CNN

Introduction:
Convolutional Neural Networks (CNN) are everywhere. It is arguably the most popular deep learning architecture.

Hopefully by end of this session, we will have a fairly better understanding of CNN and how it works.

<u>CNN:</u>
There is an input image that we're working with. We perform a series convolution + pooling operations, followed by a number of fully connected layers. If we are performing multiclass classification the output is softmax. We will now dive into each component.

<u>Input Layer:</u>
Computers see images using pixels. Pixels in images are usually related. For example, a certain group of pixels may signify an edge in an image or some other pattern.
Convolutions use this to help identify images.

<u>Convolution Layer:</u>
The main building block of CNN is the convolutional layer. Convolution is a mathematical operation to merge two sets of information. In our case the convolution is applied on the input data using a *convolution filter* to produce a *feature map*.

Feature map and **activation map:**
They mean exactly the same thing. It is called an activation map because it is a mapping that corresponds to the activation of different parts of the image, and also a feature map because it is also a mapping of where a certain kind of feature is found in the image. A high activation means a certain feature was found.

Filters
In neural network terminology, the learned filters are simply weights, yet because of the specialized two-dimensional structure of the filters, the weight values have a spatial relationship to each other and plotting each filter as a two-dimensional image is meaningful (or could be).

We typically use 3x3 filters, but 5x5 or 7x7 are also used depending on the application.

This is the most variable parameter, it's a power of two anywhere between 32 and 1024. Using more filters results in a more powerful model, but we risk overfitting due to increased parameter count. Usually we start with a small number of filters at the initial layers, and progressively increase the count as we go deeper into the network.

This was an example convolution operation shown in 2D using a 3x3 filter. But in reality these convolutions are performed in 3D. In reality an image is represented as a 3D matrix with dimensions of height, width and depth, where depth corresponds to color channels (RGB). A convolution filter has a specific height and width, like 3x3 or 5x5, and by design it covers the entire depth of its input so it needs to be 3D as well.

Padding and Stride:
When a filter traverses over the image, pixels in the middle are used more often than the pixels on corner or edges. Because of this the information on the borders of the images are not preserved or transfered to next layer. Could lead to overfitting.

Stride specifies how much we move the convolution filter at each step. We can have bigger strides if we want less overlap. This also makes the resulting feature map smaller since we are skipping over potential locations.

Pooling:
After a convolution operation we usually perform *pooling* to reduce the dimensionality. This enables us to reduce the number of parameters, which both shortens the training time and combats overfitting. Pooling layers downsample each feature map independently, reducing the height and width, keeping the depth intact.

Activation Function:
The activation function is the non linear transformation that we do over the input signal and they help to decide if the neuron would fire or not.

Fully Connected Layer:
Softmax function calculates the probabilities distribution of the event over 'n' different events.

Example 1. For example, suppose that the input volume has size [32x32x3], (e.g.

an RGB CIFAR-10 image). If the receptive field (or the filter size) is 5x5, then each neuron in the Conv Layer will have weights to a [5x5x3] region in the input volume, for a total of 5*5*3 = 75 weights (and +1 bias parameter).

This pattern was to be expected, as the model abstracts the features from the image into more general concepts that can be used to make a classification. Although it is not clear from the final image that the model saw a bird, we generally lose the ability to interpret these deeper feature maps.