GENERAL ASSEMBLY

NEURAL NETWORKS

Tan Kwan Chong

Chief Data Scientist, Booz Allen Hamilton

LEARNING OBJECTIVES

- Understand various types of neural networks
- Applications of neural networks
- Apply a neural network model for regression
- Apply a neural network model for classification

OPENING

ARTIFICIAL NEURAL NETWORKS

OPENING

- Neural networks were first studied in the 1940s (!) as a model of biological neural networks
- Many advances since then have improved the ability to train and apply neural networks
- Good for both classification and regression but difficult to interpret model behaviors
- Deep learning in the past few years has been highly successful for otherwise difficult problems

OPENING

- Today we will focus on types of neural networks and their applications, and skip some of the more technical details

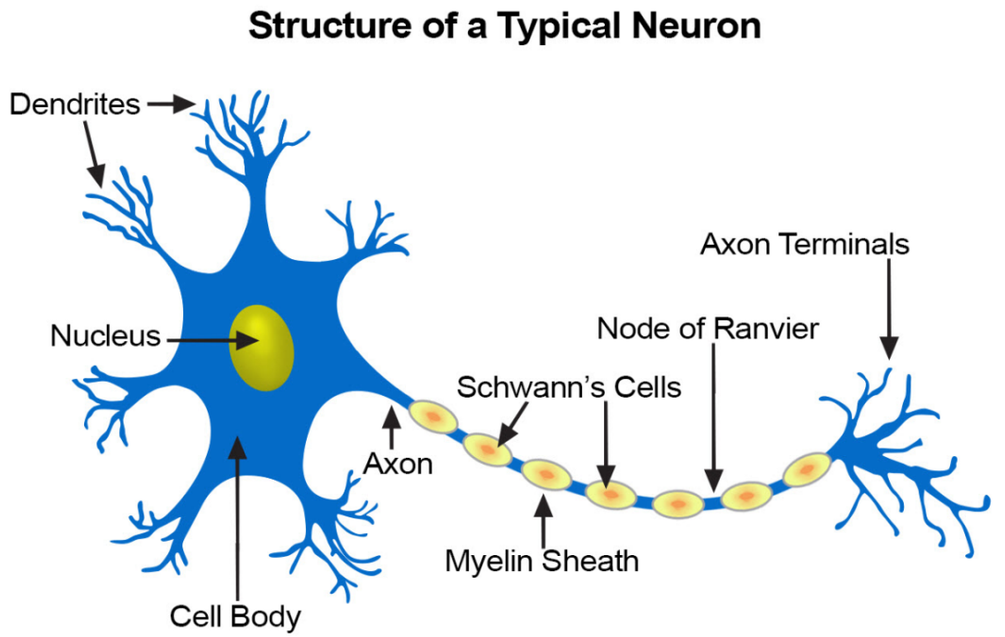
- Specifically we'll skip training neural networks -- there are many methods in various situations and the details can be tedious (but not particularly difficult)

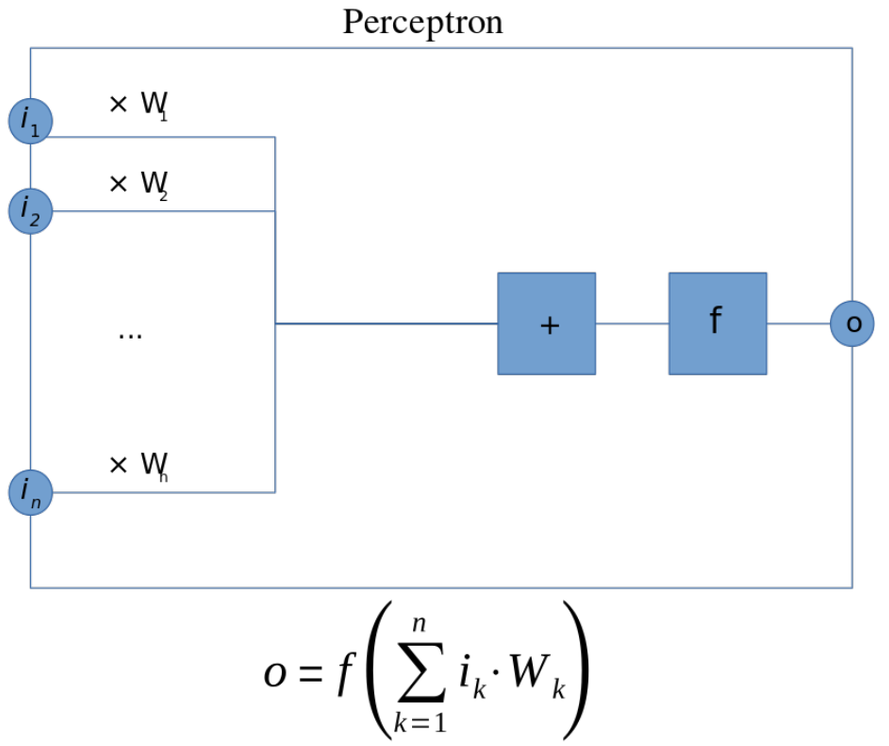
- Methods include backpropagation, gradient descent, and Hessian-free learning

PERCEPTRON

PERCEPTRON

- Perceptrons are the simplest example of a neural network
- The idea is to emulate a single neuron



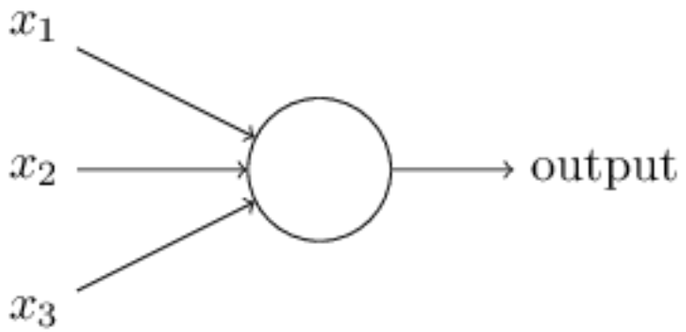


$$o = f\left(\sum_{k=1}^{n} i_k \cdot W_k\right)$$

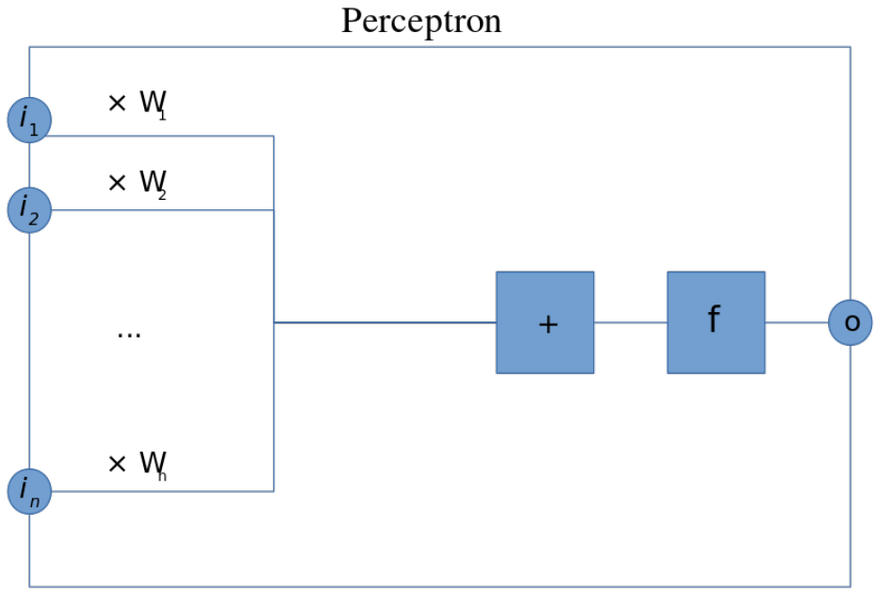
https://en.wikipedia.org/wiki/Perceptron

PERCEPTRON

- Given n inputs and an activation or link function f
- A perceptron takes several binary inputs and produces a single binary output



$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{threshold} \end{cases}$$

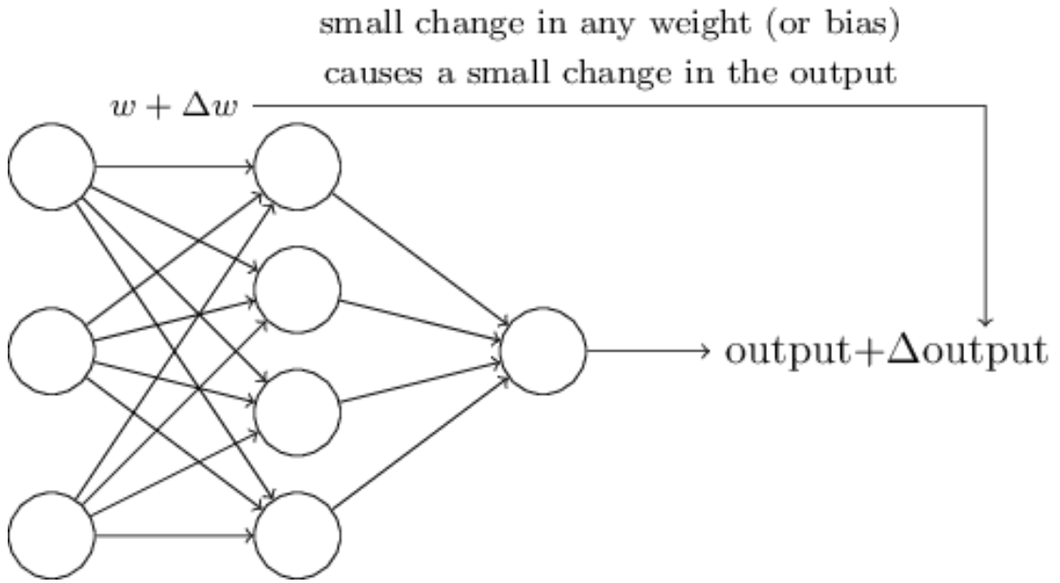


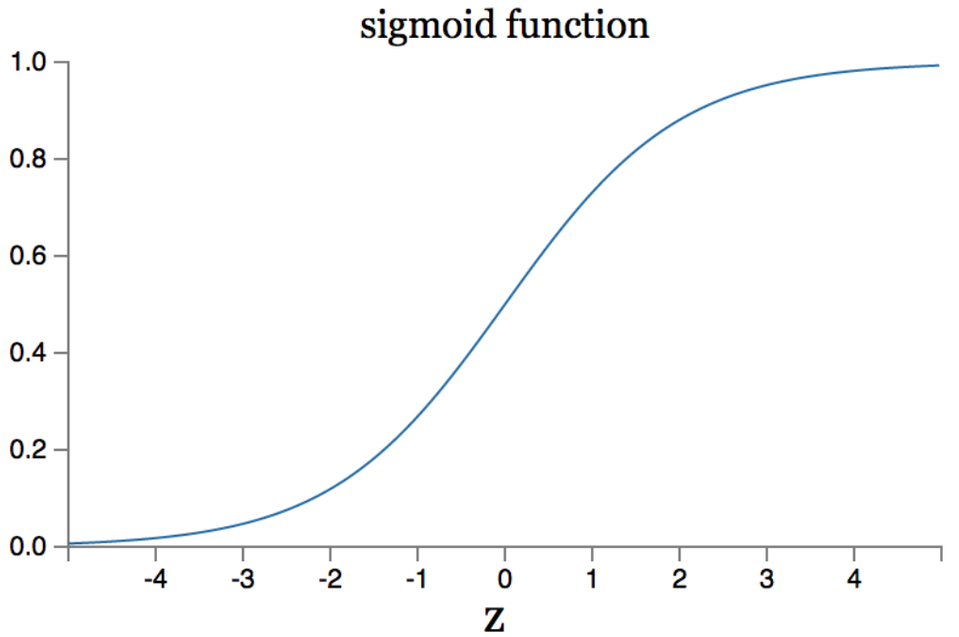
$$o = f\left(\sum_{k=1}^{n} i_k \cdot W_k\right)$$

https://en.wikipedia.org/wiki/Activation_function

PERCEPTRON

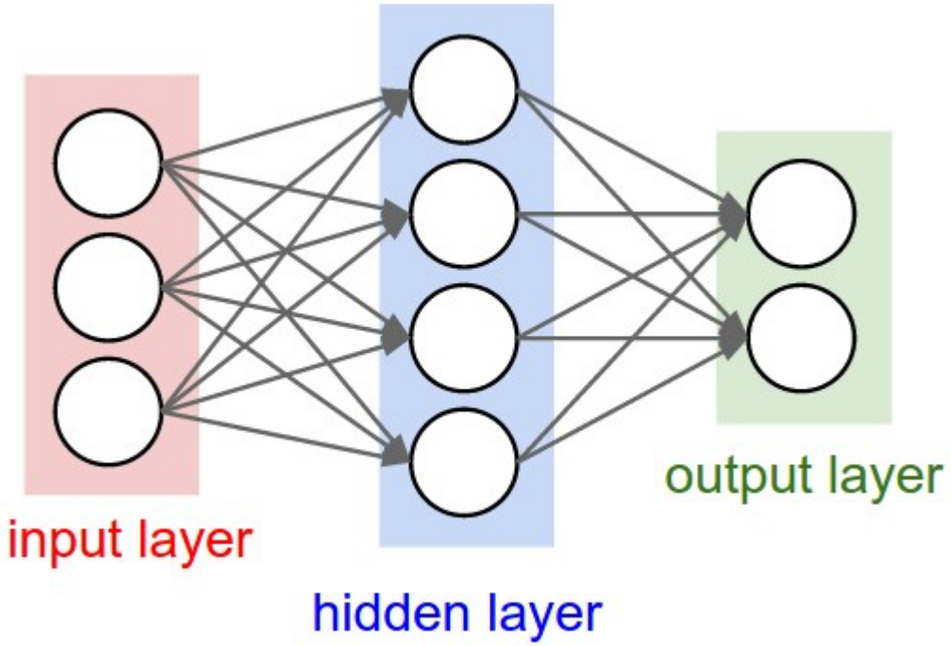
- Common activation functions are linear, logistic, tanh, and softmax
- We'll see shortly that some are better for classification, some for regression
- Perceptrons can be combined into multilayer perceptrons or feed-forward network





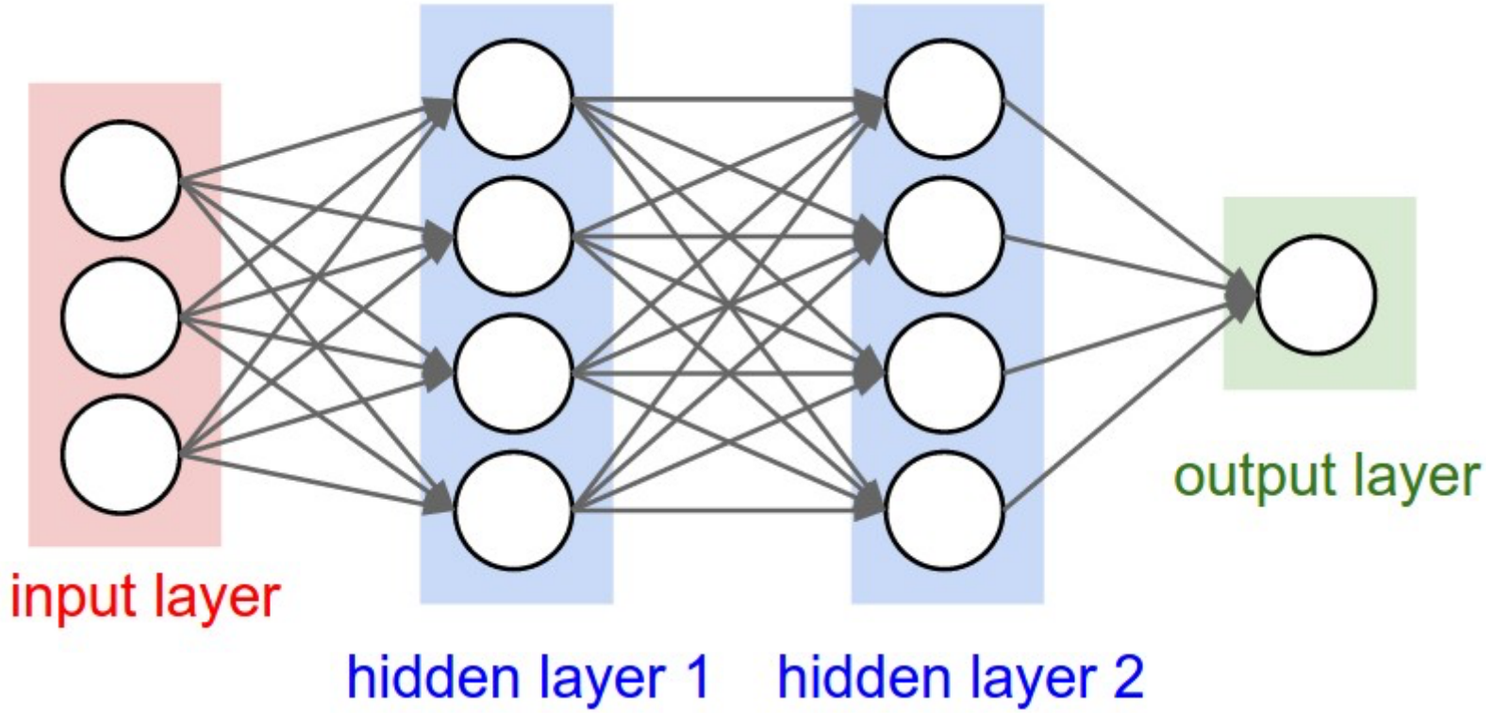
FEED FORWARD NN

- Source



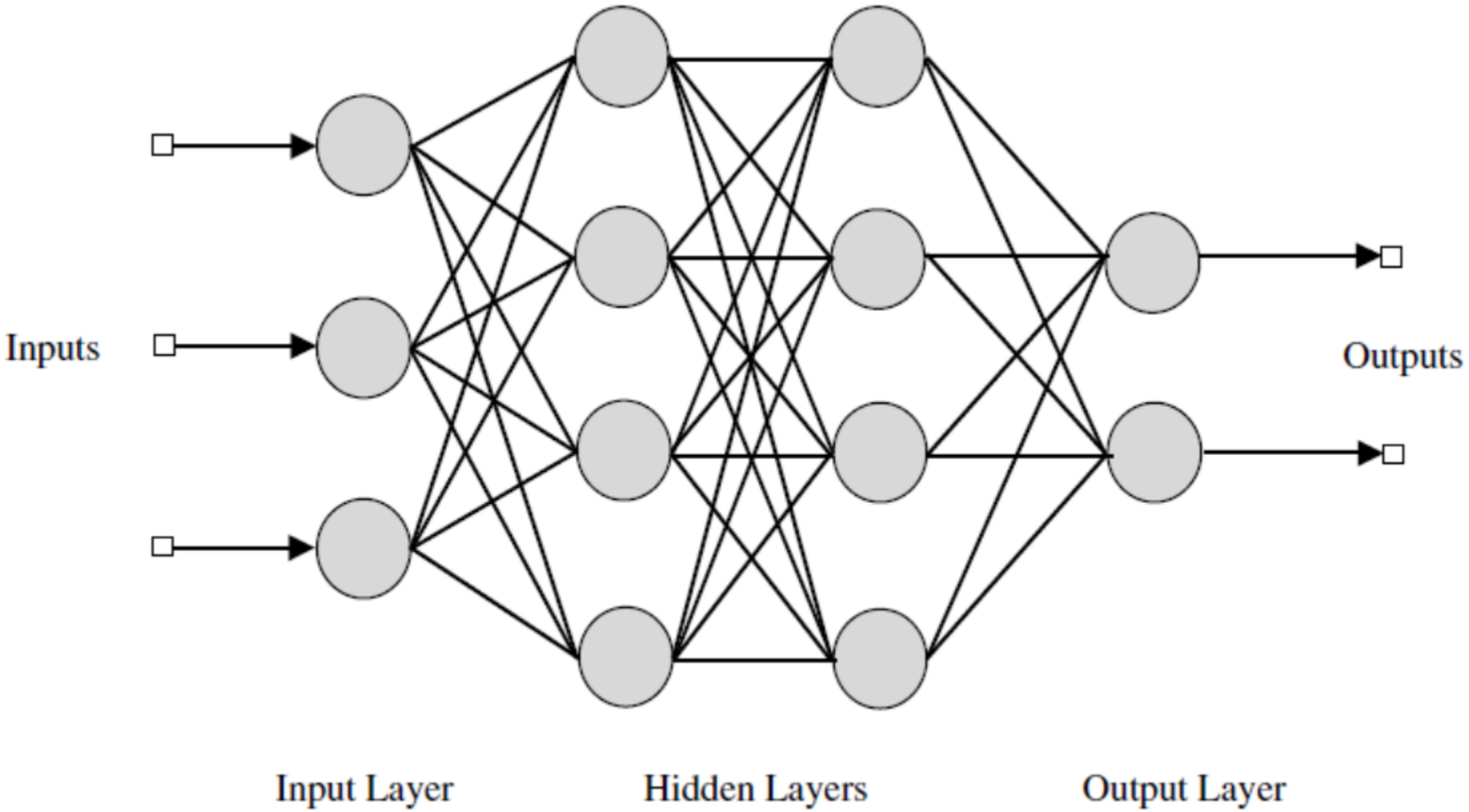
FEED FORWARD NN

- Source



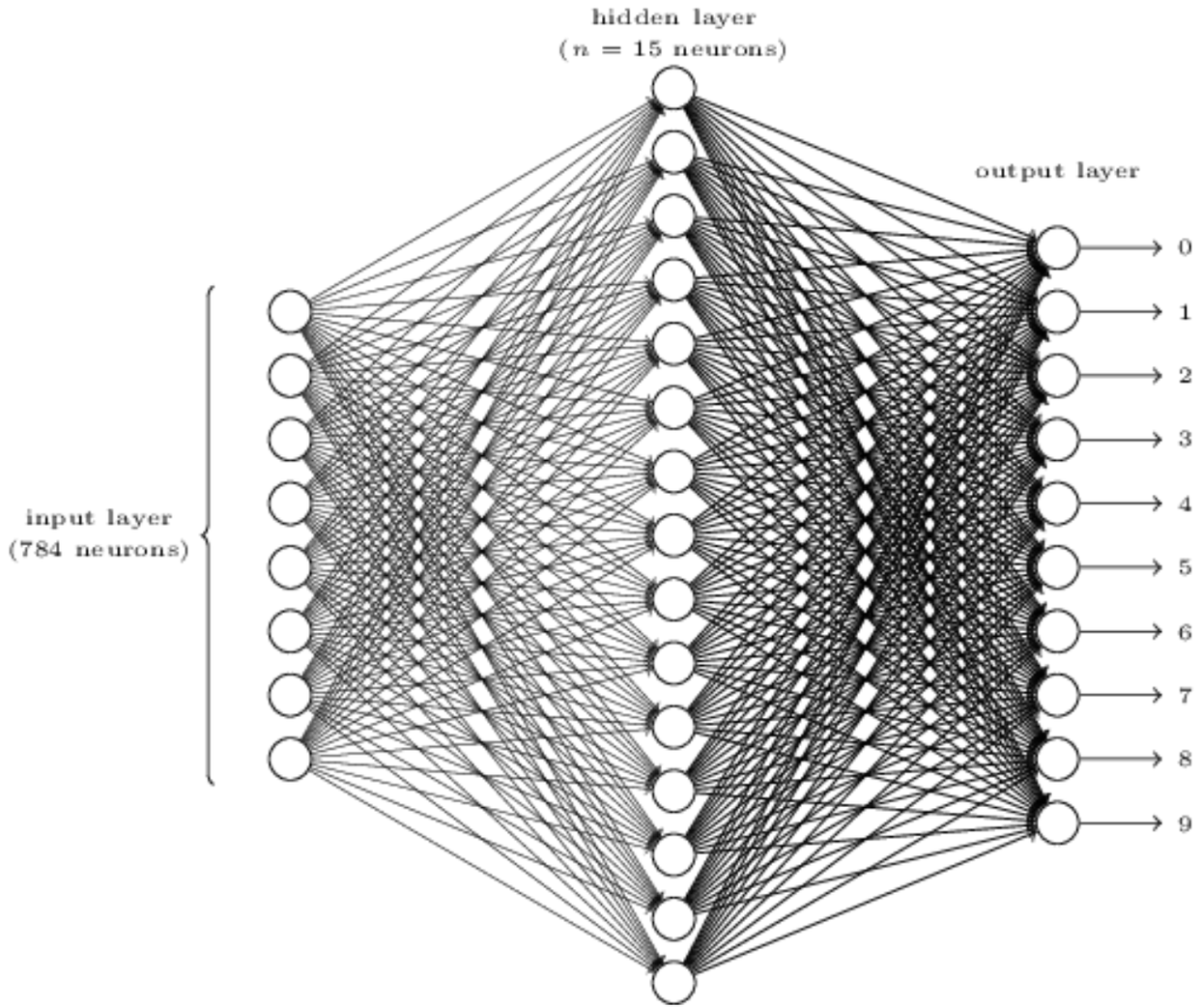
FEED FORWARD NN

- Source



FEED FORWARD NN

- Source



FEED FORWARD NN

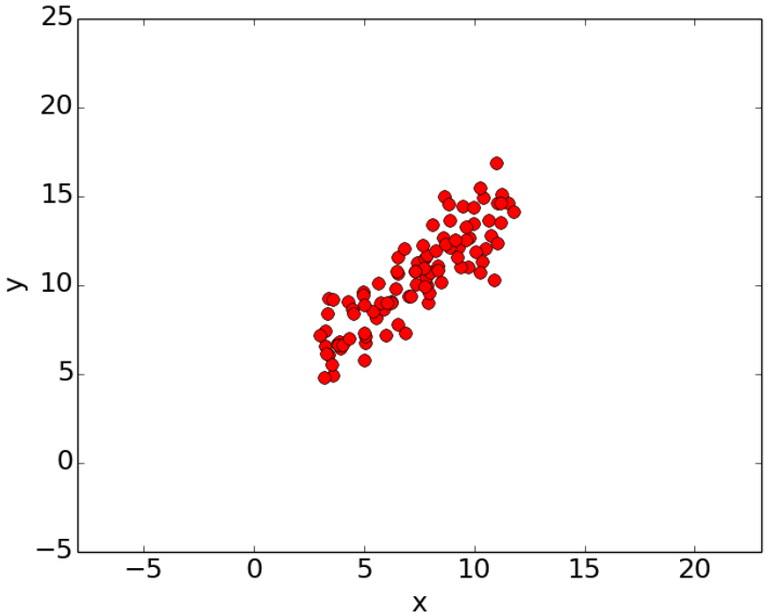
- Typically we use
  - Tanh or logistic layers for input
  - Linear layers for regression output
  - Logistic or Tanh for binary output
  - Softmax for n-class output (yields probabilities)

GUIDED PRACTICE

TRAINING

GRADIENT DESCENT

- Goal of Linear Regression is to fit a line to a set of points
- Standard approach is to define an error function (cost function) that measures how good a given line is

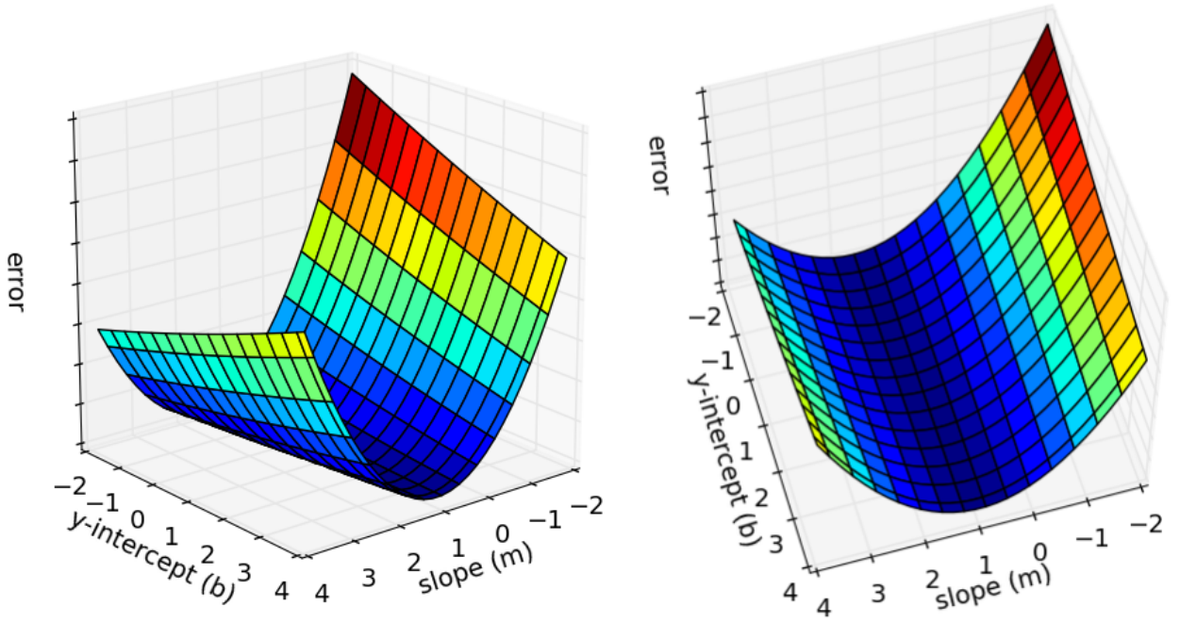


$$\text{Error}_{(\text{m,b})} = \frac{1}{N}\sum_{i=1}^{N}(y_i - (mx_i + b))^2$$

https://spin.atomicobject.com/2014/06/24/gradient-descent-linear-regression/

GRADIENT DESCENT

- Lines that fit the data better will result in lower error values
- By minimizing the error function we will get the best line for the data
- When we run gradient descent search, we will start from some location on this surface and move downhill to find the point with the lowest error

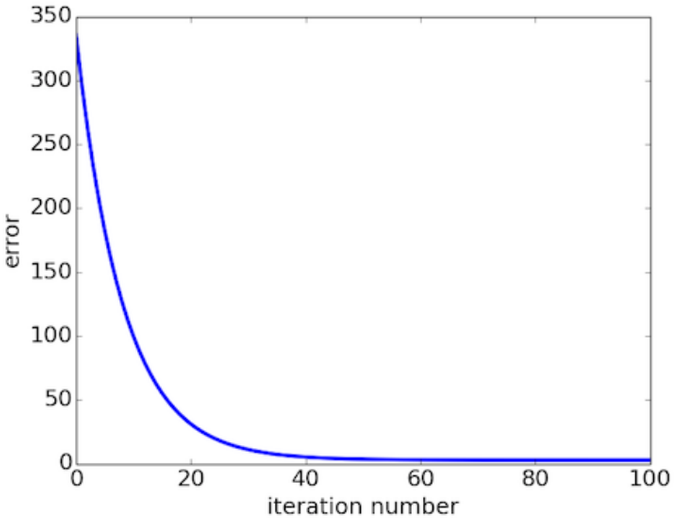


GRADIENT DESCENT

- To run gradient descent on the error function, we need to compute its gradient by differentiating the error function
- Each iteration of the algorithm updates m and b to a line that yields slightly lower error than the previous iteration
- The learning rate controls how large a step to take each iteration
- Iteration stops when error changes are smaller than threshold

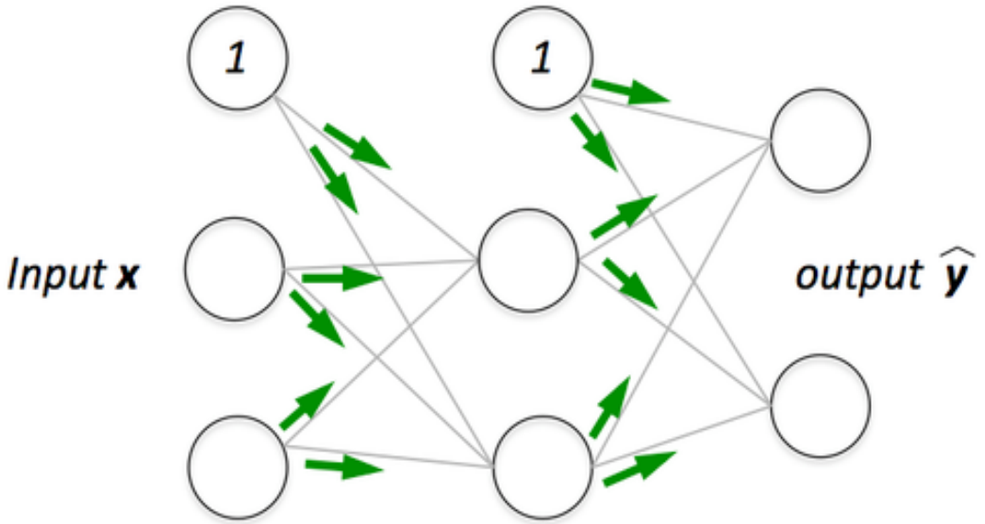
$$\frac{\partial}{\partial \mathrm{m}} = \frac{2}{N}\sum_{i=1}^{N} - x_i(y_i - (mx_i + b))$$

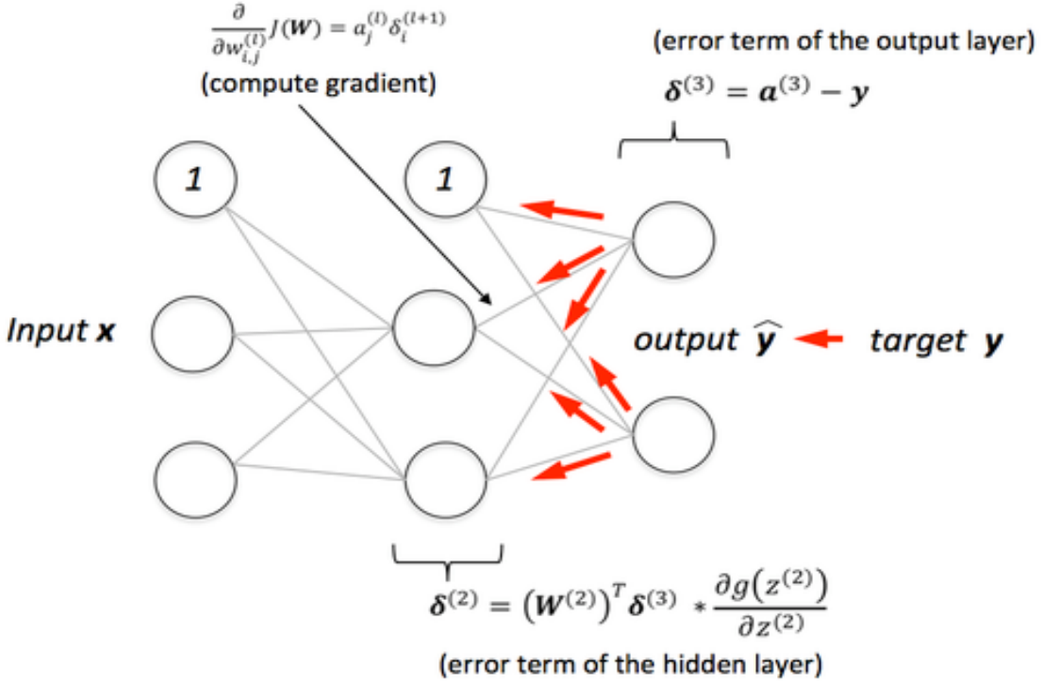
$$\frac{\partial}{\partial \mathrm{b}} = \frac{2}{N}\sum_{i=1}^{N} -(y_i - (mx_i + b))$$



TRAINING

- Feed forward neural networks can be trained with backpropagation
- Source





https://en.wikipedia.org/wiki/Backpropagation

http://neuralnetworksanddeeplearning.com/chap2.html

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

TRAINING

- Key Parameters
  - Learning Rate (gradient descent for training)
  - Epochs: number of backpropagation passes (over entire dataset)
  - Batch size: how many training points used at a time to update weights

- Model others behaves as usual with
  - model.predict
  - model.predict_classes

TRAINING

- Tips
  - If the error jumps around per epoch, decrease the learning rate
  - Taking too long to train: use higher learning rate or batch_size
  - High error after convergence?
    - More hidden layers / neurons
    - Normalize data or use PCA

UNIVERSAL APPROXIMATION

- One major reason that neural networks are useful is the Universal Approximation Theorem

- The result basically says that many real vector-valued functions can be approximated arbitrarily well with *some* feed-forward neural network

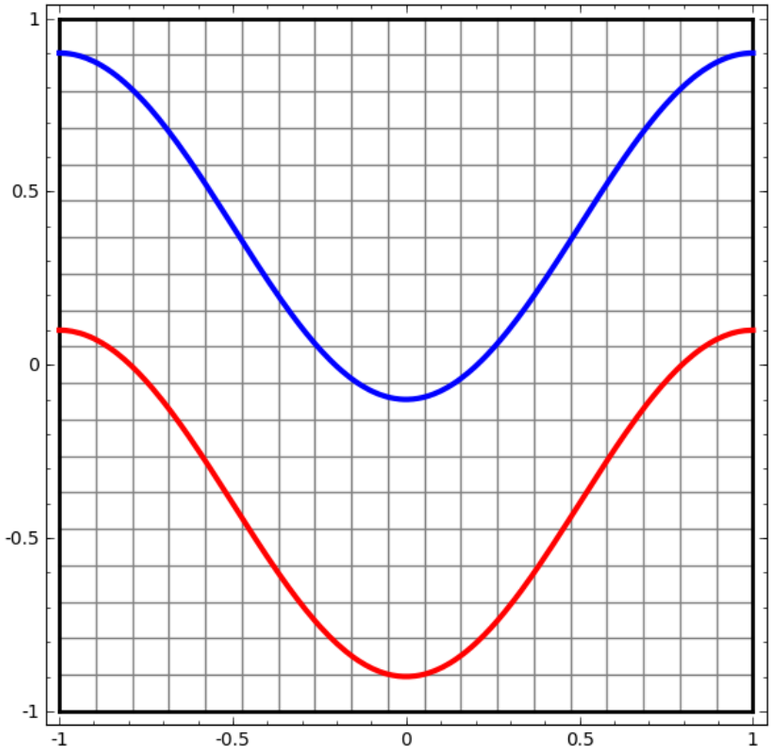
- This is why neural networks are useful for regression -- given enough data and the right network structure they can fit many common data sets

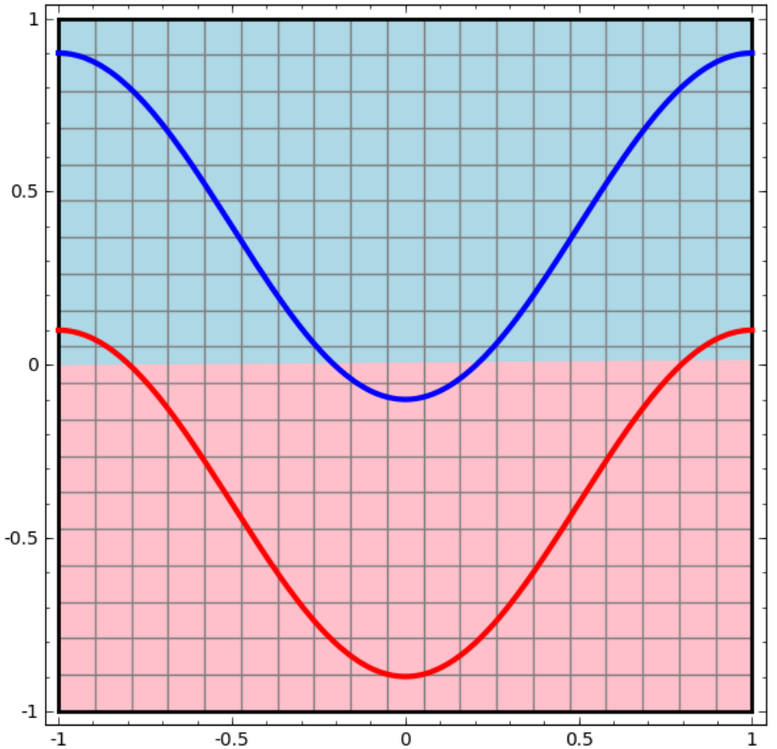
https://en.wikipedia.org/wiki/Universal_approximation_theorem

CLASSIFICATION WITH NEURAL NETWORKS

CLASSIFICATION

- Neural Networks are also extremely useful for classification (source)
- No hidden layers:

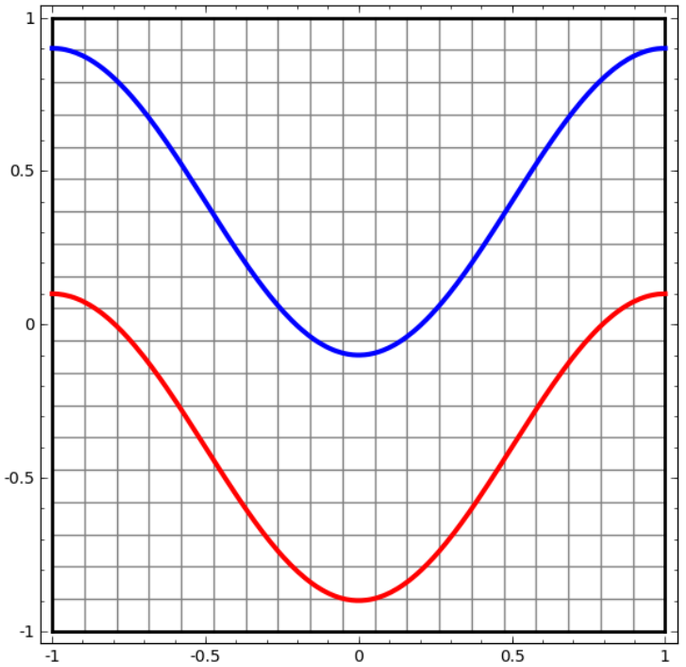


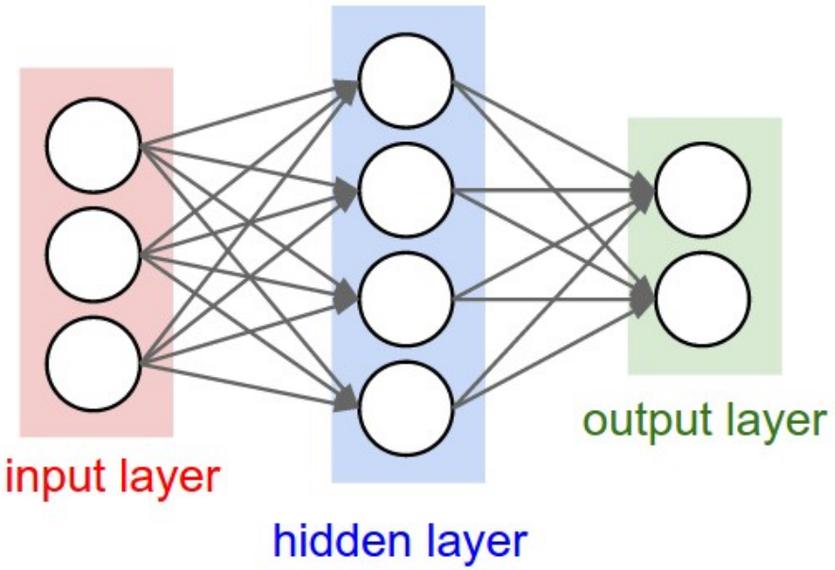


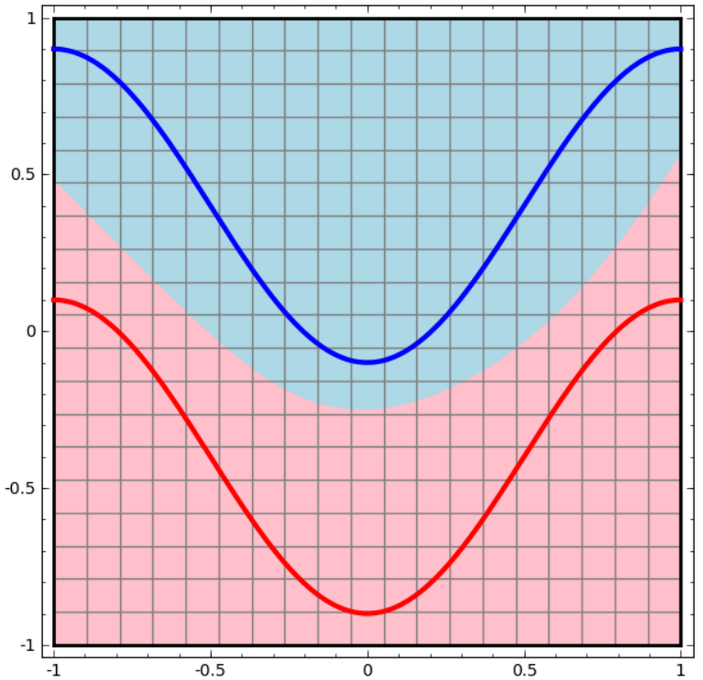
http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/

CLASSIFICATION

- Neural Networks are also extremely useful for classification (source)
- One hidden layer:

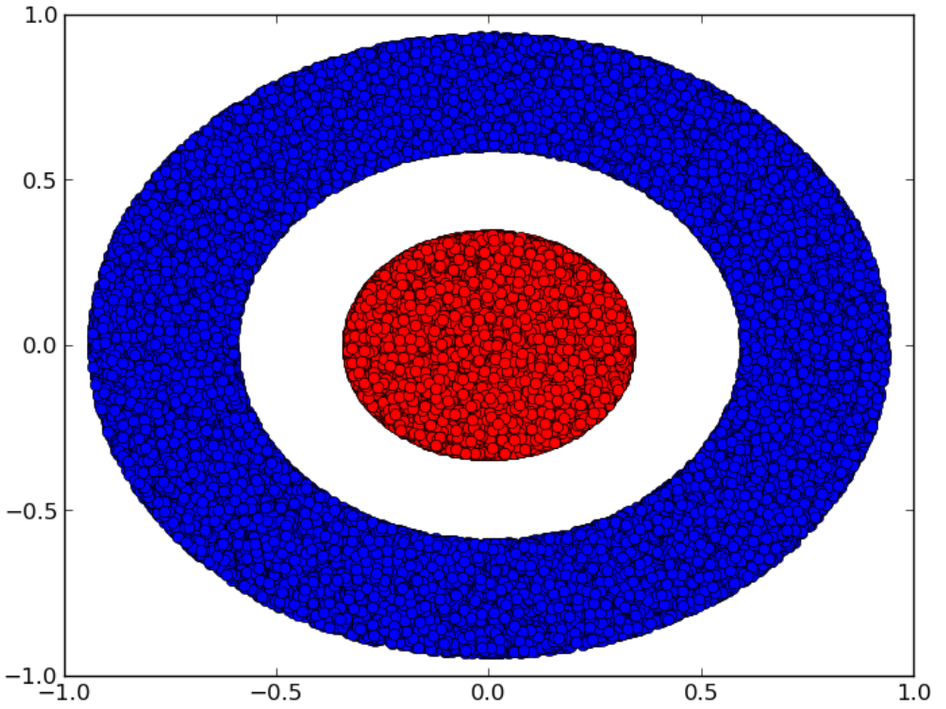


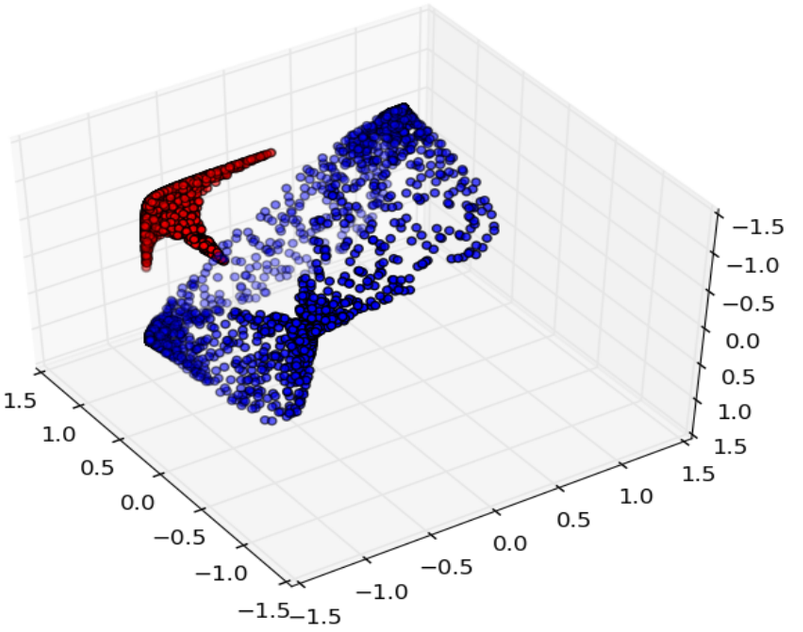




CLASSIFICATION

- Neural Networks are also extremely useful for classification (source)





CLASSIFICATION

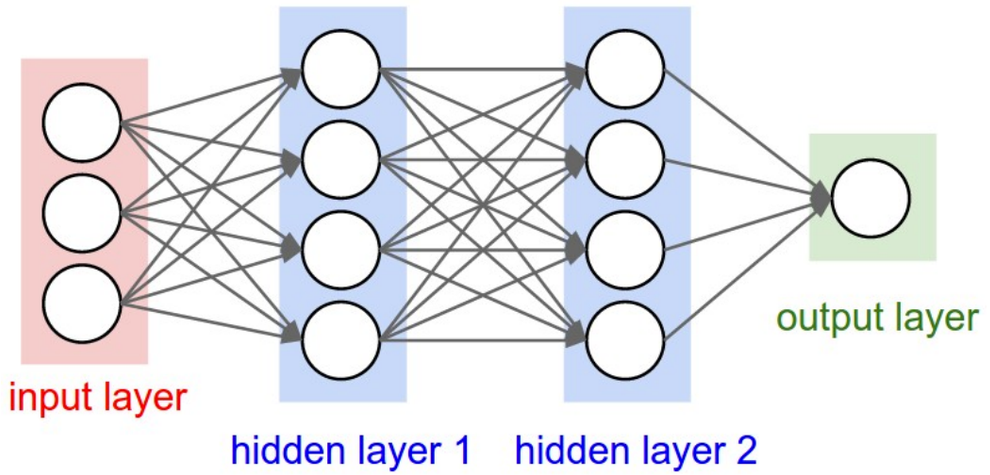
- The neural network transforms the data topologically (no tears or breaks) and then separates the data with a hyperplane

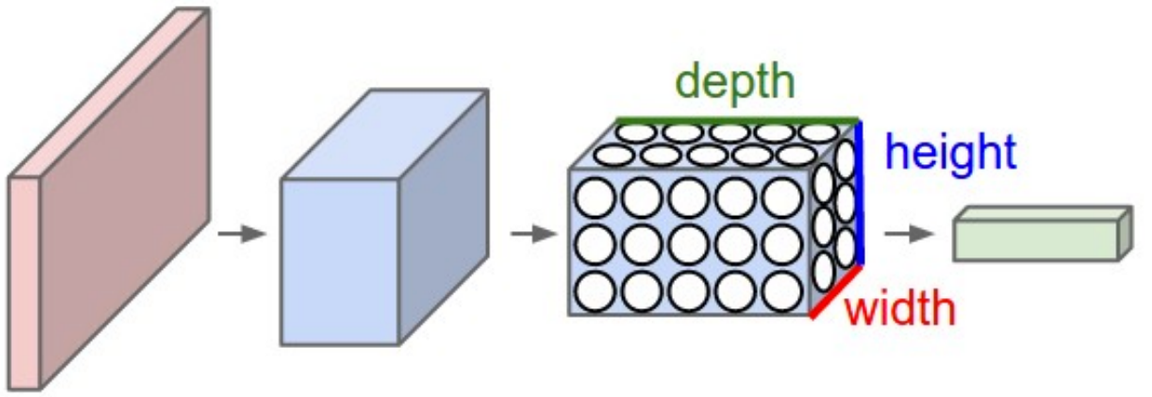
- NNs are capable of handling difficult data sets, including:
  - Image processing: recognizing hand-written characters
  - Image compression
  - Financial forecasting
  - Many others

CONVOLUTIONAL NEURAL NETWORKS

CONVOLUTIONAL NEURAL NETWORKS

- Layers of Convolutional Neural Networks have neurons arranged in 3 dimensions: width, height, and depth
- Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations

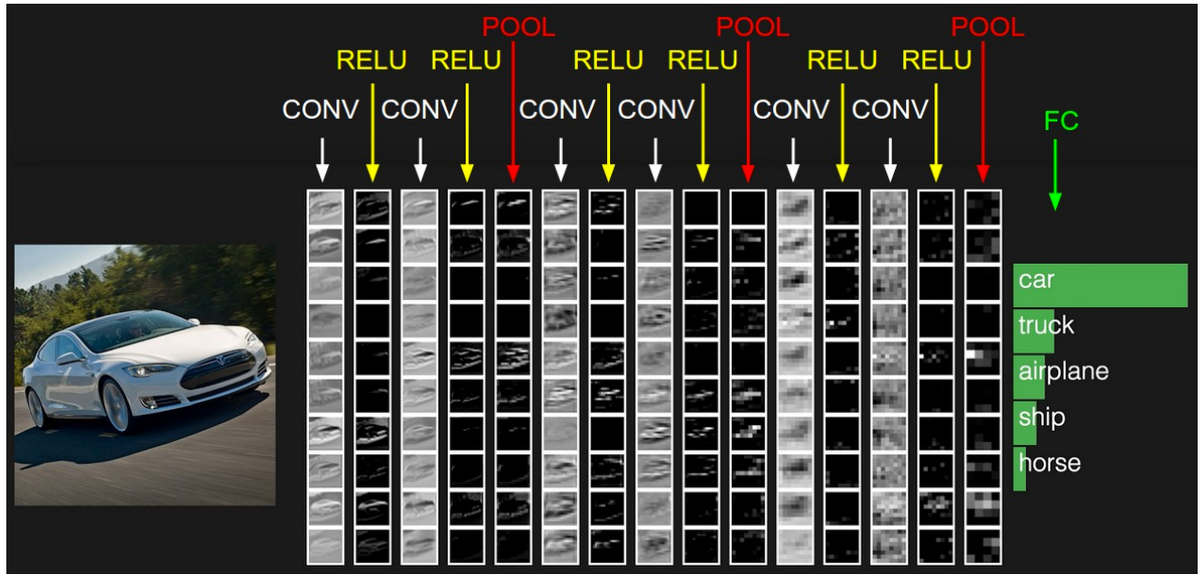




http://cs231n.github.io/convolutional-networks/

CONVOLUTIONAL NEURAL NETWORKS

- ConvNet architecture represents a list of layers that transform the image volume into an output volume (contains the class scores)
- There are a few distinct types of layers i.e. CONV/RELU/POOL/FC
- Each layer accepts an input 3D volume and transforms it to an output 3D volume through a differentiable function

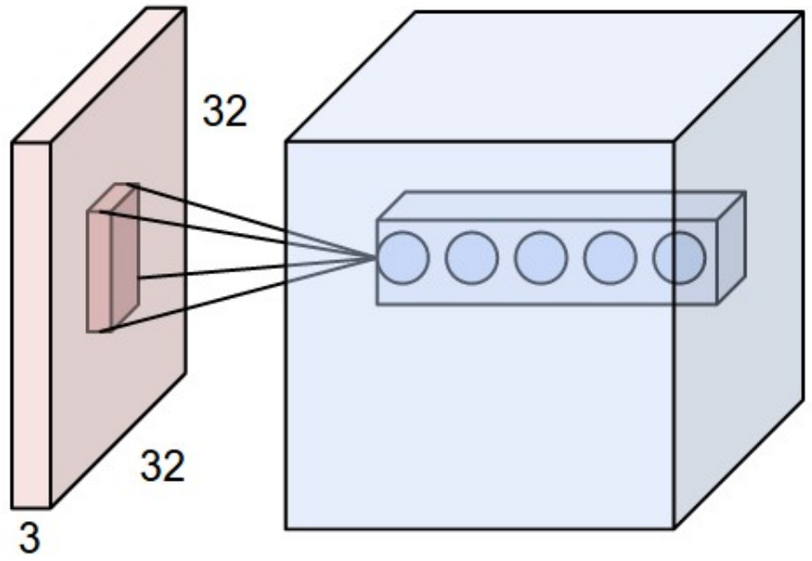


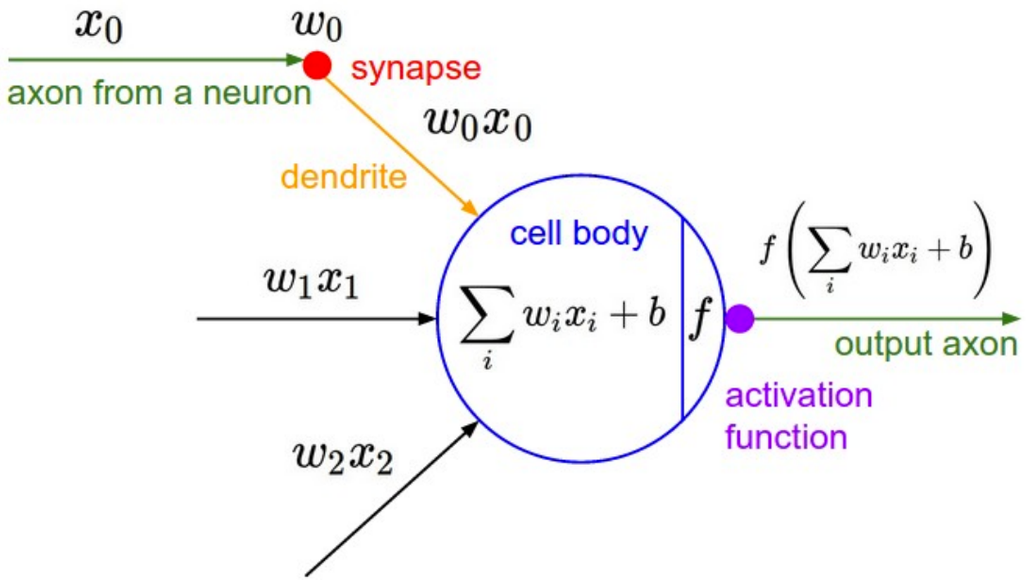
http://cs231n.github.io/convolutional-networks/

http://cs231n.stanford.edu/

CONVOLUTIONAL NEURAL NETWORKS

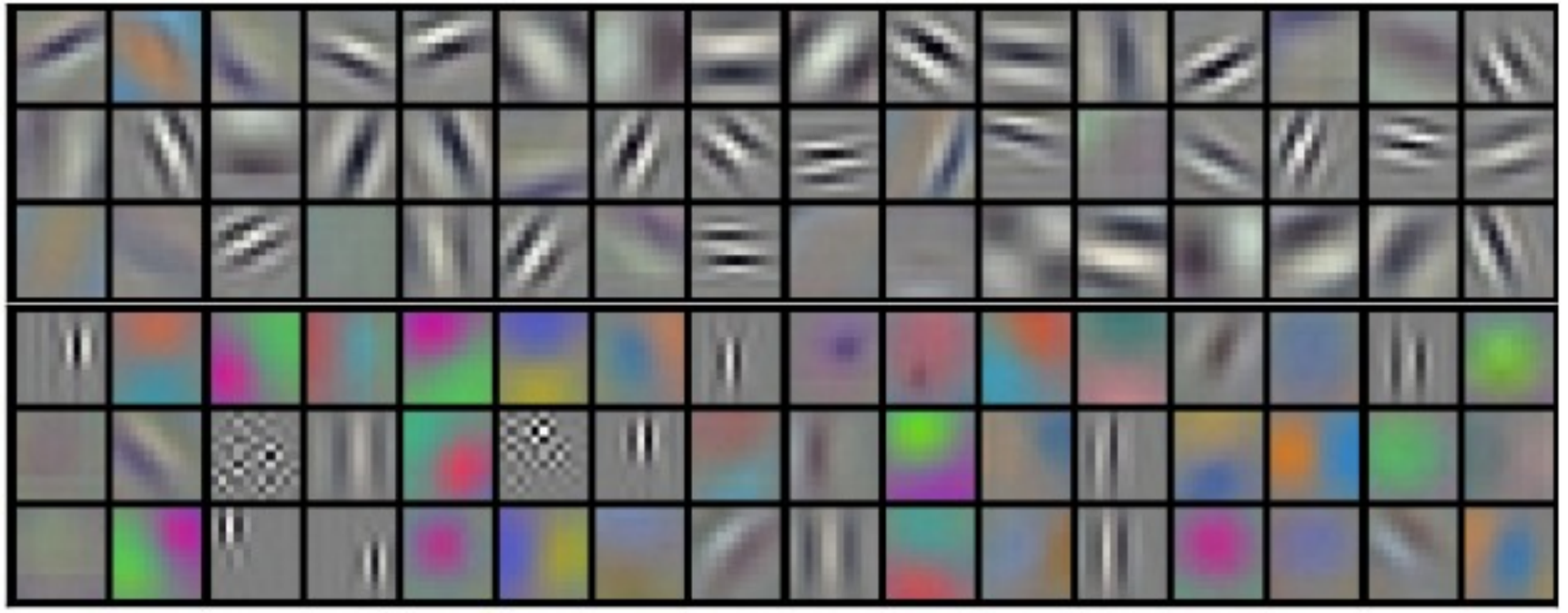
- Each neuron in the convolutional layer is connected only to a local region in the input volume spatially, but to the full depth
- There can be multiple neurons along the depth, all looking at the same region in the input





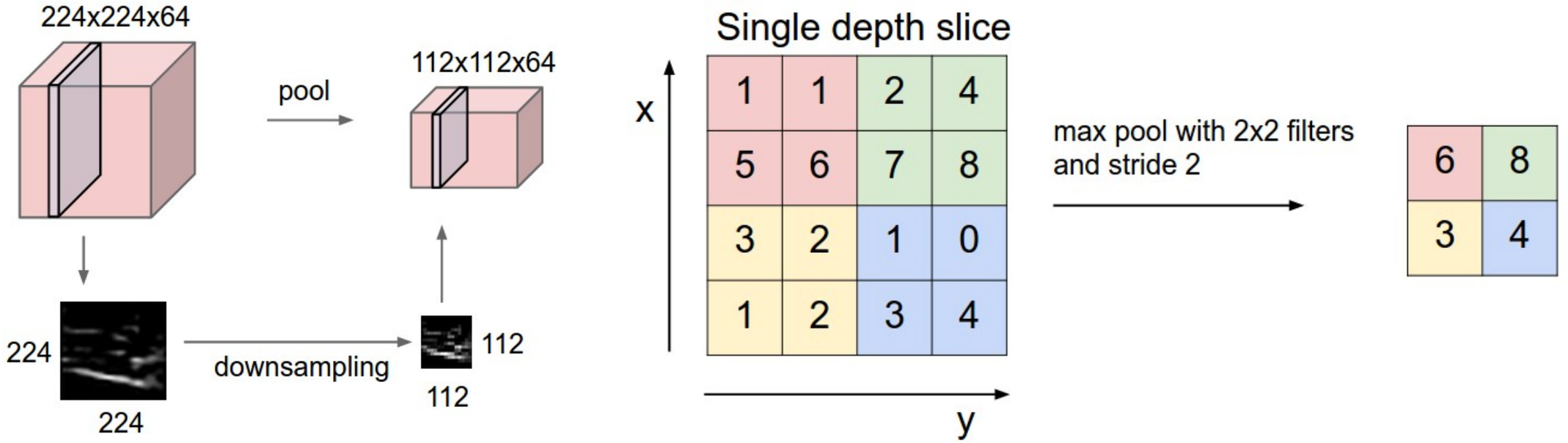
CONVOLUTIONAL NEURAL NETWORKS

- Example filters learned by a ConvNet

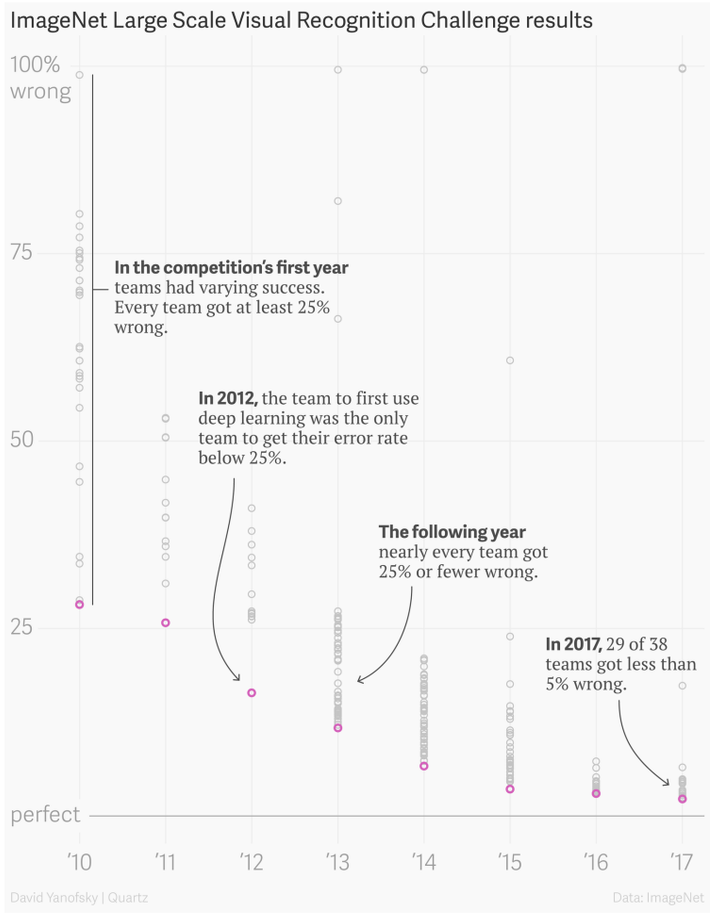


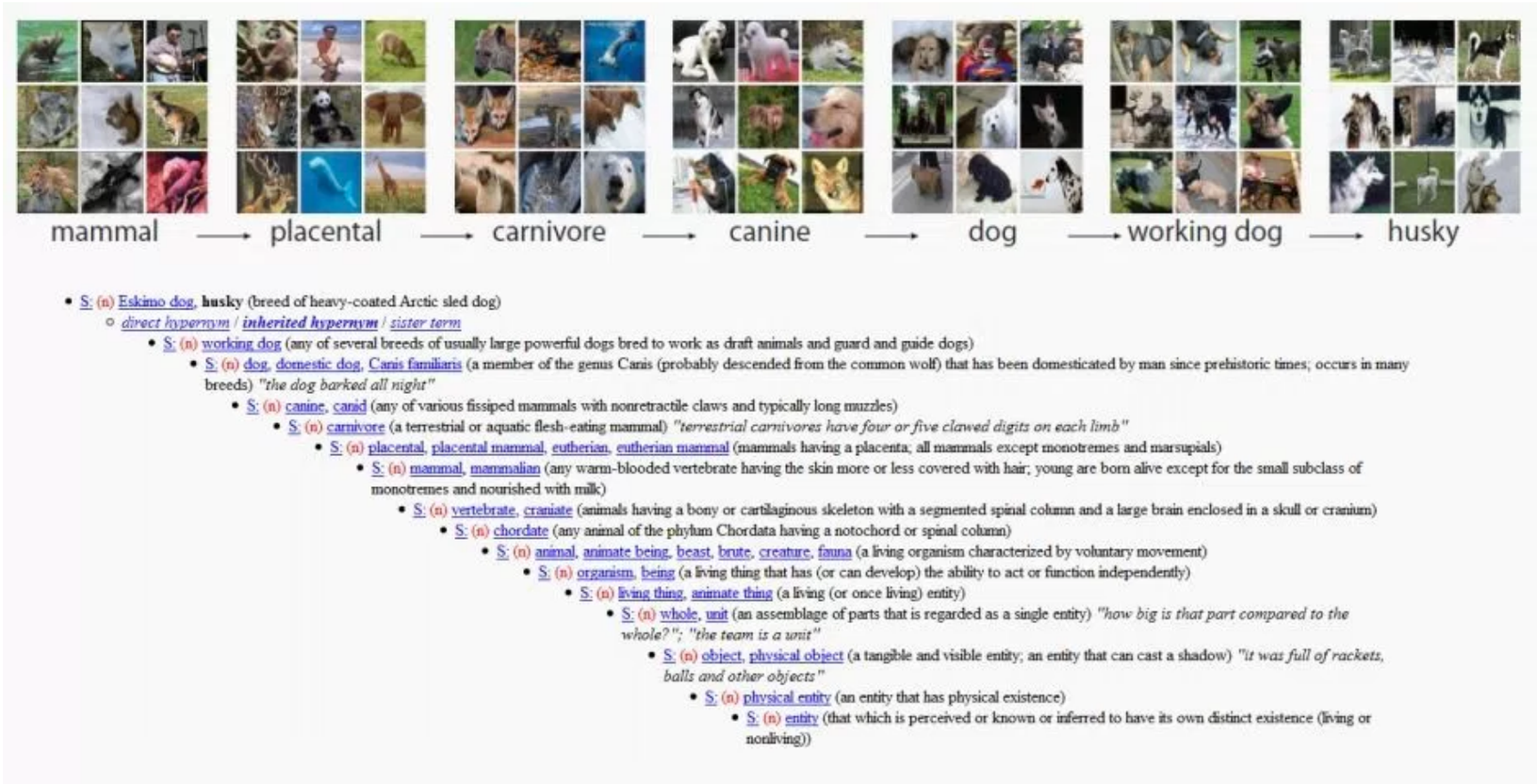
CONVOLUTIONAL NEURAL NETWORKS

- Pooling layers reduce the spatial size of the representation by down-sampling
- Max pooling is most commonly used and takes the highest value in each filter region



IMAGENET

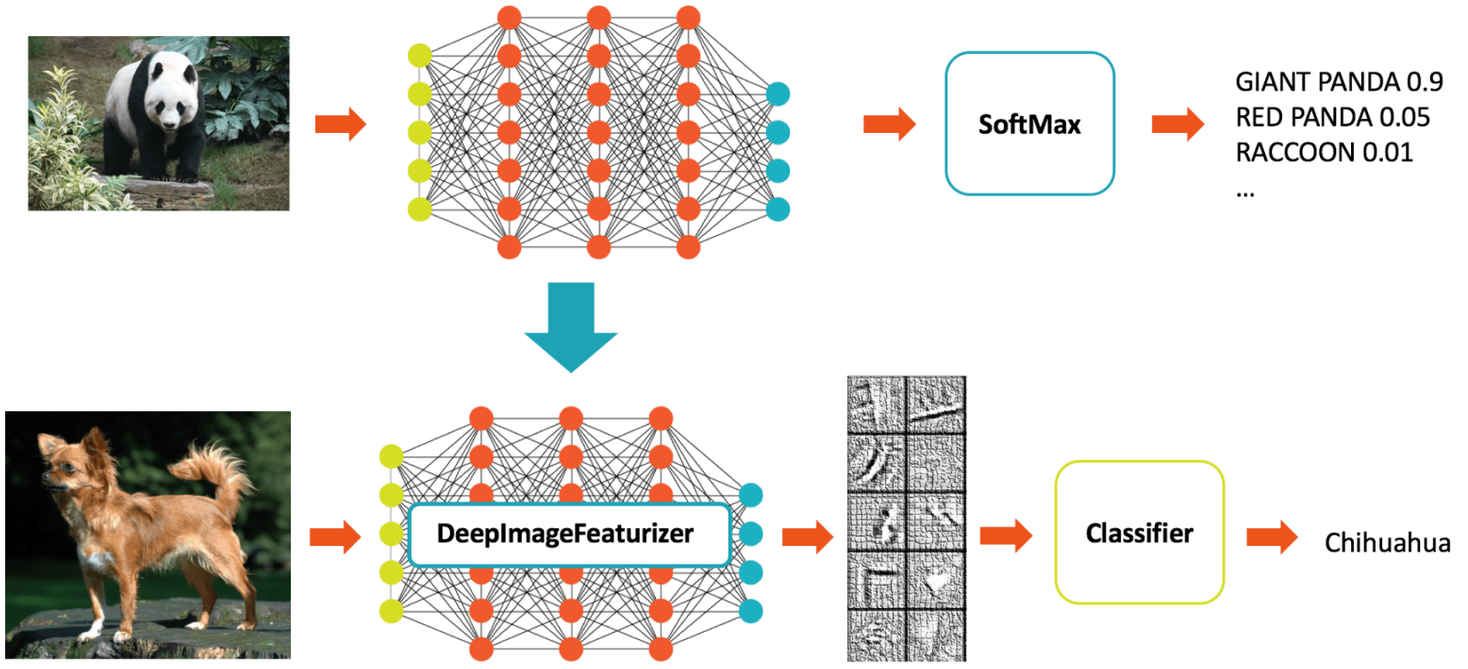




https://qz.com/1034972/the-data-that-changed-the-direction-of-ai-research-and-possibly-the-world/

TRANSFER LEARNING

- In practice, we typically do not need to create our own architecture for a problem but can instead download pre-trained models and fine tune it on our own data

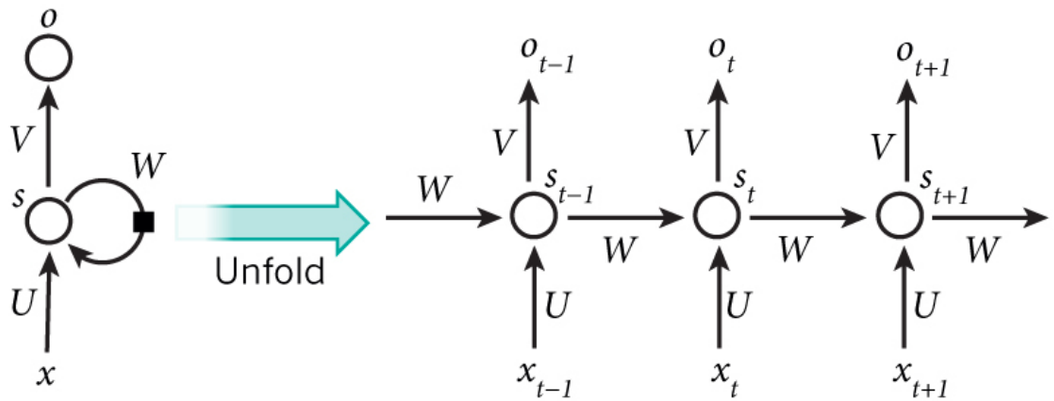


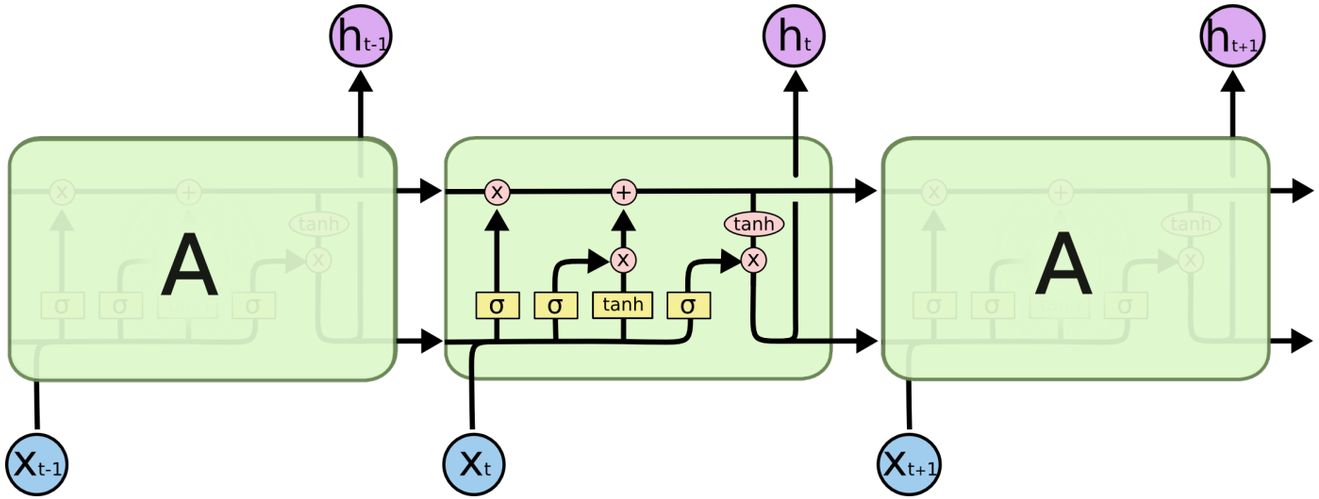
http://ruder.io/transfer-learning/

RECURRENT NEURAL NETWORKS

RECURRENT NEURAL NETWORKS

- Recurrent Neural Networks contain loops





http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

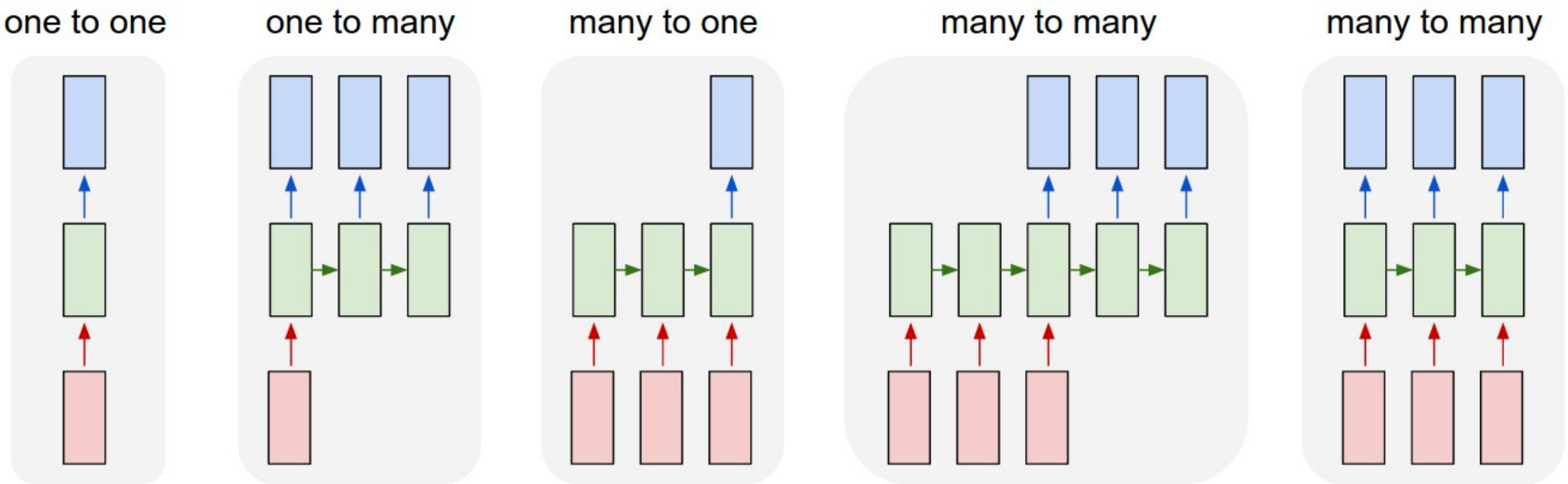
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

RECURRENT NEURAL NETWORKS

- Recurrent Neural Networks contain loops
- This implements feedback and gives neural networks “memory” or context
- Particularly good for predicting sequences, translating text, recognizing objects in images, speech translation
- Commonly referred to as **deep learning**, involving both feature extraction and modeling

RECURRENT NEURAL NETWORKS

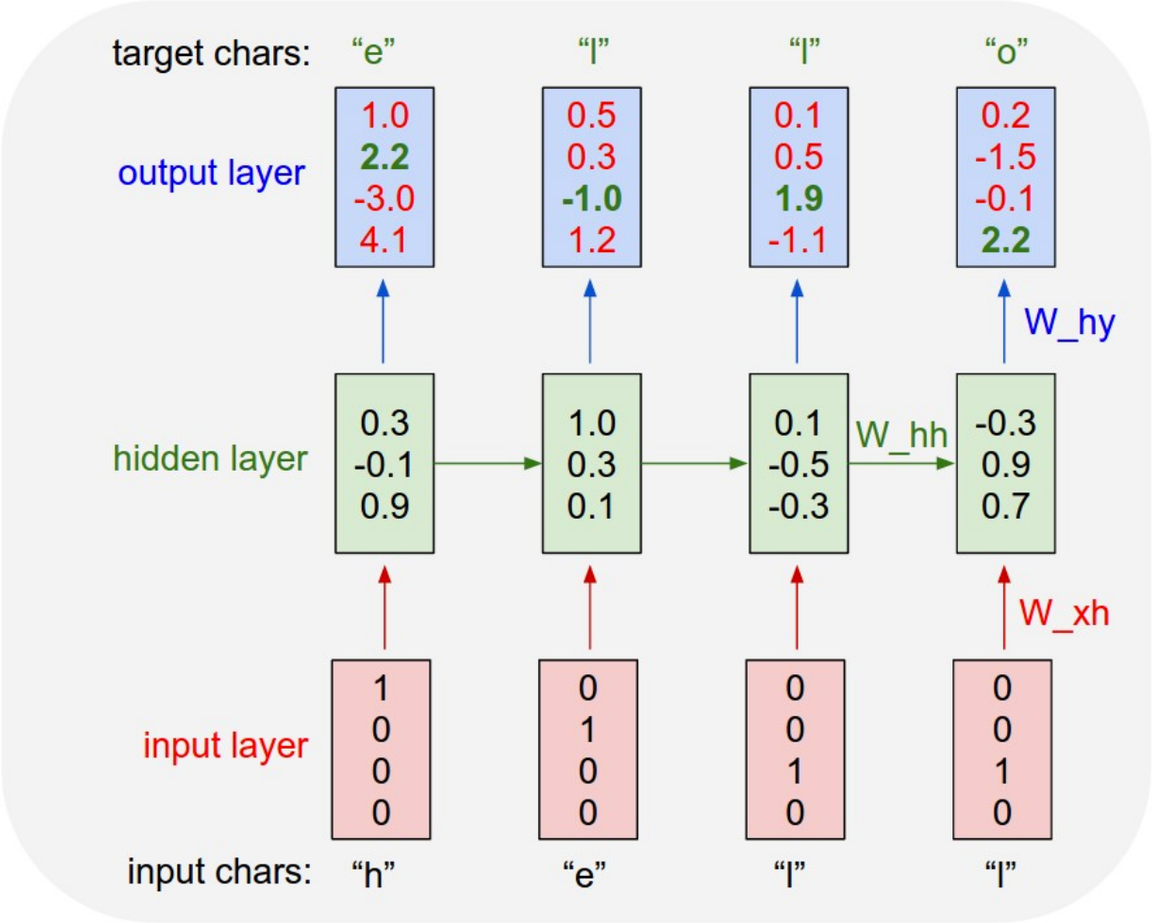
- Recurrent Neural Networks allow us to operate on sequences



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

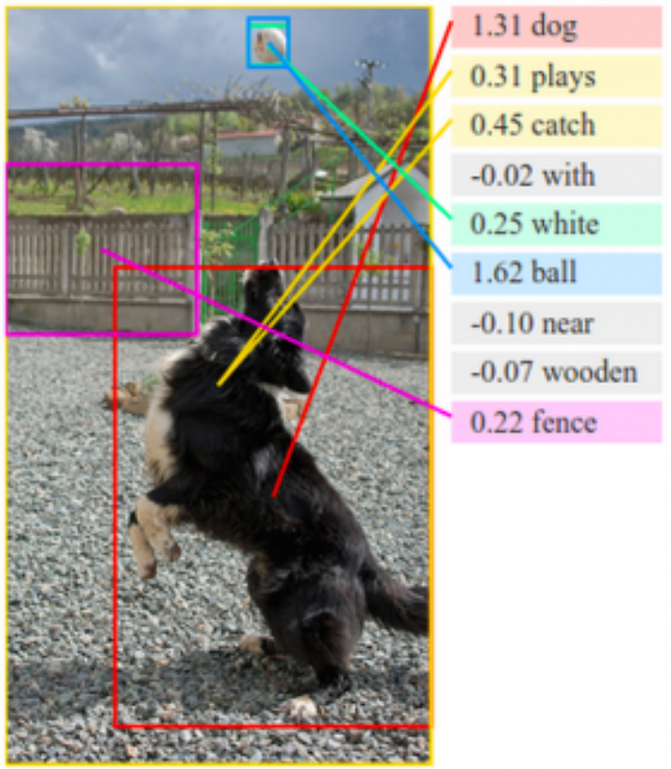
RECURRENT NEURAL NETWORKS

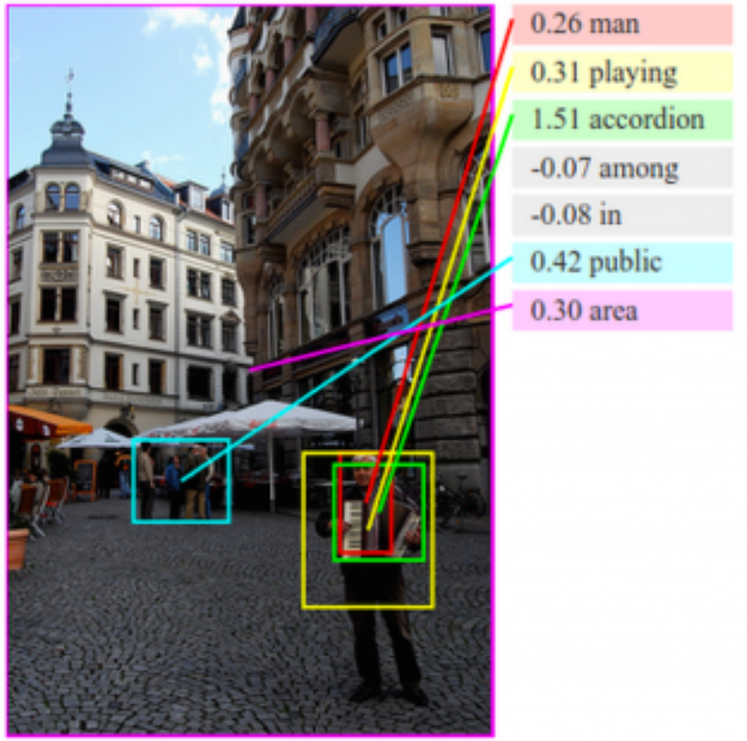
- Training of RNN model

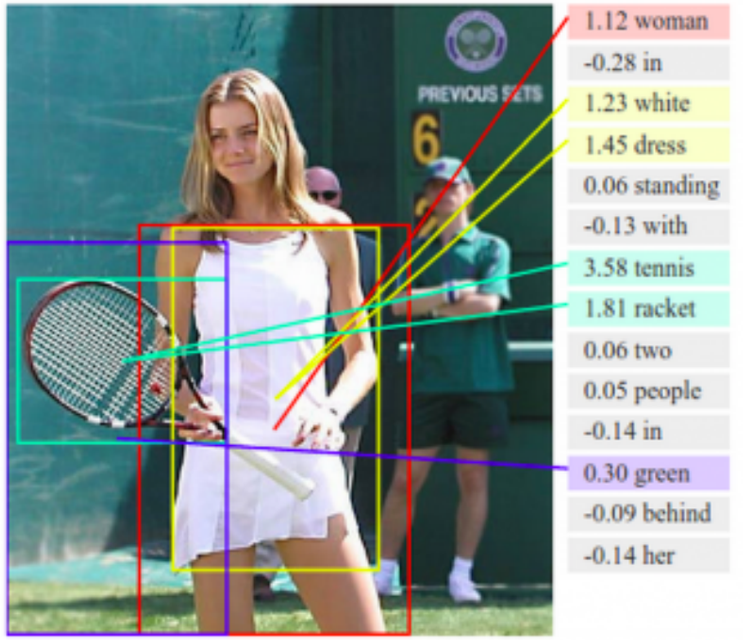


RECURRENT NEURAL NETWORKS

- Generate descriptions for unlabeled images







http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

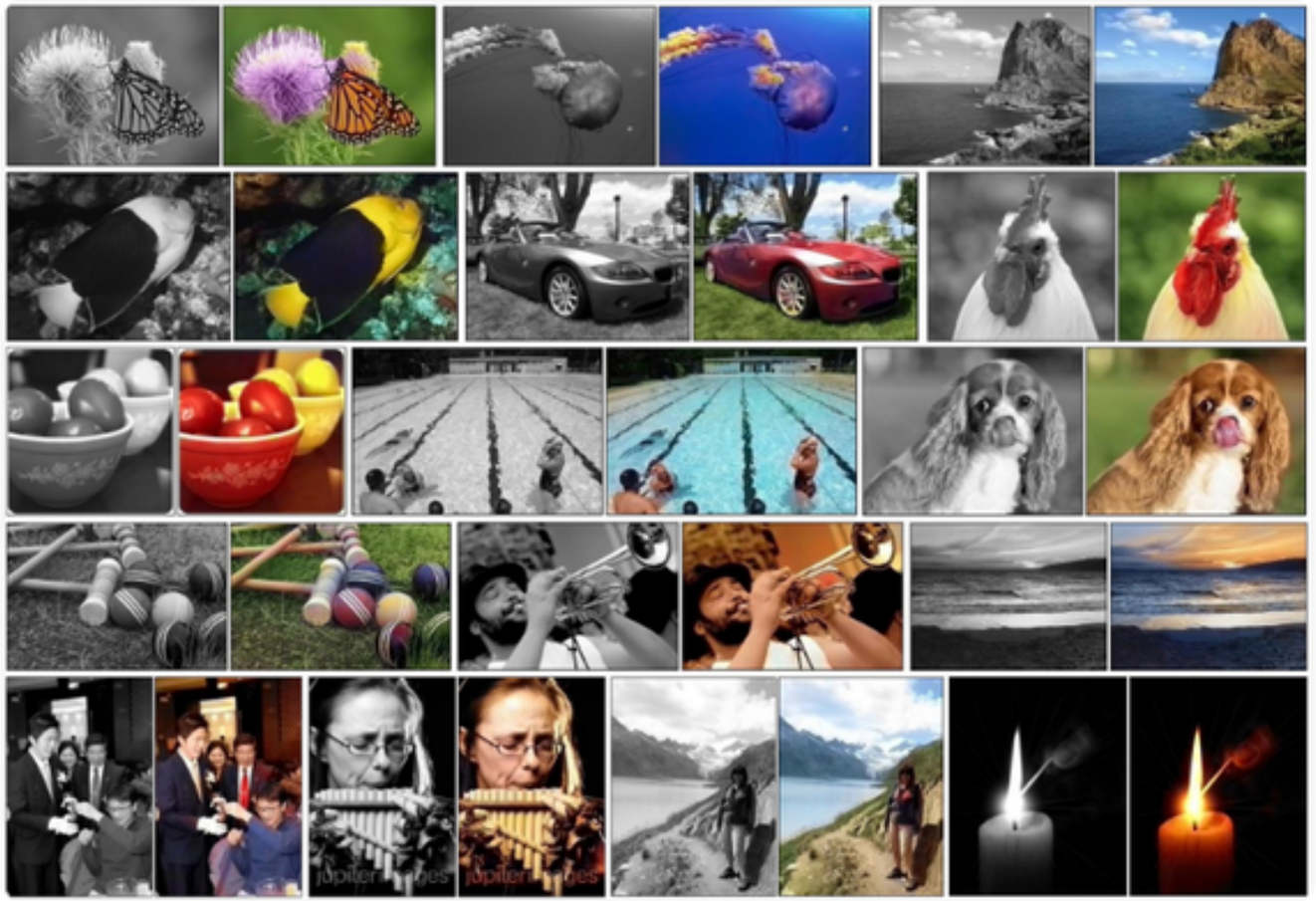
RECURRENT NEURAL NETWORKS

- RNN font analysis

https://erikbern.com/2016/01/21/analyzing-50k-fonts-using-deep-neural-networks.html

RECURRENT NEURAL NETWORKS

- Automatic Colorization with CNN



http://tinyclouds.org/colorize/

GUIDED PRACTICE

NEURAL NETWORKS IN PYTHON

ACTIVITY: KNOWLEDGE CHECK

ANSWER THE FOLLOWING QUESTIONS

1. Let's practice using neural networks for classification. For each of the four datasets, experiment with the number of layers and neurons to find the best model
2. Also take a look at this visualization

DELIVERABLE

Answers to the above questions

NN IN PYTHON

- There are many NN libraries for python and other languages
- Python
  - Theano
  - Keras
  - Lasagne
  - TensorFlow
  - Scikit Learn support for NN coming in 0.18
- Lua
  - Torch

- Some of these libraries utilize GPUs for (much) faster training

NN IN PYTHON

- Let’s look at some examples in Keras
  - Regression
  - Classification

NN IN PYTHON

- Network design is a hard problem
  - Experience helps
  - Evolutionary algorithms are useful for design
  - Nice (free) book available

http://hagan.okstate.edu/NNDesign.pdf

CONCLUSION

TOPIC REVIEW

CONCLUSION: Neural Networks

Pros:

- Flexible
- Good for a variety of tasks
- Good for many types of data

Cons:

- Can require a lot of data
- Training may be slow
- Many parameters to tune
- Many layer types and activations
- Black Box model

CONCLUSION

- Many more examples for Keras available
- Recommended articles: Convolutional NN,
- Advanced machine learning methods you should explore include Bayesian methods and deep learning

https://github.com/fchollet/keras/tree/master/examples

http://cs231n.github.io/neural-networks-1/

LESSON

Q & A

LESSON

EXIT TICKET

DON’T FORGET TO FILL OUT YOUR EXIT TICKET