CSC321: Assignment 2

Due on February 28, 2018

Wenxin Chen - 1002157676

March 1, 2018

1 Neural Network

1.1 Description of RegressionCNN

RegressionCNN has six convolutional layers. Each of these layers except for the last has a ReLU activation function. Additionally, after every convolution the results are normalized via BatchNorm2d before being sent to the activation function. The sizes of the output filters vary across the layers, depending on the usage of MaxPool2d and Upsample. The sizes of the filters do not change during the convolutions themselves due to the padding value used. The number of filters in each layer as well as their sizes are listed in the table below:

Table 1: Number of Filters and Filter Sizes

Layer	Name	Input Filters	Input Size	Output Filters	Output Size
1	downconv1	1	32×32	num_filters (32)	16×16
2	downconv2	num_filters (32)	16×16	num_filters*2 (64)	8×8
3	rfconv	num_filters*2 (64)	8×8	num_filters*2 (64)	8×8
4	upconv1	num_filters*2 (64)	8×8	num_filters (32)	16×16
5	upconv2	num_filters (32)	16×16	3	32×32
6	finalconv	3	32×32	3	32×32

1.2 Running colour_regression.py

The following results were observed after running colour_regression.py:

Figure 1: Output of Colour Regression

The pictures are ordered by input (top), reference (middle), and prediction (bottom). We see that the prediction generally appears more washed-out than the reference photos. For example, the grass and sky only appear slightly green and blue in the predicted photos, whereas in the reference photos they are very vivid in colour. Generally, the results do not look that great as the colours do not appear to have moved too far away from being completely black and white.

1.3 RGB Colour Space

RGB colour space has the issue of different channels being correlated based on image factors such as brightness and contrast. Two images with the same colours present but one brighter and one darker will appear drastically different from an RGB perspective. It is likely that with a large dataset of images of varying brightness and contrast, even with normalization, the weights will

experience a sufficient level of variance in the input (purely due to the RGB model's response to brightness and contrast) such that the final image exhibits an "averaged-out" result to a degree. This could explain the washed-out nature of the output previously observed.

1.4 Classification vs Regression

It is likely that the level of colour accuracy possible (continuity) provided by a regression model is not necessary for the majority of applications, and a discrete number of colours used in classification is sufficient to "colour" an image in most applications. After all, minimizing the squared RGB error is just attempting to match a pixel to a given colour (in this case, one of $256^3 \approx 17mil$). Clearly the human eye is not capable of distinguishing between 17 million colours and it should be sufficient to match a nearby colour in the majority of cases. Thus, a classification method can be used.

2 Colourization as Classification

2.1 Complete CNN Model

Instead of three output channels, the final convolution will receive 24 input channels and will output 24 channels as well (corresponding to each of the 24 selected colours). MyConv2d was used instead of nn.Conv2d.

2.2 Colourization Results

The following results were observed from the classification model:

Figure 2: Output of Colour Classification

The colourization appears less washed out than the regression model in general, however, there are still areas (especially of the sky) where colours are missing. It is interesting to note that since our dataset contains only horses, a colour pool of 24 colours performs artificially well as the same colours show up in general across all the pictures (blue, green, brown). Note that there are very little colours outside of these three that we can observe. It is likely that, given photos with a greater variety of colour, a pool of just 24 colours would not be sufficient in spanning the entire colour space. However, given this application the classification model works better than the regression model in general and produces more vivid and accurate images.

3 Skip Connections

3.1 Completing UNet Model

The UNet model is similar to the CNN model. However, the inputs to upconv1, upconv2, and finalconv include the outputs from the first, second, and third layers respectively. This is achieved through the use of torch.cat on dimension 1, which is the dimension that refers to the collection of filters for a given training example. E.g. if dimension 1 is 64, it means that there are 64 filters of size nxn for that training example (which itself is indexed by dimension 0). Since we are increasing the number of filters that upconv1, upconv2, and finalconv receive as inputs, the appropriate parameters to MyConv2d (namely, in_channels) have been updated to reflect this.

3.2 Training the UNet Model

The following graph displays the UNet training curve over 5 Epochs and 25 Epochs:

Figure 3: UNet Training Curve over 5 Epochs

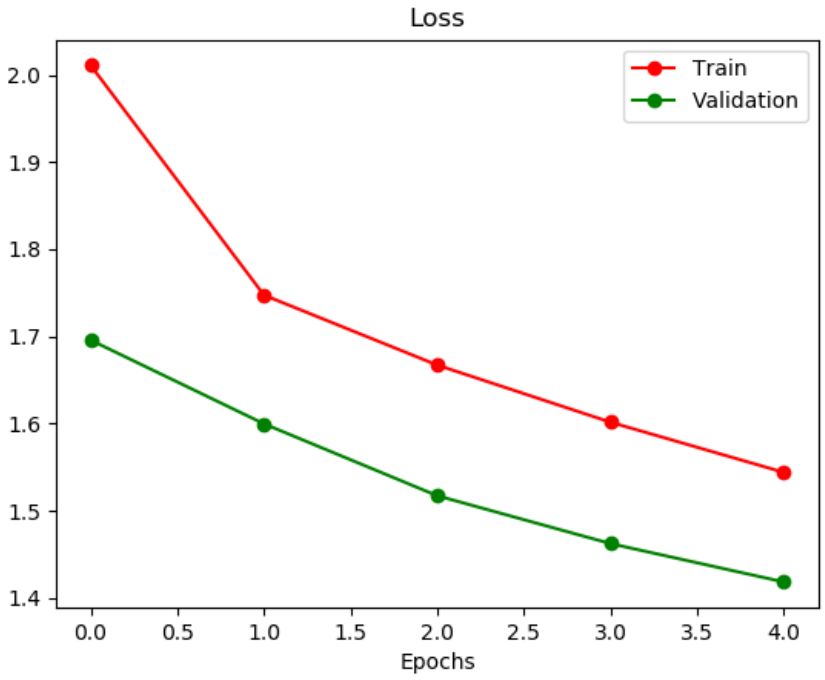
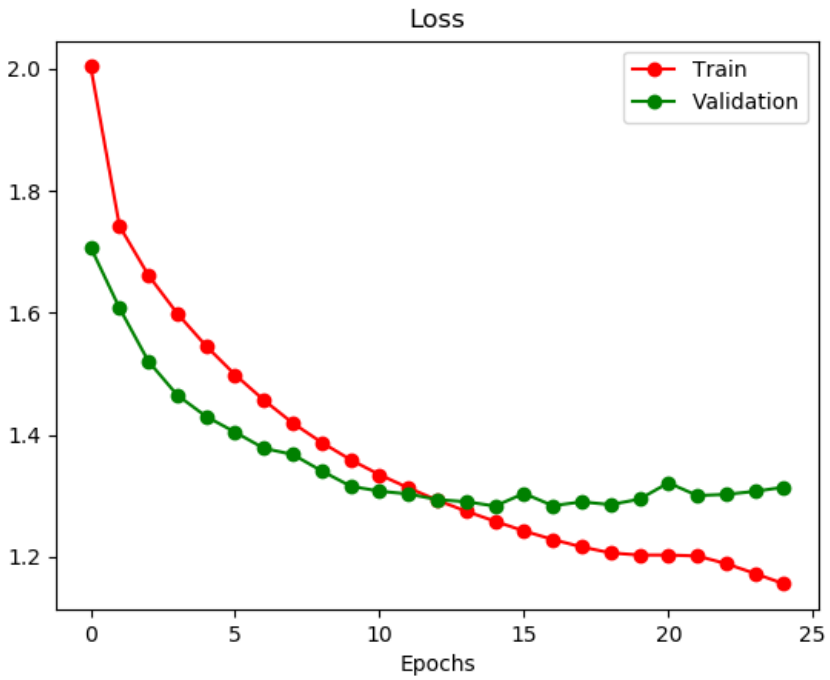


Figure 4: UNet Training Curve over 25 Epochs



3.3 Model Evaluation

The validation loss and accuracy from the CNN without skip connections were as follows:

```
Val Loss: 1.5881, Val Acc: 41.1%
```

Using skip connections, after 5 epochs the validation loss and error were as follows:

```
Val Loss: 1.4026, Val Acc: 48.0%
```

And after 25 epochs (using skip connections):

```
Val Loss: 1.3112, Val Acc: 51.3%
```

Quantitatively, skip connections result in a lower validation loss and greater validation accuracy compared to a regular CNN.

The output images from the skip connections also appear to be more accurate in colouring qualitatively (next page):

Figure 5: UNet Output after 5 Epochs

Figure 6: UNet Output after 25 Epochs

Two potential reasons for the increased accuracy due to skip connections are as follows:

1. Since we are using ReLU as an activation function, which has a limited gradient range (either 0 or 1) the network may experience vanishing gradients during training [1]. The use of skip layers means that the gradient is less likely to vanish in the initial layers, allowing for more useful weights to result from the training, increasing performance.
2. Skip connections reduce the chance of over fitting as it inherently introduces additional noise into the neural network by feeding the later layers with the noisy initial input. Thus, even if a training set of images consistently excites (i.e. causes increases in updates to) the same units and weights in the inner layers, the fact that the images are not identical will help to ensure that the network does not overfit to those excited units and weights.

4 Dilated Convolution

4.1 Number of Weights vs Receptive Field

1. 3x3 convolutions : 9 Weights, 3x3 (9) Receptive Field
2. 5x5 convolution : 25 Weights, 5x5 (25) Receptive Field
3. 3x3 convolution with dilation 1 : 9 Weights, 5x5 (25) Receptive Field

4.2 Number of Weights vs Receptive Field

Adding the convolution in the middle layer makes sense as layers closer to the beginning and end (and especially those at the beginning and end) require local details (i.e. edges, shapes, shadows) to be detected in training and output during testing. Thus a dilation at one of those layers would cause local detail to be lost as there are gaps in the receptive field, possibly compromising performance.

5 Visualizing Intermediate Activations

5.1 CNN Activations

For the CNN activations, the activations in the beginning layers appear to contain more spatial data about the image (edges are more or less preserved in most of the filters). In at least half of the filters, the horse can still be identified (albeit with some squinting). The later layers, however, contain much less spatial data and in the majority of cases the original object in the image (horse) cannot be identified. In the layer before the output some clusters can be identified that relate to clusters of the same colour in the image. For example, a white patch in the output image may be seen as a white patch surrounded by black in one of the activations.

5.2 UNet Activations

The same behaviour is observed in the UNet activations, but a bit more spatial data seems to be preserved in the later layers. However, the pattern of more spatial data in the first layers and apparent clustering based on colour in the later layers remains.

6 Conceptional Problems

6.1 Data Augmentation

1. Augmenting via flipping each image upside down
   This is not likely to help the performance of the model as upside down images, in most cases, cannot be generalized to help the model predict the colouring of right-side-up images

(which will form the entirety of the input set). Feeding upside down images is likely to improve the algorithm's performance on upside down images but will likely decrease or have no effect on its performance on right-side-up images.

2. Augmenting via flipping each image left to right
   This is likely to help the model as images can be flipped horizontally to result in another valid image, features of which are likely to appear in other input images.
3. Augmenting via shifting each image one pixel left / right
   This may help the model slightly, as a shifted image will provide the model with different inputs at each weight. However, since the images differ by only one pixel and convolutional networks are good at recognizing spatial patterns (despite shifts), the benefits may be minimal.
4. Augmenting via shifting each image one pixel up/down
   Same as previous, likely would not hurt the model but may not improve it much either.
5. Augmenting via using other of the CIFAR-10 classes
   This is hard to predict- the additional classes may help in colourizing common features such as the sky, grass, and trees, but may confuse the model when colouring specific features (e.g. the horse). For example it may accidentally colour a horse the same colour as a zebra, if enough of those are in the training set.

6.2 Hyperparameters

Some hyperparameters that could have been tuned are listed below:

1. Learning Rate
2. Number of Hidden Layers
3. Training Epochs
4. Batch Size
5. Number of Colours (K-Means Clusters)

7 Dilation Implementation

7.1 Implementation

The dilation parameter of F.conv2d was used to implement the dilated layer.

7.2 Results

The pre-trained weights resulted in a validation loss of 1.4353 and a validation accuracy of 45.8%, which is less than what was obtained from the non-dilated UNet. It does outperform the regular CNN however. The output of the DUNet is shown on the next page. Qualitatively, they appear similar to the output of UNet.

Figure 7: DUNet output using provided weights

8 Bibliography

[1] Nielsen, M. (2017). Why are deep neural networks hard to train. [online] Neural Networks and Deep Learning. Available at: http://neuralnetworksanddeeplearning.com/chap5.html [Accessed 1 Mar. 2018].