Assignment I

Deep Learning for Computer Vision(CS776)

Indian Institute of Technology,Kanpur, INDIA-208 016

Piyush Gangle (21111046)
piyushg21@iitk.ac.in

February 3, 2022

Contents

1 CNN architecture used

Neural network used has input layer with 512 neurons, single hidden layer is used having 64 neurons and output layer 10 neurons.ReLU activation funtion is used for hidden layers and Softmax activation function is used for output layers.

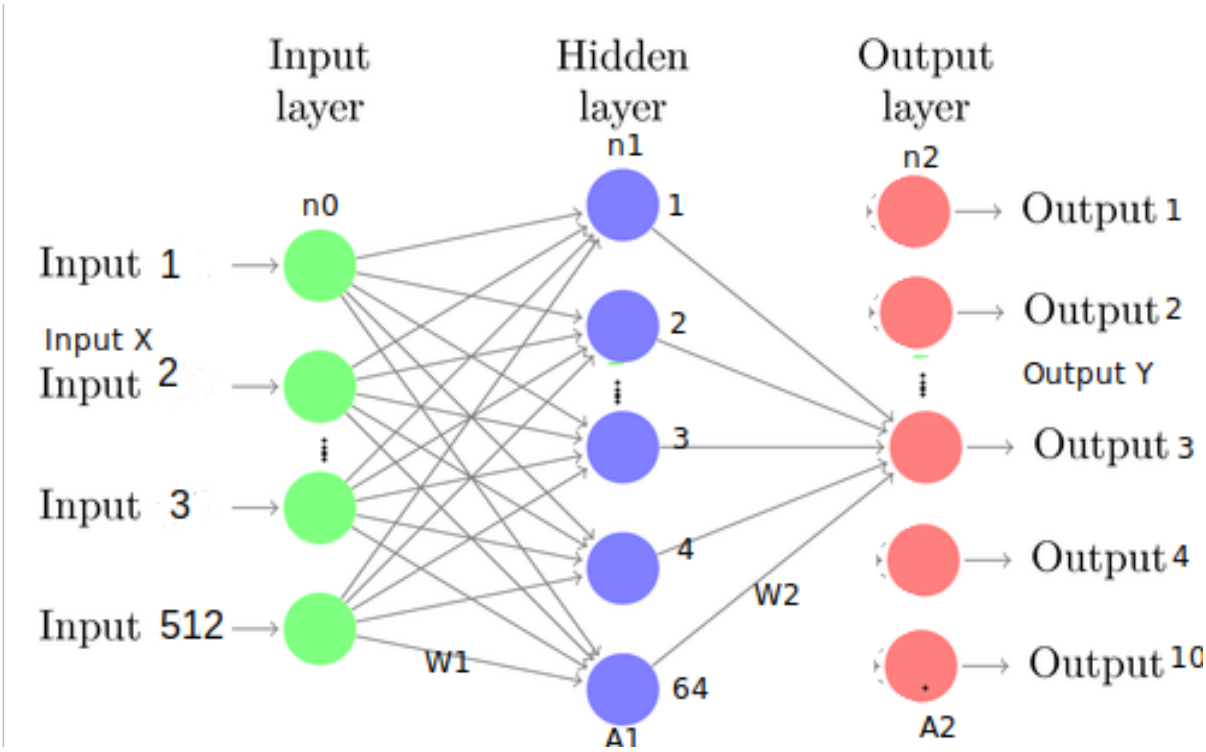

Figure 1: Diagrammatic representation of Neural network

1.1 Notations used

1. m: total number of obrservations = 50000
2. n_0: number of neurons in input layer = 512
3. n_1: number of neurons in hidden layer = 64
4. n_2: number of neurons in output layer = 10
5. Shape of W_1 : $(n_1, n_0) = 64 \times 512$
6. Shape of W_2 : $(n_2, n_1) = 10 \times 64$
7. Shape of B_1 : $(n_1, 1) = 64 \times 1$
8. Shape of B_2 : $(n_2, 1) = 10 \times 1$

2 Math used to design Neural network

2.1 Forward Propagation

- $Z_1 = W_1 * X + B_1$
- $A_1 = f_1(Z_1)$
- note: $f_1()$: ReLU activation function
- note: $f_1'()$: Derivative of ReLU activation function
- $Z_2 = W2 * A_1 + B_2$

2.2 Softmax

- $A_2 = Softmax(Z_2)$
- Softmax: $a_i = \frac{e^{z_i}}{\sum_{i=k}^{n} e^{z_k}}$

2.3 Cost function

- $Loss = -\sum_{i=k}^{n}[y_k * log(a_k)]$
- $Cost = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{n}[y_k * log(a_k)]$

2.4 Backpropagation

- $dZ_2 = (A_2 - Y)$
- $dW_2 = \frac{1}{m}.dZ_2.A_1^T$
- $dB_2 = \frac{1}{m}.sum(dZ_2, 1)$
- $dZ_1 = W_2^T.dZ_2 * f_1^{|}(Z_1)$
- $dW_1 = \frac{1}{m}.dZ_1.X^T$
- $dB_1 = \frac{1}{m}.sum(dZ_1, 1)$

2.5 Updating Parameters

- $W_2 = W_2 - \alpha * \frac{\partial Cost}{\partial W_2}$
- $B_2 = B_2 - \alpha * \frac{\partial Cost}{\partial B_2}$
- $W_1 = W_1 - \alpha * \frac{\partial Cost}{\partial W_1}$
- $B_1 = B_1 - \alpha * \frac{\partial Cost}{\partial B_1}$

3 Backpropagation equation derivation

- Loss $= -\sum_{i=k}^{n}[y_k * log(a_k)]$
- $a_i = \frac{e^{z_i}}{\sum_{i=k}^{n} e^{z_k}}$
- Our purpose is to find $dZ_2 = \frac{\partial loss}{dw_2}$ for multi class classification in our case we have 10 diffrent classes. to find $\frac{\partial loss}{dw_2}$, we need to find $\frac{\partial a_i}{\partial z_j}$ but $\frac{\partial a_i}{\partial z_j}$ will be different for 2 cases.

  1. Case 1 : $i \neq q$
     $\longrightarrow \frac{\partial a_i}{\partial z_j} = \frac{\partial}{\partial z_j}[\frac{e^{z_i}}{e^{z_j}+\sum_{k\neq j}[e^{z_k}]}]$
     $\longrightarrow \frac{(e^{z_j}+\sum_{k\neq j}[e^{z_k}])\frac{\partial}{\partial z_j}(e^{z_i}) - e^{z_i}\frac{\partial}{\partial z_j}[e^{z_j}+\sum_{k\neq j}e^{z_k}]}{(e^{z_j}+\sum_{k\neq j}e^{z_k})^2}$
     $\longrightarrow \frac{\partial a_i}{\partial z_j} = \frac{0 - e^{z_i}e^{z_j}}{(\sum_{k\neq i}e^{z_k})}$
     $\longrightarrow -\frac{e^{z_i}}{(\sum_{k\neq i}e^{z_k})} \times \frac{e^{z_j}}{(\sum_{k\neq i}e^{z_k})}$
     $\longrightarrow = -a_i.a_j$
  2. Case 2 : $i = q$
     $\longrightarrow \frac{\partial a_i}{\partial z_i} = \frac{\partial}{\partial z_i}[\frac{e^{z_i}}{e^{z_i}+\sum_{k=i}[e^{z_k}]}]$
     $\longrightarrow \frac{(e^{z_i}+\sum_{k=i}[e^{z_k}])\frac{\partial}{\partial z_j}(e^{z_i}) - e^{z_i}\frac{\partial}{\partial z_i}[e^{z_i}+\sum_{k=i}e^{z_k}]}{(e^{z_i}+\sum_{k=j}e^{z_k})^2}$
     $\longrightarrow \frac{(e^{z_j}+\sum_{k=j}[e^{z_k}])(e^{z_i}) - (e^{z_i})(e^{z_i}+0)}{(e^{z_i}+\sum_{k=i}e^{z_k})^2}$
     $\longrightarrow \frac{\partial a_i}{\partial z_i} = \frac{e^{z_i}}{\sum_{k=i}e^{z_k})}\frac{\sum_{k\neq i}e^{z_k}}{\sum_{k=i}e^{z_k})}$

     We know that : $a_i = \frac{e^{z_i}}{\sum_{k=1}^{n}e^{z_k}} \& (1-a_i) = \frac{\sum_{k\neq i}e^{z_k}}{\sum_{k=1}^{n}e^{z_k}}$
     $\longrightarrow = a_i(1-a_i)$

  $\Longrightarrow$ Thus,

$$\frac{\partial a_i}{\partial z_j} = \begin{cases} \text{-a}_i.a_j, & \text{if(i} \neq j) \\ \text{a}_i.(1-a_i), & if(i=j) \end{cases}$$

3.1 Cost function derivation

$\longrightarrow \frac{\partial L}{\partial z_i} = \frac{\partial}{\partial z_i}[-y_i log(a_i) - \sum_{k\neq i} y_k.log(a_k)]$

$\longrightarrow \frac{-y_i}{a_i}.\frac{\partial a_i}{\partial z_i} - \sum_{k\neq i}\frac{y_k}{a_k}\frac{\partial a_k}{\partial z_i}$

$\longrightarrow \frac{-y_i}{a_i}.a_i(1-a_i) + \sum_{k\neq i}\frac{y_k}{a_k}.a_k.a_i$

$\longrightarrow -y_i + y_i.a_i + a_i\sum_{k\neq i}y_k$

$\longrightarrow -y_i + a_i[y_i\sum_{k\neq i}y_k]$

$\longrightarrow -y_i + a_i[y_i \sum_{k=1}^{n} y_k$

$\longrightarrow we - know - that \sum_{k=1}^{n} y_k = 1$

$\longrightarrow Hence$
$= -y_i + a_i(1)$

$= a_i - y_i$
$\Longrightarrow thus : \frac{\partial L}{\partial z_3} = a_3 - y$
$\Longrightarrow thus : dZ_3 = \frac{\partial Cost}{\partial W_3} = A_3 - Y$

3.2 Carrying out calculation further

Calulation for $\frac{\partial Cost}{dW_2} \& \frac{\partial Cost}{dB_2}$
$\longrightarrow$ as we know that

$\longrightarrow \frac{dL}{da_3} * \frac{da_3}{dz_3} = \frac{\partial L}{\partial z_3} = dZ_3$

$\longrightarrow \frac{\partial Z_3}{\partial a_2} = \frac{partial}{\partial a_2}(W_3.a_2 + B_3) = a_2$

$\longrightarrow \frac{\partial a_2}{\partial Z_2} = f_2'(Z_2)$

$\longrightarrow \frac{\partial Z_2}{\partial W_2} = \frac{\partial}{\partial W_2}(W_2.a_1 + B_2) = a_1$

$\longrightarrow dZ_2 = \frac{\partial Cost}{\partial Z_2} = (W_3)^T.dZ_3 * f_2'(Z_2)$

$\Longrightarrow$Hence ,
$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial a_3} \times \frac{\partial a_3}{\partial Z_3} \times \frac{\partial Z_3}{\partial a_2} \times \frac{\partial a_2}{\partial Z_2} \times \frac{\partial Z_2}{\partial W_2}$

$= dZ_3.W_3.f_2'(Z_2).a_1$

$\frac{\partial Cost}{\partial W_2} = \frac{1}{m}[(W_3)^T.dZ_3 * f_2'(Z_2)](A_1)^T$

$= \frac{1}{m} dZ_2.(A_1)^T$

$\Longrightarrow$ Similarly for Bias

$\frac{\partial Cost}{\partial B_2} = \frac{1}{m} Sum(dZ_2, 1)$

3.3 Derivation of ReLU

we have used ReLU activation function in hidden layer so it is neccesary to calculate derivative of it.

$$f(z) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

$$f'(z) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

4 Tuning parameters

4.1 Unaugmented data-set

1. Epochs : 20000
2. Learning rate : 0.7
3. Training dataset accuracy : 42.35%
4. Test dataset accuracy : 41.42%

4.2 Augmented data-set

1. Epochs : 10000
2. Learning rate : 0.7
3. Training dataset accuracy : 41.51%
4. Test dataset accuracy : 40.38%

4.3 Observation :

On un-augmented data-set we got 42.35% training accuracy but on augmented data-set we got 41.51% training accuracy. We observed that there is slightly degradation in performance on augmented data-set.On augmented data-set we had performed image cutout,crop,rotate which makes it difficult to identify image thus we got less performance.

References

[1] Neural network : https://www.youtube.com/watch?v=vtx1iwmOx10

[2] Neural network : https://youtu.be/f-nW8cSa$_E c$

[3] Neural network : https://youtu.be/URJ9pP1aURo

[4] Image Rotation : https://gautamnagrawal.medium.com/rotating-image-by-any-angle-shear-transformation-using-only-numpy-d28d16eb5076

[5] https://www.cs.toronto.edu/ kriz/cifar.html

[6] https://github.com/deep-diver/CIFAR10-img-classification-tensorflow

[7] https://towardsdatascience.com/cifar-10-image-classification-in-tensorflow-5b501f7dc77c

[8] https://towardsdatascience.com/backpropagation-from-scratch-how-neural-networks-really-work-36ee4af202bf

[9] https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/

[10] https://www.kaggle.com/wwsalmon/simple-mnist-nn-from-scratch-numpy-no-tf-keras