Computer Vision_DiscussionQuiz_04

1. No, stacking more layers cannot provide a better network. Increase in depth of the network would result in vanishing gradient problem and decreases accuracy. With vanishing gradient problem, the correct values are not passed, values get diminished. And this rapid decrease in accuracy would lead to a saturation point. So, stacking more layers cannot provide a better network. The degradation problem that's discussed in paper-1 is of a shallow architecture, which has deeper counterpart that adds more levels on it. The deeper model may be solved via construction: the additional levels are identity mapping, while the remaining layers are copied from the learnt shallower model. Because of the availability of this built solution, a deeper model should not create a bigger training error than its shallower equivalent.

2. In neural networks, overfitting tends to happen often. This overfitting in deep neural networks affects training error. So, when overfitting occurs, the training error decreases. But in this network, the accuracy is reducing, and this leads to increase in training error, so with this increase in training error in this network, we could say that the degradation problem is not due to over-fitting.

3. The behavior of a model with optimal identity functions in ResNet setup is, it drives the residual to zero efficiently and comfortably if an identity mapping was optimum than to fit an identity mapping with a stack of nonlinear layers.

4. a. Methods we can use to add a residual connection with same dimension are using Identity blocks. Identity blocks are used when the input and output size is same. This block consists of a shortcut connection, same input and output size. Input is considered as 'X' and output as F(X) + X. So, to add residual connection with same dimension we use Identity blocks.

   b. To add a residual connection with different dimensions, we use Convolutional block. In this block, a convolutional layer is added in the shortcut path. This layer is added to make the input size same like output size. So, just like in identity block, this also consists of shortcut connection, and a convolutional layer is added to this skip connection. We match the output size in two ways: by either padding the input volume or performing 1x1 convolution. In both these cases, the stride value is going to be 2.

5. ResNet achieves a smaller number of FLOPs compared to VGGNet even with a deeper model than VGGNet because in VGGNet there are lots of filters unlike in ResNet, and in ResNet we use average pooling that this helps in reduction of the image size and this would also lead to reduction of number of FLOPs. And in ResNet, in convolutional layers there are with stride value as 2, this would also impact the reduction of size and helps in reduction of FLOPs. So, ResNet has a smaller number of FLOPs compared to VGGNet even with a deeper model.

6. Equation 4 of paper(2) has some interesting characteristics.

    i. Any deeper unit's input feature, L, may be represented as any shallower unit's input feature, l plus a residual function, F. This means the model is in a residual state between any two units L and l.

    ii. The summing of the outputs of all previous residual functions is a property of every deep unit L.

    iii. The backward propagation features of Equation (4) are also rather great.

7. Shortcut connections that are used in paper-2 are Identity or original block, constant scaling, exclusive gating, shortcut only gating, convolutional shortcut, dropout shortcut. Original block is used when dimensions are same, and in constant scaling, we use a scalar of 0.5 and constantly scale our residual function. In exclusive gating, we use a gating mechanism, so the weights and biases are followed by sigmoid function. In shortcut only gating, the residual function is not scaled, the shortcut path is gated. In convolutional shortcut, we use 1x1 convolutional shortcut connections. In dropout shortcut, we experiment with dropout, and there'll be a scaling of 0.5. I suppose original shortcut connection is the best connection out of all as there are not many complicities and manipulations, so original shortcut connection is best out of all.

8. When the residual function is scaled down, it fails to converge; nevertheless, when it is scaled up, it succeeds. The training error in the scaled residual function is higher than in the original ResNet, implying that scaling down the shortcut signal makes optimization difficult.

9. ResNet cannot help in obtaining better accuracy when the model is not deep as if the model is deep, ResNet helps in degradation problem and this leads to improving accuracy, but if the model is not deep, it doesn't create the degradation problem. So, this may not help improving accuracy.

10.
    a. Residual building block: Residual building block is nothing, but the input value is sent to the output through a shortcut connection or skip connection. Suppose, let 'X' is the input then in other networks 'Y = F(X)' is the output, but in residual network, if 'X' is the input then 'Y=F(X)+X' is the output, as we send the input to the output and F(X) is the cost function. We do this to reduce vanishing gradient descent problem. Our target is to make F(X)=0, such that output would be equal to input.
    b. Solution space: Solution space is nothing but the options where the model output would be in. For example, if we the model has one weight, the weight can be in

range of negative infinite values to positive infinite values. So, with using this solution space the weights would provide expected results.

c. Bottleneck architecture: Here, normally we use two layers in building block, in bottleneck architecture we use three layers instead of two. The first layer is a 1x1 convolutional layers, next is 3x3 convolutional layer and the third one is 1x1 convolutional layer. We use the first 1x1 convolutional layer to reduce the size and the other 1x1 convolutional layer to resize and increase the dimension.
d. Pre-activation: In the building block of residual network, we consider the Batch Normalization and Rectified Linear Unit activation functions that are performed as pre-activation functions. These pre-activation functions help win training the model effectively. If we use only Rectified Linear Unit to activate the weights initially, then it's called as RELU only pre-activation and if we use both RELU and Batch Normalization activation functions to activate weights in the start, then this is called as full pre-activation.
e. Post-activation: If there is element wise addition, then the next performed activation is called as post-activation. Even for post-activation we use batch normalization and Rectified Linear Unit. So, depending on the element wise addition, we call them pre-activation or post-activation.

Note: https://arxiv.org/pdf/1603.05027.pdf
https://arxiv.org/pdf/1512.03385.pdf