COMPUTER VISION AND PATTERN RECOGNITION

SPRING 22-23

Mid Report

Ahnaf Faiaz

ID: 20-42213-1

Section: A

Activation functions:

In artificial neural networks, an activation function is a function that outputs a smaller value for tiny inputs and a bigger value if its inputs are greater than a threshold. The activation function decides whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it. The purpose of the activation function is to introduce non-linearity into the output of a neuron. Some classic activation functions are discussed here.

Step function:

This function is also known as binary step function. A common component of neural networks is a unit step activation function. For negative arguments, the result is set to 0 and for positive arguments, to 1. This is how it works:

$$f(x) = 0 \text{ when } x<0,$$

$$1 \text{ when } x>= 0$$

The range is between (0,1) and the output is binary in nature. These kinds of step activation functions are useful for binary classification schemes. In other words, when we want to classify an input pattern into one of two groups, we can use a binary classifier with a step activation function.

Advantages:

- Effective when dealing with binary classification where output of neural network must be 0 or 1.
- Gives discrete output
- Easy to use

Disadvantages:

- The step function has a negative gradient. Since the gradients of the activation functions are sent for error computations during back-propagation in order to refine and optimize the findings, this renders the step function less helpful.
- It cannot be used for multi-class classification.

Sigmoid function:

The Sigmoid function performs the role of an activation function in machine learning which is used to add non-linearity in a machine learning model. Basically, the function determines which value to pass as output and what not to pass as output. When the activation function for a neuron is a sigmoid function, it is a guarantee that the output of this unit will always be between 0 and 1. Also, as the sigmoid is a non-linear function, the output of this unit would be a non-linear function of the weighted sum of inputs. Such a neuron that employs a sigmoid function as an activation function is termed as a sigmoid unit. It is also known as logistic function.

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

Advantages:

- produces output numbers between 0 and 1, which is useful for logistic regression and binary categorization issues.
- Easy to understand and apply
- Easy to train on small dataset

Disadvantages:

- Prone to Vanishing Gradient problem
- Not a zero-centric function (Always gives a positive values)
- Computationally expensive function (exponential in nature)

Tanh function:

Tanh function is very similar to the sigmoid/logistic activation function, and even has the same S-shape with the difference in output range of -1 to 1. In Tanh, the larger the input (more positive), the closer the output value will be to 1.0, whereas the smaller the input (more negative), the closer the output will be to -1.0. The equation for Tanh is:

tanh(x) = (ex - e(-x)) / (ex + e(-x))

Advantages:

- Since the tanh activation function's output is zero-centered, it is simple to translate the output values to highly negative, neutral, or strongly positive values.

- It is a smooth gradient converging function.

Disadvantages:

- Prone to Vanishing Gradient function
- Hard to train on small datasets

ReLU function:

The full form of ReLU is Rectified Linear Units. Although it gives an impression of a linear function, ReLU has a derivative function and allows for backpropagation while simultaneously making it computationally efficient. The main catch here is that the ReLU function does not activate all the neurons at the same time. The neurons will only be deactivated if the output of the linear transformation is less than 0.

The simple ReLU function can be seen in the following equation:

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

In other words, the activation is simply threshold at zero

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Advantages:

- Avoids and rectifies vanishing gradient problems
- It does not activate all the neurons at the same time
- Computationally efficient

Disadvantages:

- Can only be used with a hidden layer
- Hard to train on small datasets need much data for learning non-linear behavior
- It is not suitable for negative inputs

ELU function:

The full form of ELU is Exponential Linear Units. It modifies the slope of the negative part of the function. ELU is a strong alternative for f ReLU because it becomes smooth slowly until its output equal to -α whereas RELU sharply smooths. It maps the input values to an output range of (-1, ∞).

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

Advantages:

- Gives smoother convergence for any negative axis value
- For any positive output, it behaves like a step function and gives a constant output.
- The ELU function has a mean output closer to zero compared to ReLU

Disadvantages:

- Exploding gradient problem
- Because of the included exponential process, the calculation takes longer

SELU function:

Internal normalization, which is handled by SELU, ensures that each layer maintains the mean and variance from the levels before it. This was described in self-normalizing networks. This adjustment is made possible by SELU by modifying the mean and variation. SELU network neuronal activations automatically converge to a zero mean and unit variance.

$$f(x) = \begin{cases} \lambda x, \text{ if } x > 0 \\ \lambda\alpha(e^x - 1), \text{ if } x \leq 0 \end{cases}$$

Advantages:

- Internal normalization is faster than external normalization, which means the network converges faster
- Doesn't have vanishing gradient problem
- SELUs learn faster and better

Disadvantages:

- SELU is a relatively new activation function so it is not yet used widely in practice. ReLU stays as the preferred option
- Could be computationally expensive