Sapienza
Università di Roma

EMANUELE RUCCI

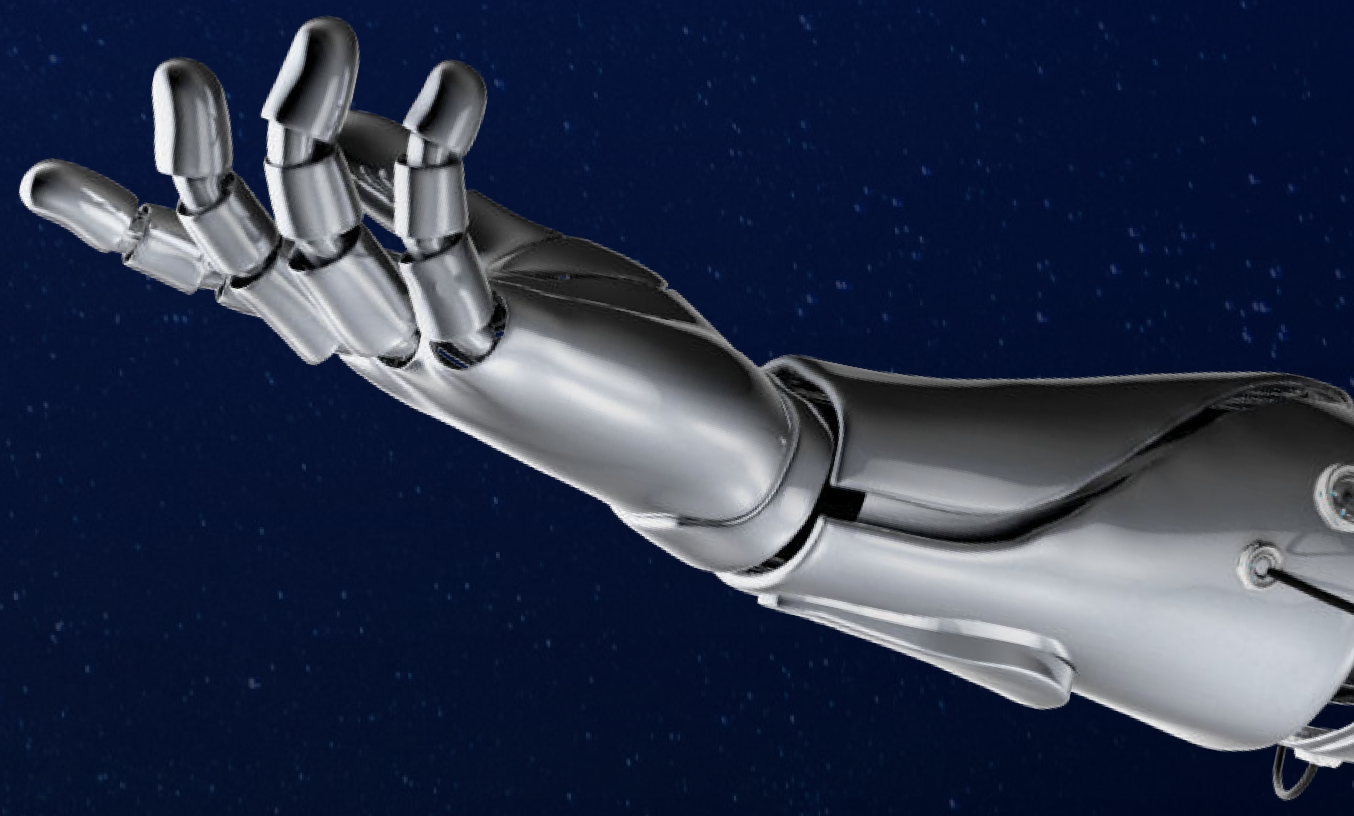
Sparse Autoencoder and efficiency constraints

INDEX

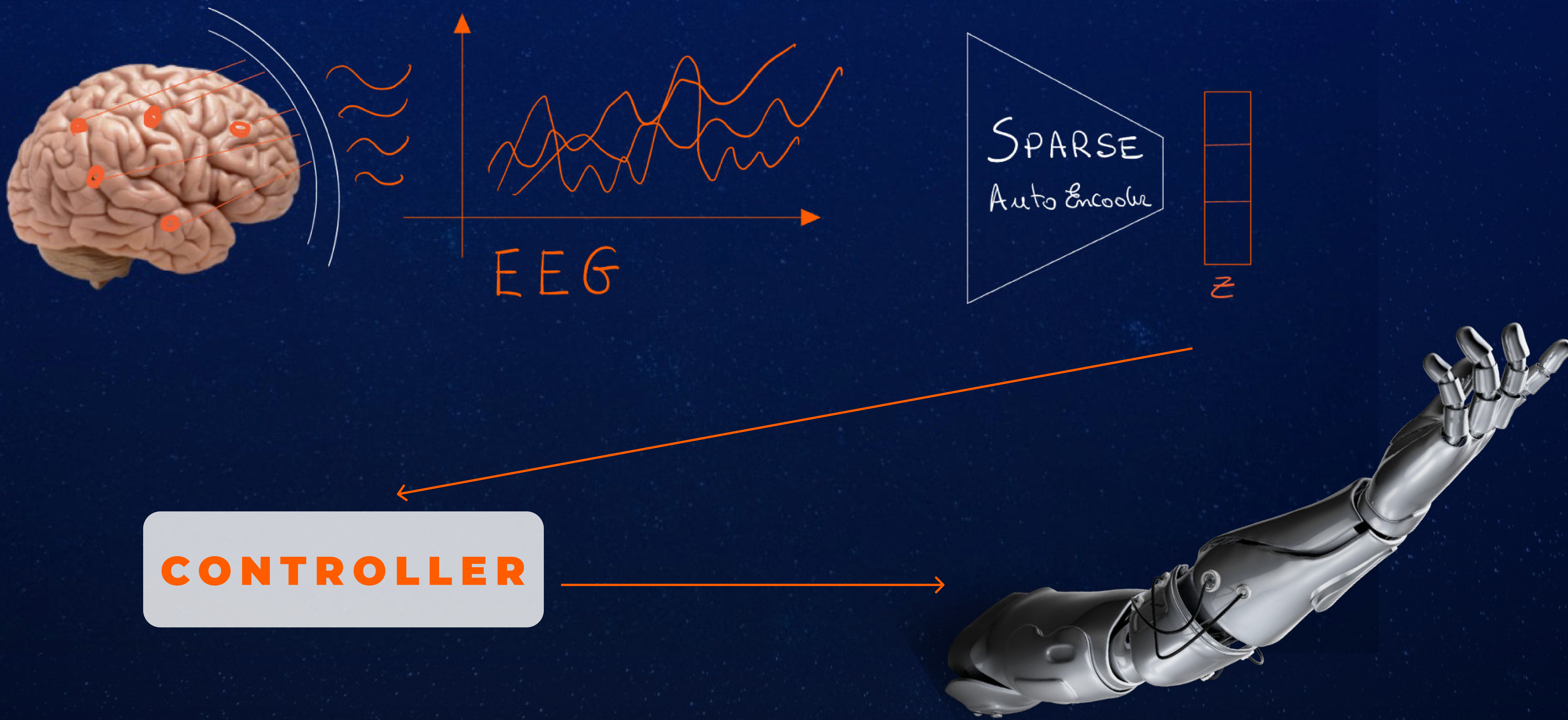
SAPIENZA
UNIVERSITÀ DI ROMA

TASK & PROBLEM

From brain to robot movement



TASK & PROBLEM



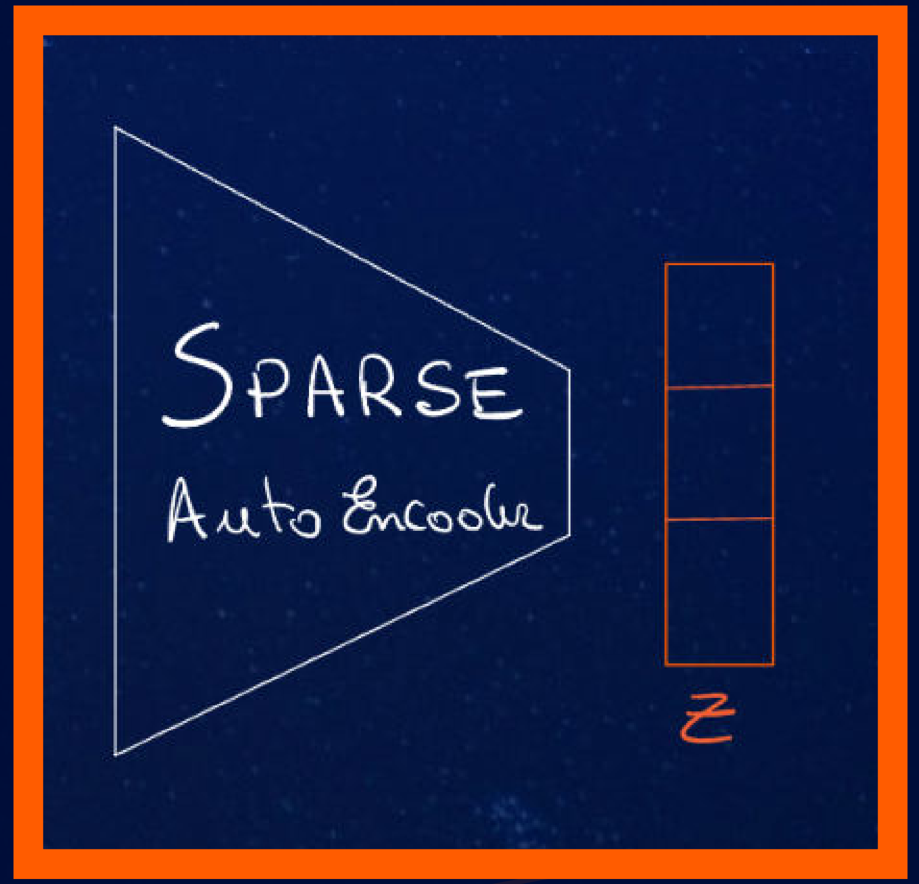
SAPIENZA
UNIVERSITÀ DI ROMA

TASK & PROBLEM

ISSUES

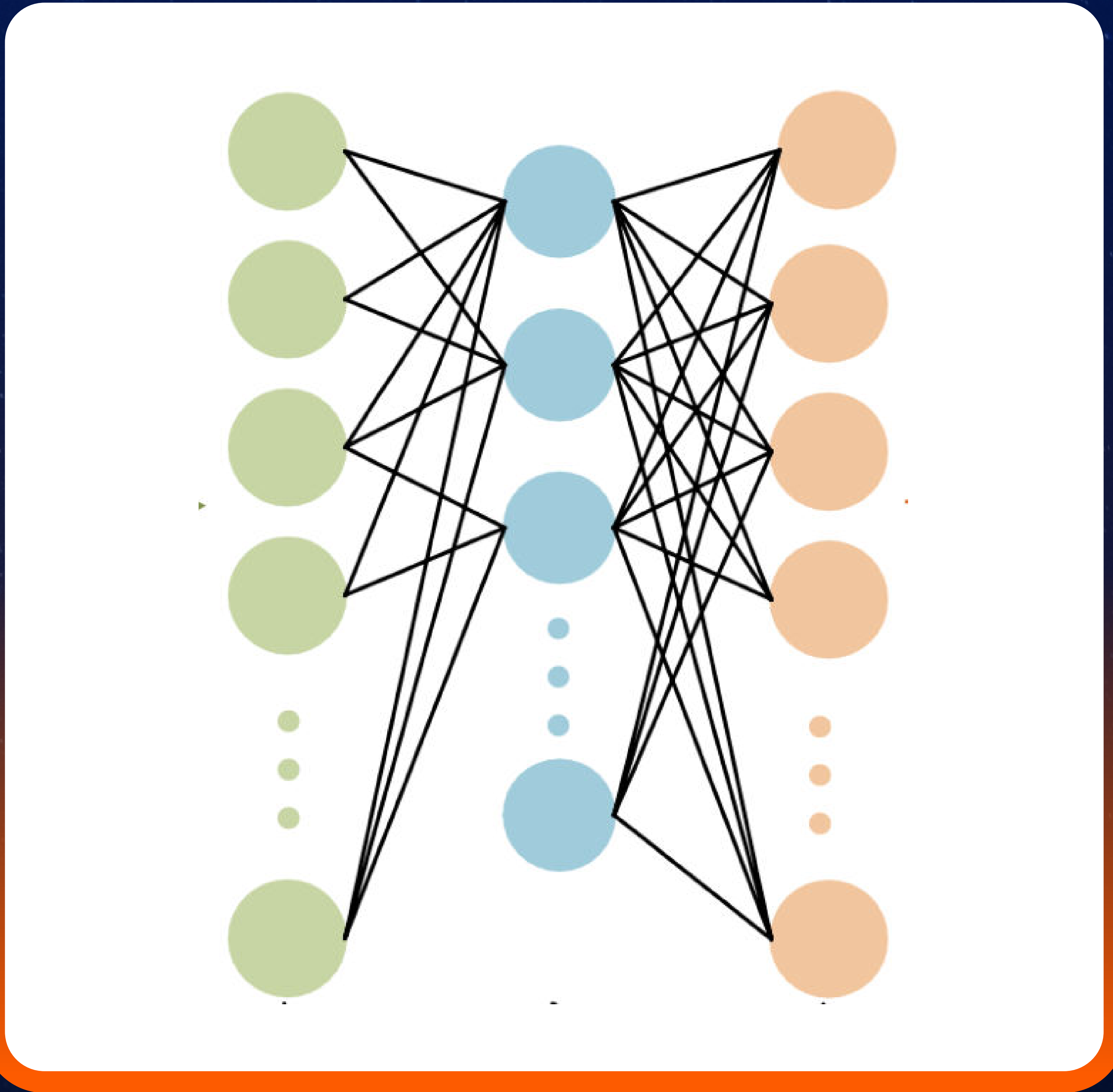
Slow feature extraction

Not suitable for realtime

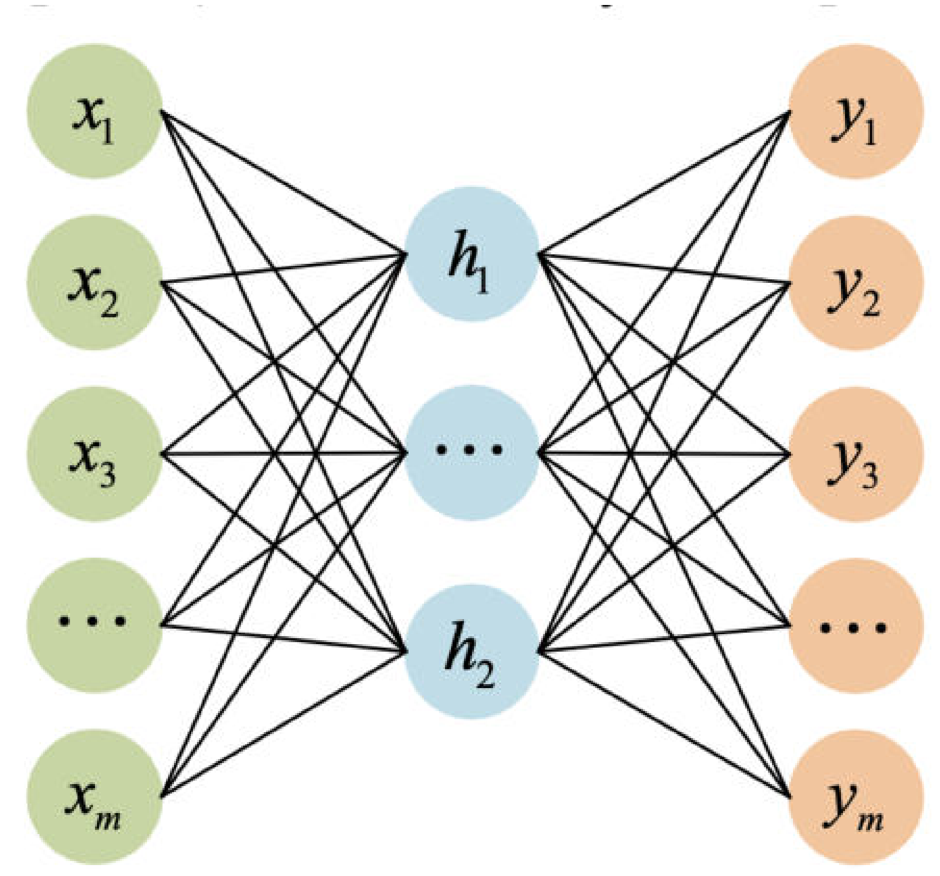


Sparse Autoencoder

Sparsity Constraint



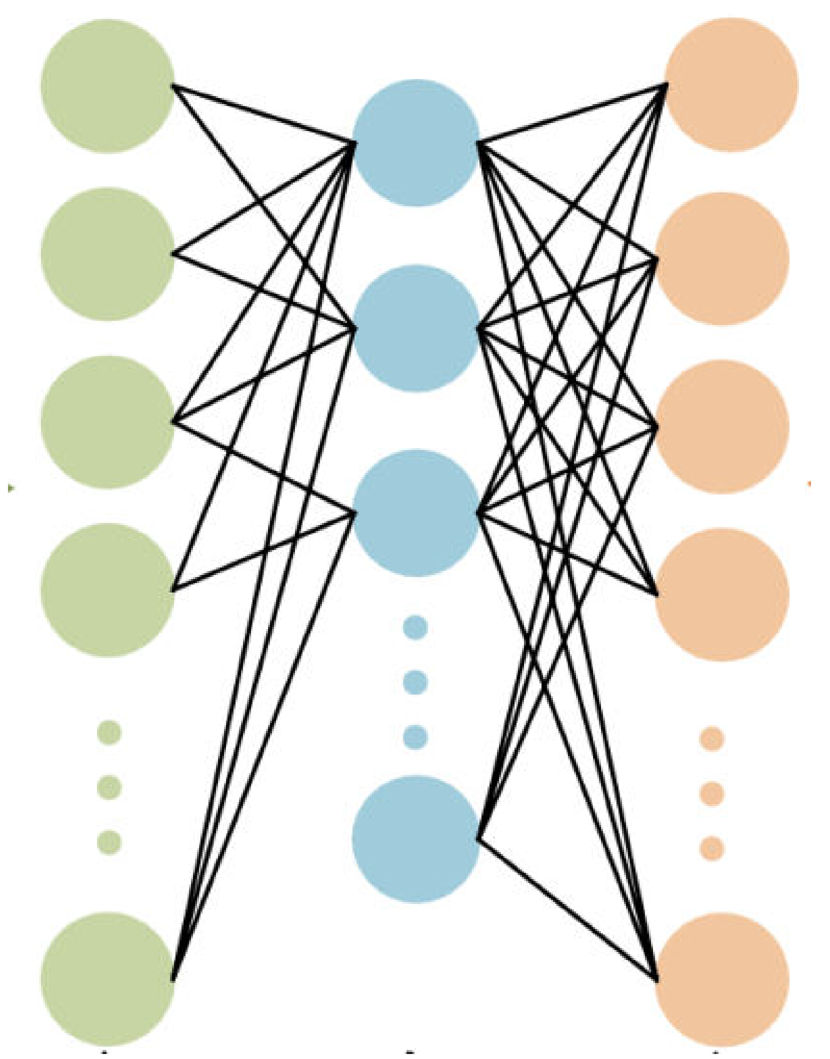
AUTOENCODER



MSE LOSS

Works as **reconstruction loss**

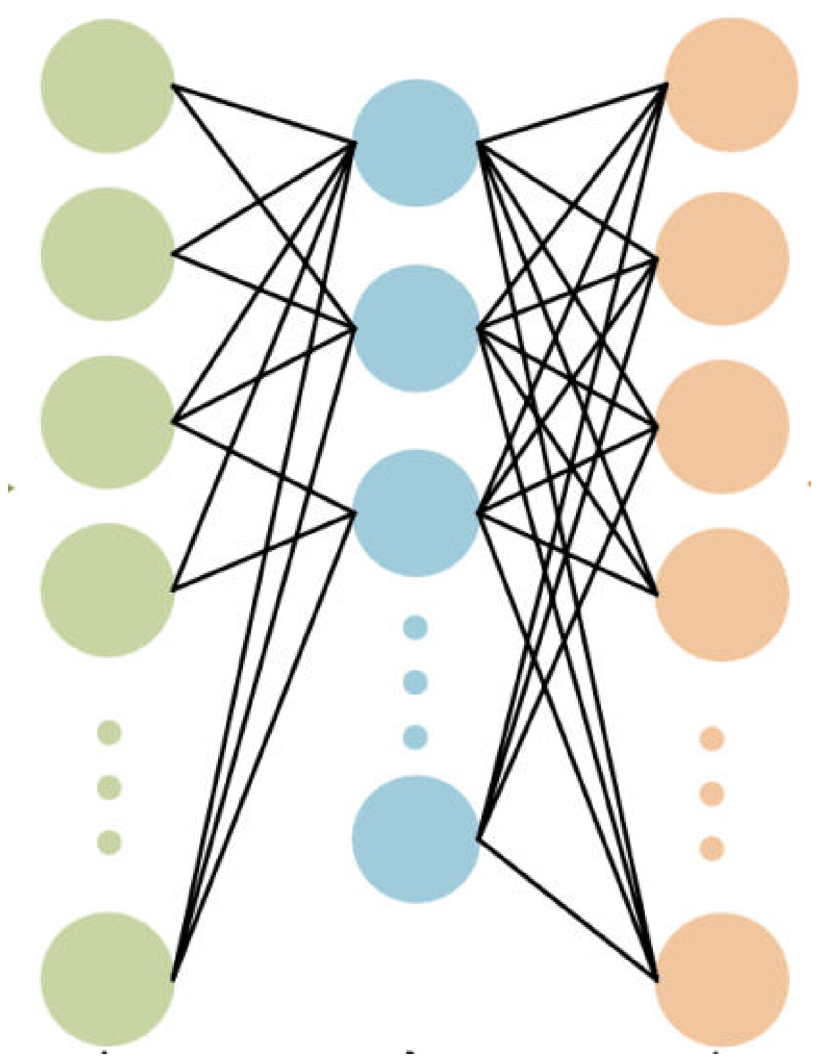
SPARSE AE



MSE LOSS

SPARSITY CONSTRAINT

SPARSE AE



Sparsity Constraint

Makes the **average activation (pj)** converge to a disired one (p)

$$J_{\text{SAEcost}}(W) = J_{\text{MSE}}(W) + J_{\text{Sparse}}(W)$$

$$J_{\text{Sparse}}(W) = \beta \sum_{i=1}^{2} KL(\rho | \rho_j)$$

$$KL(\rho | \rho_j) = \rho \log \frac{\rho}{\rho_j} + (1-\rho) \log \frac{1-\rho}{1-\rho_j}$$

CONSTRAINTS

On-Loss Constraints

LOSS = MSE + REGULARIZATION

ON-LOSS CONSTRAINTS

Weights decay

Control **overfitting** and **weigths' scale**

ON-LOSS CONSTRAINTS

Weights decay

Control **overfitting** and **weigths' scale**

LOSS =

ON-LOSS CONSTRAINTS

Weights decay

Control **overfitting** and **weigths' scale**

LOSS = **MSE**

ON-LOSS CONSTRAINTS

Weights decay

Control **overfitting** and **weigths' scale**

LOSS = **MSE** + $\lambda/2 \sum_{l=1}^{nl} \sum_{i=1}^{sl} \sum_{j=1}^{sl+1} \left(W_{ij}^{l}\right)^2$

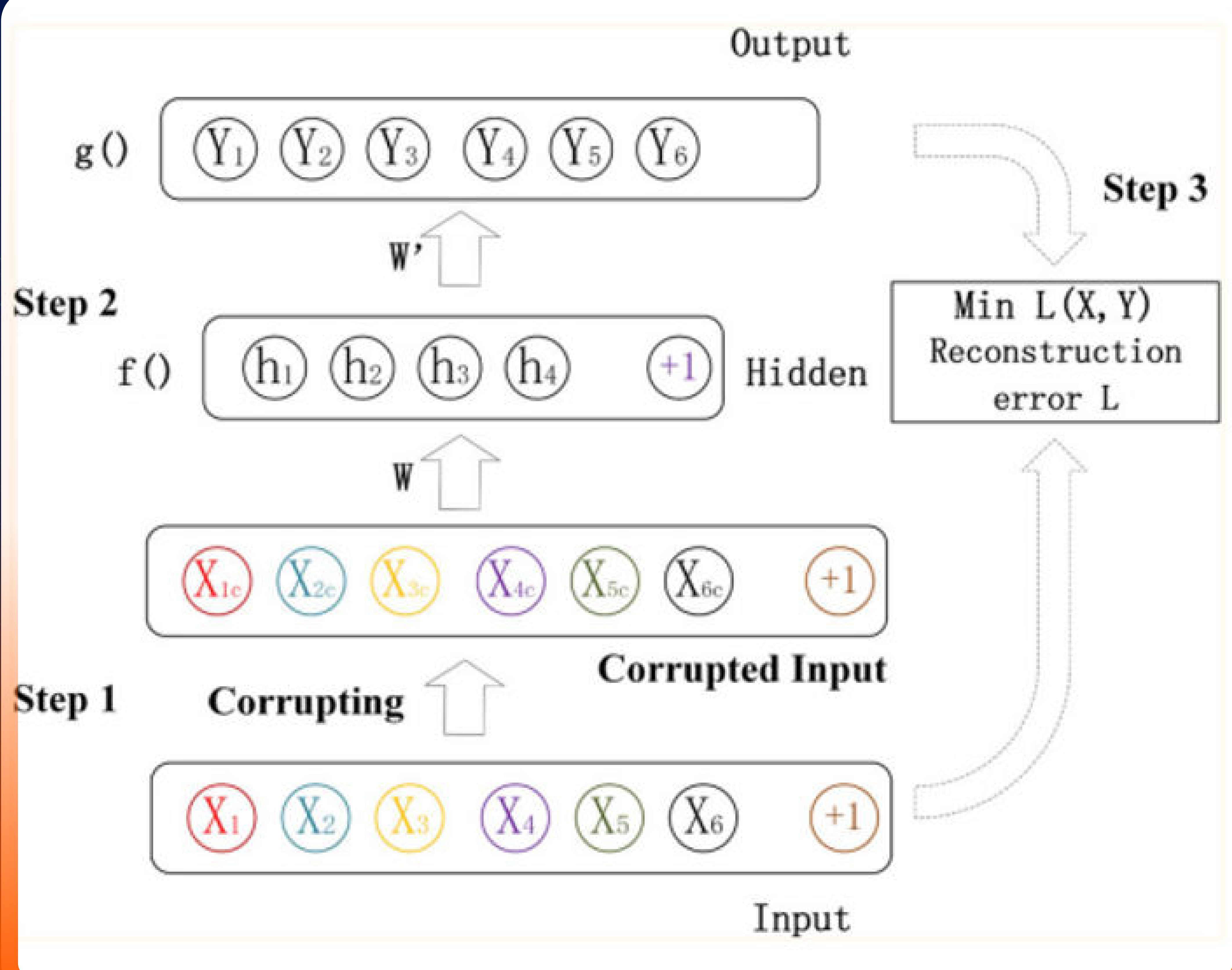
TRICK THE MODEL

Denoising autoencoder

Input + noise **= model more robust**

Choose a good noise method

TRICK THE MODEL



SAPIENZA
UNIVERSITÀ DI ROMA

ON-LOSS CONSTRAINTS

Locality Constraints

- **Overcomplete** representation for z
- Similar input, **similar z**

ON-LOSS CONSTRAINTS

Locality Constraints

Encoder

$$\mathbf{z} = s(\tilde{\mathbf{W}}^T\mathbf{x} + \mathbf{b}_e);$$

Decoder

$$\hat{\mathbf{x}} = \mathbf{W}g(\mathbf{z}; \tilde{\mathbf{W}}, \mathbf{x}) + \mathbf{b}_d$$

$$g(z_i; \tilde{\mathbf{W}}, x) = \begin{cases} z_i, & \text{if } \mathbf{w}_i \in \mathcal{N}_k(\mathbf{x}) \\ 0, & \text{otherwise} \end{cases}$$

ON-LOSS CONSTRAINTS

Locality Constraints

$$g(z_i; \tilde{\mathbf{W}}, x) = \begin{cases} z_i, & \text{if } \mathbf{w}_i \in \mathcal{N}_k(\mathbf{x}) \\ 0, & \text{otherwise} \end{cases}$$

Keep in **z only k features** according to the **k** neighbours of **x**.

ON-LOSS CONSTRAINTS

Weights decay

Control **overfitting** and **weigths' scale**

LOSS = $\frac{1}{L}\sum_{l=1}^{L}\|\mathbf{x}^{(l)} - \mathbf{W}g(\mathbf{z}^{(l)};\tilde{\mathbf{W}},\mathbf{x}^{(l)}) - \mathbf{b}_d\|_2^2$ + $+\lambda\sum_{i=1}^{N}\phi(z_i^{(l)})$

ON-LOSS CONSTRAINTS

Non negativity constraints

Another **term** on the **loss**

CONSTRAINTS IN LOGICAL FORM

CONSTRAINTS IN LOGICAL FORM

DANGER:

Logic is not differentiable

STE

Straight-Through Estimators

Replace in the **chain rule** of the **loss** wrt to **x** with a **sub differentiable function.**

STE

Straight-Through Estimators

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial b(x)} \times \frac{\partial b(x)}{\partial x}$$

STE

Straight-Through Estimators

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial b(x)} \times \frac{\partial b(x)}{\partial x}$$

b(x) = 1 IFF x>=0

b(x) = 0 OTHERWISE

STE

Straight-Throu... mators

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial b(x)}$$

b(x) = 1 IFF x>=0

b(x) = 0 OTHERWISE

STE

Straight-Through Estimators

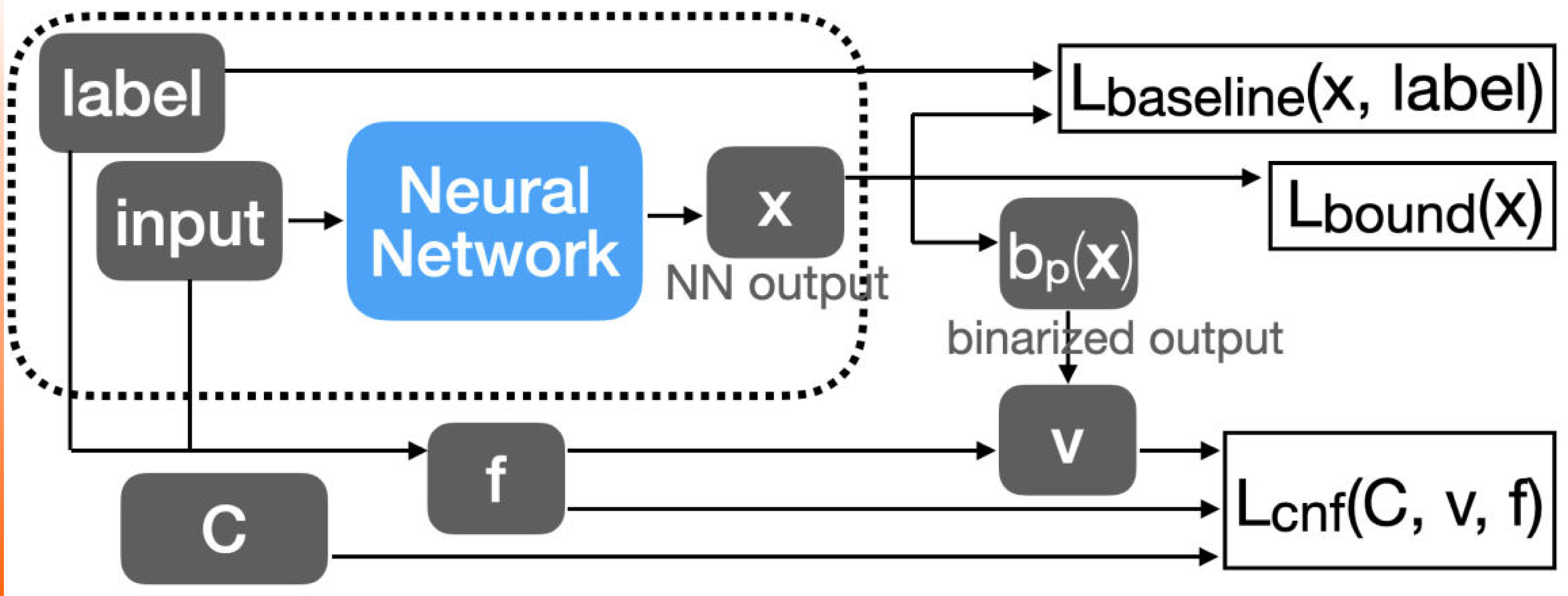
$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial b(x)} \times \boxed{\frac{\partial s(x)}{\partial x}}$$

STE

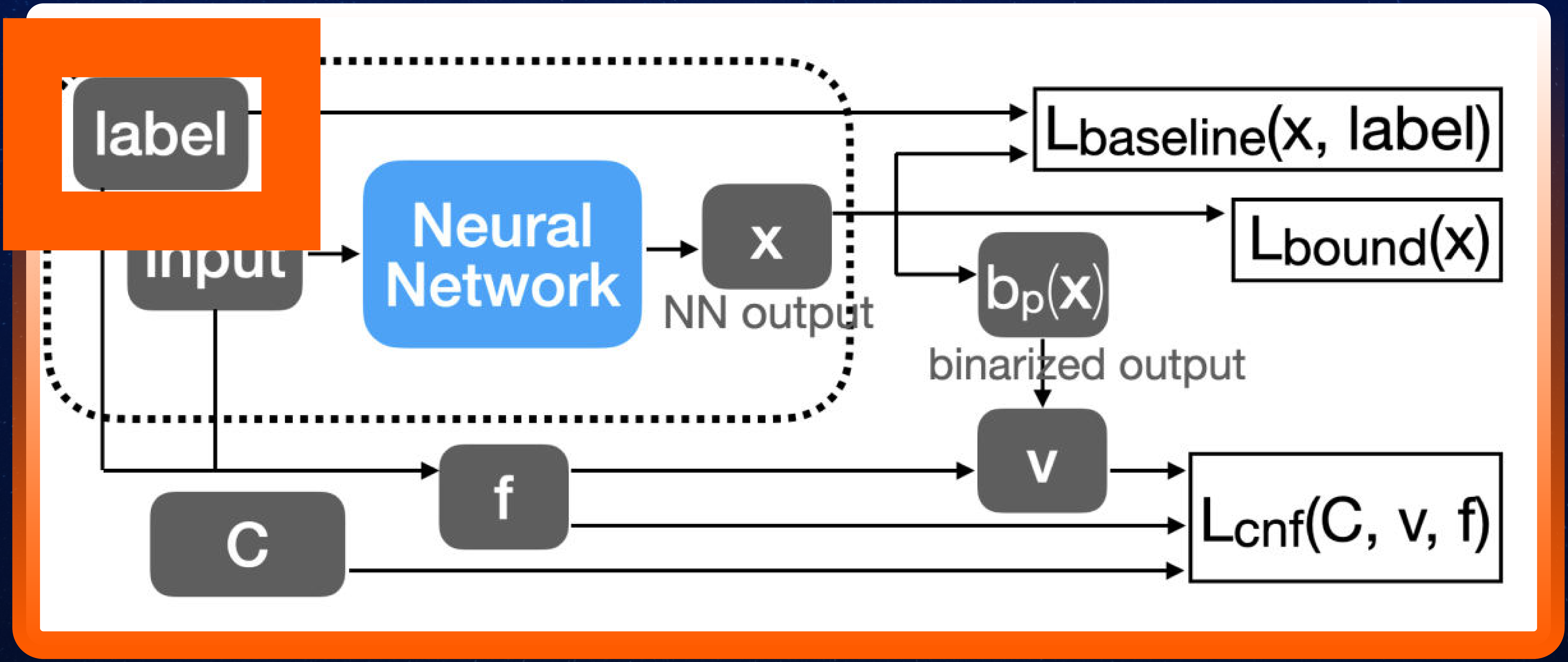
Straight-

$$\frac{\partial L}{\partial x} =$$

LOGIC CONSTRAINTS



LOGIC CONSTRAINTS



LOGIC CONSTRAINTS

HOW DOES IT WORK WITHOUT LABEL?

LOGIC CONSTRAINTS

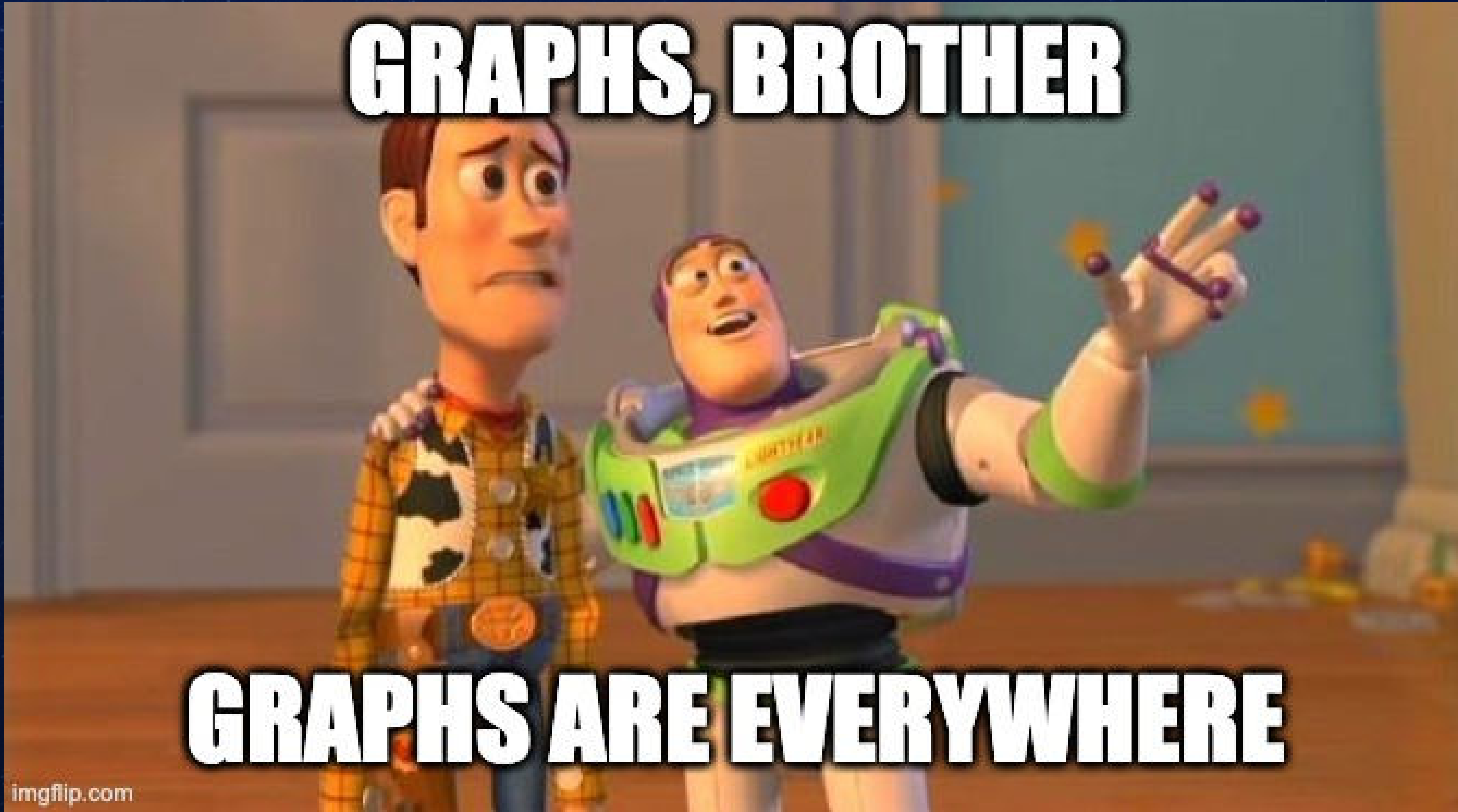
HOW DOES IT WORK WITHOUT LABEL?

Our label is the x itself..

This method has to be adapted! **IN ANY CASE**

CONSTRAINT AS GRAPH

DOMI-KNOWS (for NLP)



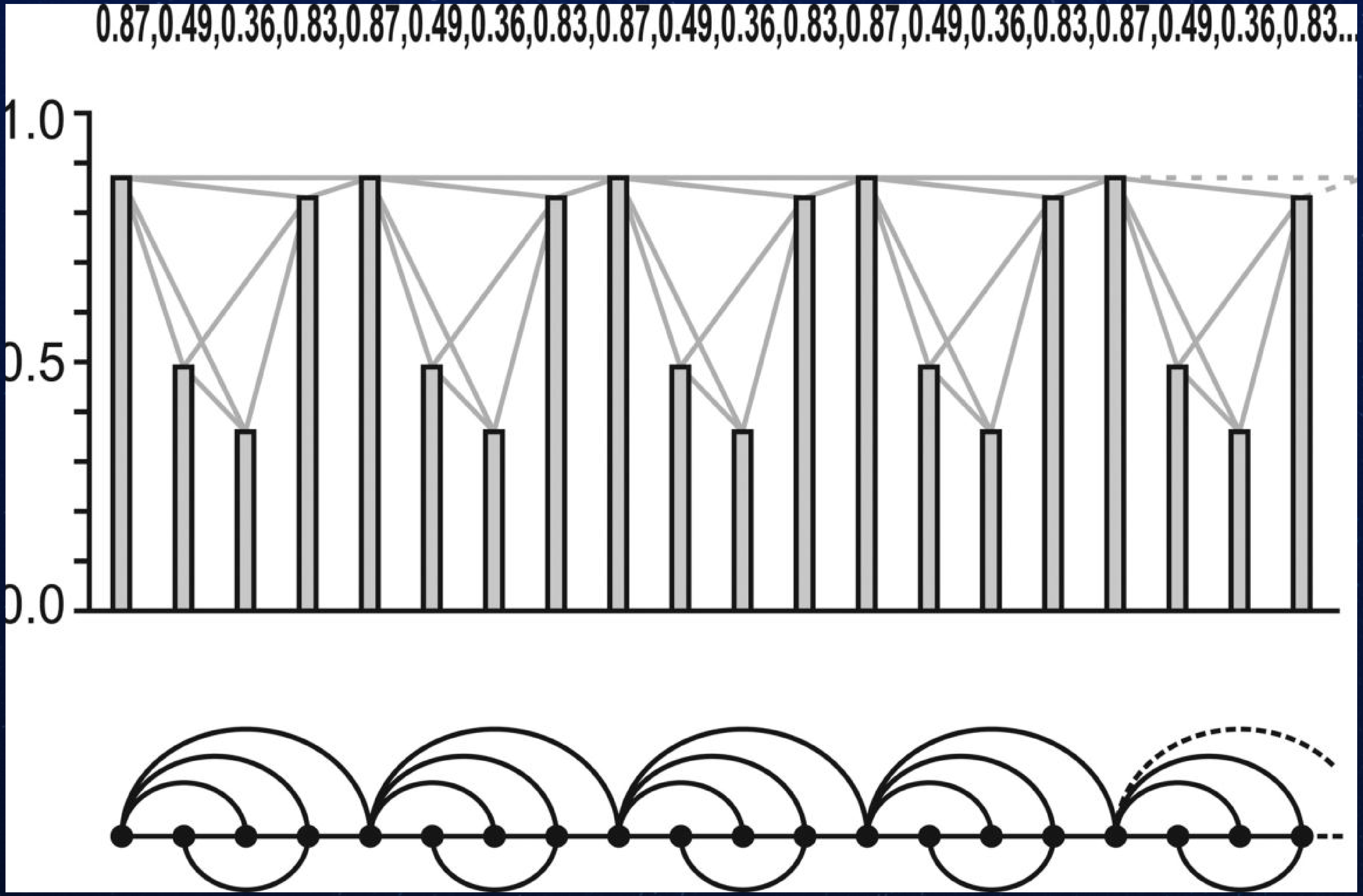
CONSTR. AS GRAPH

PROBLEM:

We have a **time series, not a graph**

CONSTR. AS GRAPH

SOLUTION: Visibility Graph



Sapienza Università di Roma

CONSTR. AS GRAPH

Steps:

Define problem domain as **graph**

- Define **concepts**:
  - **Basic** concept: a vector (a node of the graph)
  - **Compositional** concept: linked(n1,n2) if they are close enough in time (ni is a node)
- **Edges:**
  - there is a potential issue: there are only is_a, has_a, contains

CONSTR. AS GRAPH

Express constraints

Do it in term of **graph**

IS THERE ANY EXTERNAL KNOWLEDGE???

CONSTR. AS GRAPH

MODEL

SENSORS

Non **trainable**

LEARNER

Differentiable sparse autoenc.

CONSTR. AS GRAPH

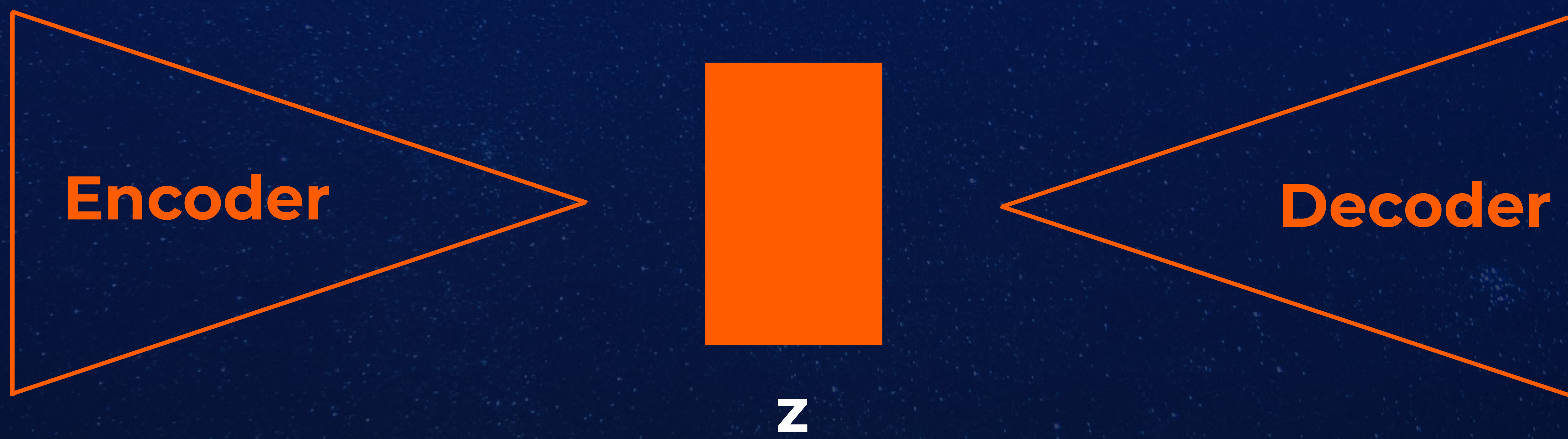
NEED TO
GO
DEEP

MY WORK

MY SOLUTION IN A 2 PHASE MODEL

PHASE 1

Training of a (sparse?) autoencoder

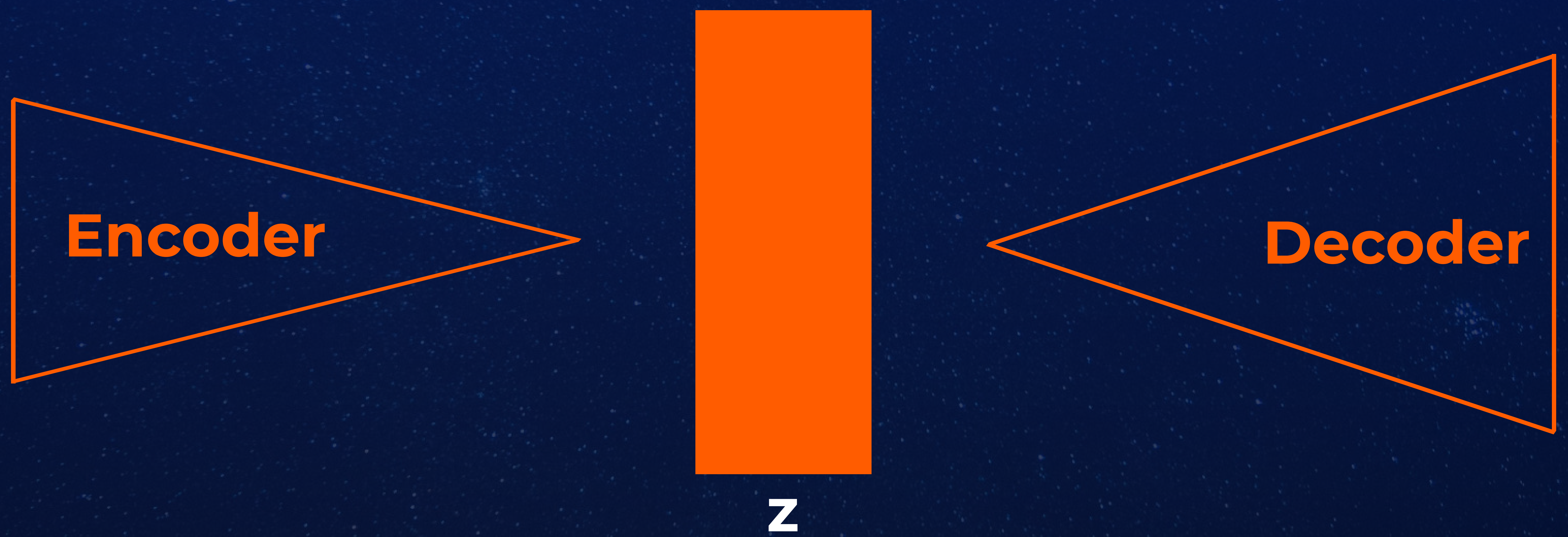


This will require **test differents losses**
and a good **HPO campaign**

SAPIENZA
UNIVERSITÀ DI ROMA

PHASE 1

Training of a (sparse?) autoencoder

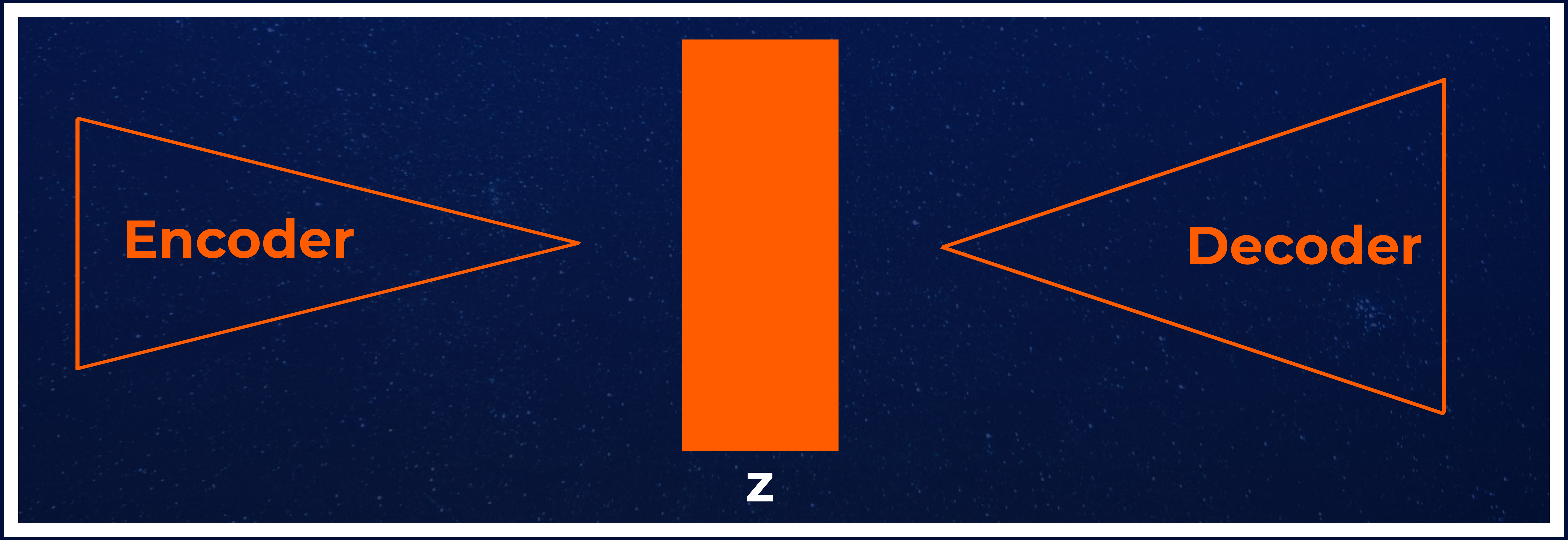


This will require **test differents losses** and a good **HPO campaign**

SAPIENZA
UNIVERSITÀ DI ROMA

PHASE 1

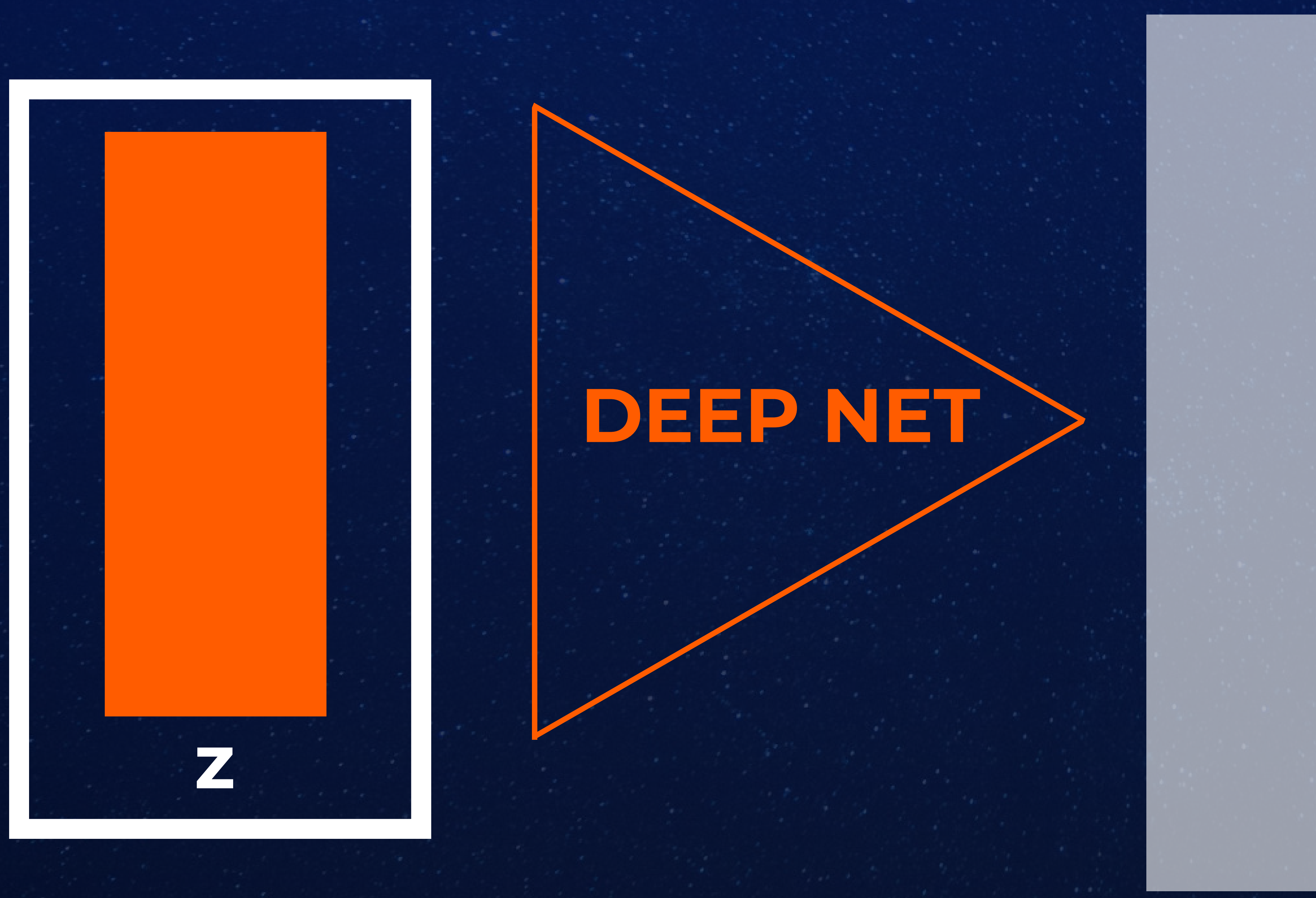
Training of a (sparse?) autoencoder



This will require **test differents losses** and a good **HPO campaign**

PHASE 2

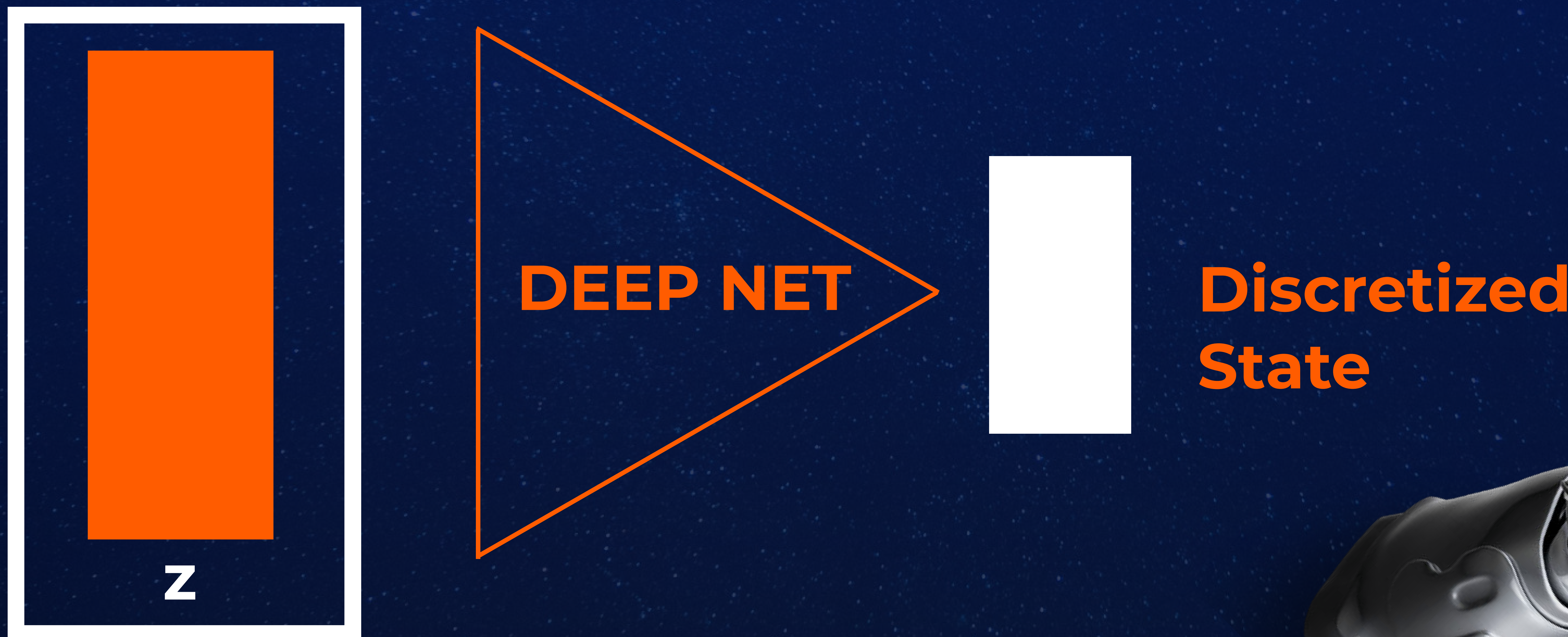
Discretization process



Discretized State

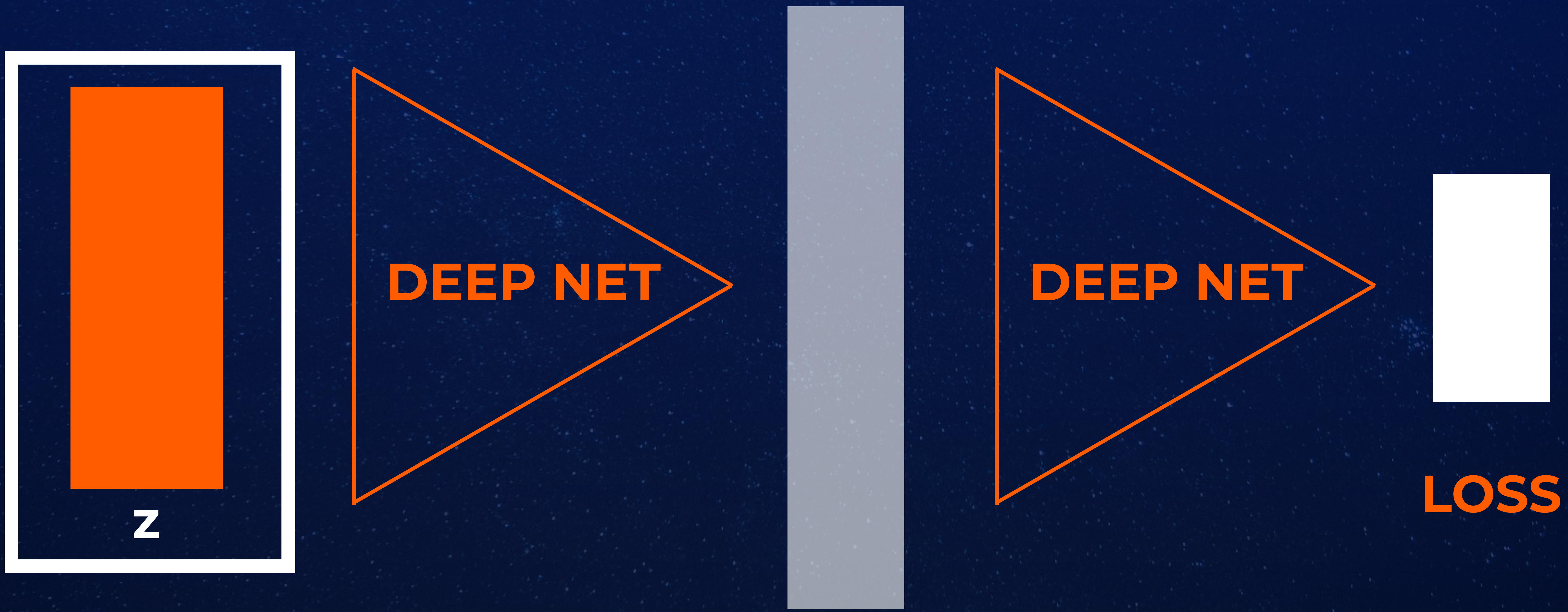
PHASE 2

Discretization process



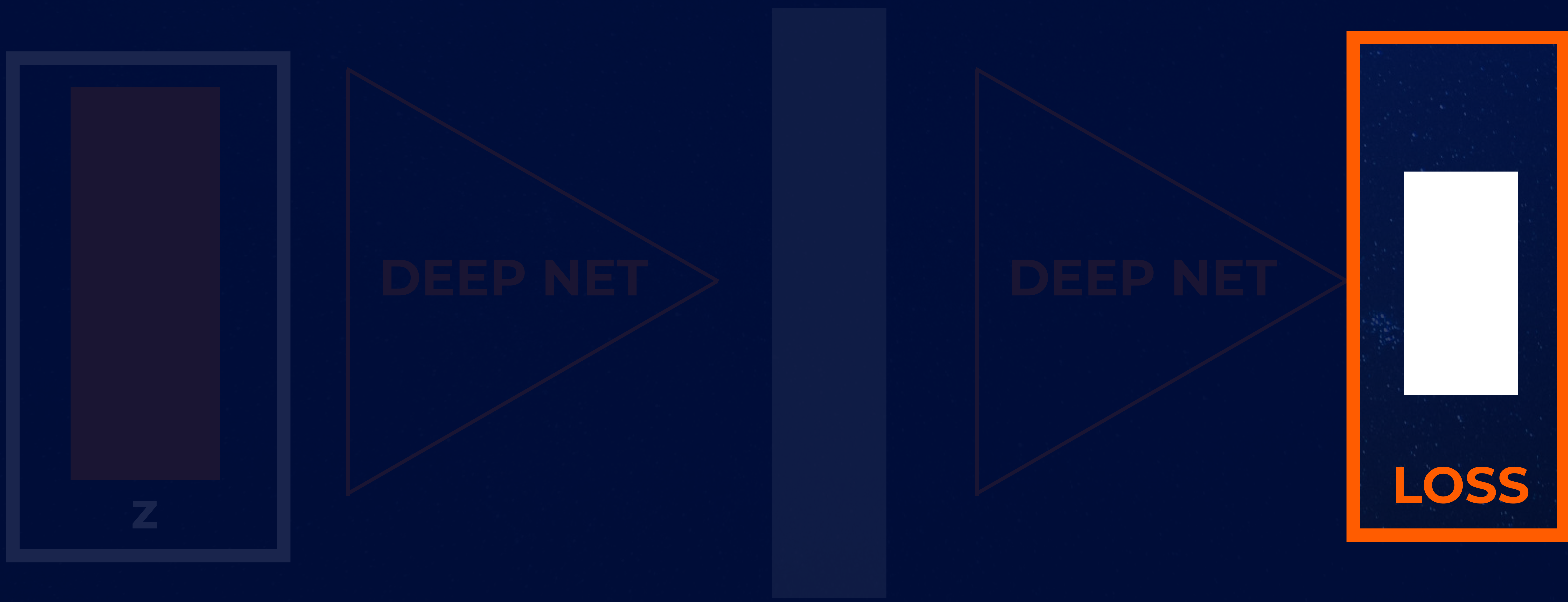
PHASE 2

Discretization Training



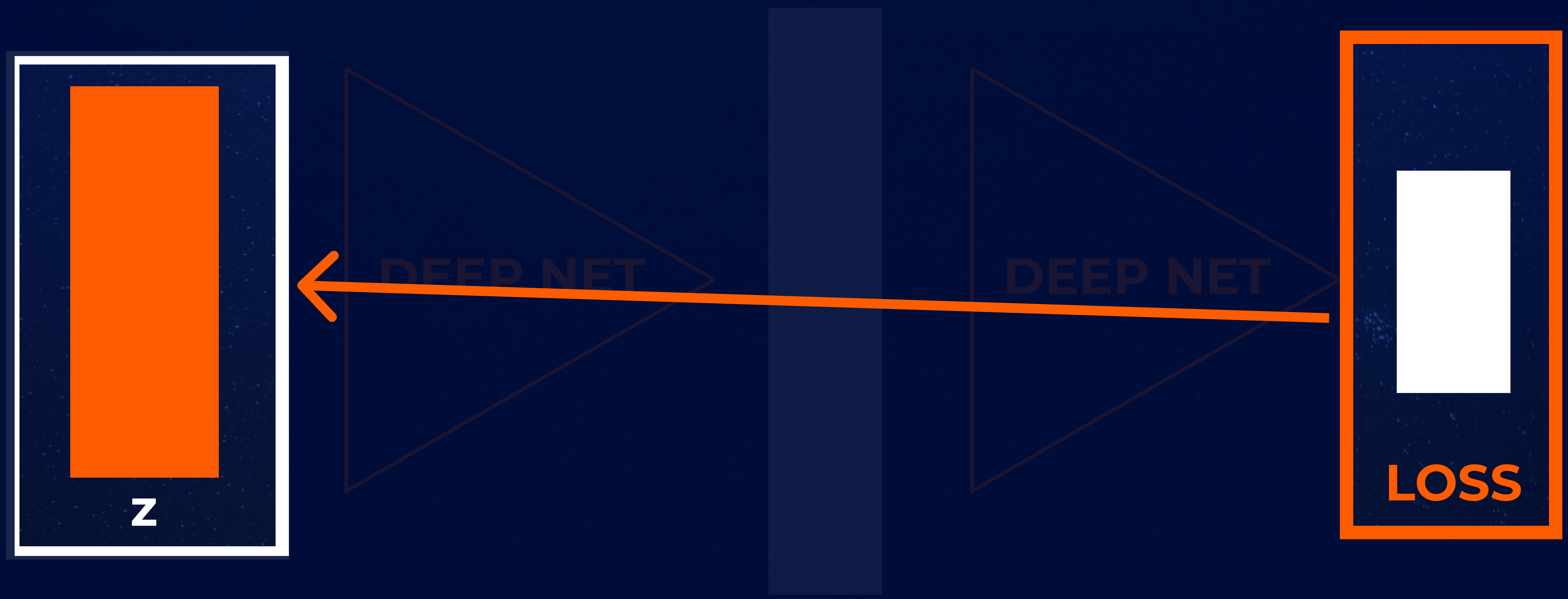
PHASE 2

Discretization Training



PHASE 2

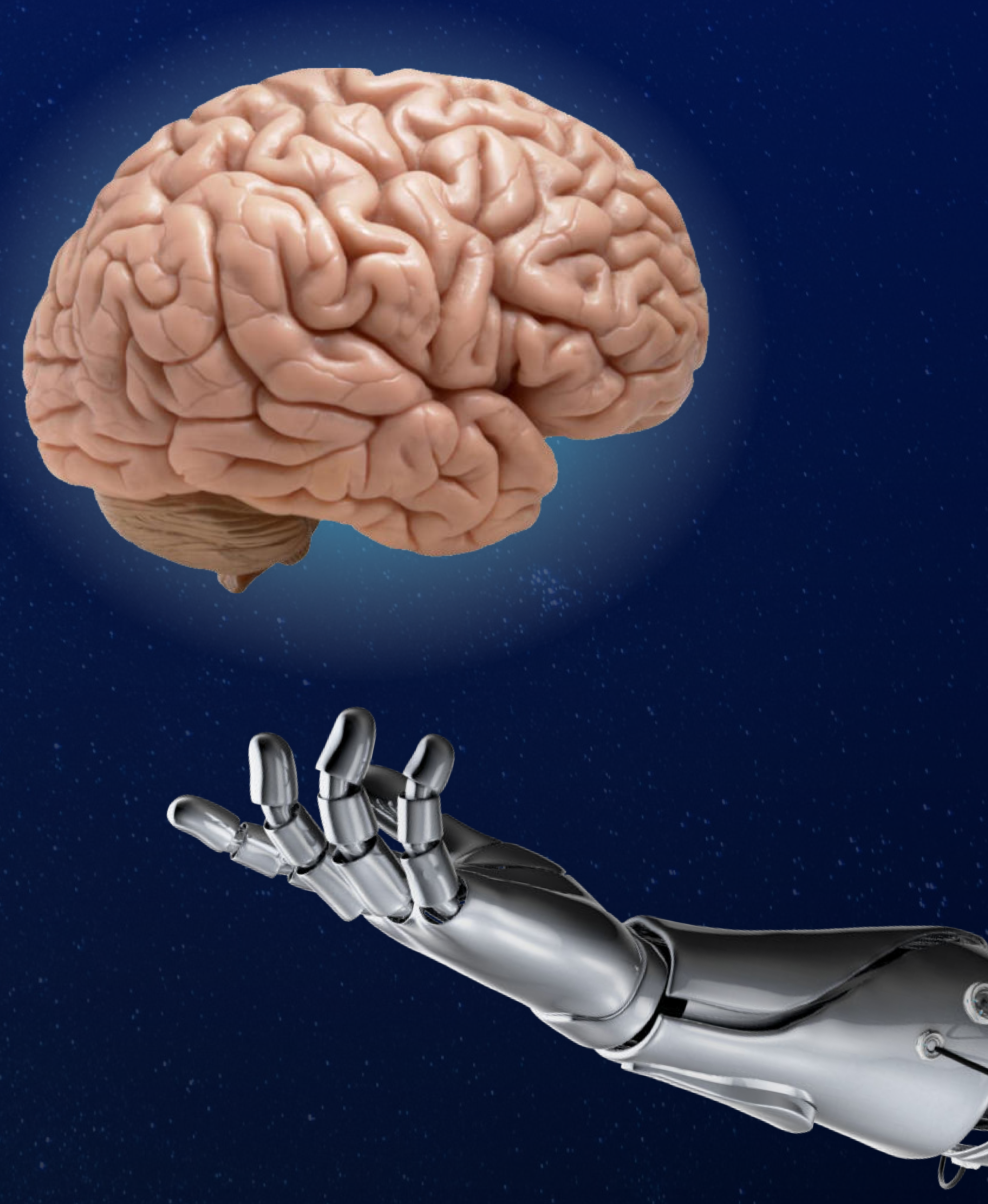
AutoEncoder Finetuning



SAPIENZA
UNIVERSITÀ DI ROMA

THE END

QUESTIONS?



EMANUELE RUCCI