CS577: Assignment 4

Jane Downer
Department of Computer Science
Illinois Institute of Technology

October 8, 2022

1. Given

$$R = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad G = \begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix} \quad f = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Result of convolution **without** zero-padding:

Row 1:

$$\begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}\right) \end{bmatrix}$$

$$+ \qquad\qquad +$$

$$45 \qquad\qquad 45$$

Row 2:

$$\begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 3 & 3 & 3 \\ 4 & 4 & 4 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 3 & 3 & 3 \\ 4 & 4 & 4 \end{bmatrix}\right) \end{bmatrix}$$

$$+ \qquad\qquad +$$

$$54 \qquad\qquad 54$$

$$\boldsymbol{result} = \begin{bmatrix} \mathbf{45} & \mathbf{45} \\ \mathbf{54} & \mathbf{54} \end{bmatrix}$$

2. Given the same information above, result of convolution **with** zero-padding:

Row 1:

$$\begin{bmatrix} sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 2 \\ 0 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 2 & 2 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 2 & 2 & 0 \\ 2 & 2 & 0 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & 2 & 0 \end{bmatrix}\right) \end{bmatrix}$$

$$+ \qquad\qquad +$$

$$18 \qquad\qquad 27$$

$$+ \qquad\qquad +$$

$$27 \qquad\qquad 18$$

Row 2:

$$\begin{bmatrix} sum\left(\begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 2 & 2 \\ 0 & 2 & 2 \\ 0 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 1 & 1 \\ 0 & 2 & 2 \\ 0 & 3 & 3 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 2 & 2 & 0 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 1 & 1 & 0 \\ 2 & 2 & 0 \\ 3 & 3 & 0 \end{bmatrix}\right) \end{bmatrix}$$

$$+ \qquad\qquad +$$

$$30 \qquad\qquad 45$$

$$+ \qquad\qquad +$$

$$45 \qquad\qquad 30$$

Row 3:

$$\begin{bmatrix} sum\left(\begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 2 & 2 \\ 0 & 2 & 2 \\ 0 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 0 & 2 & 2 \\ 0 & 3 & 3 \\ 0 & 4 & 4 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 3 & 3 & 3 \\ 4 & 4 & 4 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 2 \\ 3 & 3 & 3 \\ 4 & 4 & 4 \end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 2 & 2 & 0 \end{bmatrix}\right) \\ sum\left(\begin{bmatrix} 2 & 2 & 0 \\ 3 & 3 & 0 \\ 4 & 4 & 0 \end{bmatrix}\right) \end{bmatrix}$$

$$+ \qquad\qquad +$$

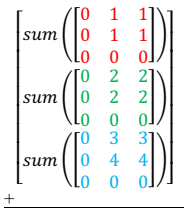
$$36 \qquad\qquad 54$$

$$+ \qquad\qquad +$$

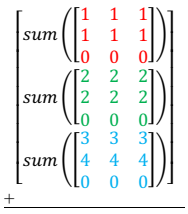
$$54 \qquad\qquad 36$$

(continued on next page)

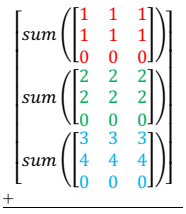
Row 4



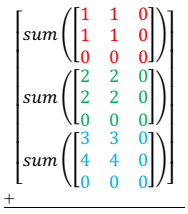
26



39



39

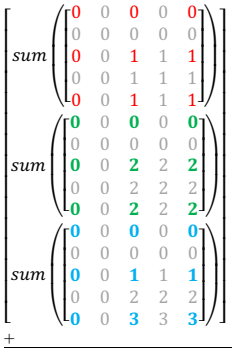


26

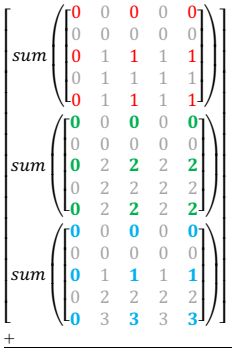
$$\boldsymbol{result} = \begin{bmatrix} \mathbf{18} & \mathbf{27} & \mathbf{27} & \mathbf{18} \\ \mathbf{30} & \mathbf{45} & \mathbf{45} & \mathbf{30} \\ \mathbf{36} & \mathbf{54} & \mathbf{54} & \mathbf{36} \\ \mathbf{26} & \mathbf{39} & \mathbf{39} & \mathbf{26} \end{bmatrix}$$

3. Given the same information, result when using zero-padding and 2-dilated convolution:

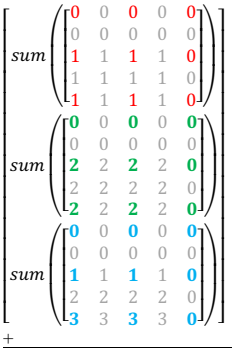
Row 1:



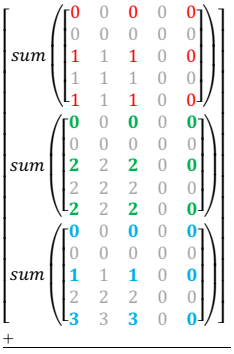
20



20

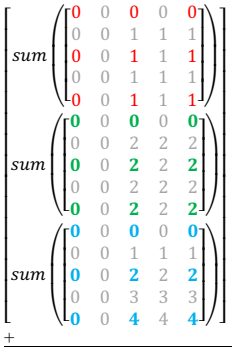


20

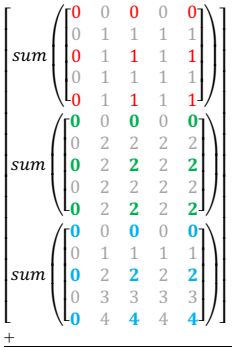


20

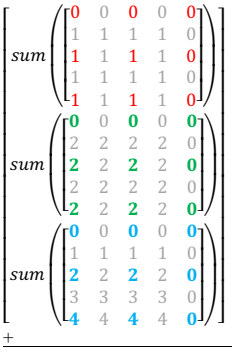
Row 2:



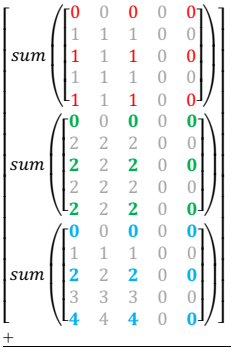
24



24

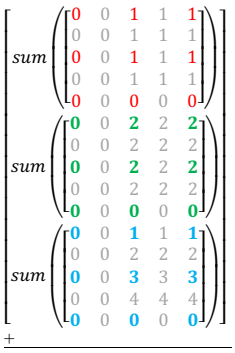


24

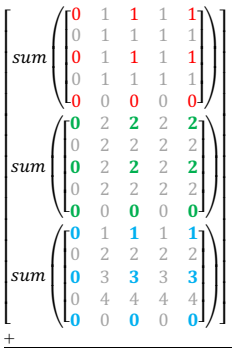


24

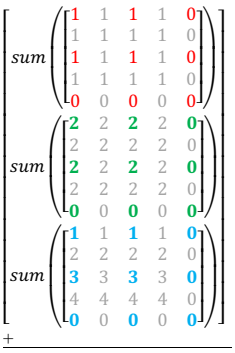
Row 3:



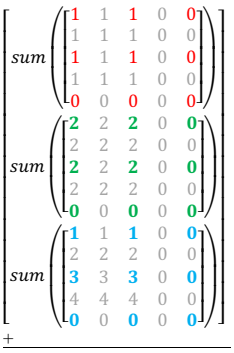
20



20



20



20

(continued on next page)

Row 4:

$$\begin{bmatrix} sum\left(\begin{bmatrix}0&0&1&1&1\\0&0&1&1&1\\0&0&1&1&1\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}0&0&2&2&2\\0&0&2&2&2\\0&0&2&2&2\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}0&0&2&2&2\\0&0&3&3&3\\0&0&4&4&4\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix}0&1&1&1&1\\0&1&1&1&1\\0&1&1&1&1\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}0&2&2&2&2\\0&2&2&2&2\\0&2&2&2&2\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}0&2&2&2&2\\0&3&3&3&3\\0&4&4&4&4\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix}1&1&1&1&0\\1&1&1&1&0\\1&1&1&1&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}2&2&2&2&0\\2&2&2&2&0\\2&2&2&2&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}2&2&2&2&0\\3&3&3&3&0\\4&4&4&4&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \end{bmatrix} \quad \begin{bmatrix} sum\left(\begin{bmatrix}1&1&1&0&0\\1&1&1&0&0\\1&1&1&0&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}2&2&2&0&0\\2&2&2&0&0\\2&2&2&0&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \\ sum\left(\begin{bmatrix}2&2&2&0&0\\3&3&3&0&0\\4&4&4&0&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\right) \end{bmatrix}$$

$$+ \qquad + \qquad + \qquad +$$

$$24 \qquad 24 \qquad 24 \qquad 24$$

$$\boldsymbol{result} = \begin{bmatrix} \mathbf{20} & \mathbf{20} & \mathbf{20} & \mathbf{20} \\ \mathbf{24} & \mathbf{24} & \mathbf{24} & \mathbf{24} \\ \mathbf{20} & \mathbf{20} & \mathbf{20} & \mathbf{20} \\ \mathbf{24} & \mathbf{24} & \mathbf{24} & \mathbf{24} \end{bmatrix}$$

4. Convolution can be thought of as measuring the similarity between the filter (the "template") and different portions of the image. For example, if the filter has high values along the diagonal, and if there is a patch of the image that has diagonal features, then convolving the image with that filter will result in a high value, because it "matches" the "template".
5. We don't know how big or small an object in an image will be, and therefore, how much of the image a filter should cover. We can obtain several versions of the image by repeatedly shrinking the size – this allows us to use a constant filter size while varying the amount of the image that a filter can "see" at a given time. We do not need to know the size of an object in an image in order for the filter to see the entire thing, because we can accomplish that by adjusting the scale the image.
6. Reducing spatial resolution allows us to have multi-scale pyramid analysis – however, we want to retain all the information we can to help in the classification task. To do so, when the image is scaled down, we increase the number of channels to such a degree that allows us to retain the same number of coefficients. For example, if a 4x4, single-channel image is scaled down to 2x2, then we can retain the same number of coefficients by increasing the number of channels from 1 to 4:

   4x4x1 = 2x2x4 = 16 coefficients
7. Input tensor: $128 \times 128 \times 32$
   Filters: $3 \times 3 \times 32$ each (16 total)
   Stride: 1
   Padding: 0

   Width: $\left\lfloor \frac{W_{in}-k[0]+2p}{s} \right\rfloor + 1 = \left\lfloor \frac{128-3}{1} \right\rfloor + 1 = 126$

   Height: $\left\lfloor \frac{H_{in}-k[1]+2p}{s} \right\rfloor + 1 = \left\lfloor \frac{128-3}{1} \right\rfloor + 1 = 126$

   Depth: $\left(\left\lfloor \frac{D_{in}-k[2]+2p}{s} \right\rfloor + 1\right) \cdot \#\, filters = \left(\left\lfloor \frac{32-32}{1} \right\rfloor + 1\right) \cdot 16 = 16$ Learned about using floor function from source [1]

   Output shape = $\mathbf{126 \times 126 \times 16}$

8. Input tensor: $128 \times 128 \times 32$
   Filters: $3 \times 3 \times 32$ each (16 total)
   Stride: 2
   Padding: 0

   Width: $\left\lfloor \frac{W_{in}-k[0]+2p}{s} \right\rfloor + 1 = \left\lfloor \frac{128-3}{2} \right\rfloor + 1 = 63$

   Height: $\left\lfloor \frac{H_{in}-k[1]+2p}{s} \right\rfloor + 1 = \left\lfloor \frac{128-3}{2} \right\rfloor + 1 = 63$

   Depth: $\left(\left\lfloor \frac{D_{in}-k[2]+2p}{s} \right\rfloor + 1\right) \cdot \#\, filters = \left(\left\lfloor \frac{32-32}{2} \right\rfloor + 1\right) \cdot 16 = 16$

   Output shape = $\mathbf{63 \times 63 \times 16}$
9. Given an input tensor T of shape $W \times H \times C_1$ and a filter F of shape $1 \times 1 \times C_2$, imagine the filter "looks" at elements $[T(i,j,c), \ldots, T(i,j,c+C_1)]$ in the input tensor. This results in the dot product of vectorized F and elements $[T(i,j,c), \ldots, T(i,j,c+C_1)]$. If $C_2 = 1$, the depth of the output will equal the depth of the input, because the filter will have to repeat this operation for each channel, resulting in C_2 different values for every (i,j). If $C_2 = C_1$, the depth of the output will be 1, since the filter will consider all depths simultaneously and produce a single projection (dot product) of those values onto the output space. Therefore, by increasing the depth of the filter, we can reduce the number of channels in the output.
10. In a convolution layer of a CNN, an input tensor is convolved with a filter. We extract individual patches/windows of the image corresponding to the size of filter, vectorize both the patch and filter and obtain the dot product. The filter slides across

the input tensor and repeats this process, and all dot products are included in the resulting activation map. As a result of padding, stride, and/or pooling, the layer's output tends to be smaller than its input, in terms of width and height – as we discuss above, we typically compensate by using multiple filters to increase the depth. Therefore, one characteristic of deeper layers is that they tend to produce tensors that are smaller (width/height) but deeper (number of channels). In addition, as we get deeper into the network, the repeated process of convolution and pooling results in emphasizing and combining features identified by previous layers. As a result, another characteristic of deeper layers is that their trained weights/filters reflect more complex and recognizable visual features than the earlier layers.

11. Given

$$\left[\begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix},\begin{bmatrix}2&2&2&2\\2&2&2&2\\2&2&2&2\\2&2&2&2\end{bmatrix},\begin{bmatrix}1&1&1&1\\2&2&2&2\\3&3&3&3\\4&4&4&4\end{bmatrix}\right] \quad f = 2 \times 2 \quad stride = 2$$

Result of max pooling: $\left[\begin{bmatrix}\mathbf{1}&\mathbf{1}\\\mathbf{1}&\mathbf{1}\end{bmatrix},\begin{bmatrix}\mathbf{2}&\mathbf{2}\\\mathbf{2}&\mathbf{2}\end{bmatrix},\begin{bmatrix}\mathbf{2}&\mathbf{2}\\\mathbf{4}&\mathbf{4}\end{bmatrix}\right]$

12. Pooling allows us to perform multiple scale analysis by allowing us to scale down the image. It also allows us to reduce the number of coefficients which speeds up training.
13. Data augmentation helps reduce overfitting and improve generalization by artificially increasing the number of examples seen by the model. Data augmentation is a good idea when a dataset is small.
14. Transfer learning is when we take a model trained for one task and incorporate it into a new model that's designed for another task. It helps save time and resources because we don't need to train from scratch. It is useful when our data is small, because performance is improved by what the source model has already learned (e.g., feature extraction, classification, etc).
15. We freeze the coefficients from the pre-trained network so they are not changed or destroyed when training the new layers.
16. With fine-tuning, we begin by training the fully-connected layers with the conv base coefficients frozen. After training the fully-connected layers, we unfreeze some of the top layers of the base network and jointly train the entire network. This allows us to constrain the search space for the first part of training, and then make minor changes to the pretrained layers after that.
17. A basic inception block takes the output from the previous layer and passes it through four parallel blocks – a 1x1 convolution, a 3x3 convolution, a 5x5 convolution, and 3x3 max pooling. The outputs of these blocks are concatenated, and that concatenated result is the output of the inception block. This allows us to look at multiple receptive fields at the same time. There is a modified inception block still has the same four blocks, except with 1x1 convolutions prior to the 3x3 and 5x5 conv blocks and after the 3x3 max pooling block. The inclusion of these 1x1 convolutions reduces the number of channels, which means it also reduces the number of parameters. There are further modifications of the inception block, all of which help with the same goal of viewing multiple receptive fields with a reduced number of parameters.
18. Sometimes deeper networks produce higher errors, because there are more degrees of freedom which makes it difficult to optimize. Essentially, because there are so many layers, the gradients vanish during backpropagation. Residual blocks incorporate "skip connections", which means the input from an earlier layer is passed through an identity function and used as an additional input to the current layer. This ensures that this earlier information isn't lost/destroyed. This process helps avoid a vanishing gradient, because during backpropagation through an identity function, the gradient is preserved. Another advantage of the residual block is that it makes it possible for the weights of a convolution to be all zeros, which means the input will just be passed through an identity function and not change. This allows the network to learn to eliminate convolution layers that aren't needed.
19. To visualize intermediate activations, take an existing model and create a list of the outputs of each layer. Create a new model by providing `existing_model.input` and `layer_outputs` as parameters, which results in a `keras.models.Model` instance. Call `run.predict()` with a sample image as input. This will return a list of arrays, and by calling the i^{th} element of that list, you are calling the i^{th} layer activation. Each individual layer activation is a tensor with one or more channels, and each of those channels is a 2D matrix. By plotting this matrix in matplotlib, we can display something that looks somewhat like the original image, but highlights features of the original image that a specific channel detects (i.e., all the diagonal lines in the image, all the edges in the image, etc).
20. We can think of filters as templates for specific visual features, and if the filter is similar to a specific window of the image, then the response to that filter will be large. Maximizing the response of the filter reflects the same goal as minimizing the loss. We can visualize the filters by plotting the input that either (a) maximizes the filter response (using gradient ascent), or (b) minimizes the loss (using gradient descent). We have covered gradient descent in previous assignments -- gradient ascent is a similar process to gradient descent, except it involves iteratively updating the input using the gradient and a step size until the results of convolution with that filter reaches a local maximum.
21. We can use the Grad-CAM (class activation map) algorithm to visualize the areas of an image that a layer thinks are important, given a particular category in the classification task. First, identify the node that outputs the score for that particular class (in the lecture example – 'elephant'). Compute the gradients of that output with respect to each channel of the layer; from those values, compute the pooled gradient (average gradient magnitude) for each channel of the layer. These pooled gradients can be thought of as a measure of how important the channel is, and we use them as weights to compute a weighted average of all channels. If we superimpose this result over the original image, it highlights the areas of the image that most contributed to the output score for that class.

References:

[1] https://towardsdatascience.com/a-comprehensible-explanation-of-the-dimensions-in-cnns-841dba49df5e