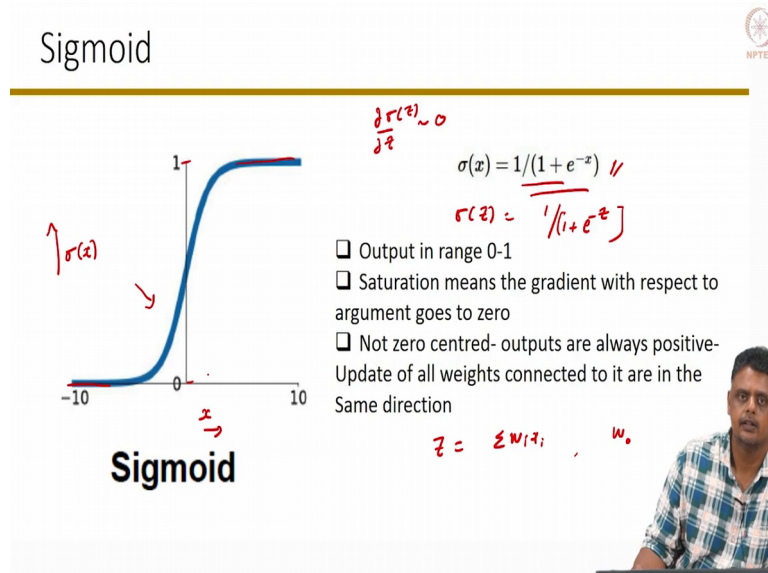


Machine Learning for Engineering and Science Applications
Professor Ganapathy Krishnamurthi
Department of Engineering Design
Indian Institute of Technology Madras
Activation Function

(Refer Slide Time: 0:14)



So we will look at the sigmoid non-linear activation now, so here is the analytical expression for this sigmoid non-linear activation okay and this is the plot of the function, you can see that, so this axis is the X axis and Y axis is sigma for X okay, so we can see that for large value of X the sigmoid function tends to 1, again for large value, positive values of X and for large negative values of X again, it tends to 0, thing to notice that it becomes flat for large values of X as well as large positive values of X as well as for large negative values of X.

So in this slide, we use X but typically the input to the nonlinearity is your Z which is summation $\sum w_i x_i$ okay for there is a w_0 which is bias term also coming in which I have not included in this summation but the Z is what is typically use, so you can also rewrite this as sigma Z this is just so that you do not confuse, get confused when you see as using Z in a letter like just sometime.

So the input to the sigmoid function you might know already is the linear combination of your input features with the weights okay, so what this says is that if Z is very large that is summation $\sum w_i x_i$ is very large than either magnitude of that, so very large positive as well as very large negative numbers $\sum w_i x_i$ summation would lead to the sigmoid function being saturated.

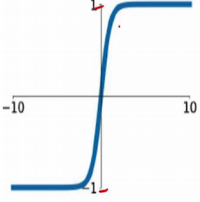
Now what does that mean? Which means that the gradient where in this case you will just to $D \sigma$ of Z by ΔZ will be close to 0 very small number okay, so this means that if you and of course certainty back propagation like just will see that this leads to negligible or 0 updates of your weights, so the learning basically code encode learning will stop, so the weights want be updated as frequently and they will not change as dynamically as you expect them to be.

So this is one of the problems with the sigmoid functions, so that, this is, of course this has many other advantages, one of them is that suppose your output layer you want interpreted as a probability score then this would be the optimal thing to use because the outputs are between the values 0 and 1, so this also known as squashing function because it squashes the output in the range 0 to 1. Okay, the next function that you are going to look at is the tanh hyperbolic which is very similar features except for the range of the output.

(Refer Slide Time: 3:04)


NPTEL

Tanh



tanh(x)

- Output in the range -1 to 1
- Saturation problems exists
- Acts like identity near origin- Sigmoid is also Linear near origin but not identity

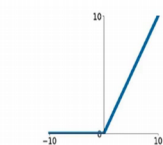


The tanh hyperbolic function is again one of the activation function that are often use, it is one of the earliest functions to be used, in the here the range of this function is between -1 and 1 and here to again saturation problems exists and acts like identity in the sense, it is linear identity, linear near origin unlike the sigmoid function, so again, this is also referred for certain type problems and saturation problems here again meets that the gradient with respect to the argument, input argument goes to 0 so, which means that weights will not get updated.

(Refer Slide Time: 3:43)



ReLU



ReLU
(Rectified Linear Unit)

$$f(x) = \max(0, x)$$

- No saturation problem for large input
- Near 0- neurons will die- no updates
- Initialize with positive bias

Computer Vision




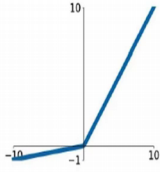
This is again the most effective, the ReLU or the rectified linear unit as it known as, it is one of the more effective squash function and it has been used widely in computer vision problems, very successful in computer vision problems and, so this is one of the preferred last functions for many of the modern networks, architectures that around their, again, it is very simple that if for all values greater than 0, it is just gives the same value, that is basically its identity for all values affects the statement 0, greater than or equal to 0, and for anything, for any input less than or equal to 0, the output is 0.

Here the problem is that very for very small values of the input than the gradient will be the F of the function value will be very small, so that is one of the drawbacks and for negative values again the gradient will go to 0 right, so this is the, because the F of X is 0, so the gradient of F of X with respect to X, if X is negative that will also be 0, so then updates will not be done to the weights, so this is here, the problem with ReLU but otherwise this is a very effective cross functions, again, you know that this is actually a non-linear function, in the sense that it exists only for values of the input it than 0 and it is 0 for all values of X less than or equal to.

(Refer Slide Time: 5:15)


Leaky ReLU





Leaky ReLU
 $f(x) = \max(0.01x, x)$

Parametric Rectifier (PReLU)
 $f(x) = \max(\alpha x, x)$



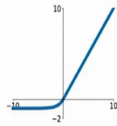
- No problems near large positive or positive inputs
- Empirically determine "leakiness"
- Or make it an hyperparameter "PReLU"

So the leaky ReLU was to again, was take care of some of the problems that ReLU had, it said is that for, it give you a small gradient value for negative values of the input right, so in the previous slide we saw that for just a plain ReLU or the rectified linear unit, if the input was negative than your gradient also go 0 but this case, having it in this, so for all negative values of X it will give you a scale the value of that input, so which means that the gradient can exist okay.

In other version of this is the parametric rectifier linear unit, wherein instead of having a fixed at scaling factor for X for negative values of X, we have an alpha which is again a parameter which is learned during the back propagation process okay.

(Refer Slide Time: 6:06)

Exponential Linear Unit

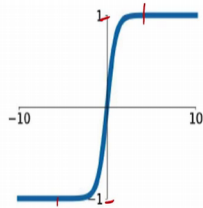


$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha (\exp(x) - 1) & \text{if } x \leq 0 \end{cases}$$

For large negative saturates – retains all other Aspects of ReLU



Tanh

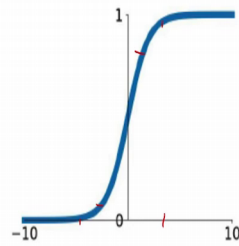


tanh(x)

- ❑ Output in the range -1 to 1
- ❑ Saturation problems exists
- ❑ Acts like identity near origin- Sigmoid is also Linear near origin but not identity



Sigmoid



$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

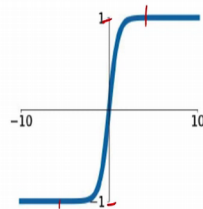
- Output in range 0-1
- Saturation means the gradient with respect to argument goes to zero
- Not zero centred- outputs are always positive- Update of all weights connected to it are in the Same direction

Sigmoid

$$\sum w_i x_i \quad \sigma(\sum w_i x_i)$$



Tanh



- Output in the range -1 to 1
- Saturation problems exists
- Acts like identity near origin- Sigmoid is also Linear near origin but not identity

tanh(x)



At another variation of this is a exponential linear unit which again for negative values of X takes this warm, this is to make sure that the average activation in a lane goes to 0, so again this has a gradient for negative values of X also okay, so these are the typical activation function and are used in the network, you can for instance have not I think shown here, the actual analytical form for tanh hyperbolic X you can look that up and convince yourself that a plot looks this way.

So you can see that the problems with sigmoid and tanh or basically the saturation here, on the other hand, they are very convenient in the sense that the output or squash between -1 to 1 or 0 to 1, so for sigmoid is 0 to 1, so that again if you are see, so you if you look at sigmoid, so let us see if you want output layer to be a probability, to be interpreted as probability score then maybe add the output layer that something like sigmoid function would be an appropriate nonlinearity, sigmoid tanh hyperbolic and ReLu or the most often use and

effective activation functions that are used in divisional networks, all flavours of artificial networks.