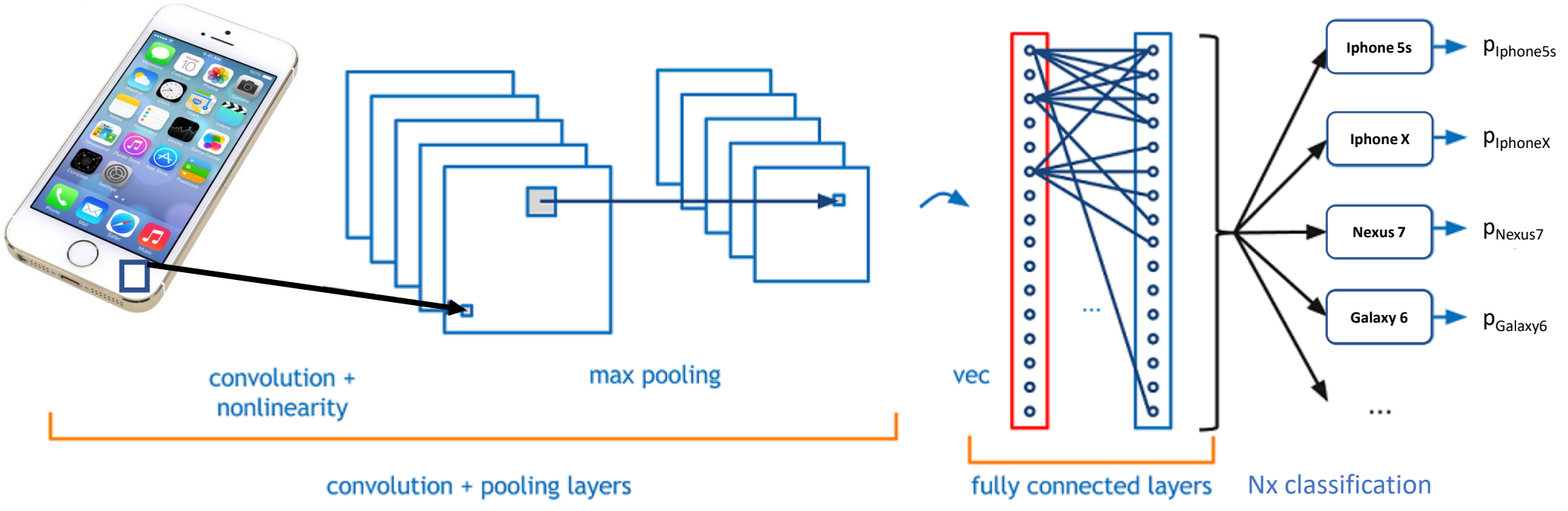
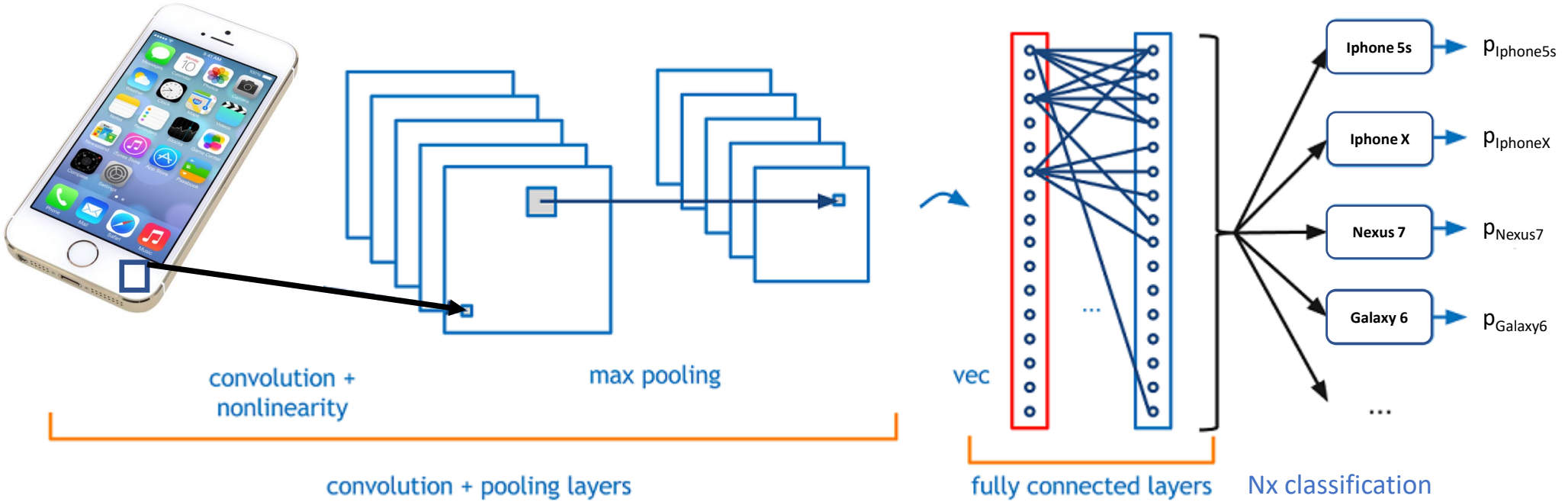
Use Convolution Neural Nets to Address

CLASSIFICATION



Use Convolution Neural Nets to Address

- Input: Images that have isolated or non-isolated products
- Training: Labeling data, pre-process data, optimize parameters (hit a button),
- Exhaustive vs. non-exhaustive set of images for training
- Output: Classified category



Ground truth: we need the labeled images

- Lots of labels
  - hire people on Amazon turk (www.Mturk.com),
  - scour the internet (require pre-processing)
  - Use people in recycling plants looking at images as they come in
    - More costly and time consuming
    - Better quality images for training because more realistic

- Few labels
  - One-shot learning – don’t need as much data, just need a few images of each

Ground truth: we need the labeled images

- Lots of labels
  - hire people on Amazon turk (www.Mturk.com),
  - scour the internet (require pre-processing)
  - Use people in recycling plants looking at images as they come in
    - More costly and time consuming
    - Better quality images for training because more realistic

- Few labels
  - One-shot learning – don't need as much data, just need a few images of each

Obtaining the training data could be a challenge and time consuming

Ground truth: we need the labeled images

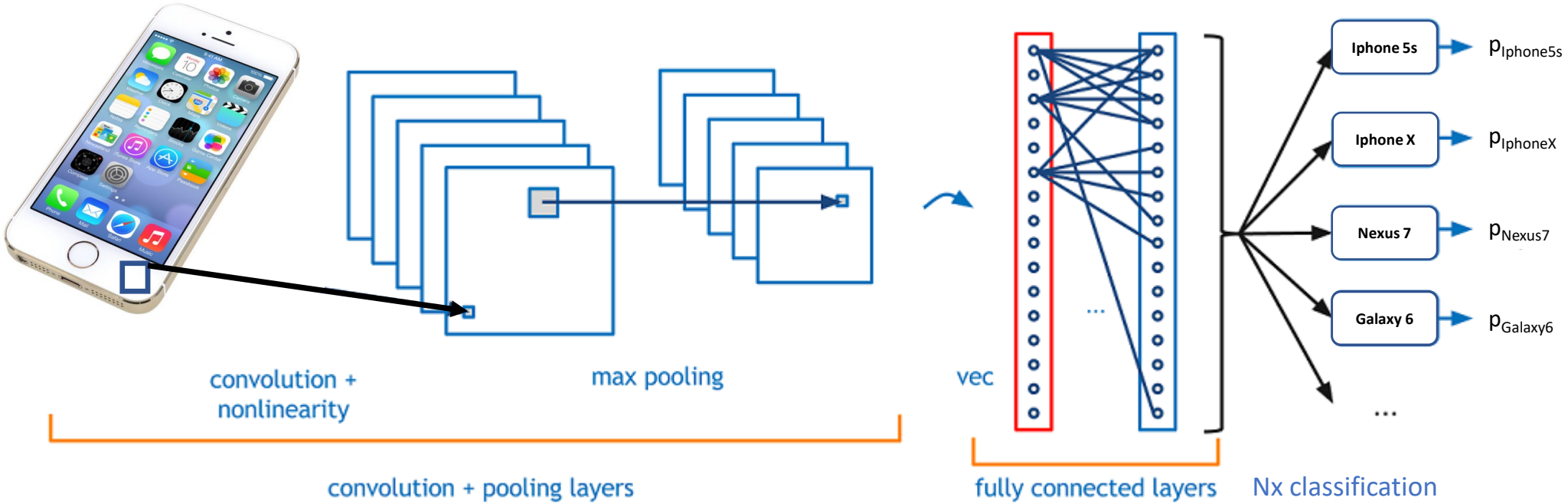
- Lots of labels
  - hire people on Amazon turk (www.Mturk.com),
  - scour the internet (require pre-processing)
  - Use people in recycling plants looking at images as they come in
    - More costly and time consuming
    - Better quality images for training because more realistic

- Few labels
  - One-shot learning – don't need as much data, just need a few images of each

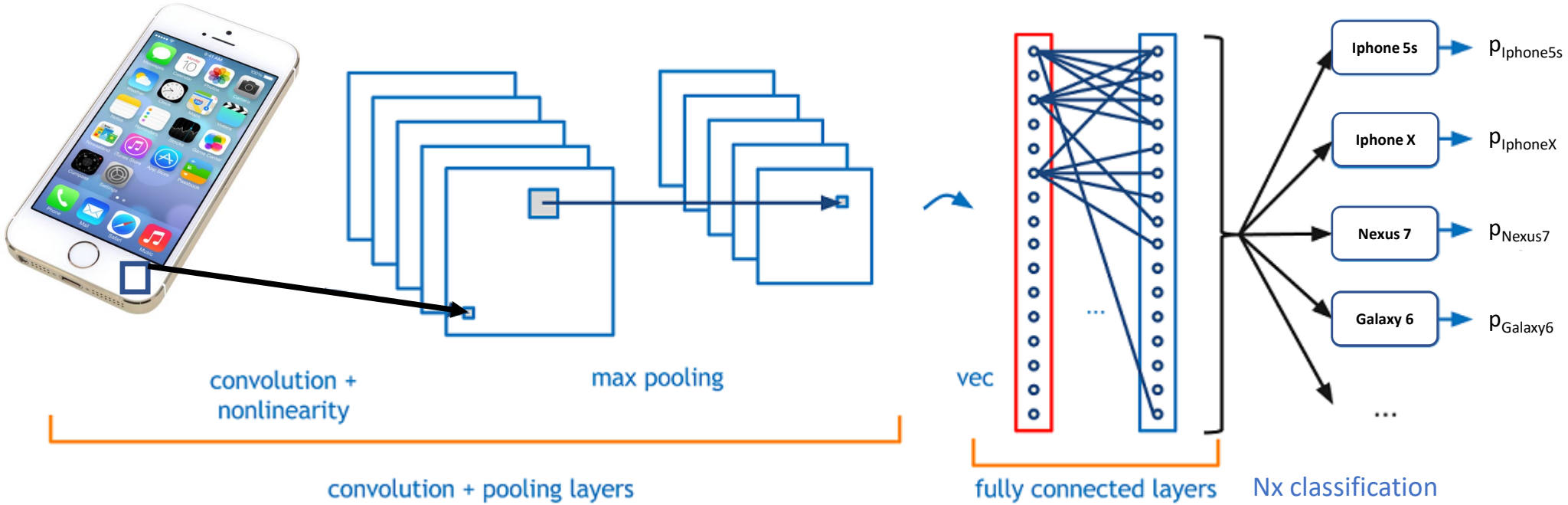
Obtaining the training data could be a challenge and time consuming

Classify smartphones by model/brand

- Input: Images have only a single phone in them
- Training: Each training image contains a label saying what make and model
- Exhaustive set of images for training
- Output: make and model



Convolutional neural network



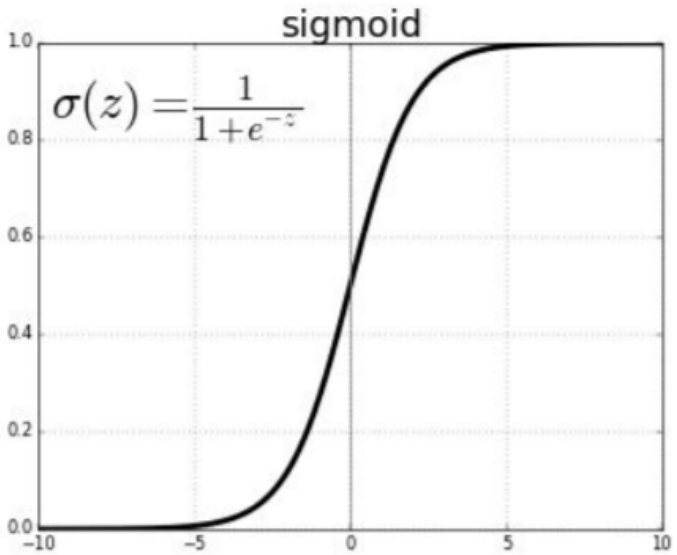
Key words:

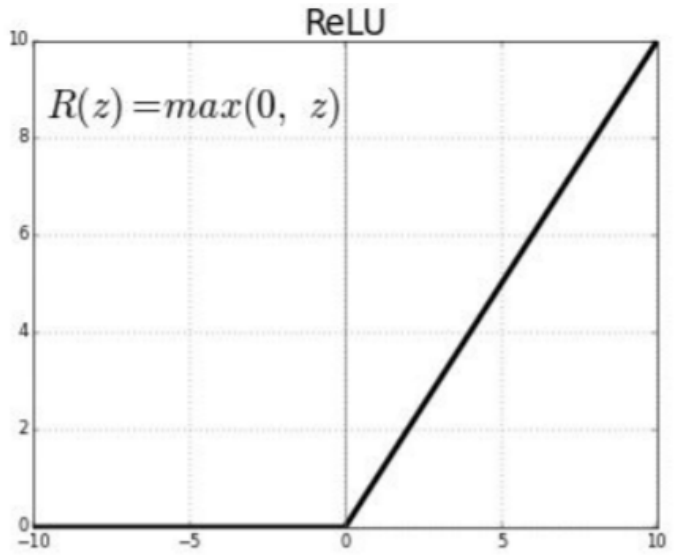
1. Convolution
2. Nonlinearlity
3. Max pooling
4. Fully connected layers
5. Classification

Convolution

- A kernel matrix (filter), moves across the image in a stride and takes dot product with the numbers on the image.
- When training the network we end up finding the numbers that make up the kernel matrix.
- Each image may be convolved by a number of kernels

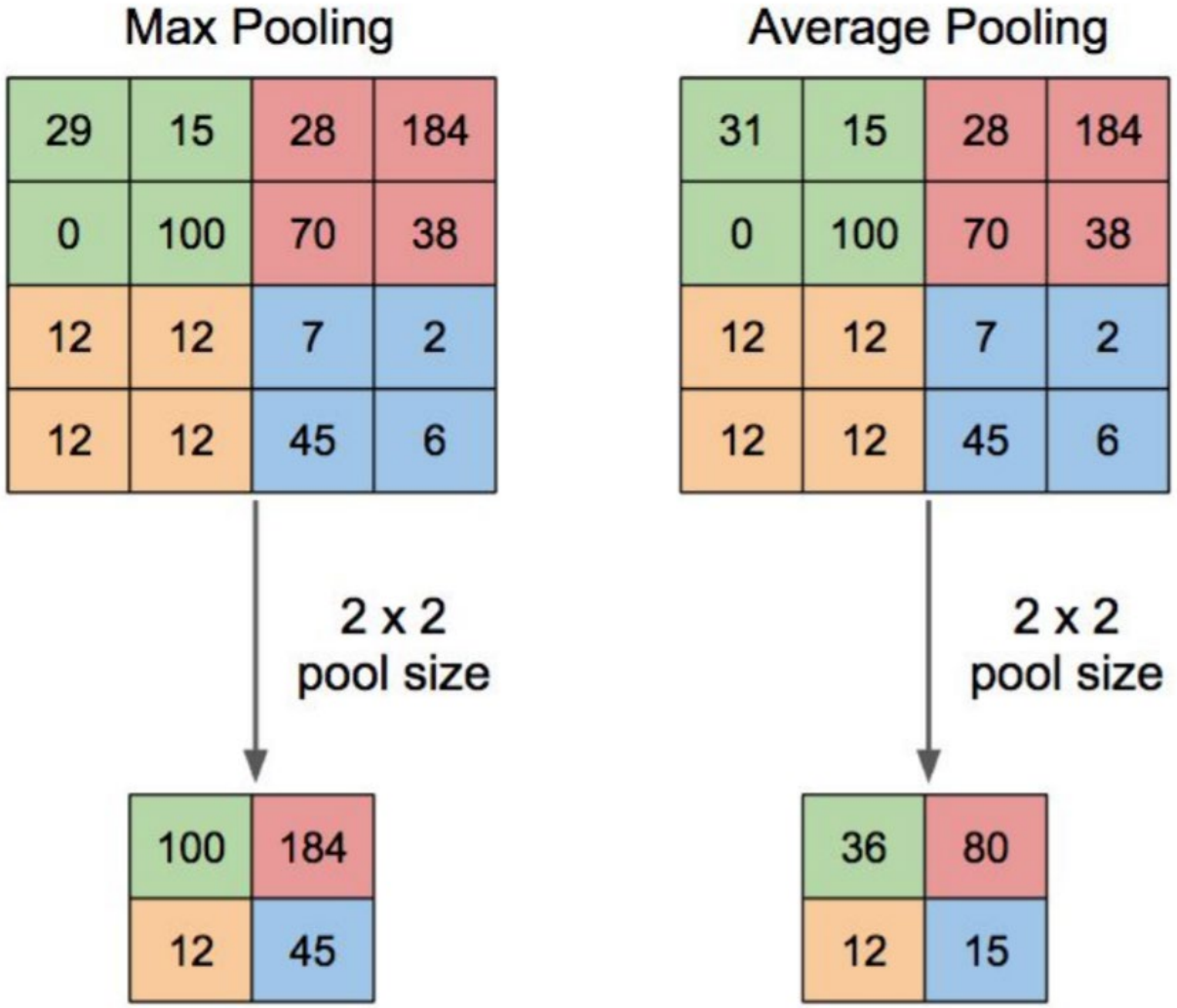
Nonlinearlity





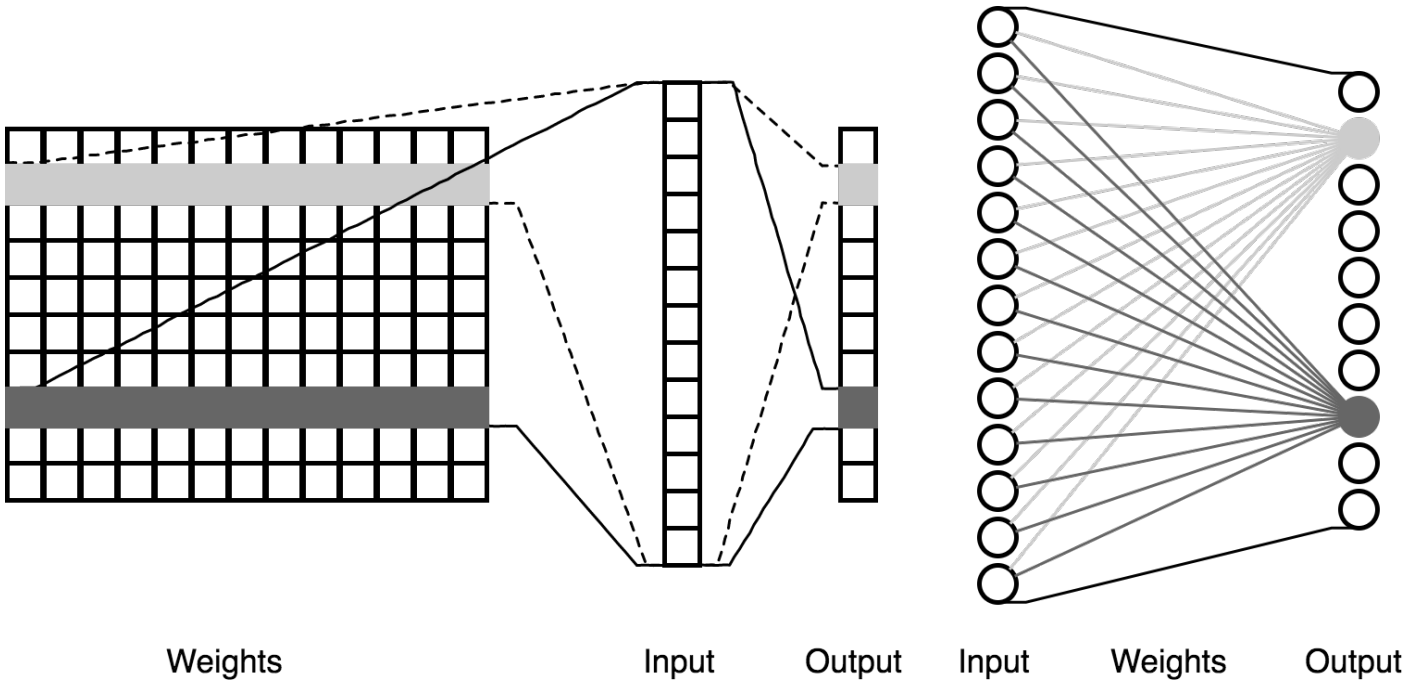
- Activation functions are used to increase the nonlinearity of the network.
- ReLU is typically preferred as it results in faster training.
- Other types of activation include:
  - Leaky ReLU
  - Tanh
  - Exponential linear units
  - Softmax

Pooling



- Down-samples input image allowing for assumptions to be made about features contained in the sub-regions.
- Helps prevent over-fitting by providing an abstracted form of the representation.
- Reduces computational cost by reducing the number of parameters to learn
- Provides basic translation invariance to the internal representation.

Fully connected layers



Obtained from the convolutional layers

- Every element from the pooling layer would be connected to every element of the output layer.
- These output elements may be some type of activation function (i.e. a softmax function) and will return a probability representing how likely a particular object with the features described in the kernels is in the image.
- In our example, output could be probability that the image is of a particular phone model/make

Classification

- A loss function is defined: Difference between the output of the fully connected layer and the ground truth
  - L1,
  - L2,
  - Cross entropy (preferred for image classification)
- Loss function is minimized over all the weights/parameters from previous layers (kernels of convolutions layers, weights of fully connected layers etc.)
- Minimization carried out using
  - Stochastic gradient descent (SGD)
  - Adaptive gradient descent (Adagrad)
  - Adaptive moment estimation (Adam)