HW4 - Tuan Tran

1,

$$\begin{array}{ccc} R & B & G \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 \end{bmatrix} & \begin{bmatrix} 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 \end{bmatrix} \end{array}$$

Filter: $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ → expand to 3dim

→ Convolution:

$$\begin{bmatrix} 45 & 45 \\ 54 & 54 \end{bmatrix}$$

2, with padding:

$$\begin{array}{ccc} R & B & G \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 \\ 0 & 3 & 3 & 3 & 3 & 0 \\ 0 & 4 & 4 & 4 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

→ Convolution: $\begin{bmatrix} 18 & 27 & 27 & 18 \\ 32 & 45 & 45 & 32 \\ 45 & 54 & 54 & 45 \\ 26 & 39 & 39 & 26 \end{bmatrix}$

3,

with previous R, B, G ~~m~~ matrices padded with 0, ~~we~~ we can think of doing 2-dilated convolution as doing normal convolution but with the filter now 5x5 ~~and~~ with 9 ~~non zero~~ 1s and 0 everywhere else

↳ Filter:

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

↳ Convolution result: $\begin{bmatrix} 24 & 24 \\ 20 & 20 \end{bmatrix}$

7,

~~We have~~ The width / height of resulting tensor (assume stride 1)

$= \frac{128-3}{1} + 1 = 126.$

↳ Size of resulting tensor: $126 \times 126 \times 16$.

8,

width / height of resulting tensor $= \frac{128-3}{2} + 1 = 63.5$

↳ Can't use stride 2.

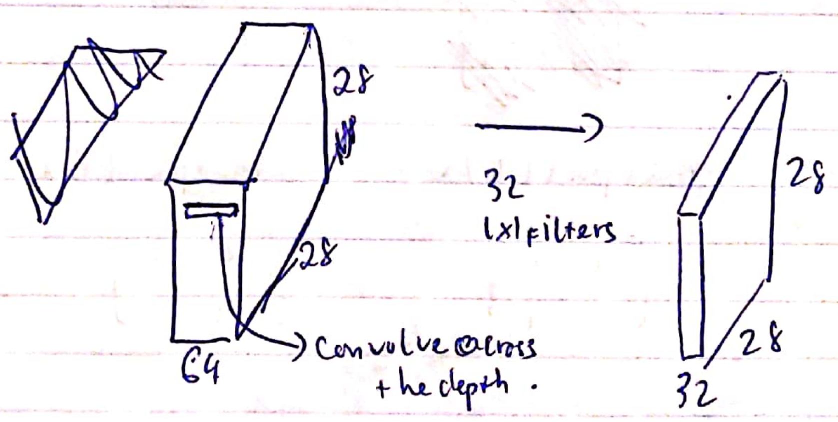
9,

By doing convolution using a 1x1 filter, we are preserving the same width and height of the input, without the need for any padding since the 1x1 filter convolve across the depth

↳ width and height ~~are intact to~~ will be the same.

If we have number of ~~of~~ filters < depth of input, then we essentially reduce the number of channels ~~X decrease di~~ since ~~de~~ height and width do not change.

For ex:



10,

Convolution layers can be thought of as feature extractors as they ~~It has~~ have multiple filters whose weights are learned as we ~~min~~ try to minimize the difference between output and true label

The difference between the early and deeper ~~to~~ conv layers is that the early layers can only pick up low level features like blobs of colors or edges, whereas the deeper layers start to pick up mid to high level features such as eyes, ~~full Circles~~ ~~&~~ nose, etc. This is because ~~the~~ for the ~~late~~ deeper layers, the filters do dot product with the ~~input~~ output of previous Conv layers

↳ We can think of this as taking ~~small~~ edges and blobs of colors and make larger pieces out of them

11,

Applying max pool, we have:

maxpool(R) = $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ maxpool(B) $\begin{bmatrix} 2 & 2 \\ 4 & 4 \end{bmatrix}$ maxpool(G) $\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$

↳ maxpool on image I ~~thus~~ results in size: $2 \times 2 \times 3$.

12,

The purpose of pooling is to ~~redu~~ downsample the ~~input~~ representation ~~inter~~ spatially (width and height). So that we can reduce the number of parameters needed to learn while still retaining "important" features. One nice thing about pooling is that it down samples without introducing new parameters.

In other words, the poolinglayers have no parameters.

13,

Data augmentation ~~to take~~ is the technique of creating different versions of the same image by rotation, scale, etc...

It is most useful when we have a powerful model but too few data → we can add additional data to our training set by creating different versions of ~~the~~ an image

↳ This helps prevent overfitting as we've enriched our training set with valid additional data.

14,

~~Transfer learning is when we have a task that some other model~~

Transfer learning is when we repurpose a pre trained model to perform our ~~related~~ different but related task. This is helpful when our model does not have sufficient data or computation resource to train on its own, thus we can reuse a pretrained model trained for a ~~dif~~ related task. This helps save a lot of time and hassle, speed up training while allowing for high performance.

15.

We need to freeze the coefficients of the convolution base because ~~ae~~ those layers are already trained and ~~very ca~~ already very capable of extracting features. If we don't freeze, we will ~~train t~~ train these layers again and ~~not t~~ it will just be normal training and we destroy the trained weights

↳ The benefit of transfer learning is not leveraged.

16.

~~The coef~~ After training our custom fully connected layers, we can unfreeze some top layers of the convolution base and jointly train the ~~#~~ custom fully connected layers and the unfrozen layers.

↳ We are fine tuning the weights of the pretrained model so that it can fit our dataset better and able to extract more relevant features

17.

The idea for ~~[illegible]~~ inception block is to deploy multiple convolutions with multiple filters and pooling layers ~~sim~~ simultaneously in parallel within the same layer.

↳ Intend to let the model learn the best weights ~~whe clus~~ when training and automatically select the more useful features.

~~when use~~ Inception block also helps reduce number of dimensions ~~[illegible]~~ ~~or~~ ~~if~~ when using 1x1 convolutions

18.

The advantages of residual blocks:

~~Skipping connection helps with vanishing gradients~~

- Zero weights in the block produce identity instead of destroying the signal like normal layers

→ The network learn to zero weights to eliminate unneeded layers

→ Information can pass through units with zero weights

- Gradients are passed directly through skip connections
  → quicker training and help with vanishing gradients

19.

First we need to perform necessary preprocessing for input image ~~by~~ (ex: normalize). We then create a new model from existing one with ~~the intermediate activations out~~ new outputs at ~~&~~ intermediate activations. We then push the input image and visualize ~~the activation~~ by examining the returned ~~active~~ intermediate activations for that particular input

The purpose of doing this is to examine what different layers of convolutional net ~~is~~ are trying to learn. For example, we can look at ~~channel #4~~ the first layer activation maps and look at its 4th channel, we might see that this channel of this particular feature maps ~~are~~ is ~~trying to lea~~ picking up diagonal edges

↳ ~~From this~~ This ~~also~~ helps us explain what the network is doing

20,

- The way we can visualize the filter weight is that given a trained ~~layer~~ network, and ~~look~~ looking at ~~sp~~ a specific layer ~~l~~ l, we can push some random input (ex: white noise) and we can update the initially random input until the response at layer l is maximized (so gradient ascend)
  - ↳ At the end iteration, the initially random image will become the visualization for the filter weights at layer l.

To maximize, we use ~~the~~ the ~~avery~~ average response as the measure.

- The purpose of this is to examine how the filters are being matched to find patterns / texture in images and what features ~~to~~ the filters are trying to match
  - ↳ ~~A~~ again, it helps with explaining what the trained network is doing and we can consider a layer to be finding a decomposition of input into a weighted sum of the filters

21,

- ~~The purp~~ The purpose of this visualization is that we can determine which parts of the image ~~for~~ a particular layer is activated the most. In other words, we can determine which parts of the image contribute to classification. ~~Heat Activ~~ This activation also helps with detecting where the object is in the image. For example, ~~in~~ the model may decide that the image ~~it~~ contains an elephant based on its trunk, ~~where when~~. When we visualize the heatmap, the trunk will have the highest ~~"heat"~~ / largest "heat".

- We can follow the steps to visualize heatmap:

+, Feed ~~into~~ input img into network

+, Compute gradients of a selected output with respect to each ~~the~~ depth / ~~ta~~ channel of target layer where ~~each~~ activations would be computed. In other words, we compute the gradients of the ~~prob~~ output prob of desired class with respect to the desired layers output, for ex: conv2_2

+, Compute the average gradient of each channel → this is ~~to~~ the pooled gradient → we can use this pooled gradient ~~as~~ to weight channels by multiplying the pooled gradient with ~~a~~ the corresponding channel output value

+, We then add the activations of all the weighted outputs. → this produces our heatmap.

+, For best visualization, we superimpose the heatmap onto the original input image.

4,

We can Consider a filter as a template. This is because when we slide ~~a~~ ~~nn~~ n x n x 3 filter through an image, we are doing dotproduct between the filter and that ~~reo~~ n x n x 3 region of the image

↳ ~~If the~~ If the region ~~elements~~ is similar to / correlates with the ~~image~~ ~~image~~, filter, then the response, or dotproduct value, will be high.

↳ The dot product measures how well the filter matches the regions in the image.

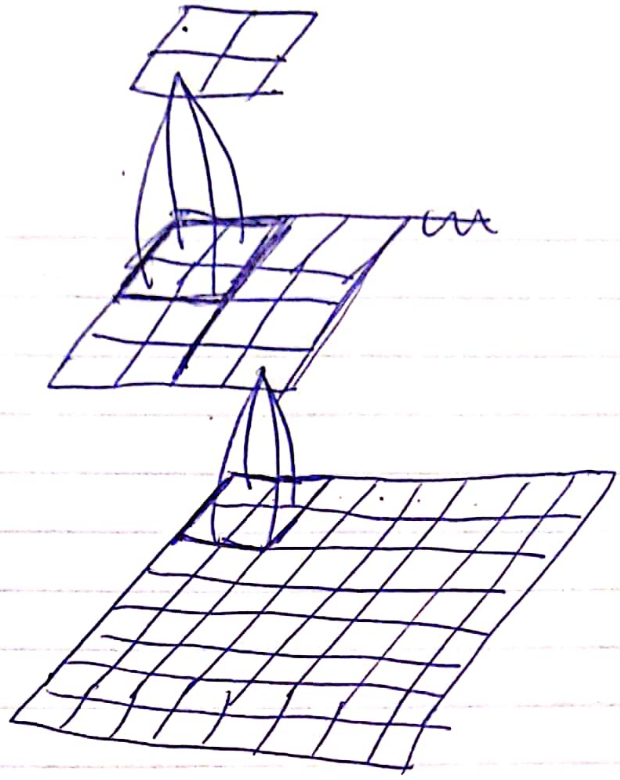
→ Training the weights in the filters is equivalent to finding templates that try to match regions of image.

~~(2)~~ 5,

- Using a fixed size window, we can take the region of image of the window size and combine them into 1 value, we keep doing that for other non overlapping region until we get the ~~the~~ new ~~image (array)~~ ~~array~~. array.

- For example: if our original input is 8x8, using a fixsized window 2x2, we ~~tt~~ find 16 non ~~x~~-overlapping 2x2 regions and we ~~can~~ combine each of them to arrive at the new 4x4 array.
  → Do that again, ~~until can~~.

~~Assuming~~ window size ~~nxn,~~ ~~then~~



↳ Using this ~~tool~~, we can ~~that~~ analyse and look for particular objects with different size.

~~In the~~

6,

We can increase the depth ~~for~~ to compensate for the decrease in spatial dimensions.

The purpose for doing so is that even though spatial dimensions decrease, we can still keep the same number of parameters