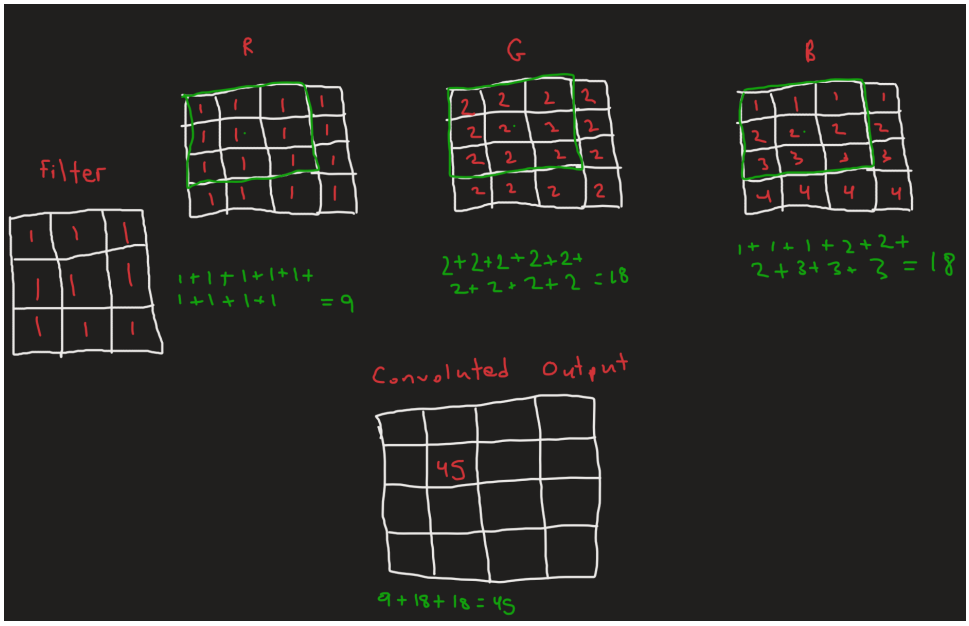
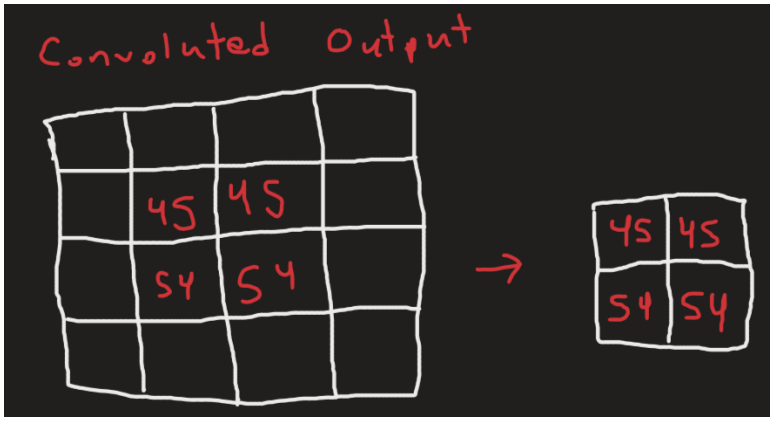
cs577 Assignment 4

Girish Rajani-Bathija
A20503736
Department of Computer Science
Illinois Institute of Technology
November 11, 2022

1.



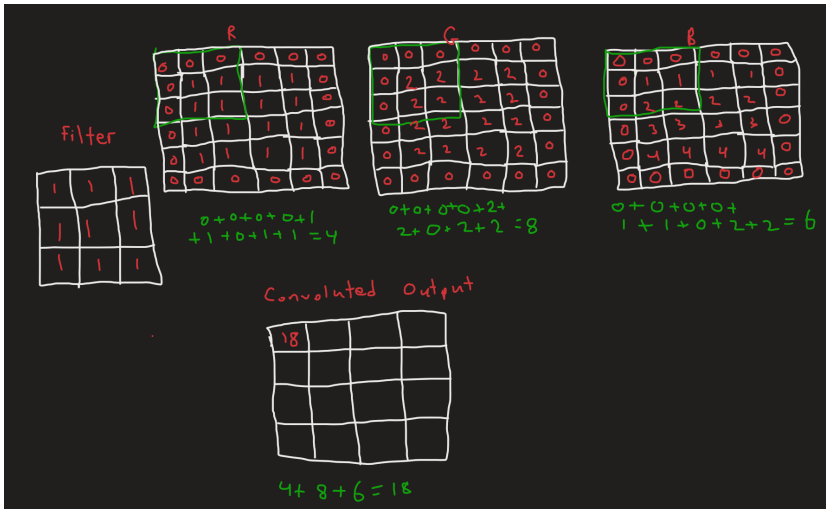
As shown above, in the Red color channel, we place the convolution filter on top of the image and take each corresponding pixel and filter, multiply them together and sum it up to get a value of 9. We perform this same operation on the Green and Blue color channels and get a value of 18 for each of them. Now, we have the convolution for each color channel so we add these values to get 45 and this is our value in the convoluted output. To get the full output, we move the filter to the next spot and perform this same operation for all other pixels in each channel:



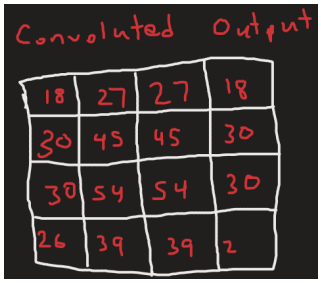
The above shows the final convoluted output and since we are not performing zero padding, so we ignore.

2.

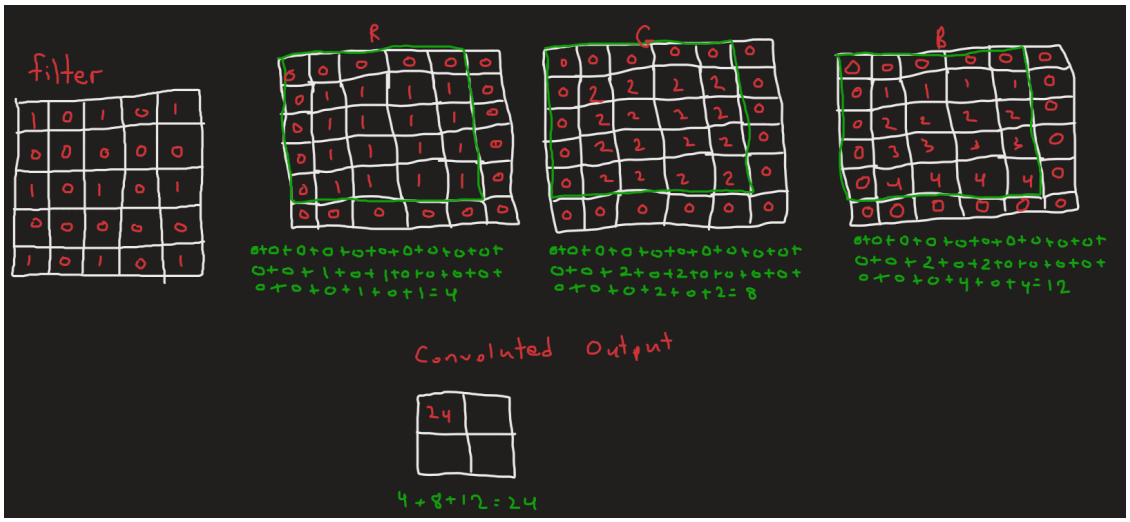
We perform the same steps as above to compute the convoluted output but instead of ignoring, we now perform zero-padding:



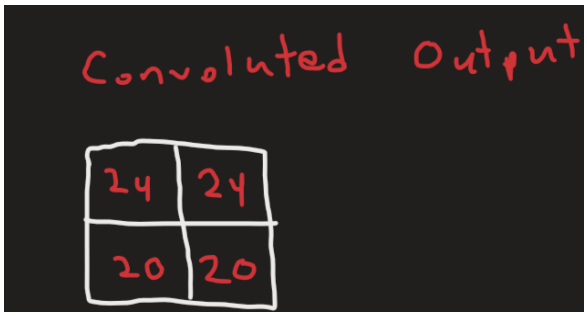
In the previous question, the filter was only able to be moved 2 times, resulting in a 2x2 but now with zero padding, we can move the filter 4 times hence getting a value in each pixel getting a 4x4 convoluted output. By performing the same steps, we can get the convoluted output for each pixel:



3.



The same image from Question 2 was used with the zero padding but now we change the filter using a dilation rate of 2. Since the filter is a 5x5 and the RGB image is a 6x6, we can only move the filter once after it has been placed, resulting in a 2x2 convoluted output. The computation shown above was done to compute the first pixel and the same operation was conducted to find the remaining:



4. Template matching interpretation of convolution:

In template matching, the convolution filter looks for similarity. When we apply the filter to an input image, we take the filter everywhere and multiply the corresponding elements to get the result output. With this interpretation, the filter amplifies in the result, the areas that are similar to it. It identifies areas that are similar to the convolution filter that is applied. What this template matching interpretation means is that we can specify some specific features in our convolution filter then when learning, it will detect those special patterns and this can allow us to perform classification problems better.

5. Explain how multiple scale analysis can be achieved with a fixed window size (using a pyramid).

In neural networks, another important aspect that classifies images is processing the images through multiple scale analysis. We can use pyramid analysis that is used in multiple scale analysis. This pyramid analysis allows us to have multiple scale analysis and instead of making a filter bigger and bigger, we can instead used a fixed size filter but make the image smaller and smaller. We keep resizing the image to make it smaller and this allows the fixed window size to cover a larger region in the new image. This is a better option than making the filter larger because we do not know the size of the image and this way, we reduce computation complexity.

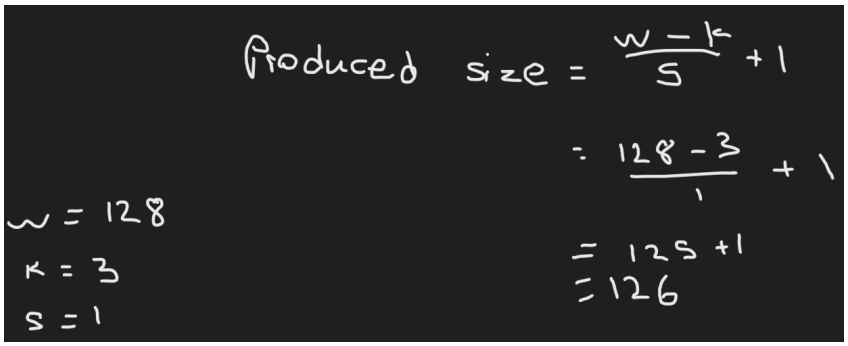
What we do during this pyramid is we take a portion of the image as the pyramid (for example every 4 pixels) and we can sample it to get another level and we repeat this process. This means that at each level, the filter is able to detect bigger and bigger objects.

6. Explain how to compensate for spatial resolution decrease using depth (number of channels) and the purpose for doing so.

As spatial resolution decreases, depth increases to compensate for reduced coefficients (We want to keep the same number of coefficients). The purpose is because when the spatial resolution decreases, we lose information (when we shrink the image, we see less details in them). We still want to shrink the image to have multi scale pyramid analysis but we don't want to lose the information so that is why we increase the depth to compensate for the spatial resolution decrease so that we don't lose information.

7. Resulting tensor when convolving without zero padding:

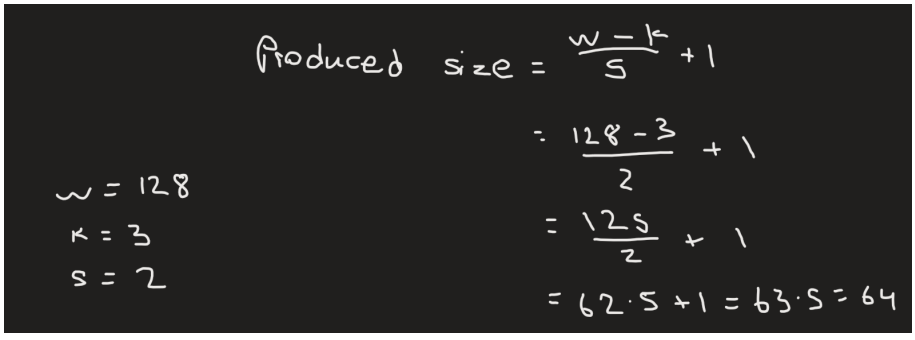
Input feature map - 128x128x32
3x3 input patches - 16x(3x3x32) convolution filters



Output feature map/Resulting tensor - 126x126x16

8. Resulting tensor when convolving without zero padding and stride of 2:

Input feature map - 128x128x32
3x3 input patches - 16x(3x3x32) convolution filters
Stride - 2



Output feature map/Resulting tensor - 64x64x16

9. Explain how the number of channels can be reduced using a 1x1 convolution

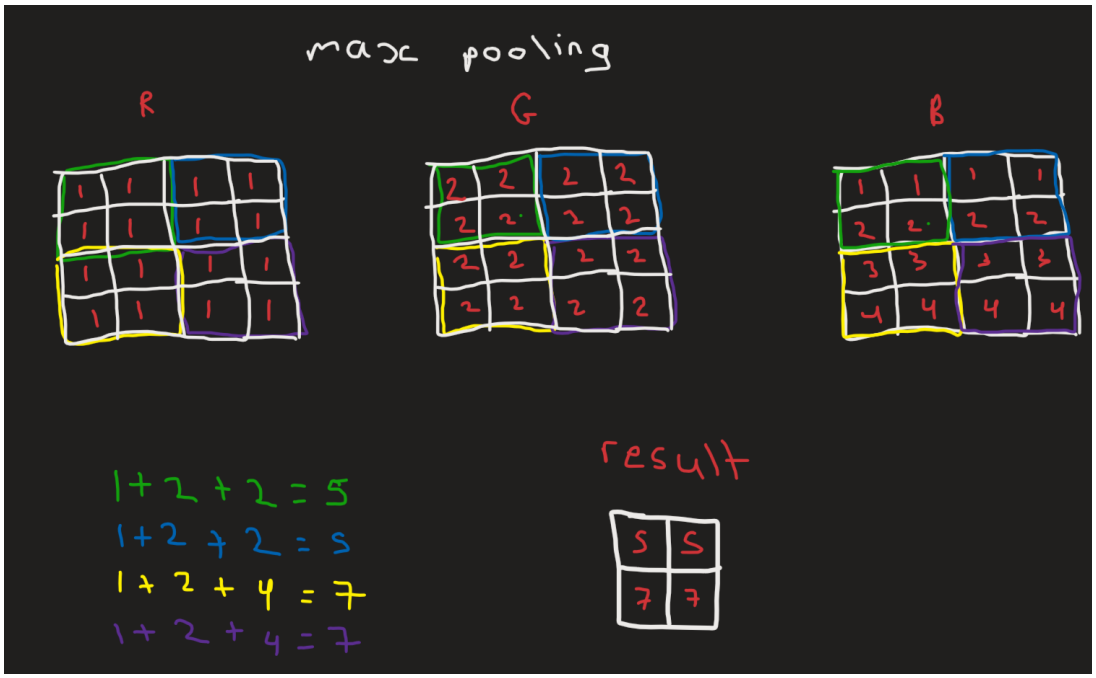
Using a 1x1 convolution means the kernel size is 1x1. When we have a convolution filter, it will span the entire depth so when we multiply the convolution filter by the image, we take the weighted sum of all the pixels within the depth. This will give 1 output (or 1 slice) when we use a single 1x1 convolution filter. Essentially, by selecting the number of 1v1 convolution filters, we can reduce the dimensions. For example, if we have a 28x28x64 tensor, by selecting 32 1x1 convolution filters and this results in a 28x28x32 tensor, hence reducing lowering the number of channels (reducing the number of coefficients). This allows us to control the complexity of the implementation.

10. Explain the interpretation of convolution layers and the difference between early and deeper convolution layers.

The convolution layers carry the main portion of the computation and processes data having a grid-like structure. As computed in previous questions, the convolution layers converts the pixels in its receptive field into a single value.

Early convolution layers would extract simple features such as learning directional derivatives (they look for edges in a particular direction). The deeper convolution layers would learn more complex patterns other than edges and direction, and as the layers go deeper and deeper, it learns to identify even more complex patterns (higher level features) that the early layers are not able to detect.

11. Result using max pooling with a 2x2 filter with a stride of 2:



12. Explain the purpose of pooling.

Pooling is used to reduce the size of the spatial size of the image. It allows us to downsample the spatial dimensions (width and height) of the image without changing the depth. For example, to go from a 128x128x16 to a 64x64x16. Pooling also helps to reduce the number of coefficients.

The process of this is similar to convolution in that it scans the image, take a window and produce a single value from that one window and repeat this process using a stride. So, we are basically scanning with a filter with stride and often, that stride is selected without filter overlap so if we use a 2x2 filter then we use a stride of 2 to avoid overlapping.

In pooling, we usually use 2x2 regions to reduce layer by 75% so if we have a 4x4 with 16 pixels then after pooling, it will become a 2x2 with 4 pixels (75% smaller image when counting the pixels).

13. Explain the purpose of data augmentation and when it is most useful.

Data augmentation allows us to take an input data and distort a bit to create many more examples so we can turn it, crop it, flip it, etc. The purpose of this is to allow for better generalization using the ImageDataGenerator as now we have a lot more samples to help in training and improve the performance. Data augmentation is most useful when collecting large amounts of data is difficult because now we can artificially create new synthetic data from existing data.

14. Explain the purpose of transfer learning and when it is most useful.

Transfer learning is the process of using a pre-trained convnet trained on a large dataset compared with small available data. For example, we can take the model that was used to train ImageNet object classification (having over 1 million samples) and use that to train the cats/dogs classification problem which we have fewer data available(both problems are classification so it does make sense to do this). Transfer learning reduces the amount of data required so the purpose of this is to achieve higher performance and save time/resources since we are reusing a pre-trained model. It is most useful when training with only a small amount of data.

15. Explain the need for freezing the coefficients of the pre-trained network.

Freezing the coefficients of the pre-trained network means not allowing the update of weights in the pre-trained network as the model is being trained. The reason for this is that when we use our new classifier that has been randomly initialized, during training, the weights in the pre-trained network will become modified and this can be a bad thing because the new classifier is randomly initialized so it is not good (untrained). During backpropagation, bad updates from the untrained classifier will result in negatively affecting the weights in the pre-trained network. Since the pre-trained network already has nice weights that have been trained, we freeze the coefficients here as to not update them to retain the nicely trained network.

16. Explain how the coefficient of a pre-trained network can be fine-tuned.

After training the fully connected layers, we unfreeze some top layers in conv-base and retrain to allow the model to fit the data (we do not unfreeze the lower levels of the conv-base because we assume that they do not change since they are low-level features such as edges, but the top layers can be fine-tuned). This fine-tuning can be done by the following:

1. Add custom network on top of trained layers
2. Freeze trained layers
3. Train custom network
4. Unfreeze top layers in the base network
5. Jointly train the custom network and unfrozen layers

17. Explain the purpose of inception blocks.

The purpose of inception blocks is to have multiple receptive fields. This means that instead of using 1 convolution filter, we can use multiple filters in a single block, concatenate them, and use that as input in the next layer. By introducing 1x1 convolutions, we can perform dimensionality reduction using inception blocks. By introducing 3x3 and 5x5 convolutions in inception block, the network can learn spatial patterns and detect features at different scales. So, the purpose of inception block is to allow the usage of different convolution filter sizes to learn spatial patterns at different scales. This results in benefits such as high-performance gain on CNN's, the ability to perform feature extraction from input data at different scales, etc.

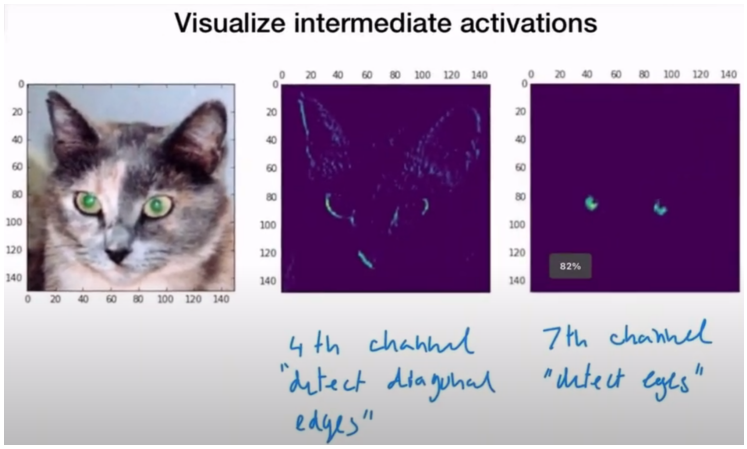
18. Explain the advantage of residual blocks.

When using residual blocks, it is easier to learn F(x) residual compared with H(x). This means that it learns deviation from identity instead of function. F(x) is derived by taking the input x and passing it through some convolution and we get the output F(x) and to get H(x), we add the input 'x' back into the output 'F(x)'. If we did not use residual blocks, and need to pass it through the convolution without destroying it, we would need to put a lot of effort to make the convolution learn not to destroy the x. However, with the identity link when using the residual block, it is easy because the x is directly given to us through the skip connection.

Additionally, skip connections in residual blocks help with vanishing gradients because as we push gradients through the network, we push the gradients through all paths (directly through the convolutions and also through the skip connection). When it is passed through the skip connection, it is passed without modification, meaning without zeroing it out (or without vanishing). That is how residual blocks help with vanishing gradients and since the gradients are passed directly through skip connections, this results in quicker training. Additionally, with residual blocks, the network can learn to zero blocks to eliminate un-needed layers because it will still have the skip connection to ensure that nothing is destroyed if we eliminate those layers. This connection allows another path for data to reach deeper parts of the network by skipping some layers which improves performance.

19. Explain how intermediate activations of convolution layers can be visualized given an input. What is the purpose for doing so?

To visualize the intermediate activations, we load the image that we want to take the activation to visualize, we add another dimension to make it a batch, normalize the image as per normal and to actually visualize the intermediate activation, we take the existing model, and create a new model with new outputs that will show the intermediate activations and since we now have multiple outputs, we need to use the “Model” instead of the “Sequential” class. We then run this new model on the loaded image to visualize what the network is doing for that image. It will return a list of all the intermediate activations and then we can examine each intermediate activation by plotting the channels of the layer to see what each channel is detecting.



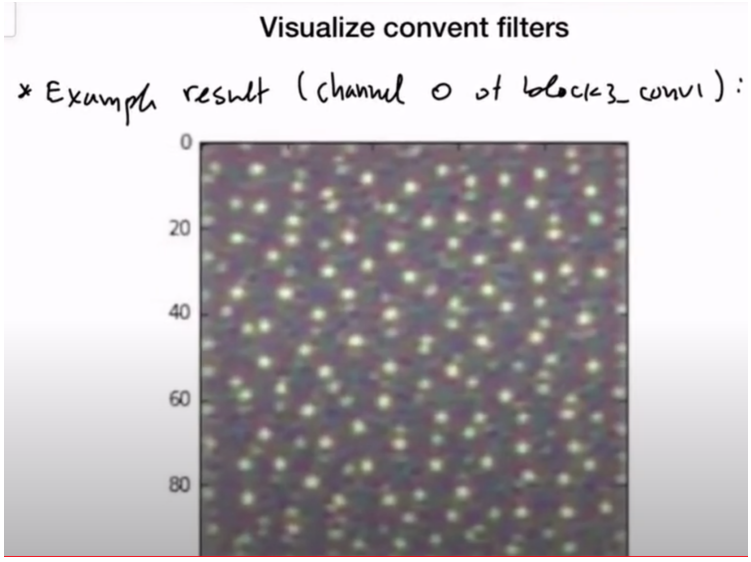
For example, as we can see above, we can see the input image, the cat and then we can see the visualization of the 4th channel to see exactly what the network is learning. In the 4th channel above, we can see that it is learning to detect diagonal edges and the 7th channel is learning to activate on the eyes within the first layer.

The purpose of visualizing the intermediate activations is to see what each filter does. We can see what the network is activating on or what the filters are doing. A reason for this is to see that the network is actually using useful information. This is because sometimes we see that a network gets good results but when we visualize to see what the network is activated on, we see that it is activated on irrelevant information. This can help us see which features in the image the network is generalizing too and if the network activates in some other region that does not contain what we are trying to classify then it is not a useful network because it needs to activate on the actual target in the image. Another purpose to visualize intermediate activations is to localize where the object is. This means that we can know exactly where the object is based on where there is a strong activation after visualizing.

20. Explain how the filter weight of the trained convolution layers can be visualized (using gradient ascent to find the input with maximal response). What is the purpose of doing so?

When visualizing the filter weight, these filters may be interpreted as templates that are being matched and we can say that if the template looks similar to the input, then the network will fire high values. To find the input that will corelate well with the different layers, we can use gradient ascent to find the input that will maximize the response of the filter. We want to know how we can change the input so that the response from this filter is higher and then find the input that maximizes the response and that is the template that the layer activates on. We will be visualizing the input that maximizes the response.

To visualize this, we load the model then define output and loss. We then compute the loss gradient with respect to the input, normalize the gradient vector, define a sub-graph to compute the value of the loss and gradient tensors given an input image. We then maximize the responsive measure with respect to the input using gradient ascent. These steps are all combined in a function to generate filter visualization from input layer name and filter index afterwhich we compute the pattern and plot it.



Showing example pattern that will maximize the output from channel 0 of block 3

The purpose of this is to see if there is a meaning to the weights and to see what the network has learned.

21. Explain how the heatmap of class activation can be visualized for a specific image and class. Explain how pooled gradients can be used to weight channels in this visualization. Explain the purpose of this visualization.

To visualize the class activation heatmap, we can use the Grad-CAM algorithm. This algorithm has the following steps:

1. Feed an image to the network
2. Compute gradients at selected output node with respect to each channel of the target layer where activation is to be computed (“conv_5”)
3. Compute the average gradients of each channel
4. Add the activations of each channel weighted by their average gradient magnitude
5. Superimpose activation on input image (heatmap)

Pooled gradients can be used to weight channels in this visualization in that we can sum the gradients with respect to each channel and if we have high gradients, it will indicate that the loss is strongly dependent on that channel therefore making that channel more important. A higher weight is given to channels with higher gradients because the higher gradients means that the solution is more sensitive to this channel and so the weight of each channel will be the average gradient magnitude (pooled gradients) for the channel.

The purpose of this visualization is to see which part of the image contributed to the classification and where is the object.

Showing example of visualizing the activation heatmap to see which part of the image contributed to the classification