UNIVERSITY OF CALIFORNIA,
IRVINE

Deep Mining and its Astronomical Applications

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Physics

by

Milad Pourrahmani

Dissertation Committee:
Professor Asantha Cooray, Chair
Professor Manoj Kaplinghat
Professor Michael Cooper

2019

DEDICATION

To all those who left their home in pursuit of science
&
to their mothers, fathers, friends, and families.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to thank my advisor Dr. Asantha Cooray and my thesis committee, Dr. Micheal Cooper and Dr. Manoj Kaplinghat, for their help and support throughout my graduate studies, especially in my last year. Throughout my time in Cooray Group, Asantha supplied me with the necessary resources to carry out my research. I am especially thankful that he let me work on one of the most exciting topics in science, enabling me to progress in a promising direction in my career which I will be presenting in this thesis.

With the intellectual freedoms that I was granted in Asantha's group, throughout the years, I have obtained the necessary skills to develop optimized CPU/GPU accelarated software packages for scientific purposes, to engage in collaborative code sharing, to work with most advanced ML libraries, and to become proficient with data science and machine learning techniques, all necessary tools needed for modern science which, unfortunately, I feel many of my peers did not have the privilege to indulge. For this, I thank my advisor, Asantha, who I am glad he has noticed and encouraged my interest in programming and recently, has given me the opportunity to work with the SPHEREx collaboration. In my opinion, this is the most exciting ongoing cosmology collaboration that will constrain inflationary models by conducting an all-sky spectroscopic survey and I am honored to take a small part in it.

Additionally, being part of Asantha's group has allowed me to grow my passion for physics alongside my passion for programming. Though I will not discuss this in this thesis, to look at the Universe from a computational perspective as opposed to the traditional analytical point of view was the most valuable thing I have learned in graduate school. This would not have been possible without the help of my dear friend Sunny Yu who introduced me to the world of quantum field theory, with patience, clarity, and immense enthusiasm.

Special thanks to Hooshang Nayyeri for his help throughout the years as a co-author and as a friend. I appreciate the Astrophysical Journal and the coauthors Hooshang Nayyeri and Asantha Cooray for granting me permission to publish our previous works in this thesis. I would like to thank Dr. Emre Neftci, Dr. Charless Fowlkes, Dr. Pierre Baldi for their valuable feedbacks and Noah Ghotbi, Aroosa Ansari, Kevin Payumo, Bencheng Li for their assistance in my research during their undergraduate years.

As an immigrant during my college and as a citizen now, I feel blessed and I am thankful for all those citizens who made it possible for me to pursue a college degree and to attend graduate school. I must give special thanks for the generous financial support by NSF grant AST-1213319 and GAANN P200A150121 without which my work would not have been possible. I am also thankful for the donated GPU by NVIDIA Grant Program, accelerating my research.

I can only imagine thehardship that my family, especially my mother, father, and brother had to go through when I left Iran at age 17 in pursuit of physics. I cannot thank them enough for supporting my decisions and giving me the courage to move forward. I also must thank my mentors, Mohammad Nasiri and Horia Petrache, for planting the seeds of curiosity

and teaching me how to think scientifically.

CURRICULUM VITAE

Milad Pourrahmani

EDUCATION

Doctor of Philosophy in Physics **2019**
University of California, Irvine *Irvine, CA*

Bachelor of Science in Physics **2012**
Purdue University *Indianapolis, IN*

RESEARCH EXPERIENCE

Graduate Research Assistant (Astrophysics) **2007–2012**
University of California, Irvine *Irvine, California*

Research Assistant (Biophysics) **2012–2014**
Purdue University *Indianapolis, IN*

TEACHING EXPERIENCE

Teaching Assistant **2014–2019**
University of California, Irvine *Irvine, California*

Laboratory Instructor **2011–2014**
Purdue University *Indianapolis, IN*

REFEREED JOURNAL PUBLICATIONS

LensFlow: A Convolutional Neural Network in Search of Strong Gravitational Lenses **2018**
Astrophysical Journal

Deep Mining Strong Gravitational Lenses in Astronomical Surveys **2020**
In Preparation

SOFTWARE

All of the developed libraries for this project are publicly available on GitHub.

EZnet `GitHub.com/Miladiouss/EZnet`
Pytorch algorithm for mining rare objects with limited training examples.

LensFlow `GitHub.com/Miladiouss/LensFlow`
Mathematica algorithm for identifying strong gravitational lenses.

LensCrafter `GitHub.com/Miladiouss/LensCrafter`
Python algorithm for fast simulation of strong gravitational lenses.

LensGenerator `GitHub.com/keviinn/GANS`
Pytorch algorithm for generating gravitational lenses using GANs.

ABSTRACT OF THE DISSERTATION

Deep Mining and its Astronomical Applications

By

Milad Pourrahmani

Doctor of Philosophy in Physics

University of California, Irvine, 2019

Professor Asantha Cooray, Chair

Buried among millions of galaxies, large sky surveys embed strong gravitational lenses essential to cosmological studies. For instance, individual lenses have been used to study the properties of dark matter and early-type galaxies. Lensed supernovae studies have resulted in model-independent constraints on the Hubble constant and the spatial curvature. If a statistically significant number of lenses in sky surveys are discovered and studied, some of the persisting mysteries of cosmology such as the Hubble tension and the cosmic curvature may be resolved. Toward this end, in the first part of this thesis, I will demonstrate deep learning can recover lenses identified by, or missed by, other methods. This proof of concept was performed with a deep neural net with only three convolutional blocks and was applied to the HST/ACS i-band observations of the COSMOS field, a well-studied region of the sky. In the second half, I present a set of fully developed computational techniques as a ready-to-use software package for lens mining, or for mining other rare events in large sky surveys using more sophisticated deep learning methods. Inspired by DenseNet, I have designed a simple and easily modifiable neural net, with more than 50 convolution blocks, which with the help of a Generative Adversarial Neural Network (GAN) augmentation can learn from a mere 50 examples. After training on 50 known lenses, I was able to identify 42 lens candidates in the Subaru/HSC-PDR1. For the first time, I provide the selection function for a lens mining algorithm using a fast and realistic lens synthesizer. This is a crucial

step in gravitational lens studies allowing us to provide a useful link between observational catalogs and theoretical cosmological predictions via counterbalancing the inevitable biases of the selection process.

Chapter 1

Introduction

1.1 Strong Gravitational Lenses and Their Significance to Cosmology

Gravitational lensing, a prediction of Einstein's general theory of relativity, is a very powerful tool in cosmological studies. It has been used extensively to understand various aspects of galaxy formation and evolution (e.g. Refsdal & Bondi (1964); Blandford & Narayan (1992); Nayyeri et al. (2016); Postman et al. (2012); Atek et al. (2015)). This involves accurate cosmological parameter estimation (Treu, 2010), studies of dark matter distribution from weak gravitational lensing events (Kaiser & Squires, 1993; Velander et al., 2014), black-hole physics (Peng et al., 2006) and searches for the most distant galaxies (Coe et al., 2012; Oesch et al., 2015), among others.

One of the main goals of observational cosmology is to constrain the main cosmological parameters that dictate the evolution of the Universe (Tegmark et al., 2004; Komatsu et al., 2009; Weinberg et al., 2013). Strong gravitational lensing has been utilized over the past few

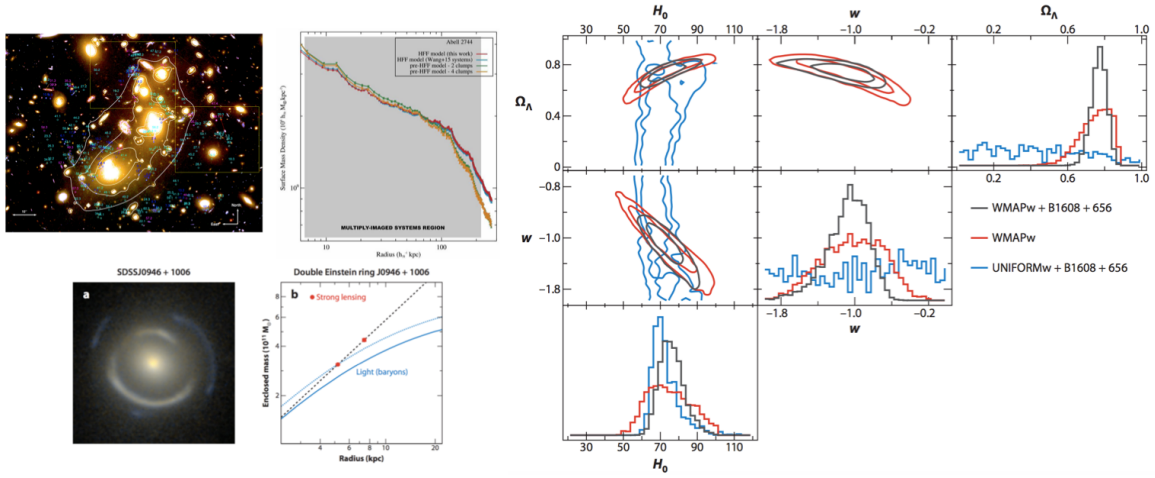

Figure 1.1: The importance of strong gravitational lensing incosmological studies. The multiple lensed images produced by lensing could be utilized to estimate the mass distributions in clusters (Jauzac et al. 2015; Top Left) and/or in galaxy-galaxy lensing (Auger et al. 2009; Bottom Left). Detailed gravitational lens modeling combined with estimates of lensing induced time delays (time-delay cosmography) could measure and put constraints on the main cosmological parameters (Treu et al. 2010, Suyu et al. 2014; Right). Here the blue line shows constraints from a strong gravitationally lensed system, which combined with limits from CMB observations (in red) could accurately measure the cosmological parameters (Treu et al. 2010, Suyu et al. 2014).

years to estimate and constrain these cosmological parameters (Broadhurst et al., 2005; Suyu et al., 2013, 2014; Goobar et al., 2017; Agnello et al., 2017; More et al., 2017). This is achieved through accurate lens modeling of such events and comparing the model predictions with observations (such as with observations of lensing induced time delays (Eigenbrod et al., 2005; Treu, 2010; Suyu et al., 2014; Rodney et al., 2016; Treu & Marshall, 2016). In a recent study, for example, Suyu et al. (2013) used combined WMAP, Keck and *HST* data on gravitational time delays in two lensed sources to constrain the Hubble constant within 4% in a ΛCDM cosmological framework. See Figure 1.1 for an example of such analysis.

One of the key aspects of gravitational lensing is its use as natural telescopes through boosting the observed signal and increasing the spatial resolution (Treu, 2010). This is quite advantageous in searches for distant and/or faint objects at moderate observing costs and has been utilized extensively in various surveys in searches for such objects, the identification

of which would not have been possible without it (Bolton et al., 2006; Heymans et al., 2012). Given that the number of identified lenses for different classes of galaxies rises sufficiently due to better lens finding algorithms, one could study the intrinsic properties of distant galaxies from such searches to understand the physics of star-formation and mass assembly (Wilson et al., 2017; Timmons et al., 2016; Nayyeri et al., 2017; Fu et al., 2012). In the past few years, deep diffraction limited observations have also taken advantage of gravitational lensing to extend the faint end of the luminosity function of galaxies by a few orders of magnitude (Atek et al., 2015) to produce the deepest images of the sky ever taken across multiple bands. Strong gravitational lensing events have been observed extensively in such surveys as galaxy-galaxy lensing in field surveys such as the Cosmic Assembly Near-infrared Deep Extragalactic Legacy Survey (CANDELS) (Grogin et al., 2011; Koekemoer et al., 2011) and the Cosmological Evolution Survey (COSMOS) (Scoville et al., 2007; Capak et al., 2007) or as cluster lensing from observations of nearby massive clusters (Postman et al., 2012; Treu et al., 2015; Lotz et al., 2017) with *Hubble* Space Telescope leading to the identification of the first generations of galaxies (out to $z \sim 11$; Oesch et al., 2015) and studies of galaxy formation and evolution at the epoch of re-ionization. This was, in fact, one of the main motivations behind *Hubble* cluster lensing studies such as Cluster Lensing and Supernova Survey with Hubble (CLASH; Postman et al., 2012) and the *Hubble* Frontier Fields (Lotz et al., 2017). The power of gravitational lensing could also be used in the detection of low surface brightness emission from extended objects such as millimeter and radio emissions from dust and molecular gas at $z \sim 2-3$ as observed with ALMA used to study the physics of the cold ISM (Spilker et al., 2016).

Strongly lensed galaxies are normally targeted and identified from dedicated surveys (Bolton et al., 2006). Traditionally these lens identifications are either catalog-based, in which lensing events are identified by looking for objects in a lensing configuration, or pixel based, with the search starting from a set of pixels. These lensing searches are normally computationally challenging in that individual pixels are constantly compared with adjacent ones and they

	Conventional Lens Search Algorithms	Machine Learning Neural Networks
Pros	Ability to pick rare lensing configurations Easier translation across surveys/instruments	Characterization of selection functions for cosmology Robust lens identification for wide-areas
Cons	Inefficient performance on wide-area surveys Strong function of the selection cuts Non straightforward selection function	Strong function of the training data and could miss non-standard lenses

Table 1.1: Comparison of the existing lens identification algorithms to that of machine learning searches.

could be biased towards a given population and/or brighter objects. Recent far-infrared wide area observations (such as with *Herschel*) advanced searches for lensed galaxies by adopting a simple efficient selection technique of lensed candidates through observations of excessive flux in the far infrared (as an indication of strong lensing events supported by number count distributions; Nayyeri et al., 2016; Wardlow et al., 2012). However such surveys are also biased towards populations of red dusty star-forming galaxies (missing any blue lenses) and are not always available across the full sky (the *Herschel* surveys that were targeted had $\sim 0.2 - 0.4\,\mathrm{deg}^{-2}$ lensing events, much lower than expected from optical surveys). Given that tests of cosmological models require simple unbiased selection functions, it is important to have a complete unbiased catalog of lensing events.

We have entered the era of big data astronomy. Sky surveys such as the LSST, Euclid, and WFIRST will produce more imaging data than humans can ever analyze by eye. The challenges of designing such surveys are no longer merely instrumentational, but they also demand powerful data analysis and classification tools that can identify astronomical objects autonomously. Fortunately, computer vision has drastically improved in the last decade making autonomous astronomy possible. The past couple of years has been the most exciting era in the field of machine learning (ML). Researchers from both the public and the private sectors have achieved landmarks in developing image recognition/classification techniques. One of the most exciting recent events in the ML community was the release of TensorFlow by Google, a parallel processing platform designed for development of fast deep learning

algorithms (Abadi et al., 2016). Packages and softwares such as MATHEMATICA, TENSORFLOW, CAFFE, and others alongside with cheaper and more powerful Graphics Processing Units (GPUs) have enabled researchers to develop very complex and fast classification algorithms. Among these deep learning programs, ConvNets have deservingly received a lot of attention in many fields of science and industry in the past few years (Krizhevsky et al., 2012). Complex ConvNets such as GOOGLENET and ALEXNET, which are publicly available, have achieved superhuman performance on the task of image classification. Google's TENSORFLOW has made it possible to easily develop parallelized deep learning algorithms which if integrated with Google's Tensor Processing Units (TPUs), could address the data mining challenges in the field of astronomy. In this work, we introduce and image classification algorithm, LENSFLOW, which is a ConvNet that can be used to search for strong gravitational lenses with the final version of the code publicly available on Github.

This thesis, in accordance to our first paper (Pourrahmani et al., 2018) is organized as follow. In Section 1.2, we will explain the principal concepts underlying neural networks, supervised learning, and ConvNets. Before feeding the images to a ConvNet, they must be normalized and should be enhanced. The details data extraction and normalization are discussed in Section 2.1. As discussed in Sections 2.3, we explain the architecture of LENSFLOW and its pre-training process on CIFAR data (Krizhevsky & Hinton, 2009). In Section 2.4 and 2.5, we discuss training LENSFLOW on COSMOS data and show its performance on recovering known tracer lenses. In Section 2.6, we share a set of new lenses found by LENSFLOW and we conclude our results in Section 2.7. Throughout this paper, we assume a standard cosmology with $H_0 = 70\,\mathrm{kms}^{-1}\mathrm{Mpc}^{-1}$, $\Omega_m = 0.3$ and $\Omega_\Lambda = 0.7$. Magnitudes are in the AB system where $\mathrm{m_{AB}} = 23.9 - 2.5 \times \log(f_\nu/1\mu\mathrm{Jy})$ (Oke & Gunn, 1983).

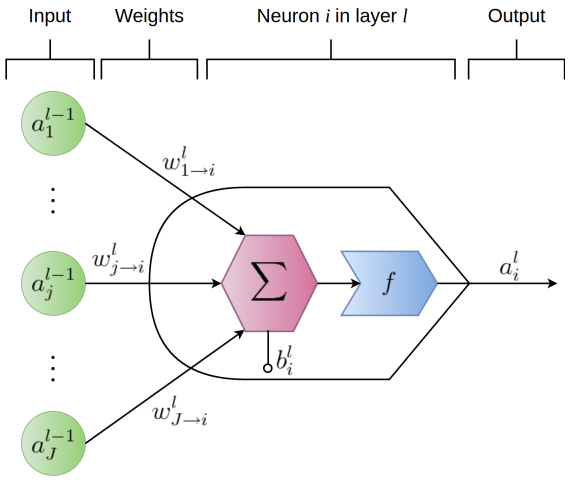

Figure 1.2: Schematic representation of an artificial neuron. The weighted sum of the neurons in the previous layer (green circles), plus the internal bias of the neuron, are mapped as the output of the neuron by an activation function. This model is captured by Equation 1.2. During the learning process, weights and biases of the neurons will be adjusted to achieve the desired network output.

1.2 Deep Learning Algorithms

Artificial neural networks are inspired by biological neurons. Just like biological neurons, artificial neurons receive input signals and send out an output signal to other neurons (see Figure 1.2). The synaptic connections between neurons are known as weights and the output of a neuron is know as its activation. To reduce the computational time and simplify neural network models, neurons are placed in consecutive layers rather than having a connection with every other neuron. This neural network setup is know as the Multi-Layer Perception where neurons from one layer cannot talk to each other or to the neurons in arbitrary layers; they may only send their signal to the neurons in the succeeding layer. A neuron receives the weighted sum of the activation of all the neurons in the previous layer, adds an internal parameter known as the bias and maps this sum to a value computed by an activation function (e.g. sigmoid, hyperbolic tangent, rectilinear, softmax). This model can be stated mathematically by the following equation:

$$a_i^l = f(\sum_j a_j^{l-1} w_{j\to i}^l + b_i^l). \tag{1.1}$$

Here, a_i^l is the activation of the neuron in hand (i.e. the i'th neuron in the l'th layer), f is the activation function of this neuron, a_j^{l-1} is the activation of the neuron j in layer $l-1$ (the previous layer), $w_{j\to i}^l$ is the synaptic weight connecting i'th neuron in layer l to the j'th neuron in layer $l-1$, and b_i^l is the bias of the neuron to adjust its activation sensitivity. The first layer, i.e. the input layer, in a deep learning neural net acts as a sensory layer, analogous to the retina. As it gets analyzed, the information from the input layer travels through multiple layers until it reaches the final layer called the classification layer. Each class of images corresponds to a classifying neuron. In our case, we have a neuron corresponding to non-lensed and another to lensed images. The neuron with the highest output determines which class an input image is placed in.

A neural net learns how to classify images by adjusting the weights between its neurons and the biases within them, having one goal in mind: minimizing the loss function $C(\boldsymbol{x}, \boldsymbol{y})$. The loss function, sometimes called the cost function, can take many forms but it has to capture the misfiring of the classification neurons, i.e. the deviation between the target class versus the predicted class. This is why such algorithms are known as supervised learning algorithms, in contrast to unsupervised techniques. A common choice for the loss function is the cross-entropy loss function with the following form (Nielsen, 2016):

$$C(\boldsymbol{x}, \boldsymbol{y}) = \sum_{\substack{j=\text{non-lens,}\\ \text{lens}}} y_j \ln a_j^L + (1-y_j)\ln(1-a_j^L). \tag{1.2}$$

a_j^L is the activation of neurons in the final (classifying) layer. $\boldsymbol{x}$ is the input data in the vector form and $\boldsymbol{y}$ represents the desired activations of the two classifying neurons. Of course, this function depends on the architecture of the neural net, weights, and biases, but they have

not been expressed explicitly. As an example, if an image is a lens, its target output has to be $(0.0, 1.0)$, meaning the activation of the non-lensed neuron should be zero and the activation of the lensed neuron should be unity. During the training process of a neural net, images from a training dataset are presented to the network and the weights and the biases are adjusted to minimize the loss function for those images. The parameter space is massive and a change in one of the parameters of a neuron will affect the activation of a series of neurons in other layers. The first challenge is solved by minimizing algorithms such as the stochastic gradient descent (SGD) and the second one is solved via back-propagation. Since optimizing over the whole training set at once is not possible, because the training data would not fit in memory, stochastic optimization algorithms provide guaranteed convergence even if the gradients are evaluated on a randomly (stochastically) selected subset (batch) of the training dataset. They provide a practical way to optimize a model over extremely large datasets. Batches yield noisy approximations to the true gradient, and larger batches can better approximate this quantity while if the batch size is too small, that approximation would be too poor and the algorithm my never converge in practice.

ConvNets are a class of neural networks with multiple convolutional layers. A convolutional layer consists of a set of convolving neurons (on the order of 10 neurons) which can be connected to a small rectangular region of an image. The set of weights of a convolving neuron is known as a filter and are subject to change as the ConvNet learns. A filter scans an entire image by striding (convolving with specified steps) over the image and assembling its output into an image known as a feature map. Feature maps contain information such as texture and edges. See Figure 1.3 as an example of a set of filters in a LensFlow convolutional layer. A few examples of feature maps has been shown in Figure 1.3. These feature maps are bundled together as an image with the same number of channels as the number of feature maps. In this image, we have selected three feature maps and represent them with different color to display what the neural network sees as the image passes through the layers.

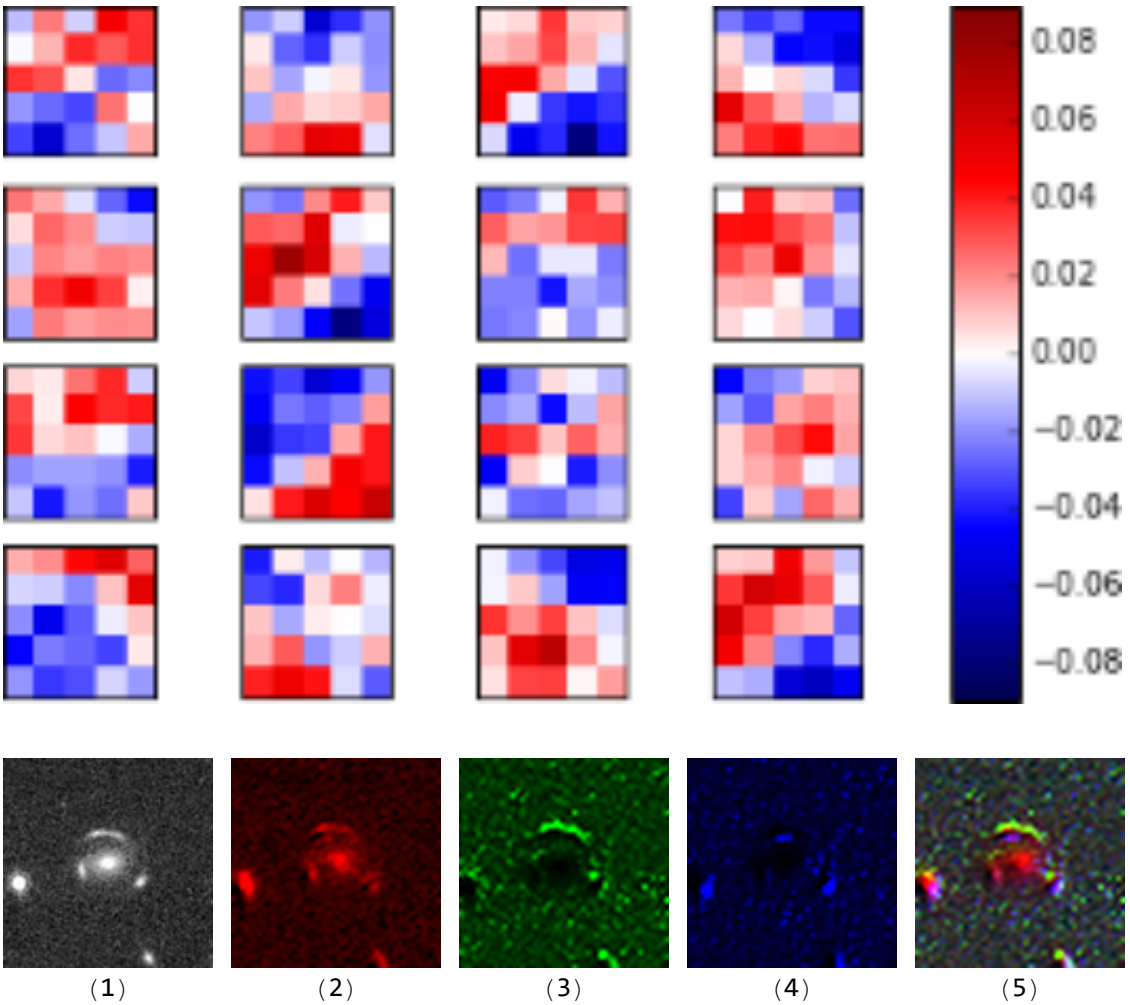

Figure 1.3: *Top:* Examples of filters used in a convolutional layer. The pixels in each box represent the weights of a convolving neuron which are connected to a 5×5 region input image. As these filters convolve over the entire input image, they generate feature maps. Red pixels have a positive contribution and blue pixels have a negative contribution toward the activation of the convolving neuron. These filters are helpful for edge and texture recognition. *Bottom:* An examples of convolutional layer feature maps. Image of a normalized physical lens has been shown in (1). (2-4) show three outputs of the second convolution layer of LensFlow. (5) shows the superposition of these three maps. As it can be seen in (5), these feature maps create a contrast between the upper arc and the foreground source, making it possible for the fully connected layers to decide whether an imaged is lensed or not.

Chapter 2

Lens Mining with Preliminary ConvNets as a Proof of Concept

This chapter/section covers the image normalization process, simulation of gravitational lenses and training and testing dataset creation, architecture of LensFlow, its pre-training on CIFAR dataset, and its training on COSMOS data in two sequential steps: course and fine classification phases.

2.1 Data Extraction and Normalization

We used *HST*/ACS i-band observations in the full COSMOS field to search for candidate gravitationally lensed sources. In order to prepare the survey data for the neural network, we created 200×200-pixel cutouts around sources identified by SExtractor which corresponds to roughly 3×3 square arcseconds. We ignored sources which extended less than 200 pixels total (not to be confused with our cutout size) and were not 1.5σ brighter than the background, totaling to 236 thousand images. These images were then down-sampled to

100×100 pixels to speed up the training and scanning process.

Before inputting the images to LensFlow, we normalized and enhanced them, which is a necessary step to ensure a stable training and prevents activation saturation issues. For deep learning purposes, there are different methods of image normalization to choose from. This is due to the fact that raw image pixels come in a wide range of values . As we will discuss in the next section, when lenses are produced by superposing simulated arcs on top of actual sources, it is crucially important to ensure superposed images are renormalized after the superposition process. If lensed images are not renormalized, the net will become sensitive to the total sum of the pixels and achieve a meaningless perfect classification on the training and test datasets with no application for searching for real lensed images.

There are different methods of image normalization used which often involve shifting the mean, normalizing the standard deviation, and bounding the pixels between two fixed values. Though sufficient for classification of daily object photos, we did not find these methods helpful to our algorithm since astronomical images require gamma correction to adjust the image contrast. Gamma correction introduces a non-linearity according to the following equation:

$$p_{out} = A p_{in}^{\gamma}, \tag{2.1}$$

where A and γ are constants and p_{in} and p_{out} are the initial and corrected pixel values. However, applying the same gamma function to different sources is not practical. For instance, the arcs in some lensed images might get enhanced while they are obscured by the foreground in other images. Similar problems happen when cutting off bright and dim pixels. To overcome these issue, we have selected one dim lensed source and have adjusted its brightness, its contrast, and have performed gamma correction so the foreground source and the arcs are clearly separated and visible while keeping all pixel values between 0 and 1. The

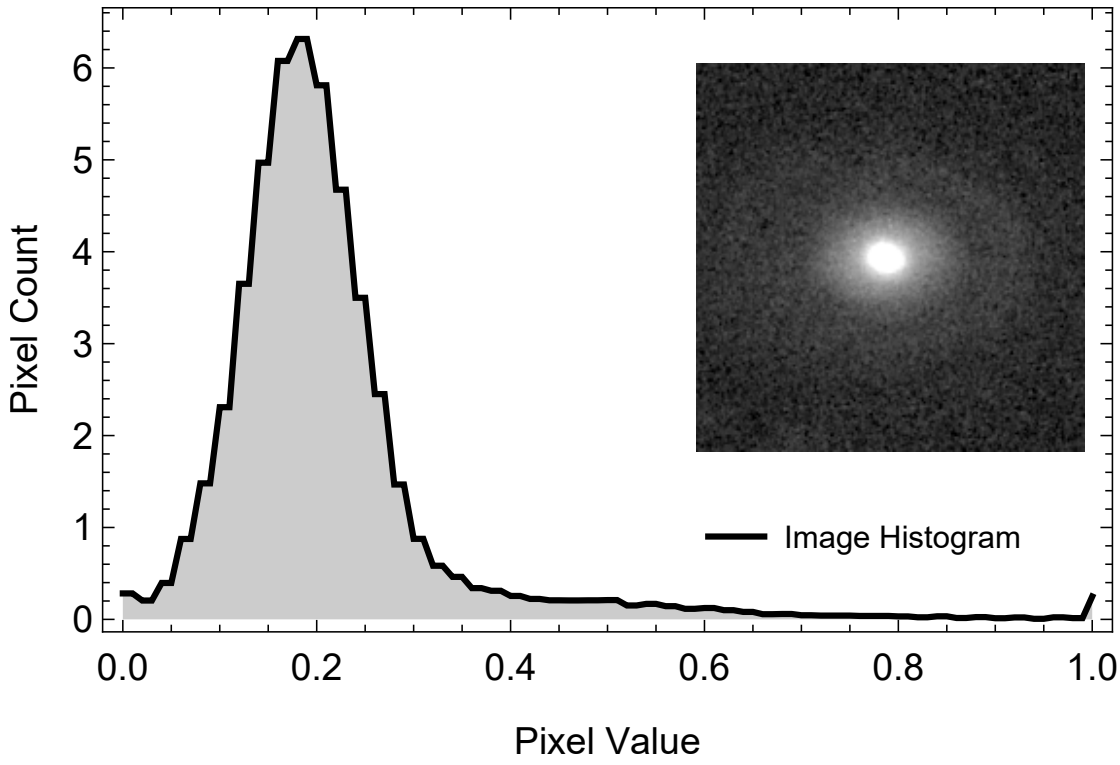

Figure 2.1: Template image histogram for image normalization. The histogram of all sources were transformed to match this template histogram which was obtained by adjusting the brightness, the contrast, and by performing gamma correction for a known dim lensed image displayed above. Not only this transformation normalizes images, but it will also enhance the contrast between the arcs and the foreground source.

image histogram (the histogram of pixel values) of this modified lensed image was extracted (See Figure 2.1) and was used as template histogram to transform the image histogram of all the extracted sources. Not only this method enhanced the arcs for all the previously known lenses from COSMOS, but it also automates the process of gamma correction and normalization since all pixel values fall between 0 and 1 and their mean and their standard deviation is roughly the same.

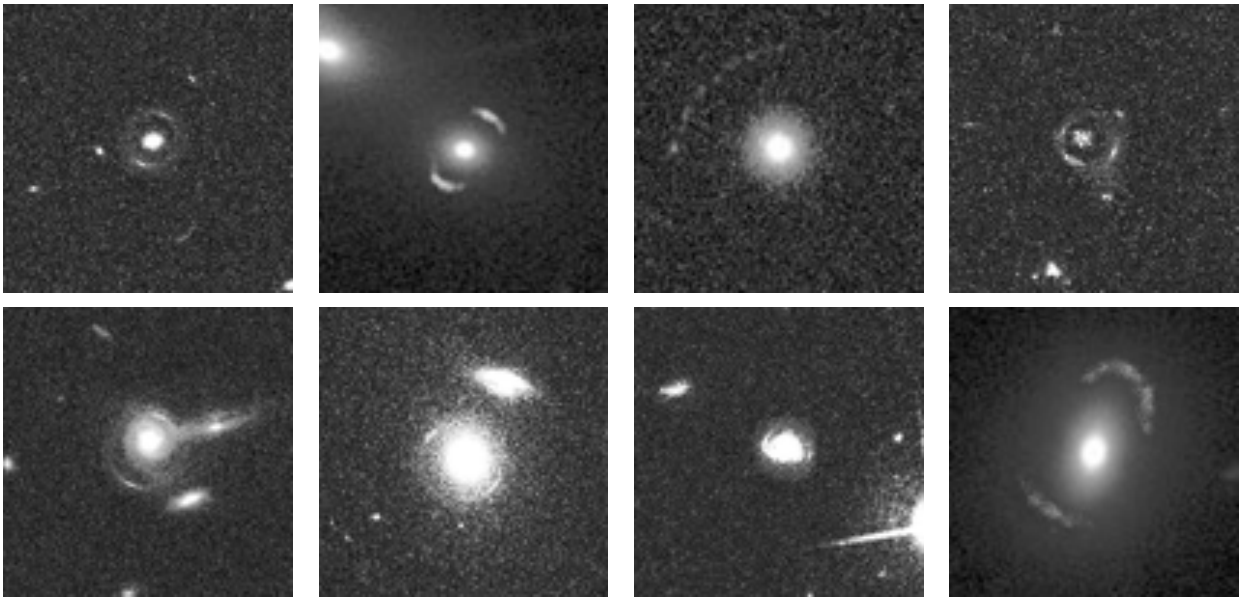

Figure 2.2: Examples of simulated lenses.

2.2 Lens Simulation

In order to train a neural network, typically a few thousand examples are needed per class. Since the number of known lenses are far more limited than the required number, these lenses have to be simulated. For these simulations we used LENSTOOL (Jullo et al., 2007) to generate image plane models of lensing systems using realistic models of randomly selected elliptical galaxies within the COSMOS field as deflectors and co-added these to the selected elliptical galaxies to generate the training set. Here, we focus on elliptical galaxies as foreground deflectors. Although known examples of spiral galaxy lensing exist (Calanog et al., 2014), most galaxy-galaxy lensing events occur around massive elliptical galaxies as foreground deflectors. We generated a training set of 200 galaxies using this method. Figure 2.2 shows several examples from the generated training set used by LensFlow.

As discussed in the previous section, simulated lenses must be renormalized to prevent the net from classifying lenses based on the total sum of the pixels since without normalization, total pixel sum of lensed images will be always higher than non-lens images.

The number of generated lenses is still very low for training purposes. To overcome this, we can use image augmentation to artificially boost up the number of training examples. We do this by rotating and reflecting images. In more details, we use 8 transformations that come from 8 elements of the symmetry group of the square, namely: $0\,^{\circ}$, 90°, 180° and 270° rotations, and horizontal, vertical, diagonal and anti-diagonal reflections. In addition to rotation and to reflection, we take eight 90×90-pixel cutouts at random positions and rescale them to 100×100 pixels. We will refer to these two processes as image augmentation.

2.3 Architecture of LensFlow ConvNet and Pre-training (Phase 0)

The architecture of the data determines the dimensionality of the ConvNet layers. We use $1 \times 100 \times 100$ images where 1 indicate the number of color channels [1]. Classifying lenses with multiple color bands will be easier and more accurate since foreground and background sources have a color contrast. However, we have chosen to use one color channel so our algorithm can be sensitive to geometry rather than color contrast in order to expand its applicability to a wider range of bands as well as eliminating its need for multi-band images when unavailable. The results of galaxy lens identification from wide-area surveys with color information will appear in a future work (Pourrahmani et al., forthcoming in 2020). As we see in Figure 2.3 and Table 2.1, after normalizing these single-channeled images, LensFlow applies an average-pooling of a kernel of size 5×5 and a stride of 1 without padding. The hyperbolic tangent function introduces non-linearity to the convolution layer. The common choice is often a rectified linear unit (ReLU) which sets pixels smaller than a self-learned threshold to zero. ReLU is ideal for edge detection but since astronomical images do not

[1] In this paper and in our code, we have adapted the $N \times C \times H \times W$ where N, C, H, and W stand for number of input images in a batch, number of color or feature channels, height, and width respectively.

have hard edges, an smoother function like hyperbolic tangent is suited. The output of this layer is fed to a pooling layer with a kernel of size 2×2 and a stride of 1, outputting the largest pixel value as it convolves its input for all channels. The result of this layer is a set of 30 downsampled (48×48) feature maps. The next two convolution layers are identical to the first set described above except that the second convolution layer has 60 and the last convolution layer has 90 filters. The output of the last convolution layer is a set of $90 \times 9 \times 9$ feature maps which are flattened from a tensor to a 1-D vector with 7290 rows which is fed to the fully connected layers. The first fully connected layer has 1000 linear neurons (identity function as f in Equation 1.2). This layer is followed by a dropout layer where the output of 50% of the neurons is set to zero. Dropout layers prevent over-fitting in early stages of training. Two more linear layers of size 800 and 600 follow this layer. Finally, all inputs are fed to a linear layer of size two with a softmax nonlinearity. These two layers act as a classifying layer where the softmax layer converts the output of the linear layer to probabilities:

$$\sigma_c(\mathbf{Z}) = \frac{e^{Z_c}}{e^{Z_{\text{non-lensed}}} + e^{Z_{\text{lensed}}}} \,. \tag{2.2}$$

Here, each component of $\mathbf{Z}$ is the output of the last linear layer (i.e. output of Layer 20) with two components. c specifies whether we are talking about the neuron corresponding to lensed or non-lensed images. The softmax function ensure the output sums to one and when used with the cross-entropy loss function, these outputs are interpreted as class probabilities. We use these probabilities to rank images from most probable lens candidates to least probable.

To optimize our ConvNet, we have chosen a cross-entropy function as our loss function which we minimize using Adam Optimizer. This adaptive optimizer algorithm computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients for the loss function (Kingma & Ba, 2014). During the training phase, 64 non-lensed and 64 lensed images were placed in a batch of 128 images. This combination

technique will prevent under- or over-representation of classes even if the training size for different classes contain a different number of examples.

Often for smaller datasets, nets are pre-trained on a different but larger dataset which trains the net how to identify features. We have selected two classes from the CIFAR dataset, a famous dataset used for testing computer vision algorithms. After reducing the images to gray-scale and changing their size to 100×100 pixels, we applied the image normalization explained above. The network was trained for a total of 8 rounds. The number of iteration rounds is determined by deviation of loss function for training and test datasets. It is an indication of over-fitting if the average loss function for the training dataset drops while it remains the same or increases for the test dataset. In simpler terms, over-fitting means the net is memorizing the training dataset rather than learning generalizable feature extraction and classification. Similarly, we determine the number of iteration rounds for other training phases discussed in the following two subsections. Pre-training is crucial for our dataset without which no learning occurs. We suspect that having soft edges as well as a central dominant object are the main causes of the trouble here, preventing the network from learning edge detection and picking up on arc-like features. Techniques such as reducing the brightness of central pixels were tested which triggered the learning process even though with a poor performance. However, by using a pre-trained net, the need for masking the central bright pixels was eliminated and a much better performance was achieved.

2.4 Course Classification Phase (Phase 1)

The training dataset for this phase consists of 3200 lenses created by augmenting the 200 simulated lenses and randomly selecting 3200 images from COSMOS. It is possible that these randomly selected images contain actual lenses, but a few misclassified examples will not affect the training process in a noticeable way. To generate a validation and a testing

Table 2.1: Tabulated Architecture of the LensFlow ConvNet.

Layer	Type	Data Dimensionality
	input	$1 \times 100 \times 100$
1	convolution	$30 \times 96 \times 96$
2	tanh	$30 \times 96 \times 96$
3	pooling	$30 \times 48 \times 48$
4	convolution	$60 \times 44 \times 44$
5	tanh	$60 \times 44 \times 44$
6	pooling	$60 \times 22 \times 22$
7	convolution	$90 \times 18 \times 18$
8	tanh	$90 \times 18 \times 18$
9	pooling	$90 \times 9 \times 9$
10	flatten	7290
11	linear	1000
12	ReLU	1000
13	dropout	1000
14	linear	800
15	ReLU	800
16	dropout	800
17	linear	600
18	ReLU	600
19	dropout	600
20	linear	2
21	softmax	2
	output	2

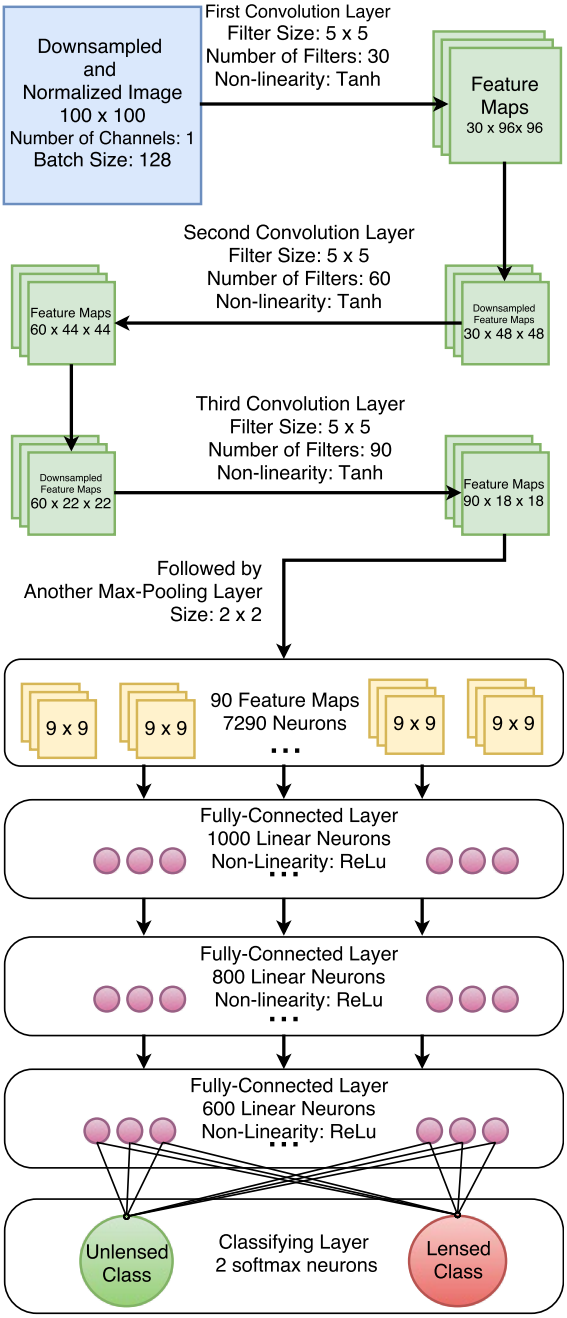

Figure 2.3: Representation of data flow through the ConvNet layers. The data is downsampled and fed to three convolutional-max-pooling layers. The data is then flattened into an array and is fed to 3 fully connected linear layers which are then connected to the classifying layers consisting of a linear layer with two neurons followed by a sofmax layer. The convolution layers are responsible for feature extraction and the fully connected layers learn the difference between lensed and non-lensed images.

dataset, we have selected 52 out of 67 discovered lens candidates by Faure et al., 2008 and have applied image augmentation (rotations and reflections only) to increase their number to 464 which were accompanied by 464 randomly selected images from COSMOS labeled as non-lensed images. Among the lens candidates that were not selected, three were larger than 3 arcseconds in diameter, one did not have an i-band image, and the rest did not have any arc features in the i-band and were classified as lenses by Faure et al., 2008 by mainly relying on other bands.

Since the convolution layers of the pre-trained net (layers 1 to 10 in Table 2.1) are used for feature extraction and are transferable from one dataset to another, during the training process on COSMOS data, the learning rate of those layers were reduced to 10%. On the other hand, the main purpose of the fully connected layers are to classify based on the extracted features and are not transferable. Hence, their learning rate was unaltered during the training process.

After 20 training epochs, we conducted two main performance tests. The first test is obtaining the receiver operating characteristic curve, i.e. plotting the ROC curve, which is a standard measure of the performance of a classifier. As plotted in Figure 2.4, the horizontal and the vertical axes indicate the false positive rate (FPR) and the true positive rate (TPR) respectively. The ROC curve is obtained by evaluating FPR and TPR for different classification thresholds (i.e. the minimum lens probability for an image to be categorized as lensed). Even though ROC curves are very useful for most classifiers, we suggest a different performance measure which is more appropriate for the field of astronomy where thousands or millions of images have to be scanned to identify desired sources. In our case, after training the net, we have placed the 52 selected lenses as tracers among the entire 236 thousand source images extracted from COSMOS. The assigned lens probability by this net has been used to rank the images from the most likely lens candidates to the least likely. The number of recovered tracer lenses as a function of relative ranks (i.e. rank of an image divided by the

total number of images) are plotted in Figure 2.5. We will refer to this curve as the tracer rank curve (TRC). We see that 100% of tracer lenses fall in the top 6% of the sorted images. A TRC can be quantified by one number which we refer to as the ranking performance. We define the ranking performance as the area between a TRC (the solid black line in Figure 2.5) and the line of no discrimination (the dashed red line in the same figure) divided by the TRC for a perfect classifier (i.e. approximately 1/2 when the number of scanned images is much larger than the number of tracer images). Therefore, a ranking performance of 1 corresponds to placing all tracer lenses in the highest ranks while a ranking performance of 0 corresponds to dispersing tracer lenses among all images which is a sign of no learning. A negative ranking performance, on the other hand, means the classifier is systematically misclassifying images. The ranking performance of our ConvNet during this phase is 0.97 which is quite good for such a simple ConvNet applied on a dataset with one color channel. However, placing all the tracer lenses in the top 6% means lenses have to be recovered among $14,000$ images. Even though this is a massive reduction from 236 thousand, examining 14 thousand images by eye is not very practical and would be impossible for larger surveys. For this reason, we have introduced another phase that specializes on finding lenses among the remaining 14 thousand images by retrain our net on the top 6% images, as discussed in the following subsection.

2.5 Fine Classification Phase (Phase 2)

In order to further reduce the number of images that have to be examined by eye, we have constructed a dataset by randomly selecting $3,200$ images from the remaining COSMOS images from Phase 1. The same $3,200$ augmented simulated lensed images were added to complete the dataset. The ConvNet was trained over 25 iteration and the remaining images from Phase 1 were scanned and ranked using this net. The first 300 images were examined

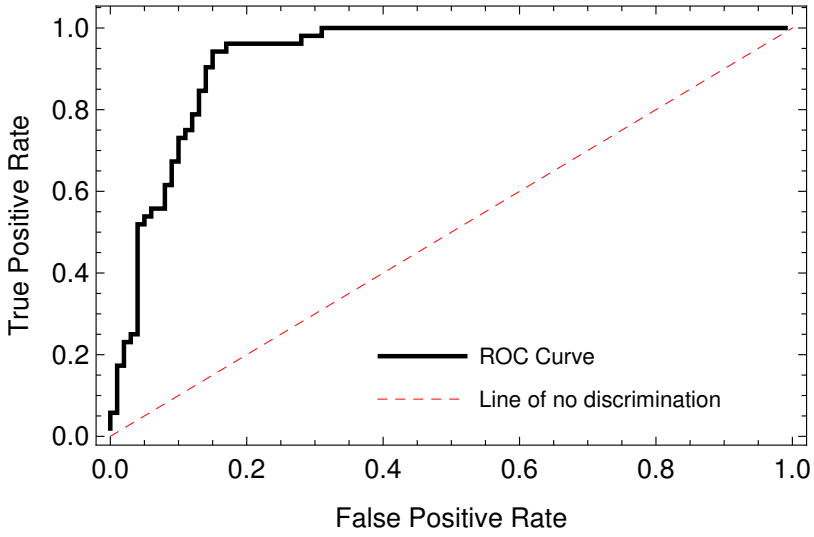

Figure 2.4: Receiver operating characteristic (ROC) diagram. The black curve shows the trend of the true positive rate verses the false positive rate of LensFlow while the red dashed curve shows an untrained classifier.

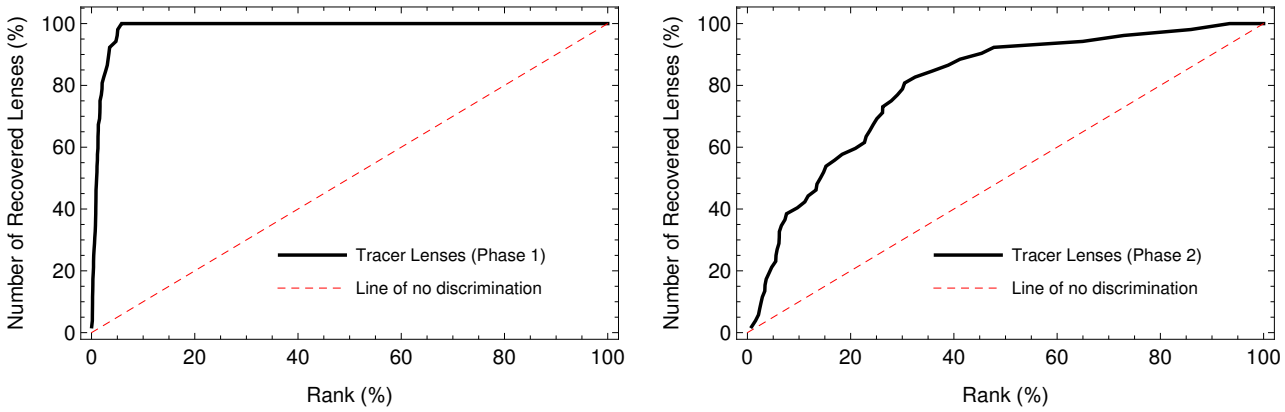

Figure 2.5: Normalized ranking of tracer lenses by LensFlow. Ranking performance is defined as the area between the black tracer rank curve (TRC) and the dashed red line of no discrimination divided by the area between a perfect TRC and the line of no discrimination (approximately 1/2 for large datasets). Left: 100% of the tracer lenses are in the top 6%. The ranking performance of Phase 1 is 0.97. Right: During Phase 2, the ConvNet was trained on the top 6% images from Phase 1. The ranking of the tracer lenses has been show. 80% of the tracer lenses are in the top 30%. The ranking performance of Phase 2 is 0.60.

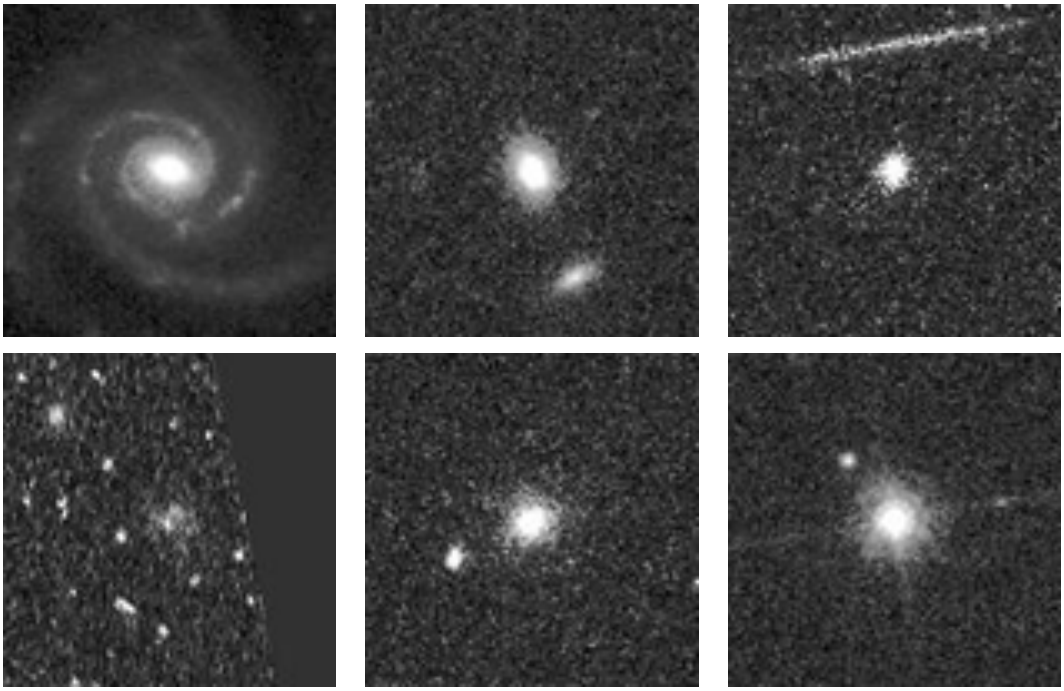

Figure 2.6: Examples of common misclassified images. These images were used to re-train LensFlow to improve its ranking performance.

which included some lenses but mostly artifacts, spiral galaxies, and satellite galaxies (see Figure 2.6). Lensed images were removed and the remaining were augmented and added to the training dataset for re-training to eliminate most probable false classifications. The TRC for this net is plotted in the right panel of Figure 2.5 with a ranking performance of 0.60. The same net with the same methodology could do extremely better if images had color information which is not present in our data. Other ways to improve the results is to create a more diverse and larger dataset which is very time consuming. Using more complex nets such as GoogLeNet with perception models might improve the results which we will investigate in a future study. The results of Phase 1 show that simple and fast deep learning algorithms such as ours is sufficient enough to reduce the data by a factor of 17 since both training and scanning are more time consuming with complex nets such as GoogLeNet.

2.6 Search Phase (Phase 3)

This phase is identical to the previous phase with the exception of including augmented tracer lenses in the training dataset in order to increase the dataset for improving the chances of finding new lens candidates. The reason the previous phase was necessary is to obtain the maximum number of training epochs and to test the performance of LensFlow. After training this ConvNet, we examined $2,000$ images and identified 46 new lens candidates that were not mentioned in Faure et al. (2008) (the examination process roughly took 20 minutes). These lens candidates are shown in Figure 2.7 and their coordinates are listed in Table 2.2 (See Appendix 2.9). Classification algorithms like ours can benefit from Citizen Science projects such as the SPACE WARPS project(More et al., 2016; Marshall et al., 2015) where volunteers are presented with real and simulated gravitational lenses in order to obtain a measure of their classification performance while identifying new lenses. Citizen Science projects such as SPACE WARPS can help with classification of images with high lensing probabilities assigned by automated classifiers.

2.7 Discussion

Non-machine learning computer algorithms have been previously used for finding gravitationally lensed arcs (Alard, 2006; Lenzen et al., 2004; More et al., 2012) and rings (Gavazzi et al., 2014). As discussed in More et al. (2012) and Gavazzi et al. (2014) is an algorithm that uses color information and ArcFinder detects arc-like pixels. In more details, ArcFinder starts by polishing the images by convolving a smoothing kernel. For each pixel, an estimator of elongation is calculated by taking the ratio between the sum of the flux of a few pixels along the horizontal line and the maximum value of a few nearby pixels along the vertical line which pass through the pixel in hand. This process is repeated for all pix-

els and those with smaller than an specified elongation threshold are set to zero to create a sharp arc map. An arc map that satisfies thresholds on the arc properties such as the size and surface brightness will be selected as an arc candidate for further visual inspection. Such techniques can be used as complementary methods to deep learning. Currently, both techniques may suffer from many false positive detections which commonly include tidally interacting galaxies, artifacts, ring and spiral galaxy. The hope is that with more developed training datasets, deep learning algorithms can resolve such false positive cases (see Figure 2.6 and Section 2.5).

Other researchers (Petrillo et al., 2017; Jacobs et al., 2017; Lanusse et al., 2017) also find deep learning a suitable solution for finding gravitational lenses. Lanusse et al. (2017) use residual ConvNets with 46 layers. Residual ConvNets are modified ConvNet that do not suffer from layer saturation as ordinary ConvNets do. After adding more than 50 layers, the accuracy of ordinary ConvNets no longer improves and the training becomes more challenging. He et al. (2016) were able to overcome this issue by providing residual maps in between layers, which has been employed by Lanusse et al. (2017). They have simulated LSST mock observations in a single band and have trained and tested their network on these images. Jacobs et al. (2017) have trained their ConvNet using multiple color bands and have applied it to Canada-France-Hawaii Telescope Legacy Survey. Petrillo et al. (2017) have searched for lenses in Kilo Degree Survey by training their ConvNet on cataloged luminous red galaxies.

In our independently developed work, we focus on the morphology of the lenses and only rely on one color band, similar to Petrillo et al. (2017) and Lanusse et al. (2017). Our lens simulation method is very similar to Petrillo et al. (2017) where we both merge simulated arcs with real images of galaxies to preserve the complexity of the physical data. In contrast to others, we do not discriminate against different sources found in the COSMOS field. That is, artifacts, starts, and other sources have been included in our training dataset so LensFlow can be directly applied to fields without a need for a catalog with galaxy type information.

The deepness of our ConvNet is comparable to Petrillo et al. (2017) and Jacobs et al. (2017) but it is shallower than Lanusse et al. (2017). As mentioned in Jacobs et al. (2017), the morphology of lenses are much simpler than the morphology of daily objects and human faces which extremely deep ConvNets are developed for. However, the cost to performance ratio of ConvNets with varying deepness has not been studied yet. The effectiveness of deeper ConvNets cannot be compared between ours (and Petrillo et al., 2017) and Lanusse et al. (2017) since this work did not apply their algorithm to physical data. However, Lanusse et al. (2017) studied the change in the performance of their ConvNet by varying the Einstein radii and signal-to-noise ratio of their lenses.

A catalog of the strong gravitational lenses in the COSMOS field has been previously generated (Faure et al., 2008) by looking at early type bright galaxies in the redshift range of $0.2 \leq z \leq 1.0$ in specific environments and by visually inspecting and cataloging sixty high and low-quality lens candidates. In contrast, we have examined 236 thousand sources in the *HST*/ACS i-band of the COSMOS field. In this paper, we reported our sample of gravitational lenses and presented an introduction to neural networks including ConvNets. Furthermore, we laid out the procedure for constructing simulated images for training and testing LensFlow. The architecture of LensFlow, its performance on test data constructed from real lenses were also presented. Finally, we used LensFlow to identify new lens candidates using *HST* data. Scanning all of the *HST*/ACS images in the COSMOS field roughly took 140 seconds on one GPU with 3840 NVIDIA CUDA cores. This corresponds to scanning 1.7 thousand 100×100-pixel images per second (or equivalently 4.2 $\mathrm{arcmin}^2 s^{-1}$ for the *Hubble ACS* camera). This speed is suitable for all-sky surveys and the computation time can be reduced further by increasing the number of employed GPUs.

2.8 Remarks on the LensFlow Code

We have developed LensFlow in the Wolram Language (a.k.a Mathematica), using its image processing and state-of-the-art machine learning functionalities. The main notebook is called LensFlow which contains all the deep learning portions of the code. Other notebooks are used for lens simulation, image normalization, etc. If the user doesn't have access to Mathematica, they can download CDF Player for free to view the code. We will also provide a PDF of the main notebook with documentation alongside the code. From a practical perspective, it is important to store the images as JPEG files or other compressed formats since non-compressed image formants such as FITS occupy a significantly larger memory and storage volume and loading these images to memory will require much longer time. Training LensFlow during each phase on a GPU with 3840 NVIDIA CUDA cores takes less than 5 minutes for the training dataset discussed in this paper. Mathematica uses MXNet, so a trained network can be easily transfered to other languages. We initially developed our algorithm using TensorFlow, later, with adoption of Keras in Python 3.5.2 in Jupyter notebooks. These codes will be provided as extras. Even though they are not polished or fully developed, the codes is briefly documented in the Jupyter notebooks and may contain useful functions for data curation, helping the user to go from tiles to cutouts around extracted sources or automatically generating random arcs using LensTool.

2.9 Identified and Recovered Lenses

LensFlow was able to identify 92 lenses in the COSMSOM field, 46 of which of which were new and the rest were previously reported in Faure et al. (2008). The coordinates, Einstein radii, magnitudes of the brightest object, the LensFlow rankings of the lens among 236 thousand images, and their grade are reported in Table 2.2. The corresponding images are shown in Figure 2.7.

Table 2.2: Catalog of identified lenses by LensFlow. The first column corresponds to the image number in Figure 2.7.

Lens	Right Ascension (deg)	Declination (deg)	Einstein Radius (arcsec)	Magnitude (AB)	Absolute Rank In 236,000	Average Grade† A/B/C
1	+149.545323	+1.614164	1.82	22.16	1211	B
2	+150.440339	+1.754854	1.36	21.90	204	A
3	+150.180910	+1.714817	1.90	22.22	383	C
4	+150.066345	+1.772114	2.11	19.85	7	C
5	+149.489741	+1.736721	1.08	21.18	1744	C
6	+150.646190	+1.840283	1.91	20.28	1018	B
7	+150.091078	+1.935850	2.53	21.13	136	B
8	+149.632091	+1.882368	2.38	20.41	260	B
9	+150.670701	+2.091367	1.30	21.19	458	B
10	+149.894802	+2.109357	0.72	20.44	845	B
11	+149.856184	+2.112118	2.33	21.61	1321	B
12	+150.549644	+2.140845	0.91	22.61	584	B
13	+150.259607	+2.209858	0.88	20.16	1275	B
14	+150.117743	+2.266765	0.86	20.34	1597	B
15	+149.730719	+2.147258	1.82	20.62	21	B
16	+149.644851	+2.135518	1.74	21.55	189	B
17	+150.656039	+2.447838	1.51	20.81	308	A
18	+150.411971	+2.308876	1.86	19.93	1069	B
19	+150.095108	+2.300498	0.43	18.57	1593	C
20	+150.085701	+2.297656	0.91	21.86	300	B

Continued on next page

Lens	Right Ascension (deg)	Declination (deg)	Einstein Radius (arcsec)	Magnitude (AB)	Absolute Rank In 236,000	Average Grade† A/B/C
21	+150.085616	+2.364097	1.79	18.72	506	B
22	+150.106308	+2.432955	1.82	21.34	1674	C
23	+149.961446	+2.349389	1.16	21.60	208	B
24	+149.628310	+2.354862	0.88	21.82	1820	B
25	+149.722715	+2.428631	1.19	21.60	359	C
26	+150.571395	+2.506658	1.55	20.18	66	C
27	+150.624611	+2.540319	0.88	18.97	910	C
28	+150.694125	+2.547939	1.87	21.52	292	C
29	+150.317117	+2.531471	2.71	20.62	1451	B
30	+150.141624	+2.464563	1.65	20.82	86	B
31	+150.063881	+2.605824	1.47	21.50	979	B
32	+149.878942	+2.574346	0.97	21.78	573	A
33	+149.542116	+2.495012	2.23	19.03	1505	B
34	+150.747020	+2.666027	1.96	21.36	153	C
35	+150.548642	+2.766168	0.84	18.91	1595	B
36	+150.329391	+2.671669	2.44	21.72	1027	B
37	+150.284903	+2.674951	1.53	19.08	321	A
38	+150.250216	+2.763947	1.60	20.72	1515	B
39	+150.101284	+2.703268	1.74	22.58	932	B
40	+150.217548	+2.659542	1.11	23.18	1567	C
41	+149.855932	+2.650953	0.87	21.97	1345	B
42	+149.621847	+2.733148	2.33	21.31	1319	B
43	+150.644507	+2.808898	1.02	21.94	1098	B
44	+150.443480	+2.847808	1.55	21.66	1442	C
45	+150.104053	+2.844371	2.24	20.66	222	B
46	+149.611769	+2.809775	2.21	22.29	328	B
47*	+150.052500	+2.337500	0.90	19.28	2470	A
48*	+150.057917	+2.380278	1.65	18.89	910	B
49*	+150.076667	+2.645833	0.40	23.60	392	A
50*	+150.159167	+2.692500	0.74	20.39	13610	A
51*	+150.198333	+1.839722	0.70	20.65	2916	A

Continued on next page

Lens	Right Ascension (deg)	Declination (deg)	Einstein Radius (arcsec)	Magnitude (AB)	Absolute Rank In 236,000	Average Grade† A/B/C
52*	+150.205000	+1.857778	2.22	19.61	693	C
53*	+150.210833	+2.816944	1.90	21.72	5333	A
54*	+150.236250	+2.207222	1.20	18.70	4041	B
55*	+150.352083	+1.855833	0.84	22.43	5125	B
56*	+150.570000	+2.498611	1.96	19.98	3521	B
57*	+150.614583	+2.080833	1.62	21.94	1495	C
58*	+150.725000	+2.241667	1.54	18.85	5783	B
59*	+149.494167	+2.256944	0.35	22.27	3868	B
60*	+149.737500	+1.996944	2.15	20.05	1164	C
61*	+149.811250	+2.205278	1.86	23.25	4700	C
62*	+149.840417	+2.110556	0.80	20.34	9218	A
63*	+149.949167	+2.797778	2.55	19.83	1	C
64*	+150.040417	+2.415278	2.63	19.49	8071	B
65*	+150.196250	+2.491944	2.00	19.56	13476	A
66*	+150.211667	+2.065833	0.66	22.31	1197	B
67*	+150.232083	+1.639167	1.05	20.86	616	A
68*	+150.272083	+2.758611	1.00	20.38	3204	B
69*	+150.334167	+1.764167	1.28	21.31	1300	B
70*	+150.450417	+2.390278	1.43	18.81	216	C
71*	+150.535417	+2.239444	1.59	20.06	8647	B
72*	+150.584167	+2.393056	1.04	20.99	283	B
73*	+150.587917	+2.577778	1.57	19.35	9564	C
74*	+150.650000	+2.801944	1.27	20.15	3865	B
75*	+149.450000	+1.923333	2.21	22.15	1236	A
76*	+149.461250	+1.938611	0.73	19.19	9936	B
77*	+149.467083	+2.349167	0.98	20.27	5965	B
78*	+149.475417	+1.997778	1.33	22.23	204	B
79*	+149.523333	+2.070278	1.61	23.00	3764	B
80*	+149.528333	+1.969167	1.50	19.14	2442	B
81*	+149.624583	+1.626111	2.97	19.95	12066	B
82*	+149.629167	+1.725556	1.17	21.06	2092	A

Continued on next page

Lens	Right Ascension (deg)	Declination (deg)	Einstein Radius (arcsec)	Magnitude (AB)	Absolute Rank In 236,000	Average Grade† A/B/C
83*	+149.672500	+2.779444	0.93	19.65	177	B
84*	+149.733750	+2.798611	2.97	19.54	5951	C
85*	+149.870833	+1.764722	1.56	19.99	200	B
86*	+149.879583	+2.041389	1.48	19.20	894	B
87*	+149.883333	+2.171667	0.68	21.92	51	B
88*	+149.902917	+2.605833	2.40	19.14	7344	C
89*	+149.912917	+2.512222	0.70	20.08	93	B
90*	+149.918333	+2.548056	1.53	19.45	186	B
91*	+149.929583	+2.471111	1.64	19.43	2684	C
92*	+149.998750	+2.063333	1.20	21.62	44	B

† Grade A corresponds to images that are clearly a strong gravitational lens. Grade B lenses correspond to images that are most likely a lens, but there is a chance they could also be artifacts, noise, structures in elliptical galaxies, satellite galaxies, tidally interacting galaxies, etc. Grade C lenses consist of images that are most likely not a lens, but there is a chance they might be lensed.

*These marked lenses previously cataloged by Faure et al. (2008).

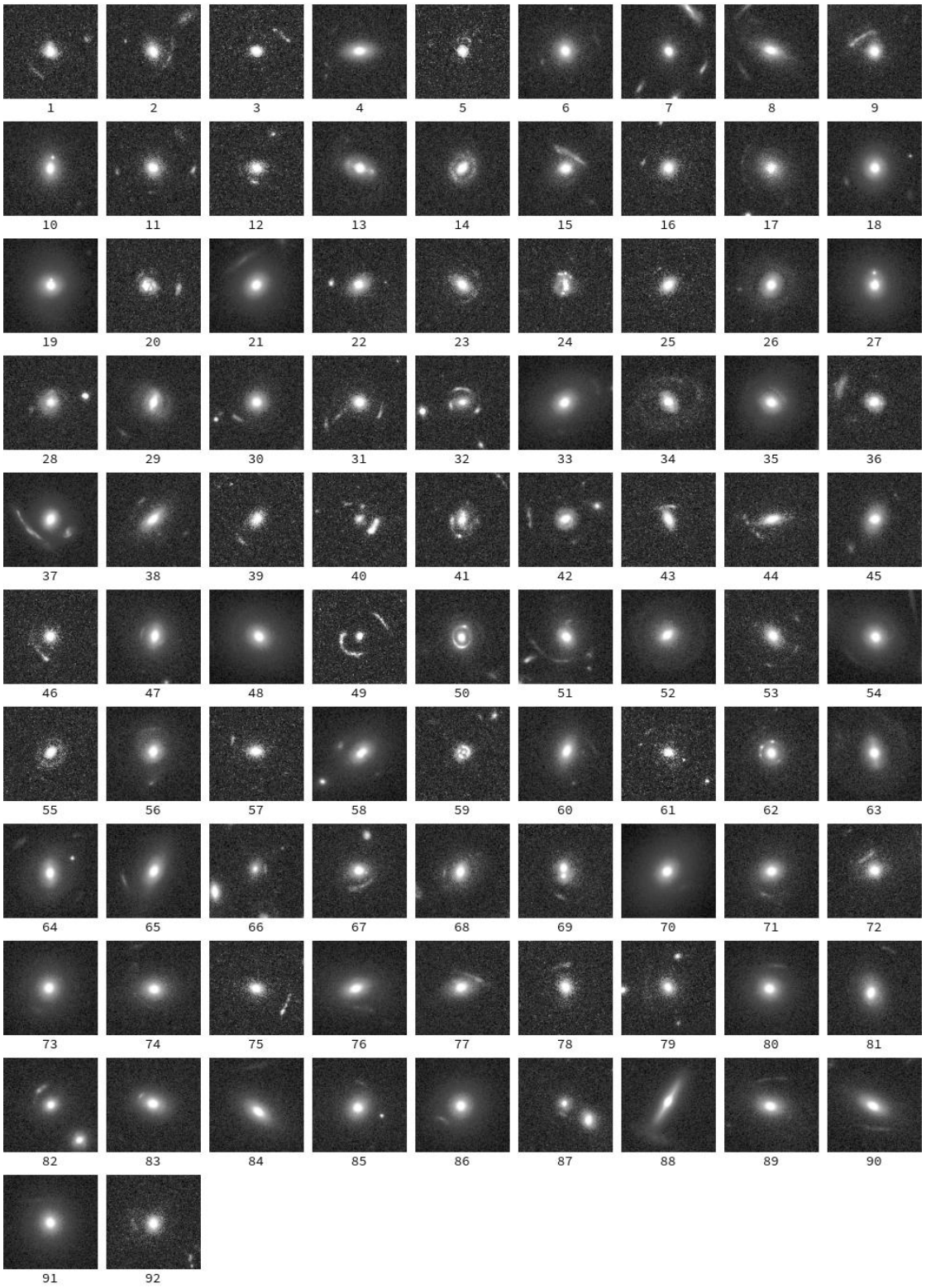

Figure 2.7: Identified COSMSOS lens candidates by LensFlow. These candidates are cataloged in Table 2.2.

Chapter 3

Advanced Lens Mining

3.1 Applications in Cosmology

Upcoming large sky surveys that span many thousands of square degrees on the sky will embed thousands of strong gravitational lensing systems among billions of galaxies that will be detected in such surveys. While time delays with individual lenses have allowed a measurement of the Hubble constant, lensing statistics such as the magnification factor distribution are a useful probe of cosmological parameters. Such statistical studies require unbiased sample of lenses and their selection function capturing the detection probability against survey parameters such as the depth and image resolution. In this work, borrowed from our manuscript (Pourrahmani et al., forthcoming in 2020), we present a set of machine learning algorithms capable of generating, mining, and synthesizing under-sampled events. With less than 50 previously known *HSC* lens examples, our Generative Adversarial Network (GAN) was capable of learning topological and statistical features of these populations to generate realistic lenses which were used to train our specialized deep mining algorithm. This trained neural network successfully identified 42 lens candidates in Subaru 90 $\deg^2$ wide

field. We measured the selection function of our lens mining neural network using a realistic and real-time lens synthesizer by scanning the lens parameter space. Our mining algorithm readily extends to upcoming astronomical large-area surveys and allow a mechanism to not only select lens candidates but also capture their selection function.

Large area multi-band observations are ideal for the identification of strong gravitational lensing events. Here, we developed a state-of-the-art machine learning algorithm to identify such events across wide-area surveys that is applicable on a small number of training examples. We achieve this by using deep convolutional Generative Adversarial Networks (GANs) (Goodfellow et al., 2014), as implemented by Radford et al. (2015) for other applications, to acquire more examples for training a classifying Convolutional Neural Networks (CNN) for the first time.

Our GAN consists of a generator which starts with a random latent vector of size 64. Through four consecutive transposed convolution blocks, the dimensionality of the latent vector is gradually increased to $3 \times 64 \times 64$ (see Figure 3.1). We use EZnet, our densely connected classification neural network as a discriminator details of which are discussed in the Chapter 4. The generator produces fake images which are bundled with random real images and passed to the discriminator. The discriminator will attempt to classify fake and real images with an associated loss function. The loss function is back-propagated through the discriminator and the generator. The discriminator parameters are updated to minimize the loss while the generator parameters are updated to maximize it, a competition that will improve the generator at creating realistic images while the discriminator improves at identifying the fake ones. We pre-trained our GAN on generic survey images before training on lenses to improve the results.

In short, our developed library includes a flexible and easy to train deeply and densely connected CNN that is used for image classification, called EZnet. In addition, our software package is supplemented with GAN augmentation to overcome the limited size of the training

dataset, and a specialized input normalization method for large range of pixel values such as in astronomical images. We have also developed and implemented the concept of negative learning, improving the ability of our classifier to identify negative classes rather than focusing on classifying both classes with the same emphasis. A new performance measure is also introduced that is more suited for mining rare events in large datasets with machine learning, as it is the case with gravitational lenses.

Our CNN is parallelizable on multiple GPUs and CPUs for training and for fast scanning powered by the deep learning library PyTorch (Paszke et al., 2017). Our scan rate is at 50 image cutouts per second on a single GPU which roughly translates to 15 minutes per square degree per band over a high-resolution image with pixel scale and resolution similar to that of *HSC* (i.e. 0.168 arcsec/pixel), making our mining algorithm ideal for future surveys such as *LSST* over hundreds of square degrees.

As an application of our selection algorithm, we make use of the high-resolution wide-area observations by Subaru's new HyperSuprime Cam instrument (*HSC*). We use the three-bands i, r, and g from *HSC*-PDR1 (Aihara et al., 2018) providing the color information needed for lens identification. The individual steps for normalizing imges are discussed in Chapter 4. Our new algorithm, once applied to the high-resolution data discussed above, managed to identify 42 lens candidates 37 of which were not previously cataloged by the Subaru team (Sonnenfeld et al., 2018).

3.1.1 Lens Selection Function

All selection algorithms suffer from an intrinsic biased selection which if not treated properly can result in false cosmological parameters obtained from skewed distributions. Knowing the selection function can counterbalance these biases and properly link observational catalogs and theoretical cosmological predictions. To extract the selection function of our lens iden-

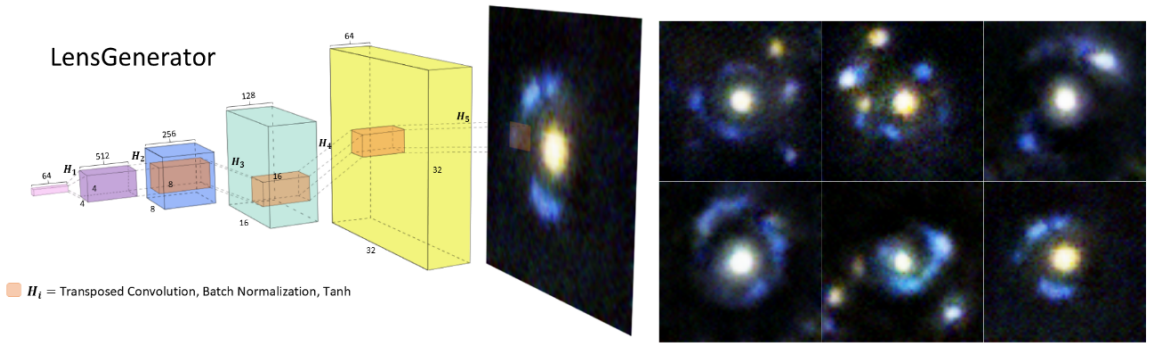

Figure 3.1: Left: Lens Generator Architecture. A series of transposed convolutions are used to upsample a latten vector of size 64 to a 64×64 RGB image. This generator, hand in hand with EZnet as a discriminator is used for image augmentation for training. Right: Examples of GAN augmented lenses. Generated lenses morphologically and statistically resemble images of real lenses but are diverse enough to be suited for training purposes as augmented images.

tifying algorithm, we explored the space of different lens configurations and obtained the corresponding lensing probabilities of synthesied lenses by our realistic and real-time lens synthesizer software, details of which are explained in Chapter 4. Multiple 2D cross-sections of our multidimensional selection function is plotted in Figure 3.3.

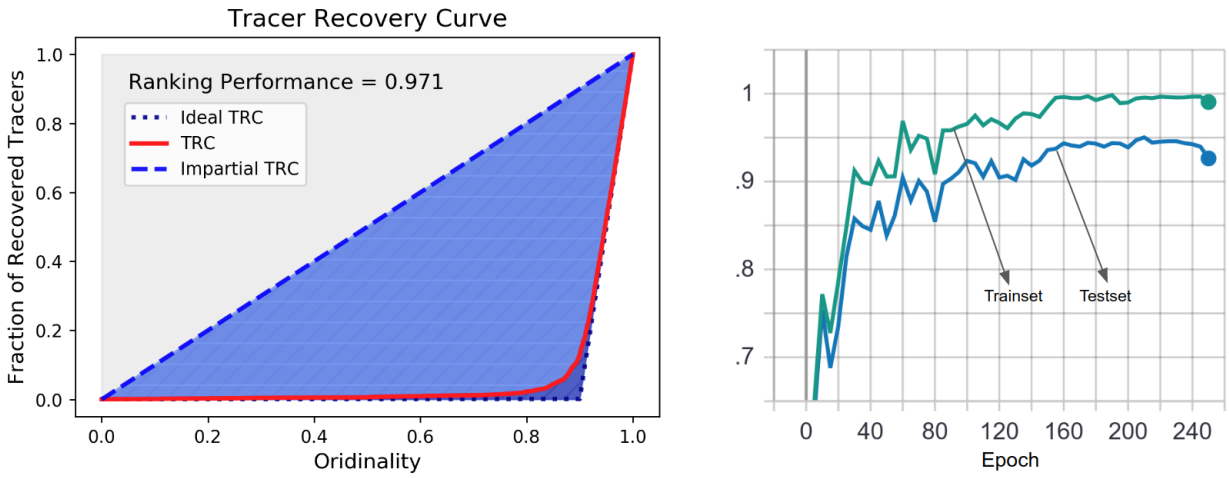

Figure 3.2: Left: An example of tracer recovery curve (TRC) on a testset. The fraction of recovered positive tracers (lenses) implanted in a pool of negatives (ordinary galaxy images) has been plotted as a function of ordinality (ranked probabilities). Ranking performance is defined as the area of the lighter blue region over the area of the darker blue region which includes the lighter region. Right: Example of ranking performance on trainset and testset over 250 epochs of training. Ranking performance is a more suitable measure of performance for deep mining algorithms as opposed to other metrics such as weighted accuracy.

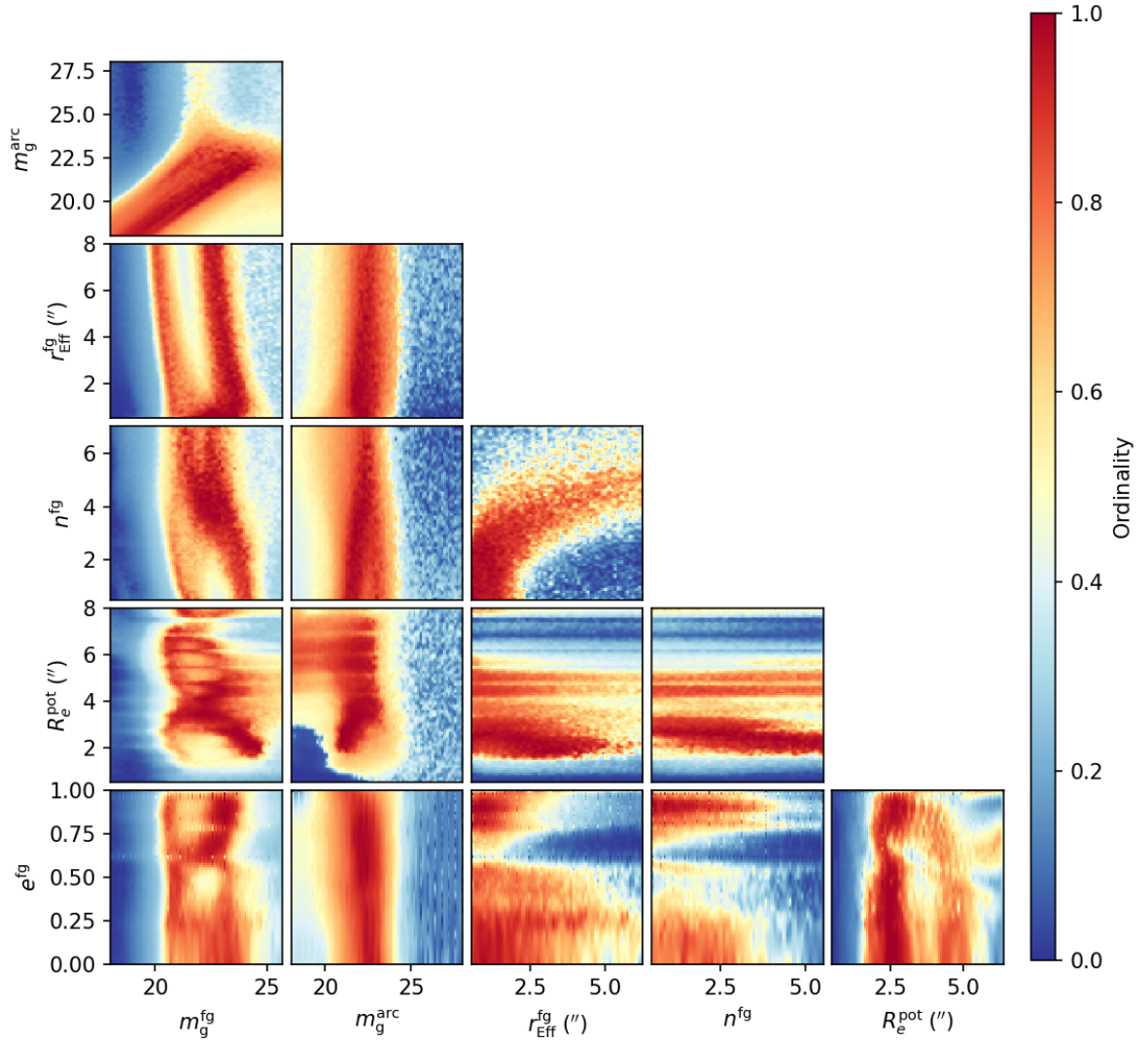

Figure 3.3: 2D-cut in the lens selection function. The sensitivity of our lens mining ConvNet has been obtained for a range of lens configurations. For visual purposes, we only show dependencies over all pairs of the following quantities, keeping other parameters fixed: foreground magnitude in g-band $m_g^{\rm fg}$, arc magnitude in g-band $m_g^{\rm arc}$, the effective radius of the foreground $r_{\rm Eff}^{\rm fg}$, foreground Sersic index $n^{\rm fg}$, Einstein radius of the gravitational potential $R_{\rm e}^{\rm pot}$, foreground ellipticity $e^{\rm fg}$. Values of other paramters are listed in Figure 4.2.

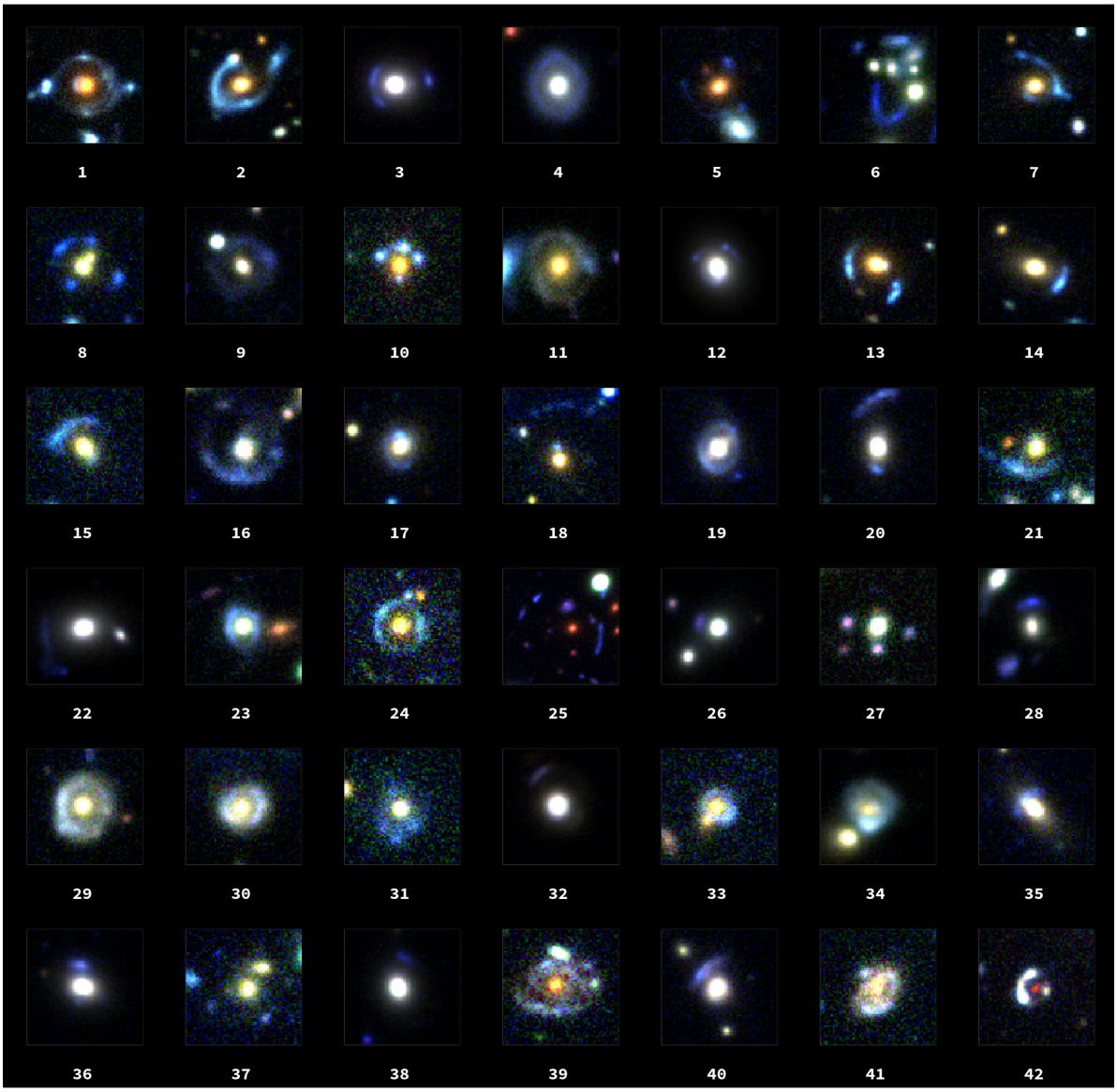

Figure 3.4: Gravitational lens candidates identified from multi-band observations in the Subaru *HSC*-PDR1 wide fields using our deep mining algorithm. The identified candidates show a variety of lens configurations as demonstrated by our GAN generated training set. The color information allows the algorithm to more easily separate the foreground lens and background system. These candidates are listed in Table 2.

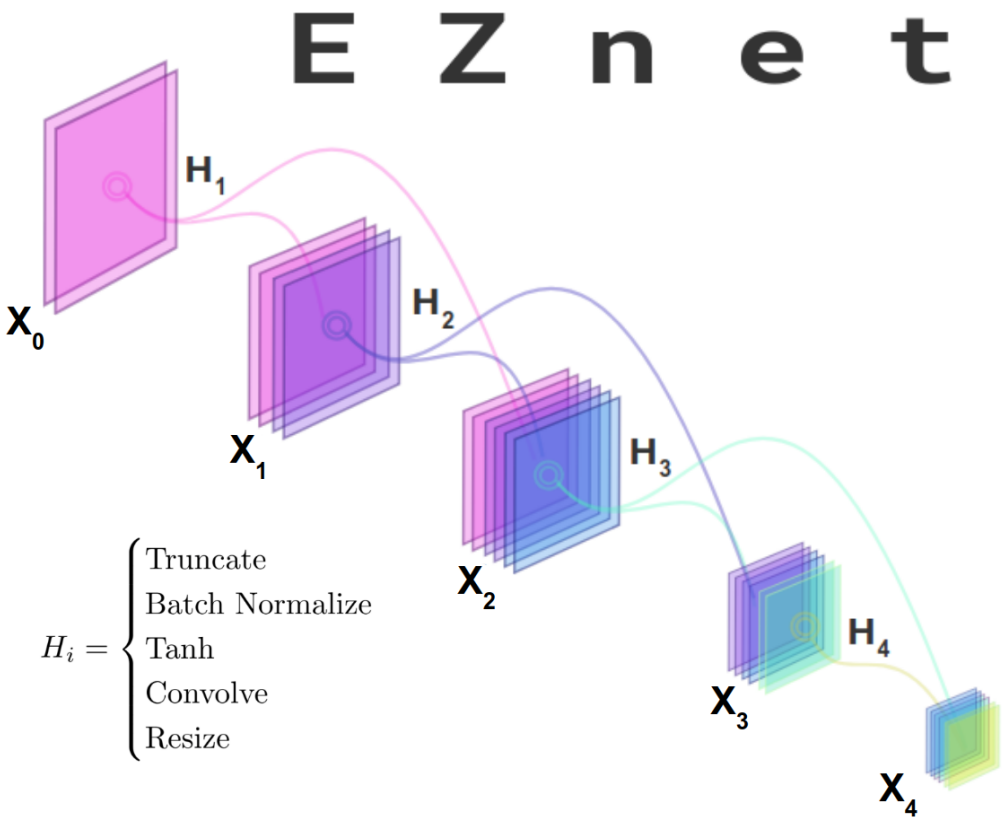

Figure 3.5: Schematic architecture of EZnet. H_i applies batch normalization, Tanh nonlinearity, and convolution to x_{i-1} and appends the resulting feature maps to a fraction of its input, i.e. to x_{i-1}. All feature maps are resized (usually downsampled) via interpolation to ensure a uniform output size. Before passing it to the next block, the first few feature maps are removed such that the next block would only have access to inputs of the b previous blocks where b is the branching rate (e.g. 2 for the diagram above). By appending two fully connected layers and two sigmoid neurons to a sequence of these blocks, EZnet architecture can be used as a binary classifier.

Chapter 4

Deep Lens Mining Applications to Hyper Suprime-Cam Subaru Strategic Program Data Release

4.1 Data processing

We use the wide-area images in the i-, r-, and g-bands of the first public data release of Subaru HyperSuprime Cam (*HSC*-PDR1 by Aihara et al. (2018)). With the help of the *HSC*-PDR1 catalog, we isolated 64×64-pixel cutouts of sources between 15 mag and 23.5 mag in i- and r-band (15 mag and 24.8 mag for g-band), totaling a 1.4 million RGB images. A percentile normalization is applied to each band where the lowest 0.5% and the highest 0.5% of the pixels were clipped (see our GitHub Gist for FITS handling). The resulting images were shifted and scaled to be bounded from 0. to 255. and stored as batches of $3 \times 64 \times 64$ `uint8` arrays to reduce storage space and increase processing speed. These images have the same format as a PNG file where the red, green, and blue channels correspond to i-, r-, and

g-band respectively. The same color representation was adapted for plotted images in this paper.

Our datasets consist of two classes, labeled as negative images and positive images. For the purposes of this paper, negative images are the ordinary cutouts from the survey while the positive images refer to lenses. Obtaining negative images is as easy as sampling from the cutouts. Positive images, on the other hand, must be heavily augmented since only a handful of *HSC* lenses are known. Section 4.3 discusses lens augmentation with GANS.

4.2 Trainset and negative learning

Since for our circumstances, the number of positive images is extremely limited while the number of negative images is orders of magnitude higher, our trainset used for training our neural net is slightly unconventional with many advantages. We have randomly isolated 50,000 negative images from the cutout pool and during training, every positive image in a mini-batch is accompanied by a randomly drawn image from this negative pool. We refer to this technique as negative learning. This means our trainset is dynamical, changing from epoch to epoch. With negative learning, the network learns to identify positive images by better generalizing the negative ones and getting exposed to sub-classes of the negative images such as spiral galaxies, elliptical galaxies, etc. Usually, a validation dataset is used to optimize its architecture while a trainset is used to optimize the parameters of a neural network. Rather than creating a validation dataset from our limited data, we have used CIFAR-10 to optimize the architecture of our network.

The number of known lenses observed by an instrument in a given field is often small. Only 51 grade A and B lenses were previously discovered in the *HSC*-PDR1 by the SuGOHI project (Sonnenfeld et al., 2018). To improve generalization, our network is equipped with

an augmentation layer that randomly applies transformations from the symmetry group of a square (horizontal, vertical, diagonal, and anti-diagonal flips and 0-, 90-, 180-, and 270-degree rotations). However, this is not sufficient and further augmentation is required.

4.3 LensGenerator: A GAN lens augmentation library

For further augmentation, we have resorted to using an implementation of deep convolutional Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) as presented by Radford et al. (2015). Our GAN consists of a generator which starts with a random latent vector of size 64. Through four consecutive transposed convolution blocks, the dimensionality of the latent vector is gradually increased to $3 \times 64 \times 64$ (see Figure 3.1). We use the EZnet architecture described in Section 4.5 as the discriminator for the GAN. The generator generates fake images which are bundled with random real images and passed on to the discriminator. The discriminator will attempt to classify fake and real images with an associated loss function. The loss function is back-propagated through the discriminator and the generator. The discriminator parameters are updated to minimize the loss while the generator parameters are updated to maximize it, a competition that will improve the generator at creating realistic images while the discriminator improves at identifying the fake ones.

Due to lack of enough positive images, we have initially pre-trained our GAN on negative images and then re-trained on positive images. 200 synthesized images then were extracted from the GAN. GANs often suffer mode collapse, resulting in less diverse output images, which we overcome by 10 cycles of retraining, totaling to 2,000 augmented images which visually look distinct (see Figure 3.1 for examples). For further details, we refer the reader to LensGenerator GitHub Page.

4.4 LensCrafter: A realistic and fast lens synthesizer

LensCrafter is a separate sub-project where we developed a realistic gravitational lens simulator that can synthesize a variety of lens configurations in real-time for multiple bands. LensCrafter models a Sersic background and a foreground and is capable of calculating the lensed image of the background for a given potential function. It also accounts for details such as observational noises and PSF. LensCrafter GitHub repository is publicly available and we will discuss the details of its algorithm in this section. See Figure 4.1 for simulation examples of variations of a lens configuration. Although this could be used for generating more training examples of lenses, instead, we only use this for extracting our selection function. As discussed in Section 3.1.1, due to control over a wide range of parameters, we are able to use LensCrafter to estimate the selection function of our lens identification algorithm plotted in Figure 3.3.

In detail, our recipe for synthesizing a gravitational lens system starts with generating the background astronomical object from a Sersic profile whose light is being lensed, known as the source (`src_map`). We also generate a deflector (`dfl_map`) caused by the matter in the foreground galaxy responsible for the lensing which is modeled by an SIE (singular isothermal ellipsoid) potential. Lastly, a foreground (`fgr_map`) is generated by a different Sersic profile capturing the emission of the foreground galaxy.

The `src_map` is the pixelated representation of the source plane, i.e. the 2D map of the source emission as it would appear to the observer in the absence of a deflector. The `dfl_map` and the `fgr_map` are the pixelated representations of the matter and the light in the lens plane, respectively. The `deflection` function receives the `dfl_map` and the `src_arr` and outputs the image (`img_map`) of the perturbed source by the deflector. The `compose` function superposes the generated foreground galaxy on top of the image to construct what we refer to as the emission (`ems_map`) which represents the light intensity of the astronomical

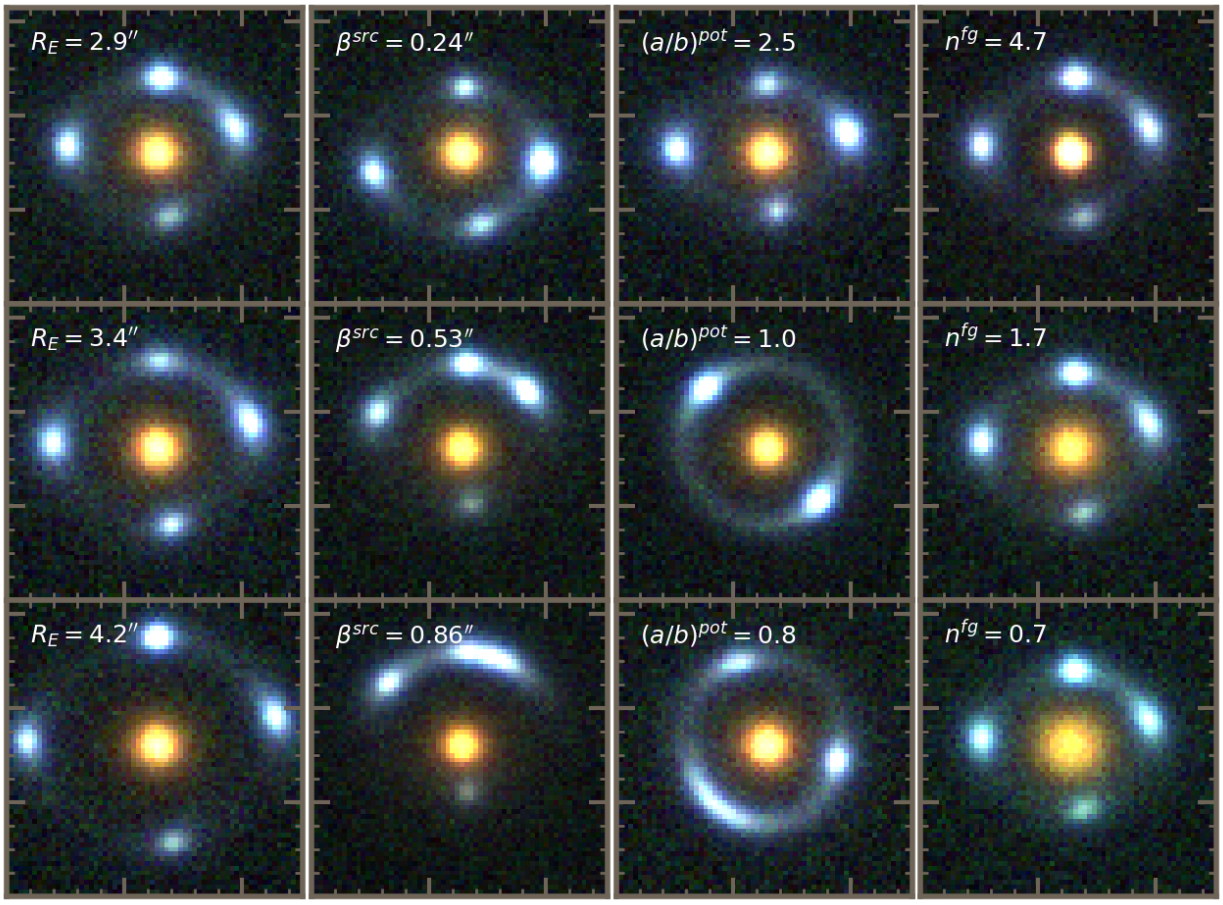

Figure 4.1: Example synthesised lenses by LensCrafter. It is used for probing the sensitivity of our lens mining neural net. Aside from the mentioned quantity, all lenses share the same property as the reference lens on the upper left corner which is sampled from Figure 4.2. In the first column the Einstein radius increases from top to bottom, In the second column the angular distance between the centers of the foreground and the source is increasing. The third and fourth columns show variations in the ellipticity of the lensing potential and Sersic index of the foreground galaxy respectively. LensCrafter can simulate lenses in real-time and includes an interactive widget suitable for educational purposes.

```
img_map = deflect(dfl_map, src_map)
ems_map = compose(fg_map, img_map)
syn_map = shot_noise(ems_map)
syn_map = convovle_psf(syn_map, psf_map)
syn_map = observe(syn_map)
png_map = percentile_normalization(syn_map)
```

	g Magnitude	r-g Flux Ratio	i-g Flux Ratio	Effective Radious ($''$)	Sersic Index	Ellipticity	Rotation ($^\circ$)	Offset ($''$)	Offset Angle ($^\circ$)
Source	18-27	1.6	2.6	6	7	0.35	-85	0.17	+45
Foreground	18-27	5.5	20.5	0.5-6.0	0.5-6.0	0.10	+5	0.00	0

Einstein Radius ($''$)	Aspect Ratio	Rotation ($^\circ$)	Offset ($''$)	Offset Angle ($^\circ$)
0.1-6.0	1.8	+5	0	0

Figure 4.2: *Top*: LensCrafter pseudocode. *Bottom*: LensCrafter example parameters. Ranges indicate variations in parameters used for generating the selection function in Figure 3.3.

objects without the observational effects. We sample the `ems_map` with a Poisson distribution using the `shot_noise` function to create a synthesized map (`syn_map`). The `convolve_psf` function convolves the synthesized array with an *HSC* PSF (`psf_map`) and the `observe` function adds instrument noise sampled from a normal distribution whose mean and variance were obtain from background of a patch of sky for each band. For visualization, we apply the same `percentile_normalization` used for our neural network that outputs an array in the PNG format (`png_map`). The parameters used for generating the upper left lens in Figure 4.1 are tabulated in Figure 4.2 which also includes the range of values used for obtaining the selection function in Figure 3.3. This algorithm is captured in the shown code block.

4.5 EZnet: A flexible densly connected ConvNet

Deeper convolution networks perform better at classification tasks; however, vanishing-gradient in the backpropagation and vanishing input information in the feedforward process imposes a limit on the depth of these networks. By overcoming these limitations, ResNet (He et al., 2015) was the most successful architecture with an impressive performance on many

classification benchmarks by surpassing one hundred convolution layers. Such deep networks have many parameters which require a large number of training data and are difficult and expensive to train. With a significantly simpler architecture, DenseNet was able to alleviate the vanishing-gradient problem and strengthen feature propagation with a substantially reduced number of parameters while producing the same or better classification performance compared to ResNet and all other competing models. For comparison, an architecture of DenseNet with only 0.8 M parameters achieves an error rate of 4.5% on augmented CIFAR-10 while a ResNet with 1.7 M parameters achieves 6.5%. Each layer of DenseNet uses the feature-maps of all preceding layers as inputs and its own feature-maps are used as inputs for all successive layers. For a better understanding of DenseNet as well as other famous architectures, we refer the user to their paper (Huang et al., 2016). DenseNet, however, does have some undesirable properties. Since all layers are connected to the preceding layers, the number feature-maps grow quickly with the number of layers. To prevent this, a transition blocks has been introduced to reduce the number of feature-maps to a smaller number via a convolution layer followed by a pooling layer. The output of the transition bock is then passed to the next DenseNet. Their paper demonstrates the results for 3 DenseNets linked with two transition blocks and a final classification layer.

Inspired by DenseNet, we propose a new architecture that resolves the same issues but with a few additional advantages. Unlike DenseNet, our architecture has a very simple and uniform structure end to end, is easily modifiable and has an fewer number of parameters while performing better at classification on smaller and unbalanced datasets, though DenseNet performs better on larger trainsets. We refer to the architecture of our network as EZnet which is constructed from a sequence of interconnected convolution blocks. Each block i will truncate the first b inputs (feature-maps) and apply batch normalization, followed by nonlinearity, followed by convolution of size k_i. The output of the convolution is appended to truncated inputs. Using interpolation, a resize layer converts all feature maps to the same and smaller size before passing it to the next block, resulting in an output size of $H_{i+1}W_{i+1}$ for

all outputting feature maps. It is thanks to this interpolation layer that blocks can gradually decrease in size while being able to stay connected and reuse a few previous feature-maps, in contrast to DenseNet. Figure 3.5 further clarifies this architecture. Conventional pooling methods may be faster than interpolation however the gained flexibility justifies the sacrifice for our porposes. The PyTorch implementation of EZnet and other utilities for employing EZnet for deep mining are available on GitHub.

We used the architecture tabulated under Table 1 and append two fully connected layers and two sigmoid neurons that output the probability of the input image belonging to the negative or the positive class. The architecture may be modified by providing a different table. We hope that the ease of use, flexibility, reliability, speed, and trainability of EZnet on small datasets encourages others to use it for different applications, including but not limited to galaxy classification, specimen classification in biology, particle classification in high energy physics, classification of spectroscopy signals in astronomy and for gravitational wave detections, wherever obtaining or simulating training data is expensive.

4.6 Training and performance evaluation

The objective of the training process is to minimize misclassification. We use a cross-entropy loss function and we optimize it using stochastic gradient descent with Nesterov momentum initialized to 0.9, weight decay of 10^{-4}, and an initial learning rate of 0.1. Learning rate is later reduced to 0.01 at the epoch of 150 and further reduced to 0.001 at the epoch of 220 to better fine-tune parameters. We stop training at the epoch of 250 (or earlier if signs of overfitting appear). This achieves a better accuracy compared to Adam optimizer, is stable, and is not as sensitive to the values of the mentioned hyperparameters (Huang et al., 2016).

The objective of deep mining is to mine a few positives in a pool of negatives by selecting

Table 4.1: Tabulated Architecture of the LensFlow ConvNet.

Basic Block Index b	Kernel Size $(C \times H \times W)$	Output Size $(H \times W)$
1	$16 \times 5 \times 5$	50×50
2	$12 \times 4 \times 4$	40×40
3	$12 \times 3 \times 3$	32×32
4	$12 \times 3 \times 3$	24×24
5-10	$12 \times 3 \times 3$	16×16
11-20	$12 \times 3 \times 3$	12×12
21-40	$12 \times 3 \times 3$	8×8
41-50	$32 \times 1 \times 1$	1×1

Table 4.2: Tabular architecture of EZnet. EZnet consists of a chain of basic blocks. For each basic block, it is specified how many input feature maps will be truncated, how many convolution kernels are applied, whether the truncated inputs are concatenated with convolution outputs, and finally the uniform height and width of all output channels are specified. As indicated, we rapidly shrink the output size through the first few blocks to increase the speed.

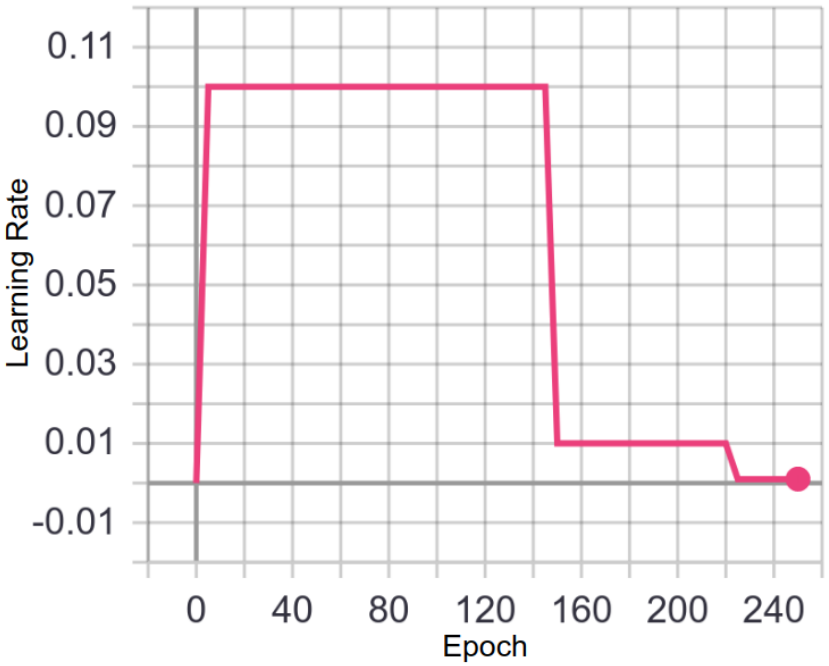

Figure 4.3: To fine-tune the model parameters, we reduce the learning rate. It has been shown by Huang et al. (2016) that stepwise decrease in learning rate can achieve slightly higher accuracy compared to ADAM optimizer which smoothly decreases the learning rate.

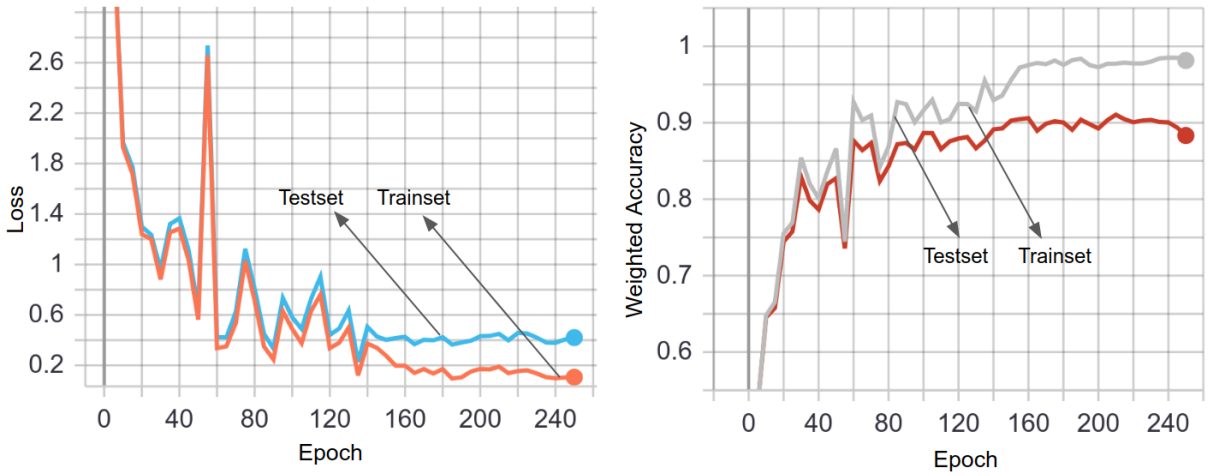

Figure 4.4: Examples of loss and accuracy for trainset and testset over 250 epochs of training. As shown in Figure 3.2, there is a better approach to measuring the performance of classifier if the goal is to mining positives in a large pool of negatives.

the points with the highest positive probabilities assigned by a properly trained model. This motivates a natural metric definition to measure the model's performance: how well a trained model can rank data points. To concretely define a ranking performance, we first must define ordinality. Ordinality is the position of a data point in an ordered list of data from lowest to highest positive probability, normalized by the size of the list. The absolute ordinality (i.e. position) of the first element of the list is 1 and absolute ordinality increments by 1 unless there is a ranking degeneracy for data points with the same positive probabilities (see this gist for details). Using this concept, we can define the tracer recovery rate as the number of recovered tracers (i.e. positive images) above a certain ordinality. Tracers are randomly seeded in a pool of negatives before being ranked by a trained model. The tracer recovery curve (TRC) of Figure 3.2 shows the recovery rate as a function of ordinality for our trained lens mining algorithm. The area between the TRC and the line of no discrimination normalized by the area of a perfect classifier captures the essence of the plot in Figure 3.2 which we will refer to as the ranking performance. Rather than using concepts such as probability, loss, weighted accuracy, we will use ordinality and ranking performance to measure the performance of our lens mining algorithm.

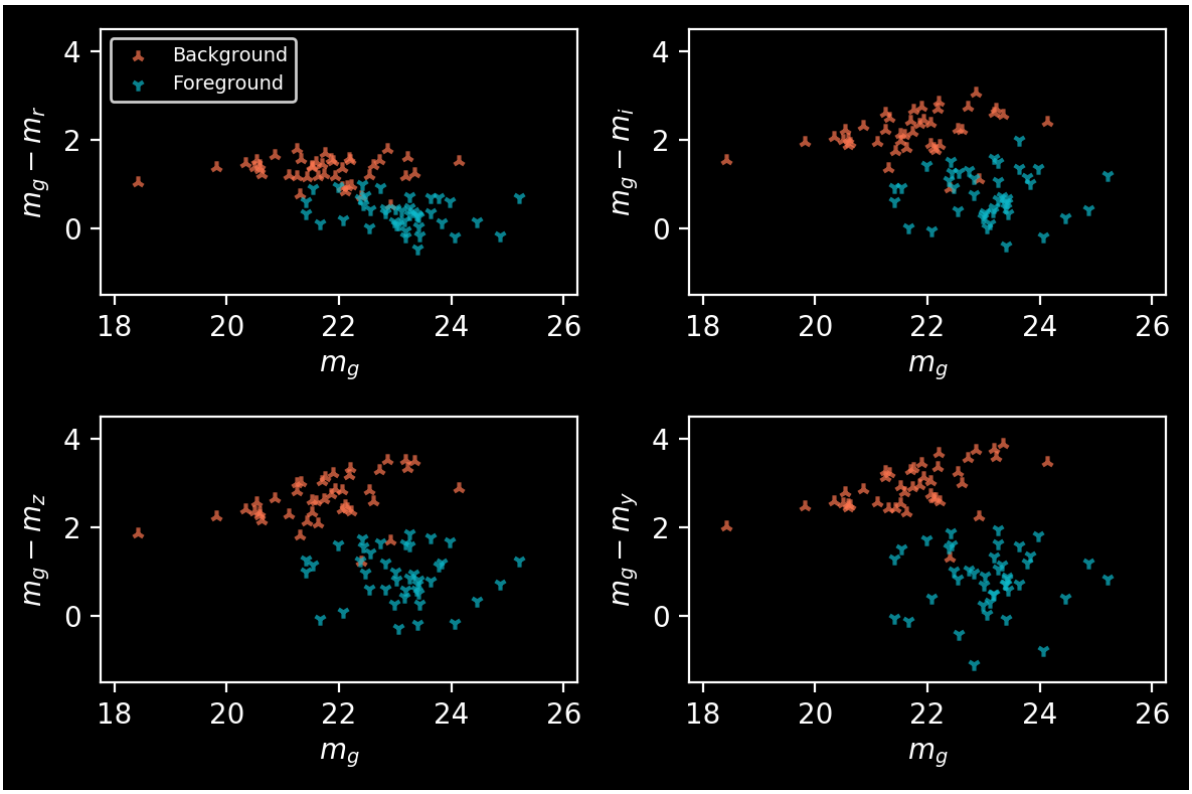

Figure 4.5: Color-magnitude diagram. Different color vs g-magnitude for the foreground lens and lensed images listed in 4.3

4.7 Results, catalog of discovered lenses

After training EZnet and scanning all cutouts, we have visually examined images with ordinality greater than 0.995 (i.e. top 0.5% images) and were able to identify 42 lens candidates majority of which were previously not cataloged by Sonnenfeld et al. (2018). We have shared this catalog in Table 4.3 and the corresponding images are shown in Figure 3.4. The different color-magnitude diagrams of these lenses have been plotted in Figure 4.5. Using these magnitudes, one can obtain the photometric redshift of the sources and their lens. The lens masses can be calculated using these redshift and the Einstein radii, resulting in a lens mass distribution. With the help of the selection function, a mass distribution for low redshift galaxies can be estimated, independent of other methods.

Table 4.3: Catalog of identified lenses by EZNET. The first column corresponds to the image number in Figure 2.7. Einstein radii and lens grades are also listed.

Lens	Right Ascension (deg)	Declination (deg)	Einstein Radius (arcsec)	Grade† A/B/C
1	+216.2043	-00.8894	3.25	A
2	+215.2654	+00.3720	2.20	A
3	+340.5898	+00.1957	2.44	A
4	+219.4564	+00.4944	2.45	A
5	+335.1725	+00.8201	2.99	A
6	+333.5790	+01.1769	2.54	A
7	+217.8079	-00.1037	2.26	A
8	+218.7266	-00.9496	2.71	A
9	+333.1764	+00.1892	2.38	A
10	+132.6942	+00.6515	1.59	A
11	+035.3941	-04.4112	2.80	A
12	+181.9302	-01.0653	2.21	A
13	+216.9515	+00.1663	2.73	A
14	+215.0377	-00.2429	2.88	A
15	+218.8296	-01.1101	2.68	A
16	+135.7236	+00.8684	3.31	B
17	+178.9889	-00.2107	1.75	B
18	+215.0287	+00.3112	4.09	B
19	+215.7069	-00.5532	1.95	B
20	+214.2848	+00.9822	2.35	B
21	+036.0038	-03.7738	2.65	B
22	+178.8366	+00.8846	3.94	B
23	+333.8429	+01.0912	1.27	B
24	+338.1610	-00.4261	2.29	B
25	+136.0186	+01.4212	7.49	B
26	+218.5273	-00.4848	1.83	B
27	+132.0146	+02.0579	3.13	B
28	+334.5879	-00.0316	2.41	C
29	+220.1407	-00.6051	2.30	C

Continued on next page

Lens	Right Ascension (deg)	Declination (deg)	Einstein Radius (arcsec)	Grade† A/B/C
30	+132.0751	+02.0876	1.79	C
31	+035.2166	-04.5833	2.65	C
32	+215.1524	-00.1260	3.88	C
33	+136.6667	+01.0630	1.25	C
34	+131.9165	+01.9710	2.06	C
35	+032.1952	-03.4577	1.45	C
36	+337.4952	+00.1038	2.18	C
37	+135.8591	-00.1686	3.18	C
38	+333.6634	+01.2666	2.89	C
39	+178.7919	-01.3283	2.52	C
40	+216.4902	+00.9385	2.19	C
41	+218.2470	-00.5127	1.86	C
42	+337.1984	+01.0535	1.35	C

† Grade A corresponds to images that are clearly a strong gravitational lens. Grade B lenses correspond to images that are most likely a lens with a the chance of being artifacts, noise, structures in elliptical galaxies, satellite galaxies, tidally interacting galaxies, etc. Grade C lenses are less likely to be lenses cannot be ruled out without higher quality observations or spectroscopic follow-ups.

*These marked lenses previously cataloged by Faure et al. (2008).

Bibliography

Abadi, M., Agarwal, A., Barham, P., et al. 2016, arXiv preprint arXiv:1603.04467

Agnello, A., Lin, H., Buckley-Geer, L., et al. 2017, ArXiv e-prints, arXiv:1702.00406

Aihara, H., Armstrong, R., Bickerton, S., et al. 2018, Publications of the Astronomical Society of Japan, 70, S8

Alard, C. 2006, ArXiv Astrophysics e-prints, astro-ph/0606757

Atek, H., Richard, J., Kneib, J.-P., et al. 2015, The Astrophysical Journal, 800, 18

Blandford, R., & Narayan, R. 1992, Annual review of astronomy and astrophysics, 30, 311

Bolton, A. S., Burles, S., Koopmans, L. V., Treu, T., & Moustakas, L. A. 2006, The Astrophysical Journal, 638, 703

Broadhurst, T., Benítez, N., Coe, D., et al. 2005, The Astrophysical Journal, 621, 53

Calanog, J. A., Fu, H., Cooray, A., et al. 2014, apj, 797, 138

Capak, P., Aussel, H., Ajiki, M., et al. 2007, apjs, 172, 99

Coe, D., Zitrin, A., Carrasco, M., et al. 2012, The Astrophysical Journal, 762, 32

Eigenbrod, A., Courbin, F., Vuissoz, C., et al. 2005, aap, 436, 25

Faure, C., Kneib, J.-P., Covone, G., et al. 2008, The Astrophysical Journal Supplement Series, 176, 19

Fu, H., Jullo, E., Cooray, A., et al. 2012, apj, 753, 134

Gavazzi, R., Marshall, P. J., Treu, T., & Sonnenfeld, A. 2014, apj, 785, 144

Goobar, A., Amanullah, R., Kulkarni, S. R., et al. 2017, Science, 356, 291

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., et al. 2014, arXiv e-prints, arXiv:1406.2661

Grogin, N. A., Kocevski, D. D., Faber, S., et al. 2011, The Astrophysical Journal Supplement Series, 197, 35

He, K., Zhang, X., Ren, S., & Sun, J. 2015, arXiv e-prints, arXiv:1512.03385

He, K., Zhang, X., Ren, S., & Sun, J. 2016, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778

Heymans, C., Van Waerbeke, L., Miller, L., et al. 2012, Monthly Notices of the Royal Astronomical Society, 427, 146

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. 2016, arXiv e-prints, arXiv:1608.06993

Jacobs, C., Glazebrook, K., Collett, T., More, A., & McCarthy, C. 2017, arXiv preprint arXiv:1704.02744

Jullo, E., Kneib, J.-P., Limousin, M., et al. 2007, New Journal of Physics, 9, 447

Kaiser, N., & Squires, G. 1993, The Astrophysical Journal, 404, 441

Kingma, D. P., & Ba, J. 2014, ArXiv e-prints, arXiv:1412.6980

Koekemoer, A. M., Faber, S., Ferguson, H. C., et al. 2011, The Astrophysical Journal Supplement Series, 197, 36

Komatsu, E., Dunkley, J., Nolta, M., et al. 2009, The Astrophysical Journal Supplement Series, 180, 330

Krizhevsky, A., & Hinton, G. 2009

Krizhevsky, A., Sutskever, I., & Hinton, G. E. 2012, in Advances in neural information processing systems, 1097–1105

Lanusse, F., Ma, Q., Li, N., et al. 2017, arXiv preprint arXiv:1703.02642

Lenzen, F., Schindler, S., & Scherzer, O. 2004, aap, 416, 391

Lotz, J., Koekemoer, A., Coe, D., et al. 2017, The Astrophysical Journal, 837, 97

Marshall, P. J., Lintott, C. J., & Fletcher, L. N. 2015, araa, 53, 247

More, A., Cabanac, R., More, S., et al. 2012, The Astrophysical Journal, 749, 38

More, A., Suyu, S. H., Oguri, M., More, S., & Lee, C.-H. 2017, apjl, 835, L25

More, A., Verma, A., Marshall, P. J., et al. 2016, mnras, 455, 1191

Nayyeri, H., Keele, M., Cooray, A., et al. 2016, The Astrophysical Journal, 823, 17

Nayyeri, H., Cooray, A., Jullo, E., et al. 2017, apj, 844, 82

Nielsen, M. 2016, Chapter 3 (WWW)

Oesch, P., Bouwens, R., Illingworth, G., et al. 2015, The Astrophysical Journal, 808, 104

Oke, J., & Gunn, J. 1983, The Astrophysical Journal, 266, 713

Paszke, A., Gross, S., Chintala, S., et al. 2017

Peng, C. Y., Impey, C. D., Rix, H.-W., et al. 2006, The Astrophysical Journal, 649, 616

Petrillo, C. E., Tortora, C., Chatterjee, S., et al. 2017, ArXiv e-prints, arXiv:1702.07675

Postman, M., Coe, D., Benítez, N., et al. 2012, The Astrophysical Journal Supplement Series, 199, 25

Pourrahmani, M., Nayyeri, H., & Cooray, A. 2018, The Astrophysical Journal, 856, 68

Pourrahmani, M., Nayyeri, H., Cooray, A., Payumo, K., & Zhaoyu, W. forthcoming in 2020, The Astrophysical Journal

Radford, A., Metz, L., & Chintala, S. 2015, arXiv e-prints, arXiv:1511.06434

Refsdal, S., & Bondi, H. 1964, Monthly Notices of the Royal Astronomical Society, 128, 295

Rodney, S. A., Strolger, L.-G., Kelly, P. L., et al. 2016, apj, 820, 50

Scoville, N., Abraham, R., Aussel, H., et al. 2007, The Astrophysical Journal Supplement Series, 172, 38

Sonnenfeld, A., Chan, J. H. H., Shu, Y., et al. 2018, Publications of the Astronomical Society of Japan, 70, S29

Spilker, J. S., Marrone, D. P., Aravena, M., et al. 2016, apj, 826, 112

Suyu, S., Auger, M., Hilbert, S., et al. 2013, The Astrophysical Journal, 766, 70

Suyu, S., Treu, T., Hilbert, S., et al. 2014, The Astrophysical Journal Letters, 788, L35

Tegmark, M., Strauss, M. A., Blanton, M. R., et al. 2004, Physical Review D, 69, 103501

Timmons, N., Cooray, A., Riechers, D. A., et al. 2016, apj, 829, 21

Treu, T. 2010, Annual Review of Astronomy and Astrophysics, 48, 87

Treu, T., & Marshall, P. J. 2016, aapr, 24, 11

Treu, T., Schmidt, K., Brammer, G., et al. 2015, The Astrophysical Journal, 812, 114

Velander, M., van Uitert, E., Hoekstra, H., et al. 2014, Monthly Notices of the Royal Astronomical Society, 437, 2111

Wardlow, J. L., Cooray, A., De Bernardis, F., et al. 2012, The Astrophysical Journal, 762, 59

Weinberg, D. H., Mortonson, M. J., Eisenstein, D. J., et al. 2013, Physics Reports, 530, 87

Wilson, D., Cooray, A., Nayyeri, H., et al. 2017, arXiv preprint arXiv:1705.00734