Week 1 Assignment

1. What are Channels and Kernels (according to EVA)?

Answer -

a. Channels - Channel is a ***container of a specific information***.

Example - if we take a image composed of RGB; R-channel contains information pertaining to red color intensity like wise G(green) and B(blue) channels. Incase of english language, we can say we have 52 channels (alphabets in both cases excluding symbols)

b. Kernels - They are used to apply some/any desired effect(s) on the target. This resultant process is called Convolution. Generally its a square matrix having odd number of rows/columns. Typically they are matrices.

Example - $\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$. It averages the information of a particular pixel with its surrounding pixels

They are also called as Filters (given above example is blur filter).

In general, in any neural network their values are learnt during model training.

---

2. Why should we (nearly) always use 3x3 kernels?

Answer - As said like above(Kernel question) they are odd ordered matrices. They holds the ***'learn-able'*** parameters or can be said as ***'weights'*** of a layer. The weights are learnt during training of any Neural network by means of back-propagation.

While designing a neural network, each layer will be convoluted with a kernel

Parent layer --> convolution operation --> Child layer

For each 3x3 convolution operation the input layer size decreases by 2 (assuming zero padding and stride 1)
Eg - input layer size is 224x224
Conv 1(3x3) => 222x222
Conv 2(3x3) => 220x220

For each 5x5 convolution operation the input layer size decreases by 4 (assuming zero padding and stride 1)
Eg - input layer size is 224x224
Conv 1(5x5) => 220x220

In former case (3x3), we need to perform 2 convolution operations to achieve the same size as former case (5x5)

As iterated earlier, as these are learn-able parameters, if we calculate number of parameters to learn

3x3 convolution	5x5 convolution
Iteration 1 = 9 parameters Iteration 2 = 9 parameters Total Parameters = 18	Only 1 Iteration 1 = 25 parameters Total Parameters = 25

As 3x3 convolution required less parameters to train. So its always better to perform convolution with 3x3 kernels at the cost of addition of more convolution layers.

---

3. How many times to we need to perform 3x3 convolutions operations to reach close to 1x1 from 199x199 (type each layer output like 199x199 > 197x197...)

Answer - Need to perform convolution 99 times. Total 100 layers are present including parent and final layer

199x199
197x197
195x195
193x193
191x191
189x189
187x187
185x185
183x183
181x181
179x179
177x177
175x175
173x173
171x171
169x169
167x167
165x165
163x163
161x161
159x159
157x157
155x155
153x153
151x151
149x149
147x147
145x145
143x143
141x141
139x139
137x137

135x135
133x133
131x131
129x129
127x127
125x125
123x123
121x121
119x119
117x117
115x115
113x113
111x111
109x109
107x107
105x105
103x103
101x101
99x99
97x97
95x95
93x93
91x91
89x89
87x87
85x85
83x83
81x81
79x79
77x77
75x75
73x73
71x71

69x69
67x67
65x65
63x63
61x61
59x59
57x57
55x55
53x53
51x51
49x49
47x47
45x45
43x43
41x41
39x39
37x37
35x35
33x33
31x31
29x29
27x27
25x25
23x23
21x21
19x19
17x17
15x15
13x13
11x11
9x9
7x7
5x5

3x3 1x1

---

4. How are kernels initialized?

Answer - Kernels are weight matrices and if we assume a 100 or any large number layer neural network. In order to complete a single forward pass we' ll have to perform a matrix multiplication between layer inputs and weights at each of the 100 layers.

While we do so, if we initialize with any random values the resultant number after 100 passes will be very large and might not fit in any of data(long, float) type.

At the same time, if we initialize with a small value less than 1 after 100 passes of matrix multiplication the number might be very near to zero(0, 1E-9)

In either of the cases the training will not happen.

Key properties -

A. **Random** - symmetry breaking. If all the neurons have the same weights, they will produce the same outputs and we won't be learning different features

B. **Mean zero distribution**, common practice in machine learning is to zero-center or normalize the input data, such that the raw input features (for image data these would be pixels) average to zero.

C. **Variance/Standard deviation** - Neither too large nor too small. If too large we face exploding gradient, if too small dimnishing gradients.

Source - https://stats.stackexchange.com/questions/200513/how-to-initialize-the-elements-of-the-filter-matrix

---

5. What happens during the training of a DNN?

Answer - The goal of neural network is to map the input to output given on training samples.

The The training process involves finding a set of weights and biases for the network that proves to be good, or good enough, at solving the specific problem.

As CNN / DNN contains several layers in between input and output layer identifying the weights in first pass is difficult.

So, the weights of neurons are initialized randomly (following the question 4 process) and output is calculated, the deviation from the correct_output from calculated_output is **error**.
The error is backpropagated throughout all the layers and will evolve the weights of the layers in order to reduce the error for next set of iterations. The training will continue till we reach desired error or accuracy depending on the problem statement.