May 19, 2025
Typeset using LaTeX **twocolumn** style in AASTeX7

Benchmarking Convolutional Neural Network on LSST-Like Strong-Lensing Simulations

Natália Raymundi Pinheiro [1]

[1] *State University of New York at Stony Brook*

ABSTRACT

Add abstract here.

Keywords: convolutional neural networks — strong gravitational lensing — image processing — data analysis

1. INTRODUCTION

2. DEEP LEARNING FRAMEWORK

Convolutional neural networks (CNNs) are good for image classification because they automatically build up a hierarchy of visual filters. In the earliest layers, small convolutional kernels learn to detect simple edges, textures, and gradients; in deeper layers, progressively larger receptive fields respond to more complex motifs such as arcs, rings, or even full Einstein-ring structures. Weight sharing across spatial locations keeps the total number of parameters manageable, and pooling layers introduce a good degree of translational invariance (both important when searching for strong-lensing features that can appear anywhere within an LSST cutout).

`PyTorch` provides an easy-to-understand API for constructing these architectures and handling the training loop. Its `Dataset`/`DataLoader` objects decouples data I/O from model logic: raw simulations are read once, split into train, validation, and test sets, and then are shuffled, and batched. This pipeline allows me to experiment rapidly with image transforms (rotations, flips, crops), batch sizes, and epoch counts without rewriting the core model code.

The ability to use networks pretrained on large image repositories (e.g. `ImageNet`) is an effective way to start learning when the custom dataset is relatively small or highly specialized. I adopted a ResNet-50 backbone with frozen weights up through the last convolutional block, replacing only the final fully connected layer with a two-class head (lens vs. non-lens). By fine-tuning this last layer, the model adapts visual filters to the nuances of strong-lensing morphologies, like faint arcs partially obscured by a luminous elliptical galaxy.

In the past, CNNs have been applied to strong-lensing detection. Early efforts used shallow architectures on hand-cropped cutouts, while more recent approaches (e.g. `LensFlow` (M. Pourrahmani et al. 2018), `DeepLens` (F. Lanusse et al. 2018)) integrate multi-scale filters or residual connections to boost sensitivity to faint arcs. Here, I build upon the `SLSim` simulation toolkit, combining raw image simulations with a flexible `PyTorch` `Dataset`/`DataLoader` pipeline to shuffle and batch hundreds of examples on the fly. This lets me experiment with architectural variants (like swapping out the loss criterion and optimization method, number of epochs and batch sizes, and more) without rewriting the data-handling code.

3. METHODOLOGY

3.1. *Lens Simulation*

I generated realistic LSST-like strong-lens images in four steps: drawing a population of deflector galaxies, drawing a background source population, lens-ray tracing to form lensed images, and building a matching set of non-lenses.

3.1.1. *Deflector Population*

First, I defined a cosmology and survey footprint:

```
1 import numpy
```

Email: natalia.raymundipinheiro@stonybrook.edu

3.2. *Data Preprocessing and Normalization*
3.3. *Architecture of* `LensNet`
3.4. *Training Procedure*
4. VALIDATION PERFORMANCE
5. DISCUSSION
6. FUTURE WORK
7. CONCLUSION
8. ACKNOWLEDGEMENTS

REFERENCES

Lanusse, F., Ma, Q., Li, N., et al. 2018, Monthly Notices of the Royal Astronomical Society, 473, 3895

Pourrahmani, M., Nayyeri, H., & Cooray, A. 2018, The Astrophysical Journal, 856, 68, doi: 10.3847/1538-4357/aaae6a