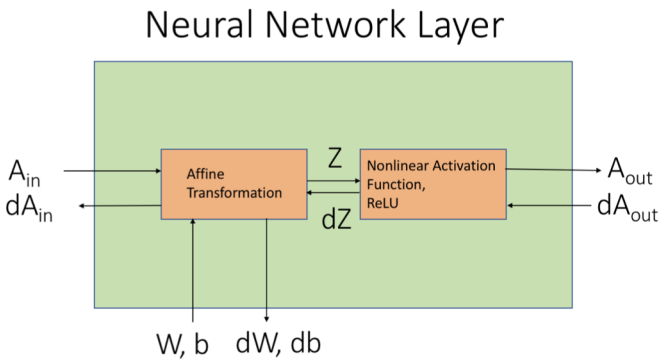
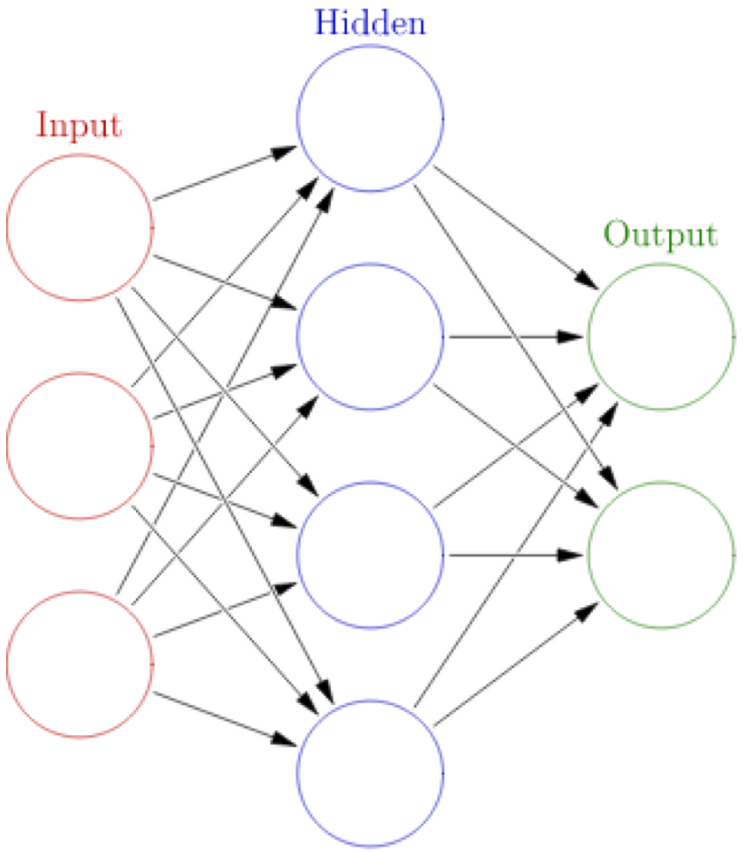
Deep Learning

Samuel Lou

April 19, 2018

Neural Network

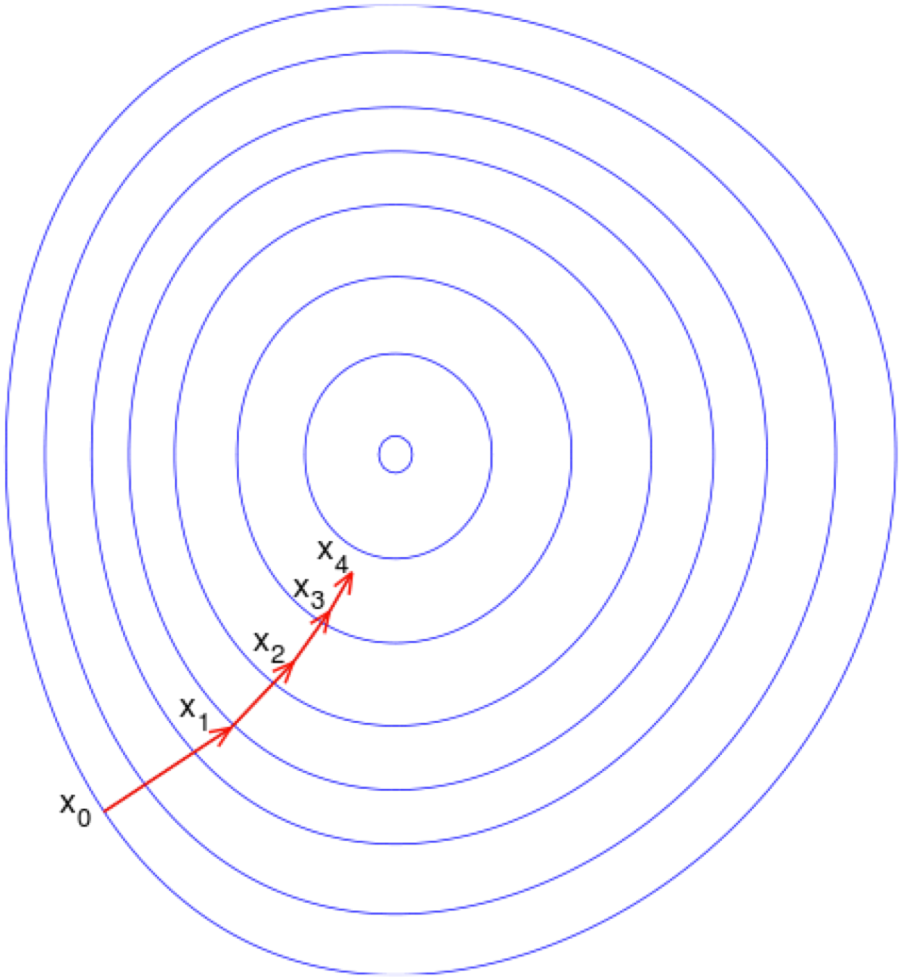




Loss Function: Cross Entropy

$$L = H(p, q) = -\frac{1}{n}\sum_{x} p(x)\log(q(x))$$

Optimization: Gradient Descent



Chain Rule: $\frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial F_{ij}} \frac{\partial F_{ij}}{\partial \cdots} \cdots \frac{\partial \cdots}{\partial W_i}$

One-layer neural network

First, compute an affine transform using parameters $\boldsymbol{W}$ and $\boldsymbol{b}$:

$$\boldsymbol{Z} = \boldsymbol{AW} + \boldsymbol{b}$$

$$\begin{bmatrix} a_{11} & \cdots & a_{1d} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nd} \end{bmatrix} \begin{bmatrix} w_{11} & \cdots & w_{1d'} \\ \vdots & \ddots & \vdots \\ w_{d1} & \cdots & w_{dd'} \end{bmatrix} + \begin{bmatrix} b_1 & \cdots & b_{d'} \\ \vdots & \ddots & \vdots \\ b_1 & \cdots & b_{d'} \end{bmatrix}$$

One-layer neural network: Backpropagation

$$Z_{ij} = \sum_{k=0}^{d-1} A_{ik} W_{kj} + b_j$$

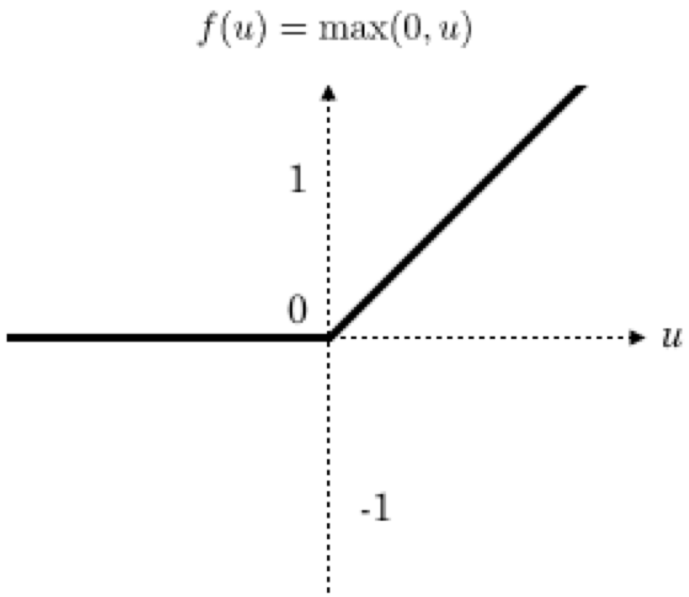
$$dW_{kj} = \frac{\partial L}{\partial W_{kj}} = \sum_{i=0}^{n-1} \frac{\partial L}{\partial Z_{ij}} \frac{\partial Z_{ij}}{\partial W_{kj}} = \sum_{i=0}^{n-1} dZ_{ij} A_{ik}$$

Nonlinearity: ReLU

Second, compute element-wise nonlinearity:

$$A_{ij} = \begin{cases} Z_{ij} & \text{if } Z_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

REctified Linear Unit



ReLU: Backpropagation

$$A_{ij} = \begin{cases} Z_{ij} & \text{if } Z_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$dZ_{ij} = \frac{\partial L}{\partial Z_{ij}} = \frac{\partial L}{\partial A_{ij}} \frac{\partial A_{ij}}{\partial Z_{ij}}$$

$$= \begin{cases} dA_{ij} & \text{if } Z_{ij} > 0 \\ 0 & \text{otherwise} \end{cases}$$

Cross Entropy and Softmax

$$L = H(y, \sigma)$$

$$= -\frac{1}{n}\sum_{i} y_i \log(\sigma_i)$$

With $\sigma_i = \frac{\exp(F_{ik})}{\sum_{k=0}^{C-1} \exp(F_{ik})}$, we have

$$L = -\frac{1}{n}\left(\sum_{i=0}^{n-1}\left(F_{iy_i} - \log\left(\sum_{k=0}^{C-1} \exp(F_{ik})\right)\right)\right)$$

Note: y_i in the above equation actually stands for the one hot encoding of y_i. For instance, if $y_i = 1$, then we should use the vector $[0, 1, 0]$ in place of y_i in the above equation.

Cross Entropy and Softmax: Backpropagation

$$L = -\frac{1}{n}\left(\sum_{i=0}^{n-1}\left(F_{iy_i} - \log\left(\sum_{k=0}^{C-1}\exp(F_{ik})\right)\right)\right)$$

$$dF_{ij} = \frac{\partial L}{\partial F_{ij}} = -\frac{1}{n}\left(1\{j = y_i\} - \frac{\exp(F_{ij})}{\sum_{k=0}^{C-1}\exp(F_{ik})}\right)$$

$$\frac{d}{dx}\log_e(x) = \frac{1}{x} \qquad \frac{d}{dx}e^x = e^x$$