Deep Learning

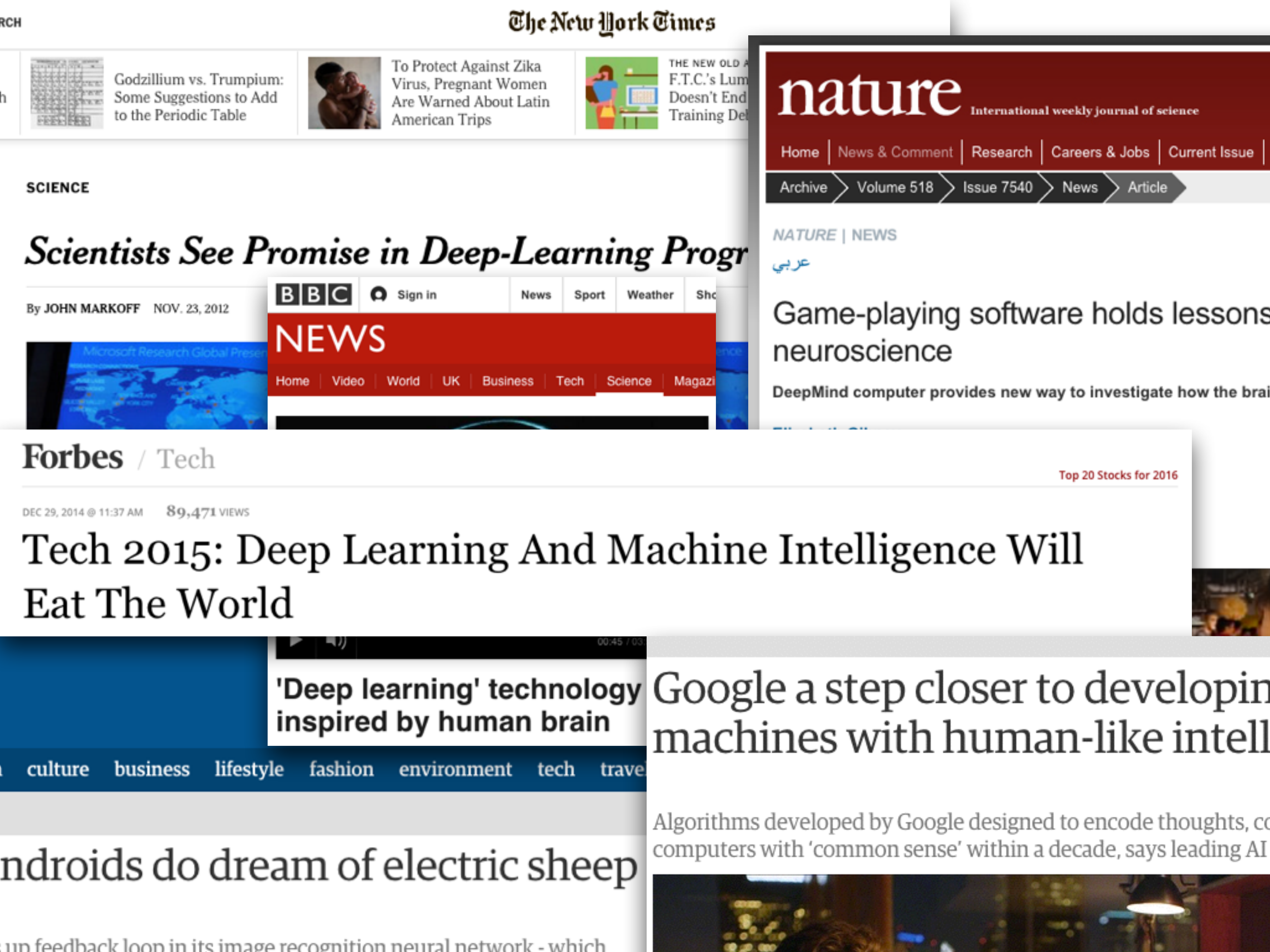
Theory, History, State of the Art & Practical Tools

by Ilya Kuzovkin
ilya.kuzovkin@gmail.com

COMPUTATIONAL NEUROSCIENCE
Research group at the University of Tartu

http://neuro.cs.ut.ee

Machine Learning Estonia
2016



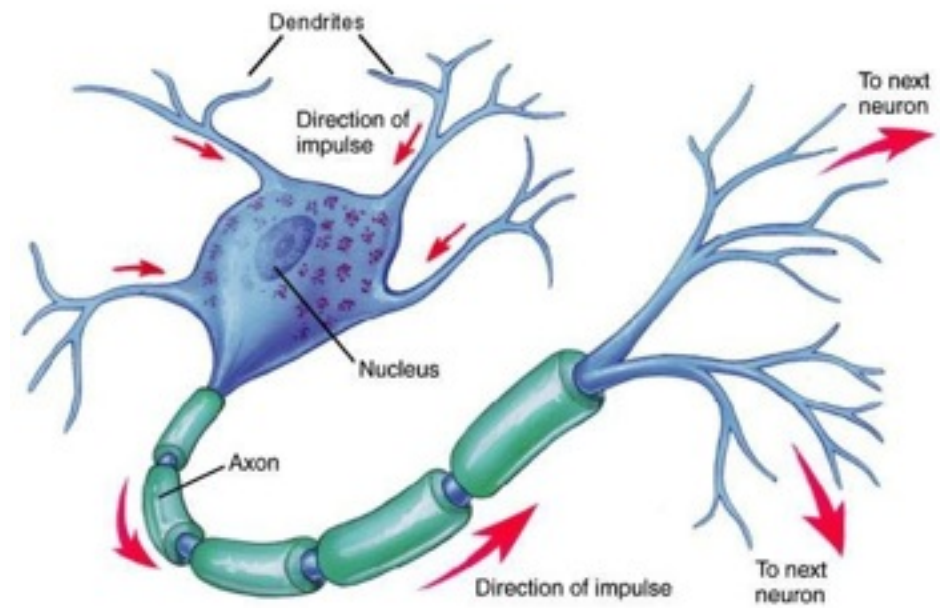
The New York Times
Godzillium vs. Trumpium: Some Suggestions to Add to the Periodic Table
To Protect Against Zika Virus, Pregnant Women Are Warned About Latin American Trips
THE NEW OLD A
F.T.C.'s Lum
Doesn't End
Training De
SCIENCE
Scientists See Promise in Deep-Learning Progr
By JOHN MARKOFF NOV. 23, 2012
nature International weekly journal of science
Home | News & Comment | Research | Careers & Jobs | Current Issue
Archive Volume 518 Issue 7540 News Article
NATURE | NEWS
عربي
Game-playing software holds lessons
neuroscience
DeepMind computer provides new way to investigate how the brai
BBC Sign in News Sport Weather Sho
NEWS
Home Video World UK Business Tech Science Magazi
Forbes / Tech
Top 20 Stocks for 2016
DEC 29, 2014 @ 11:37 AM 89,471 VIEWS
Tech 2015: Deep Learning And Machine Intelligence Will Eat The World
'Deep learning' technology inspired by human brain
culture business lifestyle fashion environment tech trave
ndroids do dream of electric sheep
up feedback loop in its image recognition neural network - which
Google a step closer to developin
machines with human-like intell
Algorithms developed by Google designed to encode thoughts, co
computers with 'common sense' within a decade, says leading AI

- Where it has started
- How it learns
- How it evolved
- What is the state now
- How can **you** use it

Where it has started

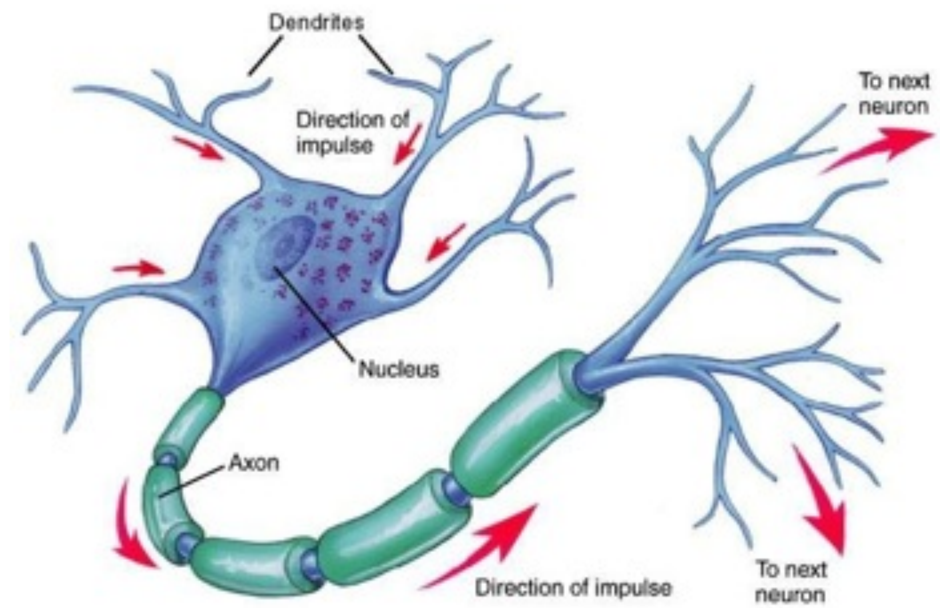
Where it has started

Artificial Neuron



Where it has started

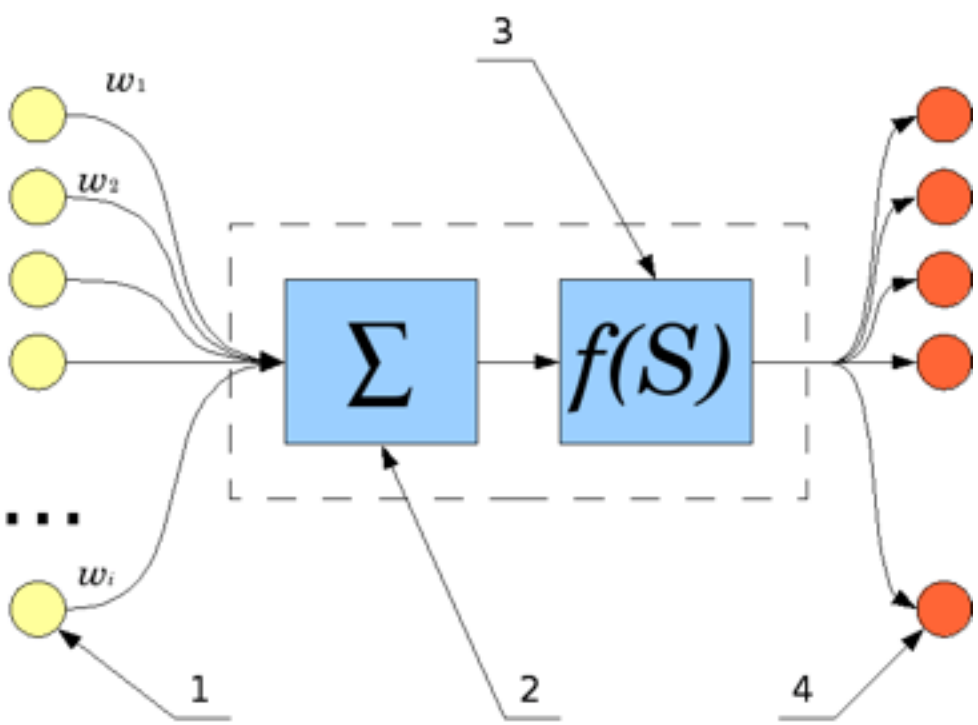
Artificial Neuron



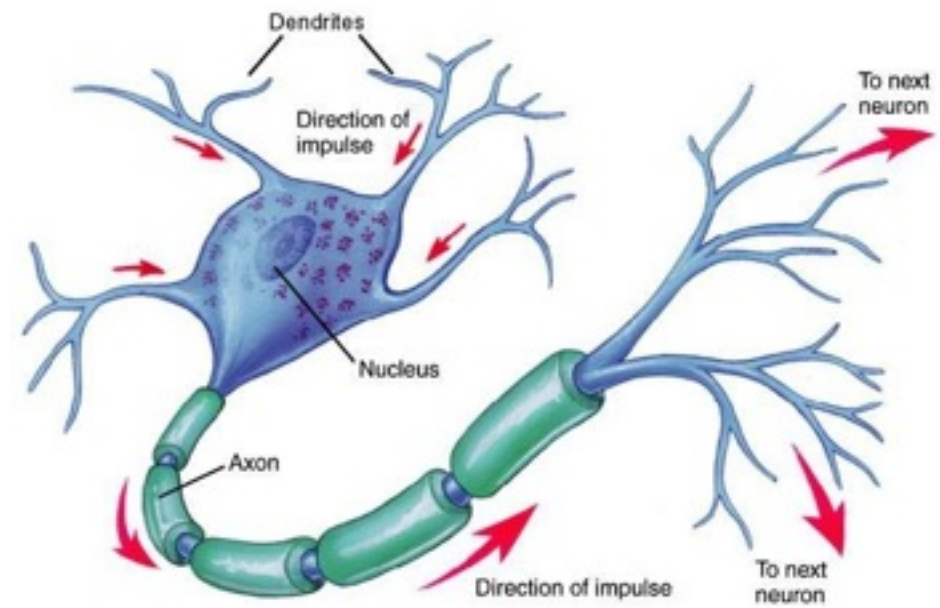
1943

McCulloch and Pitts

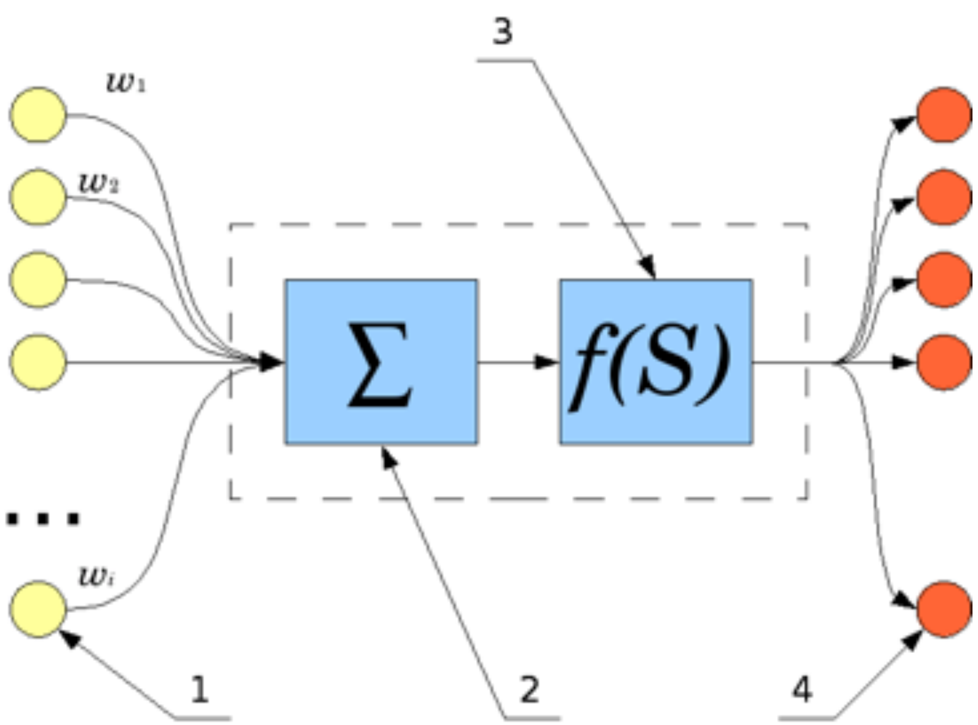
“A Logical Calculus of the Ideas Immanent in Nervous Activity”

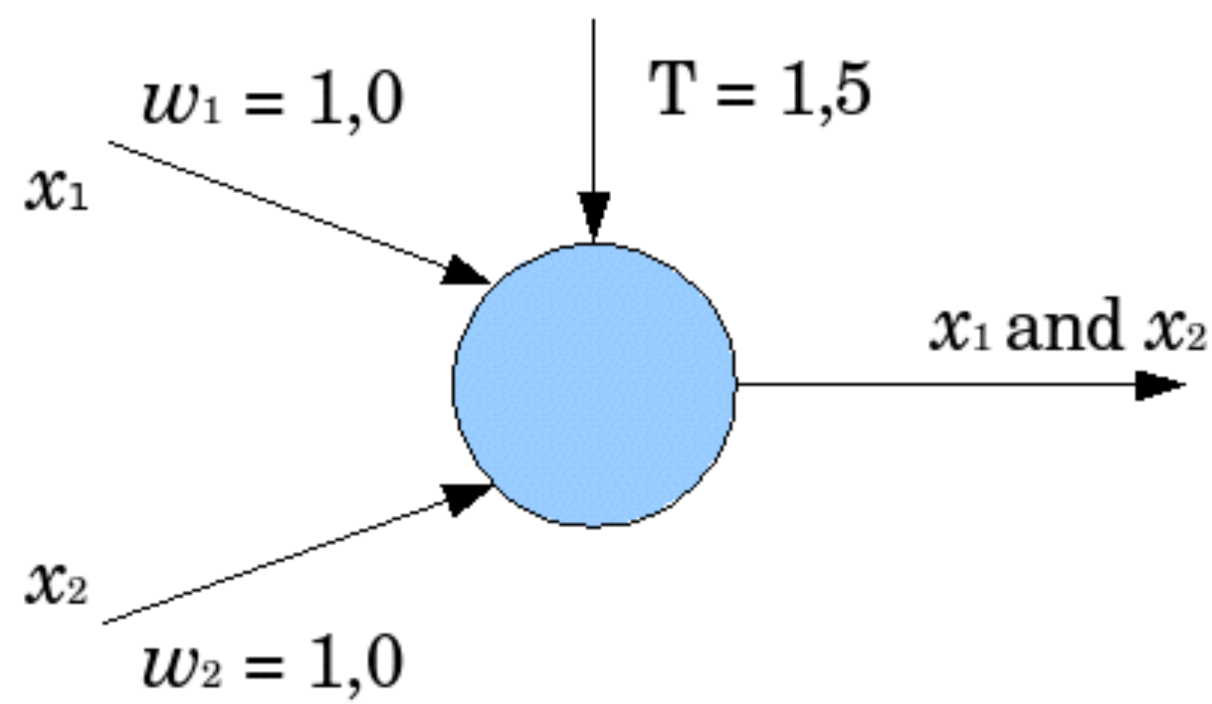


Where it has started
Artificial Neuron



1943
McCulloch and Pitts
"A Logical Calculus of the Ideas Immanent in Nervous Activity"





Where it has started

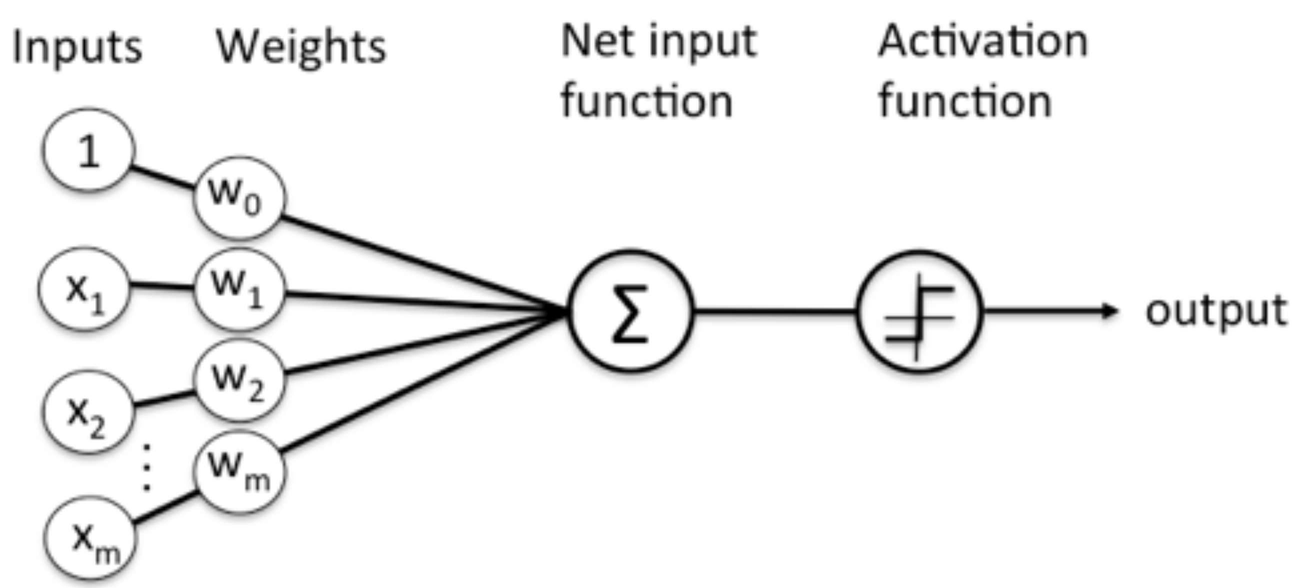
Perceptron

1957 **Frank Rosenblatt**

Where it has started

Perceptron

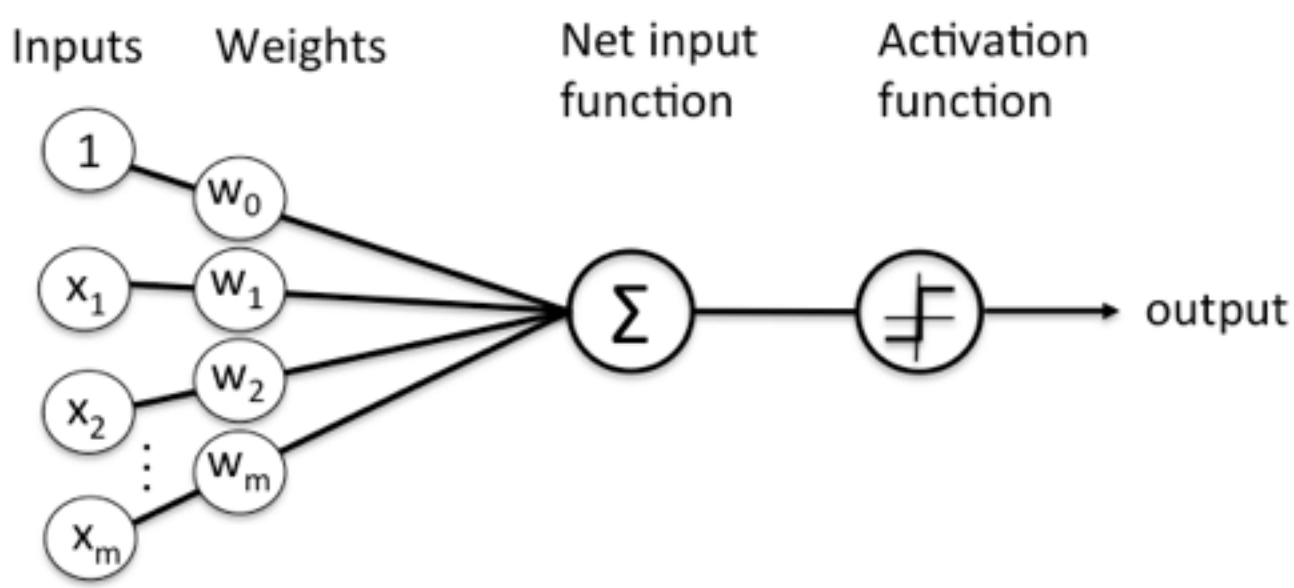
1957 **Frank Rosenblatt**

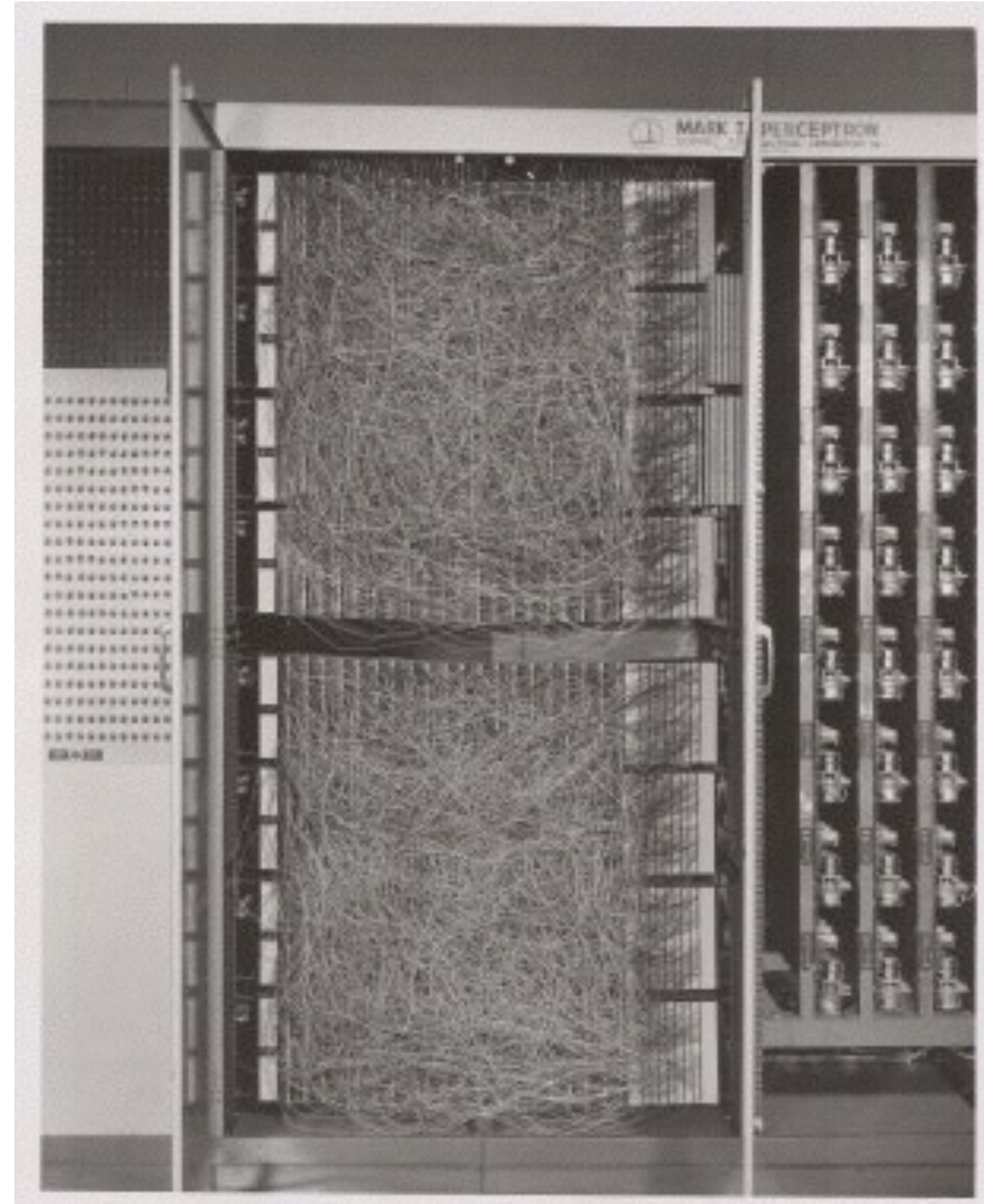


Where it has started

Perceptron

1957 **Frank Rosenblatt**

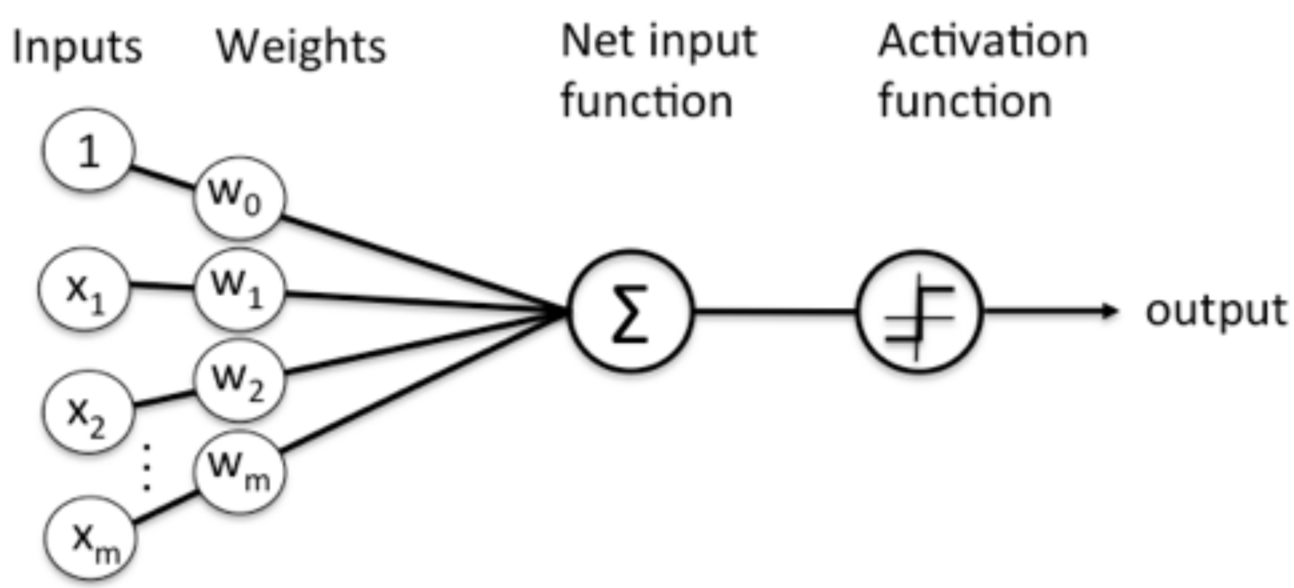


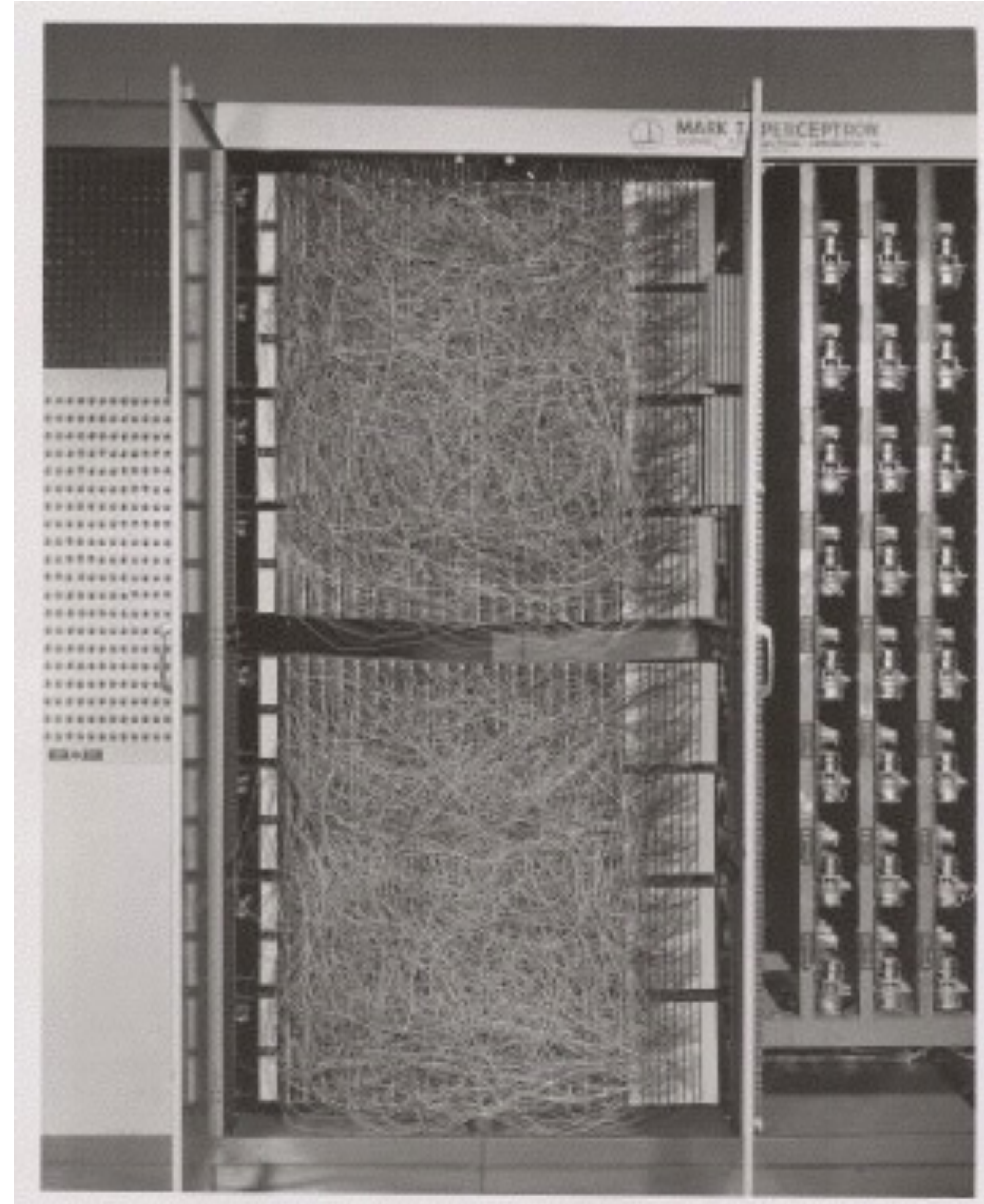


Where it has started

Perceptron

1957 **Frank Rosenblatt**





"[The Perceptron is] the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence."

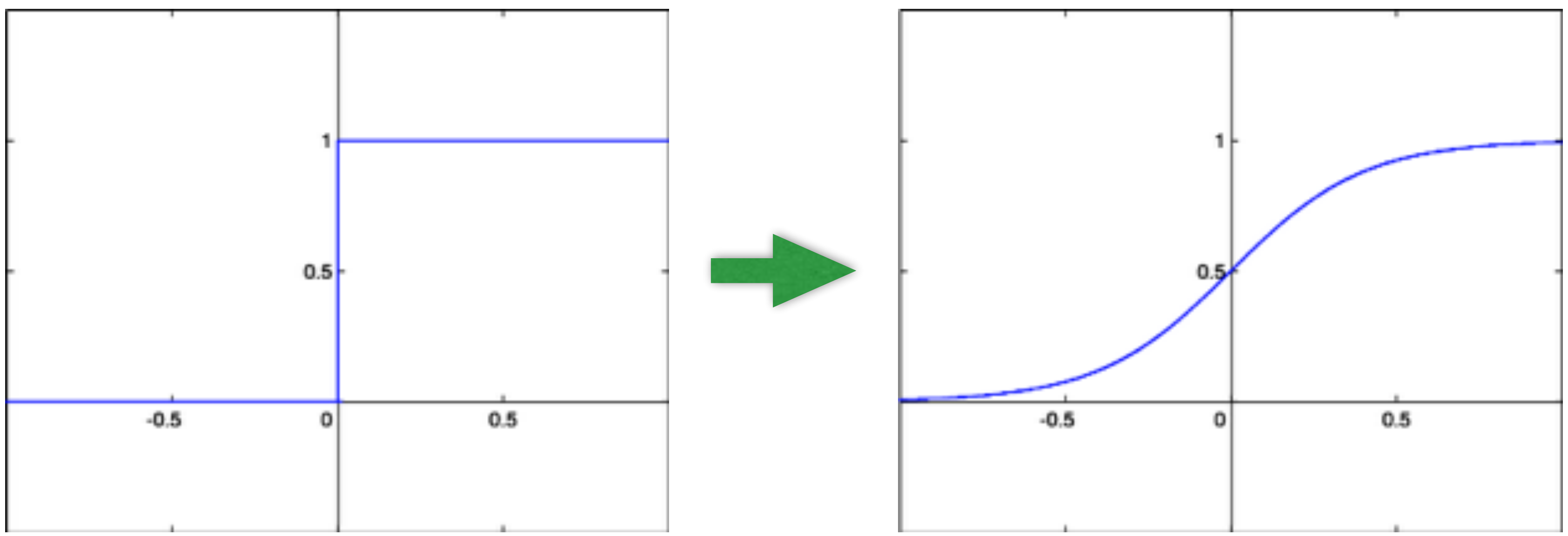
THE NEW YORK TIMES

Where it has started

Sigmoid & Backpropagation

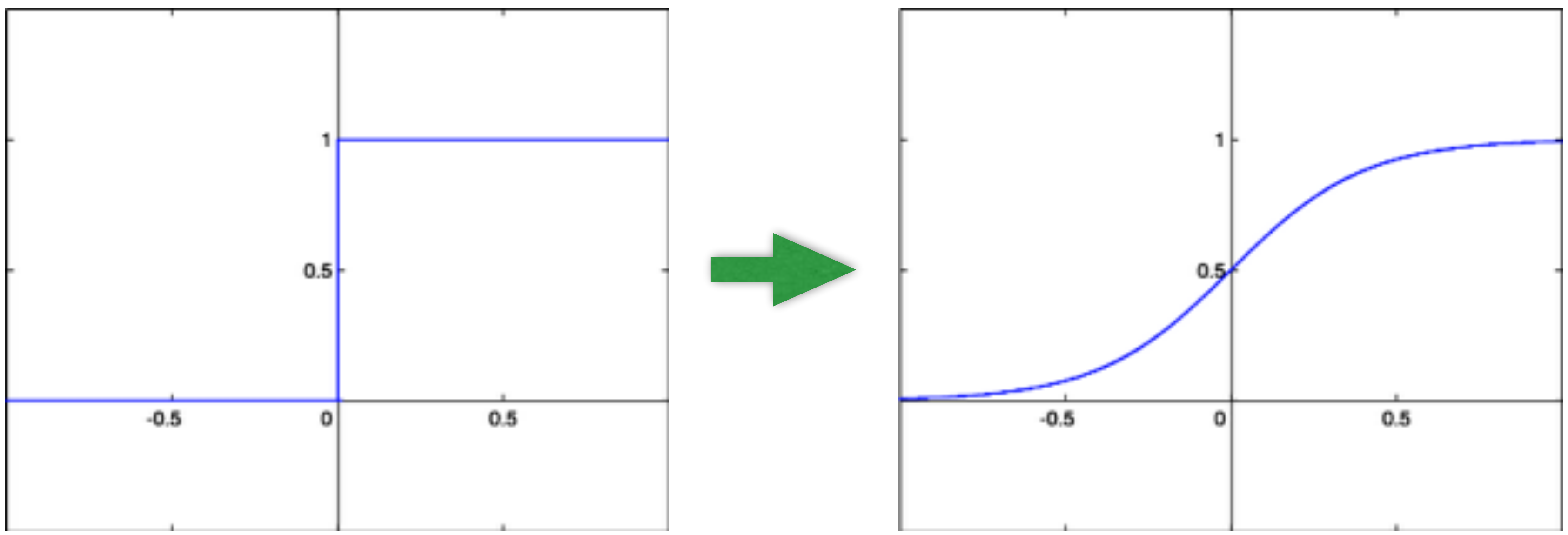
Where it has started

Sigmoid & Backpropagation



Where it has started

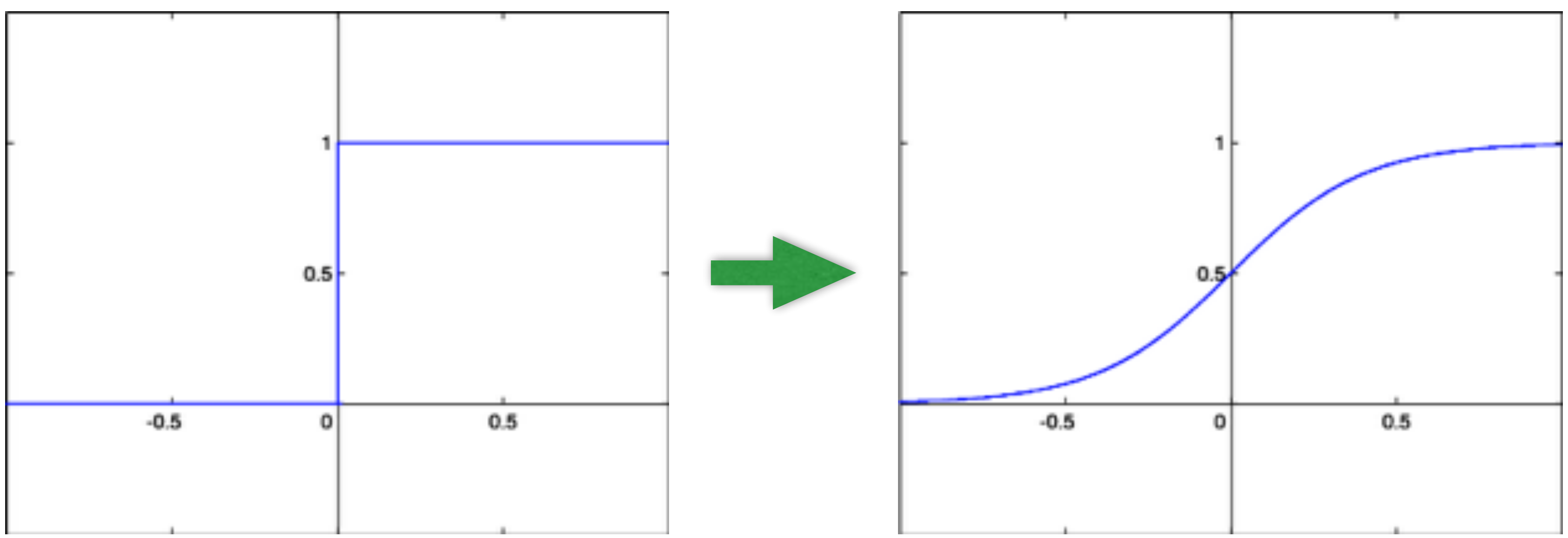
Sigmoid & Backpropagation



- Measure how **small changes** in weights **affect** output
- Can apply NN to **regression**

Where it has started

Sigmoid & Backpropagation



- Measure how **small changes** in weights **affect** output
- Can apply NN to **regression**

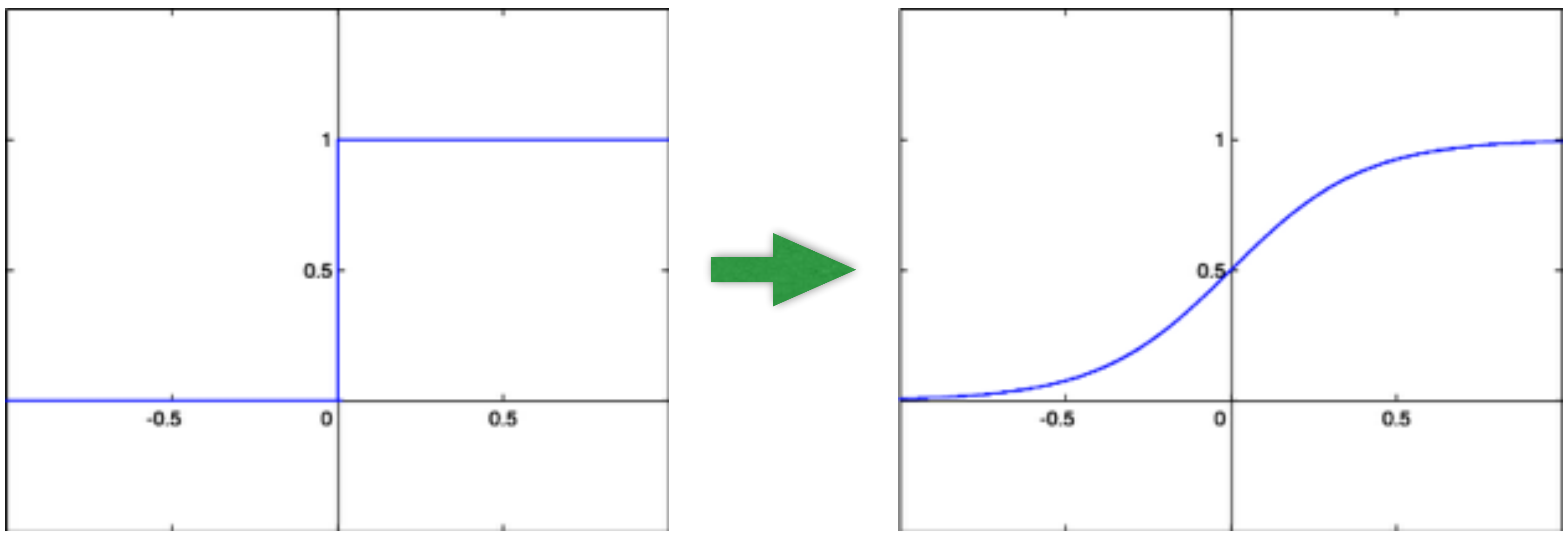
(1974) 1986

(Werbos) Rumelhart, Hinton, Williams

“Learning representations by back-propagating errors” (Nature)

Where it has started

Sigmoid & Backpropagation



- Measure how **small changes** in weights **affect** output
- Can apply NN to **regression**

(1974) 1986

(Werbos) Rumelhart, Hinton, Williams

"Learning representations by back-propagating errors" (Nature)

- **Multilayer** neural networks, etc.

Where it has started

Why DL revolution did not happen in 1986?

FROM A TALK BY GEOFFREY HINTON

Where it has started

Why DL revolution did not happen in 1986?

- **Not enough data** (datasets were 1000 times too small)

- **Computers were too slow** (1,000,000 times)

FROM A TALK BY GEOFFREY HINTON

Where it has started

Why DL revolution did not happen in 1986?

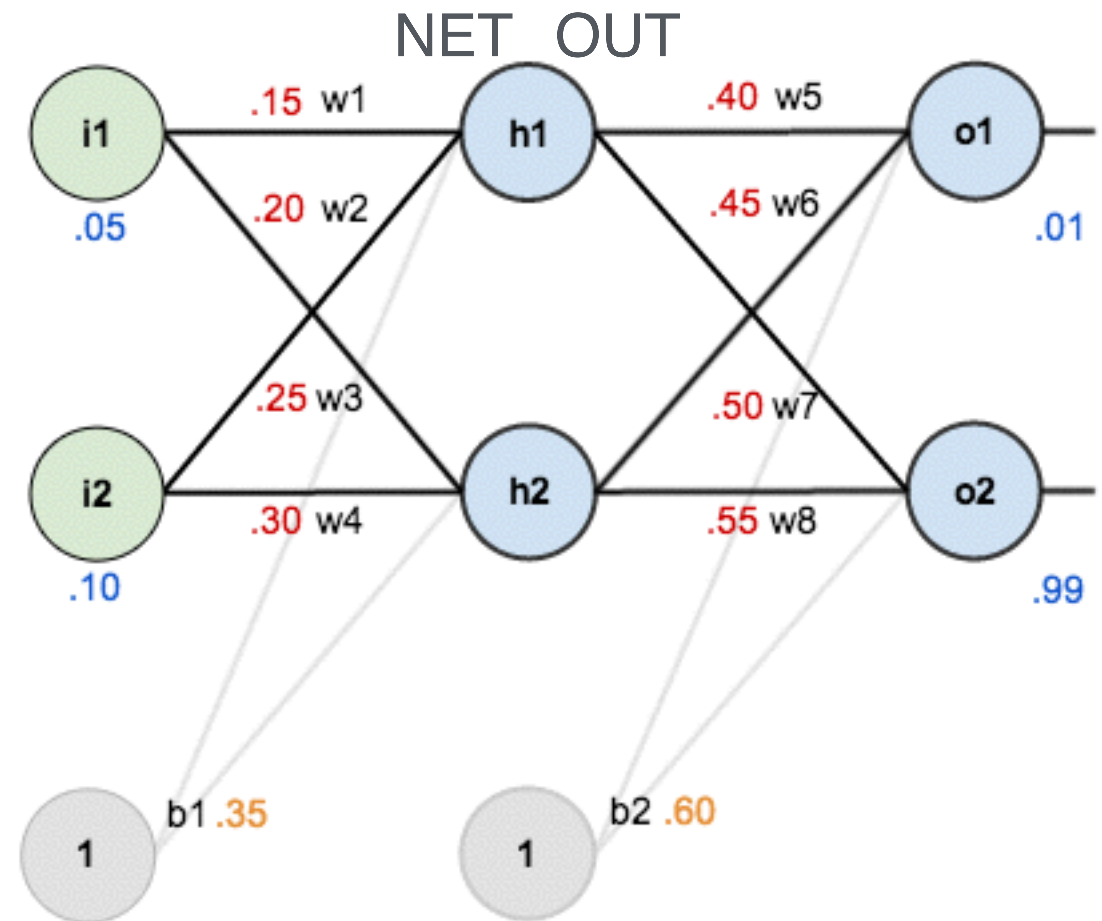
- **Not enough data** (datasets were 1000 times too small)
- **Computers were too slow** (1,000,000 times)
- Not enough attention to network initialization
- Wrong non-linearity

FROM A TALK BY GEOFFREY HINTON

How it learns

How it learns

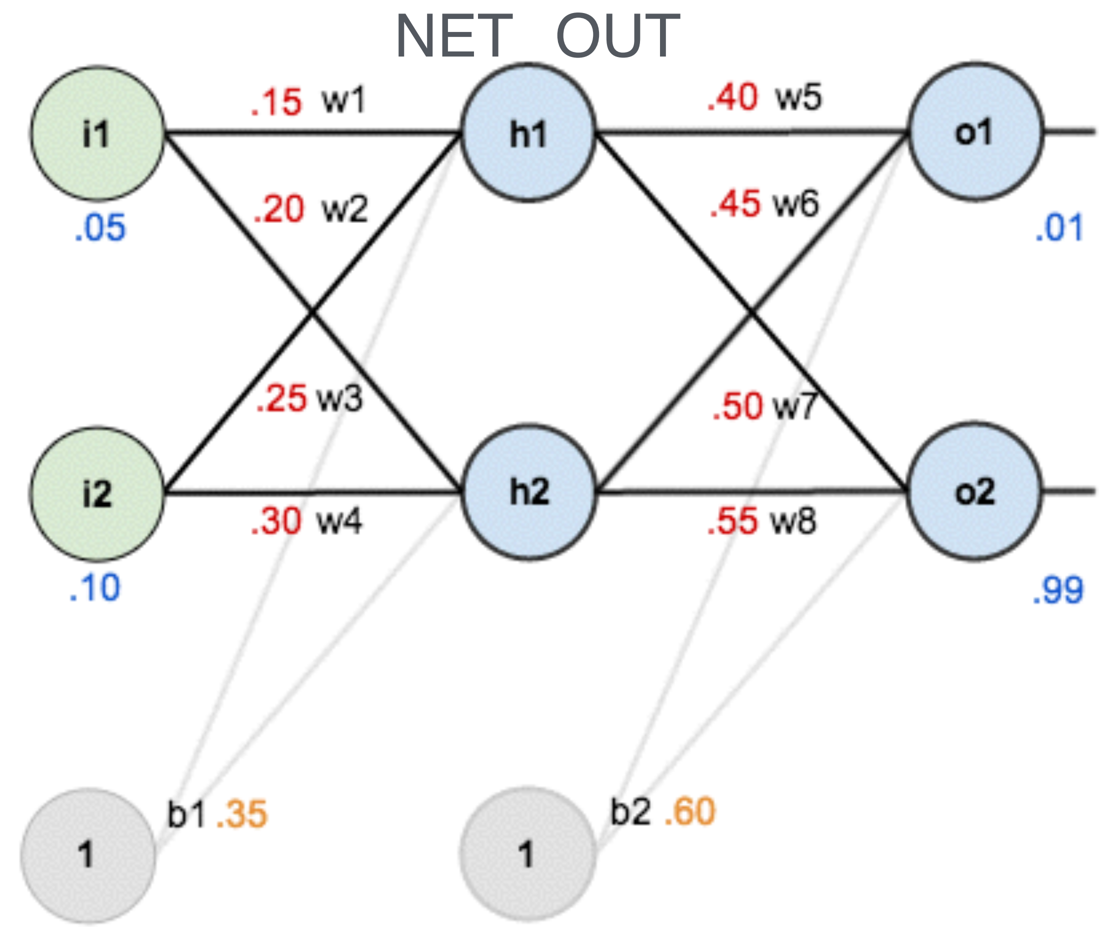
Backpropagation



http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns
Backpropagation

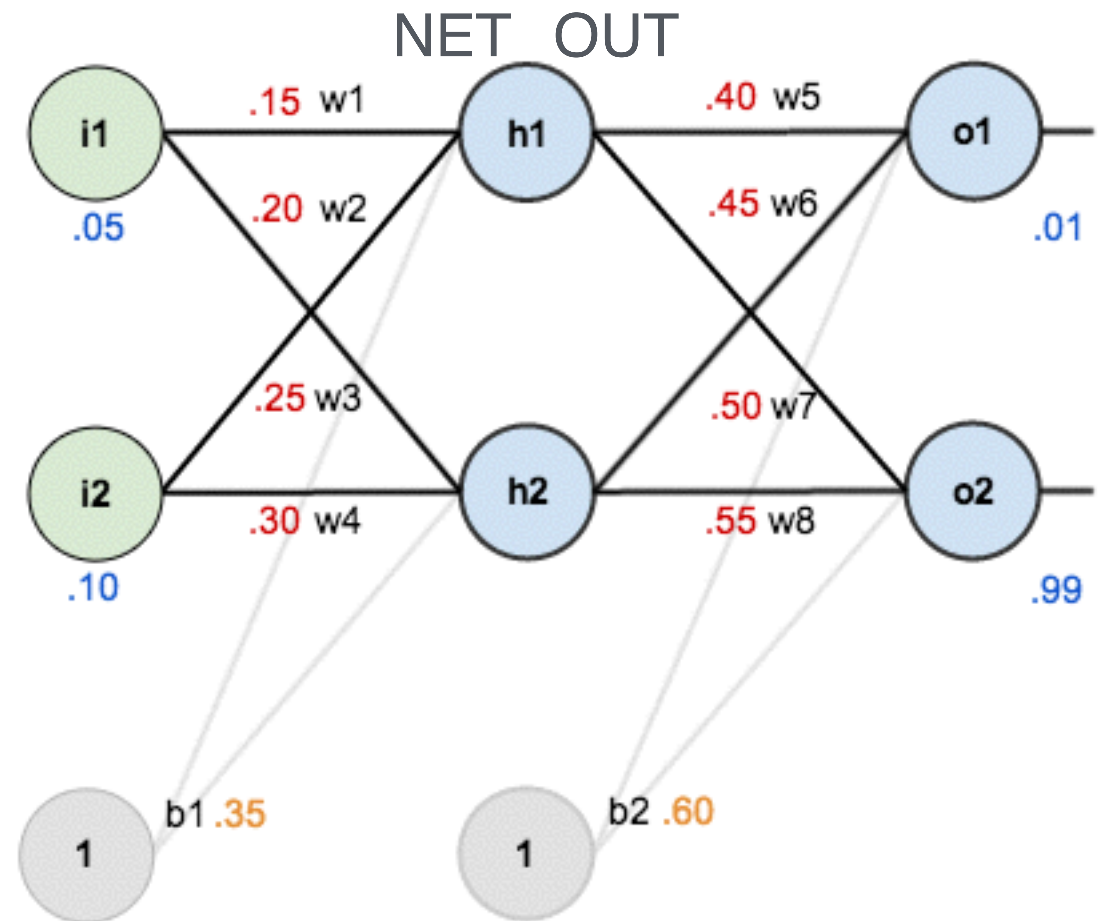
Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99



http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



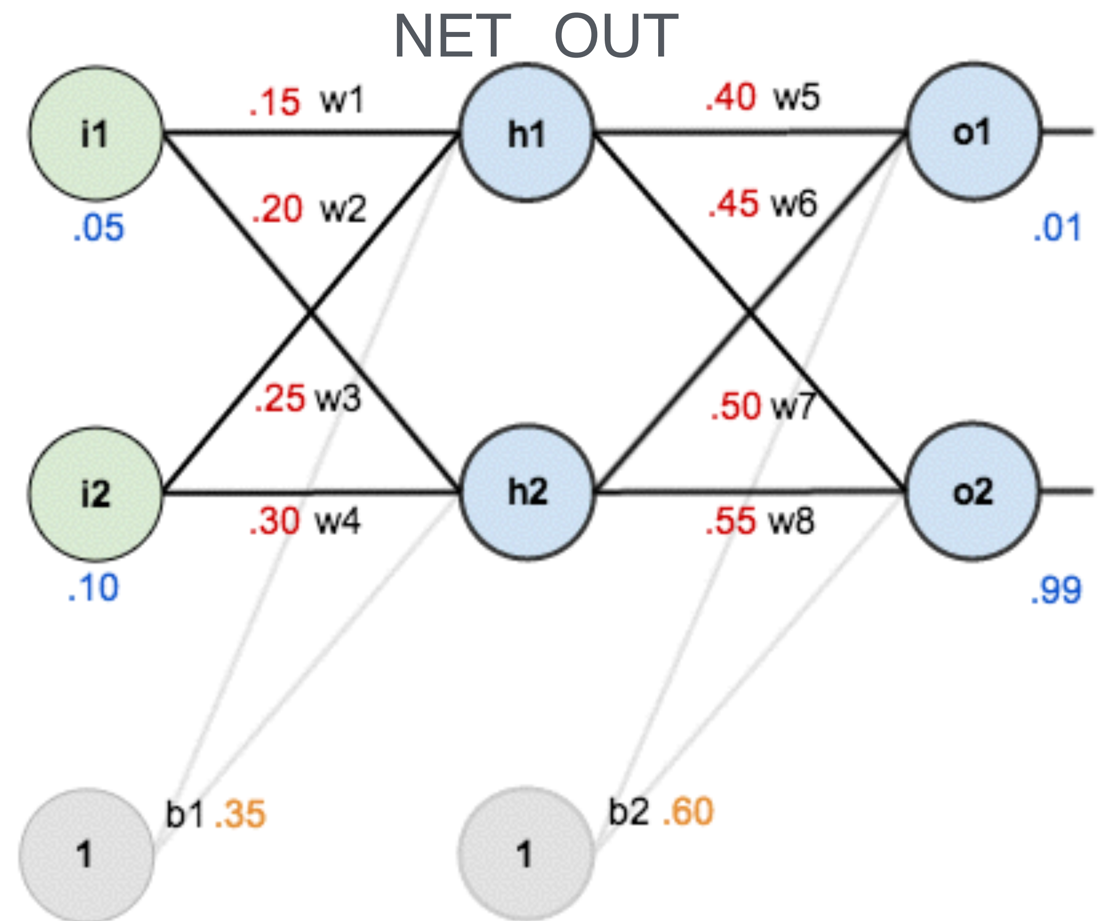
Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

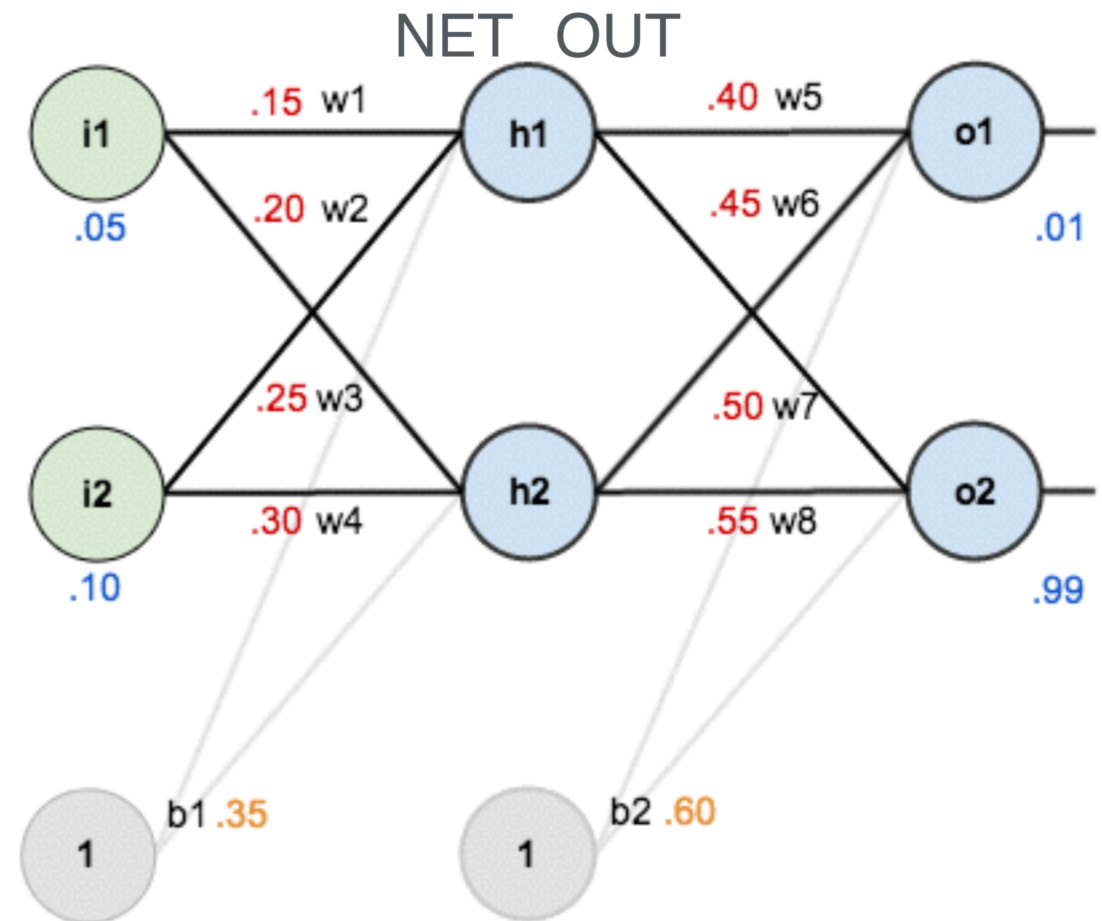
1. The Forward Pass — Calculating the total error

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$
$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

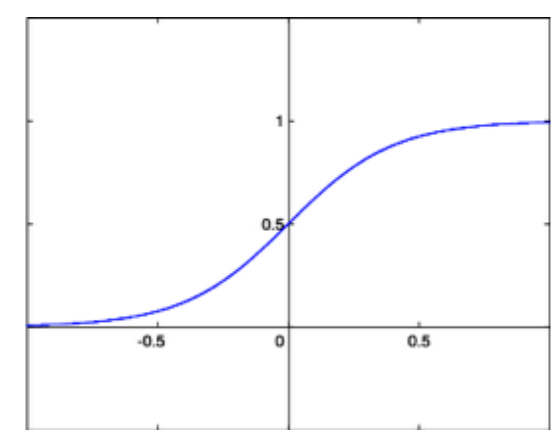
Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

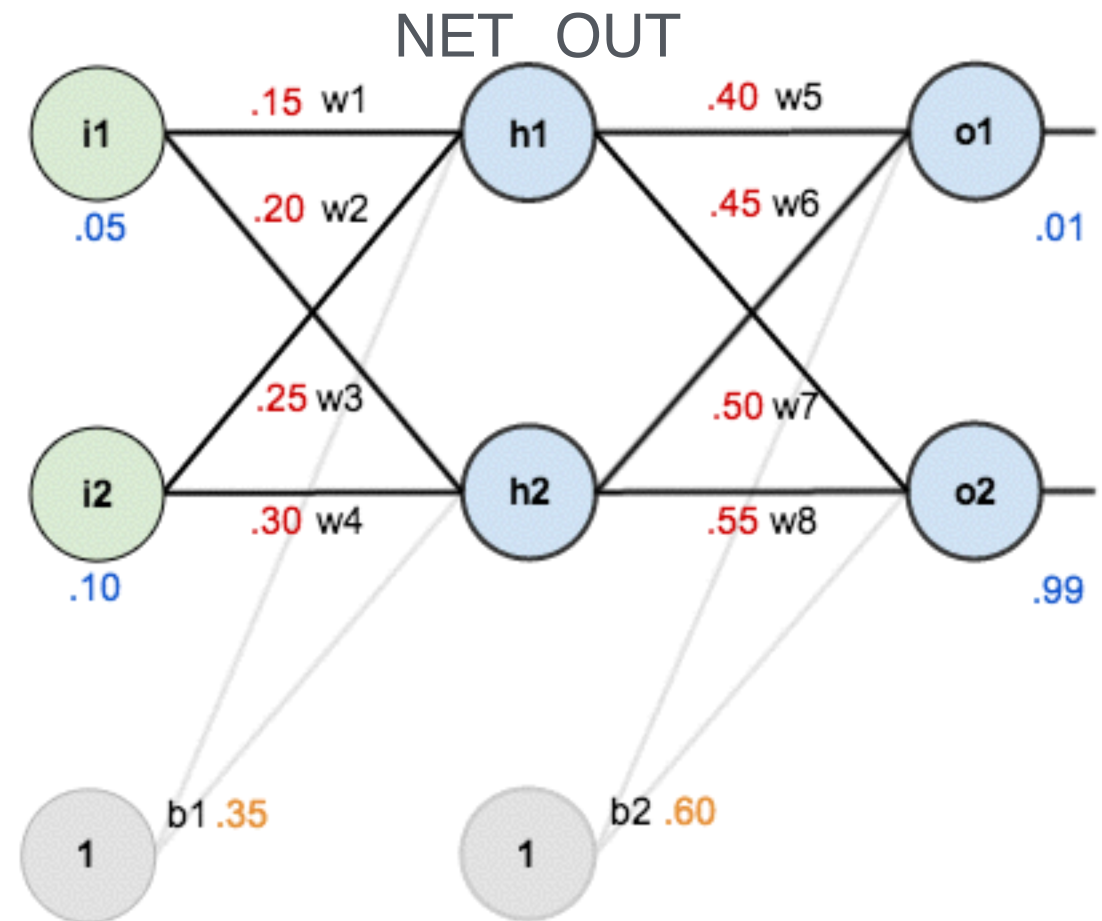
$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$
$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$



$$f(x) = \frac{1}{1 + e^{-x}}$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns
Backpropagation

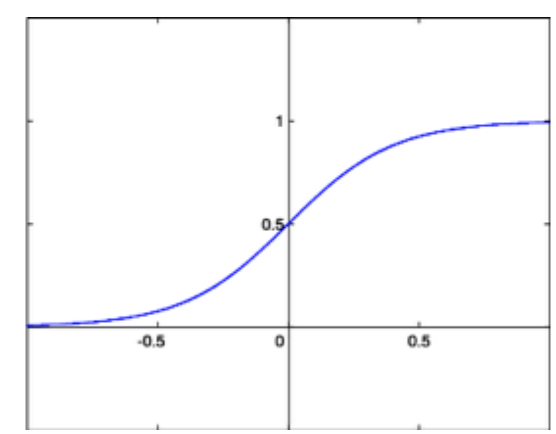


Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$
$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

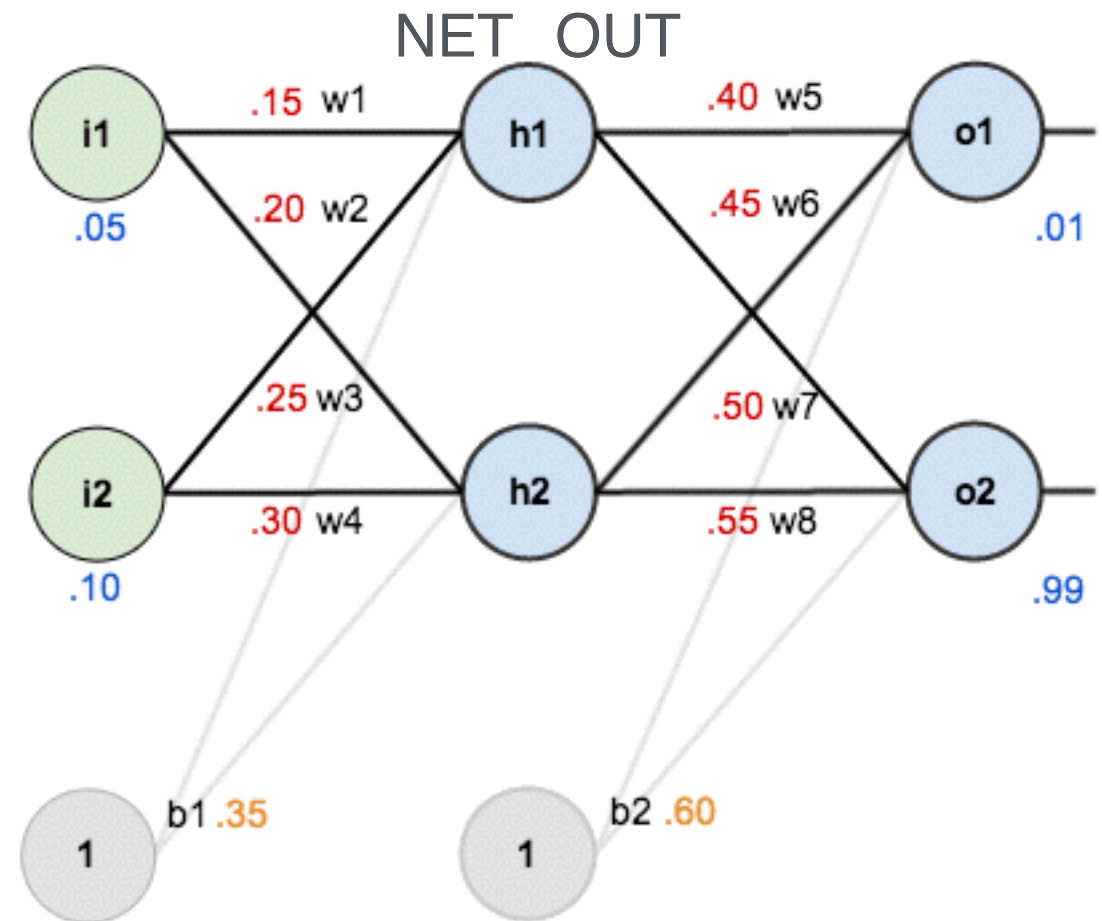
$$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0.593269992$$



$$f(x) = \frac{1}{1+e^{-x}}$$

How it learns

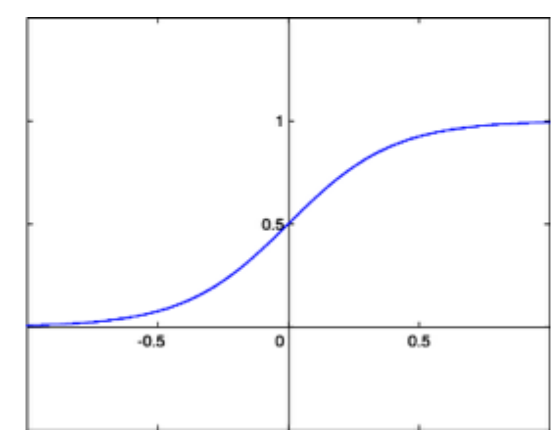
Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$
$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$



$$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0.593269992$$

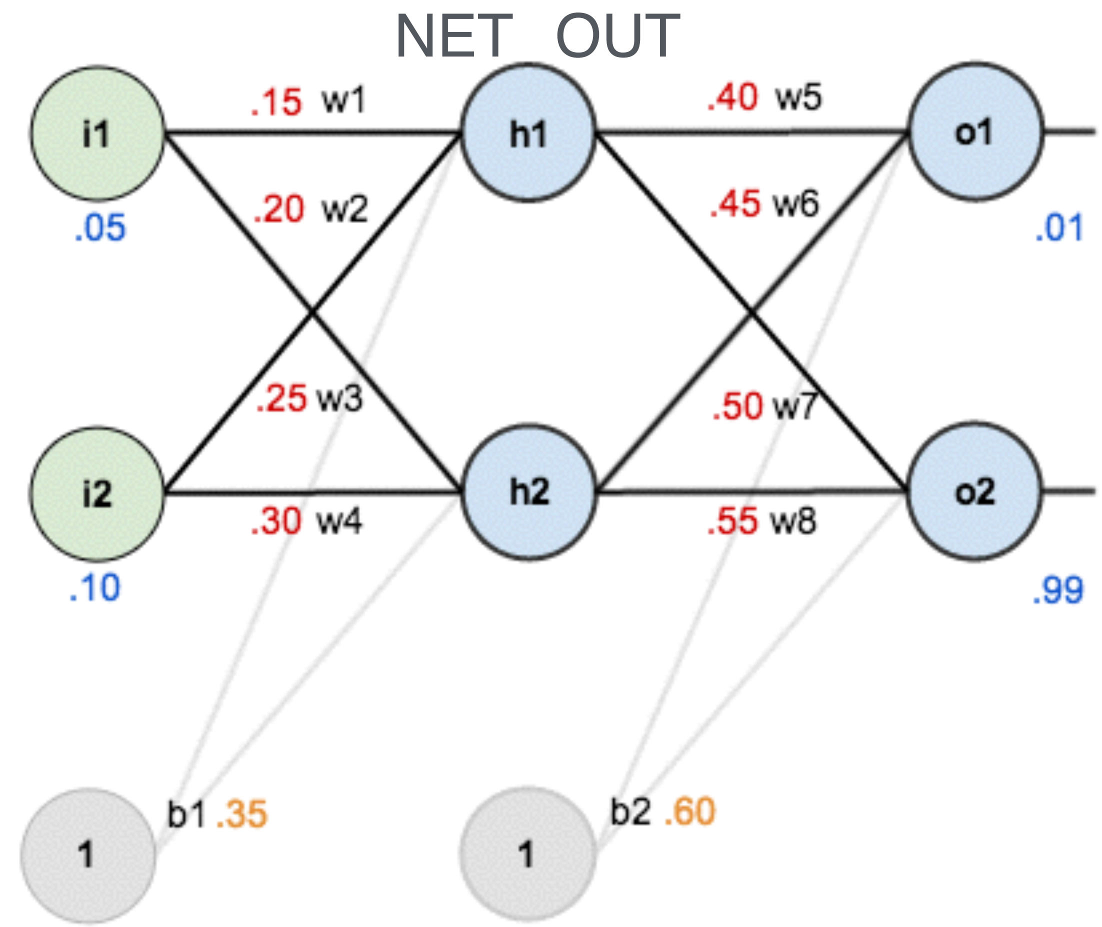
$$f(x) = \frac{1}{1+e^{-x}}$$

Repeat for h2 = 0.596, o1 = 0.751, o2 = 0.773

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

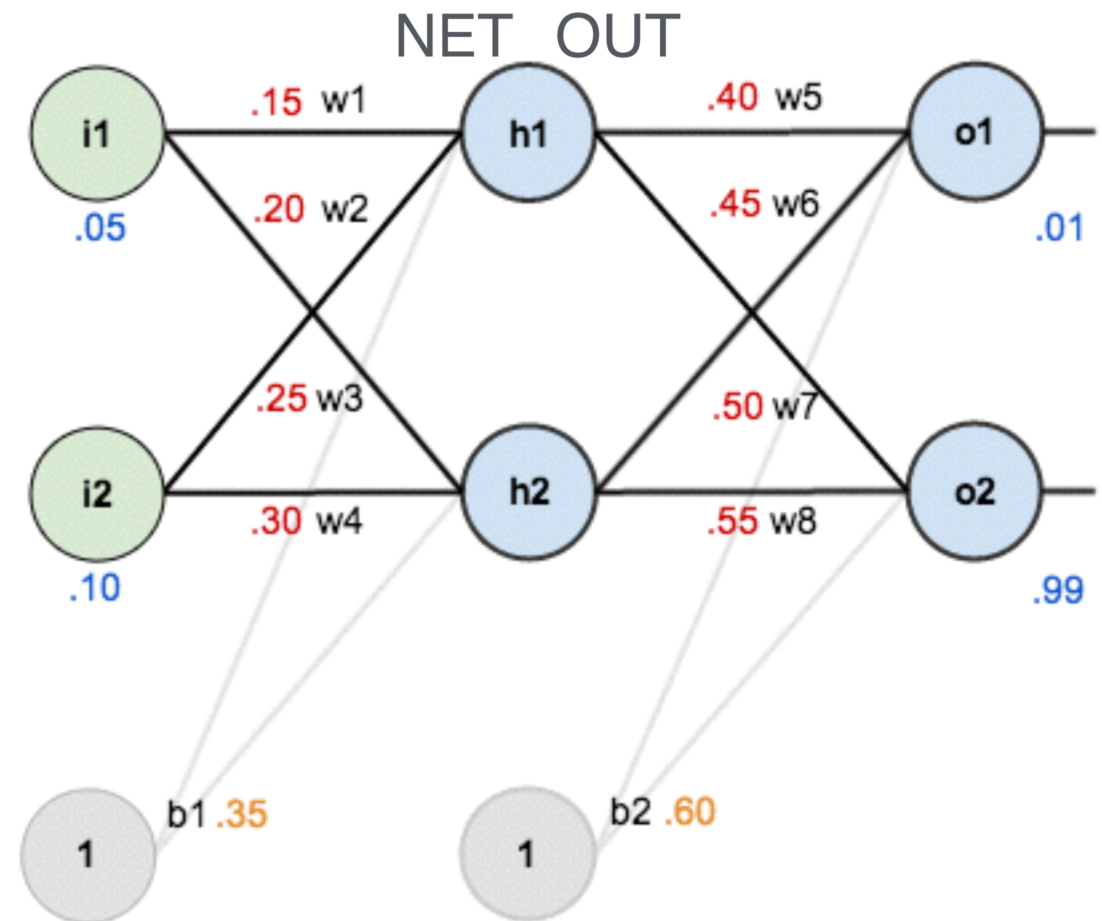
1. The Forward Pass — Calculating the total error

We have o1, o2

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

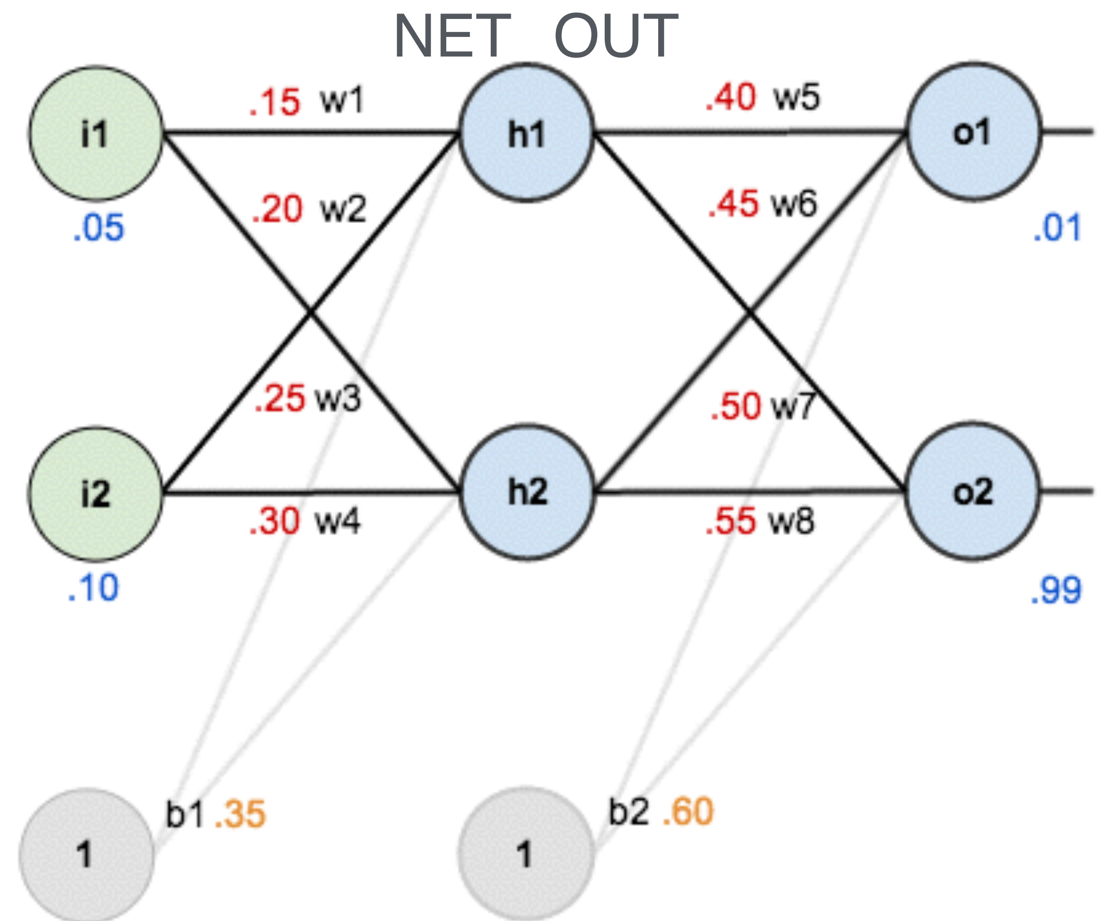
1. The Forward Pass — Calculating the total error

We have o1, o2 $$E_{total} = \sum \frac{1}{2}(target - output)^2$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

We have o1, o2

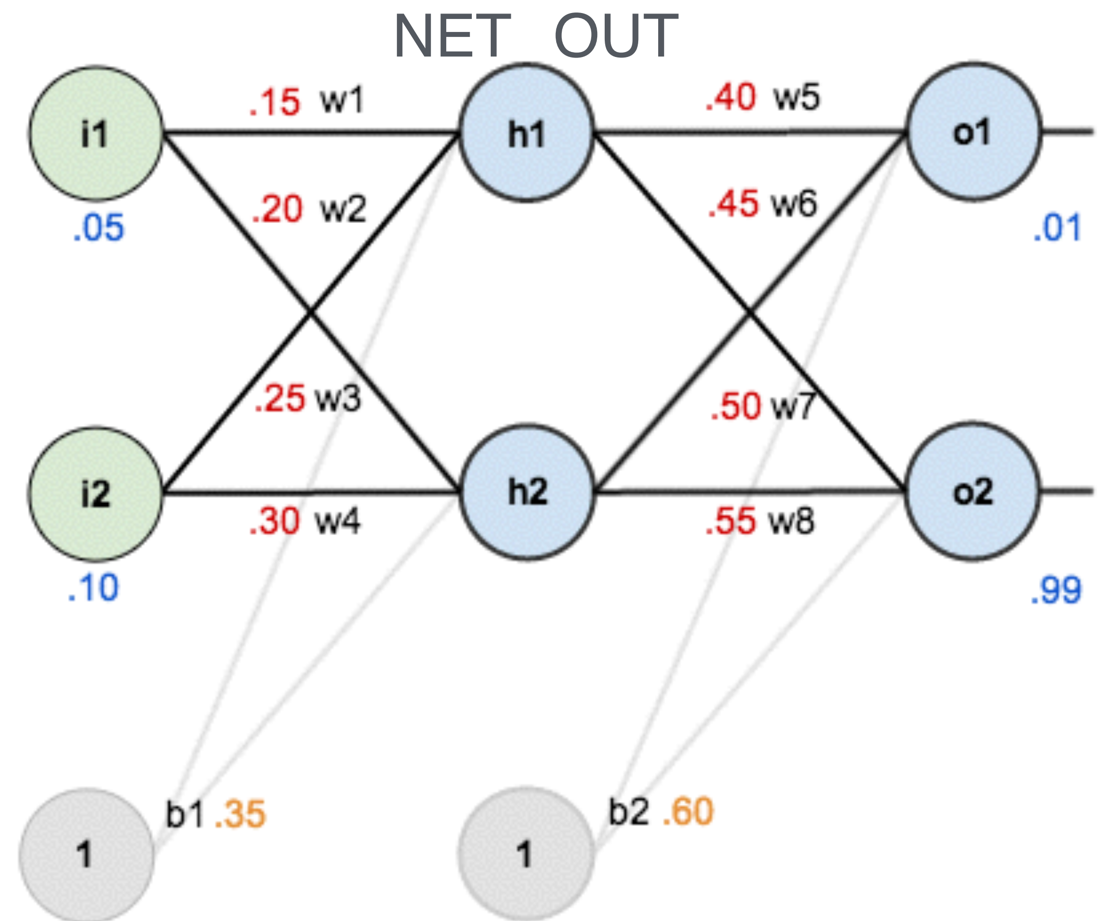
$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

We have o1, o2 $$E_{total} = \sum \frac{1}{2}(target - output)^2$$

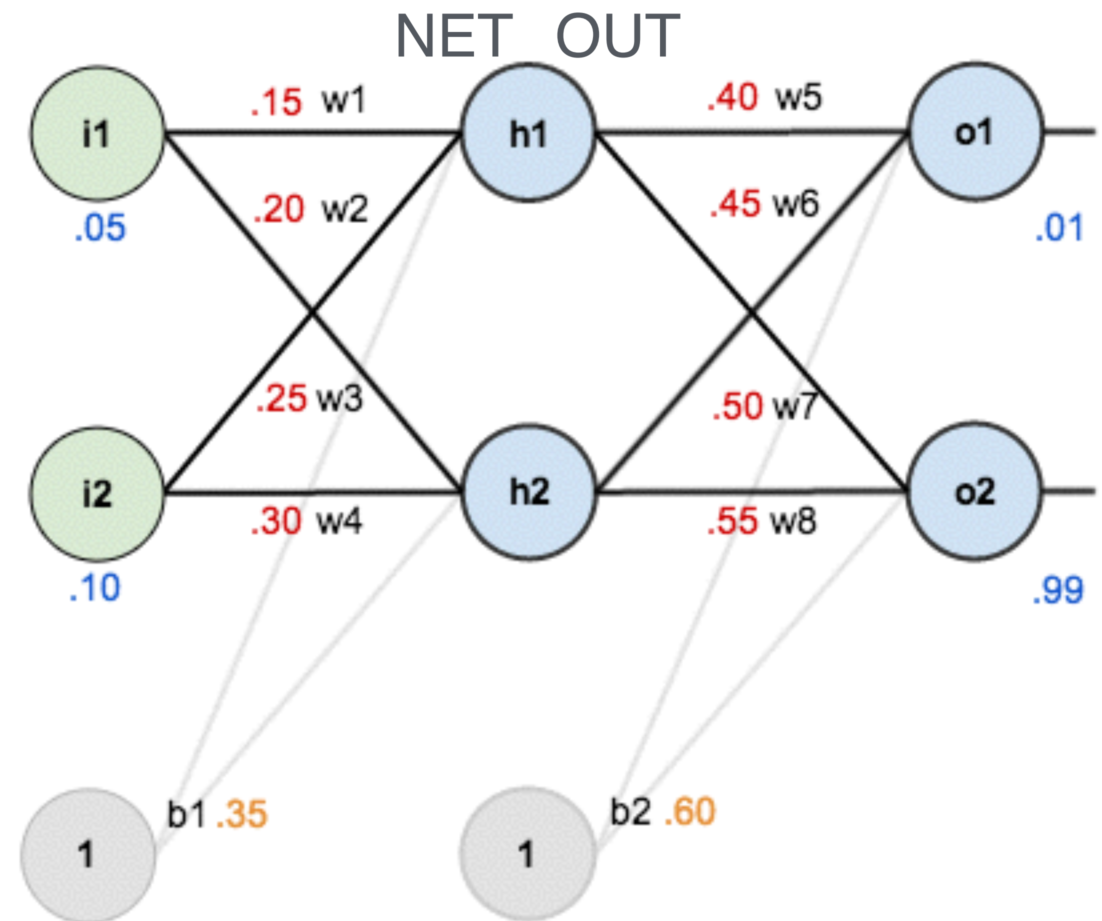
$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

$$E_{o2} = 0.023560026$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Forward Pass — Calculating the total error

We have o1, o2

$$E_{total} = \sum \frac{1}{2}(target - output)^2$$

$$E_{o1} = \frac{1}{2}(target_{o1} - out_{o1})^2 = \frac{1}{2}(0.01 - 0.75136507)^2 = 0.274811083$$

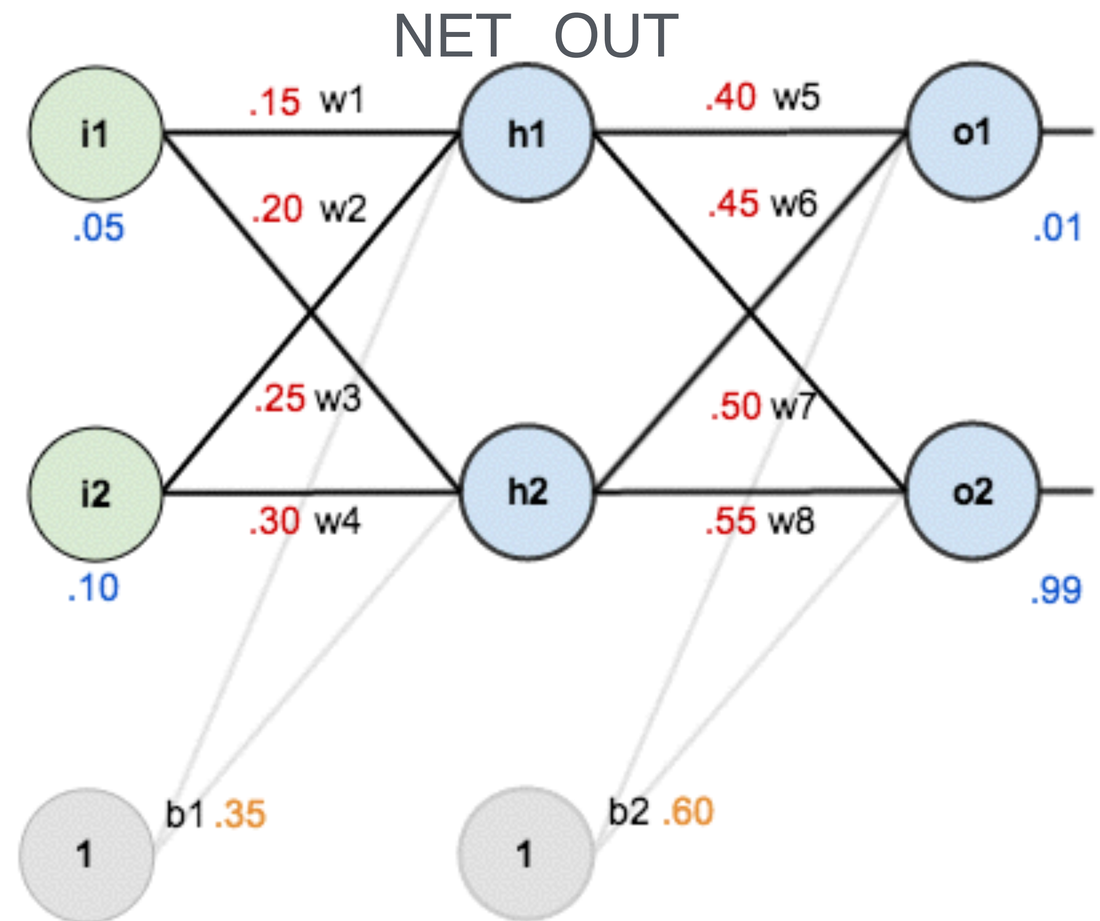
$$E_{o2} = 0.023560026$$

$$E_{total} = E_{o1} + E_{o2} = 0.274811083 + 0.023560026 = 0.298371109$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



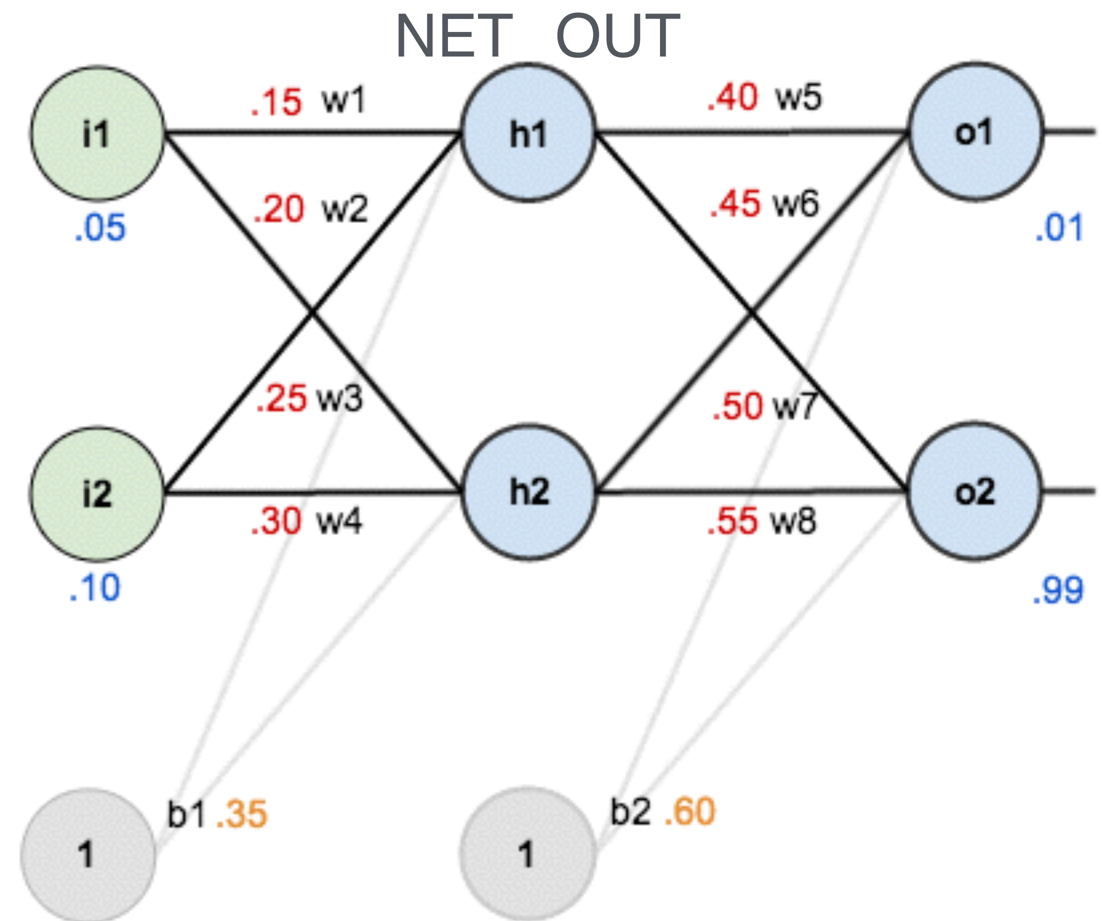
Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

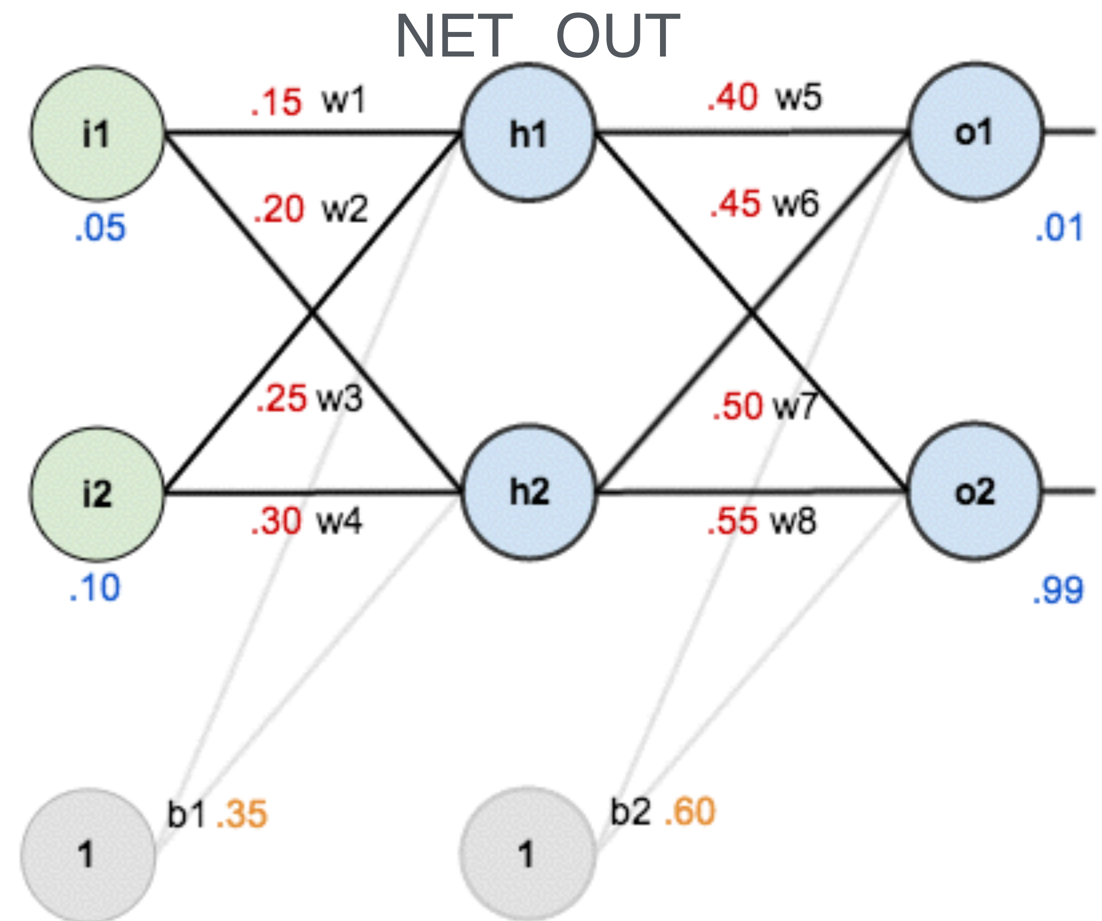
1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

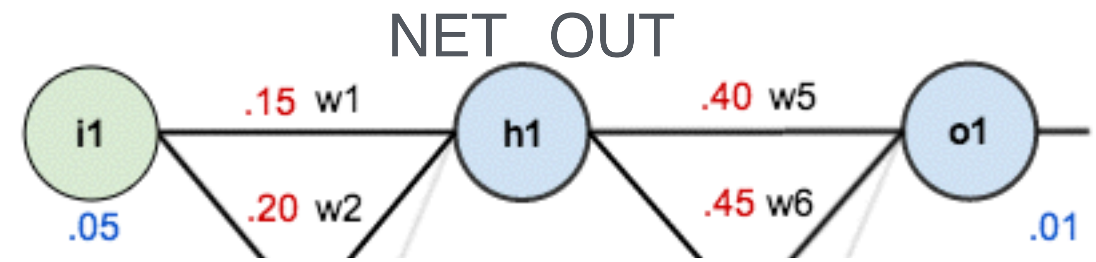
Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

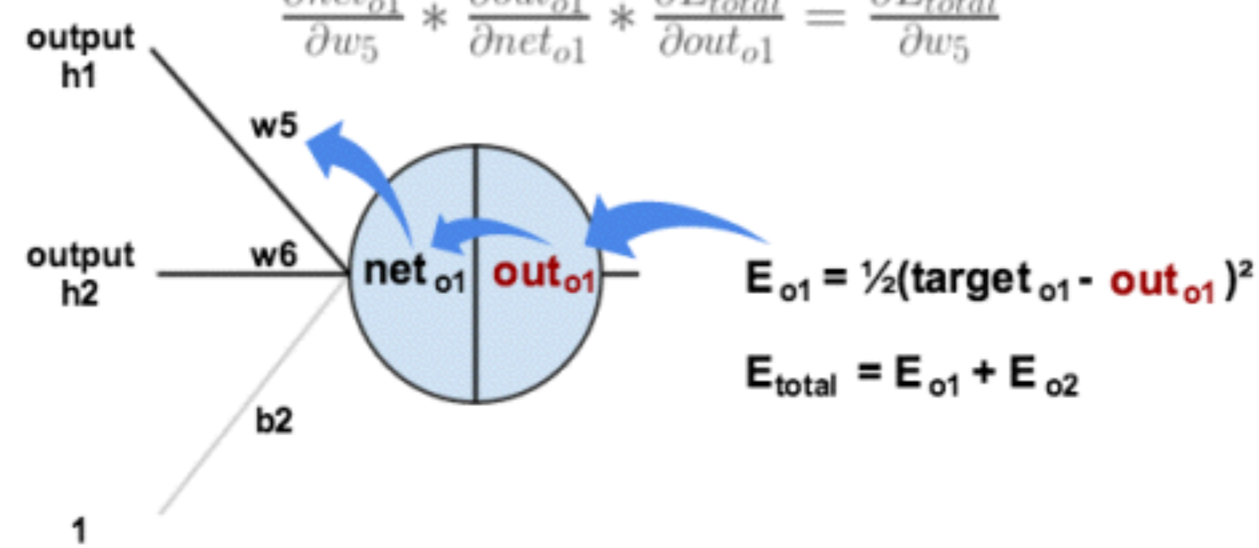
http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



$$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$$



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

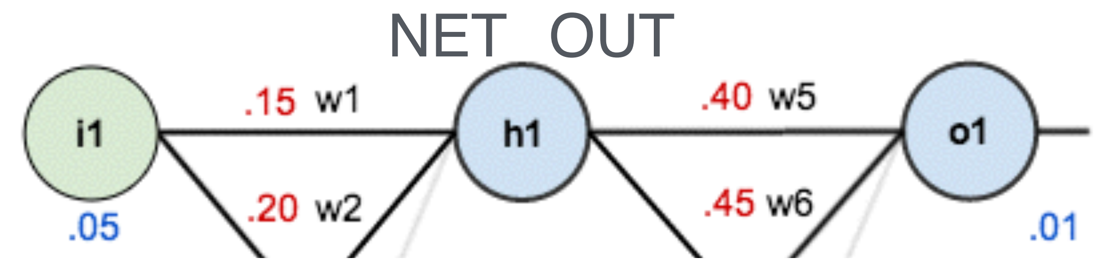
Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

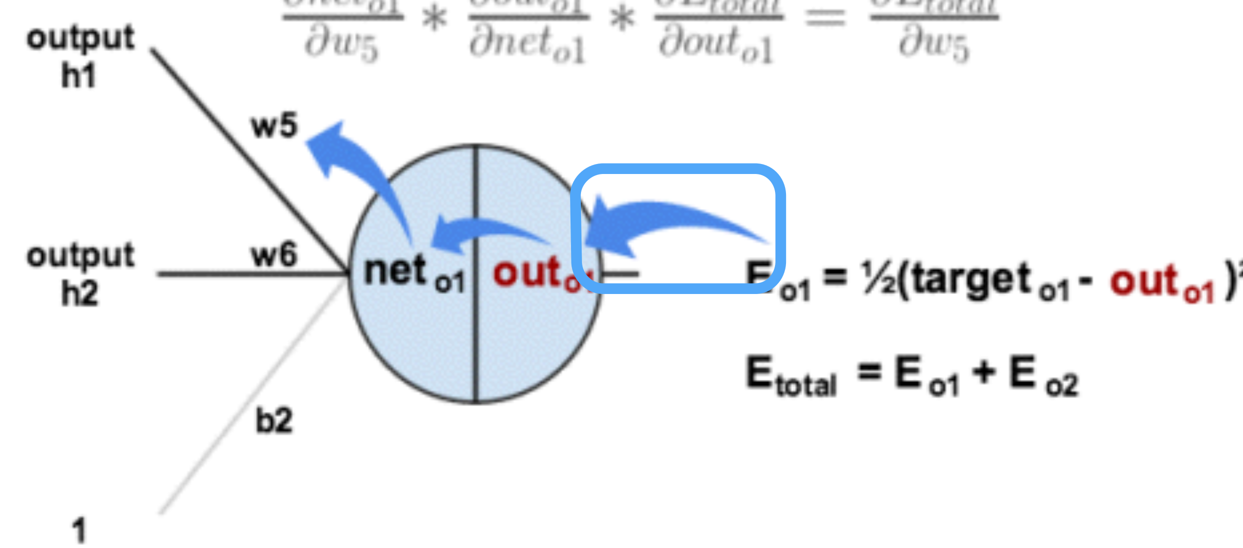
http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



$$\frac{\partial net_{o1}}{\partial w_5} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial E_{total}}{\partial out_{o1}} = \frac{\partial E_{total}}{\partial w_5}$$



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

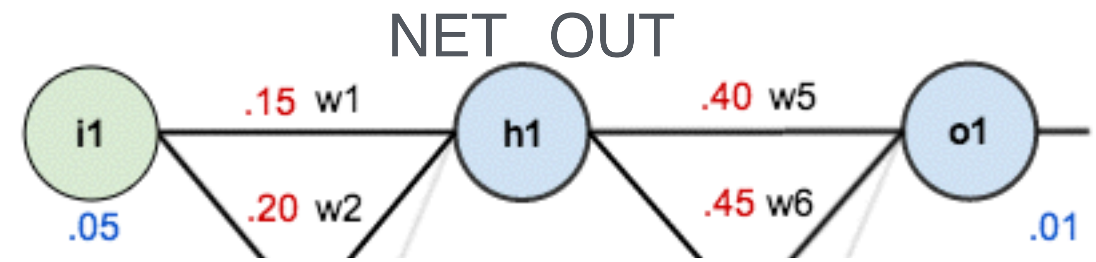
Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

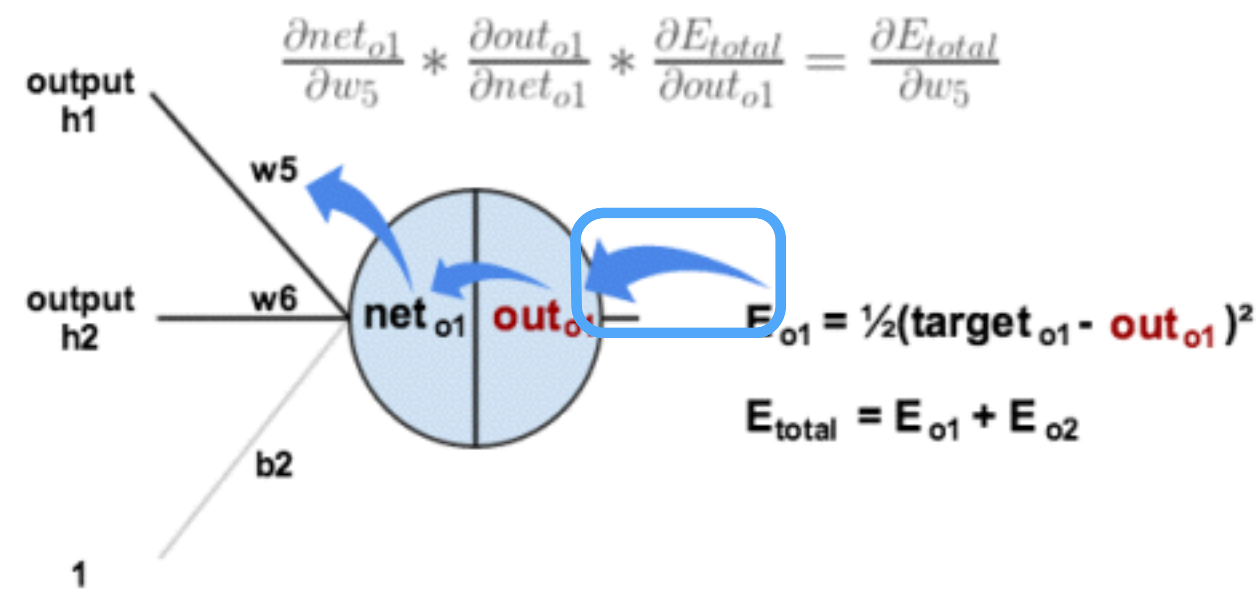
$$\frac{\partial E_{total}}{\partial w_5} = \boxed{\frac{\partial E_{total}}{\partial out_{o1}}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

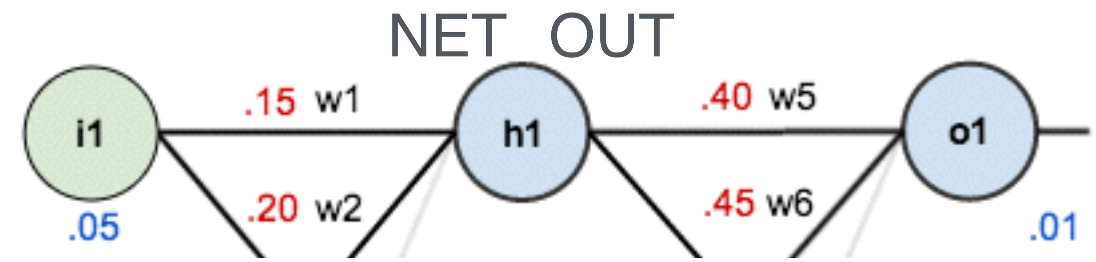
$$\frac{\partial E_{total}}{\partial w_5} = \boxed{\frac{\partial E_{total}}{\partial out_{o1}}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

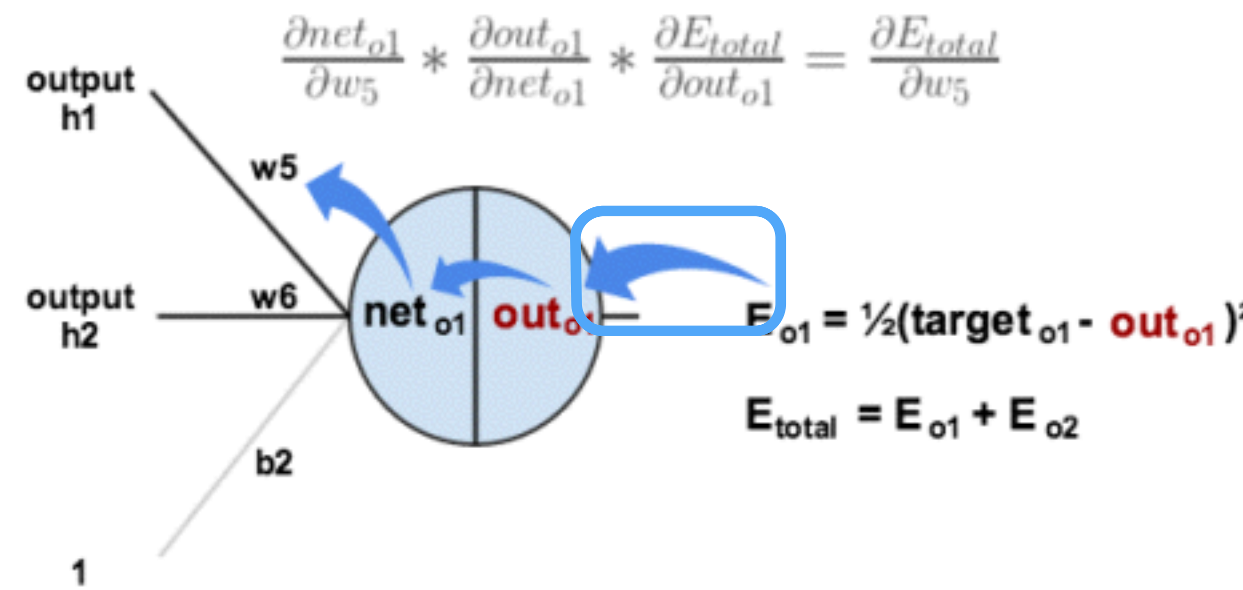
$$E_{total} = \frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \boxed{\frac{\partial E_{total}}{\partial out_{o1}}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

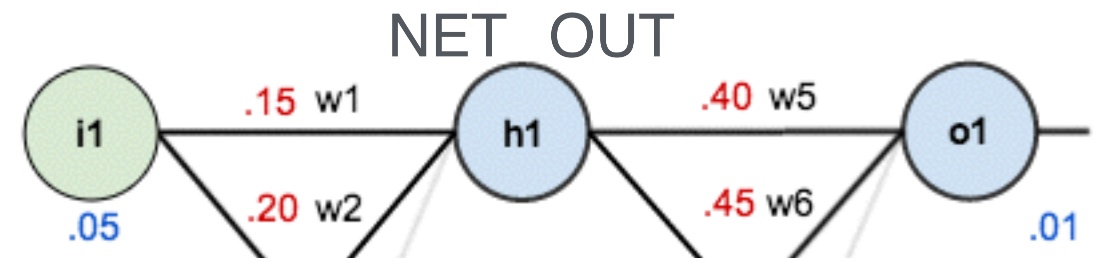
$$E_{total} = \frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2$$

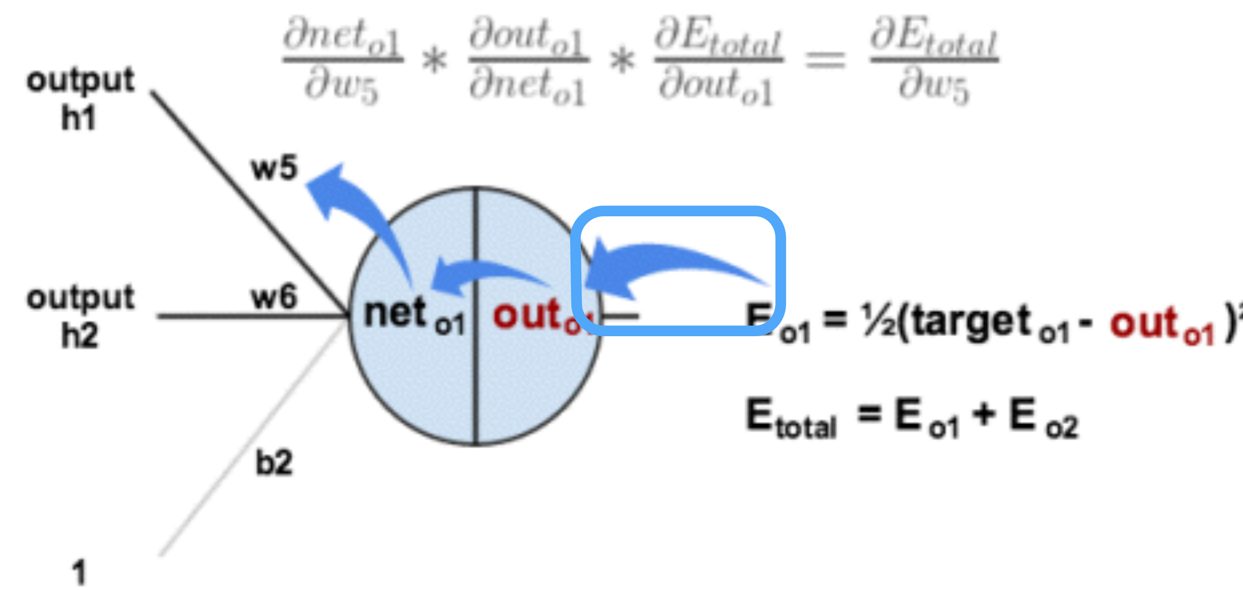
$$\frac{\partial E_{total}}{\partial out_{o1}} = 2 * \frac{1}{2}(target_{o1} - out_{o1})^{2-1} * -1 + 0$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \boxed{\frac{\partial E_{total}}{\partial out_{o1}}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

$$E_{total} = \frac{1}{2}(target_{o1} - out_{o1})^2 + \frac{1}{2}(target_{o2} - out_{o2})^2$$

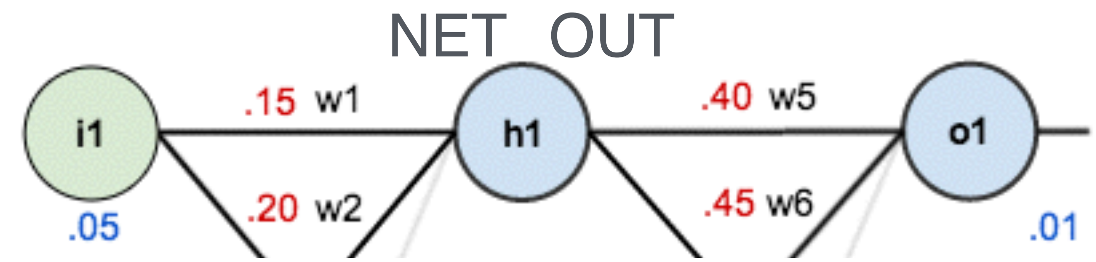
$$\frac{\partial E_{total}}{\partial out_{o1}} = 2 * \frac{1}{2}(target_{o1} - out_{o1})^{2-1} * -1 + 0$$

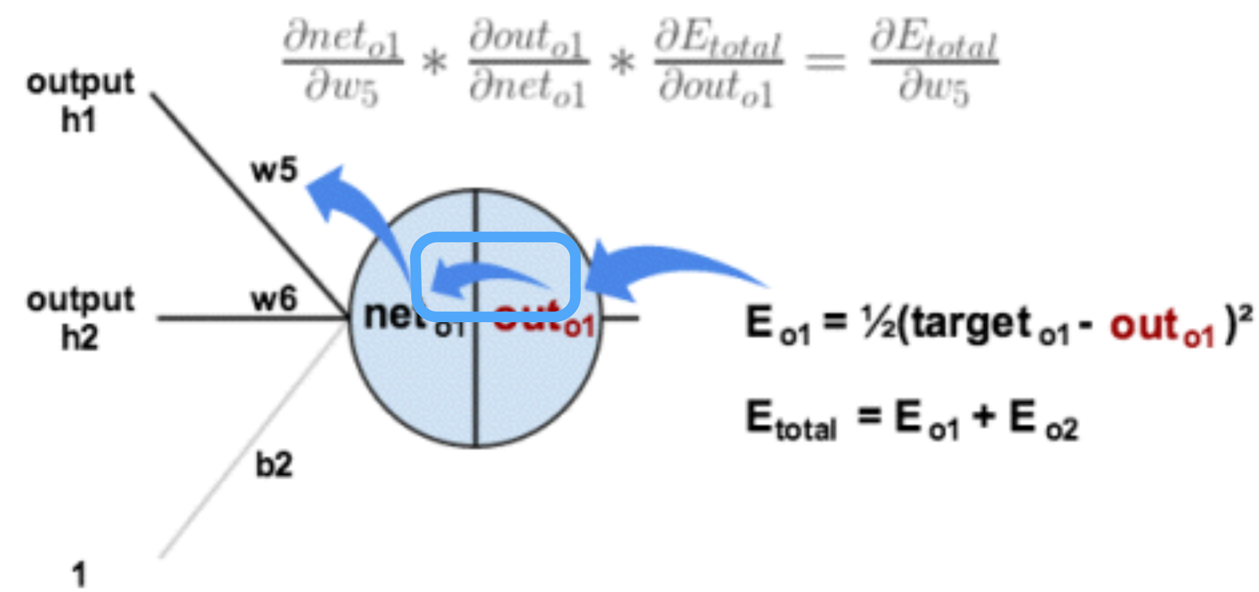
$$\frac{\partial E_{total}}{\partial out_{o1}} = -(target_{o1} - out_{o1}) = -(0.01 - 0.75136507) = 0.74136507$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

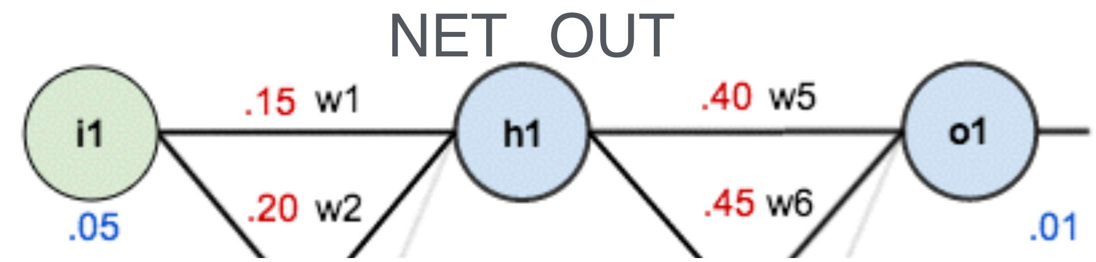
Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

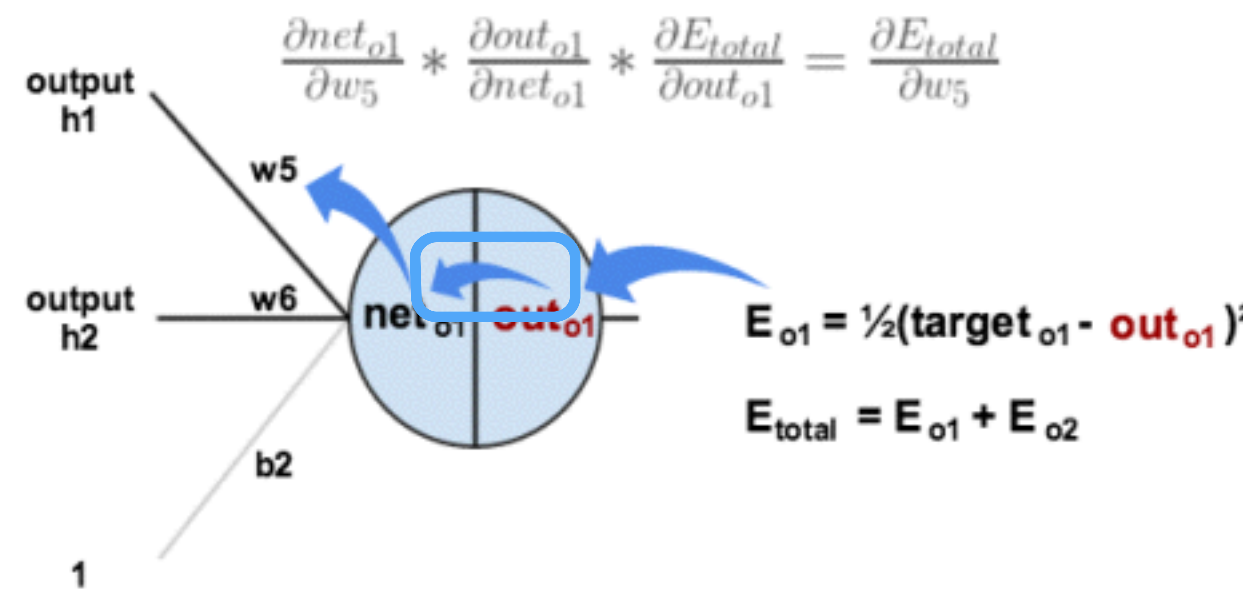
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \boxed{\frac{\partial out_{o1}}{\partial net_{o1}}} * \frac{\partial net_{o1}}{\partial w_5}$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

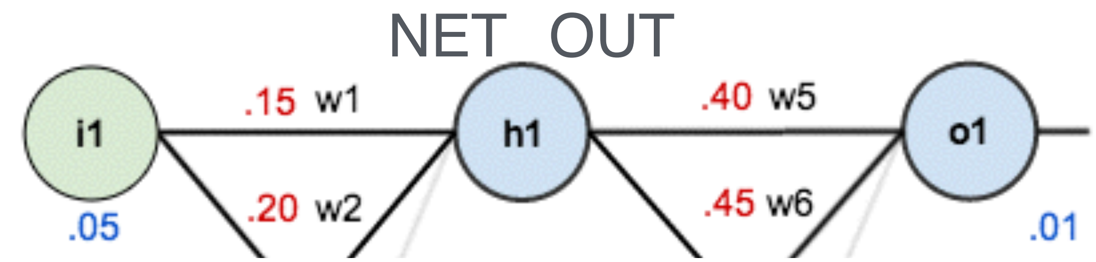
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \boxed{\frac{\partial out_{o1}}{\partial net_{o1}}} * \frac{\partial net_{o1}}{\partial w_5}$$

$$out_{o1} = \frac{1}{1+e^{-net_{o1}}}$$

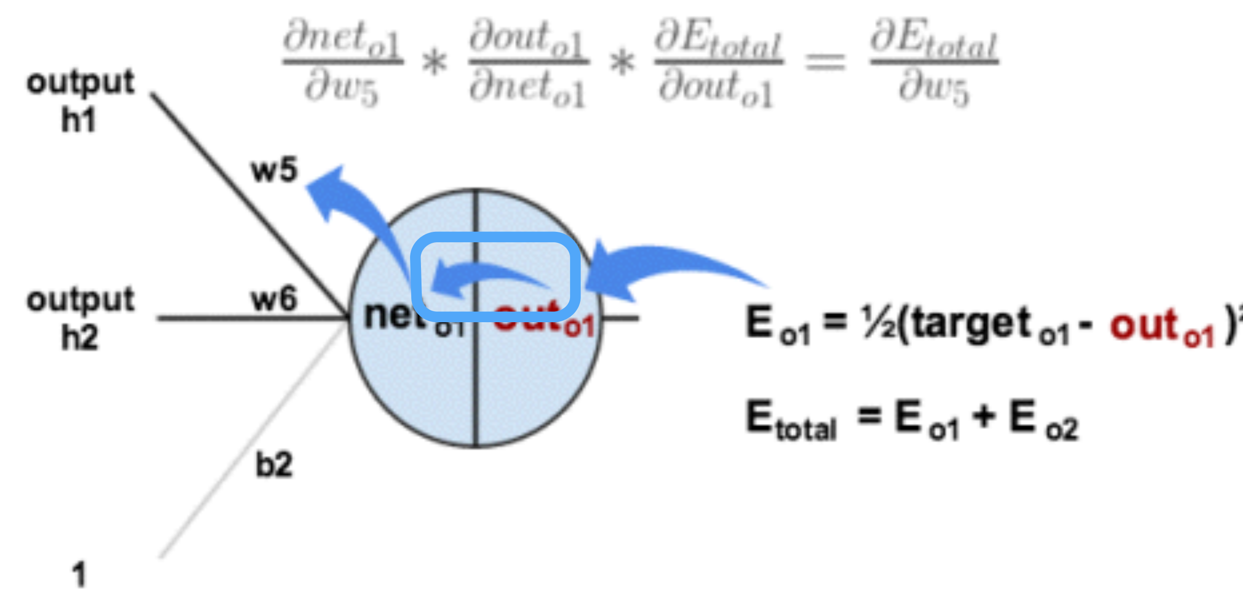
http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99



1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \boxed{\frac{\partial out_{o1}}{\partial net_{o1}}} * \frac{\partial net_{o1}}{\partial w_5}$$

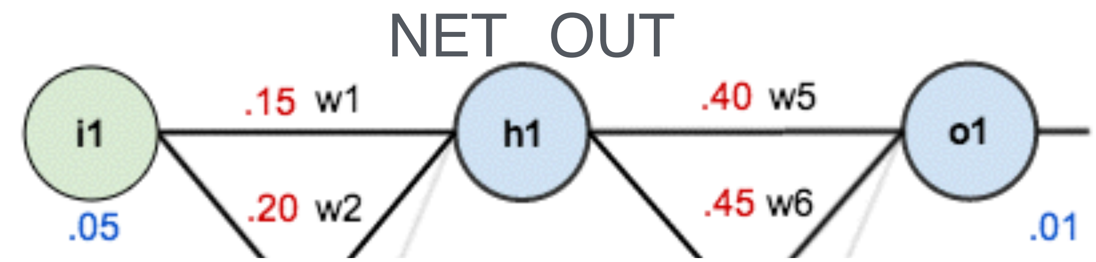
$$out_{o1} = \frac{1}{1+e^{-net_{o1}}}$$

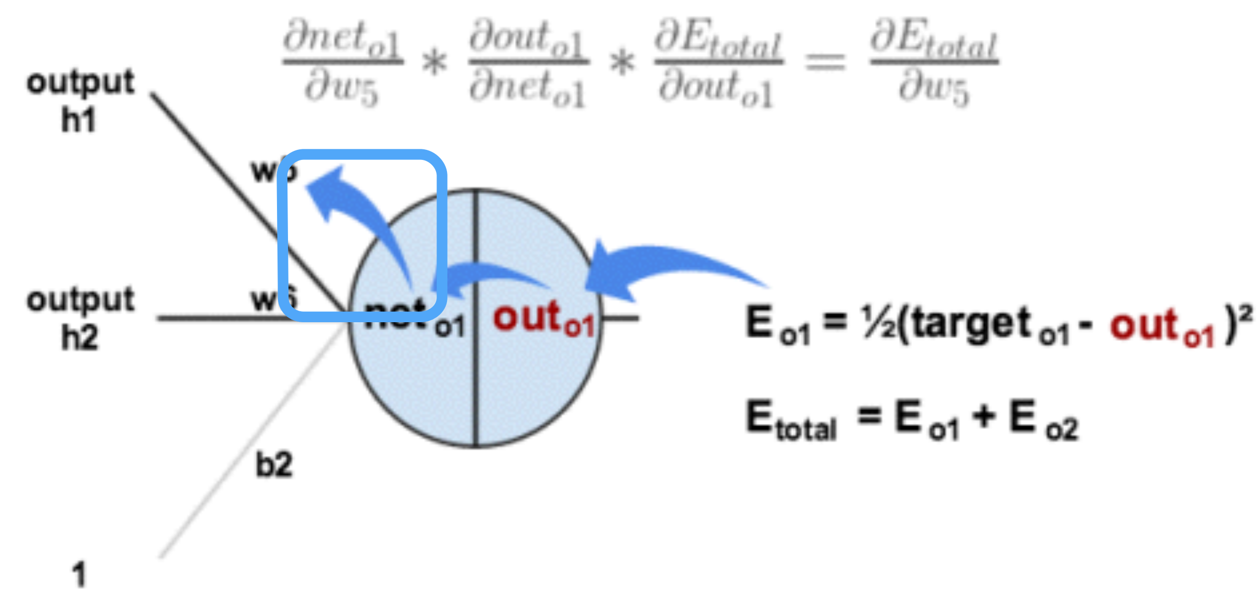
$$\frac{\partial out_{o1}}{\partial net_{o1}} = out_{o1}(1 - out_{o1}) = 0.75136507(1 - 0.75136507) = 0.18681560$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

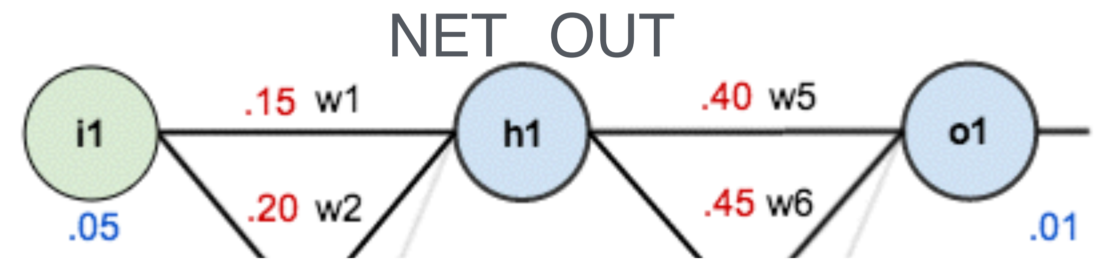
Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

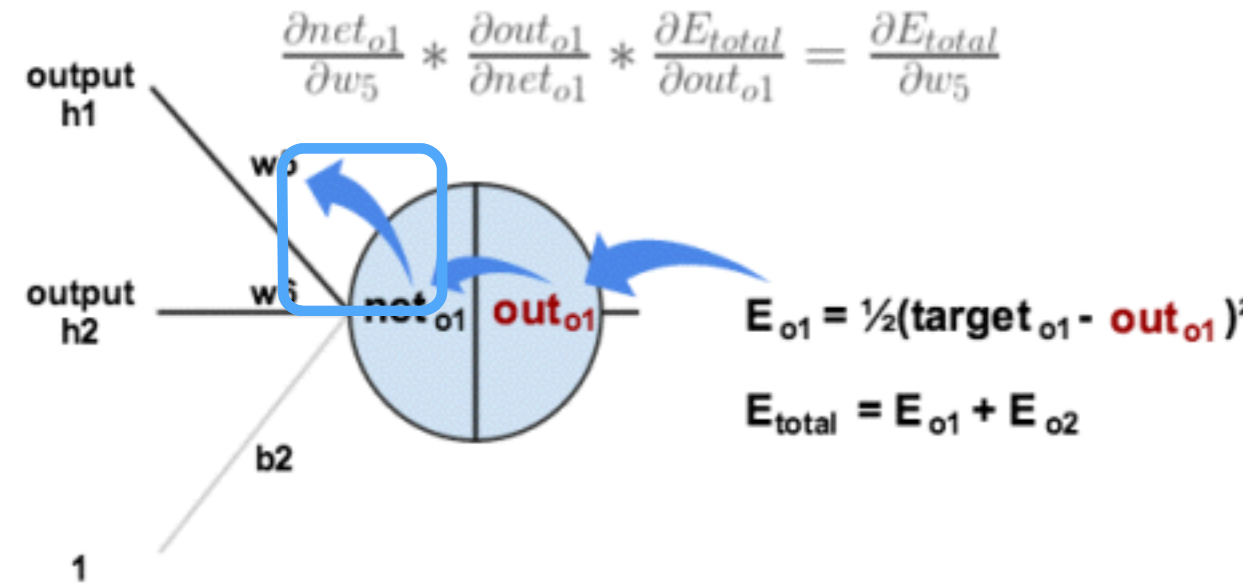
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \boxed{\frac{\partial net_{o1}}{\partial w_5}}$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

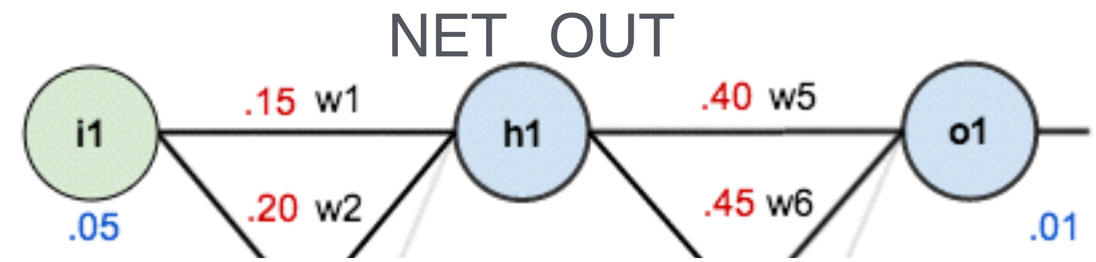
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \boxed{\frac{\partial net_{o1}}{\partial w_5}}$$

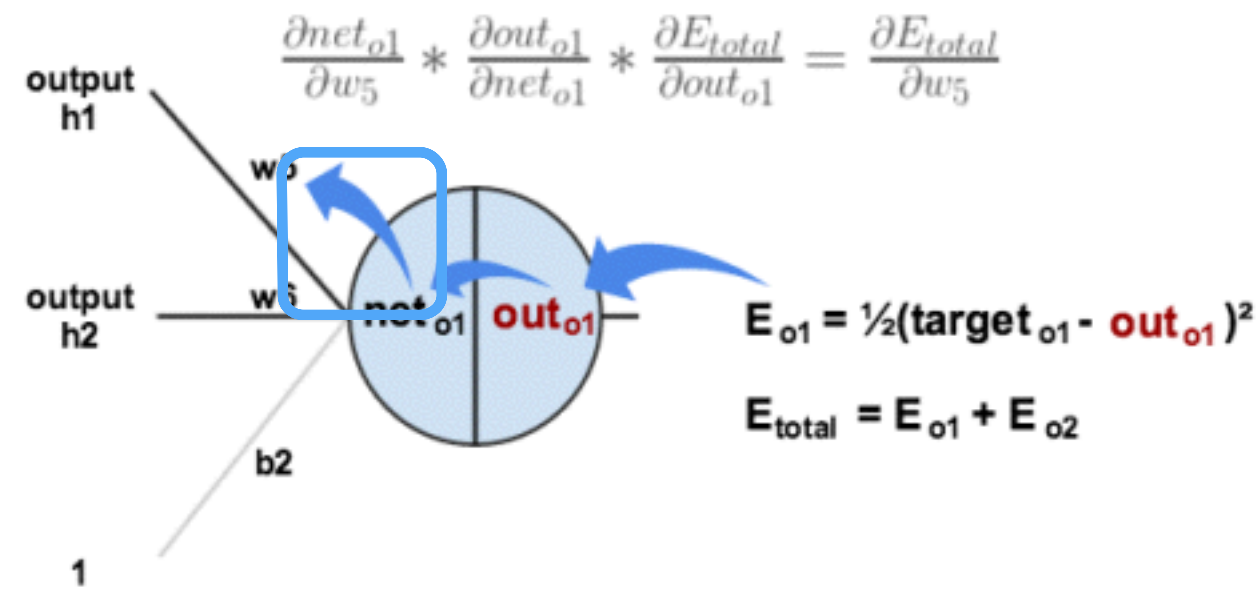
$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \boxed{\frac{\partial net_{o1}}{\partial w_5}}$$

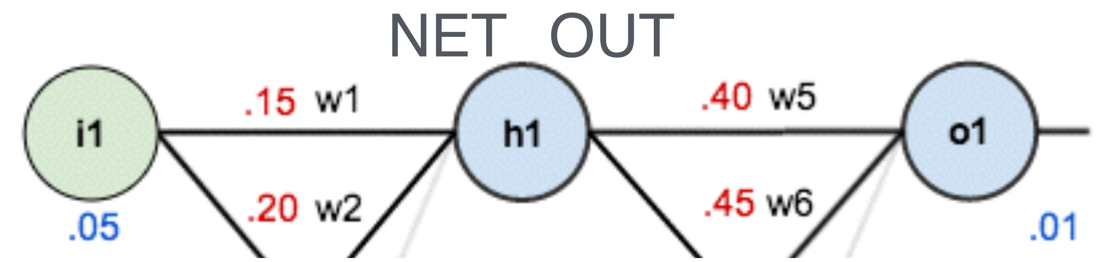
$$net_{o1} = w_5 * out_{h1} + w_6 * out_{h2} + b_2 * 1$$

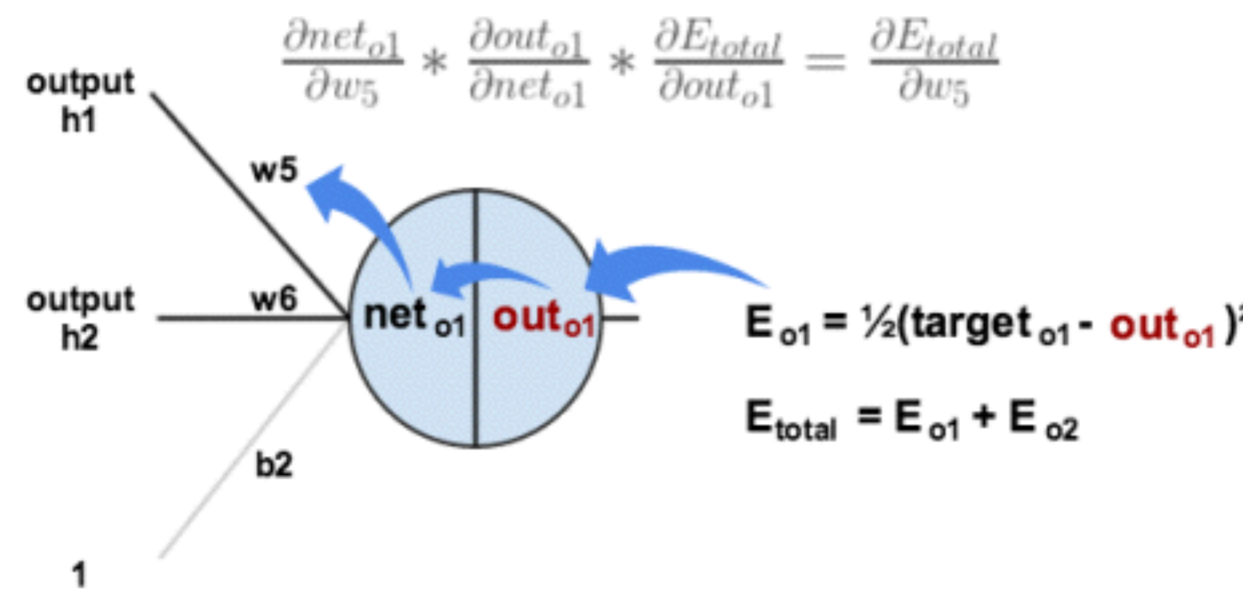
$$\frac{\partial net_{o1}}{\partial w_5} = 1 * out_{h1} * w_5^{(1-1)} + 0 + 0 = out_{h1} = 0.593269992$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

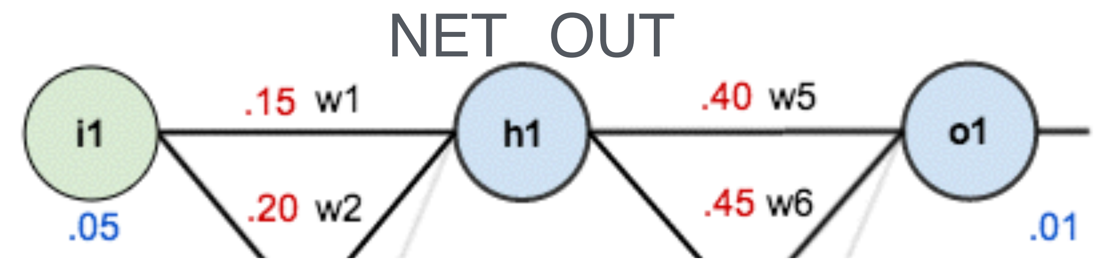
Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

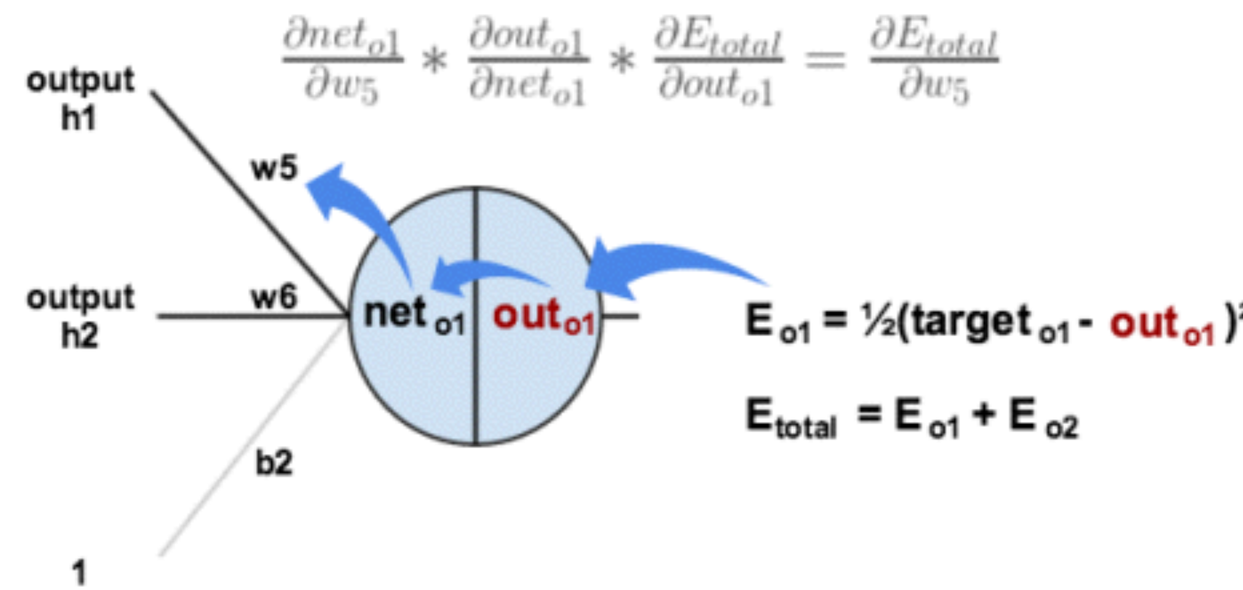
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

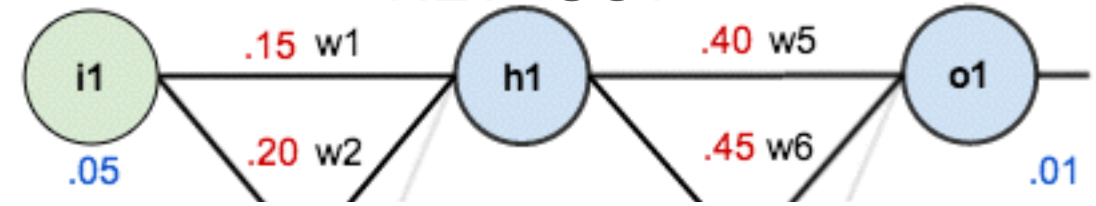
$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

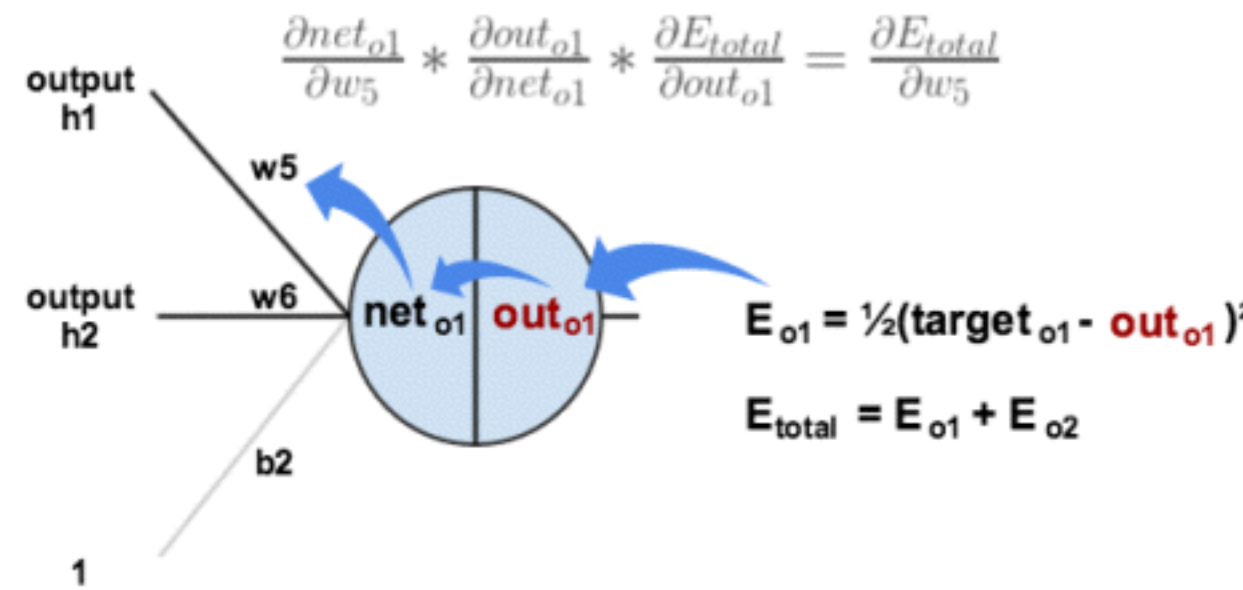
$$\frac{\partial E_{total}}{\partial w_5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

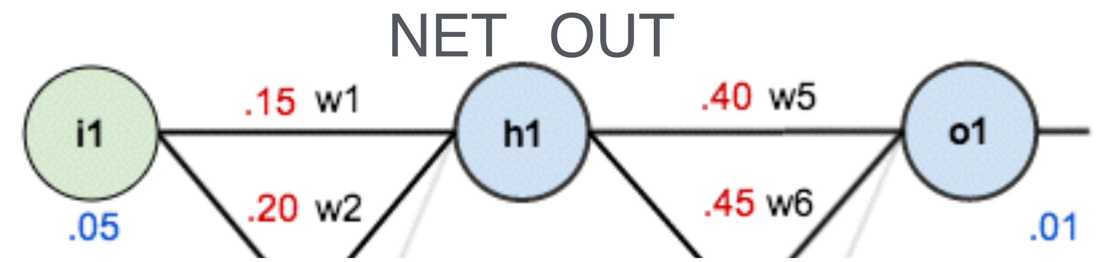
$$\frac{\partial E_{total}}{\partial w_5} = 0.74136507 * 0.186815602 * 0.593269992 = 0.082167041$$

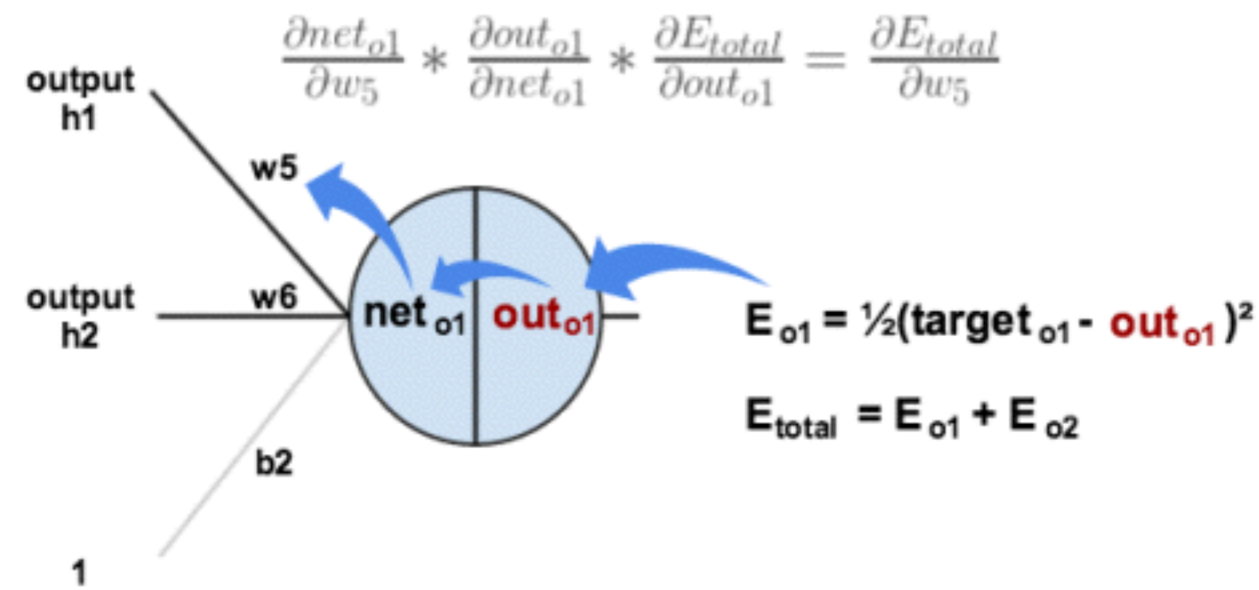
$$w_5^+ = w_5 - \eta * \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

Learning rate

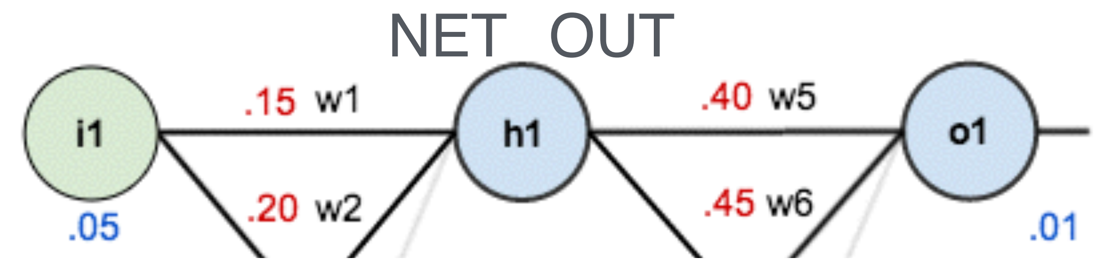
$$\frac{\partial E_{total}}{\partial w_5} = \ldots 5602 * 0.593269992 = 0.082167041$$

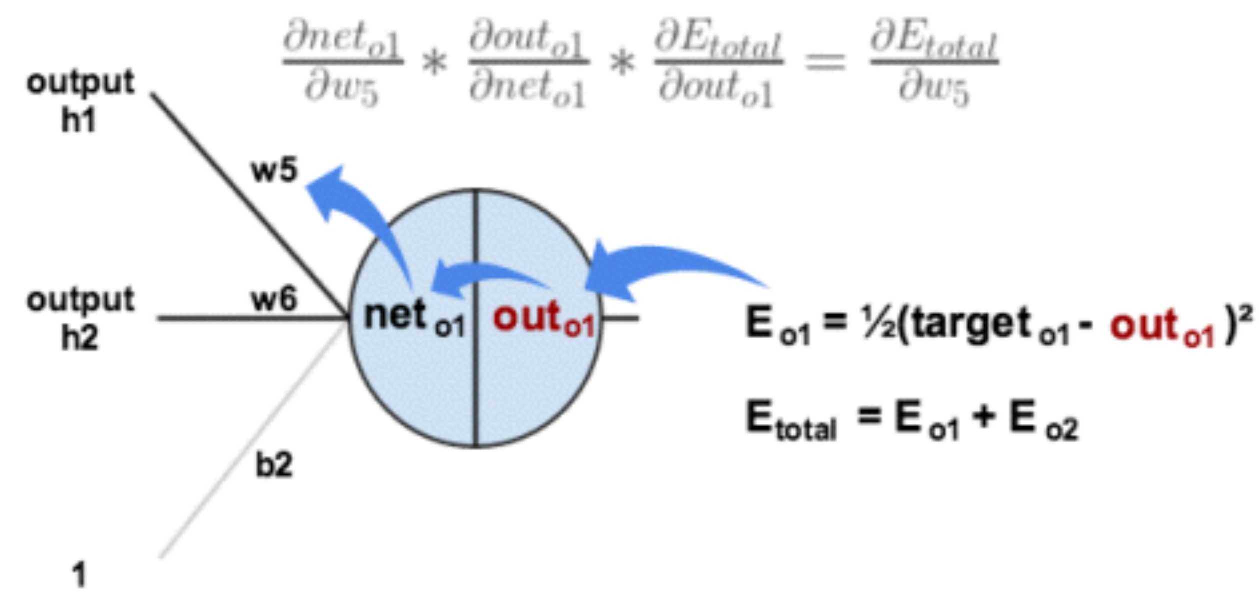
$$w_5^+ = w_5 - \eta * \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation





Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

1. The Backwards Pass — updating weights

Consider w_5. We want to know how much a change in w_5 affects the total error, aka $\frac{\partial E_{total}}{\partial w_5}$.

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial out_{o1}} * \frac{\partial out_{o1}}{\partial net_{o1}} * \frac{\partial net_{o1}}{\partial w_5}$$

Learning rate

$$\frac{\partial E_{total}}{\partial w_5} = \ldots 5602 * 0.593269992 = 0.082167041$$

$$w_5^+ = w_5 - \eta * \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 * 0.082167041 = 0.35891648$$

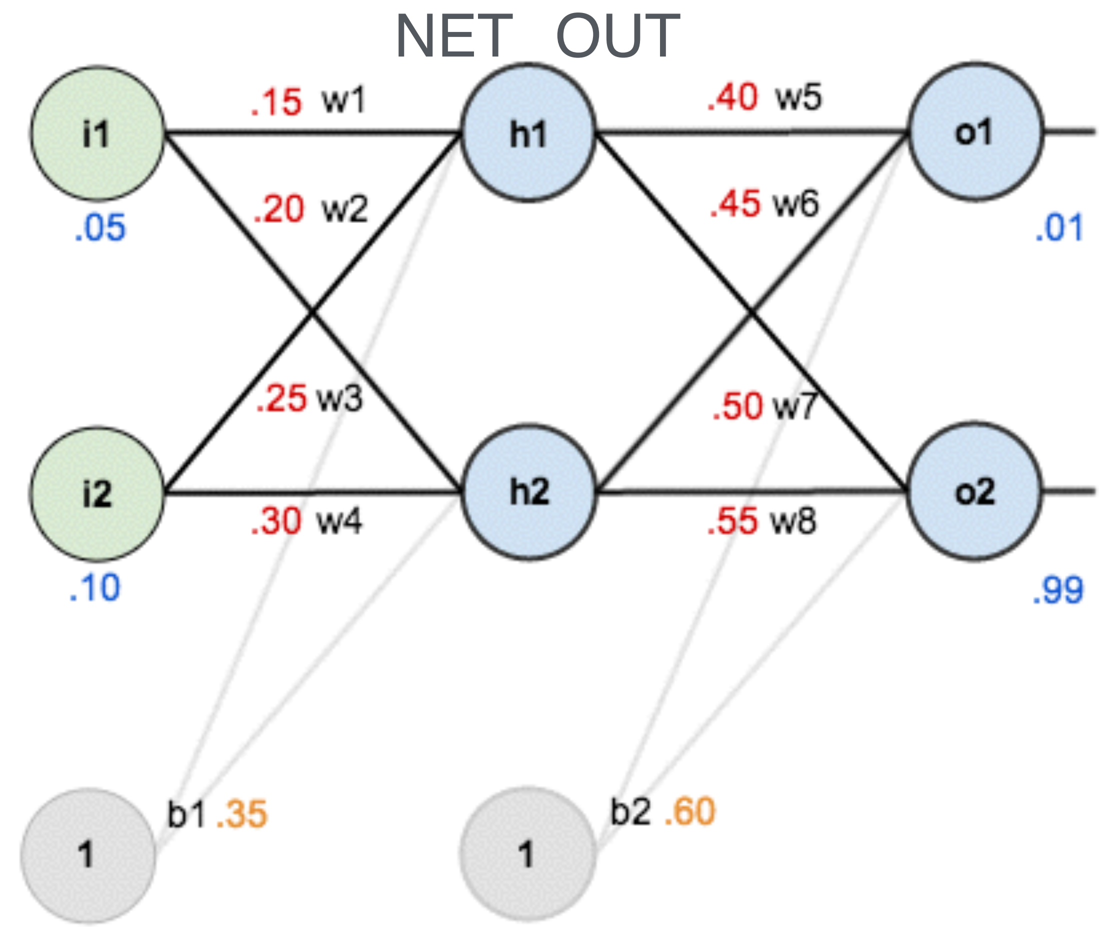
$$\mathbf{b} = \mathbf{a} - \gamma \nabla F(\mathbf{a})$$

Gradient descent update rule

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



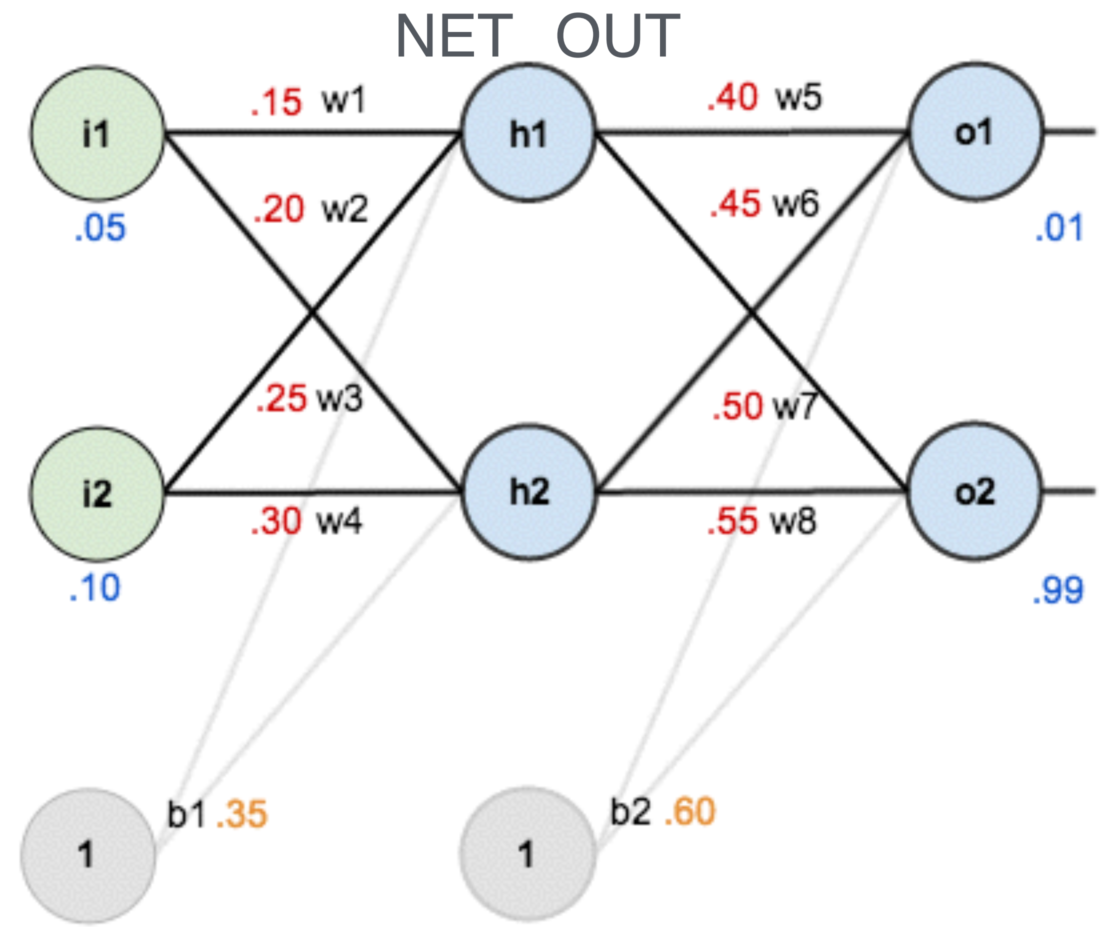
Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation

Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

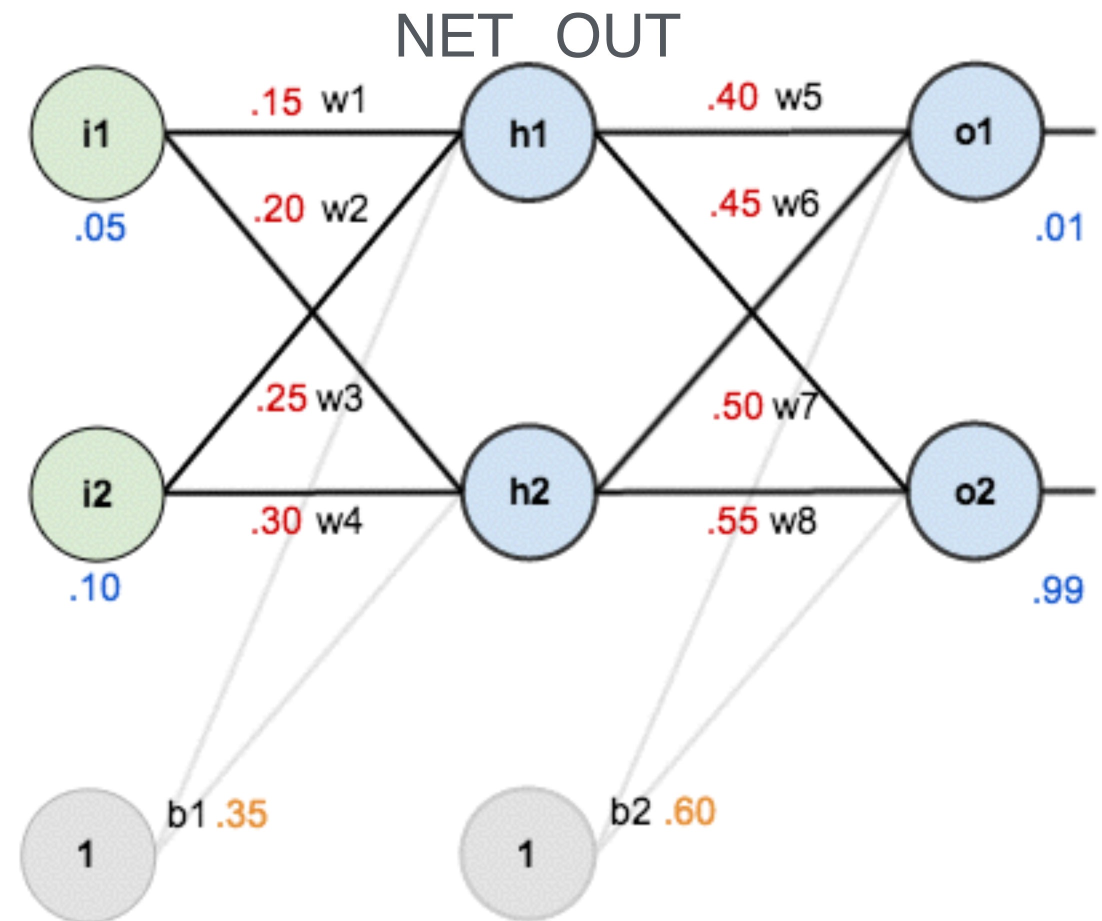


- Repeat for w6, w7, w8

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

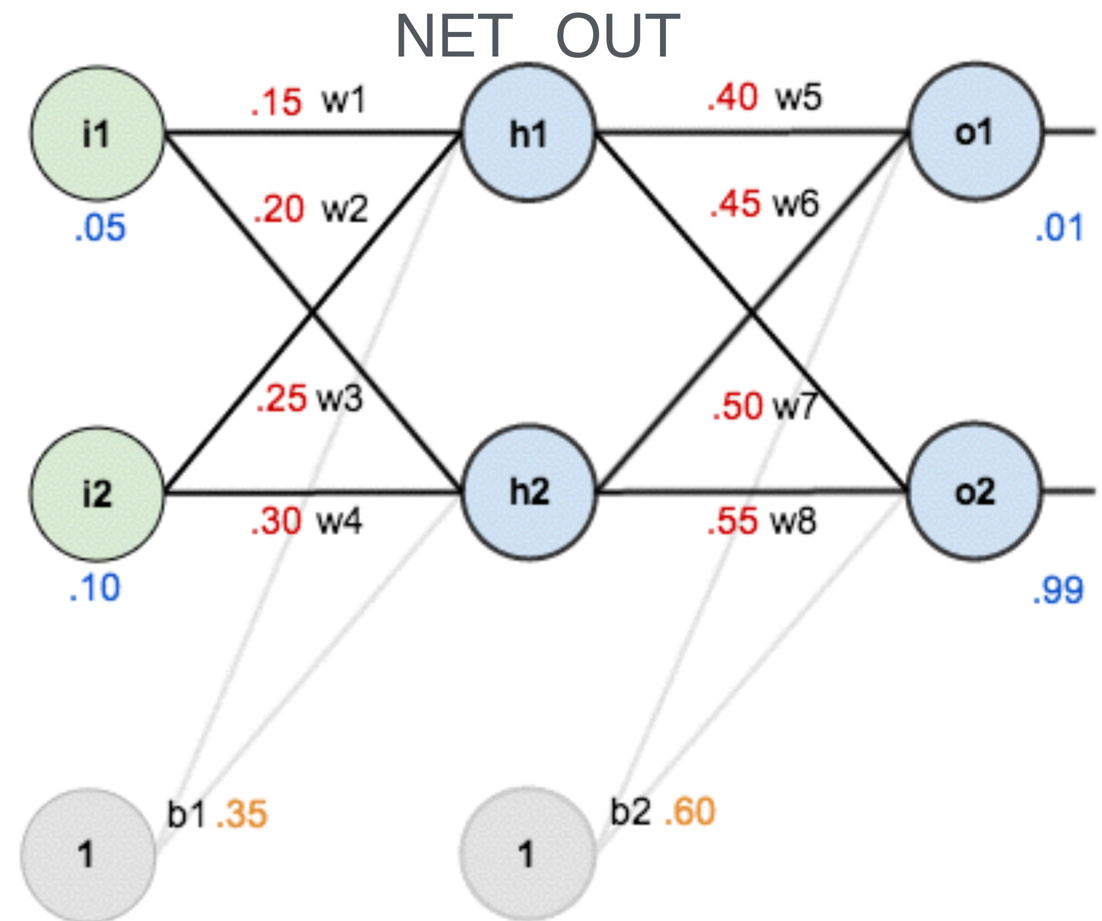
- Repeat for w6, w7, w8
- In analogous way for w1, w2, w3, w4

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns

Backpropagation

Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

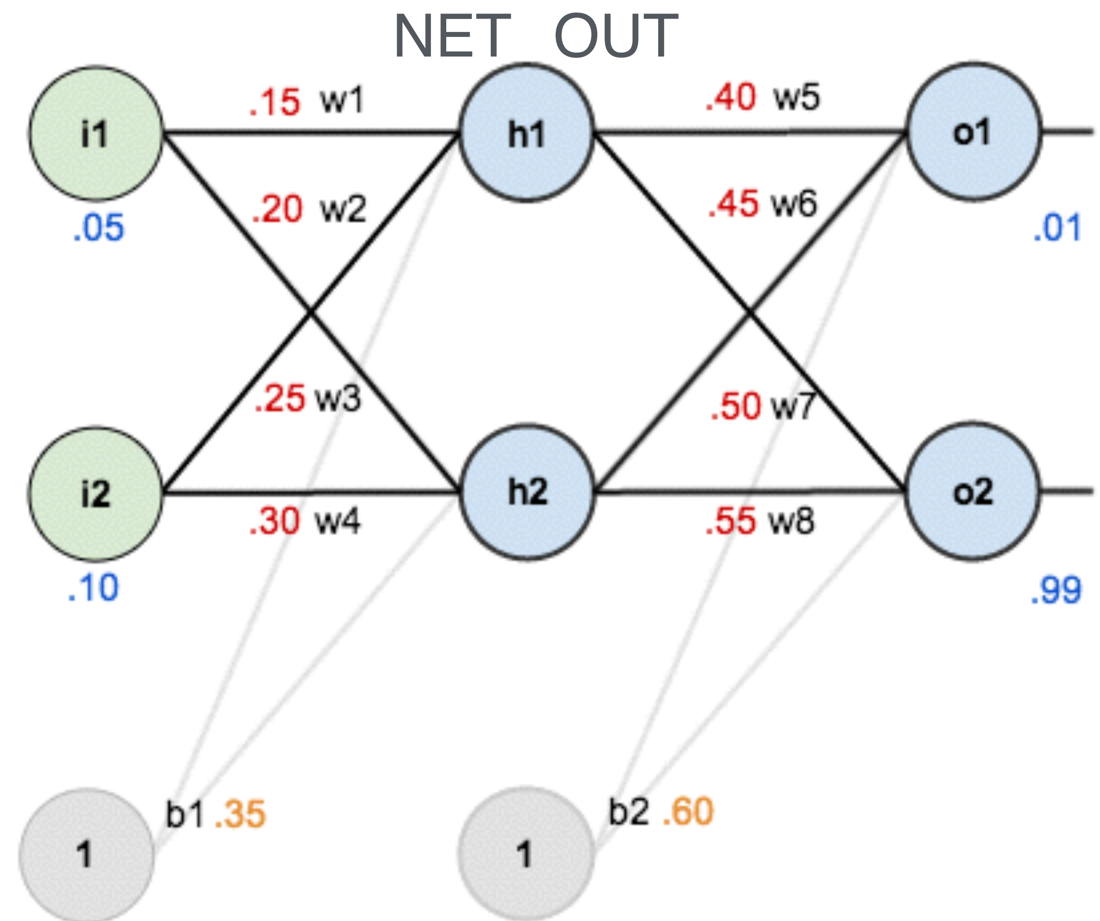


- Repeat for w6, w7, w8
- In analogous way for w1, w2, w3, w4
- Calculate the total error again: 0.291027924

  it was: 0.298371109

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

How it learns
Backpropagation



Given inputs 0.05 and 0.10, we want the neural network to output 0.01 and 0.99

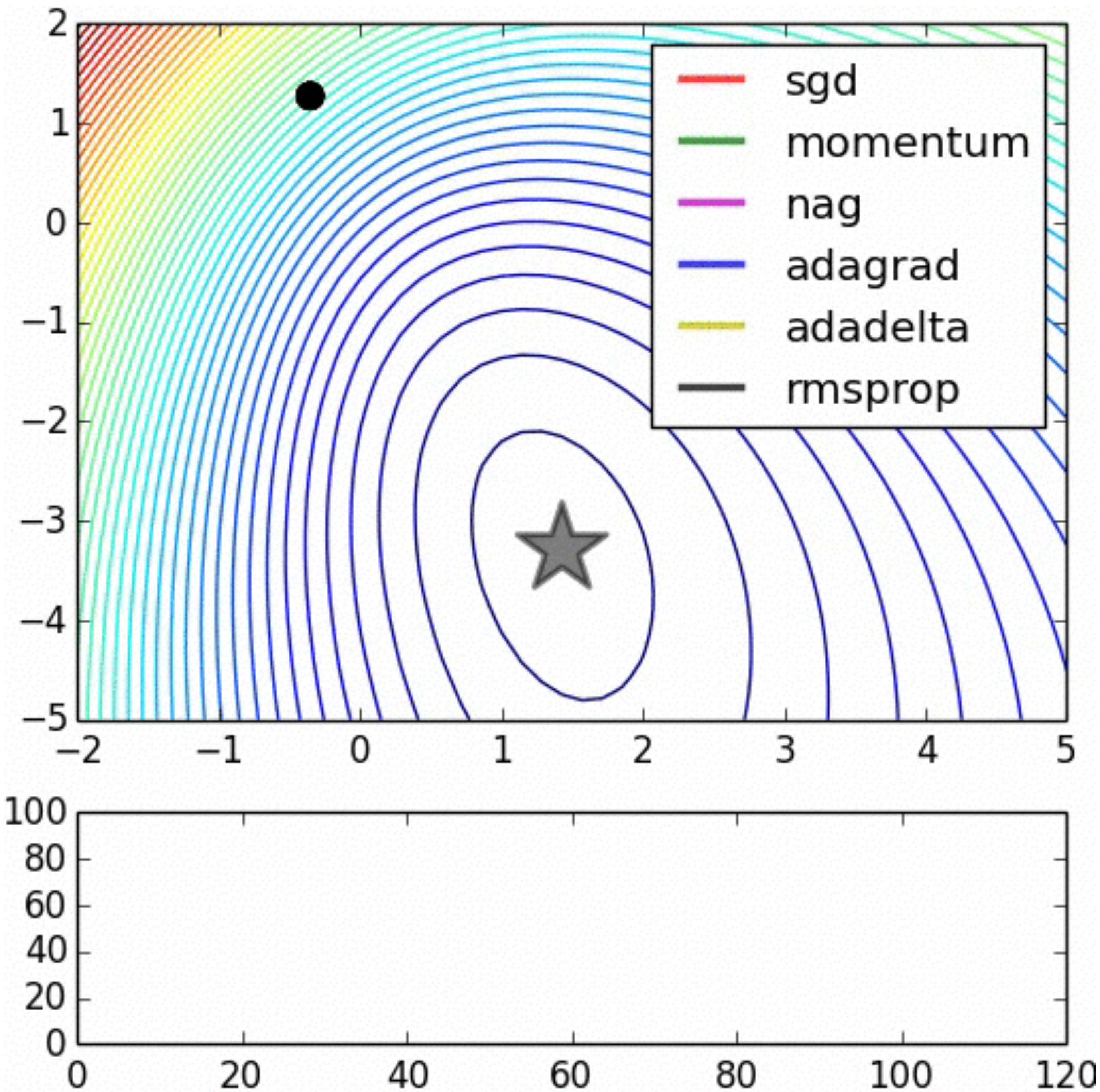
- Repeat for w6, w7, w8
- In analogous way for w1, w2, w3, w4
- Calculate the total error again: 0.291027924
  it was: 0.298371109

- Repeat 10,000 times: 0.000035085

http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example

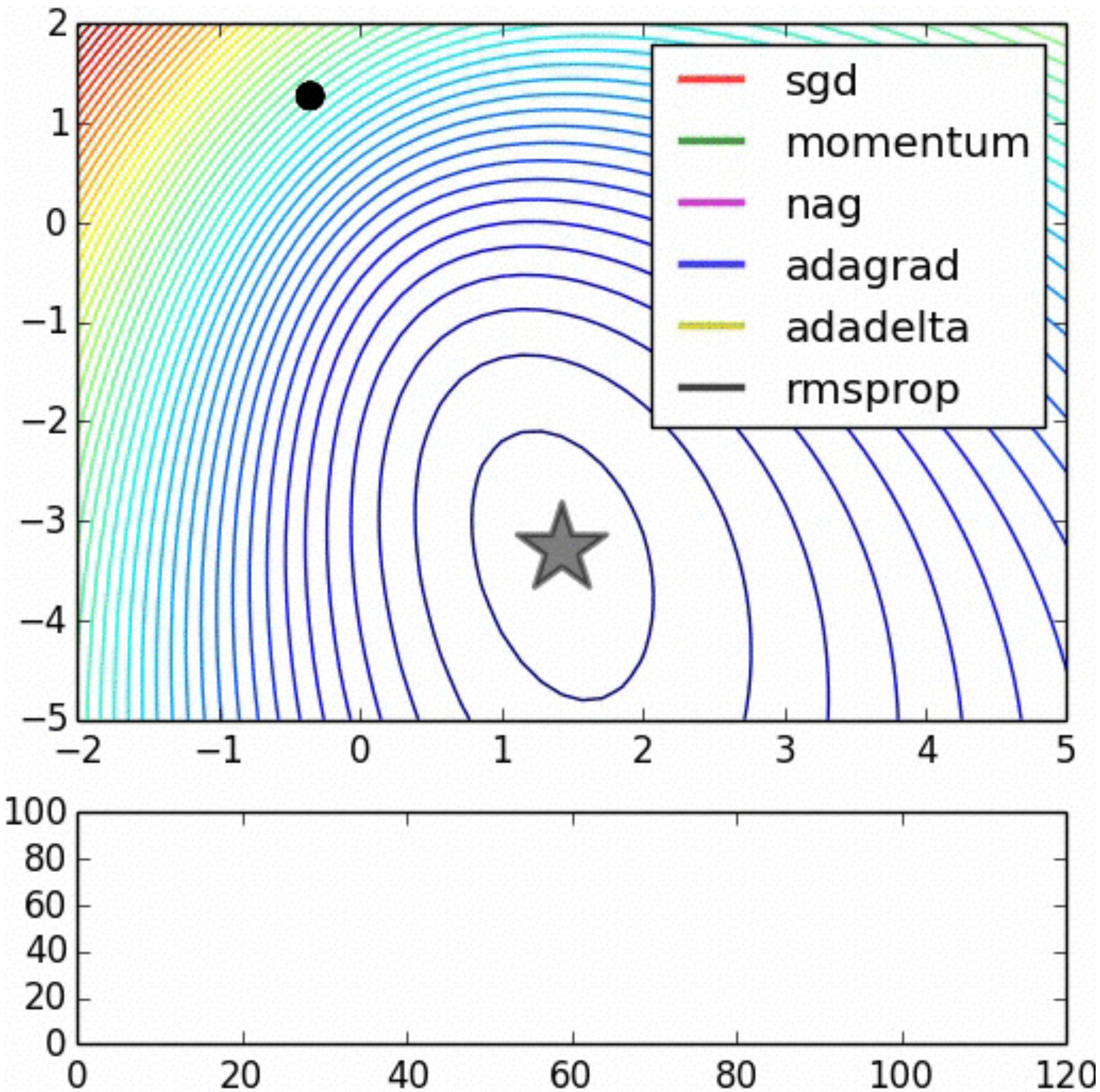
How it learns

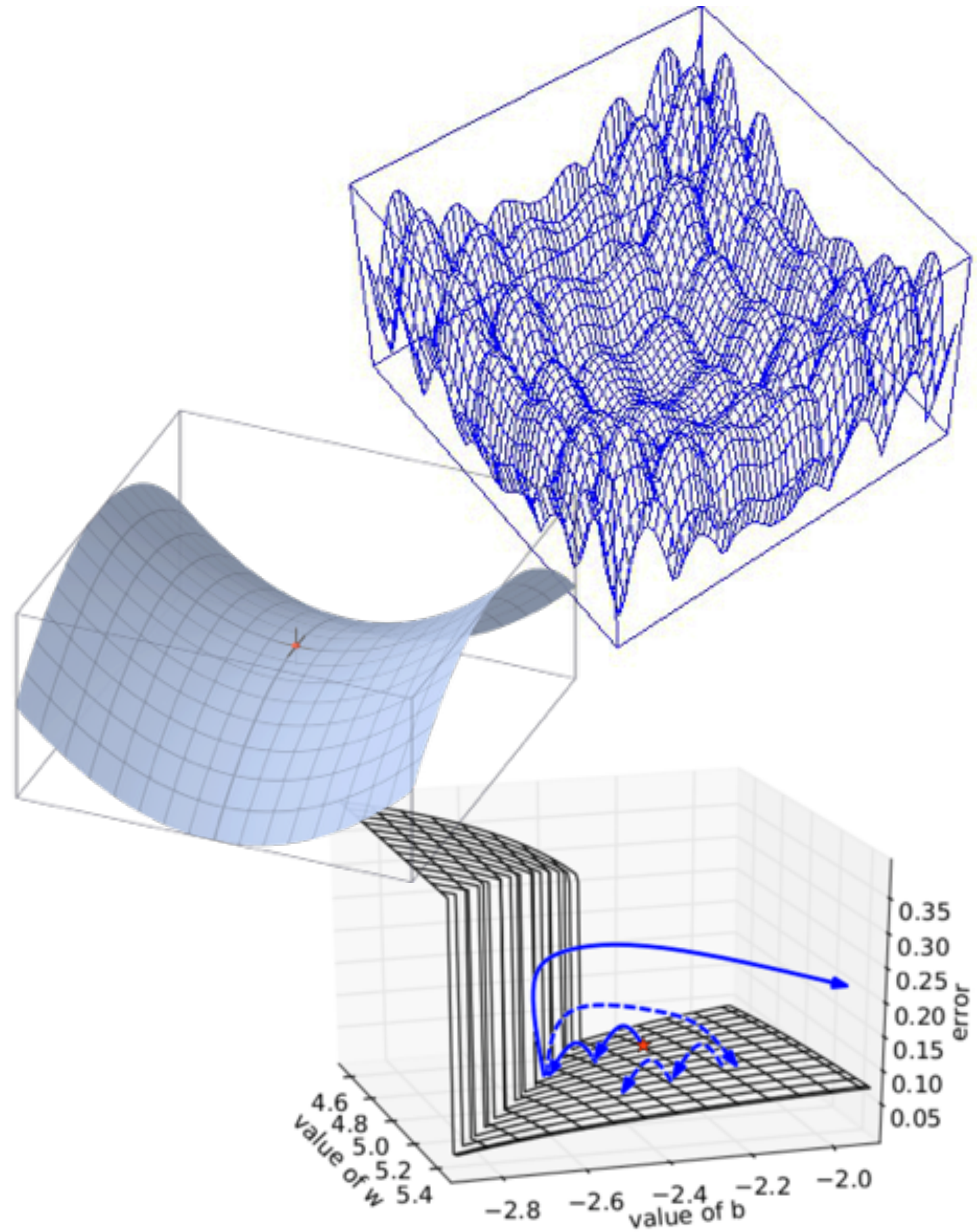
Optimization methods



Alec Radford “Introduction to Deep Learning with Python”

How it learns
Optimization methods



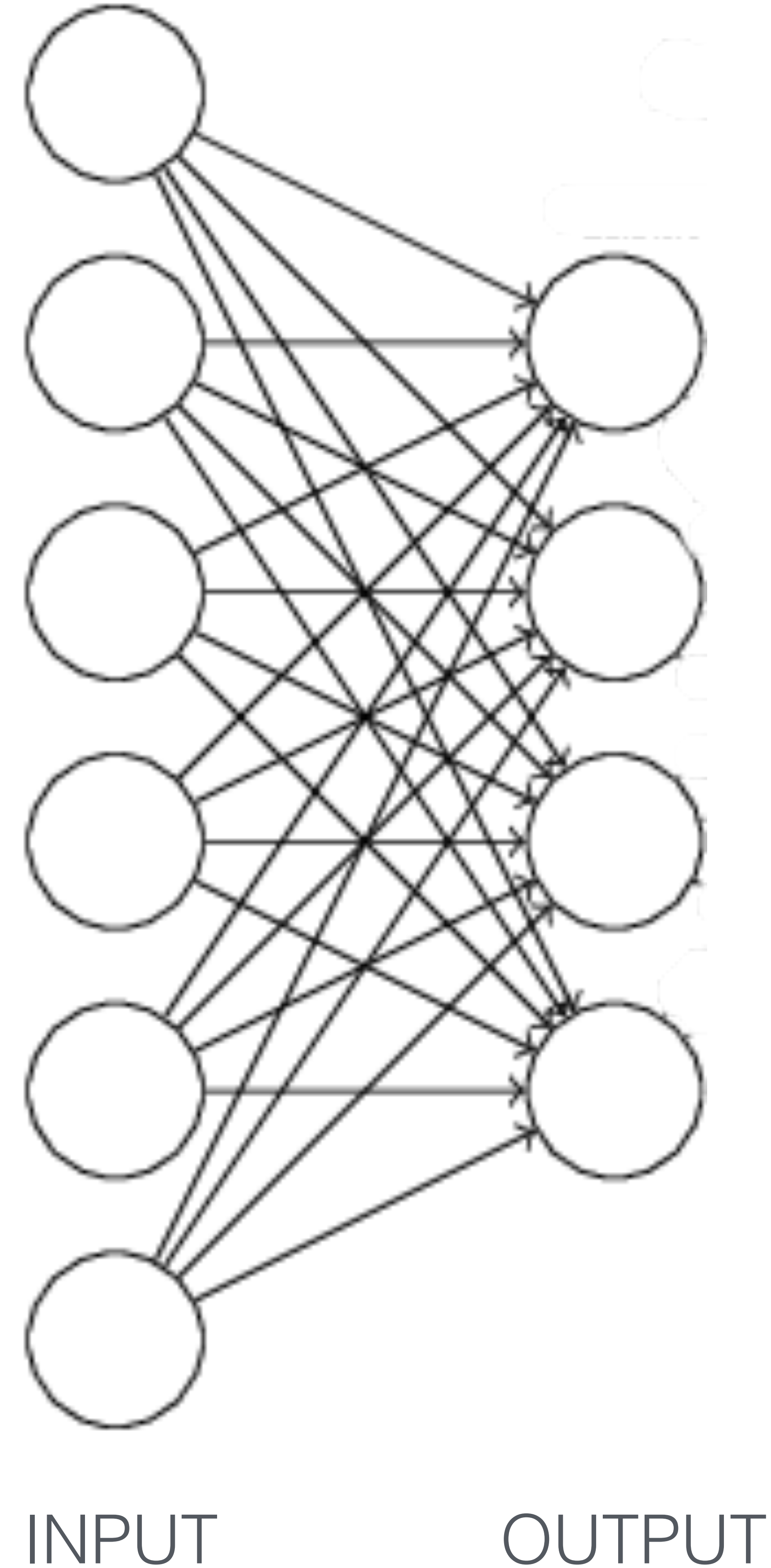


Alec Radford “Introduction to Deep Learning with Python”

How it evolved

How it evolved

1-layer NN

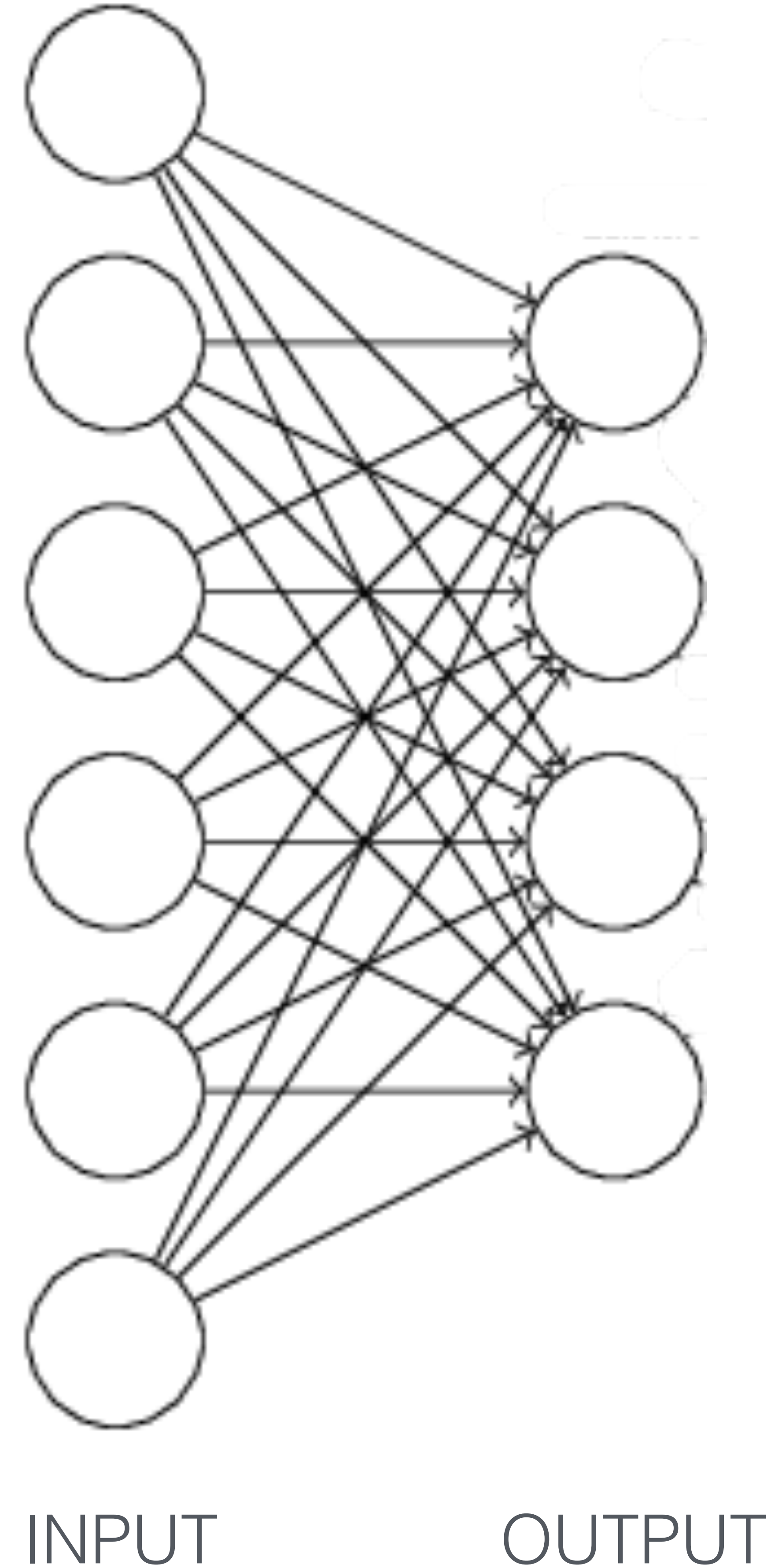


Alec Radford “Introduction to Deep Learning with Python”

How it evolved

1-layer NN

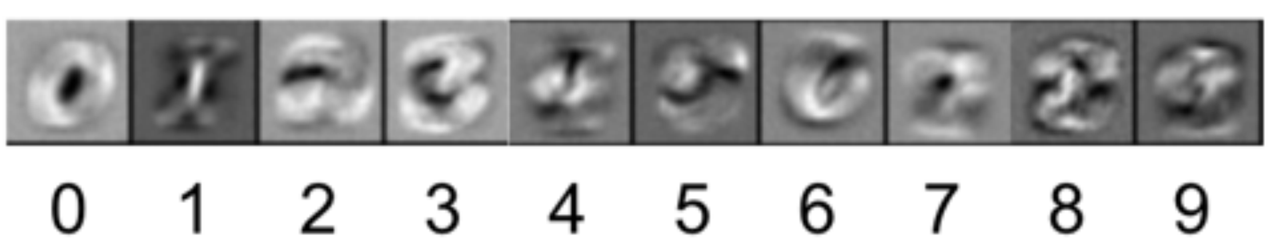
92.5% on the MNIST test set

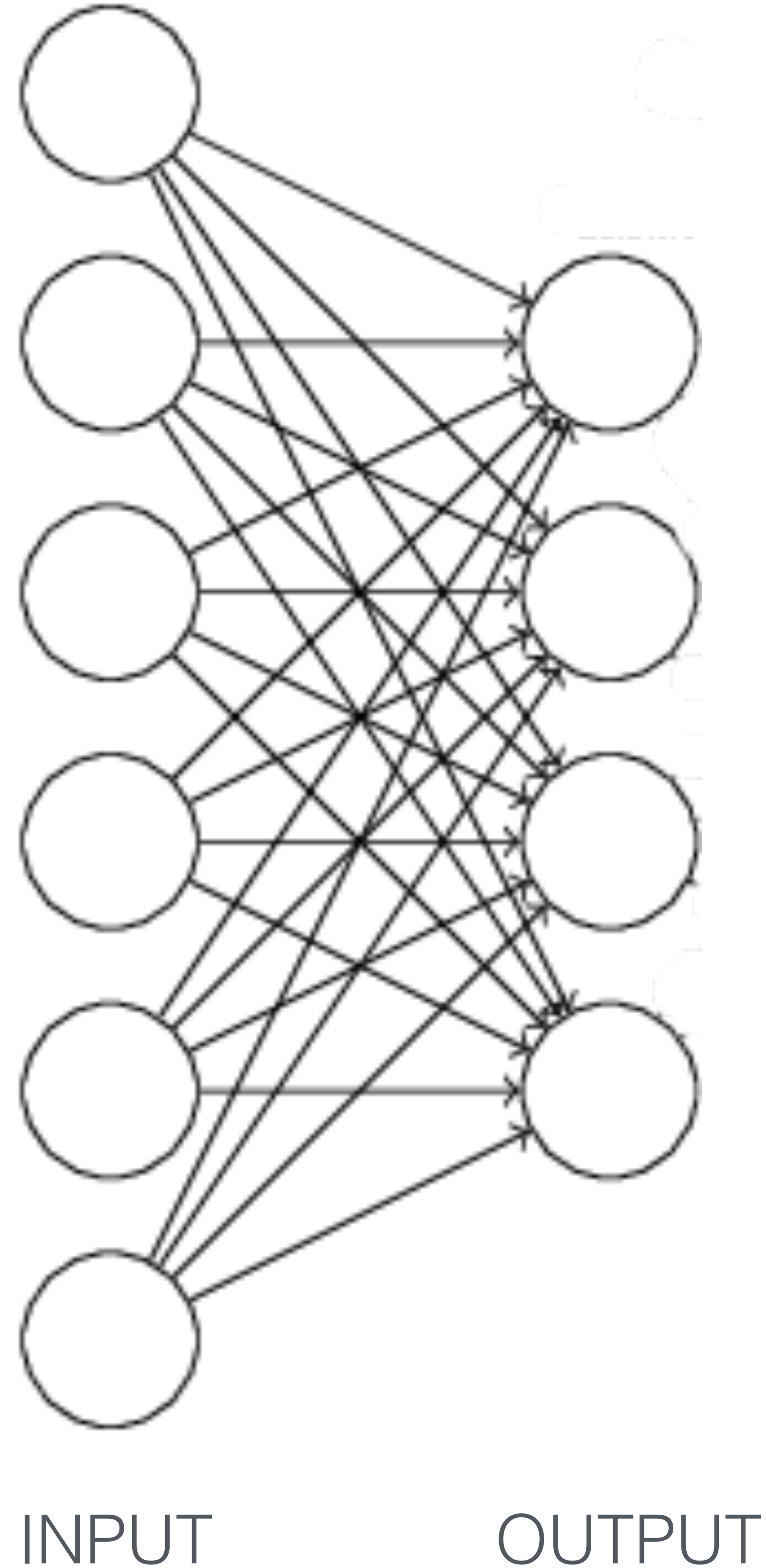


Alec Radford "Introduction to Deep Learning with Python"

How it evolved
1-layer NN

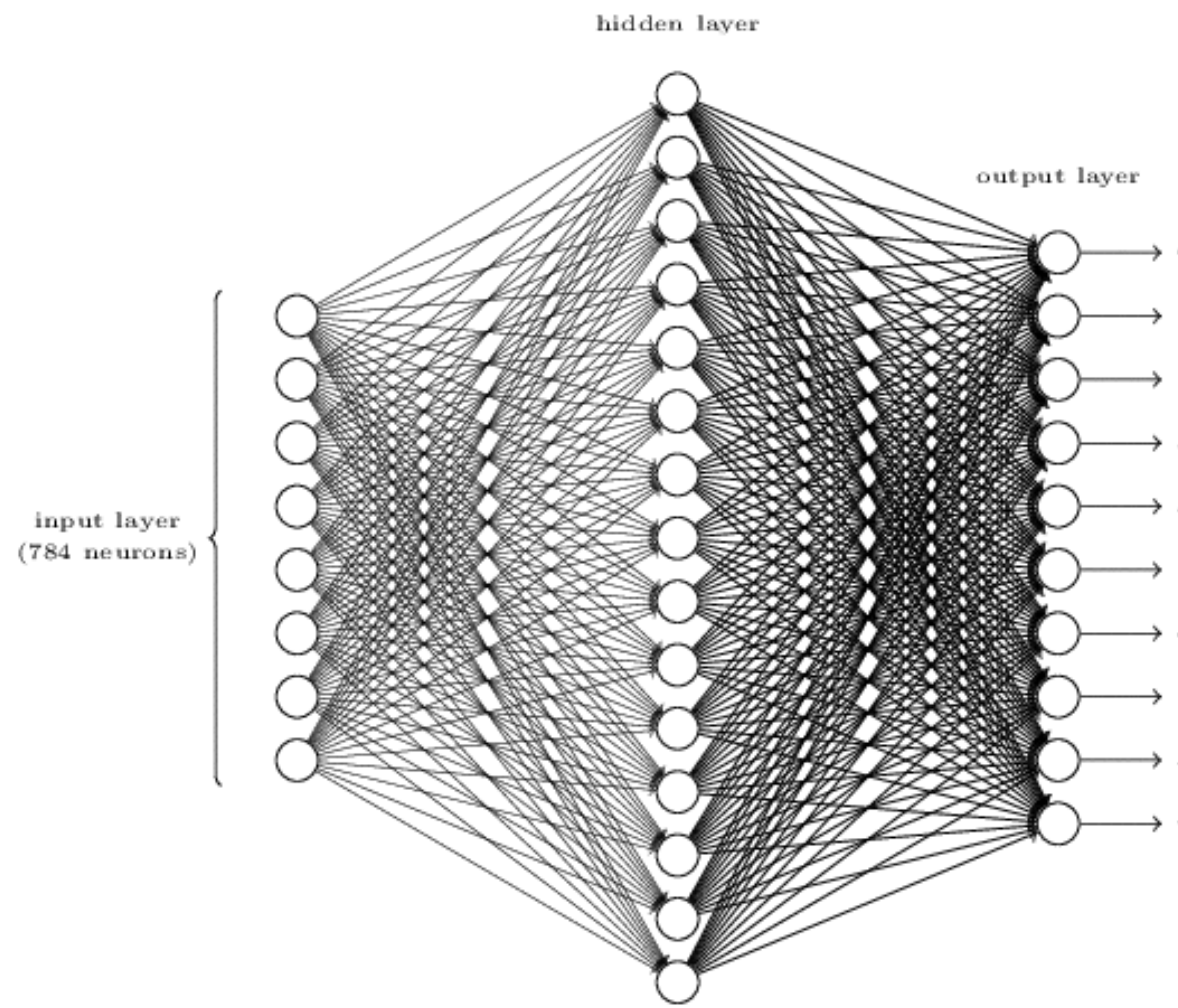
92.5% on the MNIST test set





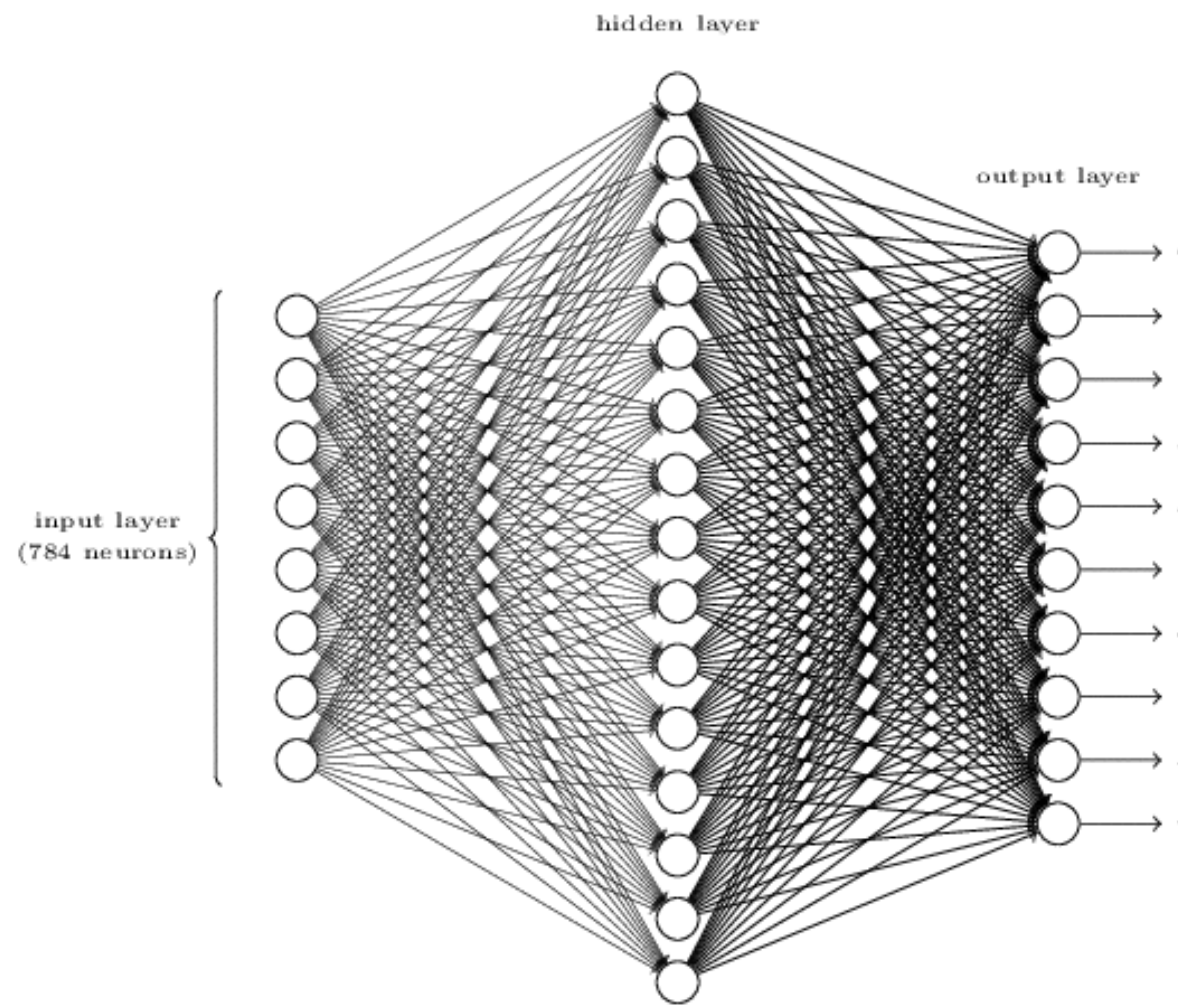
Alec Radford "Introduction to Deep Learning with Python"

How it evolved
One hidden layer



Alec Radford "Introduction to Deep Learning with Python"

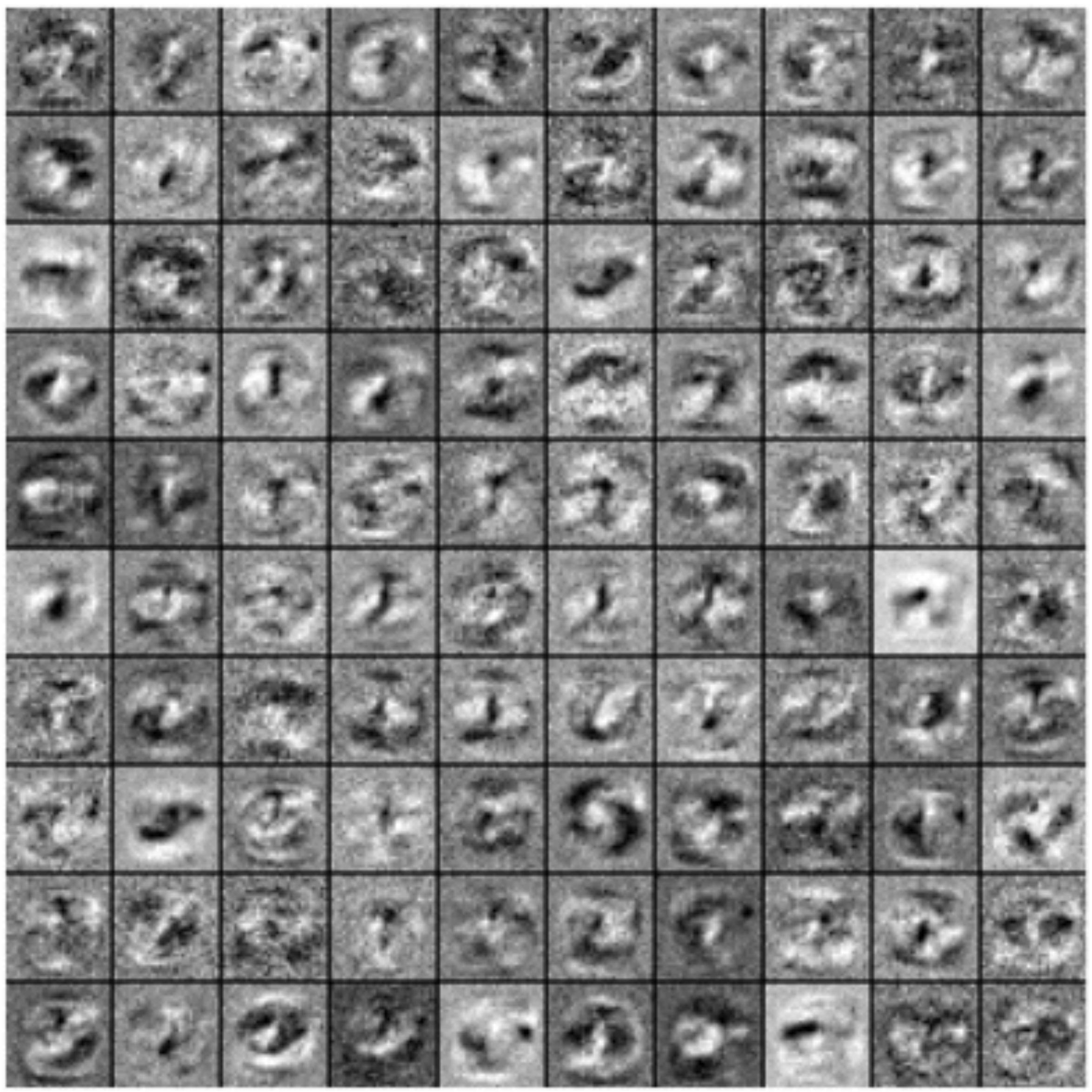
How it evolved
One hidden layer



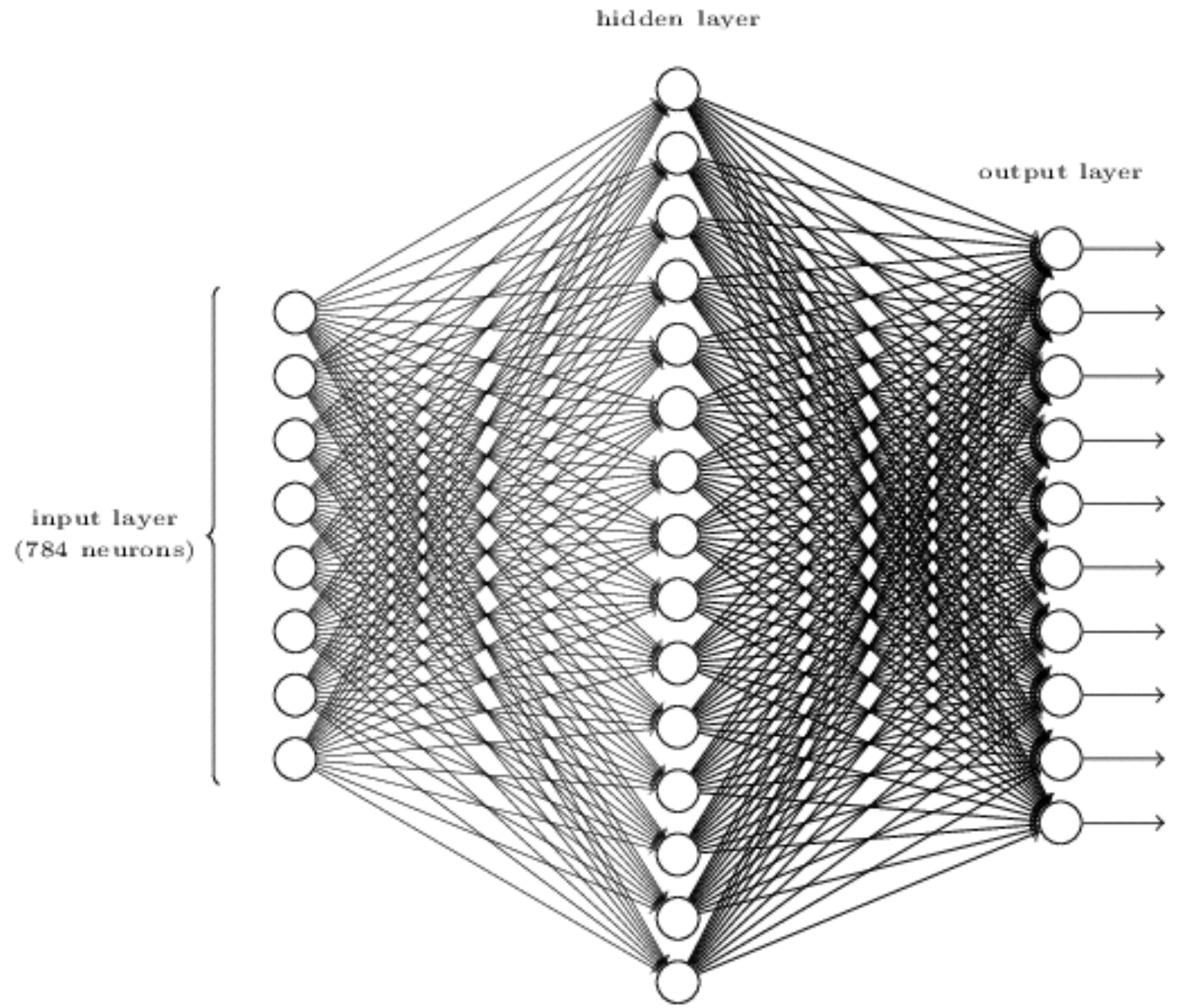
98.2% on the MNIST test set

Alec Radford "Introduction to Deep Learning with Python"

How it evolved
One hidden layer



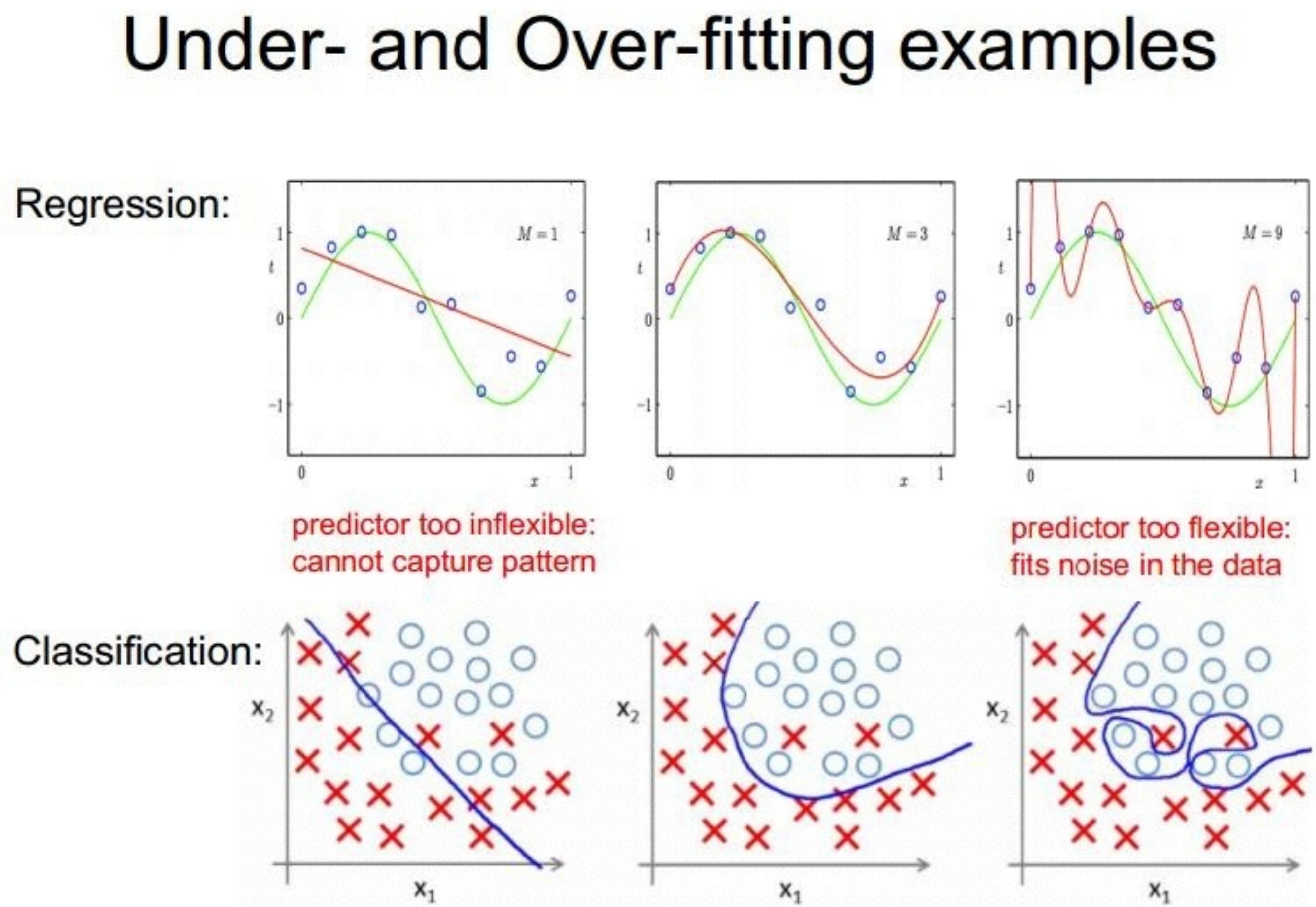
Activity of a 100 hidden neurons (out of 625)



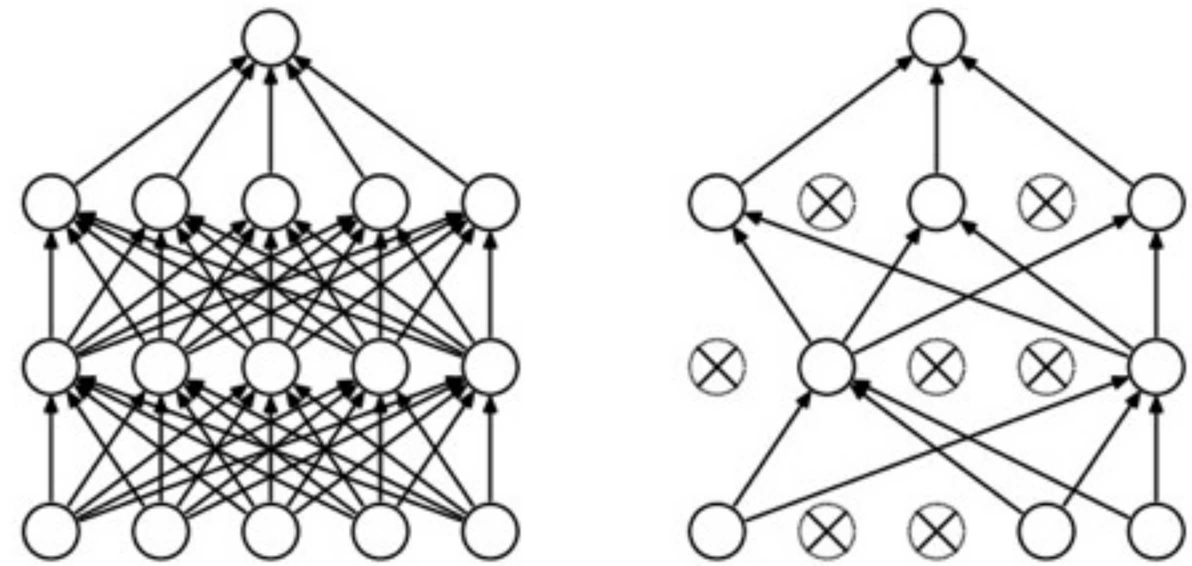
98.2% on the MNIST test set

Alec Radford "Introduction to Deep Learning with Python"

How it evolved
Overfitting



How it evolved
Dropout


(a) Standard Neural Net (b) After applying dropout.

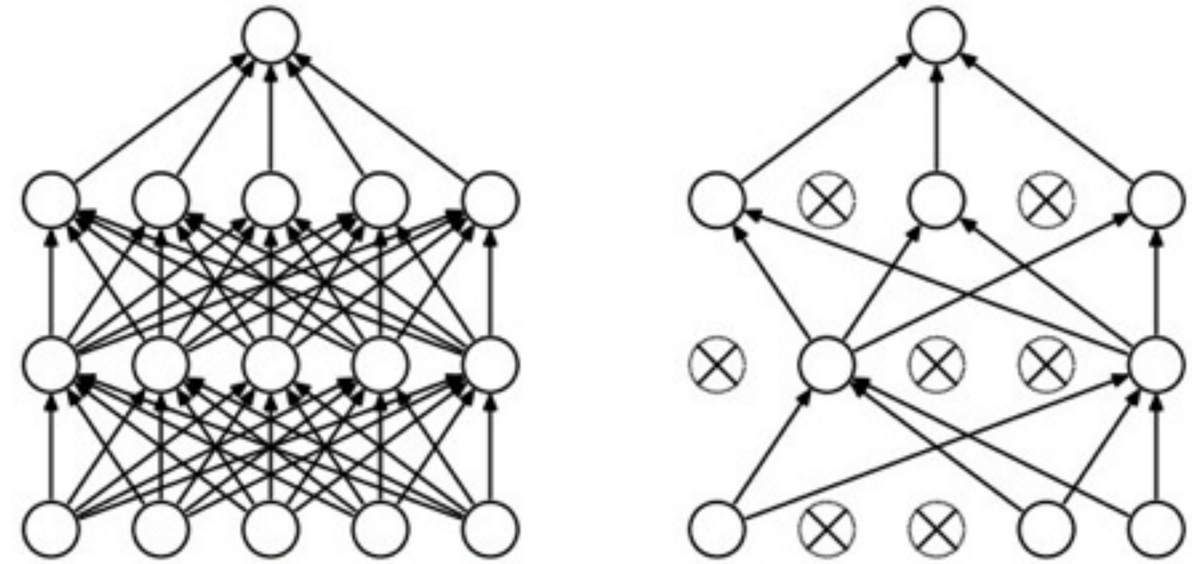
Srivastava, Hinton, Krizhevsky, Sutskever, Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, 2014

How it evolved

Dropout

Method	Test Classification error %
L2	1.62
L2 + L1 applied towards the end of training	1.60
L2 + KL-sparsity	1.55
Max-norm	1.35
Dropout + L2	1.25
Dropout + Max-norm	**1.05**

Table 9: Comparison of different regularization methods on MNIST.



(a) Standard Neural Net (b) After applying dropout.

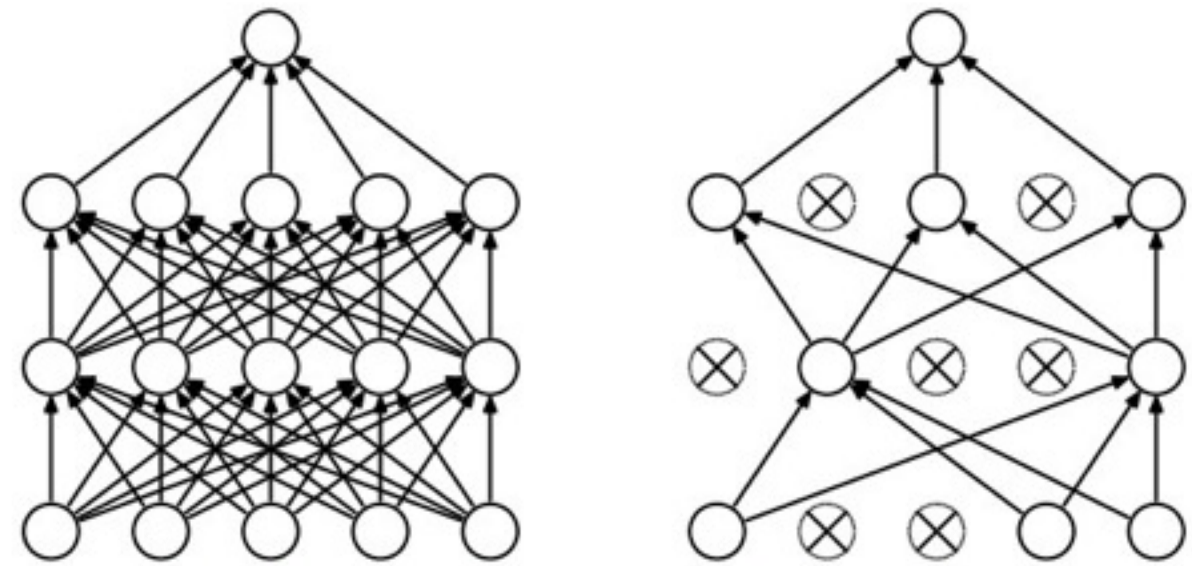
Srivastava, Hinton, Krizhevsky, Sutskever, Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, 2014

How it evolved

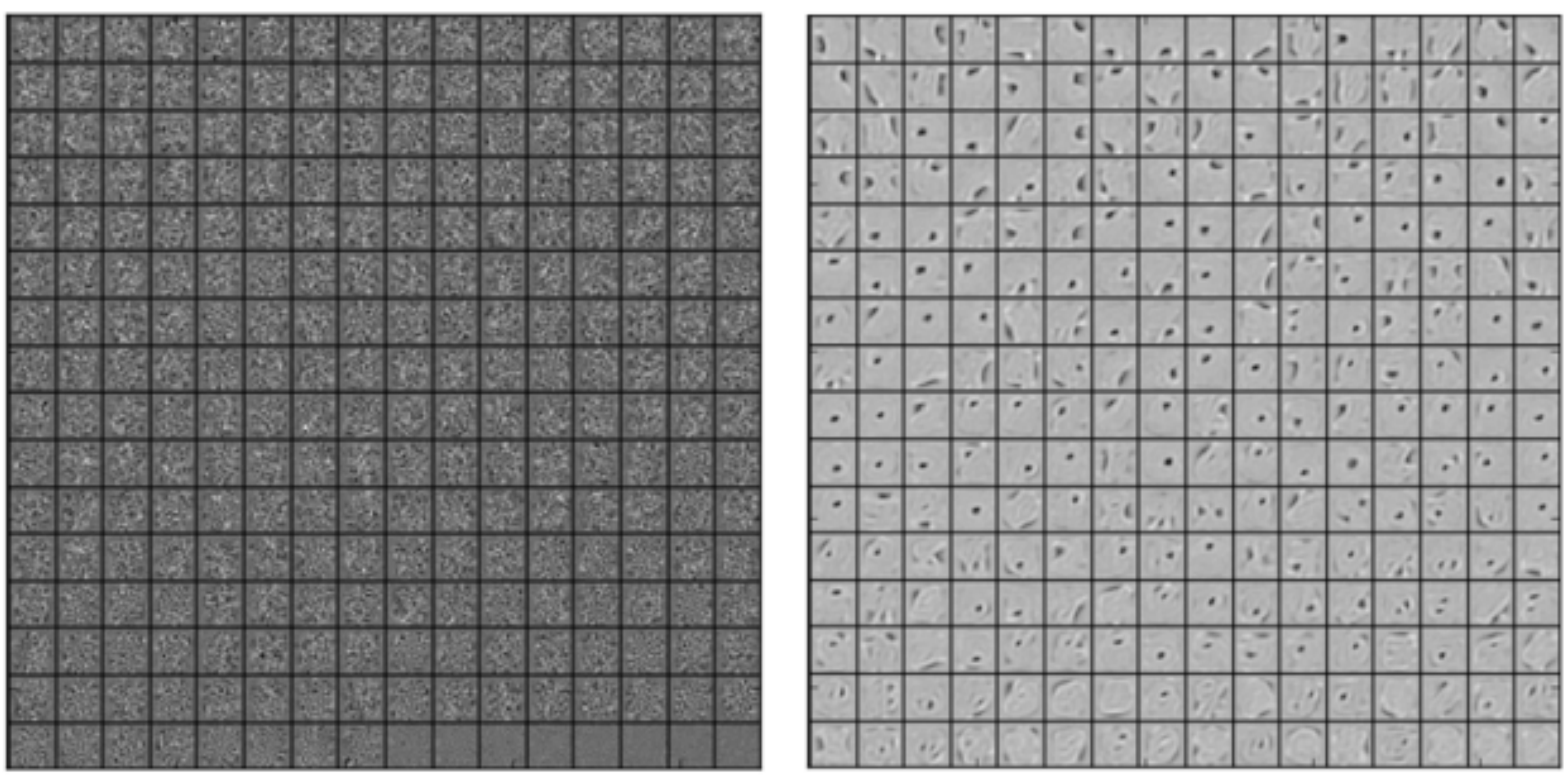
Dropout

Method	Test Classification error %
L2	1.62
L2 + L1 applied towards the end of training	1.60
L2 + KL-sparsity	1.55
Max-norm	1.35
Dropout + L2	1.25
Dropout + Max-norm	**1.05**

Table 9: Comparison of different regularization methods on MNIST.



(a) Standard Neural Net (b) After applying dropout.



(a) Without dropout (b) Dropout with $p = 0.5$.

Figure 7: Features learned on MNIST with one hidden layer autoencoders having 256 rectified linear units.

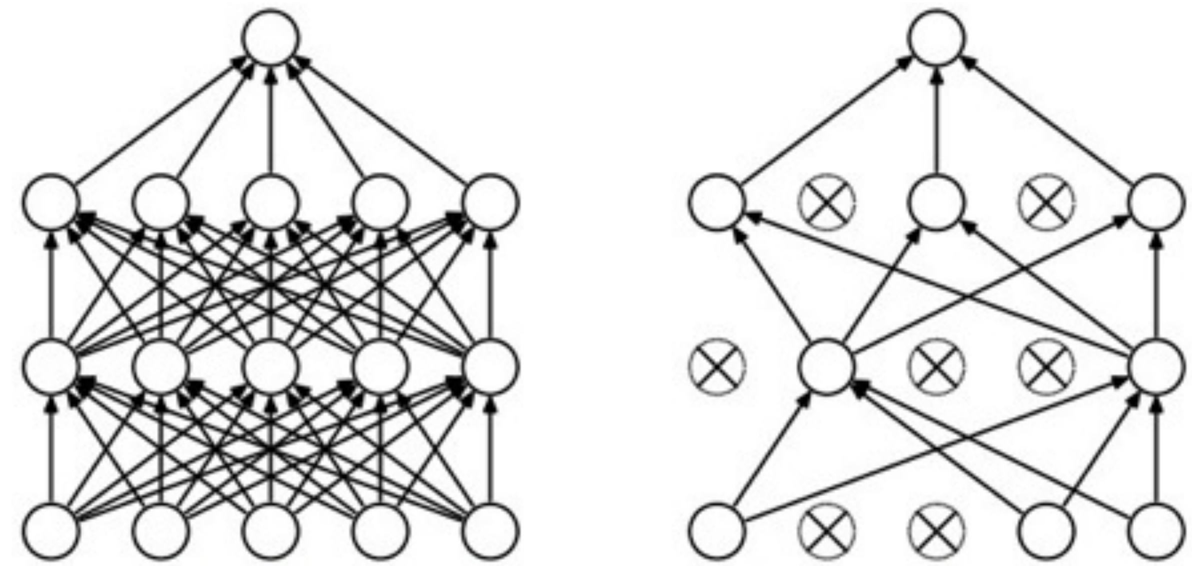
Srivastava, Hinton, Krizhevsky, Sutskever, Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, 2014

How it evolved

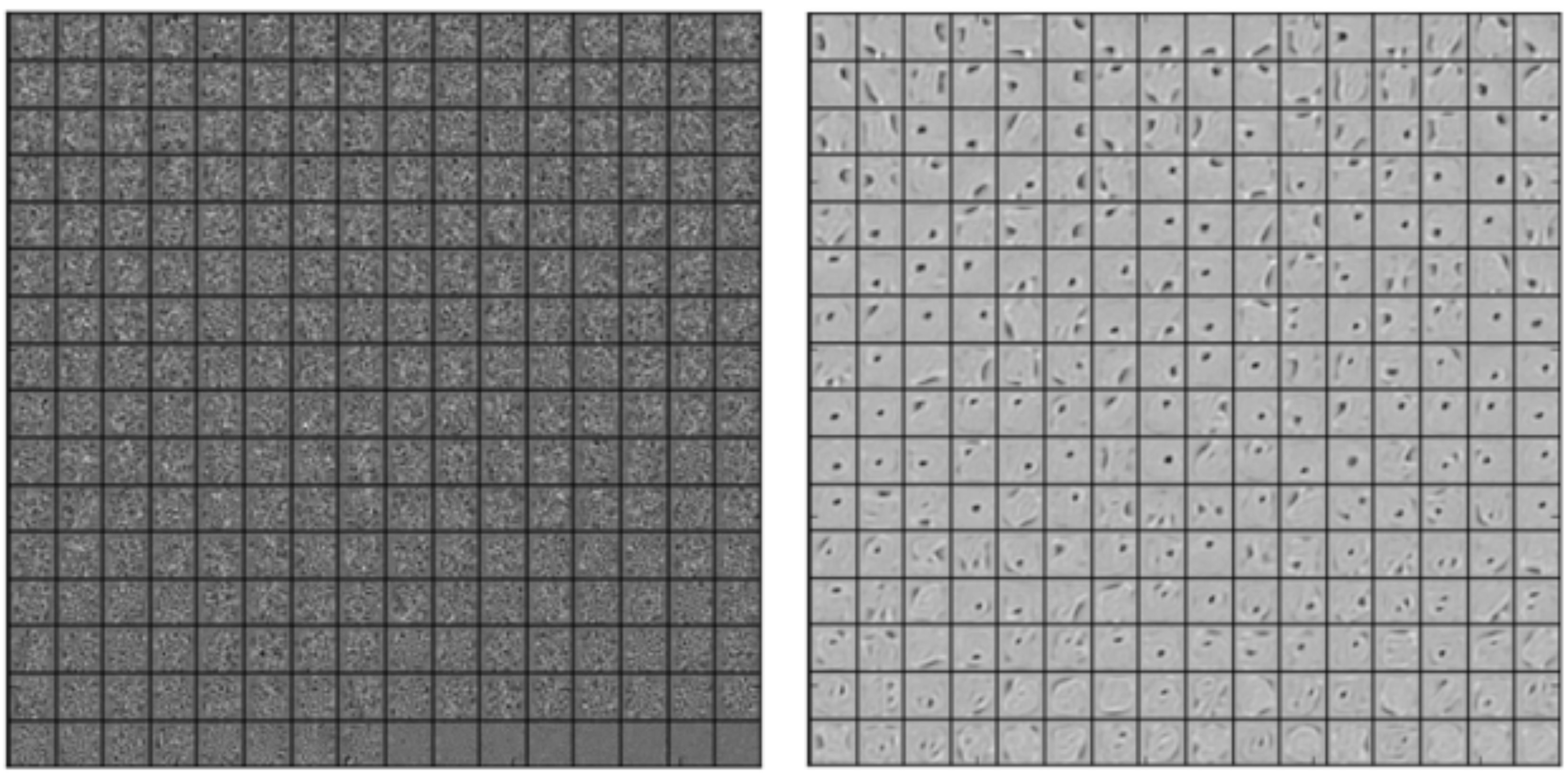
Dropout

Method	Test Classification error %
L2	1.62
L2 + L1 applied towards the end of training	1.60
L2 + KL-sparsity	1.55
Max-norm	1.35
Dropout + L2	1.25
Dropout + Max-norm	**1.05**

Table 9: Comparison of different regularization methods on MNIST.



(a) Standard Neural Net (b) After applying dropout.



(a) Without dropout (b) Dropout with $p = 0.5$.

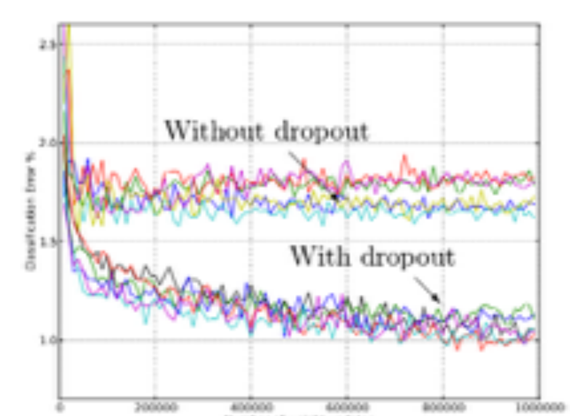
Figure 7: Features learned on MNIST with one hidden layer autoencoders having 256 rectified linear units.

Method	Phone Error Rate%
NN (6 layers) (Mohamed et al., 2010)	23.4
Dropout NN (6 layers)	21.8
DBN-pretrained NN (4 layers)	22.7
DBN-pretrained NN (8 layers) (Mohamed et al., 2010)	20.7
DBN-pretrained NN (4 layers) + dropout	**19.7**
DBN-pretrained NN (8 layers) + dropout	**19.7**

Table 7: Phone error rate on the TIMIT core test set.

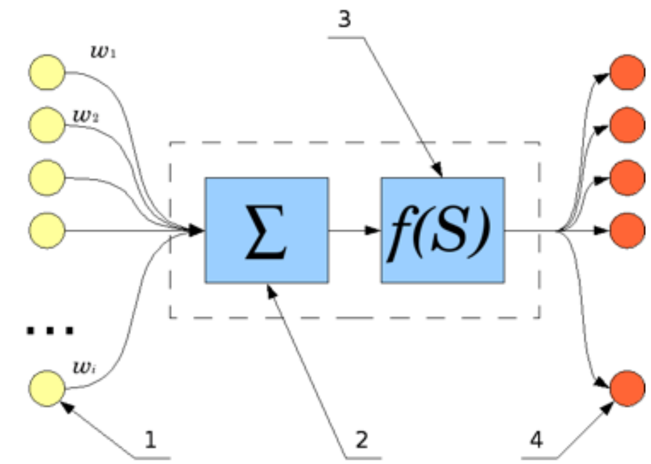
Method	CIFAR-10	CIFAR-100
Conv Net + max pooling (hand tuned)	15.60	43.48
Conv Net + stochastic pooling (Zeiler and Fergus, 2013)	15.13	42.51
Conv Net + max pooling (Snoek et al., 2012)	14.98	-
Conv Net + max pooling + dropout fully connected layers	14.32	41.26
Conv Net + max pooling + dropout in all layers	12.61	**37.20**
Conv Net + maxout (Goodfellow et al., 2013)	**11.68**	38.57

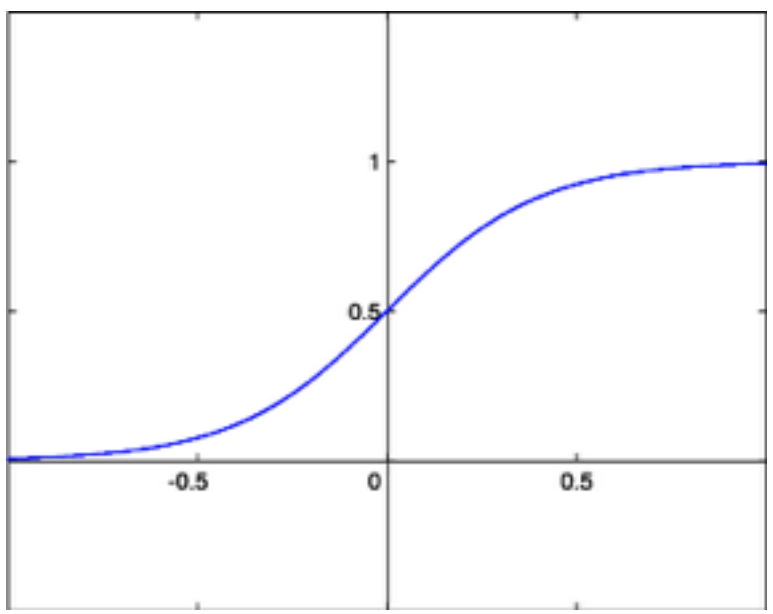
Table 4: Error rates on CIFAR-10 and CIFAR-100.

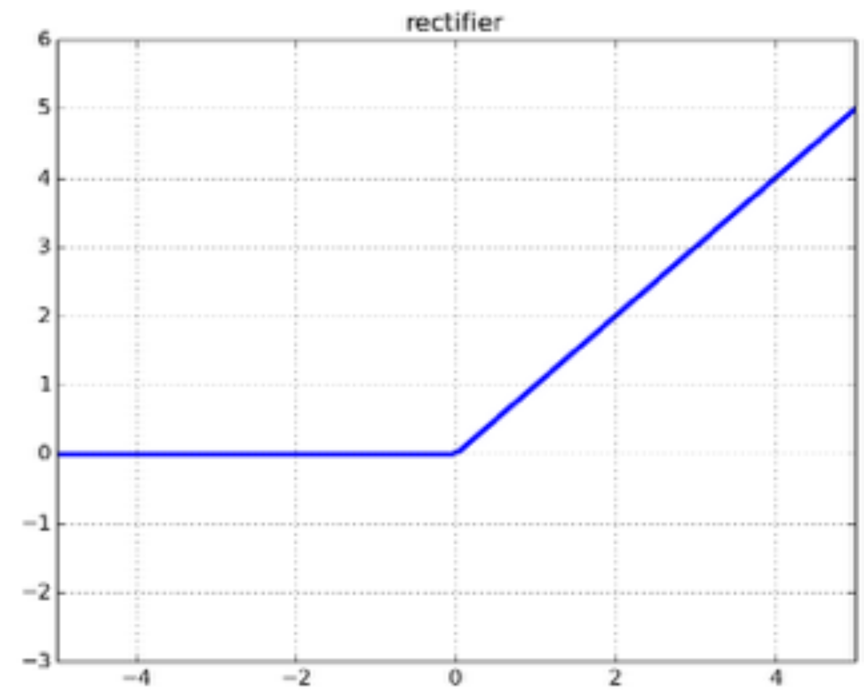


Srivastava, Hinton, Krizhevsky, Sutskever, Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, 2014

How it evolved
ReLU

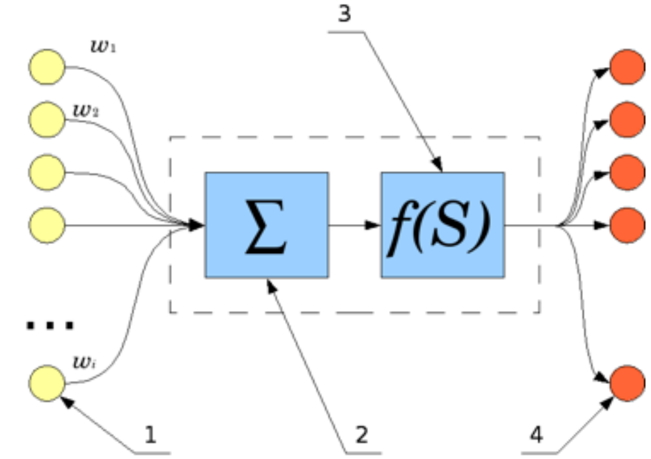


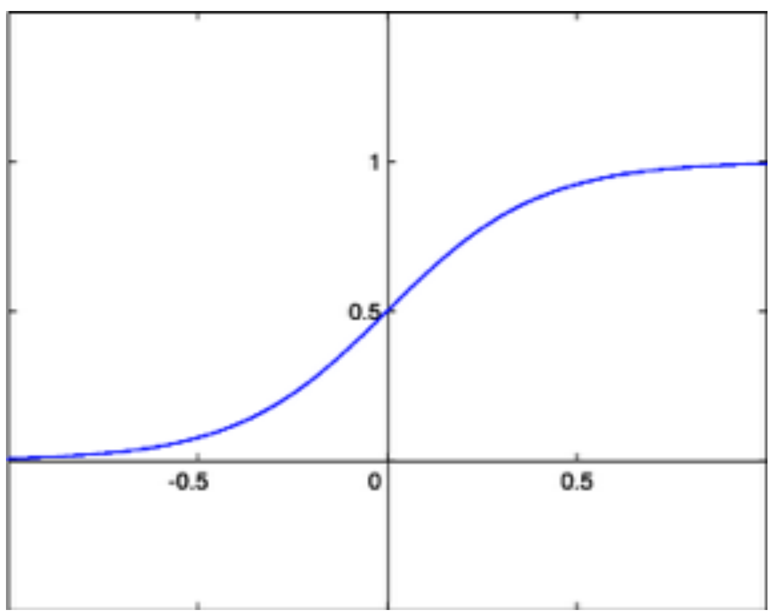


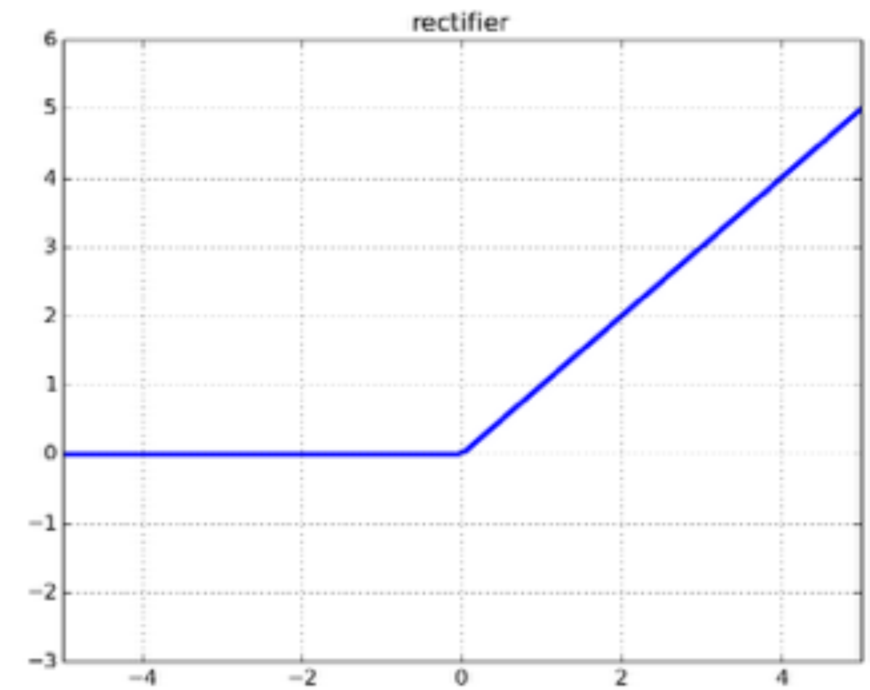


X. Glorot, A. Bordes, Y. Bengio, “Deep Sparse Rectifier Neural Networks”, 2011

How it evolved
ReLU







Neuron	MNIST	CIFAR10	NISTP	NORB
***Without* unsupervised pre-training**				
Rectifier	**1.43%**	**50.86%**	**32.64%**	**16.40%**
Tanh	1.57%	52.62%	36.46%	19.29%
Softplus	1.77%	53.20%	35.48%	17.68%

X. Glorot, A. Bordes, Y. Bengio, “Deep Sparse Rectifier Neural Networks”, 2011

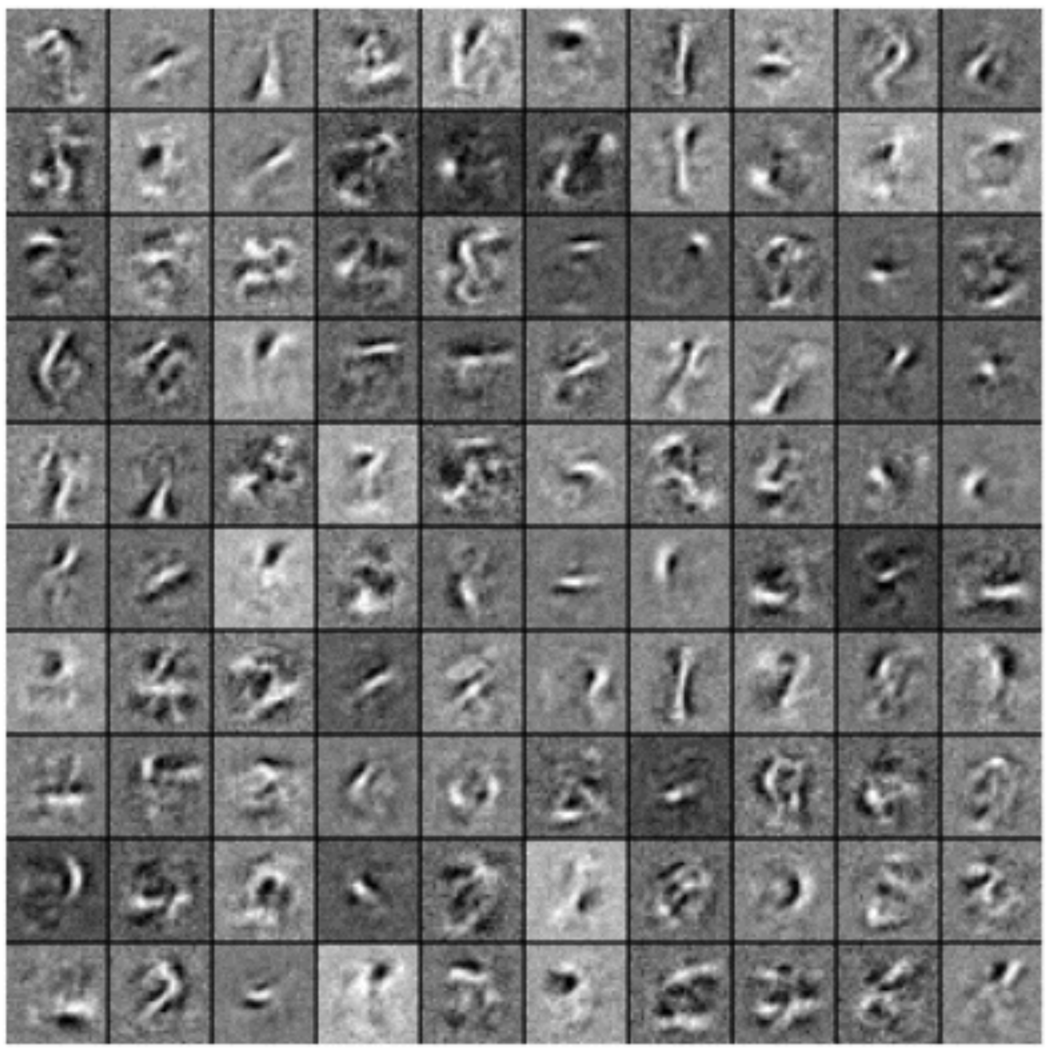
How it evolved

“Modern” ANN

- Several hidden layers
- ReLU activation units
- Dropout

How it evolved

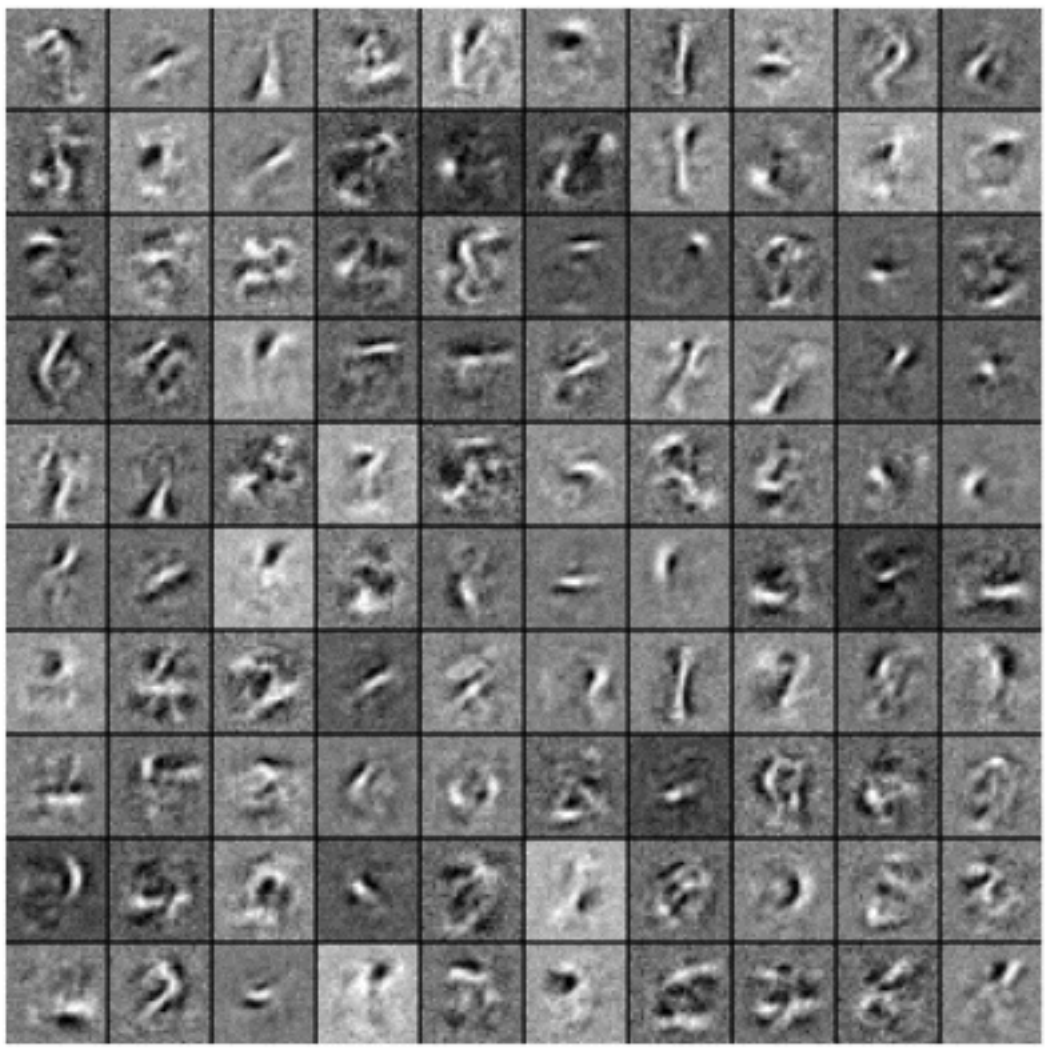
“Modern” ANN



- Several hidden layers
- ReLU activation units
- Dropout

How it evolved

“Modern” ANN



- Several hidden layers
- ReLU activation units
- Dropout

99.0% on the MNIST test set

How it evolved

Convolution

How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

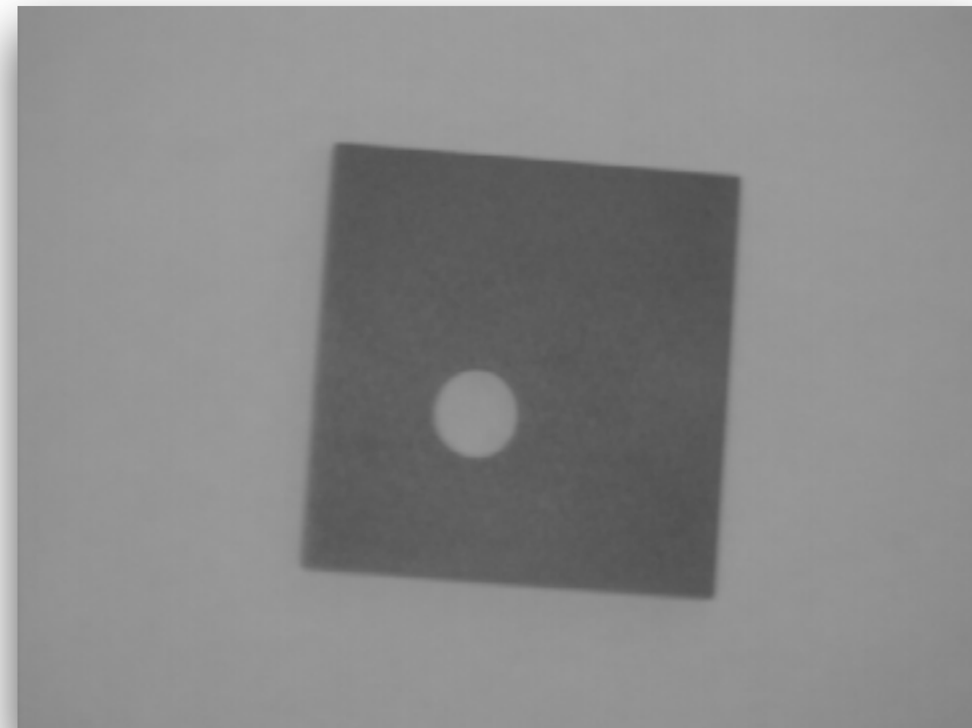
Prewitt edge detector

How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

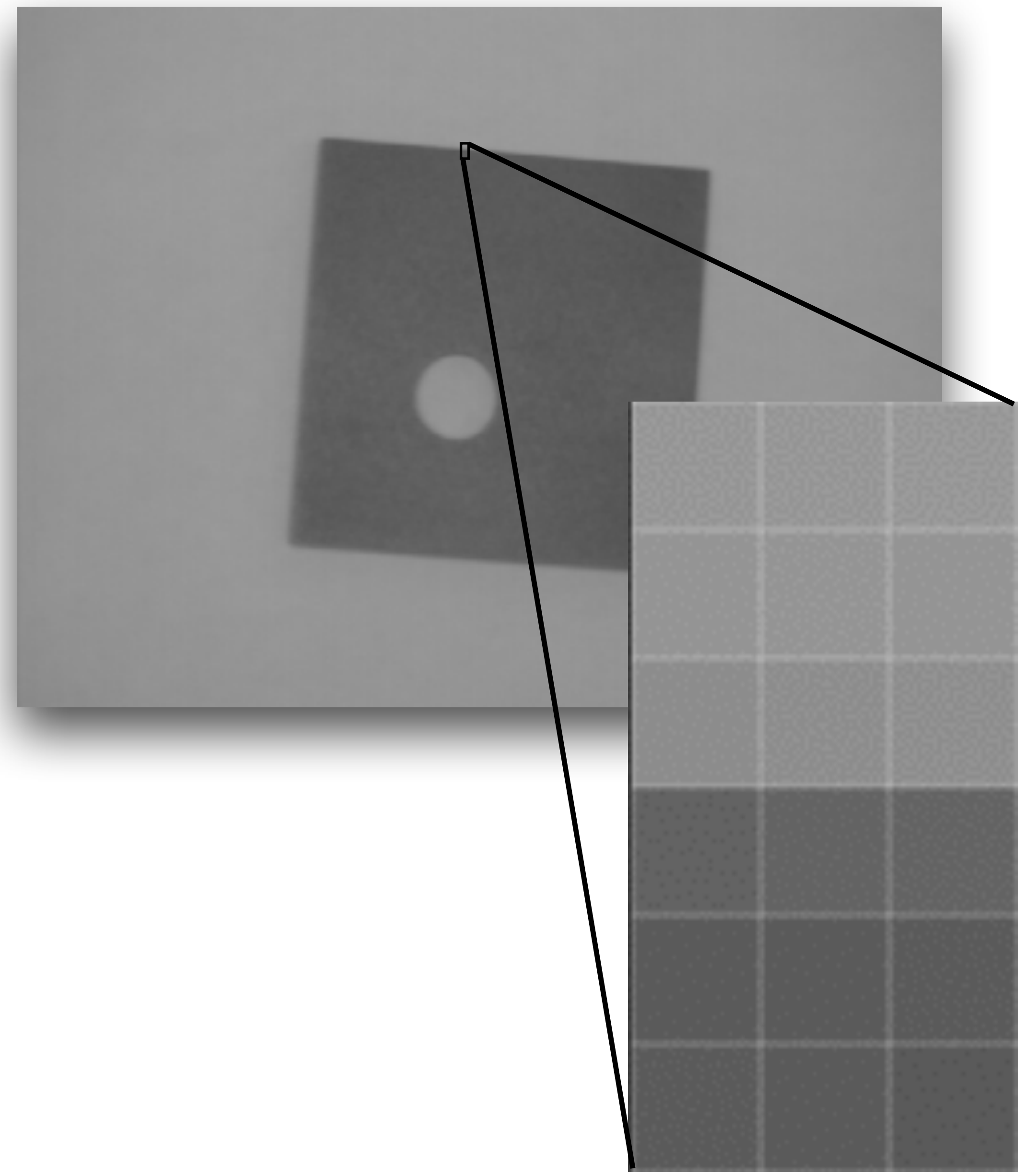
Prewitt edge detector



How it evolved
Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

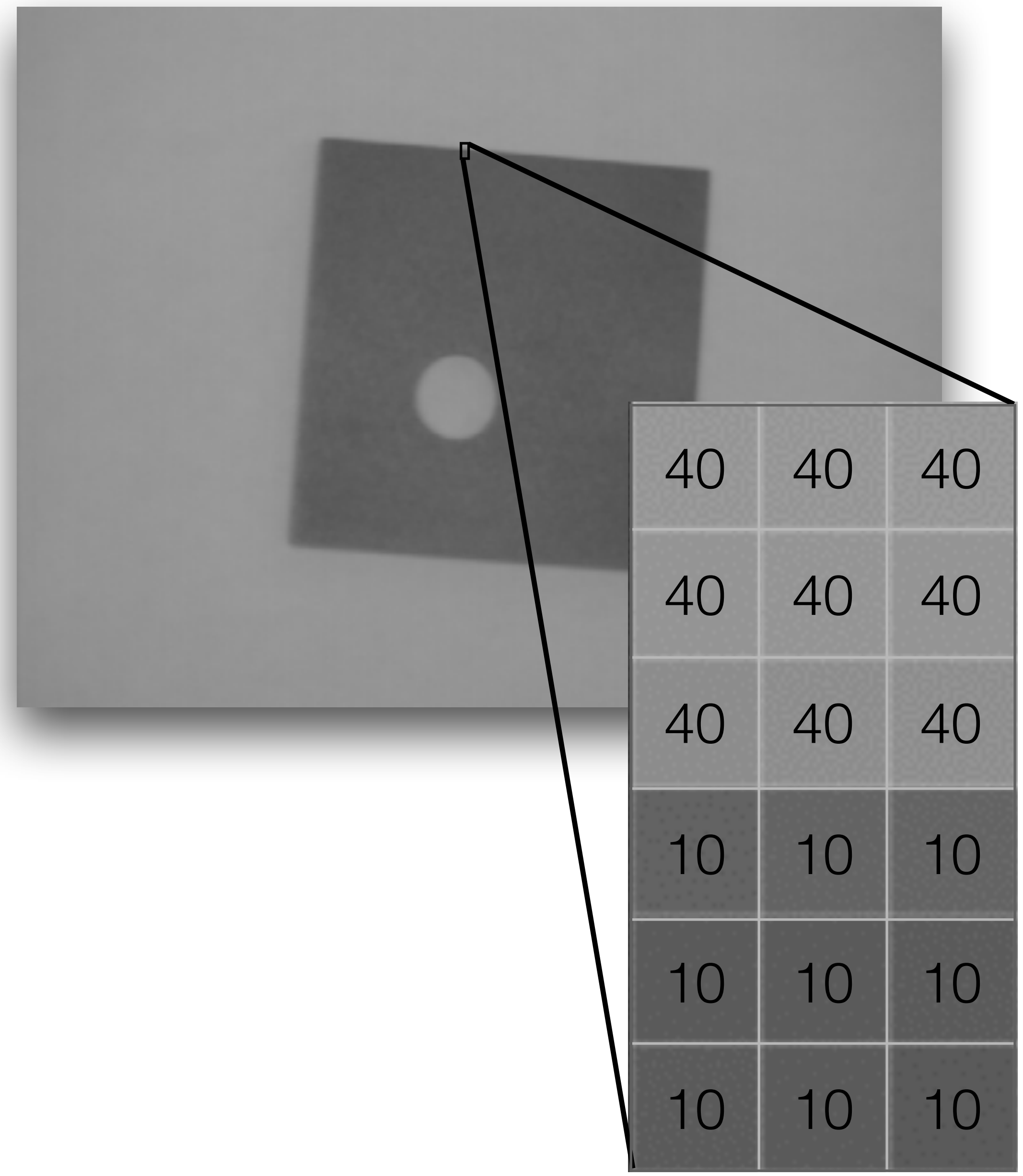


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

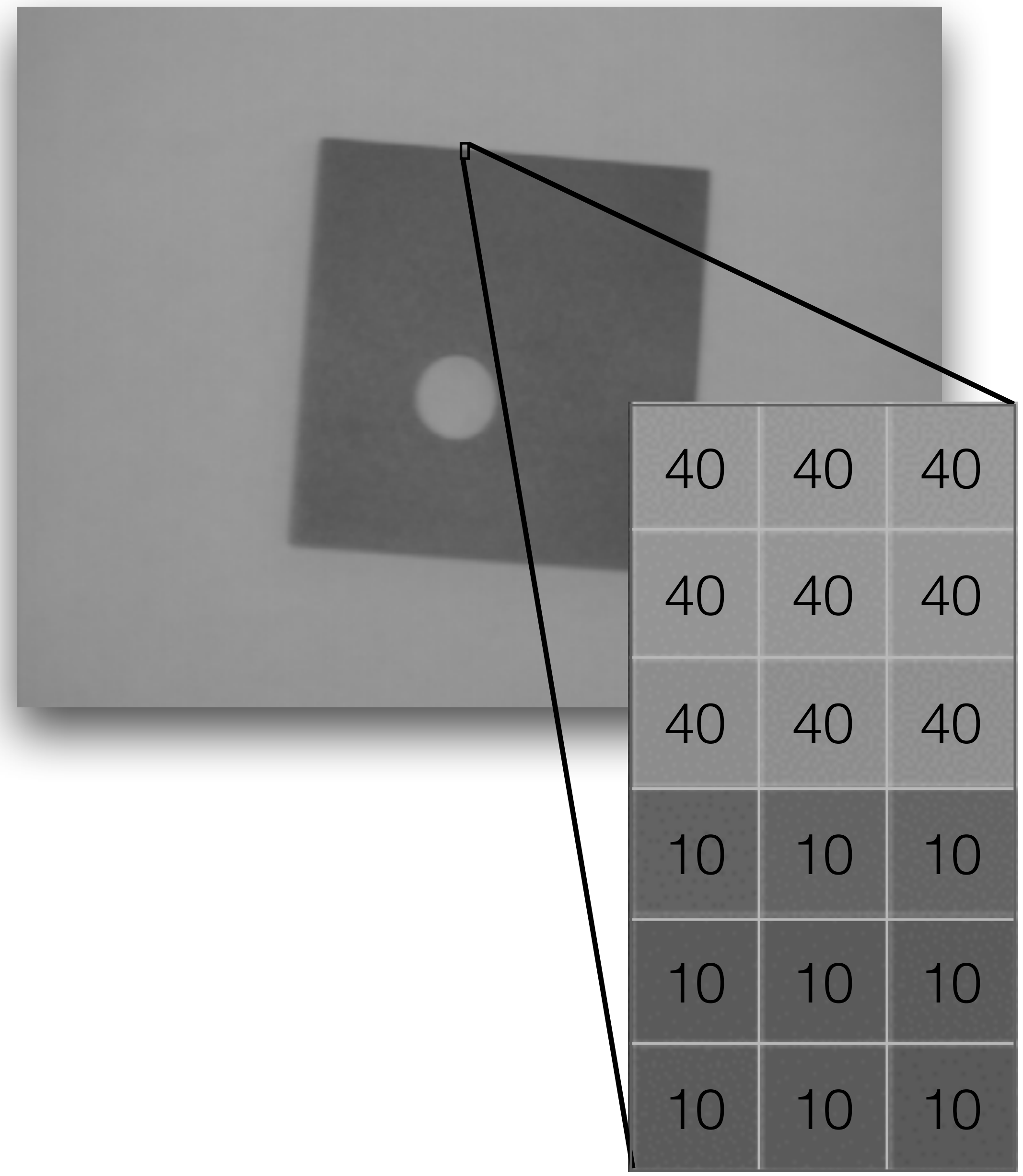


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt
edge
detector



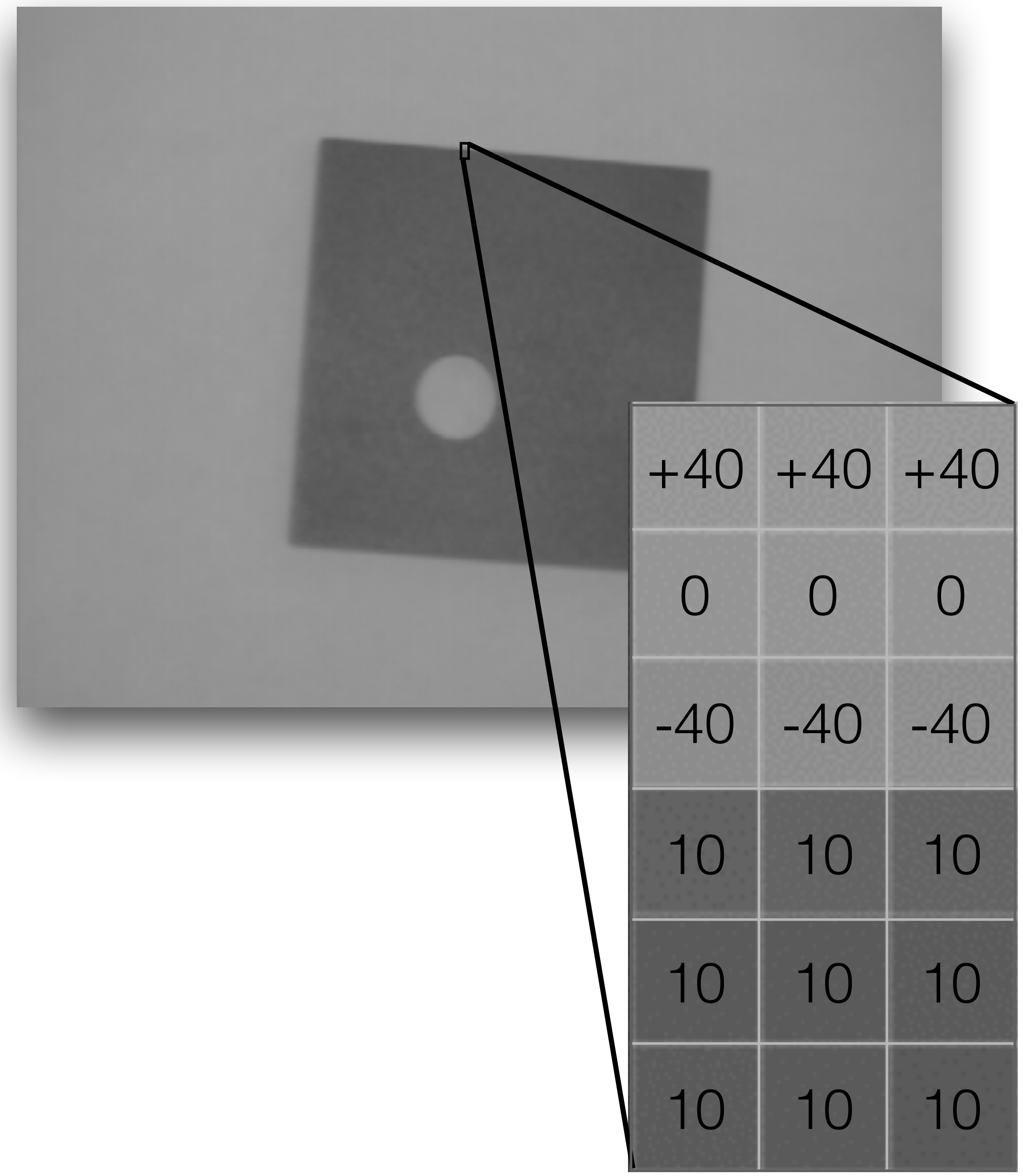
+1	+1	+1
0	0	0
-1	-1	-1

How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

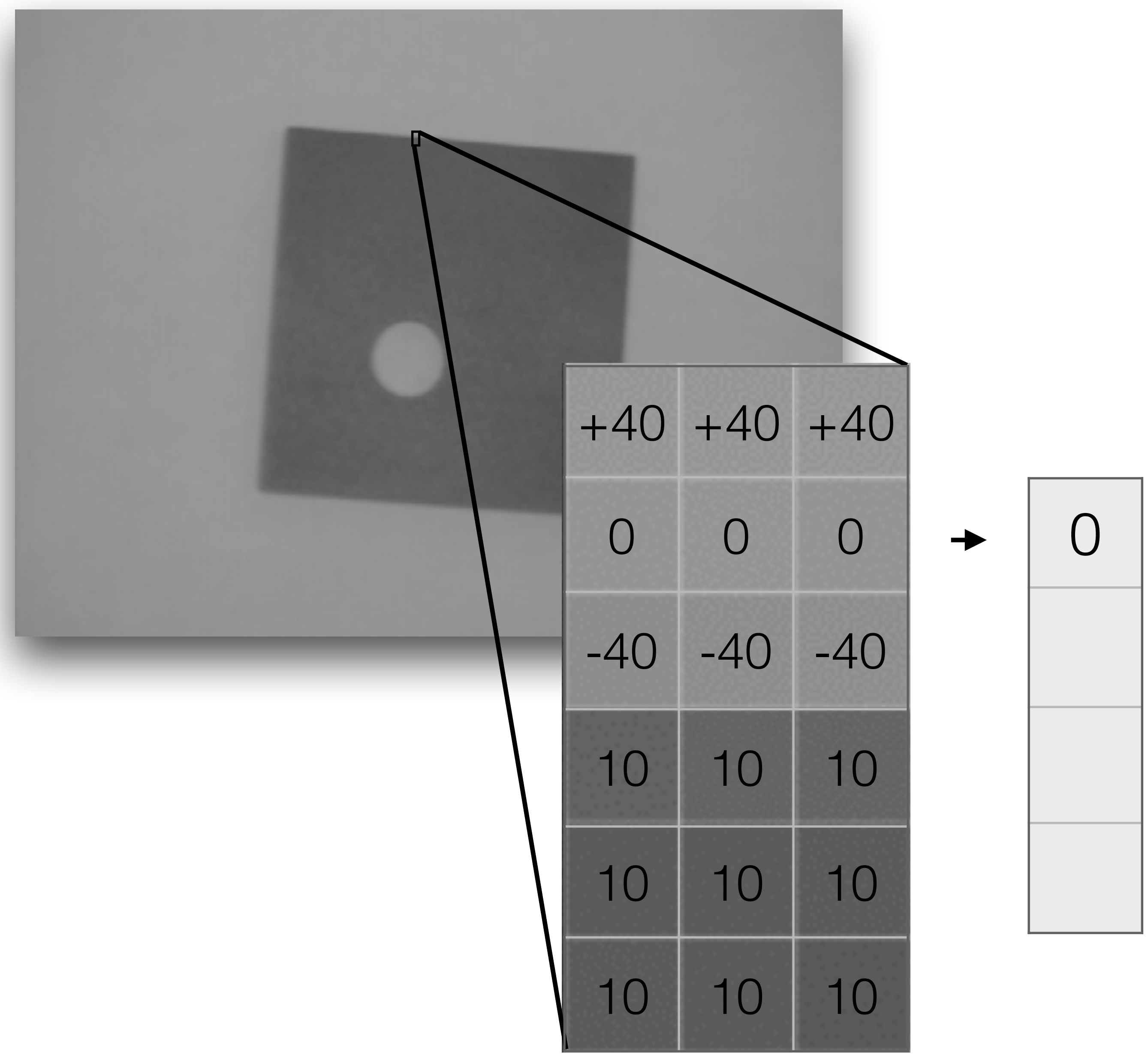


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

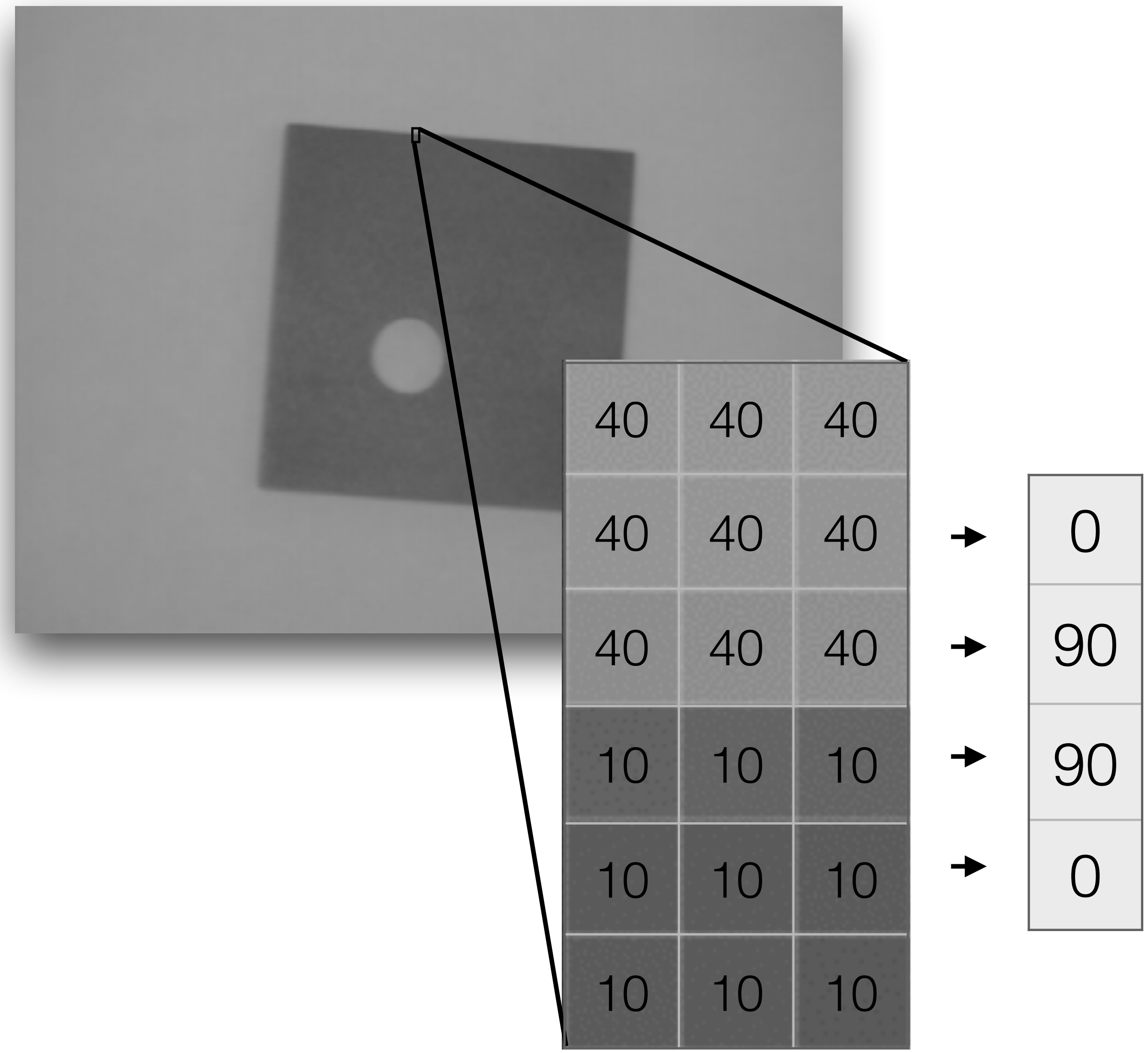


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

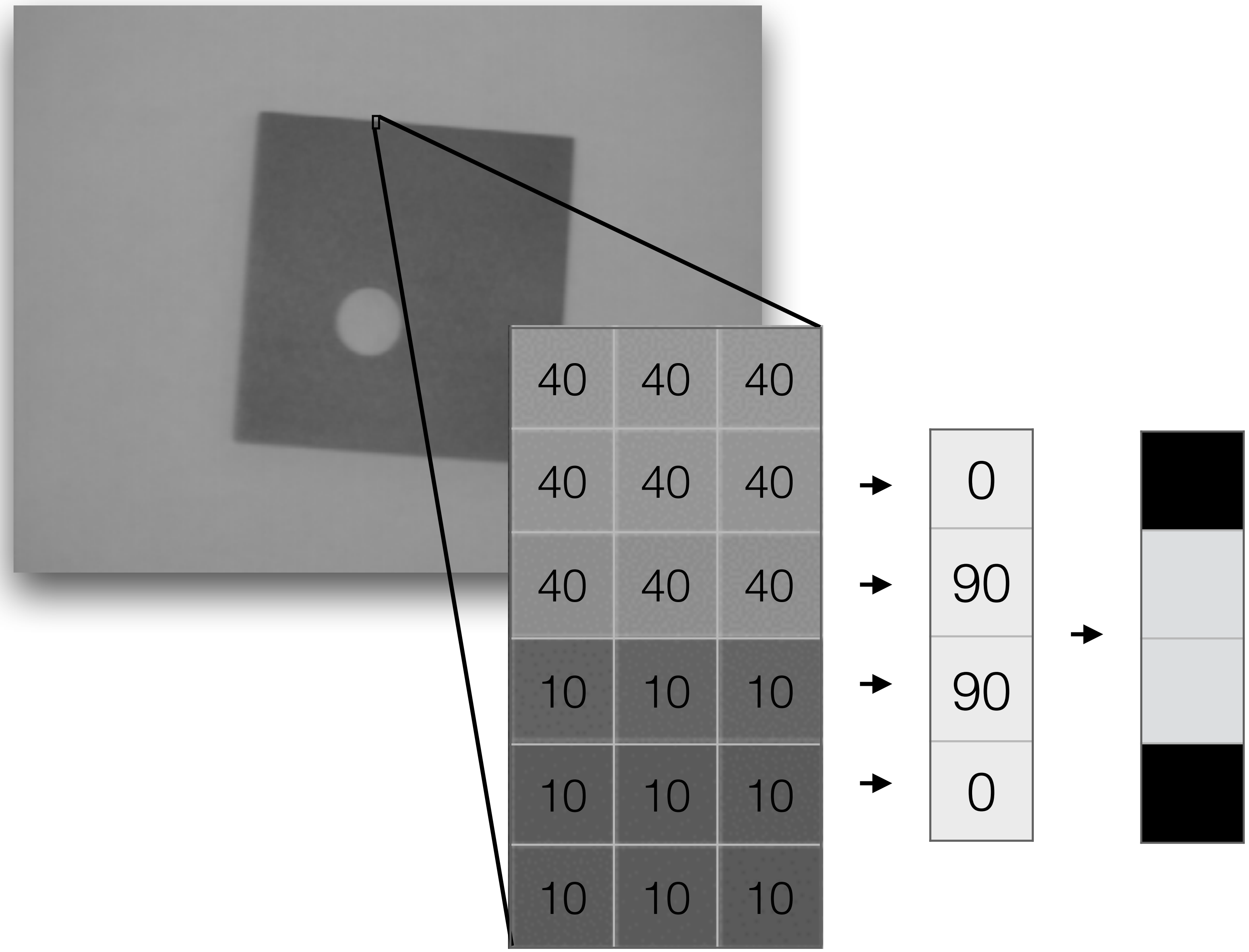


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

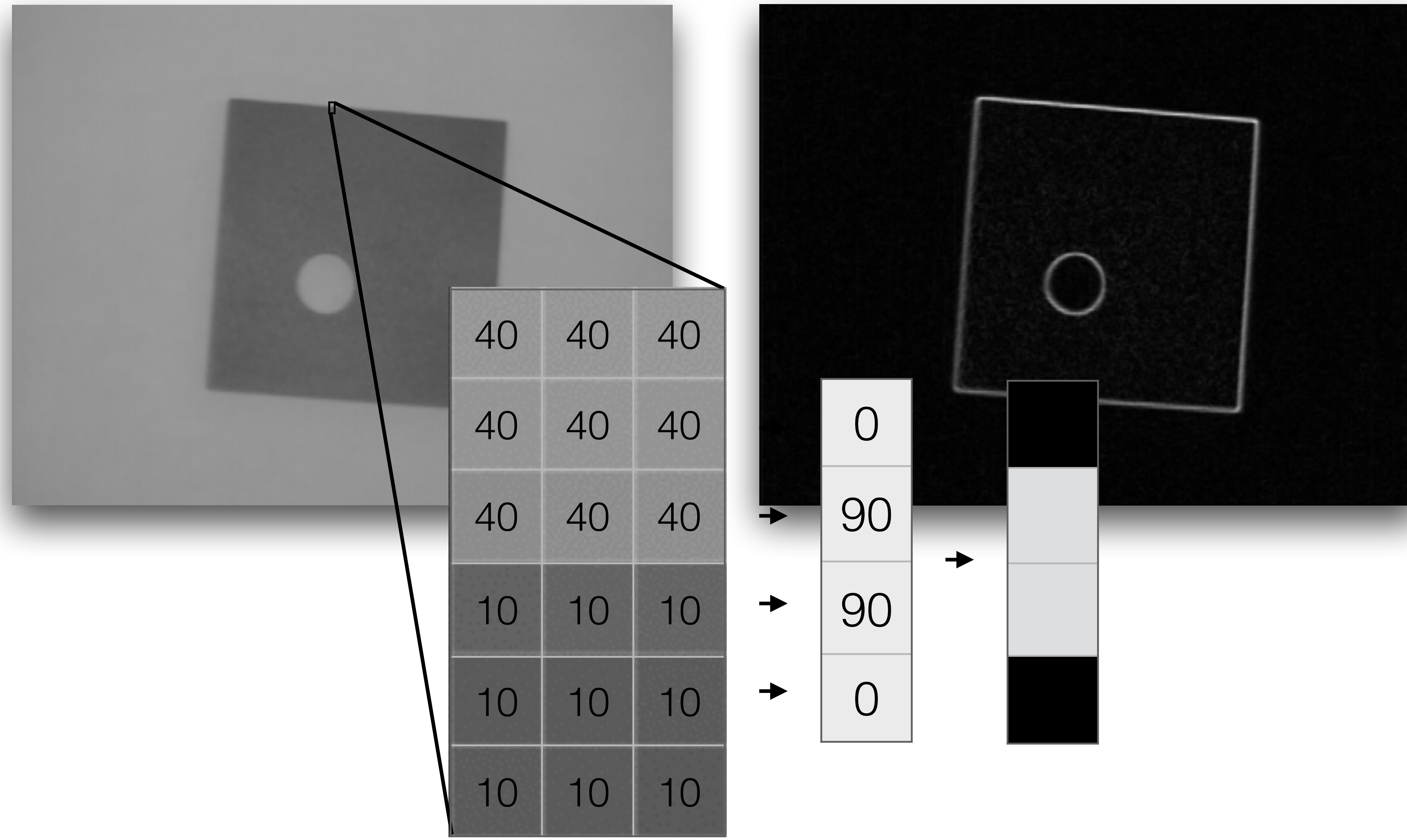


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector

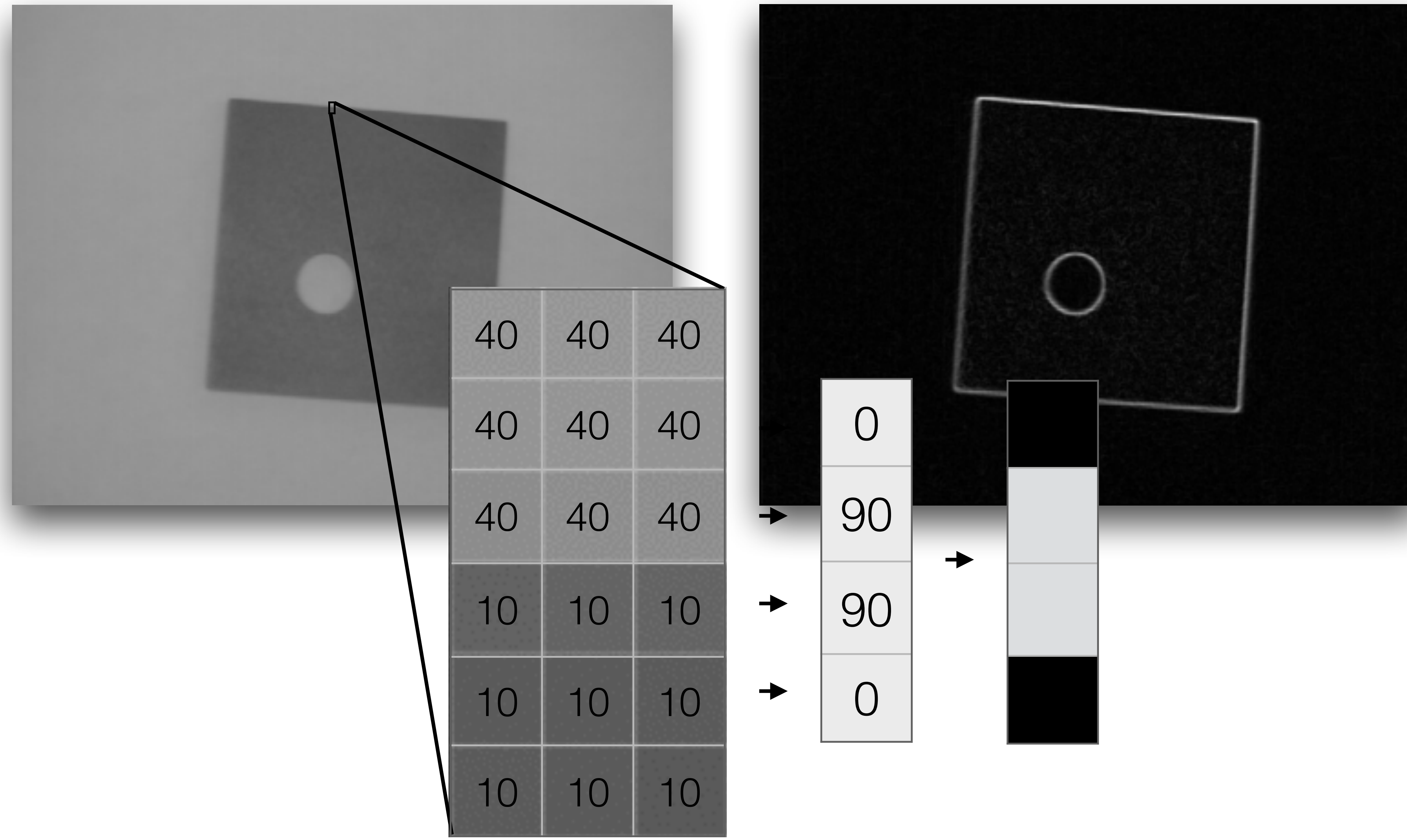


How it evolved

Convolution

+1	+1	+1
0	0	0
-1	-1	-1

Prewitt edge detector



Edge detector is a handcrafted feature detector.

How it evolved

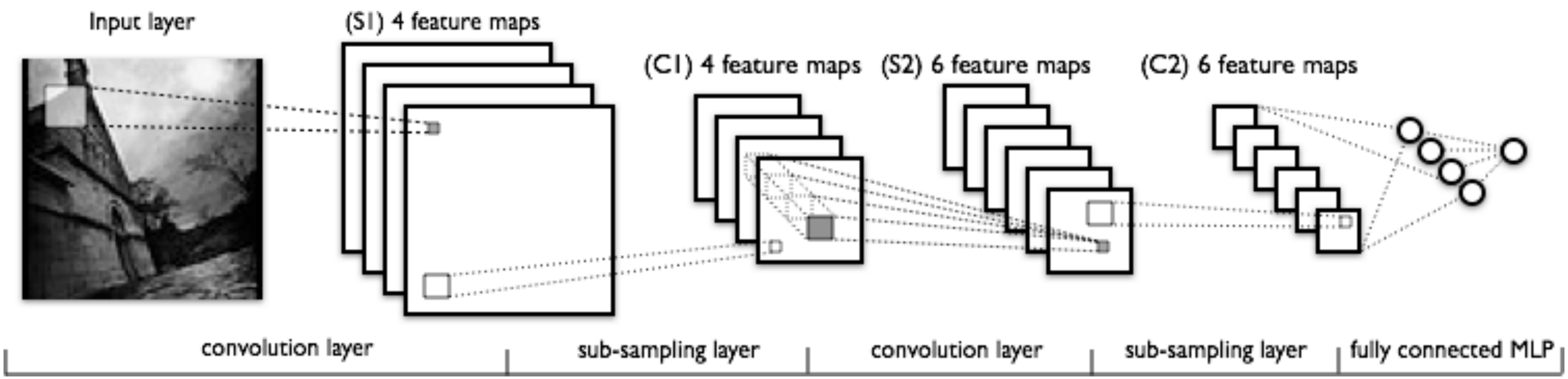
Convolution

The **idea** of a convolutional layer is to **learn** feature detectors **instead** of using handcrafted ones

How it evolved

Convolution

The **idea** of a convolutional layer is to **learn** feature detectors **instead** of using handcrafted ones

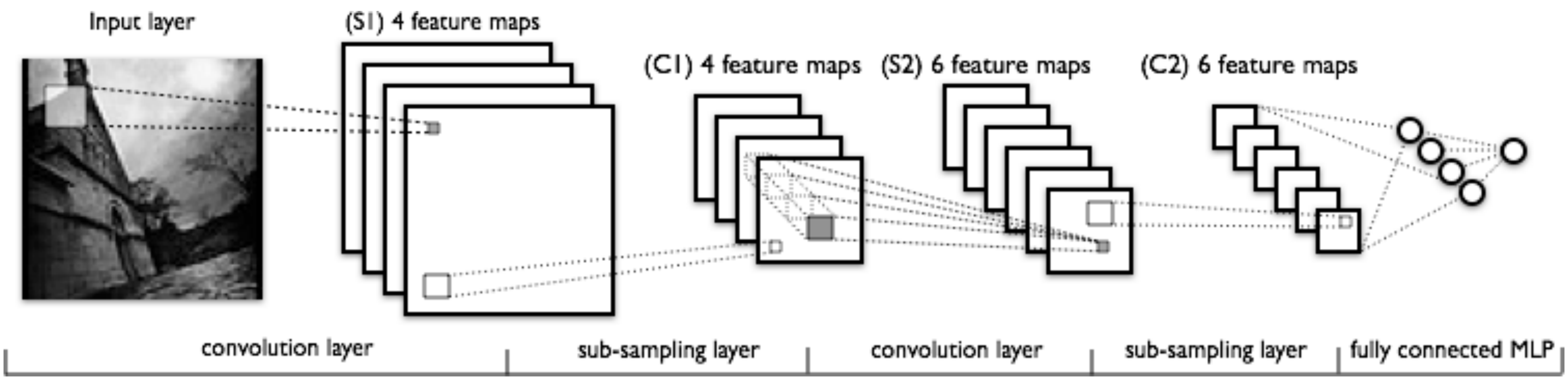


http://yann.lecun.com/exdb/mnist/

How it evolved

Convolution

The **idea** of a convolutional layer is to **learn** feature detectors **instead** of using handcrafted ones



99.50% on the MNIST test set

CURRENT BEST: **99.77%** by committee of 35 conv. nets

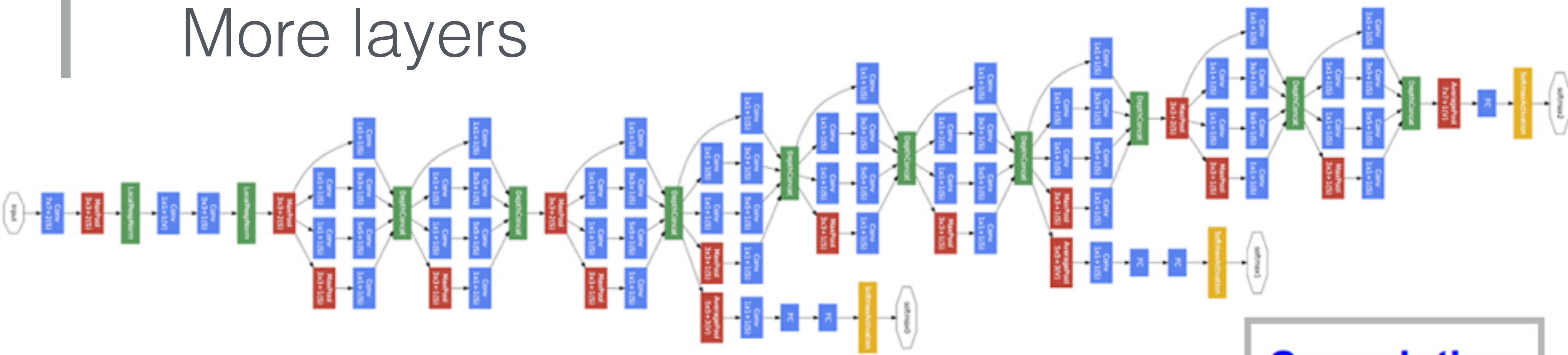
http://yann.lecun.com/exdb/mnist/

How it evolved

More layers

How it evolved

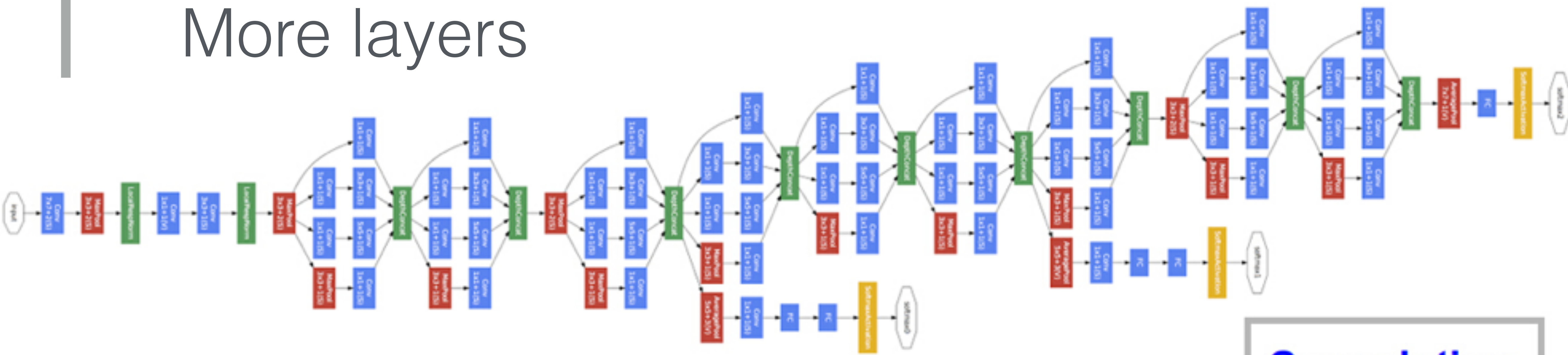
More layers



C. Szegedy, et al., “Going Deeper with Convolutions”, 2014

How it evolved

More layers



Convolution
Pooling
Softmax
Other

C. Szegedy, et al., “Going Deeper with Convolutions”, 2014

ILSVRC 2015 winner — 152 (!) layers

Task 2a: Classification+localization with provided training data

Ordered by localization error

Team name	Entry description	**Localization error**	Classification error
MSRA	Ensemble A for classification and localization.	0.090178	0.03567
MSRA	Ensemble B for classification and localization.	0.090801	0.03567
MSRA	Ensemble C for classification and localization.	0.092108	0.0369
Trimps-Soushen	combined 12 models	0.122907	0.04649
	Ensemble of 9 NeoNets with bounding box regression. Weighted		

K. He et al., “Deep Residual Learning for Image Recognition”, 2015

How it evolved

Hyperparameters

- Network:
  - architecture
  - number of layers
  - number of units (in each layer)
  - type of the activation function
  - weight initialization
- Convolutional layers:
  - size
  - stride
  - number of filters
- Optimization method:
  - learning rate
  - other method-specific constants
- …

How it evolved
Hyperparameters

Grid search :(

- Network:
  - architecture
  - number of layers
  - number of units (in each layer)
  - type of the activation function
  - weight initialization
- Convolutional layers:
  - size
  - stride
  - number of filters
- Optimization method:
  - learning rate
  - other method-specific constants
- …

How it evolved

Hyperparameters

Grid search :(

Random search :/

- Network:
  - architecture
  - number of layers
  - number of units (in each layer)
  - type of the activation function
  - weight initialization
- Convolutional layers:
  - size
  - stride
  - number of filters
- Optimization method:
  - learning rate
  - other method-specific constants
- …

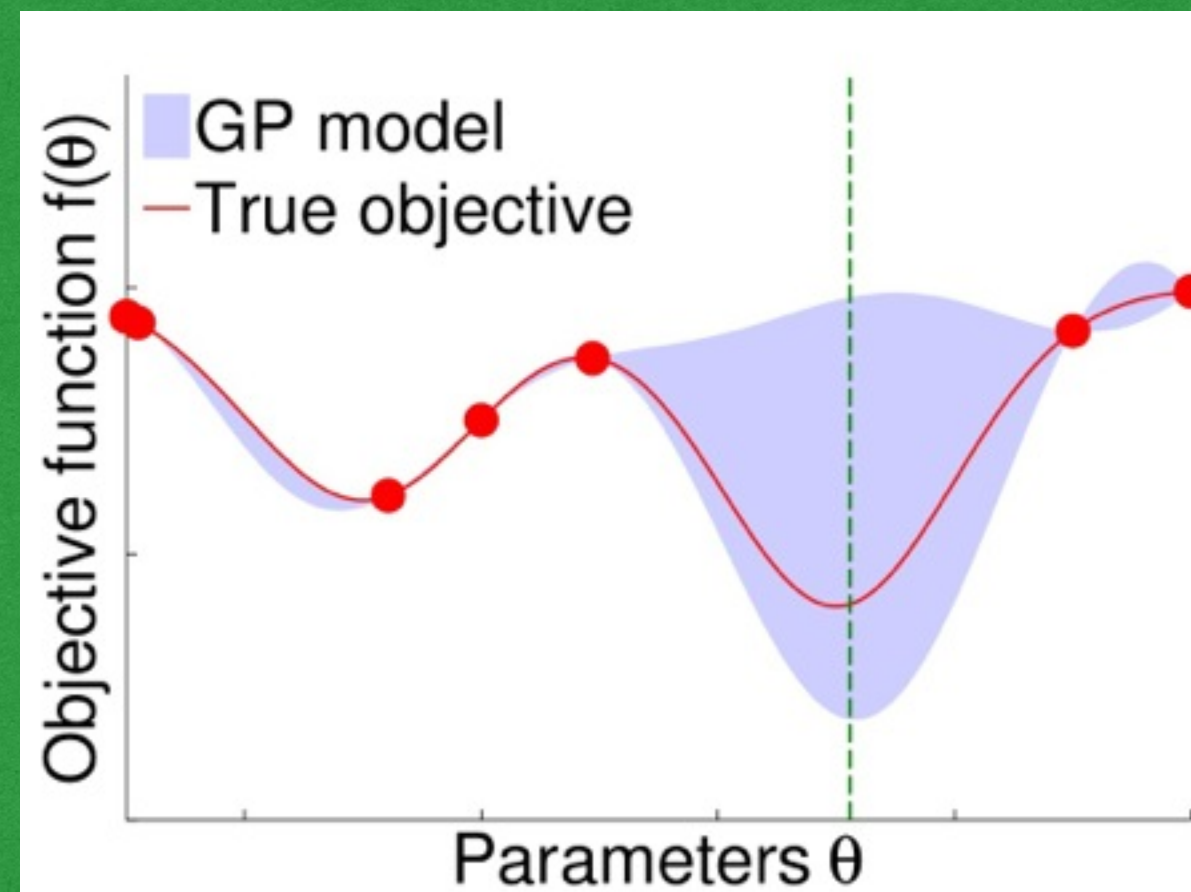
How it evolved
Hyperparameters

- Network:
  - architecture
  - number of layers
  - number of units (in each layer)
  - type of the activation function
  - weight initialization
- Convolutional layers:
  - size
  - stride
  - number of filters
- Optimization method:
  - learning rate
  - other method-specific constants
- …

Grid search :(

Random search :/

Bayesian optimization :)



Snoek, Larochelle, Adams, "Practical Bayesian Optimization of Machine Learning Algorithms"

How it evolved

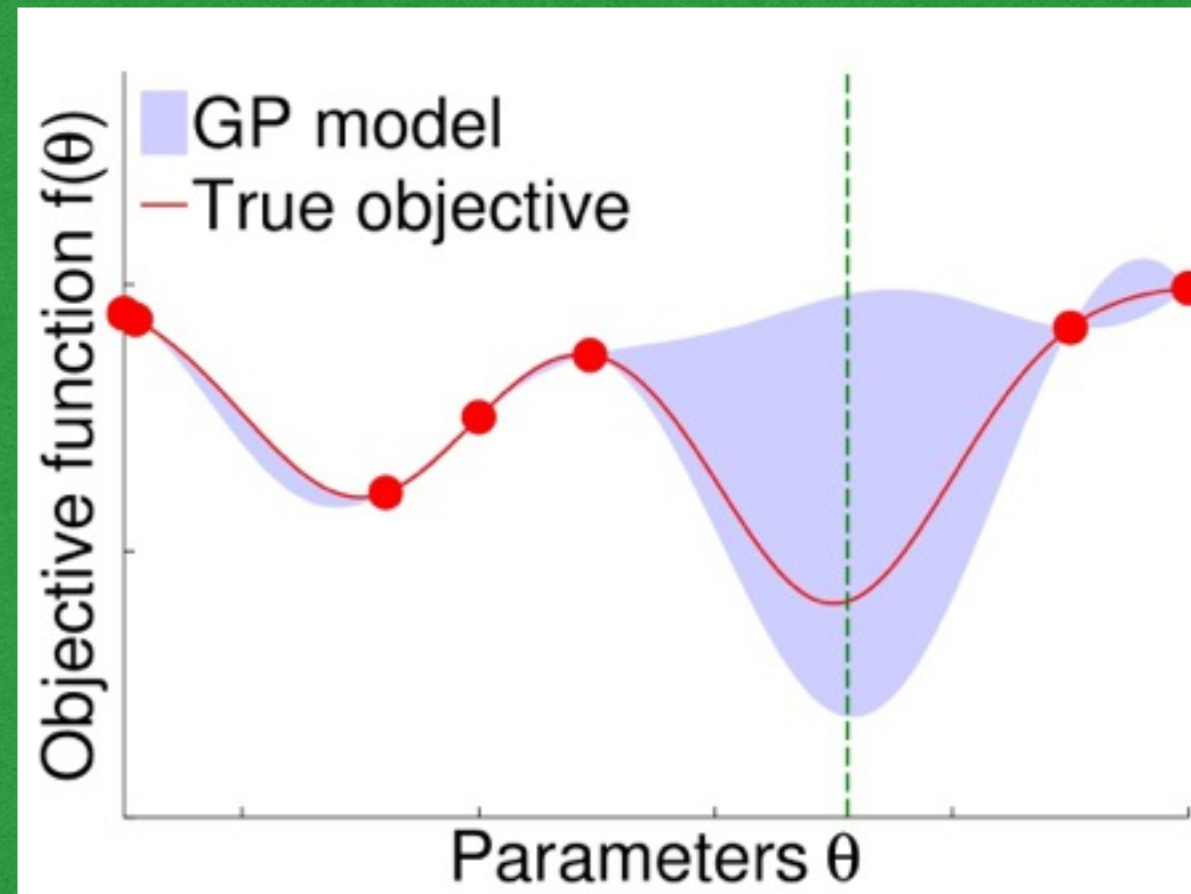
Hyperparameters

- Network:
  - architecture
  - number of layers
  - number of units (in each layer)
  - type of the activation function
  - weight initialization
- Convolutional layers:
  - size
  - stride
  - number of filters
- Optimization method:
  - learning rate
  - other method-specific constants
- …

Grid search :(

Random search :/

Bayesian optimization :)

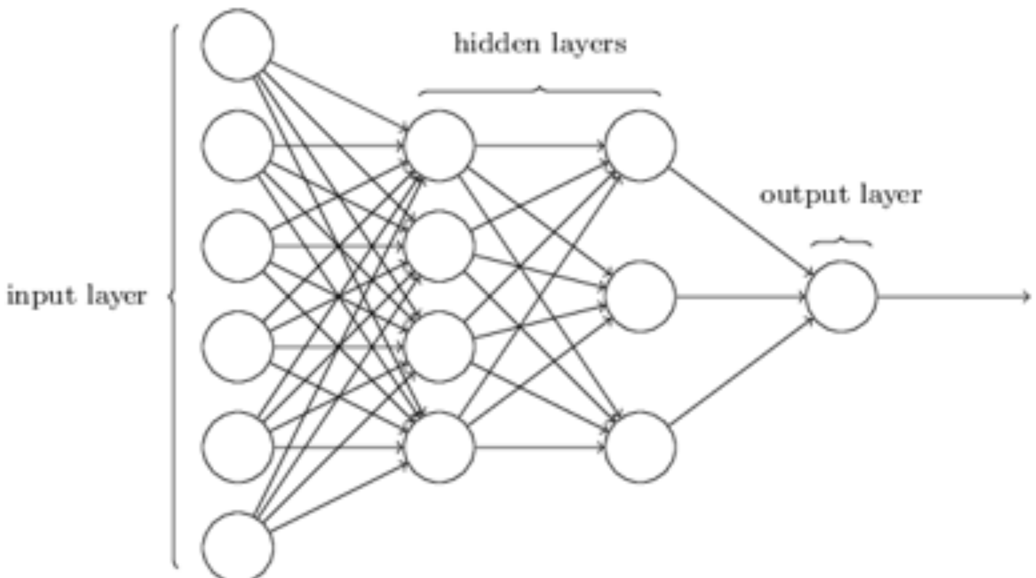


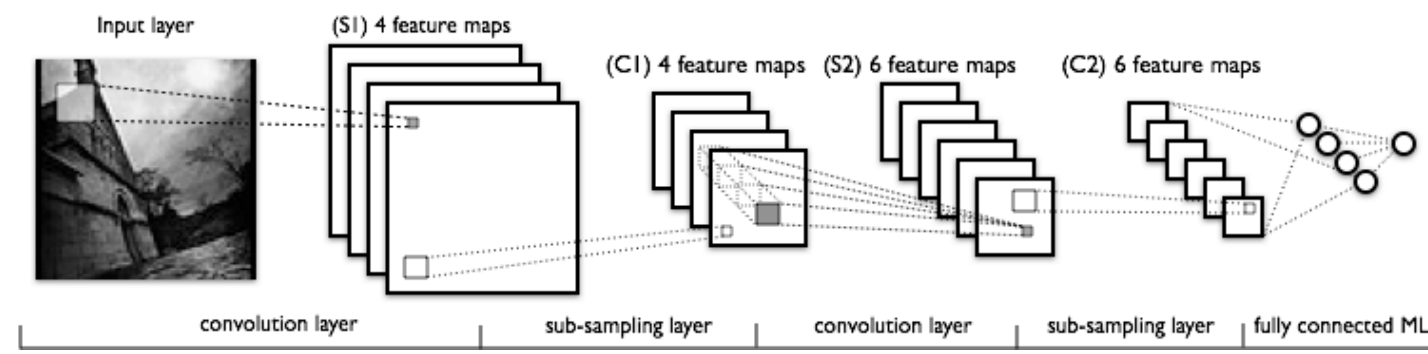
Informal parameter search :)

Snoek, Larochelle, Adams, "Practical Bayesian Optimization of Machine Learning Algorithms"

How it evolved

Major Types of ANNs



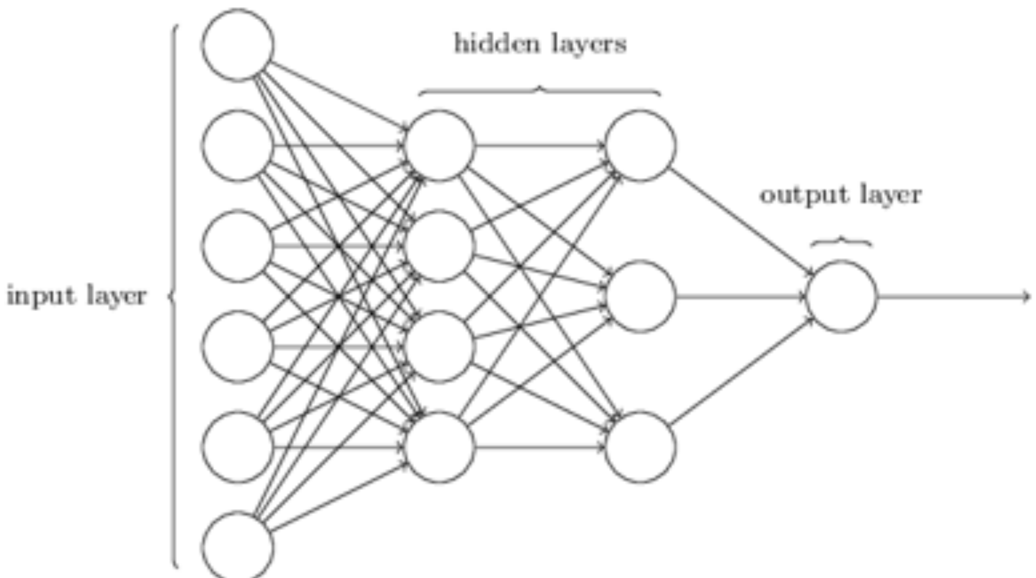


feedforward

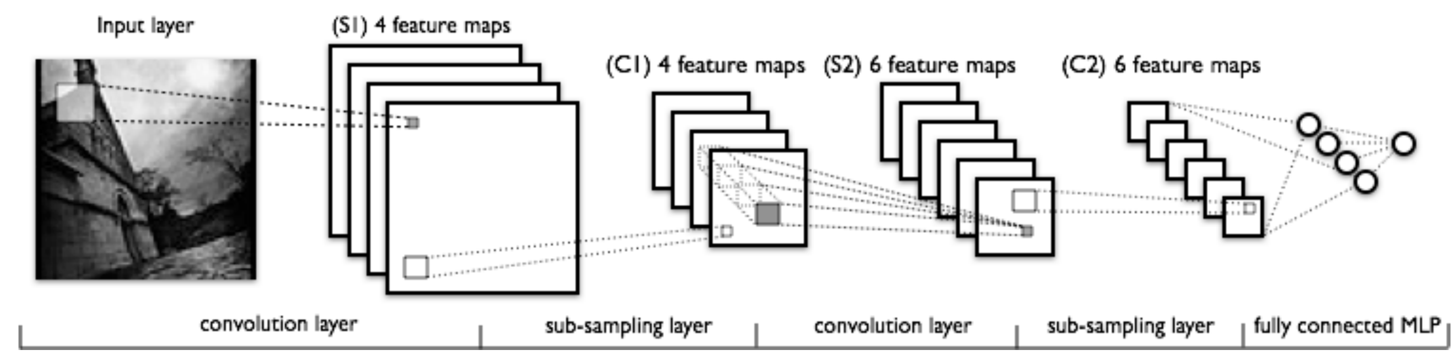
convolutional

How it evolved

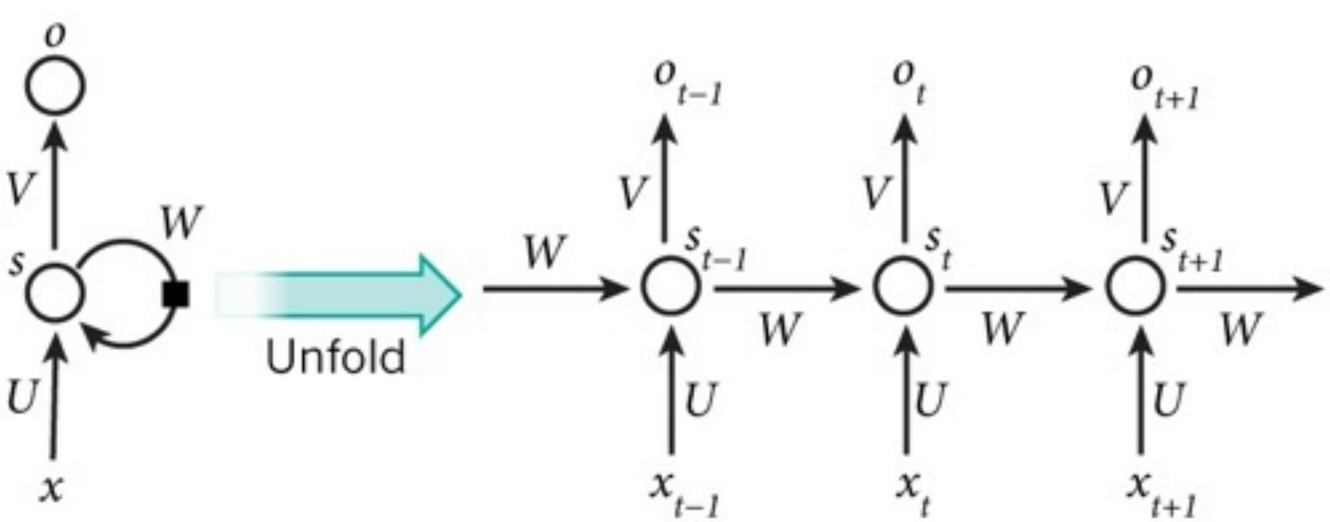
Major Types of ANNs



feedforward



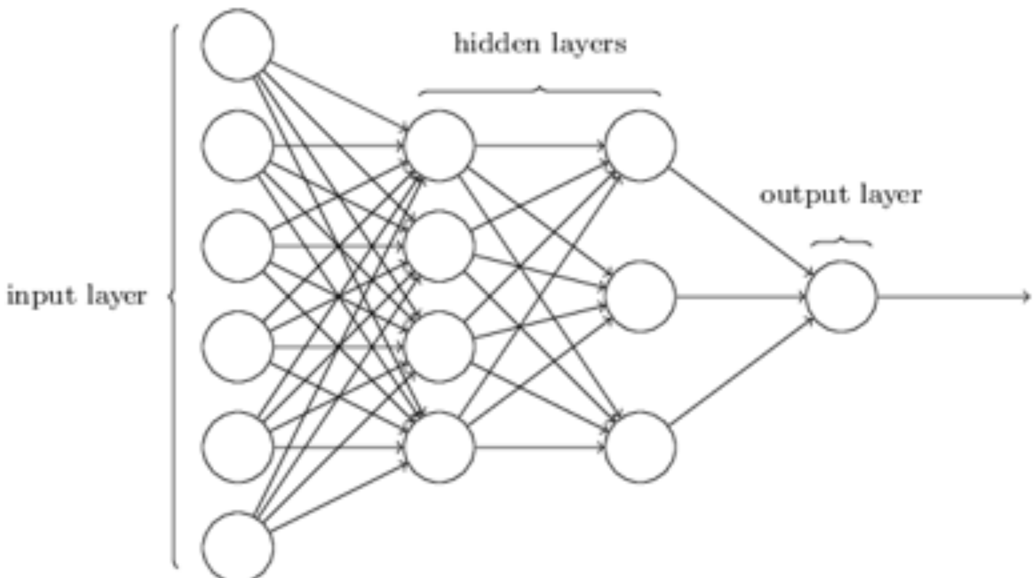
convolutional



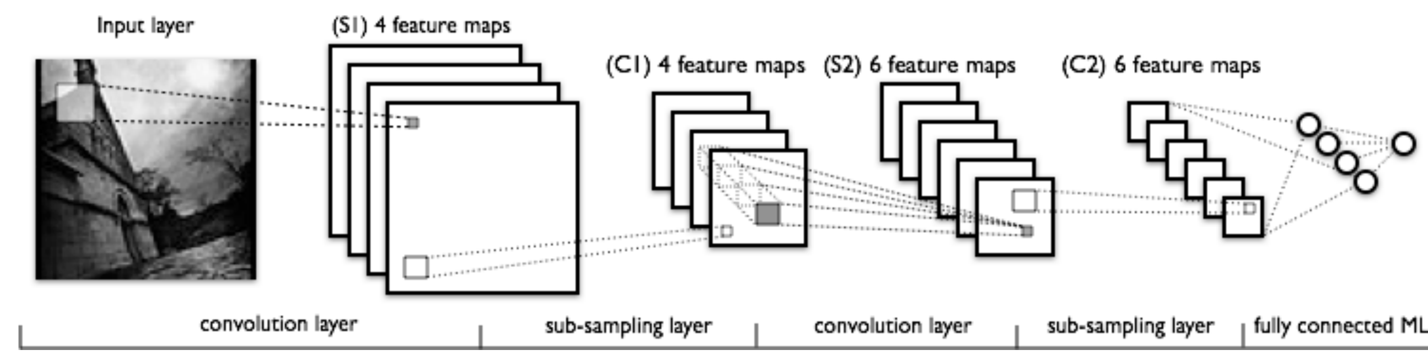
recurrent

How it evolved

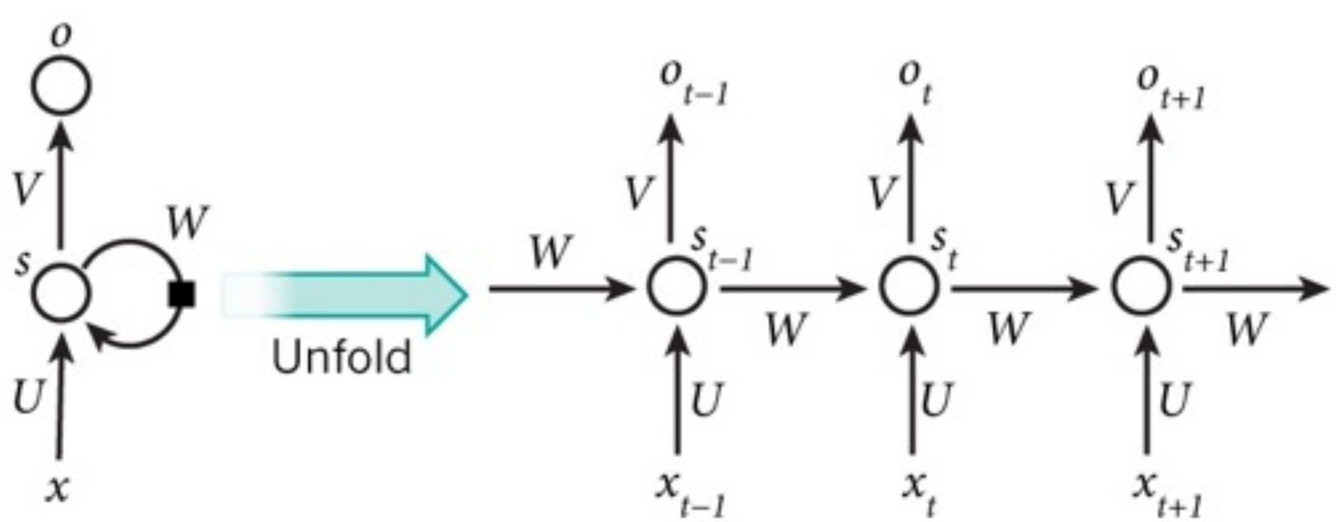
Major Types of ANNs



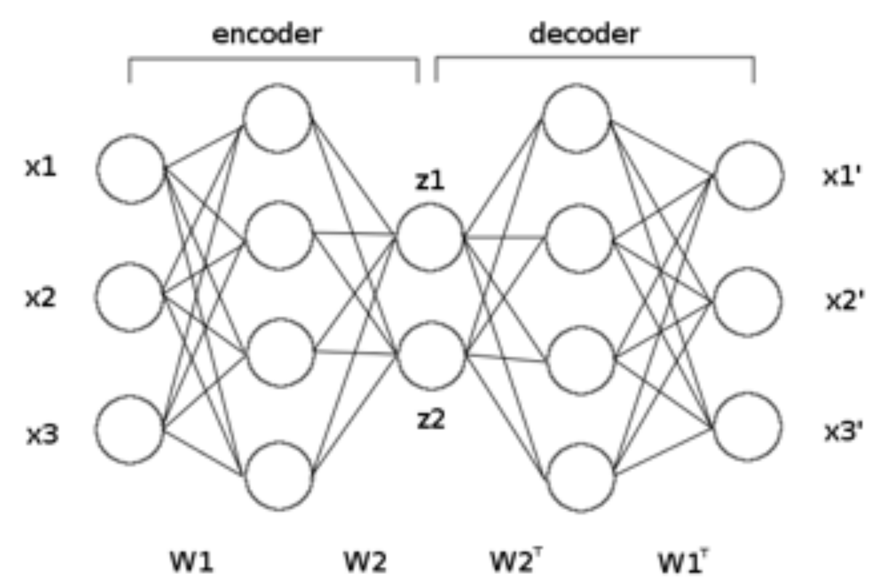
feedforward



convolutional



recurrent

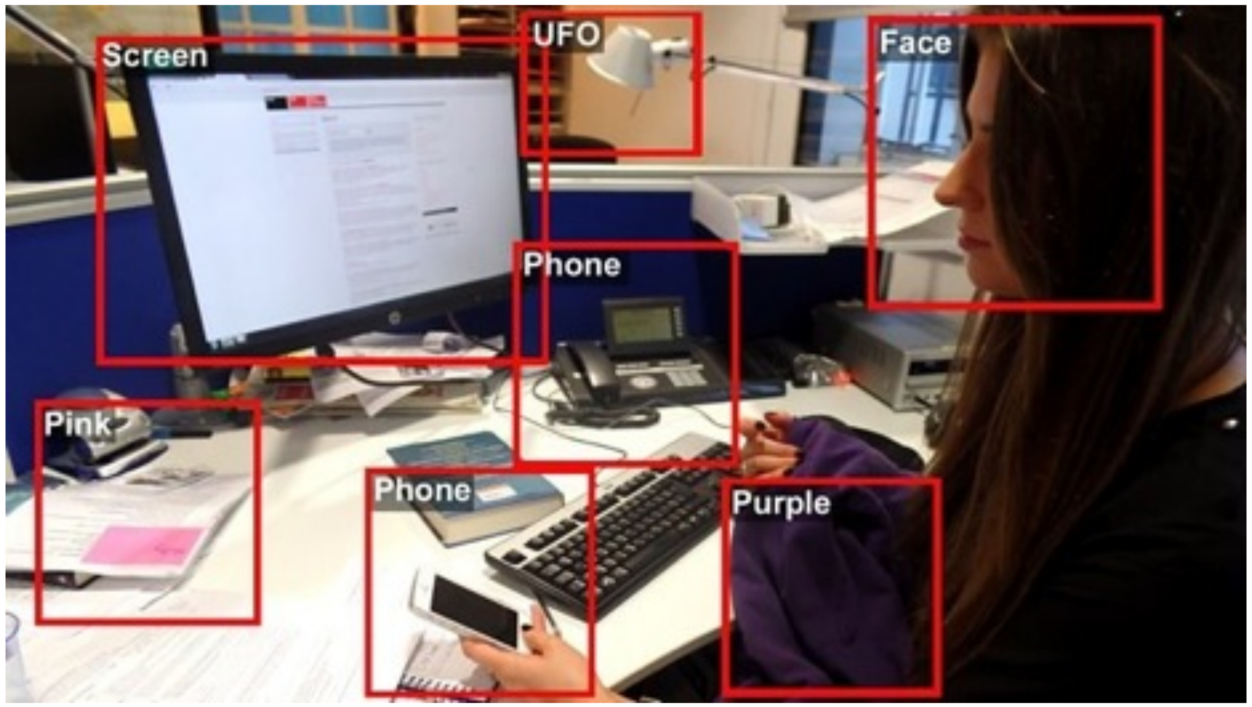


autoencoder

What is the state now

What is the state now

Computer vision



Kaiming He, et al.
“Deep Residual Learning for Image Recognition”
2015

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"girl in pink dress is jumping in air."

"black and white dog jumps over bar."

http://cs.stanford.edu/people/karpathy/deepimagesent/

What is the state now

Natural Language Processing

speech recognition + translation

http://smerity.com/articles/2015/keras_qa.html

1 John moved to the bedroom.
2 **Mary grabbed the football there.**
3 Sandra journeyed to the bedroom.
4 Sandra went back to the hallway.
5 Mary moved to the garden.
6 **Mary journeyed to the office.**
7 Where is the **football**? office 2 6

Facebook bAbi dataset: question answering

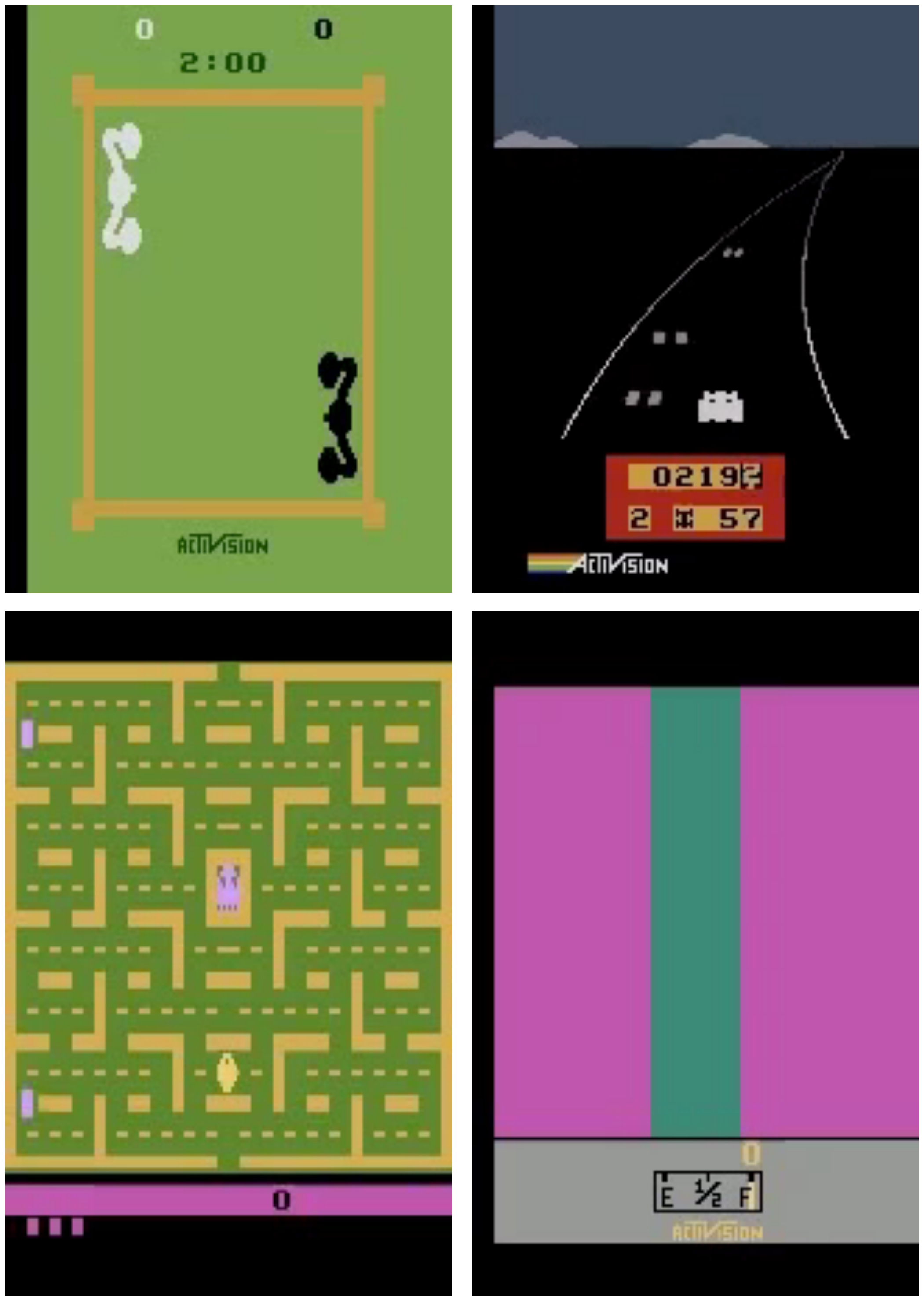
```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

```
<page>
  <title>Antichrist</title>
  <id>865</id>
  <revision>
    <id>15900676</id>
    <timestamp>2002-08-03T18:14:12Z</timestamp>
    <contributor>
      <username>Paris</username>
      <id>23</id>
    </contributor>
    <minor />
    <comment>Automated conversion</comment>
    <text xml:space="preserve">#REDIRECT [[Christianity]]</text>
  </revision>
</page>
```

What is the state now

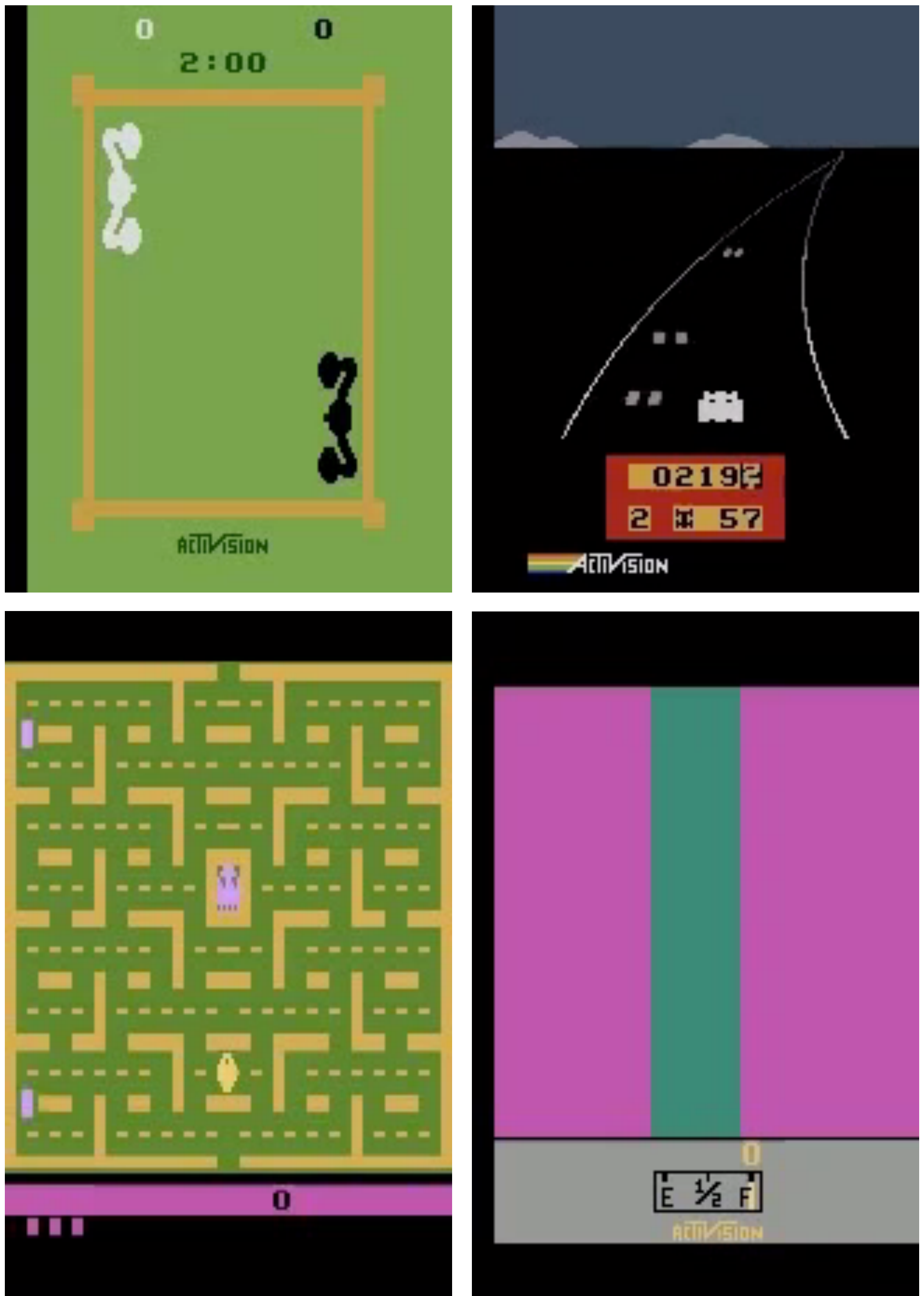
AI



DeepMind's DQN

What is the state now

AI

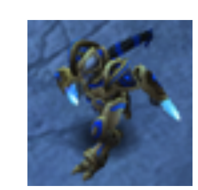


DeepMind's DQN

Figure 3: Right: StarCraft 2-vs-2 combat.
Our models learn to concentrate fire on the weaker of the two enemy bots.

	2 vs 2	Kiting	Kiting hard
Attack weakest	85%	0%	0%
2 layer NN	80% (38k)	89% (190k)	30% (275k)
MemNN	80% (83k)	92% (120k)	41% (360k)

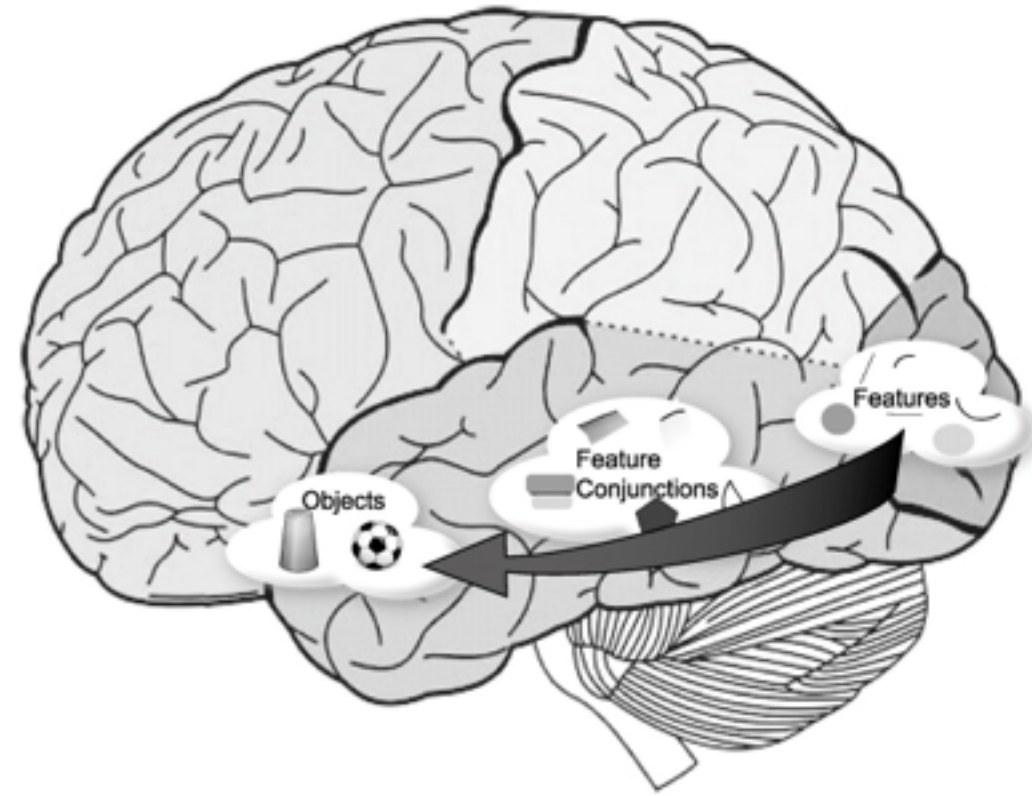
Table 2: Win rates against StarCraft built-in AI.



Sukhbaatar et al. "MazeBase: A Sandbox for Learning from Games", 2015

What is the state now

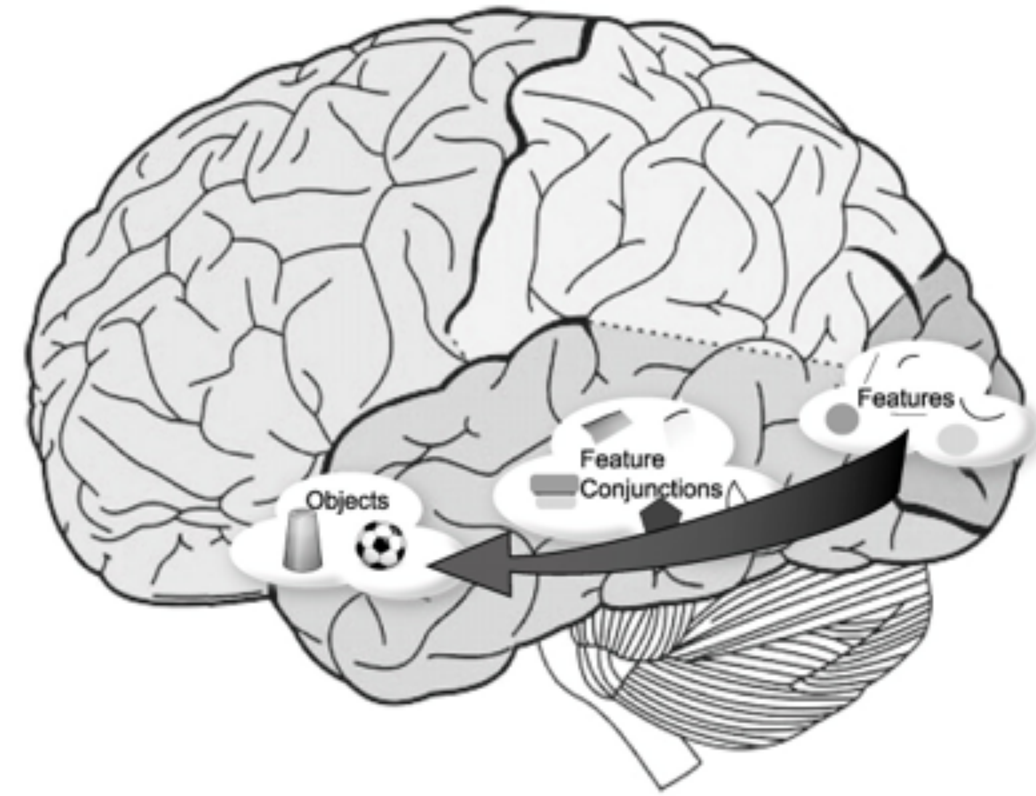
Neuroscience

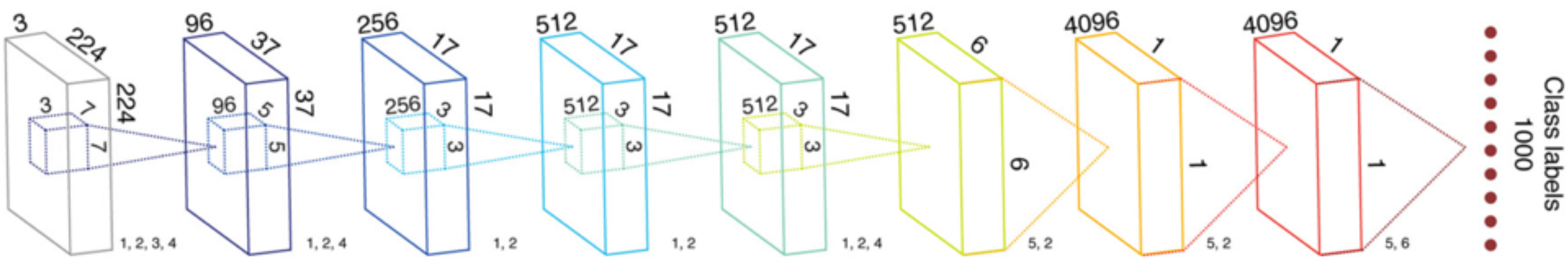


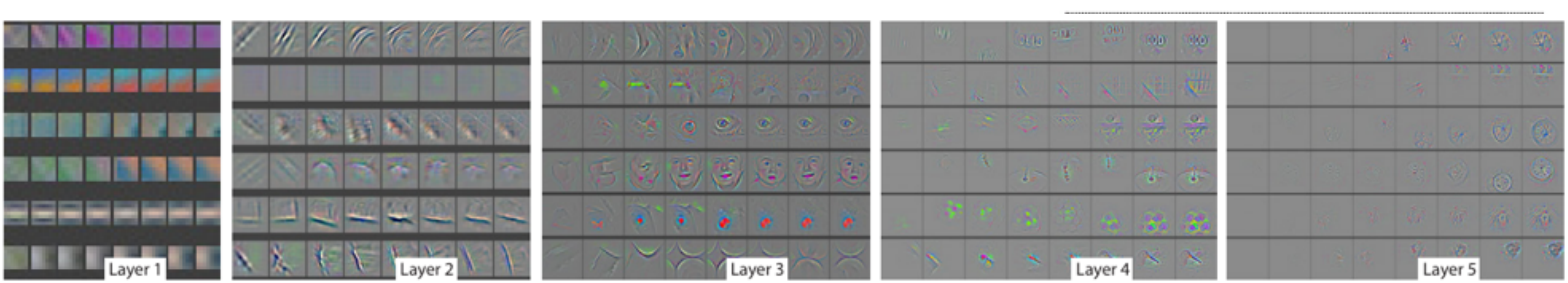
Güclü and Gerven, “Deep Neural Networks Reveal a Gradient in the Complexity of Neural Representations across the Ventral Stream”, 2015

What is the state now

Neuroscience







Güclü and Gerven, "Deep Neural Networks Reveal a Gradient in the Complexity of Neural Representations across the Ventral Stream", 2015

How can **you** use it

How can **you** use it

Pre-trained models

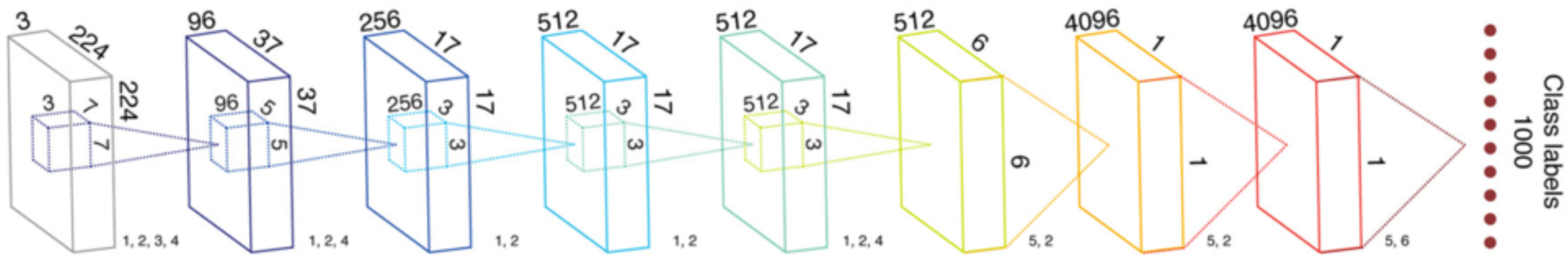
Caffe Model Zoo

How can **you** use it

Pre-trained models

Caffe Model Zoo

- Go to https://github.com/BVLC/caffe/wiki/Model-Zoo, **pick** a model



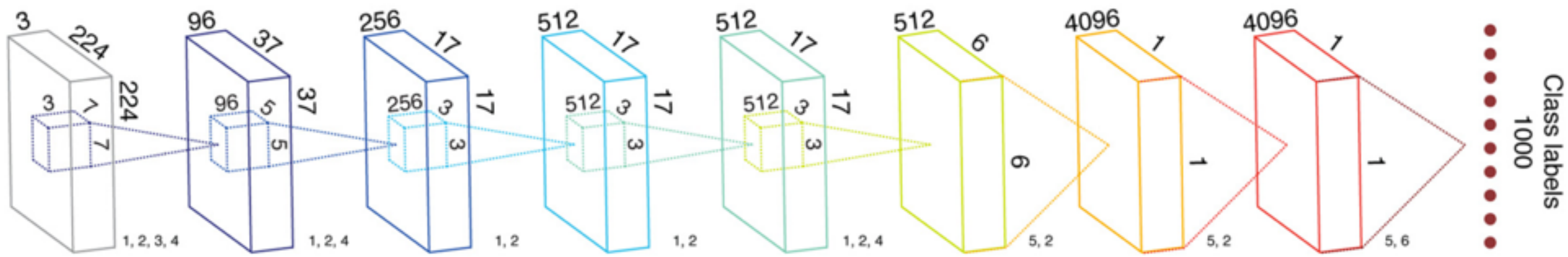
- … and **use it** in your application

How can **you** use it

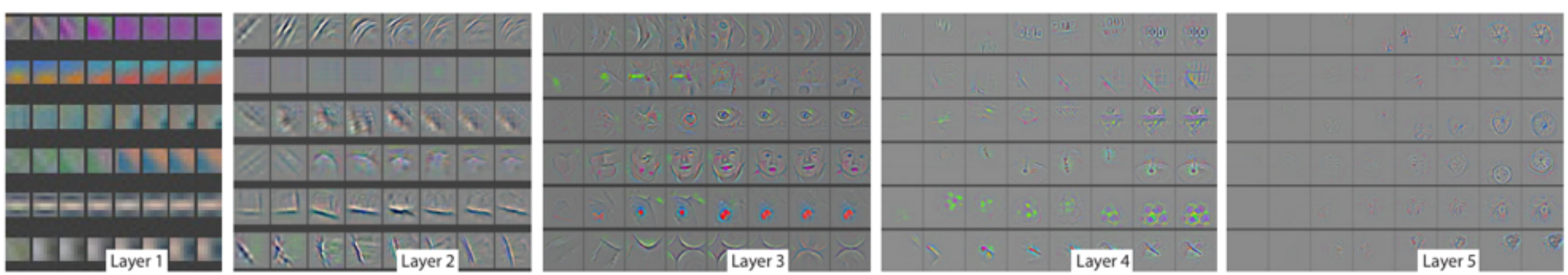
Pre-trained models

Caffe Model Zoo

- Go to https://github.com/BVLC/caffe/wiki/Model-Zoo, **pick** a model



- … and **use it** in your application
- Or …

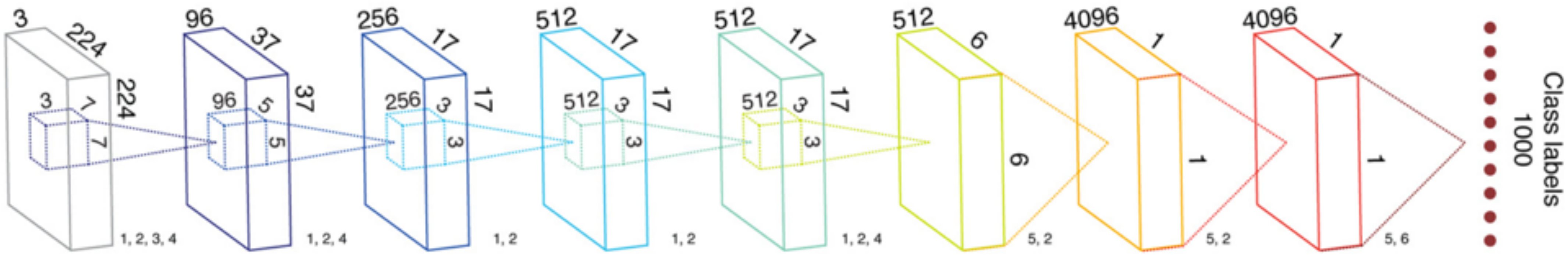


How can **you** use it

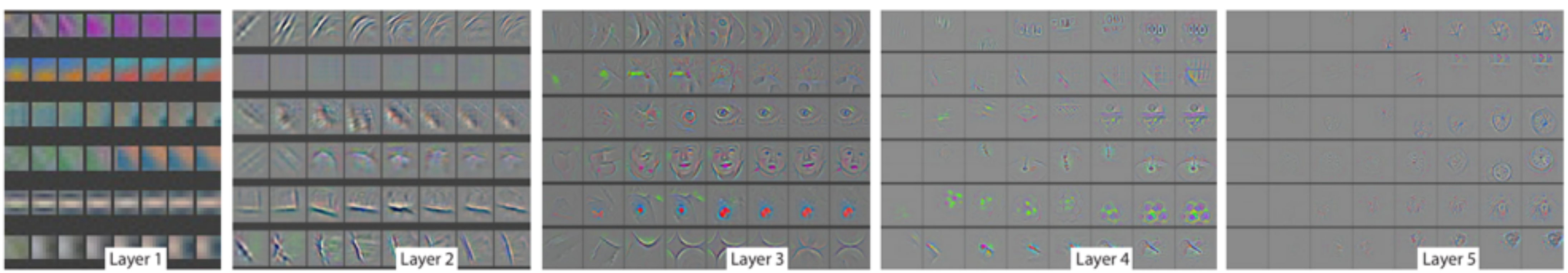
Pre-trained models

Caffe Model Zoo

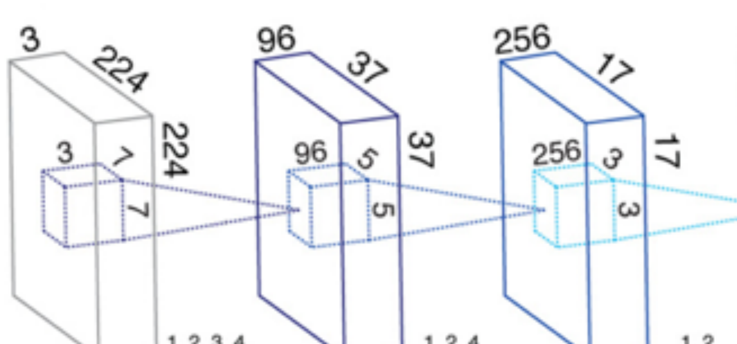
- Go to https://github.com/BVLC/caffe/wiki/Model-Zoo, **pick** a model



- … and **use it** in your application
- Or …



- … use part of it as the **starting point** for your model



. . .

How can **you** use it

Zoo of Frameworks

Low-level	High-level & Wrappers
theano	K Keras
torch	neon framework by nervana
	Caffe
	ConvNetJS Deep Learning in your browser
	Deeplearning4j Open-source, distributed deep learning for the JVM on Spark with GPUs San Francisco, Outer Spa... http://deeplearning4j.org help@skymind.io

How can **you** use it

Keras

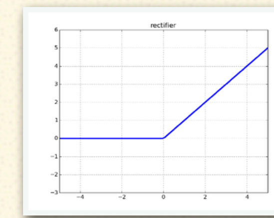
```python
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init='uniform', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init='uniform', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```

http://keras.io

How can **you** use it

Keras

```
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init='uniform', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init='uniform', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```

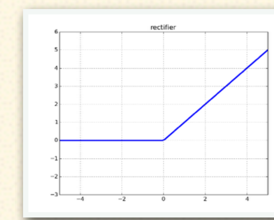


http://keras.io

How can **you** use it

Keras

```
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init='uniform', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init='uniform', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```

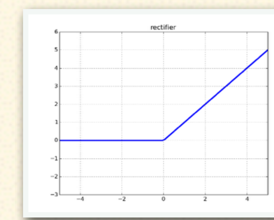


http://keras.io

How can **you** use it

Keras

```
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init='uniform', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init='uniform', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```

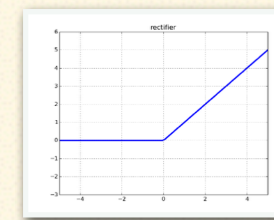


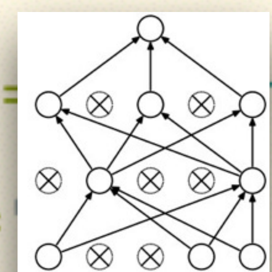
http://keras.io

How can **you** use it

Keras

```python
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init='uniform', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init='uniform', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```





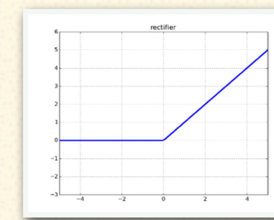
(b) After applying dropout.

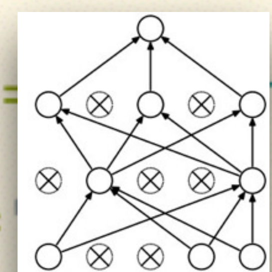
http://keras.io

How can **you** use it

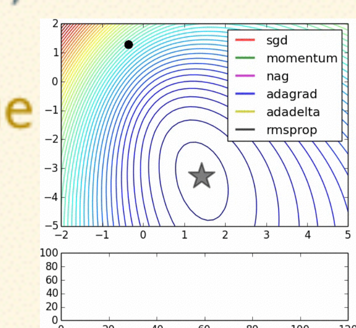
Keras

```
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init=          m', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init='          ', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```





(b) After applying dropout.

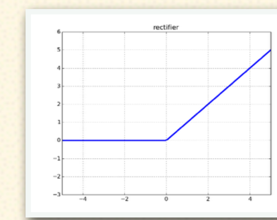


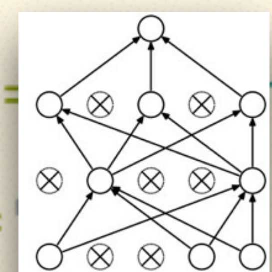
http://keras.io

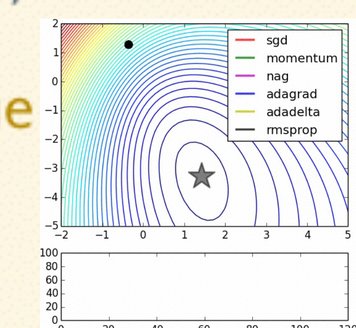
How can **you** use it

Keras

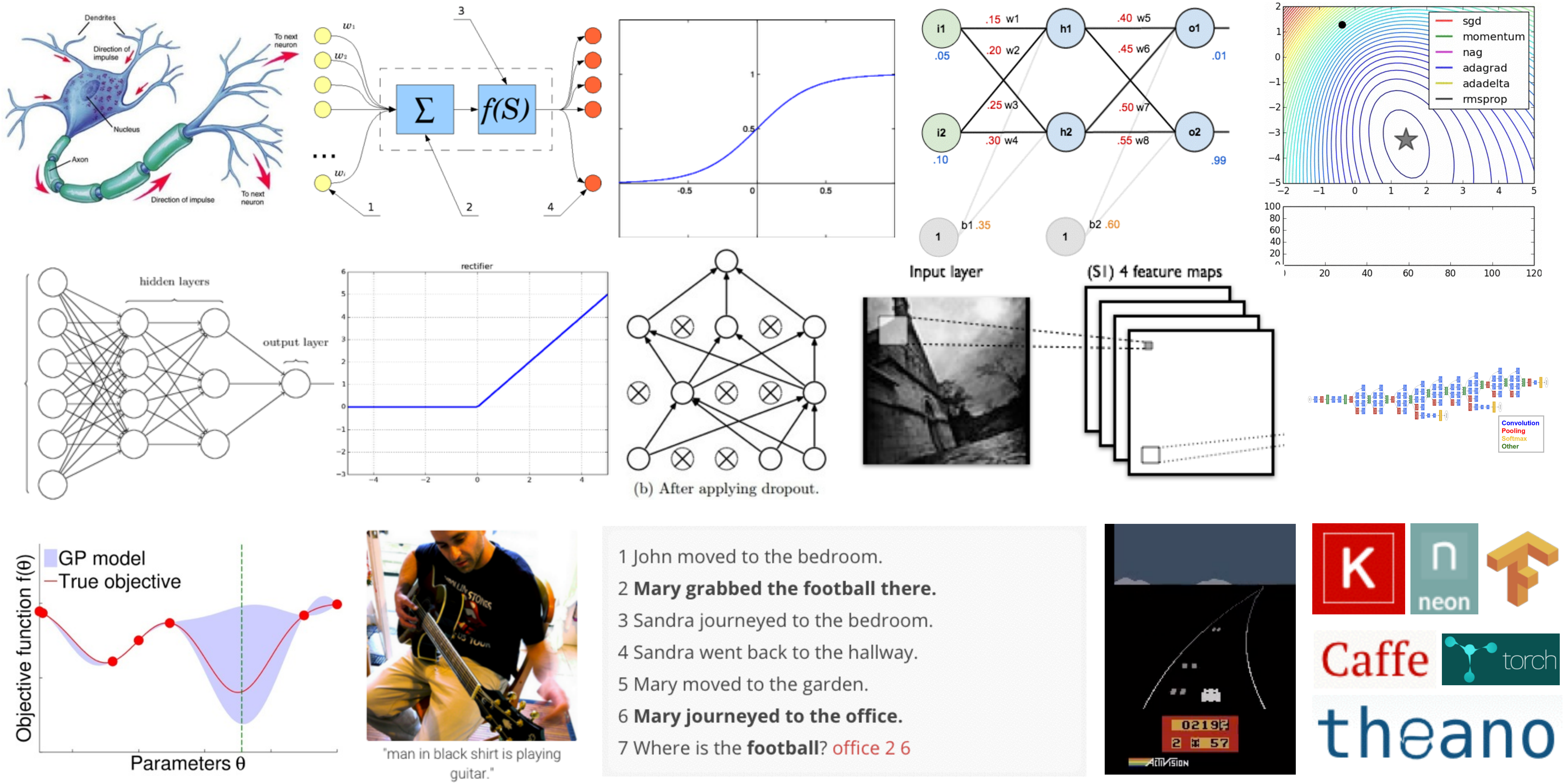
```
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation
from keras.optimizers import SGD

X_train, y_train, X_test, y_test = # LOAD YOUR DATA

model = Sequential()
model.add(Dense(64, input_dim=20, init='uniform', activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, init=                m', activation='tanh'))
model.add(Dropout(0.5))
model.add(Dense(2, init=                ', activation='softmax'))

sgd = SGD(lr=0.1, decay=1e-6, momentum=0.9, nesterov=True
model.compile(loss='mean_squared_error', optimizer=sgd)

model.fit(X_train, y_train, nb_epoch=20, batch_size=16)
score = model.evaluate(X_test, y_test, batch_size=16)
```




(b) After applying dropout.



http://keras.io



- A Step by Step Backpropagation Example
  http://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example
- Online book by By Michael Nielsen
  http://neuralnetworksanddeeplearning.com
- CS231n: Convolutional Neural Networks for Visual Recognition
  http://cs231n.stanford.edu/