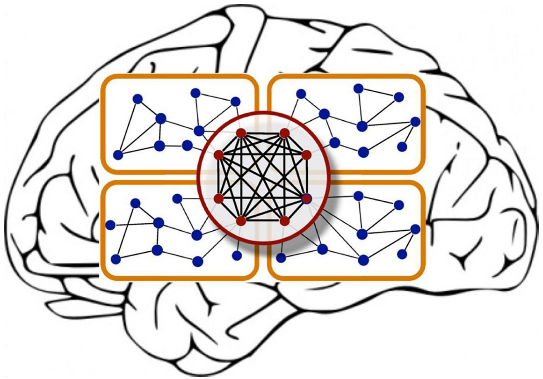


The Fundamentals Of Neural Networks

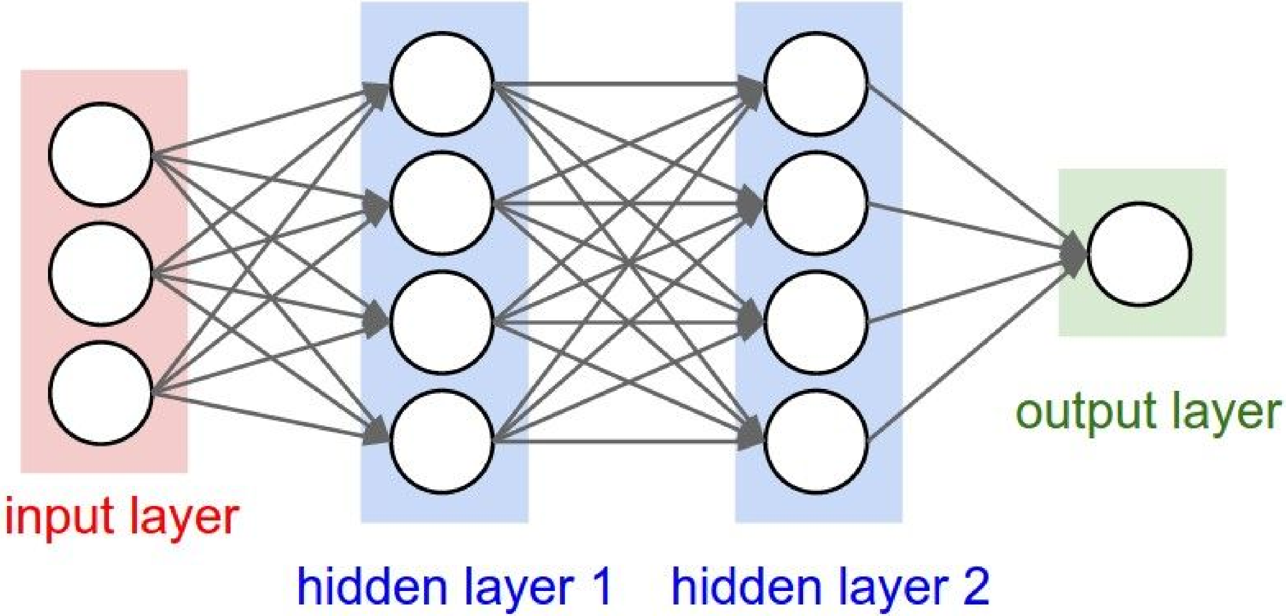
Chris Foster

Machine Learning

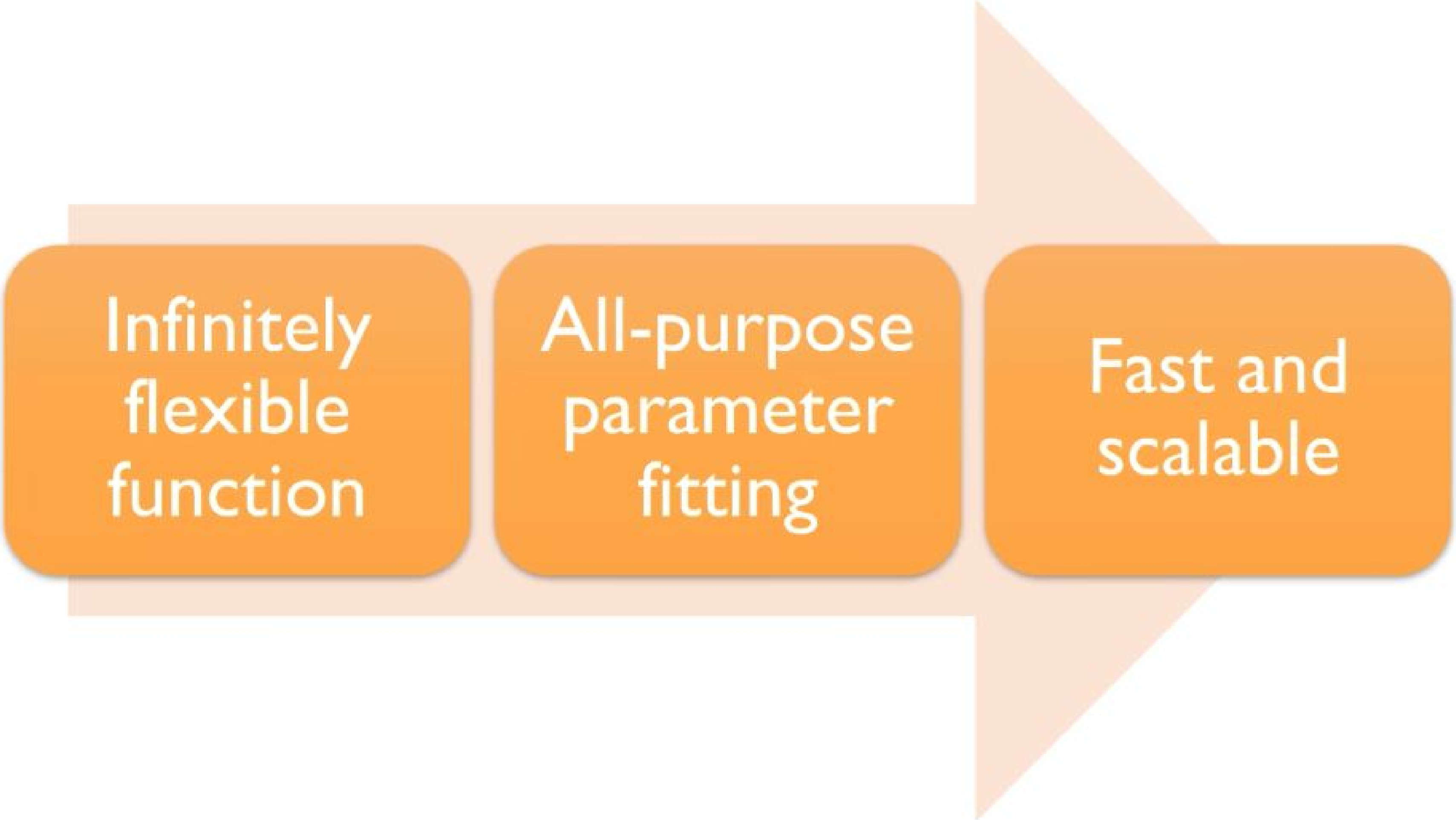
Web Development

Computer Security

Neural Network



“A computer system modeled on the human brain and nervous system”



Infinitely flexible function

All-purpose parameter fitting

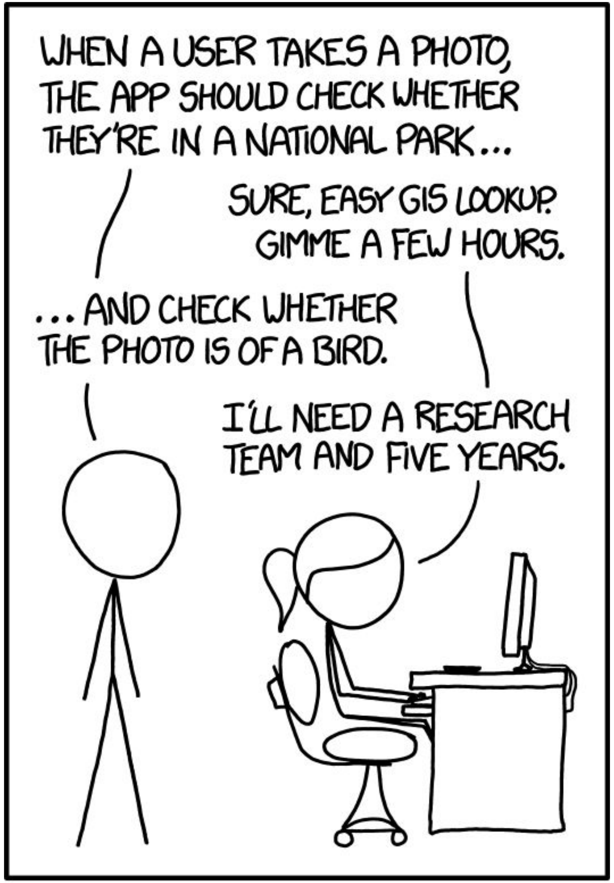
Fast and scalable

How are Neural Networks different?

Applications

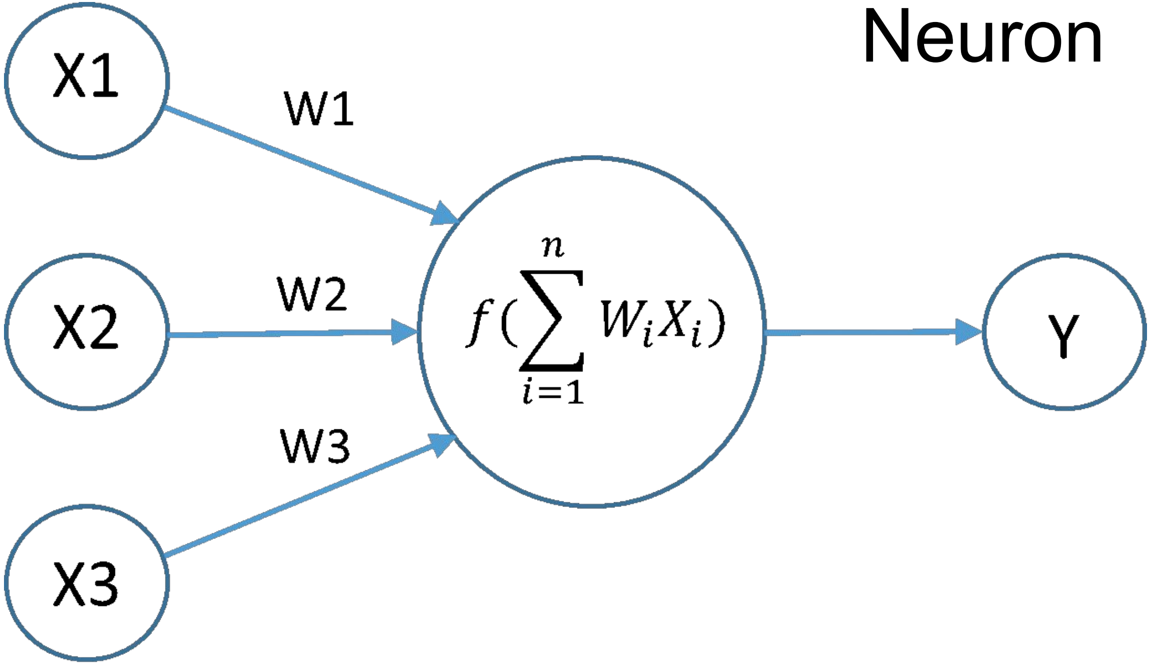
- Self Driving Cars
- Language Translation
- Sentiment Analysis
- Text Generation
- Image Generation
- Object Classification
- Neuroscience
- Image Enhancement
- Audio Transcription
- ...and much more!

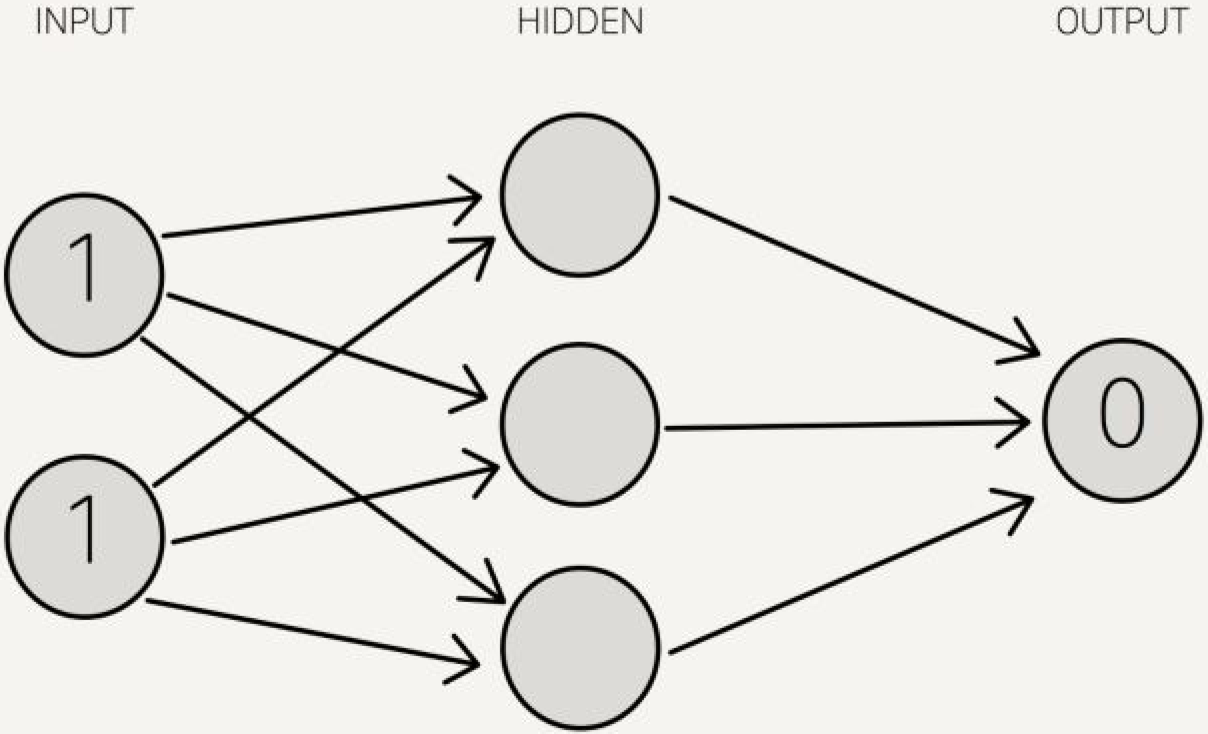
Pattern Recognition Tasks

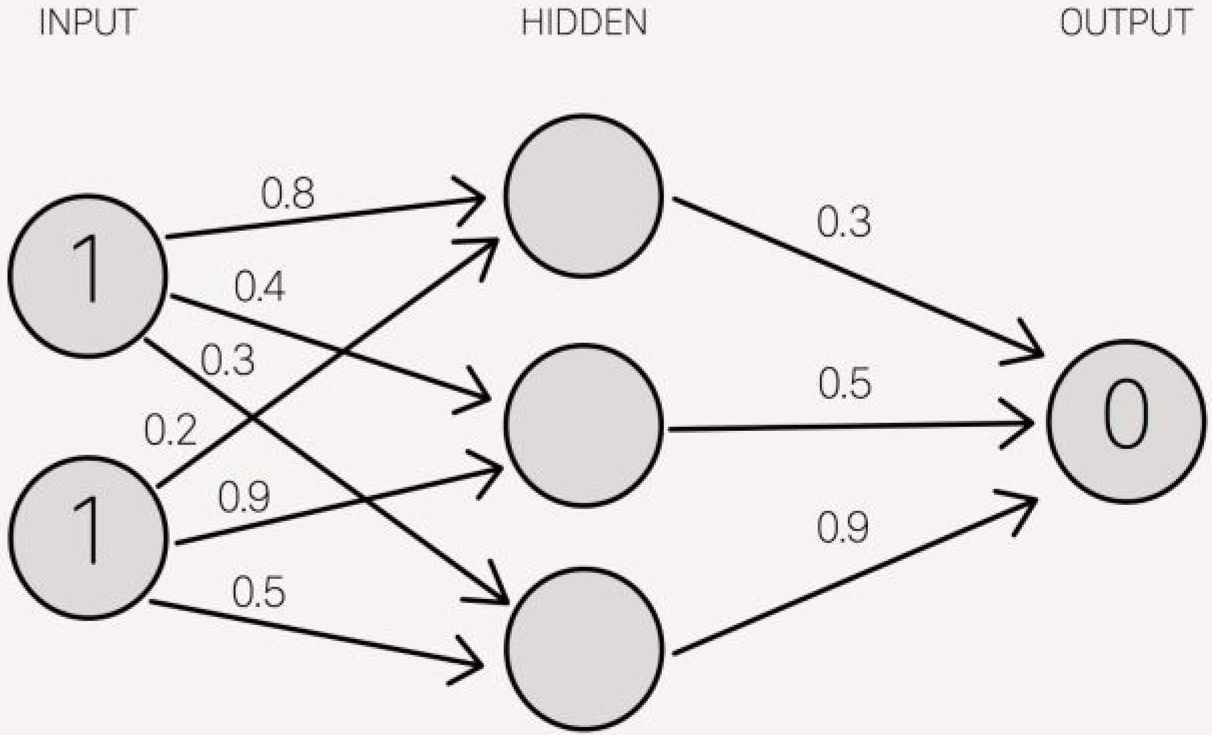

WHEN A USER TAKES A PHOTO, THE APP SHOULD CHECK WHETHER THEY'RE IN A NATIONAL PARK...
SURE, EASY GIS LOOKUP. GIMME A FEW HOURS.
...AND CHECK WHETHER THE PHOTO IS OF A BIRD.
I'LL NEED A RESEARCH TEAM AND FIVE YEARS.

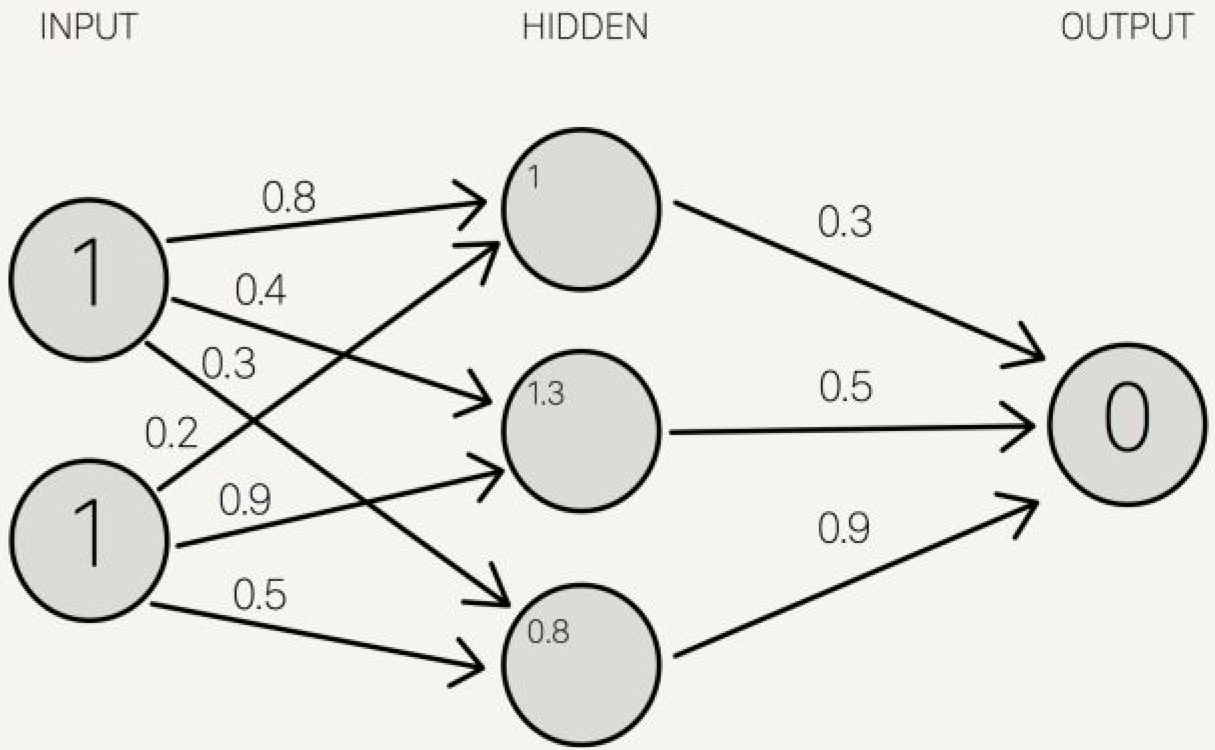
IN CS, IT CAN BE HARD TO EXPLAIN THE DIFFERENCE BETWEEN THE EASY AND THE VIRTUALLY IMPOSSIBLE.

Neuron

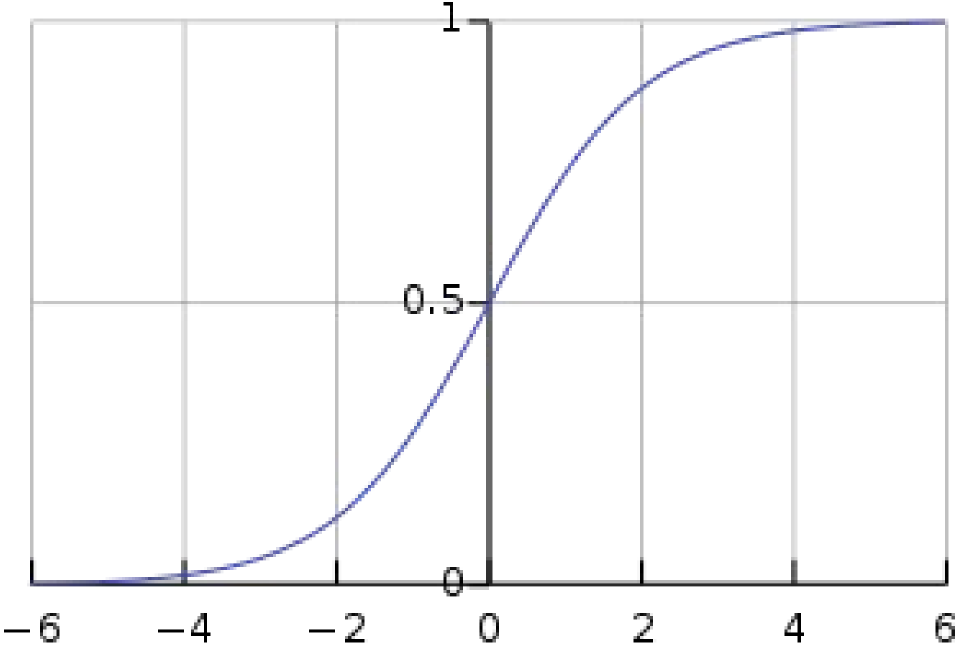


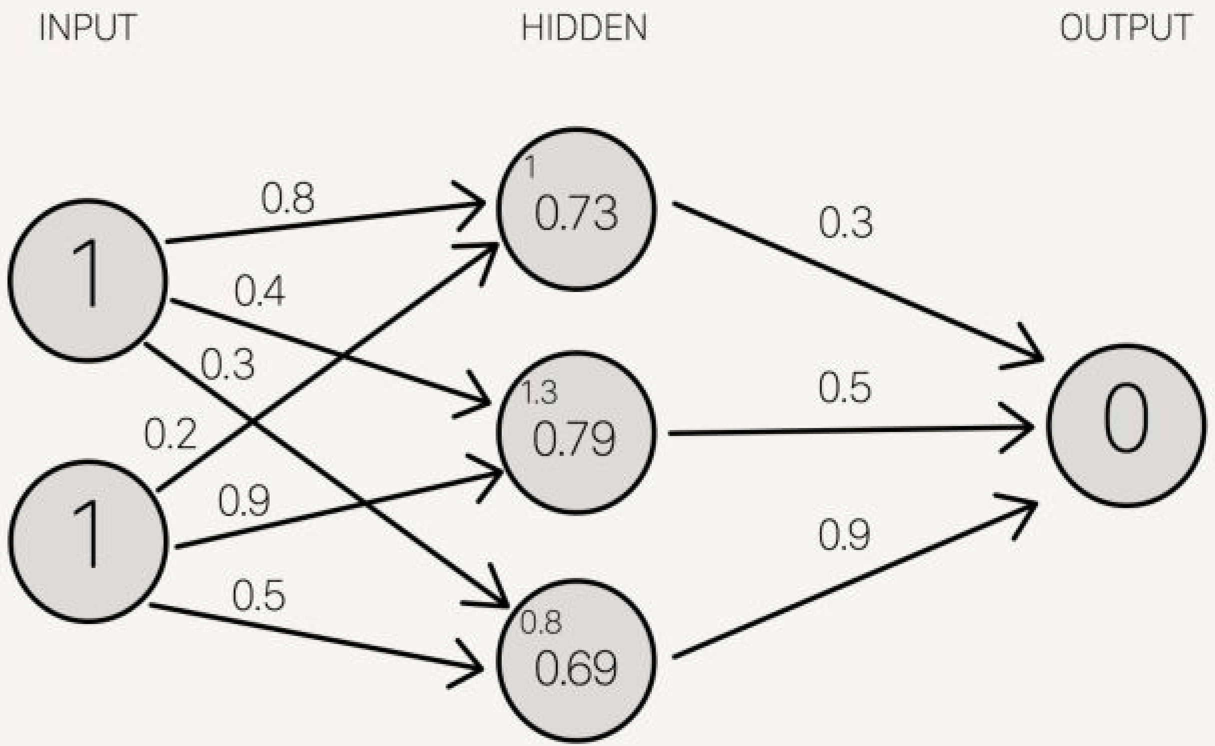

INPUT
HIDDEN
OUTPUT
1
1
0

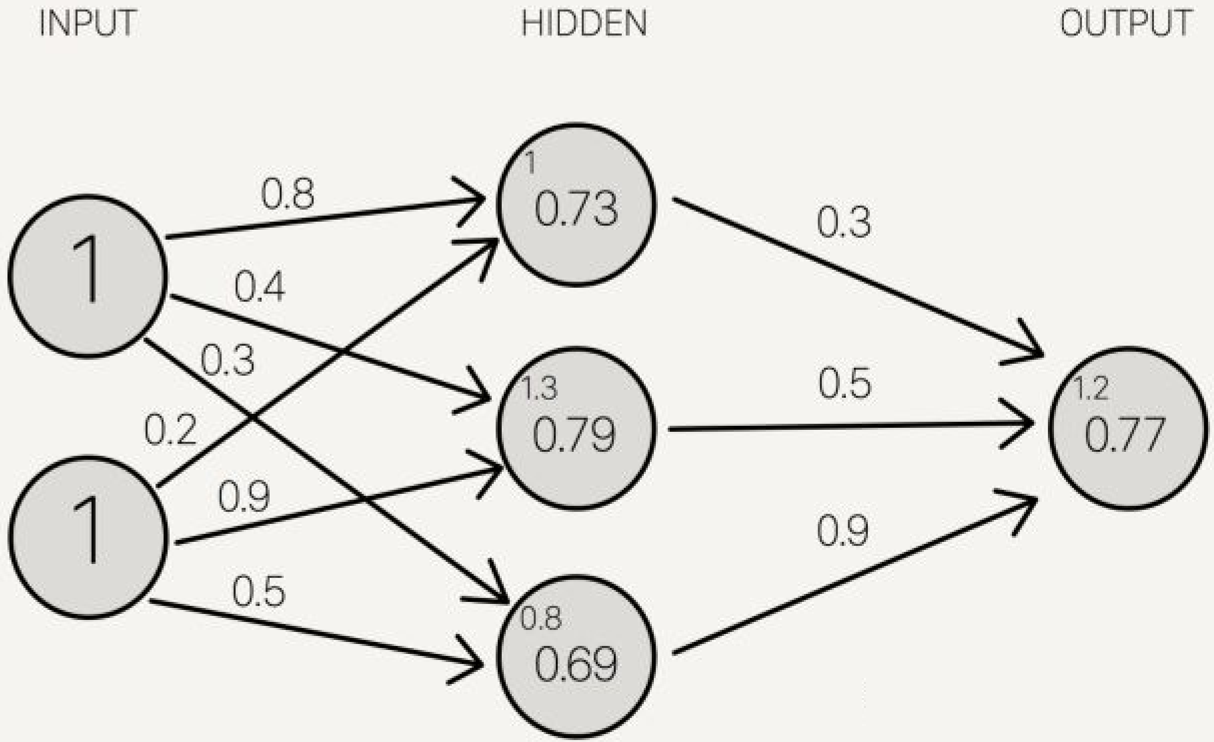

INPUT
HIDDEN
OUTPUT
1
1
0.8
0.4
0.3
0.2
0.9
0.5
0.3
0.5
0.9
0


INPUT
HIDDEN
OUTPUT
1
1
0.8
0.4
0.3
0.2
0.9
0.5
1
1.3
0.8
0.3
0.5
0.9
0

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




INPUT
HIDDEN
OUTPUT
1
1
0.8
0.4
0.3
0.2
0.9
0.5
1
0.73
1.3
0.79
0.8
0.69
0.3
0.5
0.9
0


INPUT
HIDDEN
OUTPUT
1
1
0.8
0.4
0.3
0.2
0.9
0.5
1
0.73
1.3
0.79
0.8
0.69
0.3
0.5
0.9
1.2
0.77

$$net_{h1} = w_1 * i_1 + w_2 * i_2 + b_1 * 1$$

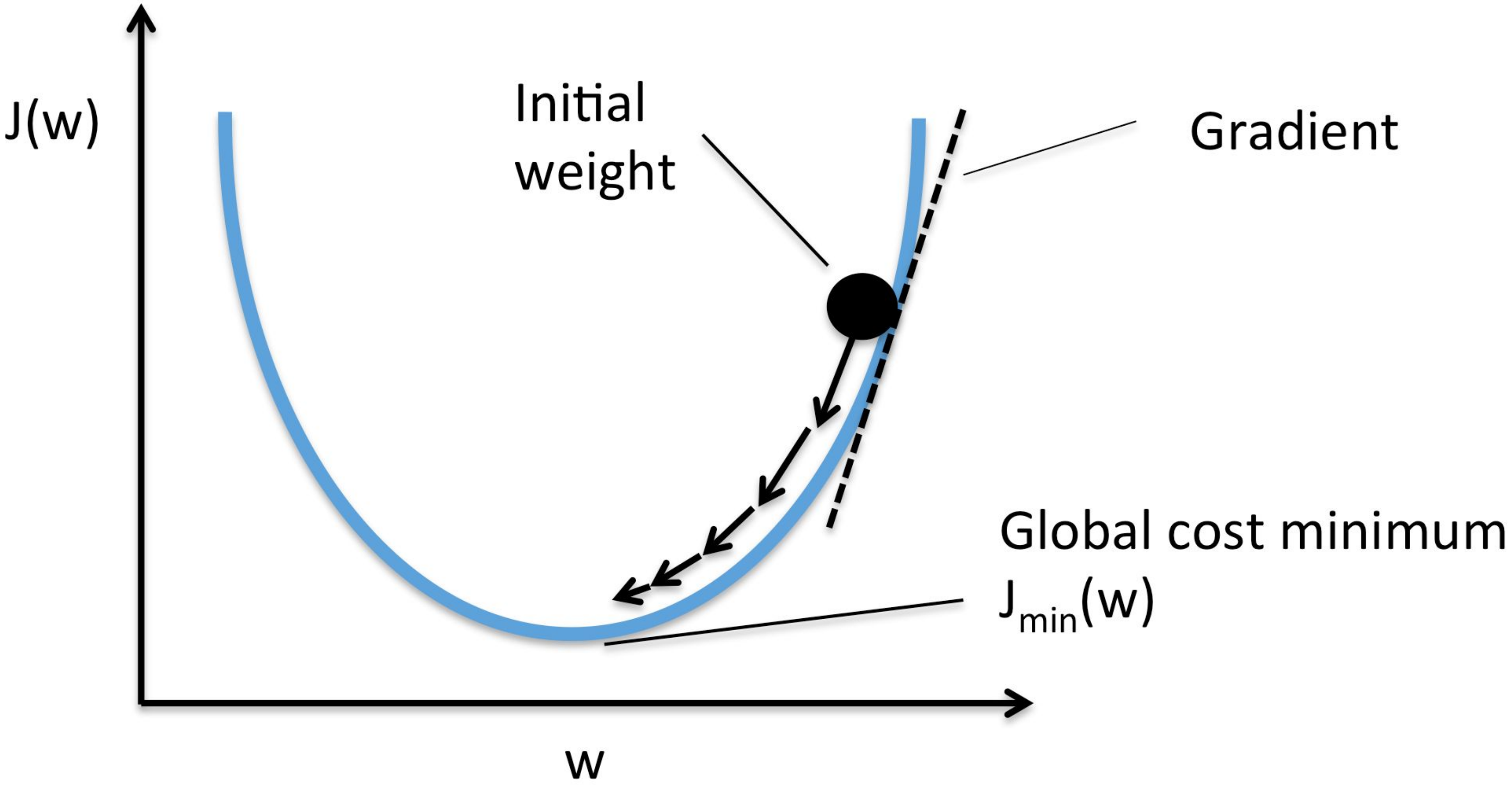
$$net_{h1} = 0.15 * 0.05 + 0.2 * 0.1 + 0.35 * 1 = 0.3775$$

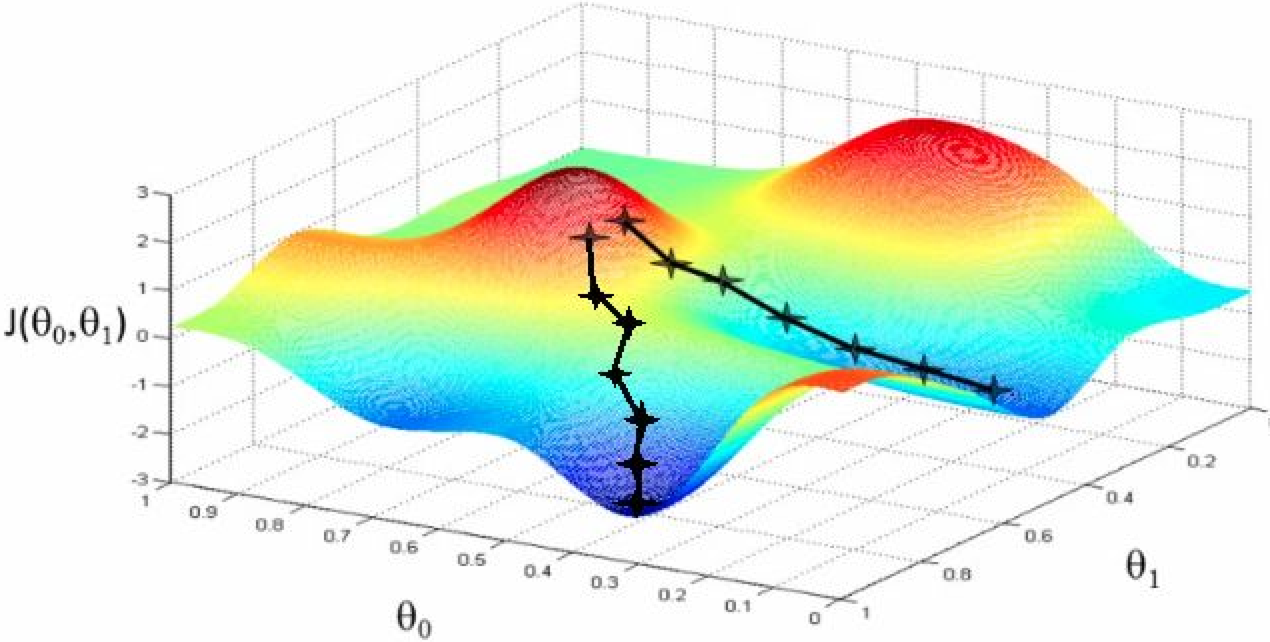
$$out_{h1} = \frac{1}{1+e^{-net_{h1}}} = \frac{1}{1+e^{-0.3775}} = 0.593269992$$

How do we get these weights?

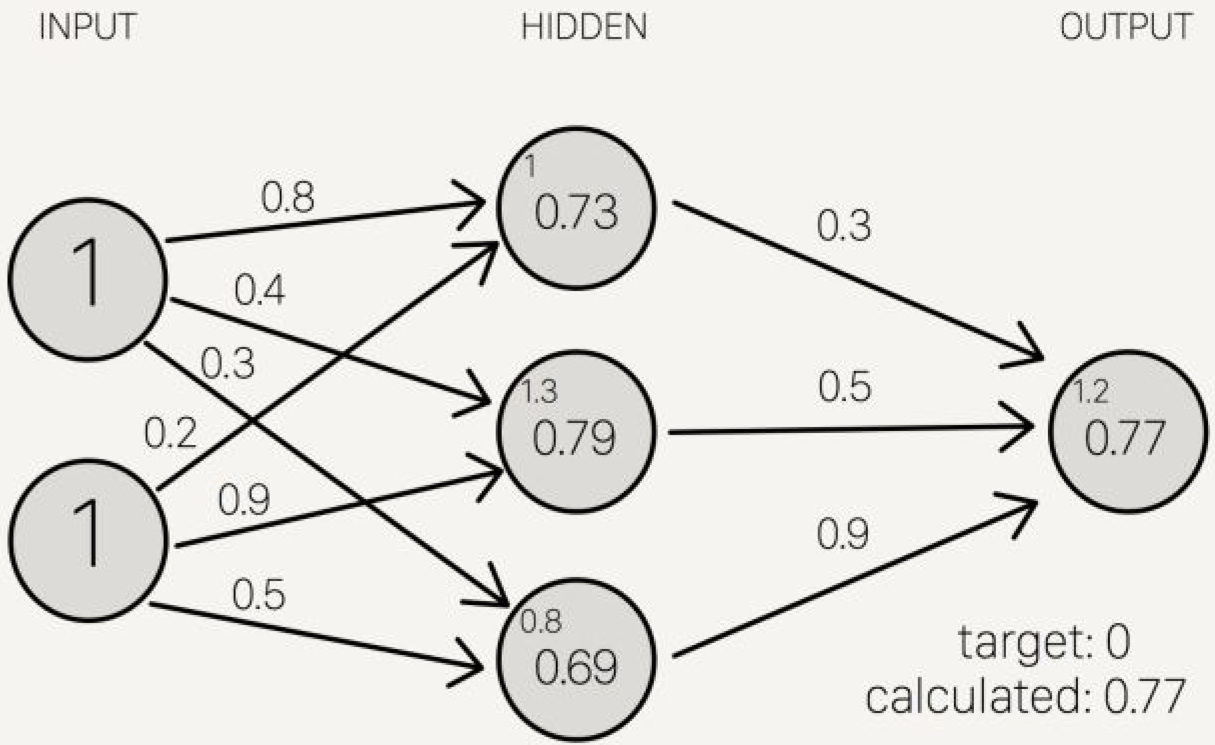
How do we get these weights?

Gradient Descent and Backpropagation


J(w)
Initial weight
Gradient
Global cost minimum
$J_{min}(w)$
w


$J(\theta_0,\theta_1)$
3
2
1
0
-1
-2
-3
θ_0
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_1
0
0.2
0.4
0.6
0.8
1

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



target: 0
calculated: 0.77

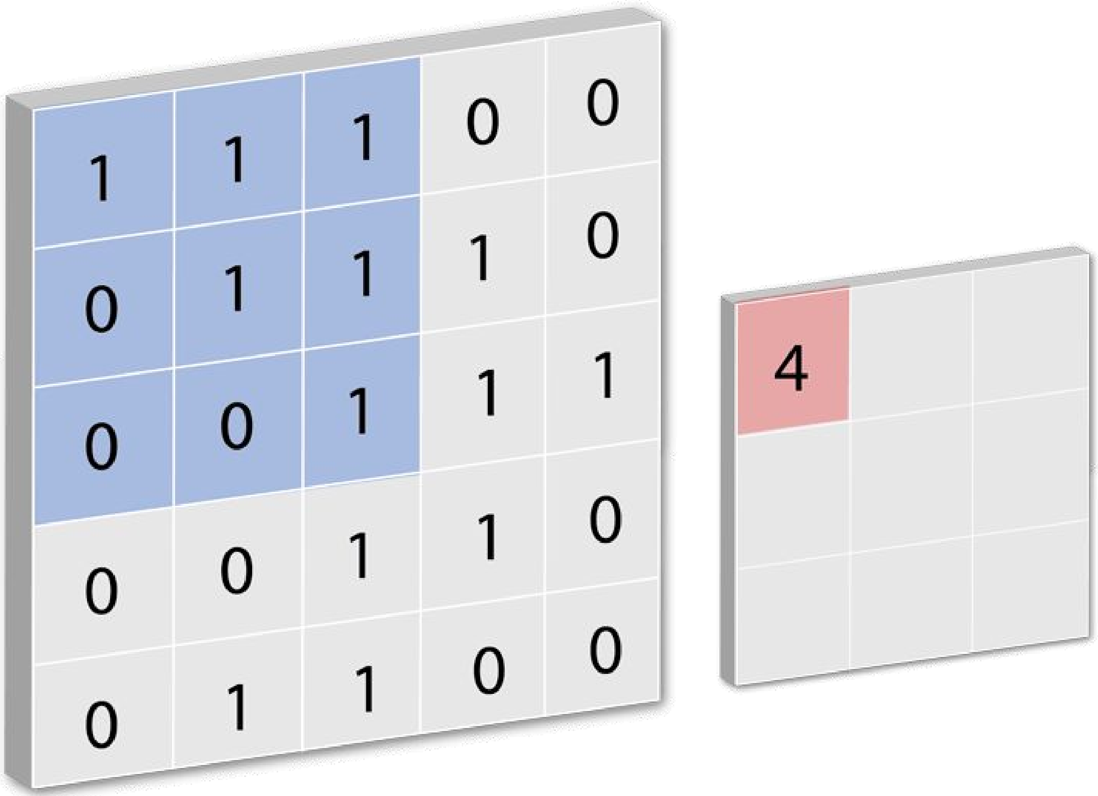
Workshop

Build your own network!

http://tinyurl.com/mq7s7od

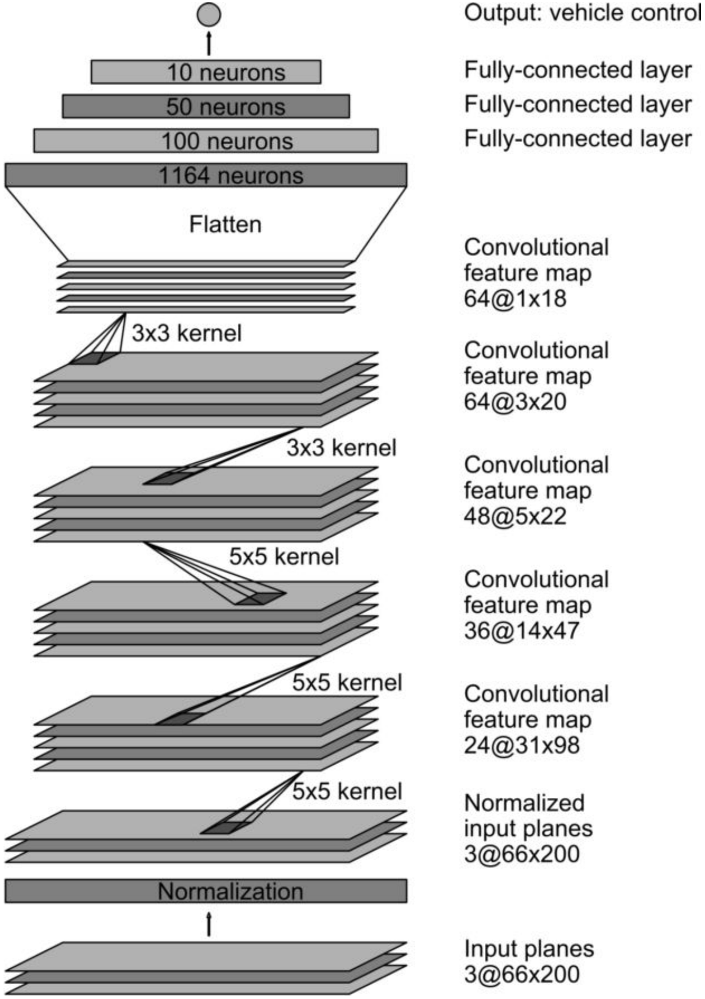
Neural Networks can get a lot more complex

Convolutional Neural Networks

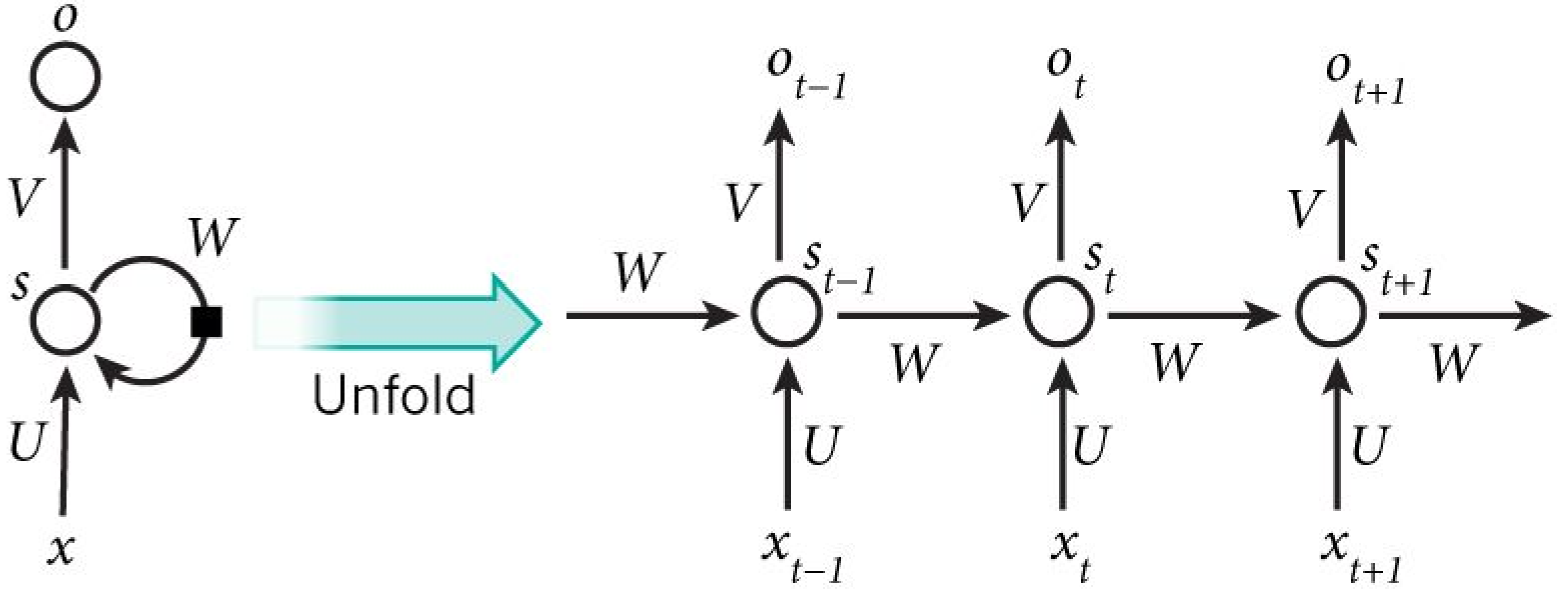


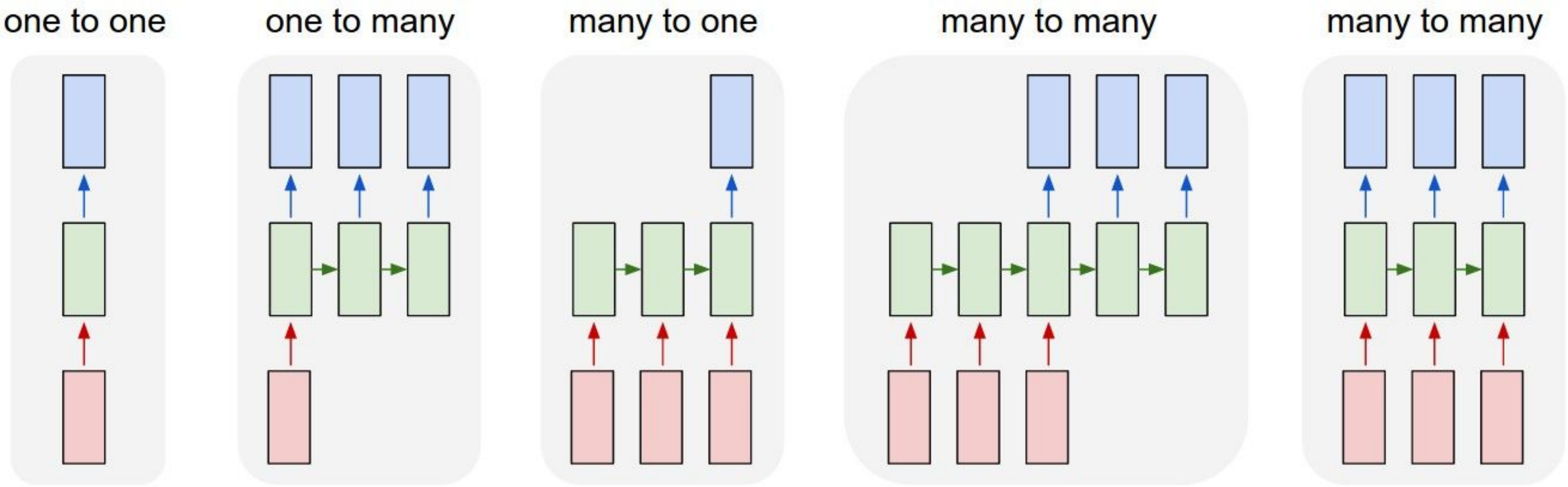
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

4		


Output: vehicle control
10 neurons
Fully-connected layer
50 neurons
Fully-connected layer
100 neurons
Fully-connected layer
1164 neurons
Flatten
Convolutional
feature map
64@1x18
3x3 kernel
Convolutional
feature map
64@3x20
3x3 kernel
Convolutional
feature map
48@5x22
5x5 kernel
Convolutional
feature map
36@14x47
5x5 kernel
Convolutional
feature map
24@31x98
5x5 kernel
Normalized
input planes
3@66x200
Normalization
Input planes
3@66x200

Recurrent Neural Networks





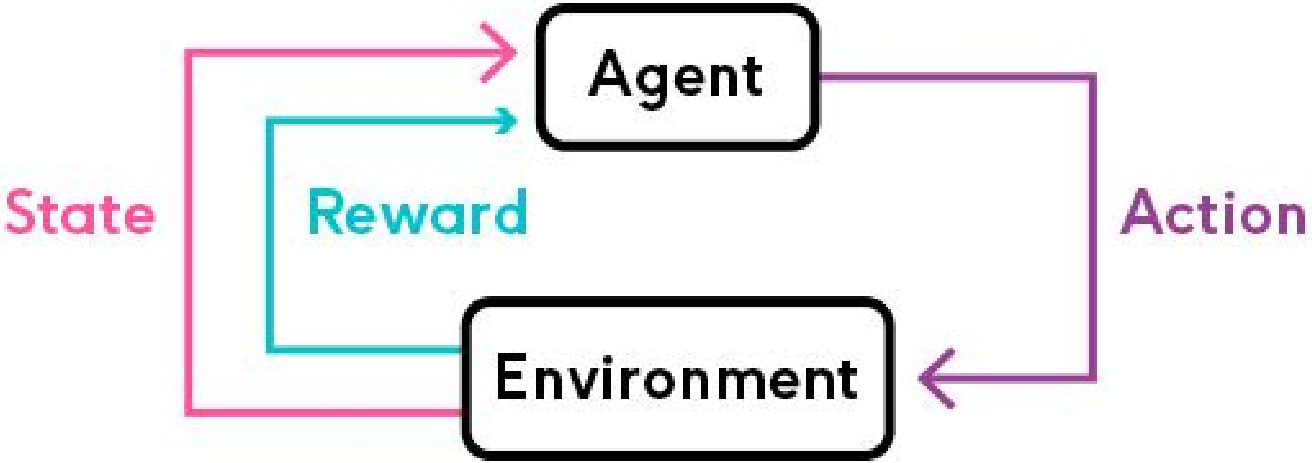
one to one

one to many

many to one

many to many

many to many



Reinforcement Learning

Workshop

Build an RNN!

http://tinyurl.com/jvvm2d9

Next Steps

- Keras - Neural Network LEGO coding
- Tensorflow - Efficient tensor computation library

- fast.ai - Developer focused Neural Network course
- Deep Learning Book - Rigorous Neural Networks

Thanks!

Sources

http://cs231n.github.io
http://www.fast.ai
https://xkcd.com
https://stevenmiller888.github.io
https://commons.wikimedia.org
http://www.scirp.org
https://sebastianraschka.com
https://medium.com/@eamonabraham
https://github.com/selva86
http://machinelearningmastery.com
https://phrasee.co
https://devblogs.nvidia.com
http://karpathy.github.io
http://www.deeplearningbook.org