TUTORIALS

- GO HERE AND INSTALL PYTHON CONDA ENVIRONMENT. SHOULD BE VERY STRAIGHTFORWARD.

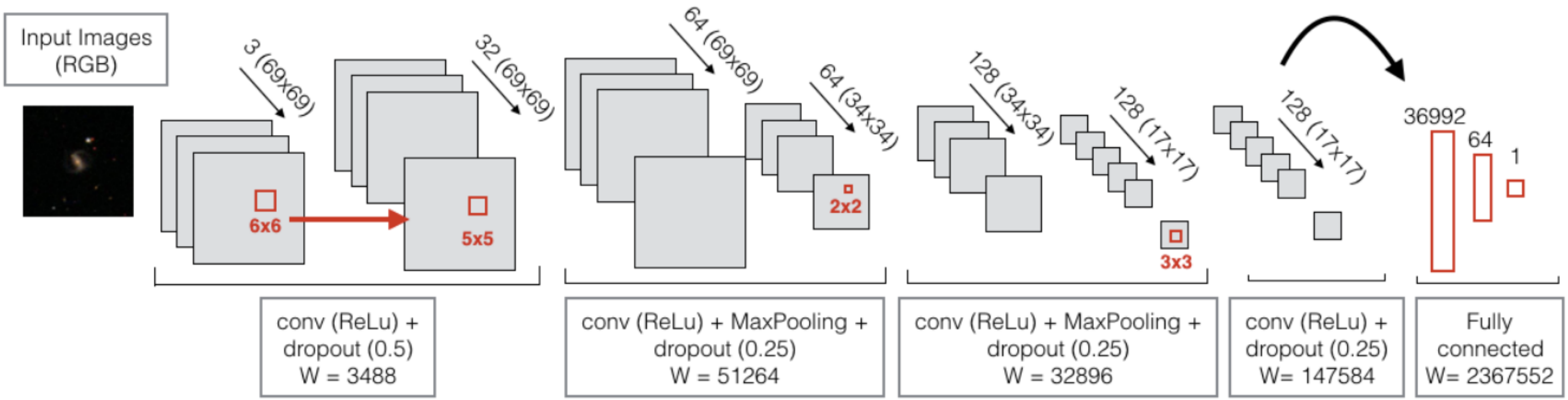
- DO NOT DOWNLOAD ANY DATA SINCE I AM CHANGING THE TUTORIALS TO SIMPLER ONES

  - MORPHOLOGY CLASSIFICATION WITH / WITHOUT DEEP LEARNING

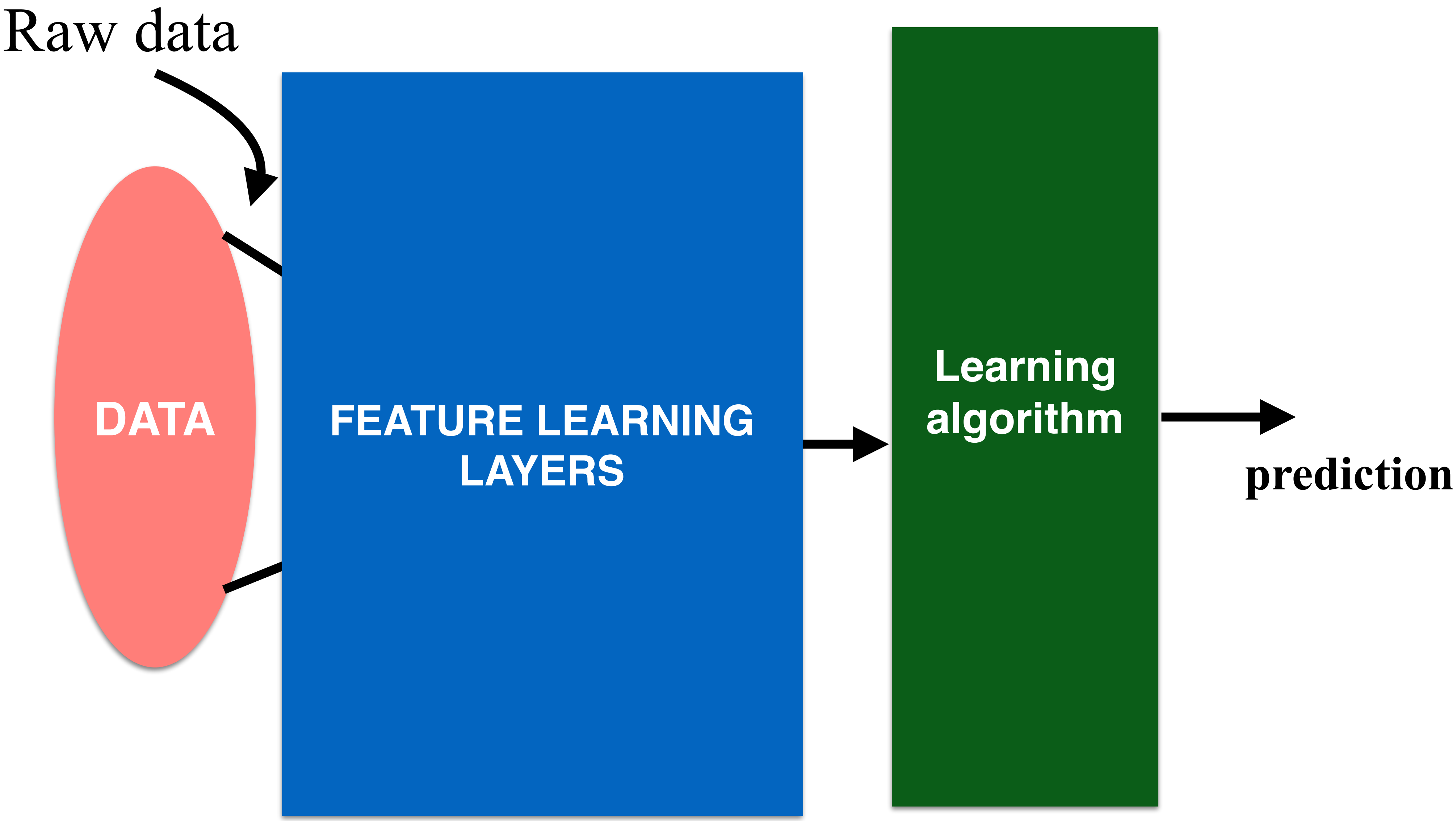
  - A BIT OF VAEs…MAYBE

PART IV: IMAGE 2 IMAGE NETWORKS + INTRODUCTION TO GENERATIVE MODELS

EXAMPLE OF VERY SIMPLE CNN

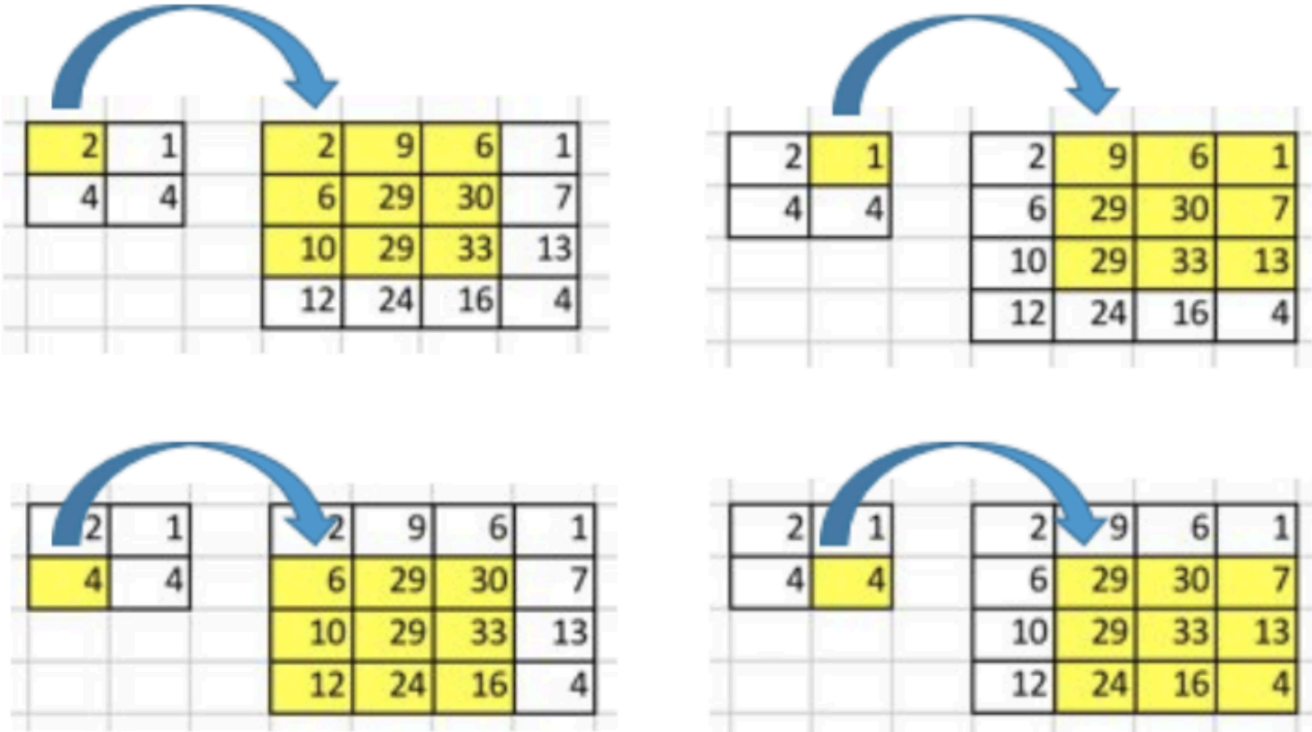


Dominguez-Sanchez+18


Raw data
DATA
FEATURE LEARNING LAYERS
Learning algorithm
prediction

TRANSPOSED CONVOLUTION

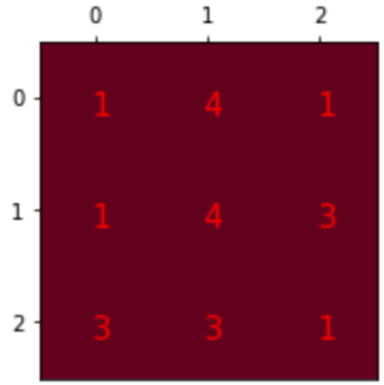
ALLOWS TO INCREASE THE SIZE



Going Backward of Convolution

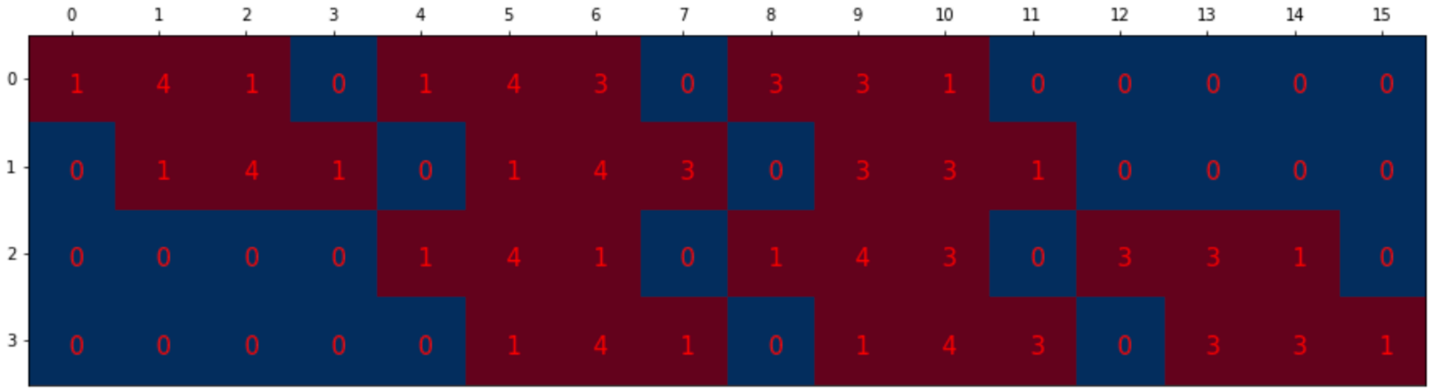
EXAMPLE TAKEN FROM HERE

CONVOLUTION MATRIX



Kernel (3, 3)

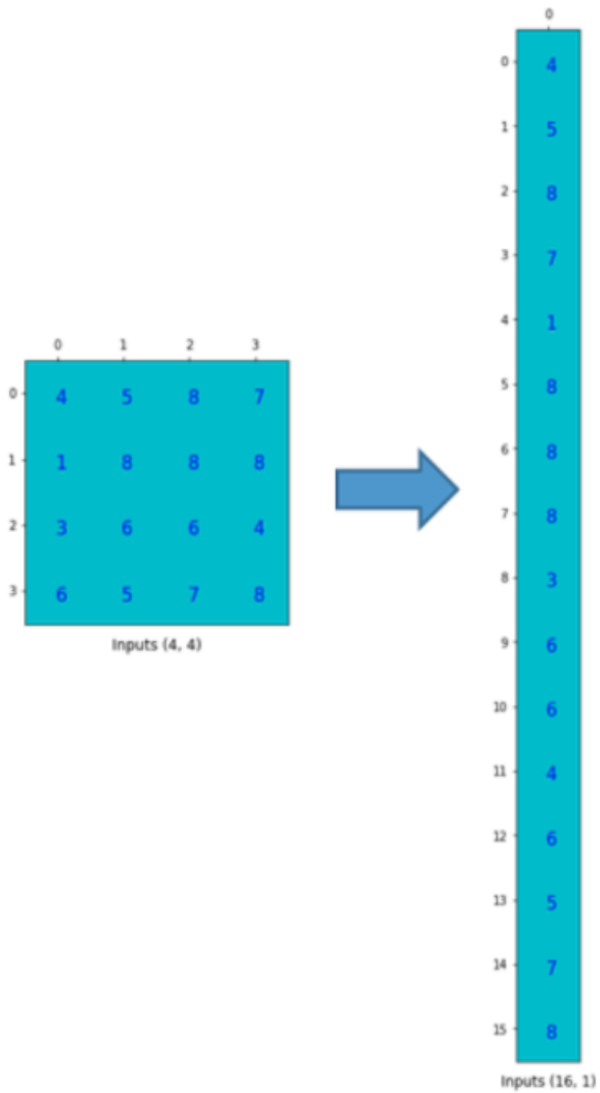
THE KERNEL CAN BE ARRANGED IN FORM OF A MATRIX:



Convolution Matrix (4, 16)

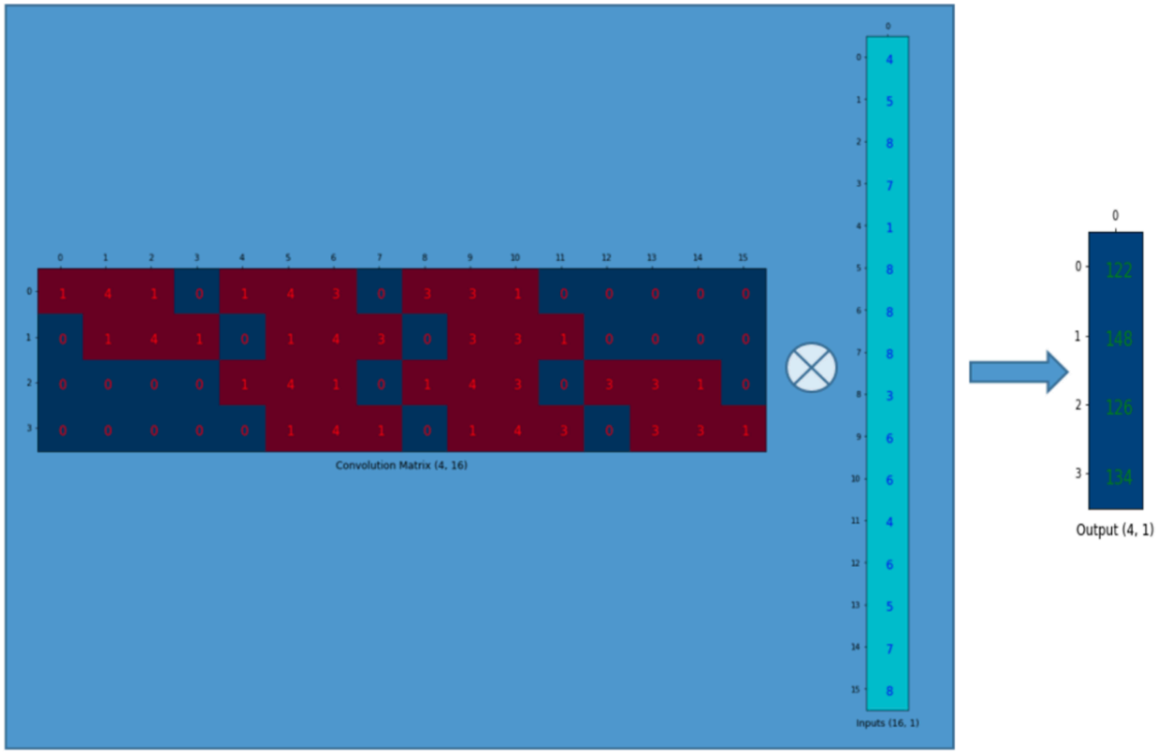
<u>EXAMPLE TAKEN FROM HERE</u>

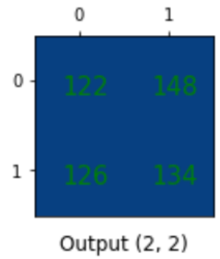
THE INPUT IS FLATTENED INTO A COLUMN VECTOR



EXAMPLE TAKEN FROM HERE

THE CONVOLUTION IS TRANSFORMED INTO A PRODUCT OF MATRICES

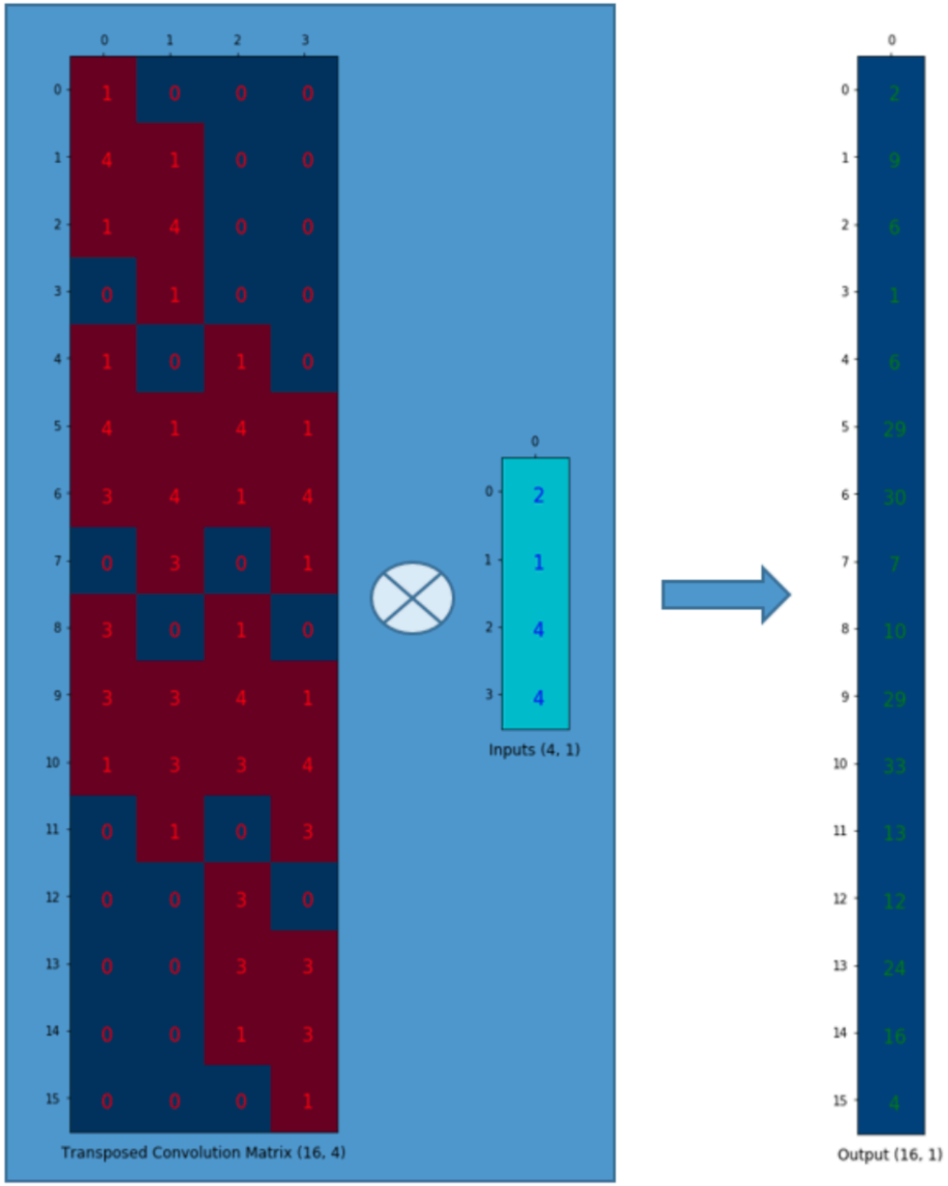


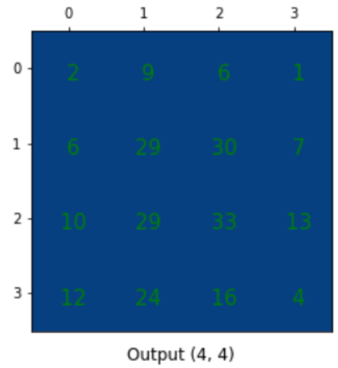


RESHAPHED OUTPUT

EXAMPLE TAKEN FROM HERE

THE TRANSPOSED CONVOLUTION IS THE INVERSE OPERATION

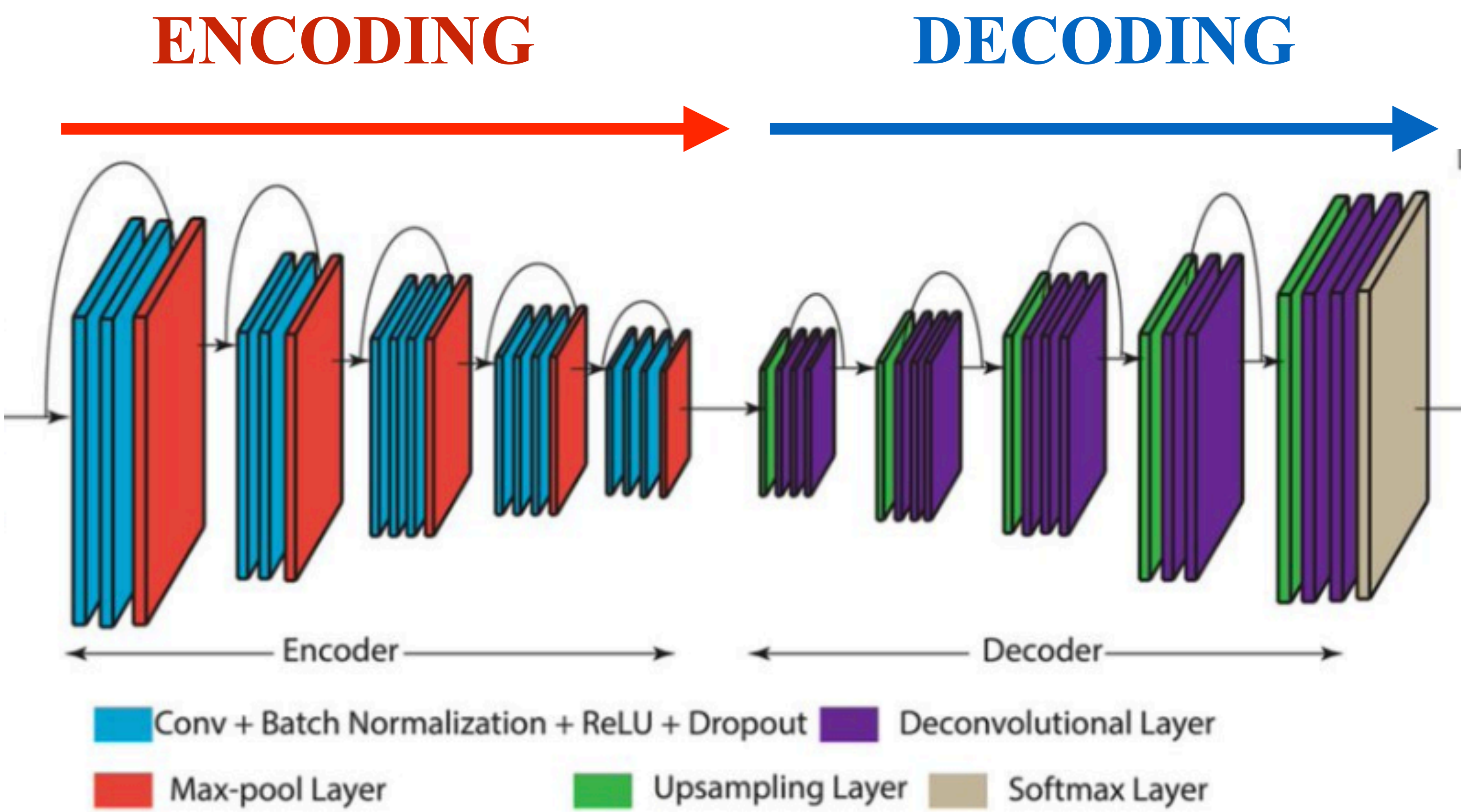




RESHAPED OUTPUT

EXAMPLE TAKEN FROM HERE

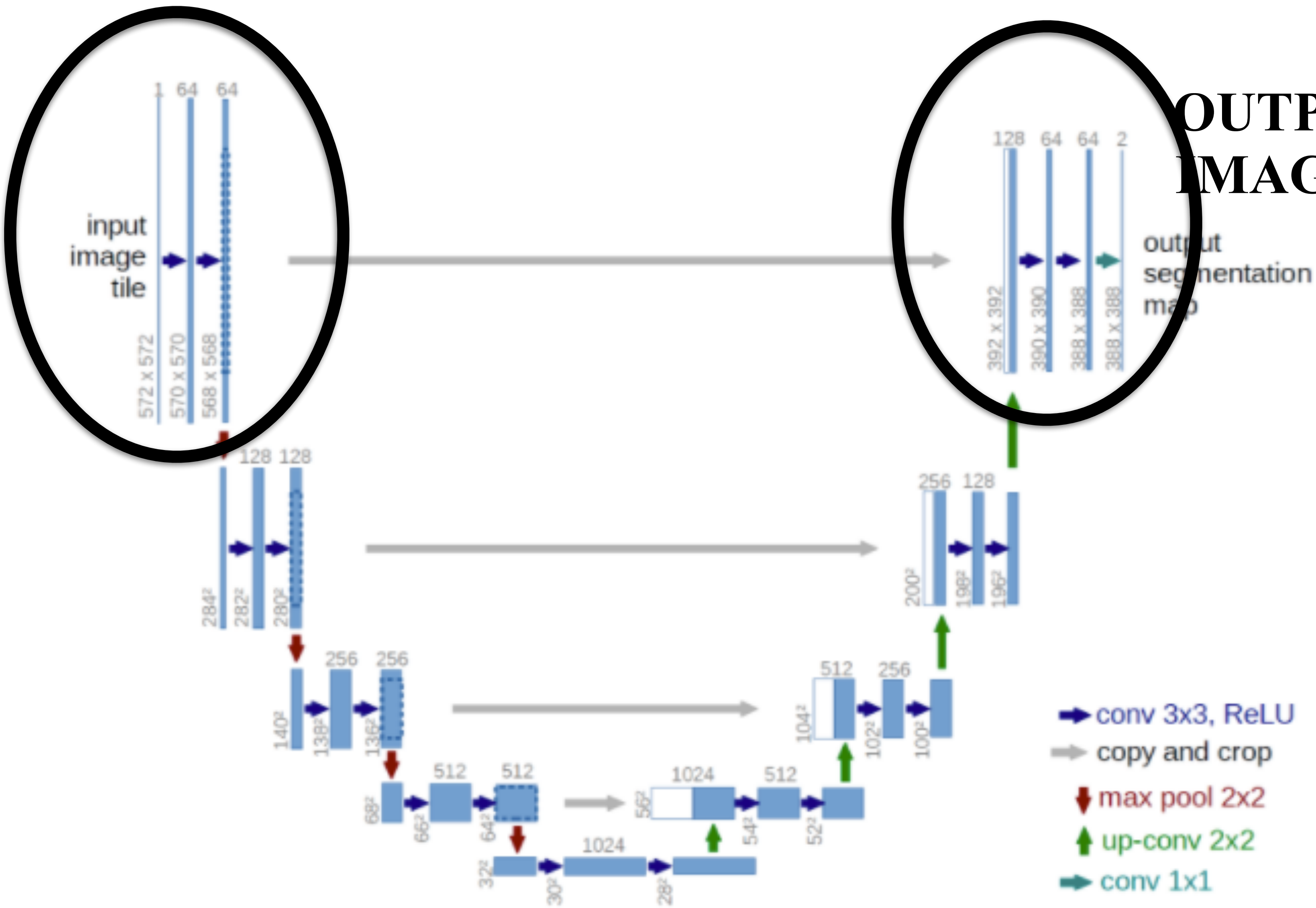
DECODER-ENCODER NETWORKS



ENCODING-DECODING TO EXTRACT IMAGE FEATURES: U-NET

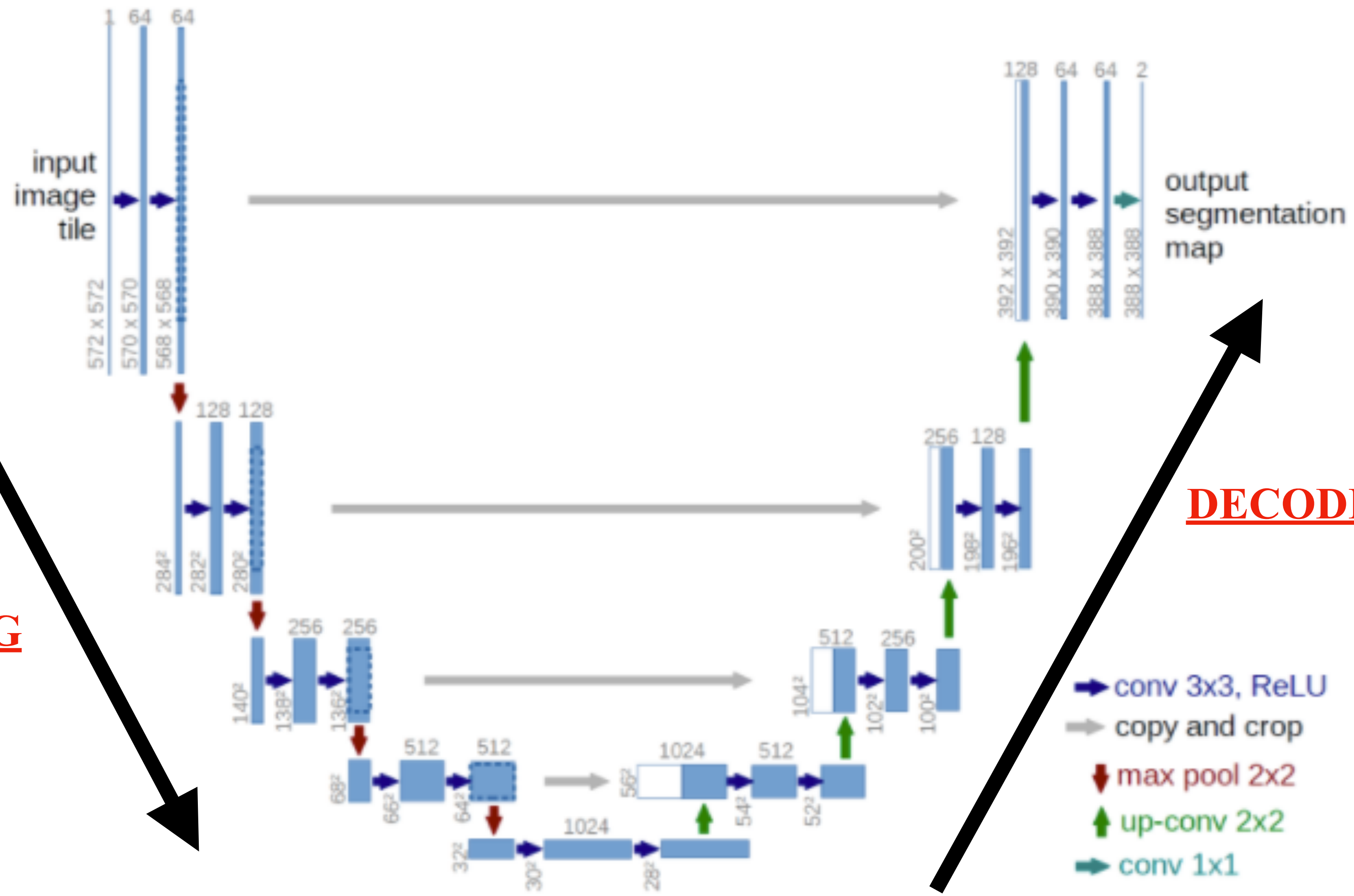
INPUT IMAGES

OUTPUT IMAGES



Ronnerberger+15

ENCODING-DECODING TO EXTRACT IMAGE FEATURES: U-NET

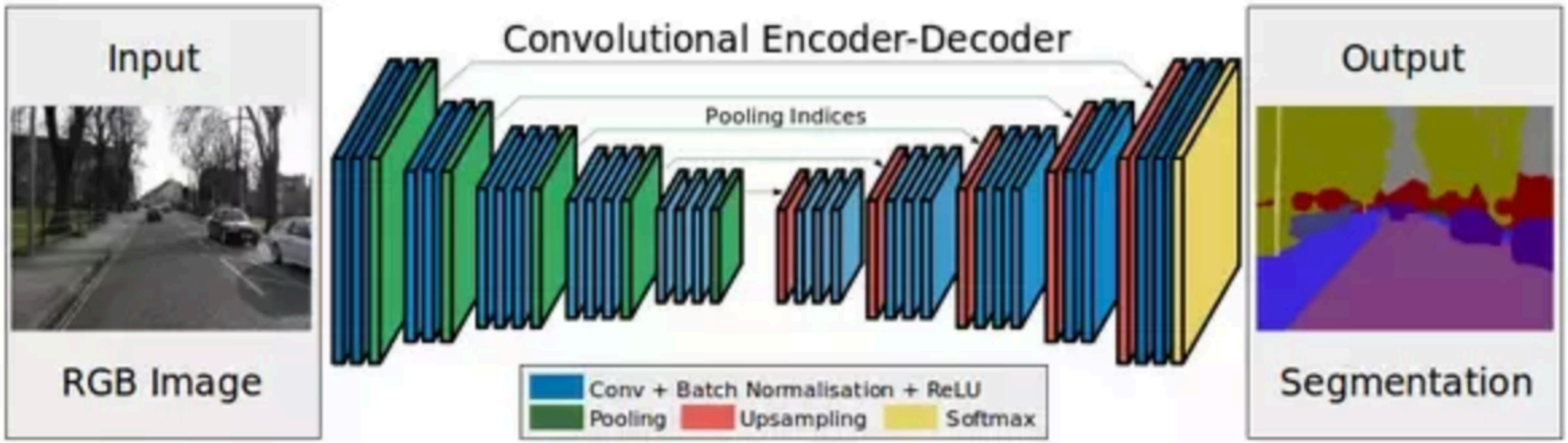


ENCODING

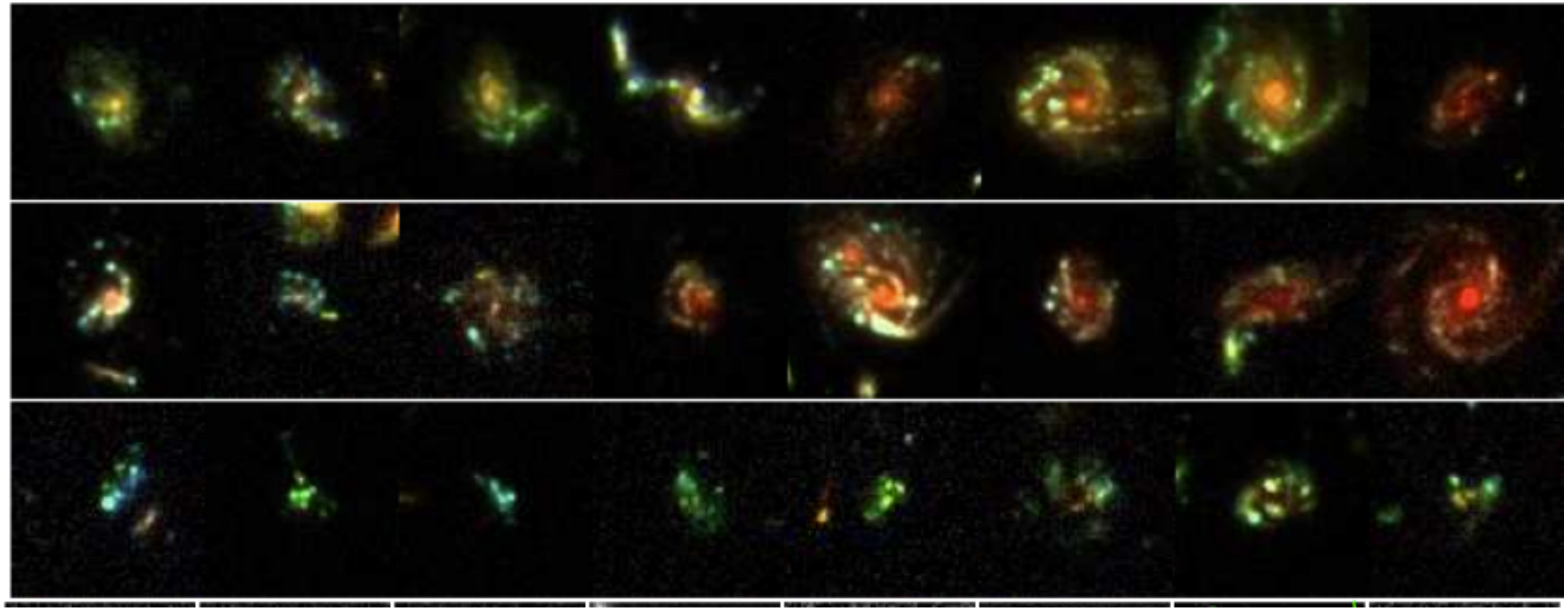
DECODING

Ronnerberger+15

MAIN APPLICATION IS IMAGE SEGMENTATION



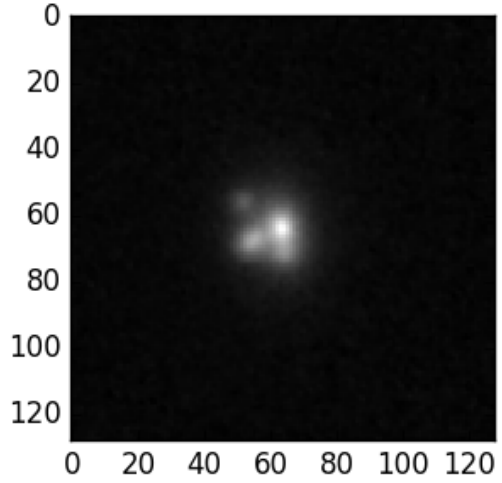
CLUMP DETECTION



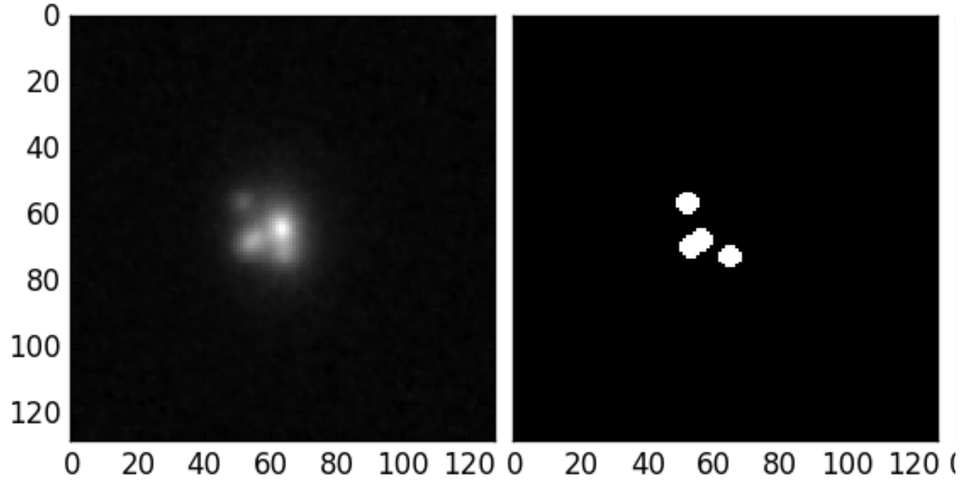
Guo+15,18

HIGH REDSHIFT GALAXIES PRESENT CLUMPS - THEIR ROLE IN BULGE FORMATION IS DEBATED

VERY SIMPLE SERSIC ANALYTIC SIMULATIONS + UNRESOLVED CLUMPS

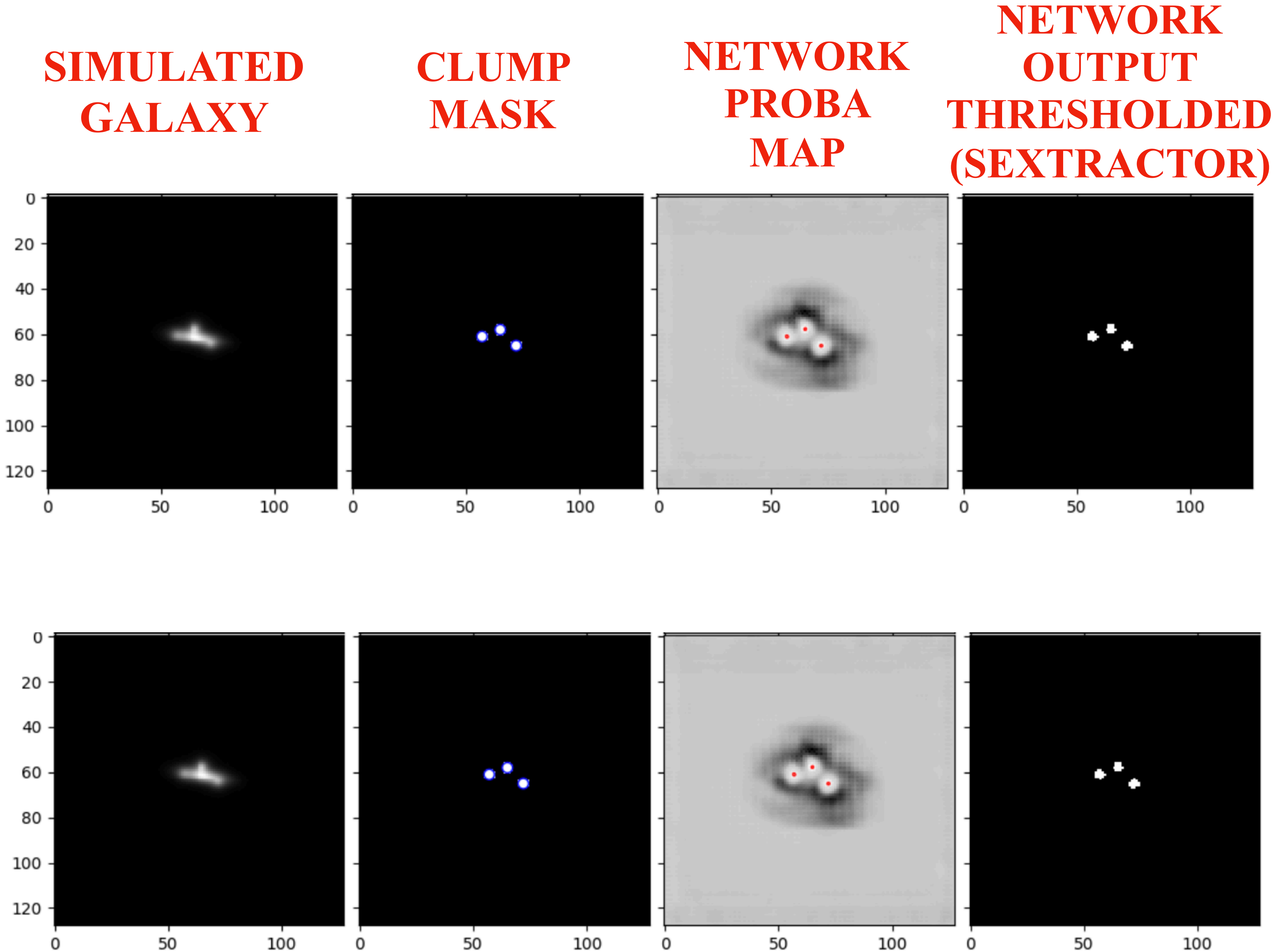


VERY SIMPLE SERSIC ANALYTIC SIMULATIONS + UNRESOLVED CLUMPS



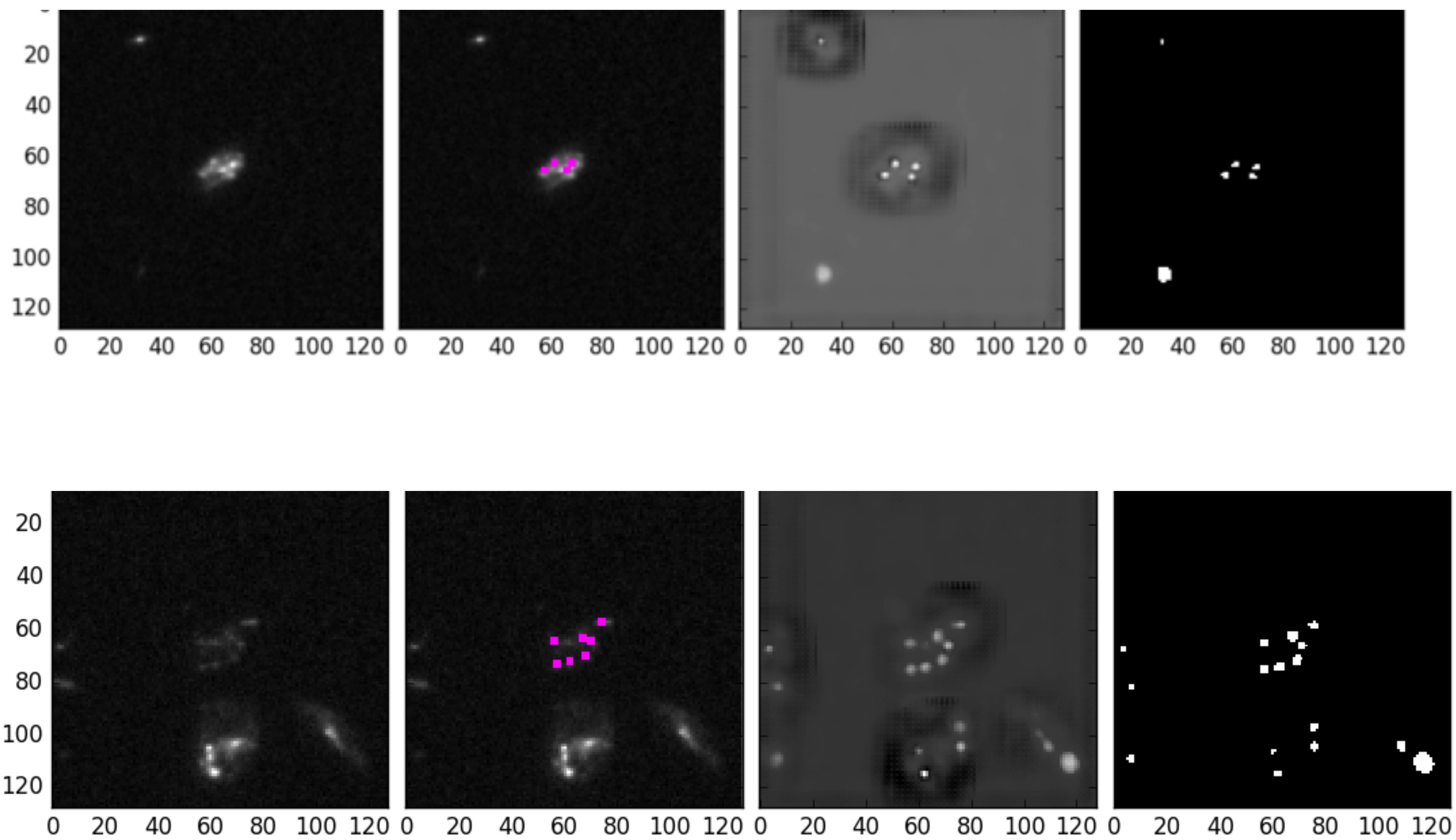
CLUMP POSITION IS KNOWN

LEE, MHC, PRIMACK, GUO+



LEE, MHC, PRIMACK+ IN PREP

SEEMS TO WORK REASONABLY WELL ON REAL OBSERVATIONS

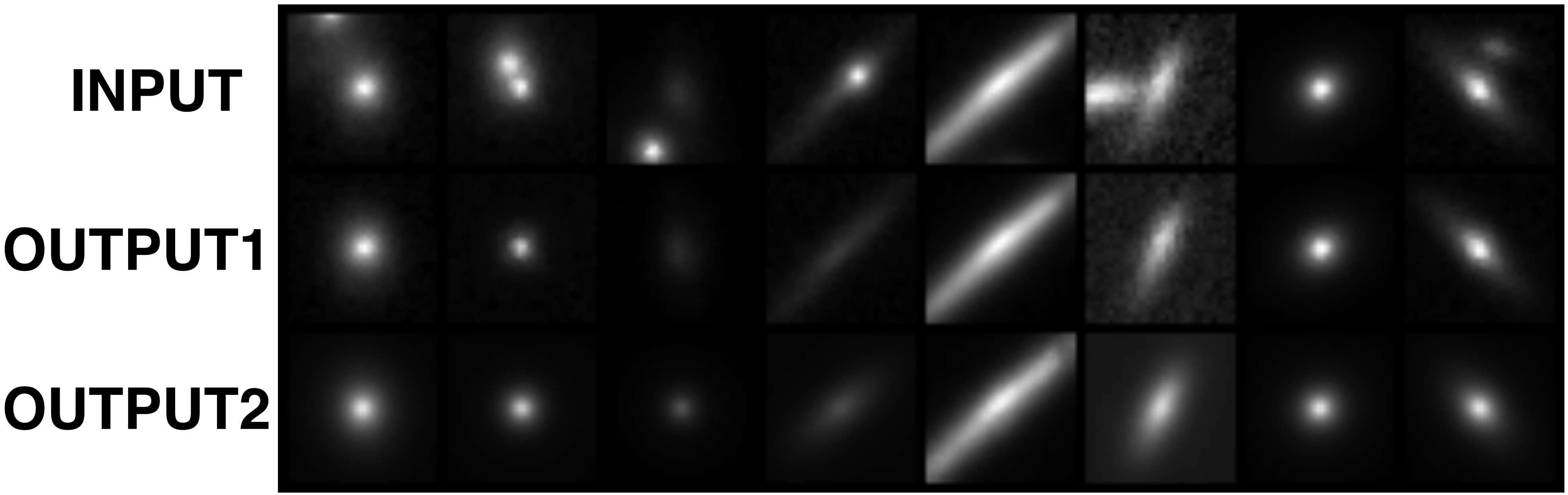


OBSERVED GALAXY

CLUMPS DETECTED BY GUO+

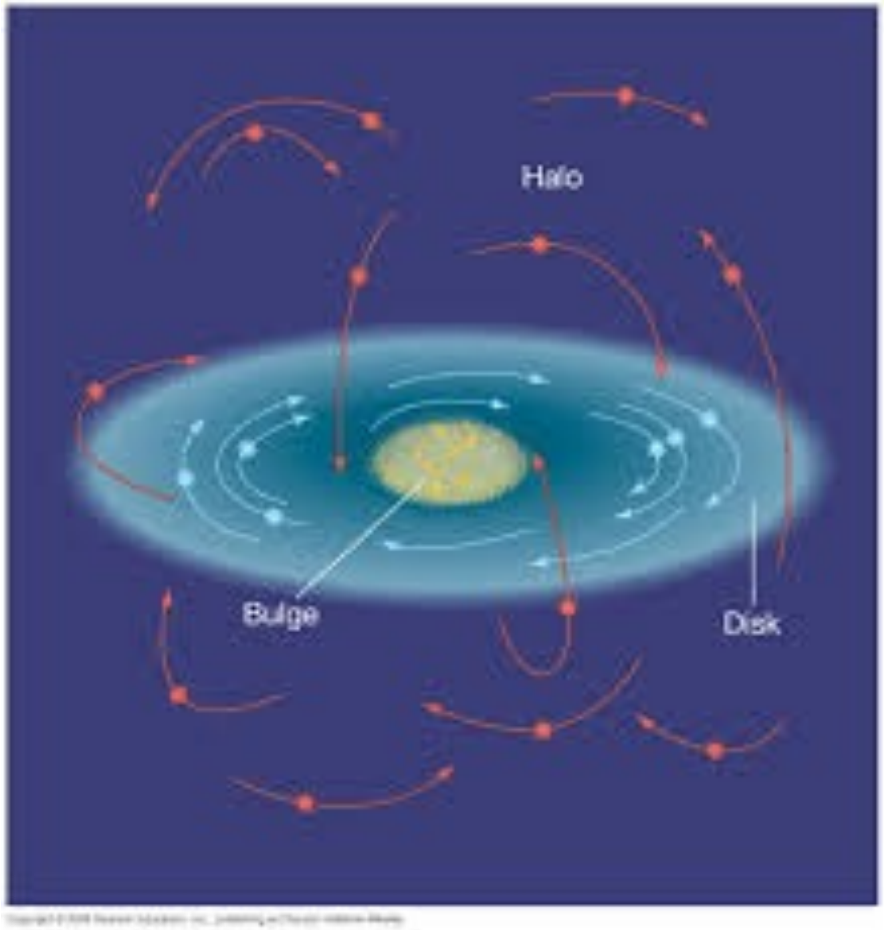
U-NET OUTPUTS

U-NET FOR GALAXY DEBLENDING

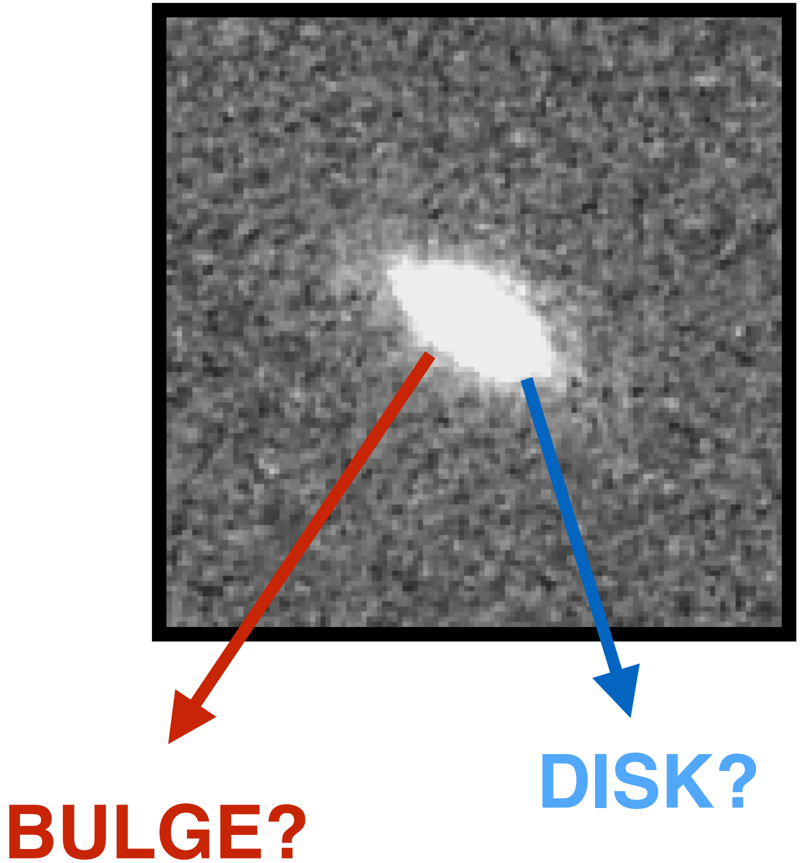


BOUCAUD, MHC+19

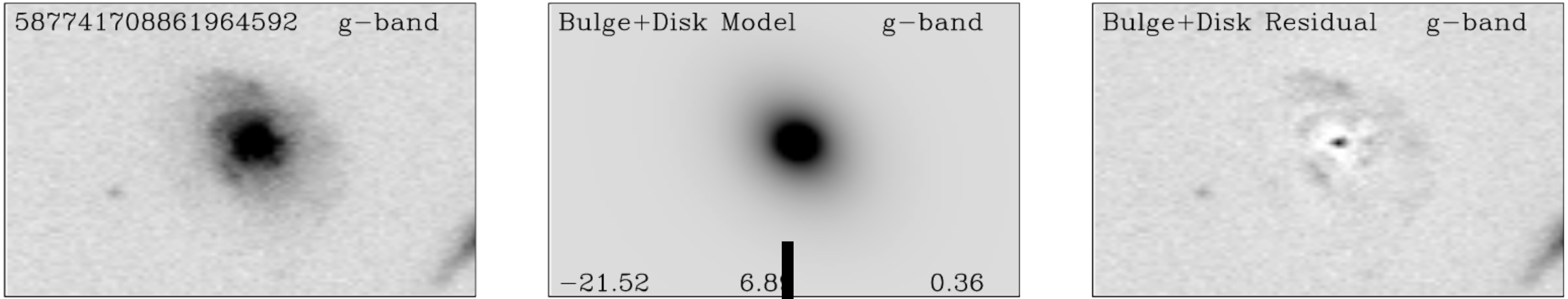
GALAXIES HAVE TWO COMPONENTS WITH DIFFERENT ASSEMBLY HISTORIES



OBSERVED GALAXY

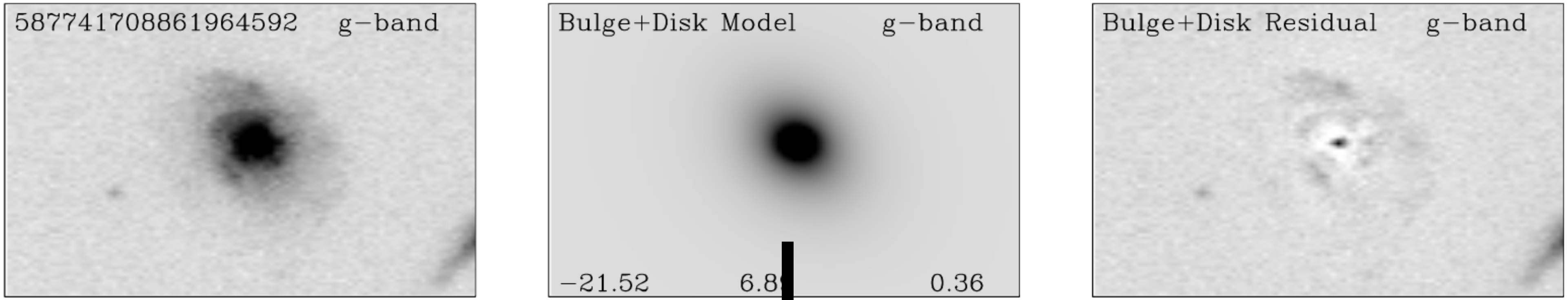


STANDARD MODEL FITTING



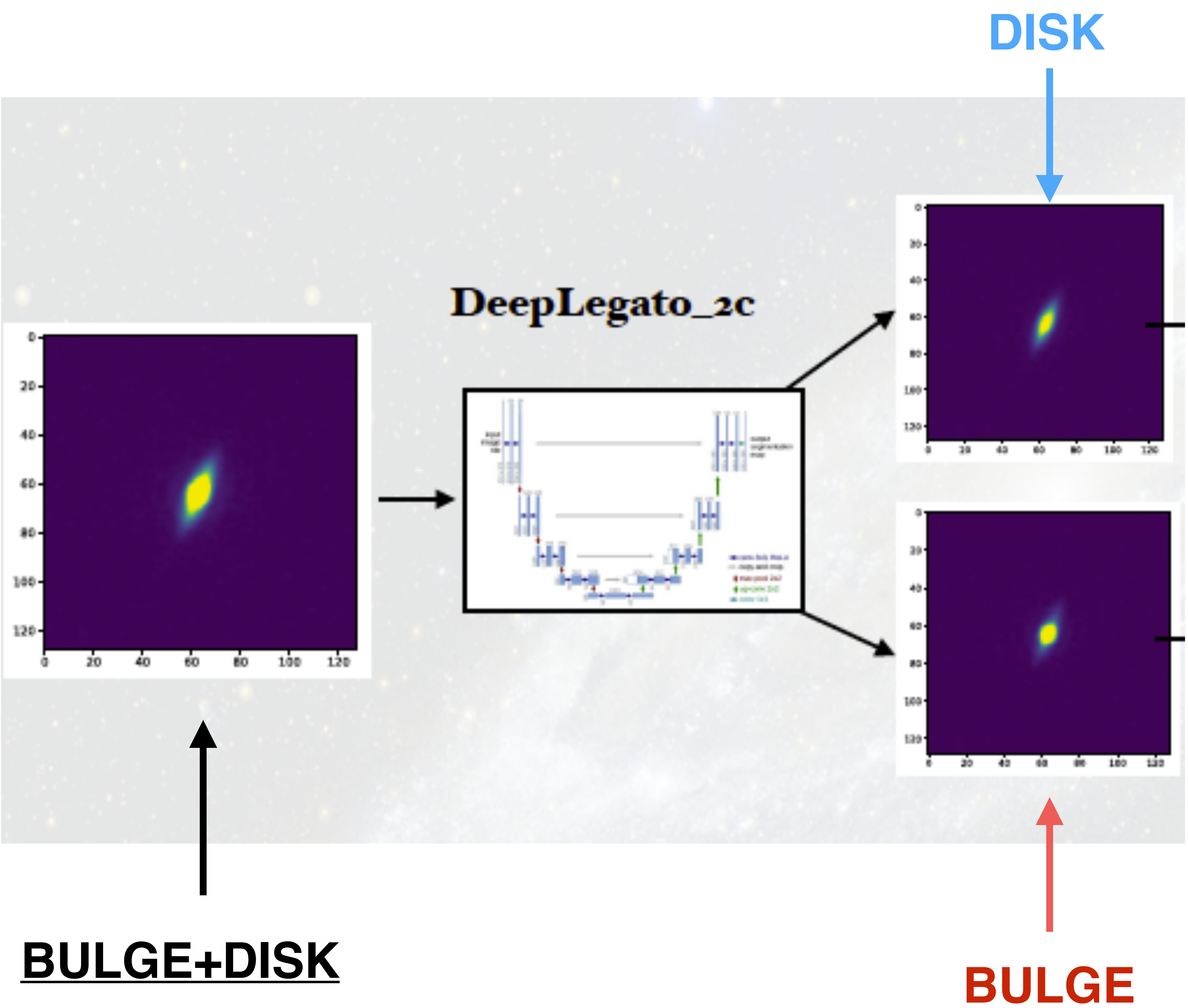
[10 parameter model]

STANDARD MODEL FITTING

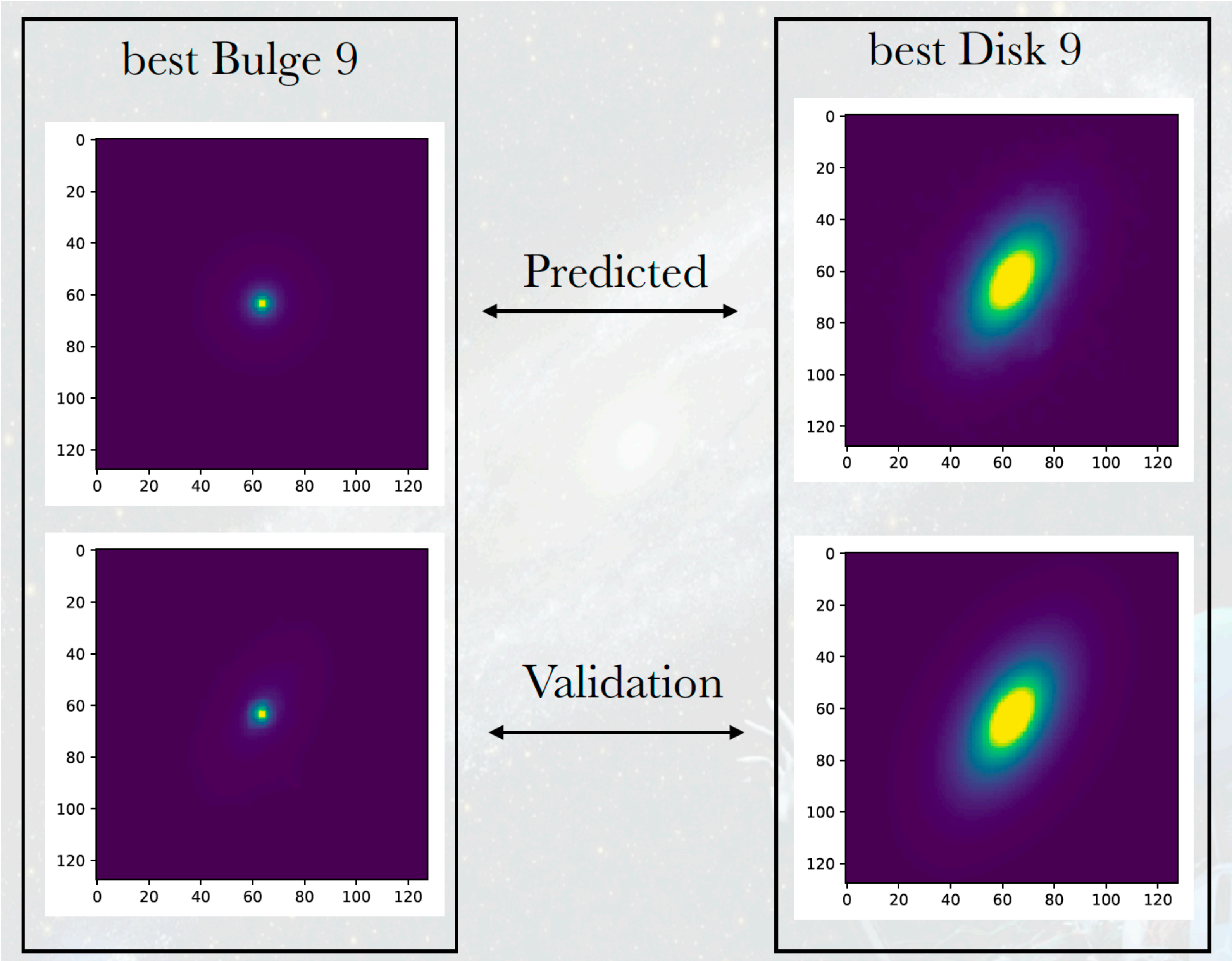


[10 parameter model]

LARGE AMOUNT OF DEGENERACIES



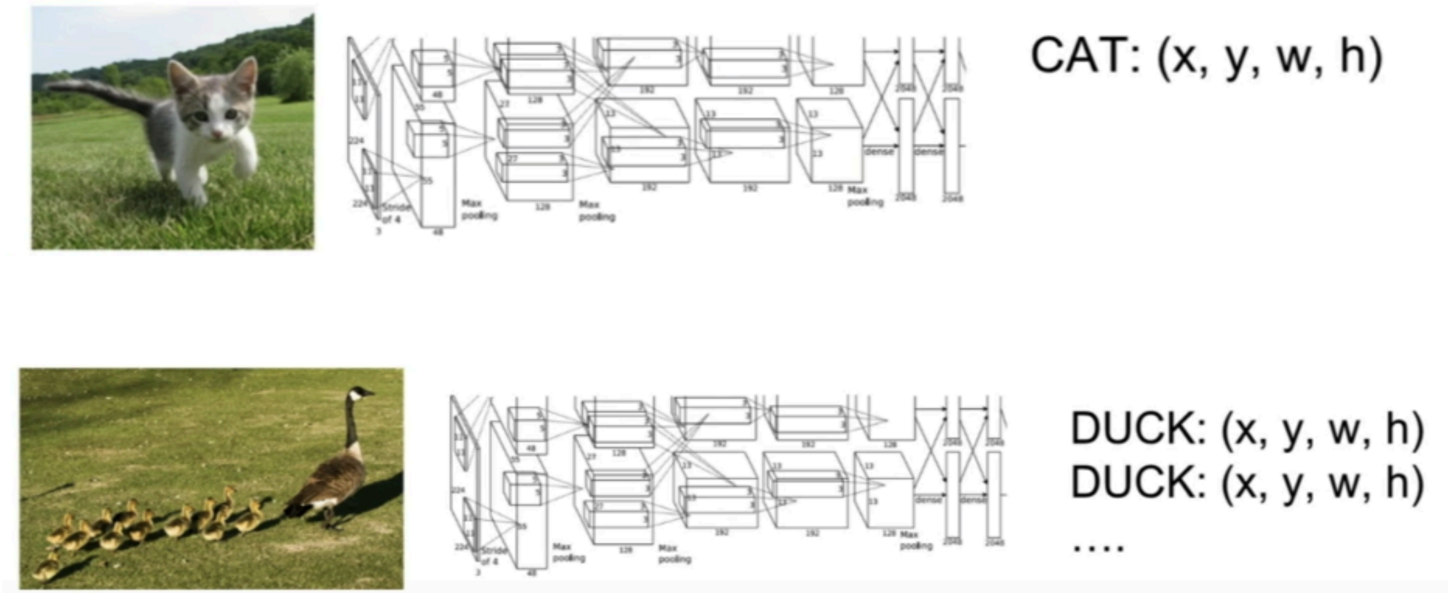
DISK
DeepLegato_2c
BULGE+DISK
BULGE



best Bulge 9
best Disk 9
Predicted
Validation

A WORKD ON R-CNNs..

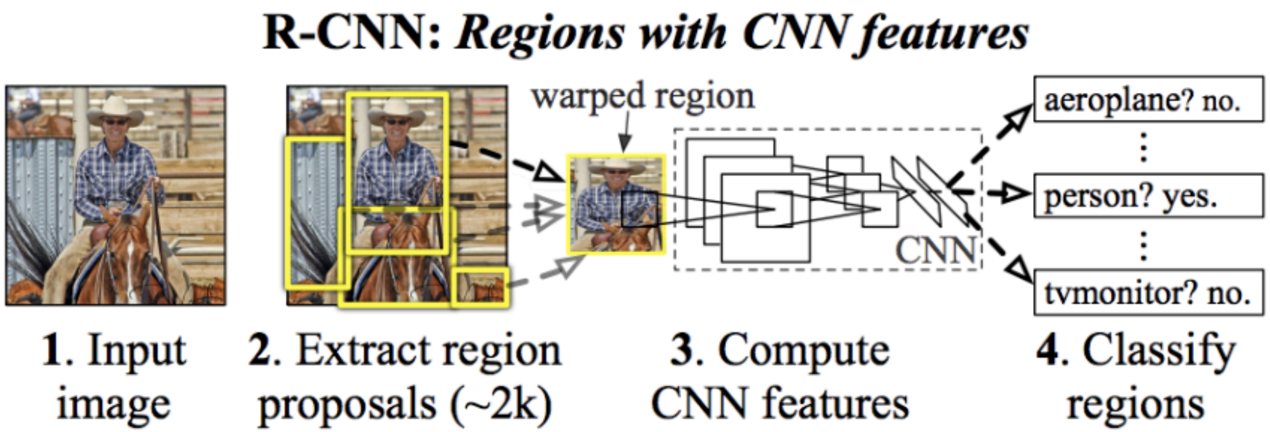
<u>JUST FOR YOUR RECORDS…</u>



Girshick+<u>14</u>

A WORKD ON R-CNNs..

<u>JUST FOR YOUR RECORDS…</u>

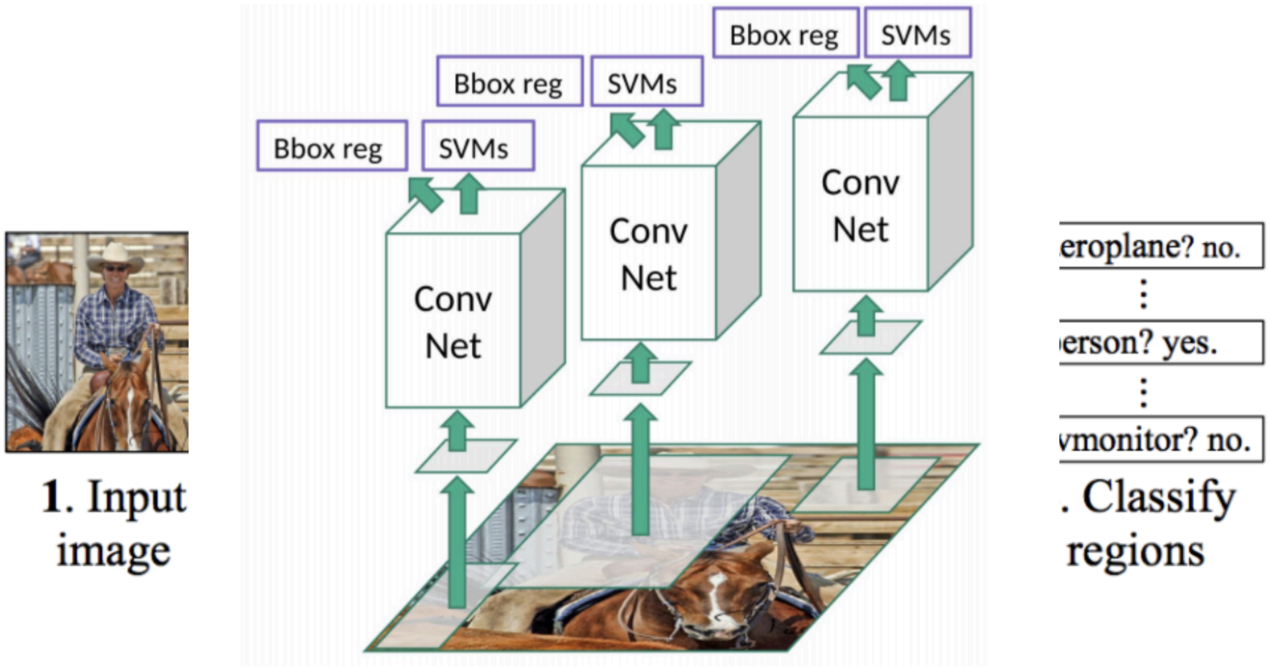


R-CNN

Girshick+<u>14</u>

A WORKD ON R-CNNs..

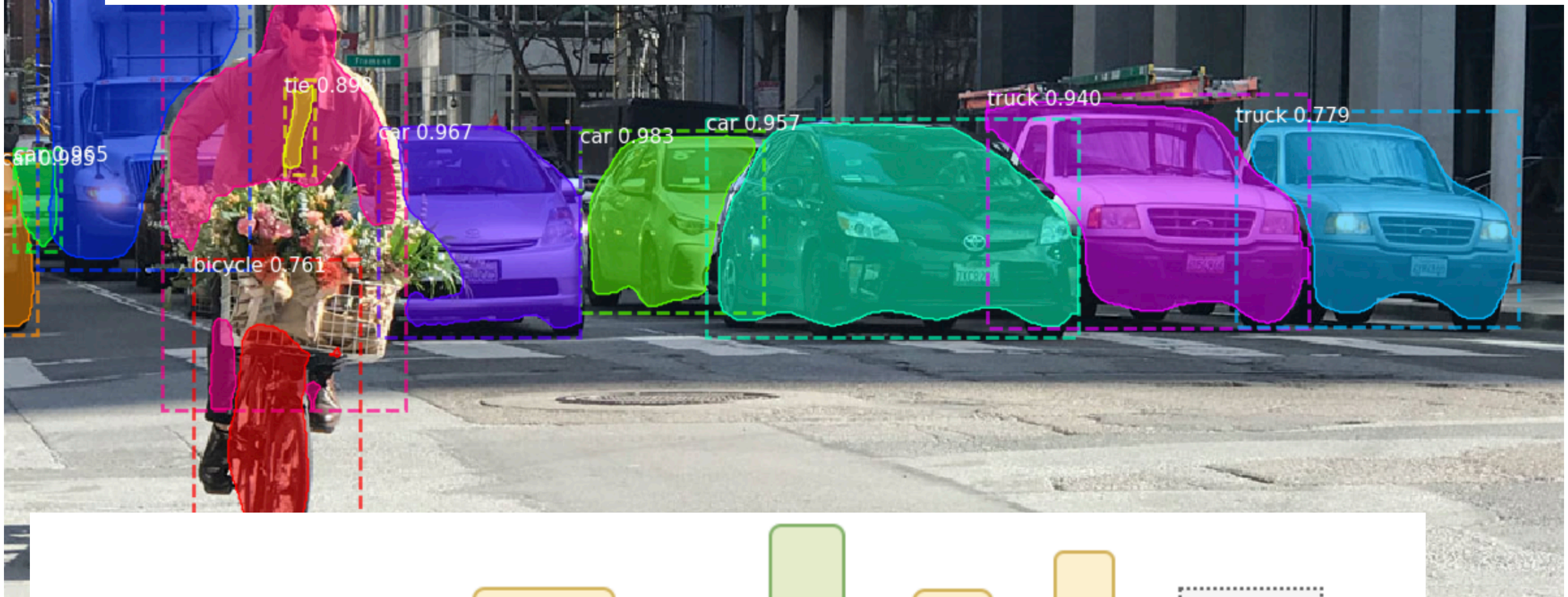
<u>JUST FOR YOUR RECORDS…</u>

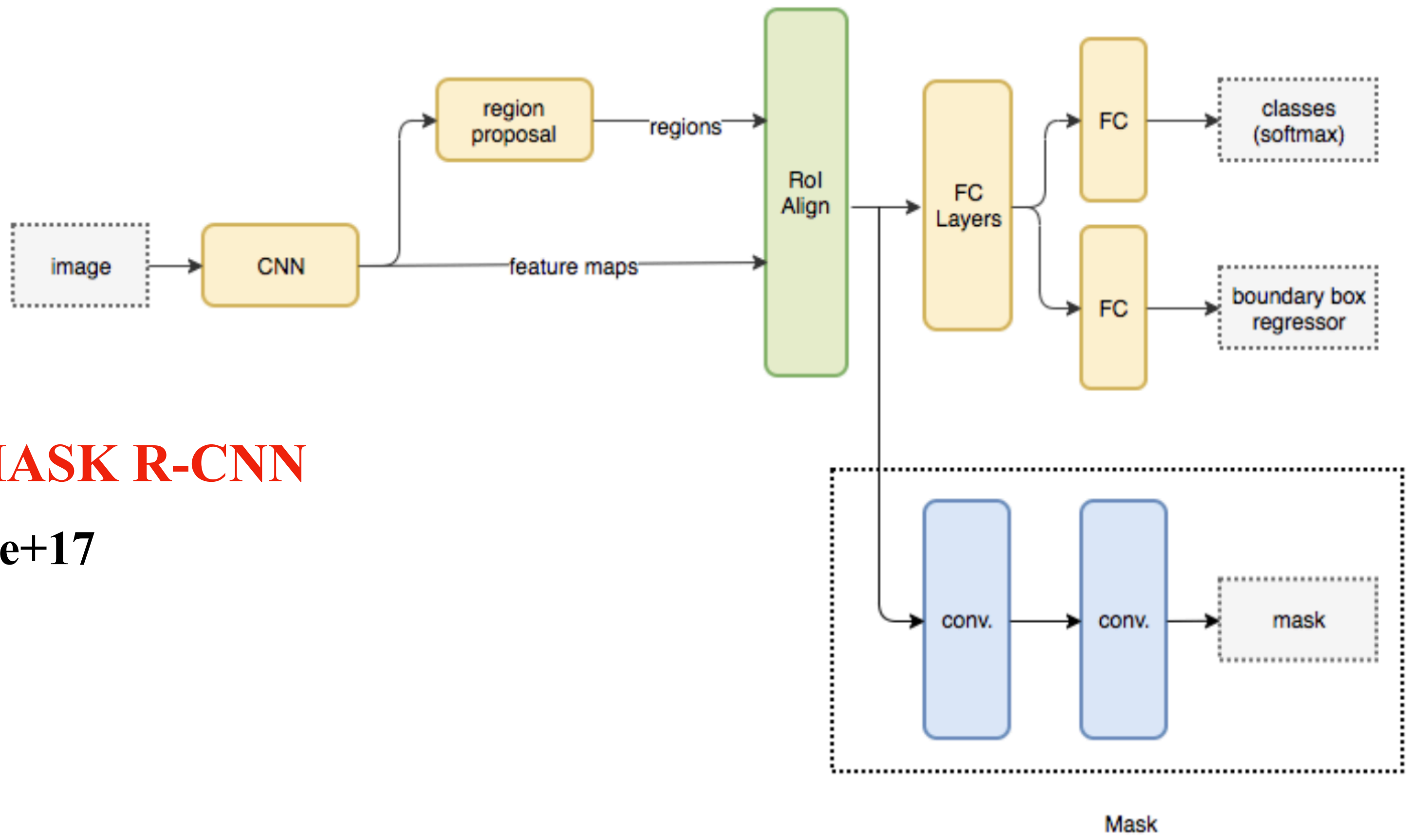


R-CNN

Girshick+14

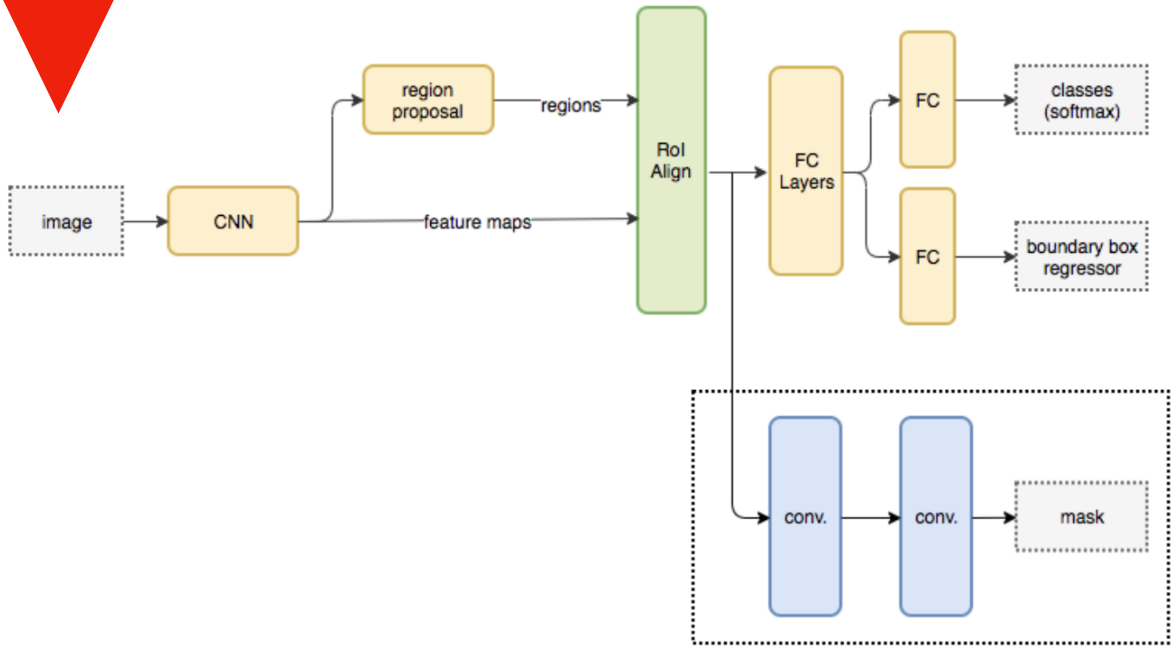
SIMULTANEOUS DETECTION + SEGMENTATION + CLASSIFICATION





MASK R-CNN

He+17



Mask

SEG-MAP

PHOTOMETRY

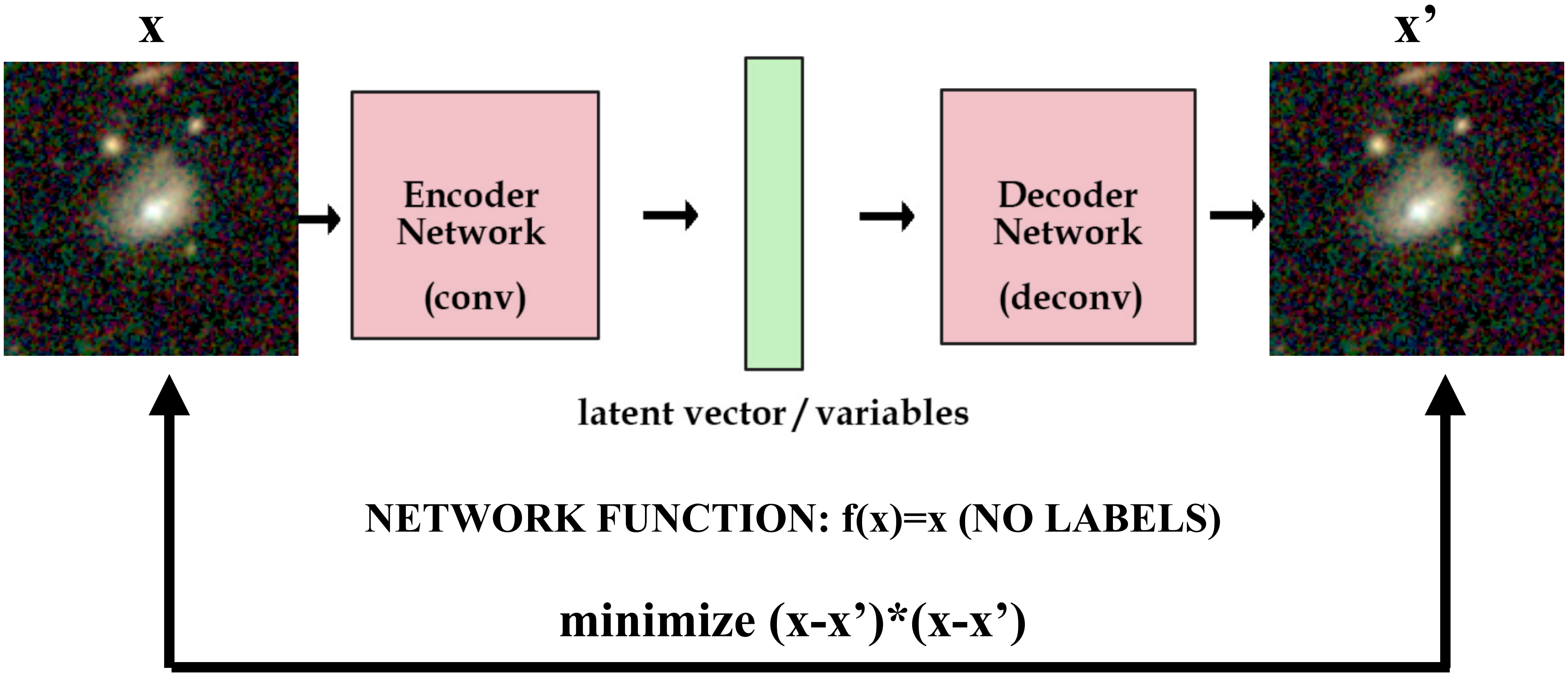
MORPHOLOGY

PHOTO-Z

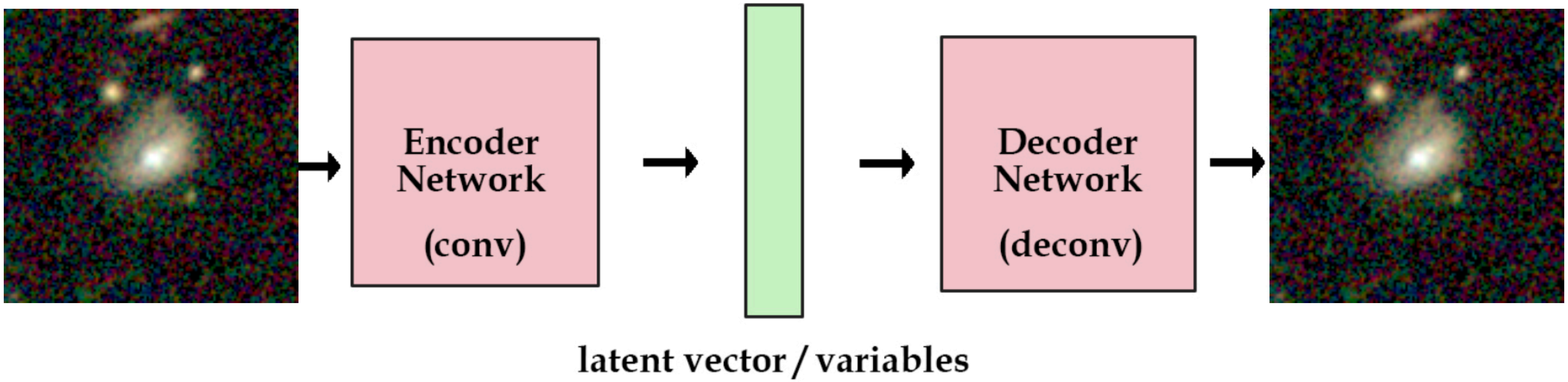
…

INTRODUCTION TO UNSUPERVISED DEEP LEARNING: GENERATIVE MODELS

AUTO-ENCODER

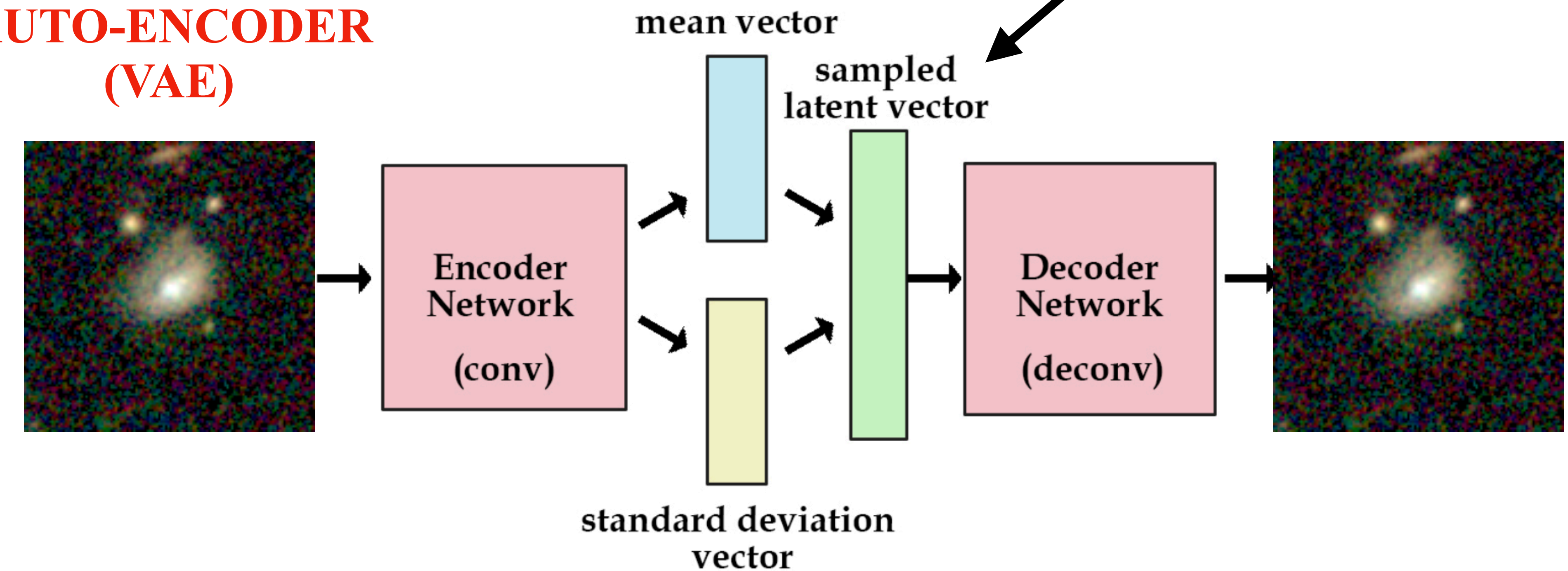


AUTO-ENCODER



“OPTIMAL” REPRESENTATION OF THE DATA

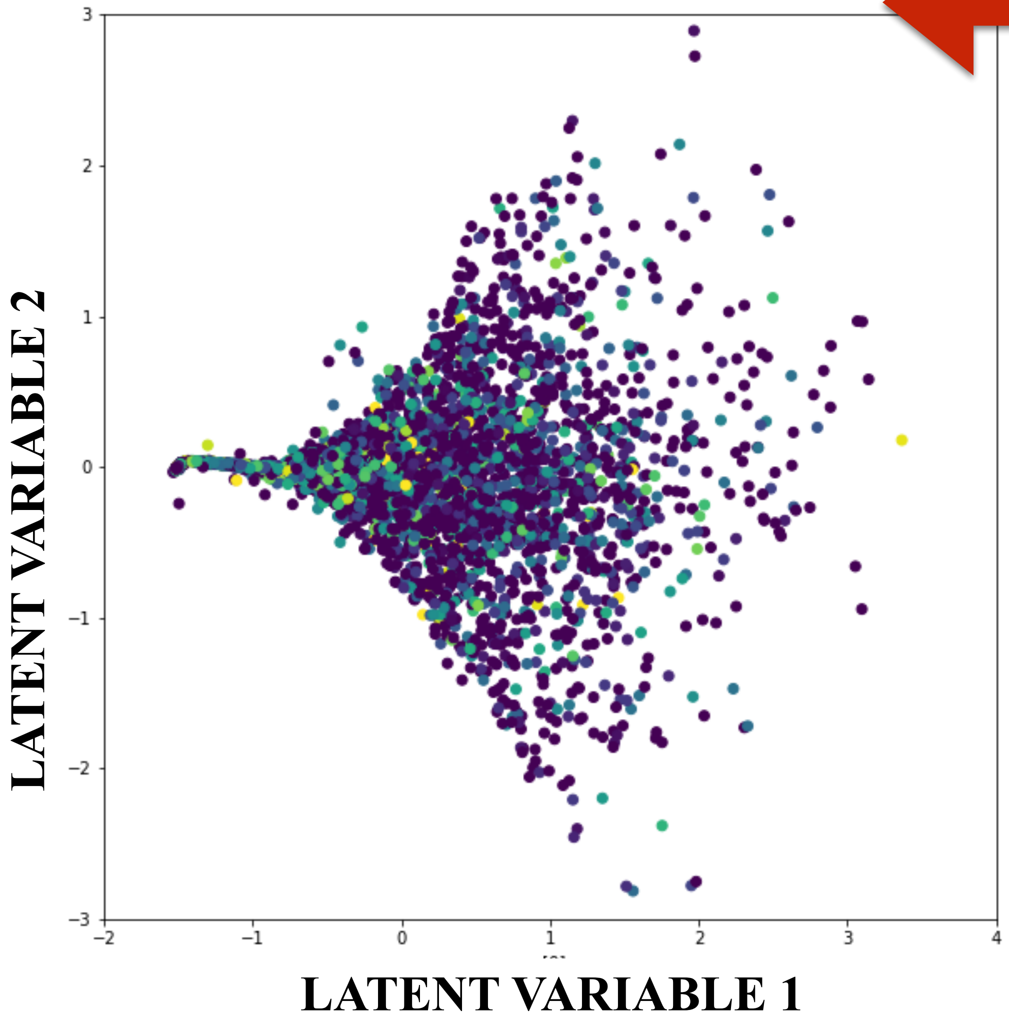
VARIATIONAL AUTO-ENCODER (VAE)

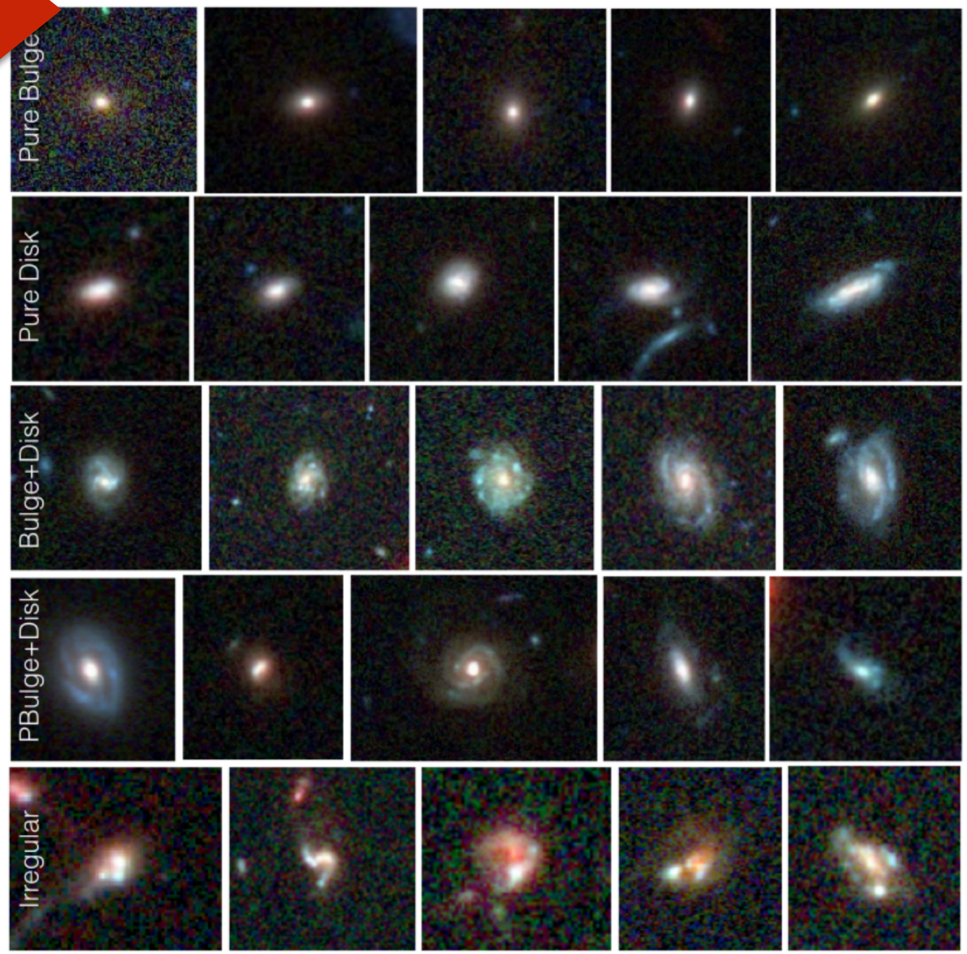


VAE DERIVED LATENT SPACE FOR CANDELS GALAXIES [H BAND]

PRELIMINARY!

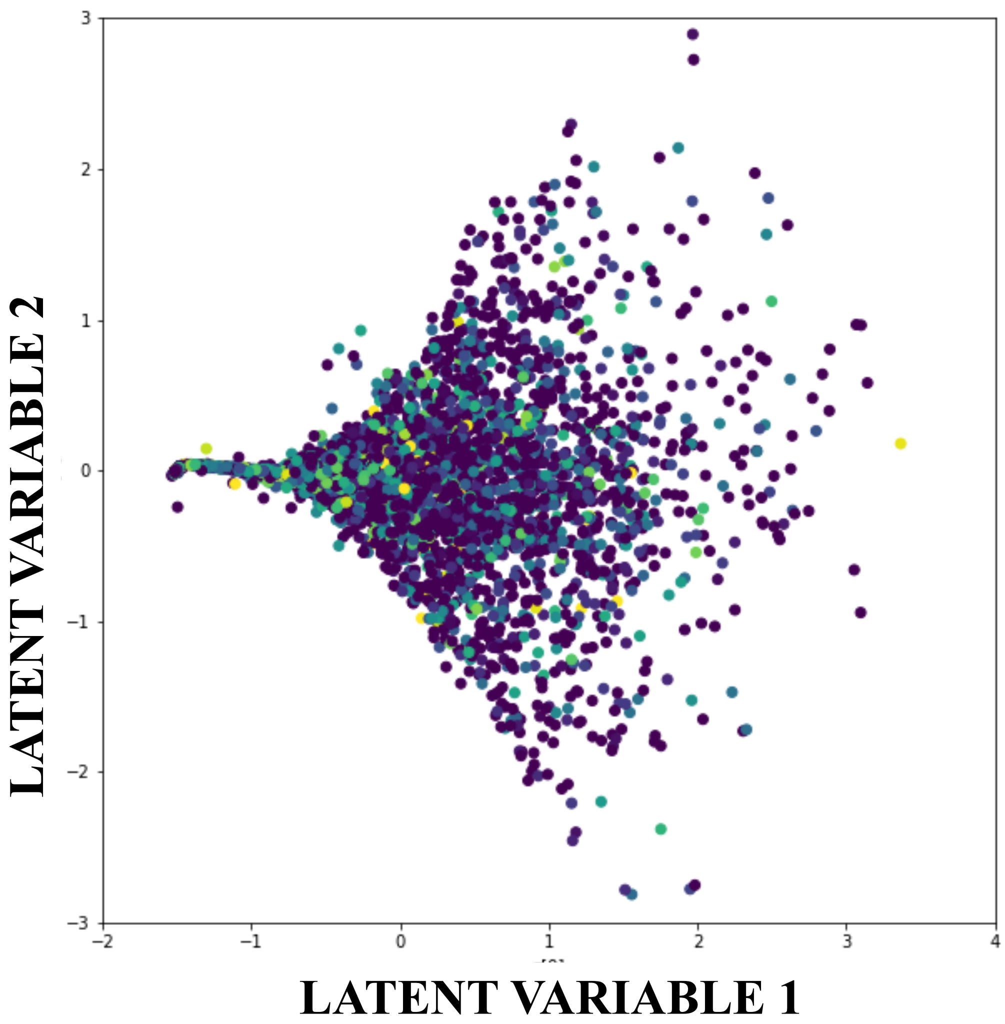
2 DIMENSIONS

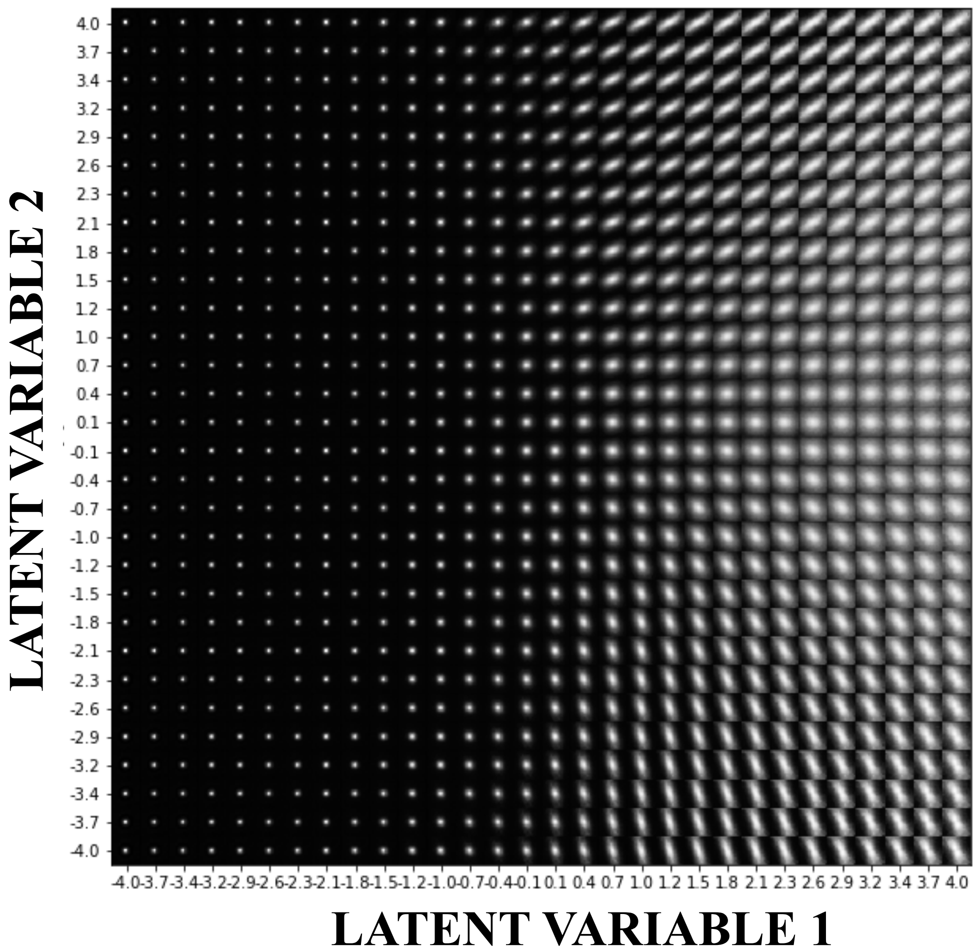




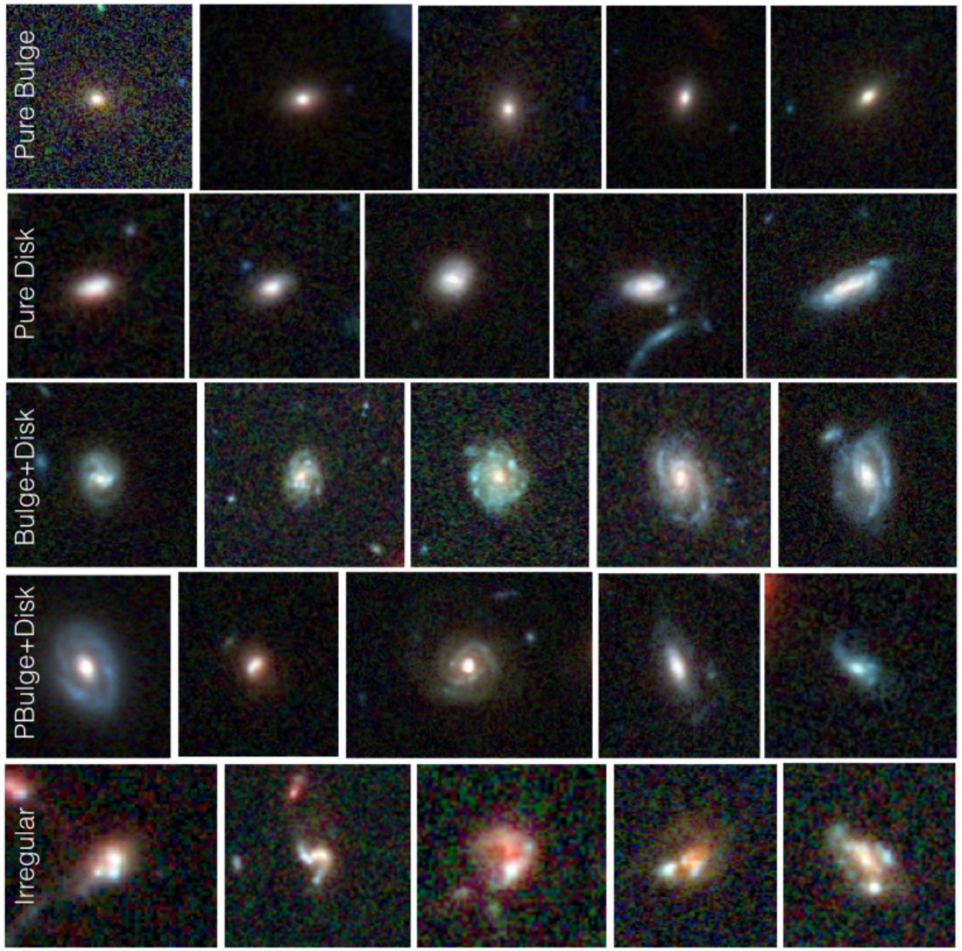
VAE DERIVED LATENT SPACE FOR CANDELS GALAXIES [H BAND]

PRELIMINARY!

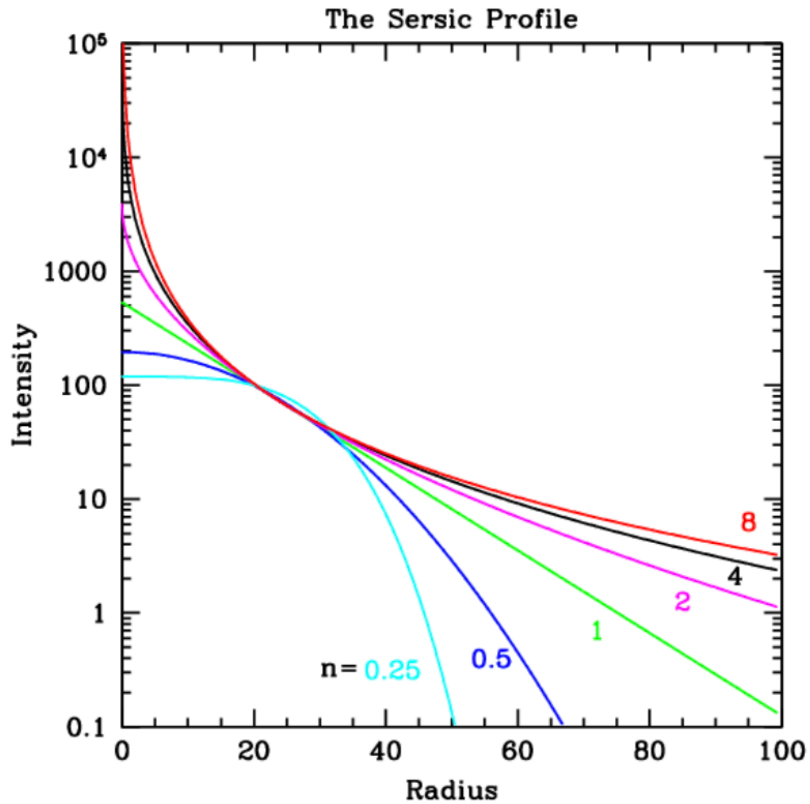




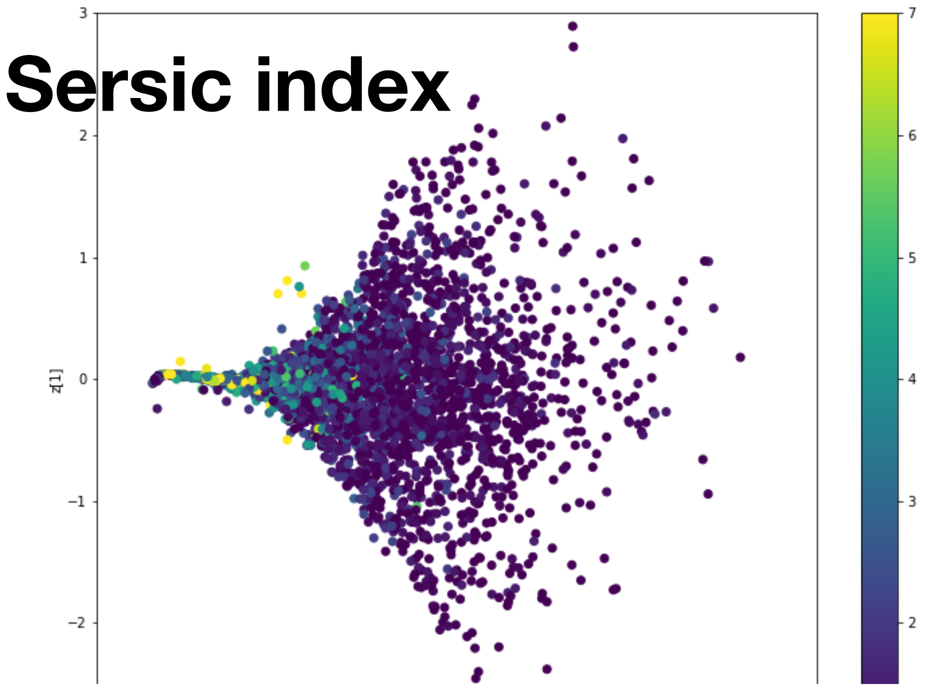
INTERPOLATION IN THE LATENT SPACE GENERATES GALAXIES WITH DIFFERENT PROPERTIES

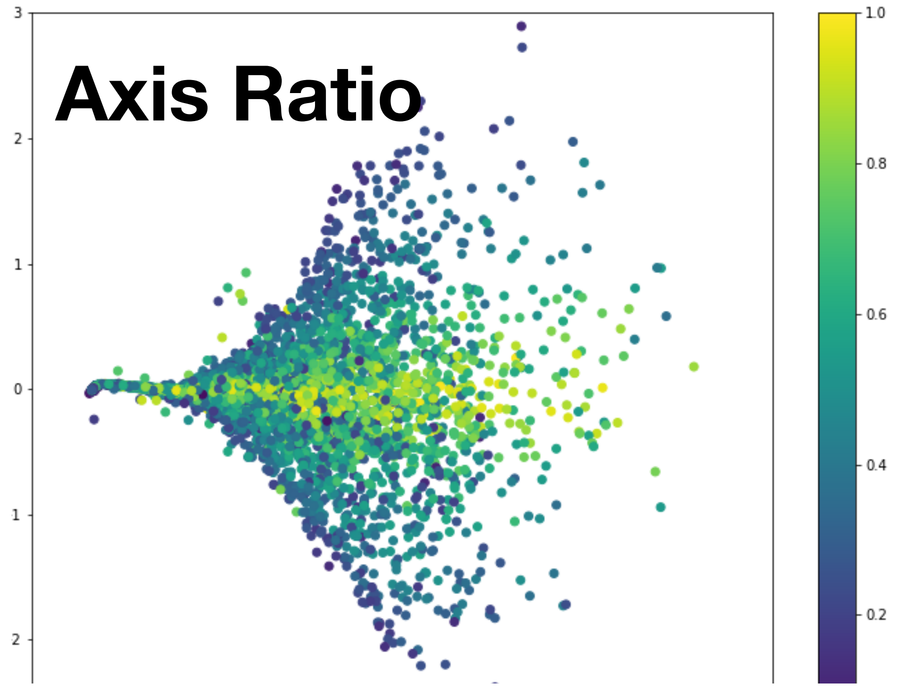


TRADITIONALLY GALAXY STRUCTURE IS DESCRIBED WITH AT LEAST 4 PARAMETERS

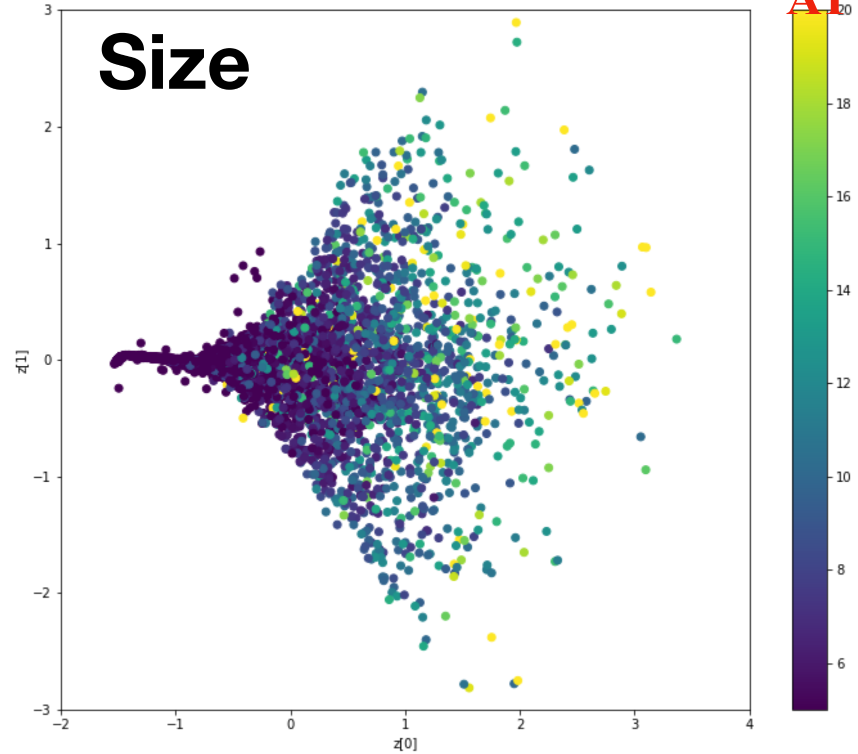


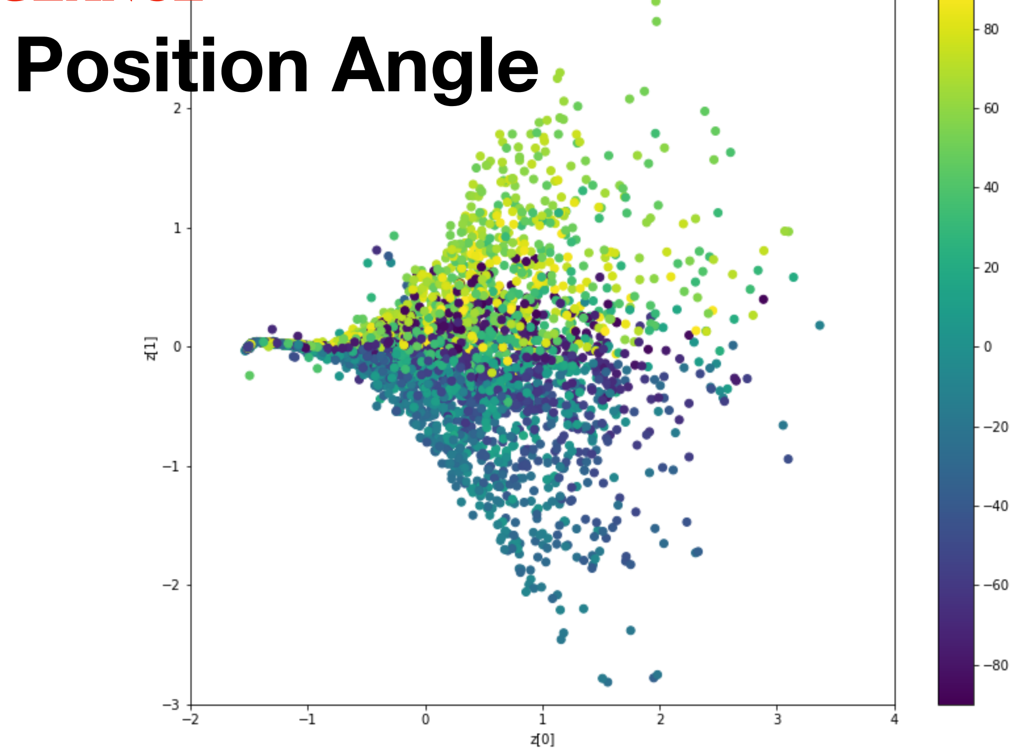
Effective Radius, Sersic Index, Axis-Ratio, Position Angle

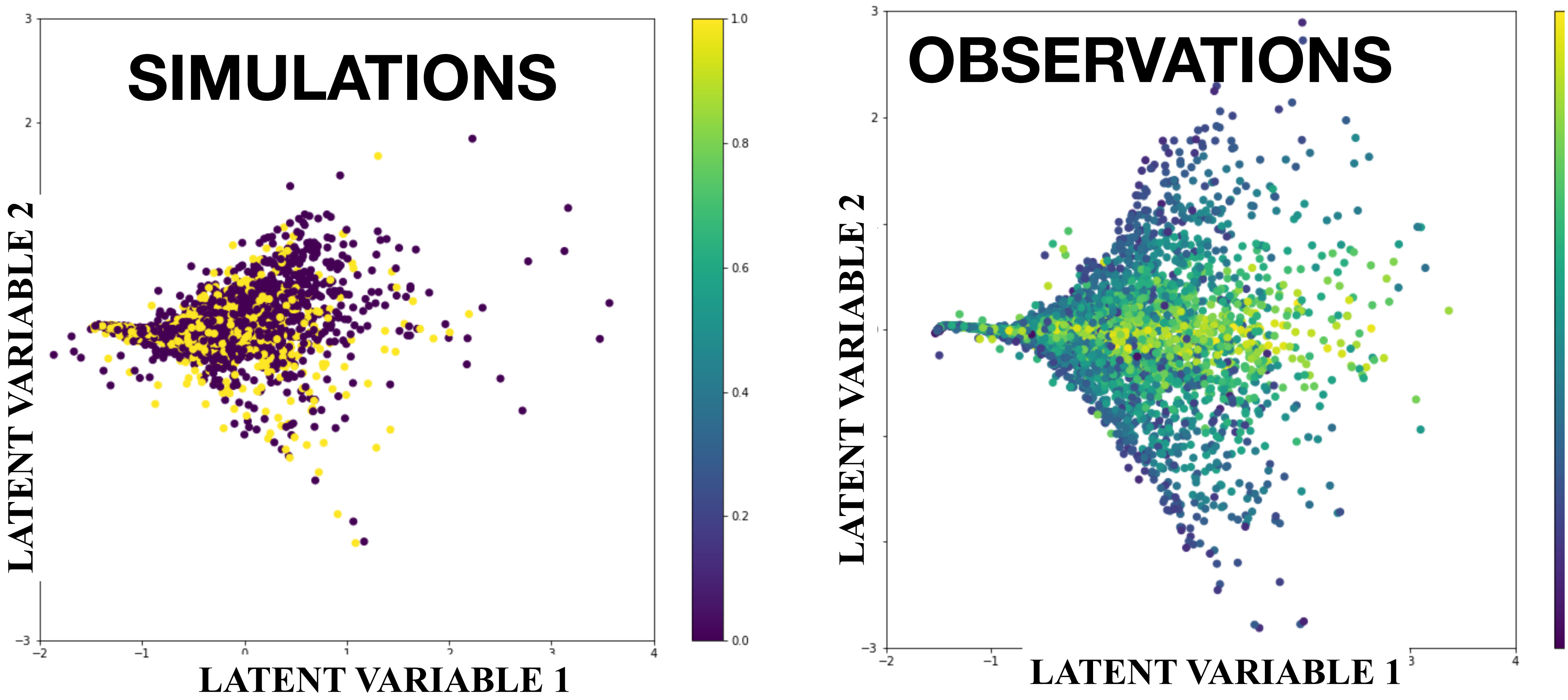




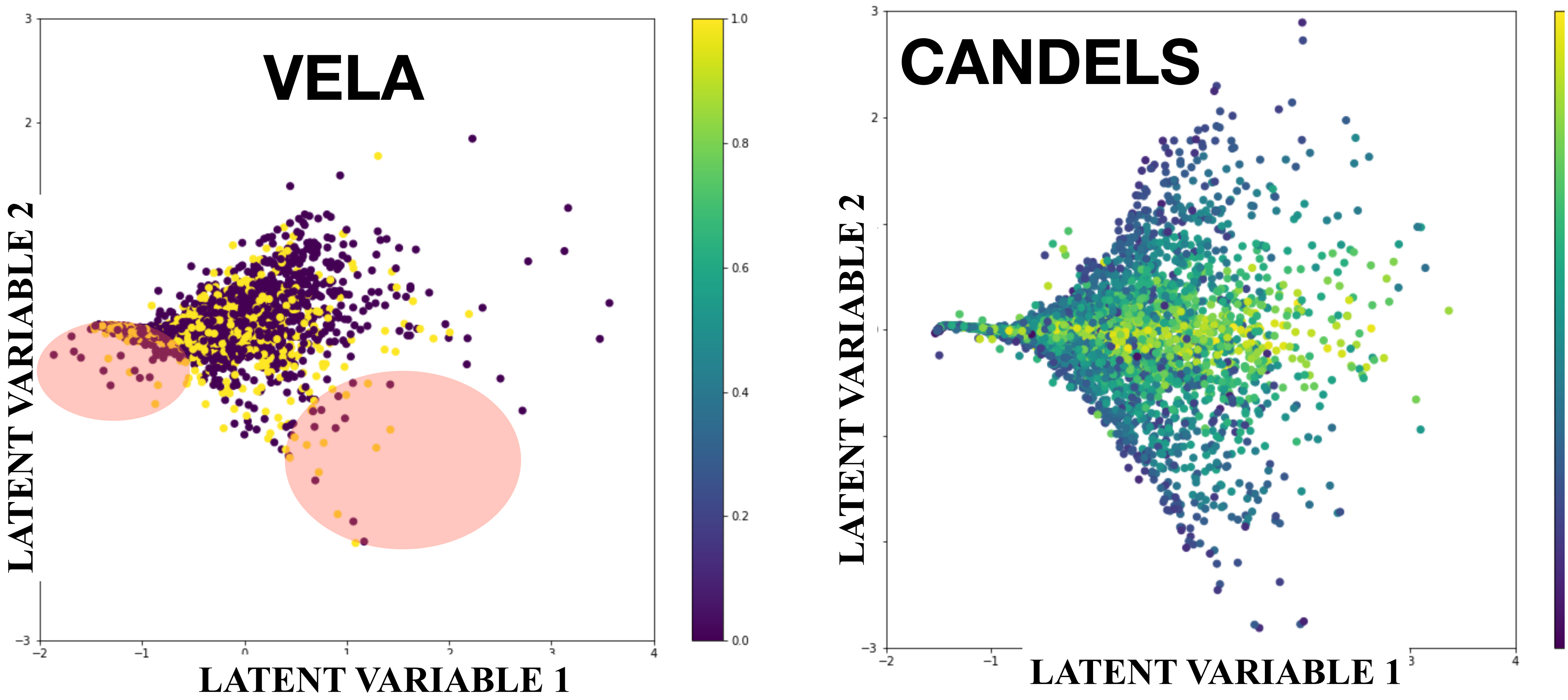
EXPLORE THE MORPHOLOGICAL DISTRIBUTION OF GALAXIES AT A GLANCE





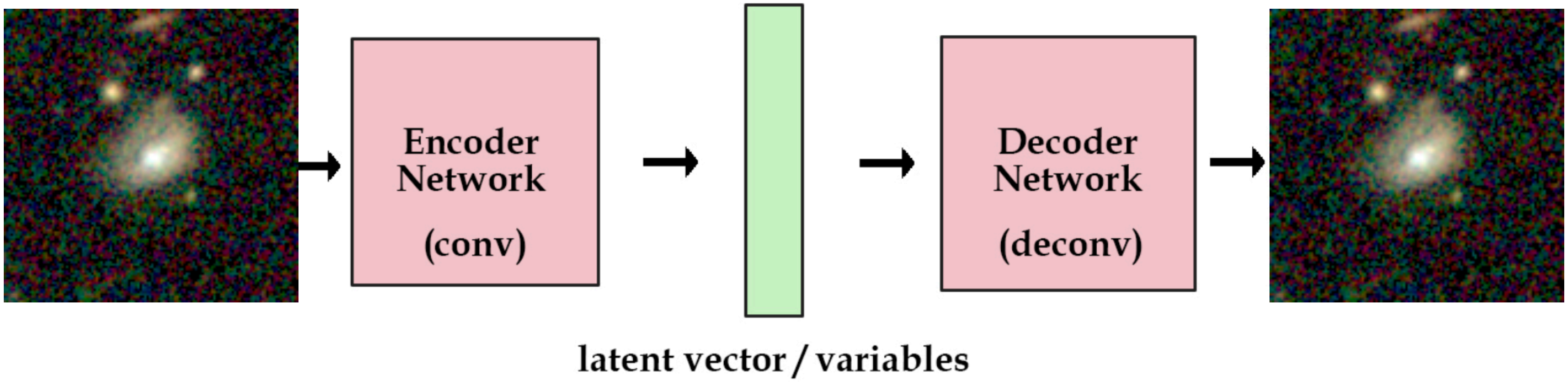


HOW SIMULATIONS POPULATE THE LATENT SPACE?



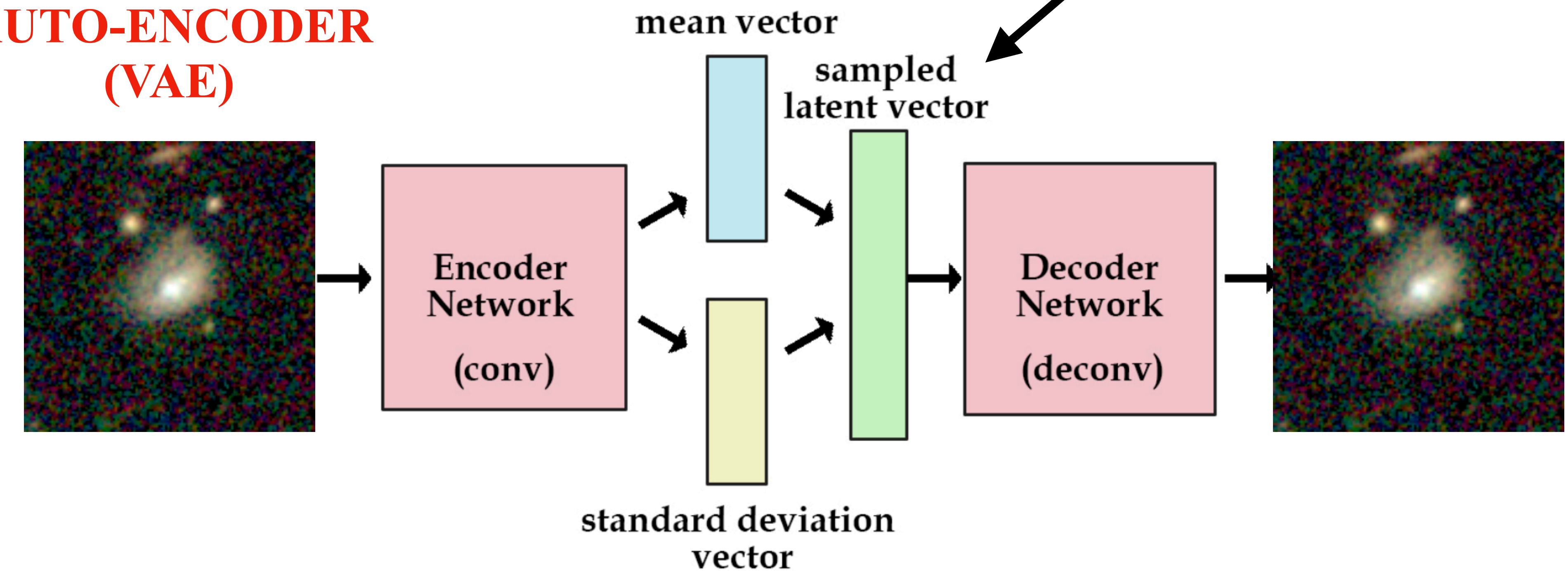
HOW SIMULATIONS POPULATE THE LATENT SPACE?

AUTO-ENCODER



“OPTIMAL” REPRESENTATION
OF THE DATA

VARIATIONAL AUTO-ENCODER (VAE)



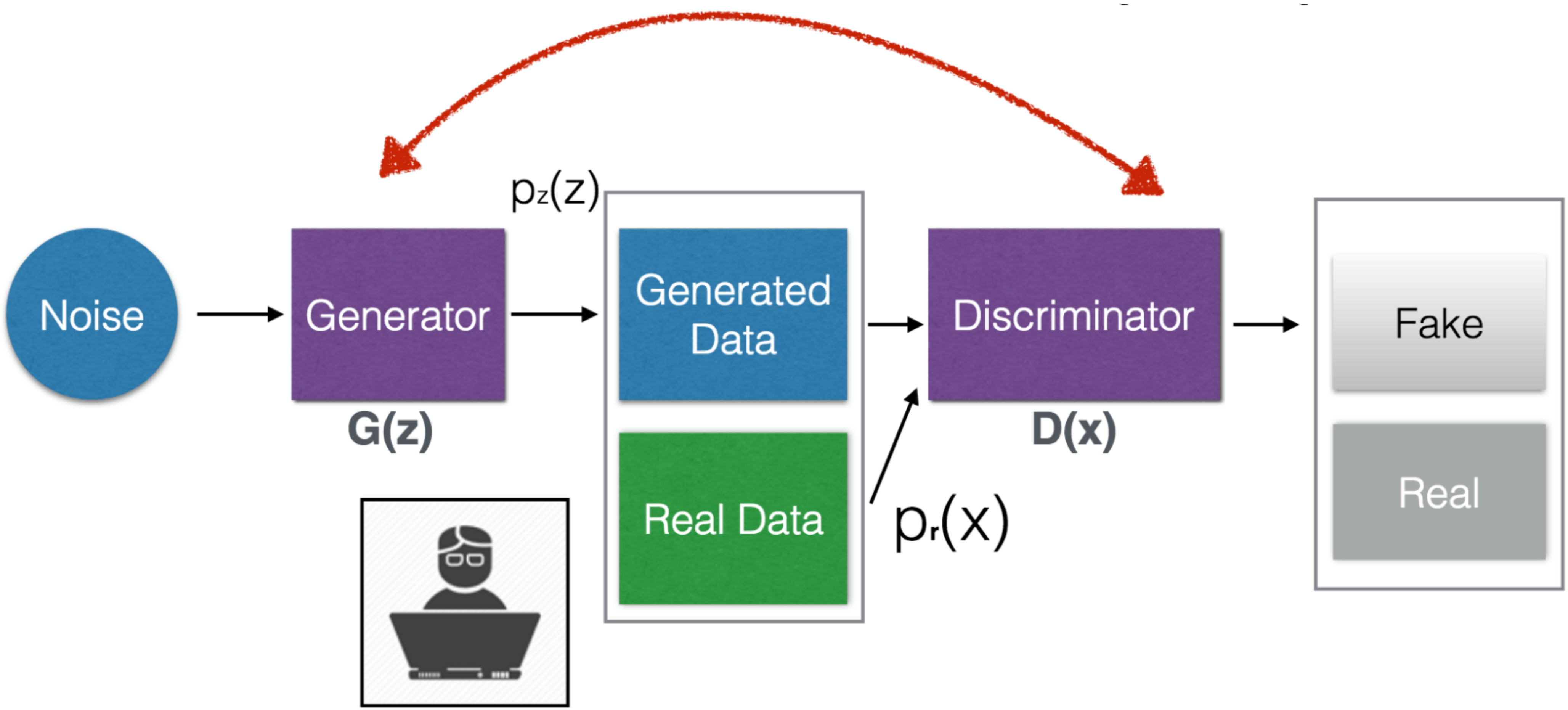
GENERATIVE ADVERSARIAL NETWORKS

(Goodfellow+14)

5570 citation in 4 years!

GENERATIVE ADVERSARIAL NETWORKS

(Goodfellow+14)

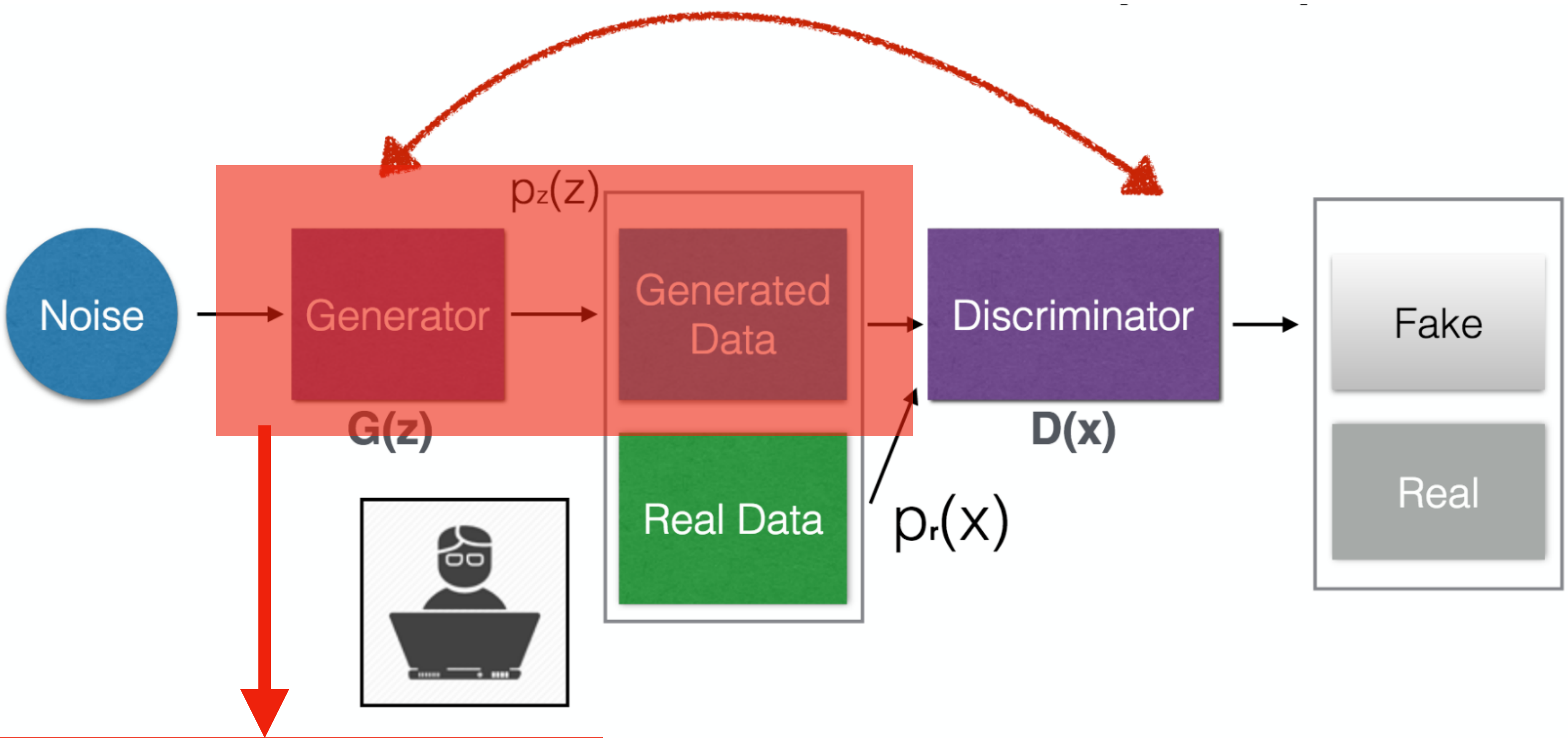


TWO COMPETING NETWORKS

GENERATIVE ADVERSARIAL NETWORKS

(Goodfellow+)

<u>TWO COMPETING NETWORKS</u>

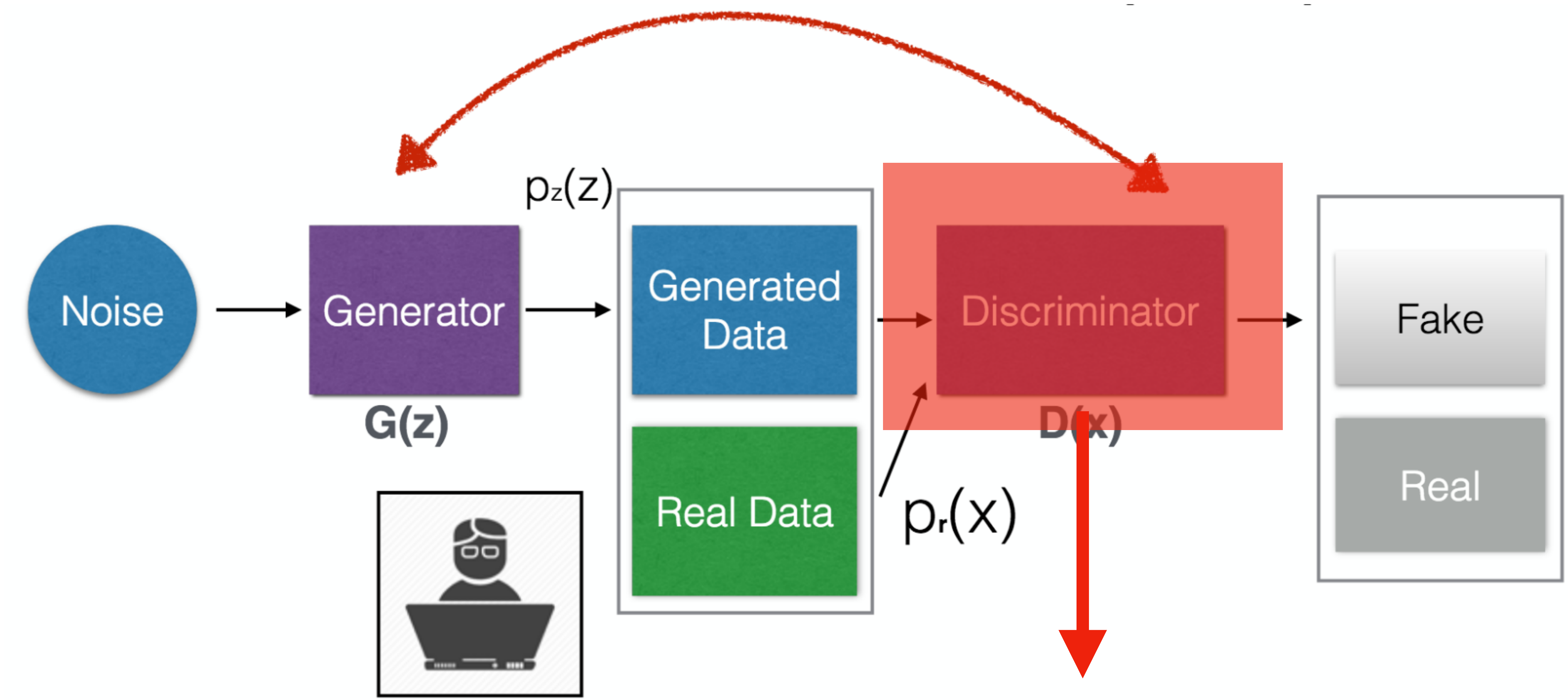


Every 2 iterations the generator is trained to force the discriminator to classify as real

GENERATIVE ADVERSARIAL NETWORKS

(Goodfellow+)

TWO COMPETING NETWORKS



Every 2 iterations the discriminator is trained to force to distinguish between real and fake

PROGRESSIVE GROWING OF GANS FOR IMPROVED QUALITY, STABILITY, AND VARIATION

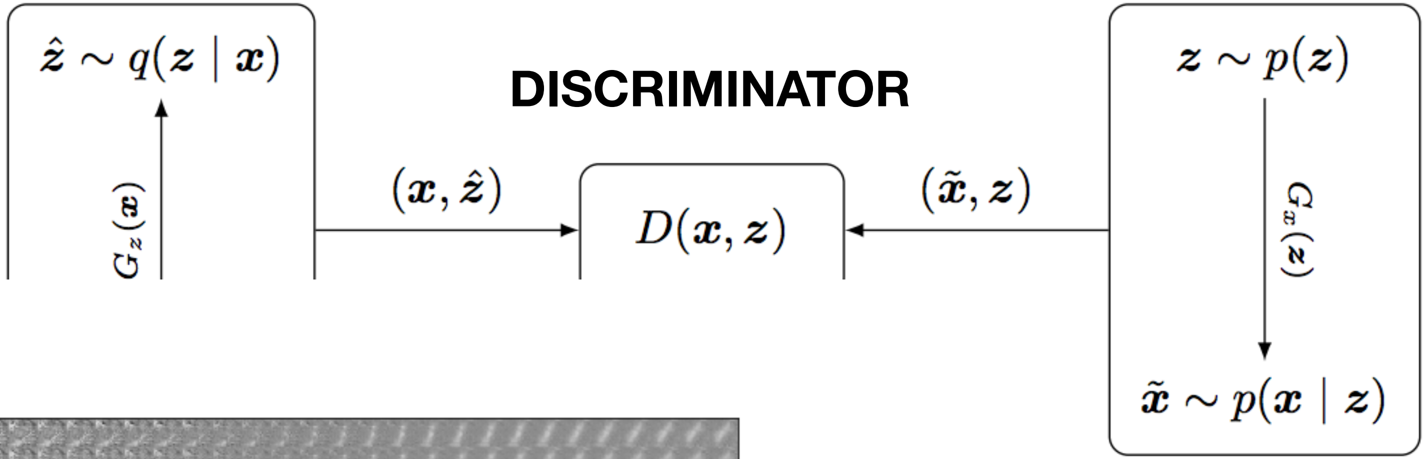
Tero Karras
NVIDIA

Timo Aila
NVIDIA

Samuli Laine
NVIDIA

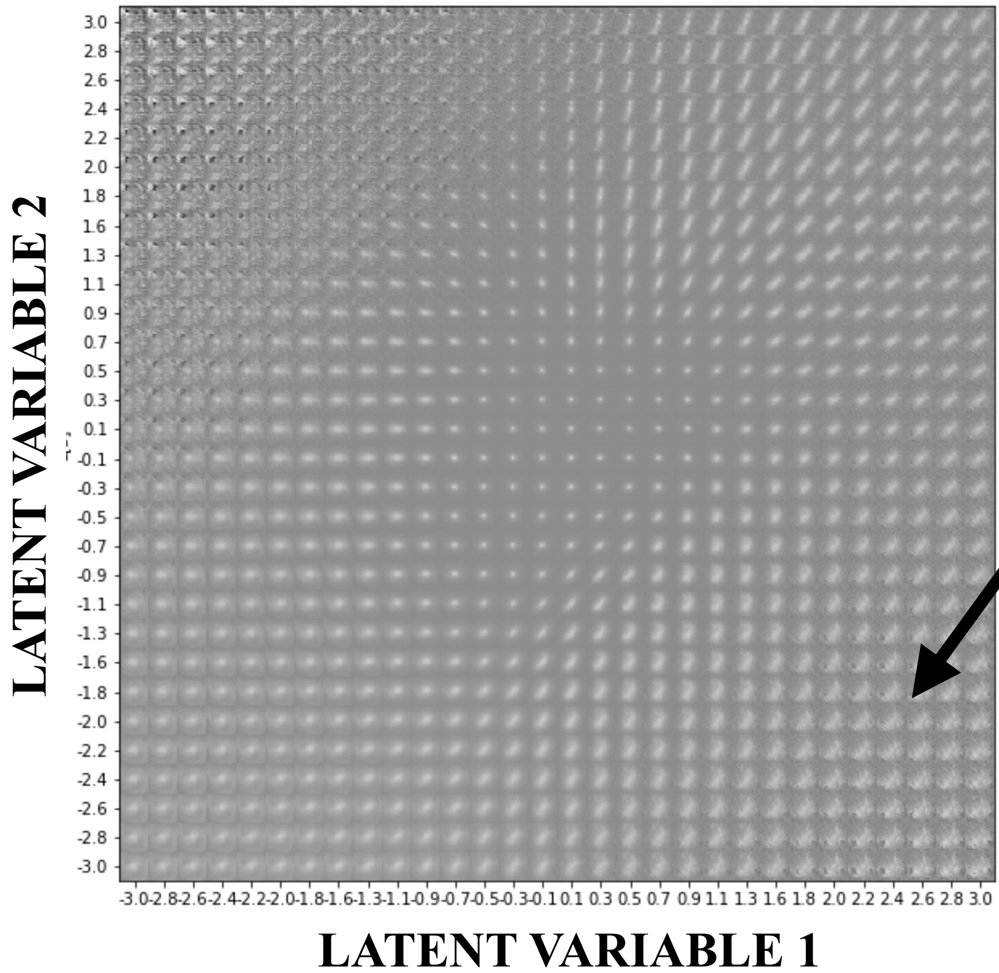
Jaakko Lehtinen
NVIDIA
Aalto University

NVIDIA

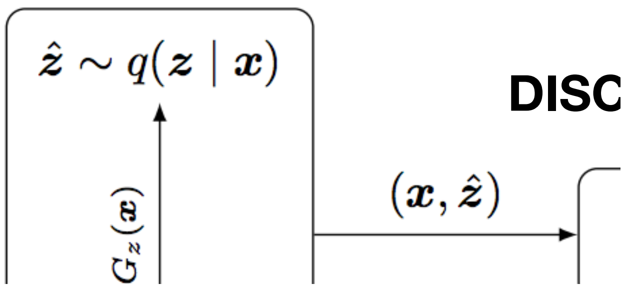


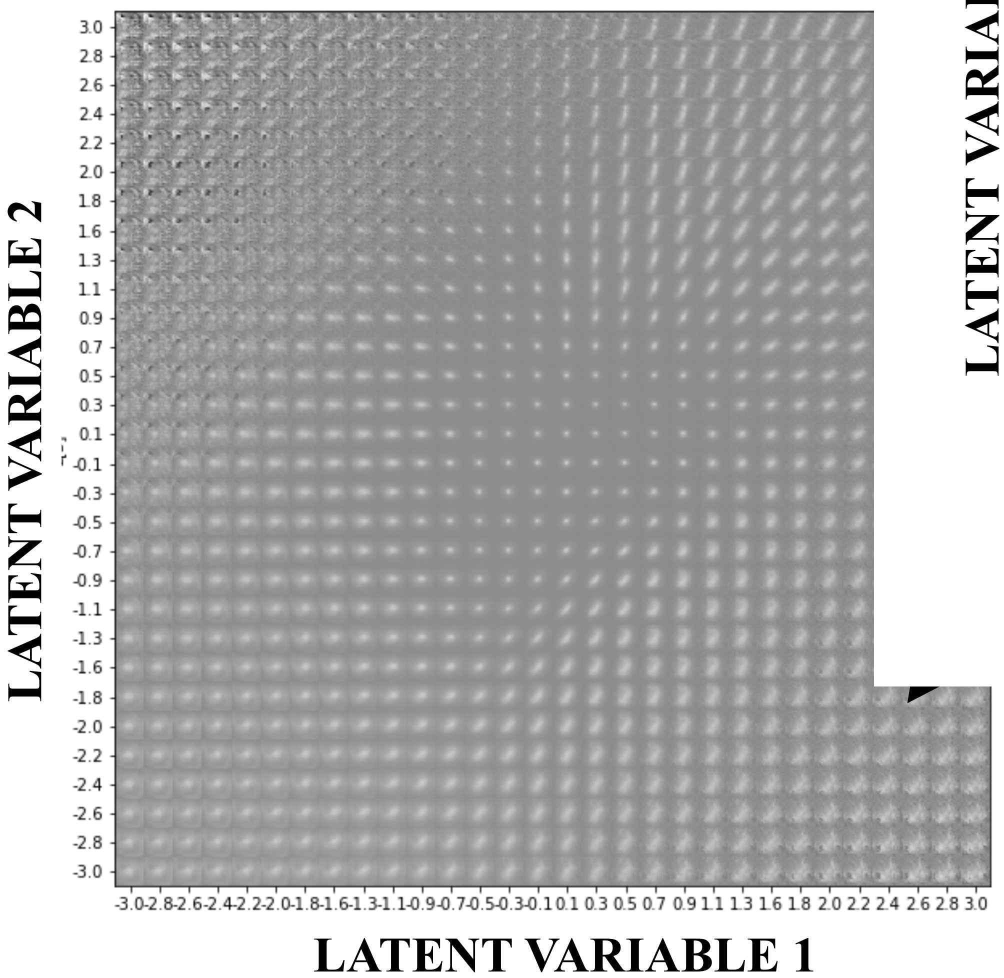
nference (ALI) game.

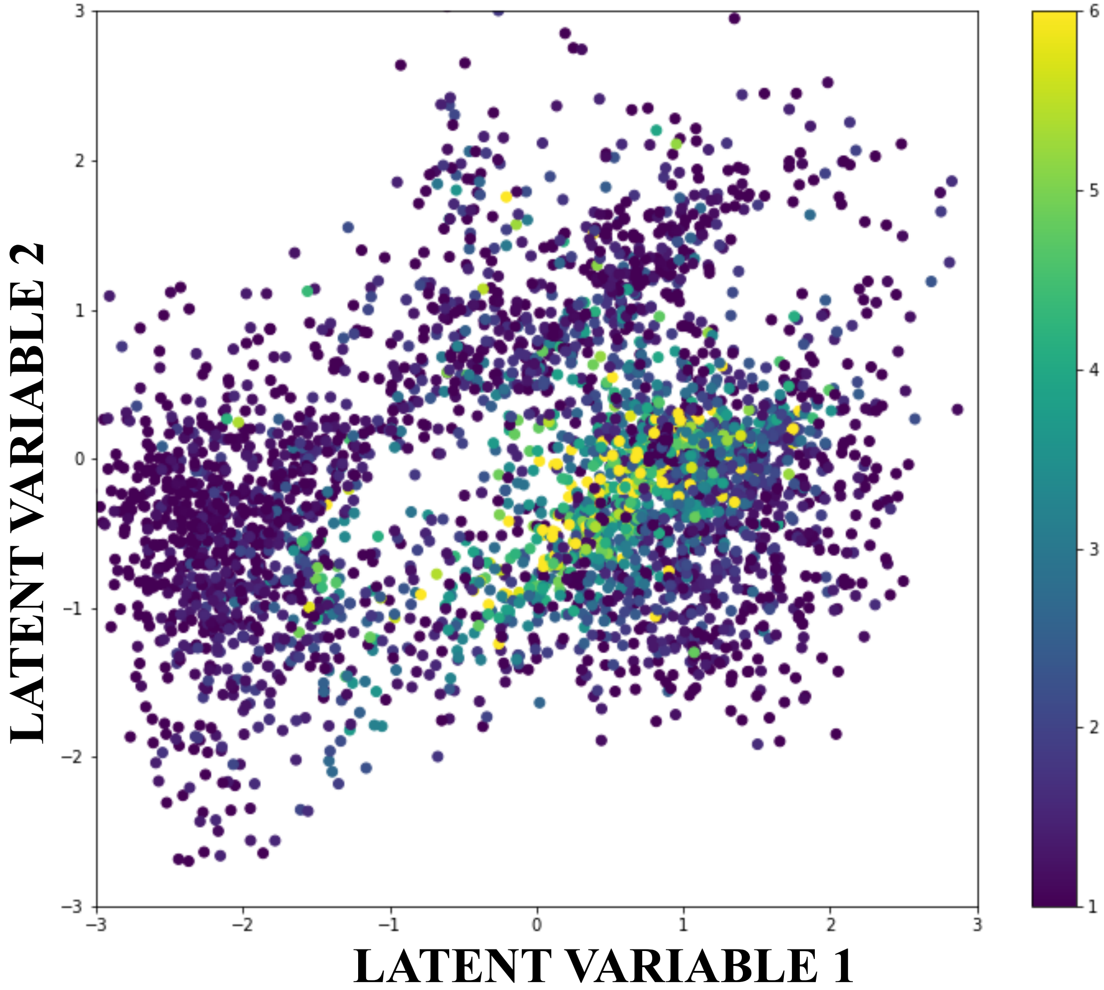
Dumoulin+17



GENERATES MORE COMPLEX MODELS

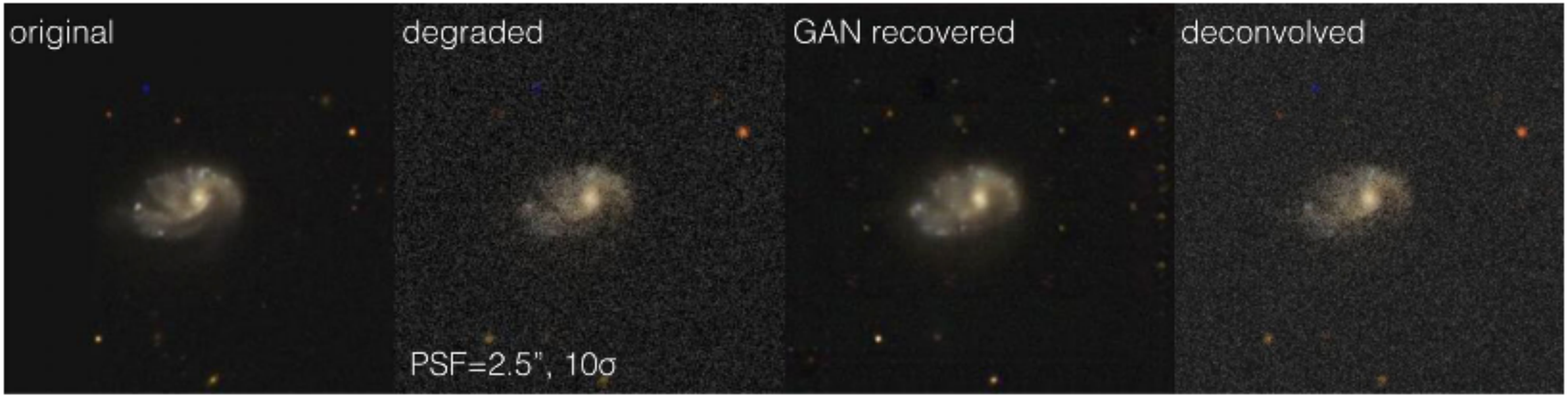


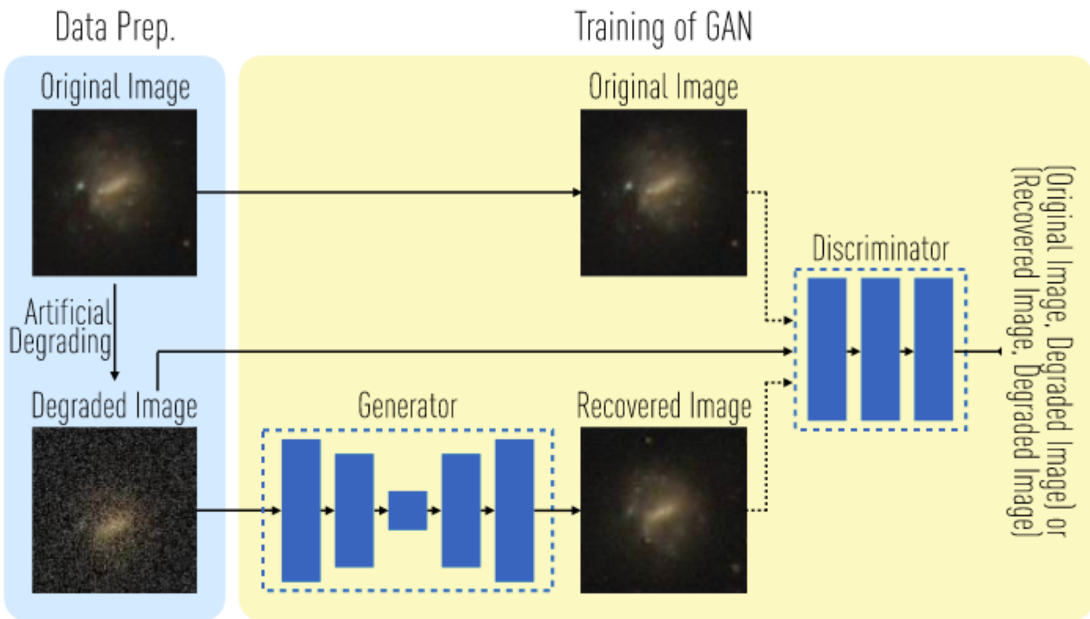




MORE DIFFICULT TO INTERPRET

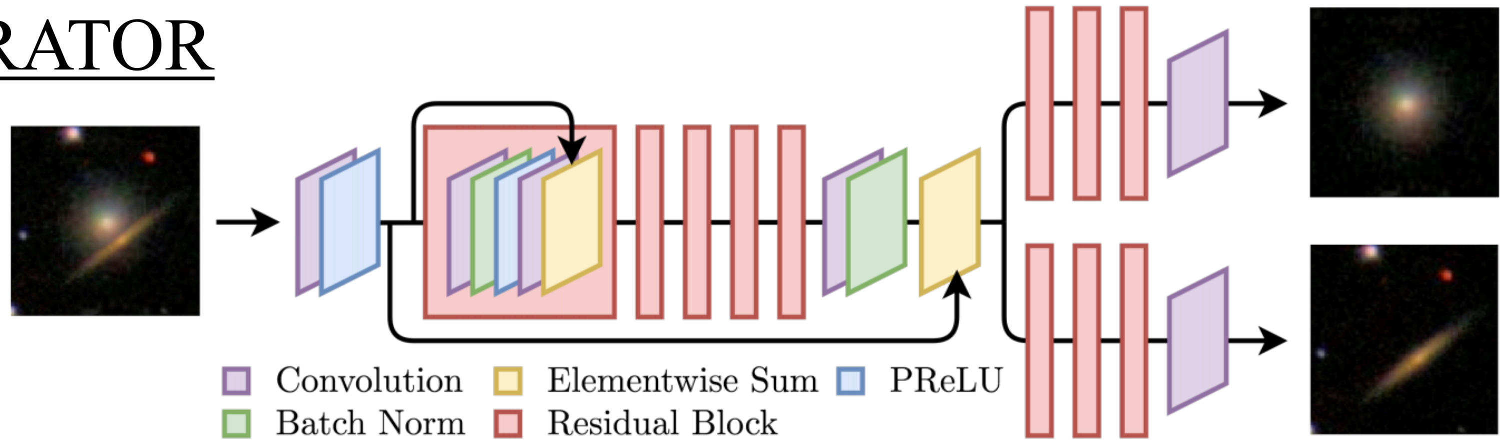
GANs: OTHER APPLICATIONS IN ASTRONOMY



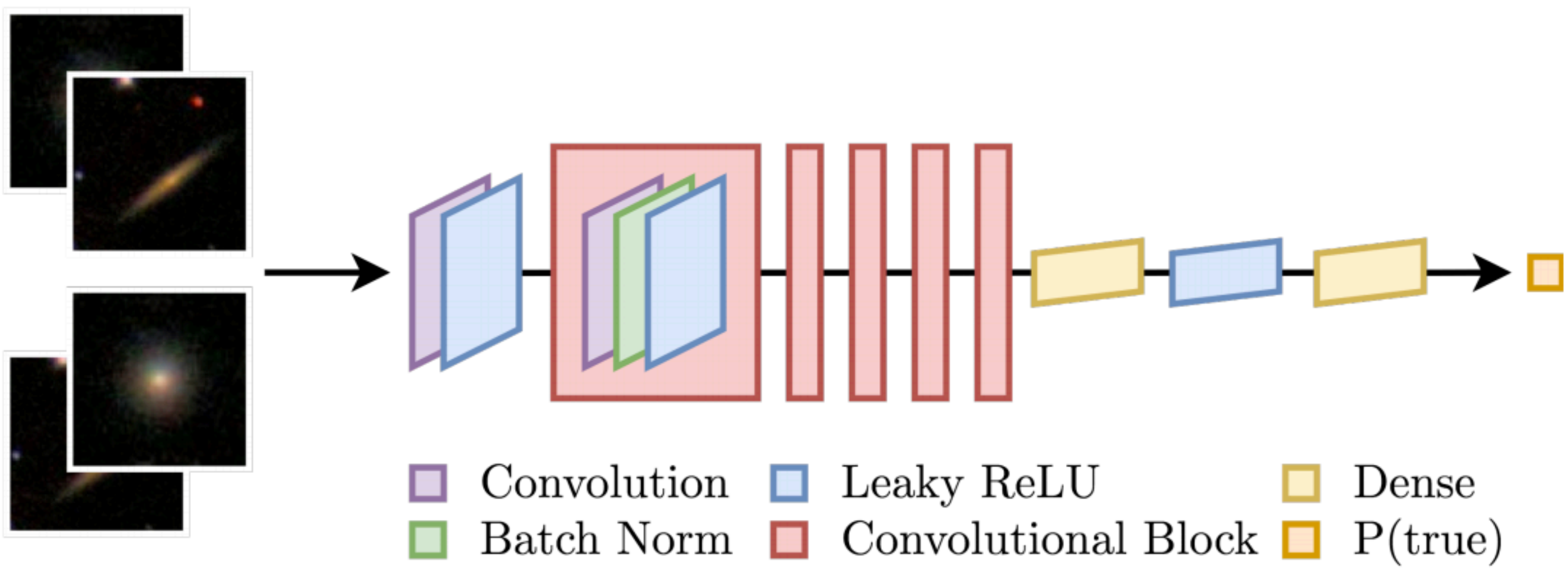


Schawinsky+17

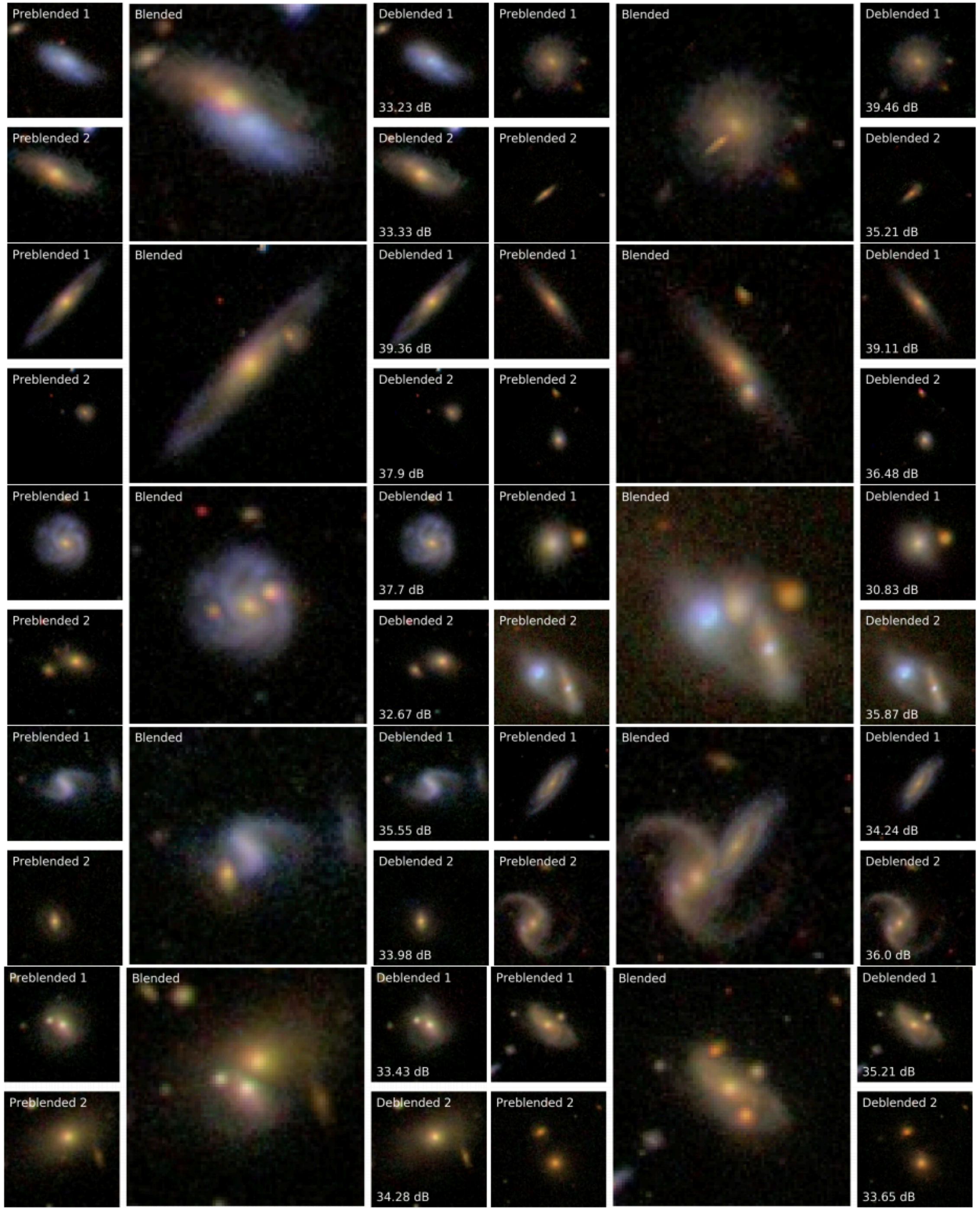
GENERATOR



DISCRIMINATOR



Reiman+18


Prebled 1
Blended
Deblended 1
33.23 dB
Preblended 2
Deblended 2
33.33 dB
Preblended 1
Blended
Deblended 1
39.46 dB
Preblended 2
Deblended 2
35.21 dB
Preblended 1
Blended
Deblended 1
39.36 dB
Preblended 2
Deblended 2
37.9 dB
Preblended 1
Blended
Deblended 1
39.11 dB
Preblended 2
Deblended 2
36.48 dB
Preblended 1
Blended
Deblended 1
37.7 dB
Preblended 2
Deblended 2
32.67 dB
Preblended 1
Blended
Deblended 1
30.83 dB
Preblended 2
Deblended 2
35.87 dB
Preblended 1
Blended
Deblended 1
35.55 dB
Preblended 2
Deblended 2
33.98 dB
Preblended 1
Blended
Deblended 1
34.24 dB
Preblended 2
Deblended 2
36.0 dB
Preblended 1
Blended
Deblended 1
33.43 dB
Preblended 2
Deblended 2
34.28 dB
Preblended 1
Blended
Deblended 1
35.21 dB
Preblended 2
Deblended 2
33.65 dB

Reiman+18

ANOMALY DETECTION WITH GANs

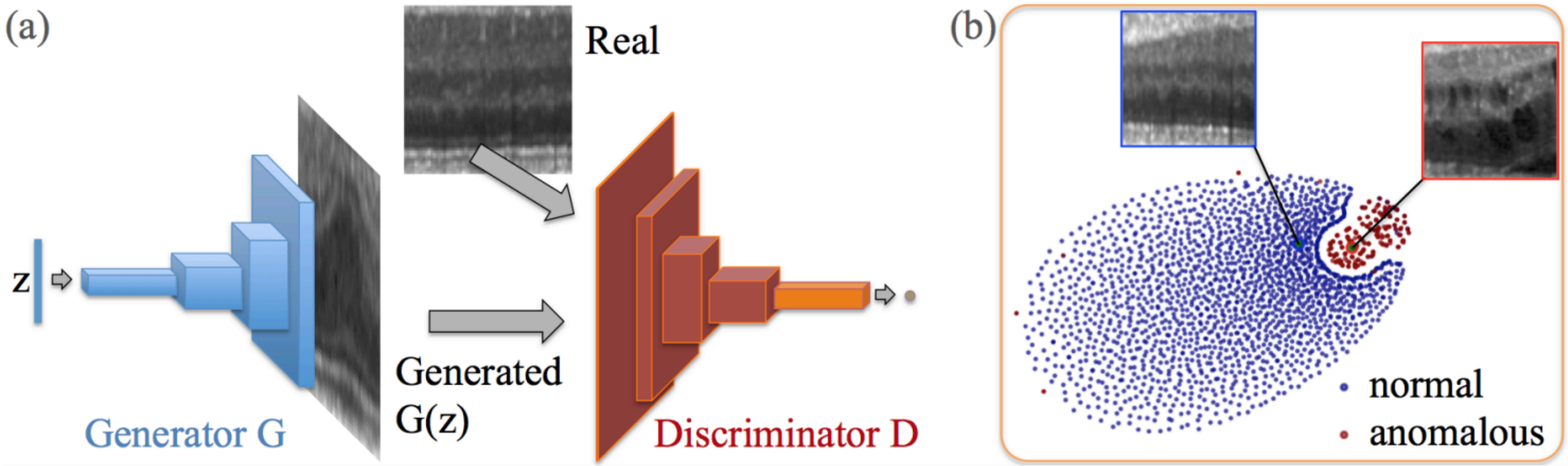


Fig. 2. (a) Deep convolutional generative adversarial network. (b) t-SNE embedding of normal (blue) and anomalous (red) images on the feature representation of the last convolution layer (orange in (a)) of the discriminator.

Schlegl+17

ANOMALY DETECTION WITH GANs

ANOMALY SCORE

(a)

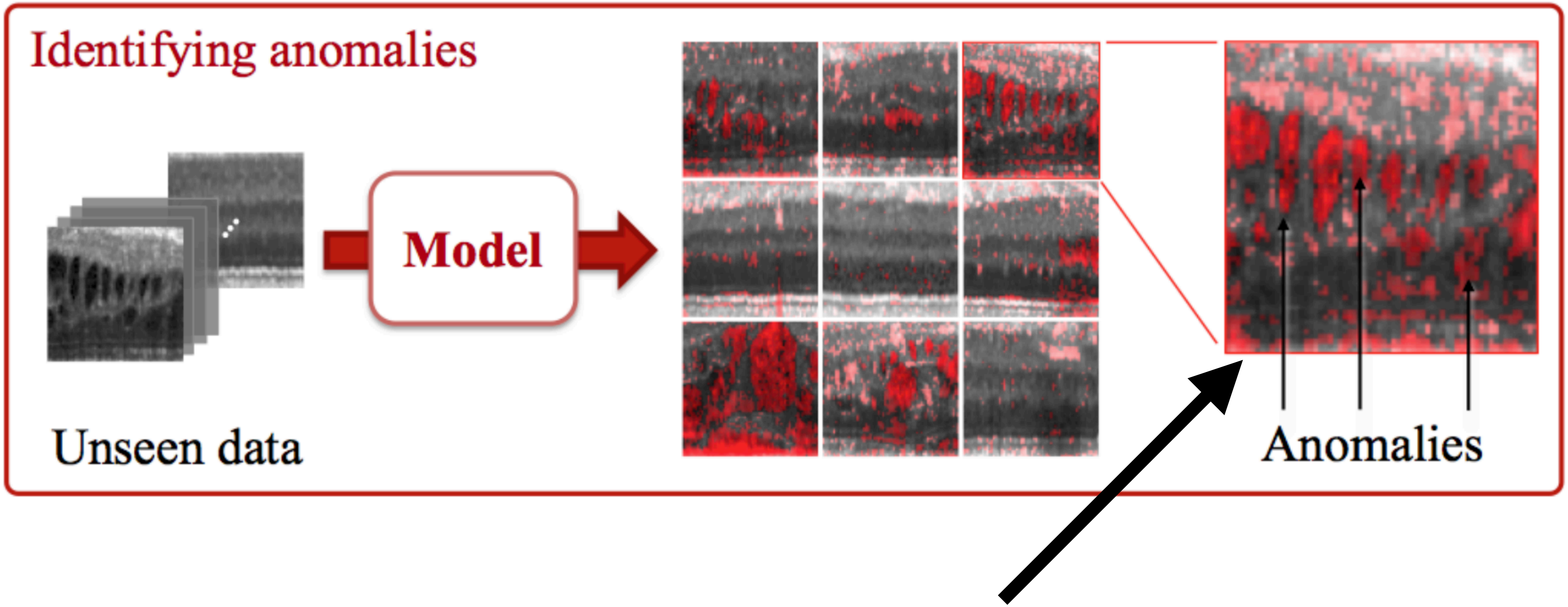
$$A(\mathbf{x}) = (1 - \lambda) \cdot R(\mathbf{x}) + \lambda \cdot D(\mathbf{x}),$$

Fig. 2. (a) Deep convolutional generative adversarial network. (b) t-SNE embedding of normal (blue) and anomalous (red) images on the feature representation of the last convolution layer (orange in (a)) of the discriminator.

Schlegl+17

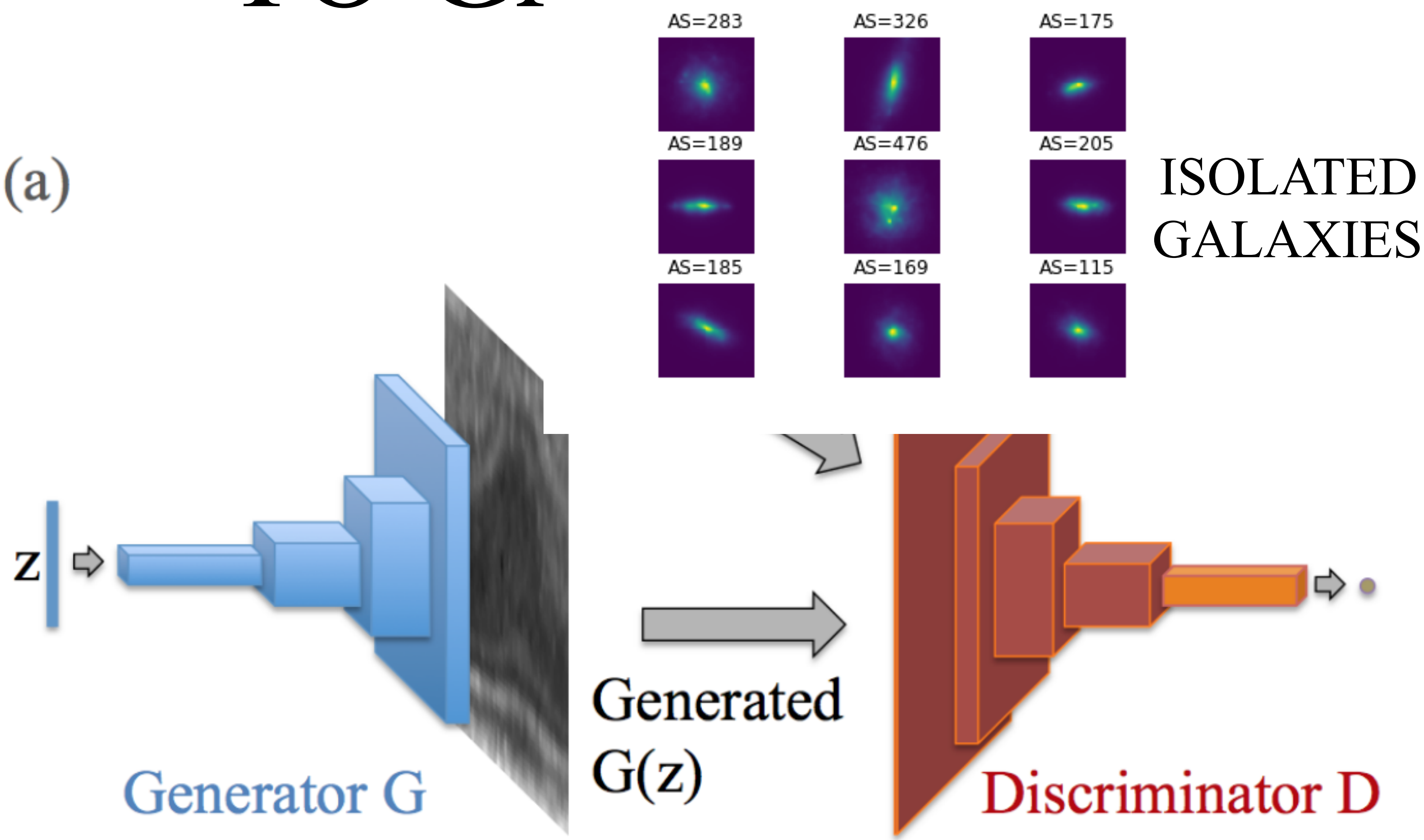
$$\mathbf{x}_R = |\mathbf{x} - G(\mathbf{z}_\Gamma)|$$

DIFFERENCE BETWEEN GENERATED FROM LATENT SPACE AND INPUT



IDENTIFY THE ANOMALOUS REGIONS IN THE IMAGE

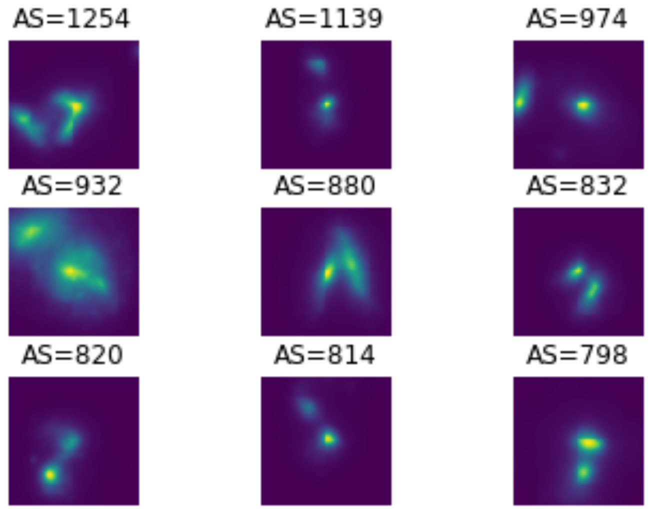
AN EXAMPLE APPLIED TO GALAXIES



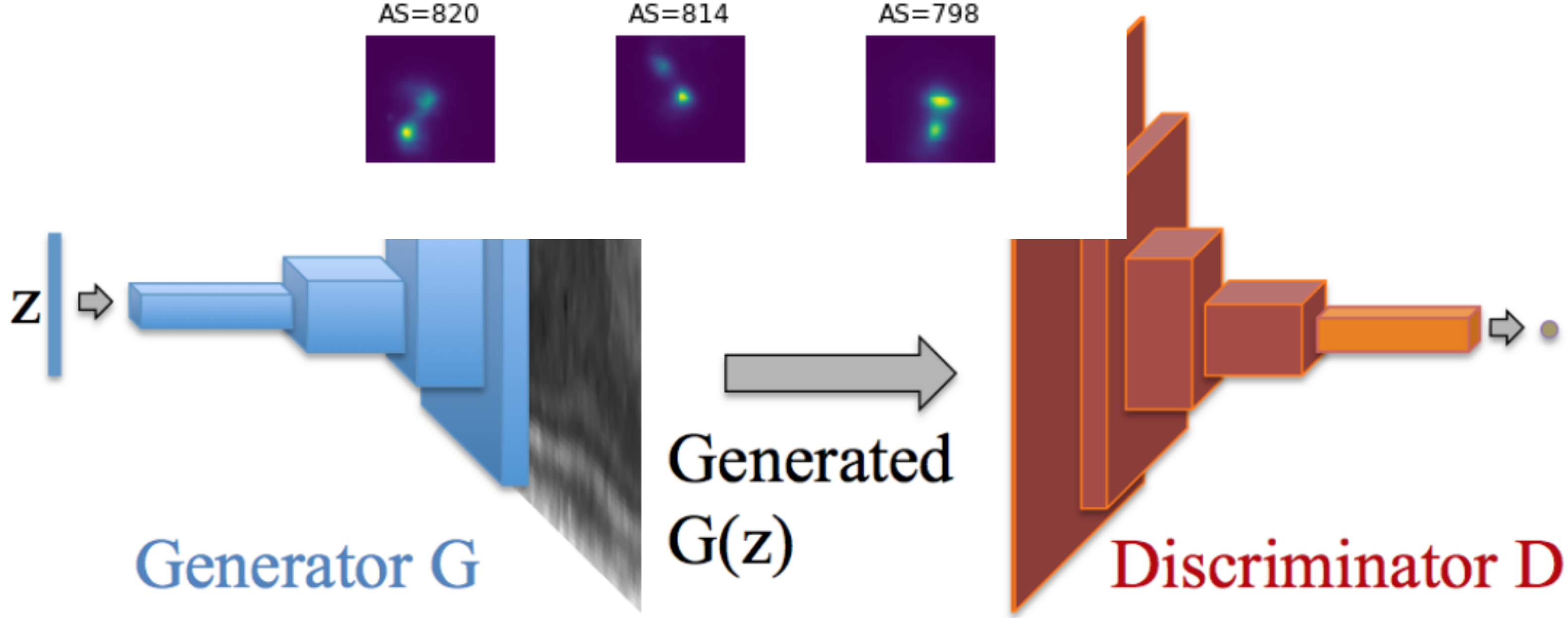
Margalef, MHC+19

AN EXAMPLE APPLIED TO GALAXIES

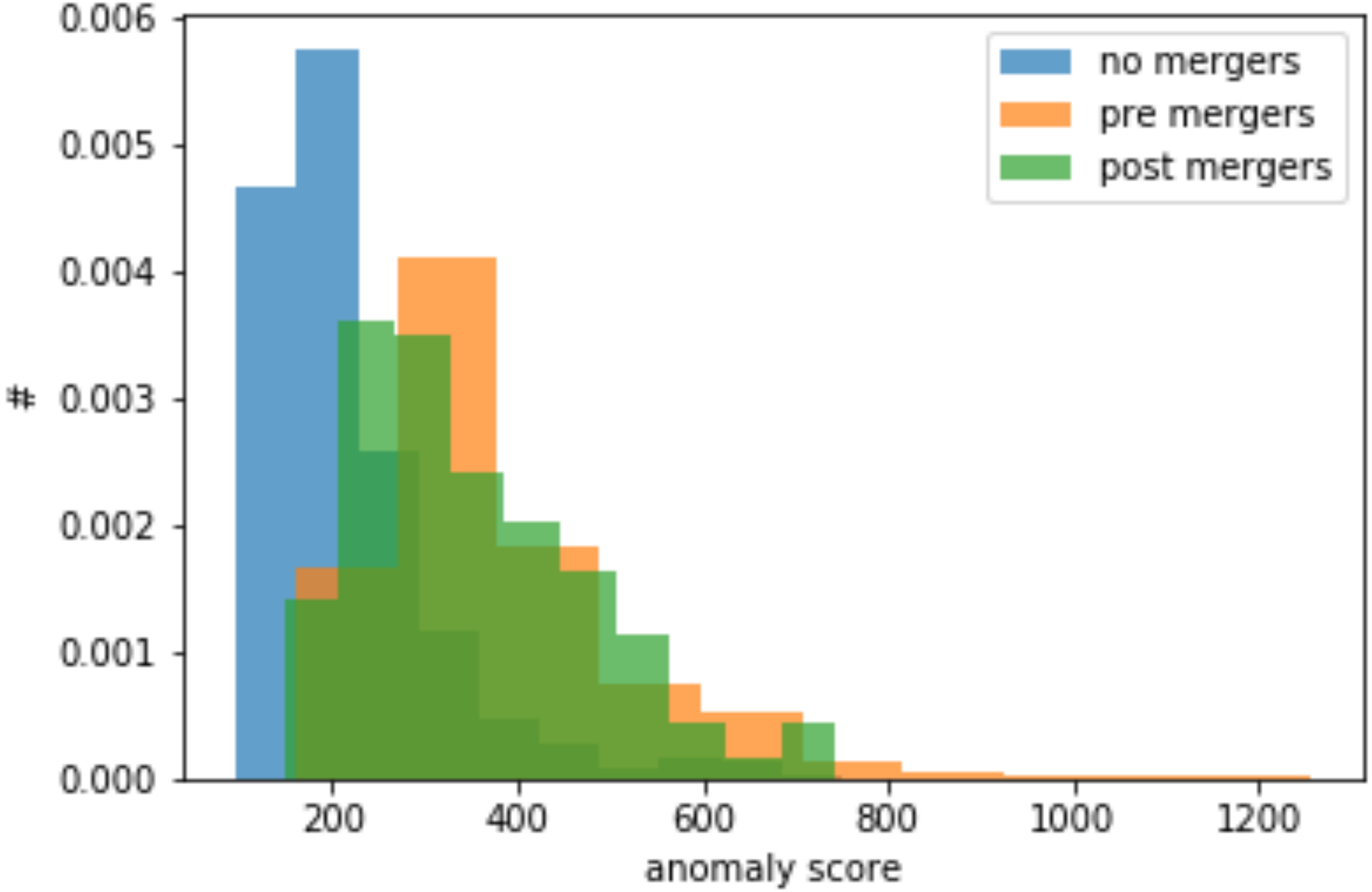
(a)



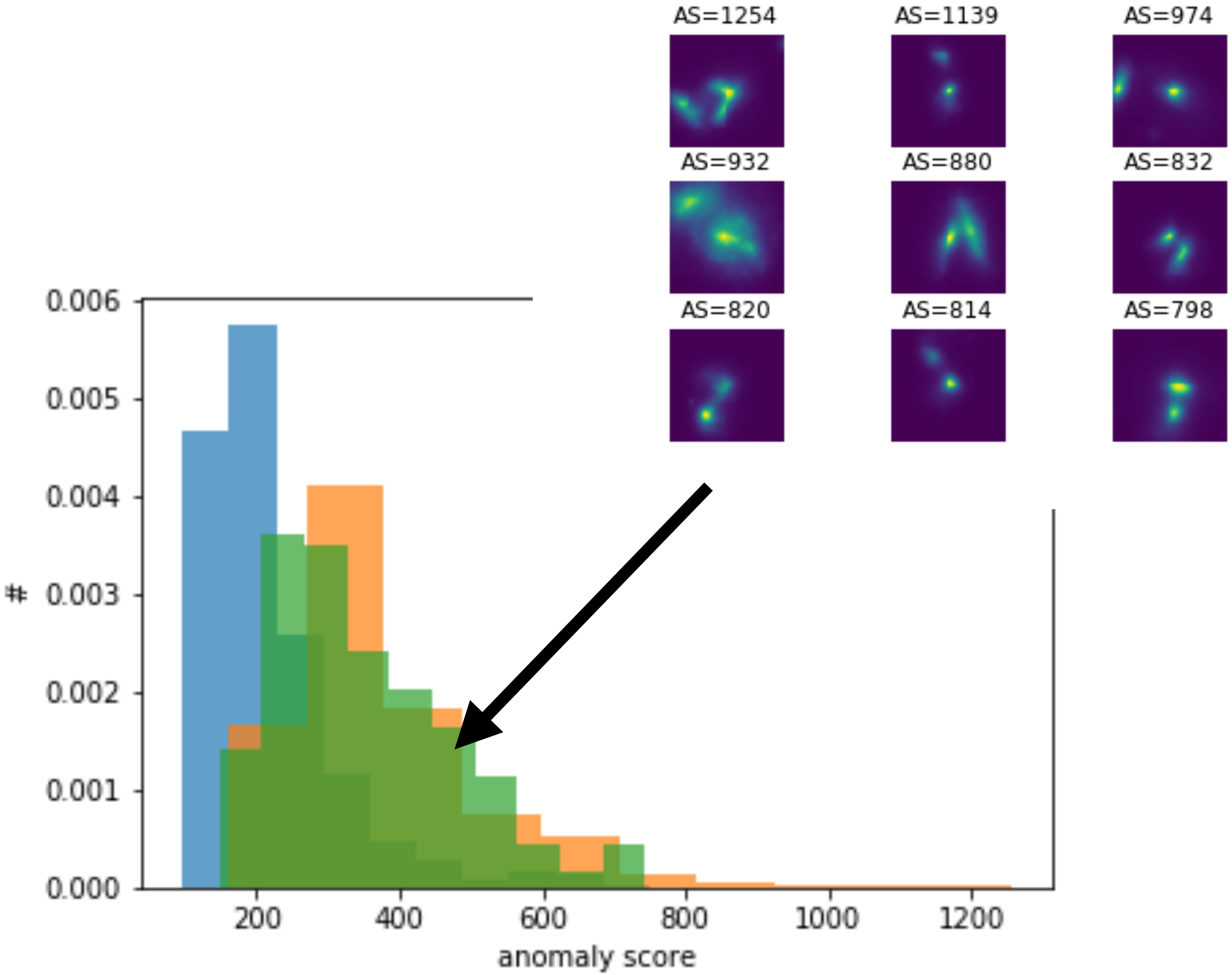
MERGERS



Margalef, MHC+19



no mergers
pre mergers
post mergers
0.006
0.005
0.004
0.003
0.002
0.001
0.000
#
200
400
600
800
1000
1200
anomaly score


AS=1254
AS=1139
AS=974
AS=932
AS=880
AS=832
AS=820
AS=814
AS=798
0.006
0.005
0.004
0.003
0.002
0.001
0.000
#
200
400
600
800
1000
1200
anomaly score

PART V: SOME PRACTICAL CONSIDERATIONS

HOW TO I KNOW THAT I HAVE REACHED CONVERGENCE?

FOR HOW MANY EPOCHS TO I HAVE TO TRAIN?

THERE IS NO “MAGIC RULE” TO MY KNOWLEDGE.

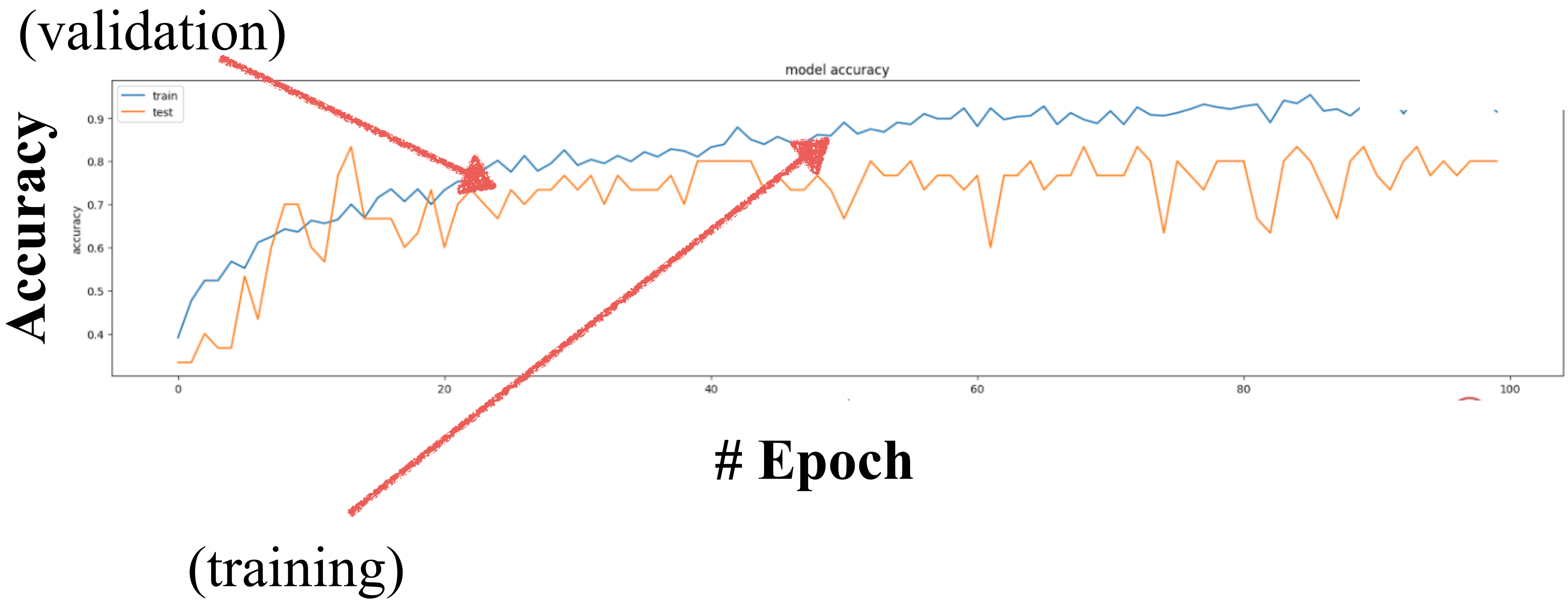
NORMALLY A DECISION IS TAKEN BASED
ON THE MONITORING OF THE VALIDATION LOSS

… “IF THE VALIDATION LOSS DOES NOT CHANGE MORE
THAN EPSILON OVER THE LAST N EPOCHS”…

KEEPING TRACK OF PERFORMANCE DURING TRAINING

THE LEARNING HISTORY

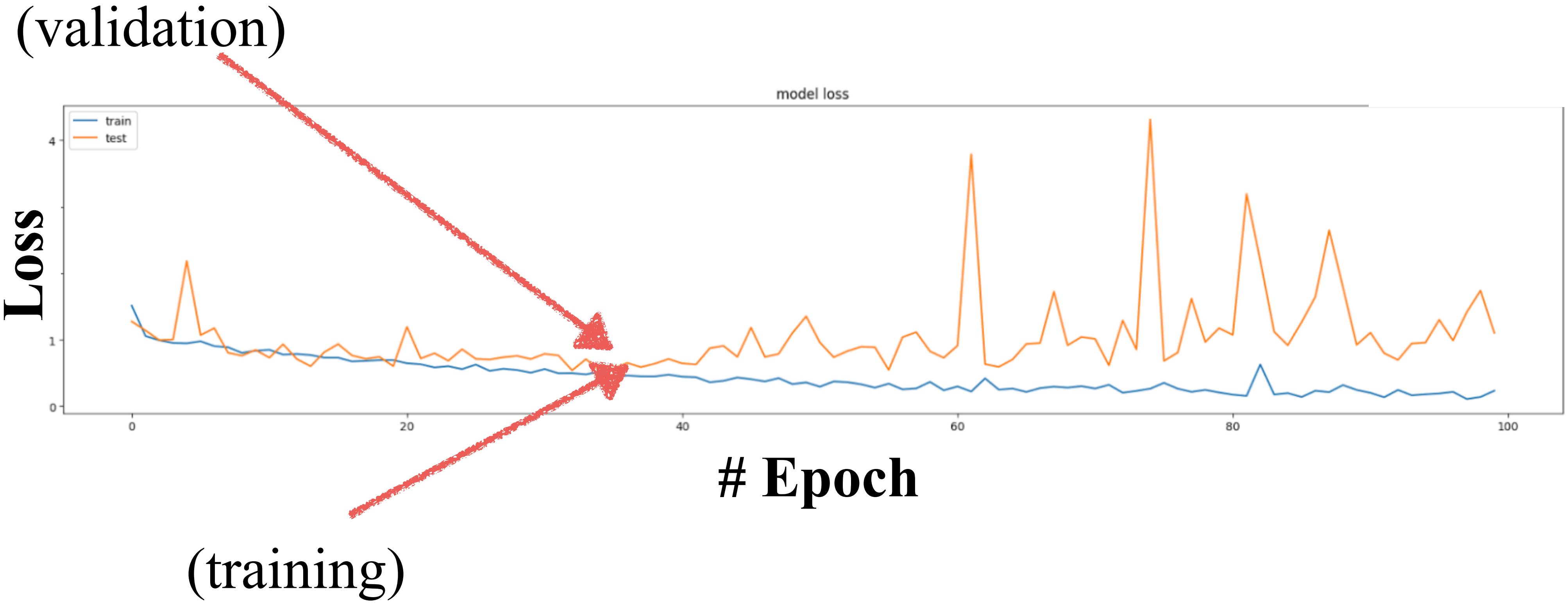
[MODEL ACCURACY AS A FUNCTION OF EPOCH]



KEEPING TRACK OF PERFORMANCE DURING TRAINING

THE LEARNING HISTORY

[MODEL LOSS AS A FUNCTION OF EPOCH]



IMPLEMENTATION IN KERAS

```
history = model.fit_generator(
            datagen.flow(X_train, Y_train, batch_size=batch_size),
            samples_per_epoch=X_train.shape[0],
            nb_epoch=nb_epoch,
            validation_data=(X_val, Y_val),
            callbacks=[earlystopping, modelcheckpoint]
        )
```

<u>THE TRAINING RETURNS A **HISTORY** OBJECT</u>

IMPLEMENTATION IN KERAS

THE TRAINING RETURNS A **HISTORY** OBJECT

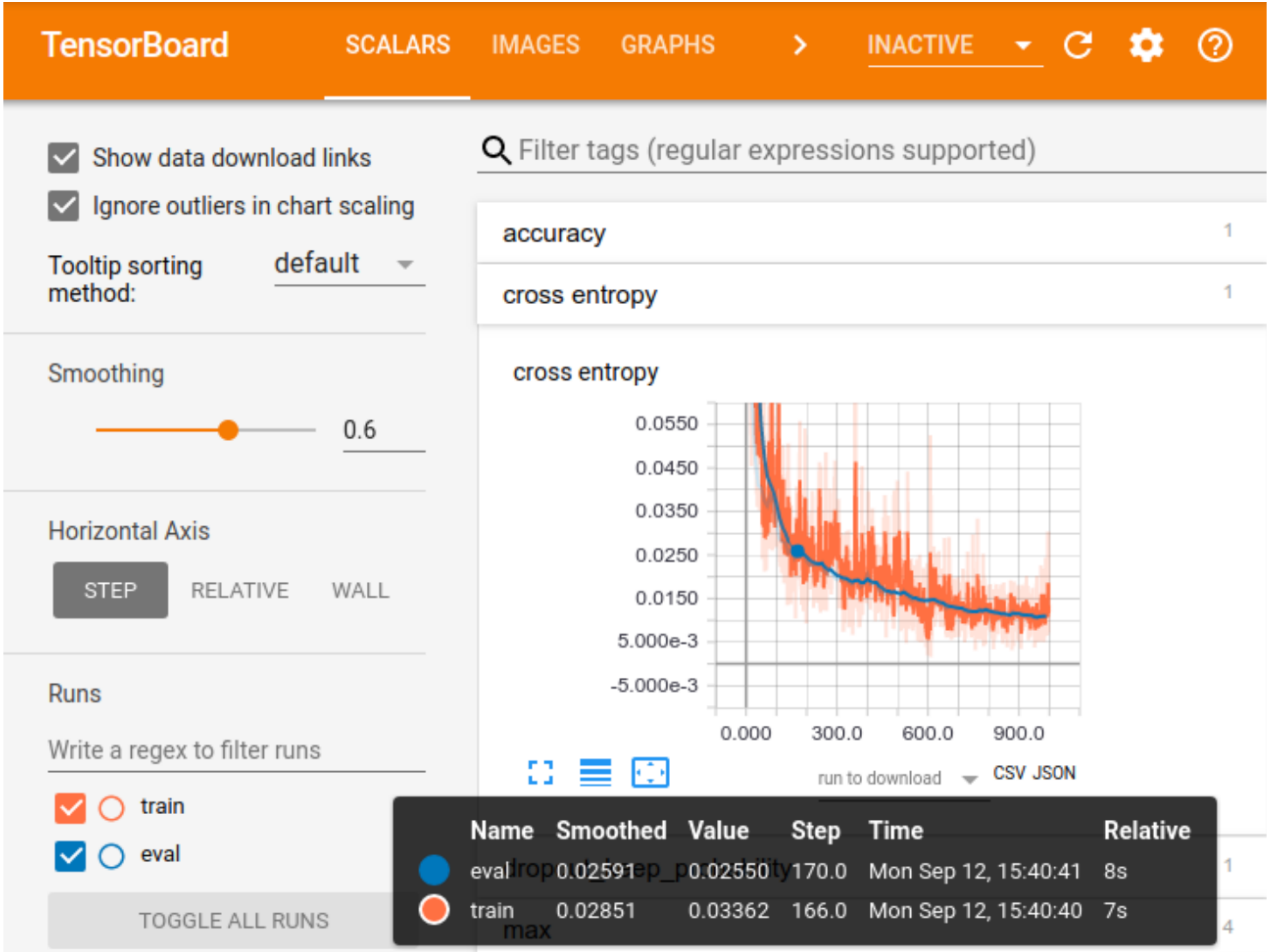
```
from keras.models import Sequential

print(history.history.keys())
['acc', 'loss', 'val_acc', 'val_loss']

plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
```

“LIVE” LEARNING HISTORY



EARLY STOPPING IN KERAS

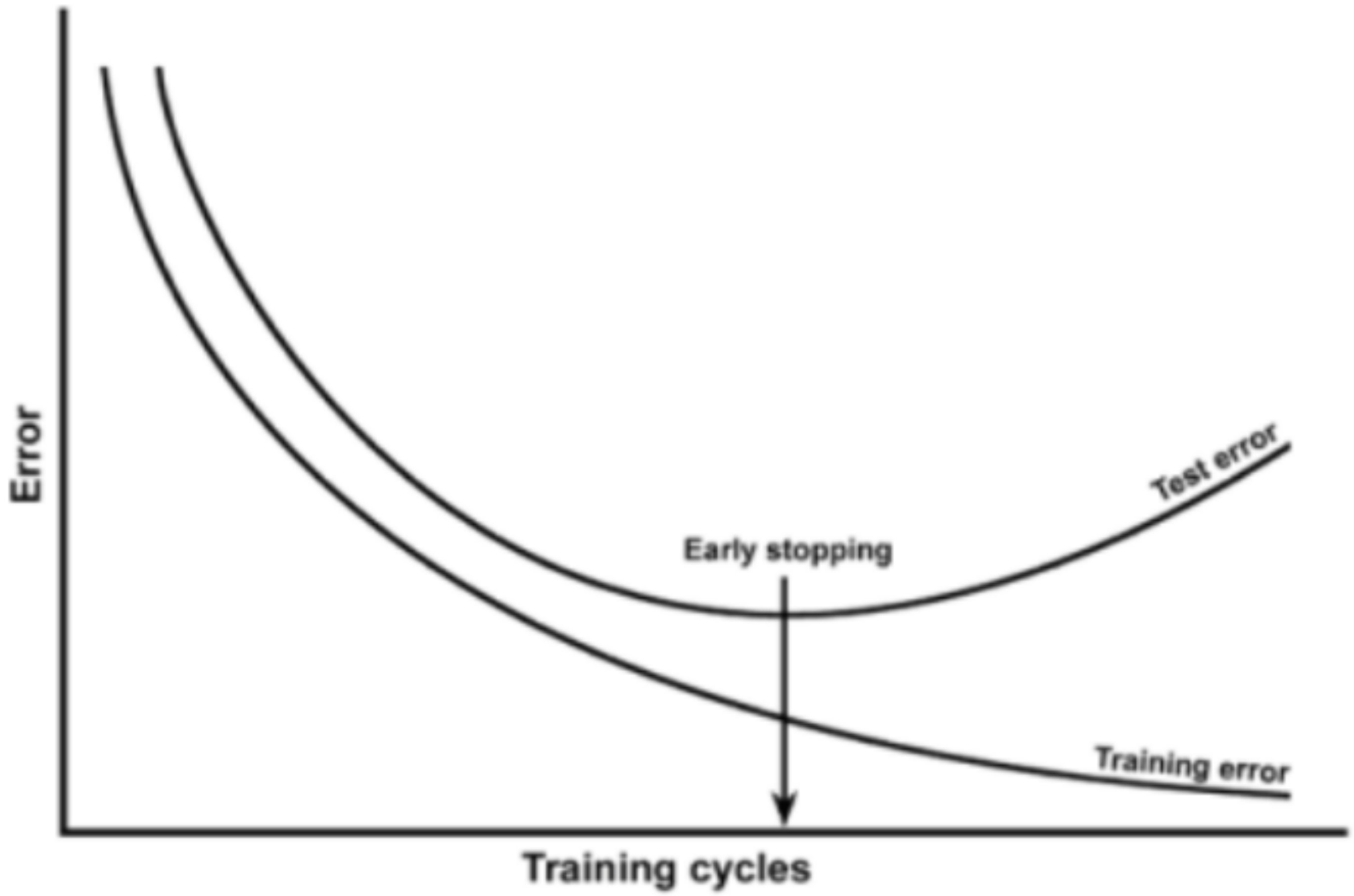
set patience (number of epochs to monitor)

```
from keras.callbacks import EarlyStopping

patience_par=10
earlystopping = EarlyStopping( monitor='val_loss',patience =
patience_par,verbose=0,mode='auto' )
history = model.fit(X, Y, validation_split=0.33, epochs=150, batch_size=10,
verbose=0, callbacks=[earlystopping])
```

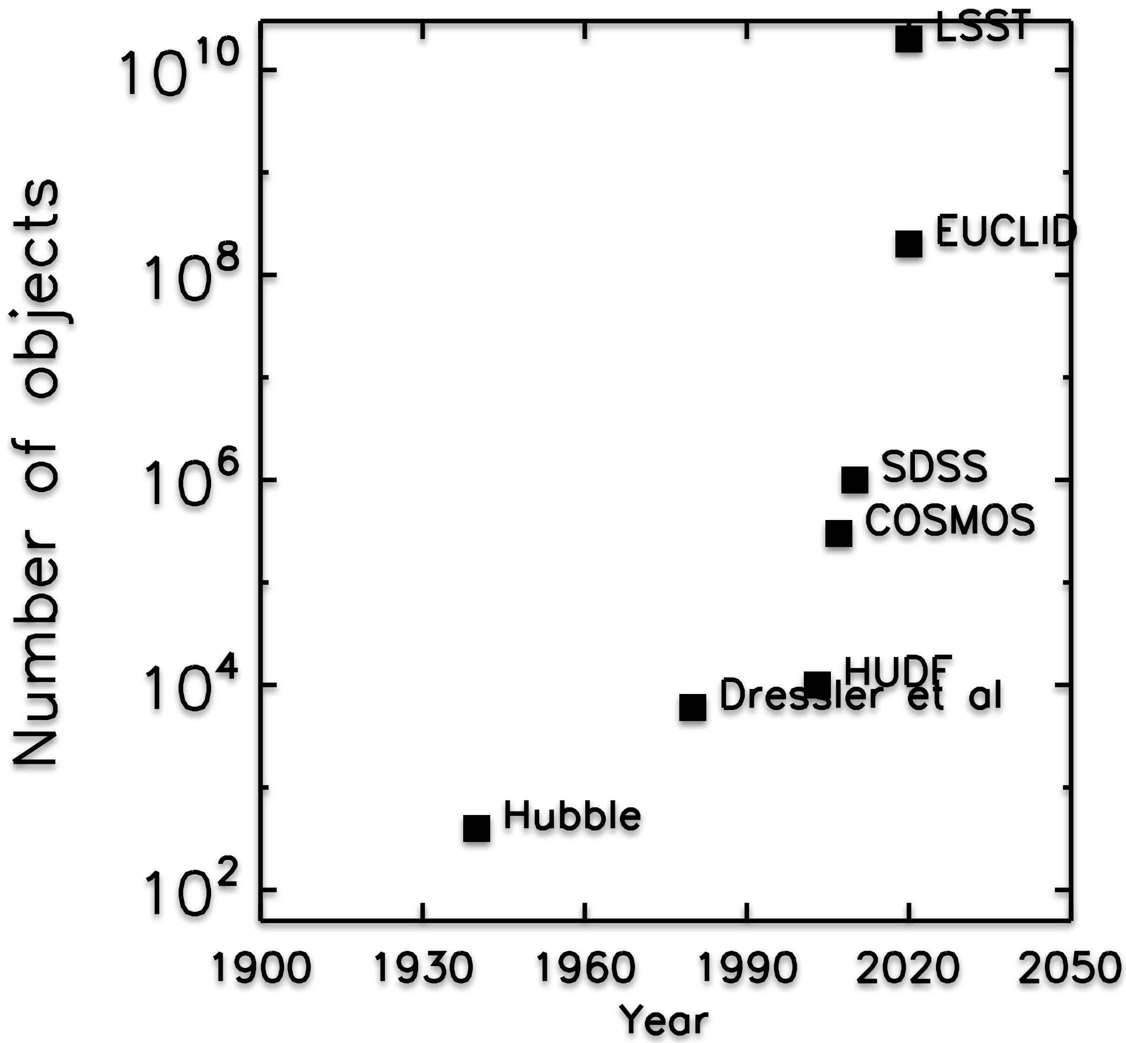
parameter to monitor [typically validation loss]

EARLY STOPPING HELPS ALSO PREVENTING OVERFITTING…



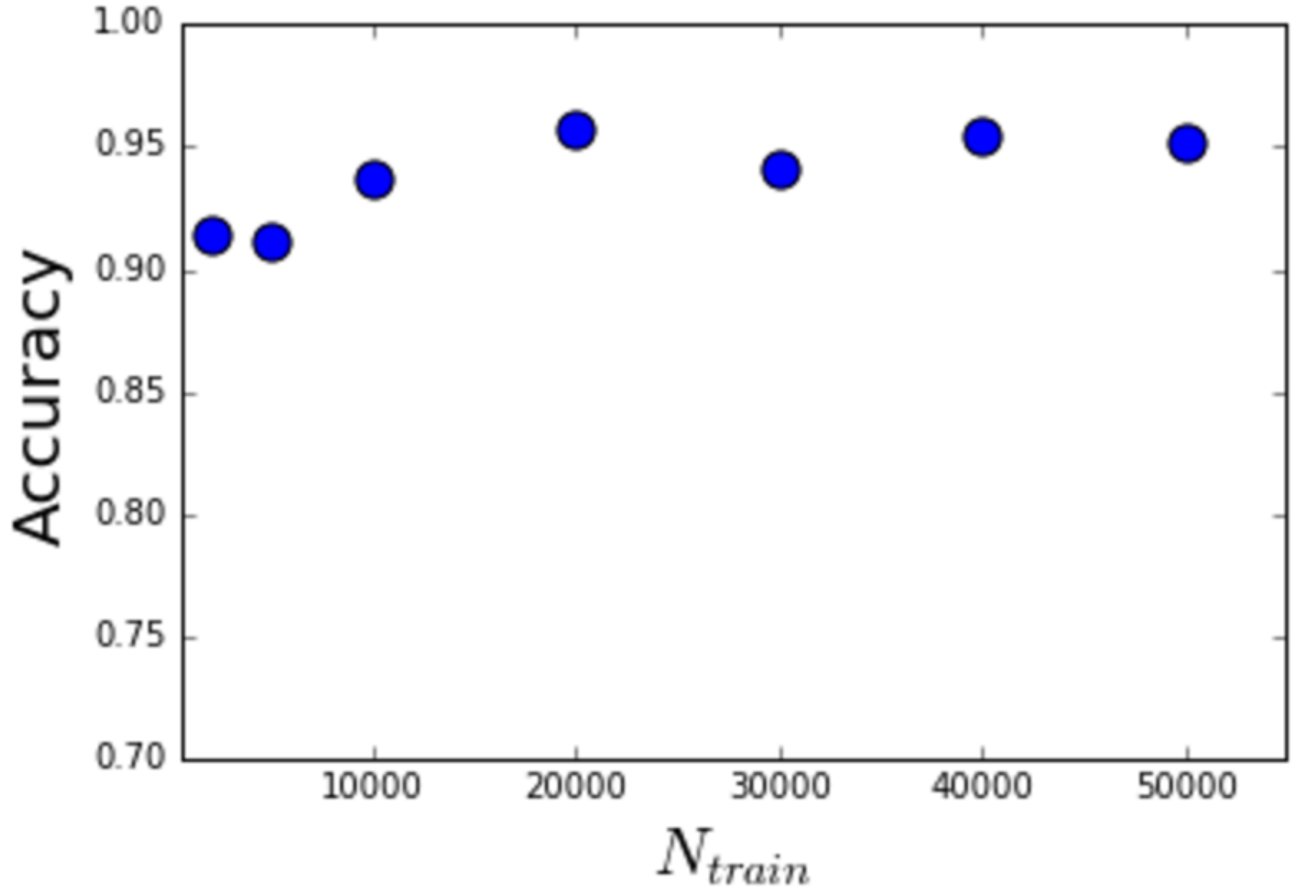
HOW LARGE DOES MY DATASET NEED TO BE?

DOES THAT MEAN THAT IF I DO NOT HAVE MILLIONS OF LABELLED EXAMPLES ALL THIS IS USELESS?



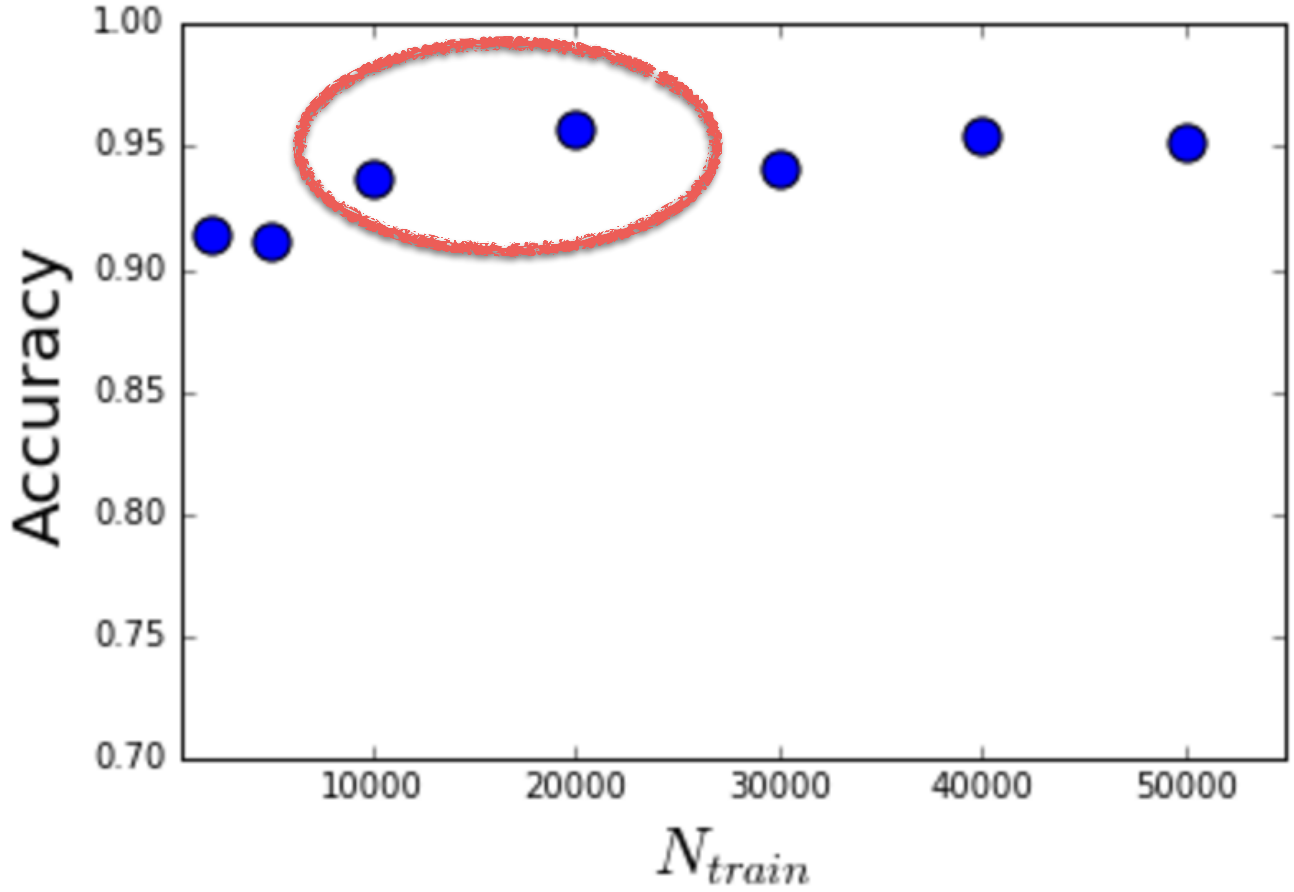
THE SIZE OF DATASETS
IS
INCREASING FAST BUT
WE DO
NOT ALWAYS HAVE
MILLIONS OF
LABELLED
EXAMPLES TO TRAIN
…

NOT REALLY…



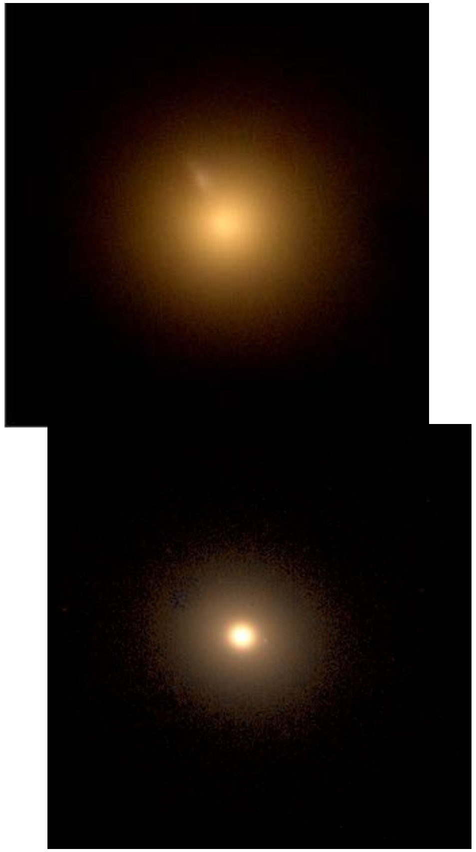
ACCURACY OF MORPHOLOGICAL CLASSIFICATIONS OF GALAXIES AS A FUNCTION OF THE SIZE OF THE TRAINING SET

NOT REALLY…



ACCURACY OF MORPHOLOGICAL CLASSIFICATIONS OF GALAXIES AS A FUNCTION OF THE SIZE OF THE TRAINING SET

WHY?



WHY?


ship dog deer bird ship cat dog dog
horse horse ship frog bird ship bird cat
automobile ship deer truck dog deer automobile horse

WHY?

IMAGES USED IN ASTROPHYSICS ARE SIMPLE…

SOME TRICKS FOR "SMALL" DATASETS

DATA AUGMENTATION

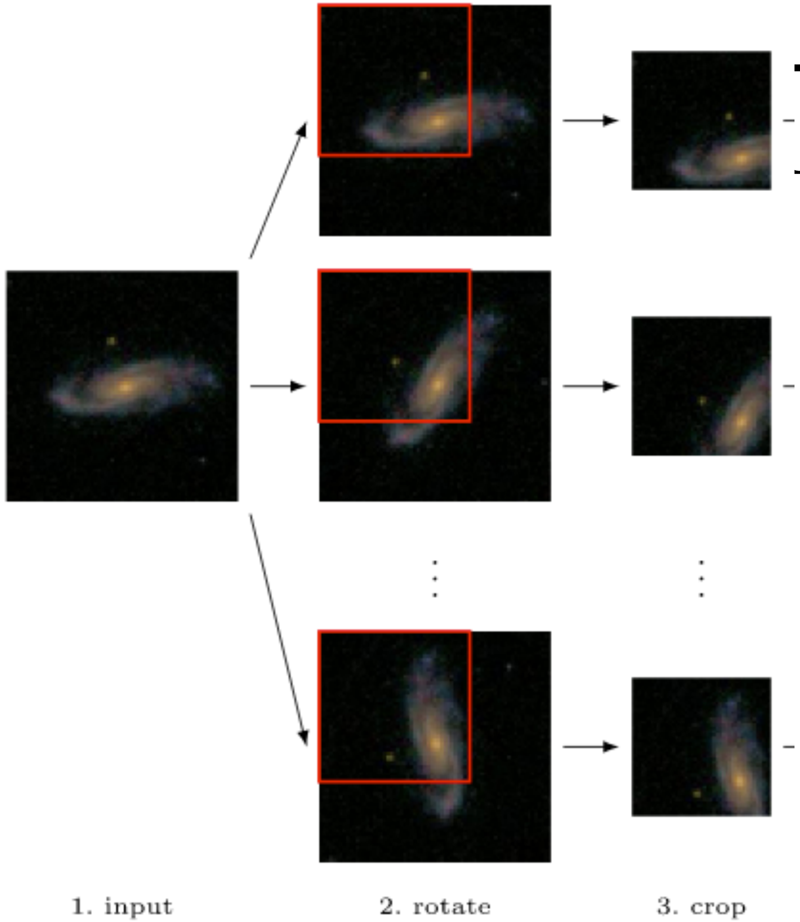
ANOTHER WAY TO REDUCE OVER-FITTING IS TO "AUGMENT" THE SIZE OF THE DATASET AVAILABLE FOR TRAINING

FOR MANY APPLICATIONS THE CLASSIFICATION SHOULD BE INDEPENDENT TO:

- TRANSALTIONS
- ROTATIONS
- SCALINGS

ETC…

DATA [...]TATION



Dieleman+15

FOR MANY APPLICATIONS THE CLASSIFICATION SHOULD BE INDEPENDENT TO:

- TRANSALTIONS
- ROTATIONS
- SCALINGS

ETC…

DATA AUGMENTATION

FOR MANY APPLICATIONS THE CLASSIFICATION SHOULD BE INDEPENDENT TO:

- TRANSALTIONS
- ROTATIONS
- SCALINGS

ETC…

DATA AUGMENTATION

CAN BE DONE "ON THE FLY" DURING THE TRAINING PHASE

KERAS IMPLEMENTATION:

DEFINES THE RANGE OF PERTURBATIONS APPLIED TO IMAGES DURING TRAINING

```
datagen = ImageDataGenerator(
    featurewise_center=False,
    samplewise_center=False,
    featurewise_std_normalization=False,
    samplewise_std_normalization=False,
    zca_whitening=False,
    rotation_range=45,
    width_shift_range=0.05,
    height_shift_range=0.05,
    horizontal_flip=True,
    vertical_flip=True,
    zoom_range=[0.75,1.3])
```

DATA AUGMENTATION

SOMETIMES ADDING NOISE INCREASES THE CLASSIFICATION / REGRESSION ACCURACY!

STANDARD DEVIATION OF GAUSSIAN NOISE

```
model = Sequential()
model.add(GaussianNoise(0.01,input_shape=(img_channels, img_rows, img_cols)))
```

A LAYER OF GAUSSIAN NOISE ADDED IN THE MODEL

DOMAIN ADAPTATION
(or knowledge transfer)

THE CONVOLUTIONAL PART OF A CNN IS
A FEATURE EXTRACTOR ….

DOMAIN ADAPTATION (or knowledge transfer)

THE CONVOLUTIONAL PART OF A CNN IS
A FEATURE EXTRACTOR

IN THAT RESPECT, THEY ARE VERY FLEXIBLE …

DOMAIN ADAPTATION
(or knowledge transfer)

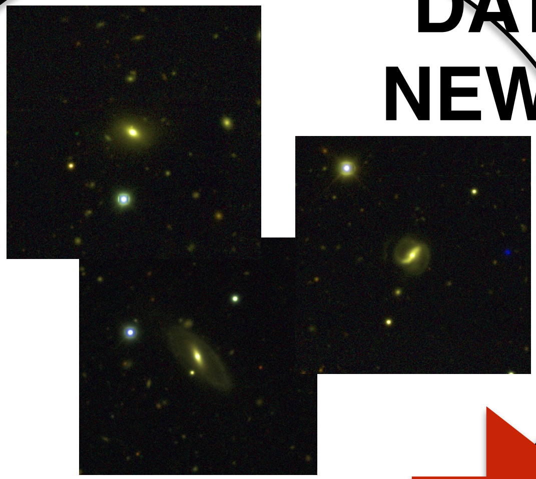
EVEN IF OUR TRAINING SET IS NOT SO LARGE …

WE CAN USE A CNN PRE-TRAINED ON A LARGER SAMPLE

DEPENDING ON HOW SIMILAR BOTH DATASETS ARE, WE CAN:

- RECYCLE THE SAME FEATURES
- FINE-TUNING THE WEIGHTS

DATA FROM NEW SURVEY



How robust to different datasets? Do we always need a big training set?

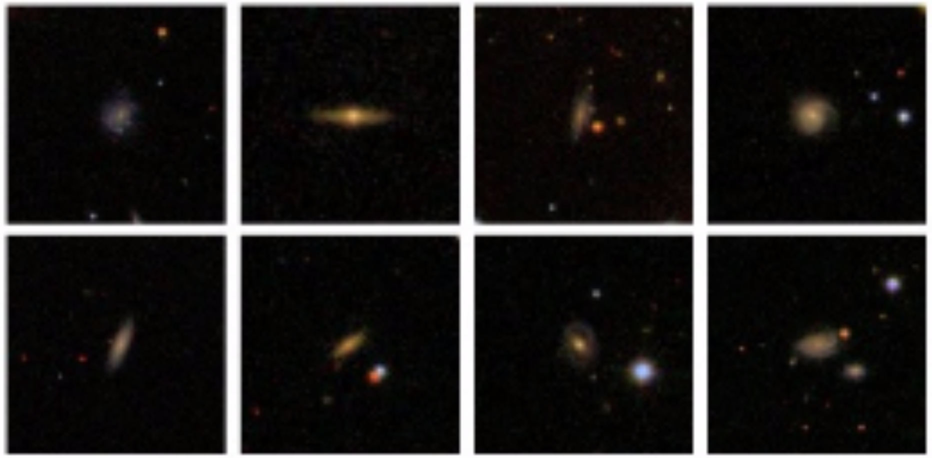
DEEP-LEARNING BASED MACHINE

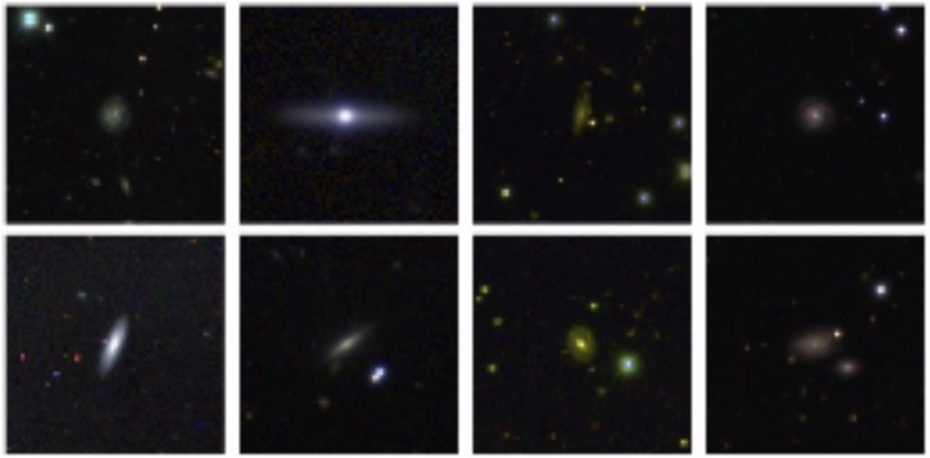
"Improved" Galaxy ZOO like classifications for for the entire sample

Transfer knowledge?

Human classifications from existing survey

Dominguez-Sanchez+18b

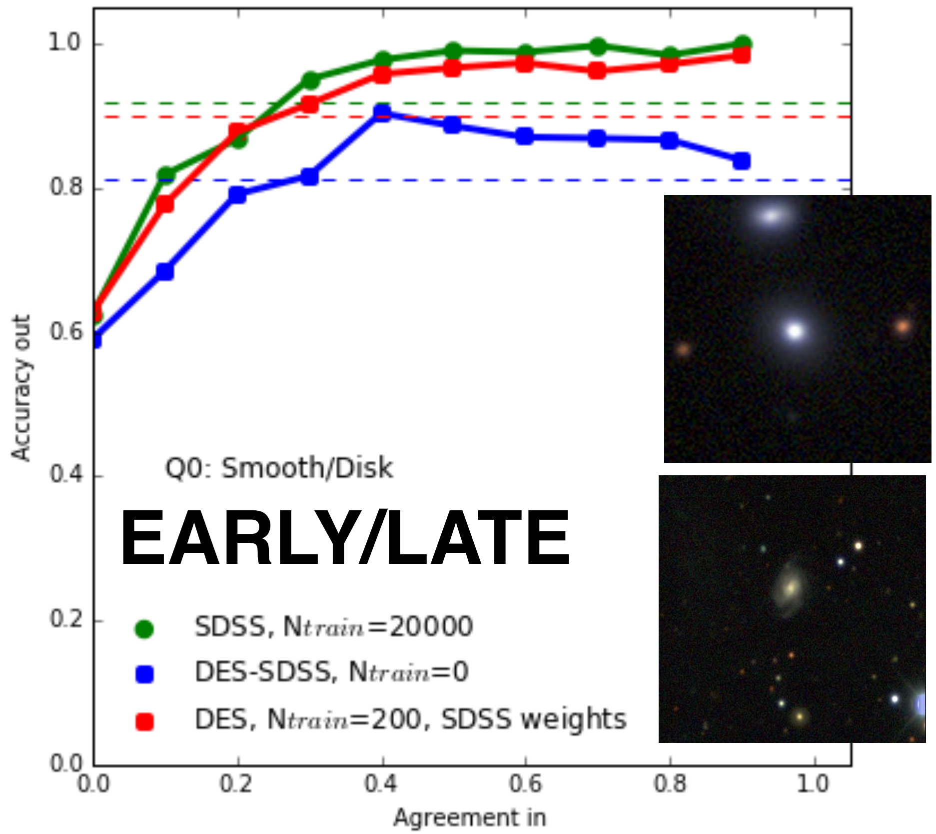


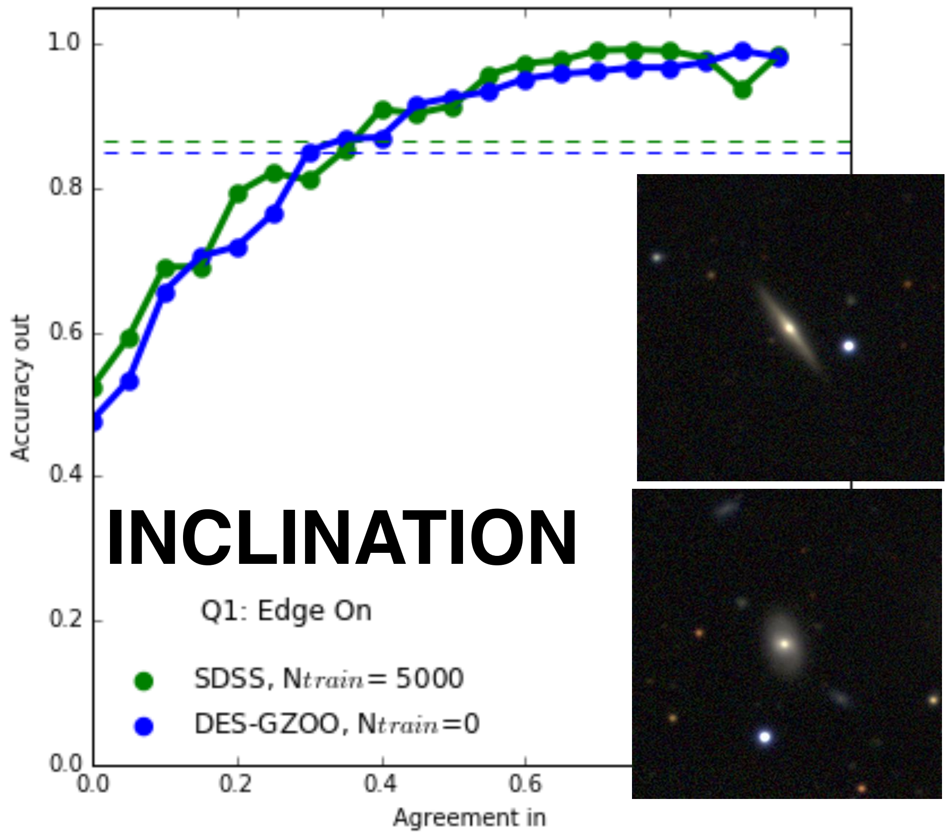


SDSS **DES**

Dominguez-Sanchez+18b

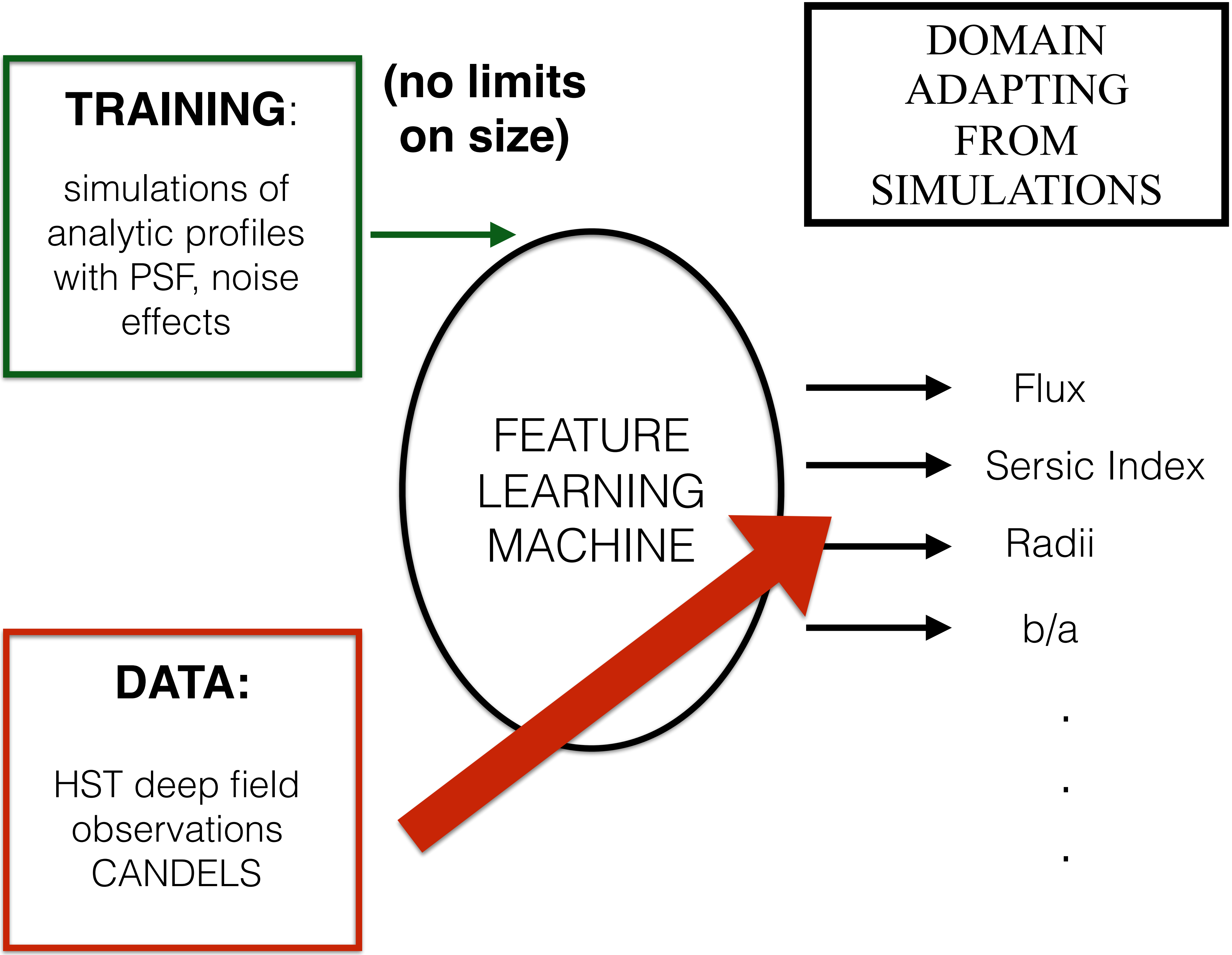
Knowledge Transfer from SDSS to DES

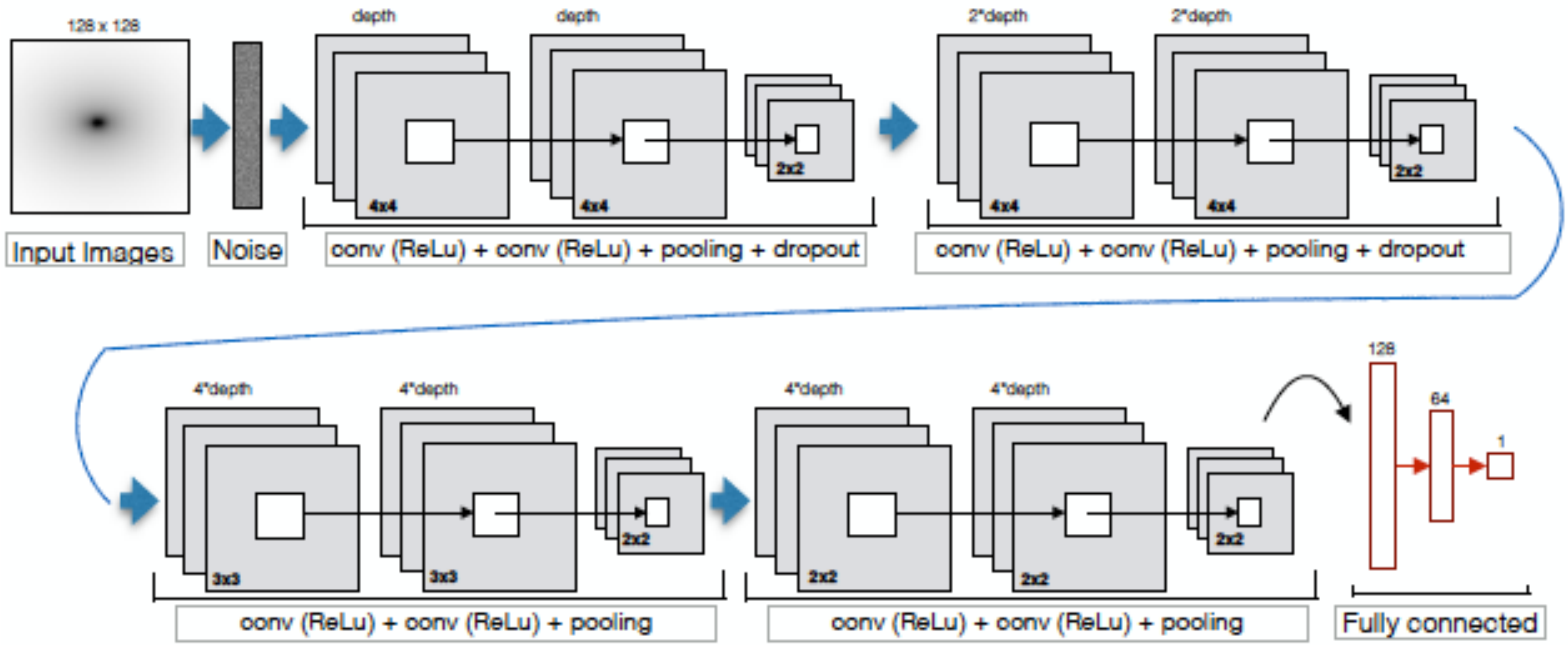




Only 200 (**1%!**) objects classified in DES are needed to reach an accuracy >90% if a machine trained on the SDSS is used

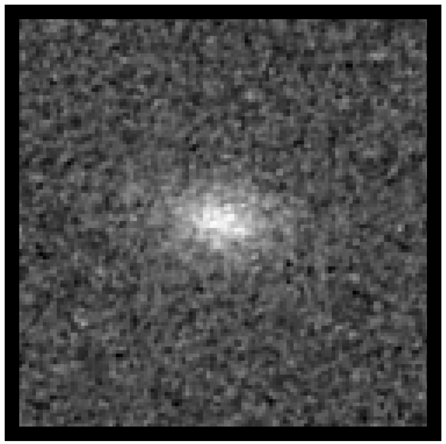
For some properties, i.e. EDGE-ON galaxies. No training at all is needed to go from SDSS to DES

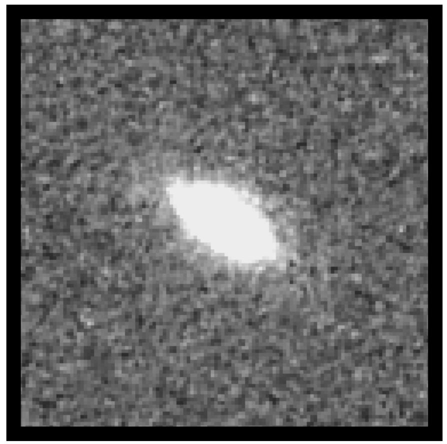

TRAINING:
simulations of analytic profiles with PSF, noise effects
(no limits on size)
DOMAIN ADAPTING FROM SIMULATIONS
FEATURE LEARNING MACHINE
Flux
Sersic Index
Radii
b/a
.
.
.
DATA:
HST deep field observations CANDELS


128 x 128
Input Images
Noise
depth
depth
4x4
4x4
2x2
conv (ReLu) + conv (ReLu) + pooling + dropout
2*depth
2*depth
4x4
4x4
2x2
conv (ReLu) + conv (ReLu) + pooling + dropout
4*depth
4*depth
3x3
3x3
2x2
conv (ReLu) + conv (ReLu) + pooling
4*depth
4*depth
2x2
2x2
2x2
conv (ReLu) + conv (ReLu) + pooling
128
64
1
Fully connected

Standard analytic profiles

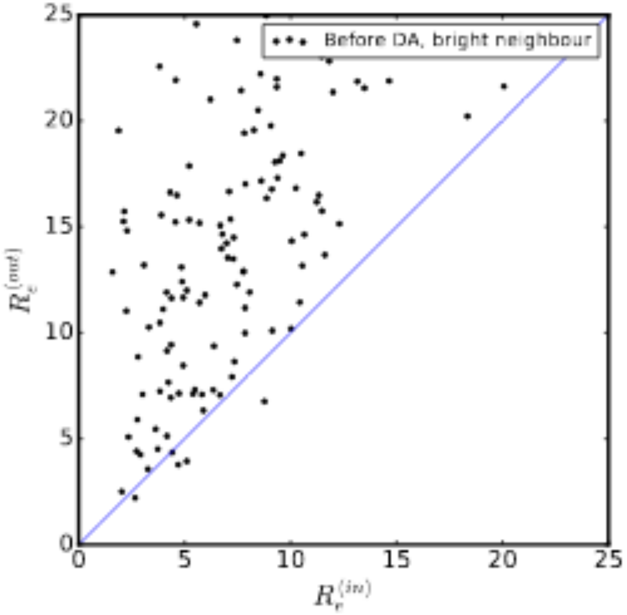
- 100.000-300.000 galaxies [GALSIM]
  - Real HST background added + PSF (F160)
  - Random distribution of parameters (uniform):
    - 18<Mag<24, 0<BT<1, <Nb<, Nd=1, 0.2<log(rb)<1.3, 0.2<log(rd)<1.5, 0.05<eb<0.95, 0.05<ed<0.95, 0<PA<180
  - 64*64 stamps
  - **FULLY IDEALISTIC -**
    NO COMPANIONS
    NO IRREGULARS
    NO CLUMPY!





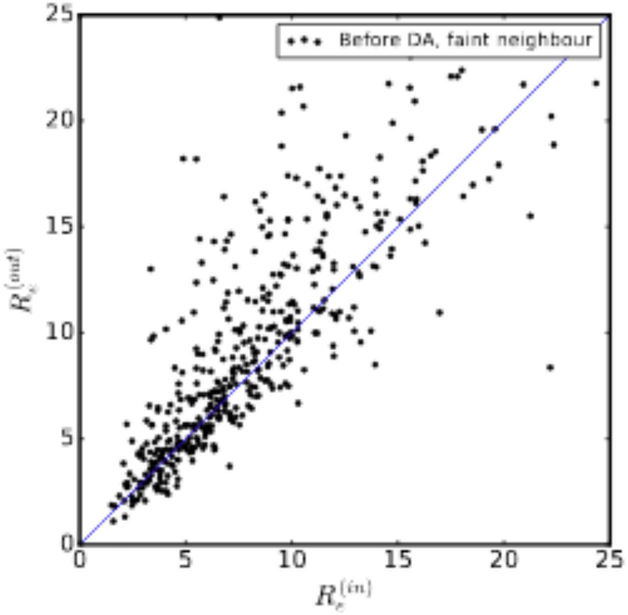
MORPHOMETRY OF REAL GALAXIES TRAINED ON ANALYTIC PROFILES

BRIGHT NEIGH.



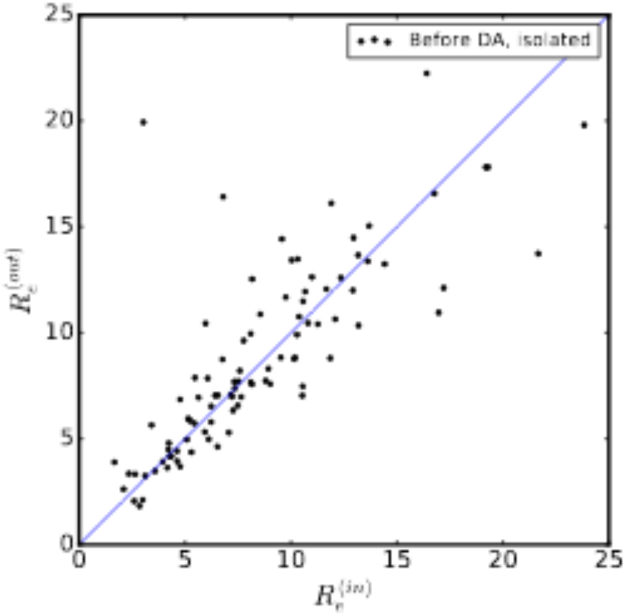
(c) BDA bright neigbours

FAINT NEIGH.



(d) BDA faint neigbours

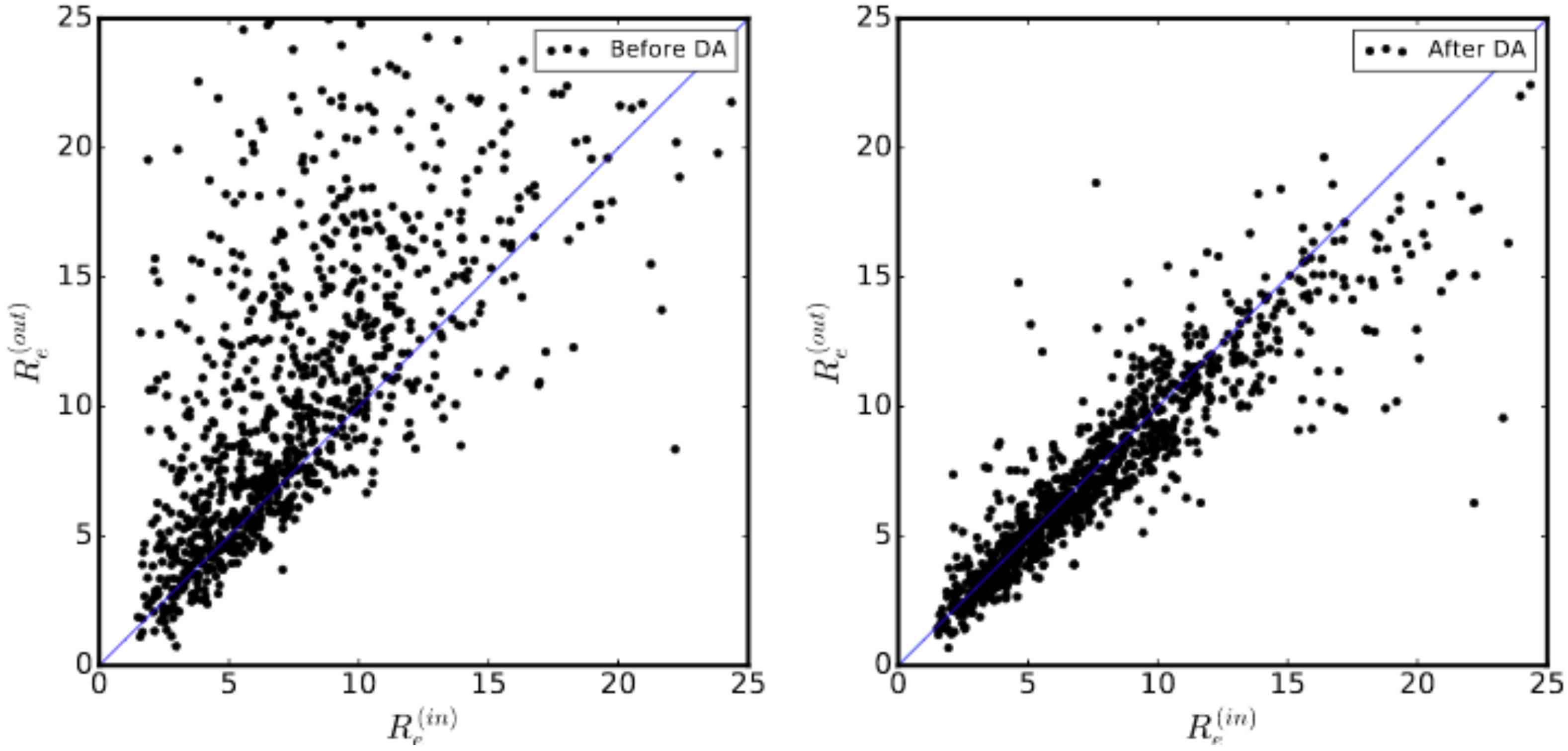
ISOLATED



(e) BDA isolated galaxies

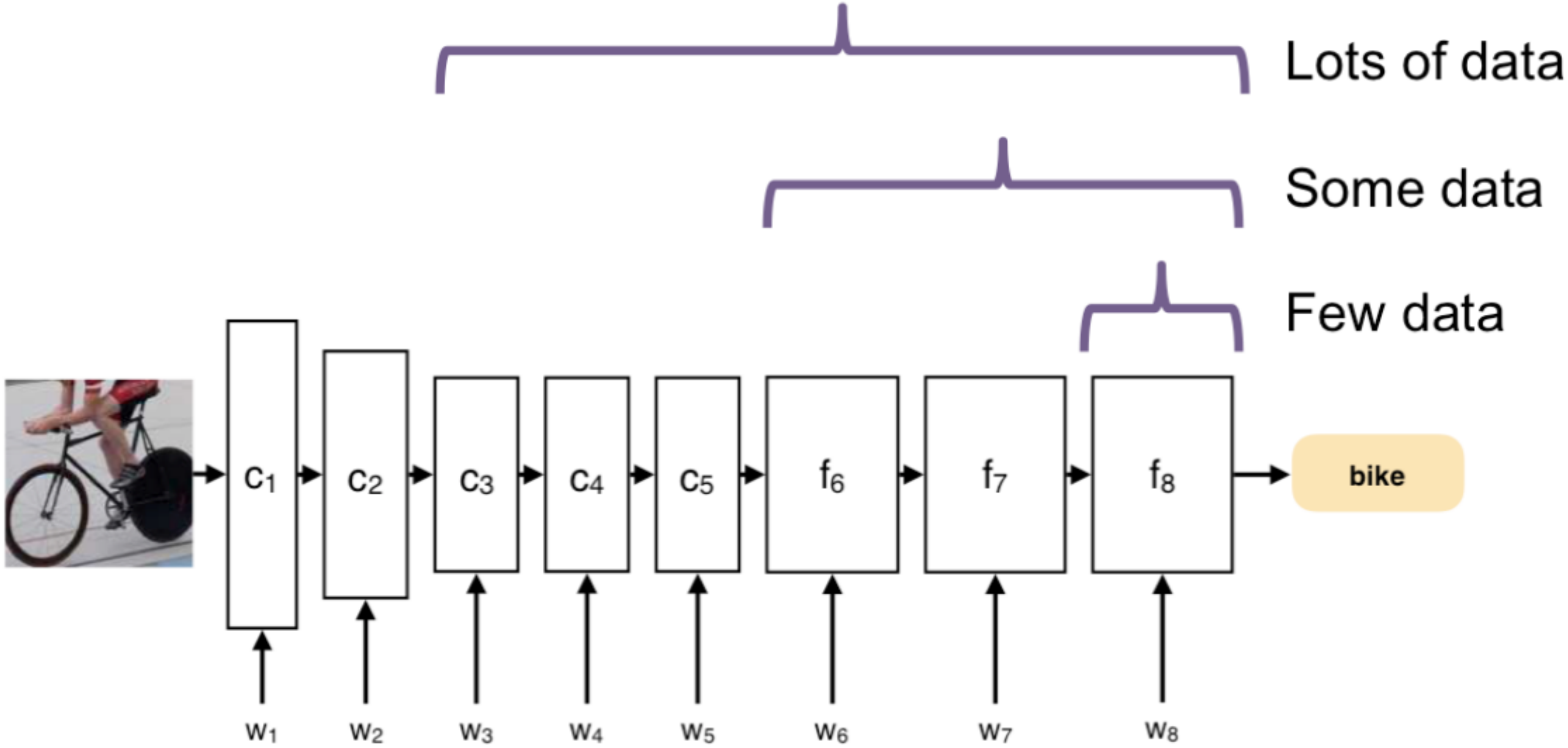
Tuccillo+18

DOMAIN ADAPTATION: 0.1% OF "REAL" GALAXIES



Tuccillo+18

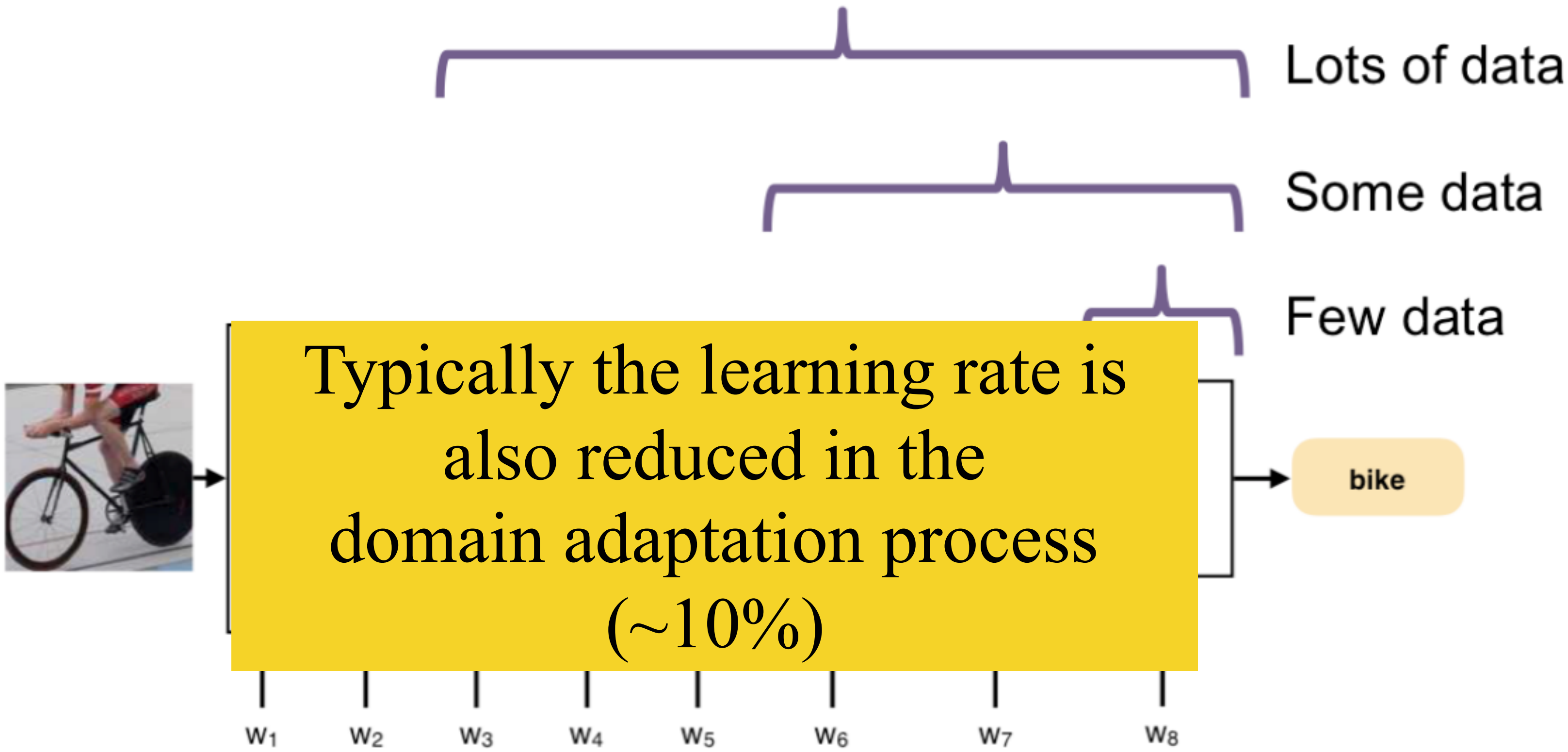
HOW MANY LAYERS "DOMAIN ADAPT" ?



DEPENDING ON HOW MUCH SIMILAR BOTH DATASETS ARE:

- FINE-TUNE ONLY FULLY CONNECTED LAYERS
- FINE-TUNE A FEW LAYERS
- FINE-TUNE ALL LAYERS

HOW MANY LAYERS “DOMAIN ADAPT” ?



DEPENDING ON HOW MUCH SIMILAR BOTH DATASETS ARE:

- FINE-TUNE ONLY FULLY CONNECTED LAYERS
- FINE-TUNE A FEW LAYERS
- FINE-TUNE ALL LAYERS

KERAS IMPLEMENTATION

"FULL" DOMAIN ADAPTATION

learning rate

```
model.save_weights(test_name+".hd5",overwrite=True)
```

```
keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
```

```
model.load_weights(model_name+".hd5")
```

start training

load weights of previous training

```
history = model.fit_generator(
            datagen.flow(X_train, Y_train, batch_size=batch_size),
            samples_per_epoch=X_train.shape[0],
            nb_epoch=nb_epoch,
            validation_data=(X_val, Y_val),
            callbacks=[earlystopping, modelcheckpoint]
        )
```

KERAS IMPLEMENTATION

"PARTIAL" DOMAIN ADAPTATION

```
model.save_weights(test_name+".hd5",overwrite=True)
```

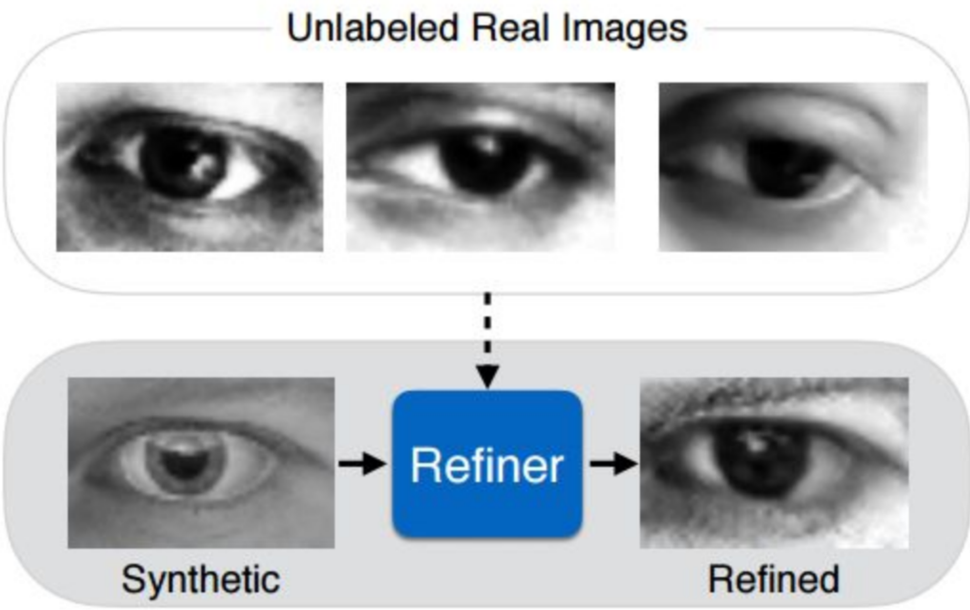
```
keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
```

```
model.load_weights(model_name+".hd5")
```

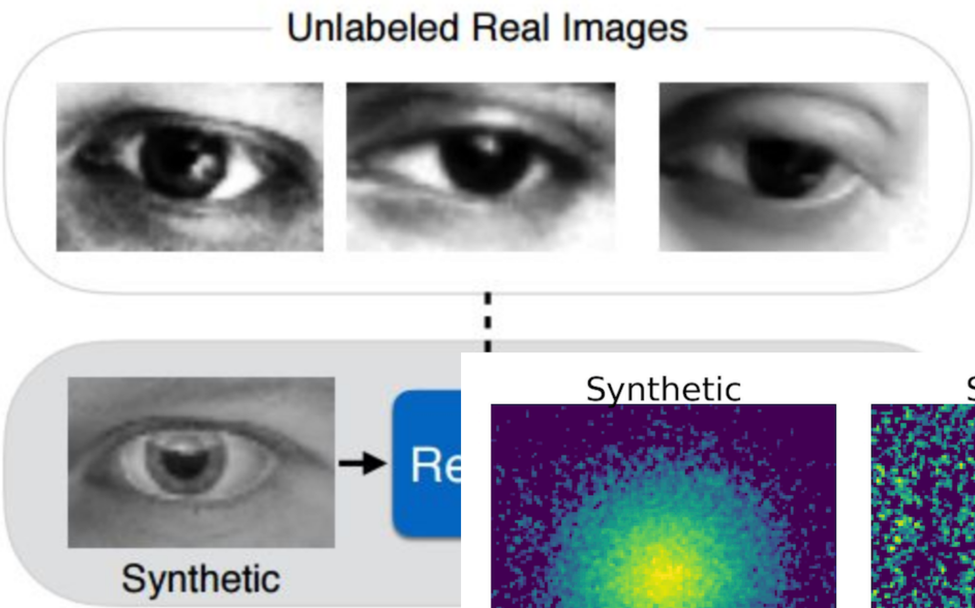
```
all_layers = model.layers
for i in range(model.layers.index(mid_start)):
    all_layers[i].trainable = False
```

Freeze some layers

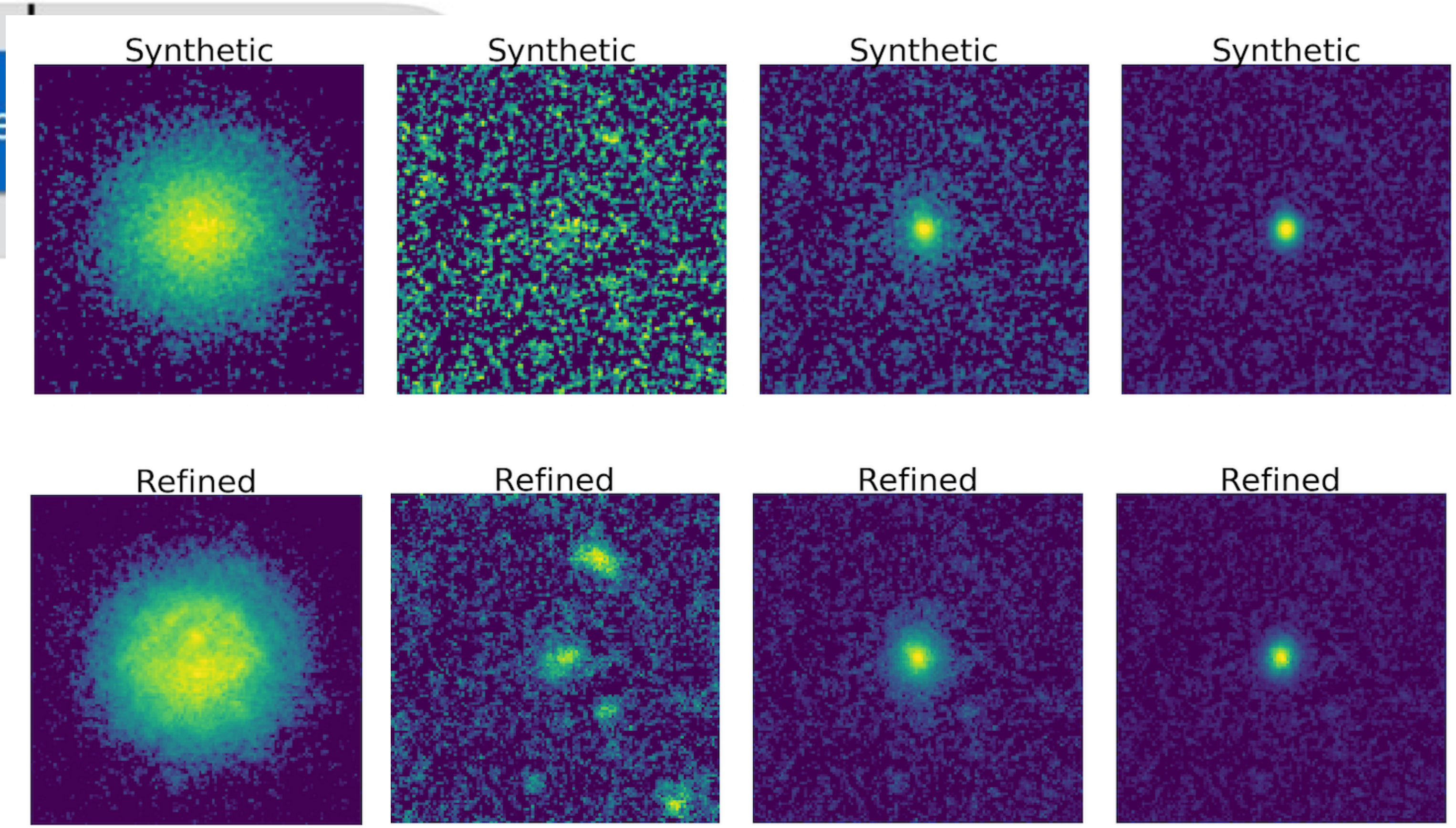
USE GANs?



USE GANs?



VERY PRELIMINAR WORK IN CANDELS



MARGALEF,MHC+ in prep

HOW DO I SETUP MY CNN? HYPER-PARAMATER SEARCH

UNFORTUNATELY, THERE IS NO MAGIC RECIPE

HOW DO I SETUP MY CNN?

UNFORTUNATELY, THERE IS NO MAGIC RECIPE

SOME GENERAL ADVICES:

- START WITH SOMETHING THAT WORKS
- CAPACITY INCREASES WITH LAYERS, CHANNELS, RECEPTIVE FIELD
- USE PRIOR KNOWLEDGE OF THE EXPECTED SCALE CONTAINING MEANINGFUL INFORMATION
- GRID SEARCH…
- AVERAGING

CREDIT

HOW DO I SETUP MY CNN?

TYPICAL PARAMETERS

NUMBER OF HIDDEN LAYERS?

- START WITH FEW LAYERS AND INCREASE COMPLEXITY IF NEEDED
- ADD MORE LAYERS, CHECK PERFORMANCE
- ADD MORE NEURONS, CHECK PERFORMANCE

CREDIT

HOW DO I SETUP MY CNN?

TYPICAL PARAMETERS

ACTIVATION FUNCTION

- PRIORITY TO ReLU
- TRY EVENTUALLY LeakyReLu, PreLU

CREDIT

HOW DO I SETUP MY CNN?

TYPICAL PARAMETERS

OPTIMIZER

- NOT REALLY A RULE FOR THIS…
- SGD, ADAM THE ONES I MOSTLY USED. ADAM MORE ROBUST TO "NAN" LOSSES

CREDIT

HOW DO I SETUP MY CNN?

TYPICAL PARAMETERS

LEARNING RATE

- THIS IS ONE OF THE MOST TWEAKED PARAMETERS
- THE LEARNING RATE SHOULD BE LARGE AT THE BEGINNING AND SMALL TOWARDS THEN WHEN CLOSER TO THE MINIMUM

CREDIT

HOW DO I SETUP MY CNN?

TYPICAL PARAMETERS

LEARNING RATE

- THIS IS ONE OF THE MOST TWEAKED PARAMETERS

- THE LEARNING RATE SHOULD BE LARGE AT THE BEGINNING AND SMALL TOWARDS THEN WHEN CLOSER TO THE MINIMUM

CREDIT

HOW DO I SETUP MY CNN?

LEARNING RATE

- THE DECAY OF THE LEARNING RATE CAN BE SET :
  - STEP DECAY [DECAY BY FIX AMOUNT EVERY FEW EPOCHS]
  - EXPONENTIAL DECAY $\lambda = \lambda_0 e^{-kt}$
  - INVERSE DECAY $\lambda = \dfrac{\lambda_0}{1 + kt}$

CREDIT

HOW DO I SETUP MY CNN?

LEARNING RATE

- THE DECAY OF THE LEARNING RATE CAN BE SET :
  - STEP DECAY [DECAY BY FIX AMOUNT EVERY FEW EPOCHS]
  - EXPONENTIAL DECAY $\lambda = \lambda_0 e^{-kt}$
  - INVERSE DECAY $\lambda = \dfrac{\lambda_0}{1 + kt}$

CREDIT

- WHEN OPTIMIZING THE NETWORK
  - FIRST PERFORM A COARSE SEARCH, USUALLY RECYCLING SOMETHING THAT WORKED, RUNE FEW EPOCH
  - THE, LONGER TRAINING, FINER SEARCH

FINAL THOUGHTS

- START SIMPLE - KEEP ALWAYS YOUR SCIENCE IN MIND. NEW ALGORITHMS ARE COOL BUT COULD BE OVERKILL.

- HOW DO I KNOW WHICH ALGORITHM TO USE?

  - DON'T KNOW

  - DEEP LEARNING: IMAGE, SPECTRAL DATA WITH SPATIAL CORRELATIONS EASILY IDENTIFIED BY YOUR EYE BUT DIFFICULT TO DESCRIBE

- STAY CONNECTED! (OR MAKE FRIENDS WHICH ARE CONNECTED). MOST OF THE THINGS I HAVE SHOWN HERE WILL BE UPDATED IN A FEW MONTHS

FINAL THOUGHTS

- TRY TO CATCH THE ATTENTION OF COMPUTER SCIENTISTS WITH CHALLENGING PROBLEMS. WE DO HAVE MANY IN ASTRONOMY!
- SPECIFIC PROPERTIES OF ASTROPHYSICAL DATA:
  - OBJECTS WITH NO BOUNDARIES
  - HUGE DYNAMICAL RANGE
  - LOW S/N REGIME
- ALSO THEY ARE UNIQUE:
  - BIG-DATA VOLUMES
  - PUBLICLY AVAILABLE ALMOST INSTANTANEOUSLY
  - NO COMMERCIAL VALUE / ETHICAL PROBLEMS

THE FUTURE IS VERY EXCITING!

MOST OF THE PROCESSING WE DO TODAY ON IMAGES CAN BE ACTUALLY SOLVED WITH AI…