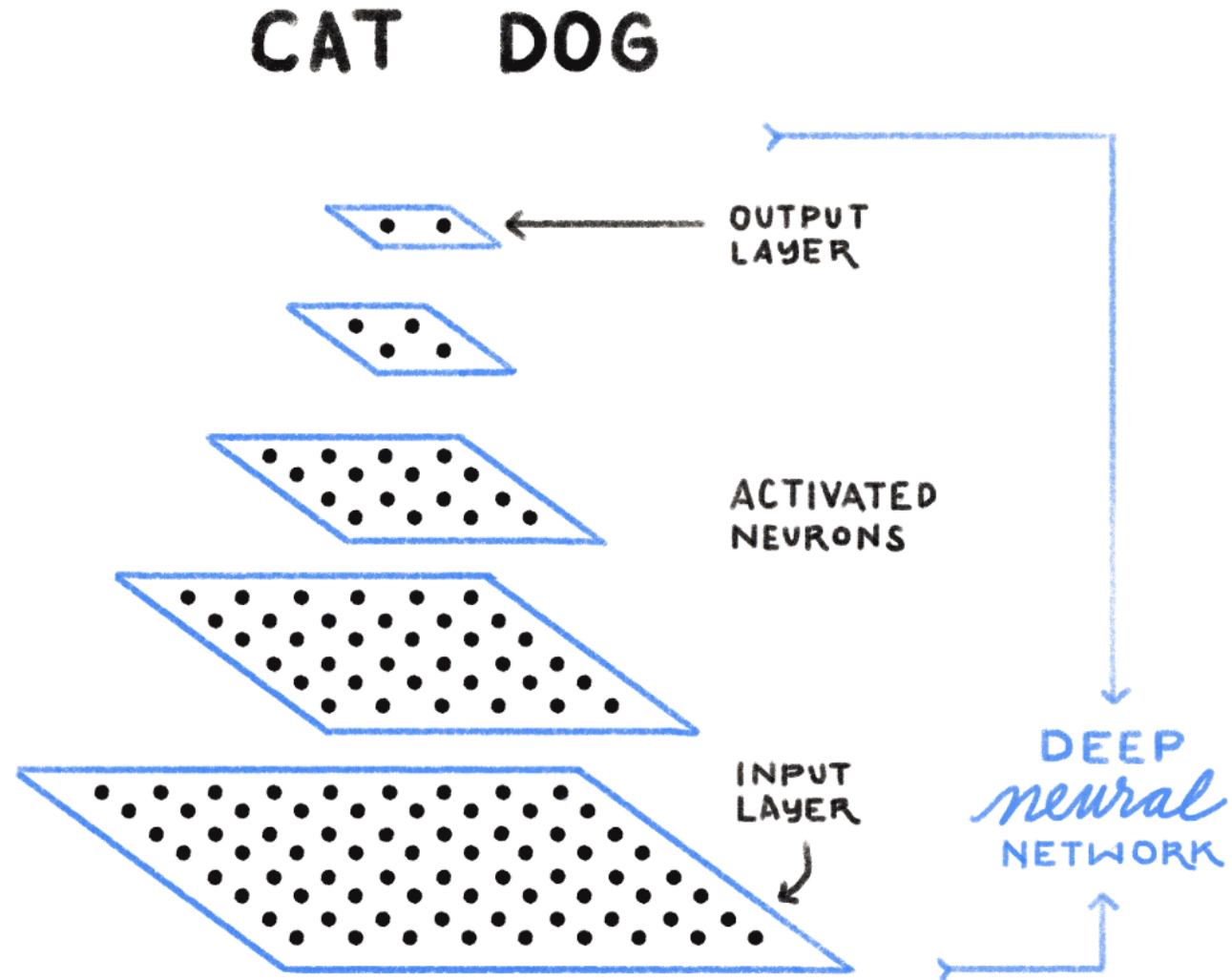


# Artificial Neural Networks

IS THIS A  
**CAT** or **DOG**?



# What are they?

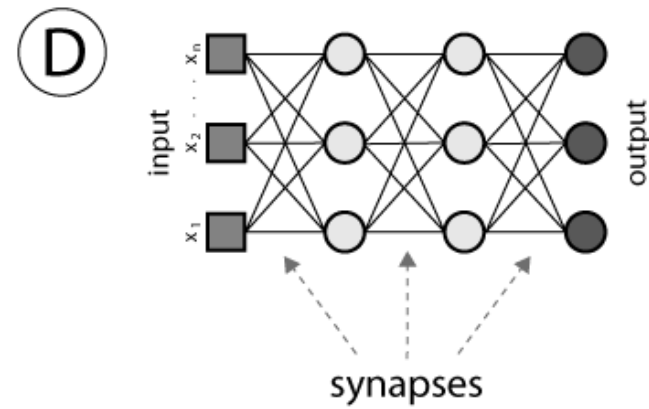
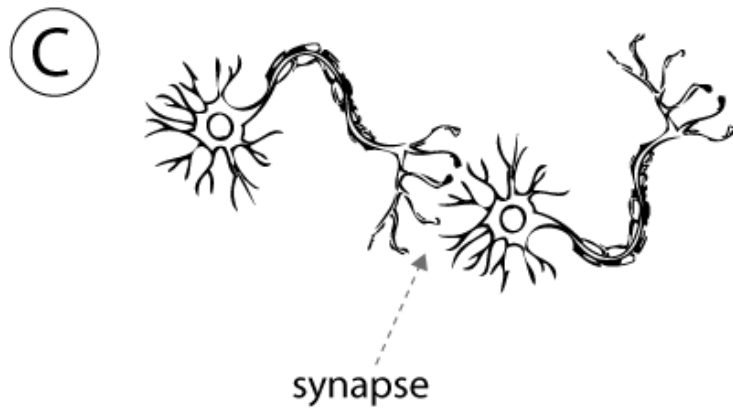
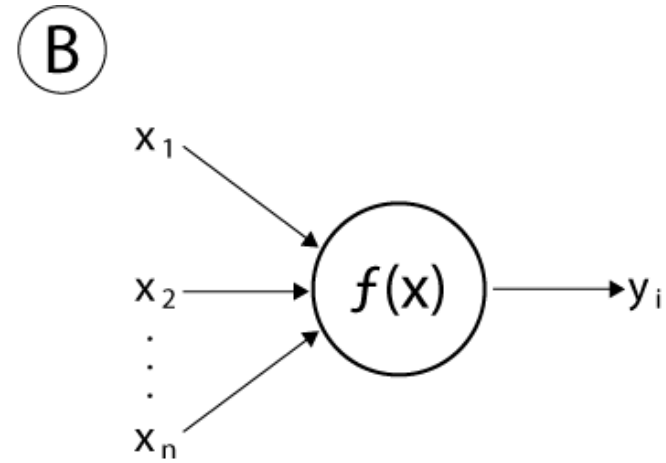
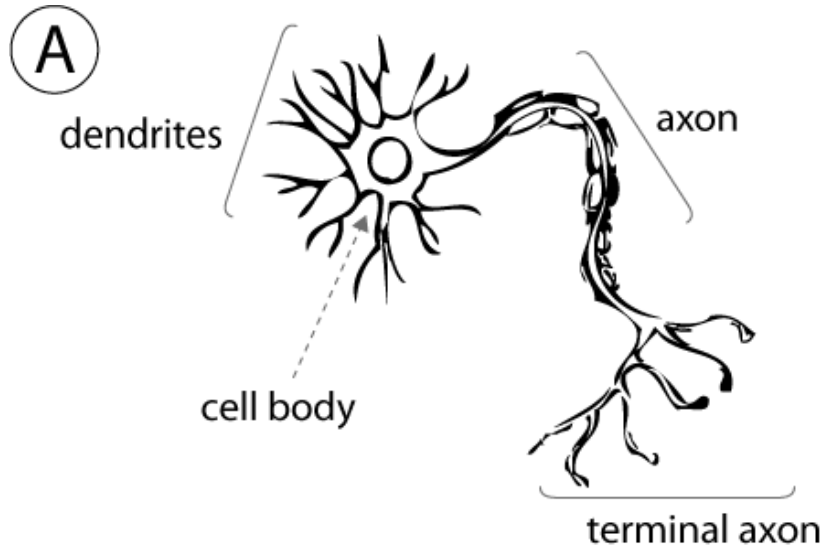
Inspired by the Human Brain.

The human brain has about 86 Billion neurons and requires 20% of your body's energy to function.

These neurons are connected to between 100 Trillion to 1 Quadrillion synapses!



# What are they?



# What are they?

1. Originally developed by [Warren McCulloch](#) and [Walter Pitts<sup>\[3\]</sup>](#) (1943)
2. Started off as an unsupervised learning tool.
  1. Had problems with computing time and could not compute XOR
  2. Was abandoned in favor of other algorithms
3. [Werbos](#)'s (1975) [backpropagation](#) algorithm
  1. Incorporated supervision and solved XOR
  2. But were still too slow vs. other algorithms e.g., Support Vector Machines
4. Backpropagation was accelerated by GPUs in 2010 and shown to be more efficient and cost effective

# GPUS

GPUS handle parallel operations much better (thousands of threads per core) but are not as quick as CPUs. However, the matrix multiplication steps in ANNs can be run in parallel resulting in considerable time + cost savings. The best CPUs handle about 50GB/s while the best GPUs handle 750GB/s memory bandwidth.

PNY - NVIDIA GeForce GT 710 VERTO 2GB DDR3 PCI Express 2.0  
Graphics Card - Black  
**2GB DDR3 (64-bit) on-board memory**

Plus 192 CUDA processing cores and up to 12.8GB/sec.

 **PRICE MATCH** GUARANTEE

**\$59.99**

AMD - 1600 Six-Core 3.2 GHz Desktop Processor - White  
Model: YD1600BBAEBOX SKU: 6091101

 **PRICE MATCH** GUARANTEE

**\$189.99**

	Google	Stanford
<b>Number of cores</b>	1K CPUs = 16K cores	3GPUs = 18K cores
<b>Cost</b>	\$5B	\$33K
<b>Training time</b>	week	week

# Applications

<http://news.mit.edu/2017/artificial-intelligence-suggests-recipes-based->



Pic2Recipe, an artificial intelligence system developed at MIT, can take a photo of an entree and suggest a similar recipe to it.

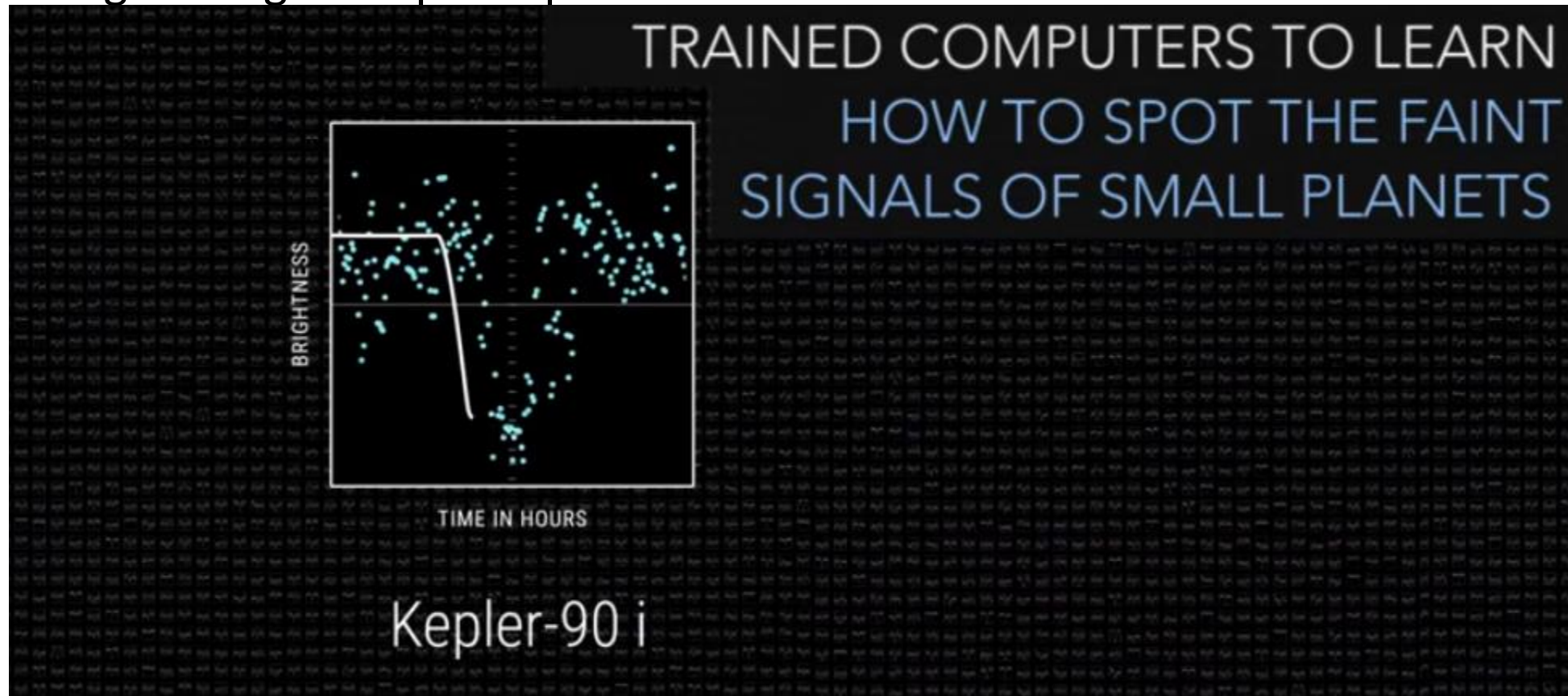
Photo: Jason Dorfman/MIT CSAIL



# Applications

<https://www.nasa.gov/press-release/artificial-intelligence-nasa-data-used-to-discover-eighth-planet-circling-distant-star>

Images of light drop compared and new ones found.



# Idea behind them

1. Obtain some structured data (always a good idea 😊).
2. Use some subset of that data as training
3. Feed each training example through the network
  1. Calculate the error for each training example
  2. Update the weights for each neuron to minimize the error using Gradient Descent (Back Propagation)
  3. Feed in the data again until you reach the desired % error or trials run out
  4. If you reached % error or trials stop and go to the next training input
    1. Else (Back Propagation)

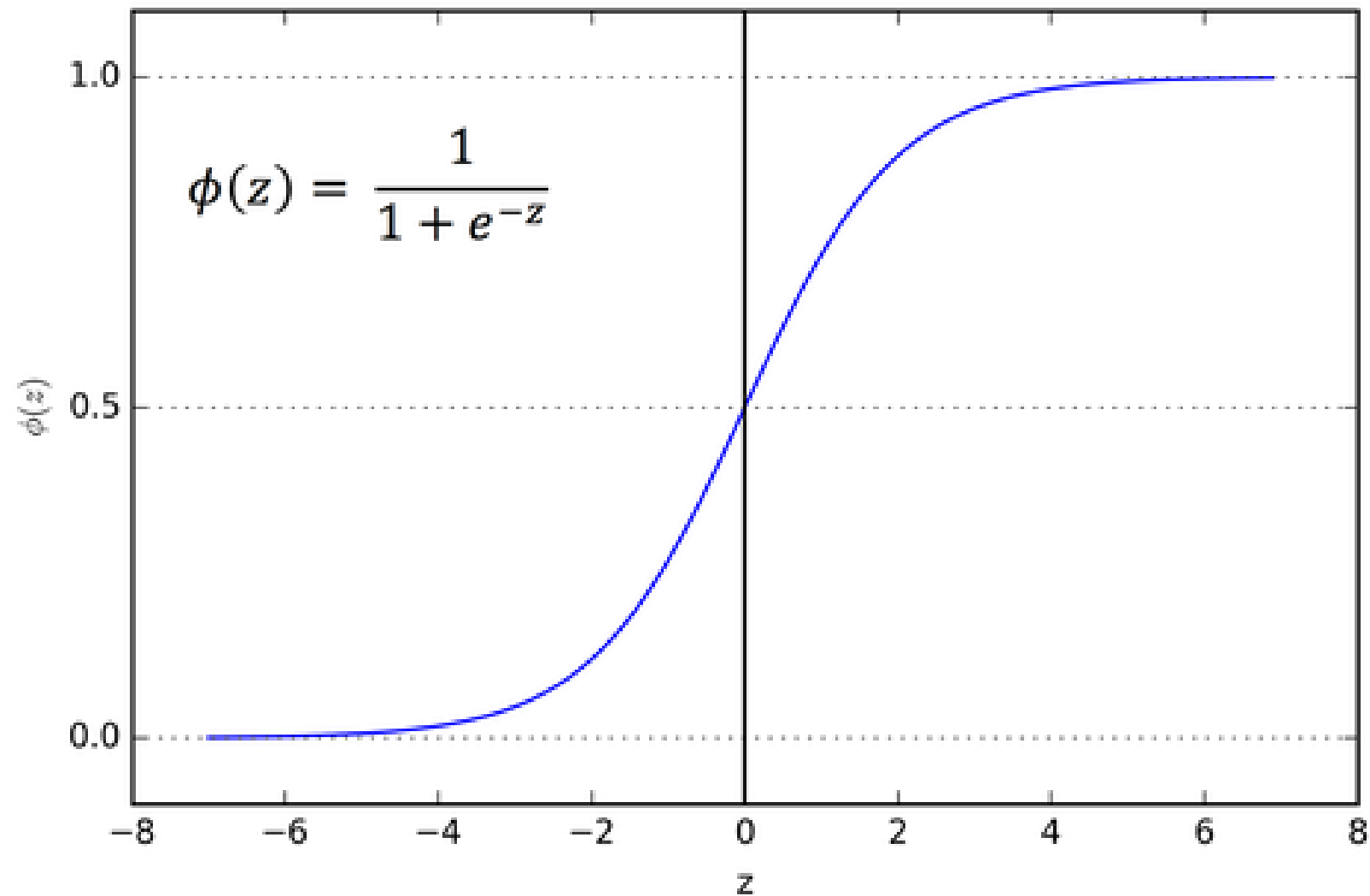


# An example

# Forward Propagation

1. Assign random weights to the synapses
2. Feed in the training data
3. Calculate the hidden layers neurons from the inputs and the weights using an activation function
4. Calculate the output from the hidden layer neurons and the output weights
5. Calculate the error from what is expected

# Activation Function



# Forward propagation

# Back propagation

We need to adjust the weights to minimize the error

# Back propagation

# Back Propagation

$$\left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2}$$




# Live Example!

<https://teachablemachine.withgoogle.com/>


← → ↻ 🏠 <https://teachablemachine.withgoogle.com/>

INPUT



LEARNING


41 EXAMPLES



CONFIDENCE

TRAIN GREEN


37 EXAMPLES



CONFIDENCE

TRAIN PURPLE

35 EXAMPLES







CONFIDENCE

TRAIN ORANGE

OUTPUT

[GIF](#) Sound Speech





# Strike the pose

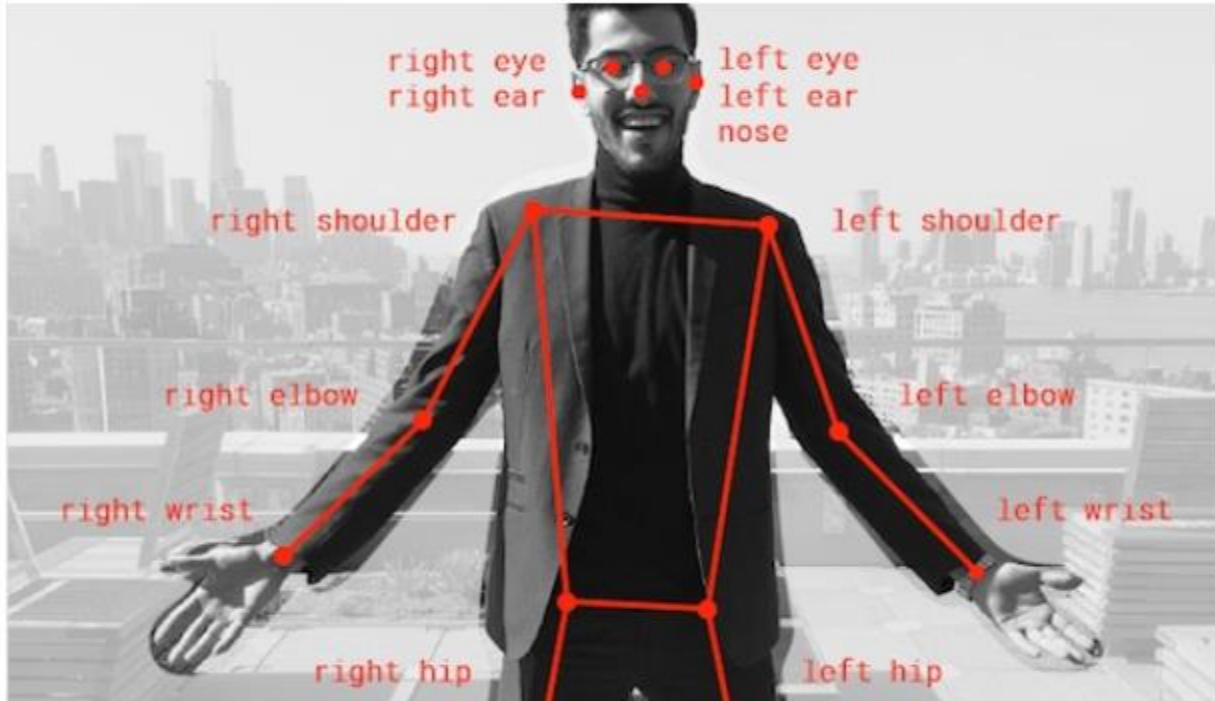
<https://storage.googleapis.com/tfjs-models/demos/posenet/camera.html>

## POSENET

Real-time [Human Pose Estimation](#) in the browser.

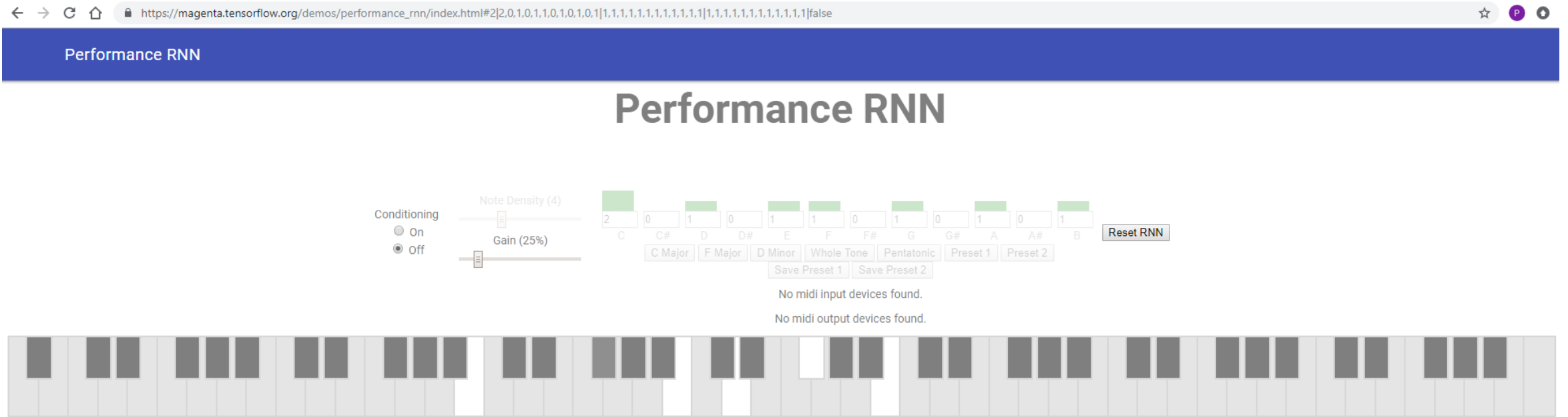
GO TO DEMO!

CODE



# Can a machine play music?

[https://magenta.tensorflow.org/demos/performance\\_rnn/index.html#2|2,0,1,0,1,1,0,1,0,1,0,1|1,1,1,1,1,1,1,1,1,1,1,1|1,1,1,1,1,1,1,1,1,1|false](https://magenta.tensorflow.org/demos/performance_rnn/index.html#2|2,0,1,0,1,1,0,1,0,1,0,1|1,1,1,1,1,1,1,1,1,1,1,1|1,1,1,1,1,1,1,1,1,1|false)



[Performance RNN](#) was trained in TensorFlow on MIDI from piano performances. It was then ported to run in the browser using only Javascript in the [TensorFlow.js](#) environment. Piano samples are from [Salamander Grand Piano](#).

# Tensor Flow & Keras

```
import numpy as np
from keras.models import Sequential
from keras.layers.core import Dense

# the four different states of the XOR gate
training_data = np.array([[0,0],[0,1],[1,0],[1,1]], "float32")

# the four expected results in the same order
target_data = np.array([[0],[1],[1],[0]], "float32")

#use sequential vs functional since we're feedforward
model = Sequential()
#Dense is used for single input data 0,1,1,0 for each neuron
model.add(Dense(16, input_dim=2, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

#this is the building of the net
model.compile(loss='mean_squared_error',
              optimizer='adam',
              metrics=['binary_accuracy'])

#now optimize
model.fit(training_data, target_data, nb_epoch=500, verbose=2)

print model.predict(training_data).round()
```

# Derivative of the sigmoid

Let's denote the sigmoid function as  $\sigma(x) = \frac{1}{1 + e^{-x}}$ .

The derivative of the sigmoid is  $\frac{d}{dx}\sigma(x) = \sigma(x)(1 - \sigma(x))$ .

Here's a detailed derivation:

$$\begin{aligned}\frac{d}{dx}\sigma(x) &= \frac{d}{dx} \left[ \frac{1}{1 + e^{-x}} \right] \\&= \frac{d}{dx} (1 + e^{-x})^{-1} \\&= -(1 + e^{-x})^{-2} (-e^{-x}) \\&= \frac{e^{-x}}{(1 + e^{-x})^2} \\&= \frac{1}{1 + e^{-x}} \cdot \frac{e^{-x}}{1 + e^{-x}} \\&= \frac{1}{1 + e^{-x}} \cdot \frac{(1 + e^{-x}) - 1}{1 + e^{-x}} \\&= \frac{1}{1 + e^{-x}} \cdot \left( \frac{1 + e^{-x}}{1 + e^{-x}} - \frac{1}{1 + e^{-x}} \right) \\&= \frac{1}{1 + e^{-x}} \cdot \left( 1 - \frac{1}{1 + e^{-x}} \right) \\&= \sigma(x) \cdot (1 - \sigma(x))\end{aligned}$$