geller
2007. Minor galaxy interactions: Star formation rates and galaxy properties. *The Astronomical Journal*, 134(2):527–540.