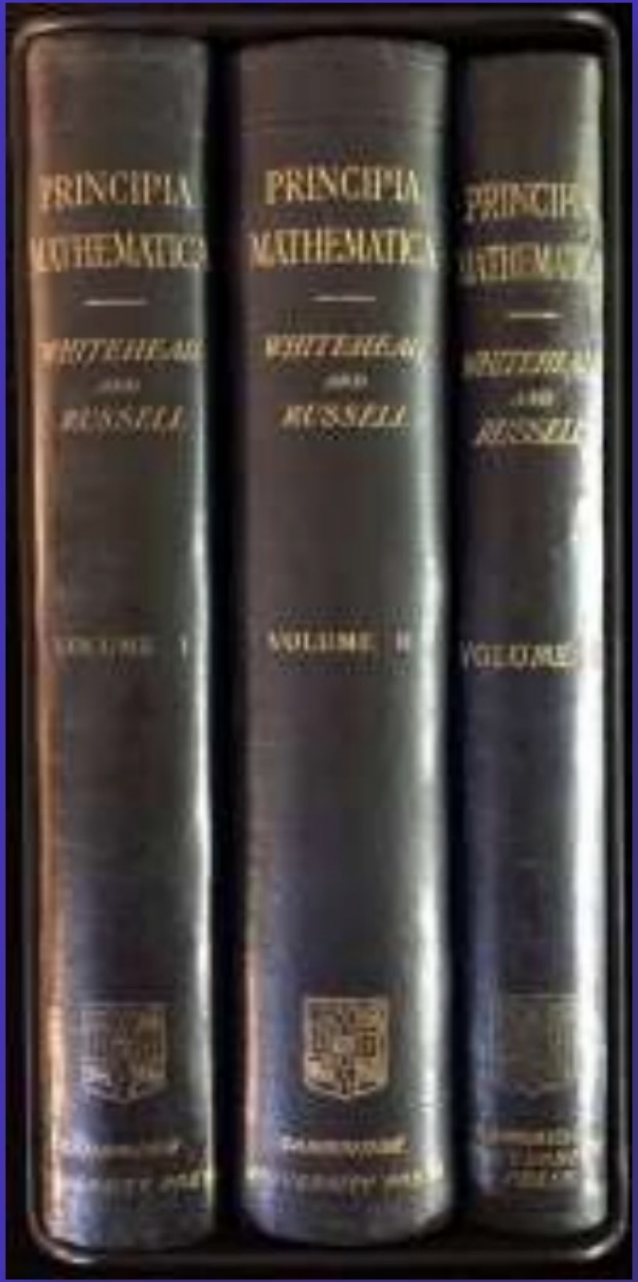
immerse
EDUCATION

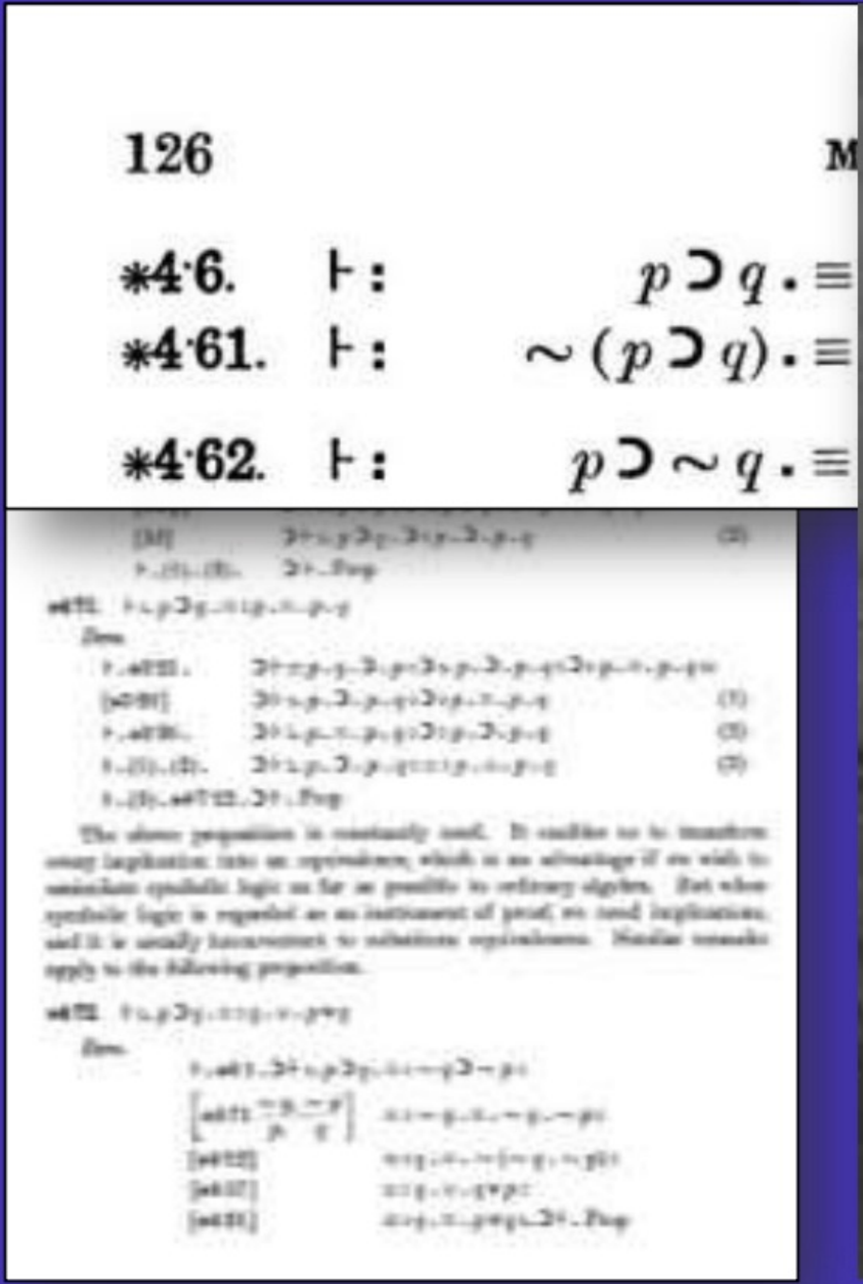
Deep Learning Introduction

Elizabeth Savochkina | 13th June

Shape of a Day

Registration 9.00am to 9.30am
Deep Learning Introduction 9.30am to 11.00am
Workshop: **SVMs and Decision trees** 11.00am to 11.30am
Break 11.30am to 13.30 pm
Workshop: **Continuation** 13.30pm to 14.30pm
Capstone Project work 14.30pm to 15.30pm





126

$*4{\cdot}6.\quad \vdash : \quad p \supset q \,.\, \equiv$

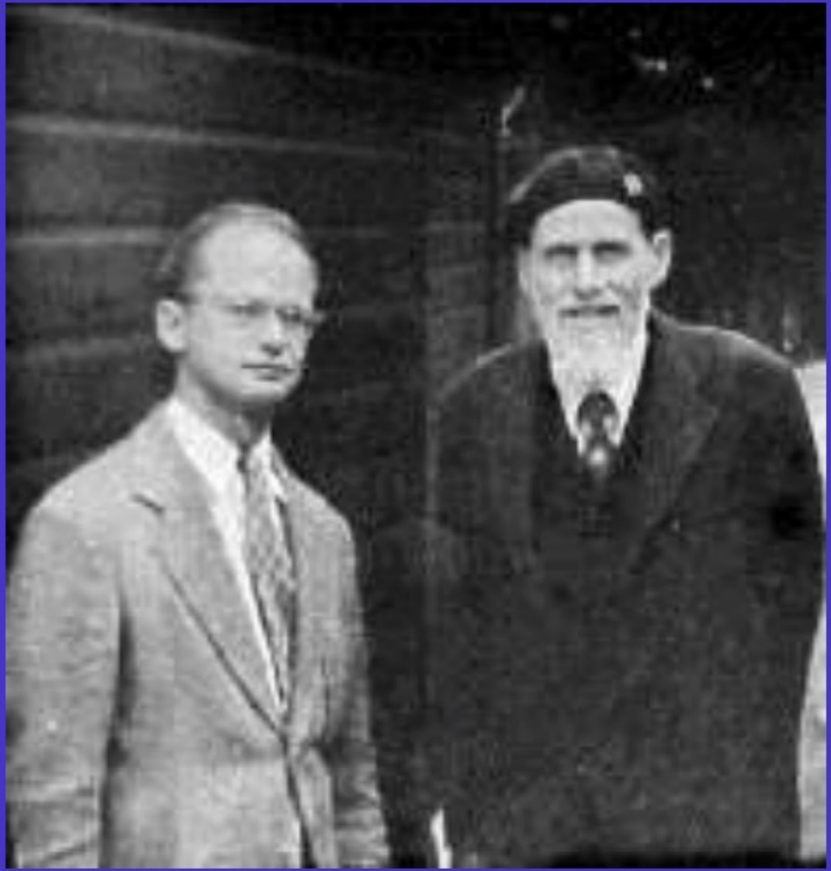
$*4{\cdot}61.\quad \vdash : \quad \sim(p \supset q) \,.\, \equiv$

$*4{\cdot}62.\quad \vdash : \quad p \supset \sim q \,.\, \equiv$

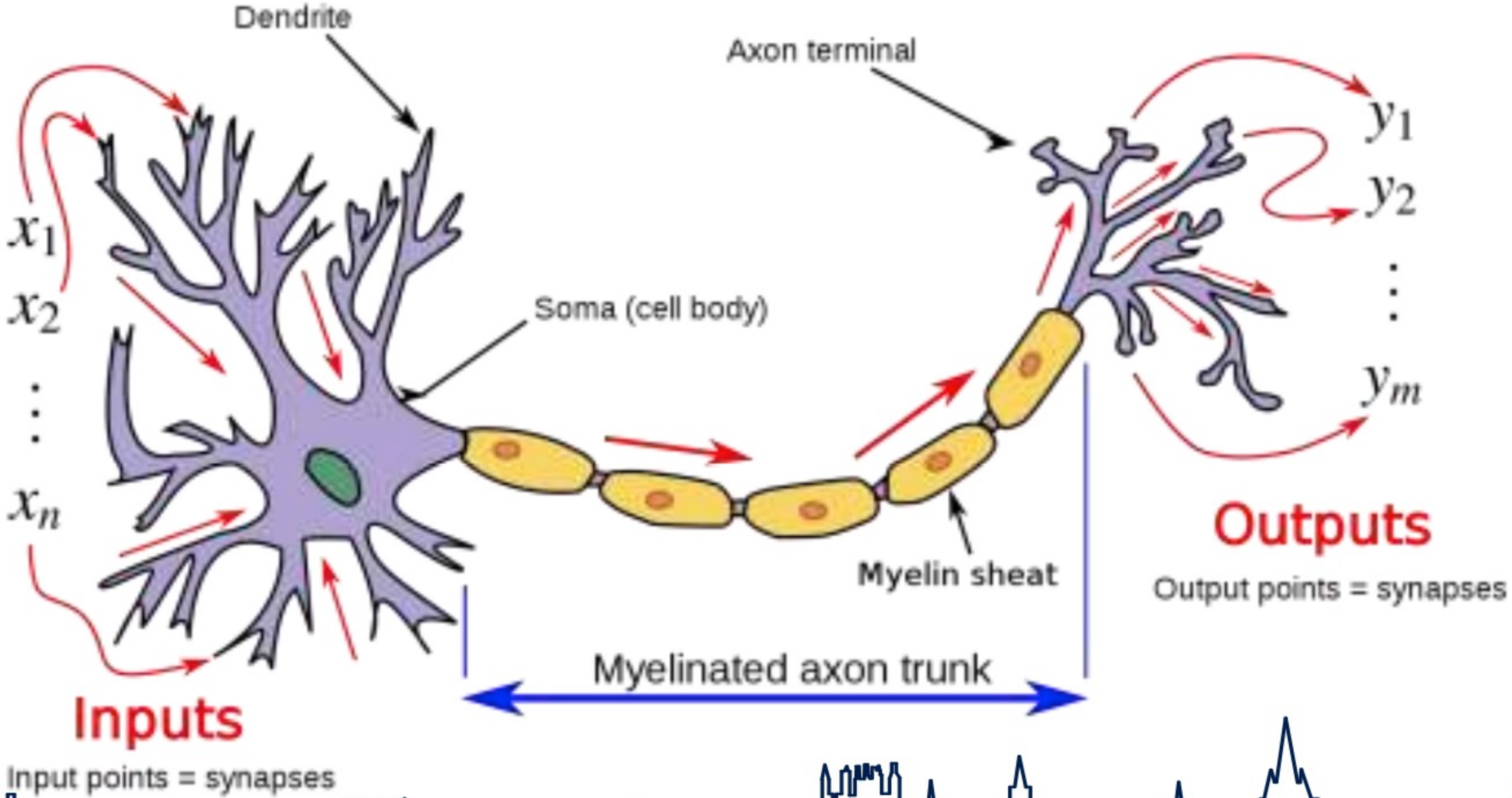
Bertrand Russell (1872-1970)

Walter Pitts (1923 – 1969)

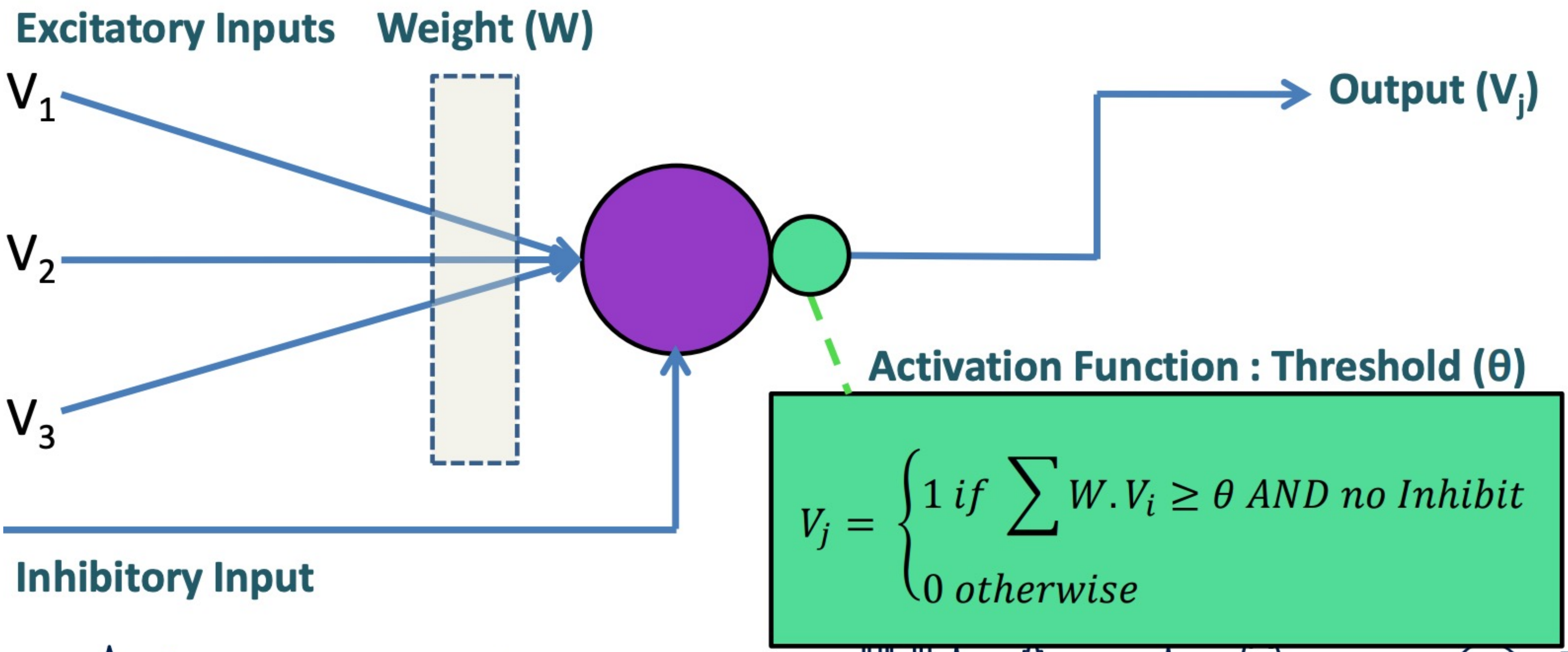
Warren Sturgis McCulloch
(1898 – 1969)



McCulloch and Pitts Attempted to Model Neurons



McCulloch-Pitts Model



"Introducing the Perceptron - A machine which senses, recognizes, remembers and responds like the human mind"

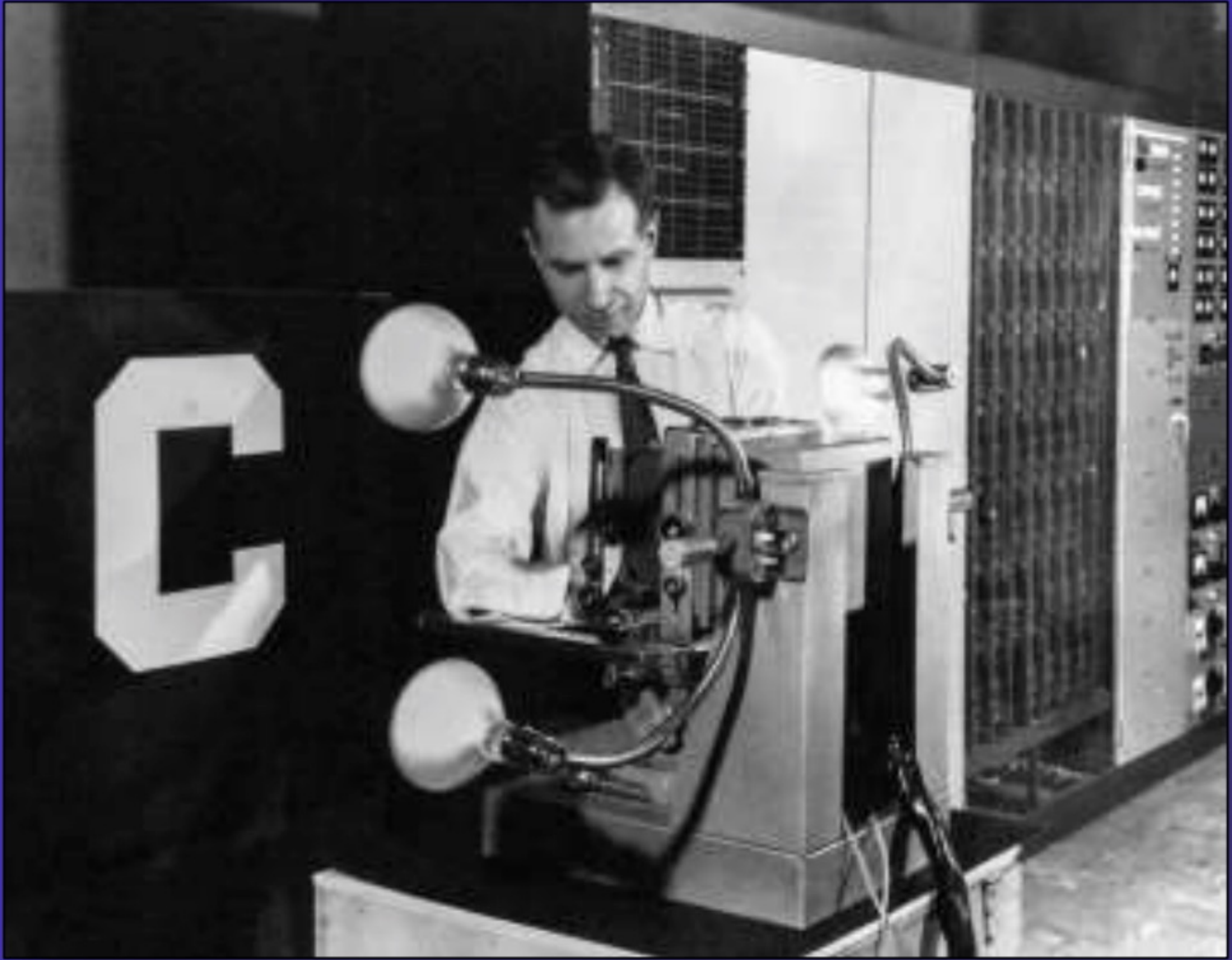
Cornell Aeronautical Laboratory, Research Trends, Summer 1958


research trends

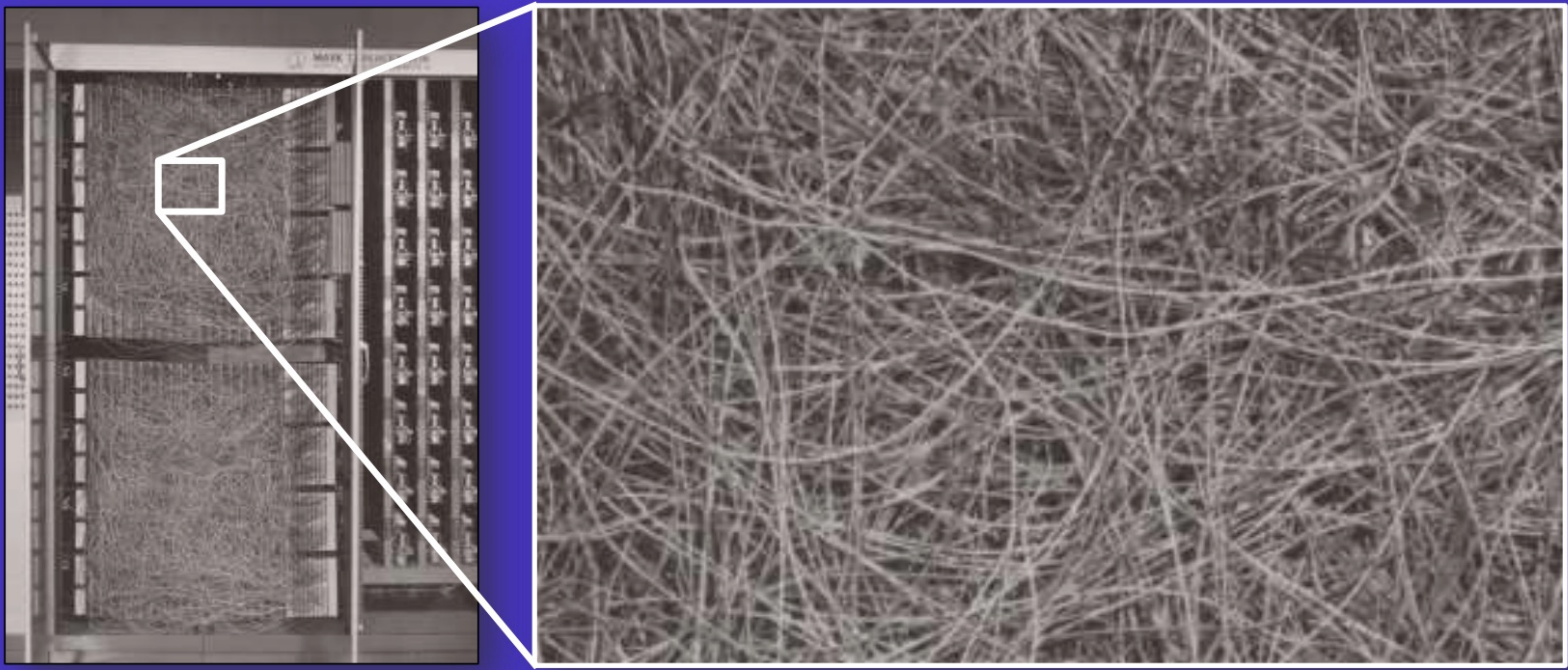
The Design of an

Intelligent AUTOMATON

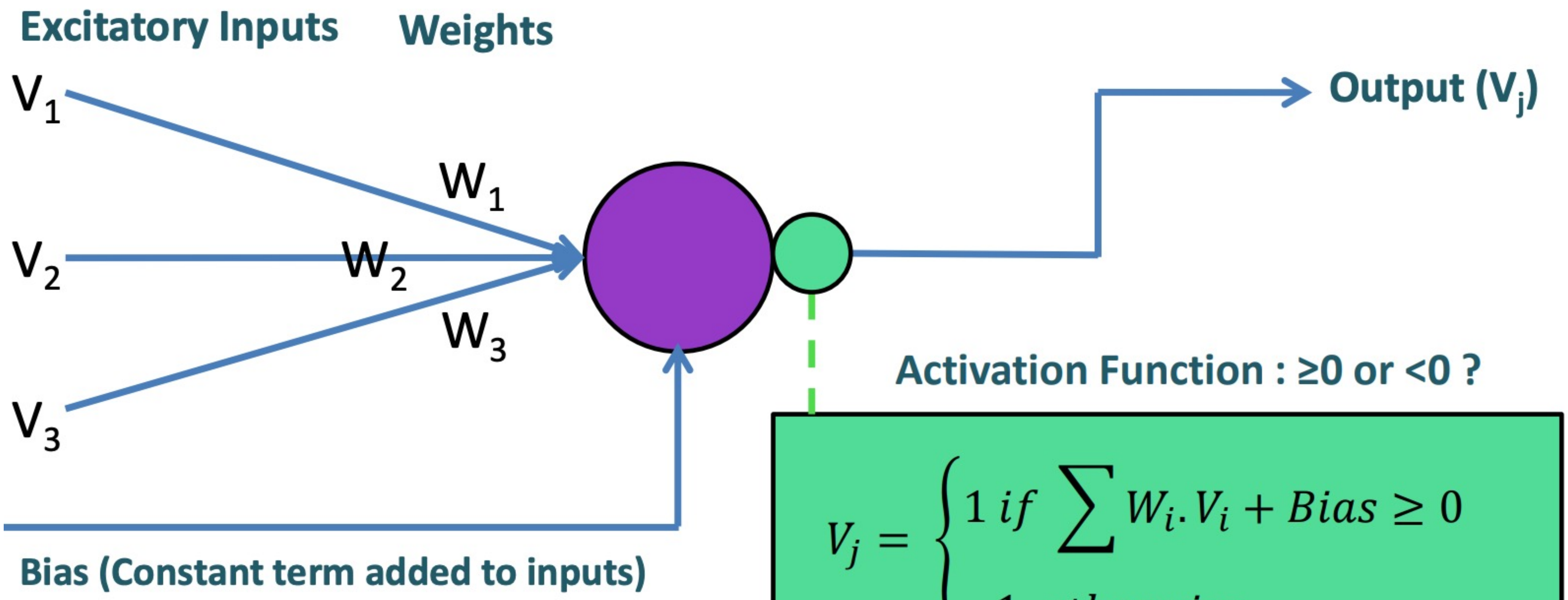
Cornell Aeronautical Laboratory,
Research Trends, Summer 1958



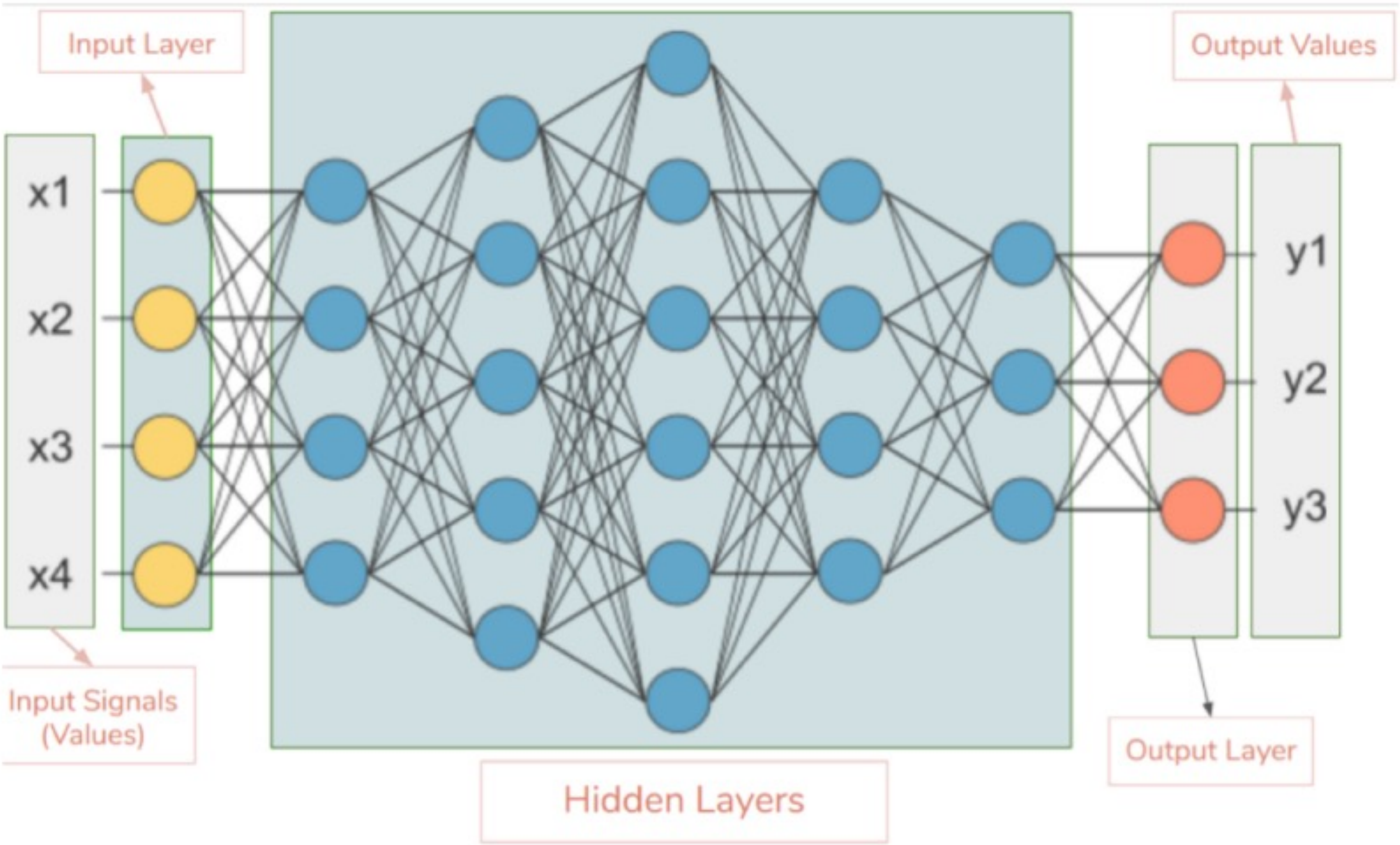
Frank Rosenblatt (1928 - 1971)
Demonstrating the Mark 1 Perceptron



Rosenblatt Perceptron



$$V_j = \begin{cases} 1 \; if \; \sum W_i . V_i + Bias \geq 0 \\ -1 \;\; otherwise \end{cases}$$



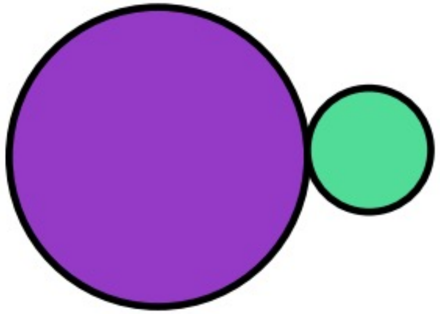
Neuron (x) is characterized by its WEIGHT, BIAS and ACTIVATION FUNCTION

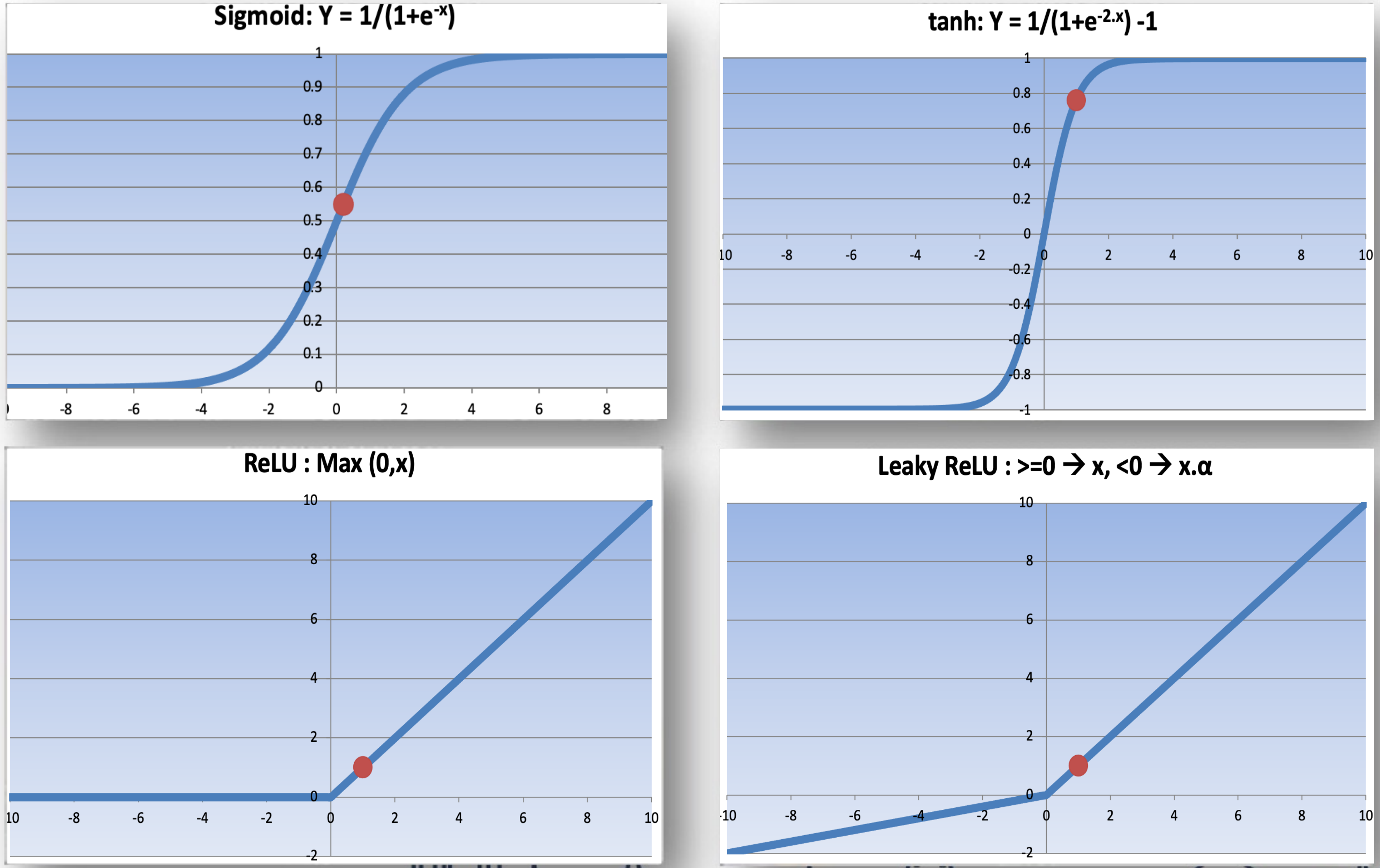
```
x = (weight * input) + bias
```

*Bias and weights explained later in slides

Next, **activation function** is applied → $Y = \mathrm{Activation}(\Sigma(weight * input) + bias)$

Different Activation Functions





Many options for Activation Function

Each has advantages and disadvantages

Activity : Understanding Activation Functions

Purpose

- Build an understanding of Activation functions and their behaviour

Groupings

<u>Individuals</u> .. You may send messages to other course members / team if you wish

Activity

- Download the simple NN Excel spread sheet
- Open the 'Activation Function' tab
- Move the slider up and down and watch the four different functions
- Notice how they each have different behaviour

Debrief

- Think about these functions and their behaviour
- Play with these after the lecture session .. As we will refer to these functions later in the course

Time and Tools

Time : 5 mins
Tools:

Can we do without an Activation Function?

- Activation function introduces an additional step at each layer during forward propagation.
- It is making the model complex. Why use it?

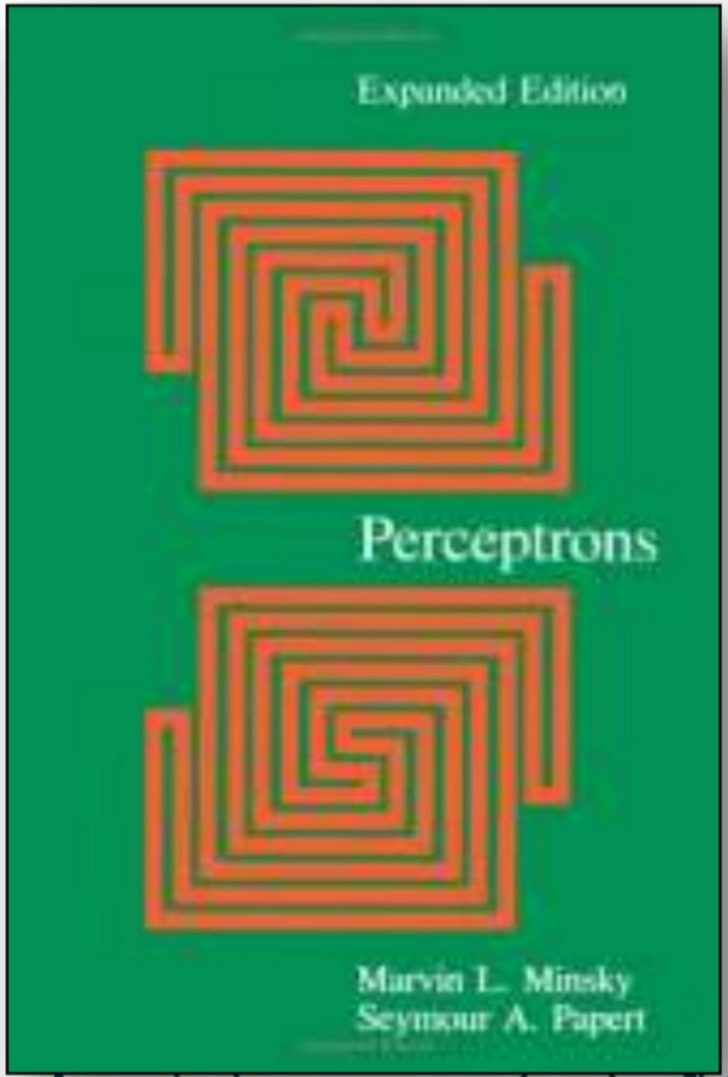
Neural Net without activation functions:

```
x = (weight * input) + bias
```

→ mx + c

- Every neuron performs a linear transformation on the inputs using the weights and biases
- The network is less powerful → cannot learn the complex patterns from the data
- Essentially a linear regression model

1969 : Minsky and Papert Reject the Perceptron Mark 1



Minsky and Papert : "It can't even do an XOR function!"*

X1	X2	XOR
0	0	0
1	0	1
0	1	1
1	1	0

- Minsky and Papert's book had a big negative impact on perception research
- The book was widely interpreted as arguing that networks of neurons could not implement some simple Boolean functions
- Strictly speaking the book does not argue that – their criticism is much more restricted
- … but maybe they did believe this limitation ..

“Virtually nothing is known about the computational capabilities of [a multi-layer neural network]. We believe that it can do little more than can a low order perceptron.”

Minsky and Papert in the 1971 Report of Project MAC

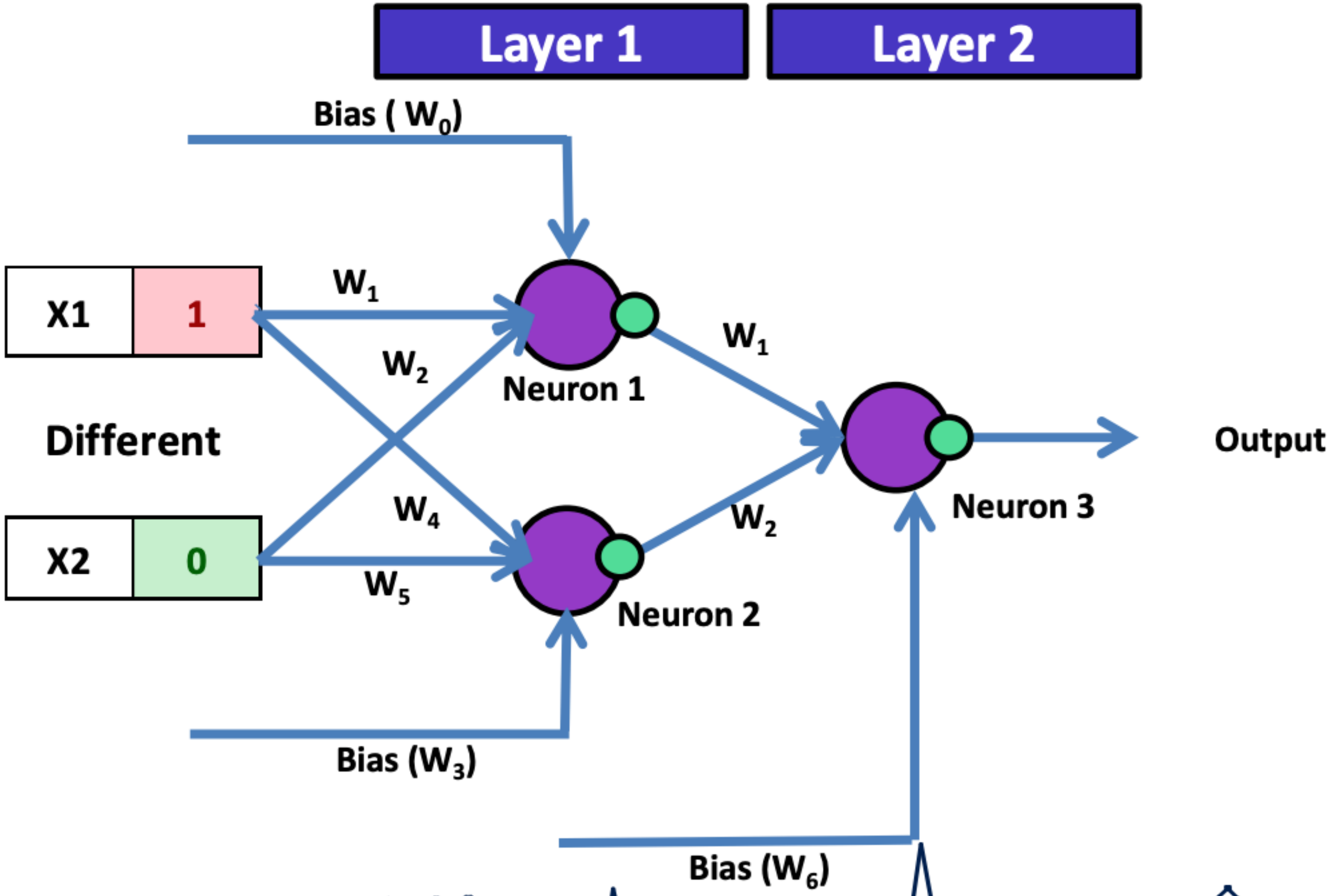
Can a Multi-layer Network perform an XOR function?

Experimenting with Multiple Layers

Modelling Problem

If the two inputs are the same, then output a "0"
If the two inputs are different, then output a "1"

Input Data		Label
X1	X2	Y
0	0	0
1	0	1
0	1	1
1	1	0



Experimenting in Excel

Slider controls make setting weights easier and quicker

Weights for the Neuron Inputs

X1	1

Different

X2	0

Input Data		Label
X1	X2	Y
0	0	0
1	0	1
0	1	1
1	1	0

Neuron 1		
Weight Name	Weight	Activation
W0 (Bias)	-11.0000	
W1	3.8000	-22.000
W2	-14.8000	0.000

Weight Controls - Neuron 1

W0 (Bias)

W1

W2

Neuron 2		
Weight Name	Weight	Activation
W3 (Bias)	2.4000	
W4	1.8000	18.60
W5	14.4000	1.00

Weight Controls Neuron 2

W3 (Bias)

W4

W5

Neuron 3		
Weight Name	Weight	Activation
W6 (Bias)	3.2000	
W7	11.0000	-12.200
W8	-15.4000	0.000

Weight Controls Neuron 3

W6 (Bias)

W7

W8

Prediction	Right or Wrong?
0	✓
0	✗
0	✗
0	✓

Activity : Can you 'tune' this network?

Purpose

- Build an understanding of how NNs work
- Understand the computational effort required to fit (tune) a NN

Groupings

Individuals .. You may send messages to other course members / team if you wish

Activity

- Download the simple NN Excel spread sheet
- Open the 'XOR' tab
- Move the sliders up and down to adjust (tune / fit) the model
- Your goal is to get 4 green ticks in the 'Right or Wrong' column
- Play with the model until you understand it

Debrief

- How much effort does it take to tune / fit the model?
- How much more effort would it take if there were thousands of neurons in the model?
- Do you think that this could be done by a human using random movements of the sliders?

Time and Tools

Time : 10 mins
Tools:

The end of the story?

Walter Pitts

- Pitts remained at MIT but became increasingly socially isolated
- In 1959 he co-wrote a book that proved that analogue processing in the eye were significant contribution to vision
- The result lead him to burn his PhD thesis along with years of unpublished research
- Died in 1969 of alcoholism

Warren Sturgis McCulloch

- Warren Sturgis McCulloch had a wide range of interests:
- Wrote books of poetry
- Architected buildings
- A founding member of the American Society for Cybernetics
- Happily married with three children

Frank Rosenblatt

- Frank Rosenblatt had wide scientific and social interest:
- He taught Psychology and Neuro-biology
- Researched transfer of learning in rats
- Built an astronomical observatory and proposed a method for detecting satellites
- Very active in politics
- Died on his 43 birthday in a boating accident

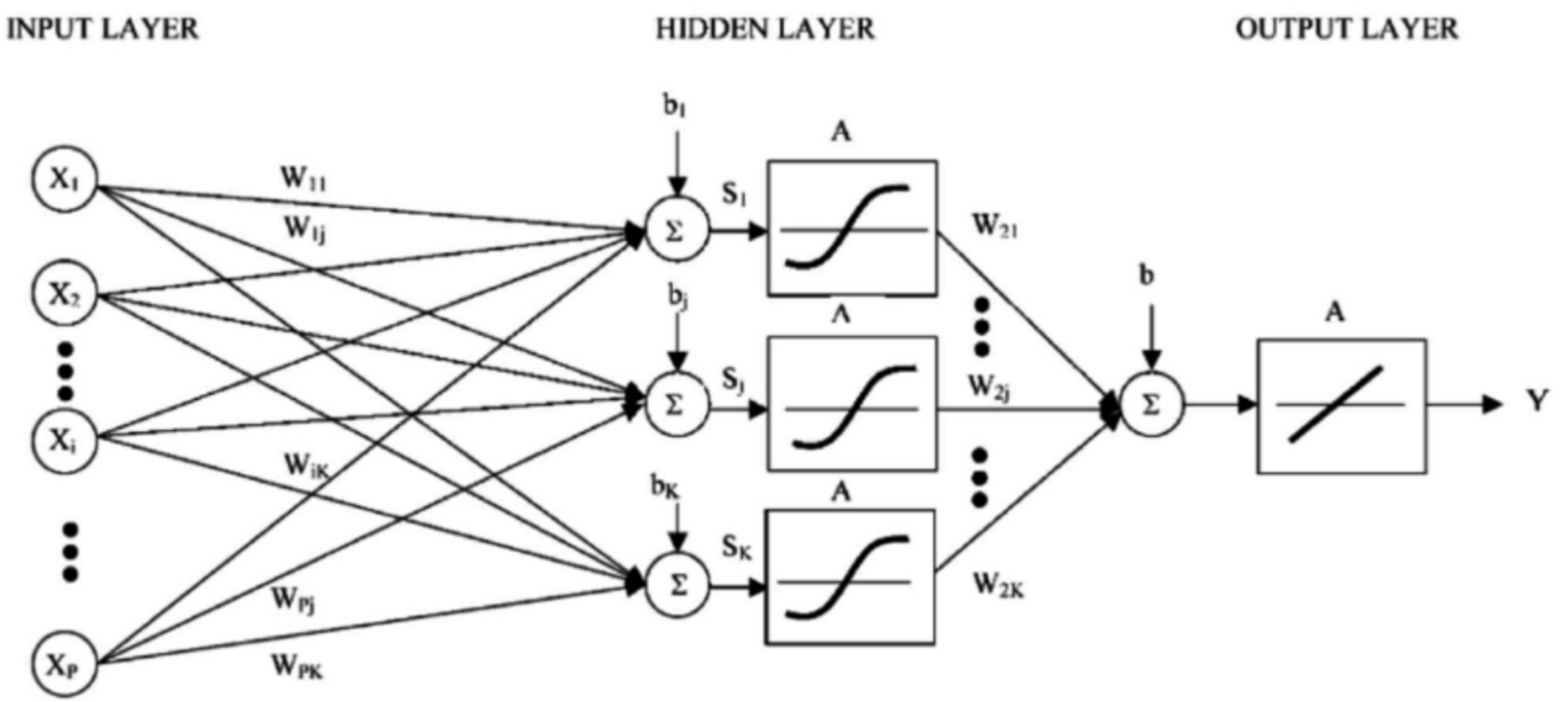
Marvin Minsky

- Co-founder of MIT's AI Laboratory
- Won a Turing award in 1969
- Invented the first head-mounted graphical display
- Invented the 'confocal microscope'
- In 1951 built the first randomly wired Neural Network
- He was an adviser on Stanley Kubrick's movie 2001: A Space Odyssey

Seymour Papert

- Born in South-Africa
- Completed a PhD in mathematics in South Africa then a second PhD at Cambridge in the UK
- Joined MIT in 1963
- Co-inventor of the logo programming language
- Co-director of the MIT AI laboratory
- Collaborated with Lego to develop their 'mindstorms' robotics construction sets

Neural Network Outlook



Bias - allows the **activation function** to be **shifted to the left or right,** to better fit data

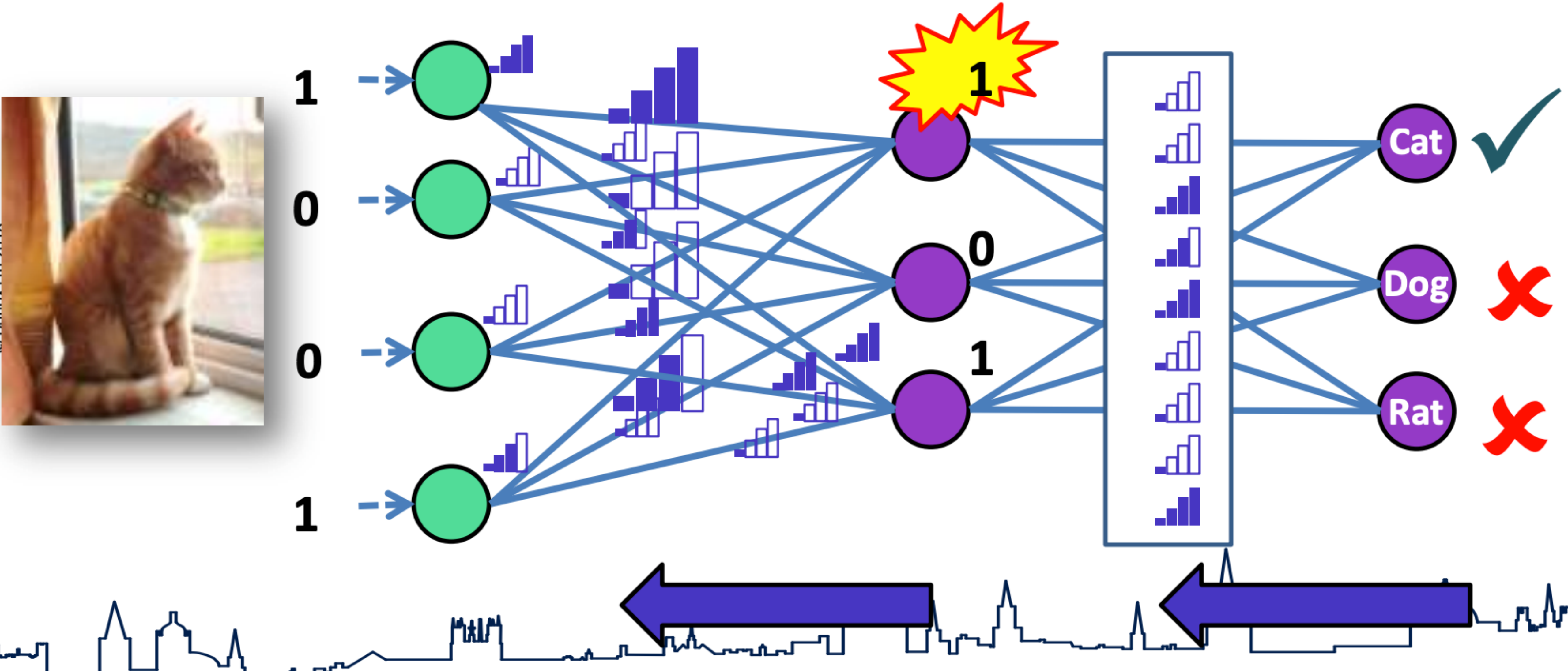
- Only influences the output values, it doesn't interact with the actual input data.
- The role of bias isn't to act as a threshold, but to help ensure the output **best fits the incoming signal**.

Weights – control the signal/strength of neuron fired (randomly initialized and fixed)

- Weights are not being adjusted correctly when our model overfits.

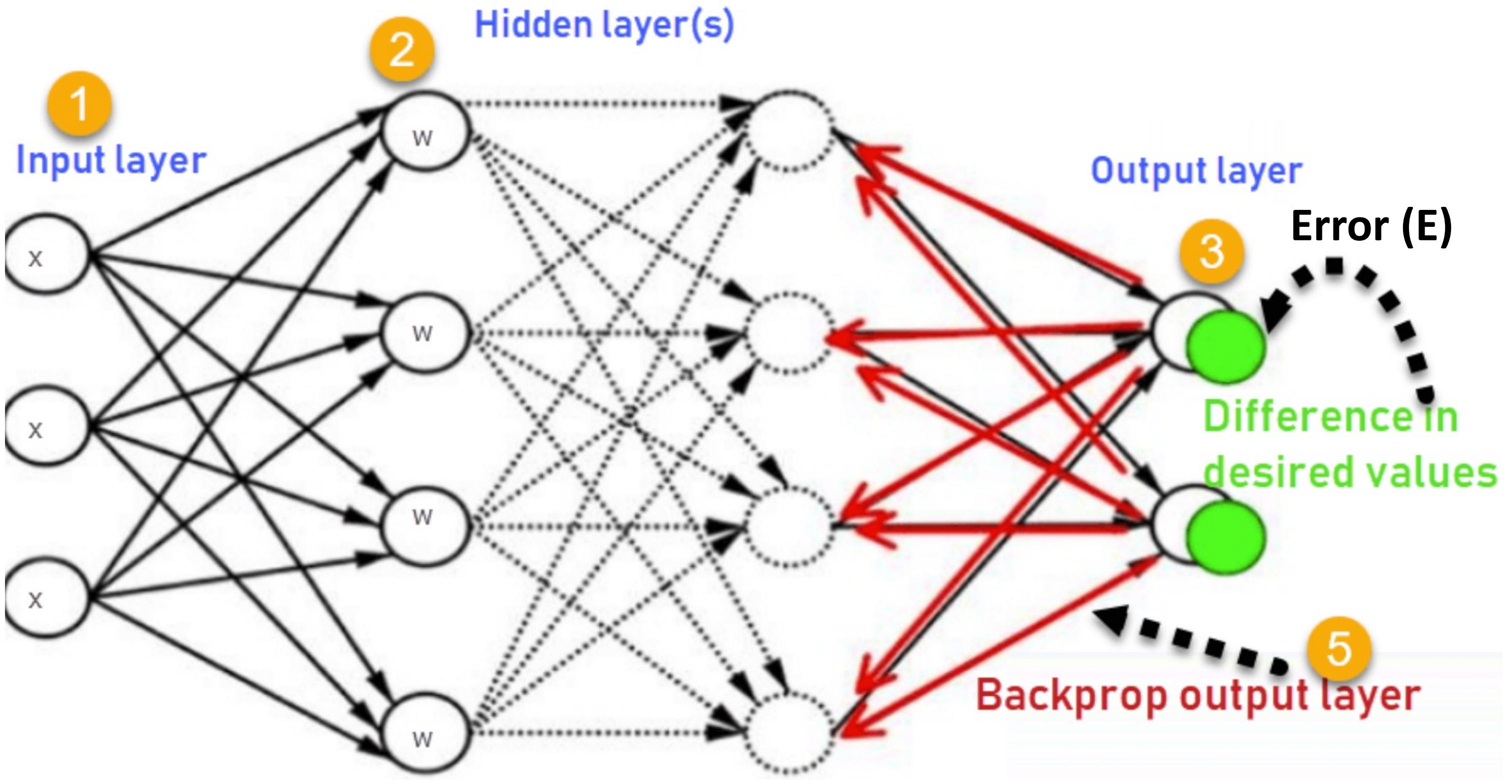
Back Propagation : Automatically Fitting Neural Networks

Back-propagation: Automatic Fitting of Neural Networks

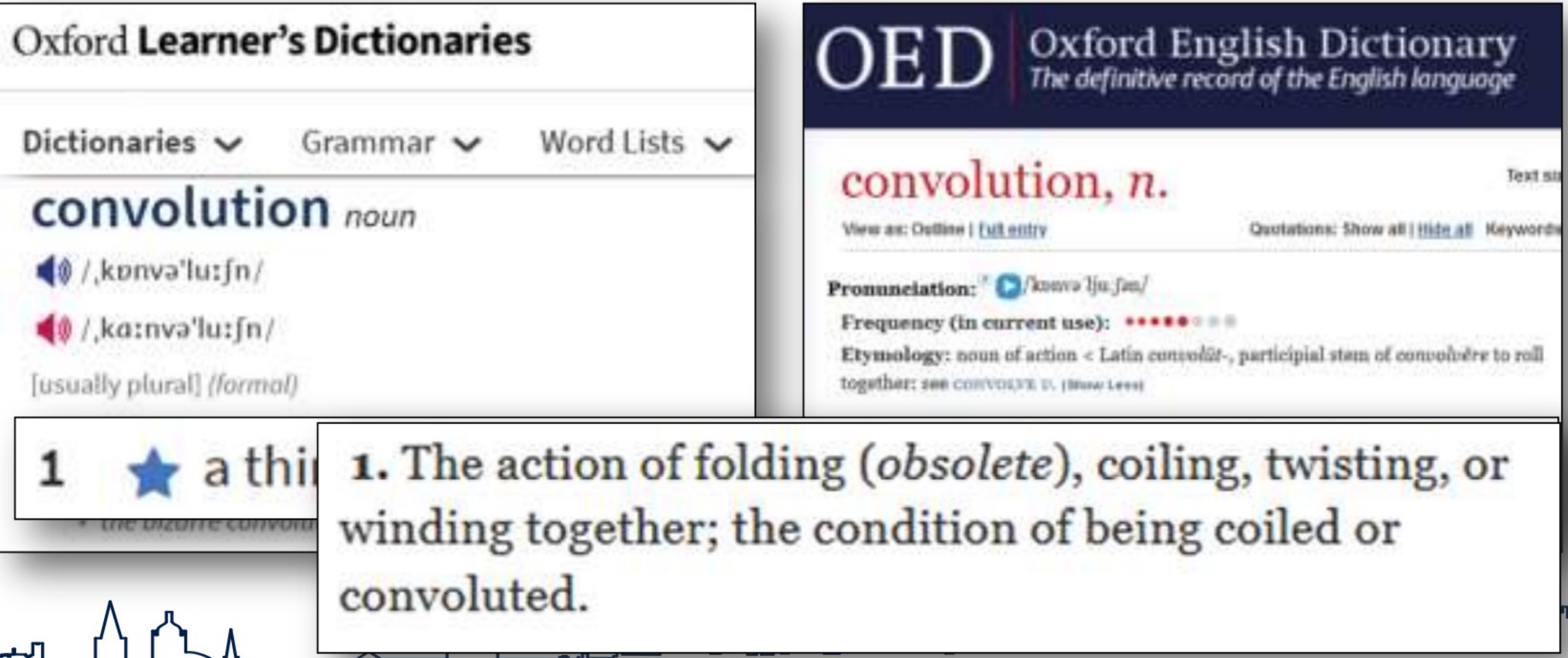


Backpropagation

The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. (Rosenblatt, F)



Convolution



The story continues ...

Learning representations by back-propagating errors

David E. Rumelhart, Geoffrey E. Hinton & Ronald J. Williams

1986

Nature **323**, 533–536 (1986) | Cite this article

Abstract

We describe a new learning procedure, back-propagation, for networks of neurone-like units. The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. As a result of the weight adjustments, internal 'hidden' units which are not part of the input or output come to represent important features of the task domain, and the regularities in the task are captured by the interactions of these units. The ability to create useful new features distinguishes back-propagation from earlie perceptron-convergence procedure[1].

$$x_j = \sum_i y_i w_{ji}$$

David E. Rumelhart

Geoffrey E. Hinton

Ronald J. Williams

Critique of Hinton's Honda Prize

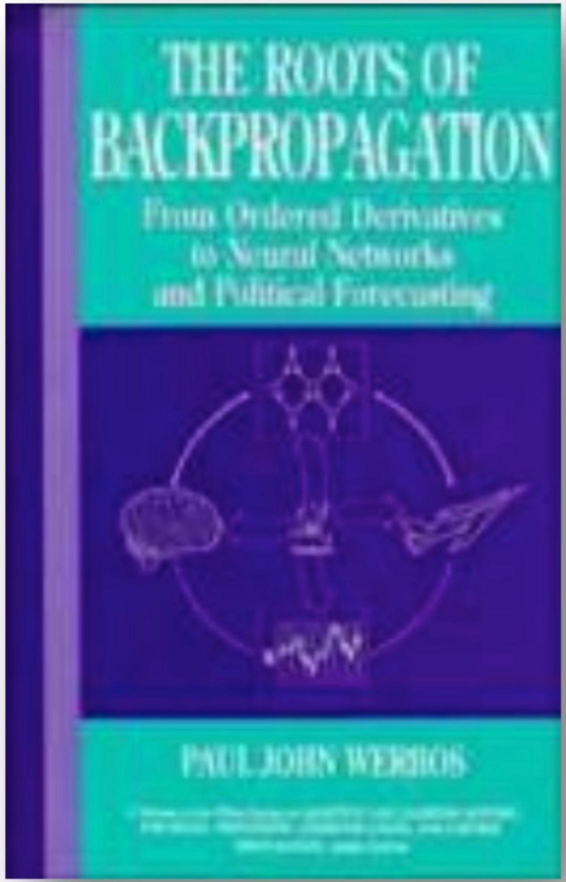
I. Honda: *"Dr. Hinton has created a number of technologies that have enabled the broader application of AI, including the backpropagation algorithm that forms the basis of the deep learning approach to AI."*

Critique: Hinton and his co-workers have made certain significant contributions to deep learning, e.g., [BM] [CDI] [RMSP] [TSNE] [CAPS]. However, **the claim above is plain wrong.** He was 2nd of 3 authors of an article on backpropagation [RUM] (1985) which failed to mention that 3 years earlier, Paul Werbos proposed to train neural networks (NNs) with this method (1982) [BP2]. **And the article [RUM] even failed to mention Seppo Linnainmaa, the inventor of this famous algorithm for credit assignment in networks [BP1] (1970), also known as "reverse mode of automatic differentiation."** (In 1960, Kelley already had a precursor thereof in the field of control theory [BPA]; compare [BPB] [BPC].) See also [R7].

Critique of Hinton winning the Honda Prize by Jürgen Schmidhuber

https://people.idsia.ch/~juergen/critique-honda-prize-hinton.html

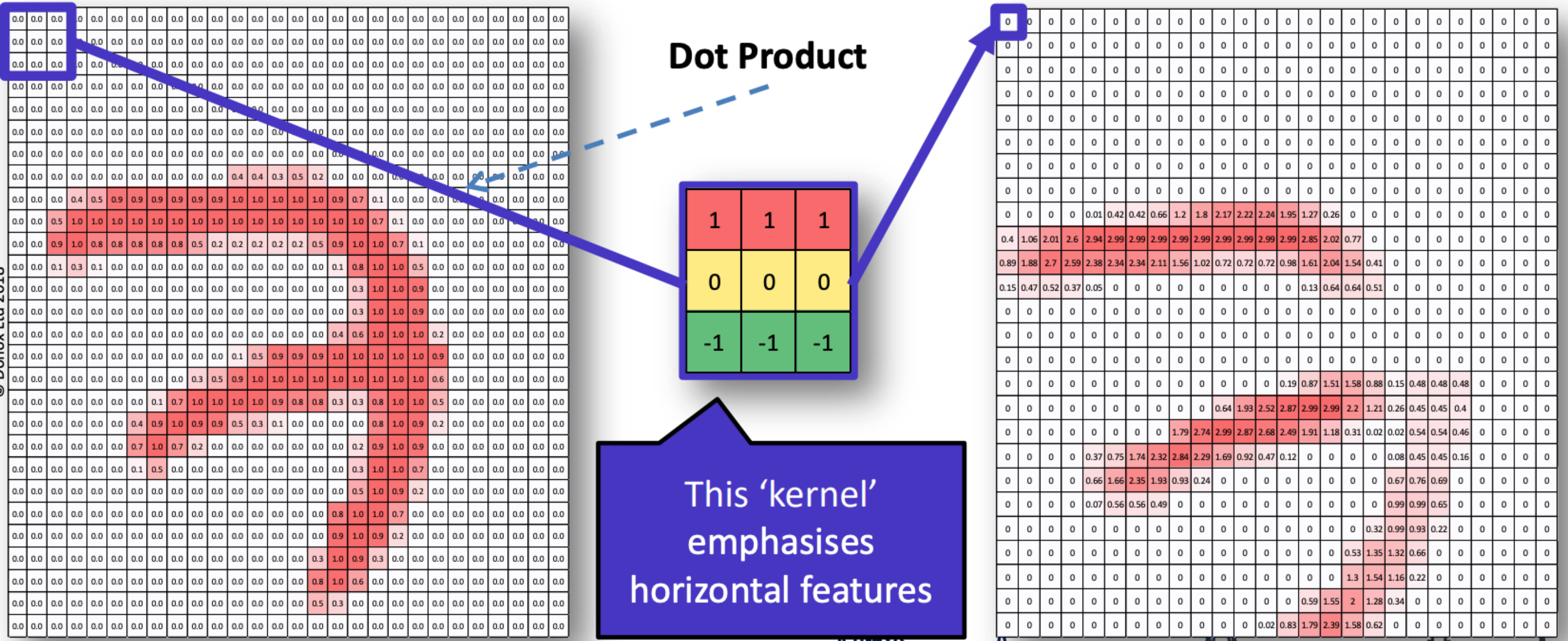
Paul Werbos



His PhD thesis described back-propagation in 1974!

Bonus Slides:
Layers of Densely connected Neurons are not the only option

Convolutions are Filters / ‘Feature Detectors’



Excel example based on Jeremy Howard’s as presented in “Practical Deep Learning for Coders”

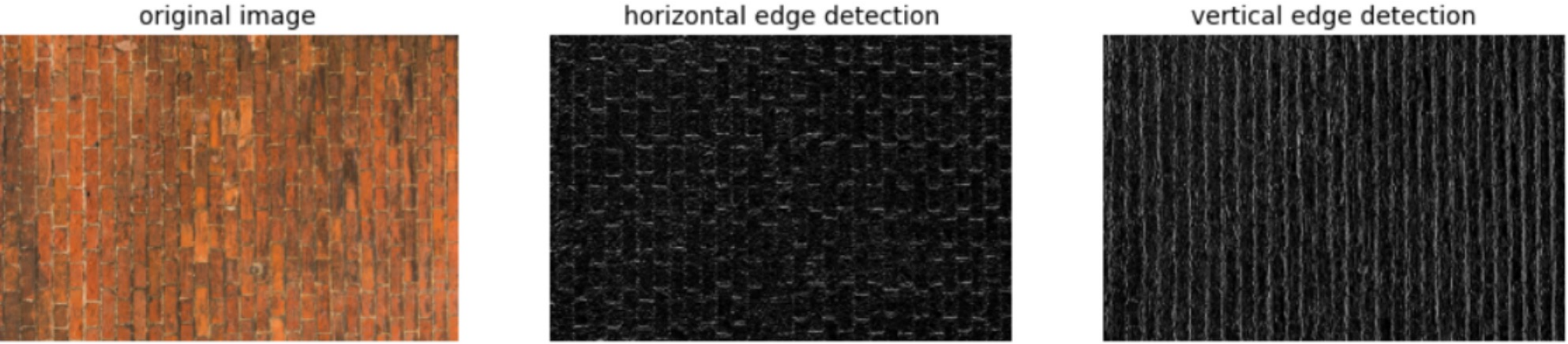
Convolutions are Filters / ‘Feature Detectors’

0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.4	0.3	0.5	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.4	0.5	0.9	0.9	0.9	0.9	0.9	0.9	1.0	1.0	1.0	1.0	1.0	0.9	0.7	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.7	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.9	1.0	0.8	0.8	0.8	0.8	0.8	0.5	0.2	0.2	0.2	0.2	0.2	0.5	0.9	1.0	1.0	0.7	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.1	0.3	0.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.8	1.0	1.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.0	1.0	0.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.0	1.0	0.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.6	1.0	1.0	1.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.5	0.9	0.9	0.9	1.0	1.0	1.0	1.0	1.0	0.9	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.5	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.7	1.0	1.0	1.0	1.0	0.9	0.8	0.8	0.3	0.3	0.8	1.0	1.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.4	0.9	1.0	0.9	0.9	0.5	0.3	0.1	0.0	0.0	0.0	0.0	0.8	1.0	0.9	0.2	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.7	1.0	0.7	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.2	0.9	1.0	0.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.0	1.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0	0.9	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.8	1.0	1.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.9	1.0	0.9	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	1.0	0.9	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.8	1.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

1	0	-1
1	0	-1
1	0	-1

This one
emphasises vertical
features

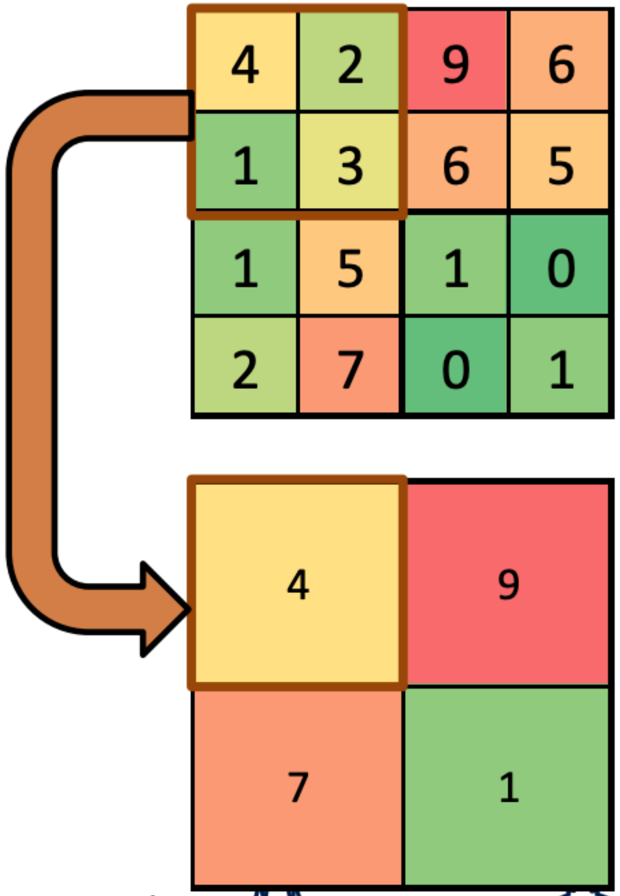
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0.08	0	0.06	0.46	0.24	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0.09	0	0.07	0.54	0.45	0.84	0.75	0.08	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0.09	0	0.07	0.54	0.45	1.09	1.65	0.82	0.09	0	0	0	0	0	0
0	0	0	0	0	0	0	0.24	0.54	0.24	0	0.01	0	0	0	0	0.97	1.91	1.74	0.83	0.08	0	0	0	0	0
0	0	0	0.53	0.09	0	0	0.24	0.54	0.31	0	0	0	0	0	0	0	1	2.2	1.83	0.53	0	0	0	0	0
0	0	0.17	0.54	0.09	0	0	0.24	0.54	0.31	0	0	0	0	0	0	0	0	1.54	2.73	1.45	0	0	0	0	0
0	0	0.1	0.32	0.05	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7	2.99	2.29	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.2	2.79	2.79	0.2	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.12	1.85	2.87	1.14	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.04	1.29	2.95	1.7	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0.07	0.42	0.44	0	0	0	1.03	2.99	1.96	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0.01	0.32	0.54	0.19	0.51	0.54	0	0	0	1.81	2.94	1.18	0	0	0	0
0	0	0	0	0	0	0	0	0.53	0.74	0.54	0.45	0.54	0.19	0.51	0.35	0	0	0	2.37	2.86	0.62	0	0	0	0
0	0	0	0	0	0	0	1.15	0.8	0.74	0.54	0.33	0.35	0.12	0	0	0	0	0.17	2.83	2.51	0.16	0	0	0	0
0	0	0	0	0	0	0.04	1.24	0.69	0.24	0	0	0	0	0	0	0	0	1.11	2.93	1.79	0	0	0	0	0
0	0	0	0	0	0	0.07	0.49	0	0	0	0	0	0	0	0	0	0	2.11	2.59	0.87	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.71	2.71	1.82	0.22	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.49	2.04	2.27	0.88	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.62	2.33	1.27	0.22	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.05	1.88	1.56	0.34	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.65	1.24	0.62	0	0	0	0	0	0	0	0

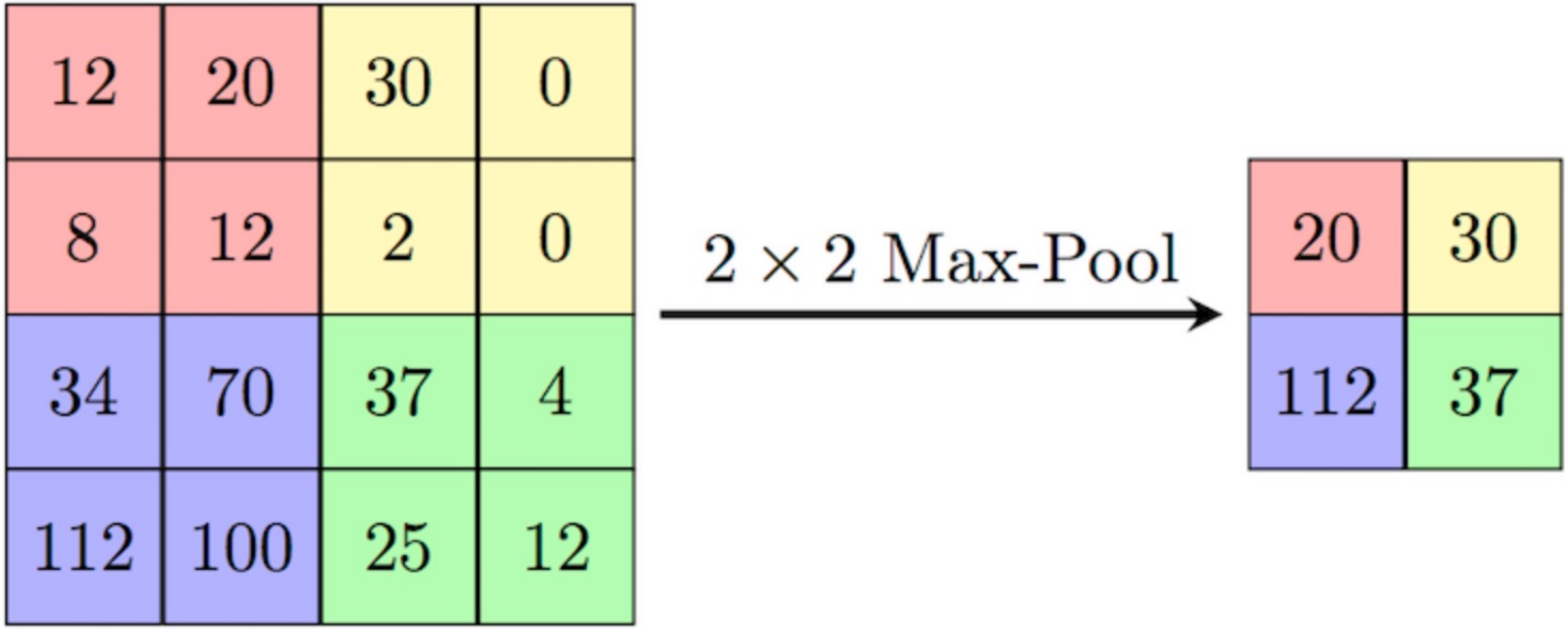


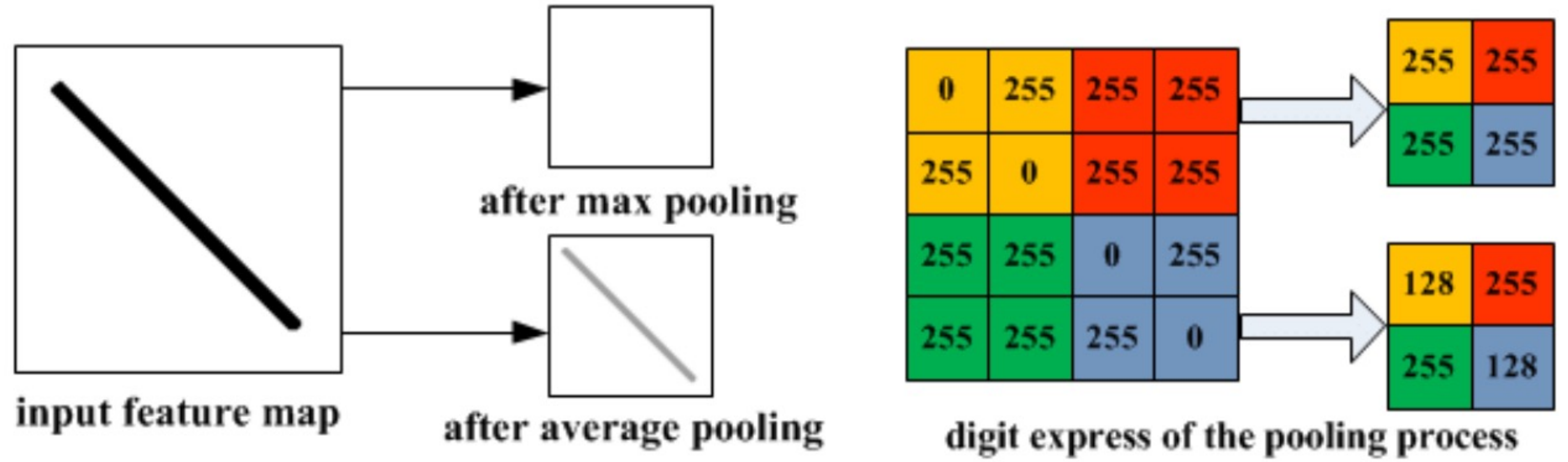
Note: In convolutional neural networks, the filters are learned the same way as hyperparameters through backpropagation during the training process.

Max-Pooling : Largest number in an array

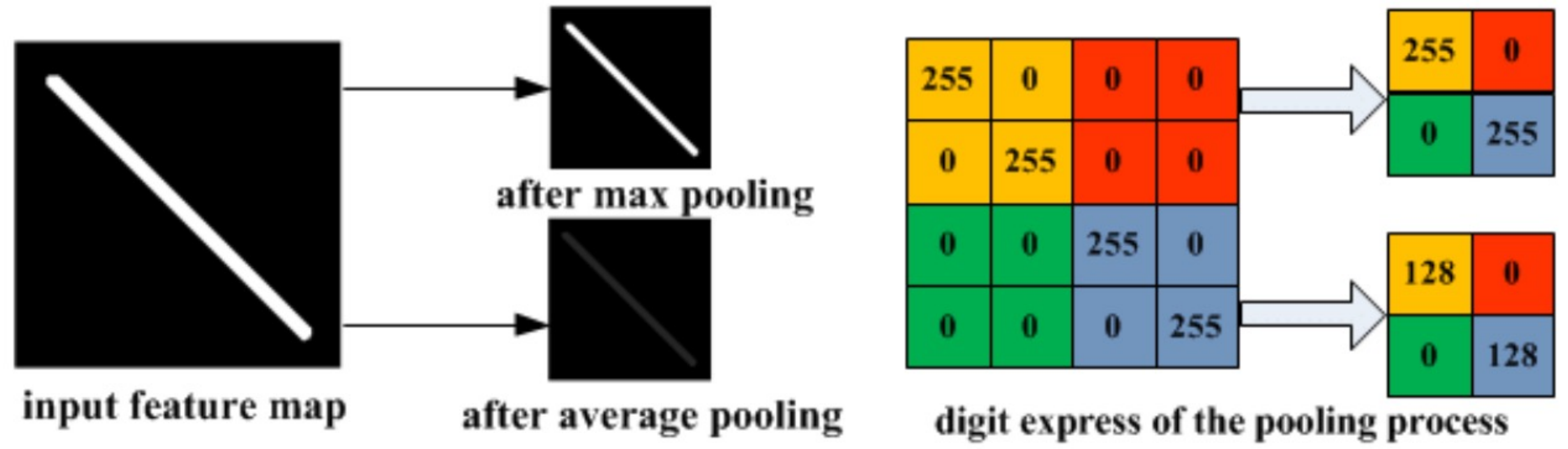
- Max Pooling selects the largest number in an area of the previous layer
- This reduces the amount of computational effort required in later layers
- Also reduces the tendency to over-fit



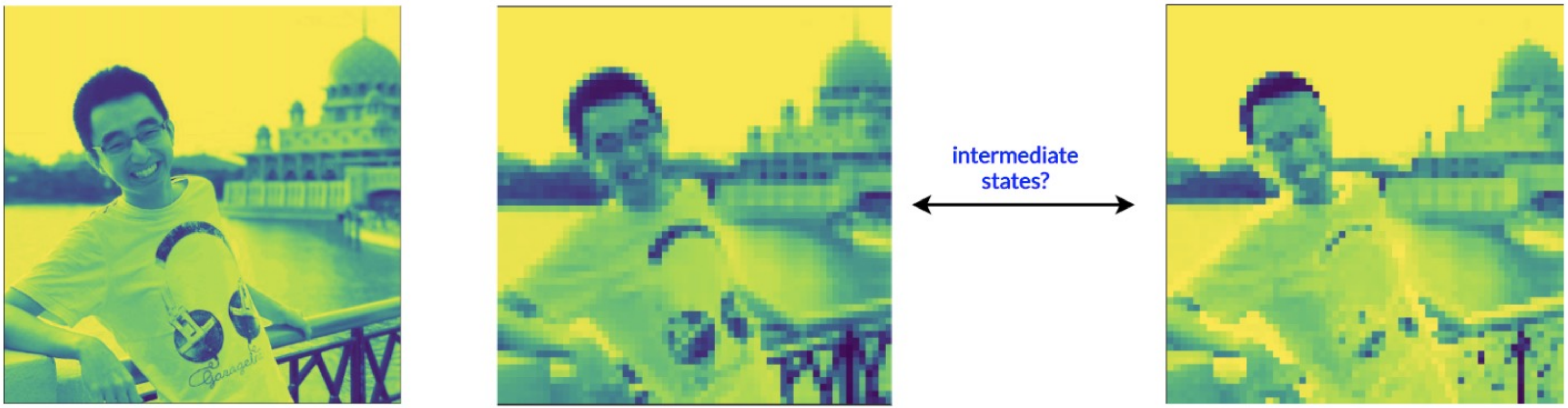

12 20 30 0
8 12 2 0
34 70 37 4
112 100 25 12
2 × 2 Max-Pool
20 30
112 37



(a) Illustration of max pooling drawback



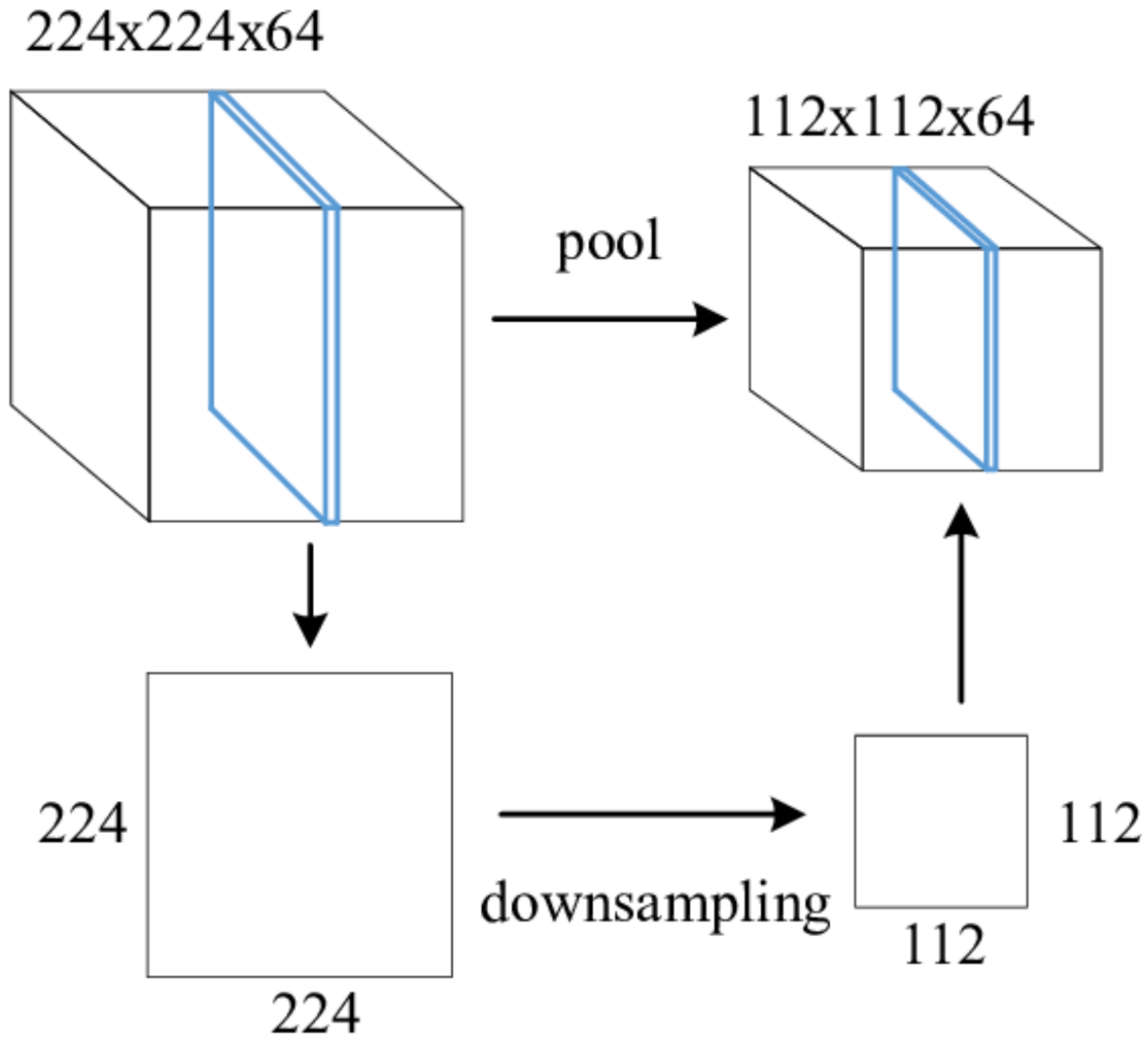
(b) Illustration of average pooling drawback



original image

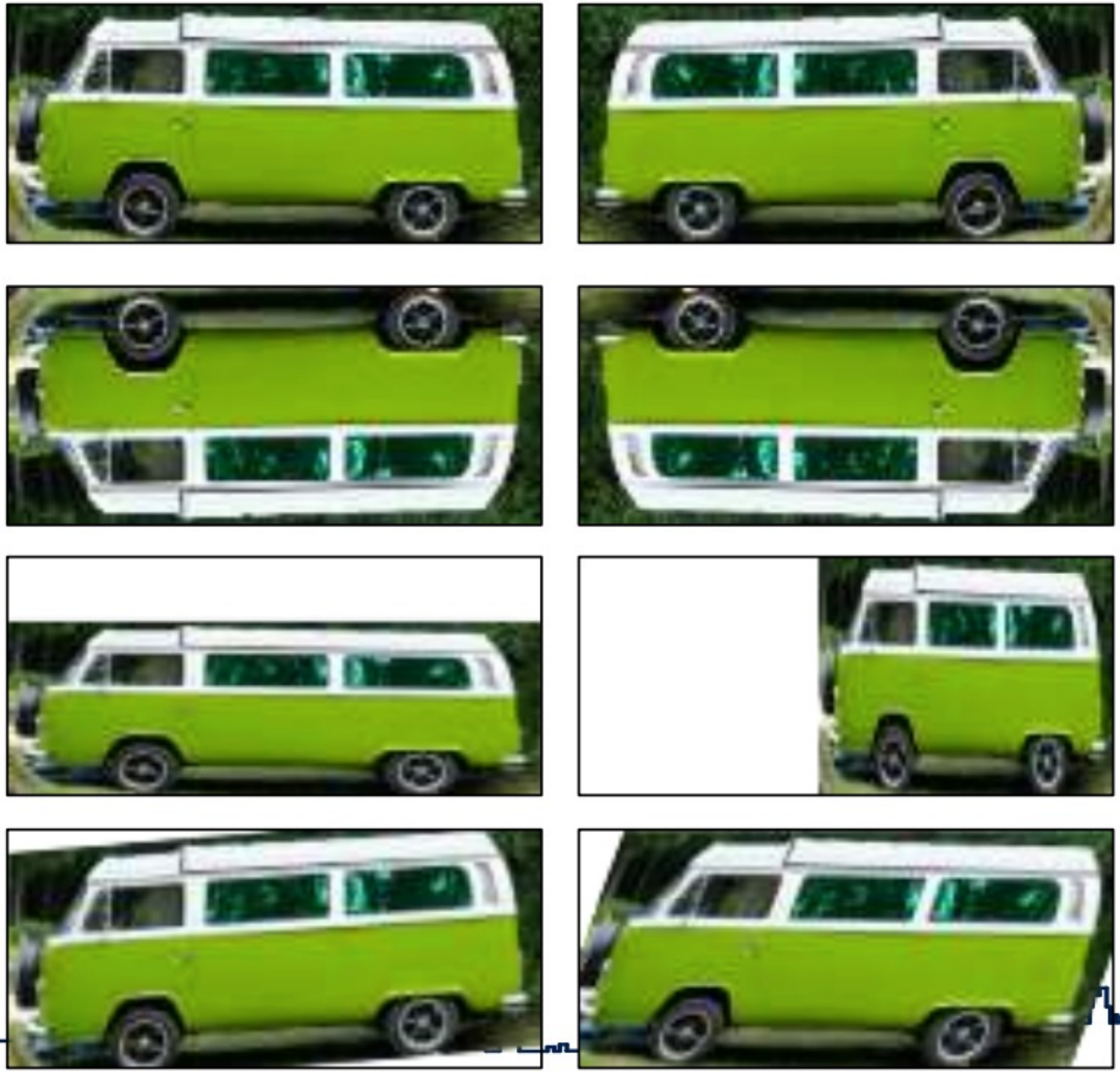
average pooling

max pooling


224x224x64
112x112x64
pool
224
224
downsampling
112
112

Augmenting Data

- If we have a shortage of data .. When can sometimes ‘augment’ it
- That is, systematically transform it
- This can also help to build models that generalise better



Architectures for Convolutional Neural Network

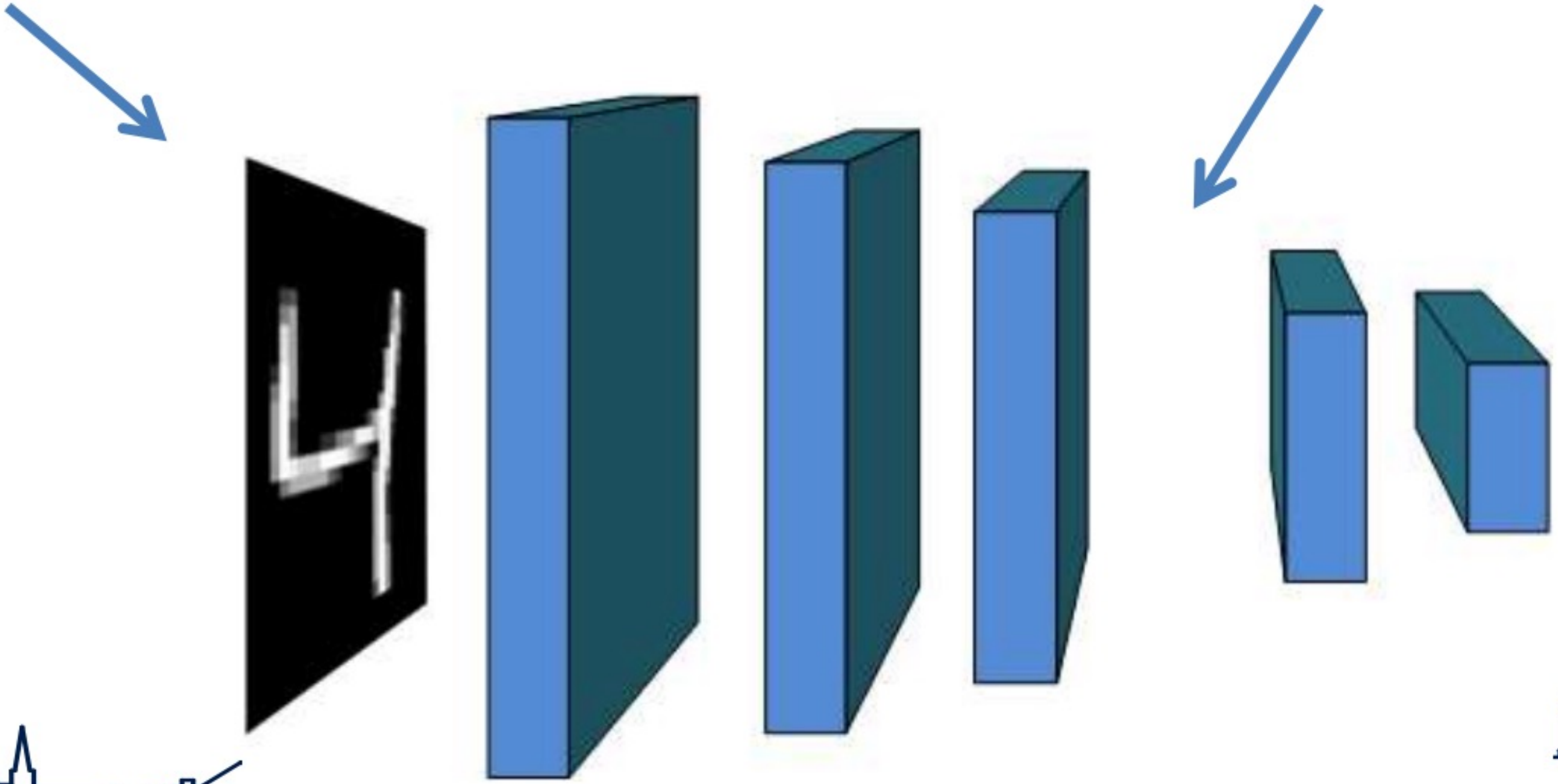
Input data

In this case a 28x28 pixel, grey-scale image of the number 4

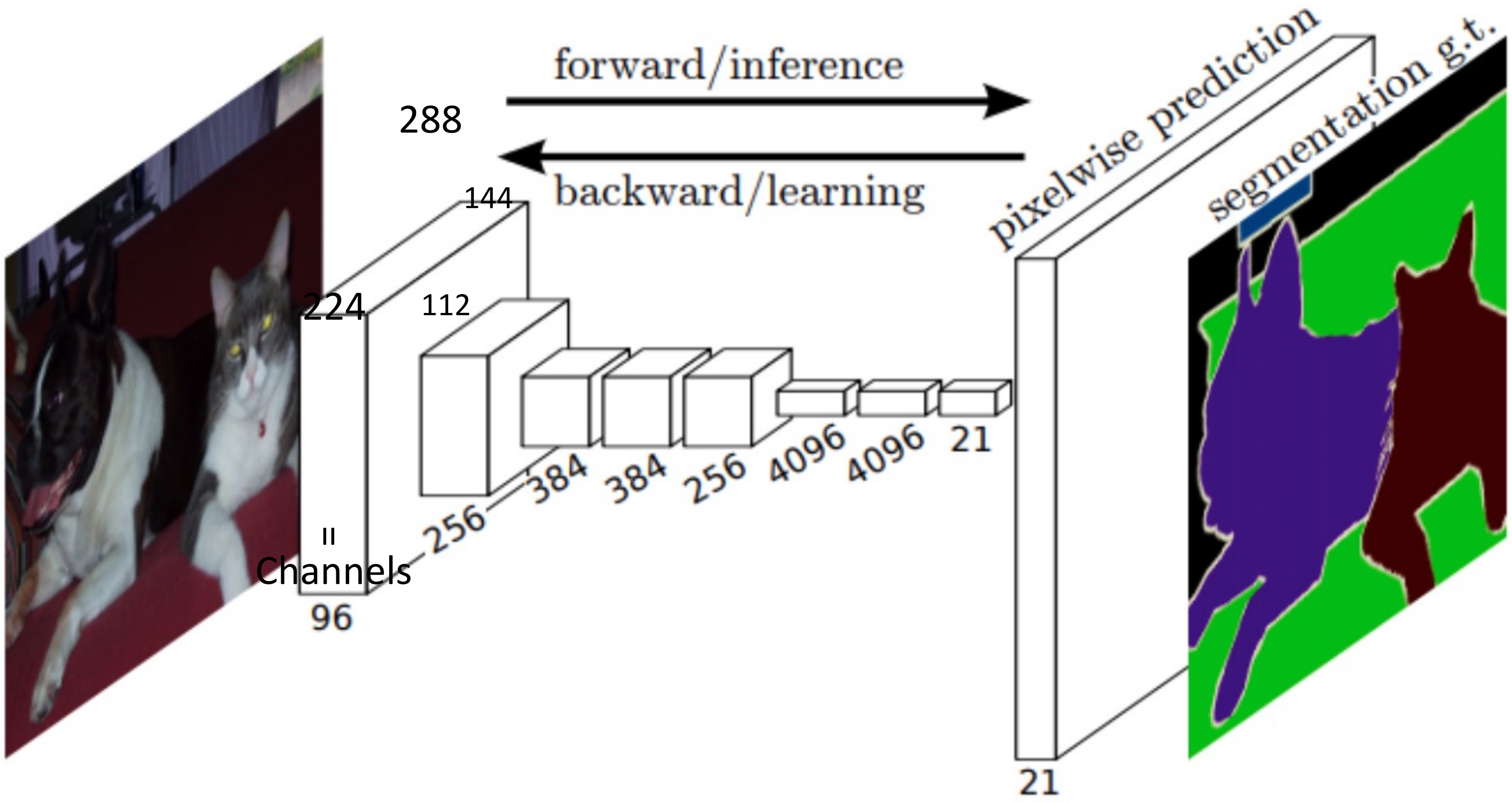
Convolutional Neural Network

A Series of layers ..

Each layer might be a convolution, a densely connected array of ‘neurons’, a max-pool or other types of filter / processing

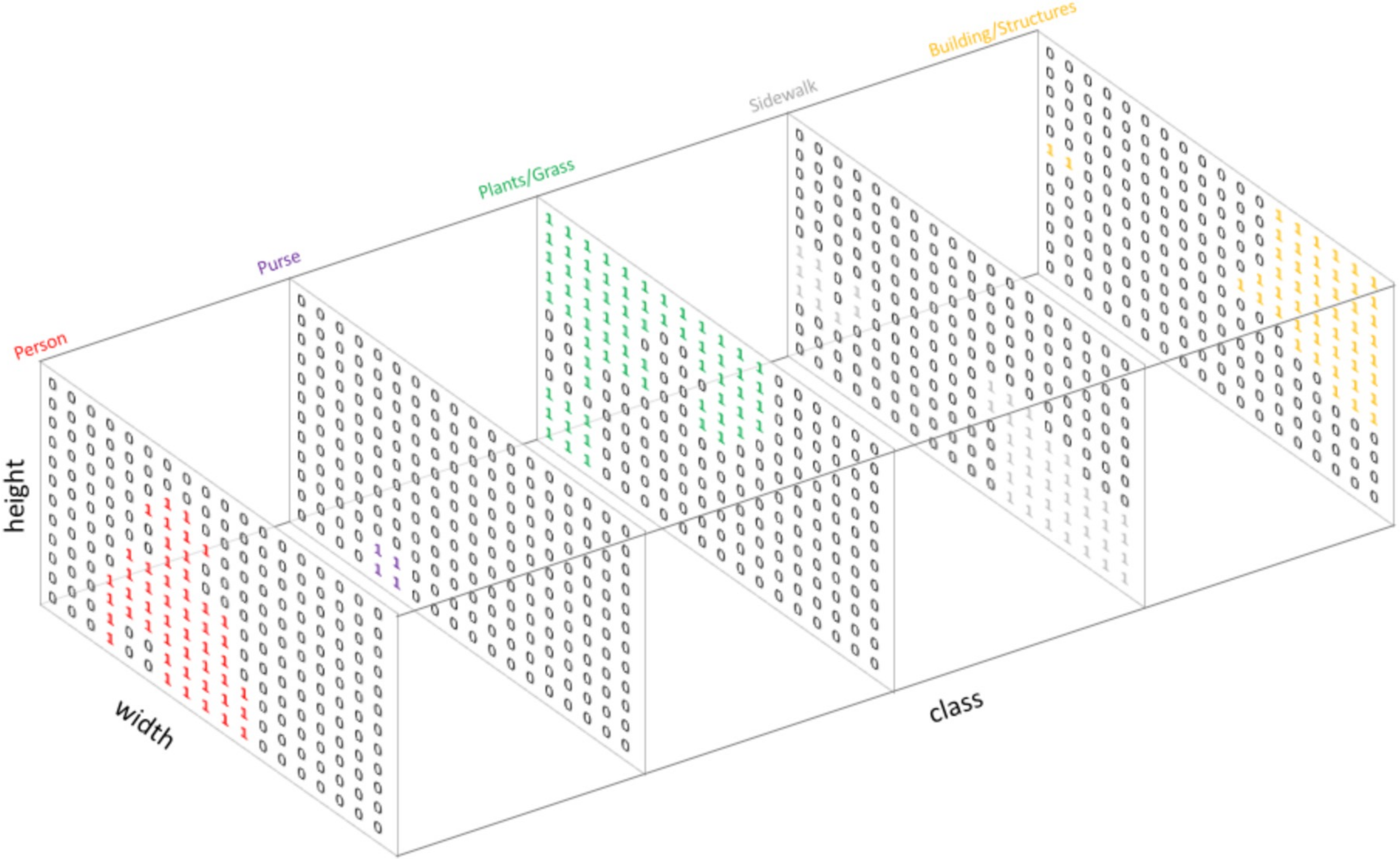


Segmentation



- **What do you see happen to an image with its size and depth?**

Make network deep and *work at a lower spatial resolution* for many of the layers.



Person
Purse
Plants/Grass
Sidewalk
Building/Structures
height
width
class

Representing the task

Simply, our goal is to take either a RGB color image ($height \times width \times 3$) or a grayscale image ($height \times width \times 1$) and output a segmentation map where each pixel contains a class label represented as an integer ($height \times width \times 1$).

segmented

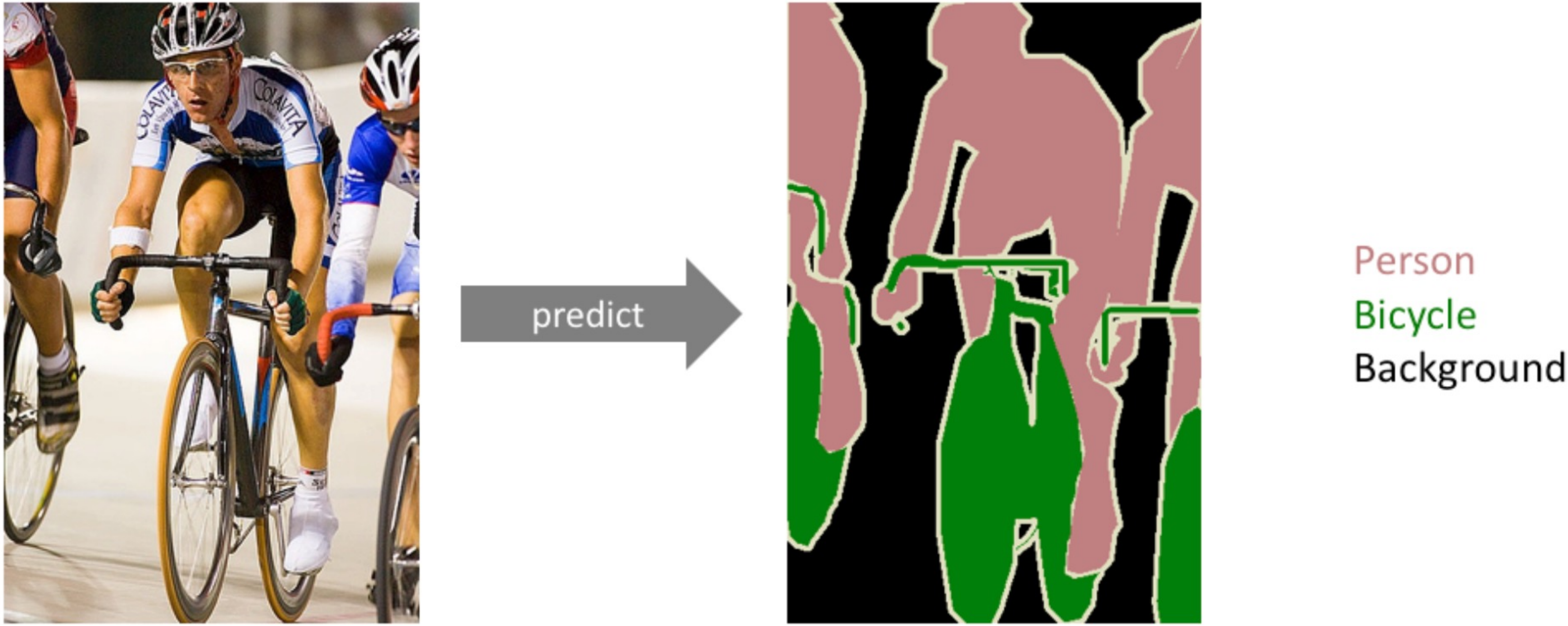
1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures

3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	3	5	5	5	5	5	5
3	3	3	3	3	1	1	1	1	3	3	3	5	5	5	5	5	5
3	3	3	3	3	3	1	1	3	3	3	5	5	5	5	5	5	5
5	5	3	3	3	3	1	1	3	3	5	5	5	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	4	4	4	4	4	5	5	5
4	4	4	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	4	4	4	4	4	4	4

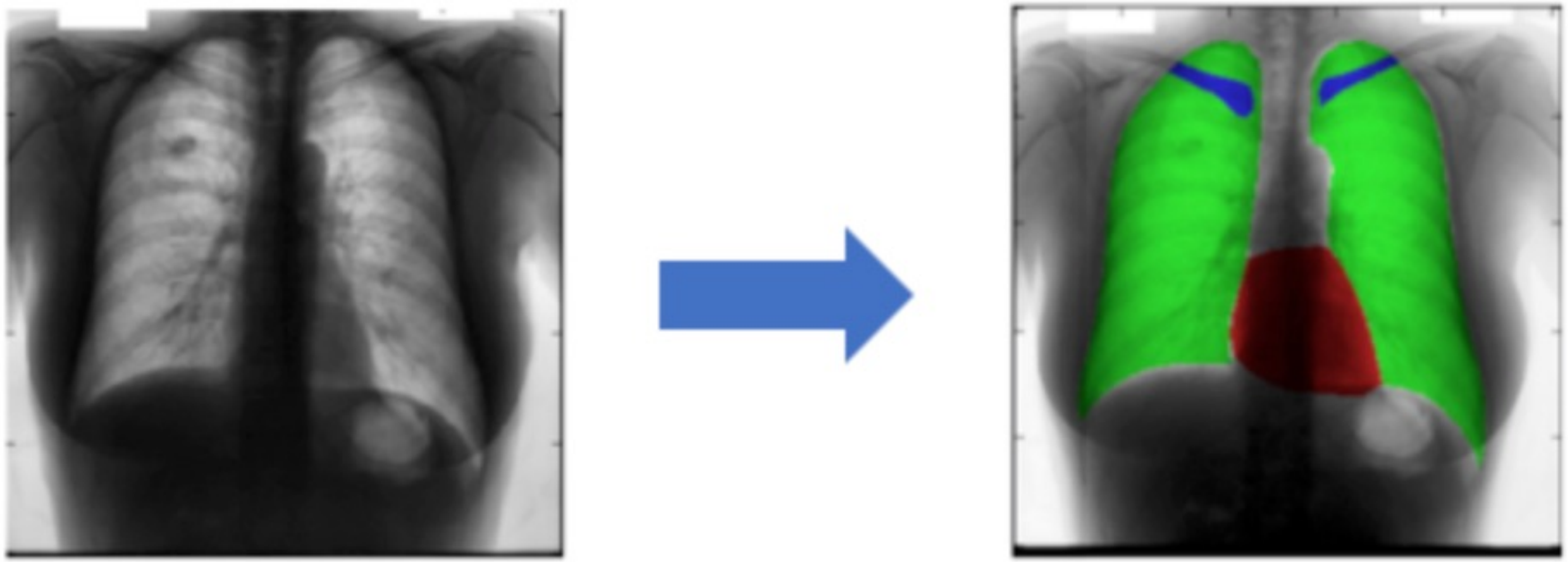
Input

Semantic Labels

0: Background/Unknown
1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures



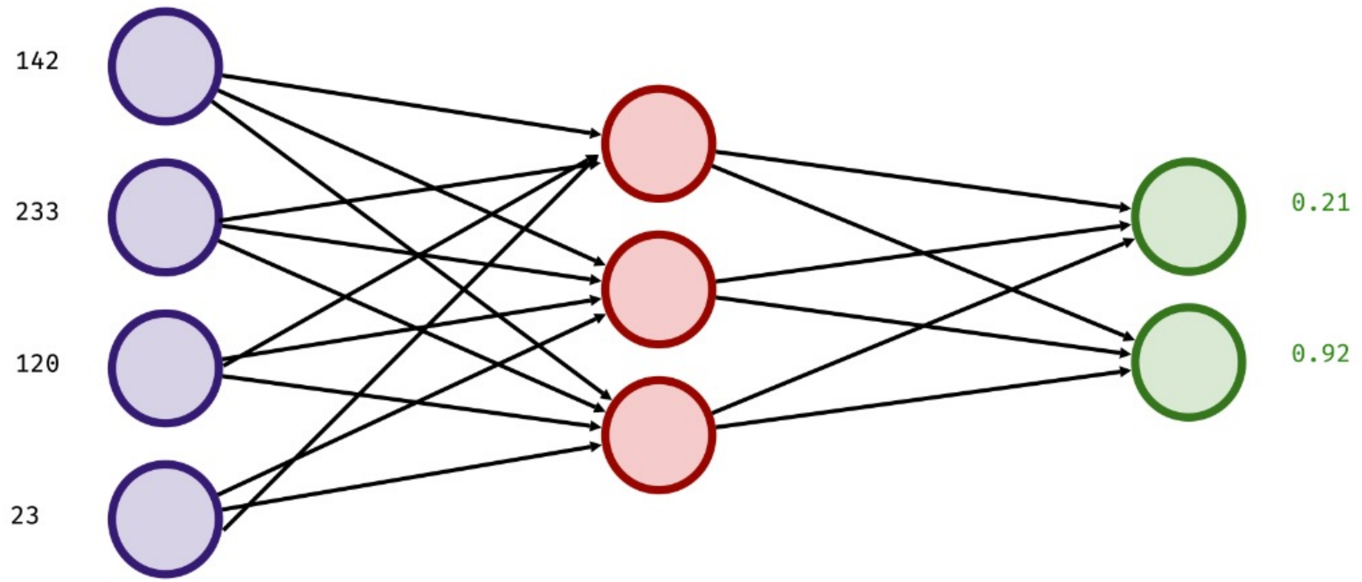
An example of semantic segmentation, where the goal is to predict class labels for each pixel in the image. (Source)



Input Image

Segmented Image

Densely Connected layers or Fully-connected layers



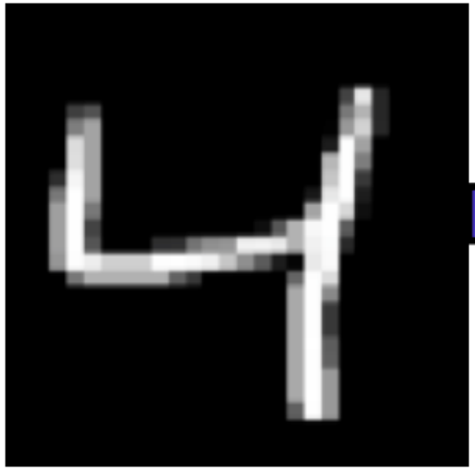
Output = Activation(dot(input, kernel) + bias)

- A layer that is deeply connected with its preceding layer
- Neurons of the layer are connected to every neuron of its preceding layer
- A neuron changes if any of the neurons from the preceding layer changed
- Most common layer used in ANNs

More Complex Inputs: Grey-scale Images

28 Pixels

28 Pixels



2D Convolution Layer

Applies a series of filters (convolutions) to detect features in the input set such

For example:
Apply 32 different filters each 3x3

Densely connected Layer

Creates and connects many neurons

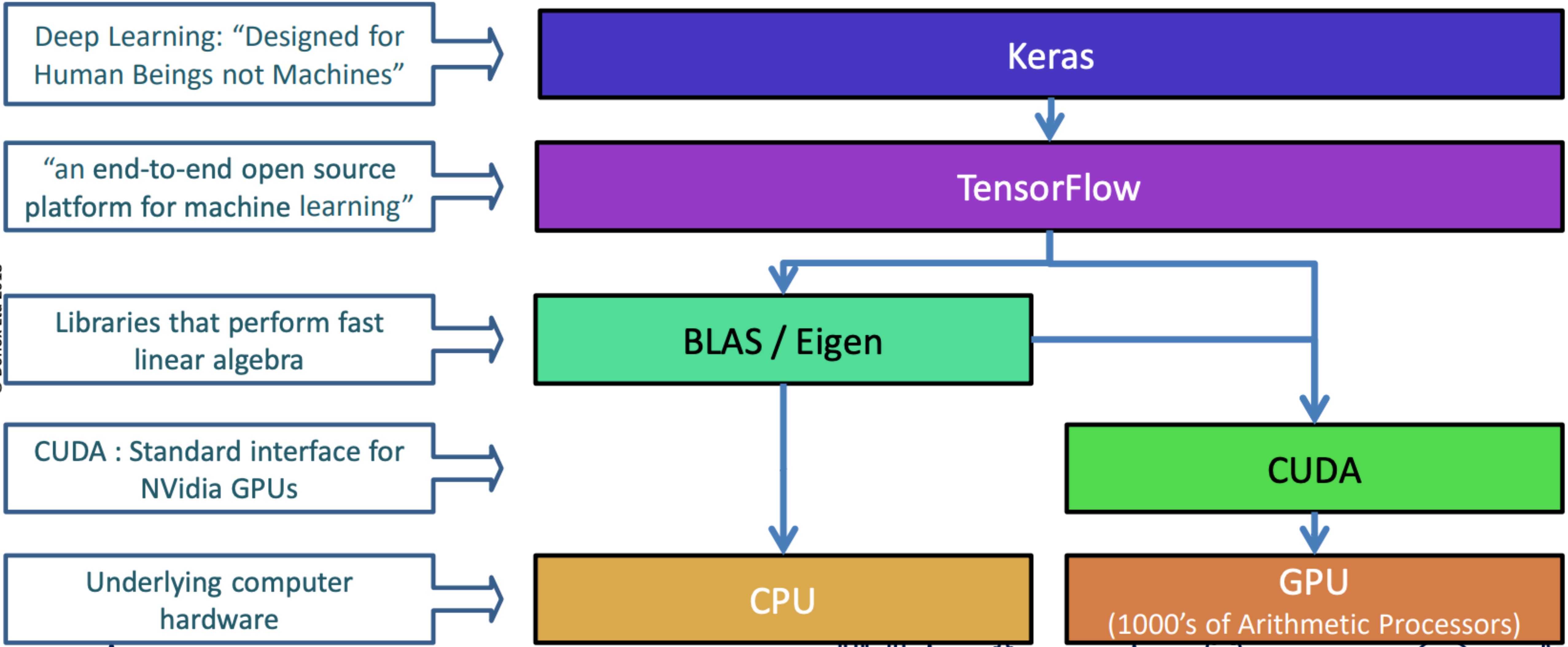
```
model.add(
    keras.Input(
        shape=(28, 28)
    )
)
```

```
model.add(
    layers.Conv2D(32, (3,3))
)
```

```
model.add(
    Dense(100,
        activation='relu‘)
)
```

Libraries to Create and Fit Neural Networks

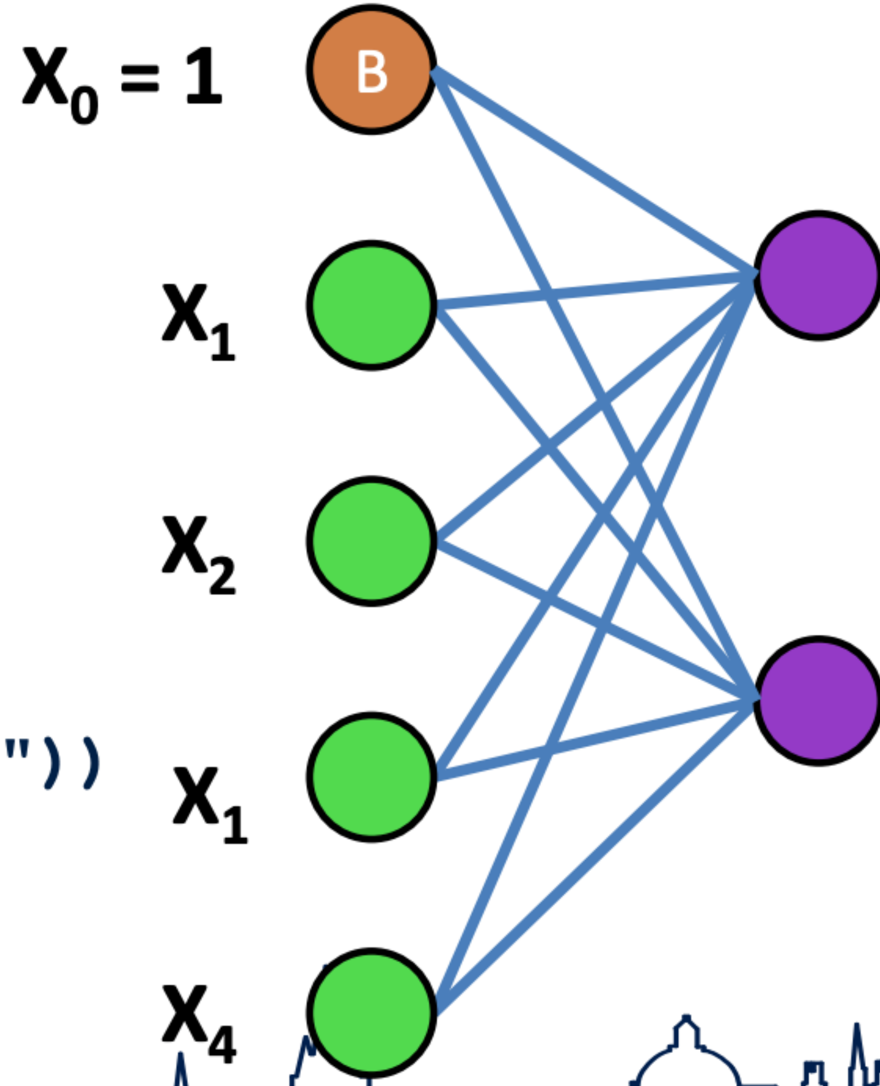
One of Several Deep Learning Library Stacks



Keras Enables CNNs to be built quickly

```
import tensorflow as tf from tensorflow
import keras from tensorflow.keras
import layers

model = keras.Sequential()
model.add(keras.Input(shape=(4,)))
model.add(layers.Dense(2, activation="relu"))
```



See: https://keras.io/guides/sequential_model

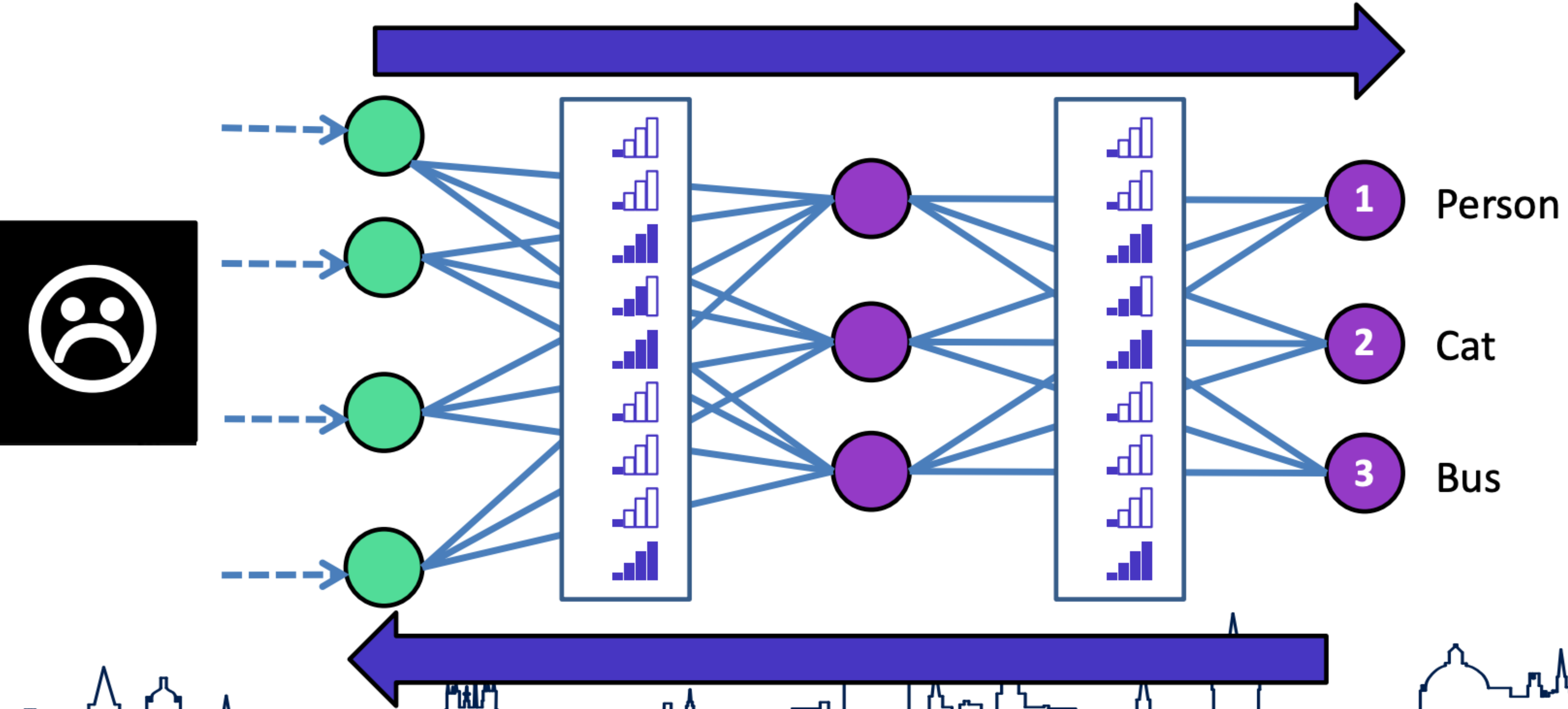
Pre-trained Neural Networks

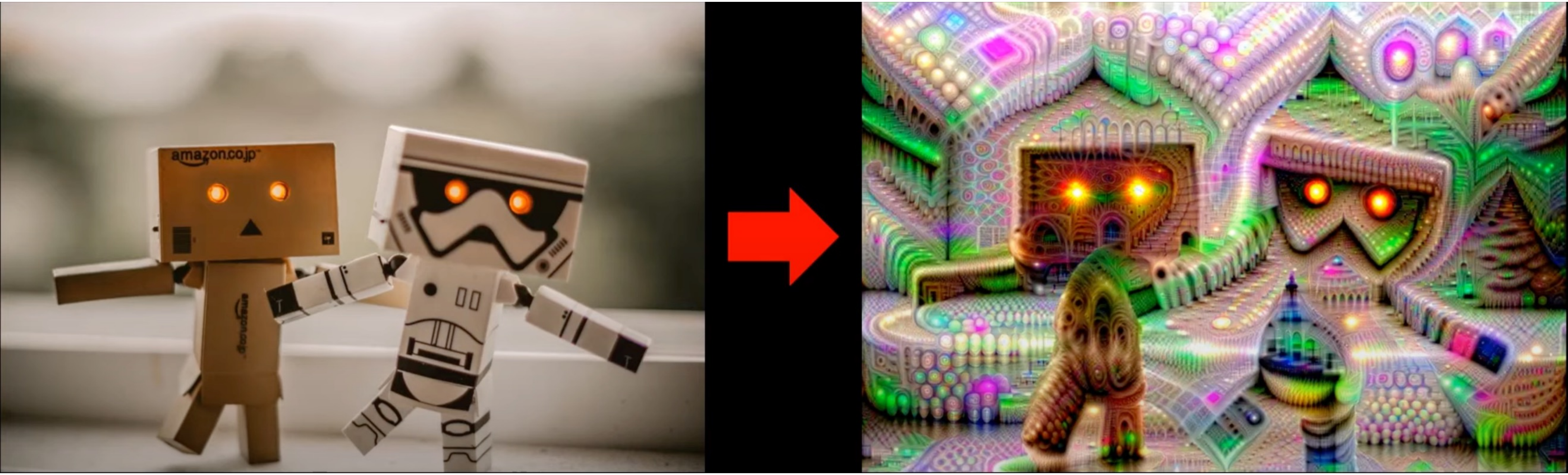
Pre-trained Networks for Image Classification

Demos: W2-02-04, W2-02-05

Gaining insights into CNN Structure using Google DeepDream

Use a Pre-trained Neural Network to Modify the Image




amazon.co.jp


DPL 292T

References

- Tara Abraham (2016) “Rebel Genius: Warren S. McCulloch's Transdisciplinary Life in Science”, MIT Press
- How to read the notation in Russel and Whitehead’s “Principia Mathematica”: https://plato.stanford.edu/entries/pm-notation/
- Lefkowitz, M. (2019) “Professor’s perceptron paved the way for AI – 60 years too soon”
  - https://news.cornell.edu/stories/2019/09/professors-perceptron-paved-way-ai-60-years-too-soon
- Advantages and disadvantages of various activation Functions
  - https://medium.com/analytics-vidhya/activation-functions-all-you-need-to-know-355a850d025e
- Eigen library for Linear Algebra: https://eigen.tuxfamily.org/index.php?title=Main_Page
- BLAS (Basic Linear Algebra Sub-programes) :
  - https://en.wikipedia.org/wiki/Basic_Linear_Algebra_Subprograms
- Building sequential models in Keras: https://keras.io/guides/sequential_model/
- Website used to generate the deepdream images of Rob’s VW Camper:
  - https://deepdreamgenerator.com/
- Who invented back-propagation?:
  - https://medium.com/syncedreview/who-invented-backpropagation-hinton-says-he-didnt-but-his-work-made-it-popular-e0854504d6d1

References

- On the controversy of Minsky and Papert's book on Perceptrons:
  - https://en.wikipedia.org/wiki/Perceptrons_(book)