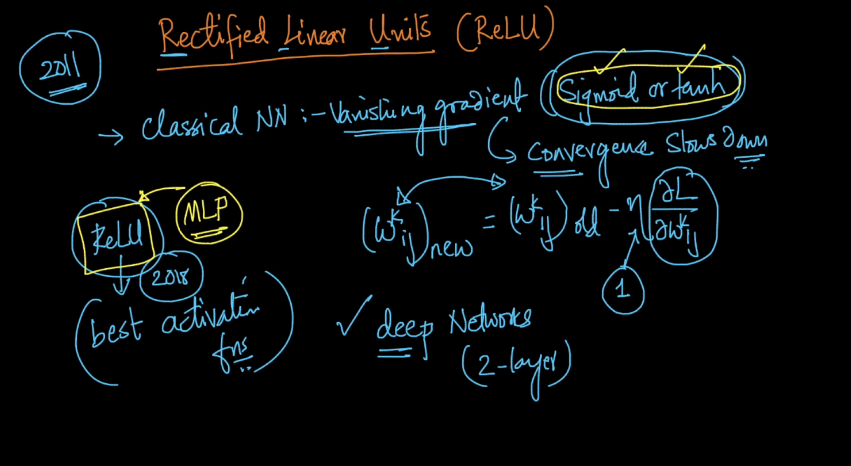
**Rectified Linear Units (ReLU)**

In classical we have a problem of vanishing gradient because we use sigmoid or tanh function and because of this convergence slow down this happens because updates are very less as gradients are vanishes.

To overcome this problem there is another best activation function specifically for MLP which is RELU



So as shown below in activation function z comes which is wTx.

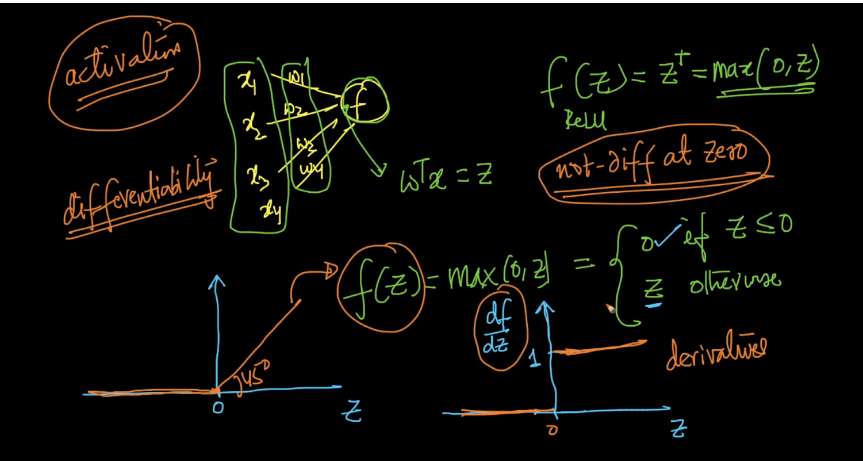
Relu represented as f(z) = z+ = max(0, z)

i.e value of f(z) = 0 when z <= 0

otherwise z

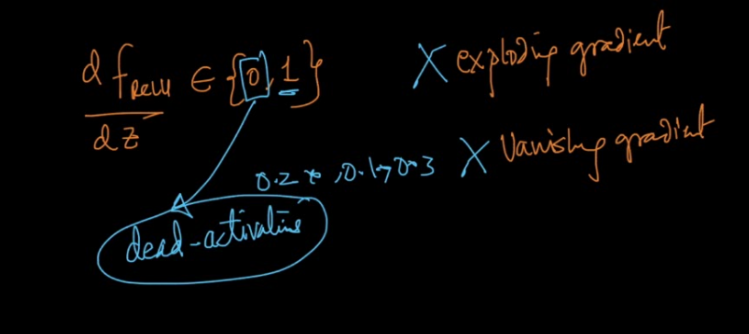
this is shown by first graph in below fig. x-axis is z and y-axis is f(z) when z = 0, f(z) = 0 and when z>=1, f(z) = z which is shown by line which is at 45o therefore its derivative/slope is 1 because tan 45 is 1 and derivative is 0 for f(z) = 0 because tan 0 = 0

but there is a problem of differentiability at z = 0 because for differentiation function should be smooth but it is not at z = 0. But we can solve this problem by using some hacks.



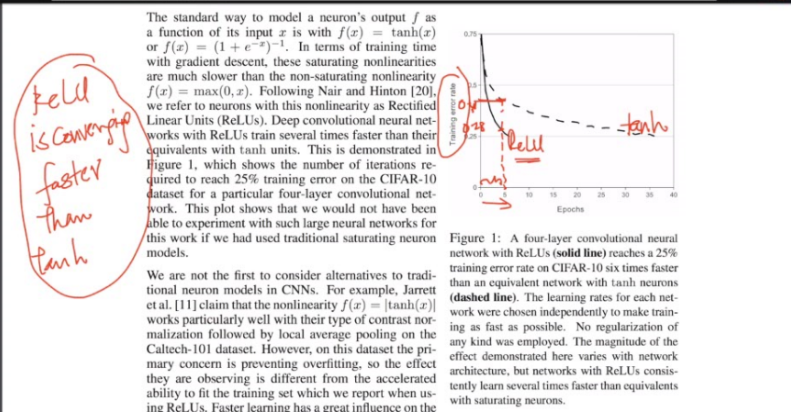
As derivatives of relu is only 0 or 1 therefore there is no problem of exploding gradient because max value of derivative is 1 and there is no problem of vanishing gradient because derivative of relu doesn’t gives values like 0.2,0.1, …..

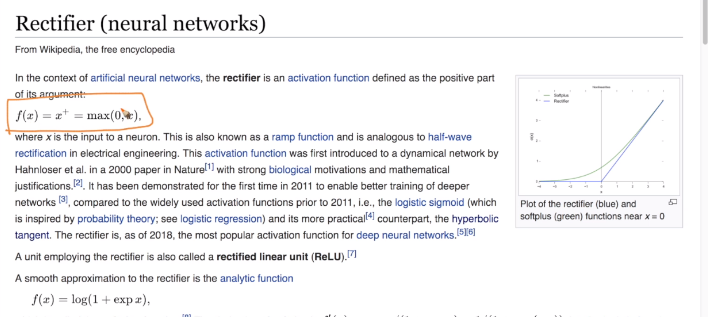
But here there is a problem of dead activation because derivatives of relu is 0 and as there is chain rule therefore some activation unit becomes dead.



In below image graph b/w epoch and training error rate is shown in which smooth line is training error rate of Relu and dotted line is training error rate of tanh. We know as epoch increases error rate decreases.

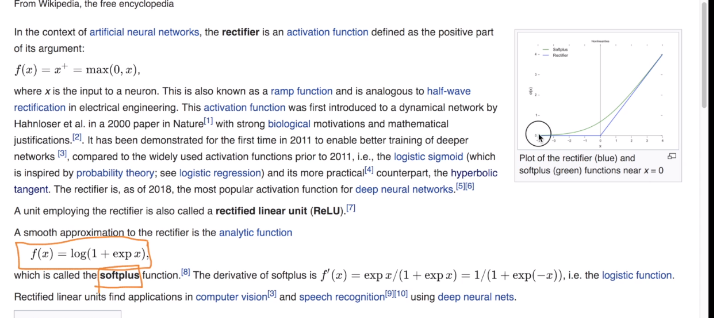
By this it seems that at epoch 5 tanh have error rate 0.4 and relu have only 0.28 means relu is converging faster than tanh.

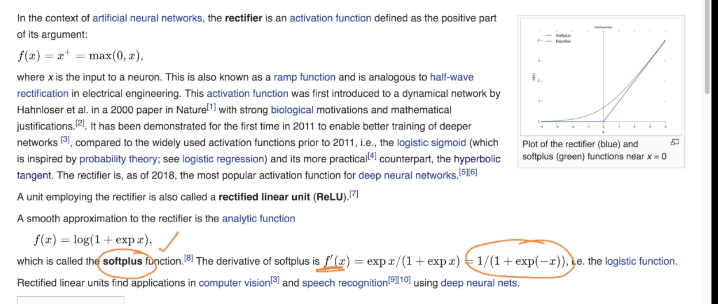




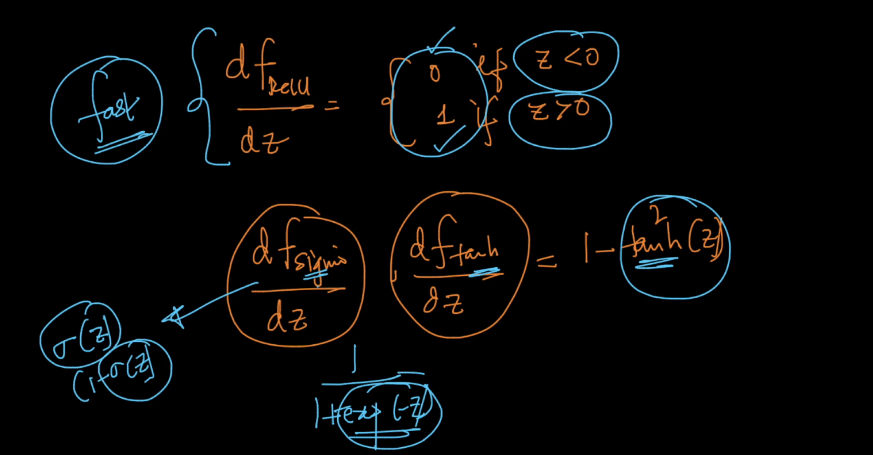
As there is a problem of diff. at z = 0 therefore there is an approximation to this rectifier function is known as softplus function f(x) = log(1 + exp(x)) and graph shown below green line is of softplus and blue is of relu. By this it seems that both lines are almost close together everywhere except at 0 but it is acceptable.

Also the derivative of f(x) of softplus is shown below which is logistic function.



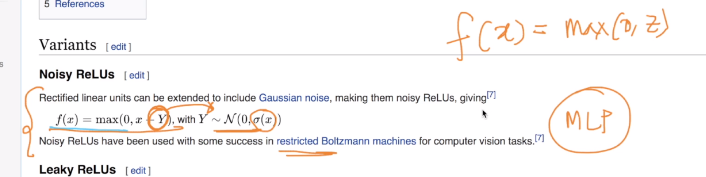


One more thing computing derivative of relu is faster than sigmoid or tanh because in relu we directly got 0 or 1 but in sigmoid we have to compute value of sigma(z) i.e exp(z) and in tanh we have to compute 1-tan2h(z)



There are many variants of Relu

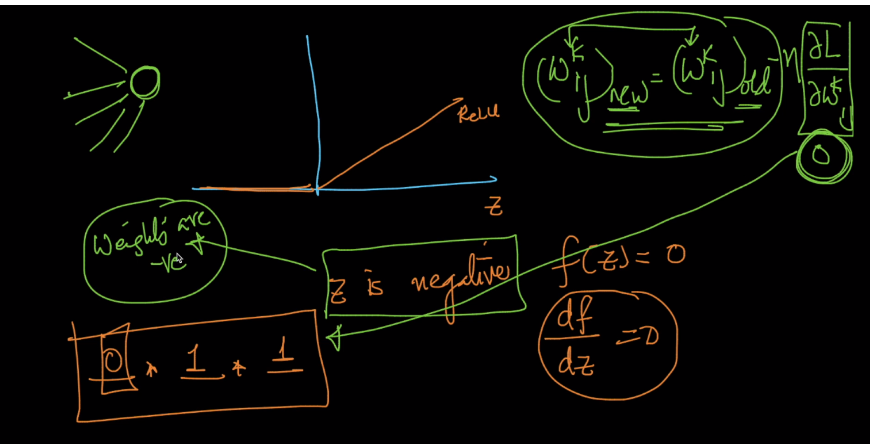
1. Noisy relu : in this we add gaussian noise to x if x > 0. But it is specifically use for restricted Boltzmann machines but it not much used in multi layer perceptron (MLP)



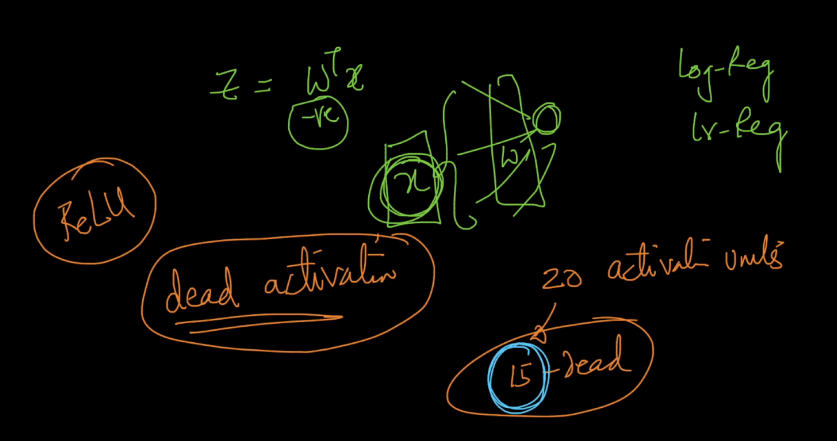
**Dead Activation :**

There is a problem in Relu is that at z <= 0 i.e when z is negative its f(z) is 0 therefore its derivative becomes 0 and by this it makes some activation function dead because as we use chain rule and therefore by this some derivative becomes 0 and if derivatives becomes 0 then new and old weights become equal i.e there is no change on updation therefore it creates problem therefore we have to make sure that there is not much activation units dead.

Why z becomes negative because if all weights coming to neuron is negative then z becomes negative as z = wTx and in this x (inputs) are always normalized at preprocessing time same as we do in logistic and linear regression we do in neural network as well therefore it can’t be negative.

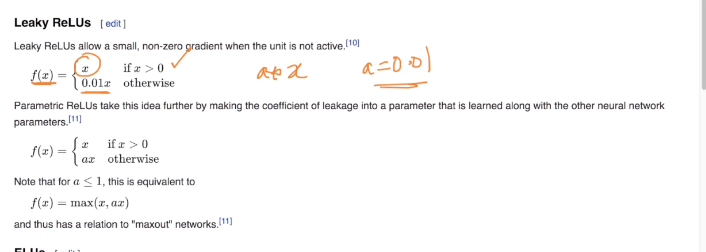


But suppose if there is 20 activation function and 15 are dead then it’s not make much sense therefore we have to make sure it that there is not much activation units dead and for this we use various techniques.



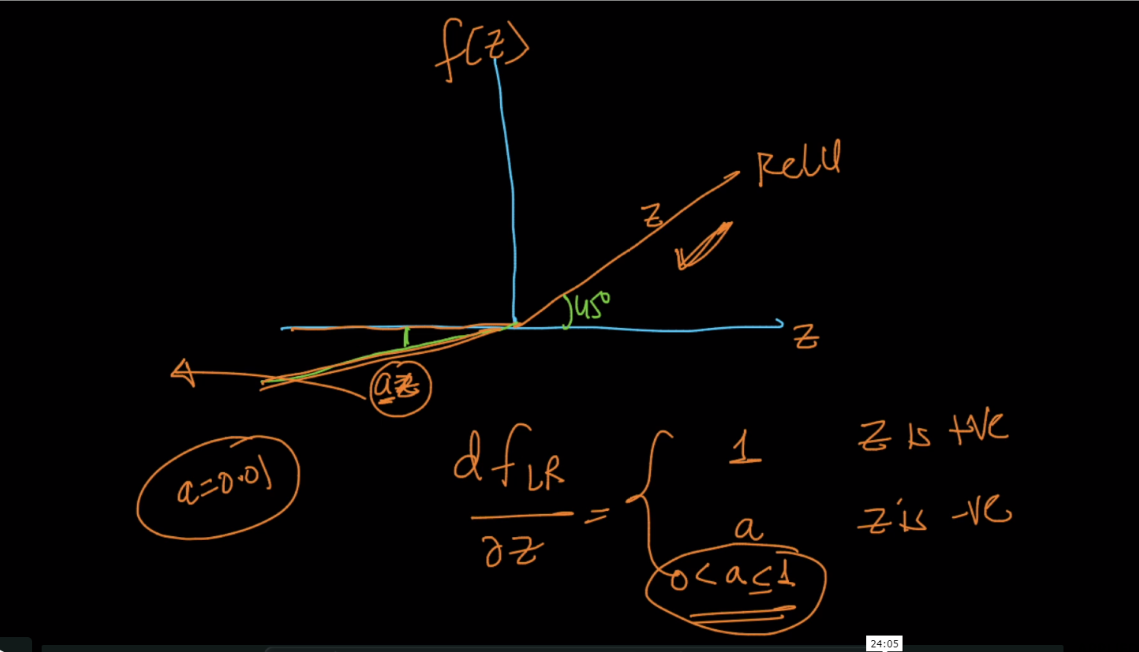
For removing above dead activation problem we can use Leaky Relu which is shown below

So what it do is when x <= 0 then f(x) becomes a\*x ,here is a is hyperparameter or we can simply use a = 0.01



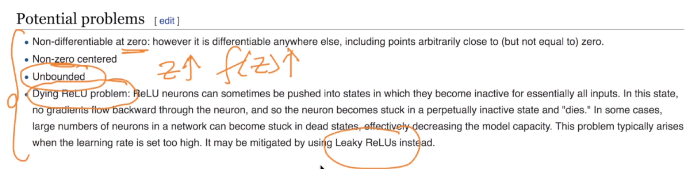
Graphical representation is shown below therefore when z<1 then it doesn’t make f(z) = 0 it multiply a with and that’s how it resolve problem of dead activation. But sometimes by this vanishing gradient may occur because by this it generate very low value after diff. but it occurs rarely.

But most of the time relu is used. It is only used when there is large no. of dead units.



Below image shows problems of Relu :

Unbounded means as z increases, f(z) also increases and thus there is no bound.



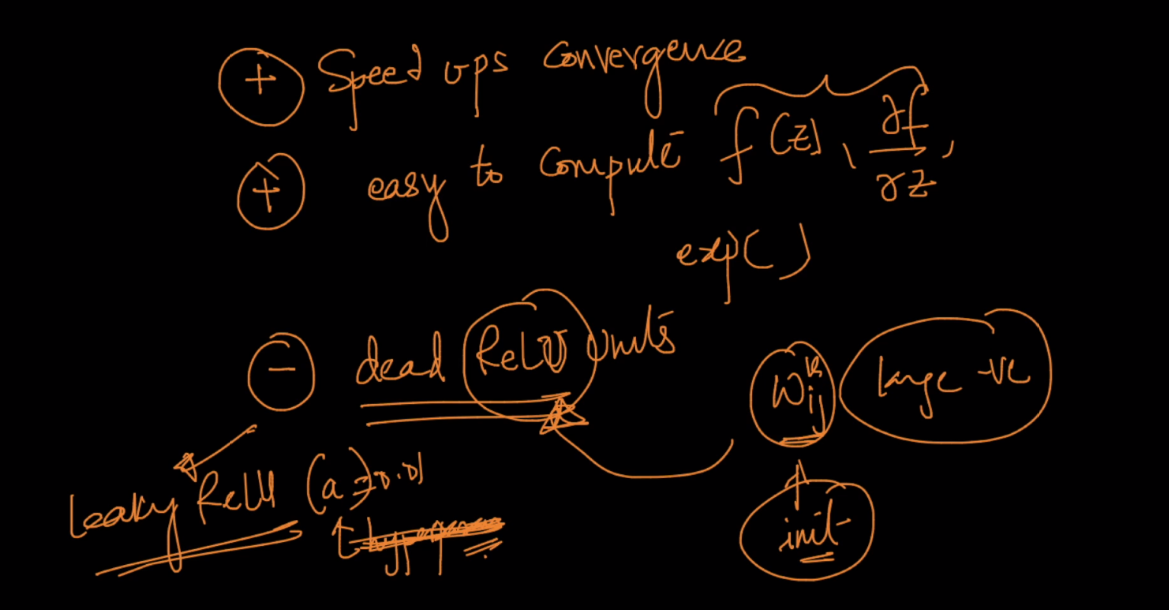
Advantages of Relu :

1. Speed up convergence
2. Easy to compute f(z) , df/dz

Disadvantage :

1. Dead relu units :

We can solve it by properly initializing weights. Or if there are large no. of dead units then we should use leaky relu.



Link : <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>

Comments :

