CV ASSIGNMENT 3

21K-3153

COLOR REGRESSION:

Q1)

There are a total of **6 convolution layers** in the RegressionCNN model:

Layer Group	**Convolution Layer(s) Count**
Downconv1	1
Downconv2	1
RFConv	1
Upconv1	1
Upconv2	1
FinalConv	1 (MyConv2d)
Total	**6 convolution layers**

Filter Size and Number of Filters in each layer:

Conv	Desc	Input Channels	Output Channels	Filter Size	Padding	Notes
DownConv1	Conv2D + BN + ReLU + MaxPool2d	1	32	3x3	Yes,1	First downsampling
Downconv2	Conv2D + BN + ReLU + MaxPool2d	32	64	3×3	Yes,1	Second downsampling
RFConv	Conv2D + BN + ReLU	64	64	3×3	Yes,1	Bottleneck
Upconv1	Conv2D + BN + ReLU + Upsample	64	32	3×3	Yes,1	First upsampling
Upconv2	Conv2D + BN + ReLU + Upsample	32	3	3×3	Yes,1	Second upsampling
FinalConv	MyConv2d	3	3	3×3	Yes,1	Final output layer

All layers use 3×3 filters. The model symmetrically doubles the filters in the encoder (1→32→64), and halves them in the decoder (64→32→3).

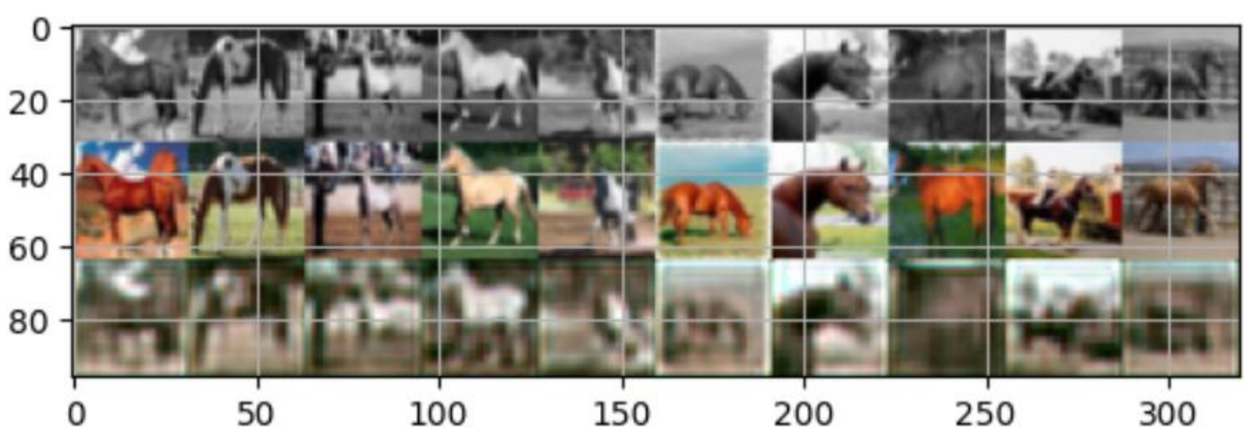
Analysis on Epochs:

10 Epochs:

The training and validation losses decrease steadily over the 10 epochs, indicating consistent learning and generalization.

- **Final Train Loss:** 0.0118
- **Final Validation Loss:** 0.0118

The close match between train and validation losses suggests the model is not overfitting and is performing well on unseen data.



25 Epochs:

The model shows a clear and steady decrease in both training and validation loss over 25 epochs.

- **Final Train Loss:** 0.0092
- **Final Validation Loss:** 0.0088

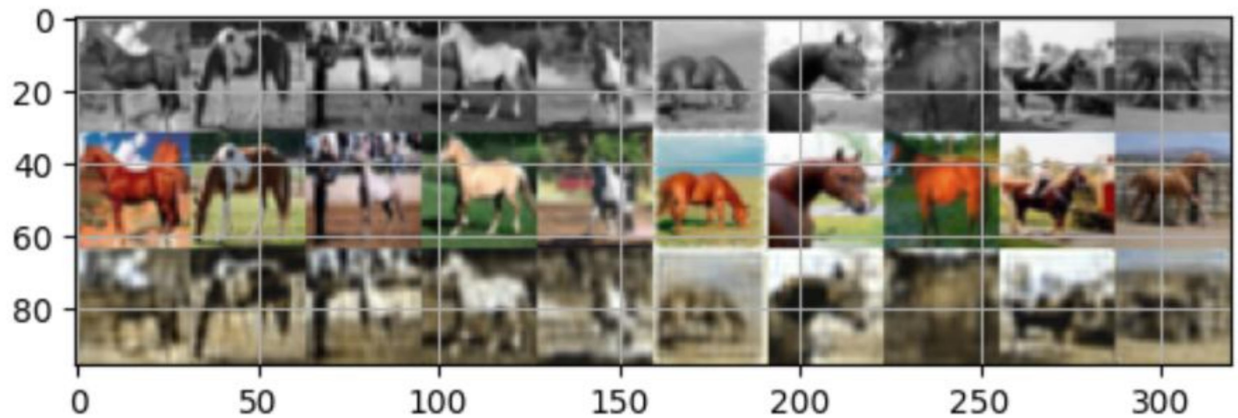
The low and closely matched final losses indicate effective learning and good generalization, with no signs of overfitting.

100 Epochs:

The model demonstrates strong and consistent learning throughout 100 epochs, with a smooth decline in both training and validation loss.

- **Final Train Loss:** 0.0060
- **Final Validation Loss:** 0.0065

The model converges well, and the slight gap between train and validation losses suggests good generalization with no overfitting.



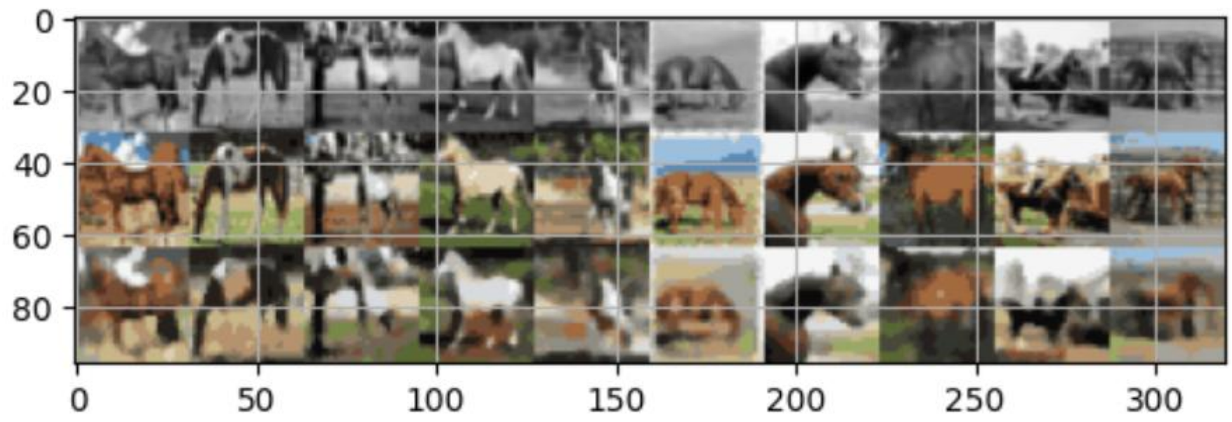
Increasing the number of epochs consistently improved both training and validation loss; however, the trade-off between computation time and visual improvements must be considered. Model performance began to plateau around 15 epochs, indicating diminishing returns beyond that point. Running for 25 epochs struck a solid balance between computational efficiency and output quality.

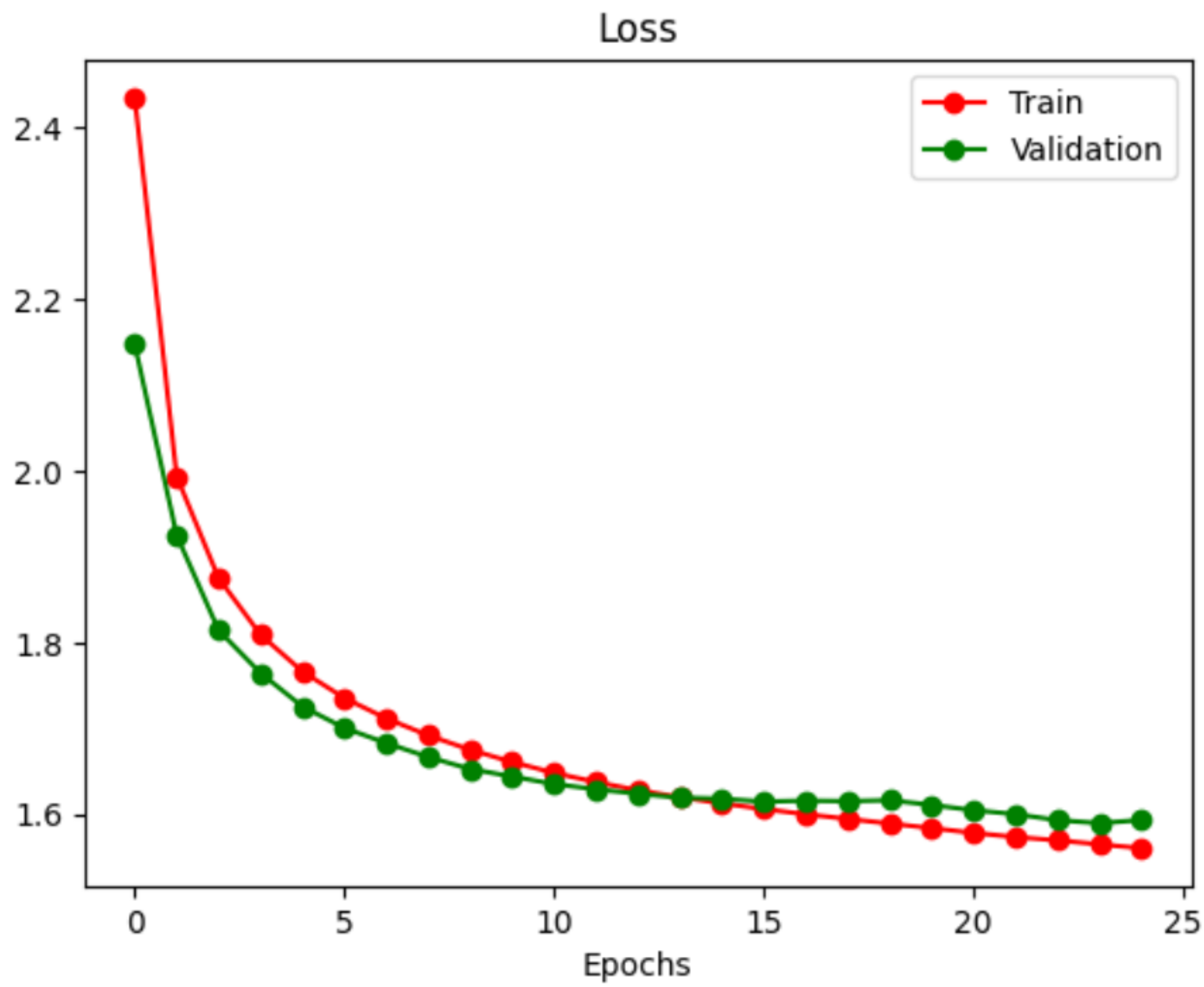
COLORIZATION REPORT:

RESULTS AFTER 25 EPOCHS:

Epoch [25/25], Loss: 1.5604, Time (s): 32

Epoch [25/25], Val Loss: 1.5930, Val Acc: 40.7%, Time(s): 33


0
20
40
60
80
0
50
100
150
200
250
300


Loss
Train
Validation
2.4
2.2
2.0
1.8
1.6
0
5
10
15
20
25
Epochs

Did the classification-based model result in better validation loss and accuracy?

Yes, comparing the losses on 25 epochs of both models shows that the classification based model performs much better than the regression model, showcasing a much higher loss.

Did the skip connections improve the validation loss and accuracy?

S**kip connections often improve validation loss and accuracy**, especially in tasks where fine details are crucial (like colorizing images), as they allow the model to retain low-level features that might otherwise be lost in deeper layers.

Did the skip connections lead to qualitatively better outputs? If so, how?

Skip connections improve output quality by preserving fine details and textures in colorization tasks. They allow the model to combine low-level features from earlier layers with deeper, abstract representations, resulting in more accurate color predictions, especially around edges.

Provide at least two reasons why skip connections may enhance the performance of CNN models.

- **Preventing information loss**: By passing information from earlier layers directly to later layers, skip connections help the network preserve fine-grained spatial features. This is particularly useful in tasks like colorization, where pixel-level accuracy is important.
- **Improving gradient flow**: Skip connections mitigate the vanishing gradient problem in deep networks by allowing gradients to flow more easily through the network during backpropagation. This makes it easier to train deeper models without losing the ability to update weights effectively.

Visualizing Intermediate Activations

In CNNs:

- **Early layers** tend to activate on low-level features like edges, textures, and simple patterns (e.g., vertical or horizontal lines).
- **Deeper layers** tend to activate on more complex, abstract features that represent higher-level patterns or objects (e.g., shapes or faces).

Visualizing Colorization UNet Activations for Test Examples

In a **UNet** architecture, the activations:

- **Early layers** will focus on capturing low-level details like edges and colors, similar to standard CNNs.
- **Later layers** (including the skip connections) will focus on reassembling those features to generate a full-colorized image, using the retained details to preserve texture and structure across the image.

Theoretical Question 1: Pixel-Level Loss vs. Human Perception

Q) The loss functions and evaluation metrics in the supplementary code operate at the pixel level. However, pixel-level measures often don't align with human perception of image quality. How can we improve the evaluation process to better match human judgment?

Problem:

1. MSE and CrossEntropy focus on exact pixel values.
2. These losses don't reflect how humans perceive images. Even small color changes that look fine to us can be seen as large errors by the model.

How to Improve:

1. **Perceptual Loss**: Instead of comparing pixels, use a pretrained network (e.g., VGG16) to compare how the model "sees" the images. This produces more natural-looking results.

2. **SSIM**: Measures structural similarity, including brightness, contrast, and structure, aligning more closely with human visual perception.
3. **LPIPS**: A learned metric designed to reflect human judgment on image similarity, improving alignment with how we perceive image quality.

Theoretical Question 2: Colorizing Larger Images than 32×32

Q) The models were trained on 32×32 images, but test images are often larger. How can the trained models be adapted to colorize larger images?

How It Works:

1. The models are fully convolutional, allowing flexibility with input sizes.
2. There's no need for fixed dimensions, so larger grayscale images (e.g., 64×64 or 128×128) can be processed and colorized to match the output size.
3. The feature maps will adjust at each layer based on the input size.

Notes:

- Ensure the input size is divisible by powers of 2 (due to pooling and upsampling layers).
- This ensures proper scaling and preserves performance when colorizing larger images.