Energy-Based Models

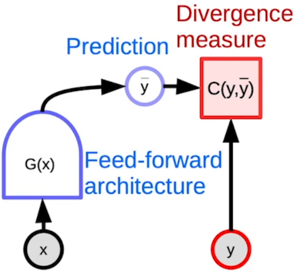
Martin Dimitrov

Feed-forward models

- Observe a set of variables $\mathbf{x}$ and predict a set of variables $\bar{\mathbf{y}}$
- Have a metric that tells you how good your prediction is
- Usually the prediction $\bar{y}$ is found by:

$$\bar{\mathbf{y}} = G(\mathbf{x}\mathbf{W} + \mathbf{b}) \tag{1}$$

$$\text{with } \mathbf{x} = (x_1, \ldots, x_N)^T, \quad \mathbf{W} = (w_1, \ldots, w_N)^T$$



Problems with feed-forward models

- Prediction step is complicated, i.e. it does not involve simply a number sums or the function G is really complicated, or output lives in a continuous space (we need infinite number of iterations or to discretize the space), or even if the output is discrete, the space can be very large (imagine predicting text)

- Multiple possible outputs for a single input?

Energy-Based models

- Instead of producing a prediction $\bar{y}$, we are now going to ask the system to tell us weather a particular x and y are compatible through an implicit function (e.g. is this text which association of that text, is y a good label for image x, is y a high-resolution version of this low-resolution image, etc.)
- Inference - search through values of y and produces a low value of F for a particular x

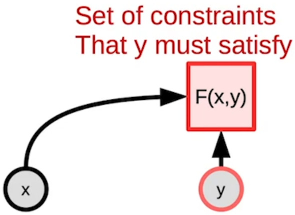


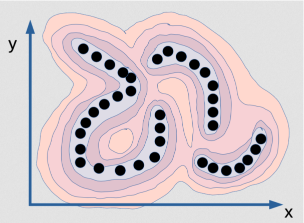
Figure: Simple energy-based model architecture

Energy function

- Energy function is not minimised during learning but minimised during inference, i.e.

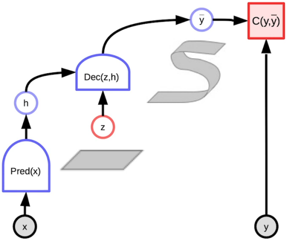
$$\bar{y} = \arg\min_{y} F(x, y) \tag{2}$$

- You want F to have such a shape that all the values of y compatible with x have low energy and all the values of y that are not compatible with a particular x to have high energy
- It would be nice for the energy function to be differentiable so that we can apply gradient-descent methods



Inference with latent variables

- Latent variables are variables whose value is never given to us (if you knew the values of the latent variables, the inference problem becomes easier)
- Example: video prediction (looking through a windshield trying to predict what comes next, or having several translating of a specific language to another)



Latent-Variable EBM: inference

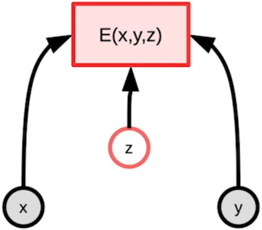
- Simultaneous minimization with respect to y and z:

$$\bar{y}, \bar{z} = \arg\min_{y,z} E(x, y, z) \quad (3)$$

- Redefinition of $F(x, y)$

$$F_\infty(x, y) = \min_z E(x, y, z) \quad (4)$$

$$F_\beta = -\frac{1}{\beta} \log \int_z e^{-\beta E(x,y,z)} \quad (5)$$



Probabilistic models

- Probabilistic models are a special case of EBM (they are not very flexible)

$$\mathbb{P}(y|x) = \frac{e^{-\beta F_\beta(x,y)}}{\int_y e^{-\beta F_\beta(x,y)}} \tag{6}$$

- However, you first need to integrate over the latent variables, i.e. we have:

$$\mathbb{P}(y,z|x) = \frac{e^{-\beta E(x,y,z)}}{\int_y \int_z e^{-\beta E(x,y,z)}} \tag{7}$$

MCMC

Algorithm Metropolis–Hastings Algorithm

Choose X_0 arbitrarily Generate a proposal or candidate value Y $\sim q(y|X_0)$ Evaluate $r \equiv r(X_0, Y)$ where:

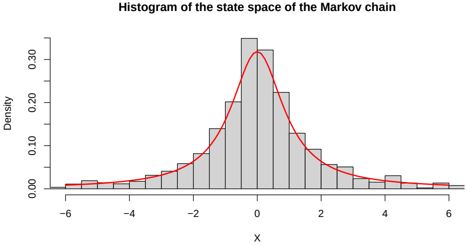
$$r(x, y) = \min\left(\frac{f(y)q(x|y)}{f(x)q(y|x)}, 1\right) \tag{8}$$

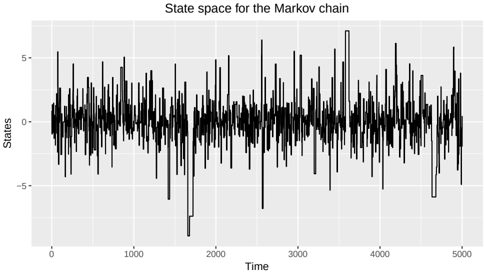
Set:

$$X_1 = \begin{cases} Y & \text{with probability } r \\ X_0 & \text{with probability 1-}r \end{cases} \tag{9}$$

Go to step 2 with X_0 substituted with X_1

MCMC - continued





Training

- Let's say we want to maximise the probability of a particular configuration y, i.e.:

$$\text{maximise } \mathbb{P}(Y|W) = \frac{e^{-\beta E(Y,W)}}{\int_y e^{-\beta E(y,W)}} \tag{10}$$

- Instead, minimize $-\log \mathbb{P}(Y, W)$:

$$L(Y, W) = E(Y, W) + \frac{1}{\beta} \log \int_y e^{-\beta E(y,W)} \tag{11}$$

Gradient of the negative log-likelihood loss for one sample Y

$$\frac{\partial L(Y,W)}{\partial W} = \frac{\partial E(Y,W)}{\partial W} - \int_y P(y|W)\frac{\partial E(y,W)}{\partial W} \tag{12}$$

$$W \leftarrow W - \gamma\frac{\partial L(Y,W)}{\partial W} \tag{13}$$

End

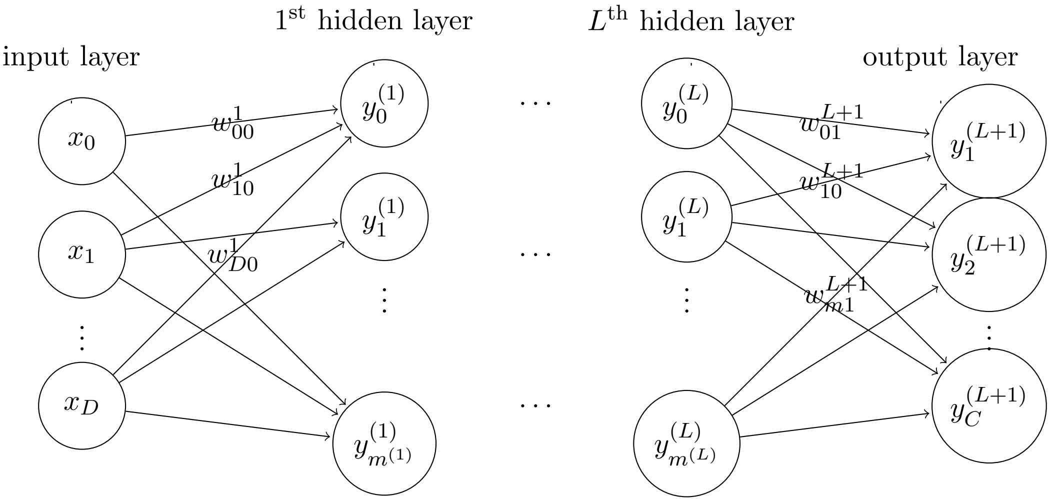
Thank you for your attention!

Graph Neural Networks

Gergana Belcheva

9th July, 2022

NN intro

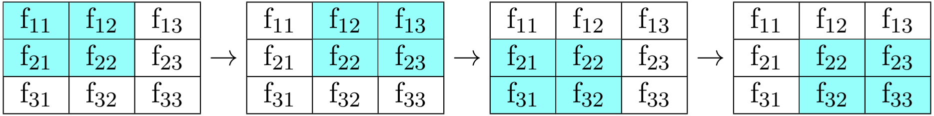


1st hidden layer
Lth hidden layer
input layer
output layer
x_0
x_1
x_D
w^1_{00}
w^1_{10}
w^1_{D0}
$y_0^{(1)}$
$y_1^{(1)}$
$y_{m^{(1)}}^{(1)}$
$y_0^{(L)}$
$y_1^{(L)}$
$y_{m^{(L)}}^{(L)}$
w^{L+1}_{01}
w^{L+1}_{10}
w^{L+1}_{m1}
$y_1^{(L+1)}$
$y_2^{(L+1)}$
$y_C^{(L+1)}$

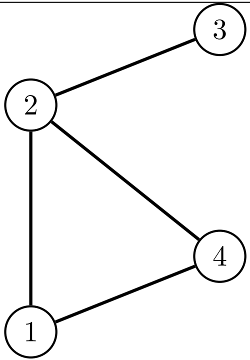
Convolution for Linear Data (Template Convolution)

$$
\begin{pmatrix} f_{11} & \cdots & f_{1m} \\ \vdots & \ddots & \vdots \\ f_{n1} & \cdots & f_{nm} \end{pmatrix} * \begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} =
$$

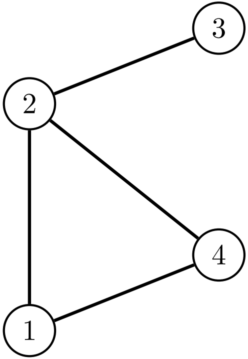
$$
\begin{pmatrix} \begin{matrix} f_{11}g_{11} + f_{12}g_{12} + \\ f_{21}g_{21} + f_{22}g_{22} \end{matrix} & \cdots & \begin{matrix} f_{1m-1}g_{11} + f_{1m}g_{12} + \\ f_{2m-1}g_{21} + f_{2m}g_{22} \end{matrix} \\ \vdots & \ddots & \vdots \\ \begin{matrix} f_{n-11}g_{11} + f_{n-12}g_{12} + \\ f_{n1}g_{21} + f_{n2}g_{22} \end{matrix} & \cdots & \begin{matrix} f_{n-1m-1}g_{11} + f_{n-1m}g_{12} + \\ f_{nm-1}g_{21} + f_{nm}g_{22} \end{matrix} \end{pmatrix}
$$



Graph Theory

Linear, ordered	Non-linear, arbitrarily indexed
$\begin{pmatrix} a_{11} & \dots & a_{1m} \\ a_{21} & \dots & a_{2m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nm} \end{pmatrix}$	

Graph Theory



Adjacency Matrix

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Degree matrix

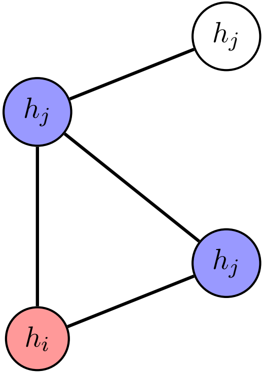
$$D = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

Laplacian Matrix

$$L = D - A = \begin{pmatrix} 2 & -1 & 0 & -1 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 1 & 0 \\ -1 & -1 & 0 & 2 \end{pmatrix}$$

Graph Theory

Normalized Laplacian:
$\Delta = I - D^{-1/2} A D^{-1/2}$



$$(\Delta h)_i = h_i - \frac{1}{d_i} \sum_j a_{ij} h_j, \text{ where } d \in D, a \in A$$

Eigen-decomposition of the Laplacian Matrix

$$\Delta = \Phi^T \overbrace{\Lambda}^{\substack{\text{Eigenvalues/}\\ \text{Spectrum}}} \underbrace{\Phi}_{\substack{\text{Eigenvectors/}\\ \text{Fourier functions/}\\ \text{Orthonormal basis}}}$$

Fourier Transform

Decomposition of h into Fourier series:

$$h = \sum_k (\Phi_k, h)\Phi_k = \sum_k \hat{h}_k \Phi_k = \Phi\hat{h} = \mathcal{F}^{-1}(\hat{h})$$

or

$$\mathcal{F}^{-1}(\hat{h}) = \Phi\hat{h}$$

$$\mathcal{F}(h) = \Phi^T h$$

Spectral Convolution

Convolution Theorem: $\boxed{\mathcal{F}(w * h) = \mathcal{F}(w)\mathcal{F}(h)}$

$$w * h = \mathcal{F}^{-1}(\mathcal{F}(h)\mathcal{F}(w)) = ... = \underbrace{\hat{w}(\Delta)}_{\substack{\text{Spectral}\\ \text{function/}\\ \text{Frequency}\\ \text{filter}\\ n \times n}} h$$

where

$\Delta = \Phi^T \Lambda \Phi$ - the Eigen-decomposition of the Laplacian matrix.

$$w * h = \hat{w}(\Delta) h$$

is called spectral convolution.

Spline Optimization

$\hat{w}$ is learned by back propagation. However

- it may not be localized
- learning complexity is $O(n^2)$

Decomposing the spectral filter into a linear combination of K smooth kernels B:

$$\hat{w}^l(\Lambda) = diag(Bw^l)$$

gives a smooth spectral filter which is localized in the original space domain of the graph and learning complexity is O(K) - constant.

Thanks

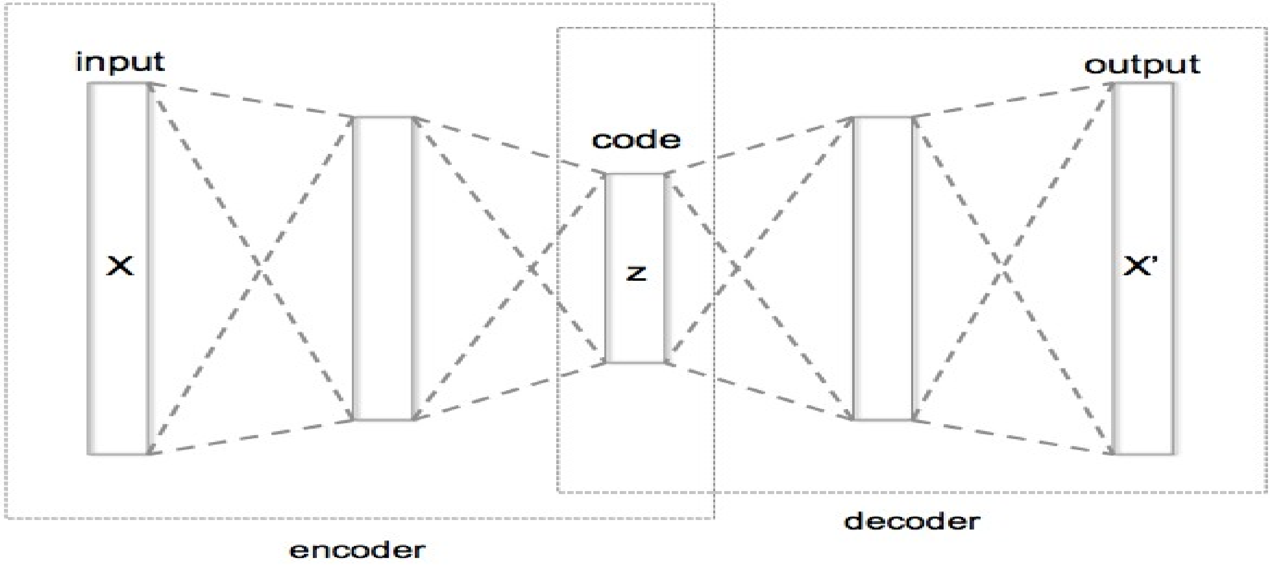
Radoslav Simeonov, Valentin Buchakchiev

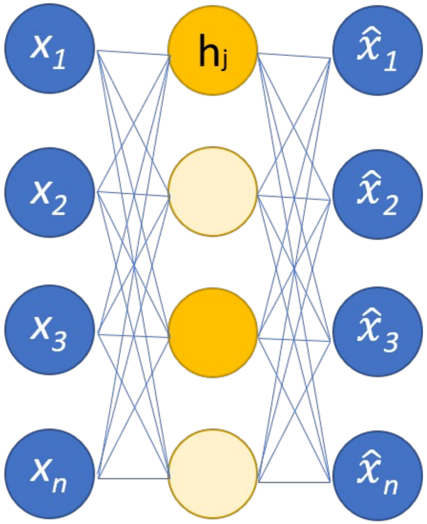
THE AUTOENCODERS

09.07.2022

NAO Rozhen

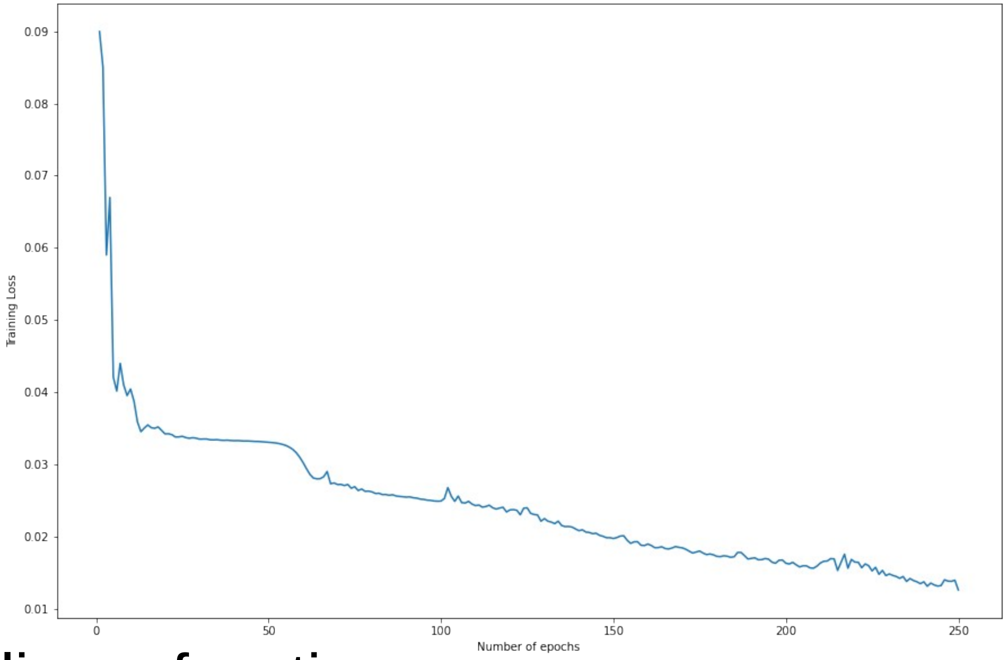
- type of artificial neural network used to learn efficient codings of unlabeled data (unsupervised learning)
- Two main parts
  - an encoder that maps the input into the code
  - decoder that maps the code to a reconstruction of the input

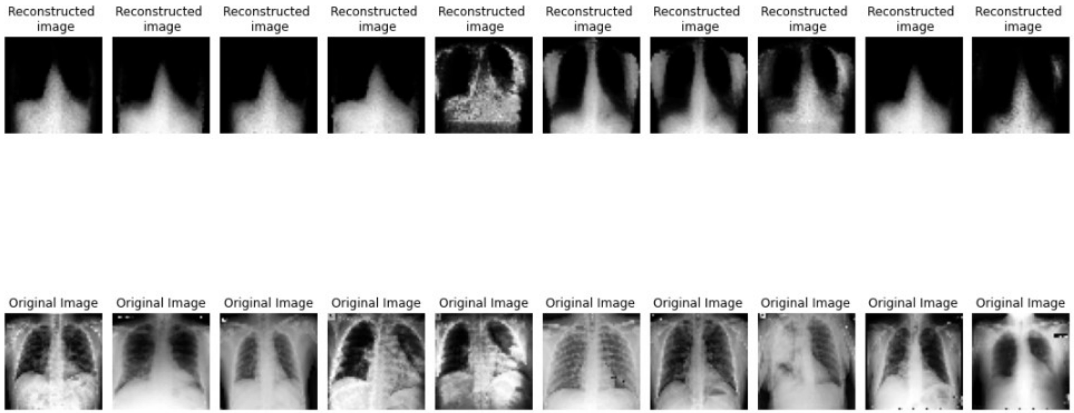




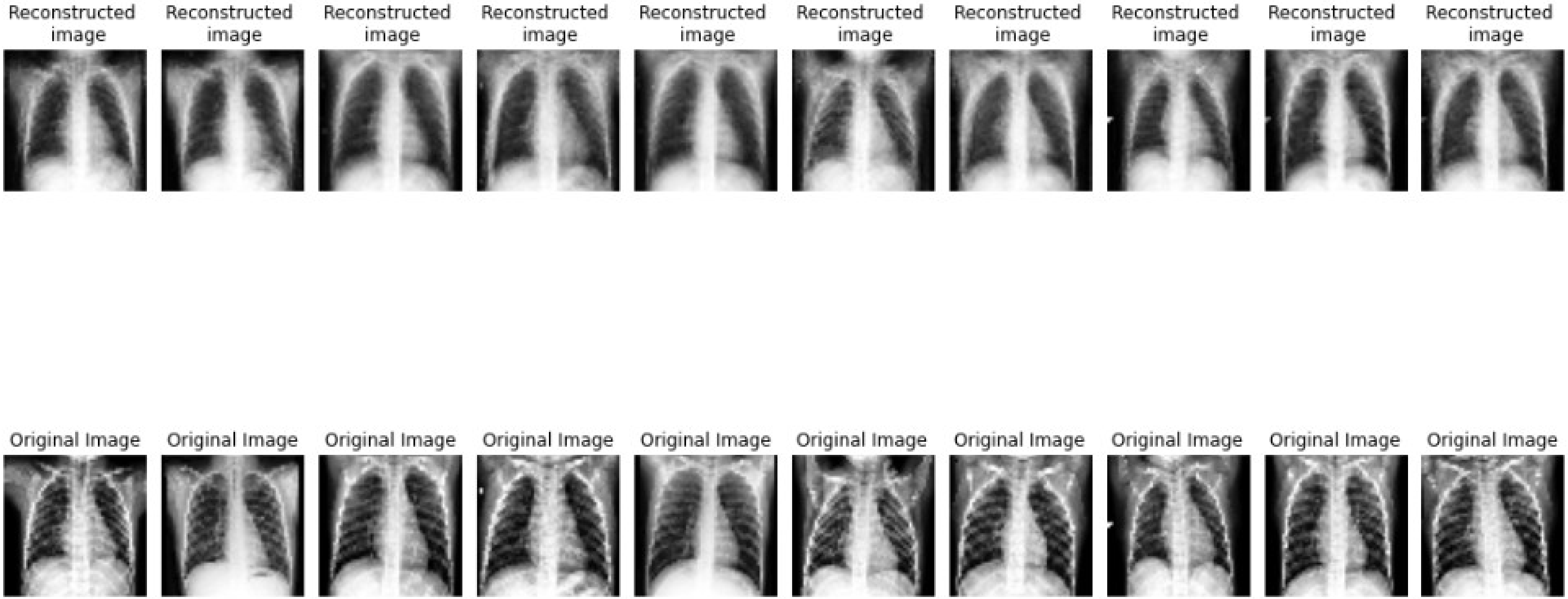
Simple schema of a single-layer sparse autoencoder. The hidden nodes in bright yellow are activated, while the light yellow ones are inactive. The activation depends on the input.

- Set of images – 80
- Resizing in terms of pixels
  - 50x50 pixels
- Layers used
  - Total number of 5 layers
  - 2500 – 1024 – 512 – 256
  - Combination of linear and non-linear functions

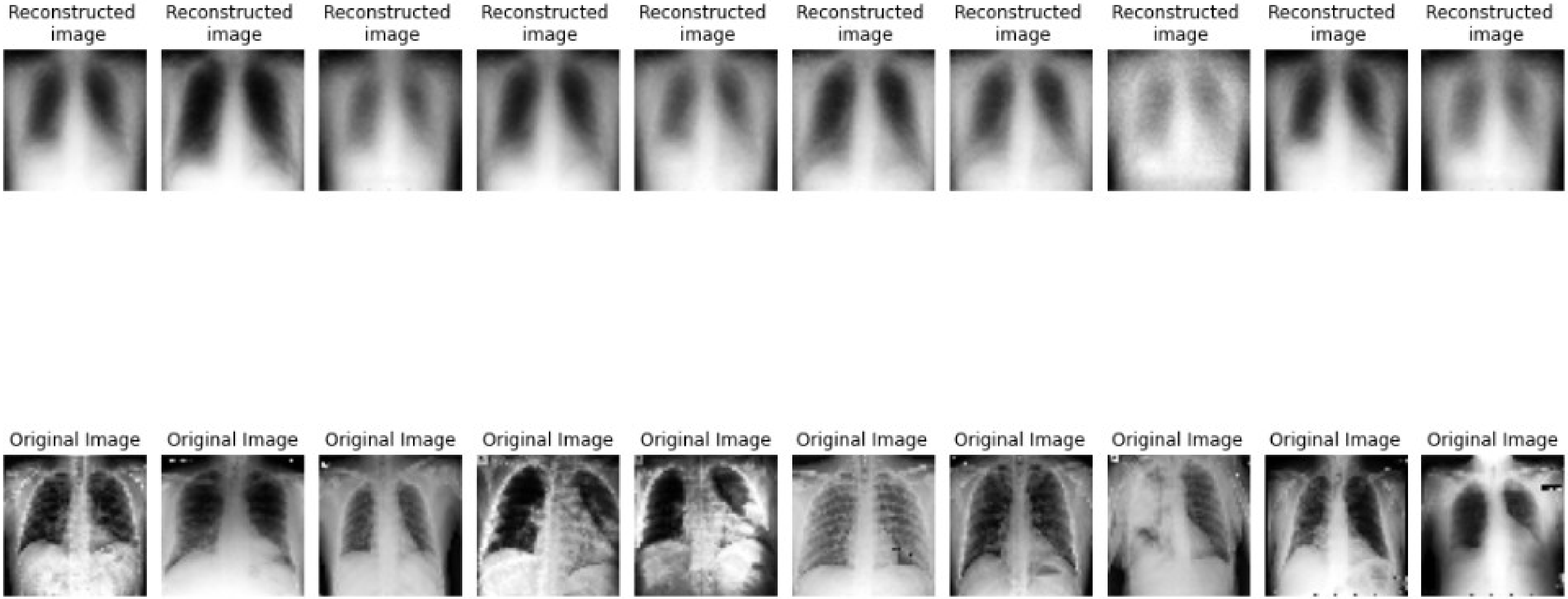




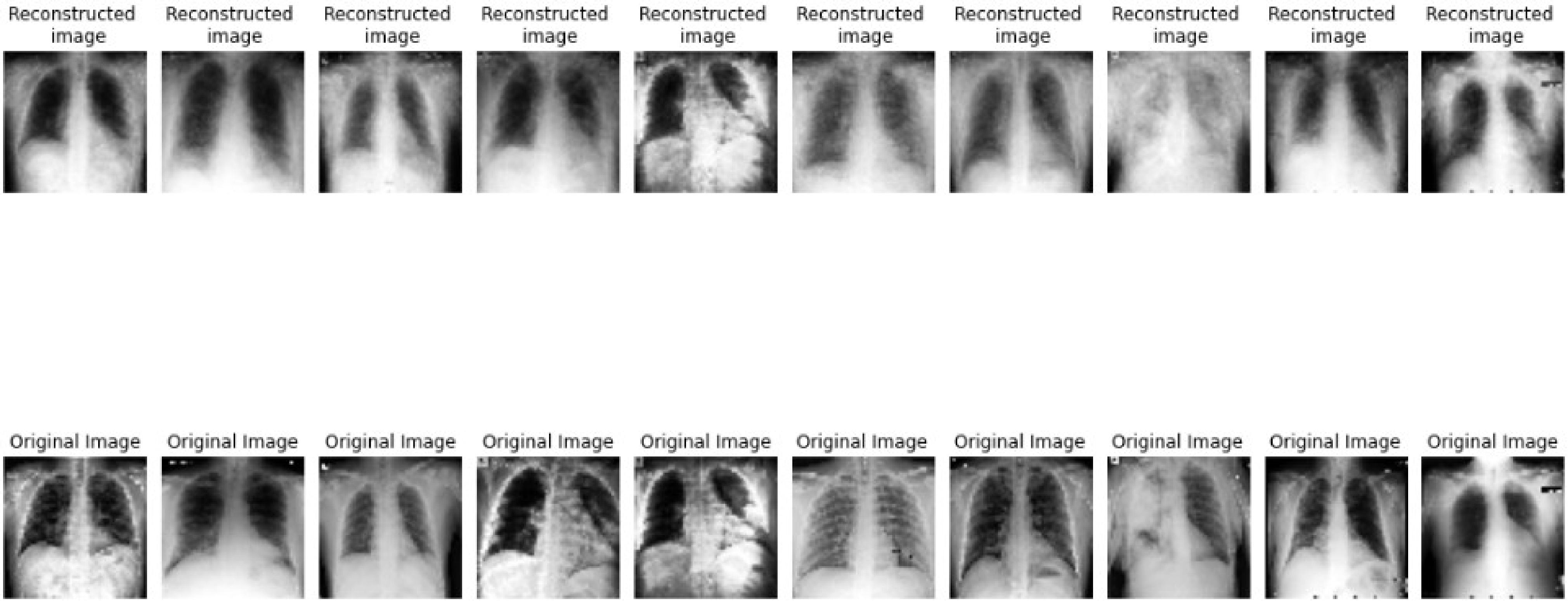
Healthy lungs – 250 epochs



Covid Lungs – 100 epochs



Covid Lungs – 250 epochs



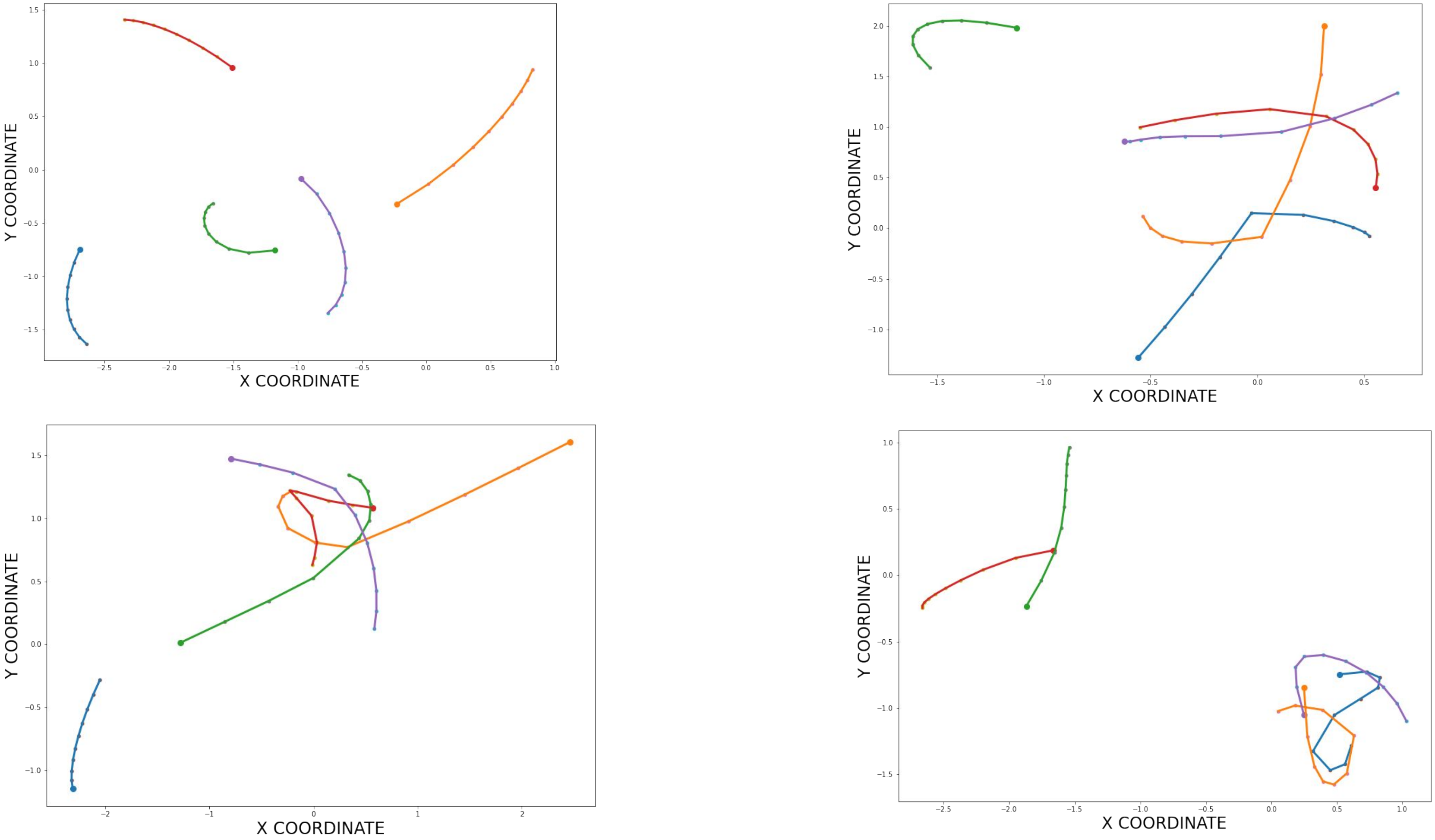
N-body simulator for data creation for Graph neural networks

```python
# create the header of the output file
with open(out_filepath, 'w') as f:
    writer = csv.writer(f)
    writer.writerow(['N_sim', 'N_star', 'time_step', 'x', 'y', 'z', 'vx', 'vy', 'vz'])

# Simulation parameters
sim_repetitions = 1 # how many times should we create a new simulation
for num_rep in range(sim_repetitions):

    N         = 5    # Number of particles
    t         = 0      # current time of the simulation
    tEnd      = 1.0   # time at which simulation ends
    dt        = 0.1   # timestep
    softening = 0.1    # softening length
    G         = 1.0    # Newton's Gravitational Constant
    plotRealTime = False #True # switch on for plotting as the simulation goes along
    x,y,z     = [], [], [] # the position columns to be used for saving the data to file
    vx,vy,vz  = [], [], [] # the velocity columns to be used for saving the data to file
    t_col     = [] # the time column to be used for saving to file
    num_repcol= [] # the column containing the number of the simulatiuon repetition
```

N-body simulator for data creation for Graph neural networks



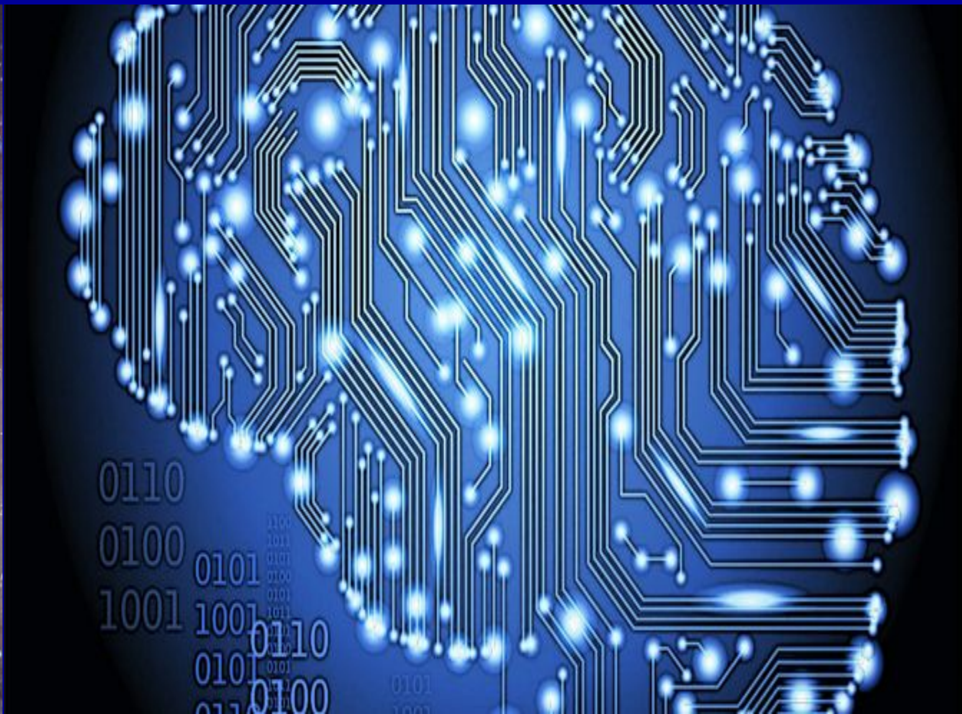
The 2nd Summer School on Space Research, Technology and Application for Young Scientists and PhD Students (3rd – 10th July 2022, Rozhen, Bulgaria)

Svetoslav Georgiev Botev

(*PhD student in* **St. Kliment Ohridski University of Sofia**)

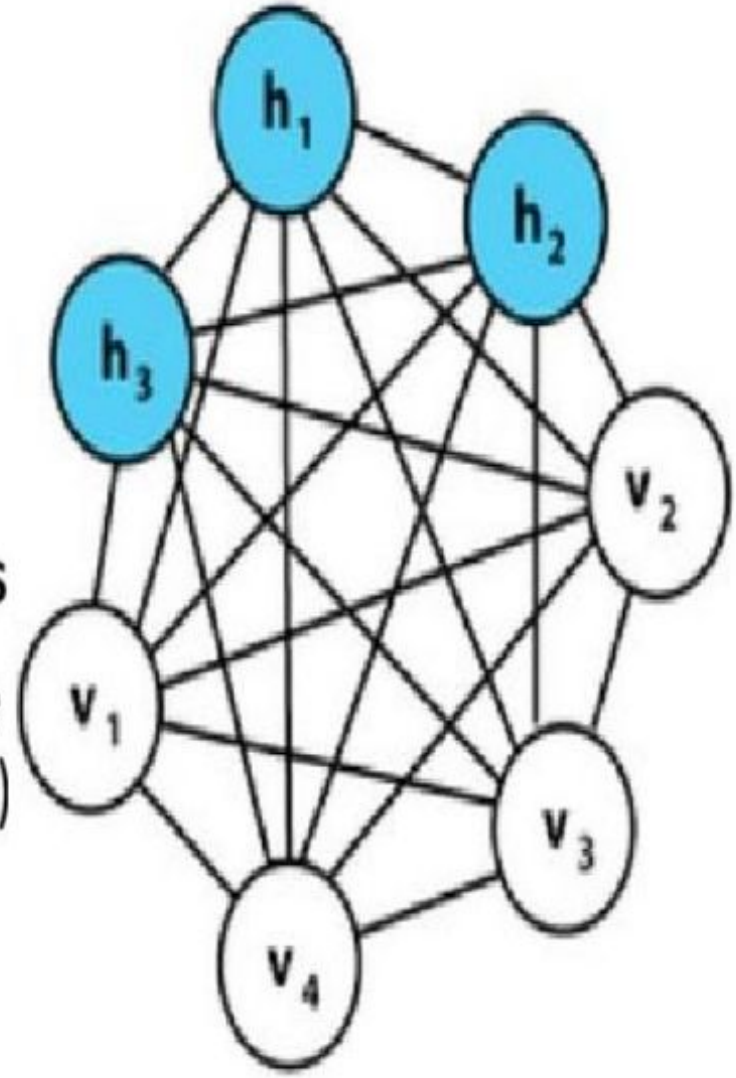
e-mail: sgbvtbg@gmail.com

Course: **Machine Learning**



Boltzmann Machine (BM)

- A type of stochastic recurrent neural network (form of Markov Random Field) invented by Geoffrey Hinton and Terry Sejnowski in 1985
- Can be seen as the stochastic, generative counterpart of Hopfield nets
- Based on the mathematics of **thermodynamics**
- Complex training algorithm that does not scale well (grows exponential with number of nodes)
- Chracteritstics
  - Unsupervised learning
  - Associative memory
  - Long training time



- Energy function

binary state of visible unit i

binary state of hidden unit j

$$E(v,h) = -\sum_{i,j} v_i h_j w_{ij}$$

Energy with configuration v on the visible units and h on the hidden units

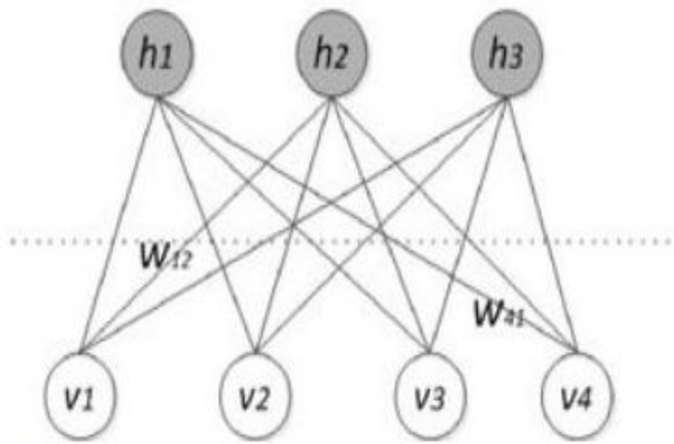
weight between units i and j

- The energy of a joint configuration of the visible and hidden units determines its probability:

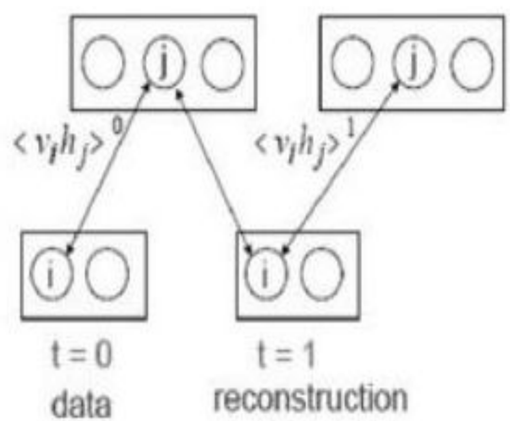
$$p(v,h) \propto e^{-E(v,h)}$$

Restricted Boltzmann Machine (RBM)

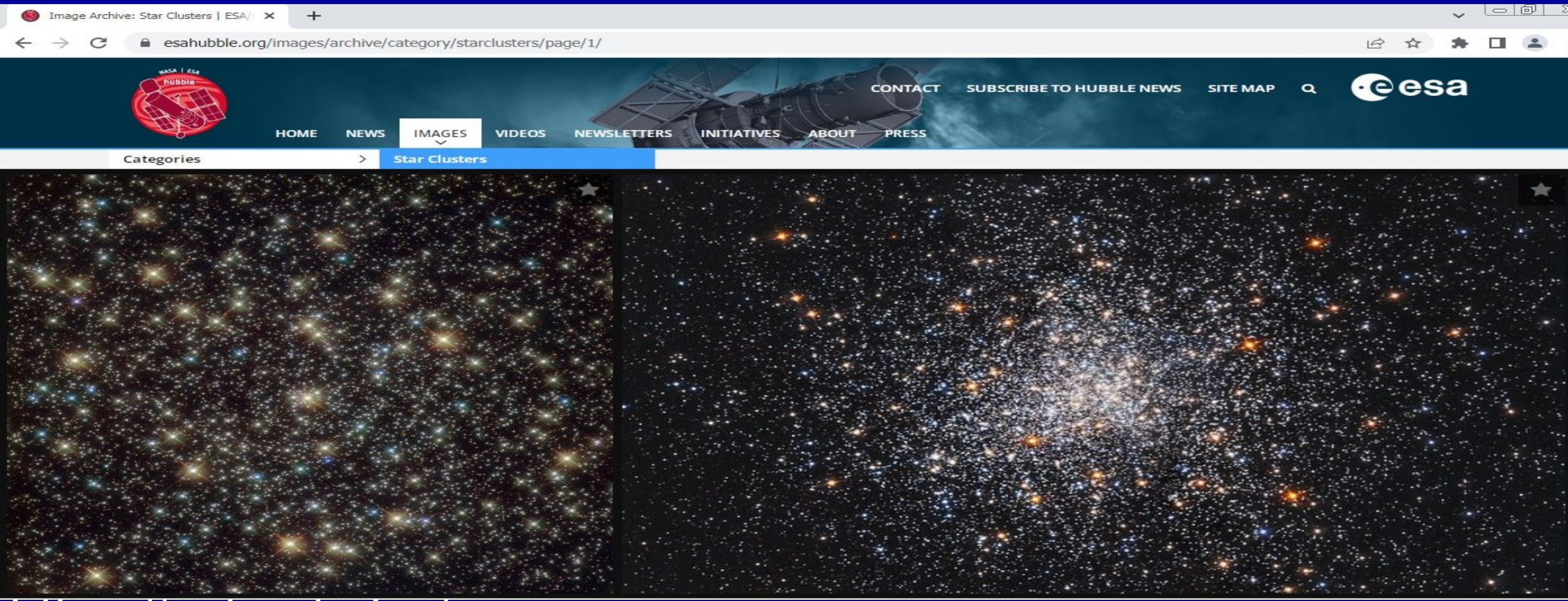
- A generative stochastic artificial neural network that can learn a probability distribution over its set of inputs.
- Restrict number of iterations in + and – phases
- Restrict connectivity of network
- Does not allow intra-layer connections connections (bipartite architecture)



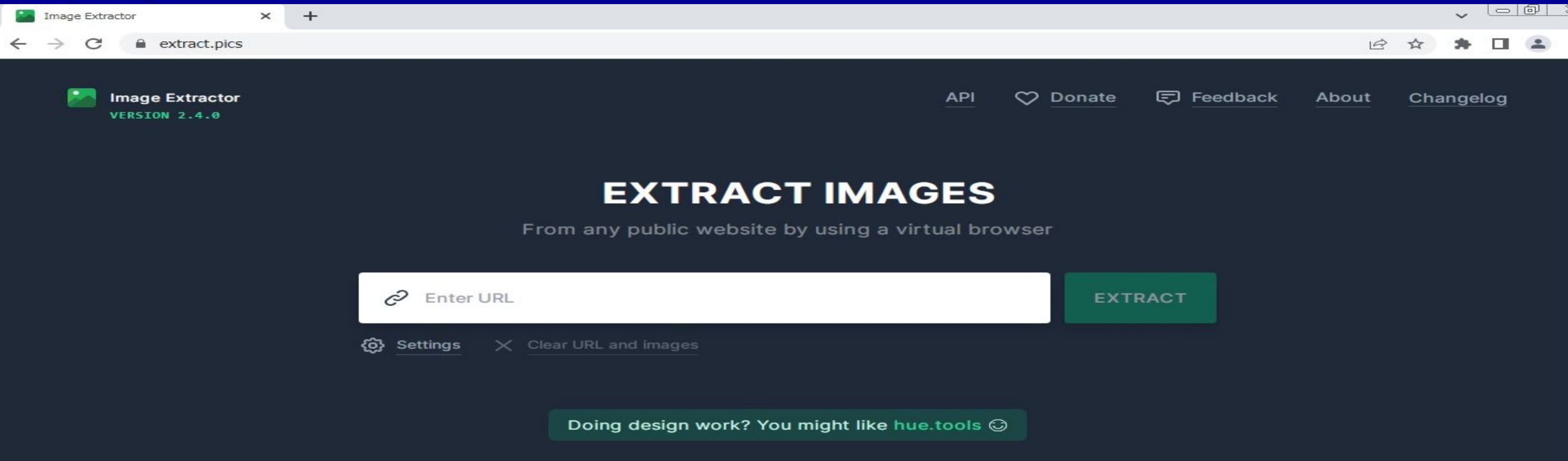
$$\Delta w_{ij} = \varepsilon \left(\langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^1 \right)$$



- Learning both recognition model and generative model https://esahubble.org/images/archive/category/starclusters/page/1

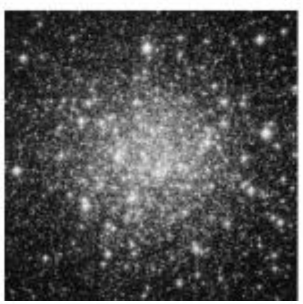


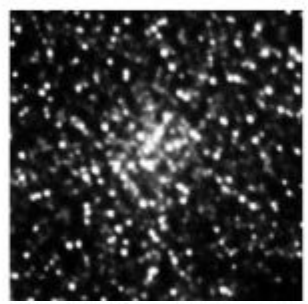
https://extract.pics/



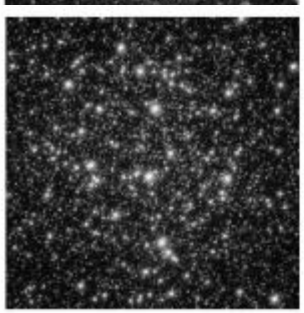
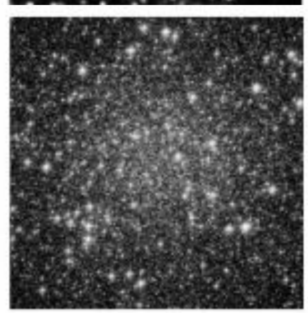
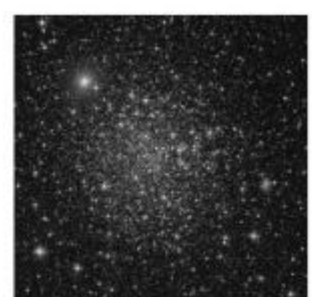
Restricted Boltzmann Machine (RBM) displays all 90 globular clusters from the chosen sample in 1 raw:

Then RBM represents the globular clusters in colour in their real size, namely 300 pixels:





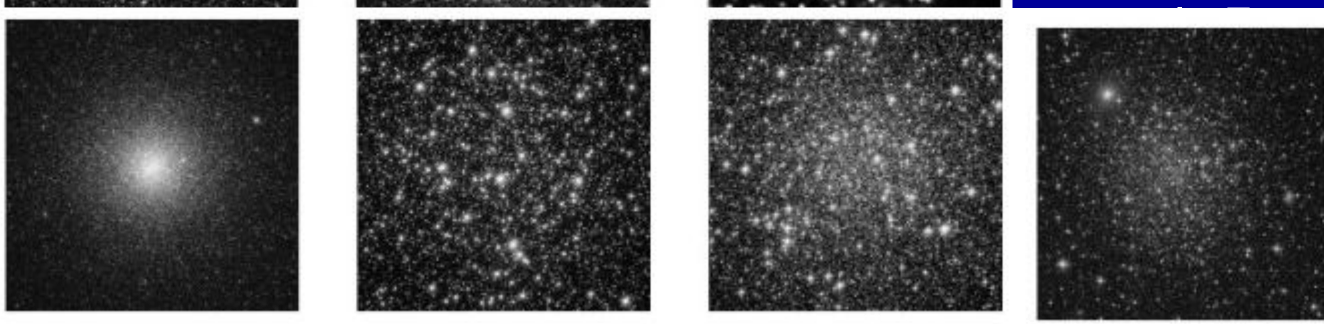
After that, RBM resizes the images to 255 pixels each and also converts them from coloured images into black-white ones and randomly can display arbitrary number of them, for example 7:

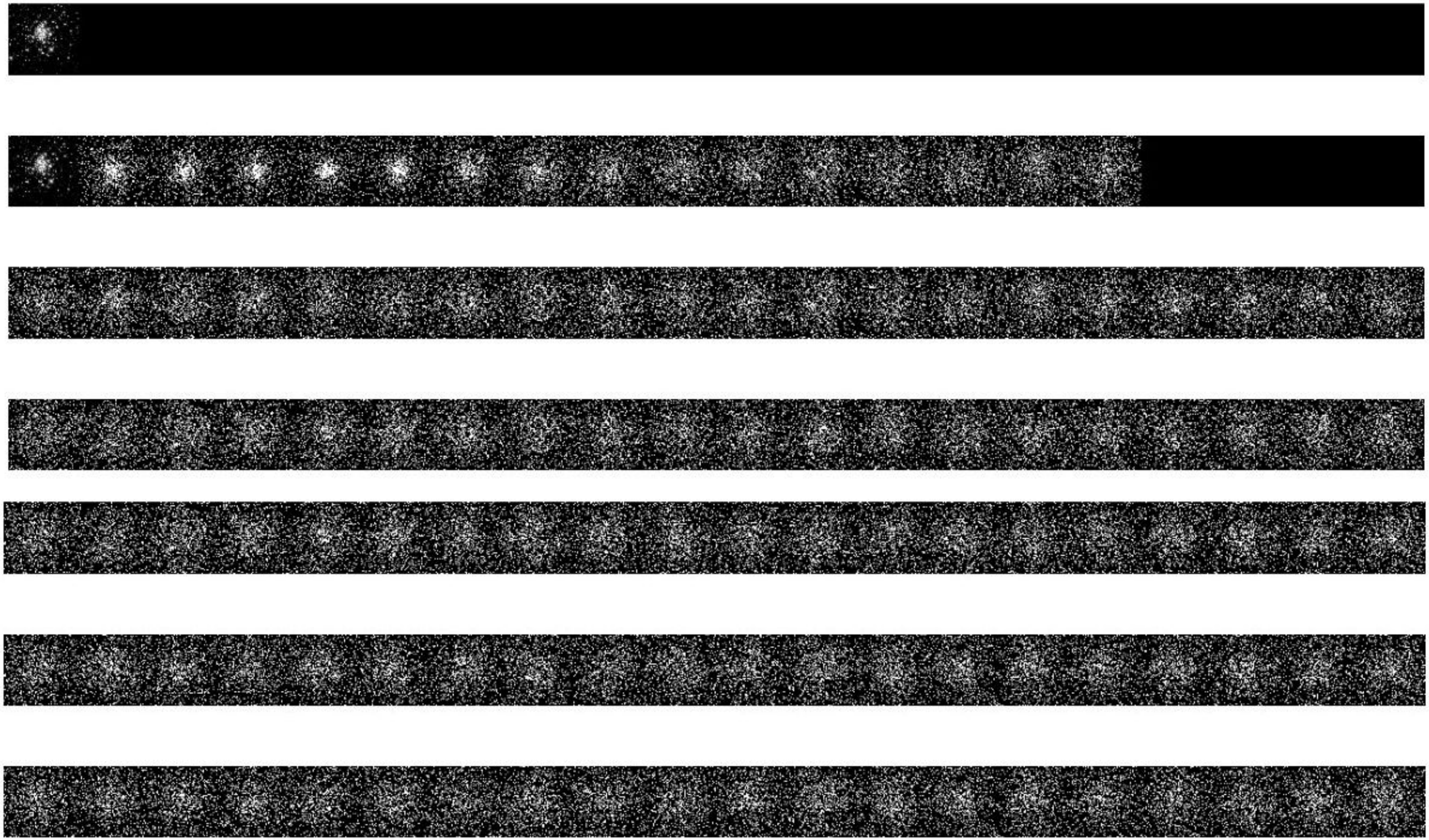
Restricted Boltzmann Machine (RBM) displays all 90 globular clusters from the chosen sample in 1 raw:

Then RBM represents the globular clusters in colour in their real size, namely 300 pixels:

After that, RBM resizes the images to 255 pixels each and also converts them from coloured images into black-and-white ones and randomly can display arbitrary number of them, for



But more interesting is the fact that the RBM can be learnt to create their own artificial images of the globular clusters on the basis of one or more of the original ones. For instance, you can see in the figure below how RBM in the beginning has taught an image and then, learning it, RMB can generate unlimited number of similar but not real images of globular clusters:



Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY
MNRAS **504**, 5656–5670 (2021) doi:10.1093/mnras/stab737
Advance Access publication 2021 March 19

Predicting images for the dynamics of stellar clusters (π -DOC): a deep learning framework to predict mass, distance, and age of globular clusters

Jonathan Chardin★ and Paolo Bianchini
Observatoire Astronomique de Strasbourg, Université de Strasbourg, CNRS UMR 7550, 11 rue de l'Université, F-67000 Strasbourg, France

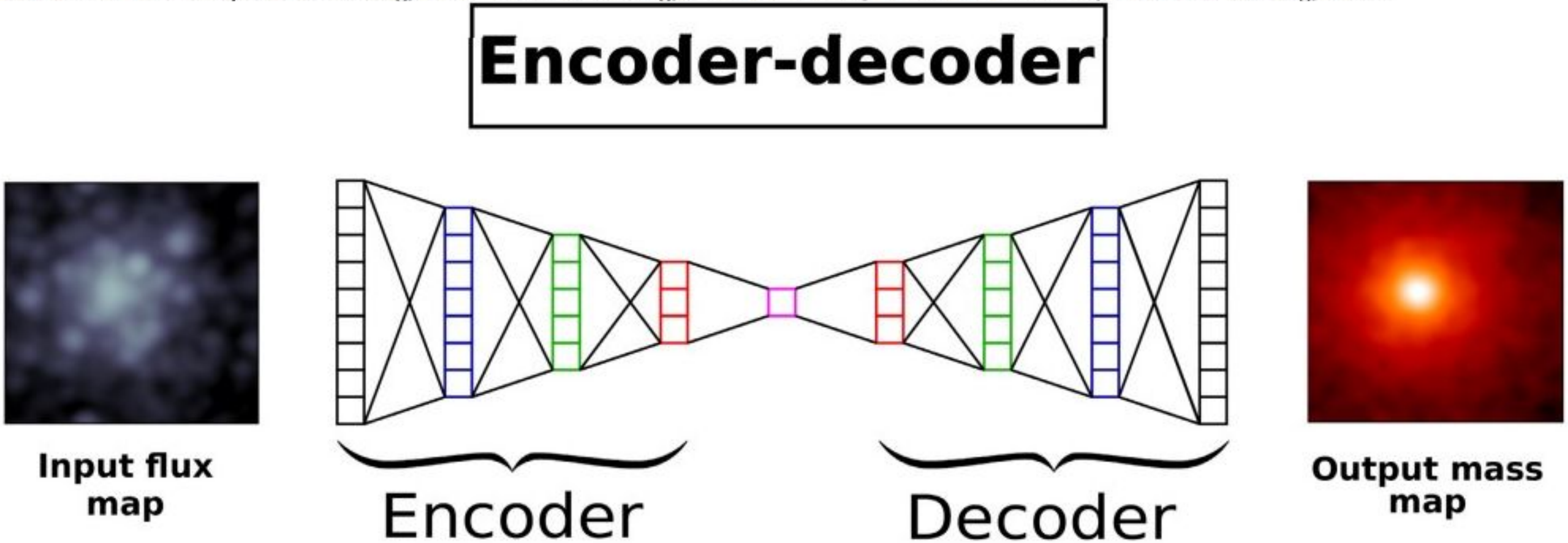


Figure 1. Architecture of the convolutional encoder–decoder part of π-DOC. This first part of the network takes a GC flux map as an input and gives a mass map as an output. Images have 160 × 160 pixels sizes both for the input and the output. The encoder (decoder) are built with four convolutions (deconvolutions) hidden layers.

Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY
MNRAS **504**, 5656–5670 (2021) doi:10.1093/mnras/stab737
Advance Access publication 2021 March 19

Predicting images for the dynamics of stellar clusters (π-DOC): a deep learning framework to predict mass, distance, and age of globular clusters

Jonathan Chardin★ and Paolo Bianchini
Observatoire Astronomique de Strasbourg, Université de Strasbourg, CNRS UMR 7550, 11 rue de l'Université, F-67000 Strasbourg, France

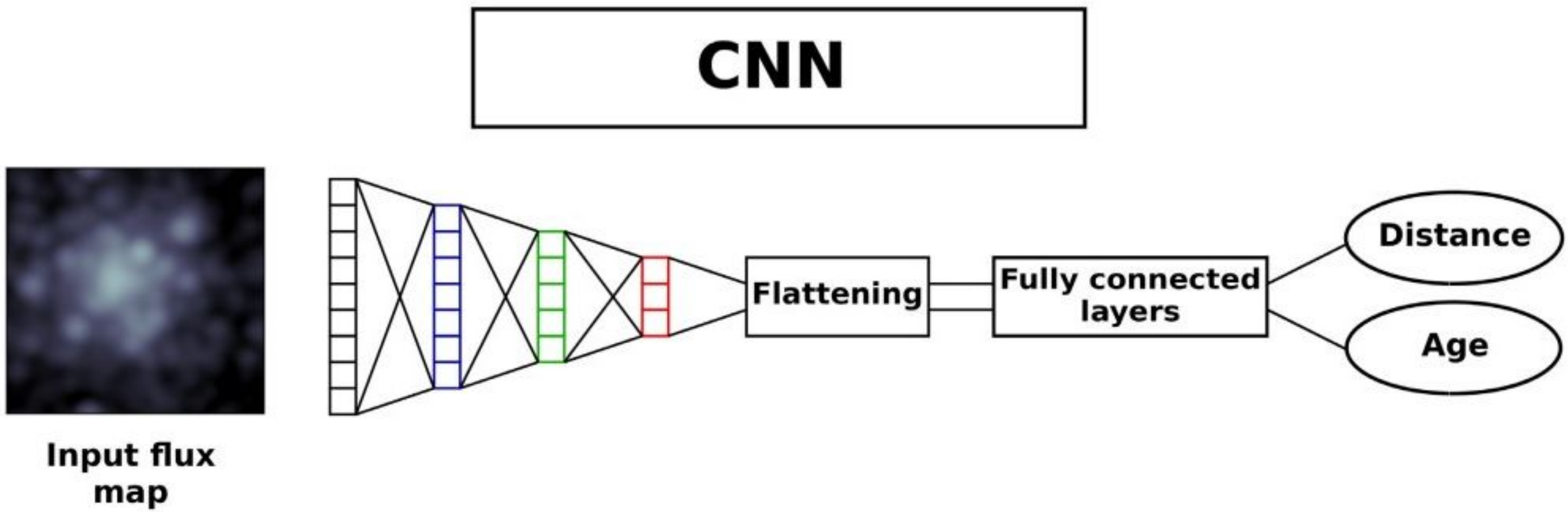


Figure 2. Architecture of the π-DOC convolutional neural network (CNN) to predict the age and distance of clusters from their observed luminosity maps.

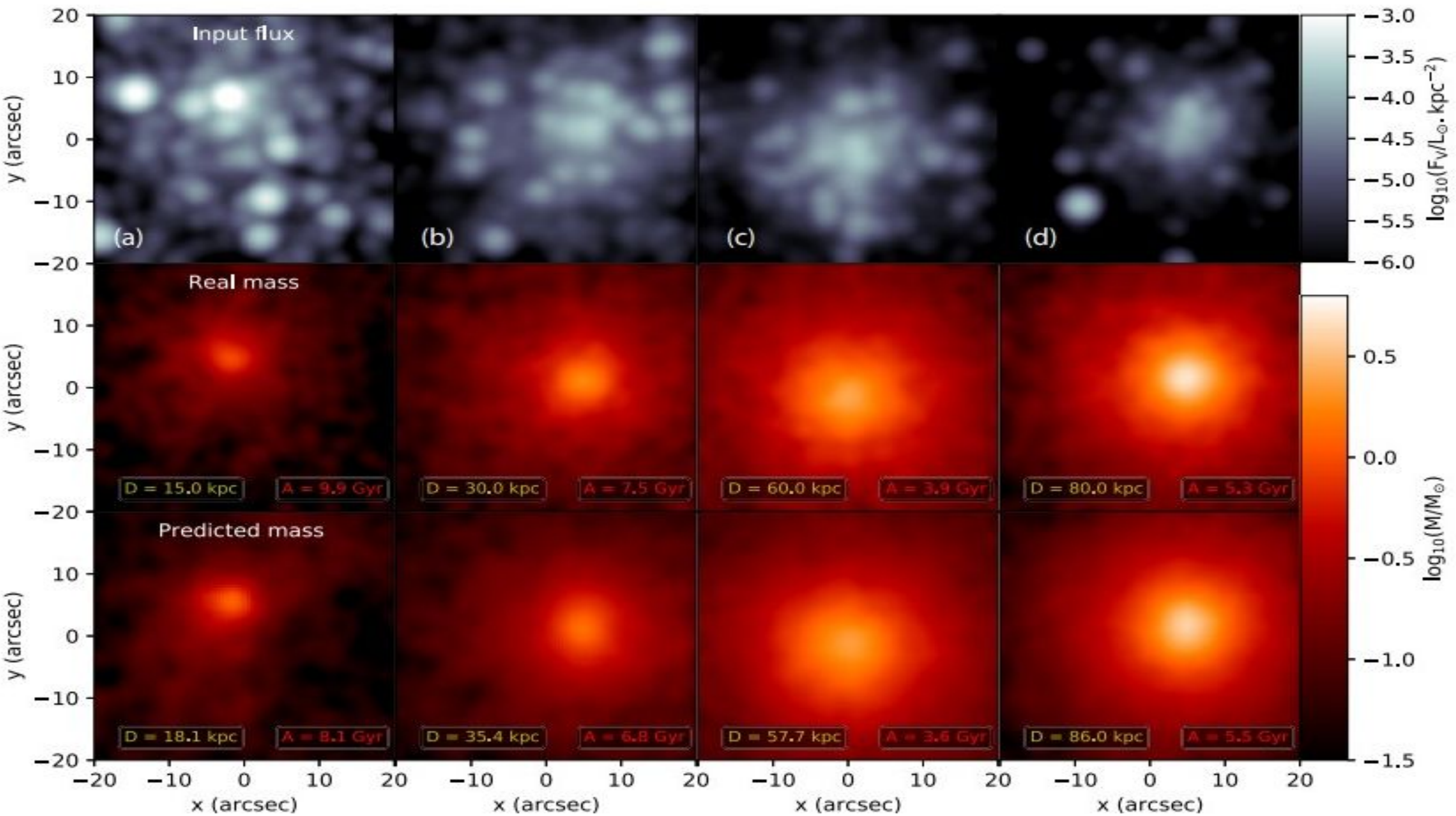


Figure 3. Full prediction of π-DOC. First row: example of four input luminosity maps from the original simulation. Second row: example of four mass distribution maps from the original simulation corresponding to the luminosity maps from the first row. Third row: example of four mass map predicted by π-DOC by taking the corresponding luminosity maps from the first row as input of the encoder–decoder part. On the different panels of the second and third row are indicated the distances and ages.

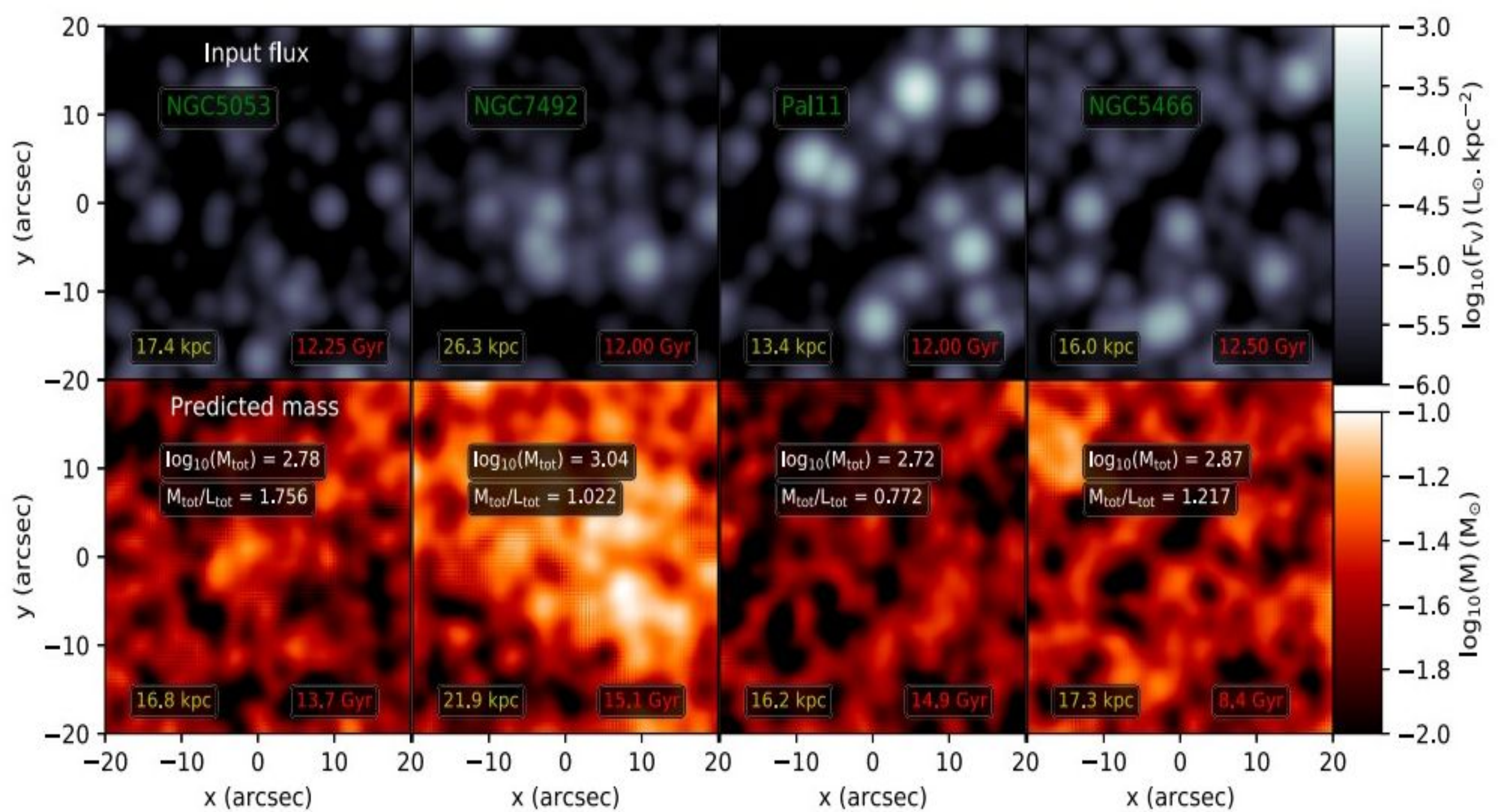


Figure 4. Predictions of π-DOC for the four least luminous GCs (NGC 5053, Pal 11, NGC 7492, NGC 5466) for which M/L, age, and distance values consistent with the literature are retrieved. The first row reports the input V-band flux from PS1 observations, with associated age and distances. The second row reports the mass maps obtained by π-DOC, the associated mass and M/L in the FoV and the age and distance predictions.

Thank you!