Variational Auto Encoders

Meeting 1 - 09/05/23

Group 9 - Bacilieri, Barbiero, Bordin, Pitteri

Generating the dataset

Potential Energy

Fluid in a cubic container of side L

Potential energy: - gravitational potential $U_g = mgr_z$

- Lennard Jones Potential $U_{LJ} = 4\epsilon[(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^6]$

A change of variables is performed $r' = \frac{r}{\sigma}$

And so $U' = 4[(\frac{1}{r'})^{12} - (\frac{1}{r'})^6] - \frac{mg\sigma}{\epsilon} r'_z$

The only trainable parameter is $\gamma = \frac{mg\sigma}{4\epsilon}$

Generating the dataset

Algorithm

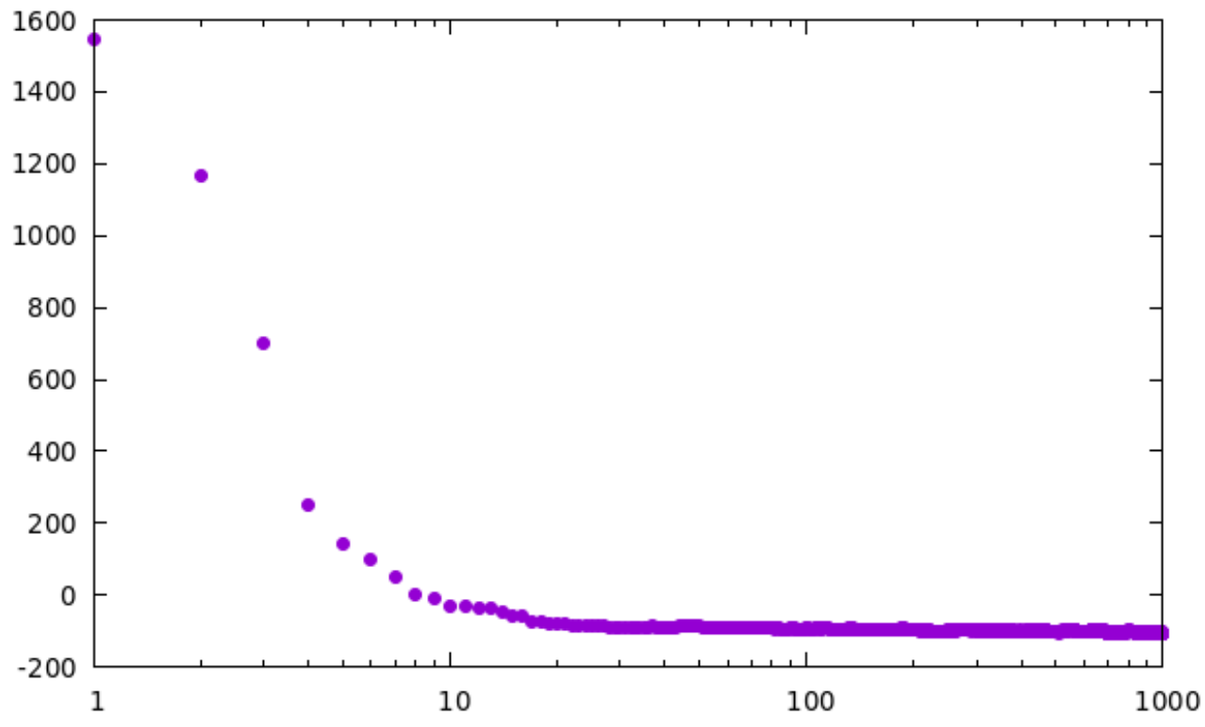
Data is generated in C++ with the Metropolis Algorithm

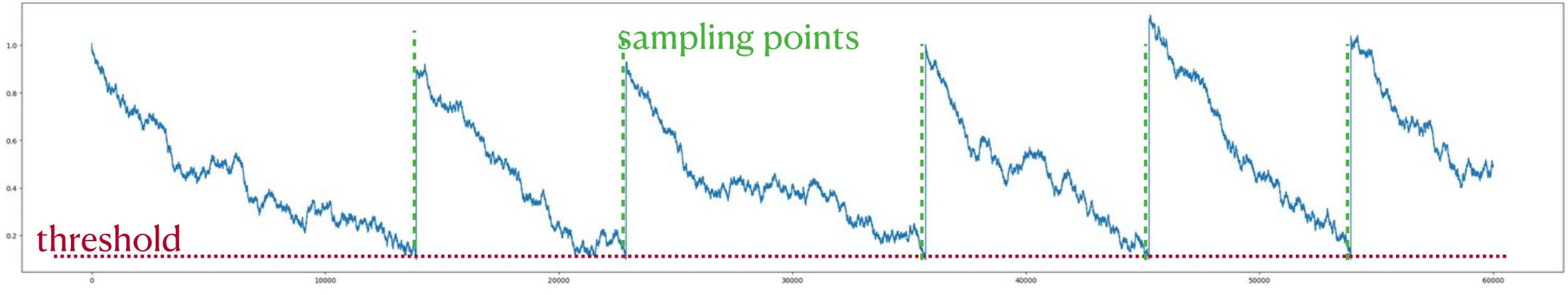
1. Start from a randomized configuration
2. Move one particle at random (kick)
3. if $\Delta U \geq 0$ the move is always accepted, else it is accepted with probability $p = \exp \frac{-\Delta U}{T}$
4. All moves that would end up outside the boundary are reflected back inside the box
5. Results are produced in cartesian coordinates

Generating the dataset

Results

After an initial transient phase the energy reaches, as expected, an equilibrium. From there we can start sampling data, after checking that the correlation between two consecutive state is below a set threshold (10% at the moment).





Generating the dataset

Optimization

In order to avoid huge outputs the mean time for decorrelation (MTFD) for a given sigma is computed in a small evolution then only data sampled at such intervals is saved for analysis

PRELIMINARY RUN

Compute all positions for a short time

Save all position

Compute correlation

Estimate MTFD

SAMPLING RUN

Compute for longer times

Save only positions for multiples

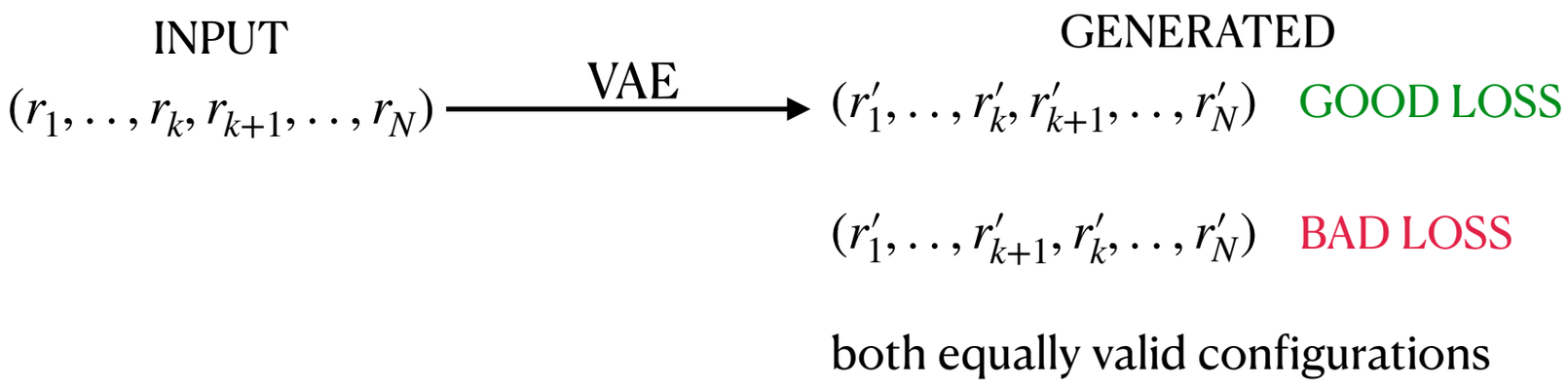
of MTFD

Variational Auto Encoder

General Framework

The main source of inspiration was the guide available in Keras' official website.

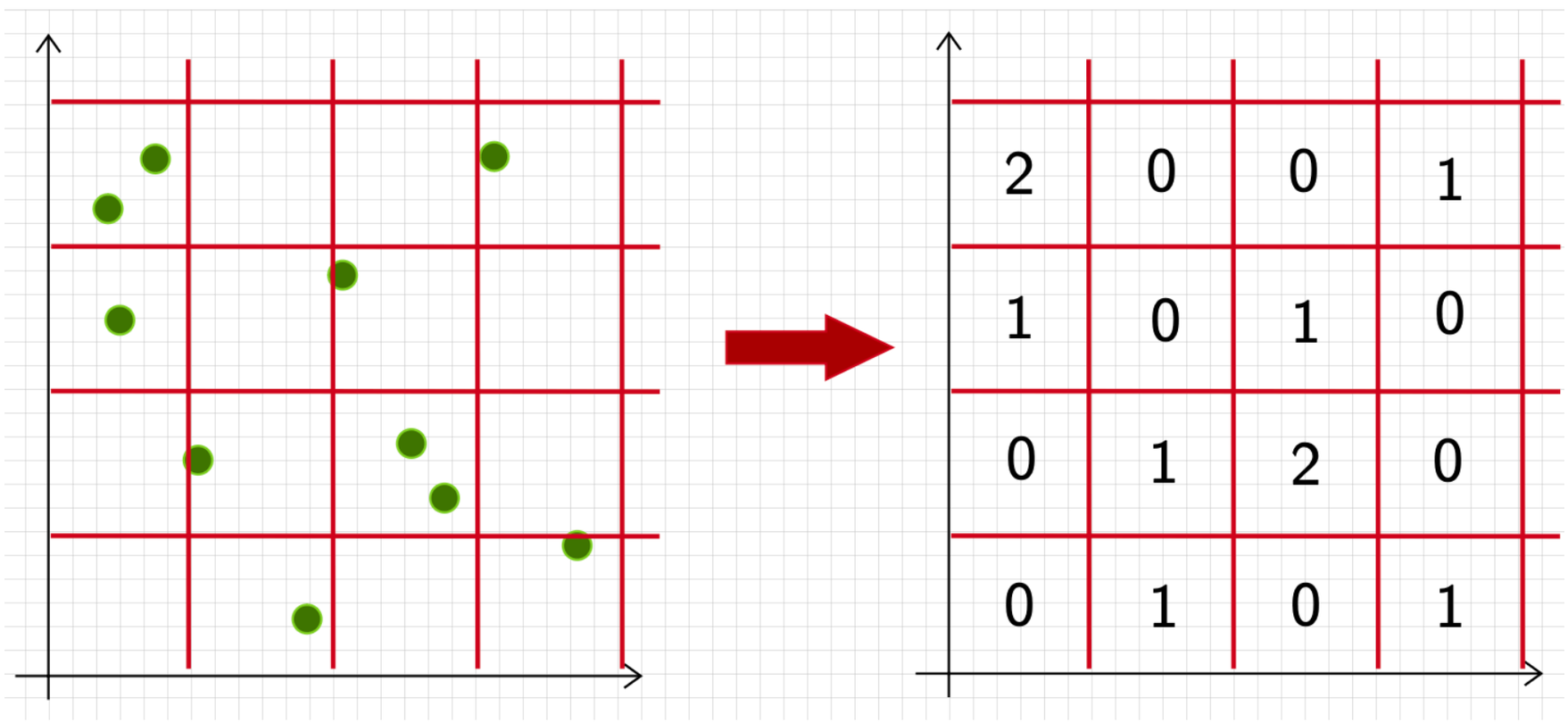
The first idea was to use a standard feed forward NN, this however turned out to be a bad choice since the learner would be sensible to the order of the tuples of the configuration



Variational Auto Encoder

General Framework

The next logical step is to find a way to decouple the positions of the particle from their order in the vector. This is achieved by a density measure in different regions of the box



Variational Auto Encoder

Input shape

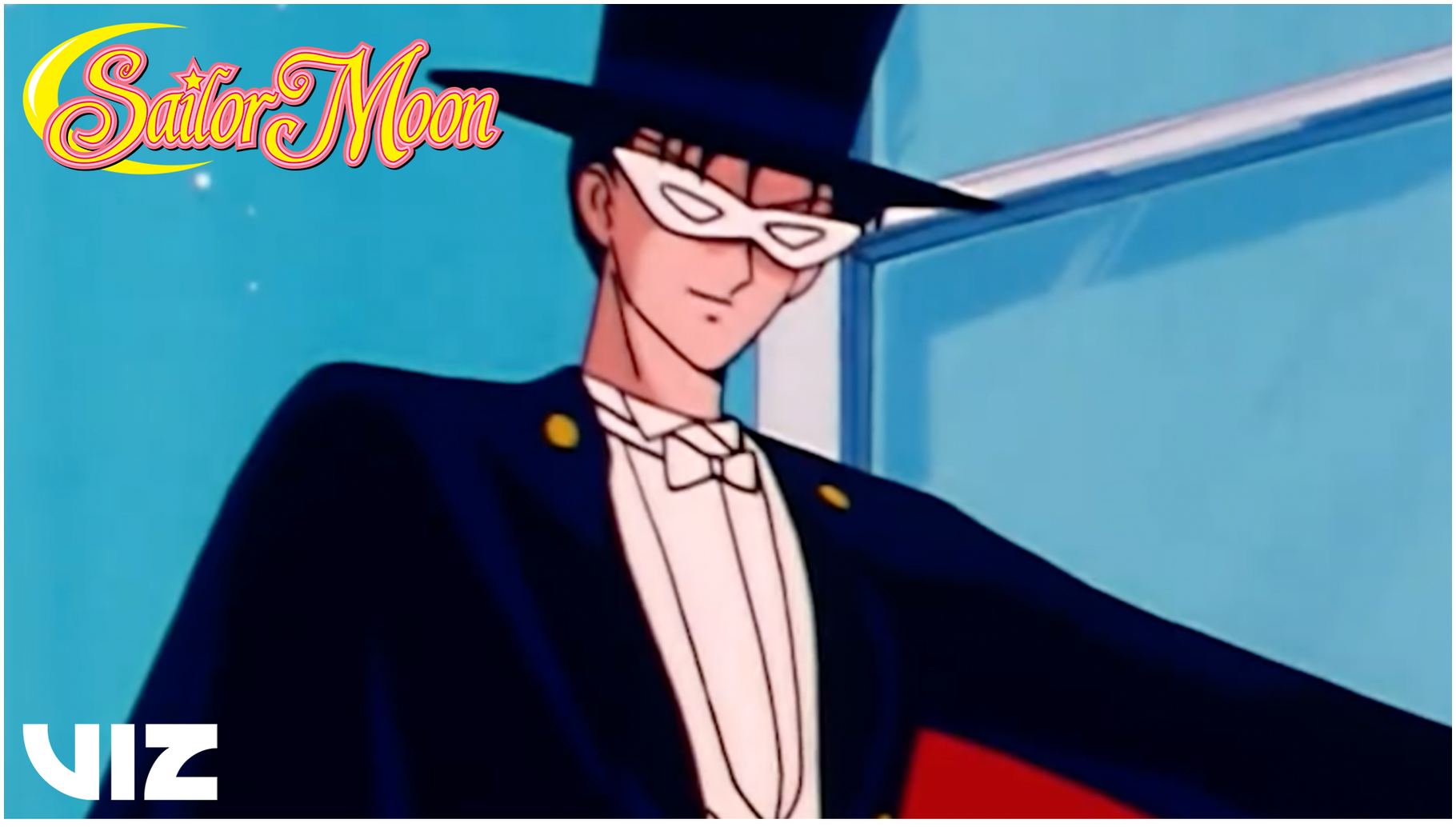
After some manipulation the input for the VAE will be a tensor of shape $(n, n, n, 1)$ where

A. $n = \frac{L}{h}$ with h size of the cell (possibly optimizable)

B. Number of points in each cell are normalized to (0,1) range (value $= \frac{\#N}{\#N_{max}}$)

Note that the size/complexity of the input does not increase with the number of simulated particles, promising for both flexibility and scalability

See you next week

- 

Variational Auto Encoders

Meeting 2 - 16/05/23

Group 9 - Bacilieri, Barbiero, Bordin, Pitteri

Data generation

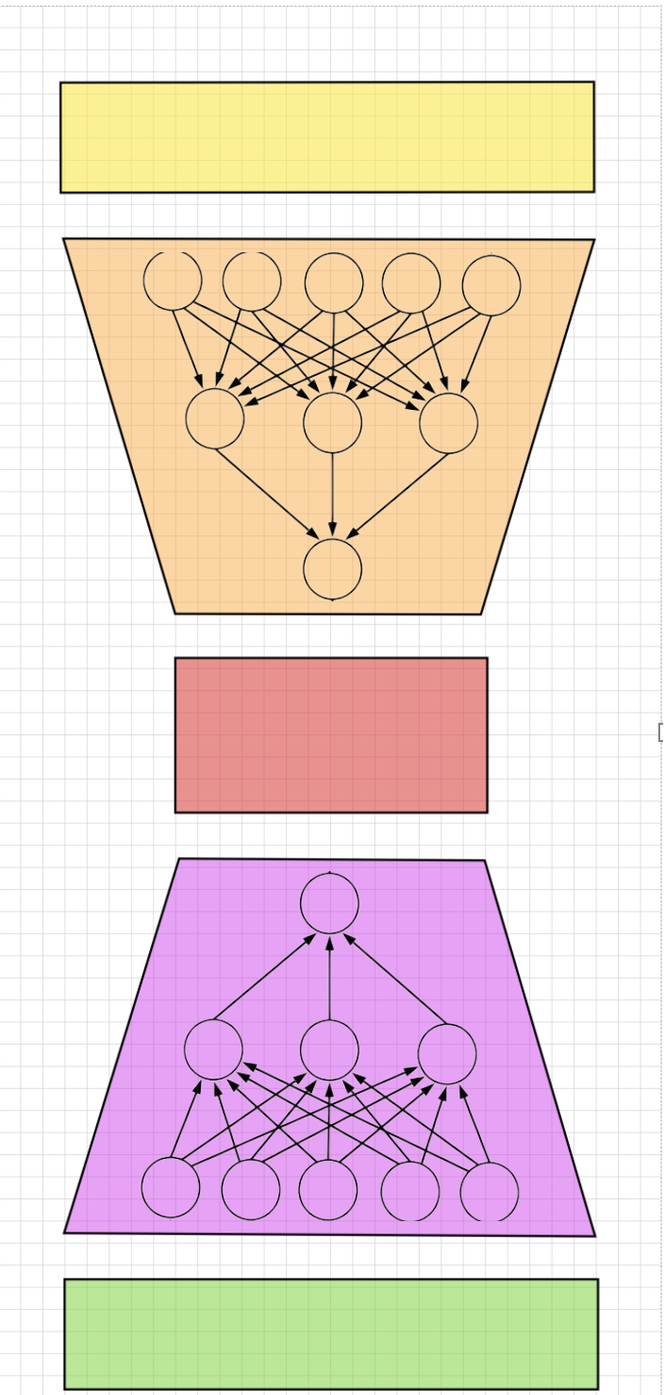
2D case

Fixed correlation by initializing the whole system at each sampling, that means

1. generate a random initial configuration
2. evolve it by a roughly estimated decorrelation time of 1000 steps
3. sample the data
4. return to step 1

Variational Auto Encoder

Feed forward NN implementation



1. Input shape: (N_{points}, dim) tuples, ordered by x value and then flattened
2. Encoder: dense architecture with three layers
3. Sampling layer: customizable latent dimension
4. Decoder: mirrored architecture w.r.t. encoder
5. Output shape: same as input shape

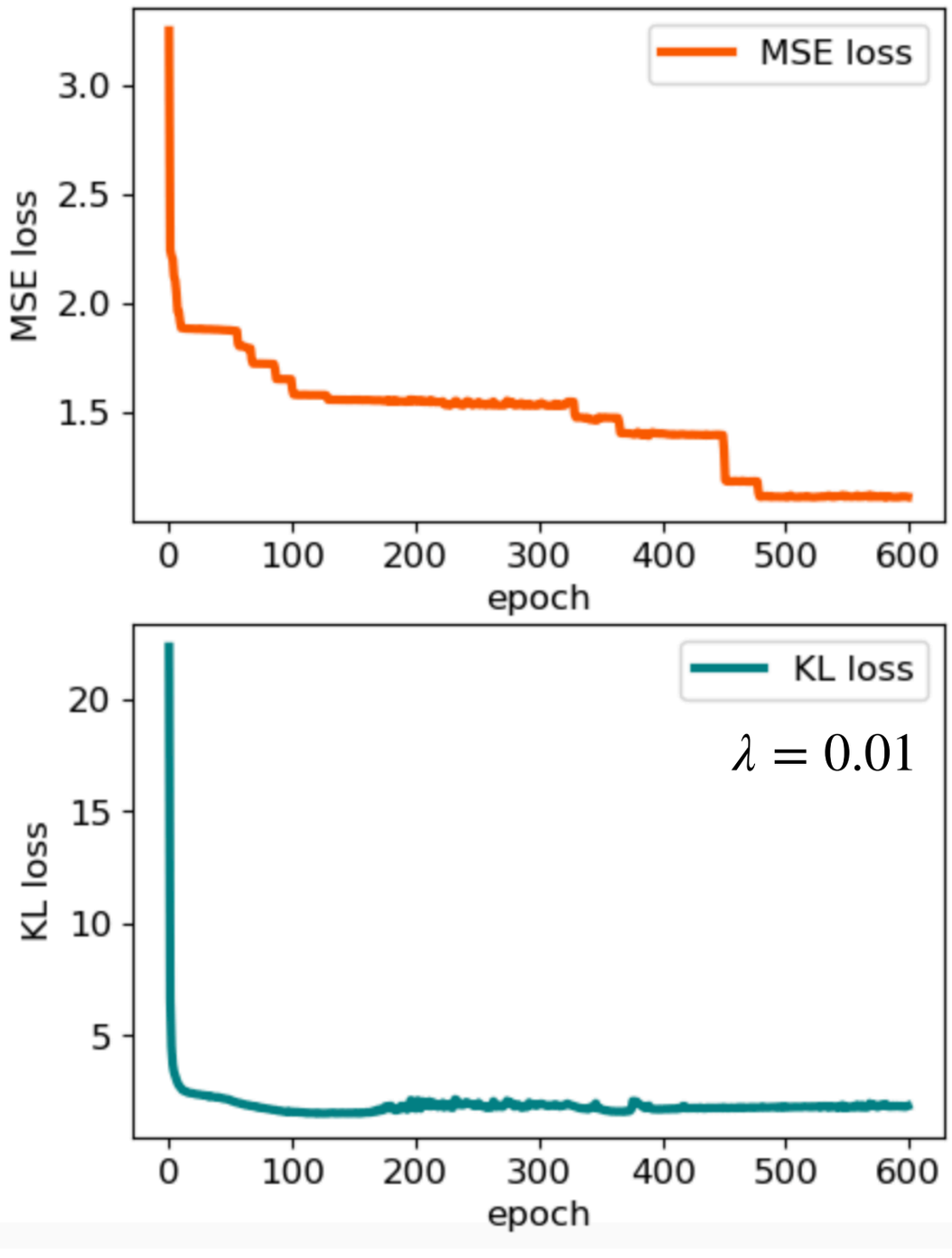
Variational Auto Encoder

Feed forward Results

1. 30 points in a 3D box of size 5σ
2. 8000 valid configurations used for training
3. 60 —> 45 —> 22 nodes layers
4. latent dimension = 2

Variational Auto Encoder

Feed forward Results



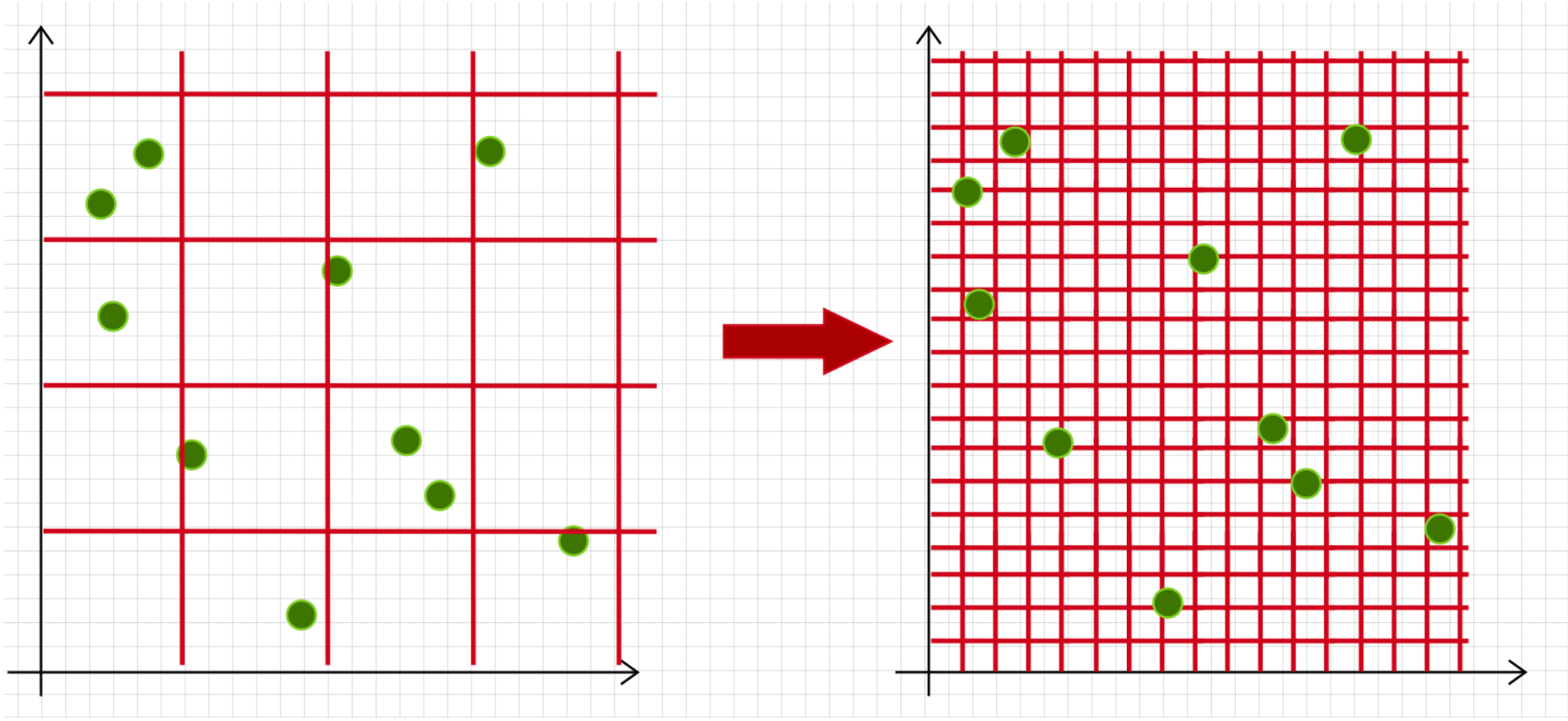
The network seems do to a good job at minimizing both MSE and KL loss, training time is quick at 70-120 ms/epoch depending on the machine.

Next up will be data prediction and latent space analysis

Variational Auto Encoder

Convolutional 3-D Implementation

Reduced grid size (pixels) of size 0.1σ should avoid having more than one particle per pixel and allow for sufficient accuracy in the computation of the potential for anomaly detection



Variational Auto Encoder

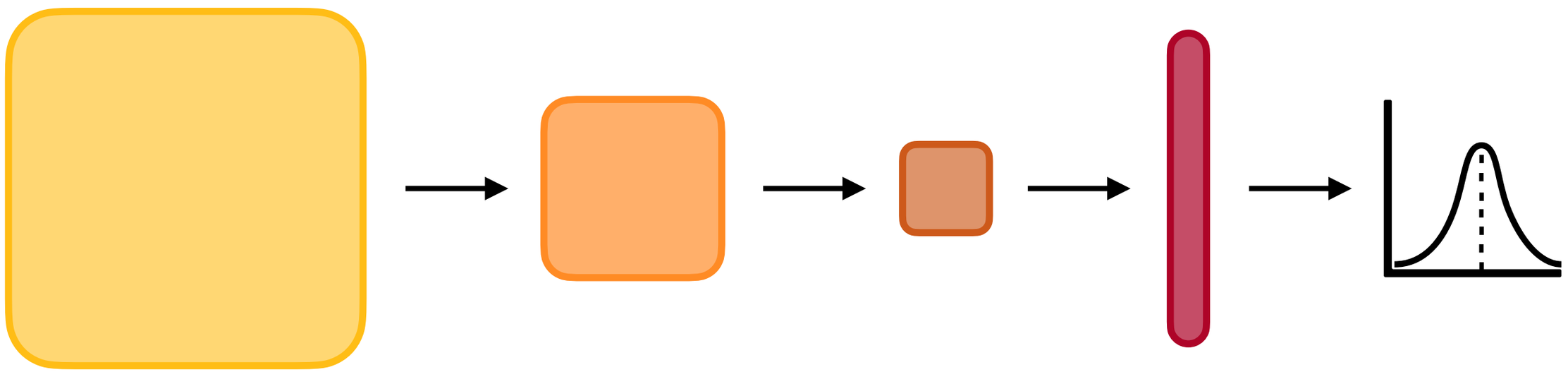
Convolutional 3-D Implementation

Encoder is composed of three layers, each with

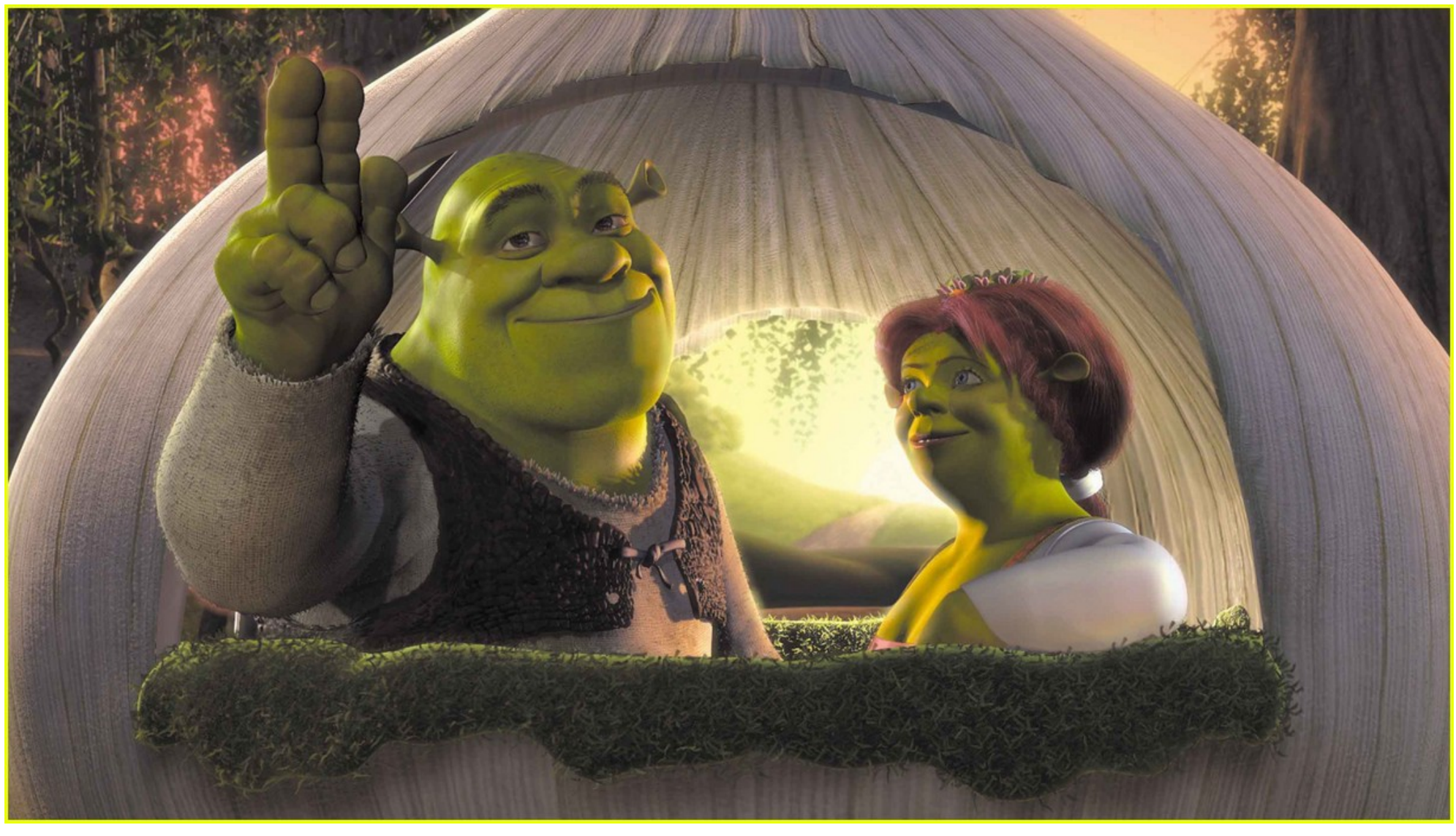
1 Convolutional layer

1 Pooling (2,2,2) layer —> Halves the size in each direction

The output of the encoder is then flattened for learning mean and log_var of the latent space variables



See you next week



Variational Auto Encoders

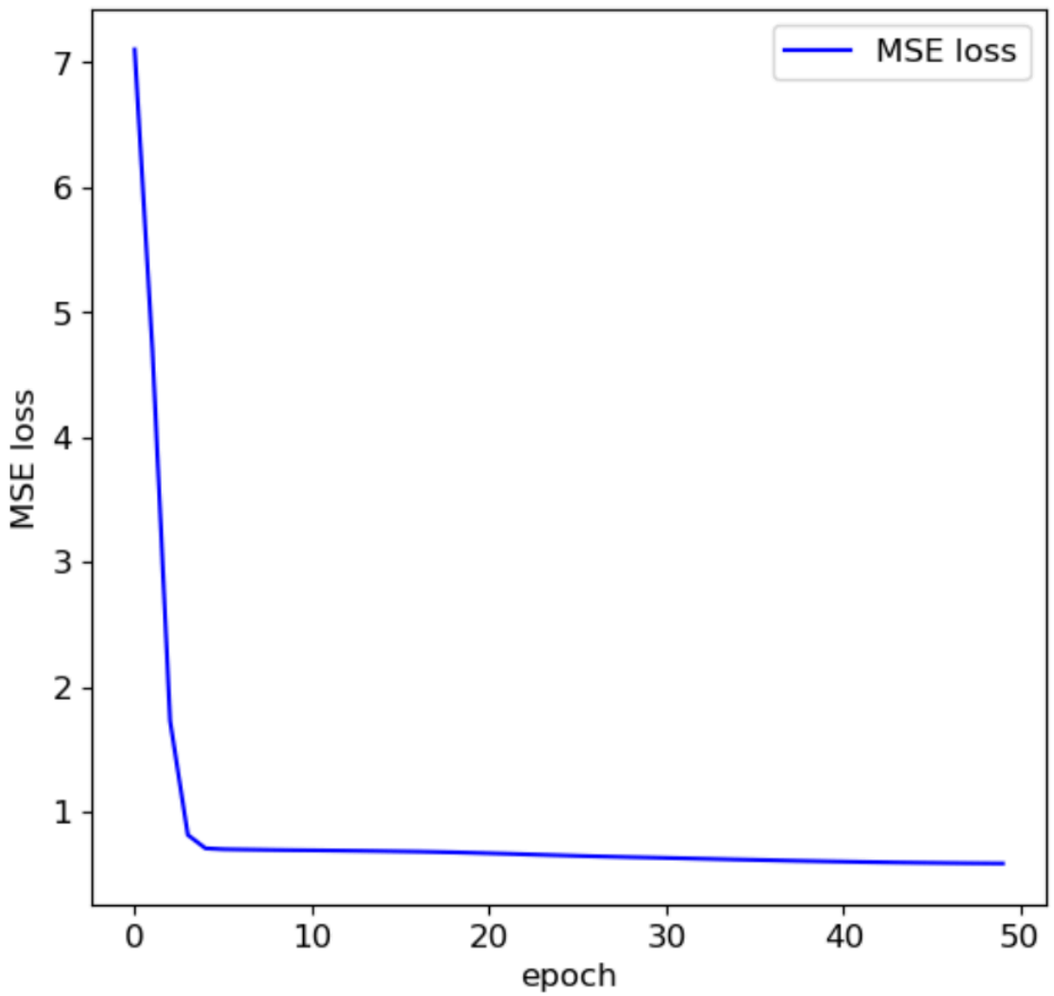
Meeting 3 - 23/05/23

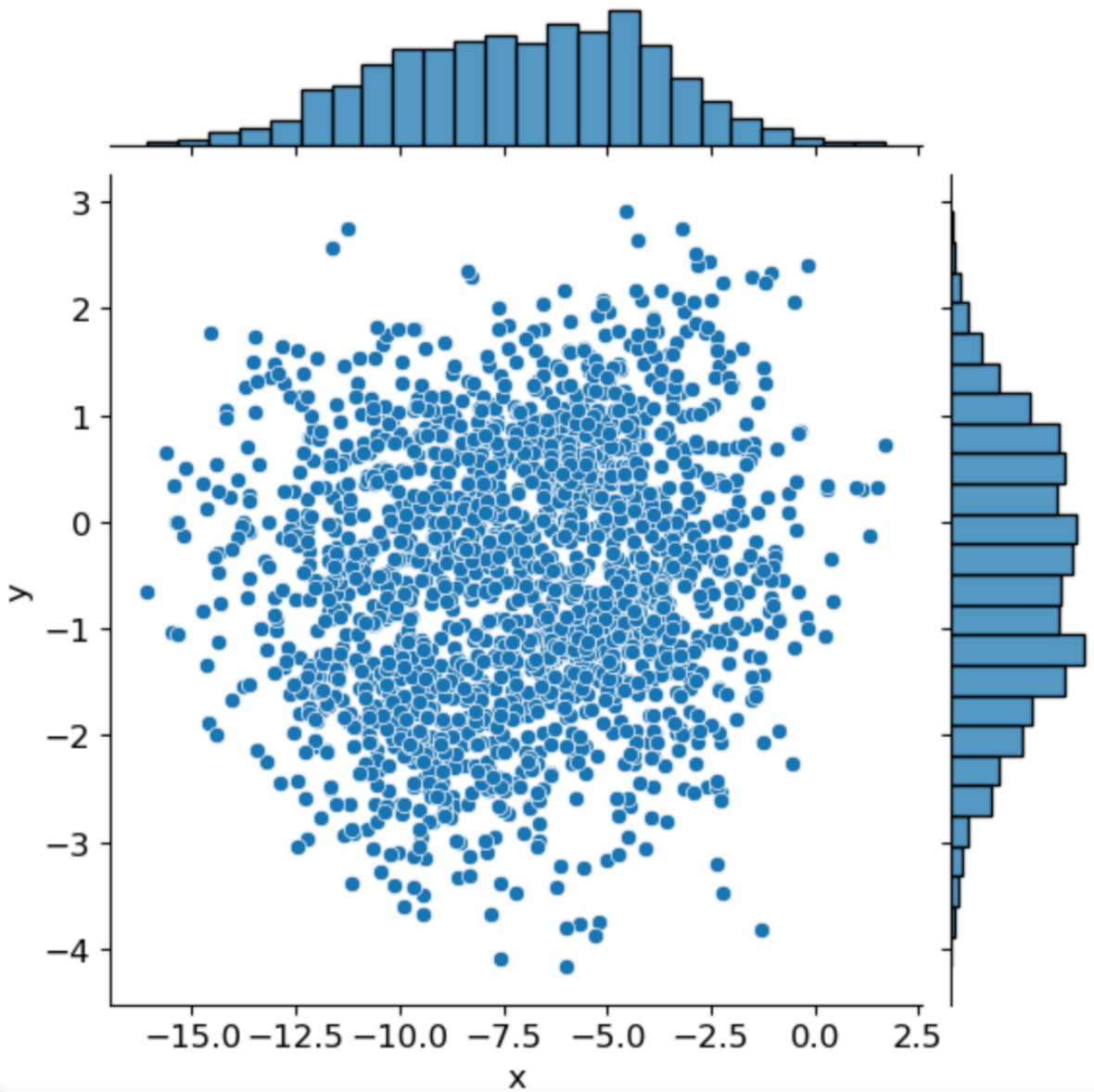
Group 9 - Bacilieri, Barbiero, Bordin, Pitteri

Training

Learning from configurations

Training the Neural Network on only the configurations was good

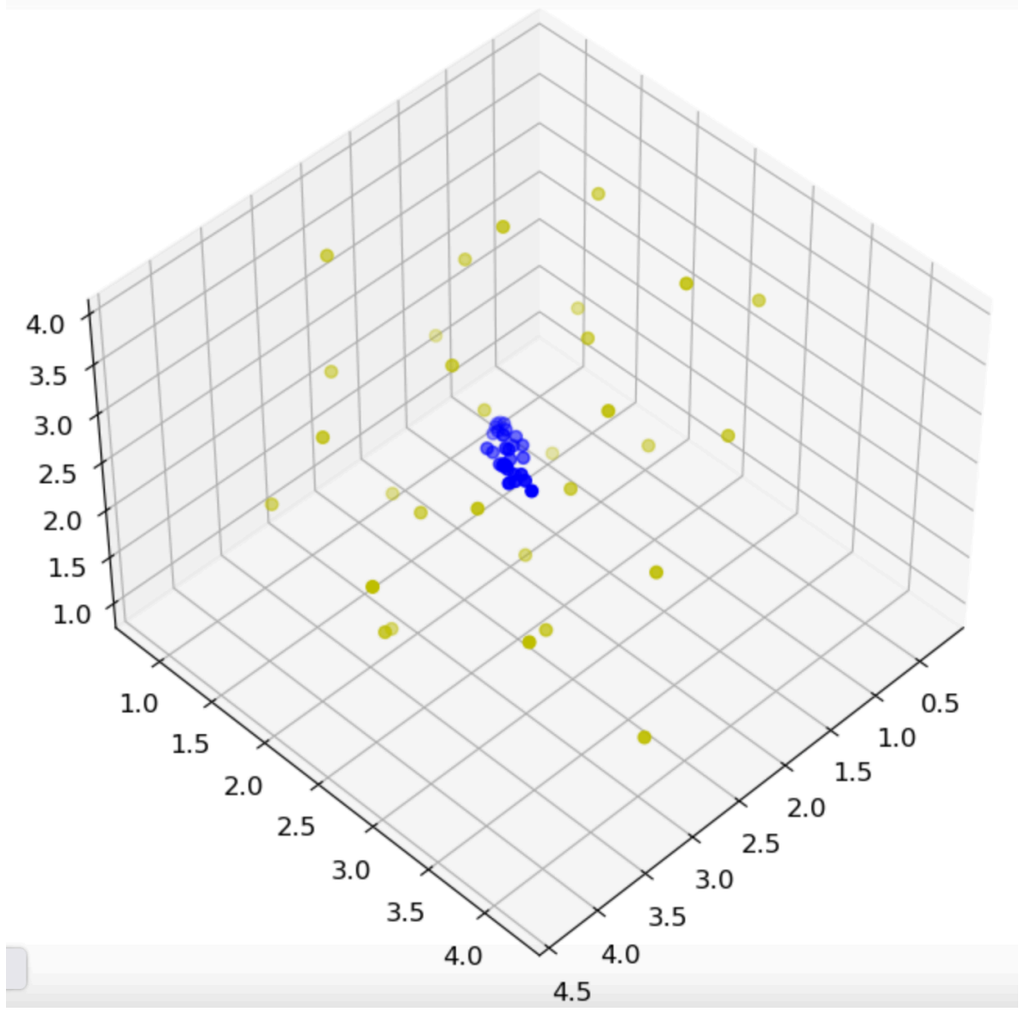




Training

Learning from configurations

Until it wasn’t anymore

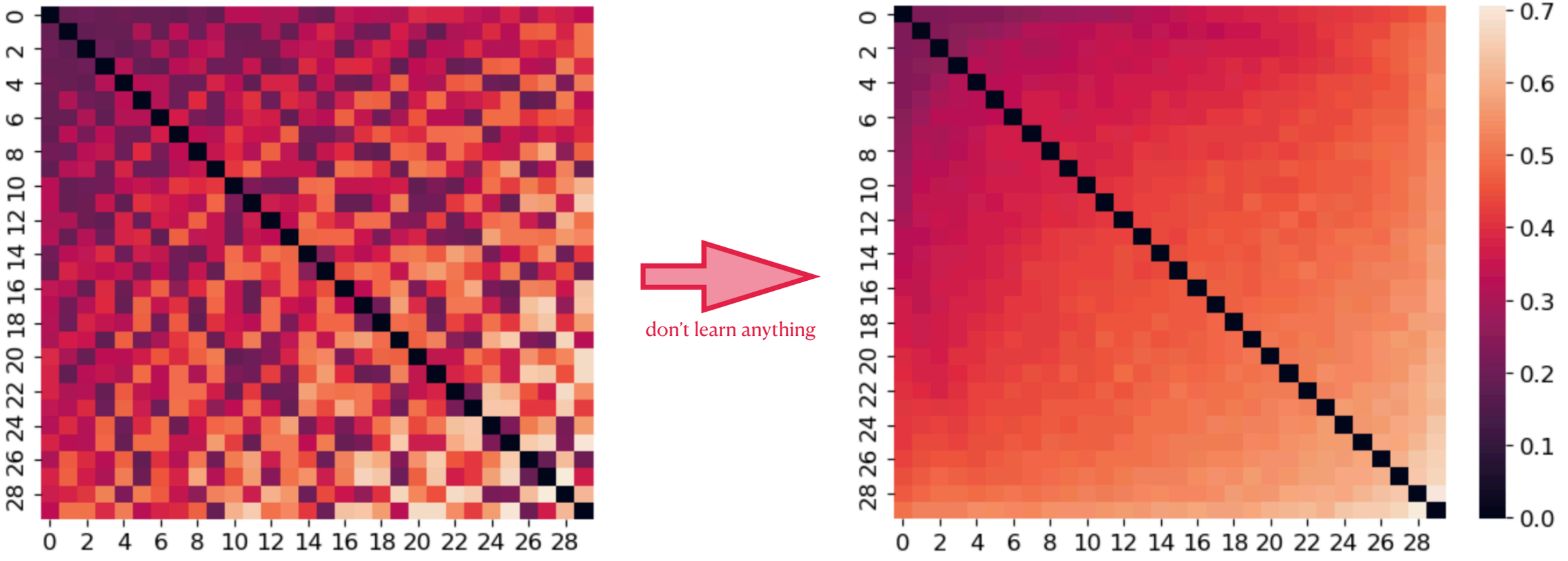


Network seems to be putting all the particles in the middle of the box, meaning it didn’t learn anything.
We decided to proceed with a change to internal coordinates plus a normalization (all data in -1,1)
Results weren’t better

Training

Learning from the Distance Matrix

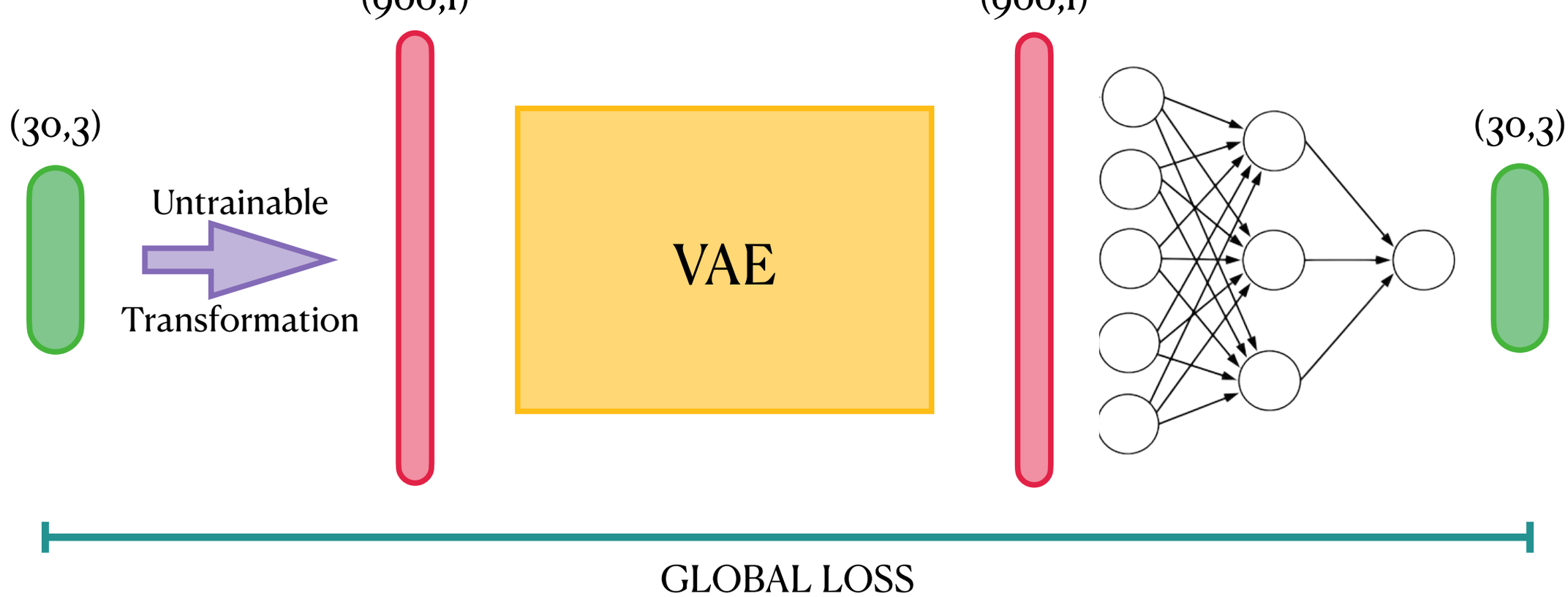
The next step was to implement a network on the distance matrices, it went from a (90,1) input shape to a (900,1) input shape but still it wasn't learning anything



Reconstruction

Putting everything together

On top of this, the challenge now is to reconstruct valid spatial configurations from the distance matrix



See you next week

Variational Auto Encoders

Meeting 4 - 30/05/23

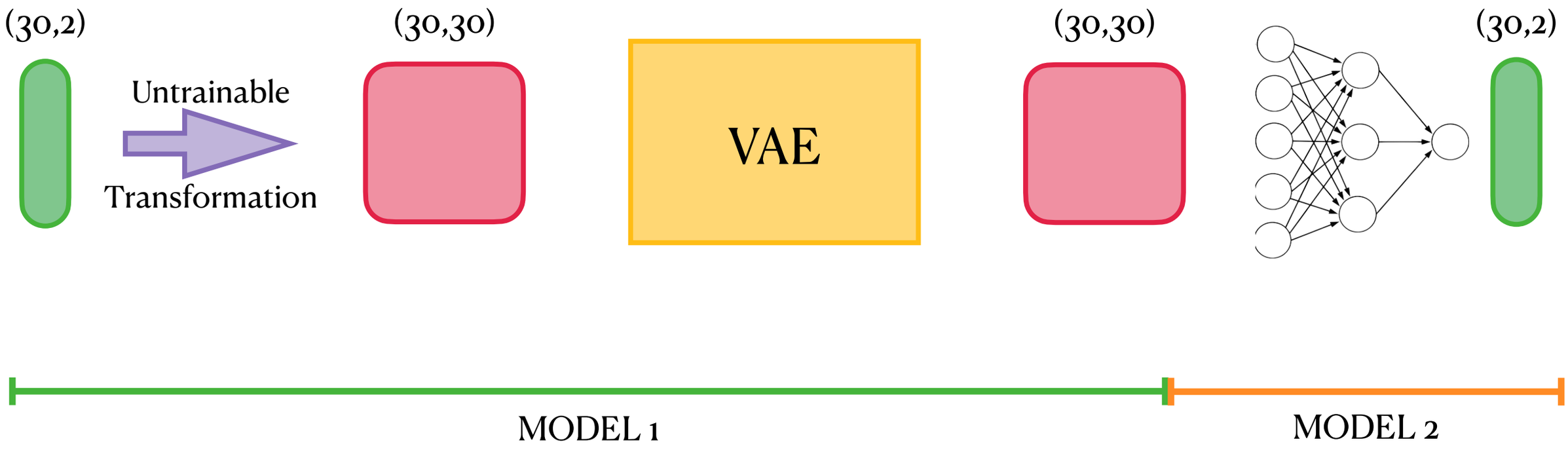
Group 9 - Bacilieri, Barbiero, Bordin, Pitteri

Going convolutional

Going convolutional

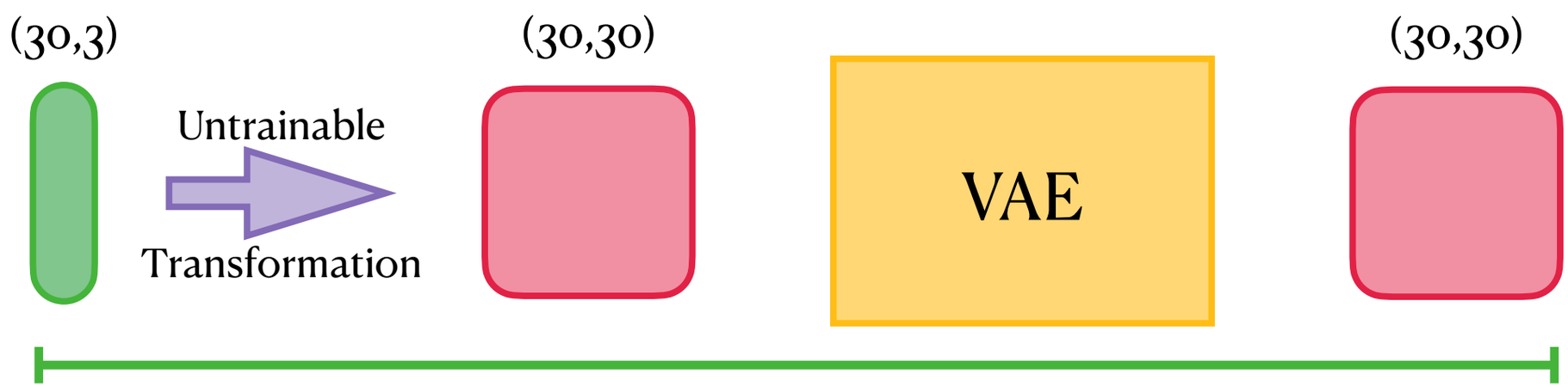
New architecture

The VAE is redesigned to be convolutional on the distance matrices



Going convolutional

Distance matrix



MODEL 1

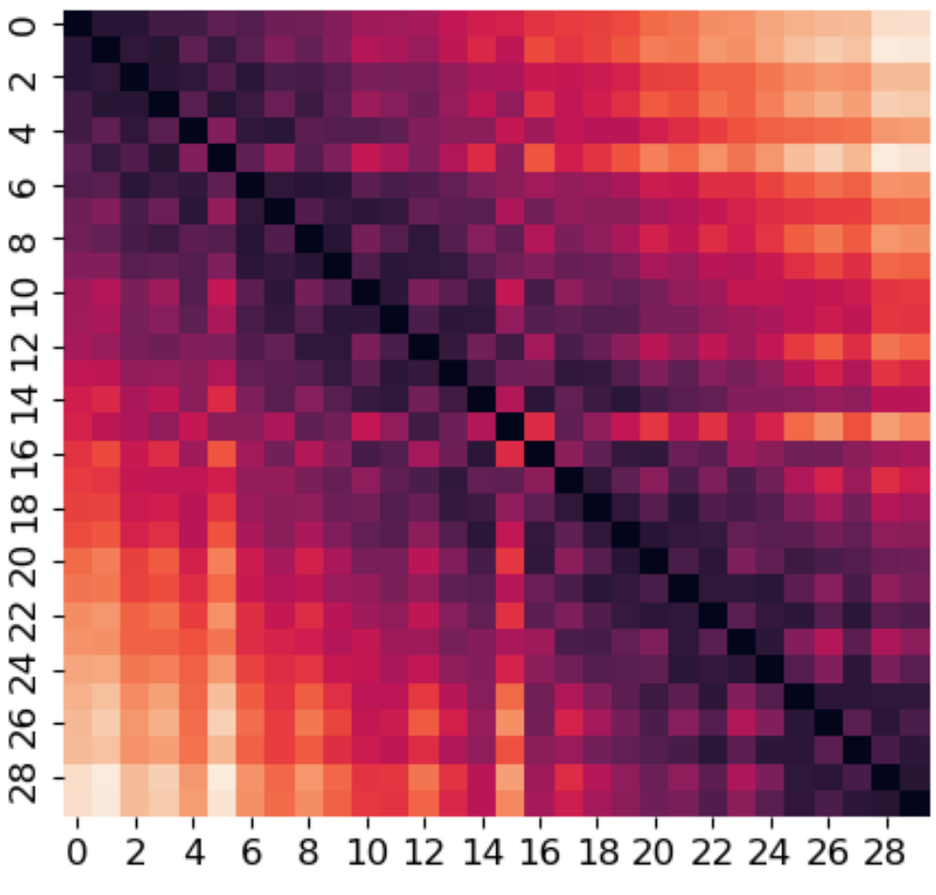
The first model is comprised of a Conv2D decoder and decoder which operates on the matrix distance obtained after ordering points by their distance from the origin and normalized to make sure that all values for the distances are between (0,1).

Depending on the architecture training can get a bit cumbersome but still reasonable even for home PCs

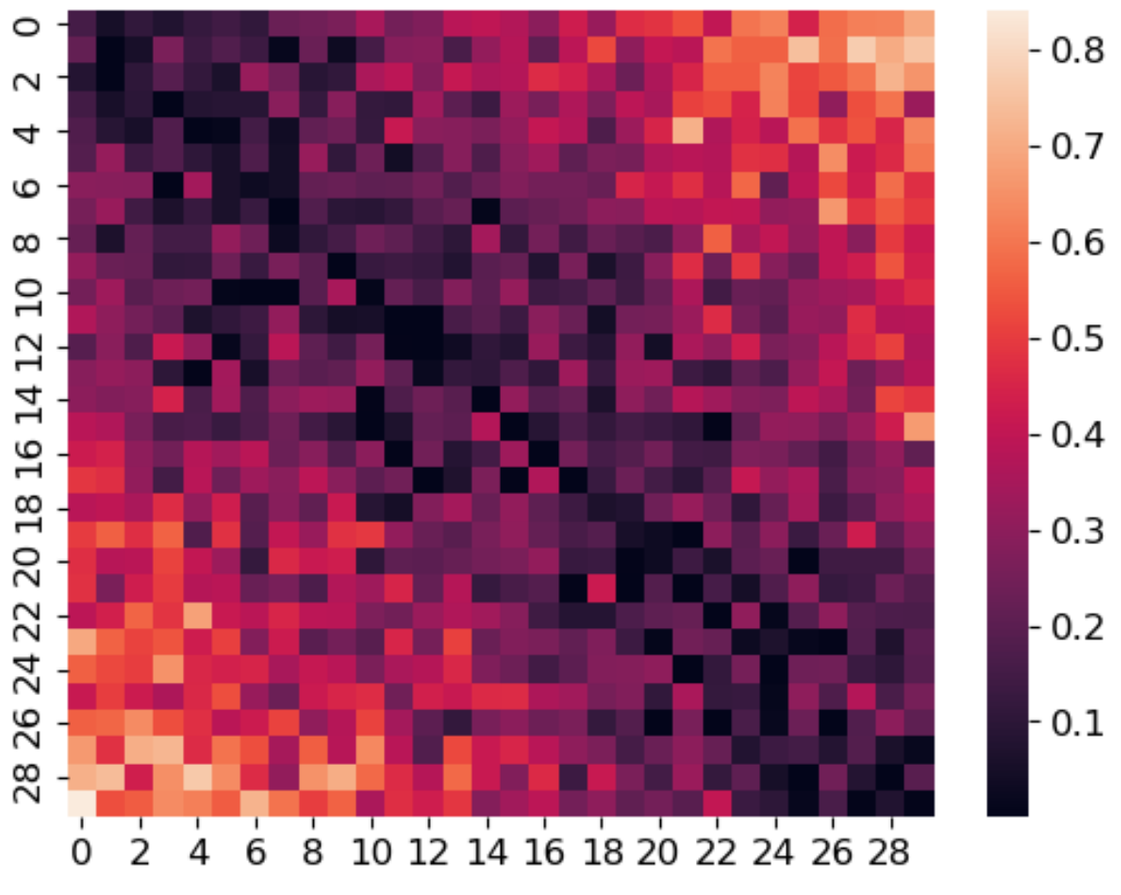
Training the model

First results

Reconstruction is not perfect but starts making sense, further testing has to be done with a more complex/optimized network



Original

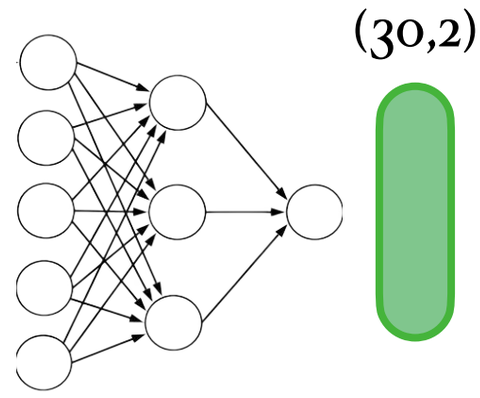


Reconstructed

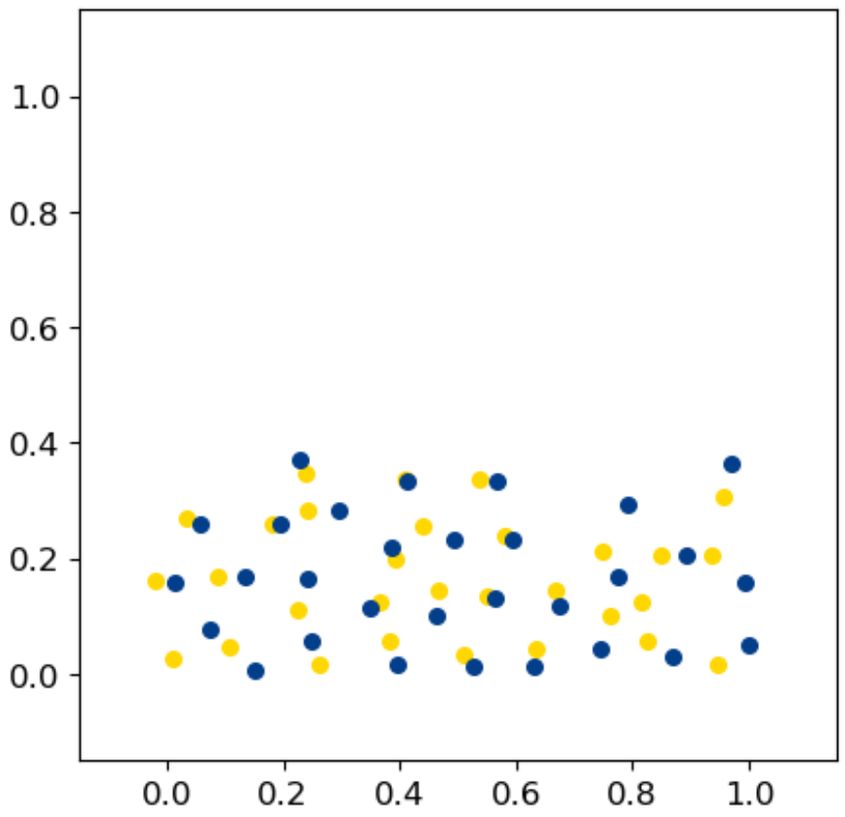
Training the model

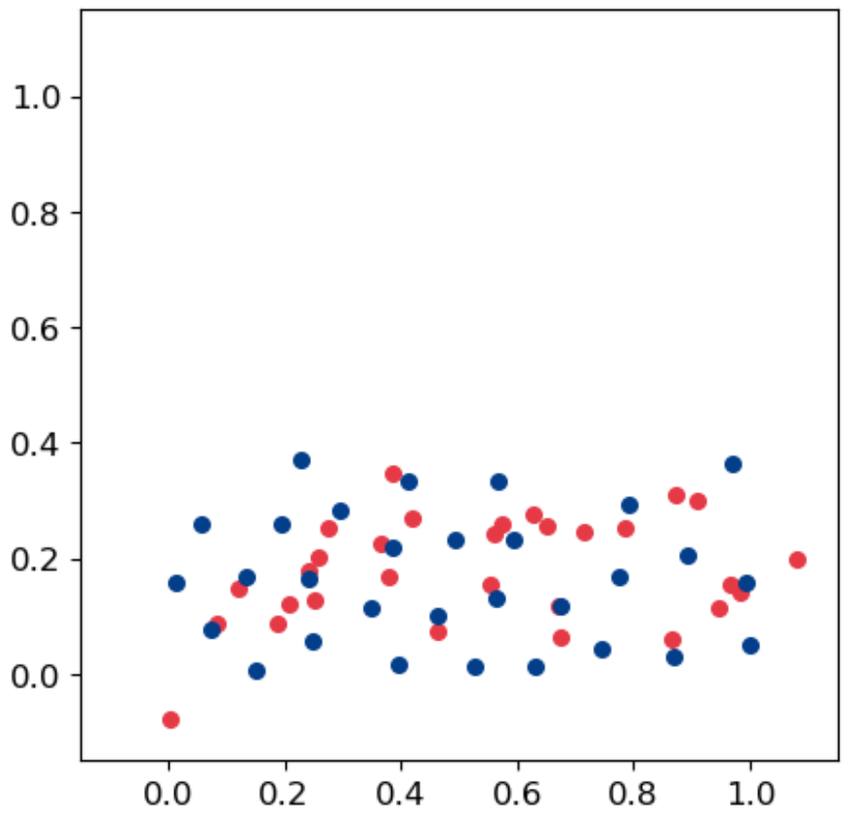
Reconstructing configurations

For the second model a simple feed forward NN is implemented, trained on the original distance matrices and then used in the reconstruction of the previously decoded ones



MODEL 2





Training the model

What's next?

1. Optimize the CNN architecture for better distance matrix learning
2. If 1. does not fix the issues enhance Model 2 for reconstruction
3. Once 1. and 2. are satisfied we can move on with anomaly detection and lambda based clustering in latent space

See you next week

Variational Auto Encoders

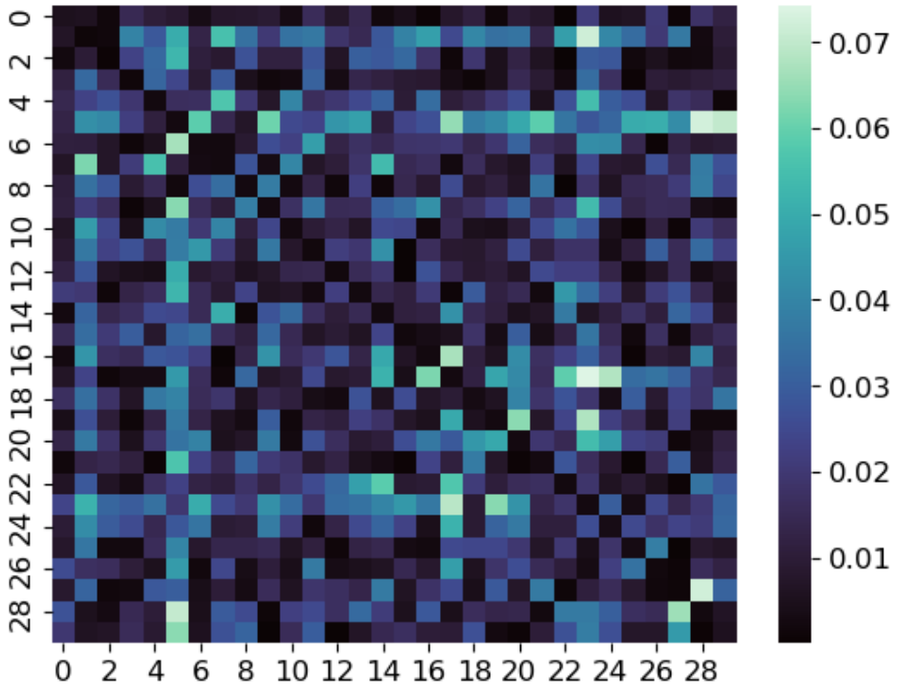
Meeting 5 - 06/06/23

Group 9 - Bacilieri, Barbiero, Bordin, Pitteri

Matrix training

Fine tuning

With a latent dimensions of (50-100) results are satisfactory, the rest of the architecture is unchanged



Reconstruction training

Another way

Another possibility we explored for coordinate reconstruction from distance matrix training is to use a “deterministic” which does not involve machine learning, to be more specific

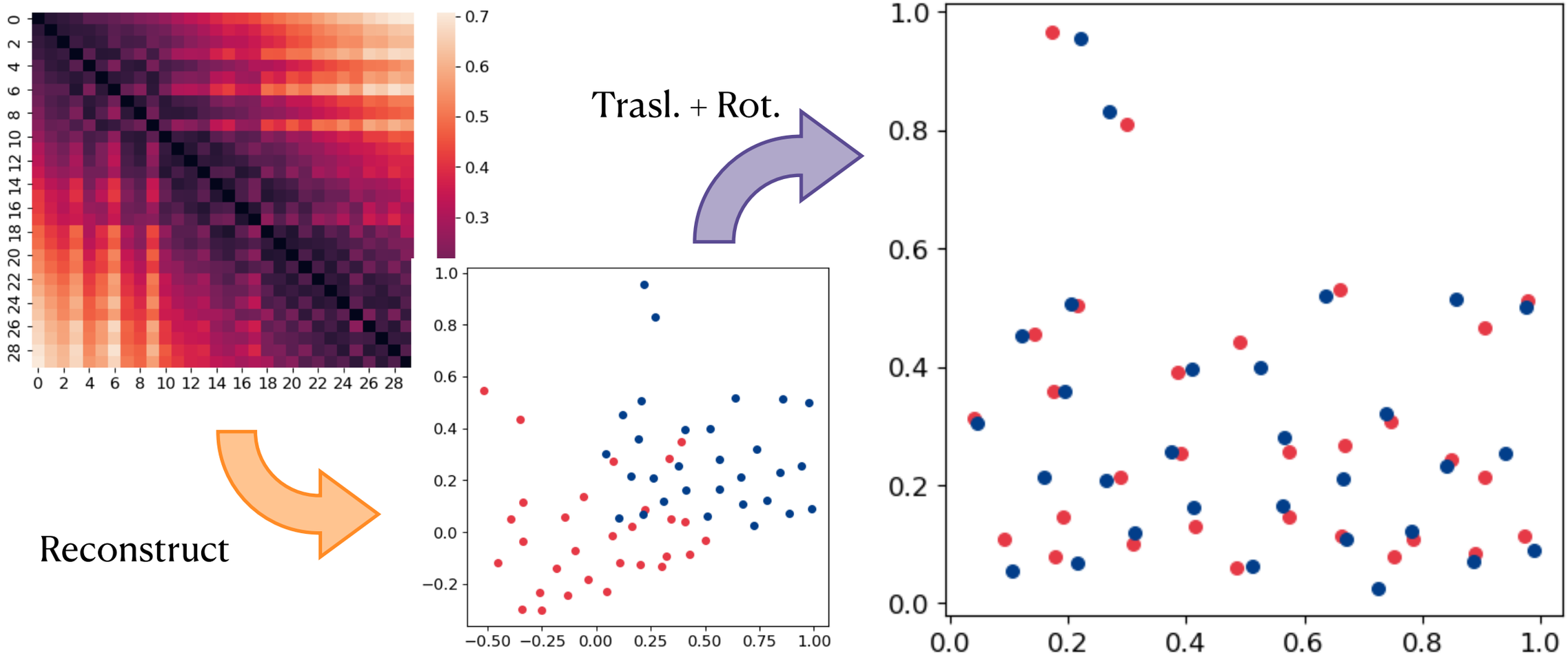
1- from the DM extract a valid configurations of points using the Gram matrices method

2- compute the translation vector and rotation angle(s) to minimize MSE w.r.t. the corresponding original configuration

Reconstruction training

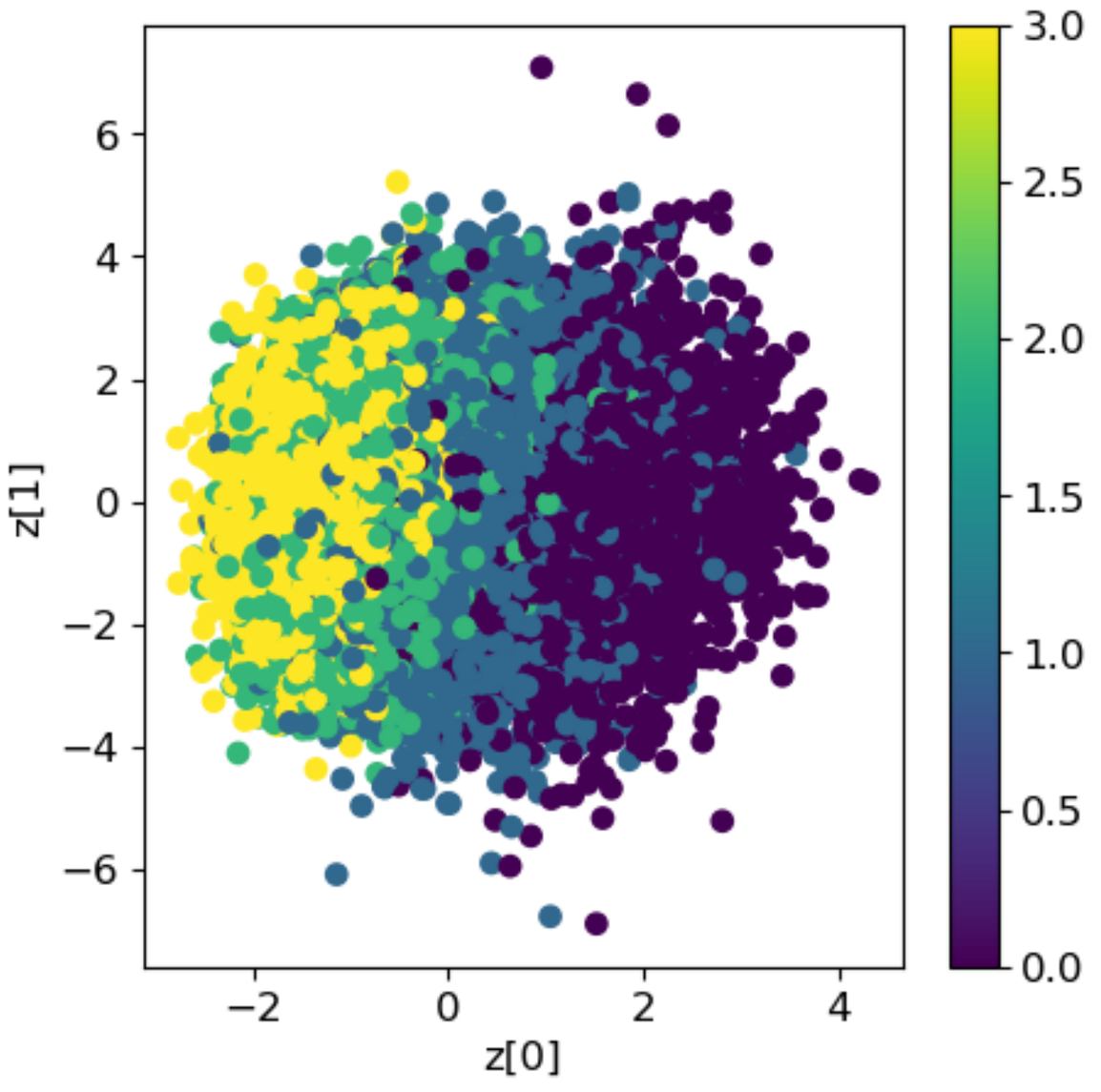
Another way

Reconstruction is (obviously) not perfect but the tendency to “bunch everything together” is diminished



Lambda clustering

Training with multiple lambdas enhances performance for the network, plus the observed clustering in a 2D PCA representation is reasonable

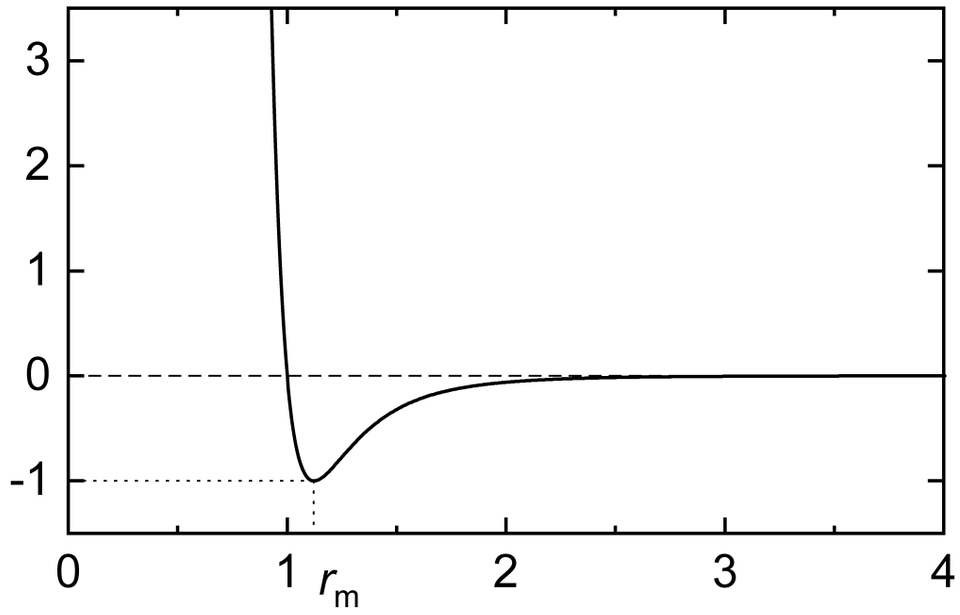


Energy Mapping

Still some issues

LJ potential is incredibly “picky” about the configurations it considers valid so reliably producing energy plots is still a challenge.

Even two points slightly too close to each other make the energy skyrocket



What's Next

Longer and semi definitive training with more data, more complex architecture and more epochs, trained model is then saved for future usage.

In parallel a full model with joint loss minimization is being developed

See you next week

...at the exam