智慧金融量化分析

(Artificial Intelligence in Finance and Quantitative Analysis)

財務金融深度學習 (Deep Learning in Finance)

財務金融強化學習 (Reinforcement Learning in Finance)

1101AIFQA09
MBA, IM, NTPU (M6132) (Fall 2021)
Tue 2, 3, 4 (9:10-12:00) (8F40)

戴敏育 副教授

Min-Yuh Day, Ph.D, Associate Professor

國立臺北大學 資訊管理研究所

Institute of Information Management, National Taipei University

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

2021-12-21

課程大綱 (Syllabus)

國立臺北大學 National Taipei University

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2021/09/28	智慧金融量化分析概論 (Introduction to Artificial Intelligence in Finance and Quantitative Analysis)
2	2021/10/05	AI 金融科技: 金融服務創新應用 (AI in FinTech: Financial Services Innovation and Application)
3	2021/10/12	投資心理學與行為財務學 (Investing Psychology and Behavioral Finance)
4	2021/10/19	財務金融事件研究法 (Event Studies in Finance)
5	2021/10/26	智慧金融量化分析個案研究 I (Case Study on AI in Finance and Quantitative Analysis I)
6	2021/11/02	財務金融理論 (Finance Theory)

課程大綱 (Syllabus)

國立臺北大學 National Taipei University

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
7	2021/11/09	數據驅動財務金融 (Data-Driven Finance)
8	2021/11/16	期中報告 (Midterm Project Report)
9	2021/11/23	金融計量經濟學 (Financial Econometrics)
10	2021/11/30	人工智慧優先金融 (AI-First Finance)
11	2021/12/07	智慧金融量化分析產業實務 (Industry Practices of AI in Finance and Quantitative Analysis)
12	2021/12/14	智慧金融量化分析個案研究 II (Case Study on AI in Finance and Quantitative Analysis II)

[演講主題：指數設計的方法論、數據分析與量化投資應用，演講者：李政剛，基金經理/元大投信]
[Invited Talk: Index Design – Methodology、Data Analysis and the Application of Quantitative Investing, Invited Speaker: Jervis J.G. Li, Fund Manager, Yuanta SITC]

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
13	2021/12/21	財務金融深度學習 (Deep Learning in Finance); 財務金融強化學習 (Reinforcement Learning in Finance)
14	2021/12/28	演算法交易 (Algorithmic Trading); 風險管理 (Risk Management); 交易機器人與基於事件的回測 (Trading Bot and Event-Based Backtesting)
15	2022/01/04	期末報告 I (Final Project Report I)
16	2022/01/11	期末報告 II (Final Project Report II)
17	2022/01/18	學生自主學習 (Self-learning)
18	2022/01/25	學生自主學習 (Self-learning)

Deep Learning in Finance

Reinforcement Learning in Finance

Outline

- Deep Learning (DL) in Finance
  - Dense Neural Networks (DNN)
  - Recurrent Neural Networks (RNN)
  - Convolutional Neural Networks (CNN)
- Reinforcement Learning (RL) in Finance
  - Q Learning (QL)
  - Improved Finance Environment
  - Improved Financial QL Agent

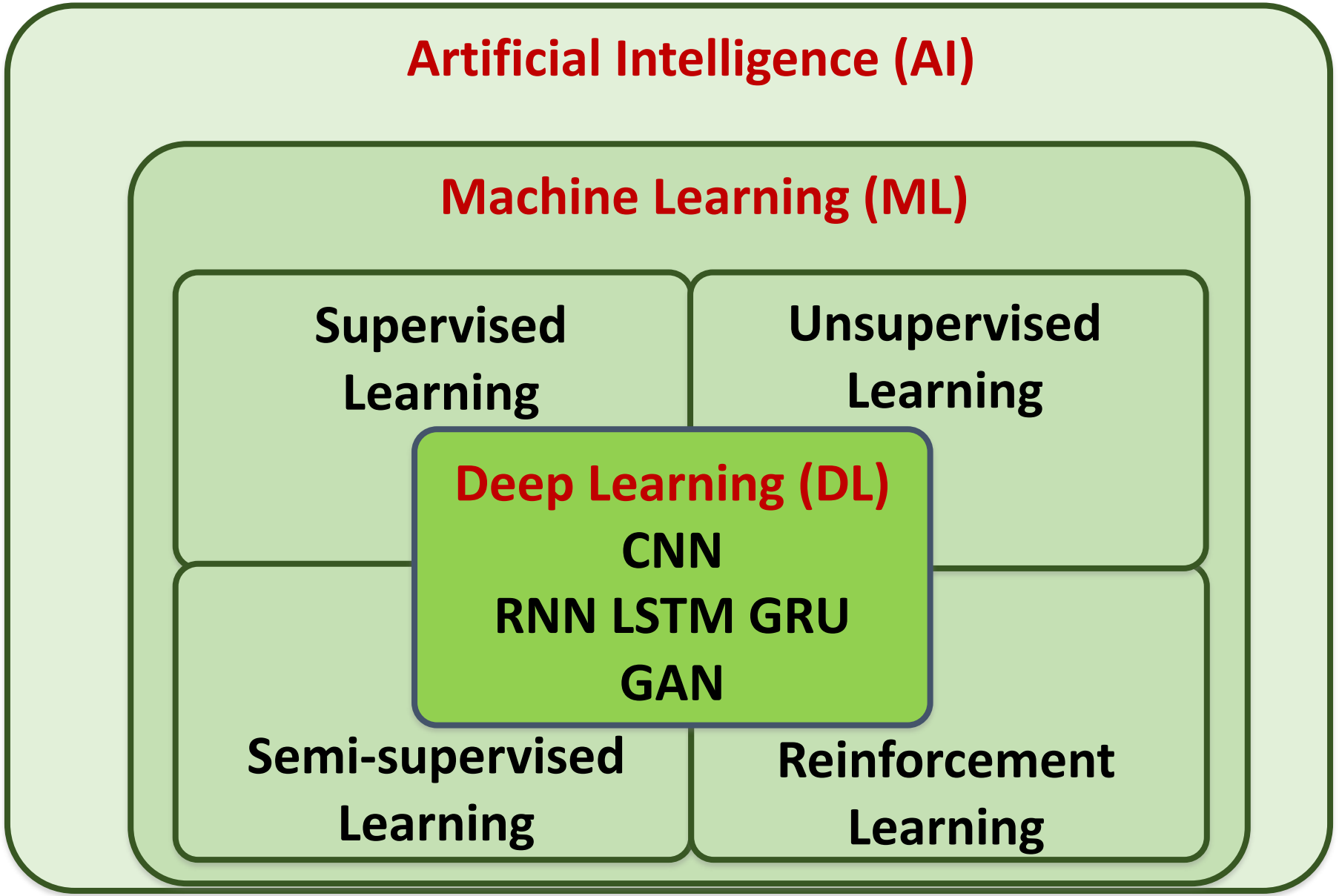
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Deep Learning in Finance

- Dense Neural Networks (DNN)
- Recurrent Neural Networks (RNN)
- Convolutional Neural Networks (CNN)

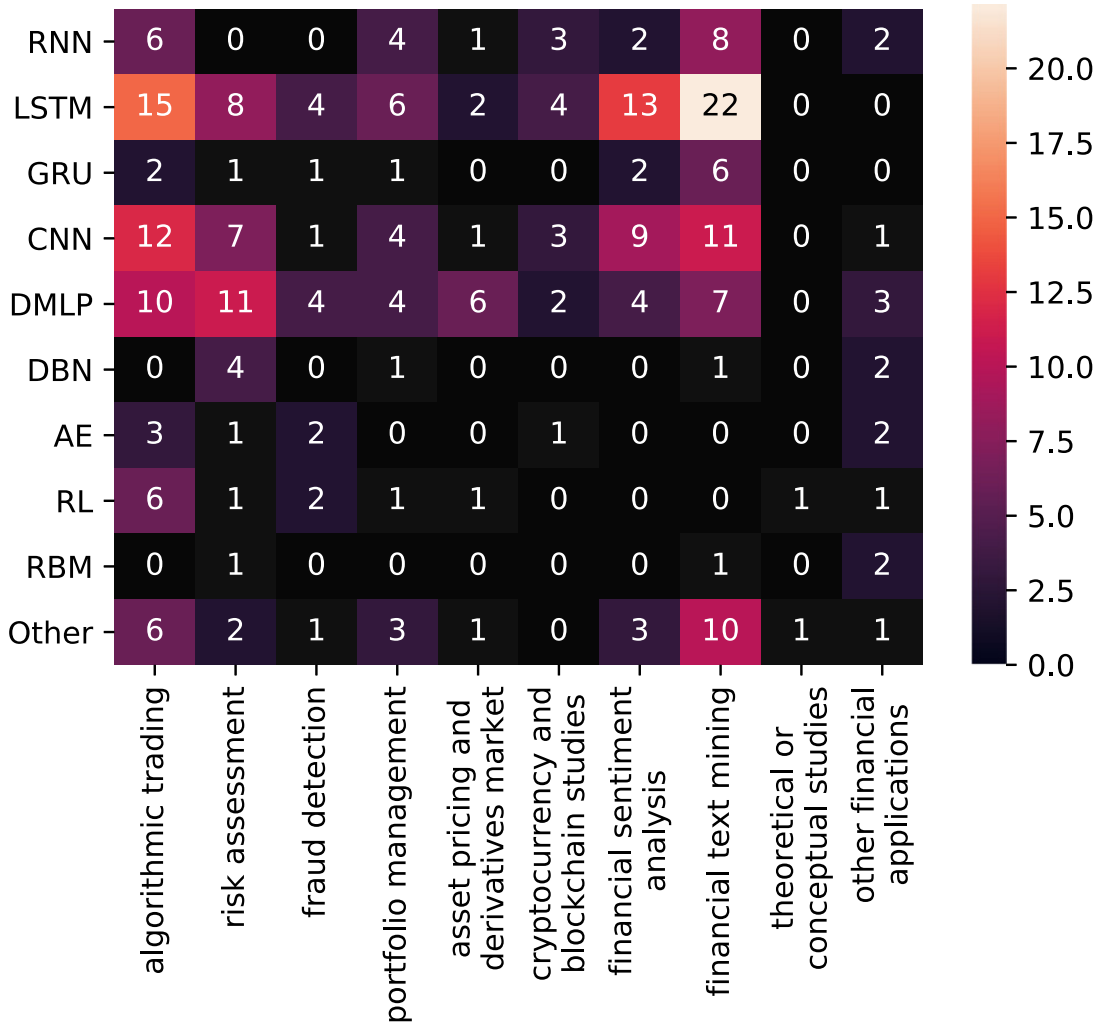
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

AI, ML, DL



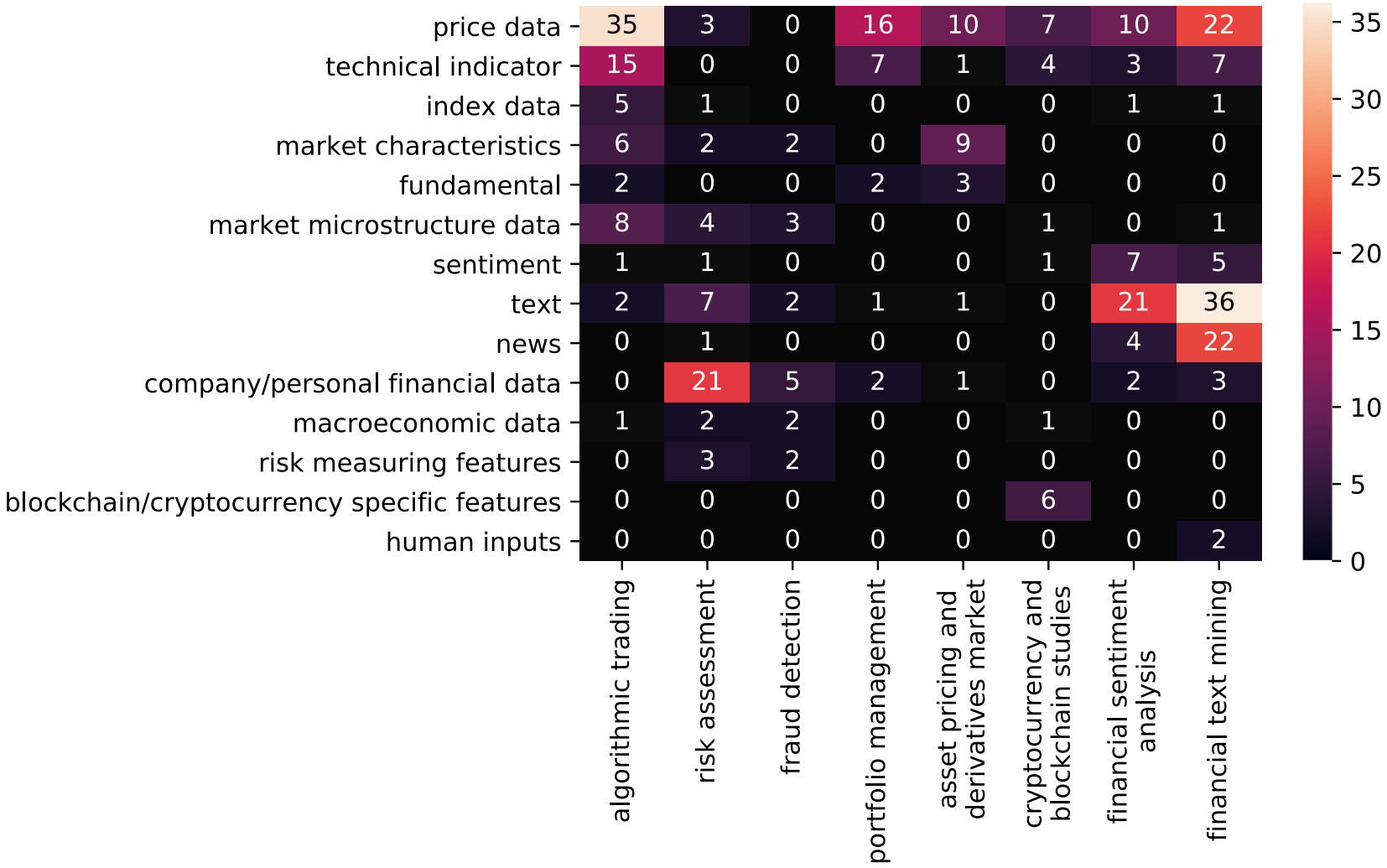
Source: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/deep_learning.html

Deep learning for financial applications: Topic-Model Heatmap



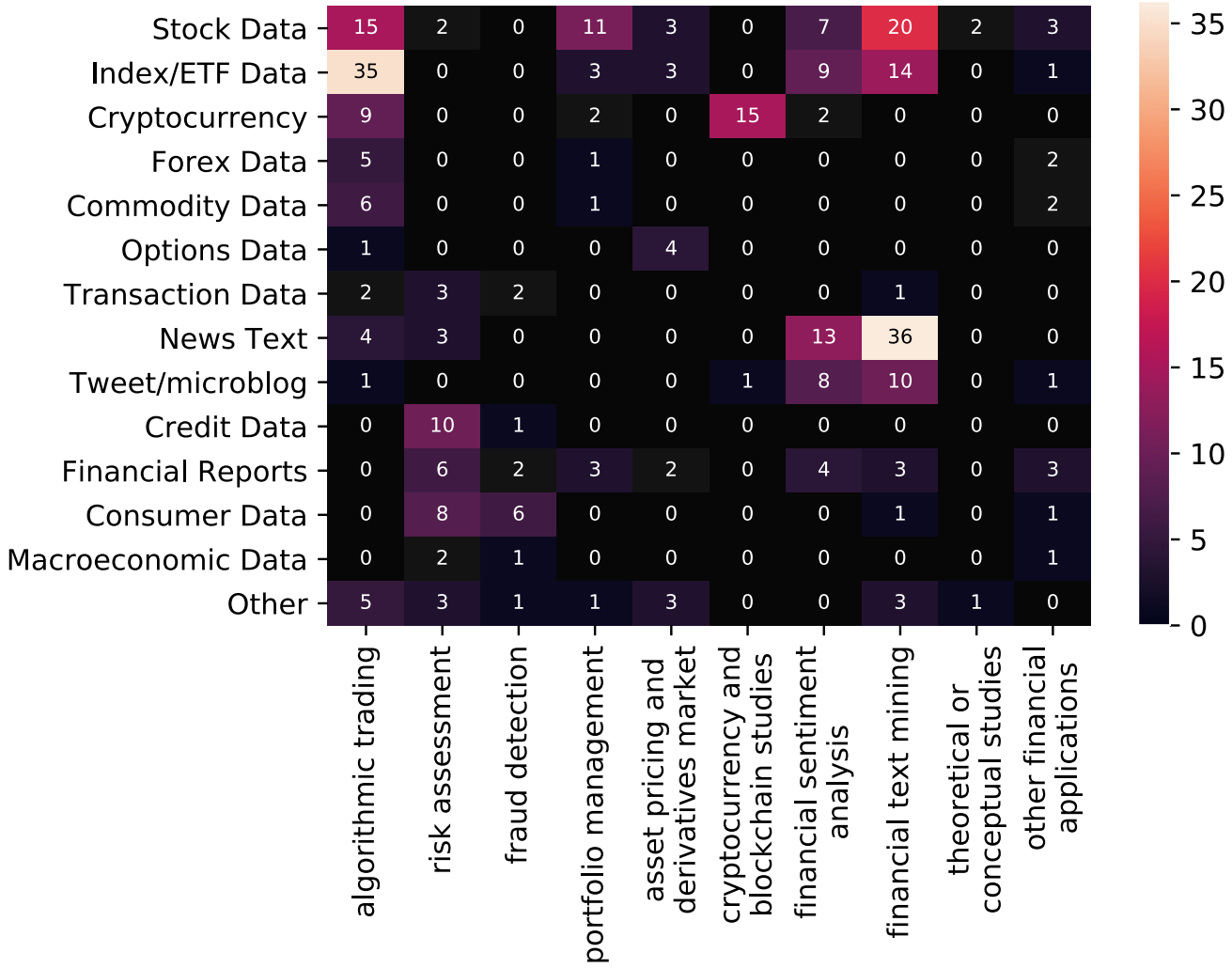
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Topic-Feature Heatmap



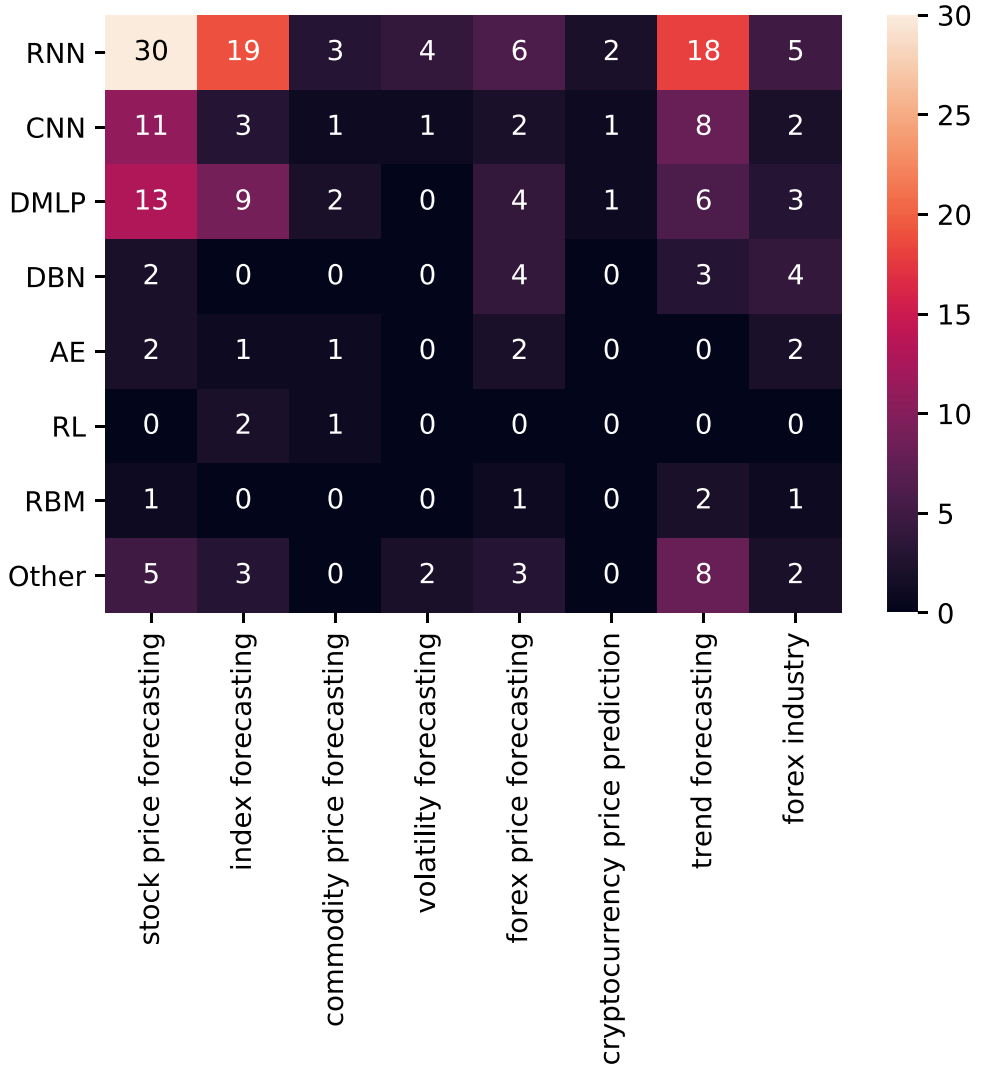
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Deep learning for financial applications: Topic-Dataset Heatmap



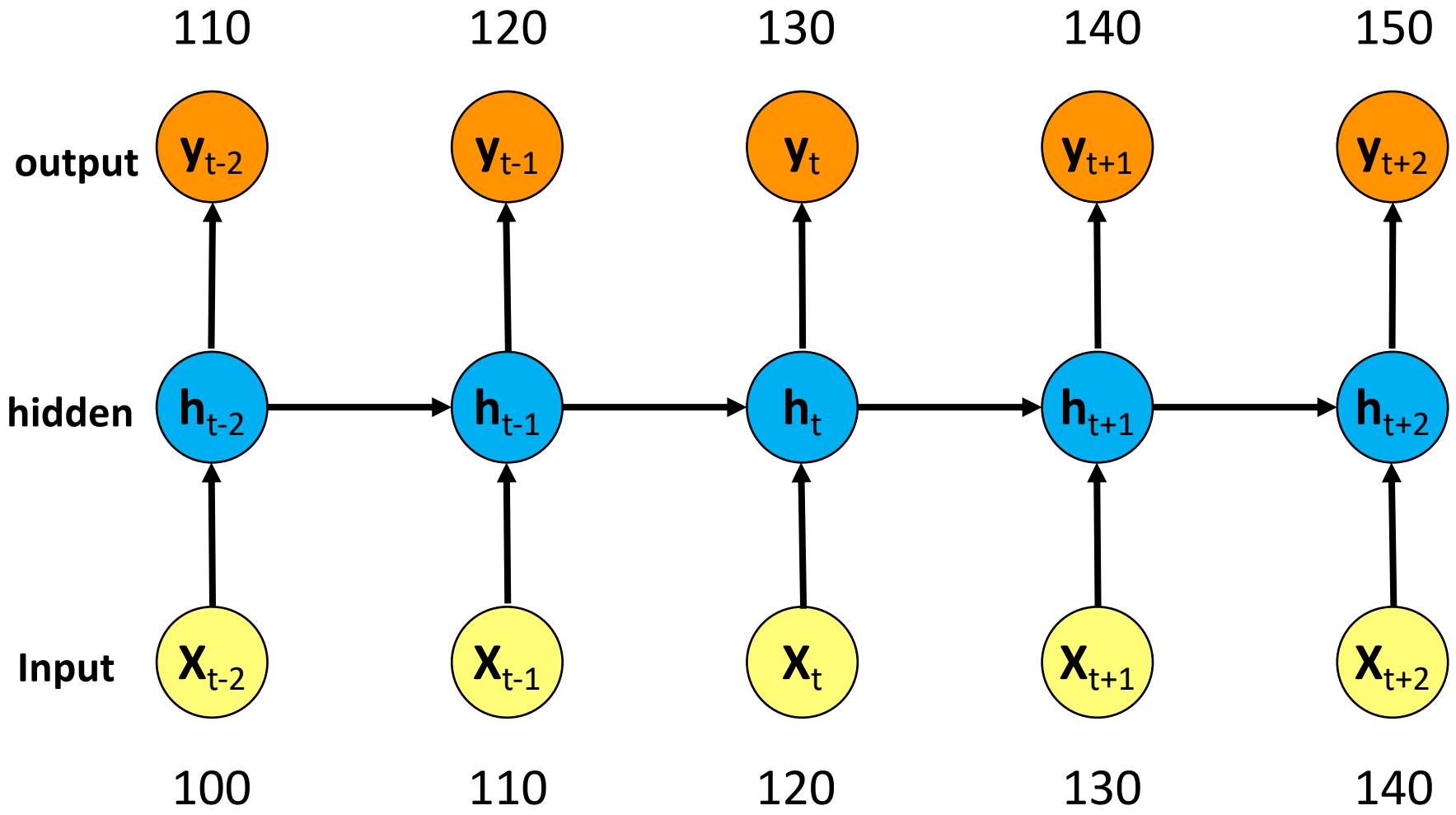
Source: Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer (2020). "Deep learning for financial applications: A survey." Applied Soft Computing (2020): 106384.

Financial time series forecasting with deep learning: Topic-model heatmap



Source: 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.

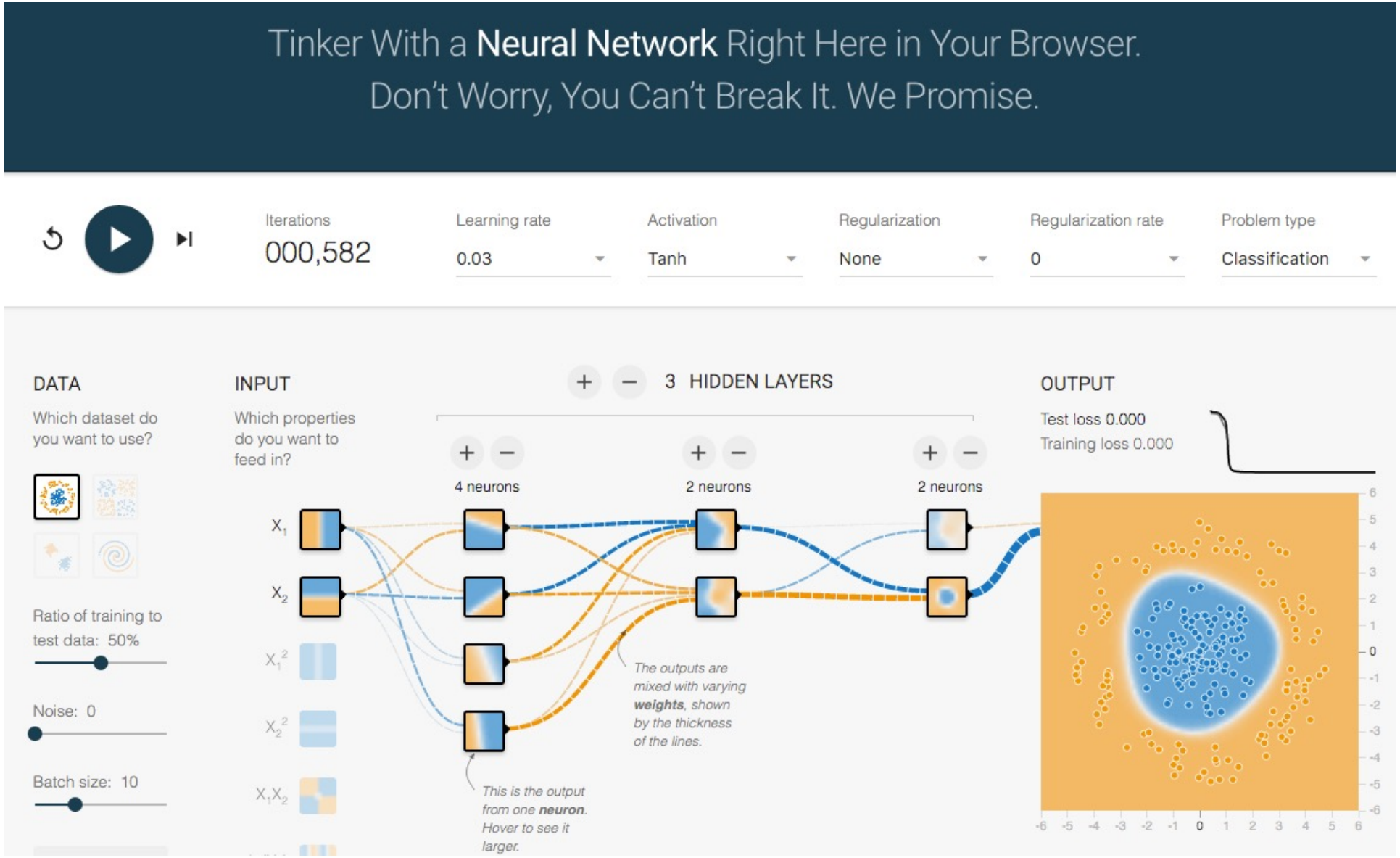
Recurrent Neural Networks (RNN)
Time Series Forecasting



Deep Learning

Deep Learning
and
Neural Networks

TensorFlow Playground



http://playground.tensorflow.org/

TensorFlow

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

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

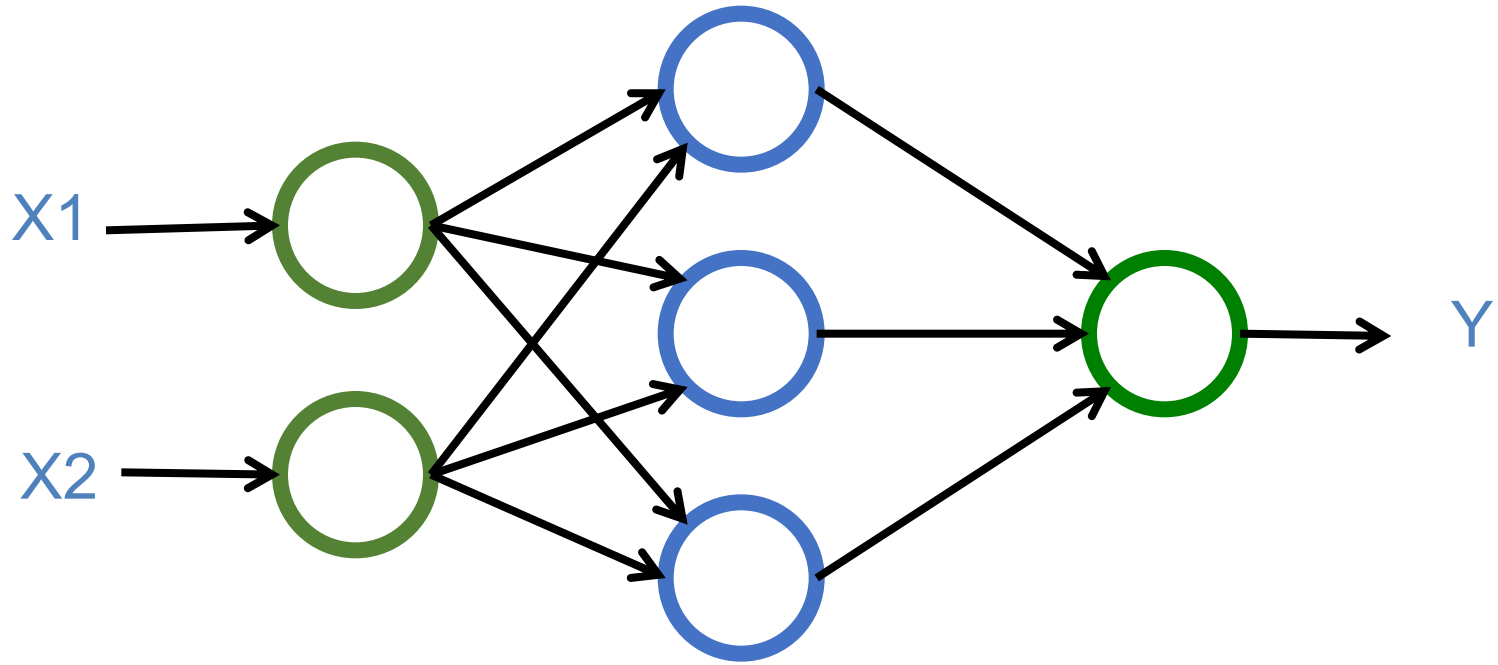
Deep Learning
and
Neural Networks

Deep Learning Foundations:

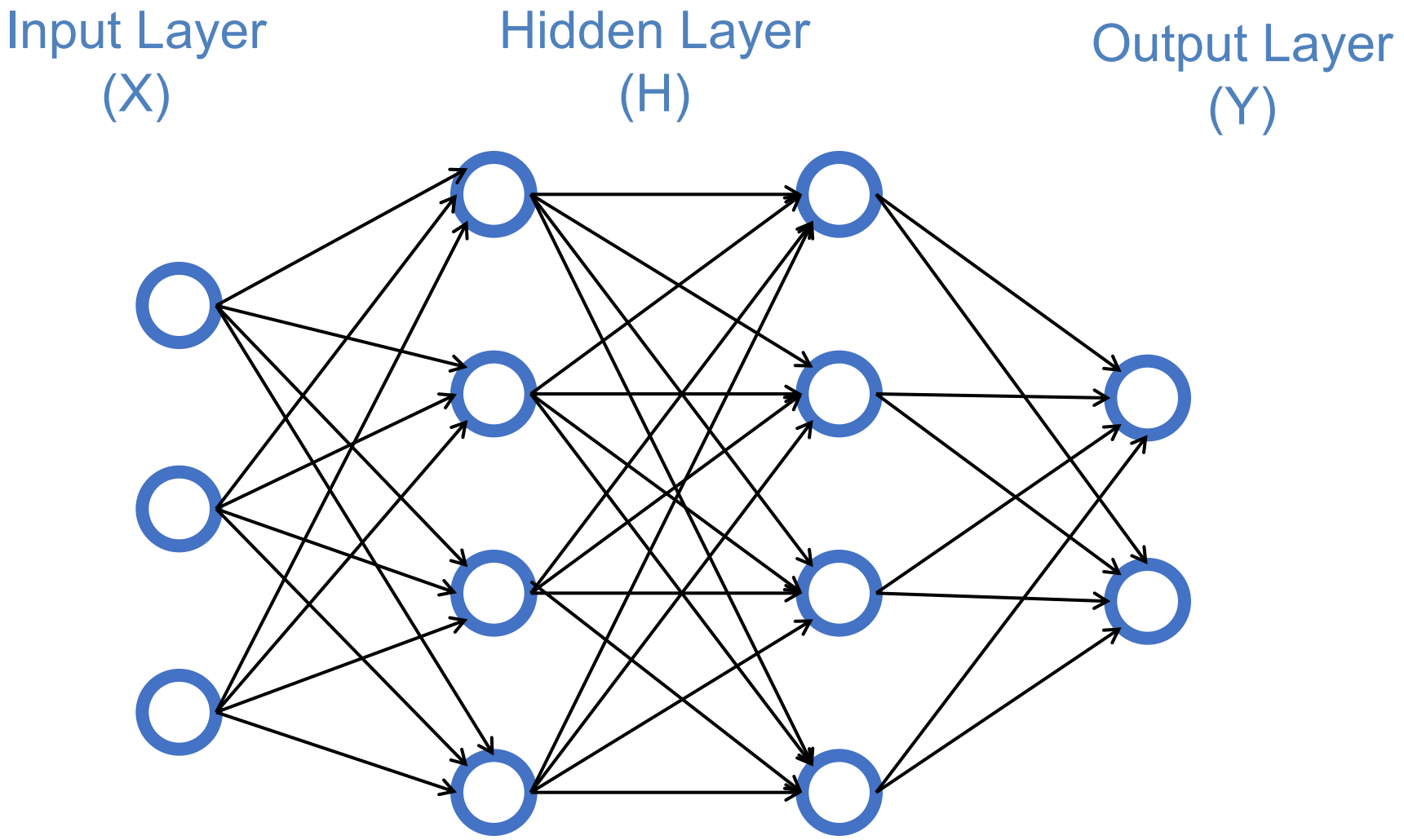
Neural Networks

Deep Learning and Neural Networks

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



Deep Learning and Neural Networks



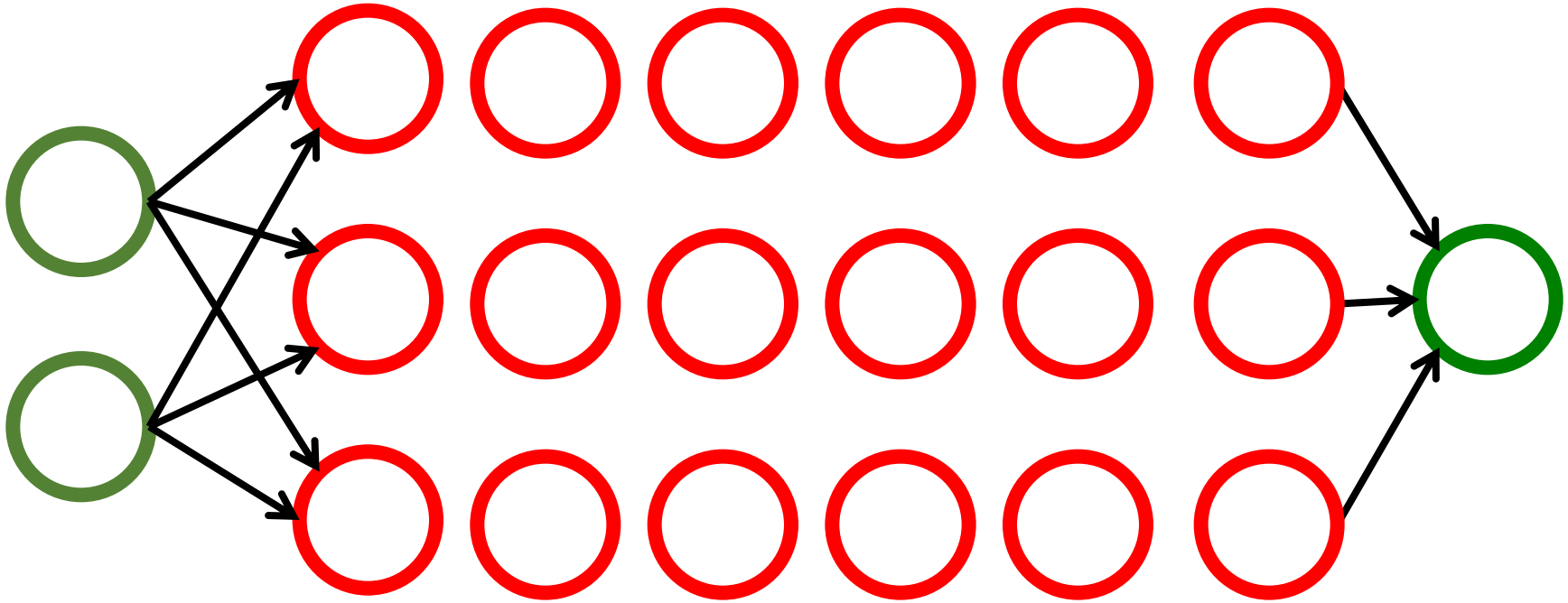
Deep Learning and Neural Networks

Input Layer (X)

Hidden Layers (H)

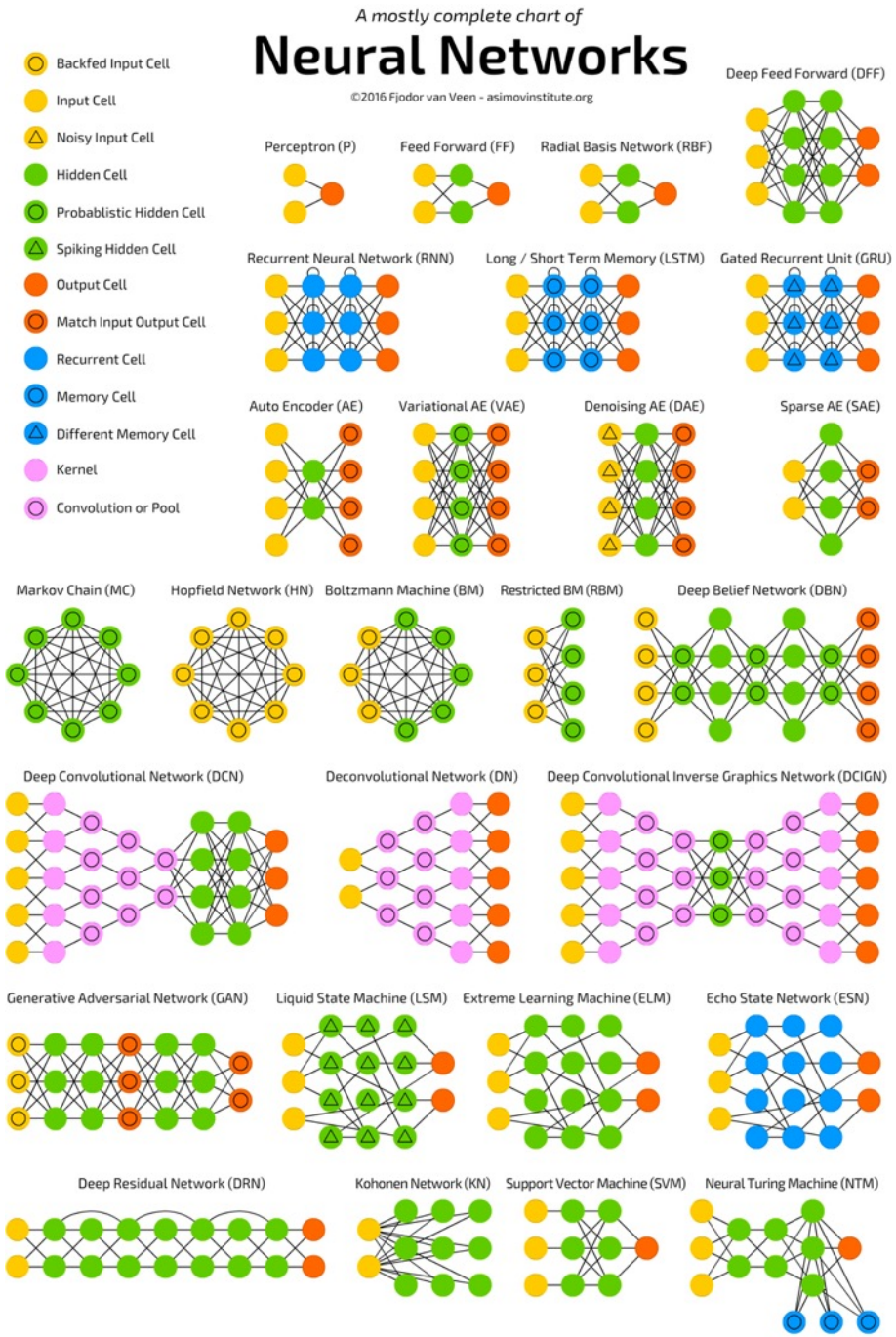
Output Layer (Y)

Deep Neural Networks
Deep Learning

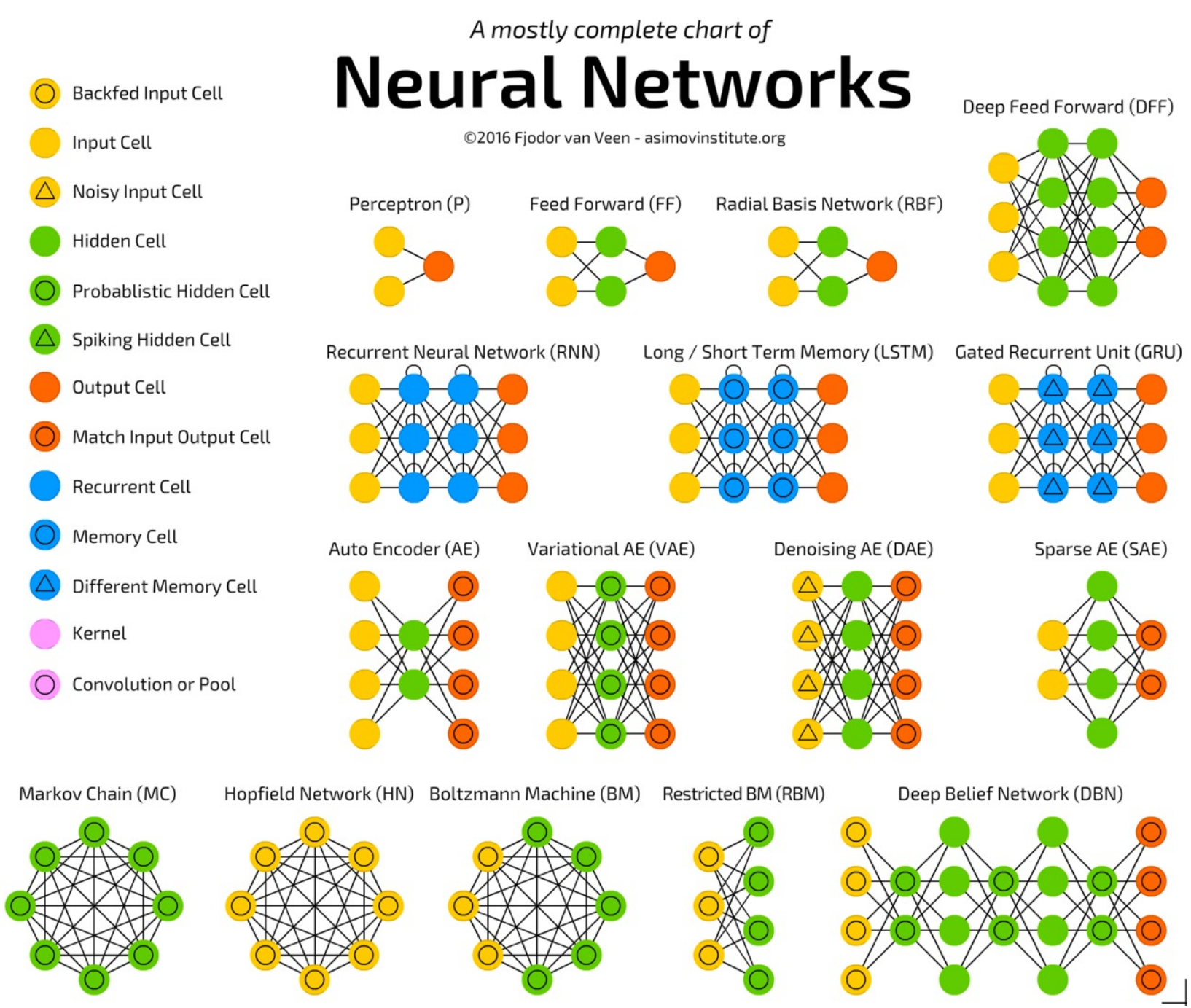


Deep Learning
and
Deep Neural Networks

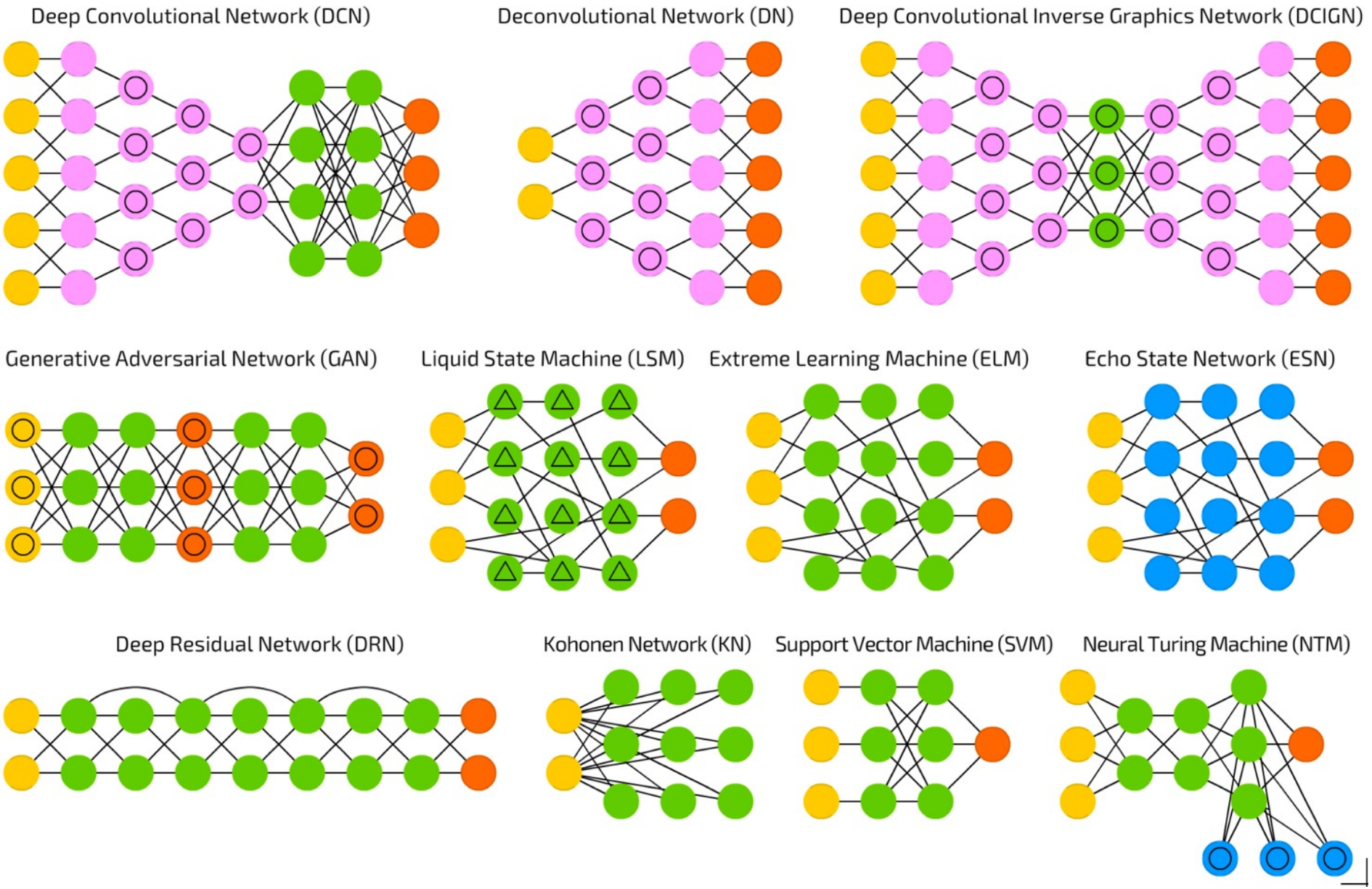
Neural Networks (NN)



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



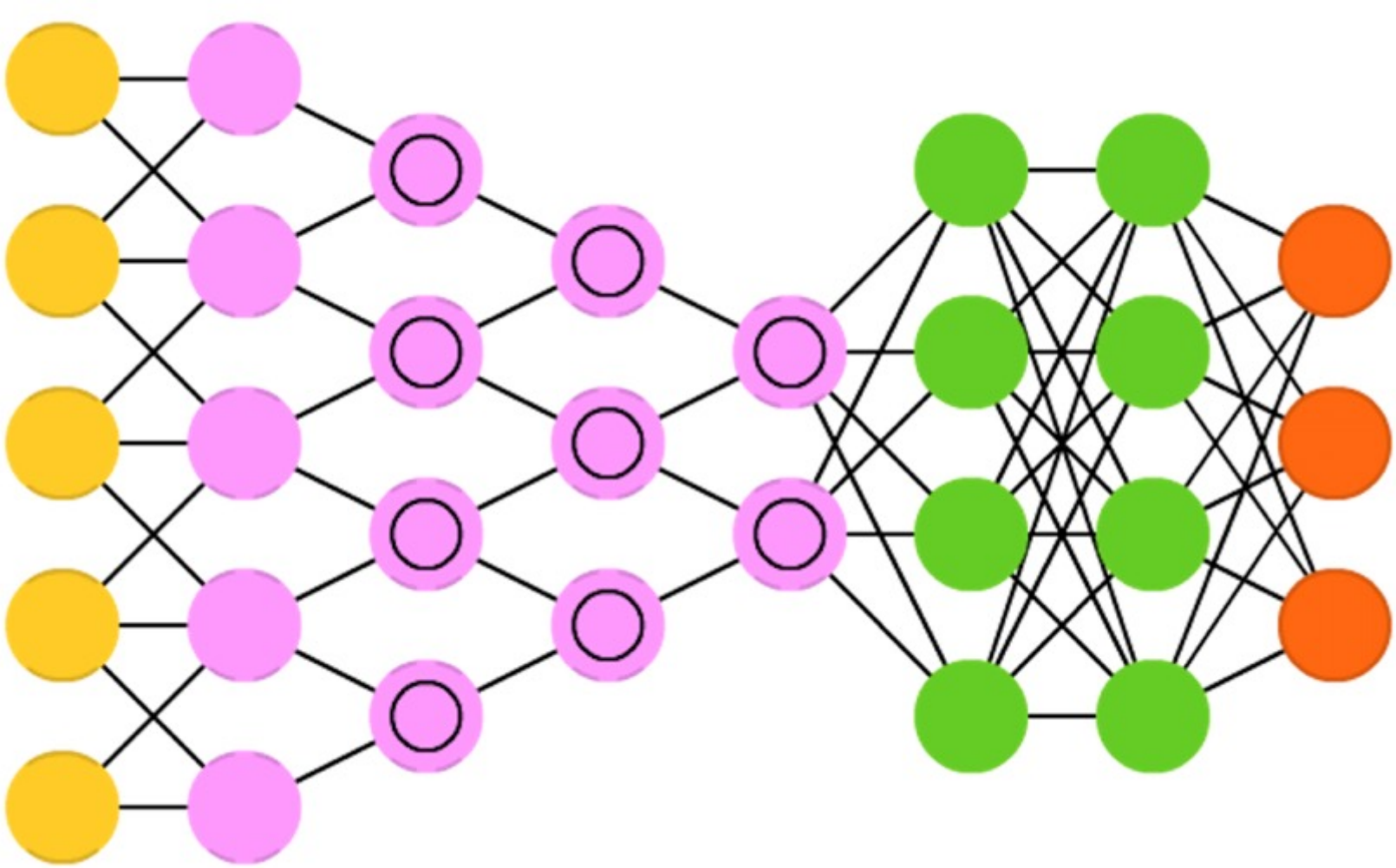
Source: http://www.asimovinstitute.org/neural-network-zoo/



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

Convolutional Neural Networks (CNN

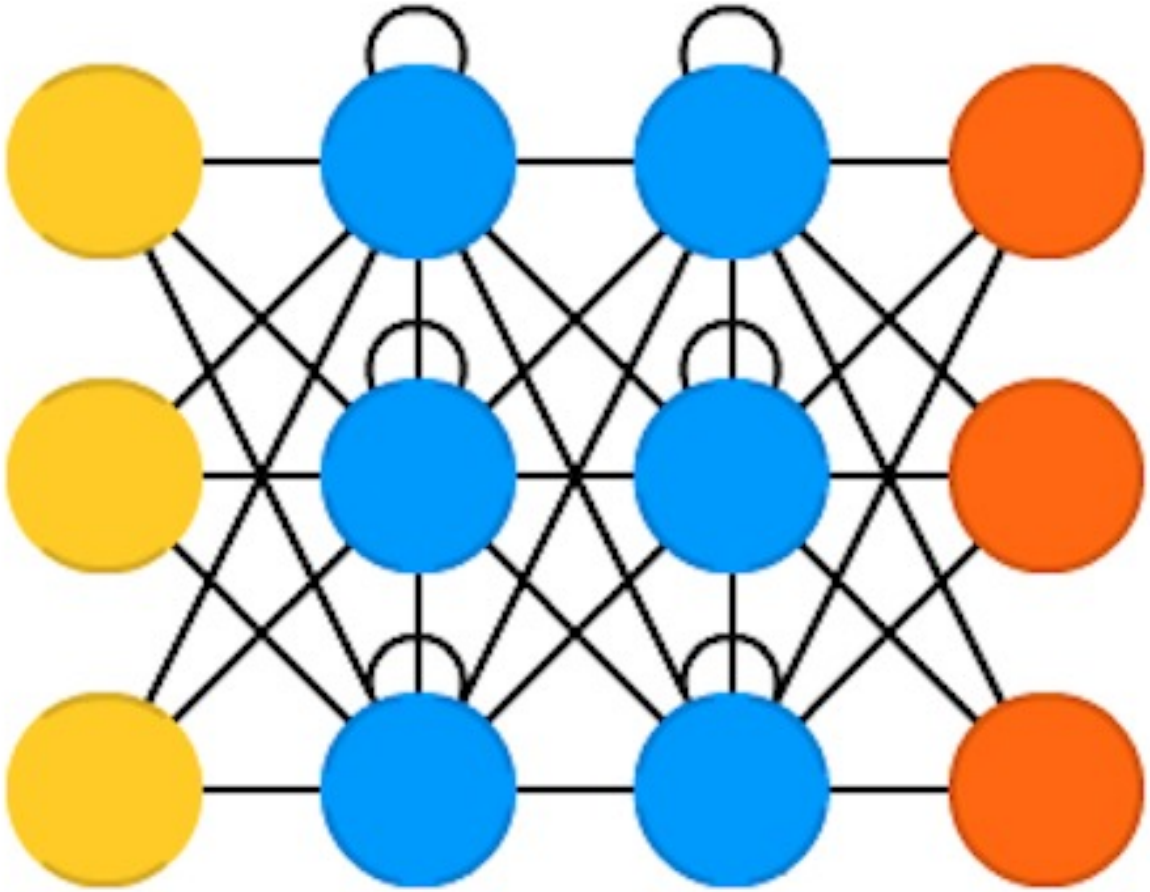
or Deep Convolutional Neural Networks, DCNN)



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/

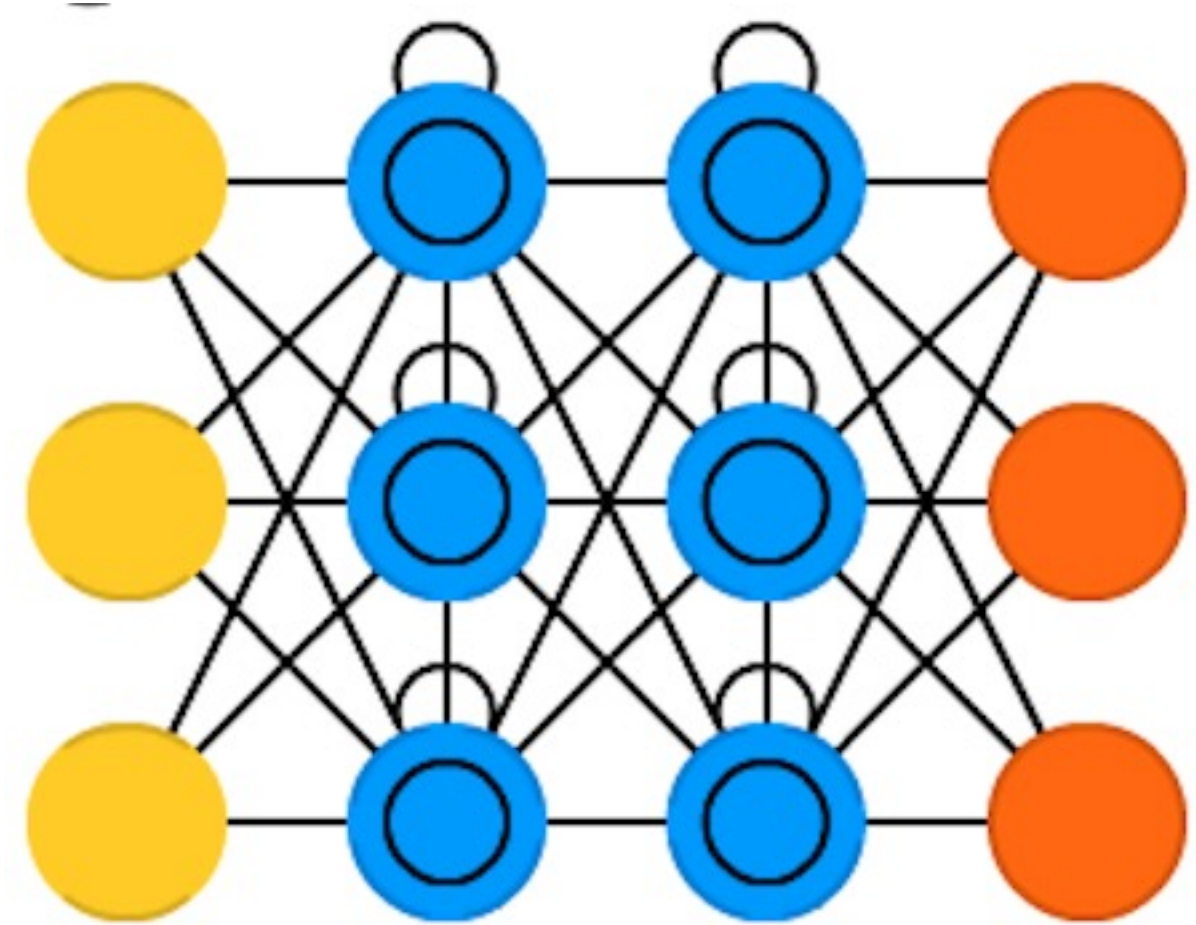
Recurrent Neural Networks (RNN)



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

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

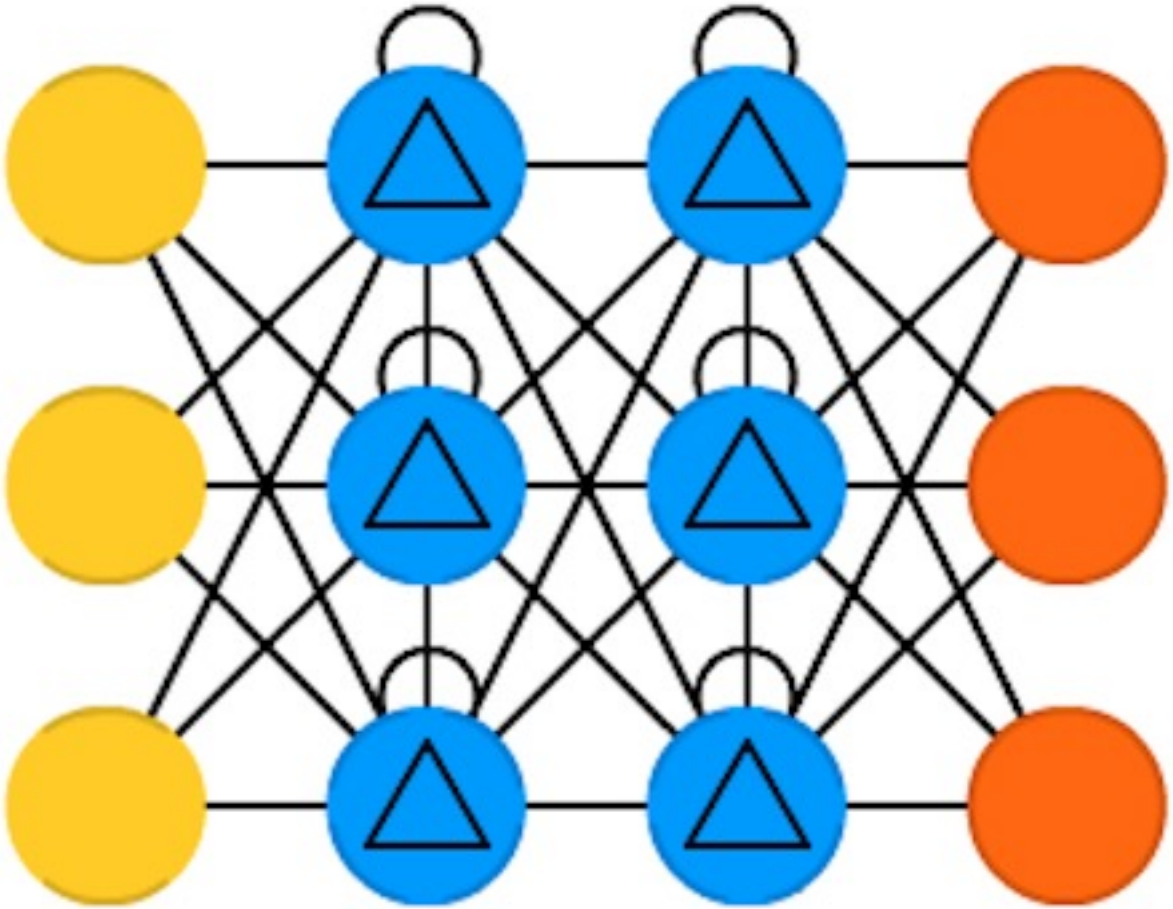
Long / Short Term Memory (LSTM)



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

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

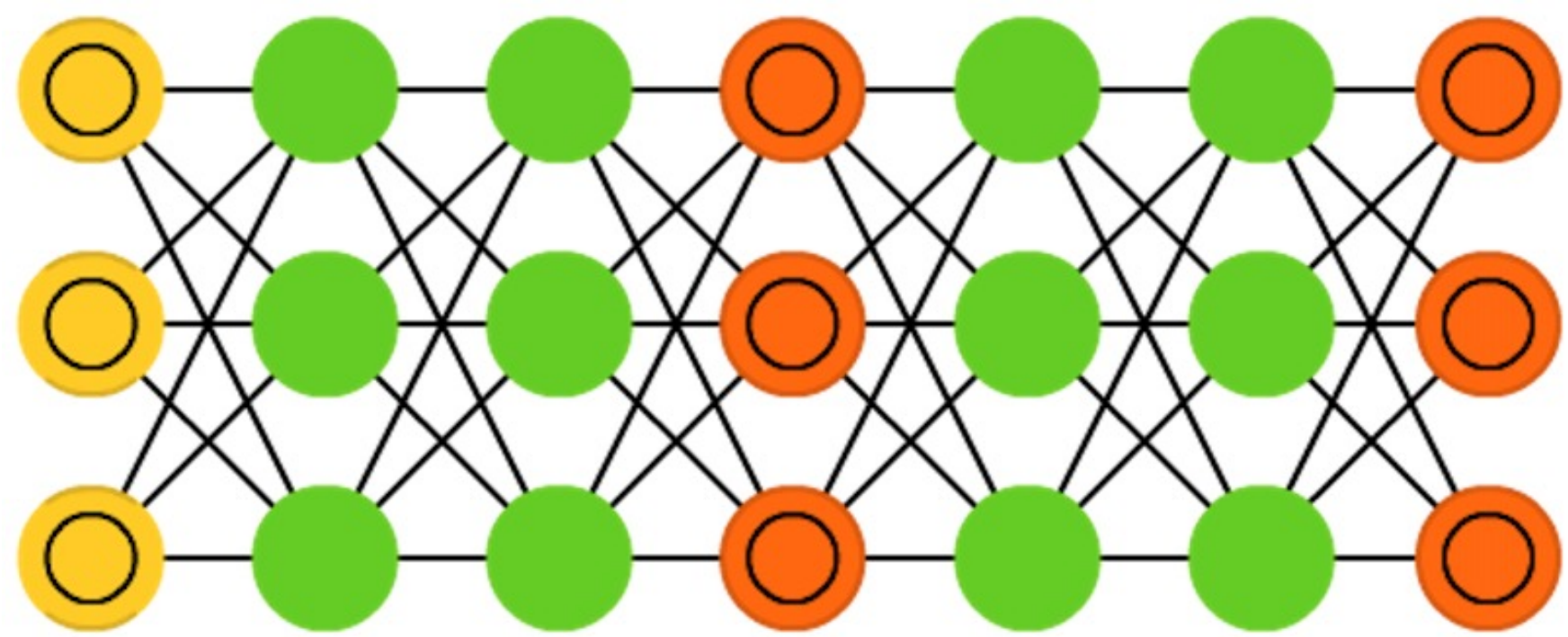
Gated Recurrent Units (GRU)



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/

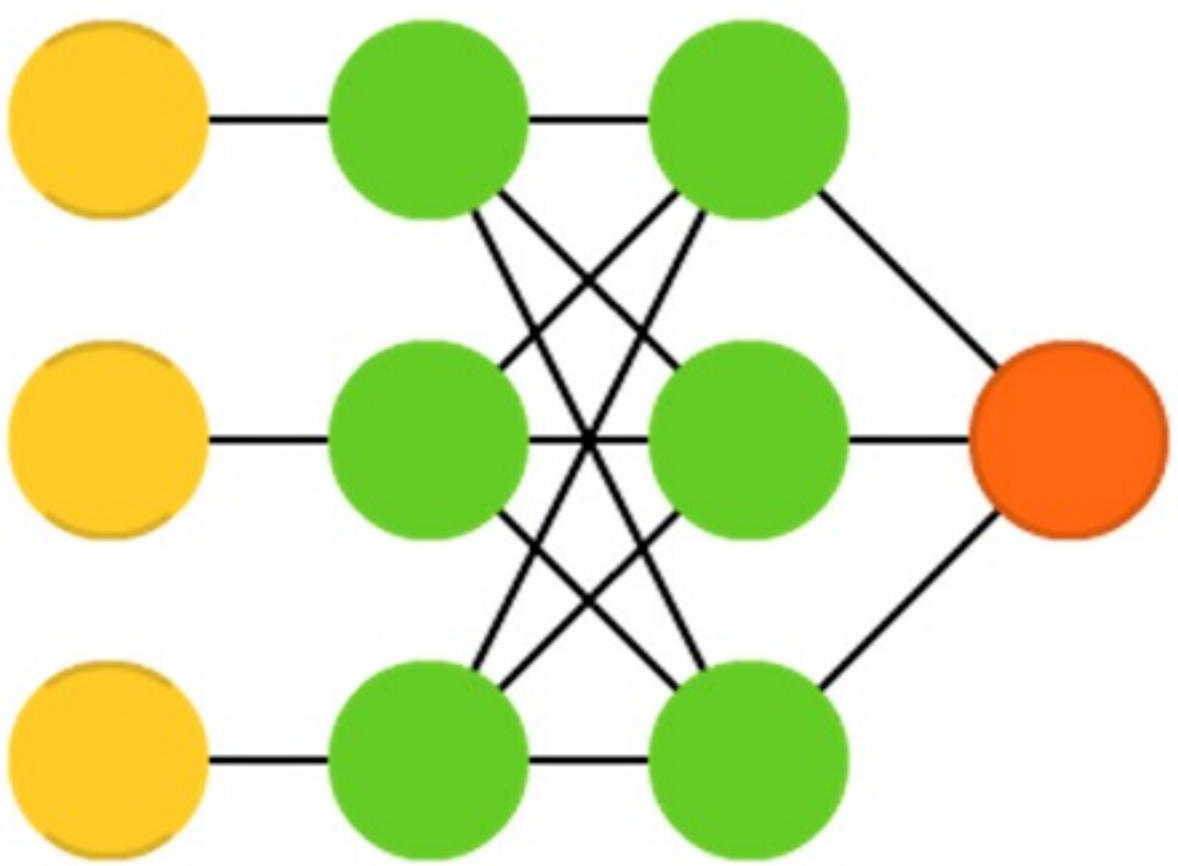
Generative Adversarial Networks (GAN)



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)

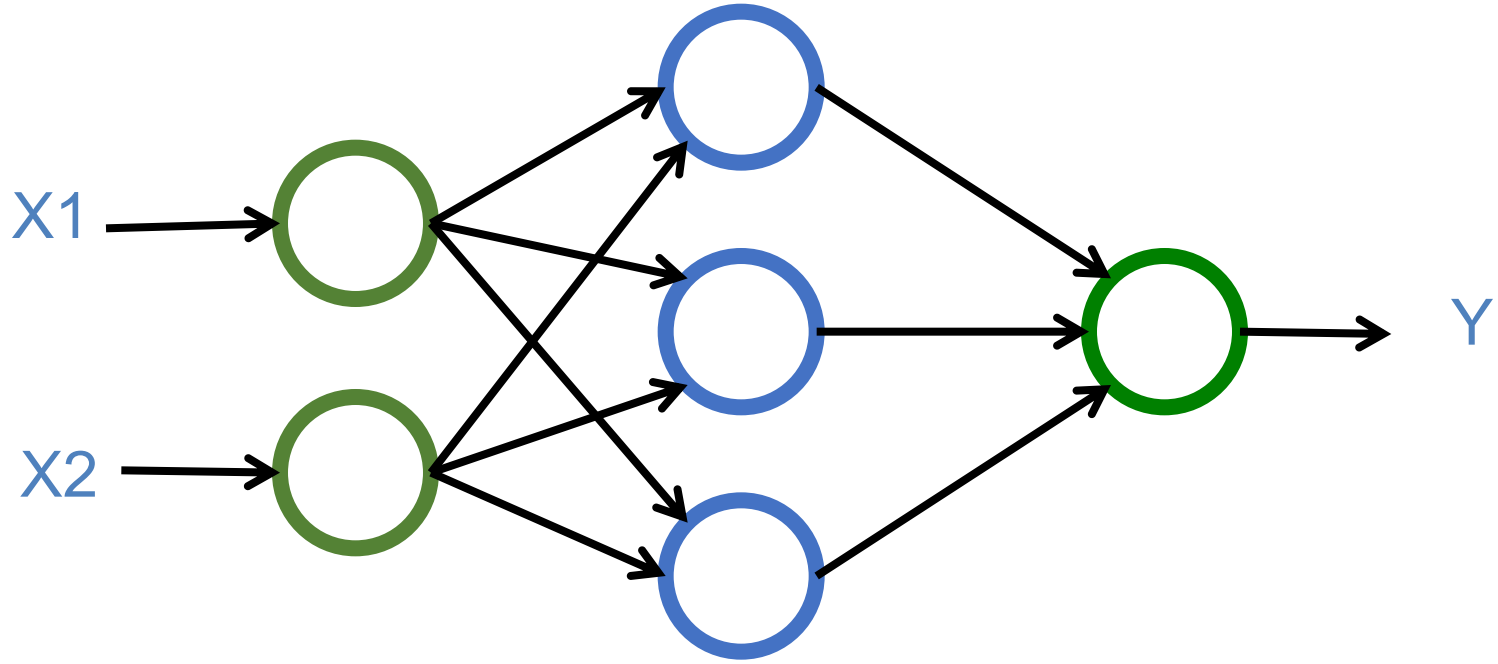


Cortes, Corinna, and Vladimir Vapnik. “Support-vector networks.” Machine learning 20.3 (1995): 273-297.

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

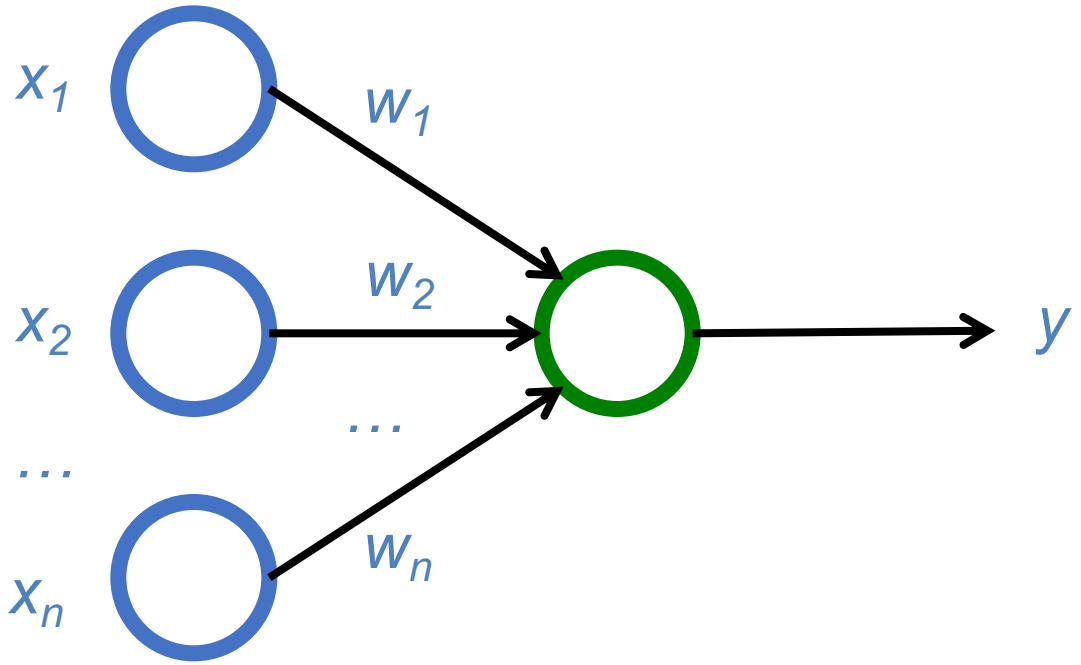
Neural Networks

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



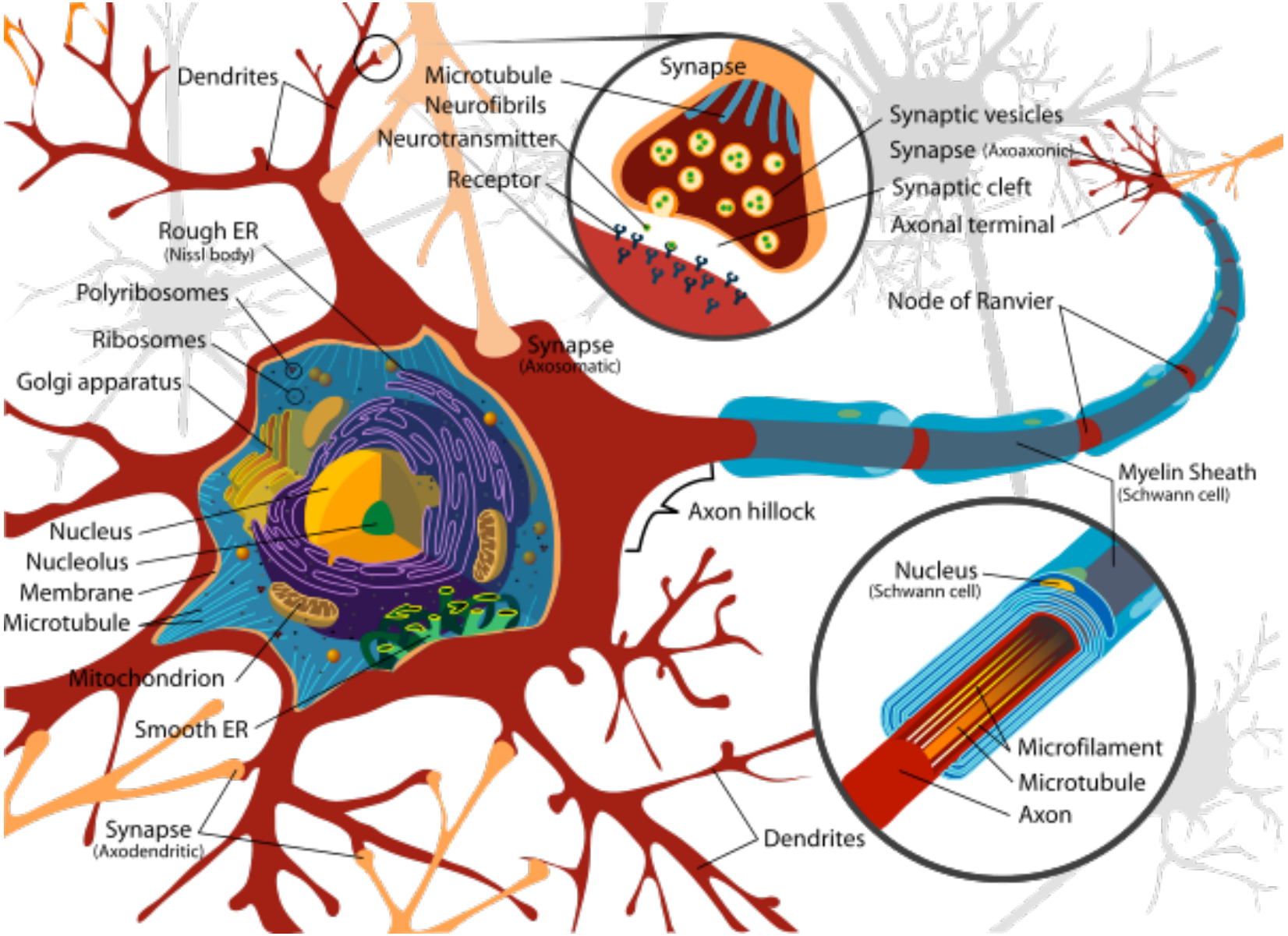
Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

The Neuron



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

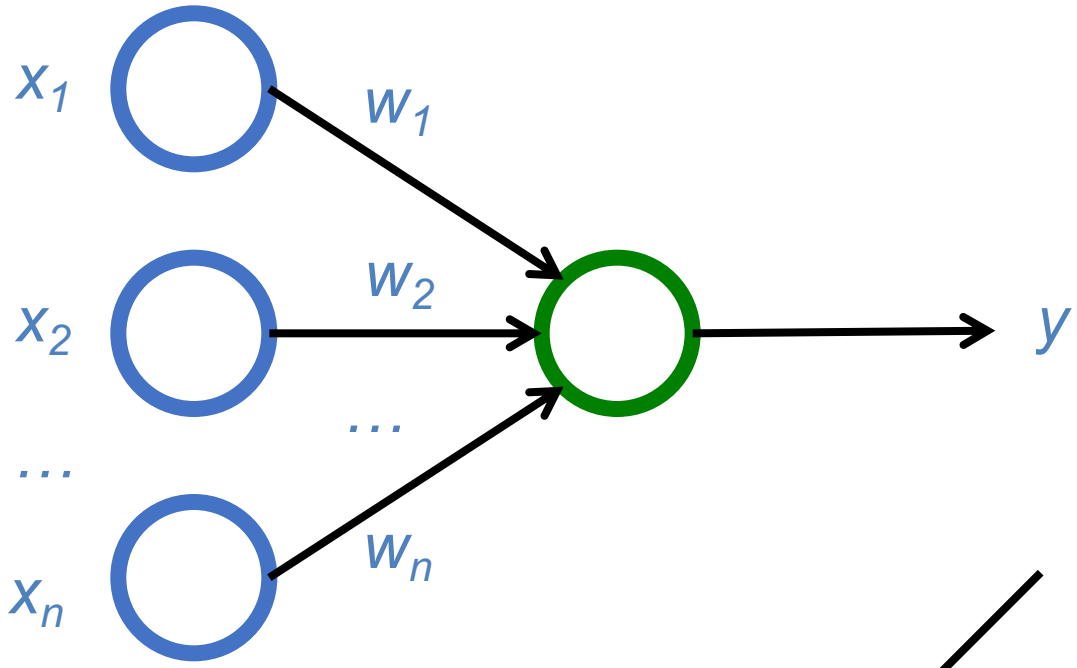
Neuron and Synapse

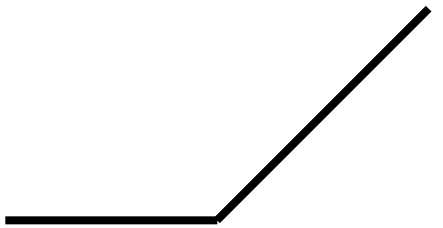


Source: https://en.wikipedia.org/wiki/Neuron

The Neuron

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

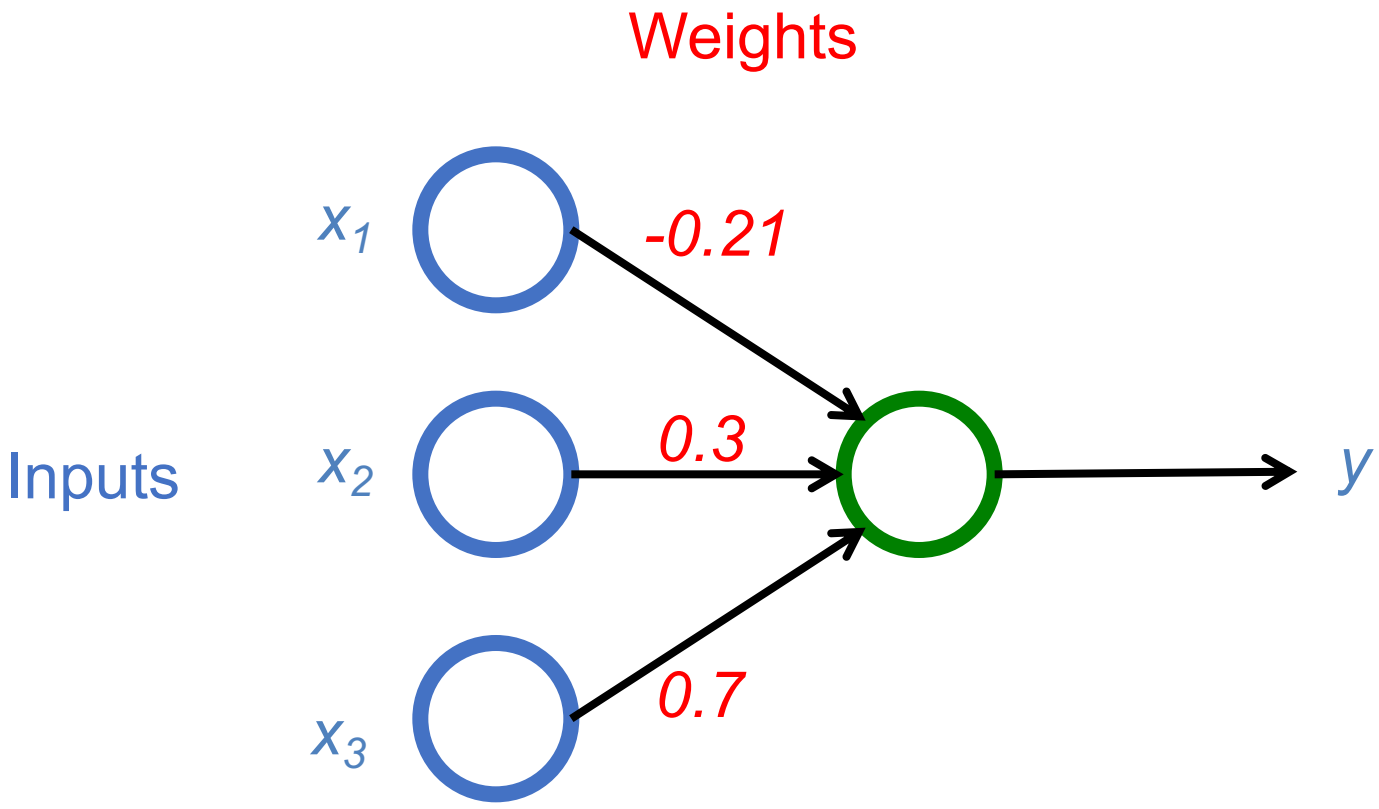




$$F(x) = \max(0, x)$$

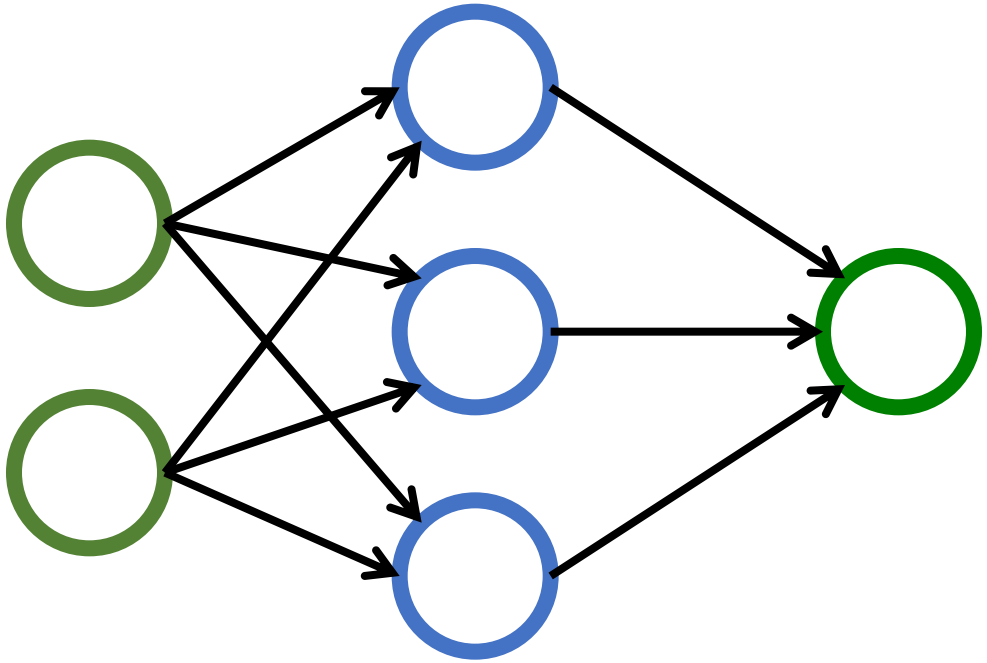
Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

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



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

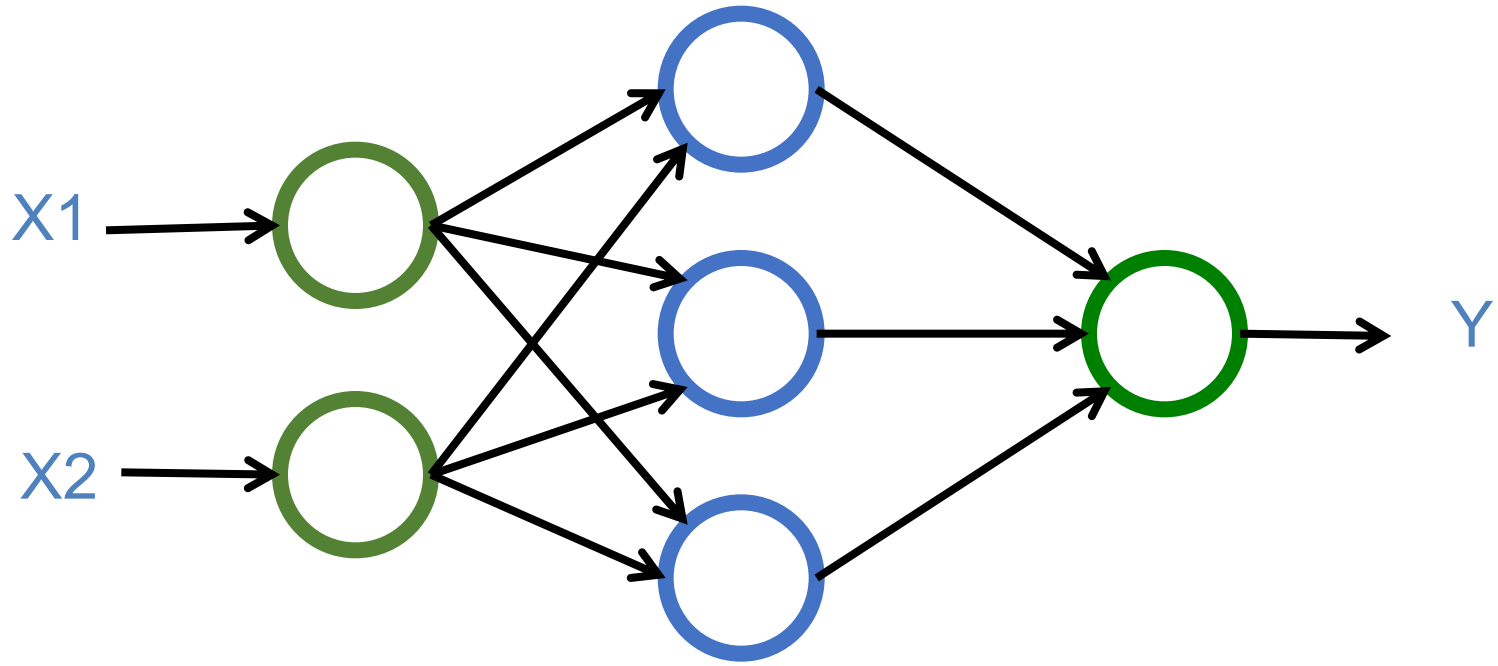
Neural Networks



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

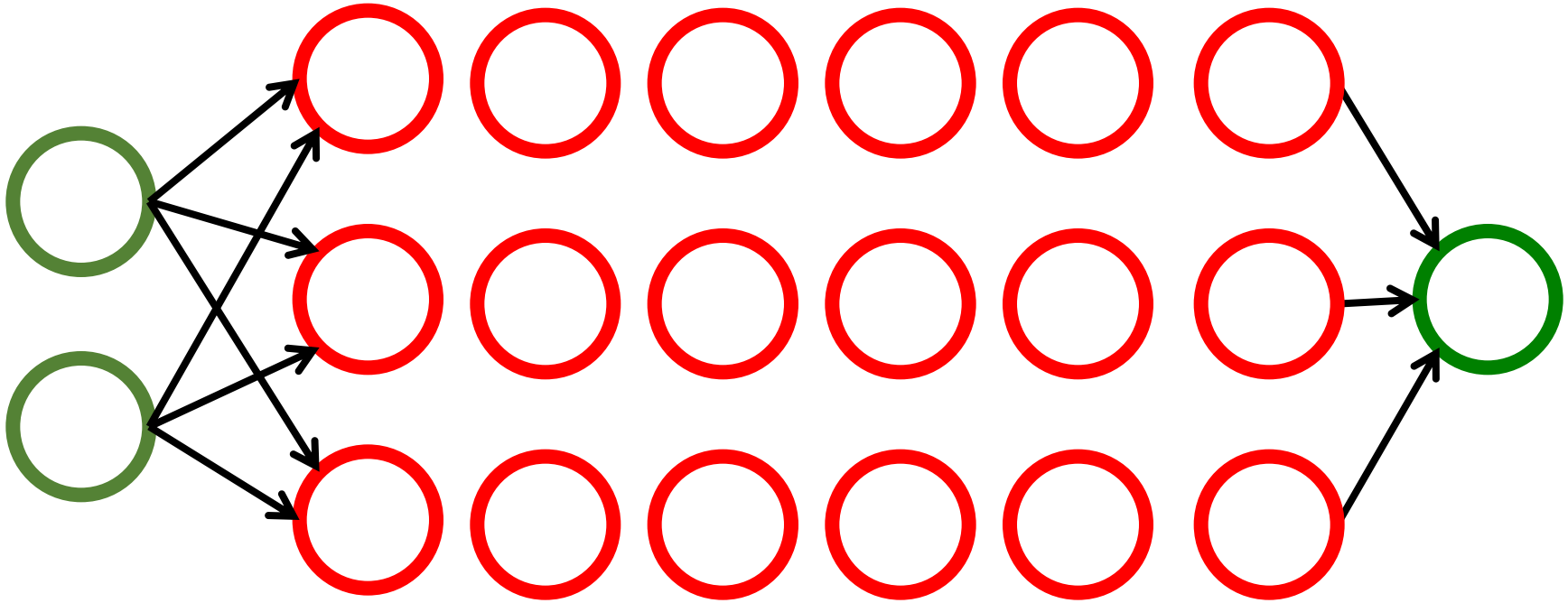
Neural Networks

Input Layer
(X)

Hidden Layers
(H)

Output Layer
(Y)

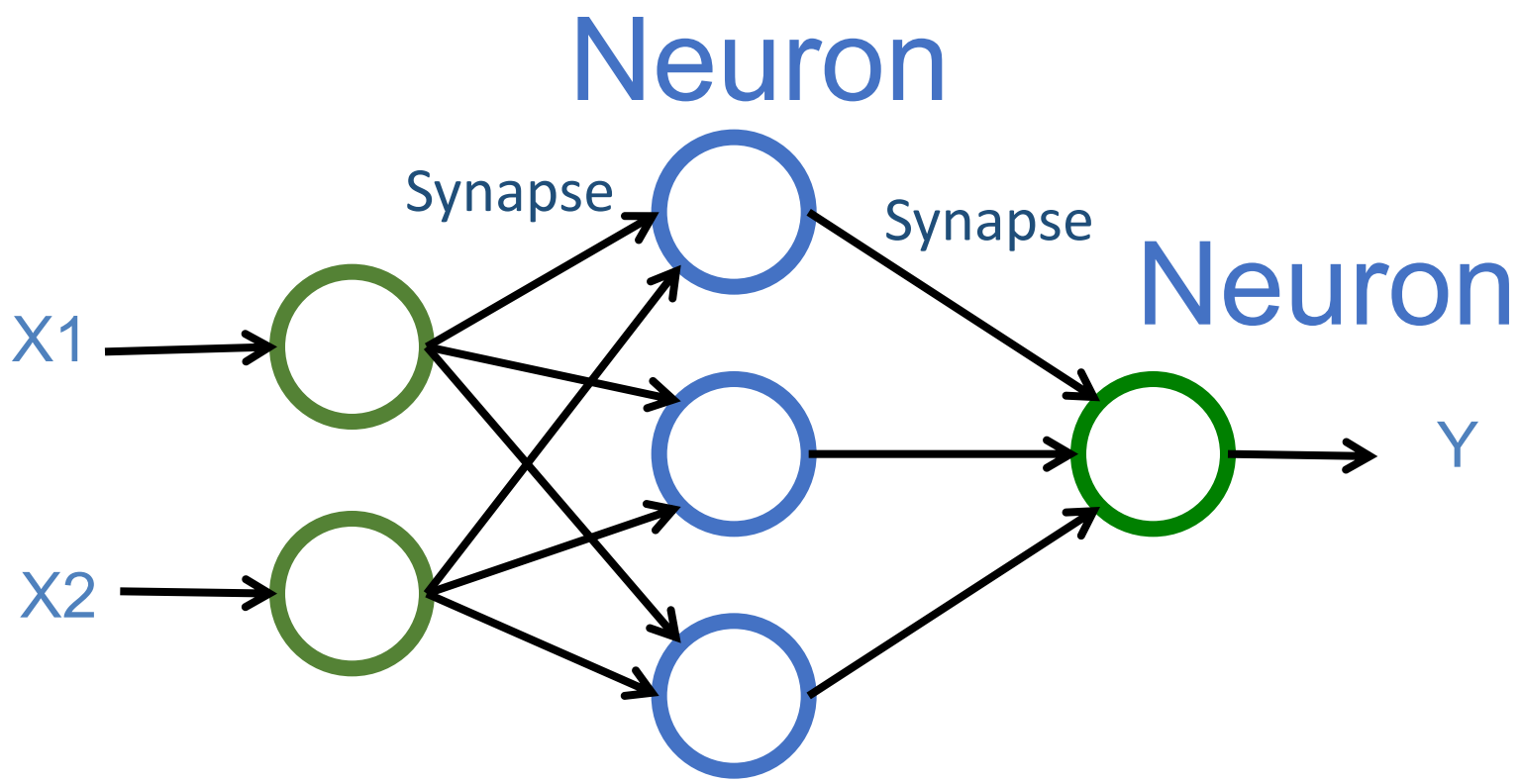
Deep Neural Networks
Deep Learning



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

Neural Networks

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



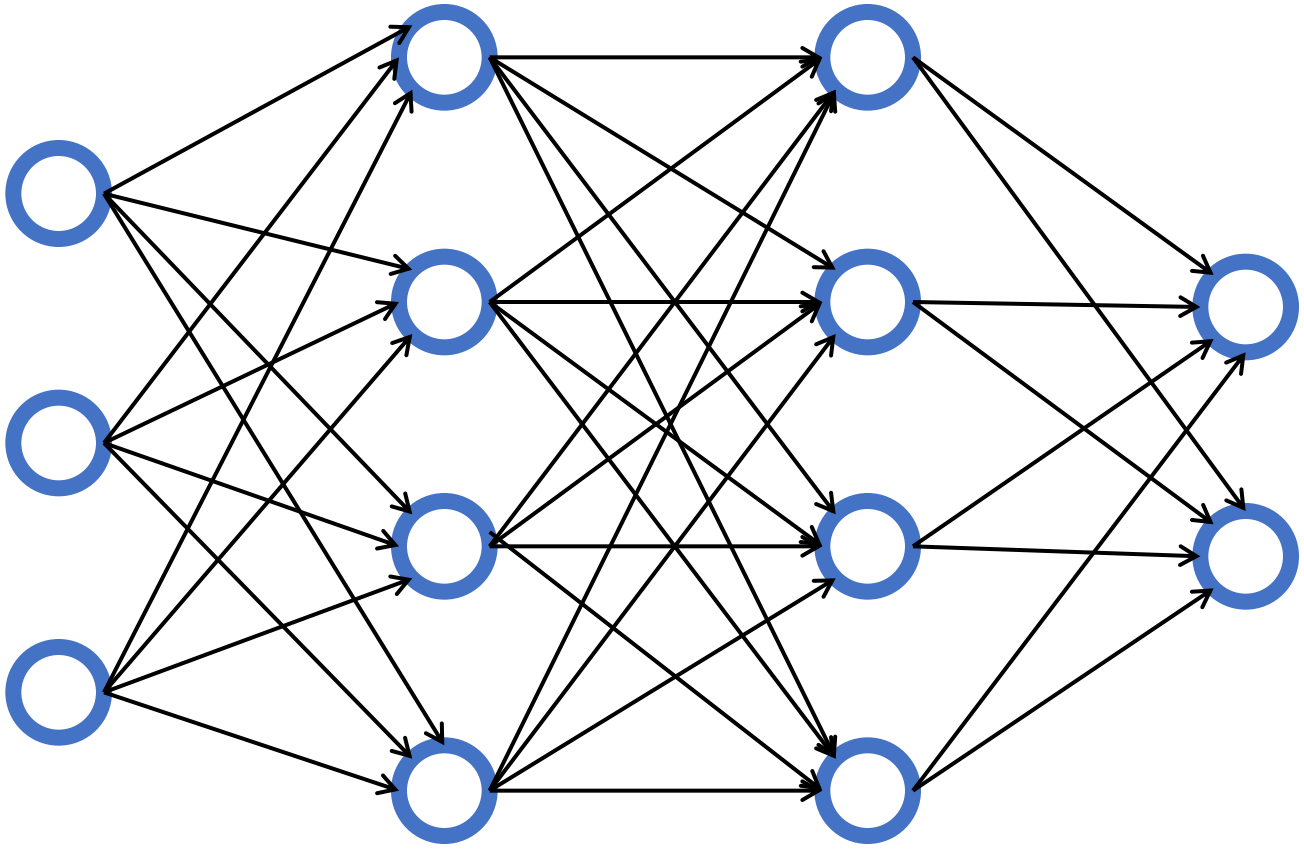
Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Neural Networks

Input Layer
(X)

Hidden Layer
(H)

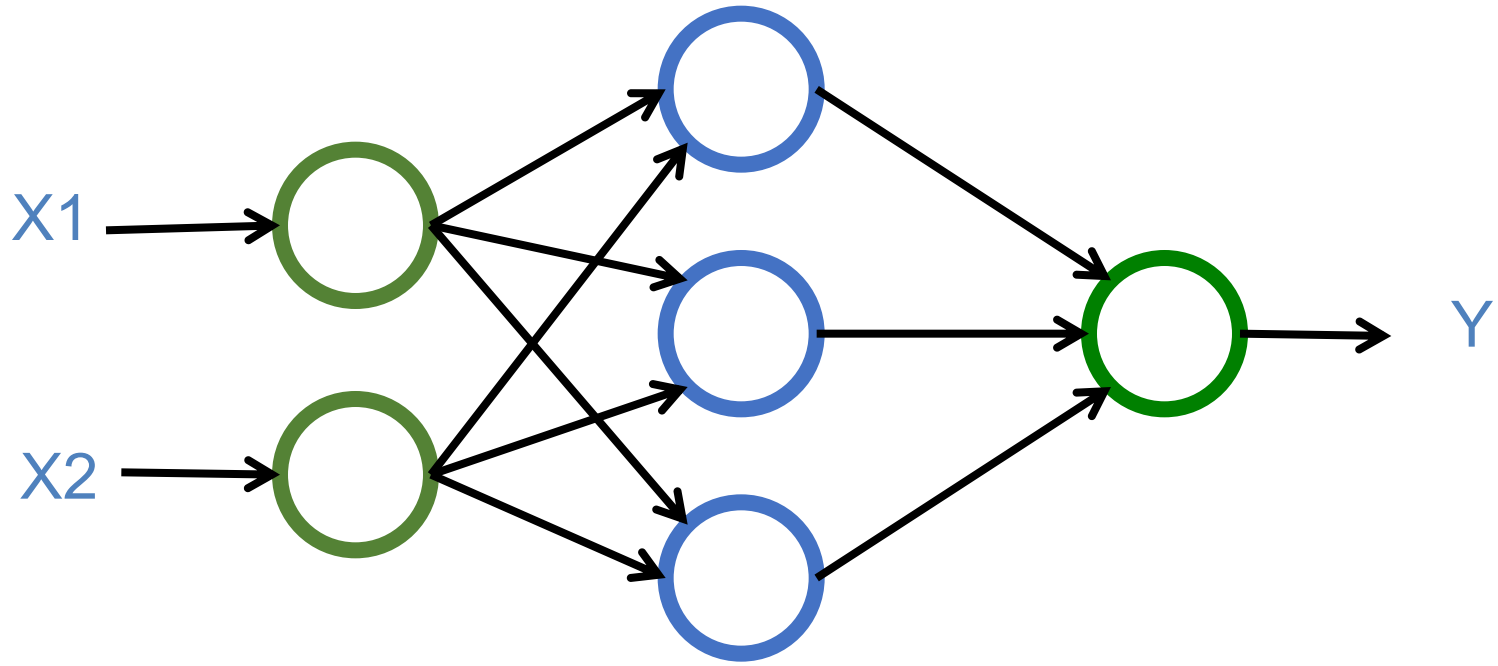
Output Layer
(Y)



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

Neural Networks

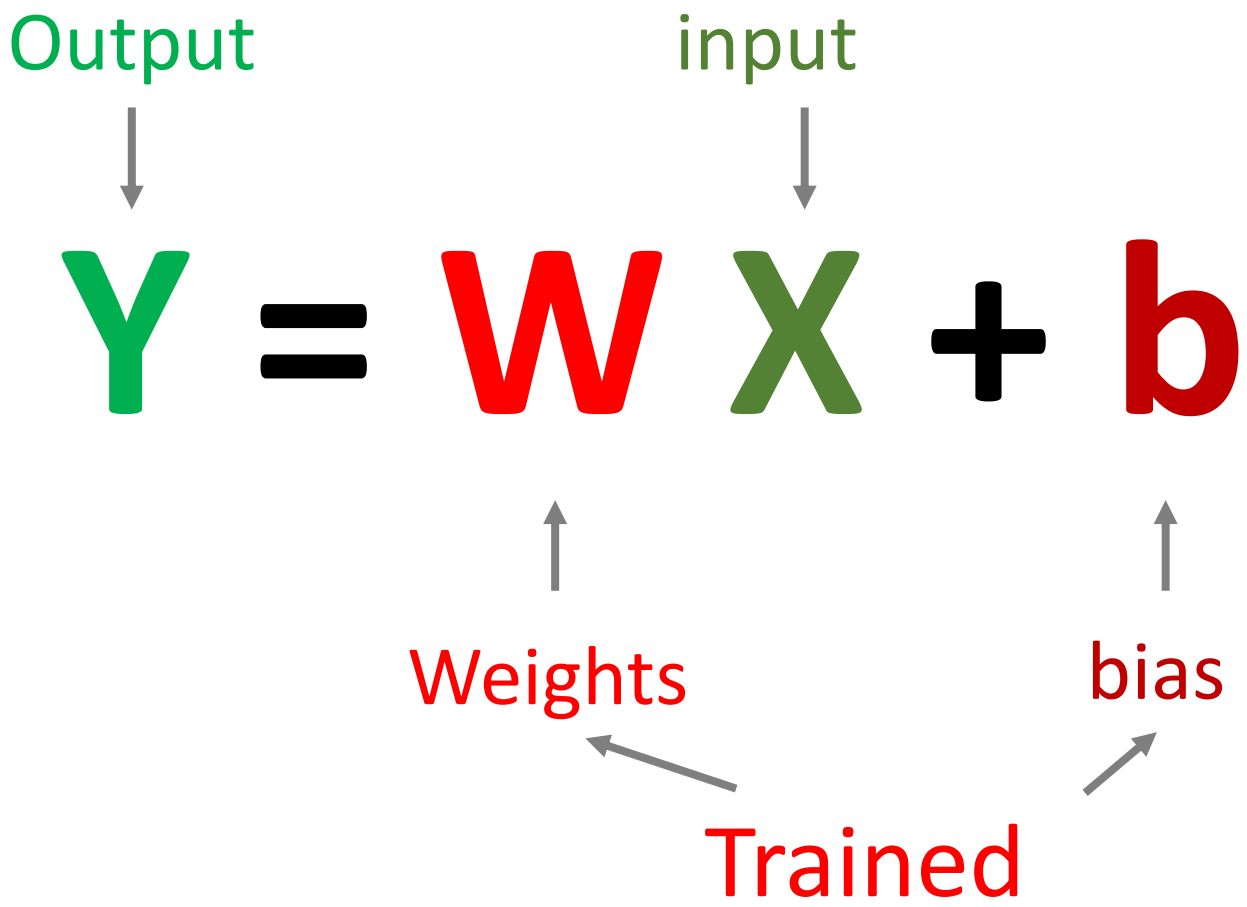
Input Layer (X) Hidden Layer (H) **Output Layer (Y)**



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

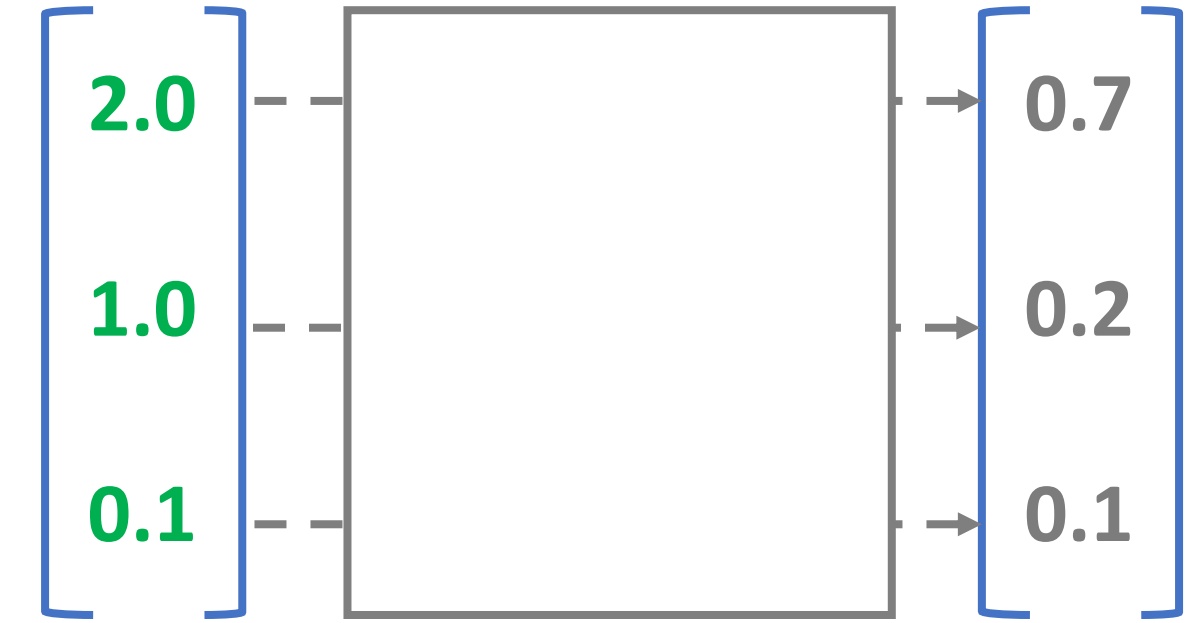
$$Y = W X + b$$

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



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

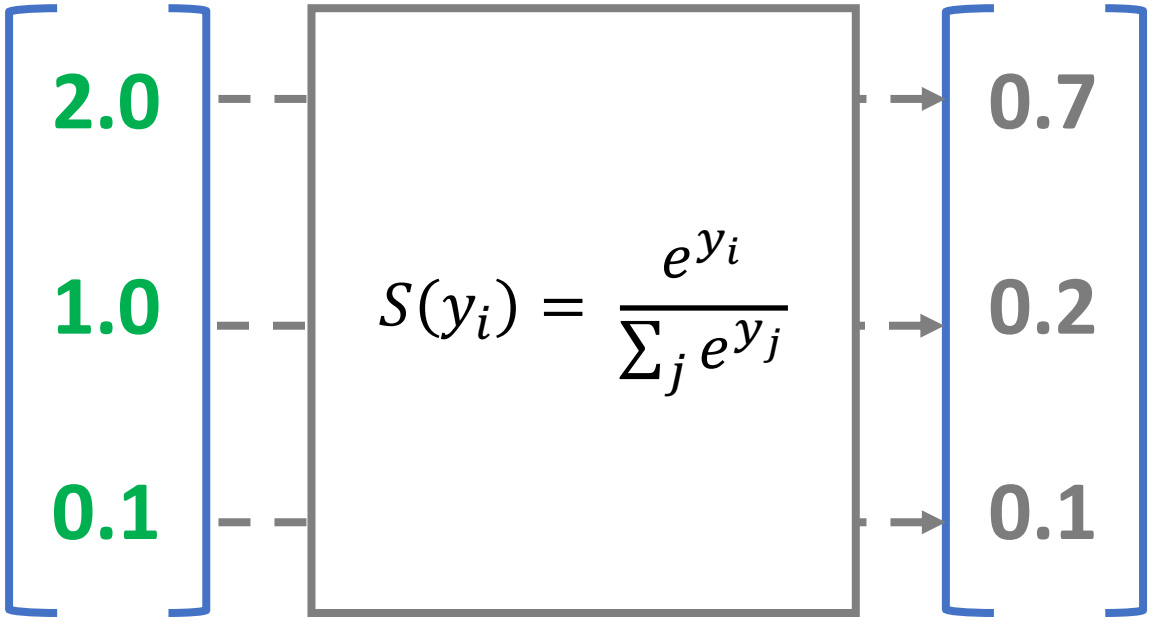
$$W X + b = Y$$



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

SoftMAX

$$W\ X + b = Y$$



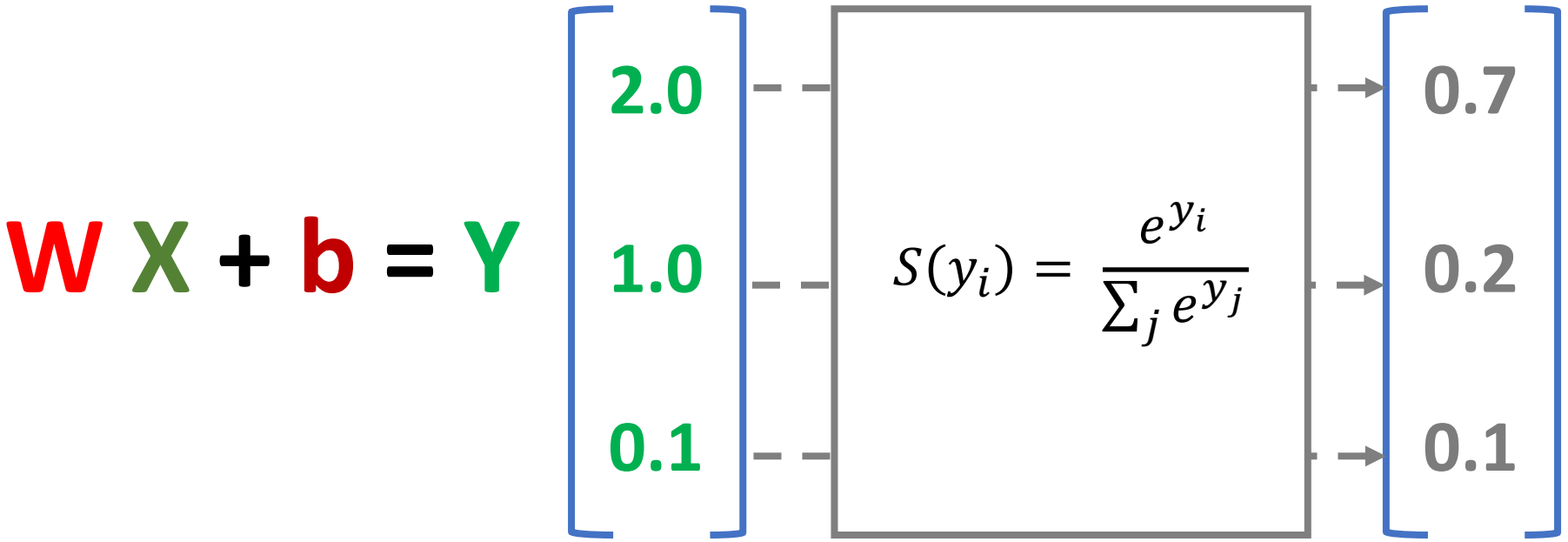
Logits Scores → Probabilities

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

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

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

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



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

=

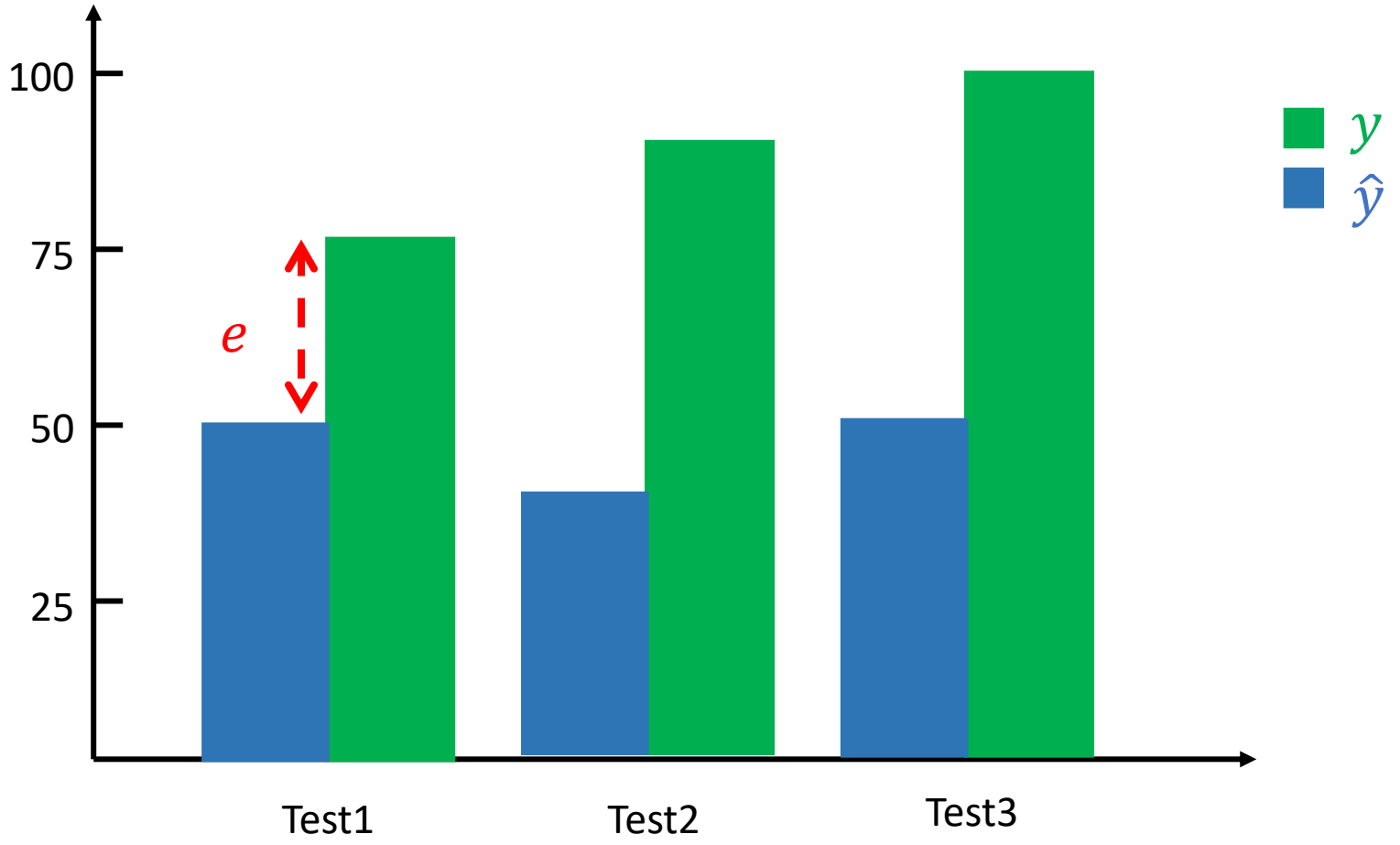
Minimize the Cost Function

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

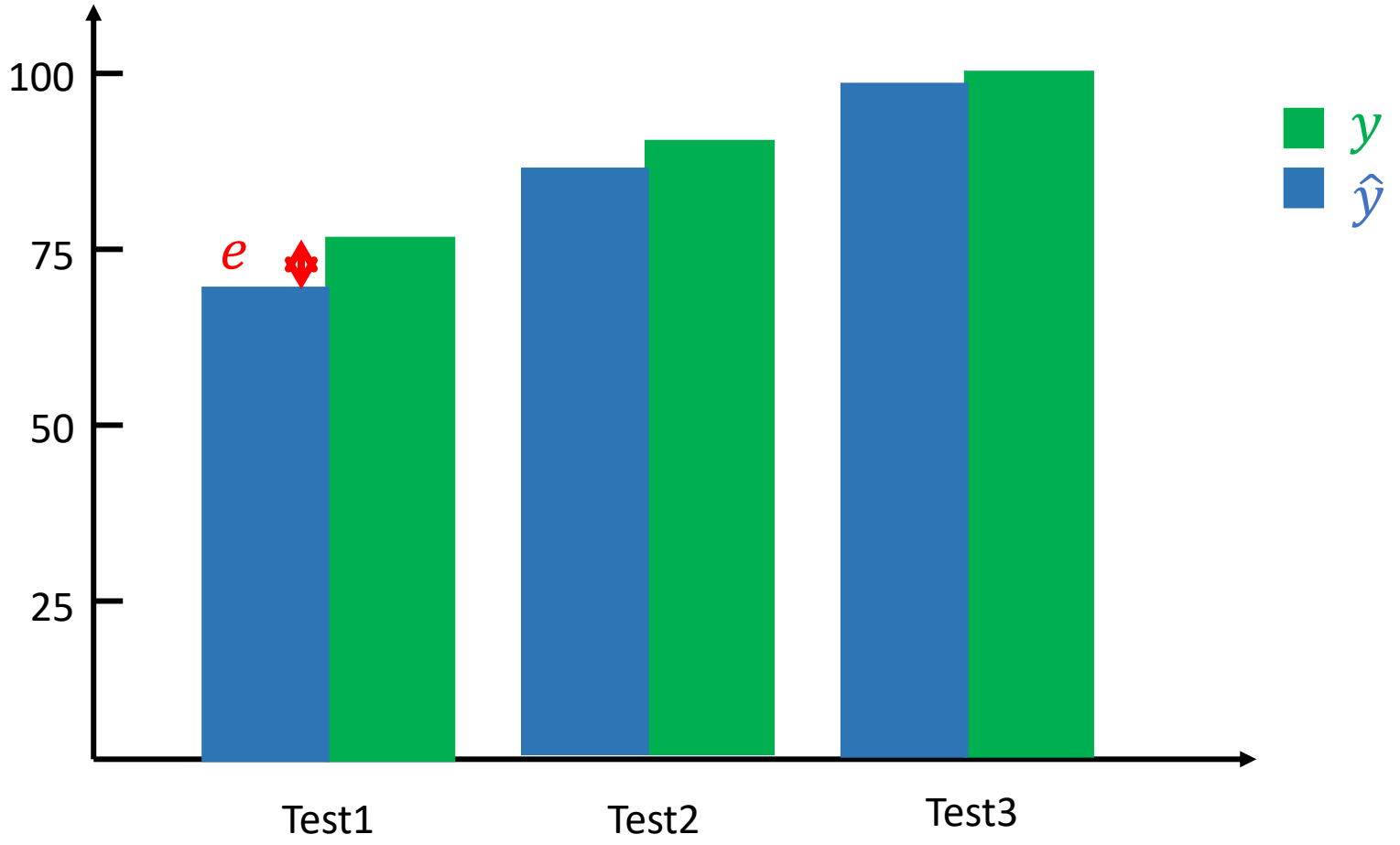
Error = Predict Y - Actual Y

Error : Cost : Loss

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

Error = Predict Y - Actual Y

Error : Cost : Loss

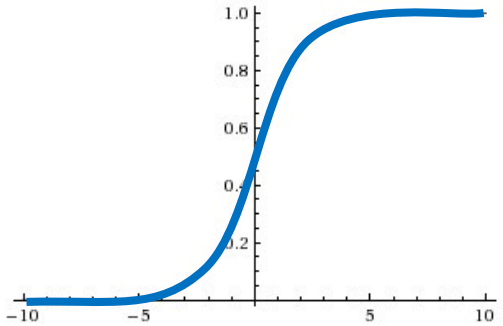


Source: https://www.youtube.com/watch?v=bxe2T-V8XRs&index=1&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU

Activation Functions

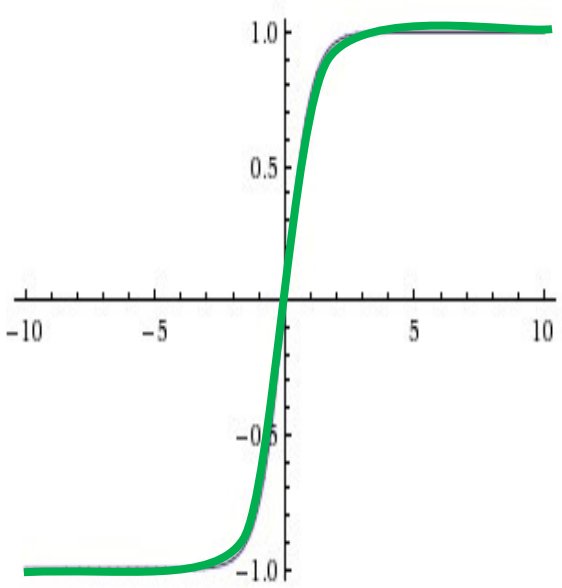
Activation Functions

Sigmoid



[0, 1]

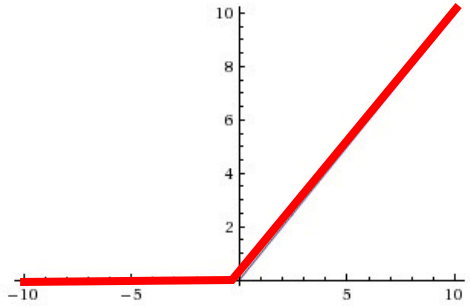
TanH



[-1, 1]

ReLU

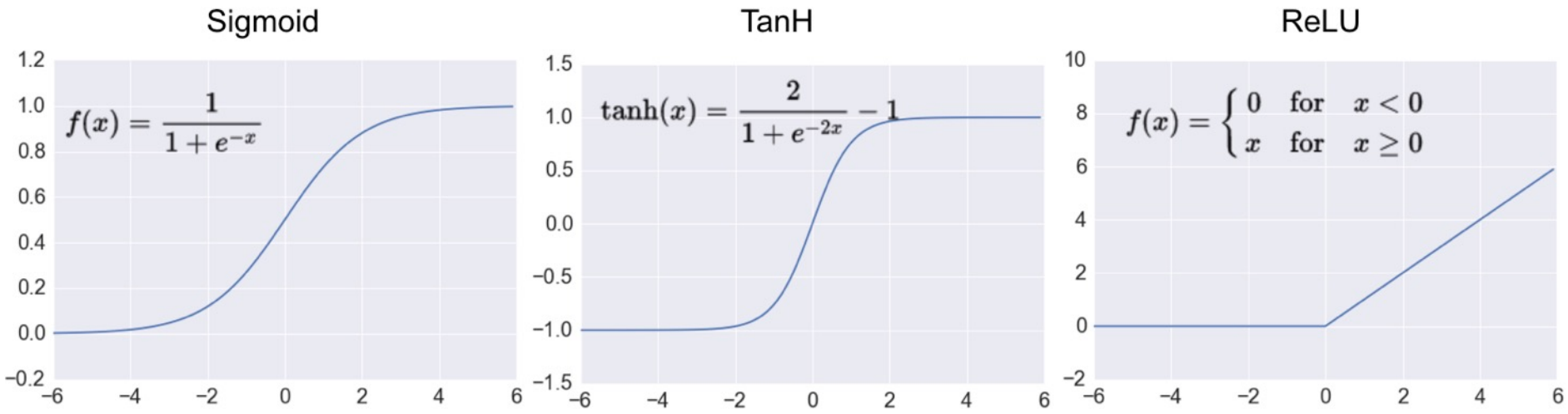
(Rectified Linear Unit)



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

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

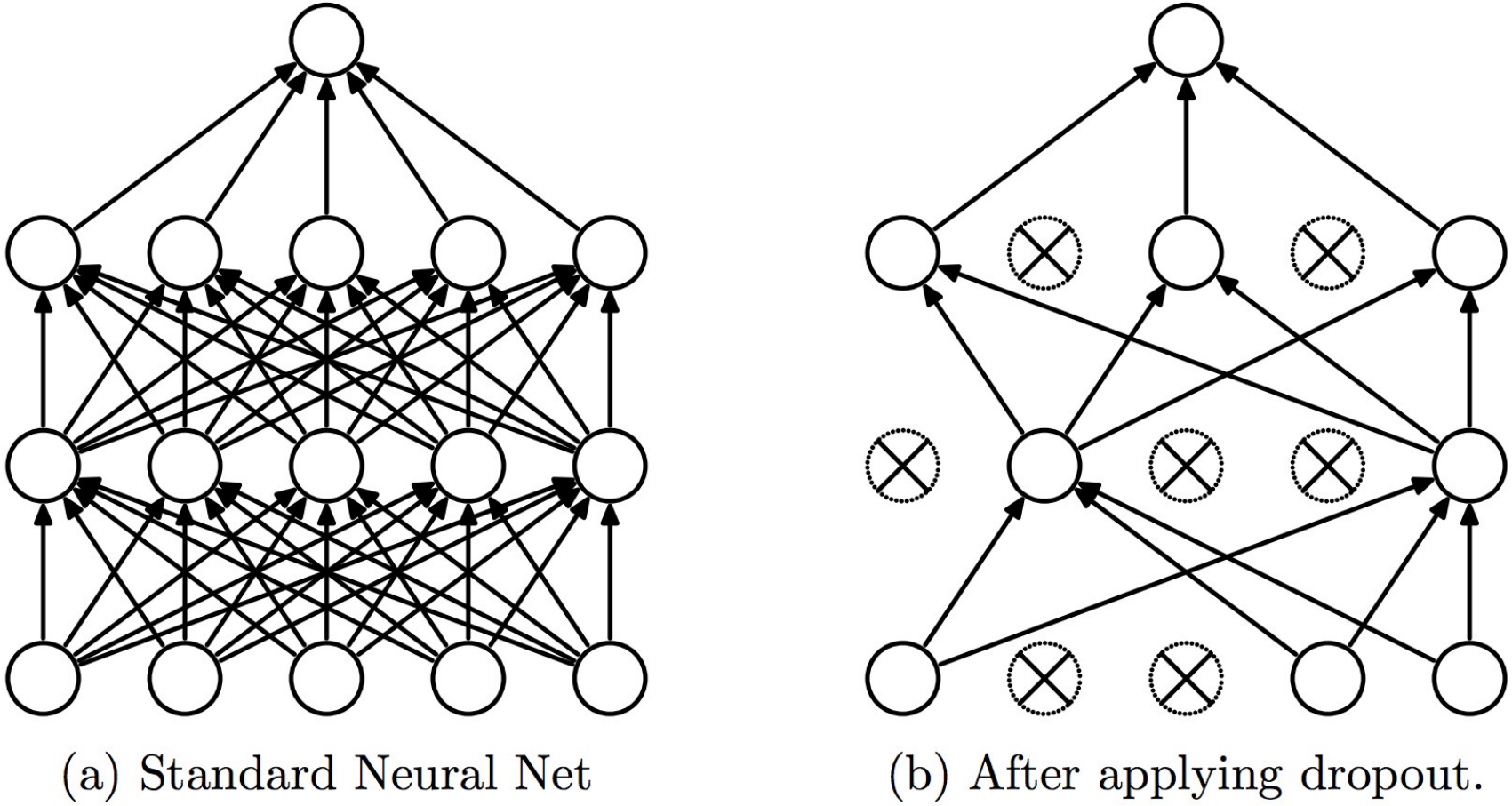
Multiple Classification: 10 Class

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.

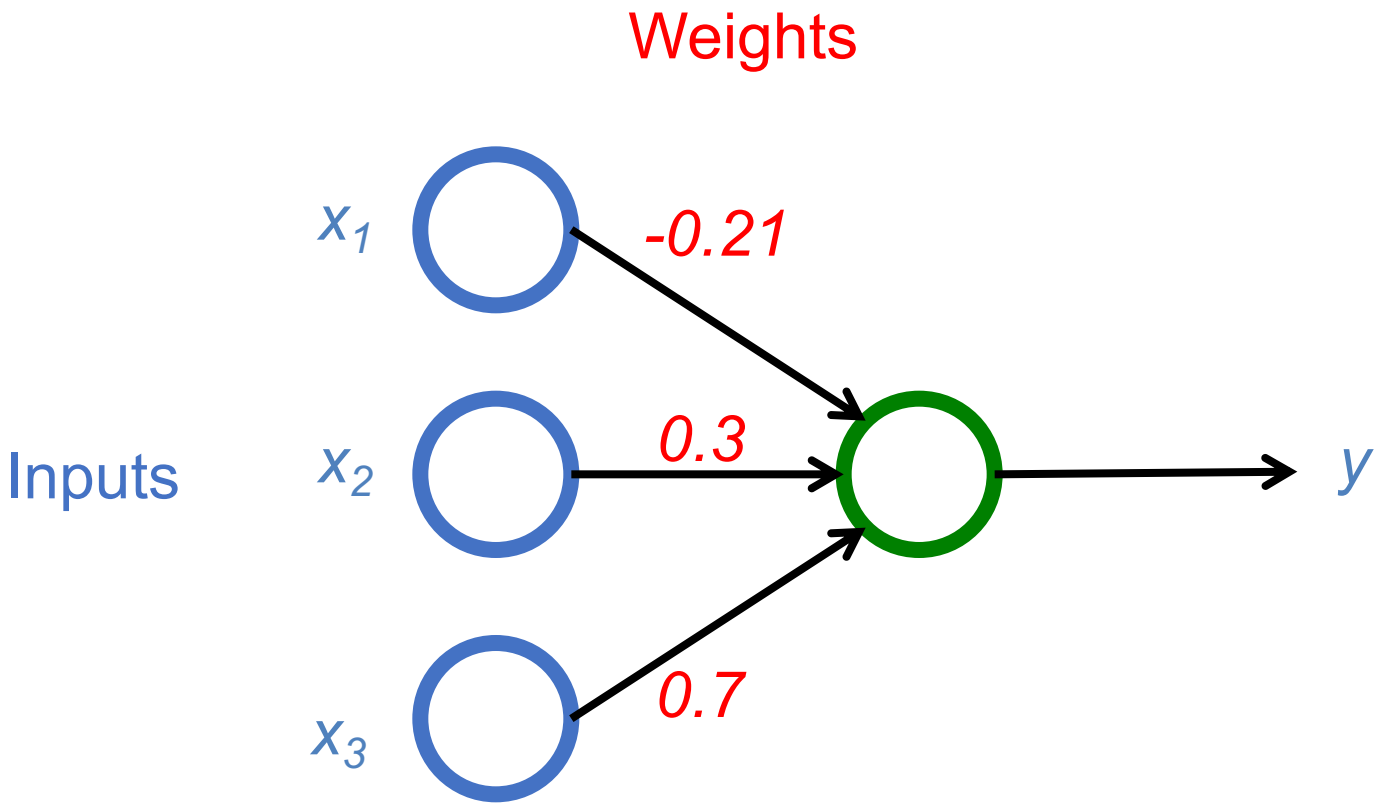
Learning Algorithm

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

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

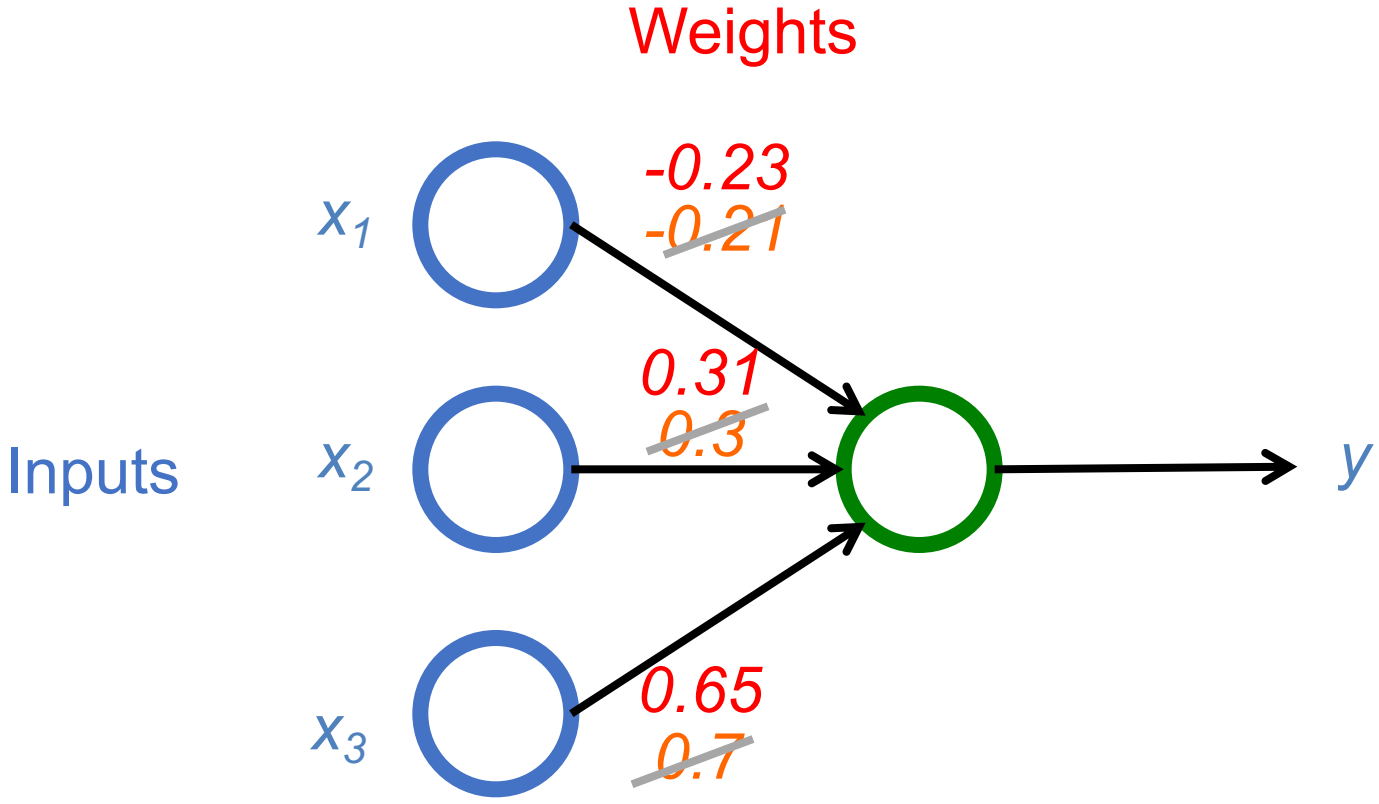


Source: Jeff Dean (2016), Large-Scale Deep Learning For Building Intelligent Computer Systems, WSDM 2016

Next time:

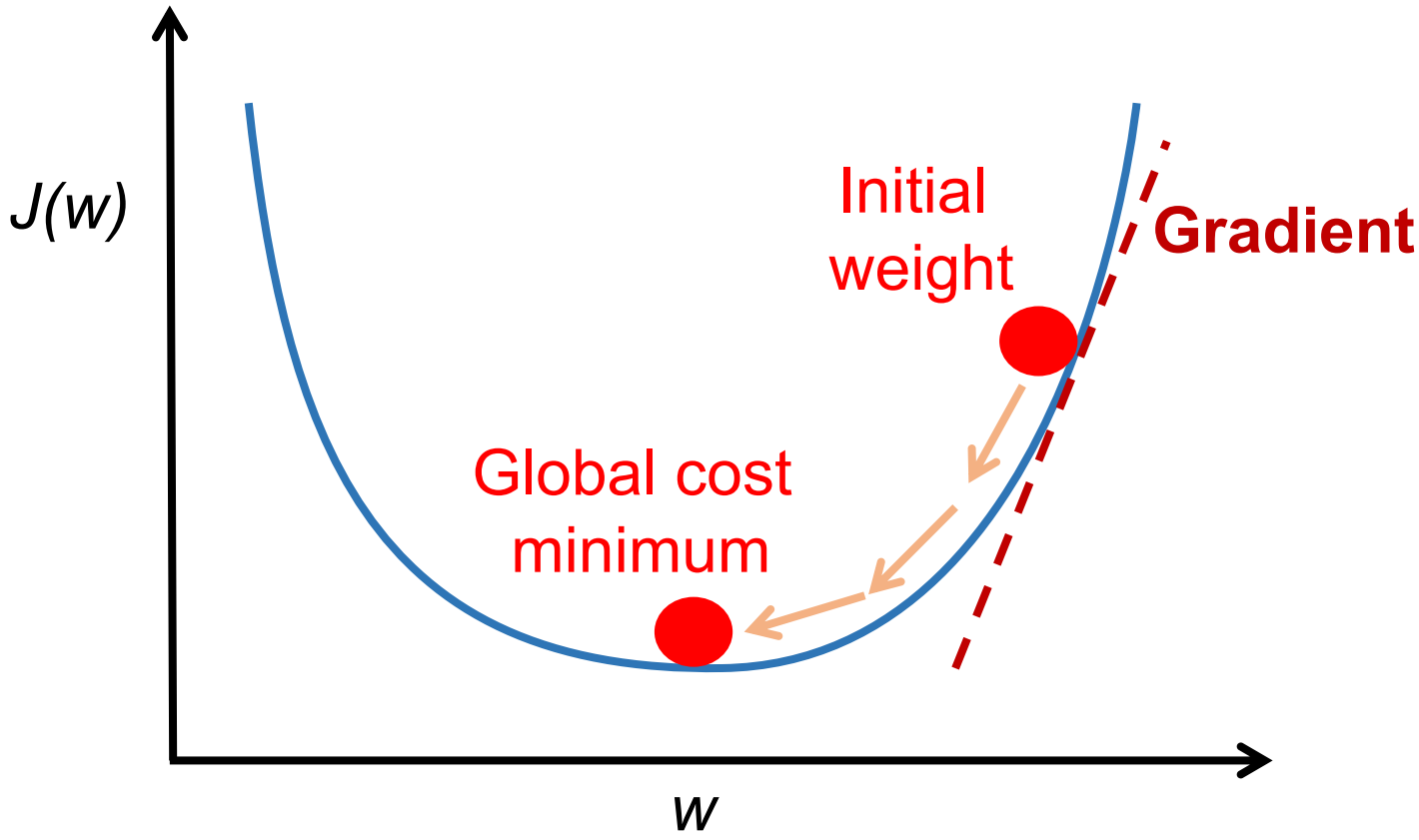
$$y = max(0, -0.23 * x_1 + 0.31 * x_2 + 0.65 * x_3)$$

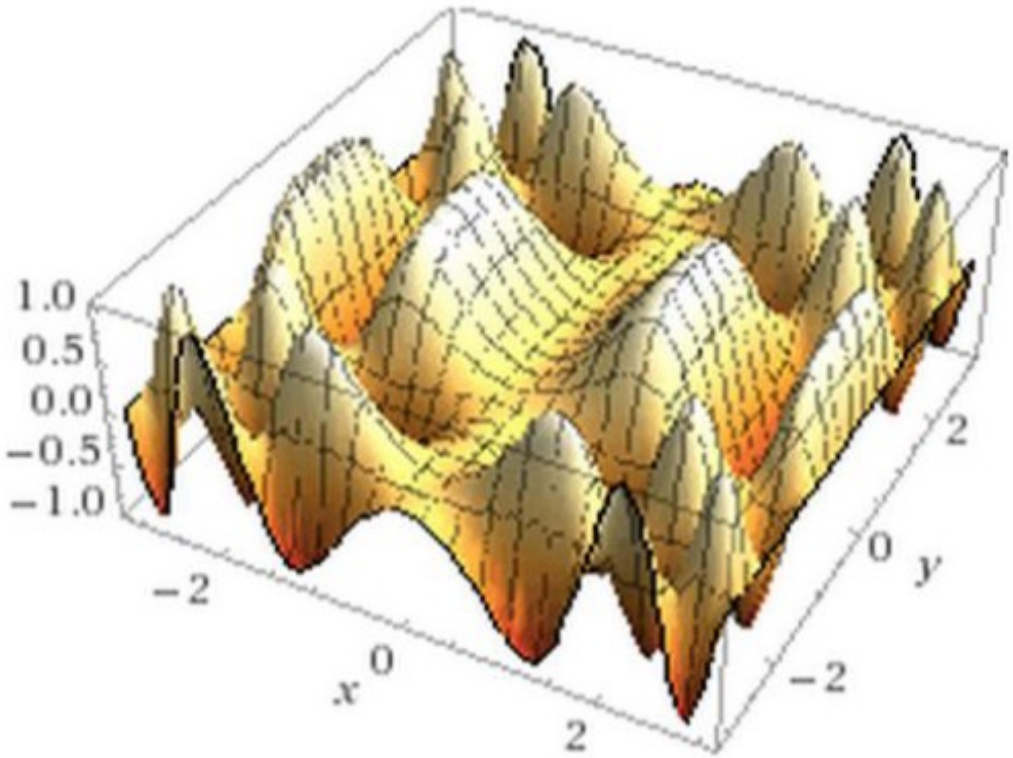
~~$$y = max(0, -0.21 * x_1 + 0.3 * x_2 + 0.7 * x_3)$$~~



Optimizer:

Stochastic Gradient Descent (SGD)

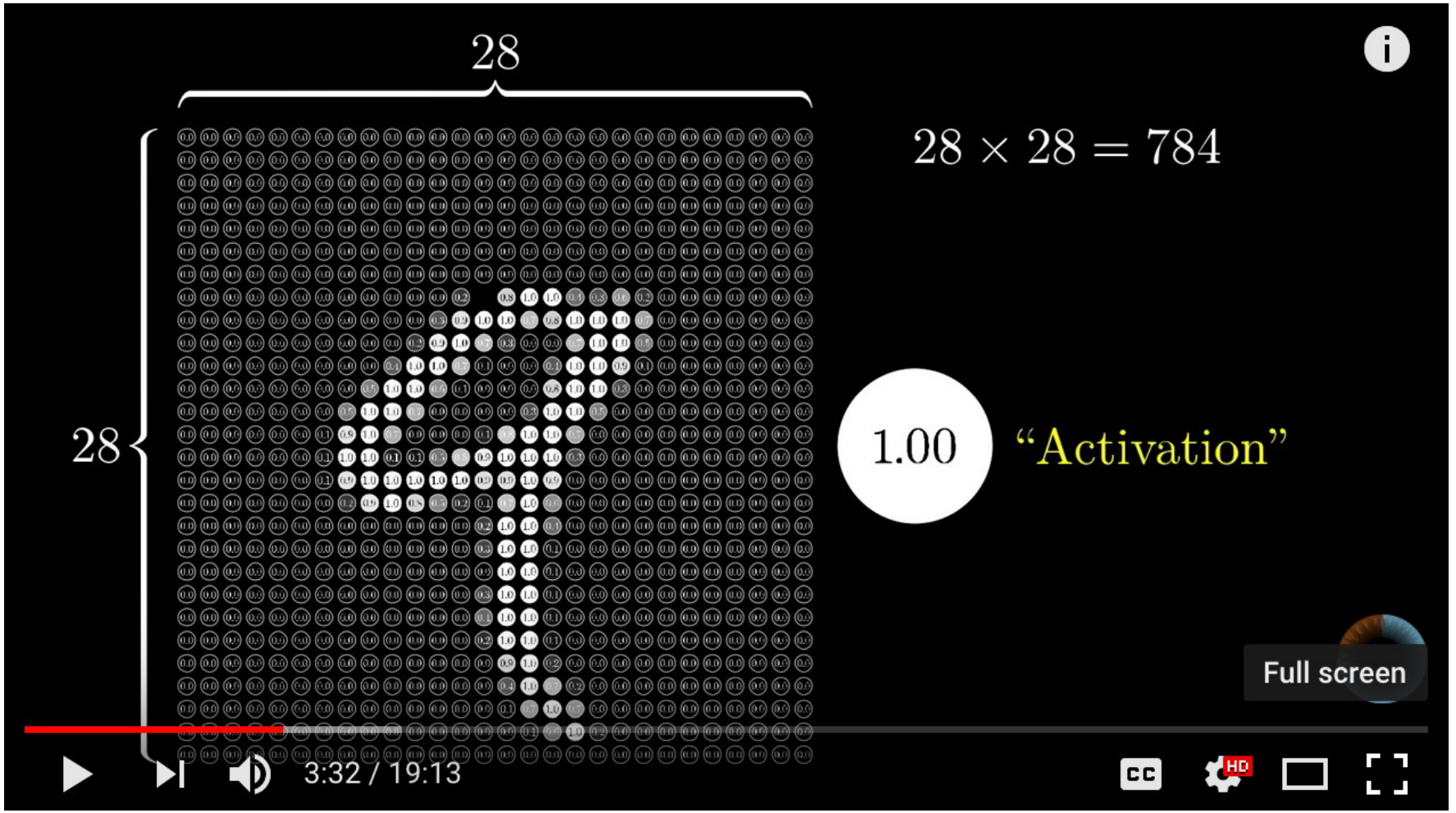




This shows a function of 2 variables: real neural nets are functions of hundreds of millions of variables!

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

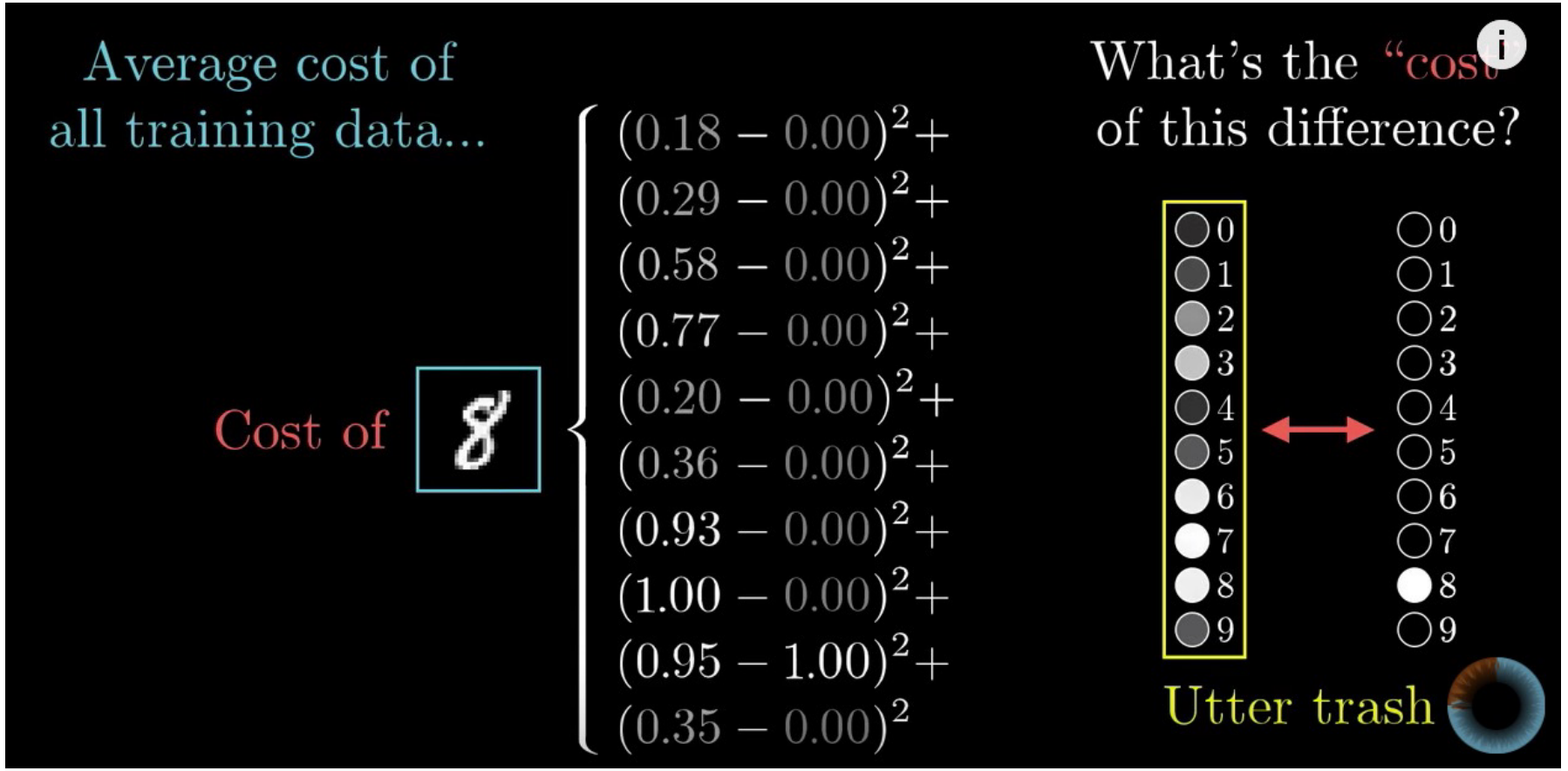
Neural Network and Deep Learning



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

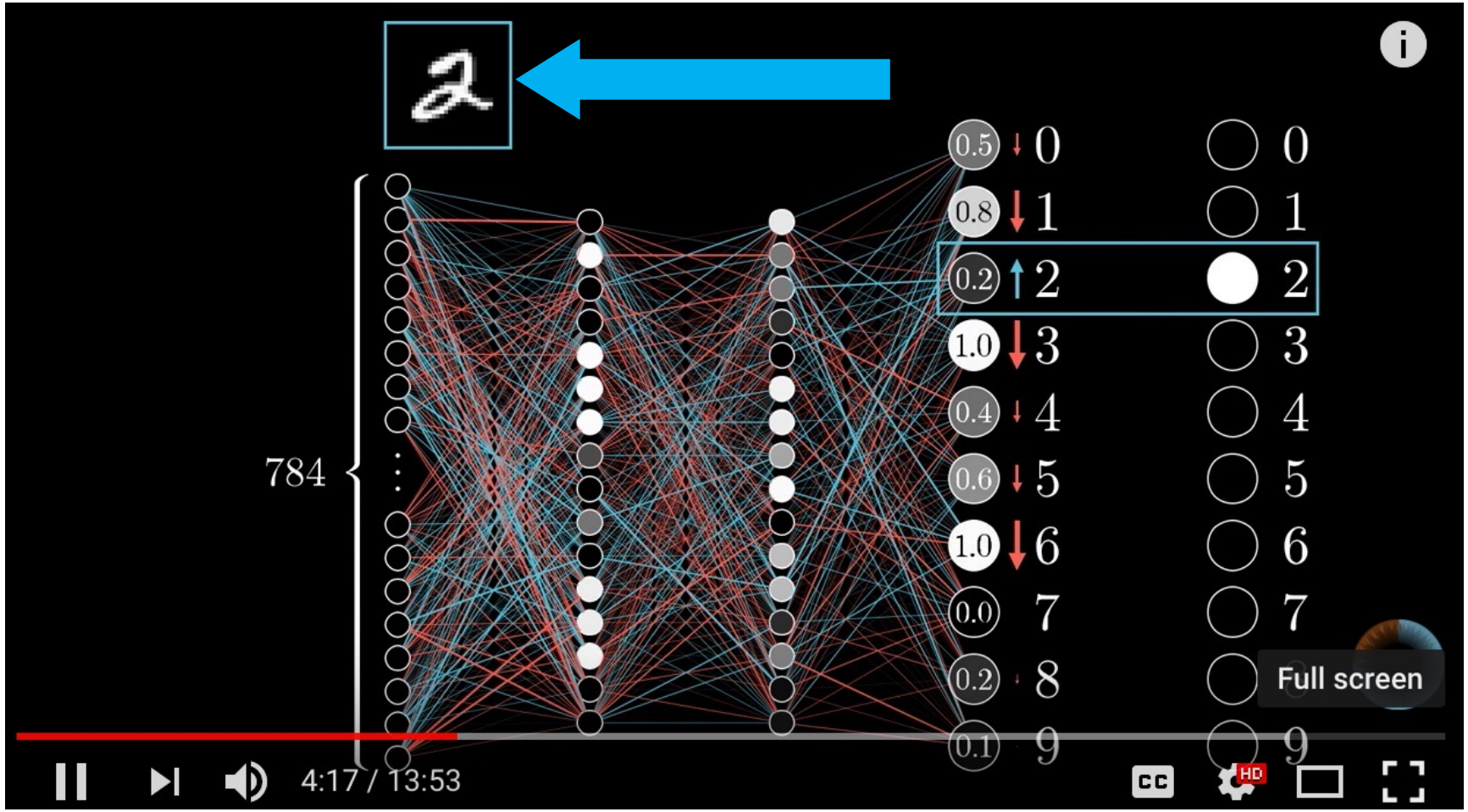
Gradient Descent

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

Learning Algorithm

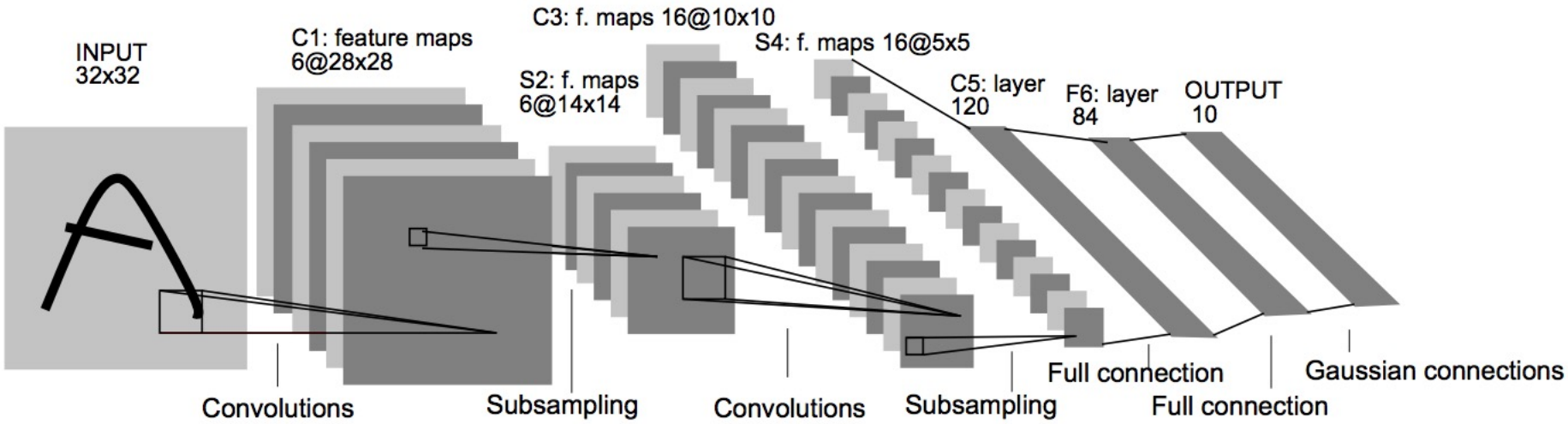
While not done:

- **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

Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNN)



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

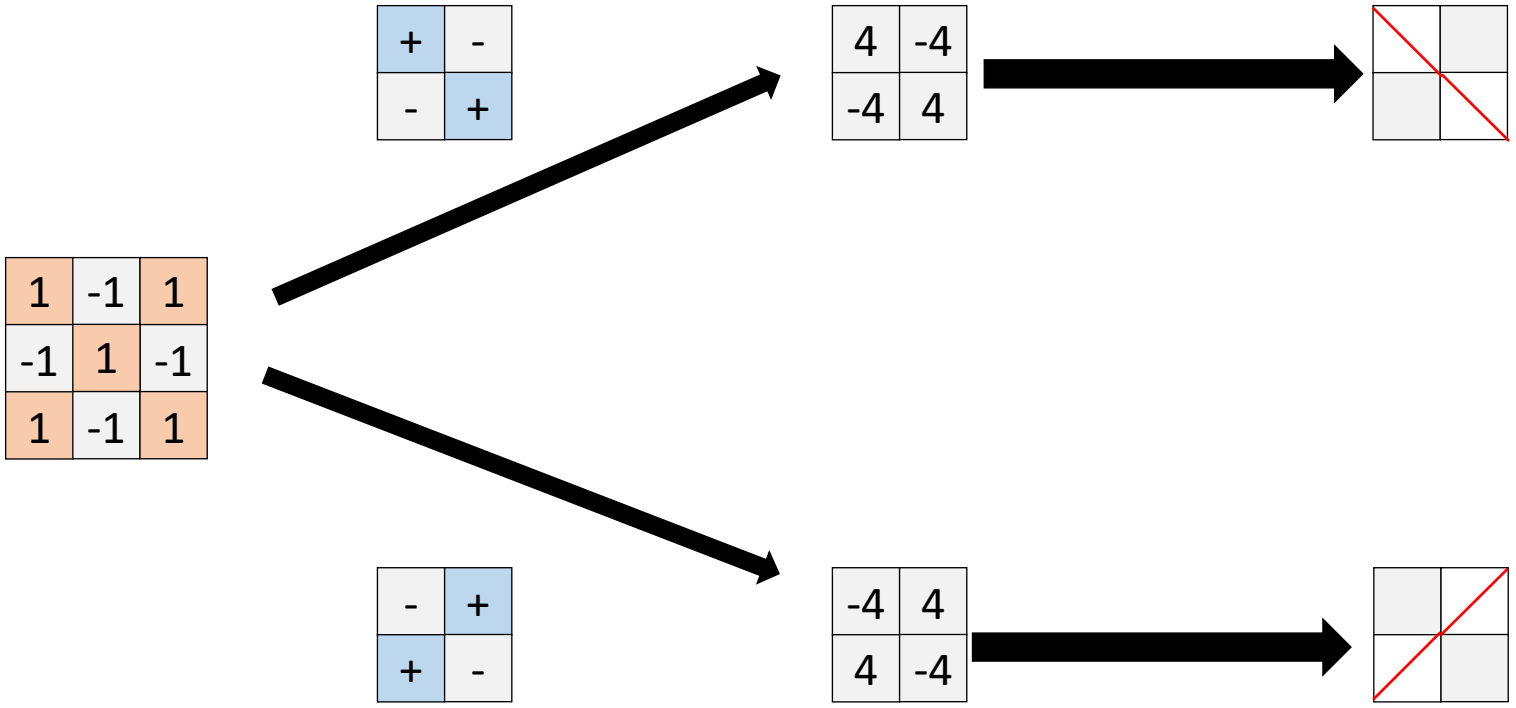
Source: http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

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)**

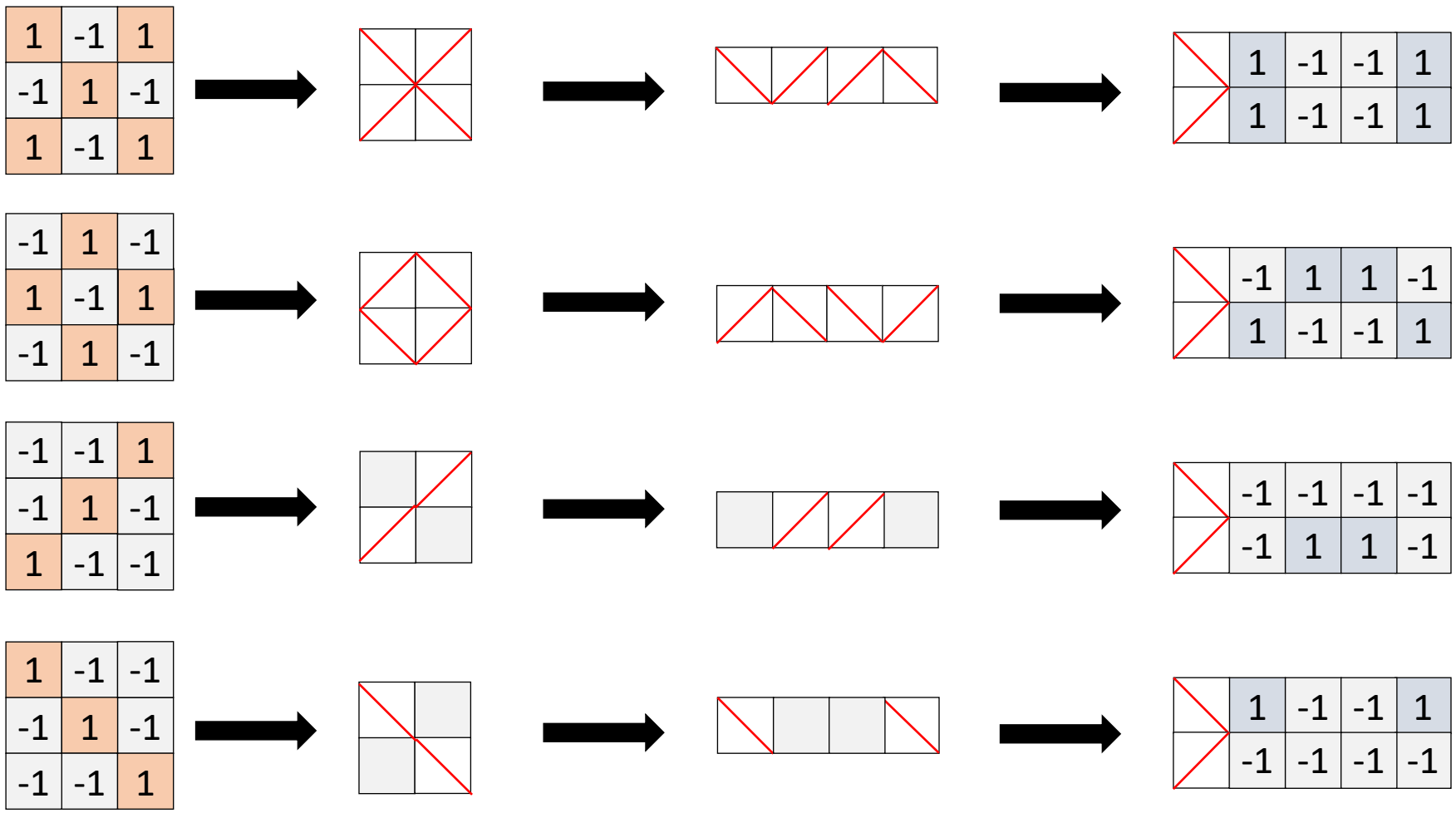
A friendly introduction to Convolutional Neural Networks and Image Recognition



Convolution Layer Pooling Layer

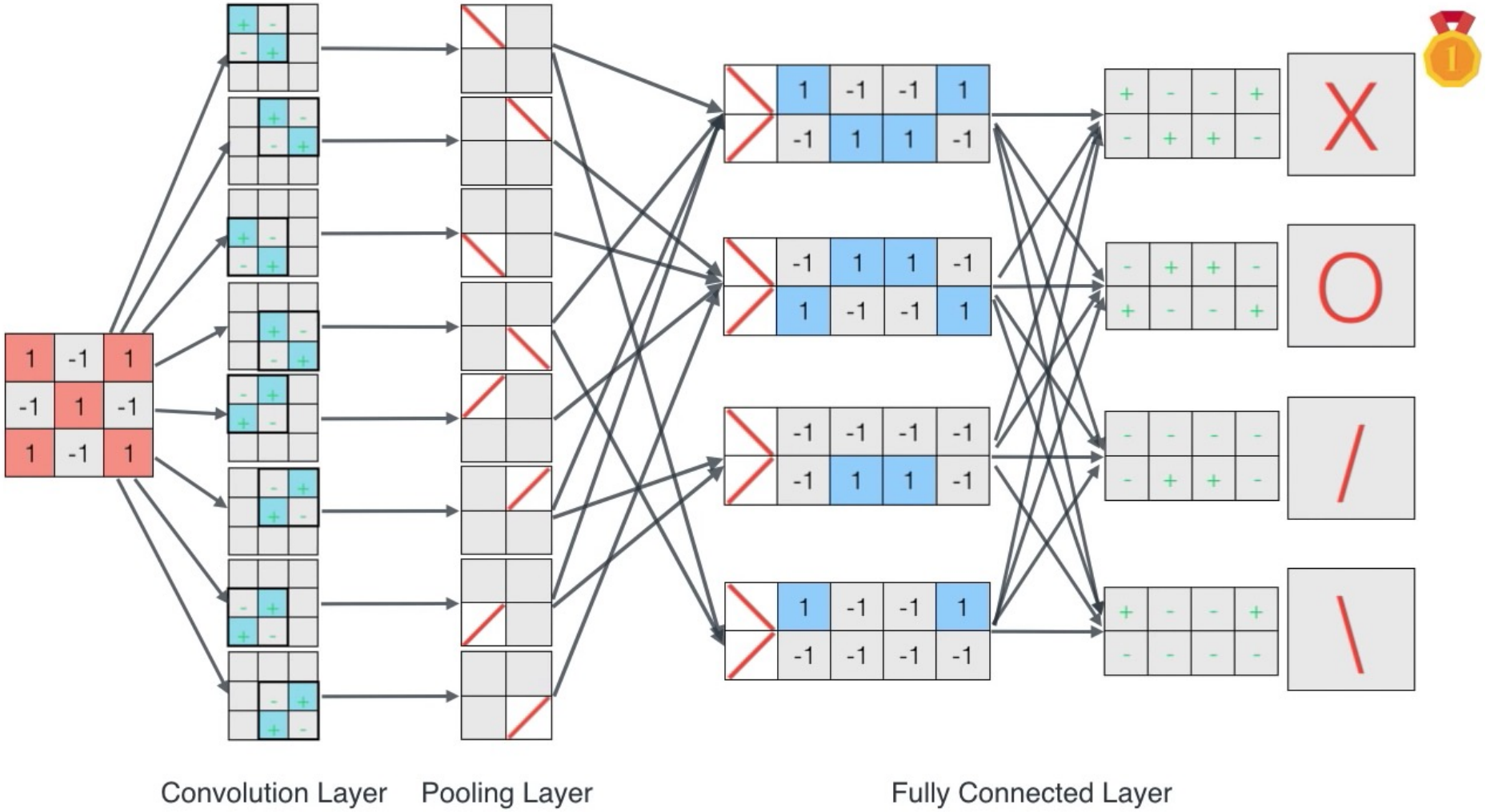
Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

A friendly introduction to Convolutional Neural Networks and Image Recognition



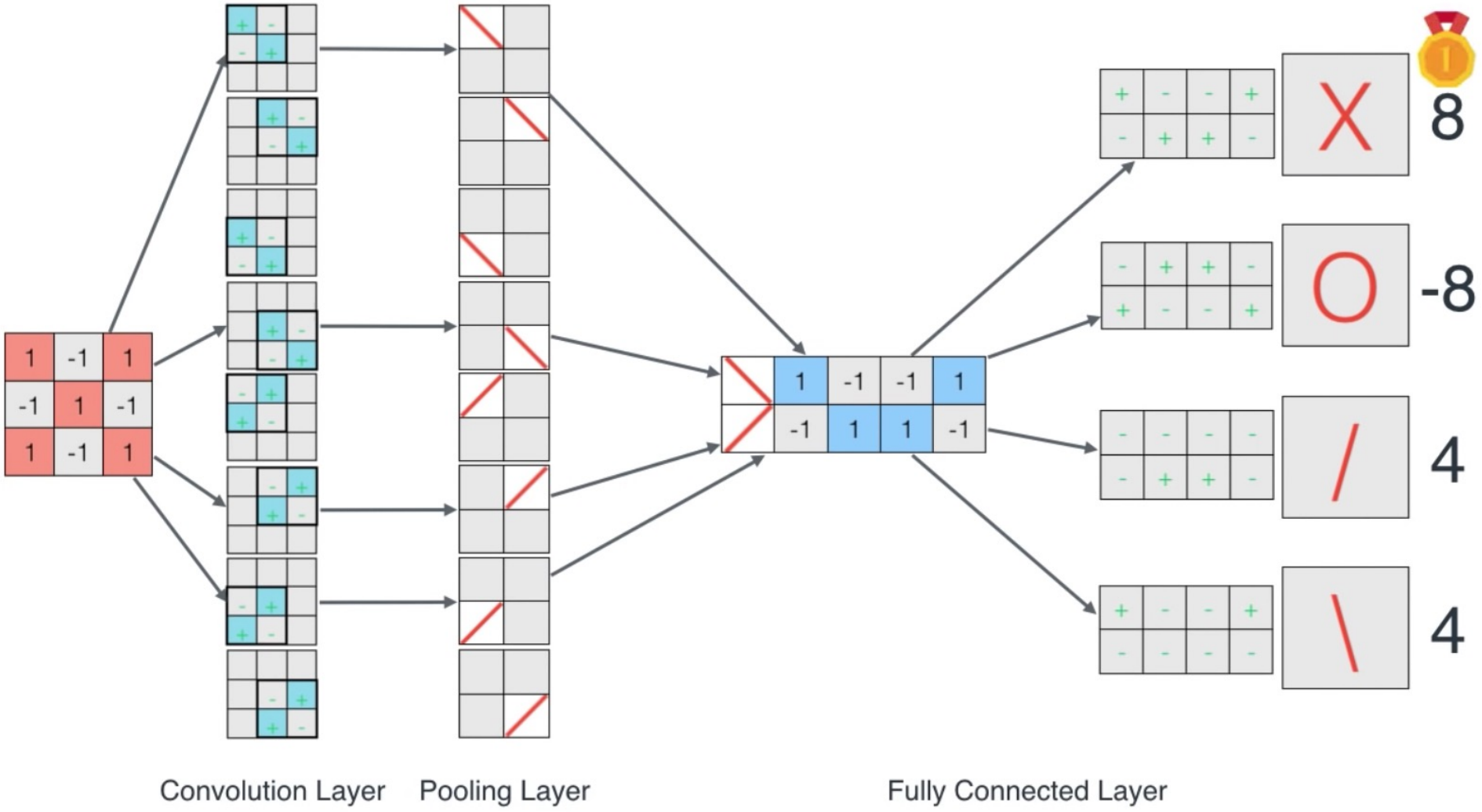
Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

A friendly introduction to Convolutional Neural Networks and Image Recognition



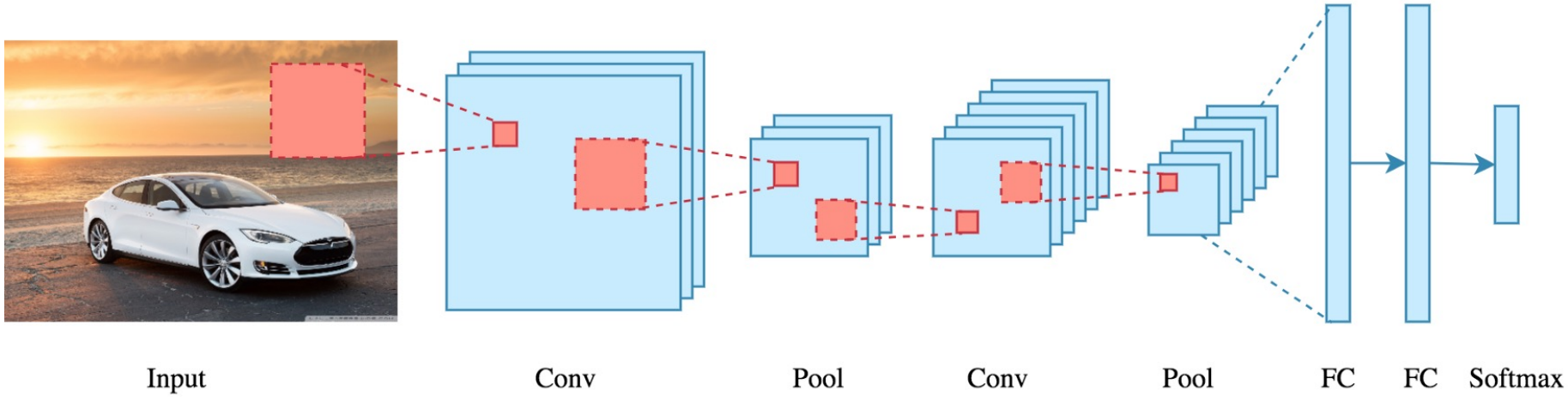
Source: Luis Serrano (2017), A friendly introduction to Convolutional Neural Networks and Image Recognition, https://www.youtube.com/watch?v=2-Ol7ZB0MmU

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

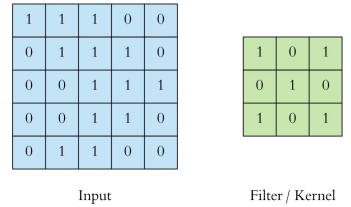
Input x Filter

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

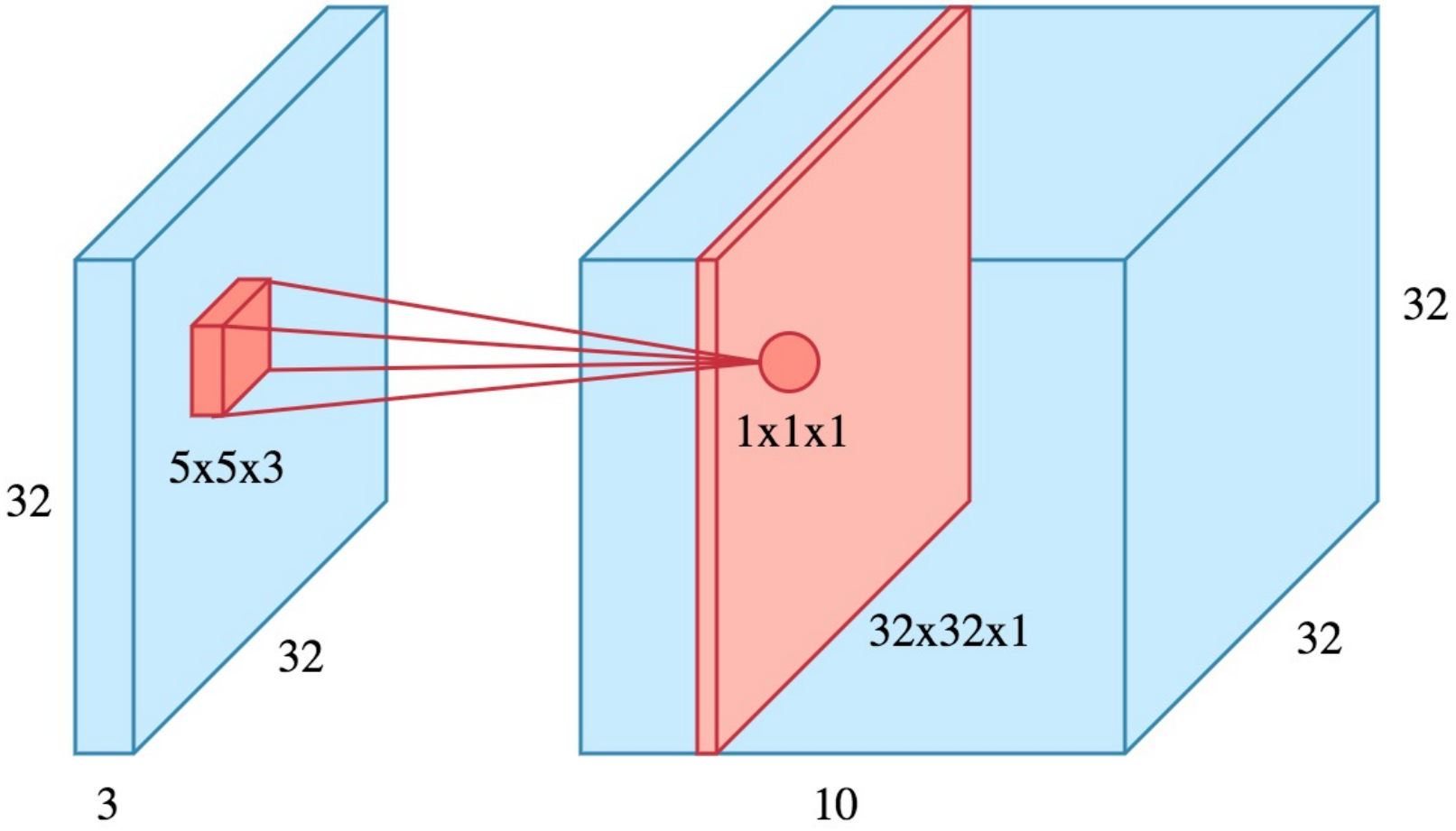
4		

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

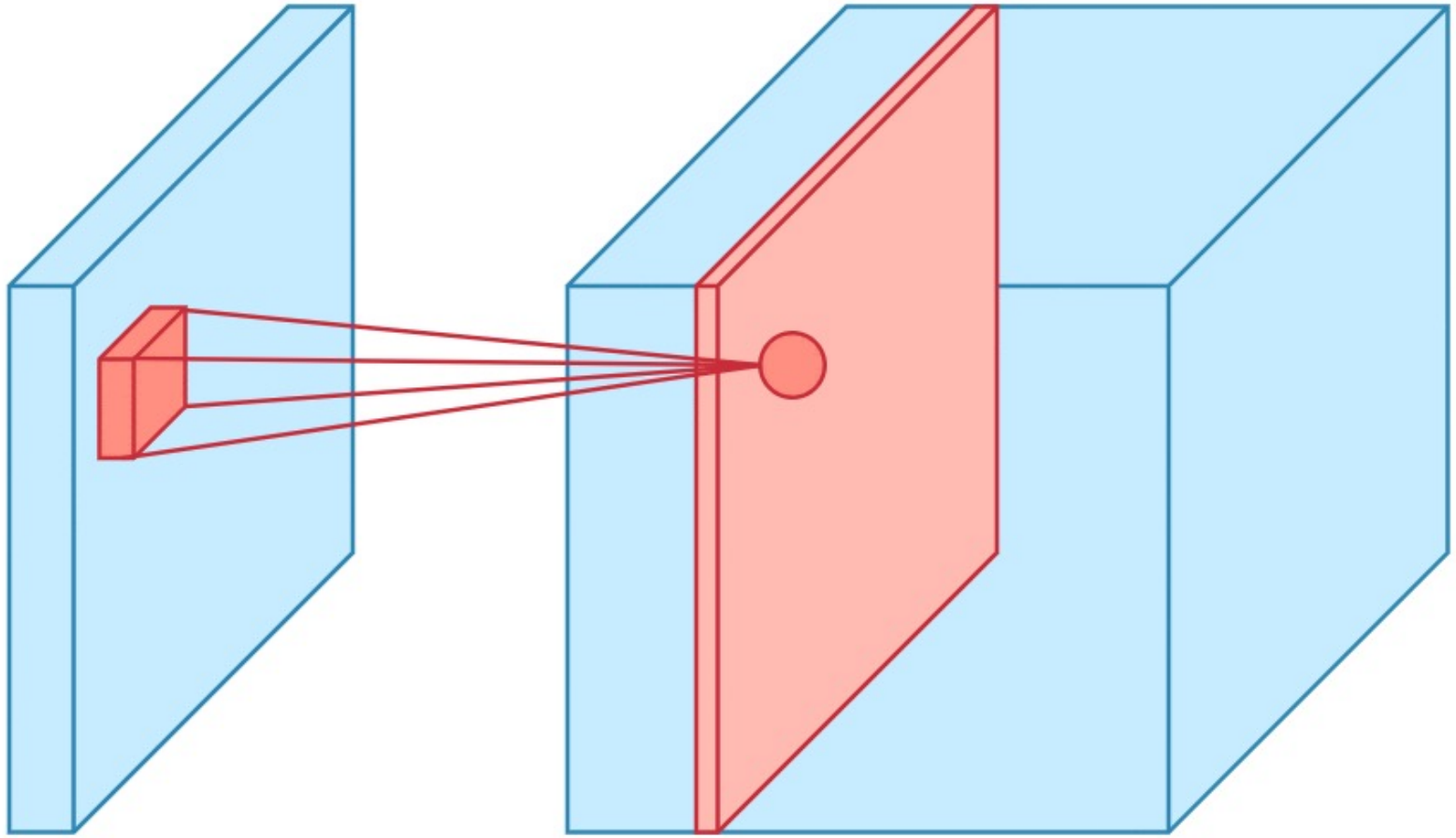


final output of the convolution layer:
a volume of size 32x32x10

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

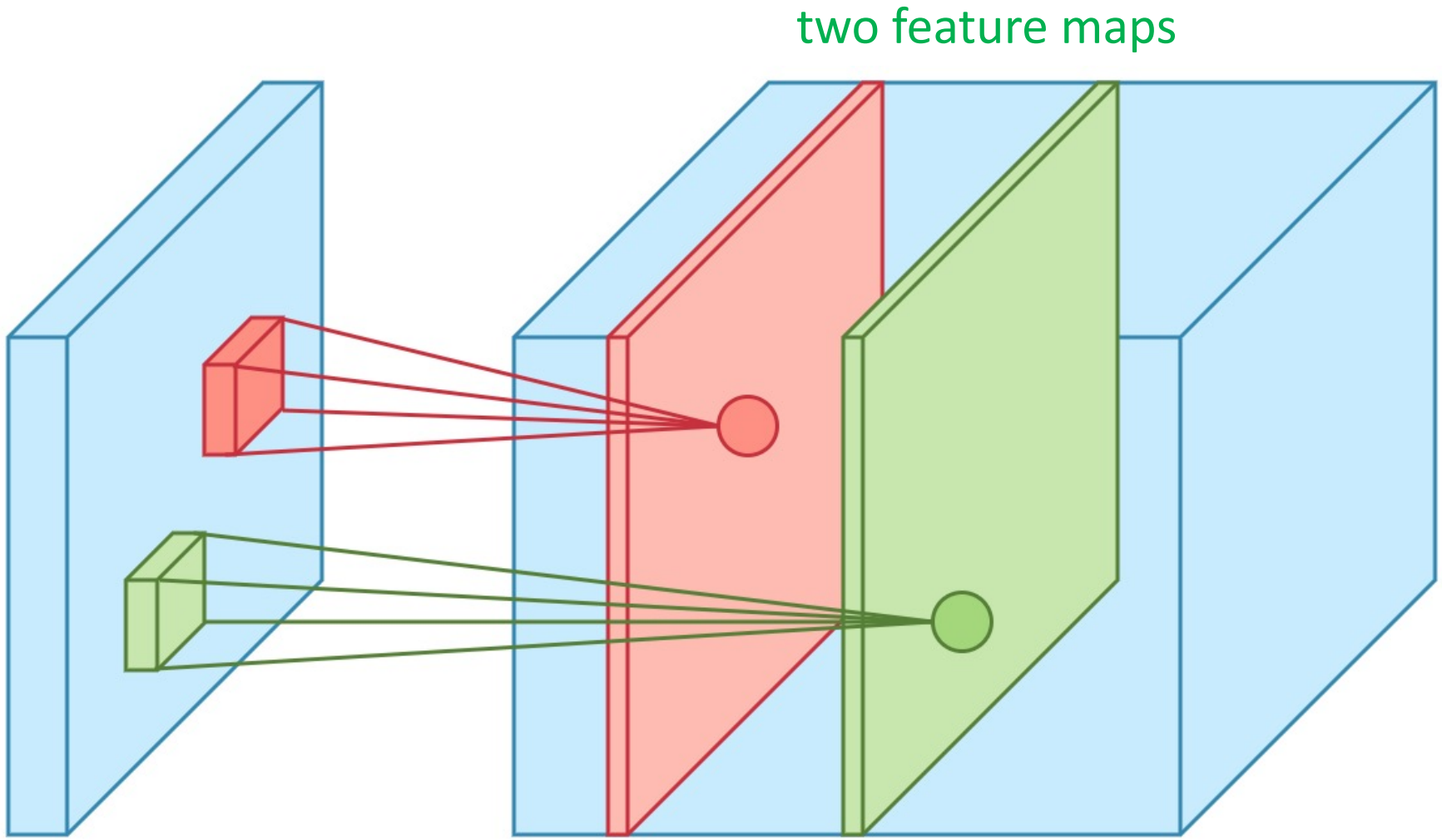
CNN Convolution Layer

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

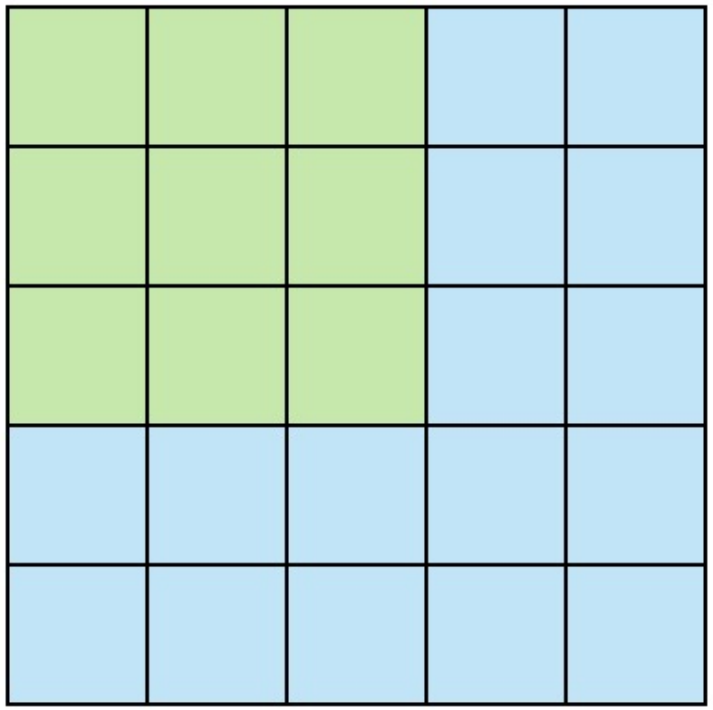
CNN Convolution Layer



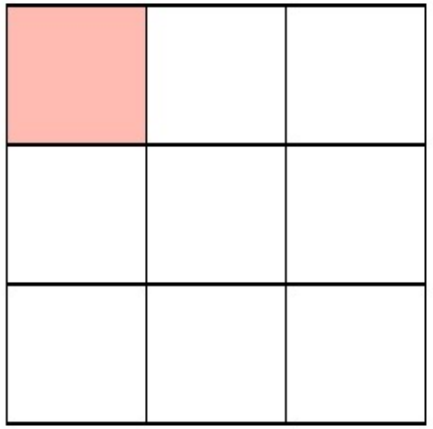
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

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



Stride 1

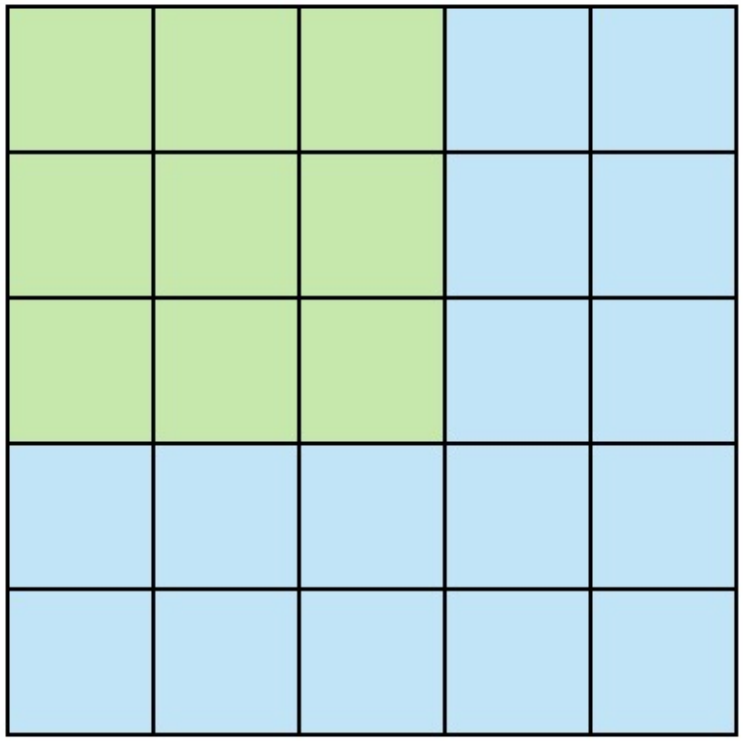


Feature Map

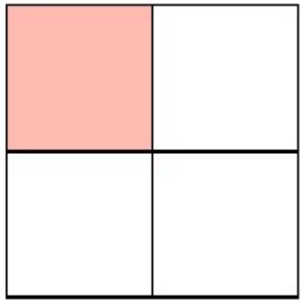
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks,
https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

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



Stride 2

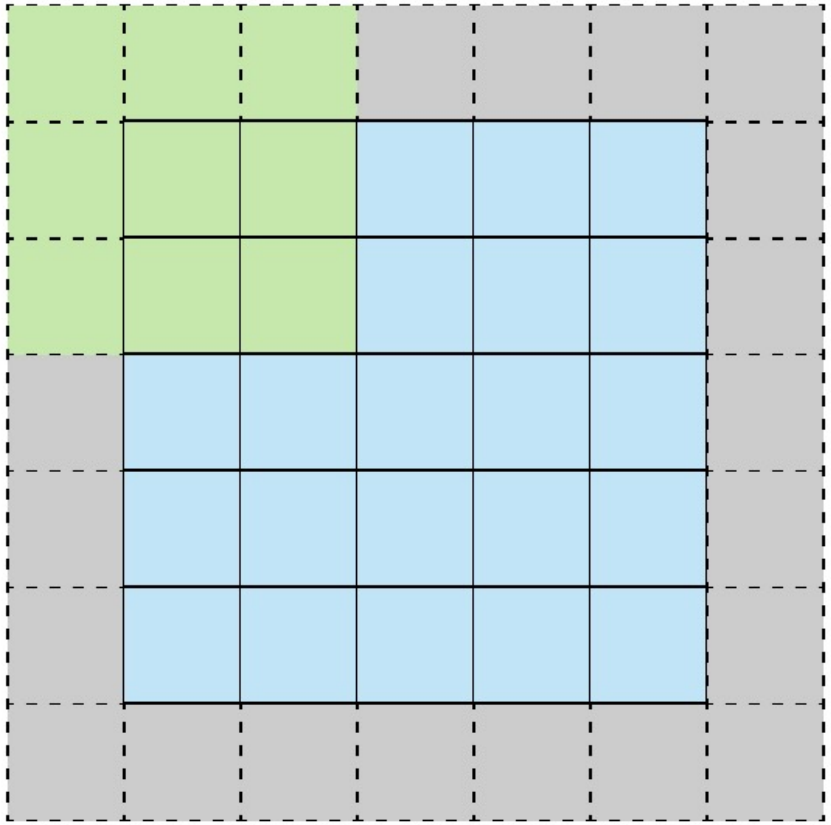


Feature Map

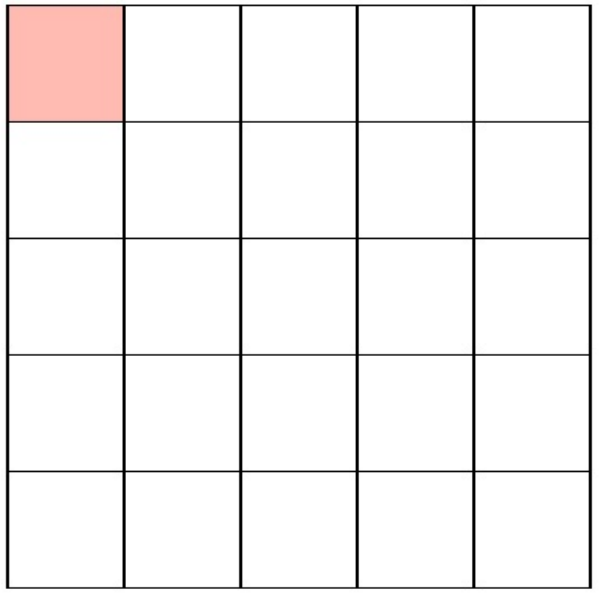
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks,
https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

CNN Convolution Layer

Stride 1 with Padding



Stride 1 with Padding

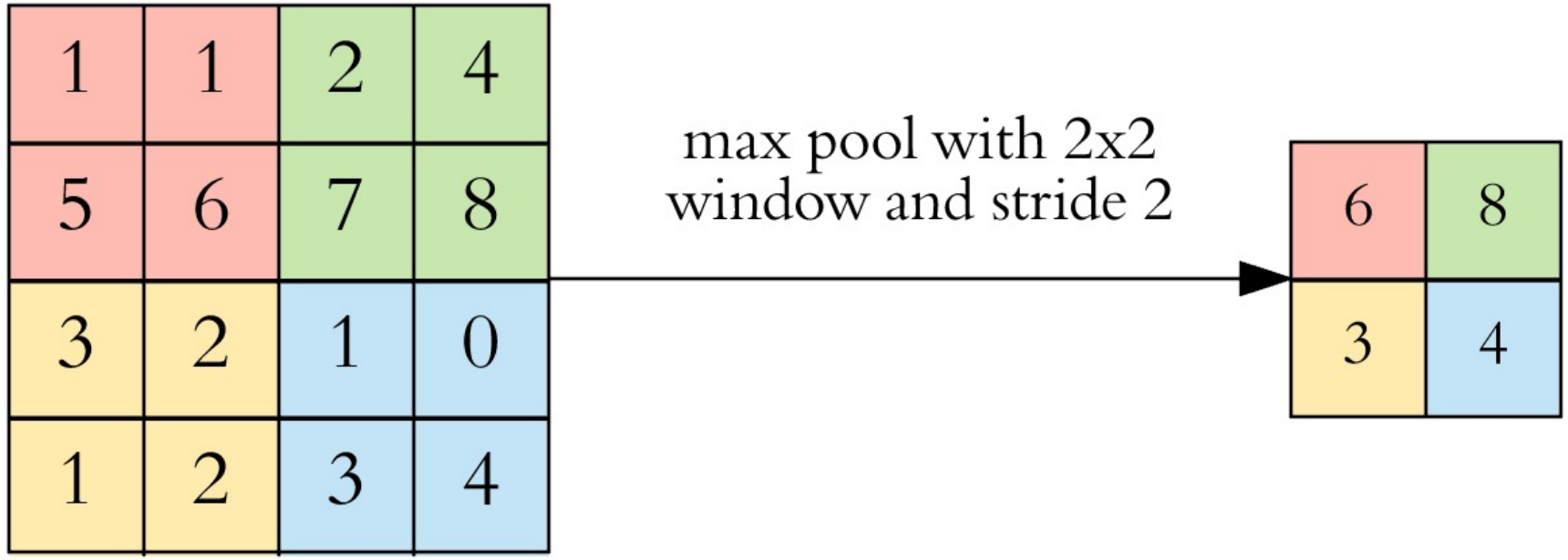


Feature Map

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

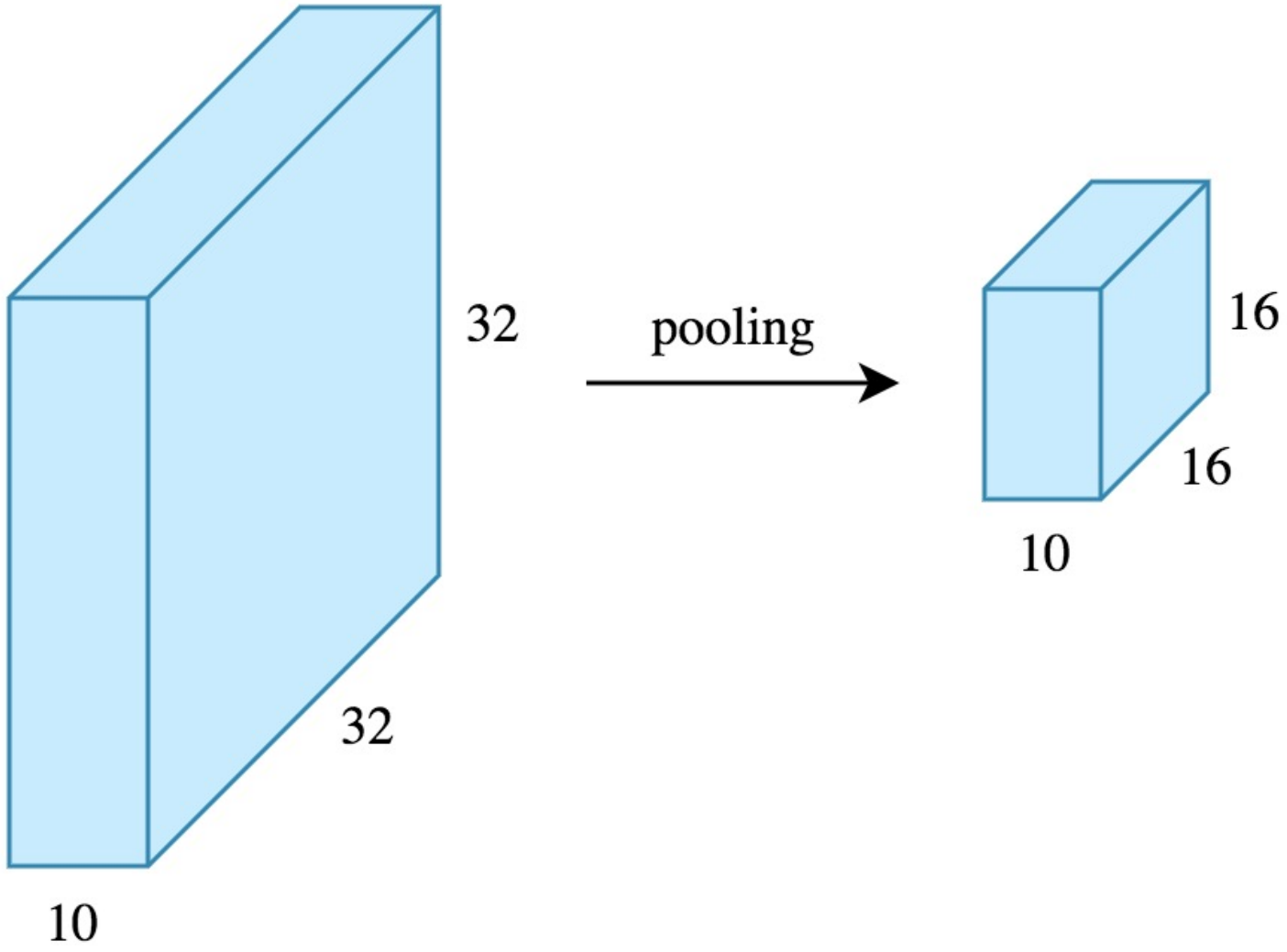
CNN Pooling Layer

Max Pooling



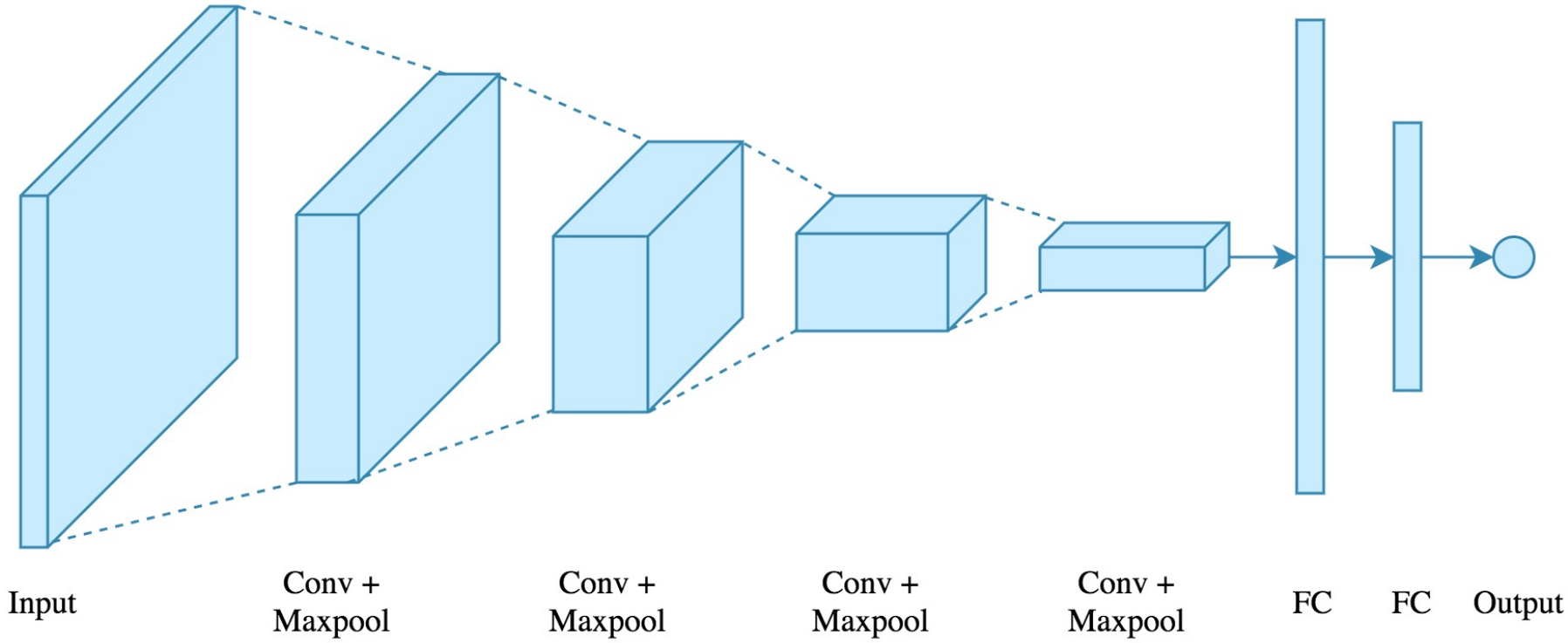
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

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

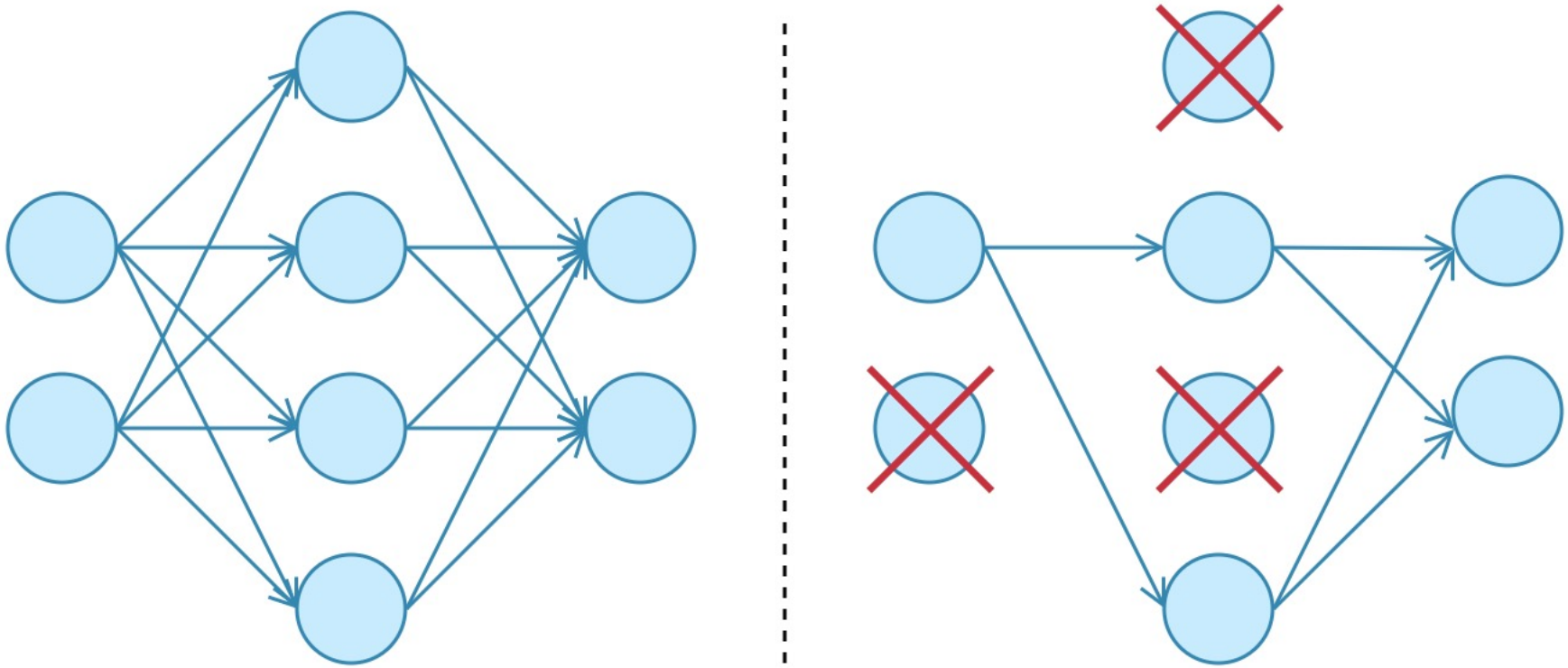
CNN Architecture
4 convolution + pooling layers, followed by 2 fully connected layers

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

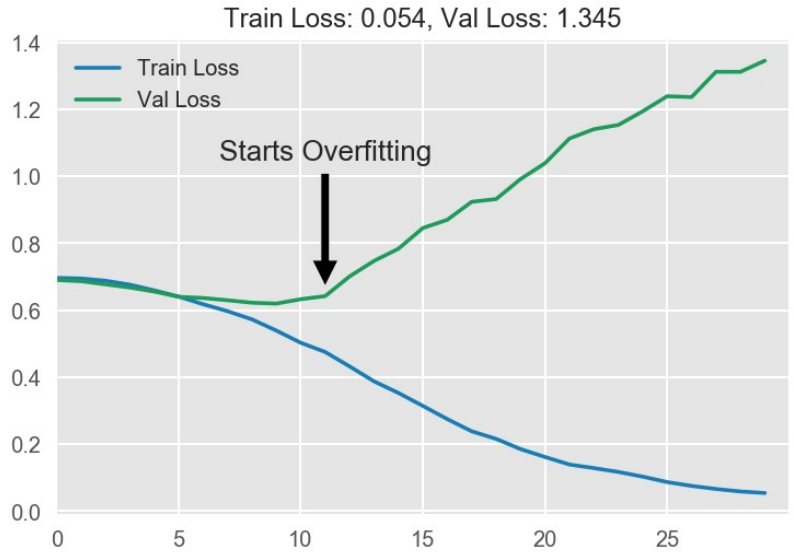


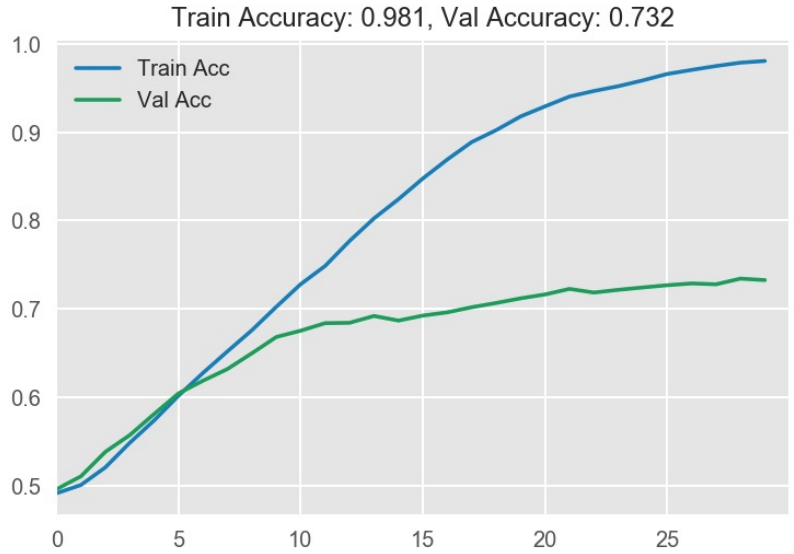
No Dropout

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

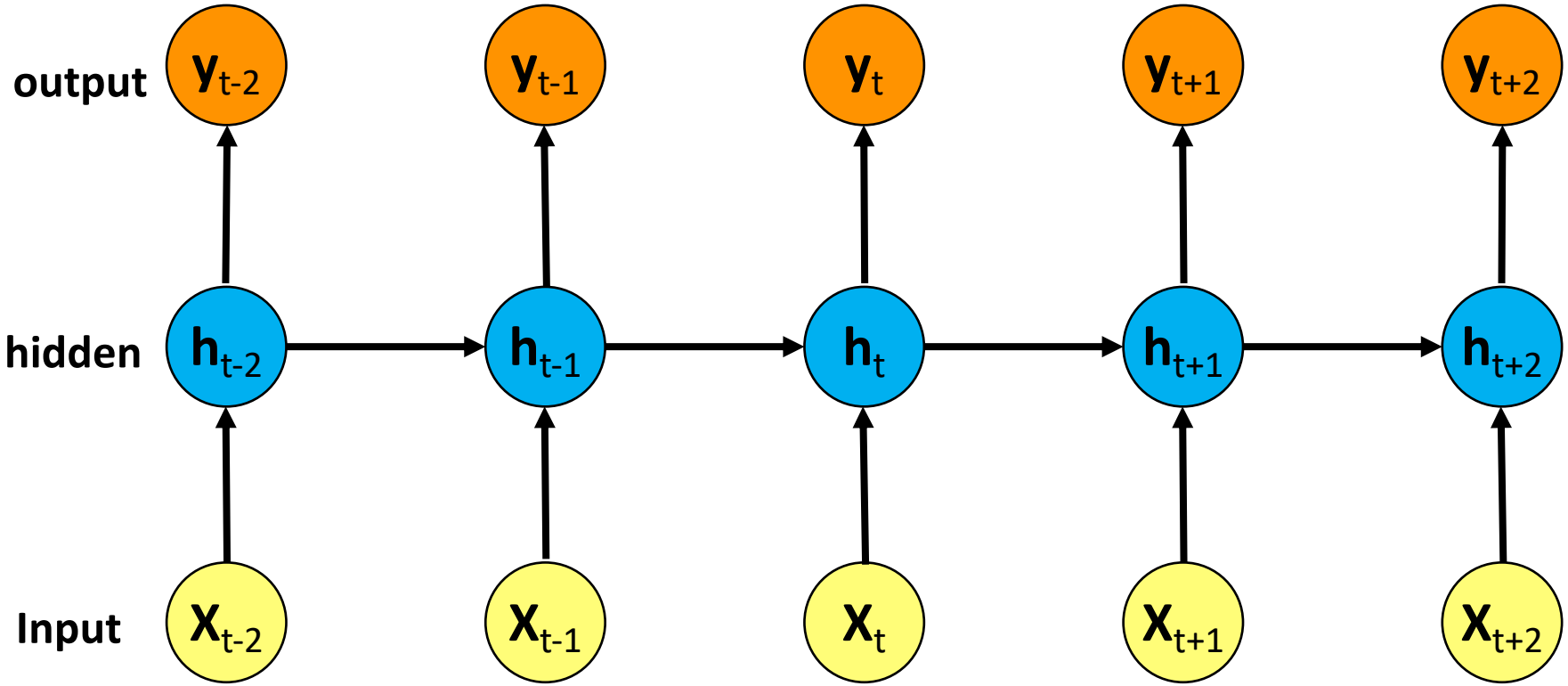




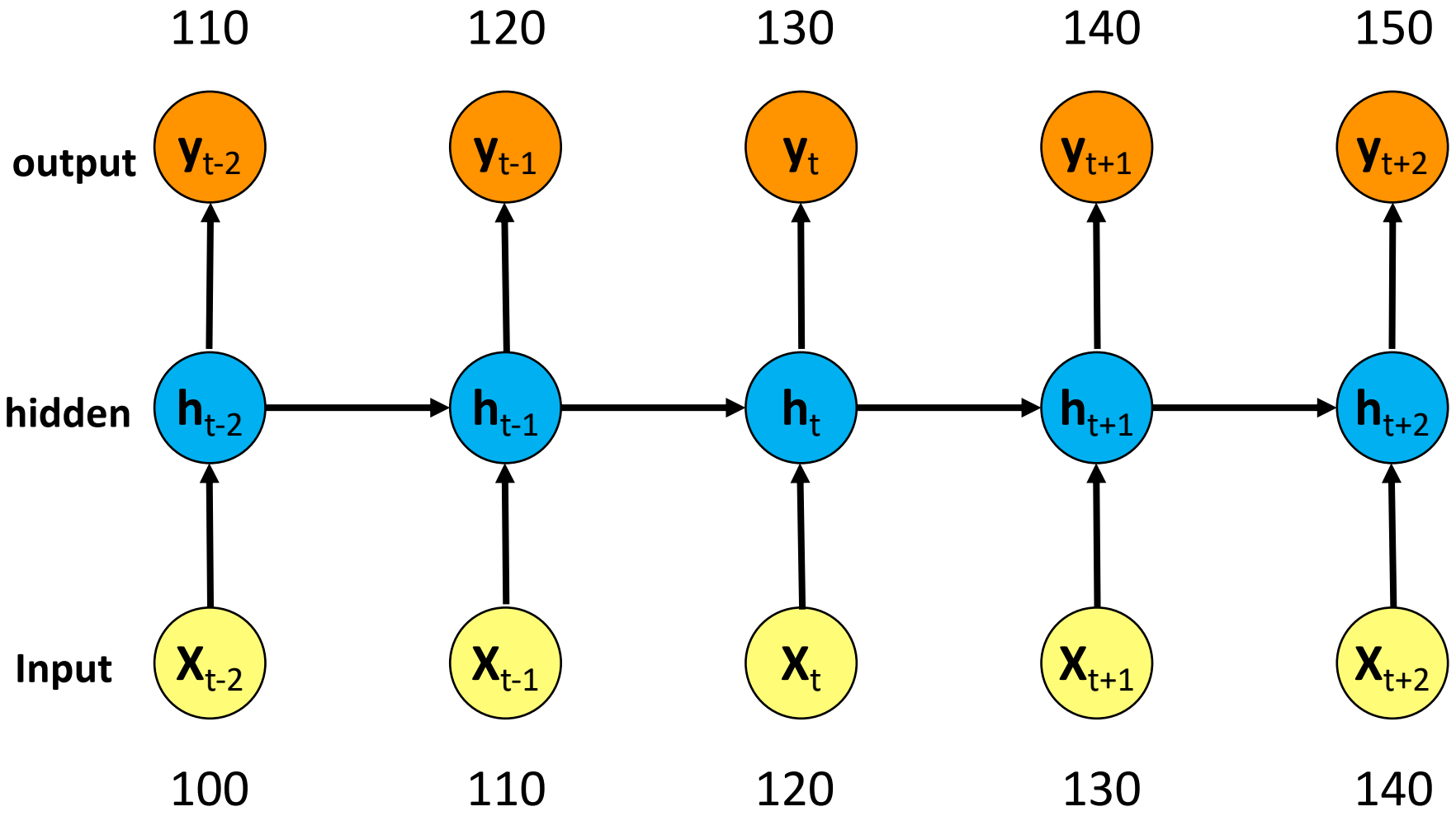
Source: Arden Dertat (2017), Applied Deep Learning - Part 4: Convolutional Neural Networks, https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2

Recurrent Neural Networks (RNN)

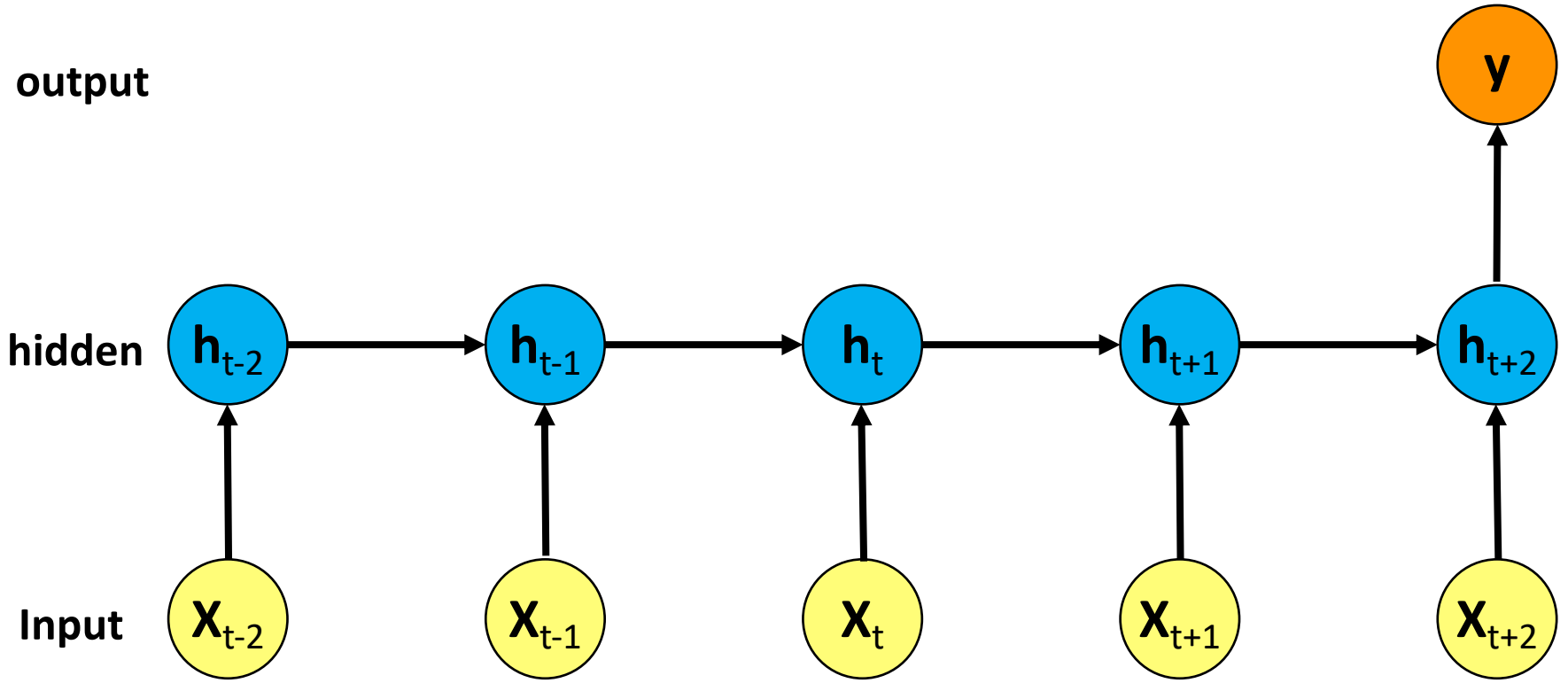
Recurrent Neural Networks (RNN)



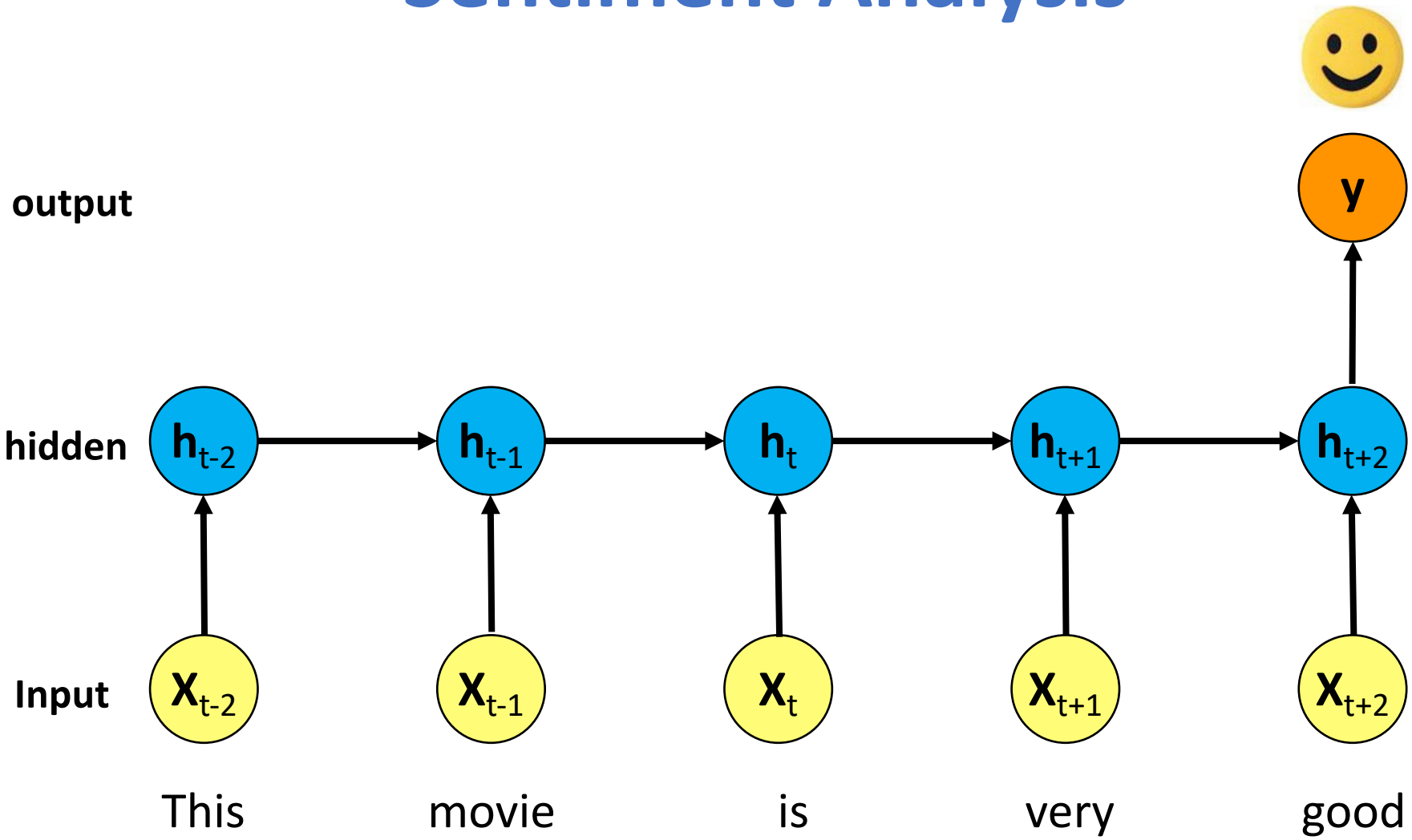
Recurrent Neural Networks (RNN)
Time Series Forecasting



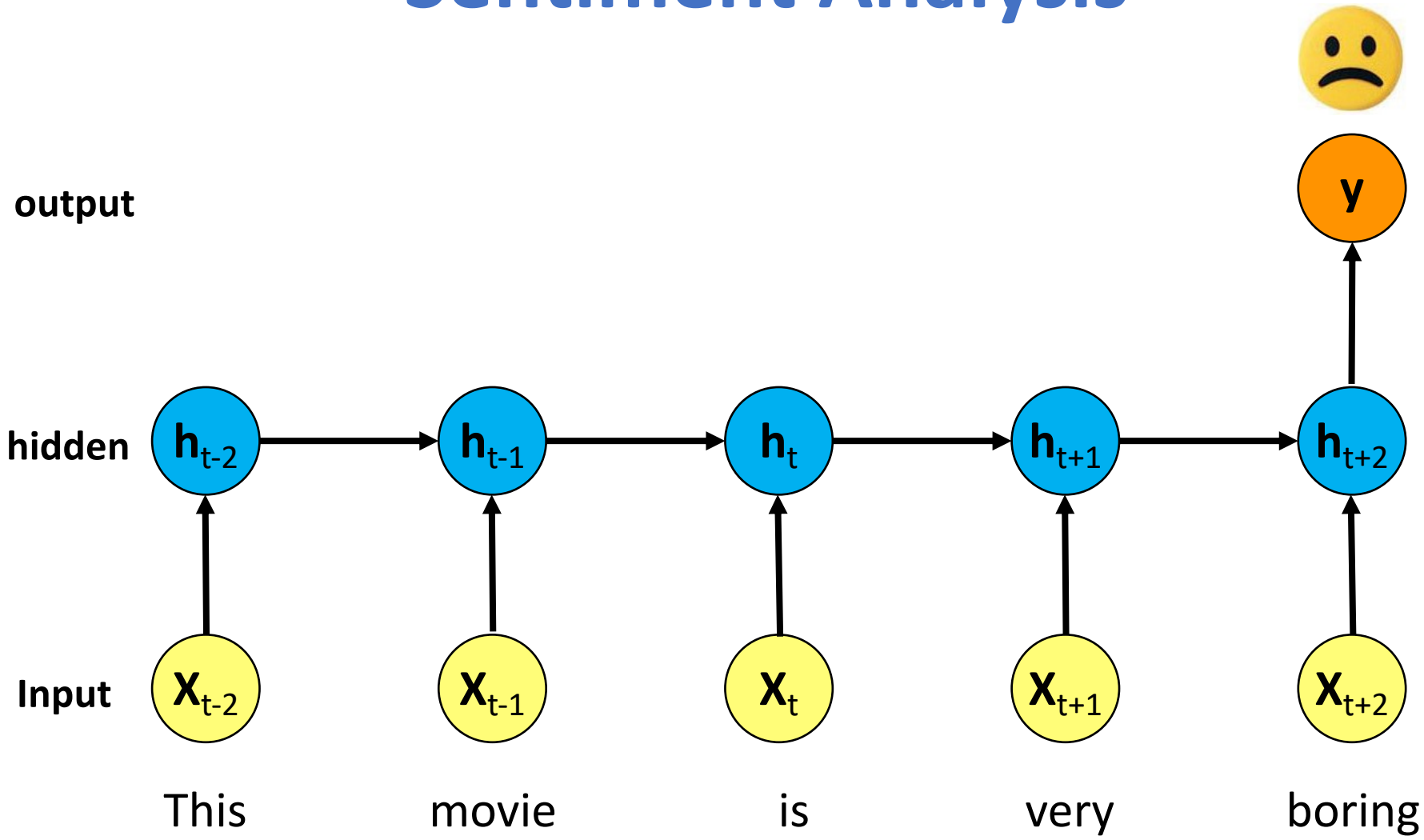
Recurrent Neural Networks (RNN)



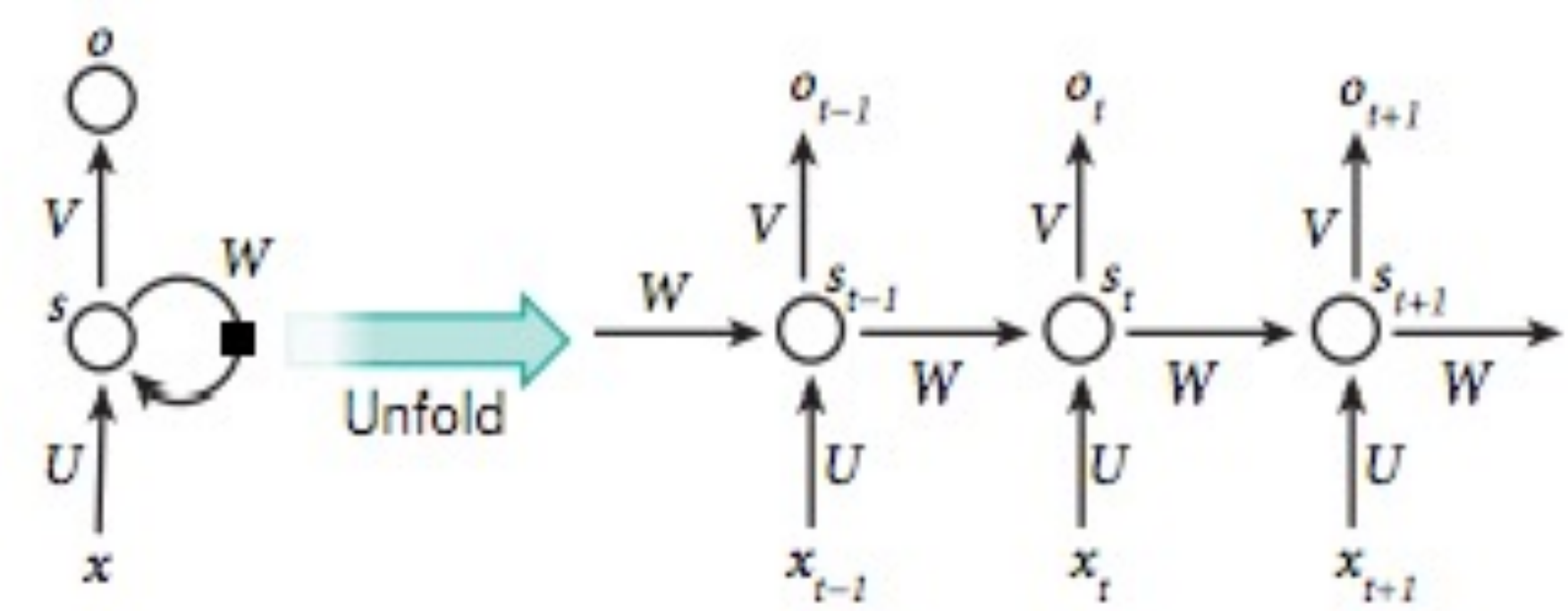
Recurrent Neural Networks (RNN)
Sentiment Analysis



Recurrent Neural Networks (RNN)
Sentiment Analysis

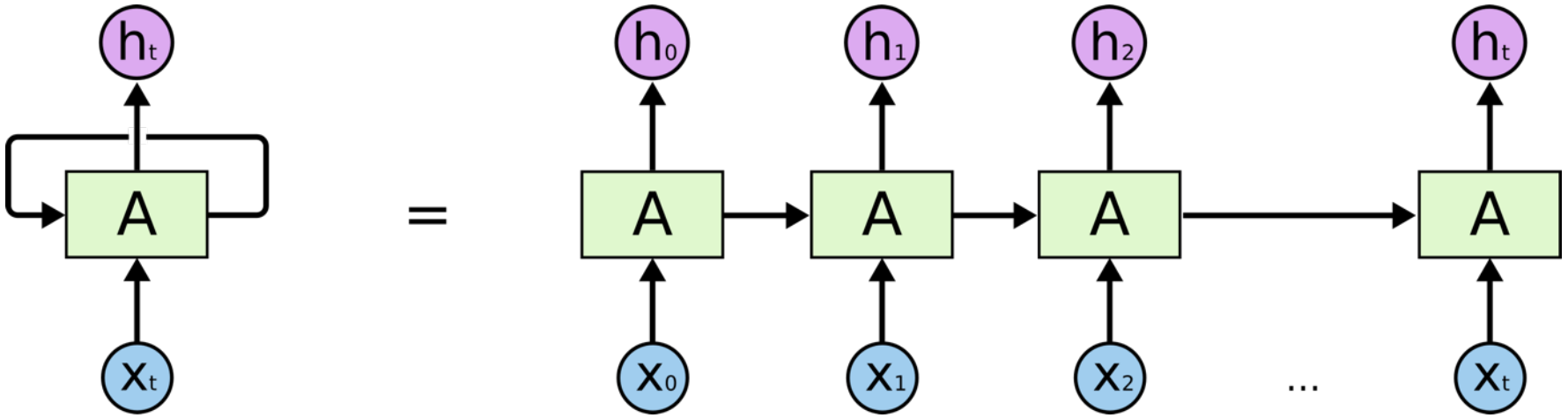


Recurrent Neural Network (RNN)



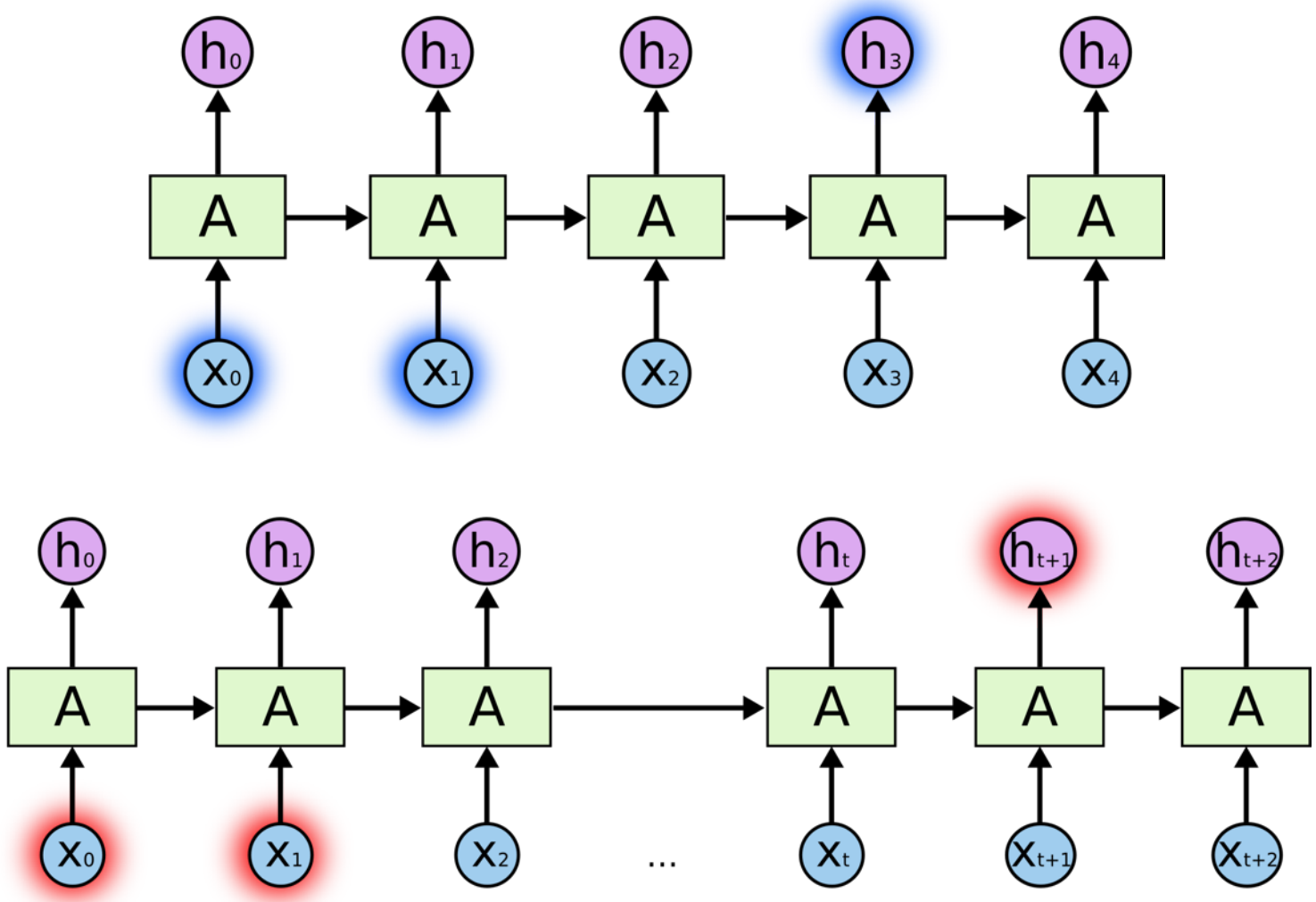
Source: LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." Nature 521, no. 7553 (2015): 436-444.

RNN



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

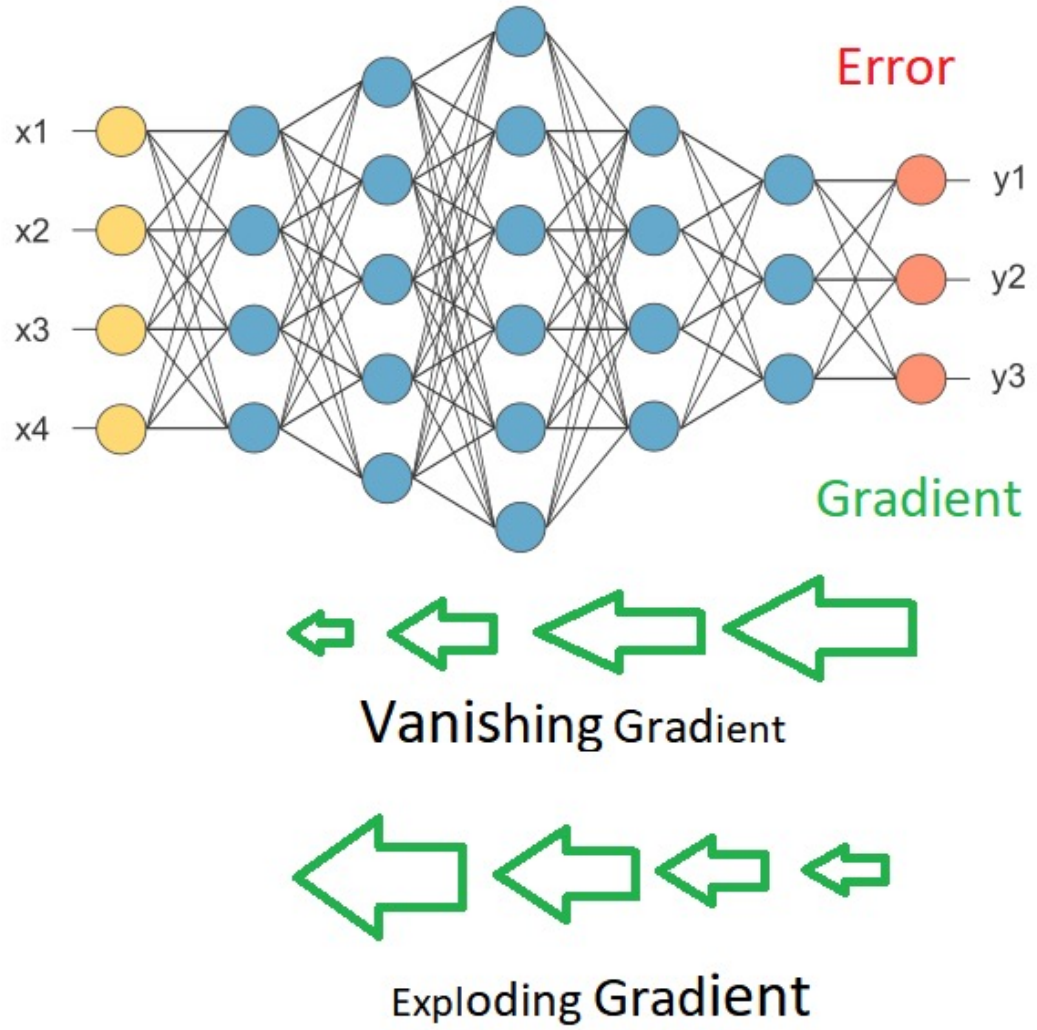
RNN long-term dependencies



I grew up in France… I speak fluent French.

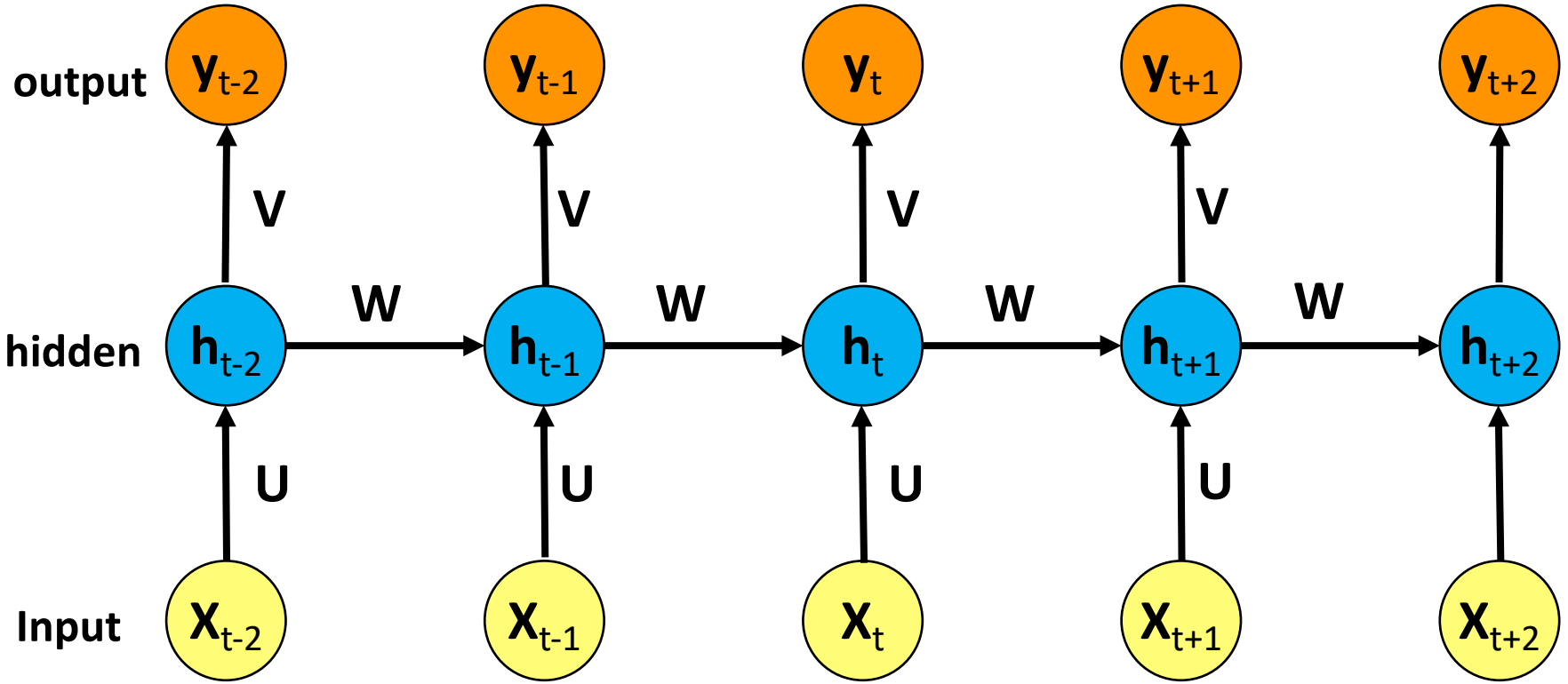
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Vanishing Gradient
Exploding Gradient

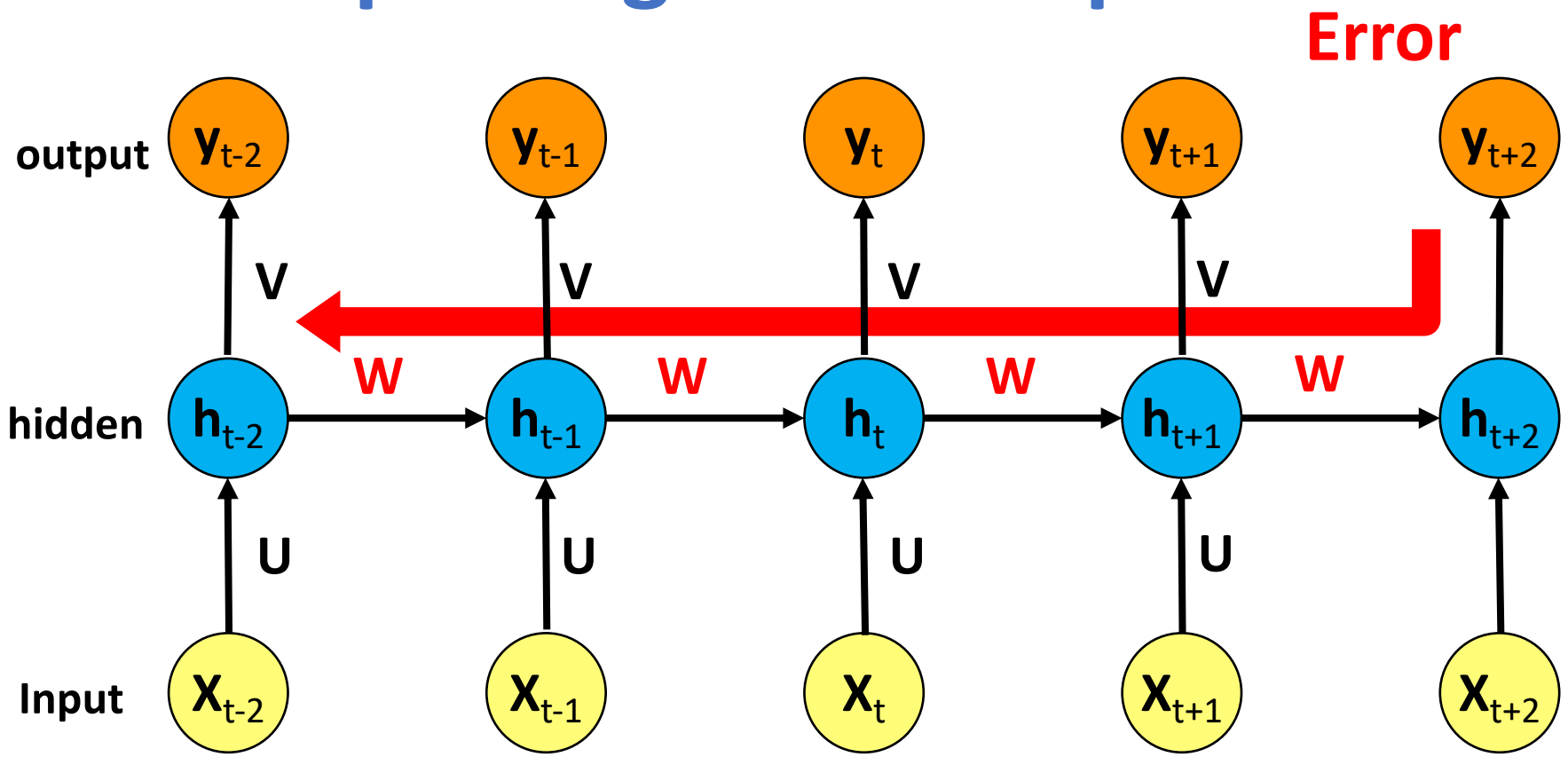


Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

Recurrent Neural Networks (RNN)



RNN
Vanishing Gradient problem
Exploding Gradient problem

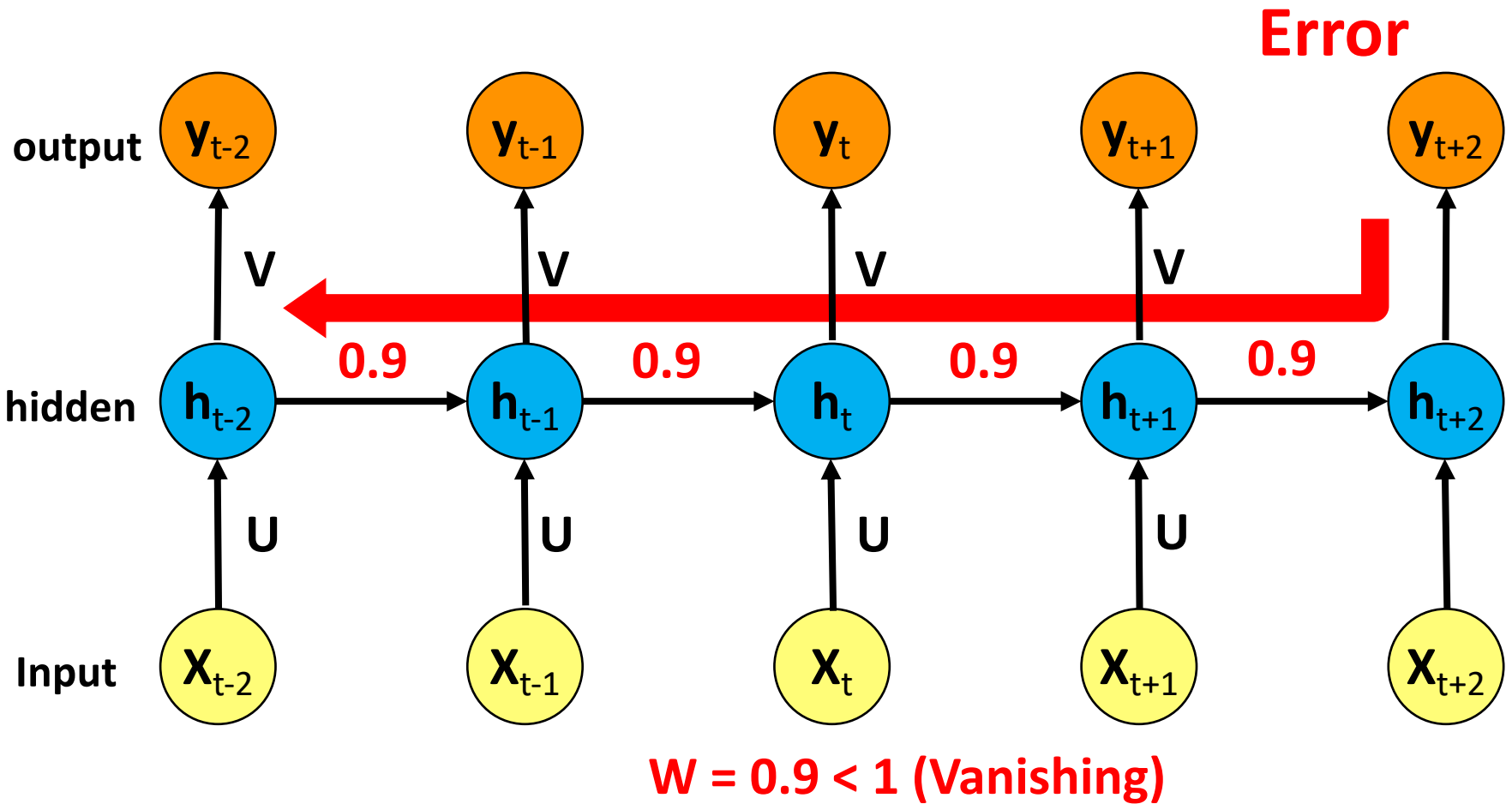


if |W| < 1 (Vanishing)

if |W| > 1 (Exploding)

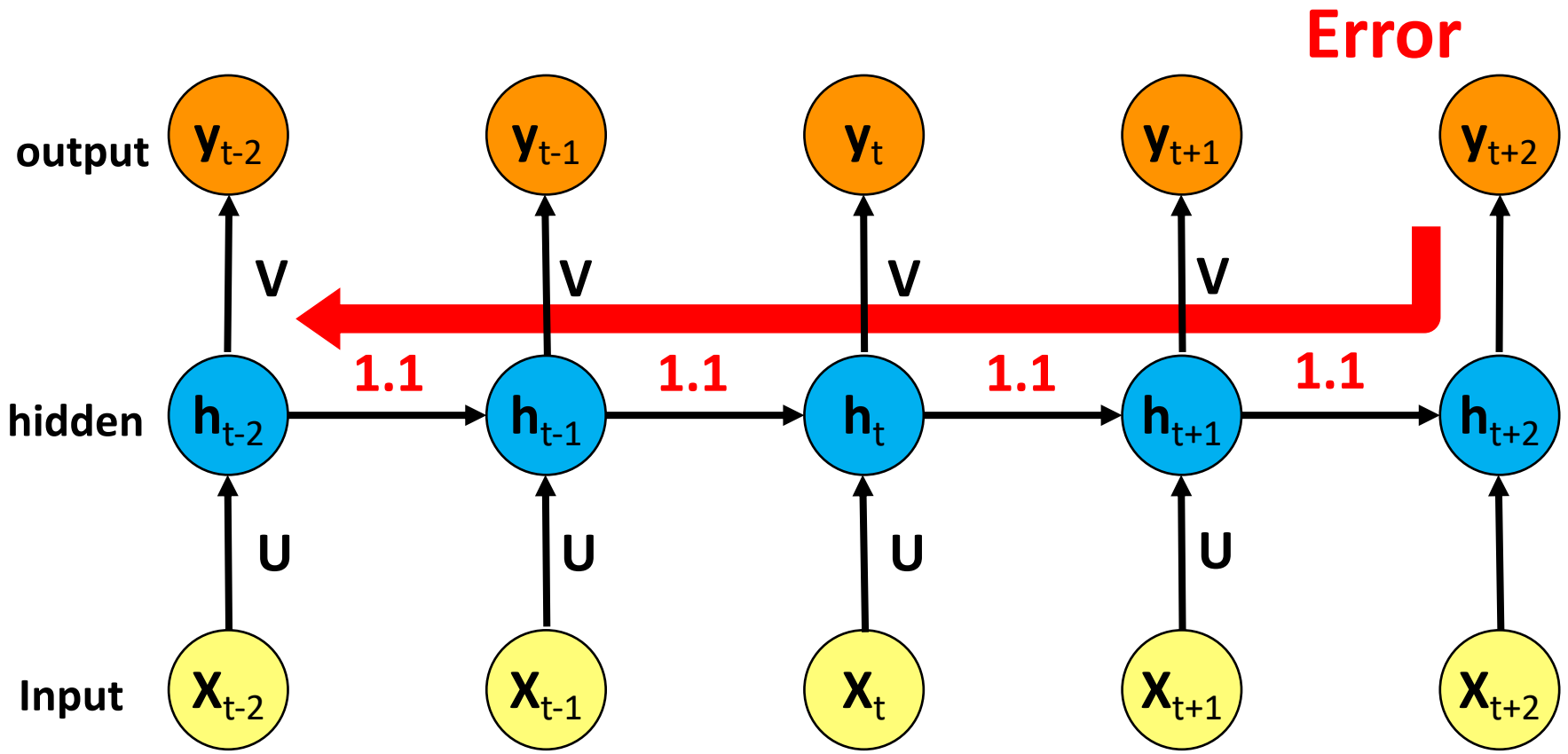
Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

RNN
Vanishing Gradient problem



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

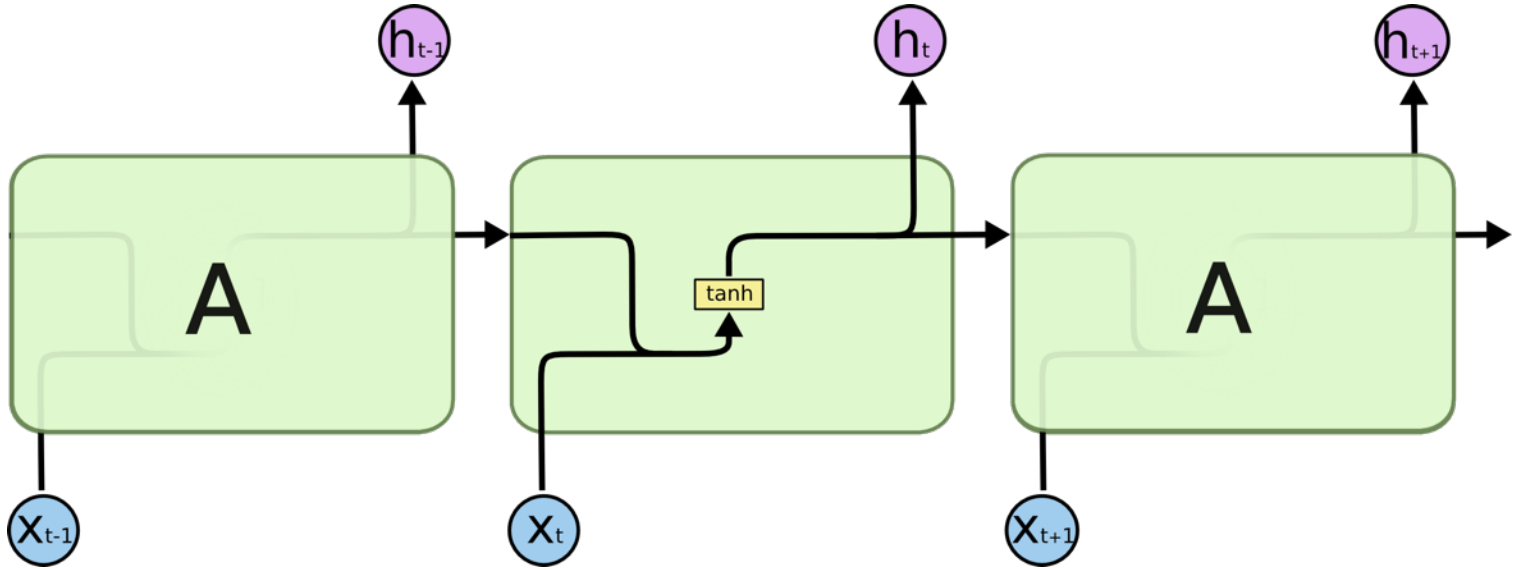
RNN
Exploding Gradient problem



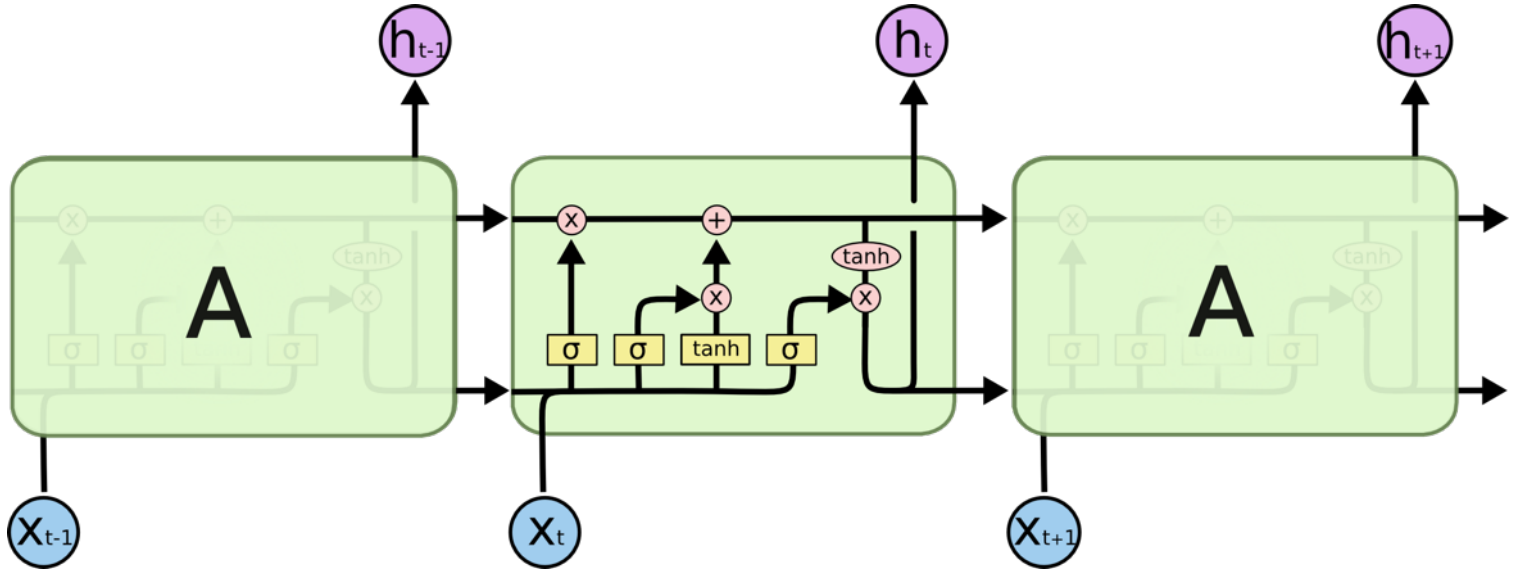
Source: https://medium.com/deep-math-machine-learning-ai/chapter-10-1-deepnlp-lstm-long-short-term-memory-networks-with-math-21477f8e4235

RNN LSTM

RNN

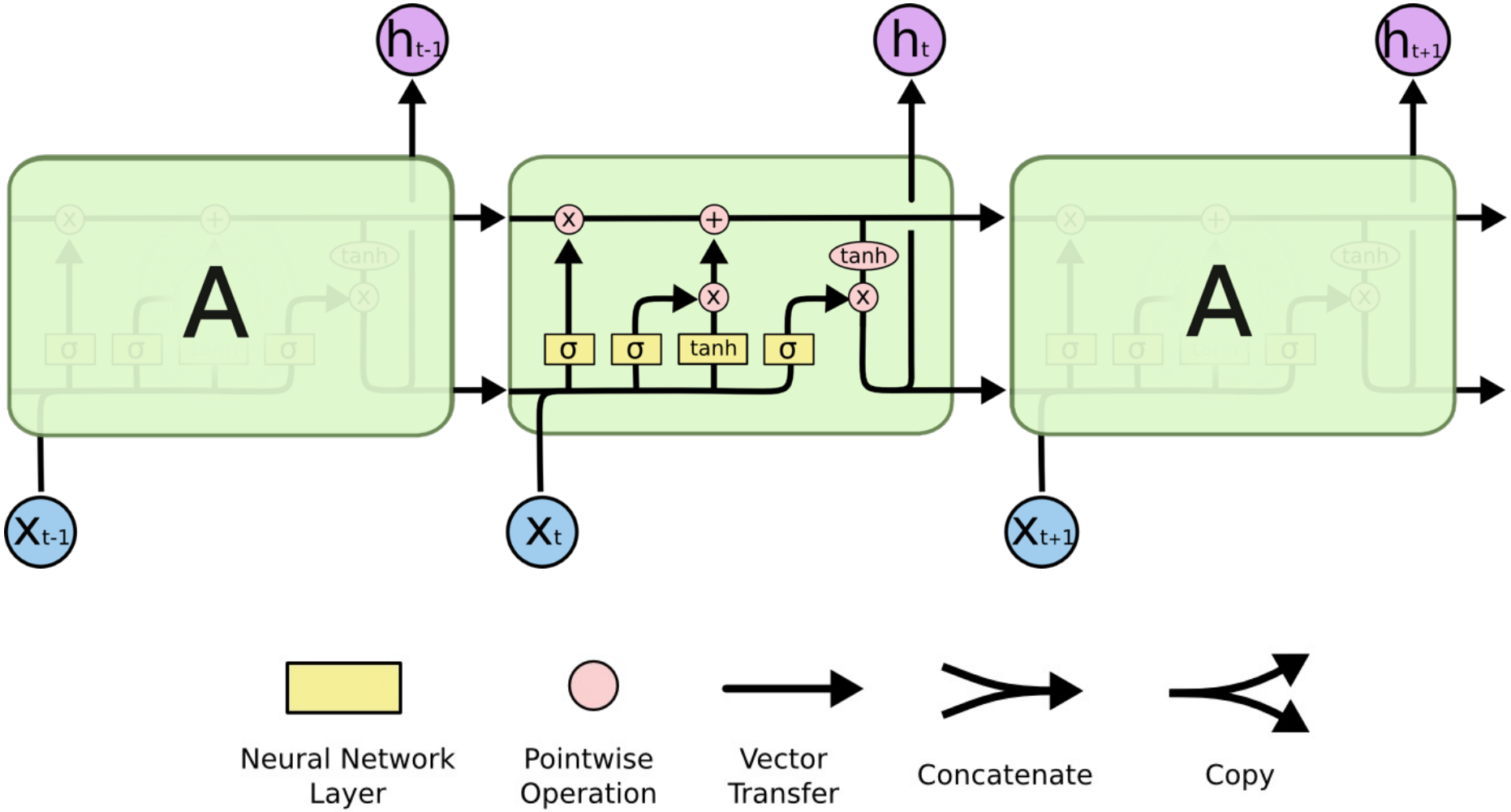


LSTM



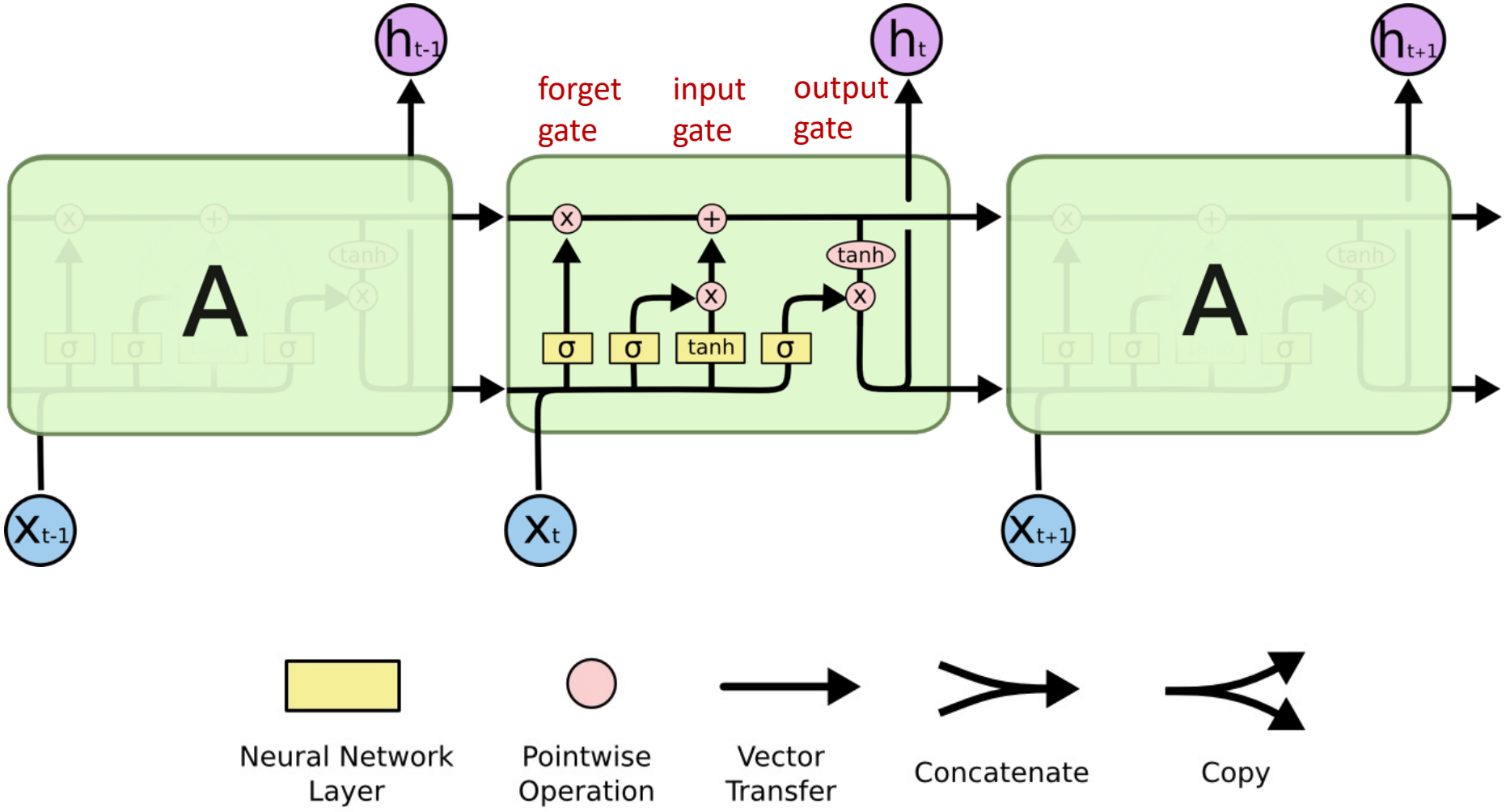
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory (LSTM)



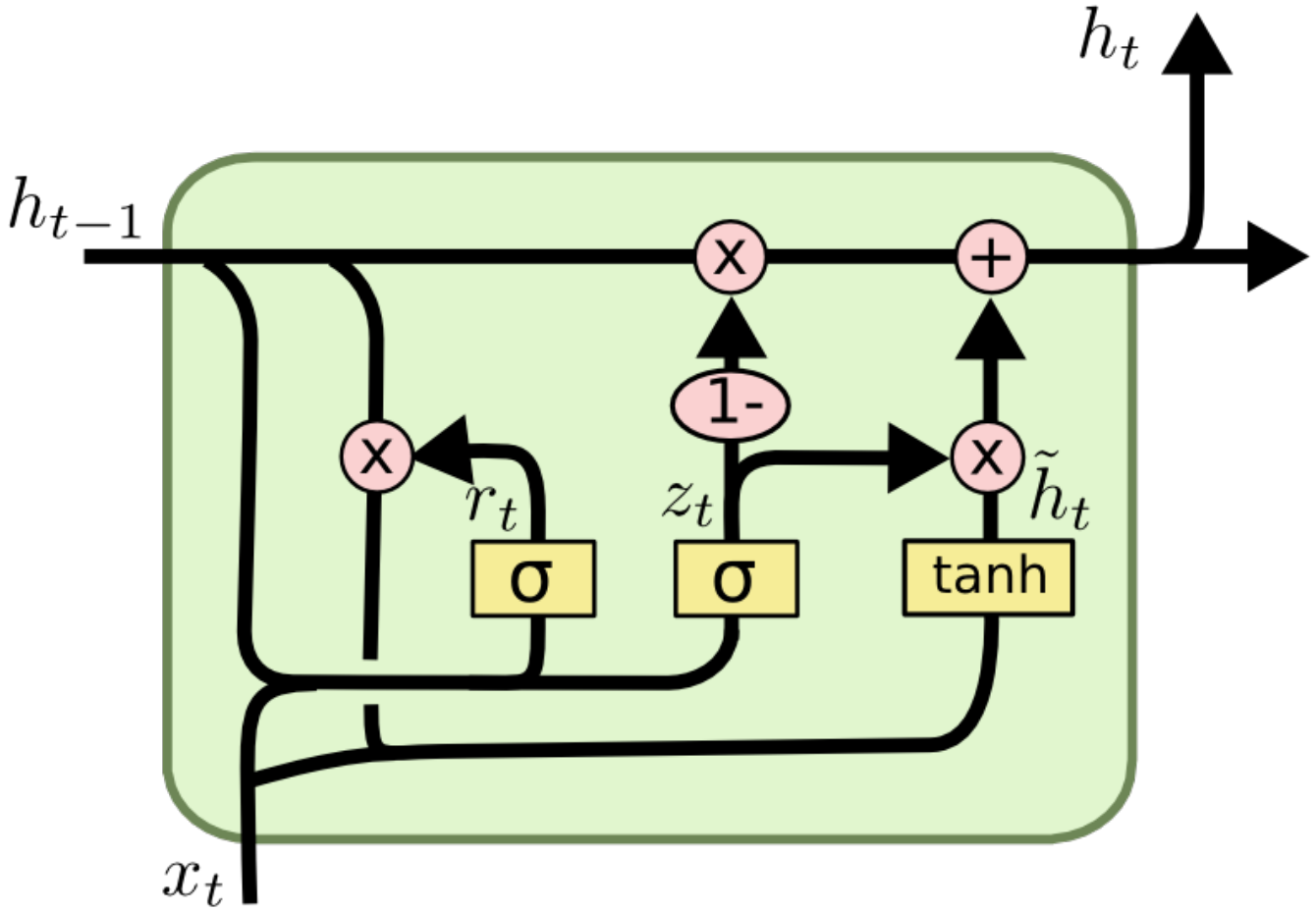
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Long Short Term Memory (LSTM)



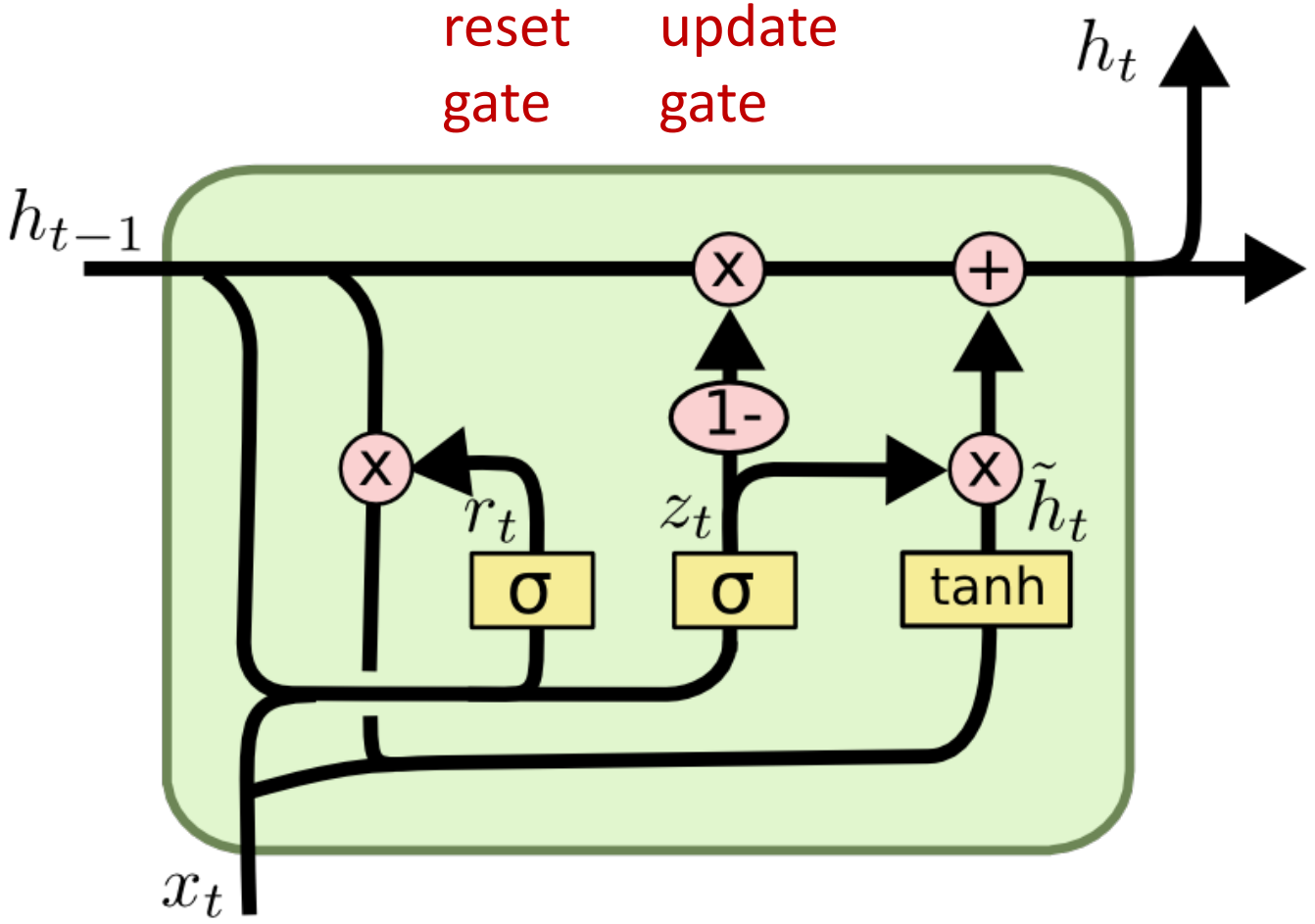
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit (GRU)



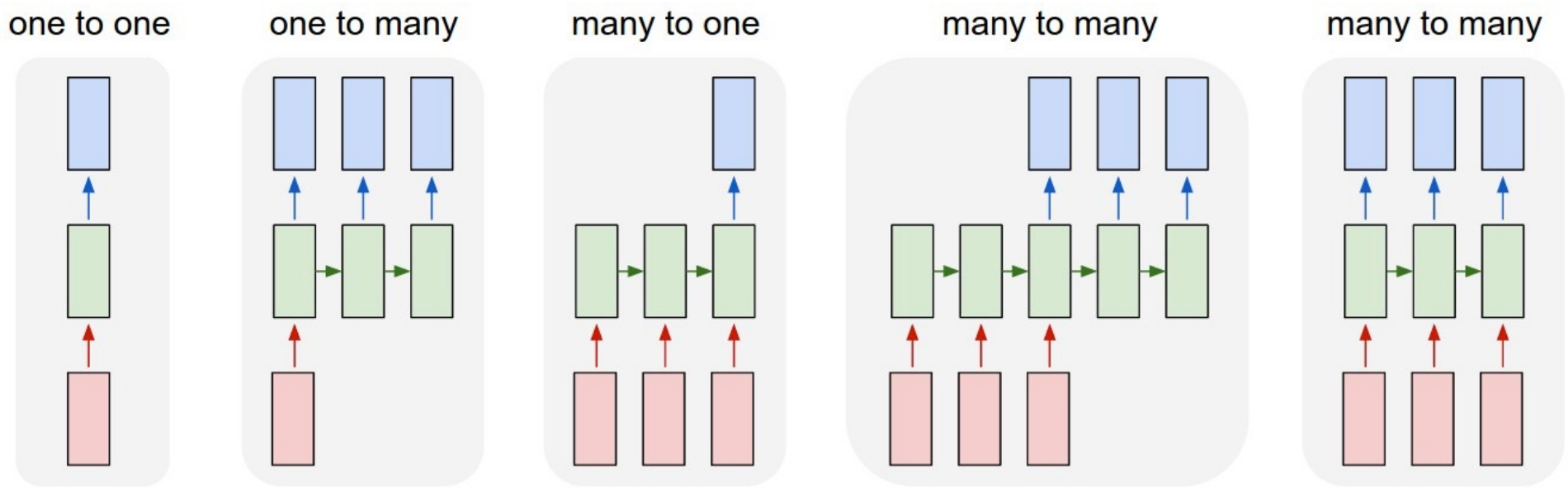
Source: Christopher Olah, (2015) Understanding LSTM Networks, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Unit (GRU)



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

LSTM Recurrent Neural Network



Traditional Neural Network

Music Generation

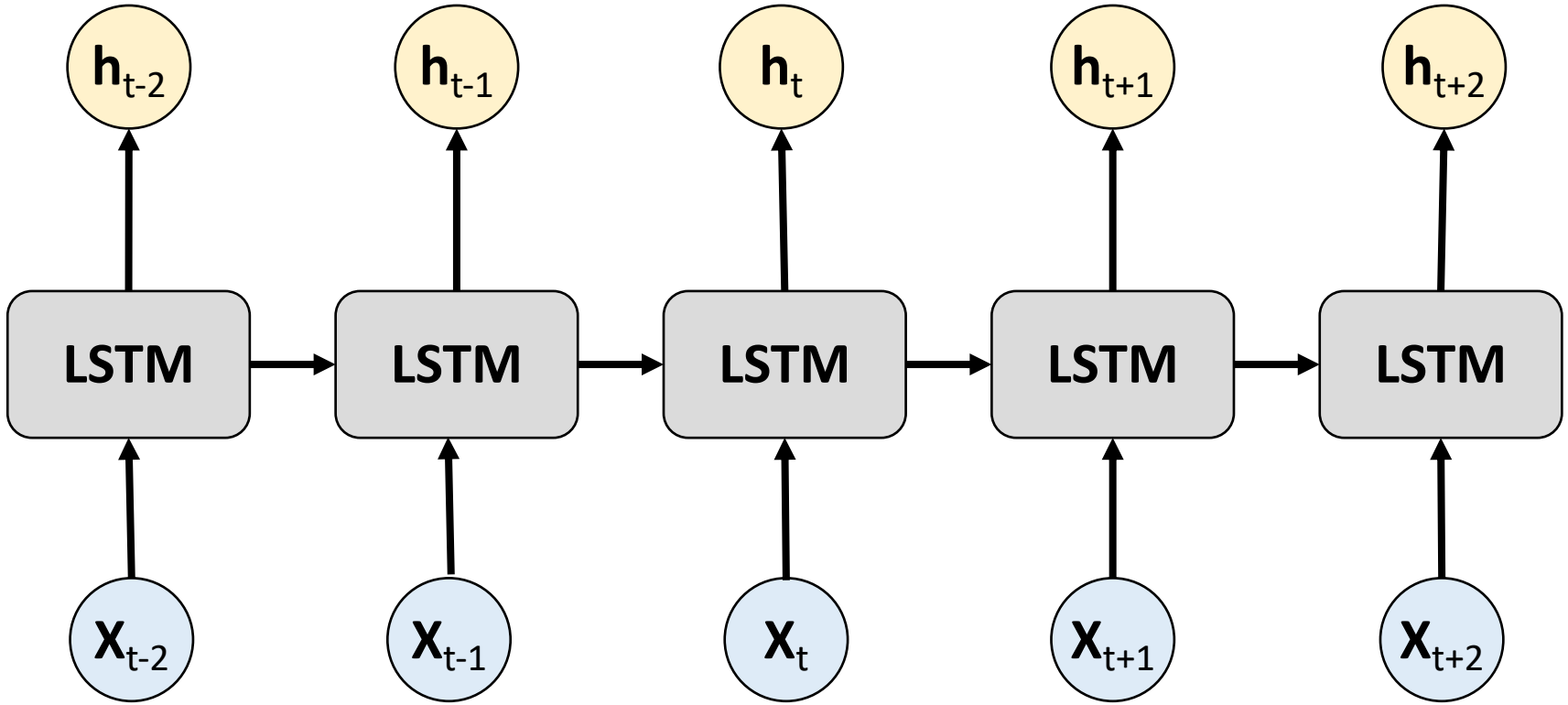
Sentiment Classification

Name Entity Recognition

Machine Translation

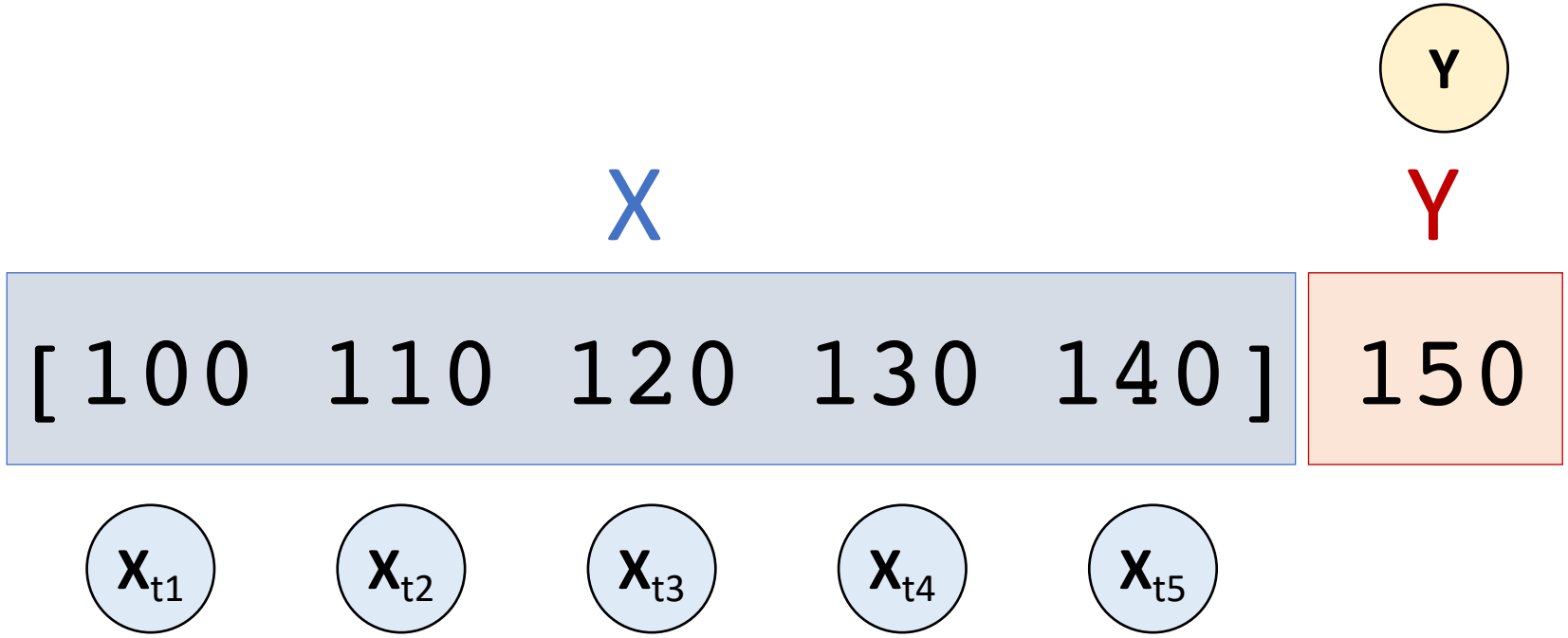
Source: https://github.com/Vict0rSch/deep_learning/tree/master/keras/recurrent

Long Short Term Memory (LSTM) for Time Series Forecasting

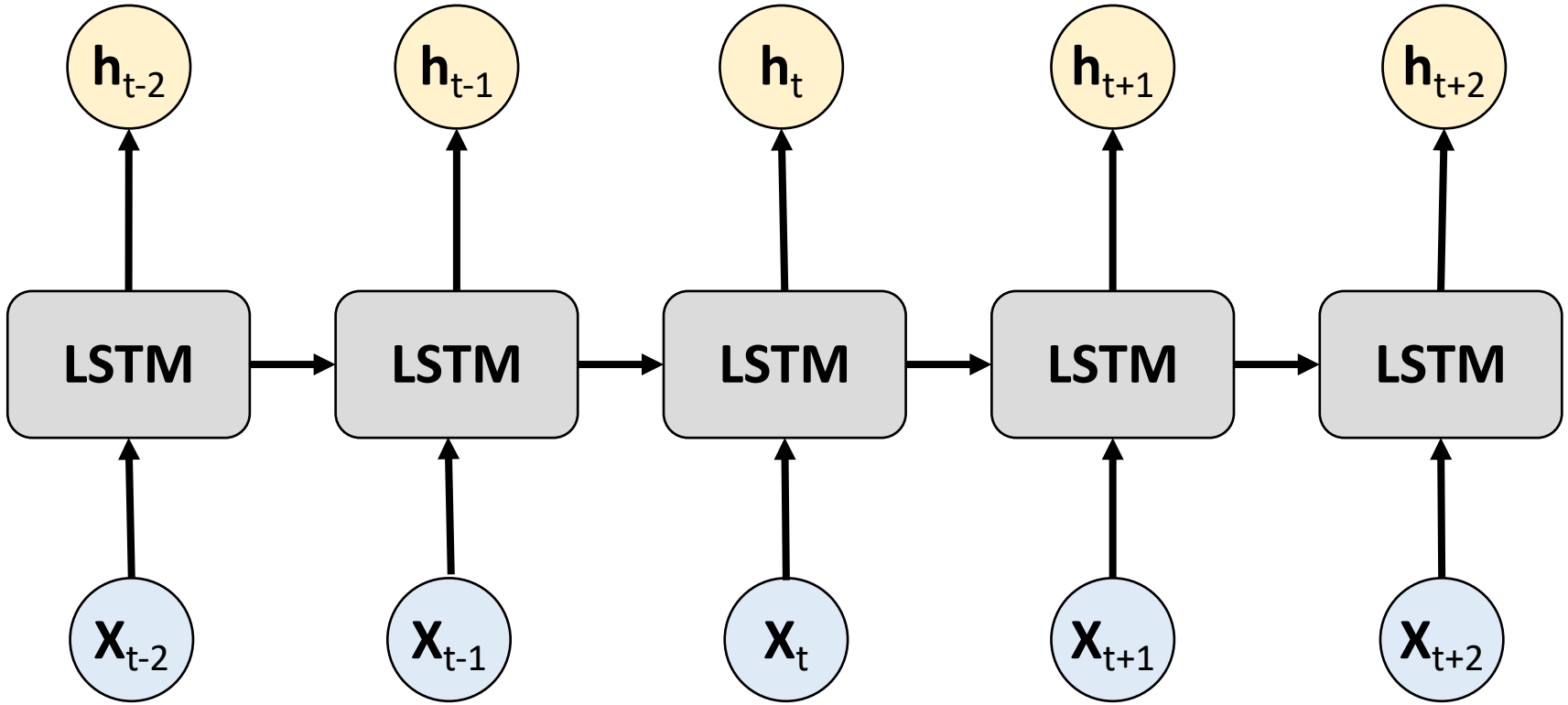


Time Series Data

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



Long Short Term Memory (LSTM) for Time Series Forecasting

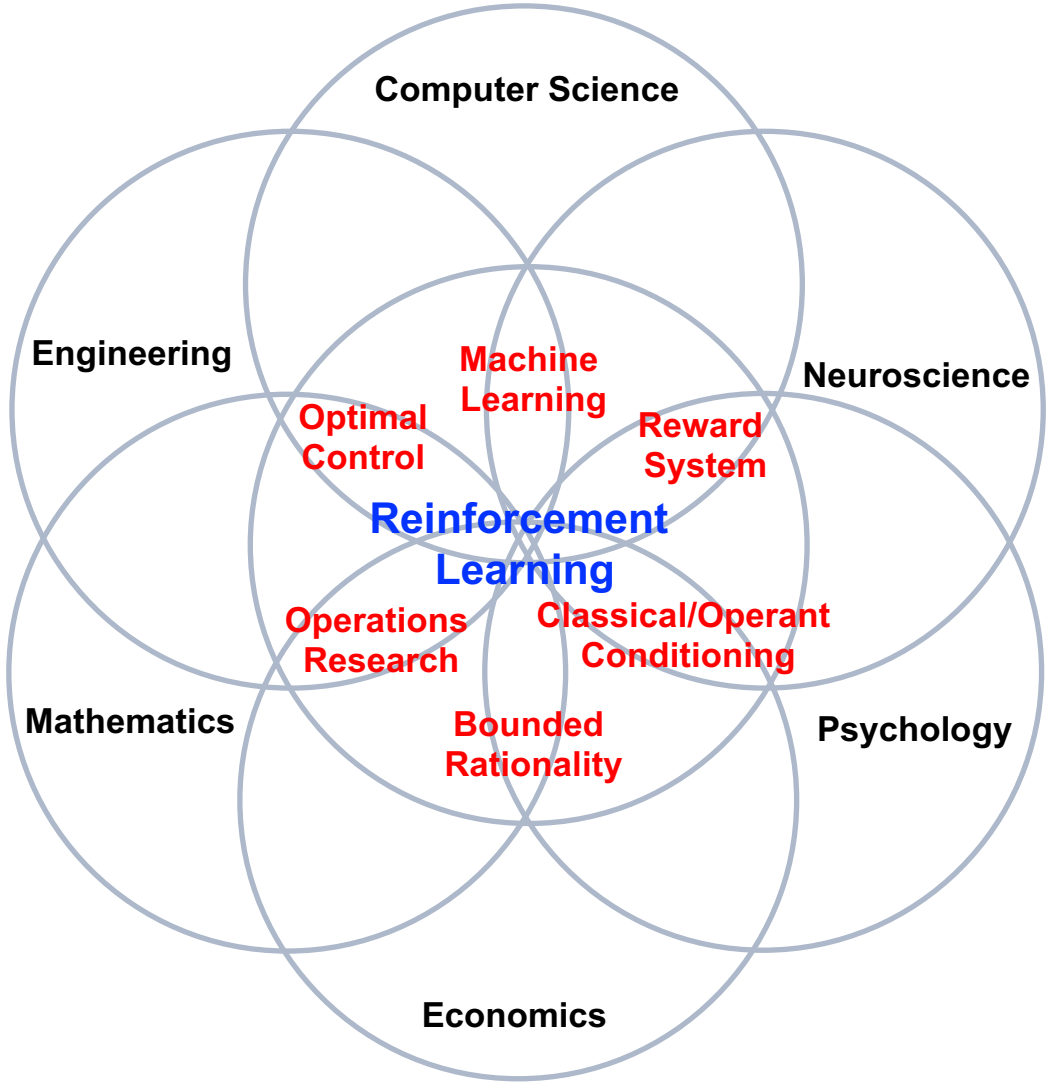


Time Series Data

[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

Reinforcement Learning (RL)

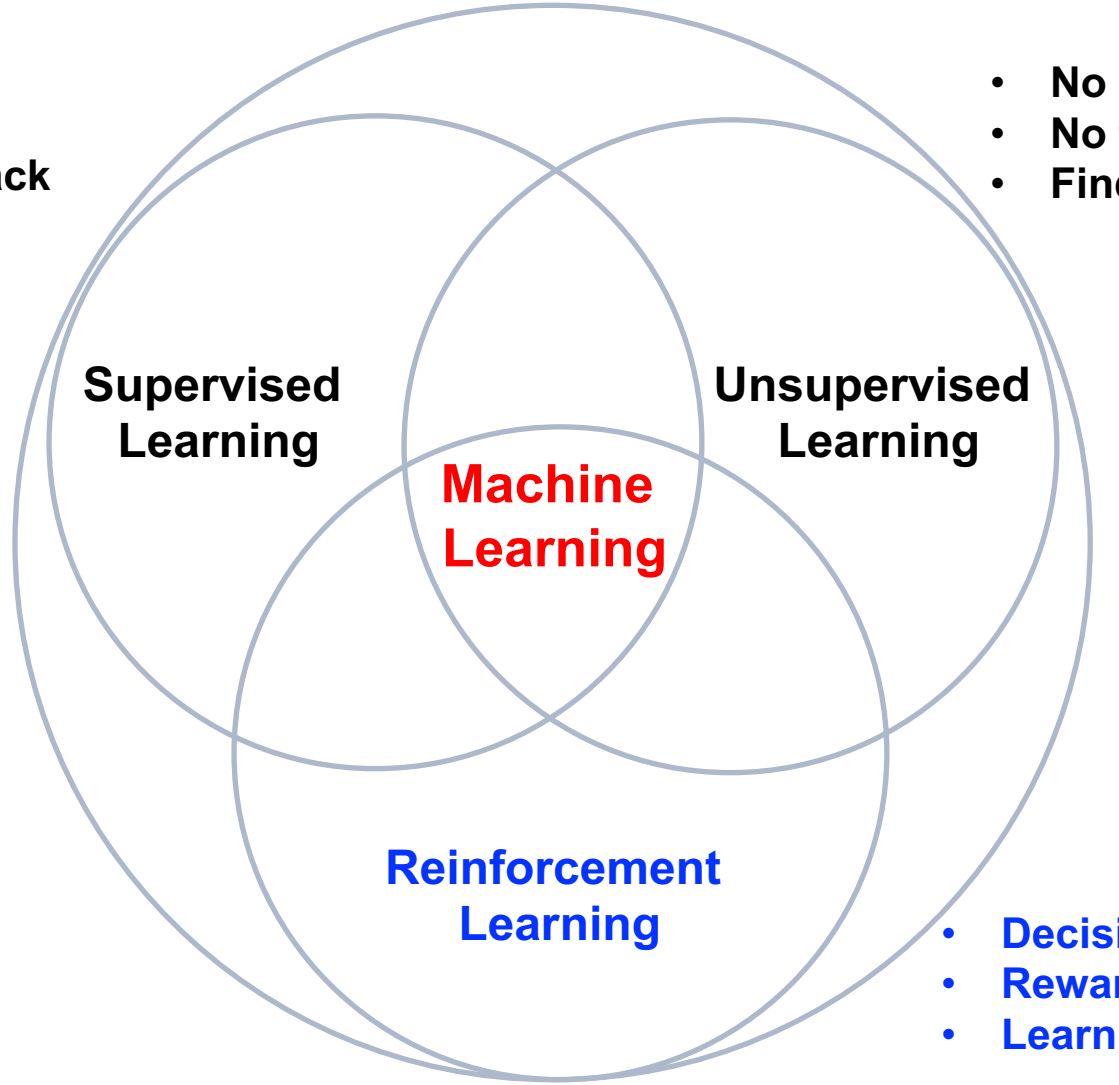


Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

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

- Labeled data
- Direct feedback
- Predict

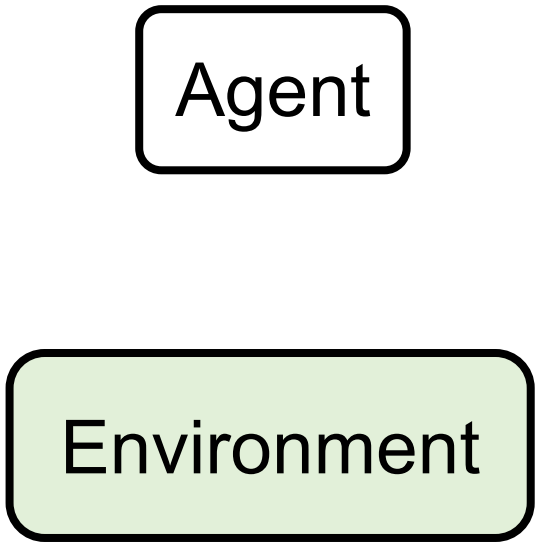
- No Labels
- No feedback
- Find hidden structure



- Decision process
- Reward system
- Learn series of actions

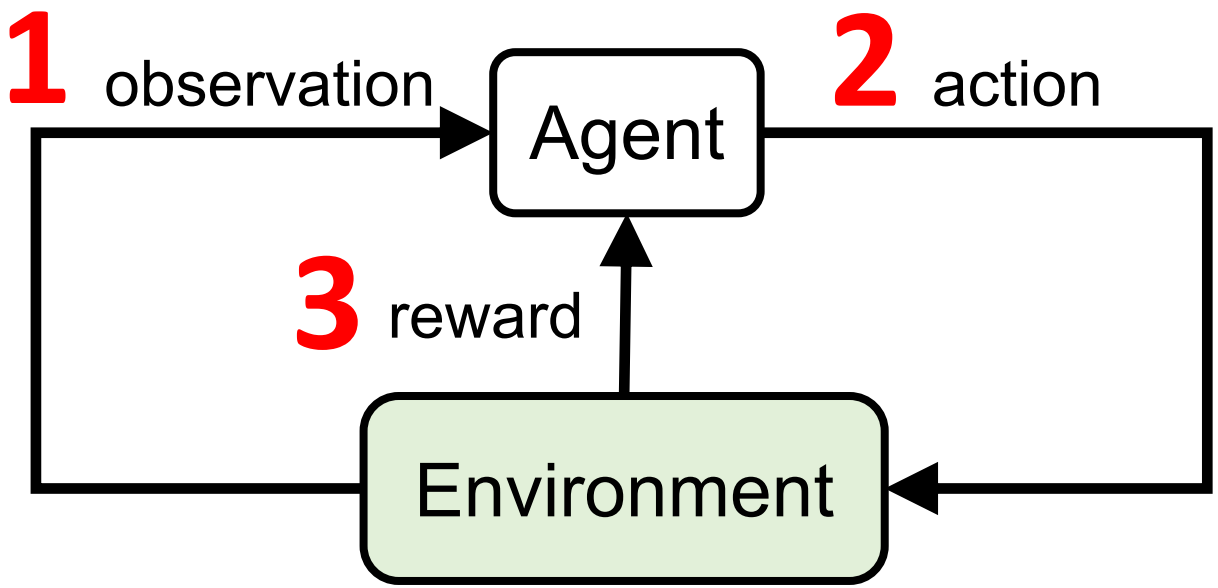
Source: David Silver (2015), Introduction to reinforcement learning, https://www.youtube.com/playlist?list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ

Reinforcement Learning (DL)



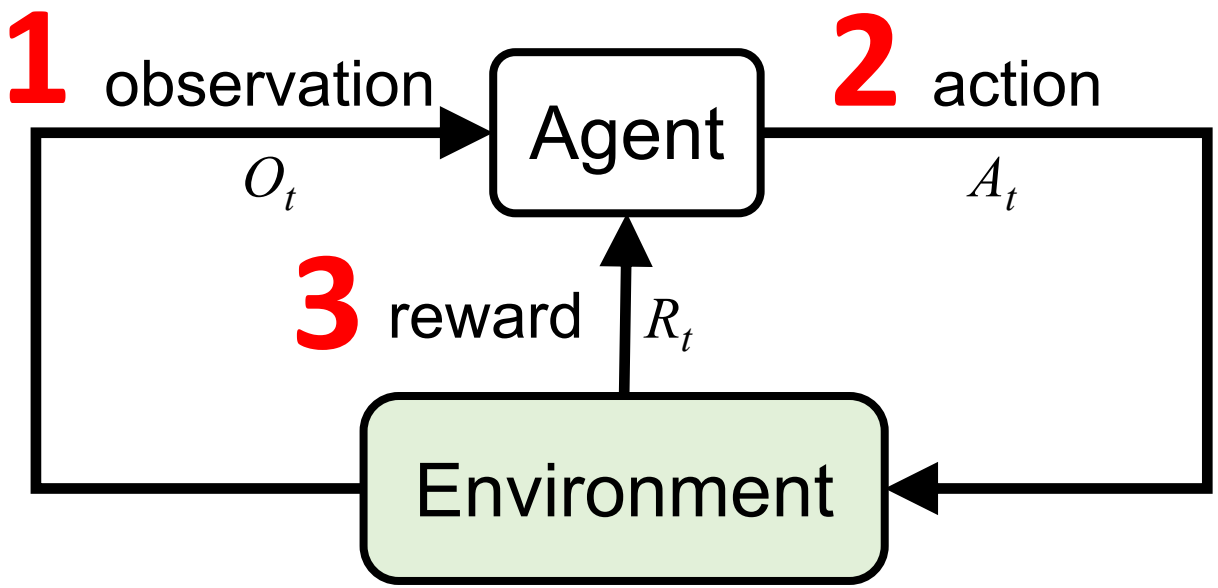
Source: Richard S. Sutton & Andrew G. Barto (2018), Reinforcement Learning: An Introduction, 2nd Edition, A Bradford Book.

Reinforcement Learning (DL)



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

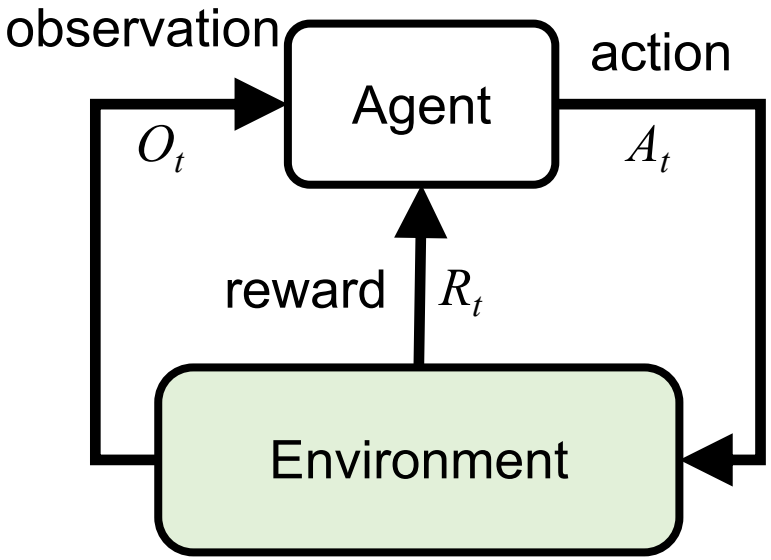
Reinforcement Learning (DL)



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

Agent and Environment

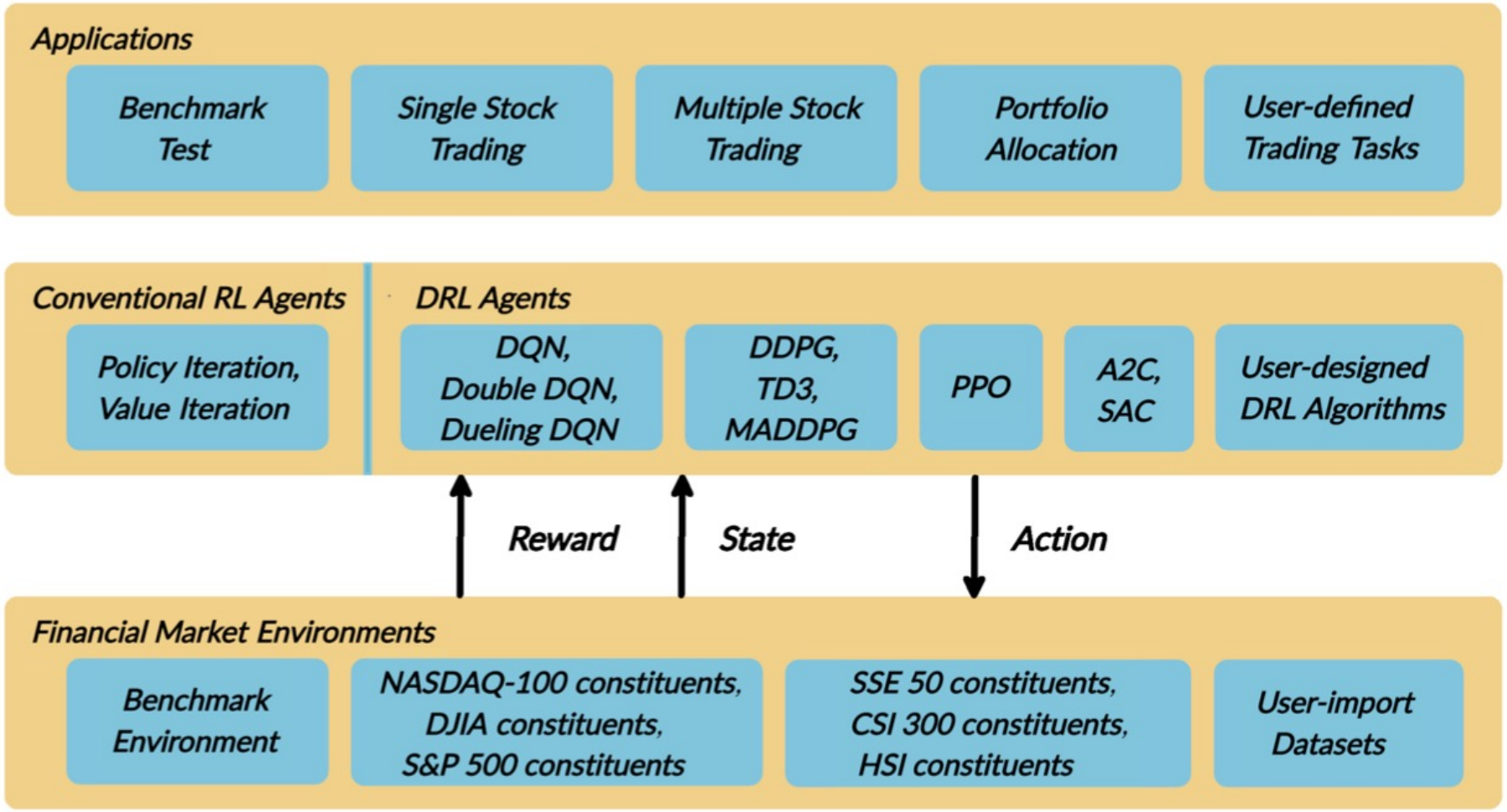
- At each step *t* the agent:
  - Executes action A_t
  - Receives observation O_t
  - Receives scalar reward R_t
- The environment:
  - Receives action A_t
  - Emits observation O_{t+1}
  - Emits scalar reward R_{t+1}
- t increments at env. step



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

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

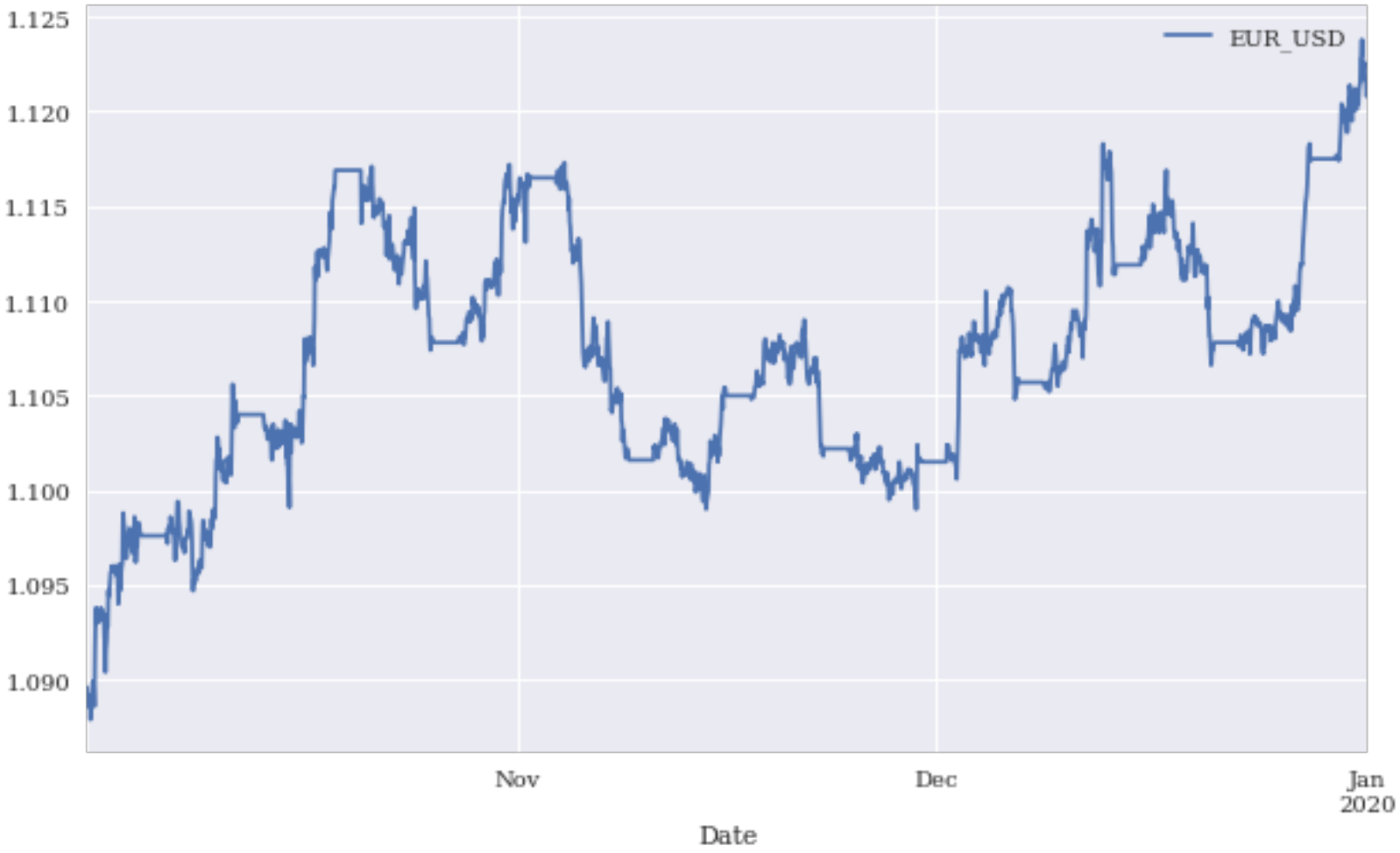
Algorithms	Input	Output	Type	State-action spaces support	Finance use cases support	Features and Improvements	Advantages
DQN	States	Q-value	Value based	Discrete only	Single stock trading	Target network, experience replay	Simple and easy to use
Double DQN	States	Q-value	Value based	Discrete only	Single 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).

```
import os
import numpy as np
import pandas as pd
from pylab import plt, mpl
plt.style.use('seaborn')
mpl.rcParams['savefig.dpi'] = 300
mpl.rcParams['font.family'] = 'serif'
pd.set_option('precision', 4)
np.set_printoptions(suppress=True, precision=4)
os.environ['PYTHONHASHSEED'] = '0'
```

```
url = 'http://hilpisch.com/aiif_eikon_id_eur_usd.csv'
symbol = 'EUR_USD'
raw = pd.read_csv(url, index_col=0, parse_dates=True)
raw.head()
```

Mid-closing prices for EUR/USD (intraday)



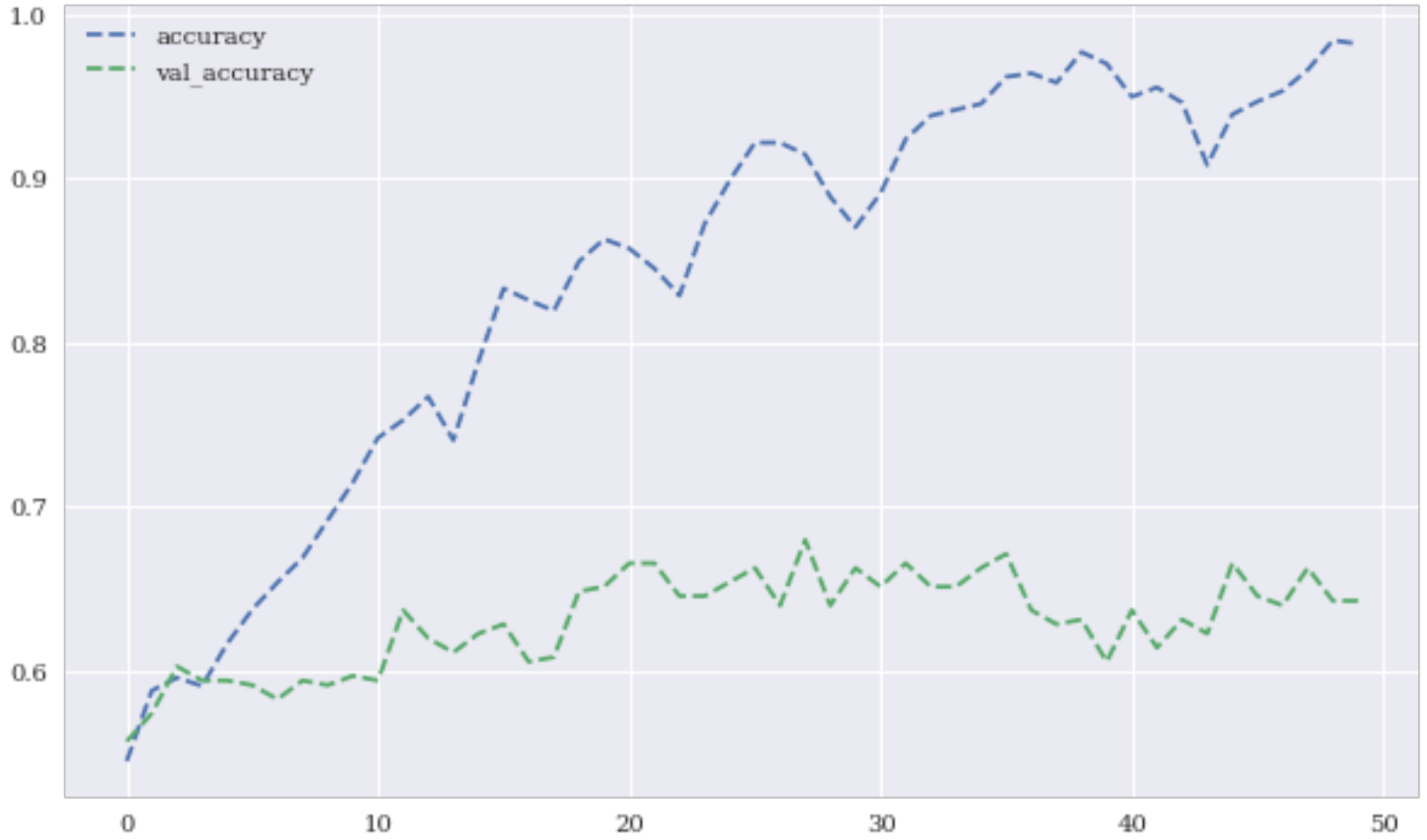
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

```
optimizer = Adam(lr=0.001)

def create_model(hl=1, hu=128, optimizer=optimizer):
    model = Sequential()
    model.add(Dense(hu, input_dim=len(cols),
                    activation='relu'))
    for _ in range(hl):
        model.add(Dense(hu, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy',
                  optimizer=optimizer,
                  metrics=['accuracy'])
    return model

set_seeds()
model = create_model(hl=1, hu=128)
model.summary()
```

Training and validation accuracy values

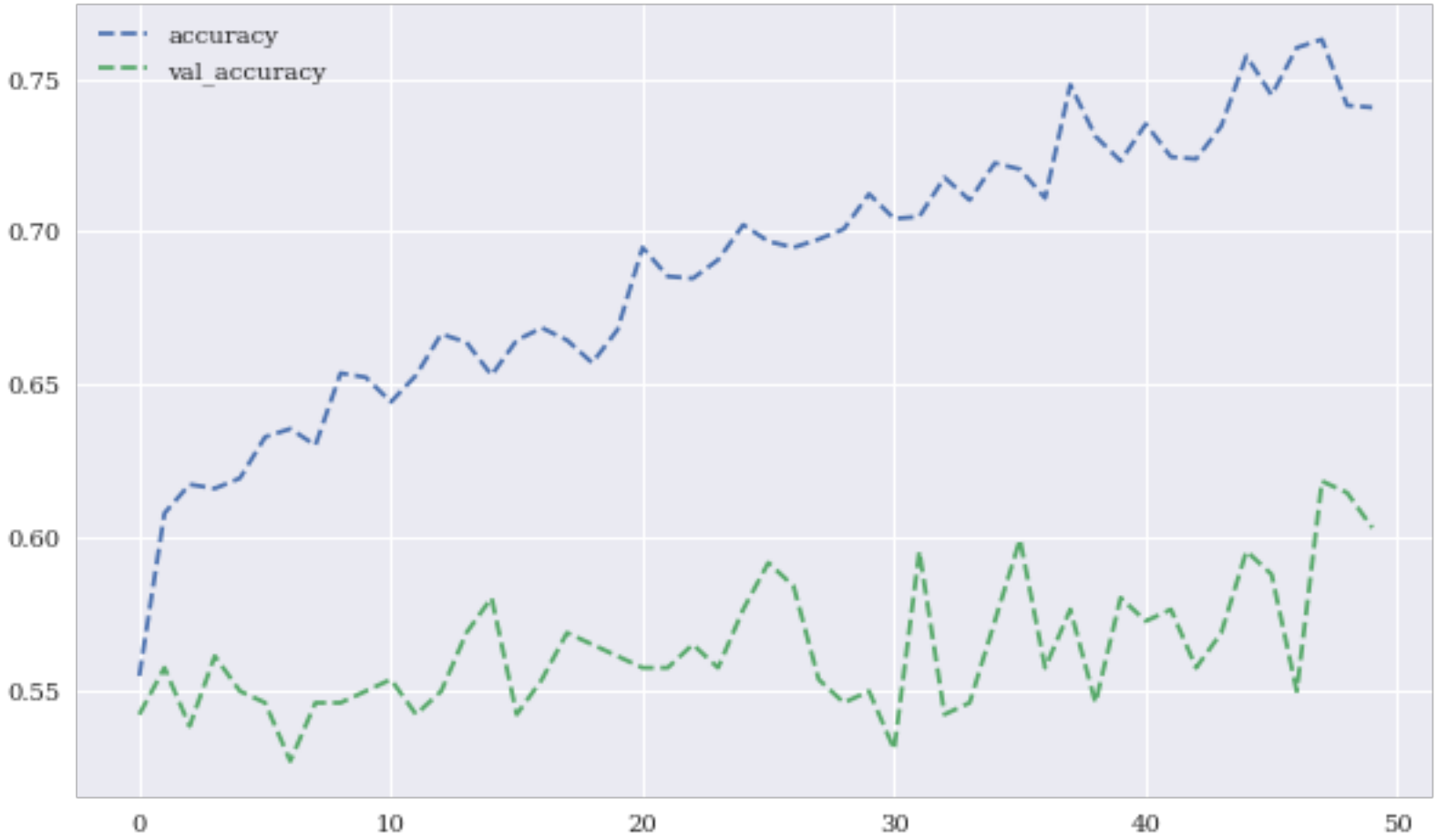
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Training and validation accuracy values (normalized features data)



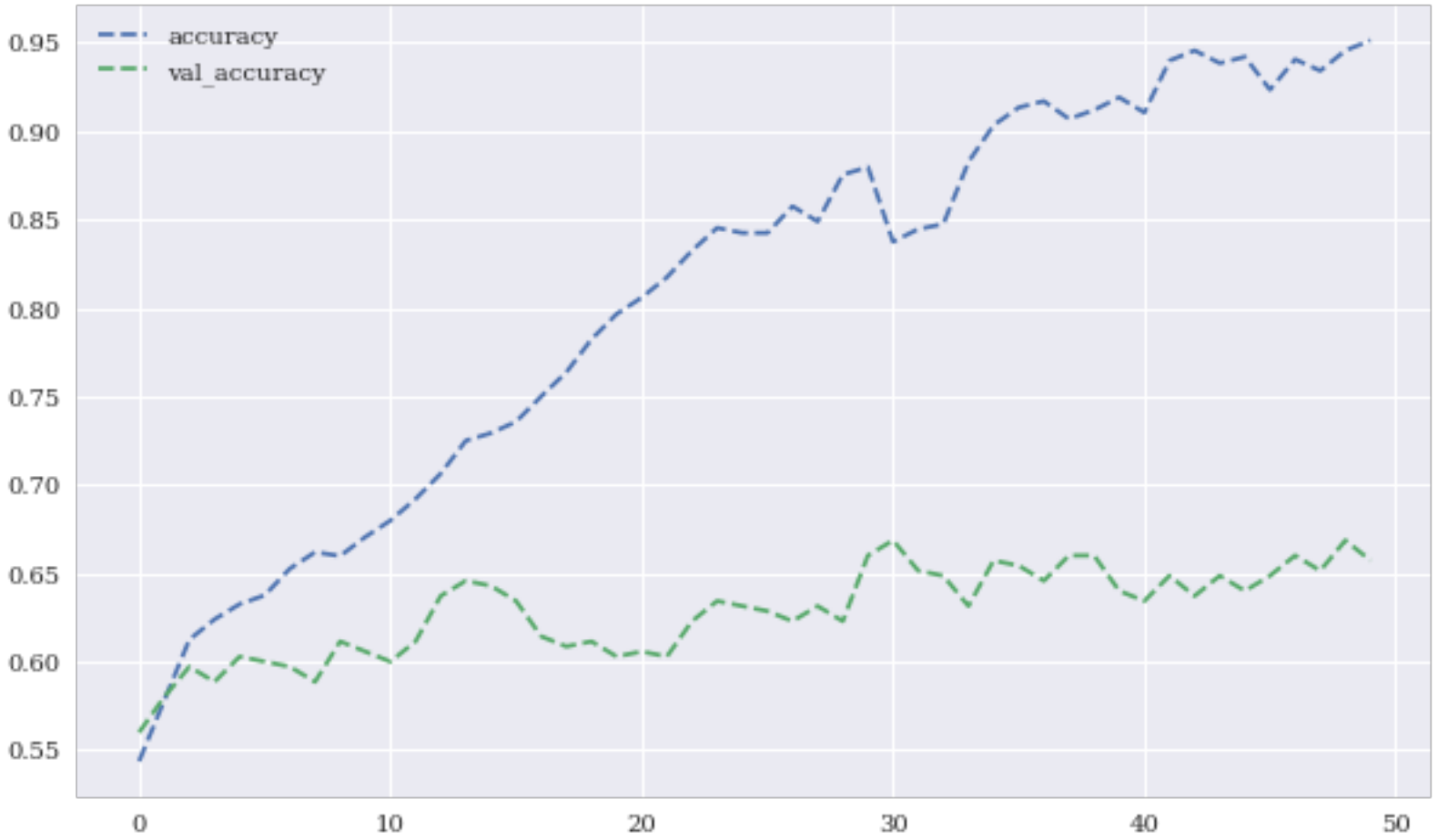
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Training and validation accuracy values (with dropout)



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Training and validation accuracy values (with regularization)



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

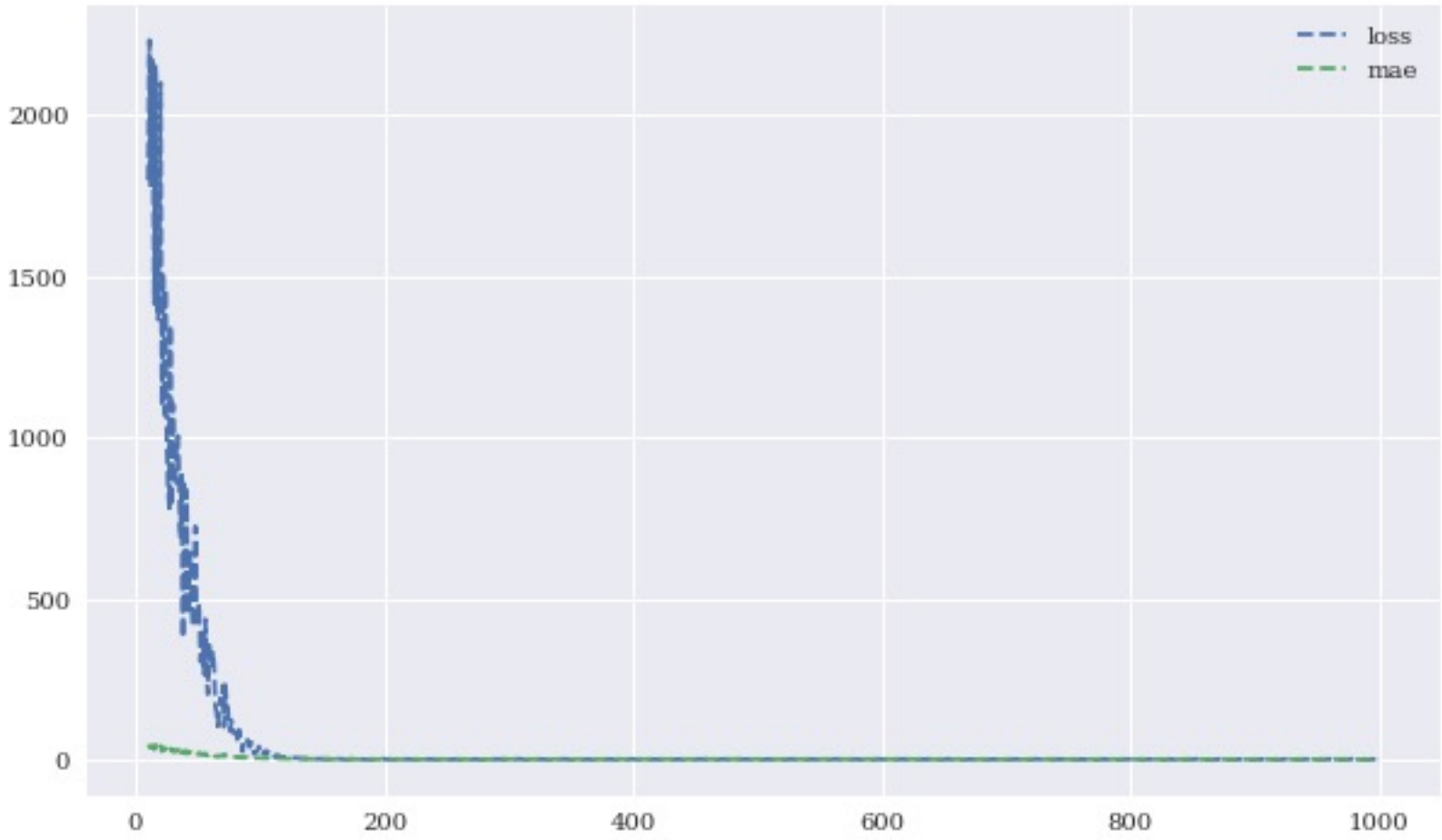
Training and validation accuracy values (with dropout and regularization)

Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

```
from keras.models import Sequential
from keras.layers import SimpleRNN, LSTM, Dense

model = Sequential()
model.add(SimpleRNN(100, activation='relu',
          input_shape=(lags, 1)))
model.add(Dense(1, activation='linear'))
model.compile(optimizer='adagrad', loss='mse',
              metrics=['mae'])

model.summary()
```

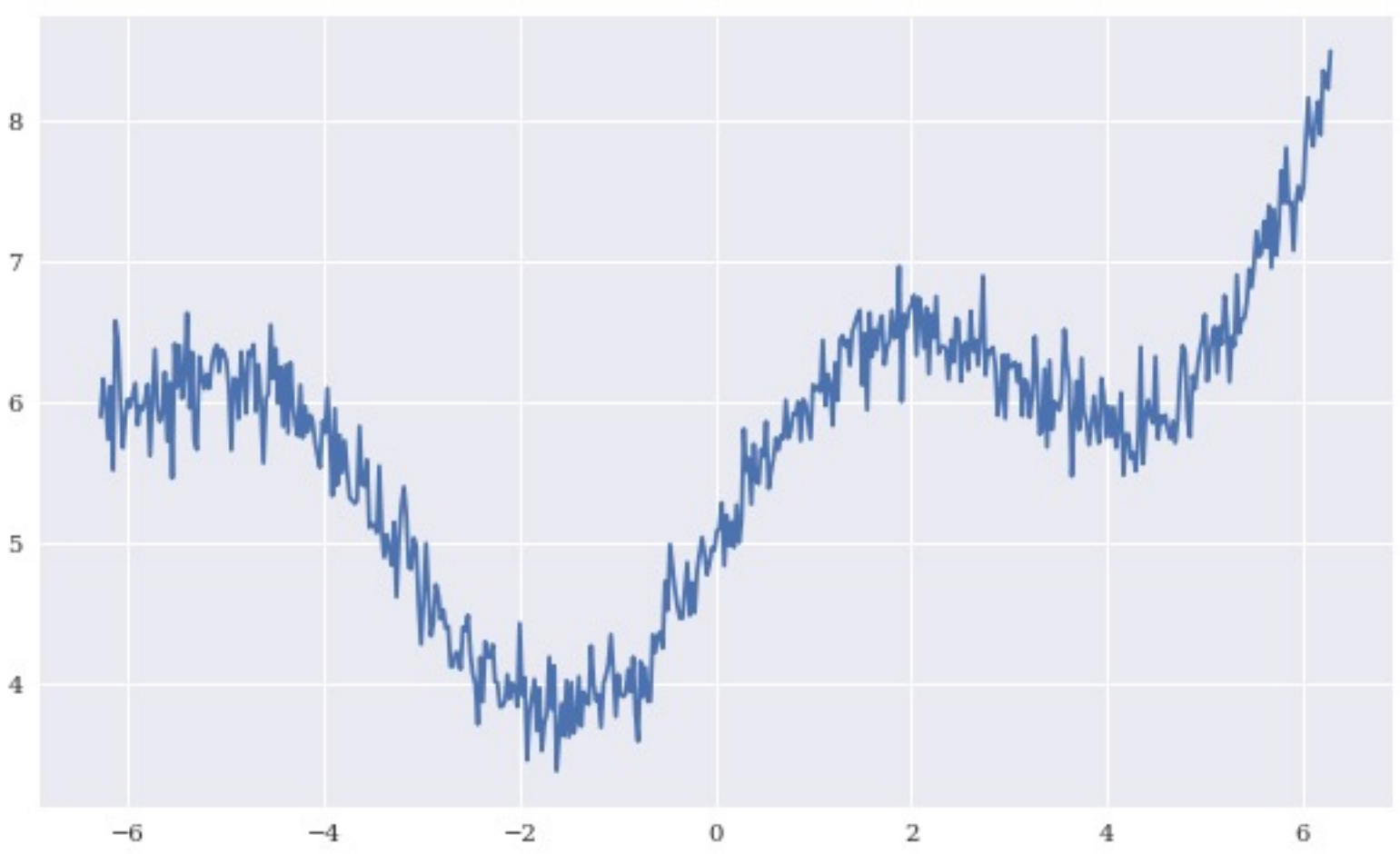
```
model.fit(g, epochs=1000, steps_per_epoch=5, verbose=False)
```

Performance metrics during RNN training



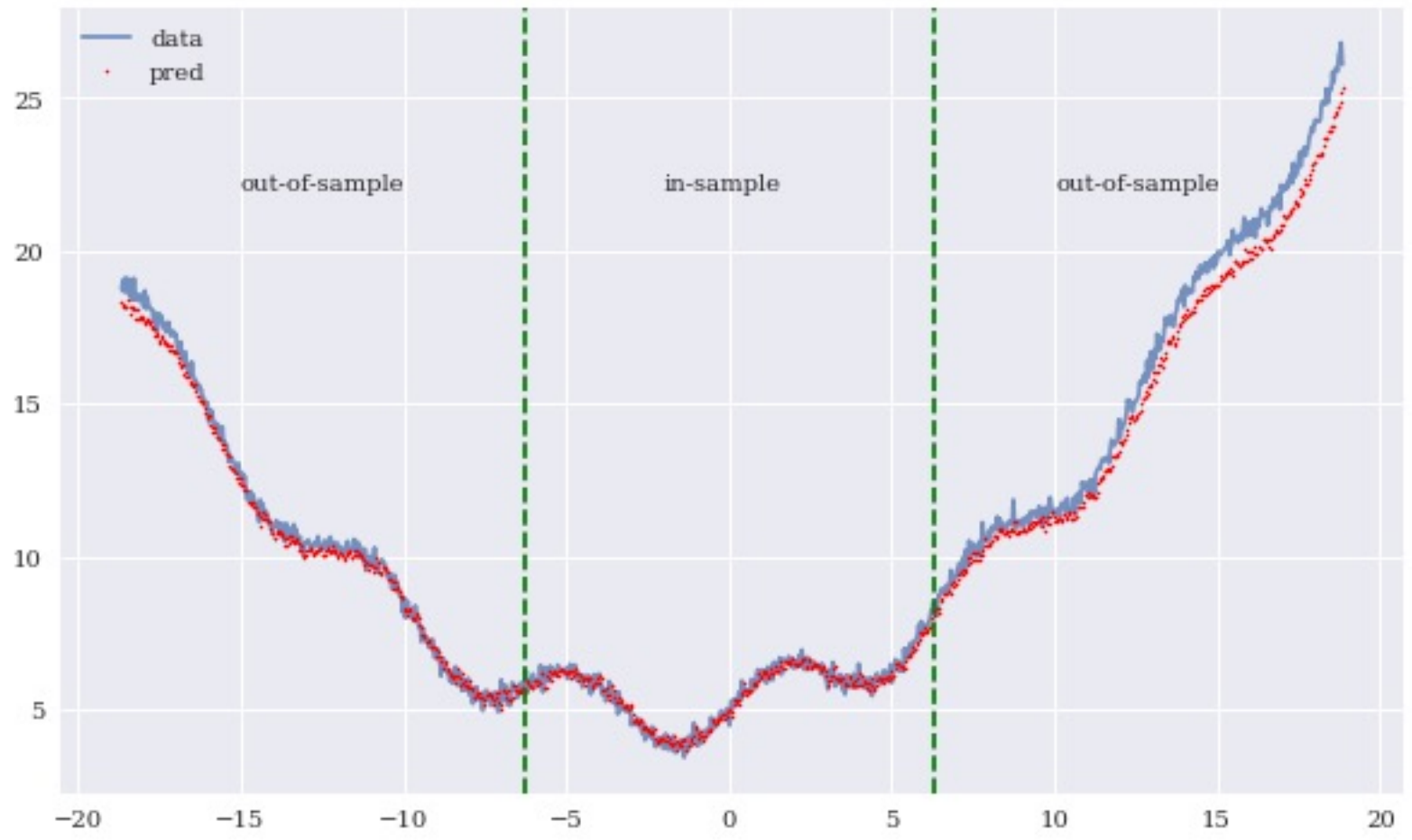
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Sample sequence data



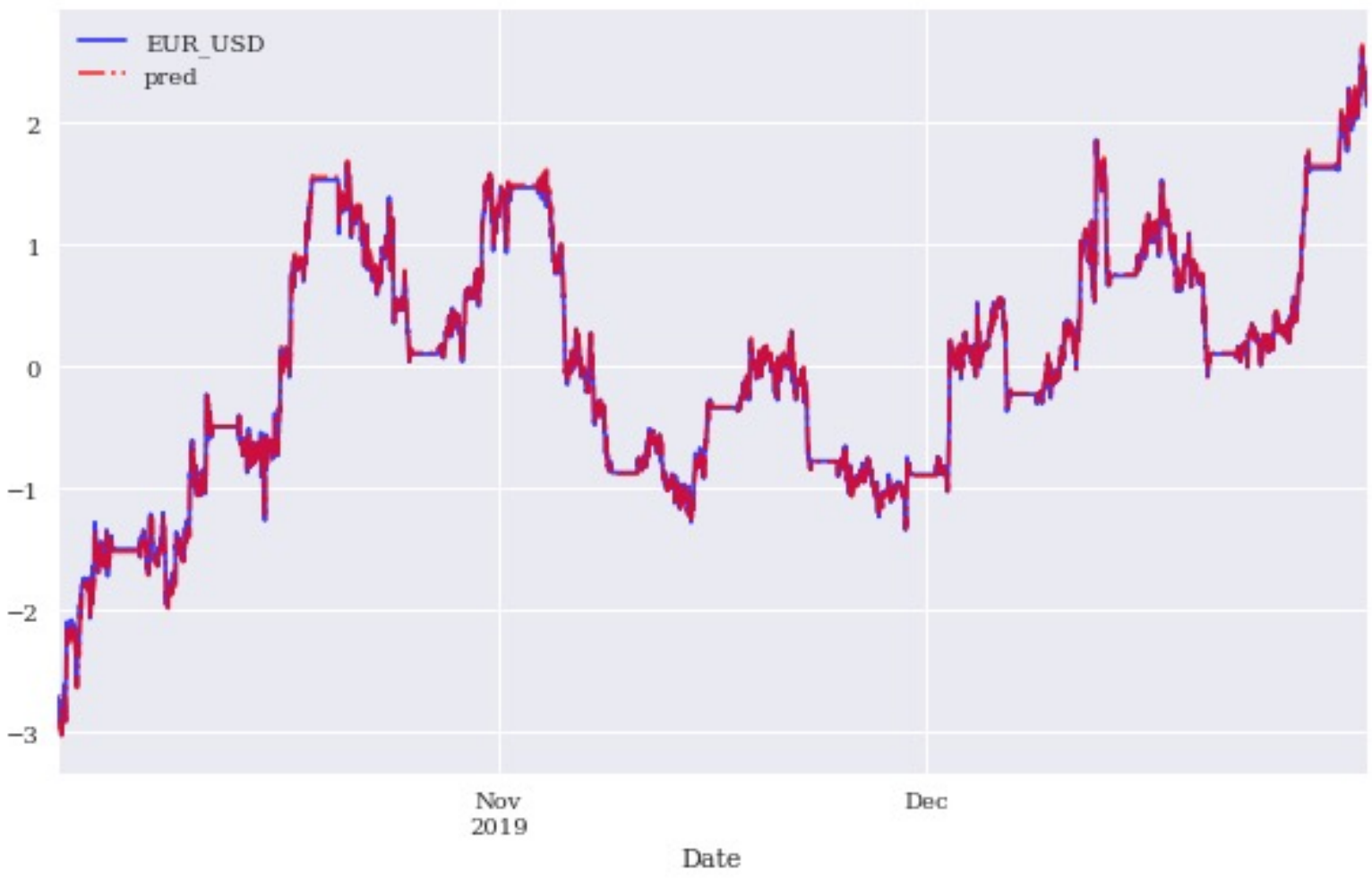
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

in-sample and out-of-sample predictions of the RNN



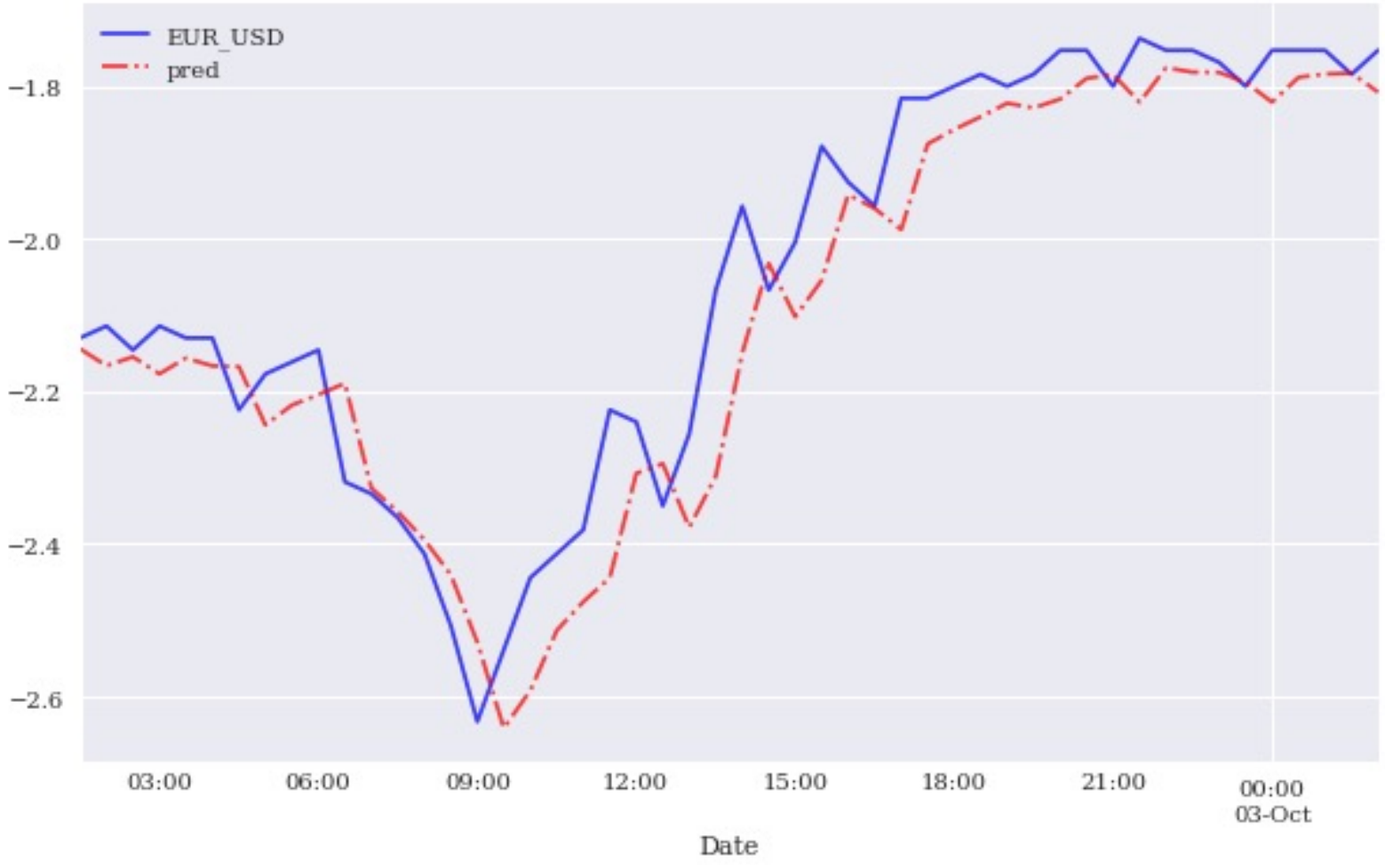
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

In-sample prediction for financial price series by the RNN (whole data set)



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

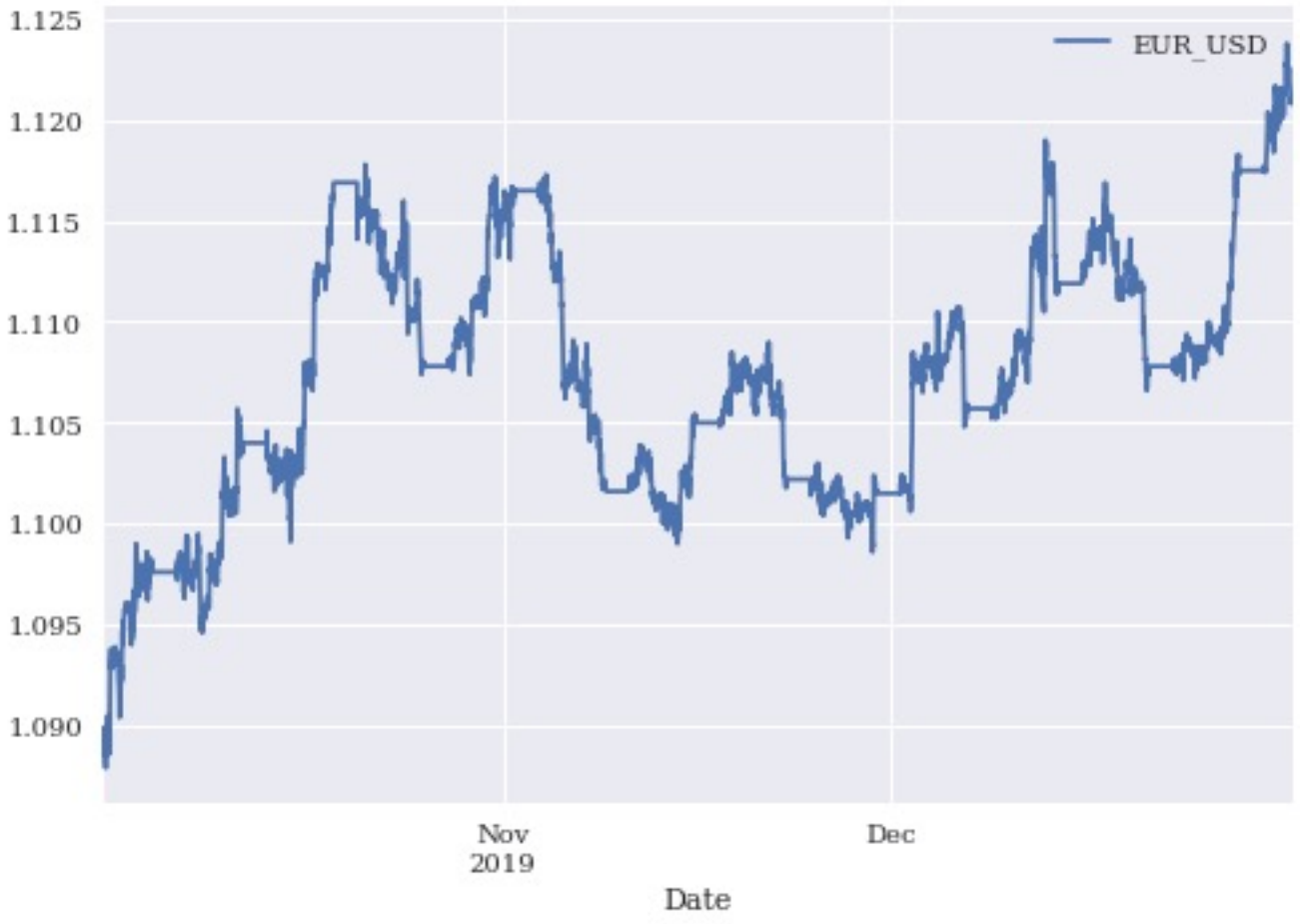
In-sample prediction for financial price series by the RNN (data sub-set)



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Financial Price Series

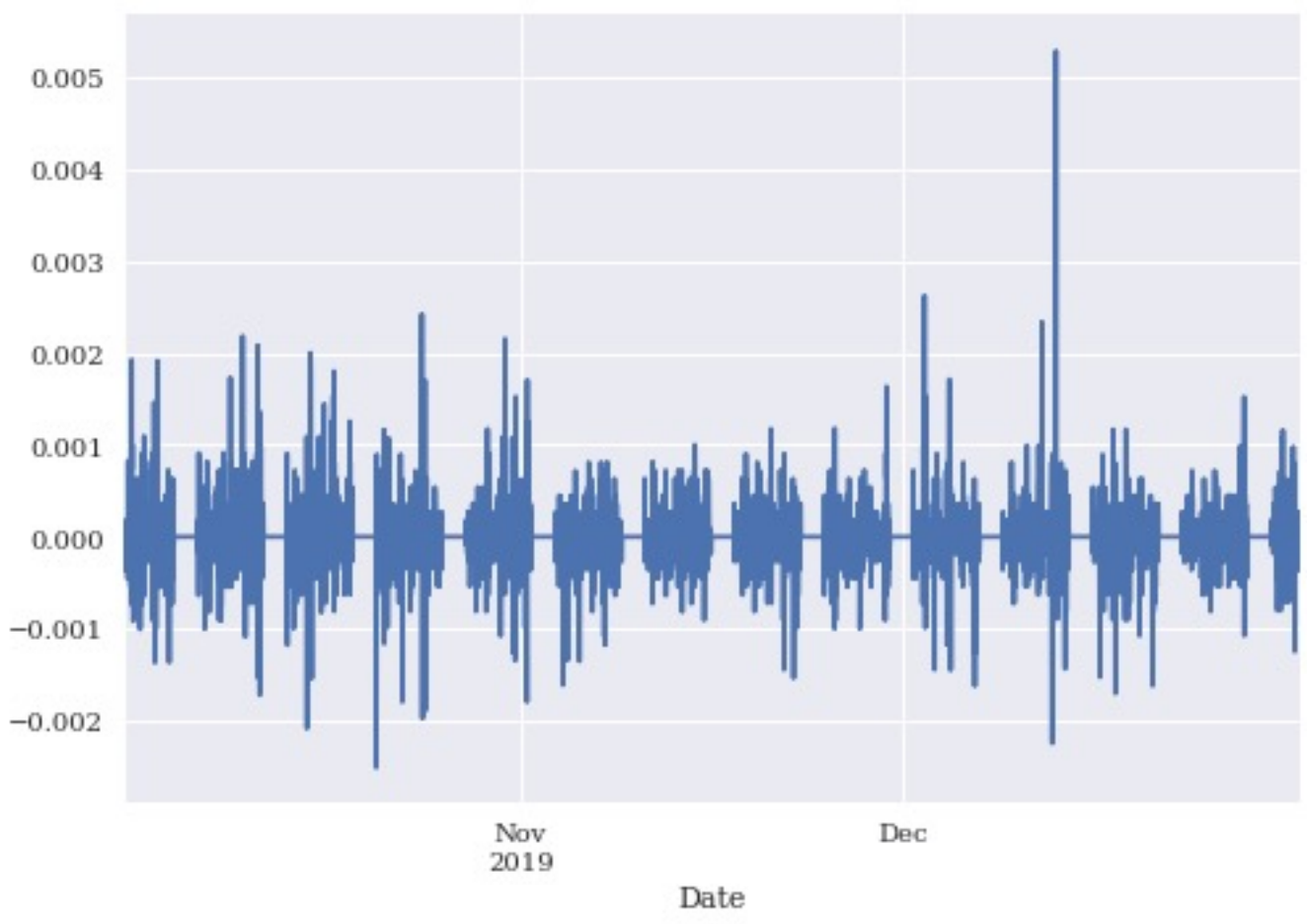
```
data = generate_data()
data.plot()
```



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Financial Return Series

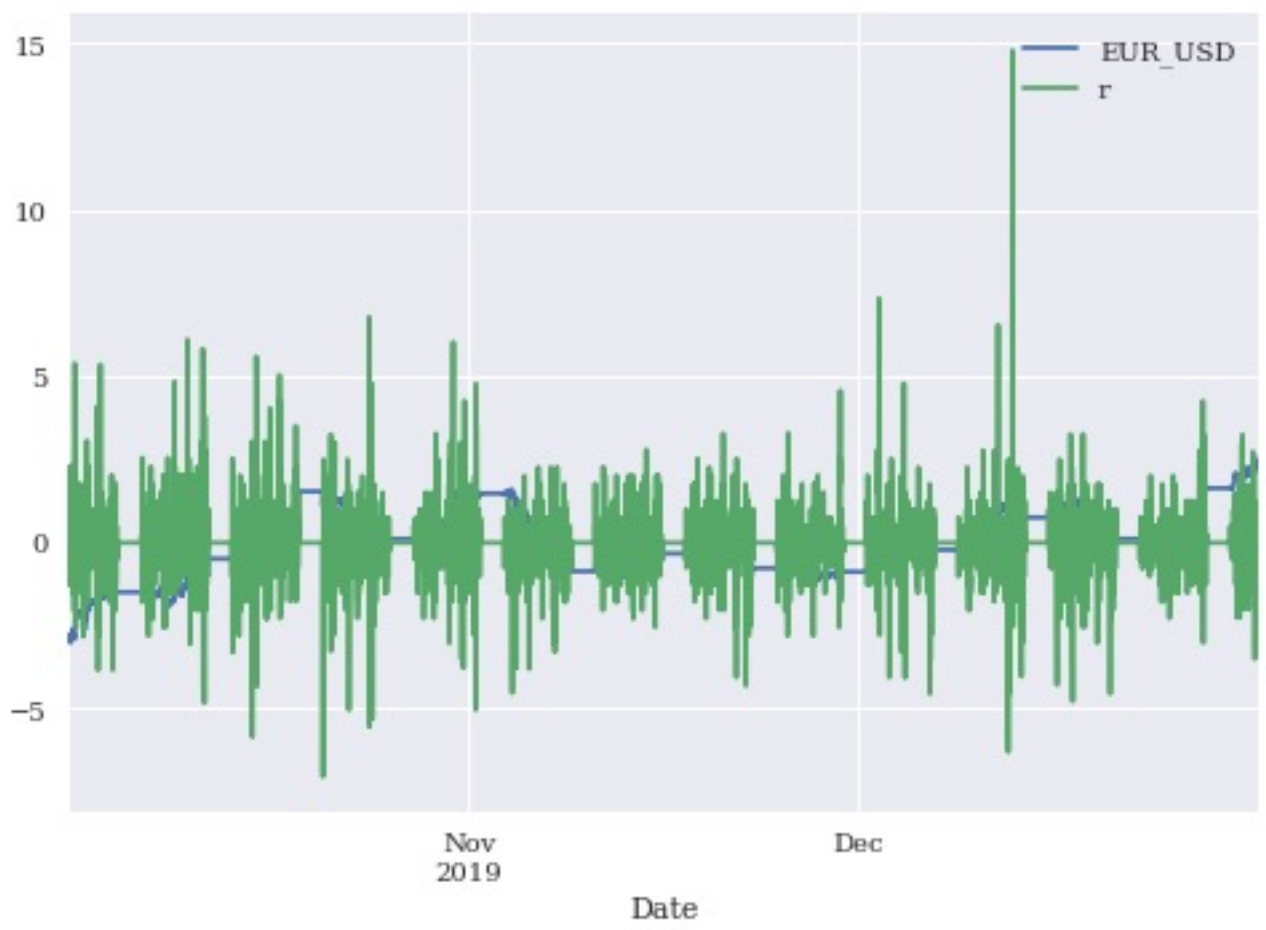
```
data['r'] = np.log(data / data.shift(1))
data['r'].plot()
```



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

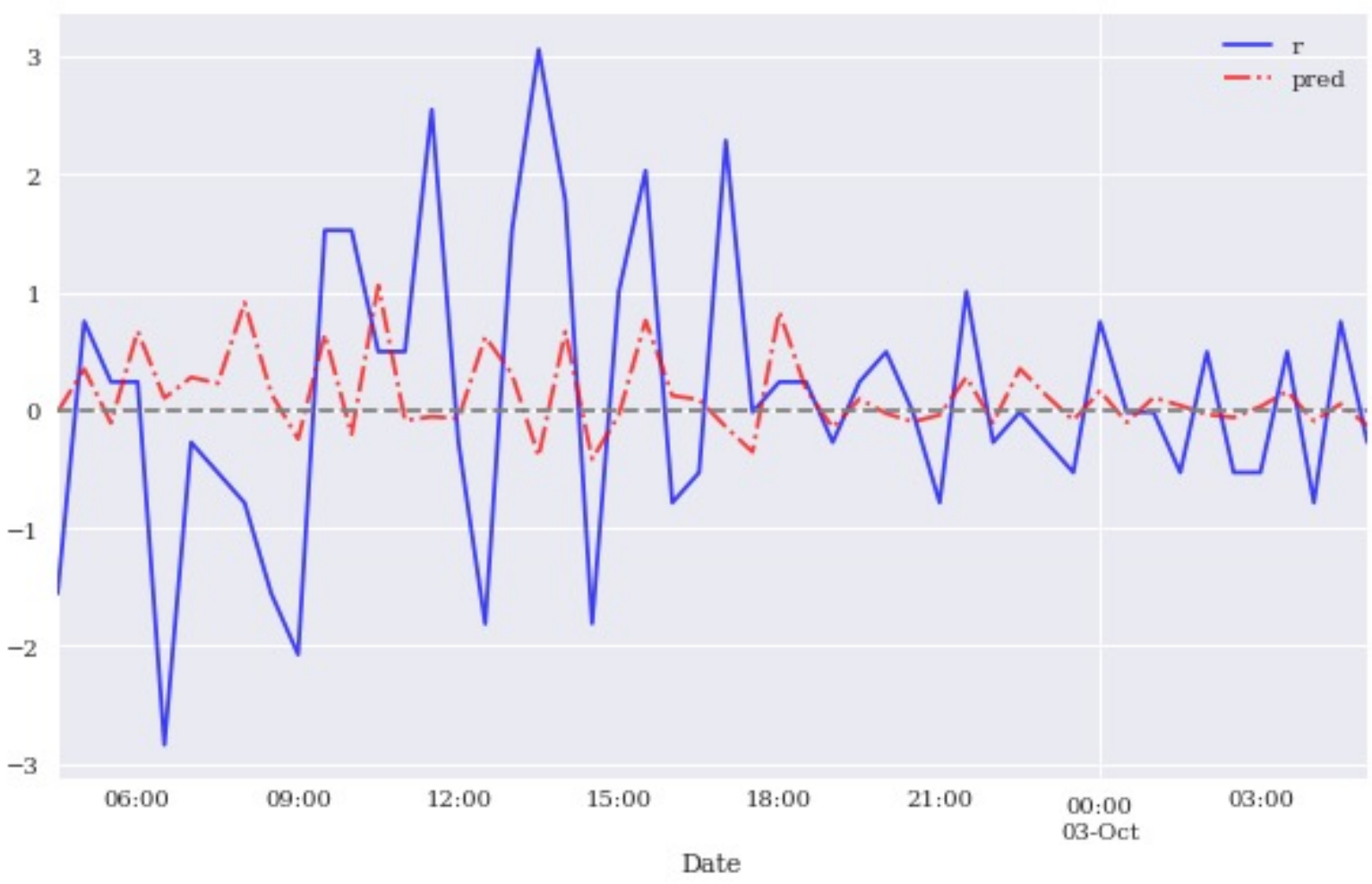
Financial Price and Return Normalization Series

```
data.dropna(inplace=True)
data = (data - data.mean()) / data.std()
data.plot()
```



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

In-sample prediction for financial return series by the RNN (data sub-set)

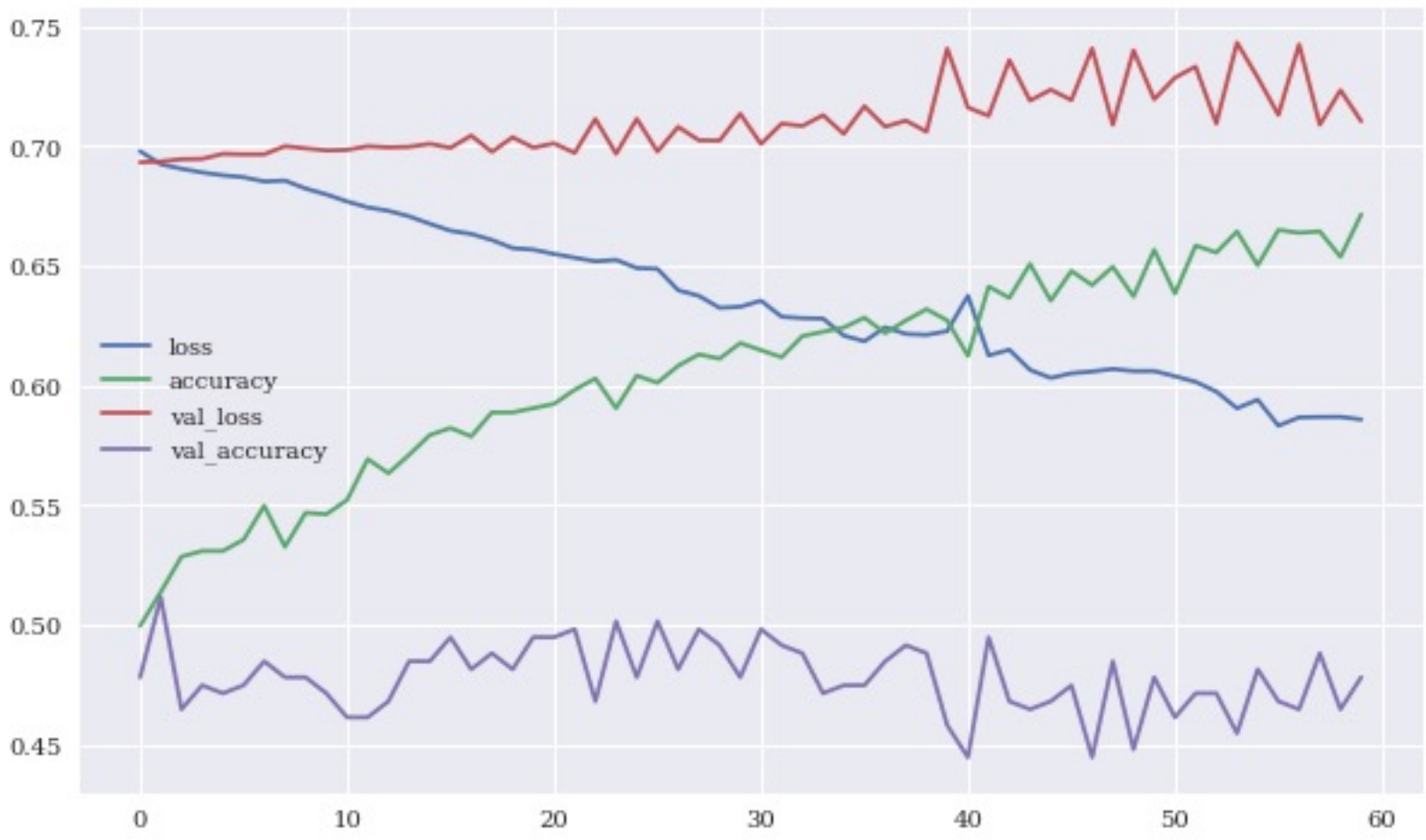


Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

```
model = Sequential()
model.add(Conv1D(filters=96, kernel_size=5,
                 activation='relu',
                 input_shape=(len(cols), 1)))
model.add(Flatten())
model.add(Dense(10, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

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

model.summary()
```

```
model.fit(np.atleast_3d(train[cols]), train['d'],
          epochs=60, batch_size=48, verbose=False,
          validation_split=0.15, shuffle=False)
```

Performance metrics for the training and validation of the CNN



Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Gross performance of passive benchmark investment and CNN strategy (before/after transaction costs)

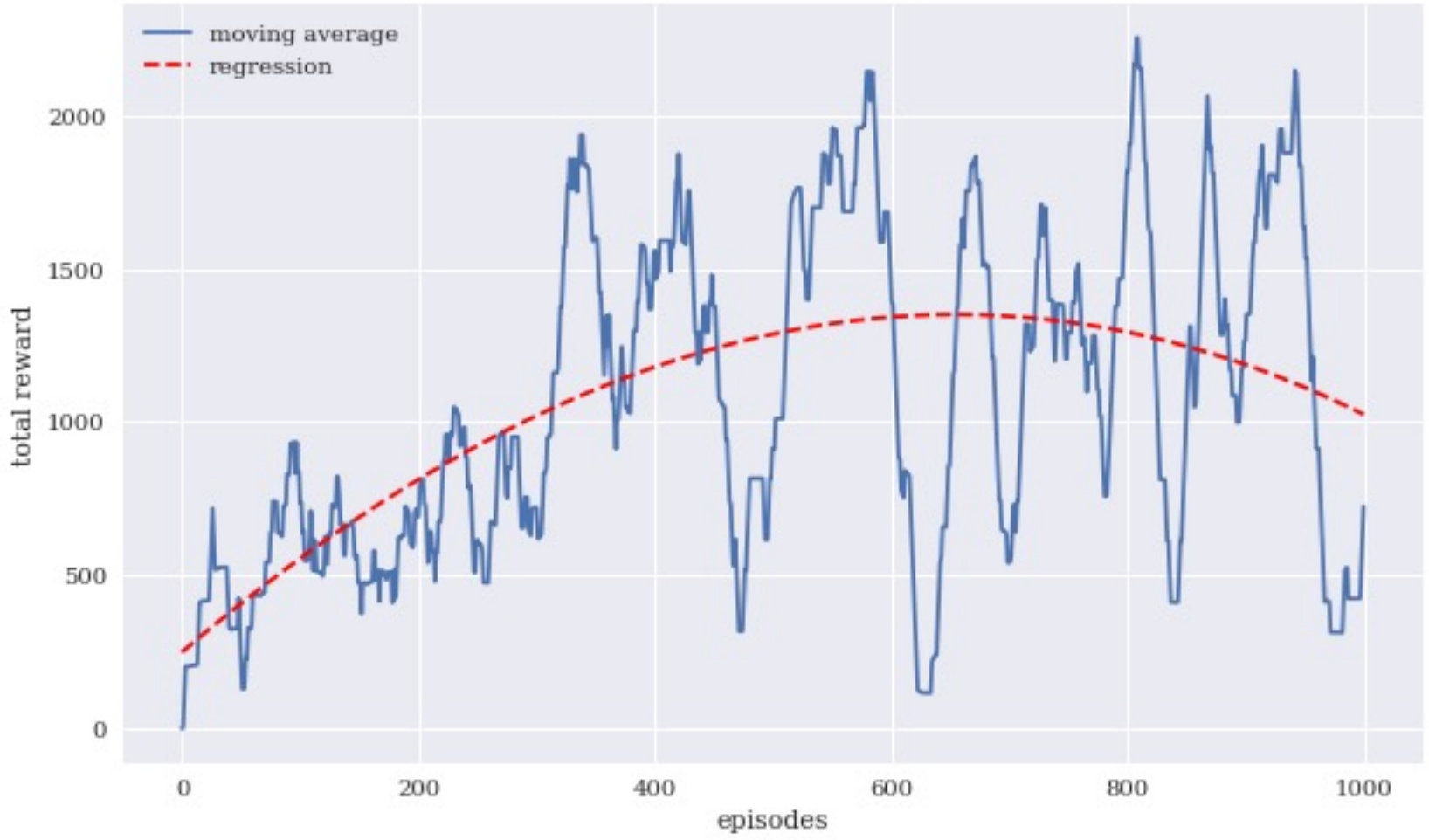
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Reinforcement Learning in Finance

- Simple Learning
- DNN Learning
- Q Learning
- Finance Environment
- Improved Finance Environment
- Improved Financial QL Agent

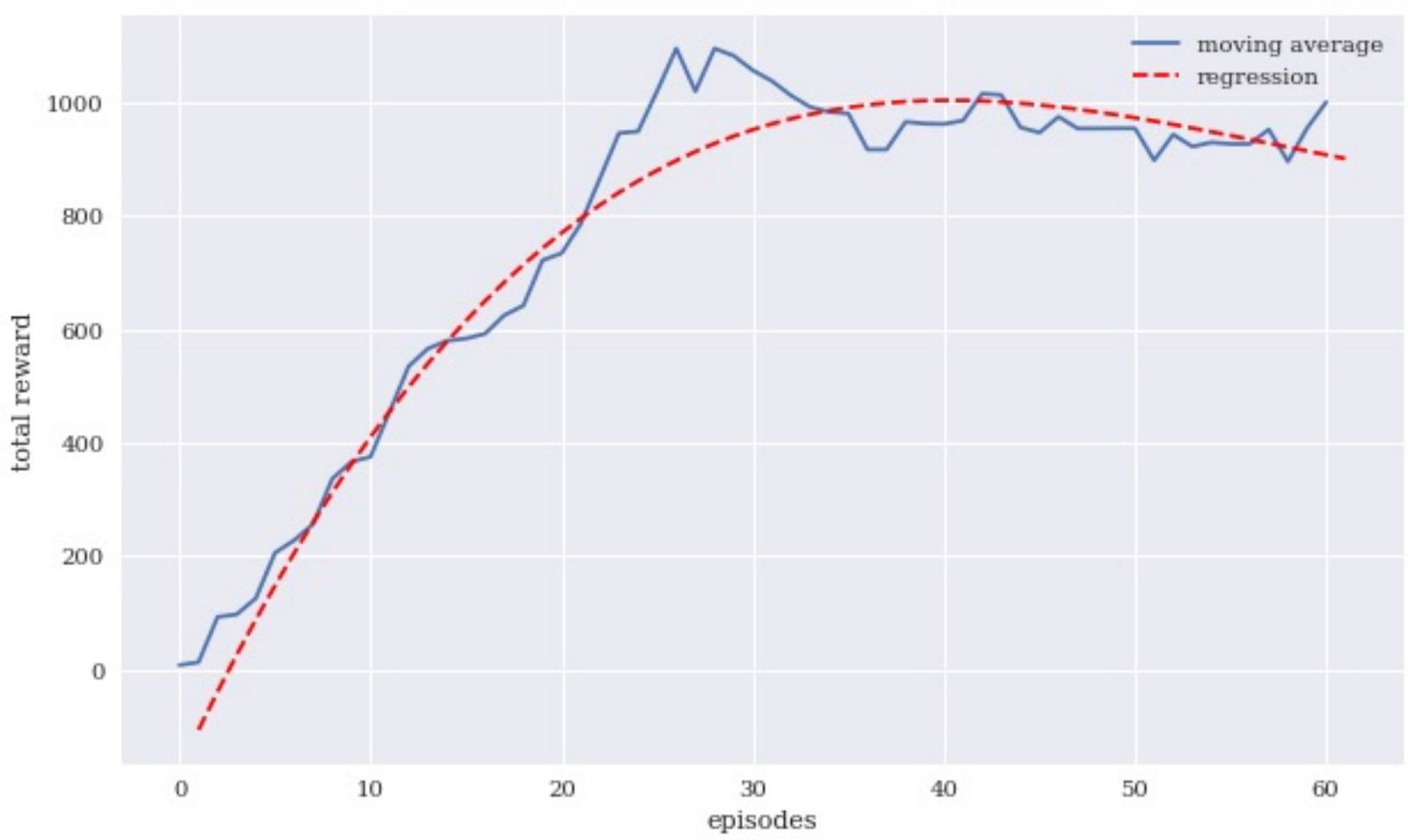
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Average total rewards of DQLAgent for CartPole



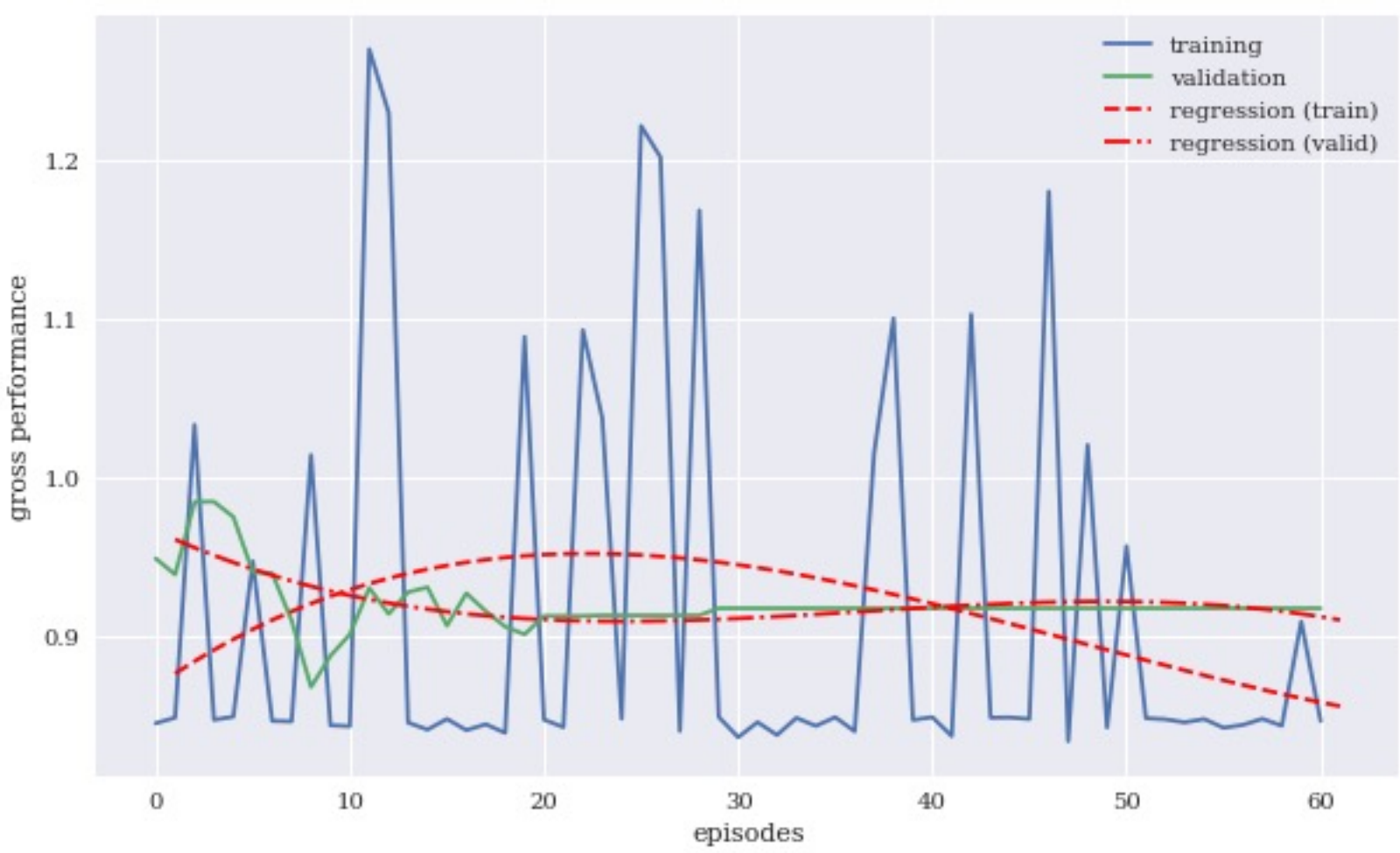
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Average total rewards of DQLAgent for Finance



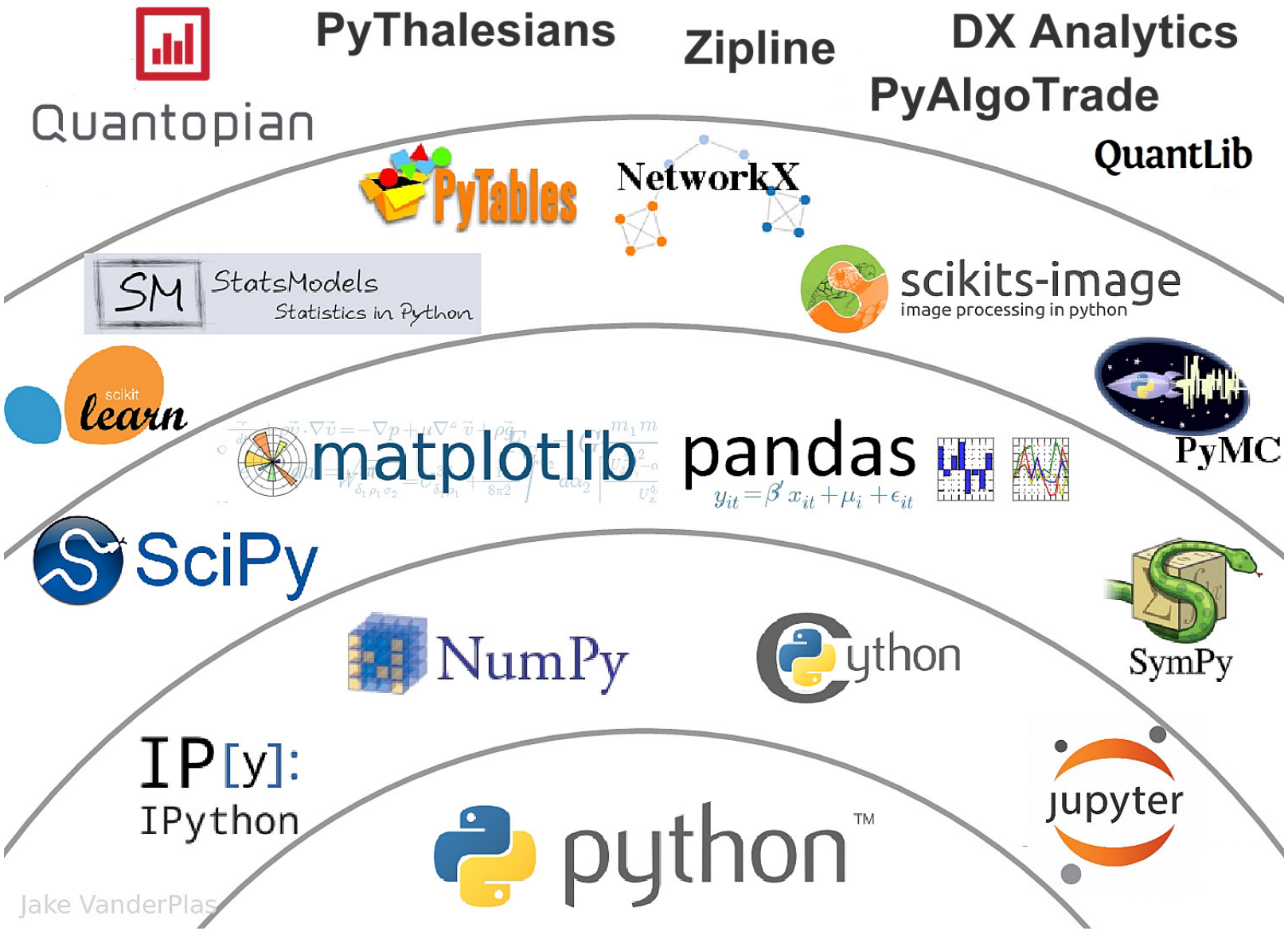
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

Training and validation performance of the FQLAgent per episode



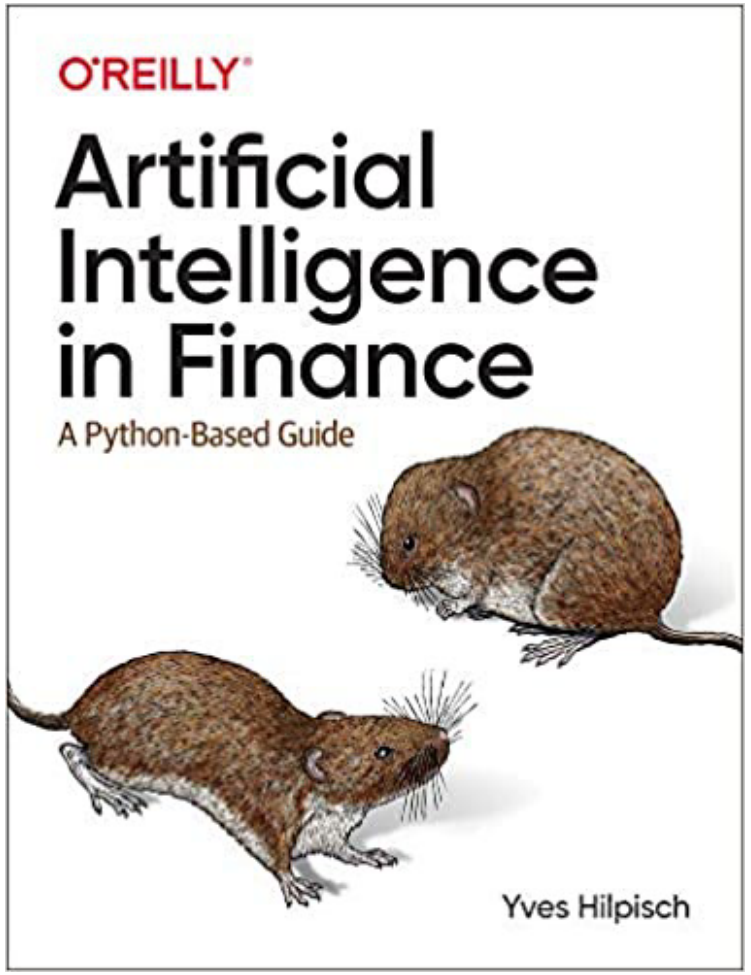
Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.

The Quant Finance PyData Stack



Source: http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb#/5

Yves Hilpisch (2020),
Artificial Intelligence in Finance:
A Python-Based Guide,
O'Reilly



Source: https://www.amazon.com/Artificial-Intelligence-Finance-Python-Based-Guide/dp/1492055433

Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly

yhilpisch / aiif Public

https://github.com/yhilpisch/aiif

Notifications | Star 98 | Fork 77

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights

main | 1 branch | 0 tags | Go to file | Code

yves Code updates for TF 2.3. e334251 on Dec 8, 2020 4 commits

code	Code updates for TF 2.3.	11 months ago
.gitignore	Code updates for TF 2.3.	11 months ago
LICENSE.txt	Code updates.	11 months ago
README.md	Code updates.	11 months ago

README.md

Artificial Intelligence in Finance

About this Repository

This repository provides Python code and Jupyter Notebooks accompanying the **Artificial Intelligence in Finance** book published by O'Reilly.

O'REILLY

About

Jupyter Notebooks and code for the book Artificial Intelligence in Finance (O'Reilly) by Yves Hilpisch.

home.tpq.io/books/aiif

Readme

View license

Releases

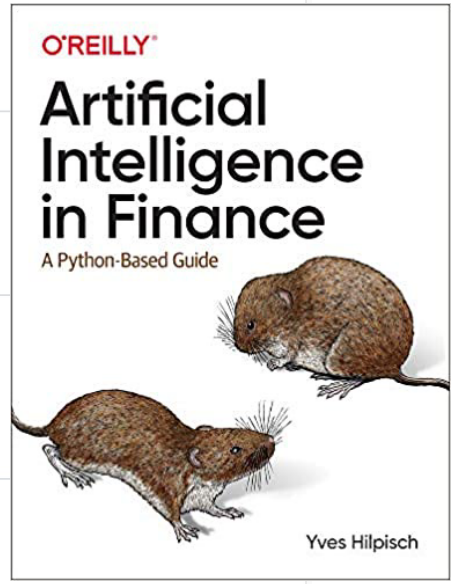
No releases published

Packages

No packages published

Languages

Jupyter Notebook 97.4% Python 2.6%



Source: https://github.com/yhilpisch/aiif

Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly

yhilpisch / aiif Public

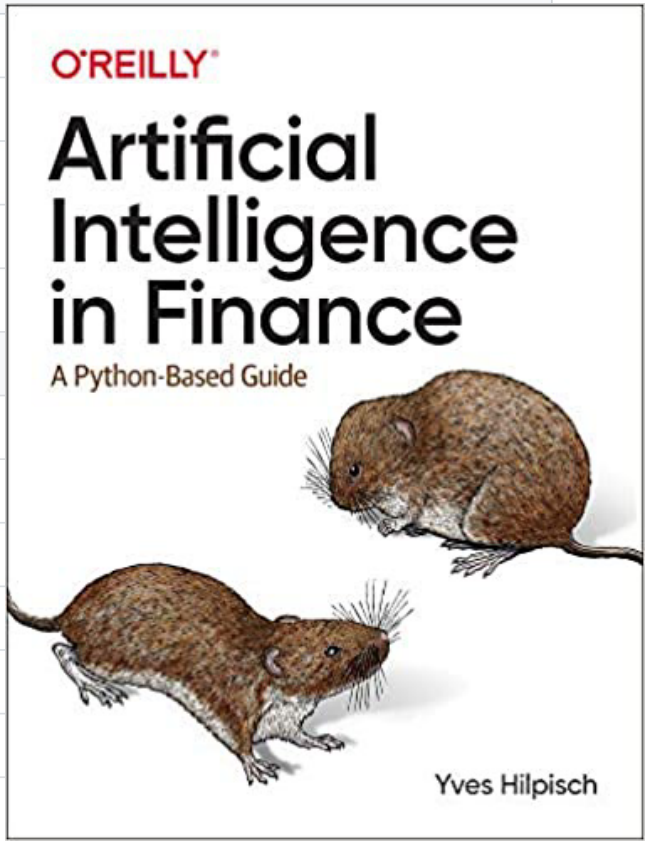
Notifications | Star 98 | Fork 77

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights

main | aiif / code / https://github.com/yhilpisch/aiif/tree/main/code | Go to file

yves Code updates for TF 2.3. e334251 on Dec 8, 2020 History

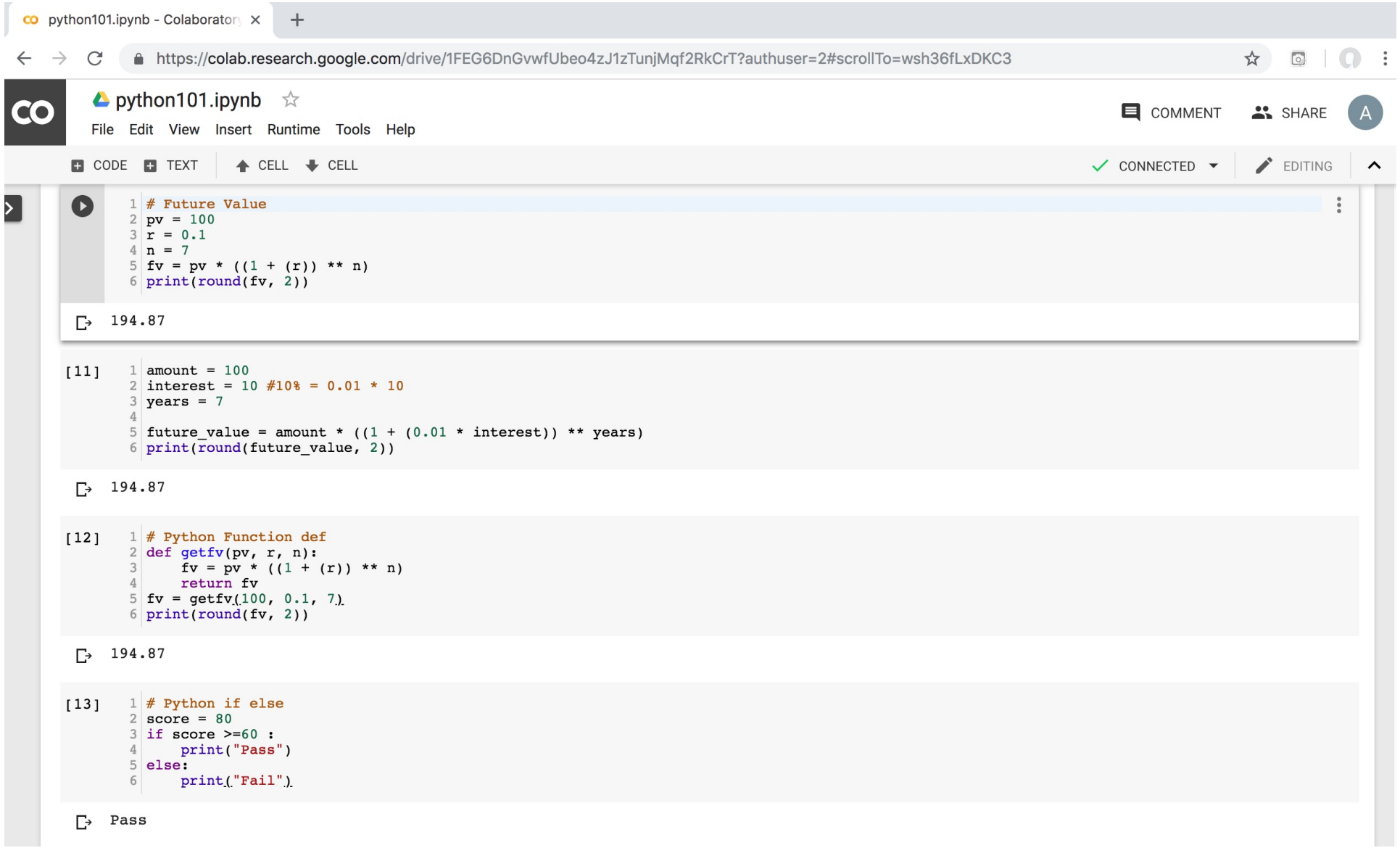
..	
oanda	Code updates for TF 2.3.
01_artificial_intelligence.ipynb	Code updates for TF 2.3.
02_superintelligence.ipynb	Code updates for TF 2.3.
03_normative_finance.ipynb	Code updates for TF 2.3.
04_data_driven_finance_a.ipynb	Initial commit.
04_data_driven_finance_b.ipynb	Initial commit.
05_machine_learning.ipynb	Code updates for TF 2.3.
06_ai_first_finance.ipynb	Code updates for TF 2.3.
07_dense_networks.ipynb	Code updates for TF 2.3.
08_recurrent_networks.ipynb	Code updates for TF 2.3.
09_reinforcement_learning_a.ipynb	Code updates.
09_reinforcement_learning_b.ipynb	Code updates for TF 2.3.



Source: https://github.com/yhilpisch/aiif/tree/main/code

Python in Google Colab (Python101)

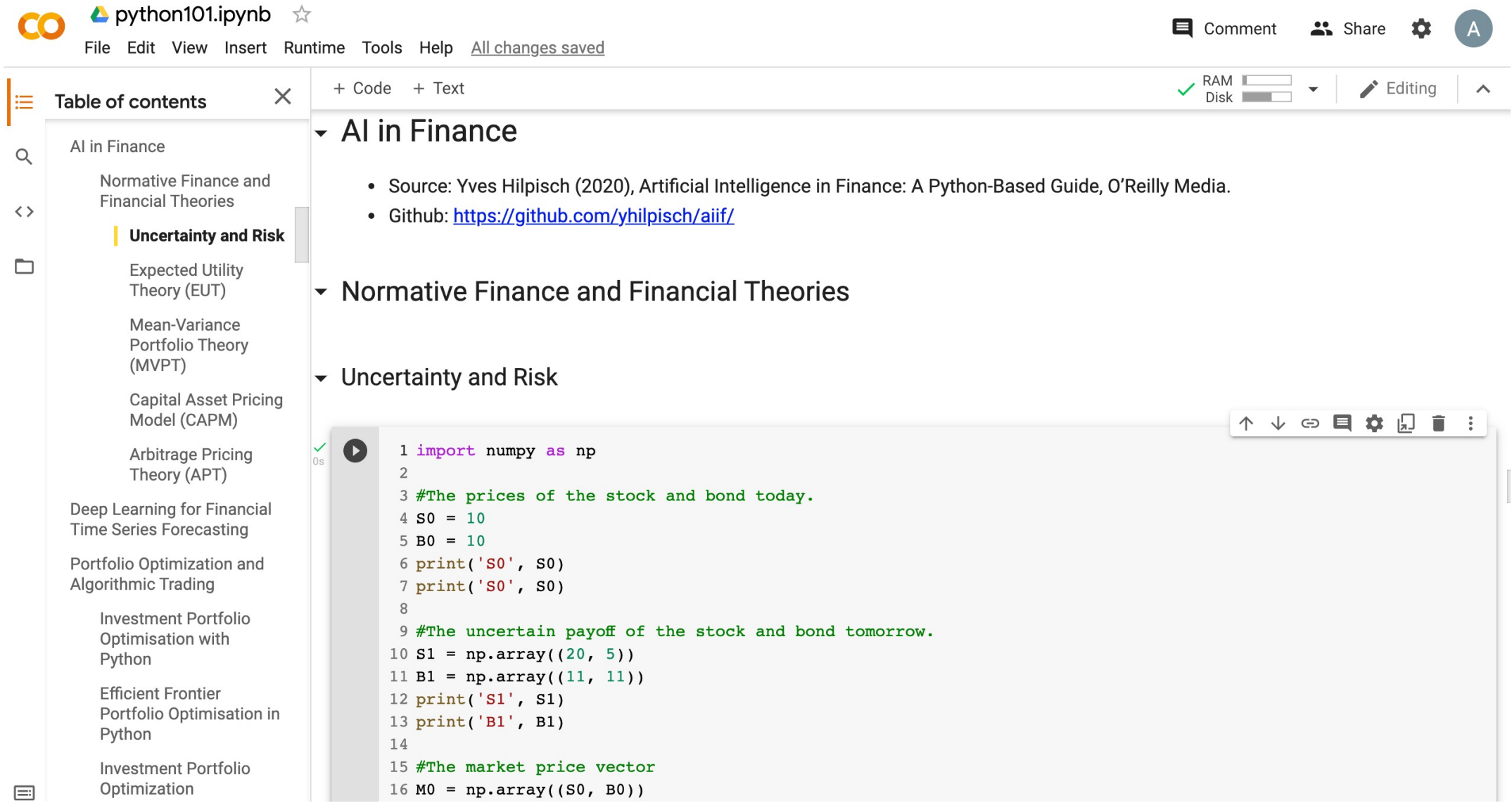
https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT



https://tinyurl.com/aintpupython101

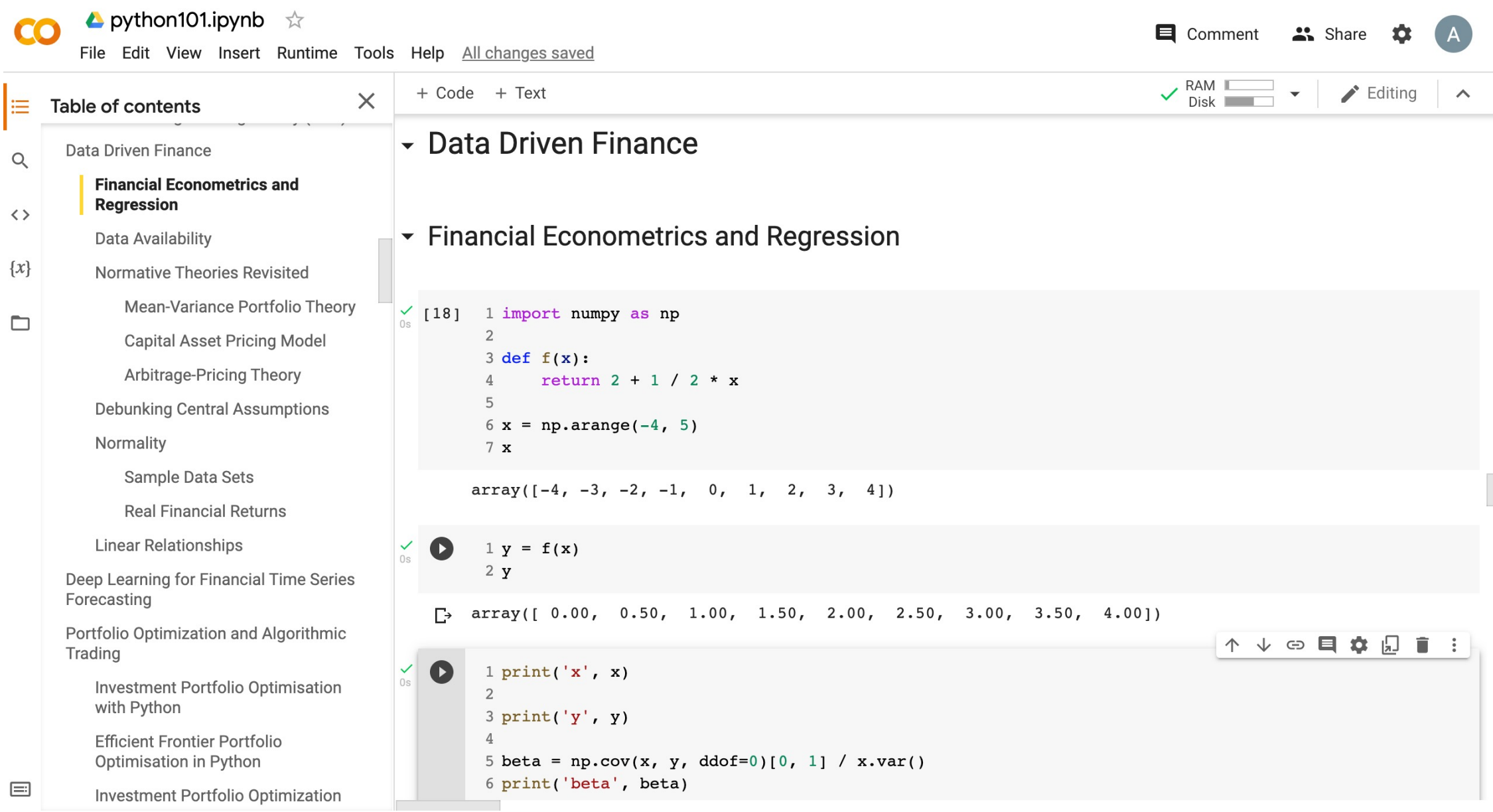
Python in Google Colab (Python101)

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



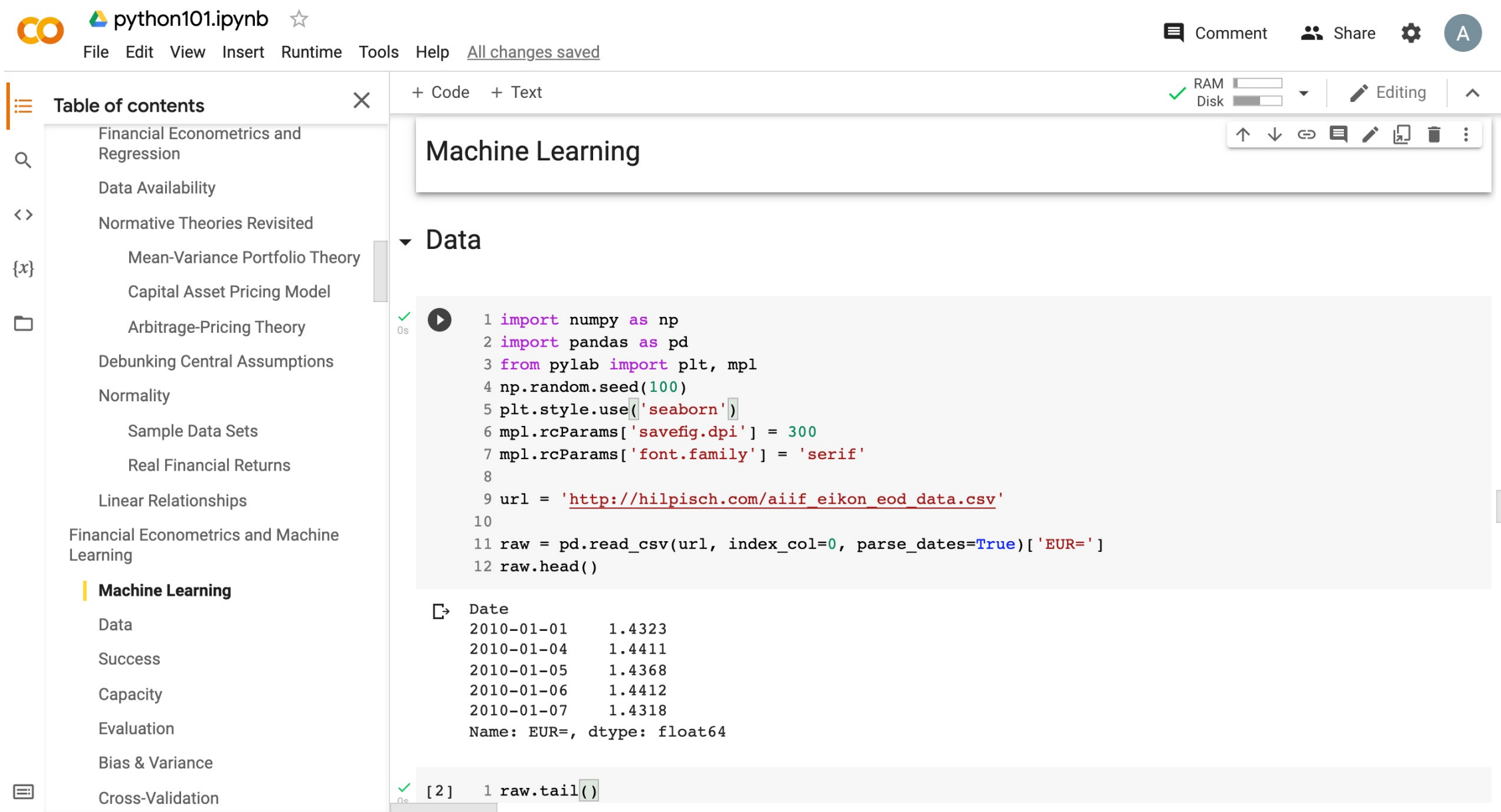
https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

Table of contents

- Mean-Variance Portfolio Theory
- Capital Asset Pricing Model
- Arbitrage-Pricing Theory
- Debunking Central Assumptions
- Normality
- Sample Data Sets
- Real Financial Returns
- Linear Relationships
- Financial Econometrics and Machine Learning
- Machine Learning
- Data
- Success
- Capacity
- Evaluation
- Bias & Variance
- Cross-Validation
- AI-First Finance
- **Efficient Markets**
- Market Prediction Based on Returns Data
- Market Prediction With More Features
- Market Prediction Intraday

+ Code + Text RAM Disk Editing

Efficient Markets

```
1 import numpy as np
2 import pandas as pd
3 from pylab import plt, mpl
4 plt.style.use('seaborn')
5 mpl.rcParams['savefig.dpi'] = 300
6 mpl.rcParams['font.family'] = 'serif'
7 pd.set_option('precision', 4)
8 np.set_printoptions(suppress=True, precision=4)
9
10 url = 'http://hilpisch.com/aiif_eikon_eod_data.csv'
11 data = pd.read_csv(url, index_col=0, parse_dates=True).dropna()
12 (data / data.iloc[0]).plot(figsize=(10, 6), cmap='coolwarm')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f29f972f210>
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

Table of contents

- Deep Learning (DL) in Finance
  - Dense Neural Networks (DNN)
    - Baseline Prediction
    - Normalization
    - Dropout
    - Regularization
    - Bagging
    - Optimizers
  - Recurrent Neural Networks (RNN)
    - First Example
    - Second Example
    - Financial Price Series
    - Financial Return Series
    - Financial Features
    - Deep RNNs
  - Convolutional Neural Networks (CNN)
- Reinforcement Learning (RL) in Finance

+ Code + Text Connect Editing

Deep Learning (DL) in Finance

- Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.
- Github: https://github.com/yhilpisch/aiif/

Dense Neural Networks (DNN)

```
1 import os
2 import numpy as np
3 import pandas as pd
4 from pylab import plt, mpl
5 plt.style.use('seaborn')
6 mpl.rcParams['savefig.dpi'] = 300
7 mpl.rcParams['font.family'] = 'serif'
8 pd.set_option('precision', 4)
9 np.set_printoptions(suppress=True, precision=4)
10 os.environ['PYTHONHASHSEED'] = '0'
```

```
[ ] 1 url = 'http://hilpisch.com/aiif_eikon_id_eur_usd.csv'
    2 symbol = 'EUR_USD'
    3 raw = pd.read_csv(url, index_col=0, parse_dates=True)
    4 raw.head()
```

	HIGH	LOW	OPEN	CLOSE

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

Table of contents

- Deep Learning (DL) in Finance
  - Dense Neural Networks (DNN)
    - Baseline Prediction
    - Normalization
    - Dropout
    - **Regularization**
    - Bagging
    - Optimizers
  - Recurrent Neural Networks (RNN)
    - First Example
    - Second Example
    - Financial Price Series
    - Financial Return Series
    - Financial Features
    - Deep RNNs
  - Convolutional Neural Networks (CNN)
- Reinforcement Learning (RL) in Finance

+ Code + Text Connect Editing

```
[0.35268253087997437, 0.8991981744766235]
```

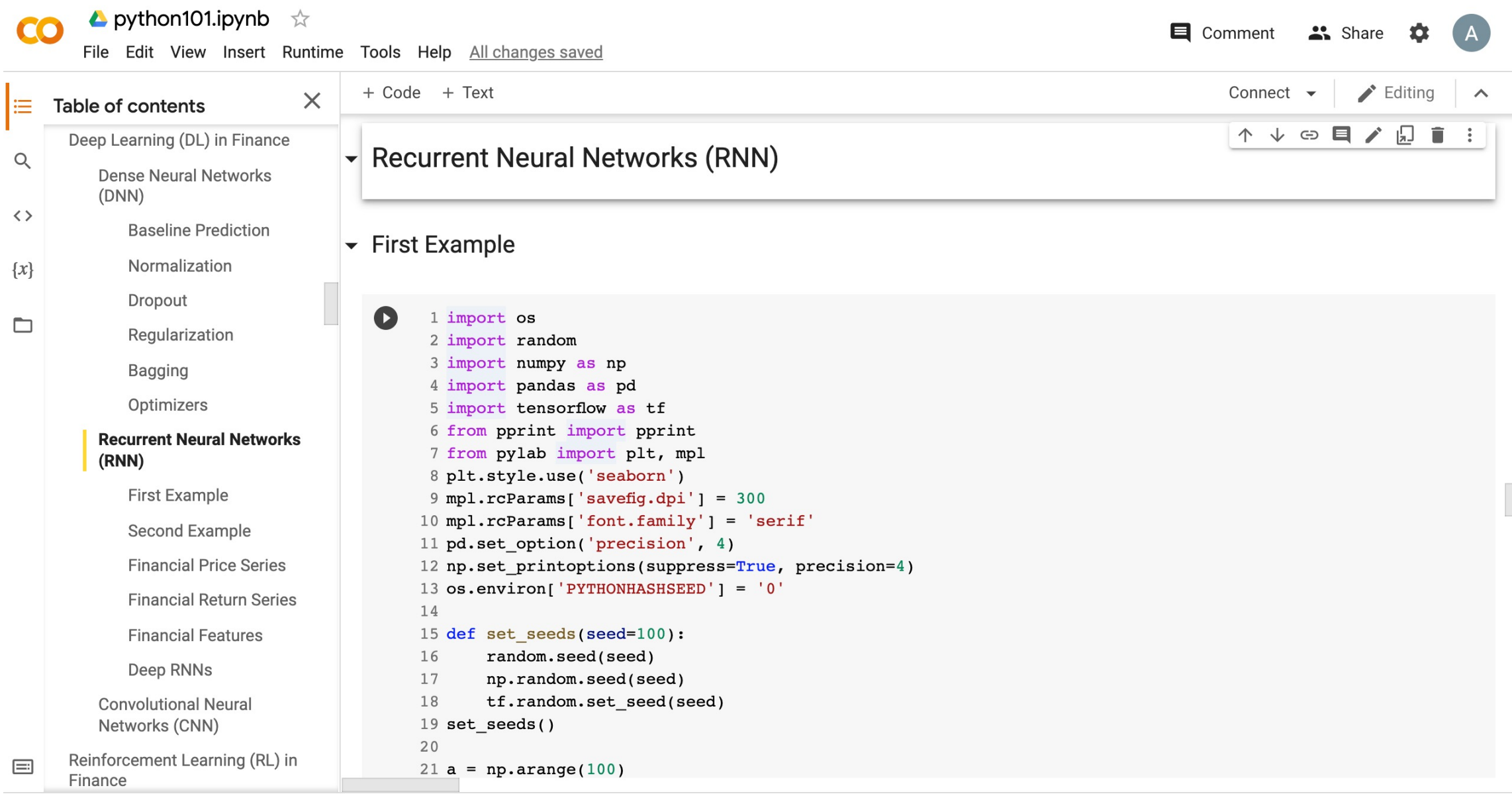
```
[ ]  1 model.evaluate(test_[cols], test['d'])
```

```
14/14 [==============================] - 0s 8ms/step - loss: 0.9098 - accuracy: 0.6705
[0.9098052382469177, 0.6704805493354797]
```

```
1 res = pd.DataFrame(h.history)
2 res[['accuracy', 'val_accuracy']].plot(figsize=(10, 6), style='--');
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)



https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb

File Edit View Insert Runtime Tools Help Saving...

Comment Share A

Table of contents

- Deep Learning (DL) in Finance
  - Dense Neural Networks (DNN)
    - Baseline Prediction
    - Normalization
    - Dropout
    - Regularization
    - Bagging
    - Optimizers
  - Recurrent Neural Networks (RNN)
    - First Example
    - Second Example
    - Financial Price Series
    - Financial Return Series
    - Financial Features
    - Deep RNNs
  - **Convolutional Neural Networks (CNN)**
- Reinforcement Learning (RL) in Finance

+ Code + Text Connect Editing

Convolutional Neural Networks (CNN)

```
1 import os
2 import math
3 import numpy as np
4 import pandas as pd
5 from pylab import plt, mpl
6 plt.style.use('seaborn')
7 mpl.rcParams['savefig.dpi'] = 300
8 mpl.rcParams['font.family'] = 'serif'
9 os.environ['PYTHONHASHSEED'] = '0'
10
11 url = 'http://hilpisch.com/aiif_eikon_eod_data.csv'
12 symbol = 'EUR='
13 data = pd.DataFrame(pd.read_csv(url, index_col=0,
14                                 parse_dates=True).dropna()[symbol])
15 data.info()
16 lags = 5
17 features = [symbol, 'r', 'd', 'sma', 'min', 'max', 'mom', 'vol']
18
19 def add_lags(data, symbol, lags, window=20, features=features):
20     cols = []
21     df = data.copy()
22     df.dropna(inplace=True)
23     df['r'] = np.log(df / df.shift(1))
24     df['sma'] = df[symbol].rolling(window).mean()
25     df['min'] = df[symbol].rolling(window).min()
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

Table of contents

- Financial Features
- Deep RNNs
- Convolutional Neural Networks (CNN)
- **Reinforcement Learning (RL) in Finance**
- Reinforcement Learning (RL)
- CartPole Environment
- Dimensionality Reduction
- Action Rule
- Total Reward per Episode
- Simple Learning
- Testing the Results
- DNN Learning
- Q Learning
- Finance Environment
- Improved Finance Environment
- Improved Financial QL Agent

+ Code + Text Connect Editing

Reinforcement Learning (RL) in Finance

- Source: Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media.
- Github: https://github.com/yhilpisch/aiif/

Reinforcement Learning (RL)

```
1 import os
2 import math
3 import random
4 import numpy as np
5 import pandas as pd
6 from pylab import plt, mpl
7 plt.style.use('seaborn')
8 mpl.rcParams['savefig.dpi'] = 300
9 mpl.rcParams['font.family'] = 'serif'
10 np.set_printoptions(precision=4, suppress=True)
11 os.environ['PYTHONHASHSEED'] = '0'
```

CartPole Environment

```
[ ]  1 import gym
     2
```

https://tinyurl.com/aintpupython101

Python in Google Colab (Python101)

python101.ipynb ☆

File Edit View Insert Runtime Tools Help All changes saved

Comment Share A

Table of contents

- Deep RNNs
- Convolutional Neural Networks (CNN)
- Reinforcement Learning (RL) in Finance
- Reinforcement Learning (RL)
  - CartPole Environment
  - Dimensionality Reduction
  - Action Rule
  - Total Reward per Episode
  - Simple Learning
  - Testing the Results
  - DNN Learning
  - Q Learning
  - Finance Environment
  - Improved Finance Environment
  - **Improved Financial QL Agent**

+ Code + Text Connect Editing

Improved Financial QL Agent

```
from collections import deque

class FQLAgent:
    def __init__(self, hidden_units, learning_rate, learn_env, valid_env):
        self.learn_env = learn_env
        self.valid_env = valid_env
        self.epsilon = 1.0
        self.epsilon_min = 0.1
        self.epsilon_decay = 0.98
        self.learning_rate = learning_rate
        self.gamma = 0.95
        self.batch_size = 128
        self.max_treward = 0
        self.trewards = list()
        self.averages = list()
        self.performances = list()
        self.aperformances = list()
        self.vperformances = list()
        self.memory = deque(maxlen=2000)
        self.model = self._build_model(hidden_units, learning_rate)

    def _build_model(self, hu, lr):
        model = Sequential()
        model.add(Dense(hu, input_shape=(
            self.learn_env.lags, self.learn_env.n_features),
                        activation='relu'))
```

https://tinyurl.com/aintpupython101

Summary

- Deep Learning (DL) in Finance
  - Dense Neural Networks (DNN)
  - Recurrent Neural Networks (RNN)
  - Convolutional Neural Networks (CNN)
- Reinforcement Learning (RL) in Finance
  - Q Learning (QL)
  - Improved Finance Environment
  - Improved Financial QL Agent

References

- Yves Hilpisch (2020), Artificial Intelligence in Finance: A Python-Based Guide, O'Reilly Media, https://github.com/yhilpisch/aiif .
- 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.
- 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.
- Min-Yuh Day (2021), Python 101, https://tinyurl.com/aintpupython101