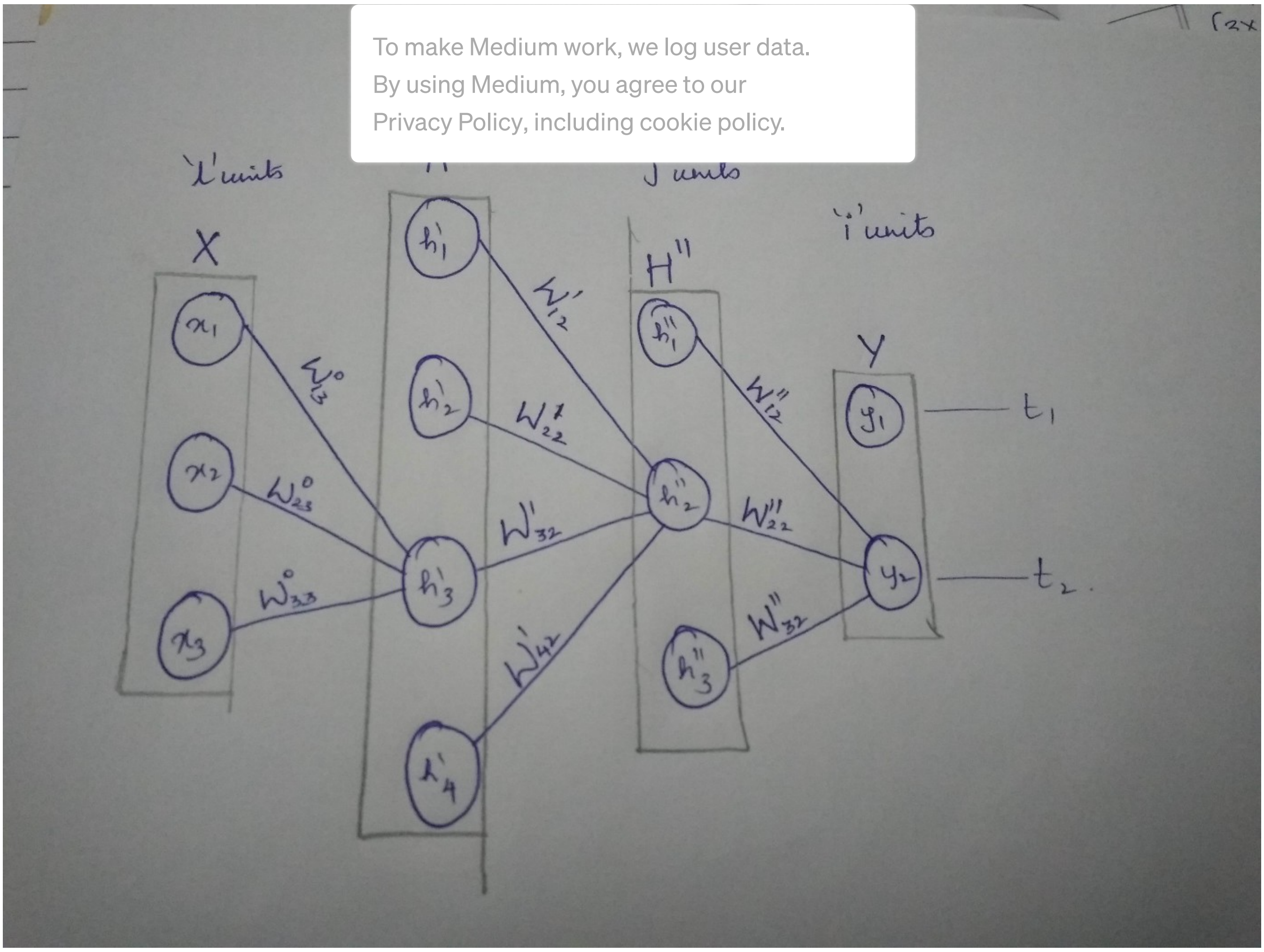
Chetan Patil

12 Followers About Follow

Derivation of Back Propagation with Cross Entropy

Chetan Patil Dec 29, 2018 · 2 min read

In order to understand the Back Propagation algorithm, we first need to understand some basic concepts such as Partial Derivatives, chain rule, Cross Entropy loss, Sigmoid function and Softmax function.



A two layered Neural Network with sigmoid activations

Assuming we have already forward-passed the inputs to get some outputs at the last layer Y, we will have to calculate the loss function E and propagate the loss to all the preceding layers by changing the weights associated with each of the layers.

$$E = -\sum_i^n \left[t_i \cdot \log(y_i) + (1-t_i)\log(1-y_i)\right], \quad y_i = \frac{e^{s_i''}}{\sum_c^n e^{s_c''}}$$

Consider $\frac{\partial y_1}{\partial s_1''} = \frac{\partial}{\partial s_1''}\left[\frac{e^{s_1''}}{e^{s_1''}+e^{s_2''}}\right]$

$$= \frac{(e^{s_1''}+e^{s_2''})\, e^{s_1''} - e^{s_1''} \cdot e^{s_1''}}{(e^{s_1''}+e^{s_2''})^2} = \frac{e^{s_1''}\left[\cancel{e^{s_1''}} + e^{s_2''} - \cancel{e^{s_1''}}\right]}{(e^{s_1''}+e^{s_2''})^2}.$$

$$\frac{\partial y_1}{\partial s_1''} = \frac{e^{s_1''}}{e^{s_1''}+e^{s_2''}} \cdot \left[1 - \frac{e^{s_1''}}{e^{s_1''}+e^{s_2''}}\right] = y_1(1-y_1).$$

$\partial s_1 \quad (e \; + \ldots$

$$\frac{\partial E}{\partial W''_{ji}} = \frac{\partial E}{\partial y_i}\,\frac{\partial y_i}{\partial s''_i}\,\frac{\partial s_i}{\partial W''_{ji}}$$

$$= \left(\frac{-t_i}{y_i} + \frac{1-t_i}{1-y_i}\right) y_i(1-y_i)\cdot h''_j\,.$$

$$= \frac{-t_i(1-y_i)+y_i(1-t_i)}{\cancel{y_i(1-y_i)}}\cdot \cancel{y_i(1-y_i)}\; h''_j$$

$$\frac{\partial E}{\partial W''_{ji}} = h''_j\cdot(y_i-t_i)$$

example

$$\frac{\partial E}{\partial W''_{32}} = h''_3\cdot(y_2-t_2)$$

Weight gradient of a weight connecting the third unit in second layer and second unit in the output layer using softmax activation.

Knowing the cross entropy loss E and the softmax activation " yi ' , we can calculate the change in loss with respect to any weight connecting the output layer using the chain rule of partial derivatives. Intuitively, we can even find the weight gradients for the whole layer using the matrix notation shown below.

$$\frac{\partial E}{\partial W''} = H''\cdot(Y-t)^T$$

Matrix notation of weights connecting the second hidden layer and output layer.

Now, finding weight gradients connecting

H' and

WKT, $h''_j = \frac{1}{1+e^{-s'_j}}$ where $s'_j = \sum_k h'_k \cdot W'_{kj}$.

example.

$$h''_2 = \text{sigmoid}\left(\underbrace{h'_1 W_{12} + h'_2 W_{22} + h'_3 W_{32} + h'_4 W_{42}}_{s'_2}\right)$$

Consider

$$\frac{\partial E}{\partial W'_{kj}} = \sum_{i=1}^{nout} \frac{\partial E}{\partial s''_i} \cdot \frac{\partial s''_i}{\partial h''_j} \cdot \frac{\partial h''_j}{\partial s'_j} \cdot \frac{\partial s'_j}{\partial W'_{kj}}$$

$$= \sum_{i=1}^{nout} (y_i - t_i) \cdot W''_{ji} \cdot h''_j (1 - h''_j) \cdot h'_k$$

example

$$\frac{\partial E}{\partial W'_{41}} = \left[(y_1 - t_1) W''_{11} + (y_2 - t_2) W''_{12}\right] \cdot h''_1 (1 - h''_1) \cdot h'_4 .$$

Weight gradient of a weight connecting the fourth unit in first hidden layer to the first unit in the second hidden layer using sigmoid activation.

In matrix notation

$$\frac{\partial E}{\quad} = H^1 . \left\{ \left(W'' . [Y(1-t)]\right) * H^2 * (1 - H^2) \right\}^T$$

$\partial W'$

Matri ayers

Now finding the weight gradients connecting X and H'

WKT, $h'_k = \frac{1}{1+e^{-s^0_k}}$ where $s^0_k = \sum_l x_l \cdot W^0_{lk}$.

example

$$h'_3 = \text{sigmoid}\left(\underbrace{x_1 \cdot W^0_{13} + x_2 \cdot W^0_{23} + x_3 W^0_{33}}_{s^0_3}\right)$$

Consider

$$\frac{\partial E}{\partial W^0_{lk}} = \sum_i^{nout} \frac{\partial E}{\partial y_i}\cdot\frac{\partial y_i}{\partial s''_i}\cdot\frac{\partial s''_i}{\partial h''_j}\cdot\frac{\partial h''_j}{\partial s'_j}\cdot\frac{\partial s'_j}{\partial h'_k}\cdot\frac{\partial h'_k}{\partial s^0_k}\cdot\frac{\partial s^0_k}{\partial W^0_{lk}}.$$

$$= \sum_i y_i(1-y_i)\cdot W''_{ji} \cdot \sum_j h''_j (1-h''_j) \cdot W'_{kj} \cdot h'_k(1-h'_k)\cdot x_l.$$

On rearranging.

$$= \sum_j \left[h''_j(1-h''_j) \cdot \left(\sum_i y_i(1-t_i)\cdot W''_{ji}\right)\right] \cdot W'_{kj} \cdot h'_k(1-h'_k)\cdot x_l.$$

Weight gradient of a weight connecting a unit L in input layer to the unit K in the first hidden layer using sigmoid activation.

$$\partial W^0 = X \cdot \left(\left(W' \cdot \left[(H''(1-H'')) * (W'' \cdot Y*(1-t))\right]\right) * (H' * (1-H'))\right)^T$$

Matrix notation of weights connecting the input layer and first hidden layer.

Machine Learning Deep Learning Backpropagation