Assignment 4

Mohit Garg (2020AM10657)

February 14, 2023

Nomenclature

b^l	bias for the l$_{th}layer$
J	Cross-Entropy Loss Function
w^l	weight matrix for the l$_{th}layer$
δ_n^l	$\frac{\partial J}{\partial z_n^l}$

Contents

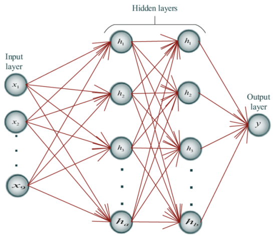

Figure 1: Neural network with 2 hidden layers

Derive systematically the steps involving the backpropagation of a neural network considered in the class with two hidden layers using the cross-entropy loss function. The activation function for each layer can be considered to be ReLU

1 Solution

1.1 Defining neural network and front propogation

Let $w^{(1)}$, $w^{(2)}$ and $w^{(3)}$ be three weight matrices for layer 1 , 2 , 3 where 1 , 2 are hidden layers and 3 is output layer.
Let $b^{(1)}$, $b^{(2)}$ and $b^{(3)}$ be three bias vectors for layer 1 , 2 , 3
X serves as input / 0_{th} layer
Let us define relu activation function and its derivative as h(x)

$$h(x) = maximum(0, x) \tag{1}$$

$$h'(x) = \frac{sign(x) + 1}{2} \tag{2}$$

Now , $a^{(1)}$,$a^{(2)}$,$z_m^{(1)}$,$z_m^{(2)}$,$z_m^{(3)}$ will be determined by:

$$z_m^{(1)} = \sum_i w_{im}^{(1)} X + b_m^{(1)} \tag{3}$$

$$a_m^{(1)} = h(z_m^{(1)}) \tag{4}$$

$$z_m^{(2)} = \sum_i w_{im}^{(2)} a_i^{(1)} + b_m^{(2)} \tag{5}$$

$$a_m^{(2)} = h(z_m^{(2)}) \tag{6}$$

$$z_m^{(3)} = \sum_i w_{im}^{(3)} a_i^{(2)} + b_m^{(3)} \tag{7}$$

If we apply Relu to last layer also then,

$$a_m^{(3)} = h(z_m^{(3)}) \tag{8}$$

Note that i for each layer in sum will be determined by size of layer i.e. number of neurons in it. Also define $\delta_n^l = \frac{\partial J}{\partial z_n^l}$. We need to find $\frac{\partial J}{\partial W^L}$ and $\frac{\partial J}{\partial b^L}$ for layers 3 through 0

1.2 Backpropogation in General layer

Now consider

$$\frac{\partial J}{\partial w^l_{mn}} = \frac{\partial J}{\partial z^l_n} . \frac{\partial z^l_n}{\partial w^l_{mn}} = \tag{9}$$

$$\frac{\partial J}{\partial w^l_{mn}} = \delta^{(l)}_n . \frac{\partial z^l_n}{\partial w^l_{mn}} \tag{10}$$

Now $\frac{\partial z^l_n}{\partial w^l_{mn}}$ is a^{l-1}_m. Using this, we get

$$\frac{\partial J}{\partial w^l_{mn}} = \delta^{(l)}_n . a^{l-1}_m \tag{11}$$

Similarly, for b^l_n , we have

$$\frac{\partial J}{\partial b^l_n} = \delta^{(l)}_n \tag{12}$$

Now

$$\delta^{(l)}_n = \frac{\partial J}{\partial z^l_n} = \sum_m \frac{\partial J}{\partial z^{l+1}_m} . \frac{\partial z^{l+1}_m}{\partial z^l_n} = \sum_m \delta^{l+1}_m . \frac{\partial z^{l+1}_m}{\partial z^l_n} \tag{13}$$

With $\frac{\partial z^{l+1}_m}{\partial z^l_n} = \frac{\sum_i w^{(l+1)}_{im} * a^{(l)}_i + b^{(l)}_m}{\partial z^l_n}$, we get

$$\frac{\partial z^{l+1}_m}{\partial z^l_n} = w^{l+1}_{nm} . \frac{\partial h(z^l_n)}{\partial z^l_n} = w^{l+1}_{nm} . h'(z^l_n) \tag{14}$$

$$\delta^{(l)}_n = \sum_m \delta^{l+1}_m . w^{l+1}_{nm} . h'(z^l_n) = h'(z^l_n) . \sum_m \delta^{l+1}_m . w^{l+1}_{nm} \tag{15}$$

Reducing our equations to vector form,

$$\delta^l = \frac{\partial J}{\partial z^l} = h'(z^l) . W^{l+1} \delta^{l+1} \tag{16}$$

1.3 Last layer

If we use relu in last layer then we get

$$\delta^{(3)}_n = h'(z^{(3)}_n) . - \frac{y_n}{a^{(3)}_n} \tag{17}$$

This is using cross entropy cost function,

$$J = \sum_{i=1}^{n} \sum_{j=1}^{m} -y^{(i)}_j ln(a^{(L)}_j) \tag{18}$$

which can be written for ease:

$$J = -\sum_1^n y_m . ln(a^{(3)}_m) \tag{19}$$

we get

$$\delta^{(3)}_n = \frac{\partial J}{\partial z^{(3)}_n} = \sum_m \frac{\partial J}{\partial a^{(3)}_m} . \frac{\partial a^{(3)}_m}{\partial z^{(3)}_n} \tag{20}$$

We can write

$$\frac{\partial J}{\partial a^{(3)}_n} = \frac{\partial (-\sum_1^n y_m . ln(a^{(3)}_m))}{\partial a^{(3)}_n} = -\frac{\partial (y_n . ln(a^{(3)}_n))}{\partial a^{(3)}_n} = -\frac{y_n}{a^{(3)}_n} \tag{21}$$

So using $a^{(3)}_n = h(z^{(3)}_n)$,our equation reduces to

$$\delta^{(3)}_n = h'(z^{(3)}_n) . - \frac{y_n}{a^{(3)}_n} \tag{22}$$

1.4 Summary in the end

$$\delta^{(3)} = h'(z^{(3)}). - \frac{y}{a^{(3)}} \tag{23}$$

$$\delta^{(2)} = h'(z^2).w^{(3)}\delta^3 \tag{24}$$

$$\delta^{(1)} = h'(z^1).w^{(2)}\delta^2 \tag{25}$$

$$\frac{\partial J}{\partial w^3} = a^{(2)} * (\delta^{(3)})^T \tag{26}$$

$$\frac{\partial J}{\partial w^2} = a^{(1)} * (\delta^{(2)})^T \tag{27}$$

$$\frac{\partial J}{\partial w^1} = X * (\delta^{(1)})^T \tag{28}$$

$$\frac{\partial J}{\partial b^3} = (\delta^{(3)}) \tag{29}$$

$$\frac{\partial J}{\partial b^2} = (\delta^{(2)}) \tag{30}$$

$$\frac{\partial J}{\partial b^1} = (\delta^{(1)}) \tag{31}$$

where

$$h'(x) = \frac{sign(x) + 1}{2} \tag{32}$$

1.5 Implementation

The proof helps us only arrive at the equations; the algorithm is what employs them. It considers one example at a time and goes as follows:
1- Find a^L and z^L for layers 0 through 3 by feeding an example into the network. This is known as the “forward pass”.
2- Compute δ^L for layers 3 through 0 using the formulae for $\delta^{(3)}$, δ^L respectively.
3- Simultaneously compute $\frac{\partial J}{\partial W^L}$ and $\frac{\partial J}{\partial b^L}$ for layers 3 through 0 as once we have δ^L we can find both of these. (Use the last two equations.) This is known as the “backward pass”.
4- Repeat for more examples until the weights and biases of the network can be updated through gradient descent (depends on your batch size)

1.6 Extra reference for softmax in last layer

But for accurate results, we implement softmax at output layer. Now, we’ll be only dealing with the last layer in the derivation so we might as well drop the superscript for now and keep in mind that we’re targeting the last layer whenever we write a, z or δ
Let’s start with categorical cross-entropy. For this loss function our y’s are one-hot encoded to denote the class our image (or whatever) belongs to. Thus for any x, y is of length equal to the number of classes and the last layer in our model has a neuron for each class. We use Softmax in our last layer to get the probability of x belonging to each of the classes. These probabilities sum to 1. The loss function is

$$J = -\sum_{1}^{n} y_m.ln(a_m^{(3)}) \tag{33}$$

with the activation of the nth neuron in the last layer being Softmax Activation

$$a_n = h(z_n) = \frac{e^{z_n}}{\sum_m e^{z_m}} \tag{34}$$

Notice that the activation of the nth neuron depends on the pre-activations of all other neurons in the layer. This would've not been the case if the last layer involved Sigmoid or ReLU activations. On this account, to find for some neuron in the last layer we use the chain-rule by writing

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_m \frac{\partial J}{\partial a_m} . \frac{\partial a_m}{\partial z_n} \tag{35}$$

Considering m=n and m!=n then adding,It will simplify things a bit

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m!=n} \frac{\partial J}{\partial a_m} . \frac{\partial a_m}{\partial z_n} + \frac{\partial J}{\partial a_n} . \frac{\partial a_n}{\partial z_n} \tag{36}$$

We can write

$$\frac{\partial J}{\partial a_n} = \frac{\partial(-\sum_1^n y_m . ln(a_m^{(3)}))}{\partial a_n} = \frac{\partial(y_n . ln(a_n^{(3)}))}{\partial a_n} = -\frac{y_n}{a_n} \tag{37}$$

We have

$$\frac{\partial a_n}{\partial z_n} = \frac{\partial(e^{z_n}/\sum_m e^{z_m})}{\partial z_n} = a_n . (1 - a_n) \tag{38}$$

Now multiplying both of our results we and plugging in the original equation we get

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m!=n} \frac{\partial J}{\partial a_m} . \frac{\partial a_m}{\partial z_n} - y_n . (1 - a_n) \tag{39}$$

For the remaining sum, m not equal to n

$$\frac{\partial a_m}{\partial z_n} = \frac{\partial(e^{z_m}/\sum_{m'} e^{z_{m'}})}{\partial z_n} = \frac{e^{z_m}}{\sum_{m'} e^{z_{m'}}} . - \frac{\partial \sum_{m'} e^{z_{m'}}/\partial z_n}{\sum_{m'} e^{z_{m'}}} = -a_m a_n \tag{40}$$

Now multiplying both results, we get

$$\frac{\partial J}{\partial a_m} . \frac{\partial a_m}{\partial z_n} = -\frac{y_m}{a_m} . - a_m a_n = y_m a_n \tag{41}$$

and propagating that back to the original equation we get

$$\delta_n = \frac{\partial J}{\partial z_n} = \sum_{m!=n} \frac{\partial J}{\partial a_m} . \frac{\partial a_m}{\partial z_n} - y_n . (1 - a_n) = \sum_{m!=n} y_m a_n - y_n . (1 - a_n) = \sum_m y_m a_n - y_n = a_n \sum_m y_m - y_n \tag{42}$$

Since y is one-hot vector, its submission should be 1 so we get

$$\delta_n = \frac{\partial J}{\partial z_n} = a_n - y_n \tag{43}$$

or in vector form after reimposing the superscript (H to denote the last layer)

$$\delta^{(3)} = \frac{\partial J}{\partial z^{(3)}} = a^{(3)} - y \tag{44}$$