CSC321: Programming Assignment #2

Xiangyu Kong
kongxi16

February 28, 2018

Problem 1

1. The model RegressionCNN has 6 convolution layers. The following table describes each layer's filter size and number of filters in it. Note that we are using the default arguments: kernel = 3 and num_filters = 32

Convolution layer	filter size	number of filters
downconv1	3	32
downconv2	3	64
rfconv	3	64
upconv1	3	32
upconv2	3	3
finalconv	3	3

2. As shown in Fig 1, the predicted results have different colors than the actual result. Also, the predictions are very monotonic. The color selections are nearly the same for every prediction. For example, the green color of the image in the third column is not correctly predicted, and the colors of the image in the fifth column are not correctly predicted.

Figure 1: Regression output

3. RGB color space with squared error could be problematic because if we use different combinations of color intensities, the squared error could still be the same. This will make the network think it already has a confident result that minimizes the loss, while the colorization is actually incorrect.

4. By treating a colorization model as a classification problem, we allow different combination of coloring, and by picking the argmax, we obtain the most likely prediction.

Problem 2

1. The implementation for CNN can be found in colorization.py.

2. The training result has Validation Loss: 1.5881, Validation Accuracy: 41.1% The resultant image is shown in Fig 2. Although the predictions are not entirely correct, the results are better than the results produced by RegressionCNN. The background colors are predicted to be more colorful than RegressionCNN's background color and different images have different color selections.

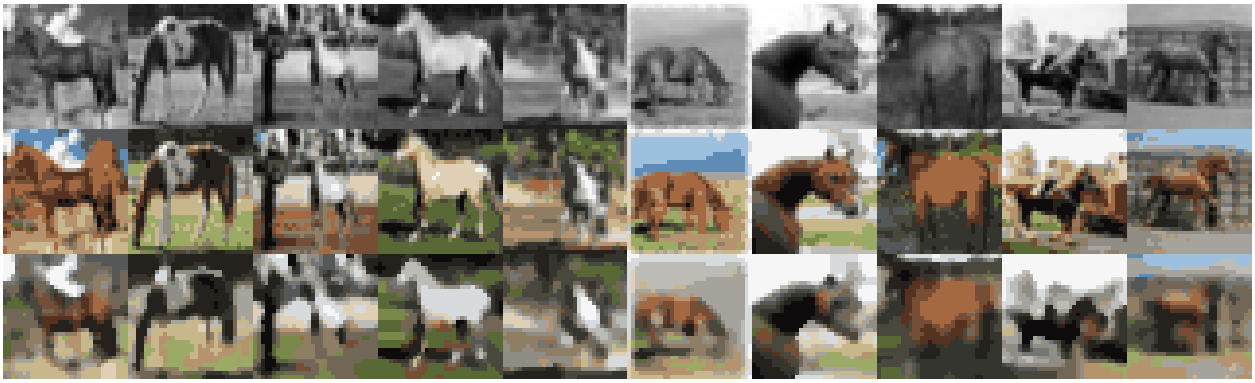

Figure 2: CNN predictions

Problem 3

1. The implementation for UNet can be found in colorization.py.

2. Training the model for 25 epochs with batch size of 10, the final Validation Loss: 1.3174, Validation Accuracy: 51.0%

3. Training the model for 25 epochs leads to an improvement of the validation performance: the loss has reduced by 0.2706 and the accuracy has increased by 9.9%. Also, the overall quality of the image (Fig 3) has increased and the predictions are more like the actual image.

Figure 3: UNet predictions with 25 epochs

Problem 4

1. (a) 3×3 convolution:
   filter size: 3×3, filter number: 32
   Weights $= 3 \times 3 \times 32 = 288$
   Receptive field $= 3 \times 3 = 9$

   (b) 5×5 convolution:
   filter size: 5×5, filter number: 32
   Weights $= 800$
   Receptive field $= 5 \times 5 = 25$

   (c) 3×3 convolution with dilation 1:
   filter size: 3×3, filter number: 32
   Weights $= 3 \times 3 \times 32 = 288$
   Receptive field $= 7 \times 7 = 49$

2. This is because the first few layers focus on the small details and parts of the whole image. Adding diluted layers there contradicts to this purpose because it increases the receptive area.

   Also, the middle layer does not contain any rescaling and contains the minimum dimension. This makes the diluted convolution layer cover more part of the image (i.e. the reception field is large). Since diluted convolution will increase the reception field, the effect will be even more significant.

Problem 5

1. The first few layers focus on the specific part of the horse, for example, their eyes and mouth. For the later layers, the activations focus on the bigger picture, like the whole horse and the background.

2. The activations for UNet is different than the activations for CNN. For activations in UNet, the first few layers still focus on the horse and its details. However, the last few layers not only focus on the bigger picture, but also focus on the details. This is because of the skip-connection connects the first few layers' output to the last few layers.

Problem 6

1. Flipping each image upside down will not be helpful because flipping each horse upside down does not make sense and will make the network give incorrect results.

   Flipping each image left to right will be helpful. This is because the direction the horse is facing should not affect the colorization.

   Shifting each image one pixel (left / right and up / down) will help our CNN model because the position of the horse should not matter when we are colorizing.

   Augmenting via using other of CIFAR-10 classes would not be helpful since our current model focuses on colorizing HORSE only, so introducing other classes is not very helpful. However, other classes' background could be helpful for our network to learn to color the horse class' background.

2. The hyperparameters could have been tuned are:

   (a) the learning rate α

   (b) the kernel size $kernel$

   (c) the number of color classes num_color

   (d) the activation function

   (e) the loss function