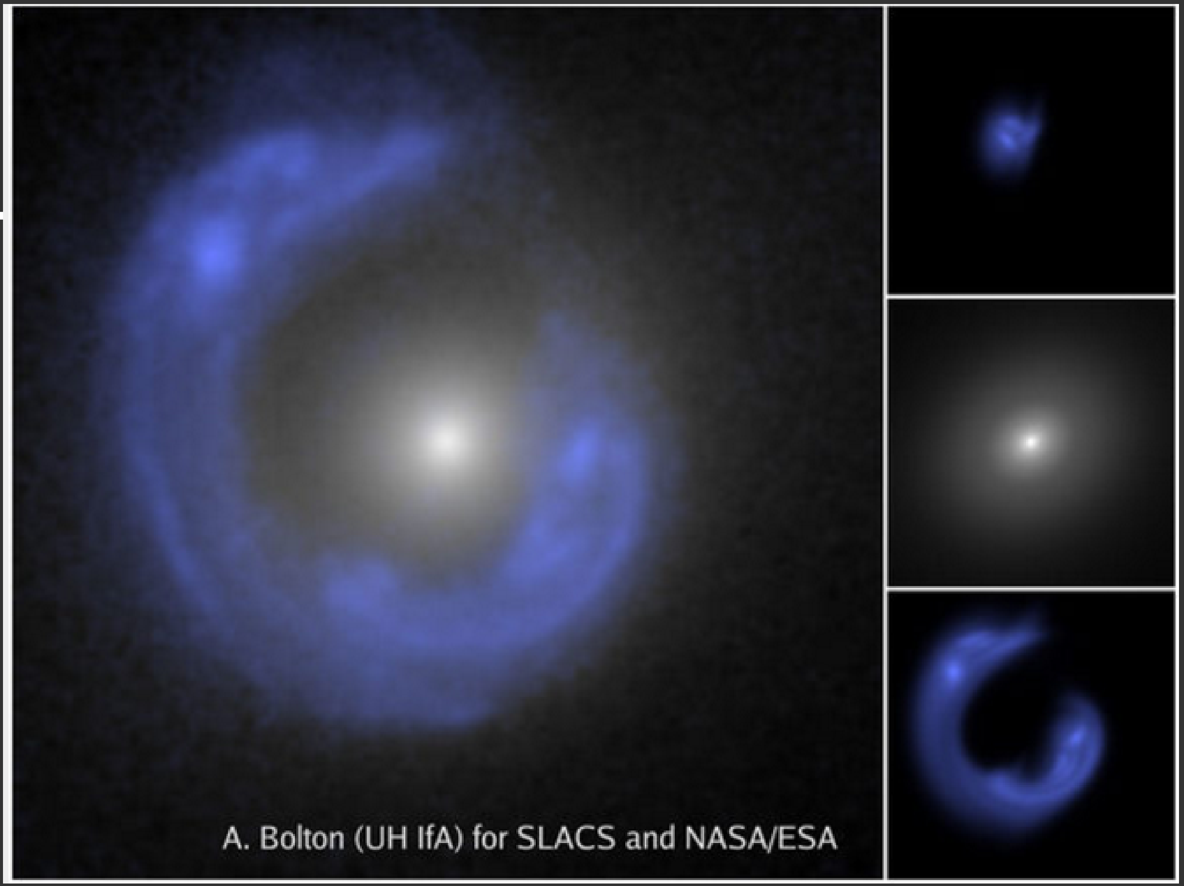
Strong Lensing analysis using Deep Neural Networks



Nesar Ramachandra
Wasikul Islam
Nan Li
J. Taylor Childers
Prasanna Balaprakash
Lindsey Bleem
Salman Habib

18 July 2017

Argonne NATIONAL LABORATORY

KU THE UNIVERSITY OF KANSAS

Motivation:

- Strong galaxy-galaxy lensing
  - Details of matter density profiles, evolution
  - Constrain cosmological constants

- Strong Lensing detection
  - Visual inspection
  - Automated codes using
    - Morphology
    - Machine learning

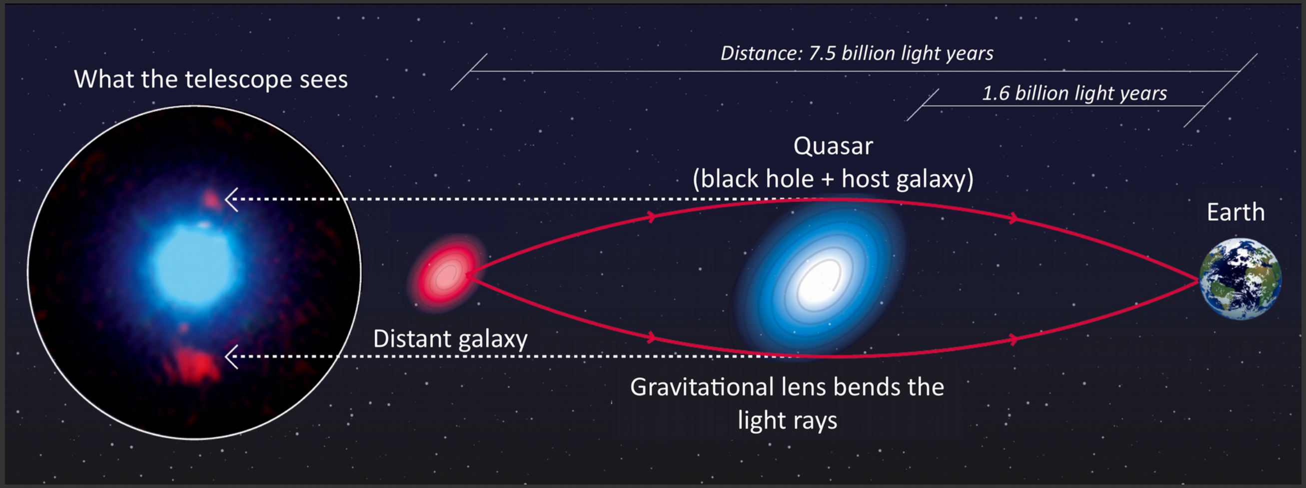
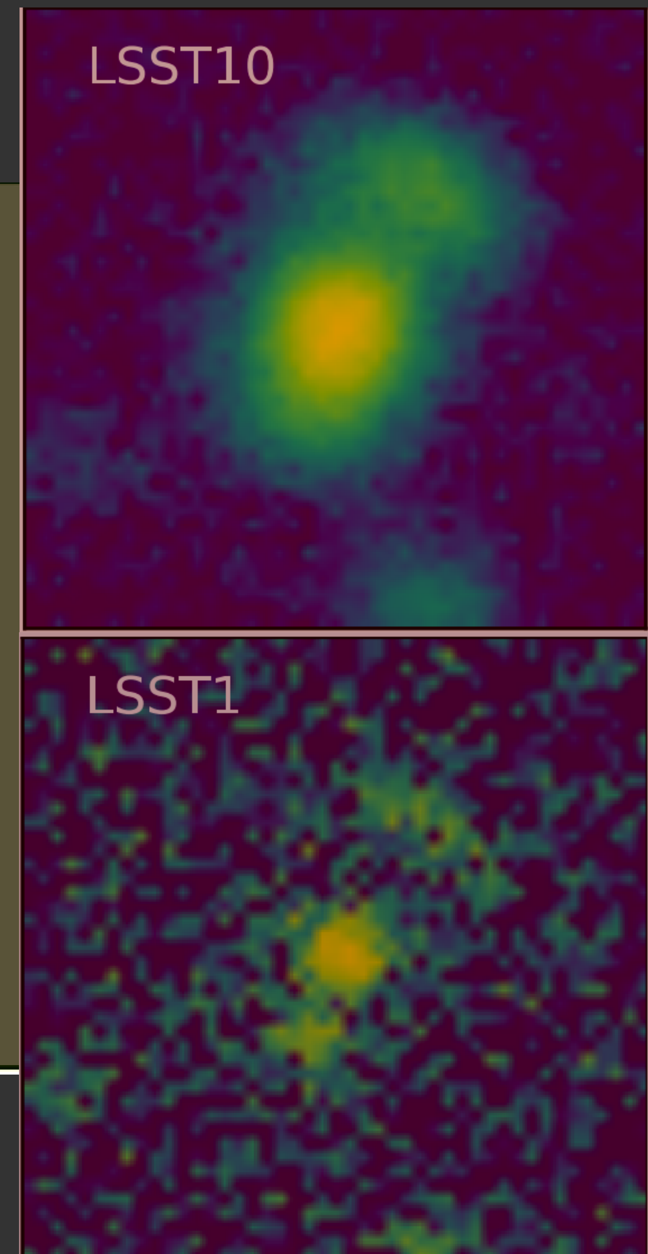


Image credits: F. Courbin, S. G. Djorgovski, G. Meylan, et al., Caltech / EPFL / WMKO

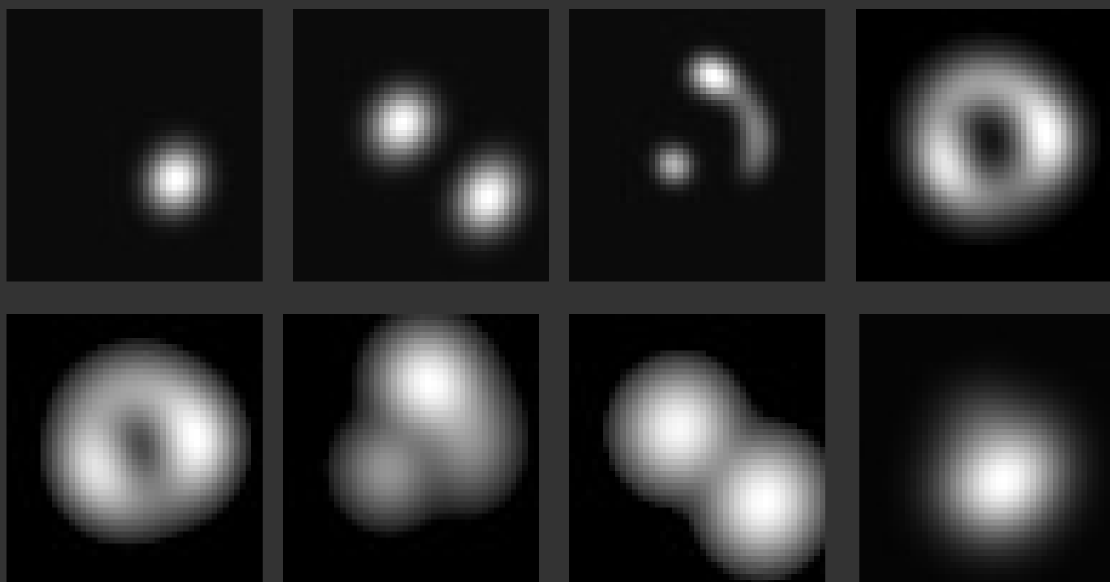
- Expected number of galaxy-galaxy strong lenses (eg. Collett 2015)
    - DES: 2,400
    - LSST: 120,000
    - Euclid: 170,000

Mock telescope images: (Avestruz et. al. 2017)

- Present:
  - HST: high res, low noise
- Future:
  - LSST: low res, 6 bands
  - Euclid: high res, gray-scale



Simulated images (From Nan Li)



← Noiseless single

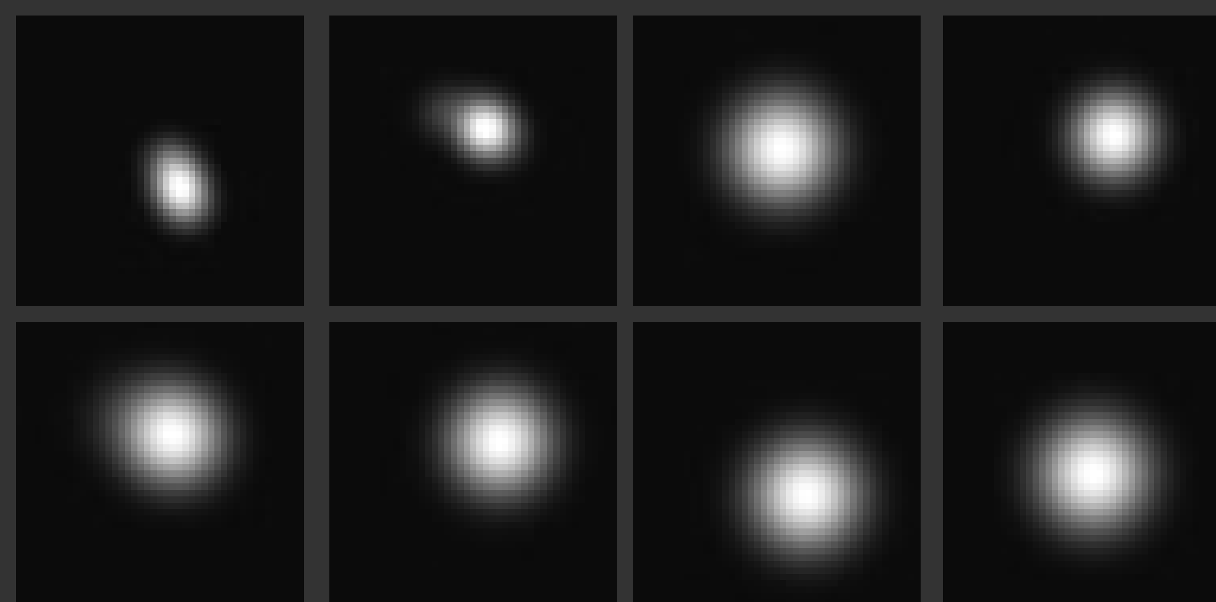
← Lensed

← Noiseless stack

Noiseless single →

Unlensed →

Noiseless stack →

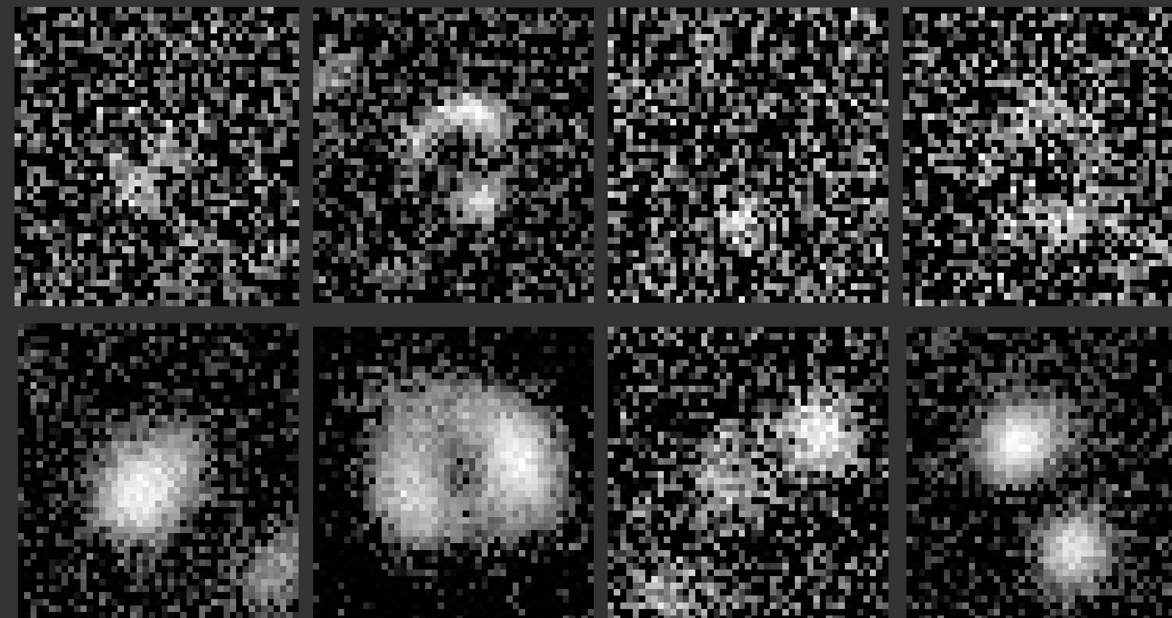


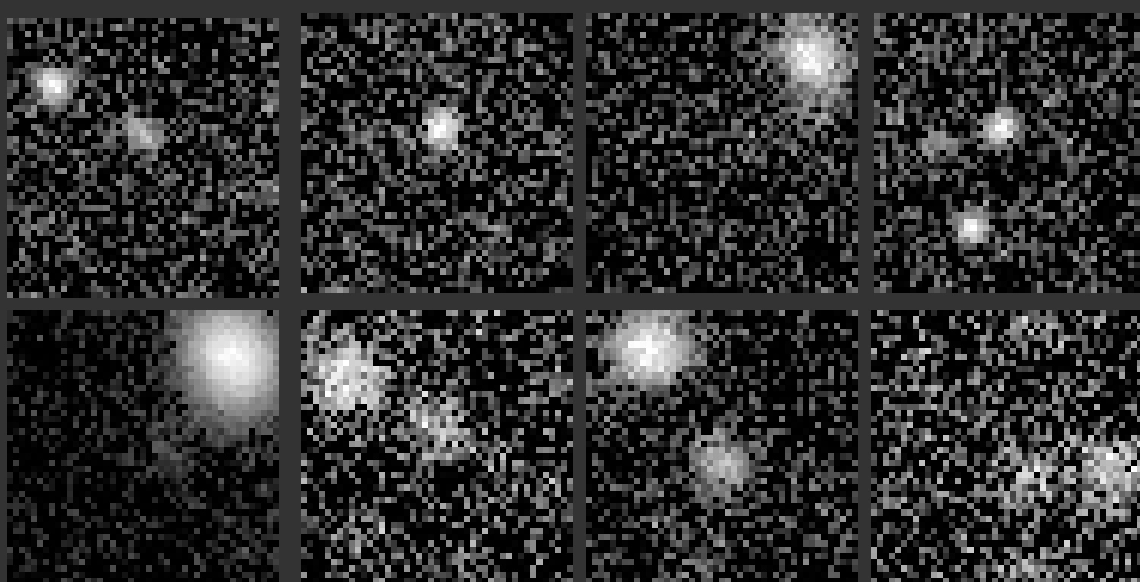
Noisy data for training: 8,000 each, 45x45 pixels (0.18 arcsec/pixel)

Lensed →

Noisy single →

Noisy stack →





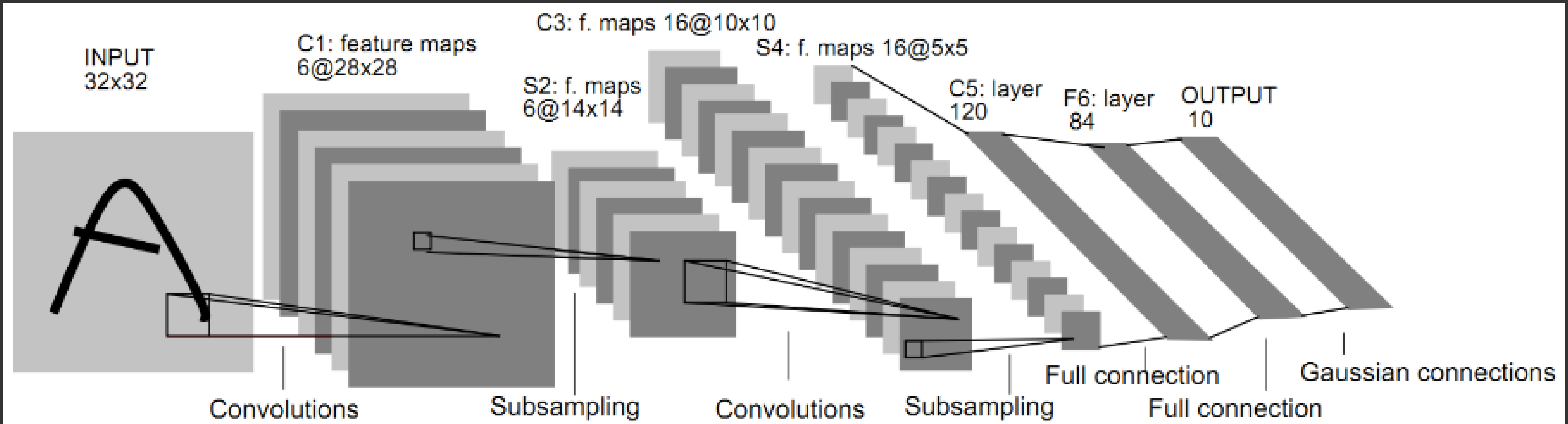
← Noisy single

← Unlensed

← Noisy stack

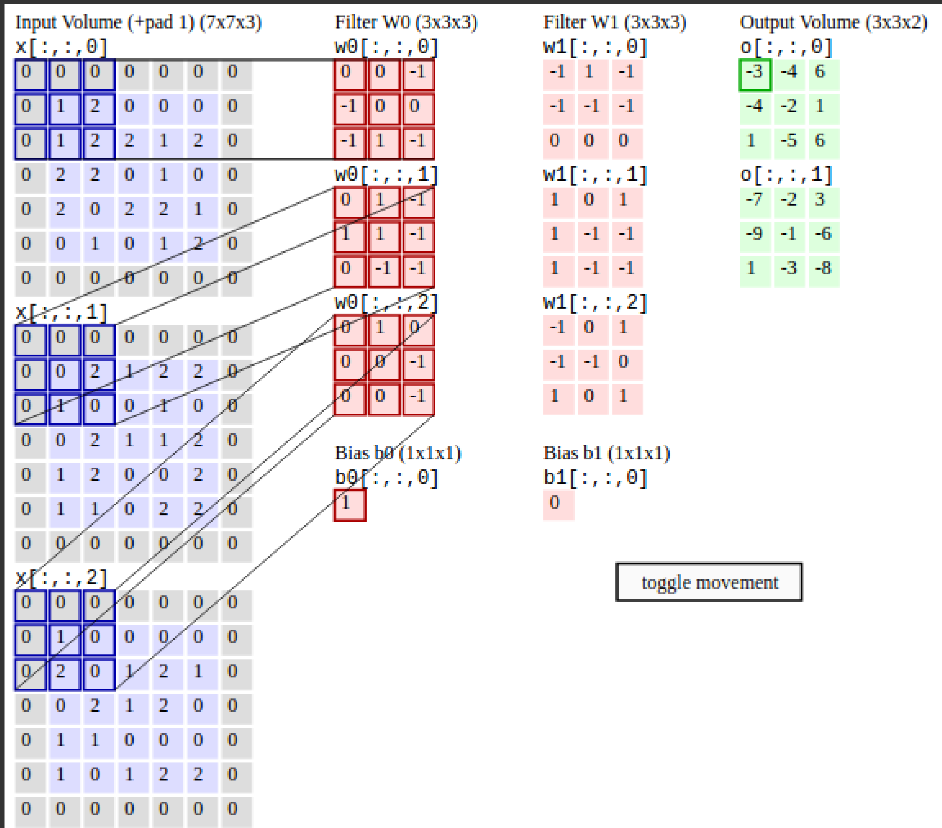
Convolutional neural networks (CNNs/ConvNets)

- Been around since early 1990s;
  - Recently became mainstream due to GPUs. Surpassed human ability ~ 2015
- Applied in new image recognition systems, language processing, AlphaGo
- Lensing images study:
  - Petrillo et. al. 2017 (Kilo Degree Survey)
  - Lanusse et. al. 2017 (CMU-DeepLens)



LeNet, 1998

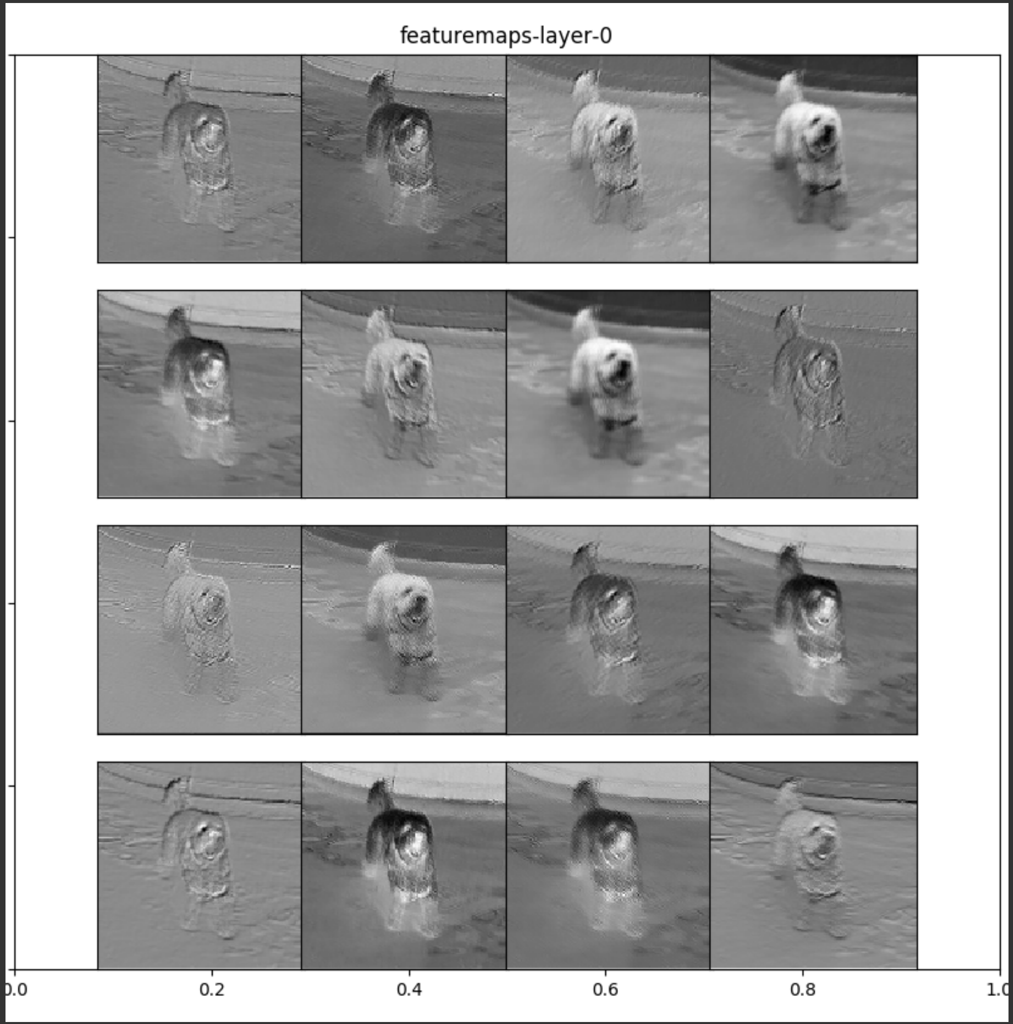
Convolution layer



http://cs231n.github.io/convolutional-networks/

- Filters/kernels of various types
  - Stride through every image
  - Pick up features, which are used as inputs for activation

Convolution →



Other layers

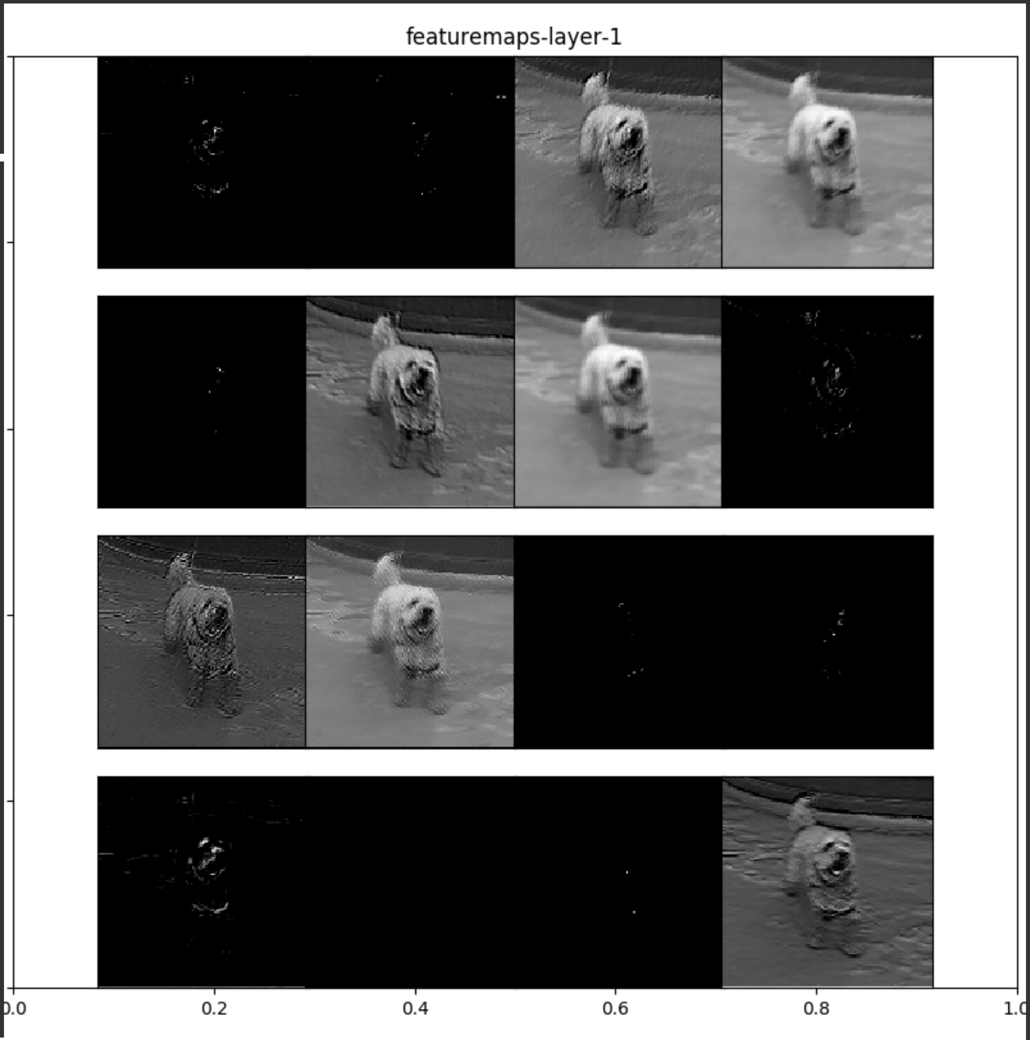
- Activation layer
  - Introduces non-linearity
  - Activation function: f(x) = max(0, x) applied to all the values of input array

- Pooling layer

- Dropout layer

- Dense layer
  - Fully connected layer that checks correlation between input and output
  - Generally around the final layers

→

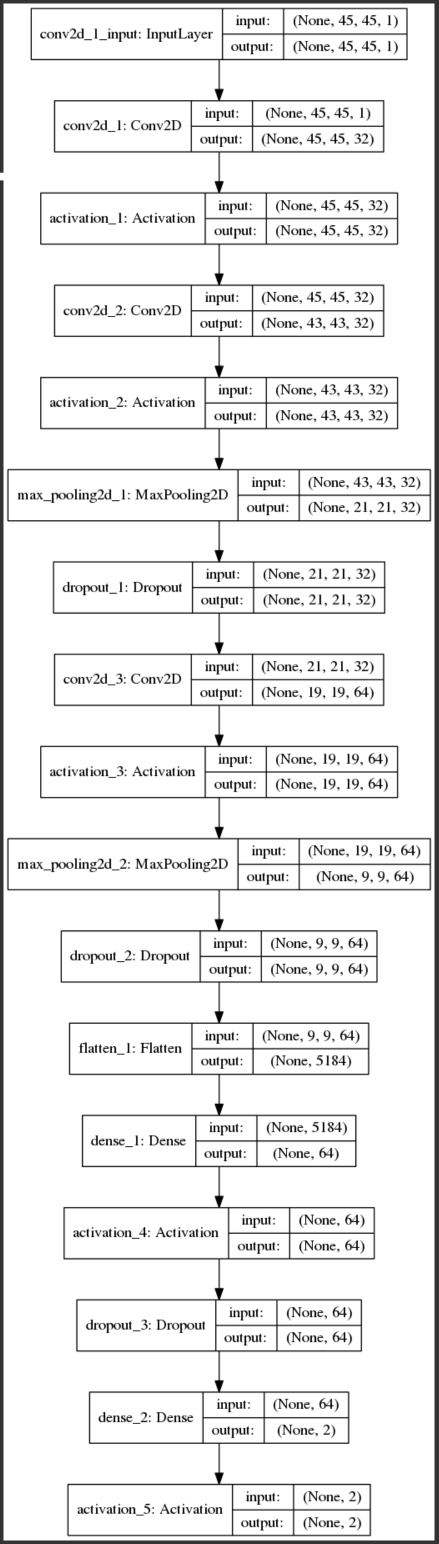


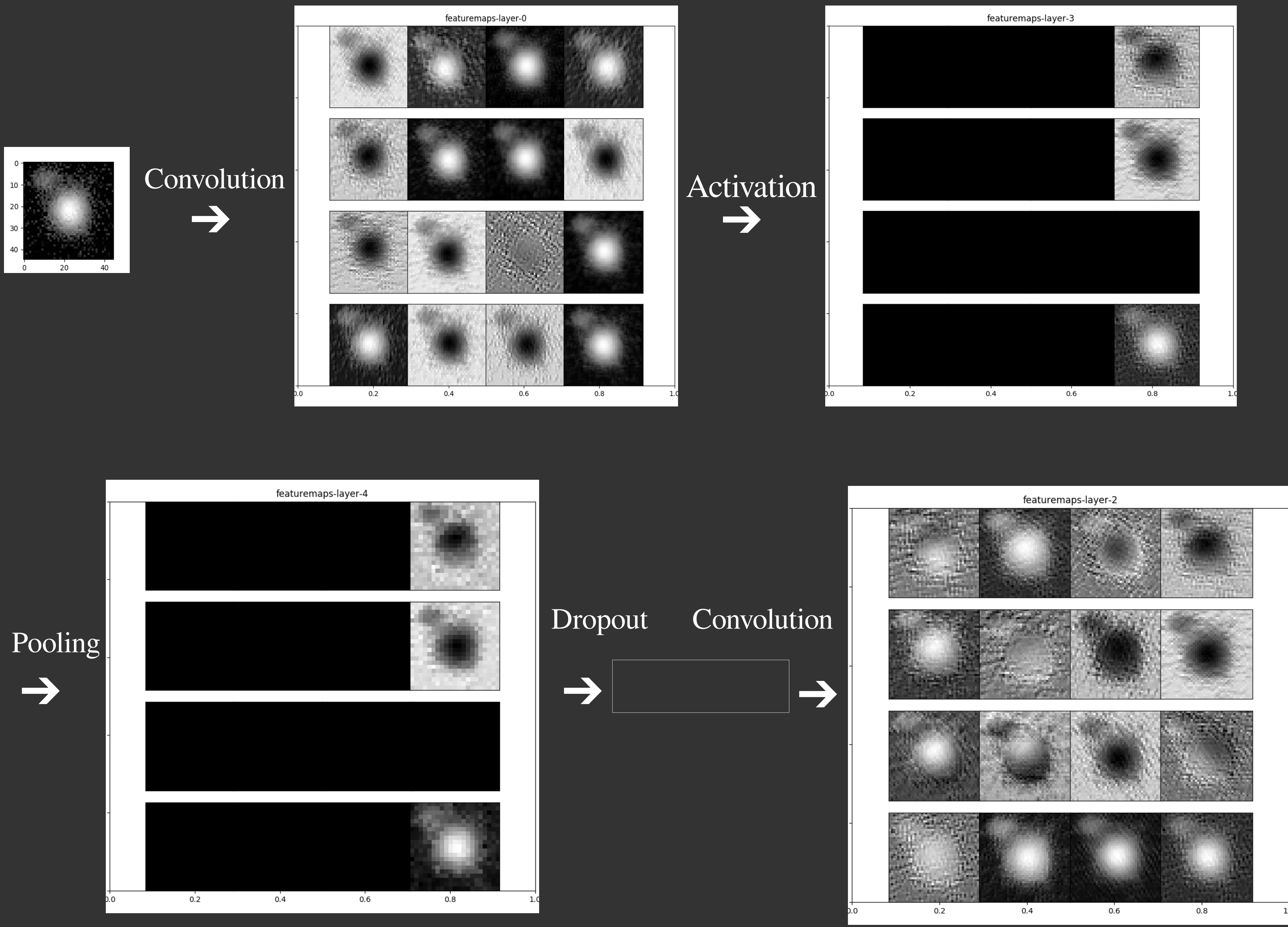
- Backpropagation
  - End of every epoch, predicted labels are checked against real labels, and loss (error) is calculated.

  - We try to minimize this error in the next epoch, by updating weights

Our SL detection framework

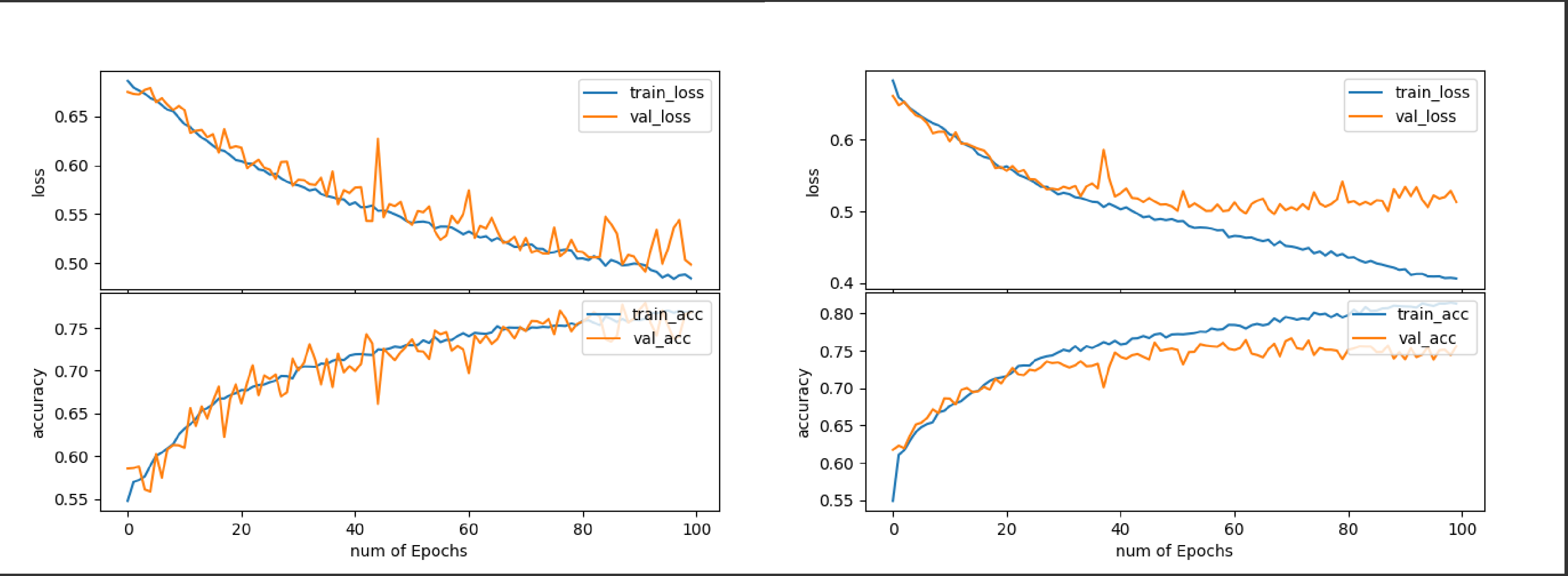
- Our network currently has 17 layers.
  - **Input image [45x45]** → Conv → Actv → Conv → Act → Pool → Drop → Conv → Actv → Pool → Drop → Flat → Dense → Actv → Drop → Dense → Actv → **Output label [Prob(lensed), 1-Prob(lensed)]**
  - Can be made deeper or wider
- Lots of parameters to optimize: order of 10^6
- Hyper-parameters to choose ~ 10 to 15
  - Learning rate, decay rate
  - Number of epochs
  - Batch size
  - Dropout percentage
  - Back-propagation optimizers (SGD, RMSprop)
  - Loss functions




Convolution
featuremaps-layer-0
Activation
featuremaps-layer-3
Pooling
featuremaps-layer-4
Dropout
Convolution
featuremaps-layer-2

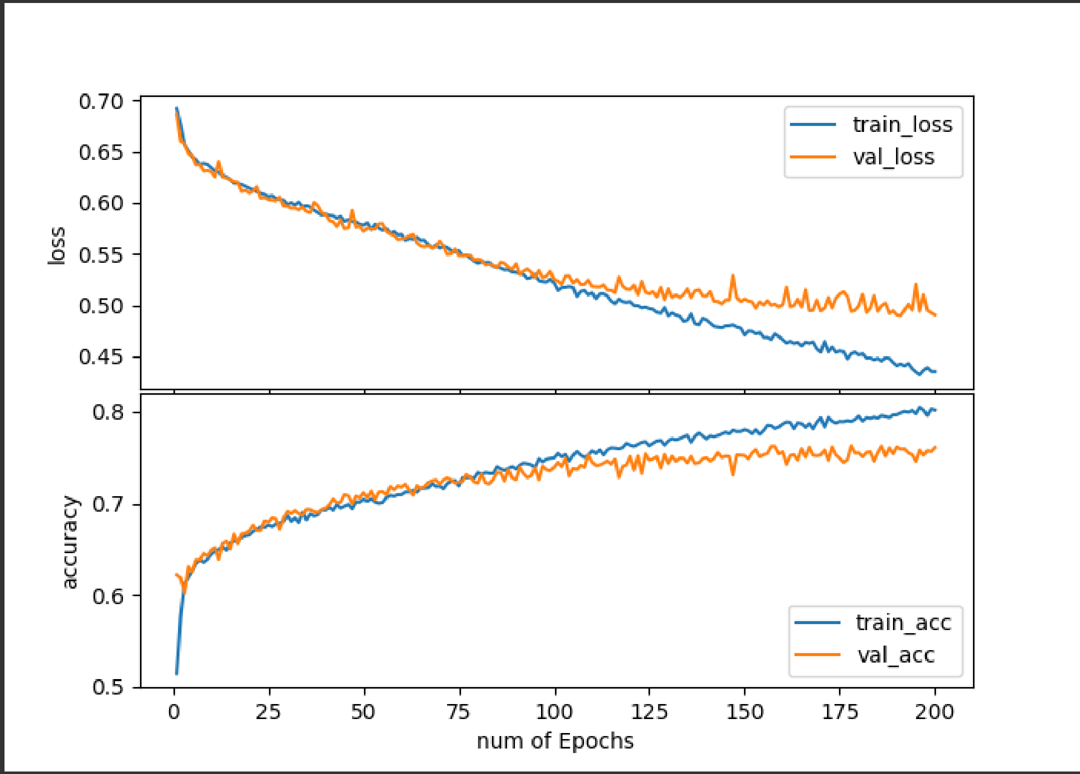
Hyper-parameters fine-tuning

- No quick rule to find the best hyper-parameters
  - Sweep across all ranges, or choose randomly
- Monitor a few values during training and decide from there:
  - Loss, Validation loss – how good are the weights
  - Accuracy, Validation accuracy – how accurate is the model



Sample training

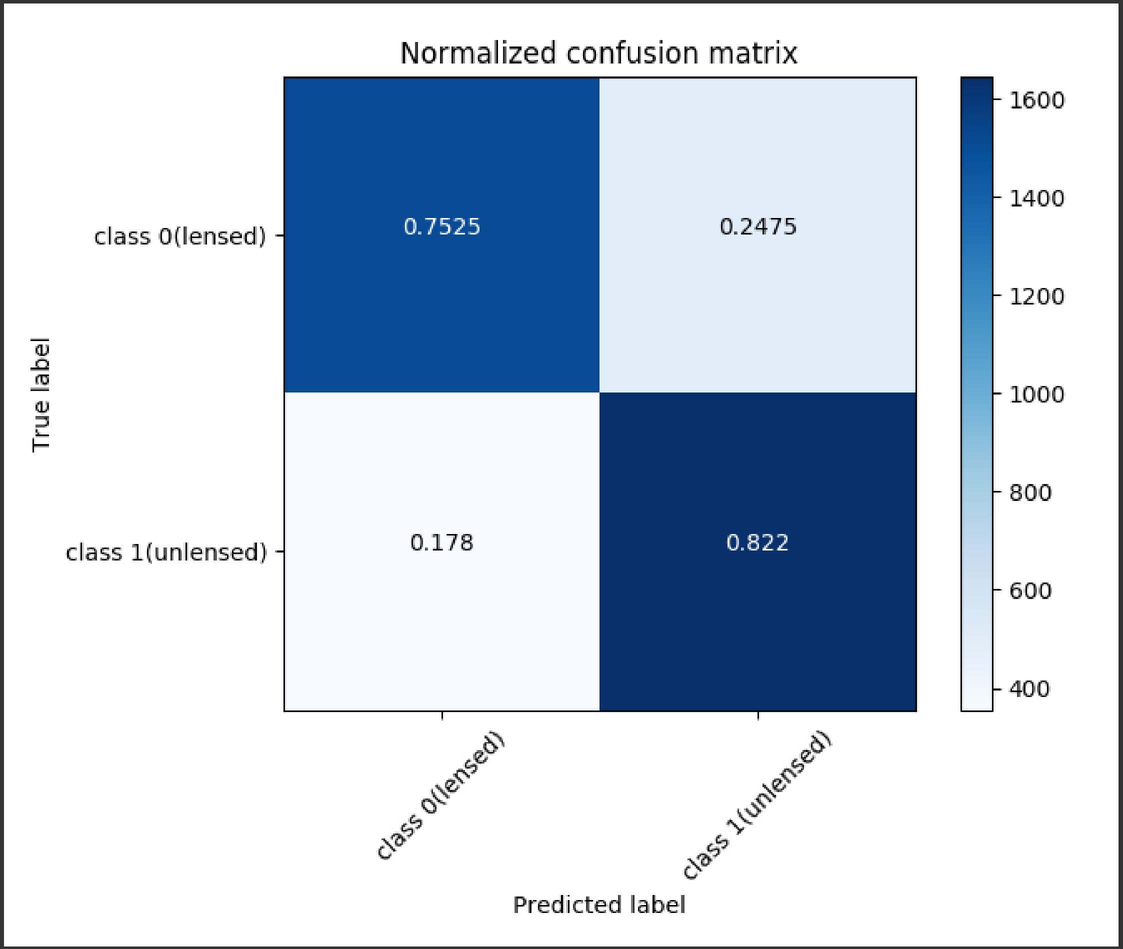
- Loss decreases, accuracy increases with epochs
- Deviation of validation loss/accuracy after 100 epochs
- Hyper-parameters
  - Learning rate: 0.001
  - Decay rate: 0.01
  - Total epochs: 200
  - Batch size: 32



- 80 per cent accuracy in 30 mins on 1 Intel-Haswell node with 16 CPU cores (Cori)
- About the same time on NVIDIA GeForce GT 755M with 384 cores.

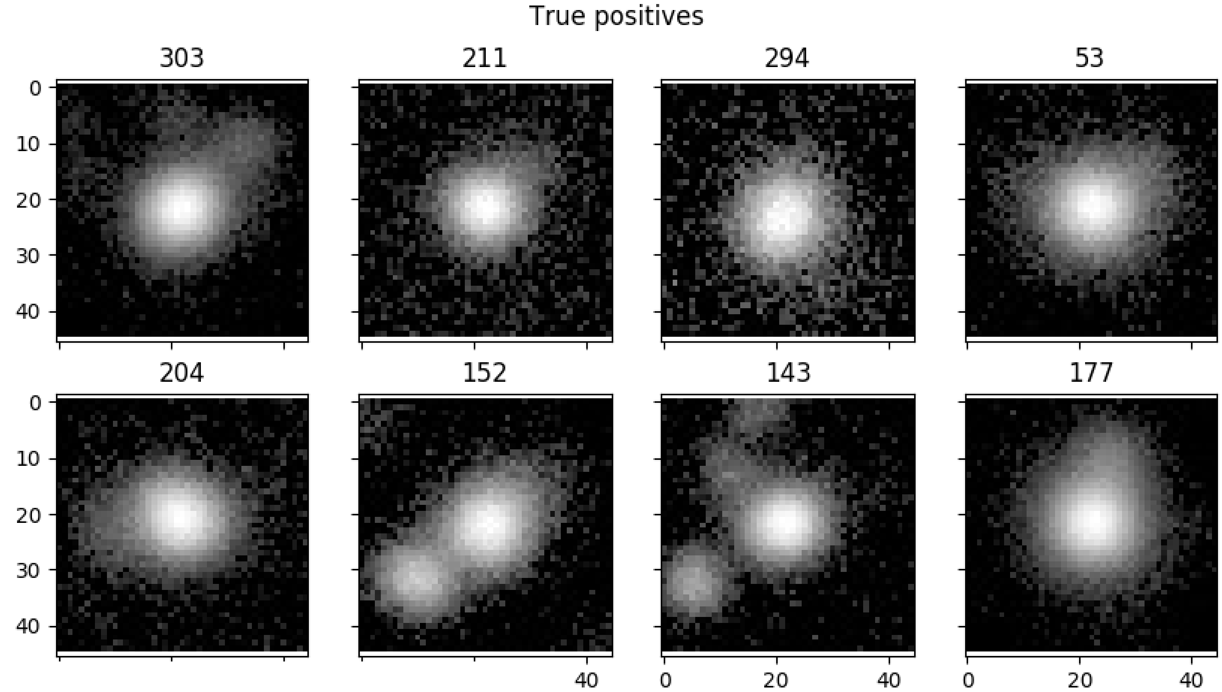
Testing data:

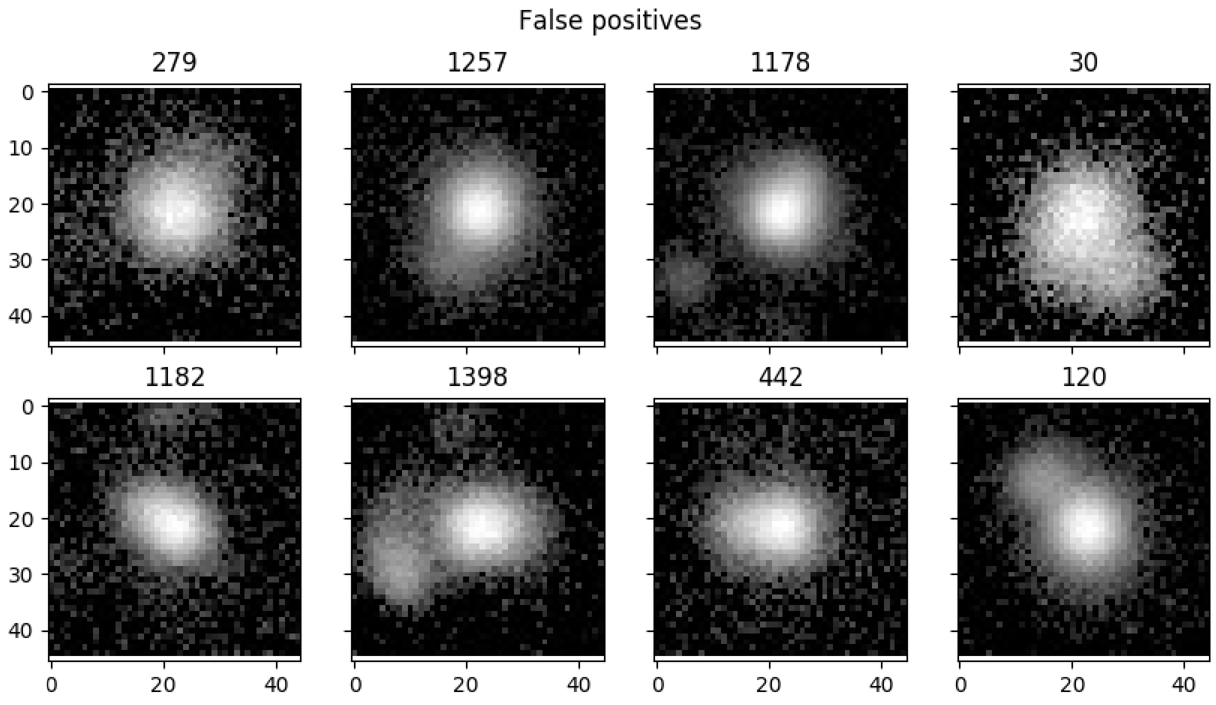
- Using the fully trained model, with optimized weights
- Testing on completely new data
- Classification time: $O(10^{-3})$ seconds per image
- Confusion matrix: probabilities of correct and incorrect classifications.



Preliminary testing results: True and False positives

Correct classifications →





← Incorrect classifications

Summary and future plans

- We’ve reached 80-90 percent accuracy within 200 epochs.
  - Around 75-82 percent accuracy on new images
  - Lot of improvements can be made:
    - Data augmentation
    - Better hyper-parameter sweeps
    - Deeper architectures can be trained using the state-of-the-art GPUs at Argonne.

- Quantitative analysis of strong lensing
  - Can we constrain properties of the lens using simulation-trained ConvNets?
  - Currently we are working on regression problems

Questions?

https://xkcd.com/1838/

References

- https://github.com/hep-cce/ml_classification_studies
- http://cs231n.github.io/convolutional-networks/
- CMU DeepLens: Deep Learning For Automatic Image-based Galaxy-Galaxy Strong Lens Finding, Lanusse et. al. 2017 (arXiv: 1703.02642)
- Finding Strong Gravitational Lenses in the Kilo Degree Survey with Convolutional Neural Networks, Petrillo et. al. 2017 (arXiv: 1702.07675)
- Automated Lensing Learner - I: An Automated Strong Lensing Identification Pipeline, Avestruz et. al. 2017 (arXiv: 1704.02322)
- The population of galaxy-galaxy strong lenses in forthcoming optical imaging surveys, Collett (arXiv: 1507.02657)