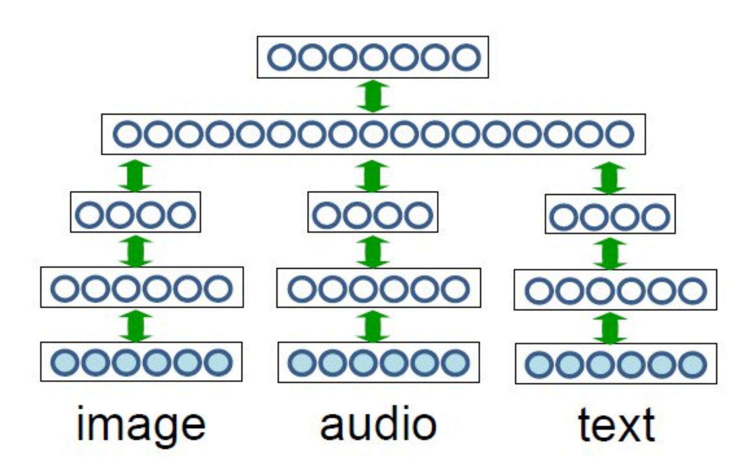
Foundations of Machine Learning

Module 6: Neural Network
Part D: Deep Neural Network

Sudeshna Sarkar
IIT Kharagpur

Deep Learning

- Breakthrough results in
  - Image classification
  - Speech Recognition
  - Machine Translation
  - Multi-modal learning



Deep Neural Network

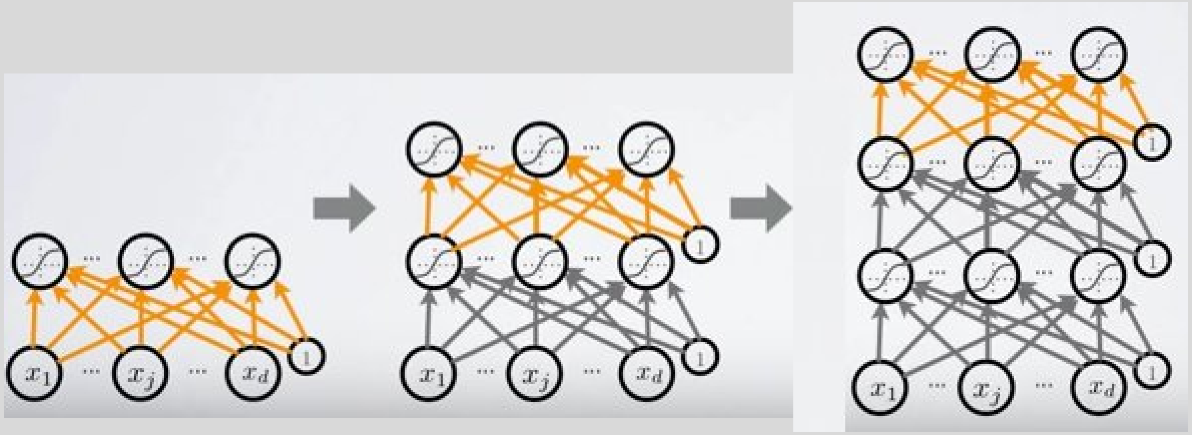
- Problem: training networks with many hidden layers doesn’t work very well
- Local minima, very slow training if initialize with zero weights.
- Diffusion of gradient.

Hierarchical Representation

- Hierarchical Representation help represent complex functions.
- NLP: character ->word -> Chunk -> Clause -> Sentence
- Image: pixel > edge -> texton -> motif -> part -> object
- Deep Learning: learning a hierarchy of internal representations
- Learned internal representation at the hidden layers (trainable feature extractor)
- Feature learning

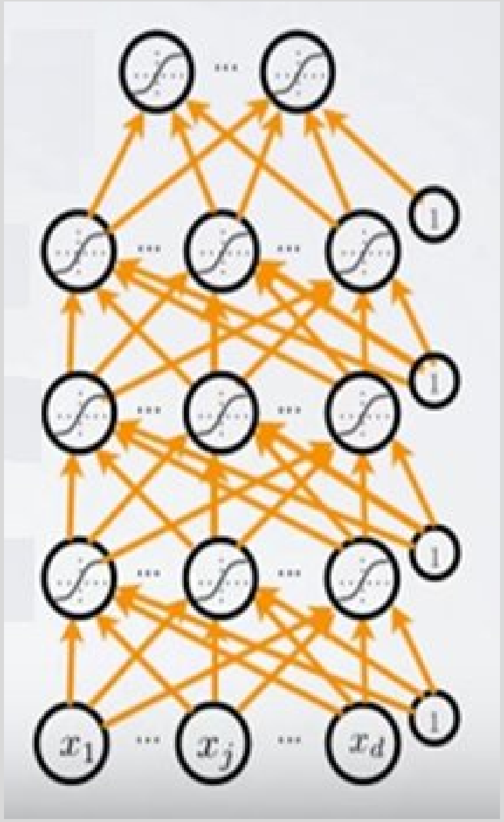
Unsupervised Pre-training

- We will use greedy, layer wise pre-training
  - Train one layer at a time
  - Fix the parameters of previous hidden layers
  - Previous layers viewed as feature extraction
- find hidden unit features that are more common in training input than in random inputs



Tuning the Classifier

- After pre-training of the layers
  - Add output layer
  - Train the whole network using supervised learning (Back propagation)



Deep neural network

- Feed forward NN
- Stacked Autoencoders (multilayer neural net with target output = input)
- Stacked restricted Boltzmann machine
- Convolutional Neural Network

A Deep Architecture: Multi-Layer Perceptron

Output Layer

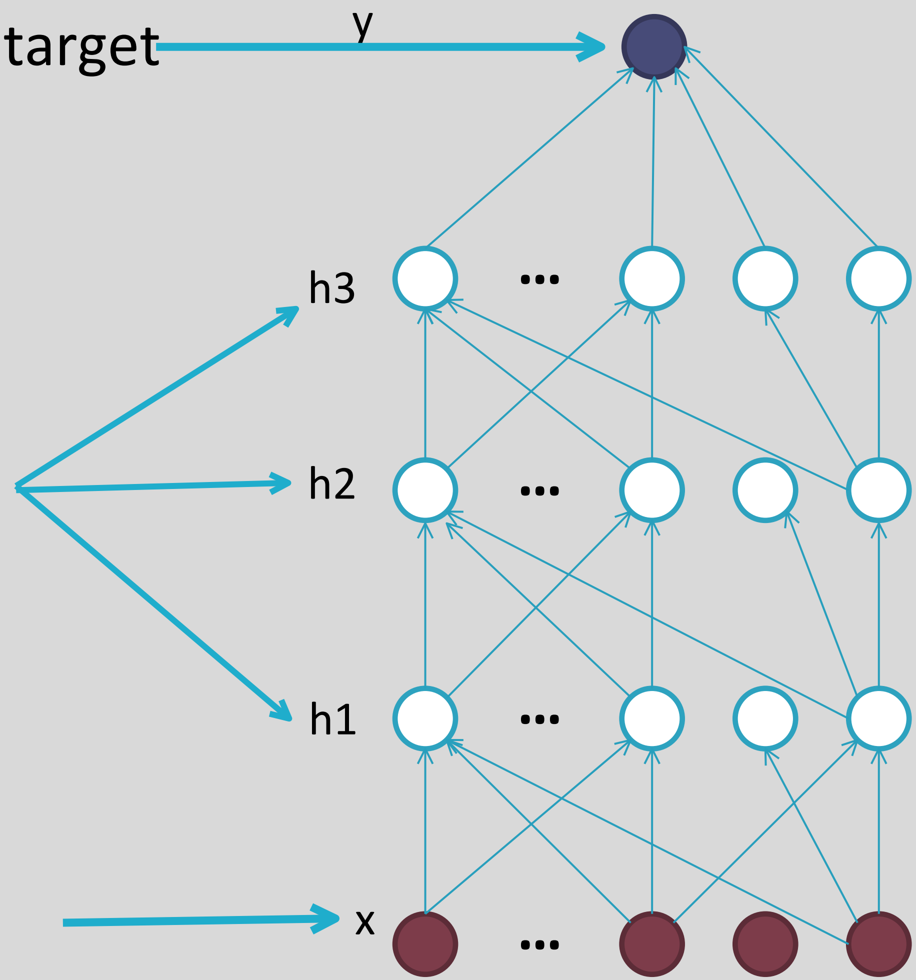
Here predicting a supervised target

Hidden layers

These learn more abstract representations as you head up

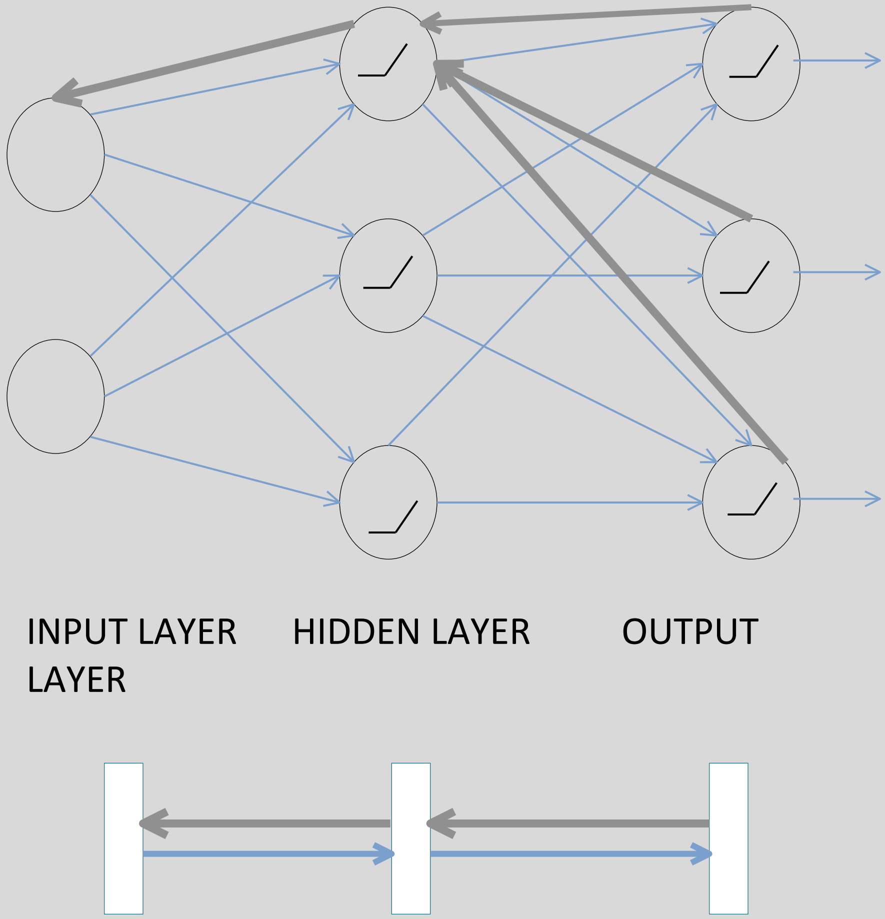
Input layer

Raw sensory inputs



A Neural Network

- Training : Back Propagation of Error
  - Calculate total error at the top
  - Calculate contributions to error at each step going backwards
  - The weights are modified as the error is propagated

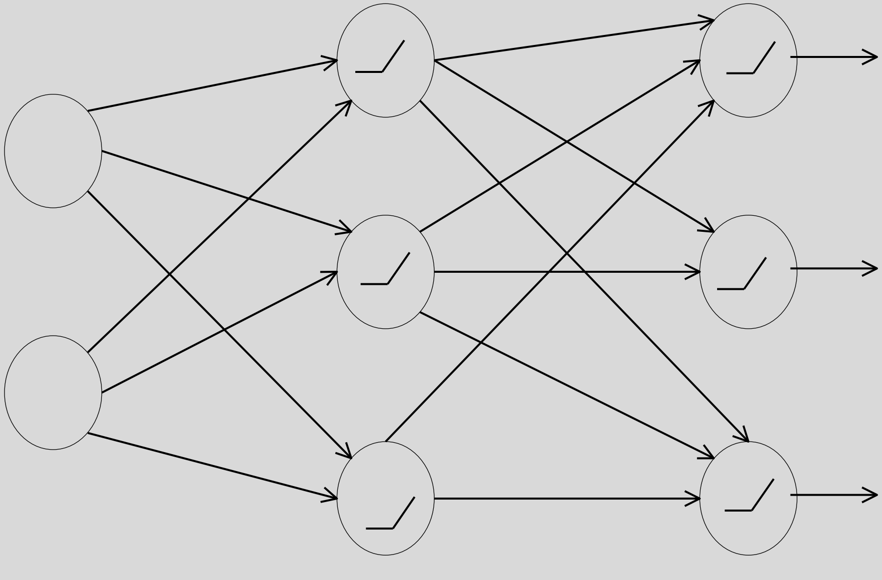


Training Deep Networks

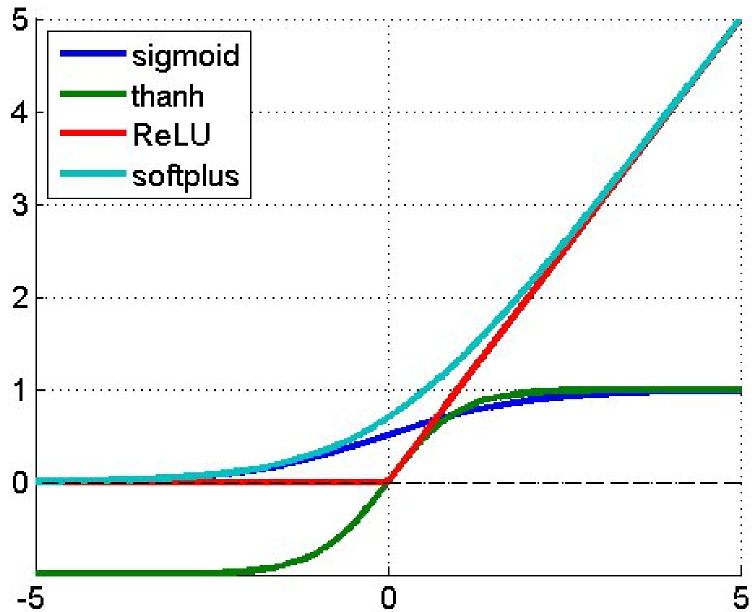
- Difficulties of supervised training of deep networks
  1. Early layers of MLP do not get trained well
     - Diffusion of Gradient – error attenuates as it propagates to earlier layers
     - Leads to very slow training
     - the error to earlier layers drops quickly as the top layers "mostly" solve the task
  2. Often not enough labeled data available while there may be lots of unlabeled data
  3. Deep networks tend to have more local minima problems than shallow networks during supervised training

Training of neural networks

- Forward Propagation :
  - Sum inputs, produce activation
  - feed-forward



Activation Functions examples



Activation Functions

Non-linearity

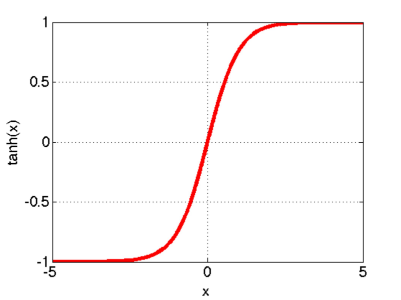
- $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$

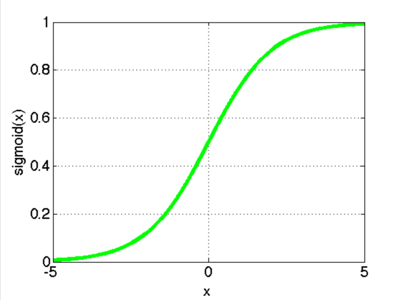
- $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$

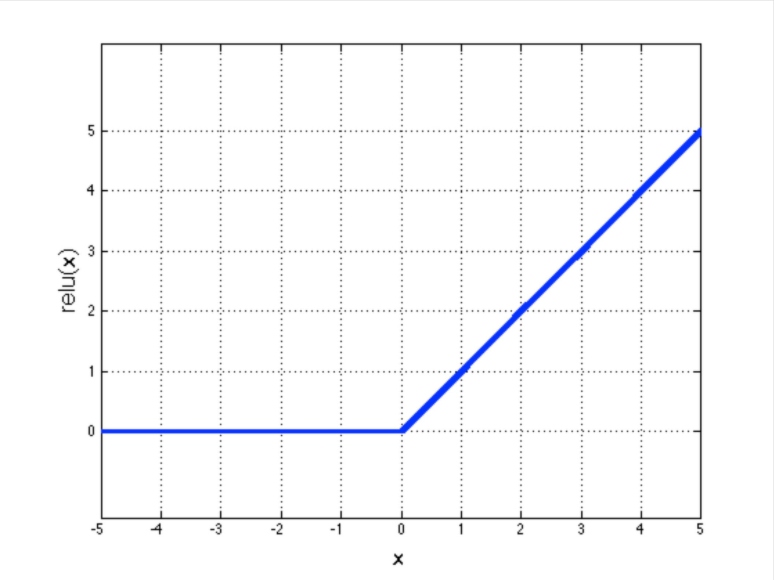
- Rectified linear
  relu(x) = max(0,x)

- Simplifies backprop
- Makes learning faster
- Make feature sparse

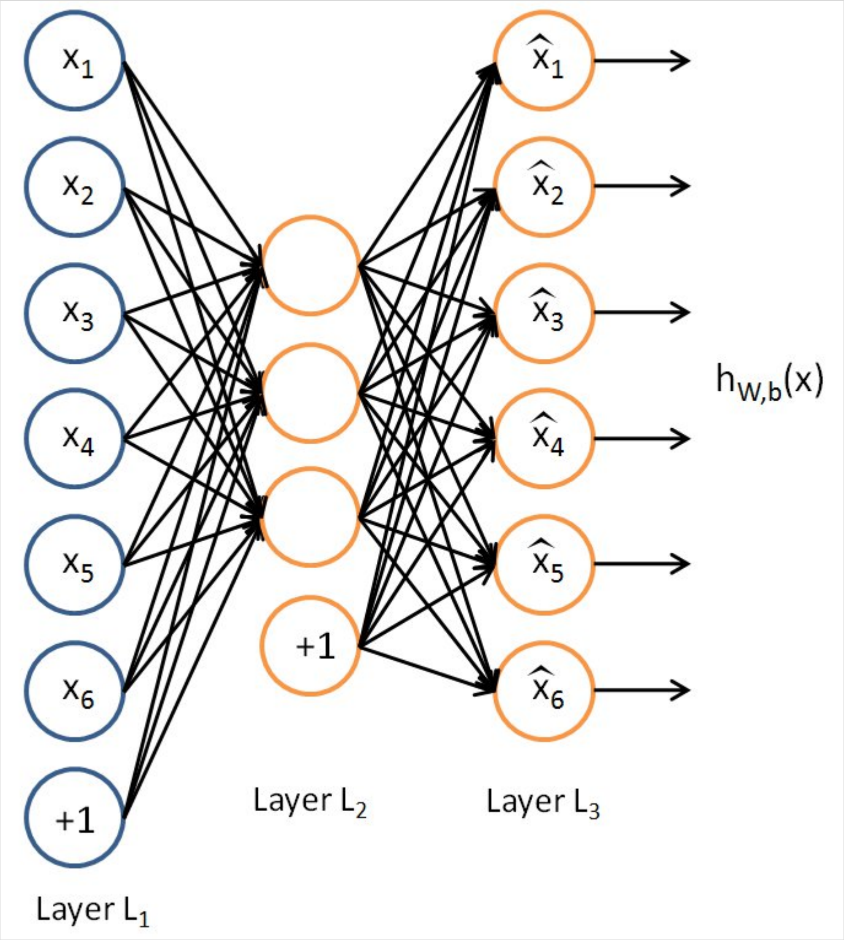
→ Preferred option







Autoencoder



Unlabeled training examples set

$\{x^{(1)}, x^{(2)}, x^{(3)} \ ...\}, \ x^{(i)} \in \mathbb{R}^n$

Set the target values to be equal to the inputs. $y^{(i)} = x^{(i)}$

Network is trained to output the input (learn identify function).

$$h_{w,b}(x) \approx x$$

Solution may be trivial!

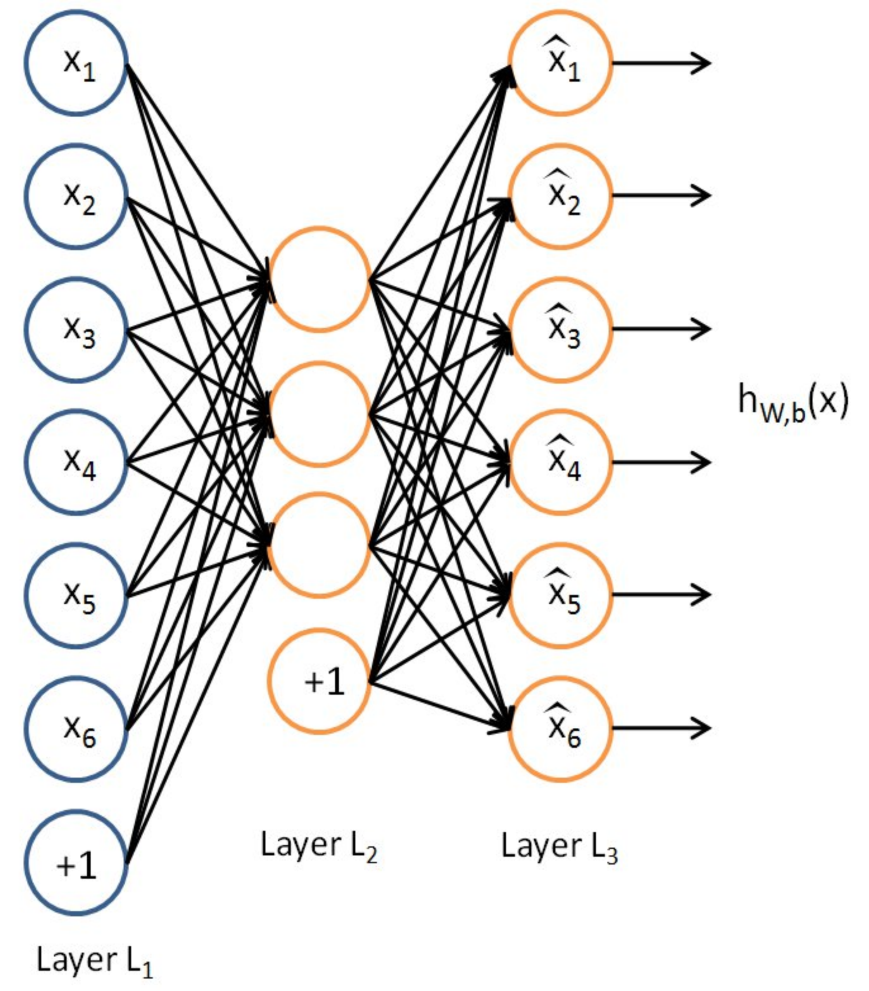
Autoencoders and sparsity

1. Place constraints on the network, like limiting the number of hidden units, to discover interesting structure about the data.
2. Impose sparsity constraint.

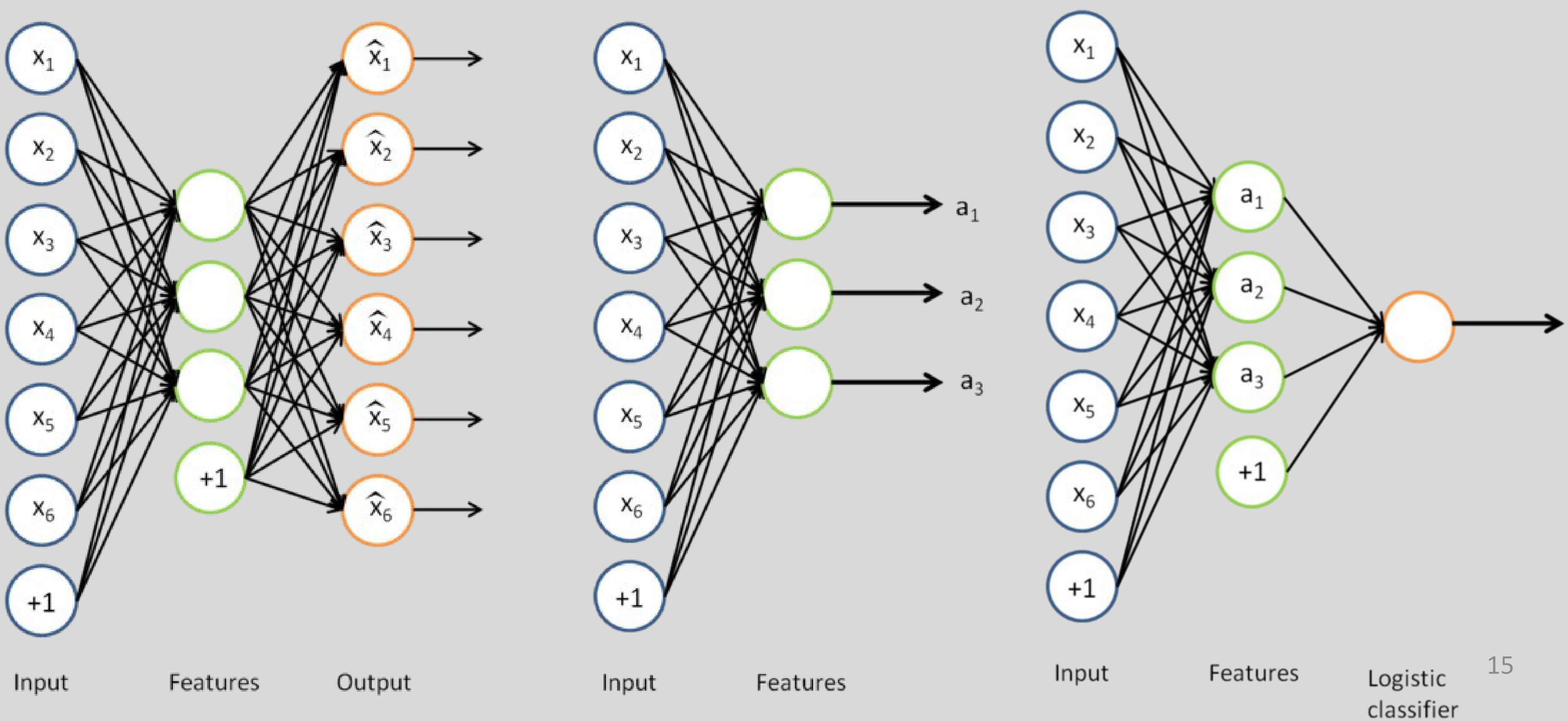
a neuron is “active” if its output value is close to 1

It is “inactive” if its output value is close to 0.

constrain the neurons to be inactive most of the time.

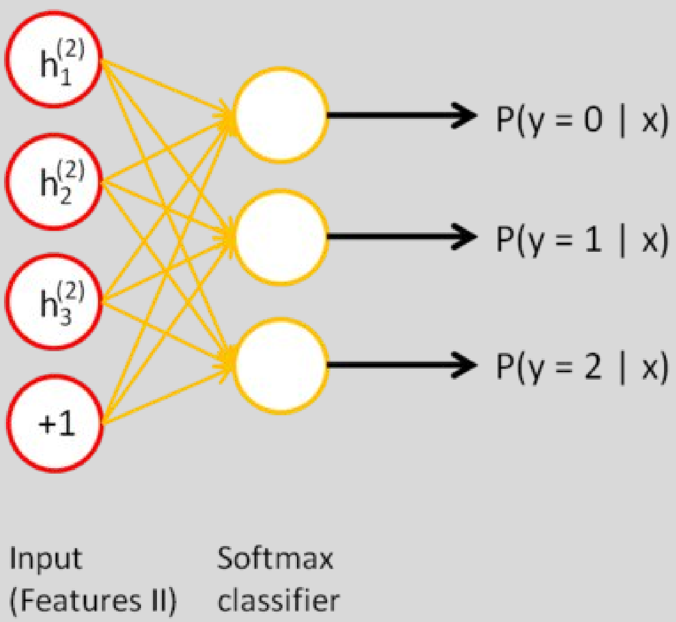


Auto-Encoders

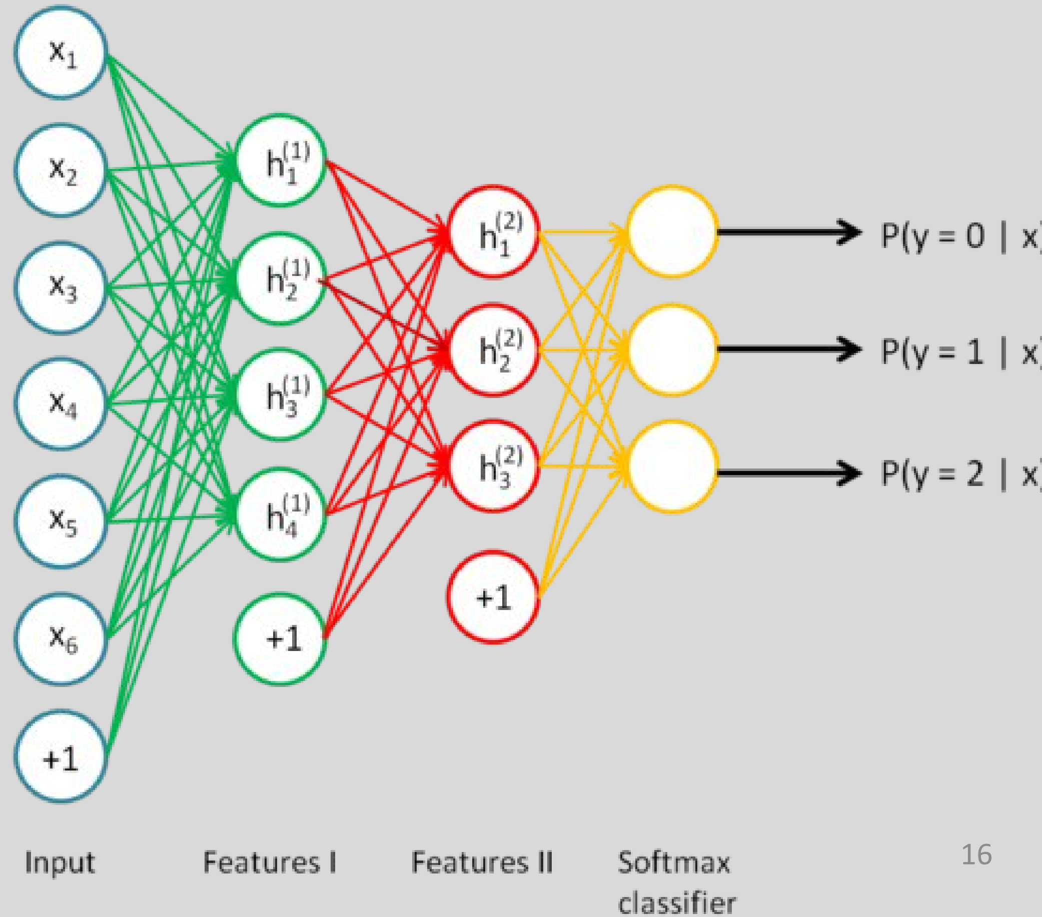


Stacked Auto-Encoders

- Do supervised training on the last layer using final features
- Then do supervised training on the entire network to fine- tune all weights



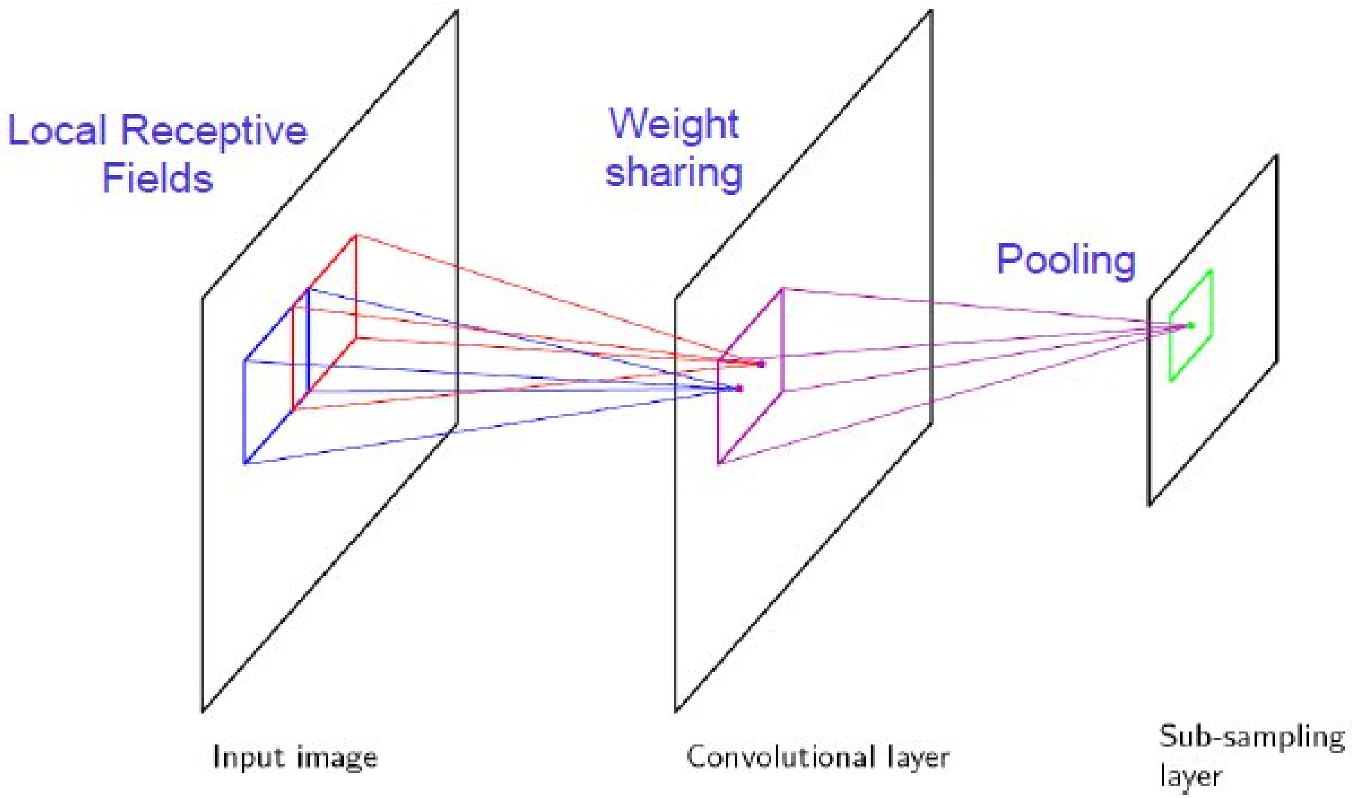
$$y_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$



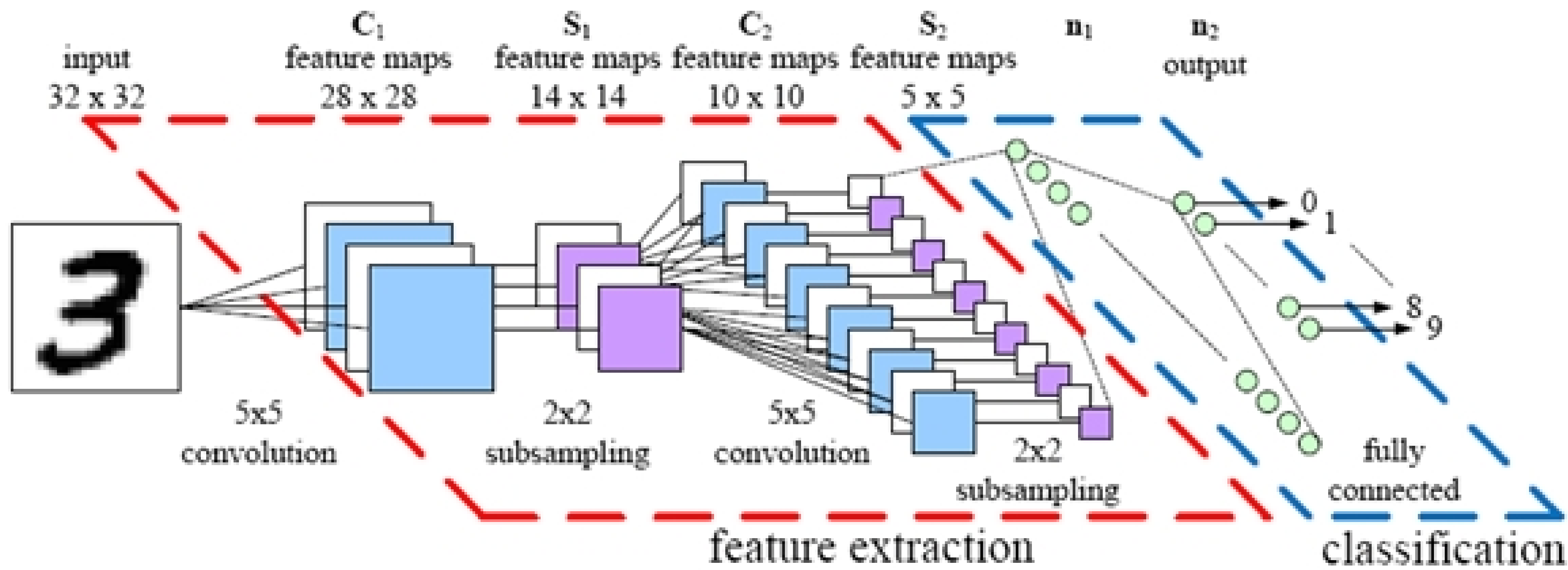
Convolutional Neural netwoks

- A CNN consists of a number of convolutional and subsampling layers.
- Input to a convolutional layer is a m x m x r image where m x m is the height and width of the image and r is the number of channels, e.g. an RGB image has r=3
- Convolutional layer will have k filters (or kernels)
- size n x n x q
- n is smaller than the dimension of the image and,
- q can either be the same as the number of channels r or smaller and may vary for each kernel

Convolutional Neural Networks

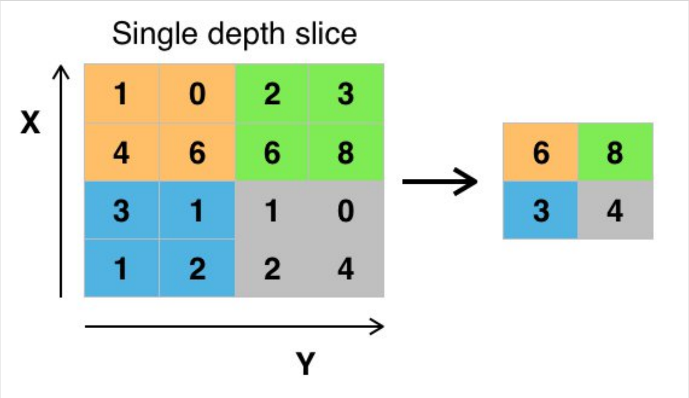


Convolutional layers consist of a rectangular grid of neurons
Each neuron takes inputs from a rectangular section of the previous layer
the weights for this rectangular section are the same for each neuron in the convolutional layer.



Pooling: Using features obtained after Convolution for Classification

The pooling layer takes small rectangular blocks from the convolutional layer and subsamples it to produce a single output from that block : max, average, etc.

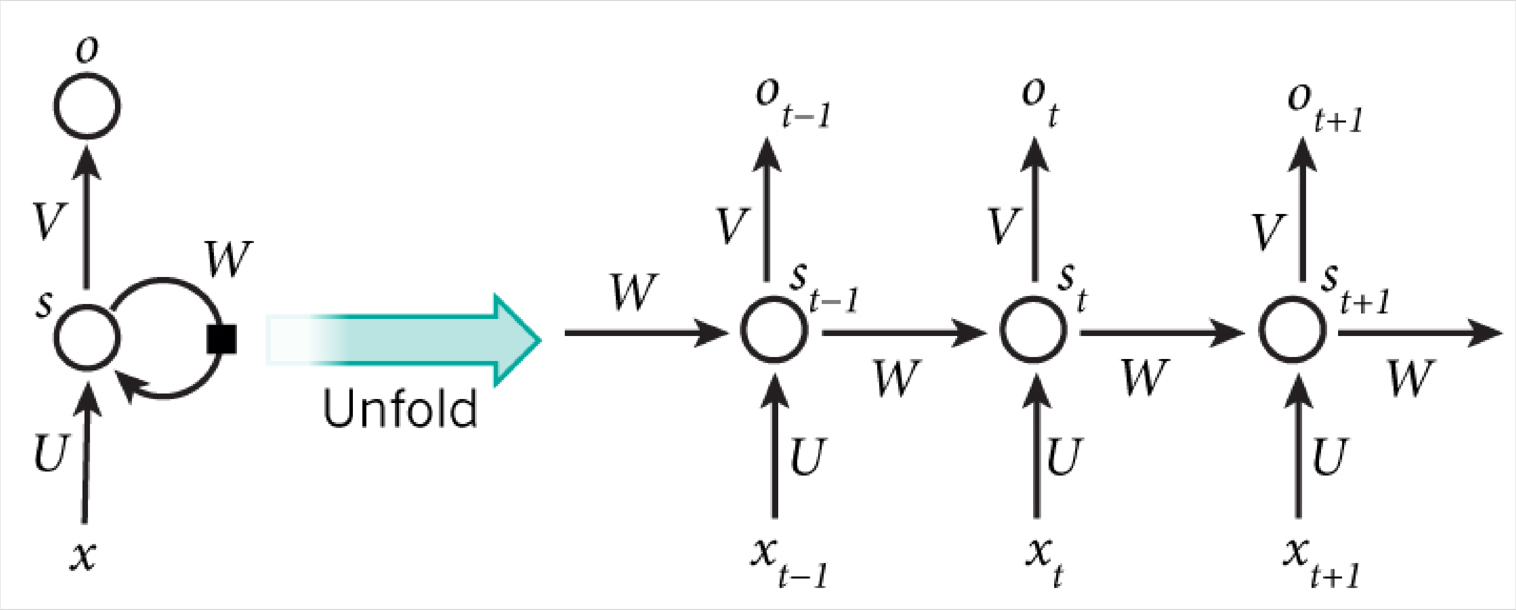


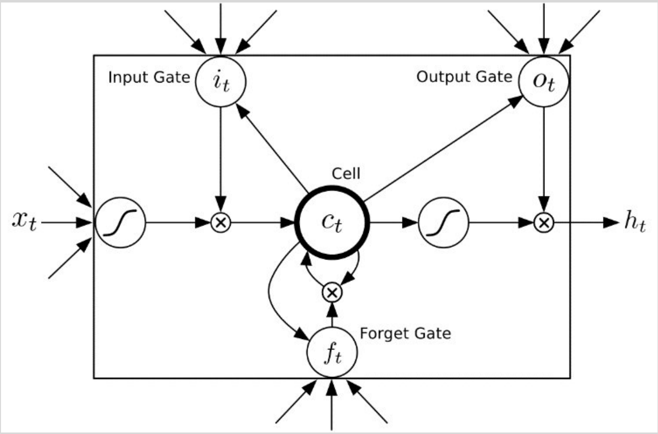
CNN properties

- CNN takes advantage of the sub-structure of the input
- Achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features.

- CNN are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units.

Recurrent Neural Network (RNN)





Thank You