CSC421: Programming Assignment 2

Due on Thrusday, Feb 28, 2019

Zhongtian Ouyang
1002341012

Part 1

Colourization as Regression

1.
The RegressionCNN has 6 convolution layers. For filter sizes, the number in the bracket is the depth of each filter. I put them in brackets because I am not sure whether we should count this in this case. Also, the answers are for the current training settings. If we change kernel to 5, kernel size would be 5 x 5. If we change num_filters to 50, 32 will becom 50, while 64 will become 100.

Layer	Filter/Kernel Size	Number of filters
downconv1	3 x 3 (x 1)	32
downconv2	3 x 3 (x 32)	64
rfconv	3 x 3 (x 64)	64
upconv1	3 x 3 (x 64)	32
upconv2	3 x 3 (x 32)	3
finalconv	3 x 3 (x 3)	3

2.
In the given setting, we are training the CNN model for 25 epochs.

3.
I have tried training for 10, 15, 25 and 50 epochs. While the training loss is decreasing as we increase the number of epochs, the output image doesn't have much difference. The output images are not coloured well. They look very dull and are very similar in color.

4.
The color we observe are not divided in three channels like colors in RGB. For example, suppose the target is (120, 50, 240), (120, 0, 240) and (120, 100, 240) will same the same loss, while for human, they look very different. Using the least square as a loss, if the network always output an average color, such as gray, it would have a relative small loss to the different colors of images in the dataset. That is why the output image looks dull.

5.
Framing colourization problem as a classification problem force the neural network to choose an color that it believes to be the most accurate instead of the mediocre, average guess.

Part 2

Colourization as Classification

NOTE: The answers for this question and all following questions are based on the implementation where the number of output channels for upconv2 is num_colours(24).

1.
In colourization.ipynb

2.
Compare the the previous regression model, the colour are overall much more accurate. Even though there are pixels with wrong colour or are still gray, many of the objects have similar color compare to the targets. Some of the wrong-coloured objects from validation images include a gray sky instead of a blue sky, and a white horse instead of a yellow one.

Part 3

Skip Connections

1.
In colourization.ipynb

2.
The result improved by a moderate amount compare to the previous model. Both the validation loss and accuracy are improved. The validation loss decreased from 1.59 to 1.36 and the accuracy increased from 41% to 48%. The output image is also better. Some of the pixels that were gray previously are now filled with correct color.
The reasons may be that:
a. Some information was lost during the pooling in the previous layers, adding a skip connection provide some of those lost information to later layers.
b. Passing the original image to the last layer helps it to fill in the colours while maintaining the original intensity and the image content.

3.
The batch size I have tested are 10, 50, 100, 250, 1000. I trained the U-net for 25 epochs using these batch sizes.
With batch size being 10, the overall outcome is slightly better, with lower validation and training loss and higher accuracy. However, it takes a significantly longer time to train. Also, the training curve is less stable for the validation set.
Using 50 as batch size generate similar result to the one using original 100 batch size.
Using a batch size of 250 or 1000 will have worse outcomes after 25 epochs of training because they don't update as much. Both the validation and training loss increase and the accuracy drop significantly. And the time used for training is only slightly shorter.

Part 4

Super-Resolution

1.
The resolution of the output image is $\frac{1}{16}$ of the input image because each average pool layer with kernel size = 2 reduce the resolution of image to $\frac{1}{4}$ of the original.

2.
The output from both models are almost identical while UNet has a little bit lower loss. The bilinear interpolation results looks like a blurred version of target image. Compare to the bilinear interpolation results, the neural network result has much more distinct boundarys between objects and more details.
Conv nets are better because:
a.With more parameters and nonlinearlity, a neural net is capable of capturing more information and regularities in reconstructing the picture. While bilinear interpolation is only a weighted average of two weighted averages
b.Conv nets can learn and accomodate to some specific characterestics of the target. In our example, for example, are all images with horses. And capturing such information in the parameters can help the neural network make better guesses. As a comparison, bilinear interpolation apply the same set of rules to any kind of images; so it is a much more general algorithm with some general assumptions.

Part 5

Visualizing Intermediate Activations

1.
The activations in the first layer of CNN seems to resemble the input picture a lot. It seems to be extracting different information such as gradient of intensity in the input picture, recognizing boundary of objects, etc. Many details from the original input image is still visible. In the later layers, the activation is more obscure. While some still roughly maintain the shape of input, others are not really recognizable.

2.
The first three layers should be the same as CNN's. In the later layers, especially the last layer, for those activations that resemble the shape of the input picture, more details can be observed. This should be a result of passing in the image directly into last layer.

3.
Different from the previous activations for colorization, the activations for super-resolution are mostly blurry and obscure. The reason could be that the input image itself is with low resolution. The activations for the later layers ressemble the target image a little bit more.

Part 6

Conceptional Problems

1.
a) learning rate
b) kernel size
c) number of filters in each layer
d) number of layers
e) activation functions

2.
The output won't change because maxpooling is outputting the max value among a group of values. And since ReLU function is an non-decreasing function, the value get chosen is the same. ReLU(max(a,b,c,d)) will be the same as max(ReLU(a), ReLU(b), ReLU(c), ReLU(d))
3.
According to the paper, a way to improve the evaluation to match with human assessment better is to use a pretrained image classification neural network as a fixed loss network. So if the pretrained neural network can well handle the output of the currently training neural network, we have a small loss. Vice Versa.
4.
In our case, we don't need to modify our trained model at all for test images that are larger than 32 x 32. The reason is that all the layers in our neural network are convolution layers. For those layers, the size of input don't matter since they are more like feature extractors.