Neural Networks

UCL Data Science Society

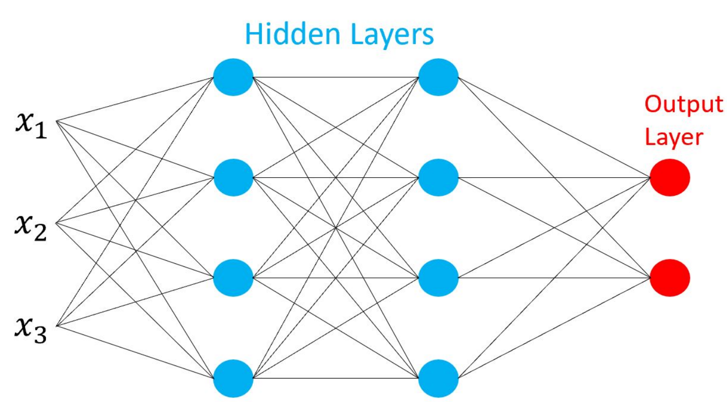
February 5, 2020

1 Introduction

In this document we briefly go over the basic ideas behind a neural network. This is by no means a comprehensive overview of neural networks.

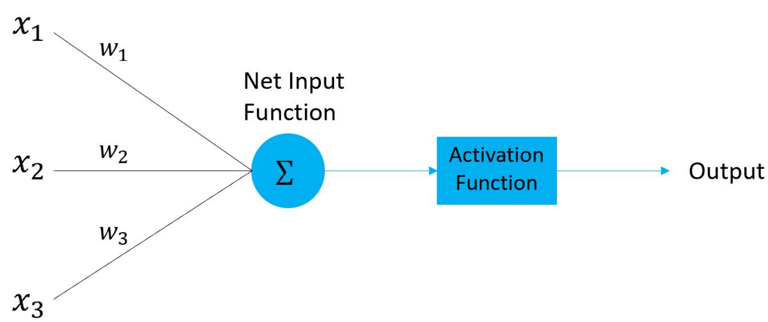
2 What is a Neural Network?

A neural network like any learning machine is designed to process data given to it and then use that data to make predictions. A feedforward neural network, as can be seen below, is comprised of three parts: an input layer, hidden layers and an output layer.

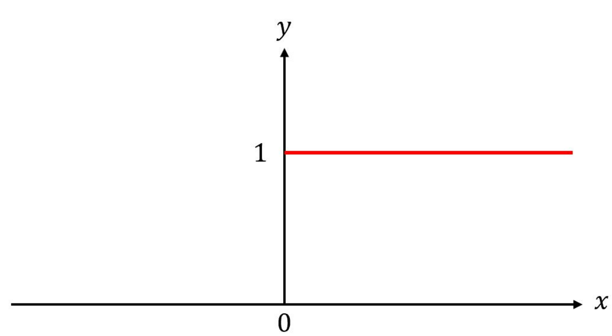


As the name suggests the input layer represented by the variable x_n simply consists of the input you provide to the neural network. For each x_n you connect it once with every neuron (denoted in blue) in the first hidden layer.

We shall now consider what happens at an individual neuron through the diagram below.



The neuron is fed the input data from each x_n as in the previous diagram. Here each connection is assigned its own weight w_n. A sum of these inputs is taken in the net input function. This is then passed through an activation function of some kind. A common example of such a function is a step function (an example of this is provided in a diagram below). The output of this activation function is then passed on to the next set of neurons as we can see in the first diagram. Each neuron is connected once to each neuron in the next layer so the inputs for subsequent layers are provided by the outputs of the previous neurons rather than x_n as we had in the case above. The data is passed through each hidden layer until eventually we reach the output layer.



Above is a simple example of a step function. In this example if given an input $x < 0$ the function will return 0 as output. If $x > 0$ then the output of the function will be 1. Here the output is only ever 0 or 1 which will not always be the case. If we chose a different activation function such as a sigmoid then continuous values between 0 and 1 are possible. We will not cover the workings of other activation functions but be aware they exist.

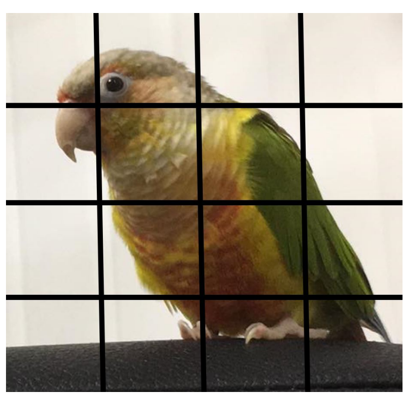
Each neuron in the output layer receives input through connections to each input in the last hidden layer. If we are using a softmax activation function, the output layer should consist of the same number of neurons as there are outcomes in the data we are observing. For instance, if we are looking at whether a picture is of a rabbit or a fox then there are of course only two outcomes. In this case the output layer must be comprised of two neurons. The output of each of these neurons is a number from 0 to 1 which represents the probability of that outcome. The sum of the outputs must therefore be equal to 1 so we don't end up breaking the laws of probability. As our prediction we choose the outcome with the highest value associated with it since that will be the outcome with the highest probability.

The number of input variables will depend of course on the data set. The number of neurons and hidden layers is something that you'll have to adjust depending on the problem at hand. It is possible to have one hidden layer though most problems you'll encounter are too complicated to be solved by just one. Neural networks with more than one hidden layer are known as deep neural networks.

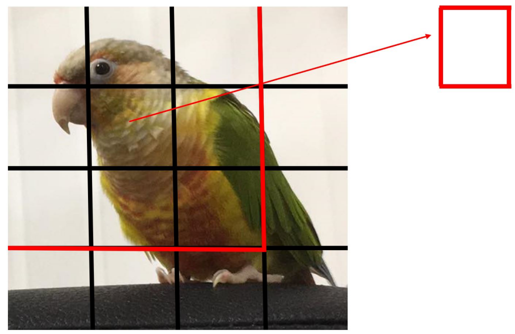
3 Convolutional Neural Networks

In addition to the basic feedforward neural network there are other types of neural network. One such network we will explore is known as the convolutional neural network. These are often used in processing images with complicated features that a standard feedforward network may not be able to discern.

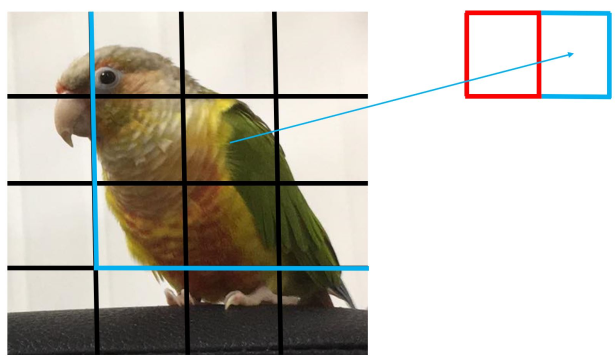
We begin by considering an image we want to use to train our network. The image is that of a fine upstanding member of green-cheeked conure parrot species. To recognise a bird the network will need to identify features that make a bird a bird. For instance, it should look for feathers, wings and a beak. The image is first broken down into pixels. To simplify our pixels will just be squares on an 8×8 grid though in practice you'll want more dimensions for an image to truly pick out meaningful features.



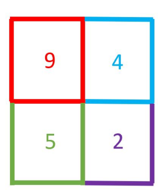
Convolution begins when we take a window an $n \times m$ square of pixels from our image and begin to extract features from it. The features in the window are used to create a single pixel.



Then the window slides along the image so that features from different parts of the image can be observed. From this we get a second pixel.



This is done until the whole image has been mapped to pixels on whats called a feature map. Below is an image of our complete featuremap for the image of the conure. Each pixel has a value determined by the features from the window which it was extracted from. The values chosen here are merely placeholders so don't worry about what they mean.

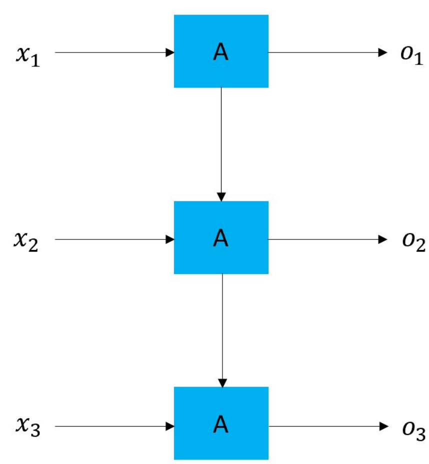


With the convolution phase concluded we now begin pooling our featuremap. The most commonly used type of pooling is max pooling. Essentially, we take windows of our featuremap like we did for the original image and select the maximum value of that in that window. That will be the value of the pixel that represents that region in a new featuremap. Once we've pooled the entire featuremap we then have our output to go to the next layer. Together convolution and pooling constitute a hidden layer. These should be connected to further convolutional layers or to fully connected layers as we saw previously with the standard feedforward neural network.

4 Recurrent Neural Networks

Previously the order in which our data was input did not matter. However, there are certain instances where we want to attribute some meaning to the position of the data. When considering exchange rates for instance often the dates will hold information that our network might need to make any real sense of the data. Any natural language processing relies on accounting for the position of the data. For instance, consider the two sentences: "the bird flew onto my shoulder" and "the shoulder flew onto my bird" both consist of exactly the same words but have a different meaning due to the different positioning of those words.

Below we have an example of what a layer within a recurrent neural network might look like in the diagram below.



Here x_n denotes the input data, A denotes the activation function of the cell and o_n the cell output. So how has the position or ordering of the data been accounted for? First, notice that the layer isn't fully connected so each x_n is assigned to one neuron. More importantly, each neuron provides input to the next neuron along. This means the first neuron can only really feel the effect of the first x_n whereas the last neuron will indirectly receive inputs from all the x_n. This now means that the position of the data is now ultimately going to effect the results.