Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **473,** 3895–3906 (2018)
Advance Access publication 2017 July 7

doi:10.1093/mnras/stx1665

CMU DeepLens: deep learning for automatic image-based galaxy–galaxy strong lens finding

François Lanusse,[1]★ Quanbin Ma,[2] Nan Li,[3,4] Thomas E. Collett,[5] Chun-Liang Li,[2] Siamak Ravanbakhsh,[2] Rachel Mandelbaum[1] and Barnabás Póczos[2]

[1]*McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[2]*School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[3]*High Energy Physics Division, Argonne National Laboratory, Lemont, IL 60439, USA*
[4]*Department of Astronomy & Astrophysics, The University of Chicago, 5640 South Ellis Avenue, Chicago, IL 60637, USA*
[5]*Institute of Cosmology and Gravitation, University of Portsmouth, Burnaby Rd, Portsmouth, PO1 3FX, UK*

Accepted 2017 June 29. Received 2017 June 6; in original form 2017 March 7

ABSTRACT

Galaxy-scale strong gravitational lensing can not only provide a valuable probe of the dark matter distribution of massive galaxies, but also provide valuable cosmological constraints, either by studying the population of strong lenses or by measuring time delays in lensed quasars. Due to the rarity of galaxy-scale strongly lensed systems, fast and reliable automated lens finding methods will be essential in the era of large surveys such as Large Synoptic Survey Telescope, *Euclid* and *Wide-Field Infrared Survey Telescope*. To tackle this challenge, we introduce `CMU DeepLens`, a new fully automated galaxy–galaxy lens finding method based on deep learning. This supervised machine learning approach does not require any tuning after the training step which only requires realistic image simulations of strongly lensed systems. We train and validate our model on a set of 20 000 LSST-like mock observations including a range of lensed systems of various sizes and signal-to-noise ratios (S/N). We find on our simulated data set that for a rejection rate of non-lenses of 99 per cent, a completeness of 90 per cent can be achieved for lenses with Einstein radii larger than 1.4 arcsec and S/N larger than 20 on individual g-band LSST exposures. Finally, we emphasize the importance of realistically complex simulations for training such machine learning methods by demonstrating that the performance of models of significantly different complexities cannot be distinguished on simpler simulations. We make our code publicly available at https://github.com/McWilliamsCenter/CMUDeepLens.

Key words: gravitational lensing: strong – methods: statistical.

1 INTRODUCTION

Strong gravitational lensing, the serendipitous appearance of multiple images or extended arcs due to distant quasars or galaxies almost directly behind a massive object along the same line of sight, finds many important applications both astrophysical and cosmological. It is in particular a well-established probe of overall gravitational potential of massive galaxies. For example, moderate-sized samples of galaxy-scale strong lenses from the Sloan Lens ACS (SLACS) survey (Bolton et al. 2006) have been used to derive ensemble constraints on the total matter profile in massive elliptical galaxies (e.g. Koopmans et al. 2006; Auger et al. 2010; Barnabè et al. 2011). As a cosmological probe, time delays in multiply imaged strongly lensed quasars can be used to derive independent constraints on the Hubble constant H_0 (e.g. Refsdal 1964; Suyu et al. 2010; Bonvin et al. 2017), but even without time delays strongly lensed systems can constrain cosmological parameters and in particular the dark energy equation of state (e.g. Collett & Auger 2014; Cao et al. 2015). See e.g. Treu (2010) for a review of results using strong lensing by galaxies.

Upcoming large sky surveys such as the Large Synoptic Survey Telescope (LSST;[1] LSST Science Collaboration et al. 2009), *Euclid*[2] (Laureijs et al. 2011), and the *Wide-Field Infrared Survey Telescope* (*WFIRST*;[3] Spergel et al. 2015) will produce data sets that include unprecedented numbers of galaxy-scale strong lenses. For example, Collett (2015) predicts $>10^5$ galaxy-scale strong lens systems in LSST and *Euclid*. This lens population is expected to be dominated by intermediate-redshift ($z \sim 0.5$–1) elliptical

★ E-mail: francois.lanusse@gmail.com

[1] https://www.lsst.org/lsst/
[2] http://sci.esa.int/euclid/, http://www.euclid-ec.org
[3] https://wfirst.gsfc.nasa.gov/

© 2017 The Authors
Published by Oxford University Press on behalf of the Royal Astronomical Society

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

galaxies, with blue source galaxies. Clearly, the challenge will be to detect those galaxy-scale strong lenses efficiently and in a way that results in an easily quantifiable selection function for further scientific investigation. LSST and *Euclid* present different challenges for automated galaxy-scale strong lens detection. The LSST imaging will have six photometric bands, enabling us to distinguish between objects with different colours, but at low resolution compared to space-based imaging. In contrast, the *Euclid* imaging will be far higher resolution but with only a single optical passband. Thus, neither survey achieves the optimal setup for galaxy-scale strong lens detection, i.e. high-resolution multicolour imaging. The space-based, multiband near-infrared data from *WFIRST* in principle provides a data set that is closer to optimal, though the galaxy populations probed may differ due to the fact that the imaging is at near-infrared wavelengths.

Most automated image-based lens finding methods proposed so far have been based on the detection of arc and ring structures in the images. The `ARCFINDER` method (Alard 2006; More et al. 2012) relies for instance on a local elongation estimator, at the pixel level. Other approaches to arc finding purely based on morphology include those described by Seidel & Bartelmann (2007); Kubo & Dell'Antonio (2008); Bom et al. (2017) as well as Avestruz et al. (2017) which uses histograms of oriented gradients to detect edge patterns in the image. When considering specifically galaxy-scale strong lenses, arcs can often be obscured by the light of the lens galaxy. Several methods have been proposed to perform a first subtraction step of the lens galaxy to facilitate the detection of faint arcs. The `RingFinder` algorithm proposed in Gavazzi et al. (2014) uses a multiband differencing scheme to reveal faint blue arcs surrounding early-type galaxies. Joseph et al. (2014) proposed an alternative subtraction scheme, which does not require multiband images, based on a Principal Component Analysis (PCA) decomposition of the lens light profile. A different approach based on physical lens modelling was initially proposed in Marshall et al. (2009) for high-resolution space-based images and revisited in Brault & Gavazzi (2015) for ground-based images. In this class of methods, a simple model including a background source and a foreground deflector is fitted on each lens candidate, and a classification as a possible lens is made based on the predicted model parameters.

As an example of automated lens searches, some of these methods were applied to the Strong Lensing Legacy Survey (SL2S; Cabanac et al. 2007), a survey from the Canada–France–Hawaii Telescope Legacy Survey (CFHTLS) covering an approximate area of 150 deg^2. The `ARCFINDER` was applied to the survey in More et al. (2012), producing a final sample of 127 potential lenses, mostly on group and cluster mass scales. The authors report that the algorithm produced an average of 1000 candidates per deg^2 requiring a significant amount of further visual inspection. The `RingFinder` algorithm was also applied to the CFHTLS data (Gavazzi et al. 2014) but concentrated on galaxy-scale lenses by pre-selecting early-type galaxies. This algorithm produced on average a more manageable 18 candidates per deg^2 and a final sample of 330 high-probability lenses after visual inspection. In this particular case, the authors reported that visual inspection required an estimated 30 person-minutes per deg^2. At this rate, next generation surveys such as *Euclid* ($\sim$15 000 deg^2) or LSST ($\sim$20 000 deg^2) would still require a very significant investment of human time.

One approach to scale the visual inspection effort to the size of these surveys is to use crowdsourcing. This is the idea behind the `Space Warps` project (Marshall et al. 2015; More et al. 2015), which crowdsourced the visual inspection of a sample of 430 000 images from the CHFTLS to a crowd of 37 000 citizen scientists, yielding a new sample of gravitational lens candidates. 59 of these candidates were previously missed by robotic searches, while 20 known lenses were missed by the volunteers. These results show a complementarity between the two approaches, and crowdsourced visual inspection could prove very valuable in screening lens candidates found by automated searches. The authors further estimate that a similar crowdsourcing effort can be scaled up to LSST sizes, where a considerable crowd of 10^6 volunteers could visually inspect 10^6 LSST targets in a matter of weeks.

The image classification problem involved in strong lens finding is a notoriously challenging task that has received considerable attention in the broader field of computer vision and machine learning. Very recently, a new class of models based on the *deep learning* framework (LeCun, Bengio & Hinton 2015) has been able to surpass human accuracy in similar image classification tasks (He et al. 2015a). Such models are therefore extremely promising for gravitational lens searches as they could prove more reliable than non-expert human inspection and therefore dramatically reduce the amount of human time investment for future surveys. Some deep learning models have already been proposed in an astrophysical context, most notably for automatic identification of galaxy morphologies (Dieleman, Willett & Dambre 2015), performing star–galaxy classification (Kim & Brunner 2017), estimating photometric redshifts (Hoyle 2016), or for generative models of galaxy images (Ravanbakhsh et al. 2017).

The idea of using deep learning for detecting strongly lensed systems has recently been explored by several groups. For instance, Petrillo et al. (2017) recently published an application on the Kilo Degree Survey (KiDS; de Jong et al. 2015). This method, as well as all other deep learning methods currently under investigation for this task, is based on deep *convolutional neural networks* (CNNs; Lecun et al. 1998; Krizhevsky, Sutskever & Hinton 2012), a powerful architecture for image detection and classification tasks.

In this paper, we present a new approach to strong lens finding, based on deep *residual networks* (He et al. 2015a), an advanced variation of CNNs that constitutes the current state of the art in image classification. As a result, our proposed architecture, named `CMU DeepLens`, or `DeepLens` in short, has recently been found to outperform most CNN-based lens finders in a recent blind challenge organized by the *Euclid* strong lensing working group[4] (Metcalf et al., in preparation).

Contrary to most previous methods, such a supervised machine learning approach does not make any prior assumptions on specific features or physical models and instead lets the machine learn from the provided training data whose features are the most relevant to the detection of strong lenses. In addition to characterizing the performance of our baseline architecture using simulations of lens systems in LSST, we also explore the impact of model and simulation complexity on machine learning-based lens finders.

This paper is structured as follows. In Section 2, we provide some background on the deep learning framework and introduce the building blocks used in our classifier. In Section 3, we detail the `DeepLens` architecture itself as well as the training procedure. We then describe in Section 4 the strong lensing simulation pipeline that was used to generate a set of training and validation images. These simulations are used in Section 5 to quantify the performance of our model. Finally, Section 6 discusses current limitations as well as several avenues for further improvement of our results, with conclusions provided in Section 7.

[4] http://metcalf1.bo.astro.it/blf-portal/gg_challenge.html

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

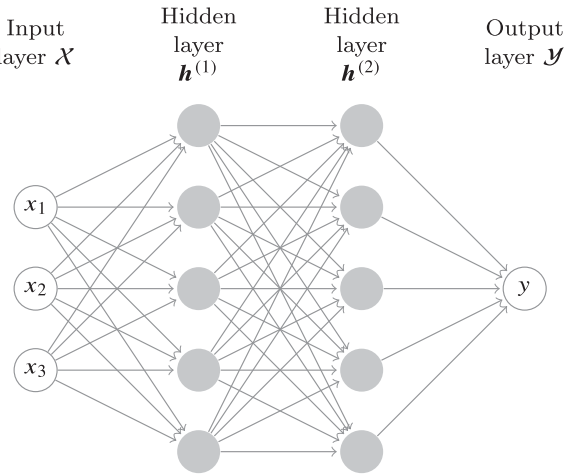


Figure 1. Conventional multilayer perceptron, a feedforward neural network with fully connected hidden layers. Each arrow represents a directed weighted connection.

2 DEEP LEARNING BACKGROUND

In the first section, we provide a brief overview of the deep learning framework and introduce the specific components used in our model.

2.1 The deep learning revolution

Artificial neural networks (ANNs) have been an established tool for classification and regression tasks for several decades. In fully connected feedforward models, such as the Multilayer Perceptron (MLP), each layer is composed of elementary units (the neurons) performing a simple weighted linear combination over the outputs of all units in the previous layer, followed by the application of a non-linear transform, also known as the *activation function*. This architecture is illustrated in Fig. 1. Let $1 \leq \ell \leq L$ denote the neural layer in a deep architecture and let N_ℓ denote the number of outputs of layer $\boldsymbol{h}^{(\ell)}$. Using tensor notations, the output of a given layer $\boldsymbol{h}^{(\ell)} \in \Re^{N_\ell}$ of the network can be expressed in terms of the output of the previous layer $\boldsymbol{h}^{(\ell-1)}$ according to

$$\boldsymbol{h}^{(\ell)} = f(\boldsymbol{W}^{(\ell)} \cdot \boldsymbol{h}^{(\ell-1)} + \boldsymbol{b}^{(\ell)}), \quad (1)$$

where $f : \Re^{N_\ell} \rightarrow \Re^{N_\ell}$ is the element-wise activation function, such as the sigmoid-shaped logistic function $f(\boldsymbol{h}) = 1/(1 + \exp(-\boldsymbol{h}))$, $\boldsymbol{W}^{(\ell)} \in \Re^{N_\ell \times N_{\ell-1}}$ is a dense weight matrix and $\boldsymbol{b}^{(\ell)} \in \Re^{N_\ell}$ is a vector of additive biases applied before the activation function. Here, the input $\boldsymbol{x} = \boldsymbol{h}^{(0)}$ is identified by $\ell = 0$ and output $\boldsymbol{y} = \boldsymbol{h}^{(L)}$ is the final layer.

Such a model is trained to perform a specific task by optimizing its parameters $\{\boldsymbol{W}, \boldsymbol{b}\}$ as to minimize a given *loss function*. This optimization is performed by a Stochastic Gradient Descent (SGD) algorithm which iteratively updates the model weights by taking small gradient steps computed over a randomly selected sub-sample of the training set. The computation of these gradients is made tractable in practice using the *backpropagation* algorithm (Rumelhart, Hinton & Williams 1986), which simply applies the chain rule to efficiently obtain the derivative of the the loss function with respect to the model parameters. This relies on the idea that the gradients at layer ℓ can efficiently be computed by using the gradients at layer $\ell + 1$. A crucial point of this training procedure is that the gradients of the model are computed starting from the last hidden layer $\ell = L$ and then propagated through the network back to the first layer $\ell = 1$.

If trained well, the performance of such a feedforward neural network is expected to increase with the depth of the model, as additional layers allow for a more complex mapping from the input to the output. Even using one hidden layer, a feedforward network – with sufficient number of hidden units – is known to be a universal approximator (Hornik, Stinchcombe & White 1989), that is it can approximate any function to arbitrary precision.

However, until very recently, deep networks with many hidden layers had remained completely unpractical as they were notoriously difficult to train. The main reason is the so-called *vanishing gradient* effect where the gradient of the cost function decreases exponentially due to the repeated chain rules and eventually vanishes as it is backpropagated through the layers during training. As a result, parameters in lower-level layers could not be tuned well in practice, thus greatly limiting the depth of typical models to a few hidden layers.

While this limitation had remained unsolved for several decades, the deep learning revolution was brought about by a conjunction of factors: the emergence of effective procedures to train deep architectures, the explosion of the volume of available training data and the increase in computing power through Graphics Processing Units (GPUs). Amongst the innovations that allowed for deeper models, the ReLU (for rectified linear unit) activation function was introduced in Nair & Hinton (2010). This simple function, defined as $f(x) = \max(x, 0)$, does not saturate contrary to conventional sigmoid functions, leading to much stronger gradients which can be propagated deeper in the models during training. The availability of much larger training sets as well as new regularization techniques (e.g. dropout regularization; Hinton et al. 2012; Srivastava et al. 2014) have further made it possible to train large networks without overfitting. Finally, efficient implementations of neural network architectures on GPUs have accelerated the training process by several orders of magnitudes, compared to CPU implementations. Combining these factors and innovations with the pre-existing CNN architecture (Lecun et al. 1998, see the next section for details) deep neural networks have suddenly been able to reach significant depth and achieve state-of-the-art results in image classification problems (Krizhevsky et al. 2012).

Deep architectures now achieve state-of-the-art (and sometimes superhuman) performance in a wide range of applications in computer vision, natural language processing, bioinformatics, data mining and computer games. We refer interested readers to Goodfellow, Bengio & Courville (2016) for a general introduction.

2.2 Convolutional neural networks

Taking into account the specific topological structure and properties of natural images, Lecun et al. (1998) introduced the CNN as an efficient alternative to fully connected architectures for image processing. The building block of CNNs is the *convolutional layer* which outputs a set of *feature maps* by convolving an input image with learned local filters. This process is illustrated in Fig. 2, where an input RGB image is convolved with a set of eight 3×3 filters to yield eight output feature maps.

More formally, let us consider $\boldsymbol{h}^{(\ell-1)}$ the input of convolution layer ℓ. Considering square images for simplicity, we will denote the height and width in pixels of $\boldsymbol{h}^{(\ell-1)}$ by $N_{\ell-1}$, while its *depth* (i.e. the number of bands or feature maps) will be denoted by $K_{\ell-1}$, so that $\boldsymbol{h}^{(\ell-1)} \in \Re^{K_{\ell-1} \times N_{\ell-1} \times N_{\ell-1}}$. In a similar fashion, we will note that $\boldsymbol{h}^{(\ell)} \in \Re^{K_\ell \times N_\ell \times N_\ell}$ the output of convolution layer ℓ. In the case

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

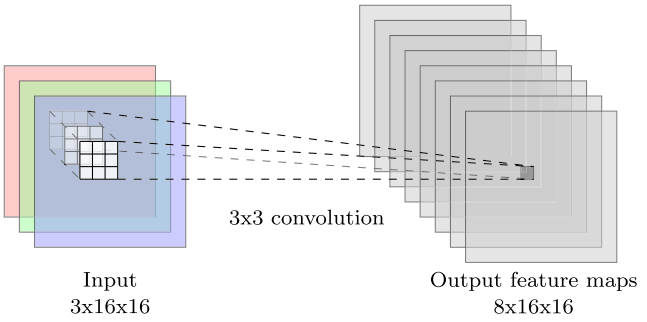


Figure 2. Illustration of one convolutional layer, applying 3×3 convolution kernels on an RGB image to produce a set of eight feature maps. Only the convolution by the first kernel is illustrated for clarity; each feature map is obtained by convolving the input image with a different convolution kernel.

of the illustration in Fig. 2, $N_{\ell-1} = N_\ell = 16$ while $K_{\ell-1} = 3$ for the input RGB image and $K_\ell = 8$ for the output feature maps. With these notations, a single output feature map of $\boldsymbol{h}^{(\ell)}$ for $1 \leq k \leq K_\ell$ is expressed as

$$\boldsymbol{h}_k^{(\ell)} = f\left(\sum_{k'} \boldsymbol{W}_{k',k}^{(\ell)} * \boldsymbol{h}_{k'}^{(\ell-1)} + b_k^{(\ell)}\right), \quad (2)$$

where $\boldsymbol{W}^{(\ell)} \in \Re^{K_{\ell-1} \times K_\ell \times I_\ell \times I_\ell}$ is referred to as the *convolution kernel* and it contains a different filter of size $I_\ell \times I_\ell$ for each combination of input and output channels. As in the fully-connected architecture, $b_k^{(\ell)} \in \Re$ is a bias parameter (i.e. one scalar parameter per output channel) and f is a non-linearity.

The size of the convolution kernel is generally limited to a few pixels (e.g. $I_\ell = 3$). To capture larger scale information, CNNs follow a hierarchical multiscale approach by interleaving convolution layers with so-called *pooling layers*, which apply a downsampling operation to the feature maps. A CNN architecture is a stack of convolution layers and pooling layers, converting the input image into an increasing number of feature maps of progressively coarser resolution. The final feature maps $\boldsymbol{h}^{(L)}$ can capture information on the scale of the input image and can reach a high level of abstraction. Combined with efficient GPU implementations, CNNs have become a standard model for image detection and classification tasks.

2.3 Deep residual networks

As mentioned in the previous section, the performance of a deep network generally increases with the number of layers, up to the point at which the model becomes too difficult to train and performances start to degrade. This general rule still applies to CNN architectures, which despite reaching greater depths than previous models have been limited to around a dozen layers. To overcome this well-known difficulty, machine learning research has recently been focused on developing alternative architectures beyond simple CNNs, in an attempt to build deeper models that can still be efficiently trained.

Several very recent developments have led to significant improvements in model accuracy for classification and detection tasks. For instance, the *Inception* architecture (Szegedy et al. 2015) allowed the GoogLeNet model to reach a depth of 22 layers and as a result to win the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC2014), one of the benchmarks in image classification and object detection. Even more recently, He et al. (2015a) introduced the *deep residual learning* framework, which allowed the authors

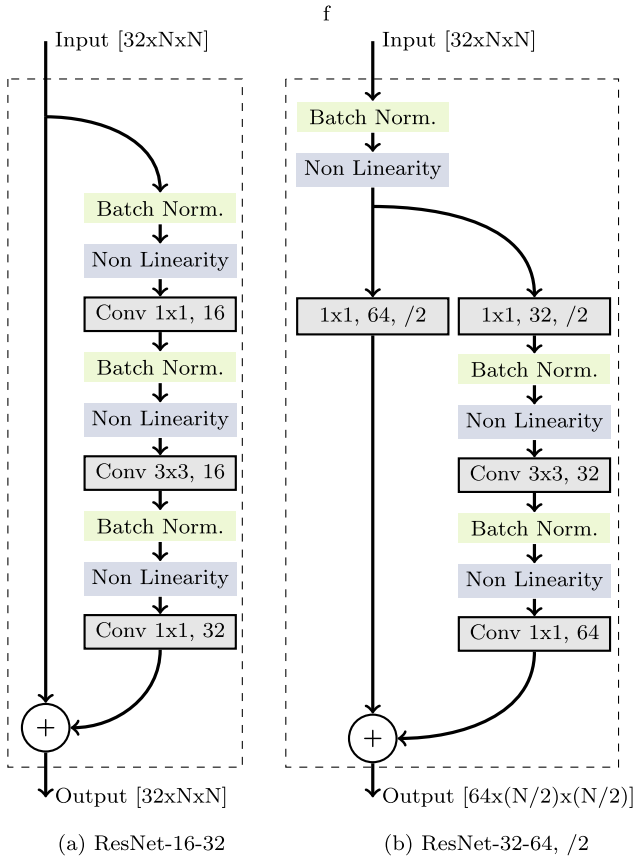


Figure 3. Pre-activated bottleneck residual units, the building blocks of a residual network architecture. (a) Undecimated ResNet-16-32 unit, preserving the size and depth of the input. (b) ResNet-32-64,/2 unit simultaneously increasing the depth of the output (from 32 channels to 64) and downsampling its resolution by a factor of 2.

to increase the depth of their convolutional model by one order of magnitude to 152 layers while still improving detection and classification accuracy and ultimately winning the ILSVRC2015. These results were even further improved in a follow-up work (He et al. 2016) where the authors successfully trained models as deep as 1000 layers while still improving classification accuracy. The model proposed in our work is directly based on this state-of-the-art deep residual network, or *resnet*, architecture.

Residual learning aims to tackle the problem of vanishing gradients and difficult optimization of deep networks by introducing so-called *shortcut* connections between the input and output of a stack of a few convolution layers, so that instead of learning how to map the input to the output, the convolution layers are only learning their difference, hence the term residual learning. In other words, instead of learning directly a given non-linear mapping $\mathcal{H}(\boldsymbol{x})$ between input and outputs, residual networks are trained to learn the residual mapping with respect to the identity: $\mathcal{F}(\boldsymbol{x}) = \mathcal{H}(\boldsymbol{x}) - \boldsymbol{x}$. An illustration of a simple residual unit proposed in He et al. (2016) is provided in Fig. 3(a), where the left branch corresponds to the shortcut connection and the right branch is learning the residual mapping with a few convolution layers. Deep residual networks are then built by stacking a large number of these residual units.

While this residual learning architecture may seem like a trivial recasting of the mappings in the network, it has been found in practice to be much easier to train in deep networks (He et al. 2015a), for several reasons. First and foremost, this architecture nearly

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

eliminates the vanishing gradient problem by providing an unhindered route for the gradients to backpropagate through the layers using the shortcut connection. Another major advantage is that weight initialization can be made more robust, given that the weights in a residual connection should be close to zero. Finally, the residual mappings are expected to be simpler than the full mappings, leading to an easier overall optimization problem.

In this work, we will adopt a specific type of units advocated by He et al. (2016), the *pre-activated bottleneck residual unit*. Pre-activation refers to the inversion of the conventional ordering of convolution and activation functions. In a classical CNN architecture, the activation function is applied to the output of the convolution, as described in equation (2), but in a pre-activated architecture, the element-wise activation function is first applied to the input image, followed by the convolution. This alternative architecture is empirically found to yield better performances. These units are built around a stack of 1×1, 3×3 and 1×1 convolution layers (hence the 'bottleneck' appellation). This is also empirically found to yield similar performance as a stack of two 3×3 convolutions but for a reduced number of parameters. Fig. 3 illustrates two variants of these residual units; Fig. 3(a) preserves the dimensions and depth of the input, while Fig. 3(b) allows for a downsampling in resolution and an increase in depth by inserting a convolution layer in the shortcut branch. This downsampling of a factor 2 is performed by using a strided convolution instead of using a pooling layer as in a conventional CNN. Finally, these blocks use *batch normalization* layers, which, as their name implies, standardize their outputs by removing a mean value and dividing by a standard deviation estimated during training.

3 CMU DEEPLENS

Having provided some general background on deep learning in the previous section, we now present the details of our proposed `CMU DeepLens` strong-lens finder. In particular, we describe its architecture, training procedure and implementation.

3.1 Architecture

Our `DeepLens` model is a direct adaptation of the residual network architecture proposed in He et al. (2015a, 2016). The model described here corresponds to our fiducial choice of depth and complexity, which we found was suitable for the complexity of our task and the size of training data.

The architecture of our model is illustrated in Fig. 4. The input image is first processed by a convolution layer, which can accommodate single- or multiband images, before going through a stack of pre-activated bottleneck residual units (illustrated in Fig. 3), arranged in several levels of progressively coarser resolution. In total, our model is 46 layers deep.

The output of the residual network is finally processed through a single fully connected layer with a sigmoid activation function. Apart from the last layer, we use the exponential linear unit (ELU) activation (Clevert, Unterthiner & Hochreiter 2015) throughout, which slightly differs from the ReLU activation:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ e^x - 1, & \text{otherwise} \end{cases} . \tag{3}$$

If $y \in \{0, 1\}$ is the class of an input image $\boldsymbol{x}$, the output of our model represents the estimated probability $\hat{y} = q(y = 1 \mid \boldsymbol{x}) \in [0, 1]$ of the input image $\boldsymbol{x}$ being a strongly lensed system (i.e. belonging to the class $y = 1$), where $\boldsymbol{\theta}$ are the parameters of the model (the weights of the neural network). A detection will correspond to $\hat{y}$ crossing a given detection threshold, which will balance the trade-off between false and true detections as will be discussed in Section 5.

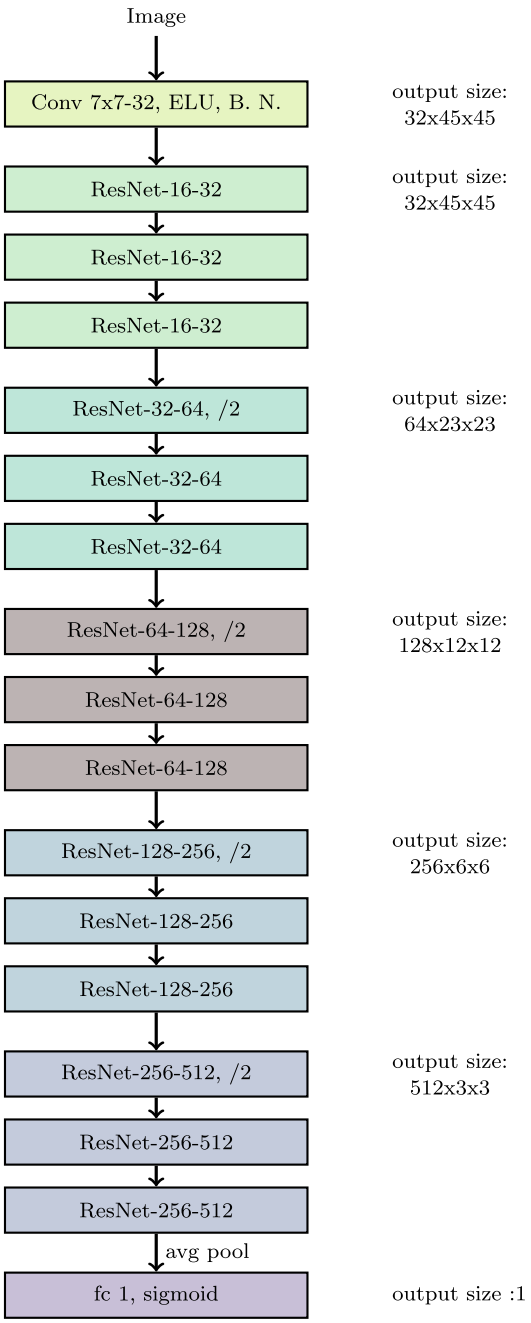


Figure 4. Architecture of `CMU DeepLens`. The first block is a single convolution layer with an ELU activation function and batch normalization, the following residual units are illustrated in Fig. 2. The last block is a single fully connected layer with a sigmoid (logistic function) activation function, which outputs a probability between 0 and 1.

3.2 Classification cost function

For the binary classification problem of strong lens detection, we use the *binary cross-entropy* cost function.

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

Let $y[n] \in \{0, 1\}$ denote the label of each instance $\boldsymbol{x}[n]$ and let $\hat{y}[n] \in [0, 1]$ be the probability $q(y[n] = 1 \mid \boldsymbol{x}[n])$ estimated from the model. The likelihood of the binary classification can be written as $\prod_n \hat{y}[n]^{y[n]} \times (1 - \hat{y}[n])^{1-y[n]}$ which results in the following negative log-likelihood:

$$-\sum_{n=1}^{N} y[n] \log \hat{y}[n] + (1 - y[n]) \log(1 - \hat{y}[n]), \tag{4}$$

where N is the number of training instances. This loss function can also be interpreted as the cross-entropy between $p(y \mid \boldsymbol{x})$ and our model $q_\theta(y \mid \boldsymbol{x})$. The network is then trained to minimize this cross-entropy (negative log likelihood) objective.

A common concern in such a classification problem is class imbalance, i.e. a situation where the ratio between the number of instances of the two classes can be different between the training and testing sets. When uncertain about the class of an example, the model will rely on the prior $p(y)$ learned from the training data to make a prediction. If the ratio of classes changes in the training set, the probabilities predicted by the model will be different, and if that ratio does not match that of the testing set, the model predictions will be sub-optimal.

However, it can be shown that the effect of class imbalance can be accounted for by an overall mapping of the probabilities predicted by the model (see Saerens, Latinne & Decaestecker 2002). Let us consider a first data set with a given class ratio i.e. a given prior $p(y)$ and a model $q_\theta(y|\boldsymbol{x})$ trained on this data set. Let us now consider a second data set statistically identical to the first one except for a different class ratio defined by a different prior probability $p'(y)$. The posterior probability $q_{\theta'}(y \mid \boldsymbol{x})$ predicted by a model directly trained on this second data set can be mapped to the output of the first model evaluated on that second data set according to

$$q_{\theta'}(y|\boldsymbol{x}) = \frac{\lambda\, q_\theta(y|\boldsymbol{x})}{\lambda\, q_\theta(y|\boldsymbol{x}) + \mu\, (1 - q_\theta(y|\boldsymbol{x}))}, \tag{5}$$

where $\lambda = \frac{p'(y)}{p(y)}$ and $\mu = \frac{1-p'(y)}{1-p(y)}$ are the ratios of the two classes between the two data sets. If the class ratios are known between the two data sets (λ and μ), then this expression can be used to apply a model trained on one data set to the other. Even if the ratios are not known, an important point to stress is that this mapping is monotonic, therefore the ordering of the probabilities predicted by the model is preserved in case of class imbalance. As will be discussed in Section 5, this last property allows us to define a procedure to set a detection threshold insensitive to class imbalance.

3.3 Pre-processing and data augmentation

We apply minimal pre-processing to the input images. We subtract the mean image from the training set and normalize the images by the noise standard deviation σ in each band evaluated over the whole data set. Finally, extreme values are clipped to restrict the dynamic range of the input images to within a given $k\sigma$ (we use here $k = 250$, but this value can be adjusted). Although these steps are usual and sensible pre-processing techniques, we do not find a significant impact on the results if they are omitted. If masked pixels are present in the image, they are set to 0 as a last stage of pre-processing (i.e. after mean subtraction and normalization).

More crucial to effectively train the model, we further perform several data augmentation steps, as a way to increase the effective size of the training set and to make the model invariant to specific transformations. In particular, we want our model to be rotationally invariant, for which we apply random rotations to the training images. The following steps are applied to the training data:

(i) Random rotation of the image in the range $[-90, 90^\circ]$, using a spline interpolation scheme.
(ii) Random mirroring along the vertical and horizontal axes.
(iii) Random zooming of the image in the range [0.9, 1], meaning that the image is randomly compressed (or stretched) by a factor within this range.

When these operations access pixels outside of the input image, a simple wrapping strategy is used and the augmented image remains the size of the original image.

3.4 Training procedure

We initialize the weights of our model with random normal values using the strategy proposed in He et al. (2015b) and all layers are trained from scratch. The network is trained over 120 epochs (i.e. passes over the whole training set) in minibatches of 128 images using ADAM (Kingma & Ba 2015) using the default exponential decay rates and a staring learning rate of $\alpha = 0.001$, which is divided by 10 every 40 epochs.

3.5 Implementation

Our model itself is implemented using the `Theano`[5] and `Lasagne`[6] libraries. We make our code publicly available at: https://github.com/McWilliamsCenter

The results presented in this work were obtained on an Nvidia Titan X (Pascal) GPU. As will be described in the next section, we trained our model on a set of 16 000 images of size 45×45 pixels. Despite the relatively small size of our training set, we find our data augmentation scheme to be very effective and we do not find any evidence of overfitting. On this data set and hardware, the training procedure requires approximately 1 h. Once the network is trained however, the classification itself is extremely efficient, requiring approximately 350 µs per image.

4 STRONG LENSING SIMULATIONS

In this section, we detail the simulation pipeline that was used to produce training and testing sets for our lens finding model. We simulate LSST images of strong lenses using PICS (Pipeline for Images of Cosmological Strong lensing; Li et al. 2016) for the strong lensing ray-tracing and LensPop[7] (Collett 2015) for mock LSST observing.

All simulated images contain a central early-type galaxy as well as some additional galaxies in the field of view. In half of the simulated images, we further include a strongly lensed background galaxy.

4.1 Lens galaxy model

For these simulations, we use a population of elliptical lens galaxies, as they are expected to dominate the population of galaxy-scale strong lenses (Oguri & Marshall 2010). To model the mass profile of these elliptical galaxies, we assume a singular isothermal ellipsoid

[5] http://deeplearning.net/software/theano/
[6] https://github.com/Lasagne/Lasagne
[7] https://github.com/tcollett/LensPop

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

(SIE) profile. This model is not only analytically tractable, but also found to be consistent with models of individual lenses and lens statistics on the length scales relevant for strong lensing (e.g. Koopmans et al. 2006; Gavazzi et al. 2007; Dye et al. 2008). The SIE density profile is defined as

$$\rho(x, y) = \frac{\sigma_v^2}{2\pi G\left(x^2/q + qy^2\right)}, \tag{6}$$

where σ_v is the velocity dispersion of the lens and q is the axis ratio of the ellipsoid. For such a profile, the convergence of the lens is given by

$$\kappa(x, y) = \frac{\theta_E}{2}\frac{1}{\sqrt{x^2/q + y^2 q}}, \tag{7}$$

where θ_E is the Einstein radius, which can be calculated according to the redshift of the lens z_l, the redshift of the source z_s, and the velocity dispersion of the lens galaxy as

$$\theta_E = 4\pi\left(\frac{\sigma_v}{c}\right)^2 \frac{D_{ls}(z_l, z_s)}{D_s(z_s)}, \tag{8}$$

where c is the speed of light, D_{ls} and D_s are the angular diameter distance from the source plane to the lens plane and from the source plane to the observer, respectively.

Given these relations, a lens is entirely described by only a few parameters: $\{\sigma_v, q, z_l, z_s, \phi\}$, where ϕ is a rotation angle. We uniformly sample the velocity dispersion, ellipticity and orientation of the lenses from a range of typical values, so that $\sigma_v \in [150, 350\,\mathrm{km\,s^{-1}}]$, $q \in [0.5, 1.0]$ and $\phi = [0, 2\pi]$. The lens redshifts are obtained by matching the velocity dispersion in our simulations to the catalogue of elliptical galaxies in the COSMOS survey from Zahid et al. (2015). The resulting redshift range of our lenses is $z_l \in [0.2, 0.7]$. The source galaxies will be assumed to be at a fixed redshift of $z_s = 2.0$.

To model the light distribution of the lens galaxies, we use an elliptical Sérsic profile, defined as

$$I(R) = I_{\mathrm{eff}} \exp\left\{-b_n\left[\left(\frac{R}{R_{\mathrm{eff}}}\right)^{1/n} - 1\right]\right\}, \tag{9}$$

where $R = \sqrt{x^2/q + y^2 q}$, R_{eff} is the effective radius in units of arcsecond, I_{eff} is the intensity at the effective radius, n is the index of the Sérsic profile. For each lens galaxy, we use for the light profile the same axis ratio and orientation as the SIE mass profile and we set the effective radius, luminosity and Sérsic index to the matched COSMOS galaxy.

4.2 Background sources and additional line-of-sight galaxies

We use for the lensed background sources a set of detailed images of low-redshift bright galaxies ($z \sim 0.45$) extracted from the mosaics produced by the CANDELS team (Grogin et al. 2011; Koekemoer et al. 2011) and selected from the CANDELS UDS catalogue Galametz et al. (2013). These galaxies are rescaled and placed at a fixed redshift $z_s = 2.0$, near the caustics of the lensing system so as to produce lensed arcs. The lensed images themselves are then produced by ray-tracing simulations as part of the PICS pipeline.

To add to the complexity of the generated image and make them look more realistic, we further populate the field with galaxies drawn from the *Hubble Ultra Deep Field*. These galaxies are placed along the line of sight but in these simulations, their images are not lensed even if they are located behind the lens.

Table 1. Parameters of the mock LSST observations. Only the median values are reported for the seeing and sky brightness.

Band	Seeing (arcsec)	Sky	Exposure time (s)	Pixel scale (arcsec)
g	0.81	21.7	3000	0.18

Note that our simulated fields contain a single deflector on a single lens plane, we do not include any additional perturbative weak lensing nor any compound lensing effects (e.g. Collett & Bacon 2016).

4.3 Mock LSST observations

To produce LSST-like images from the ideal images produced in the previous steps, we use the LensPop software (Collett 2015) to perform mock observing. Images are re-sampled to match the detector pixel scale and convolved with a circularly symmetric Gaussian point spread function discretised at the same pixel scale.

A noisy realization of this image is generated assuming a Poisson model based on the sky plus signal and an additional Gaussian read-out noise. Parameters for these simulations follow Collett (2015) and are based on the LSST observation simulator (Connolly et al. 2010). They are summarized in Table 1.

To account for seeing and sky-brightness variations over the course of the survey, each simulated exposure is drawn from a stochastic distribution of these parameters. We then consider two different strategies to use these exposures. For each field, we build one *single-epoch* image by keeping only the best seeing exposure and another 'worst-case' *stack* image by co-adding all individual exposures after degrading them all in resolution to match the one with the worst seeing. These two sets of images will allow us to investigate the trade-off between signal to noise and resolution for our automated lens search.

Examples of final mock observations with these two strategies are shown in Fig. 5 for different arc sizes and signal-to-noise ratios (S/N). This ratio is defined in our final images in terms of the total flux of the lensed arc:

$$\mathrm{S/N} = \frac{F_{\mathrm{tot}}}{\sigma\sqrt{N_{\mathrm{pix}}}}, \tag{10}$$

where F_{tot} is the total flux of the lensed source, σ is the rms noise in the observed image and N_{pix} is the number of pixels associated with the lensed source (measured by segmentation on a noiseless image of the arc using a threshold of $0.5\,\sigma$, where σ is the rms noise in the corresponding full image).

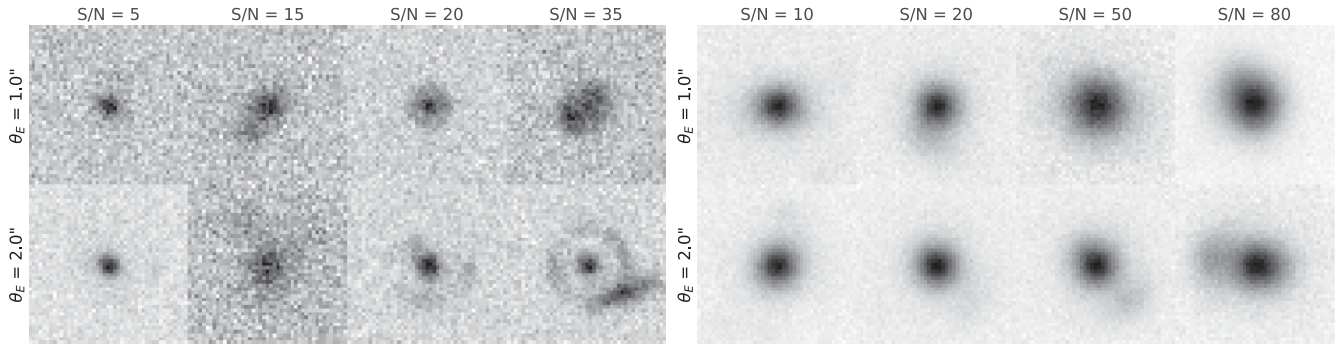
Finally, these simulations were also used to test the lens finding pipeline proposed in Avestruz et al. (2017) and are made publicly available[8] with that paper in the interest of reproducible research.

5 RESULTS

In this section, we test the performance of our lens finder on the simulations described in the previous section. Following Joseph et al. (2014), we perform these tests as a function of Einstein radius θ_E and S/N of the lensed image. Note that this definition of S/N implicitly encapsulates the impact of source angular size and brightness as well as lens magnification into a single meaningful parameter. For each class of simulations (single-epoch and stack), we train a

[8] http://portal.nersc.gov/project/hacc/nanli/lsst_sl_mocks

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018



(a) Best single epoch (b) Stack images

Figure 5. Randomly selected simulated lens images of various Einstein radius $\theta_{\rm E}$ and arc S/N levels, shown in linear grey-scale. Single-epoch images (left) are produced by using only the best seeing exposure while the stack images are produced by co-adding all exposures down to the worst seeing in the stack.

model on a random subset of 16 000 images, following the procedure described in Section 3.4. We then evaluate the performance of the classifier on a test set of 4000 images as a function of various cuts in Einstein radius and S/N.

To quantify the performance of our lens finder, we measure the true positive rate (TPR) and false positive rate (FPR). The TPR, also known as *completeness* or recall, is defined as the ratio of detected lenses to the total number of lenses:

$$\mathrm{TPR} = \frac{N_{\text{true positives}}}{N_{\text{true positives}} + N_{\text{false negatives}}} \,. \tag{11}$$

The FPR, which can also be interpreted as a *contamination* rate is defined as the fraction of non-lens images wrongly identified as lenses:

$$\mathrm{FPR} = \frac{N_{\text{false positives}}}{N_{\text{false positives}} + N_{\text{true negatives}}} \,. \tag{12}$$

This statistic gives us a handle on the expected contamination of a lens sample while, like the TPR, being independent of the ratio of lens to non-lens images in the testing set, which is not representative of a real survey in our simulations.

These statistics are a function of the detection threshold applied to the probability of an image containing a lens outputted by the model. This threshold balances completeness versus contamination of the lens sample. This trade-off is typically illustrated by the receiver operating characteristic (ROC) curve which is obtained by plotting the TPR as a function of the FPR while varying this detection threshold from 0 to 1. By definition, the ROC curve is also insensitive to the ratio of lenses to non-lenses in the testing set. Fig. 6 shows the ROC curves of our classifier evaluated on the testing set, for various cuts in S/N and Einstein radius in the lens sample. We use a fiducial FPR value of 1 per cent to derive a detection threshold for the rest of this analysis. This FPR value, illustrated by a vertical dashed line in Fig. 6, would be set in practice based on what would be considered to be an admissible level of contamination in a given survey, the associated detection threshold would be derived from simulations.

This strategy to set the detection threshold makes our results completely insensitive to the ratio of lenses to non-lenses in training and testing sets, which therefore do not need to reflect the class ratio in observational data. Indeed, as mentioned in Section 3.2, the effect of class imbalance in the training set is an overall monotonic mapping of the probabilities predicted by the model, a transformation that does not affect the ROC curve. Despite our model being trained on an extremely unbalanced data set compared to observational data, its ROC curve is the same as if it were trained on a balanced data set. As the detection threshold is defined in terms of the FPR, the testing set used to derive this threshold does not need to be in-line with the observational ratio either.

For our fiducial detection level, we compare the corresponding completeness achieved by `DeepLens` for samples of increasingly larger and brighter arcs. These different completeness levels are marked by horizontal dashed lines in Fig. 6.

We find that our lens finder exhibits very similar behaviour for the two sets of images. The increase in S/N seems to roughly compensate the loss in resolution. We note that in single exposure images, we achieve a completeness of 90 per cent for our fiducial choice of 1 per cent FPR, when considering arcs larger than 1.4 arcsec. We reach similar completeness levels in stacked images for arcs of that size with S/N larger than 80.

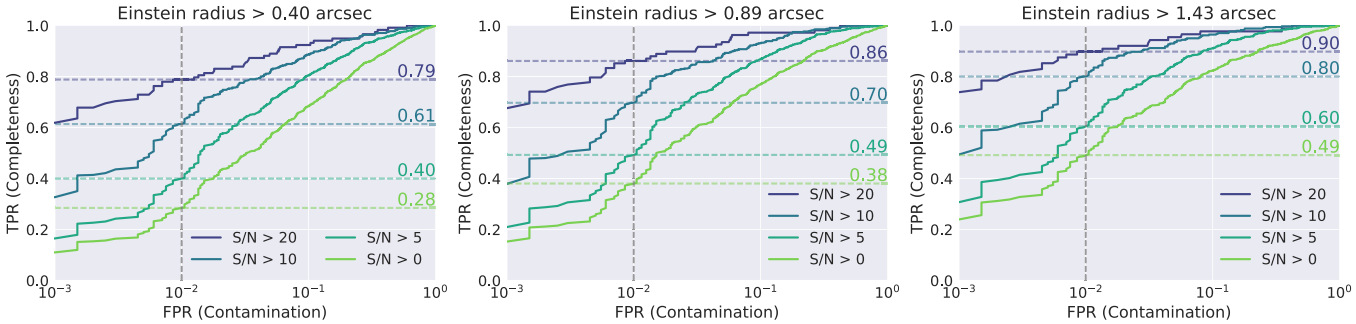
To further illustrate the impact of S/N and resolution on the recovery of lenses, we show in Fig. 7 the distributions of S/N and Einstein radius of our simulated lens population as well as the distribution of these properties in the recovered sample for our fiducial 1 per cent FPR threshold. Interestingly, we find that in stacked images, `DeepLens` can still recover some poorly resolved lenses with Einstein radius lower than the median seeing. To visually investigate some failure modes of the model, we show in Fig. 8 some examples of true and false positives for our two sets of simulations. We retain on this figure only the images with the highest predicted lens probability. As can be seen, the lenses for which `DeepLens` is the most certain have clearly visible rings and multiple images. False positives are dominated by the presence of multiple objects at the vicinity of the lens. We expect these types of failures to be dramatically reduced in multiband images, as the colour would provide the necessary additional information to discriminate real lenses from these false positives.

6 DISCUSSION

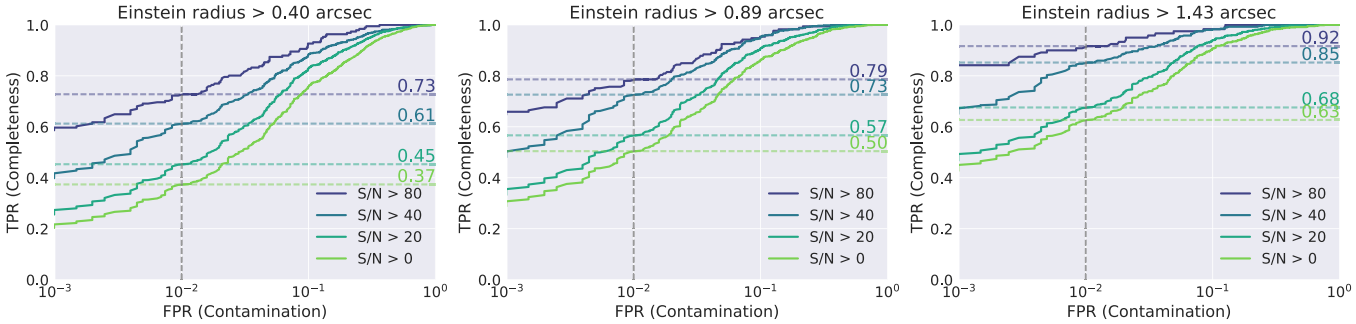
In this section, we discuss the importance of simulations in a supervised machine learning method and provide several avenues to further improve our proposed model.

As in any supervised machine learning approach, the quality of the training set is a major factor in the performance of the method on actual data. For instance, Petrillo et al. (2017) find with a conventional CNN approach that most of the false positives that the method produces on real KiDS images come from contaminants

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

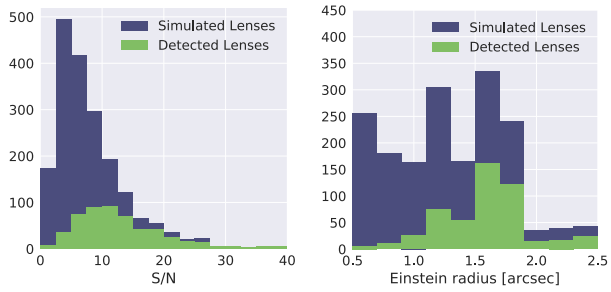


(a) Single best epoch images

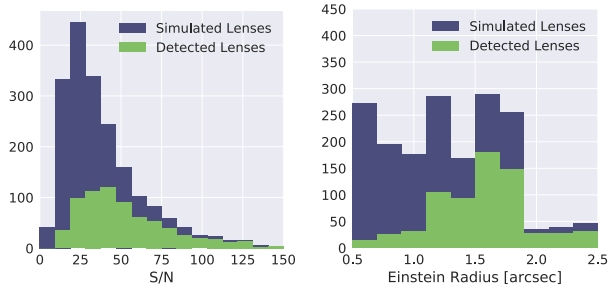


(b) Stack images

Figure 6. ROC curves for different cuts in Einstein radius and S/N of the input lensed image. The dashed vertical lines correspond to our fiducial 1 per cent contamination threshold while the horizontal dashed lines indicate the corresponding completeness for that contamination threshold. As we exclude from the input sample fainter and smaller lenses, the completeness progressively increases.



(a) Single best epoch images



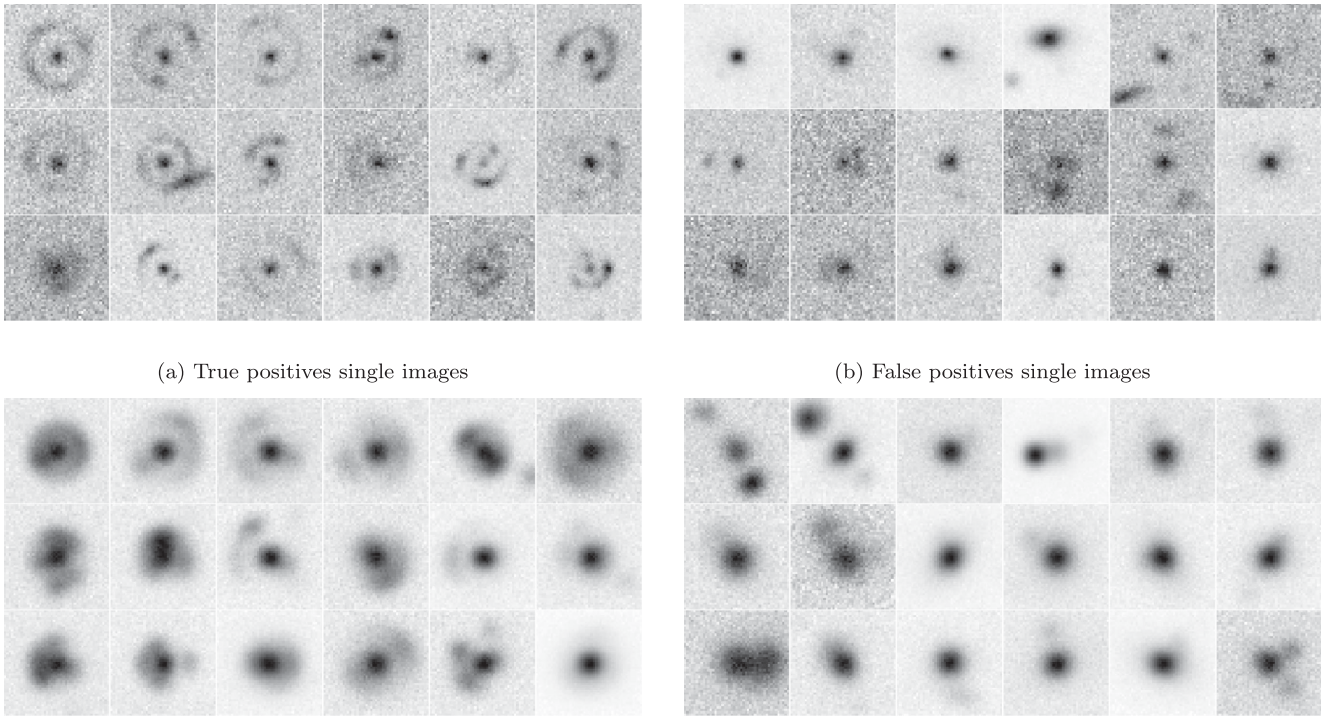
(b) Stack images

Figure 7. S/N and Einstein radius distribution of the simulated and recovered lens populations, for our fiducial 1 per cent FPR detection threshold.

such as ring galaxies, mergers or star-forming rings. For practical applications, it is paramount to train deep learning models on representative data sets including all the diversity and variability of real survey images. We should therefore stress that the reported completeness and contamination levels on our simulations are only optimistic estimates of the performance of our model on real data.

Realistic simulations are important not only for the final application to real data but also for method development. In particular, we find that our simulations are not complex enough to discriminate between our proposed `DeepLens` model and a classical CNN model such as the one proposed in Petrillo et al. (2017), despite the greater complexity of our model (46 versus 7 layers). As an illustration, we show in Fig. 9 the ROC curve obtained by training and then evaluating a CNN following closely the description provided in Petrillo et al. (2017) on our set of simulations, along with the ROC curve obtained with our `DeepLens` model on the same training and testing set (single best epoch images). As can be seen, we find the two models to exhibit exactly the same performance when evaluated on our set of simulations, despite our model having outperformed most CNN-based methods in the *Euclid* challenge (Metcalf et al., in preparation) and residual networks being known to significantly outperform CNNs in more complex image classification tasks (He et al. 2015a). This result shows that a simpler CNN is already complex enough to capture all the variability present in our non-trivial yet simple simulations, and no significant gains are made by increasing the complexity of the model. More complex simulations including more variability and non-trivial contaminants would reveal the limits of simpler models, whereas the additional complexity of a deeper model would facilitate a better interpretation of complex galaxy images to find robust features for lens detection. This result also illustrates the important point that when using supervised machine learning approaches, most of the burden is shifted from the development of the method itself to the production of realistic simulations for training purposes.

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018



(a) True positives single images

(b) False positives single images

(c) True positives stack images

(d) False positives stack images

Figure 8. Illustrations of true and false positives with the highest predicted lens probability in our two sets of simulations, shown in linear grey-scale.

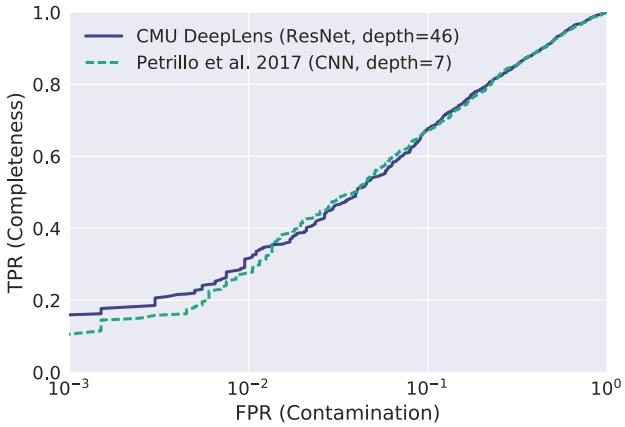


Figure 9. Comparison of ROC curves between our proposed model and the CNN proposed in Petrillo et al. (2017), evaluated on the same training and testing sets of single epoch images. Note that these curves are computed on our full lens sample, without S/N and Einstein radius cuts.

It should also be noted that we limited the complexity of our fiducial model to 46 layers but it can easily be made much deeper, and similar architectures have been successfully trained up to a thousand layers (He et al. 2016). There is still room for substantially more model complexity to handle more realistic simulations and data.

In the simulations presented in this work, we have only used single band images, which demonstrates our model's ability to identify lenses from their morphologies alone. This is a valuable aspect of the method, especially for the *Euclid* survey, where lens searches will be conducted on the VIS single band images. However, colour information is also very helpful to identify strongly lensed systems and is already at the heart of some methods (e.g. Gavazzi et al. 2014). As mentioned in the description of our architecture, our model can seamlessly handle multiband images and we expect its performance to be significantly improved by the addition of colour information.

Finally, we have only considered in this work generic galaxy–galaxy lenses. However, a number of particularly interesting science cases require exotic lens systems such as double source plane lenses (e.g. Collett & Auger 2014), lensing catastrophes (Orban De Xivry & Marshall 2009) or lensed supernovae (e.g. Rodney et al. 2016). However, these systems are much rarer and their specific detection is all the more challenging. In order to make our lens finder specifically sensitive to these rare cases, instead of a binary lens/non-lens classification, our method can be extended to a multiclass classification problem. A weighted cost function can also be used to further promote the purity of exotic lens candidate samples.

7 CONCLUSIONS

In this work, we presented `CMU DeepLens`, a new strong gravitational lens finder based on the most recent advances in deep learning. Our fully automated method does not require any manual tuning or human intervention beyond the training step. This new class of machine learning techniques represents an exciting prospect for conducting large-scale lens searches, as they have been shown to surpass human abilities in similar image classification tasks. They have the potential to significantly cut down on the need for human visual inspection and make the search for strong lenses tractable in the era of deep yet wide-field surveys such as LSST, *Euclid* and *WFIRST*. Being entirely automated, the selection function of such methods is also easier to quantify (given realistic simulations), which is a crucial point for precision cosmology (e.g. Collett & Cunnington 2016).

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

We demonstrated on simple yet non-trivial strong lensing simulations of LSST *g*-band observations that our algorithm is extremely efficient, rejecting a vast majority of non-lenses while preserving a high level of completeness. As a point of reference, we find that on single LSST exposures, for a fiducial rejection rate of 99 per cent, we still reach a completeness of 90 per cent for lenses with Einstein radii larger than 1.43 arcsec and S/N of the lensed image larger than 20. We also investigated the trade-off between better S/N at the cost of lower resolution by applying our lens finder to co-added image stacks, degraded to the worst seeing in the stack. We find very similar performance in this case compared to single best seeing exposures. As a result, we expect an optimal co-adding strategy to further improve on our results.

However, we note that these results are optimistic as our admittedly simple simulations do not include likely contaminants such as merging systems, spiral or ring galaxies. Our quoted completeness levels are therefore an optimistic estimate of the performance of our model on real data. We further demonstrate that our simulations are too simple to discriminate between deep learning models of vastly different complexities, meaning that a conventional CNN model exhibits the same performance as our more advanced and far deeper residual network model. On more realistic simulations however, such as the ones used for the strong lens finding challenge (Metcalf et al., in preparation), our residual network architecture was found to benefit from this added complexity and to outperform more conventional CNNs. This result illustrates the need for realistic simulations when developing and applying supervised machine learning methods, thus shifting part of the effort from the model development to the production of realistic simulations.

Finally, in the spirit of reproducible research, we make our code publicly available at https://github.com/McWilliamsCenter/CMUDeepLens as well as the strong lensing simulations used for training and testing at http://portal.nersc.gov/project/hacc/nanli/lsst_sl_mocks.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Michelle Ntampaka, Hy Trac, Phil Marshall, Frederic Courbin, Remy Joseph, Sukhdeep Singh, Noble Kennamer and Siyu He for useful remarks and discussions. We would also like to acknowledge the Euclid Strong Gravitational Lensing Challenge and the Bologna Lens Factory project for fostering the development of automated strong lens finders. This research was supported in part by the U.S. Department of Energy under grant DESC0011114 and by the National Science Foundation under grant IIS1563887.

REFERENCES

Alard C., 2006, preprint (arXiv:astro-ph/0606757)
Auger M. W., Treu T., Bolton A. S., Gavazzi R., Koopmans L. V. E., Marshall P. J., Moustakas L. A., Burles S., 2010, ApJ, 724, 511
Avestruz C., Li N., Lightman M., Collett T. E., Luo W., 2017, preprint (arXiv:1704.02322)
Barnabè M., Czoske O., Koopmans L. V. E., Treu T., Bolton A. S., 2011, MNRAS, 415, 2215
Bolton A. S., Burles S., Koopmans L. V. E., Treu T., Moustakas L. A., 2006, ApJ, 638, 703
Bom C. R., Makler M., Albuquerque M. P., Brandt C. H., 2017, A&A, 597, A135
Bonvin V. et al., 2017, MNRAS, 465, 4914
Brault F., Gavazzi R., 2015, A&A, 577, A85
Cabanac R. A. et al., 2007, A&A, 461, 813
Cao S., Biesiada M., Gavazzi R., Piórkowska A., Zhu Z.-H., 2015, ApJ, 806, 185
Clevert D.-A., Unterthiner T., Hochreiter S., 2015, preprint (arXiv:1511.07289)
Collett T. E., 2015, ApJ, 811, 20
Collett T. E., Auger M. W., 2014, MNRAS, 443, 969
Collett T. E., Bacon D. J., 2016, MNRAS, 456, 2210
Collett T. E., Cunnington S. D., 2016, MNRAS, 462, 3255
Connolly A. J. et al., 2010, Proc. SPIE Conf. Ser. Vol. 4477, Modeling, Systems Engineering, and Project Management for Astronomy IV. SPIE, Bellingham, p. 77381O–77381O–10.
de Jong J. T. A. et al., 2015, A&A, 582, A62
Dieleman S., Willett K. W., Dambre J., 2015, MNRAS, 450, 1441
Dye S., Evans N. W., Belokurov V., Warren S. J., Hewett P., 2008, MNRAS, 388, 384
Galametz A. et al., 2013, ApJS, 206, 10
Gavazzi R., Treu T., Rhodes J. D., Koopmans L. V. E., Bolton A. S., Burles S., Massey R. J., Moustakas L. A., 2007, ApJ, 667, 176
Gavazzi R., Marshall P. J., Treu T., Sonnenfeld A., 2014, ApJ, 785, 144
Goodfellow I., Bengio Y., Courville A., 2016, Deep learning. MIT Press, Cambridge, MA
Grogin N. A. et al., 2011, ApJS, 197, 3535
He K., Zhang X., Ren S., Sun J., 2015a, preprint (arXiv:1512.03385)
He K., Zhang X., Ren S., Sun J., 2015b, preprint (arXiv:1502.01852)
He K., Zhang X., Ren S., Sun J., 2016, in Leibe B., Matas J., Sebe N., Welling M., eds, Identity Mappings in Deep Residual Networks. Computer Vision ECCV 2016. Lecture Notes in Computer Science, Vol. 9908. Springer, Cham, p. 630
Hinton G. E., Srivastava N., Krizhevsky A., Sutskever I., Salakhutdinov R. R., 2012, preprint, (arXiv:1207.0580)
Hornik K., Stinchcombe M., White H., 1989, Neural Netw., 2, 359
Hoyle B., 2016, Astron. Comput., 16, 34
Joseph R. et al., 2014, A&A, 566, A63
Kim E. J., Brunner R. J., 2017, MNRAS, 464, 4463
Kingma D. P., Ba J. L., 2015, International Conference on Learning Representations 2015, San Diego, CA
Koekemoer A. M. et al., 2011, ApJS, 197
Koopmans L. V. E., Treu T., Bolton A. S., Burles S., Moustakas L. A., 2006, ApJ, 649, 599
Krizhevsky A., Sutskever I., Hinton G. E., 2012, in Pereira F., Burges C. J. C., Bottou L., Weinberger K. Q., eds, Advances In Neural Information Processing Systems. Curran Associates, Inc., Morehouse Lane, NY, p. 1097
Kubo J. M., Dell'Antonio I. P., 2008, MNRAS, 385, 918
Laureijs R. et al., 2011, preprint (arXiv:1110.3193)
Lecun Y., Bottou L., Bengio Y., Haffner P., 1998, Proc. IEEE, 86, 2278
LeCun Y., Bengio Y., Hinton G., 2015, Nature, 521, 436
Li N., Gladders M. D., Rangel E. M., Florian M. K., Bleem L. E., Heitmann K., Habib S., Fasel P., 2016, ApJ, 828, 54
LSST Science Collaboration et al., 2009, preprint (arXiv:0912.0201)
Marshall P. J., Hogg D. W., Moustakas L. A., Fassnacht C. D., Bradač M., Schrabback T., Blandford R. D., 2009, ApJ, 694, 924
Marshall P. J. et al., 2015, MNRAS, 455, 1171
More A., Cabanac R., More S., Alard C., Limousin M., Kneib J.-P., Gavazzi R., Motta V., 2012, ApJ, 749, 38
More A. et al., 2015, MNRAS, 455, 1191
Nair V., Hinton G. E., 2010, , in Furnkranz J., Joachims T., eds, Proceedings of the 27th International Conference on Machine Learning, Rectified Linear Units Improve Restricted Boltzmann Machines, p. 807
Oguri M., Marshall P. J., 2010, MNRAS, 405, 2579
Orban De Xivry G., Marshall P., 2009, MNRAS, 399, 2
Petrillo C. E. et al., 2017, MNRAS, preprint (arXiv:1702.07675)
Ravanbakhsh S., Lanusse F., Mandelbaum R., Schneider J., Poczos B., 2017, In AAAI Conference on Artificial Intelligence. Available at: https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14765
Refsdal S., 1964, MNRAS, 128, 307

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018

Rodney S. A. et al., 2016, ApJ, 820, 50
Rumelhart D. E., Hinton G. E., Williams R. J., 1986, Nature, 323, 533
Saerens M., Latinne P., Decaestecker C., 2002, Neural Comput., 14, 21
Seidel G., Bartelmann M., 2007, A&A, 472, 341
Spergel D. et al., 2015, preprint (arXiv:1503.03757)
Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R., 2014, J. Mach. Learn. Res., 15, 1929
Suyu S. H., Marshall P. J., Auger M. W., Hilbert S., Blandford R. D., Koopmans L. V. E., Fassnacht C. D., Treu T., 2010, ApJ, 711, 201
Szegedy C. et al., 2015, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Press, Boston, MA
Treu T., 2010, ARA&A, 48, 87
Zahid H. J., Damjanov I., Geller M. J., Chilingarian I., 2015, ApJ, 806, 122

This paper has been typeset from a TEX/LATEX file prepared by the author.

Downloaded from https://academic.oup.com/mnras/article-abstract/473/3/3895/3930852
by University of Portsmouth Library user
on 29 January 2018