Lecture 16: neural networks, deep learning

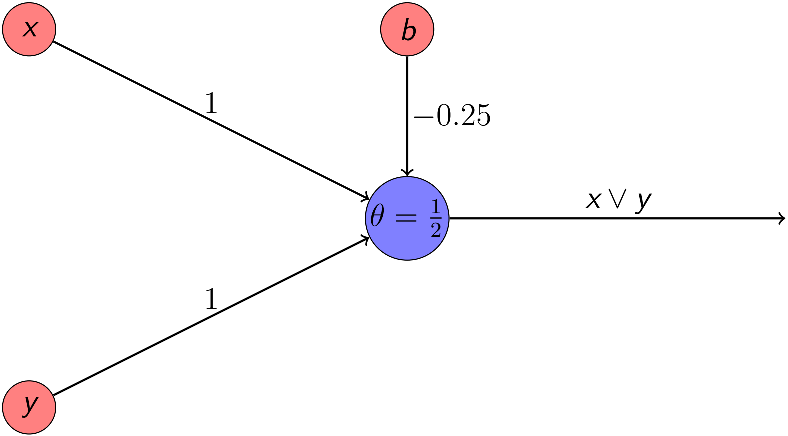
Instructor: Prof. Ganesh Ramakrishnan

Non-linear perceptron?

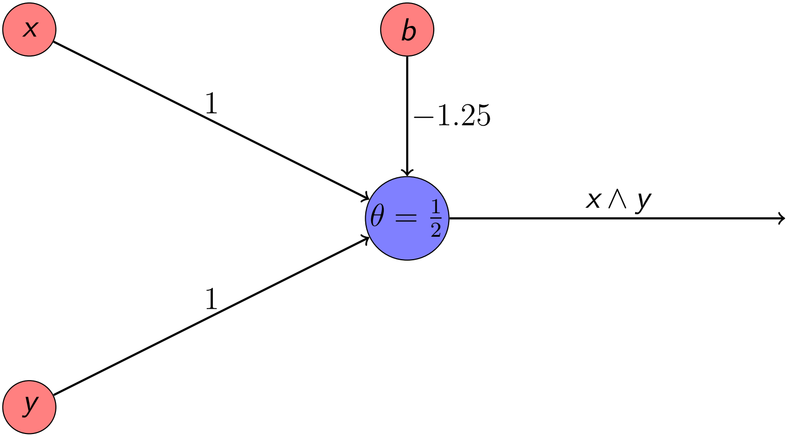
- Kernelized perceptron: $f(x) = sign\left(\sum_i \alpha_i y_i K(x, x_i) + b\right)$
  - INITIALIZE: α=zeroes()
  - REPEAT: for $< x_i, y_i >$
    - If $sign\left(\sum_j \alpha_j y_j K(x_j, x_j) + b\right) \neq y_i$
    - then, $\alpha_i = \alpha_i + 1$
    - endif
- Neural Networks: Cascade of layers of perceptrons giving you non-linearity
  - $sign\left((w^*)^T \phi(x)\right)$ replaced by $g\left((w^*)^T \phi(x)\right)$ where $g(s)$ is a
    1. step function: $g(s) = 1$ if $s \in [\theta, \infty)$ and $g(s) = 0$ otherwise OR
    2. sigmoid function: $g(s) = \frac{1}{1+e^{-s}}$

    Threshold changes as bias is changed.

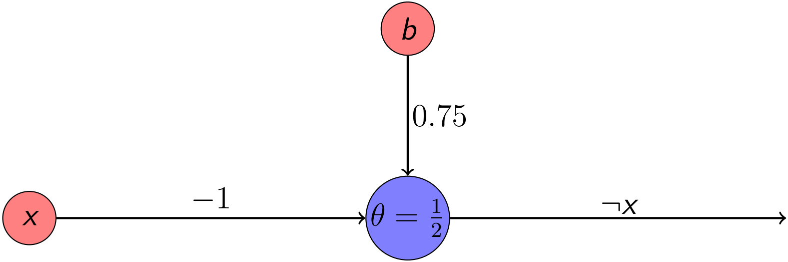
OR using perceptron



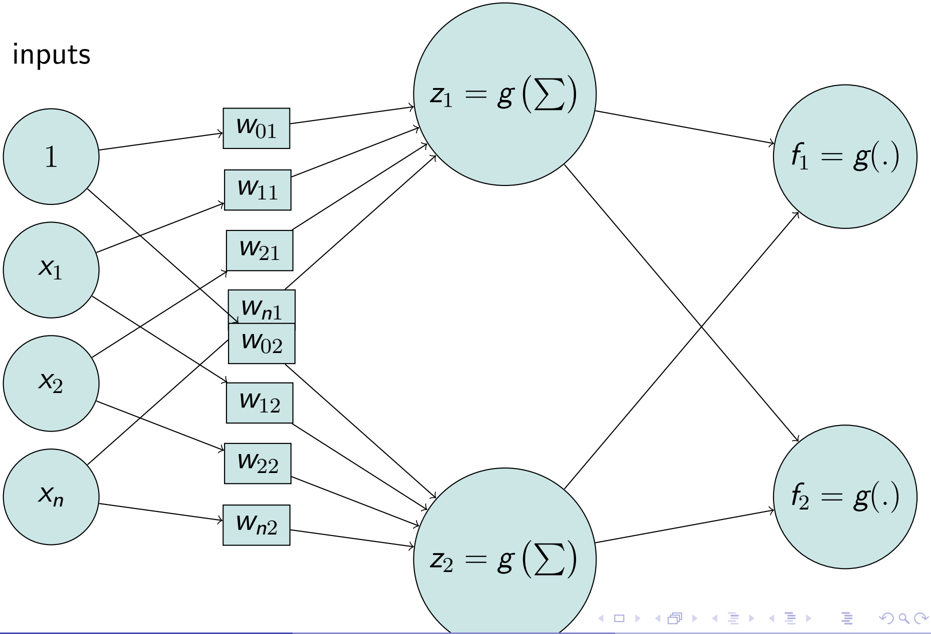
AND using perceptron



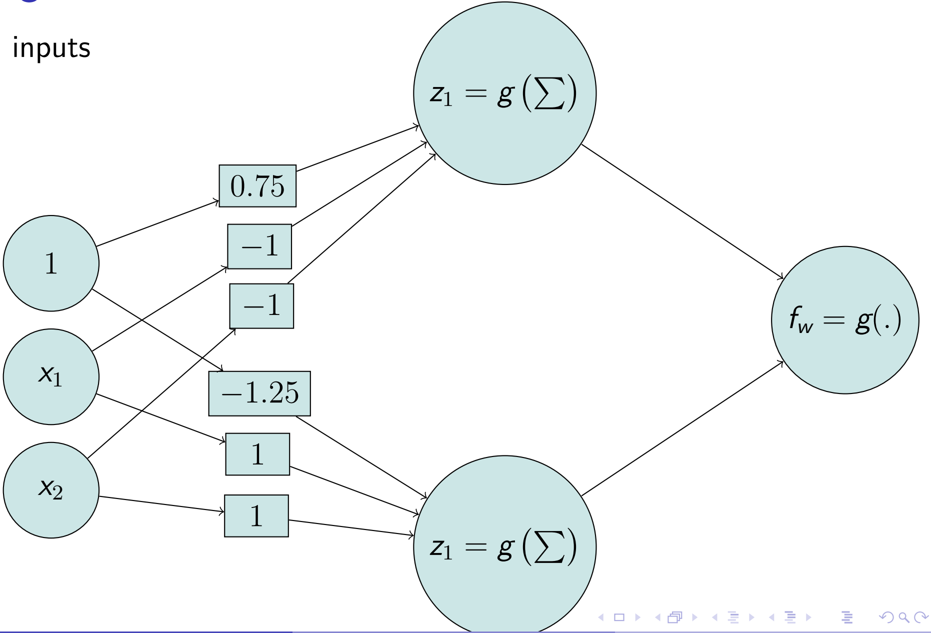
NOT using perceptron



Feed-forward Neural Nets



Eg: Feed-forward Neural Net for XOR



Training a Neural Network

STEP 0: Pick a network architecture

- Number of input units: Dimension of features $x^{(i)}$.
- Number of output units: Number of classes.
- Reasonable default: 1 hidden layer, or if $>$1 hidden layer, have same number of hidden units in every layer.
- Number of hidden units in each layer a constant factor (3 or 4) of dimension of x.
- Logistic Loss function:

$$E(w) = -\left[\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log f_w\left(x^{(i)}\right) + \left(1 - y^{(i)}\right)\log\left(1 - f_w\left(x^{(i)}\right)\right)\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2 \tag{1}$$