Artificial intelligence in data science

Backpropagation

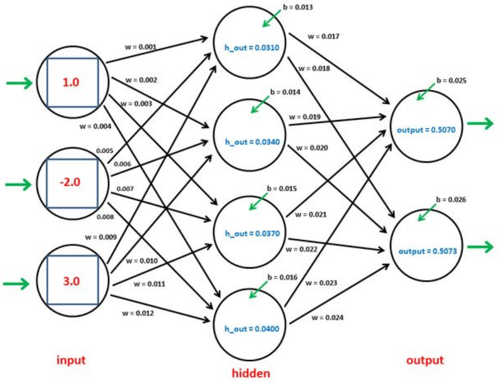
Janos Török

Department of Theoretical Physics

September 28, 2022

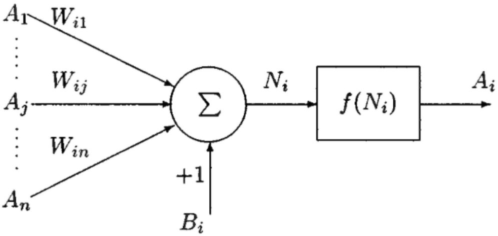
Fully connected neural networks

- Ideas from Piotr Skalski (practice), Pataki Bálint Ármin (lecture) and HMKCode (lecture)



Fully connected neural networks

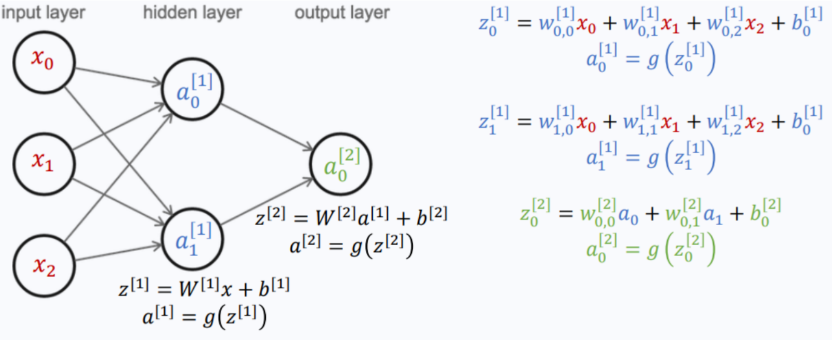
- Model:
  - Inputs (x_j) or for hidden layer l: A_j^{l-1}
  - Weight w_{ij}^l
  - Bias b_i^l
  - Weighted sum of input and bias: $z_i^l = \sum_j A_j^{l-1} w_{ij}^l + b_i^l$
  - Activation function (nonlinear) g: $A_i^l = g(z_i^l)$



Yang et el, 2000.

Feed forward

- Example



- We have an output, how to change weights and biases to achieve the desired output?
- Error L

Backpropagation

- $$\Delta W = -\alpha \frac{\partial L}{\partial W}$$
- W is a large three dimansional matrix
- Chain rule!

Geoffrey Hinton

FOLLOW

Emeritus Prof. Comp Sci, U.Toronto & Engineering Fellow, Google
Verified email at cs.toronto.edu - Homepage

machine learning   neural networks   artificial intelligence   cognitive science   computer science

TITLE	CITED BY	YEAR
Learning internal representations by error-propagation DE Rumelhart, GE Hinton, RJ Williams Parallel Distributed Processing: Explorations in the Microstructure of ...	63004 *	1986

Backpropagation

- Chain rule

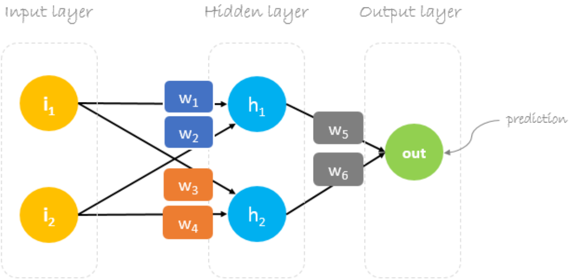
$$a_i = g(z_i) = g(w_{ij}a_j + b_i)$$

$$\frac{\partial L}{\partial w_{ij}} = \frac{\partial L}{\partial a_i}\frac{\partial a_i}{\partial z_i}\frac{\partial z_i}{\partial w_{ij}} = \frac{\partial L}{\partial a_i}g'(z_i)a_j$$

$$\frac{\partial L}{\partial a_i} = \sum_{l \in L}\frac{\partial L}{\partial a_l}\frac{\partial a_l}{\partial z_l}\frac{\partial z_l}{\partial a_i} = \sum_{l \in L}\frac{\partial L}{\partial a_l}g'(z_l)w_{li}$$

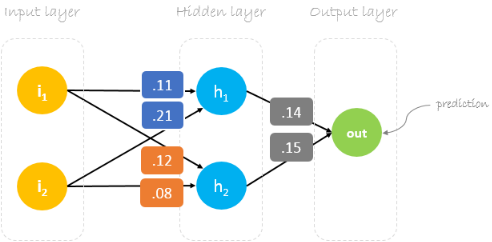
Backpropagation: Example

- From HMKCode
- Note that there is no activation function (it would just add one more step in the chain rule)



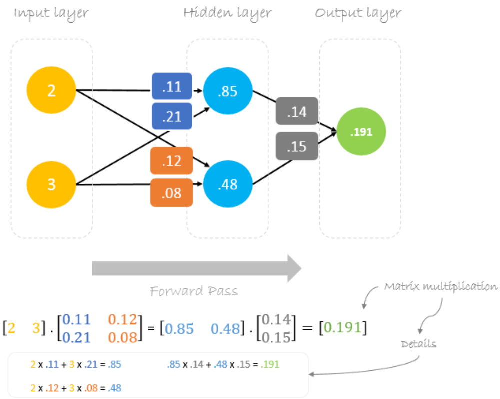
Backpropagation: Example

- Weights



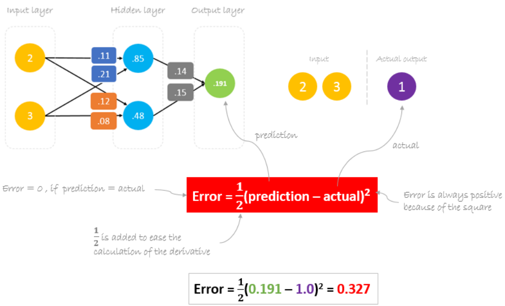
Backpropagation: Example

- Feedforward



Backpropagation: Example

- Error from the desired target



Backpropagation: Example

- Prediction function

$$\textbf{prediction} = out$$

$$\textbf{prediction} = (h_1)\, w_5 + (h_2)\, w_6$$

$h_1 = i_1 w_1 + i_2 w_2$

$h_2 = i_1 w_3 + i_2 w_4$

$$\textbf{prediction} = (i_1\, w_1 + i_2\, w_2)\, w_5 + (i_1\, w_3 + i_2\, w_4)\, w_6$$

to change ***prediction*** value,
we need to change ***weights***

Backpropagation: Example

- Gradient descent

Old weight

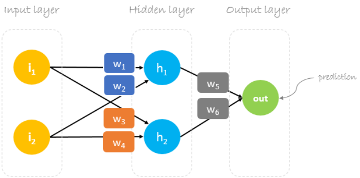
Derivative of Error with respect to weight

$$^{*}W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

New weight

Learning rate

$$^{*}W_6 = W_6 - a\left(\frac{\partial Error}{\partial W_6}\right)$$



Backpropagation: Example

- Chain rule

$$\frac{\partial Error}{\partial W_6} = \frac{\partial Error}{\partial prediction} * \frac{\partial prediction}{\partial W_6}$$

chain rule

Error = $\frac{1}{2}$(prediction – actual)^2

prediction = $(i_1 w_1 + i_2 w_2)\, w_5 + (i_1 w_3 + i_2 w_4)\, w_6$

$$\frac{\partial Error}{\partial W_6} = \frac{\frac{1}{2}(predictoin - actula)^2}{\partial prediciton} * \frac{\partial\, (i_1 w_1 + i_2 w_2)\, w_5 + (i_1 w_3 + i_2 w_4)\, w_6}{\partial W_6}$$

$$\frac{\partial Error}{\partial W_6} = 2 * \frac{1}{2}(\text{predictoin} - \text{actula}) \frac{\partial(predictoin - actula)}{\partial prediciton} * (i_1 w_3 + i_2 w_4)$$

$h_2 = i_1 w_3 + i_2 w_4$

$$\frac{\partial Error}{\partial W_6} = (\text{predictoin} - \text{actula}) * (h_2)$$

Δ = prediction – actual

delta

$$\frac{\partial Error}{\partial W_6} = \Delta h_2$$

Backpropagation: Example

- Chain rule

$$\frac{\partial Error}{\partial W_6} = \frac{\partial Error}{\partial prediction} * \frac{\partial prediction}{\partial W_6}$$

chain rule

$$Error = \frac{1}{2}(prediction - actual)^2$$

$$prediction = (i_1 w_1 + i_2 w_2)\, w_5 + (i_1 w_3 + i_2 w_4)\, w_6$$

$$\frac{\partial Error}{\partial W_6} = \frac{\frac{1}{2}(predictoin - actula)^2}{\partial prediciton} * \frac{\partial\, (i_1 w_1 + i_2 w_2)\, w_5 + (i_1 w_3 + i_2 w_4)\, w_6}{\partial W_6}$$

$$\frac{\partial Error}{\partial W_6} = 2 * \frac{1}{2}(predictoin - actula) \frac{\partial (predictoin - actula)}{\partial prediciton} * (i_1 w_3 + i_2 w_4)$$

$$h_2 = i_1 w_3 + i_2 w_4$$

$$\frac{\partial Error}{\partial W_6} = (predictoin - actula) * (h_2)$$

$$\Delta = prediction - actual$$

delta

$$\frac{\partial Error}{\partial W_6} = \Delta h_2$$

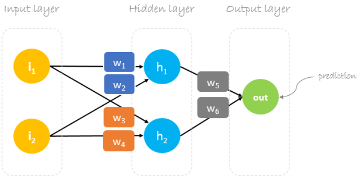
$$^*W_6 = W_6 - a\, \Delta h_2$$

Backpropagation: Example

- Chain rule

$$^*W_6 = W_6 - a\,\Delta h_2$$

$$^*W_5 = W_5 - a\,\Delta h_1$$



Backpropagation: Example

- Chain rule

$$\frac{\partial Error}{\partial W_1} = \frac{\partial Error}{\partial prediction} * \frac{\partial prediction}{\partial h_1} * \frac{\partial h_1}{\partial W_1}$$

chain rule

Error $= \frac{1}{2}(\text{prediction} - \text{actual})^2$

prediction $= (h_1)\, w_5 + (h_2)\, w_6$

$h_1 = i_1 w_1 + i_2 w_2$

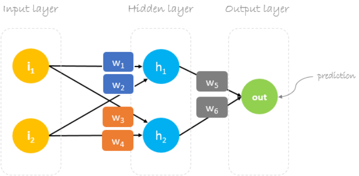
$$\frac{\partial Error}{\partial W_1} = \frac{\partial \frac{1}{2}(predictoin - actula)^2}{\partial prediciton} * \frac{\partial\, (h_1)\, w_5 + (h_2)\, w_6}{\partial h_1} * \frac{\partial\, i_1 w_1 + i_2 w_2}{\partial w_1}$$

$$\frac{\partial Error}{\partial W_1} = 2 * \frac{1}{2}(\text{predictoin} - \text{actula})\, \frac{\partial (predictoin - actula)}{\partial prediciton} * (w_5) * (i_1)$$

$$\frac{\partial Error}{\partial W_1} = (\text{predictoin} - \text{actula}) * (w_5 i_1)$$

Δ = prediction – actual ← delta

$$\frac{\partial Error}{\partial W_1} = \Delta\, w_5 i_1$$



Backpropagation: Example

- Summarized

updated weights

$$^{*}w_6 = w_6 - a\,(h_2 \,.\, \Delta)$$

$$^{*}w_5 = w_5 - a\,(h_1 \,.\, \Delta)$$

$$^{*}w_4 = w_4 - a\,(i_2 \,.\, \Delta w_6)$$

$$^{*}w_3 = w_3 - a\,(i_1 \,.\, \Delta w_6)$$

$$^{*}w_2 = w_2 - a\,(i_2 \,.\, \Delta w_5)$$

$$^{*}w_1 = w_1 - a\,(i_1 \,.\, \Delta w_5)$$

Backpropagation: Example

- Summarized in matrix form

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - a\,\Delta \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \begin{bmatrix} ah_1\Delta \\ ah_2\Delta \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - a\,\Delta \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} . \begin{bmatrix} w_5 & w_6 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \begin{bmatrix} a\, i_1\Delta w_5 & a\, i_1\Delta w_6 \\ a\, i_2\Delta w_5 & a\, i_2\Delta w_6 \end{bmatrix}$$

Backpropagation: Multiple data points

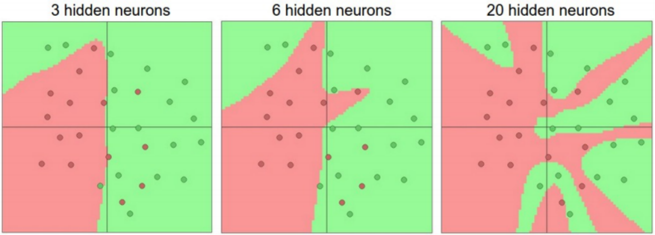
- Generally Δ is a vector, with the dimension of the number of training data points.
- The error can be the average of the error, so repeate the equations below for all training points and average the changes (the part after a)
- Fortunately numpy does not care about the number of dinemsions, so insted of the multiplication in the right matrices we can use dot product.

$$\begin{bmatrix} w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - a\,\boldsymbol{\Delta} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} w_5 \\ w_6 \end{bmatrix} - \begin{bmatrix} a h_1 \boldsymbol{\Delta} \\ a h_2 \boldsymbol{\Delta} \end{bmatrix}$$

$$\begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - a\,\boldsymbol{\Delta} \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} . \begin{bmatrix} w_5 & w_6 \end{bmatrix} = \begin{bmatrix} w_1 & w_3 \\ w_2 & w_4 \end{bmatrix} - \begin{bmatrix} a\, i_1 \boldsymbol{\Delta} w_5 & a\, i_1 \boldsymbol{\Delta} w_6 \\ a\, i_2 \boldsymbol{\Delta} w_5 & a\, i_2 \boldsymbol{\Delta} w_6 \end{bmatrix}$$

How many layers?

- Neural network with at least one hidden layer is a universal approximator (can represent any function).



Do Deep Nets Really Need to be Deep? Jimmy Ba, Rich Caruana,