Are the laws of Physics the most general ones?

A machine learning approach to face the problem

Enrico Lari

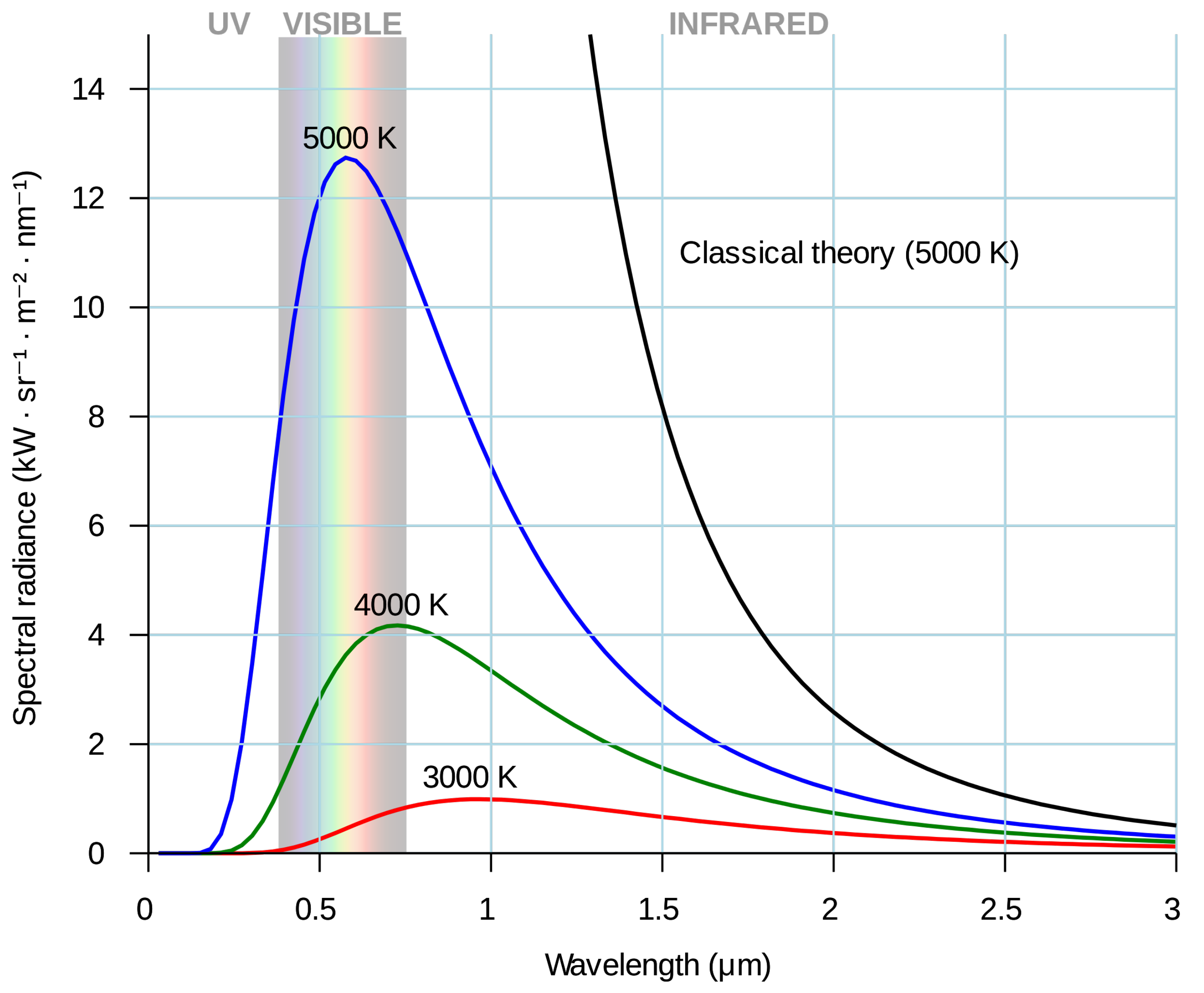
24/06/2019

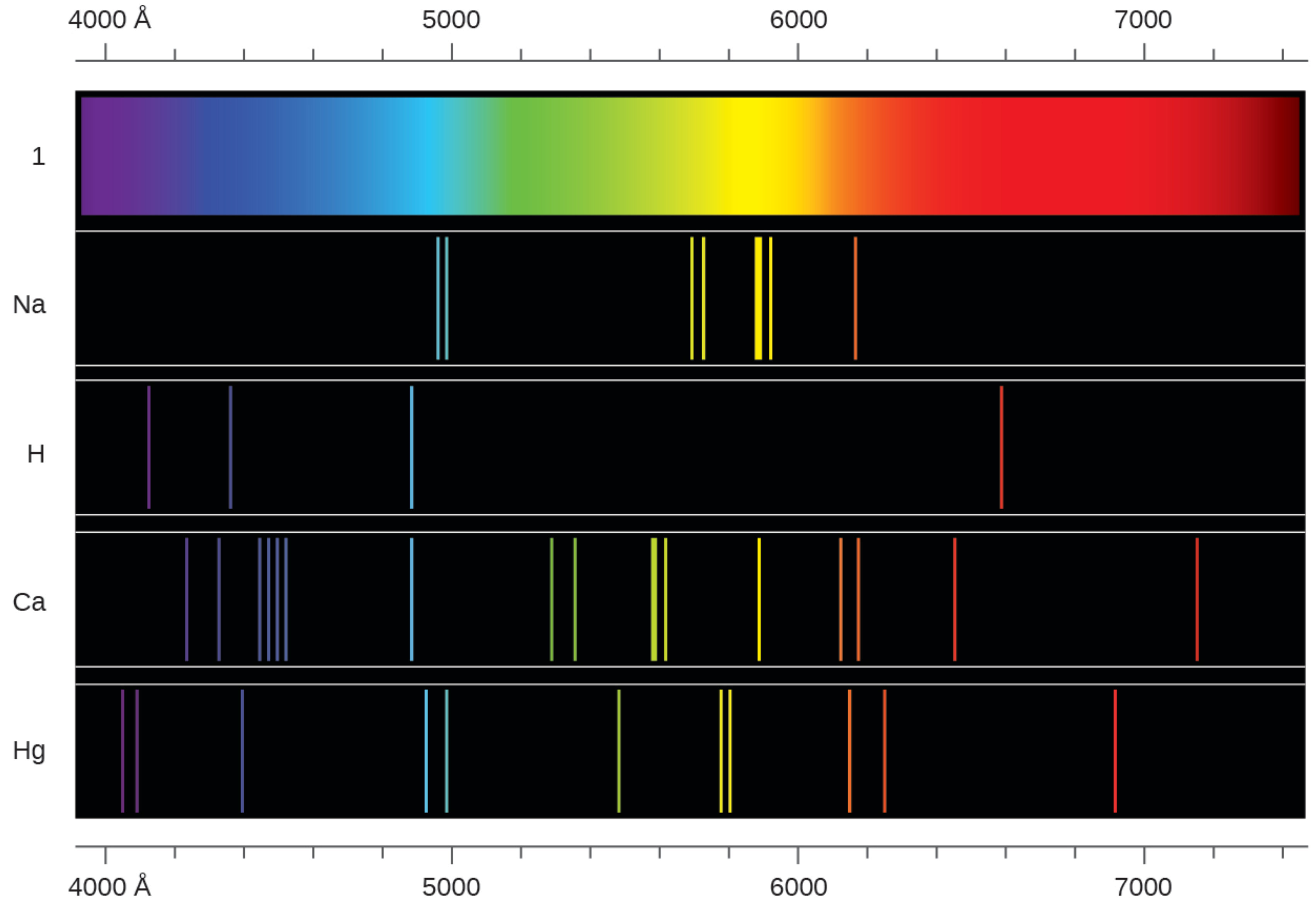
Outline

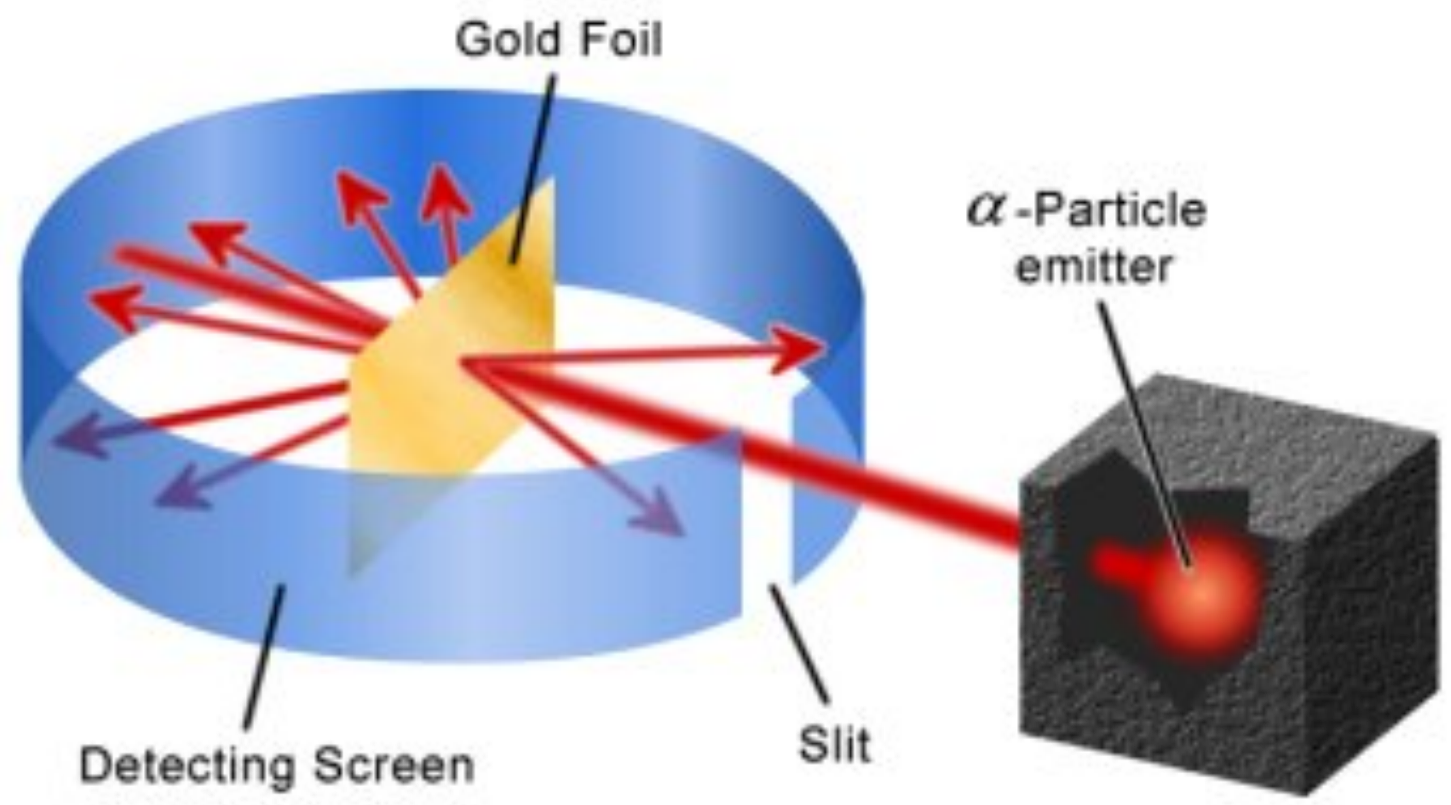
- Definition of the problem
- Modeling of the Physicist’s reasoning
- Implementation
- Examples
- Conclusions

Definition of the problem

Suppose to be in the early years of '900s…

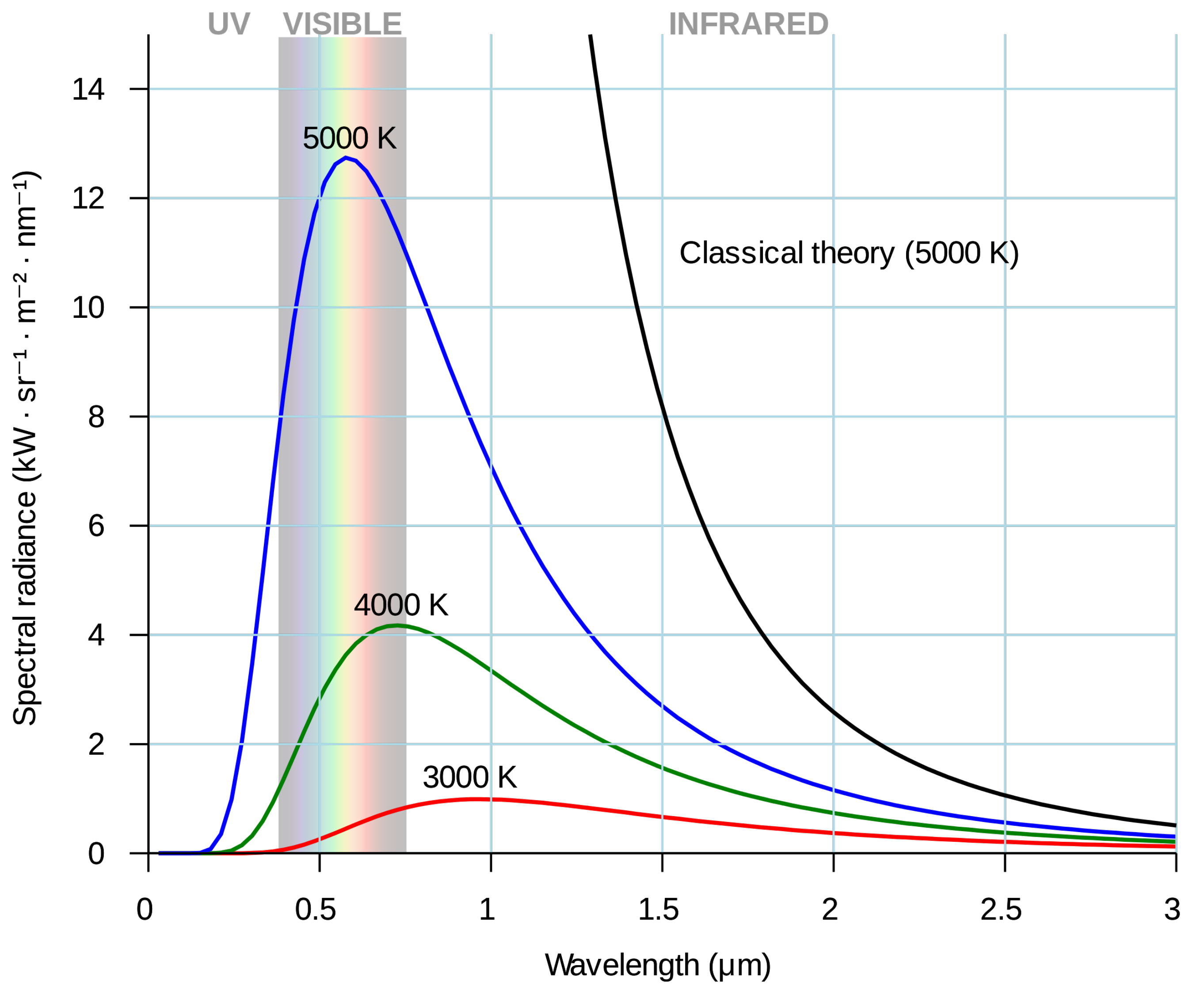


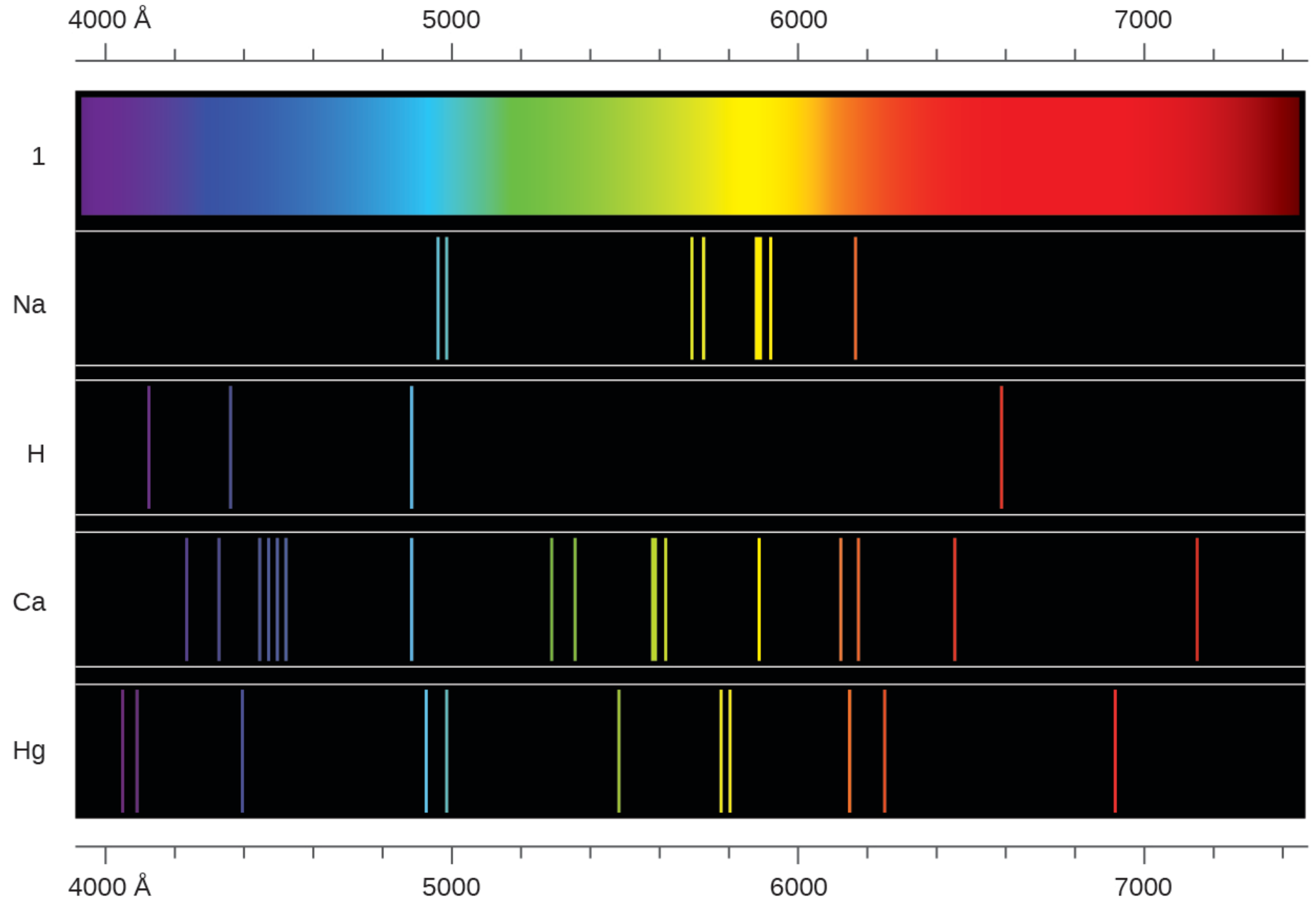


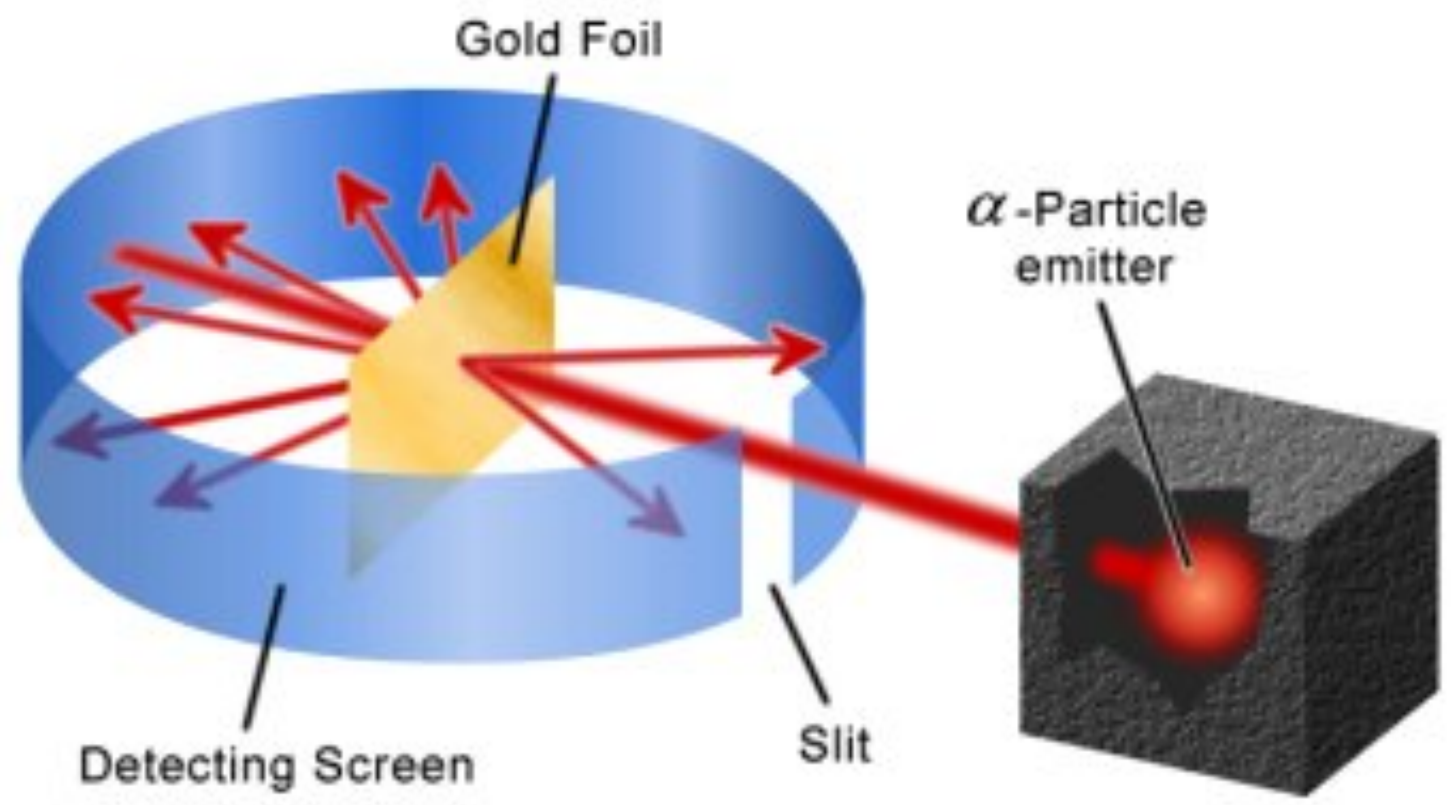


Definition of the problem

Suppose to be in the early years of ‘900s…



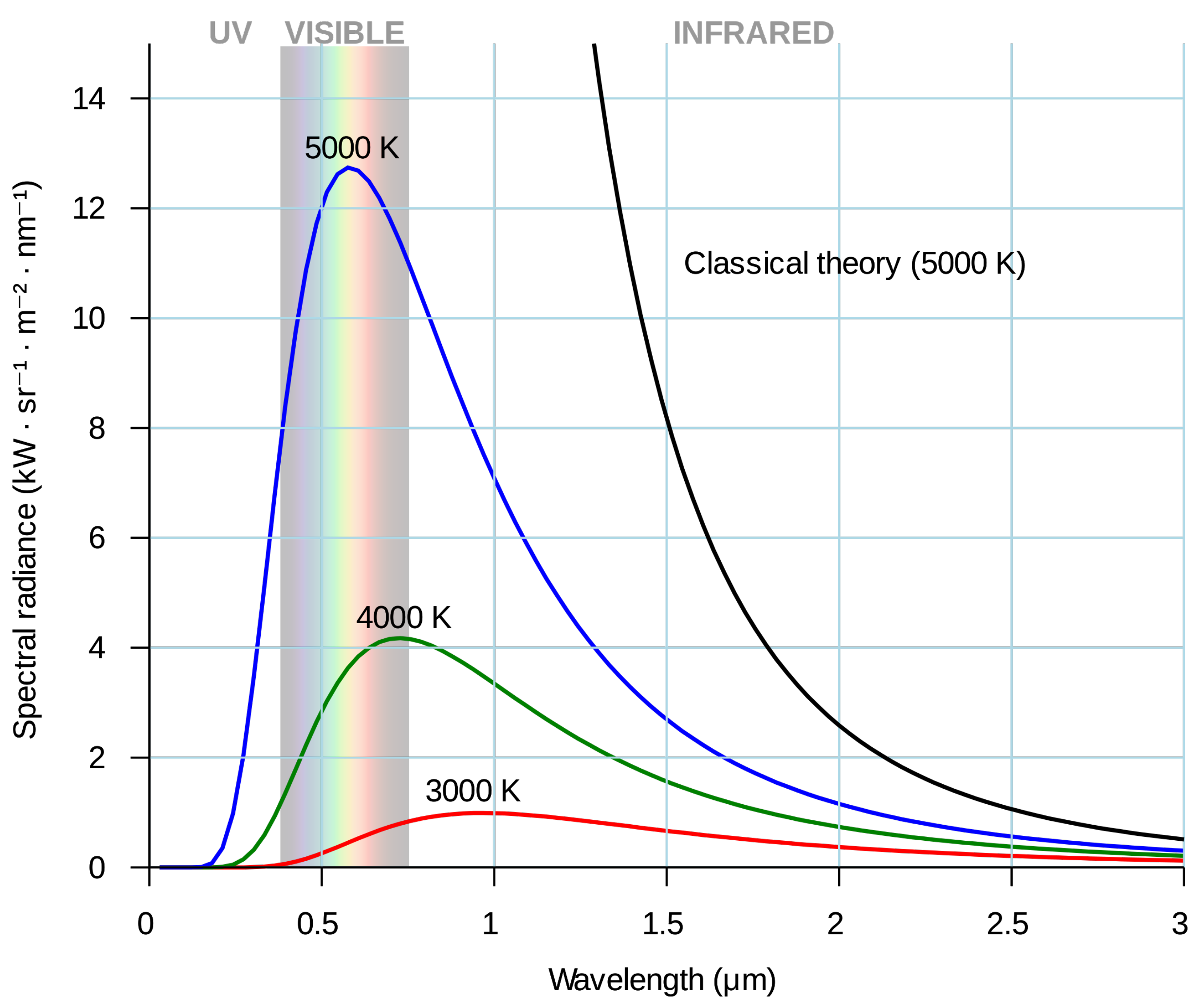


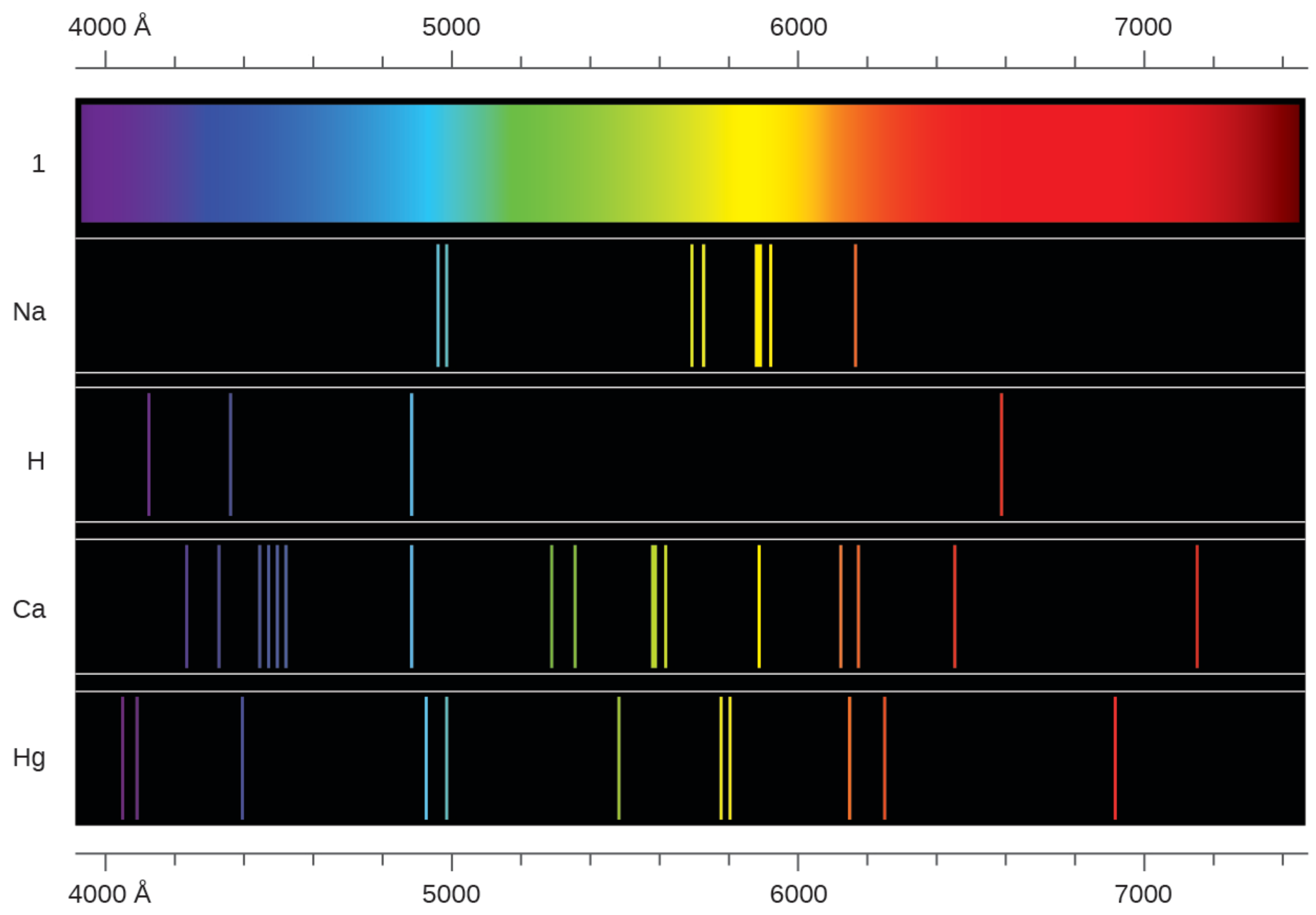


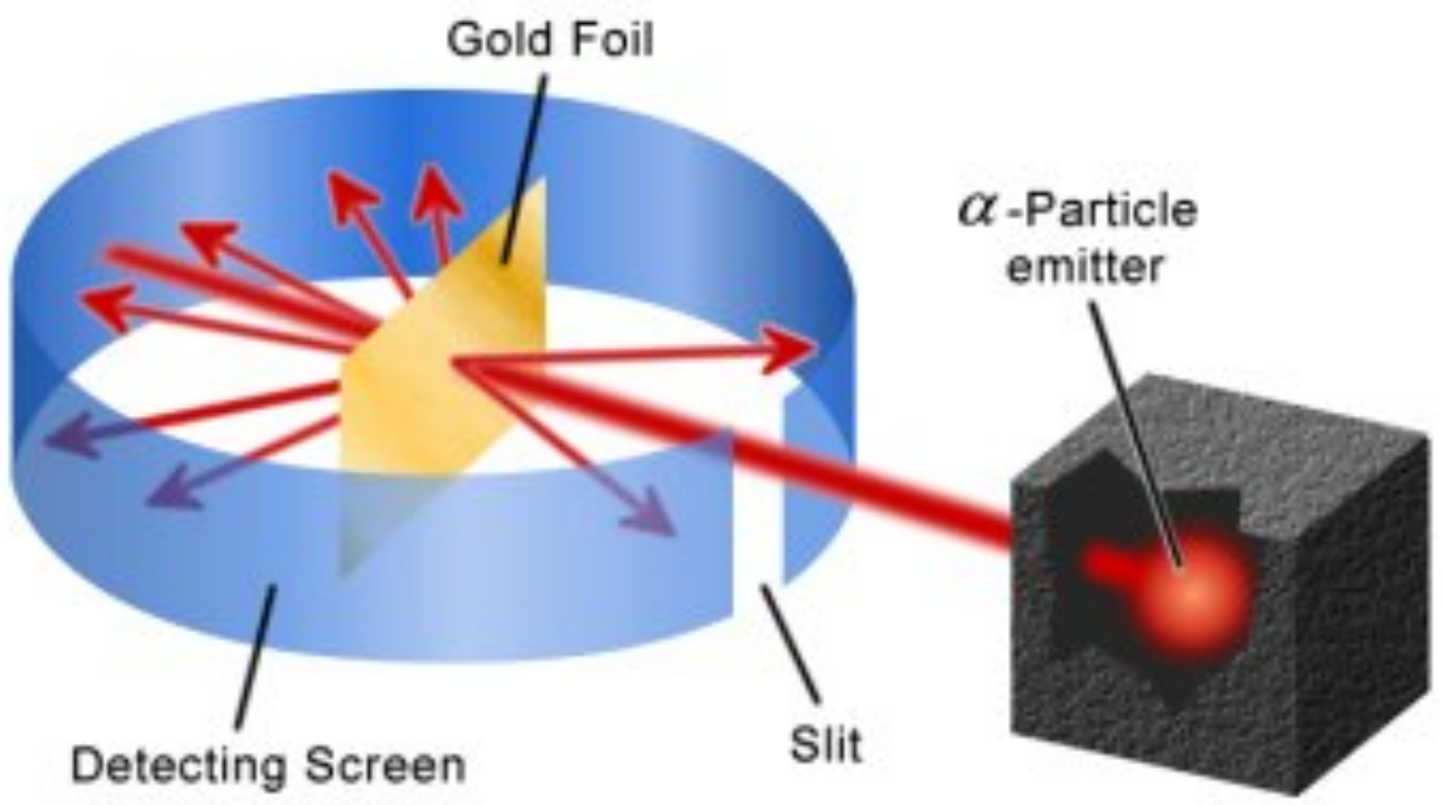
Classical Physics

Definition of the problem

Suppose to be in the early years of ‘900s…







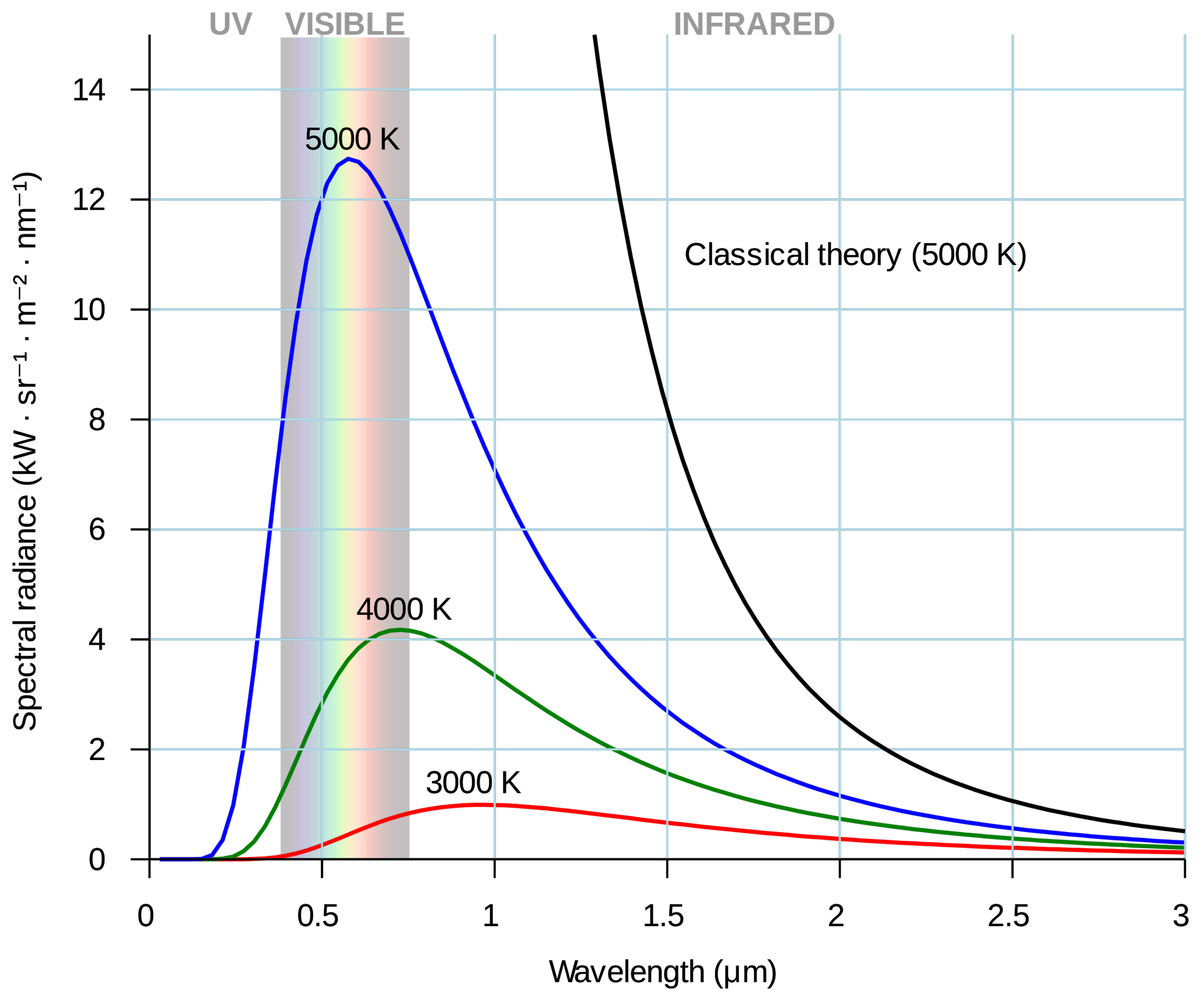
$$q_i \to \hat{q}_i$$

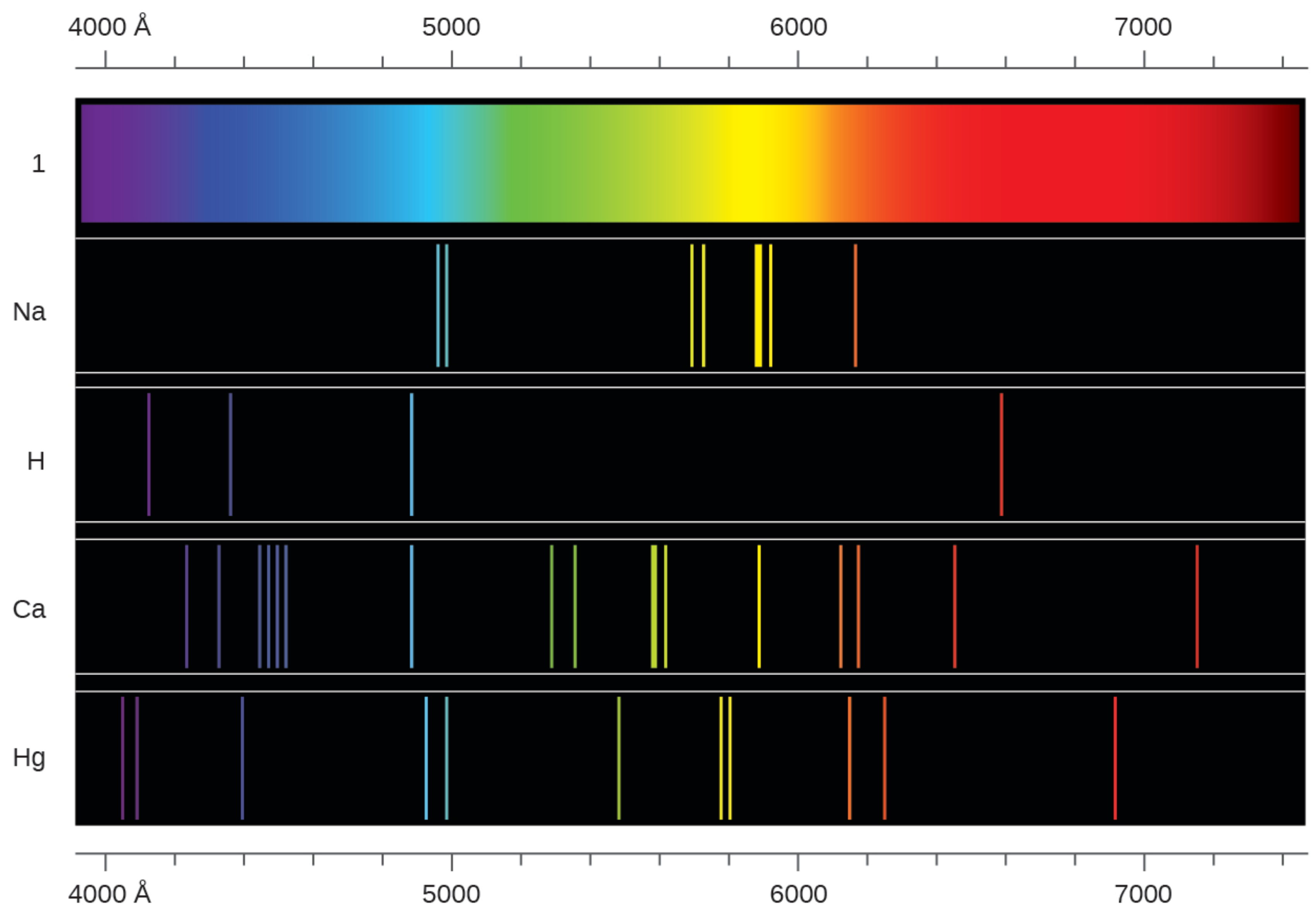
Classical Physics ⟶ $$p_i \to \hat{p}_i$$

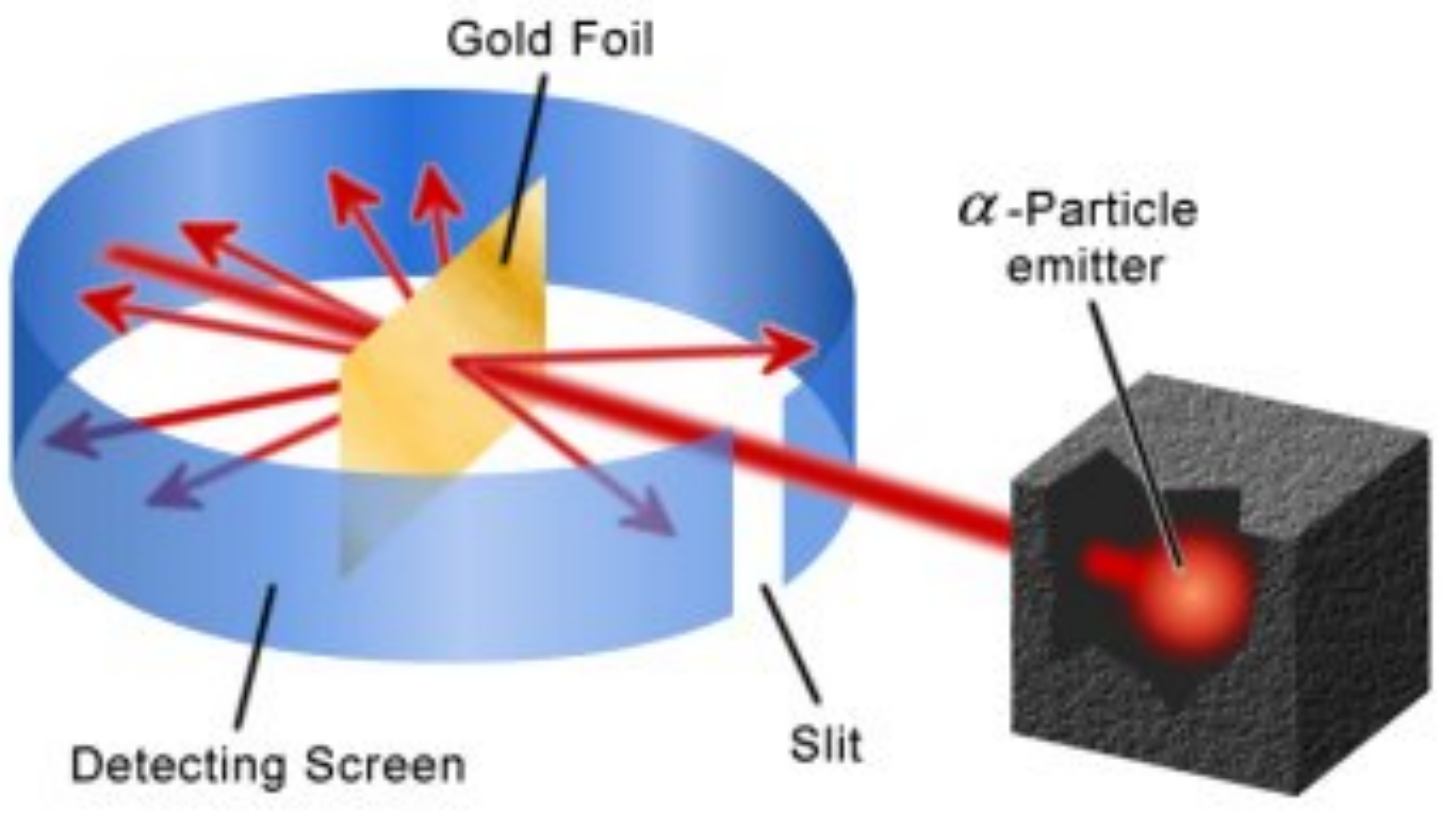
$$[\hat{q}_i, \hat{p}_j] = i\hbar\delta_{ij}$$

Definition of the problem

Suppose to be in the early years of ‘900s…







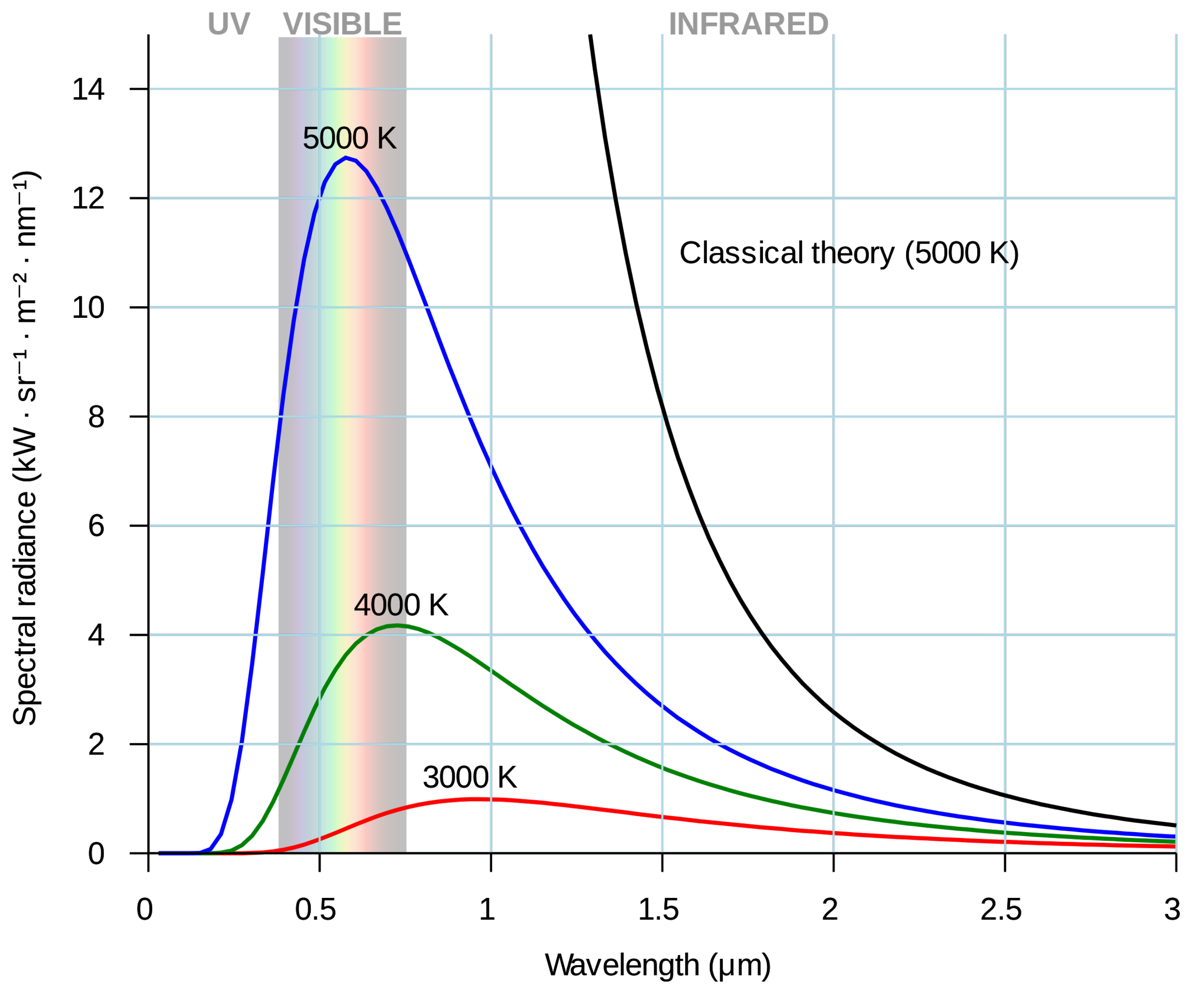
$$q_i \to \hat{q}_i$$

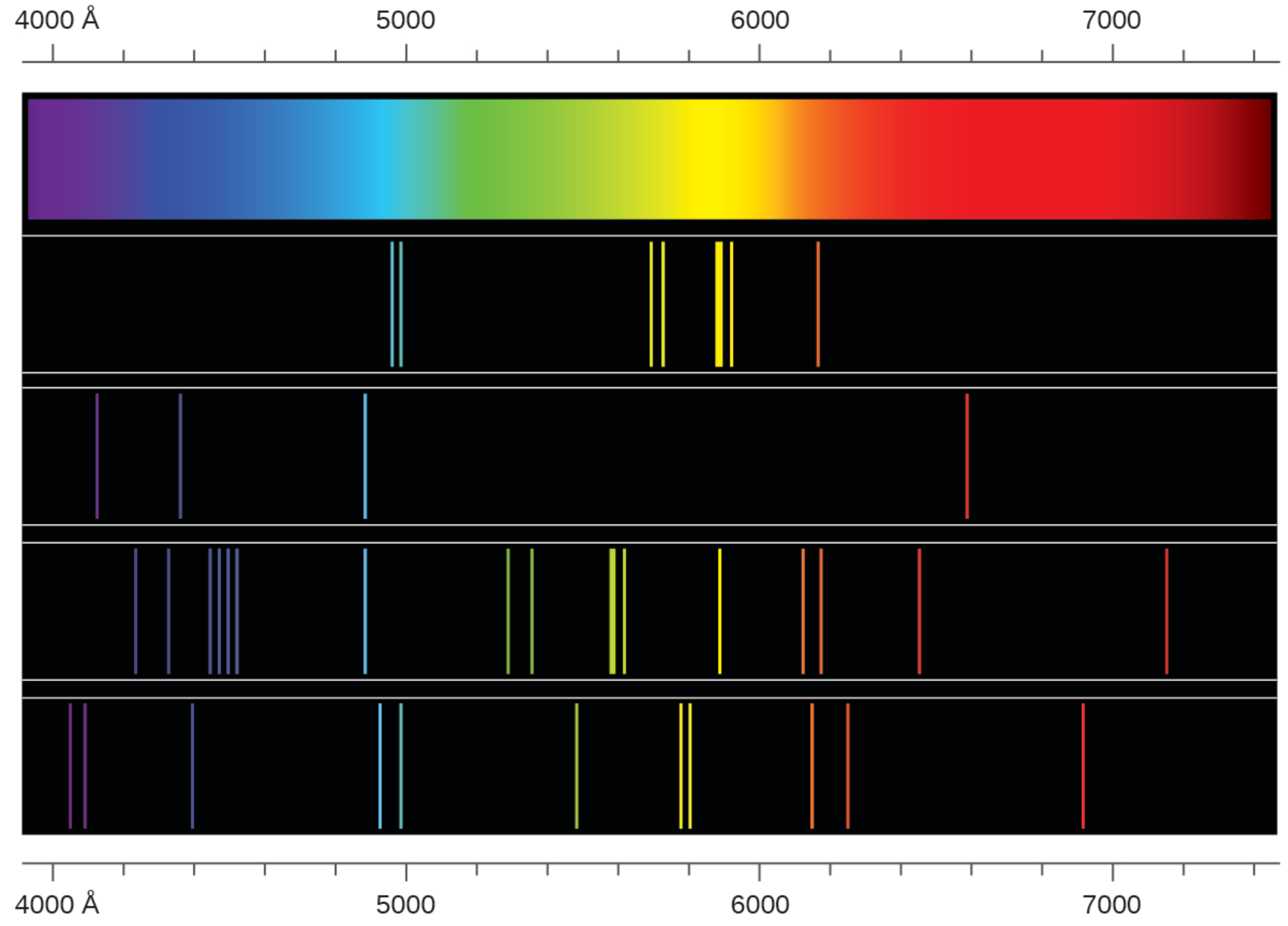
Classical Physics ⟶ $$p_i \to \hat{p}_i$$ ⟶ **Quantum Physics**

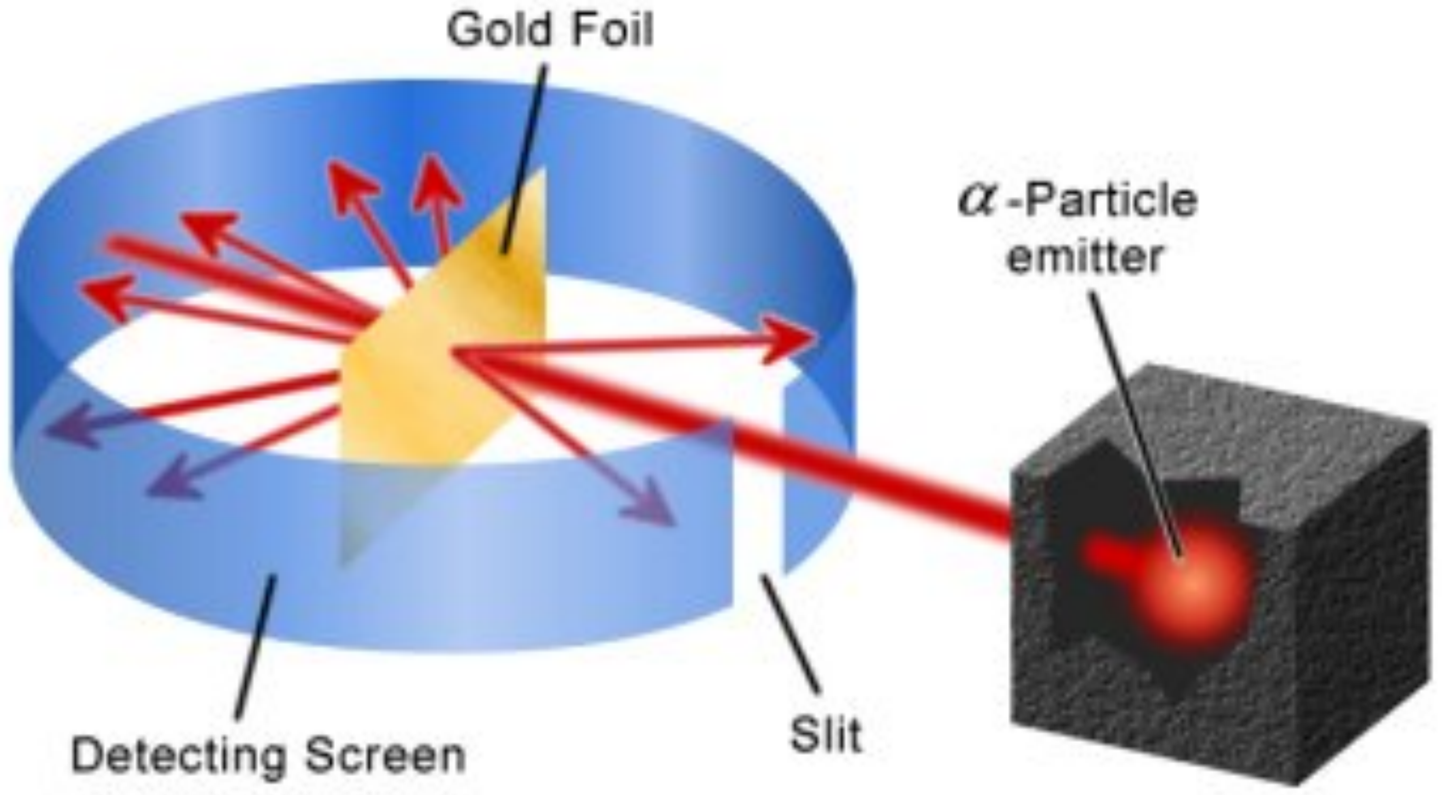
$$[\hat{q}_i, \hat{p}_j] = i\hbar\delta_{ij}$$

Definition of the problem

Suppose to be in the early years of ‘900s…





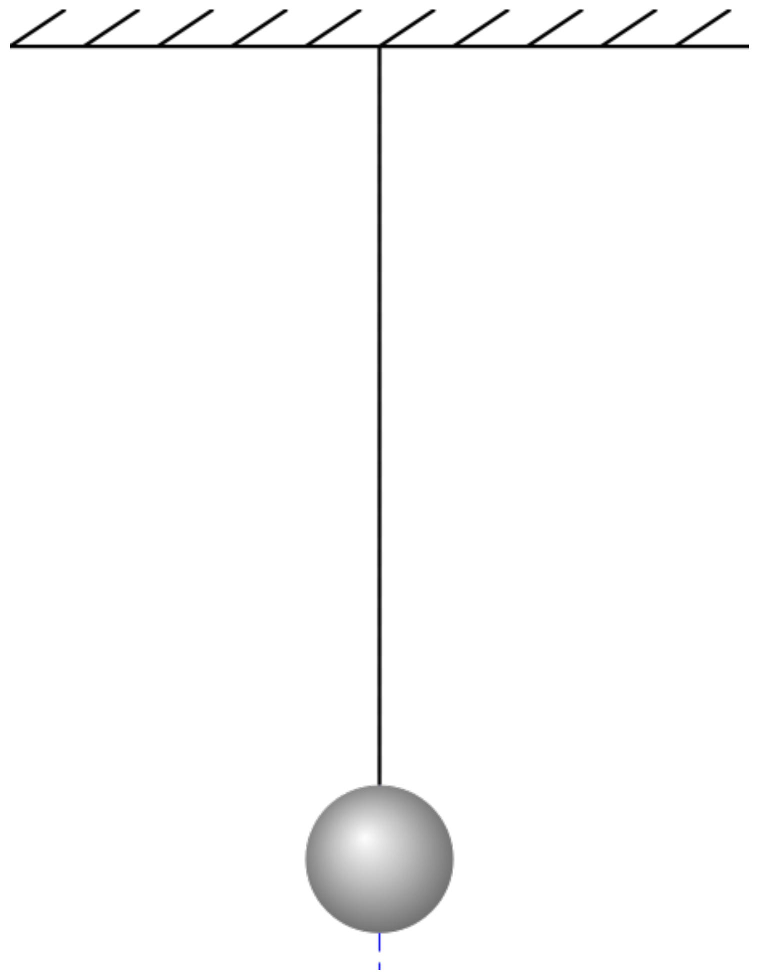


Classical Physics ⟶ **Quantum Physics**

$[q_i, \hat{p}_j] = i\hbar\delta_{ij}$

Theoretical physics is influenced by the schools of thought prevalent at the time of development

Definition of the problem



Lookup table:
write down measured pairs $(t_i, x(t_i))$

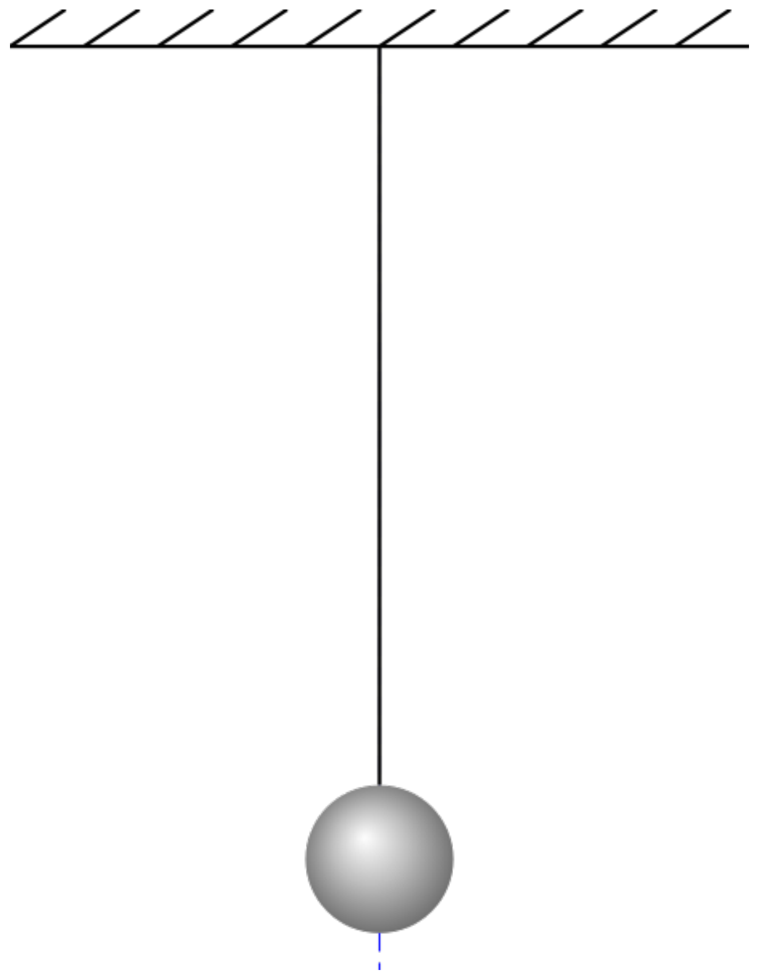
Write down the law of motion and evaluate it at

$$t_{pred} \to x(t_{pred})$$

Where will the pendulum be at time t_{pred}?

The law of motion is a **compressed representation** of reality

Definition of the problem



Where will the pendulum be at time t_{pred}?

Lookup table:
write down measured pairs $(t_i, x(t_i))$

Write down the law of motion and evaluate it at

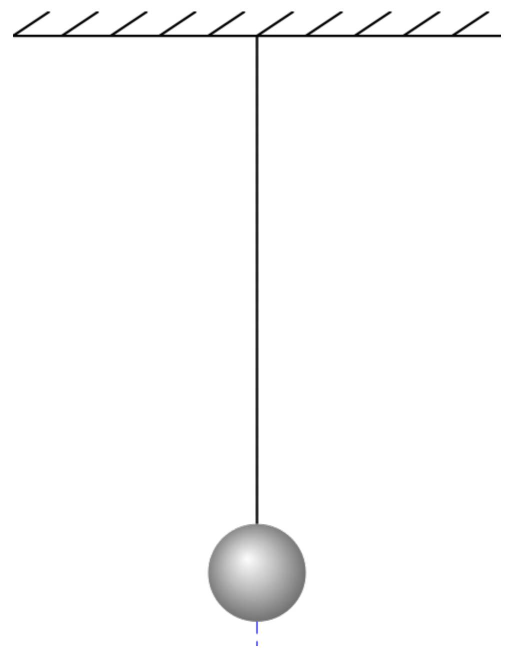
$$t_{pred} \to x(t_{pred})$$

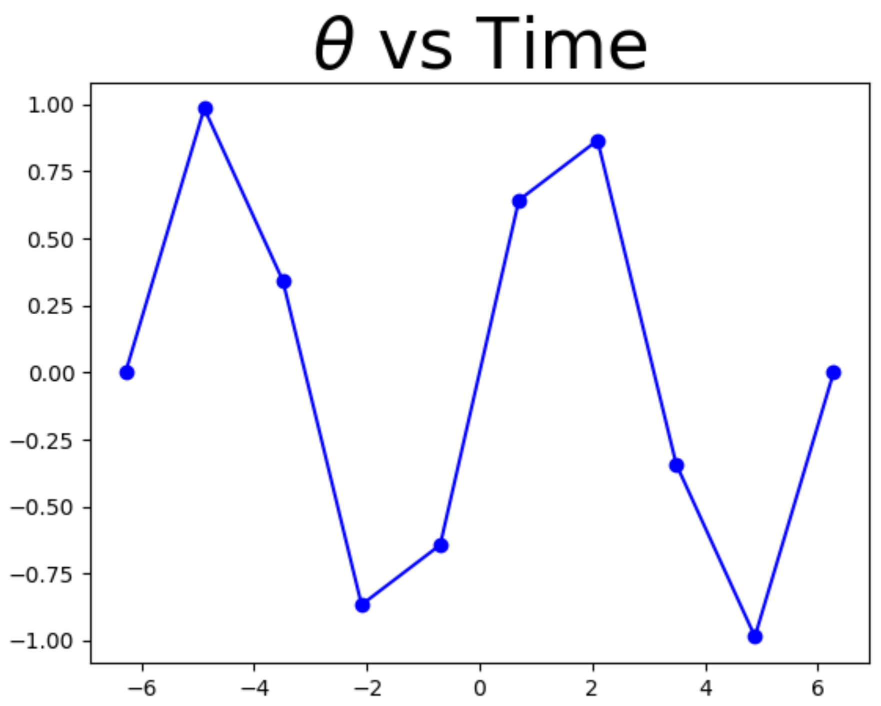
The law of motion is a compressed representation of reality

Use Machine Learning

Implementation

A Physicist is a (very efficient) “Machine”





Build a representation

$$\begin{cases} \theta_0 = \dots \\ \omega = \dots \end{cases}$$

Make predictions

$$\theta(t_{pred}) = \theta_0 + \omega \cdot t_{pred}$$

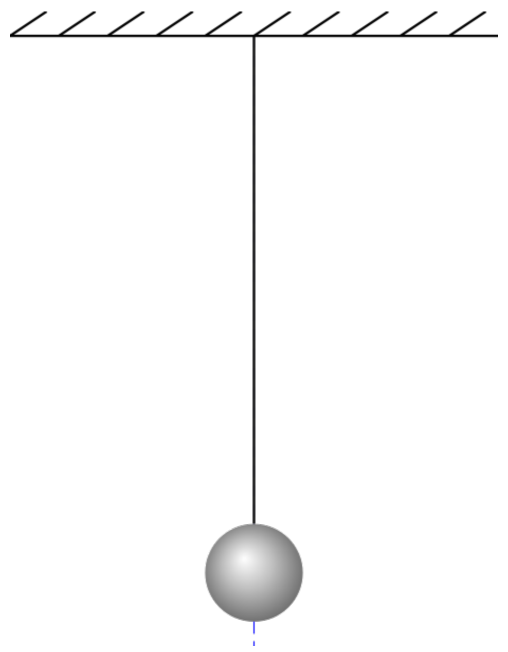
Encoding

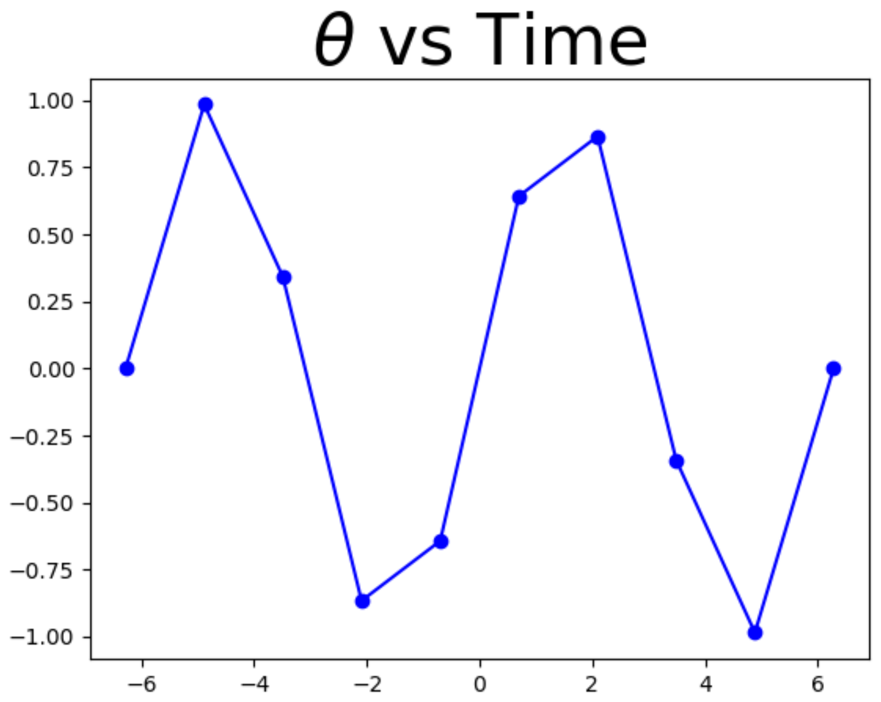
$\theta(t_{pred})$?

Decoding

Implementation

A Physicist is a (very efficient) “Machine”





Build a representation

$$\begin{cases} \theta_0 = \dots \\ \omega = \dots \end{cases}$$

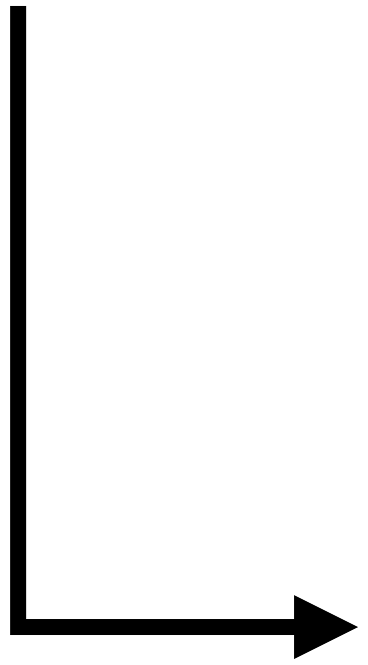
Make predictions

$$\theta(t_{pred}) = \theta_0 + \omega \cdot t_{pred}$$

Encoding

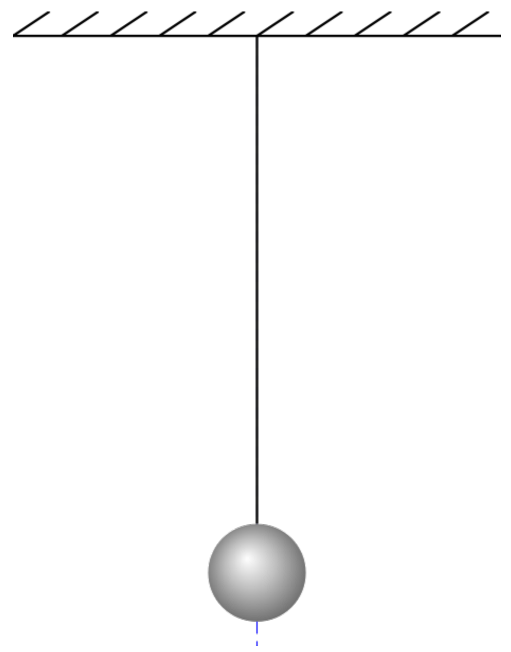
$\theta(t_{pred})$?

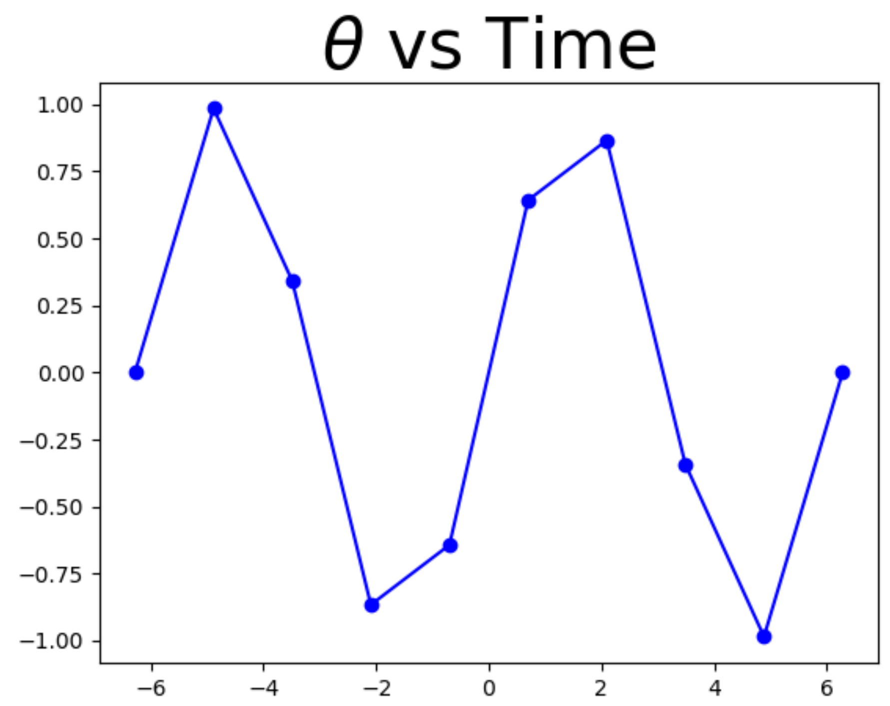
Decoding



Implementation

A Physicist is a (very efficient) "Machine"





Build a representation

$$\begin{cases} \theta_0 = \dots \\ \omega = \dots \end{cases}$$

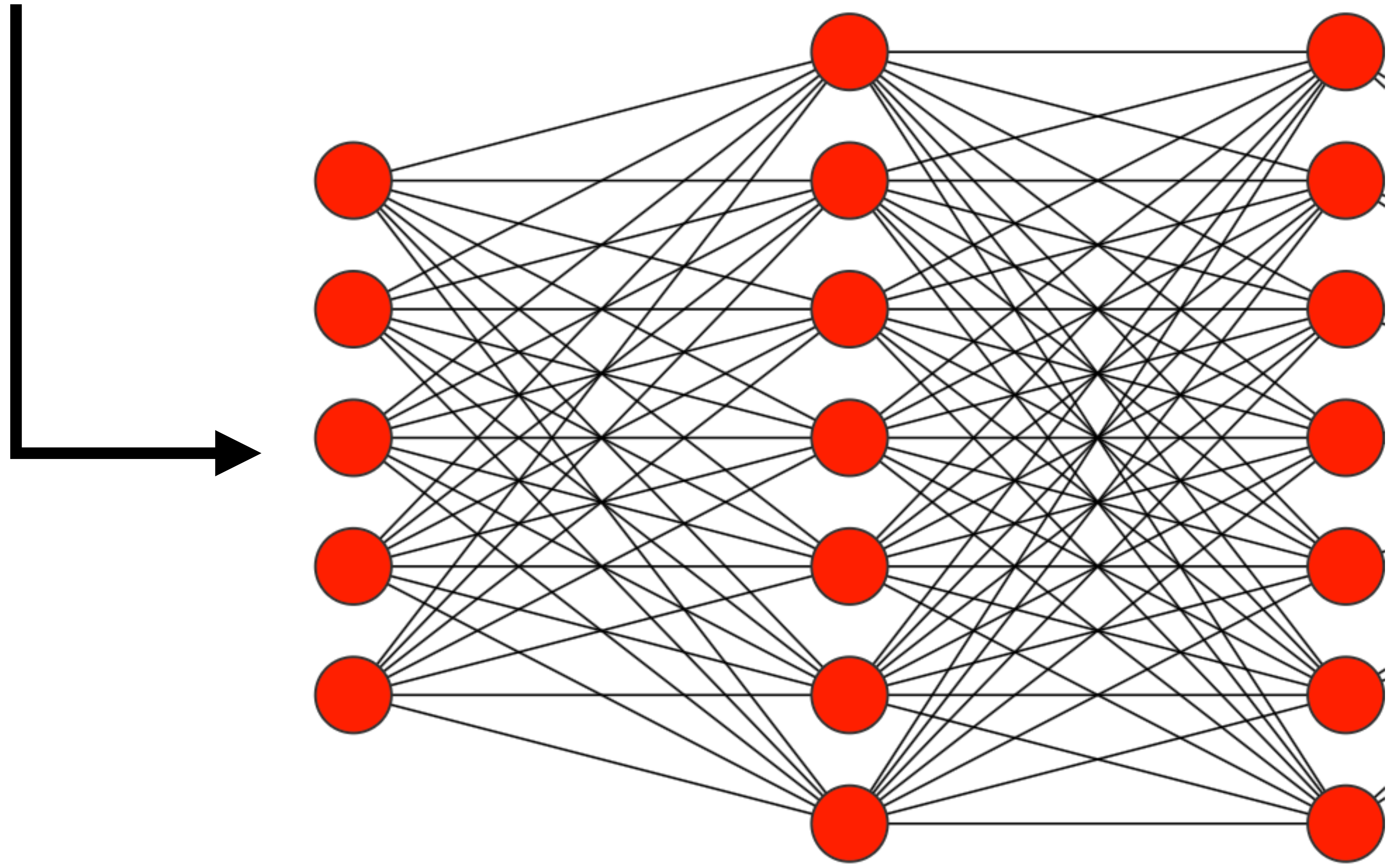
Make predictions

$$\theta(t_{pred}) = \theta_0 + \omega \cdot t_{pred}$$

$\theta(t_{pred})$?

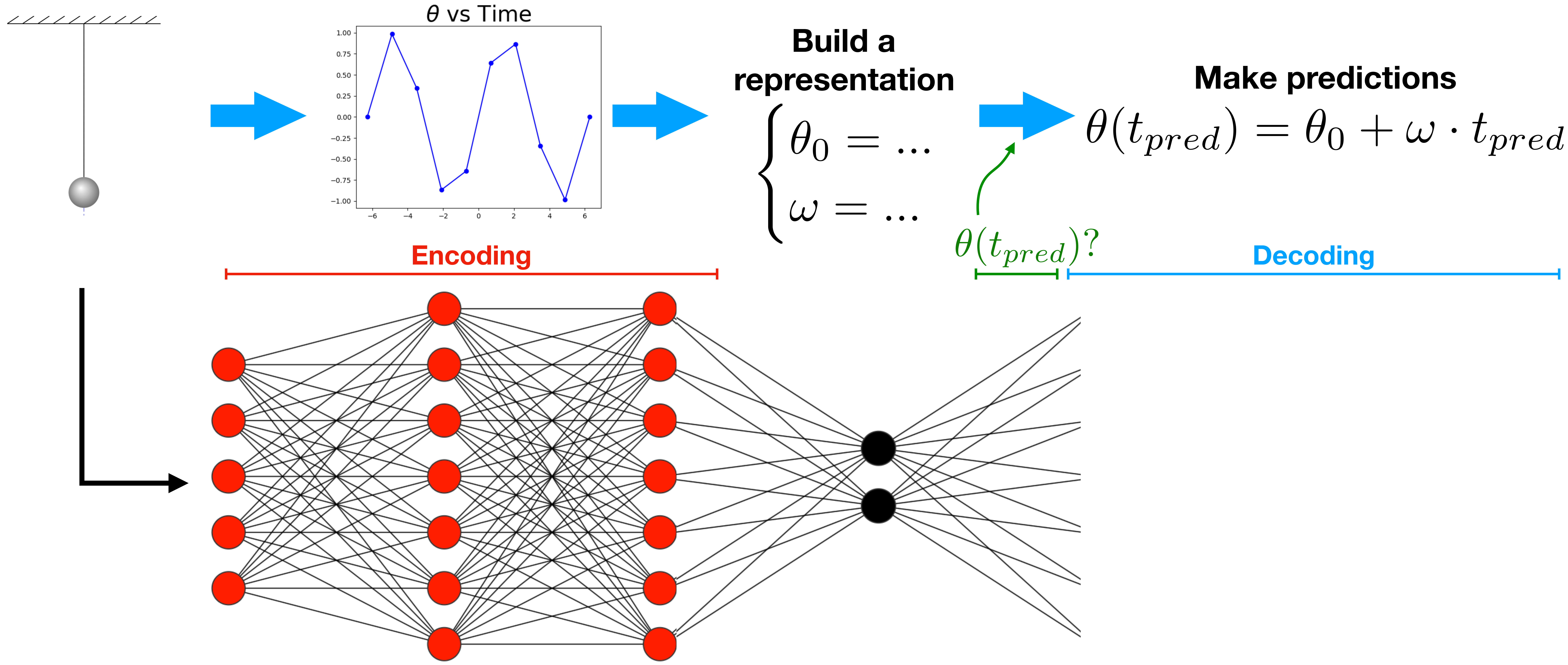
Encoding

Decoding



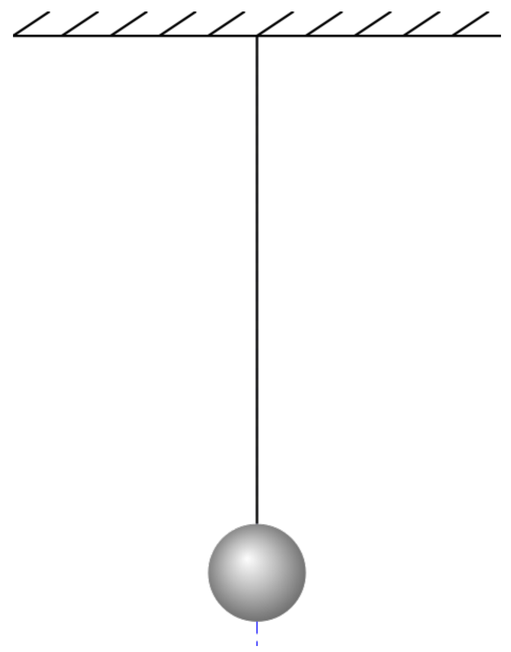
Implementation

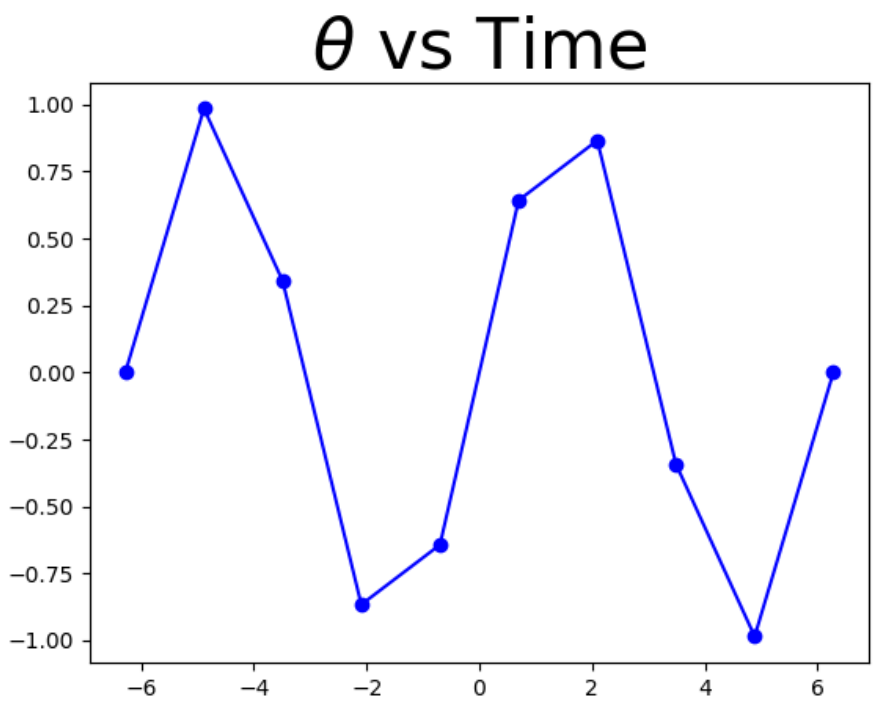
A Physicist is a (very efficient) “Machine”



Implementation

A Physicist is a (very efficient) “Machine”





Build a representation

$$\begin{cases} \theta_0 = \dots \\ \omega = \dots \end{cases}$$

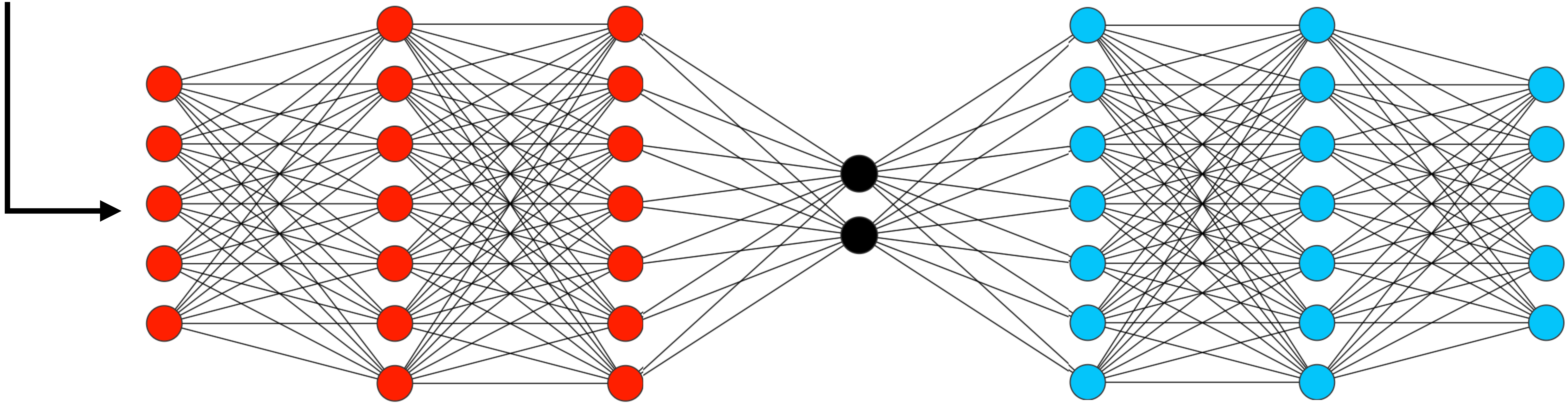
Make predictions

$$\theta(t_{pred}) = \theta_0 + \omega \cdot t_{pred}$$

$\theta(t_{pred})$?

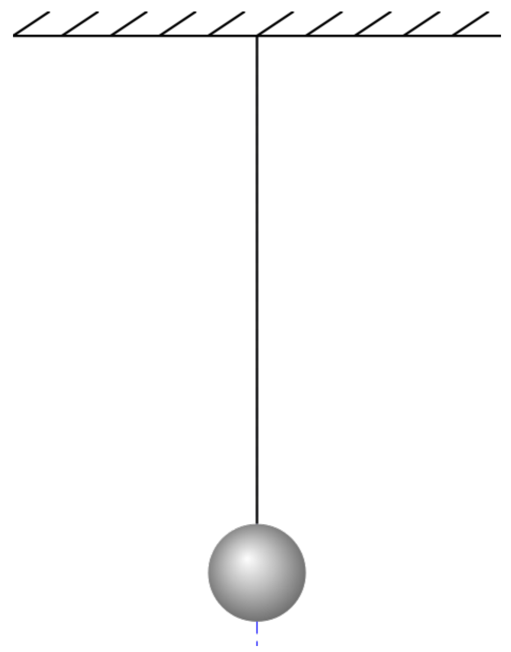
Encoding

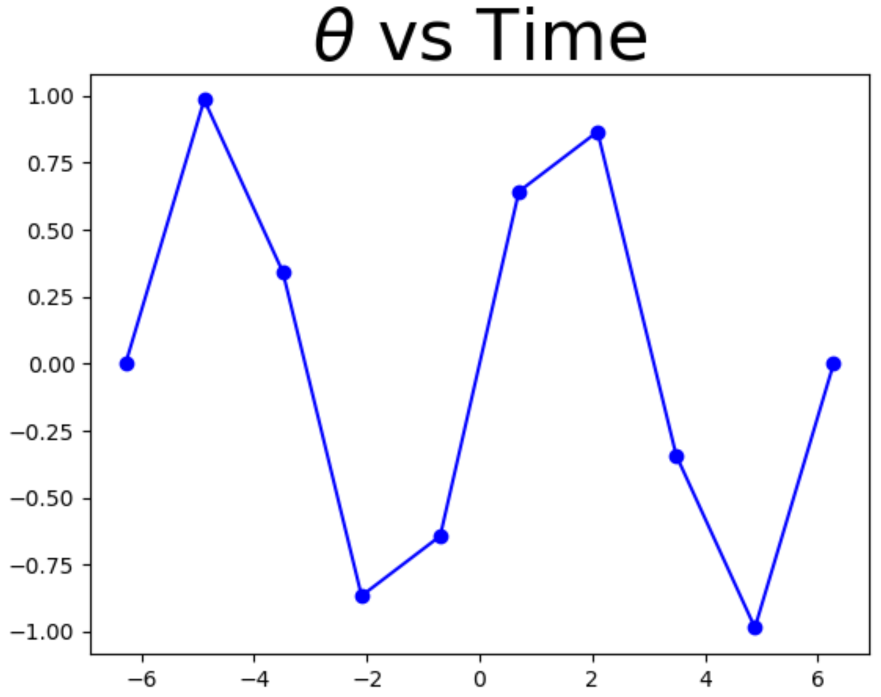
Decoding



Implementation

A Physicist is a (very efficient) “Machine”



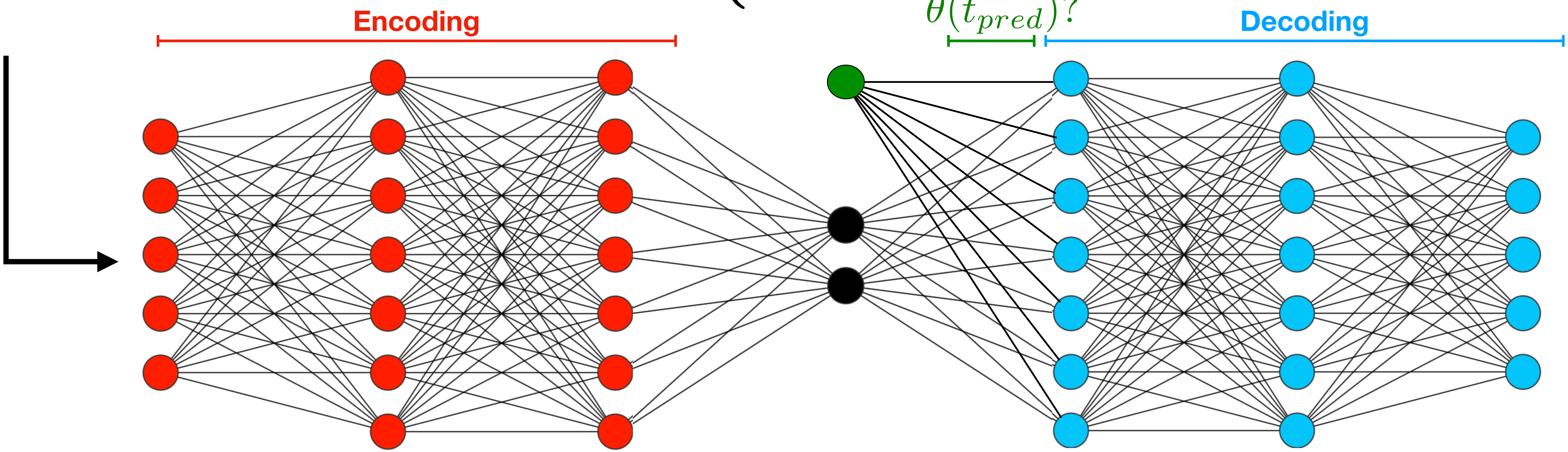


Build a representation

$$\begin{cases} \theta_0 = \dots \\ \omega = \dots \end{cases}$$

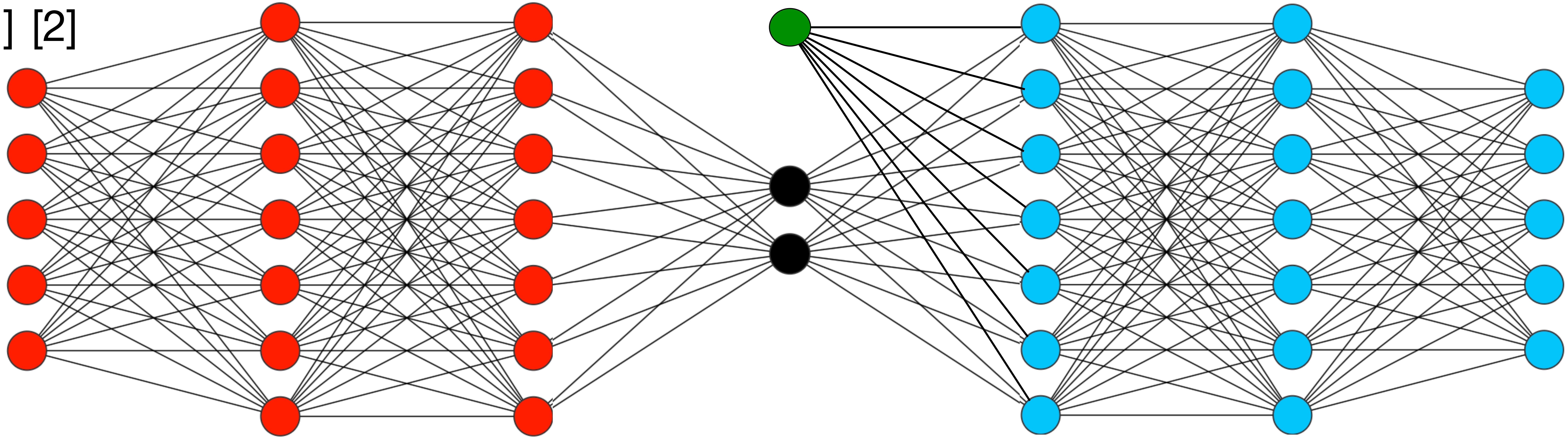
Make predictions

$$\theta(t_{pred}) = \theta_0 + \omega \cdot t_{pred}$$



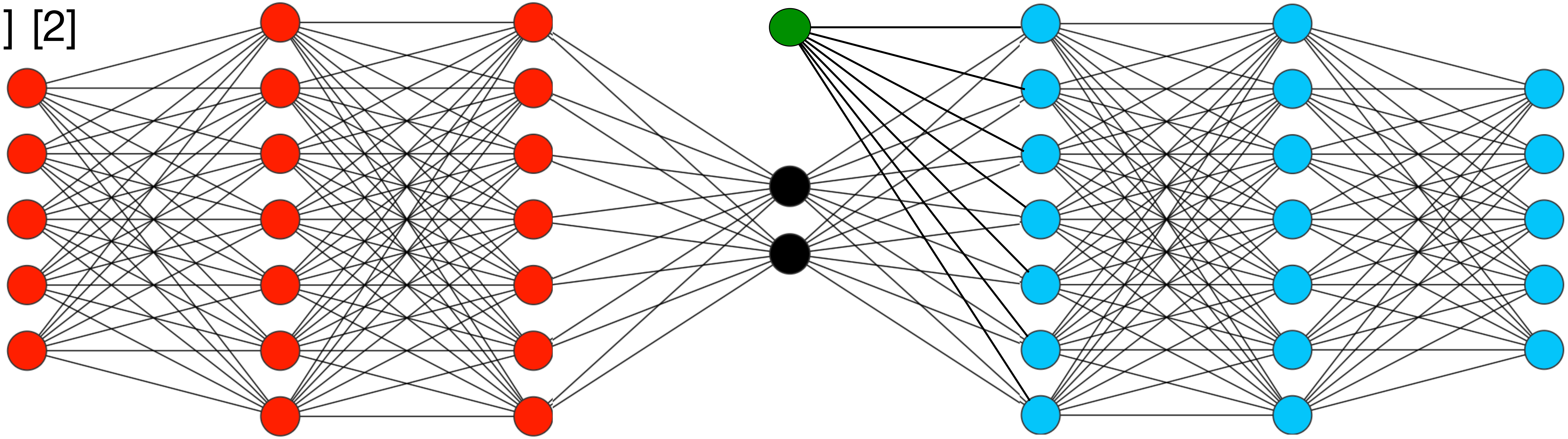
Implementation

What is this? [1] [2]



Implementation

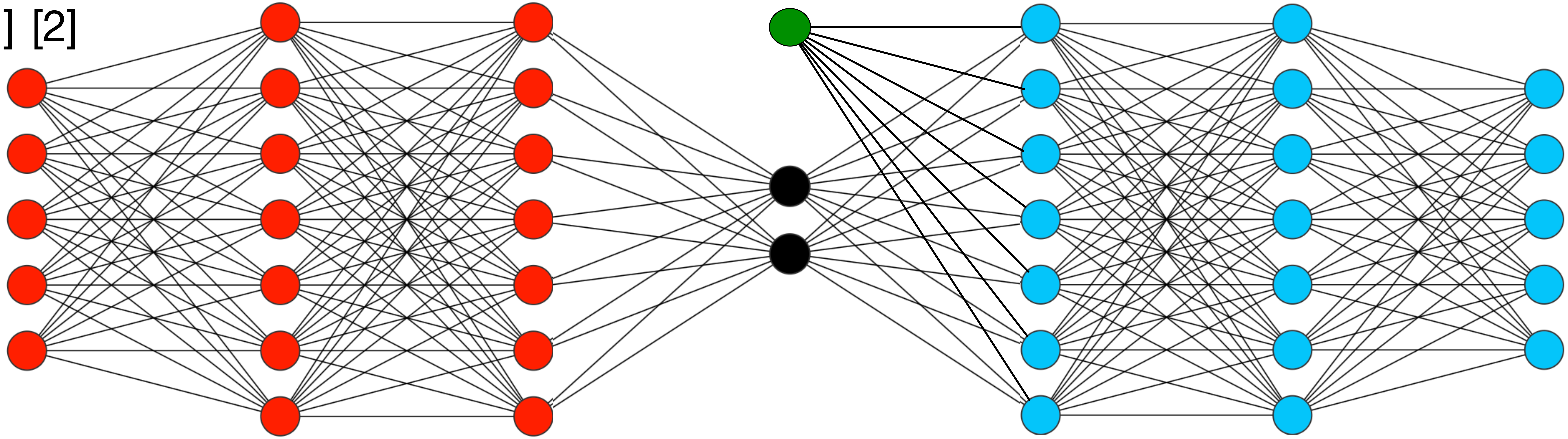
What is this? [1] [2]



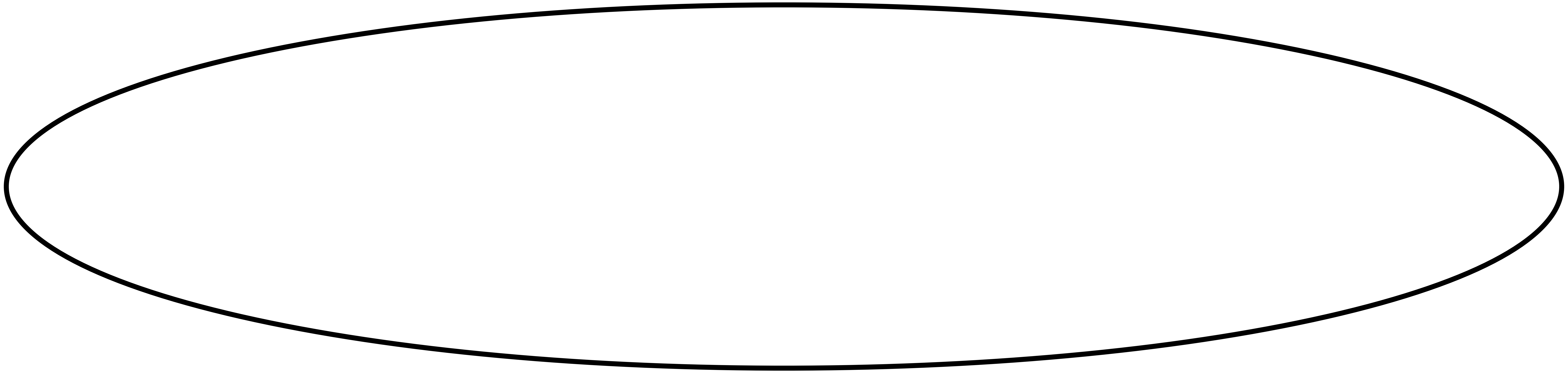
$$SciNet =$$

Implementation

What is this? [1] [2]

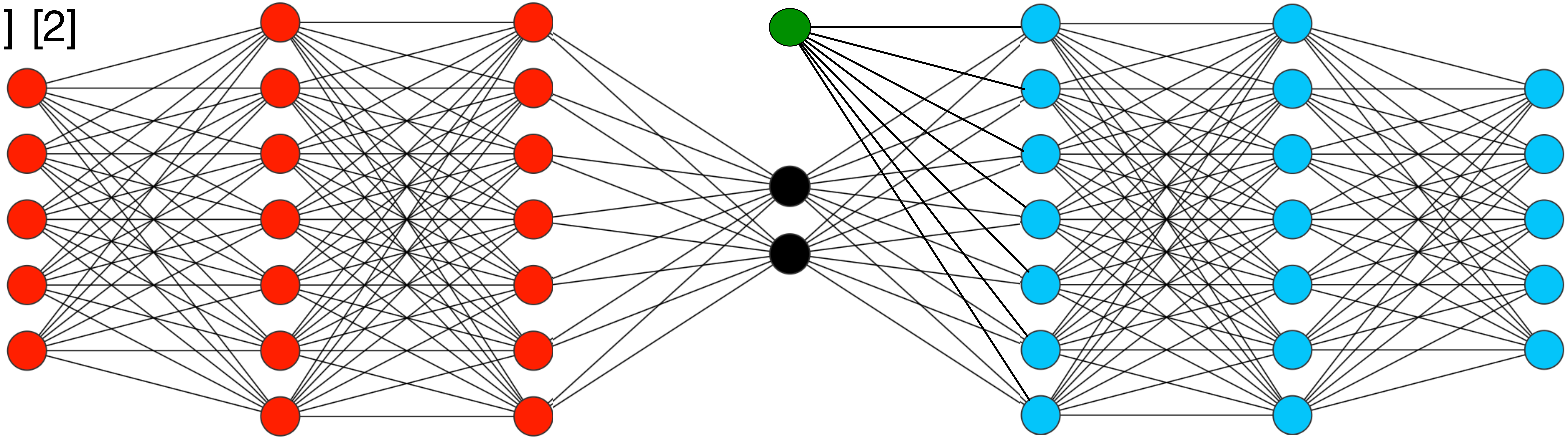


$SciNet =$



Implementation

What is this? [1] [2]

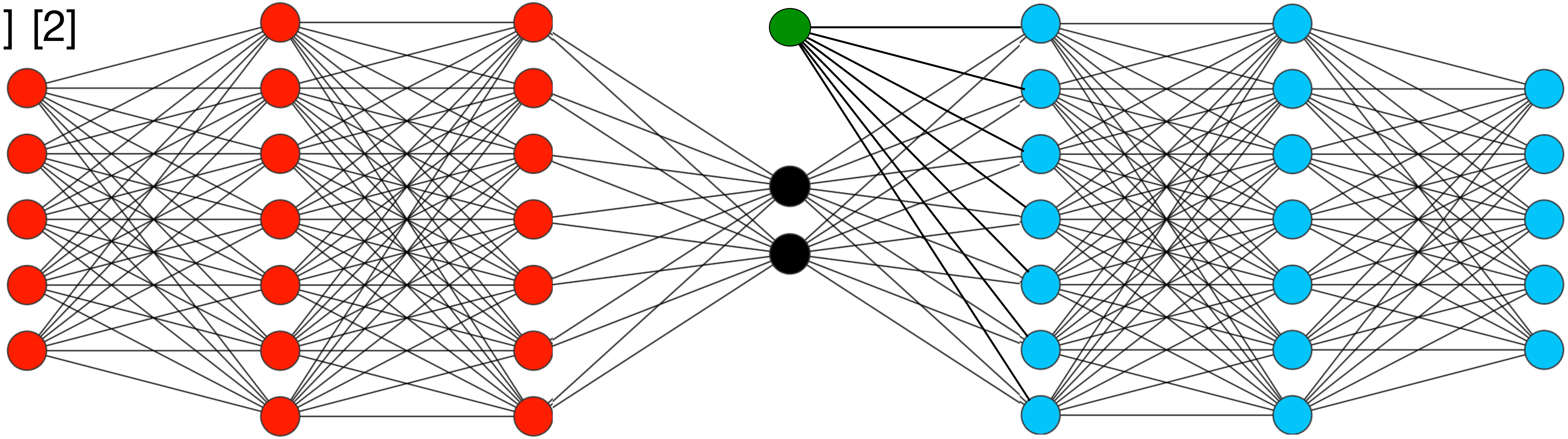


$$SciNet =$$

Machine Learning Algorithm

Implementation

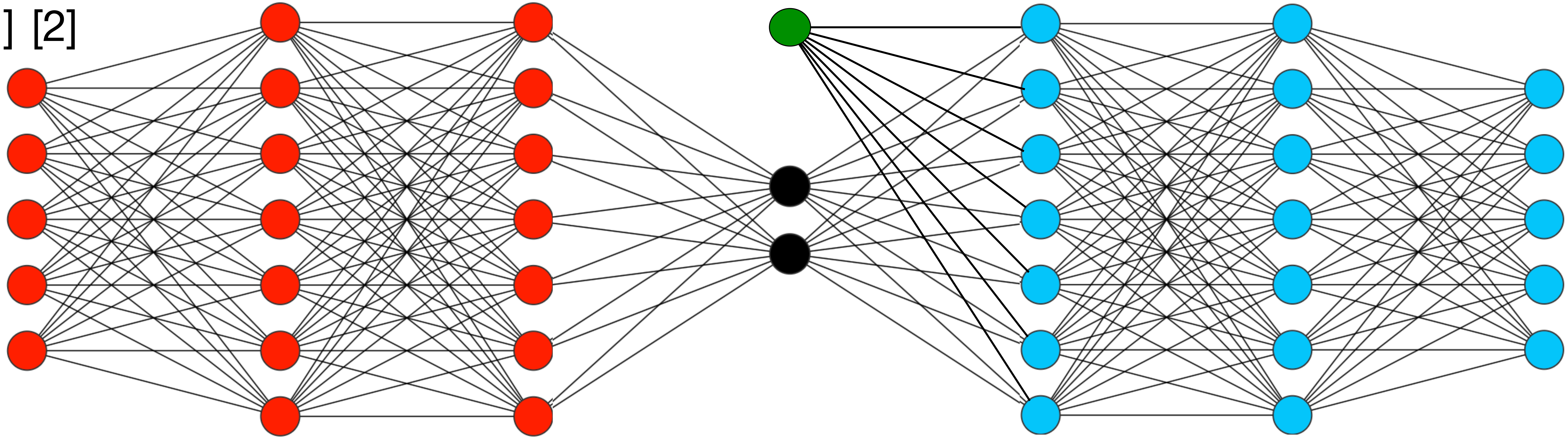
What is this? [1] [2]



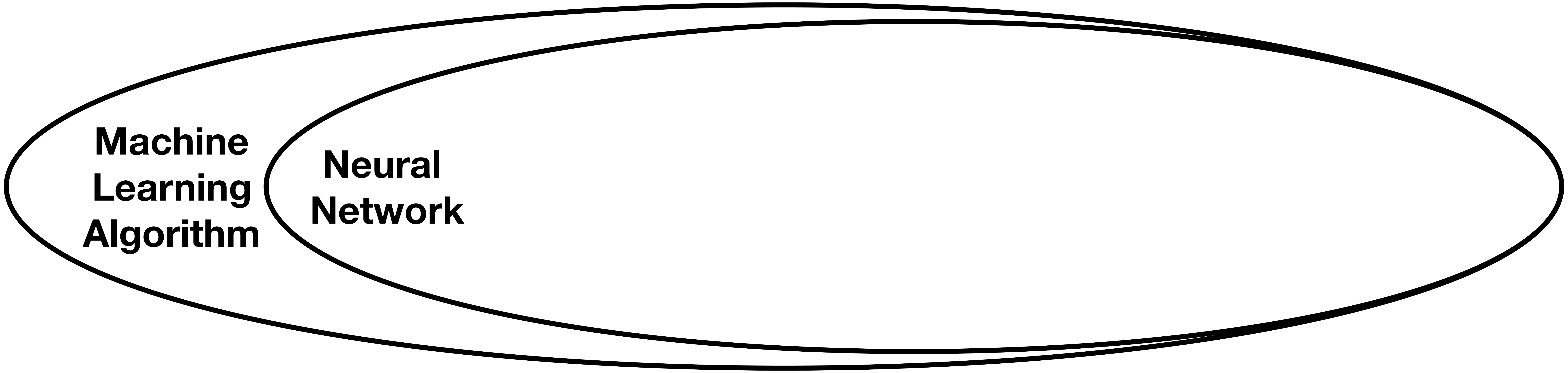
$SciNet =$ **Machine Learning Algorithm**

Implementation

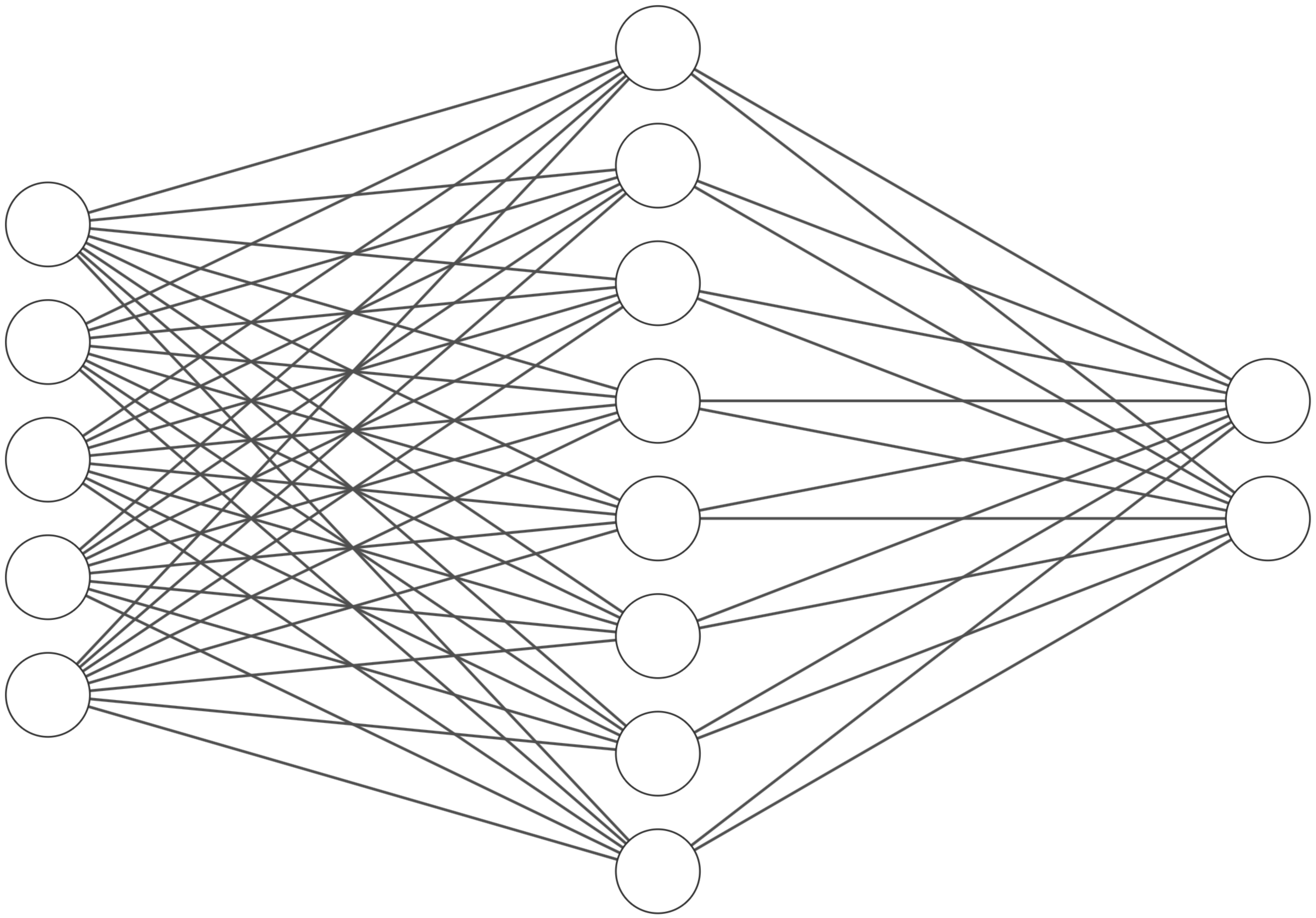
What is this? [1] [2]



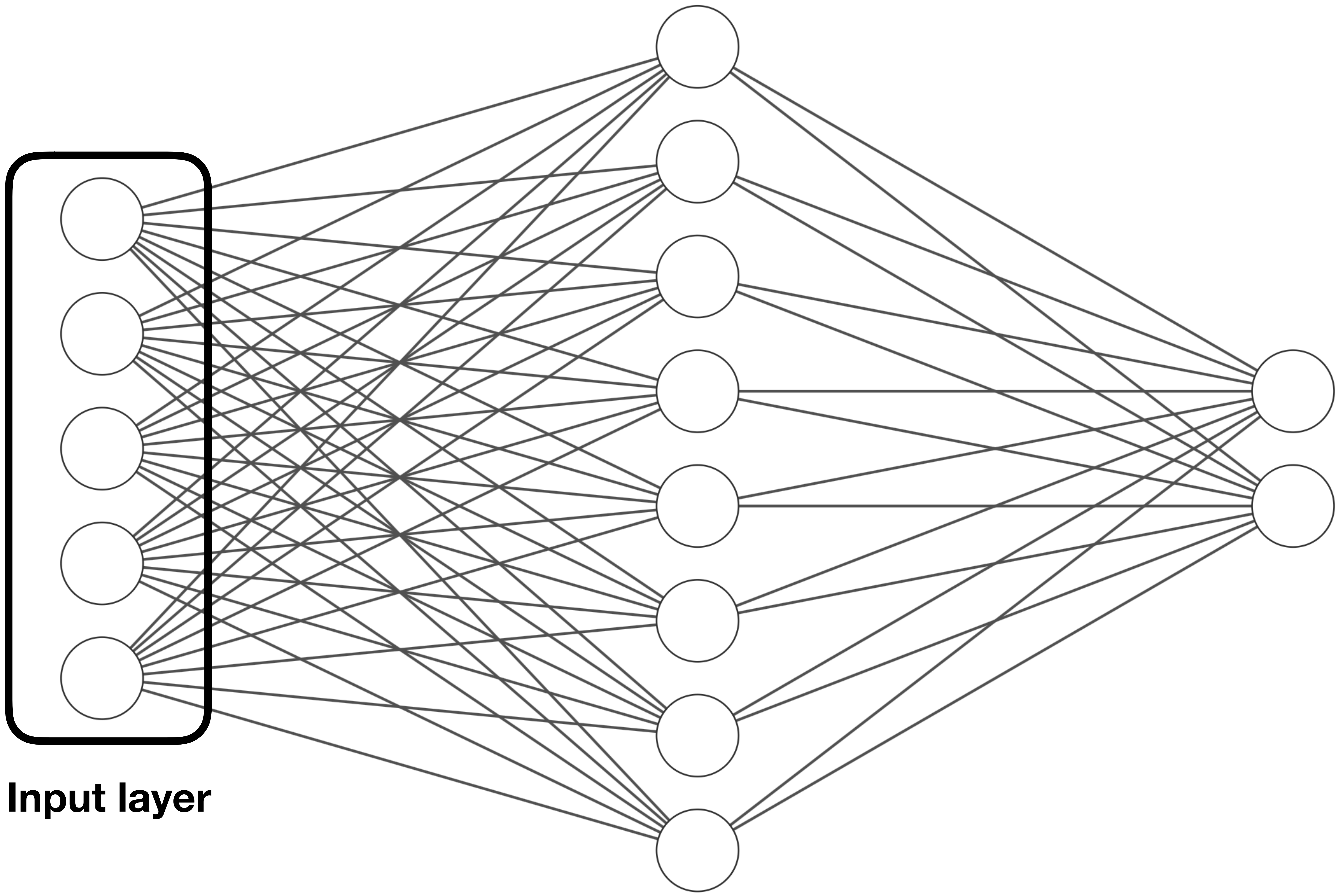
$$SciNet =$$



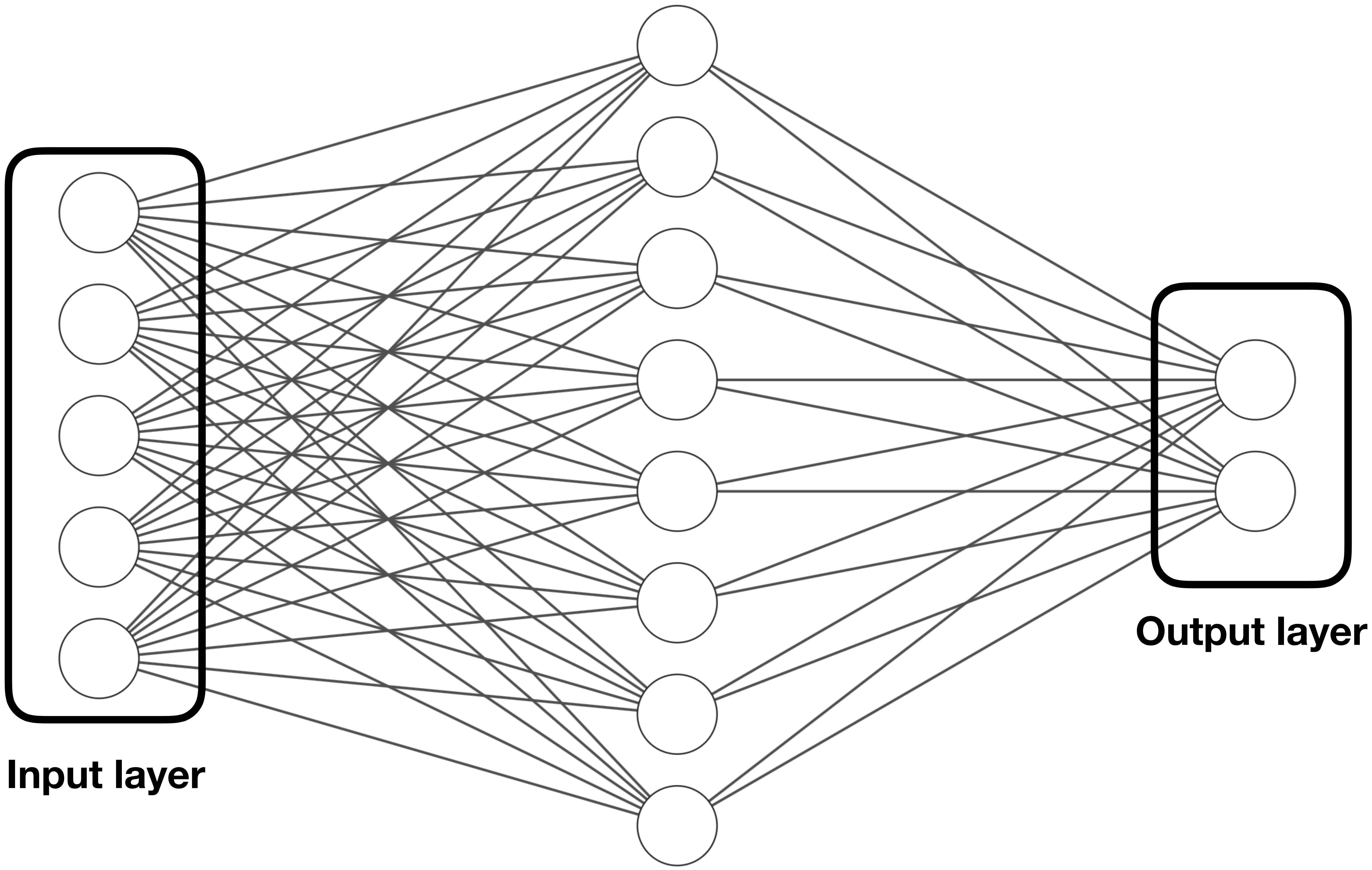
Implementation



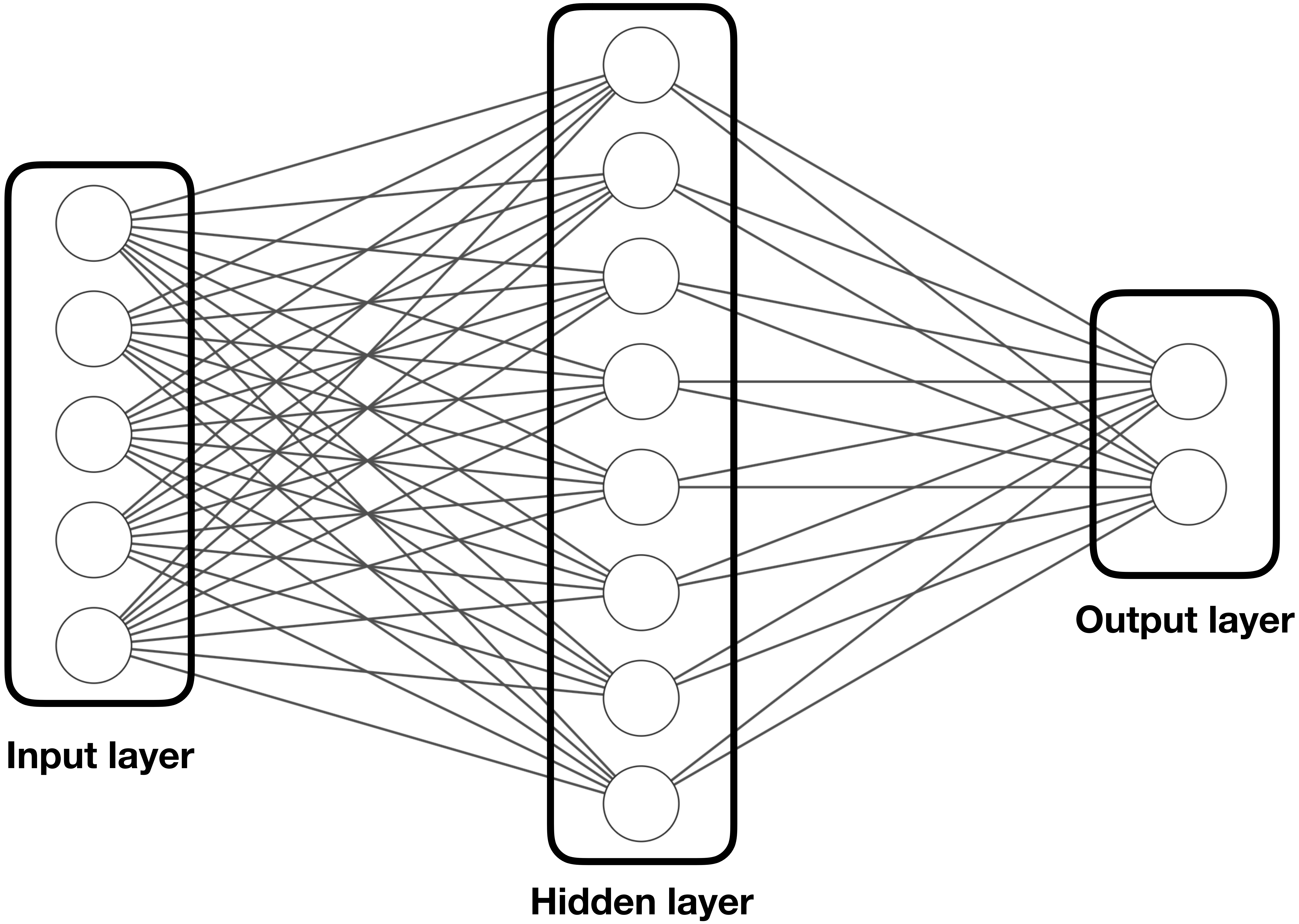
Implementation



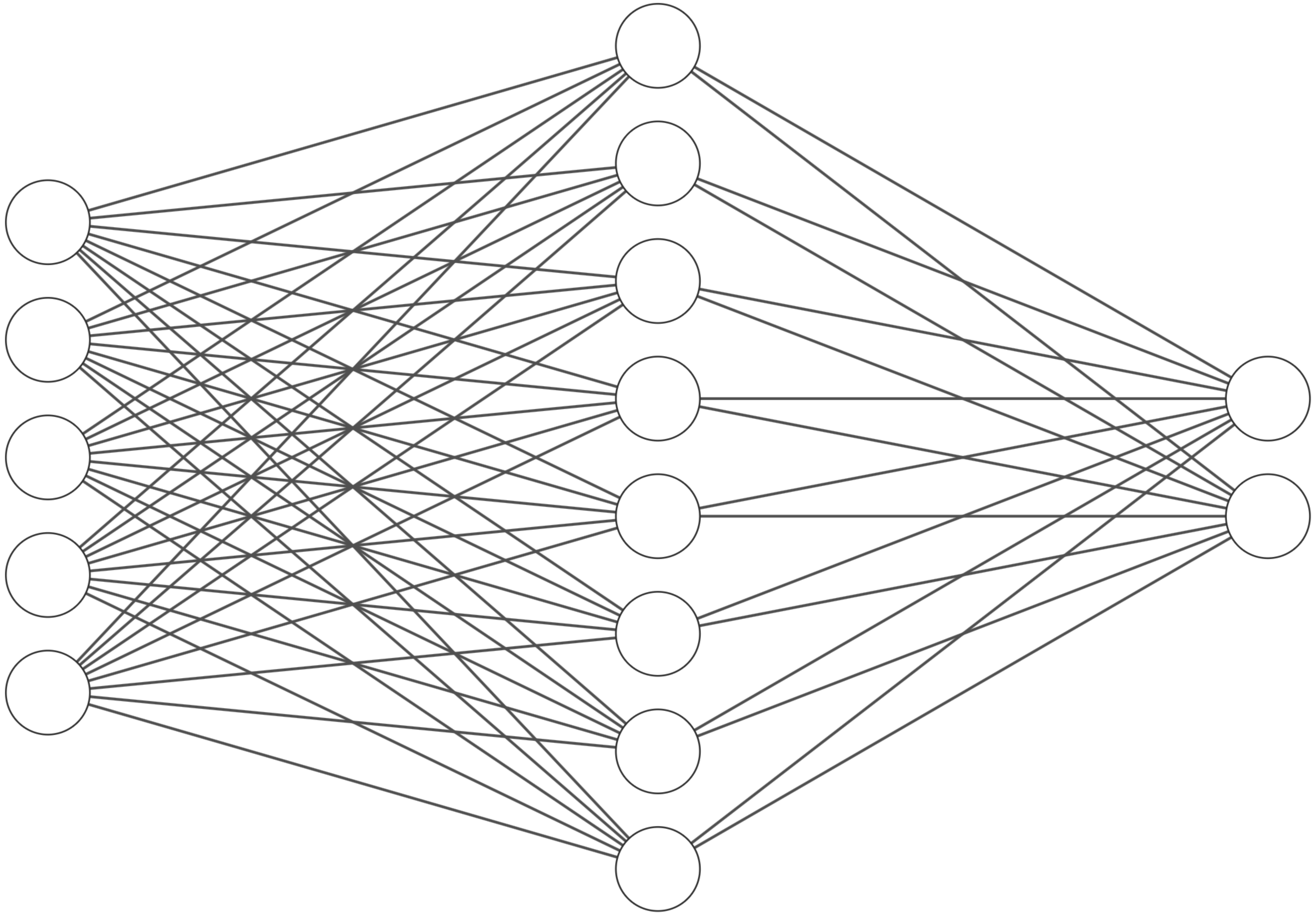
Implementation



Implementation

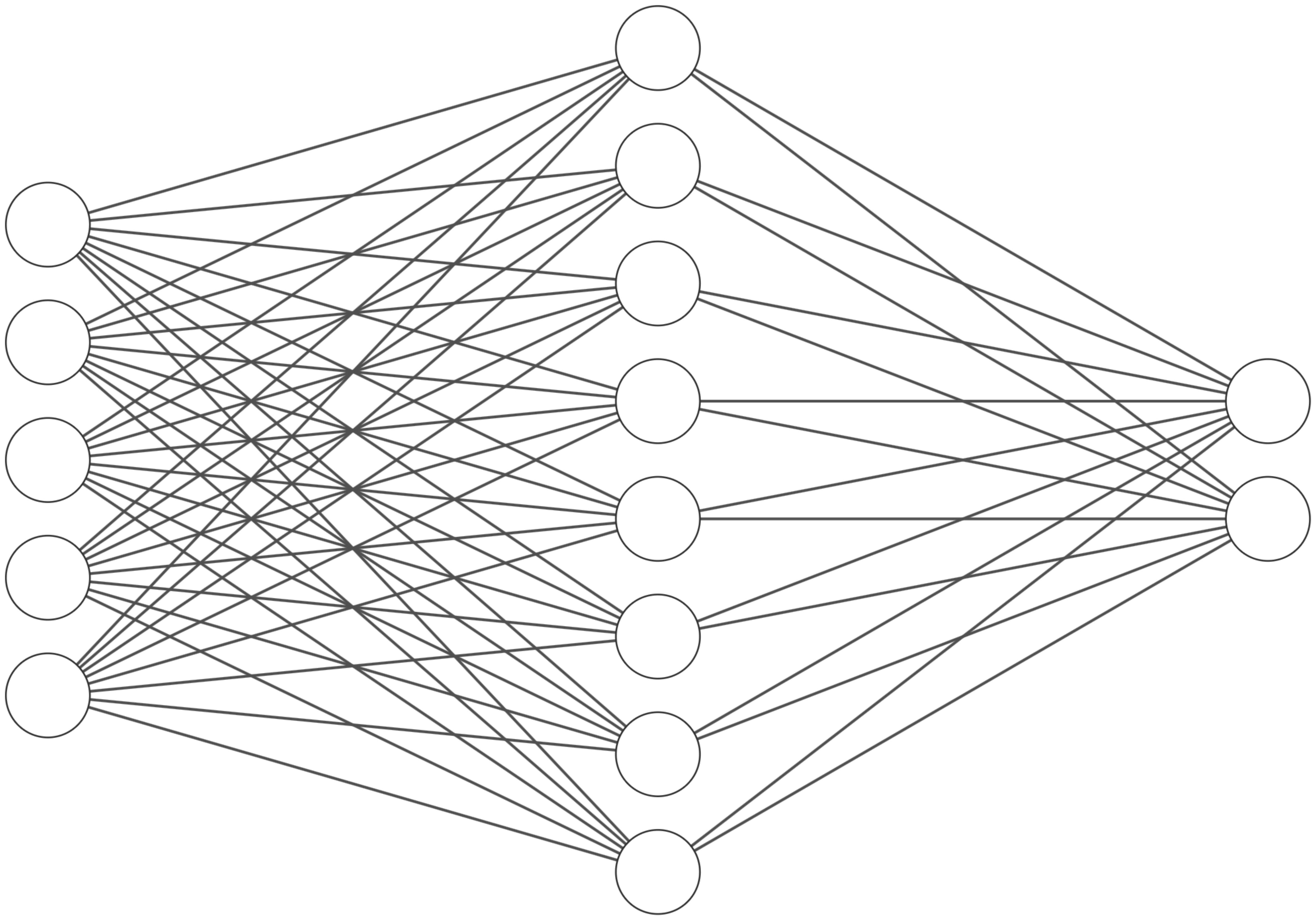


Implementation



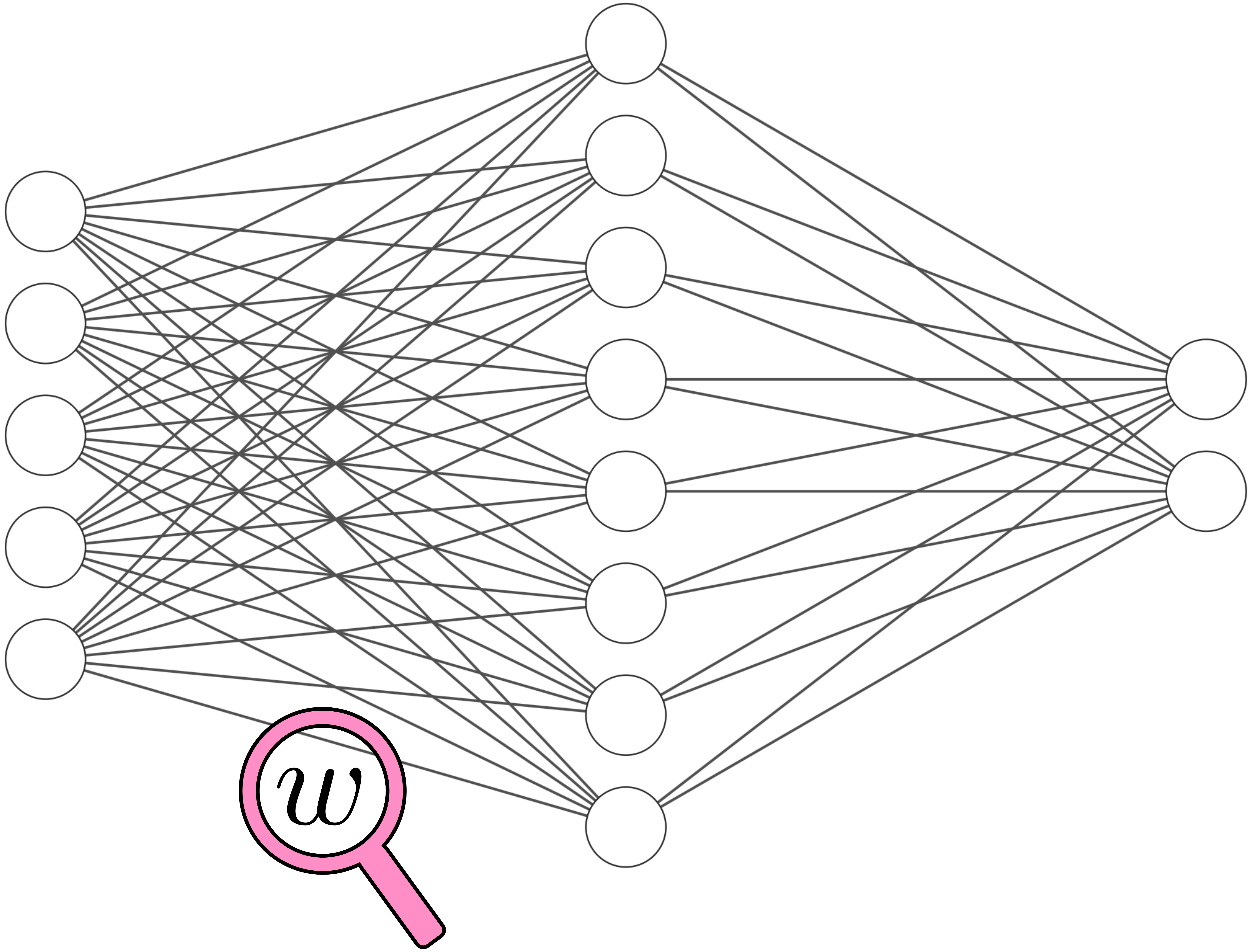
Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$



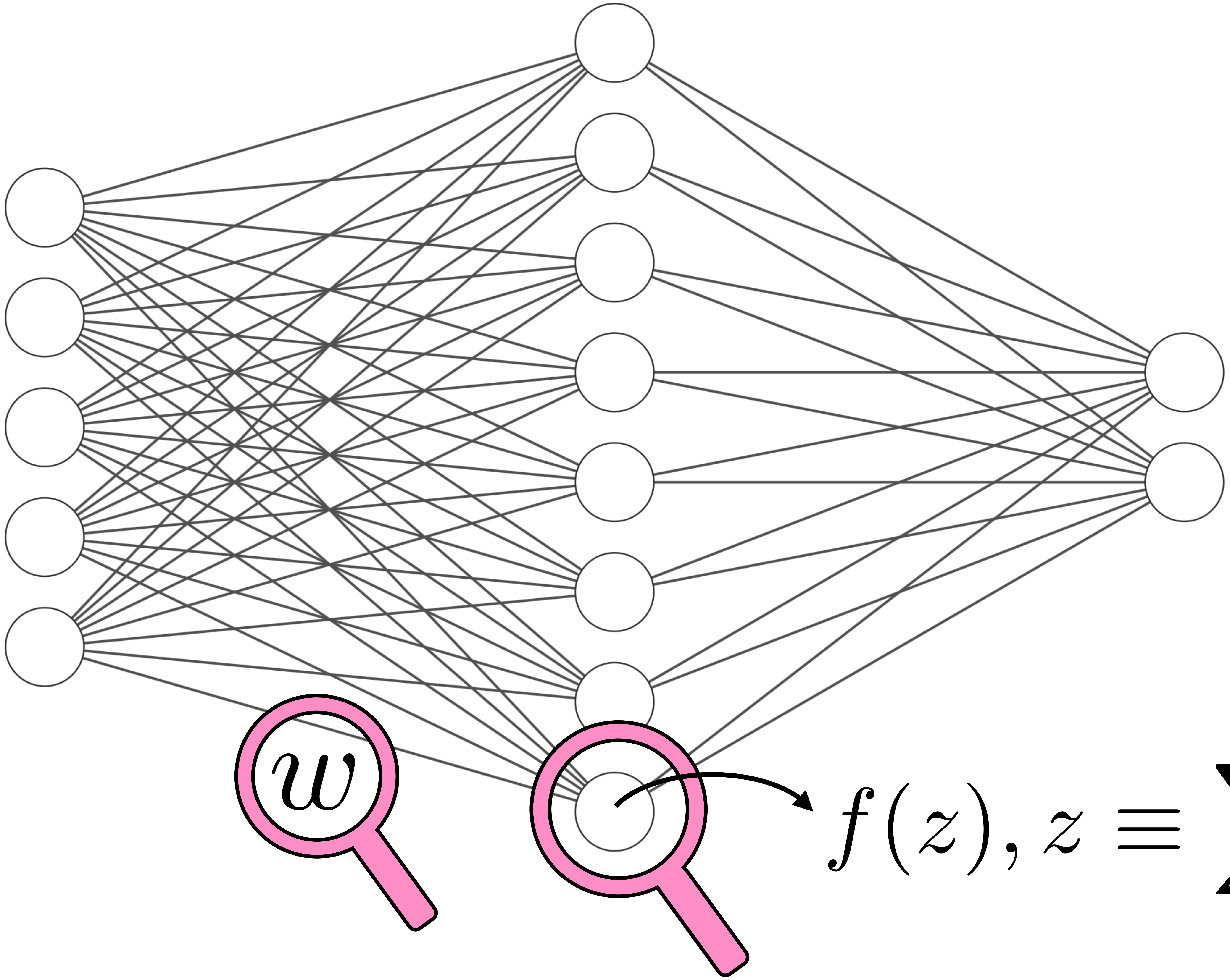
Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$



Implementation

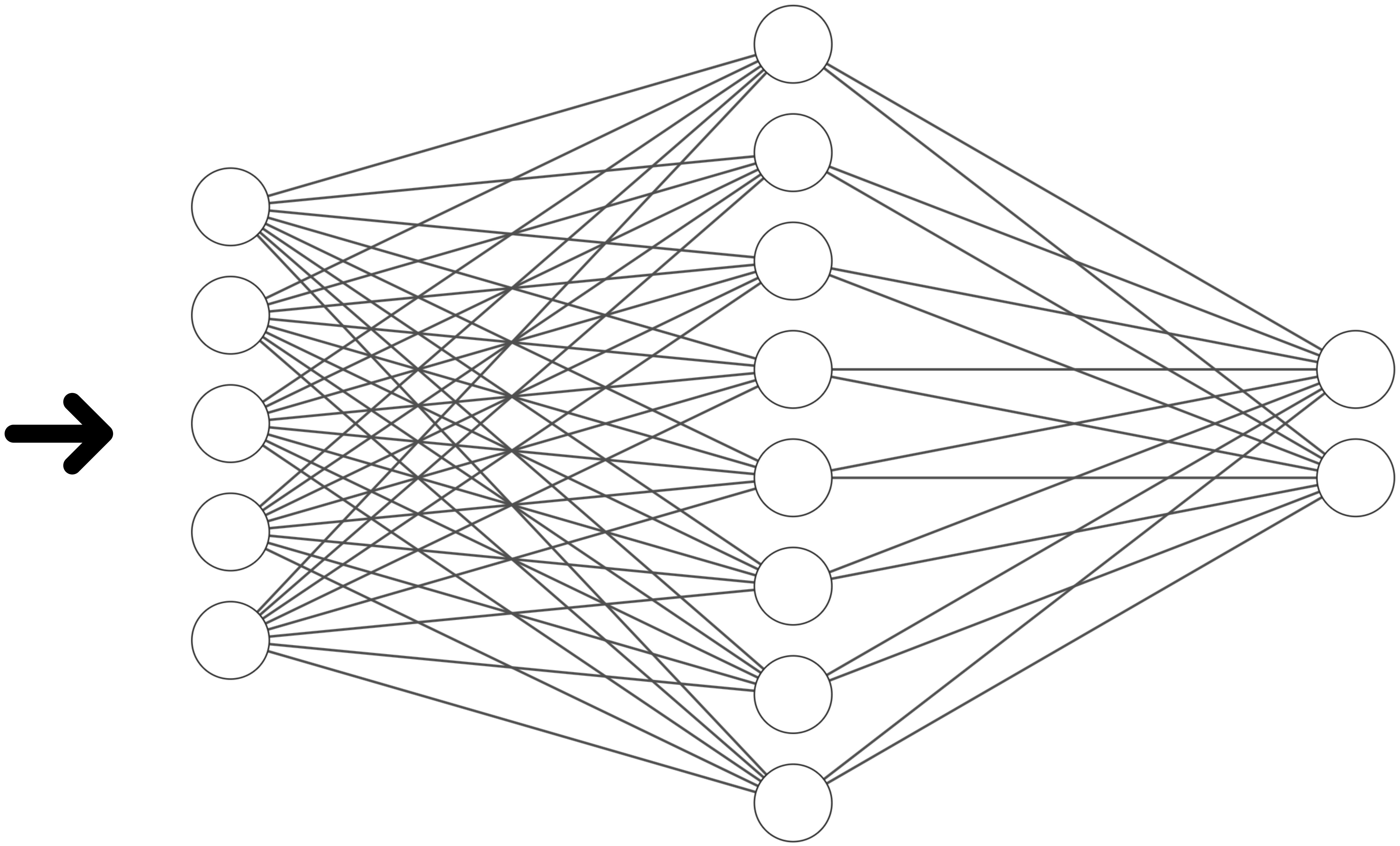
$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$



$$f(z), z \equiv \sum_i w_i x_i + b$$

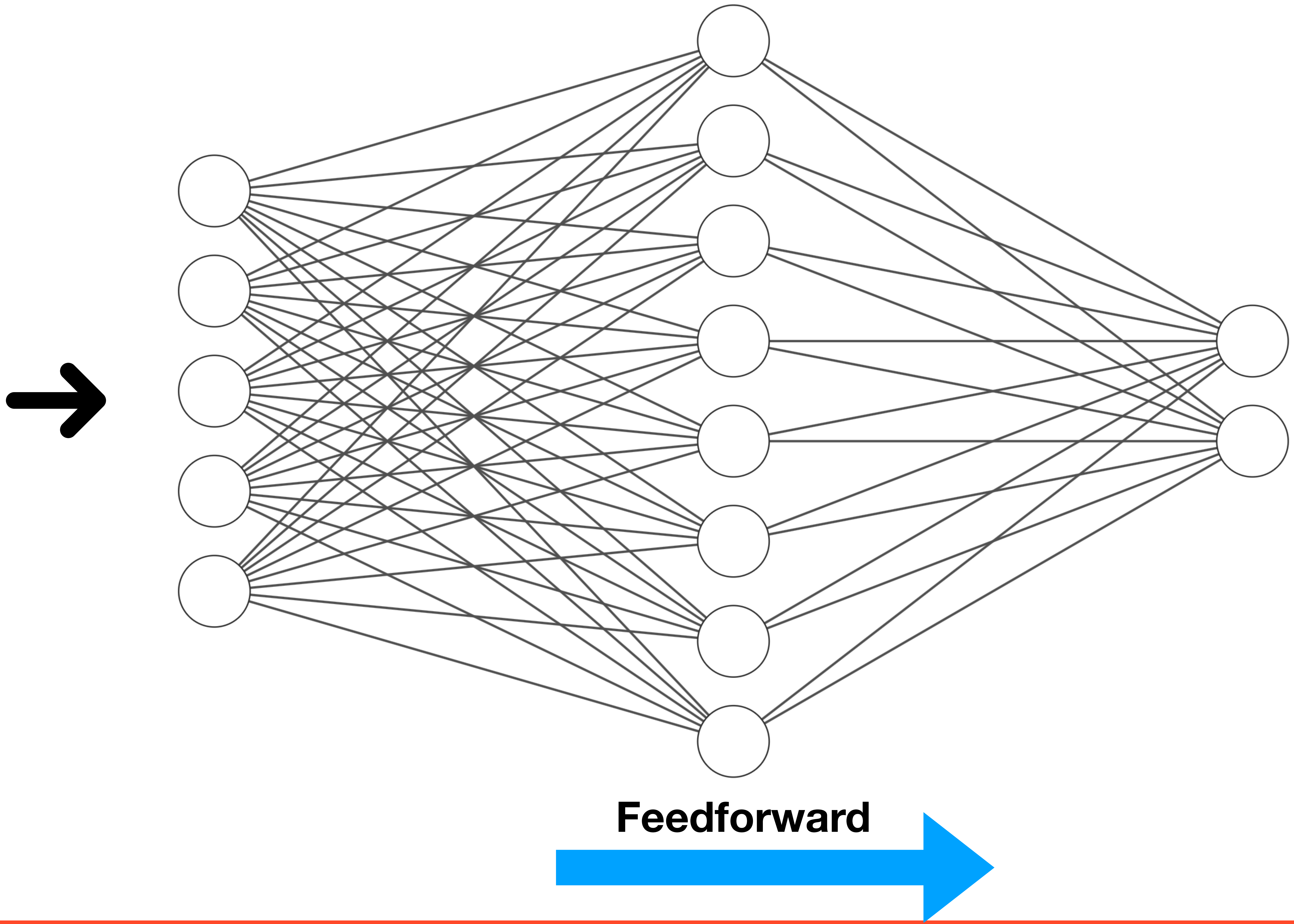
Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$$



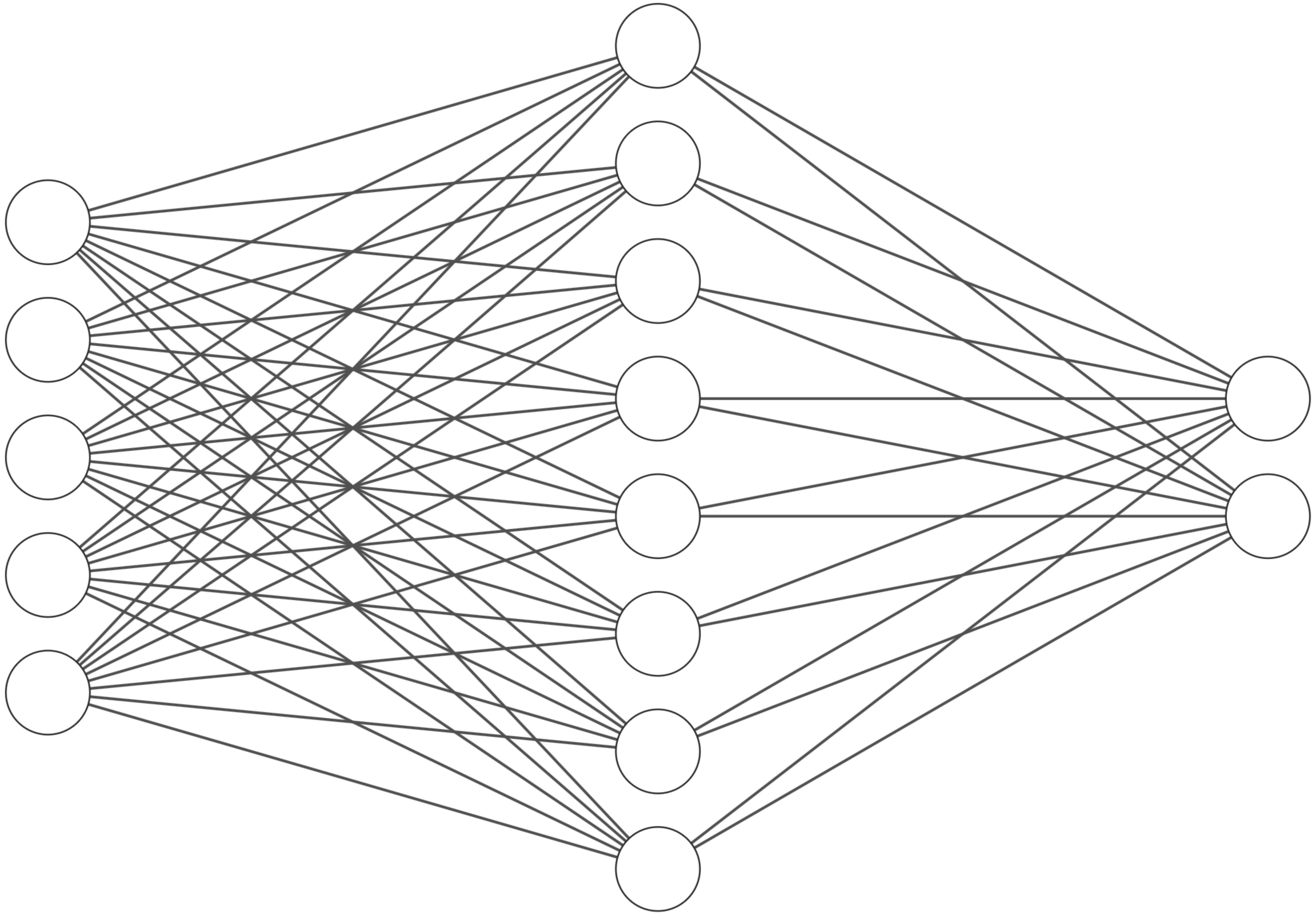
Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$$



Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$

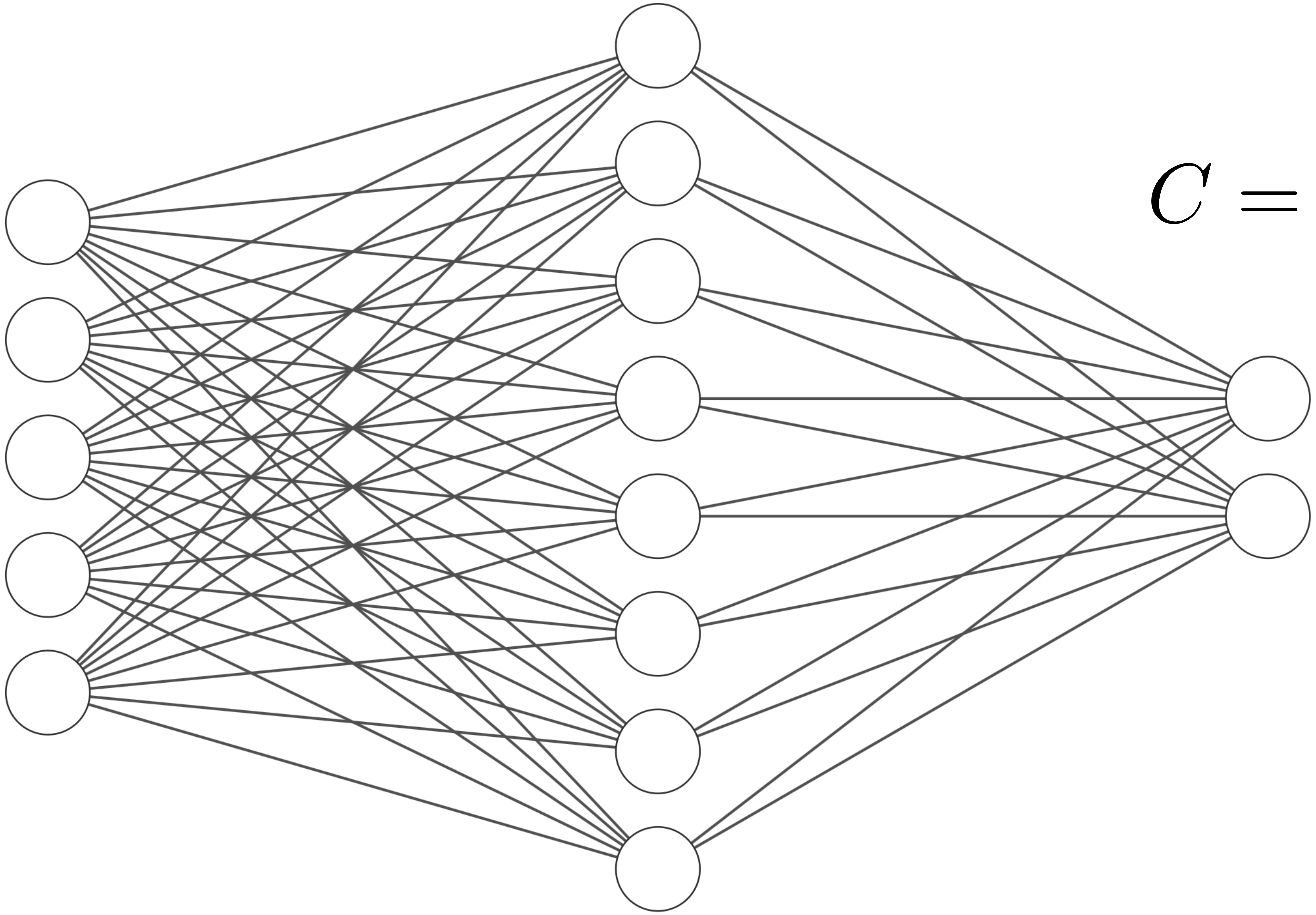


$$\begin{bmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_m(x_1, \ldots, x_n) \end{bmatrix} \in \mathbb{R}^m$$

Feedforward →

Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$



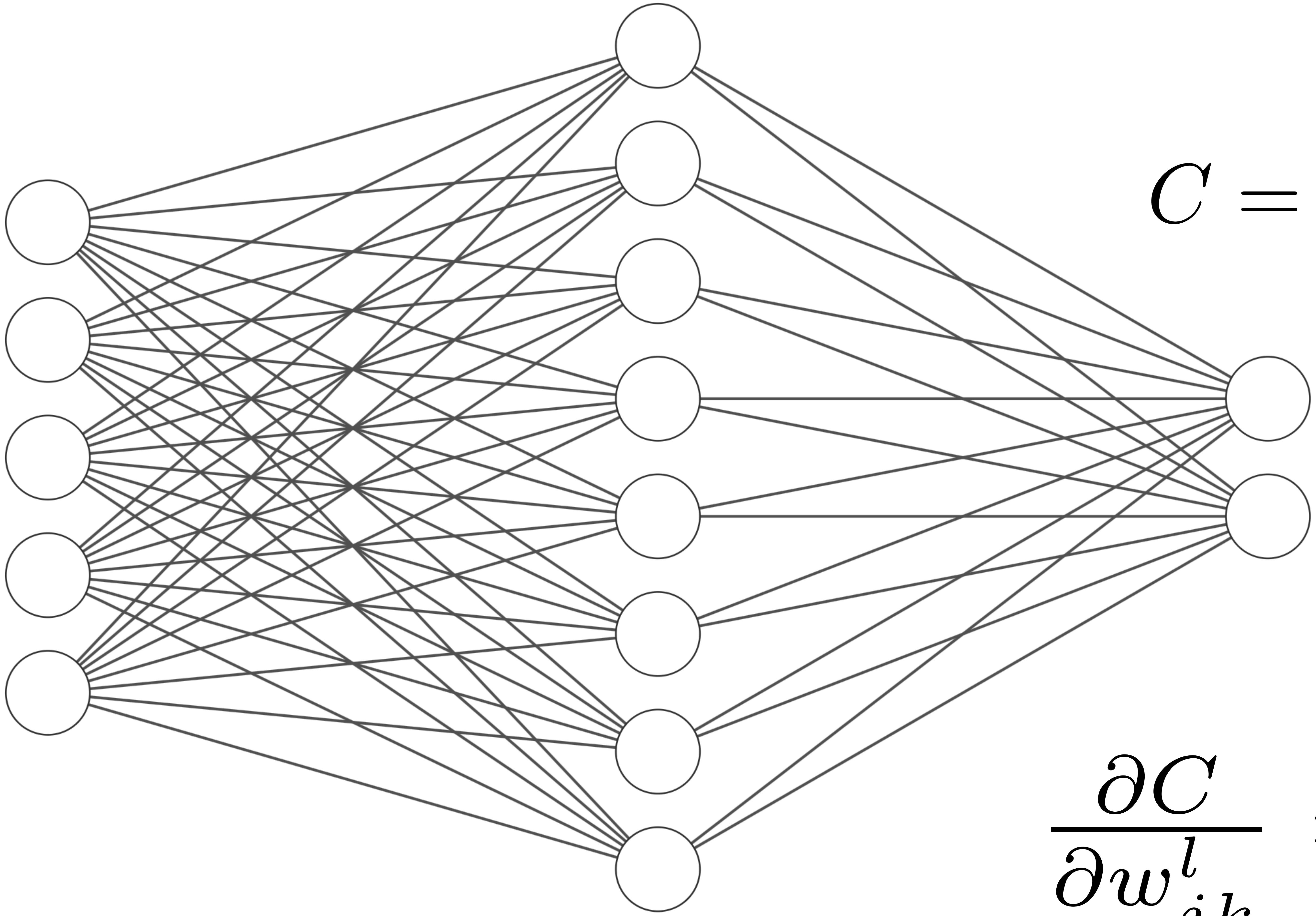
$$C = ||\vec{a} - \vec{f}(\vec{x})||^2$$

$$\begin{bmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_m(x_1, \ldots, x_n) \end{bmatrix} \in \mathbb{R}^m$$

Feedforward

Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$



$$C = ||\vec{a} - \vec{f}(\vec{x})||^2$$

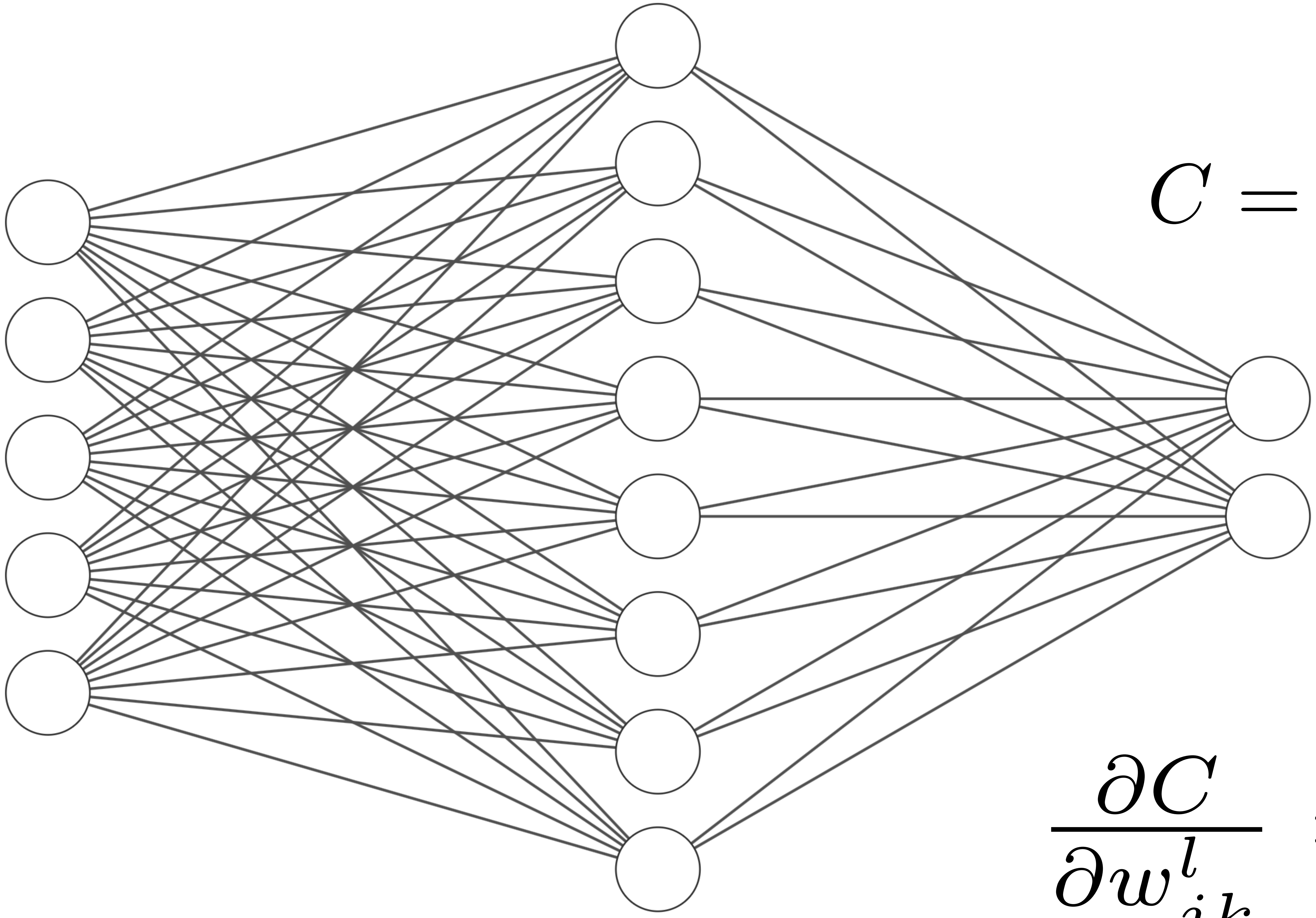
$$\begin{bmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_m(x_1, \ldots, x_n) \end{bmatrix} \in \mathbb{R}^m$$

$$\frac{\partial C}{\partial w^l_{jk}} = \frac{\partial C}{\partial a_i} \frac{\partial a_i}{\partial z_i} \cdots$$

Feedforward →

Implementation

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n \rightarrow$$



$$C = ||\vec{a} - \vec{f}(\vec{x})||^2$$

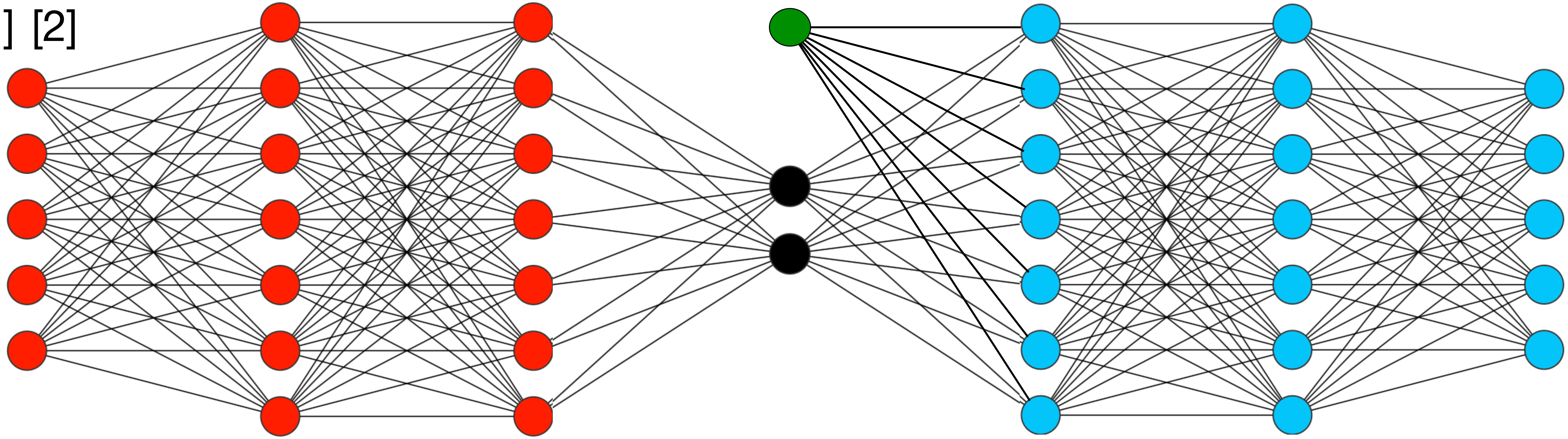
$$\begin{bmatrix} f_1(x_1, \dots, x_n) \\ f_2(x_1, \dots, x_n) \\ \vdots \\ f_m(x_1, \dots, x_n) \end{bmatrix} \in \mathbb{R}^m$$

$$\frac{\partial C}{\partial w^l_{jk}} = \frac{\partial C}{\partial a_i} \frac{\partial a_i}{\partial z_i} \dots$$

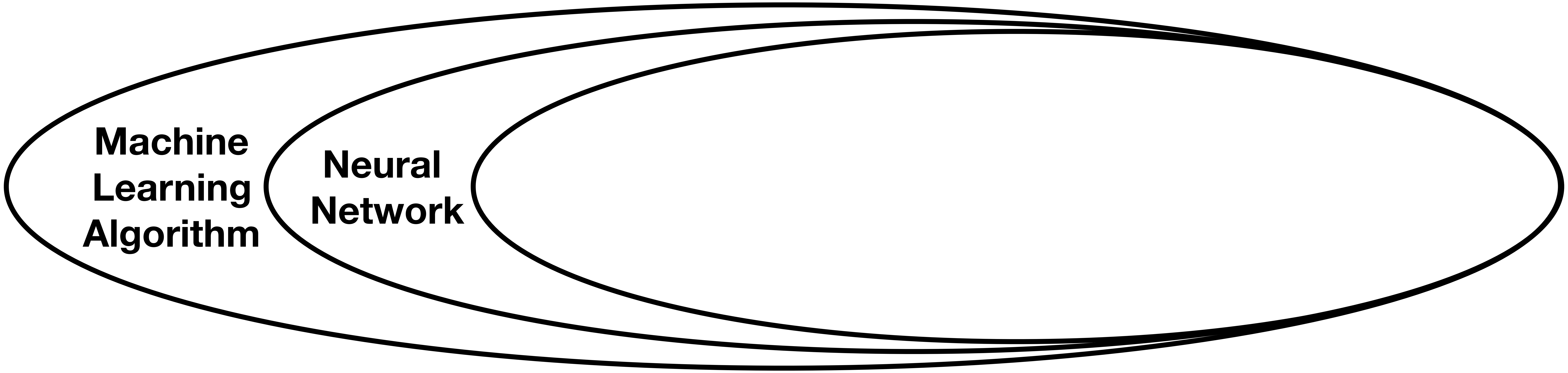
Backpropagation

Implementation

What is this? [1] [2]

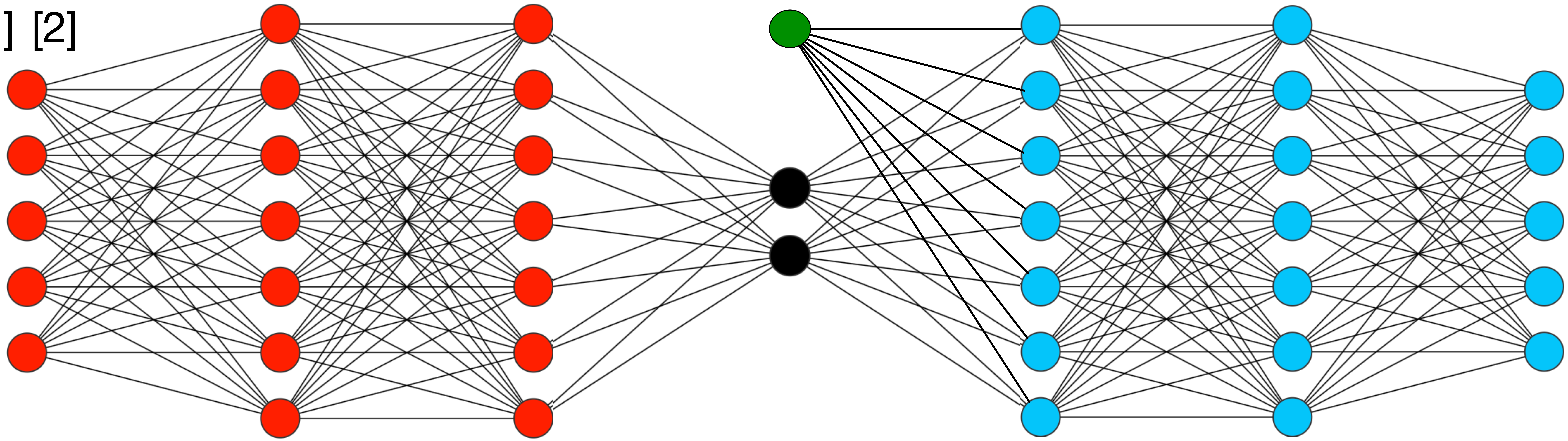


$SciNet =$

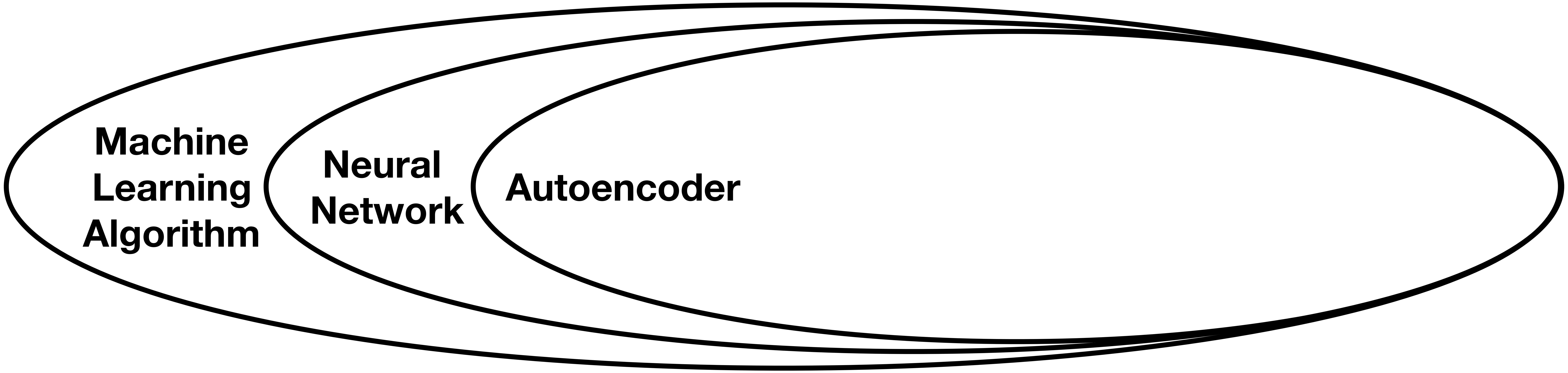


Implementation

What is this? [1] [2]



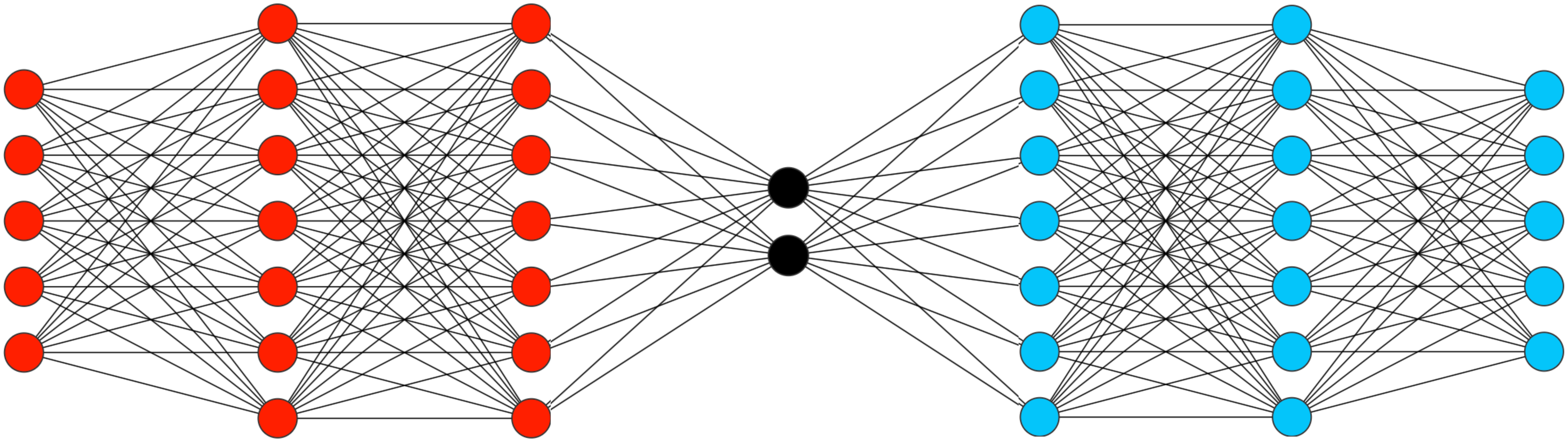
$SciNet =$



Implementation

Main features:

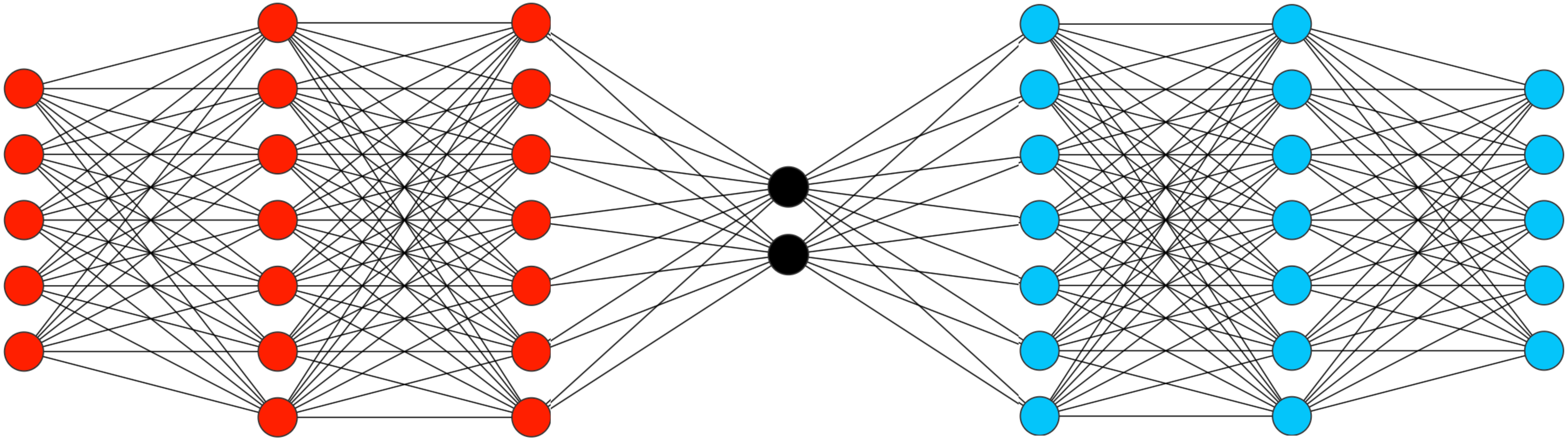
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



Implementation

Main features:

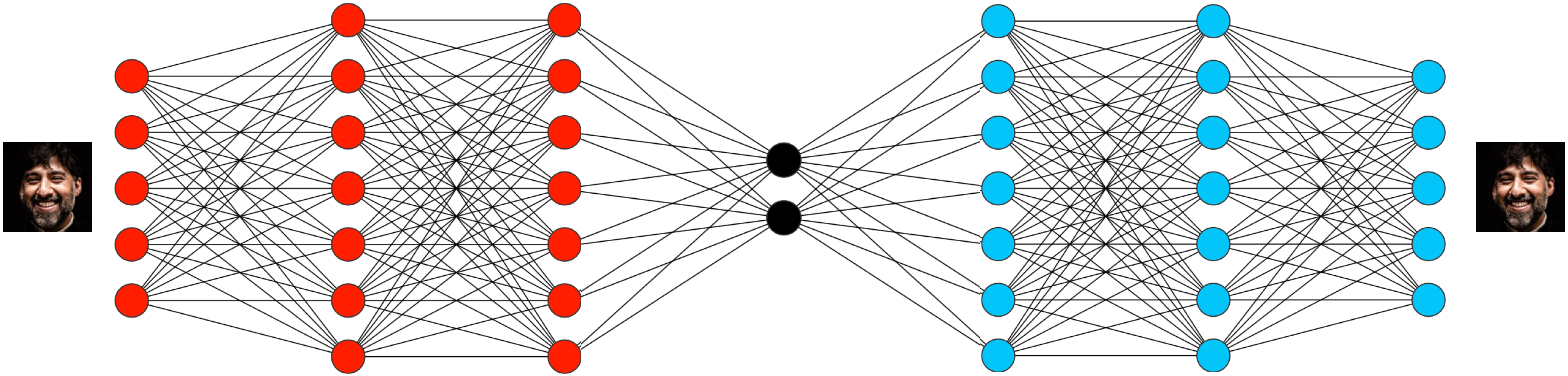
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



Implementation

Main features:

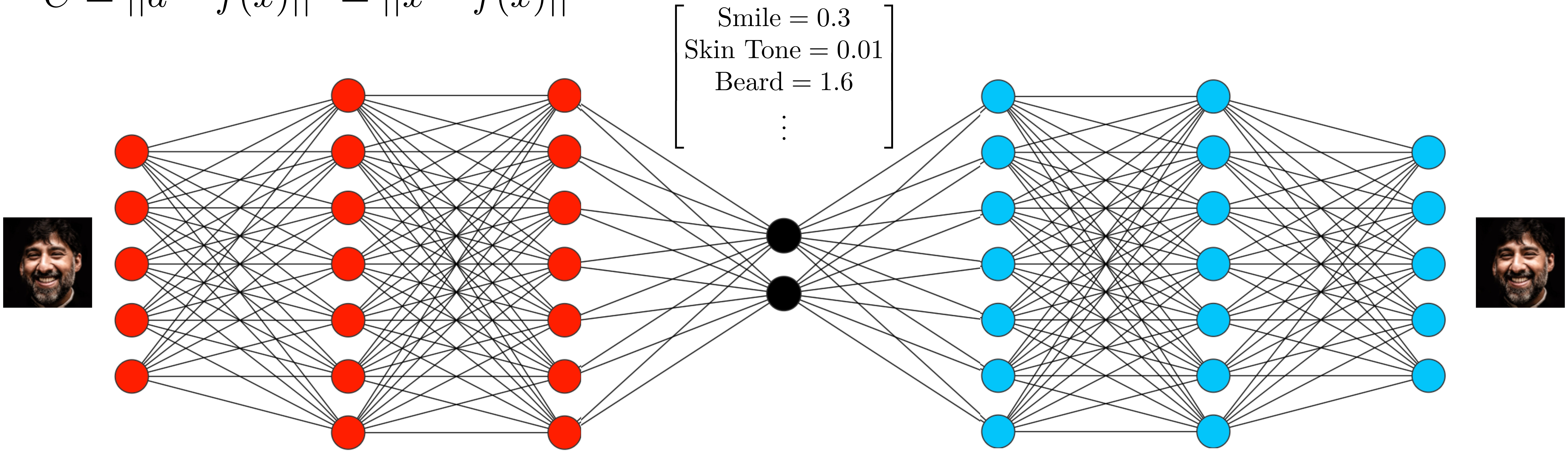
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



Implementation

Main features:

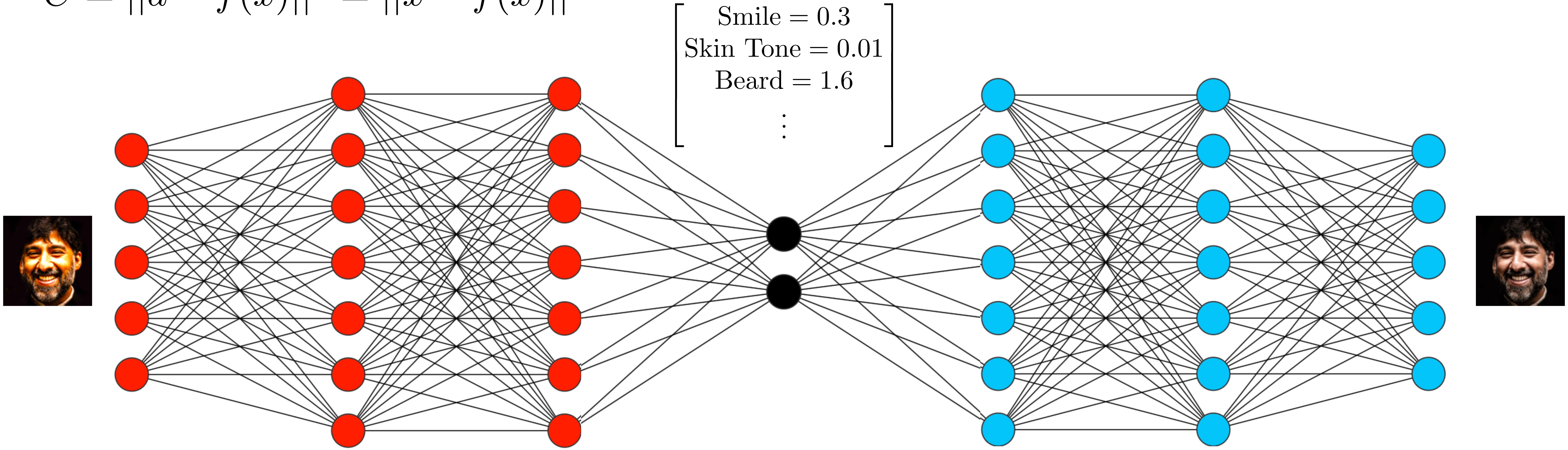
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



Implementation

Main features:

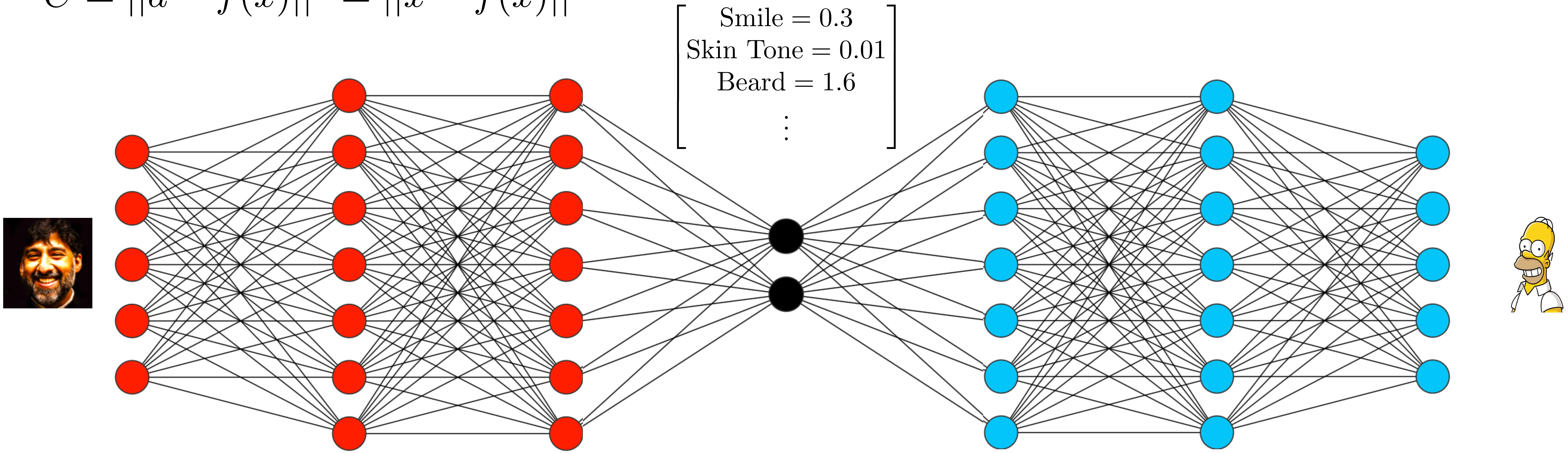
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



Implementation

Main features:

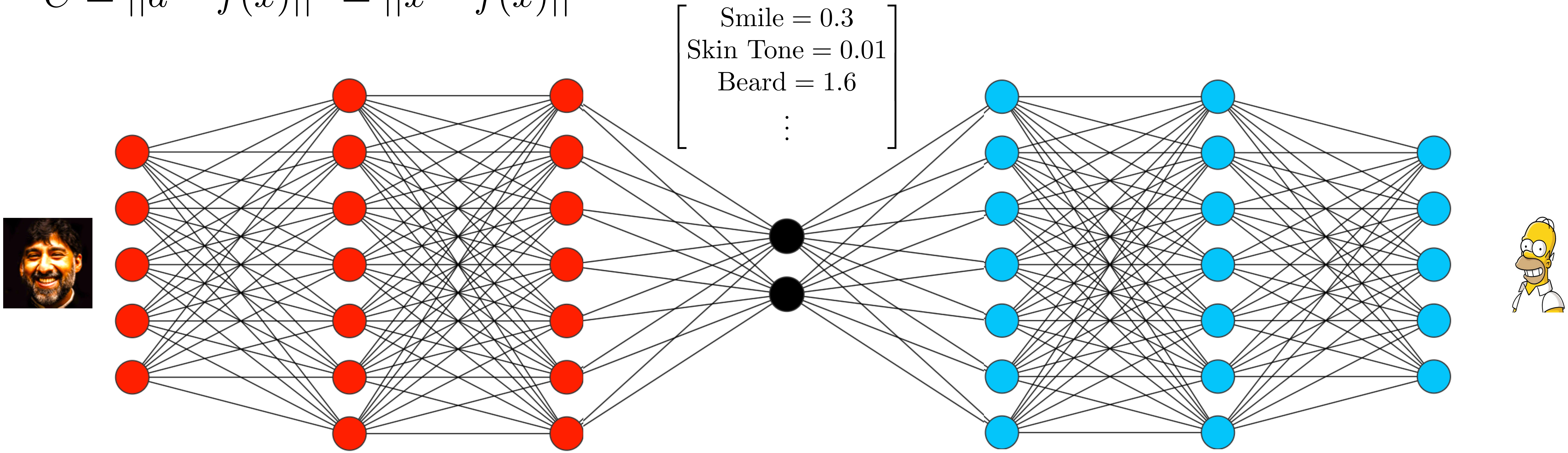
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



Implementation

Main features:

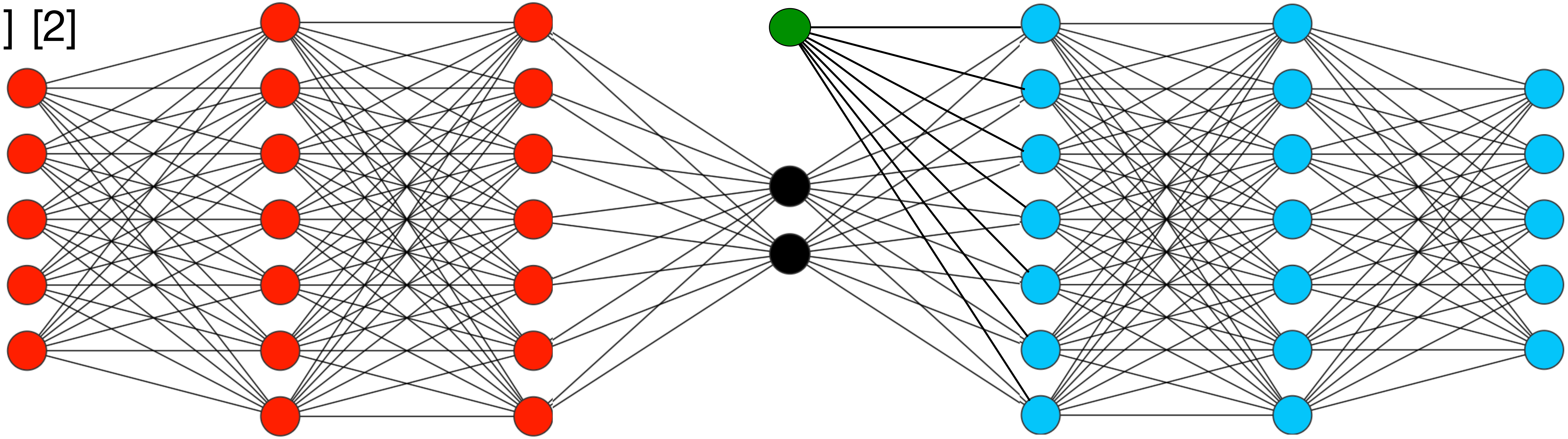
- **Number of neurons in the input layer = number of neurons in the output layer**
- **Number of neurons in the central hidden layer < number of neurons in the input layer**
- $C = ||\vec{a} - \vec{f}(\vec{x})||^2 = ||\vec{x} - \vec{f}(\vec{x})||^2$



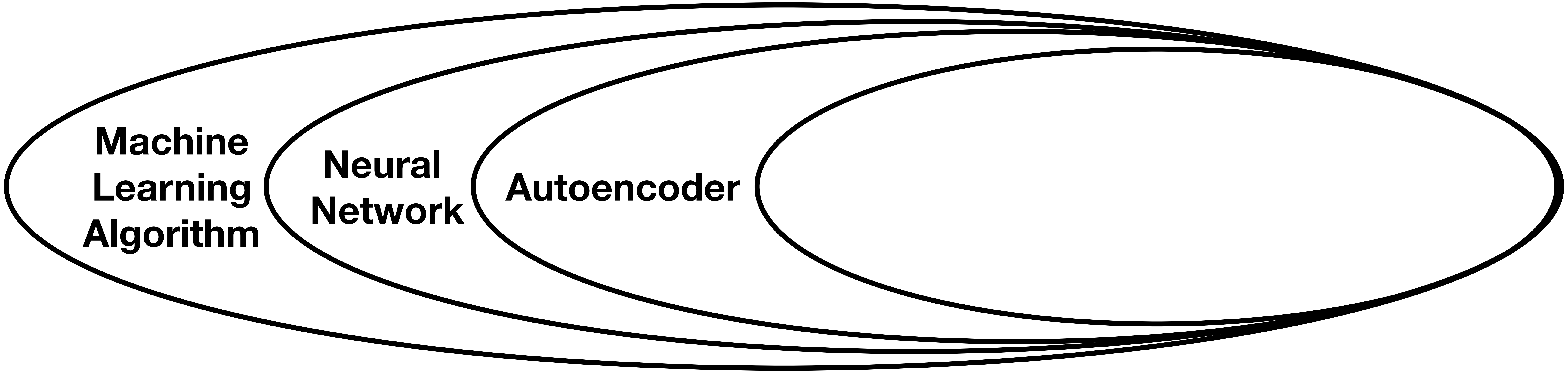
The Autoencoder is not “*Generative*”

Implementation

What is this? [1] [2]

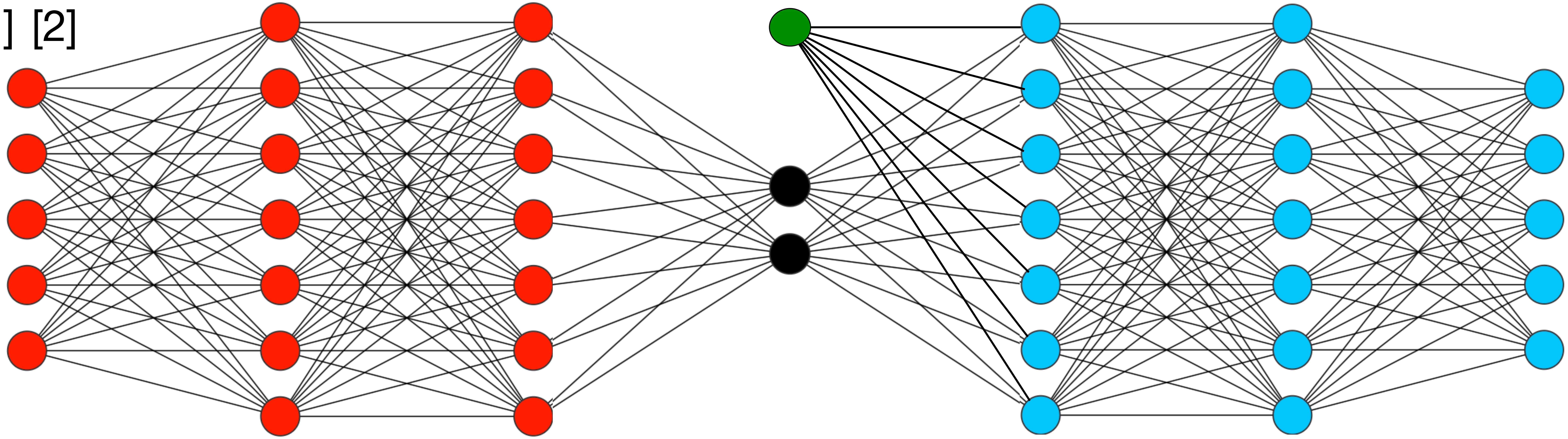


$SciNet =$

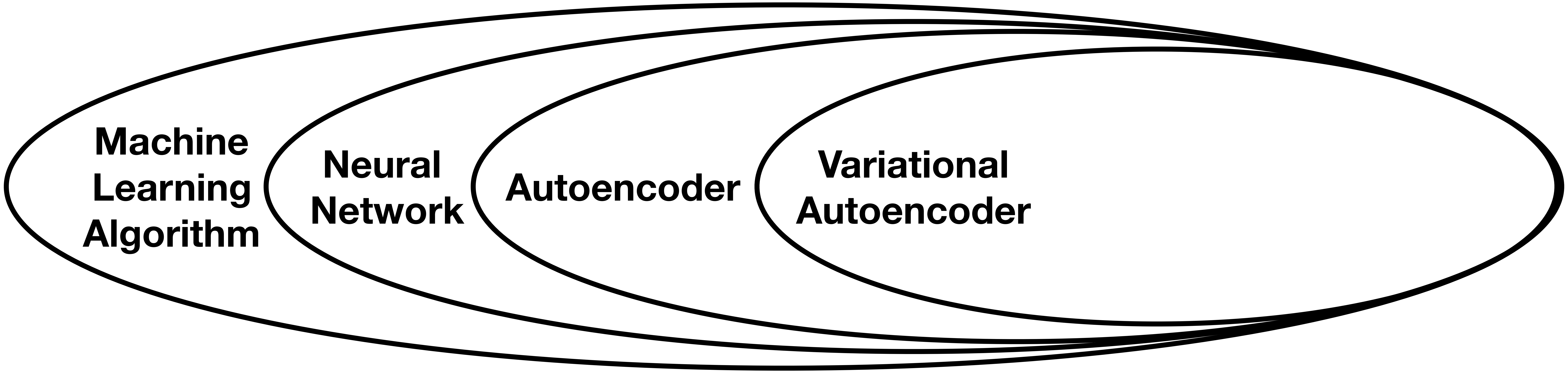


Implementation

What is this? [1] [2]

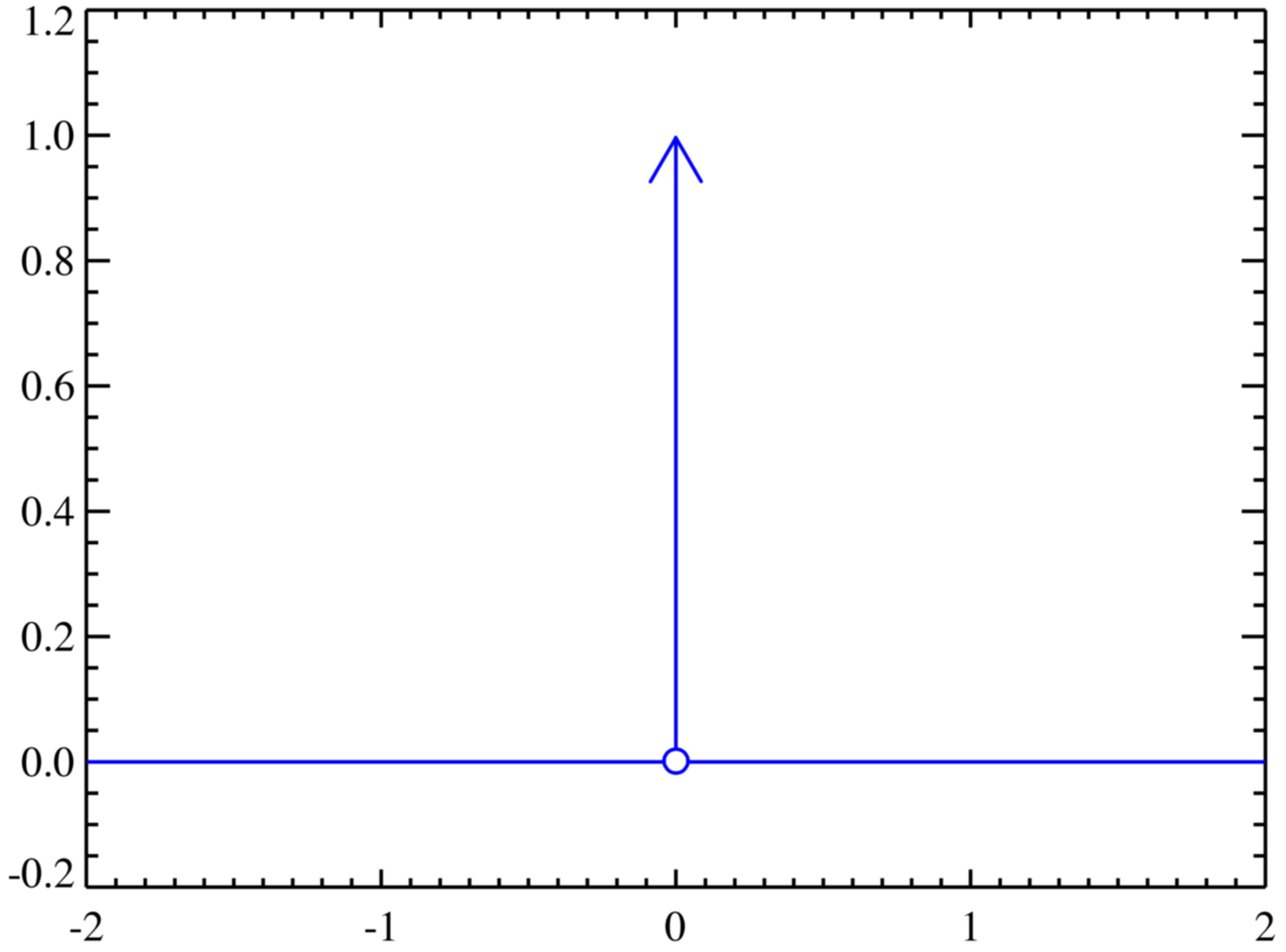


$SciNet =$



Implementation

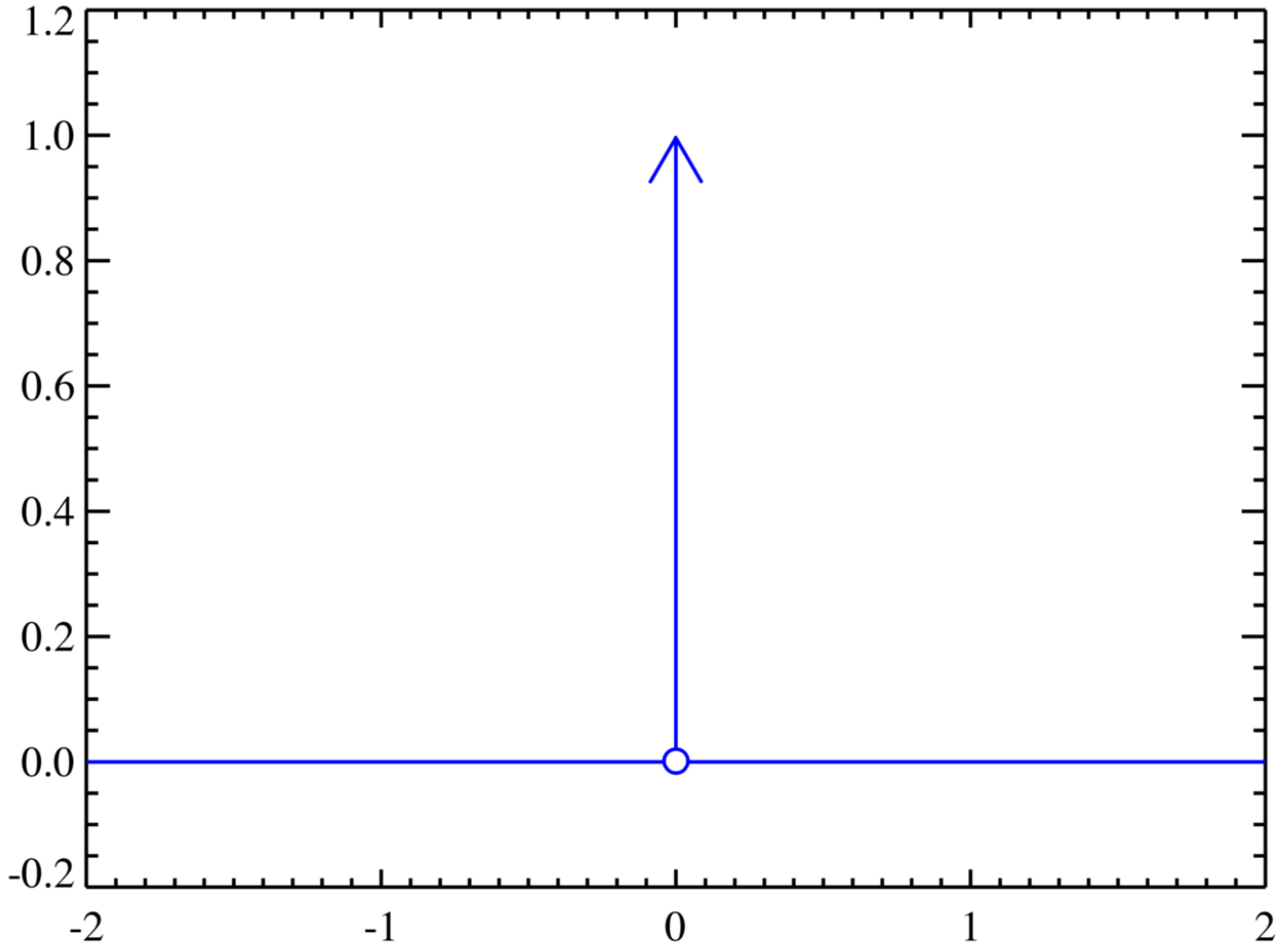
$$\text{Skin Tone} = \mu$$



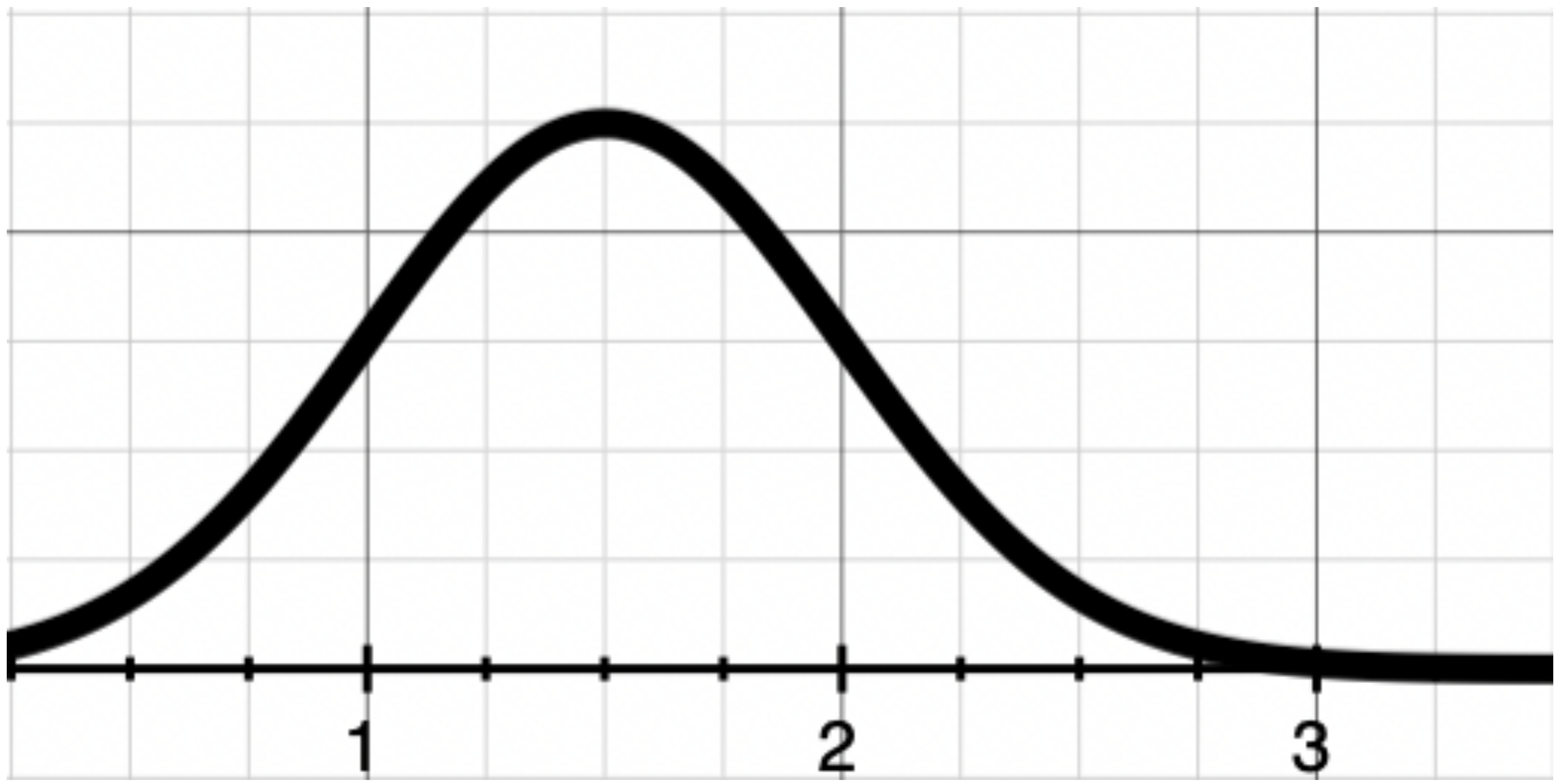
Implementation

$$\text{Skin Tone} = \mu$$

$$\text{Skin Tone} \sim f(\text{Skin Tone}; \mu, \sigma)$$



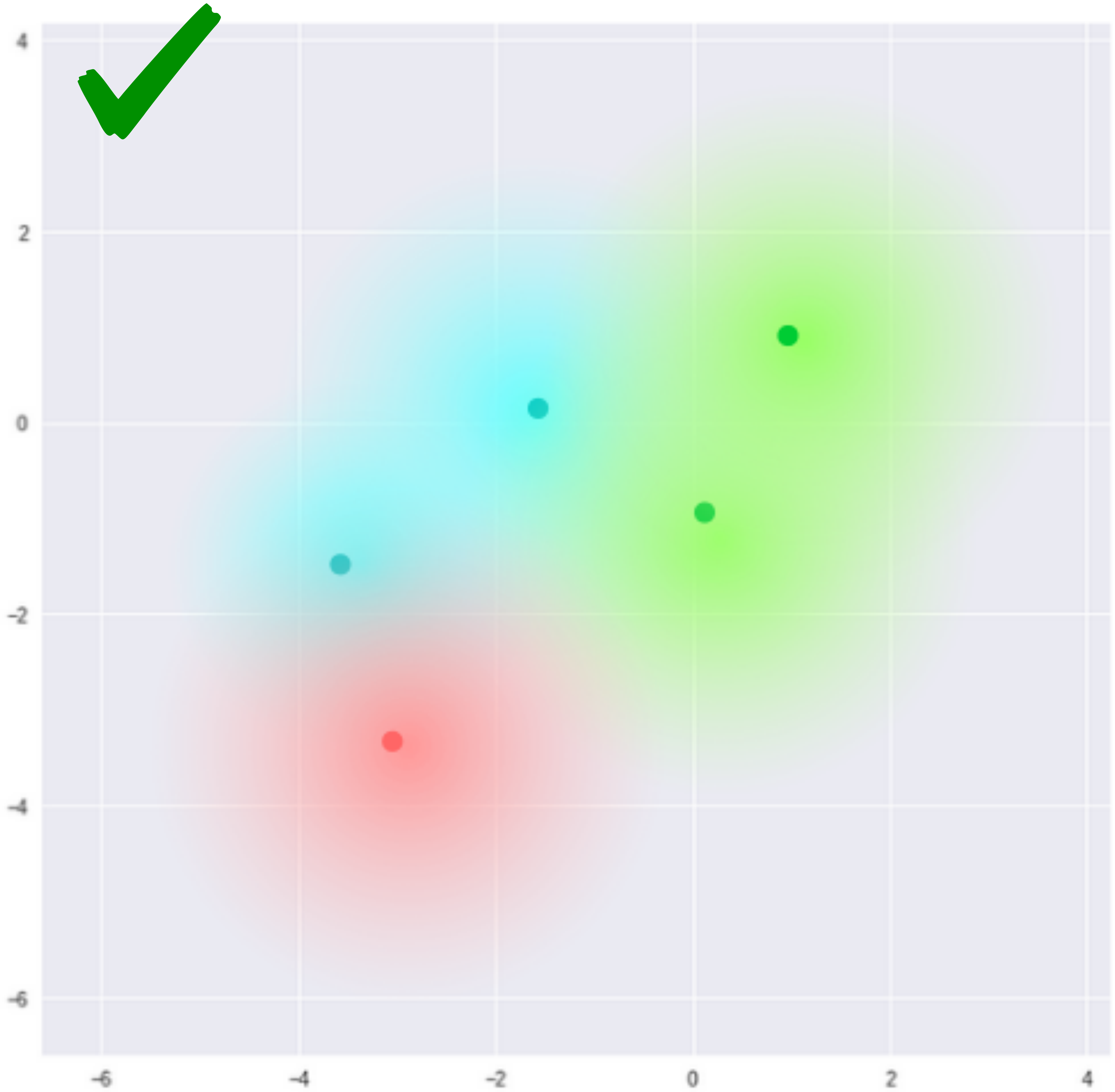
→

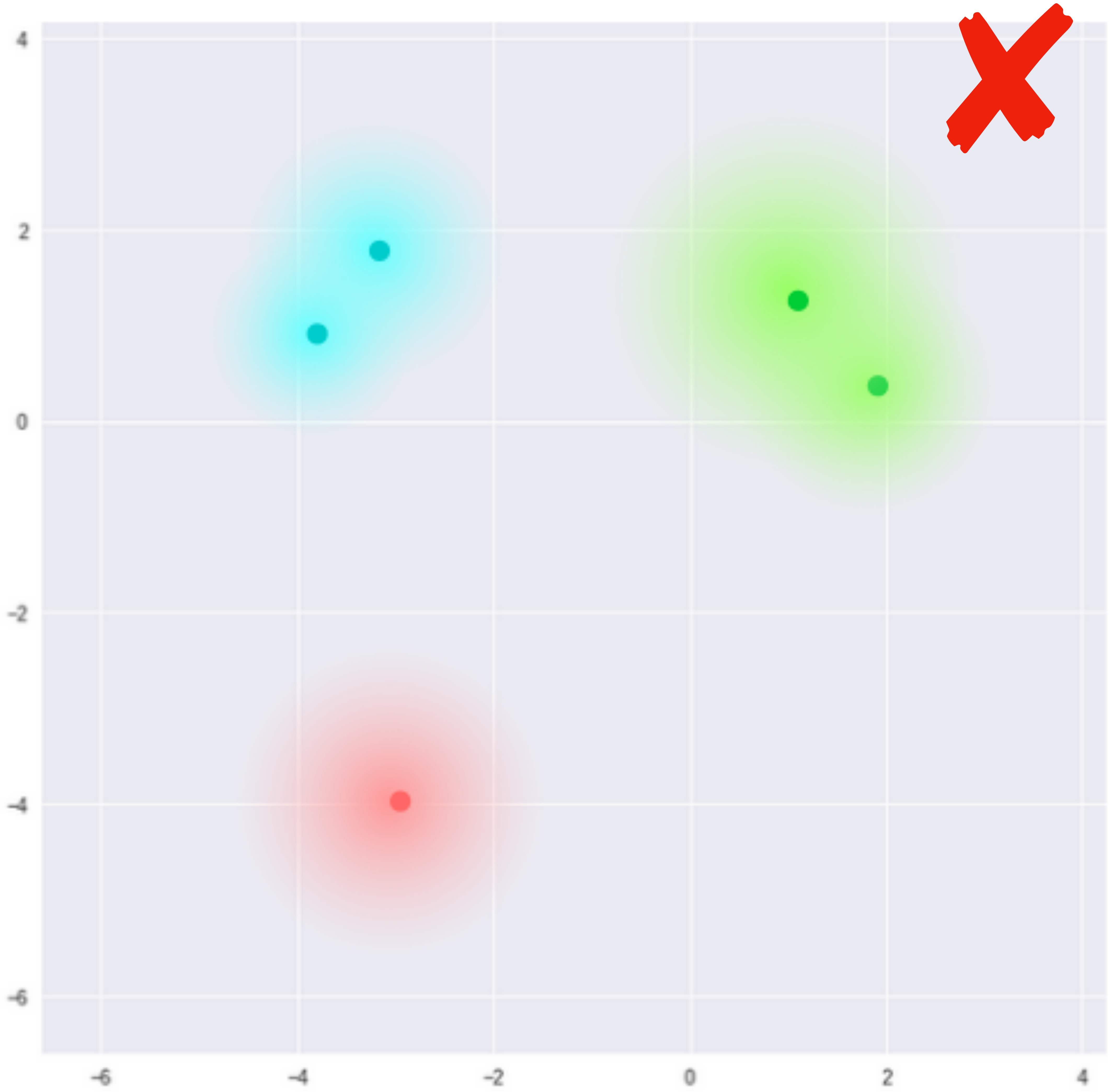


Implementation

Implementation

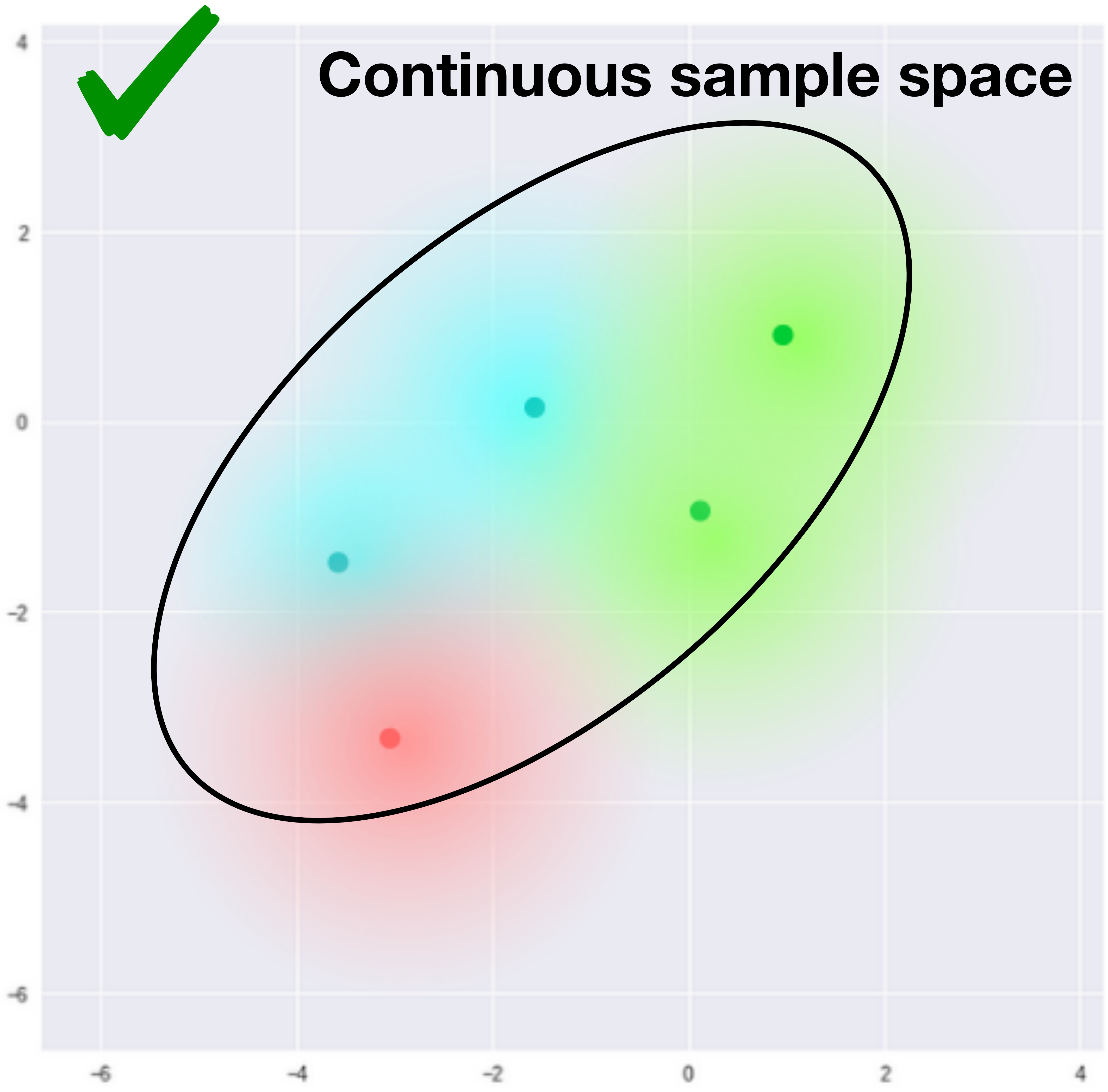
Space of Features

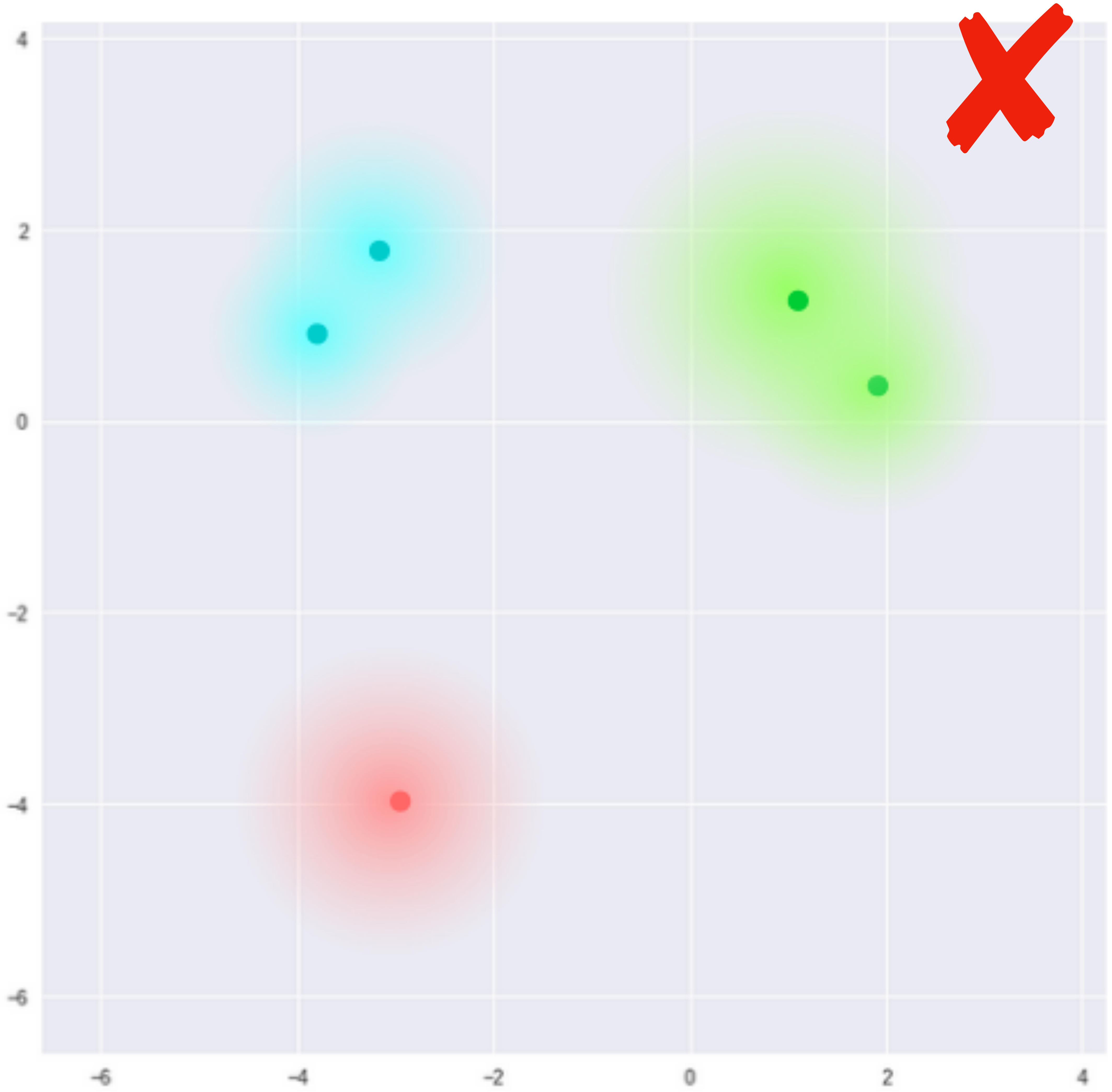




Implementation

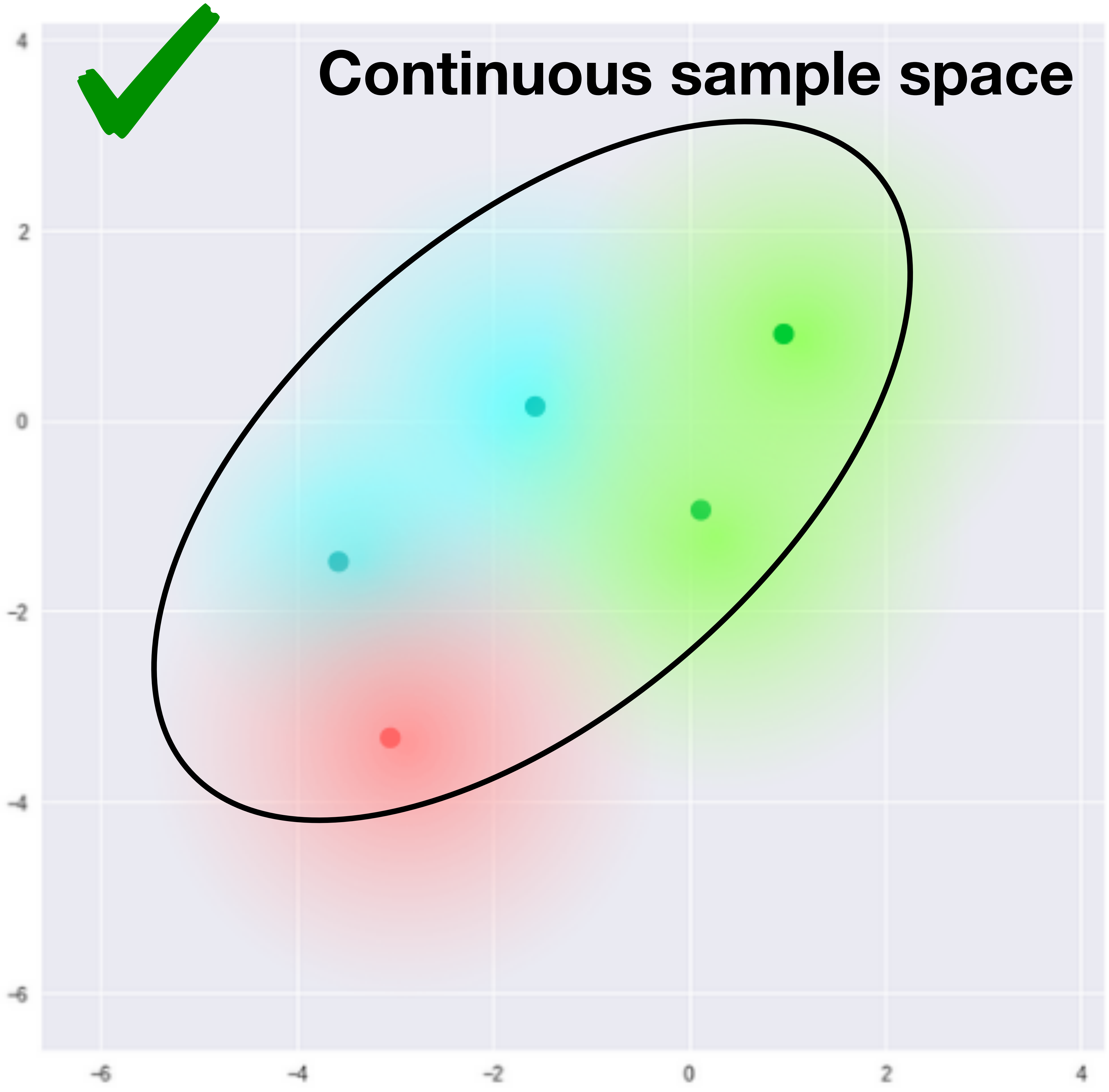
Space of Features





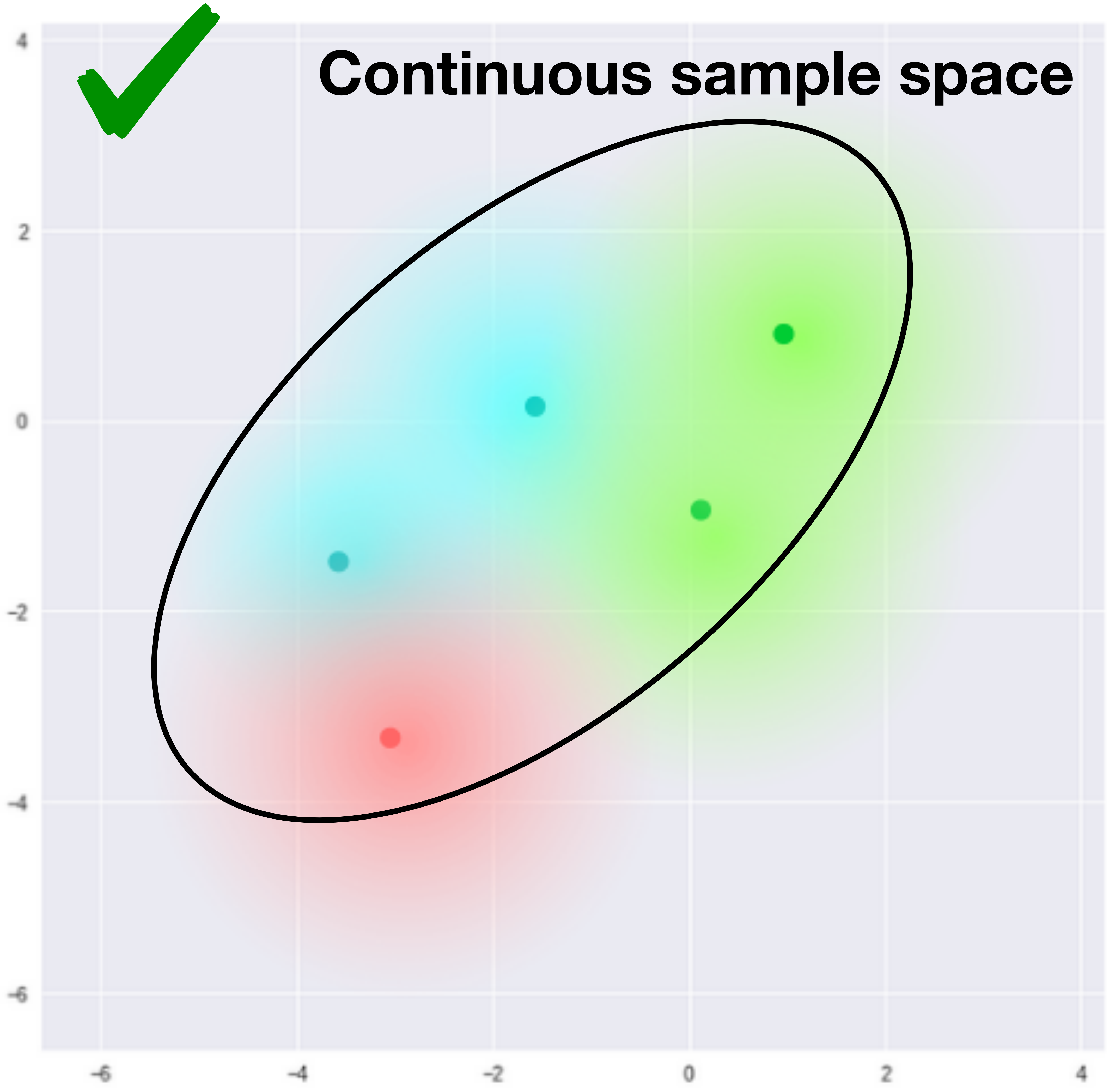
Implementation

Space of Features



Implementation

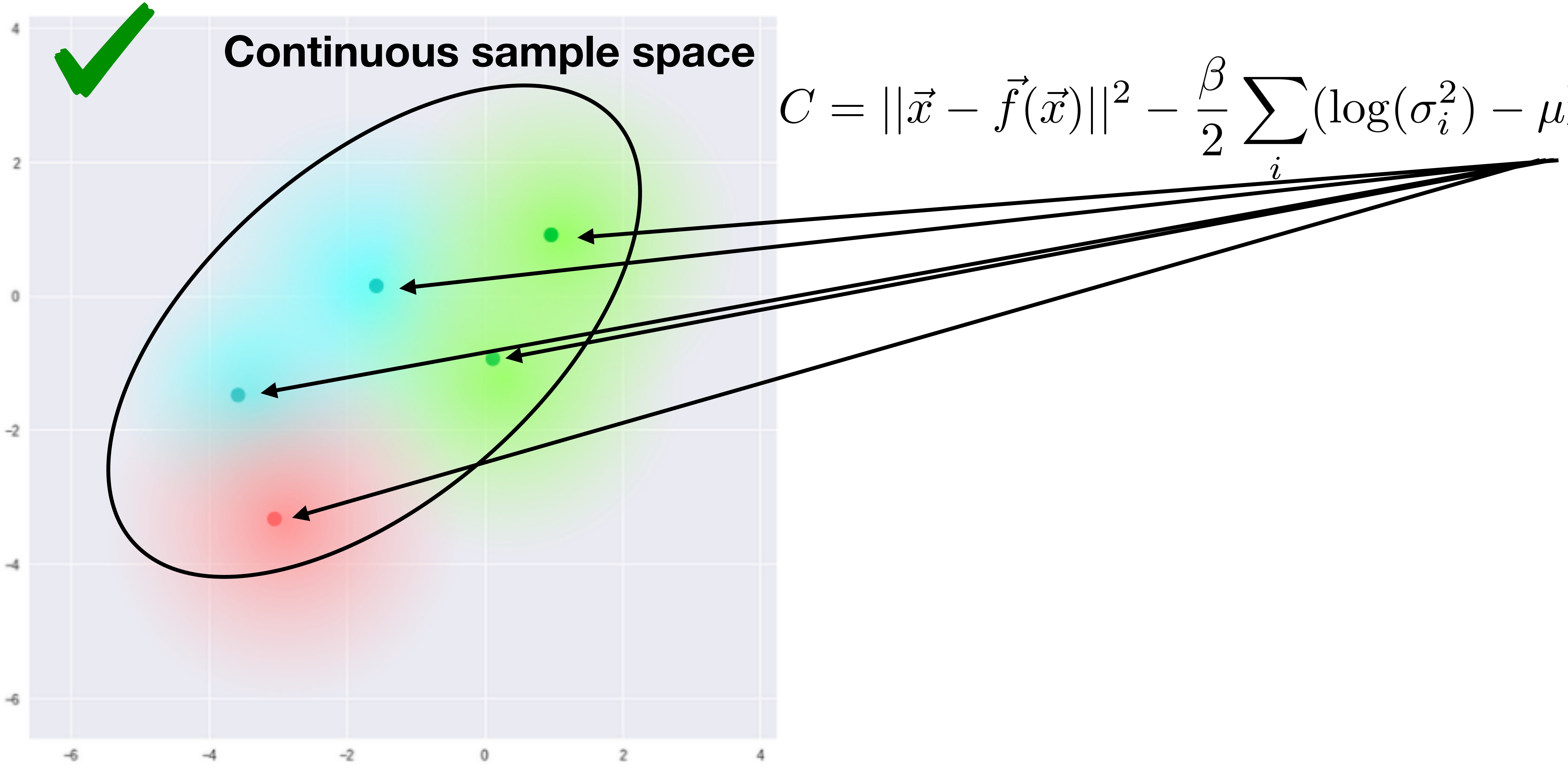
Space of Features



$$C = ||\vec{x} - \vec{f}(\vec{x})||^2 - \frac{\beta}{2}\sum_i (\log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)$$

Implementation

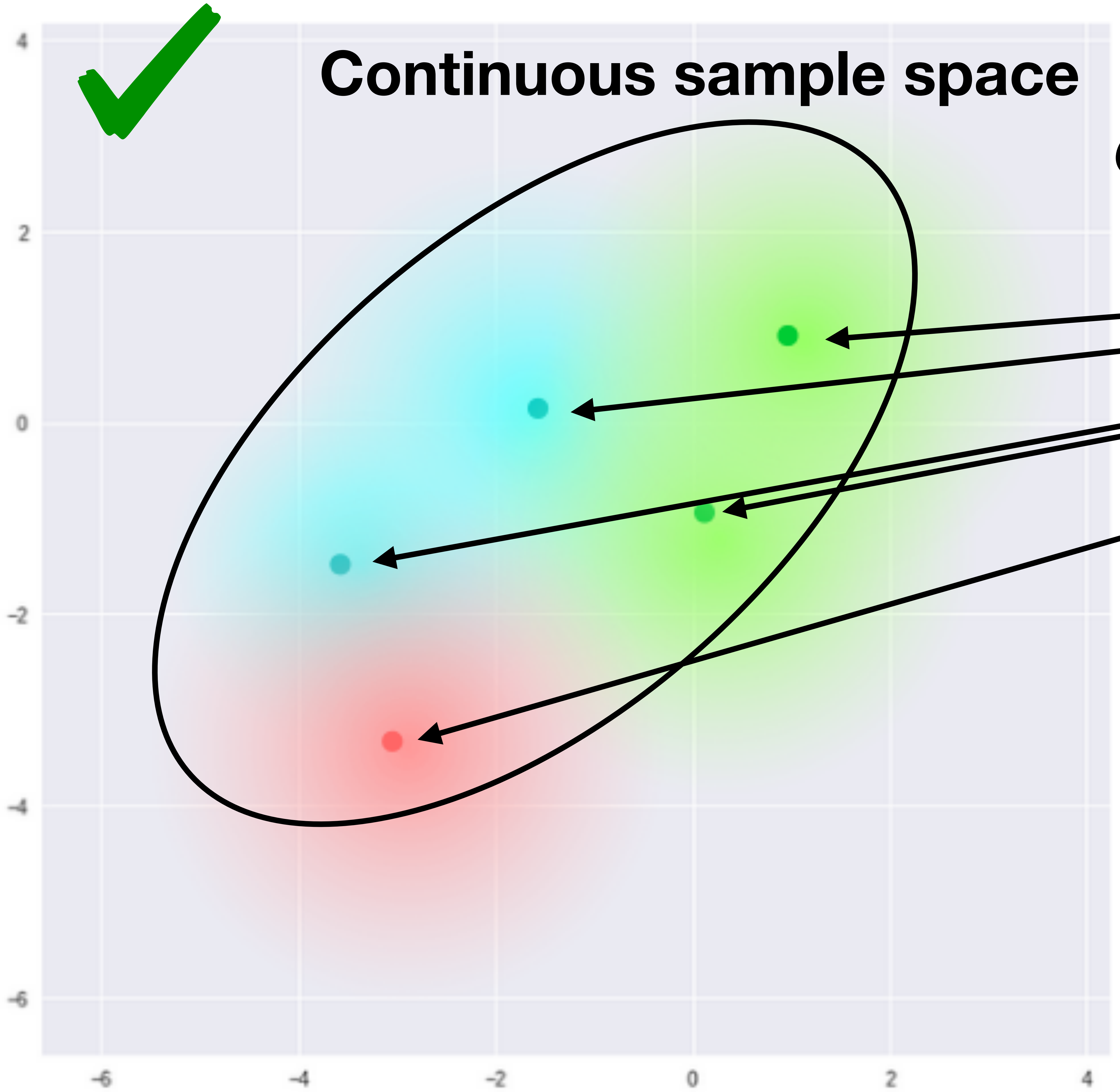
Space of Features



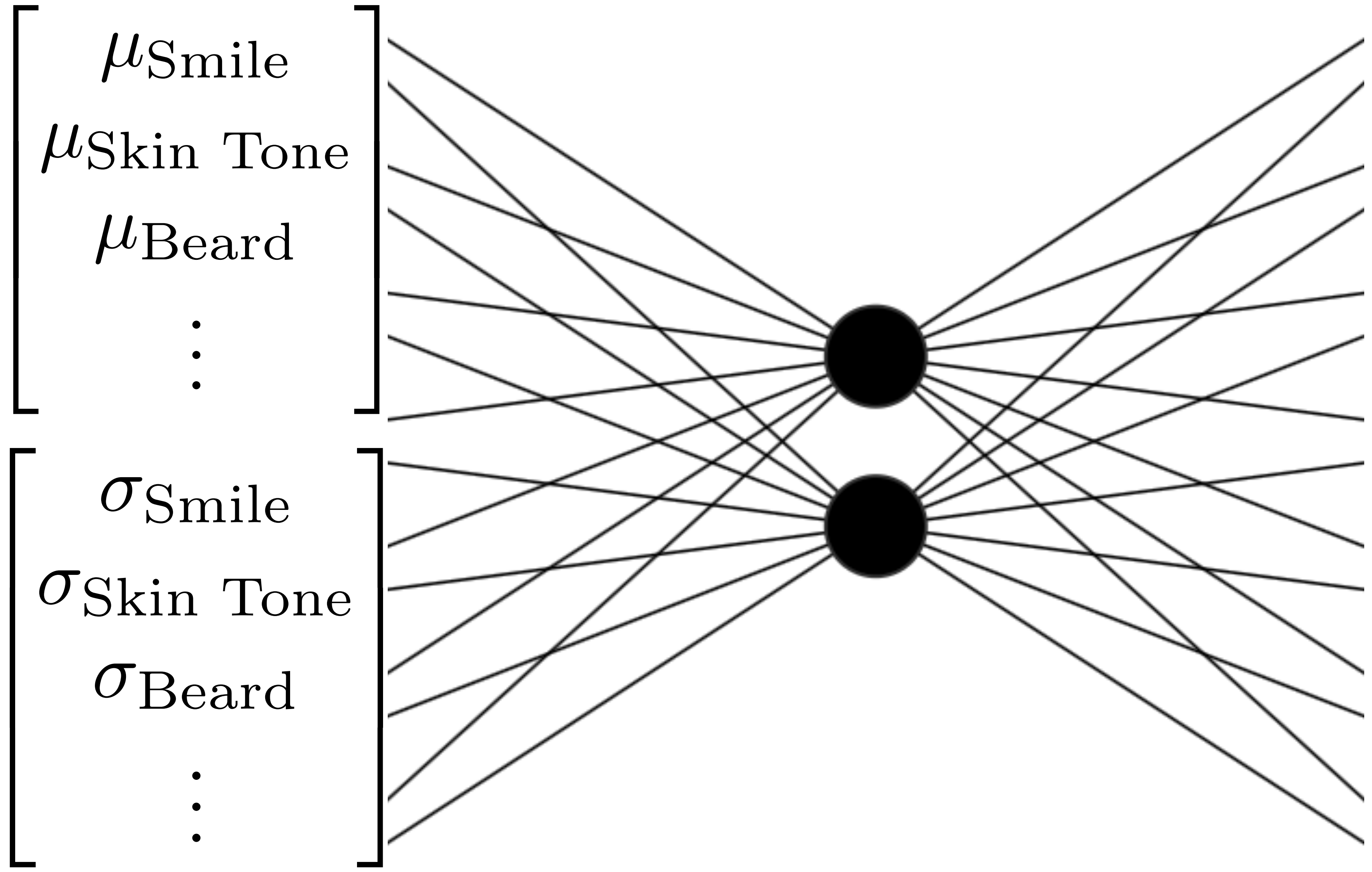
$$C = ||\vec{x} - \vec{f}(\vec{x})||^2 - \frac{\beta}{2}\sum_i (\log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)$$

Implementation

Space of Features

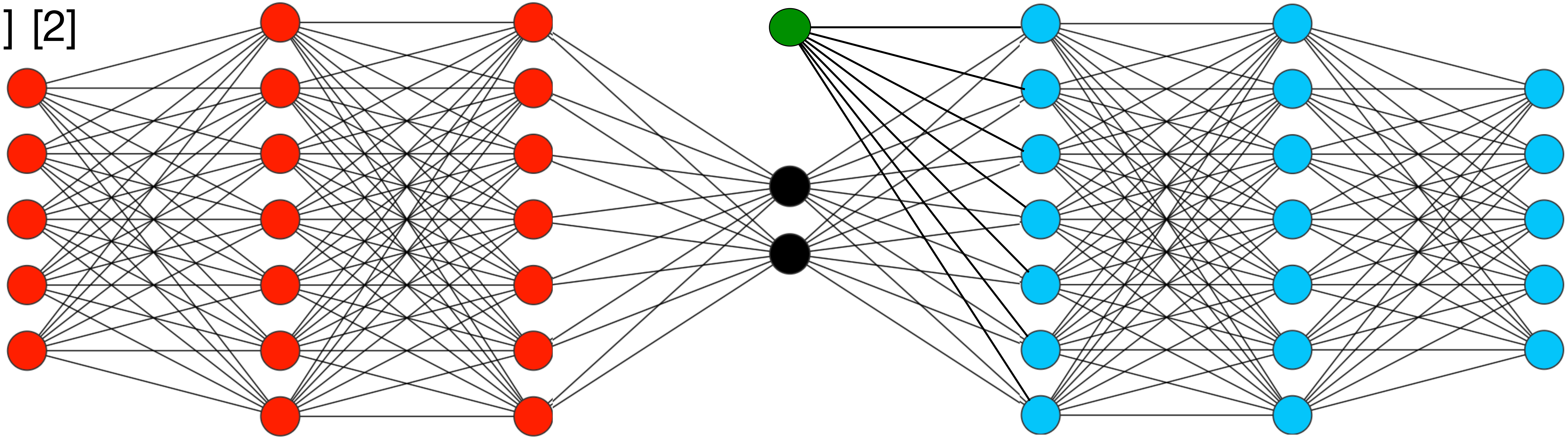


$$C = ||\vec{x} - \vec{f}(\vec{x})||^2 - \frac{\beta}{2}\sum_i (\log(\sigma_i^2) - \mu_i^2 - \sigma_i^2)$$

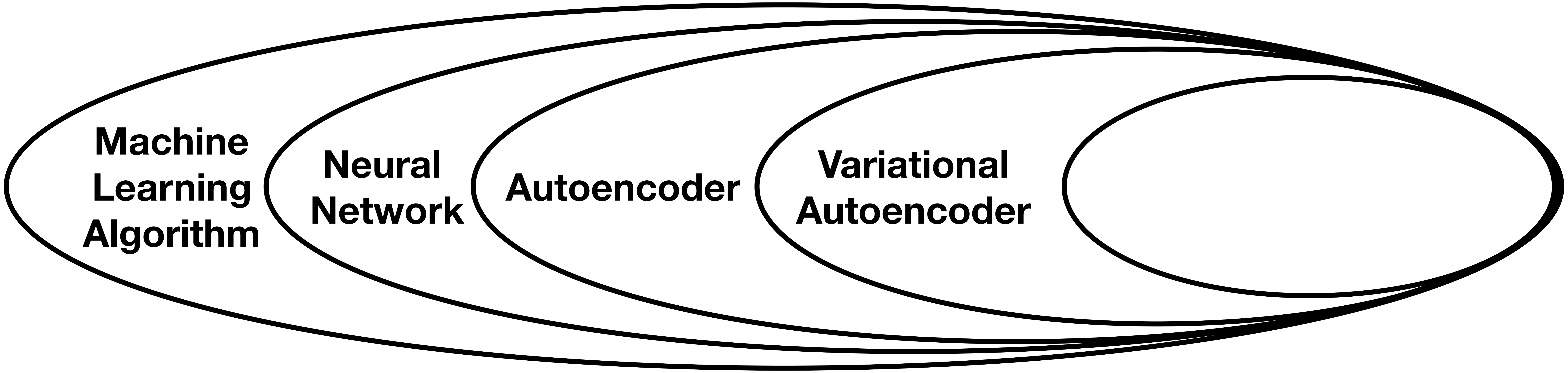


Implementation

What is this? [1] [2]

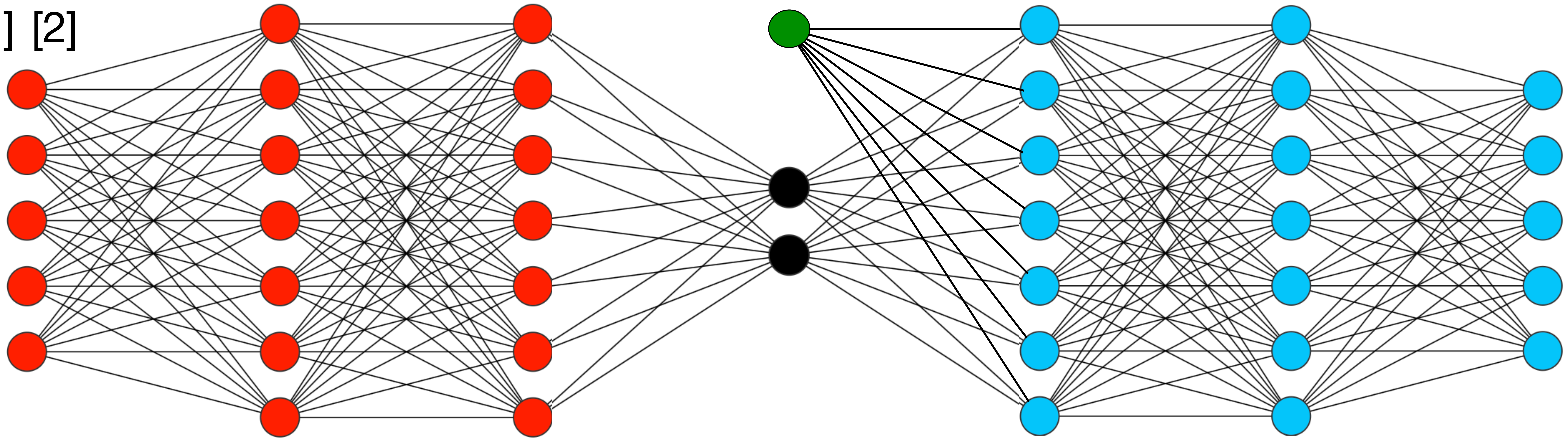


$SciNet =$

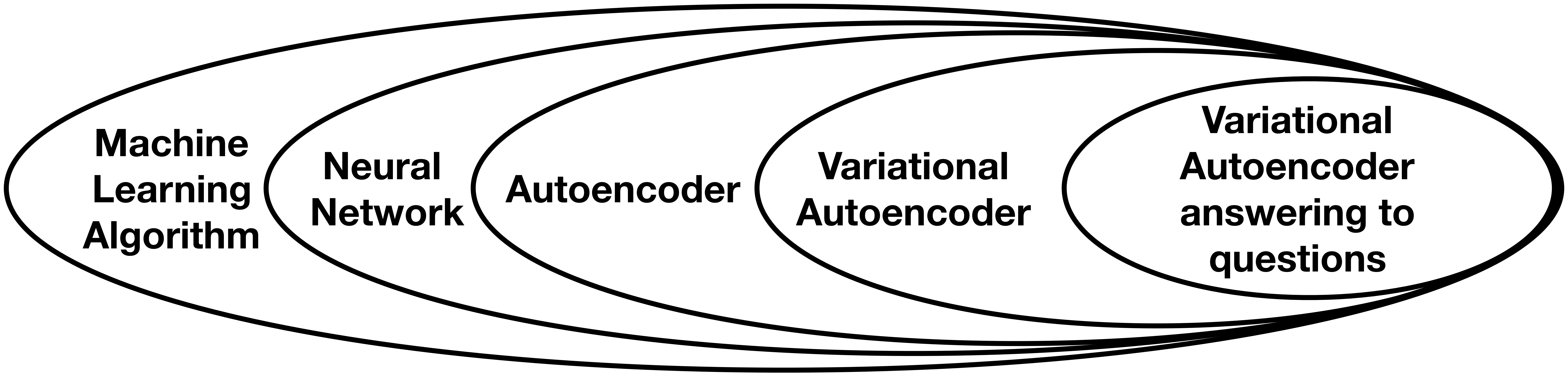


Implementation

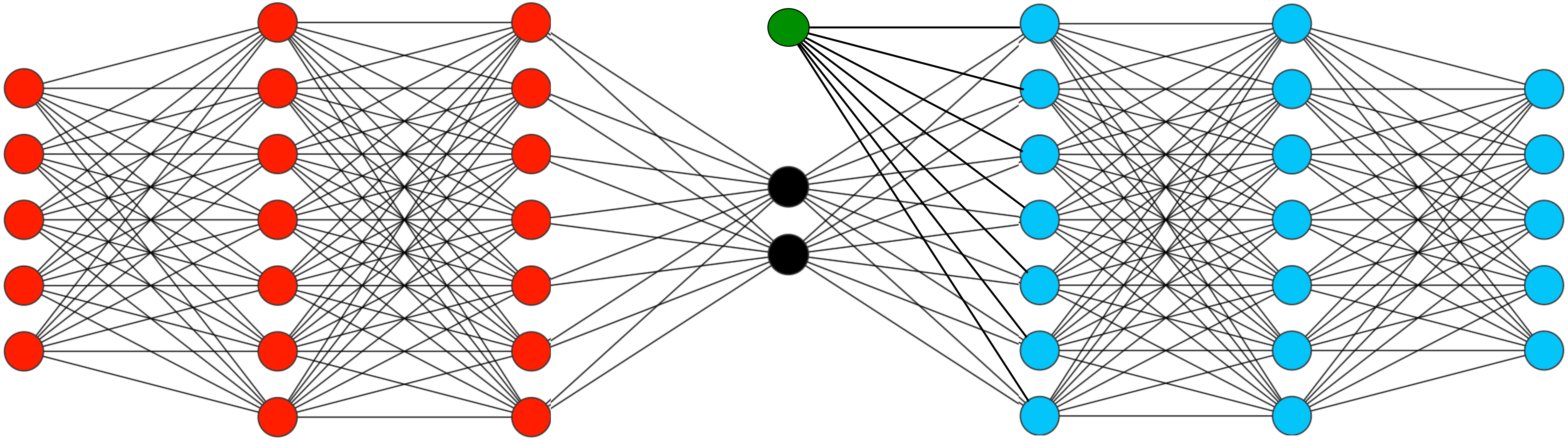
What is this? [1] [2]



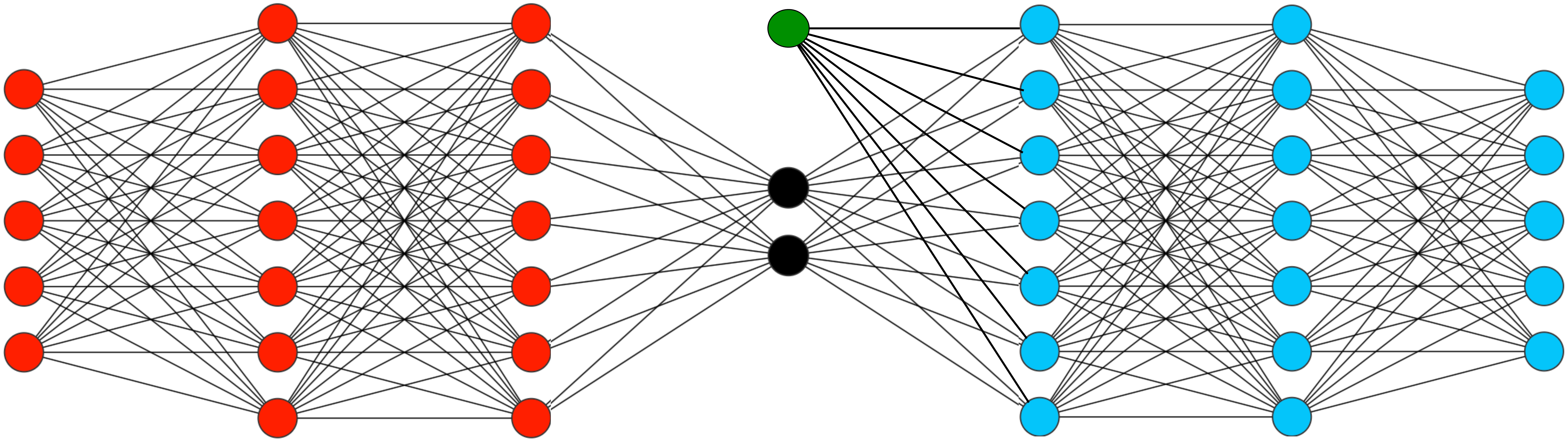
$$SciNet =$$



Implementation



Implementation



How to check if *Scinet* is "understanding":

- ✦**The number of fundamental quantities of the system = number of latent neurons of the network (NLN), being NLN selected by repeating the cycle "add 1 latent neuron + training + testing" until a decided accuracy is reached**
- ✦**Plot the correlations of the known fundamental quantities we know are necessary to make predictions respect to the ones *Scinet* stores in its latent neurons**

Examples: recovering of physical variables [1][2]

Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Implementation:
VAE with 3 latent neurons

Results:

Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

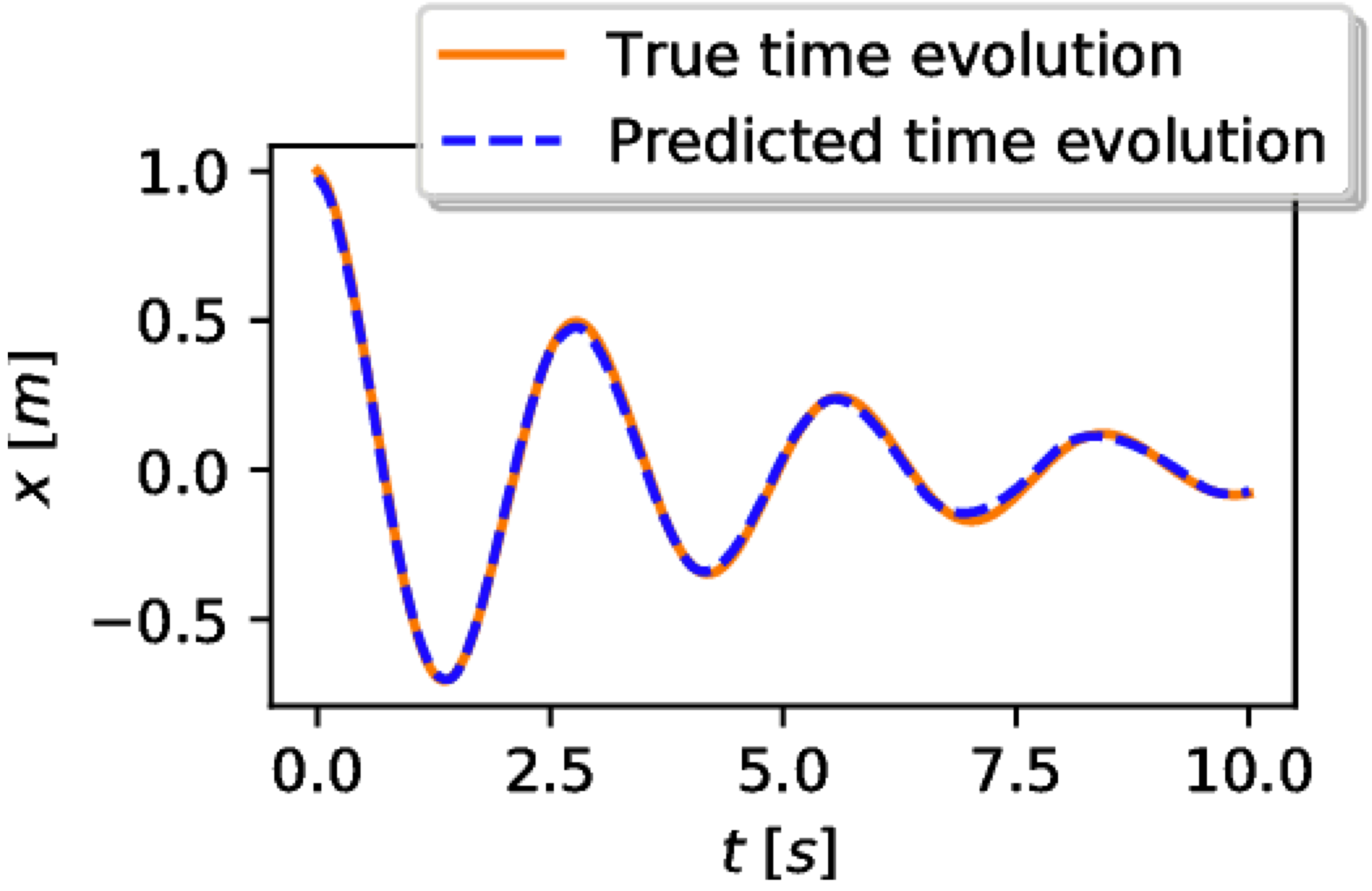
$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Implementation:
VAE with 3 latent neurons

Results:



Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

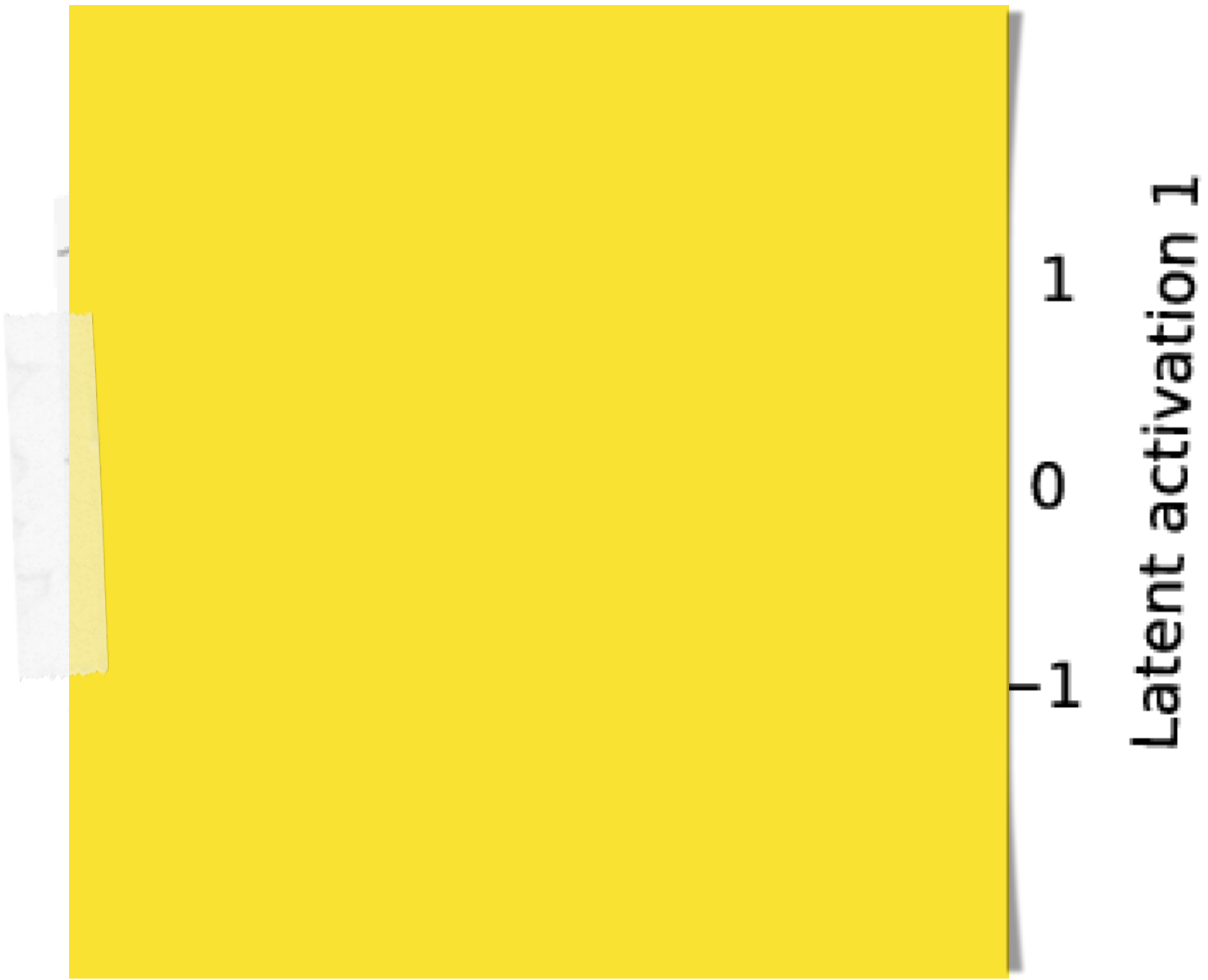
$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Implementation:
VAE with 3 latent neurons

Results:



Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

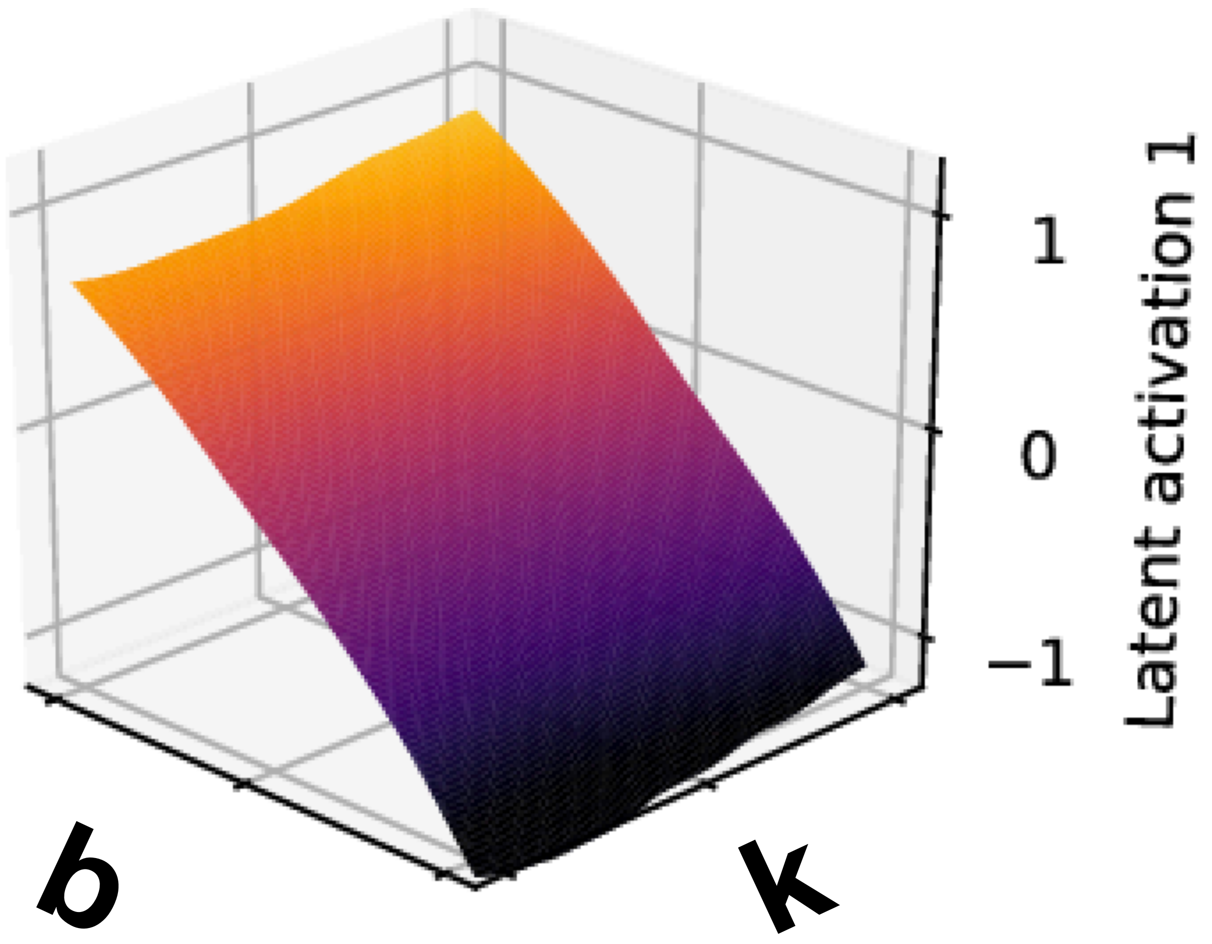
$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

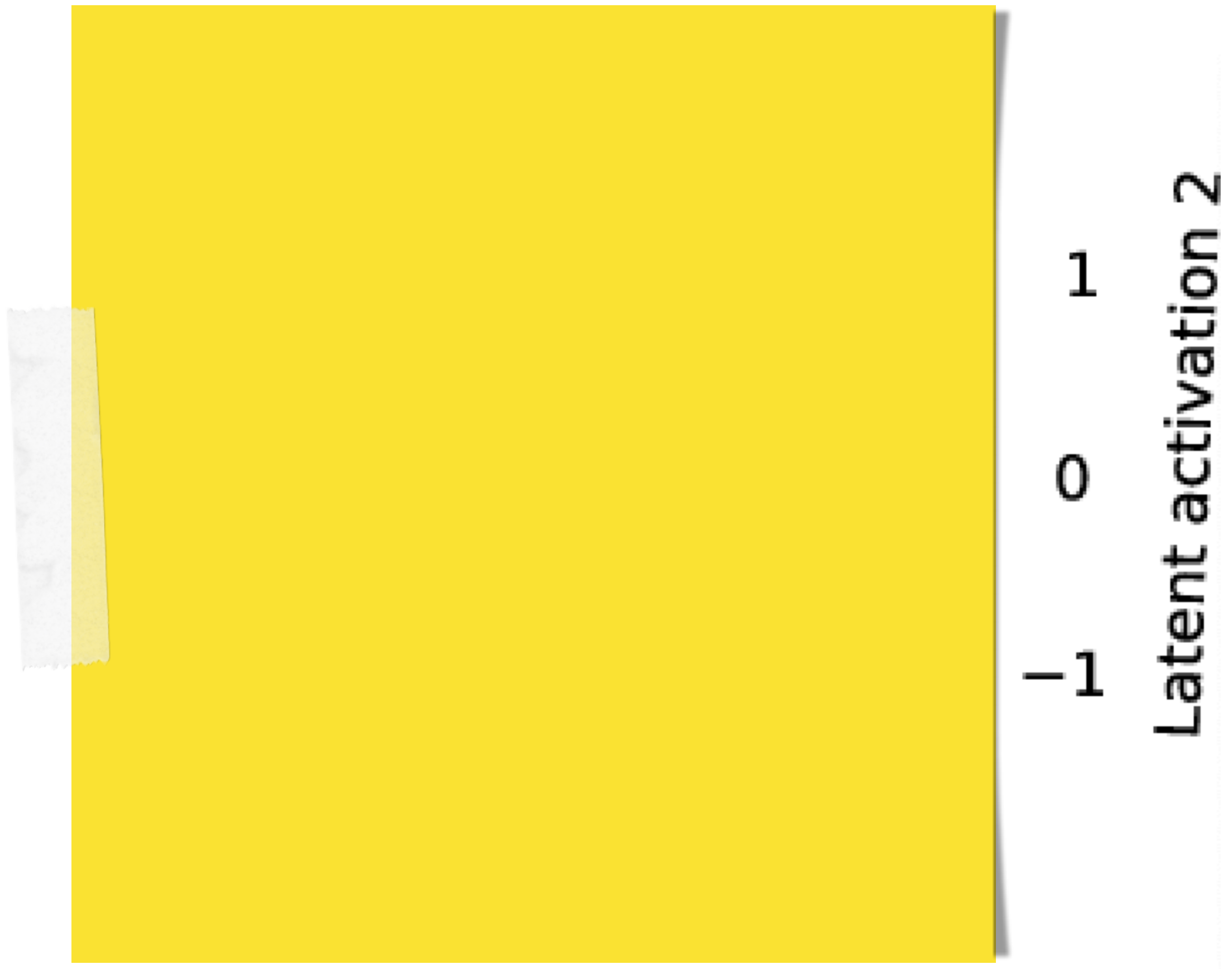
$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Implementation:
VAE with 3 latent neurons

Results:





Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos\omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

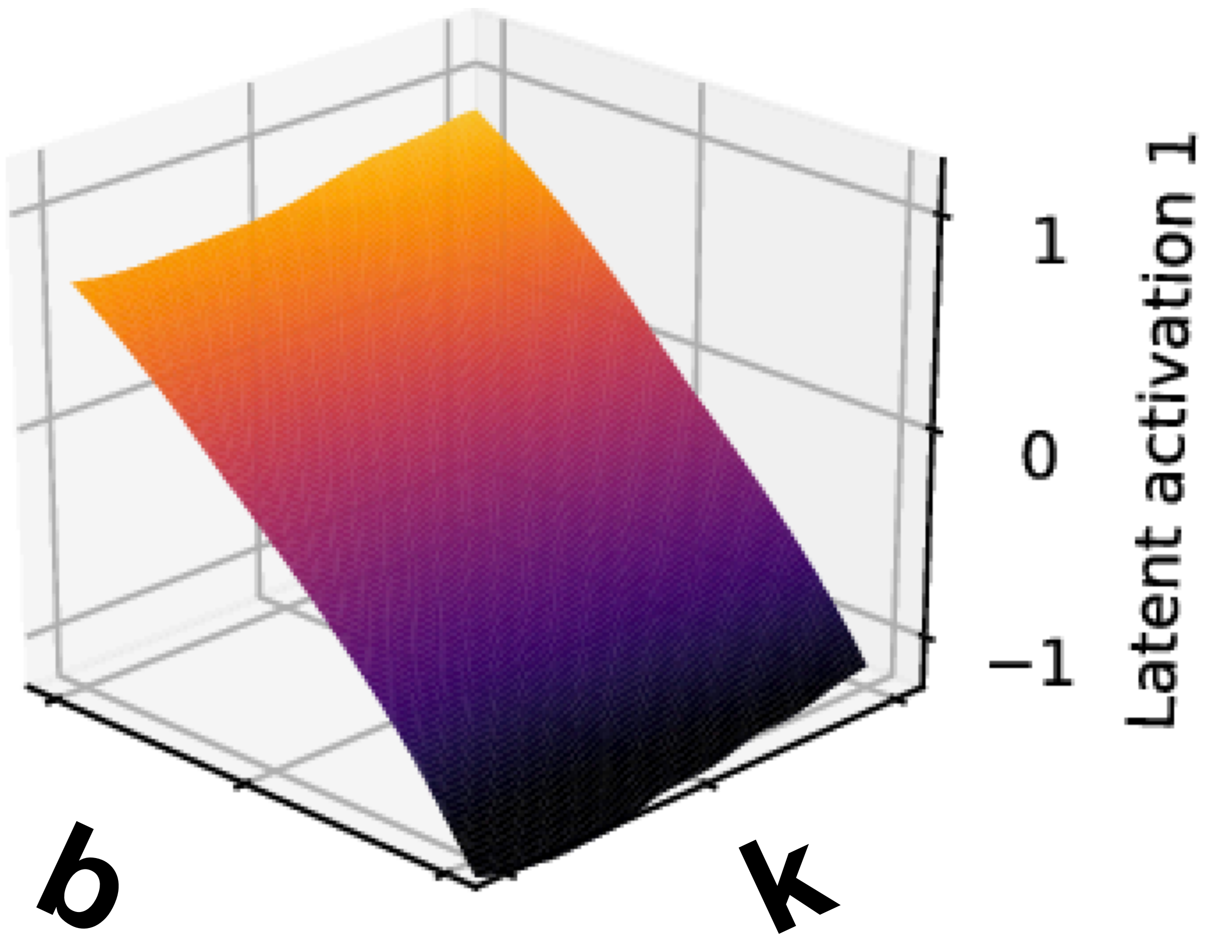
$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

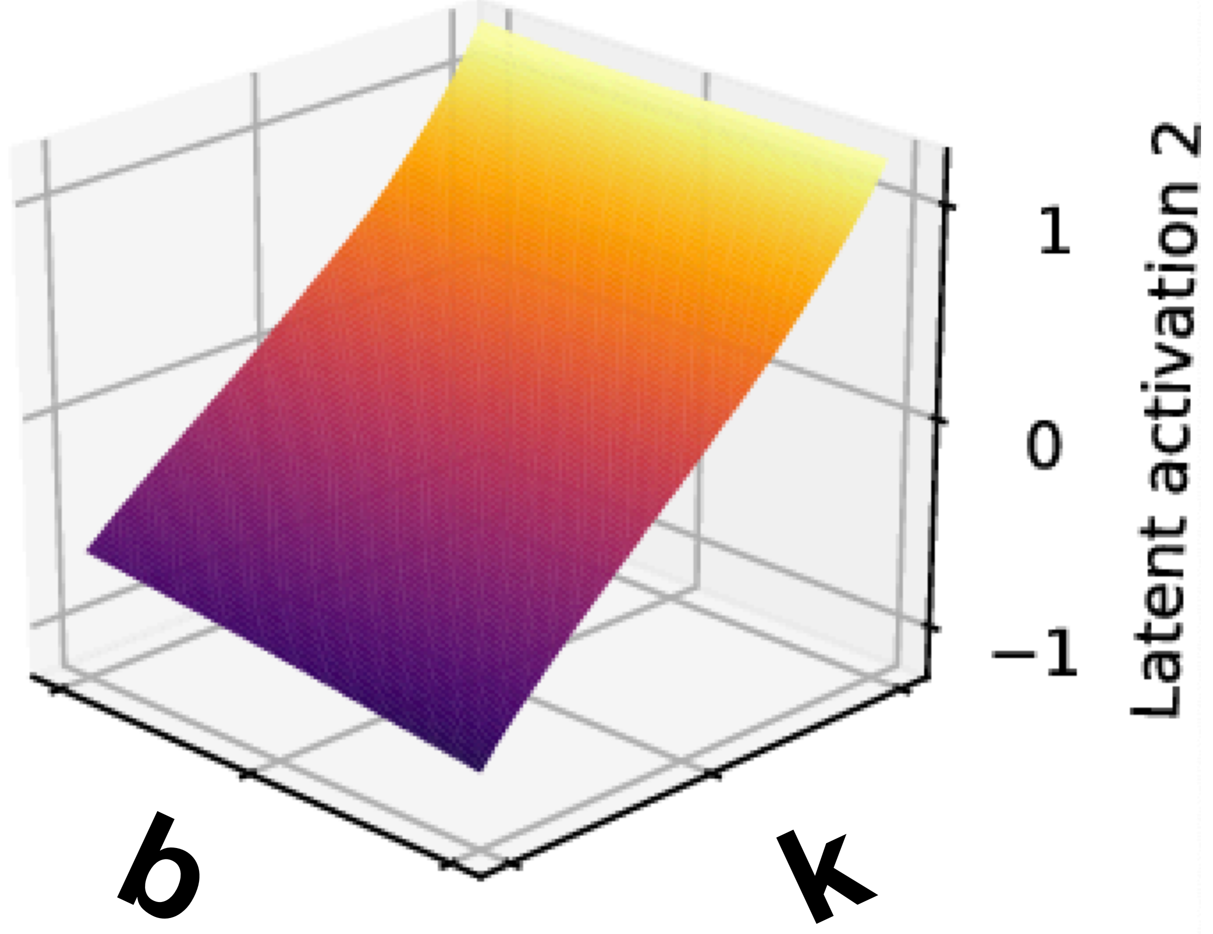
$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Implementation:
VAE with 3 latent neurons

Results:





Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

Implementation:
VAE with 3 latent neurons

Results:

Examples: recovering of physical variables [1][2]

Problem:
Predict the position of a dumped pendulum at different times

Equations:

$$m\ddot{x} = -kx - b\dot{x}$$

$$x(t) = A_0 e^{\frac{-b}{2m}t} \cos \omega t + \delta_0, \omega = \sqrt{\frac{k}{m}}\sqrt{1 - \frac{b^2}{4mk}}$$

Quantities:

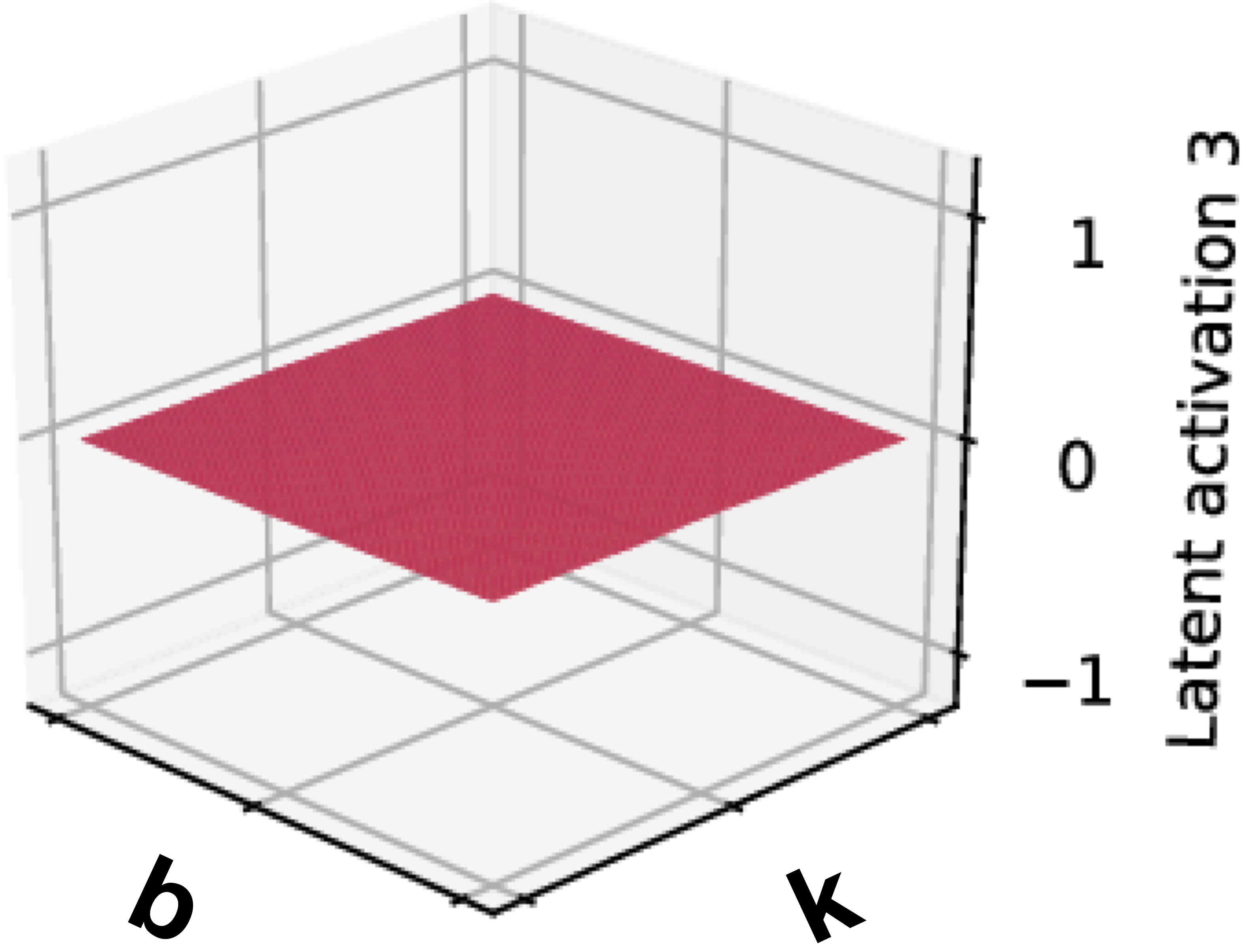
$$o = \begin{bmatrix} x(t_1)) \\ \vdots \\ x(t_{50}) \end{bmatrix} \in \mathbb{R}^{50}$$

$$q = t_{pred} \in \mathbb{R}$$

$$a_{corr}(o, q) = x(q) \in \mathbb{R}$$

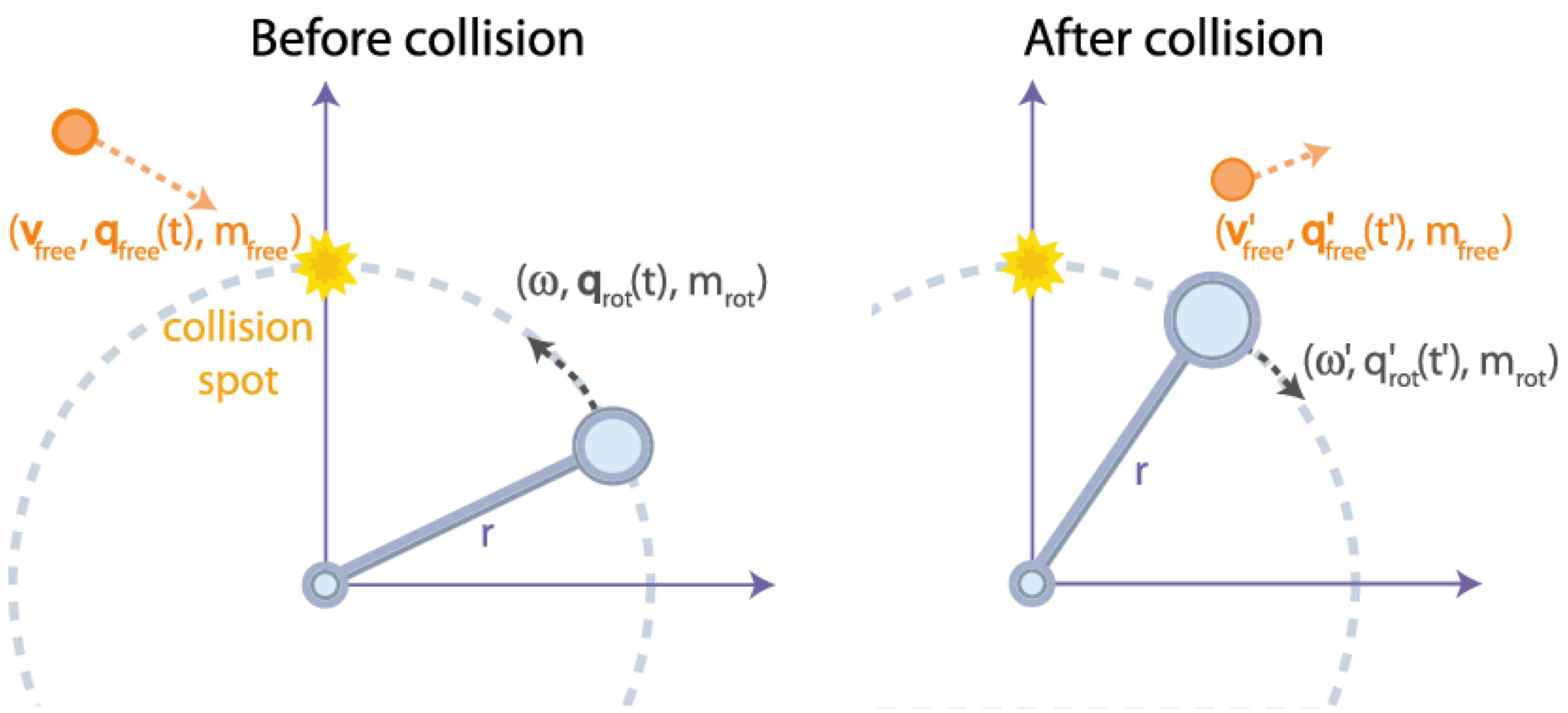
Implementation:
VAE with 3 latent neurons

Results:



Examples: recovering of conserved quantities [1][2]

Angular Momentum



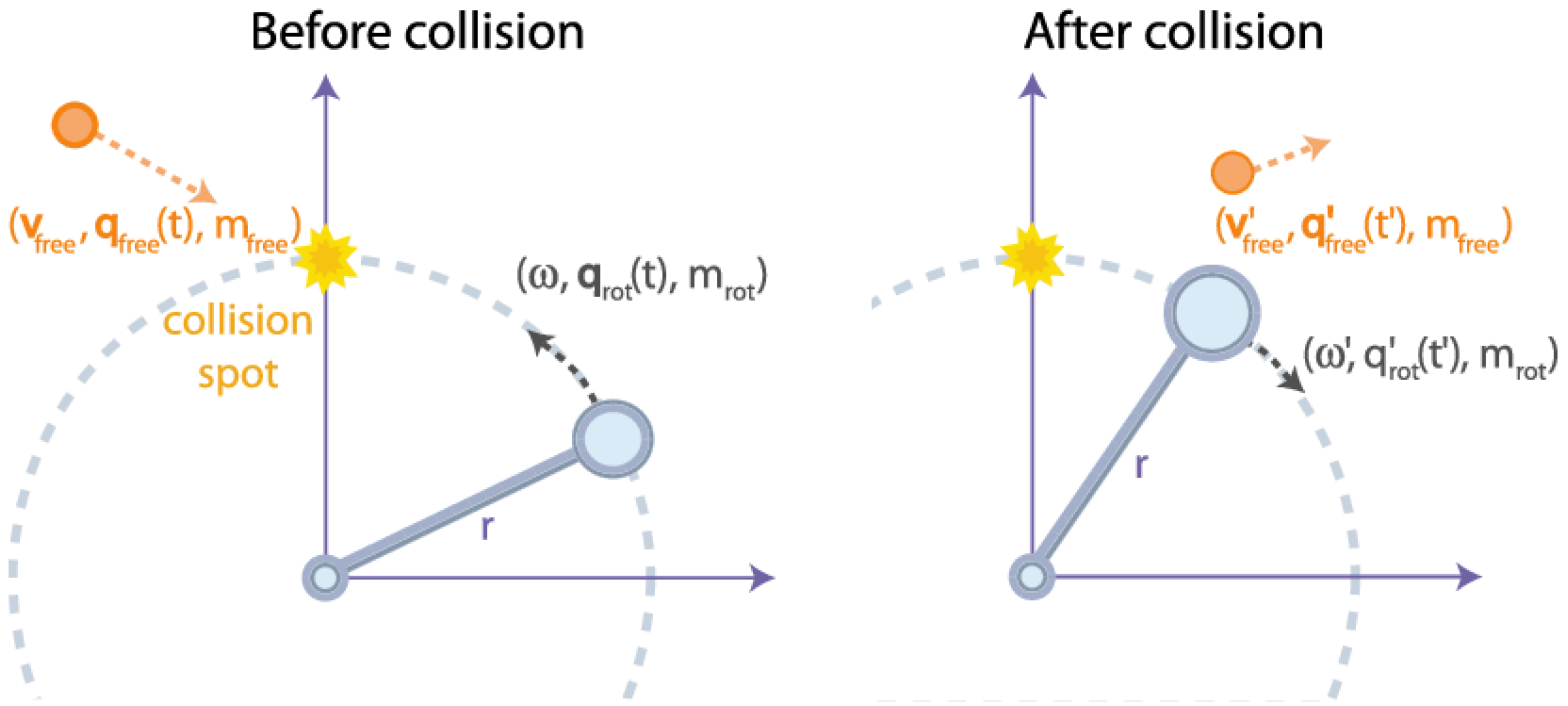
$$J = m_{rot} r^2 \omega - r m_{free} (\vec{v}_{free})_x = m_{rot} r^2 \omega' - r m (\vec{v}'_{free})_x = J'$$

$$\begin{cases} o = [(t_{rot}, \vec{q}(t_{rot})), (t_{free}, \vec{q}(t_{free}))] \\ q = (t'_{pred}, \vec{q'}_{free}(t_{pred})) \\ a_{cor} = \vec{q}_{rot}(t'_{pred}) \end{cases}$$

Total angular momentum

Examples: recovering of conserved quantities [1][2]

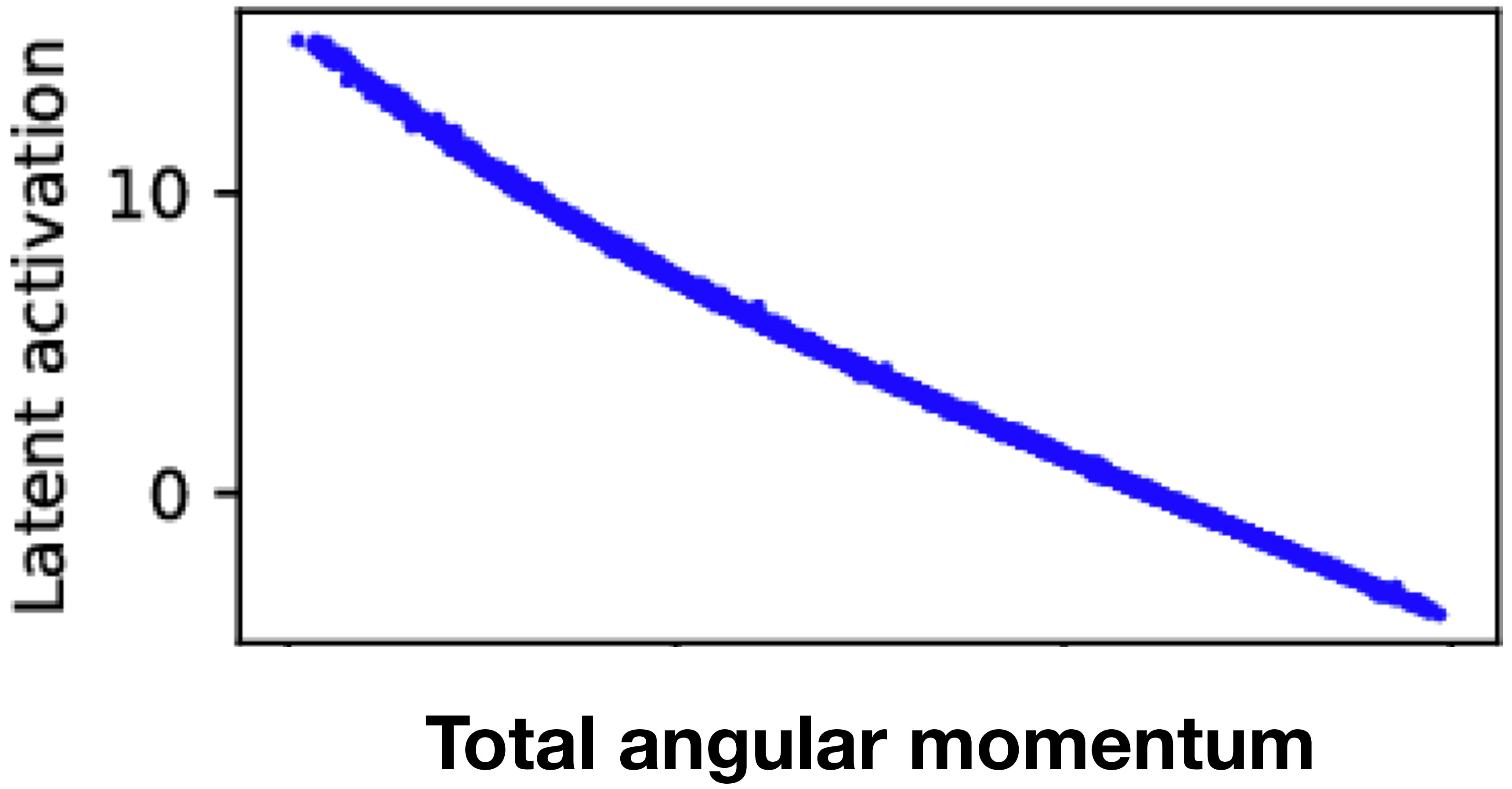
Angular Momentum



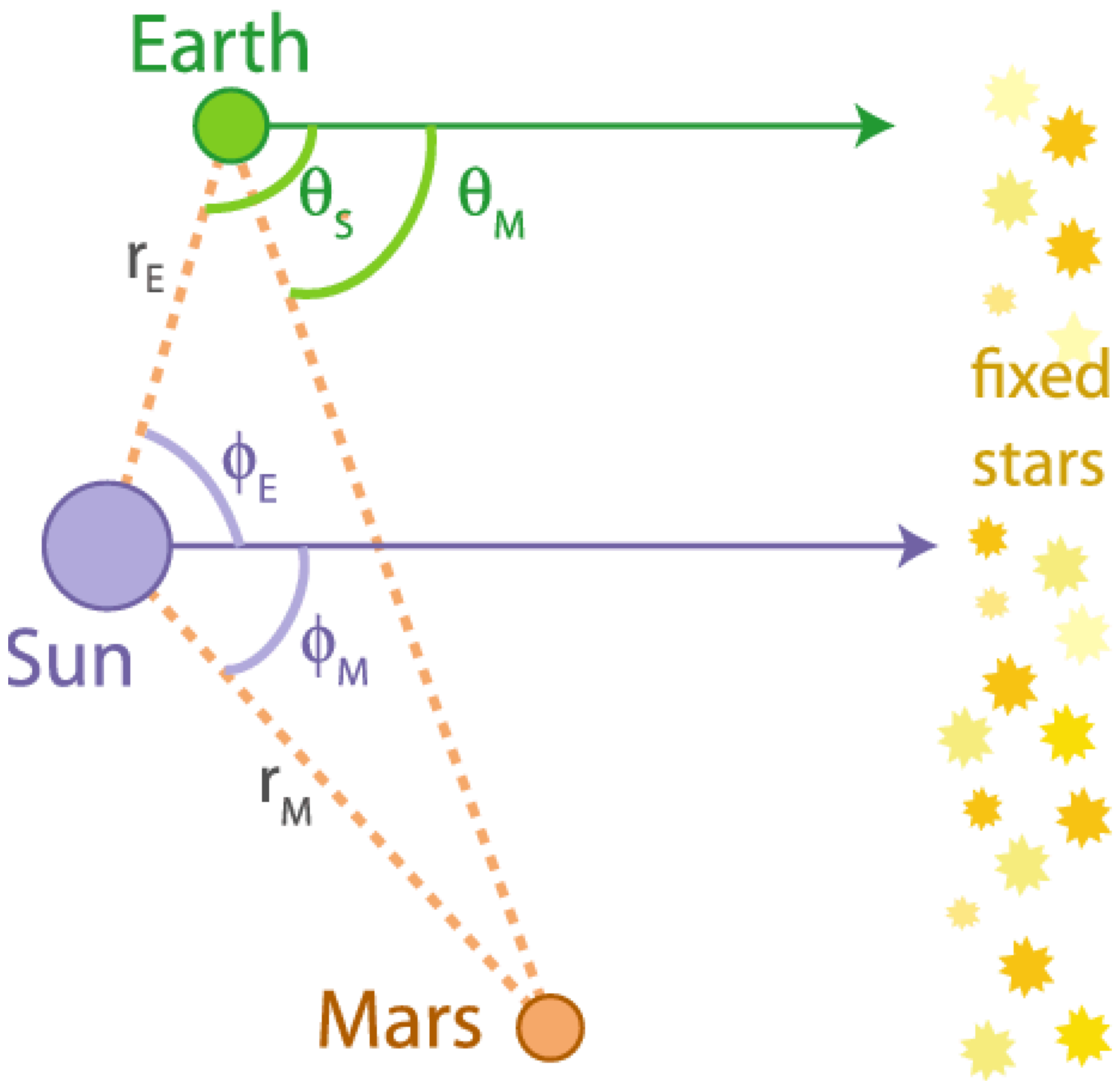
$$J = m_{rot}r^2\omega - rm_{free}(\vec{v}_{free})_x = m_{rot}r^2\omega' - rm(\vec{v}'_{free})_x = J'$$

$$\begin{cases} o = [(t_{rot}, \vec{q}(t_{rot})), (t_{free}, \vec{q}(t_{free}))] \\ q = (t'_{pred}, \vec{q'}_{free}(t_{pred})) \\ a_{cor} = \vec{q}_{rot}(t'_{pred}) \end{cases}$$

→



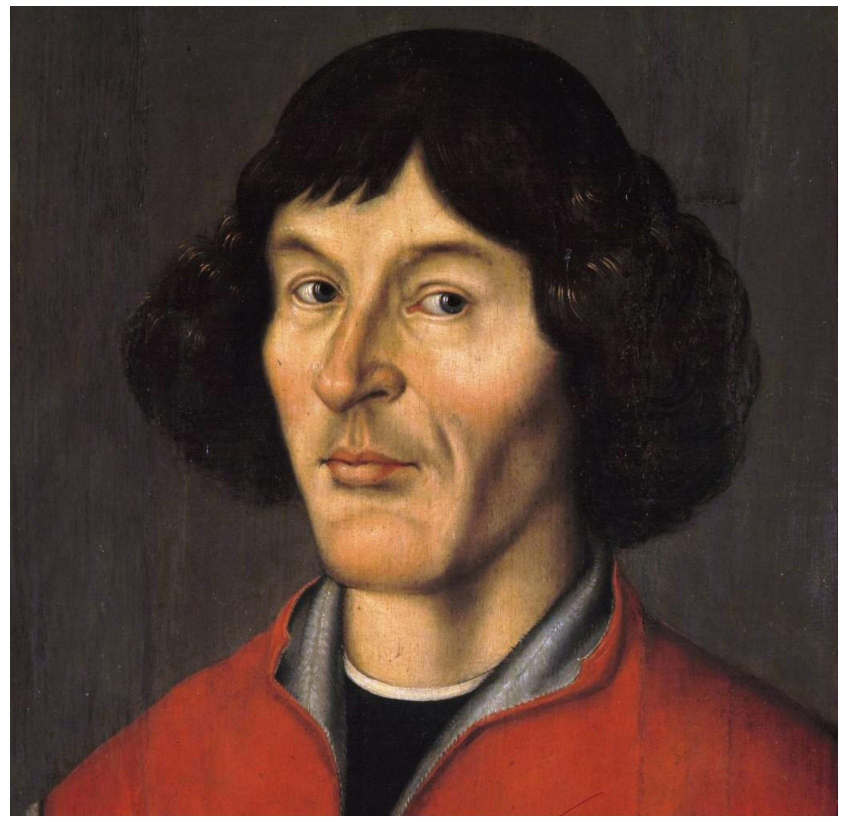
Examples: recovering of physical models [1][2]



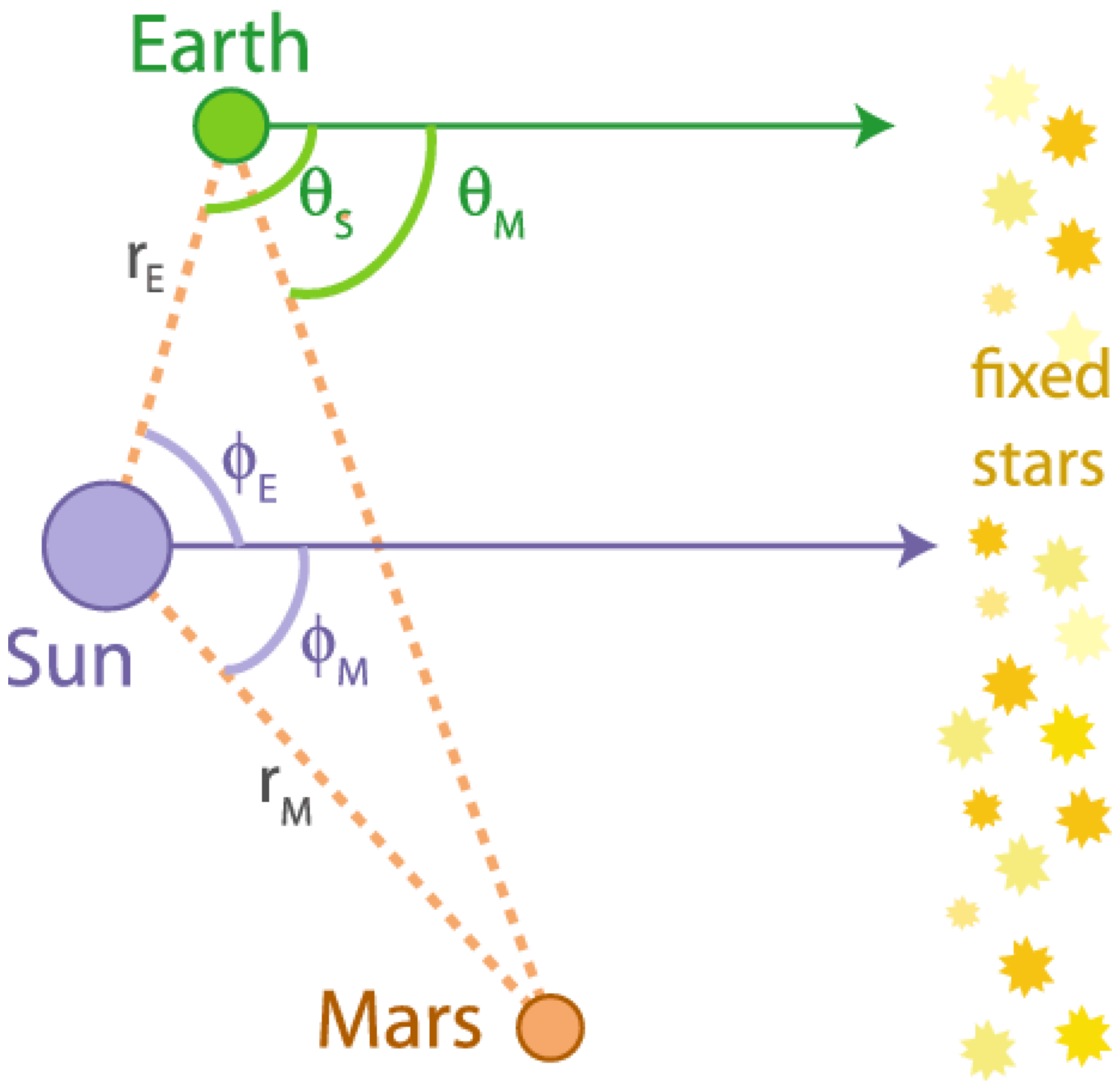
$$\begin{cases} o = (\theta_M(t_0), \theta_S(t_0)) \\ q = t'_{pred} \\ a_{corr} = (\theta_M(t'_{pred}), \theta_S(t'_{pred})) \end{cases}$$

Heliocentric Model

The orbit of Mars seems very complicated as seen from Earth … And if it was the Sun at the center?



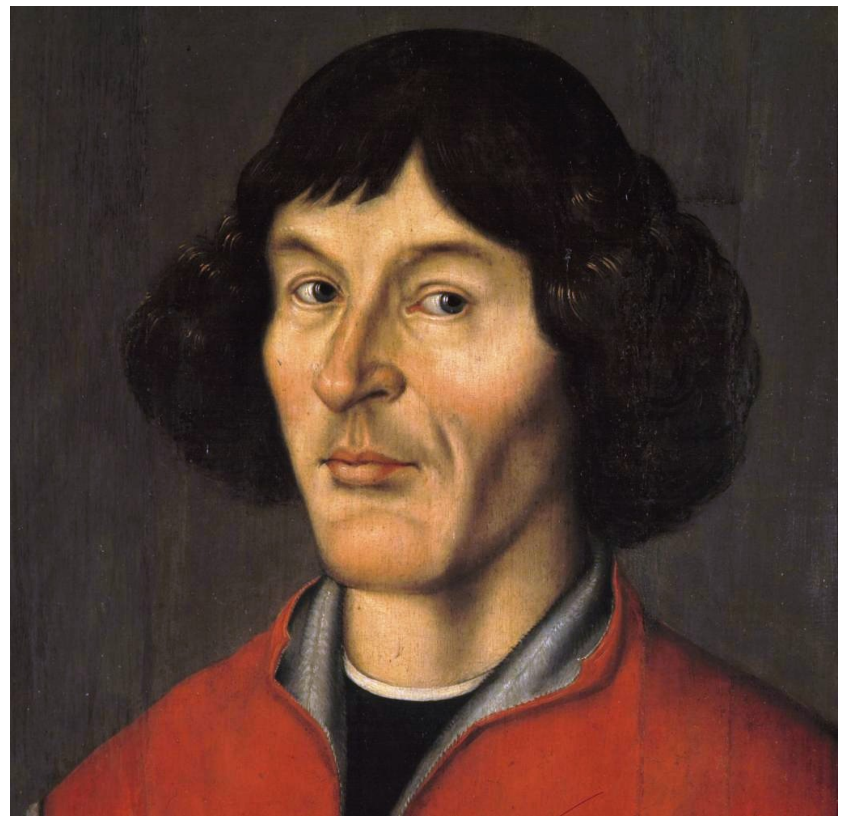
Examples: recovering of physical models [1][2]

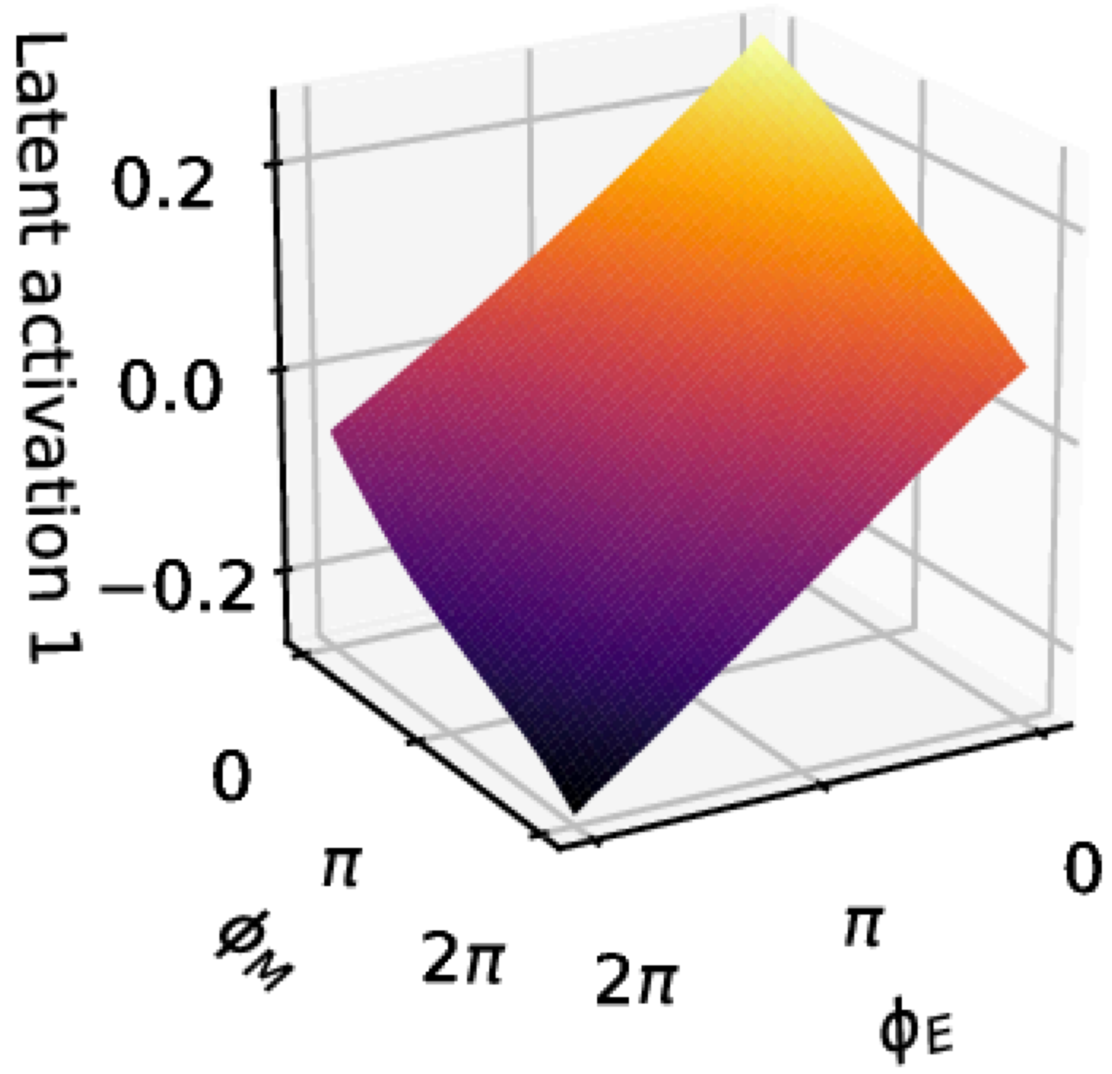


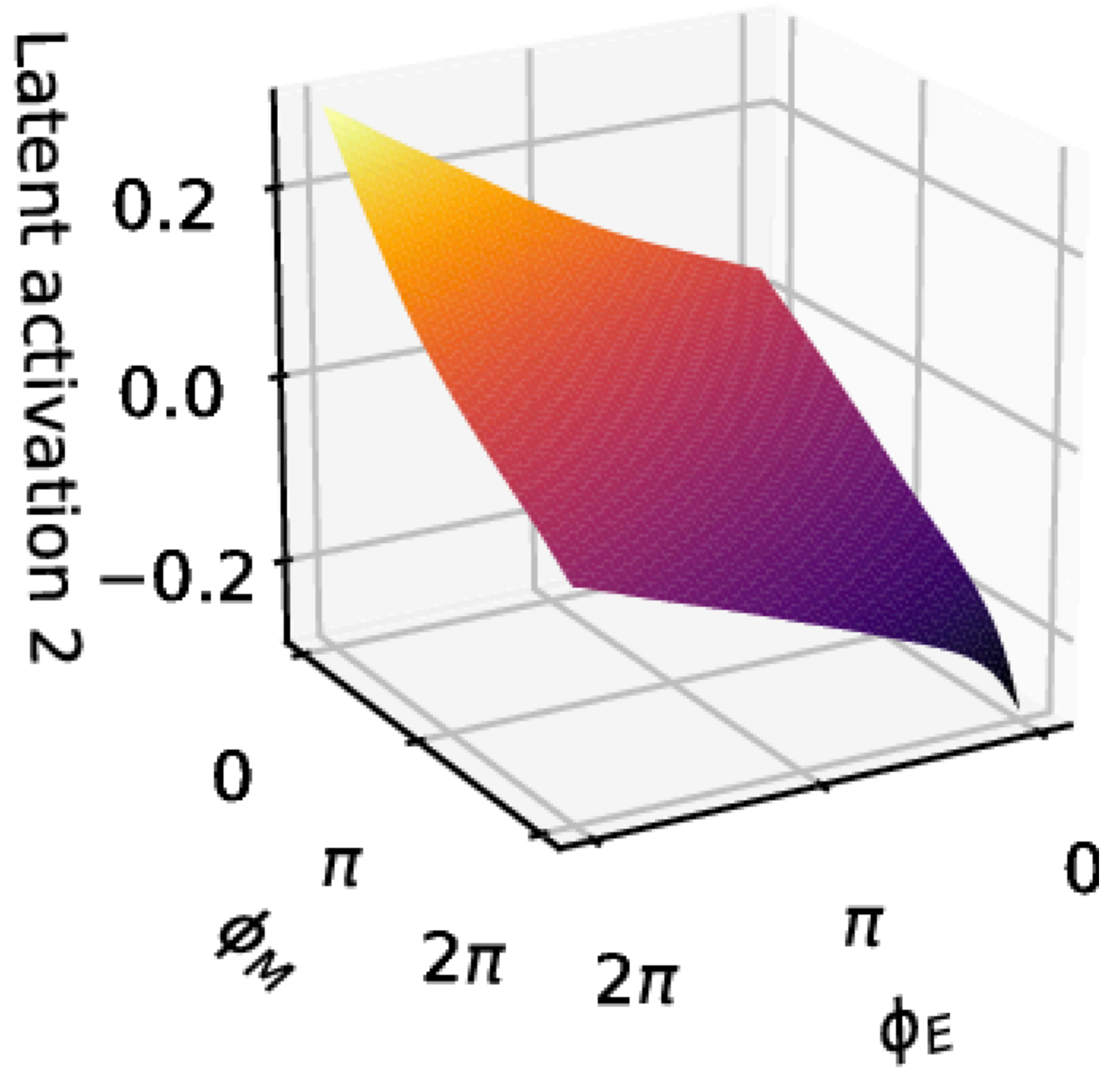
$$\begin{cases} o = (\theta_M(t_0), \theta_S(t_0)) \\ q = t'_{pred} \\ a_{corr} = (\theta_M(t'_{pred}), \theta_S(t'_{pred})) \end{cases}$$

Heliocentric Model

The orbit of Mars seems very complicated as seen from Earth … And if it was the Sun at the center?







Conclusions

What we have LEARNED so far?

- ❖ There exists a way to reproduce well the way of reasoning of a physicist and it's awesome that this model can be expressed as a (complicated!) algorithm

- ❖ Maybe the answer to the initial question is "yes" since surely the network didn't attend any lessons of physics

What are the next studies in this sector?

- ❖ Here the interpretation of latent variables is simple because the model of the analyzed physical system already exists. But, what about use this approach to new physical settings which have not a model describing them?

References

1. *Discovering physical concepts with neural networks:*
   Raban Iten, Tony Metger, Henrik Wilming, Lidia del Rio, Renato Renner [1]
   https://arxiv.org/abs/1807.10300

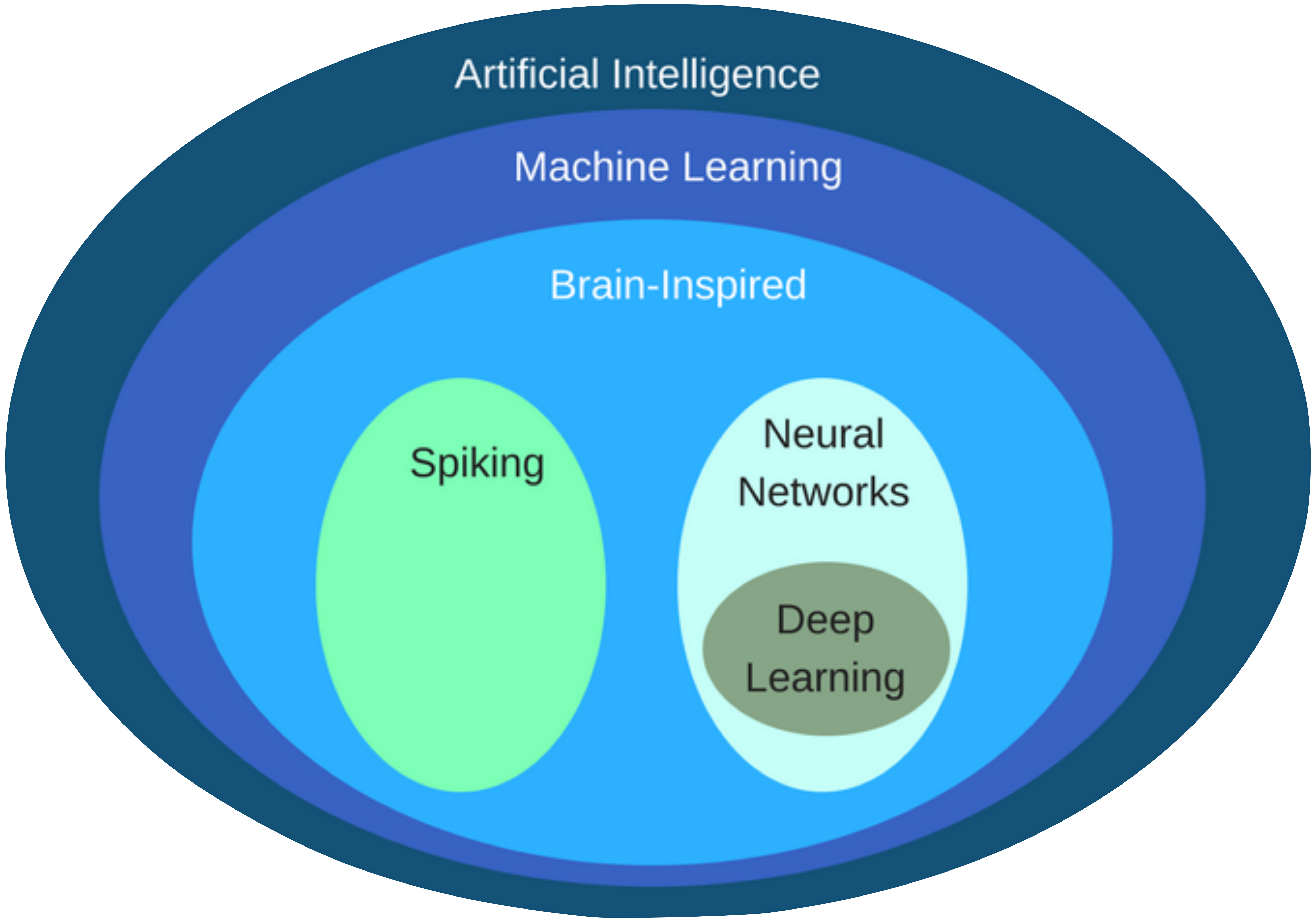
2. Git repository of the *SciNet* project:
   https://github.com/eth-nn-physics/nn_physical_concepts [2]

3. Neural Networks and DeepLearning:
   http://neuralnetworksanddeeplearning.com/index.html

4. *Who needs Copernicus if you have machine learning?*
   https://www.technologyreview.com/s/611798/who-needs-copernicus-if-you-have-machine-learning/

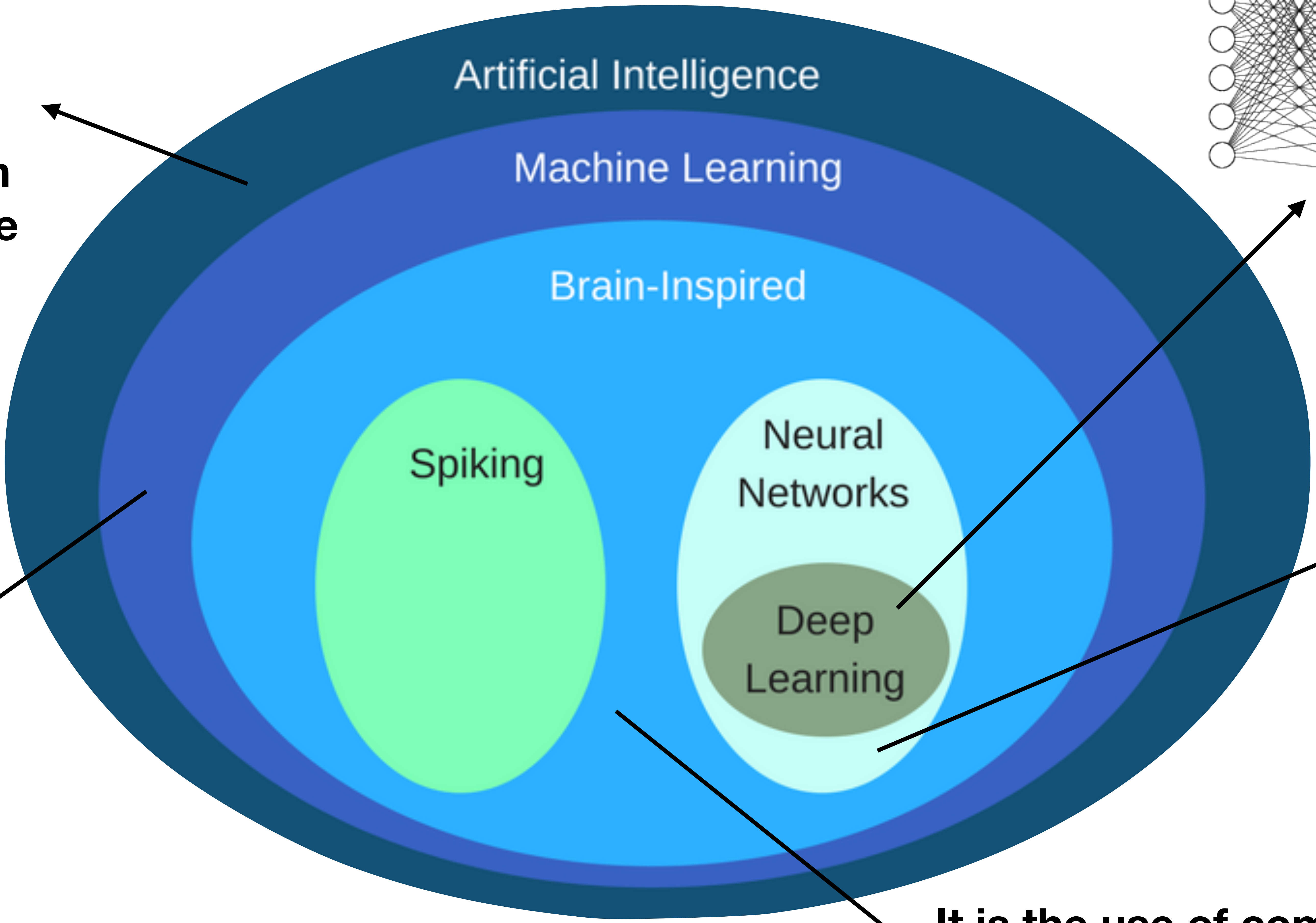
Spares

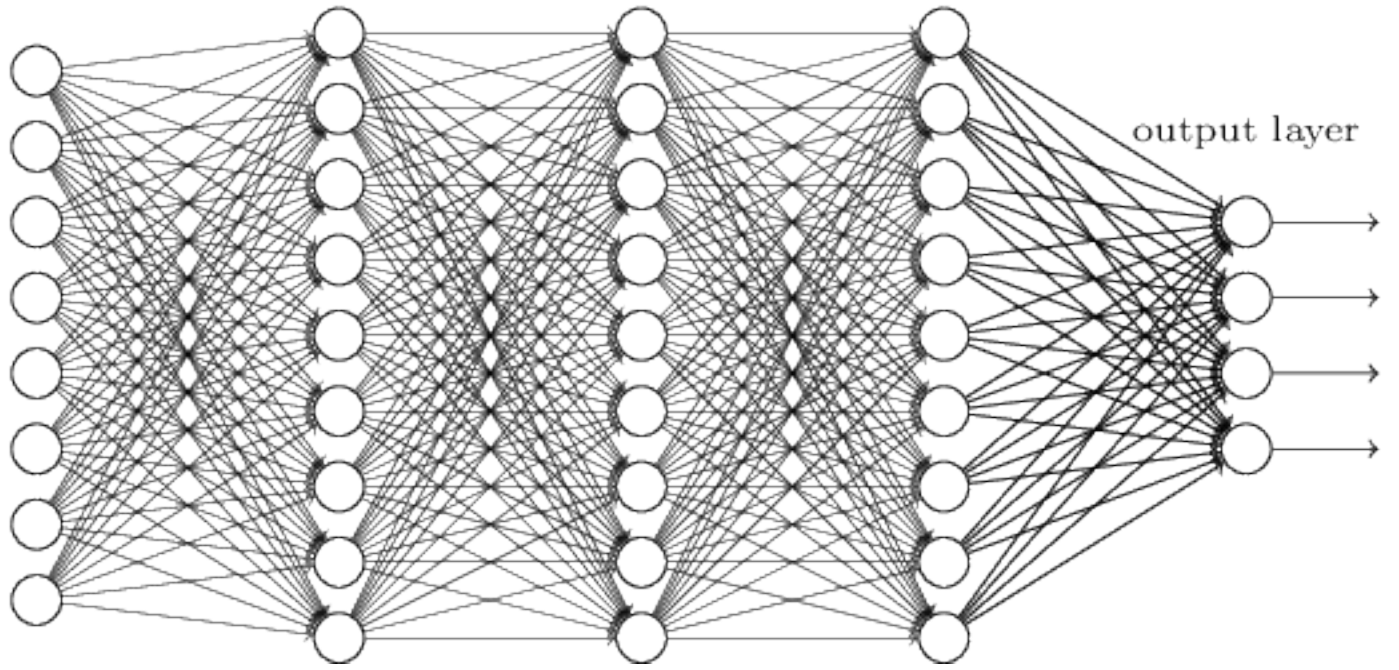
Structure of the AI field

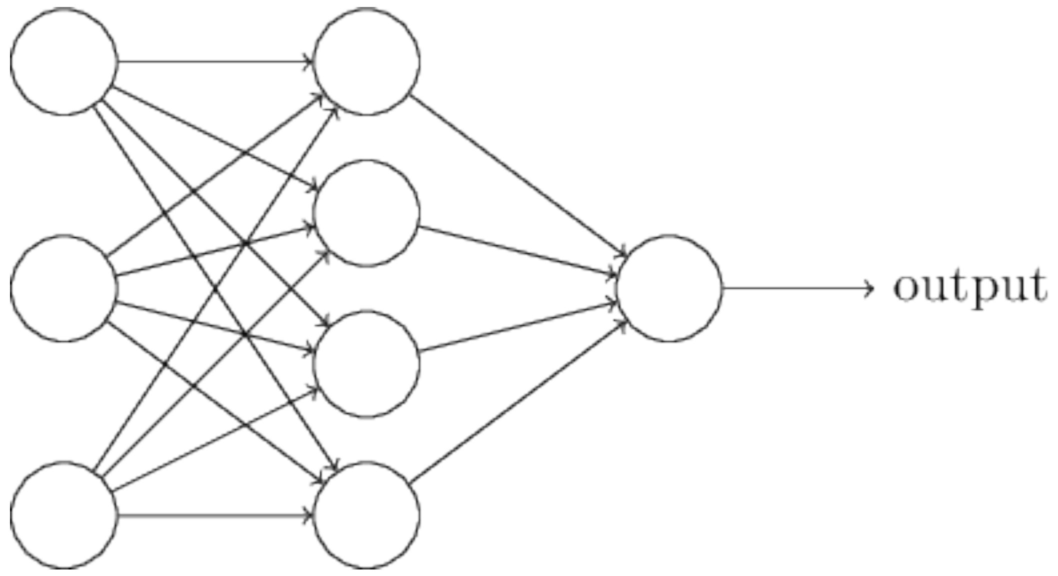


Structure of the AI field

A.I. is a field of research studying the hardware and software implementations through which a machine can develop similar features of the human reasoning







Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead

It is the use of computers to model the living phenomena

Activation functions of *Scinet*

$$f(z) = \begin{cases} z, \text{if } \ z > 0 \\ \alpha(e^z - 1), \text{if } \ z \leq 0 \end{cases}$$