Programming Test: Learning Activations in Neural Networks

ANJANA S KUMAR

April 24, 2021

1 Need of Activation Function

An activation function(AF) in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network.Sometimes the activation function is called a "transfer function." If the output range of the activation function is limited, then it may be called a "squashing function." Many activation functions are nonlinear and may be referred to as the "nonlinearity" in the layer or the network design. The choice of activation function has a large impact on the capability and performance of the neural network, and different activation functions may be used in different parts of the model. Technically, the activation function is used within or after the internal processing of each node in the network, although networks are designed to use the same activation function for all nodes in a layer. A network may have three types of layers: input layers that take raw input from the domain, hidden layers that take input from another layer and pass output to another layer, and output layers that make a prediction.

2 Choosing the right AF

All hidden layers typically use the same activation function. The output layer will typically use a different activation function from the hidden layers and is dependent upon the type of prediction required by the model. There are perhaps three activation functions you may want to consider for use in hidden layers; they are:

1.Rectified Linear Activation (ReLU)

2.Logistic (Sigmoid)

3.Hyperbolic Tangent (Tanh)

Now that we have seen so many activation functions, we need some logic / heuristics to know which activation function should be used in which situation. Good or bad — there is no rule of thumb. However depending upon the properties of the problem we might be able to make a better choice for easy and quicker convergence of the network.

•Sigmoid functions and their combinations generally work better in the case of classifiers

•Sigmoids and tanh functions are sometimes avoided due to the vanishing gradient problem

•ReLU function is a general activation function and is used in most cases these days

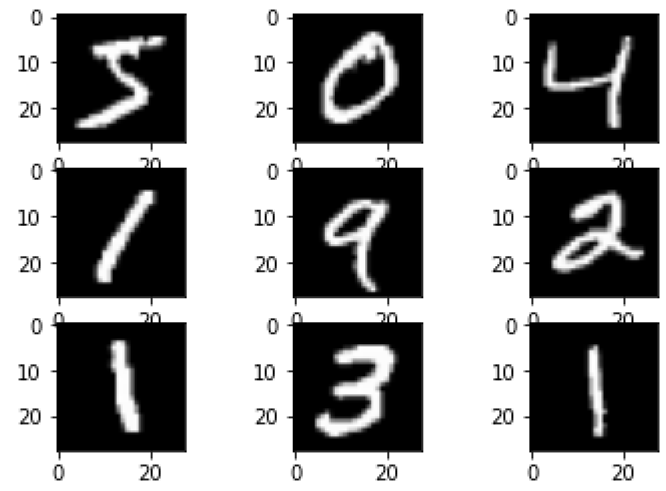
•If we encounter a case of dead neurons in our networks the leaky ReLU function is the best choice

•Always keep in mind that ReLU function should only be used in the hidden layers As a rule of thumb, you can begin with using ReLU function and then move over to other activation functions in case ReLU doesn't provide with optimum results

3 MNIST DATASET

MNIST data is a set of approximately 70000 photos of handwritten digits, each photo is of size 28x28, and it's black and white. That means that our input data shape is (70000,784) and our output (70000,10).Here I am using a basic fully connected Neural Network with a single hidden layer.There're 784 neurons in the input layer, one for each pixel in the photo, 512 neurons in the hidden layer, and 10 neurons in the output layer, one for each digit.

```
Train: X=(60000, 28, 28), y=(60000,)
Test: X=(10000, 28, 28), y=(10000,)
```



We can see that there are 60,000 examples in the training dataset and 10,000 in the test dataset and that images are indeed square with 28×28 pixels.The dataset already has a well-defined train and test dataset that we can use.we know that the images are all pre-aligned (e.g. each image only contains a hand-drawn digit), that the images all have the same square size of 28×28 pixels, and that the images are grayscale.Therefore, we can load the images and reshape the data arrays to have a single color channel.

We also know that there are 10 classes and that classes are represented as unique integers.We can, therefore, use a one hot encoding for the class element of each sample, transforming the integer into a 10 element binary vector with a 1 for the index of the class value, and 0 values for all other classes. We can achieve this with the to$_categorical()utilityfunction.$

A good starting point is to normalize the pixel values of grayscale images, e.g. rescale them to the range [0,1]. This involves first converting the data type from unsigned integers to floats, then dividing the pixel values by the maximum value.

Next step is building the network using keras layer Sequential.Firstly training the network using these parameters: The loss function used was categorical cross entropy optimizer is RMS Prop and evaluation metrics is accuracy,since the dataset is balanced and it is a multi class classification.

Now training the network using a for loop without and with different activation that will continually evaluate performance of each activation function after every set of epochs.Epochs were set to 20 functions We can see that all the activation functions are achieve 98percent accuracy eventu-

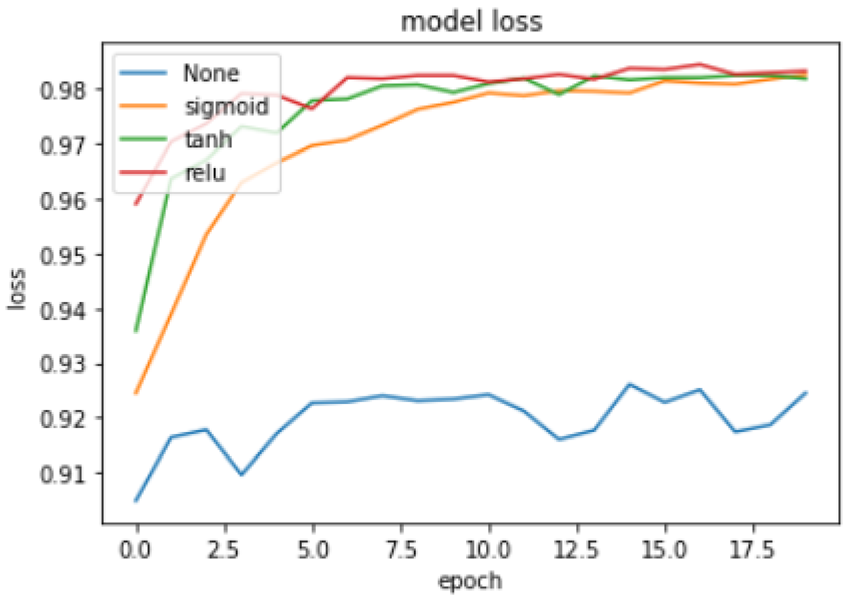


Figure 1: Plot of model loss using different activation functions

ally.Without using activation function shows poor performance.

It is important to note that there's no best activation function. One may be better than other in many cases, but will be worse in some other cases. Another important note is that using different activation functions doesn't affect what our network can learn, only how fast (how many data/epochs it needs).