Understanding Deep Neural Networks

Chapter Two

Network Structure

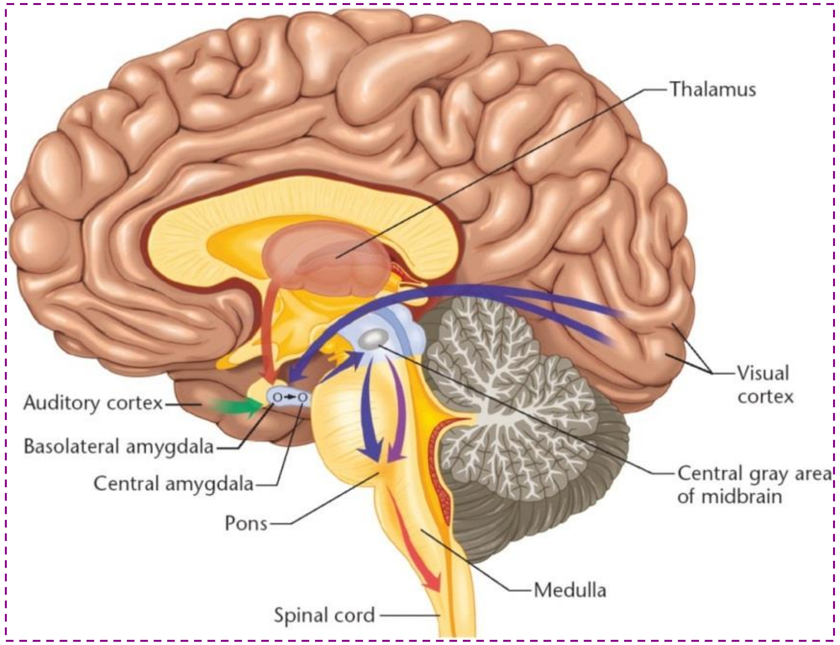
Zhang Yi, *IEEE Fellow*
Autumn 2019

Outline

- Brief Review of Brain Structure
- Computational Model of Neurons
- Computational Model of Neural Networks
- Continuous Time Neural Networks
- Assignments

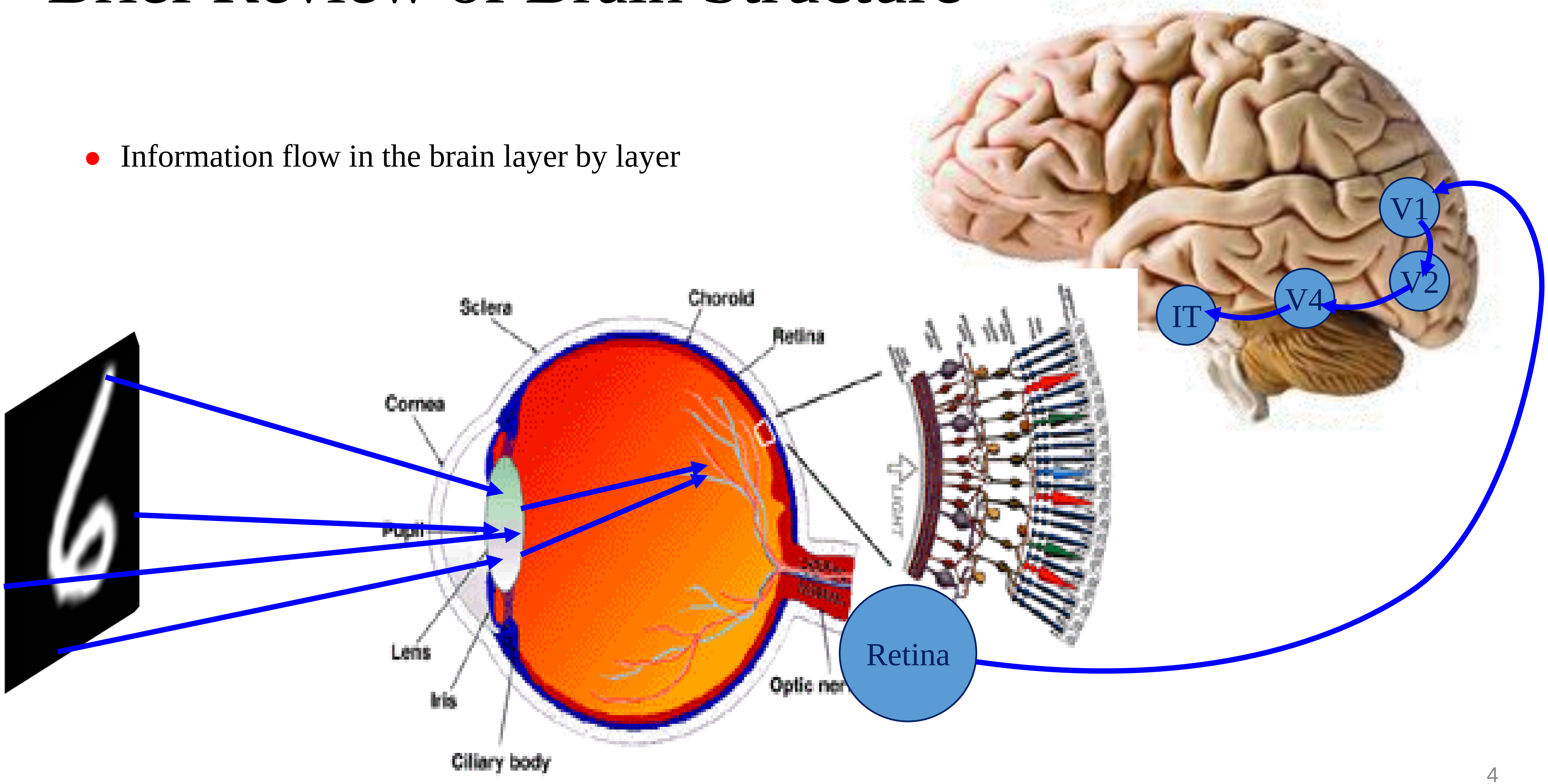
Brief Review of Brain Structure

- A brain contains about 10^{11} neurons
- Each neuron has about 10^4 connections

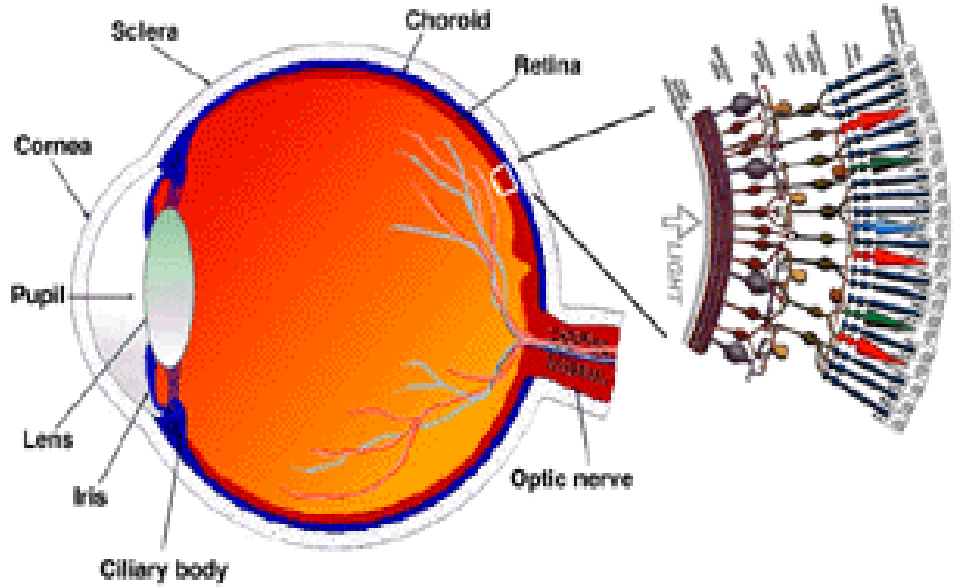


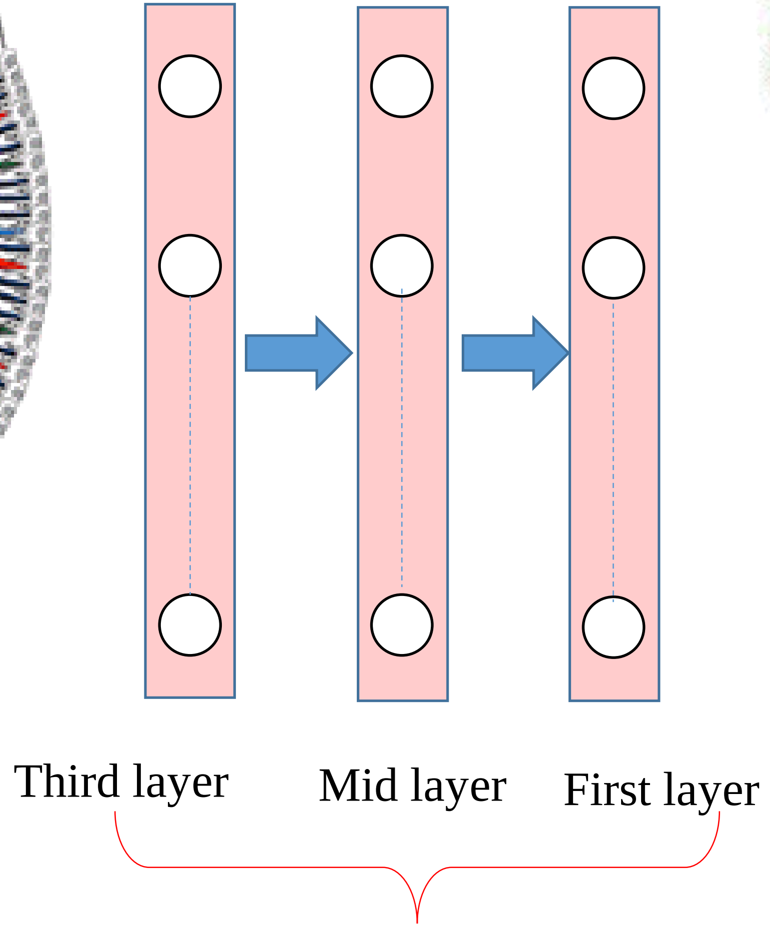
Brief Review of Brain Structure

- Information flow in the brain layer by layer

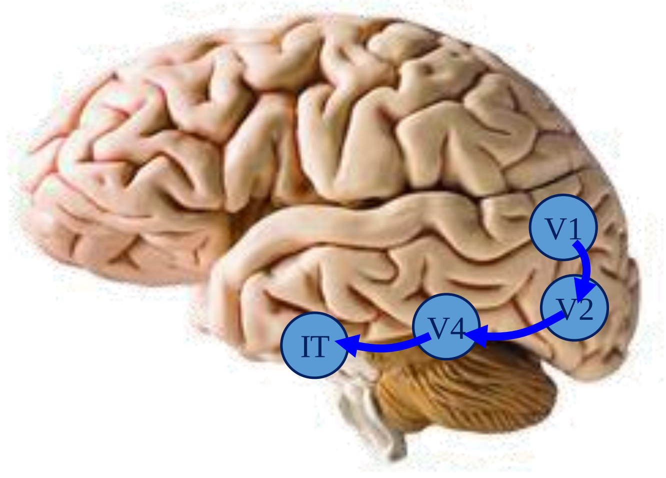


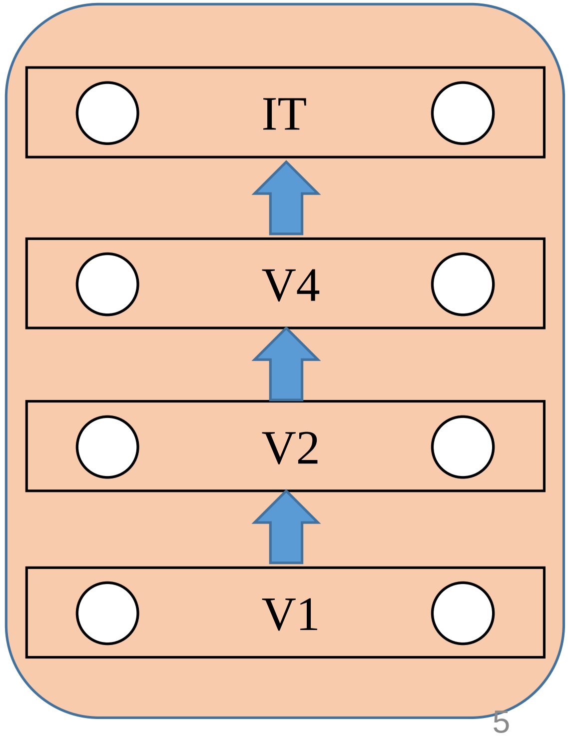
Brief Review of Brain Structure





Concept of Layers



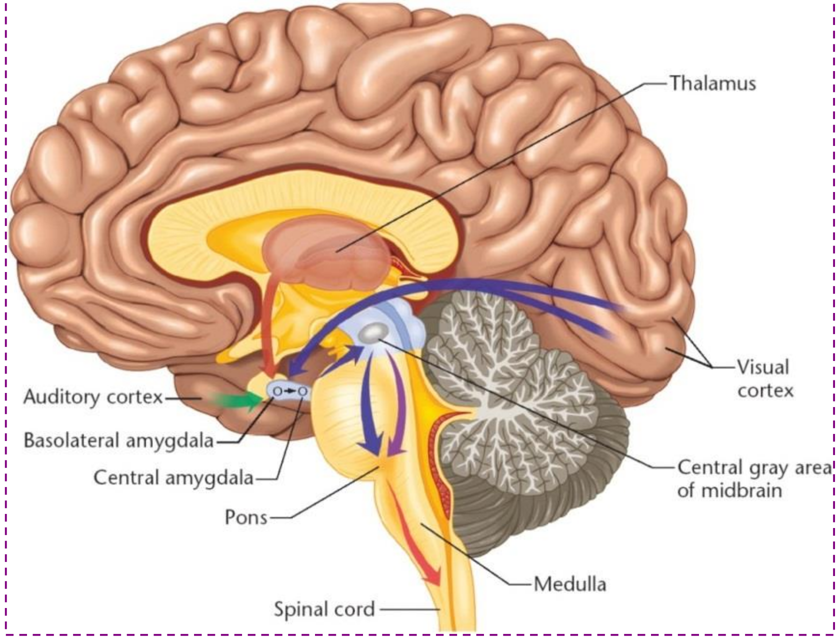


Concept of Layer

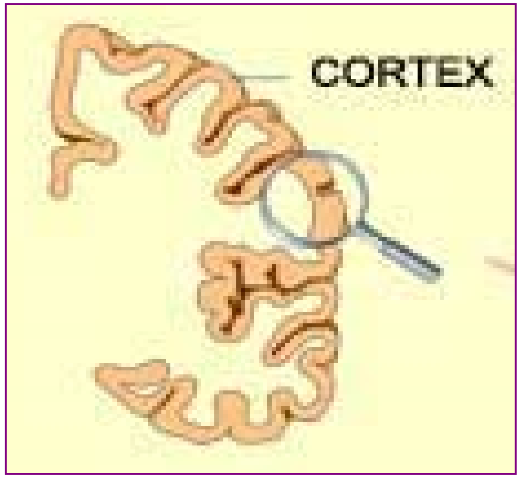
1. Neural network with layers
2. Neurons receive the outputs of neurons at previous layer as inputs.

Brief Review of Brain Structure

- The typical human neocortex:
  - $1000cm^2$
  - Stretched flat, the human neocortical sheet is roughly the size of a large dinner napkin.
  - 2mm thick
  - 30 billion neurons
  - A tiny square millimeter contains an estimated 100,000 neurons.
  - 100 trillion synapses.



Almost everything related to intelligence such as: perception, language, imagination, mathematics, art, music, and planning— occurs on the neocortex.

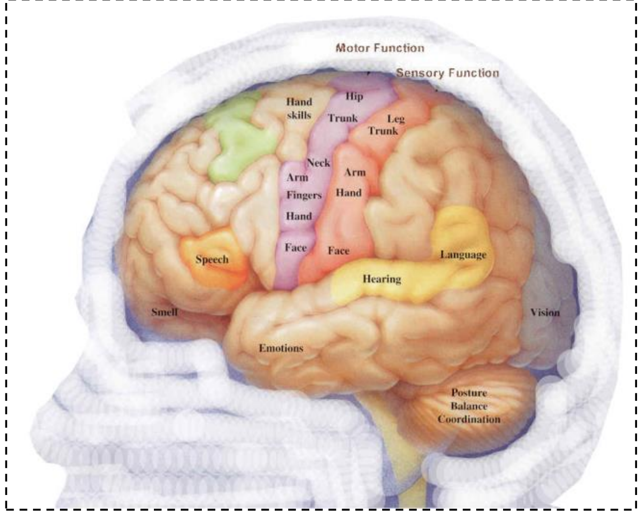


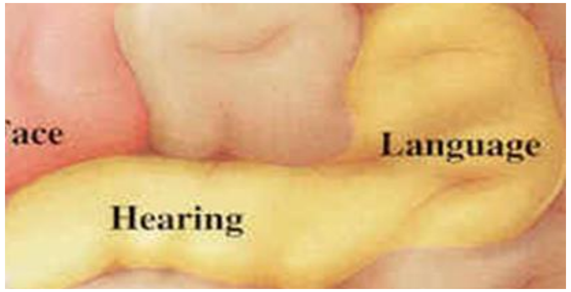
Brief Review of Brain Structure

- A neocortex is divided into several functional regions, such as visual area, auditory area, touch area, associate area, etc..
  - The functional regions are arranged in an irregular patchwork quilt physically.
  - Nearly identical architecture.

Cortex map

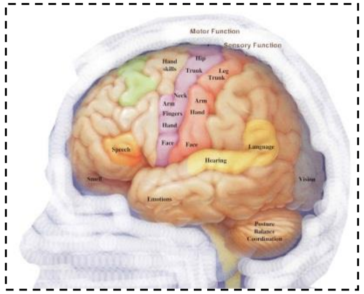
Concept of regions

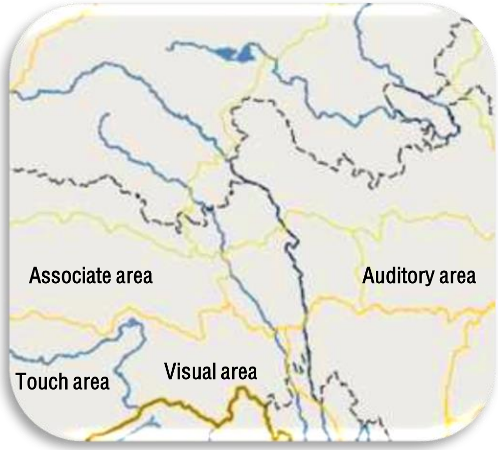


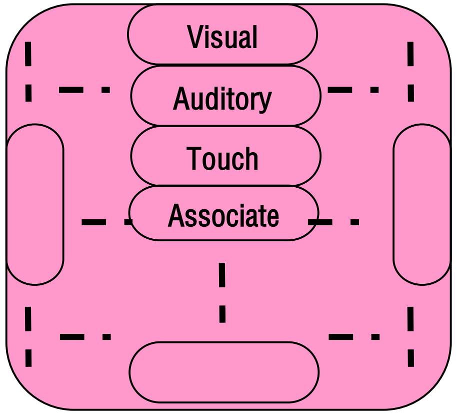


Brief Review of Brain Structure

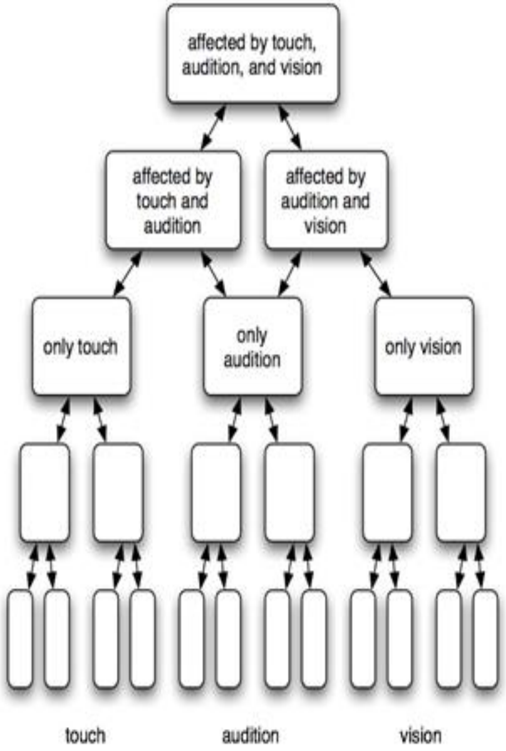
- How are the regions connected?
  - Functionally the regions are arranged in a branching hierarchy.
  - Lower regions feed information up to higher regions.
  - Higher regions send feedback down to lower regions.





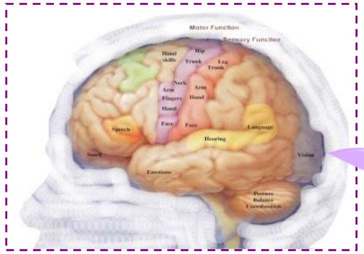


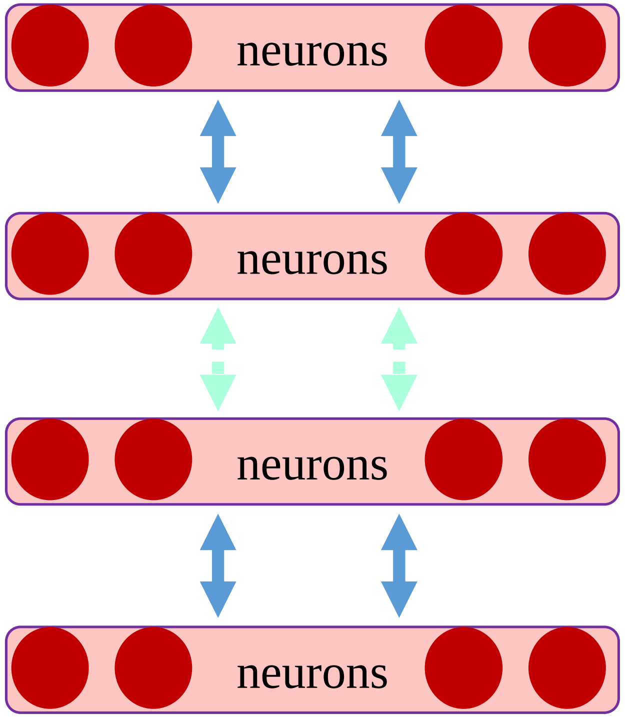
Concept of Hierarchy Connection

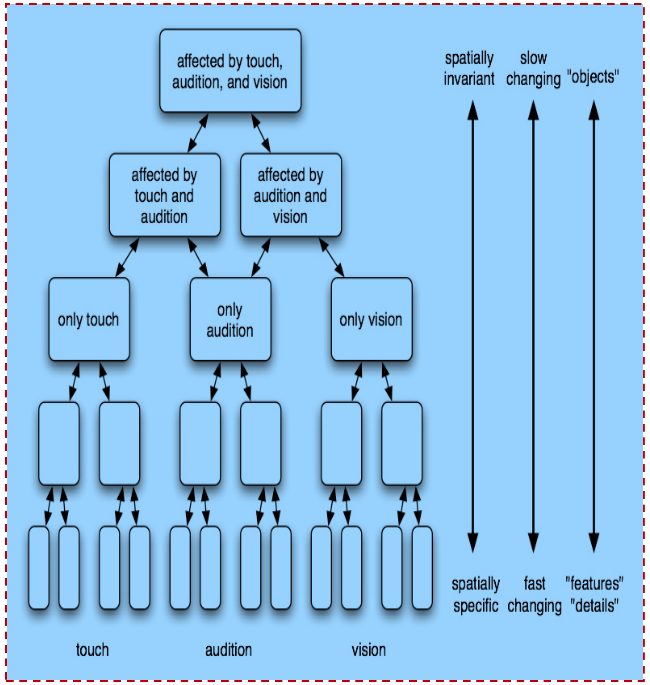


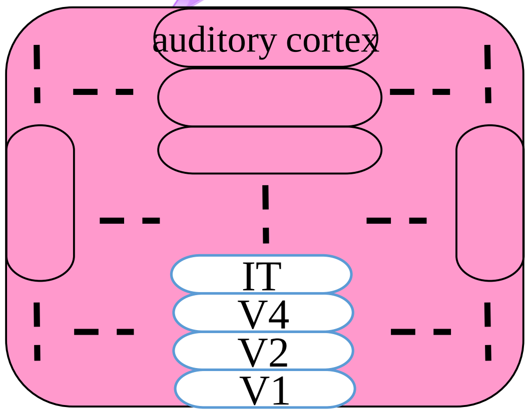
Brief Review of Brain Structure

Physically: irregular quilt.
Functionally: hierarchy.

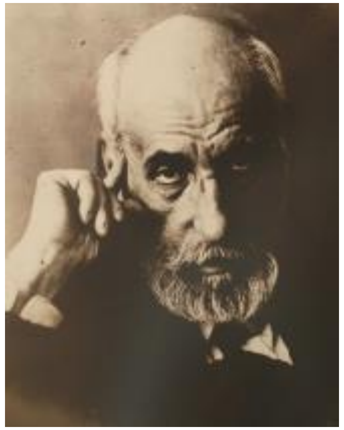


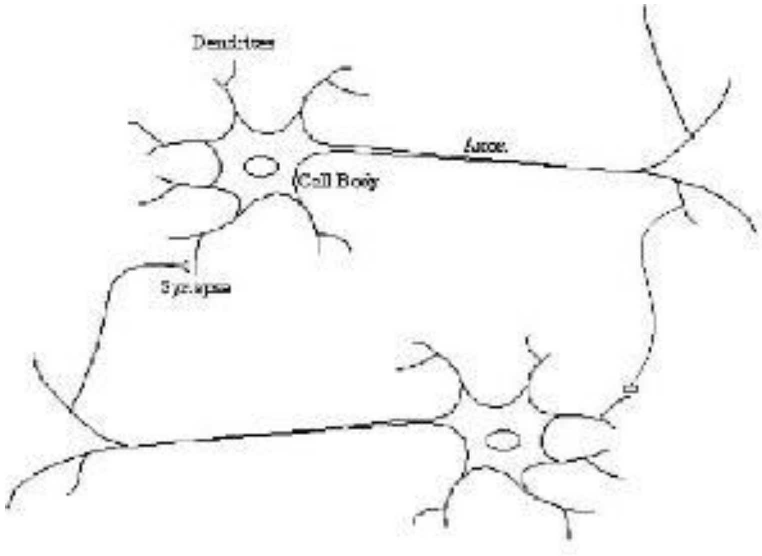


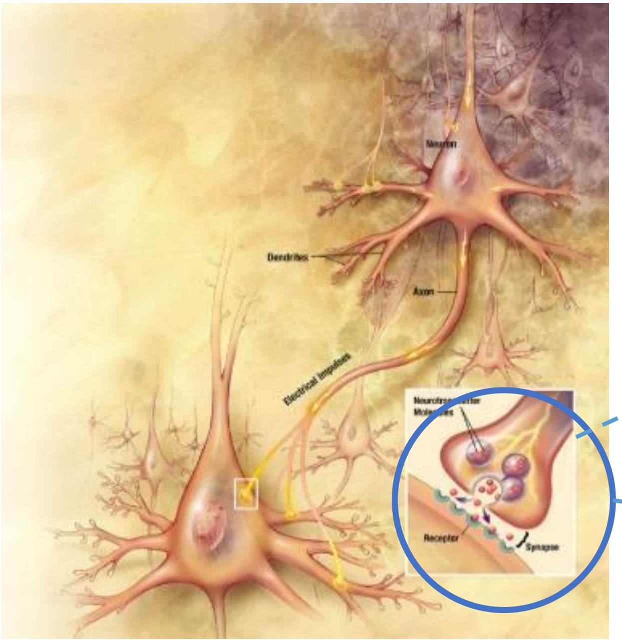




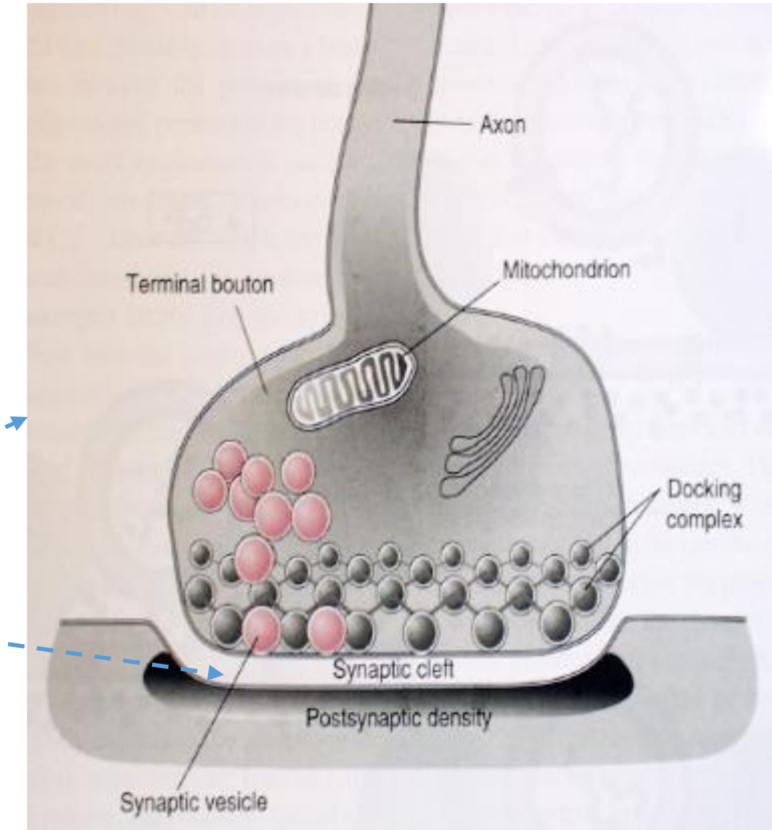
Brief Review of Brain Structure



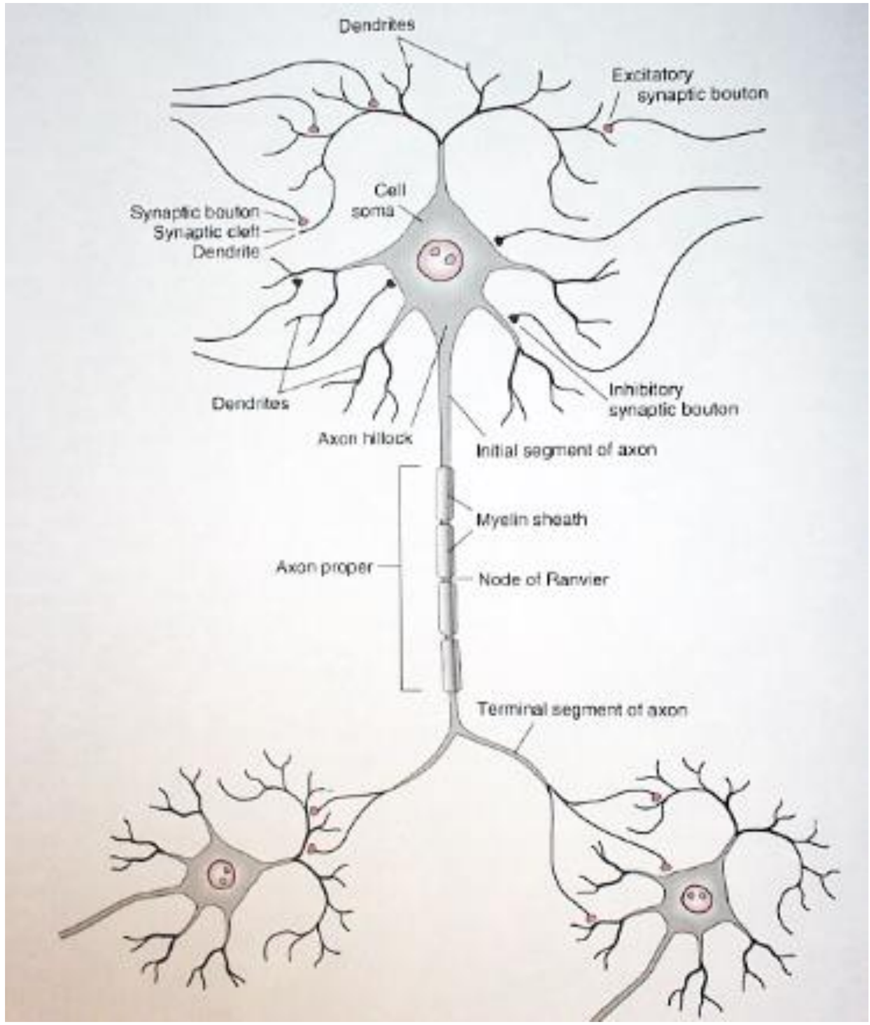




Synapse



Brief Review of Brain Structure



Neural Network = Neurons + Connections

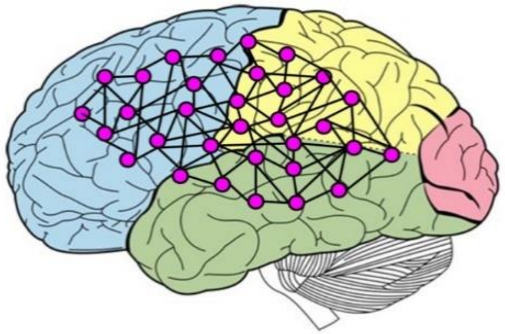
The information flow in the network by some kind of electricity.

Problem: Can we develop computational models for the neural network?

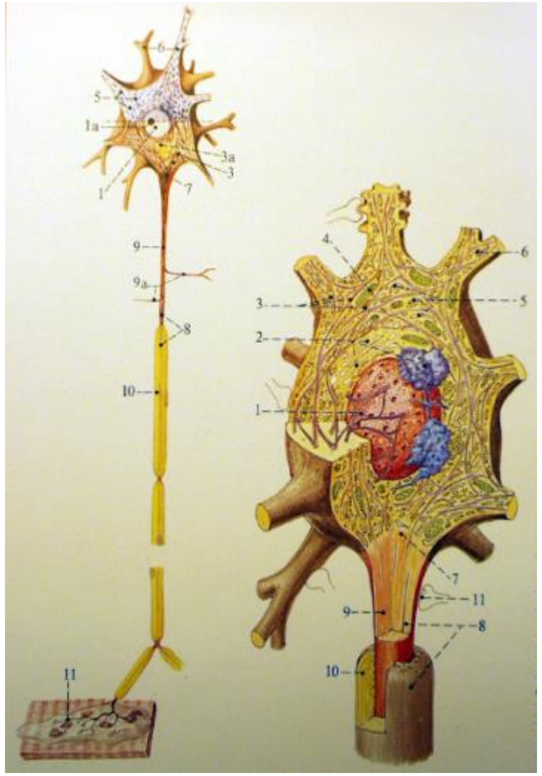
Outline

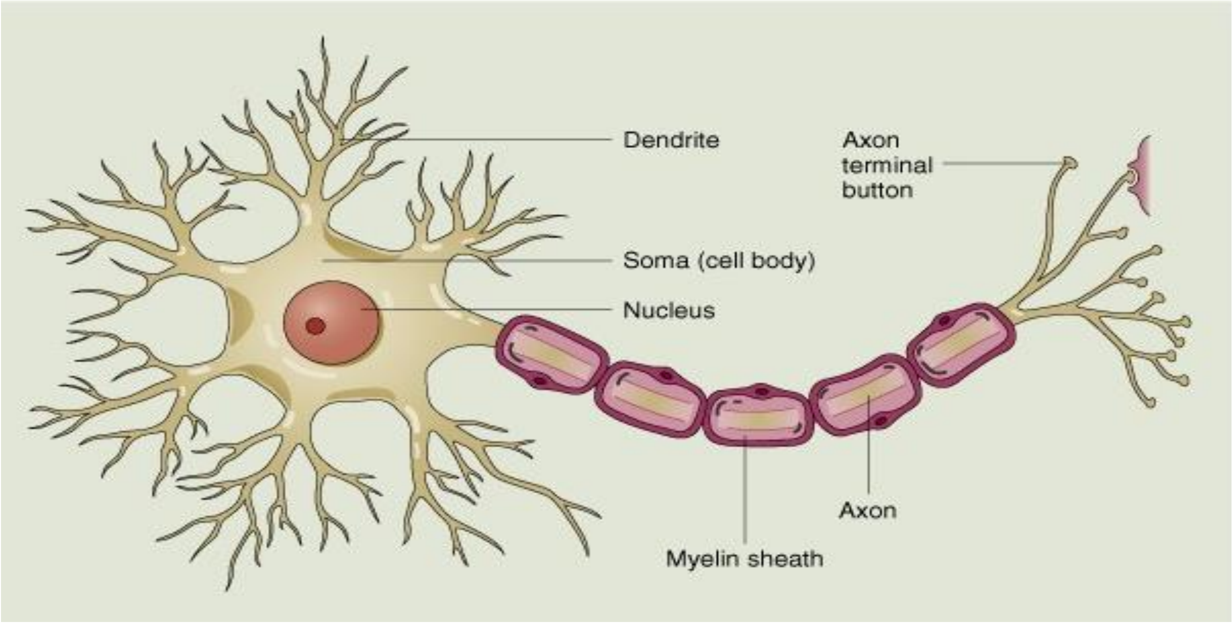
- Brief Review of Brain Structure
- Computational Model of Neurons
- Computational Model of Neural Networks
- Continuous Time Neural Networks
- Assignments

Computational Model of Neurons

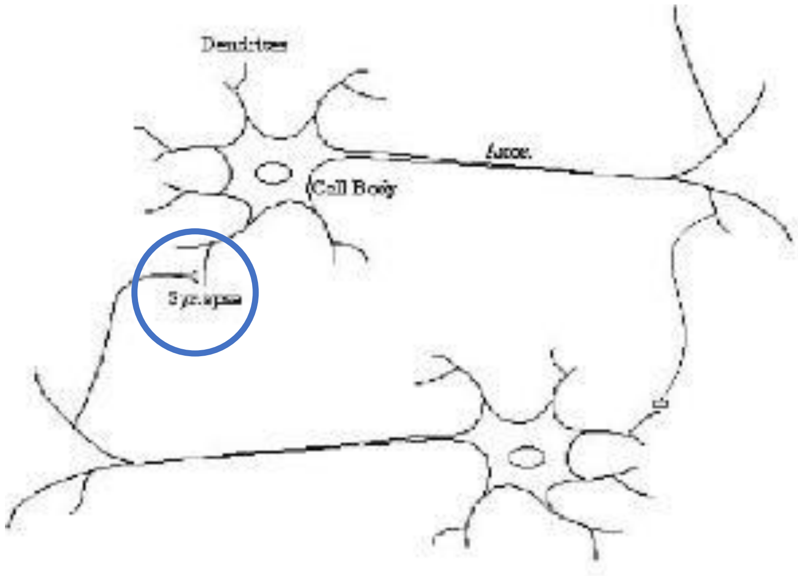


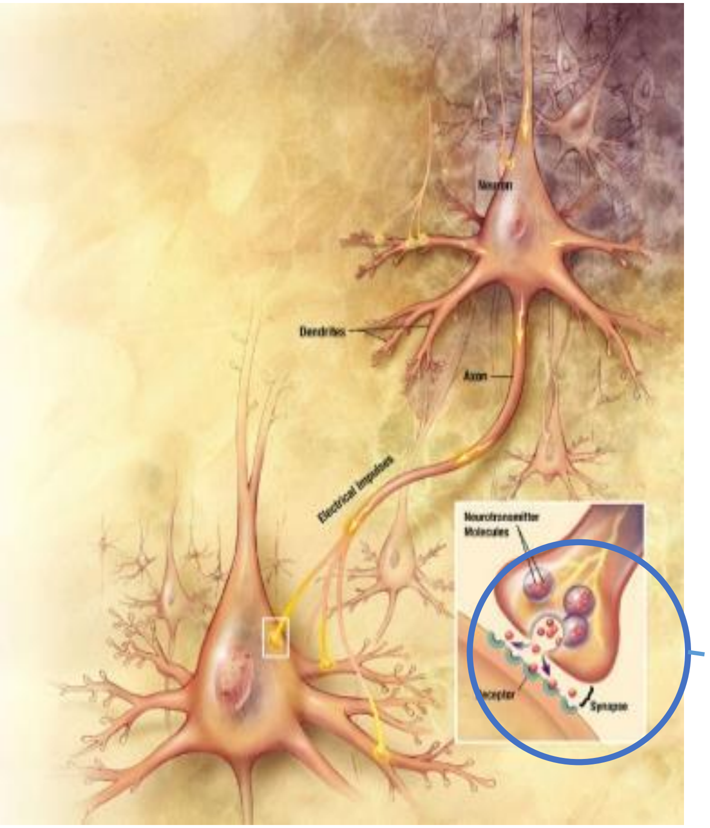
- Basic Components
  - Soma (cell body)
  - Dendrite
  - Axon
- Basic Functions
  - Collecting
  - Functioning
  - Transferring
- Characters
  - Multi-inputs
  - Mon-output

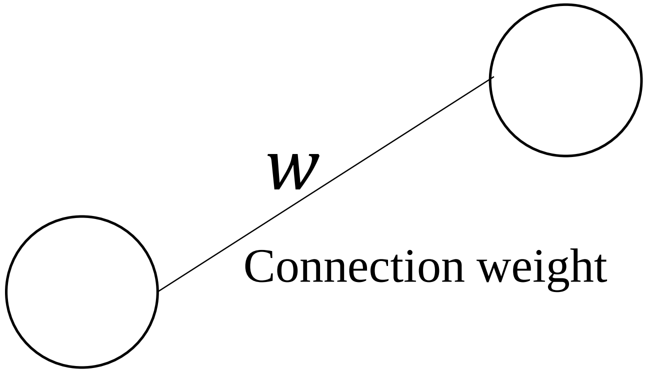




Computational Model of Neurons

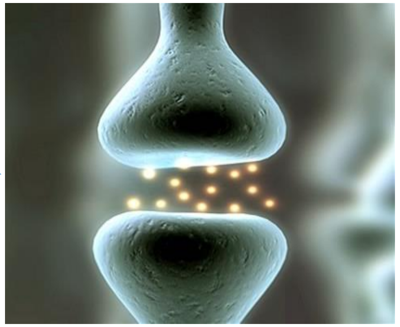




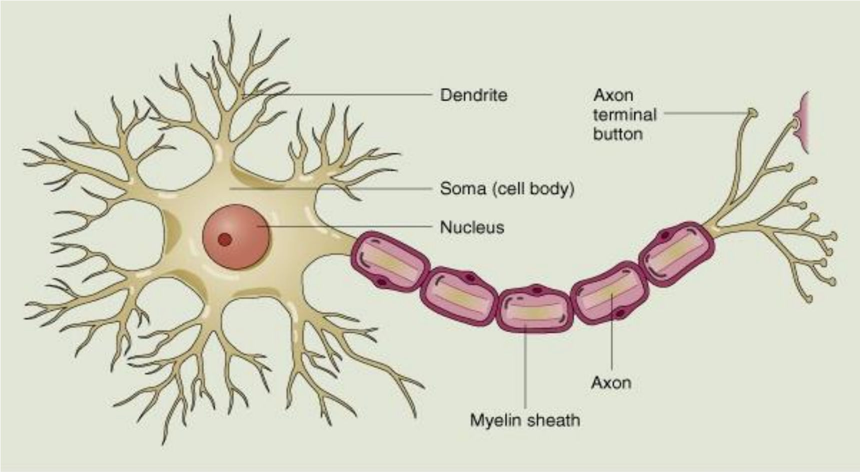


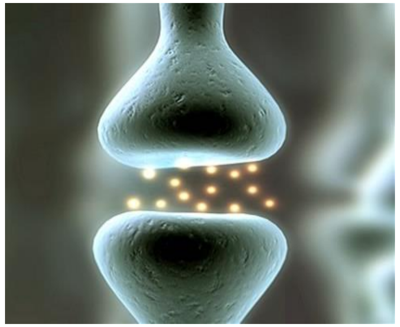
$w > 0, exciting\ connection$
$w < 0, inhibition\ connection$

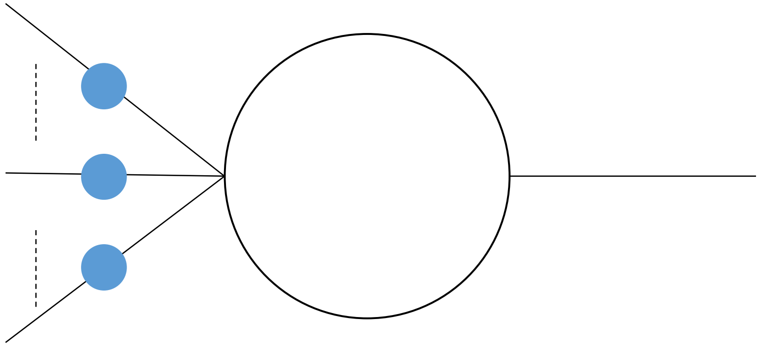
Synapse



Computational Model of Neurons

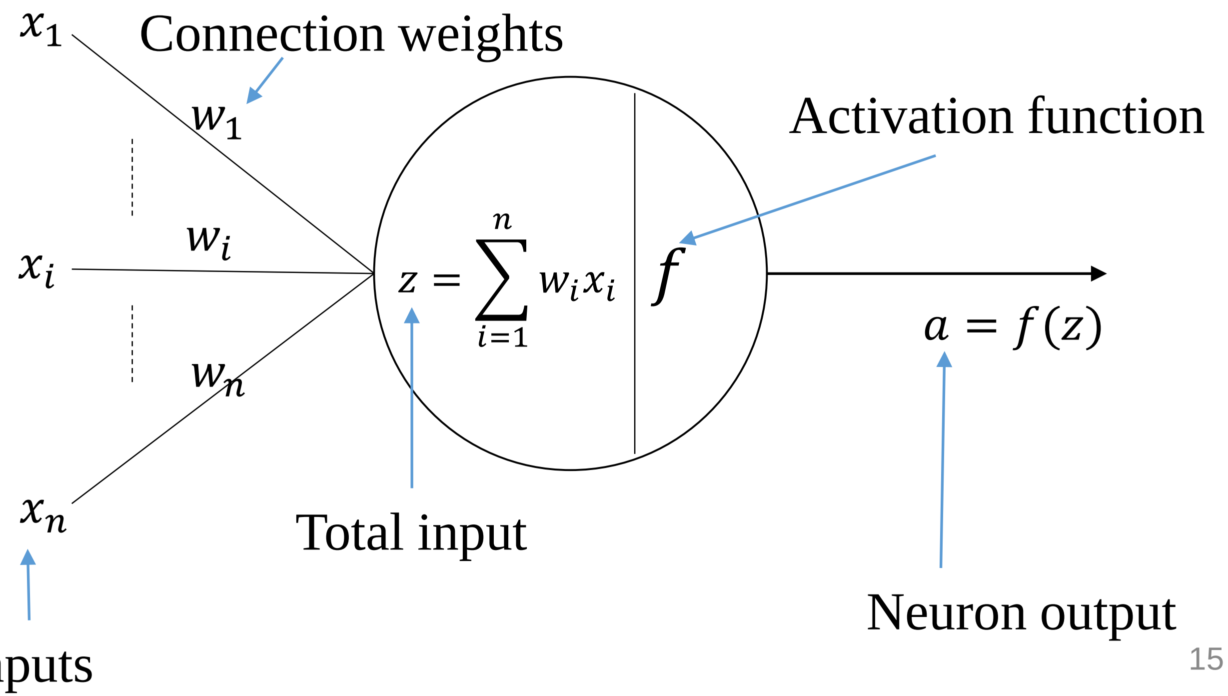






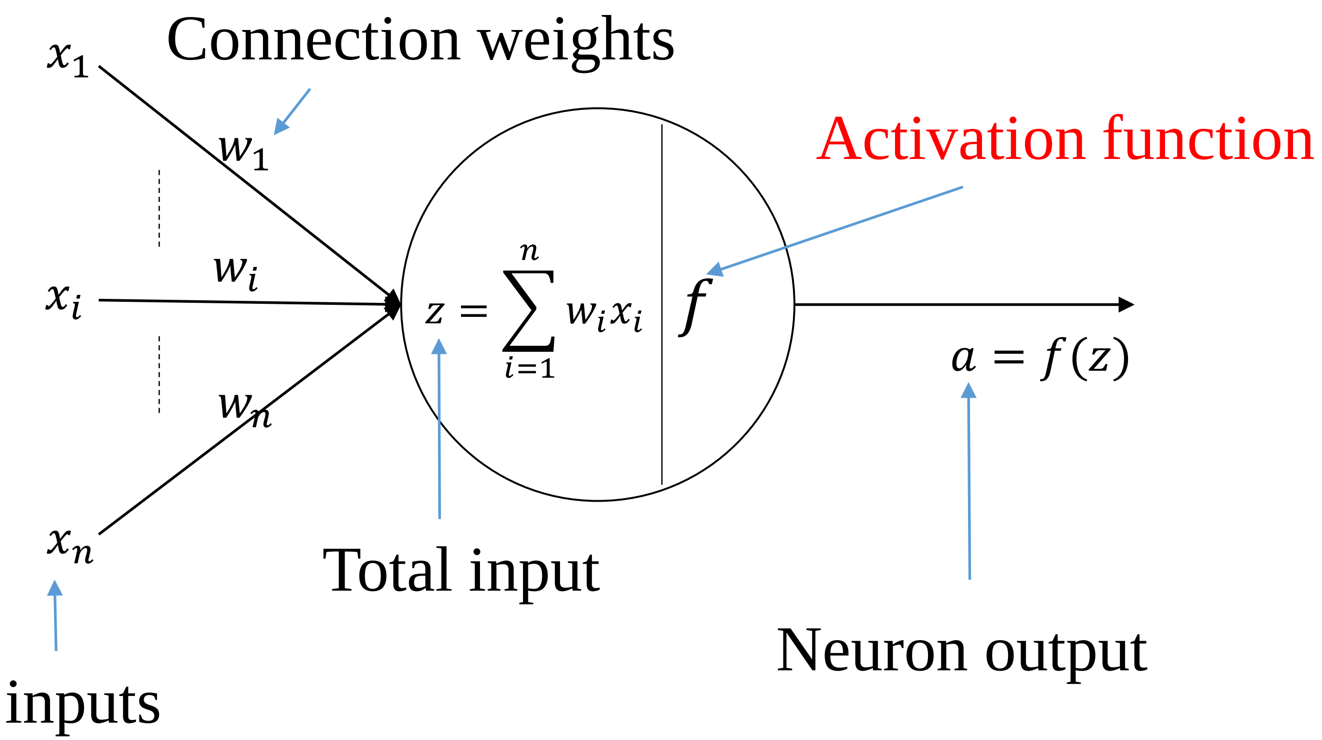
$$a = f\left(\sum_{i=1}^{n} w_i x_i\right)$$

Math model



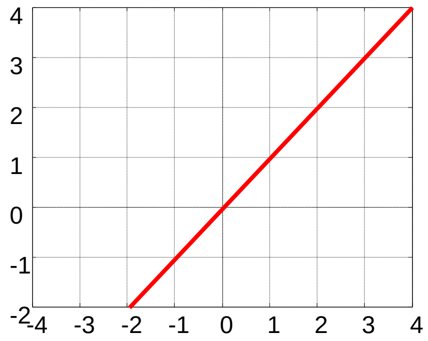
Computational Model of Neurons

$$a = f\left(\sum_{i=1}^{n} w_i x_i\right)$$



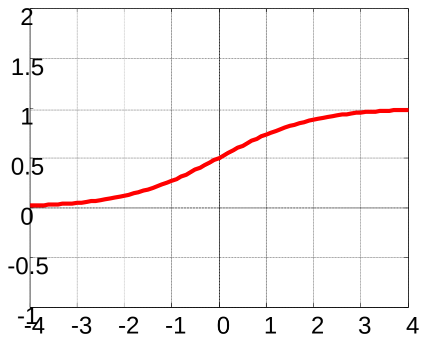
Linear function

$$f(z) = z$$



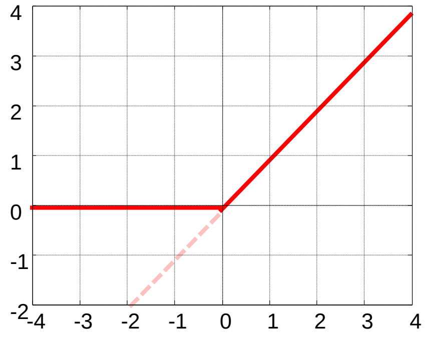
Sigmoid function

$$f(z) = \frac{1}{1 + e^{-z}}$$



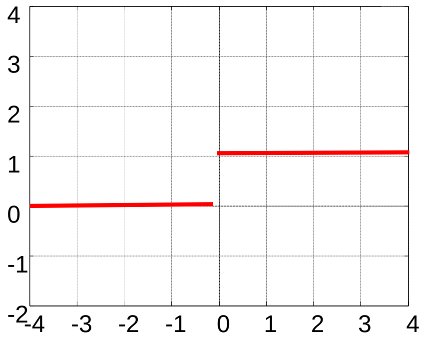
Rectifier function

$$f(z) = \begin{cases} z, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

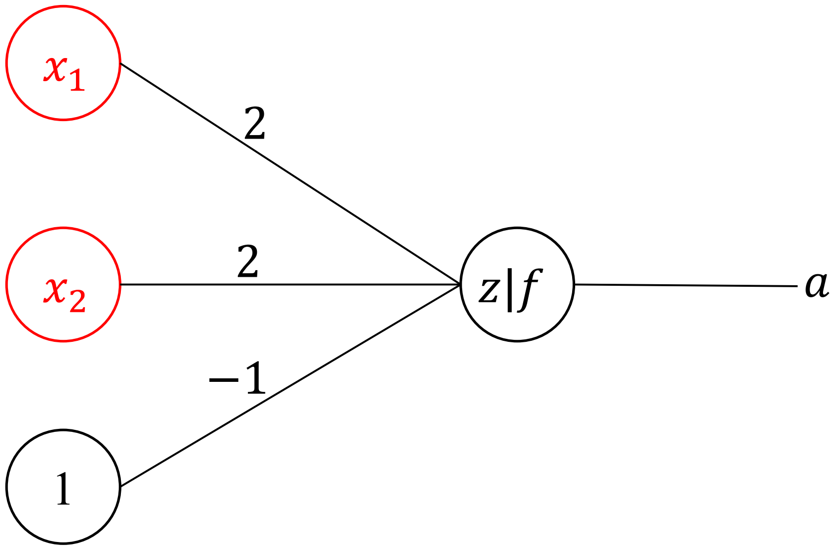


Hard-limit function

$$f(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$$



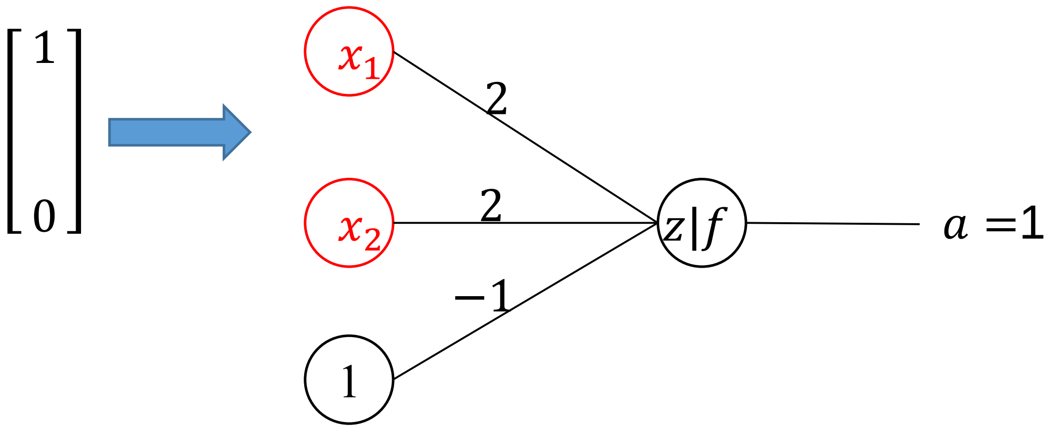
A Simple Example

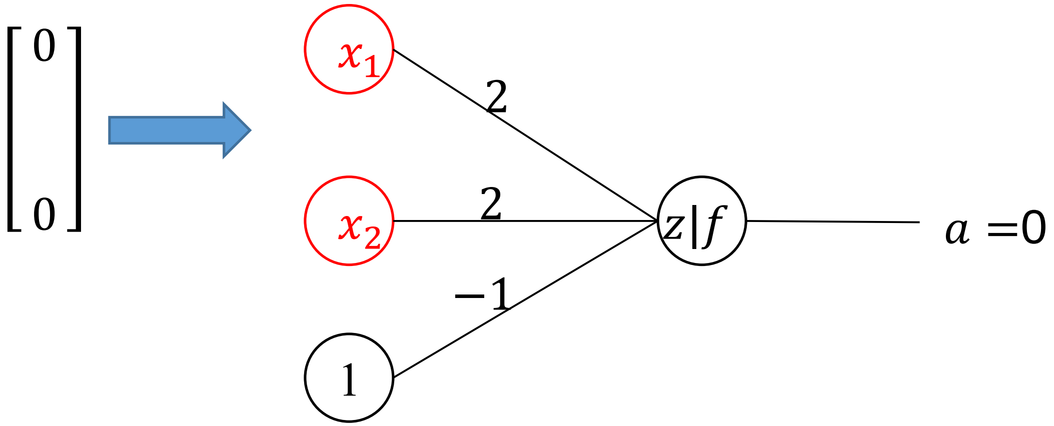


$$a = f(2x_1 + 2x_2 - 1)$$

$$z = 2x_1 + 2x_2 - 1$$

$$f(s) = \begin{cases} 1, & s \geq 0 \\ 0, & otherwise \end{cases}$$

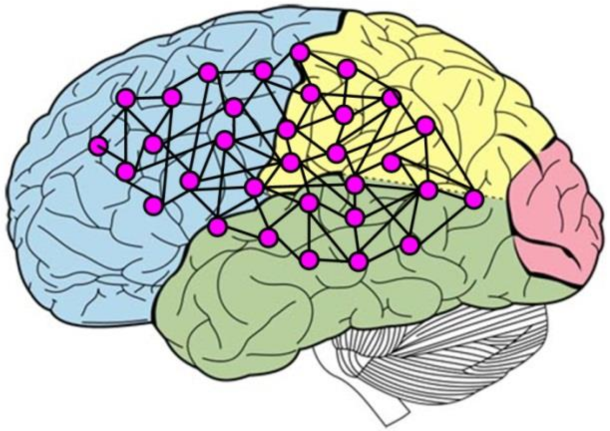


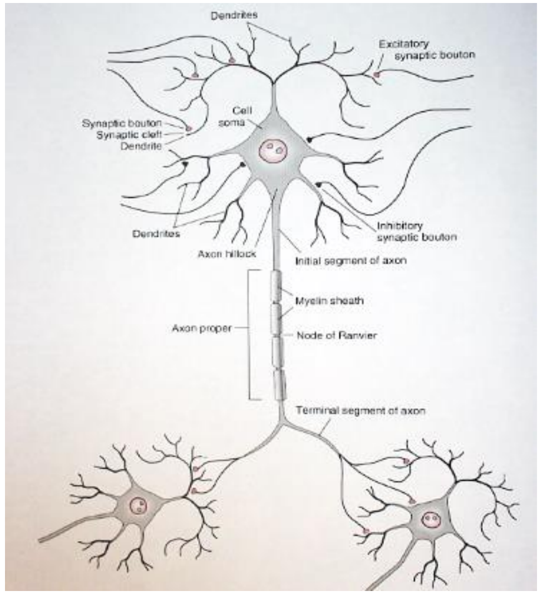


Outline

- Brief Review of Brain Structure
- Computational Model of Neurons
- Computational Model of Neural Networks
- Continuous Time Neural Networks
- Assignments

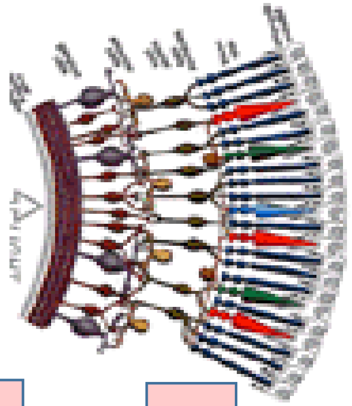
Computational Model of Neural Networks



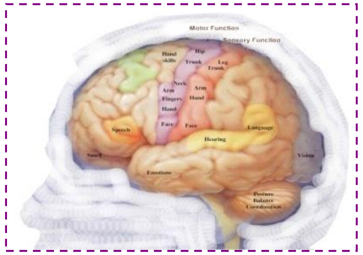


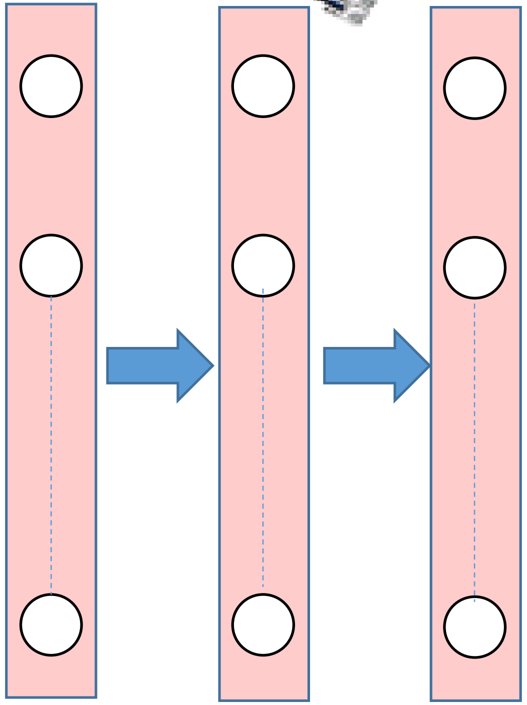
Neural Network = Neurons + Connections

Computational Model of Neural Networks

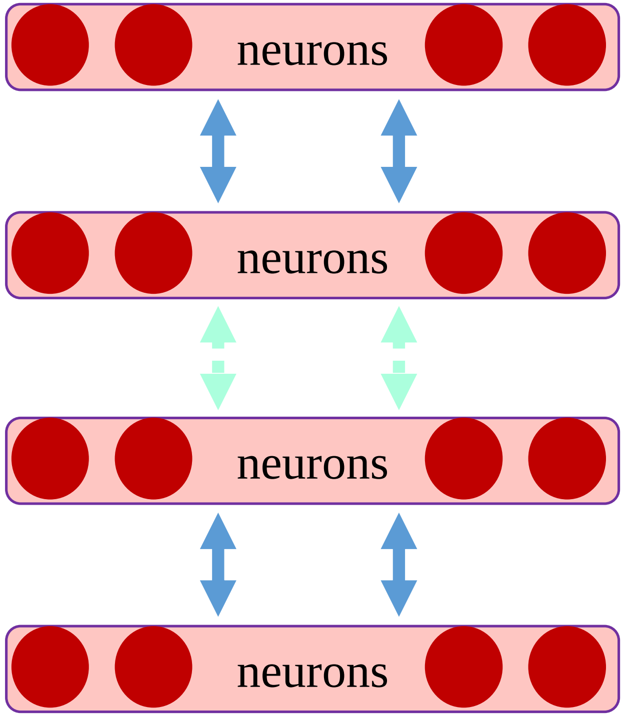


There exist layer structures for neurons.

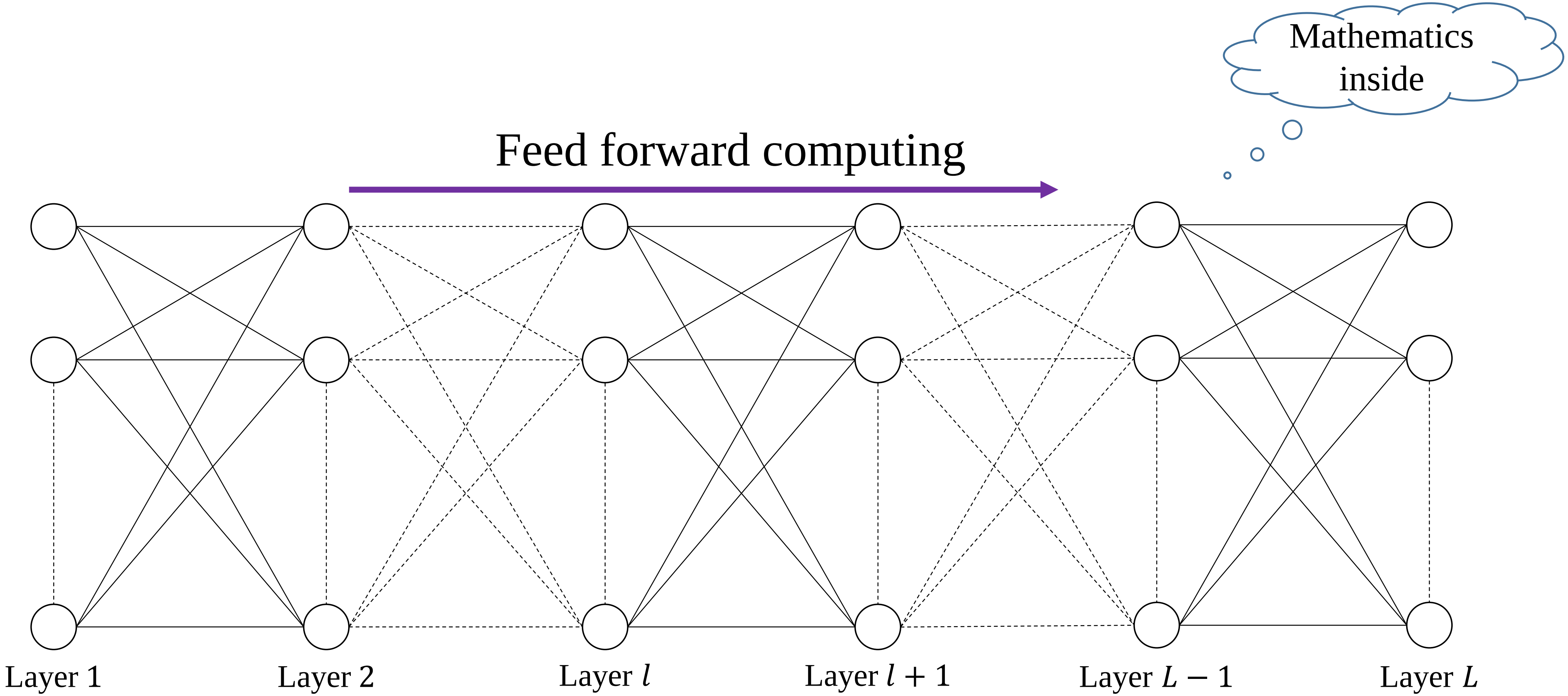




Neurons are structured in Layer way.



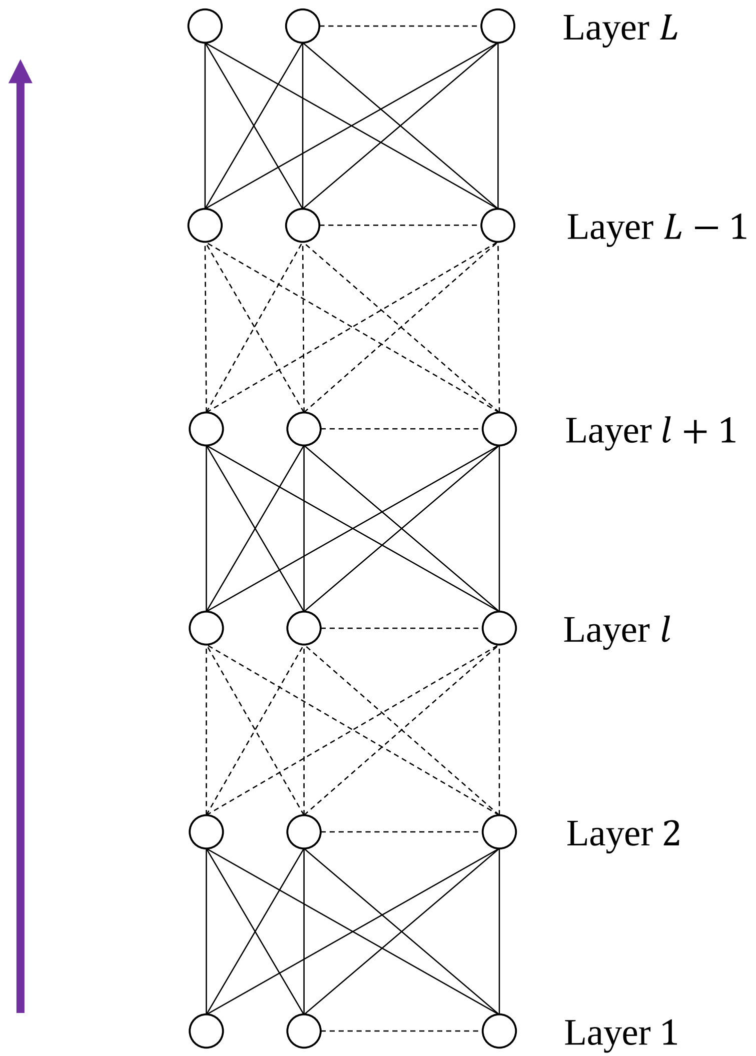
Computational Model of Neural Networks



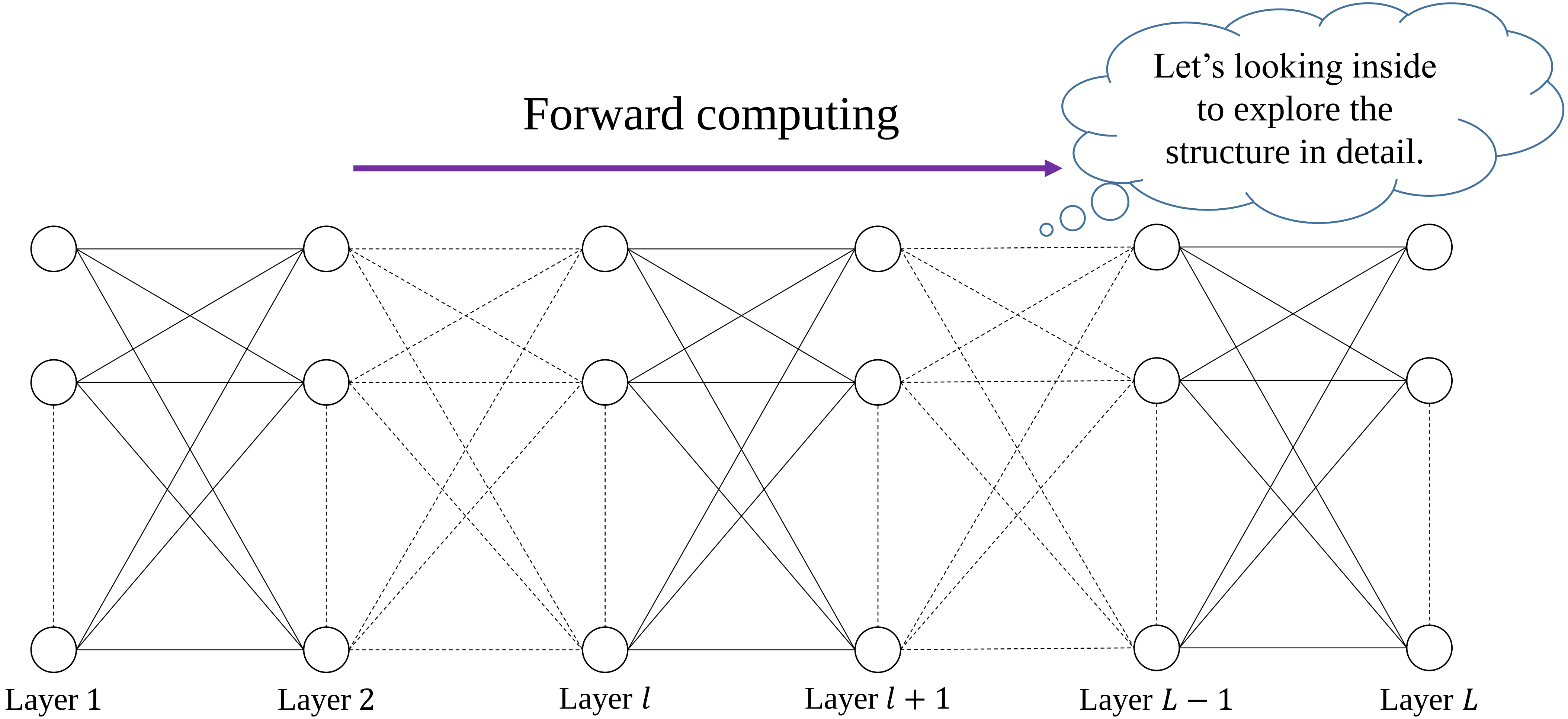
Topological structure of neural networks

Computational Model of Neural Networks

Another view to NN structure

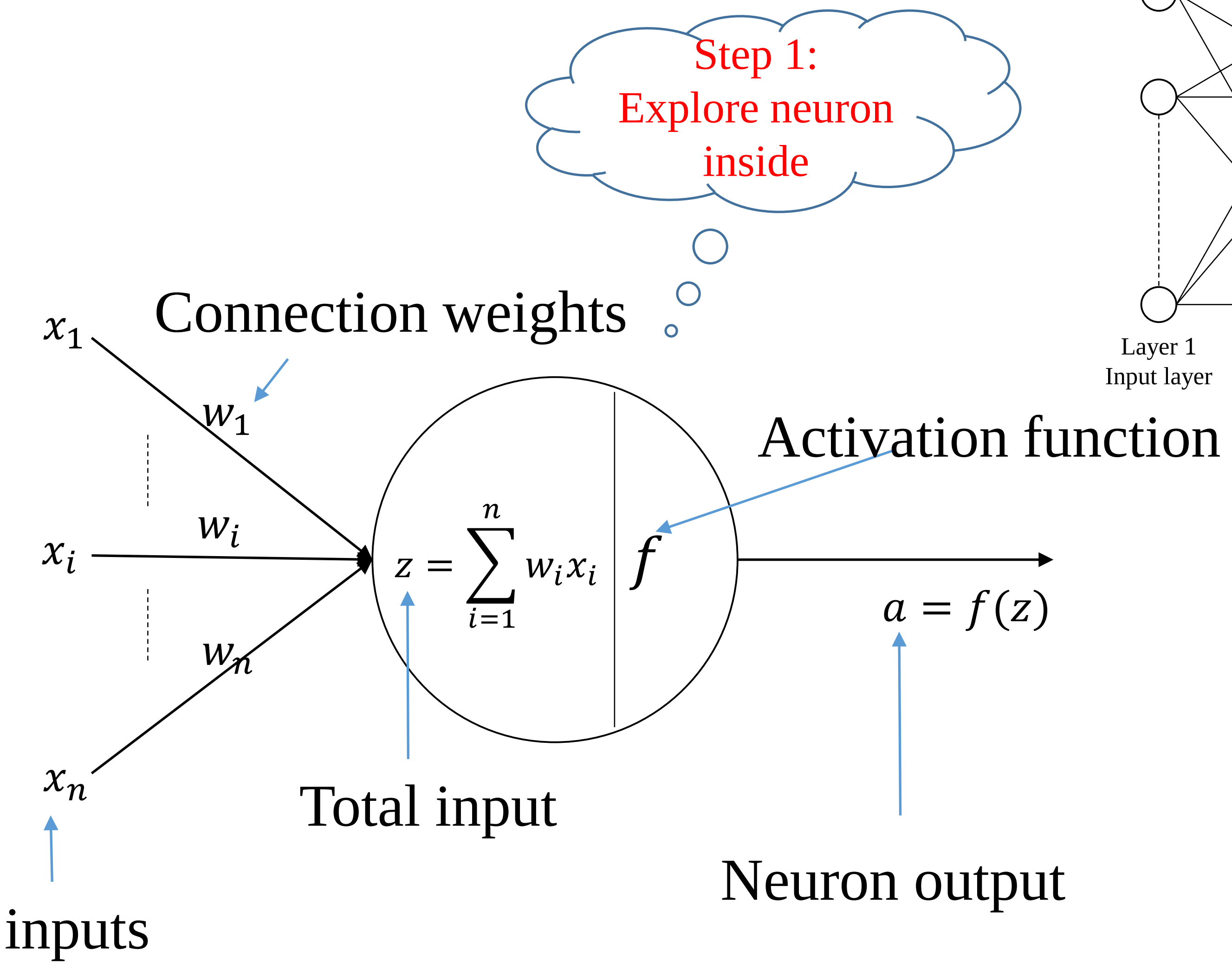


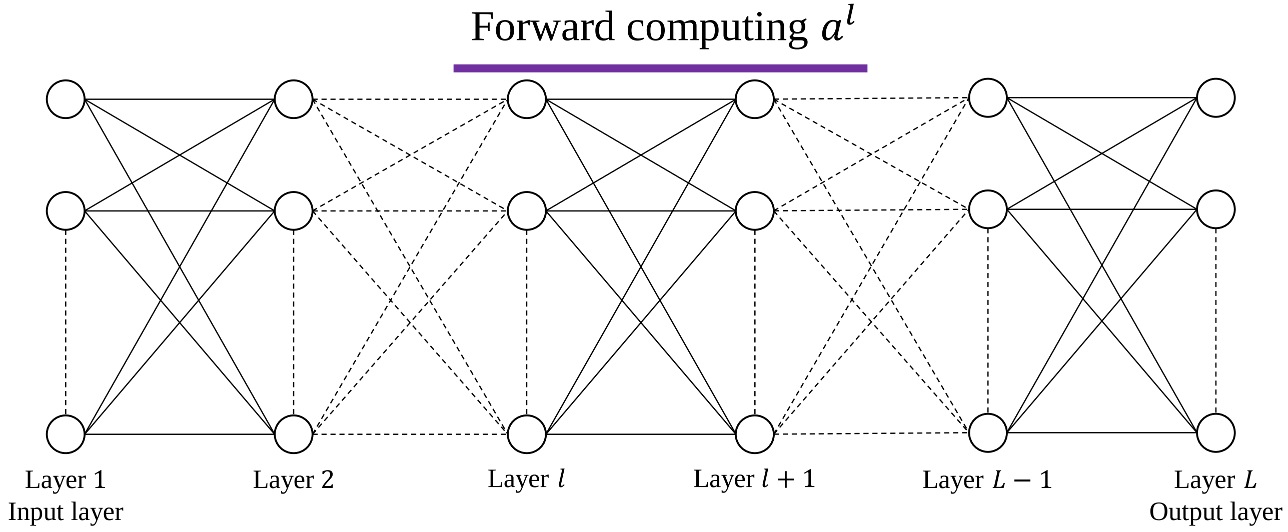
Computational Model of Neural Networks

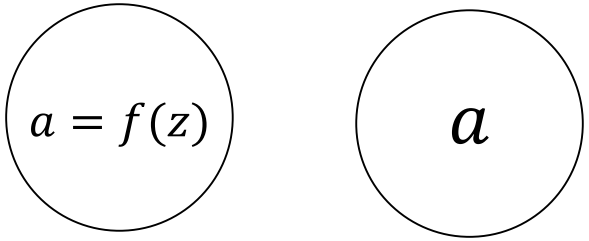


Topological structure of neural networks

Computational Model of Neural Networks





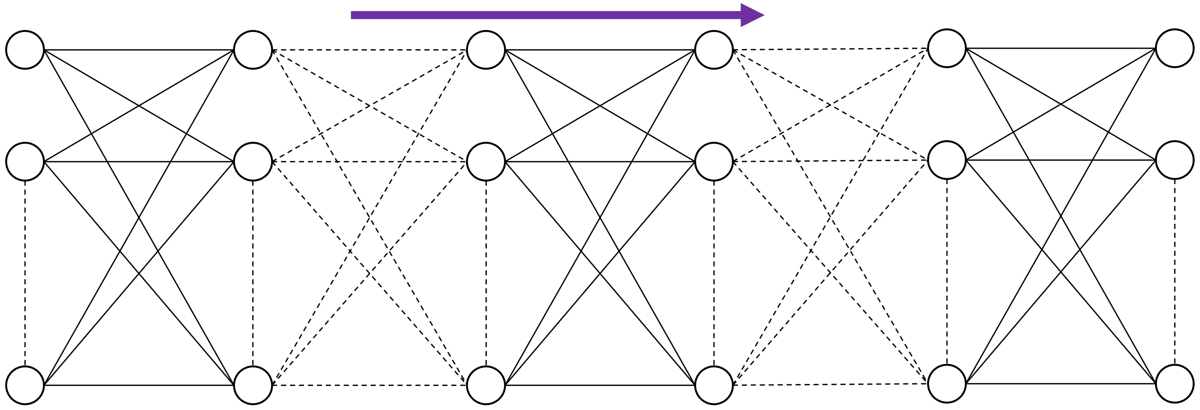


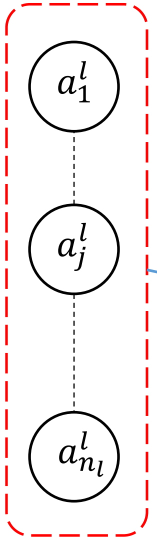
Computational Model of Neural Networks

Step 2:
Explore layer inside

Layer l contains n_l neurons.

Forward computing a^l





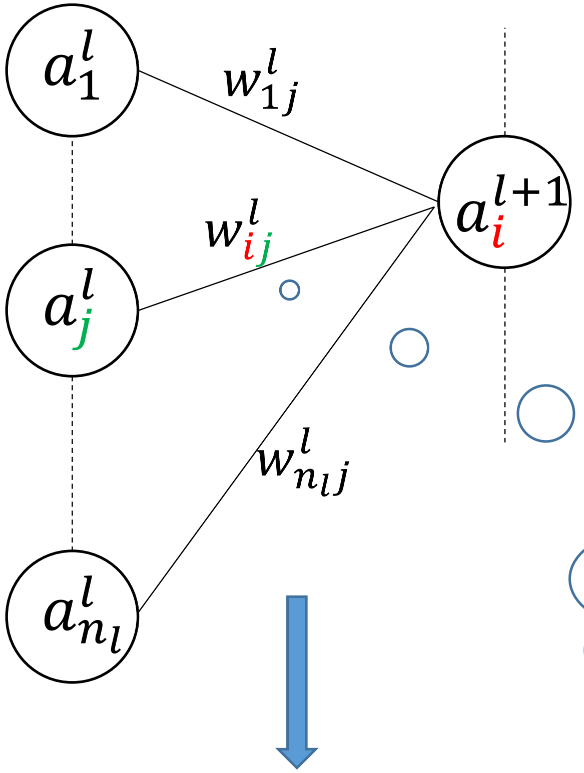
Layer l

$$a_j^l = f(z_j^l)$$

Vector form $\quad a^l = \begin{bmatrix} a_1^l \\ \vdots \\ a_j^l \\ \vdots \\ a_{n_l}^l \end{bmatrix}$

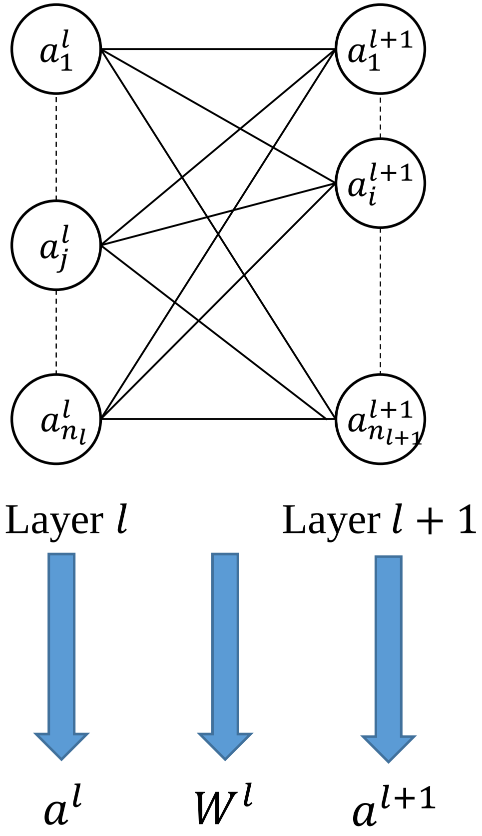
The neuron located in l layer j^{th} place,
a_j^l denotes the output value of the neuron.

Computational Model of Neural Networks



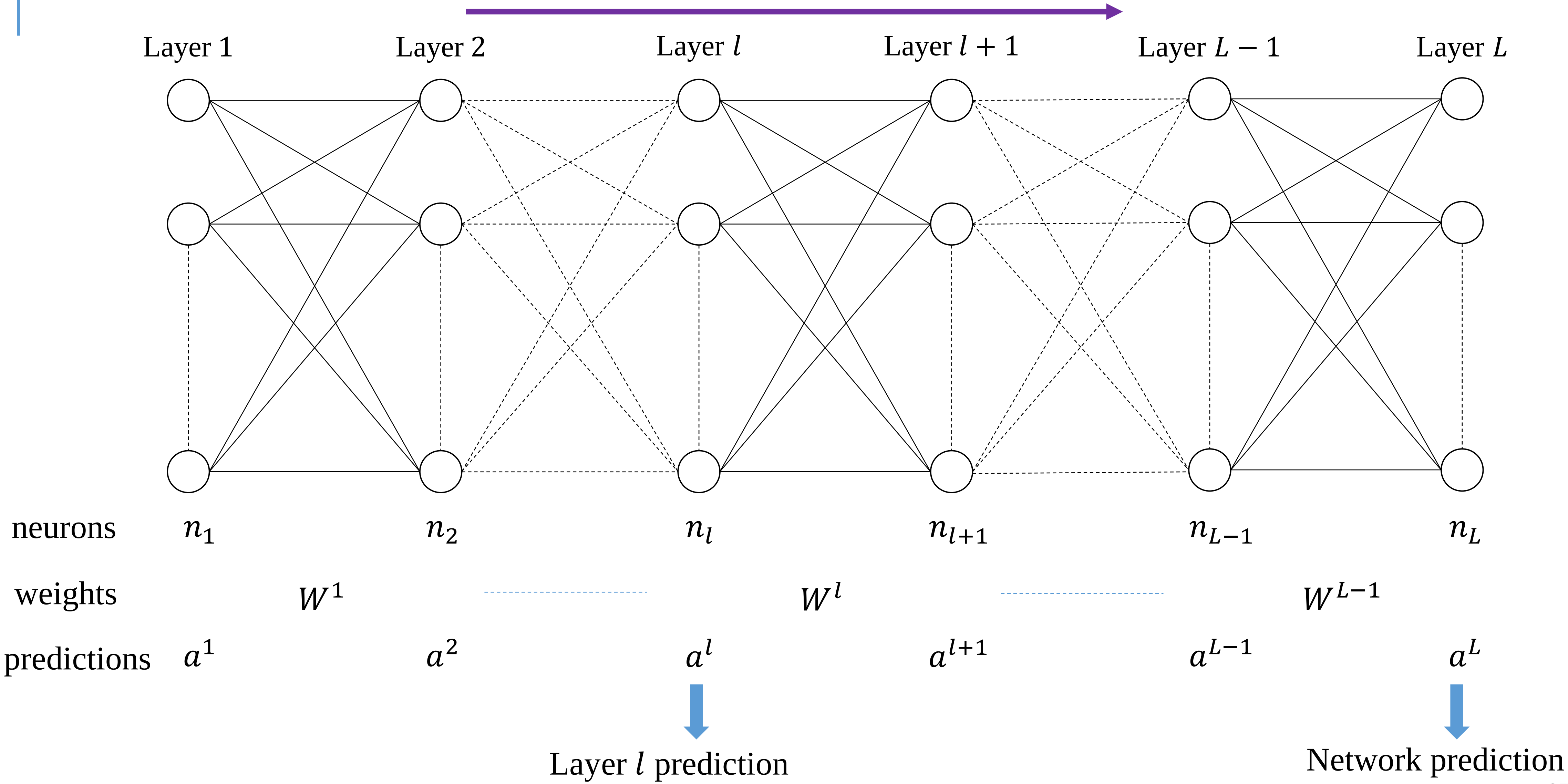
$$W^l = \begin{bmatrix} w_{11}^l & \cdots & w_{1n_l}^l \\ \vdots & w_{ij}^l & \vdots \\ w_{n_{l+1}1}^l & \cdots & w_{n_{l+1}n_l}^l \end{bmatrix}_{n_{l+1} \times n_l}$$

Step 3:
Explore connections between layers

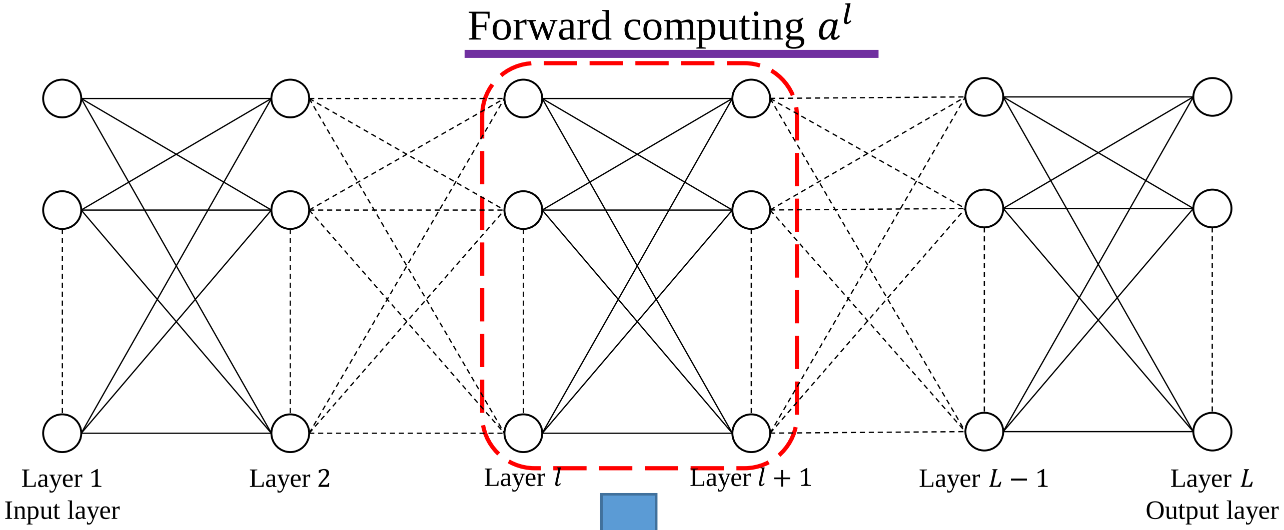


a^l is the input of $l+1$ layer. a^{l+1} is produced from a^l. a^l is called the network prediction at l layer. a^L is called the network prediction.

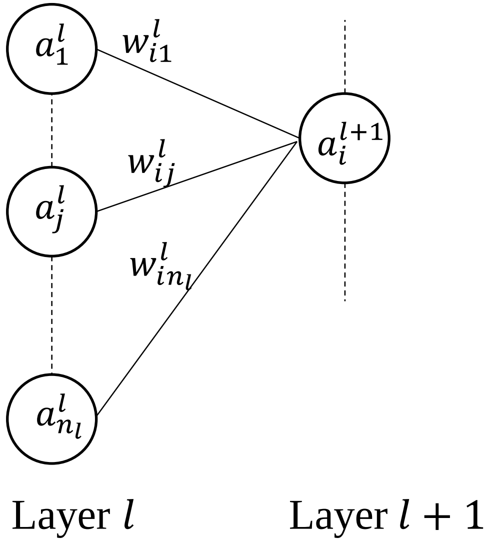
Forward computing



Computational Model of Neural Networks

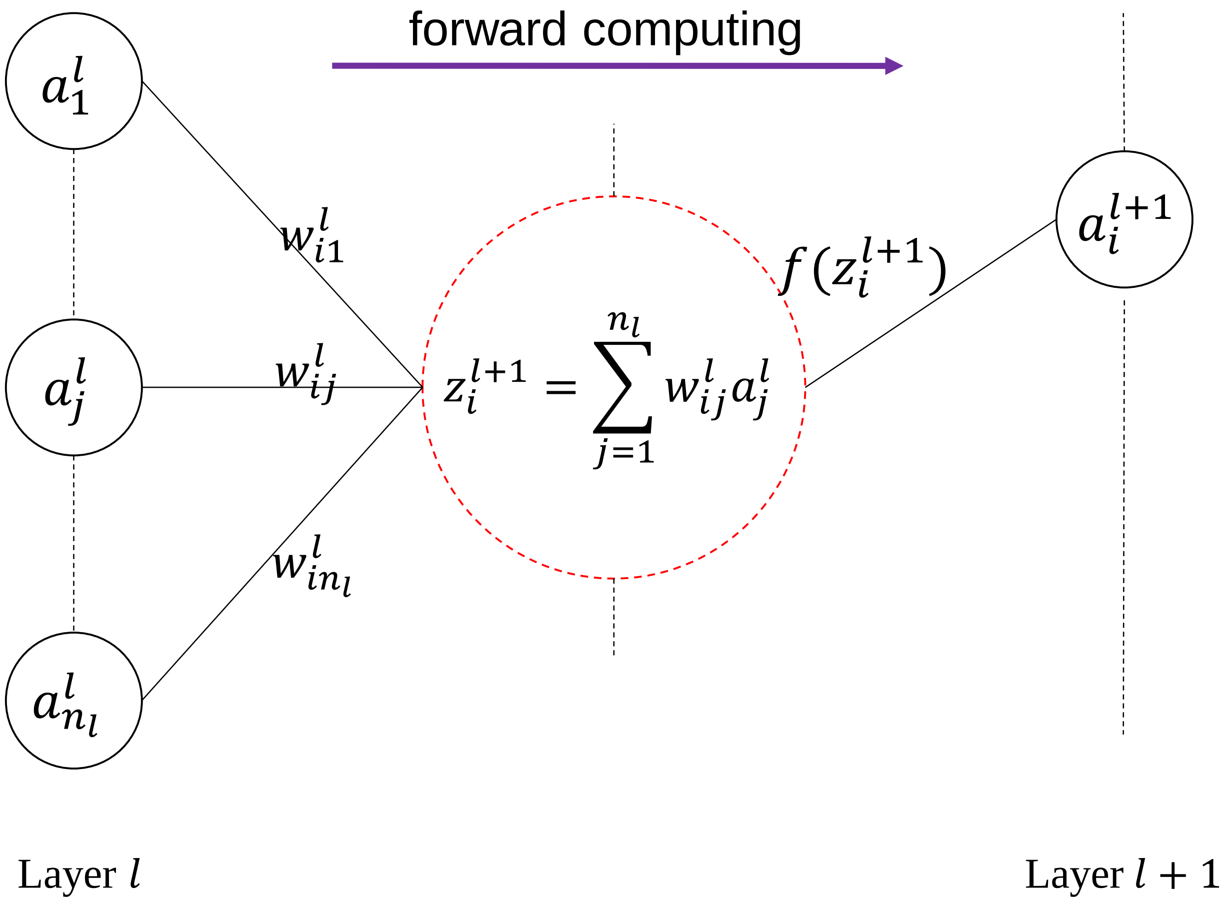


Problem:
How to do the forward computing?



How to use $a_1^l, \cdots, a_{n_l}^l$ and $w_{i1}^l, \cdots, w_{in_l}^l$ to compute a_i^{l+1}?

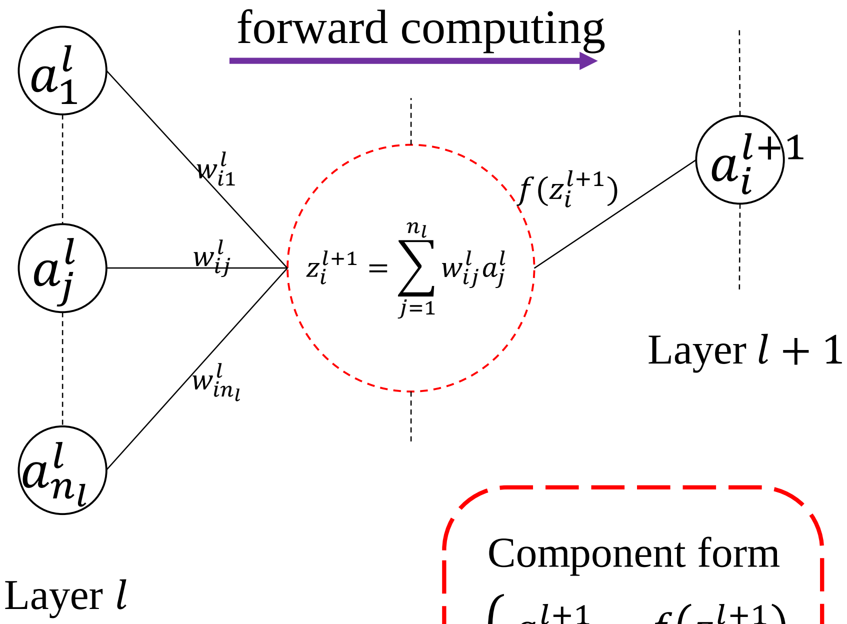
Computational Model of Neural Networks



$$\begin{cases} a_i^{l+1} = f\left(z_i^{l+1}\right) \\ z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l \end{cases}$$

$$a_i^{l+1} = f\left(\sum_{j=1}^{n_l} w_{ij}^l a_j^l\right)$$

Computational Model of Neural Networks

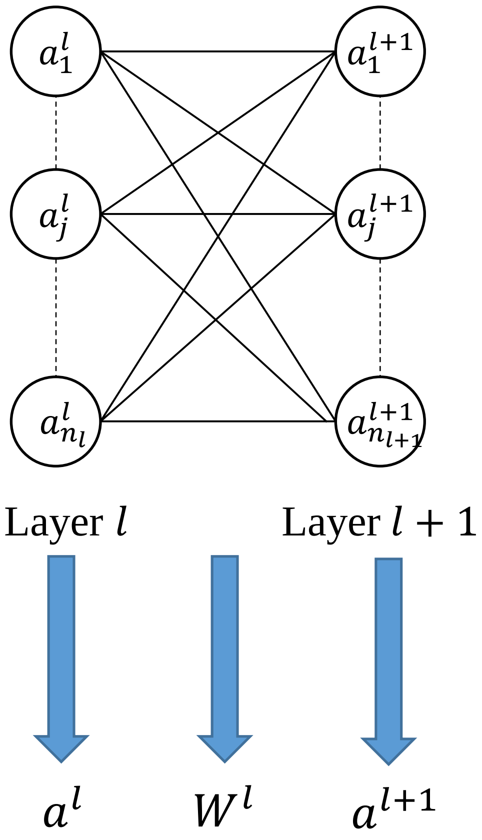


Component form

$$\begin{cases} a_i^{l+1} = f\left(z_i^{l+1}\right) \\ z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l \end{cases}$$

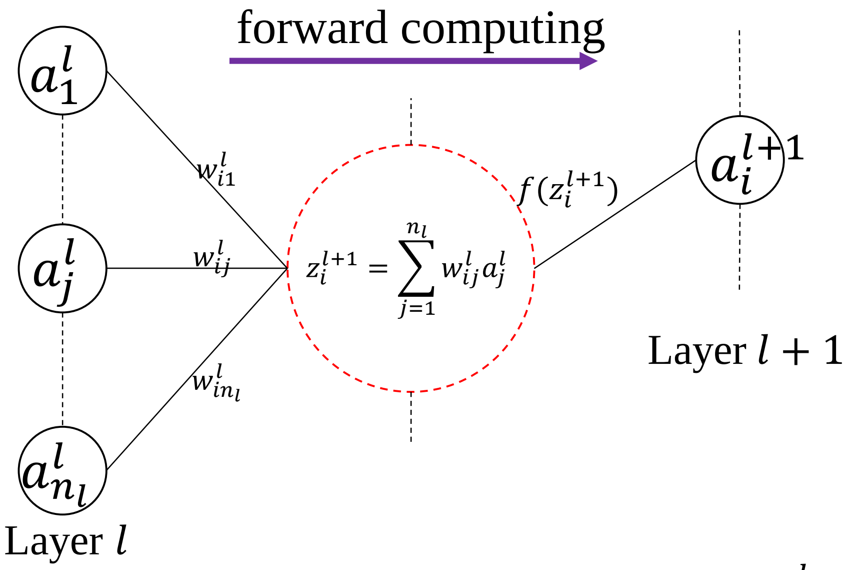
Vector form

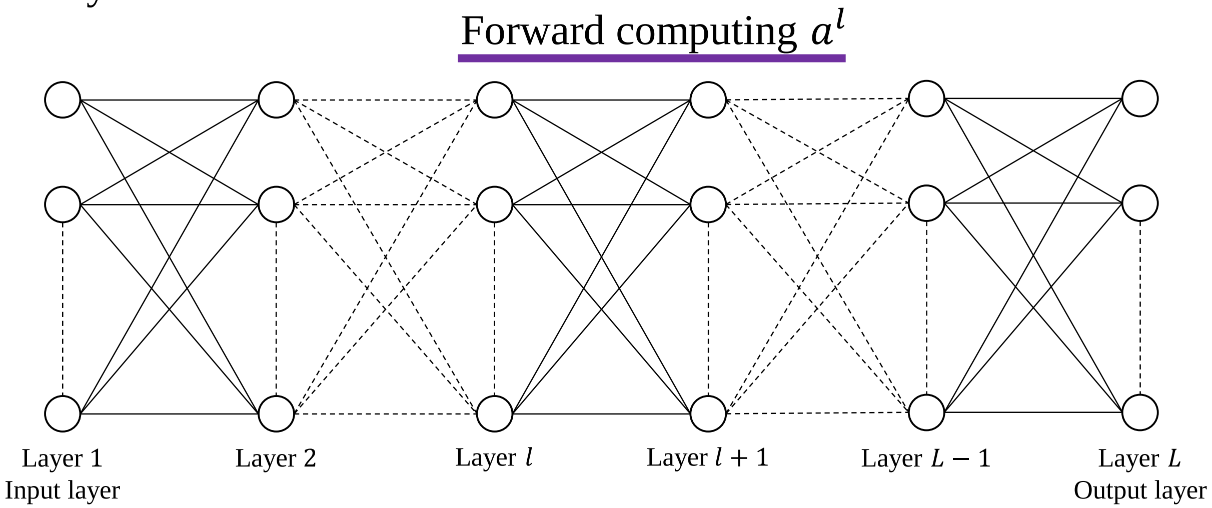
$$\begin{cases} a^{l+1} = f\left(z^{l+1}\right) \\ z^{l+1} = w^l a^l \end{cases}$$



a^l is the input of $l + 1$ layer.
a^{l+1} is the representation of a^l.

One page to understand forward computing





Algorithm:

$Input\ W^l, a^1$
$for\ l = 1{:}L$
$\quad a^{l+1} = fc(W^l, a^l)$
$return$

Function $fc(W^l, a^l)$
$for\ i = 1{:}n_{l+1}$

$$z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l$$

$$a_i^{l+1} = f(z_i^{l+1})$$

end

Component form

$$\begin{cases} a_i^{l+1} = f(z_i^{l+1}) \\ z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l \end{cases}$$

Vector form

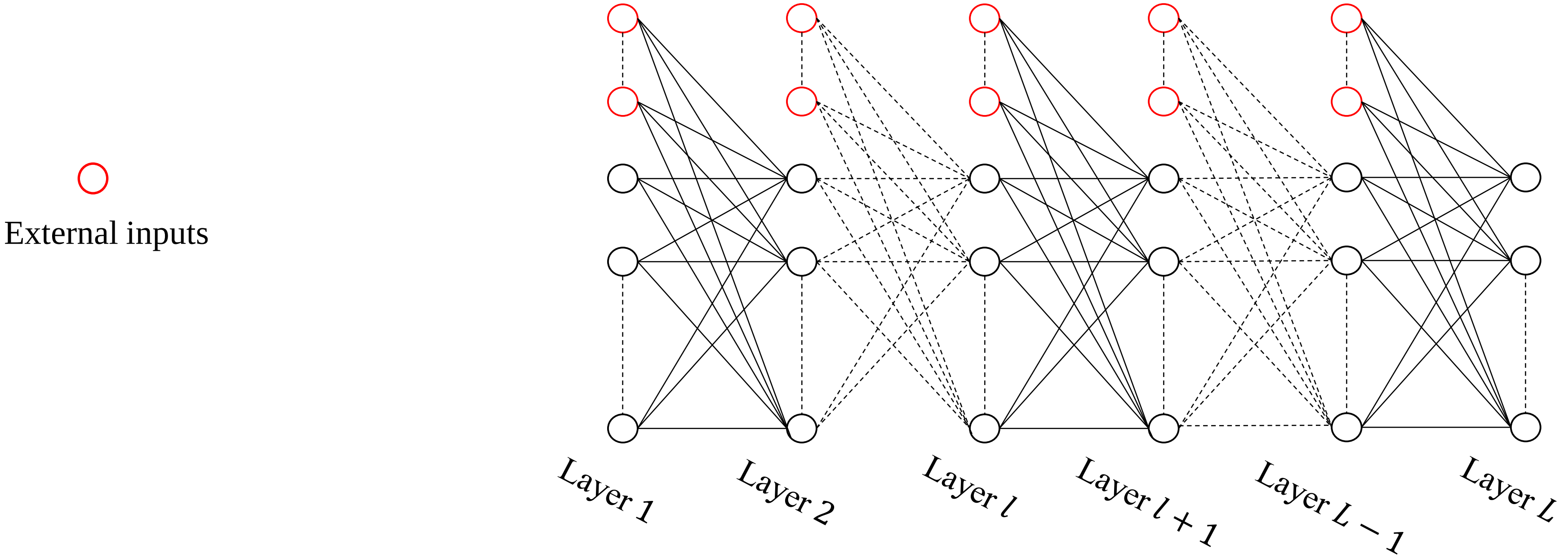
$$\begin{cases} a^{l+1} = f(z^{l+1}) \\ z^{l+1} = w^l a^l \end{cases}$$

$$a_i^{l+1} = f\left(\sum_{j=1}^{n_l} w_{ij}^l a_j^l\right)$$

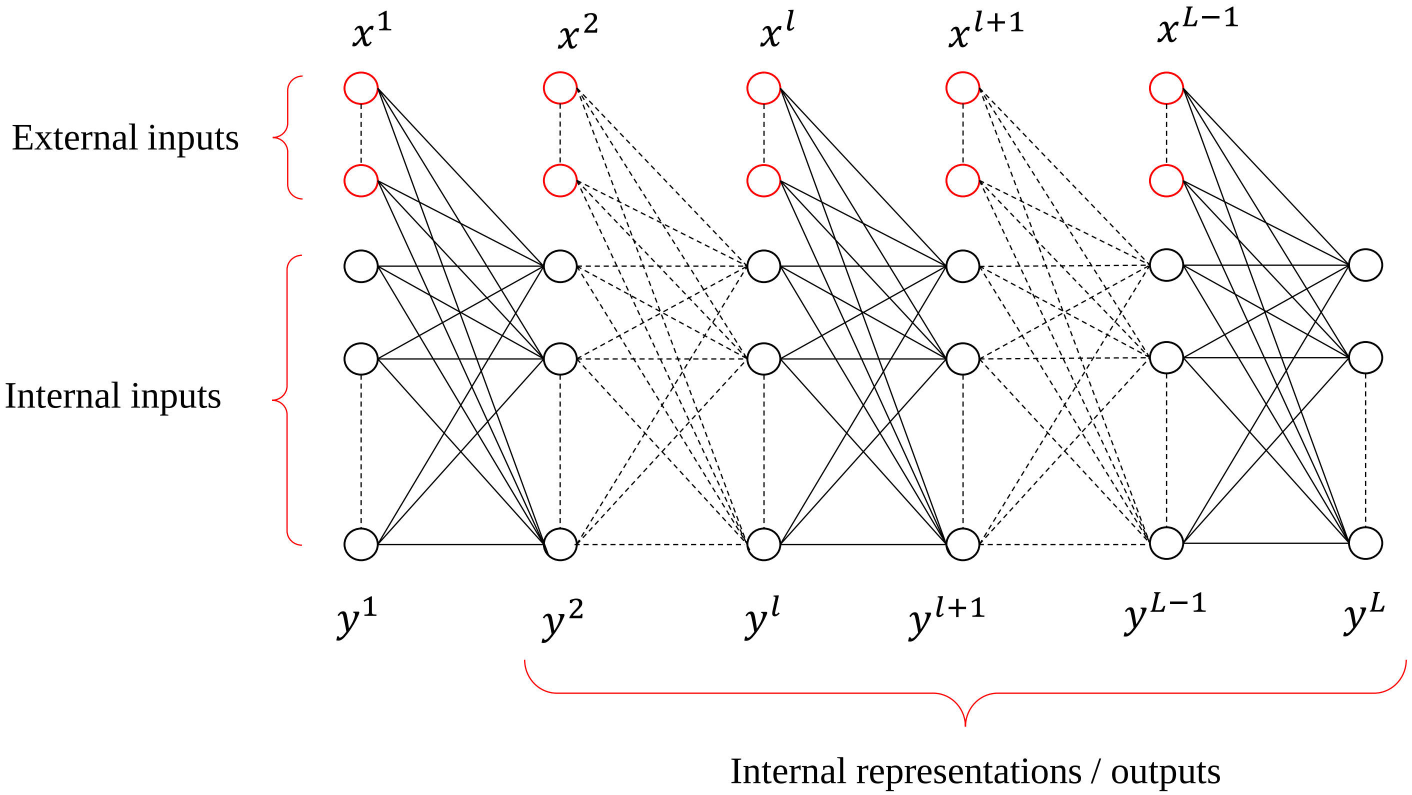
External Inputs

External inputs:

If neurons in l layer are not connected to any neurons in previous layer, these neurons are called external inputs of $l + 1$ layer.
External inputs can exist in any layer except the last one.



External Inputs

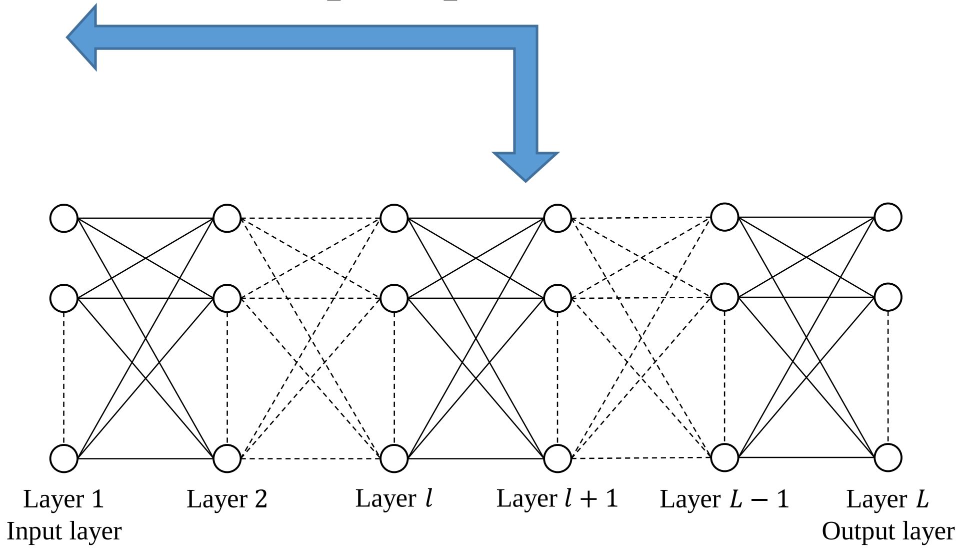


$$a^l = \begin{bmatrix} x^l \\ y^l \end{bmatrix}$$

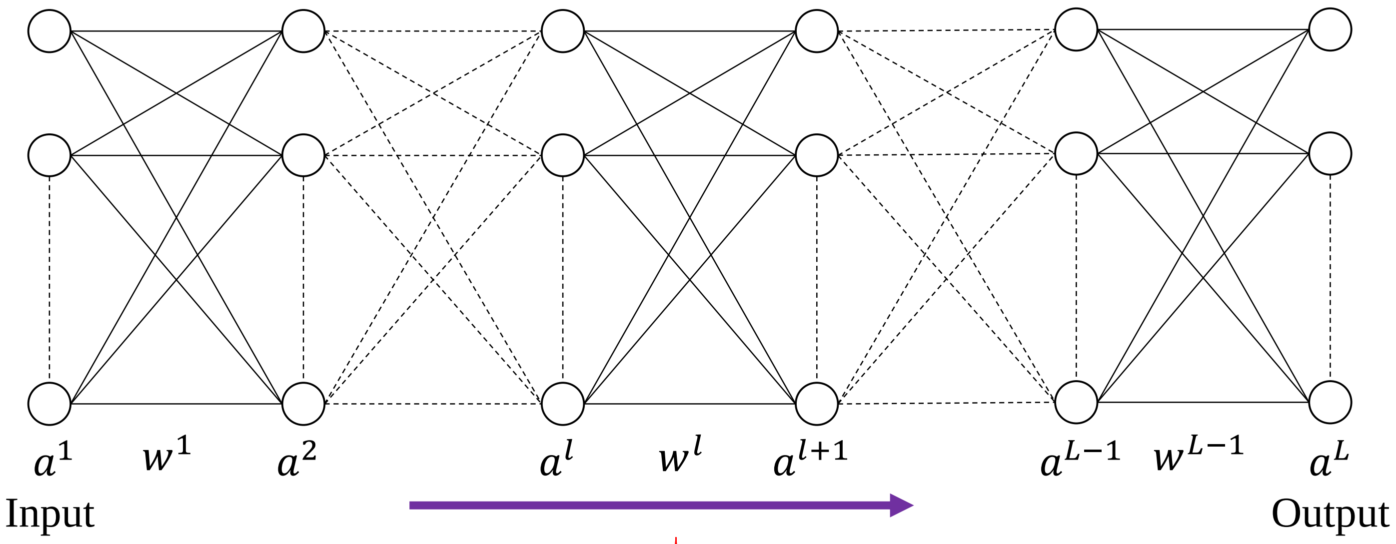
$$z^{l+1} = W^l a^l$$

$$y^{l+1} = f\left(z^{l+1}\right)$$

$$a^{l+1} = \begin{bmatrix} x^{l+1} \\ y^{l+1} \end{bmatrix}$$



Nonlinear Mapping / Dynamical Systems



A neural network can be looked as a nonlinear mapping or a dynamical system.

$$a^L = f\left(W^{L-1} f\left(W^{L-2} f\left(W^{L-3} \cdots f\left(W^1 a^1\right)\right)\right)\right)$$

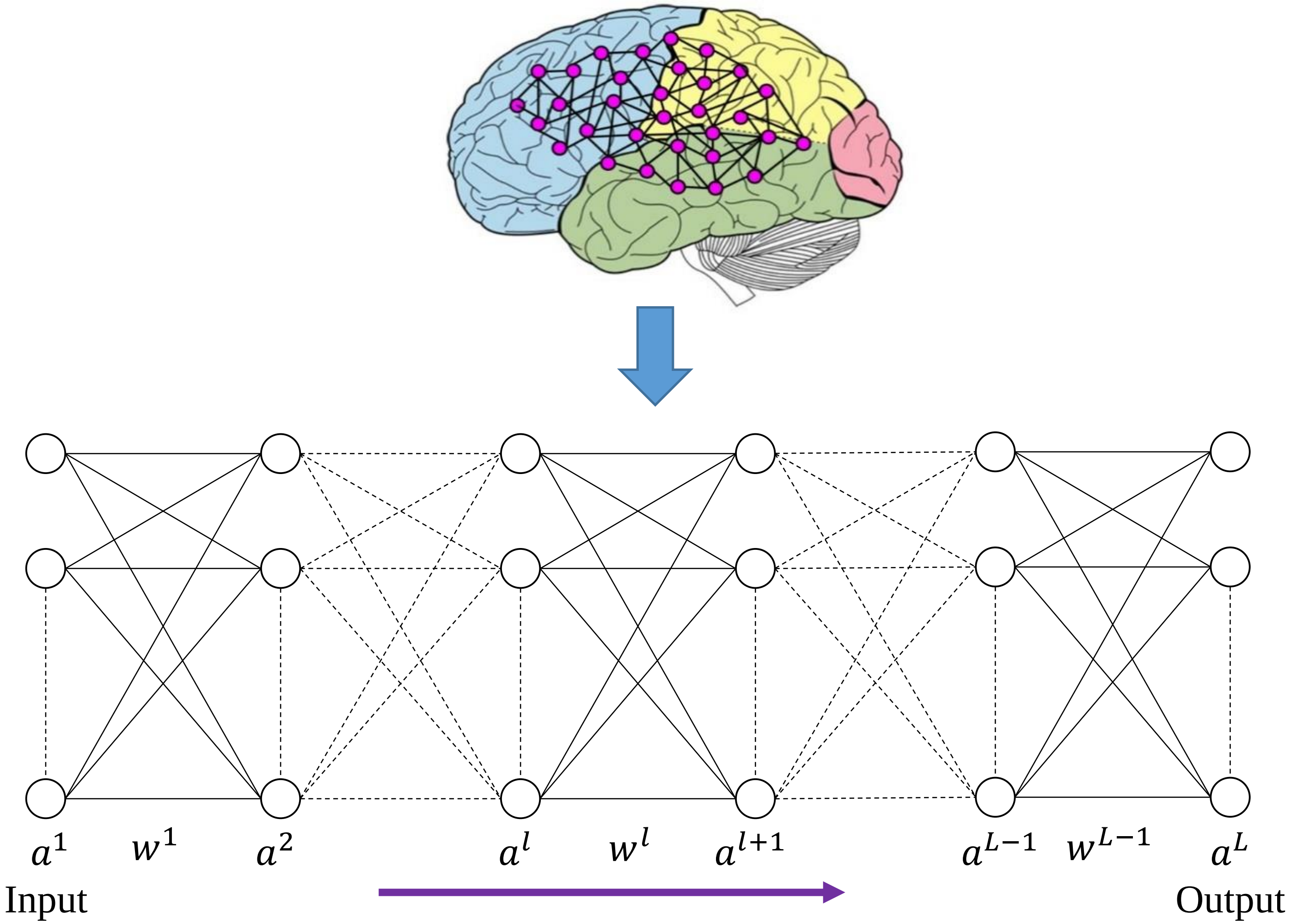
$$\boxed{R^{n_1}} \longrightarrow \boxed{R^{n_L}}$$

Nonlinear mapping

$$a_i^{l+1} = f\left(\sum_{j=1}^{n_l} w_{ij}^l a_j^l\right) \Rightarrow a_i(l+1) = f\left(\sum_{j=1}^{n_l} w_{ij}(l) a_j(l)\right)$$

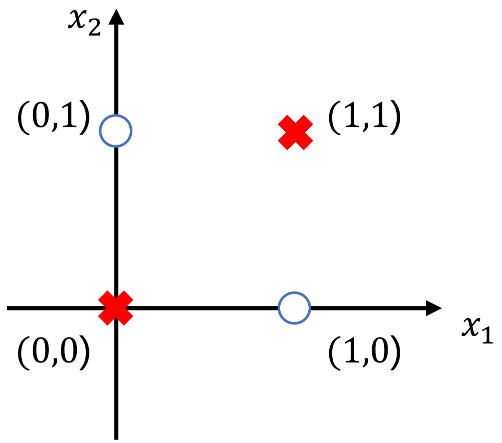
Dynamical system

Nonlinear Mapping / Dynamical Systems



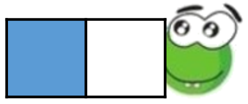
$$a_i^{l+1} = f\left(\sum_{j=1}^{n_l} w_{ij}^l a_j^l\right)$$

An example: XOR-worms problem



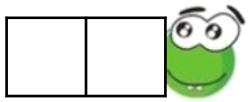
Doted worms

$\begin{bmatrix}0\\1\end{bmatrix}$

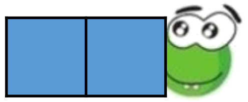


$\begin{bmatrix}1\\0\end{bmatrix}$

Smooth worms



$\begin{bmatrix}0\\0\end{bmatrix}$



$\begin{bmatrix}1\\1\end{bmatrix}$

1

0

An example: XOR-worms problem

$$\begin{bmatrix}0\\1\end{bmatrix} \quad \begin{bmatrix}1\\0\end{bmatrix}$$

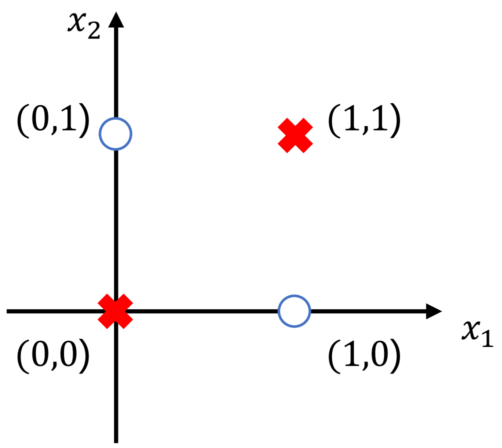
$$\begin{bmatrix}0\\0\end{bmatrix} \quad \begin{bmatrix}1\\1\end{bmatrix}$$

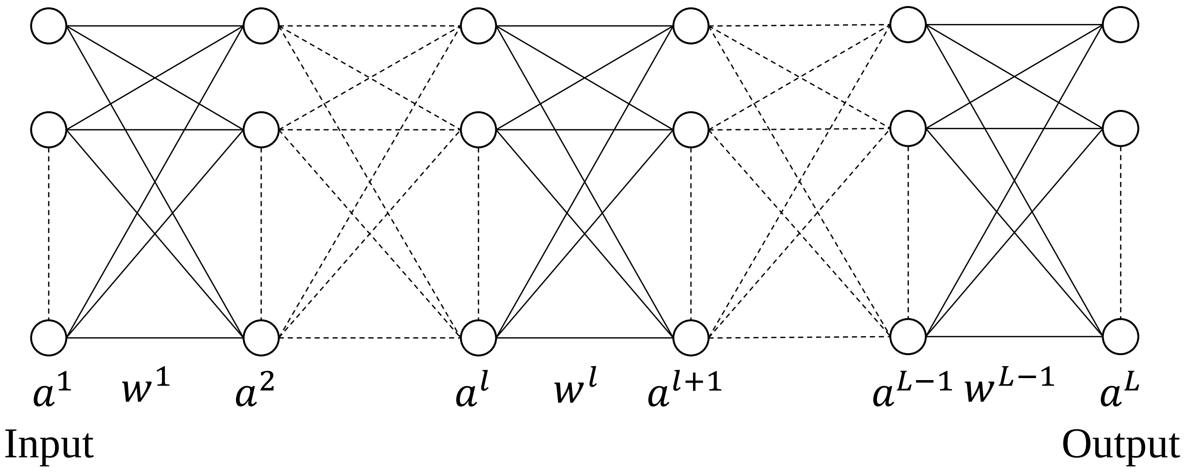
$$F\left(\begin{bmatrix}x_1\\x_2\end{bmatrix}\right) = f[f(2x_1 + 2x_2 - 1) + f(-x_1 - x_2 + 1.5) - 1.5] \Rightarrow \boxed{1}$$

$$f(s) = \begin{cases}1, & s \geq 0\\0, & otherwise\end{cases} \Rightarrow \boxed{0}$$

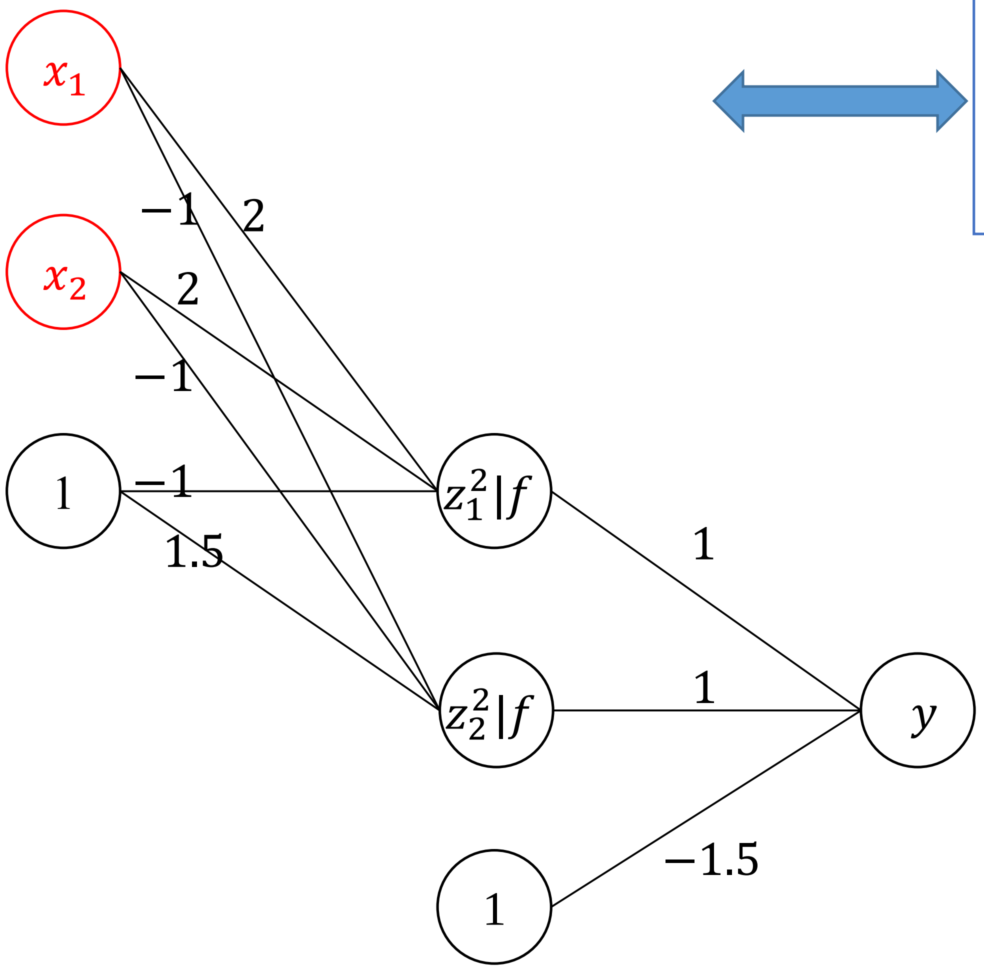
An FNN is a nonlinear mapping.

Problem: Can we construct an FNN to replace *F*?



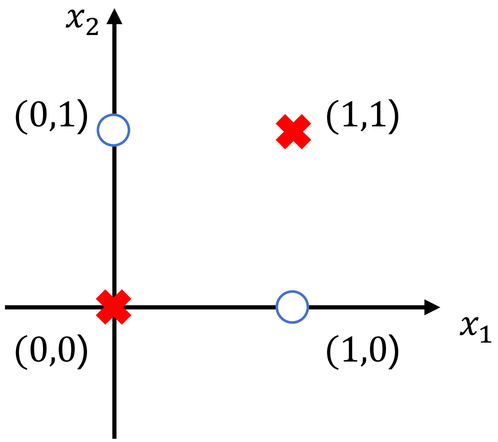


An example: XOR-worms problem



$$F\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = f[f(2x_1 + 2x_2 - 1) + f(-x_1 - x_2 + 1.5) - 1.5]$$

$$f(s) = \begin{cases} 1, & s \geq 0 \\ 0, & otherwise \end{cases}$$

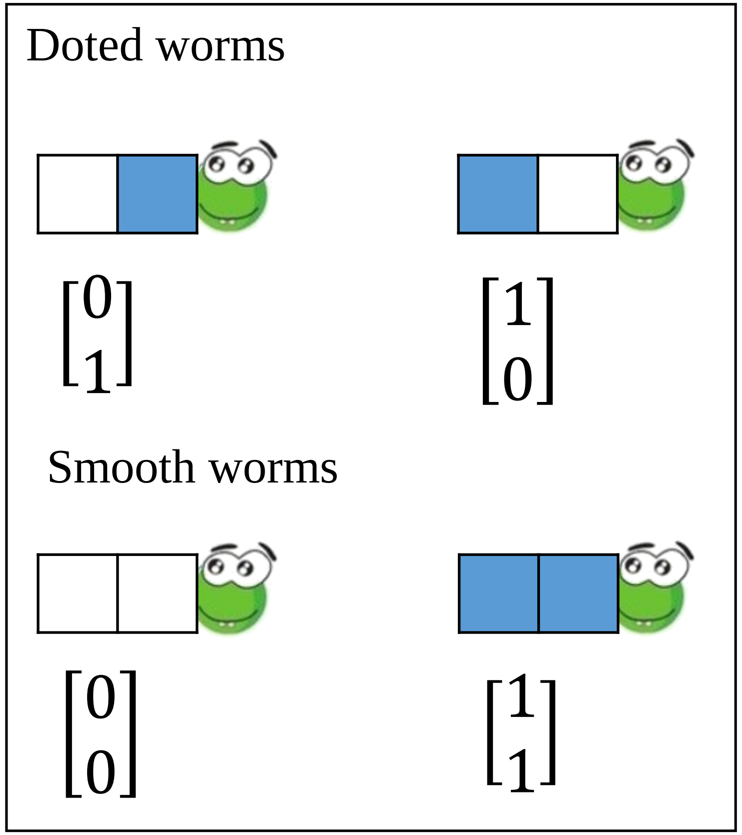


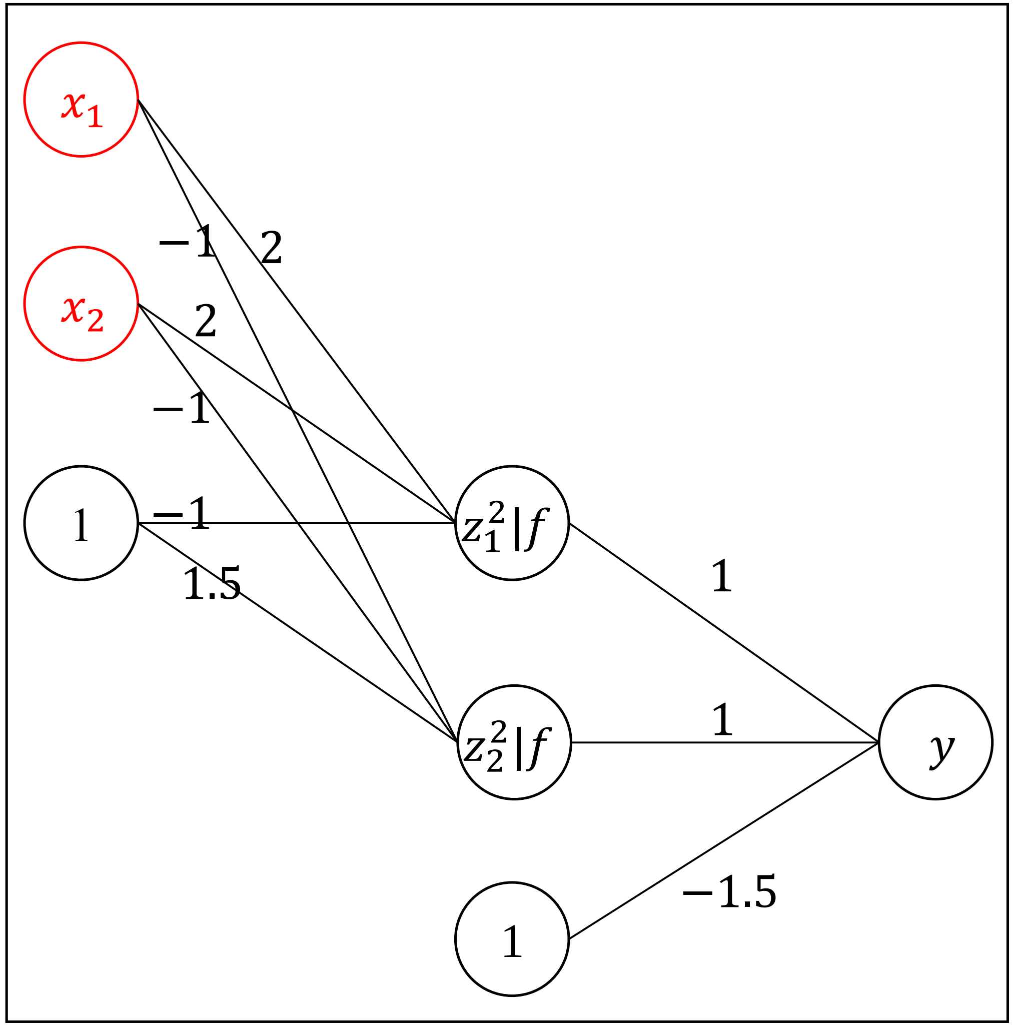
$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Can this network really replace F?
Check it!

An example: XOR-worms problem

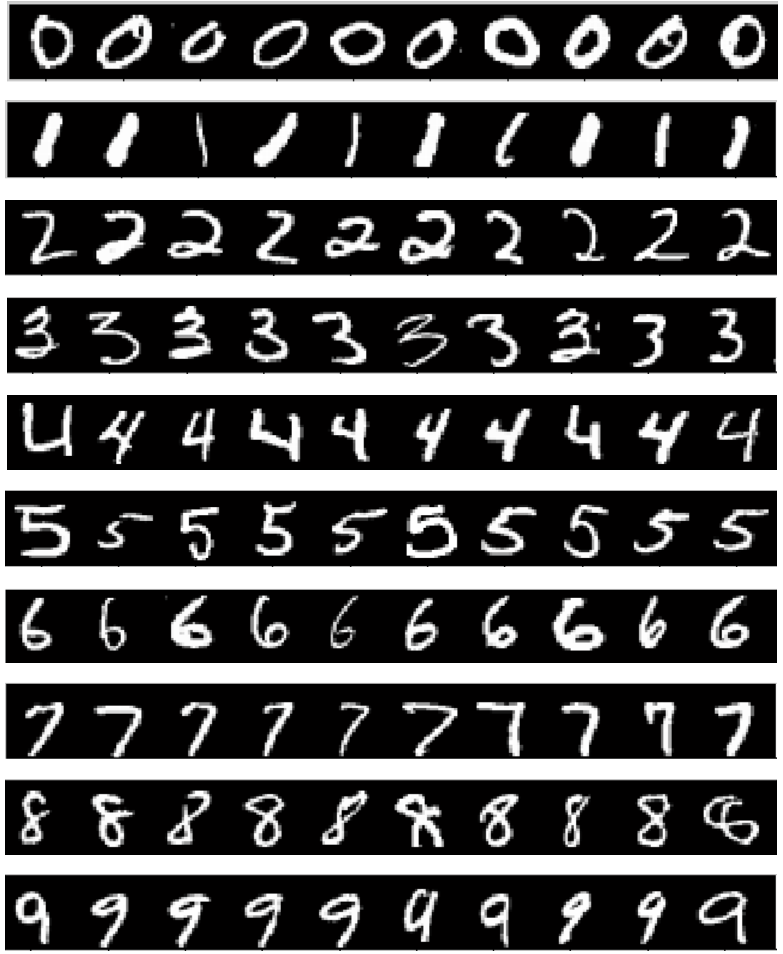




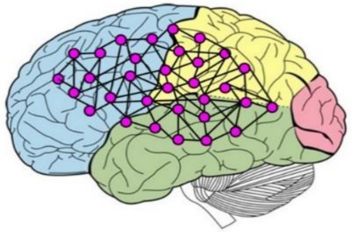
1

0

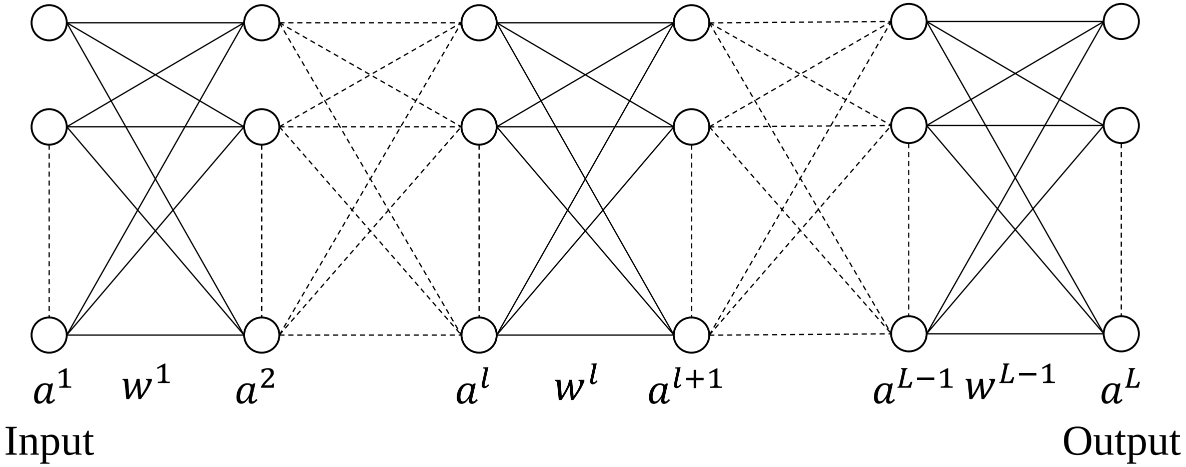
Handwritten Digits Recognition



digitizing → $F: x \to y$ → representation → 0, 1, 2, 3, 4, 5, 6, 7, 8, 9



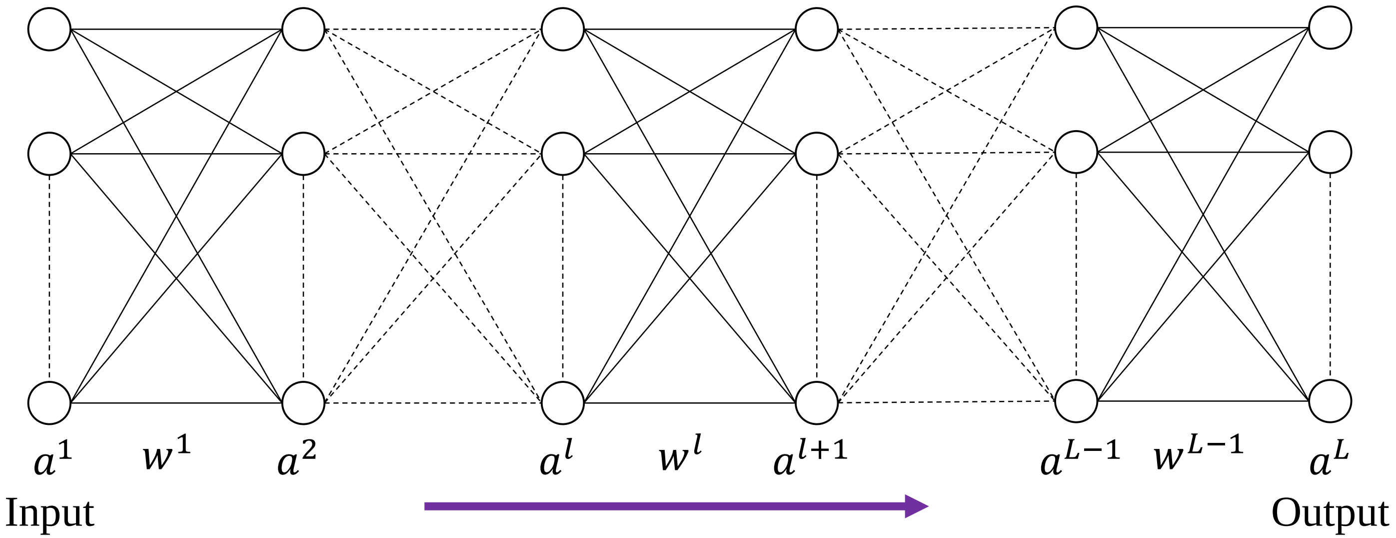
Problem: Can we construct an FNN to replace *F*?



Outline

- Brief Review of Brain Structure
- Computational Model of Neurons
- Computational Model of Neural Networks
- Continuous Time Neural Networks
- Assignments

Discrete Time Neural Networks

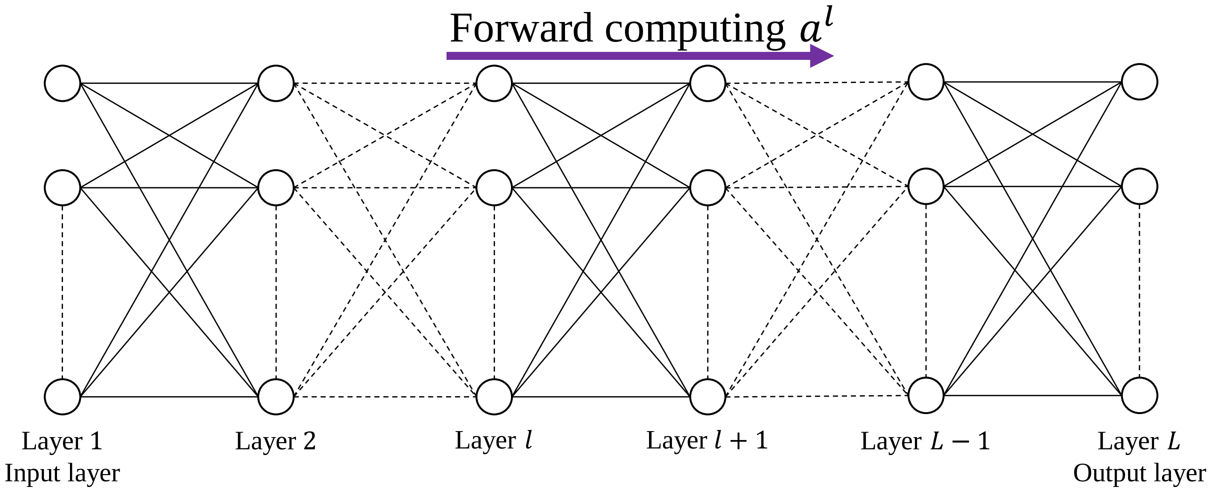


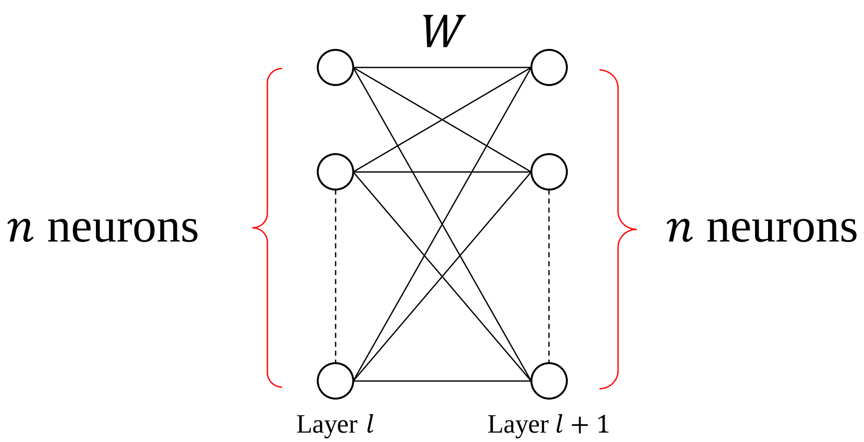
$$a_i^{l+1} = f\left(\sum_{j=1}^{n_l} w_{ij}^l a_j^l\right) \Rightarrow a_i(l+1) = f\left(\sum_{j=1}^{n_l} w_{ij}(l) a_j(l)\right) \xrightarrow{l \to t} a_i(t+1) = f\left(\sum_{j=1}^{n_t} w_{ij}(t) a_j(t)\right)$$

Discrete time

Continuous Time Neural Networks

Continuous Time Neural Networks



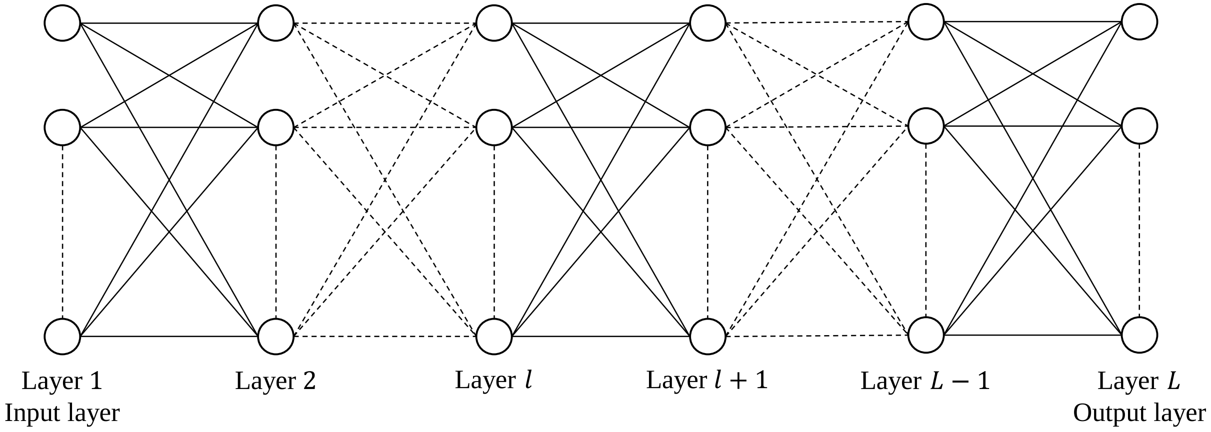


Suppose that
1. $n_1 = n_2 = \cdots = n_L = n$
2. $W^1 = W^2 = \cdots = W^L = W$

$$a_i^{l+1} = f\left(\sum_{j=1}^{n} w_{ij} a_j^l\right)$$

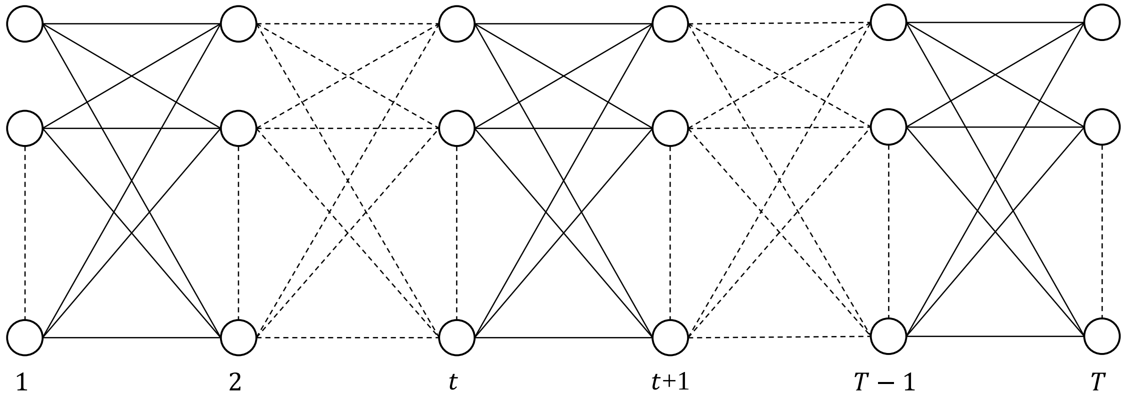
$$a^{l+1} = f(Wa^l)$$

Continuous Time Neural Networks



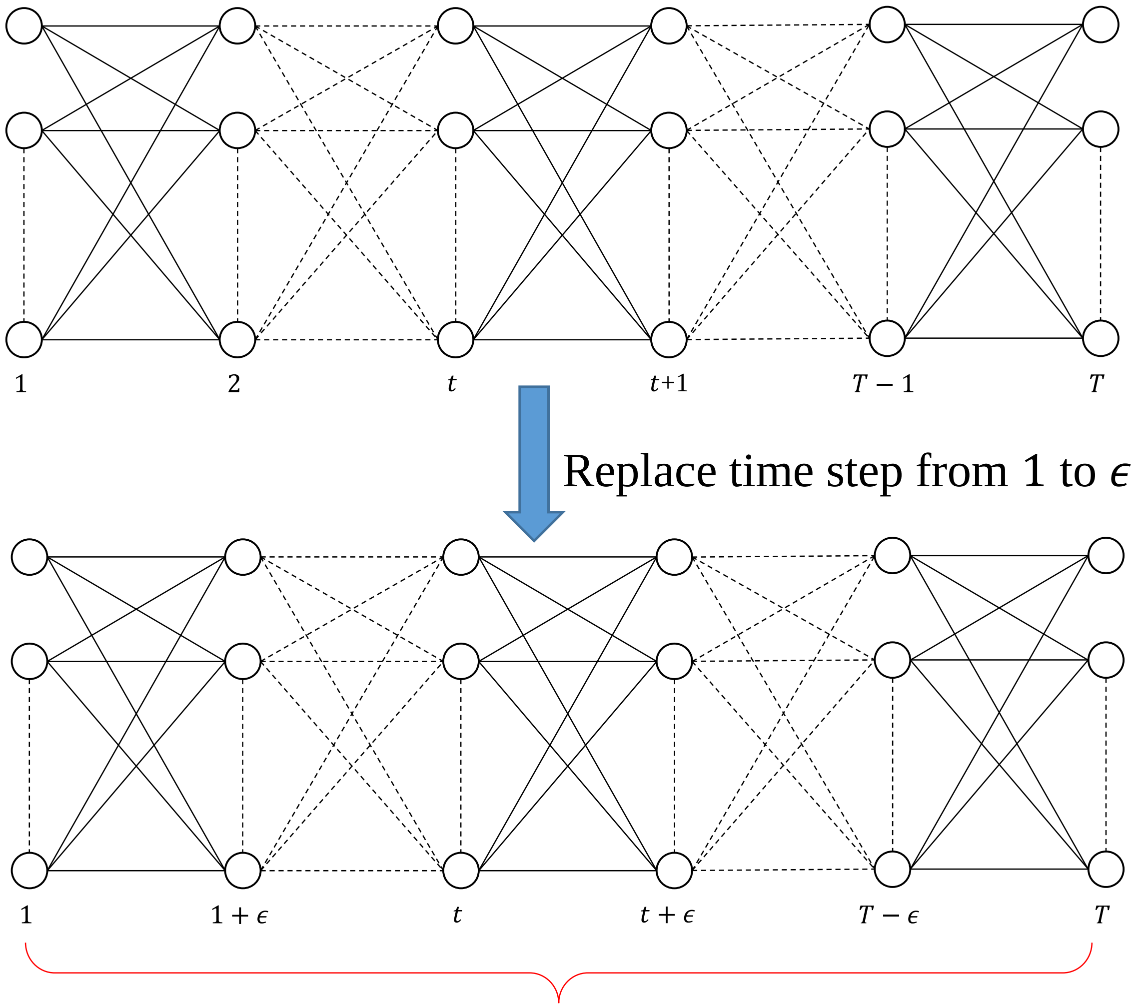
$$a_i^{l+1} = f\left(\sum_{j=1}^{n} w_{ij} a_j^l\right)$$

replace l by t replace a_i^l by $a_i(t)$



$$a_i(t+1) = f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$

Continuous Time Neural Networks



$$a_i(t+1) = f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$

Problem:
How to develop model for continuous time neural networks?

ϵ is an infinitesimal variable, thus, there are infinite layers

Starting from here: $a_i(t+1) = f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$

$$a_i(t+1) - a_i(t) = -a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$

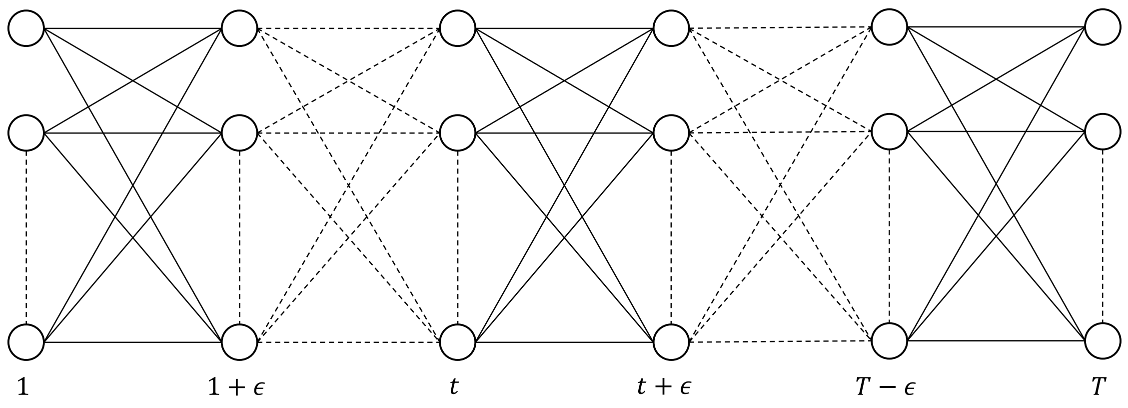
$$a_i(t+1) - a_i(t) = 1 \cdot \left[-a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)\right]$$

$$a_i(t+\epsilon) - a_i(t) = \epsilon \cdot \left[-a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)\right]$$

$$\frac{a_i(t+\epsilon) - a_i(t)}{\epsilon} = \left[-a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)\right]$$

$\epsilon \to 0$

$$\frac{da_i(t)}{dt} = -a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$

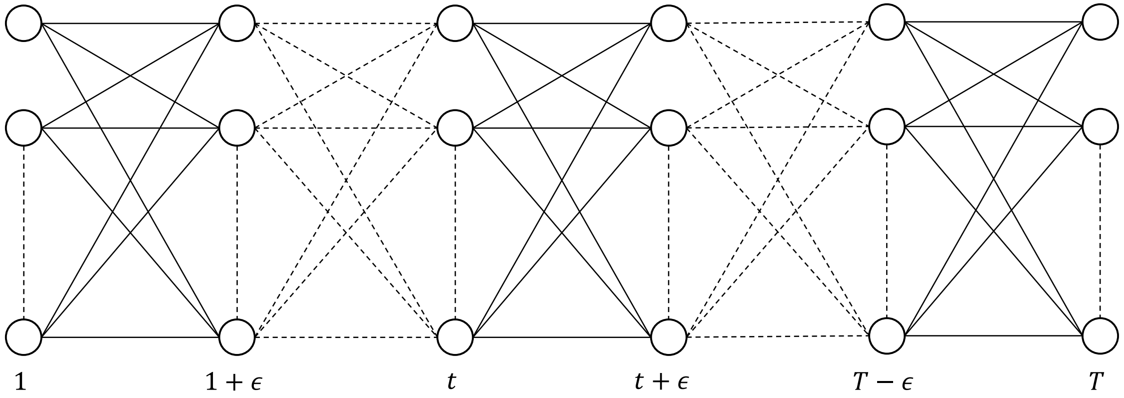


ϵ is an infinitesimal variable, thus, there are infinite layers

$\epsilon \to 0$

$$\frac{da_i(t)}{dt} = -a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$

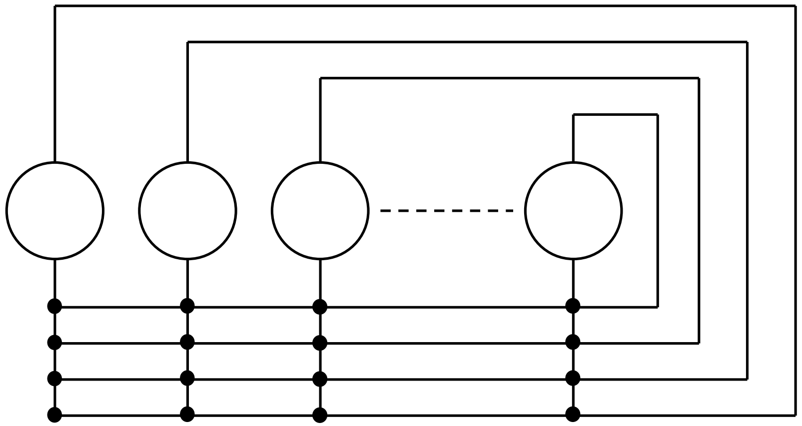
Continuous Time Neural Networks



ϵ is an infinitesimal variable, thus, there are infinite layers

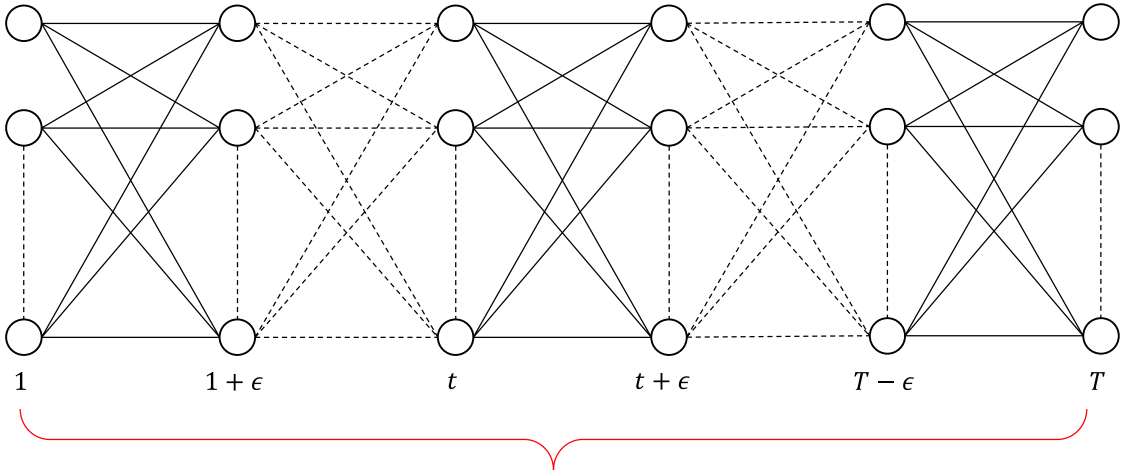
$$\Big\downarrow \ \epsilon \to 0$$

$$\frac{da_i(t)}{dt} = -a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$



Summary

$$y = f\left(\sum_{i=1}^{n} w_i x_i\right)$$



$\epsilon = 1, t = l$

$$a_i^{l+1} = f\left(\sum_{j=1}^{n_l} w_{ij}^l a_j^l\right)$$

$\epsilon \to 0$

1. $n_1 = n_2 = \cdots = n_L = n$
2. $W^1 = W^2 = \cdots = W^L = W$

$$\frac{da_i(t)}{dt} = -a_i(t) + f\left(\sum_{j=1}^{n} w_{ij} a_j(t)\right)$$

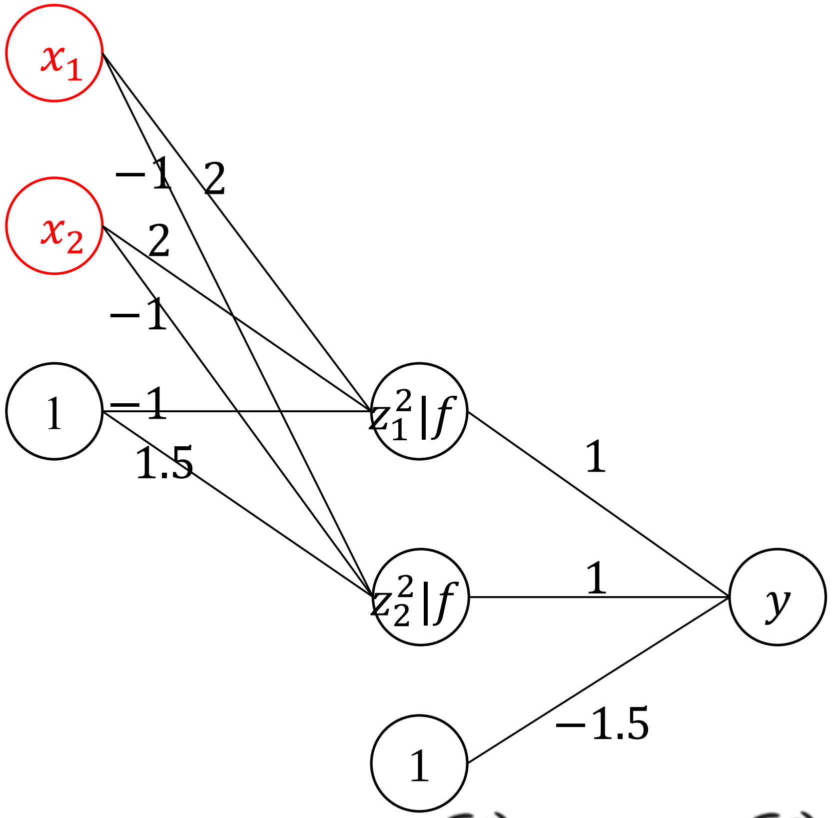
Outline

- Brief Review of Brain Structure
- Computational Model of Neurons
- Computational Model of Neural Networks
- Continuous Time Neural Networks
- Assignments

Assignment

Implement the forward computing of this NN:

- in component form
- in vector form



Algorithm in Component form:

$Input\ W^l, a^1$
$for\ l = 1{:}L$
$\qquad a^{l+1} = fc\_c(W^l, a^l)$
$return$

Function $fc\_c(W^l, a^l)$
$for\ i = 1{:}n_{l+1}$

$$z_i^{l+1} = \sum_{j=1}^{n_l} w_{ij}^l a_j^l$$

$$a_i^{l+1} = f(z_i^{l+1})$$

end

Algorithm in Vector form:

$Input\ W^l, a^1$
$for\ l = 1{:}L$
$\qquad a^{l+1} = fc\_v(W^l, a^l)$
$return$

Function $fc\_v(W^l, a^l)$

$$z^{l+1} = W^l a^l$$

$$a^{l+1} = f(z^{l+1})$$

end

Doted worms

$\begin{bmatrix}0\\1\end{bmatrix}$ $\begin{bmatrix}1\\0\end{bmatrix}$

Smooth worms

$\begin{bmatrix}0\\0\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$

Please fill your answer in the MATLAB file, submit it to the emails of TAs.

Thanks