Lecture 3 : Paper Work

Activation Functions

Prasoon Raghuwanshi (15MI440) **Date:22/08/2019**

Nonlinear activation functions have brought neural networks their nonlinear capabilities, which means neural networks can differentiate those data that can not be classified linearly in the current data space.

1. **Sigmoid**

   Sigmoid function is one of the most common forms of activation functions. The definition of sigmoid function is as follows.

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

   in which $x \in (-\infty, +\infty)$, $g \in (0,1)$. It is known that the modelling and training process of a multi-layer neural network can be divided into two parts: forward propagation and back propagation. And in the back propagation, the derivatives of activation functions in each layer should be calculated. The sigmoid function is a continuous function, which means that it is differentiable everywhere. The derivatives of sigmoid function

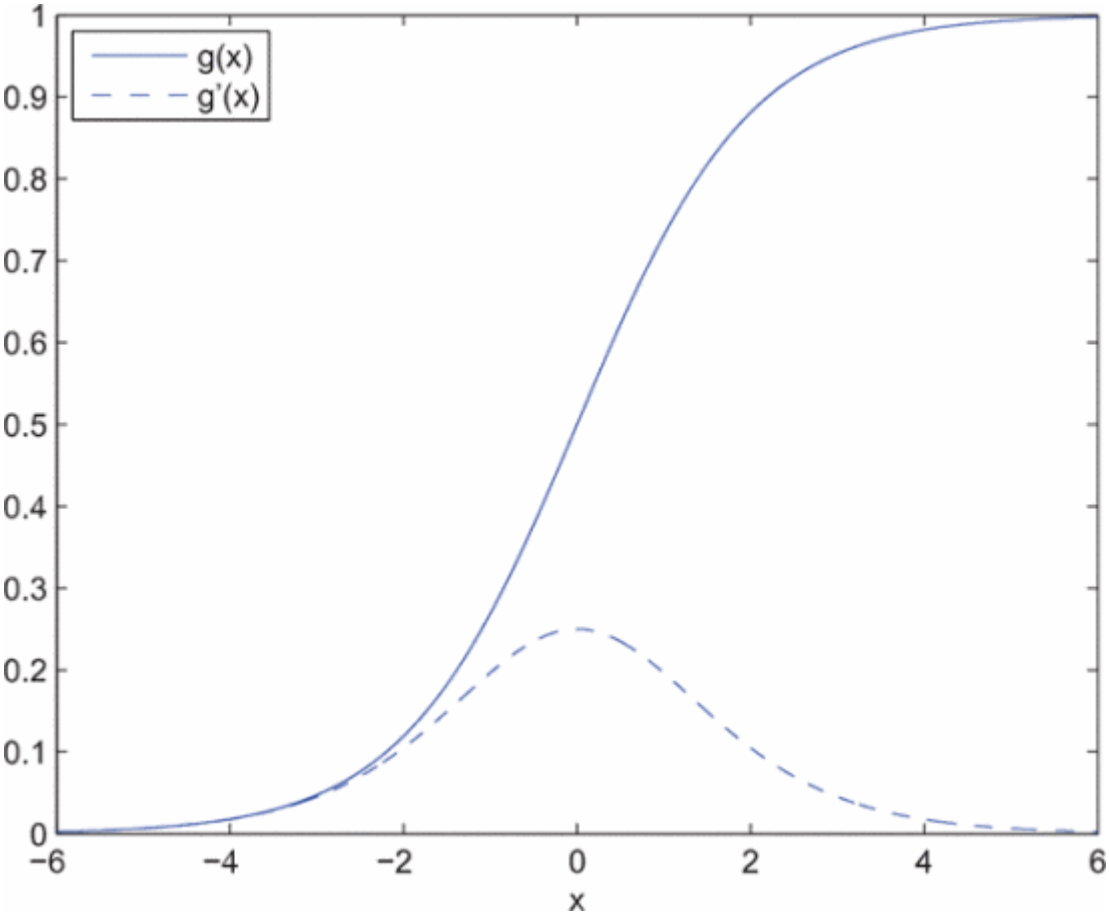
   $$g'(x) = \frac{e^{-x}}{(1 + e^{-x})^2}$$

   is easy to be calculated. Therefore, the sigmoid function was commonly used in shallow neural networks. In addition, the sigmoid function is frequently employed in the output level of a neural network owing to its value distribution. While sigmoid function is rarely adopted in the deep neural networks except the output level since its saturation. Exactly, it is soft saturate since it only achieves zero gradient in the limit

   $$\lim_{x \to +\infty} g'(x) = 0$$

   $$\lim_{x \to -\infty} g'(x) = 0$$

   The soft saturation results in the difficulties of training a deep neural network. More specific, in the process of optimizing loss function, the derivatives of sigmoid function, which is contributed to update the weights and the bias, will reduce to zero when it comes to the saturation area, which brings about the less contributions of the first several layers in the knowledge learning from training samples. In fact, the situation is called vanishing gradient. Generally, the vanishing gradient will emerge in a neural network with more than five layers



The graphic depiction of sigmoid function and its derivative

2. **Hyperbolic Tangent**

Hyperbolic tangent function can be easily defined as the ratio between the sine and the cosine functions.

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

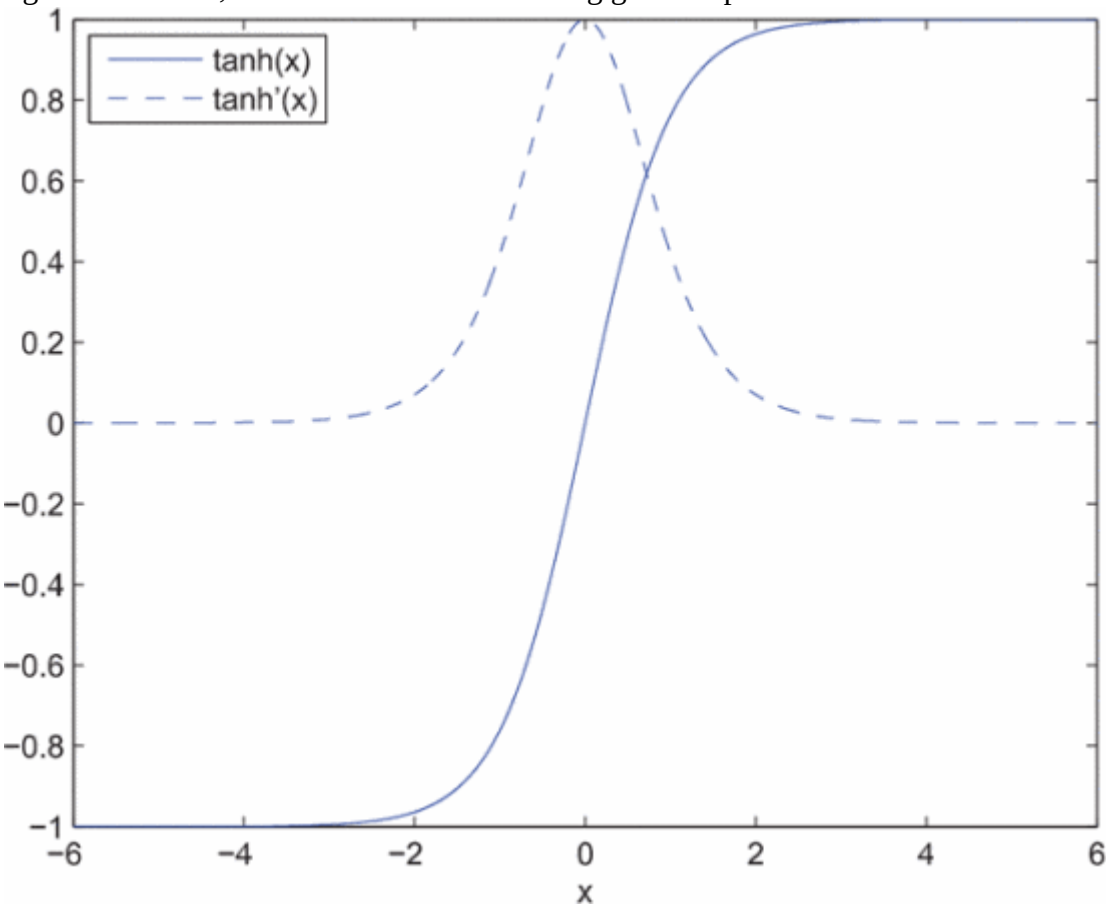
It is similar to sigmoid function and can be deducted from the sigmoid function

$$tanh(x) = 2sigmoid(2x) - 1$$

where sigmoid(·) is g(x). The hyperbolic tangent function ranges outputs between −1 and 1. It is also a continuous and monotonic function, and it is differentiable everywhere. It is symmetric about the origin, of which the outputs, namely the inputs of next layer, are more likely on average close to zero. Therefore, the hyperbolic tangent functions are more preferred than sigmoid functions. In addition, neural networks with hyperbolic tangent activation functions converge faster than those with sigmoid activation functions and the neural networks with hyperbolic tangent activation functions have lower classification error than those with sigmoid activation functions. However, the calculation of the derivatives of hyperbolic tangent functions, listed as follows,

$$tanh'(x) = 2sigmoid'(2x) - 1 = \frac{4e^{-2x}}{(1+e^{-2x})^2}$$

is more complicated than sigmoid function. And it has the same soft saturation as sigmoid function, which also has the vanishing gradient problem.



The graphic depiction of hyperbolic tangent function and its derivatives.

3. ReLU

In the neural networks with sigmoid or hyperbolic tangent activation functions, almost one half of the neuron units are activated at the same time. Furthermore, activating more neuron units will bring about more difficulties in the training of a deep neural network. Rectified linear units (ReLU) will help hidden layer of the neural network to obtain the sparse output matrix, which can improve the efficiency. ReLU function and its improvements are almost the most popular activation functions used in deep neural networks currently. The definition of ReLU is as follows

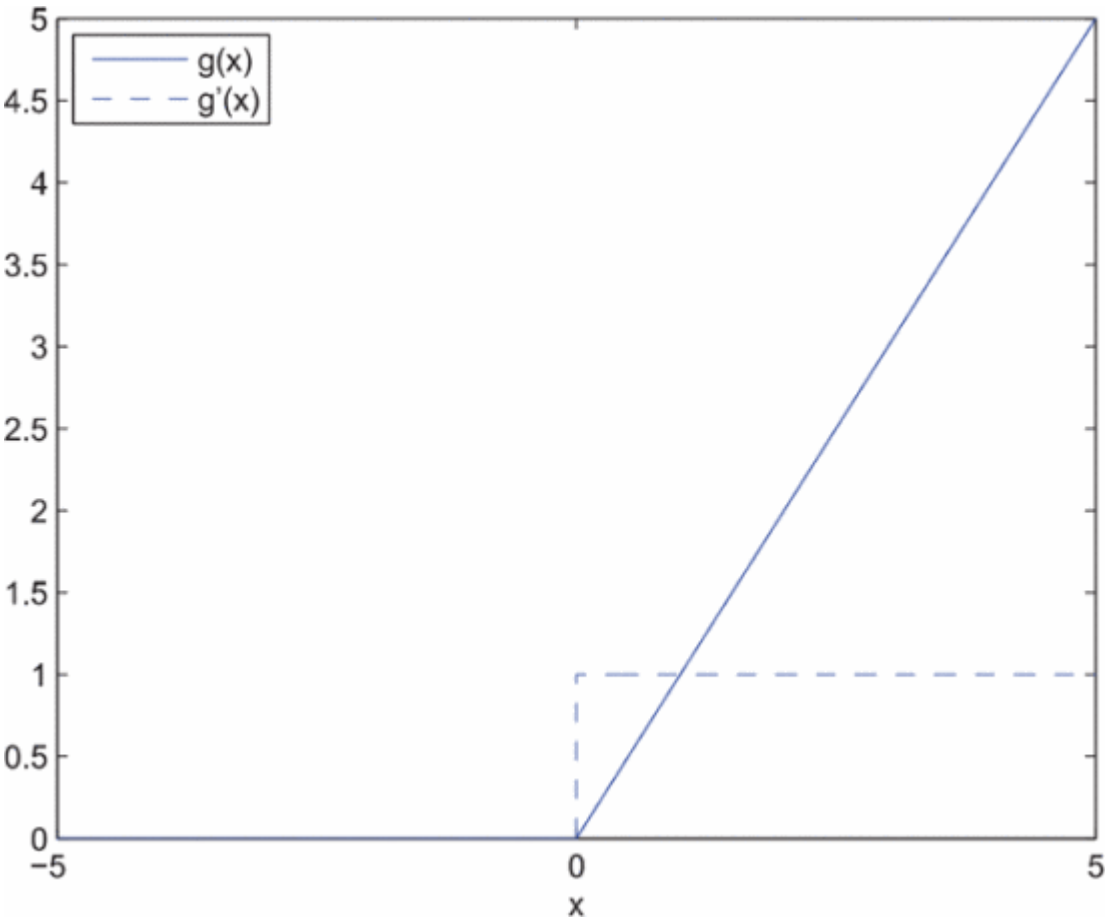
$$g(x) = max(0, x) = \begin{cases} x & if \ \ x \geq 0 \\ 0 & if \ \ x < 0 \end{cases}$$

ReLU activation function is non-saturated, and its derivative function is

$$g'(x) = \begin{cases} 1 & if \ \ x \geq 0 \\ 0 & if \ \ x < 0 \end{cases}$$

which is a constant when the input x>0. Thus, the vanishing gradient problem can be released. Specifically, ReLU function has the following advantages:

- Computations of neural networks with ReLU functions are cheaper than sigmoid and hyperbolic tangent activation functions because there are no need for computing the exponential functions in activations.
- The neural networks with ReLU activation functions converge much faster than those with saturating activation functions in terms of training time with gradient descent.
- The ReLU function allows a network to easily obtain sparse representation. More specific, the output is 0 when the input x<0, which provides the sparsity in the activation of neuron units and improves the efficiency of data learning. When the input x≥0, the features of the data can be retained largely.
- The derivatives of ReLU function keep as the constant 1, which can avoid trapping into the local optimization and resolve the vanishing gradient effect occurred in sigmoid and hyperbolic tangent activation functions.
- Deep neural networks with ReLU activation functions can reach their best performance without requiring any unsupervised pre-training on purely supervised tasks with large labeled datasets.



The graphic depiction of ReLU function and its derivatives.

However, the derivatives g′(x)=0 when x<0 so the ReLU function is left-hard-saturating. And the relative weights might not be updated any more and that leads to the death of some neuron units, which means that these neuron units will never be

activated. Another disadvantage of ReLU is that the average of the units' outputs is identically positive, which will cause a bias shift for units in the next layer. The above two attributes both have a negative impact on convergence of the corresponding deep neural networks.

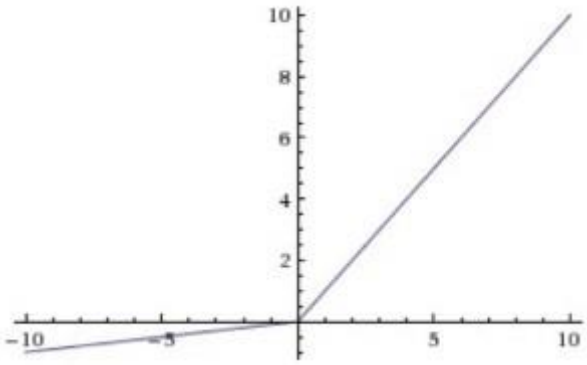
4. LReLU

The possible death of some neuron units of neural networks with ReLU functions are resulted from the compulsive operation of letting g(x)=0 when x<0. In order to alleviate this potential problem, the leaky rectified linear units (LReLU) is used

$$g(x) = max(0, x) = \begin{cases} x & if \ \ x \geq 0 \\ 0.01x & if \ \ x < 0 \end{cases}$$

Its derivative function is

$$g'(x) = \begin{cases} 1 & if \ \ x > 0 \\ 0.01 & otherwise \end{cases}$$



The graphic depiction of LReLU function.

5. Softmax

Softmax function calculates the probabilities distribution of the event over 'n' different events. In general way of saying, this function will calculate the probabilities of each target class over all possible target classes. Later the calculated probabilities will be helpful for determining the target class for the given inputs.

The main advantage of using Softmax is the output probabilities range. The range will 0 to 1, and the sum of all the probabilities will be equal to one. If the softmax function used for multi-classification model it returns the probabilities of each class and the target class will have the high probability.

The formula computes the exponential (e-power) of the given input value and the sum of exponential values of all the values in the inputs. Then the ratio of the exponential of the input value and the sum of exponential values is the output of the softmax function.

$$\text{softmax}(\mathsf{y})_i = \frac{\exp(\mathsf{y}_i)}{\sum_j \exp(\mathsf{y}_j)}$$

References:

Bin Ding ; Huimin Qian ; Jun Zhou, "Activation functions and their characteristics in deep neural networks", 2018 Chinese Control And Decision Conference (CCDC), https://ieeexplore.ieee.org/document/8407425/authors#authors.

Chigozie Enyinna Nwankpa, Winifred Ijomah, Anthony Gachagan, and Stephen Marshall, "Activation Functions: Comparison of Trends in Practice and Research for Deep Learning", https://arxiv.org/pdf/1811.03378.pdf.