CENG 483

Introduction to Computer Vision

Fall 2023-2024

Take Home Exam 3: Image Colorization

Full Name: Andaç Berkay Seval
Student ID: 2235521

1 Baseline Architecture (30 pts)

For hyperparameter tuning, instead of doing random search, although I know I am not expected to run all possible combinations of hyperparameters, I trained 45 models for all combinations since I had the time and computational resources. Therefore, I had a chance to experiment all combinations of hyperparameters and their effects on the model performance. Before going to discuss the effects of hyperparameter choices, these are the ones for models that I used:

number of convolutional layers = [1, 2, 4]
number of kernels = [2, 4, 8]
learning rates = [0.0001, 0.001, 0.01, 0.1, 10]

Batch-size is fixed and it is 16 for all experiments. Also, kernel size is 3. The algorithm chooses number of epochs automatically by using an early stopping technique with a patience. It basically compares the best 12-margin error with the last 12-margin error for the validation set. If there is no improvement for the 12-margin error in the last patience number of epochs, training stops. If the last 12-margin error is less than the best one, best one becomes the last and the counter reset to 0. Thus, for tuning the number of epochs, 12-margin error is used instead of the MSE loss in the validation set. However, MSE loss for the validation set is also kept track for the comparison purposes.

- Discuss effect of the number of conv layers: In this discussion, number of kernels is 4 and learning rate is 0.1 and both are fixed. For the model with 1 convolutional layer, automatically chosen number of epochs is 32. For this model (1 conv layer, 4 kernels, 0.1 learning rate),

  - training set MSE loss is 0.009159536490710779
  - validation set MSE loss is 0.009309748567640781
  - validation set 12-margin error is 0.24844578

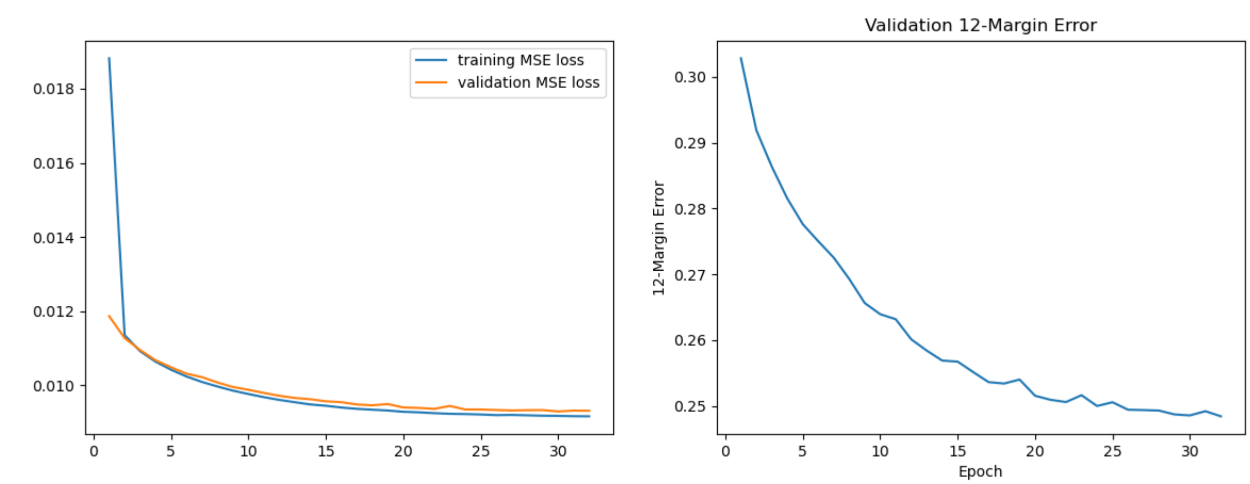
  For the model with 2 convolutional layers, automatically chosen number of epochs is 47. For this model (2 conv layers, 4 kernels, 0.1 learning rate),

  - training set MSE loss is 0.008477284015117648
  - validation set MSE loss is 0.008633652400225402
  - validation set 12-margin error is 0.23414488

For the model with 4 convolutional layers, automatically chosen number of epochs is 10. For this model (4 conv layers, 4 kernels, 0.1 learning rate),

- training set MSE loss is 0.01234520353323307
- validation set MSE loss is 0.010068228907883166
- validation set 12-margin error is 0.26177073

The model with 1 convolutional layer has larger 12-margin error than the model with 2 convolutional layers over validation set. Also, it has larger MSE loss over both training and validation sets. Also, the model with 4 convolutional layers has larger 12-margin error over validation set and MSE loss over training and validation sets than the model with 2 convolutional layers. Therefore, the model with the best 12-margin error over validation set and the best MSE loss over training and validation sets has 2 convolutional layers. These result are gained without using regularization. Theoretically, bigger neural networks have more capacity. Hence, using neural network size as a regularizer is a wrong choice. In that case, using stronger regularization is the right way. Thus, I trained the model with 4 convolutional layers with different regularization strengths for L2 regularization by using "weight_decay" parameter in the "torch.optim" class. Hence, I managed to reduce the 12-margin error over validation set to 0.24017422 and MSE loss over training set to 0.009026813657043841 and MSE loss over validation set to 0.008878542132675648 with the automatically chosen number of epochs of 25 with regularization strength 0.0001. However, these values are still greater than the model with 2 convolutional layers which has 12-margin error of 0.2341. Thus, the model with 2 convolutional layers gives the best result. For the models with convolutional layers 1, 2 and 4, training and validation set MSE loss plots and validation set 12-margin error plots are given together with the qualitative results on the validation set.

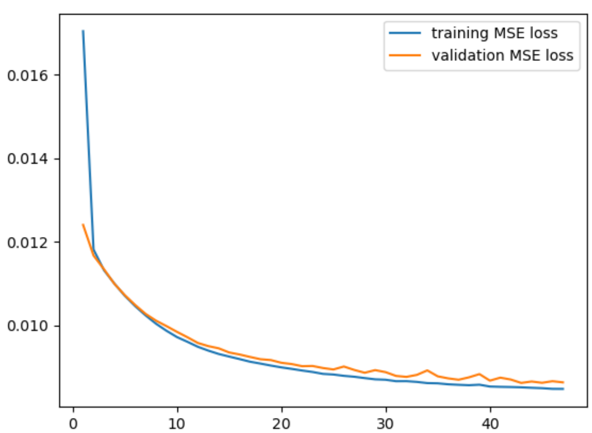


(a) Training and validation set MSE loss plot for 1 conv layer model

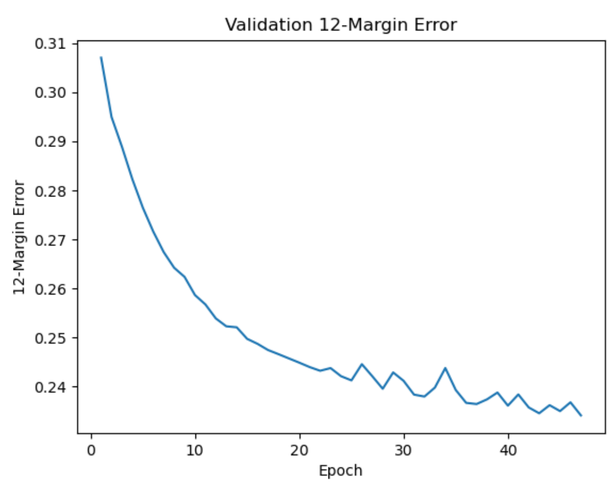
(b) Validation set 12-margin error plot for 1 conv layer model

Figure 1: Plots for 1 conv layer model

Figure 2: Gray scale images, Predictions and Targets for 1 conv layer model



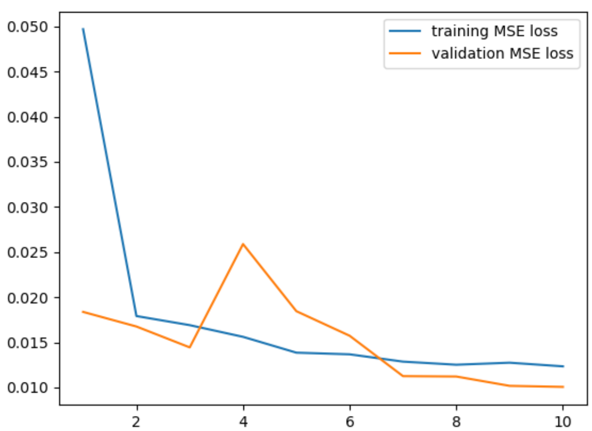
(a) Training and validation set MSE loss plot for 2 conv layer model



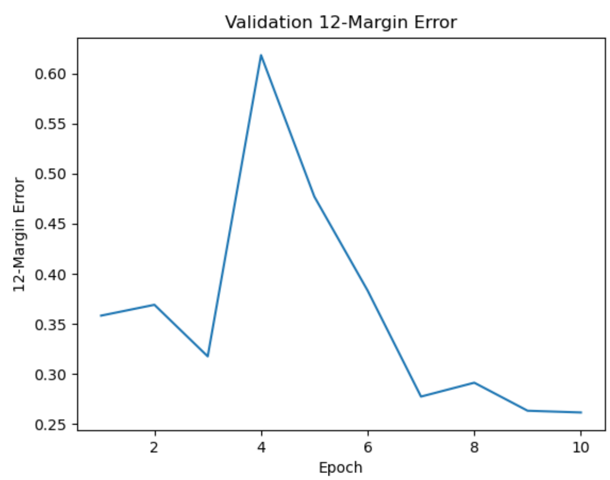
(b) Validation set 12-margin error plot for 2 conv layer model

Figure 3: Plots for 2 conv layer model

Figure 4: Gray scale images, Predictions and Targets for 2 conv layer model



(a) Training and validation set MSE loss plot for 4 conv layer model



(b) Validation set 12-margin error plot for 4 conv layer model

Figure 5: Plots for 4 conv layer model

Figure 6: Gray scale images, Predictions and Targets for 4 conv layer model

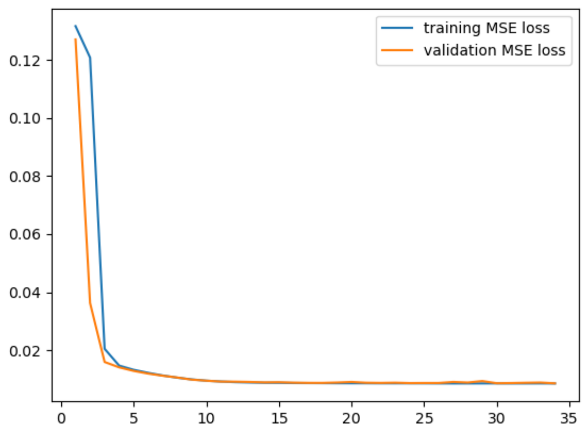
- Discuss effect of the number of kernels(except the last conv layer): In this discussion, number of conv layers is 2 and learning rate is 0.1 and both are fixed. For the model with 2 kernels, automatically chosen number of epochs is 34. For this model (2 conv layers, 2 kernels, 0.1 learning rate),
  - training set MSE loss is 0.00858771997375991
  - validation set MSE loss is 0.008631347257643938
  - validation set 12-margin error is 0.2346004

  For the model with 4 kernels, automatically chosen number of epochs is 47. For this model (2 conv layers, 4 kernels, 0.1 learning rate),
  - training set MSE loss is 0.008477284015117648
  - validation set MSE loss is 0.008633652400225402
  - validation set 12-margin error is 0.23414488

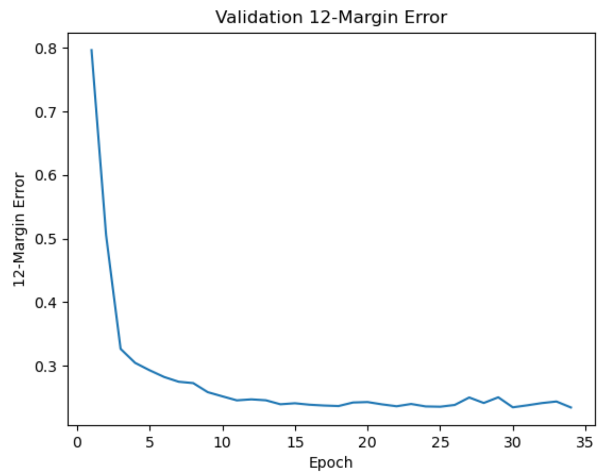
  For the model with 8 kernels, automatically chosen number of epochs is 46. For this model (2 conv layers, 8 kernels, 0.1 learning rate),
  - training set MSE loss is 0.008399437735089289
  - validation set MSE loss is 0.008512241151183843
  - validation set 12-margin error is 0.2320971

  The model with 2 kernels has larger 12-margin error over validation set than the model with 4 kernels and the model with 8 kernels. However, 12-margin error results for the models with 2 kernels and 4 kernels are very close to each other (2 kernels $\Rightarrow$ 0.2346, 4 kernels $\Rightarrow$ 0.2341). The model with 8 kernels has the least 12-margin error over the validation set that is 0.2321. Furthermore, it has the least MSE loss over the training set (0.0084) and the least MSE loss over the validation set

(0.0085). Thus, as the parameter size increases (more neurons), model capacity also increases. Hence, the model performance increases. Therefore, the model with 8 kernels gives the best result. For the models with 2, 4 and 8 kernels, training and validation set MSE loss plots and validation set 12-margin error plots are given together with the qualitative results on the validation set.



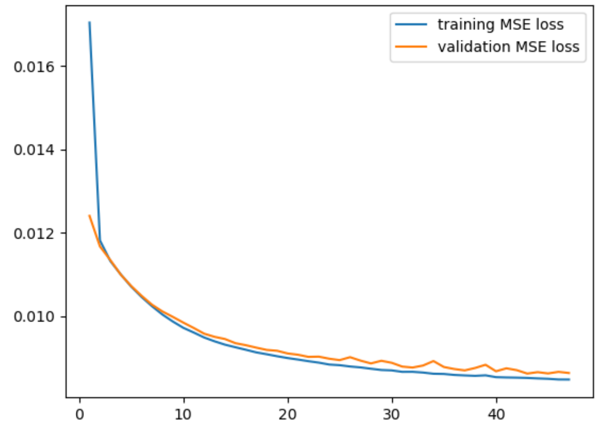
(a) Training and validation set MSE loss plot for kernel number 2 model



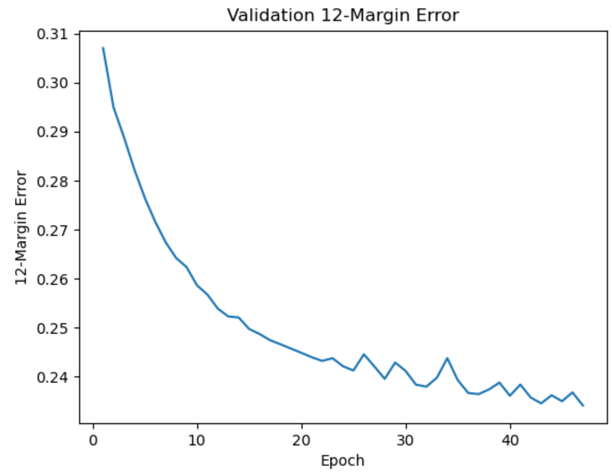
(b) Validation set 12-margin error plot for kernel number 2 model

Figure 7: Plots for kernel number 2 model

Figure 8: Gray scale images, Predictions and Targets for kernel number 2 model



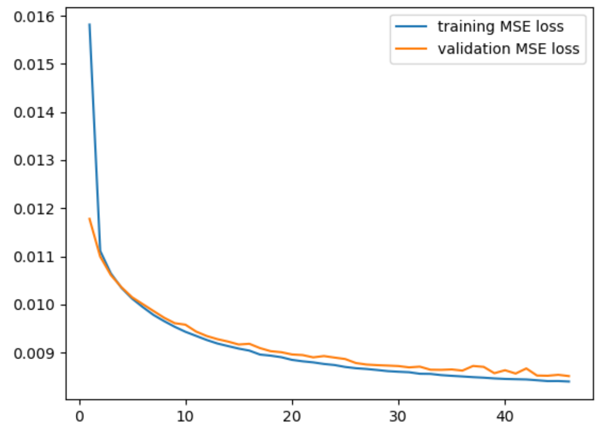
(a) Training and validation set MSE loss plot for kernel number 4 model



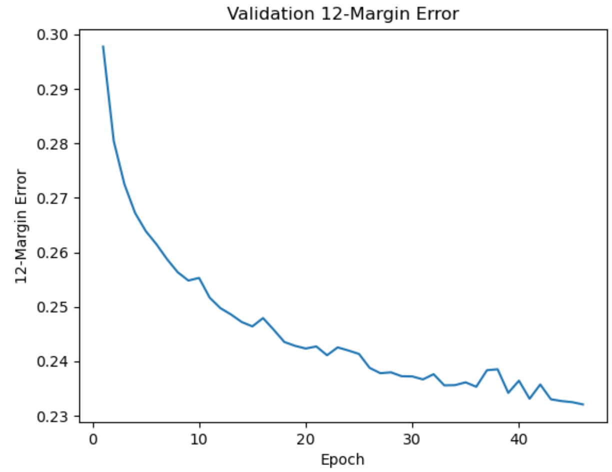
(b) Validation set 12-margin error plot for kernel number 4 model

Figure 9: Plots for kernel number 4 model

Figure 10: Gray scale images, Predictions and Targets for kernel number 4 model



(a) Training and validation set MSE loss plot for kernel number 8 model



(b) Validation set 12-margin error plot for kernel number 8 model

Figure 11: Plots for kernel number 8 model

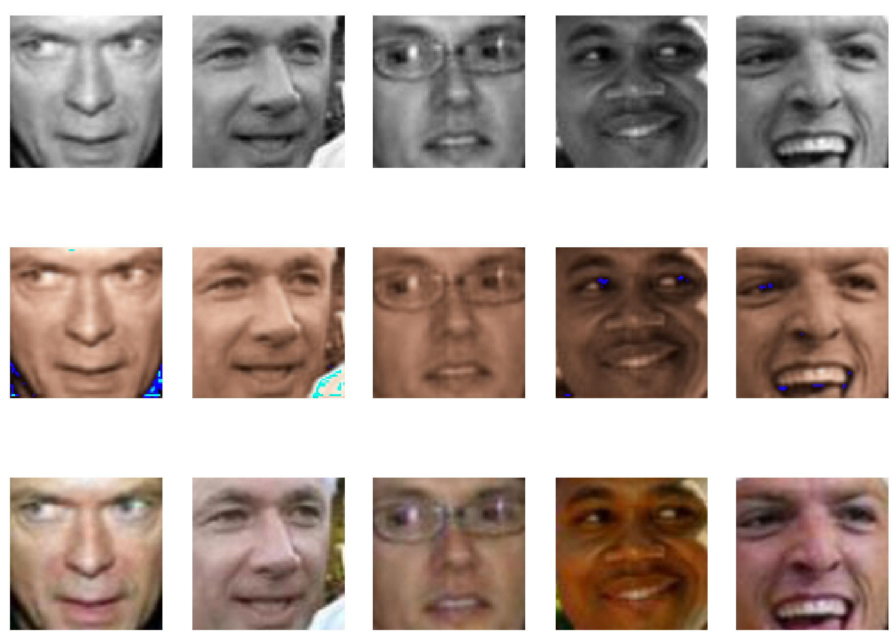


Figure 12: Gray scale images, Predictions and Targets for kernel number 8 model

- Discuss effect of the learning rate by choosing three values: a very large one, a very small one and a value of your choice: In this discussion, number of conv layers is 2 and number of kernels is 8 and both are fixed. The choice of the very large learning rate is 10, very small learning rate is 0.0001 and my choice is 0.1 which gives the best results. Although I give the results for 3 learning rate choices (10, 0.0001, 0.1) in this discussion, I trained models with extra learning rates (0.001, 0.01) too in order to examine model performances. For the model with the learning rate 10, automatically chosen number of epochs is 1. For this model (2 conv layers, 8 kernels, 10 learning rate),

- training set MSE loss is nan
- validation set MSE loss is nan
- validation set 12-margin error is undefinable

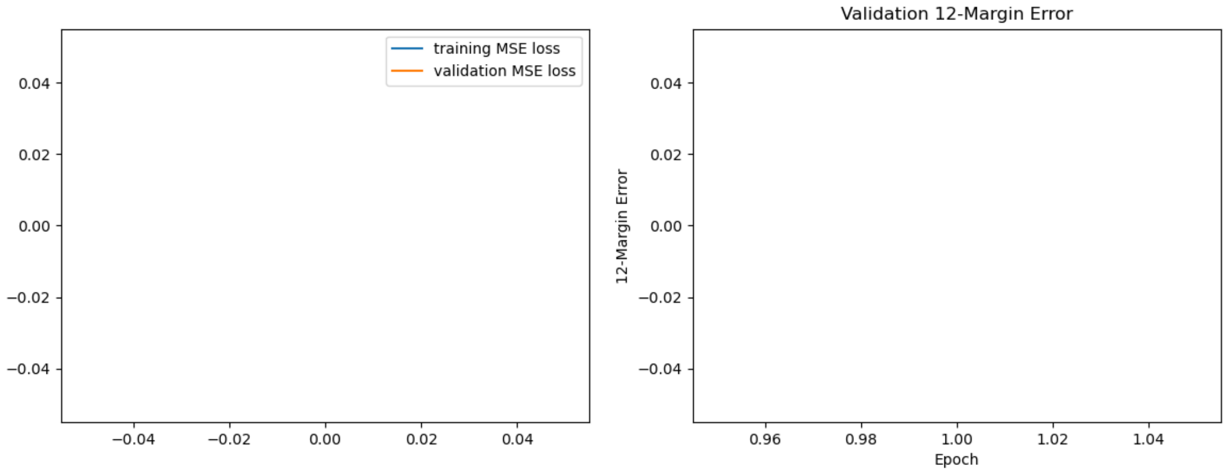
For the model with the learning rate 0.0001, automatically chosen number of epochs is 100. Thus, it did not break out early. For this model (2 conv layers, 8 kernels, 0.0001 learning rate)

- training set MSE loss is 0.018224987745903934
- validation set MSE loss is 0.018297040678560732
- validation set 12-margin error is 0.4082442

For the model with the learning rate 0.1, automatically chosen number of epochs is 46. For this model (2 conv layers, 8 kernels, 0.1 learning rate),

- training set MSE loss is 0.008399437735089289
- validation set MSE loss is 0.008512241151183843
- validation set 12-margin error is 0.2320971

The model with the learning rate 10 runs for 6 epochs. Moreover, in each epoch, MSE loss over both training set and validation set values are nans and it did not change. Also in each epoch, 12-margin error is undefinable since all values of the predictions over validation set are nans. Thus, this means that learning rate is too high. Hence, the plots below are empty for MSE loss over training and validation set and 12-margin error over validation set. Also, predictions over validation set are just black (due to the nan values). The model with the learning rate 0.0001 runs for 100 epochs. It did not stopped early. MSE loss over training and validation set and 12-margin error over validation set are gradually but very slowly decreasing. This means that learning rate is too low. In epoch 100, 12-margin error over validation set is 0.408 and MSE losses over training set and validation set are 0.018 both. Hence, this model needs to be trained over 100 epochs to give comparable results. The model with an average value of learning rate (0.1) gives much better results compared to a model with very large or very small choice of learning rate. The model with the learning rate 0.1 has 12-margin error over validation set of 0.2321, MSE loss over training set of 0.0084 and MSE loss over validation set of 0.0085. For the models with learning rates 10, 0.0001 and 0.1, training and validation set MSE loss plots and validation set 12-margin error plots are given together with the qualitative results on the validation set.

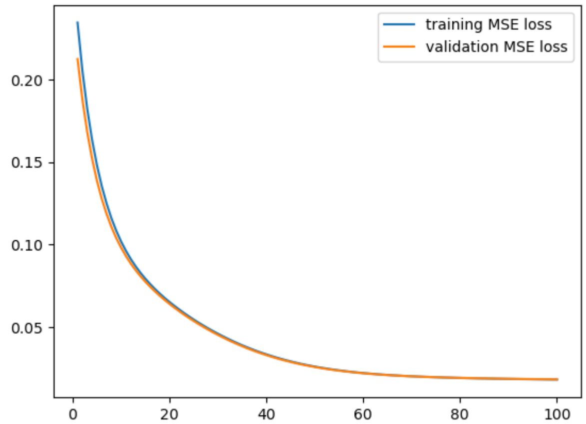


(a) Training and validation set MSE loss plot for learning rate 10 model

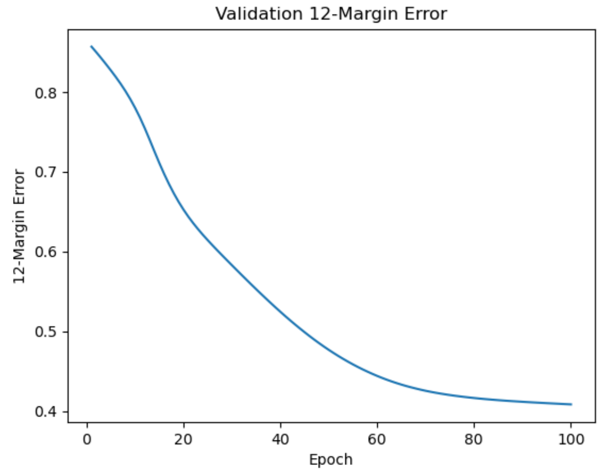
(b) Validation set 12-margin error plot for learning rate 10 model

Figure 13: Plots for learning rate 10 model

Figure 14: Gray scale images, Predictions and Targets for learning rate 10 model



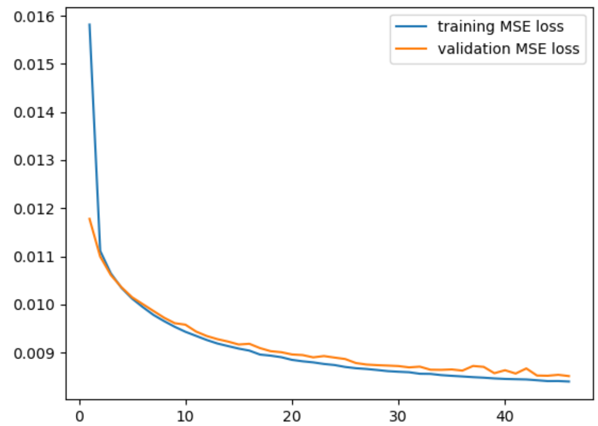
(a) Training and validation set MSE loss plot for learning rate 0.0001 model



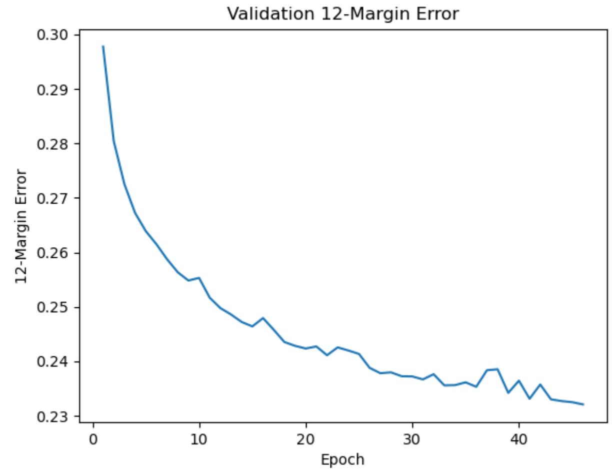
(b) Validation set 12-margin error plot for learning rate 0.0001 model

Figure 15: Plots for learning rate 0.0001 model

Figure 16: Gray scale images, Predictions and Targets for learning rate 0.0001 model



(a) Training and validation set MSE loss plot for learning rate 0.1 model



(b) Validation set 12-margin error plot for learning rate 0.1 model

Figure 17: Plots for learning rate 0.1 model

Figure 18: Gray scale images, Predictions and Targets for learning rate 0.1 model

2 Further Experiments (20 pts)

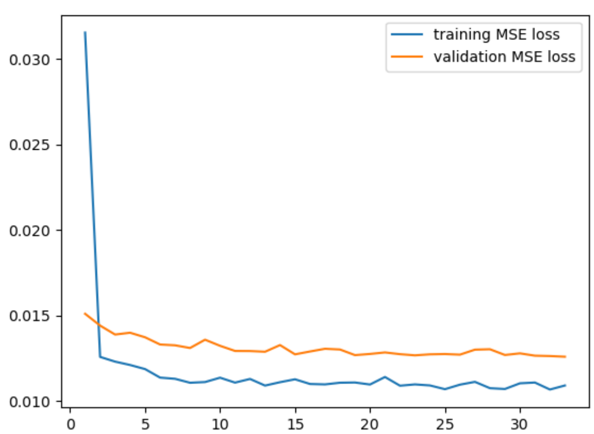
After observing model performances with different hyperparameters, the baseline architecture is finalized as 2 convolutional layers, 8 kernels and 0.1 learning rate since it gives the best results.

- Try adding a batch-norm layer (torch.nn.BatchNorm2d) into each convolutional layer. How does it affect the results, and, why? Keep it if it is beneficial.

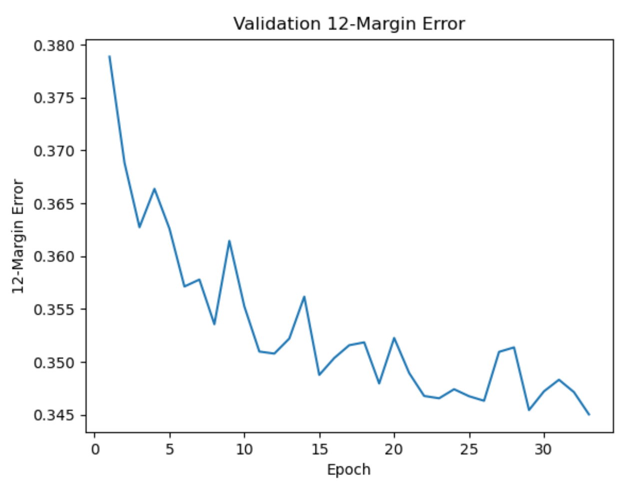
For the model with batch-norm layers, automatically chosen number of epochs is 33. For this model (2 conv layers, 8 kernels, 0.1 learning rate and batch-norm layers),

- training set MSE loss is 0.010905676161328825
- validation set MSE loss is 0.012587362065911293
- validation set 12-margin error is 0.34502774

Therefore, 12-margin error over validation set is increased with respect to baseline architecture while using batch-norm layers. Also, MSE loss over training and validation sets is increased too. Hence, adding batch-norm layer into each convolutional layer is decreased the model performance. This could be the case because of the size of the batches. If the batch size would be bigger than 16 (32, 64, 128, 256), results would be different since batch-normalization layer compute the empirical mean and variance for each dimension of the previous outputs across the batch. With higher batch sizes, these statistical estimations would be more accurate. Also, adding batch-norm layer into each convolutional layer may increase the model complexity in a model like baseline architecture (pretty much small). Batch-normalization allows higher learning rates, so I tried training the model with learning rates higher than 0.1 (0.2-0.5), but result were similar or worse. Also, I tried L2 regularization with the original learning rate 0.1 using different regularization strengths (0.1, 0.01, 0.001, 0.0001). With regularization strength 0.01, automatically chosen number of epochs is 8 and I managed to reduce the 12-margin error over validation set to 0.34150118 and MSE loss over training set to 0.010550340113881678 and MSE loss over validation set to 0.012412468764930963. However, compared to the baseline architecture, these results are still higher. Thus, I did not keep it. For the model with 2 convolutional layers, 8 kernels, 0.1 learning rate and batch-norm layers, training and validation set MSE loss plots and validation set 12-margin error plots are given together with the qualitative results on the validation set.



(a) Training and validation set MSE loss plot for the model with batch-norm layers



(b) Validation set 12-margin error plot for the model with batch-norm layers

Figure 19: Plots for the model with batch-norm layers

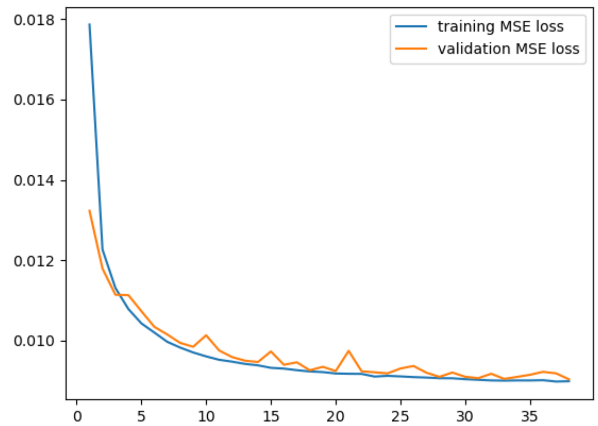
Figure 20: Gray scale images, Predictions and Targets for the model with batch-norm layers

- Try adding a tanh activation function after the very last convolutional layer. How does it affect the results, and, why? Keep it if it is beneficial.

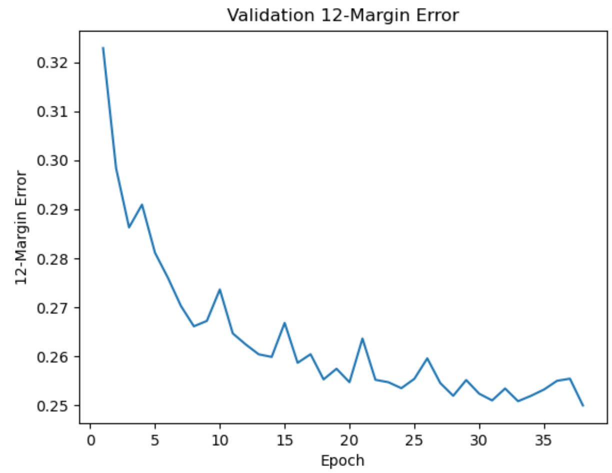
  For the model with tanh activation function after the last conv layer, automatically chosen number of epochs is 38. For this model (2 conv layers, 8 kernels, 0.1 learning rate and tanh activation function),

  - training set MSE loss is 0.008991535815496604
  - validation set MSE loss is 0.00904074165225029
  - validation set 12-margin error is 0.25000706

  Thus, 12-margin error over validation set is higher than the baseline architecture with adding tanh activation function after the last convolutional layer. Moreover, MSE loss over training and validation sets is also higher. Therefore, adding tanh activation function after the last convolutional layer hurt the model performance. This could be the case since tanh activation function kills gradients when saturated. tanh activation function is $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ and derivative of the tanh activation function is $g'(z) = 1 - g^2(z)$. Therefore, tanh activation function squashes inputs to range [-1, 1] and when saturated (output of the tanh activation function is near -1 or near 1), all gradients become zero or near zero. Hence, this effects training of the model and causes a slow learning process. Although the results are not dramatically different from the baseline model, they are still higher so the model performance is decreased. Thus, I did not keep it. For the model with 2 convolutional layers, 8 kernels, 0.1 learning rate and tanh activation function after the last convolutional layer, training and validation set MSE loss plots and validation set 12-margin error plots are given together with the qualitative results on the validation set.



(a) Training and validation set MSE loss plot for the model with tanh activation



(b) Validation set 12-margin error plot for the model with tanh activation

Figure 21: Plots for the model with tanh activation

Figure 22: Gray scale images, Predictions and Targets for the model with tanh activation

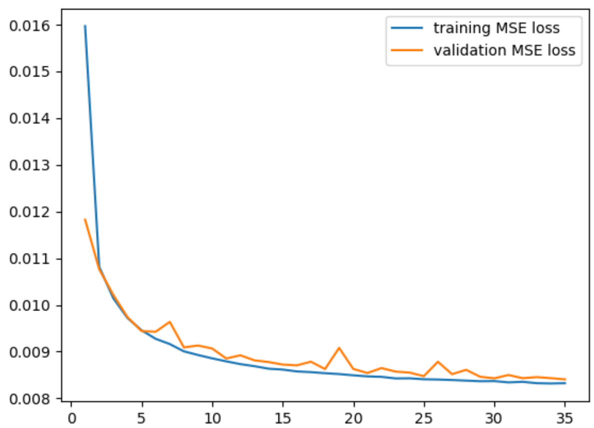
- Try setting the number of channels parameter to 16. How does it affect the results, and, why? Keep it if it is beneficial.

  For the model with 16 number of channels, automatically chosen number of epochs is 35. For this model (2 conv layers, 16 kernels, 0.1 learning rate),

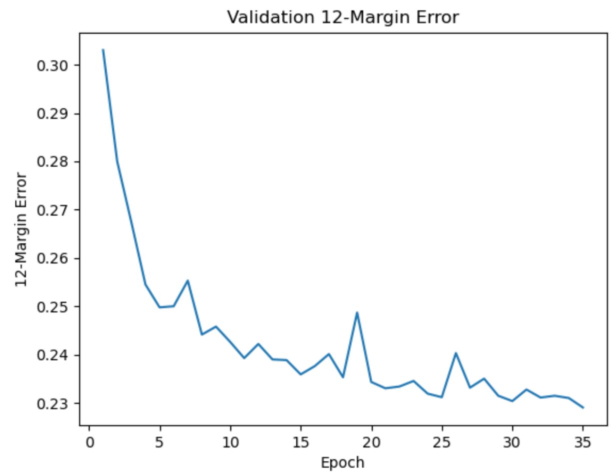
  - training set MSE loss is 0.008325346189137465
  - validation set MSE loss is 0.008407266069203615

- validation set 12-margin error is 0.2290477

The model with 16 channels has lower 12-margin error over validation set and MSE losses over training and validation sets than the baseline model. Thus, setting number of channels parameter to 16 increased the model performance (12-margin error over validation set with 16 channels ⇒ 0.2290, 8 channels ⇒ 0.2321). Increasing number of neurons increases the model capacity. Hence, each kernel produce a separate activation map. This means, increasing number of channels (increasing separate activation maps) makes model detect more accurate features of the input data which increases the model performance. Thus, I keep it. Finally, the best model configurations became 2 convolutional layers, 16 kernels, 0.1 learning rate. For the model with 2 convolutional layers, 16 kernels, 0.1 learning rate, training and validation set MSE loss plots and validation set 12-margin error plots are given together with the qualitative results on the validation set.



(a) Training and validation set MSE loss plot for kernel number 16 model



(b) Validation set 12-margin error plot for kernel number 16 model

Figure 23: Plots for kernel number 16 model

Figure 24: Gray scale images, Predictions and Targets for kernel number 16 model

3 Your Best Configuration (20 pts)

The best model that I obtained has the following hyperparameters: 2 convolutional layers, 16 kernels, 0.1 learning rate.

- The automatically chosen number of epochs (what was your strategy?): The automatically chosen number of epochs is 35. As I mentioned in the first section, an early break out method is used for automatic epoch detection with a patience of 5. Thus, this method compares the best 12-margin error over validation set with the 12-margin error over the validation set in the last epoch. If the 12-margin error in the last epoch is less than the best one, best 12-margin error becomes the error in the last epoch and the patience counter becomes its default value (0). Otherwise, counter is increased by 1. If the counter reaches 5, the model stops training with early break out. Hence, if the counter reaches 5, automatically chosen number of epochs would be the last epoch number in which early break out happened - 5.

- The plot of the training mean-squared error loss over epochs: In this section, plot of the training set MSE loss over epochs and plot of the validation set MSE loss over epochs are given together. Also, the last plot shows training set MSE loss and validation set MSE loss together on the same graph for cleaner comparison.

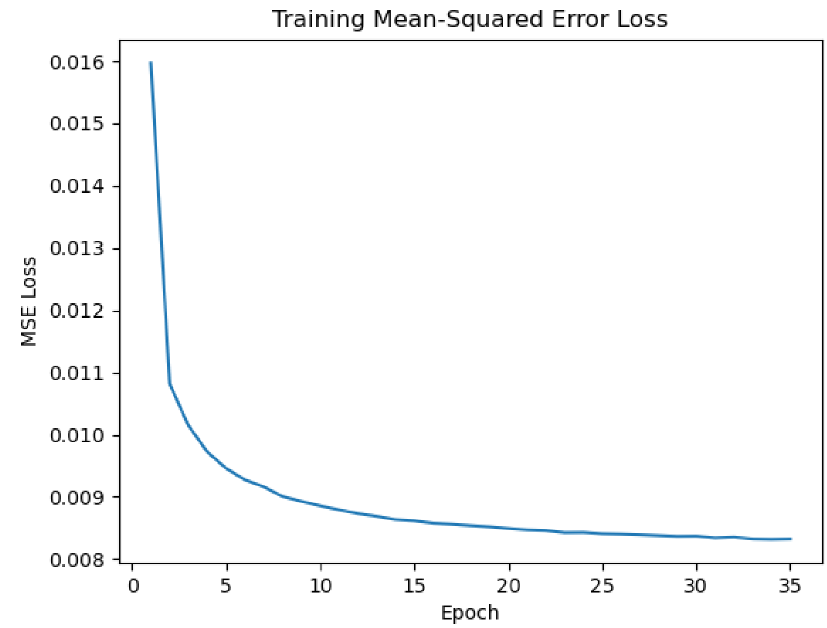


Figure 25: Training set MSE loss plot for the best model

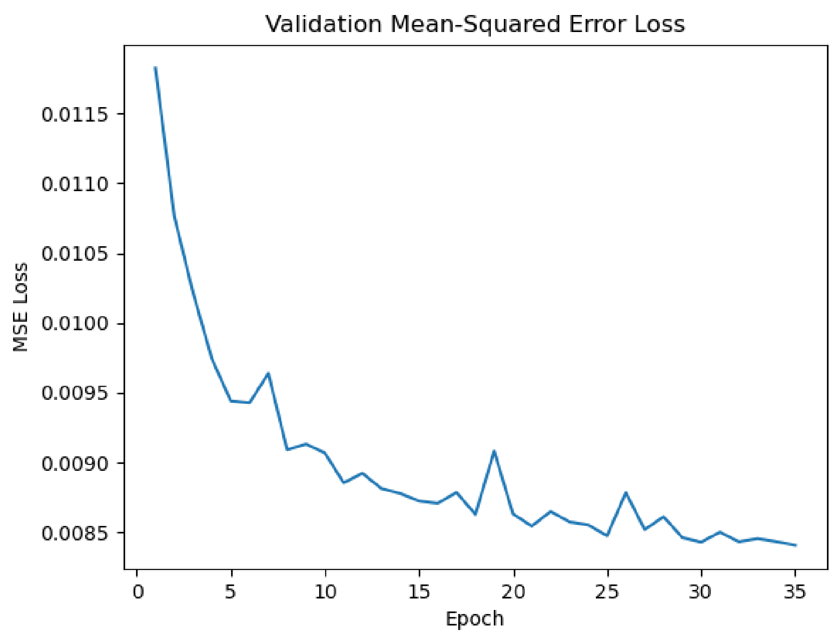


Figure 26: Validation set MSE loss plot for the best model

- At least 5 qualitative results on the validation set, showing the prediction and the target colored image:

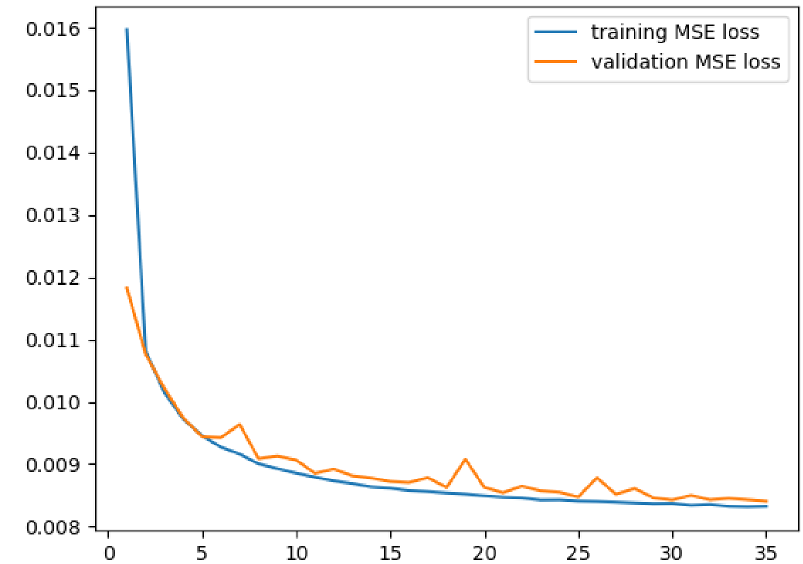


Figure 27: Training and validation set MSE loss plot for the best model

- The plot of the validation 12-margin error over epochs (see the3 text for details):

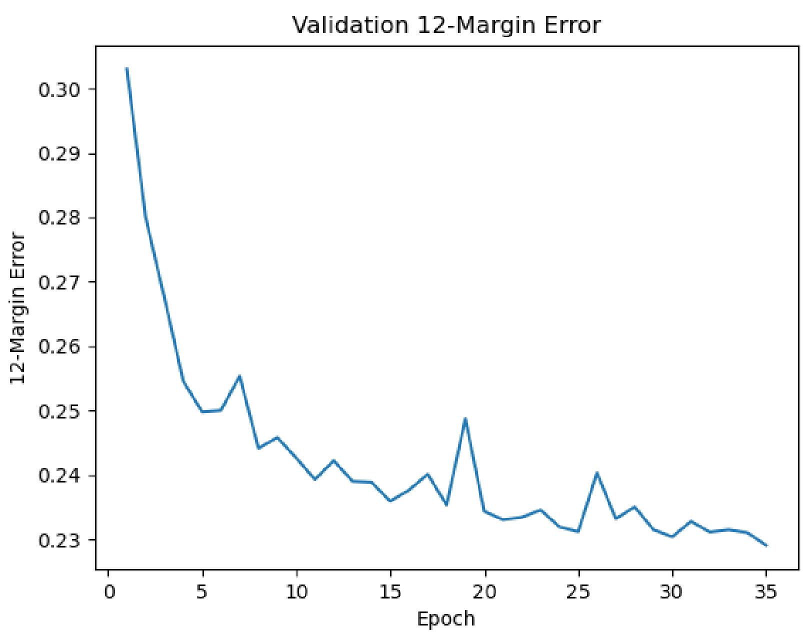


Figure 28: Validation set 12-margin error plot for the best model

- At least 5 qualitative results on the validation set, showing the prediction and the target colored image:

Figure 29: Gray scale images, Predictions and Targets for the best model part I

Figure 30: Gray scale images, Predictions and Targets for the best model part II

- Discuss the advantages and disadvantages of the model, based on your qualitative results, and, briefly discuss potential ways to improve the model:

Advantages of the model:

- The model can learn high level features in the input data without human intervention. It detects patterns automatically. Hence, it can generalize it to the unseen data.
- It works good enough on the validation set. 12-margin error over validation set is 0.2290. Thus, the accuracy is 0.771. Also, training set MSE loss is 0.008325 and validation set MSE loss is 0.008407. These results are acceptable.
- The model chooses the number of epochs in training automatically. Therefore, it tries to prevent overfitting to training data and increasing 12-margin error over validation set.

Disadvantages of the model:

- The model needs huge amount of labeled data for training. Training set size is 5001. Hence, if training set size would be smaller, the model performance could not be as good as the large training set size case. Therefore, generalization power would be decreased.
- Although the model performance results are not too bad, 12-margin error over validation set could be lower. After examining predicted images on the test set and validation set, there are some defects (odd blue and green colors), especially in the white or black colors on not all but some of the images. Thus, with the decreasing validation 12-margin error, these flaws could be lower.

Potential ways to improve the model:

- Applying learning rate update schedule could be beneficial since after some number of epochs, there are some fluctuations in the MSE loss over training set (it is decreasing and increasing one after another) and a little bit later, both 12-margin error and MSE loss over validation set are starting to increase. Therefore, at that stage, using smaller learning rates could prevent this to some extent.
- Using regularization could be beneficial for generalization performance. Thus, using L2/Dropout regularization could help the model performance while reducing overfitting and allow lower 12-margin error over validation set.
- Using a better working optimization algorithm such as Adam or RMSprop could be used to increase model performance rather than SGD.

4 Your Results on the Test Set (30 pts)

This part will be obtained by us using the estimations you will provide. Please tell us how should we run your code in case of a problem: Just run the file "the3.py" as "python the3.py". It will train the best model then load the model parameters to predict 100 color images from test dataset. Finally, it creates "estimations_test.npy" file with the predicted images. If you will run the code to create a new "estimations_test.npy", please make sure you delete "test_images.txt" file first because it is opened with "append" mode. You can examine model performance from "best.txt" file. If you want to observe predicted images, uncomment lines 344, 345, 357, then look at "tests" directory. If you want to plot graphs, uncomment lines 264-284, then look at "best" directory. Also, grayscale test images are extracted as "test/images_grayscale/x.jpg" since pytorch data loader wants 2 layer directory hierarchy. Thus, paths of the images were written like this to "test_images.txt".

5 Additional Comments and References

References

[1] "Neural Networks Part 1: Setting up the Architecture" class notes for CS231n: Convolutional Neural Networks for Visual Recognition, Dept. of Comp. Sci., Stanford University, Palo Alto, CA, USA, Spring 2023. Available: https://cs231n.github.io/neural-networks-1/

[2] "Neural Networks Part 2: Setting up the Data and the Loss" class notes for CS231n: Convolutional Neural Networks for Visual Recognition, Dept. of Comp. Sci., Stanford University, Palo Alto, CA, USA, Spring 2023. Available: https://cs231n.github.io/neural-networks-2/

[3] "Neural Networks Part 3: Learning and Evaluation" class notes for CS231n: Convolutional Neural Networks for Visual Recognition, Dept. of Comp. Sci., Stanford University, Palo Alto, CA, USA, Spring 2023. Available: https://cs231n.github.io/neural-networks-3/

[4] PyTorch Documentation (n.d.). https://pytorch.org/docs/stable/index.html#pytorch-documentation

[5] Welcome to PyTorch Tutorials (n.d.). https://pytorch.org/tutorials/