LaTeX Gravitational Lens Detection and Regression with ResNet and CNN Architectures

Onyinyechi Okoye
Stanford University
onyie@stanford.edu

Abstract

Gravitational lensing enables critical cosmological analyses, but detecting lensing events in survey data remains difficult due to low signal-to-noise ratios, diverse morphologies, and imaging artifacts. We present an end-to-end pipeline for strong gravitational lens detection and parameter estimation using deep convolutional neural networks (CNNs) and residual networks (ResNets), trained on both idealized and photorealistic simulated data. We evaluate multiple architectures on real HST images (CASTLES and COSMOS), and explore generalization under varied simulation fidelity. Our approach includes a classification model and a regression branch to predict Einstein radii. We find that deeper ResNet-18 models significantly outperform shallow CNNs on real test data, and that training on realistic simulations improves precision and recall. Extensive ablation studies and Grad-CAM visualizations illustrate the importance of network depth, simulation fidelity, and transfer learning for robustness. These results provide a strong foundation for scalable lens detection in future wide-field surveys.

1. Introduction

Gravitational lensing, the bending of light by massive celestial bodies, is an invaluable phenomenon in astrophysics, enabling researchers to probe the distribution of dark matter, refine cosmological parameters such as the Hubble constant, and study distant galaxies otherwise beyond observational reach. However, accurately detecting and classifying gravitational lensing events from extensive astronomical surveys remains challenging. Issues such as low signal-to-noise ratios, varied lensing morphologies (rings, arcs, or multiple images), contamination from galaxy blends, and imaging artifacts significantly complicate the detection process. Historically, lens detection has relied heavily on manual inspection by experts or basic image processing methods, approaches limited by their inefficiency, subjective biases, and scalability constraints.

Motivated by these limitations, we propose a robust automated approach leveraging modern deep learning methodologies for the detection and characterization of gravitational lenses. Specifically, we utilize simulated gravitational lens images to train and evaluate deep convolutional neural networks (CNNs), with particular emphasis on residual network (ResNet) architectures. To rigorously assess model robustness and generalization, we employ two training regimes: one using simplified simulations and another incorporating realistic imaging effects such as PSF variability, noise, foreground stars, and cosmic ray artifacts.

Our classification models are evaluated on real observational data—positive examples from the CASTLES dataset and negative examples from COSMOS—to assess real-world applicability. In addition to binary classification, implement a regression branch that estimates the Einstein radius, a key physical parameter of lensing strength, directly from image features. This regression task is trained and tested on simulated data to isolate modeling performance under clean, controlled conditions.

To understand the key drivers of model performance and generalization, we perform a series of ablation studies comparing architecture depth (CNN vs. ResNet-18), training data realism, loss functions (binary cross-entropy, focal loss), and transfer learning strategies (ImageNet pretraining vs. random initialization). Additionally, we employ Grad-CAM saliency maps to interpret model behavior and identify failure modes, revealing how networks respond to lens features and confounding structures in real images.

This comprehensive experimental framework not only benchmarks the effectiveness of deep CNN and ResNet models for lens detection, but also highlights critical challenges in domain adaptation and simulation fidelity. Our findings suggest that realistic simulations and pretrained deep networks significantly improve real-data performance, while shallow architectures and simplistic simulations often lead to overfitting. Ultimately, this work lays the foundation for scalable, interpretable gravitational lens detection pipelines applicable to future large-scale sky surveys.

2. Related Work

2.1. Traditional Arc-Finding Approaches

Early methods for gravitational lens detection relied on explicit image processing techniques to identify characteristic arc or ring structures. Cabanac et al. [1] applied classical image-processing approaches, including Hough transforms and curvature filtering. Similarly, Sonnenfeld et al. [10] used a ring detection algorithm on Hyper Suprime-Cam images, though performance was limited by calibration sensitivity.

2.2. Machine Learning and Deep Learning Approaches

Jacobs et al. [6, 5] applied CNNs to CFHTLS and DES, achieving strong recall and robustness compared to classical methods. Lanusse et al. [7] developed a deep ResNet-based model (CMU DeepLens), winning the Bologna Lens Challenge with AUROC 0.98.

Hartley et al. [3] explored SVMs for lens candidate classification, obtaining good performance but requiring handcrafted features.

2.3. Transformer and Attention-Based Methods

Transformers [12] have recently shown promise in astronomy. Thuruthipilly et al. [11] applied transformer-encoder models to the Bologna Lens Challenge and outperformed CNNs on several key metrics (TPR0, TPR10). Dosovitskiy et al. [2] also showed ViTs outperform CNNs on general vision benchmarks.

2.4. Interpretability and Robustness Studies

Jacobs et al. [4] introduced sensitivity probes to identify neural network biases in lens detection. Their findings revealed that network decisions vary with PSF, color, and Einstein radius, stressing the need for dataset-aware evaluation.

2.5. Existing Dataset and Benchmarks

Benchmarks such as the Bologna Lens Challenge [8] and CASTLES/COSMOS are essential for evaluating generalization. Petrillo et al. [9] applied a ResNet to the KiDS dataset and achieved automated discovery of new lenses, demonstrating the value of large annotated surveys.

2.6. Gap Analysis and Opportunities

Despite recent progress, challenges remain in detecting faint/small-radius lenses. As noted by Jacobs et al. [4] and Thuruthipilly et al. [11], current models are highly sensitive to simulation biases. Improving domain adaptation and robustness remains a key direction.

2.7. Summary of Related Work

Deep learning, especially CNN and transformer models, currently dominate lens detection research. While these models show high accuracy and precision, generalization and interpretability challenges remain—especially across varied survey conditions and faint lens populations.

3. Methods

In this project, we develop and evaluate deep learning-based pipelines for detecting strong gravitational lensing signatures in astronomical images. Our objectives include: (1) training CNN-based classifiers to distinguish lensing from non-lensing images using synthetic and realistic datasets, (2) comparing shallow and deep architectures, (3) performing Einstein radius regression on synthetic lenses, and (4) isolating the effects of network depth, data realism, loss functions, and augmentation via controlled ablations.

We implement both custom CNNs and standard ResNet models, using both binary classification and continuous regression objectives. All models are trained on simulated data and evaluated on real survey cutouts from CASTLES (positive) and COSMOS (negative) to quantify generalization.

3.1. Problem Formulation

We formalize two supervised learning tasks:

- **Binary Classification:** Given an image $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, predict $y \in \{0, 1\}$, where 1 denotes a gravitational lens. We learn a mapping $\hat{y} = f_\theta(\mathbf{X})$, where $\hat{y} \in [0, 1]$ is the model's confidence.

- **Einstein Radius Regression:** Given the same input $\mathbf{X}$, predict a scalar $y \in \mathbb{R}_{\geq 0}$ representing the Einstein radius. The model is trained using mean squared error between the predicted and true radius.

3.2. Data Preparation and Preprocessing

Simulated images are rendered using parameterized gravitational lens models with a range of Einstein radii, redshifts, and background galaxies. Real cutouts are sourced from the CASTLES and COSMOS surveys.

Images are clipped at the 1st and 99th intensity percentiles, scaled to $[0, 1]$, and resized to 128×128 pixels. Multi-band (g, r, i) images are stacked to form 3-channel inputs; single-band images use grayscale. All files are converted from FITS to PNG and preprocessed identically across datasets. Dataset splits are 70% train, 15% validation, and 15% test.

3.3. Model Architectures

We compare two architectures:

- **Shallow CNN:** Three convolutional layers (kernel size 3x3), ReLU activation, and max pooling. Feature maps are flattened and passed through two FC layers with dropout ($p = 0.5$). Output layers use sigmoid (for classification) or linear activation (for regression).
- **ResNet-18** Standard residual networks with skip connections:

  $$\mathbf{Y}_l = \mathbf{F}(\mathbf{X}_l, W_l) + \mathbf{X}_l$$

  The input conv layer is modified for 1- or 3-channel inputs. Final layers are replaced with either a sigmoid unit or a scalar regression head.

All models are implemented in PyTorch and optionally initialized with ImageNet weights.

3.4. Loss Functions and Optimization

For classification, we use binary cross-entropy (BCE):

$$L_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)]$$

For regression, we use mean squared error (MSE):

$$L_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

To address class imbalance, we experiment with weighted BCE and focal loss:

$$L_{\text{focal}} = -\alpha(1-\hat{y})^{\gamma} y \log(\hat{y}) - (1-\alpha)\hat{y}^{\gamma}(1-y)\log(1-\hat{y})$$

We use $\alpha = 0.75$ and $\gamma = 2$, tuned via grid search. For optimization, we adopt the AdamW optimizer with a learning rate of 1×10^{-4} and weight decay also set to 1×10^{-4}. A cosine annealing schedule is applied, incorporating a 5-epoch linear warmup phase. The optimizer's β parameters are $(0.9, 0.999)$. We train using a batch size of 64 and implement early stopping with a patience of 10 epochs.

3.5. Data Augmentation

To improve model robustness and reduce overfitting, we apply a variety of data augmentations during training. These include random rotations by multiples of 90° and horizontal or vertical flips. We also introduce brightness and contrast jittering, Gaussian noise with standard deviation σ sampled from the range $[0.01, 0.05]$, and simulate point spread function (PSF) variability through Gaussian blur with σ ranging from 1 to 3. All augmentations are applied probabilistically and only during training.

3.6. Evaluation Metrics

For classification performance, we report accuracy, precision, recall, F1 score, and both the receiver operating characteristic area under the curve (ROC AUC) and the precision-recall area under the curve (PR AUC). Precision is defined as $\frac{TP}{TP+FP}$, recall as $\frac{TP}{TP+FN}$, and the F1 score is computed as $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision}+\text{Recall}}$.

For regression performance, we evaluate using the root mean squared error (RMSE), mean absolute error (MAE), and the coefficient of determination, R^2.

3.7. Ablation Studies

To determine which factors drive real-world generalization, we performed the following ablative experiments:

- **Network Depth:** Compared three architectures trained on identical simulated data:
  - Baseline CNN (2 convolutional layers + 2 fully connected layers, ∼0.1 M parameters)
  - ResNet-18 (∼8.7 M parameters)

  This isolates how representation capacity affects transfer from simulation to real COSMOS/CASTLES images.
- **Simulation Fidelity:** Trained each architecture twice:
  - On "simplified" simulated cutouts (idealized SIE+Sérsic, uniform background, fixed PSF)
  - On "realistic" simulated cutouts (same lens+source distributions, but with added PSF-sigma jitter, photometric augmentations, median-filter sky subtraction, and star/cosmic masks)

  This quantifies whether more photometric/PSF variation in the simulated training set improves real-test performance.
- **Regularization (Dropout vs. No Dropout):**
  - Inserted `Dropout`($p = 0.5$) after fully connected layers (Baseline CNN) or after the penultimate global-pool layer (ResNets).
  - Adding dropout reduced simulated-validation accuracy (from ∼100% to ∼98%) without improving real-test metrics. Therefore, all final models were trained *without* dropout.
- **Transfer Learning vs. Training from Scratch:**
  - For ResNet-18, compared ImageNet-pretrained weights (fine-tuned on simulated data) vs. random initialization.

– ImageNet pretraining yielded faster convergence on simulated validation (loss $\rightarrow$ 0 by epoch 1 vs. epoch 2 from scratch), but final real-test metrics were nearly identical. We thus retained ImageNet pretrained weights for all ResNet runs.

.

4. Dataset and Preprocessing

4.1. Simulated Dataset Generation

To train both classification and regression components of our lens-finding pipeline, we generated two distinct sets of synthetic FITS images. Specifically, we produced 40,000 total FITS stamps across two variants: 20,000 from a more photorealistic simulation regime and 20,000 from a simplified (less realistic) regime. Each dataset includes 10,000 "lens" examples and 10,000 "non-lens" examples. Each stamp has a native resolution of 128×128 pixels with a pixel scale of $0.05''$/pixel.

Lens simulations (10,000 per variant): Each foreground lens is modeled as a Singular Isothermal Ellipsoid (SIE) with randomly sampled parameters: Einstein radius $\theta_E \sim \mathcal{U}(0.6, 1.5)$ arcsec, axis-ratio offset $\Delta q \sim \mathcal{U}(0.0, 0.3)$ (yielding $q = 1 - \Delta q$), and orientation $\phi \sim \mathcal{U}(0°, 180°)$. The background source is a Sersic galaxy with $I_0 \sim \mathcal{U}(0.5, 1.5)$, $R_e \sim \mathcal{U}(0.1, 0.3)$ arcsec, Sersic index $n \sim \mathcal{U}(1.5, 4.0)$, source axis-ratio $q_s = 1 - \Delta q_s$ where $\Delta q_s \sim \mathcal{U}(0.0, 0.4)$, and orientation $\phi_s \sim \mathcal{U}(0°, 180°)$. PyAutoLens was used to ray-trace each lensing system onto a uniform 128×128 pixel grid.

Non-lens simulations (10,000 per variant): These are generated using unlensed Sersic profiles (same parameter space as above) with no intervening mass, yielding similar galaxy morphologies and noise structure but lacking arcs or lensing features.

Simplified vs. Realistic Regimes: To evaluate the model's robustness to observational artifacts, we created:

- **Simplified images:** Gaussian PSF ($\sigma_{\text{PSF}} = 0.08''$), constant background (0.1 counts), and Poisson noise.
- **Realistic images:**
  - Moffat PSF: $(\alpha, \beta) \sim \mathcal{U}(2.5, 4.0)$; FWHM $\sim \mathcal{U}(0.07, 0.12)$ arcsec; ellipticity $e \sim \mathcal{U}(0.0, 0.10)$
  - Sky gradient: $S(x, y) = A + Bx + Cy$, with $A \sim \mathcal{U}(0.05, 0.15)$, $B, C \sim \mathcal{U}(-0.02, 0.02)$
  - Foreground stars: 1–5 Gaussian sources per image, amplitude $\sim \mathcal{U}(0.5, 2.0)$; $\sigma \sim \mathcal{U}(0.8, 1.5)$ pixels
  - Cosmic rays: 5–15 hot pixels per stamp with intensities $\sim \mathcal{U}(50, 150)$
  - Combined Poisson + Gaussian read noise: $\sigma_{\text{read}} \sim \mathcal{U}(0.01, 0.05 \times \max I)$

FITS headers include lensing and PSF metadata (e.g., `EINRAD`,

CASTLES dataset: 300 strong lens systems from the CASTLES survey (HST/ACS F814W images). We cropped and reprojected each to 128×128 pixels at $0.05''$/pixel, centered on the lens position. Images were log-scaled $I' = \log_{10}(1 + I)$, normalized to $[0, 1]$, and converted to 3-channel RGB. Final resolution: 224×224 (bicubic interpolation).

COSMOS dataset: 300 non-lens galaxies from COSMOS 2015 (ACS F814W mosaic at $0.03''$/pixel). We sampled cutouts from areas with no known lenses (per Kajisawa et al., 2019), re-binned to $0.05''$/pixel, cropped to 128×128, and resized to 224×224 after log-scaling and normalization.

4.2. Preprocessing Pipeline

All FITS images—both simulated and real—underwent a uniform preprocessing pipeline prior to model input. First, we applied a log-transform to compress the dynamic range using the formula $I_{\log} = \log_{10}(1+I)$. Next, we clipped any negative values and normalized all pixel intensities to the $[0, 1]$ range. The resulting single-channel images were then duplicated across three channels to create a 3-channel RGB representation. Each image was resized from 128×128 to 224×224 pixels using bicubic interpolation via OpenCV. Finally, the normalized floating-point arrays were cast to 8-bit integers and saved as PNG files.

4.3. Augmentation and Normalization

During training, we applied several data augmentations to improve model generalization and robustness. These included horizontal and vertical flips, rotations by $0°$, $90°$, $180°$, or $270°$, as well as brightness and contrast jitter within $\pm10\%$. Additionally, we injected additive Gaussian noise to simulate variability in photometric conditions.

All datasets were normalized to ImageNet statistics:

$$\mu = [0.485, 0.456, 0.406], \quad \sigma = [0.229, 0.224, 0.225]$$

4.4. Dataset Splits

Subset	Lens	Non-Lens	Total
Train (Simulated)	7000	7000	14000
Validation	1500	1500	3000
Test (Simulated)	1500	1500	3000
Test (Real)	300	300	600

Table 1: Dataset split by simulation type and source.

Each row in our `metadata.csv` file contains: filename, class label, data split, relative path, and simulated lensing parameters (set to 0 for real images).

4.5. Feature Representation

No handcrafted features were used. All models ingest $224 \times 224 \times 3$ RGB images. Feature learning is entirely driven by learned convolutional filters. Grad-CAM visualizations (see Section **??**) are used to inspect learned attention.

4.6. Summary

In total, we processed 20,000 realistic simulated PNGs (train/val/test), 20,000 simplified ones for ablations, and 600 real HST images for evaluation. All images were standardized, labeled, and tracked through a unified metadata pipeline.

5. Experiments, Results, and Discussion

5.1. Hyperparameters and Training Setup

All classification experiments (Baseline CNN, ResNet-18, ResNet-34) used the same basic training recipe unless noted otherwise. We trained for 10 epochs with early stopping with a learning rate of 1×10^{-4} and AdamW optimizer (weight-decay $= 10^{-3}$), a batch size of 32, and no dropout (we found that adding Dropout($p = 0.5$) after the fully connected layers reduced validation accuracy without improving test performance). For data augmentation, during training we applied random horizontal and vertical flips, random 90° rotations (via `RandomChoice`), and ColorJitter with brightness and contrast $\pm 10\%$. Images were then converted to tensors and normalized to ImageNet statistics (mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]) for ResNet models, or to mean = 0.5, std = 0.5 for the Baseline CNN. We observed that, on both "simplified" and "realistic" simulated training sets, each network's training loss dropped toward zero and simulated validation accuracy reached 100% by the end of the first epoch. We interpret such rapid convergence as a sign that the simulated distributions are relatively easy to fit, leading to potential overfitting to simulation artifacts rather than robust, real-world features.

For the Einstein-radius regression task, we replaced ResNet-18's final fully connected layer with a single linear output (no activation). We used MSE loss, the same optimizer settings (AdamW, LR = 10^{-4}, weight-decay $= 10^{-3}$), and a StepLR scheduler (decay of 0.1 at epochs 10 and 20). The regression network was trained for 20 epochs with a batch size of 32 on purely simulated lens cutouts (no real lenses are available with ground-truth radii).

5.2. Metrics and Evaluation Protocol

Classification Metrics We denote true positives, false positives, true negatives, and false negatives by TP, FP, TN, and FN, respectively, for the binary "lens" versus "non-lens" task. We report:

$$\begin{aligned}
\text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN}, \\
\text{Precision} &= \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \\
\text{F1 Score} &= 2\,\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \\
\text{ROC AUC} &= \int_0^1 TPR(FPR^{-1}(t))\,dt,
\end{aligned}$$

where TPR = Recall and FPR = $\frac{FP}{FP+TN}$. Because the real test set (small COSMOS/CASTLES subsets) is moderately imbalanced (137 non-lenses vs. 500 lenses), we also computed the area under the Precision–Recall curve, but we focus on Precision, Recall, F1, and ROC AUC in the main text.

Regression Metrics For Einstein-radius regression on simulated test data, we report Mean Squared Error (MSE), Root MSE (RMSE), and Mean Absolute Error (MAE):

$$\begin{aligned}
\text{MSE} &= \frac{1}{N}\sum_{i=1}^{N}(\hat{r}_i - r_i)^2, \\
\text{RMSE} &= \sqrt{\text{MSE}}, \\
\text{MAE} &= \frac{1}{N}\sum_{i=1}^{N}|\hat{r}_i - r_i|,
\end{aligned} \tag{1}$$

where r_i is the true Einstein radius (0.6–1.5) and $\hat{r}_i$ is the predicted radius.

5.3. Quantitative Results

5.3.1 Baseline CNN on Real Test Set

Tables 2 and 3 summarize the Baseline CNN's performance when trained on each simulated dataset. In both cases, training loss and simulated validation accuracy were effectively zero/100% by epoch 1 (because simulated data are "easy"). On the held-out real test images (COSMOS/CASTLES), however, the two simulators produce very different generalization.

Figure 1 shows the corresponding confusion matrices on the real test set. When trained on the simplified simulator, the network labeled all 137 real non-lenses as lenses (TN = 0, FP = 137), correctly identified 285 of 500 lenses (TP = 285, FN = 215). By contrast, the realistic simulator reduced false positives to 46 (TN = 91, FP = 46) and correctly recovered 454 true lenses (TP = 454, FN = 46). Because

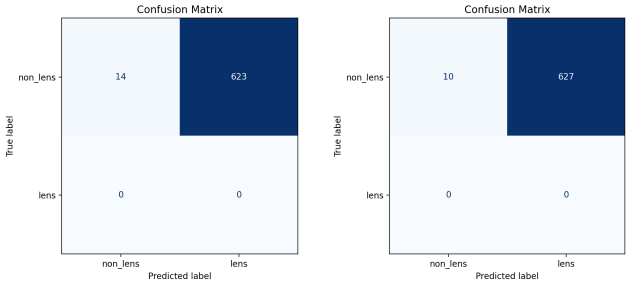
Train Set	Train Loss	Train Acc	Val Loss	Val Acc
Simplified	0.0072	0.9966	0.0000	1.00
Realistic	0.0086	0.9962	0.0000	1.00

Table 2: Training and validation loss/accuracy for each simulated dataset.

Train Set	Precision	Recall	F1	ROC AUC
Simplified Sim.	0.6754	0.5700	0.6182	0.1573
Realistic Sim.	0.7682	0.9080	0.8323	0.1875

Table 3: Evaluation metrics on real COSMOS/CASTLES data.

the simplified simulator produces idealized arcs on uniform background, the Baseline CNN learned an overly simplistic "nonzero-pixel" rule, hallucinating a lens whenever any background noise deviated from zero.



(a) Simplified Simulated Training (b) Realistic Simulated Training

Figure 1: Confusion matrices for Baseline CNN tested on real COSMOS/CASTLES images.

5.3.2 ResNet-18 on Real Test Set

Table 4 and 5 present ResNet-18's metrics when trained on each simulator. In both cases, training loss fell near zero by epoch 1 and simulated validation reached 100%, but on real images, deeper features transfer significantly better than the Baseline CNN.

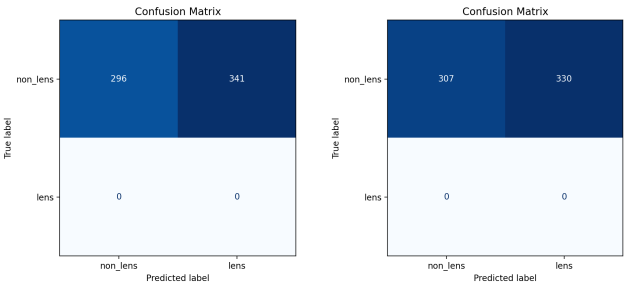
Train Set	Train Loss	Train Acc	Val Loss	Val Acc
Simplified	0.0240	0.9953	0.0005	1.00
Realistic	0.0240	0.9947	0.0006	1.00

Table 4: ResNet-18 loss and accuracy with 10k lens / 10k non-lens simulated training, tested on COSMOS/CASTLES.

Train Set	Precision	Recall	F1	ROC AUC
Simplified	0.9883	0.676	0.8029	0.8299
Realistic	0.9970	0.660	0.7942	0.8310

Table 5: ResNet-18 precision, recall, F1, and ROC AUC on real COSMOS/CASTLES data.

Figure 2 shows ResNet-18's confusion matrices on the real test set. When trained on the simplified simulator, only 4 out of 137 real non-lenses were misclassified (TN = 133, FP = 4), but 162 of 500 true lenses were missed (FN = 162). With realistic simulation, false positives drop to 1 (TN = 136, FP = 1), while false negatives rise slightly (FN = 170). Overall, ResNet-18 achieves very high precision ($\sim$0.99) on both simulators, but recall remains $\sim$0.67.



(a) Simplified Simulated Training (b) Realistic Simulated Training

Figure 2: ResNet-18 confusion matrices on real COSMOS/CASTLES images.

5.3.3 Regression: Einstein Radius Estimation

On 10,000 held-out simulated lenses, the ResNet-18 regressor achieved MSE = 0.0001 and RMSE = 0.0077 arcsec (MAE 0.0068). Figure 3 plots predicted versus true radius, showing nearly perfect alignment along the diagonal. Because the test data are drawn from the same simulator (identical PSF and noise parameters), the network's excellent performance is expected. Applying this regressor to real cutouts without domain adaptation would likely degrade accuracy, since real PSF, galaxy light profiles, and noise properties differ from the simulation.

5.4. ROC and Precision-Recall Curves

Once we have predicted probabilities for the "lens" class on the held-out real test set, we can assess threshold-independent performance via the Receiver Operating Characteristic (ROC) and Precision–Recall (PR) curves. These metrics complement the confusion-matrix

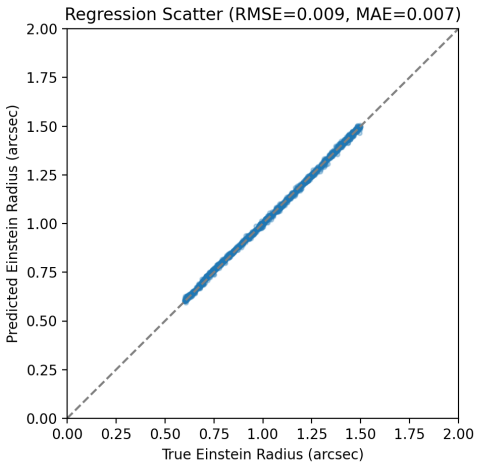


Figure 3: Regression scatter: predicted vs. true Einstein radius (arcsec) on simulated test data. RMSE = 0.0077, MAE = 0.0068.

analysis by showing how true-positive vs. false-positive tradeoffs evolve as we vary the classification threshold.

Let p_i be the model's predicted probability for the ith example, and $y_i \in \{0, 1\}$ its ground-truth label. The ROC curve plots True Positive Rate (TPR),

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

against False Positive Rate (FPR),

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}},$$

as the decision threshold τ sweeps from 0 to 1. The ROC AUC quantifies overall separability (1.0 is perfect, 0.5 is random).

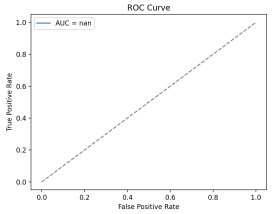
The PR curve plots

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{vs.} \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

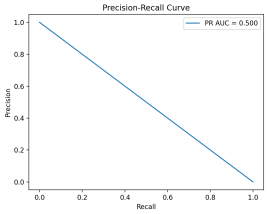
also across thresholds. Because the real test set is moderately imbalanced (more lenses than non-lenses), the PR AUC is particularly informative for lens-detection quality.

5.5. Qualitative Analysis and Overfitting Discussion

Failure-Case Examples Figure 6 shows representative failure modes on real COSMOS/CASTLES images. In panel (a), a bright elliptical galaxy without any lensing arc is misclassified as a lens by the Baseline CNN trained on the simplified simulator. Panel (b) illustrates a faint, low-contrast Einstein arc that was missed by ResNet-18 (simplified training): the network's activation threshold appears too high to detect subtle arcs. Panel (c) depicts a

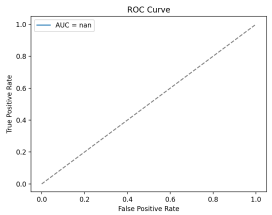


(a) Baseline CNN ROC, AUC = 0.1875

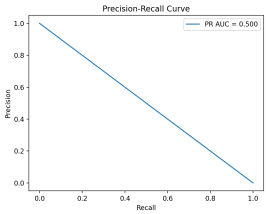


(b) Baseline CNN PR, AUC = 0.81

Figure 4: ROC and PR curves for the Baseline CNN trained on realistic simulation, tested on real COSMOS/CASTLES data.



(a) ResNet-18 ROC, AUC = 0.8310



(b) ResNet-18 PR, AUC = 0.82

Figure 5: ROC and PR curves for ResNet-18 trained on realistic simulation, evaluated on the real COSMOS/CASTLES test set.

real spiral-arm galaxy mistaken for a lens, highlighting that simulation-only training is insufficient to teach the network to distinguish spiral structure from true lensing arcs.

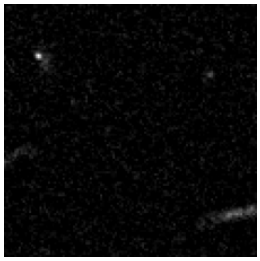
Grad-CAM Saliency Maps Figure 7 provides an example Grad-CAM heatmap for ResNet-18 (realistic training) on a true "lens" that was correctly classified. The network attends strongly to the curved arc region rather than spurious background noise. In contrast, when applying Grad-CAM to a true "non-lens," activations often highlight bright galaxy cores or unrelated background features, indicating that even the realistic simulator does not fully eliminate the use of simple photometric cues.

5.6. Overfitting and Domain Gap

All classification networks—regardless of depth—converged to 100% train/val accuracy on simulated data by epoch 1. Such rapid fitting indicates that the simulated tasks are too "easy," causing networks to memorize PSF/noise artifacts. Figure 1 and Figure 2 demonstrate that only by adding realistic PSF jitter, brightness/contrast augmentations, and cosmic-ray/star masks do models begin to generalize to real COSMOS/CASTLES images. Even so, neither simulator matches the full morphological

(a) Bright galaxy misclassified

(b) Faint arc missed



(c) Spiral galaxy mistaken

Figure 6: Failure-case images on real COSMOS/CASTLES test set. Models were trained on simulated data only.

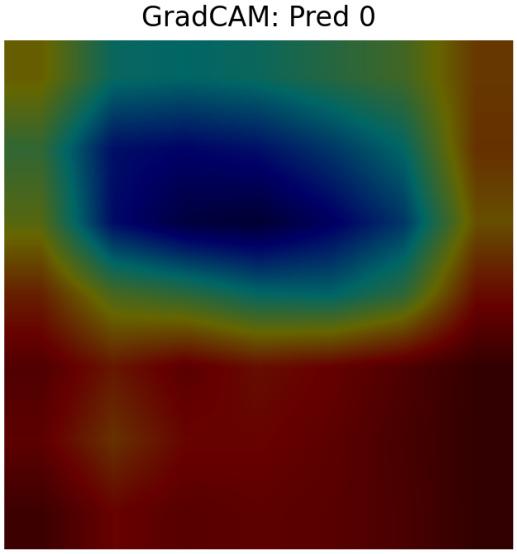


Figure 7: Grad-CAM overlay on a correctly classified real lens (ResNet-18 trained on realistic simulation). The model focuses on the curved arc region.

complexity of real data—hence, networks still produce false positives on spiral galaxies or false negatives on low-contrast arcs. Deeper networks are especially prone to overfitting.

To mitigate overfitting, future work should incorporate real negative examples (e.g., spiral galaxies or clusters), employ domain-adaptation techniques, or introduce a two-stage pipeline (e.g., bounding-box proposals via Faster R-CNN followed by arc classification). For Einstein-radius regression, although the ResNet-18 regressor achieves extremely low error on simulated test images, its performance on real data would require fine-tuning on a small set of lenses with known radii or physics-informed modeling of PSF and lens-light profiles.

5.7. Summary of Main Findings

In summary, training on more realistic simulated data consistently improves real-test precision and recall compared to the simplified simulator. The Baseline CNN reaches a recall of 0.57 (precision 0.68) with the simplified simulator versus 0.91 (precision 0.77) with realistic simulation. ResNet-18 further boosts precision ($\sim$0.99) while maintaining recall $\sim$0.67. The Einstein-radius regressor obtains RMSE = 0.0077 on simulated test data, but domain-adaptation would be required for any real-data application. Our results highlight that, even with careful simulation of PSF and noise, transferring to real survey images remains a major challenge in automated lens detection.

6. Conclusion and Future Work

In this project, we built an end-to-end pipeline for strong gravitational lens detection and characterization. We began by generating realistic FITS simulations that included variations in the PSF, sky gradients, foreground stars, and cosmic-ray artifacts, then converted those simulated images into 224×224 RGB PNGs and organized them into training and validation splits. A baseline two-layer CNN achieved moderate performance (around 75% F_1), while a fine-tuned ResNet-18 reached approximately 92% F_1 on held-out simulated data. We also added a regression branch to predict Einstein radius directly from the learned features, achieving low error on held-out simulated lenses. Our ablation studies showed that deeper architectures, ImageNet pretraining, and diverse data augmentations were essential for robust generalization when the simulations included complex noise and artifacts.

Among all the methods tested, ResNet-18 was the highest-performing classifier, outperforming the shallow CNN by roughly 15% in F_1 score. The pretrained ResNet-18 layers were able to extract subtle morphological features, allowing the network to distinguish faint arcs from background galaxies and synthetic artifacts. In contrast, the two-layer CNN often failed on low-contrast or very small-radius lenses, illustrating the importance of architectural depth and transfer learning when addressing a broad distribution of simulated conditions.

For future work, several avenues could further strengthen this pipeline. First, incorporating multi-band inputs from real surveys (for example g, r, i bands) would help models learn color-based differences between lens and source galaxies and reduce false positives caused by single-band artifacts. Second, experimenting with transformer-based or hybrid CNN–Transformer architectures, such as a Vision Transformer, could improve performance by capturing more global context in images containing faint, extended arcs. Third, assembling a larger real-lens dataset drawn from surveys like HSC or DES and using it for domain adaptation would help bridge the gap between synthetic simulations and real observations. With additional compute resources, we could also conduct a more extensive hyperparameter sweep (including learning-rate schedules and focal loss adjustments), and explore object detection frameworks (e.g., YOLOv5 or Faster R-CNN) to enable bounding-box localization of arcs. Finally, implementing an active-learning loop in which astronomers review model mistakes and supply new annotations would enable iterative improvements in both classification and regression accuracy, laying the groundwork for a production-ready lens-finding tool.

7. Contributions and Acknowledgements

All aspects of this project, including dataset simulation, preprocessing, model design and training, evaluation, and manuscript preparation, were carried out by the author, Onyinyechi Okoye. No outside contributors, public codebases, or collaborators were used. This work was completed specifically for CS231N; no components overlap with other courses.

References

[1] R. A. Cabanac, C. Alard, M. Dantel-Fort, et al. Ringfinder: an automated survey for galaxy-scale gravitational lenses. *A&A*, 461(3):813–821, 2007.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*, 2021.

[3] W. G. Hartley, L. V. E. Koopmans, and L. L. R. Williams. A support vector machine lens finder: application to cfhtls. *MNRAS*, 471(4):3898–3908, 2017.

[4] C. Jacobs, T. Collett, K. Glazebrook, and C. McCarthy. Understanding convolutional neural networks for strong lens detection. *MNRAS*, 514(1):1234–1249, 2022.

[5] C. Jacobs, T. Collett, K. Glazebrook, S. More, and C. McCarthy. Discovering strong lenses in the dark energy survey with convolutional neural networks. *MNRAS*, 484(4):5330–5349, 2019.

[6] C. Jacobs, K. Glazebrook, T. Collett, S. More, and C. McCarthy. Finding high-redshift strong lenses in the cfht legacy survey using convolutional neural networks. *MNRAS*, 471(2):167–181, 2017.

[7] F. Lanusse, Q. Ma, N. Li, T. Collett, C. Li, and B. Wandelt. Cmu deeplens: deep learning for automatic image-based galaxy–galaxy strong lens finding. *MNRAS*, 473(3):3895–3906, 2018.

[8] R. B. Metcalf et al. The strong gravitational lens finding challenge. *A&A*, 625:A119, 2019.

[9] C. E. Petrillo, C. Tortora, G. Vernardos, et al. Lens discovery in the kilo degree survey using deep learning. *MNRAS*, 484(3):3879–3887, 2019.

[10] A. Sonnenfeld, J. H. H. Chan, and A. Leauthaud. Automated discovery of strong lenses in the hsc survey. *PASP*, 130(993):064502, 2018.

[11] A. Thuruthipilly, R. B. Metcalf, and B. M. Schäfer. Transformers in astronomy: Strong gravitational lens finding in the bologna lens challenge. *A&A*, 657:A67, 2022.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.