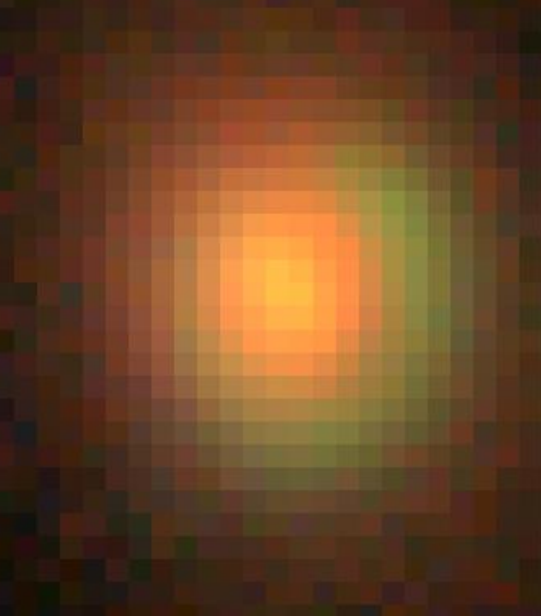
Efficient mass modeling of strong lenses through deep learning

HOLISMOKES IV. - Schuldt et al. (2021)



Stefan Schuldt (MPA/TUM)

"Time-Domain Cosmology with Strong Gravitational Lensing" workshop

January, 2021

Upcoming Lens Detections

Upcoming surveys (like LSST) will find hundred thousands of strongly lensed galaxies

-> automated and fast way to model

-> Convolutional Neural Network trained on images

Upcoming Lens Detections

Upcoming surveys (like LSST) will find hundred thousands of strongly lensed galaxies

-> automated and fast way to model

-> Convolutional Neural Network trained on images

- statistical studies
- needed for follow-up observations

Upcoming Lens Detections

Upcoming surveys (like LSST) will find hundred thousands of strongly lensed galaxies

-> automated and fast way to model

-> Convolutional Neural Network trained on images

- statistical studies
- needed for follow-up observations

-> assume SIE profile: lens center, position angle, axis ratio, and Einstein radius

Previous work

Previous work

- Hezaveh et al. 2017, Levasseur et al. 2017

  -> modeling of HST-like lensed arcs, single filter

  -> removing of lens necessary

Previous work

- Hezaveh et al. 2017, Levasseur et al. 2017
  - -> modeling of HST-like lensed arcs, single filter
  - -> removing of lens necessary
- Pearson et al. 2019
  - -> fully mocked up images

Previous work

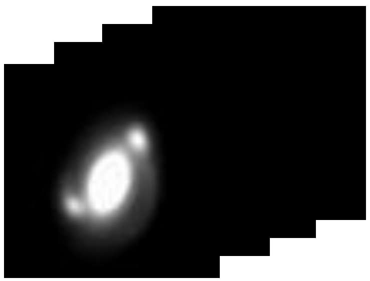
- Hezaveh et al. 2017, Levasseur et al. 2017
  - -> modeling of HST-like lensed arcs, single filter
  - -> removing of lens necessary
- Pearson et al. 2019
  - -> fully mocked up images
- Schuldt et al. (2021)
  - -> use real galaxy observation
  - -> only simulate lensing effect
  - -> additionally predict lens center

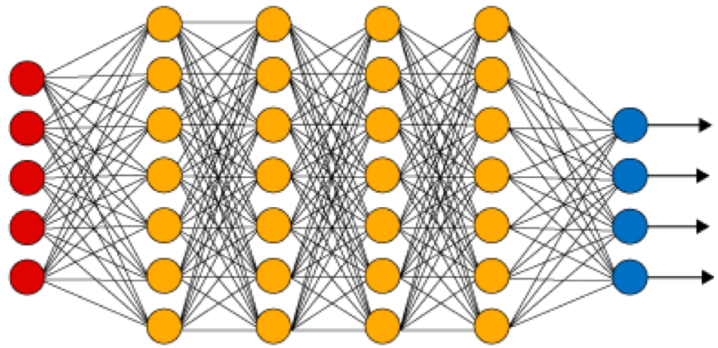
Network architecture

(hidden) layers

Inspired by LeNet

Input: images





Output:

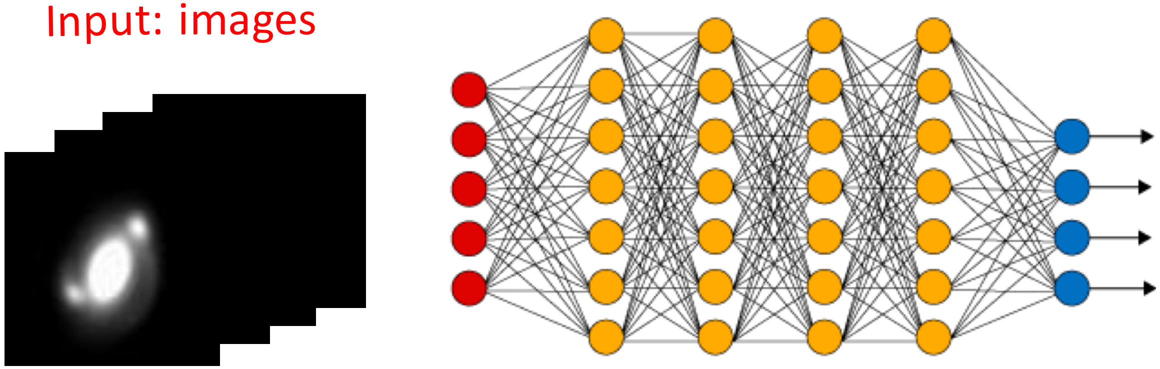
$\beta = (\mathbf{x}, \mathbf{y}, \mathbf{e}_x, \mathbf{e}_y, \boldsymbol{\theta}_E)$

Network architecture

(hidden) layers

Inspired by LeNet

Input: images

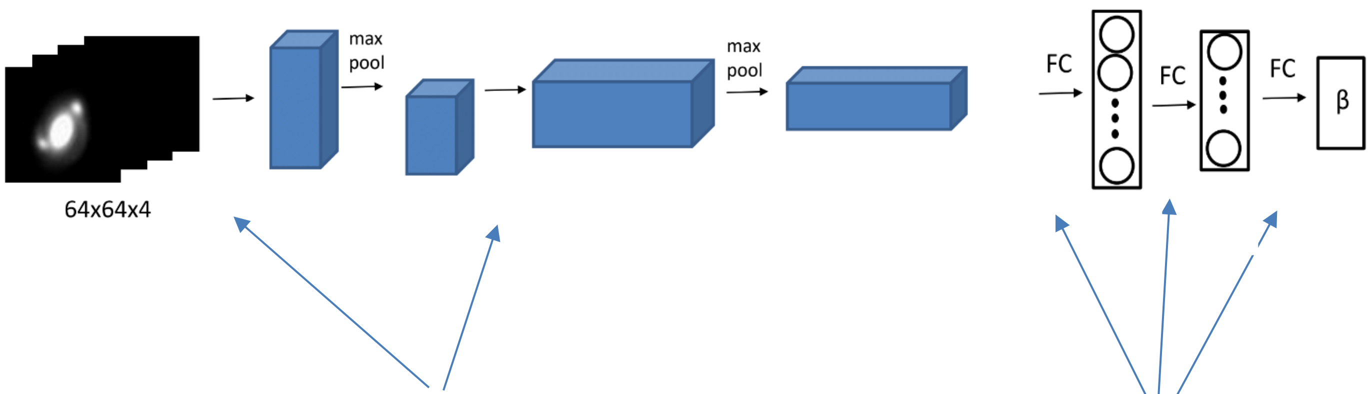


Output:

$\beta = (x, y, e_x, e_y, \theta_E)$

Input: images

output:



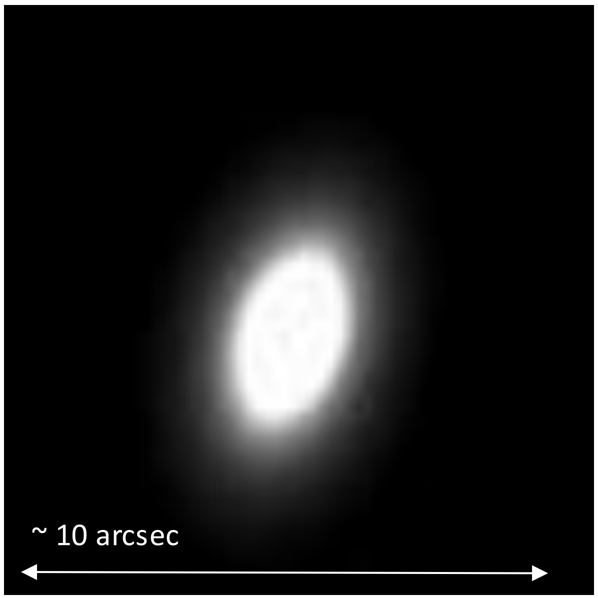
convolutional layers

fully connected layers

mock up images

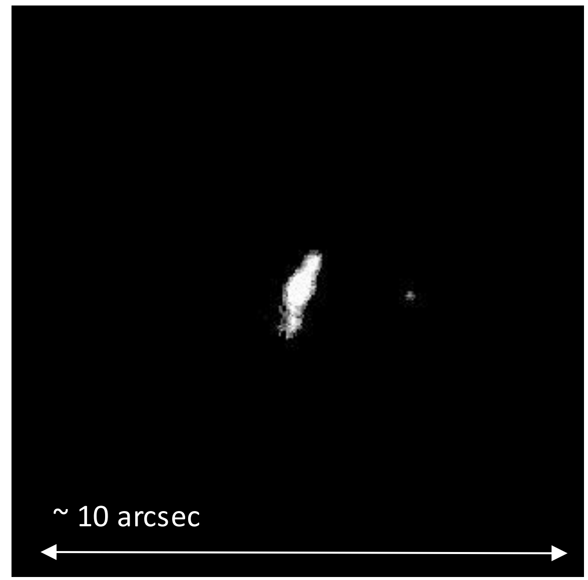
based on real observed galaxy images

Lens



HSC with SDSS velocity dispersion
-> axis ratio and position angle
-> with Gaussian spread for possible offset of mass from light distribution

Source



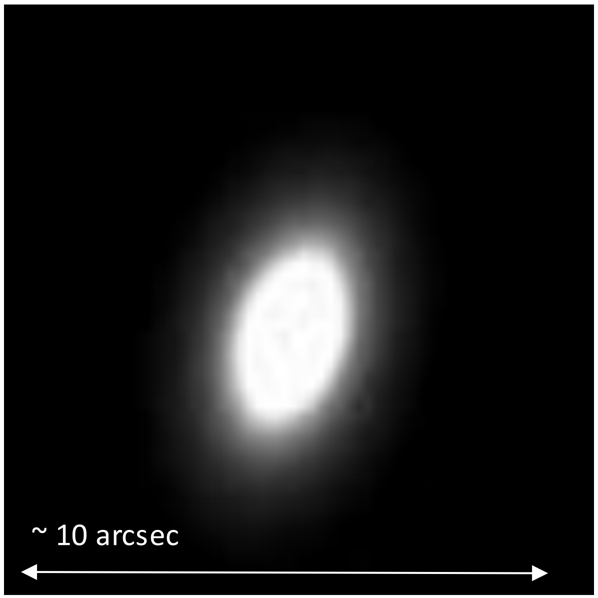
HUDF images, masked out

-> high redshift, high resolution
-> random picked galaxy for given lens and randomly located

mock up images

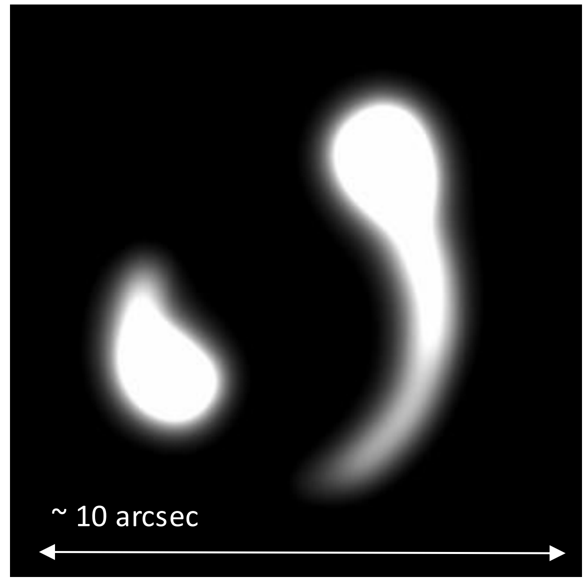
based on real observed galaxy images

Lens



HSC with SDSS velocity dispersion
-> axis ratio and position angle
-> with Gaussian spread for possible offset of mass from light distribution

Source



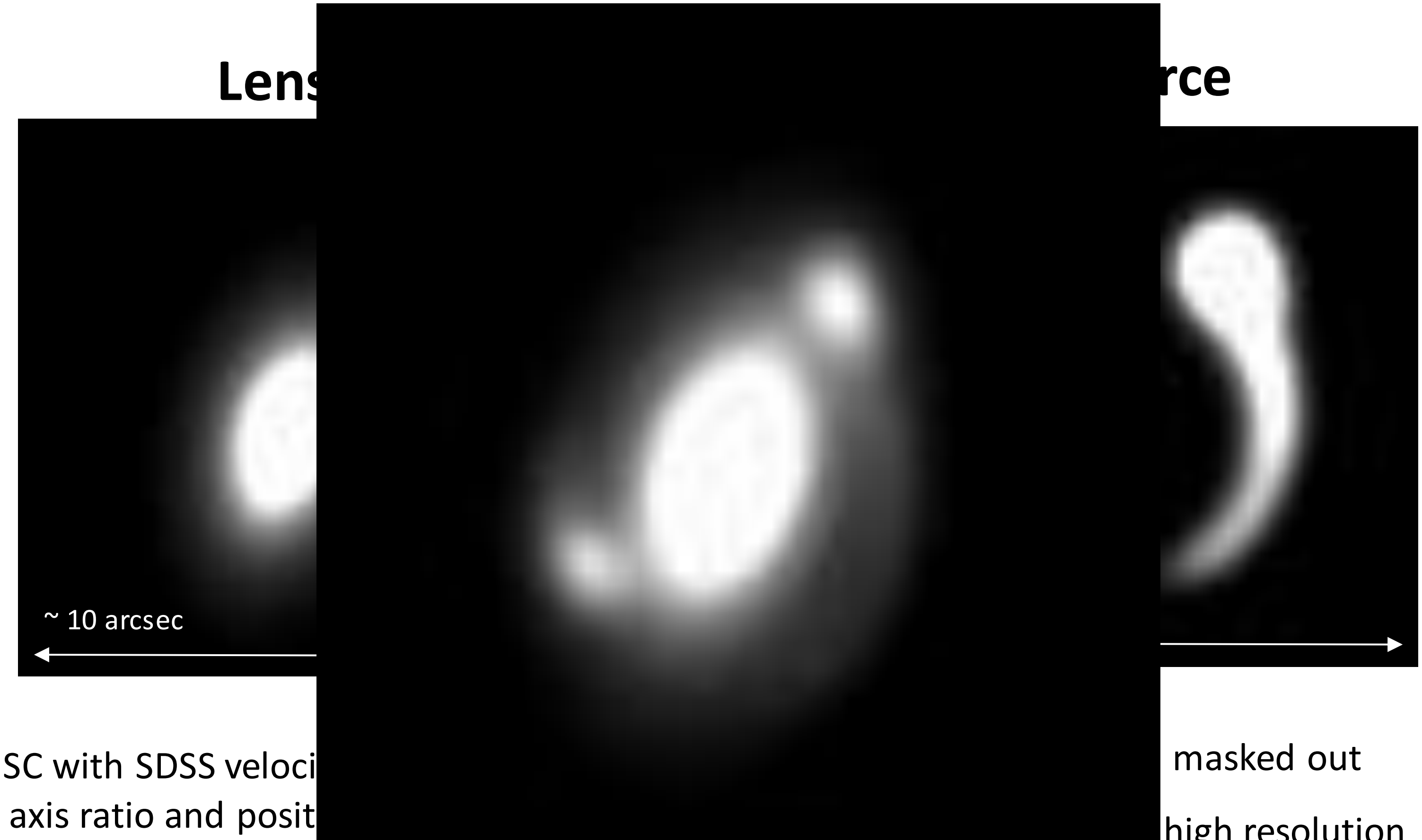
HUDF images, masked out

-> high redshift, high resolution
-> random picked galaxy for given lens and randomly located

mock up images

based on real observed galaxy images

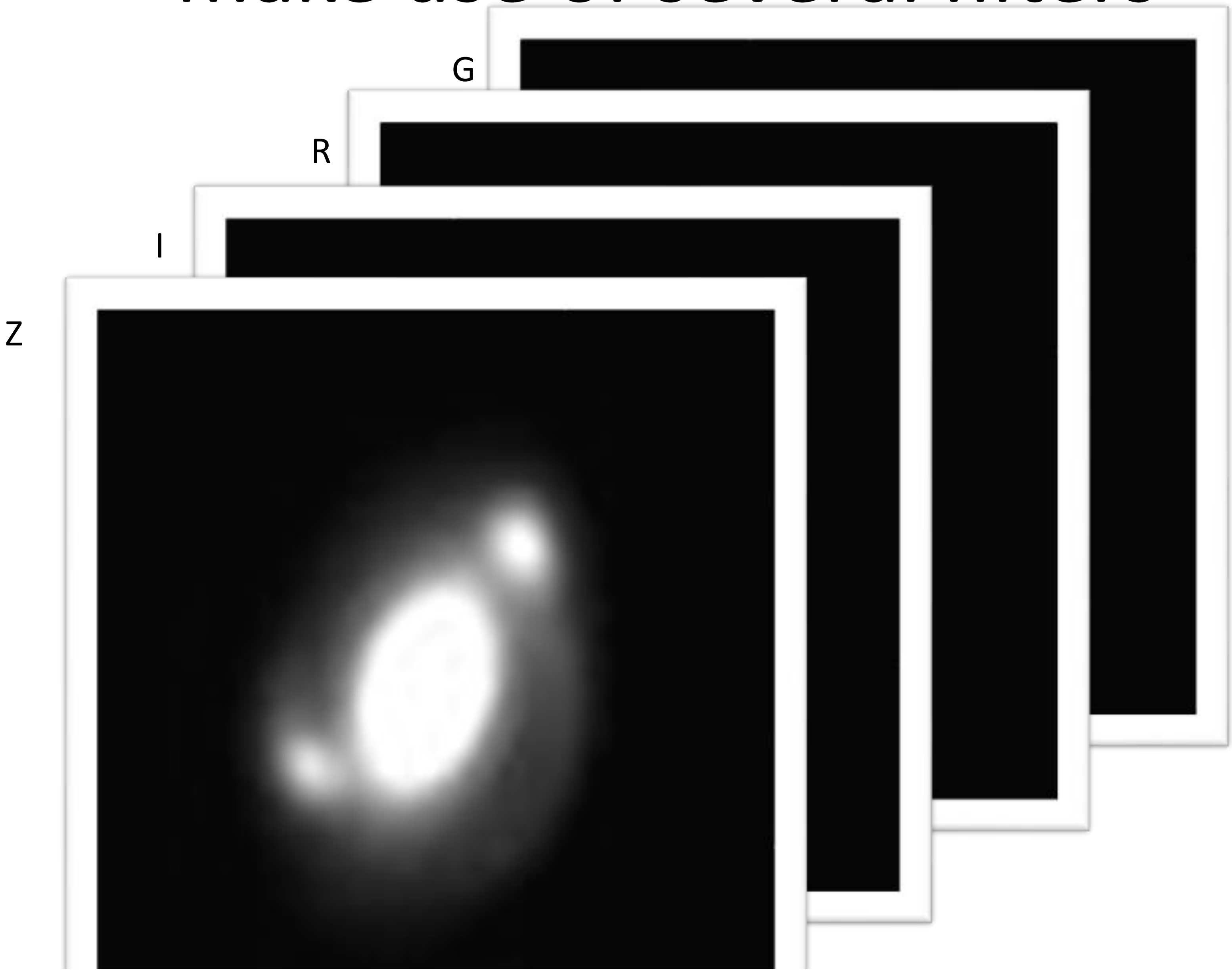
Lens **rce**



HSC with SDSS veloci
-> axis ratio and posit
-> with Gaussian spread for possible offset of mass from light distribution

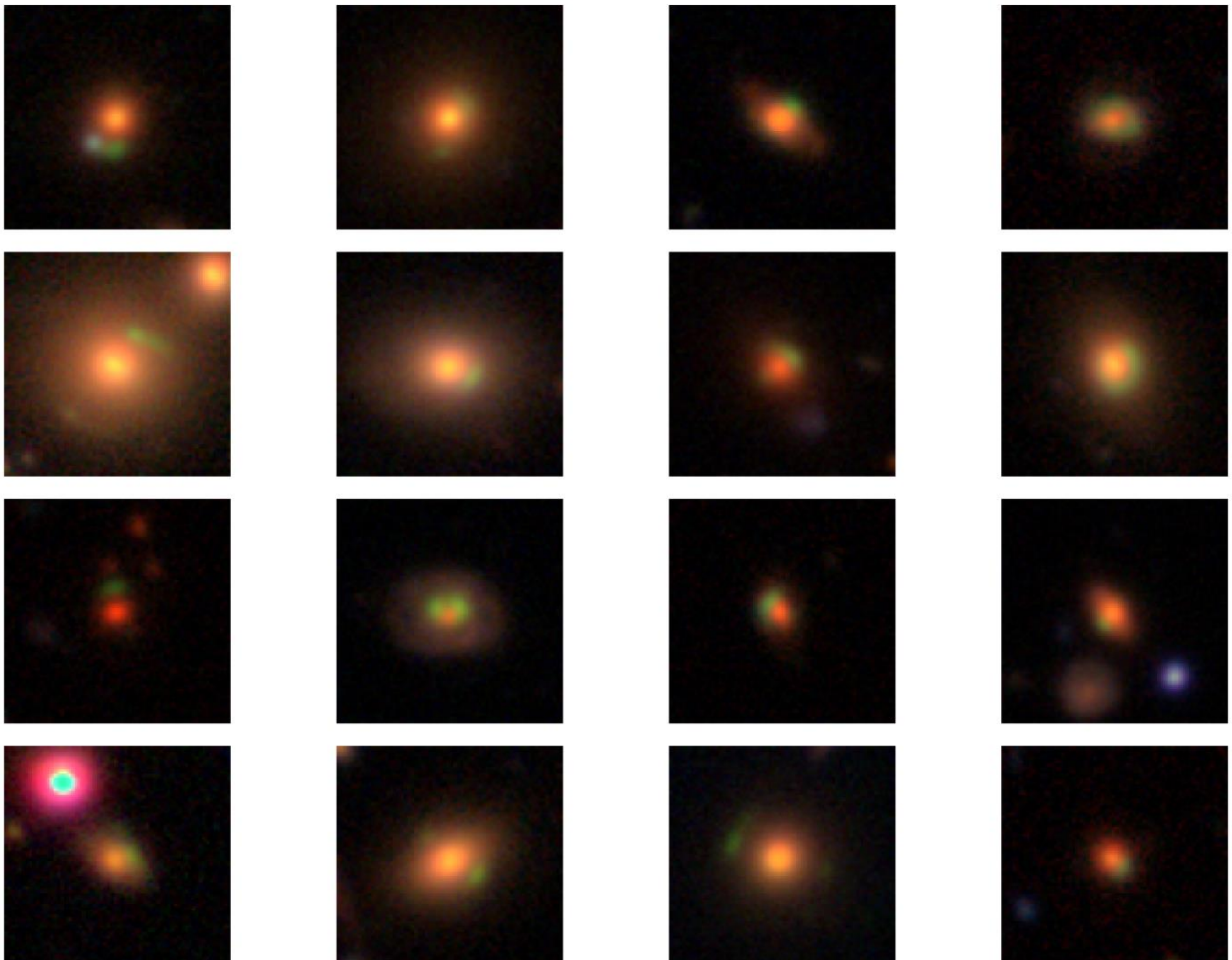
masked out
high resolution
-> random picked galaxy for given lens and randomly located

make use of several filters



Example colour images

Image size 10.8"x10.8", based on gri filters



Test different data sets

- per data set ~100,000 images, each 4 filters
- variations of Einstein radius distribution
  - naturally distributed, lower limit 0.5"

    -> peaks at 0.5"

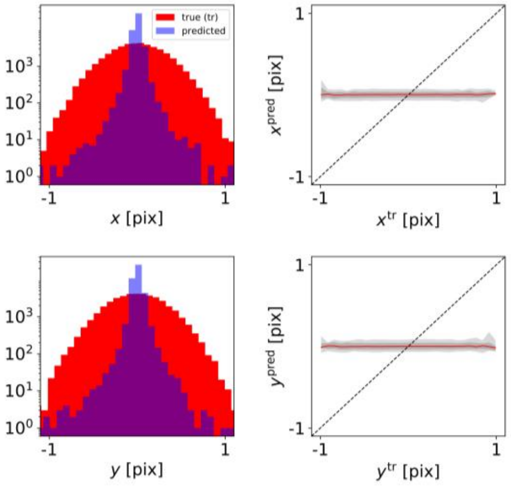
    -> performance drops for larger θ_E significantly
  - Naturally distributed, lower limit 2"
  - Equally distributed, lower limit 0.5"

    -> flat up to ~2"

Equally distributed sample

Quads + doubles

Lens center

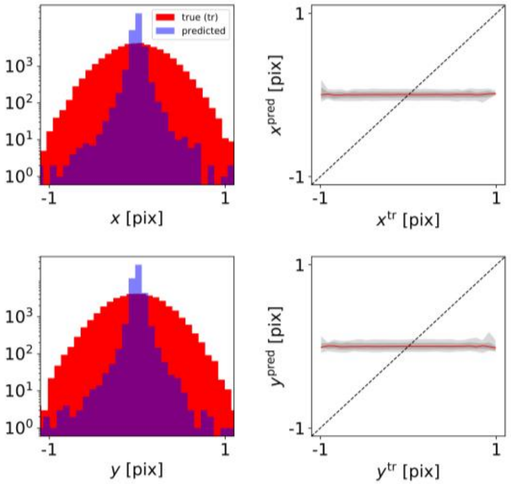


-> within pixels
-> kept for completeness

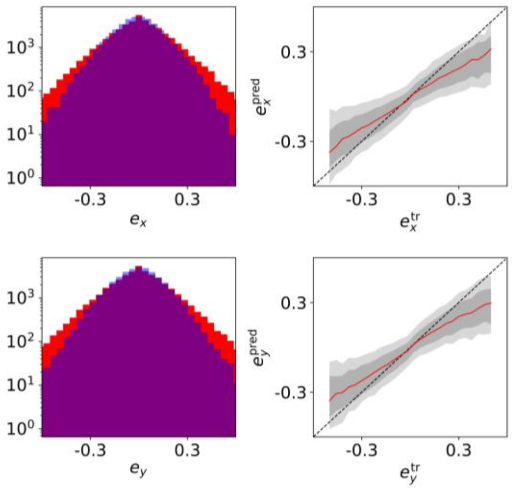
Equally distributed sample

Quads + doubles

Lens center



Complex ellipticity



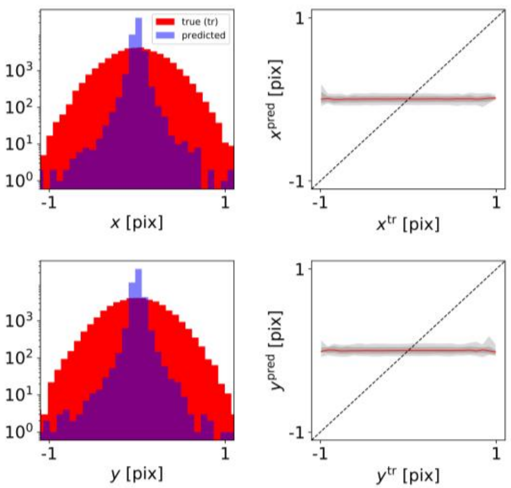
-> within pixels

-> kept for completeness

Equally distributed sample

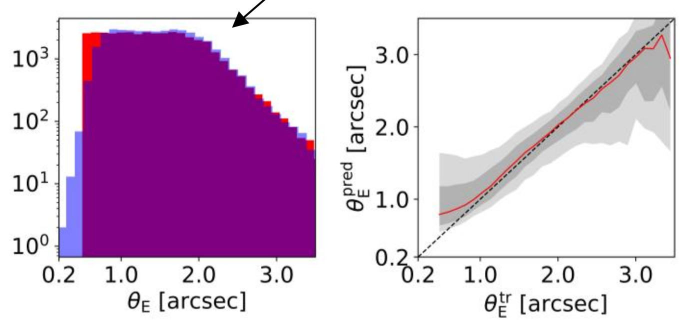
Quads + doubles

Lens center

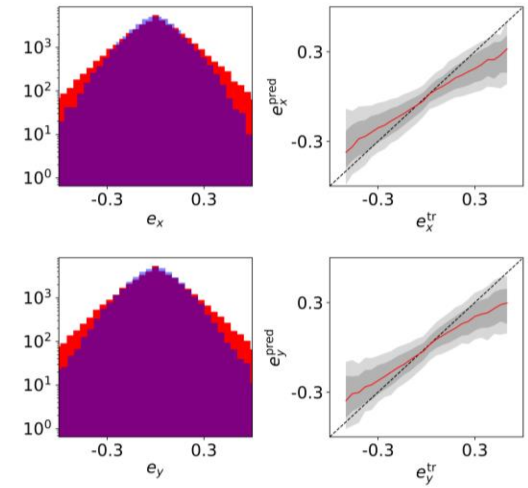


Einstein radius

Flat up to ~2"



Complex ellipticity

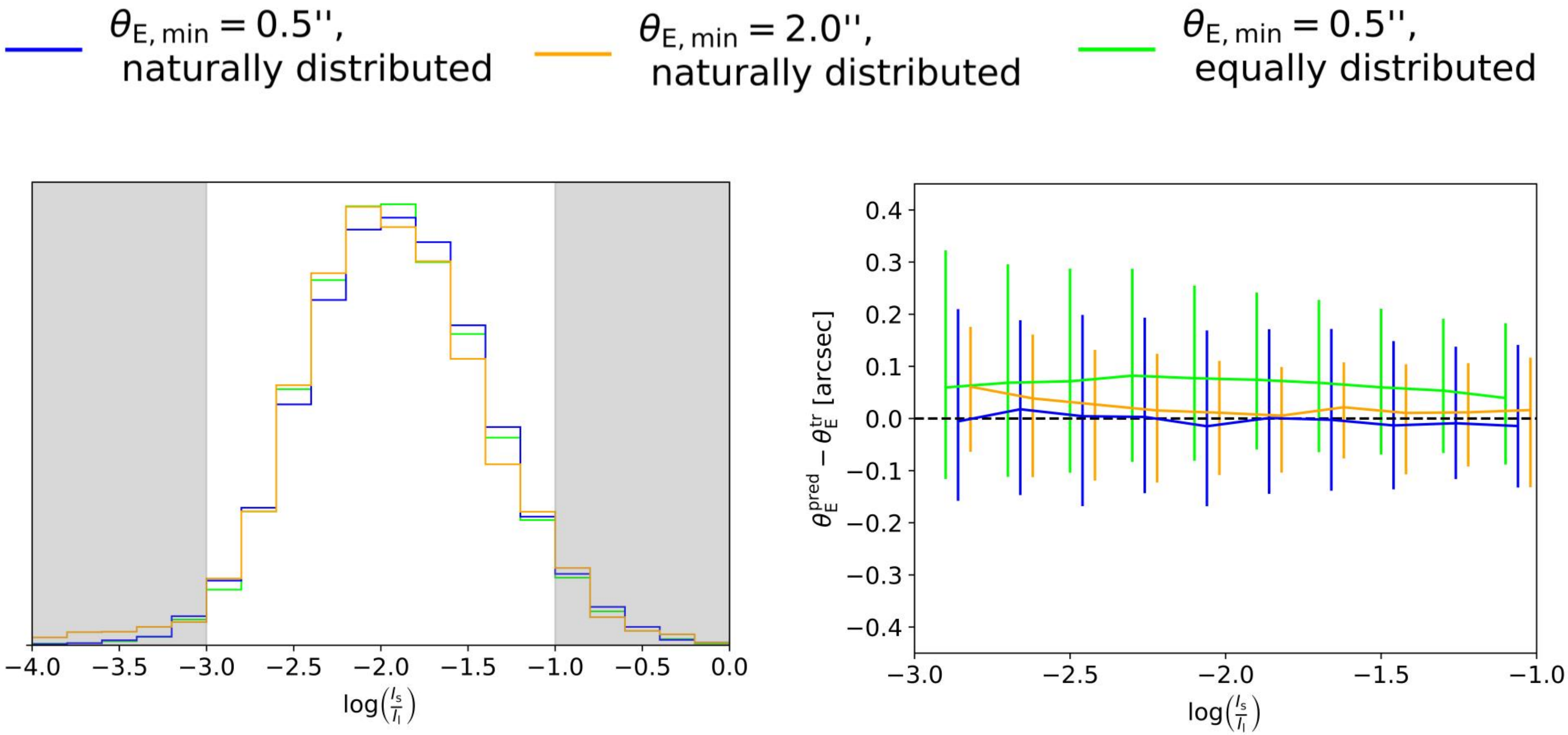


-> within pixels

-> kept for completeness

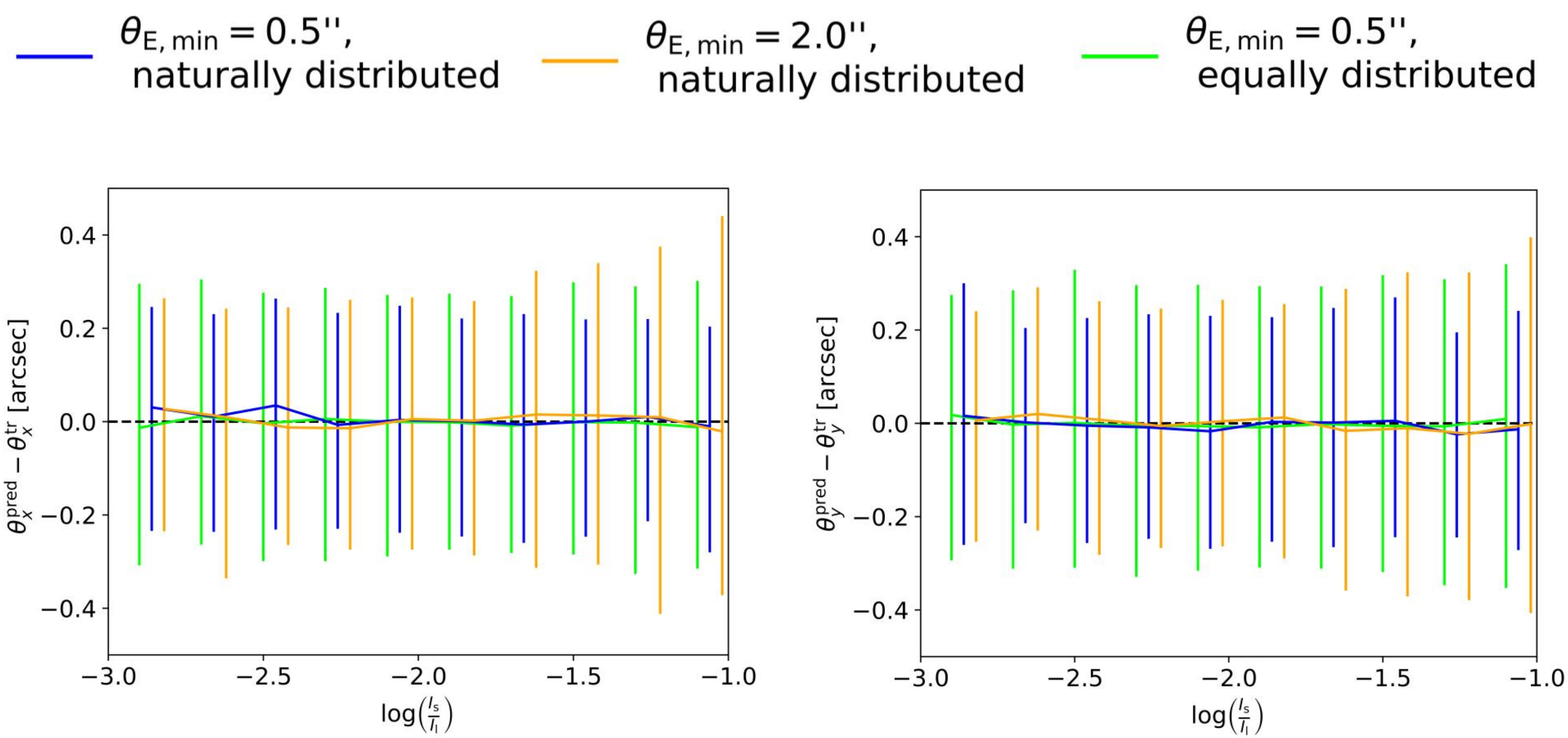
-> better for quads only

Performance on Einstein radius



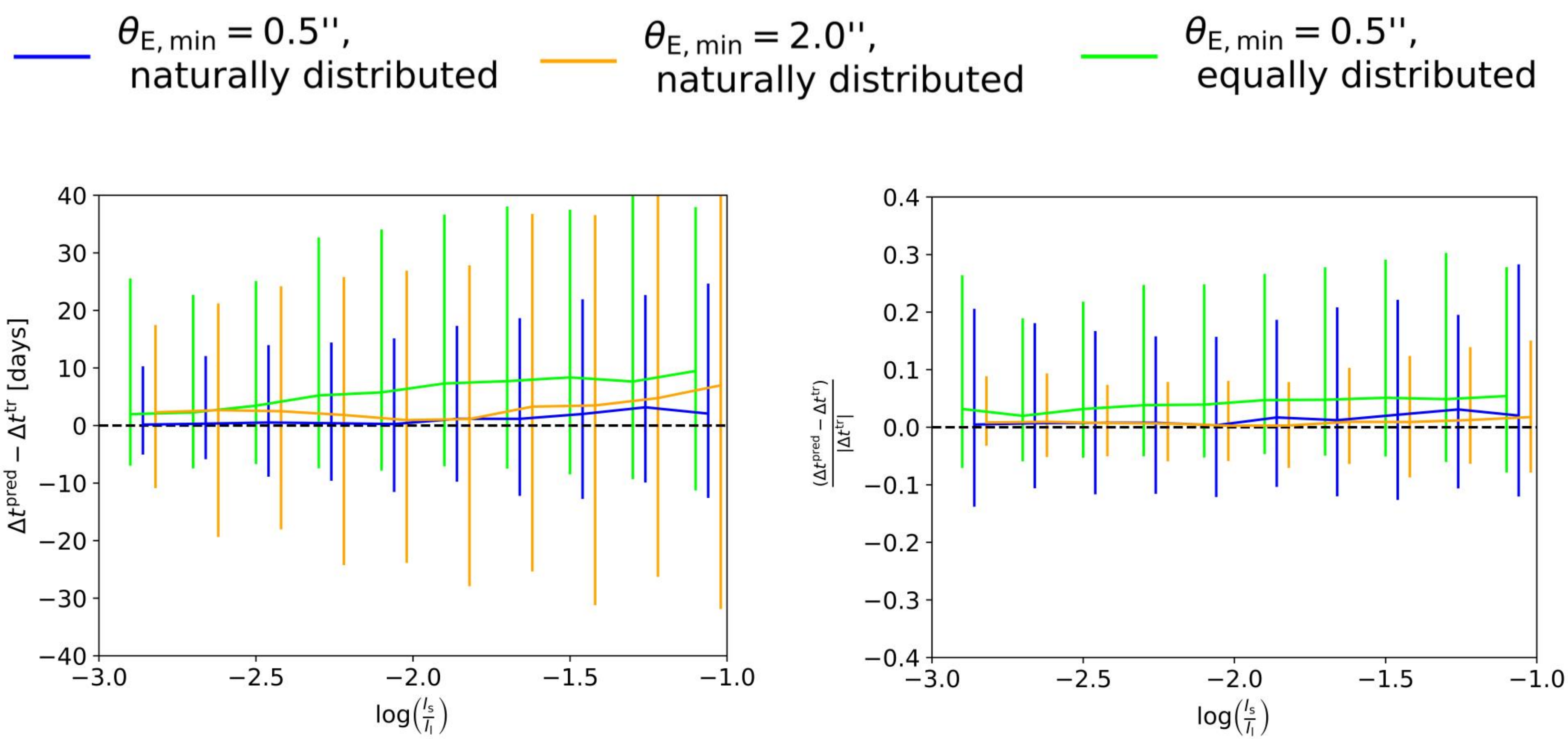
Performance on image position(s)

Use SIE mass model from CNN to predict image positions



Performance on time delay(s)

Use SIE mass model from CNN to predict time delay(s)



Summary and outlook

- Mock images based on real galaxies

  -> be as realistic as possible (galaxy structure, line-of-side objects)

  -> but limited in number
- Good regression network performance
- Distribution of Einstein radius is important
- CNN predictions good enough for image position and time delay predictions
  - > important for follow-up planning

data used for simulation

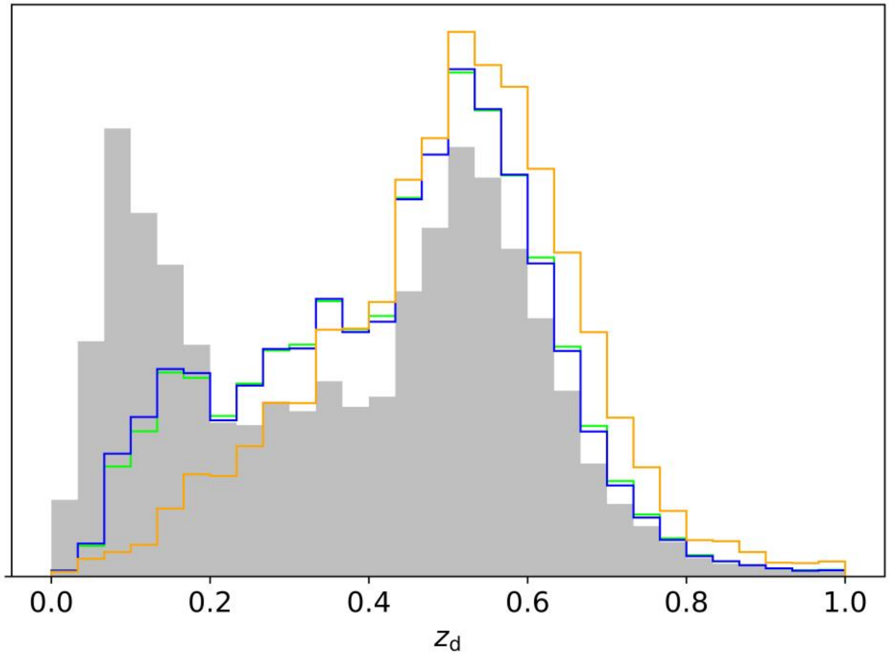
Lens

- Hyper Suprime-Cam (HSC)

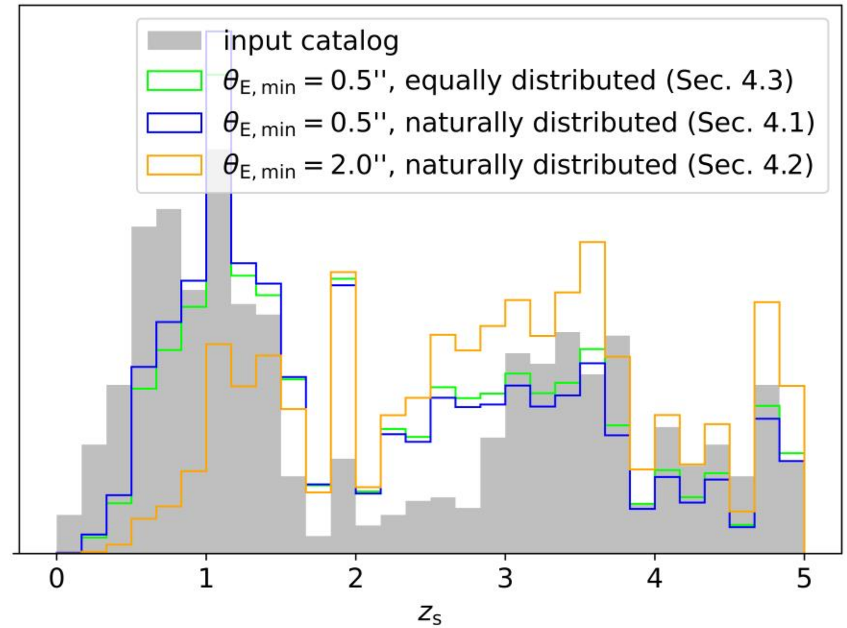
Source

- Hubble telescope (HUDF)

Redshift from SDSS



Redshift from Hubble

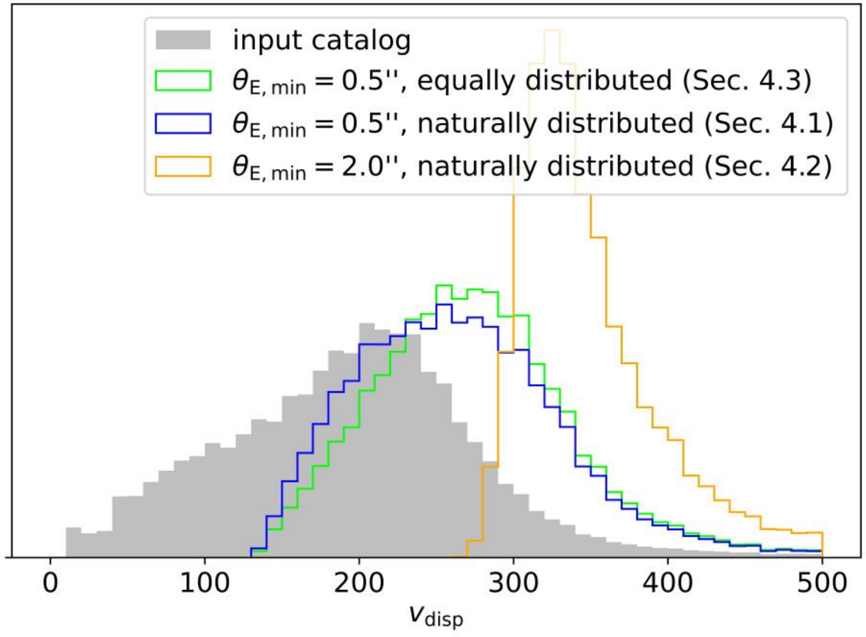


data used for simulation

Lens

- Hyper Suprime-Cam (HSC)

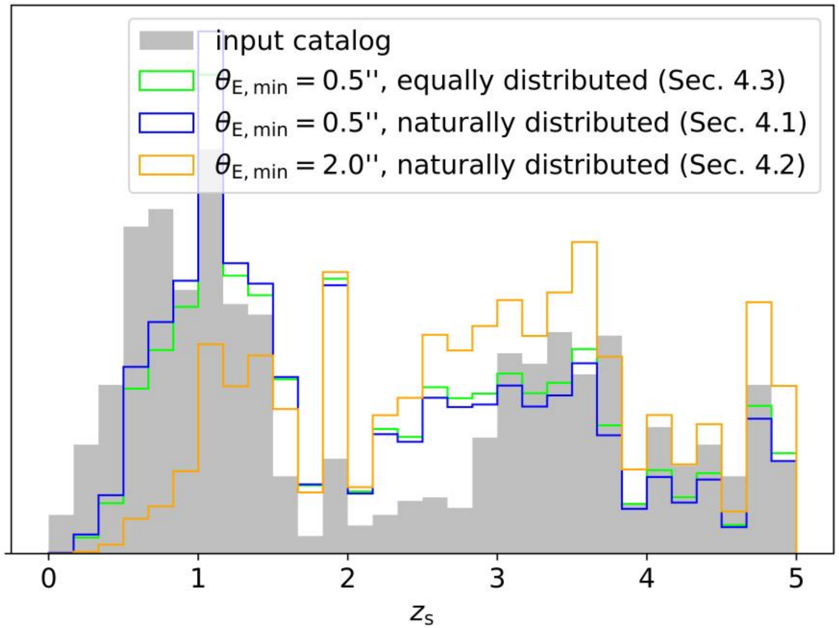
Velocity dispersion from SDSS



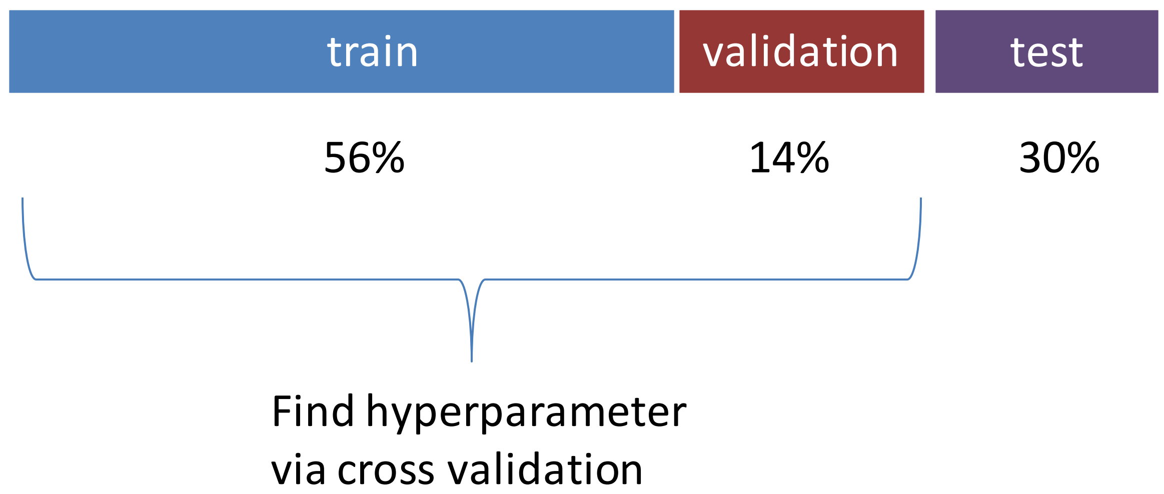
Source

- Hubble telescope (HUDF)

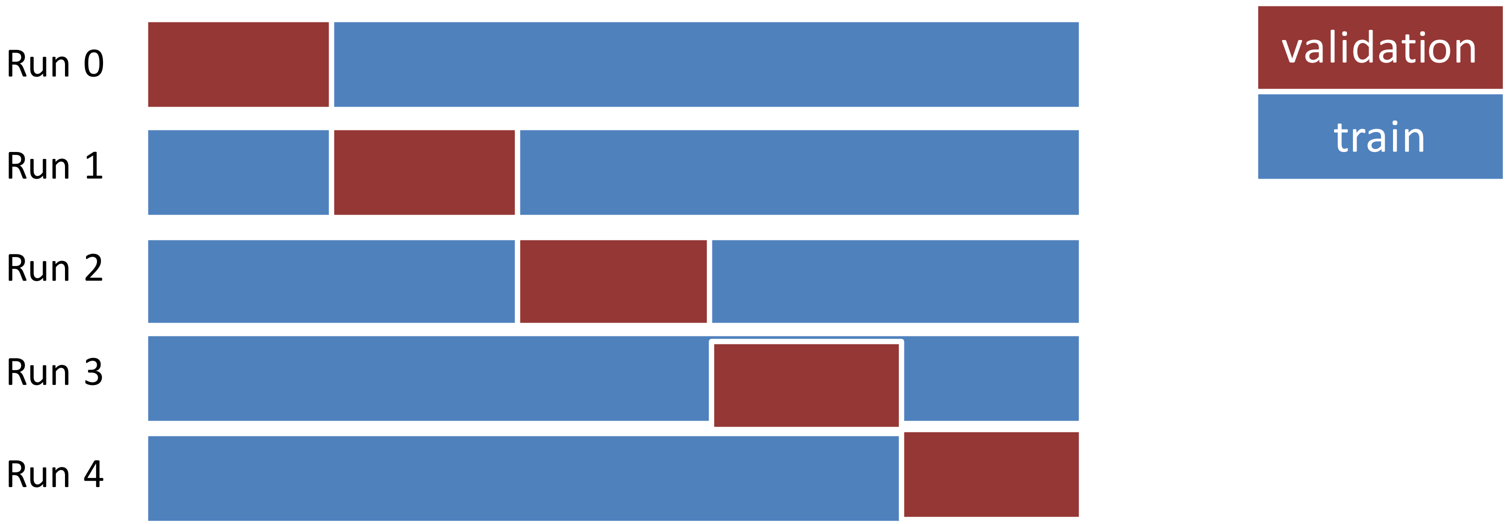
Redshift from Hubble



Data set splitting



Cross validation



- train 5 runs over 300 epochs
- best epoch to stop training:
  mean of all five losses is minimal

-> train final network up to this epoch using train and validation set

-> use test set to finally test the performance