

3

Deeper the network more are the chances of the gradients becoming smaller and smaller or keep growing...

11/26/2023

pra-sâmi

4

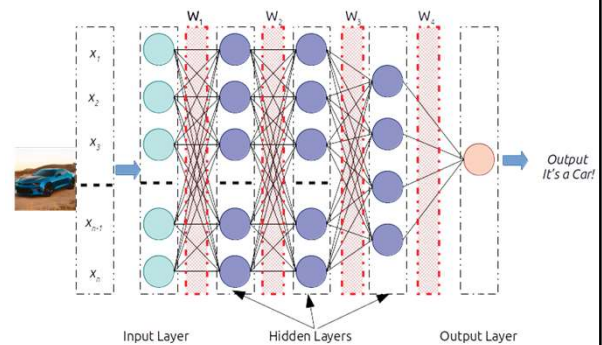
## Weight multiplied

□ We know that :

- ❖  $z = X * W + b$
- ❖  $\hat{y} = a = \sigma(z)$
- ❖  $a_1 = \sigma(a_0 \cdot W_1)$

□ That in multilayer network

- ❖  $\hat{y} = \sigma(\sigma(\sigma(\sigma(\sigma(a_0 \cdot W_1) \cdot W_2) \cdot W_3) \cdot W_4) \dots)$
- ❖ For explanation purpose assume  $\sigma(z) = z$  (say ReLU)
- ❖  $y = W_1 \cdot W_2 \cdot W_3 \cdot W_4$
- ❖ so any change in  $y$  will result in  $W_1 * W_2 * W_3 * W_4$  times  $y$  in layer 1
- ❖ Longer the chain, more  $W_s$  will be multiplied.



11/26/2023

pra-sâmi

5

## More Layers... More problems

- ❑ Assume we have 150 layers
- ❑ Also assume our weight is say 1.1  
 $\Rightarrow 1.1^{150} = 1.6 \text{ million}$
- ❑ On the other hand assume our weight is 0.9  
 $\Rightarrow 0.9^{150} = 1.4 \text{ e}^{-7}$

11/26/2023

pra-sâmi

6

## It's a Severe Problem...

- ❑ No silver bullet solution....
- ❑ There is multi-prong approach to it...
- ❑ First, Initialise your weights as close to 1 as possible ( not 1)
- ❑ It is found that for tanh activation function
  - ❖ Divide by  $\sqrt{\text{number of nodes in the previous layer}}$
  - ❖ for Gaussian distribution it normalises the data with var =1
- ❑ Some cases :  $\frac{2}{\sqrt{\text{number of nodes in the previous layer}}}$
- ❑ In ReLU , :  $\frac{2}{\sqrt{\text{number of nodes in the previous layer} + \text{number of nodes in current layer}}}$
- ❑ Some literature, even  $\frac{K}{\sqrt{\text{number of nodes in the previous layer}}}$  ; K is a another parameter to tune

11/26/2023

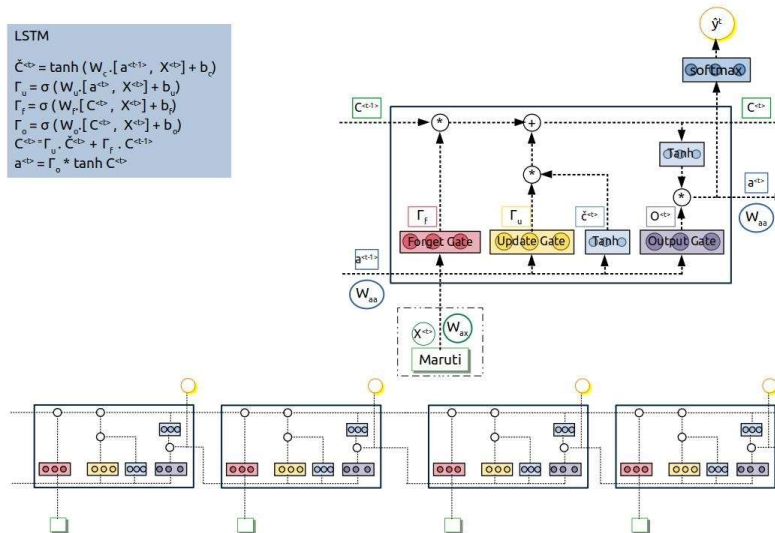
pra-sâmi

7

## Change your architecture - LSTM

LSTM

$$\begin{aligned}\tilde{C}^{(t)} &= \tanh(W_c \cdot [a^{(t-1)}, X^{(t)}] + b_c) \\ \Gamma_u &= \sigma(W_u \cdot [a^{(t-1)}, X^{(t)}] + b_u) \\ \Gamma_f &= \sigma(W_f \cdot [C^{(t-1)}, X^{(t)}] + b_f) \\ \Gamma_o &= \sigma(W_o \cdot [C^{(t-1)}, X^{(t)}] + b_o) \\ C^{(t)} &= \Gamma_u \cdot \tilde{C}^{(t)} + \Gamma_f \cdot C^{(t-1)} \\ a^{(t)} &= \Gamma_o \cdot \tanh C^{(t)}\end{aligned}$$

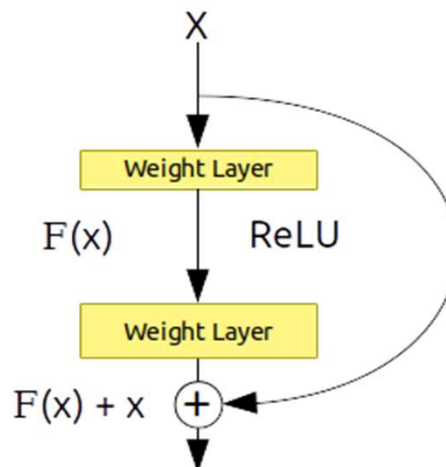


11/26/2023

pra-sâmi

8

## As in ResNet



Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun  
Microsoft Research

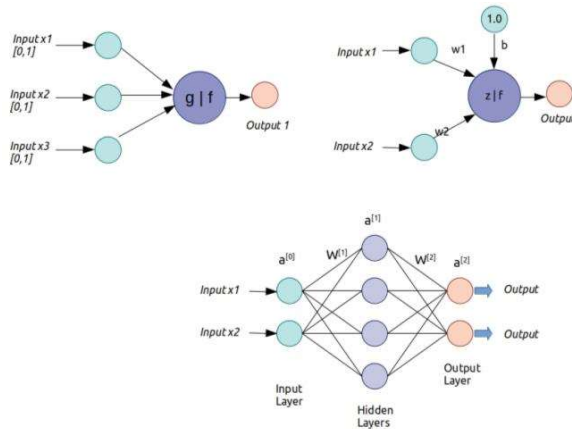
11/26/2023

pra-sâmi

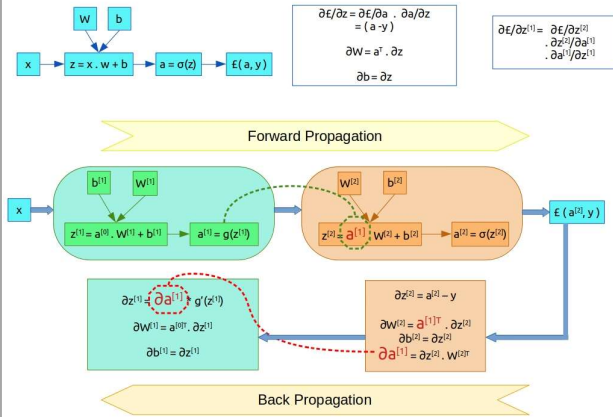
9

## Gradient Checks

### Recap



### The math



11/26/2023

pra-sâmi

10

## Loss Function and its Derivative

- The loss for our prediction  $\hat{y}$  with respect to the true labels  $y$  is given by:

$$L(\hat{y}, y) = -y \cdot \log \hat{y} - (1 - y) \cdot \log (1 - \hat{y})$$

- For all samples:

$$J(\hat{y}, y) = -\frac{1}{m} \sum_{i \in m} y_i \log \hat{y}_i - (1 - y_i) \cdot \log (1 - \hat{y}_i)$$

$$\text{Where } \hat{y} = \sigma(a \cdot W + b)$$

- Therefore, we can say that:

$$J(\hat{y}, y) = J(W, b) = J(W_1, W_2, W_3, \dots, W_n, b_1, b_2, b_3, \dots, b_n)$$

11/26/2023

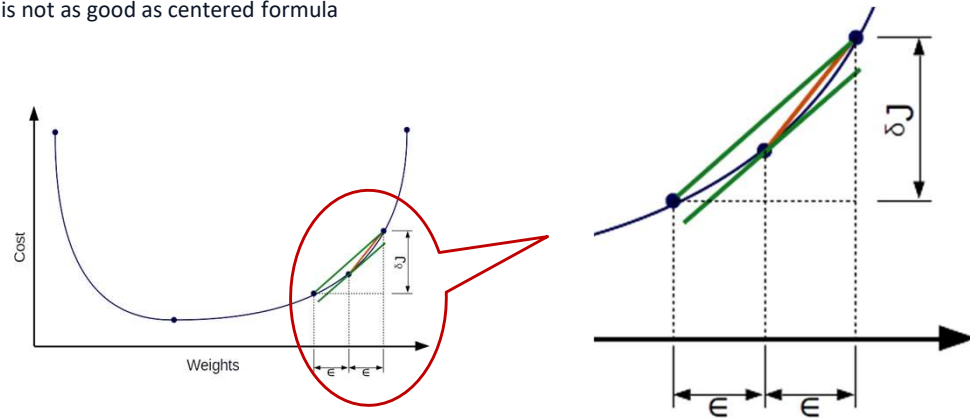
pra-sâmi

11

## Calculation of derivative

### □ Use the centered formula

- ❖ The formula you may have seen for the finite difference approximation when evaluating the numerical gradient is not as good as centered formula



11/26/2023

pra-sâmi

12

## Gradient Checking

### □ Also called "Grad Check"

### □ Do it to verify the model's math(Debug) only

- ❖ Too heavy for training, switch off once the model is verified.

### □ For all values of $W$ s and $B$ s, we can calculate:

$$\delta\theta_{approx} = J(W_1, \dots, W_i + \epsilon, \dots, W_n, b_1, b_2, b_3, \dots, b_n) - J(W_1, \dots, W_i - \epsilon, \dots, W_n, b_1, b_2, b_3, \dots, b_n) / (2 * \epsilon)$$

### □ To check if $\delta\theta_{approx}$ and $\delta\theta$ are close

$$\frac{\|\delta\theta_{approx} - \delta\theta\|_2}{\|\delta\theta_{approx}\|_2 + \|\delta\theta\|_2} \text{ is very small}$$

### □ For $\epsilon = 1e-7$

- ❖ Relative error  $> 1e-2$  usually means the gradient is probably wrong
- ❖  $1e-2 > \text{relative error} > 1e-4$  should make you feel uncomfortable
- ❖  $1e-4 > \text{relative error}$  is usually okay for objectives with kinks. But if there are no kinks (e.g. use of tanh nonlinearities and softmax), then  $1e-4$  is too high.
- ❖  $1e-7$  and less you should be happy

11/26/2023

pra-sâmi

13

## Grad Check Steps

- ❑ Recall: Our model has all weights and biases stored
  - ❖ Model = { " $W_1$ ": ..., " $b_1$ ": ..., " $W_2$ ": ..., " $b_2$ ": ..., ... " $W_n$ ": ..., " $b_n$ ": ... }
  - ❖ We have implemented our forward prop and back prop
- ❑ Step 1 : Pick model and convert all weights and biases into a vector  $\theta$
- ❑ Step 2: Similarly pick  $\delta W$  and  $\delta b$  and convert to a vector  $\delta\theta$
- ❑ Step 3: for each of the value in the vector  $\theta$ 
  - ❖ Make copy of  $\theta$  and  $\delta\theta$
  - ❖ Increase  $\theta_i$  to  $\theta_i + \epsilon$
  - ❖ Calculate  $J +$  ( Cost with increased  $\theta$  )
  - ❖ Similarly calculate  $J -$  ( Cost with decreased  $\theta$  )
  - ❖ Use  $J +$  and  $J -$  to calculate if  $\delta\theta_{approx}$
  - ❖ Calculate  $\delta\theta$  as usual
  - ❖ Find error

11/26/2023

pra-sâmi

14

## Grad Check

- ❑ Phew... too lengthy calculations....
- ❑ Good News!
  - ❖ Both **Tensorflow** and **Torch** have **autograd** implementation for us. So in real implementation we will be using those functions for gradient checking
- ❑ Caution!
  - ❖ This check is resource hungry
  - ❖ Once the verification is done, comment/switch off the code
- ❑ Deeper the network → the higher the relative errors
  - ❖ For the input data for a 10-layer network, a relative error of  $1e-2$  might be okay because the errors build up on the way
  - ❖ Conversely, an error of  $1e-2$  for a single differentiable function likely indicates incorrect gradient

11/26/2023

pra-sâmi

15

## Learning

- ❑ So far, we've discussed the static parts of a Neural Networks:
  - ❖ How we can set up the network connectivity,
  - ❖ The data
  - ❖ The loss function
- ❑ Time to look at the dynamics:
  - ❖ The process of learning the parameters
  - ❖ Finding good hyper-parameters

11/26/2023

*pra-sâmi*

16

Next Session...  
The process of learning the parameters...  
Finding good hyper-parameters...

11/26/2023

*pra-sâmi*

17



11/26/2023

*pra-sâmi*