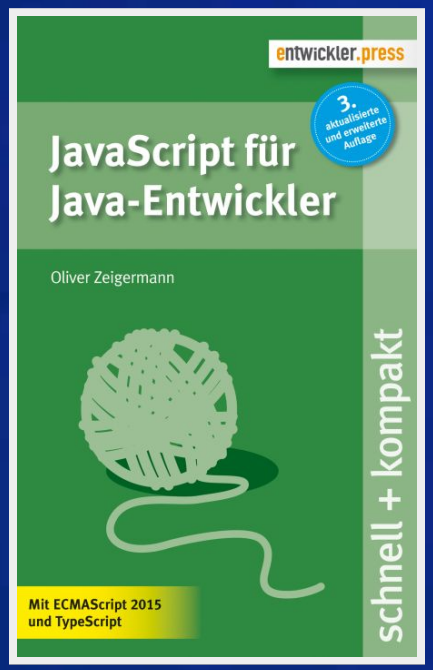
Machine Learning in the Browser with Deep Neural Networks

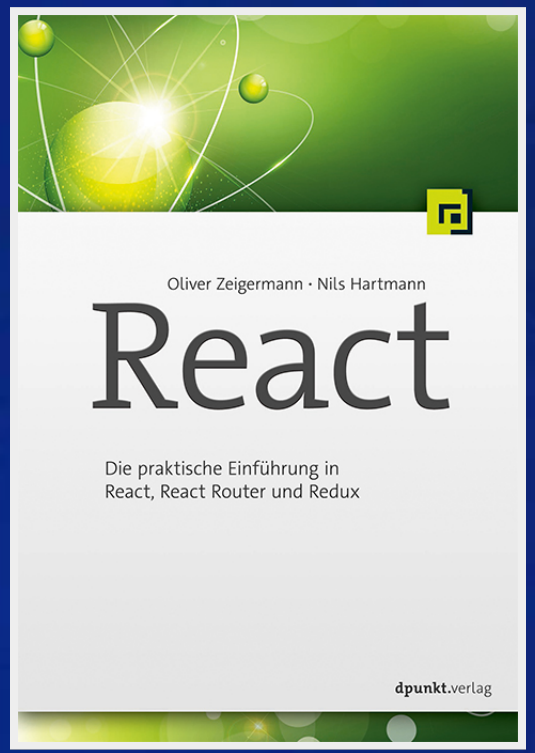
data2day 2016

Oliver Zeigermann / @DJCordhose

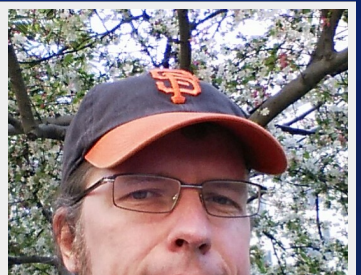
http://bit.ly/ml-data2day


entwickler.press
3.
aktualisierte und erweiterte Auflage
JavaScript für Java-Entwickler
Oliver Zeigermann
schnell + kompakt
Mit ECMAScript 2015 und TypeScript


entwickler.press
JavaScript und TypeScript für C#-Entwickler
Oliver Zeigermann
schnell + kompakt


Oliver Zeigermann · Nils Hartmann
React
Die praktische Einführung in React, React Router und Redux
dpunkt.verlag

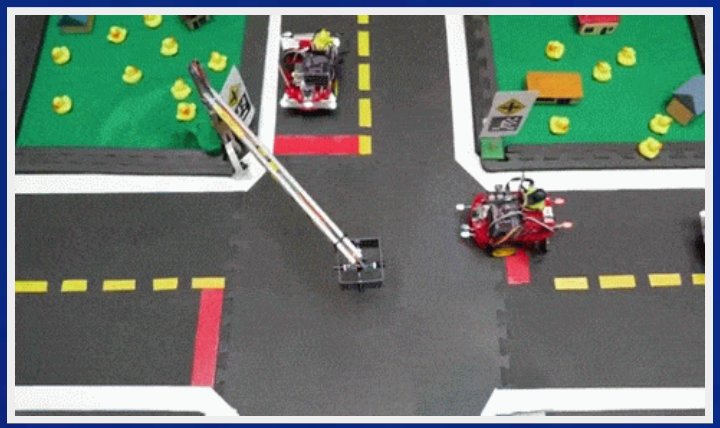

embarc

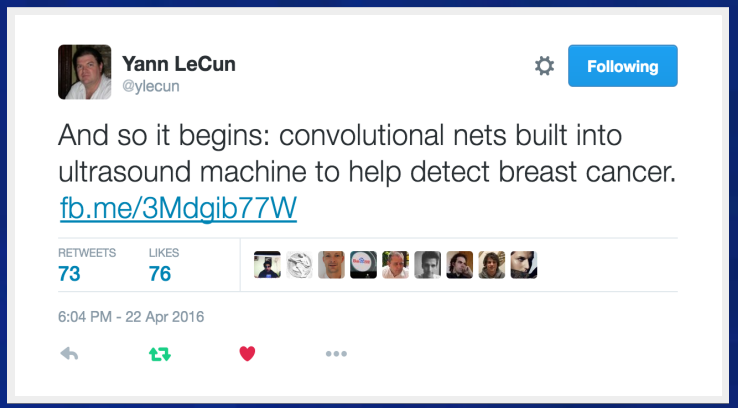


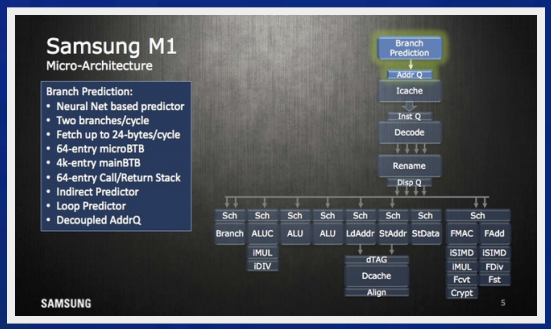
What is machine learning?

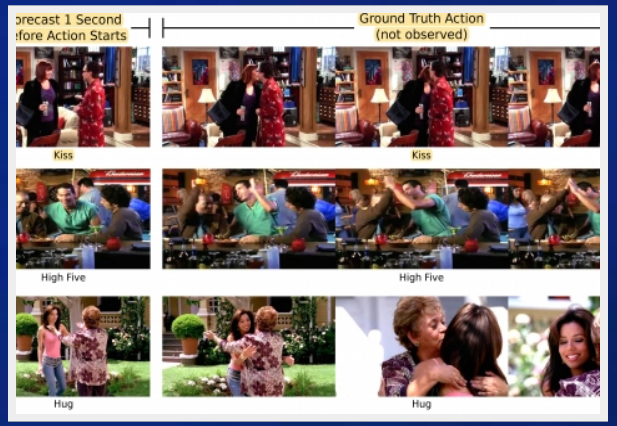
science of getting computers to act without being explicitly programmed

Applications of machine learning?









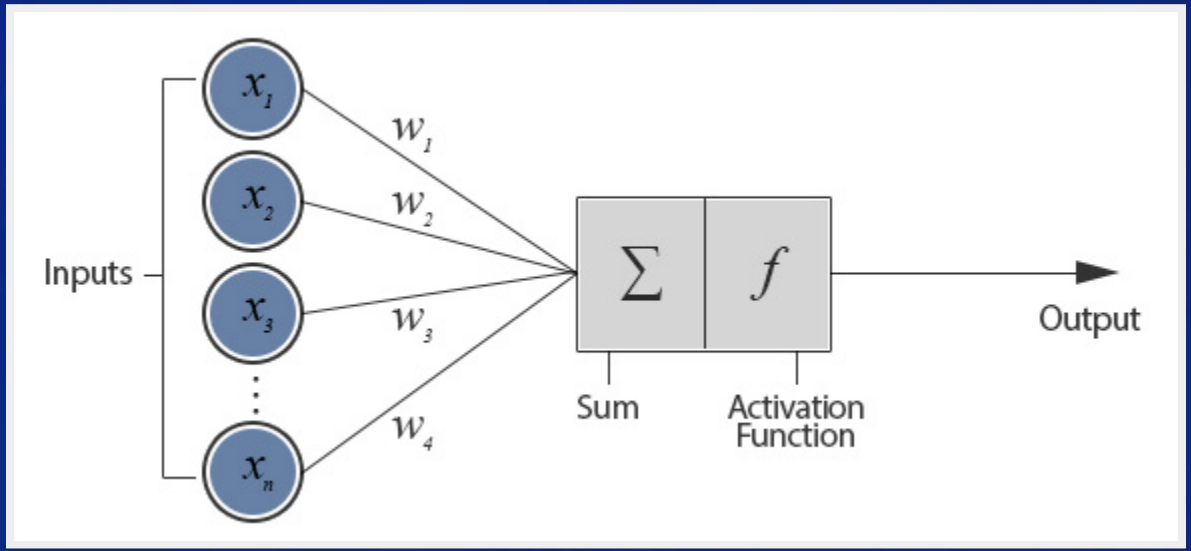
Why JavaScript for Machine Learning?

- Python and R are predominant
  - Have a large and mature set of libs
  - Are reasonably fast
  - Using binding to C/C++ or Fortran
- JavaScript has benefits, though
  - might be the language you are most comfortable with
  - might be the only language around (because all you have is a browser)
  - zero installation, easy to get started
  - combination with interactive visualizations

Visualizing the Basics

The perceptron - where it all begins

- mathematical model of a biological neuron
- creates a single output based on sum of many weighted inputs
- uses a sigmoid activation function to create boolean like output



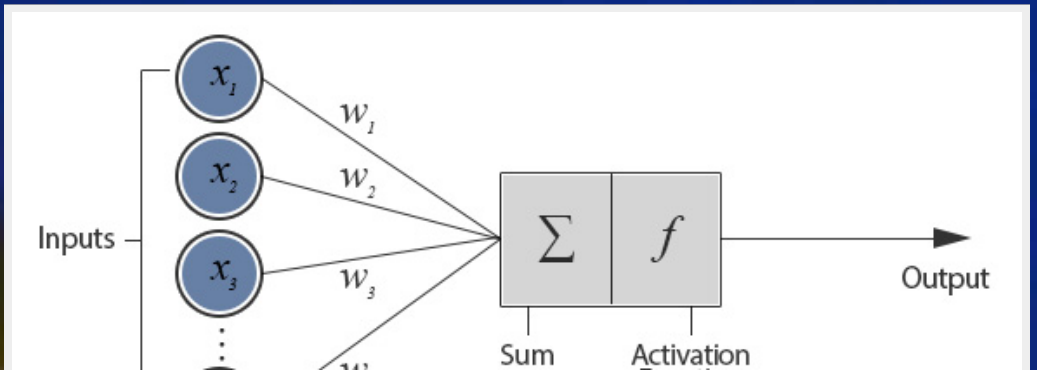
http://www.theprojectspot.com/tutorial-post/introduction-to-artificial-neural-networks-part-1/7

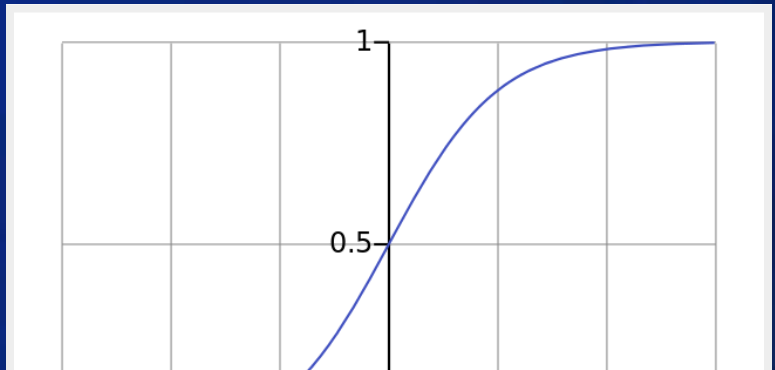
Implementing it in pure JavaScript

```
// Initial weights
let w0 = 3, w1 = -4, w2 = 2;
```

```
function perceptron(x1, x2) {
    const sum = w0 + w1 * x1 + w2 * x2;
    return sigmoid(sum);
}
```

```
function sigmoid(z) {
    return 1 / (1 + Math.exp(z * -1));
}
```



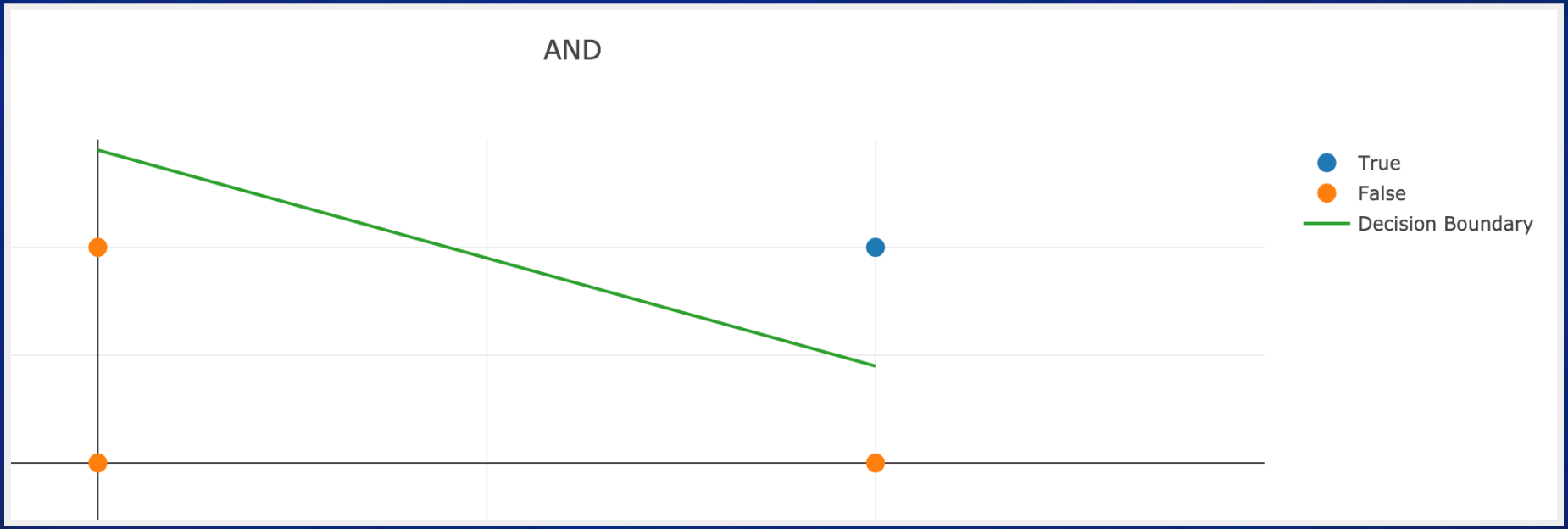


Right, that is surprisingly simple

Code for training is a little bit more complex (but only a little bit)

Visualizing what a neuron can do

- output separates plane into two regions using a line
- such regions are called *linearly separable*
- can emulate most logic functions (NOT, AND, OR, NAND)
- can be trained by adjusting weights of inputs based on error in output



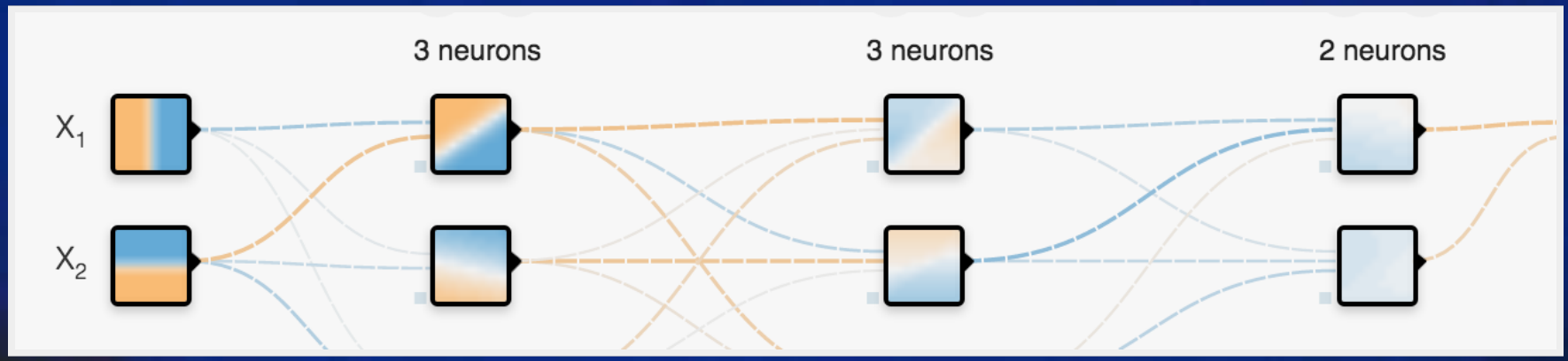
percepton training visualization (initial version provided as a courtesy of Jed Borovik)

A single neuron is not very powerful

but becomes much more powerful when organized in layers

Feedforward Neural Networks

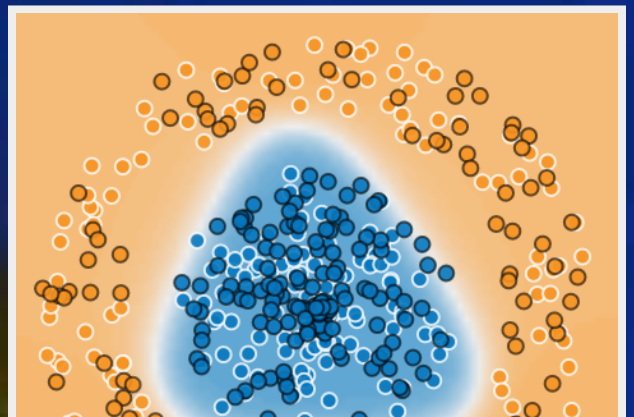
- neurons can be arranged in layers
- data flows forward in one direction, there are no cycles
- first layers takes inputs
- output layer (typically softmax) classifies summed up output
- middle layer(s) called hidden layer(s)
- more than one hidden layer is possible (deep neural network)
- each neuron in each hidden layer gets exactly the same input
- no connection between neurons in same layer
- each neuron in one layer feeds all neurons in the next layer



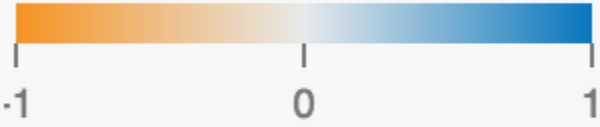
A Classification example

Using the Tensorflow Playground

- result of the neural network is a classification
- each spot in a plane gets a prediction
- either blue or orange
- prediction can be discrete or continuous
- network is trained to give a good prediction

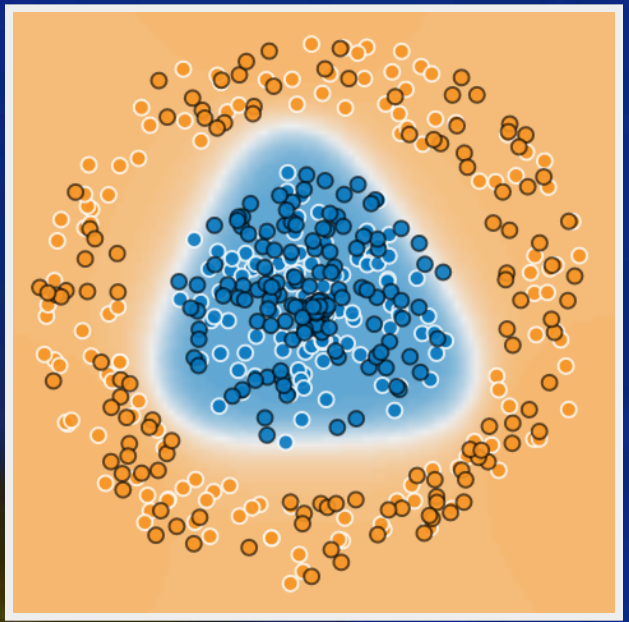


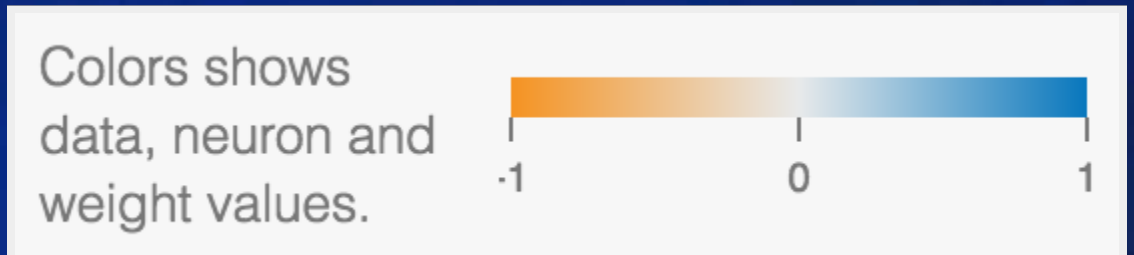
Colors shows data, neuron and weight values.



Training and Loss

- data is separated into training and test data
- in image below: training: white border, test: black border
- training data is used to train neural network
- both test and training data are used to check accuracy of prediction
- loss is to be minimized
- overfitting: training loss low, test loss much higher (to be avoided)





Tensorflow Internals

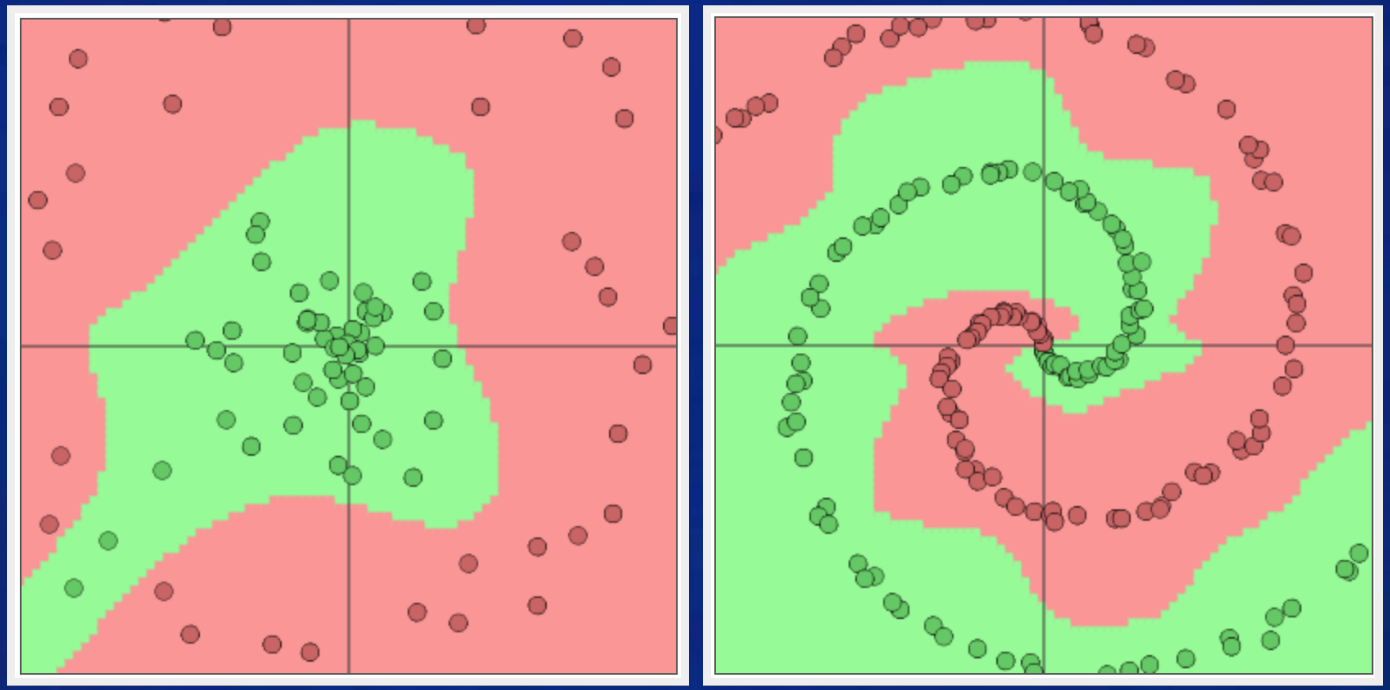
- Written ins TypeScript
- Visualizations dones with D3.js
- Everything implemented from scratch
- No performance optimizations
- Easy yo understand
- NN simulation fairly similar to code from scratch as shown before
- More general for as many dimensions as you want

Now for the JavaScript

ConvNetJS

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

Code can be changed directly in the browser



http://cs.stanford.edu/people/karpathy/convnetjs/demo/classify2d.html

Code for classifier

That's what you can change directly in the browser

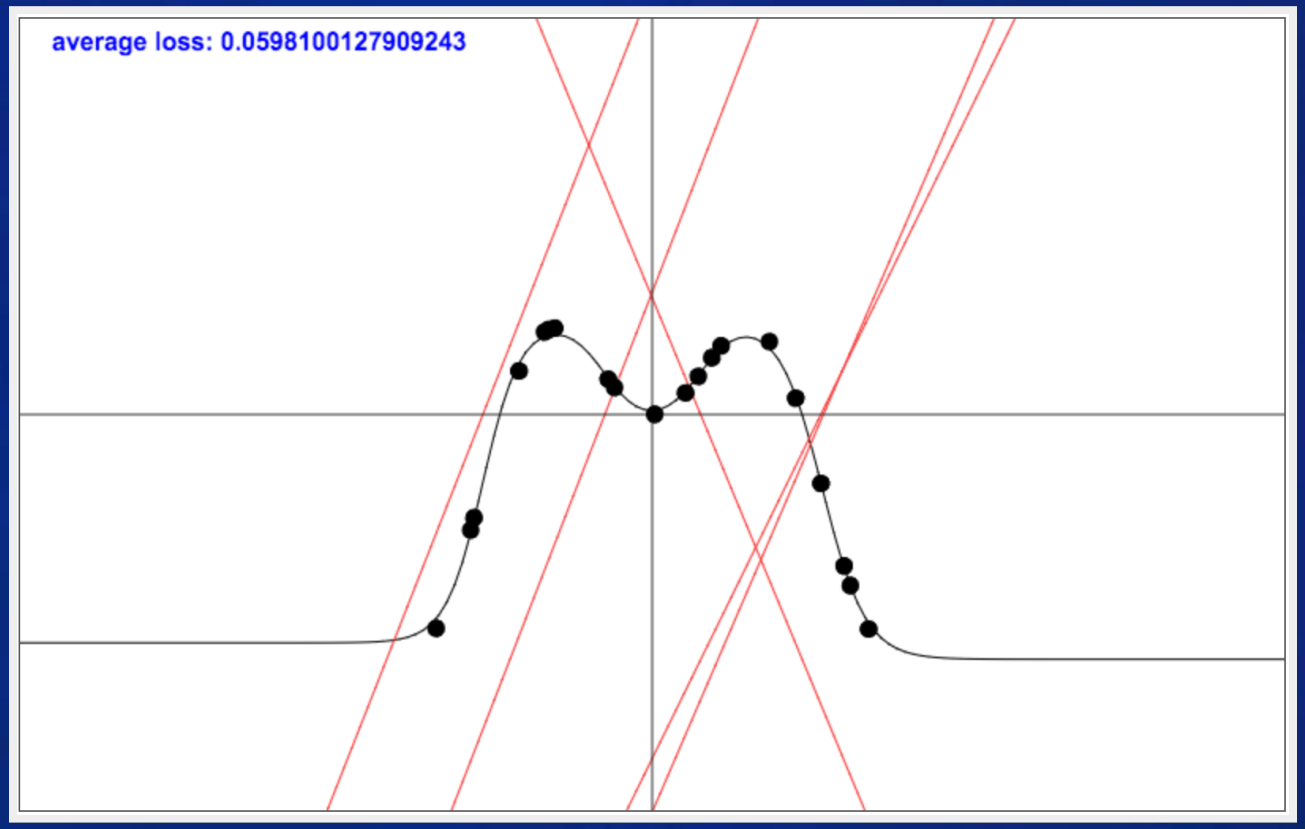
```
layer_defs = [
    {type:'input', out_sx:1, out_sy:1, out_depth:2},
    {type:'fc', num_neurons:6, activation: 'tanh'},
    {type:'fc', num_neurons:2, activation: 'tanh'},
    {type:'softmax', num_classes:2}
];

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.Trainer(net);
```

Regression

Does not classify, but tries to find a coninuous function that goes through all data points



http://cs.stanford.edu/people/karpathy/convnetjs/demo/regression.html

Regression Example - Code

Quick Quiz: How many neurons in hidden layer?

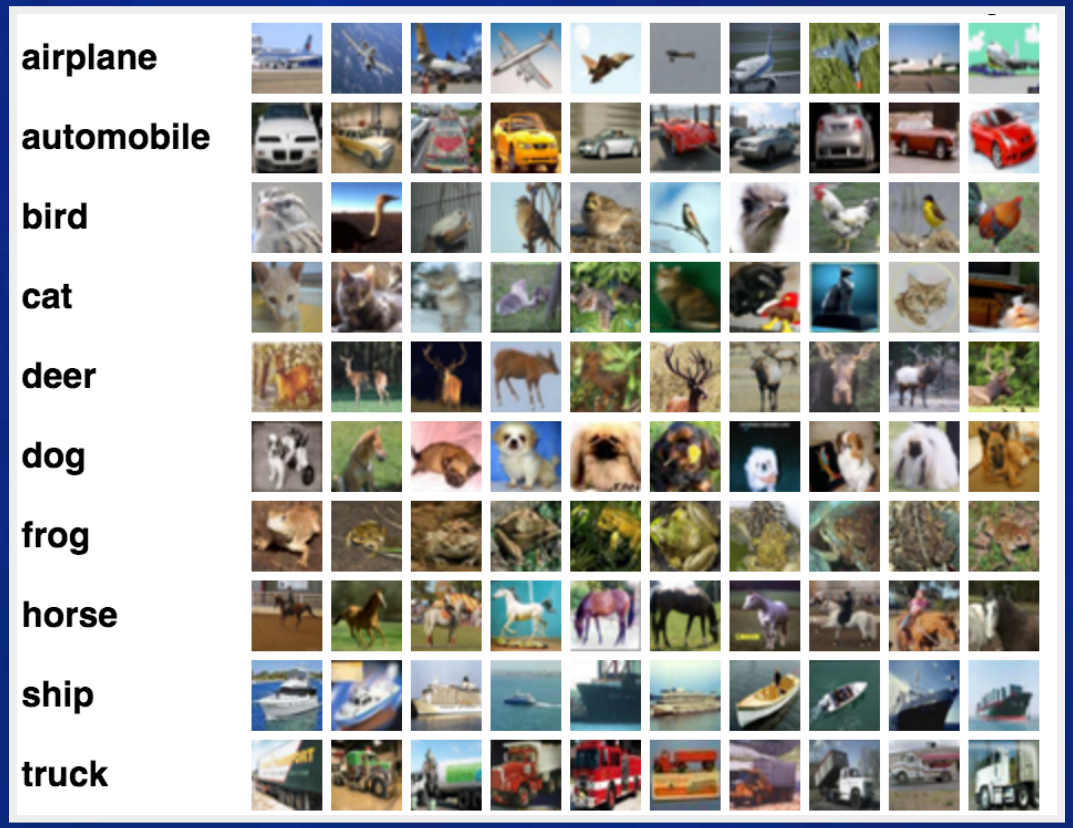
```
layer_defs = [
  {type:'input', out_sx:1, out_sy:1, out_depth:1},
  {type:'fc', num_neurons:5, activation:'sigmoid'},
  {type:'regression', num_neurons:1}];
```

CIFAR-10 with Convolutional Deep Neural Networks

uses convolutional hidden layers for filtering

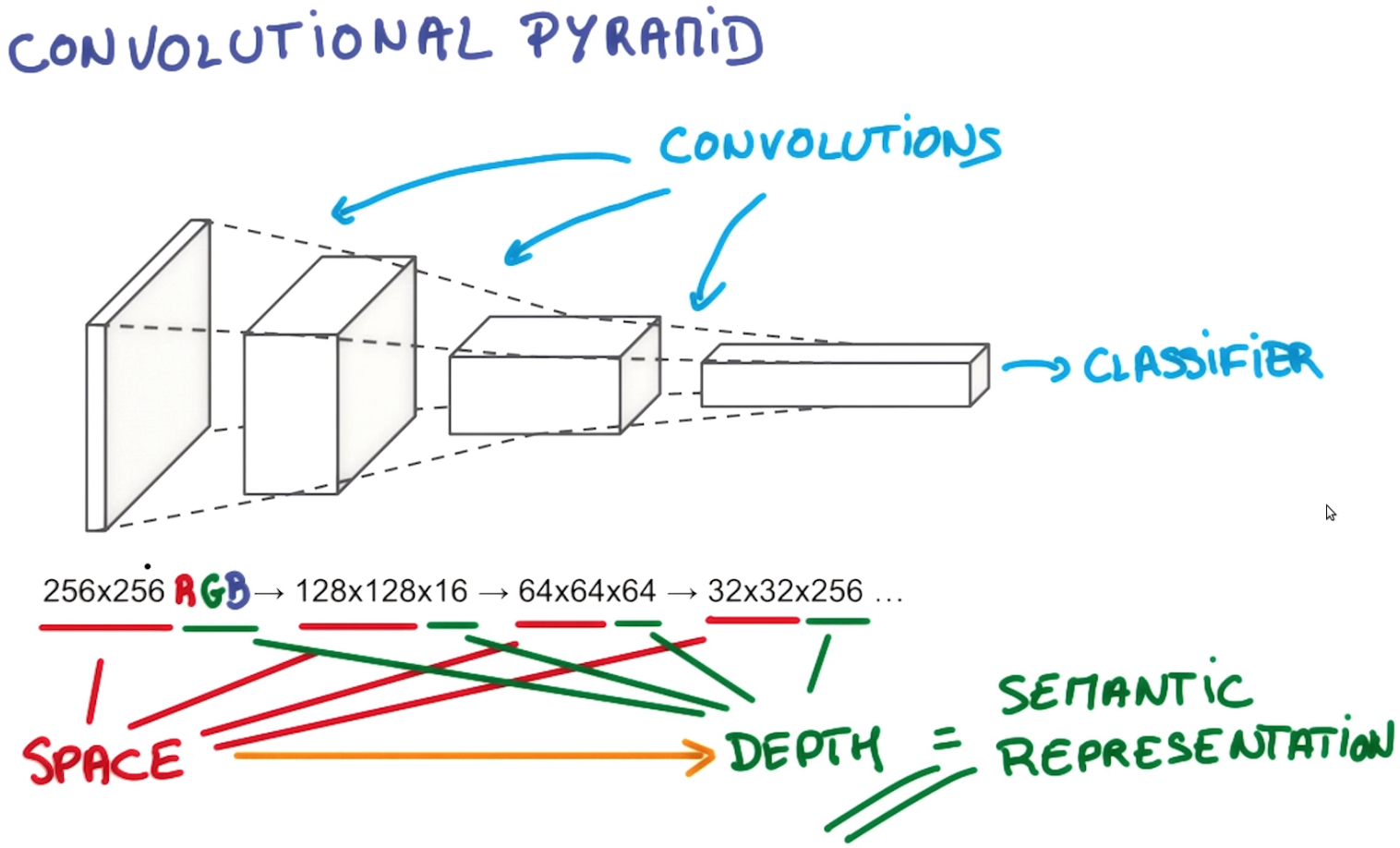
The CIFAR-10 dataset

Tiny images in 10 classes, 6000 per class in training set



http://www.cs.toronto.edu/~kriz/cifar.html

The Convolutional Pyramid



Udacity Course 730, Deep Learning (L3 Convolutional Neural Networks > Convolutional Networks)

CIFAR-10 Example - Code

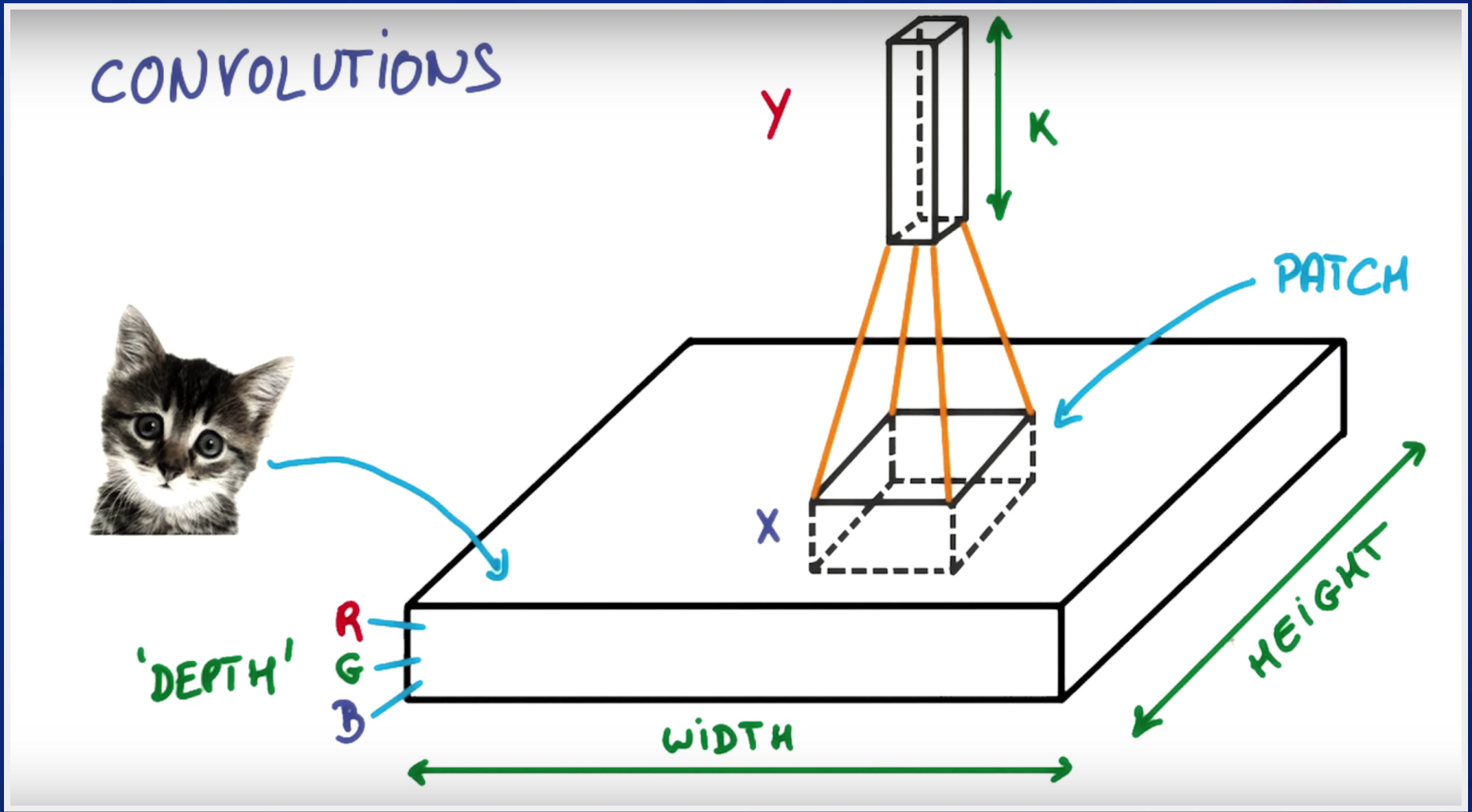
```
layer_defs = [
{type:'input', out_sx:32, out_sy:32, out_depth:3},
{type:'conv', sx:5, filters:16, stride:1, pad:2, activation:'relu'},
{type:'pool', sx:2, stride:2}),
{type:'conv', sx:5, filters:20, stride:1, pad:2, activation:'relu'}.
{type:'pool', sx:2, stride:2},
{type:'conv', sx:5, filters:20, stride:1, pad:2, activation:'relu'},
{type:'pool', sx:2, stride:2},
{type:'softmax', num_classes:10}];
```

Input: 32x32 RGB images

input (32x32x3)
max activation: 0.5, min: -0.5
max gradient: 0.10624, min: -0.12467

Activations:

Convolutions



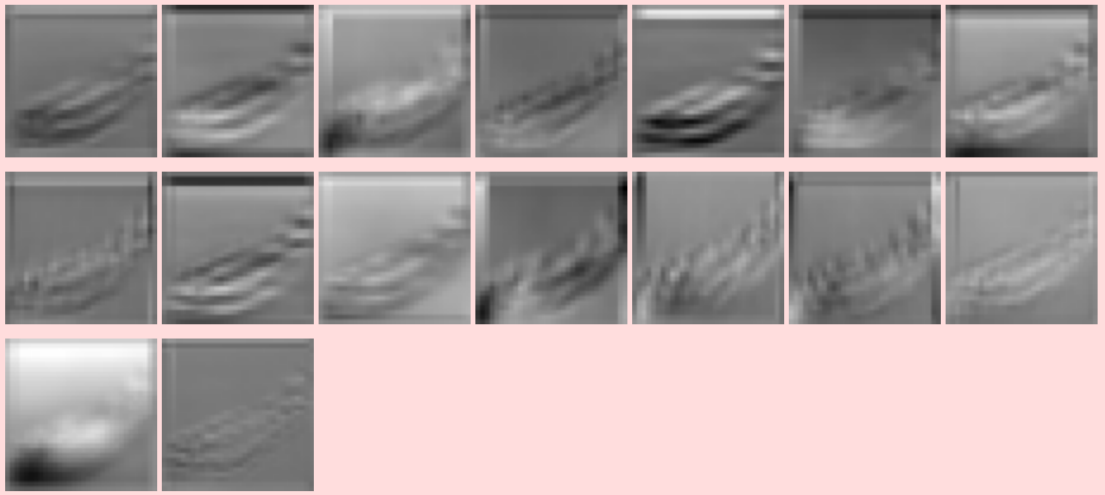
Udacity Course 730, Deep Learning (L3 Convolutional Neural Networks > Convolutional Networks)

Convolution (Filtering) step #1: 16 32x32 images of filtered data

```
// 16 5x5 filters will be convolved with the input
// output again 32x32 as stride (step width) is 1
// 2 pixels on each edge will be padded with 0 matching 5x5 pixel filter
{type:'conv', sx:5, filters:16, stride:1, pad:2, activation:'relu'}
```

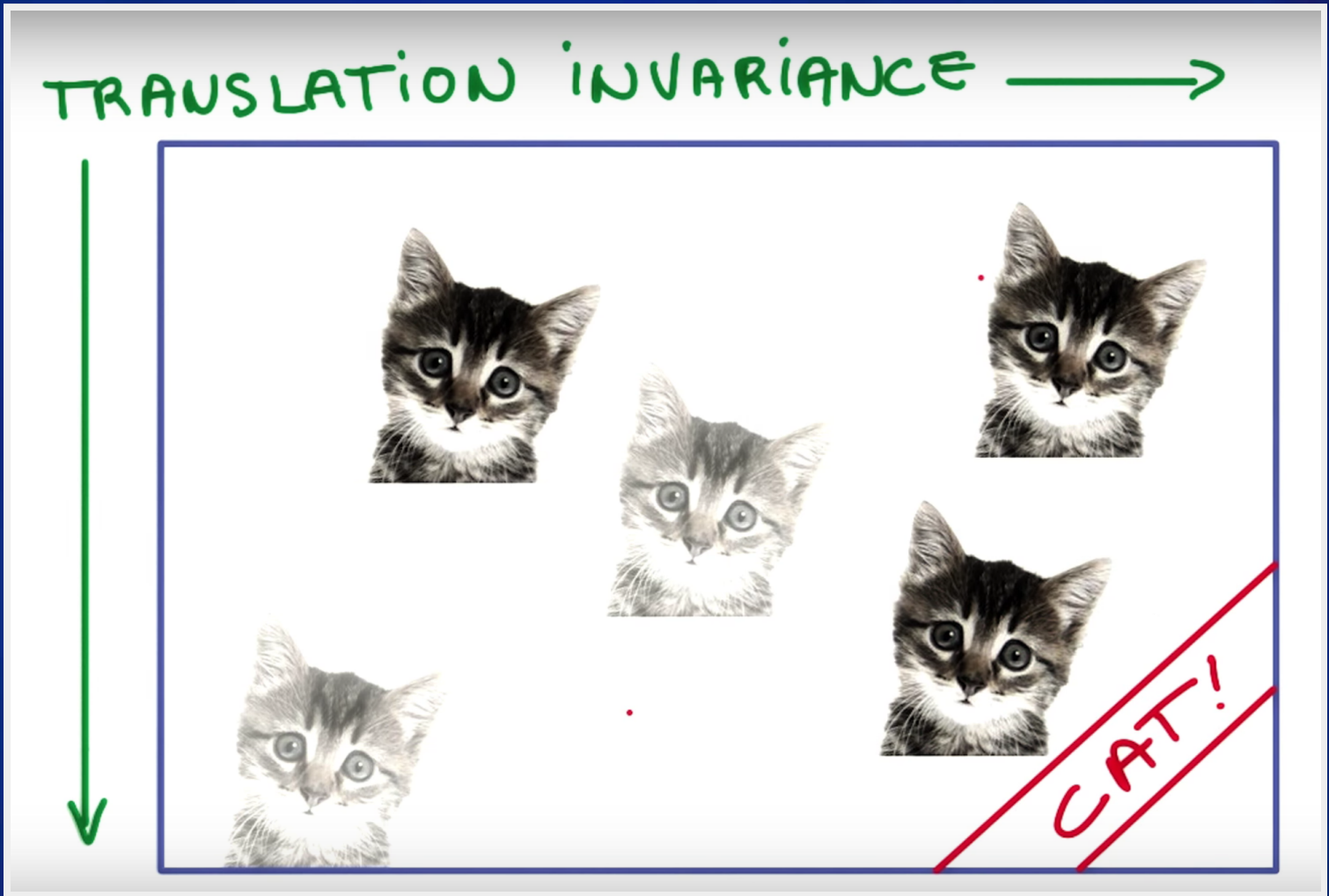
conv (32x32x16)
filter size 5x5x3, stride 1
max activation: 1.33221, min: -1.66454
max gradient: 0.01757, min: -0.01873
parameters: 16x5x5x3+16 = 1216

Activations:



convolutional layers remove noise, add semantics

Translation Invariance



Udacity Course 730, Deep Learning (L3 Convolutional Neural Networks > Convolutional Networks)

Pooling

```
// perform max pooling in 2x2 non-overlapping neighborhoods
// output will be 16x16 as stride (step width) is 2
{type:'pool', sx:2, stride:2}),
```

pool (16x16x16)
pooling size 2x2, stride 2
max activation: 1.45481, min: 0
max gradient: 0.0798, min: -0.11926

Activations:

pooling layers in between provide translation invariance

Softmax

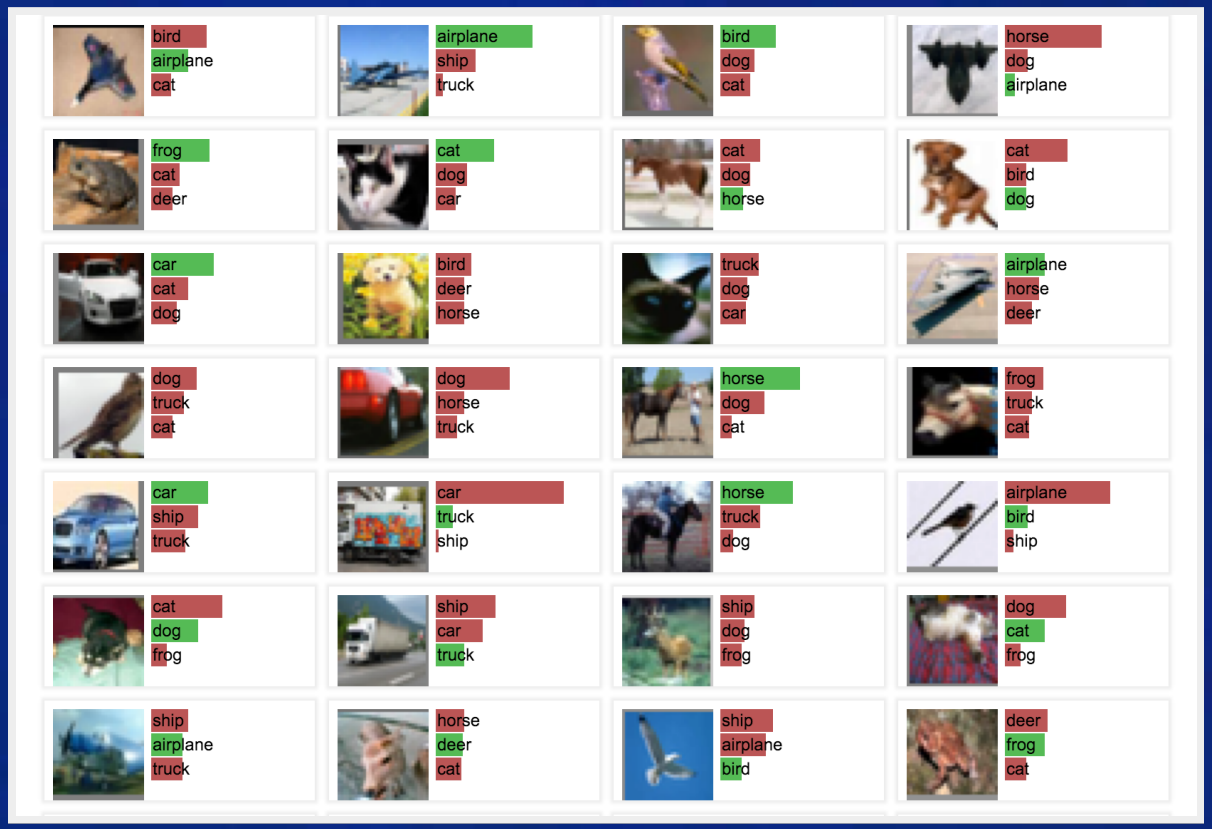
```
{type:'softmax', num_classes:10}
```

softmax (1x1x10)
max activation: 0.60745, min: 0.00011
max gradient: 0, min: 0

Activations:

assigns a probability to each of the 10 categories

Sample classification results

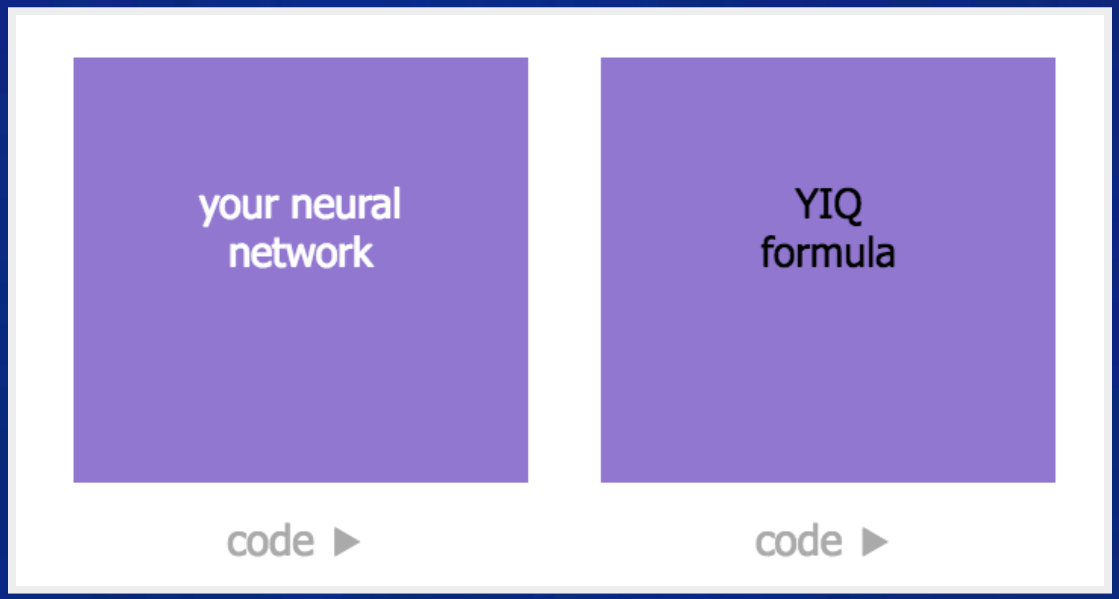


http://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html

Other JavaScript libraries that run in the Browser

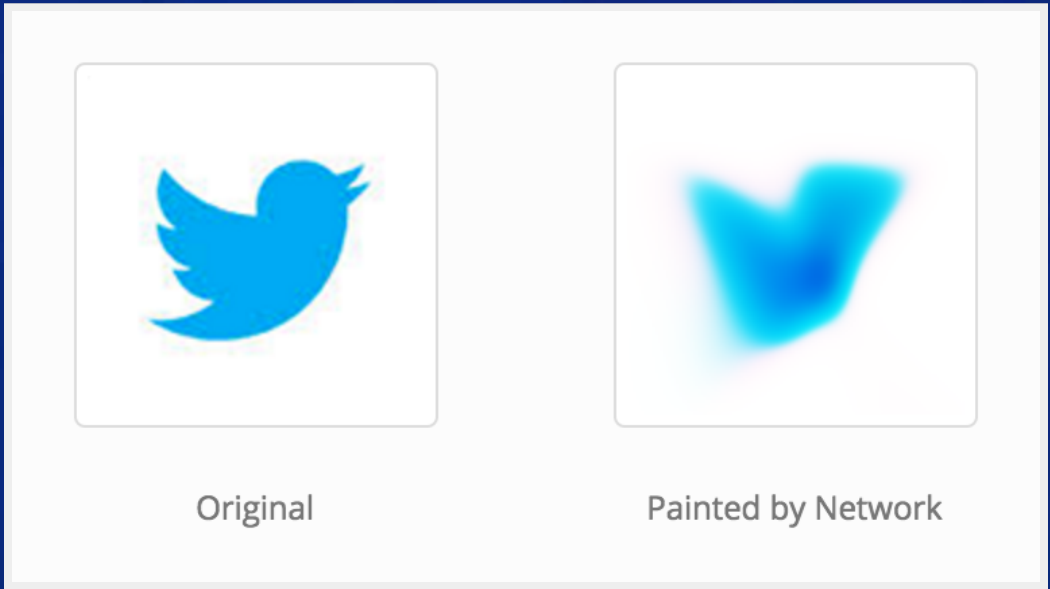
Brain.js

- very simple lib
- used to be unmaintained, but taken over by new maintainers



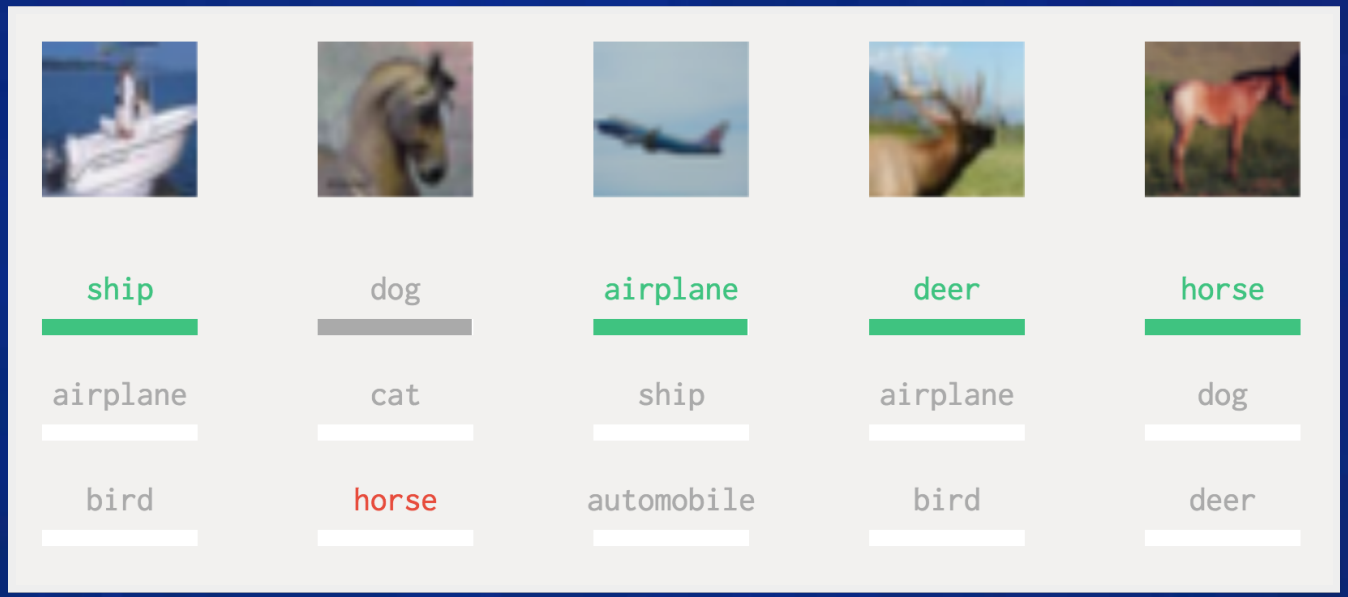
Synaptic.js

- small general lib for Neural Network
- nice and simple API
- good documentation
- actively maintained



neocortex.js

- Runs pre-trained deep neural networks (Keras models)
- Some really nice examples including Python Notebooks
- Clear code written in ES6



Wrap-Up

- Deep Neural Networks are a game changer in Machine Learning and AI
- enabled by large Data Sets, Computing Power, and GPUs
- the browser makes Deep Learning even more accessible
- all the fancy deep learning stuff works in the browser
- especially good for learning and education
- direct visualization and interactivity
- Tensorflow Playground makes this especially easy to try out
- ConvnetJS for experiments with JavaScript in the browser

Thank you!

Questions / Discussion

Oliver Zeigermann / @DJCordhose

http://bit.ly/ml-data2day