PART V: A VERY BRIEF INTRODUCTION TO DEEP UNSUPERVISED LEARNING

*elements taken from D. Kirkby lectures at KSPA

TYPES OF MACHINE LEARNING

the machine is told what to look for

the machine is NOT told what to look for

SUPERVISED

UN-SUPERVISED

TWO VERY BROAD TYPES OF MACHINE LEARNING ALGORITHMS

TYPES OF MACHINE LEARNING

the machine is told what to look for

SUPERVISED

LEARNS A MAP FROM X [FEATURES] TO Y [LABELS]

$$P(X|Y)$$

the machine is NOT told what to look for

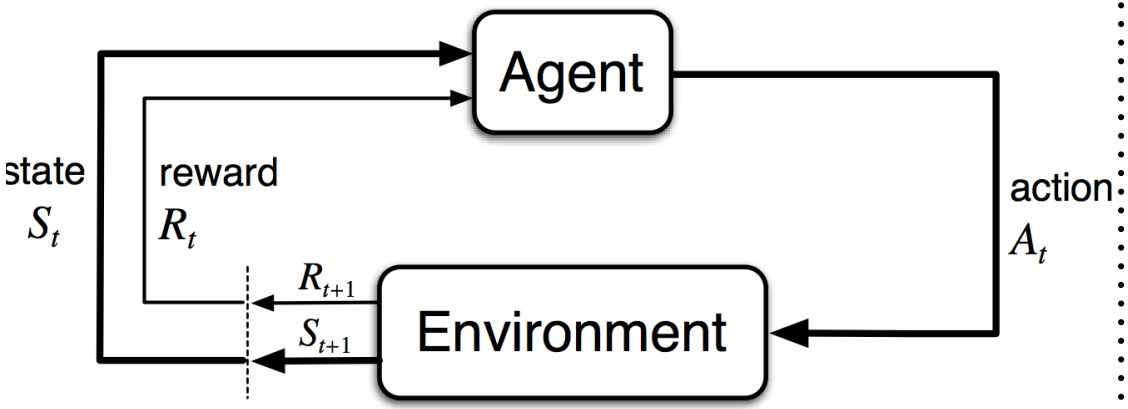
UN-SUPERVISED

NO LABELS - DISCOVER PATTERNS

$$P(X)$$

ACTUALLY, THERE IS A THIRD AND FORTH TYPE THAT WE WILL NOT HAVE TIME TO COVER

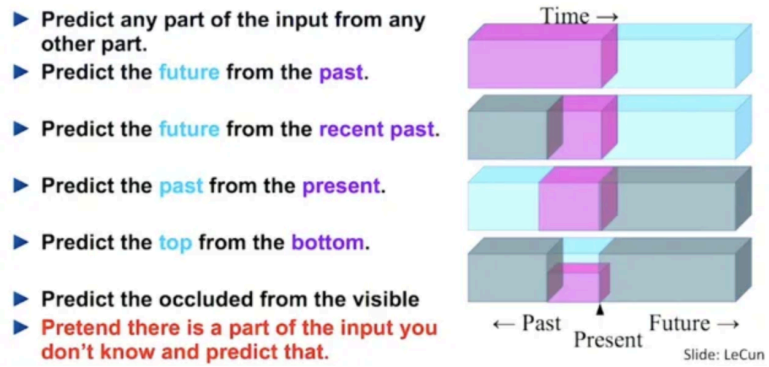
EINFORCMENT LEARNING:



source: Sutton & Barto

ITERATIVE LEARNING THROUGH TRIAL/ERROR: USED FOR ALPHAGO FOR EXAMPLE

SELF-SUPERVISED LEARNING:



> That's also why more knowledge about the structure of the world can be learned through self-supervised learning than from the other two paradigms: the data is unlimited, and amount of feedback provided by each example is huge.
>
> Y. LeCun

a self-supervised learning system attempts to predict parts of its inputs based on the other parts of its inputs

IT IS AN ACTIVE FIELD OF RESEARCH BECAUSE LABELLING IS ONE OF THE MAIN BOTTLENECKS OF MACHINE LEARNING

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

@ Y. LECUN

IN THIS LAST PART WE ARE GOING TO BRIEFLY INTRODUCE CURRENT TECHNIQUES OF UNSUPERVISED LEARNING WITH NEURAL NETWORKS

THERE ARE 3 MAJOR APPLICATIONS TO ASTRONOMY (THAT I CAN THINK OF):

- **DIMENSIONALITY REDUCTION:** HOW CAN I REPRESENT MY COMPLEX DATA MORE EFFICIENTLY TO GET NEW INSIGHTS INTO ITS STRUCTURE?

- **GENERATIVE MODELING:** HOW CAN I INTERPOLATE / EXTRAPOLATE A (SMALL, SPARSE) DATASET TO GENERATE NEW DATA SAMPLED FROM THE SAME (UNKNOWN) DISTRIBUTION?

- **PROBABILISTIC MODELING:**WHAT IS THE PROBABILITY THAT A NEW OBSERVATION IS DRAWN FROM THE SAME (UNKNOWN) DISTRIBUTION AS SOME REFERENCE (SMALL, SPARSE) DATASET?

DIMENSIONALITY REDUCTION

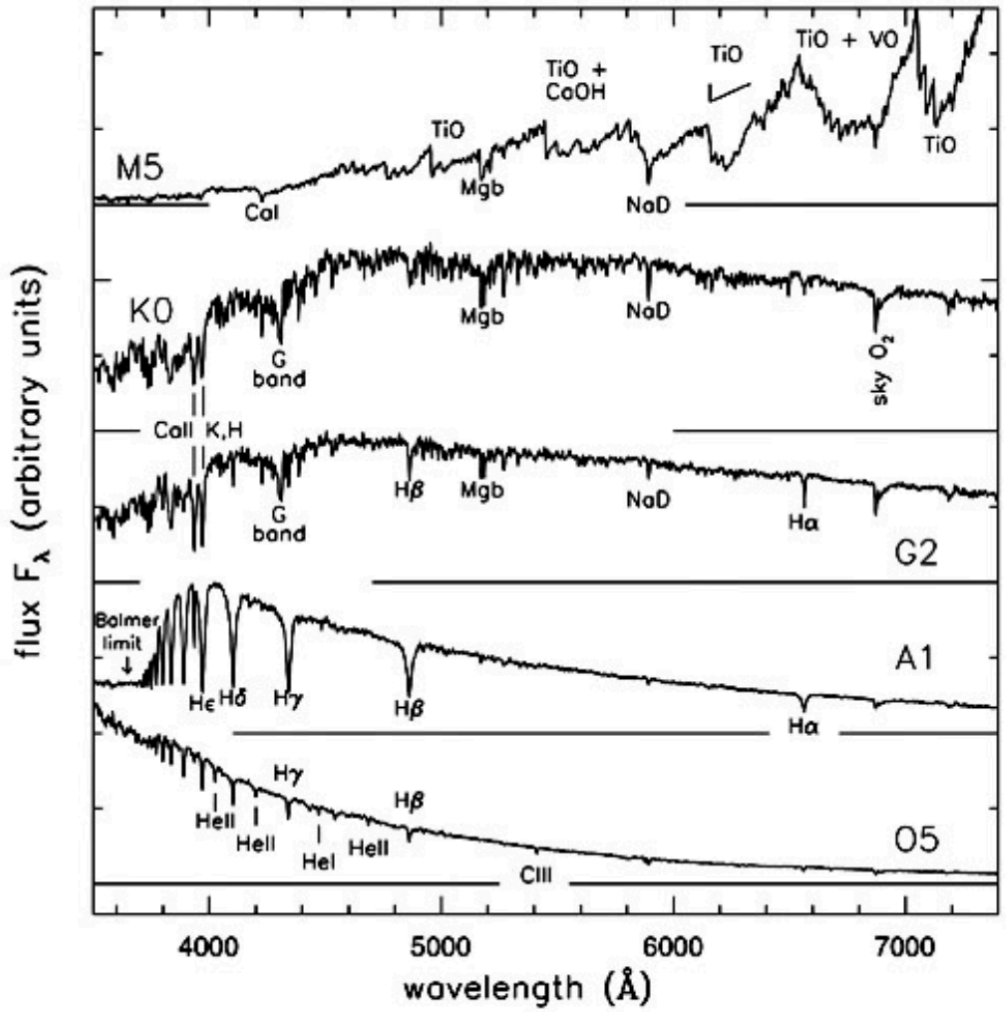
NEED OF REPRESENTING THE DATA IN A SMALLER DIMENSIONALITY SPACE

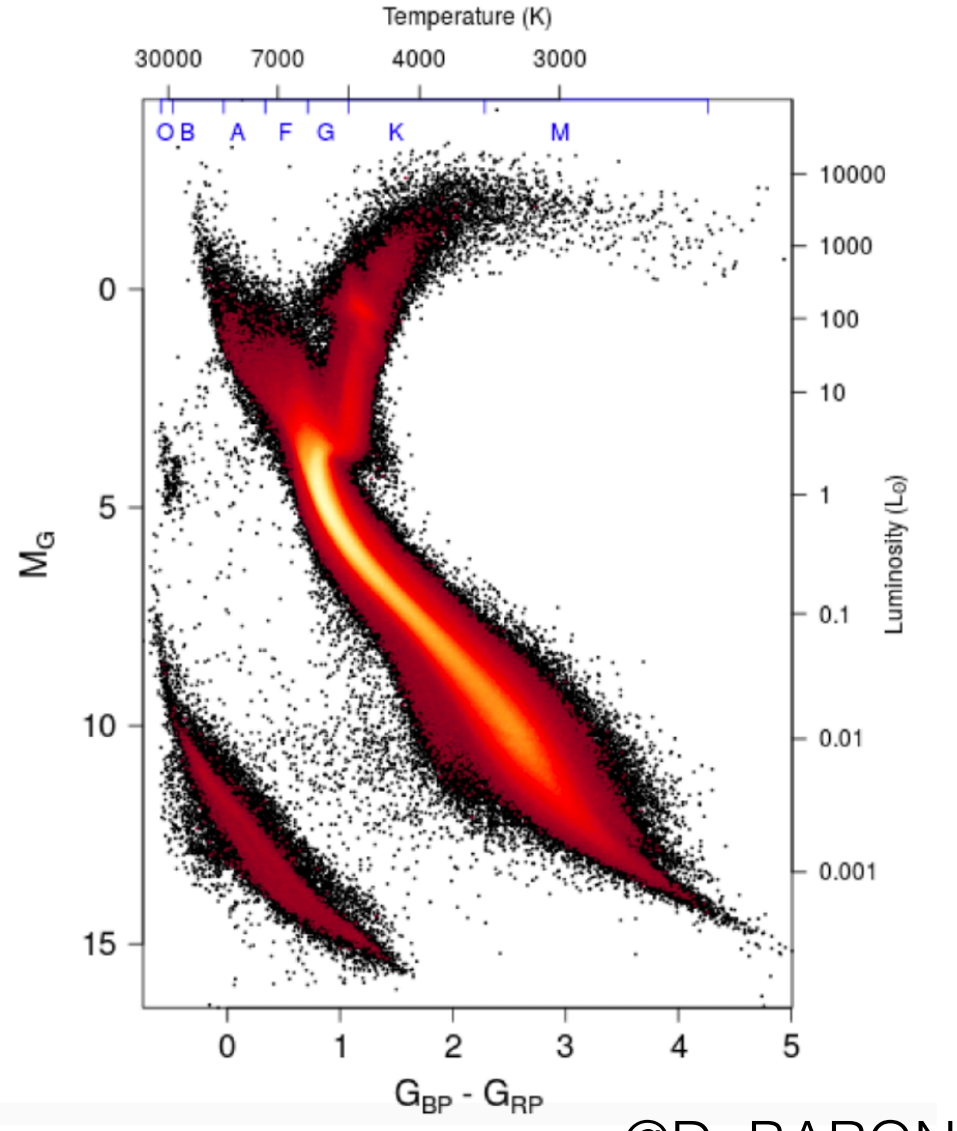
$$G_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

@D. BARON

MOST OF THE SCIENTIFIC WORK IS ABOUT COMPRESSION OF NATURE IN SOME EXPLANATORY EQUATIONS

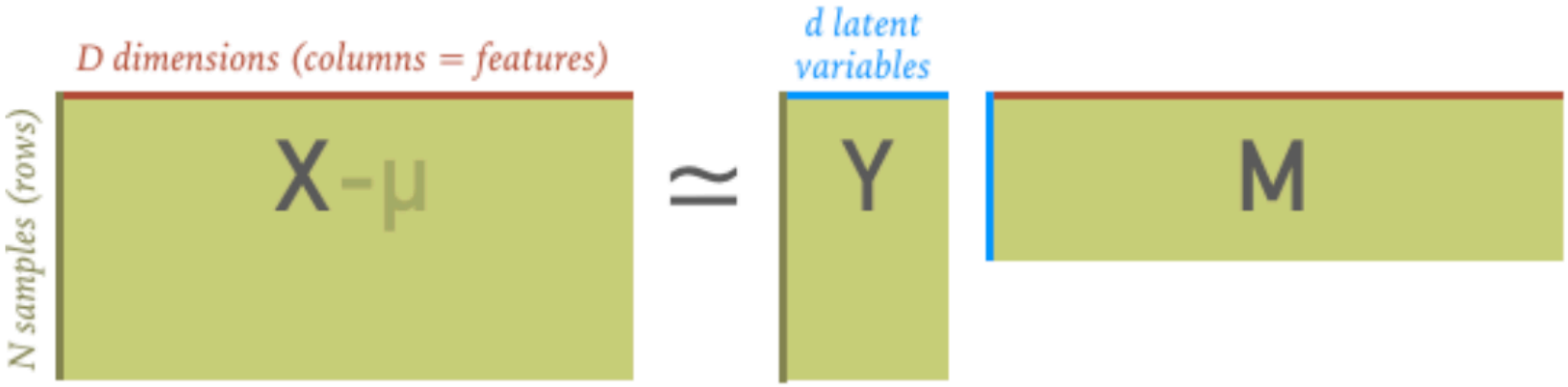
From: Gaia Collaboration et al. 2018





@D. BARON

THE STELLAR MAIN SEQUENCE IS A DIMENSIONALITY REDUCTION



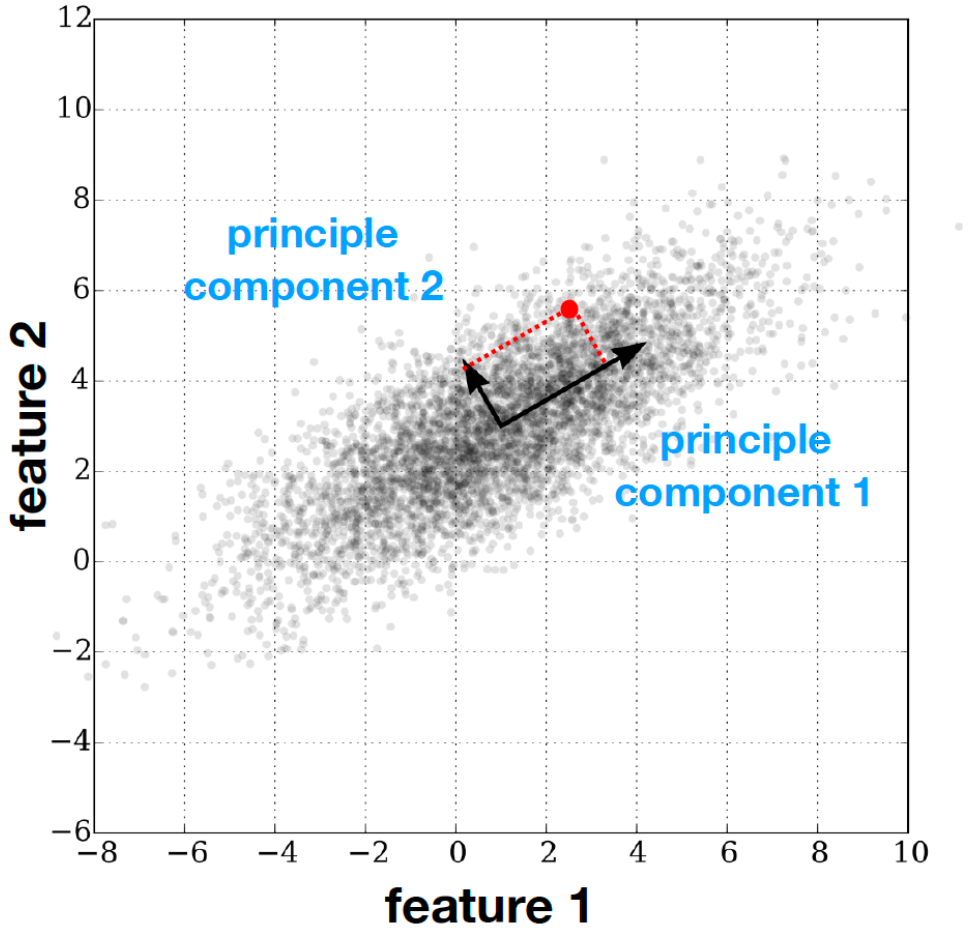
CLASSICAL METHODS FOR DIMENSIONALITY REDUCTION SEEK A LINEAR DECOMPOSITION THAT BEST EXPLAINS THE OBSERVATIONS X IN TERMS OF **LATENT VARIABLES Y**

PRINCIPAL COMPONENT ANALYSIS

PCA CONVERT A SET OF (CORRELATED) VARIABLES INTO A SET OF VALUES LINEARLY UNCORRELATED

1. THE FIRST PRINCIPLE COMPONENT ("PROTOTYPE"), HAS THE LARGEST POSSIBLE VARIANCE

2. THE FOLLOWING COMPONENTS HAVE THE LARGEST VARIABLES WITH THE ADDITIONAL CONSTRAIN THAT THEY ARE ORTHOGONAL TO THE PRECEDING COMPONENTS



PRINCIPAL COMPONENT ANALYSIS

1. COMPUTE THE **COVARIANCE MATRIX** OF YOUR DATA

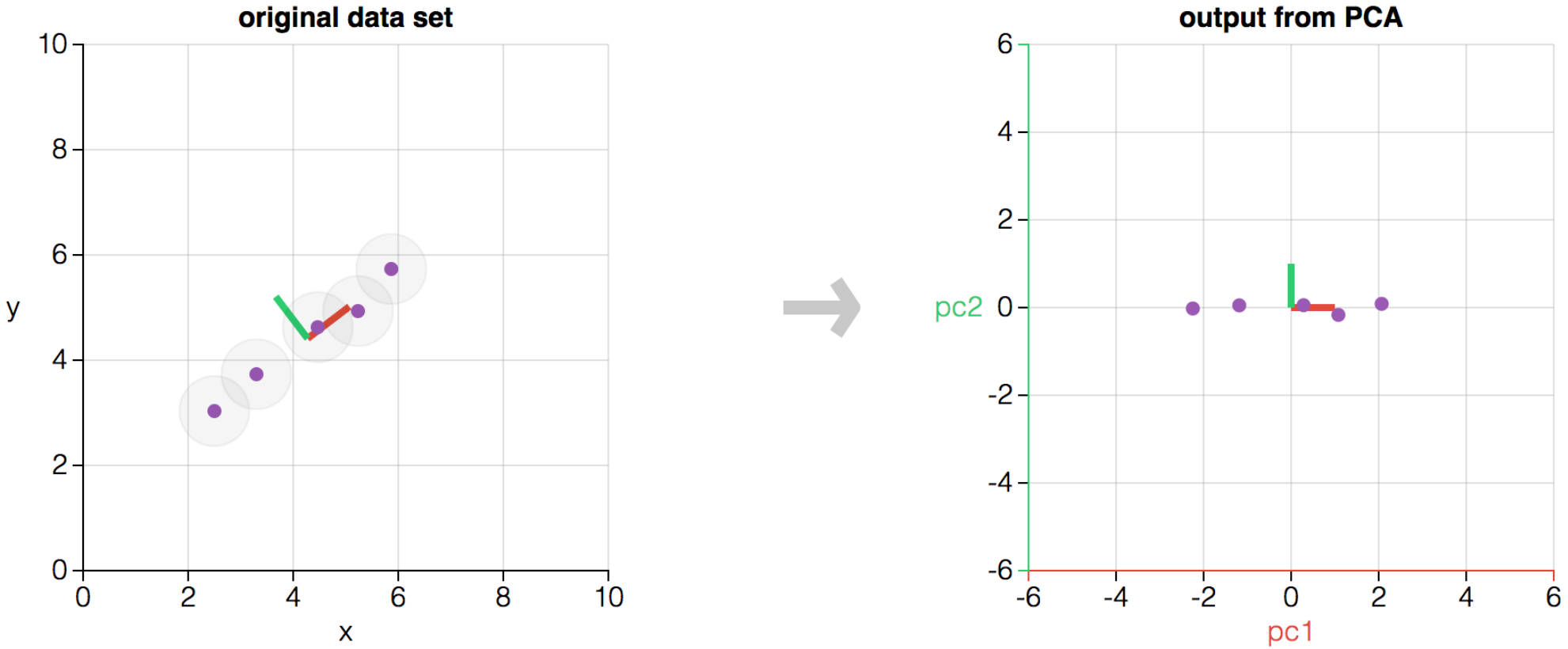
2. COMPUTE THE **EIGENVALUES AND EIGENVECTORS** OF THE **COVARAINCE MATRIX**

3.THE **EIGENVECTORS** PROVIDE THE DIRECTION OF MAXIMUM VARIANCE AND THE **EIGENVALUES** THE ‘IMPORTANCE’ OF THAT PARTICULAR FEATURE

REMEMBER, THE COVARIANCE MATRIX IS NOTHING ELSE THAN THAT:

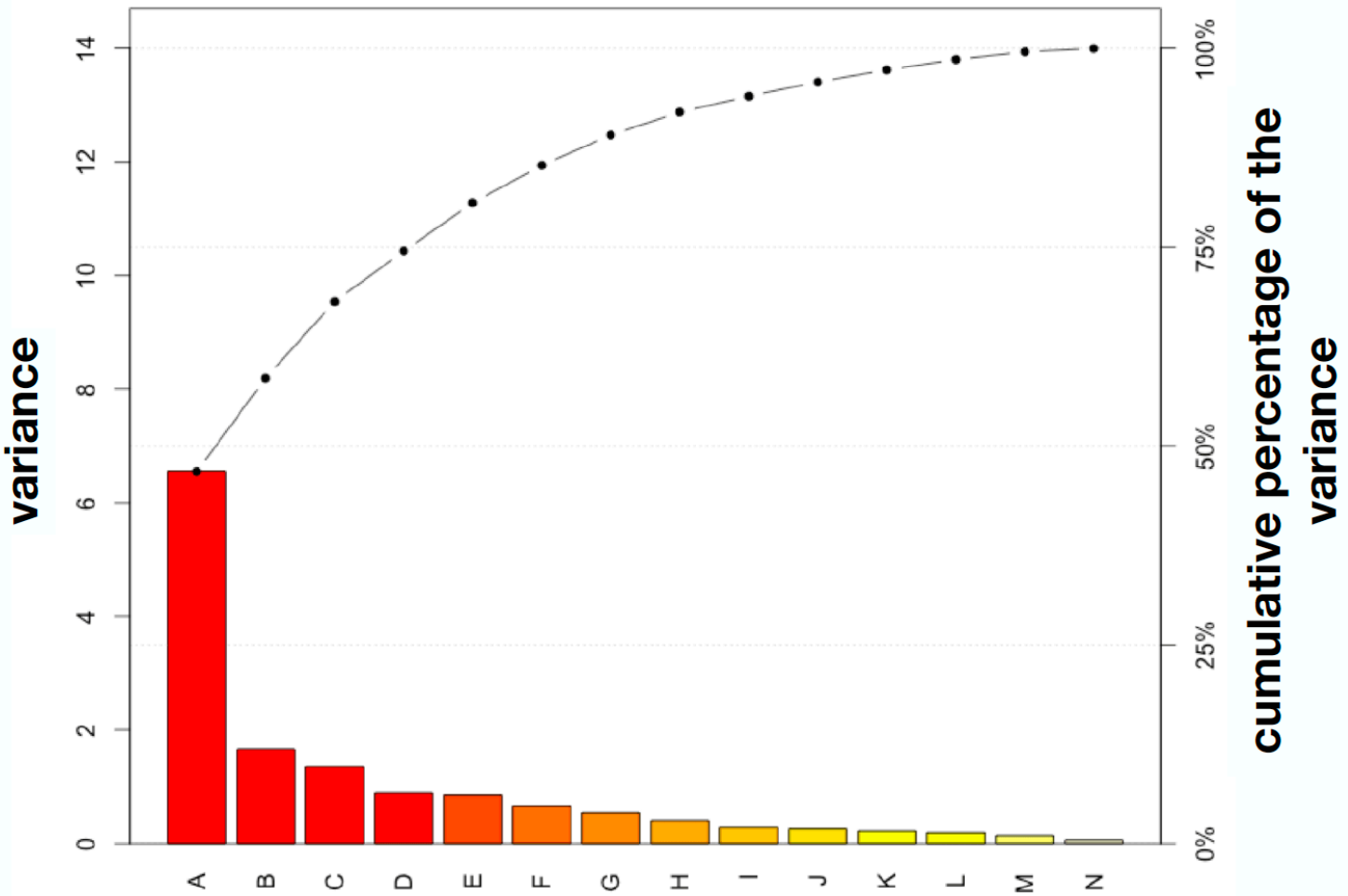
$$\mathrm{K}_{\mathbf{XX}} = \begin{bmatrix} \mathrm{E}[(X_1 - \mathrm{E}[X_1])(X_1 - \mathrm{E}[X_1])] & \mathrm{E}[(X_1 - \mathrm{E}[X_1])(X_2 - \mathrm{E}[X_2])] & \cdots & \mathrm{E}[(X_1 - \mathrm{E}[X_1])(X_n - \mathrm{E}[X_n])] \\ \mathrm{E}[(X_2 - \mathrm{E}[X_2])(X_1 - \mathrm{E}[X_1])] & \mathrm{E}[(X_2 - \mathrm{E}[X_2])(X_2 - \mathrm{E}[X_2])] & \cdots & \mathrm{E}[(X_2 - \mathrm{E}[X_2])(X_n - \mathrm{E}[X_n])] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n - \mathrm{E}[X_n])(X_1 - \mathrm{E}[X_1])] & \mathrm{E}[(X_n - \mathrm{E}[X_n])(X_2 - \mathrm{E}[X_2])] & \cdots & \mathrm{E}[(X_n - \mathrm{E}[X_n])(X_n - \mathrm{E}[X_n])] \end{bmatrix}$$

https://en.wikipedia.org/wiki/Covariance_matrix



https://setosa.io/ev/principal-component-analysis/

PRINCIPAL COMPONENT ANALYSIS

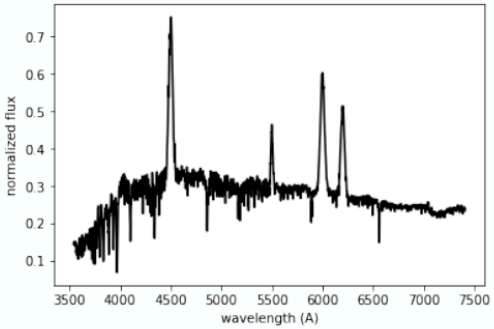


IT RESULTS IN DATA COMPRESSION, BY REPRESENTING EACH OBJECT AS A PROJECTION OF THE FIRST PRINCIPLE COMPONENTS

@D. BARON

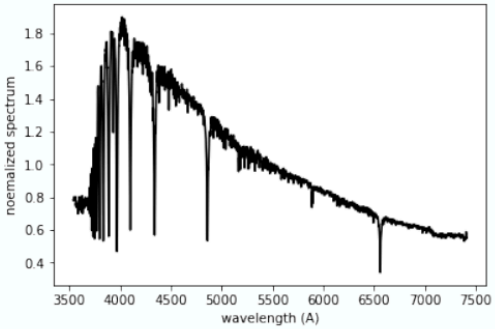
PRINCIPAL COMPONENT ANALYSIS

observed object



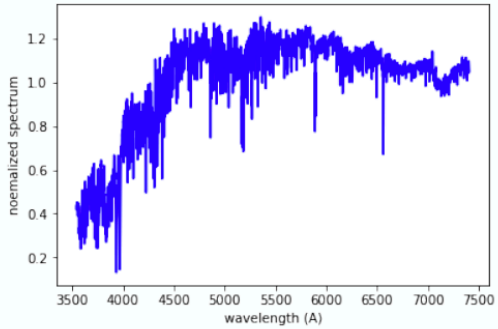
= A *

principle comp. 1



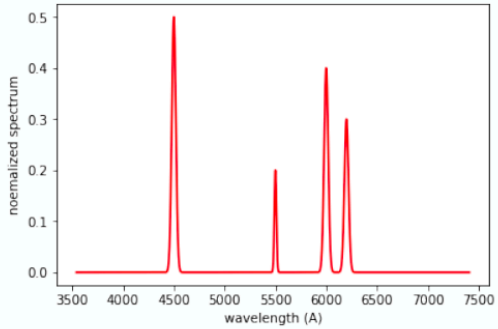
+ B *

principle comp. 2



+ C *

principle comp. 3



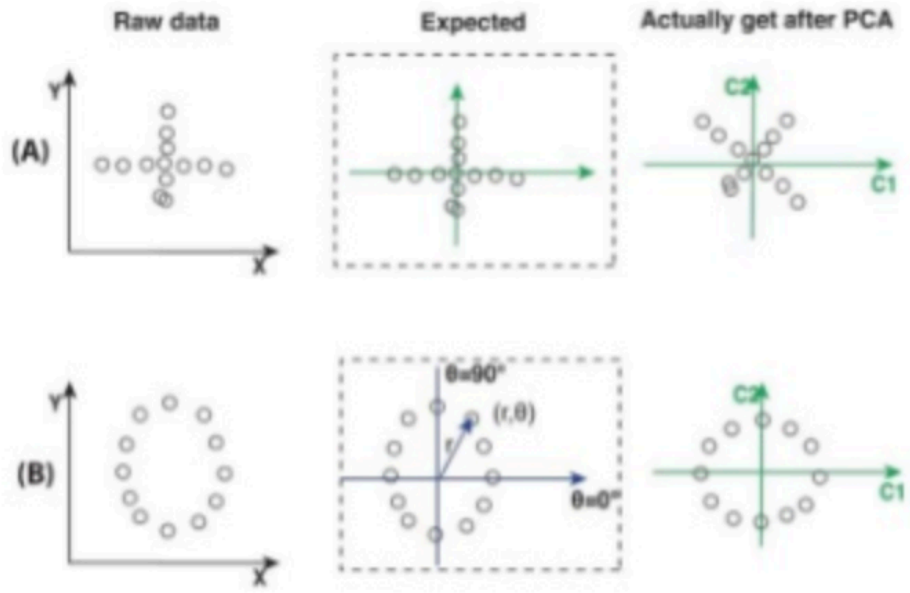
OBJECTS ARE DIVIDED INTO MAJOR PROTOTYPES AND ALL OBJECTS CAN BE OBTAINED AS LINEAR COMBINATIONS

@D. BARON

LIMITATIONS OF PCA

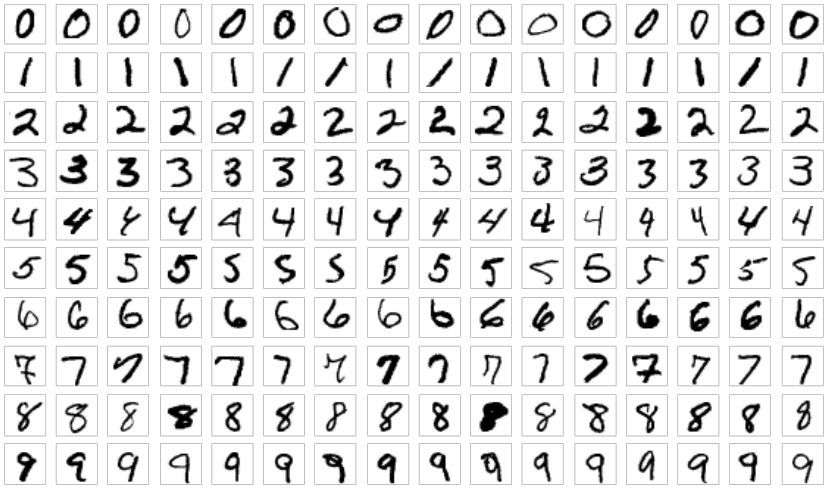
PCA APPLY LINEAR TRANSFORMATIONS

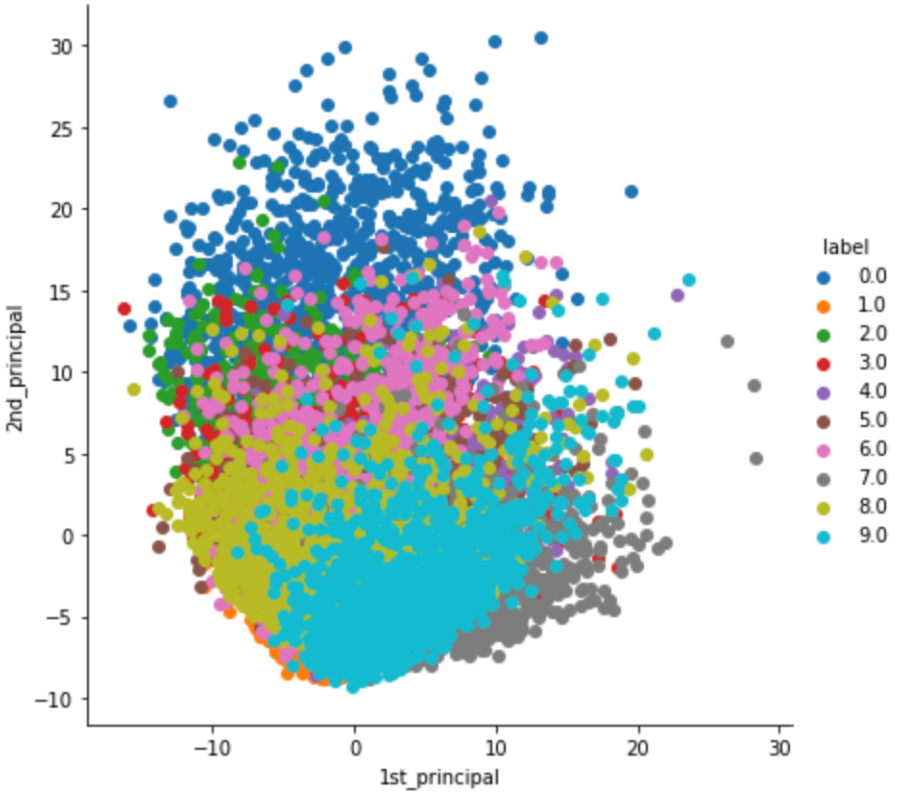
SINCE WE USE THE COVARIANCE MATRIX, IT ASSUMES THAT THE DATA FOLLOWS A **MULTIDIMENSIONAL NORMAL DISTRIBUTION**



LIMITATIONS OF PCA

AND DATA IS NOT ALWAYS GAUSSIAN….

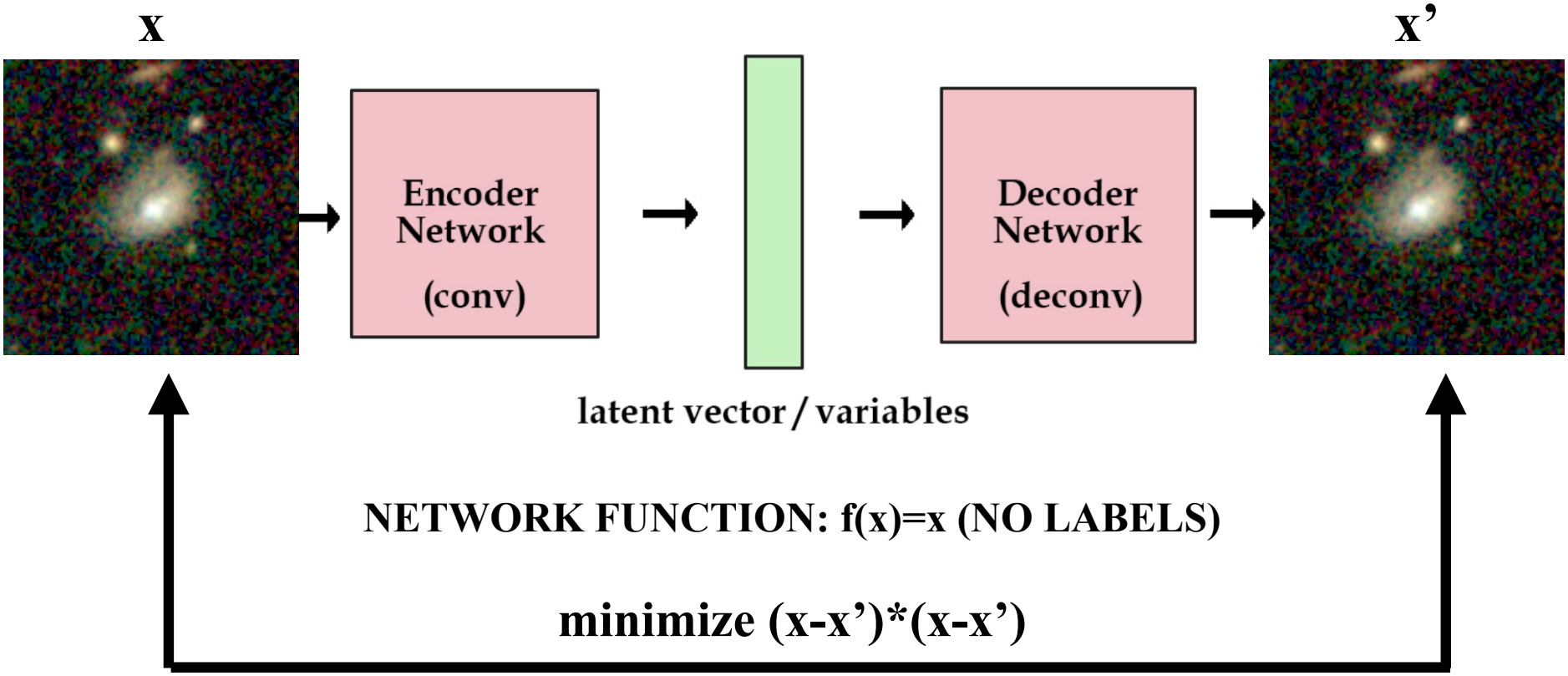




https://medium.com/analytics-vidhya/principal-component-analysis-pca-with-code-on-mnist-dataset-da7de0d07c22

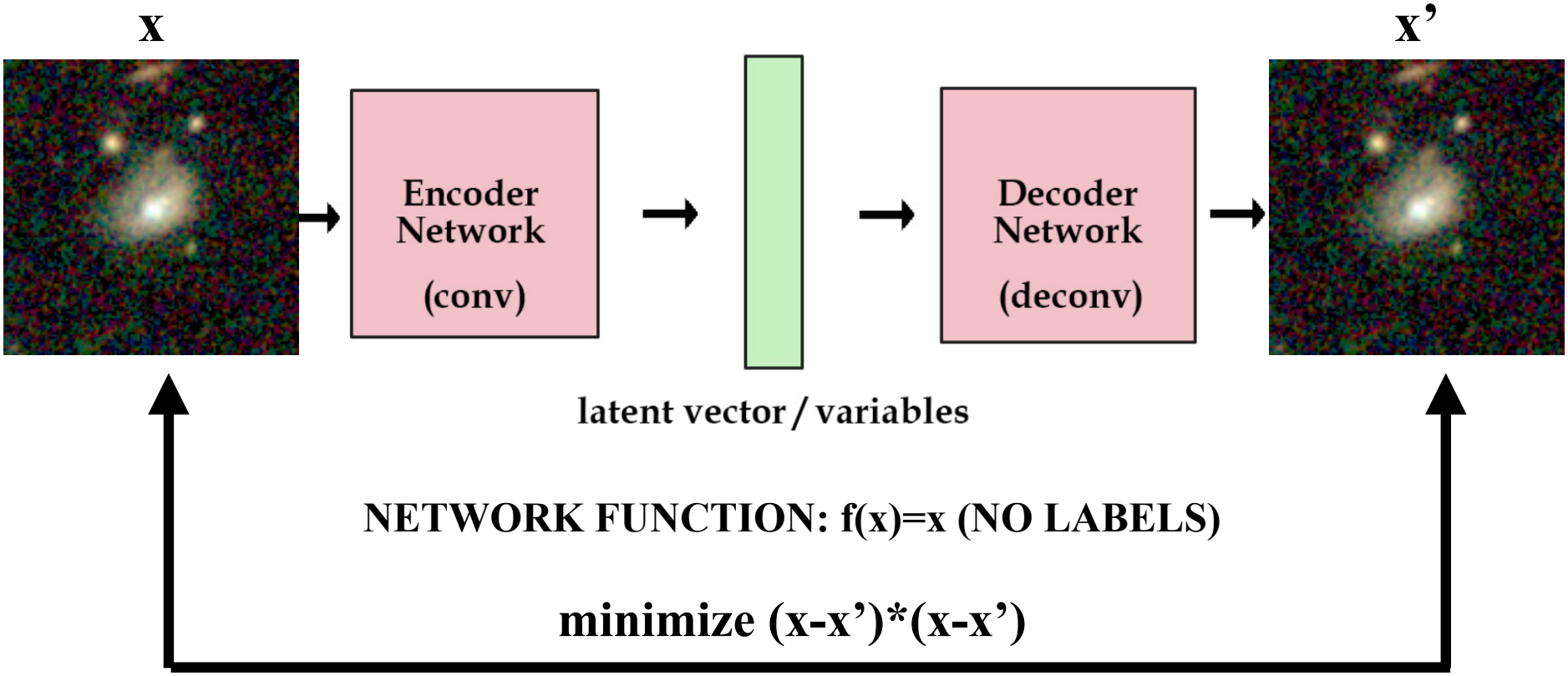
CAN WE GENERALIZE THAT?

CONVOLUTIONAL AUTO-ENCODER



AN AUTO-ENCODER IS ANY NETWORK WITH IDENTICAL INPUT AND OUTPUT

CONVOLUTIONAL AUTO-ENCODER



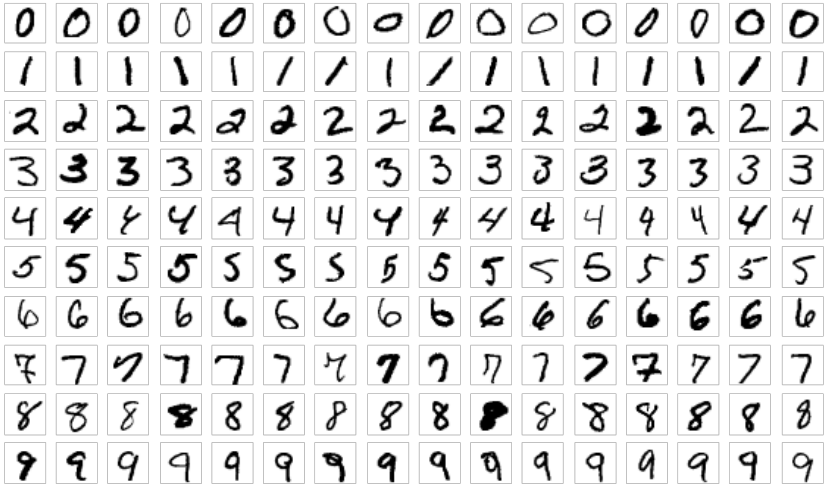
BY REDUCING THE DIMENSIONALITY IN THE LATENT SPACE WE FORCE THE NETWORK TO LEARN A REPRESENTATION OF THE INPUT DATA IN A LOWER DIMENSIONALITY SPACE

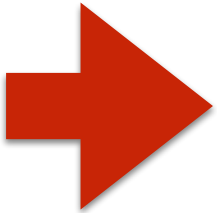
* NO NEED TO BE CONVOLUTIONAL - ANY NEURAL NETWORK WITH A BOTTLENECK WILL DO THE JOB

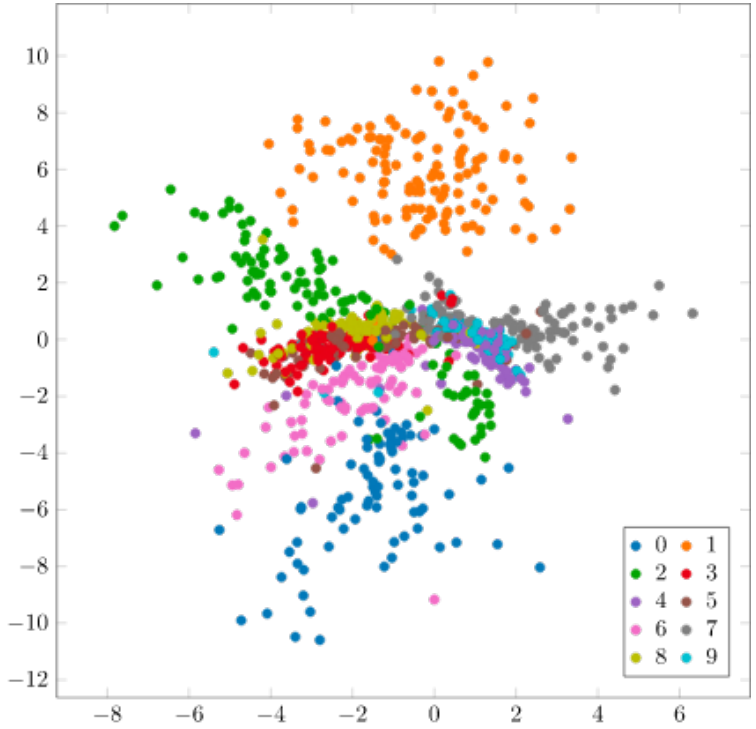
* **QUESTION**: WHAT WOULD HAPPEN IF WE SET AN AUTOENCODER WITH NO ACTIVATION FUNCTIONS?

SEE EXAMPLE FROM TUTORIALS FOR STAR-GALAXY SEPARATION

AUTOENCODER REPRESENTATION OF MNIST

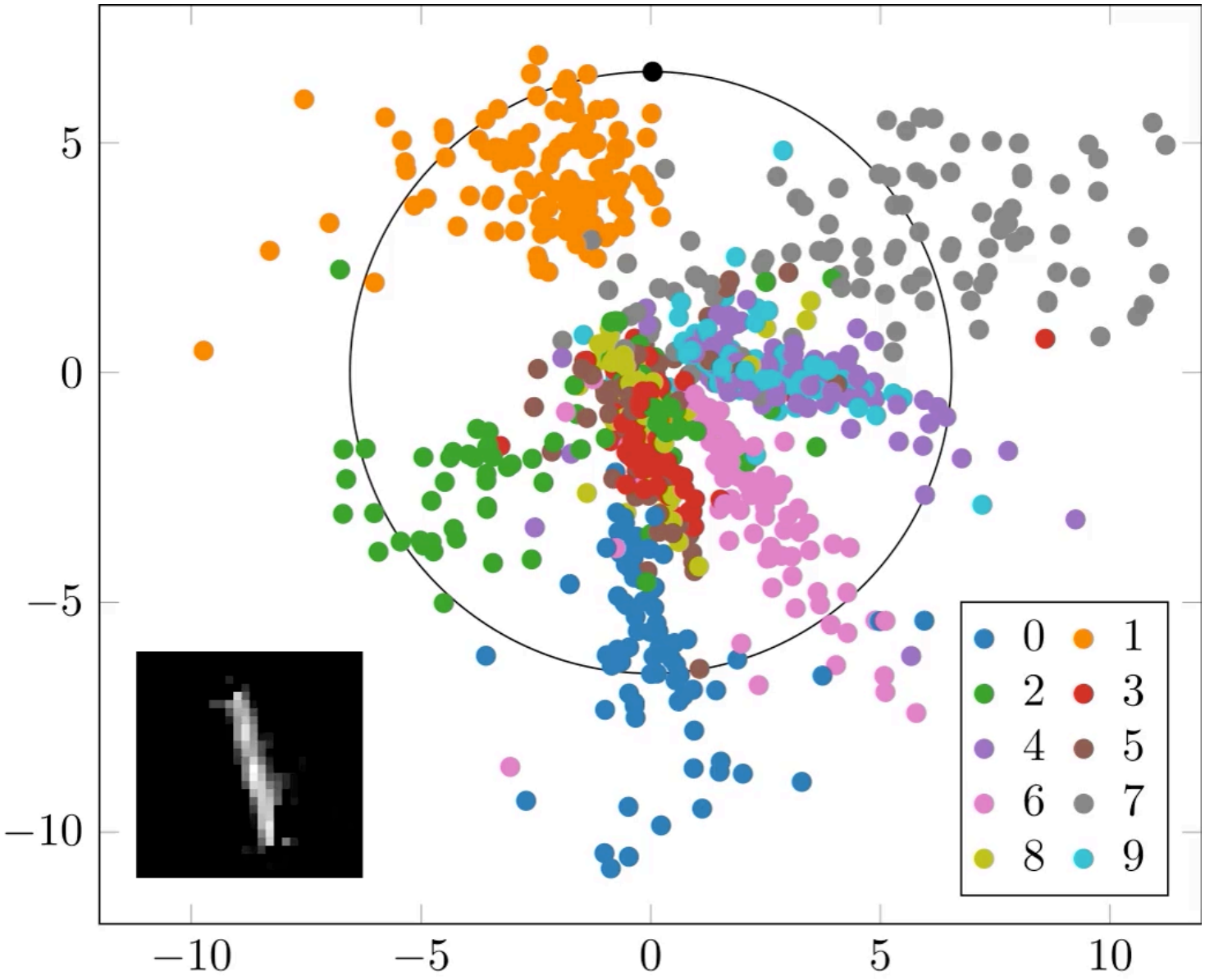




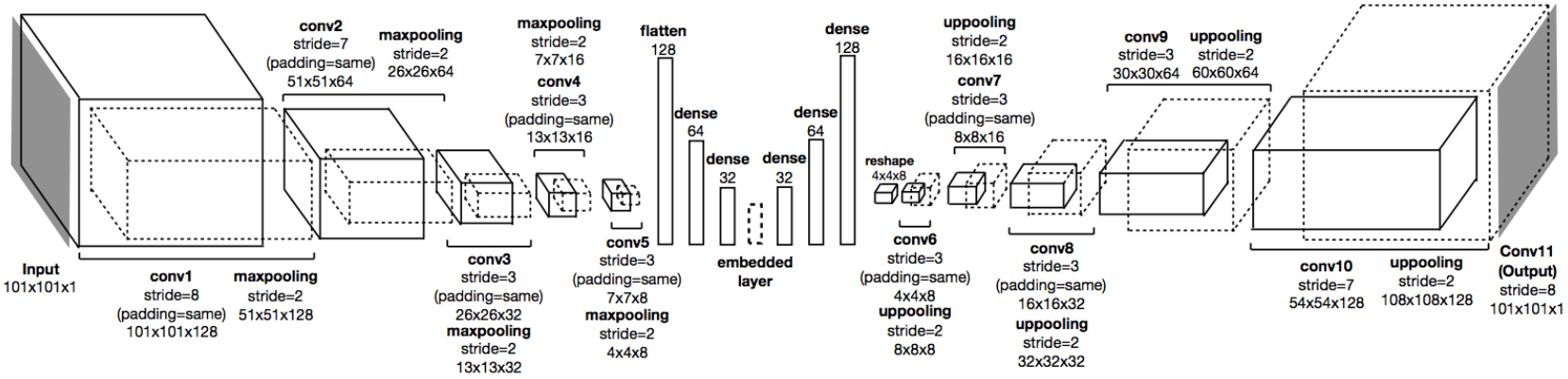


https://gertjanvandenburg.com/blog/autoencoder/

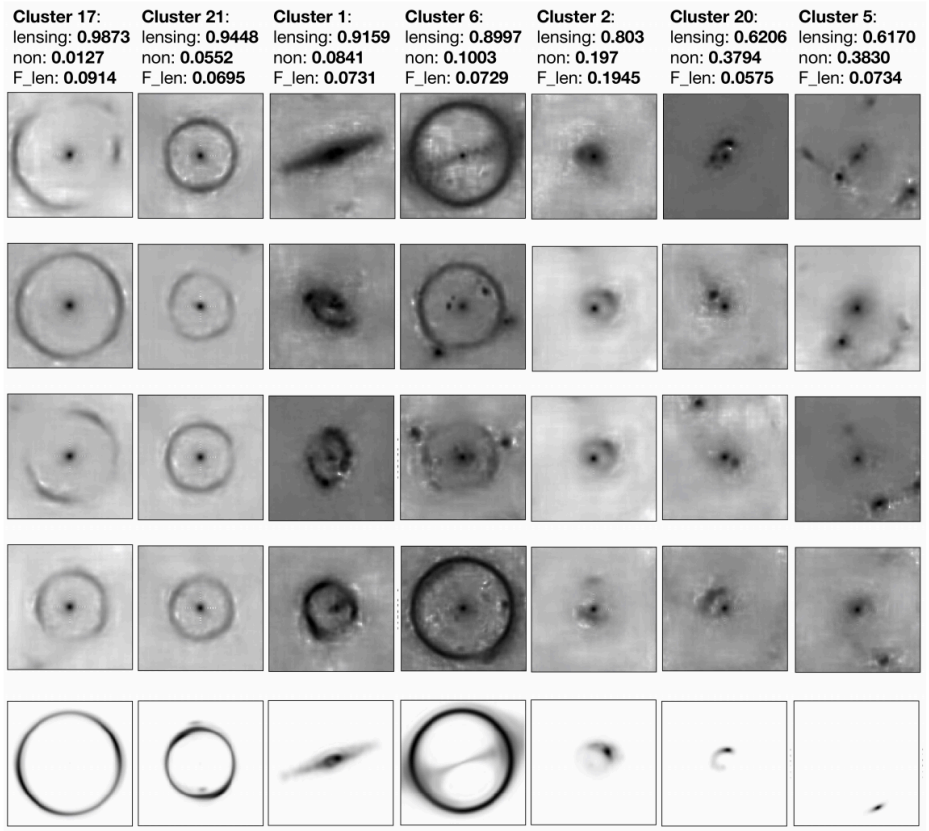
AUTOENCODER REPRESENTATION OF MNIST

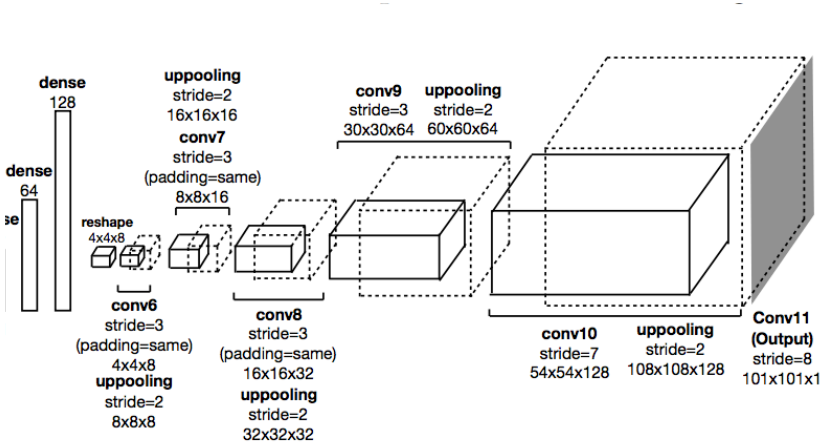


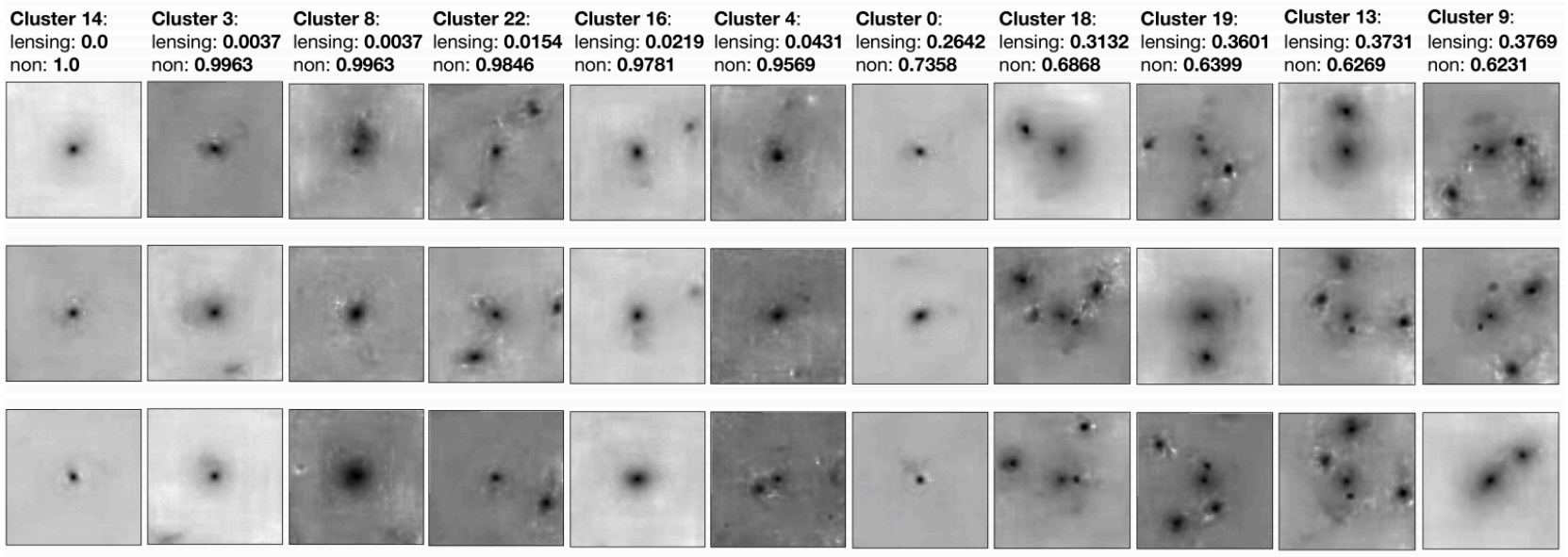
https://gertjanvandenburg.com/blog/autoencoder/


Input
101x101x1
conv1
stride=8
(padding=same)
101x101x128
maxpooling
stride=2
51x51x128
conv2
stride=7
(padding=same)
51x51x64
maxpooling
stride=2
26x26x64
conv3
stride=3
(padding=same)
26x26x32
maxpooling
stride=2
13x13x32
maxpooling
stride=2
7x7x16
conv4
stride=3
(padding=same)
13x13x16
conv5
stride=3
(padding=same)
7x7x8
maxpooling
stride=2
4x4x8
flatten
128
dense
64
dense
32
embedded
layer
dense
32
dense
64
dense
128
reshape
4x4x8
conv6
stride=3
(padding=same)
4x4x8
uppooling
stride=2
8x8x8
uppooling
stride=2
16x16x16
conv7
stride=3
(padding=same)
8x8x16
conv8
stride=3
(padding=same)
16x16x32
uppooling
stride=2
32x32x32
conv9
stride=3
30x30x64
uppooling
stride=2
60x60x64
conv10
stride=7
54x54x128
uppooling
stride=2
108x108x128
Conv11
(Output)
stride=8
101x101x1

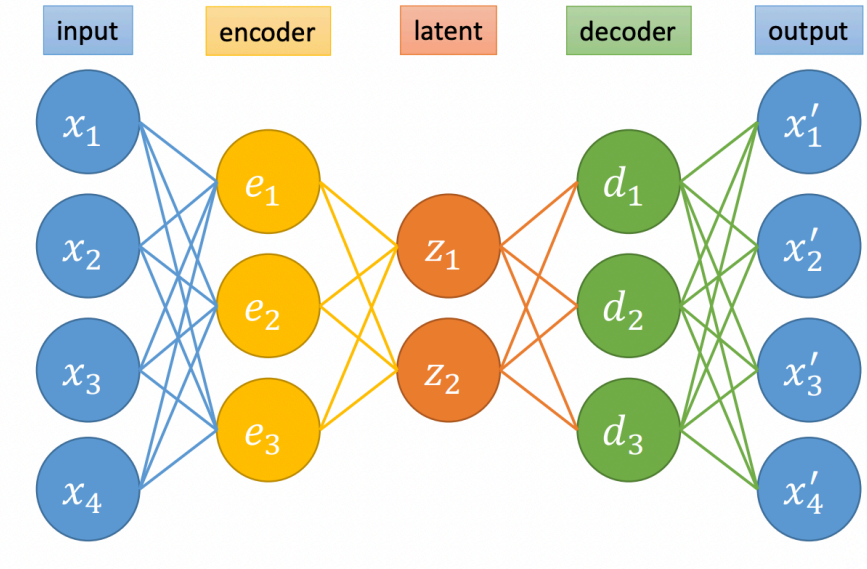
Cheng+20

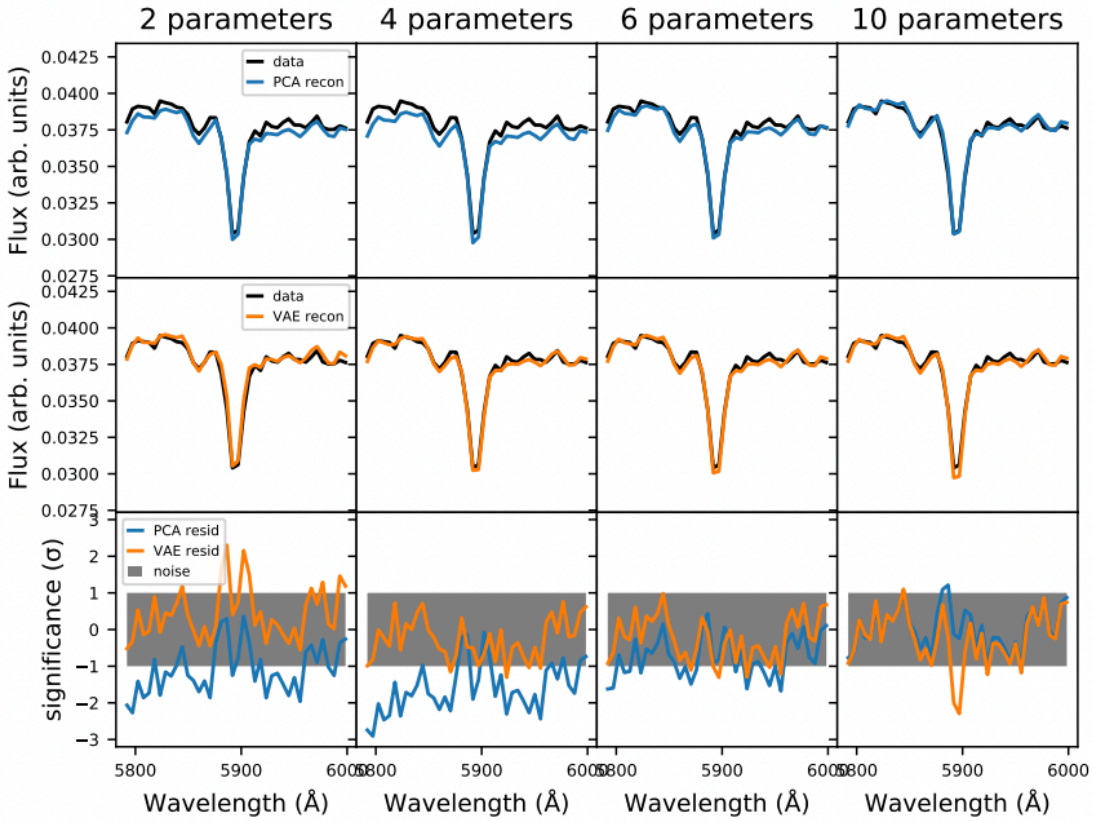

Cluster 17: lensing: 0.9873 non: 0.0127 F_len: 0.0914
Cluster 21: lensing: 0.9448 non: 0.0552 F_len: 0.0695
Cluster 1: lensing: 0.9159 non: 0.0841 F_len: 0.0731
Cluster 6: lensing: 0.8997 non: 0.1003 F_len: 0.0729
Cluster 2: lensing: 0.803 non: 0.197 F_len: 0.1945
Cluster 20: lensing: 0.6206 non: 0.3794 F_len: 0.0575
Cluster 5: lensing: 0.6170 non: 0.3830 F_len: 0.0734


dense
128
dense
64
reshape
4x4x8
conv6
stride=3
(padding=same)
4x4x8
uppooling
stride=2
8x8x8
uppooling
stride=2
16x16x16
conv7
stride=3
(padding=same)
8x8x16
conv8
stride=3
(padding=same)
16x16x32
uppooling
stride=2
32x32x32
conv9
stride=3
30x30x64
uppooling
stride=2
60x60x64
conv10
stride=7
54x54x128
uppooling
stride=2
108x108x128
Conv11
(Output)
stride=8
101x101x1


Cluster 14: lensing: 0.0 non: 1.0
Cluster 3: lensing: 0.0037 non: 0.9963
Cluster 8: lensing: 0.0037 non: 0.9963
Cluster 22: lensing: 0.0154 non: 0.9846
Cluster 16: lensing: 0.0219 non: 0.9781
Cluster 4: lensing: 0.0431 non: 0.9569
Cluster 0: lensing: 0.2642 non: 0.7358
Cluster 18: lensing: 0.3132 non: 0.6868
Cluster 19: lensing: 0.3601 non: 0.6399
Cluster 13: lensing: 0.3731 non: 0.6269
Cluster 9: lensing: 0.3769 non: 0.6231

Cheng+20


input
encoder
latent
decoder
output
x_1
x_2
x_3
x_4
e_1
e_2
e_3
z_1
z_2
d_1
d_2
d_3
x'_1
x'_2
x'_3
x'_4


2 parameters
4 parameters
6 parameters
10 parameters
Flux (arb. units)
data
PCA recon
data
VAE recon
PCA resid
VAE resid
noise
significance (σ)
Wavelength (Å)

Portillo+21

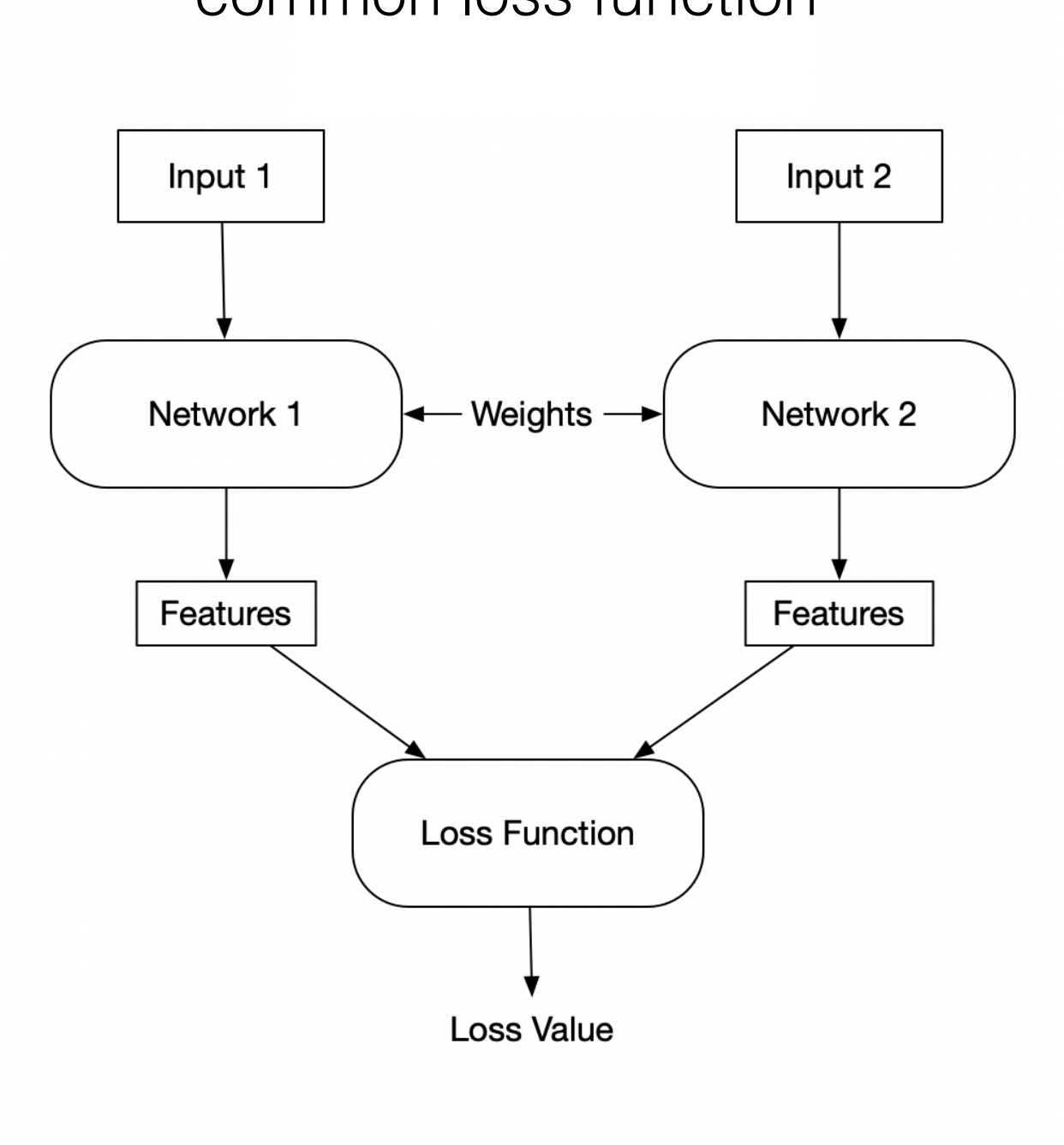
SELF-SUPERVISED CONTRASTIVE LEARNING

Humans tend to identify objects without remembering all the details, by creating some abstract representations which are used to identify new objects of similar time.

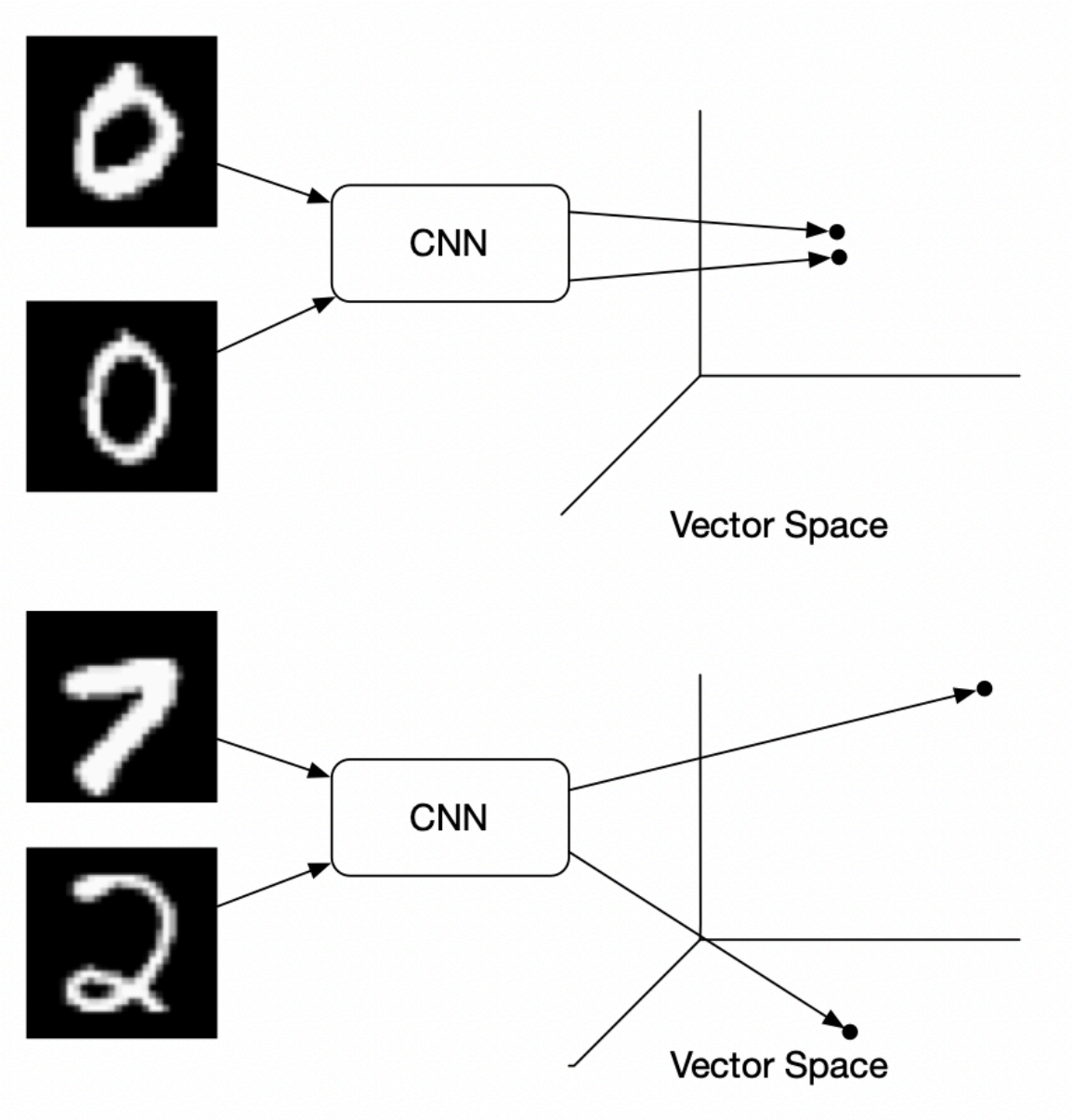
The main goal of self-supervised contrastive learning is to create and generalize these representations.

Therefore, it aims at building some general representations that can be used for other “downstream tasks”

Siamese networks are two networks sharing weights and common loss function



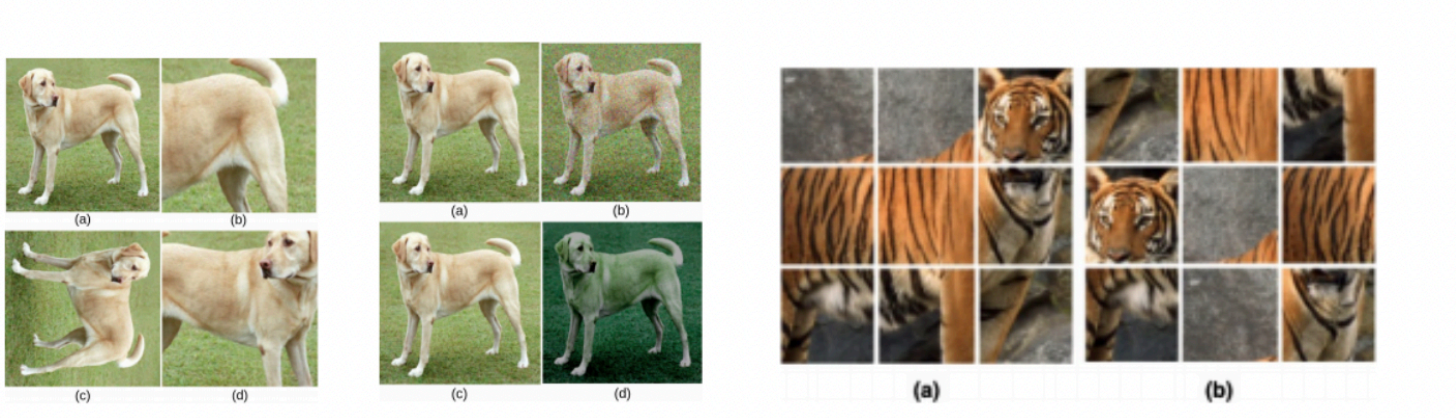
In a supervised setting, similar objects are used to be together by simply using a crossentropy loss



In most cases, labels are not available.

How do you recognise similar objects?

What would be the loss function?

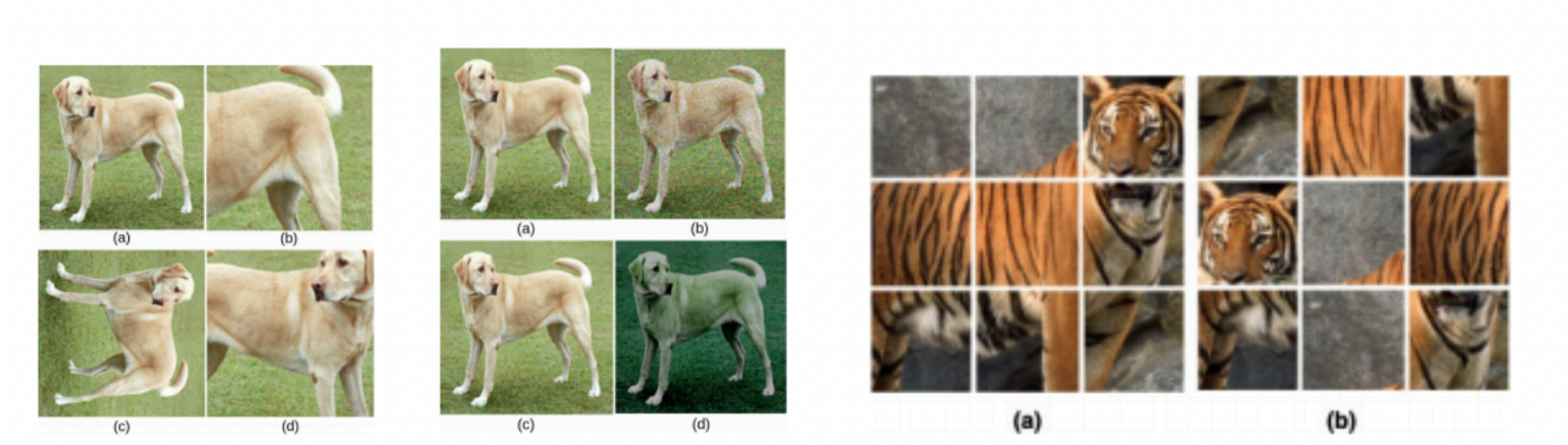


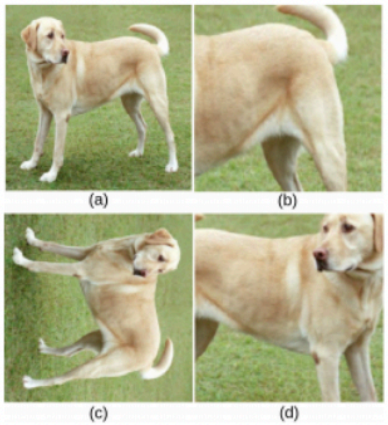
In most cases, labels are not available.

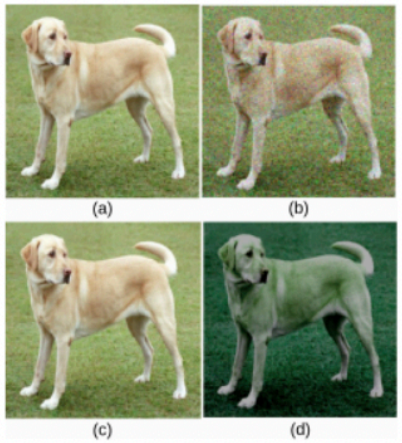
How do you recognise similar objects?

What would be the loss function?

We create“pretext tasks” from a unique image:







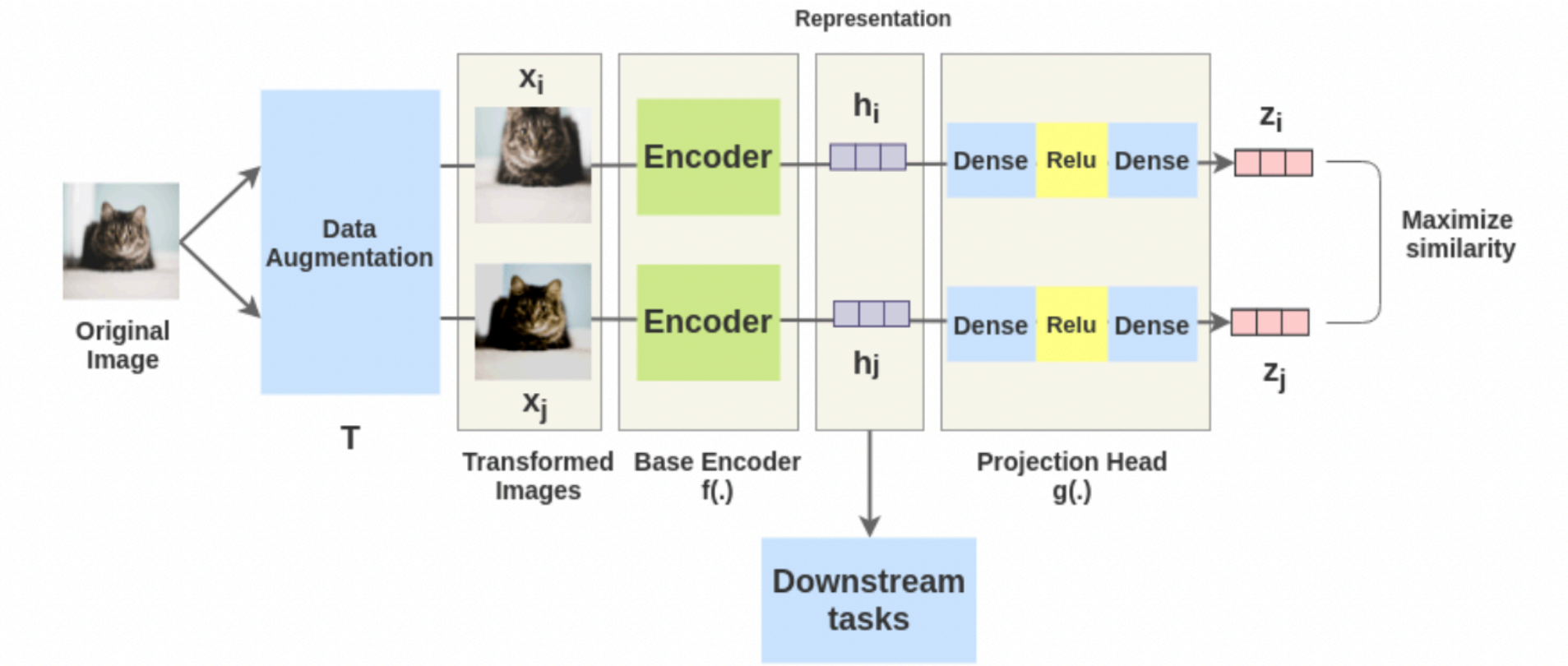
Color Augmentation

Image Rotation

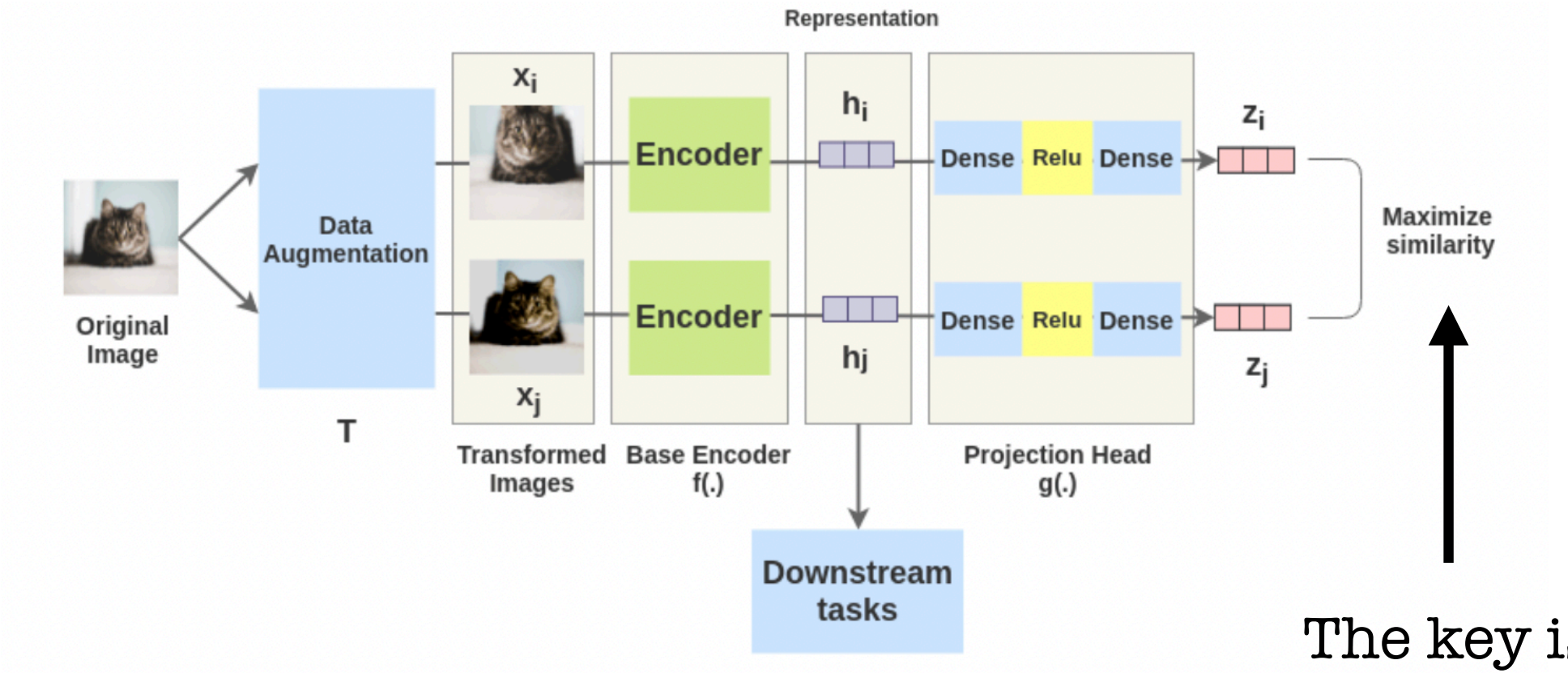
Image Cropping

Any geometrical transformation …

The augmented versions of the images are passed through siamese networks and projected into a latent variable z



The augmented versions of the images are passed through siamese networks and projected into a latent variable z



The key is the loss function

The contrastive loss:

Similarly between
two representations of positive pairs

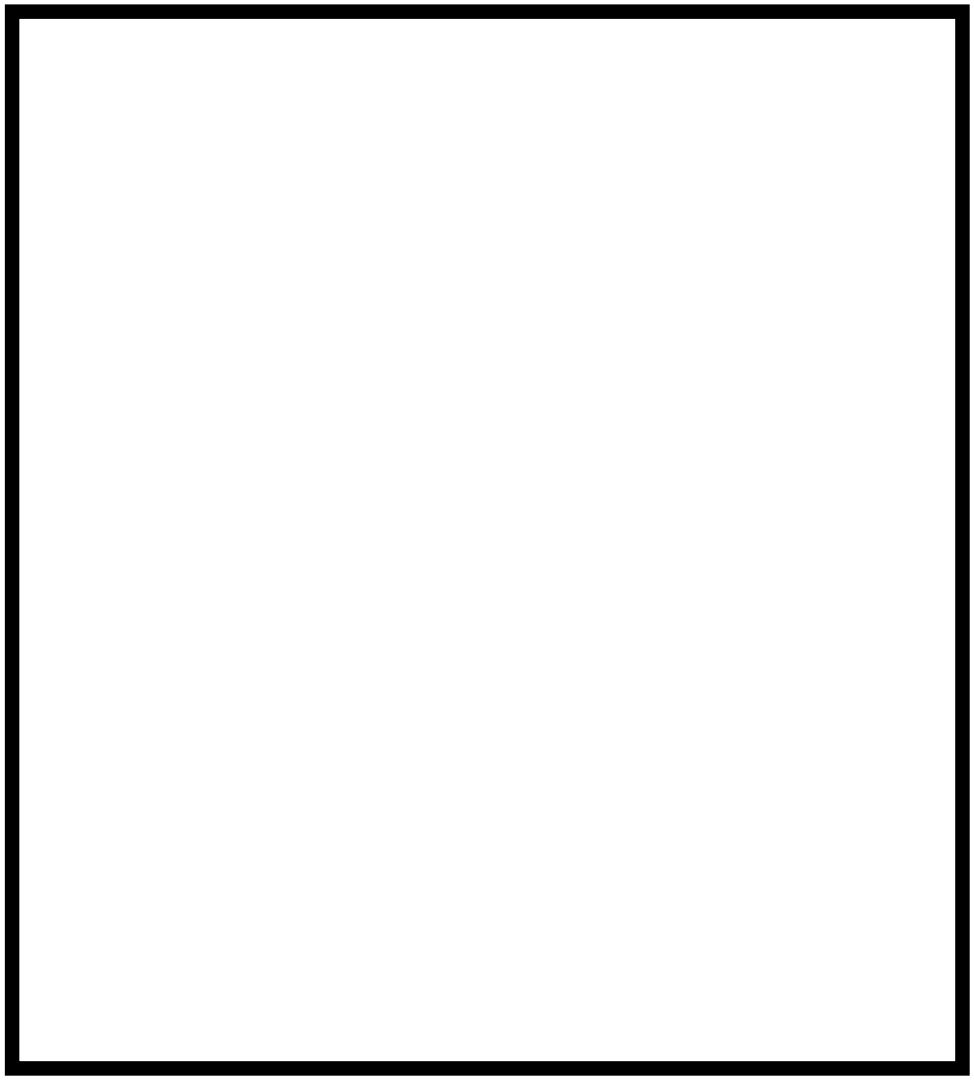
$$l_{i,j} = -\log \frac{\exp(\langle z_i, z_j \rangle / h)}{\sum_{k=1,k\neq i}^{2N} \exp(\langle z_i, z_k \rangle / h)},$$

Sum of all similarities between
negative pairs

LEARNING REPRESENTATIONS THROUGH CONTRASTIVE LEARNING

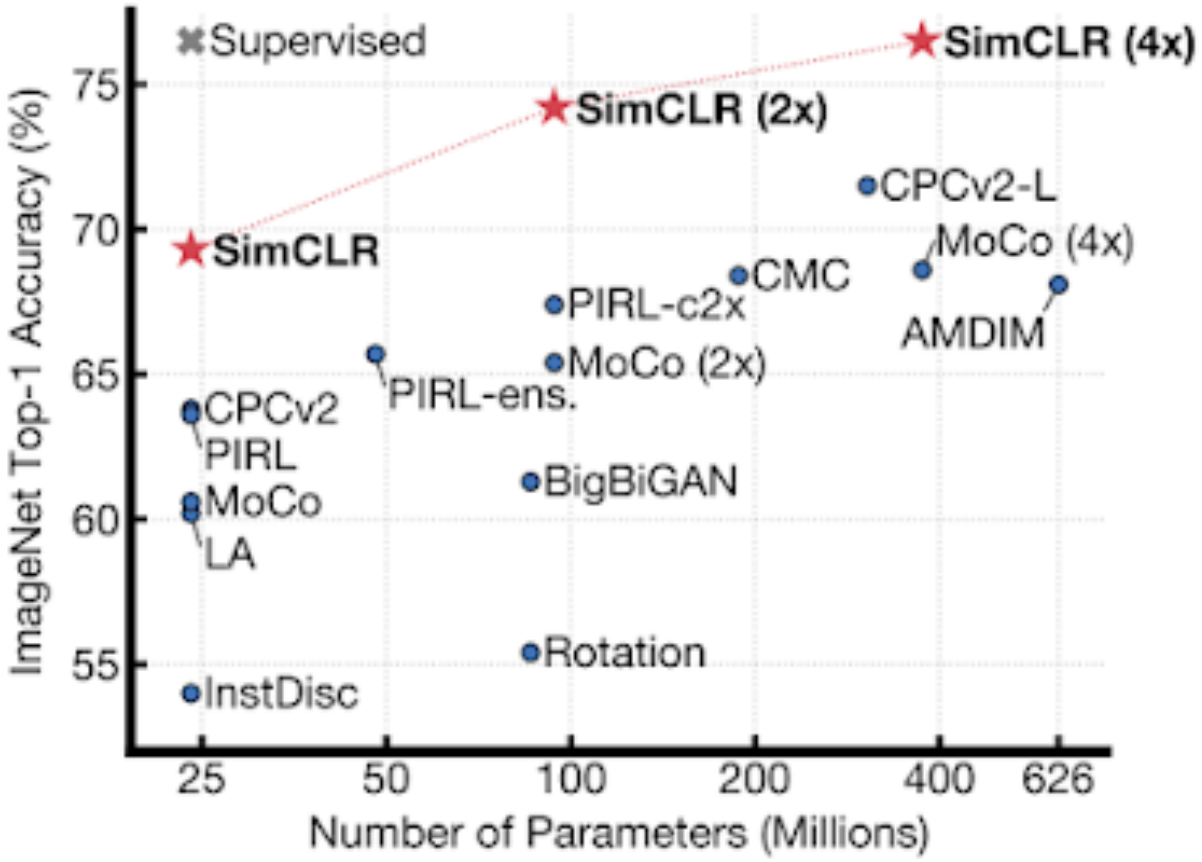
CONTRASTIVE LOSS:

$$l_{i,j} = -\log \frac{\exp(\langle z_i, z_j \rangle / h)}{\Sigma_{k=1,k\neq i}^{2N} \exp(\langle z_i, z_k \rangle / h)},$$

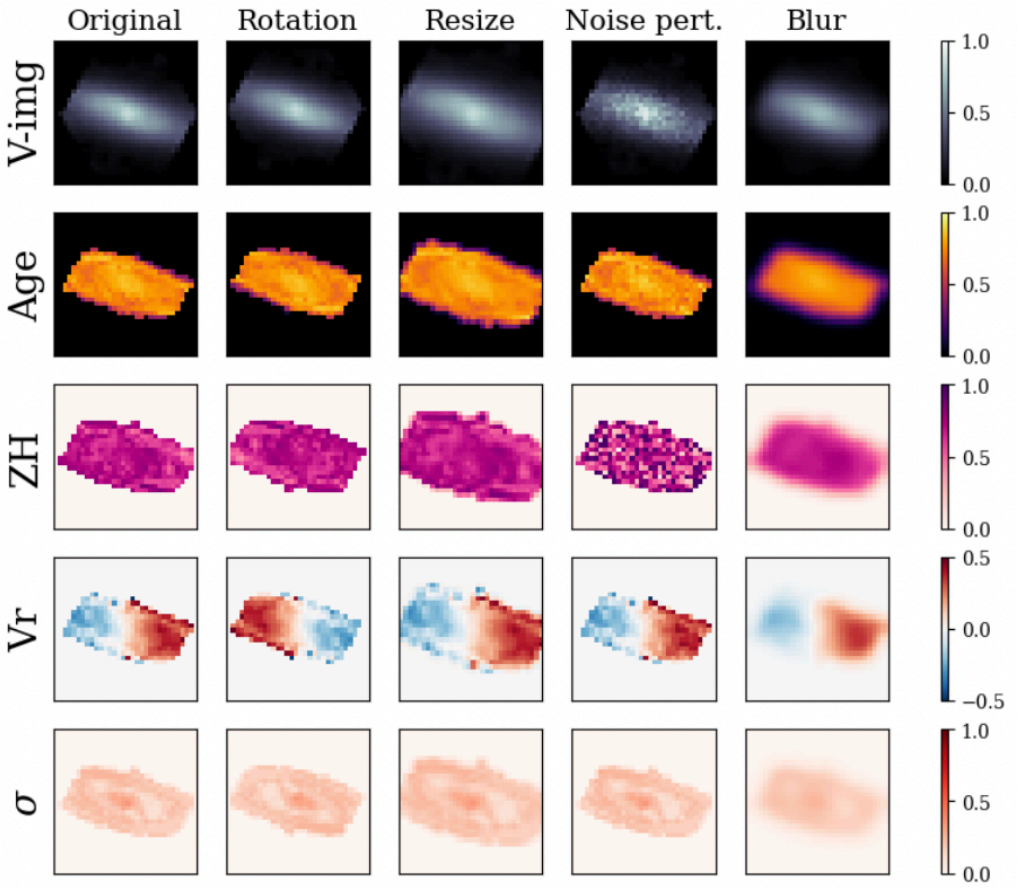


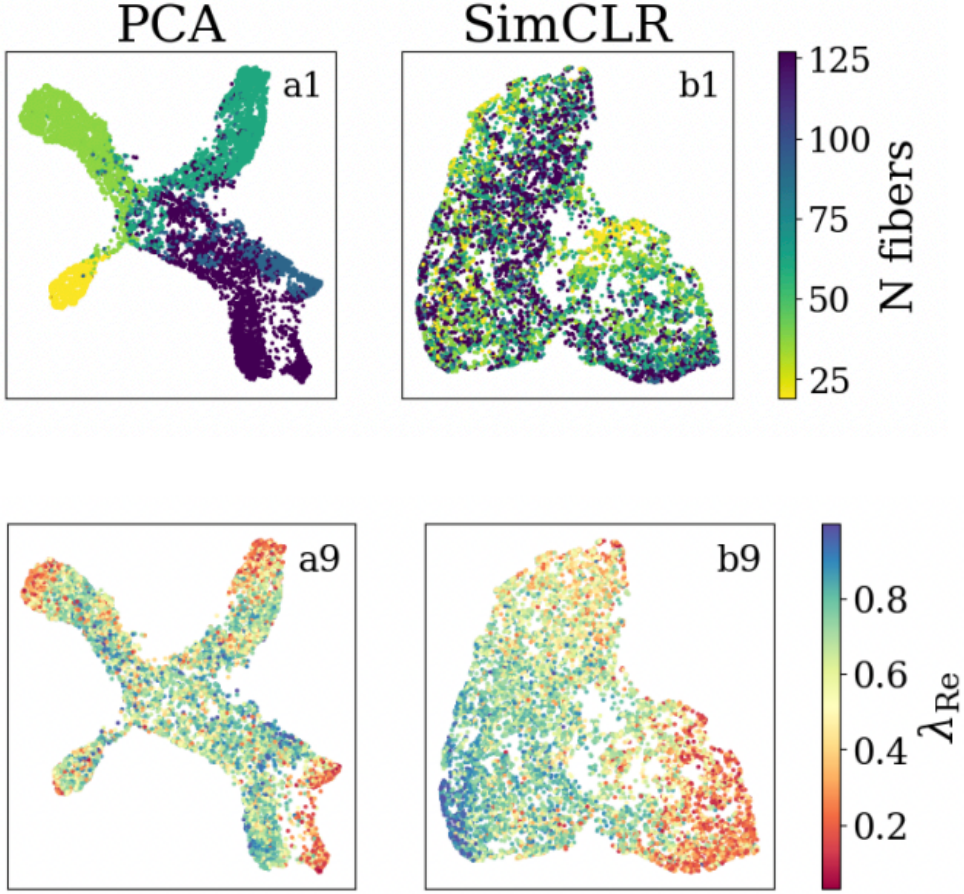
Chen+20

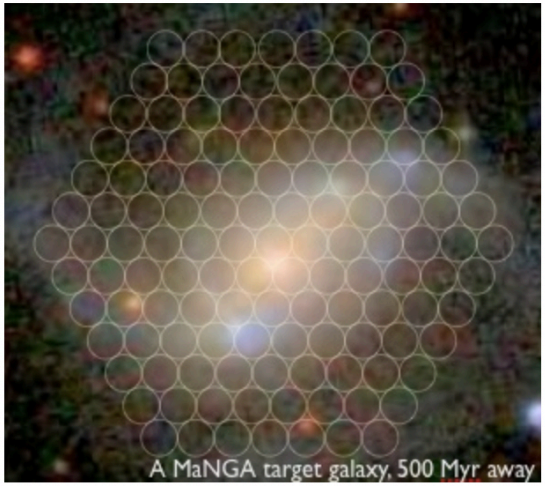
SELF-SUPERVISED LEARNING REACHES COMPARABLE ACCURACY TO FULLY SUPERVISED APPROACHES...



Chen+20







Contrastive learning representation of Manga galaxies

Sarmiento+21