Applied Machine Learning

Dr. Harikrishnan N B

Computer Science and Information Systems

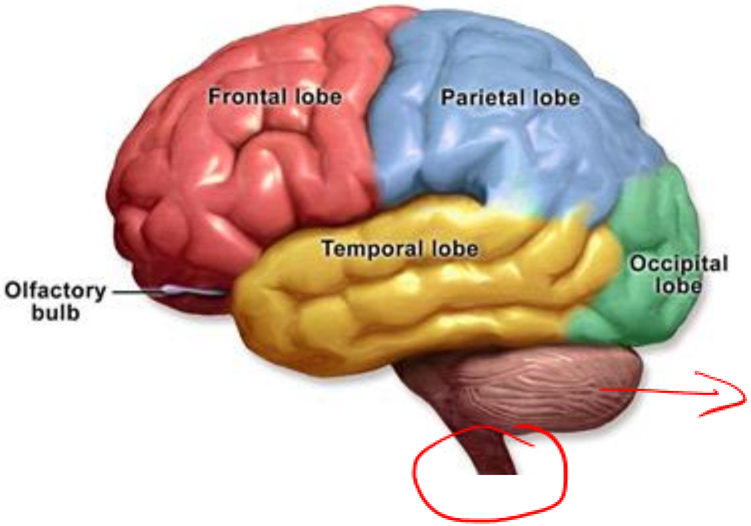
BITS Pilani

Pilani Campus

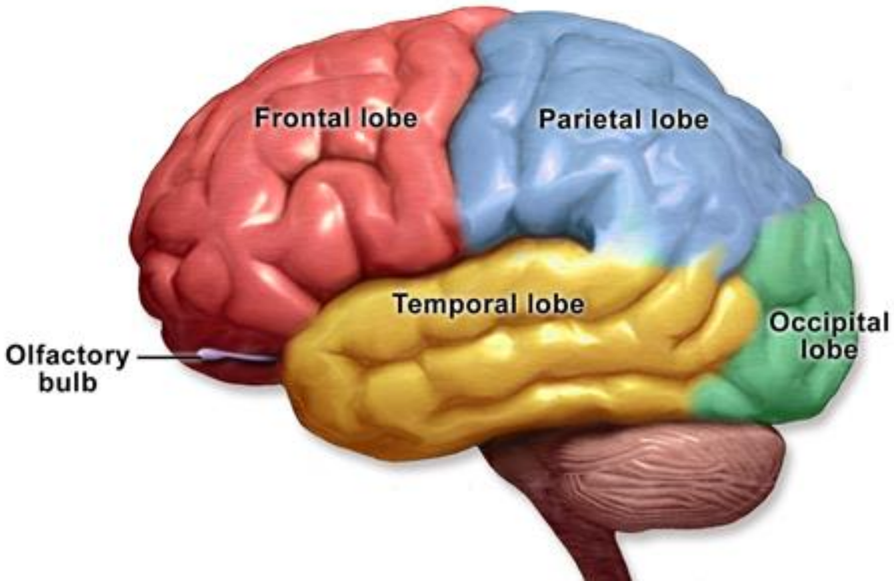
innovate achieve lead

SE ZG568 / SS ZG568, Applied Machine Learning Lecture No. 16

Human Brain


Frontal lobe
Parietal lobe
Temporal lobe
Occipital lobe
Olfactory bulb

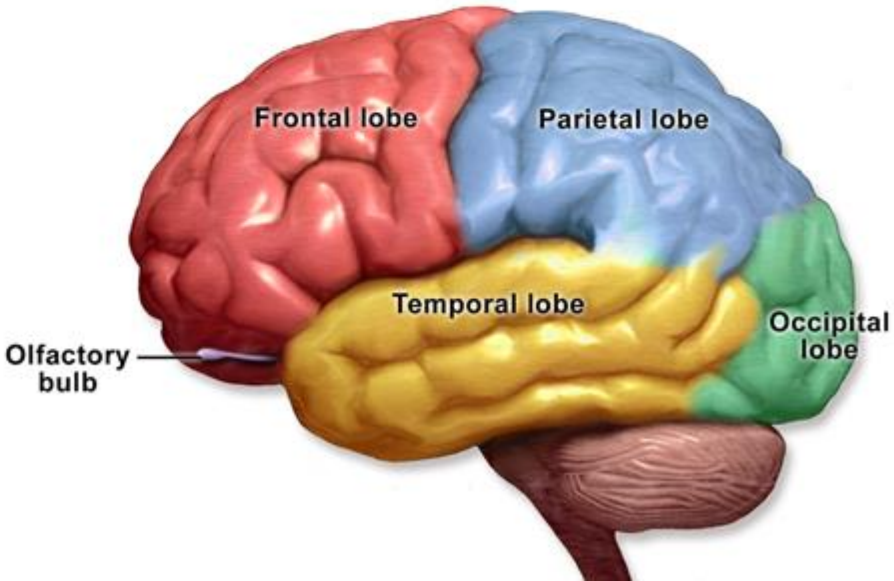
Frontal Lobe



Frontal Lobe

Motor Functions
Planning, Emotions, Social Behaviour

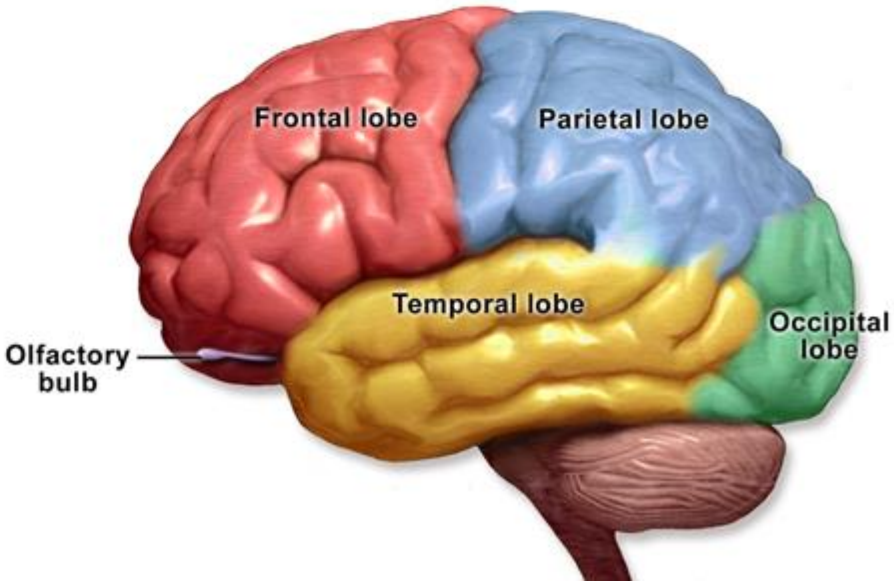
Parietal Lobe



Parietal Lobe

Sensations, touch, pain

Temporal Lobe



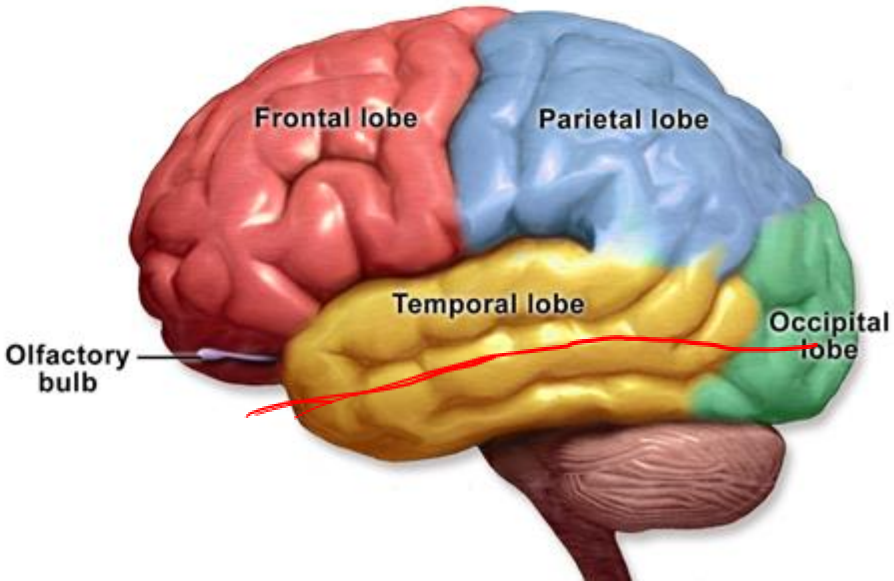
Temporal Lobe

Hearing, Memory

Occipital Lobe

Occipital Lobe

Vision, optic nerve arising from retina goes to occipital lobe through other lobes.

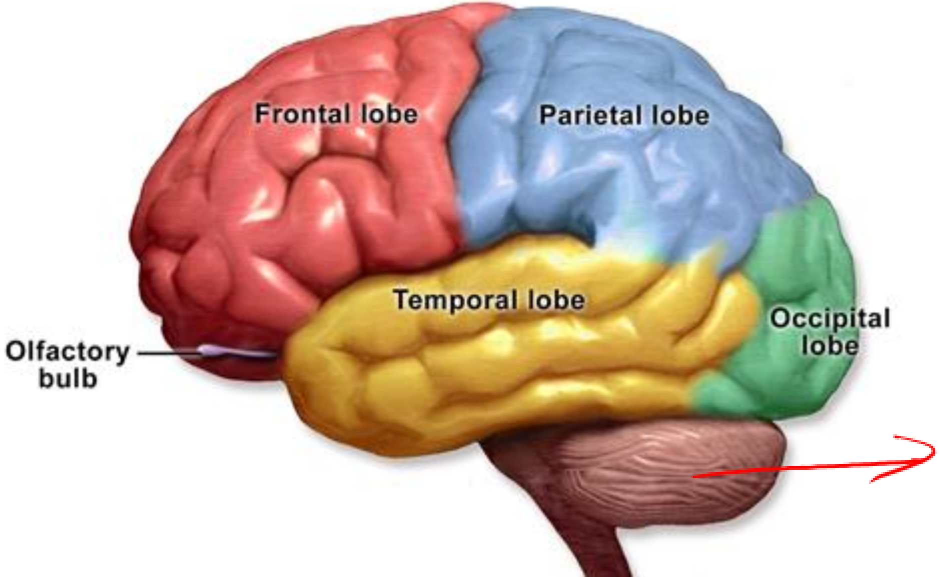


Brain Stem

Brain Stem

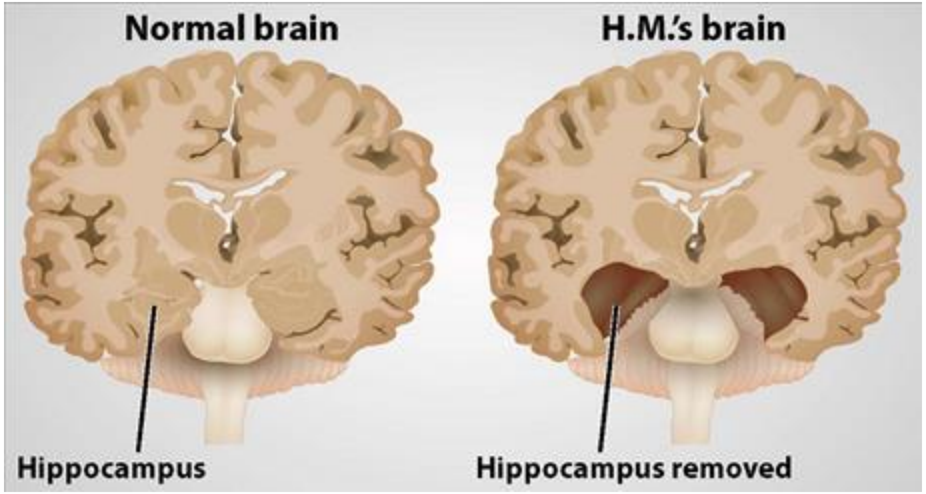
Mid Brain - controls eye movement
Pons - controls facial movement
Medulla Oblongata - control of respiration

86 billion neurons

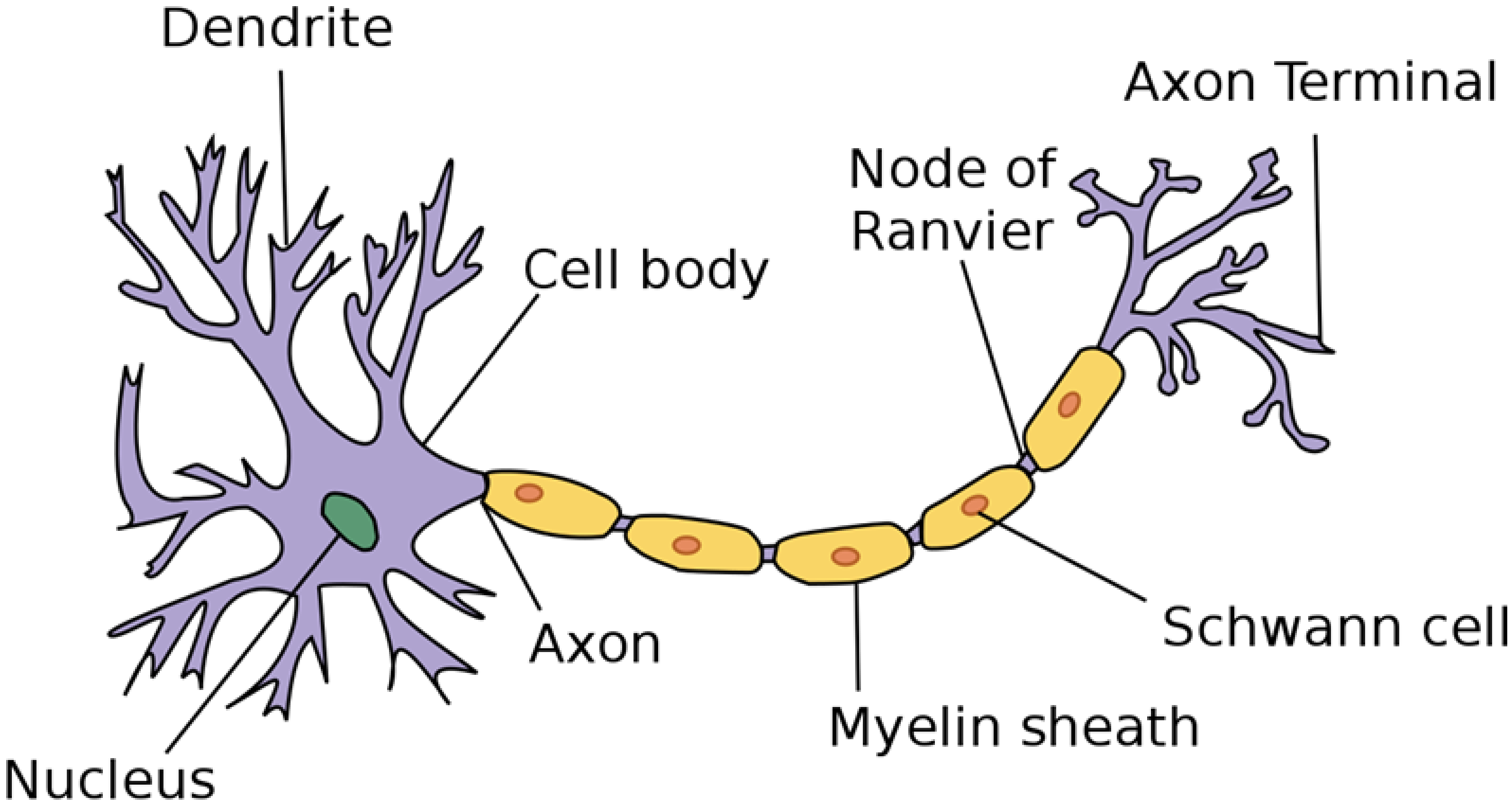


Patient HM and Memory

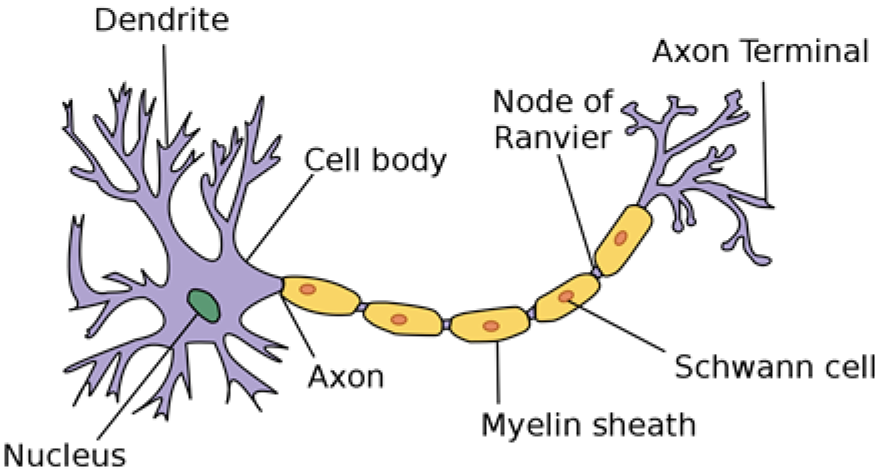
Patient HM (Henry Molaison)and Memory:
https://www.youtube.com/watch?v=i488aUN5RXA

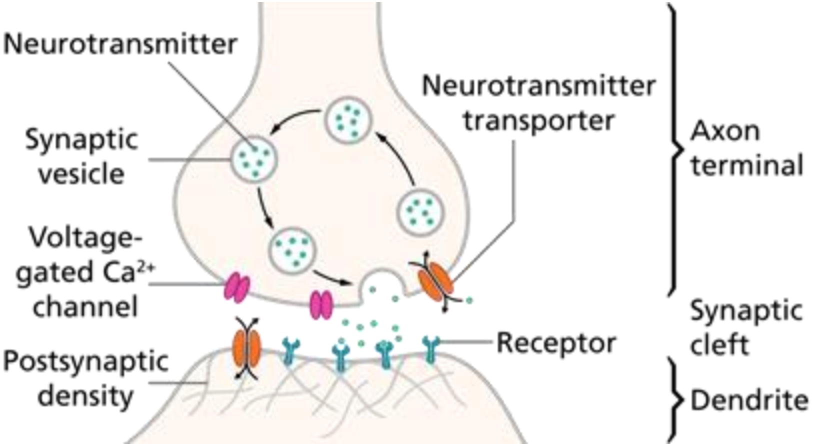


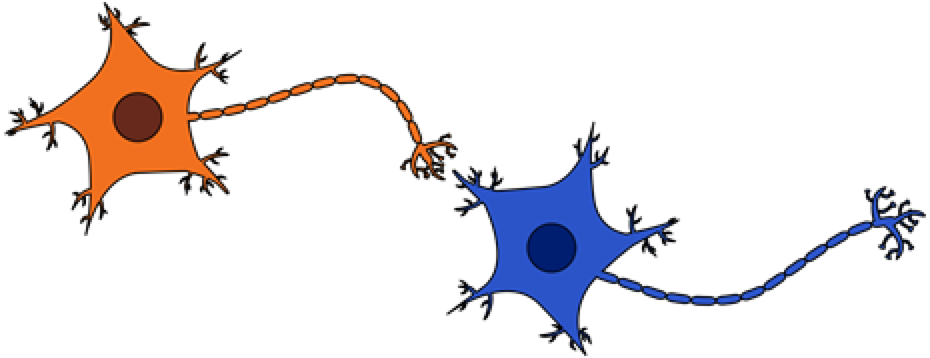
Lobes


Dendrite
Axon Terminal
Node of
Ranvier
Cell body
Axon
Schwann cell
Myelin sheath
Nucleus

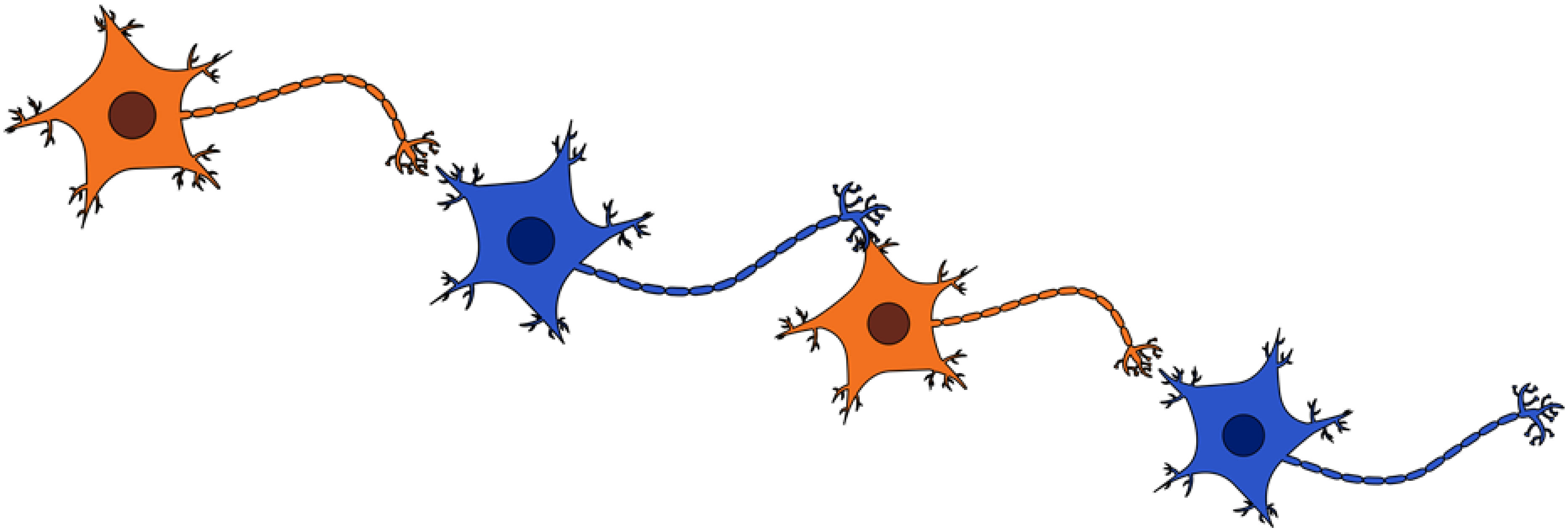
Lobes


Dendrite
Axon Terminal
Node of
Ranvier
Cell body
Axon
Schwann cell
Myelin sheath
Nucleus


Neurotransmitter
Neurotransmitter
transporter
Synaptic
vesicle
Voltage-
gated Ca2+
channel
Receptor
Postsynaptic
density
Axon
terminal
Synaptic
cleft
Dendrite



Neural Circuit



Brain: The Ultimate Machine

Stochastic Resonance

- High complexity (non-linear)
- Very high neural noise and interference
- Very low SNR: -29 dB to -20 dB*
- Neural signal multiplexing¶
- Low power “neural computation” (~12.6 Watts**)
- How does it work? No idea!

***Ref:** G. Czanner et al., Measuring the signal-to-noise ratio of a neuron, PNAS 112 (23), 2015
¶Ref: M L R Meister et al., Signal multiplexing and single-neuron computations in Lateral Intraparietal Area during decision-making, J. Neurosci. 33 (6), 2013
****Ref:** Scientific American, 18 July 2012

Neuroscience Inspired AI

- Rich source of inspiration for new types of algorithms and architectures, independent and complementary to the mathematical and logic based methods.
- Neuroscience can provide a validation of AI techniques that already exists.

Reinvestigate the current learning algorithms

- Brain science is still in *Faraday Stage* [1]
- Brain has ~86 billion neurons [2]
- Complex network of neurons
- Neurons are inherently **non-linear** & found to exhibit **Chaos**
- ***Current AI only loosely inspired from the brain***

1. Ramachandran, Vilayanur S., Sandra Blakeslee, and Neil Shah. *Phantoms in the brain: Probing the mysteries of the human mind*. 1998.
2. Azevedo, Frederico AC, et al. "Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain." *Journal of Comparative Neurology* 513.5 (2009): 532-541.

Artificial vs. Biological Neural Networks

Research and Development Gap

Artificial Neural Networks (ANN)	*Biological Neural Networks*
Linearity + Non-linear activation.	**Non-linearity at the neuronal level.** [3, 4]
Current deep learning architectures does not exhibit chaotic behaviour at the neuronal level for classification.	**Exhibits different behaviours - from periodic to chaotic at different spatiotemporal scales.**
Not robust to noise.	Robust to noise and interference.
Need huge amount of training data.	**Learning from limited samples.**

3. Faure, Philippe, and Henri Korn. "Is there chaos in the brain? I. Concepts of nonlinear dynamics and methods of investigation." *Comptes Rendus de l'Académie des Sciences-Series III-Sciences de la Vie* 324.9 (2001): 773-793.
4. Korn, Henri, and Philippe Faure. "Is there chaos in the brain? II. Experimental evidence and related models." *Comptes rendus biologies* 326.9 (2003): 787-840.

4 data instance

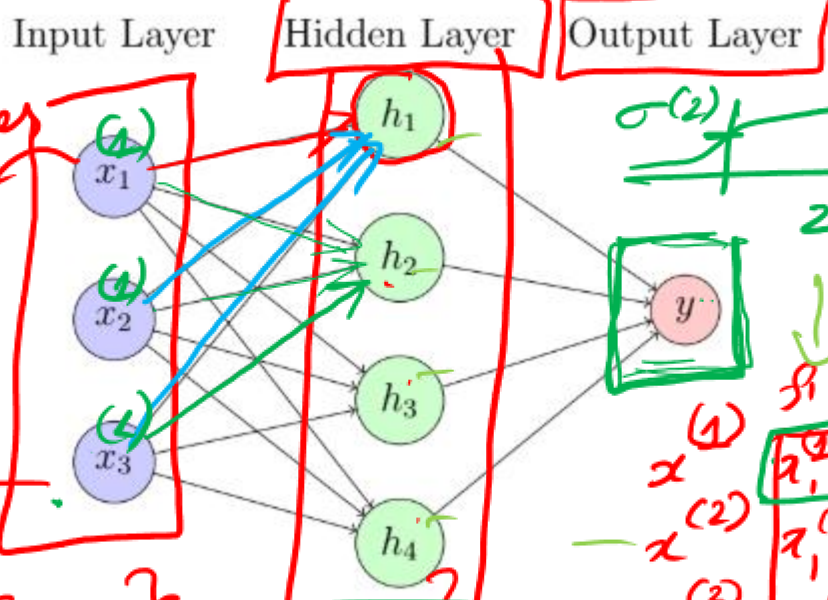
3 features

2 layer NN

1 Hidden layer

1 o/p layer

0.2 Analysis of a 2-Layer Neural Network



features

3 Nodes in the i/p layer

Neurons

$\sigma^{(2)}$

classification

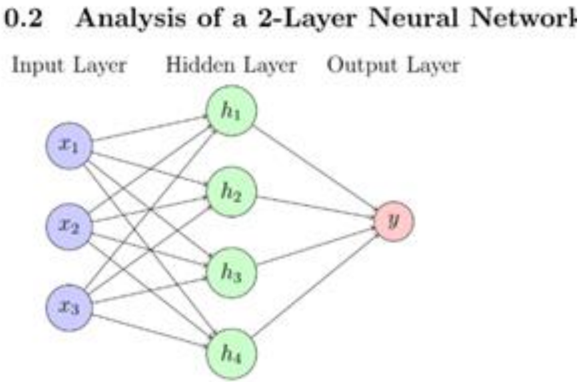
↳ binary

2 Layer NN architecture

$y^{(i)} \in \{0, 1\}$

	f_1	f_2	f_3	
$x^{(1)}$	$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$y^{(1)}$
$x^{(2)}$	$x_1^{(2)}$	$x_2^{(2)}$	$x_3^{(2)}$	$y^{(2)}$
$x^{(3)}$	$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$y^{(3)}$
$x^{(4)}$	$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$y^{(4)}$

Example



Forward Propagation
Computation of Loss
Backward Propagation
Updatqation of Weights and Bias

Test data

i/p layer

o/p layer

$\boxed{\text{softmax}}$

0 — 0.7 → 1 constant

1 — 0.2 — 0

2 — 0.1 — 0

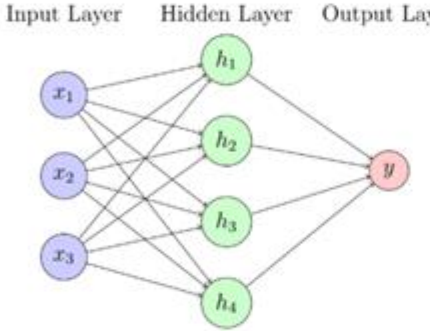
Sigmoid function

Forward Propagation

Hidden Layer 1 and Neuron 1

i^{th} data instance

0.2 Analysis of a 2-Layer Neural Network



h_3: $\sigma(w_{31}^{[1]} x_1^{(i)} + w_{32}^{[1]} x_2^{(i)} + w_{33}^{[1]} x_3^{(i)} + b_3^{[1]})$

h_4: $\sigma(w_{41}^{[1]} x_1^{(i)} + w_{42}^{[1]} x_2^{(i)} + w_{43}^{[1]} x_3^{(i)} + b_4^{[1]})$

h_2: $\sigma(w_{21}^{[1]} x_1^{(i)} + w_{22}^{[1]} x_2^{(i)} + w_{23}^{[1]} x_3^{(i)} + b_2^{[1]})$

[1] → Layer

w_{13} → from which i/p is contributing to the neuron.

1 → index of the neuron in the first hidden layer

$h_1 \Rightarrow \underbrace{w_{11}^{[1]} x_1^{(i)} + w_{12}^{[1]} x_2^{(i)} + w_{13}^{[1]} x_3^{(i)} + b_1^{[1]}}$

(activation sigmoidal activation) $\sigma(w_{11}^{[1]} x_1^{(i)} + w_{12}^{[1]} x_2^{(i)} + w_{13}^{[1]} x_3^{(i)} + b_1^{(1)})$

$$a_1^{[1]} = \sigma(w_{11}^{[1]} x_1^{(i)} + w_{12}^{[1]} x_2^{(i)} + w_{13}^{[1]} x_3^{(i)} + b_1^{(1)})$$

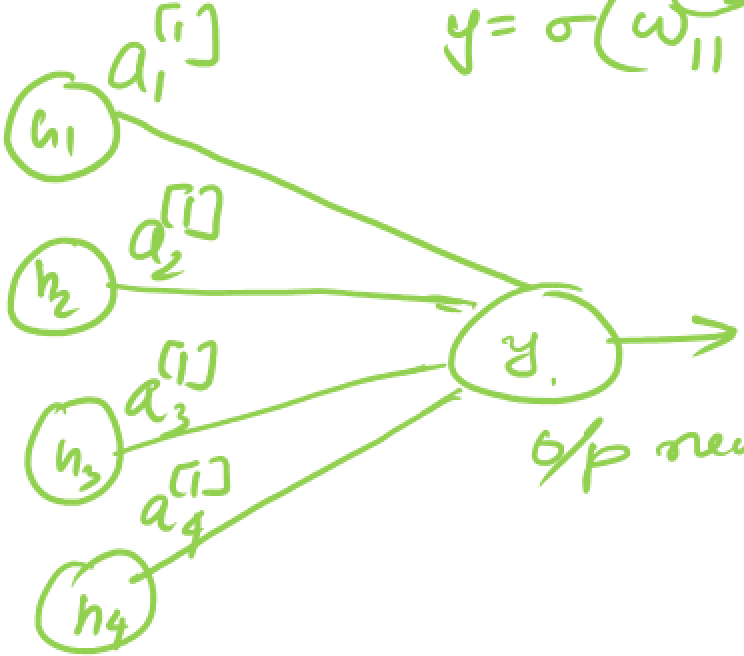
$a_2^{[1]}$

$a_3^{[1]}$

$a_4^{[1]}$

weights & bias are learnable

$$y = \sigma(w_{11}^{[2]} a_1^{[1]} + w_{12}^{[2]} a_2^{[1]} + w_{13}^{[2]} a_3^{[1]} + w_{14}^{[2]} a_4^{[1]} + b_1^{[2]})$$



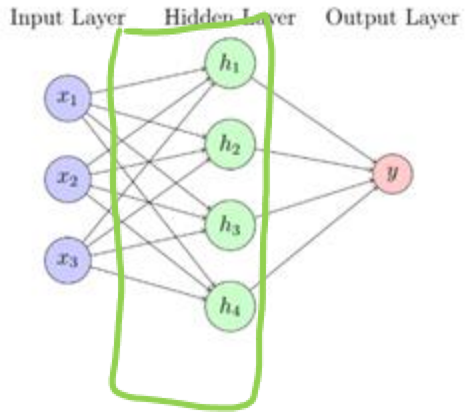
o/p neuron

Layer 2

$$x^{(i)} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \\ x_3^{(i)} \end{bmatrix} \qquad W^{[1]} x^{(i)} + b^{[1]}$$

For our example, $m = 4$, $n = 3$ and $n_h = 4$. We get the following matrices.

0.2 Analysis of a 2-Layer Neural Network



$$W^{[1]} = \begin{bmatrix} w_{11}^{[1]} & w_{12}^{[1]} & w_{13}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} & w_{23}^{[1]} \\ w_{31}^{[1]} & w_{32}^{[1]} & w_{33}^{[1]} \\ w_{41}^{[1]} & w_{42}^{[1]} & w_{43}^{[1]} \end{bmatrix} \qquad W^{[2]} = \begin{bmatrix} w_{11}^{[2]} & w_{12}^{[2]} & w_{13}^{[2]} & w_{14}^{[2]} \end{bmatrix}$$

$$b^{[1]} = \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \\ b_4^{[1]} \end{bmatrix} \qquad b^{[2]} = \begin{bmatrix} b_1^{[2]} \end{bmatrix}$$

$$W^{[2]} = \begin{bmatrix} w_{11}^{[2]} & w_{12}^{[2]} & w_{13}^{[2]} & w_{14}^{[2]} \end{bmatrix} \qquad b^{[2]} = \begin{bmatrix} b_1^{[2]} \end{bmatrix}$$

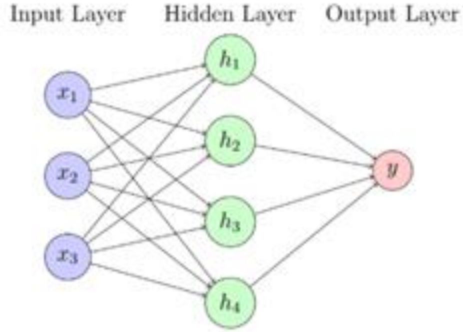
$$W^{[1]} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{[1]} \\ w_{21}^{[1]} & w_{22}^{[1]} & w_{23}^{[1]} \\ w_{31}^{[1]} & w_{32}^{[1]} & w_{33}^{[1]} \\ w_{41}^{[1]} & w_{42}^{(1)} & w_{43}^{[1]} \end{bmatrix} \qquad b^{[1]} = \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \\ b_4^{(1)} \end{bmatrix}$$

Applying sigmoidal on each rows

$$A^{[1]} = \sigma\left(W^{[1]} x^{(i)} + b^{[1]}\right) = \sigma\left(\begin{bmatrix} w_{11}^{(1)} x_1^{(i)} + w_{12}^{[1]} x_2^{(i)} + w_{13}^{[1]} x_3^{(i)} + b_1^{[1]} \\ w_{21}^{[1]} x_1^{(i)} + w_{22}^{(1)} x_2^{(i)} + w_{23}^{[1]} x_3^{(i)} + b_2^{[1]} \\ w_{31}^{[1]} x_1^{(i)} + w_{32}^{(1)} x_2^{(i)} + w_{33}^{[1]} x_3^{(i)} + b_3^{[1]} \\ w_{41}^{[1]} x_1^{(i)} + w_{42}^{[1]} x_2^{(i)} + w_{43}^{[1]} x_3^{(i)} + b_4^{[1]} \end{bmatrix}\right) = \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \\ a_4^{[1]} \end{bmatrix}$$

$$\sigma\left(W^{(2)} A^{(1)} + b^{[2]}\right) = y$$

0.2 Analysis of a 2-Layer Neural Network



i^{th} data instance

$$Z^{[1](i)} = W^{[1]}x^{(i)} + b^{[1]}$$

$$\boxed{a^{[1](i)} = \sigma(Z^{[1](i)})}$$

$$Z^{[2](i)} = W^{[2]}a^{[1](i)} + b^{[2]}$$

$$a^{[2](i)} = \sigma(Z^{[2](i)})$$

Computation of Loss

$z_1^{[2](1)}$

$W^{(1)}, W^{(2)}$

$b^{(1)}\ b^{(2)}$

$a_1^{[2](i)} = \sigma(w_{11}^{[2]} a_1^{[1](i)} + w_{12}^{[2]} a_2^{[1](i)} + \dots + w_{14}^{[2]} a_4^{[1](i)} + b_1^{[2]})$

$\sigma(W^{[2]} A^{[1](i)} + b^{[2]})$

Let the total loss be denoted by J.

J depends on $a_1^{[2](i)}$ $w_{11}^{[2]}$

$$J = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

i is 1 to m considering all training data.

$$J = \frac{1}{m}\sum_{i=1}^{m} L(a_1^{[2](i)}, y^{(i)}) = -\frac{1}{m}\sum_{i=1}^{m}(y^{(i)}log(a_1^{[2](i)}) + (1-y^{(i)})log(1-a_1^{[2](i)}))$$

binary cross entropy!

Update my weight & bias so that my J will be min

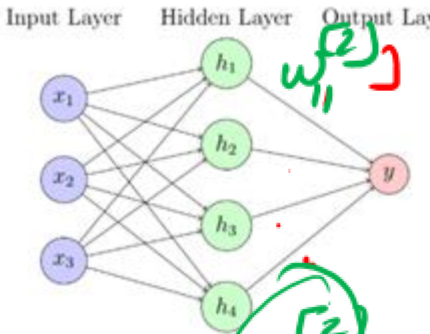
Backpropagation

$$\frac{\partial J}{\partial w^{(1)}}, \frac{\partial J}{\partial w^{(2)}}, \frac{\partial J}{\partial b^{(1)}}, \frac{\partial J}{\partial b^{(2)}}$$

Objective is to compute the following:

$$\frac{\partial J}{\partial W^{[1]}}, \frac{\partial J}{\partial W^{[2]}}, \frac{\partial J}{\partial b^{[1]}}, \frac{\partial J}{\partial b^{[2]}}$$

0.2 Analysis of a 2-Layer Neural Network



$w_{11}^{(2)}$

$$z_1^{[2]} = w_{11}^{(2)} a_1^{(1)} + w_{12}^{(2)} a_2^{(1)} + w_{13}^{(2)} a_3^{(1)} + w_{14}^{(2)} a_4^{(1)} + b_1^{(2)}$$

$$\frac{\partial J}{\partial w_{11}^{(2)}}$$

For the i^{th} training data, we can calculate the gradient for $w_{11}^{[2]}$ using chain rule as follows:

$$\boxed{\frac{\partial J^{(i)}}{\partial w_{11}^{[2]}} = \frac{\partial J^{(i)}}{\partial a^{[2](i)}} \times \frac{\partial a^{[2](i)}}{\partial z^{[2](i)}} \times \frac{\partial z^{[2](i)}}{\partial w_{11}^{[2]}}}$$

$$da^{[2](i)} = \frac{\partial J^{(i)}}{\partial a^{[2](i)}} = \frac{-y^{(i)}}{a^{[2](i)}} + \frac{1 - y^{(i)}}{1 - a^{[2](i)}}$$

$$\frac{\partial a^{[2](i)}}{\partial z^{[2](i)}} = a^{[2](i)}(1 - a^{[2](i)})$$

$$\frac{\partial z^{[2](i)}}{\partial w_{11}^{[2]}} = a_1^{[1]}$$

$$dw_{11}^{[2]} = \frac{\partial J^{(i)}}{\partial w_{11}^{[2]}} = \frac{-y^{(i)}}{a^{[2](i)}} + \frac{1 - y^{(i)}}{1 - a^{[2](i)}} \times a^{[2](i)}(1 - a^{[2](i)}) \times a_1^{[1]}$$

$$\boxed{= (-y^{(i)} + a^{[2](i)})a_1^{[1]}}$$

Now, extending the same for the other weights:

$$\boxed{\frac{\partial J^{(i)}}{\partial w_{12}^{[2]}} = (-y^{(i)} + a^{[2](i)})a_2^{[1]}}$$

$$a_1^{(2)} = \sigma(z_1^{(2)})$$

We can continue this gradient calculation to get the gradients corresponding to the weights and biases to the hidden layer:

$$\frac{\partial J^{(i)}}{\partial w_{11}^{[1]}} = (a^{[2](i)} - y^{(i)}) w_{11}^{[2]} a_1^{[1](i)} (1 - a_1^{[1](i)}) x_1^{(i)}$$

$$\frac{\partial J^{(i)}}{\partial w_{12}^{[1]}} = (a^{[2](i)} - y^{(i)}) w_{11}^{[2]} a_1^{[1](i)} (1 - a_1^{[1](i)}) x_2^{(i)}$$

$$\frac{\partial J^{(i)}}{\partial w_{21}^{[1]}} = (a^{[2](i)} - y^{(i)}) w_{12}^{[2]} a_2^{[1](i)} (1 - a_2^{[1](i)}) x_1^{(i)}$$

$$\frac{\partial J^{(i)}}{\partial w_{22}^{[1]}} = (a^{[2](i)} - y^{(i)}) w_{12}^{[2]} a_2^{[1](i)} (1 - a_2^{[1](i)}) x_2^{(i)}$$

We can continue this gradient calculation to get the gradients corresponding to the weights and biases to the hidden layer:

$$\frac{\partial J^{(i)}}{\partial w_{11}^{[1]}} = (a^{[2](i)} - y^{(i)})w_{11}^{[2]}a_1^{[1](i)}(1 - a_1^{[1](i)})x_1^{(i)}$$

$$\frac{\partial J^{(i)}}{\partial w_{12}^{[1]}} = (a^{[2](i)} - y^{(i)})w_{11}^{[2]}a_1^{[1](i)}(1 - a_1^{[1](i)})x_2^{(i)}$$

$$\frac{\partial J^{(i)}}{\partial w_{21}^{[1]}} = (a^{[2](i)} - y^{(i)})w_{12}^{[2]}a_2^{[1](i)}(1 - a_2^{[1](i)})x_1^{(i)}$$

$$\frac{\partial J^{(i)}}{\partial w_{22}^{[1]}} = (a^{[2](i)} - y^{(i)})w_{12}^{[2]}a_2^{[1](i)}(1 - a_2^{[1](i)})x_2^{(i)}$$

Weight Updation

gradient descent

$$W^{[1]} \leftarrow W^{[1]} - \alpha \frac{\partial J}{\partial W^{[1]}}$$

$$W^{[2]} \leftarrow W^{[2]} - \alpha \frac{\partial J}{\partial W^{[2]}}$$

$$b^{[1]} \leftarrow b^{[1]} - \alpha \frac{\partial J}{\partial b^{[1]}}$$

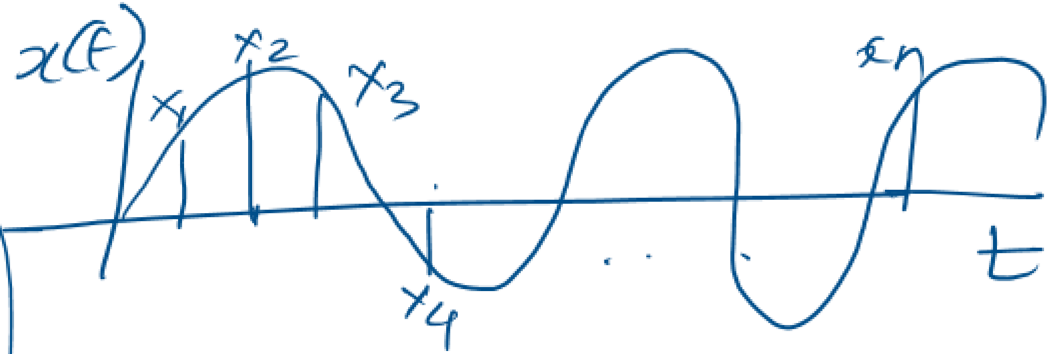
$$b^{[2]} \leftarrow b^{[2]} - \alpha \frac{\partial J}{\partial b^{[2]}}$$

1 forward
1 backward } 1 epoch
↓
1 weight updation

100 epochs

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$

$n = 4$



$$\frac{dx(t)}{dt} = \frac{x(t+1) - x(t)}{\Delta t} \quad \frac{}{1}$$

$n = 4$

$$\underline{\underline{D_{3\times 4}}} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}_{4\times 1} = \begin{bmatrix} x_2 - x_1 \\ x_3 - x_2 \\ \vdots \\ x_n - x_{n-1} \end{bmatrix}_{3\times 1}$$

Neural N/ws

forward propagation, Computation of loss

Backward ", weight updation.