Artificial Intelligence

NTPU
國立臺北大學
National Taipei University

Deep Learning and Reinforcement Learning

1111AI07
MBA, IM, NTPU (M6132) (Fall 2022)
Wed 2, 3, 4 (9:10-12:00) (B8F40)

https://meet.google.com/miy-fbif-max

Min-Yuh Day, Ph.D,
Associate Professor

Institute of Information Management, National Taipei University

https://web.ntpu.edu.tw/~myday

2022-11-09

Syllabus

Week	Date	Subject/Topics
1	2022/09/14	Introduction to Artificial Intelligence
2	2022/09/21	Artificial Intelligence and Intelligent Agents
3	2022/09/28	Problem Solving
4	2022/10/05	Knowledge, Reasoning and Knowledge Representation; Uncertain Knowledge and Reasoning
5	2022/10/12	Case Study on Artificial Intelligence I
6	2022/10/19	Machine Learning: Supervised and Unsupervised Learning

Syllabus

Week	Date	Subject/Topics
7	2022/10/26	The Theory of Learning and Ensemble Learning
8	2022/11/02	Midterm Project Report
9	2022/11/09	Deep Learning and Reinforcement Learning
10	2022/11/16	Deep Learning for Natural Language Processing
11	2022/11/23	Invited Talk: AI for Information Retrieval
12	2022/11/30	Case Study on Artificial Intelligence II

Syllabus

Week	Date	Subject/Topics
13	2022/12/07	Computer Vision and Robotics
14	2022/12/14	Philosophy and Ethics of AI and the Future of AI
15	2022/12/21	Final Project Report I
16	2022/12/28	Final Project Report II
17	2023/01/04	Self-learning
18	2023/01/11	Self-learning

Deep Learning and Reinforcement Learning

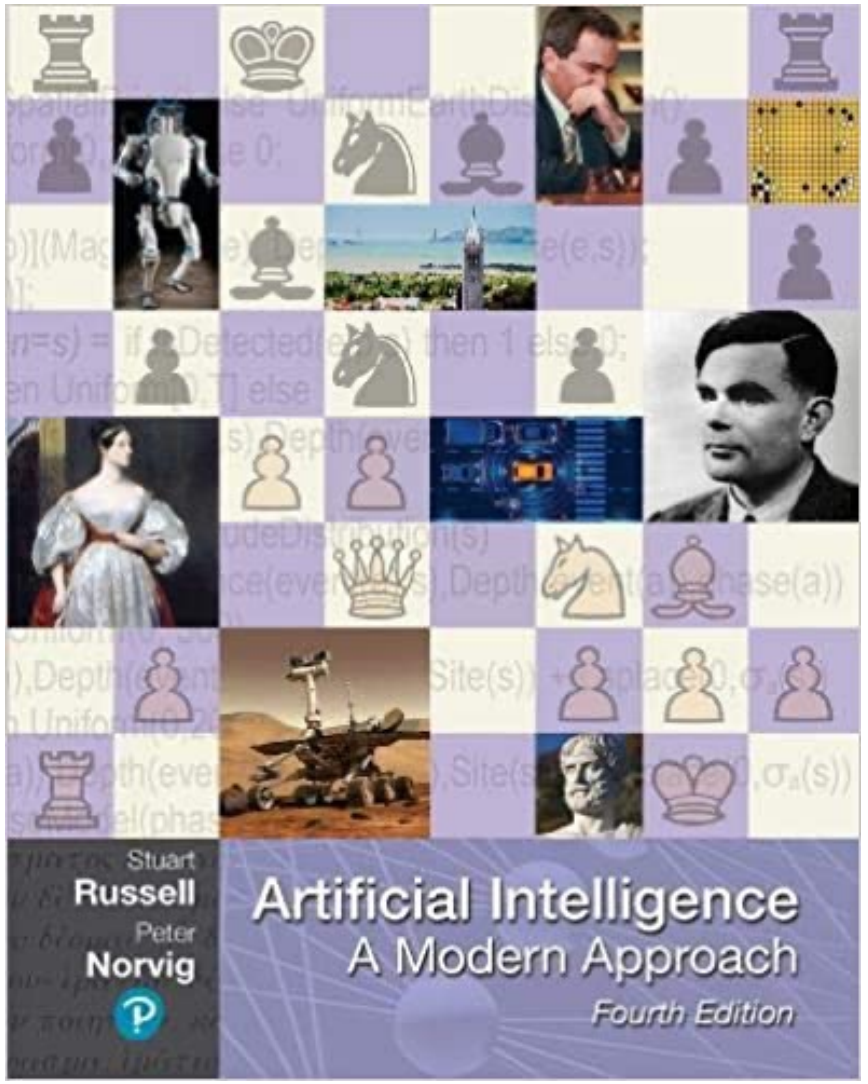
Outline

- Deep Learning (DL)
  - Neural Networks (NN)
  - Convolutional Neural Networks (CNN)
  - Recurrent Neural Networks (RNN)
- Reinforcement Learning (RL)
  - Markov Decision Processes (MDP)
  - Deep Reinforcement Learning (DRL) Algorithms
    - SARSA
    - Q-Learning
    - DQN, A3C, Rainbow

Stuart Russell and Peter Norvig (2020),

Artificial Intelligence: A Modern Approach,

4th Edition, Pearson



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

https://www.amazon.com/Artificial-Intelligence-A-Modern-Approach/dp/0134610997/

Artificial Intelligence: A Modern Approach

1. Artificial Intelligence
2. Problem Solving
3. Knowledge and Reasoning
4. Uncertain Knowledge and Reasoning
5. Machine Learning
6. Communicating, Perceiving, and Acting
7. Philosophy and Ethics of AI

Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Artificial Intelligence:
Machine Learning

Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Artificial Intelligence:
5. Machine Learning

- Learning from Examples
- Learning Probabilistic Models
- Deep Learning
- Reinforcement Learning

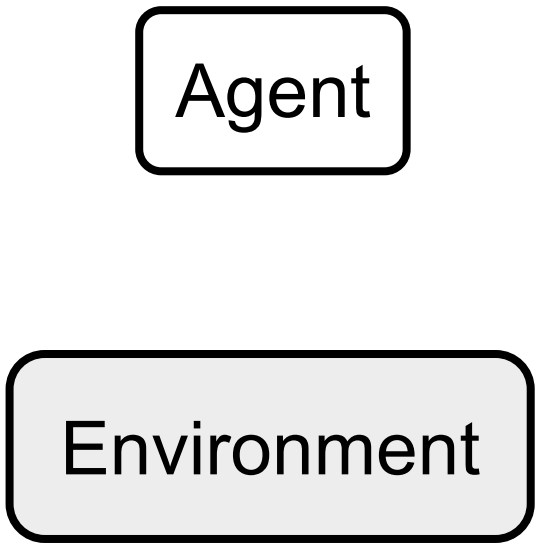
Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Artificial Intelligence:
Reinforcement Learning

- Learning from Rewards
- Passive Reinforcement Learning
- Active Reinforcement Learning
- Generalization in Reinforcement Learning
- Policy Search
- Apprenticeship and Inverse Reinforcement Learning
- Applications of Reinforcement Learning

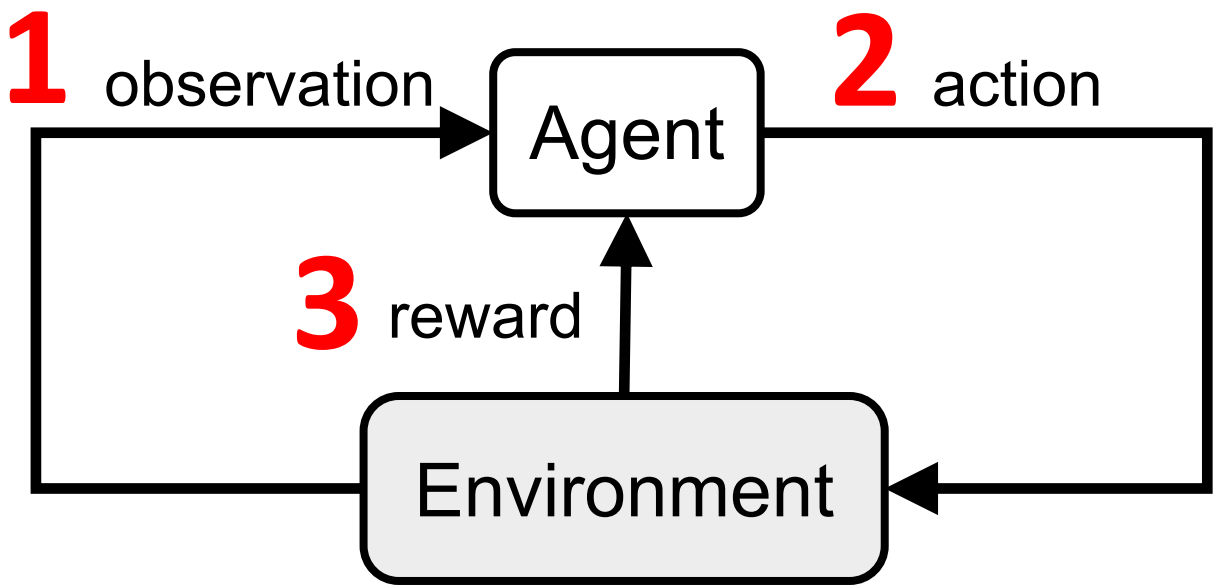
Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Reinforcement Learning (DL)



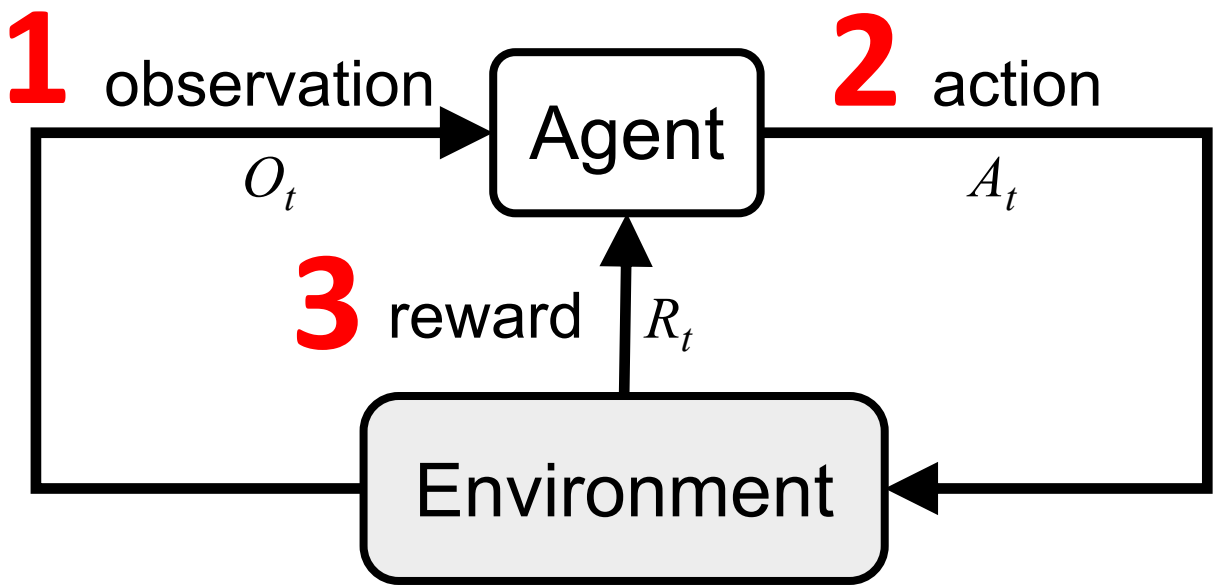
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning (DL)



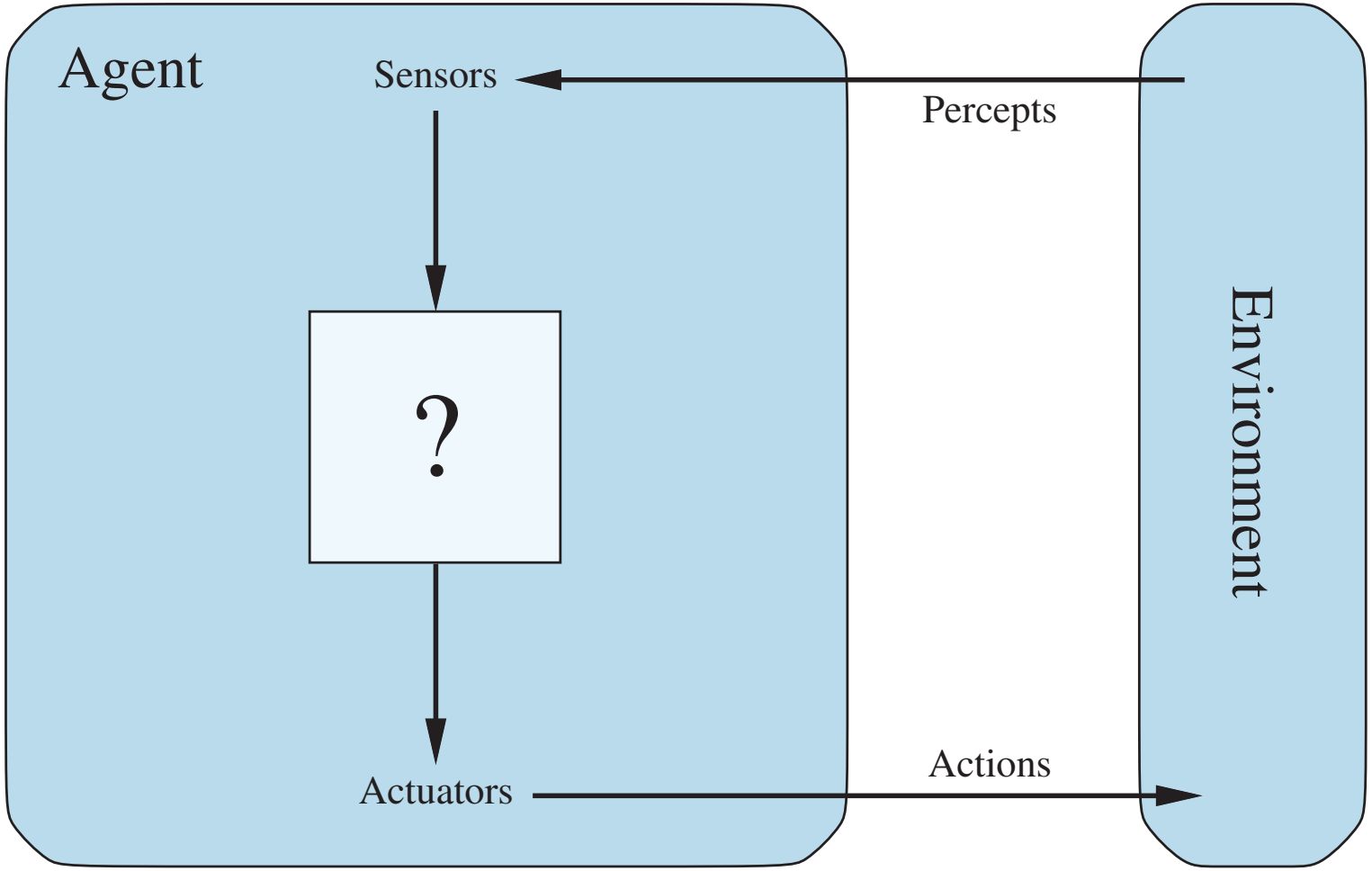
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning (DL)



Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Agents interact with environments through sensors and actuators



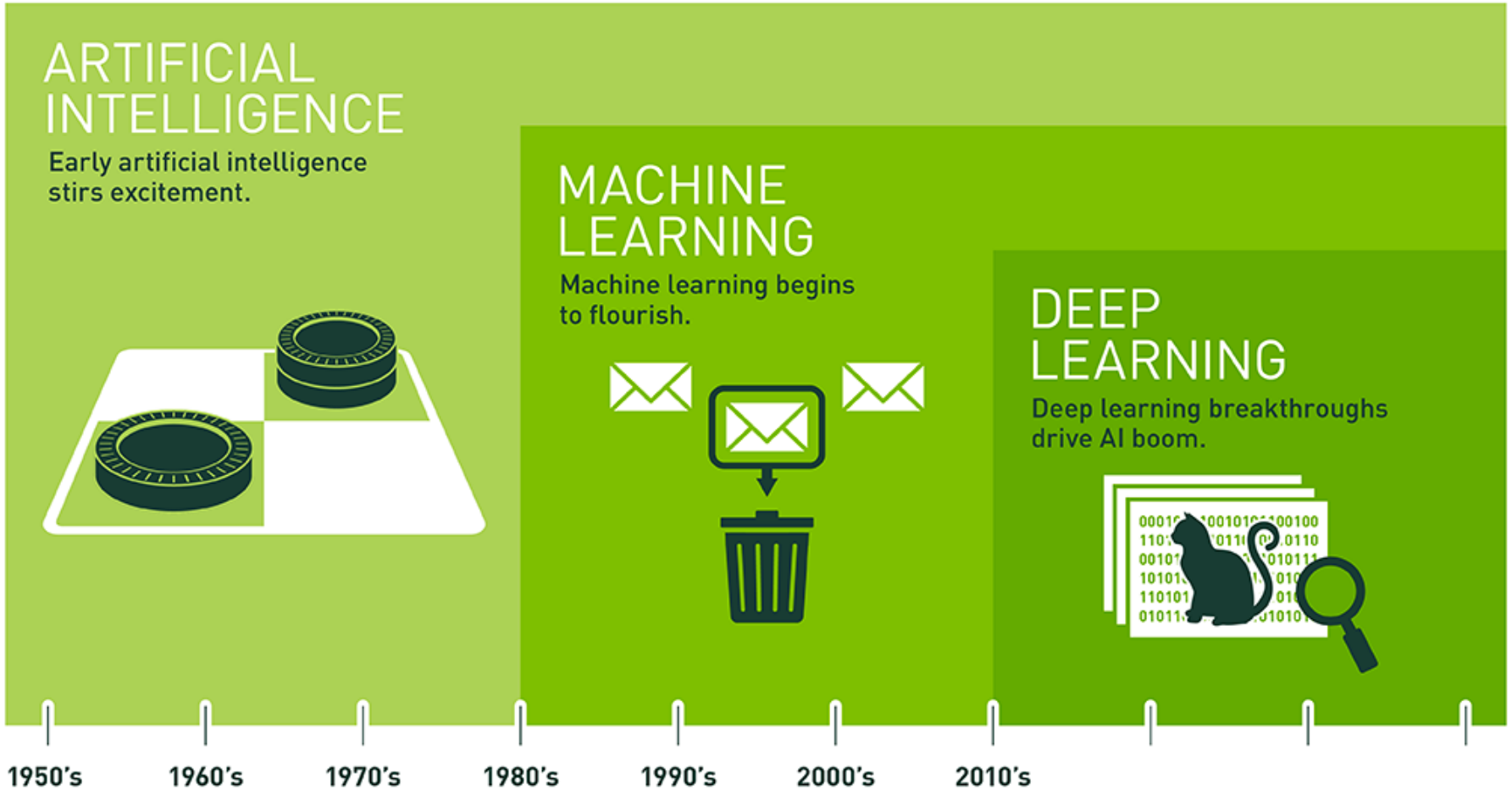
Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

AI Acting Humanly:
The Turing Test Approach
(Alan Turing, 1950)

- Knowledge Representation
- Automated Reasoning
- Machine Learning (ML)
  - Deep Learning (DL)
- Computer Vision (Image, Video)
- Natural Language Processing (NLP)
- Robotics

Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

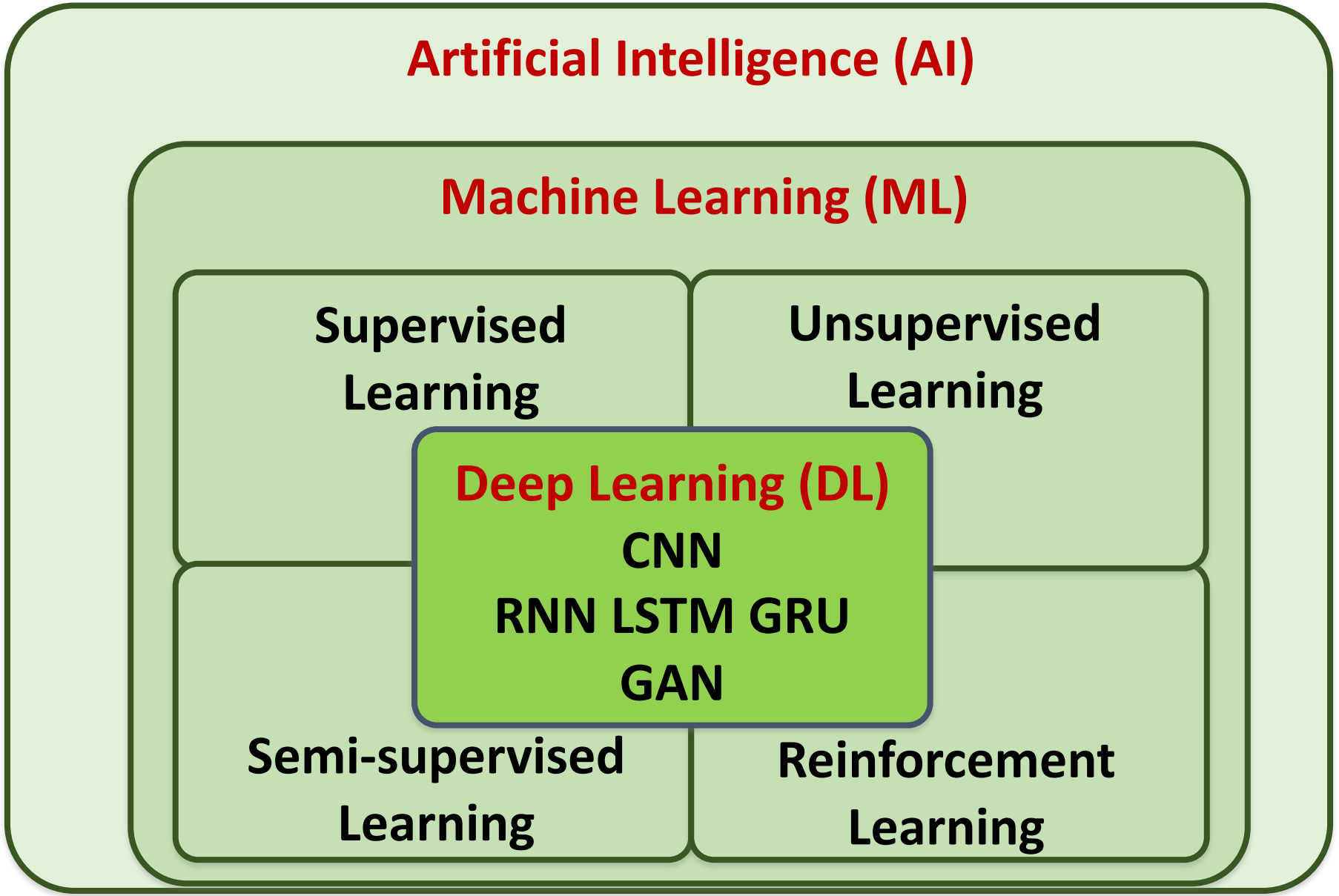
Artificial Intelligence
Machine Learning & Deep Learning



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

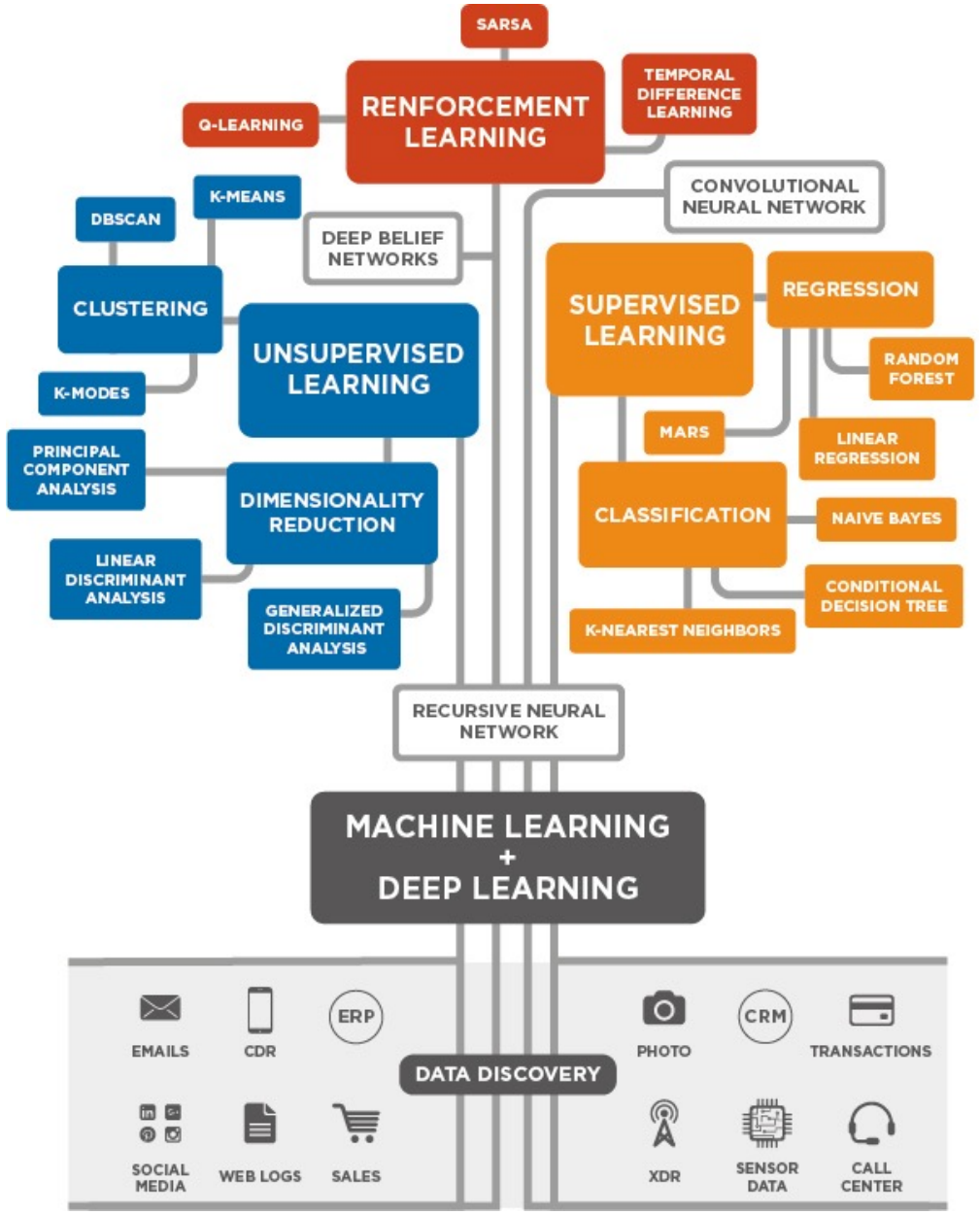
Source: https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

AI, ML, DL



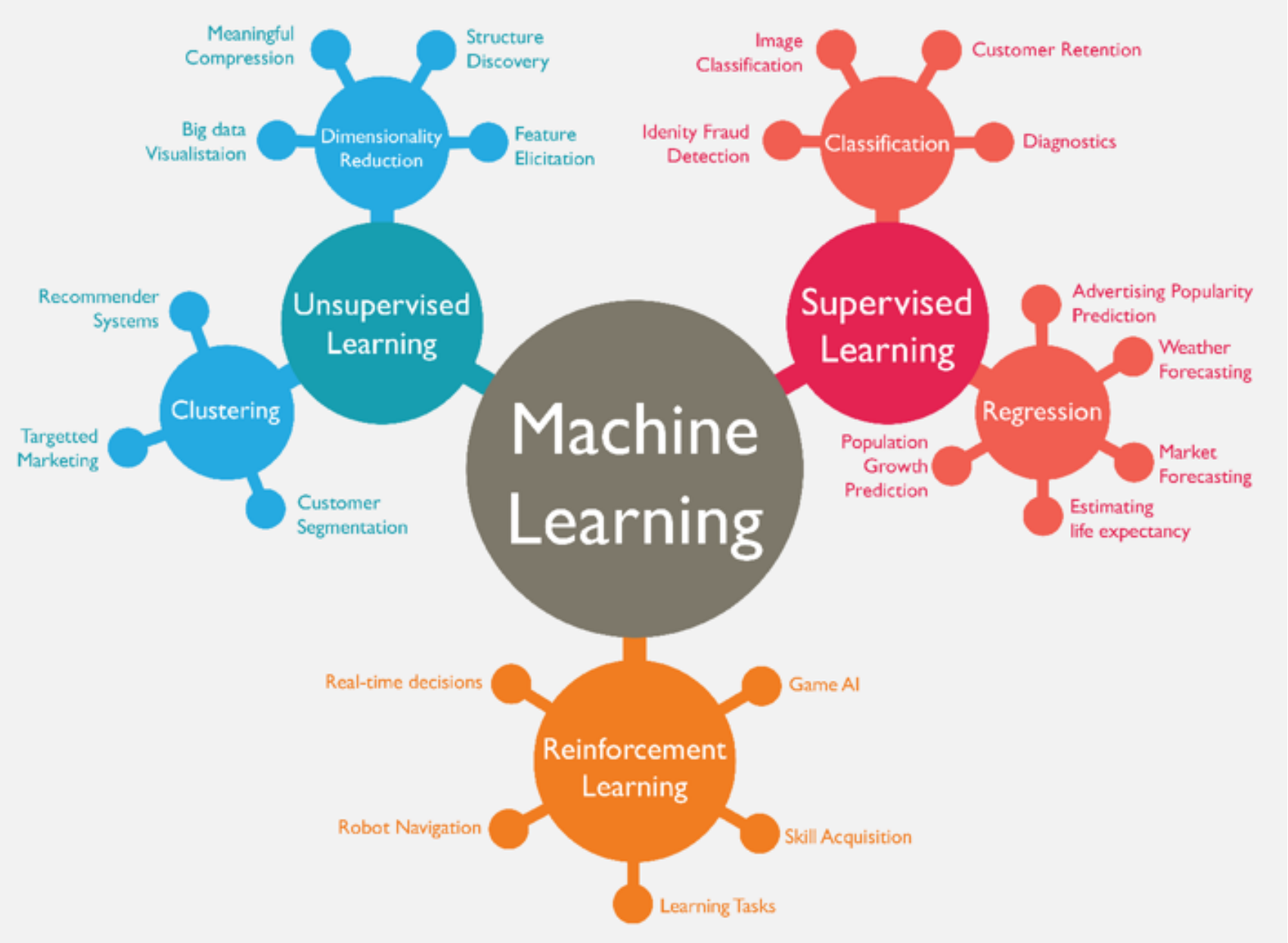
Source: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/deep_learning.html

3 Machine Learning Algorithms



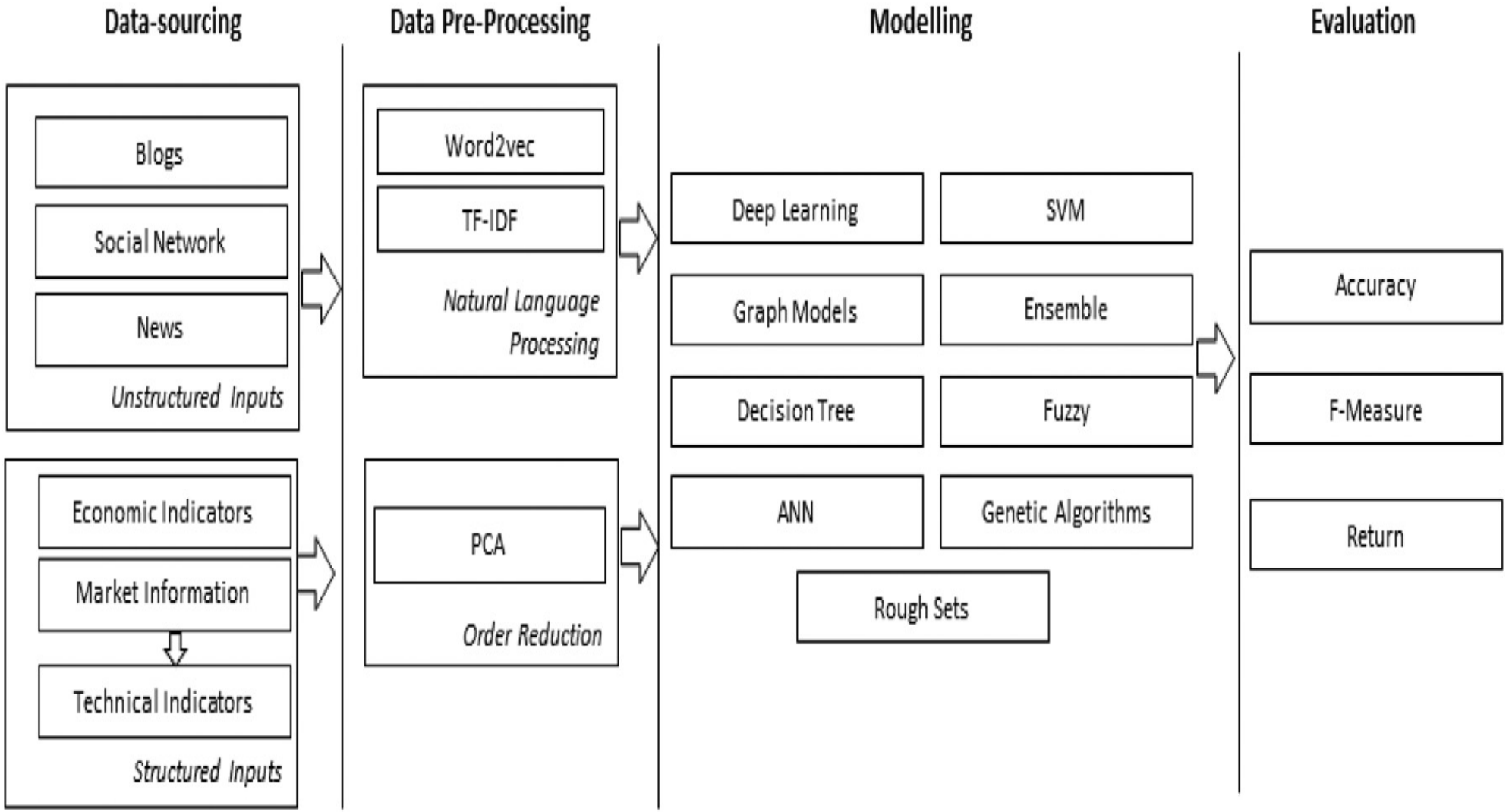
Source: Enrico Galimberti, http://blogs.teradata.com/data-points/tree-machine-learning-algorithms/

Machine Learning (ML)



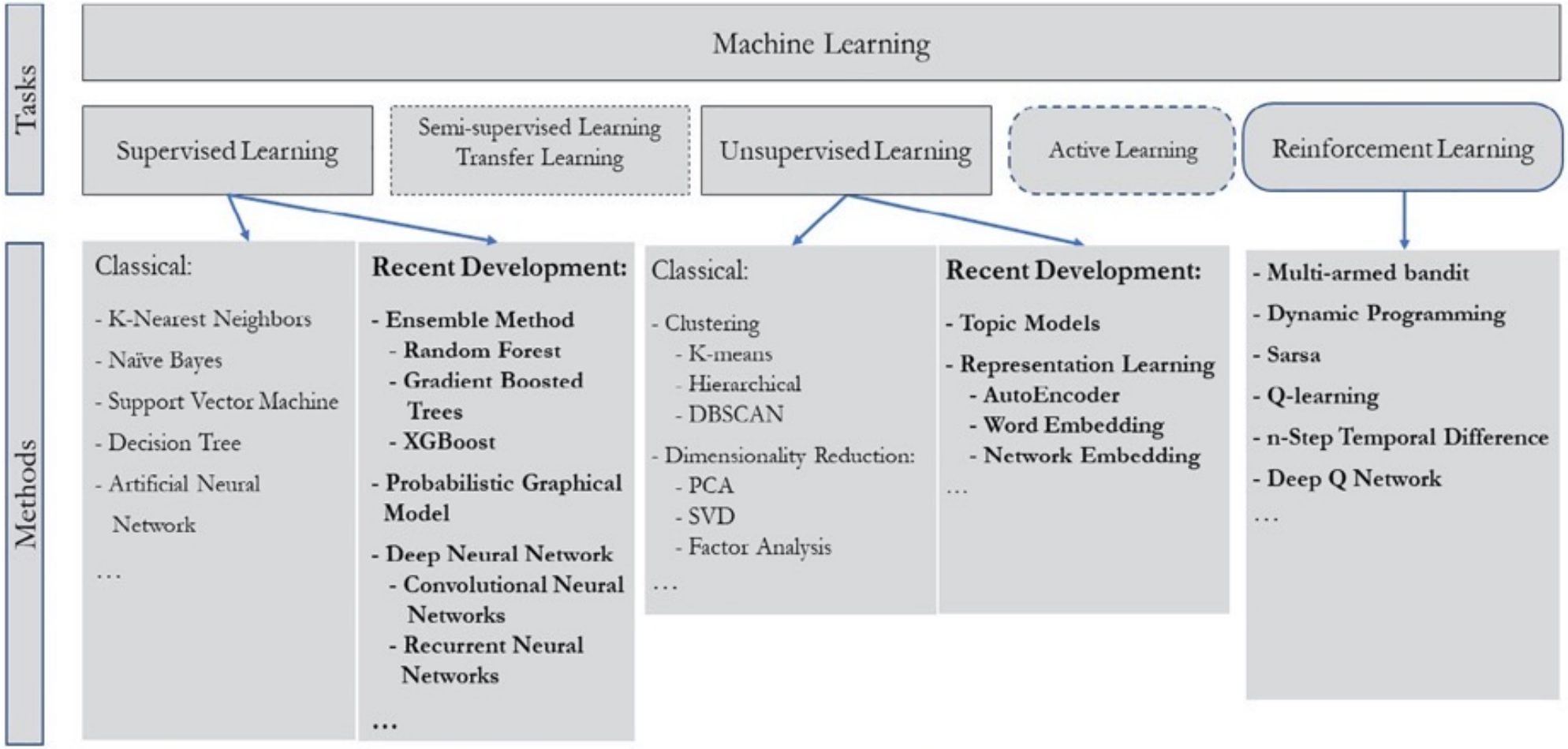
Source: https://www.mactores.com/services/aws-big-data-machine-learning-cognitive-services/

Stock Market Movement Forecast:
ML Phases of the stock market modeling



Source: O. Bustos and A. Pomares-Quimbaya (2020), "Stock Market Movement Forecast: A Systematic Review." Expert Systems with Applications (2020): 113464.

Machine Learning Tasks and Methods



Note: Several entries in the diagram, e.g. word embedding or multi-armed bandit, refer to specific problem formulations for which a collection of methods exist.

: Tasks that take input data as given

: Tasks that involve interactive data acquisition

Dashed border: methods not elaborated in paper text

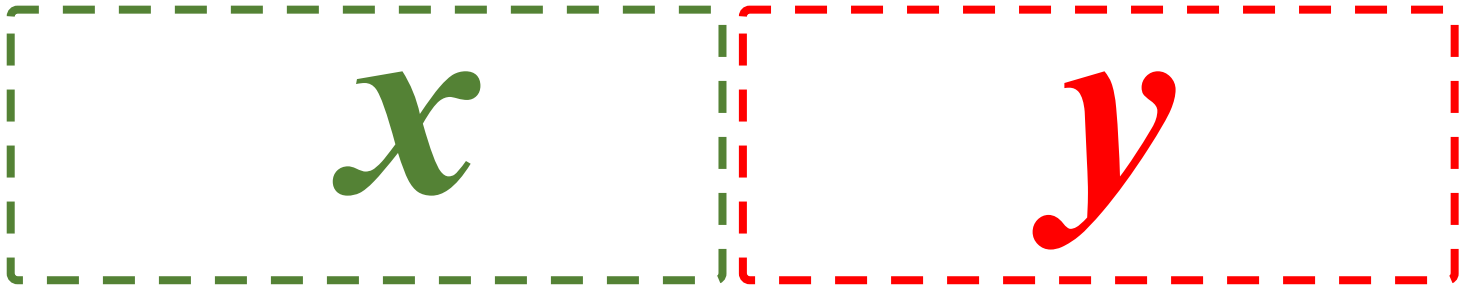
Bold type: highlights recent developments

Source: Liye Ma and Baohong Sun (2020), "Machine learning and AI in marketing – Connecting computing power to human insights." International Journal of Research in Marketing, 37, no. 3, 481-504.

Machine Learning

Machine Learning
Supervised Learning (Classification)
Learning from Examples

$$y = f(x)$$



Machine Learning
Supervised Learning (Classification)
Learning from Examples

$y = f(x)$

Example

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
6.3,3.3,6.0,2.5,Iris-virginica
5.8,2.7,5.1,1.9,Iris-virginica
7.1,3.0,5.9,2.1,Iris-virginica
```

x

y

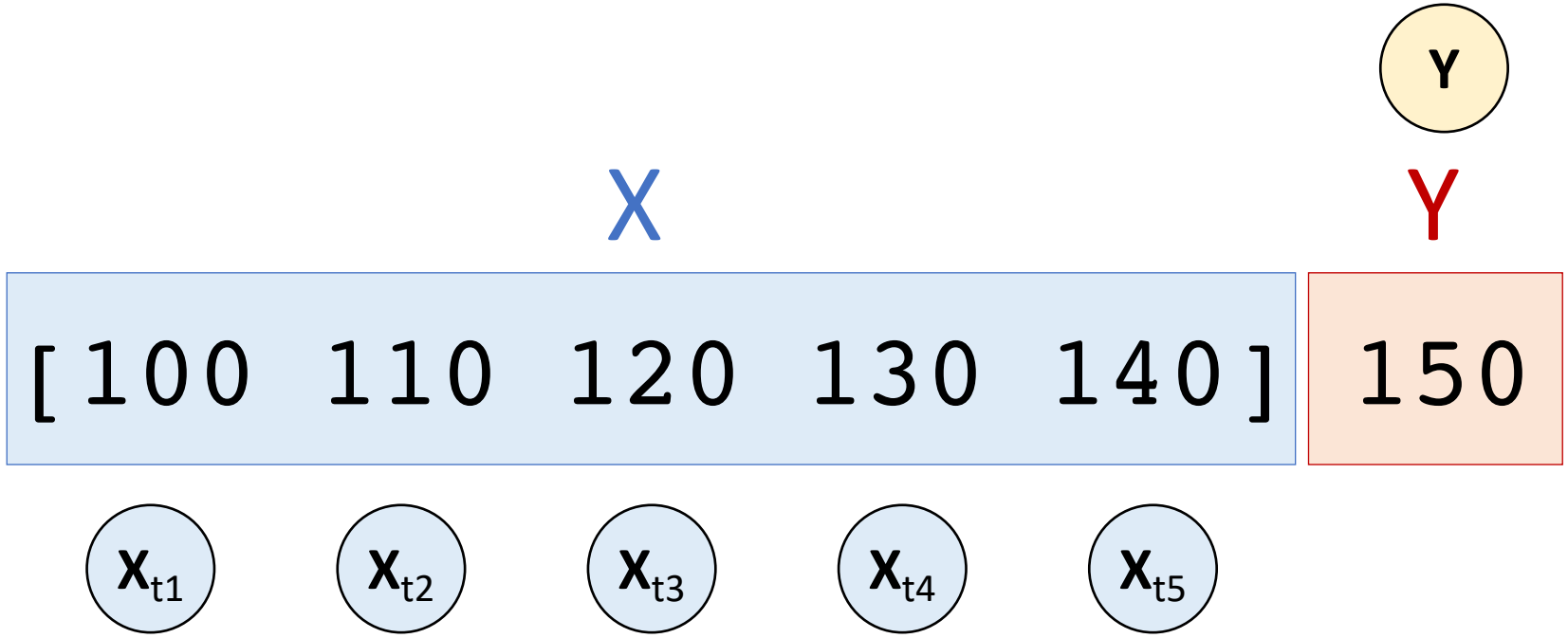
Time Series Data

[10, 20, 30, 40, 50, 60, 70, 80, 90]

X	Y
[10 20 30]	40
[20 30 40]	50
[30 40 50]	60
[40 50 60]	70
[50 60 70]	80
[60 70 80]	90

Time Series Data

[100, 110, 120, 130, 140, 150]



Linear function

$$y = f(x)$$

$$y = w_1 x + w_0$$

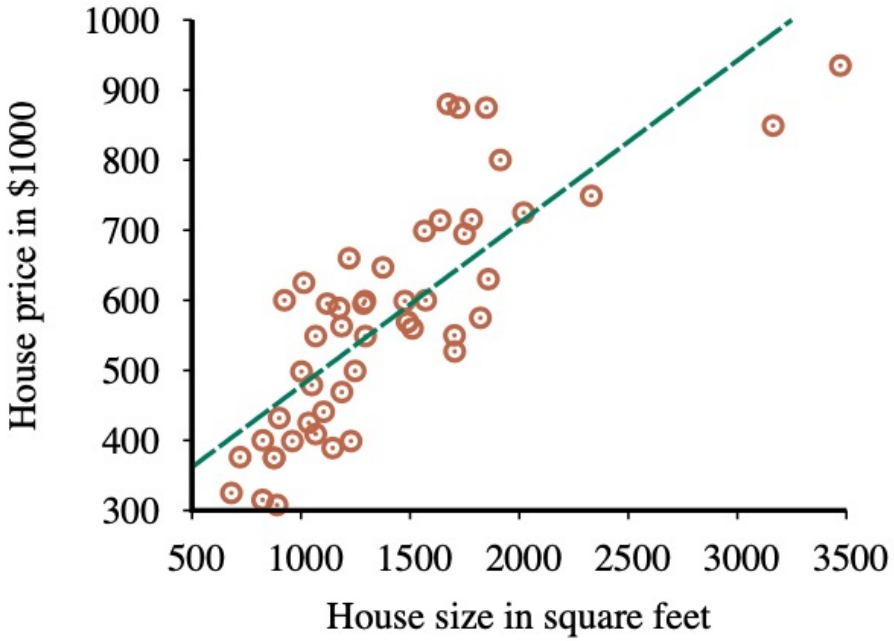
$$h_w(x) = w_1 x + w_0$$

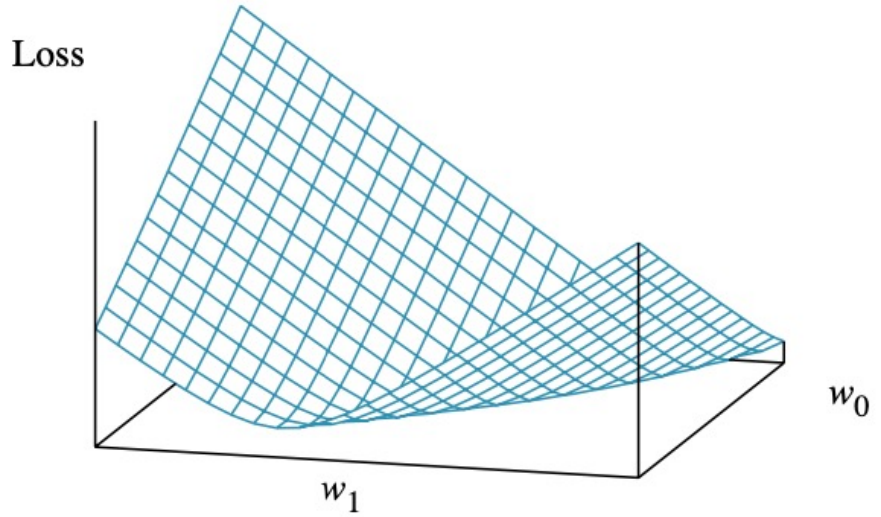
Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

Linear Regression Weight Space

$$h_w(x) = w_1 x + w_0$$

$$w^* = \text{argmin}_w \, Loss(h_w)$$





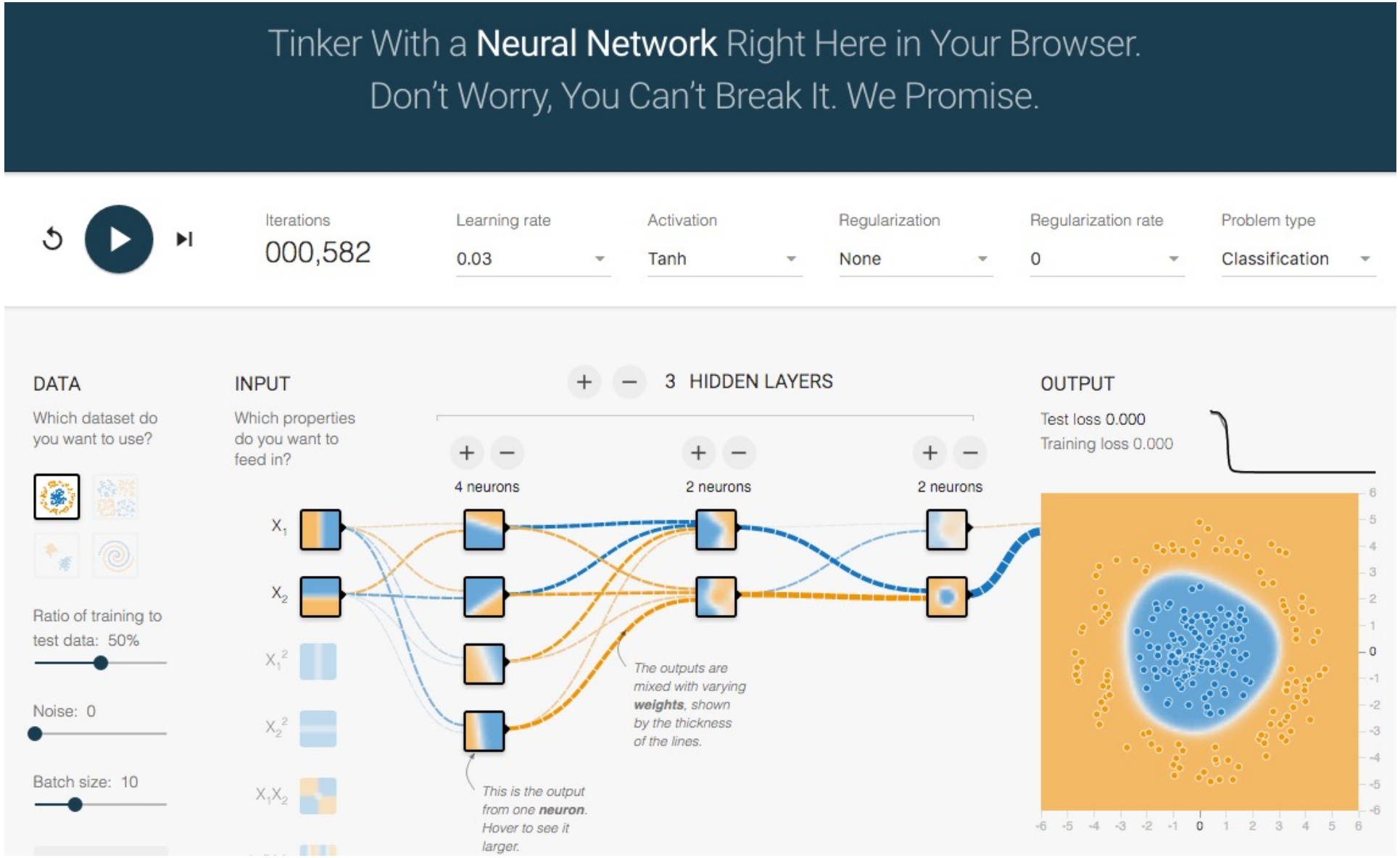
$y = 0.232\,x + 246$

Loss function for Weights (w_1, w_0)

Deep Learning

Deep Learning
and
Neural Networks

TensorFlow Playground



http://playground.tensorflow.org/

Tensor

- 3
  - # a rank 0 tensor; this is a scalar with shape []
- [1. ,2., 3.]
  - # a rank 1 tensor; this is a vector with shape [3]
- [[1., 2., 3.], [4., 5., 6.]]
  - # a rank 2 tensor; a matrix with shape [2, 3]
- [[[1., 2., 3.]], [[7., 8., 9.]]]
  - # a rank 3 tensor with shape [2, 1, 3]

https://www.tensorflow.org/

Scalar 80

Vector [50 60 70]

Matrix

$$\begin{bmatrix} 50 & 60 & 70 \\ 55 & 65 & 75 \end{bmatrix}$$

Tensor

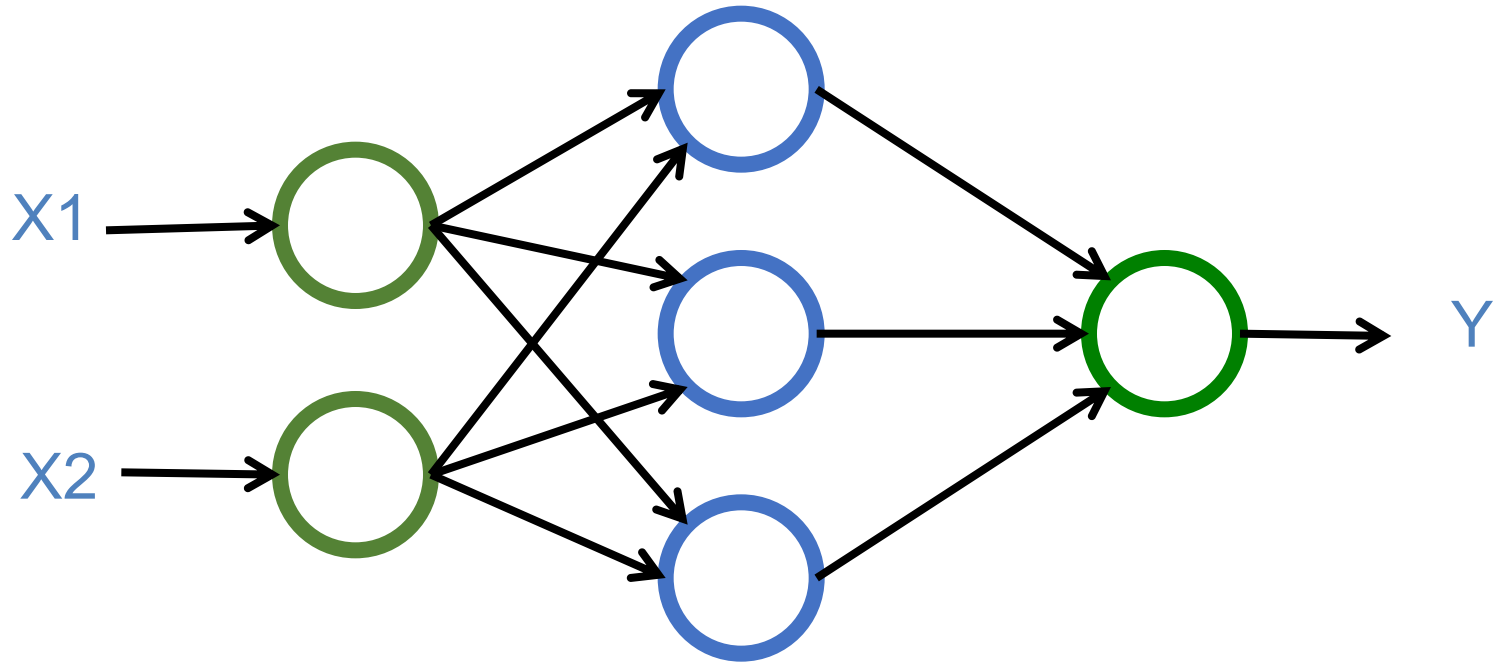
$$\begin{bmatrix} [50 & 60 & 70] & [70 & 80 & 90] \\ [55 & 65 & 75] & [75 & 85 & 95] \end{bmatrix}$$

Deep Learning
and
Neural Networks

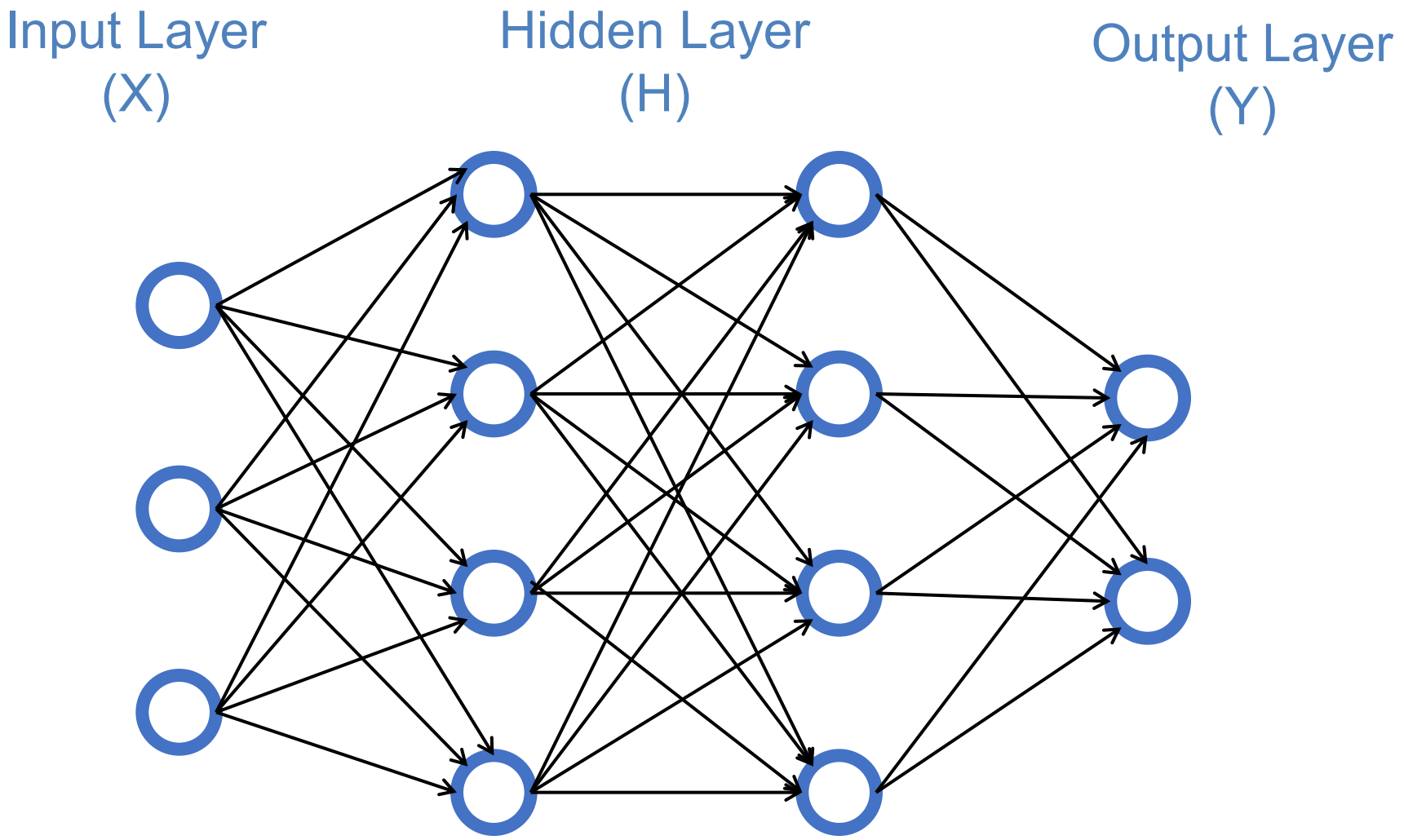
Deep Learning Foundations: Neural Networks

Deep Learning and Neural Networks

Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



Deep Learning and Neural Networks



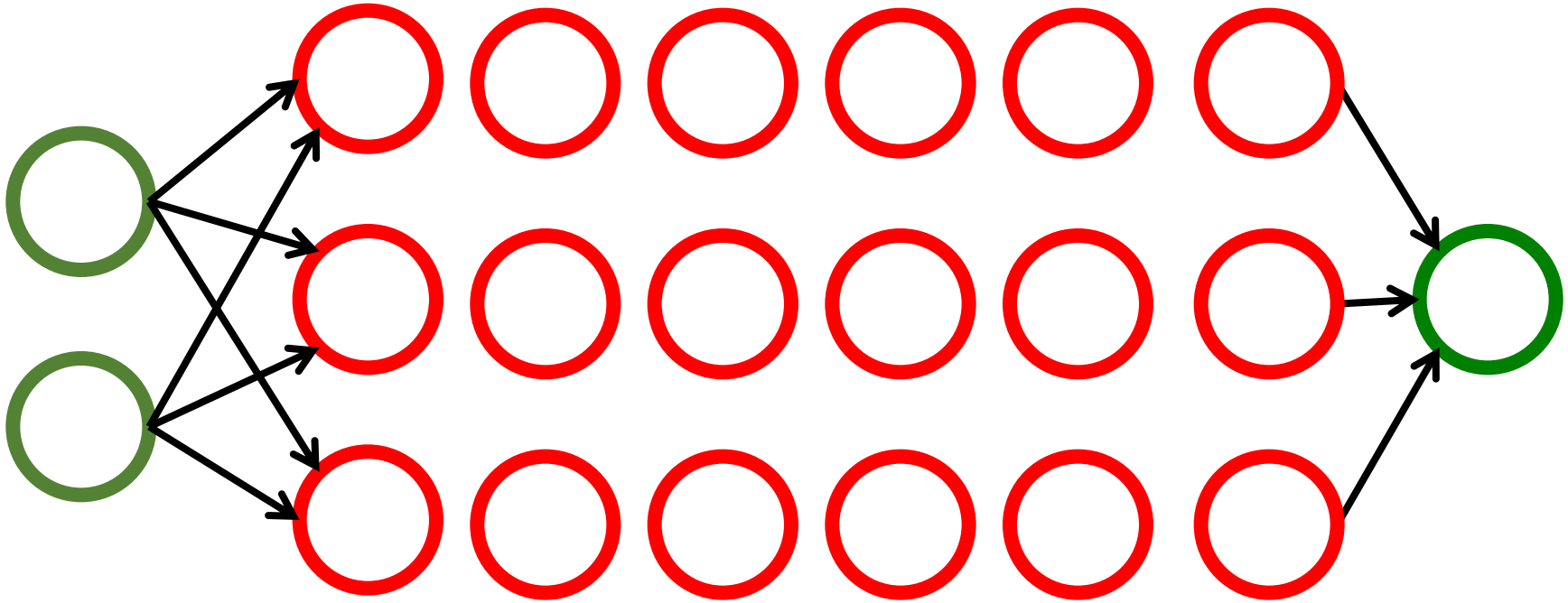
Deep Learning and Neural Networks

Input Layer (X)

Hidden Layers (H)

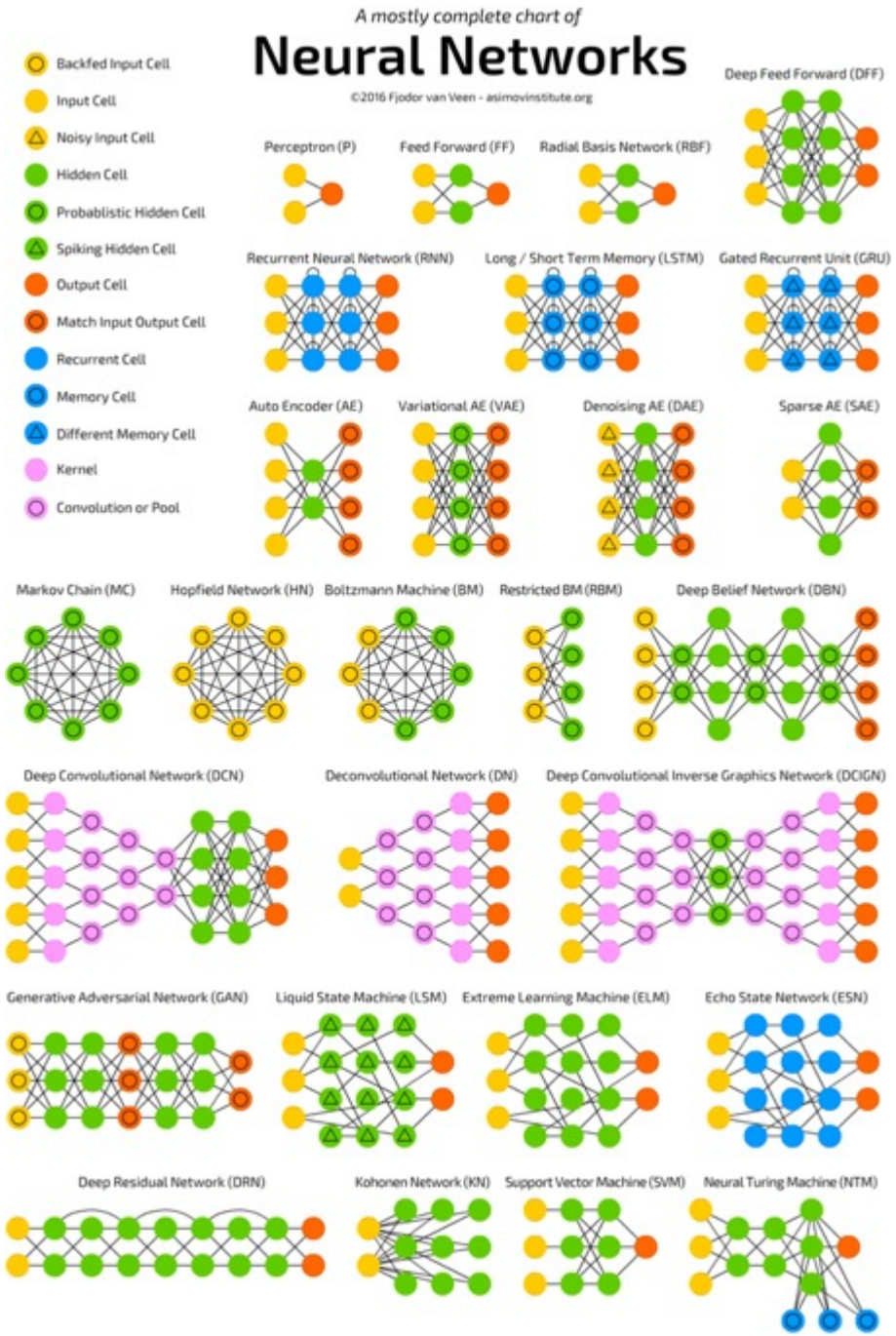
Output Layer (Y)

Deep Neural Networks
Deep Learning

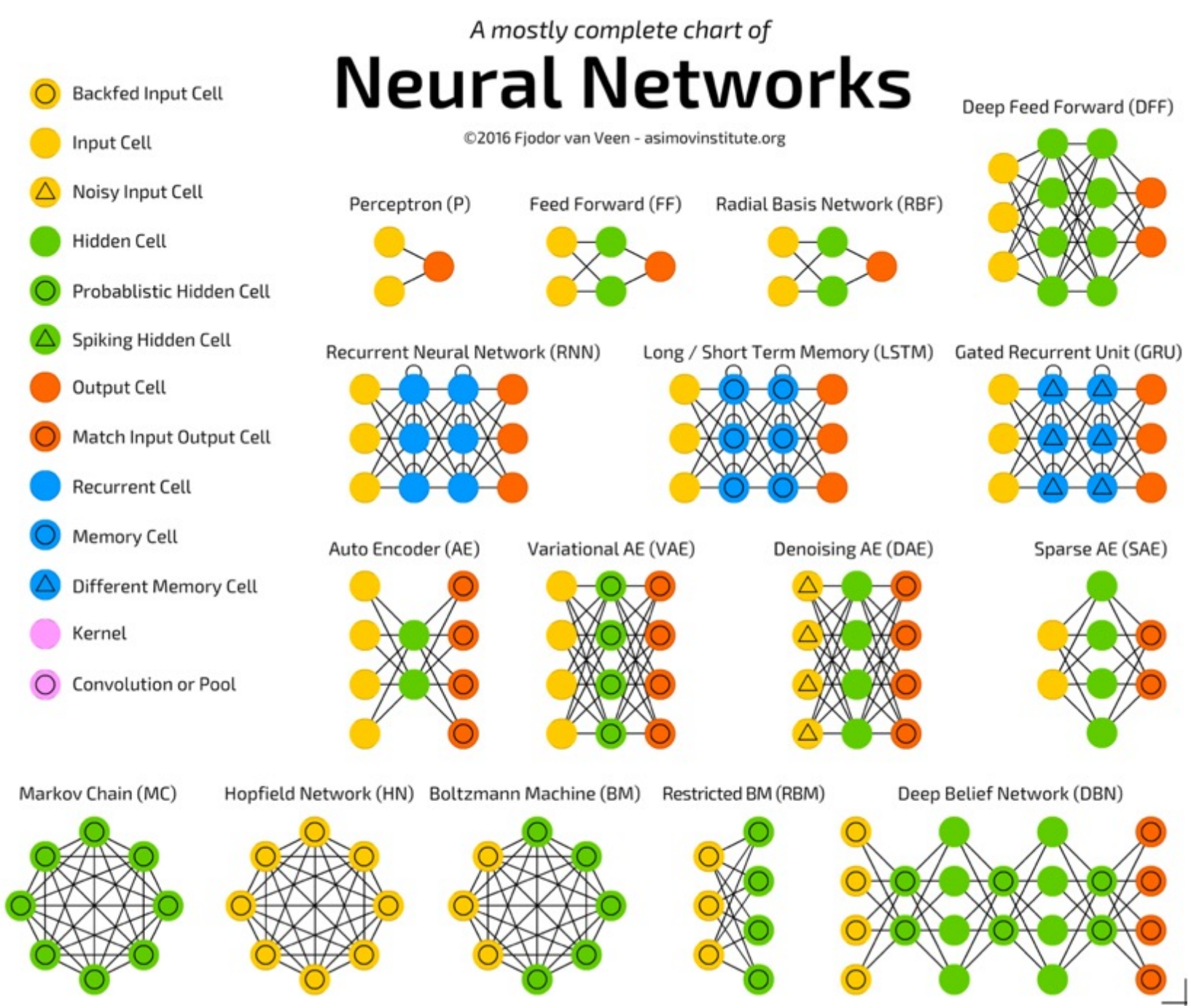


Deep Learning
and
Deep Neural Networks

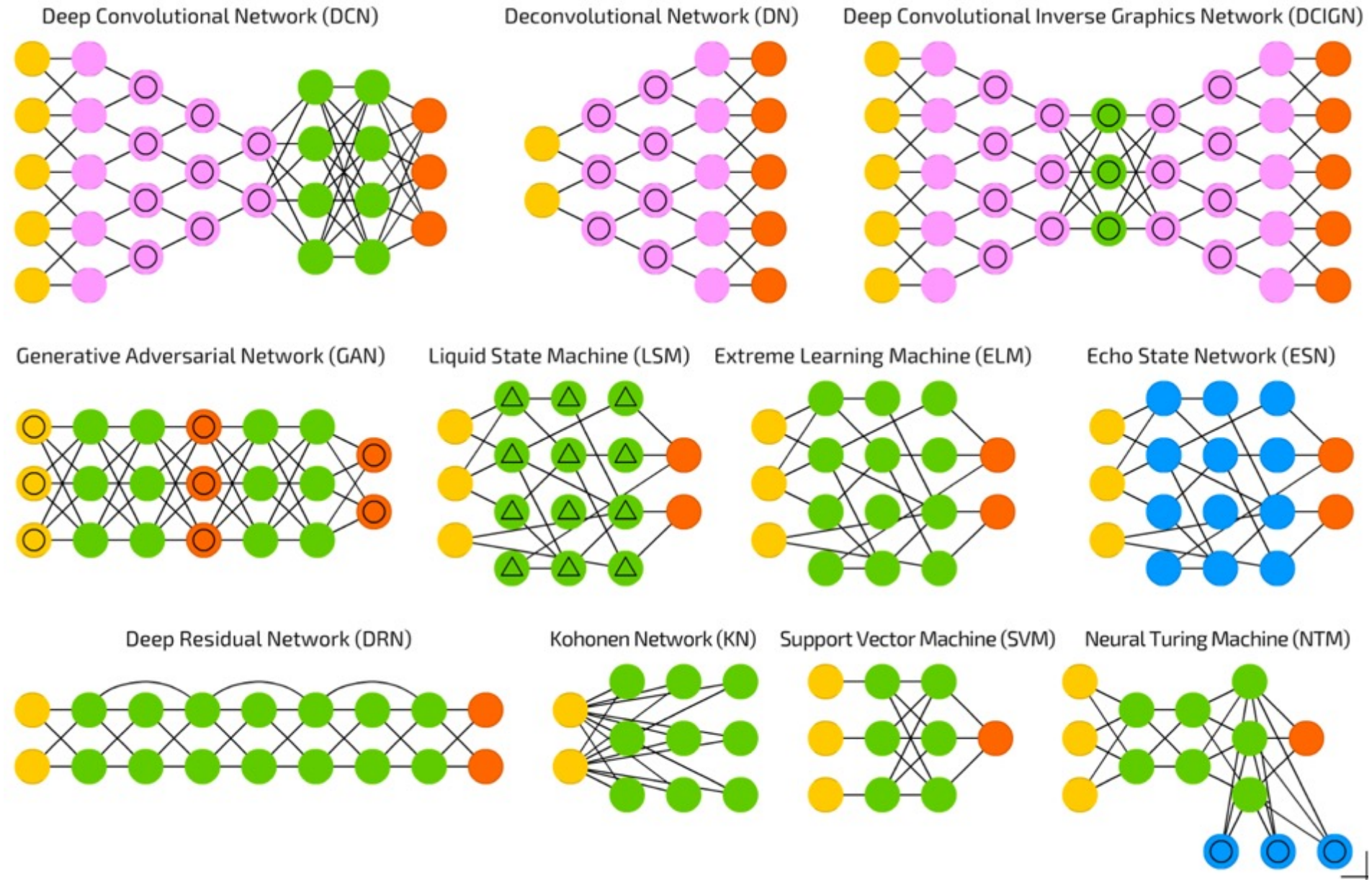
Neural Networks (NN)



Source: http://www.asimovinstitute.org/neural-network-zoo/



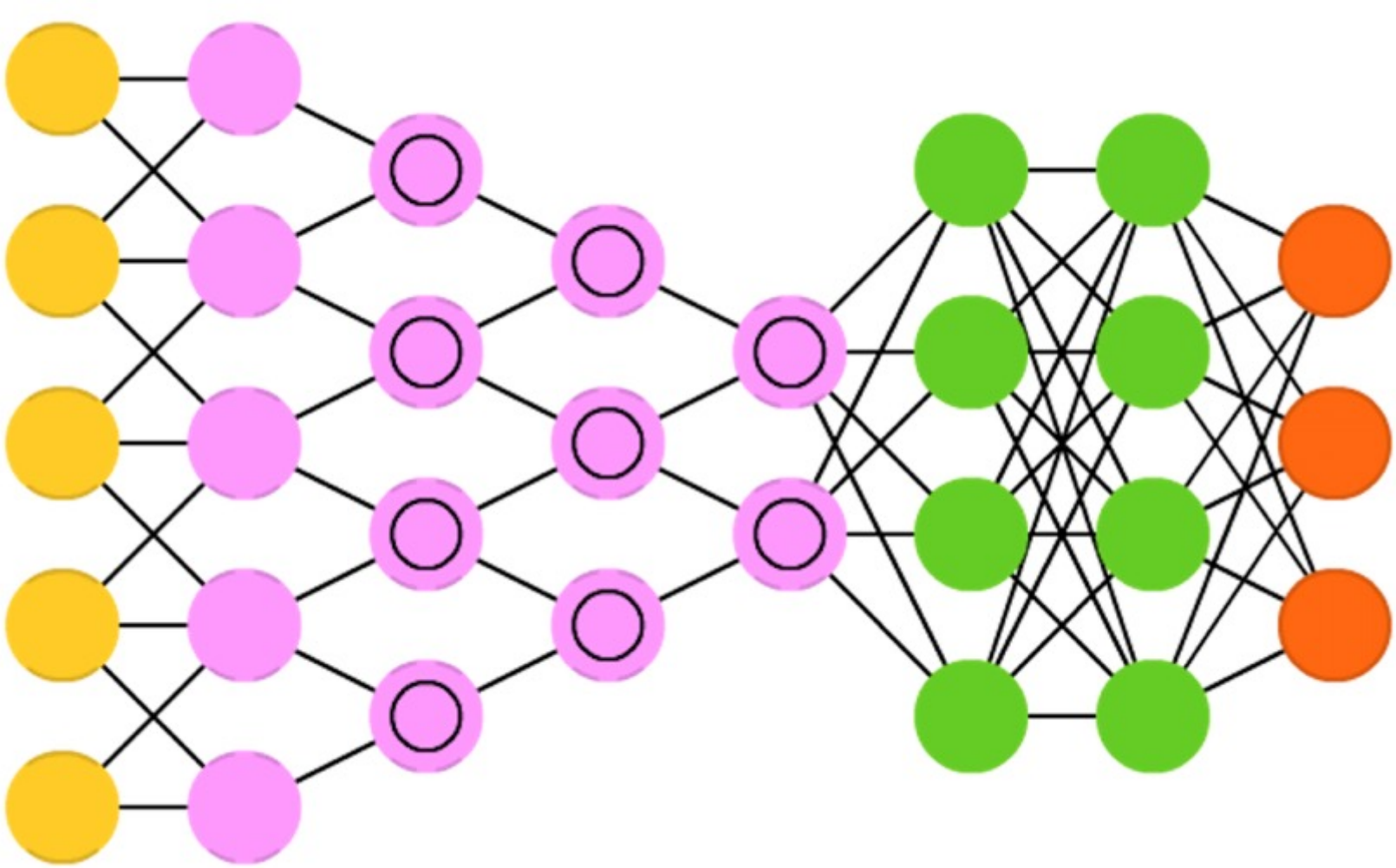
Source: http://www.asimovinstitute.org/neural-network-zoo/



Source: http://www.asimovinstitute.org/neural-network-zoo/

Convolutional Neural Networks

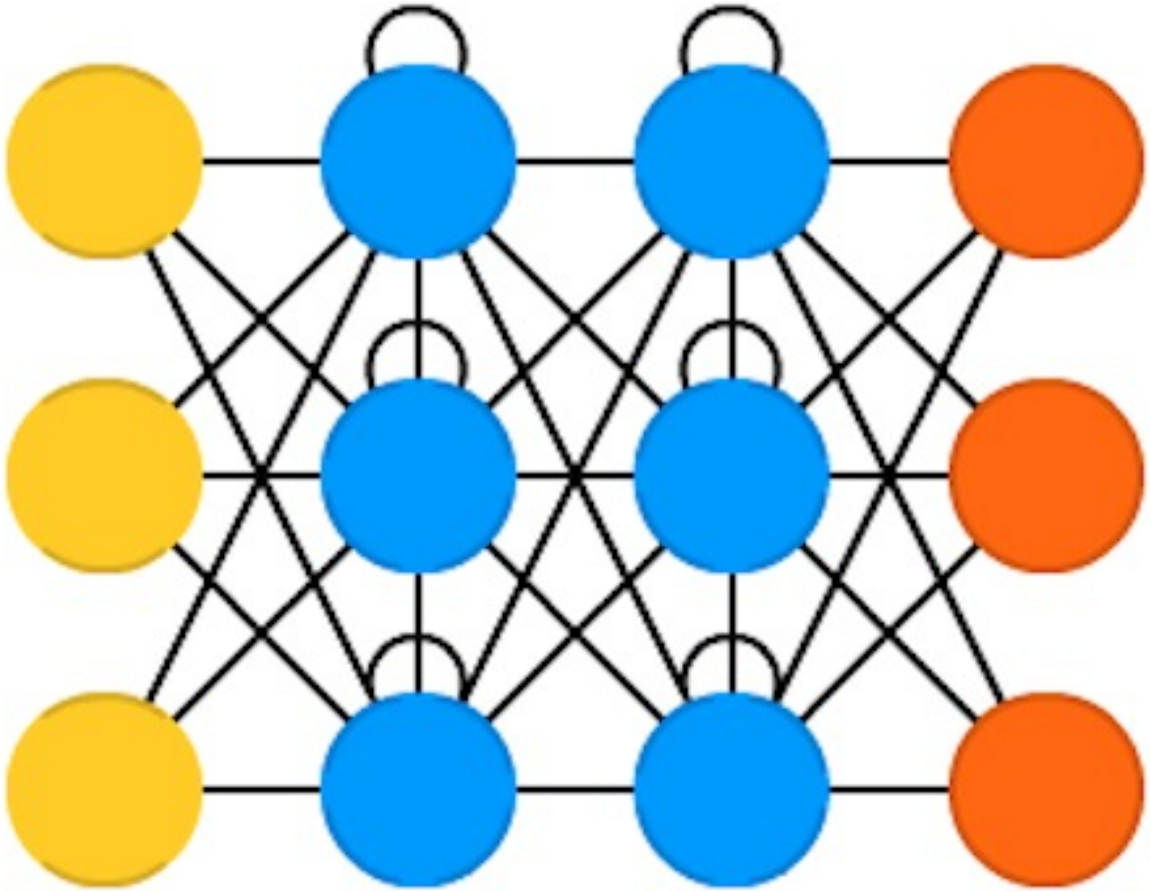
(CNN or Deep Convolutional Neural Networks, DCNN)



LeCun, Yann, et al. “Gradient-based learning applied to document recognition.” Proceedings of the IEEE 86.11 (1998): 2278-2324.

Source: http://www.asimovinstitute.org/neural-network-zoo/

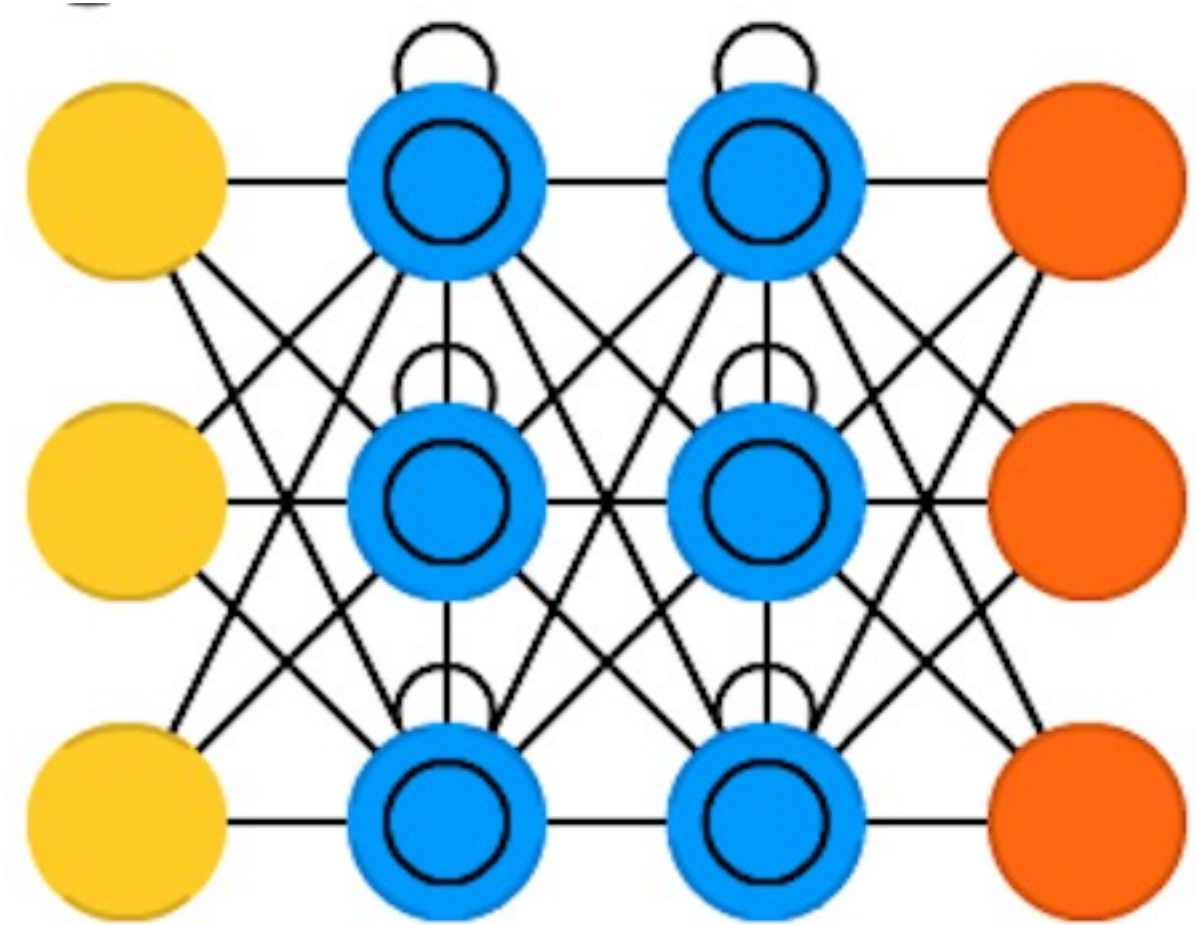
Recurrent Neural Networks (RNN)



Elman, Jeffrey L. “Finding structure in time.” Cognitive science 14.2 (1990): 179-211

Source: http://www.asimovinstitute.org/neural-network-zoo/

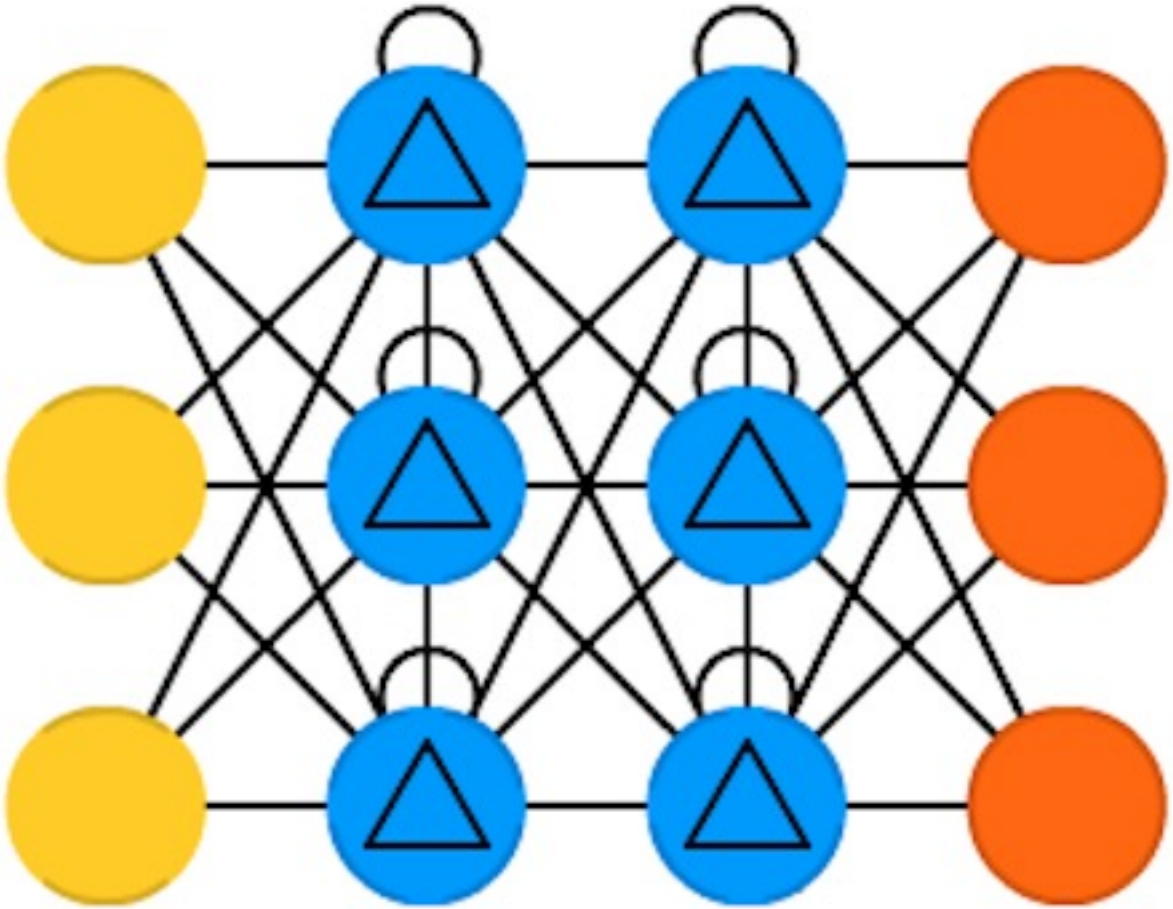
Long / Short Term Memory (LSTM)



Hochreiter, Sepp, and Jürgen Schmidhuber. “Long short-term memory.” Neural computation 9.8 (1997): 1735-1780.

Source: http://www.asimovinstitute.org/neural-network-zoo/

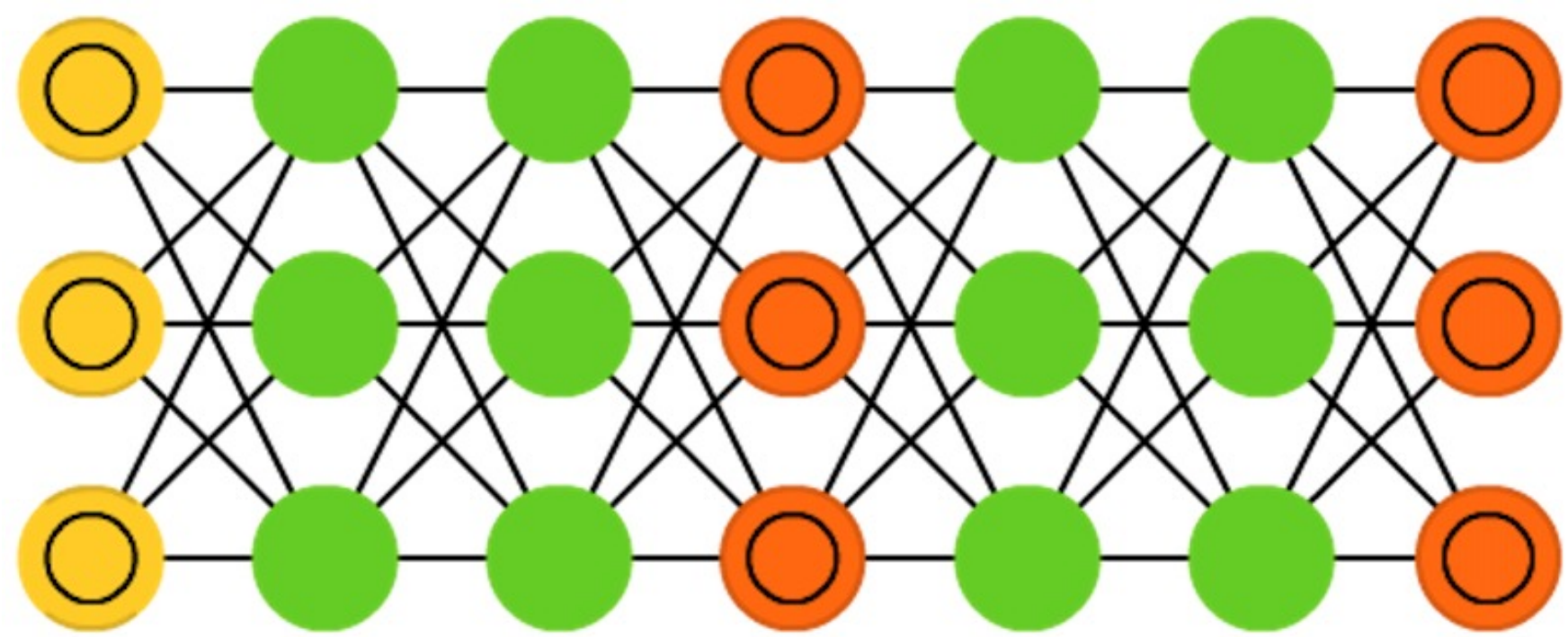
Gated Recurrent Units (GRU)



Chung, Junyoung, et al. “Empirical evaluation of gated recurrent neural networks on sequence modeling.” arXiv preprint arXiv:1412.3555 (2014).

Source: http://www.asimovinstitute.org/neural-network-zoo/

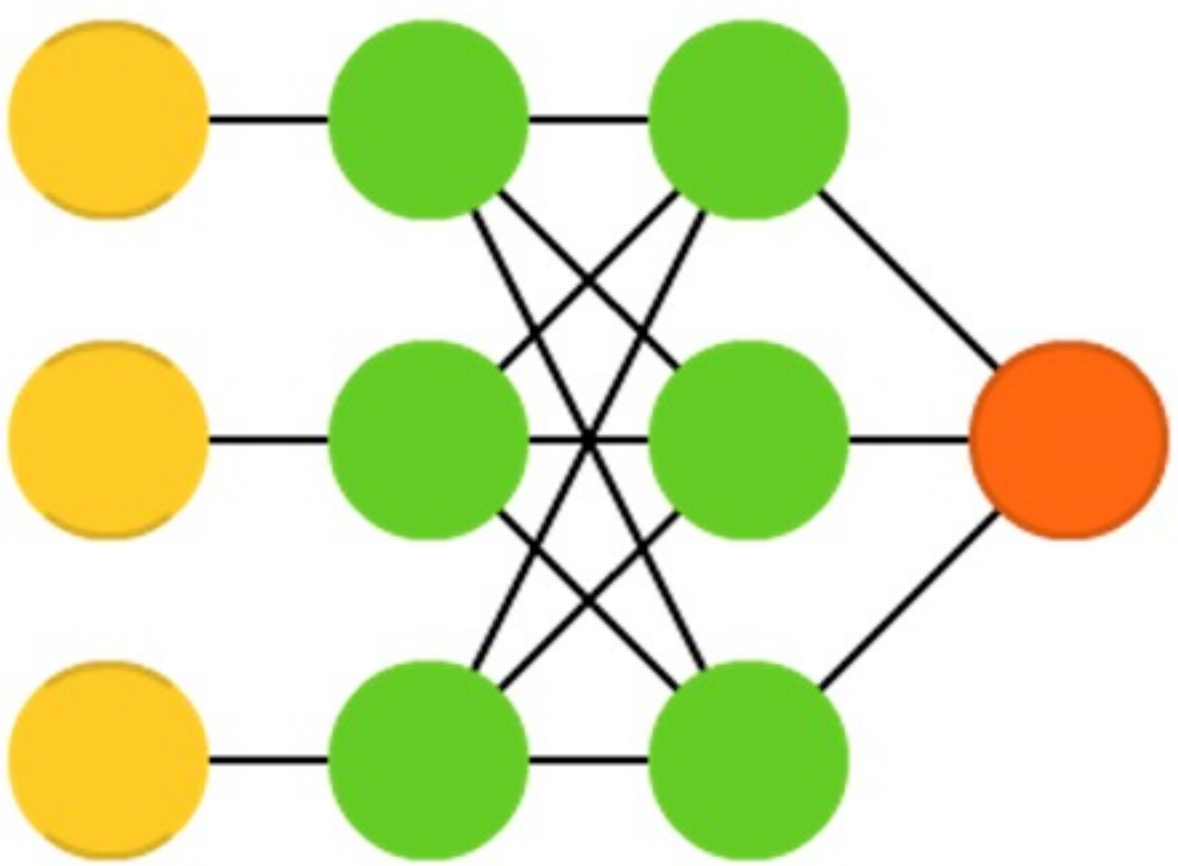
Generative Adversarial Networks (GAN)



Goodfellow, Ian, et al. “Generative adversarial nets.” Advances in Neural Information Processing Systems. 2014.

Source: http://www.asimovinstitute.org/neural-network-zoo/

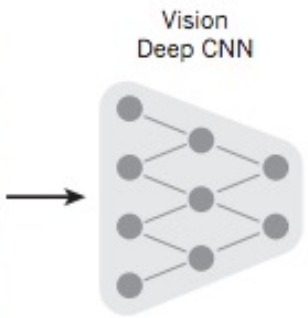
Support Vector Machines (SVM)

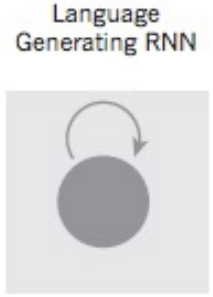


Cortes, Corinna, and Vladimir Vapnik. “Support-vector networks.” Machine learning 20.3 (1995): 273-297.

Source: http://www.asimovinstitute.org/neural-network-zoo/

From image to text





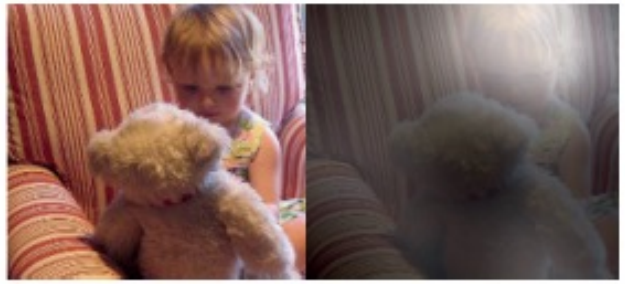
A group of people shopping at an outdoor market.

There are many vegetables at the fruit stand.

A woman is throwing a **frisbee** in a park.

A **dog** is standing on a hardwood floor.

A **stop** sign is on a road with a mountain in the background



A little **girl** sitting on a bed with a teddy bear.

A group of **people** sitting on a boat in the water.

A giraffe standing in a forest with **trees** in the background.

Source: LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." Nature 521, no. 7553 (2015): 436-444.

From image to text

Image: deep convolution neural network (CNN)

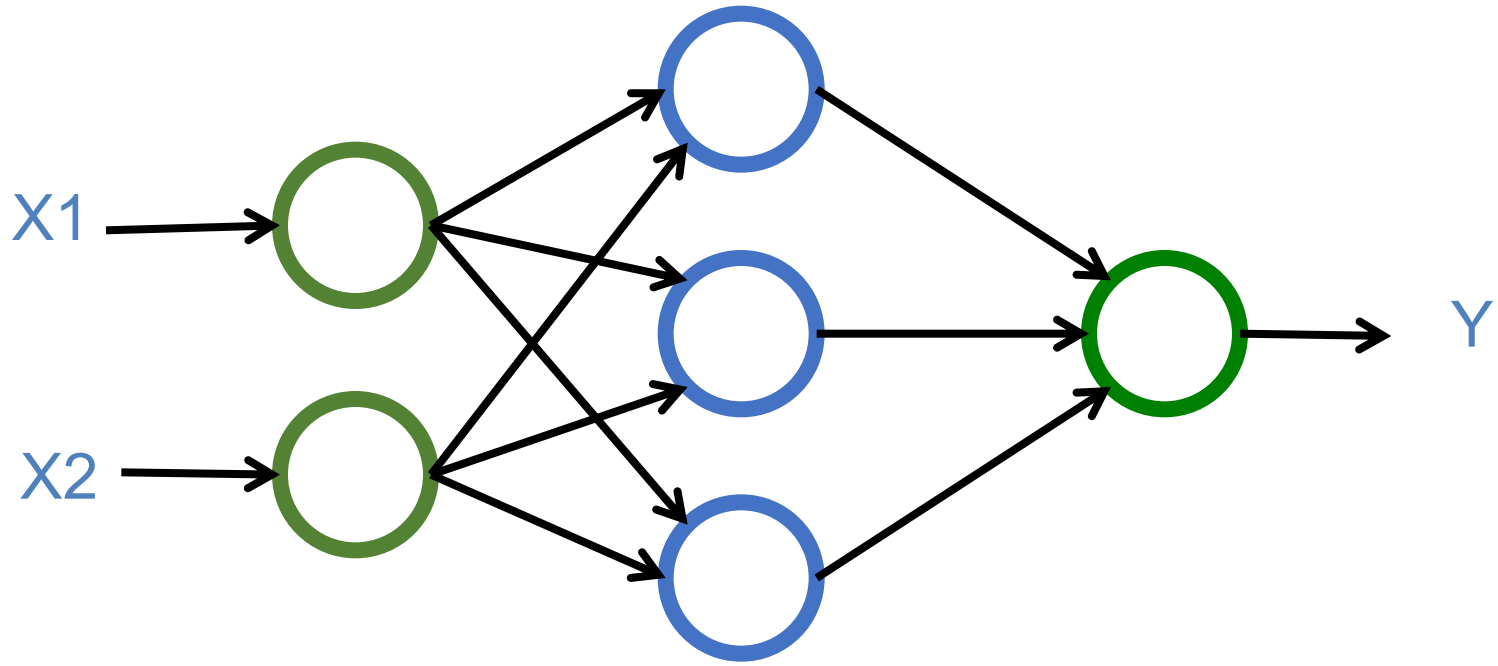
Text: recurrent neural network (RNN)

A group of **people** sitting on a boat in the water.

Source: LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." Nature 521, no. 7553 (2015): 436-444.

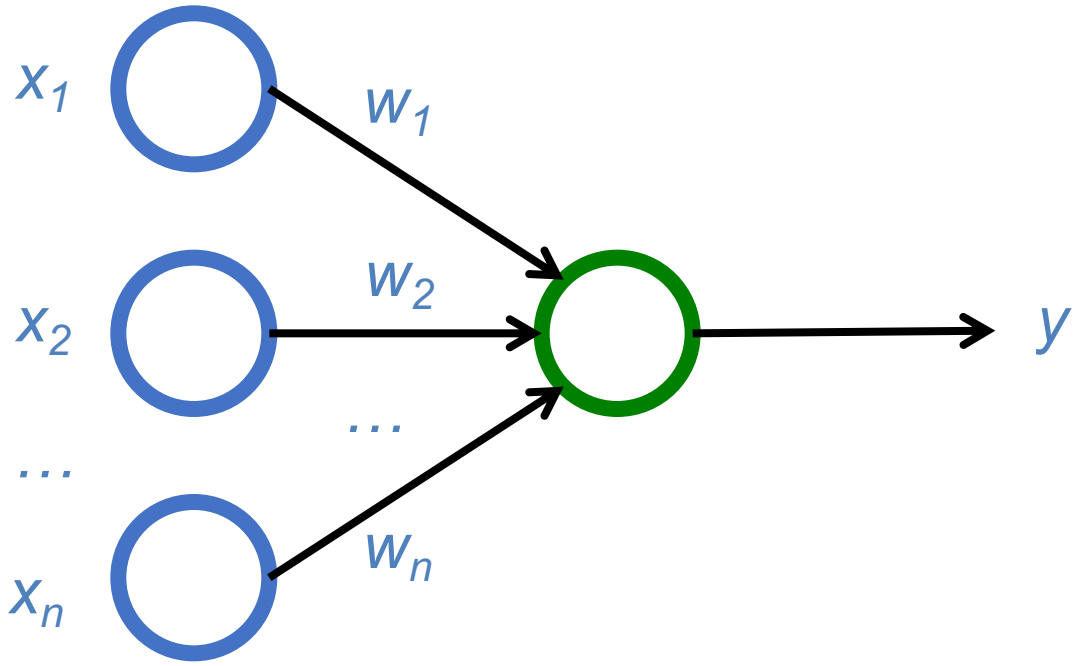
Neural Networks

Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



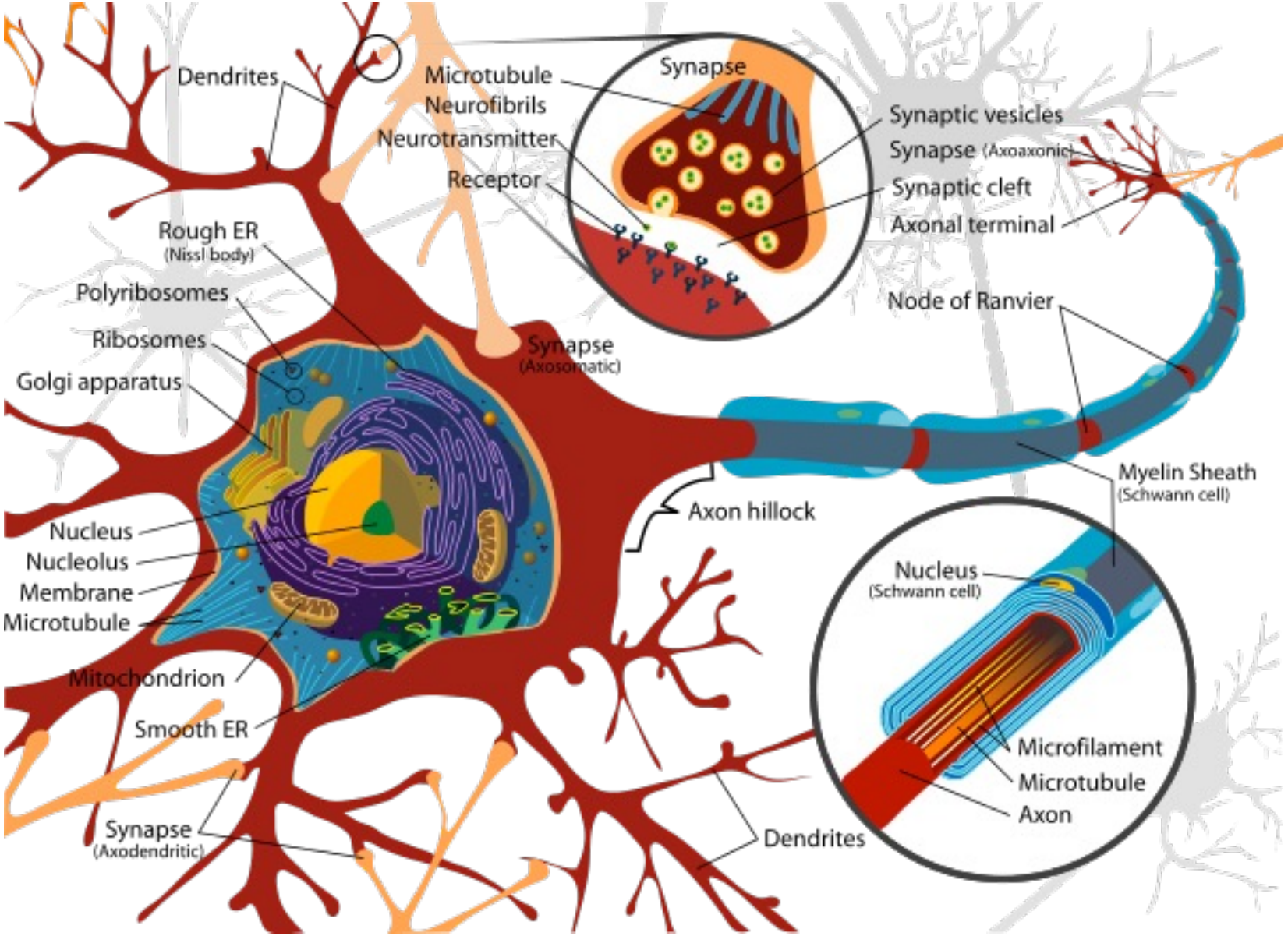
Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

The Neuron



Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

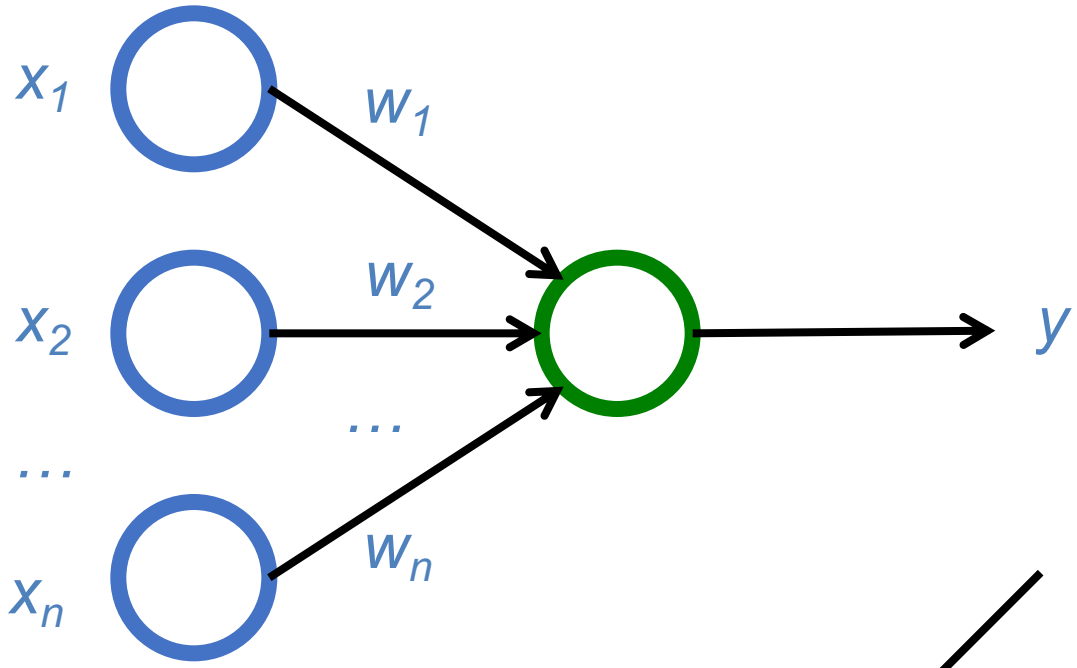
Neuron and Synapse

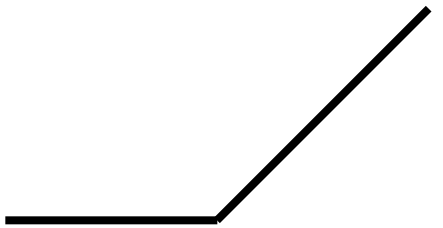


Source: https://en.wikipedia.org/wiki/Neuron

The Neuron

$$y = F\left(\sum_i w_i x_i\right)$$

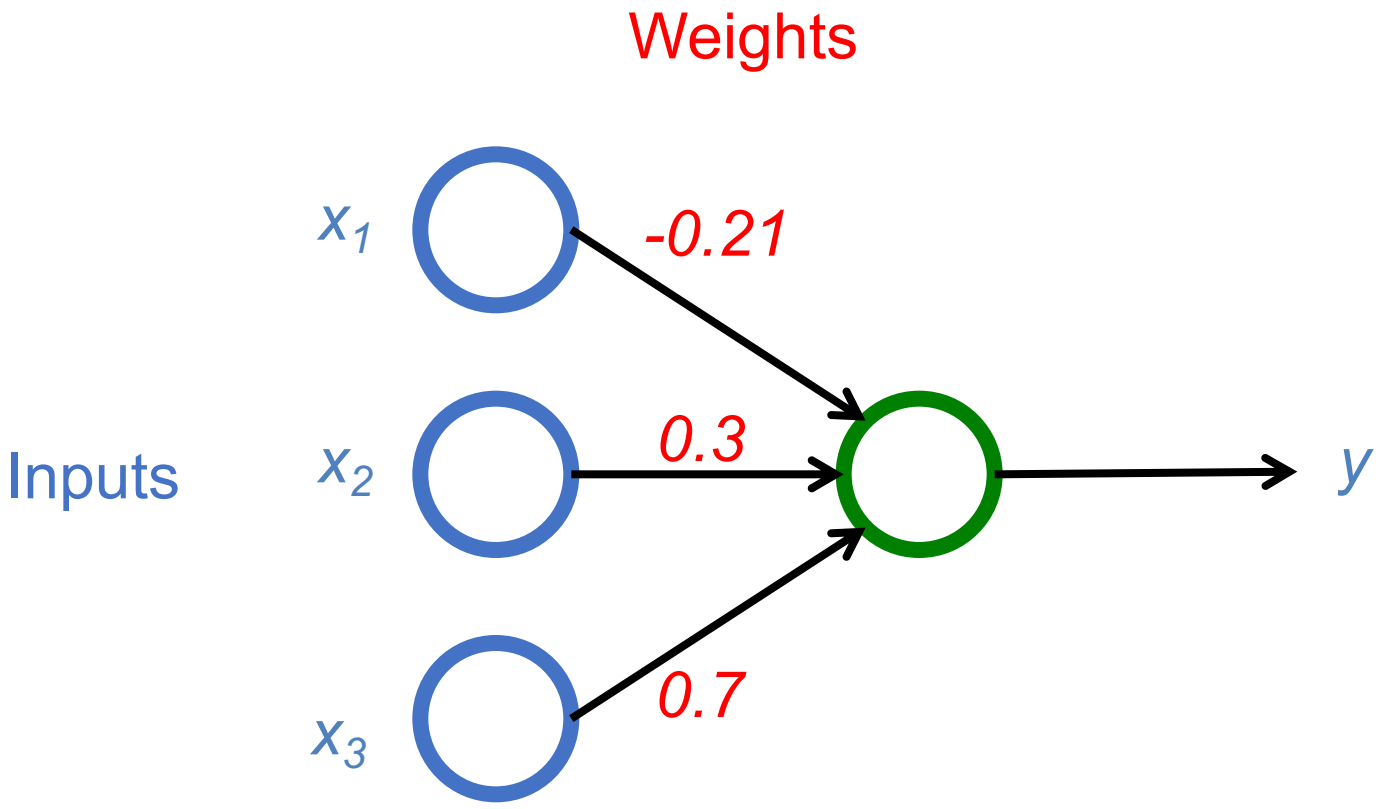




$$F(x) = \max(0, x)$$

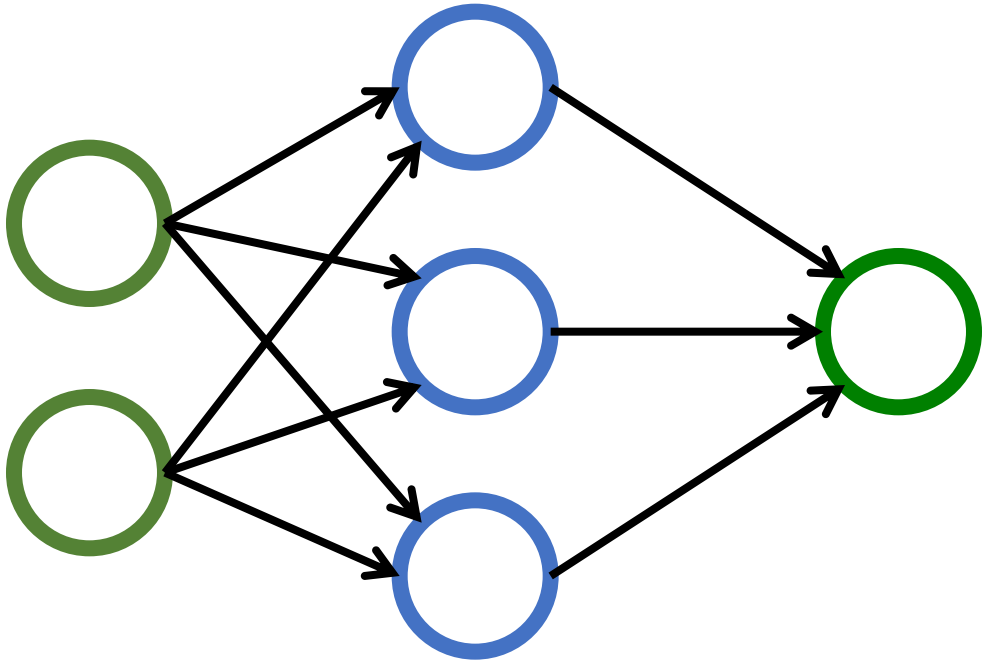
Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

$$y = max\ (\ 0,\ -0.21\ *\ x_1 + 0.3\ *\ x_2 + 0.7\ *\ x_3\ )$$



Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

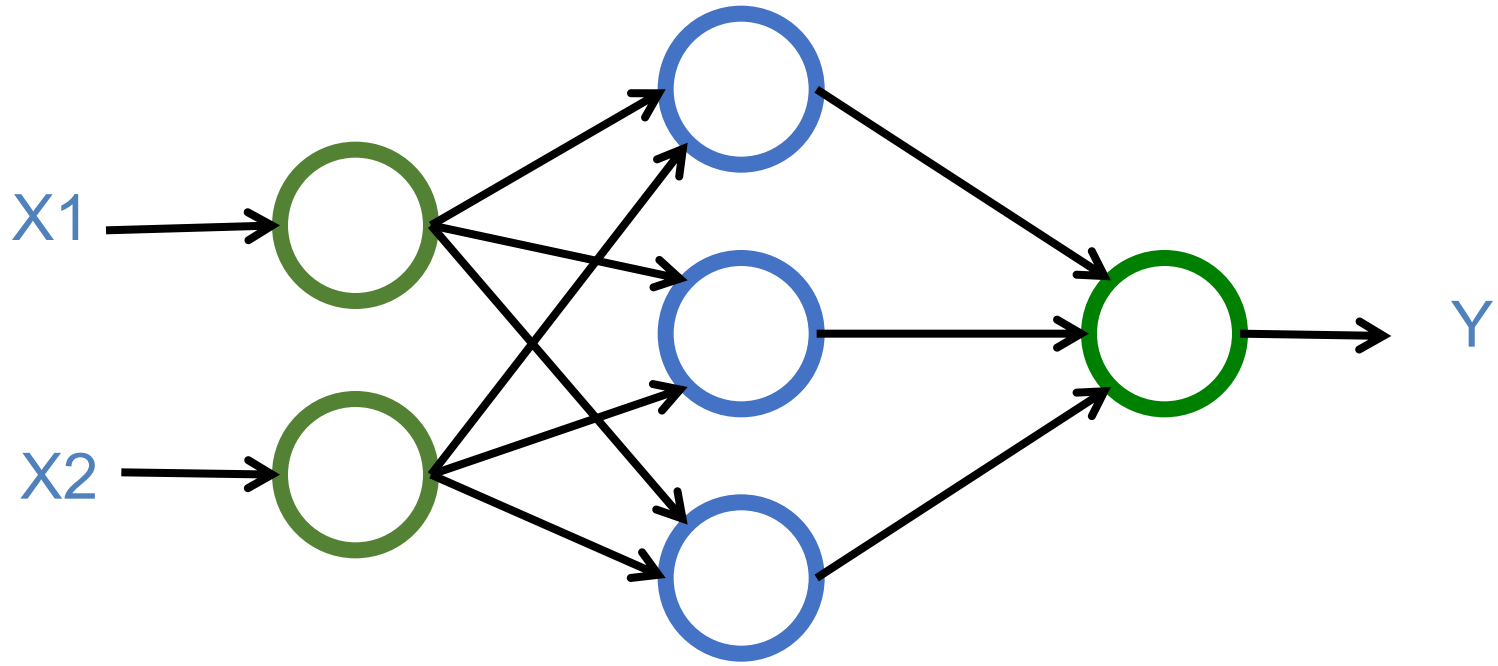
Neural Networks



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Neural Networks

Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

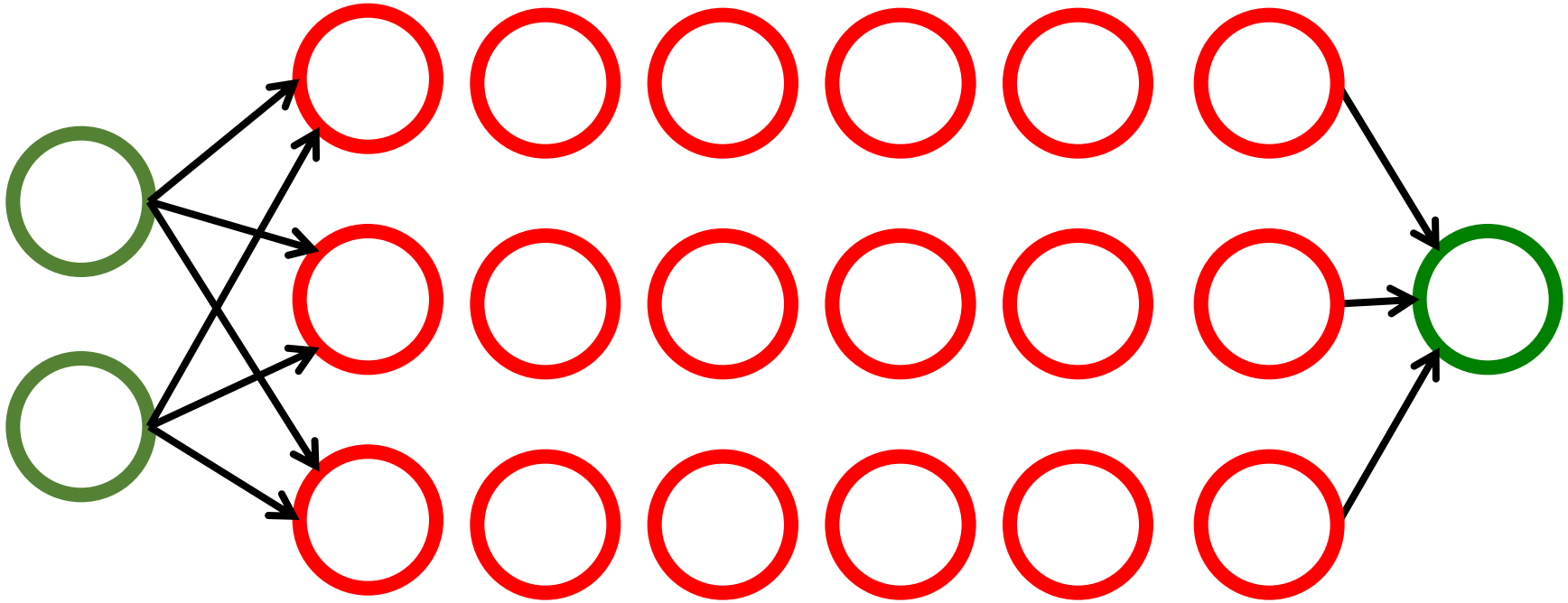
Neural Networks

Input Layer
(X)

Hidden Layers
(H)

Output Layer
(Y)

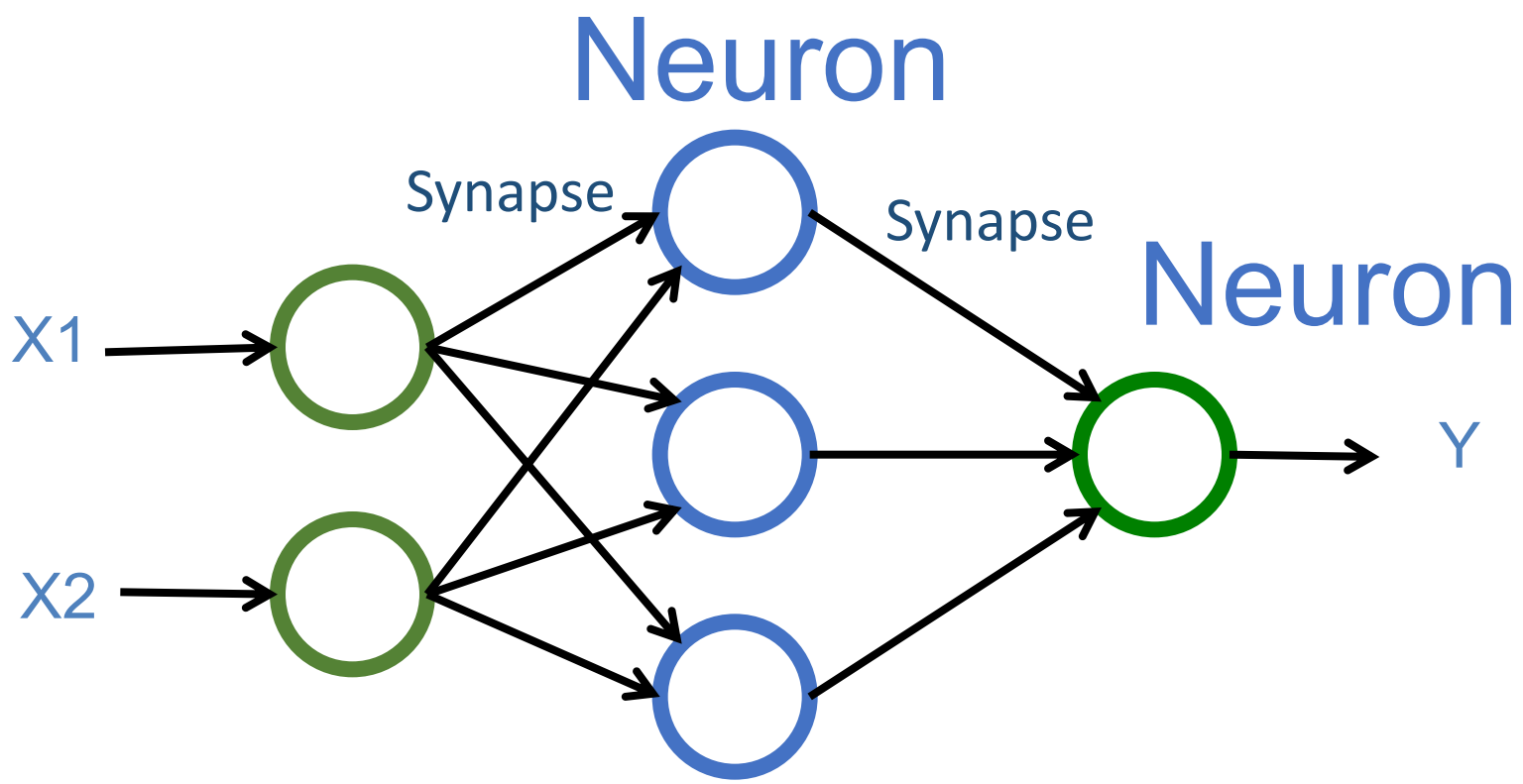
Deep Neural Networks
Deep Learning



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Neural Networks

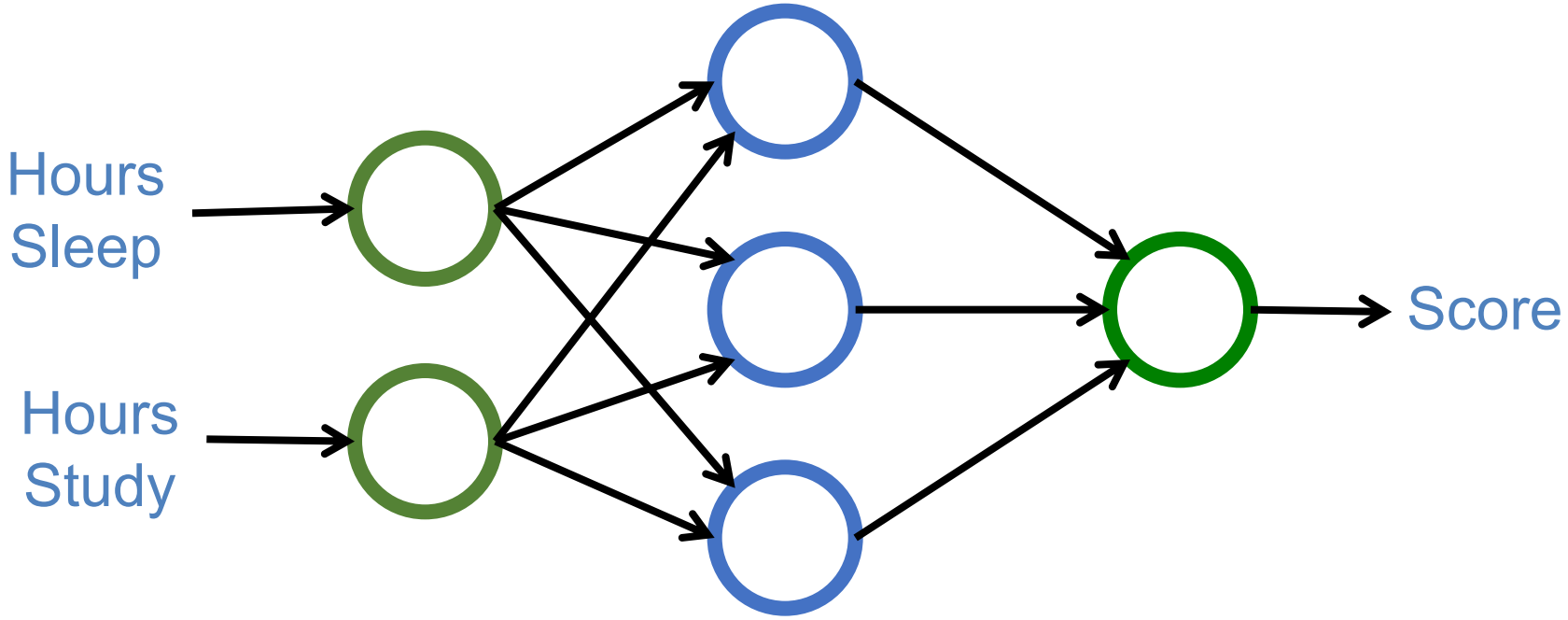
Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Neural Networks

Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



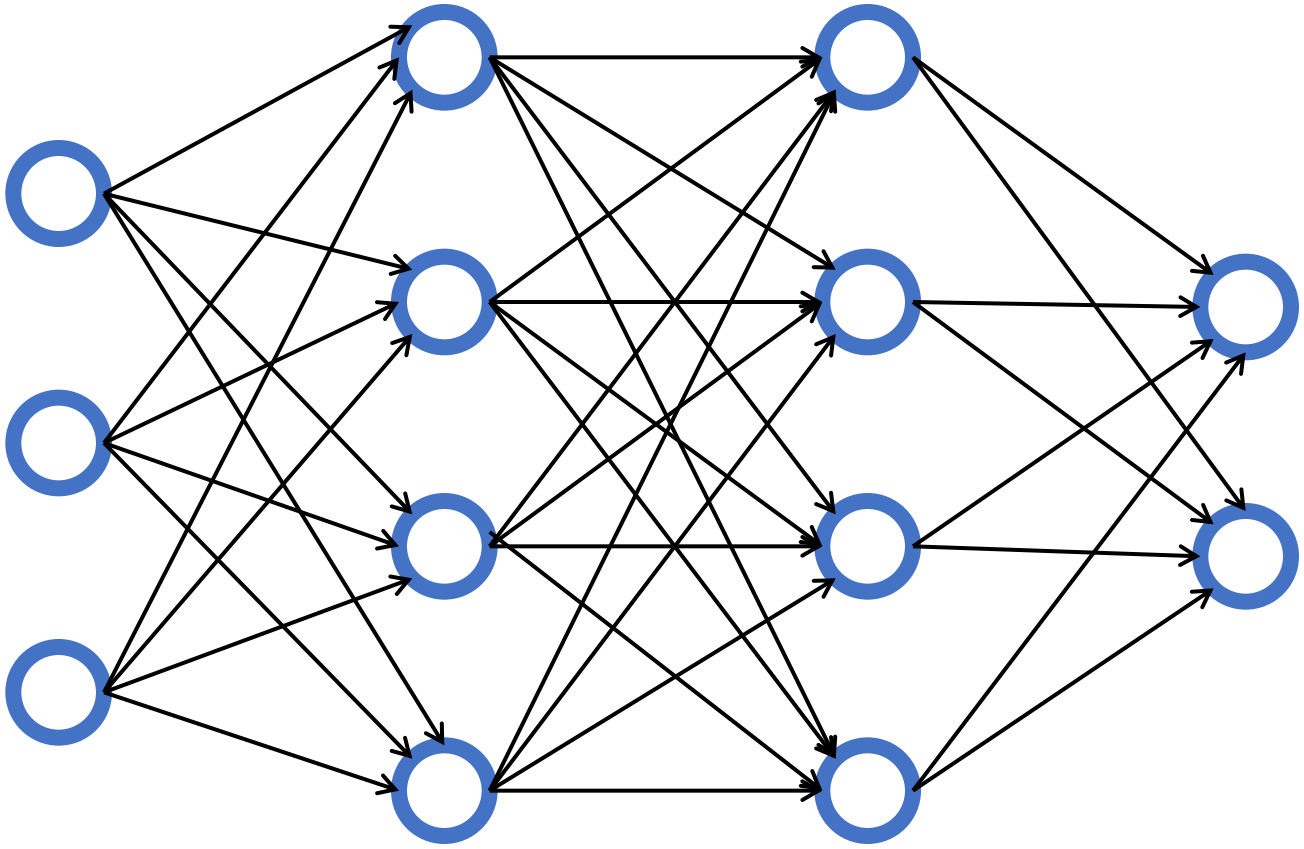
Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Neural Networks

Input Layer
(X)

Hidden Layer
(H)

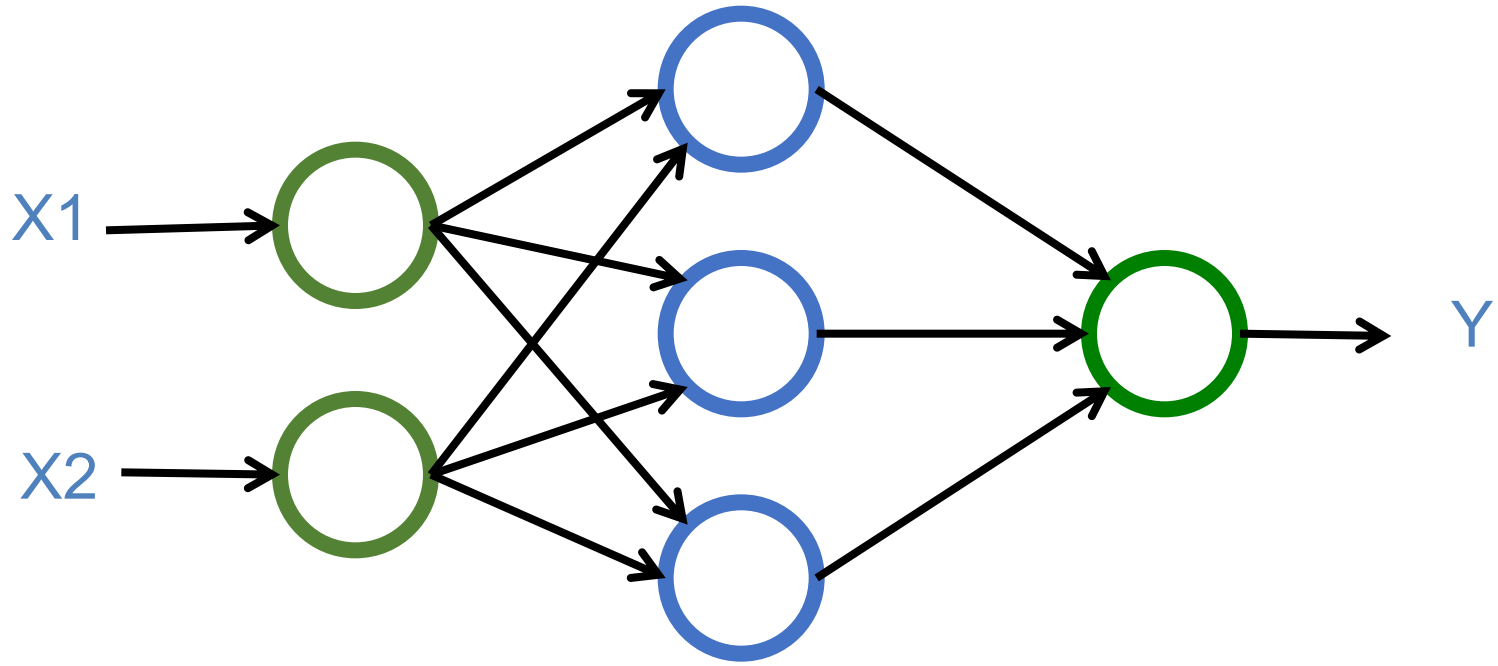
Output Layer
(Y)



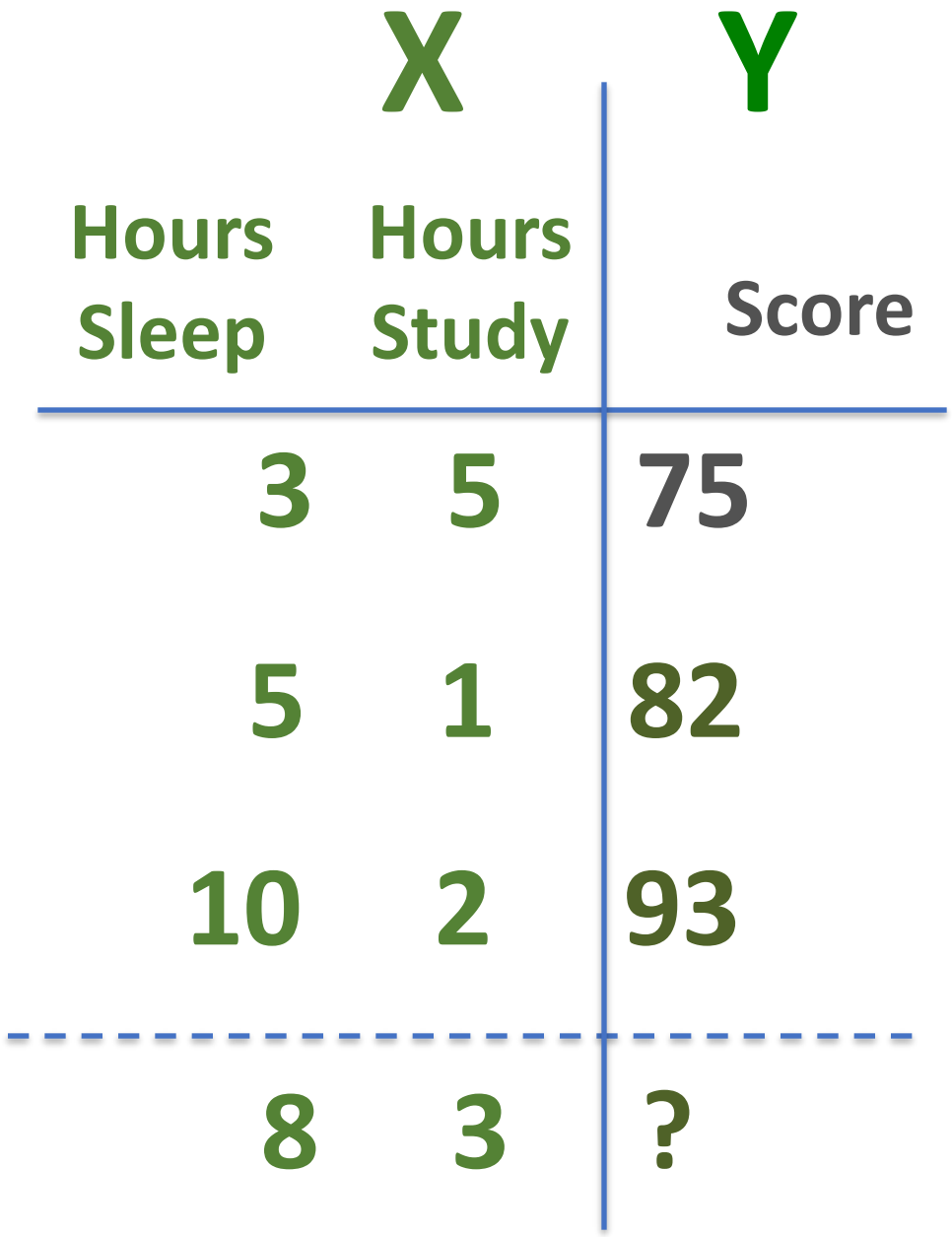
Source: https://www.youtube.com/watch?v=P2HPcj8lRJE&list=PLjJh1vlSEYgvGod9wWiydumYl8hOXixNu&index=2

Neural Networks

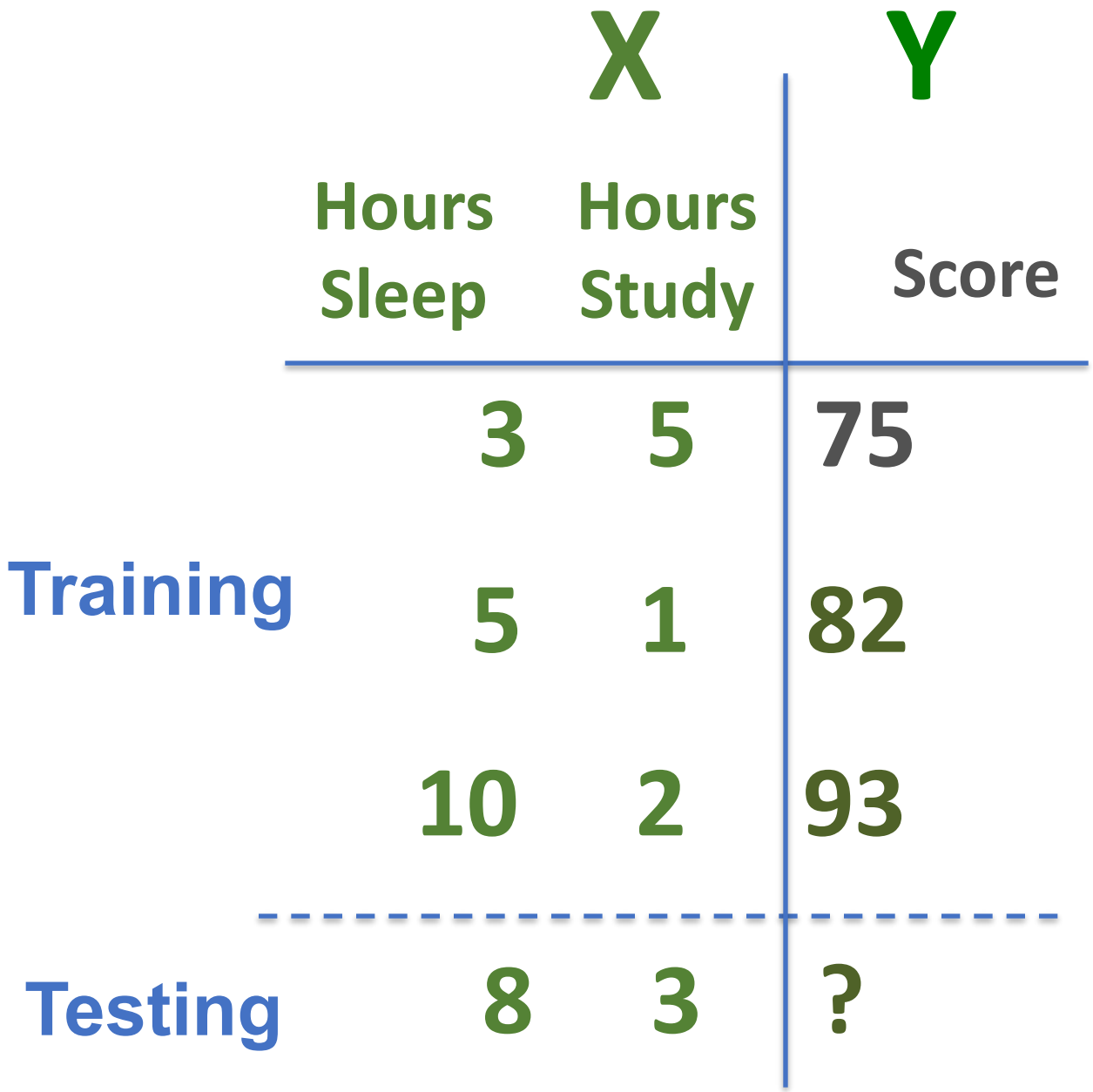
Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU



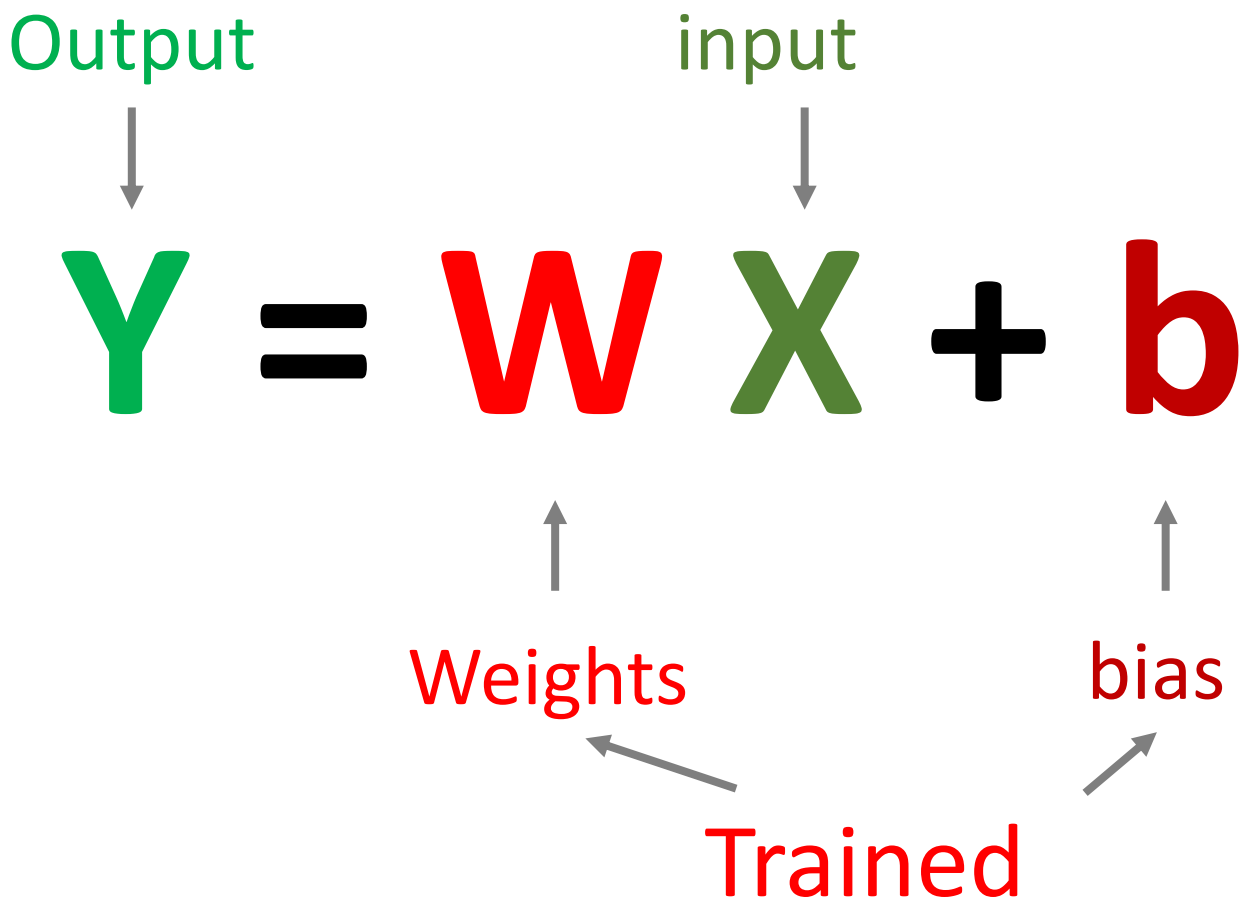
Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

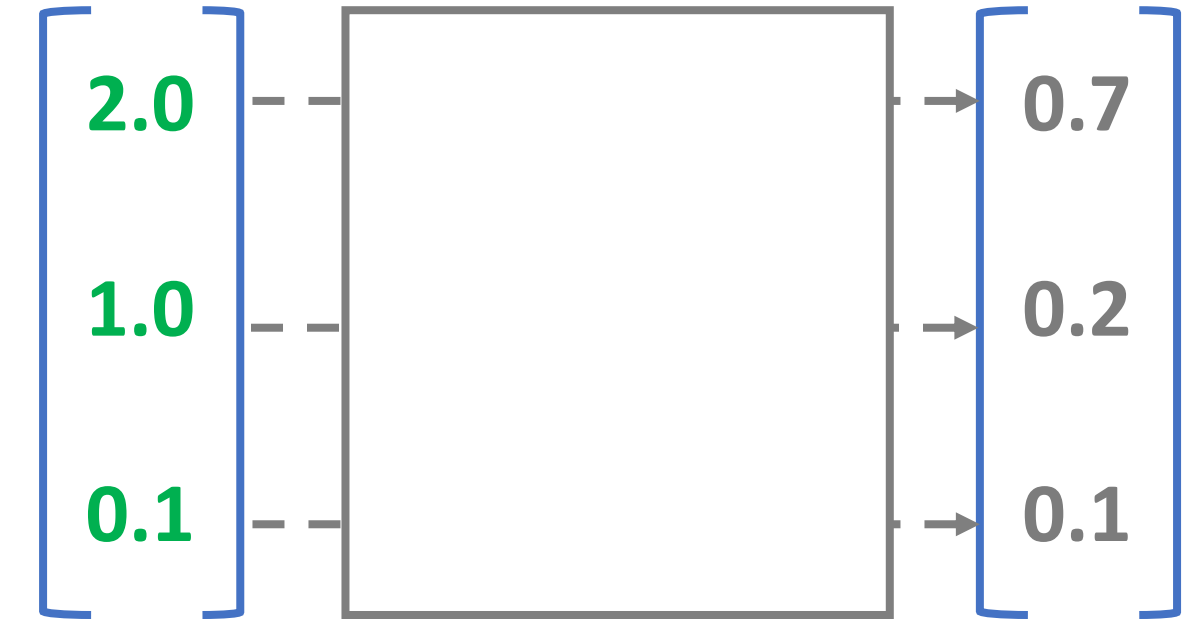
$$Y = W X + b$$

Source: https://www.youtube.com/watch?v=G8eNWzxOgqE



Source: https://www.youtube.com/watch?v=G8eNWzxOgqE

$$W X + b = Y$$

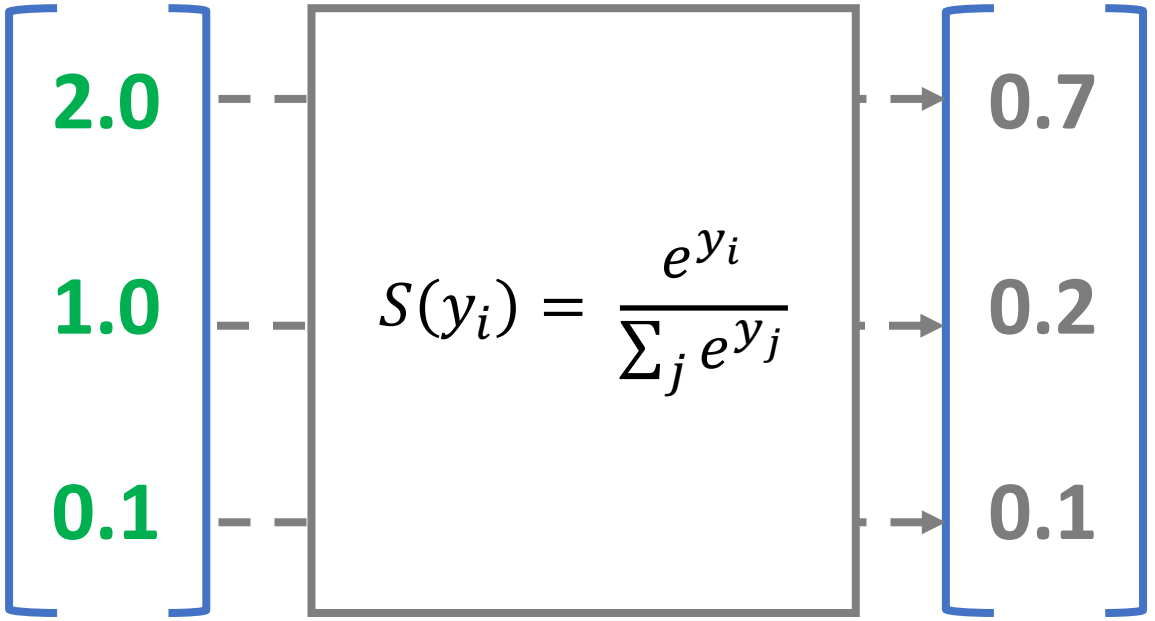


Scores → Probabilities

Source: https://www.youtube.com/watch?v=G8eNWzxOgqE

SoftMAX

$$W\,X + b = Y$$



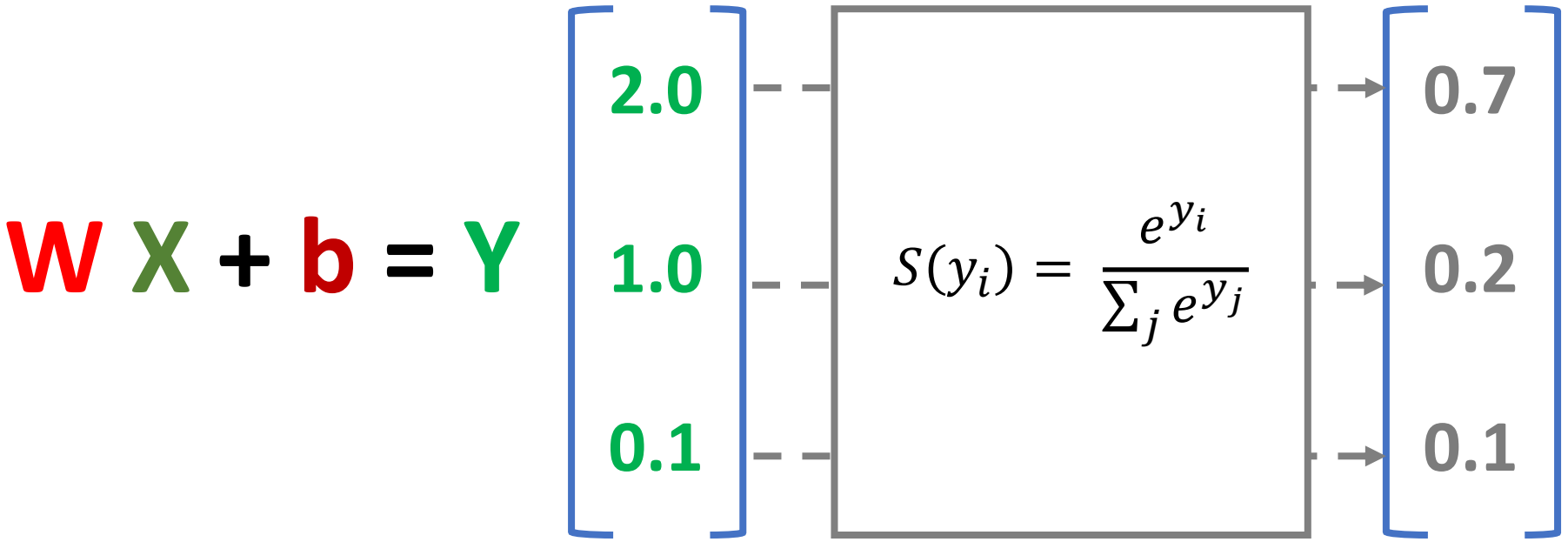
Logits Scores → Probabilities

Source: https://www.youtube.com/watch?v=G8eNWzxOgqE

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_i}} = \frac{e^{2.0}}{e^{2.0}+e^{1.0}+e^{0.1}} = \frac{2.7182^{2.0}}{2.7182^{2.0}+2.7182^{1.0}+2.7182^{0.1}} = 0.7$$

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} = \frac{e^{1.0}}{e^{2.0}+e^{1.0}+e^{0.1}} = \frac{2.7182^{1.0}}{2.7182^{2.0}+2.7182^{1.0}+2.7182^{0.1}} = 0.2$$

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} = \frac{e^{0.1}}{e^{2.0}+e^{1.0}+e^{0.1}} = \frac{2.7182^{0.1}}{2.7182^{2.0}+2.7182^{1.0}+2.7182^{0.1}} = 0.1$$



Logits Scores → Probabilities

Source: https://www.youtube.com/watch?v=G8eNWzxOgqE

Training a Network

=

Minimize the Cost Function

Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Training a Network

=

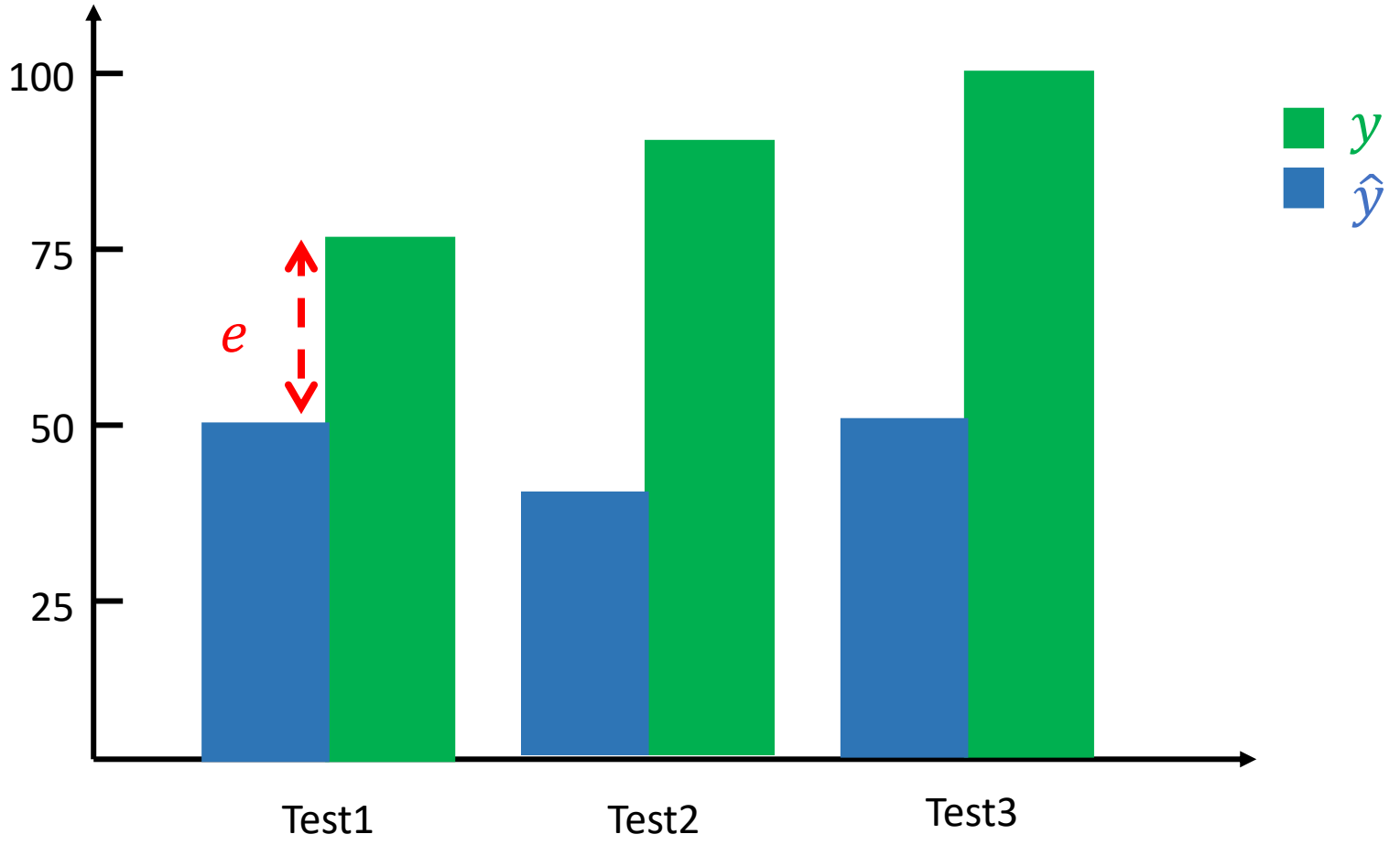
Minimize the Cost Function

Minimize the Loss Function

Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Error = Predict Y - Actual Y

Error : Cost : Loss



Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

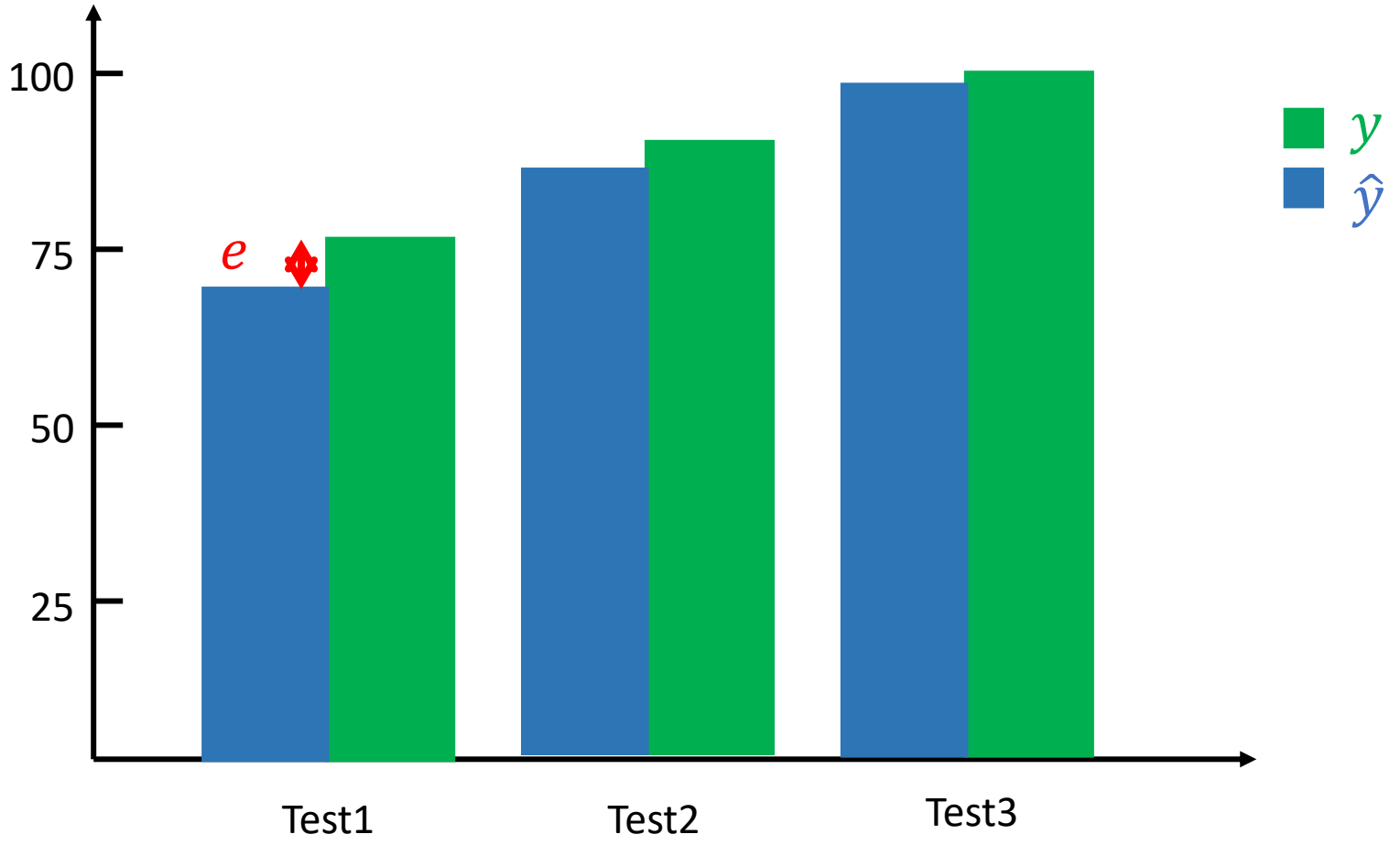
Error = Predict Y - Actual Y

Error : Cost : Loss

Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Error = Predict Y - Actual Y

Error : Cost : Loss

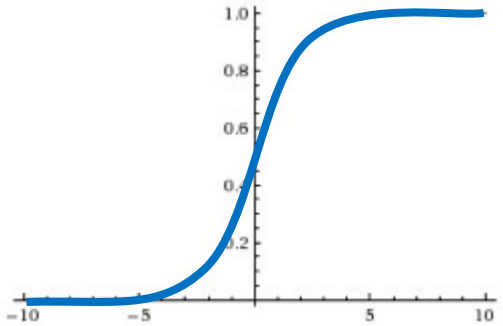


Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Activation Functions

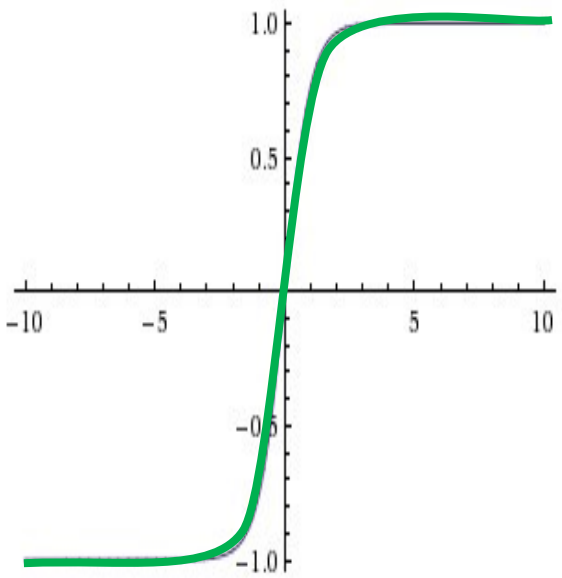
Activation Functions

Sigmoid



[0, 1]

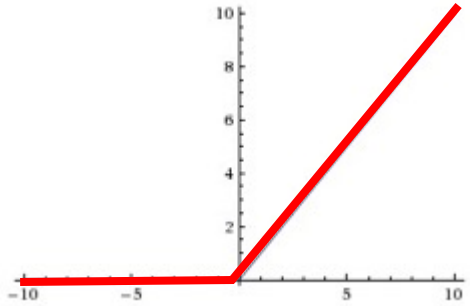
TanH



[-1, 1]

ReLU

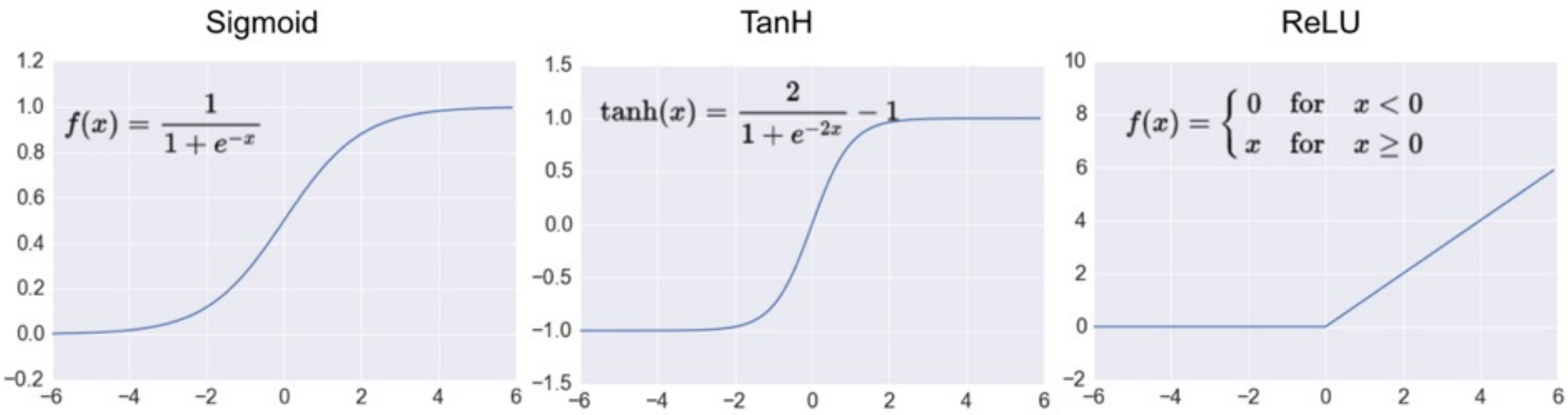
(Rectified Linear Unit)



$f(x) = max(0, x)$

Source: http://cs231n.github.io/neural-networks-1/

Activation Functions



Source: http://adilmoujahid.com/posts/2016/06/introduction-deep-learning-python-caffe/

Loss Function

Binary Classification: 2 Class

Activation Function:
Sigmoid

Loss Function:
Binary Cross-Entropy

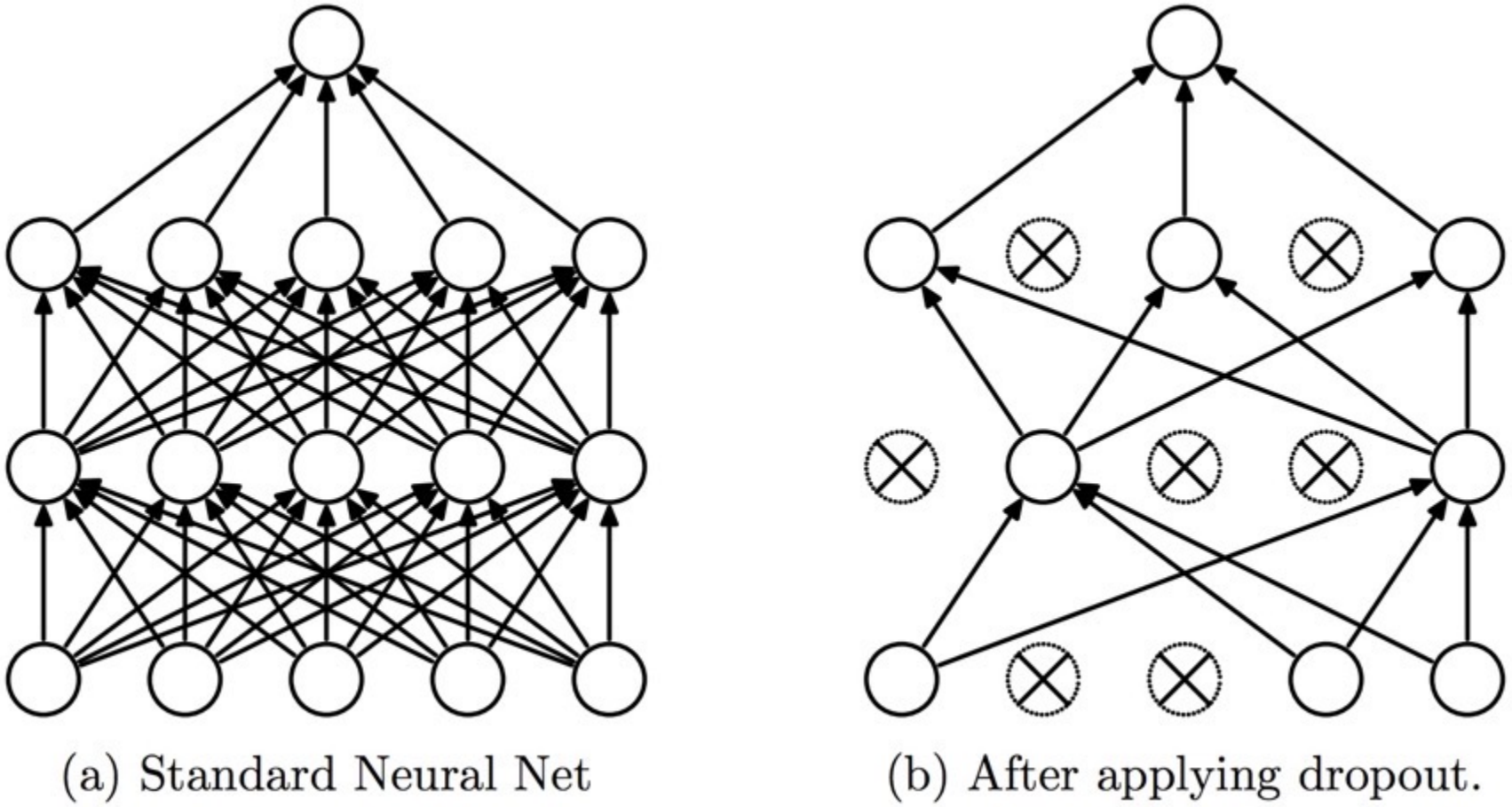
Multiple Classification: 10 Class

Activation Function:
SoftMAX

Loss Function:
Categorical Cross-Entropy

Dropout

Dropout: a simple way to prevent neural networks from overfitting



(a) Standard Neural Net

(b) After applying dropout.

Source: Srivastava, Nitish, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov.
"Dropout: a simple way to prevent neural networks from overfitting." *Journal of machine learning research* 15, no. 1 (2014): 1929-1958.

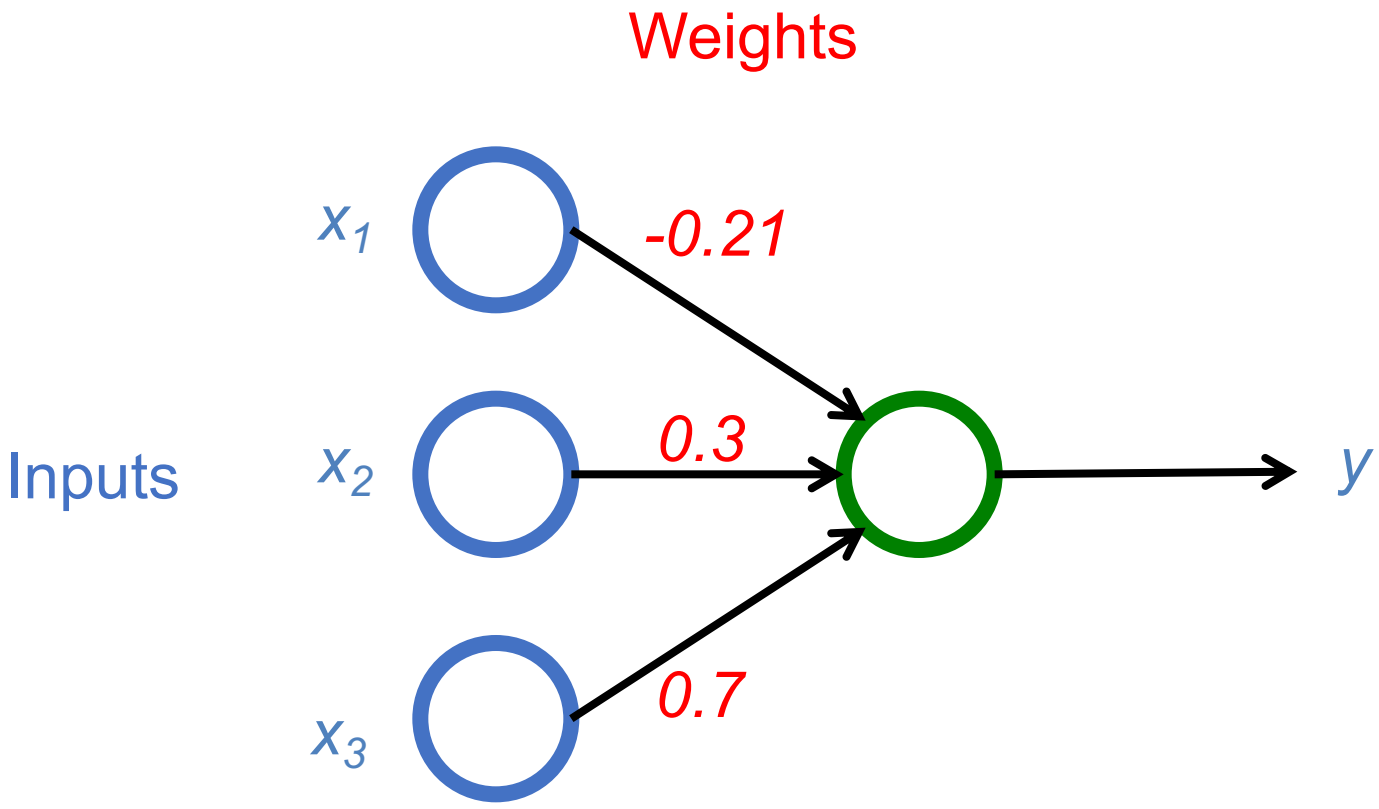
Learning Algorithm

While not done:

- **Pick a random training example "(input, label)"**
- **Run neural network on "input"**
- **Adjust weights on edges to make output closer to "label"**

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

$$y = max\ (\ 0,\ -0.21\ *\ x_1 + 0.3\ *\ x_2 + 0.7\ *\ x_3\ )$$

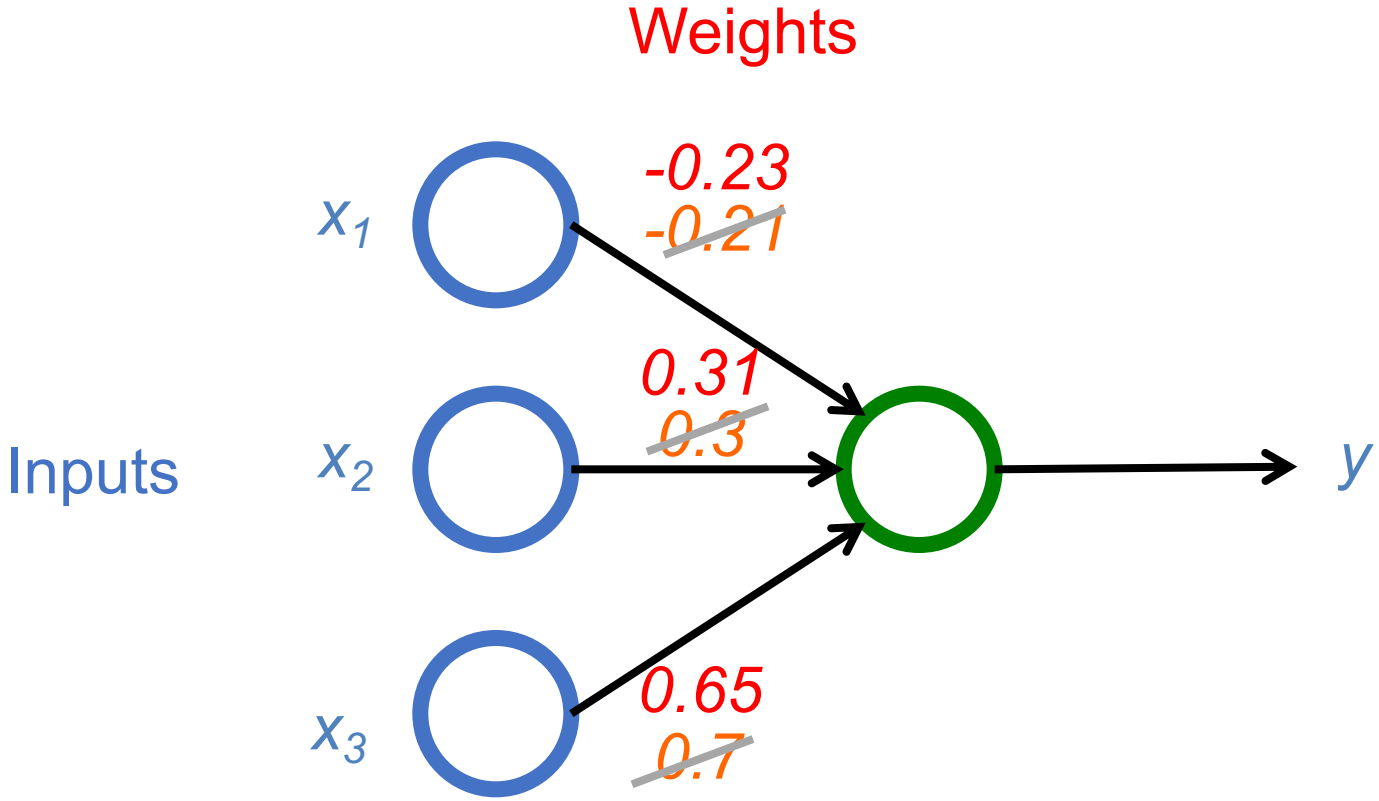


Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

Next time:

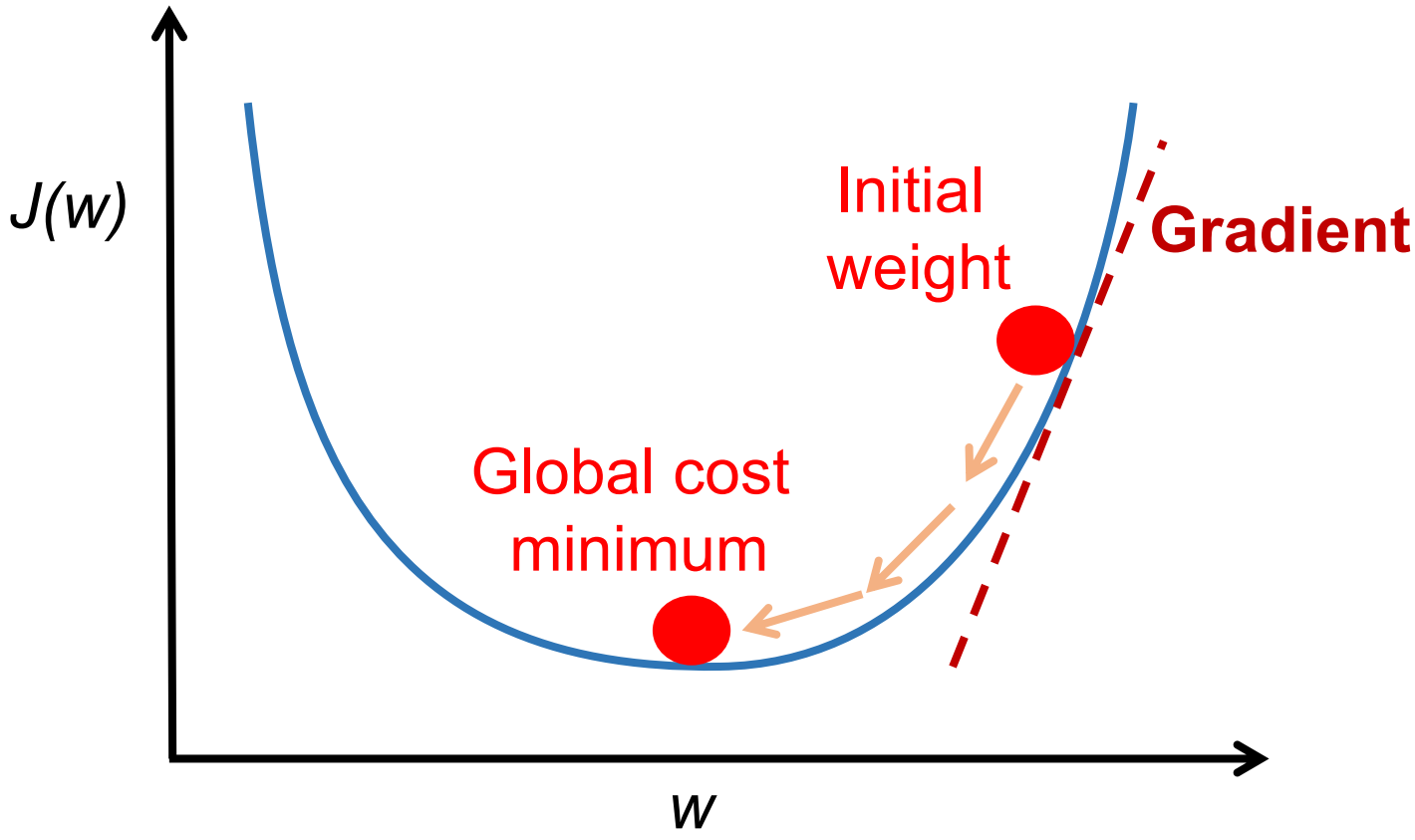
$$y = max(0, -0.23 * x_1 + 0.31 * x_2 + 0.65 * x_3)$$

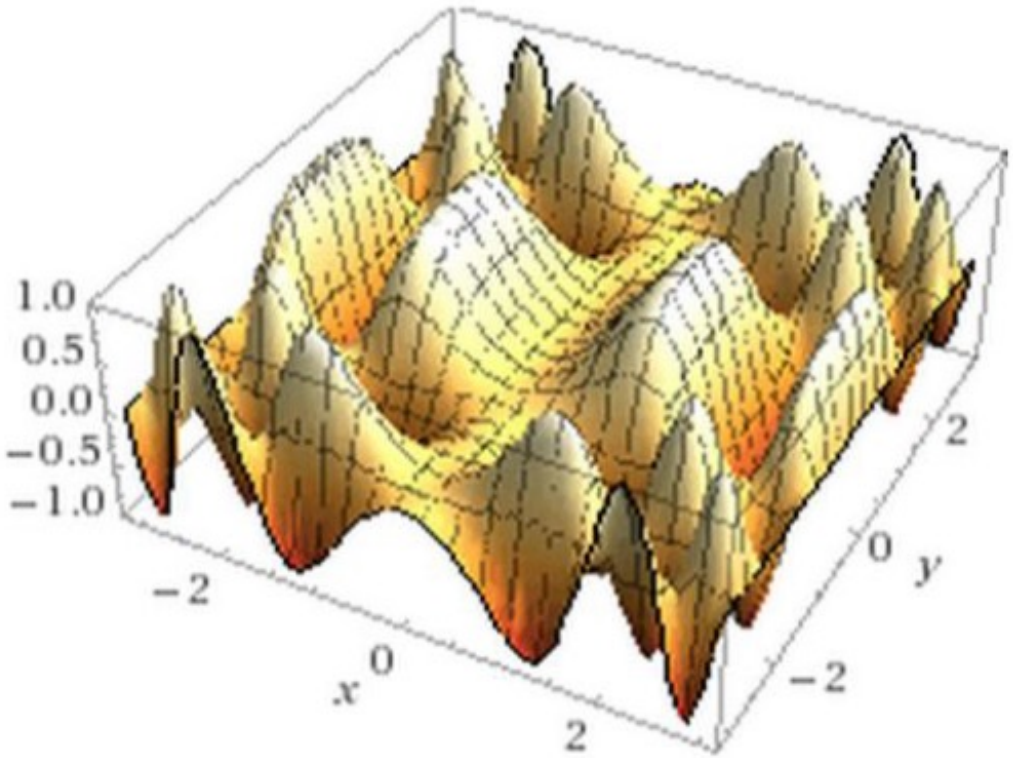
~~$$y = max(0, -0.21 * x_1 + 0.3 * x_2 + 0.7 * x_3)$$~~



Optimizer:

Stochastic Gradient Descent (SGD)

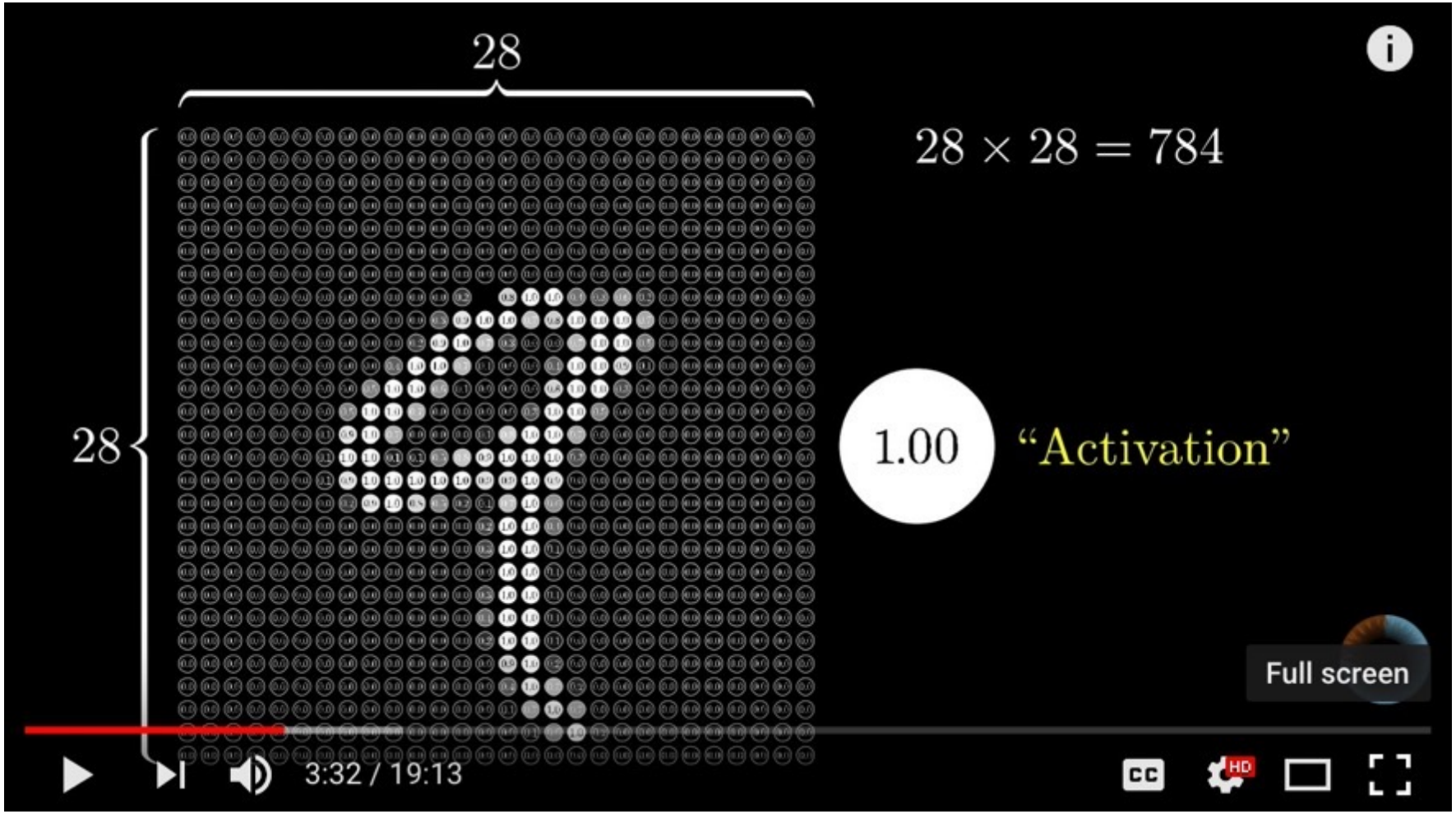




This shows a function of 2 variables: real neural nets are functions of hundreds of millions of variables!

Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

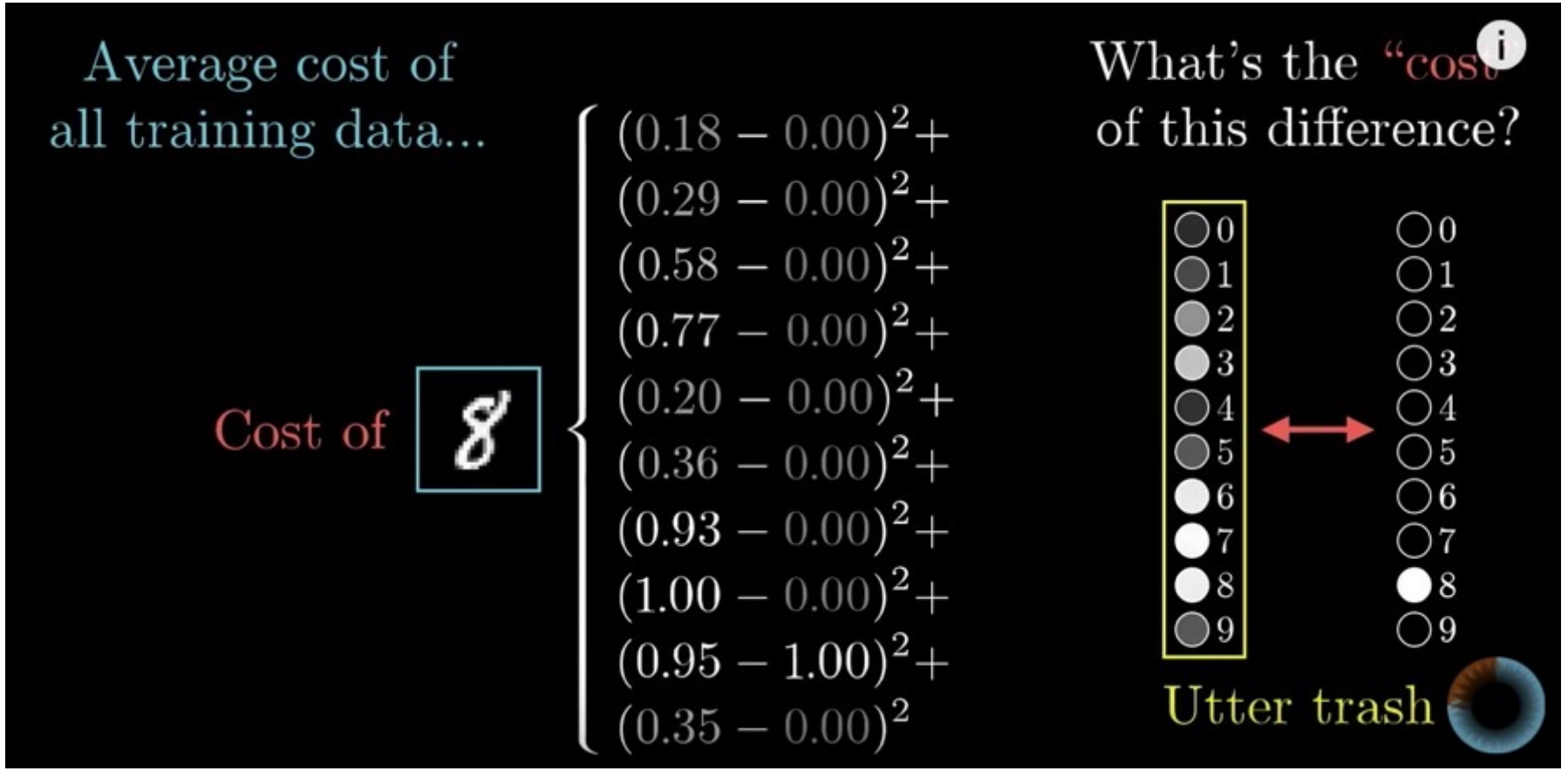
Neural Network and Deep Learning



Source: 3Blue1Brown (2017), But what *is* a Neural Network? | Chapter 1, deep learning, https://www.youtube.com/watch?v=aircAruvnKk

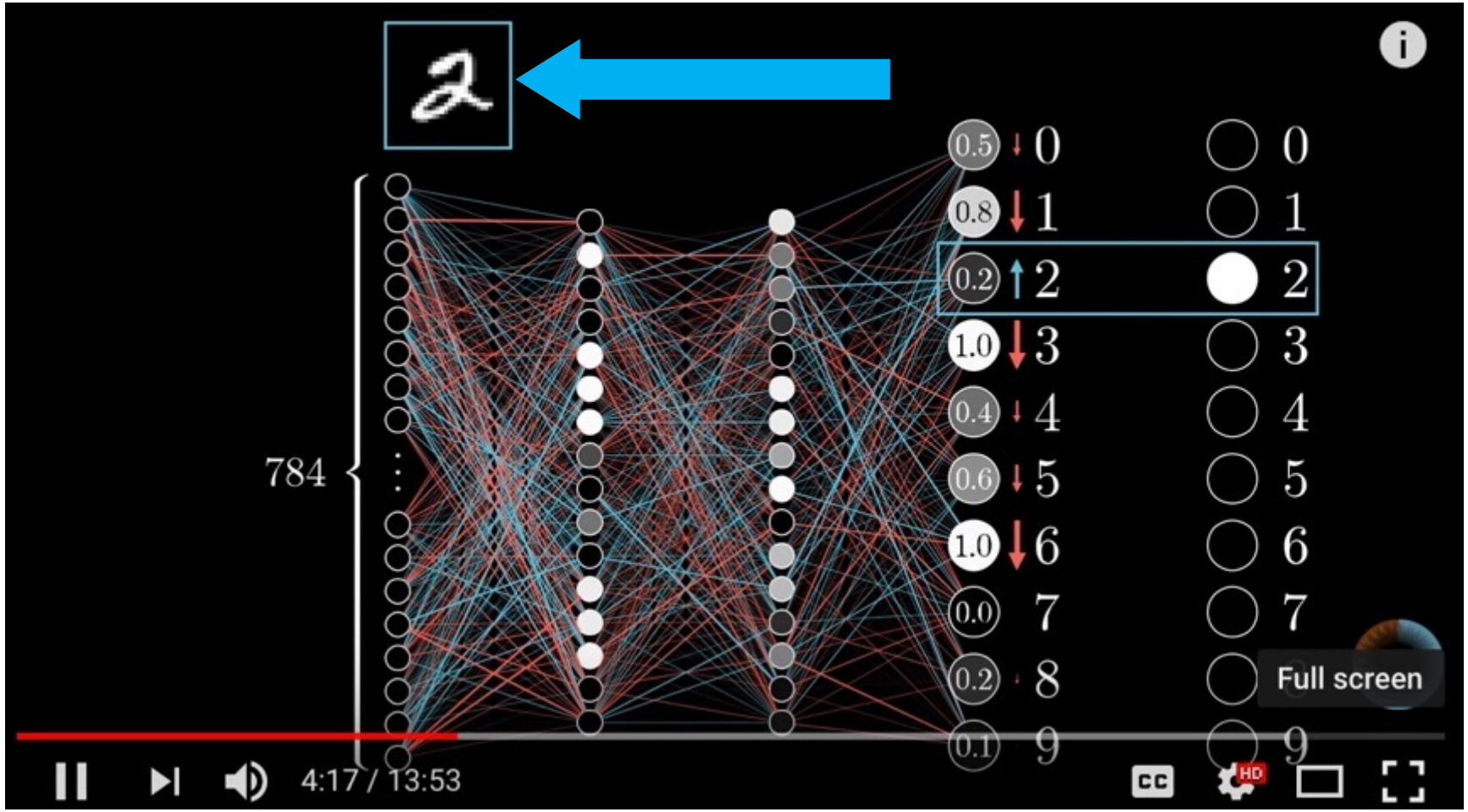
Gradient Descent

how neural networks learn



Source: 3Blue1Brown (2017), Gradient descent, how neural networks learn | Chapter 2, deep learning, https://www.youtube.com/watch?v=IHZwWFHWa-w

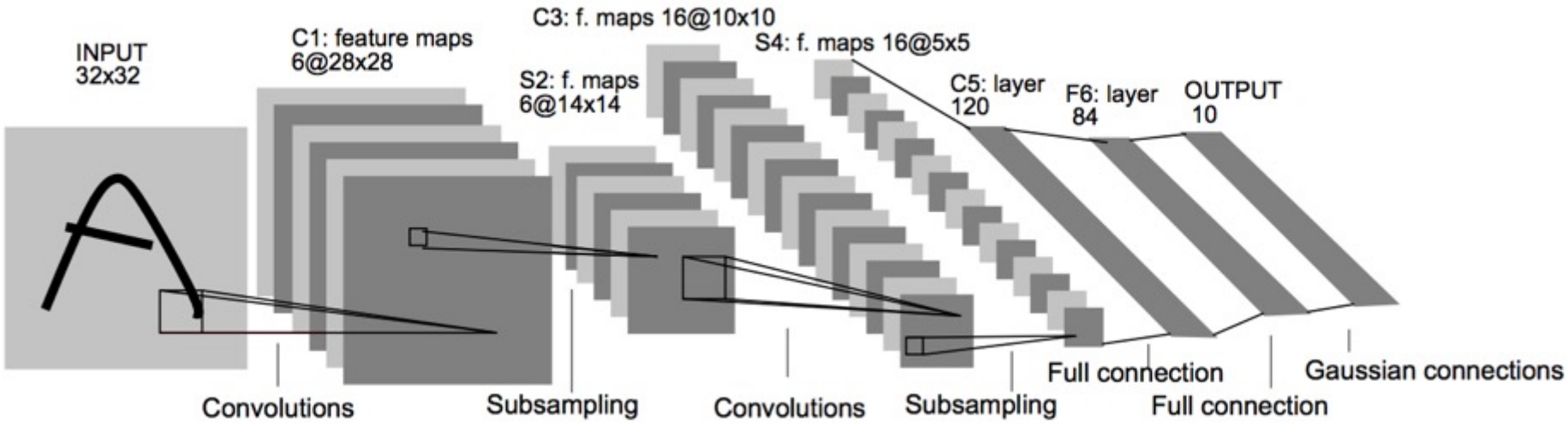
Backpropagation



Source: 3Blue1Brown (2017), What is backpropagation really doing? | Chapter 3, deep learning, https://www.youtube.com/watch?v=Ilg3gGewQ5U

Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN)



Architecture of LeNet-5 (7 Layers)
(LeCun et al., 1998)

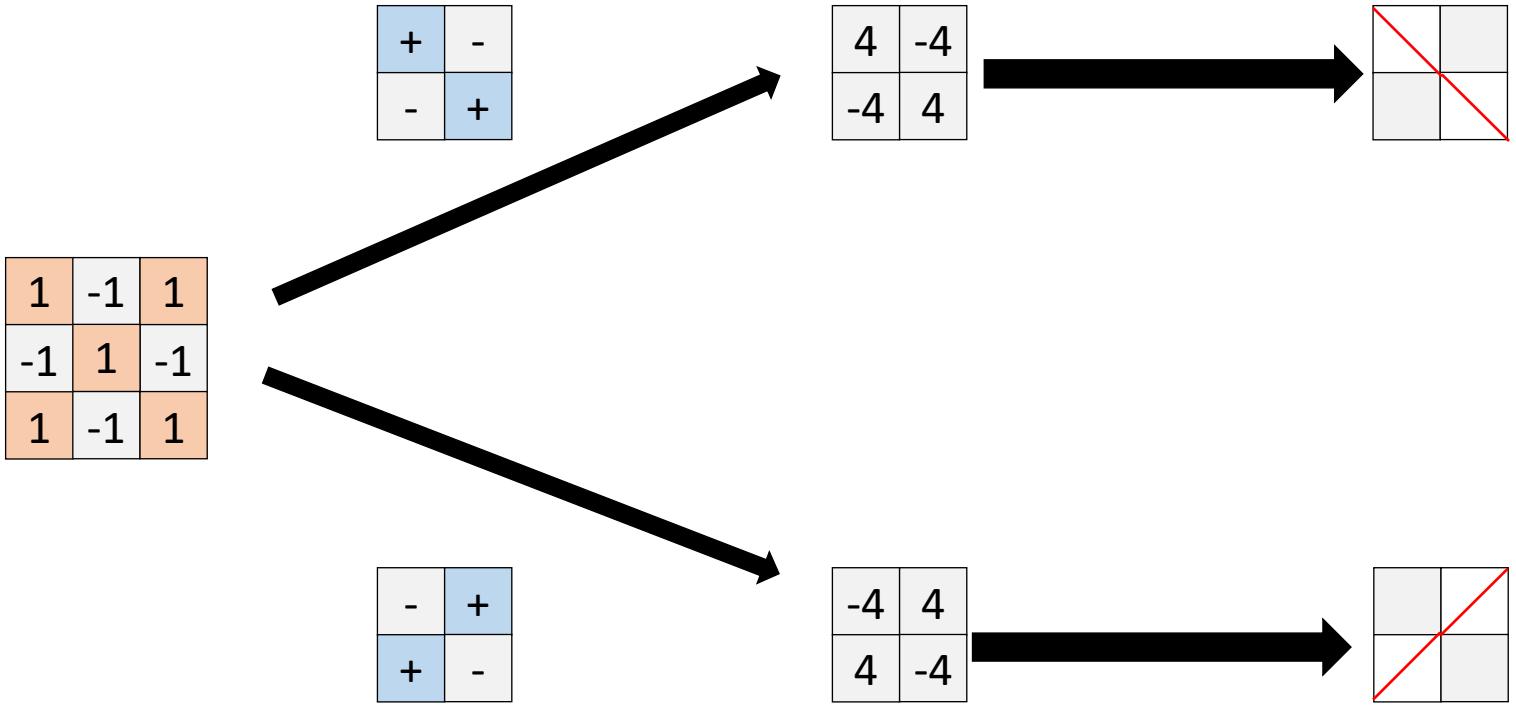
Source: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

Source: LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner.
"Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86, no. 11 (1998): 2278-2324.

Convolutional Neural Networks (CNN)

- **Convolution**
- **Pooling**
- **Fully Connection (FC) (Flattening)**

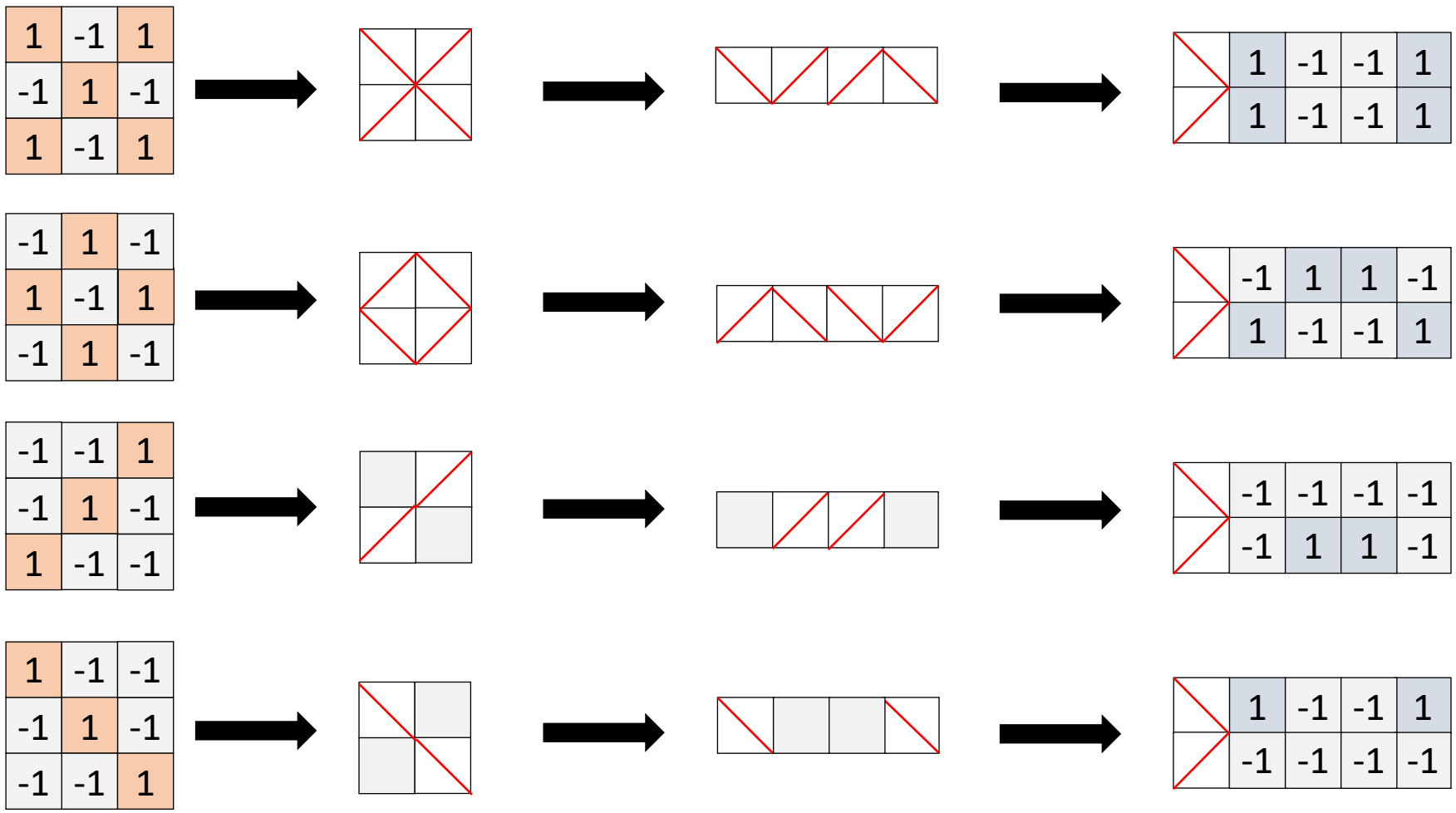
A friendly introduction to Convolutional Neural Networks and Image Recognition



Convolution Layer Pooling Layer

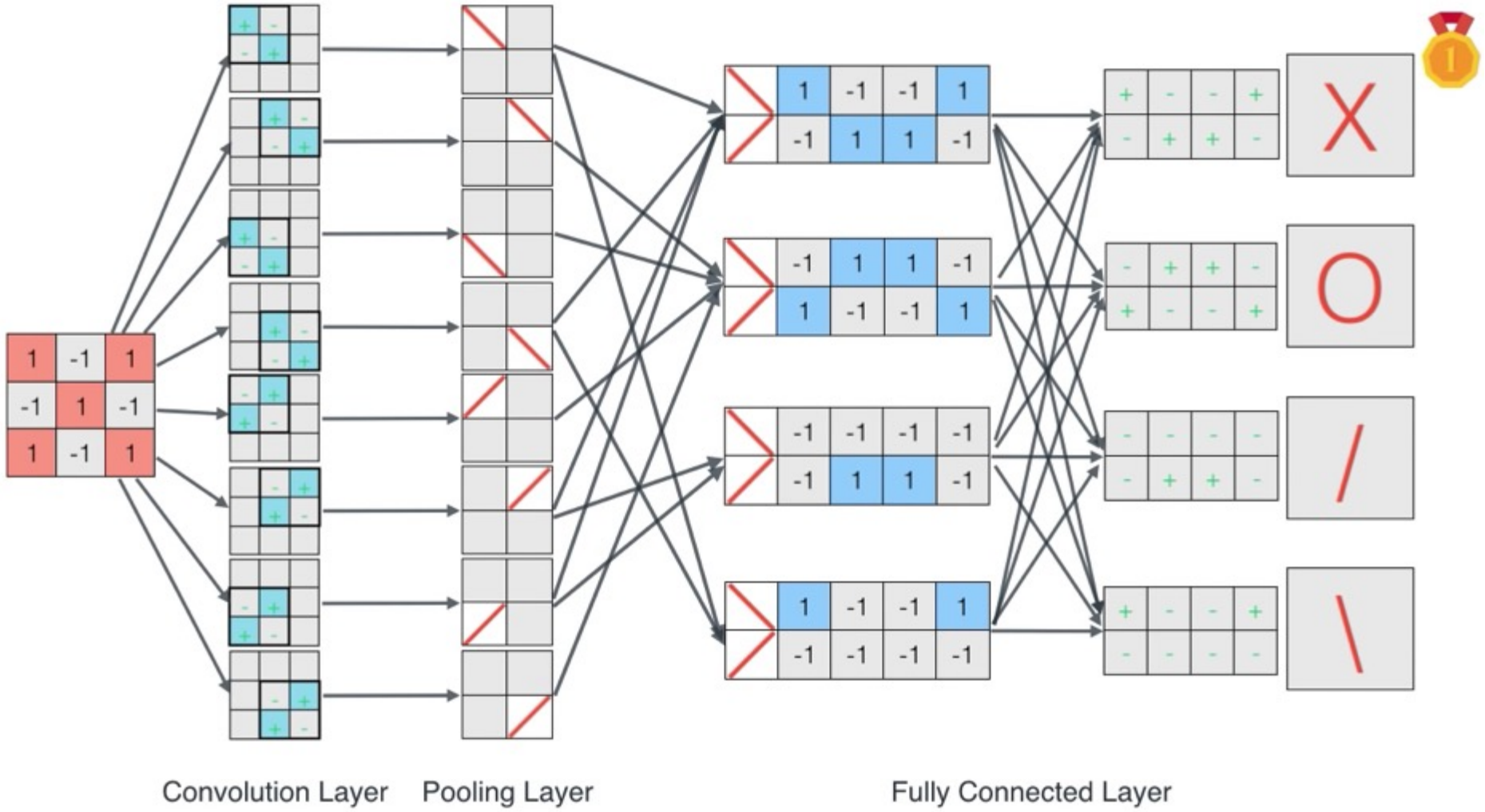
Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

A friendly introduction to Convolutional Neural Networks and Image Recognition



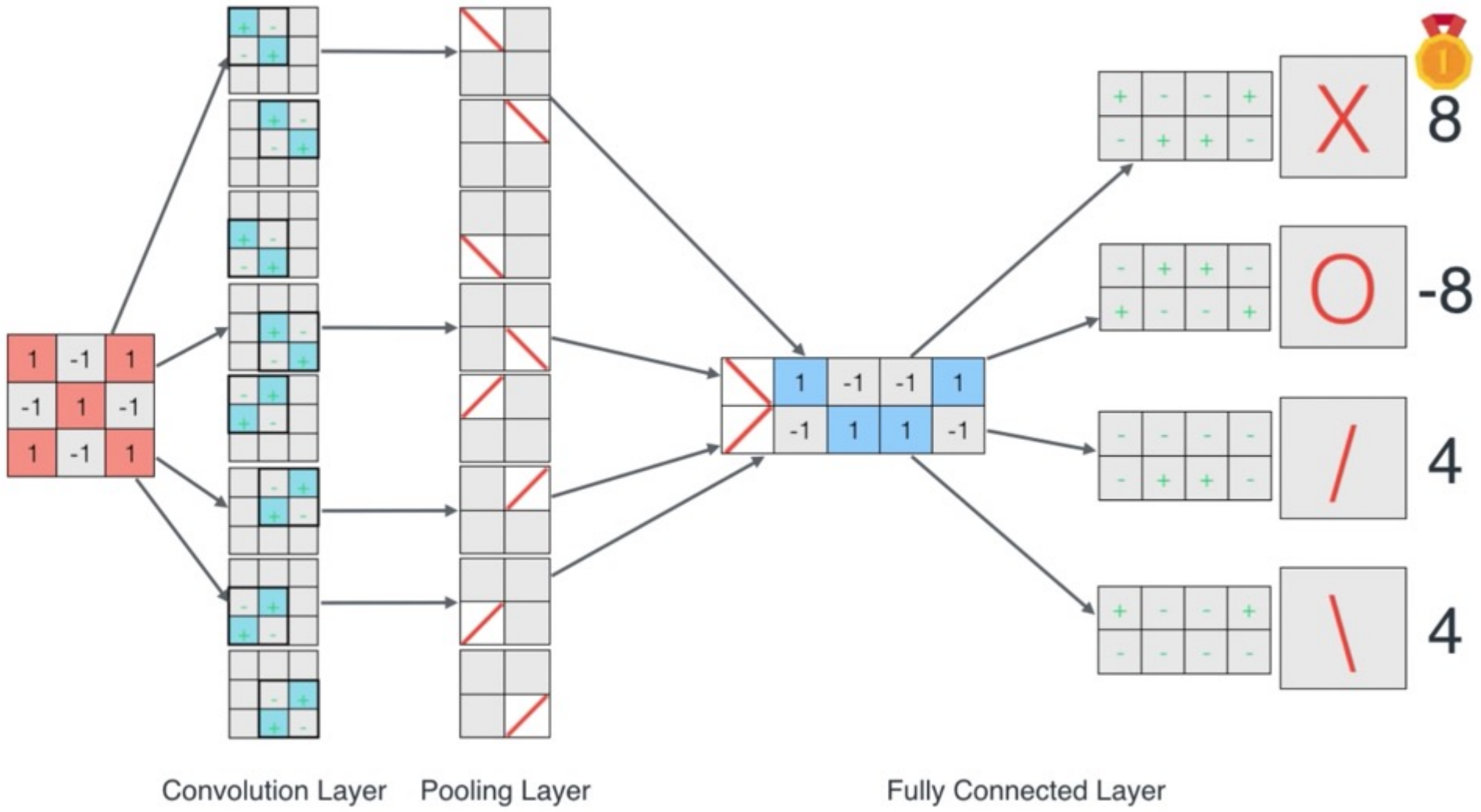
Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

A friendly introduction to Convolutional Neural Networks and Image Recognition



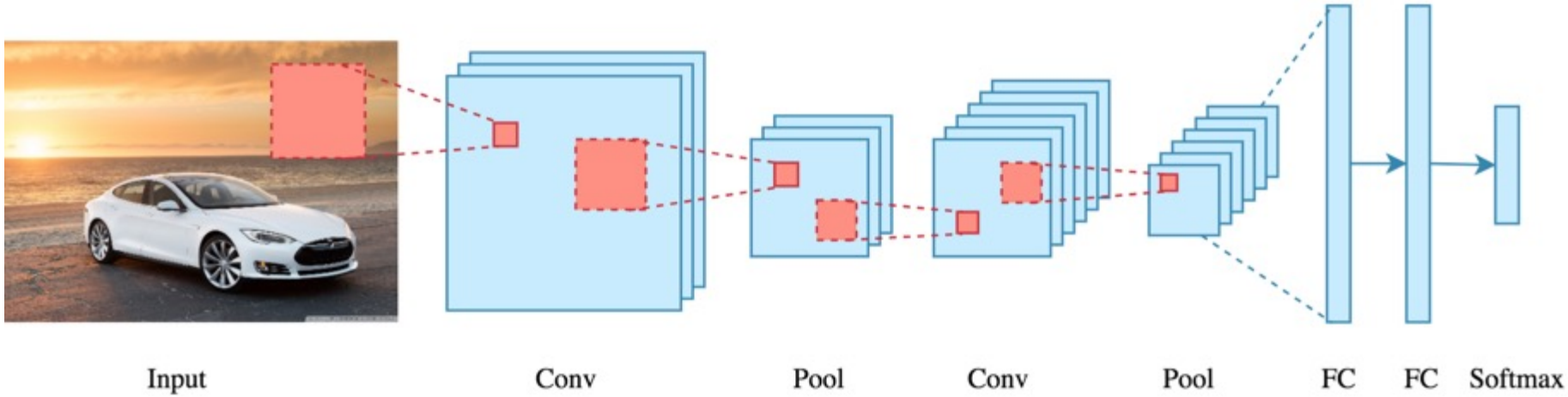
Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

A friendly introduction to Convolutional Neural Networks and Image Recognition



Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

CNN Architecture



Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

Convolution is a **mathematical operation** to **merge two sets** of information

3x3 convolution

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

Filter / Kernel

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer
Input x Filter --> Feature Map

receptive field: 3x3

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

Input x Filter

4		

Feature Map

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer
Input x Filter --> Feature Map

receptive field: 3x3

1	1x1	1x0	0x1	0
0	1x0	1x1	1x0	0
0	0x1	1x0	1x1	1
0	0	1	1	0
0	1	1	0	0

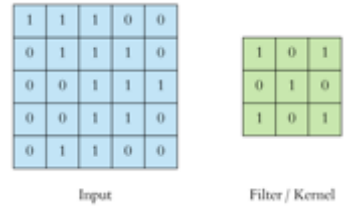
Input x Filter

4	3	

Feature Map

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer



1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

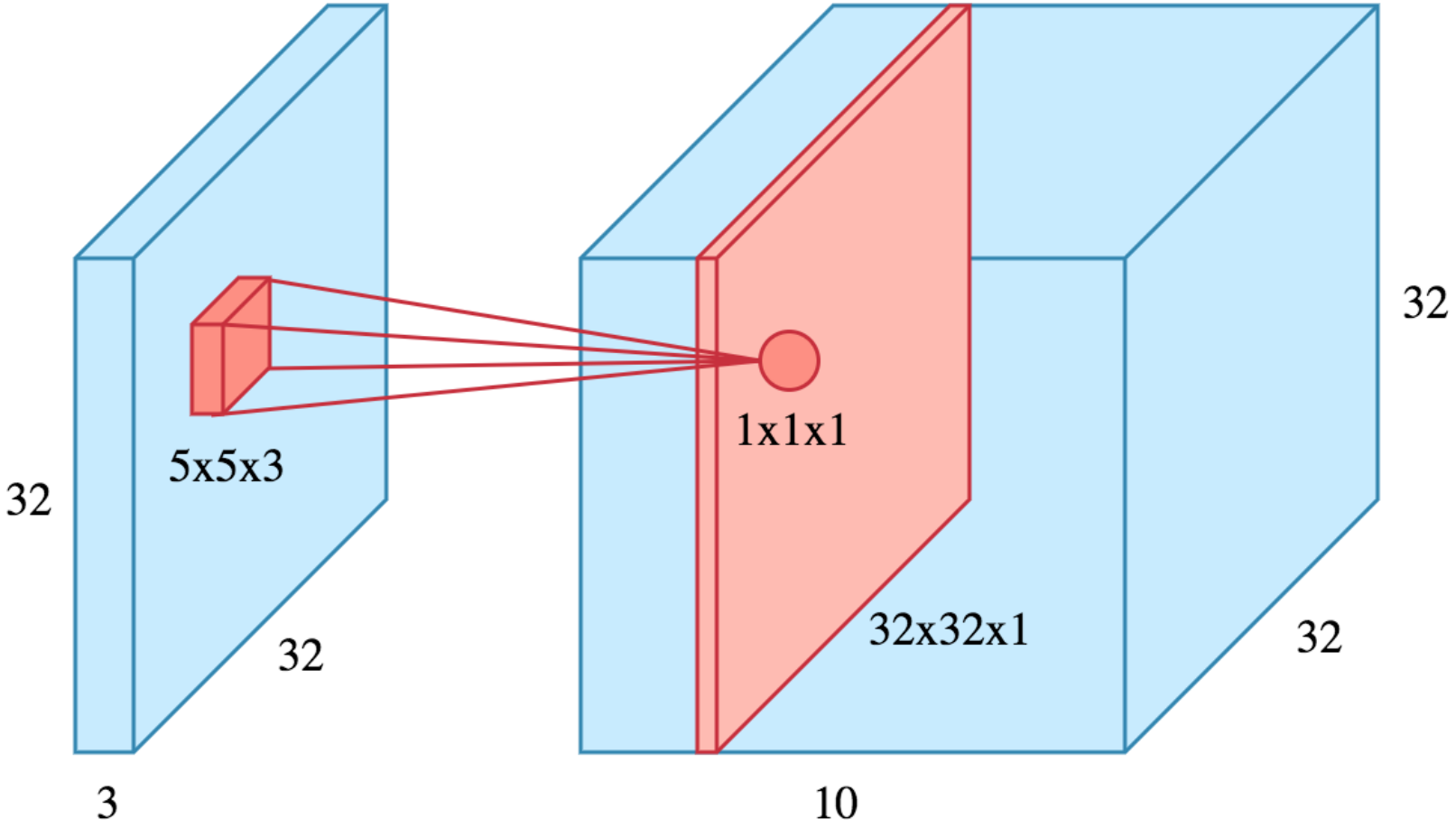
4		

Example convolution operation shown in 2D using a 3x3 filter

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

10 different filters 10 feature maps of size 32x32x1

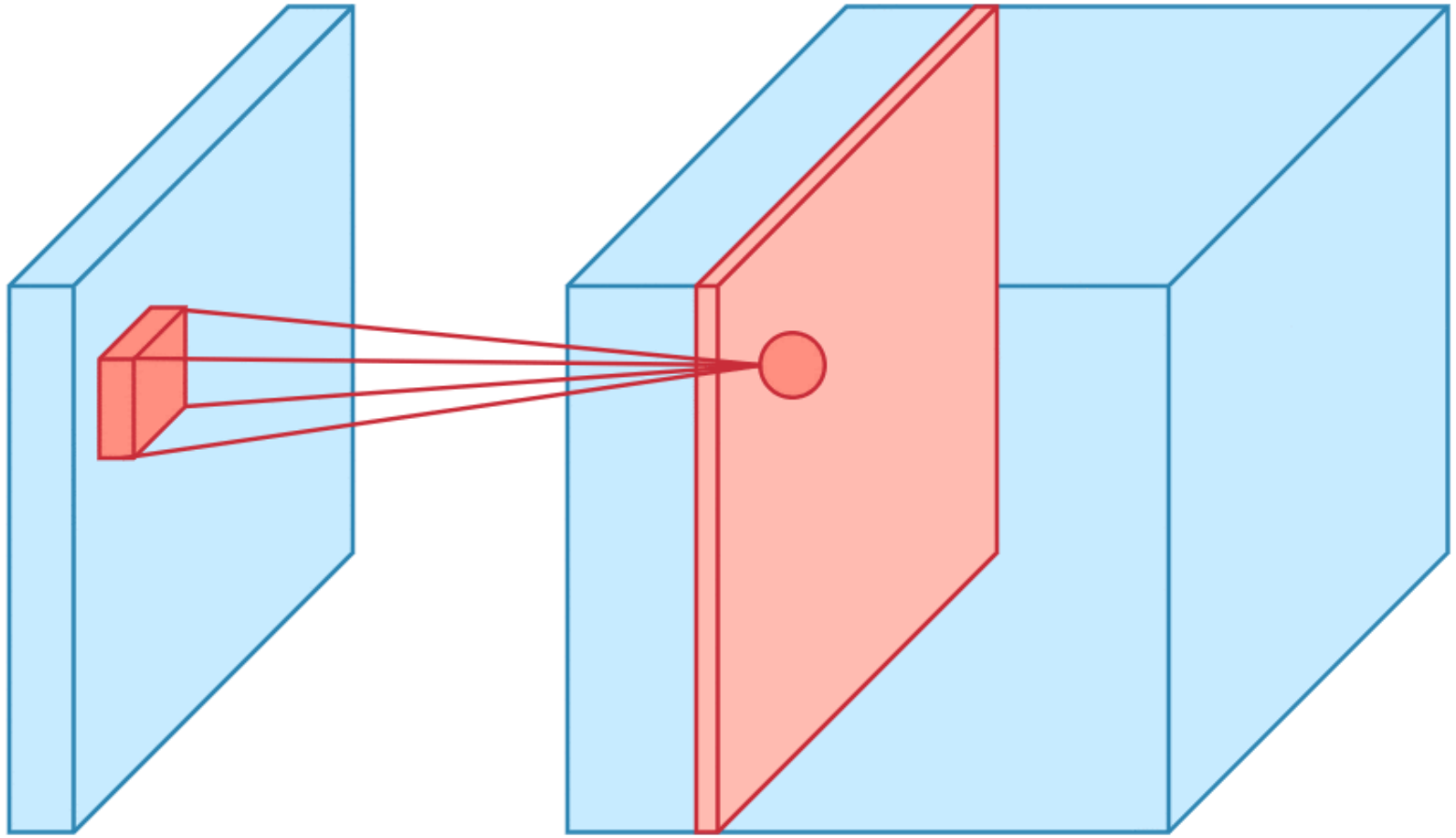


final output of the convolution layer:
a volume of size 32x32x10

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

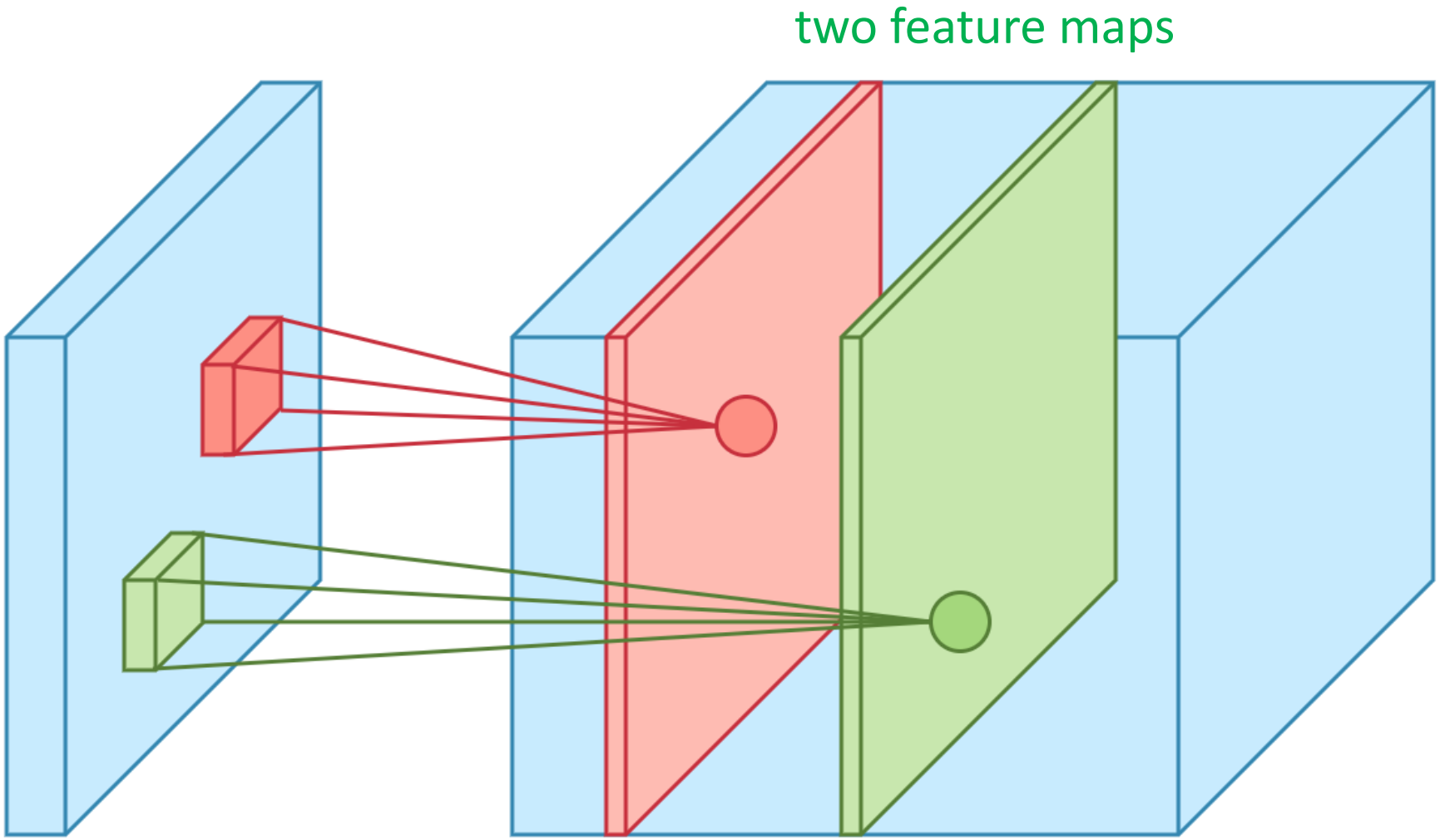
CNN Convolution Layer

Sliding operation at 4 locations



Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

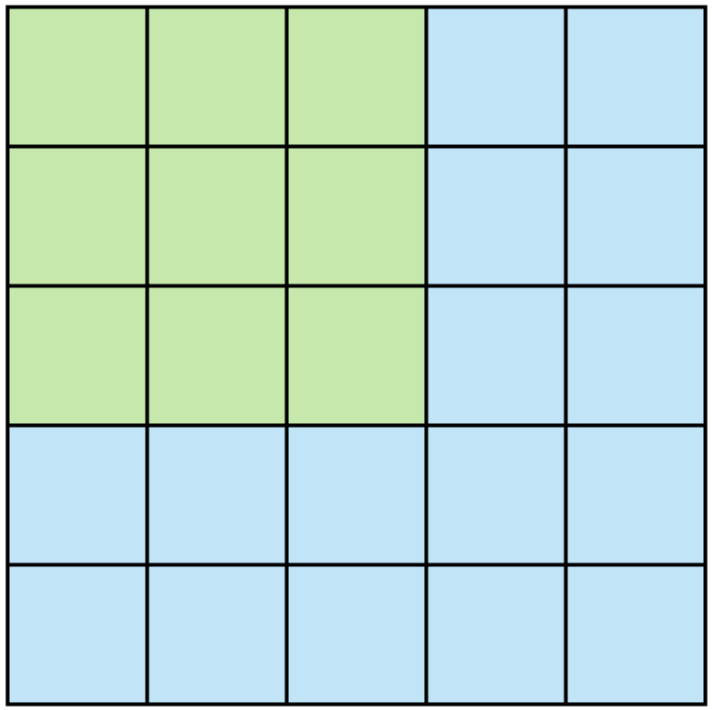
CNN Convolution Layer



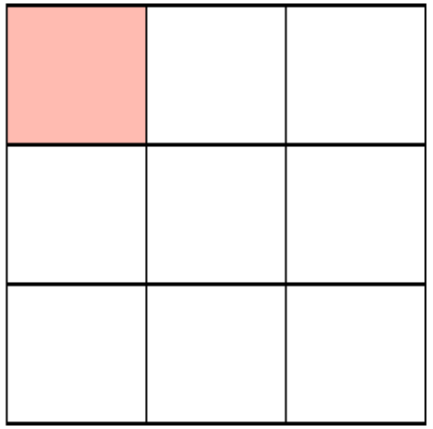
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks,
https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

Stride specifies how much
we move the convolution filter at each step



Stride 1

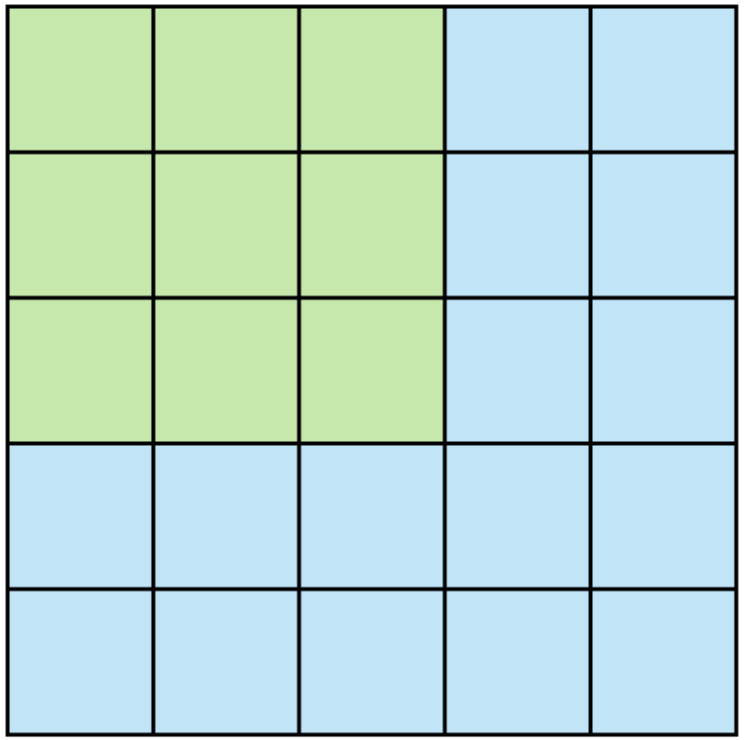


Feature Map

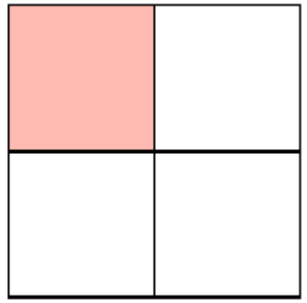
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks,
https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

Stride specifies how much
we move the convolution filter at each step



Stride 2

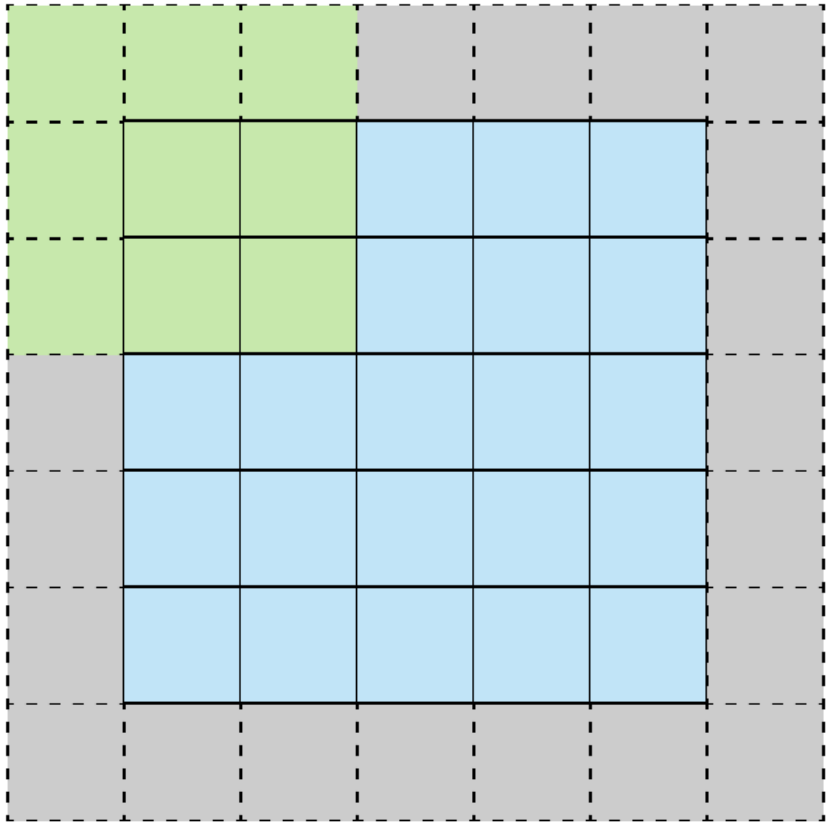


Feature Map

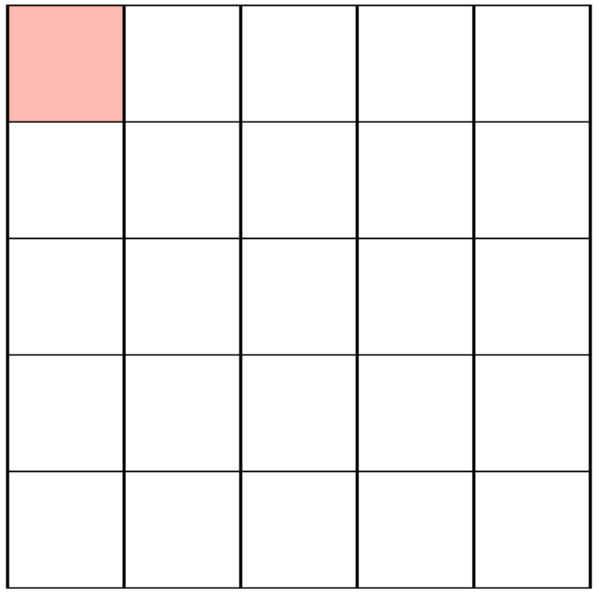
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks,
https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

Stride 1 with Padding



Stride 1 with Padding

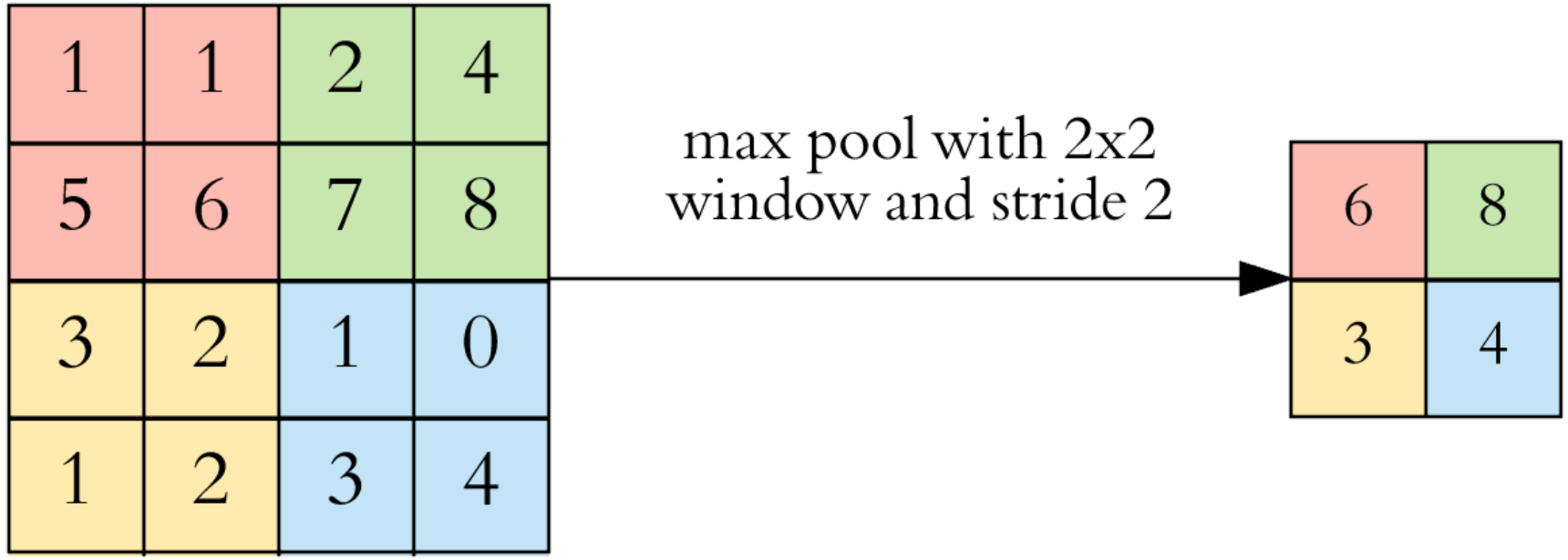


Feature Map

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

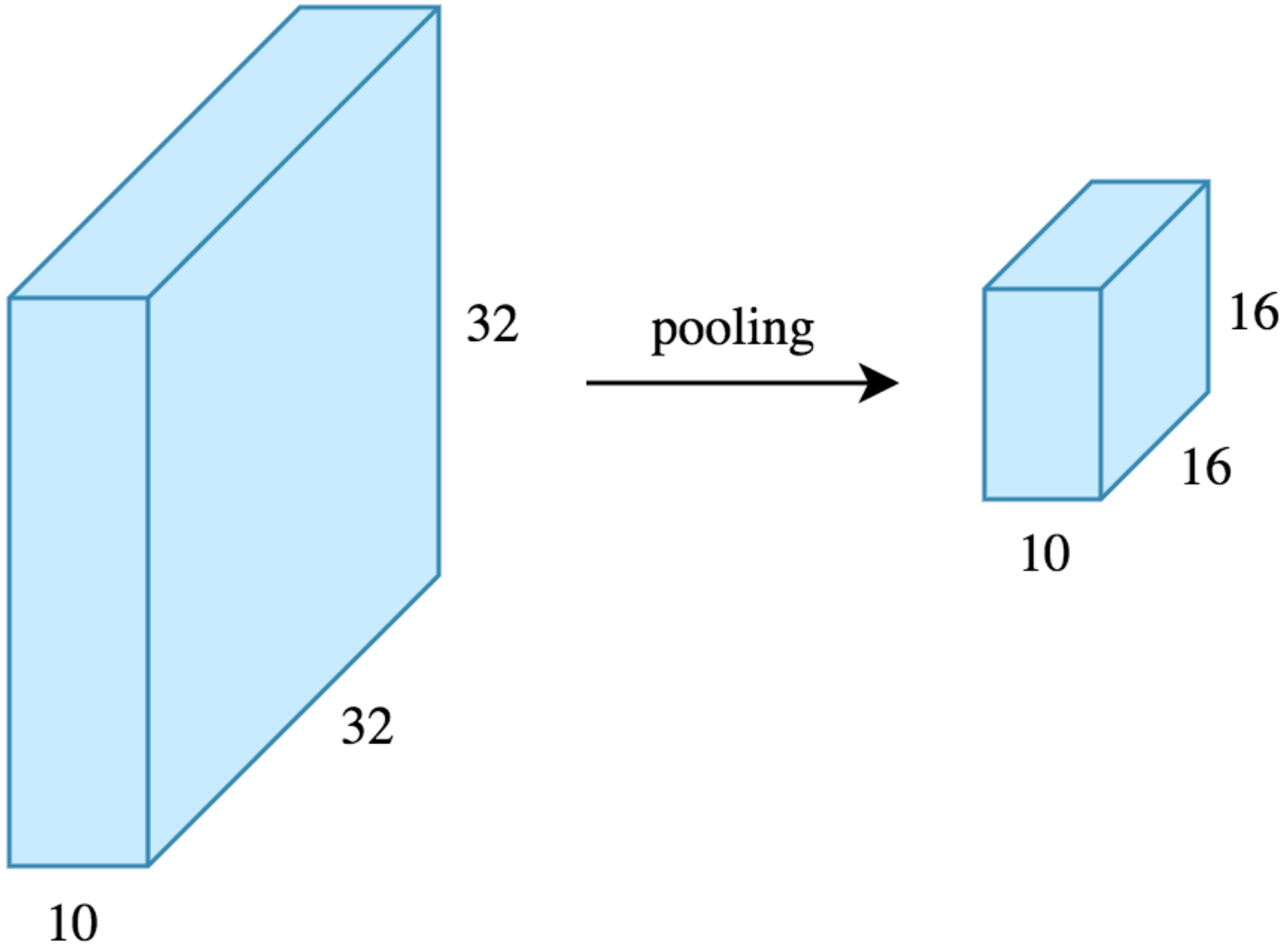
CNN Pooling Layer

Max Pooling



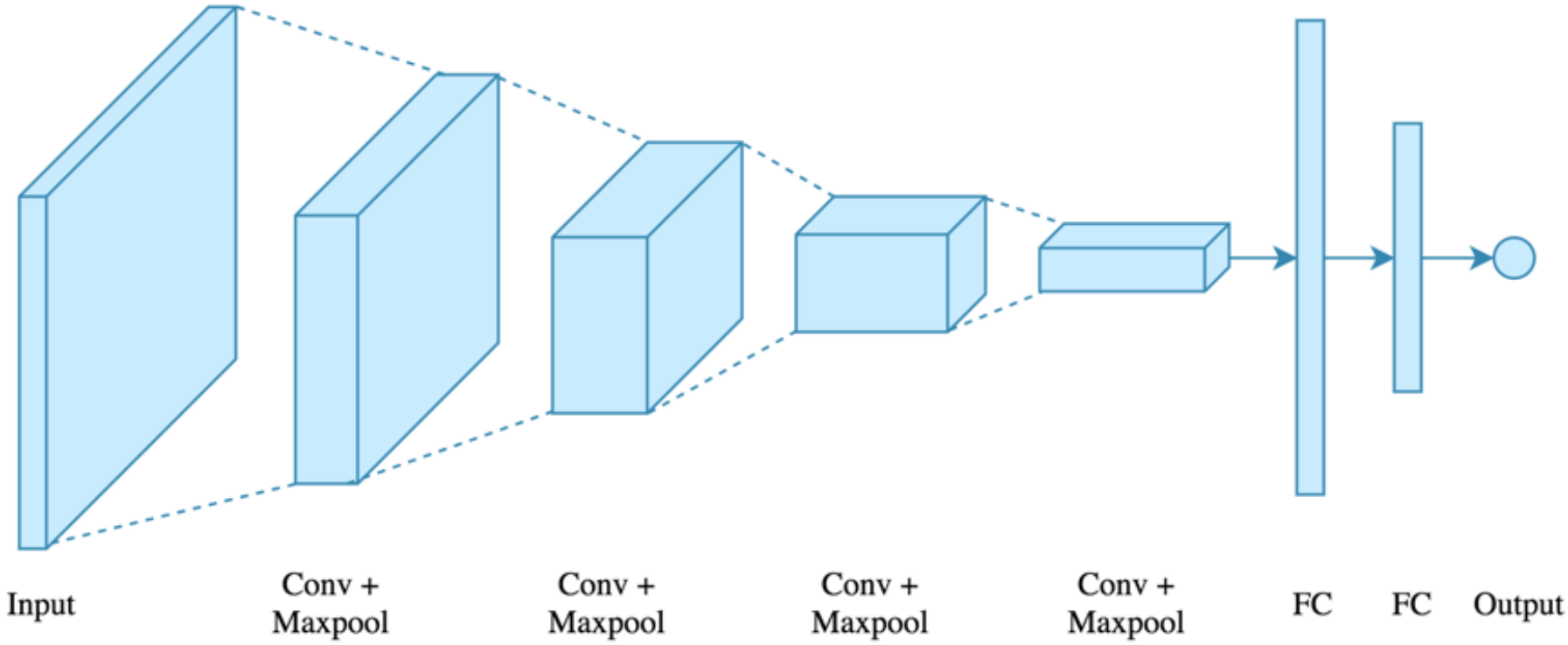
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Pooling Layer



Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Architecture
4 convolution + pooling layers, followed by 2 fully connected layers



Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

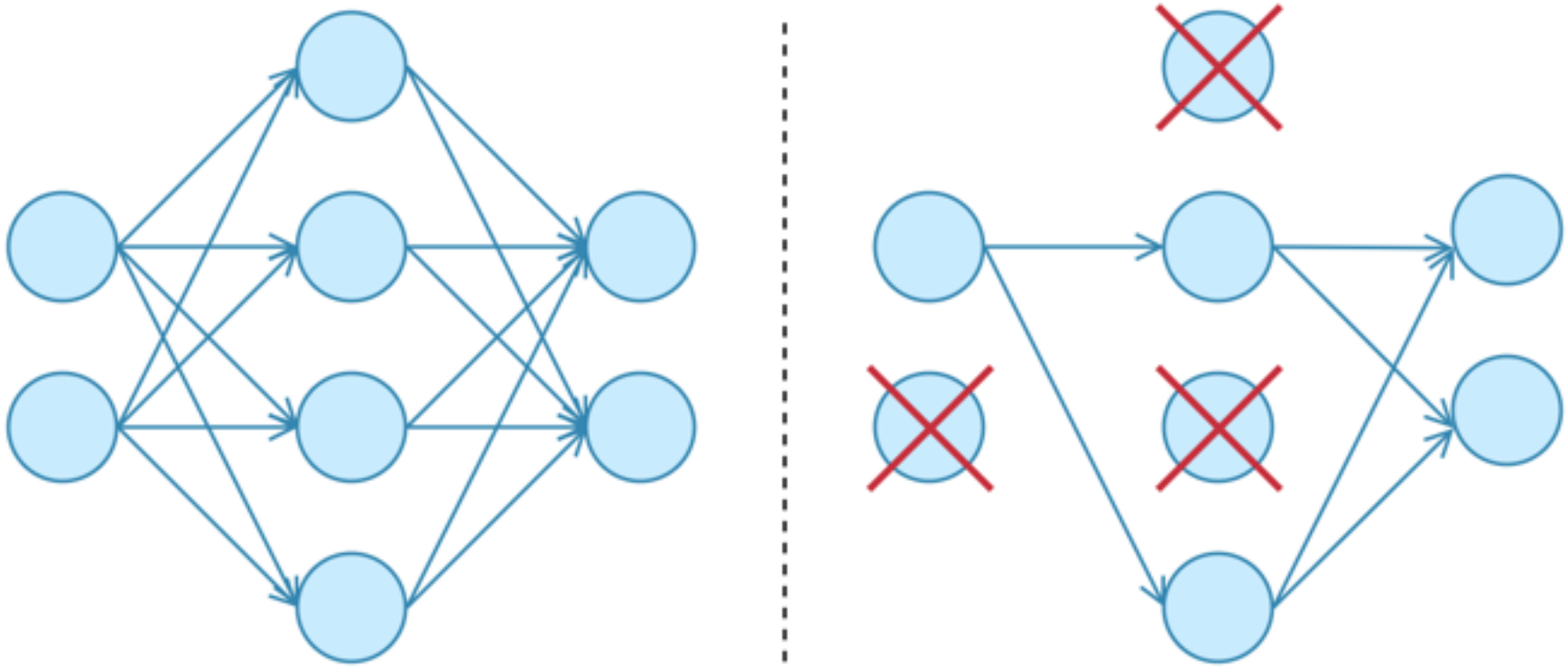
CNN Architecture
4 convolution + pooling layers, followed by 2 fully connected layers

https://gist.github.com/ardendertat/0fc5515057c47e7386fe04e9334504e3

```
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', padding='same', name='conv_1',
                 input_shape=(150, 150, 3)))
model.add(MaxPooling2D((2, 2), name='maxpool_1'))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same', name='conv_2'))
model.add(MaxPooling2D((2, 2), name='maxpool_2'))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same', name='conv_3'))
model.add(MaxPooling2D((2, 2), name='maxpool_3'))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same', name='conv_4'))
model.add(MaxPooling2D((2, 2), name='maxpool_4'))
model.add(Flatten())
model.add(Dropout(0.5))
model.add(Dense(512, activation='relu', name='dense_1'))
model.add(Dense(128, activation='relu', name='dense_2'))
model.add(Dense(1, activation='sigmoid', name='output'))
```

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Dropout

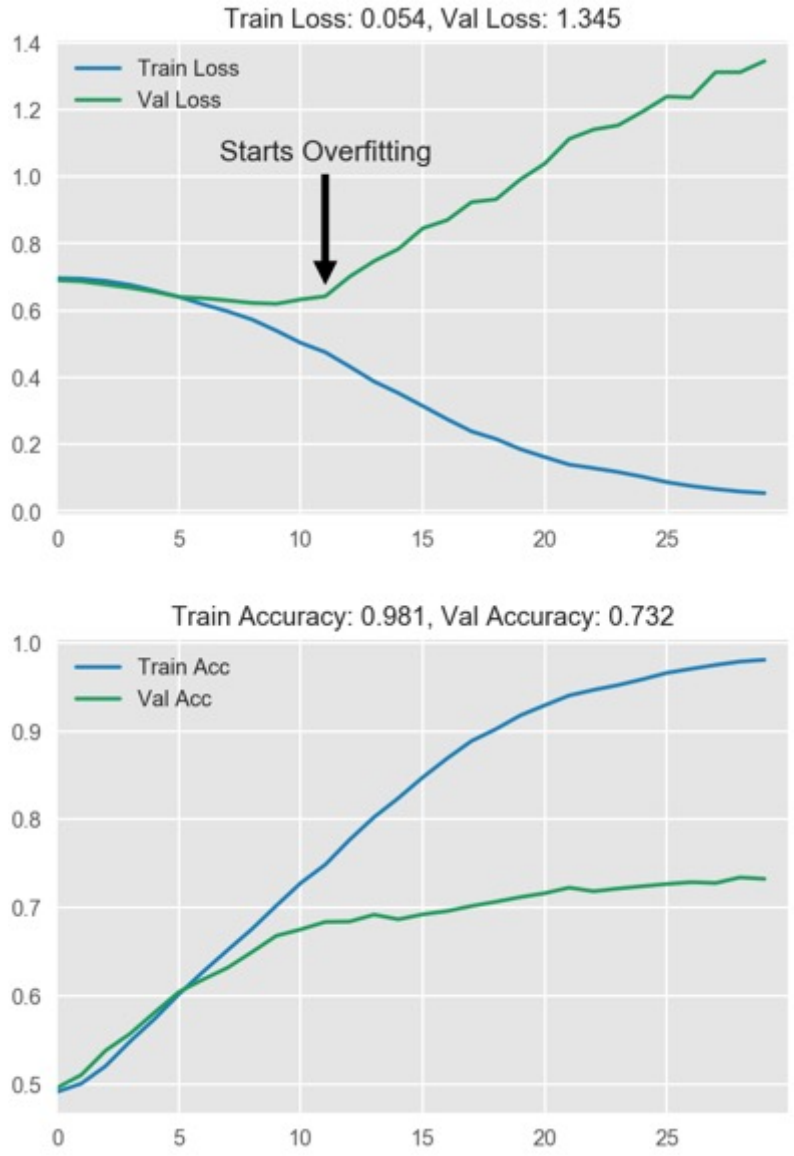


No Dropout

With Dropout

Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

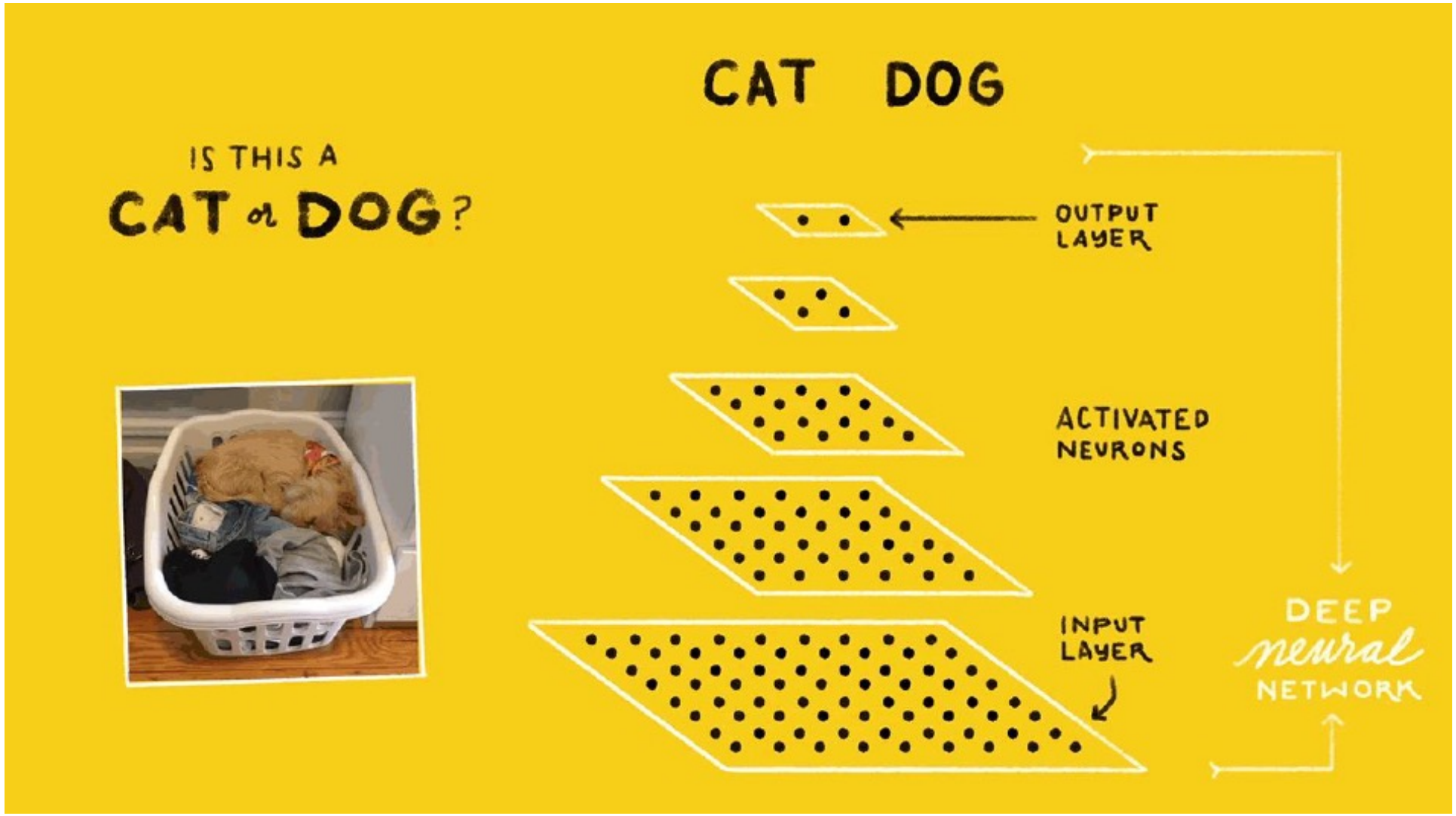
Model Performance



Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Visual Recognition

Image Classification

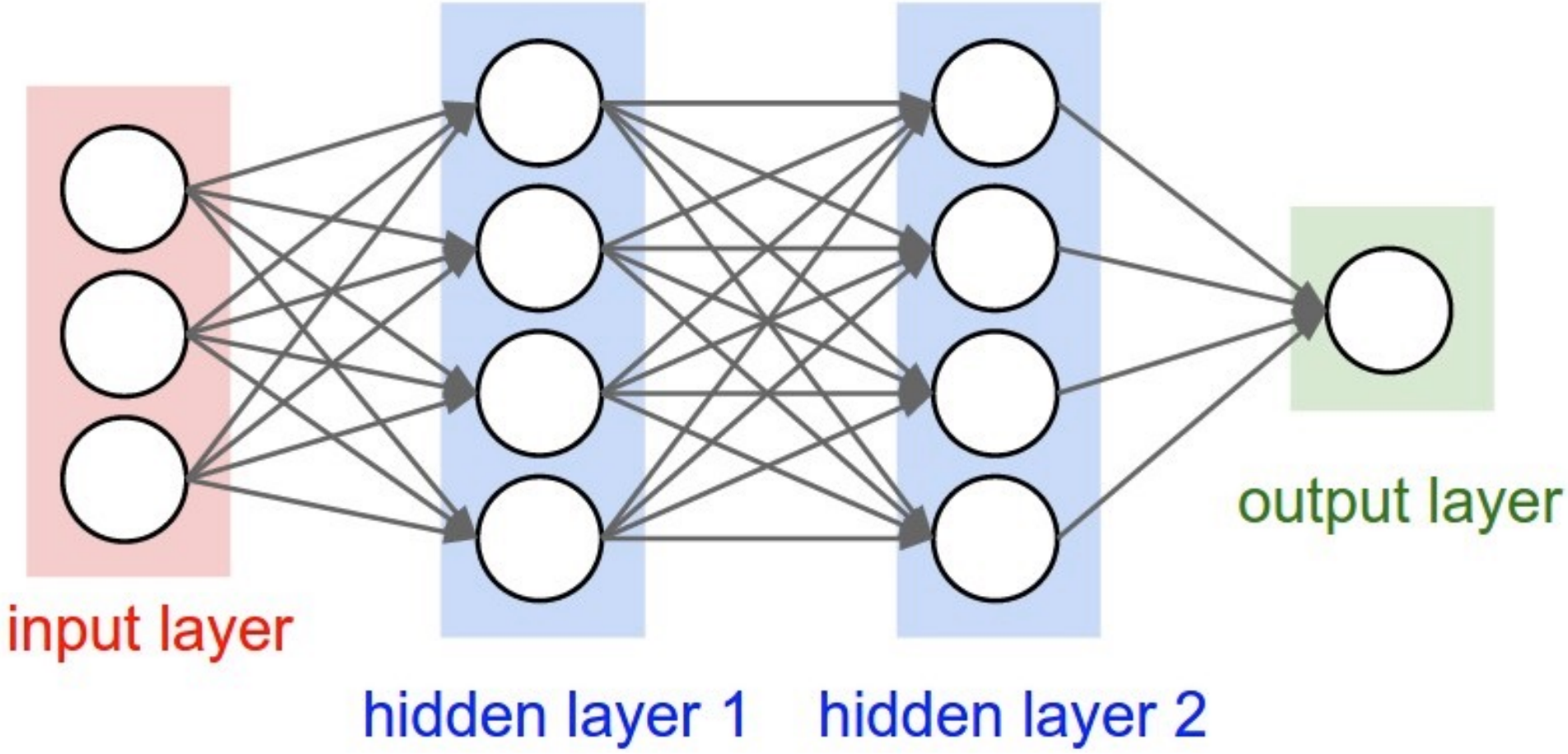


Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

Convolutional Neural Networks
(CNNs / ConvNets)

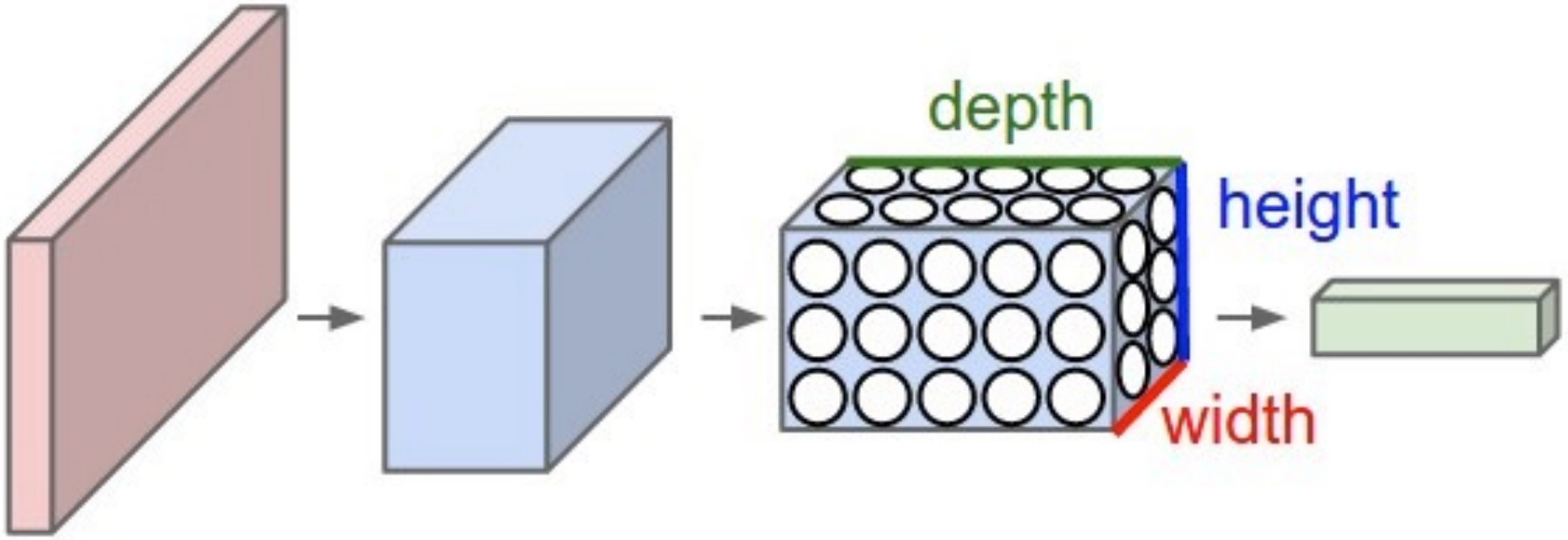
http://cs231n.github.io/convolutional-networks/

A regular 3-layer Neural Network



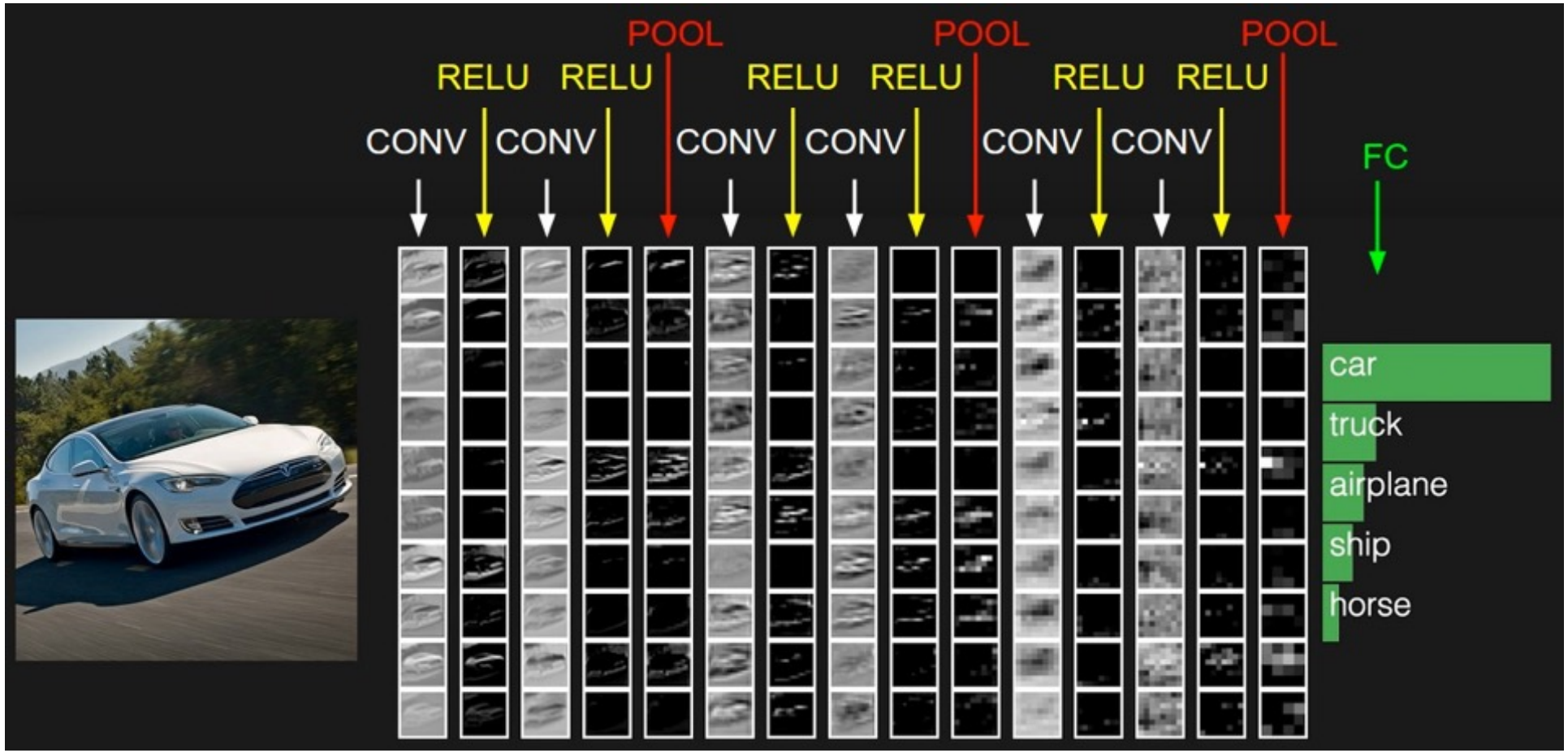
http://cs231n.github.io/convolutional-networks/

A ConvNet arranges its neurons in three dimensions (width, height, depth)



http://cs231n.github.io/convolutional-networks/

The activations of an example ConvNet architecture.

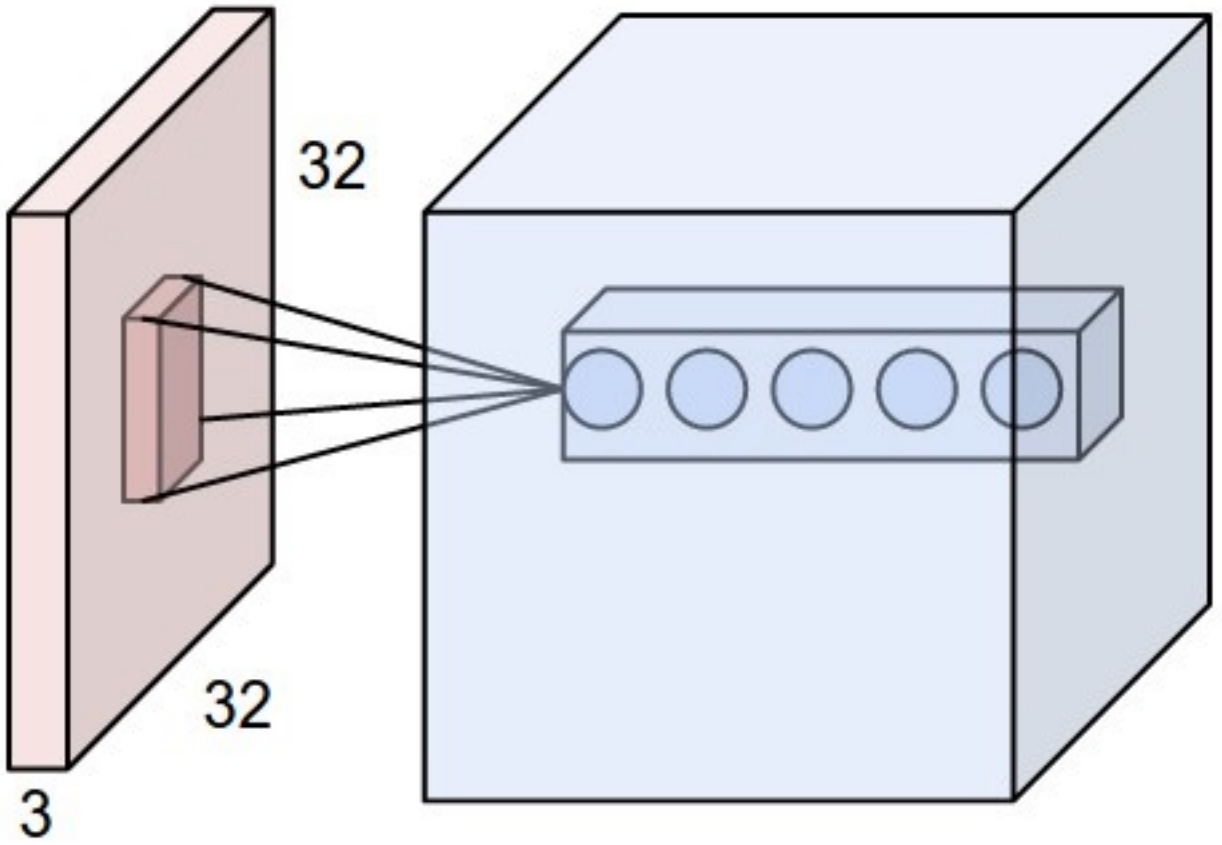


http://cs231n.github.io/convolutional-networks/

ConvNets

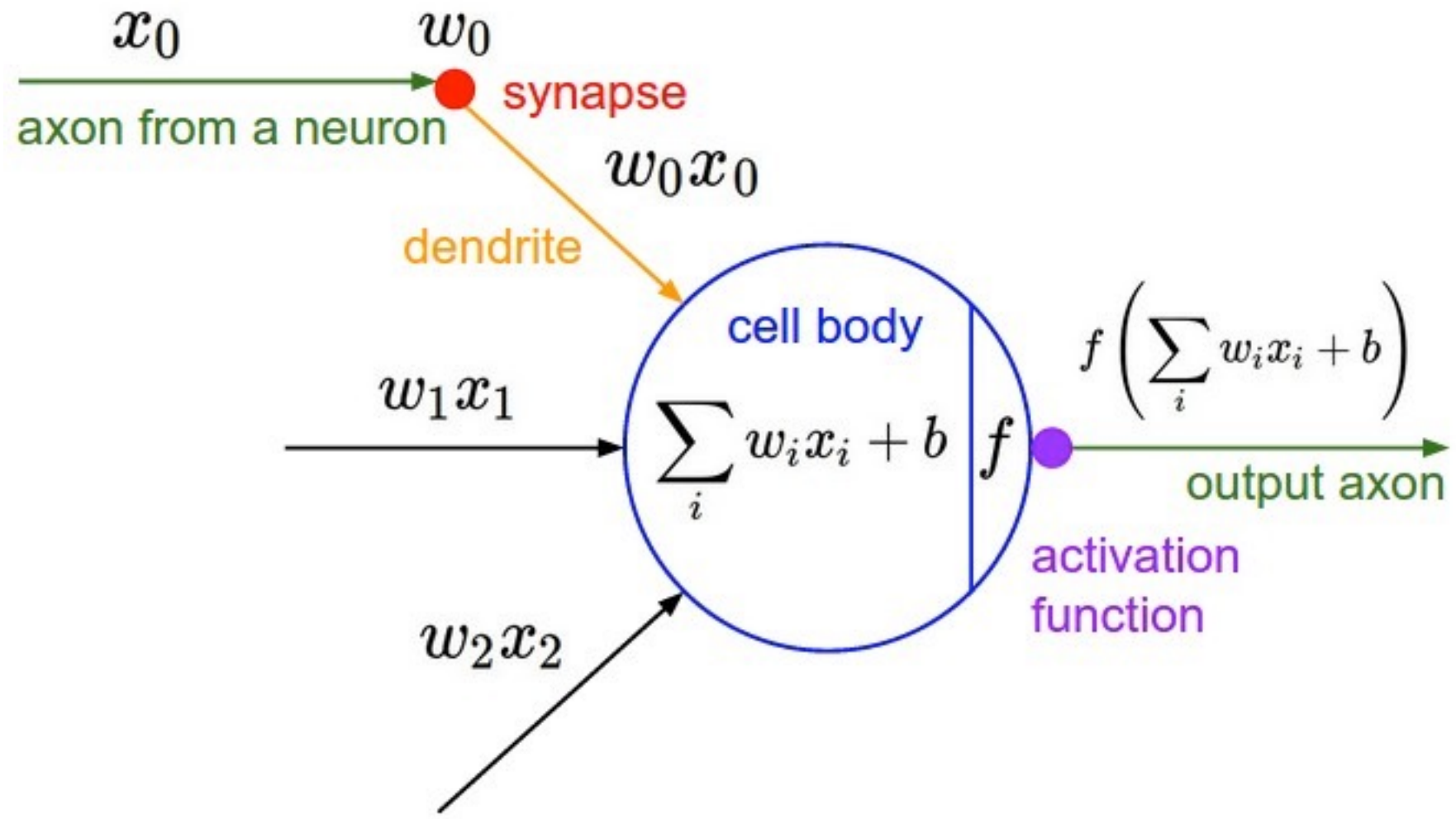
32x32x3 CIFAR-10 image

first Convolutional layer



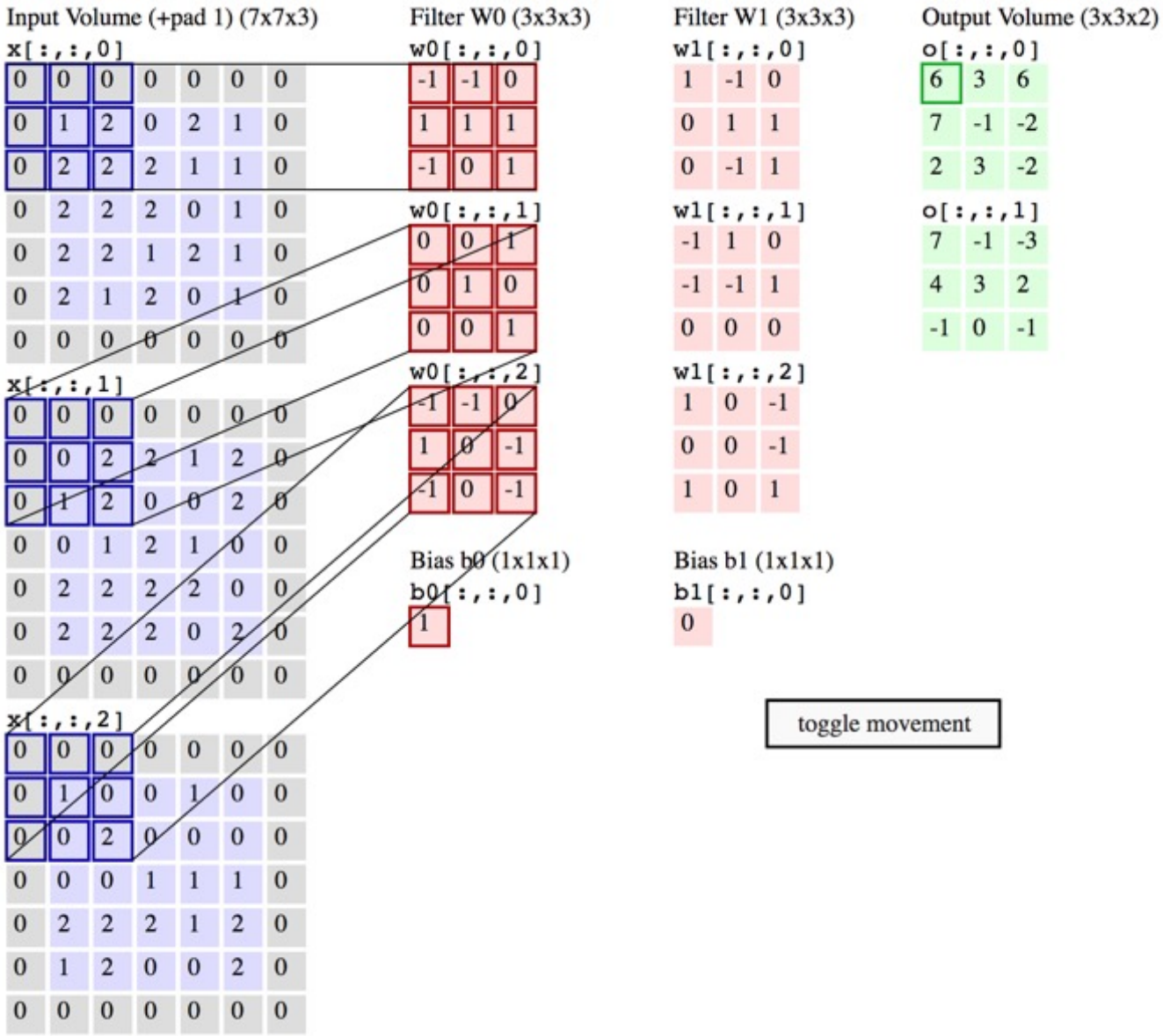
http://cs231n.github.io/convolutional-networks/

ConvNets



http://cs231n.github.io/convolutional-networks/

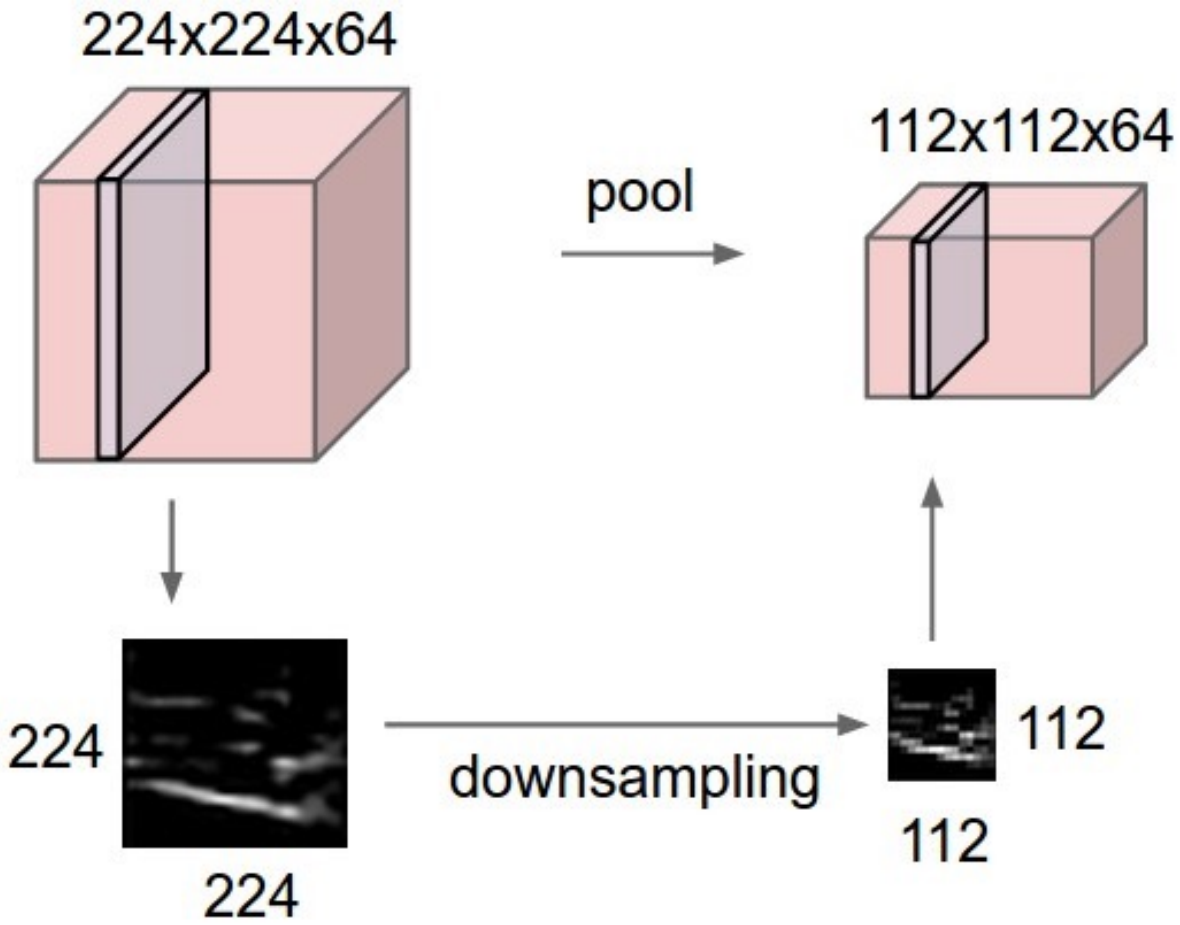
Convolution Demo



http://cs231n.github.io/convolutional-networks/

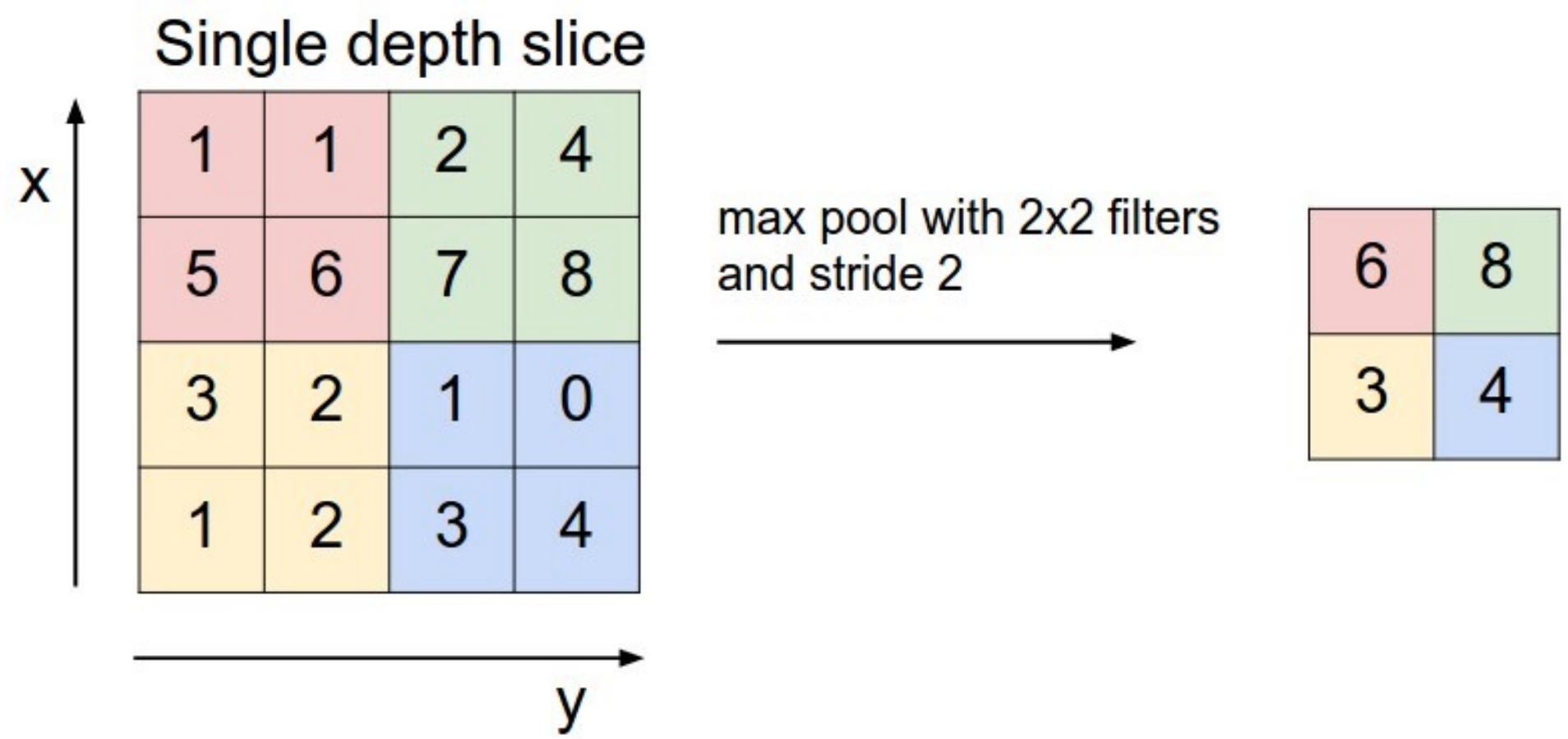
ConvNets

input volume of size [224x224x64]
is pooled with **filter** size 2, **stride** 2
into output volume of size [112x112x64]



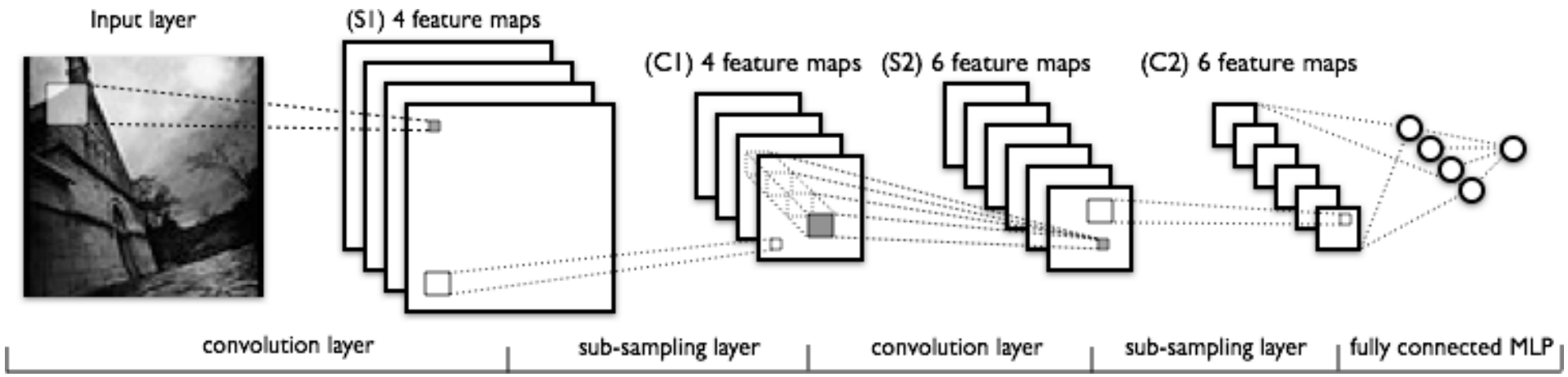
http://cs231n.github.io/convolutional-networks/

ConvNets
max pooling



http://cs231n.github.io/convolutional-networks/

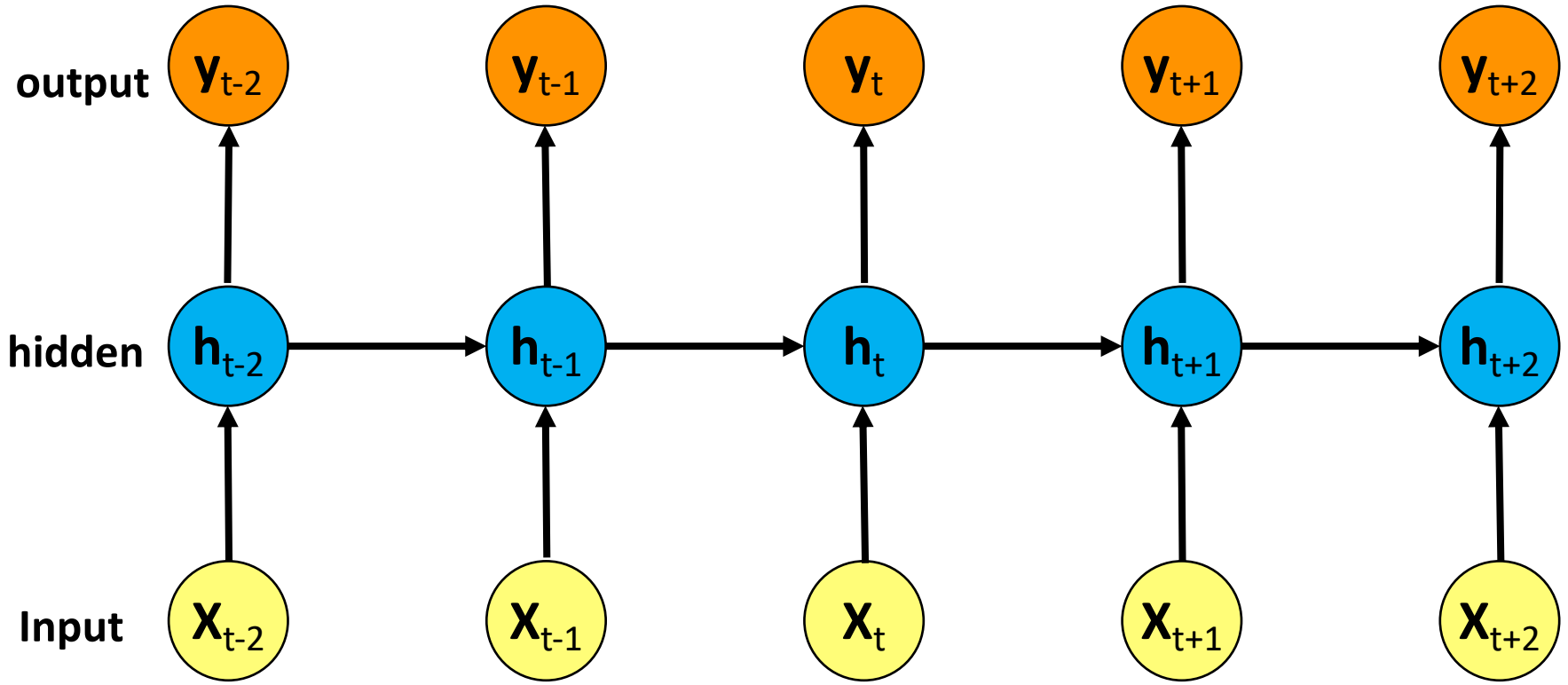
Convolutional Neural Networks (CNN) (LeNet)



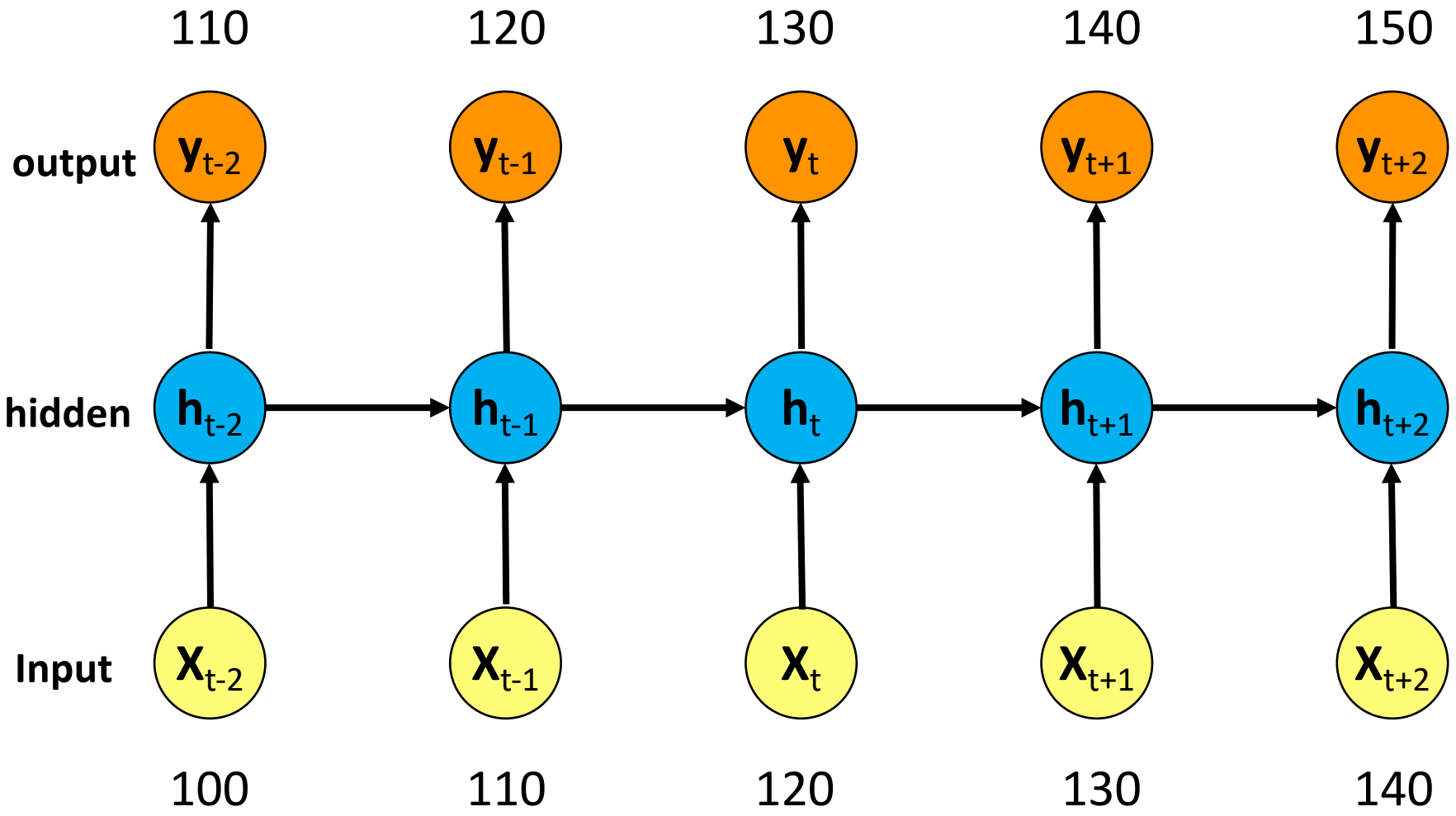
Source: http://deeplearning.net/tutorial/lenet.html

Recurrent Neural Networks (RNN)

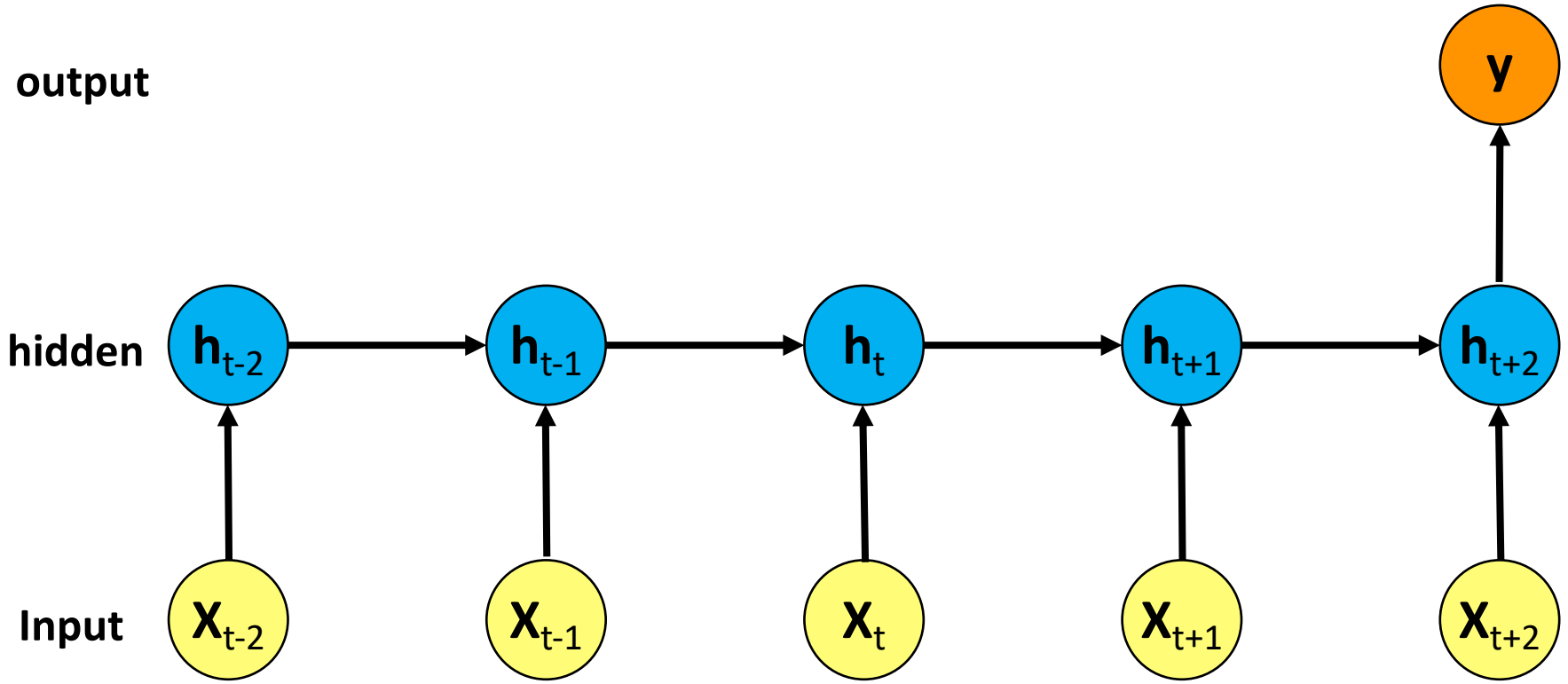
Recurrent Neural Networks (RNN)



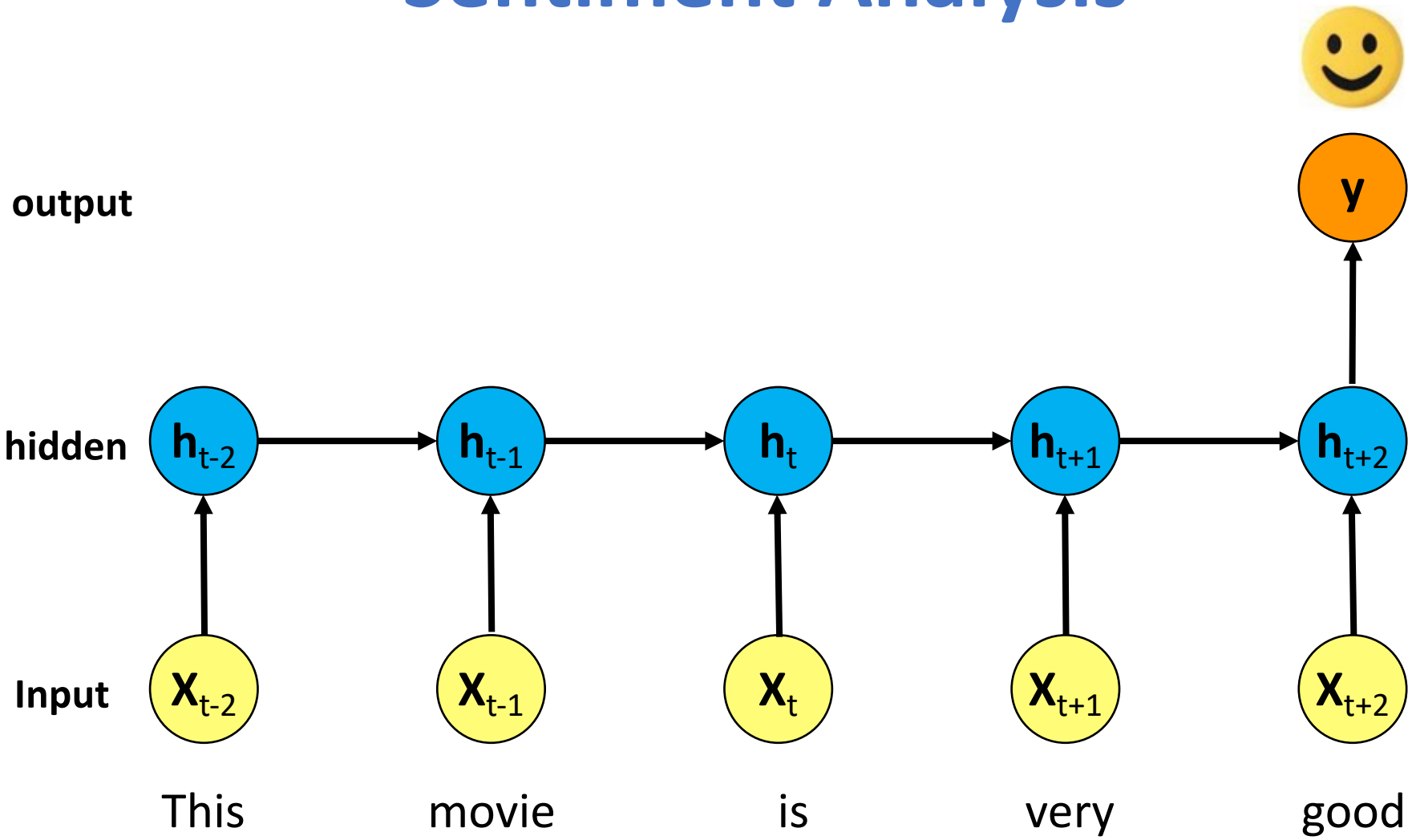
Recurrent Neural Networks (RNN)
Time Series Forecasting



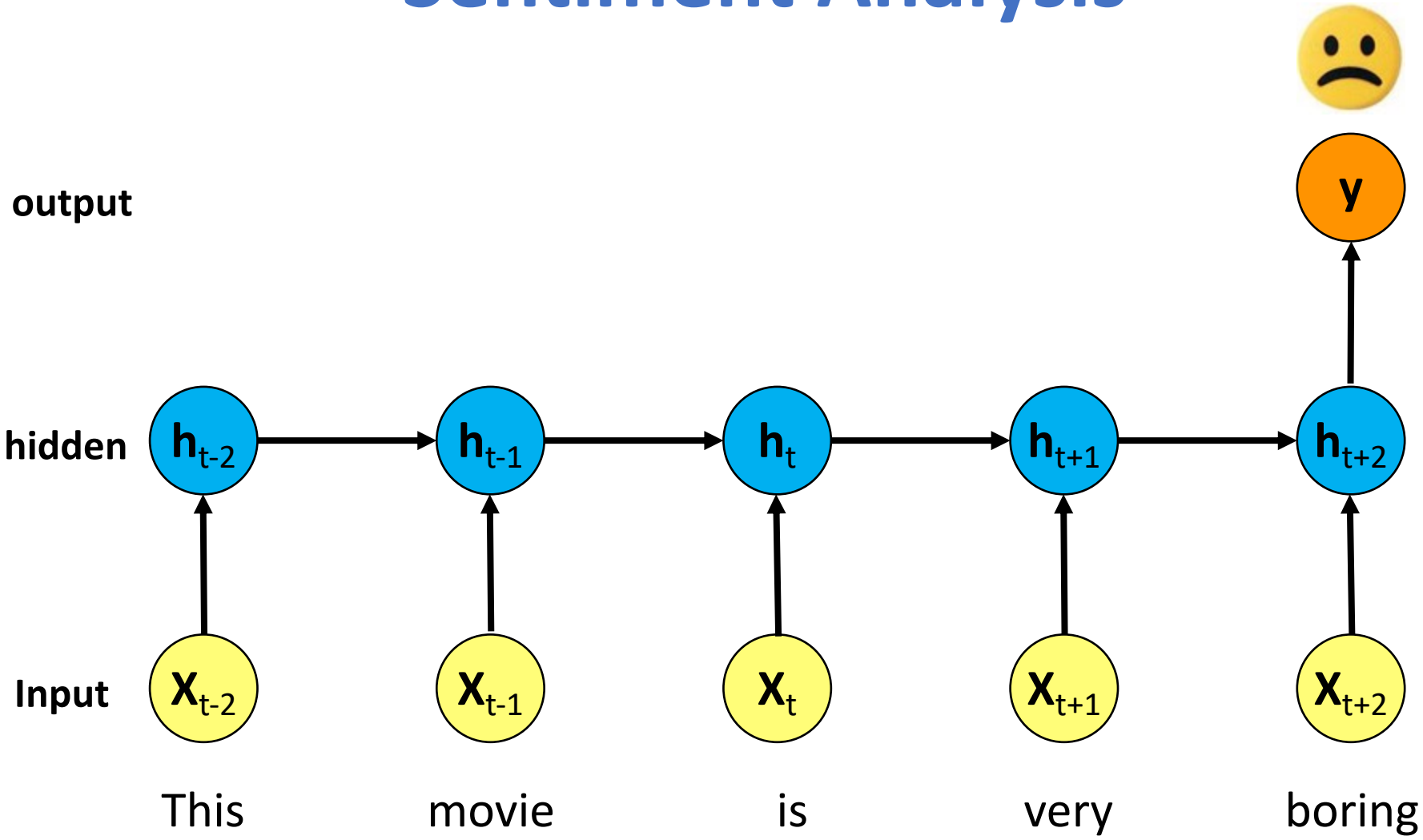
Recurrent Neural Networks (RNN)



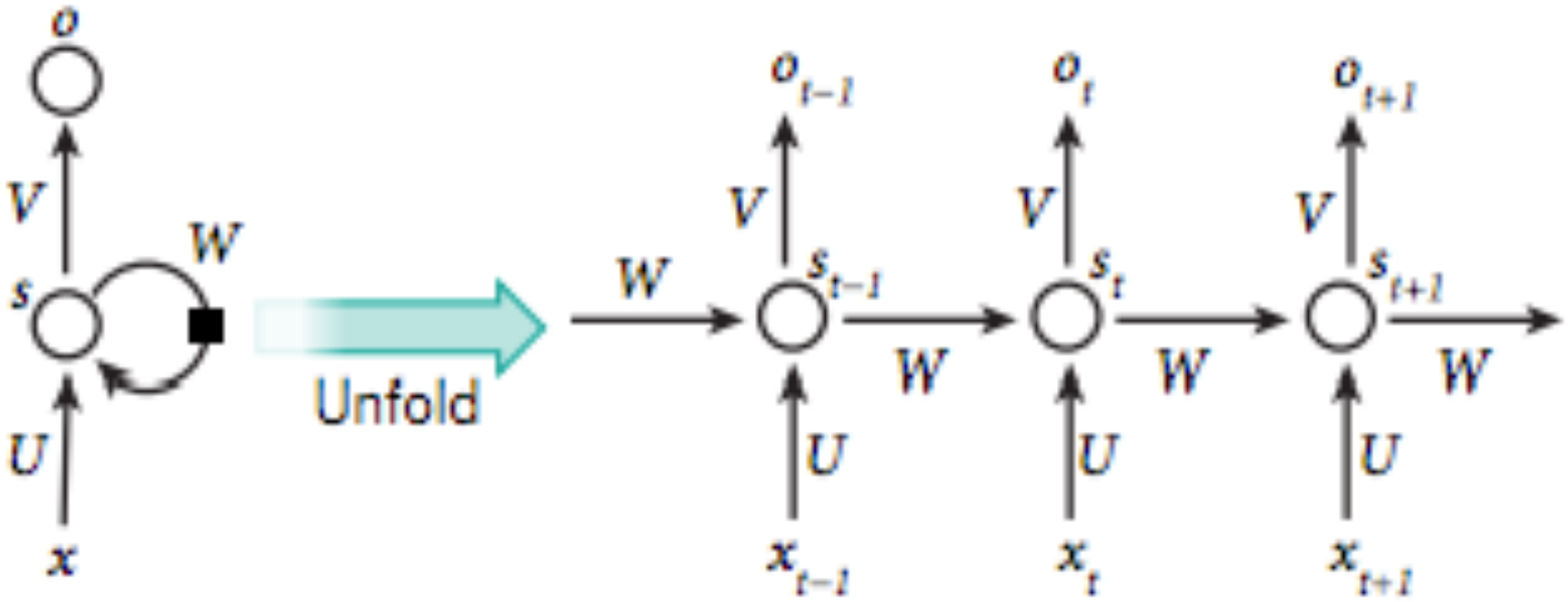
Recurrent Neural Networks (RNN)
Sentiment Analysis



Recurrent Neural Networks (RNN)
Sentiment Analysis

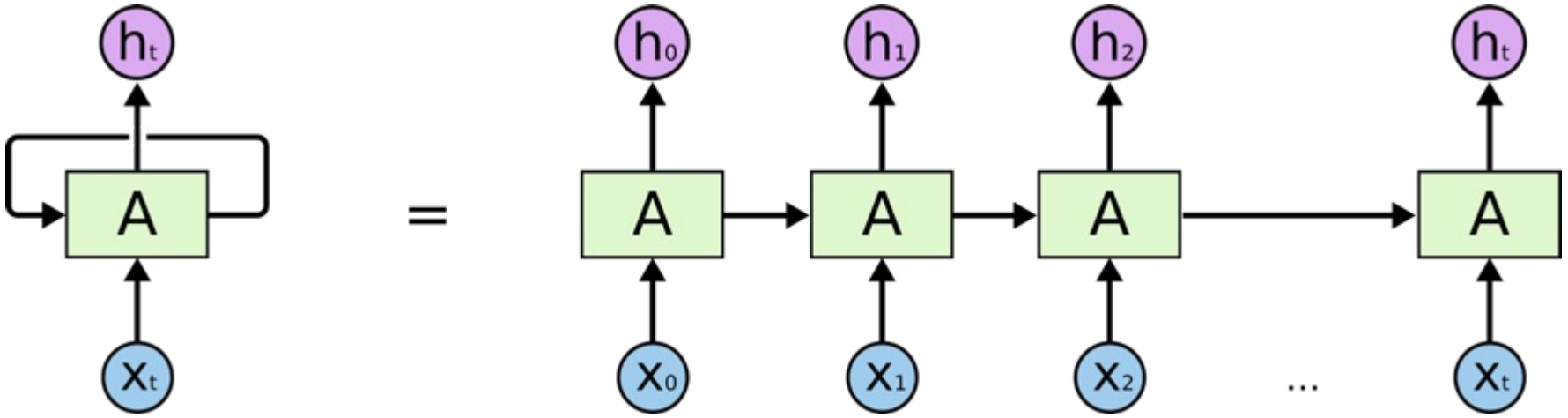


Recurrent Neural Network (RNN)



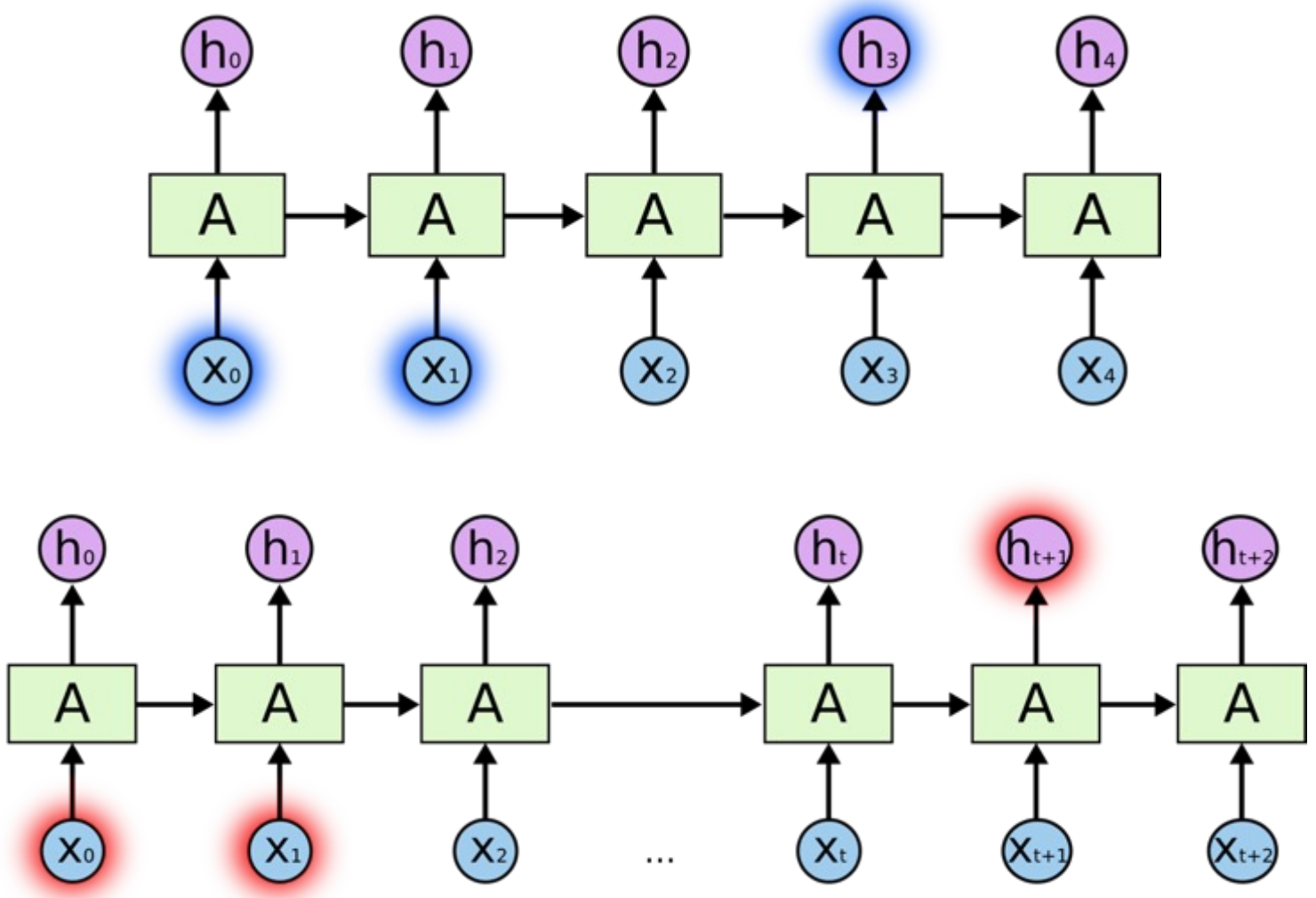
Source: LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." Nature 521, no. 7553 (2015): 436-444.

RNN



Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

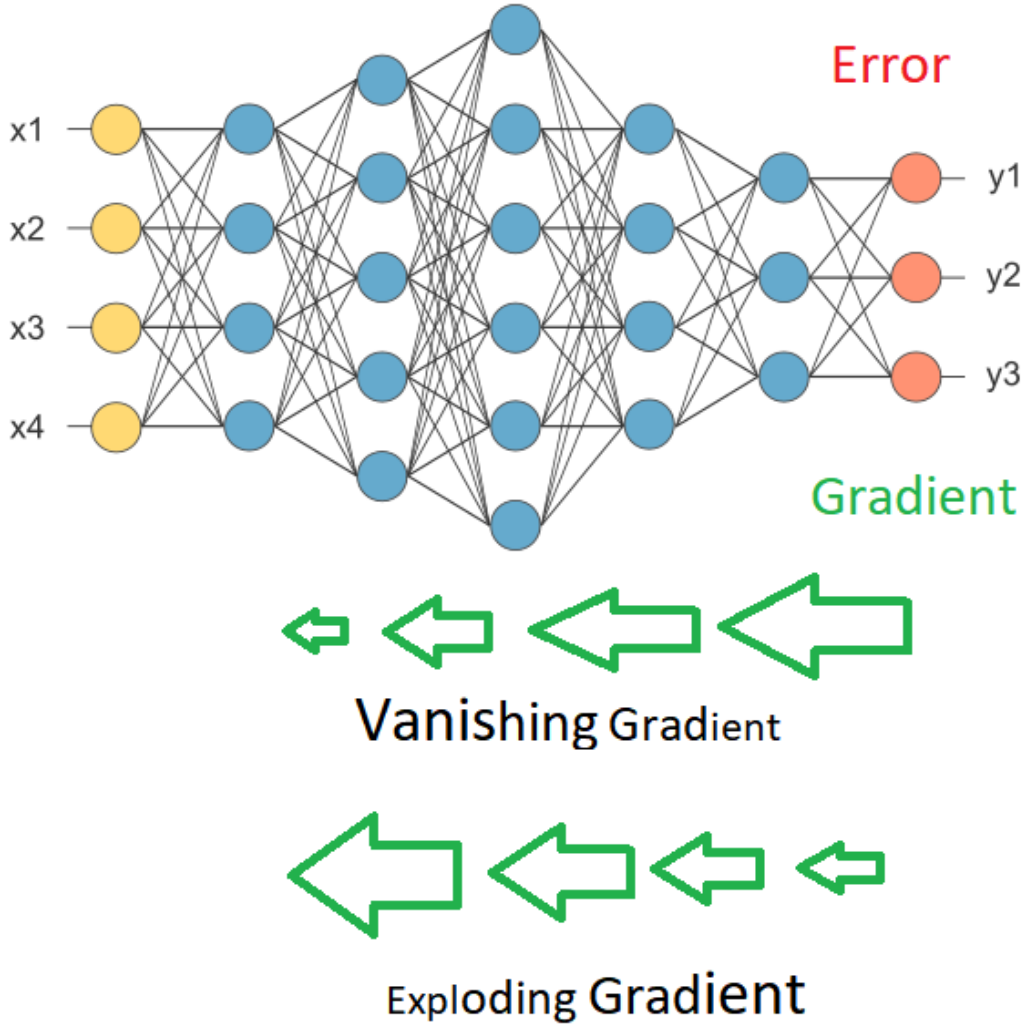
RNN long-term dependencies



I grew up in France… I speak fluent French.

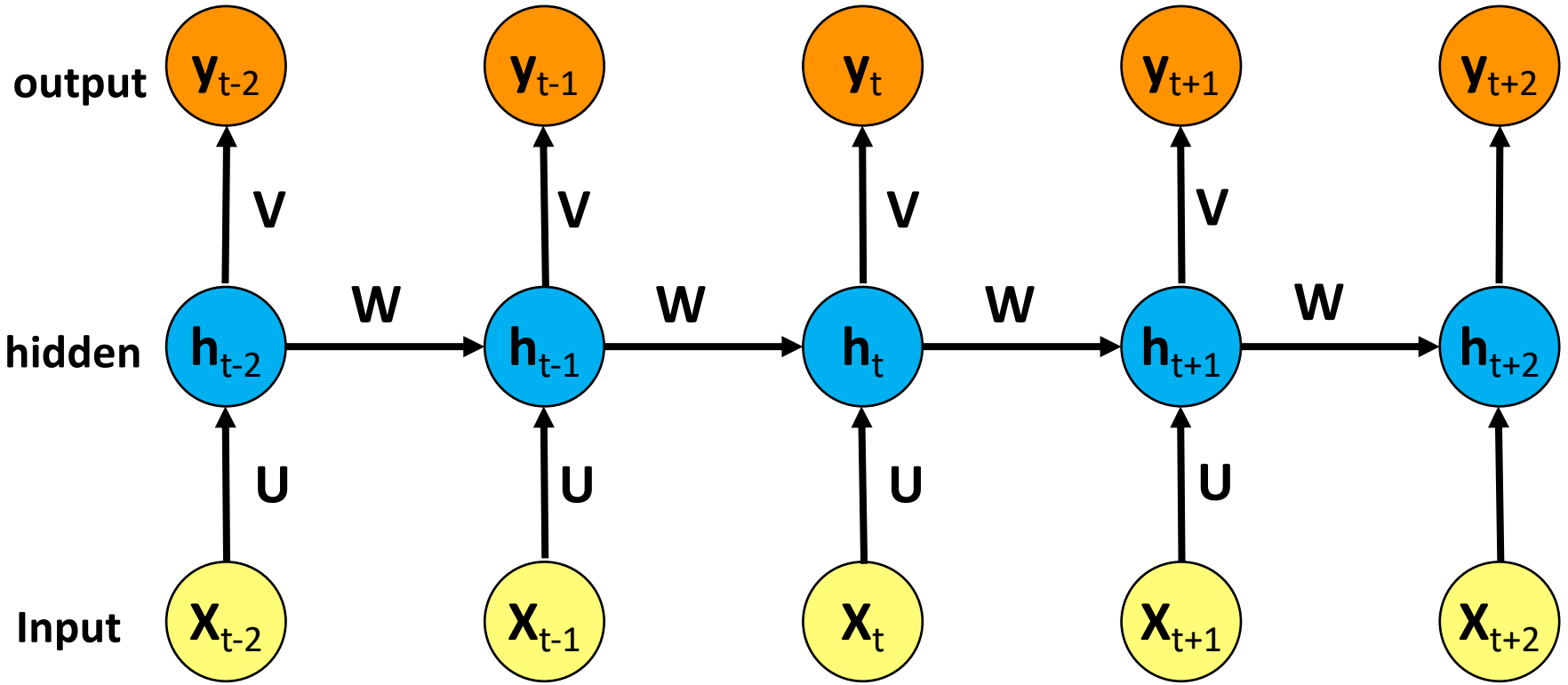
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Vanishing Gradient
Exploding Gradient

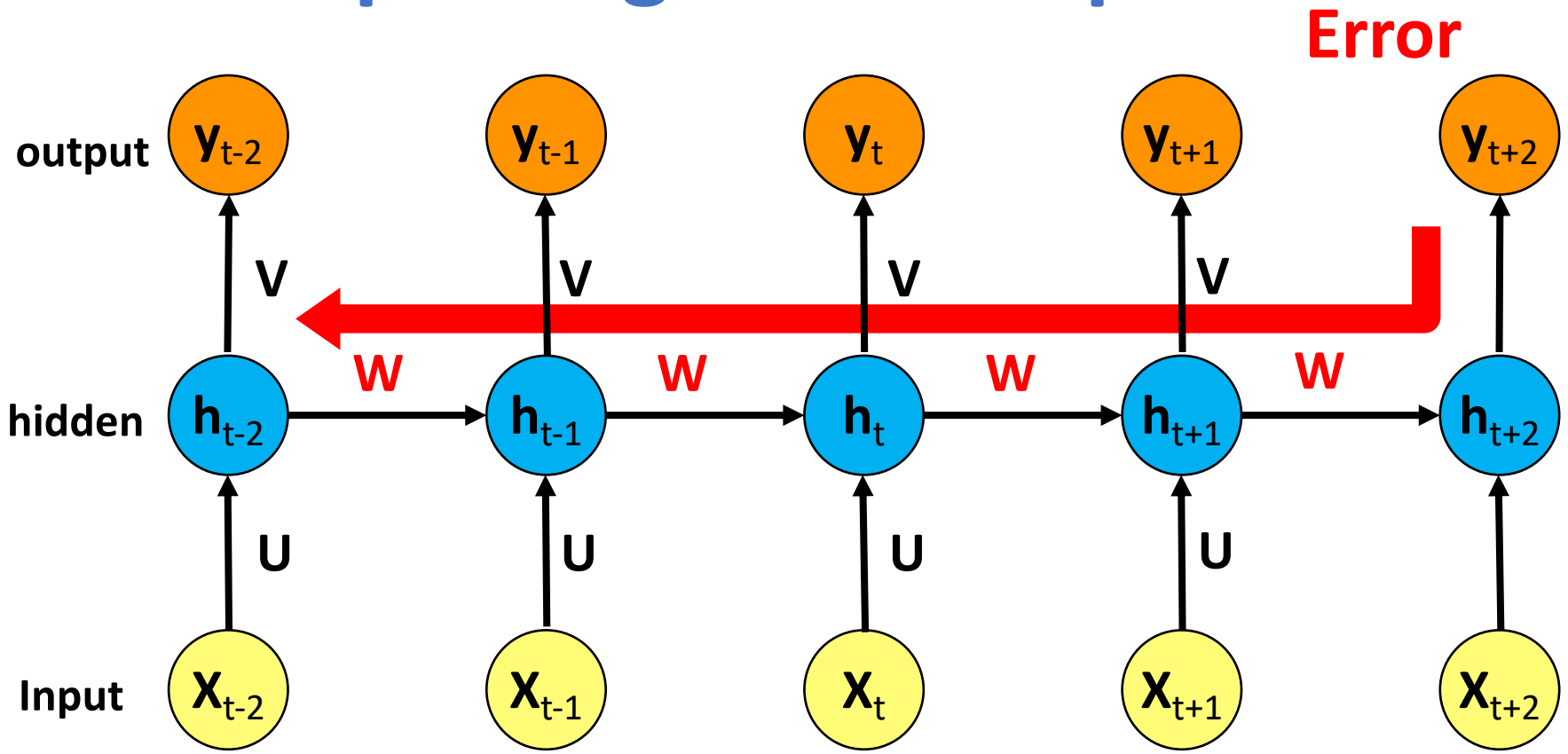


Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

Recurrent Neural Networks (RNN)



RNN
Vanishing Gradient problem
Exploding Gradient problem

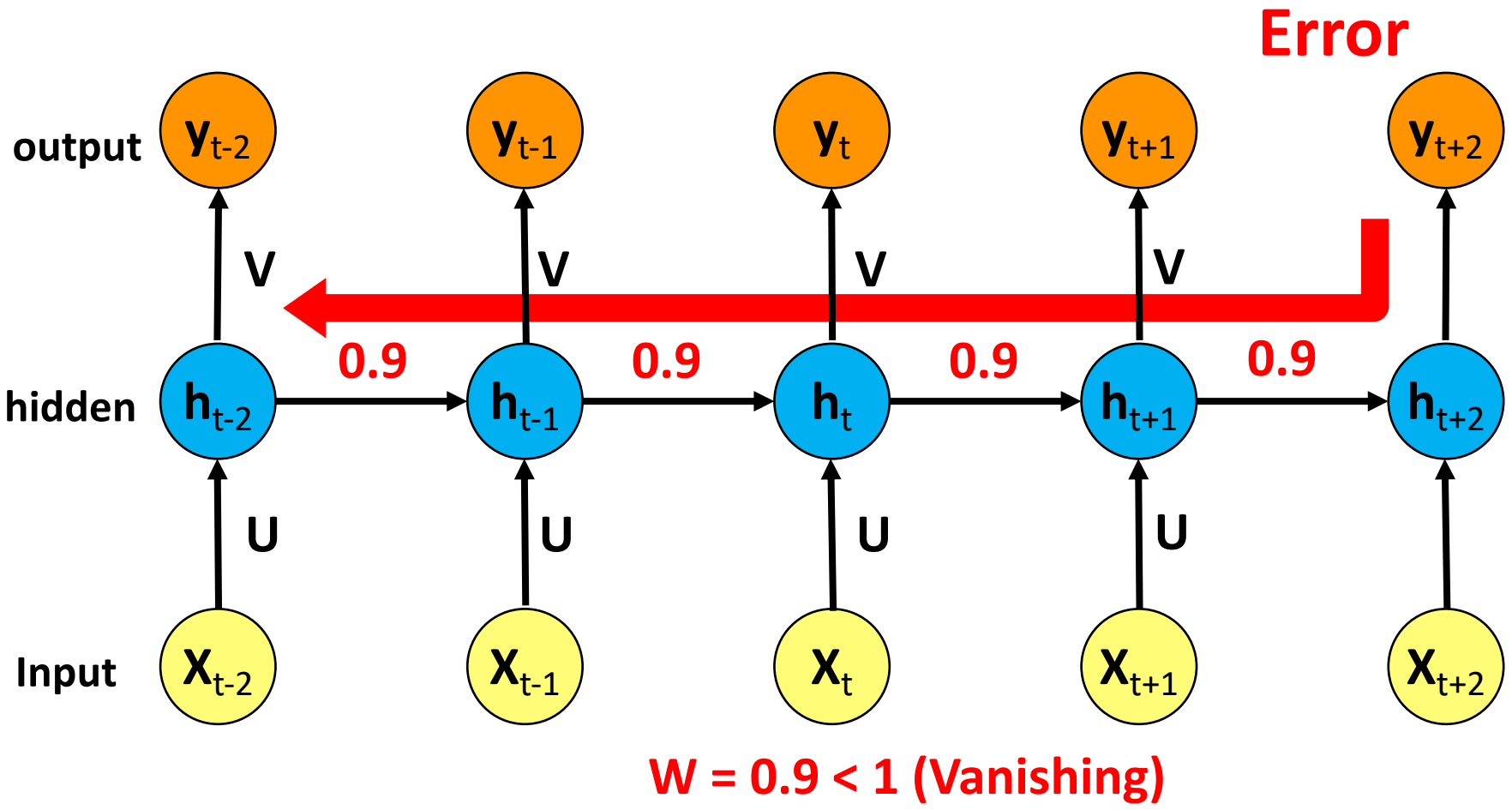


if |W| < 1 (Vanishing)

if |W| > 1 (Exploding)

Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

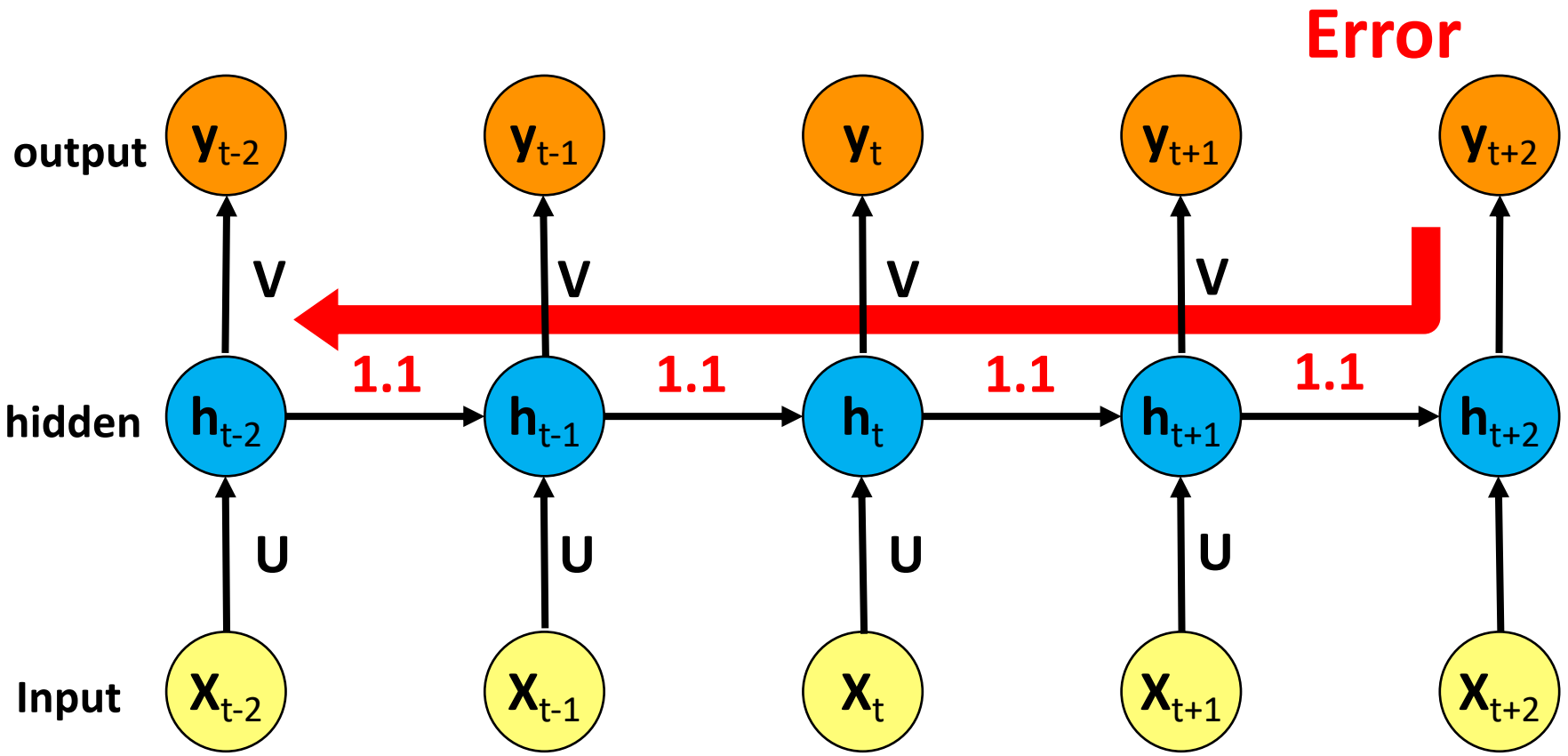
RNN
Vanishing Gradient problem



W = 0.9 < 1 (Vanishing)

Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

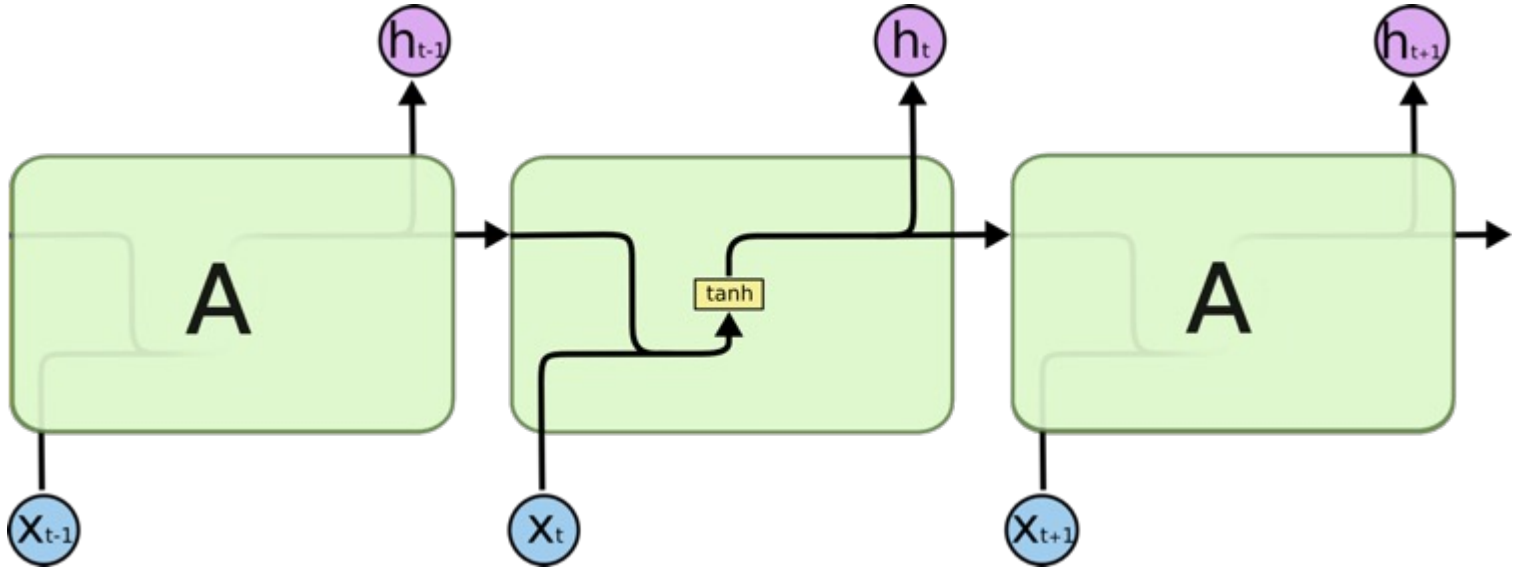
RNN
Exploding Gradient problem



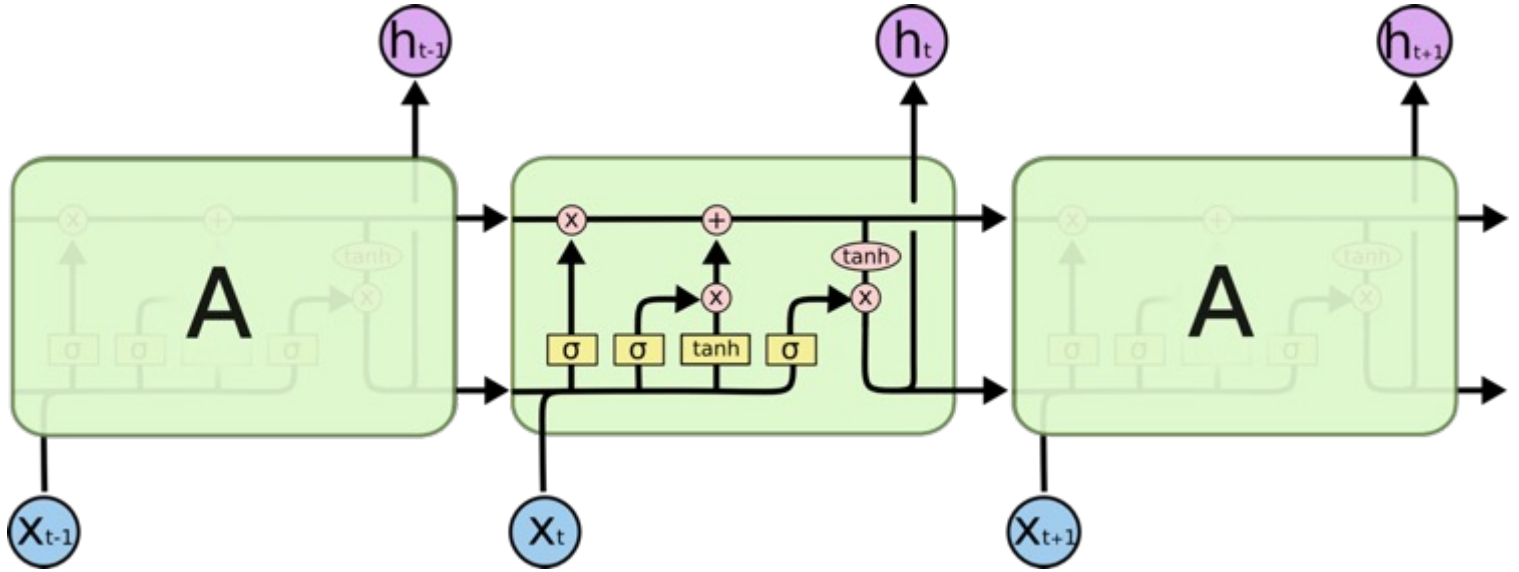
Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

RNN LSTM

RNN

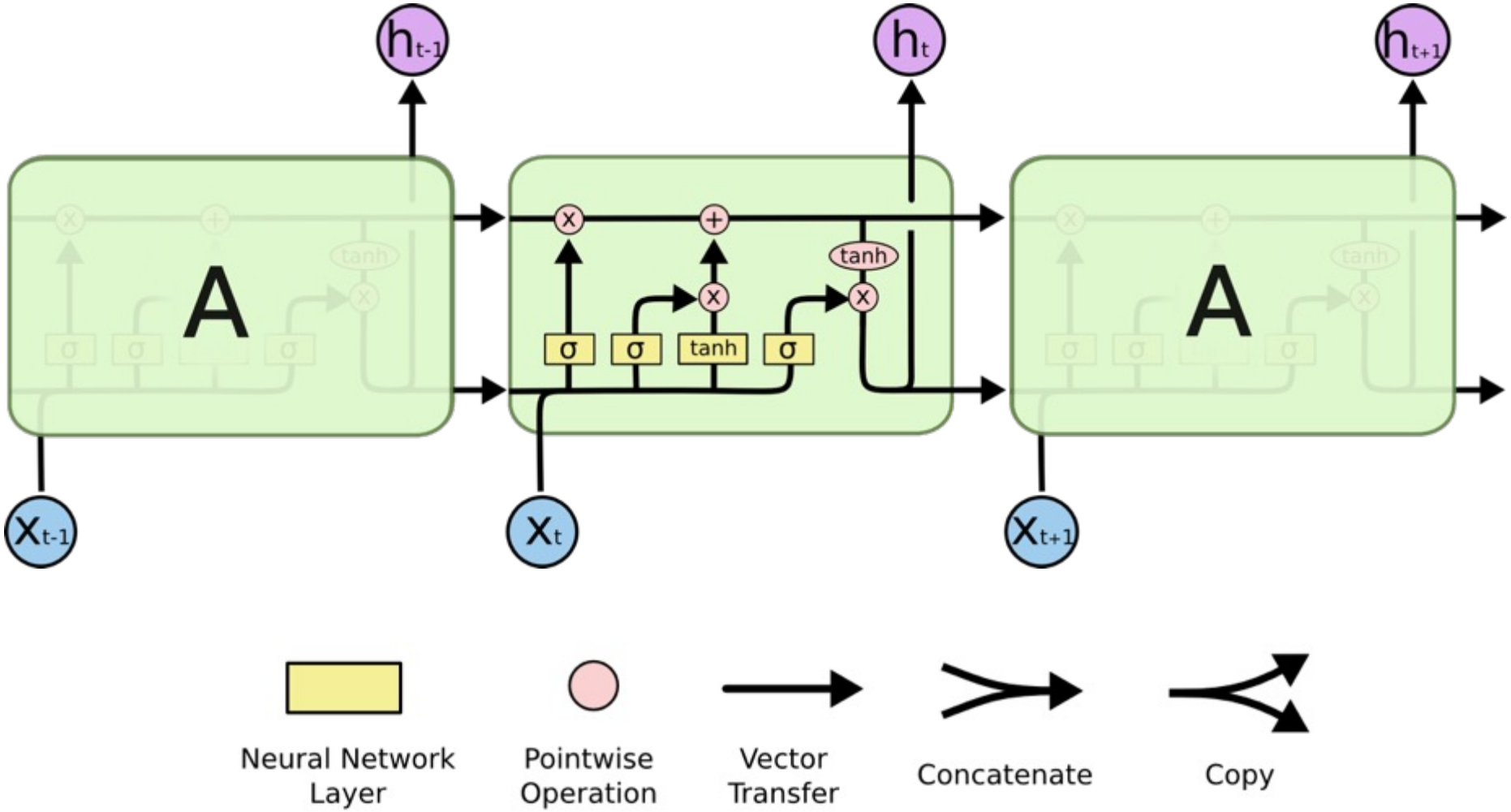


LSTM



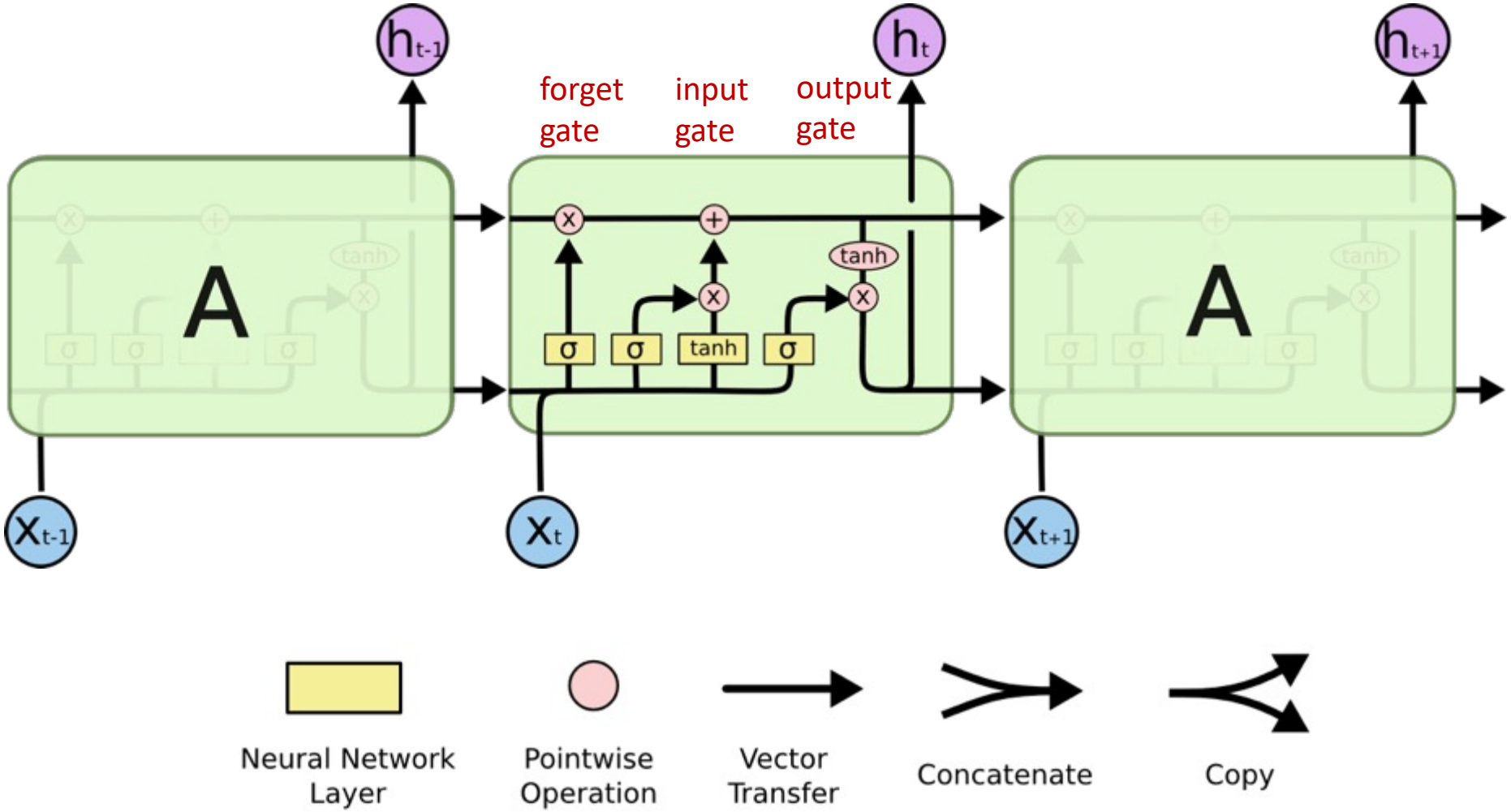
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory (LSTM)



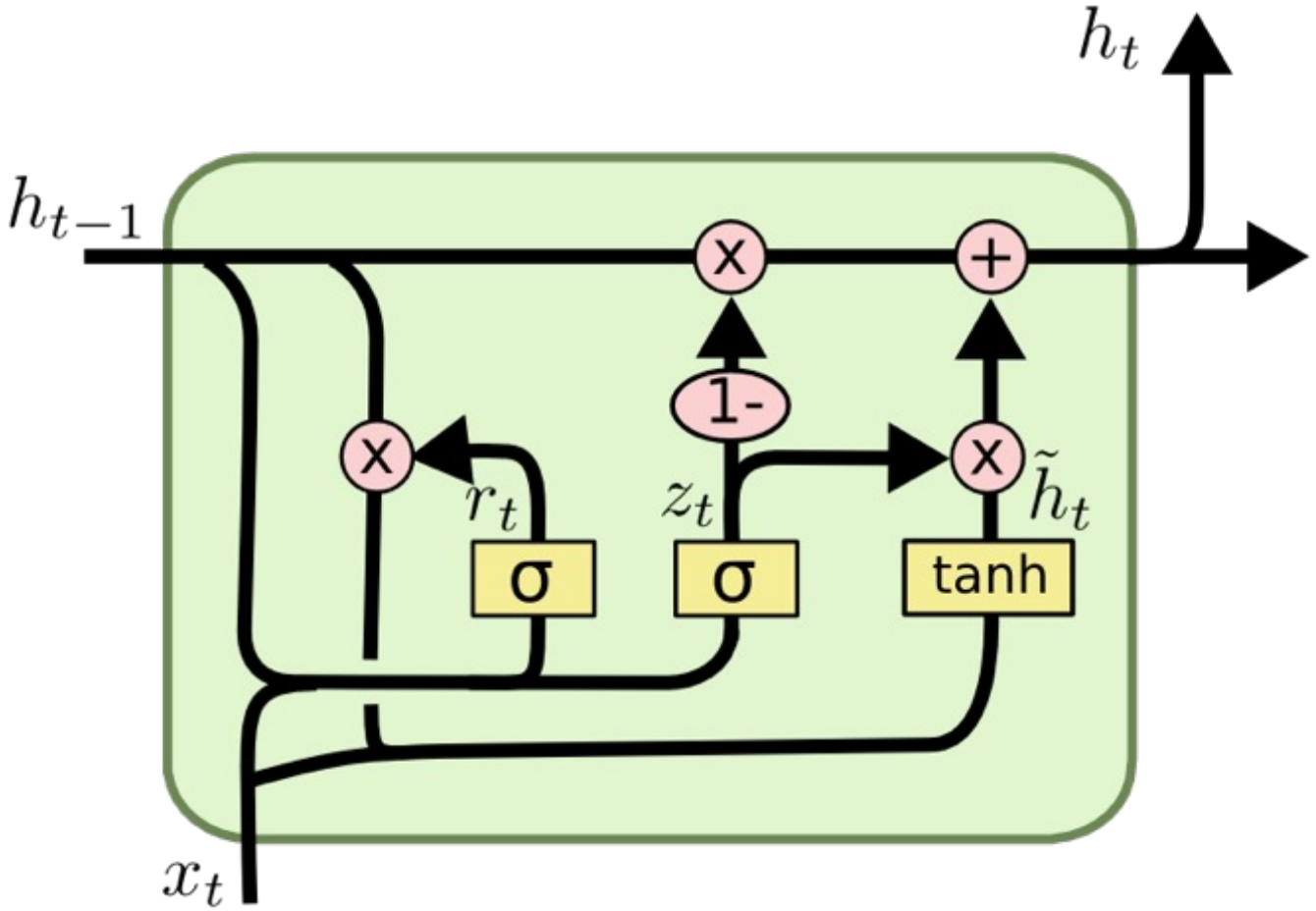
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory (LSTM)



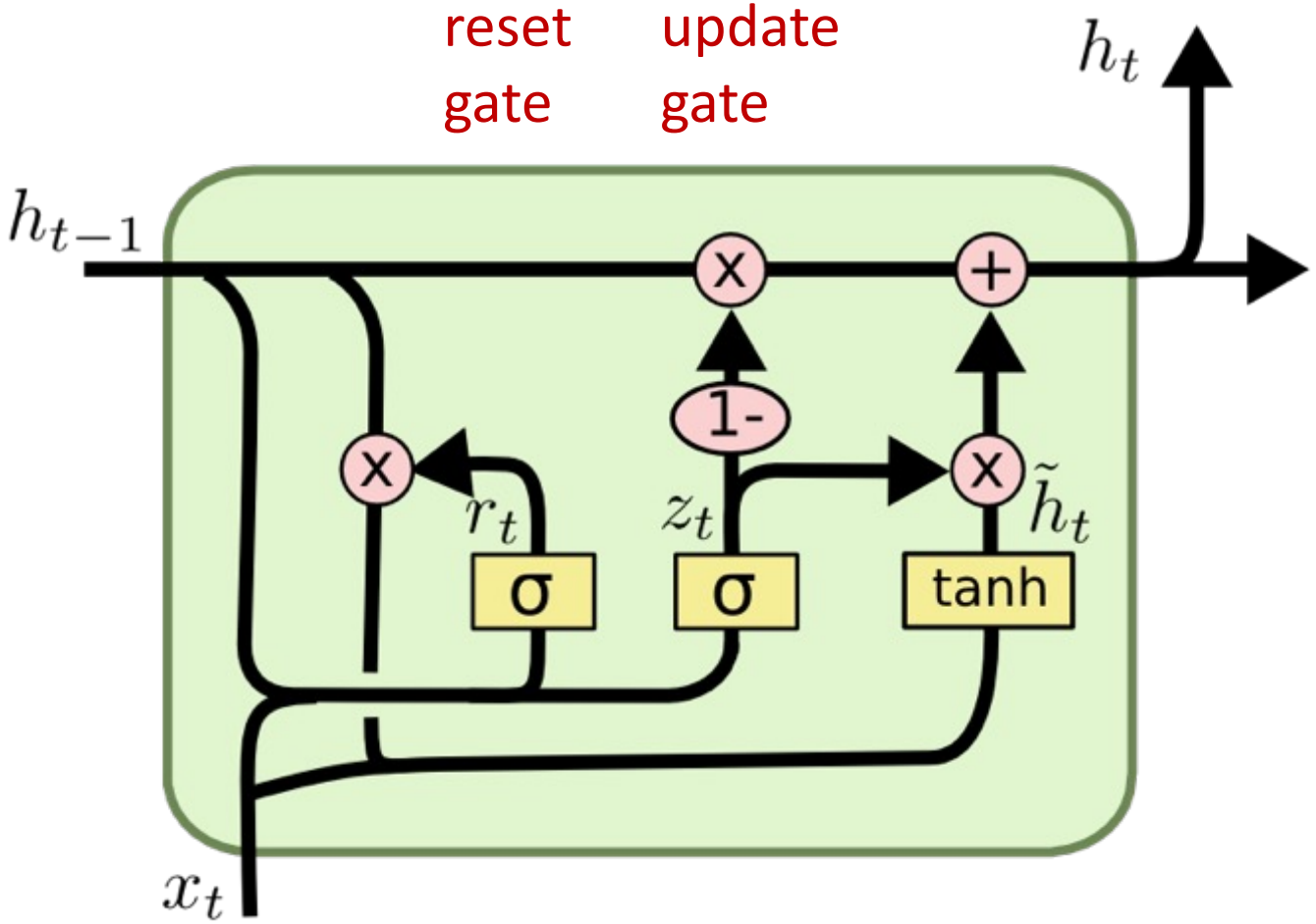
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit (GRU)



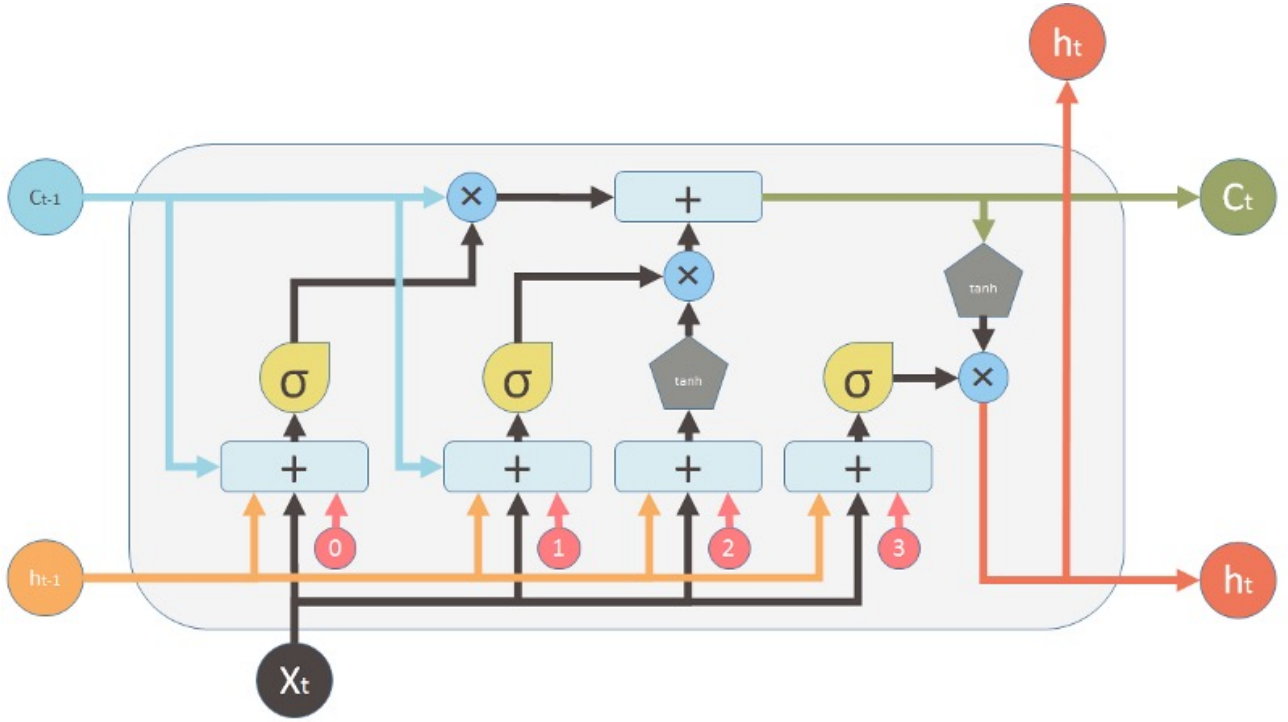
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit (GRU)



Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM



Inputs:

- X_t Input vector
- C_{t-1} Memory from previous block
- h_{t-1} Output of previous block

outputs:

- C_t Memory from current block
- h_t Output of current block

Nonlinearities:

- σ Sigmoid
- tanh Hyperbolic tangent

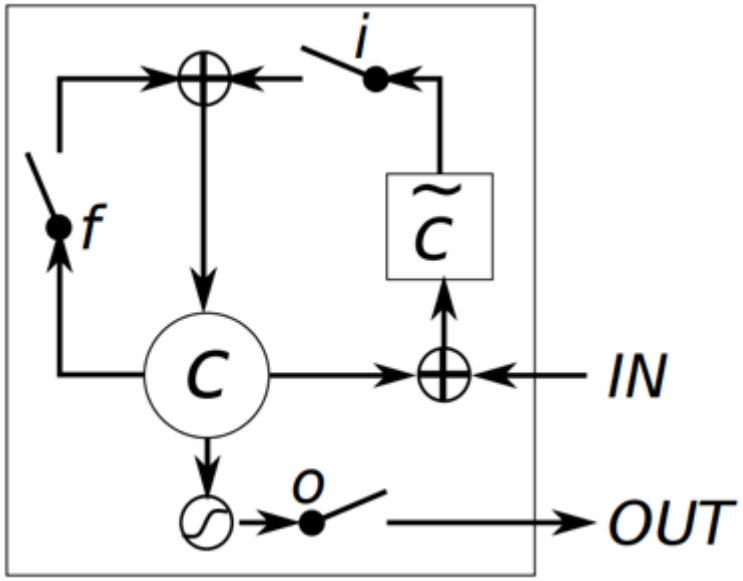
Bias: 0

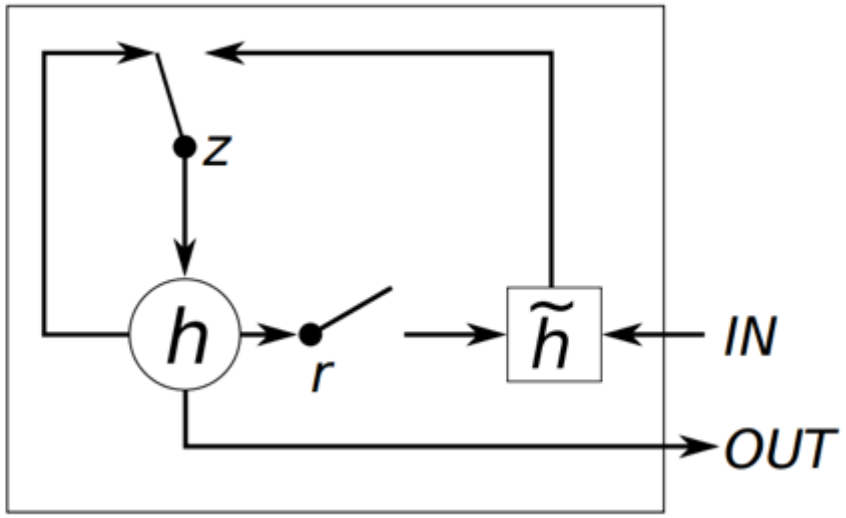
Vector operations:

- × Element-wise multiplication
- \+ Element-wise Summation / Concatenation

Source: Shi Yan (2016), Understanding LSTM and its diagrams, https://medium.com/mlreview/understanding-lstm-and-its-diagrams-37e2f46f1714

LSTM vs GRU





LSTM

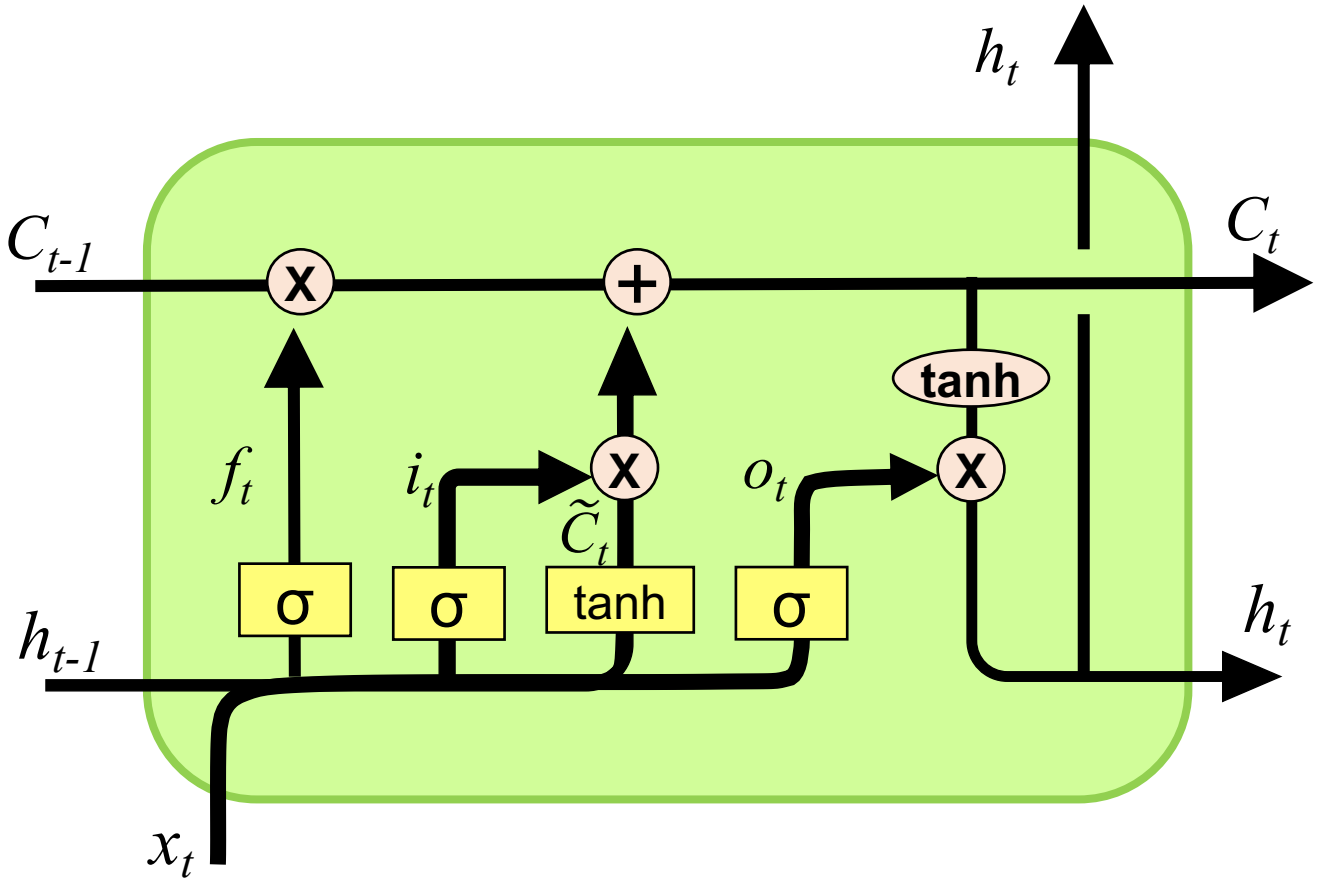
i, f and o are the input, forget and output gates, respectively.
c and c˜ denote the memory cell and the new memory cell content.

GRU

r and z are the reset and update gates, and h and h˜ are the activation and the candidate activation.

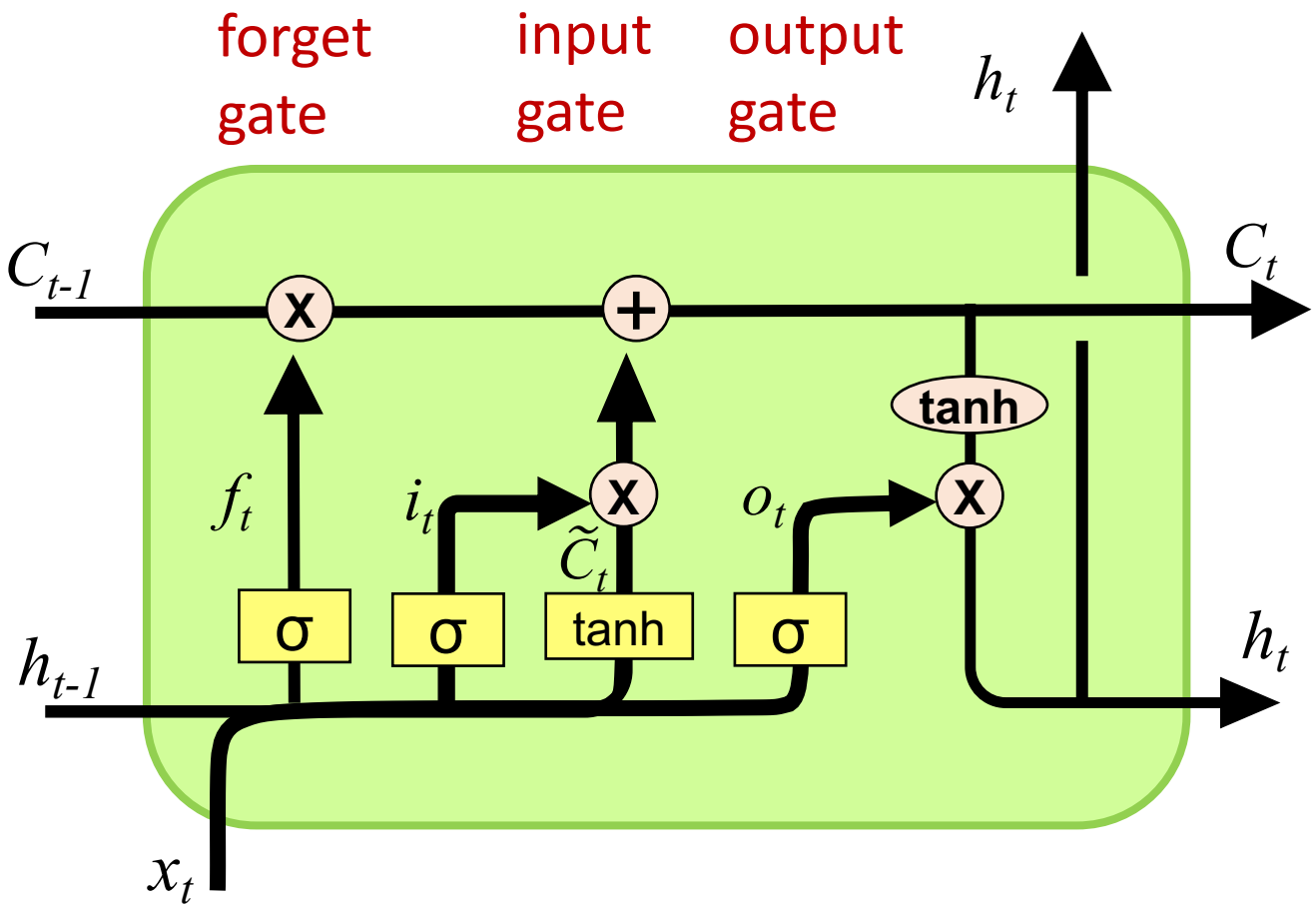
Source: Chung, Junyoung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. "Empirical evaluation of gated recurrent neural networks on sequence modeling." *arXiv preprint arXiv:1412.3555* (2014).

Long Short Term Memory (LSTM)



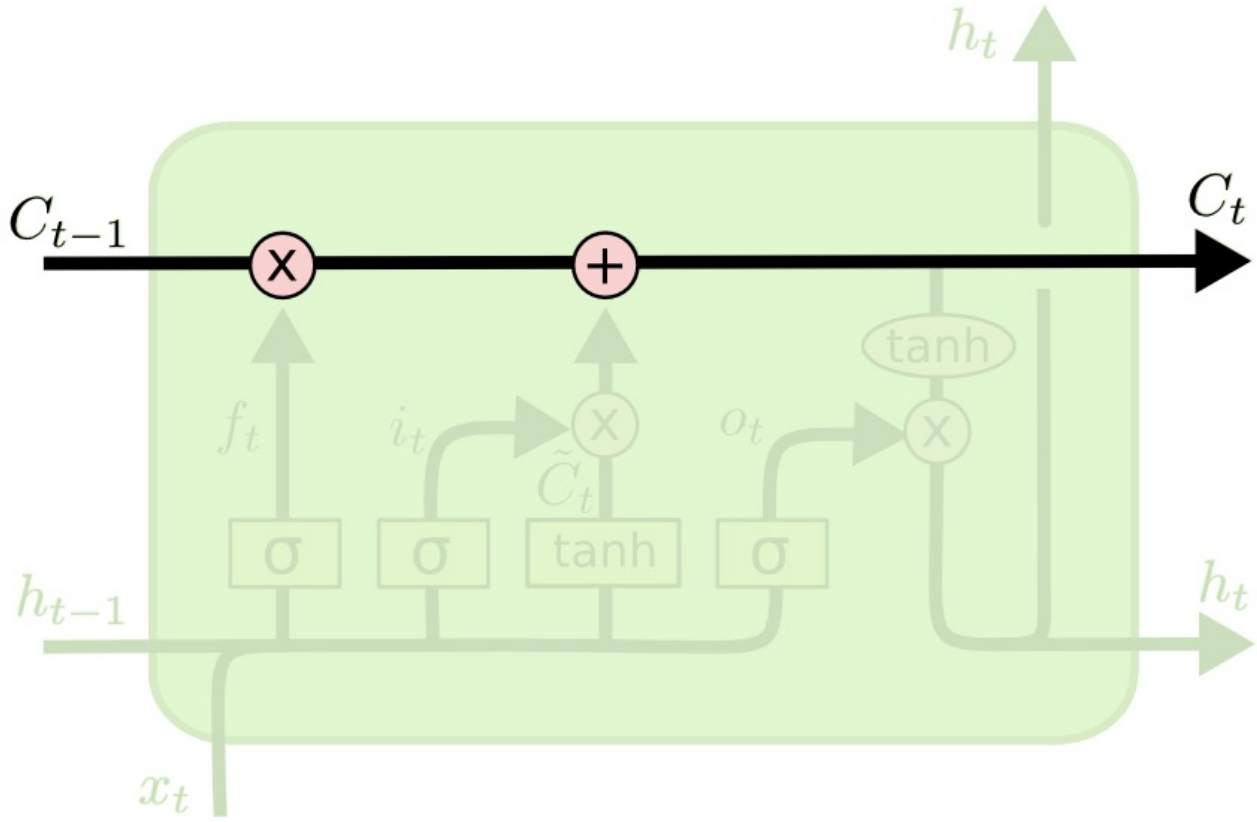
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory (LSTM)



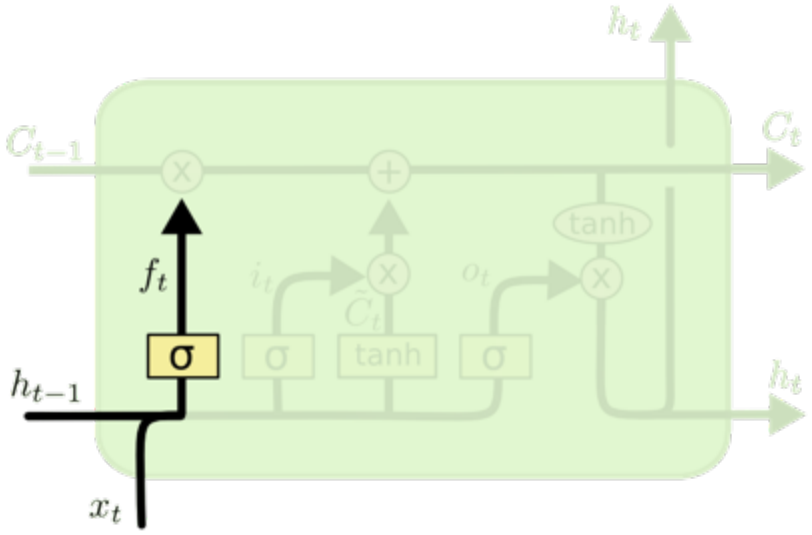
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM
Memory state (C)



Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

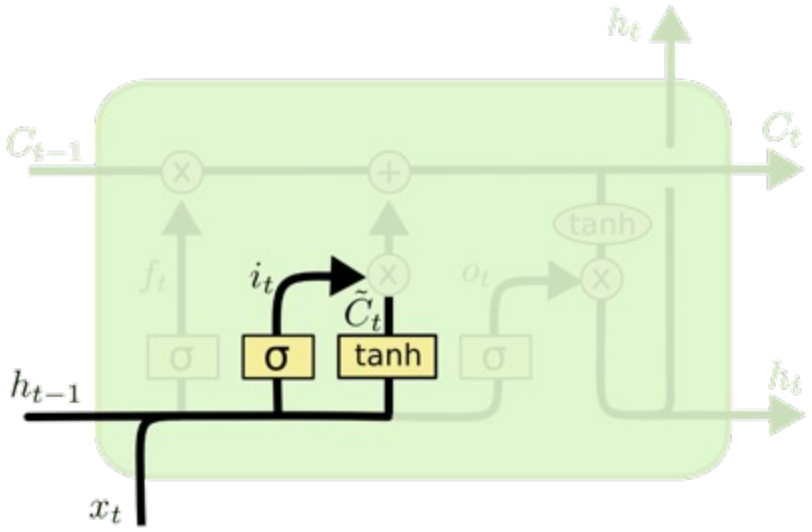
LSTM
forget gate (f)



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM
input gate (i)

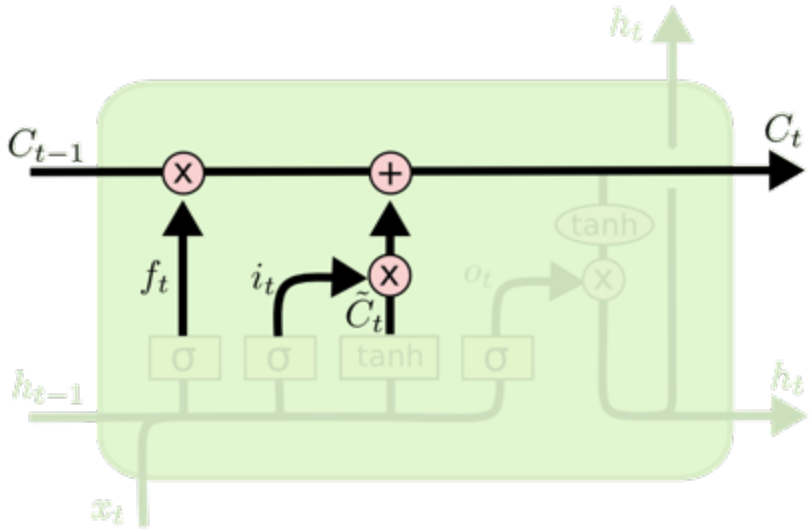


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

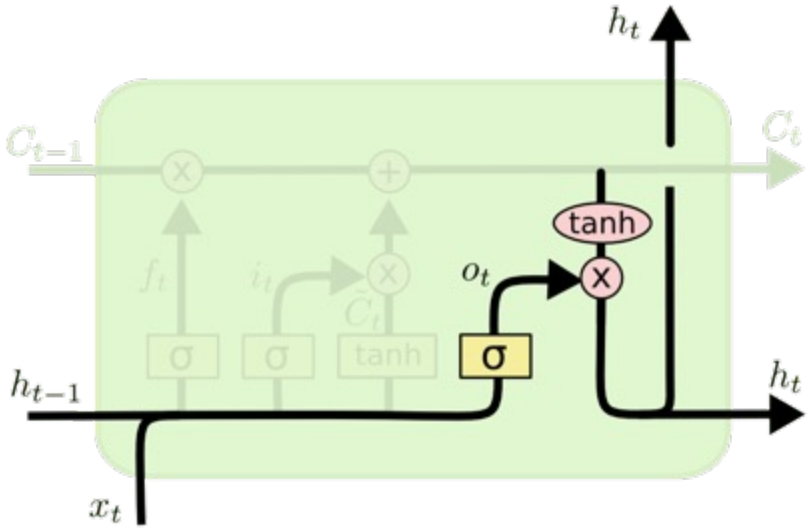
LSTM
Memory state (C)



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM
output gate (o)

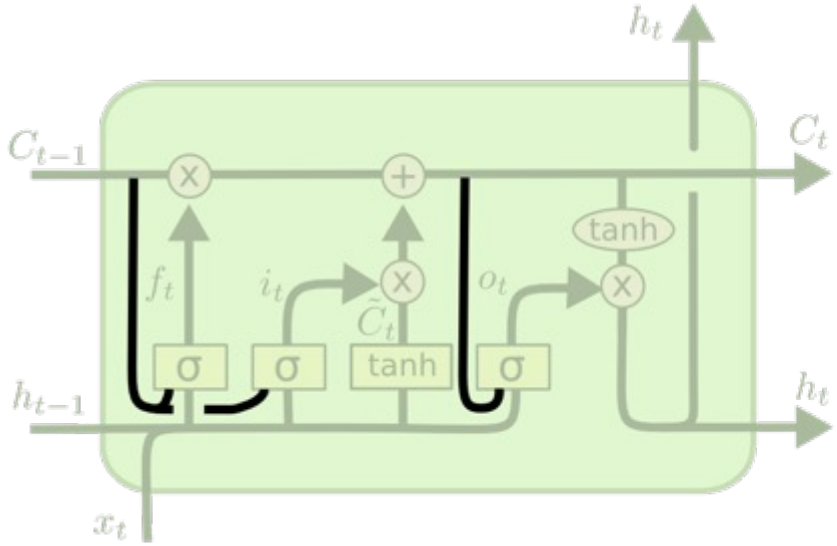


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM
forget (f), input (i), output (o) gates

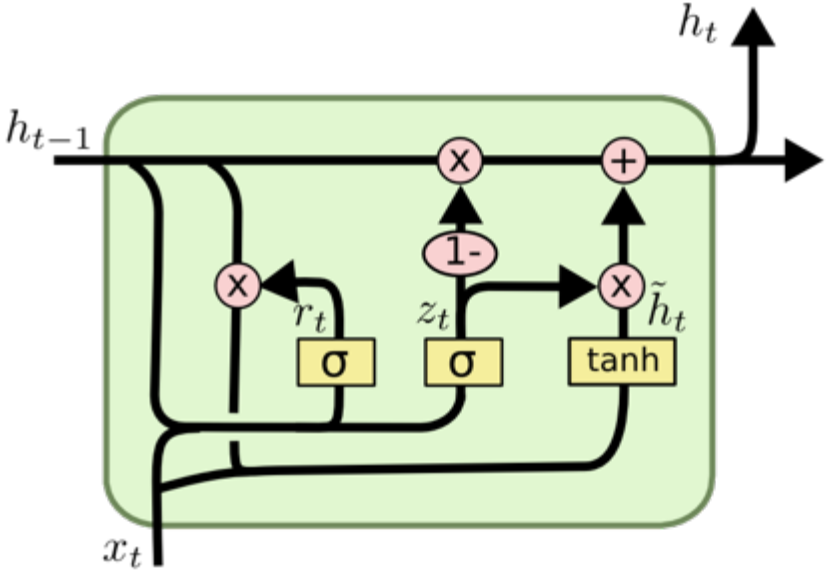


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit (GRU)

update (z), reset (r) gates



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

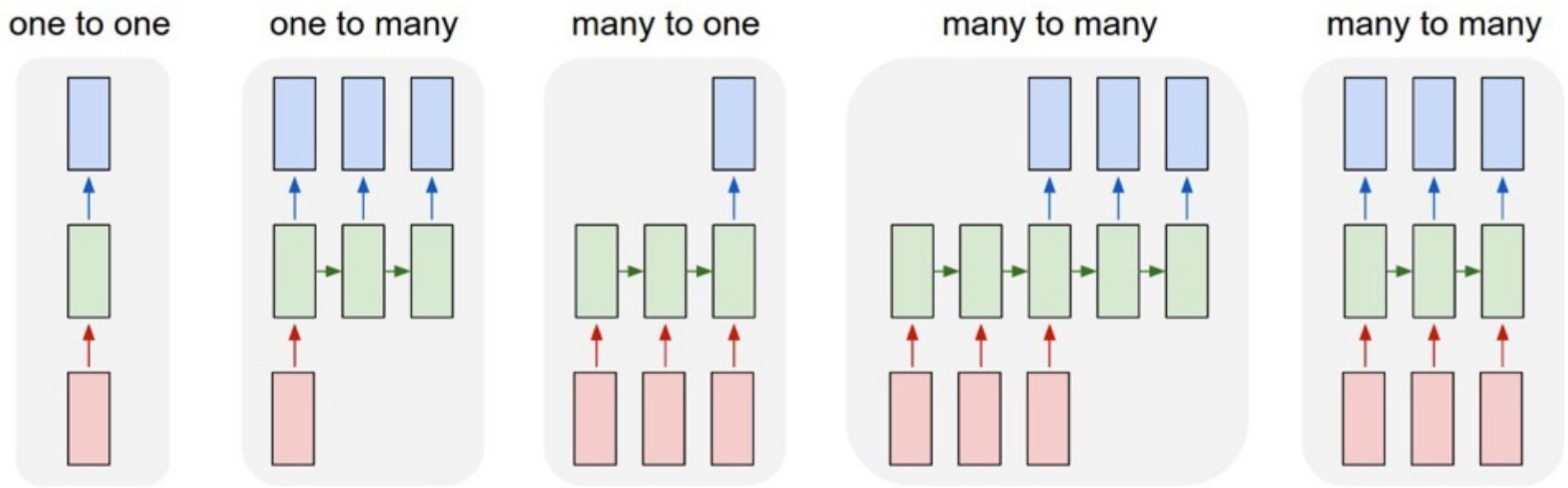
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM Recurrent Neural Network



Traditional Neural Network

Music Generation

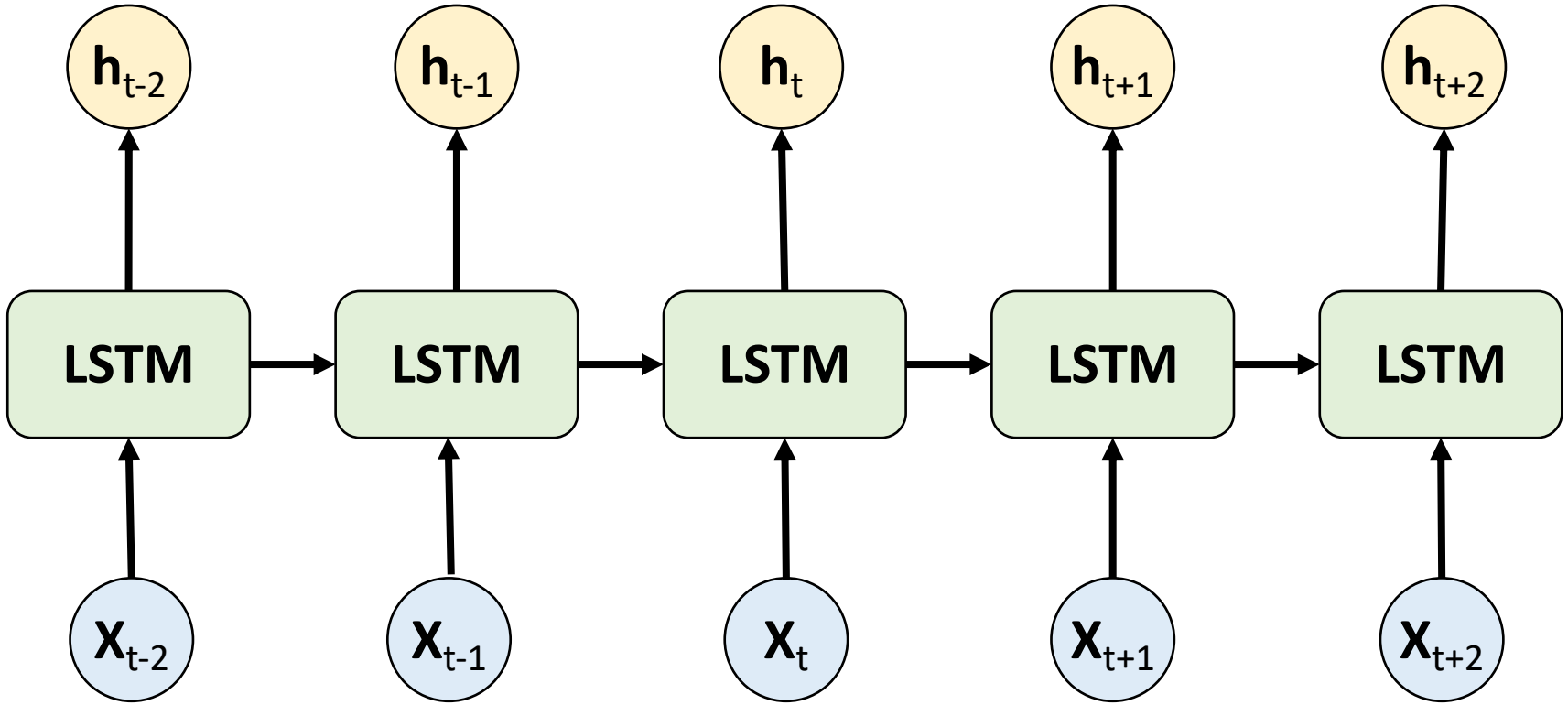
Sentiment Classification

Name Entity Recognition

Machine Translation

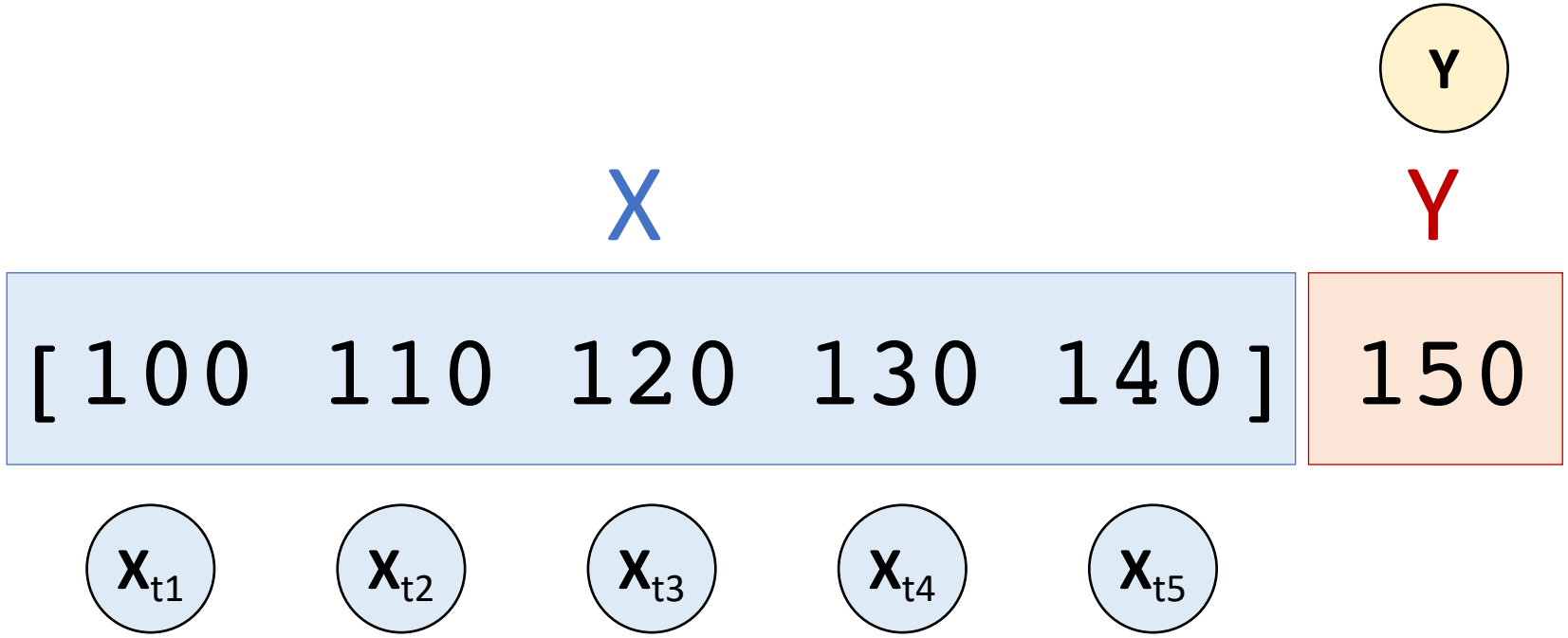
Source: https://github.com/Vict0rSch/deep_learning/tree/master/keras/recurrent

Long Short Term Memory (LSTM) for Time Series Forecasting



Time Series Data

```
[100, 110, 120, 130, 140, 150]
```



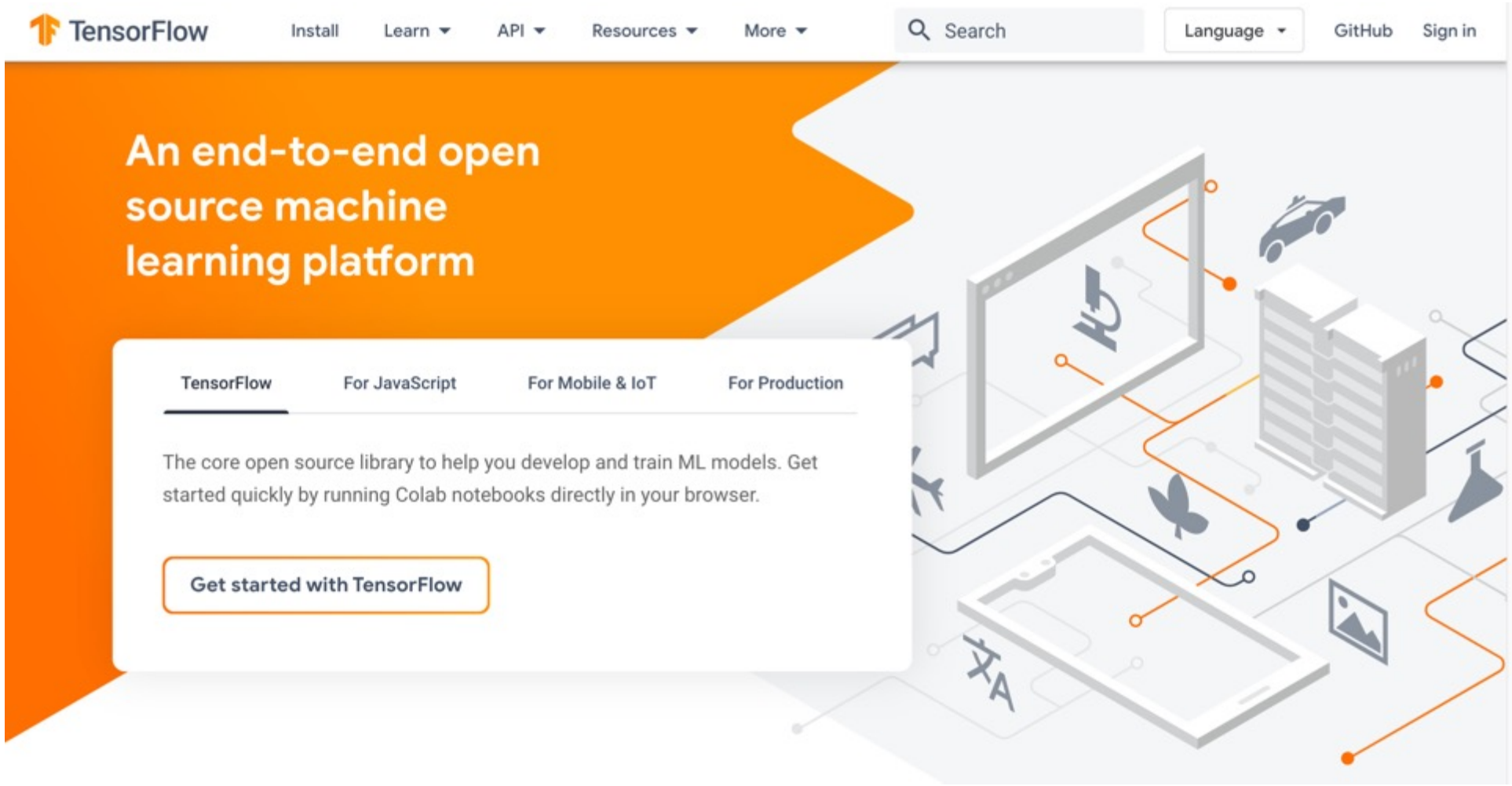
Time Series Data

[10, 20, 30, 40, 50, 60, 70, 80, 90]

X	Y
[10 20 30]	40
[20 30 40]	50
[30 40 50]	60
[40 50 60]	70
[50 60 70]	80
[60 70 80]	90

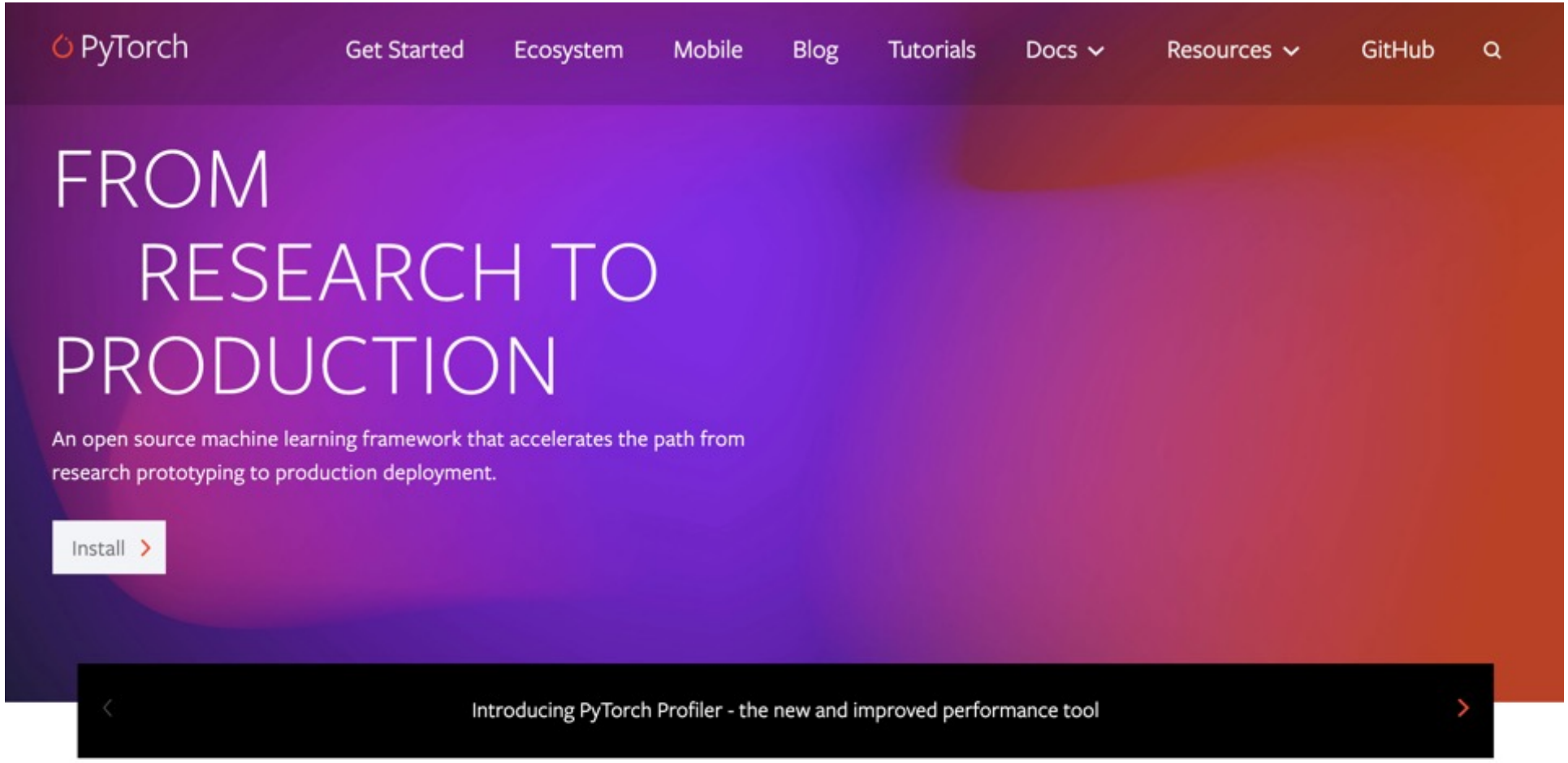
TensorFlow

TensorFlow



https://www.tensorflow.org/

PyTorch



https://pytorch.org/

TensorFlow

- **An end-to-end open source machine learning platform.**
- **The core open source library to help you develop and train ML models.**
- **Get started quickly by running Colab notebooks directly in your browser.**

https://www.tensorflow.org/

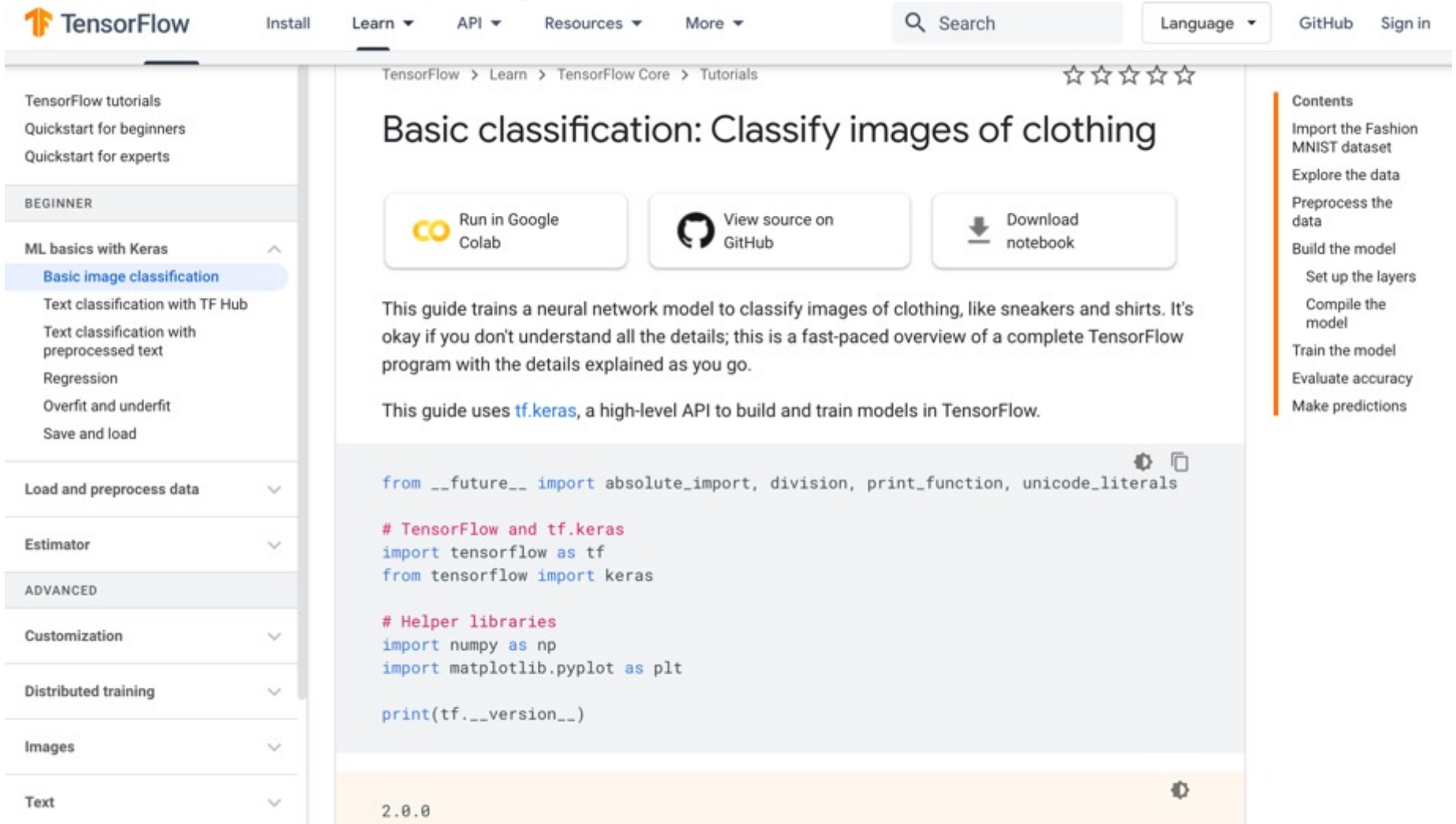
TensorFlow

TensorFlow 2.0

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

https://www.tensorflow.org/overview/

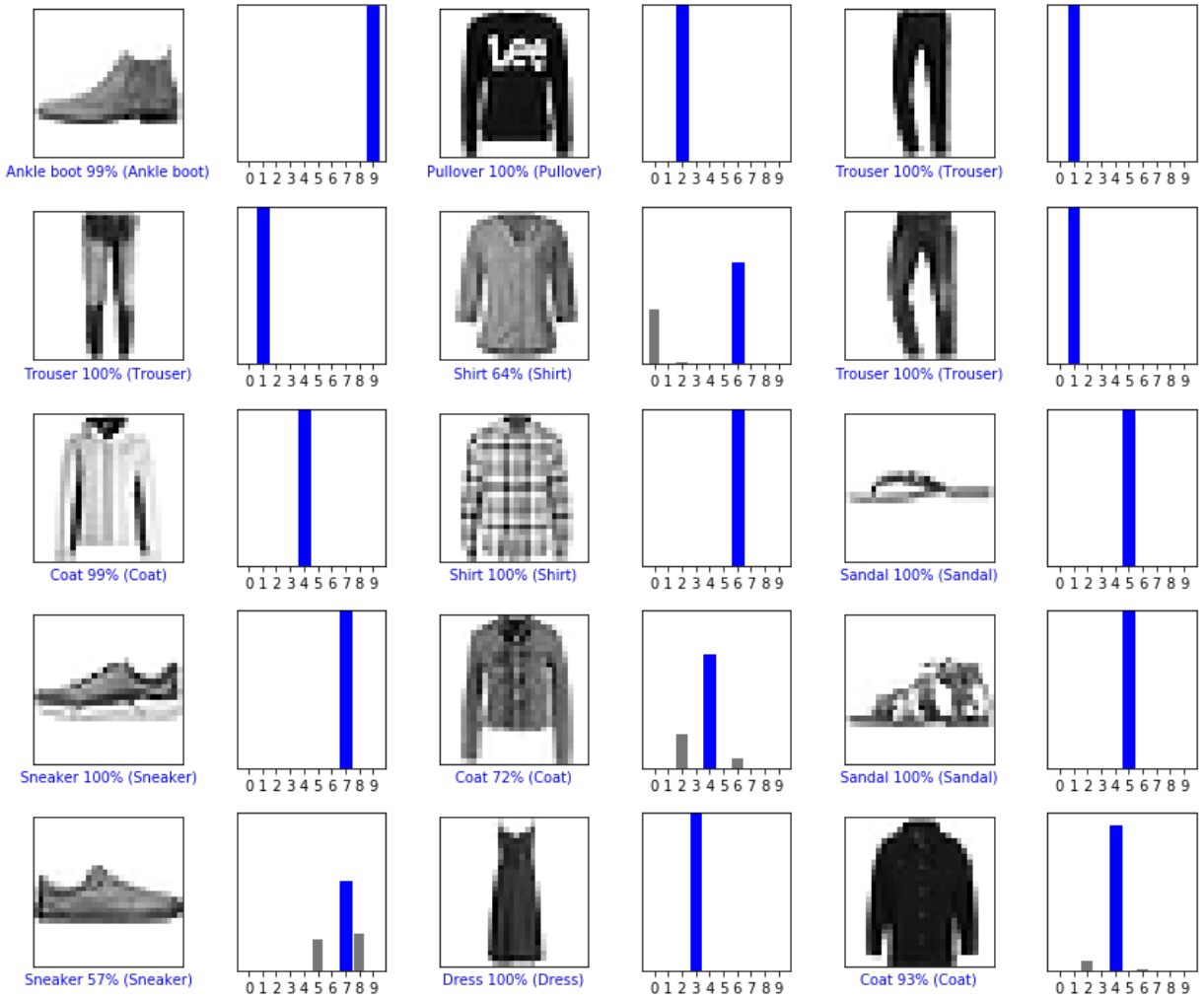
TensorFlow
Image Classification



https://www.tensorflow.org/tutorials/keras/classification

TensorFlow

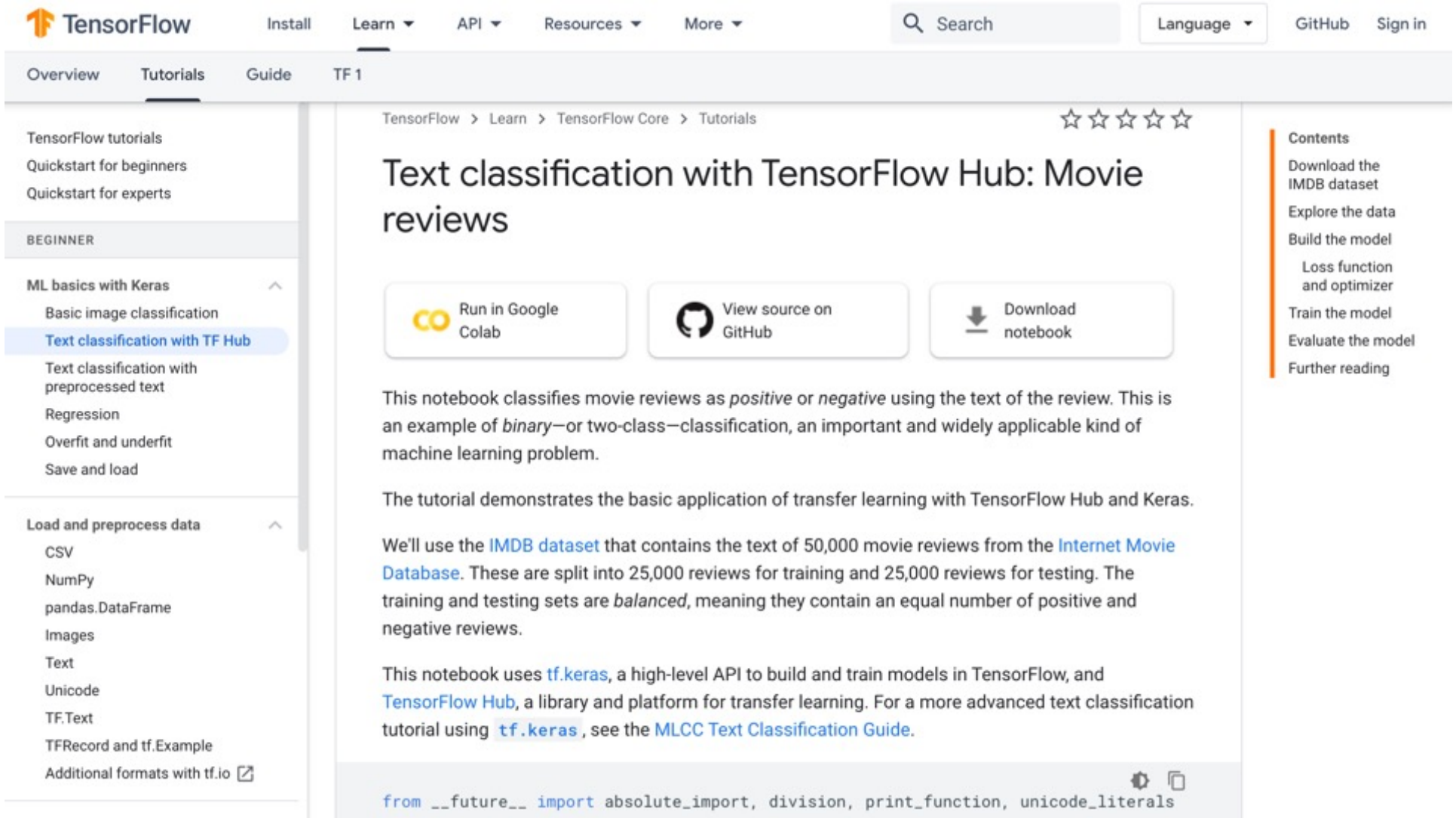
Image Classification
Fashion MNIST dataset



https://www.tensorflow.org/tutorials/keras/classification

TensorFlow

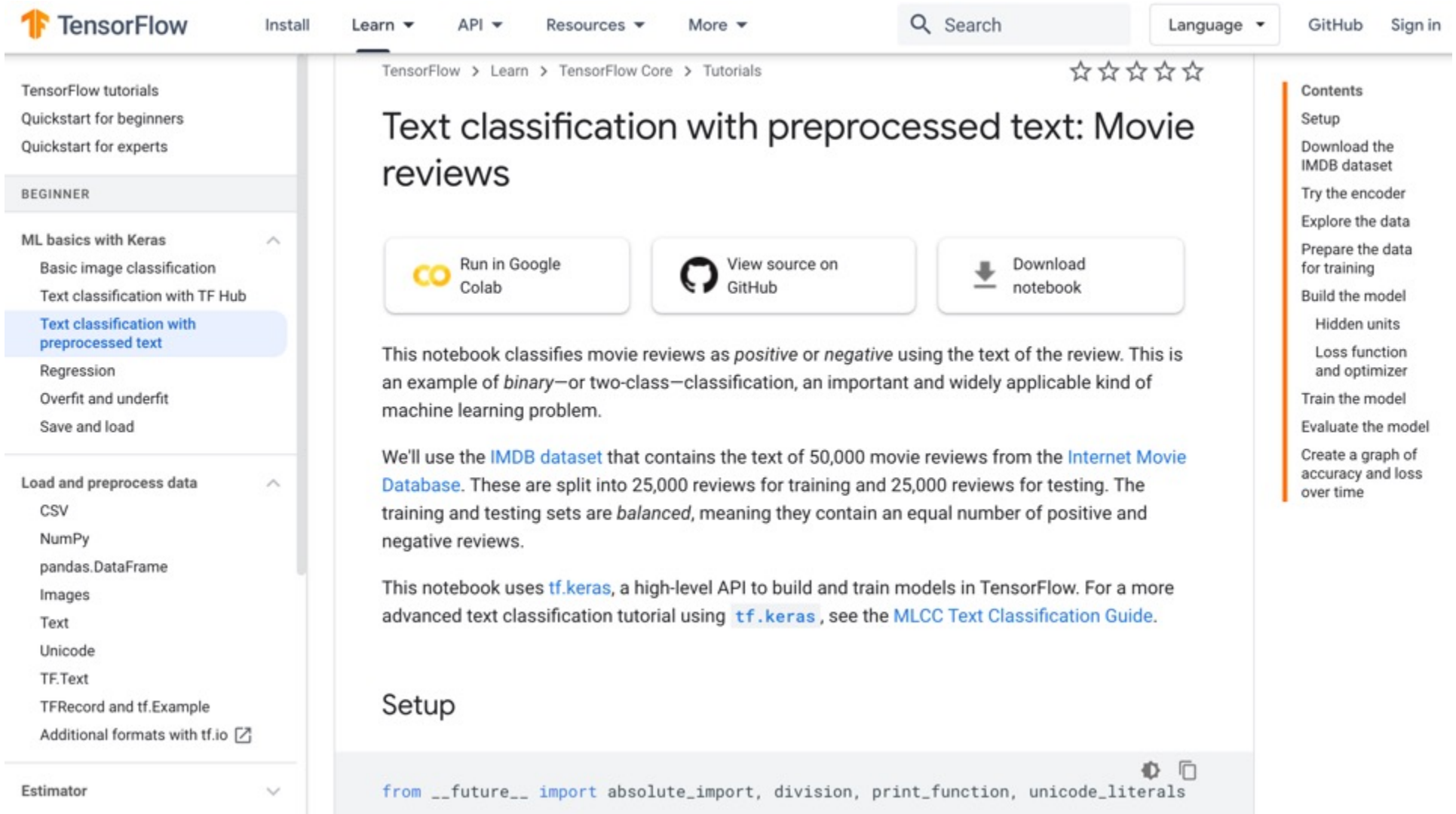
Text Classification with TF Hub



https://www.tensorflow.org/tutorials/keras/text_classification_with_hub

TensorFlow

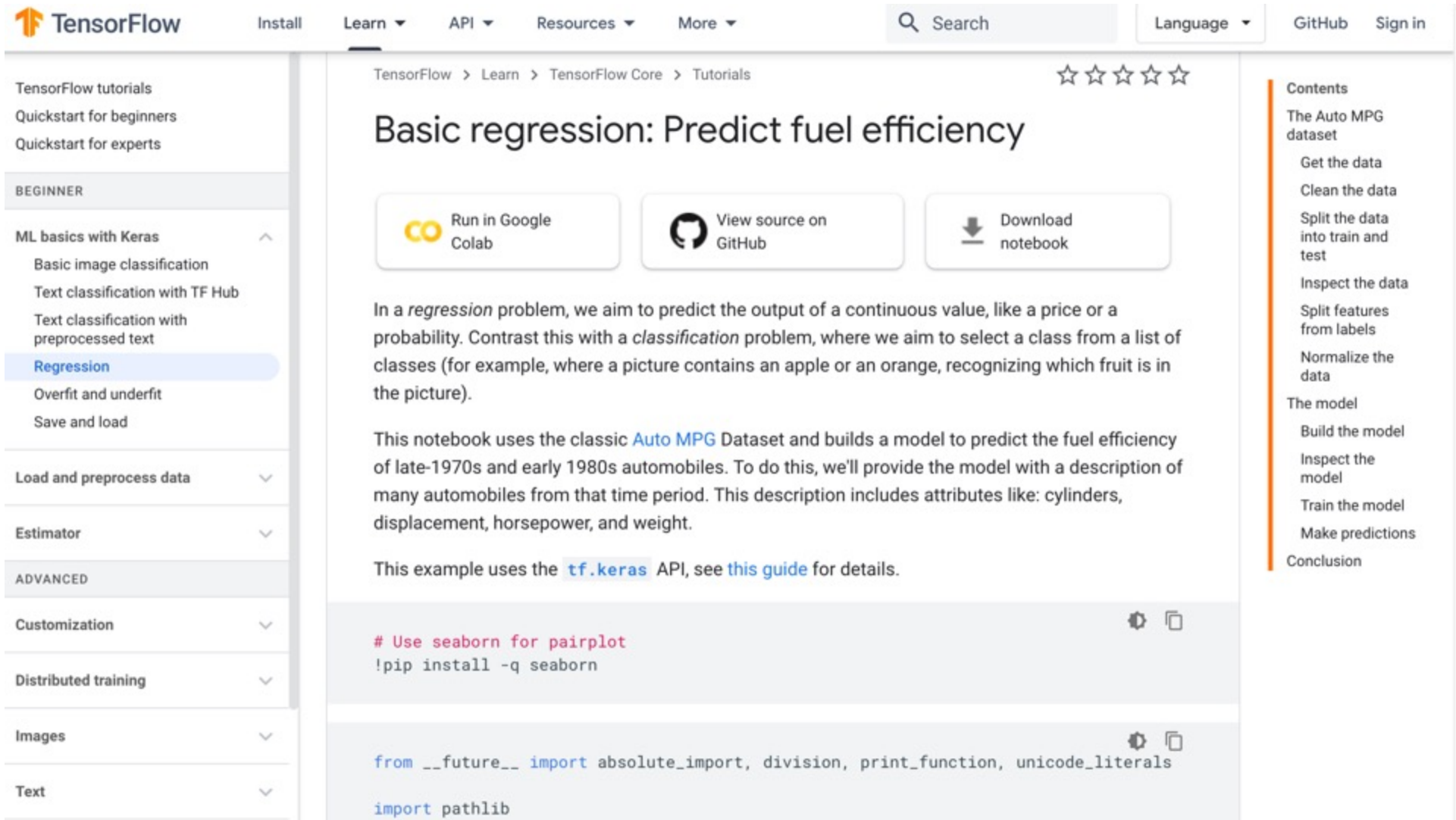
Text Classification with Pre Text



https://www.tensorflow.org/tutorials/keras/text_classification

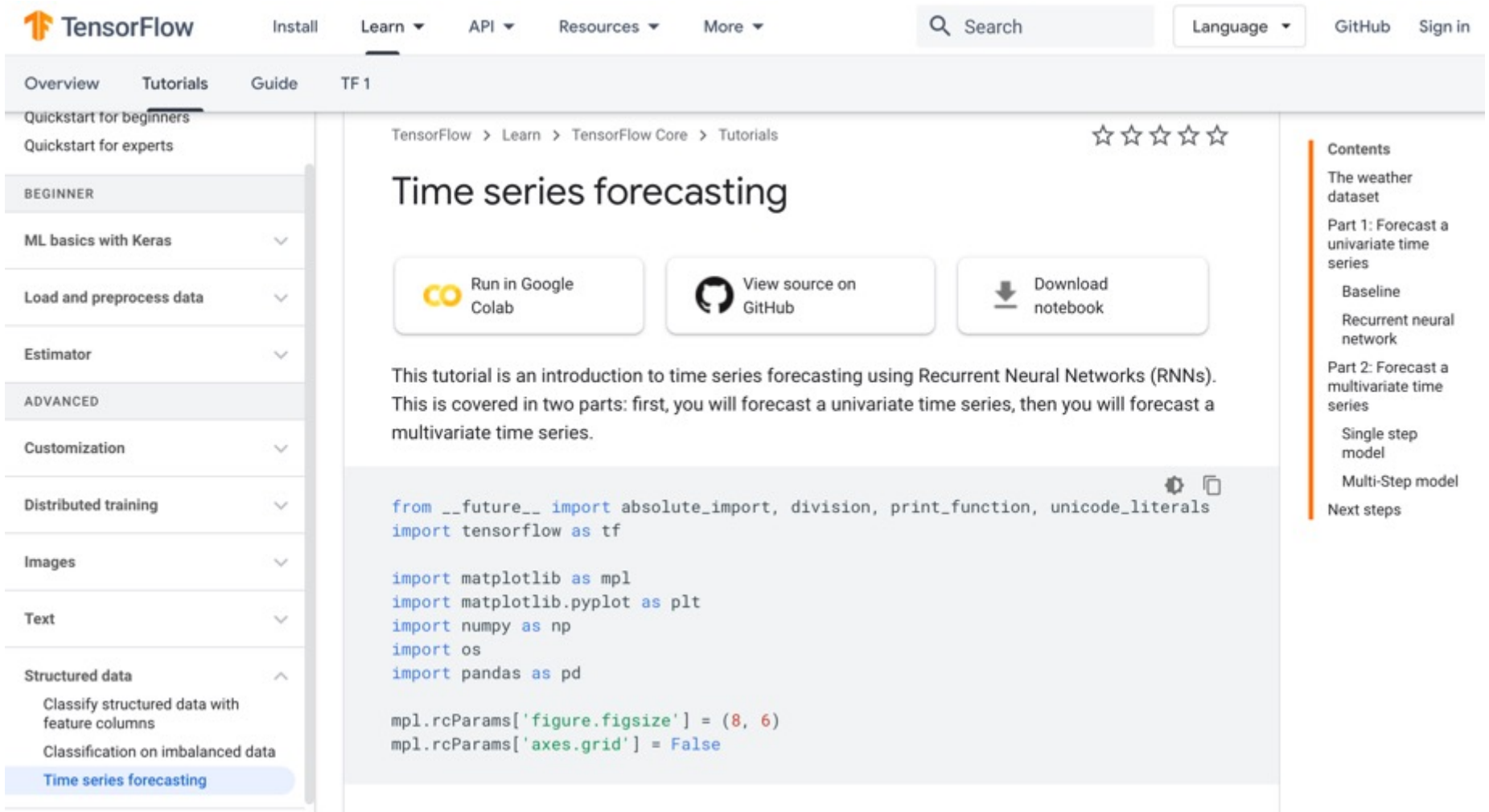
TensorFlow

Regression



https://www.tensorflow.org/tutorials/keras/regression

TensorFlow 2.0
Time Series Forecasting

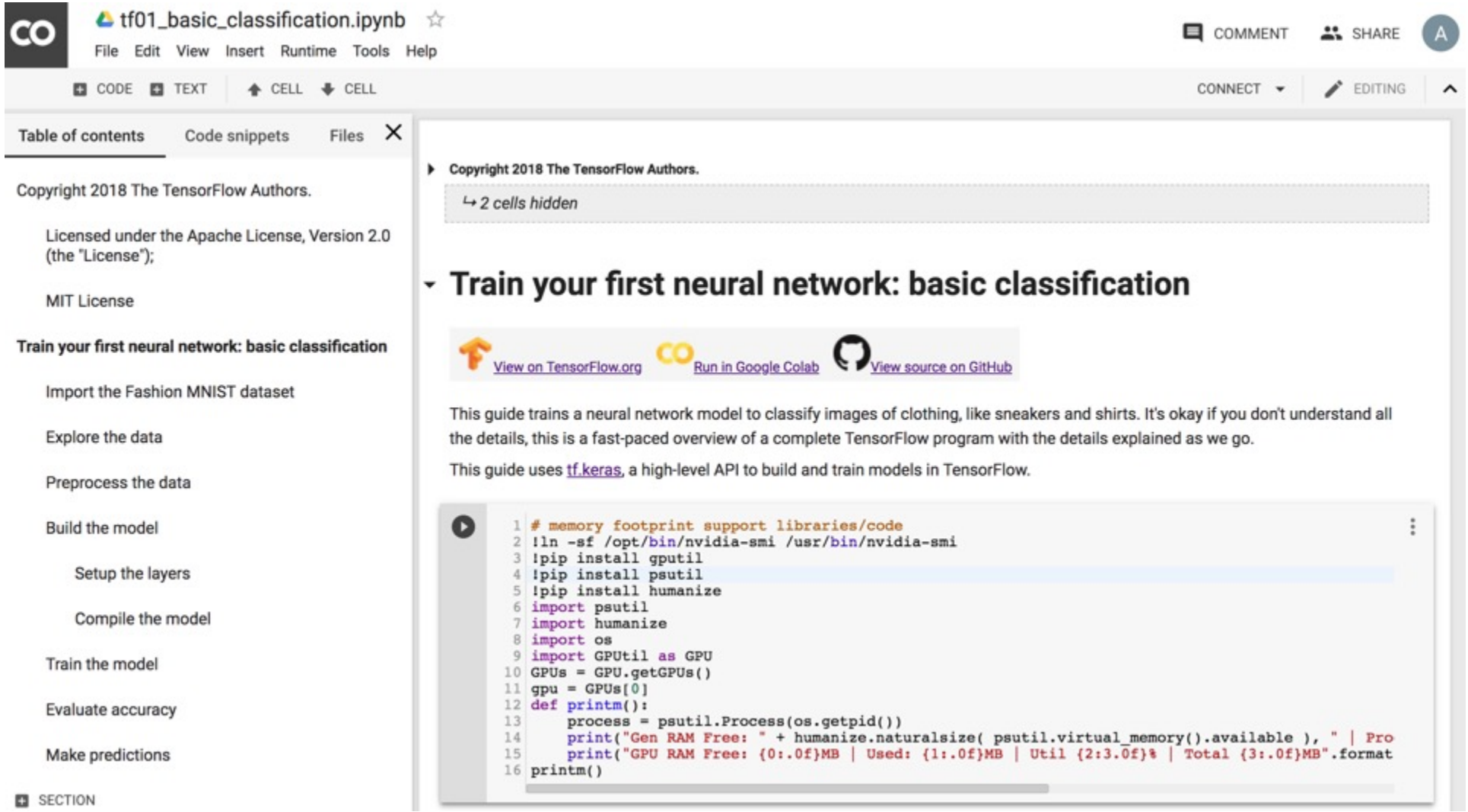


https://www.tensorflow.org/tutorials/structured_data/time_series

Basic Classification

Fashion MNIST Image Classification

https://colab.research.google.com/drive/19PJOJi1vn1kjcutlzNHjRSLbeVl4kd5z

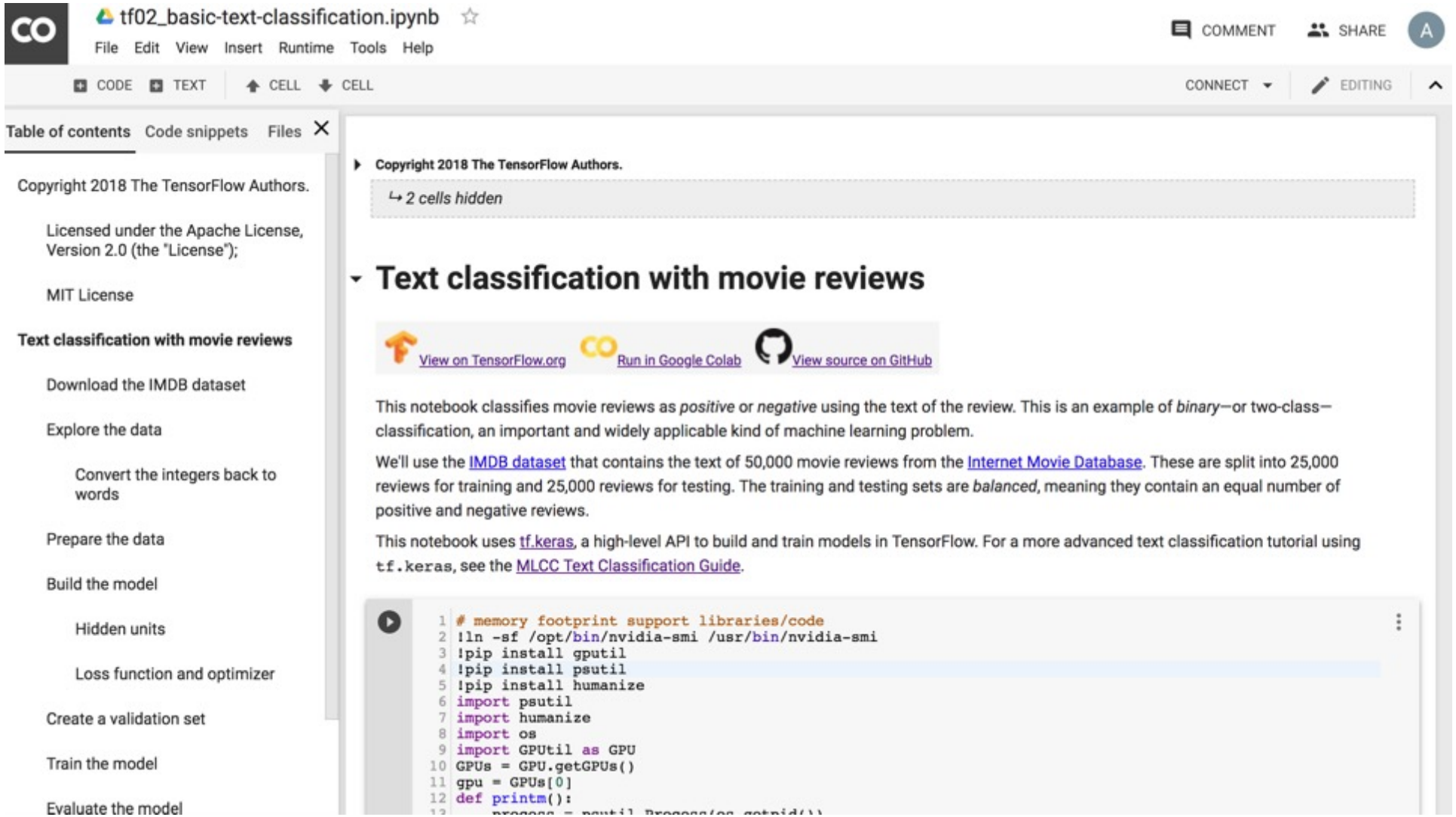


Source: https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/keras/basic_classification.ipynb

Text Classification

IMDB Movie Reviews

https://colab.research.google.com/drive/1x16h1GhHsLIrLYtPCvCHaoO1W-i_gror

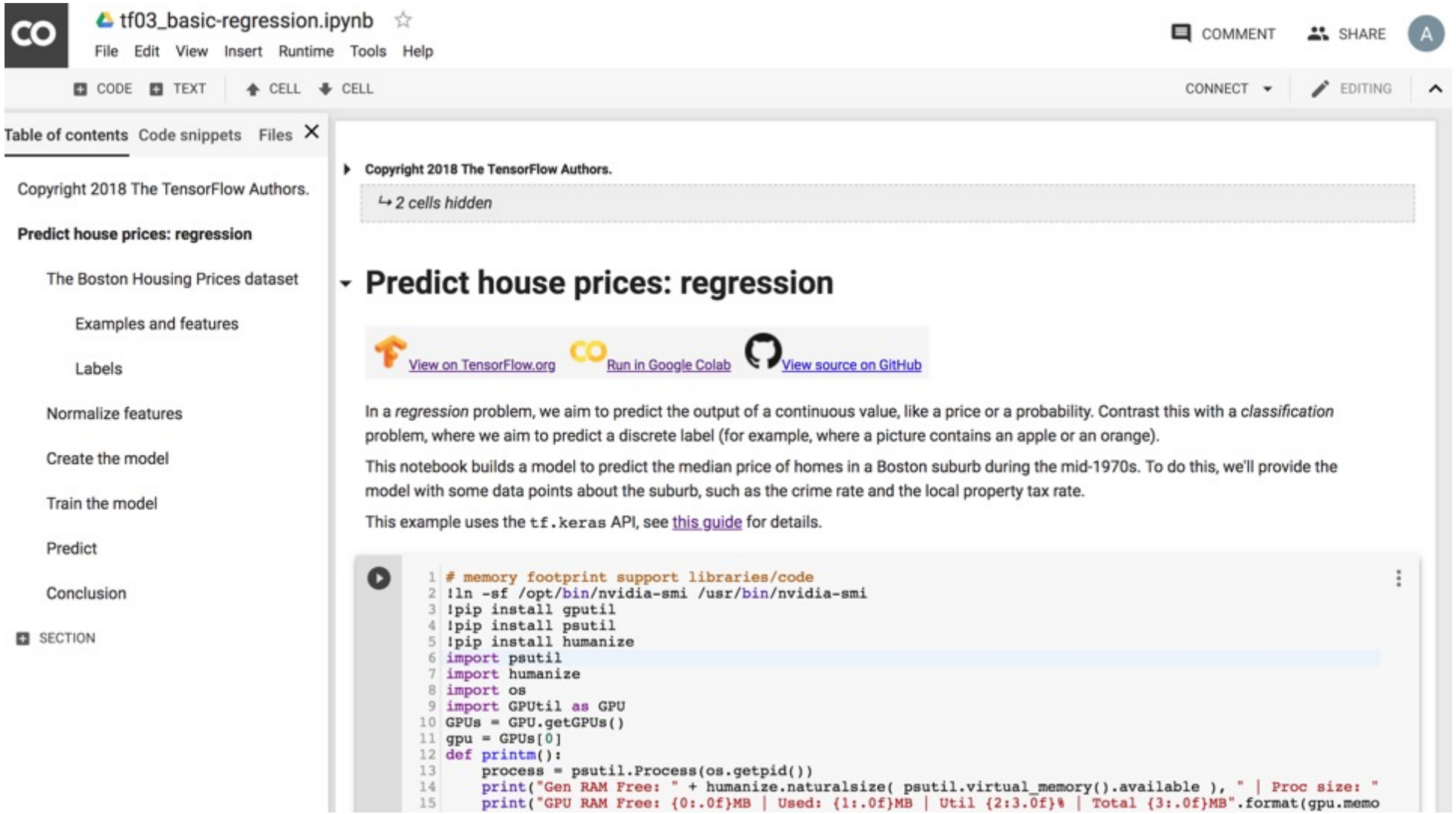


Source: https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/keras/basic_text_classification.ipynb

Basic Regression

Predict House Prices

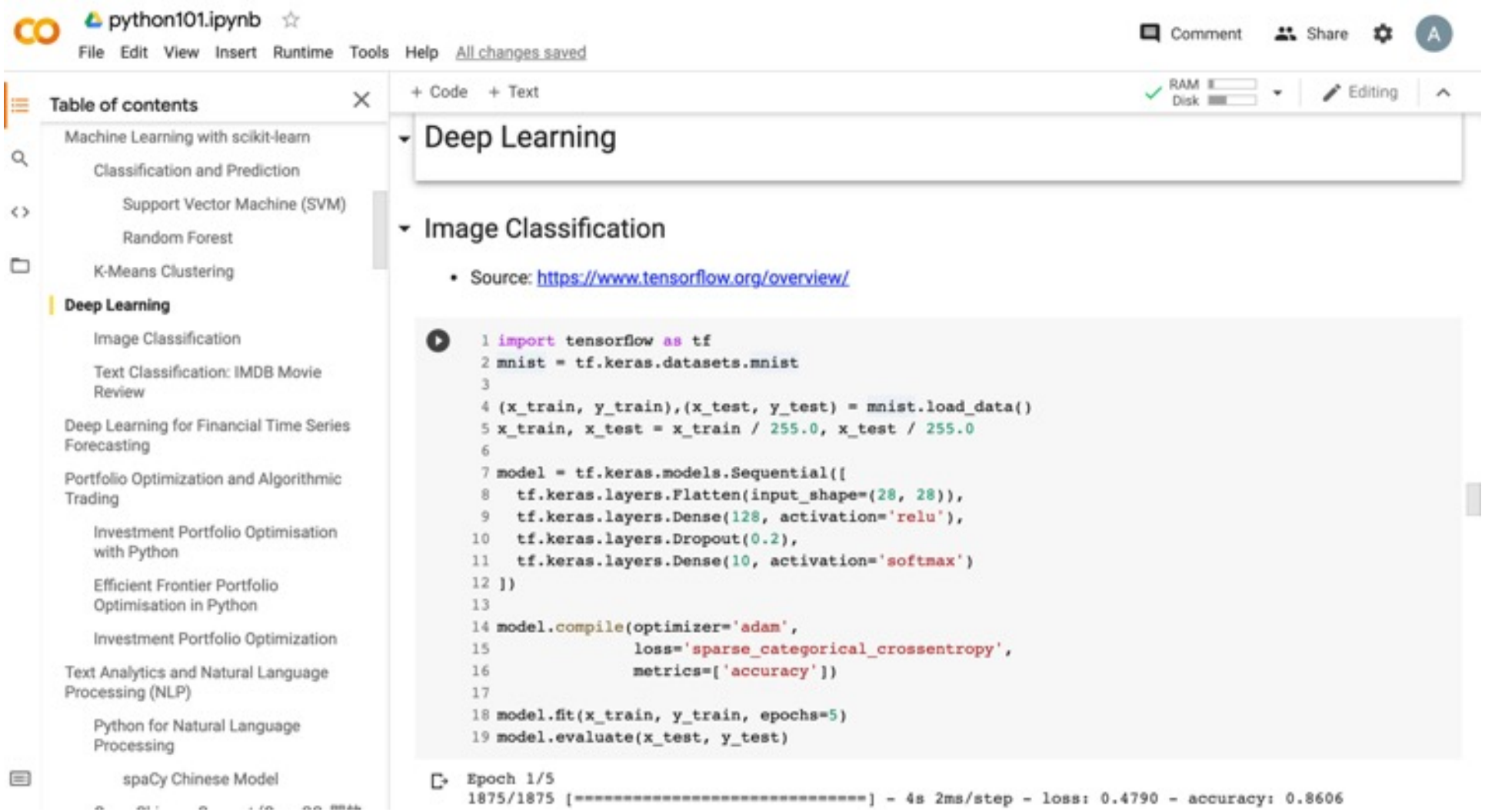
https://colab.research.google.com/drive/1v4c8ZHTnRtgd2_25K_AURjR6SCVBRdlj



Source: https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/keras/basic_regression.ipynb

Python in Google Colab (Python101)

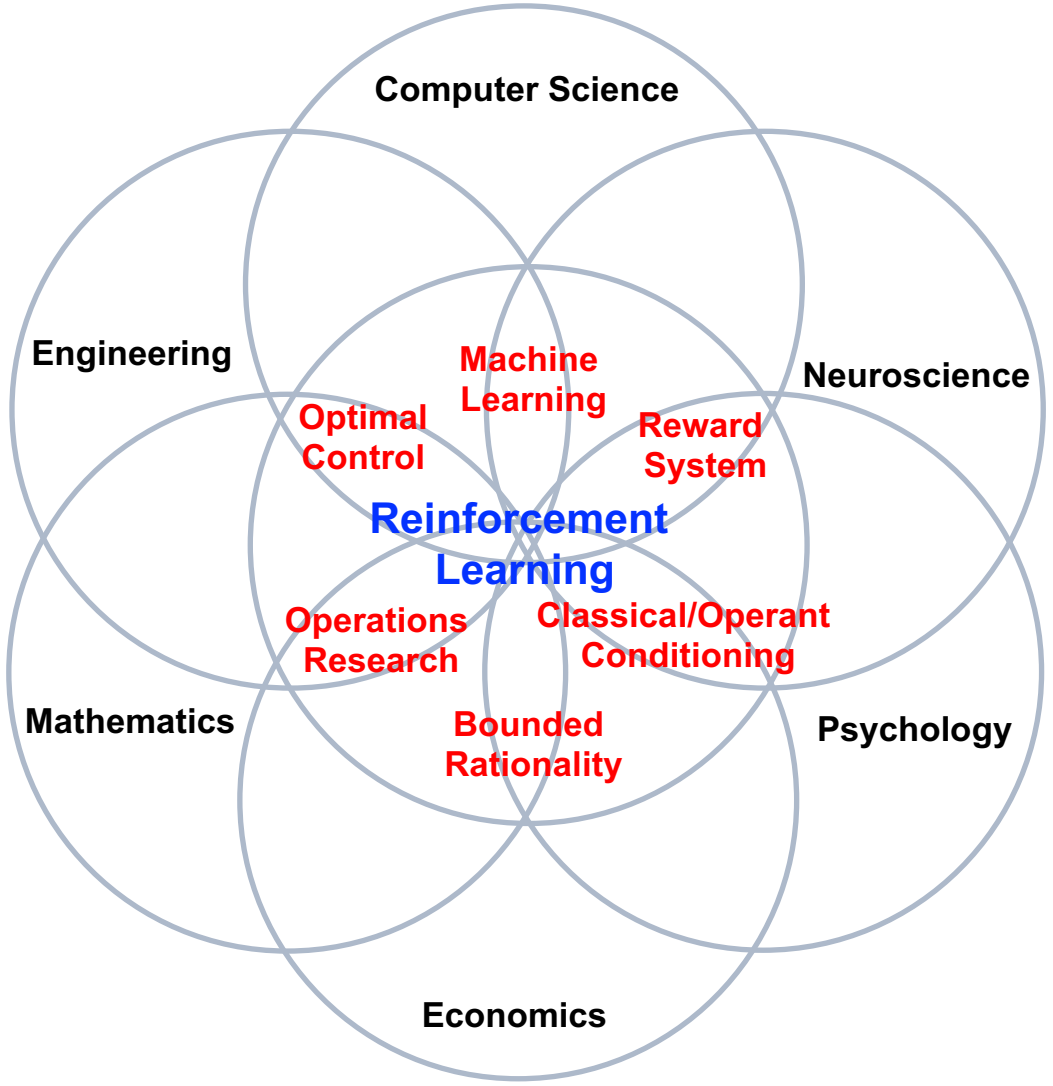
https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT



https://tinyurl.com/aintpupython101

Reinforcement Learning

Reinforcement Learning (RL)

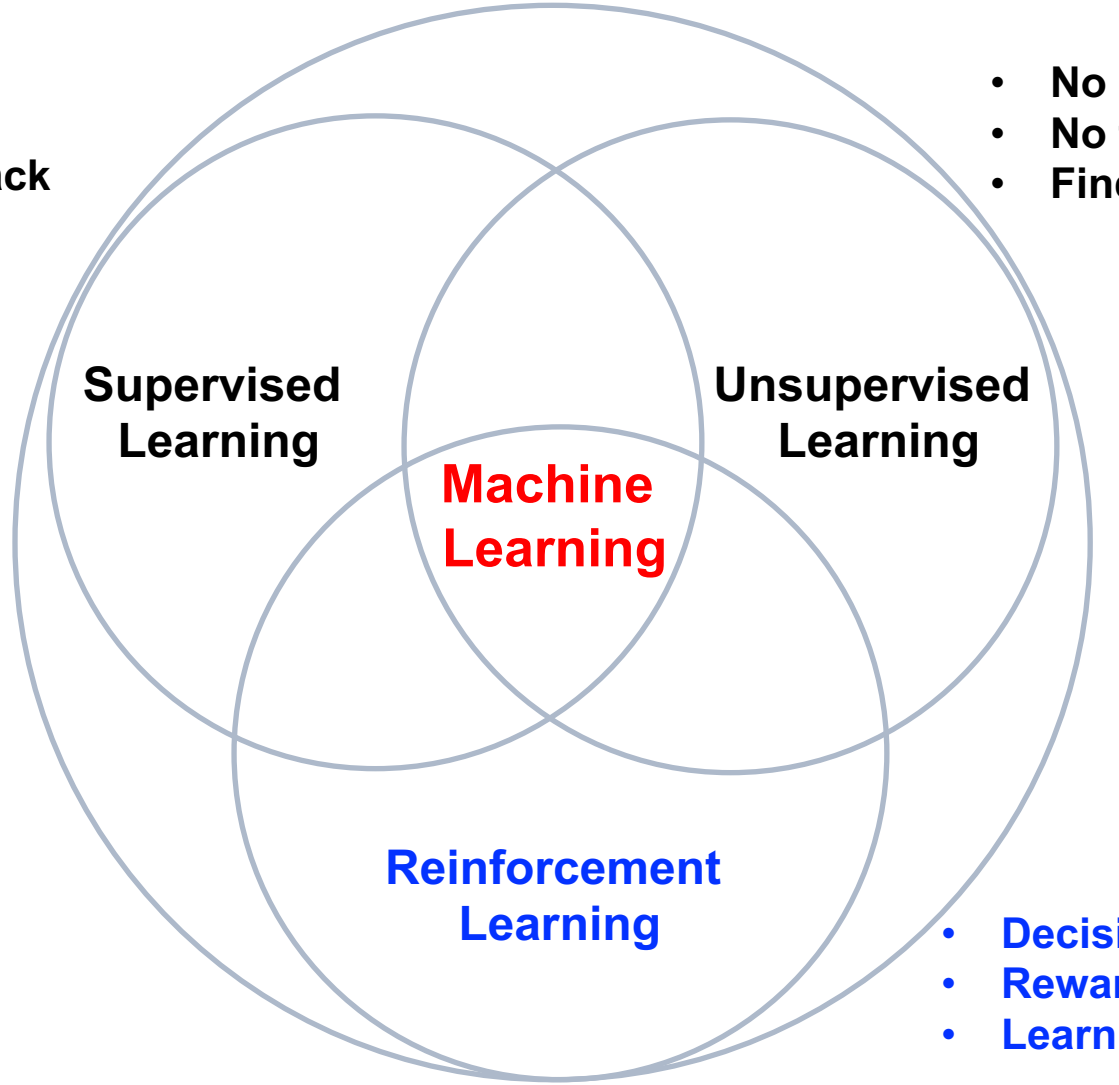


Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Branches of Machine Learning (ML)
Reinforcement Learning (RL)

- Labeled data
- Direct feedback
- Predict

- No Labels
- No feedback
- Find hidden structure



- Decision process
- Reward system
- Learn series of actions

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

David Silver (2015), Introduction to reinforcement learning

- Elementary Reinforcement Learning
  - 1: Introduction to Reinforcement Learning
  - 2: Markov Decision Processes
  - 3: Planning by Dynamic Programming
  - 4: Model-Free Prediction
  - 5: Model-Free Control
- Reinforcement Learning in Practice
  - 6: Value Function Approximation
  - 7: Policy Gradient Methods
  - 8: Integrating Learning and Planning
  - 9: Exploration and Exploitation
  - 10: Case Study: RL in Classic Games

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning
AlphaZero (AZ) and AlphaGo Zero (AZ0)

- **AlphaZero (Silver et al., 2018)**
  - **A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. (Science)**
- **AlphaGo Zero (Silver et al., 2017)**
  - **Mastering the game of Go without human knowledge (Nature)**

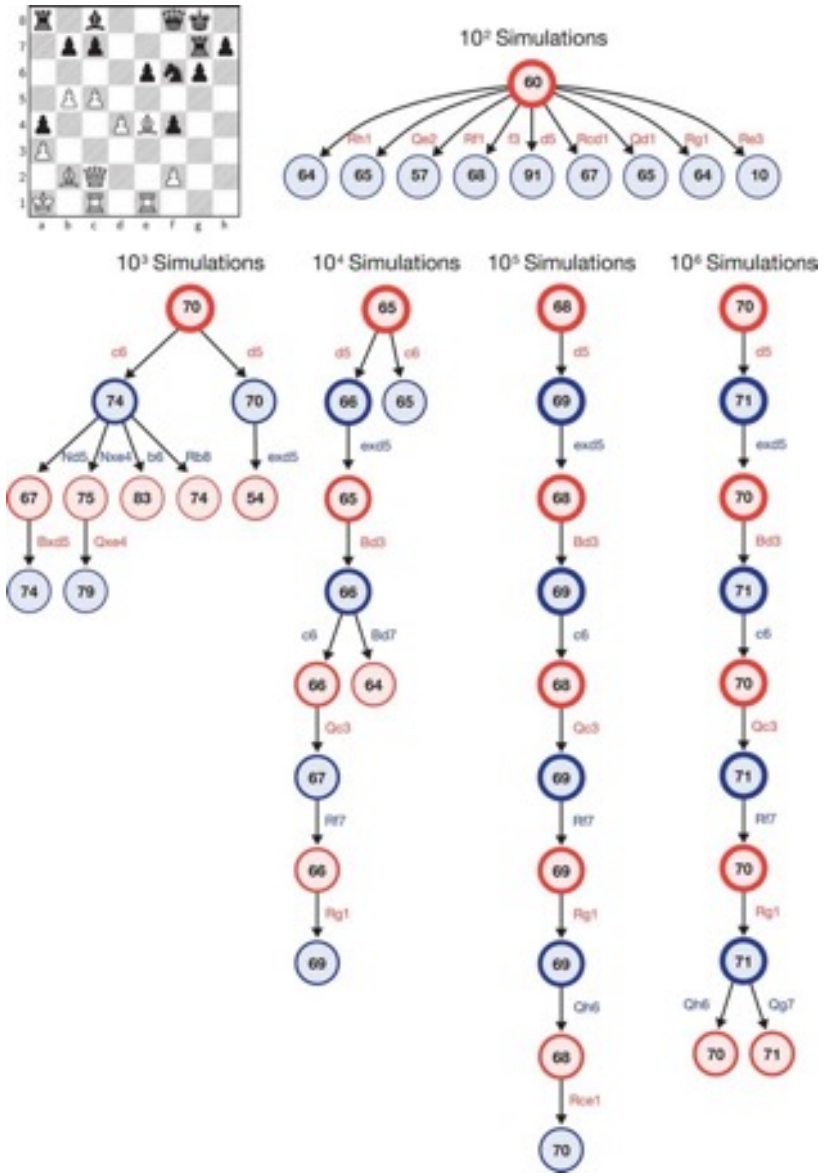
AlphaZero:
Shedding new light on the grand games of chess, shogi and Go

https://www.youtube.com/watch?v=7L2sUGcOgh0

AlphaZero
A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play

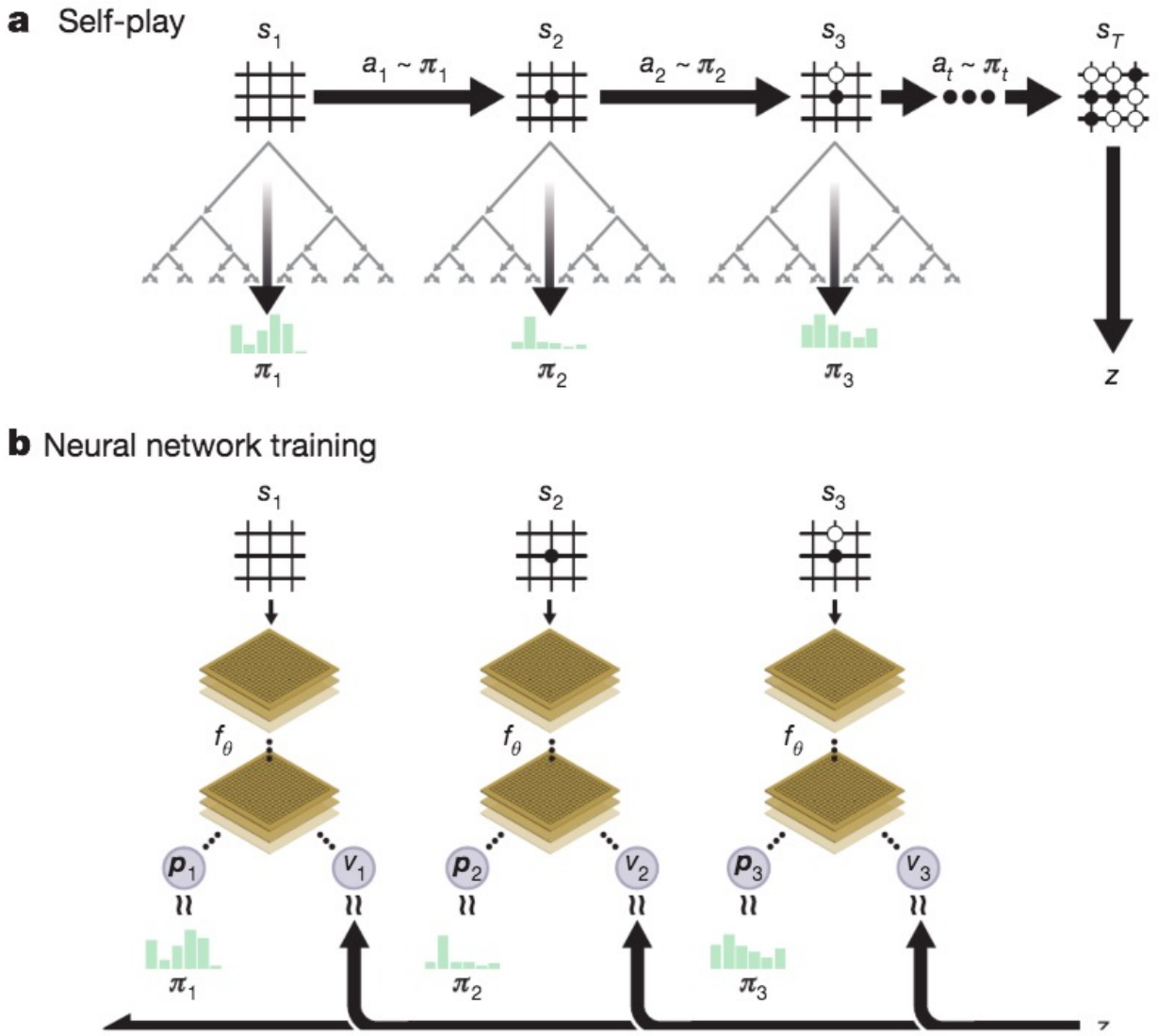
Source: David Silver et al. (2018), "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play." Science 362, no. 6419 (2018): 1140-1144.

AlphaZero's search procedure



Source: David Silver et al. (2018), "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play." Science 362, no. 6419 (2018): 1140-1144.

Self-play reinforcement learning in AlphaGo Zero

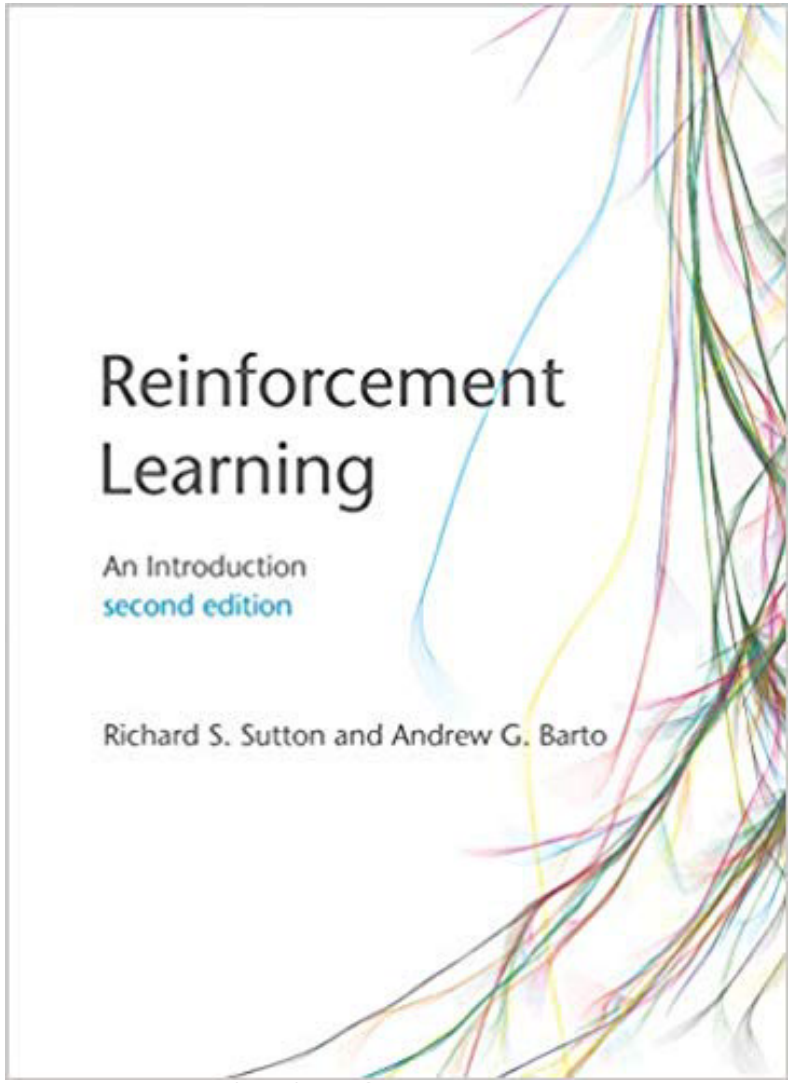


Source: David Silver et al. (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.

Richard S. Sutton & Andrew G. Barto (2018),

Reinforcement Learning: An Introduction,

2nd Edition, A Bradford Book



Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.
https://www.amazon.com/Reinforcement-Learning-Introduction-Adaptive-Computation/dp/0262039249

Reinforcement learning

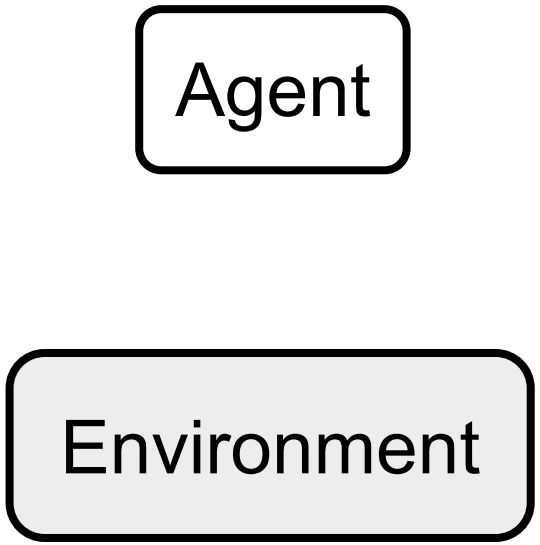
- **Reinforcement learning is**
  learning what to do
  —how to map situations to actions
  —so as to maximize a numerical reward signal.

Two most important distinguishing features of reinforcement learning

- **trial-and-error search**
- **delayed reward**

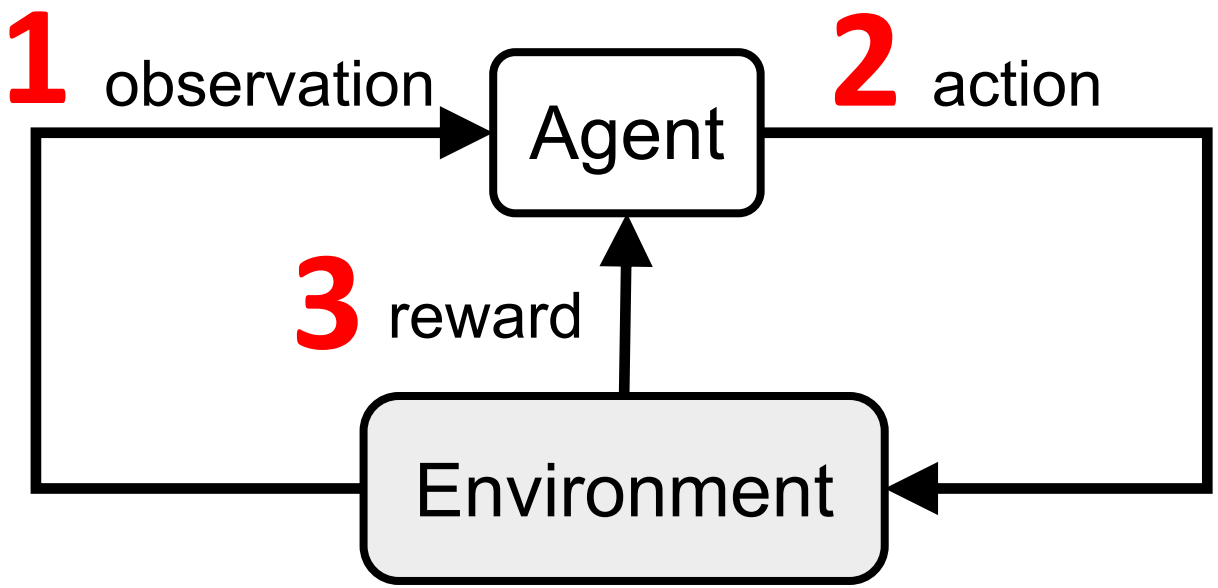
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning (DL)



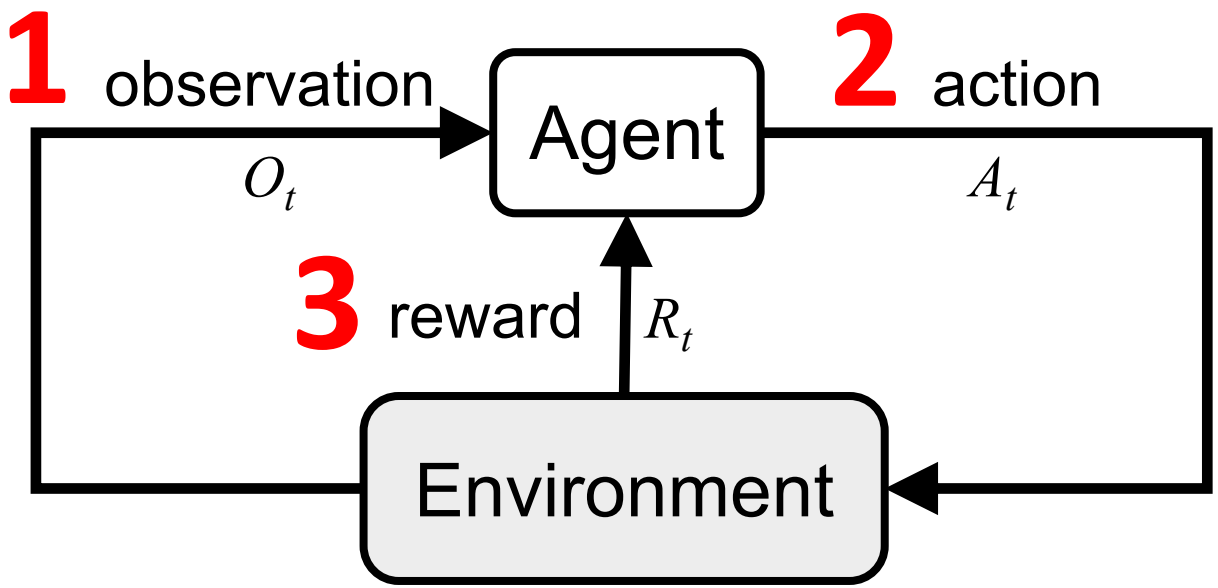
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning (DL)



Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

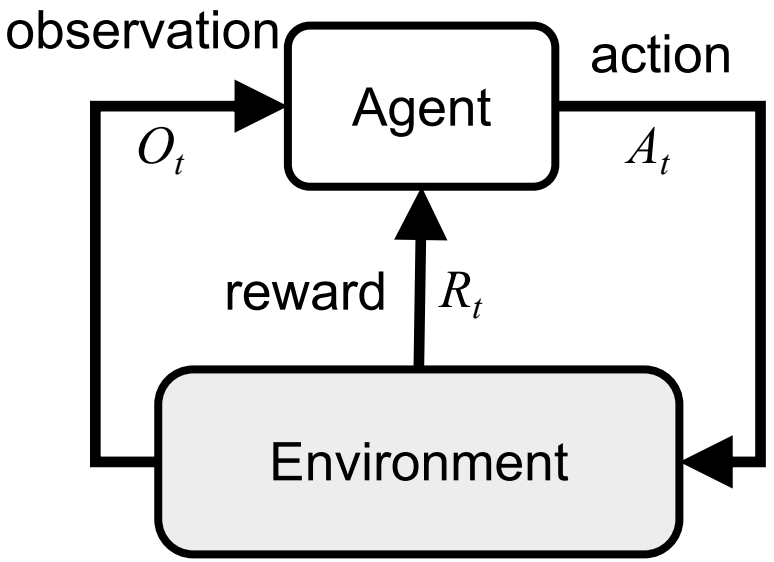
Reinforcement Learning (DL)



Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Agent and Environment

- At each step *t* the agent:
  - Executes action A_t
  - Receives observation O_t
  - Receives scalar reward R_t
- The environment:
  - Receives action A_t
  - Emits observation O_{t+1}
  - Emits scalar reward R_{t+1}
- t increments at env. step



Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

History and State

- The history is the sequence of observations, actions, rewards

$$H_t = O_1, A_1, R_1, \ldots, A_{t-1}, O_t, R_t$$

- i.e. all observable variables up to time t
- i.e. the sensorimotor stream of a robot or embodied agent
- What happens next depends on the history:
  - The agent selects actions
  - The environment selects observations/rewards
- State is the information used to determine what happens next
- Formally, state is a function of the history:

$$S_t = f(H_t)$$

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Information State

- **An information state (a.k.a. Markov state) contains all useful information from the history.**
- **Definition**

  A state S_t is Markov if and only if

  $$P[S_{t+1} \mid S_t] = P[S_{t+1} \mid S_1, \ldots, S_t]$$

- **"The future is independent of the past given the present"**

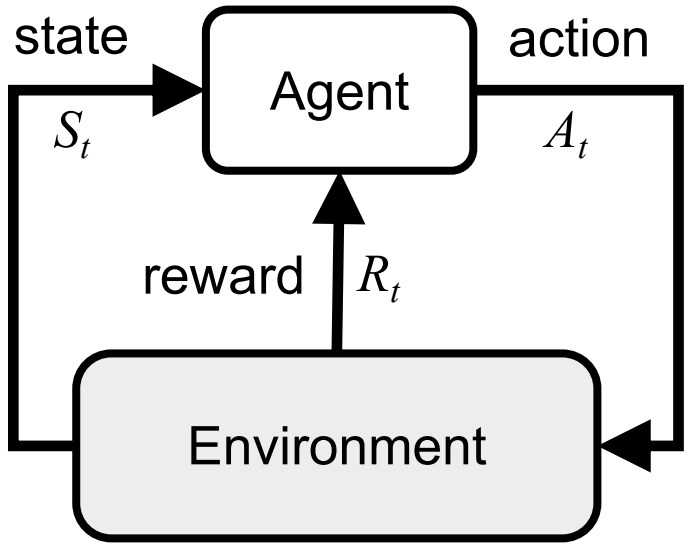
  $$H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty}$$

- **Once the state is known, the history may be thrown away i.e. The state is a sufficient statistic of the future**
- **The environment state S_t^e is Markov**
- **The history H_t is Markov**

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Fully Observable Environments

- **Full observability:**
  - **agent directly observes environment state**
  - **Agent state = environment state = information state**
  - **Formally, this is a Markov decision process (MDP)**



Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

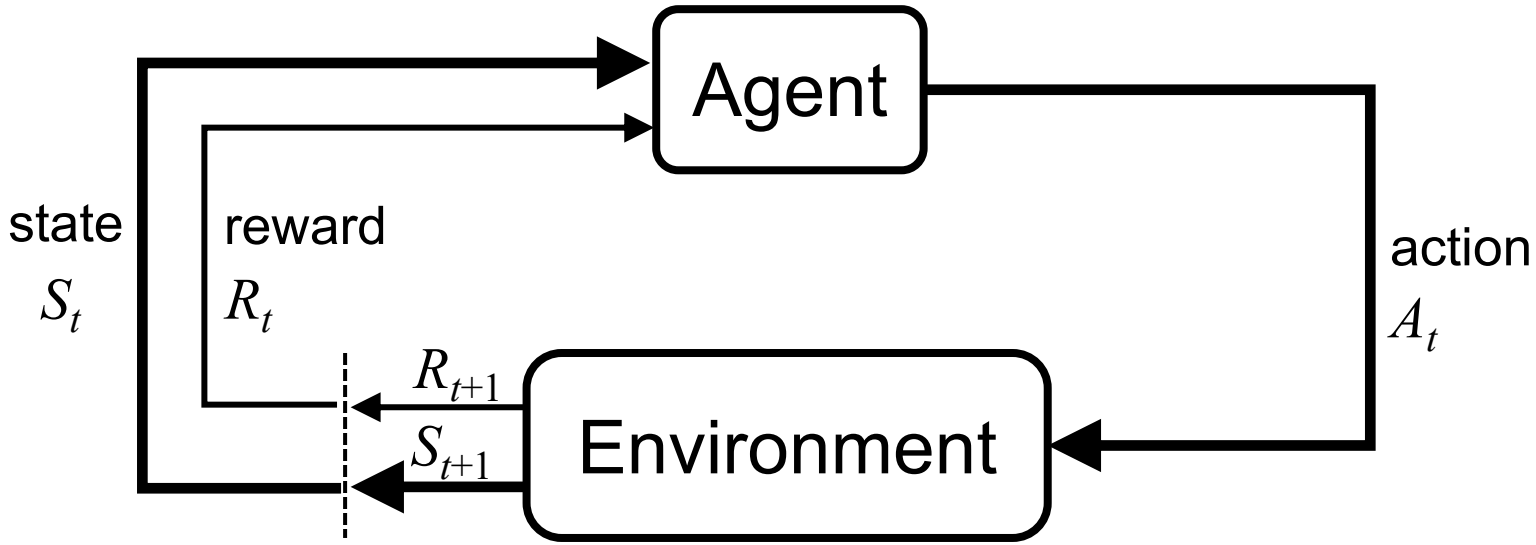
Partially Observable Environments

- **Partial observability**: agent **indirectly** observes environment
  - A robot with camera vision isn't told its absolute location
  - A trading agent only observes current prices
  - A poker playing agent only observes public cards
- Now agent state ≠ environment state
- Formally this is a **partially observable Markov decision process (POMDP)**
- Agent must construct its own state representation S^a_t, e.g.
  - Complete history: $S^a_t = H_t$
  - **Beliefs** of environment state: $S^a_t = (P[S^e_t = s_1],\ldots,P[S^e_t = s_n])$
  - Recurrent neural network: $S^a_t = \sigma(S^a_{t-1} W_s + O_t W_o)$

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning (DL)

The Agent-Environment Interaction in a Markov Decision Process (MDP)



Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Characteristics of Reinforcement Learning

- No supervisor, only a reward signal
- Feedback is delayed, not instantaneous
- Time really matters (sequential, non i.i.d data)
- Agent's actions affect the subsequent data it receives

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Examples of Reinforcement Learning

- **Make a humanoid robot walk**
- **Play may different Atari games better than humans**
- **Manage an investment portfolio**

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Examples of Rewards

- Make a humanoid robot walk
  - +ve reward for forward motion
  - -ve reward for falling over
- Play may different Atari games better than humans
  - +/-ve reward for increasing/decreasing score
- Manage an investment portfolio
  - +ve reward for each $ in bank

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Sequential Decision Making

- Goal: select actions to maximize total future reward
- Actions may have long term consequence
- Reward may be delayed
- It may be better to sacrifice immediate reward to gain more long-term reward
- Examples:
  - A financial investment (may take months to mature)
  - Blocking opponent moves (might help winning chances many moves from now)

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Elements of Reinforcement Learning

- Agent
- Environment
- Policy
- Reward signal
- Value function
- Model

Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Elements of Reinforcement Learning

- Policy
  - Agent's behavior
  - It is a map from state to action
- Reward signal
  - The goal of a reinforcement learning problem
- Value function
  - How good is each state and/or action
  - A prediction of future reward
- Model
  - Agent's representation of the environment

Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Major Components of an RL Agent

1. **Policy**: agent's behaviour function
2. **Value** function: how good is each state and/or action
3. **Model**: agent's representation of the environment

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Policy

- A **policy** is the agent's behaviour
- It is a map from state to action, e.g.
  - Deterministic policy: $a = \pi(s)$
  - Stochastic policy: $\pi(a|s) = P[A_t = a|S_t = s]$

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Value Function

- **Value function is a prediction of future reward**
- **Used to evaluate the goodness/badness of states**
- **And therefore to select between actions, e.g.**

$$v_\pi(s) = E_\pi [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s]$$

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Model

- A **model** predicts what the environment will do next
- P predicts the next state
- R predicts the next (immediate) reward, e.g.

$$P^{a}_{ss'} = P[S_{t+1} = s' \mid S_{t+1} = s, A_t = a]$$

$$R^{a}_{s} = E[R_{t+1} \mid S_t = s, A_t = a]$$

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning

- **Value Based**
  - **No Policy (Implicit)**
  - **Value Function**
- **Policy Based**
  - **Policy**
  - **No Value Function**
- **Actor Critic**
  - **Policy**
  - **Value Function**

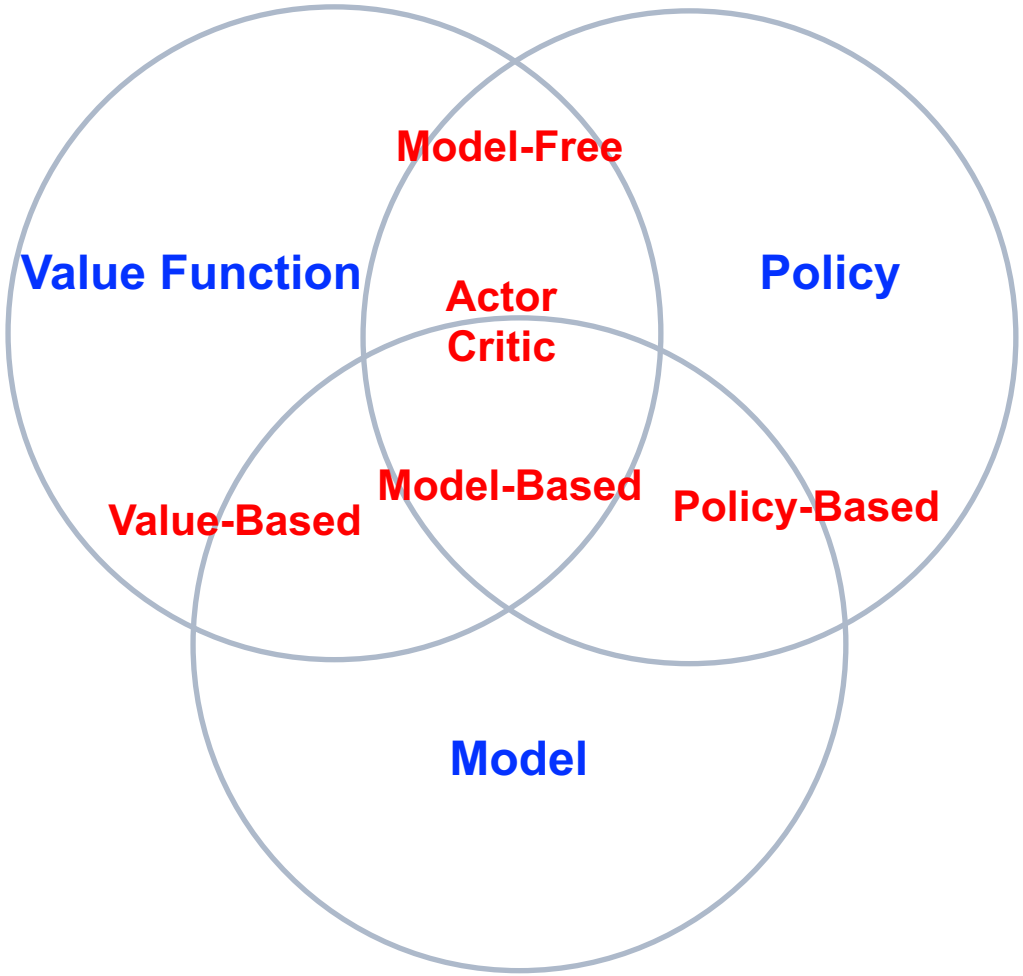
Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning

- Model Free
  - Policy and/or Value Function
  - No Model
- Model Based
  - Policy and/or Value Function
  - Model

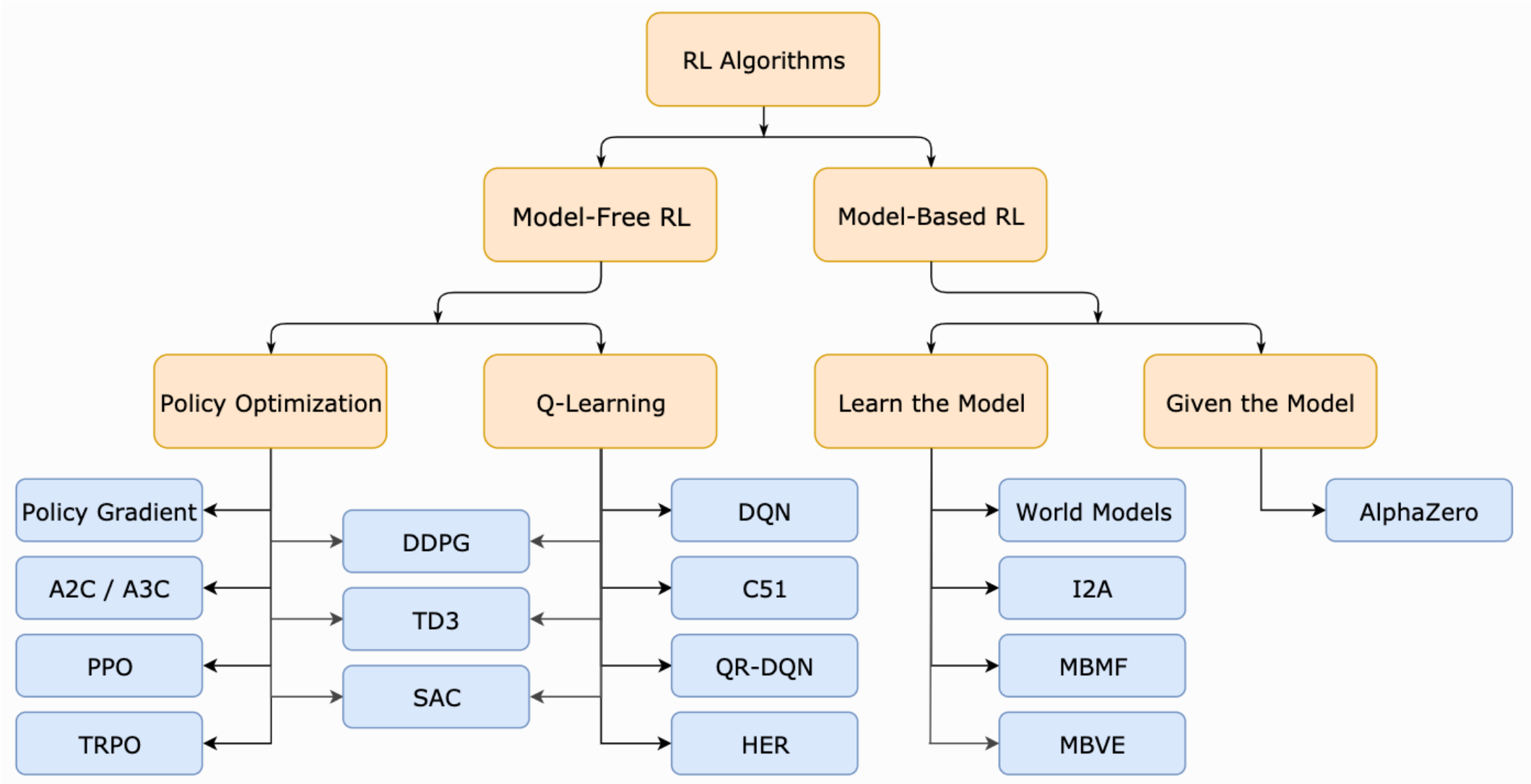
Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning (RL) Taxonomy



Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning (RL)
A Taxonomy of RL Algorithms



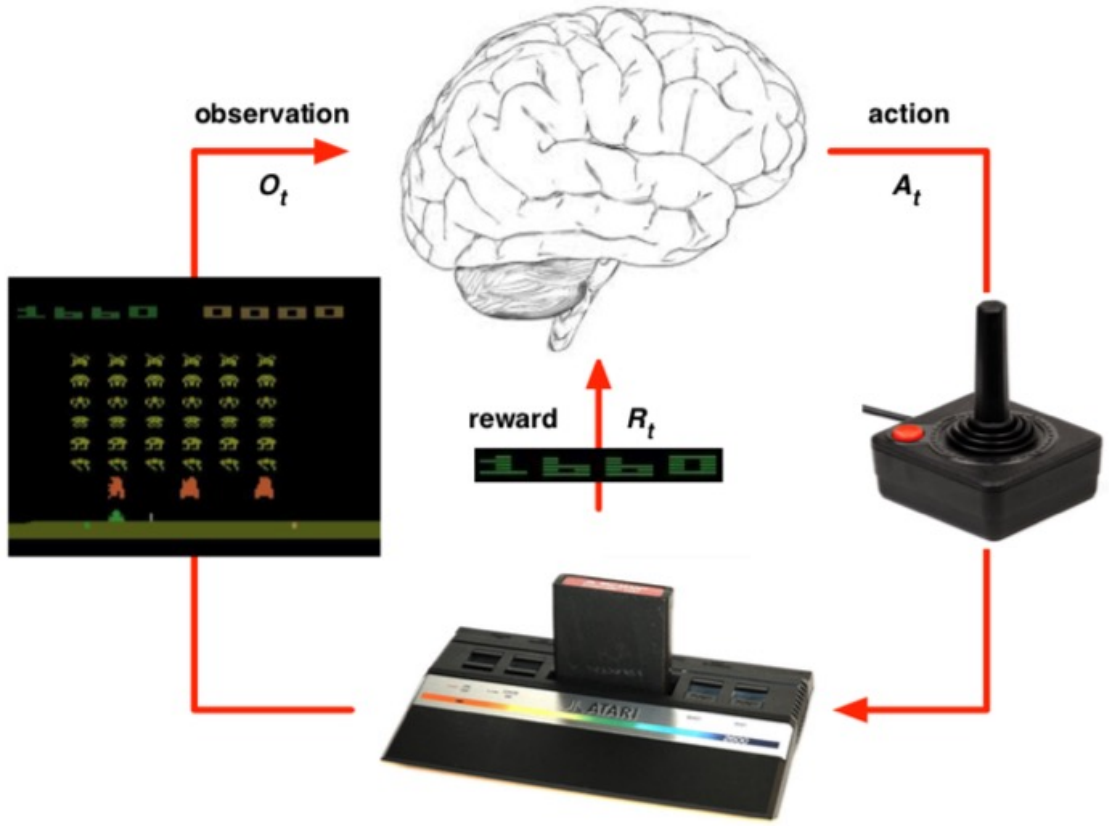
Source: https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

Learning and Planning

- **Two fundamental problems in sequential decision making**
  - **Reinforcement Learning**
    - The environment is initially unknown
    - The agent interacts with environment
    - The agent improves its policy
  - **Planning**
    - A model of the environment is known
    - The agent performs computations with its model (without any external interaction)
    - The agent improves its policy
    - a.k.a deliberation, reasoning, introspection, pondering, thought, search

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Atari Example: Reinforcement Learning

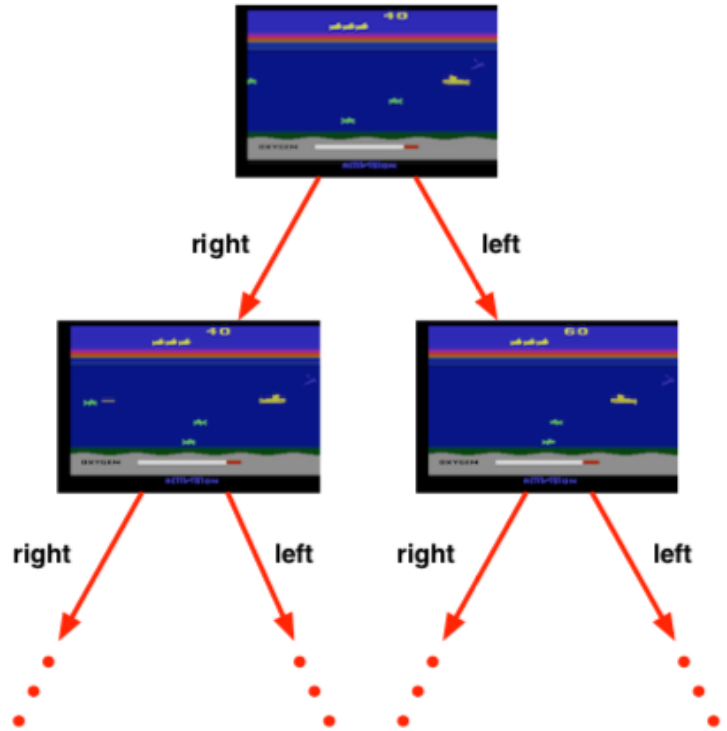


- Rules of the game are unknown
- Learn directly from interactive game-play
- Pick actions on joystick, see pixels and scores

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Atari Example: Planning

- Rules of the game are known
- Can query emulator
  - perfect model inside agent's brain
- If I take action a from state s:
  - what would the next state be?
  - what would the score be?
- Plan ahead to find optimal policy
  - e.g. tree search



Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Exploration and Exploitation

- Reinforcement learning is like trial-and-error learning
- The agent should discover a good policy
- From its experiences of the environment
- Without losing too much reward along the way
- Exploration finds more information about the environment
- Exploitation exploits known information to maximise reward
- It is usually important to explore as well as exploit

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Exploration and Exploitation Examples

- **Restaurant Selection**
  - **Exploitation: Go to your favorite restaurant**
  - **Exploration: Try a new restaurant**
- **Online Banner Advertisements**
  - **Exploitation: Show the most successful advert**
  - **Exploration: Show a different advert**

Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Exploration and Exploitation Examples

- Oil Drilling
  - Exploitation: Drill at the best known location
  - Exploration: Drill at a new location
- Game Playing
  - Exploitation: Play the move you believe is best
  - Exploration: Play an experimental move

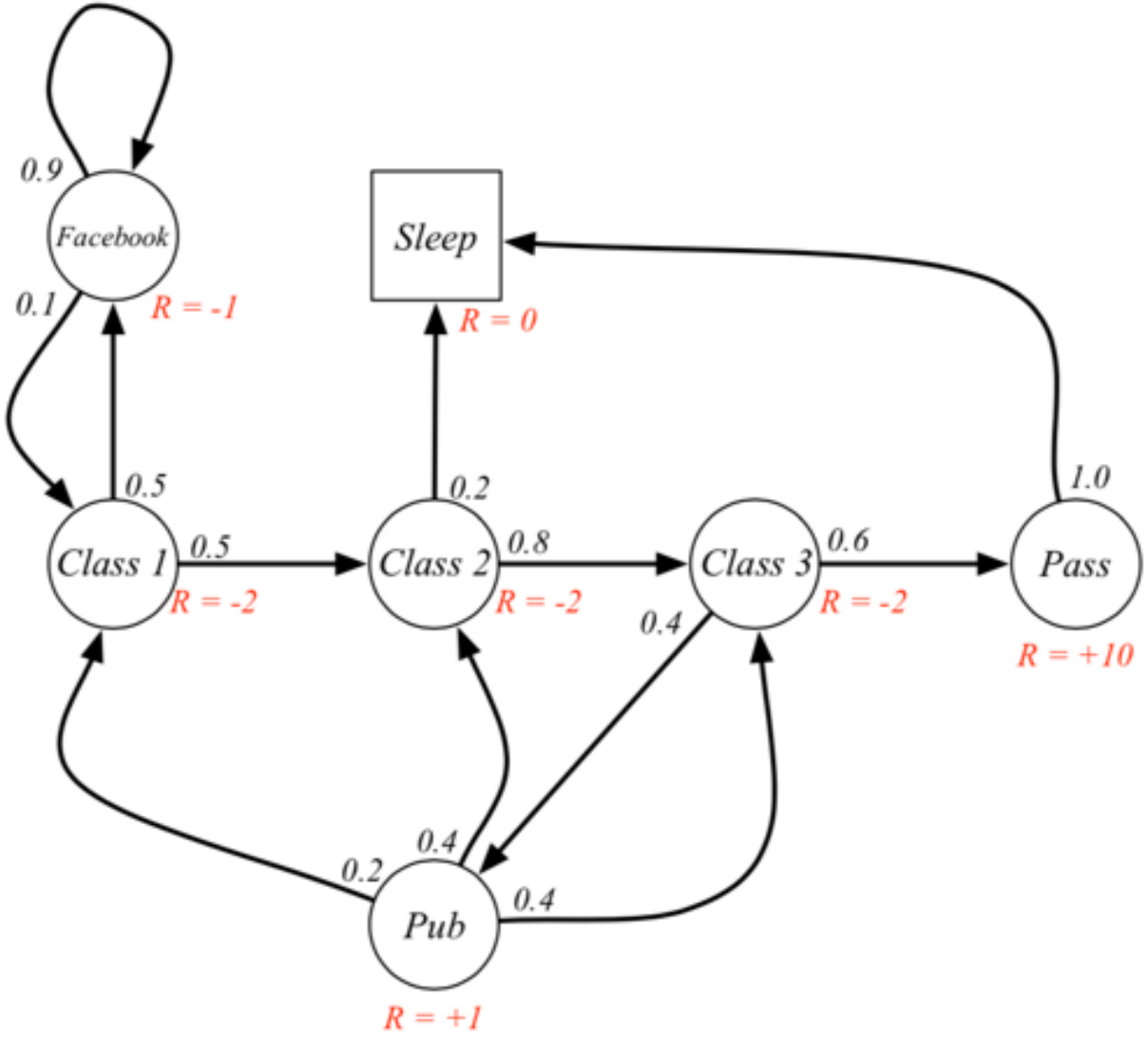
Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Prediction and Control

- **Prediction: evaluate the future**
  - **Given a policy**
- **Control: optimize the future**
  - **Find the best policy**

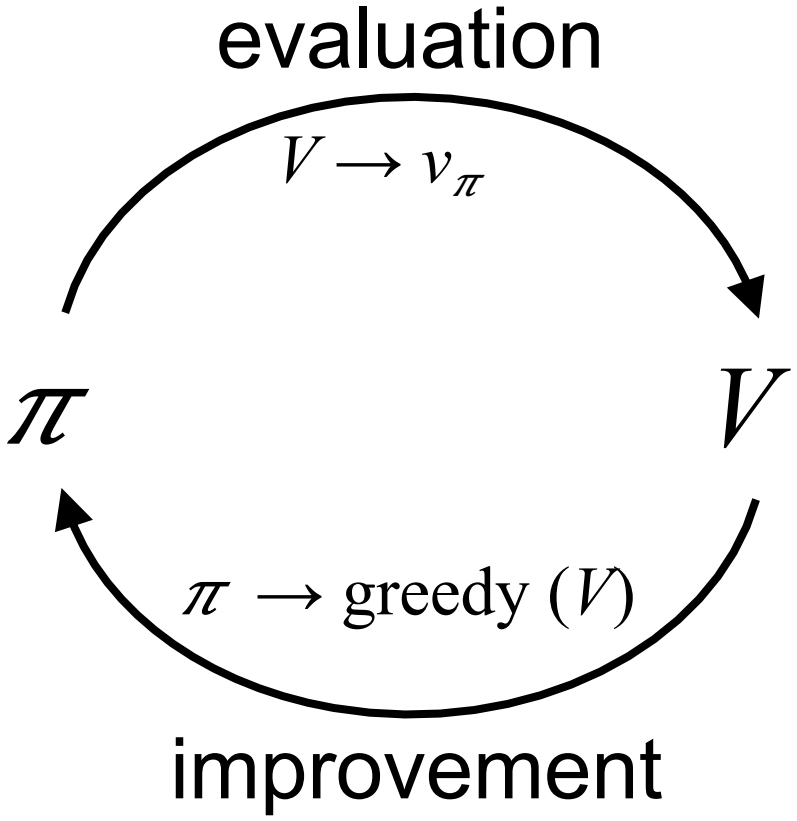
Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

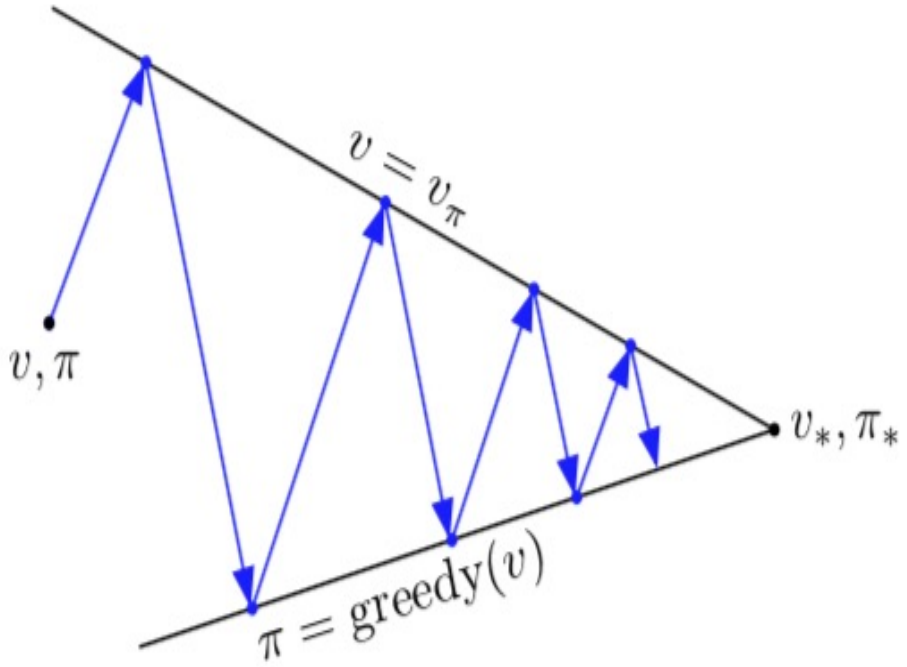
Markov Decision Processes (MDP)
Example: Student MDP



Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Generalized Policy Iteration (GPI)



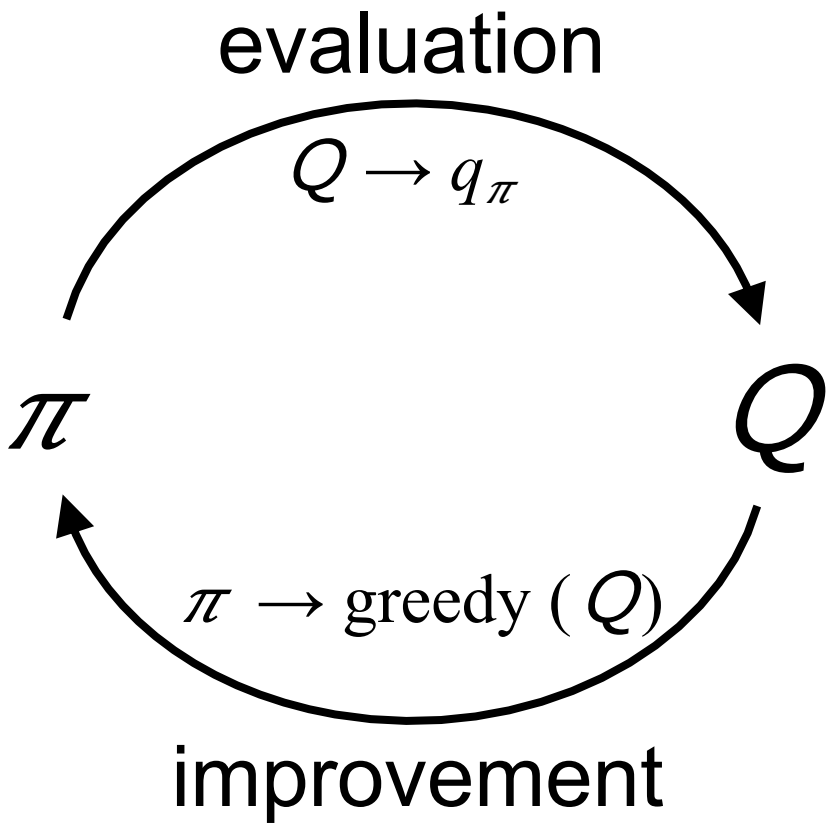


$$\pi_* \rightleftharpoons v_*$$

Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Generalized Policy Iteration (GPI)

Any iteration of **policy evaluation** and **policy improvement**, independent of their granularity.



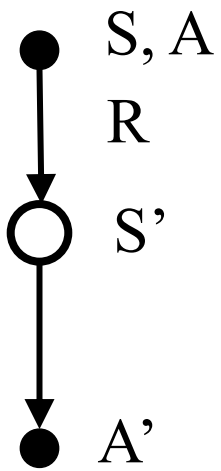
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Temporal-Difference (TD) Learning

- **SARSA: On-policy TD Control**
- **Q-learning: Off-policy TD Control**

SARSA
(state-action-reward-state-action)
On-policy TD Control

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \; Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)]$$

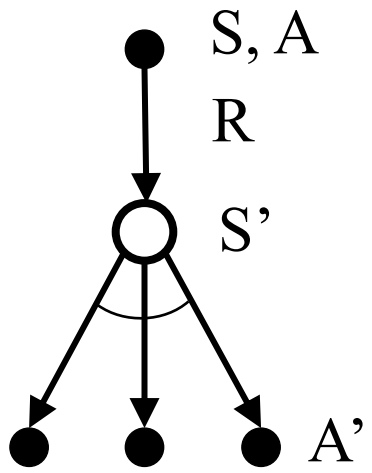


SARSA

Q-learning (Watkins, 1989)

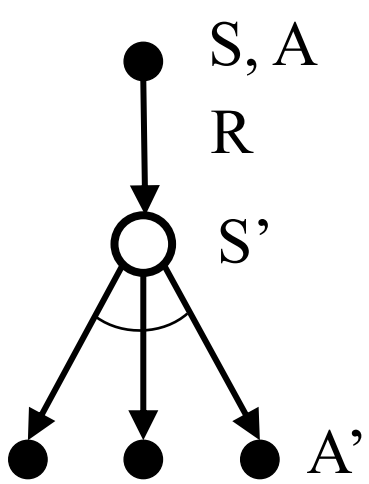
Off-policy TD Control

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max Q(S_{t+1}, a) - Q(S_t, A_t)]$$

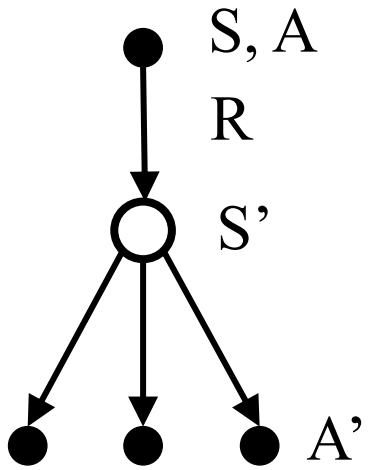


Q-learning

Q-learning and Expected SARSA



Q-learning



Expected SARSA

Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Q-learning and Double Q-learning

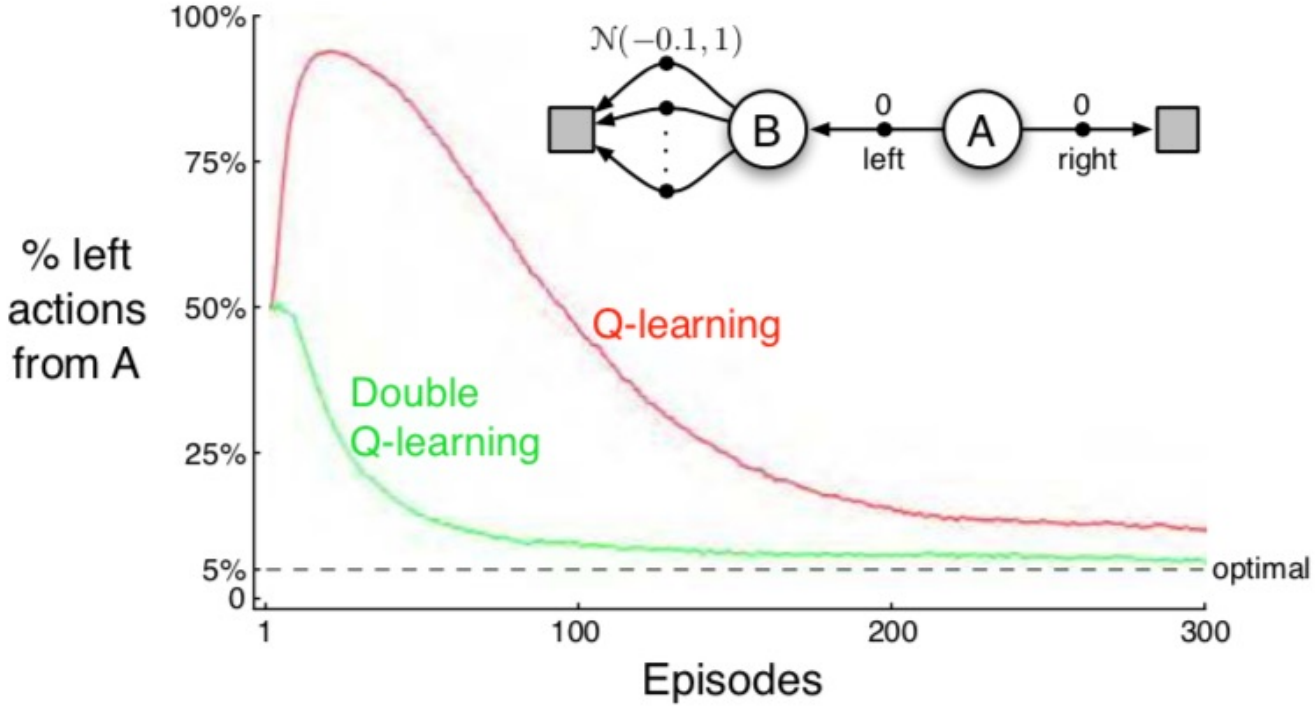


Figure 6.5: Comparison of Q-learning and Double Q-learning on a simple episodic MDP (shown inset). Q-learning initially learns to take the left action much more often than the right action, and always takes it significantly more often than the 5% minimum probability enforced by ε-greedy action selection with $\varepsilon = 0.1$. In contrast, Double Q-learning is essentially unaffected by maximization bias. These data are averaged over 10,000 runs. The initial action-value estimates were zero. Any ties in ε-greedy action selection were broken randomly.

Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

n-step methods for sate-action value

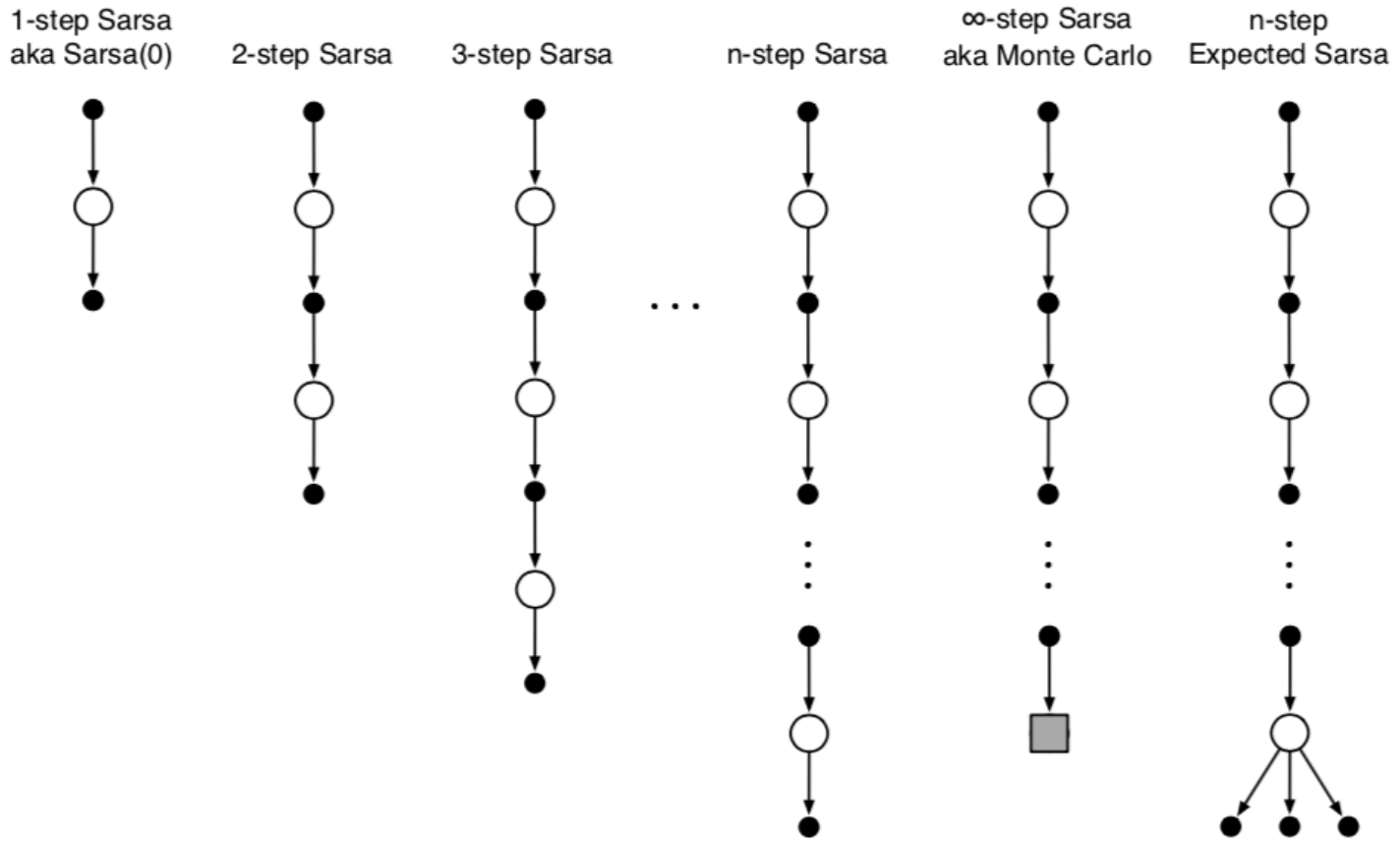
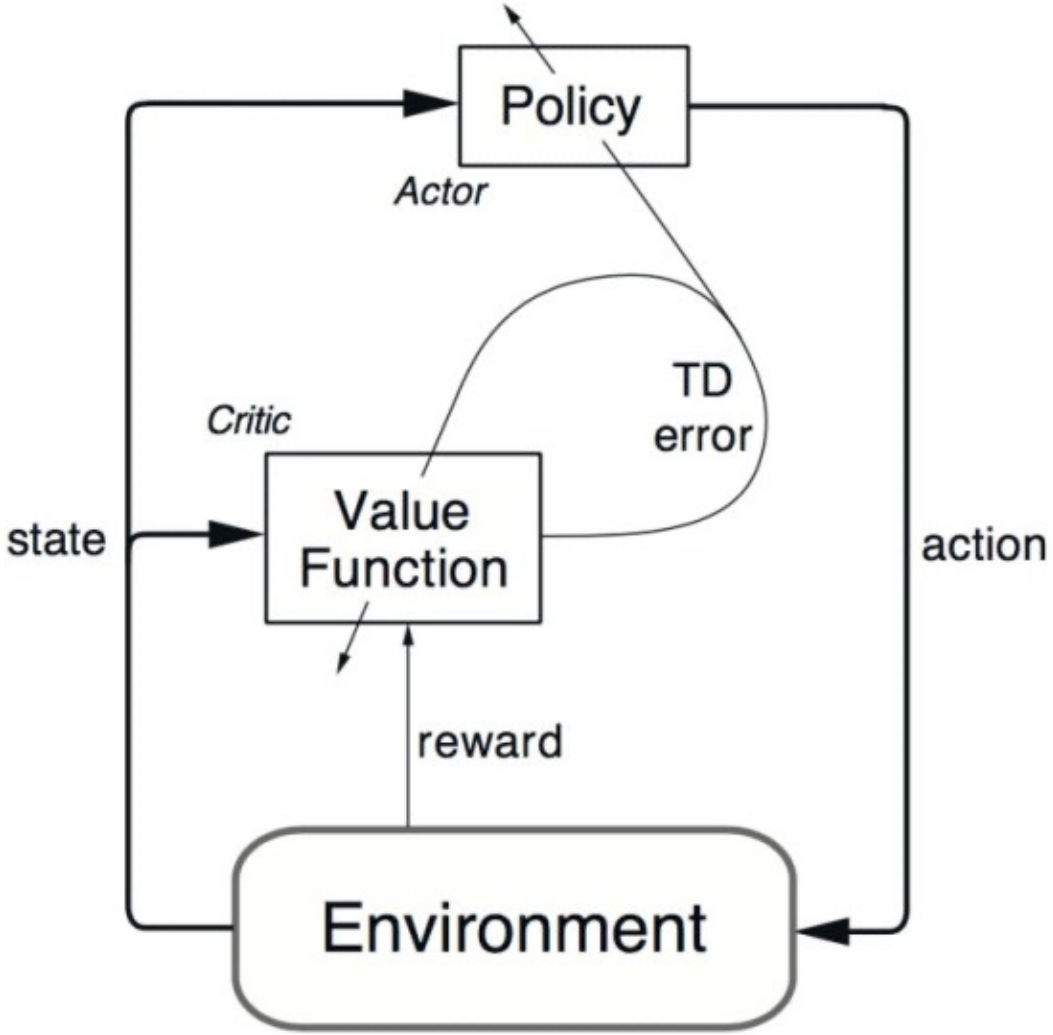


Figure 7.3: The backup diagrams for the spectrum of n-step methods for state–action values. They range from the one-step update of Sarsa(0) to the up-until-termination update of the Monte Carlo method. In between are the n-step updates, based on n steps of real rewards and the estimated value of the nth next state–action pair, all appropriately discounted. On the far right is the backup diagram for n-step Expected Sarsa.

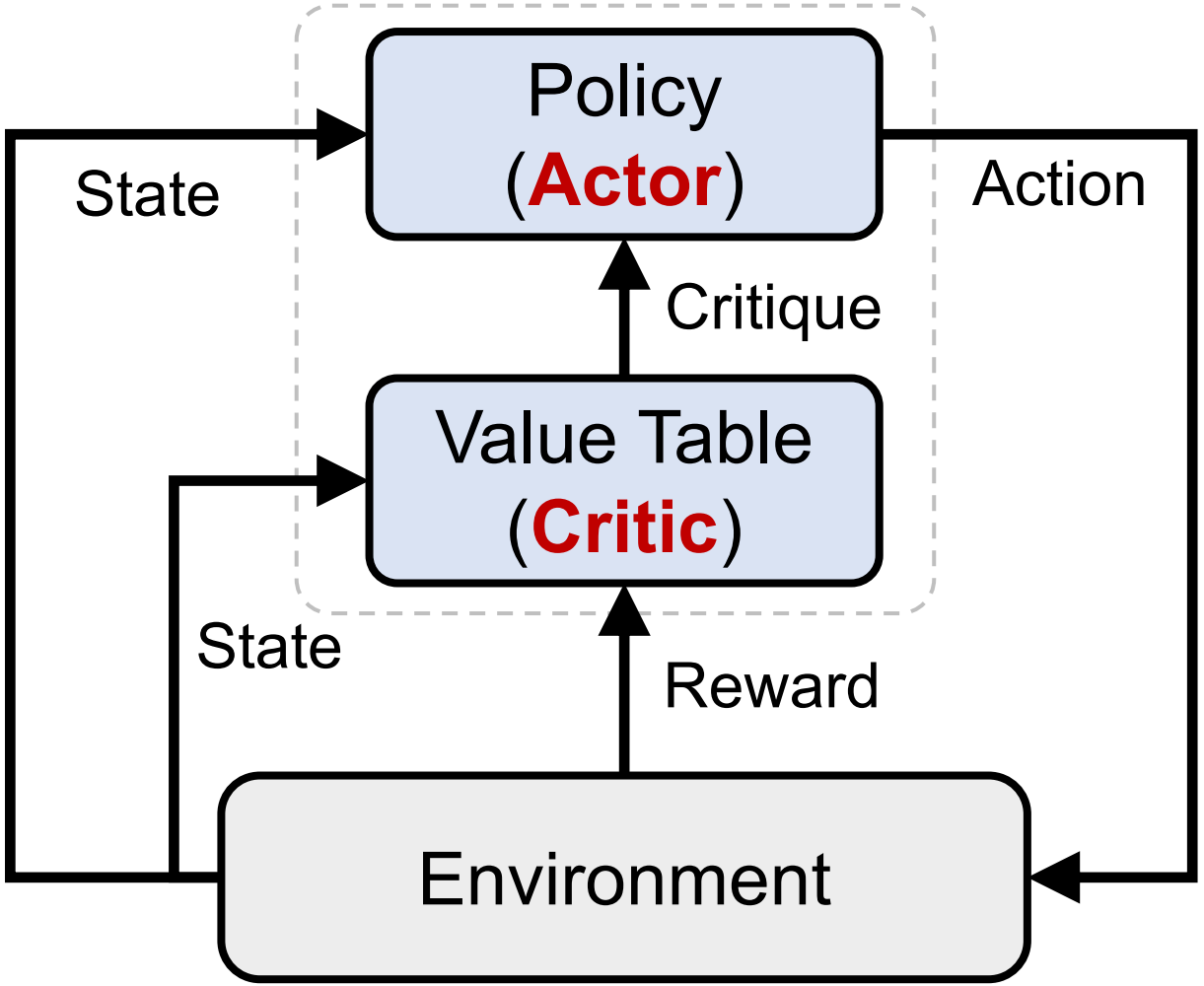
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning
Actor-Critic (AC) Architecture



Source: https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html

Reinforcement Learning
Actor-Critic (AC) Learning Methods



Reinforcement Learning Methods

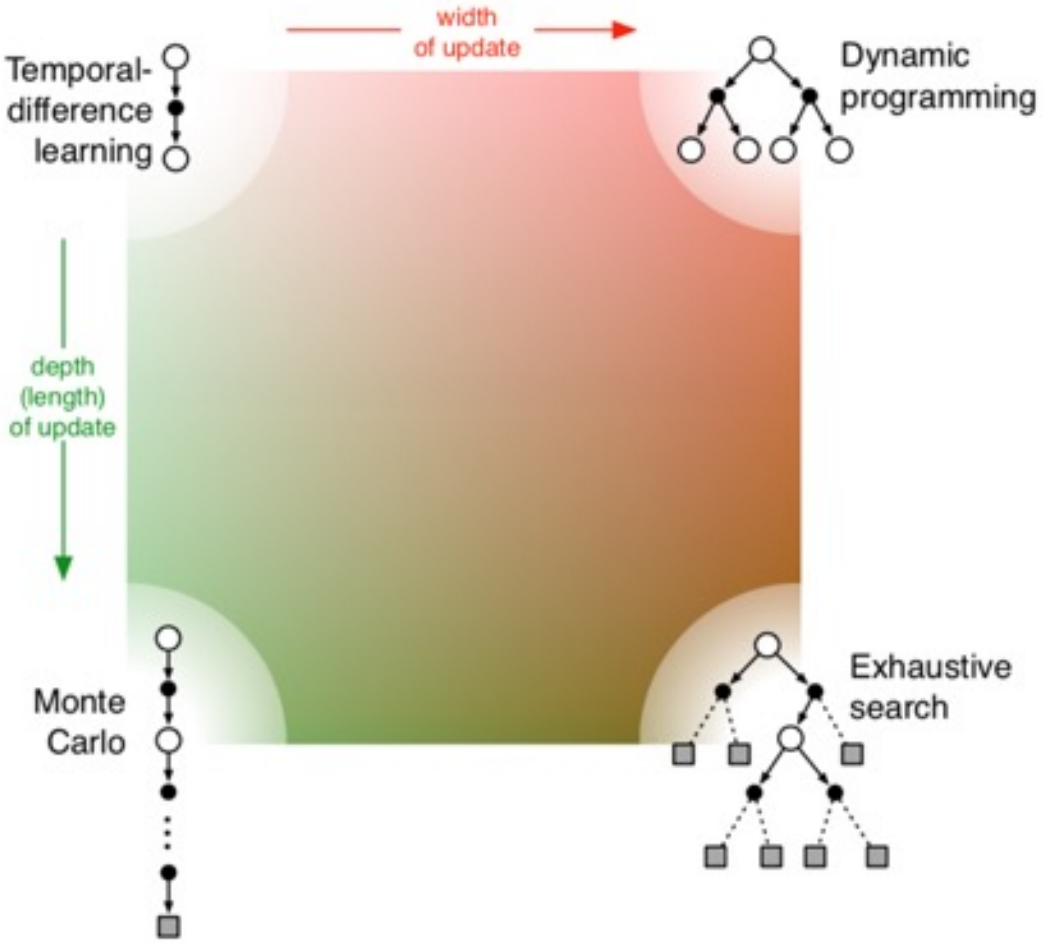


Figure 8.11: A slice through the space of reinforcement learning methods, highlighting the two of the most important dimensions explored in Part I of this book: the depth and width of the updates.

Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Monte Carlo Tree Search (MCTS)

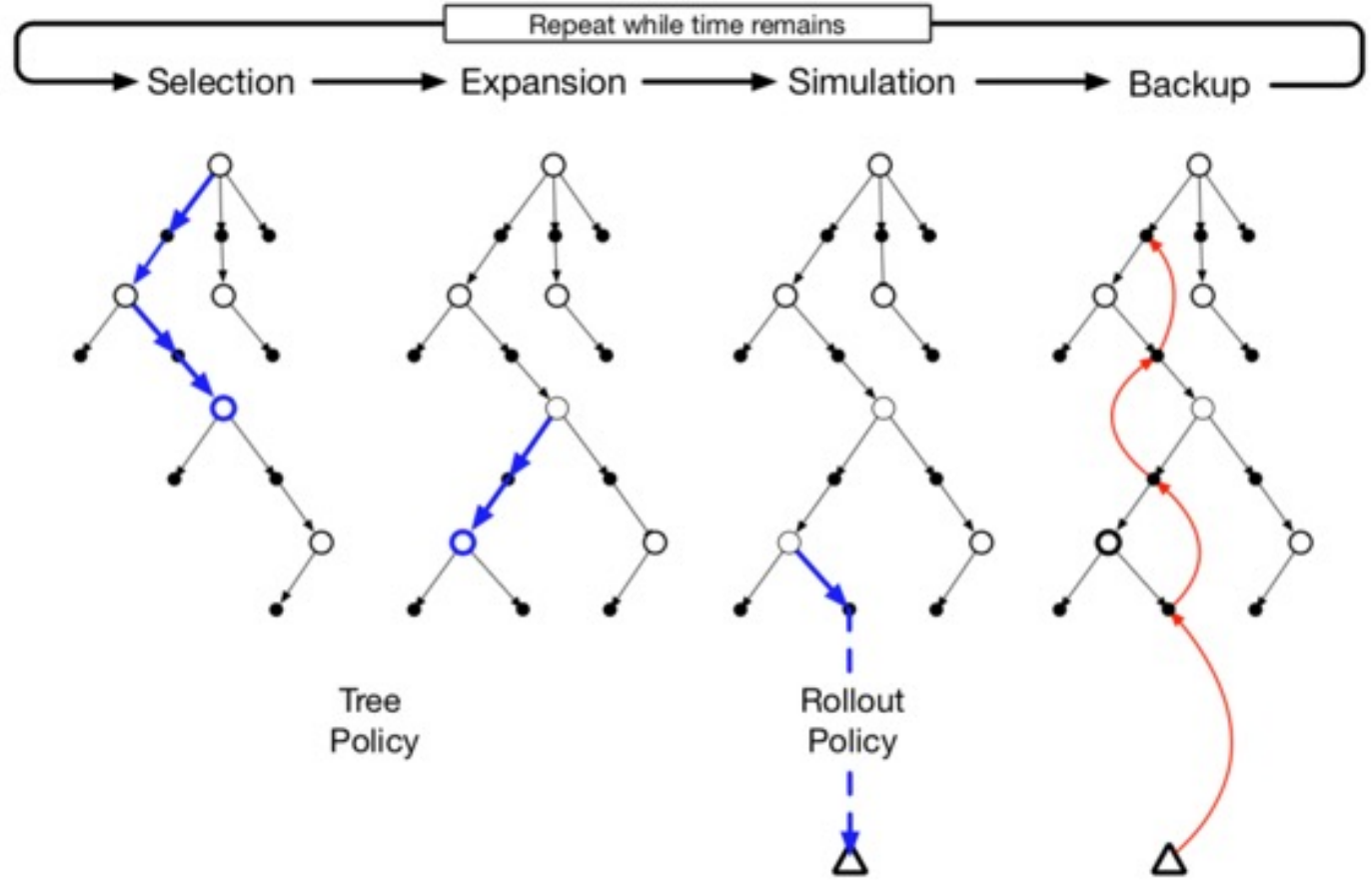
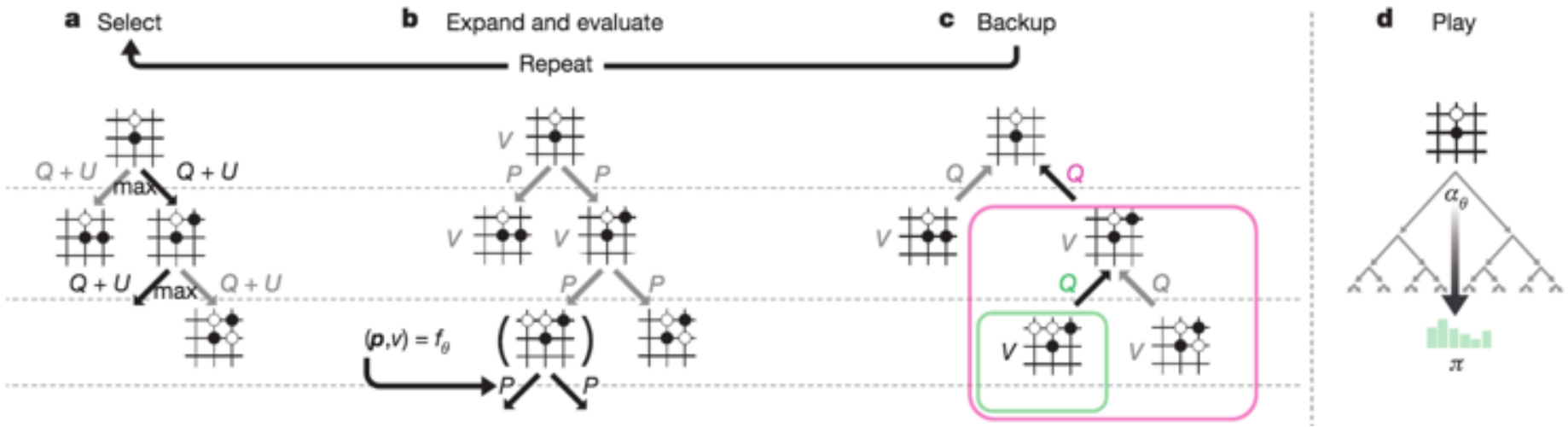


Figure 8.10: Monte Carlo Tree Search. When the environment changes to a new state, MCTS executes as many iterations as possible before an action needs to be selected, incrementally building a tree whose root node represents the current state. Each iteration consists of the four operations **Selection**, **Expansion** (though possibly skipped on some iterations), **Simulation**, and **Backup**, as explained in the text and illustrated by the bold arrows in the trees. Adapted from Chaslot, Bakkes, Szita, and Spronck (2008).

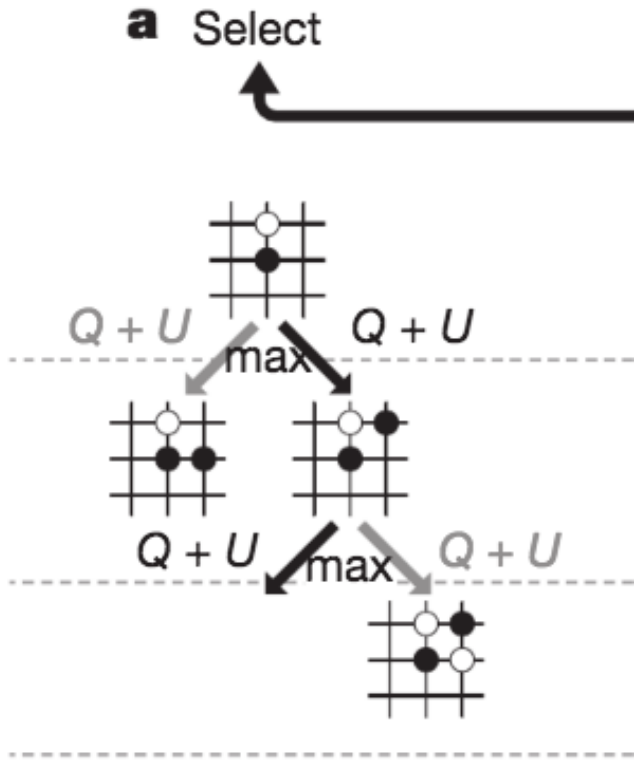
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Monte Carlo Tree Search (MCTS)
MCTS in AlphaGo Zero



Source: David Silver et al. (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.

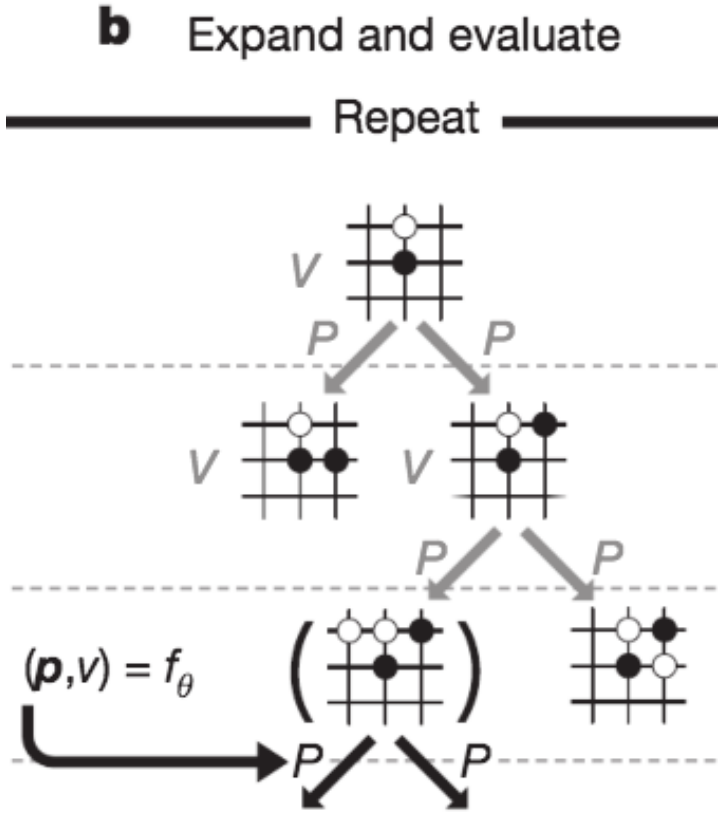
MCTS in AlphaGo Zero



a: Each simulation traverses the tree by selecting the edge with maximum action value Q, plus an upper confidence bound U that depends on a stored prior probability P and visit count N for that edge (which is incremented once traversed).

Source: David Silver et al. (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.

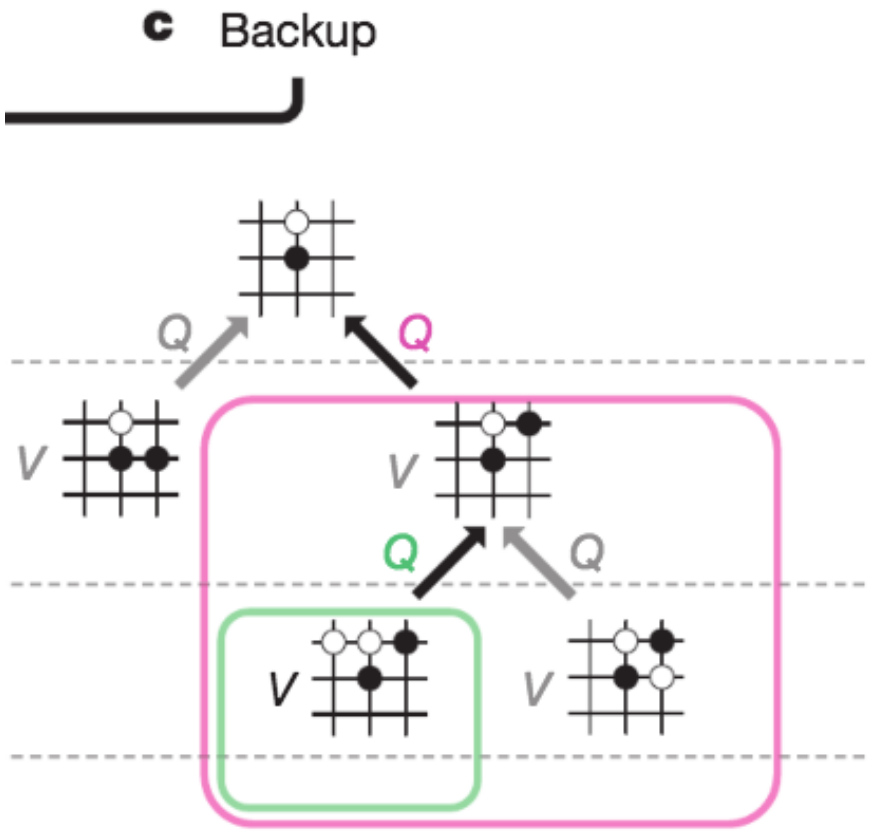
MCTS in AlphaGo Zero



b: The leaf node is expanded and the associated position s is evaluated by the neural network $(P(s, \cdot), V(s)) = f_\theta(s)$; the vector of P values are stored in the outgoing edges from s.

Source: David Silver et al. (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.

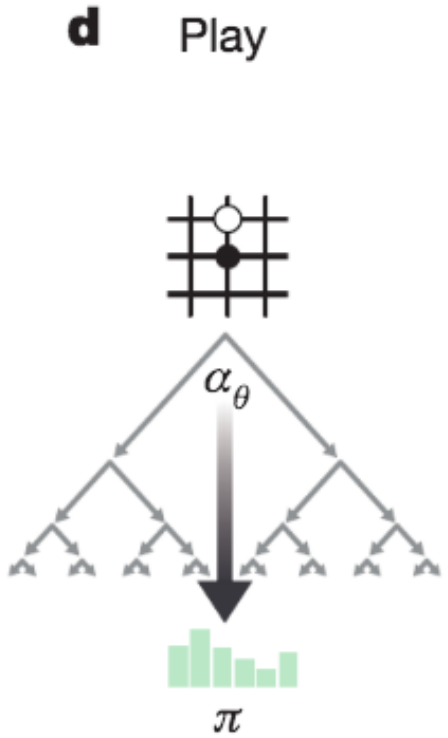
MCTS in AlphaGo Zero



c: Action value Q is updated to track the mean of all evaluations V in the subtree below that action

Source: David Silver et al. (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.

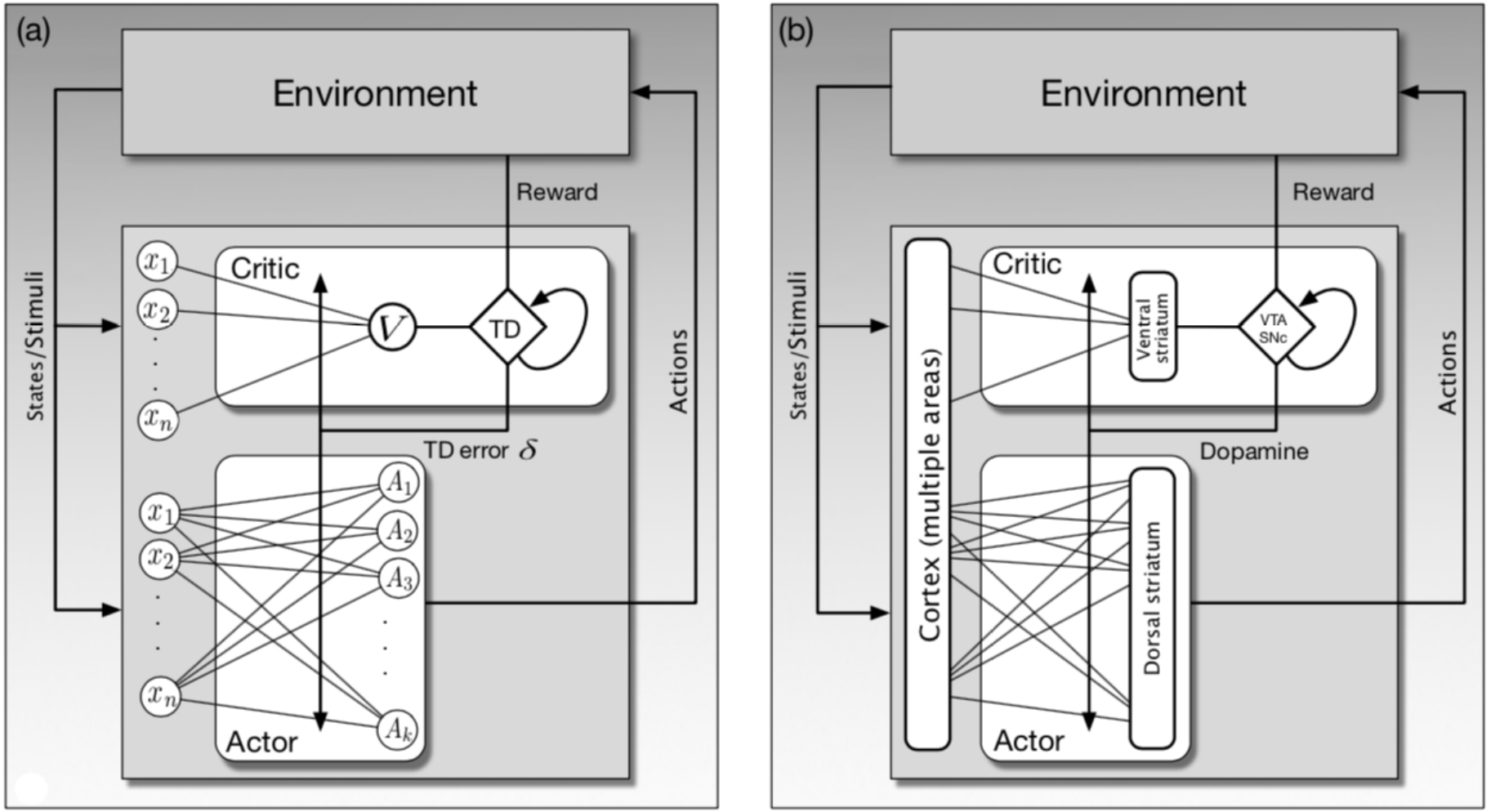
MCTS in AlphaGo Zero



d: Once the search is complete, search probabilities π are returned, proportional to $N^{1/\tau}$, where N is the visit count of each move from the root state and τ is a parameter controlling temperature.

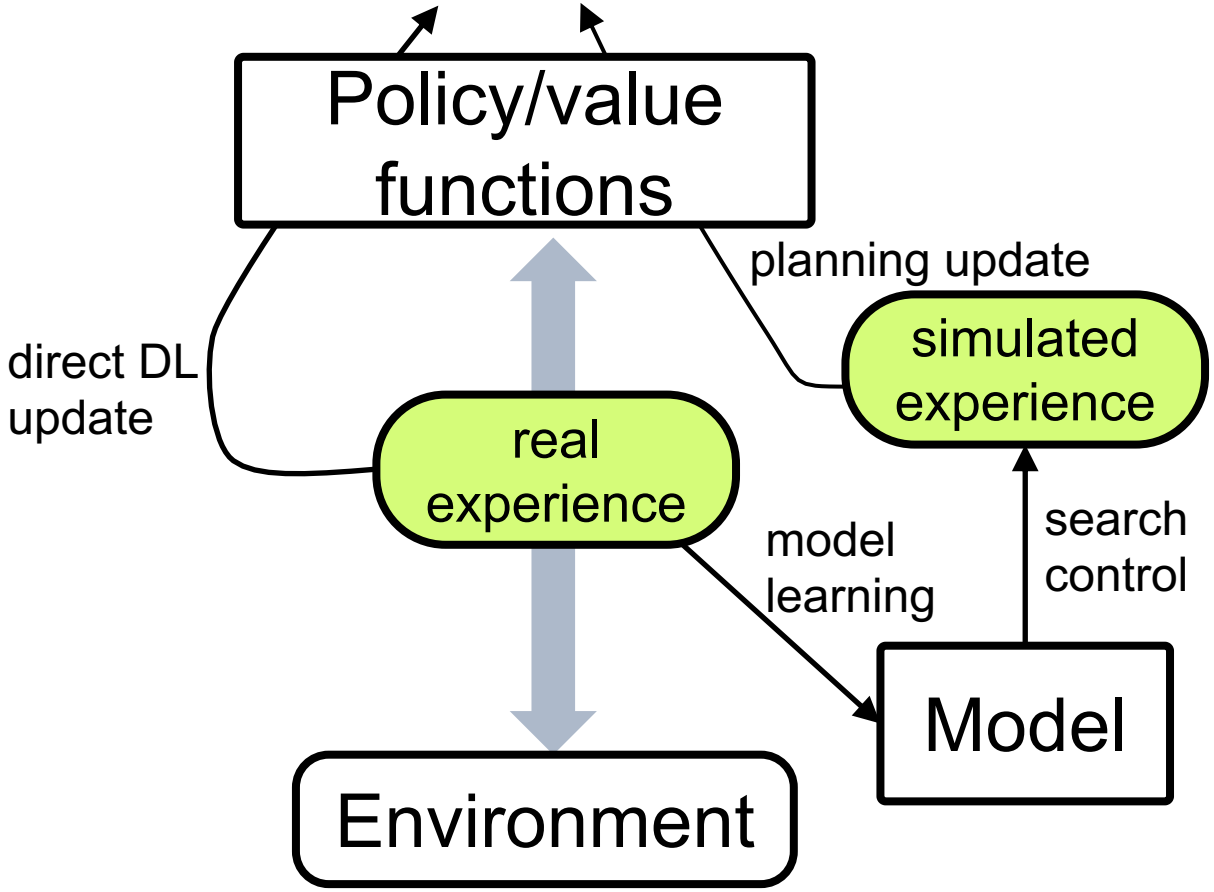
Source: David Silver et al. (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.

Reinforcement Learning
Actor Critic ANN



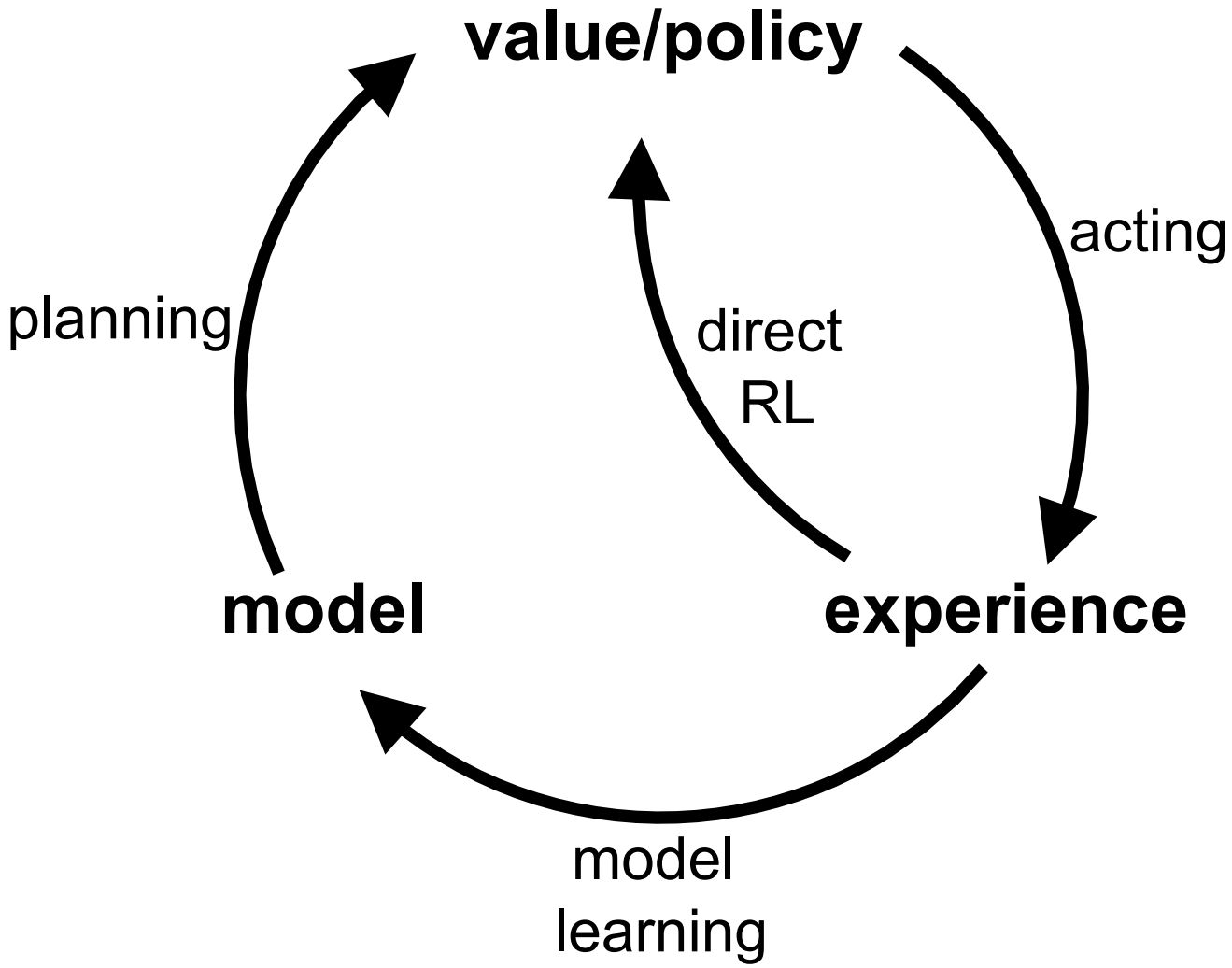
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning
General Dyna Architecture



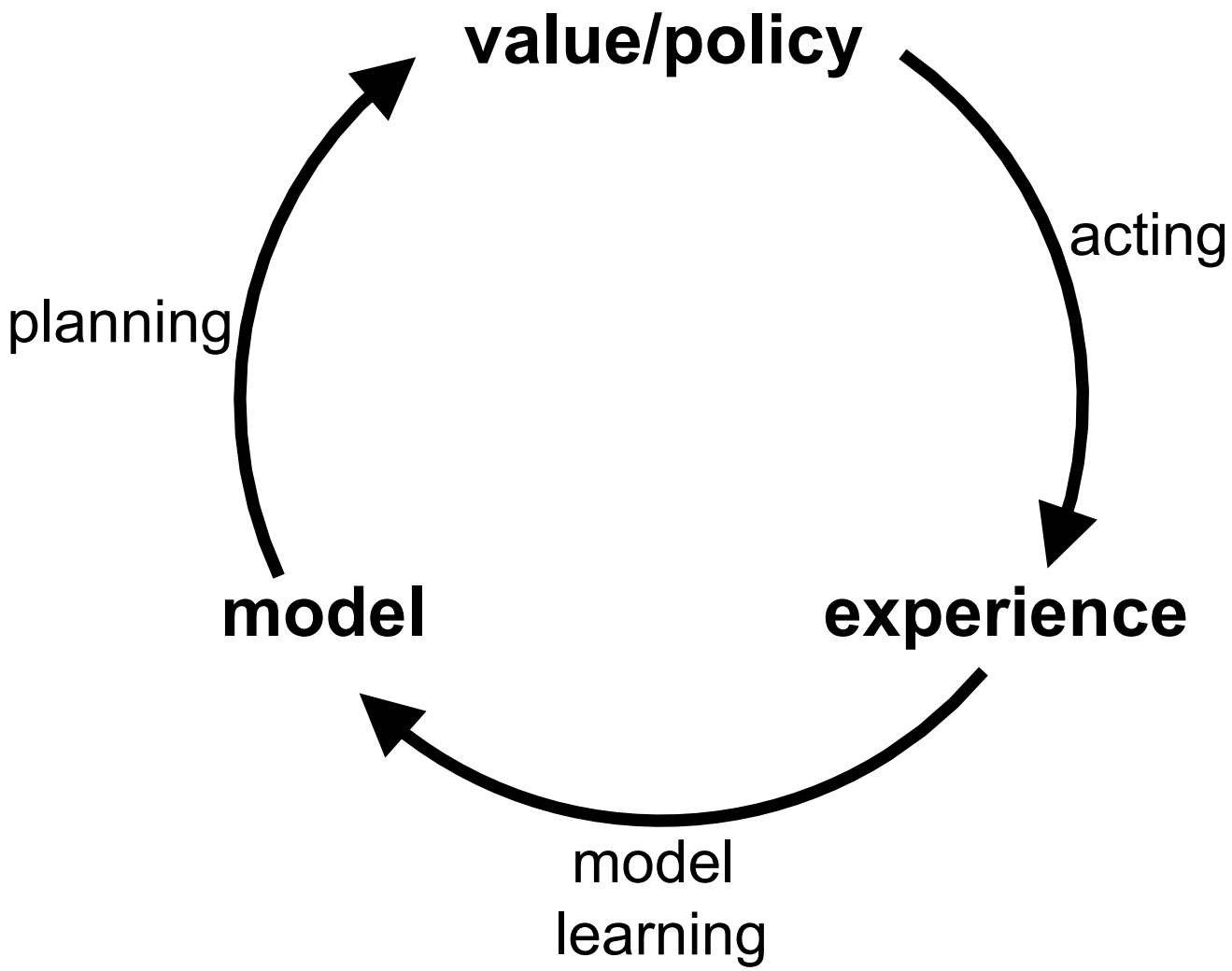
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Dyna:
Integrated Planning, Acting, and Learning



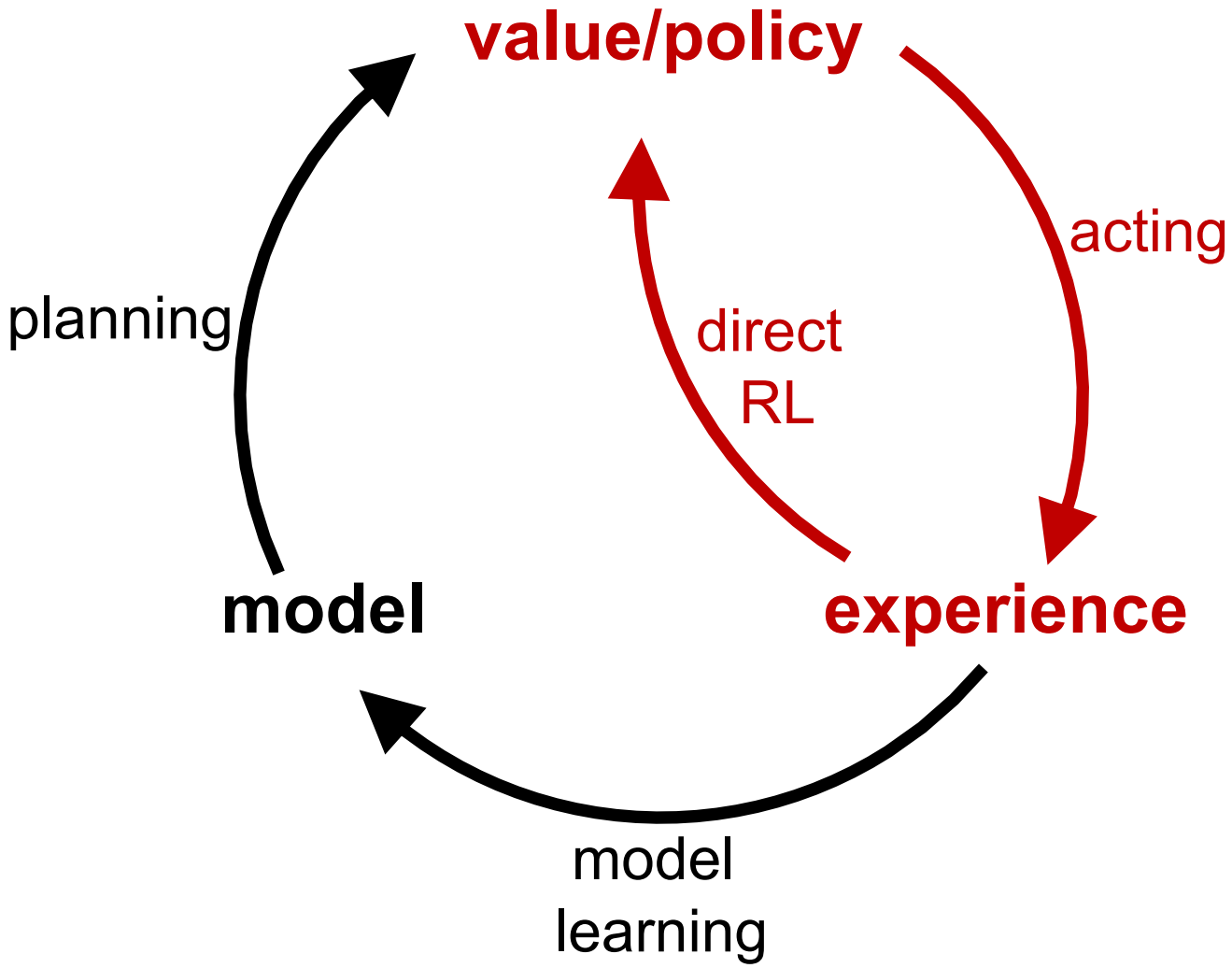
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Model-Based RL



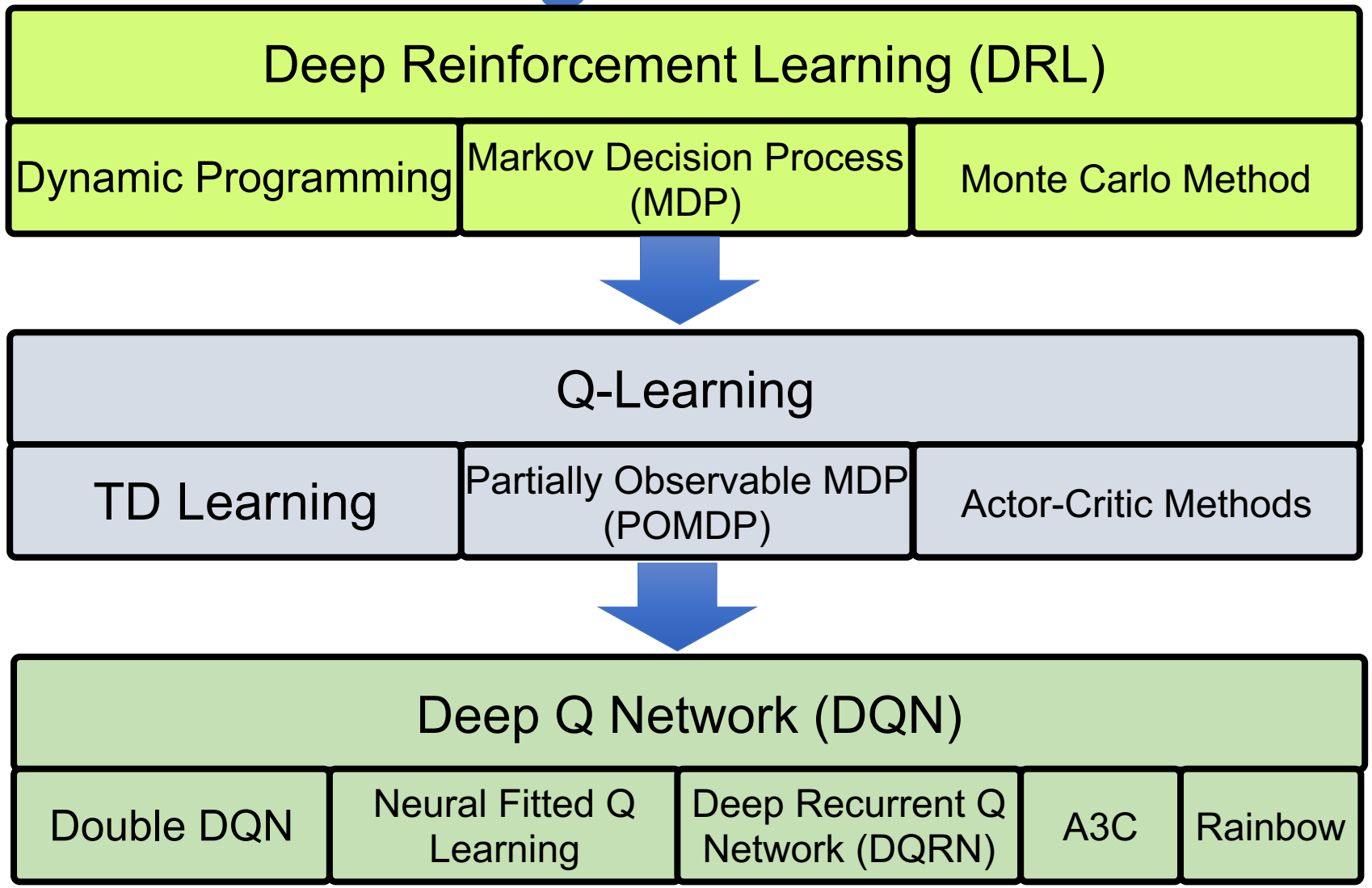
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Model-Free RL (DQN, A3C)



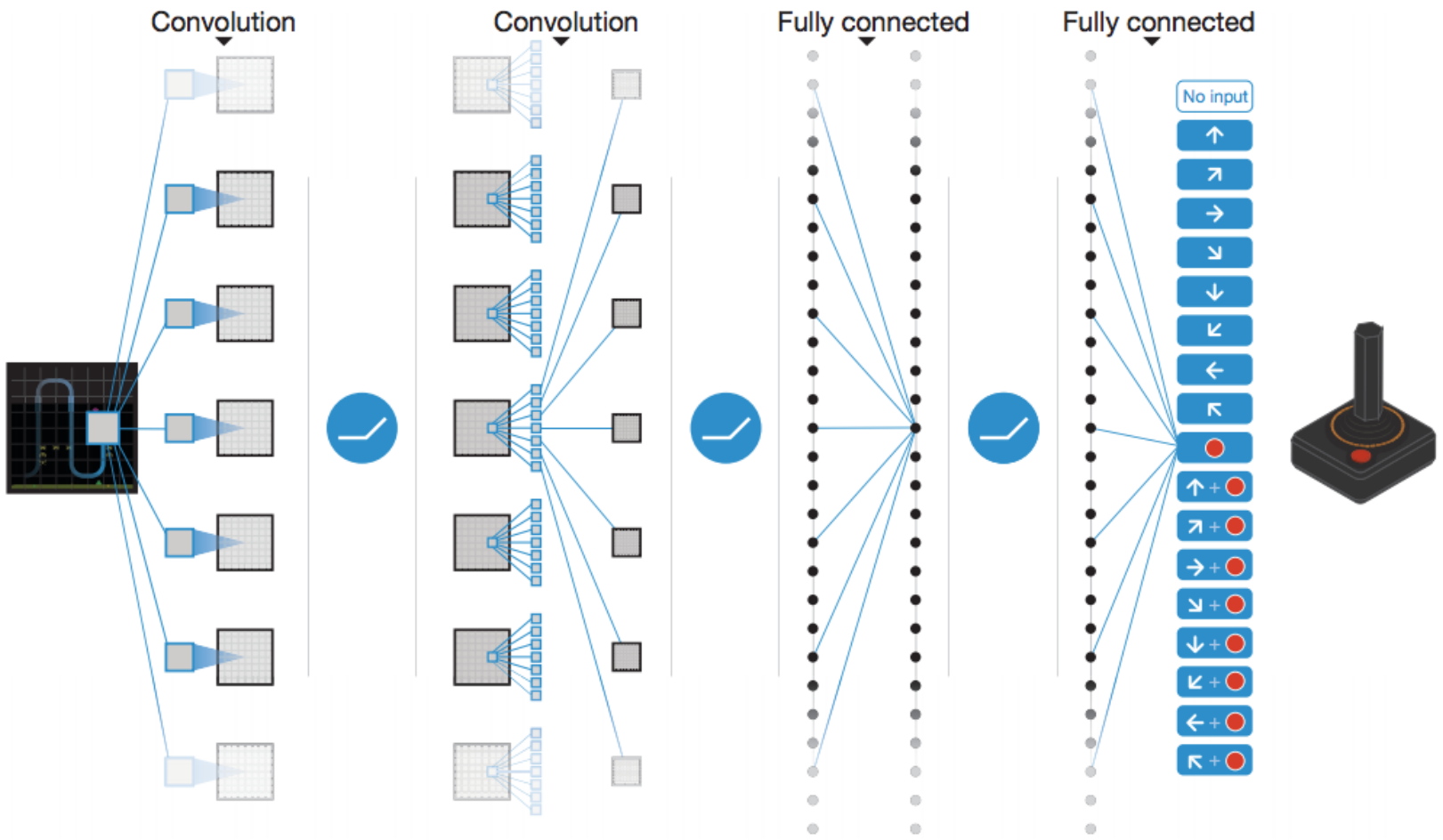
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning Algorithms



Source: https://amitray.com/deep-learning-past-present-and-future-a-review/

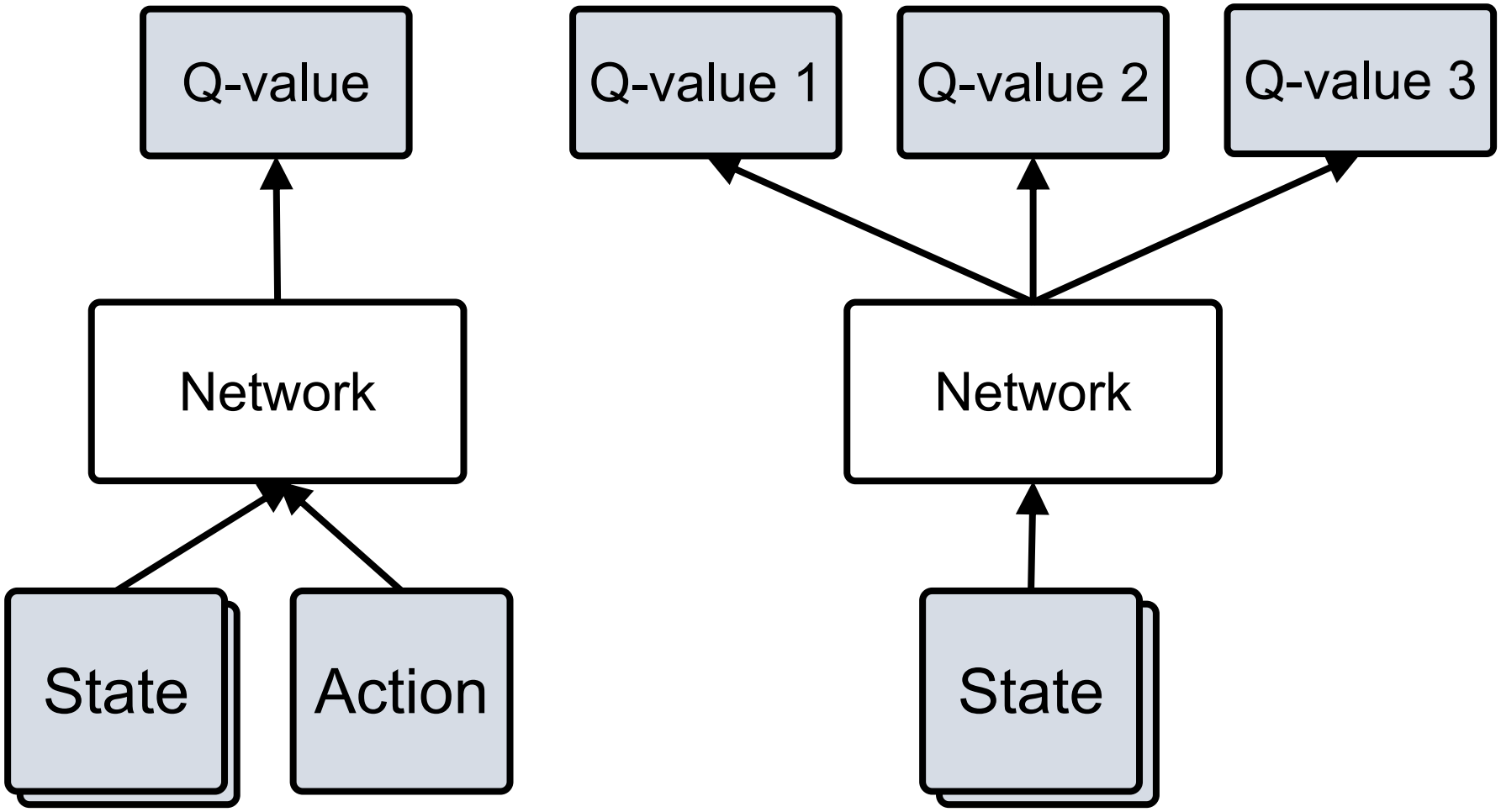
Human-level control through deep reinforcement learning (DQN)



Schematic illustration of the convolutional neural network

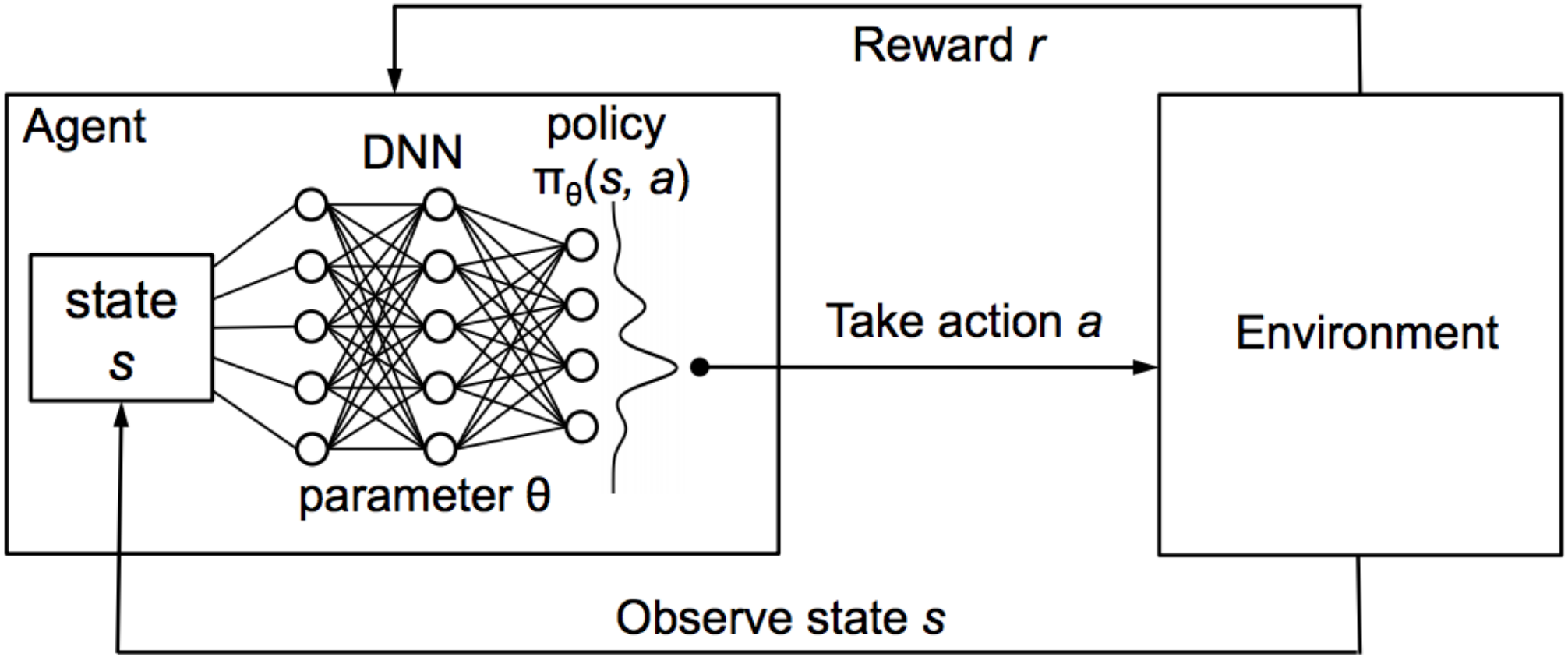
Source: Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." Nature 518, no. 7540 (2015): 529.

Deep Q-Network (DQN)



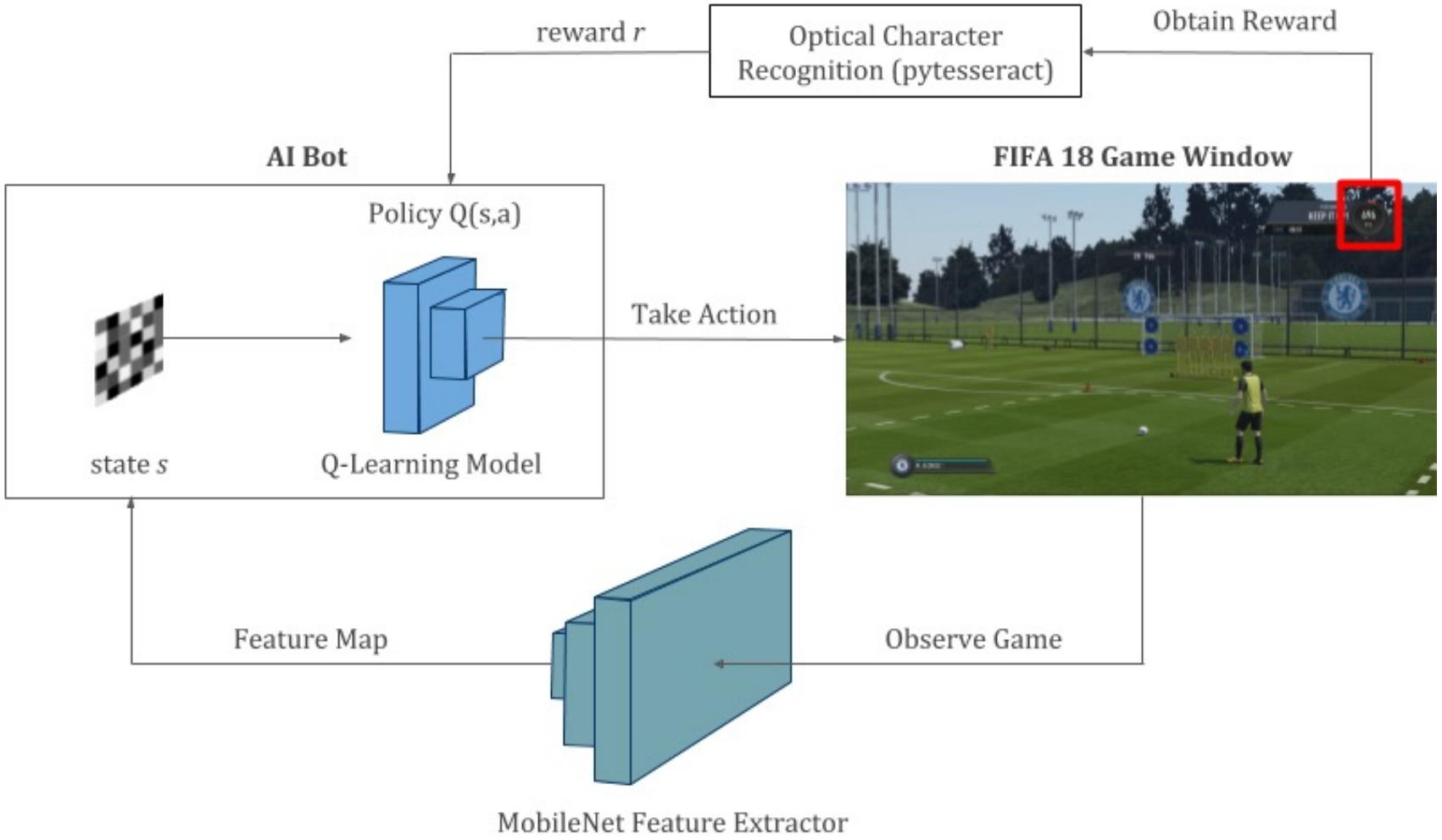
Source: https://neuro.cs.ut.ee/demystifying-deep-reinforcement-learning/

Reinforcement Learning with policy represented via DNN



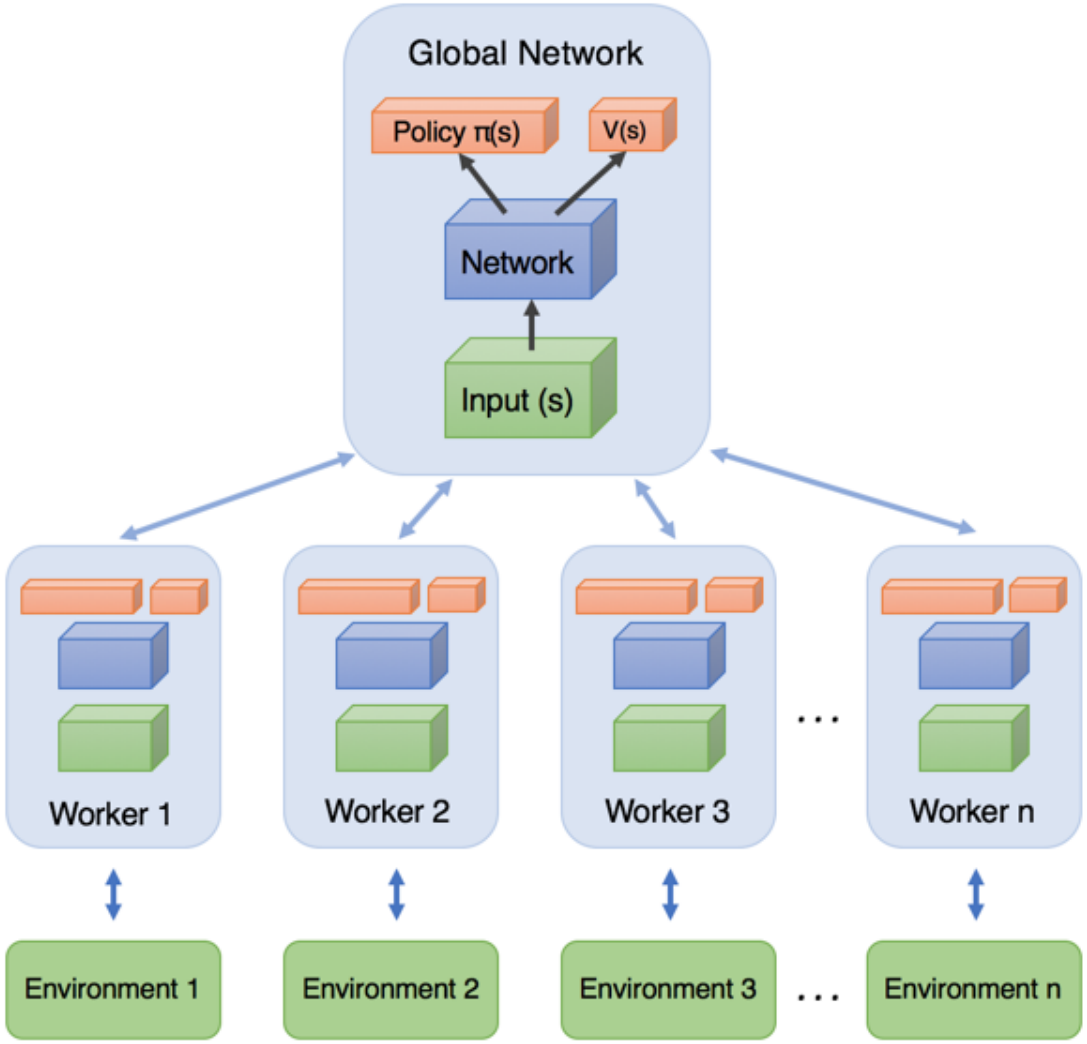
Source: Hongzi Mao, Mohammad Alizadeh, Ishai Menache, and Srikanth Kandula. (2016) "Resource management with deep reinforcement learning." In Proceedings of the 15th ACM Workshop on Hot Topics in Networks, pp. 50-56. ACM, 2016.

Reinforcement Learning
Deep Q-Learning in FIFA 18



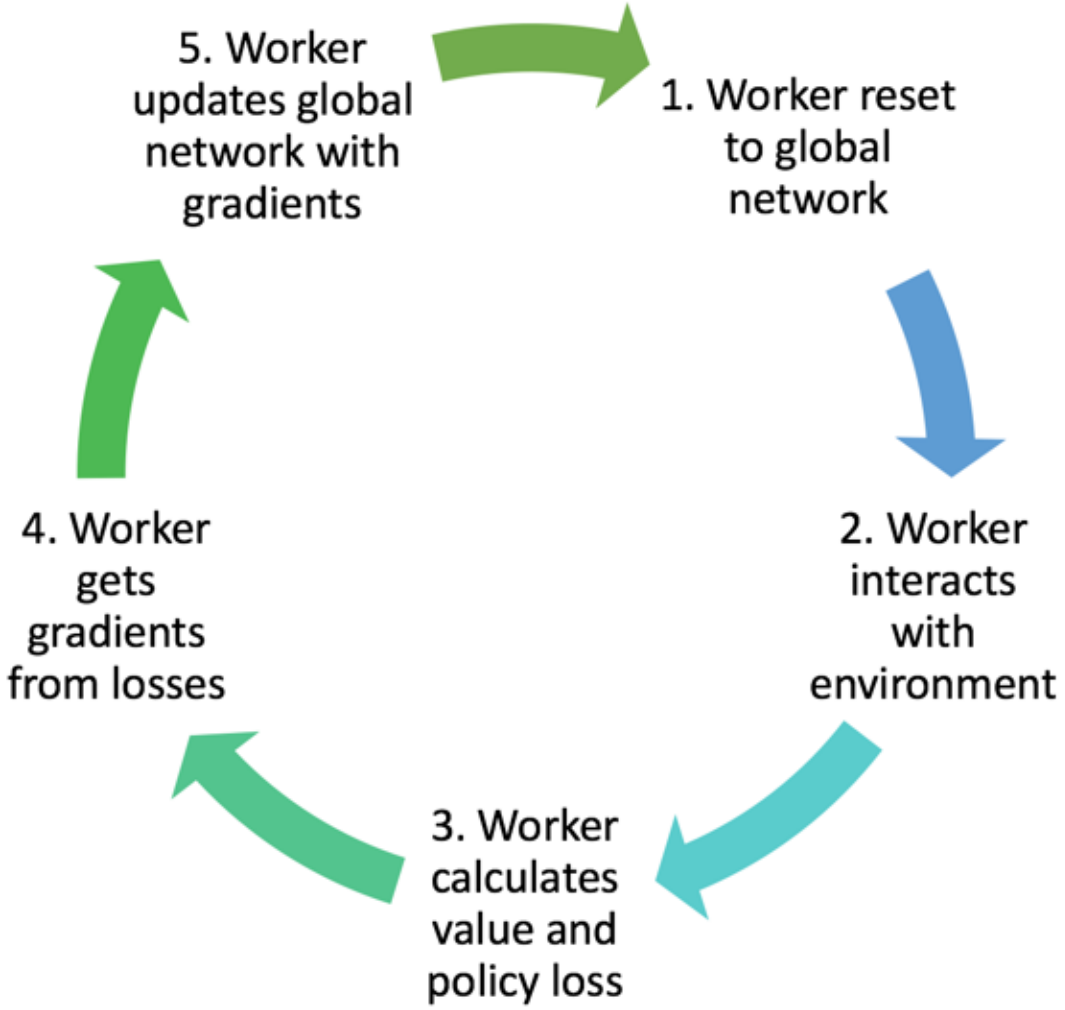
Source: https://towardsdatascience.com/using-deep-q-learning-in-fifa-18-to-perfect-the-art-of-free-kicks-f2e4e979ee66

Asynchronous Advantage Actor-Critic (A3C)



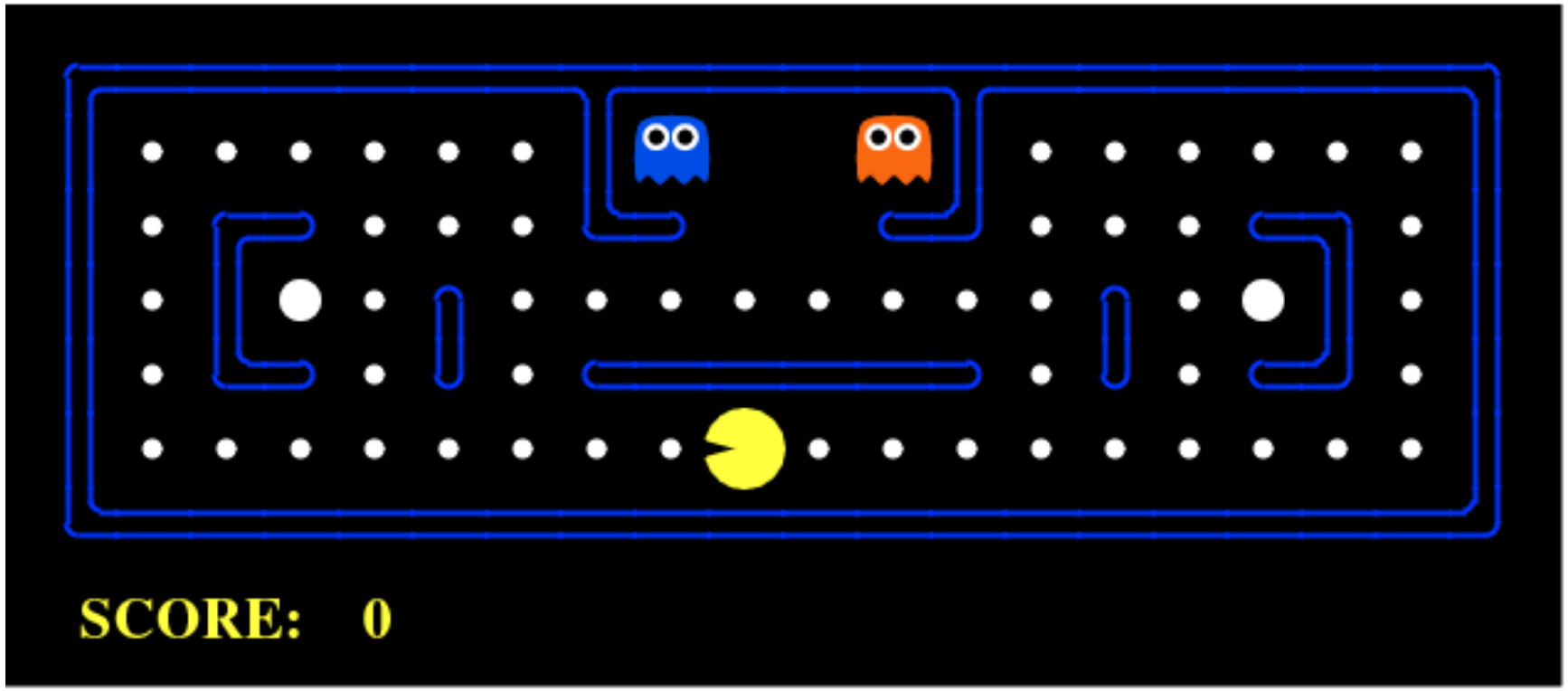
Source: https://medium.com/emergent-future/simple-reinforcement-learning-with-tensorflow-part-8-asynchronous-actor-critic-agents-a3c-c88f72a5e9f2

Training workflow of each worker agent in A3C



Source: https://medium.com/emergent-future/simple-reinforcement-learning-with-tensorflow-part-8-asynchronous-actor-critic-agents-a3c-c88f72a5e9f2

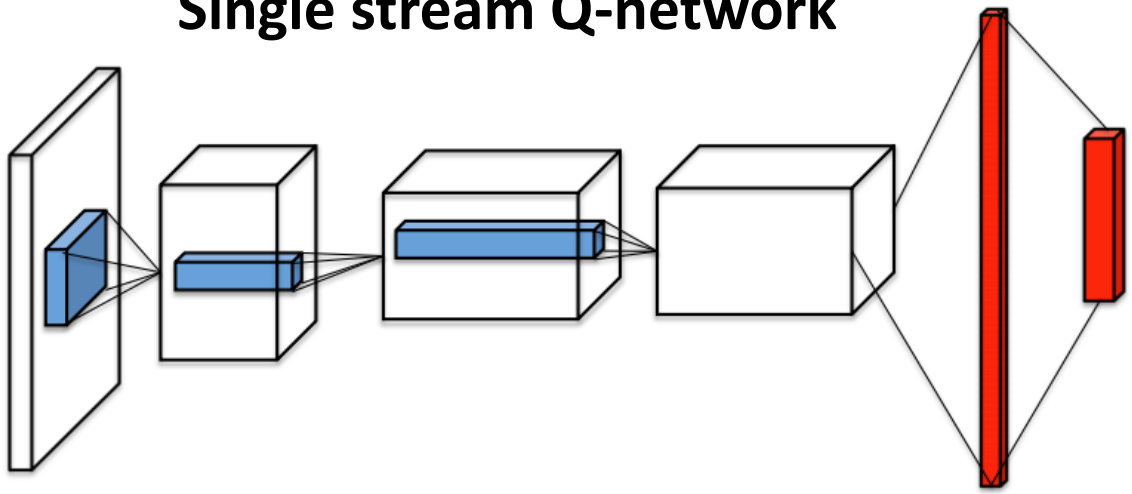
Reinforcement Learning Example: PCMAN



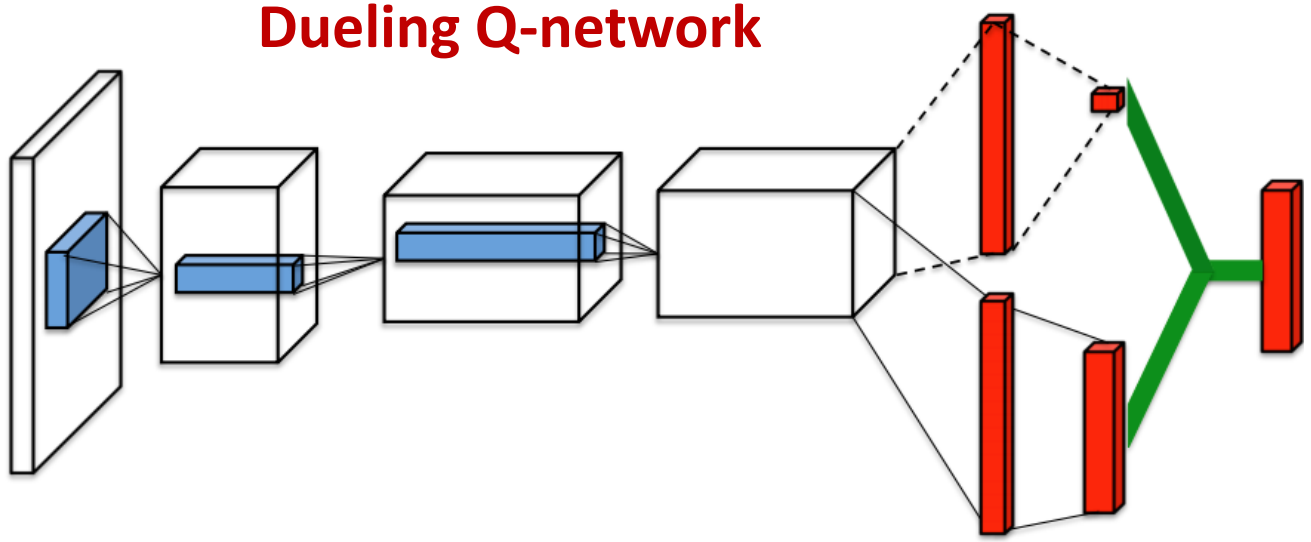
Source: https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html

Dueling Network Architectures for Deep Reinforcement Learning

Single stream Q-network

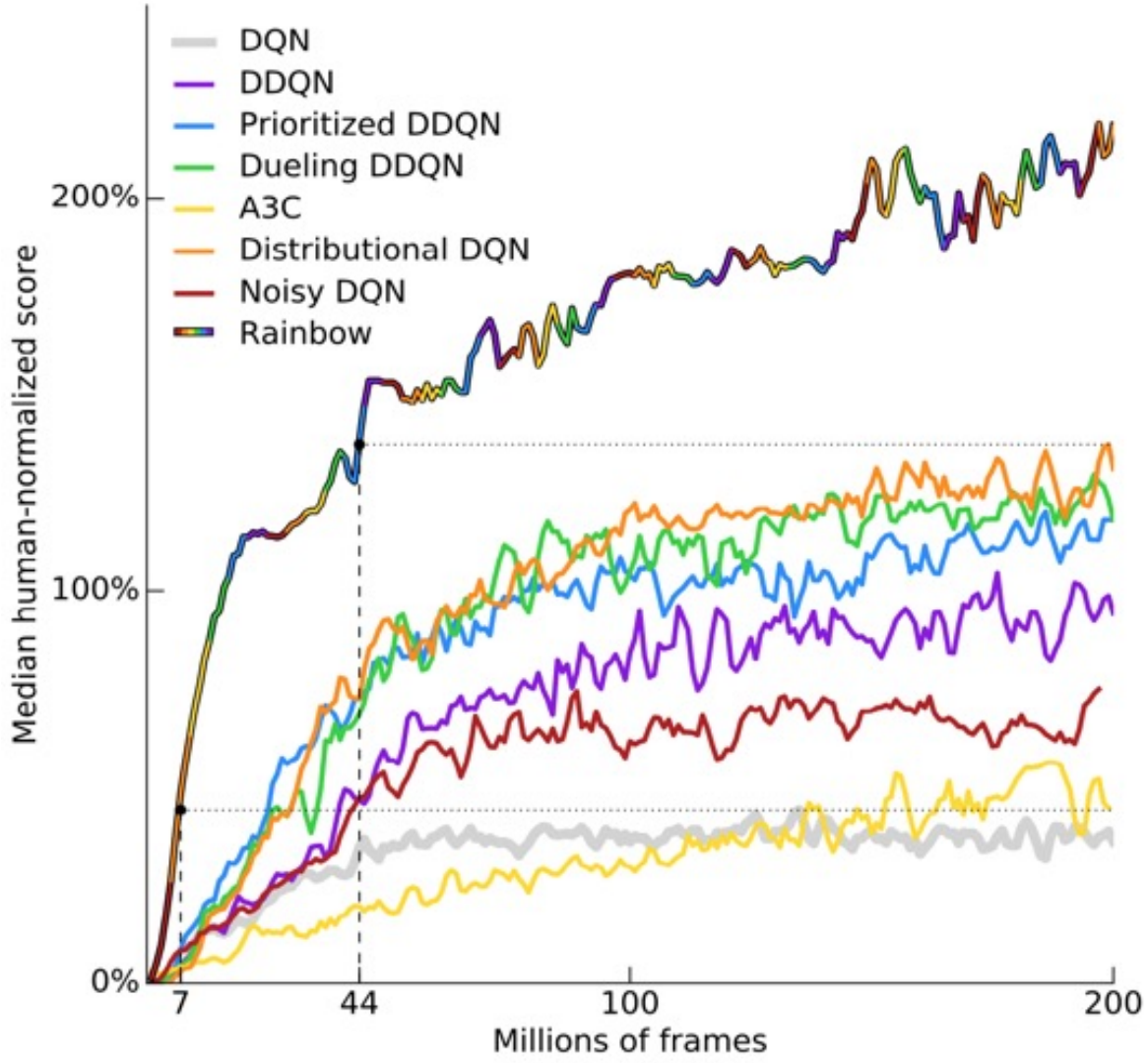


Dueling Q-network



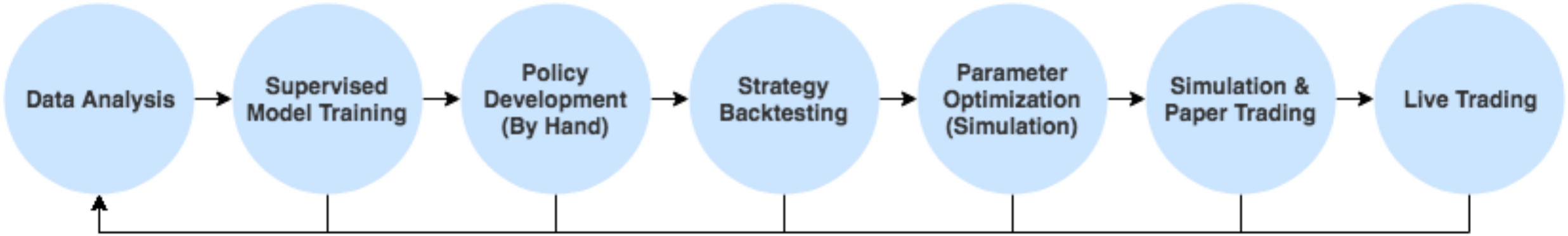
Source: Ziyu Wang et al. (2015). "Dueling network architectures for deep reinforcement learning." arXiv preprint arXiv:1511.06581 (2015)

Rainbow: Combining improvements in deep reinforcement learning



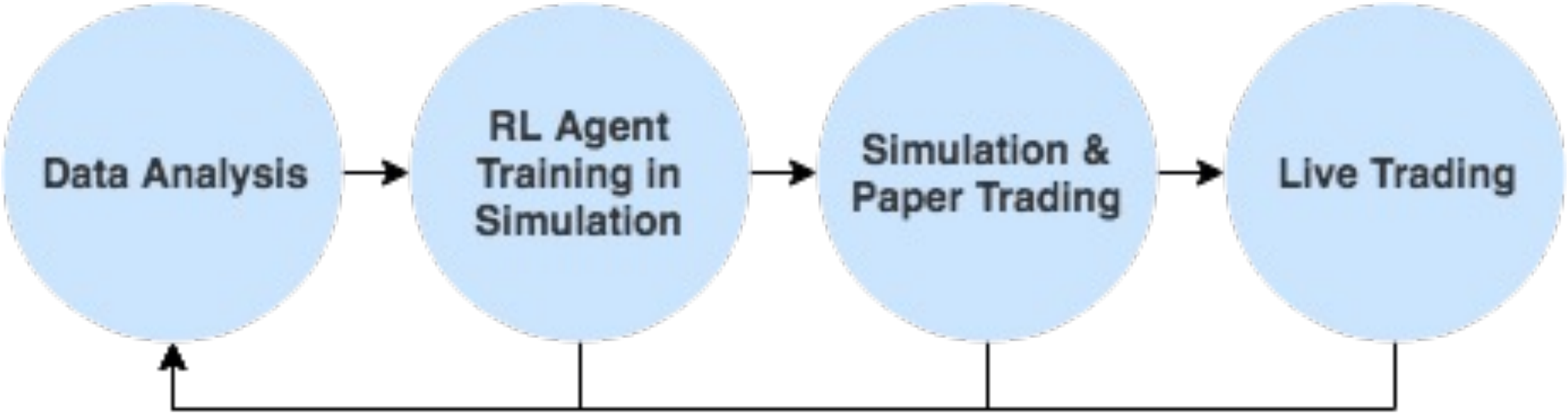
Source: Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver (2017). "Rainbow: Combining improvements in deep reinforcement learning." arXiv preprint arXiv:1710.02298 (2017).

A Typical Strategy Development Workflow



Source: http://www.wildml.com/2018/02/introduction-to-learning-to-trade-with-reinforcement-learning/

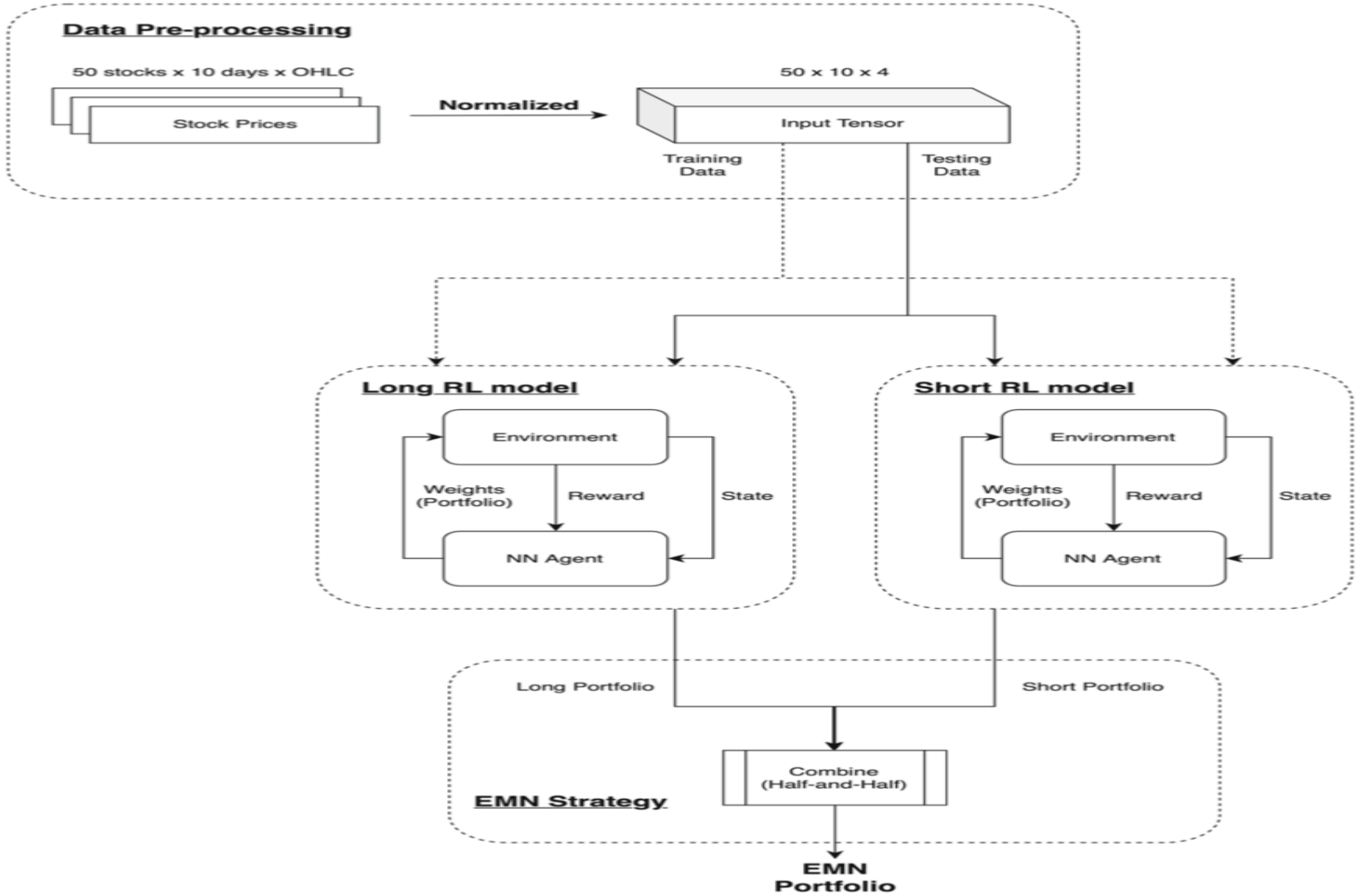
Reinforcement Learning (RL) in Trading Strategies



Source: http://www.wildml.com/2018/02/introduction-to-learning-to-trade-with-reinforcement-learning/

Portfolio management system in equity market neutral using reinforcement learning

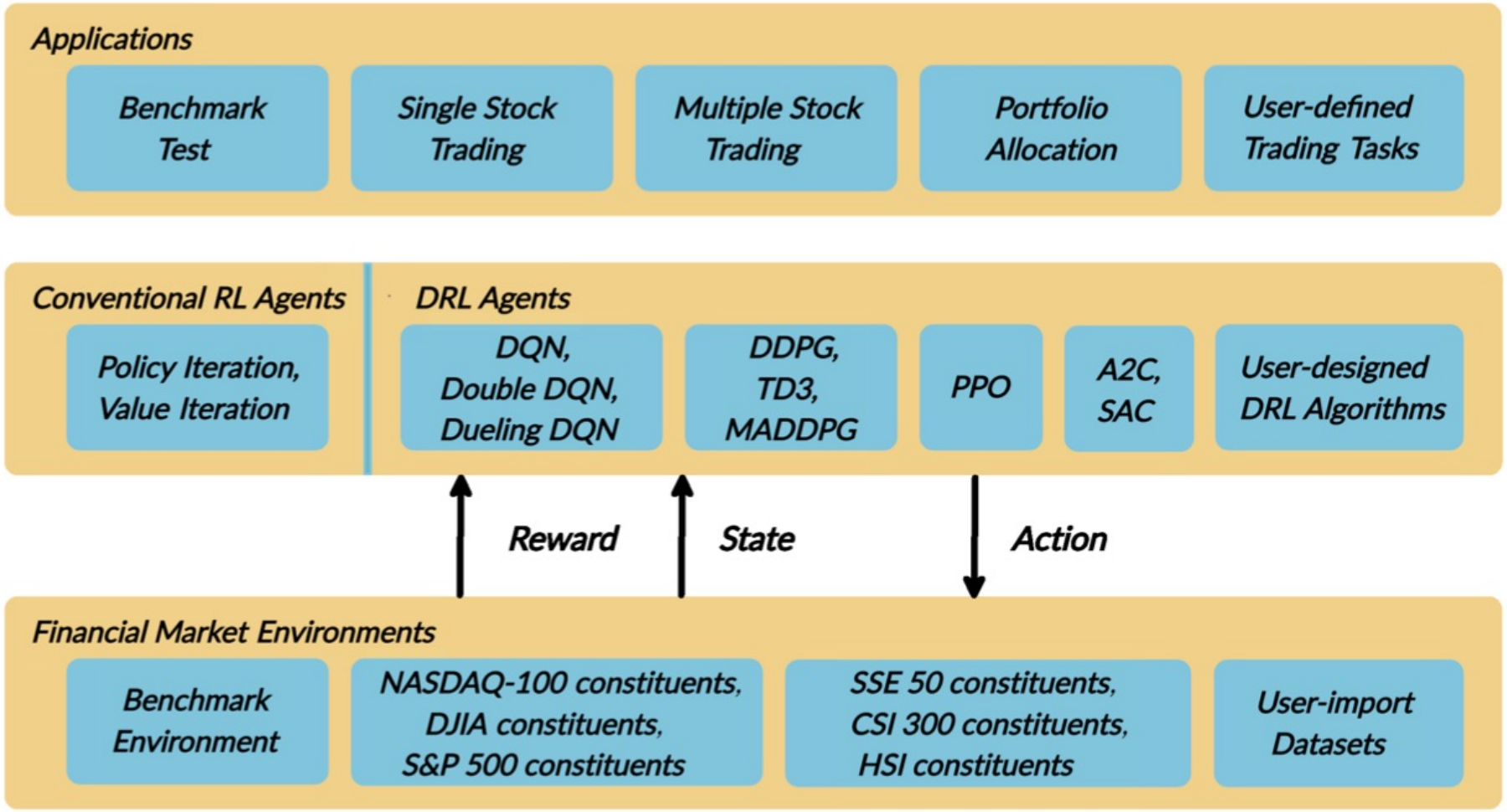
(Wu et al., 2021)



Source: Mu-En Wu, Jia-Hao Syu, Jerry Chun-Wei Lin, and Jan-Ming Ho. "Portfolio management system in equity market neutral using reinforcement learning." Applied Intelligence (2021): 1-13.

FinRL:

A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance



Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

FinRL

Deep Reinforcement Learning Algorithms

Algorithms	Input	Output	Type	State-action spaces support	Finance use cases support	Features and Improvements	Advantages
DQN	States	Q-value	Value based	Discrete only	Single stock trading	Target network, experience replay	Simple and easy to use
Double DQN	States	Q-value	Value based	Discrete only	Single stock trading	Use two identical neural network models to learn	Reduce overestimations
Dueling DQN	States	Q-value	Value based	Discrete only	Single stock trading	Add a specialized dueling Q head	Better differentiate actions, improves the learning
DDPG	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Being deep Q-learning for continuous action spaces	Better at handling high-dimensional continuous action spaces
A2C	State action pair	Q-value	Actor-critic based	Discrete and continuous	All use cases	Advantage function, parallel gradients updating	Stable, cost-effective, faster and works better with large batch sizes
PPO	State action pair	Q-value	Actor-critic based	Discrete and continuous	All use cases	Clipped surrogate objective function	Improve stability, less variance, simply to implement
SAC	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Entropy regularization, exploration-exploitation trade-off	Improve stability
TD3	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Clipped double Q-Learning, delayed policy update, target policy smoothing.	Improve DDPG performance
MADDPG	State action pair	Q-value	Actor-critic based	Continuous only	Multiple stock trading, portfolio allocation	Handle multi-agent RL problem	Improve stability and performance

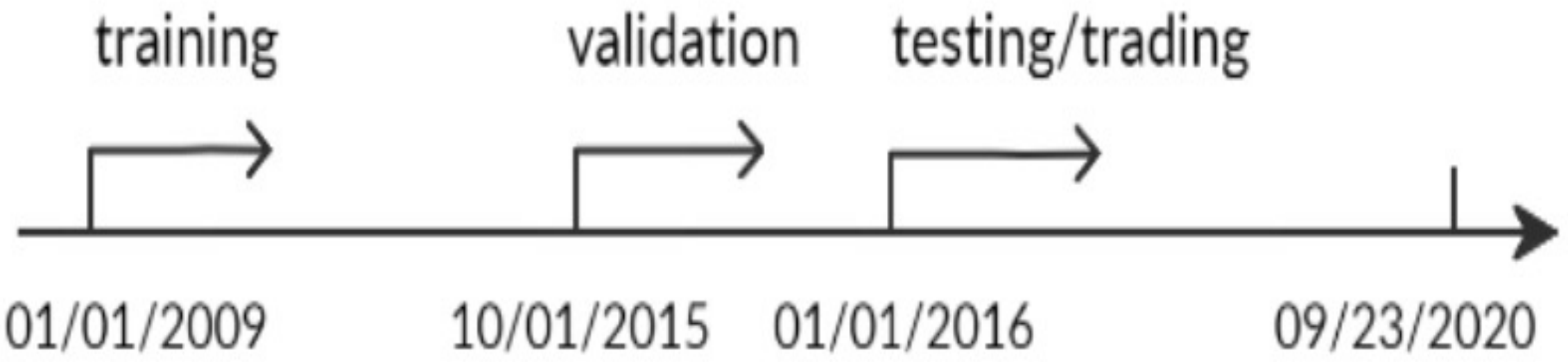
Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

FinRL:

A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance

Evaluation of Trading Performance

Training-Validation-Testing Flow



Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

Reinforcement Learning (RL)
FinRL

Performance of single stock trading

using Proximal policy optimization (PPO) in the FinRL library

Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

Reinforcement Learning (RL)
FinRL

Performance of multiple stock trading and portfolio allocation using the FinRL library

Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

Reinforcement Learning (RL)

FinRL

Performance of single stock trading using Proximal policy optimization (PPO) in the FinRL library

2019/01/01-2020/09/23	SPY	QQQ	GOOGL	AMZN	AAPL	MSFT	S&P 500
Initial value	100,000	100,000	100,000	100,000	100,000	100,000	100,000
Final value	127,044	163,647	174,825	192,031	173,063	172,797	133,402
Annualized return	14.89%	32.33%	37.40%	44.94%	36.88%	36.49%	17.81%
Annualized Std	9.63%	27.51%	33.41%	29.62%	25.84%	33.41%	27.00%
Sharpe ratio	1.49	1.16	1.12	1.40	1.35	1.10	0.74
Max drawdown	20.93%	28.26%	27.76%	21.13%	22.47%	28.11%	33.92%

Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

Reinforcement Learning (RL)
FinRL
Performance of multiple stock trading and portfolio allocation
over the DJIA constituents stocks using the FinRL library

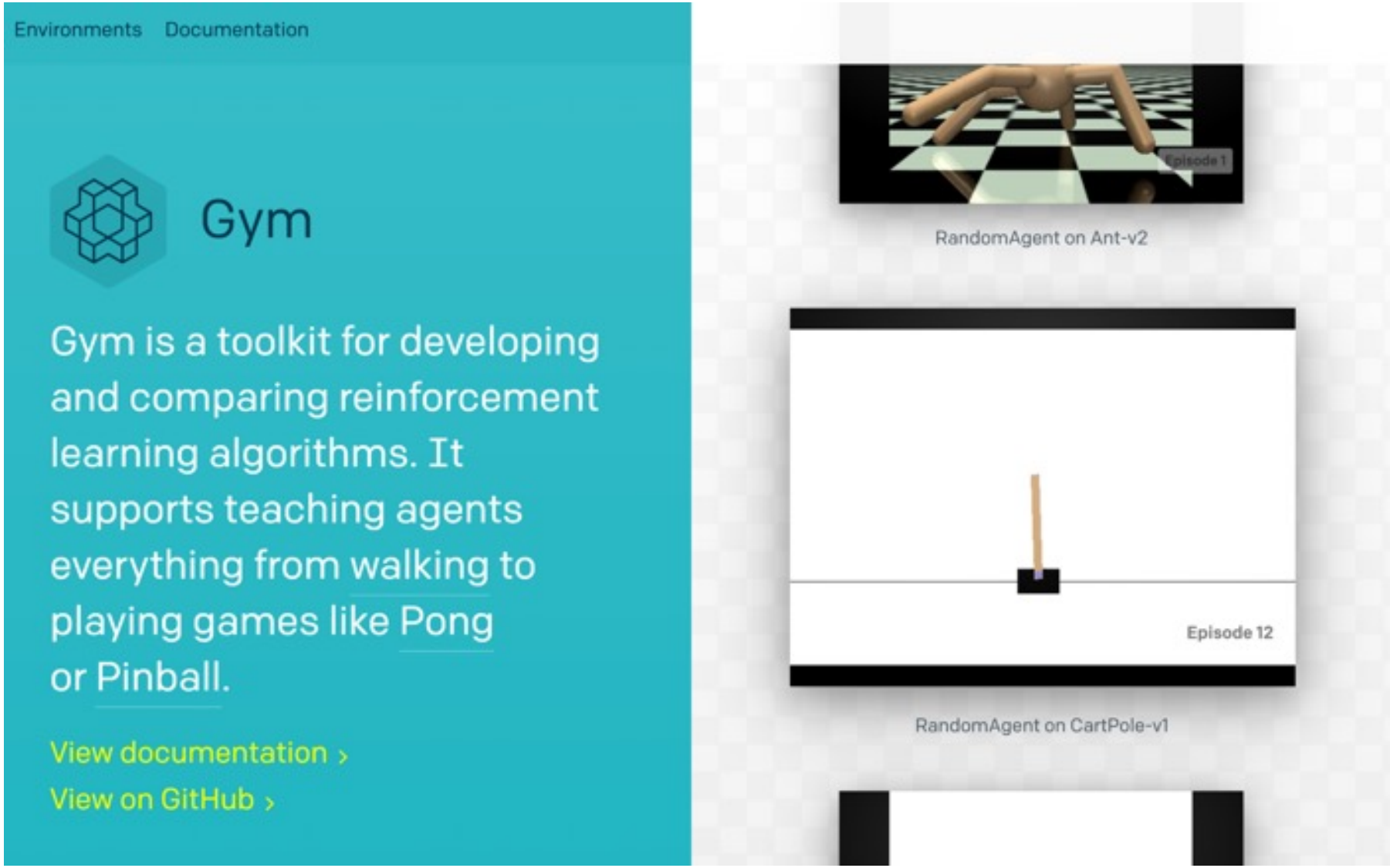
2019/01/01-2020/09/23	TD3	DDPG	Min-Var.	DJIA
Initial value	1,000,000	1,000,000	1,000,000	1,000,000
Final value	1,403,337; 1,381,120	1,396,607; 1,281,120	1,171,120	1,185,260
Annualized return	21.40%; 17.61%	20.34%; 15.81%	8.38%	10.61%
Annualized Std	14.60%; 17.01%	15.89%; 16.60%	26.21%	28.63%
Sharpe ratio	1.38; 1.03	1.28; 0.98	0.44	0.48
Max drawdown	11.52% 12.78%	13.72%; 13.68%	34.34%	37.01%

Source: Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).

Deep Reinforcement Learning Library

- OpenAI Gym
- Google Dopamine
- RLlib
- Horizon
- FinRL

Open AI Gym



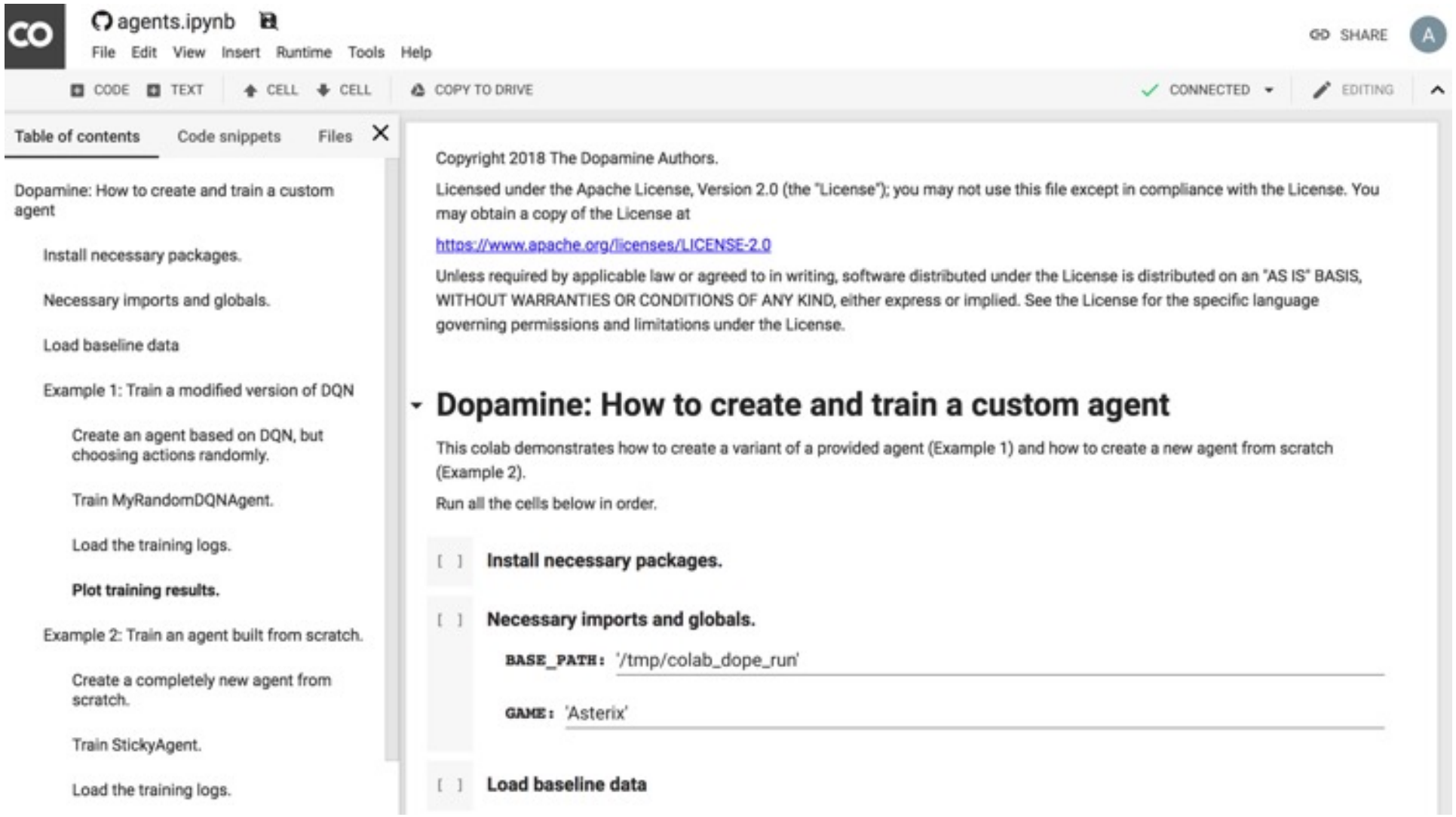
https://gym.openai.com/

Google Dopamine

Dopamine is a research framework
for fast prototyping of
reinforcement learning algorithms.

https://github.com/google/dopamine

Deep Reinforcement Learning
Dopamine Colab Examples
DQN Rainbow

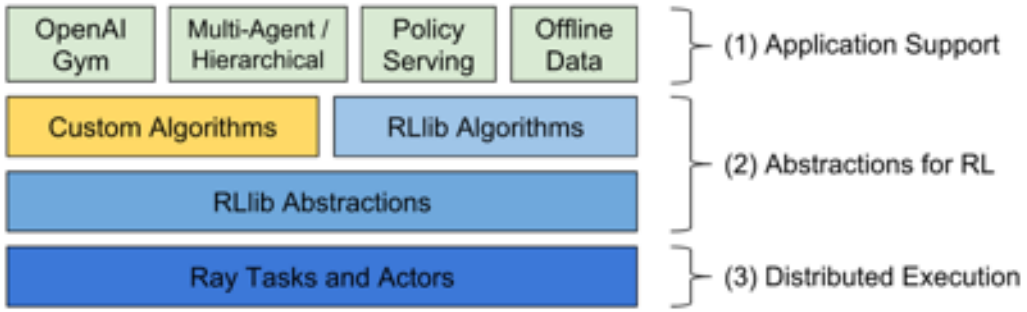


https://colab.research.google.com/github/google/dopamine/blob/master/dopamine/colab/agents.ipynb

RLlib:
Scalable Reinforcement Learning

RLlib: Scalable Reinforcement Learning

RLlib is an open-source library for reinforcement learning that offers both high scalability and a unified API for a variety of applications. RLlib natively supports TensorFlow, TensorFlow Eager, and PyTorch, but most of its internals are framework agnostic.



To get started, take a look over the custom env example and the API documentation. If you're looking to develop custom algorithms with RLlib, also check out concepts and custom algorithms.

RLlib in 60 seconds

The following is a whirlwind overview of RLlib. For a more in-depth guide, see also the full table of contents and RLlib blog posts. You may also want to skim the list of built-in algorithms. Look out for the and icons to see which algorithms are available for each framework.

v: master

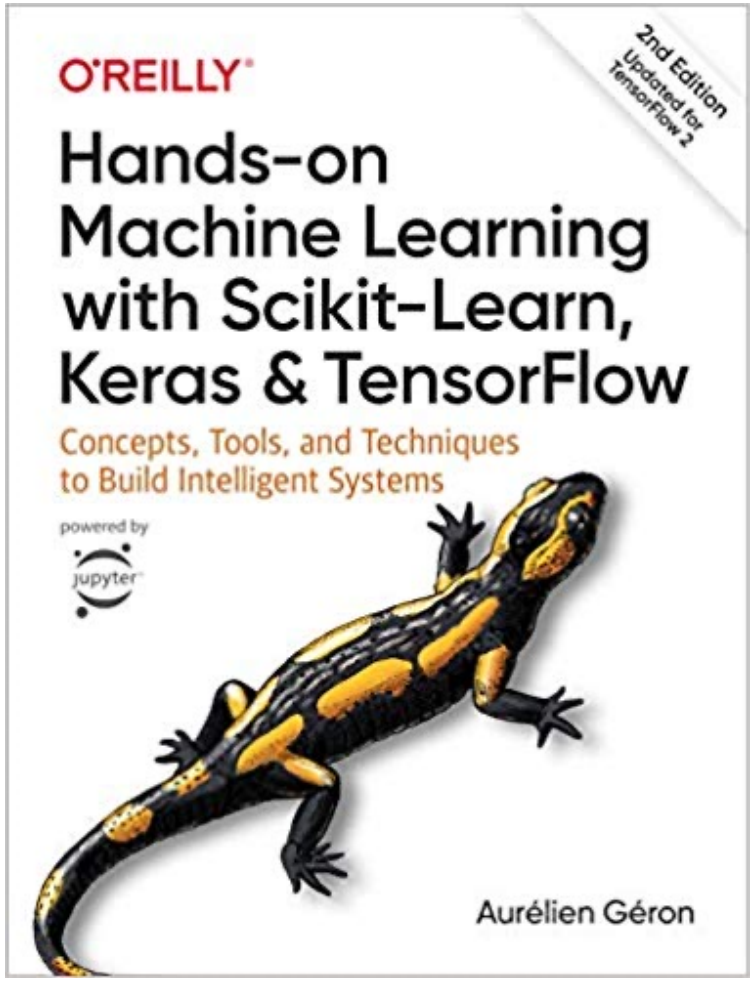
https://docs.ray.io/en/master/rllib.html

Aurélien Géron (2019),

Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow:

Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition

O'Reilly Media, 2019



https://github.com/ageron/handson-ml2

Source: https://www.amazon.com/Hands-Machine-Learning-Scikit-Learn-TensorFlow/dp/1492032646/

Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow

Notebooks

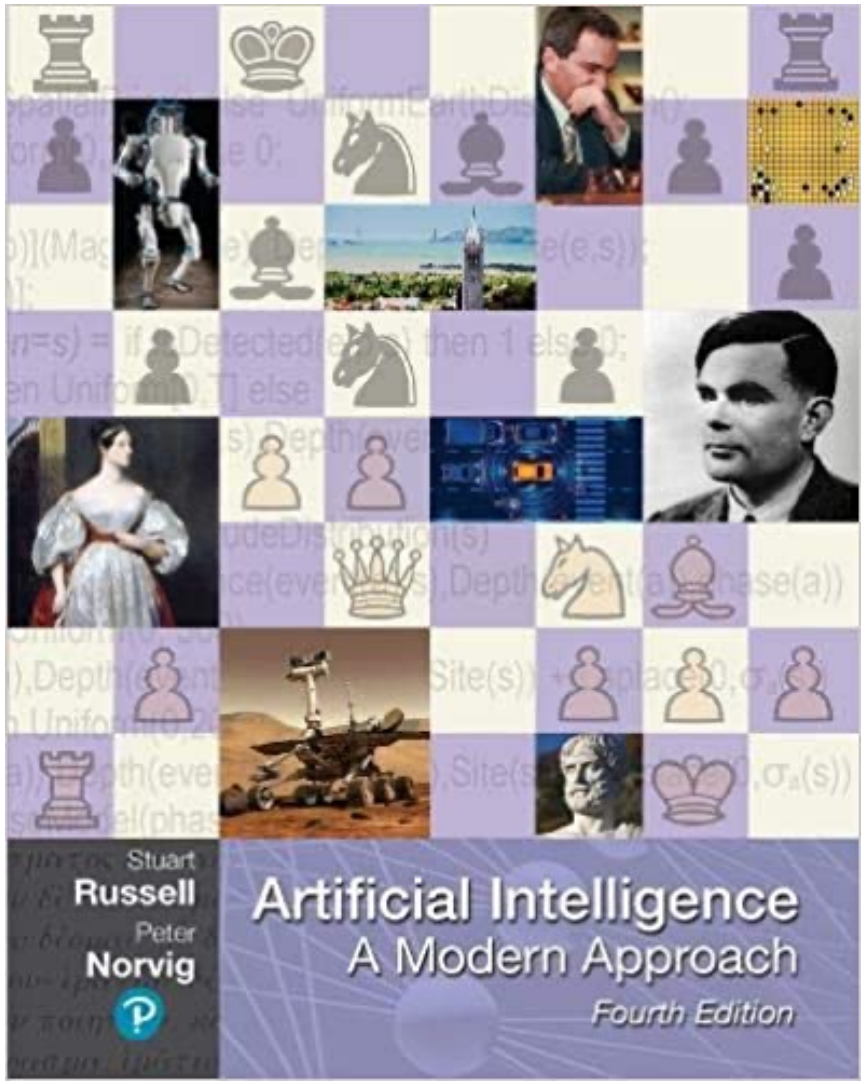
1. The Machine Learning landscape
2. End-to-end Machine Learning project
3. Classification
4. Training Models
5. Support Vector Machines
6. Decision Trees
7. Ensemble Learning and Random Forests
8. Dimensionality Reduction
9. Unsupervised Learning Techniques
10. Artificial Neural Nets with Keras
11. Training Deep Neural Networks
12. Custom Models and Training with TensorFlow
13. Loading and Preprocessing Data
14. Deep Computer Vision Using Convolutional Neural Networks
15. Processing Sequences Using RNNs and CNNs
16. Natural Language Processing with RNNs and Attention
17. Representation Learning Using Autoencoders
18. Reinforcement Learning
19. Training and Deploying TensorFlow Models at Scale

https://github.com/ageron/handson-ml2

Stuart Russell and Peter Norvig (2020),

Artificial Intelligence: A Modern Approach,

4th Edition, Pearson



Source: Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson

https://www.amazon.com/Artificial-Intelligence-A-Modern-Approach/dp/0134610997/

Artificial Intelligence: A Modern Approach (AIMA)

- **Artificial Intelligence: A Modern Approach (AIMA)**
  - http://aima.cs.berkeley.edu/
- **AIMA Python**
  - http://aima.cs.berkeley.edu/python/readme.html
  - https://github.com/aimacode/aima-python
- **Learning**
  - http://aima.cs.berkeley.edu/python/learning.html

Artificial Intelligence: A Modern Approach (AIMA)

Artificial Intelligence: A Modern Approach, 4th US ed.

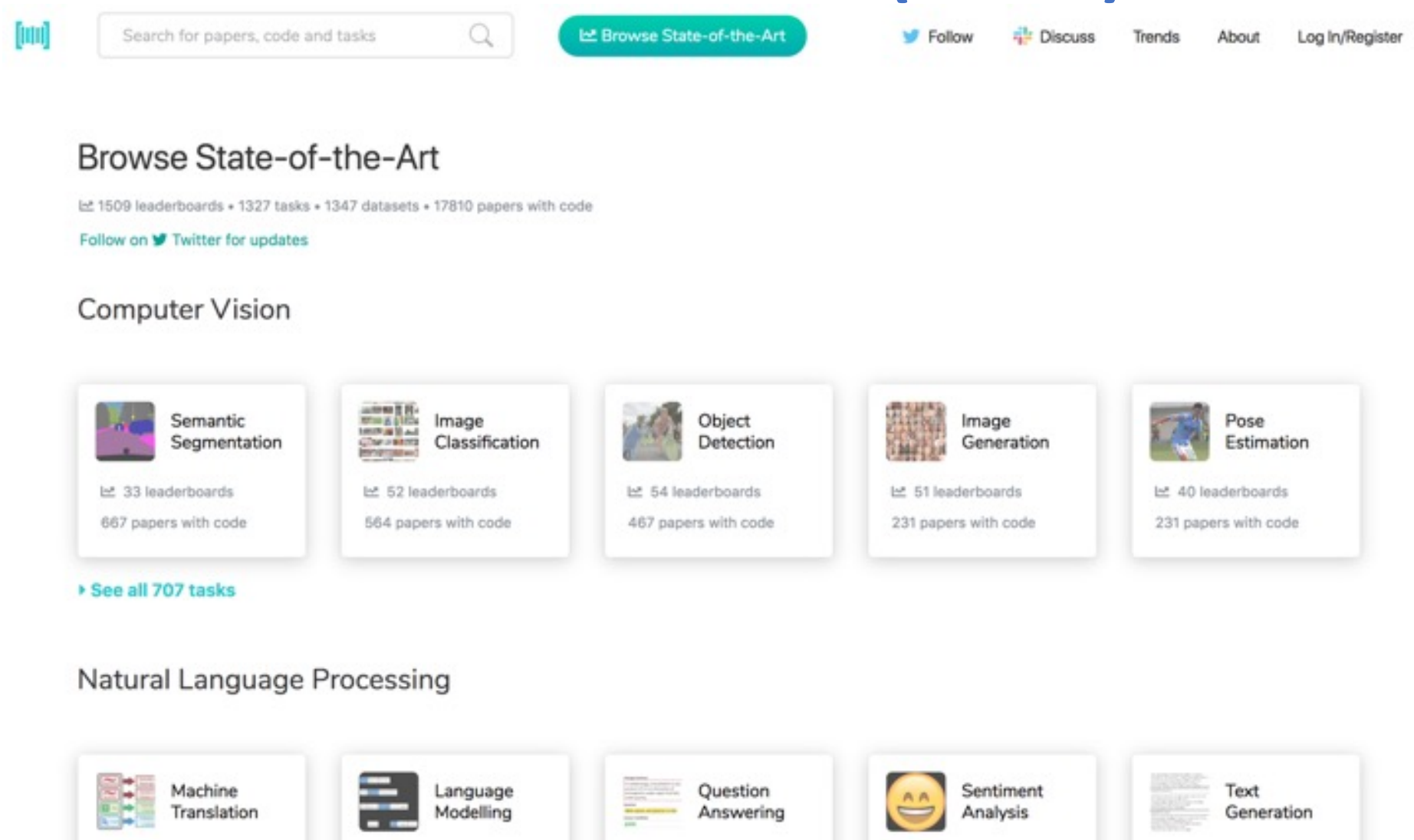
by Stuart Russell and Peter Norvig

The authoritative, most-used AI textbook, adopted by over **1500** schools.

Table of Contents for the US Edition (or see the Global Edition)

http://aima.cs.berkeley.edu/

Papers with Code
State-of-the-Art (SOTA)



https://paperswithcode.com/sota

Summary

- Deep Learning (DL)
  - Neural Networks (NN)
  - Convolutional Neural Networks (CNN)
  - Recurrent Neural Networks (RNN)
- Reinforcement Learning (RL)
  - Markov Decision Processes (MDP)
  - Deep Reinforcement Learning (DRL) Algorithms
    - SARSA
    - Q-Learning
    - DQN, A3C, Rainbow

References

- Stuart Russell and Peter Norvig (2020), Artificial Intelligence: A Modern Approach, 4th Edition, Pearson.
- Aurélien Géron (2019), Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition, O'Reilly Media.
- Steven D'Ascoli (2022), Artificial Intelligence and Deep Learning with Python: Every Line of Code Explained For Readers New to AI and New to Python, Independently published.
- Nithin Buduma, Nikhil Buduma, Joe Papa (2022), Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms, 2nd Edition, O'Reilly Media.
- Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.
- Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu (2020), "Financial time series forecasting with deep learning: A systematic literature review: 2005–2019." Applied Soft Computing 90 (2020): 106181.
- Deep Learning Basics: Neural Networks Demystified, https://www.youtube.com/playlist?list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU
- Deep Learning SIMPLIFIED, https://www.youtube.com/playlist?list=PLjJh1vlSEYgvGod9wWiydumYl8hOXixNu
- 3Blue1Brown (2017), But what *is* a Neural Network? | Chapter 1, deep learning, https://www.youtube.com/watch?v=aircAruvnKk
- 3Blue1Brown (2017), Gradient descent, how neural networks learn | Chapter 2, deep learning, https://www.youtube.com/watch?v=IHZwWFHWa-w
- 3Blue1Brown (2017), What is backpropagation really doing? | Chapter 3, deep learning, https://www.youtube.com/watch?v=Ilg3gGewQ5U
- Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.
- David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ
- David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, Demis Hassabis (2018), "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play." Science 362, no. 6419 (2018): 1140-1144.
- David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis (2017), "Mastering the game of Go without human knowledge." Nature 550 (2017): 354–359.
- Hado Van Hasselt, Arthur Guez, and David Silver (2016). "Deep Reinforcement Learning with Double Q-Learning." In AAAI, vol. 2, p. 5. 2016.
- Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver (2017). "Rainbow: Combining improvements in deep reinforcement learning." arXiv preprint arXiv:1710.02298 (2017).
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. (2015) "Human-level control through deep reinforcement learning." Nature 518, no. 7540 (2015): 529.
- Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas (2015). "Dueling network architectures for deep reinforcement learning." arXiv preprint arXiv:1511.06581 (2015).
- Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang (2020). "FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance." arXiv preprint arXiv:2011.09607 (2020).
- Mu-En Wu, Jia-Hao Syu, Jerry Chun-Wei Lin, and Jan-Ming Ho. "Portfolio management system in equity market neutral using reinforcement learning." Applied Intelligence (2021): 1-13.
- Min-Yuh Day (2022), Python 101, https://tinyurl.com/aintpupython101