1. Why don't we start all of the weights with zeros?

2. Why is it beneficial to start weights with a mean zero distribution?

3. What is dilated convolution, and how does it work?

4. What is TRANSPOSED CONVOLUTION, and how does it work?

5.Explain Separable convolution

6.What is depthwise convolution, and how does it work?

7.What is Depthwise separable convolution, and how does it work?

8.Capsule networks are what they sound like.

9. Why is POOLING such an important operation in CNNs?

10. What are receptive fields and how do they work?

Answer:

1. Starting all weights with zeros would result in all neurons computing the same output. This is because, during backpropagation, all gradients would be the same and the weights would be updated in the same way, causing the network to behave identically.
2. Starting weights with a mean zero distribution helps break the symmetry in the network. If all weights start with the same value, they will all update in the same way during backpropagation, leading to symmetry. By starting with a mean zero distribution, weights can be initialized with random values, and the network can learn different features for different neurons.
3. Dilated convolution is a type of convolution that allows for a larger receptive field without increasing the number of parameters in the network. It works by skipping over some input values, effectively increasing the spacing between the kernel elements. This creates a larger receptive field while maintaining the same number of parameters.
4. Transposed convolution, also known as deconvolution, is an operation used to increase the spatial resolution of the output of a convolutional layer. It involves using a learned kernel to upsample the input, resulting in an output with a larger spatial dimension. It is often used in the decoder portion of an autoencoder or in the upsampling step of a segmentation network.
5. Separable convolution is a type of convolution that splits the standard convolution operation into two separate operations: depthwise convolution and pointwise convolution. Depthwise convolution applies a single filter to each input channel separately, producing a set of feature maps. Pointwise convolution then applies a set of 1x1 convolutional filters to these feature maps to produce the final output. This approach can reduce the computational cost of convolutional layers while still maintaining good performance.
6. Depthwise convolution is a type of convolution that applies a separate filter to each input channel, producing a set of output feature maps. It is typically followed by a 1x1 convolution to reduce the number of channels in the output. This approach can reduce the computational cost of convolutional layers while still maintaining good performance.
7. Depthwise separable convolution combines depthwise convolution and pointwise convolution to create a more efficient convolutional layer. It involves applying depthwise convolution first to capture spatial features, followed by pointwise convolution to combine the spatial features across channels. This approach can significantly reduce the number of parameters and computational cost of the convolutional layer while still maintaining good performance.
8. Capsule networks are a type of neural network architecture designed to better handle hierarchical relationships between visual features. They involve replacing standard scalar outputs from convolutional layers with "capsules" that output vectors that represent different properties of the input. These capsules are then combined in a way that allows the network to learn more complex relationships between features.
9. Pooling is an important operation in CNNs because it helps to reduce the dimensionality of the output from convolutional layers while still preserving important features. This helps to reduce the computational cost of the network and prevent overfitting. Additionally, pooling can help to make the network more robust to small translations in the input image.
10. Receptive fields refer to the area of the input image that is used to calculate the output of a neuron. The receptive field size of a neuron is determined by the size of the kernel used in the convolutional layer that precedes it. As the input passes through multiple convolutional layers, the receptive field size of neurons in later layers grows, allowing them to capture more complex features that span a larger portion of the input image.