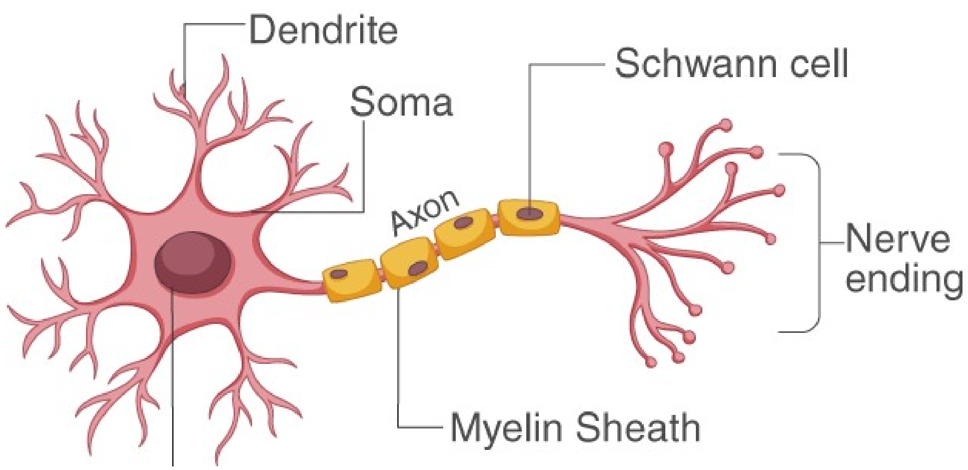
Discovering AI

Concept level

Discovering AI
Sasan Azimi

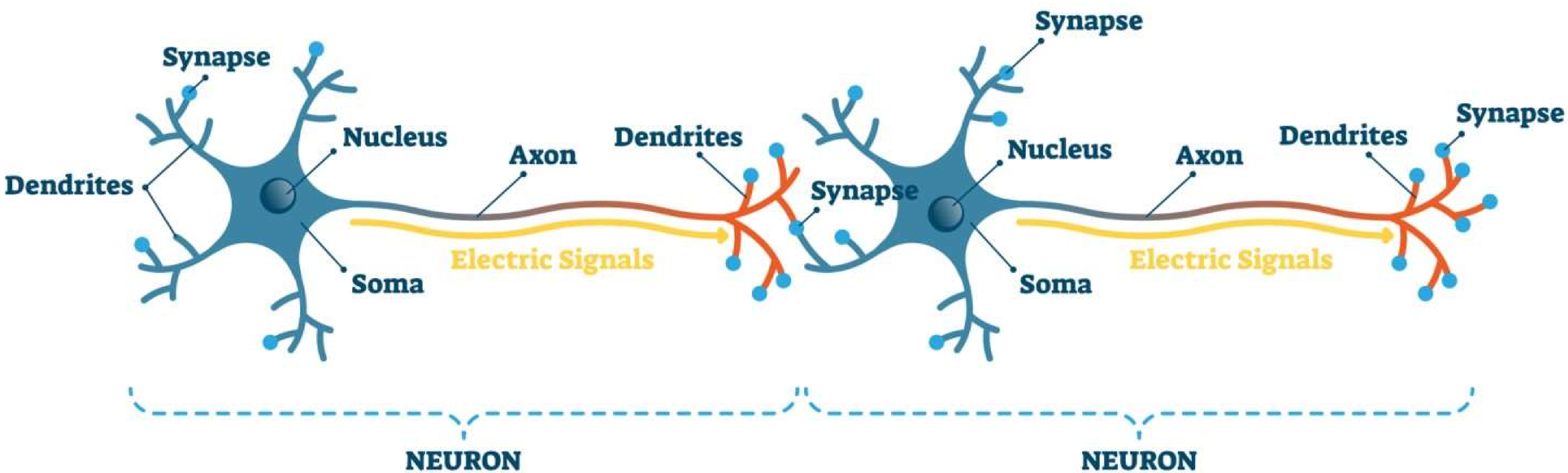
STRUCTURE OF NEURON

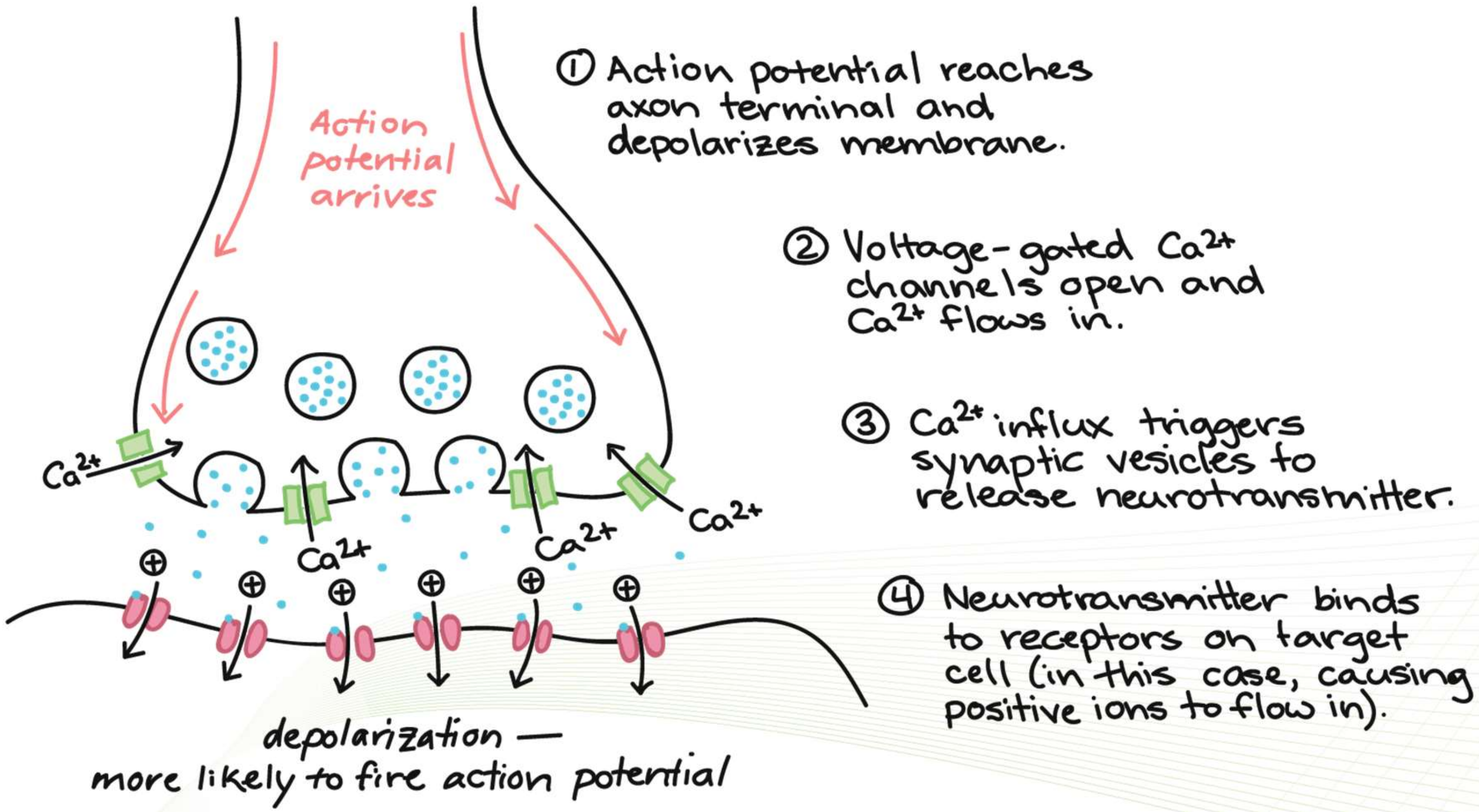


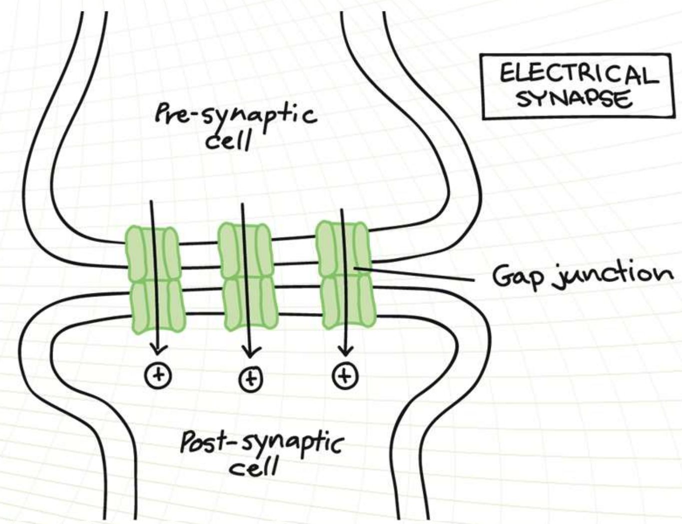
Number of Neurons: 86 billon

A neuron is connected to 10,000 to 100,000 other neurons

Average connection: 1000

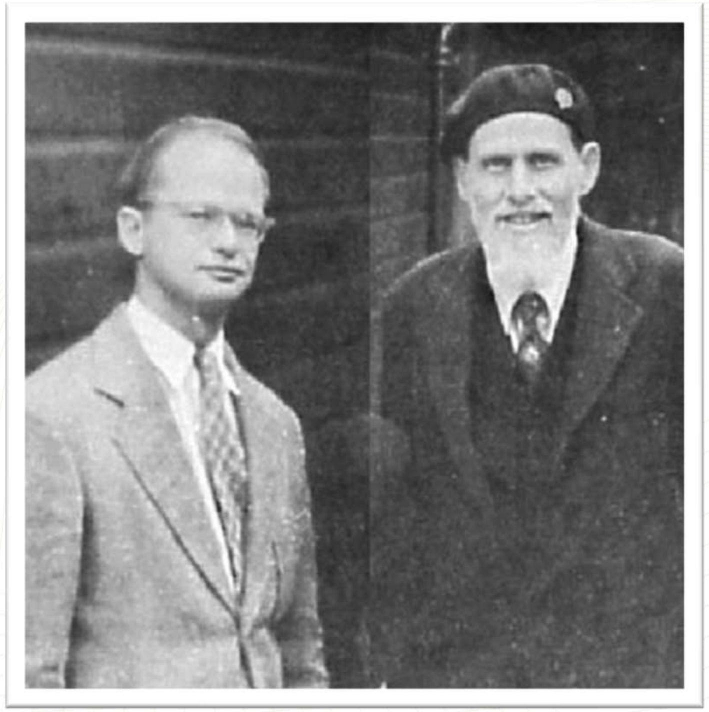



Action potential arrives
Ca2+
Ca2+
Ca2+
Ca2+
depolarization — more likely to fire action potential
① Action potential reaches axon terminal and depolarizes membrane.
② Voltage-gated Ca2+ channels open and Ca2+ flows in.
③ Ca2+ influx triggers synaptic vesicles to release neurotransmitter.
④ Neurotransmitter binds to receptors on target cell (in this case, causing positive ions to flow in).

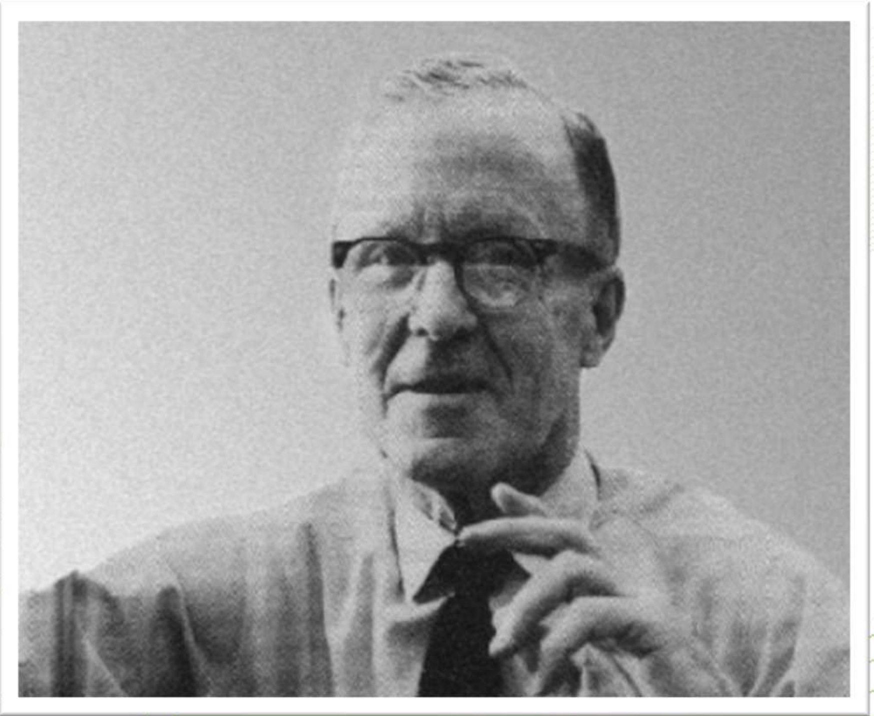

ELECTRICAL SYNAPSE
Pre-synaptic cell
Gap junction
Post-synaptic cell

History

Walter Pitts and Warren McCulloch
(1943)



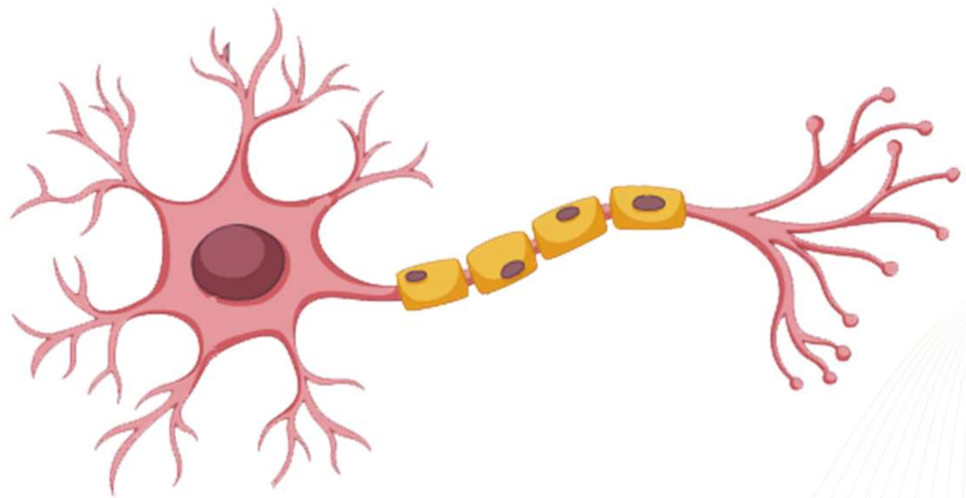
- First model of neurons with the On/Off capability
- Firing in response to sufficient stimulation from surrounding neurons

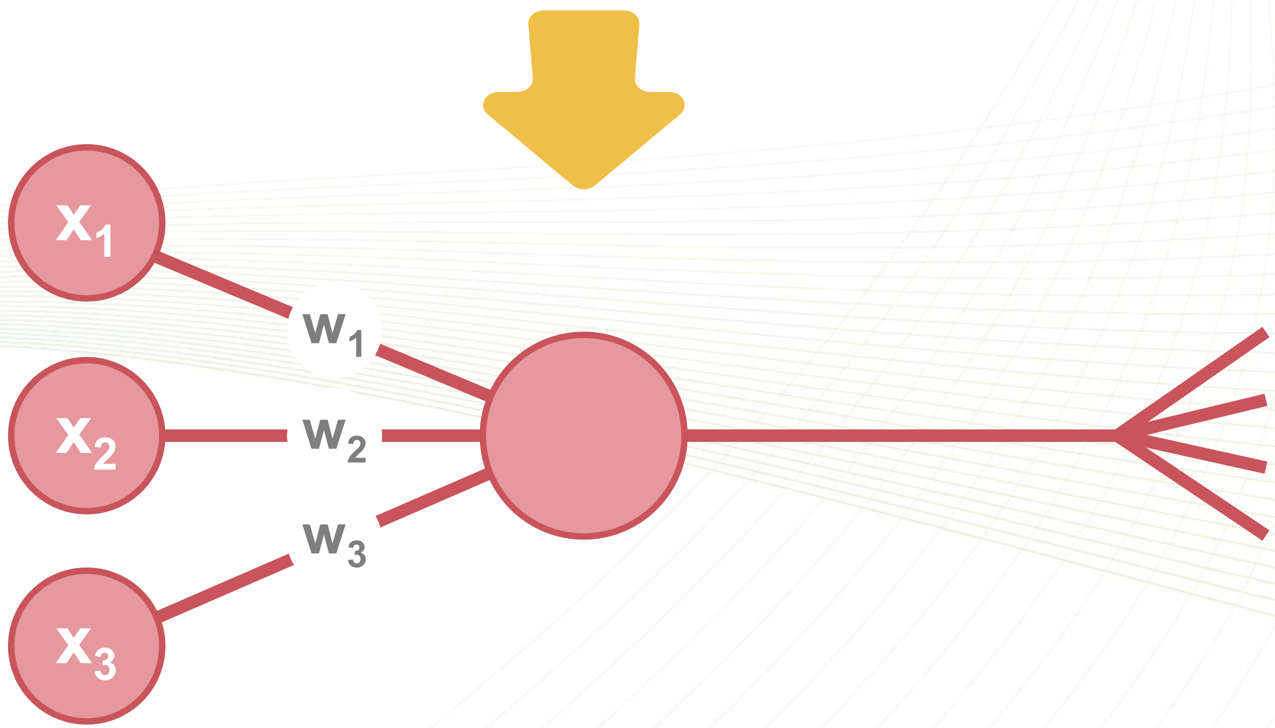


Donald Hebb
(1949) Psychology

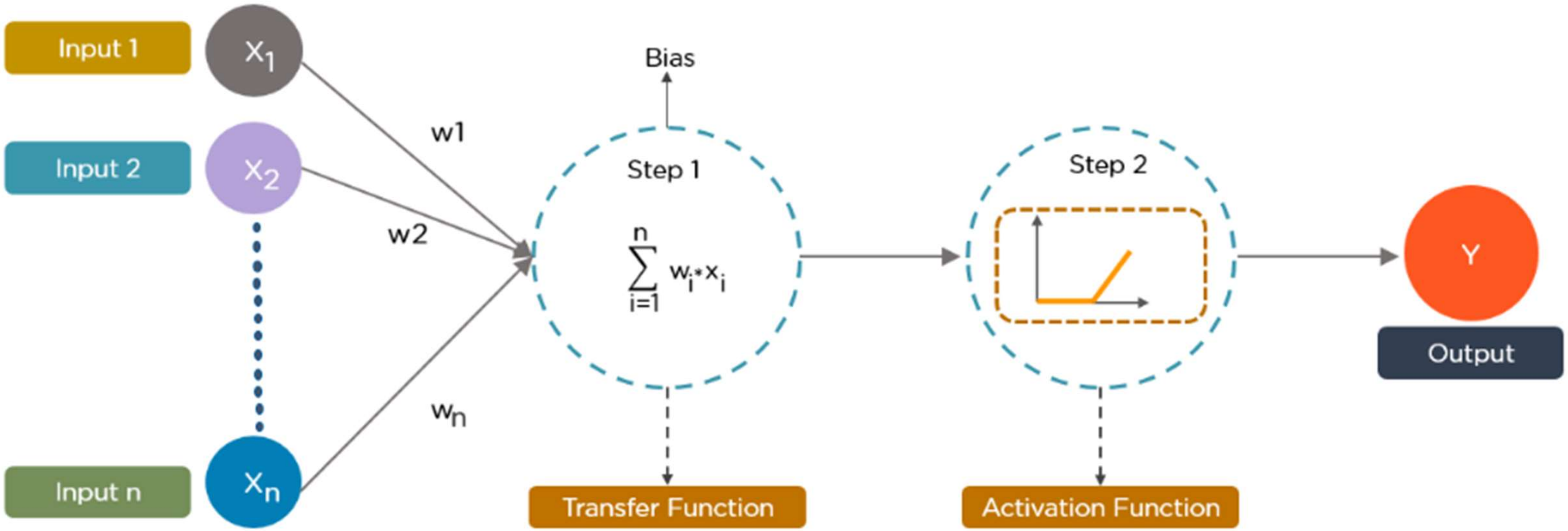
- Inventing the concept of learning in artificial neurons
- A simple method to update the connection strength between neurons

Reference: www.simplypsychology.org





Reference: www.simplypsychology.org



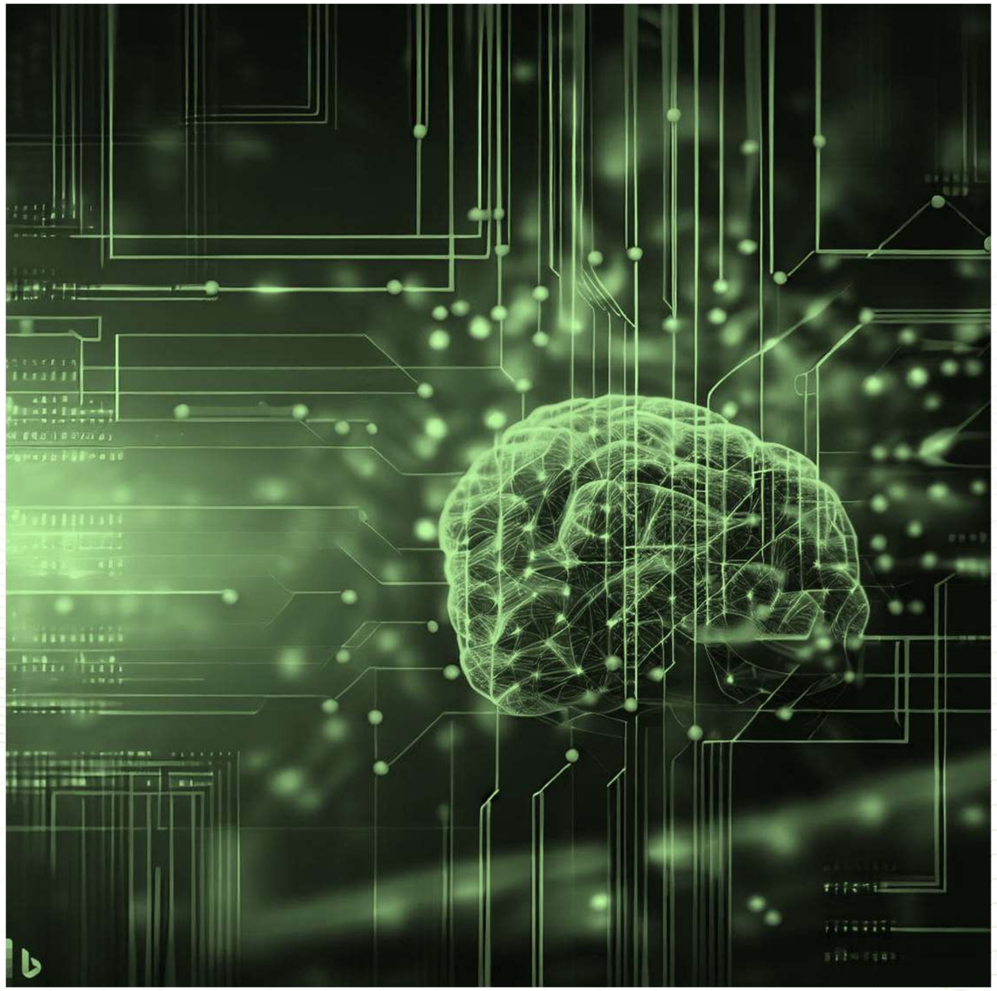
- https://www.simplilearn.com/tutorials/deep-learning-tutorial/deep-learning-algorithm

Types of Data

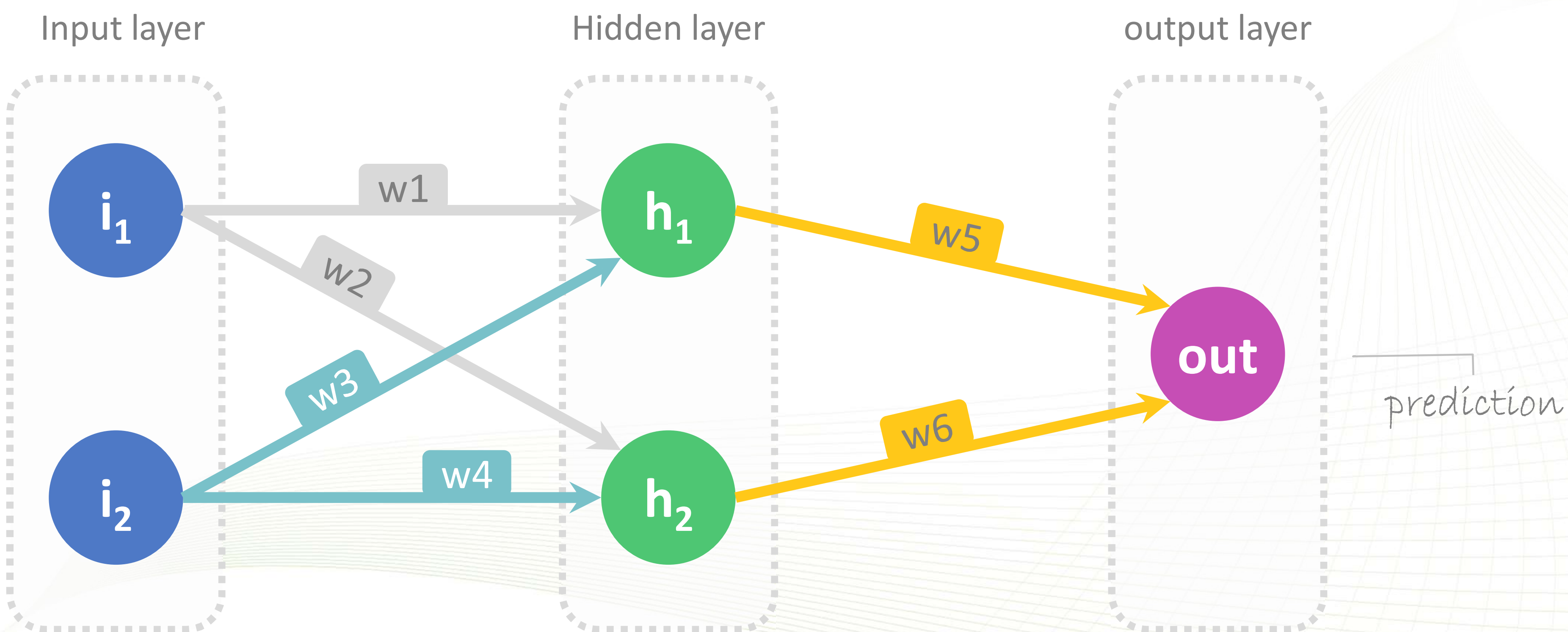
Tables	(Structured data)
XML, json	(Semi-Structured data)
Image, audio, text	(unstructured data)

Machine learning

- Supervised Machine learning
- Unsupervised Machine learning
- Semi-supervised learning
- Reinforcement Machine learning
- Self-supervised learning

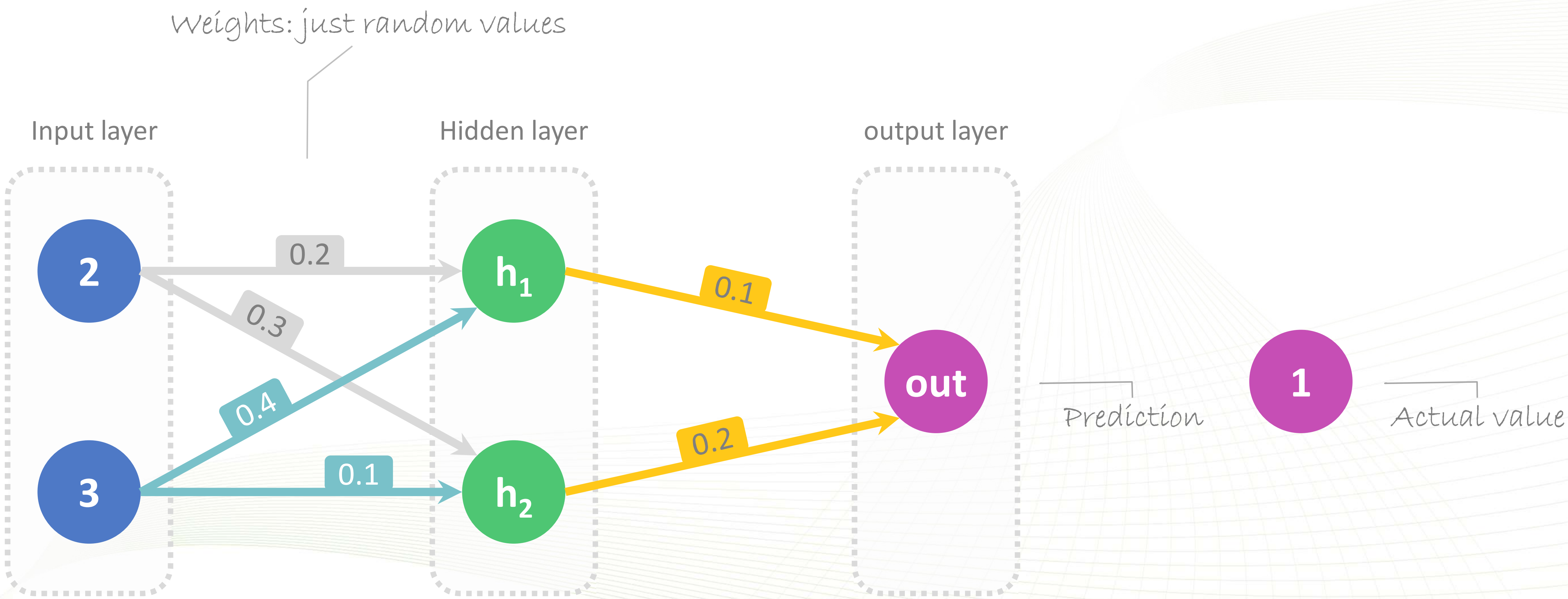


gorilla fishing in the ocean,

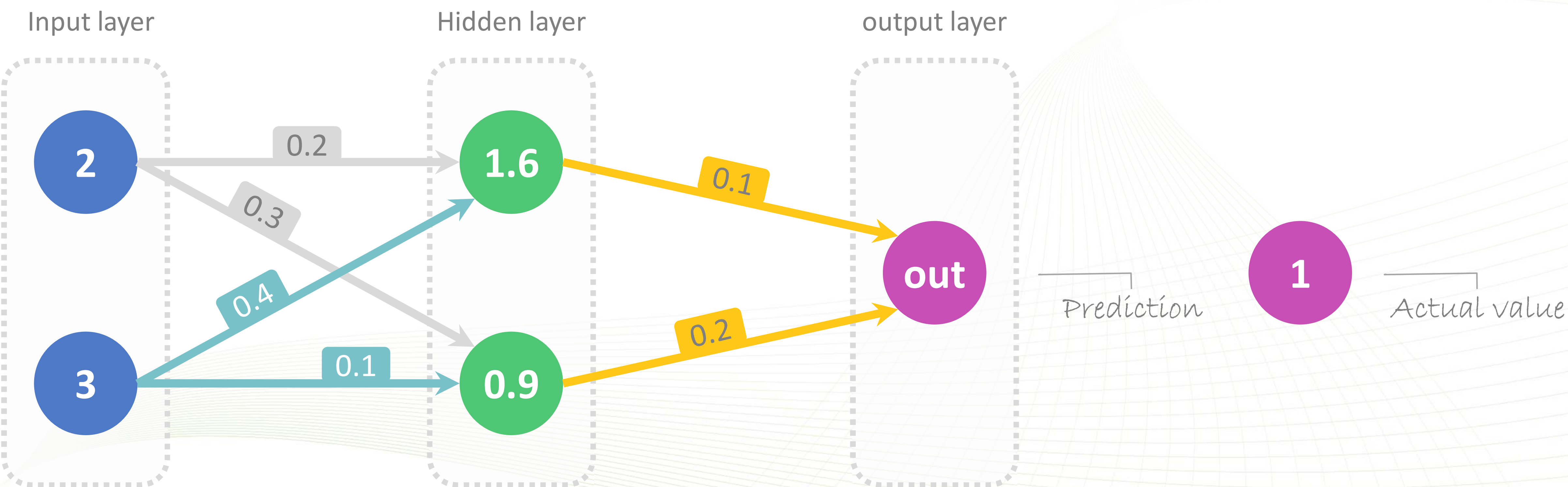


$$h_1 = i_1 w_1 + i_2 w_3$$
$$h_2 = i_1 w_2 + i_2 w_4$$

$$out = h_1 w_5 + h_2 w_6$$

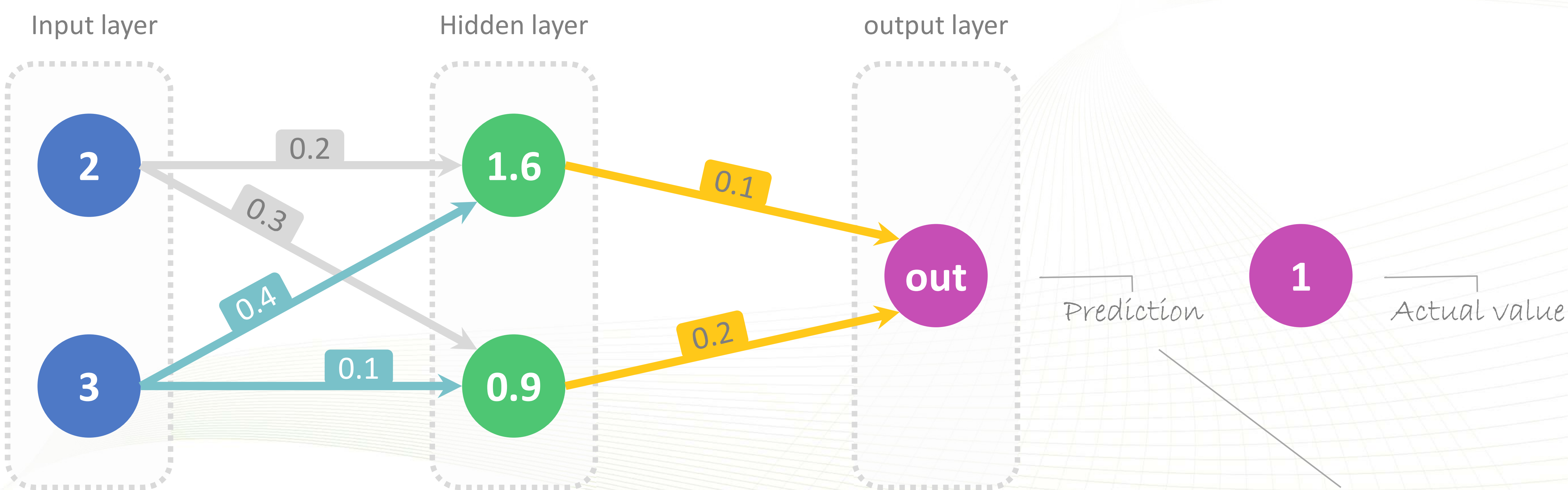


$$h_1 = 2 \times 0.2 + 3 \times 0.4 = 1.6$$
$$h_2 = 2 \times 0.3 + 3 \times 0.1 = 0.9$$



$$h_1 = 2 \times 0.2 + 3 \times 0.4 = 1.6$$

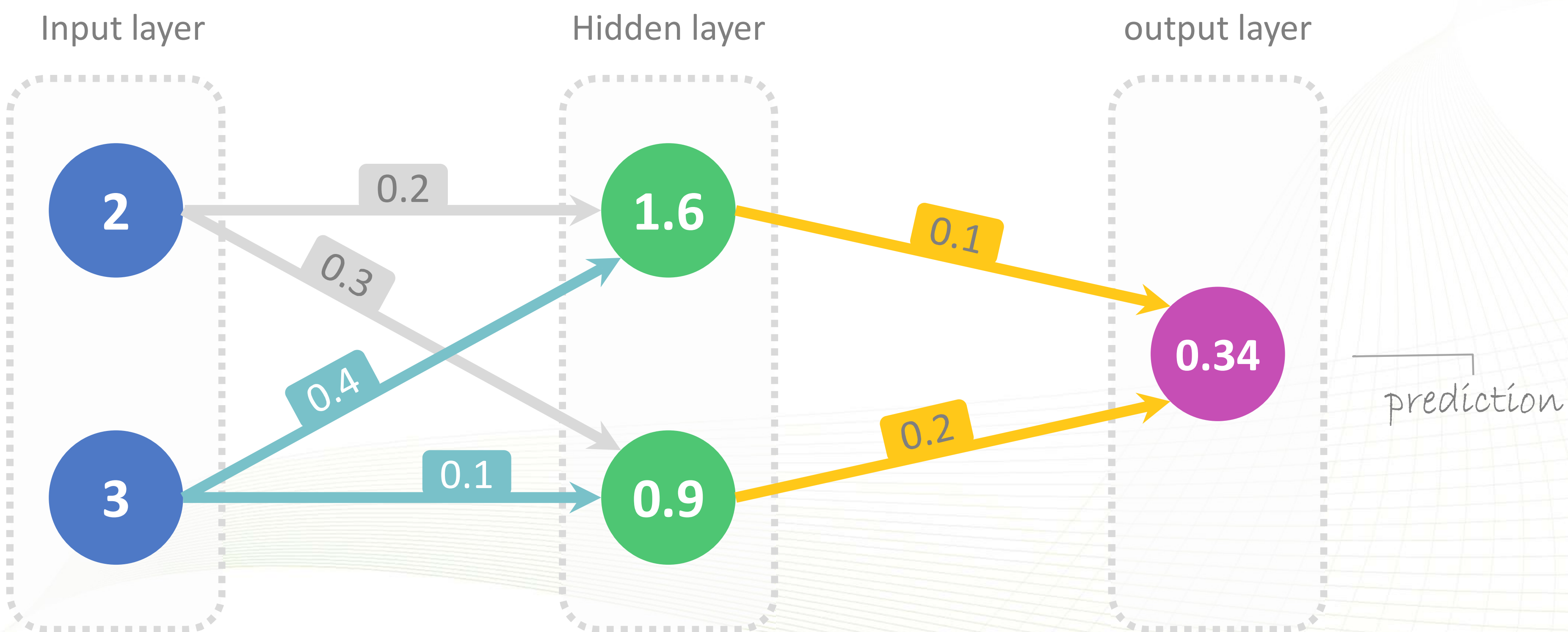
$$h_2 = 2 \times 0.3 + 3 \times 0.1 = 0.9$$



$$out = 1.6 \times 0.1 + 0.9 \times 0.2 = 0.34$$

$$h_1 = 2 \times 0.2 + 3 \times 0.4 = 1.6$$

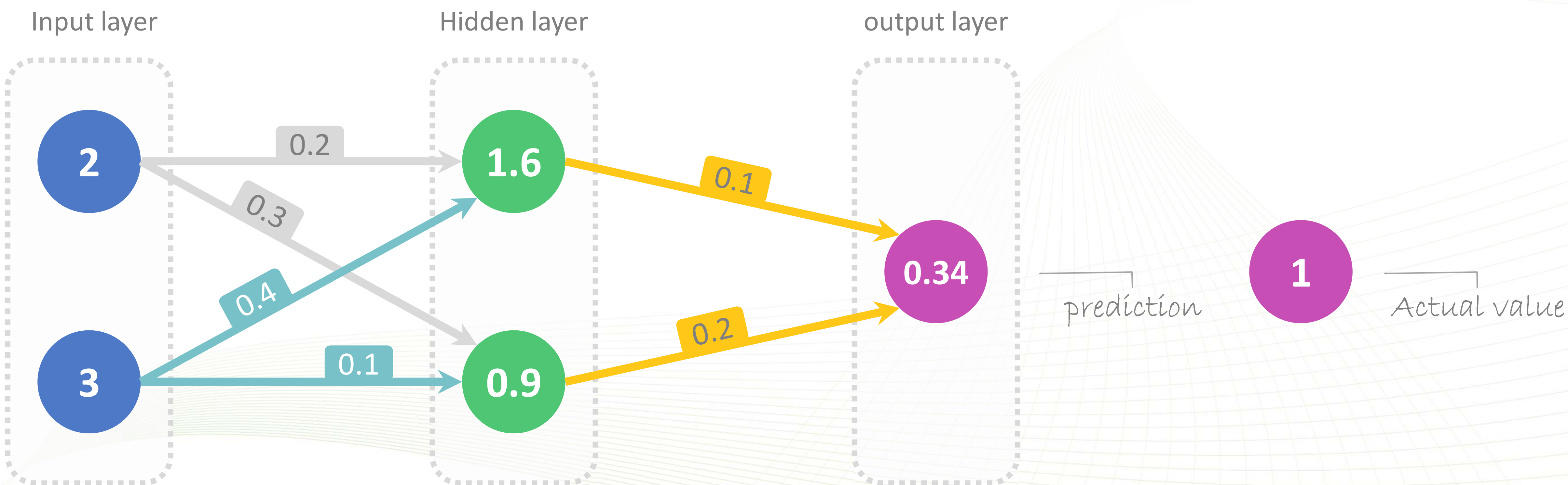
$$h_2 = 2 \times 0.3 + 3 \times 0.1 = 0.9$$



$$out = 1.6 \times 0.1 + 0.9 \times 0.2 = 0.34$$

$$h_1 = 2 \times 0.2 + 3 \times 0.4 = 1.6$$

$$h_2 = 2 \times 0.3 + 3 \times 0.1 = 0.9$$



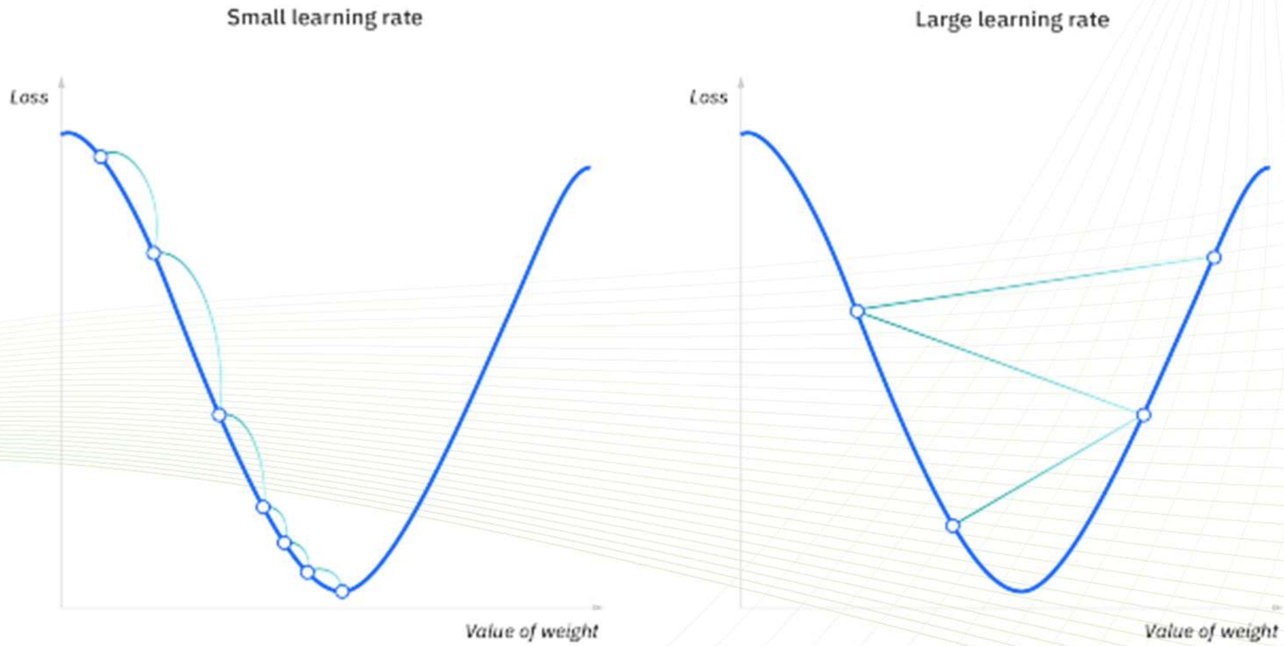
$$error = \frac{1}{2}(predictioin\ - actual\ value)^2$$

$$\frac{1}{2}(0.34\ - 1)^2 = 0.218$$

Why squared?

why multiply by ½

Learning rate



Backpropagation

Which one is changeable?

Error= actual value - prediction

Target: minimizing the error

How could we change the prediction?

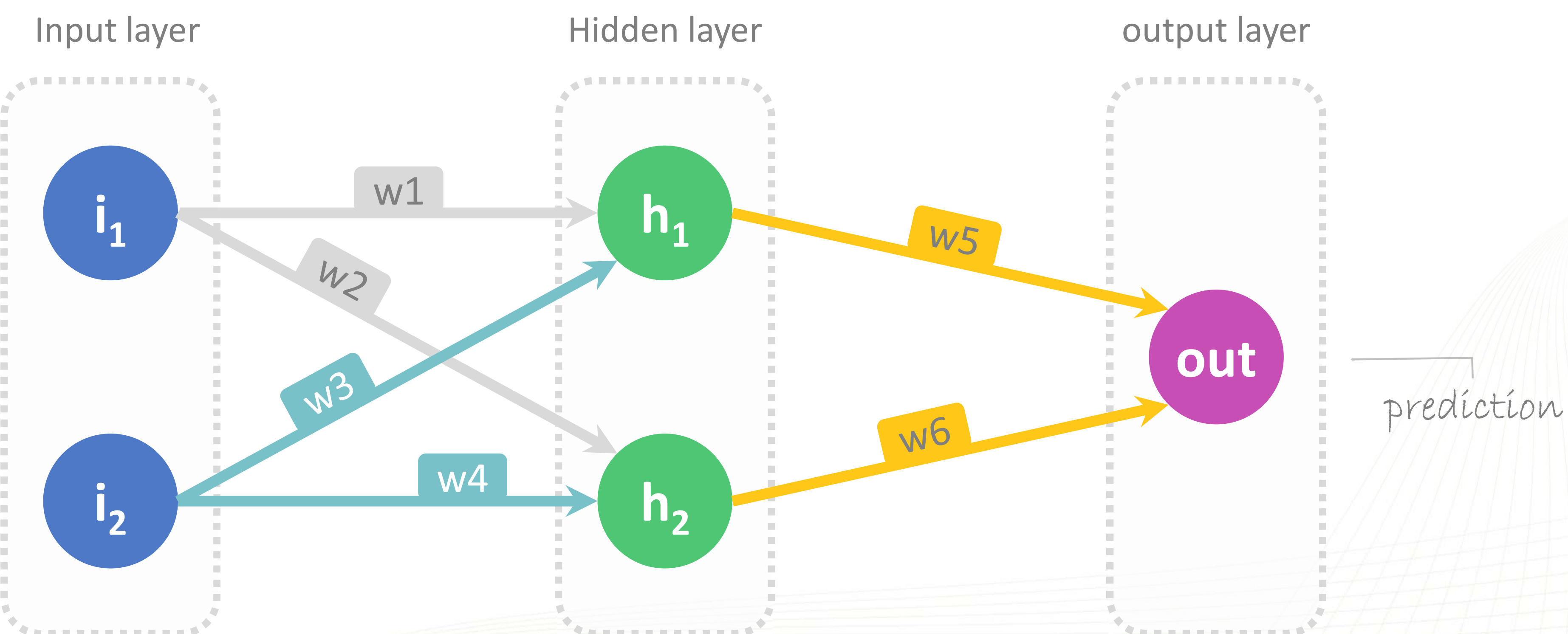
Using derivation to recognize the direction of movement → Gradient descent

Improving the prediction by changing the weights

$$w_x = w_x - a\left(\frac{\partial\ error}{\partial\ w_x}\right)$$

Learning rate

$$\frac{\partial\ (x^2y^3 + xy^2)}{\partial\ x} = 2xy^3 + y^2$$



$$h_1 = i_1 w_1 + i_2 w_3$$
$$h_2 = i_1 w_2 + i_2 w_4$$

$$out = h_1 w_5 + h_2 w_6$$

$$out = (i_1 w_1 + i_2 w_3) w_5 + (i_1 w_2 + i_2 w_4) w_6$$

i_1 و i_2 fixed

$$y = u^n \cdots\rightarrow \acute{y} = n.\acute{u}.u^{n-1}$$

$$error = \frac{1}{2}(\overbrace{predictioin\ - actual\ value}^{u\ \ = \delta})^2$$

$$w_x = w_x - a\left(\frac{\partial\ error}{\partial\ w_x}\right)$$

$$w_5 = w_5 - a\underbrace{\left(\frac{\partial\ error}{\partial\ w_5}\right)}$$

$$u = \overbrace{h_1 w_5 + h_2 w_6} - 1 = \delta$$

$$\acute{u} = h_1$$

$$err\acute{o}r = n.\acute{u}.u^{n-1} = \frac{1}{2} 2h_1\delta$$

$$= h_1\delta$$

$$w_5 = w_5 - a(h_1\delta)$$

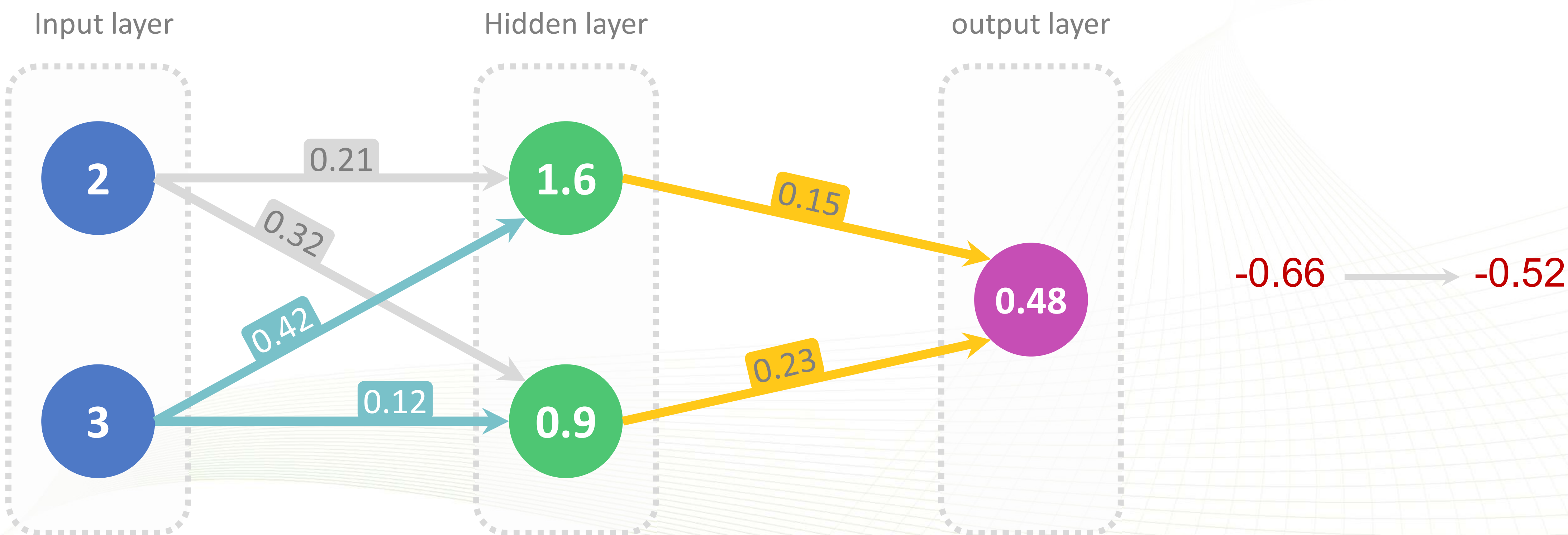
$$w_6 = w_6 - a(h_2\delta)$$

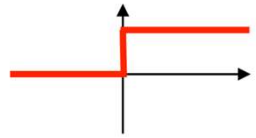
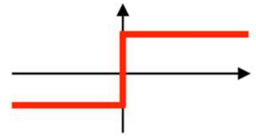
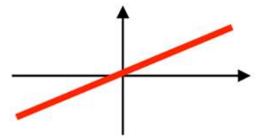
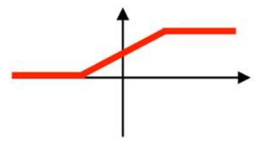
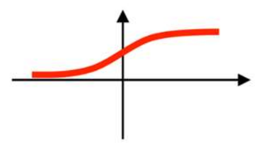
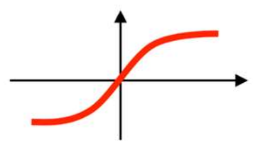
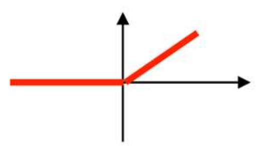
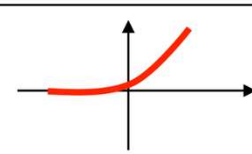
$$w_4 = w_4 - a(i_2.\delta w_6)$$

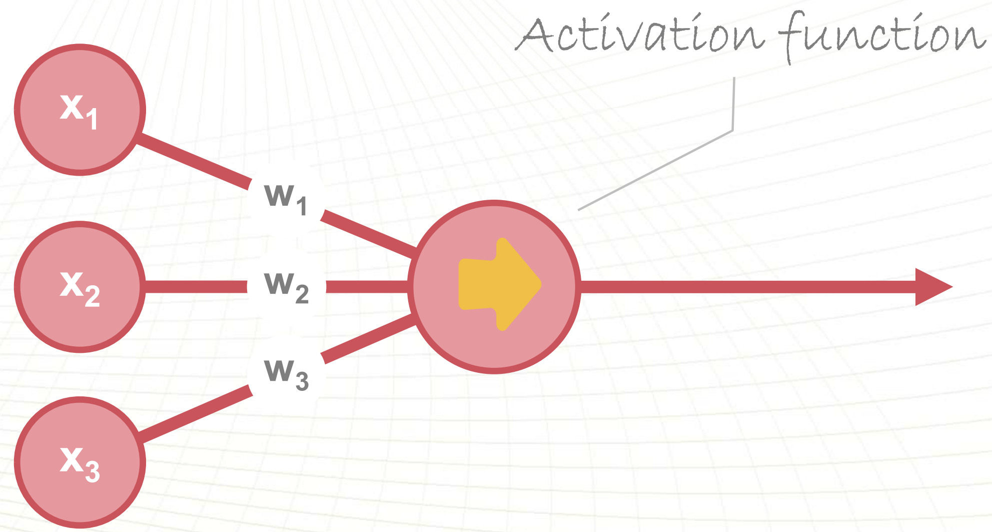
$$w_3 = w_3 - a(i_1.\delta w_6)$$

$$w_2 = w_2 - a(i_2.\delta w_5)$$

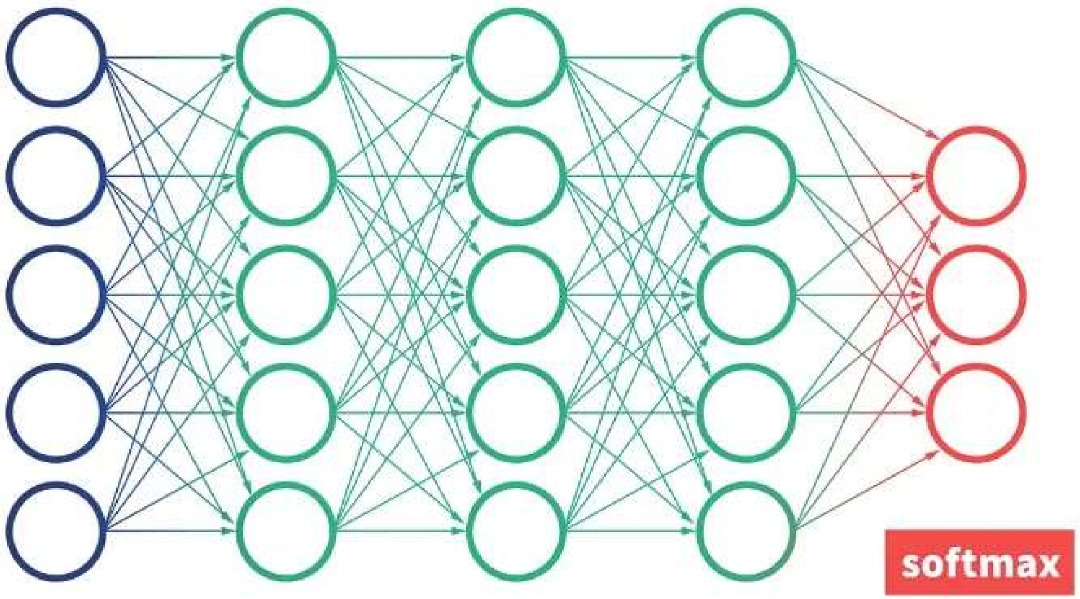
$$w_1 = w_1 - a(i_1.\delta w_5)$$

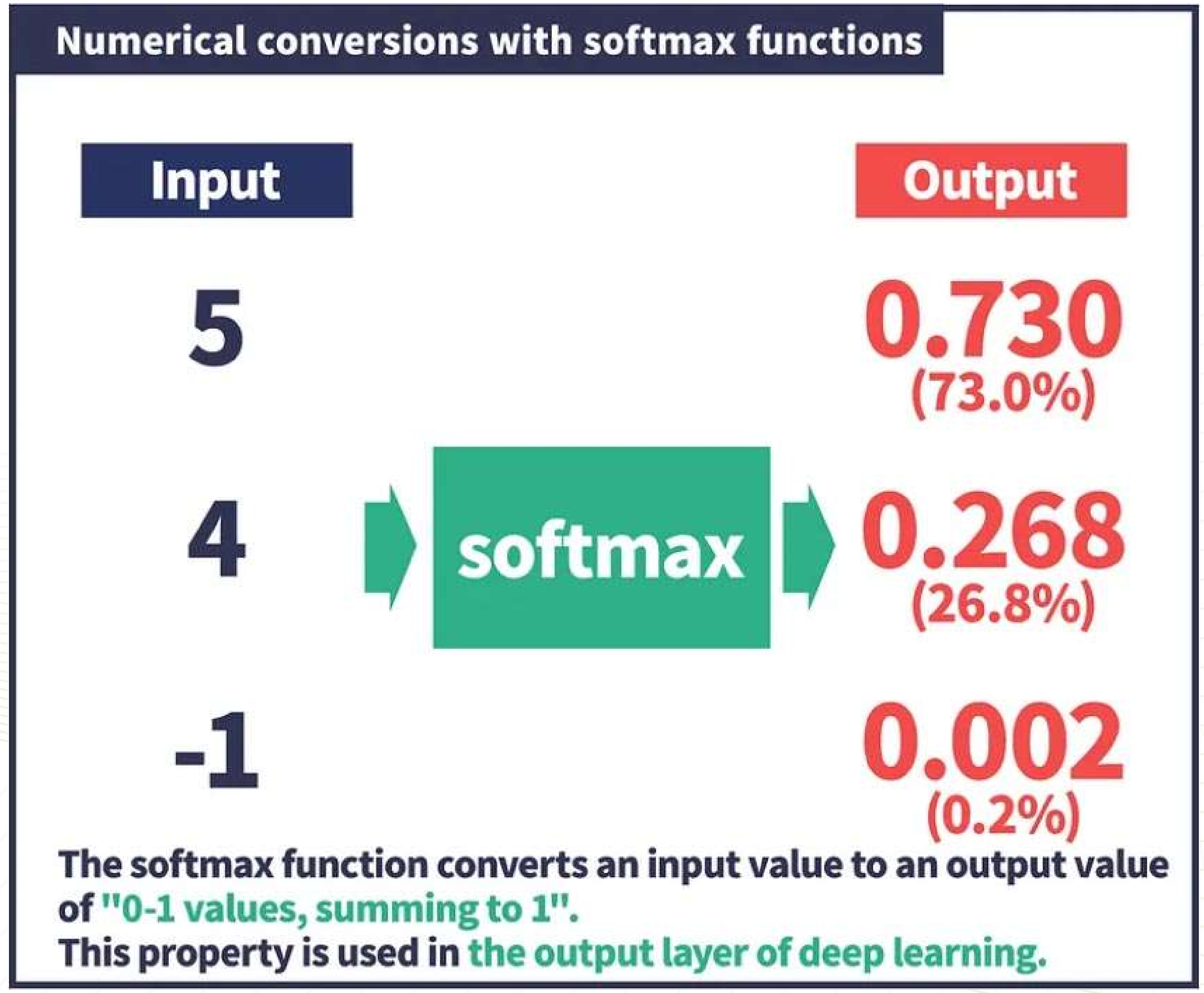

Input layer
Hidden layer
output layer
2
3
0.21
0.32
0.42
0.12
1.6
0.9
0.15
0.23
0.48
-0.66
-0.52

Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer Neural Networks	
Rectifier, ReLU (Rectified Linear Unit)	$\phi(z) = max(0, z)$	Multi-layer Neural Networks	
Rectifier, softplus	$\phi(z) = \ln(1 + e^z)$	Multi-layer Neural Networks	



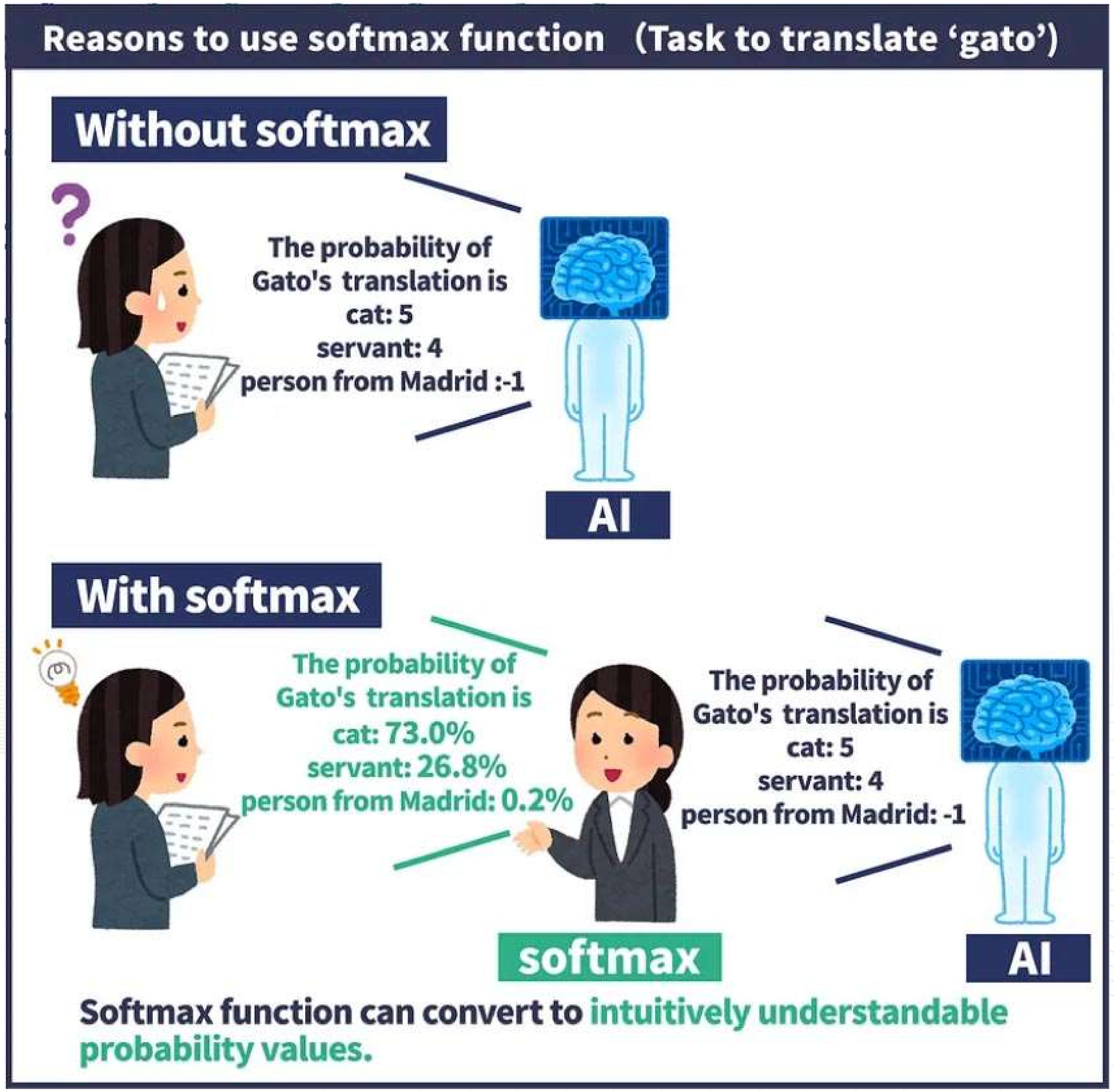
Softmax Function

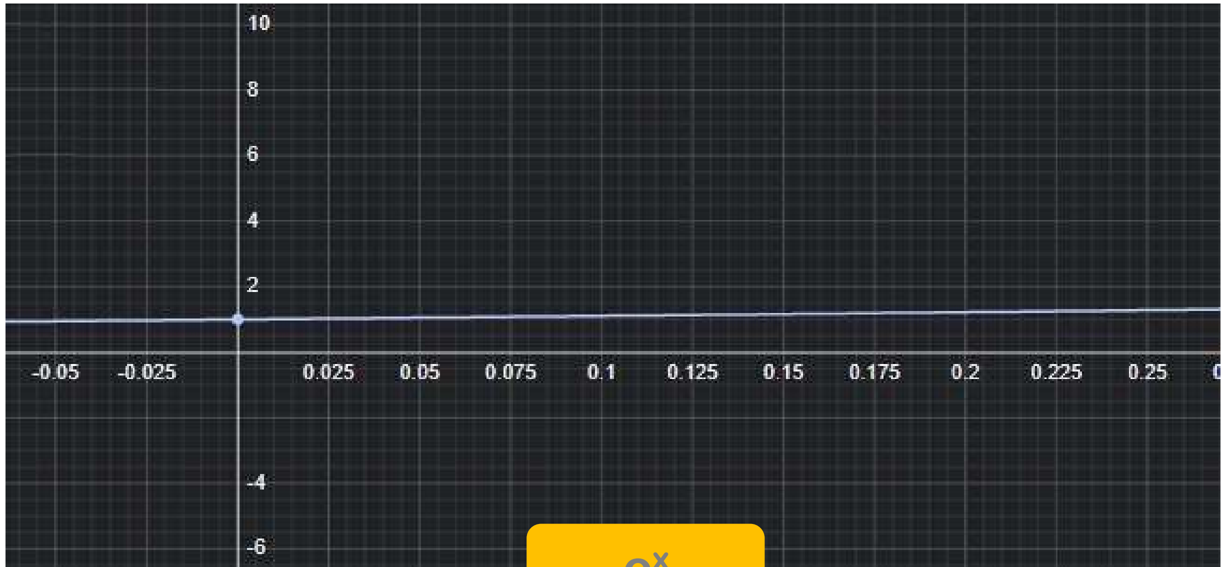




- **Softmax Formula**

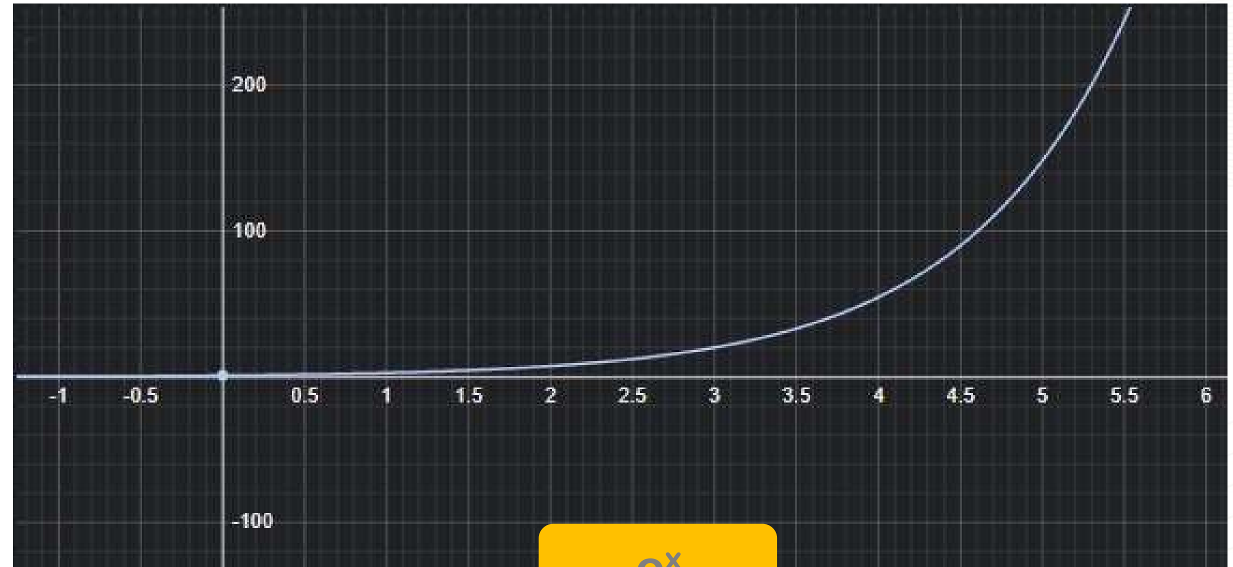
$$s(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$$


Reasons to use softmax function (Task to translate 'gato')
Without softmax
The probability of Gato's translation is
cat: 5
servant: 4
person from Madrid :-1
AI
With softmax
The probability of Gato's translation is
cat: 73.0%
servant: 26.8%
person from Madrid: 0.2%
The probability of Gato's translation is
cat: 5
servant: 4
person from Madrid: -1
softmax
AI
Softmax function can convert to intuitively understandable probability values.



e^x

e=2.718



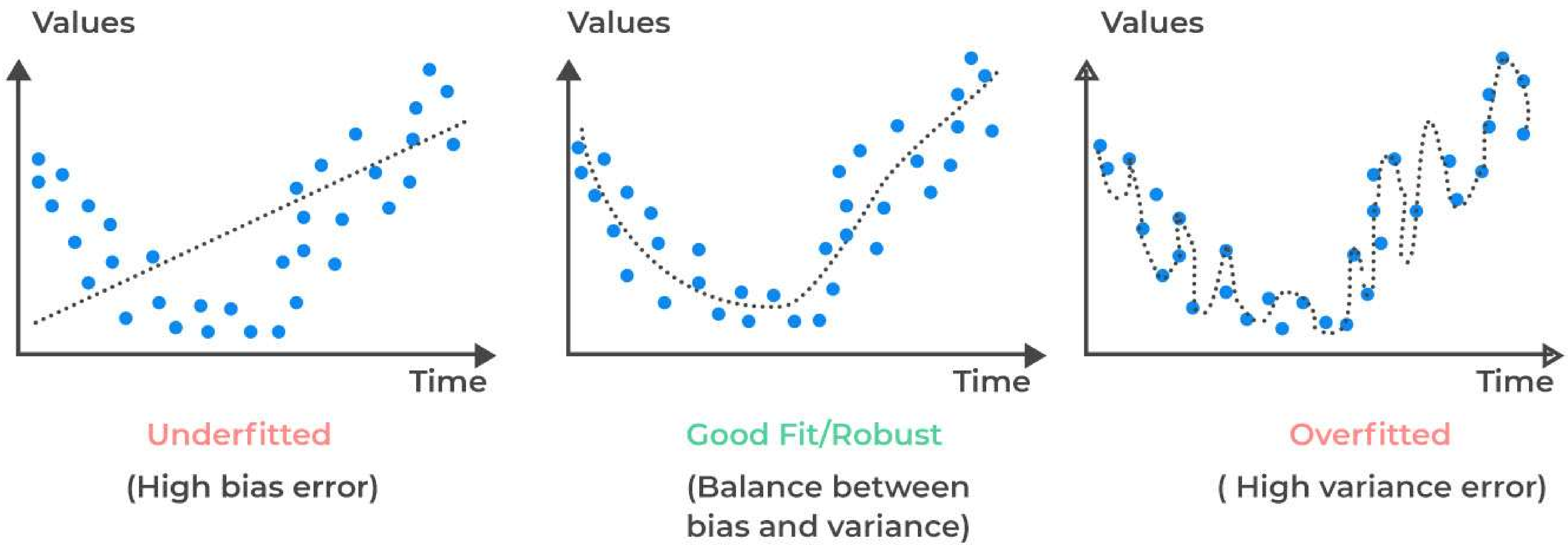
e^x

Ln(x)

Log(x)

Overfit & underfit

overfitting (high variance):
when the model fits perfectly to the training examples => limited generalization.
underfitting (high bias): didn't learn the data well enough



- https://analystprep.com/study-notes/cfa-level-/addressing-methods-method/overfitting-quantitative/٢

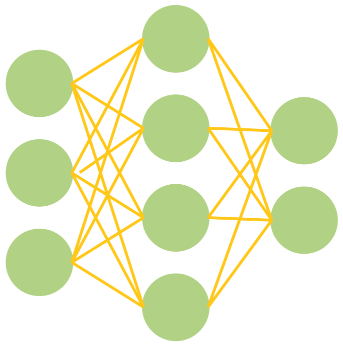
Difference: data science, artificial intelligence, machine learning & deep learning

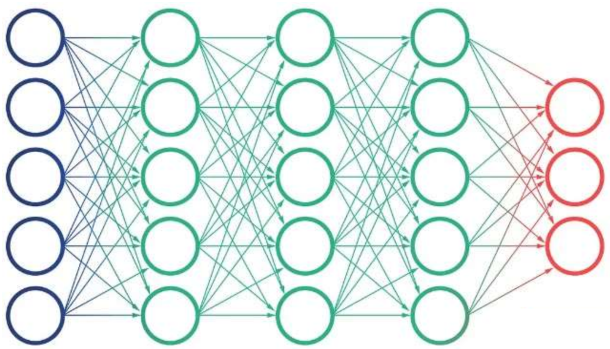
Artificial intelligence refers to computer systems that can behave **intelligently**, **reason**, and **learn** like humans.

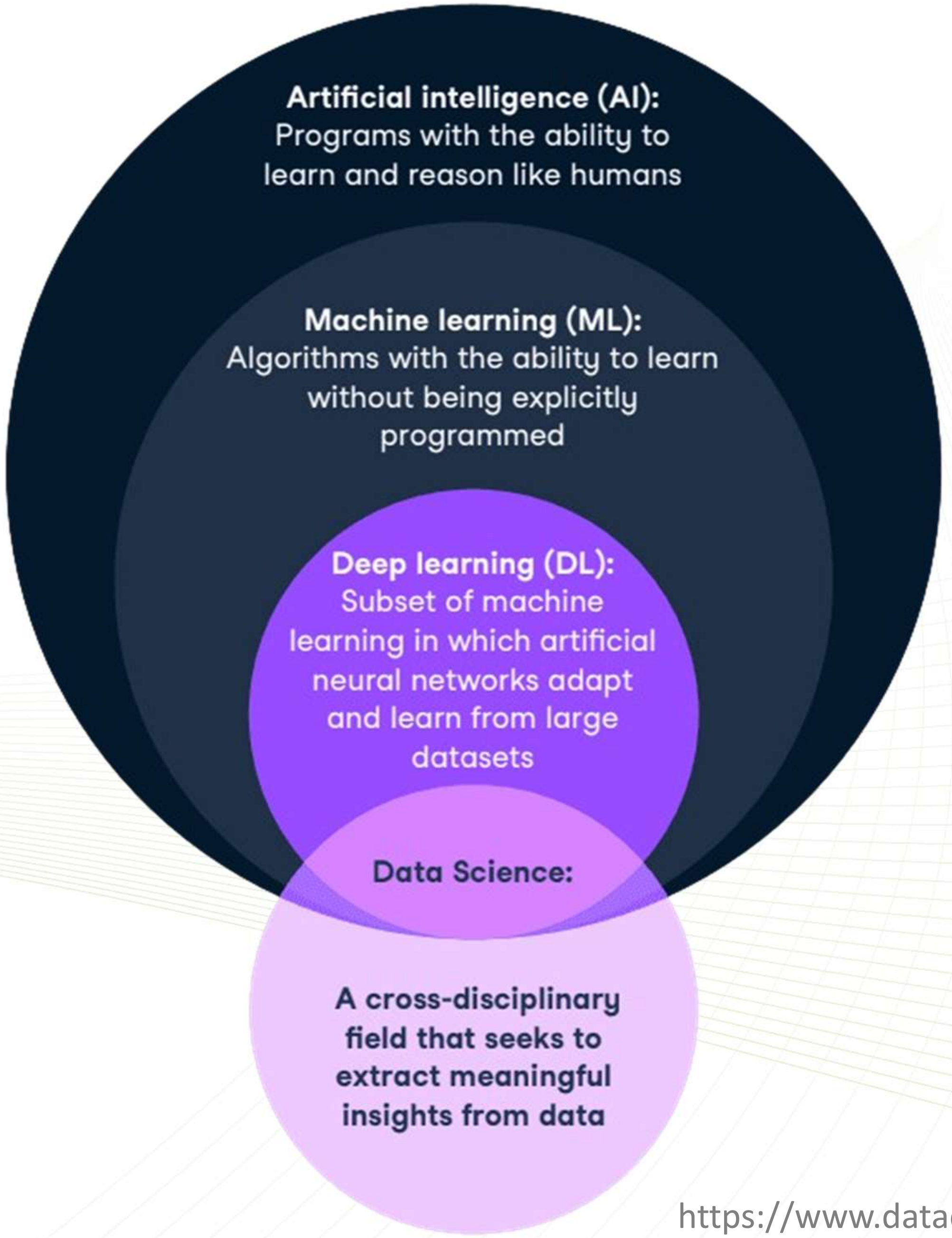
Machine learning is a subset of artificial intelligence focused on developing algorithms with the ability to learn without explicitly being programmed.

Deep learning is a subset of machine learning. It is responsible for many of the awe-inspiring news stories about AI in the news (e.g., self-driving cars, ChatGPT). Deep learning algorithms are inspired by the brain's structure and work exceptionally well with unstructured data such as images, videos, or text.

Data science is a cross-disciplinary field that uses all of the above, amongst other skills like data analysis, statistics, data visualization, and more, to get insight from data.

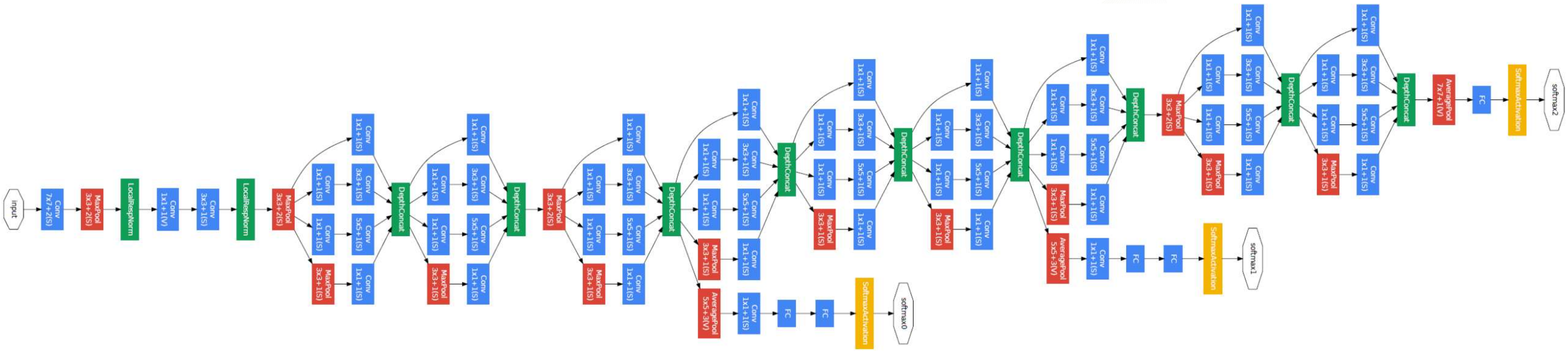






https://www.datacamp.com

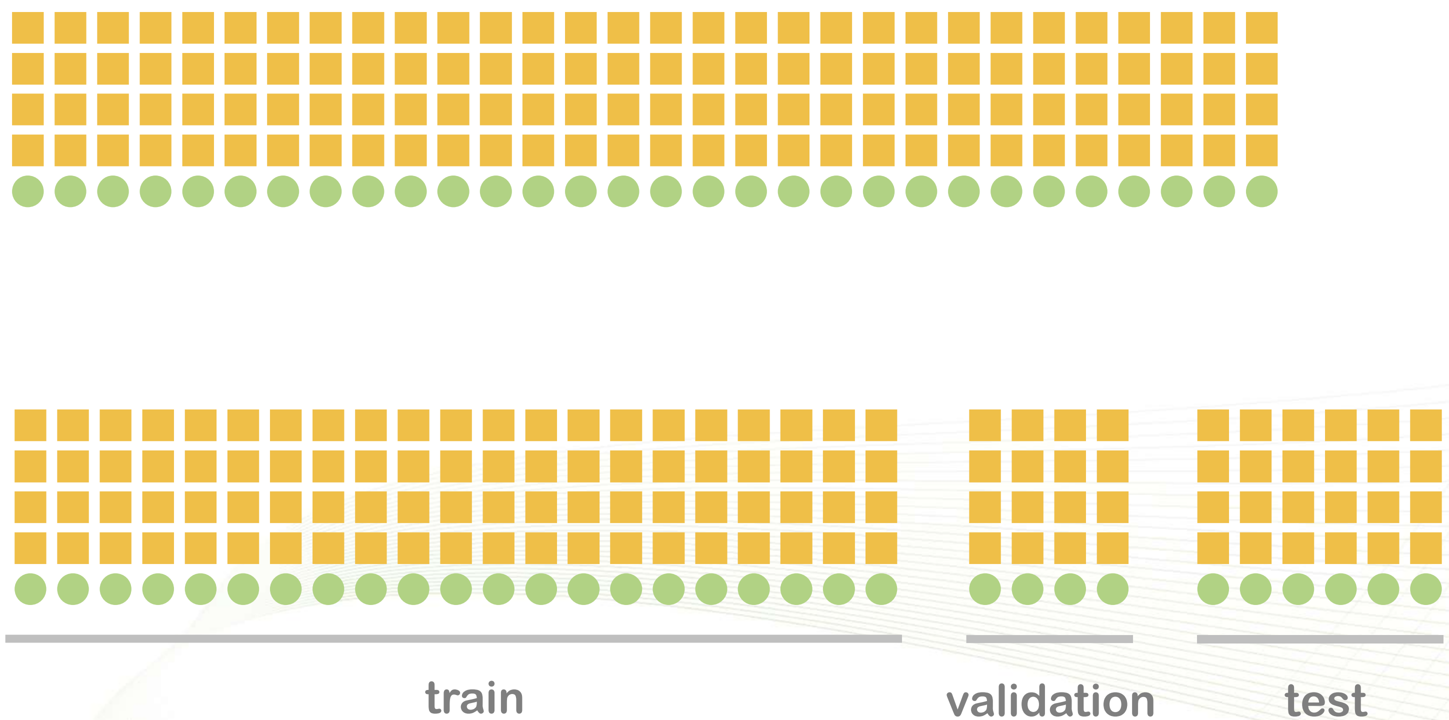
GoogLeNet



Convolution
Pooling
Softmax
Others

Some famous Neural Network Architectures

LeNet5	First convolutional neural network, leading role at the beginning of the deep learning - yann lecun - 1994
Dan Ciresan Net	First implementation of GPU neural -2010. 9 layers - NVIDIA GTX 280
AlexNet	Won the challenging competition of imagenet
Overfeat	Derivative of AlexNet
VGG	Used smaller 3×3 filters in each convolutional layers – based on LeNet by Oxford
GoogLeNet	first architecture to decrease the burden of computation of deep neural networks.
ResNet	Straightforward, More than a hundred and thousand layers trained first time
SqueezeNet	Inception and ResNet’s concepts have been re-hashed in SqueezeNet
ENet	Very light-weight, low computations and parameters. Combining modern architectures’ features.



x_train
y_train

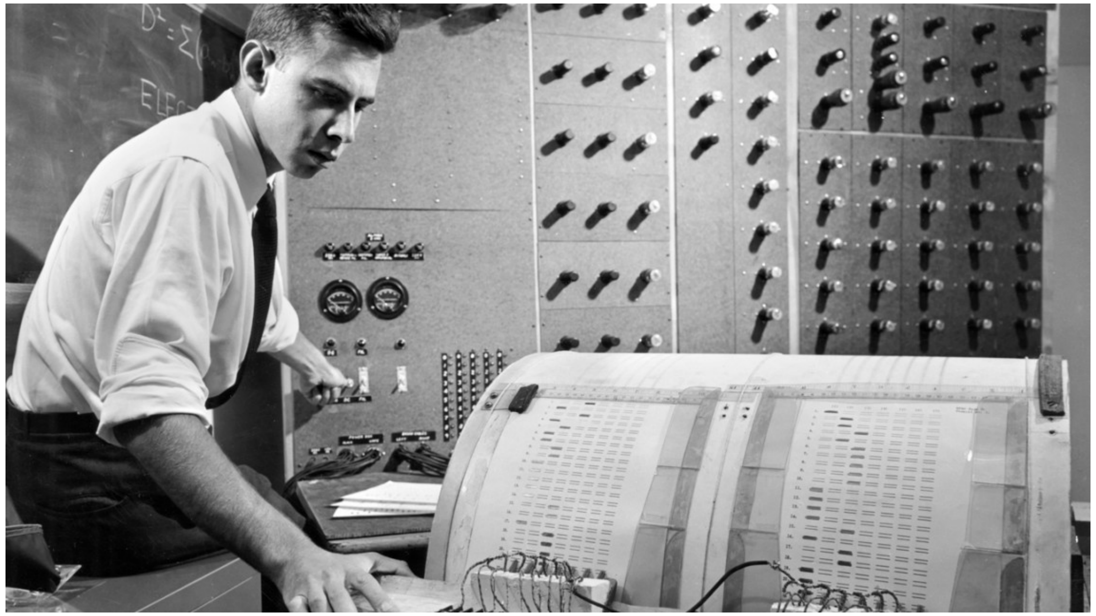
x_test
y_test

Train

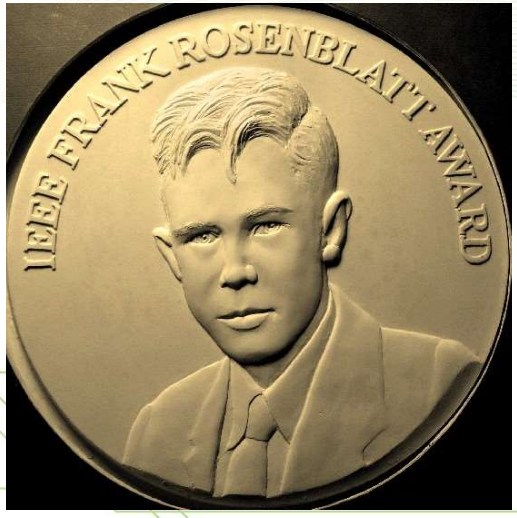
Email	Label
Click here to claim your prize!	Spam
What's new?	Not spam
Hang out later?	Not spam
...	...

Test

Email	Label
Pick up groceries	Not spam
Free free free!	Spam
Android 5.0 questions	Not spam
...	...



Frank Rosenblatt
psychologist



First Model of Perceptron for Supervised learning

Obtaining the weights using the input data by the neural network itself

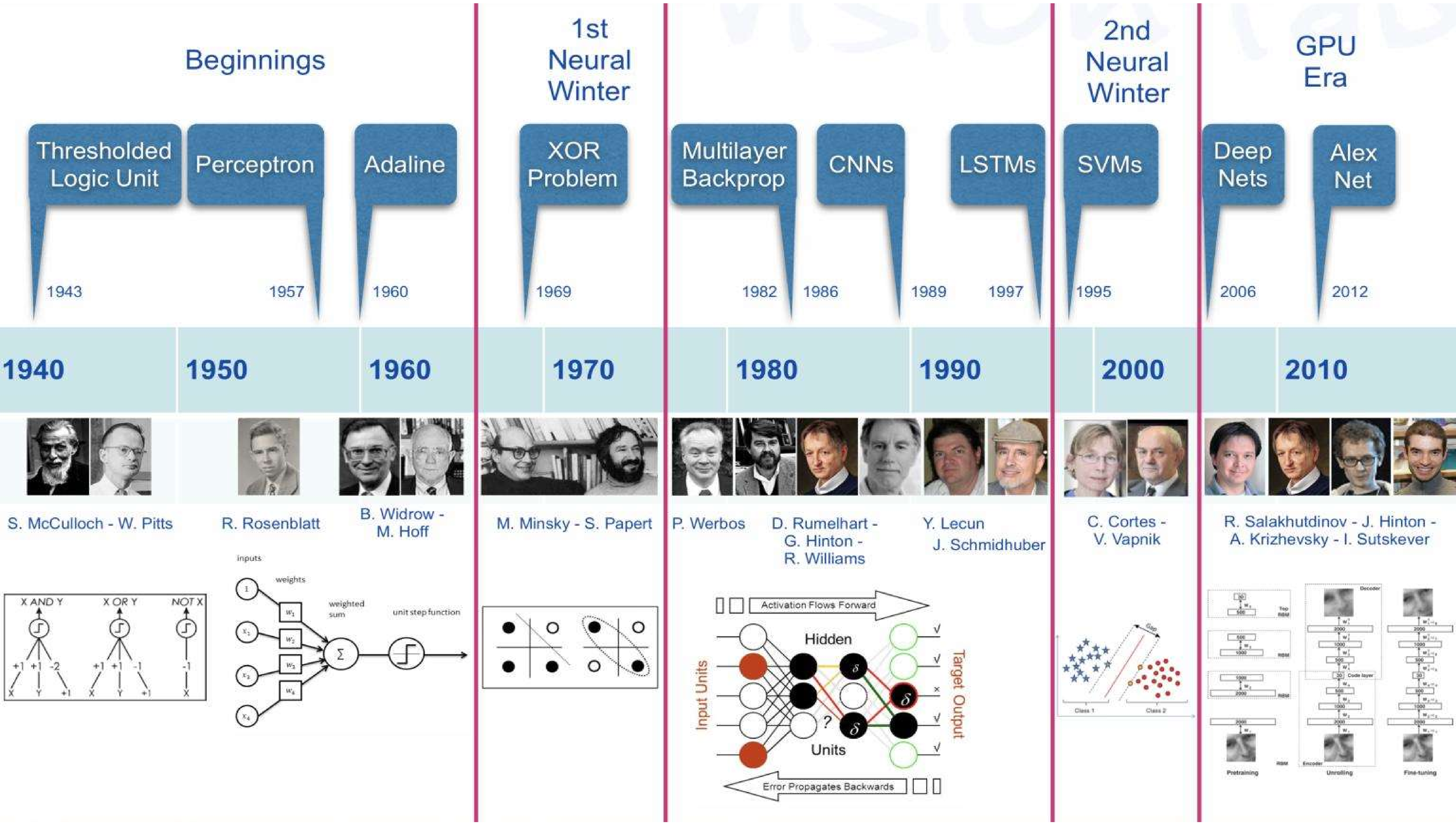
Similar to the todays perceptron networks

IEEE annual award

After 1952

- Early enthusiasm, great expectations (1952–1969)
- Expert systems (1969–1986)
- The return of neural networks (1986–present)
- Probabilistic reasoning and machine learning (1987–present)
- Big data (2001–present)
- Deep learning (2011–present)

Reference: www.simplypsychology.org


Beginnings
1st Neural Winter
2nd Neural Winter
GPU Era
Thresholded Logic Unit
Perceptron
Adaline
XOR Problem
Multilayer Backprop
CNNs
LSTMs
SVMs
Deep Nets
Alex Net
1943
1957
1960
1969
1982
1986
1989
1997
1995
2006
2012
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
R. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
P. Werbos
D. Rumelhart - G. Hinton - R. Williams
Y. Lecun
J. Schmidhuber
C. Cortes - V. Vapnik
R. Salakhutdinov - J. Hinton - A. Krizhevsky - I. Sutskever
X AND Y
X OR Y
NOT X
inputs
weights
weighted sum
unit step function
Activation Flows Forward
Hidden
Units
Input Units
Target Output
Error Propagates Backwards
Pretraining
Unrolling
Fine-tuning

Implementing Machine Learning

Examine pillars of a practicing AI team

1. Business case
2. Domain expertise
3. Data science
4. Algorithms
5. Application integration

Bettering Machine Learning Model Management

State of tools – understanding machine learning stacks

Machine Learning Methods and Algorithms

1. Decision Trees
2. Support Vector Machines
3. Regression
4. Naïve Bayes Classification
5. Hidden Markov Models
6. Random Forest
7. Recurrent Neural Networks
8. Convolutional Neural Networks
9. …

Developing Validation Sets

Developing Training Sets

Accelerating Training

Encoding Domain Expertise in Machine Learning

Automating Data Science

Deep Learning

Types of Algorithms used in Deep Learning

1. Convolutional Neural Networks (CNNs)
2. Long Short Term Memory Networks (LSTMs)
3. Recurrent Neural Networks (RNNs)
4. Generative Adversarial Networks (GANs)
5. Radial Basis Function Networks (RBFNs)
6. Multilayer Perceptrons (MLPs)
7. Self Organizing Maps (SOMs)
8. Deep Belief Networks (DBNs)
9. Restricted Boltzmann Machines(RBMs)
10. Autoencoders

Attention mechanisms

Neural network focus on the most relevant parts of the input or output for a given task.

Different types

Compute a score or a weight for each element in the input or output sequence, and then use these scores to aggregate the elements into a context vector.

The attention mechanism borrows from the signal processing mechanism of the human brain.

Its application in convolutional neural networks is reflected in generating differences in the importance of each piece of input information

Attention mechanisms

Examples of attention mechanisms:

Scaled Dot-Product Attention: This is a type of attention that computes the similarity between a query vector and a set of key vectors, and then uses these similarities to weight the corresponding value vectors. This is used in the Transformer model

Additive Attention: Computes the score for each element by concatenating the query vector and the key vector, and then passing them through a feed-forward network. This is also known as Bahdanau attention.

Channel Attention: Computes the importance of each channel in a feature map, and then uses these weights to rescale the feature map. This is used in Squeeze-and-Excitation Networks.

Ultra realistic alien artic fractal fungal plant tree proportinal geometry dynamic composition, asymmetrical composition,golden ratio, photorealism, Lens with Nikon d850, 28mm, Film Light, Overall Light, Ethereal Light, Depth of Field, f/2.8, Ultra HD, 128k, 3D Shadows, Tone Mapping, Ray Traced Global Illumination, Super Resolution, Gigapixel, Color Correction, Retouching, Enhancement, blue sky, PBR, Blender, V-ray, Procreate, Unreal Engine 5, Cinema 4D, ROMM RGB, Adobe After Effects, 3DCG, VFX, SFX, FXAA , award-winning photography, natural lighting, to p light, studio light, rim light, Volumetric Lights , RTX Renderer, High Qual enviornment, Extreme Realism, RTX On, Realistic, Hyper Realism, Realistic Textures, Amazing Props, Making Amazing Visual Effects, Crazy Details, High Details, Intrinsic Details, Post Production, cool palette, depth of field, IMAX, ALEXA LF, --ar 3:2 --s 999 --w 2311

Understanding Generative Adversarial Networks (GANs)

Ilya Sutskever

OpenAI

Ion Goodfellow

Self-supervised learning

Has taken the world

The most interesting idea proposed in the field of artificial intelligence in the last ten years

Yann LeCun – Meta-NYU

Machine learning sucks!