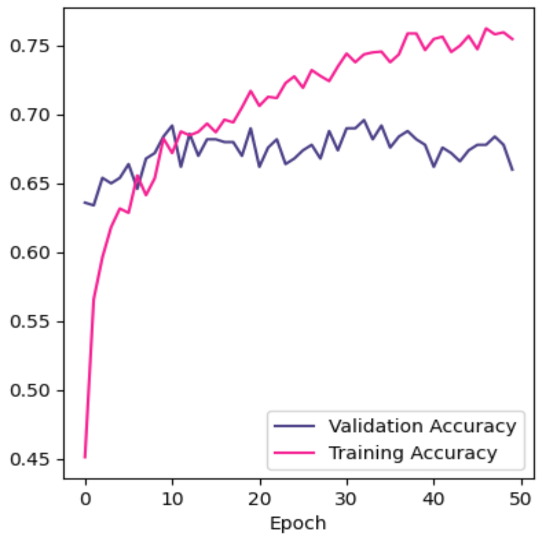
Commonalities

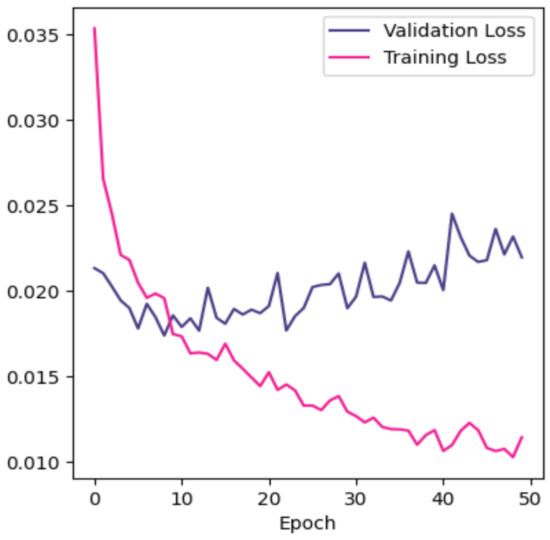
- 3500 training data, 500 validation data, 1000 test data
- Image was resized, converted to tensors, and normalized
- Took the tensors and network to CUDA (not sure if it's optimized correctly) (runs quite faster than Google Colab K-80, possibly due to SSD)
- Training and validation sets were permutated in random order
- Different batch values were tested (batch_size = 32, 64)
- If the model doesn't train better after the next Epoch, the previous one is maintained (i.e., lower accuracy and higher loss)
- Adam optimizer was used used without fully understanding its mechanism; lr and weight decay values could be made to dynamically change although I didn't explore that here
- Dropout (p = 0.5) was used to reduce the change of overfitting (p = 0.3 didn't help much)
-

Fully connected NN

- Layer 1: 512, Layer 2: 256, Layer 3: 128, output: 5
- Activation: ReLU, dropout
- nn.CrossEntropyLoss for loss function

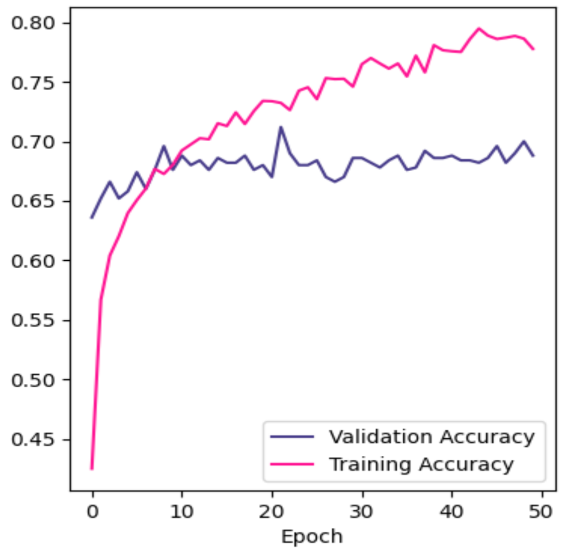
Batch_size = 32 (NN_32 files)

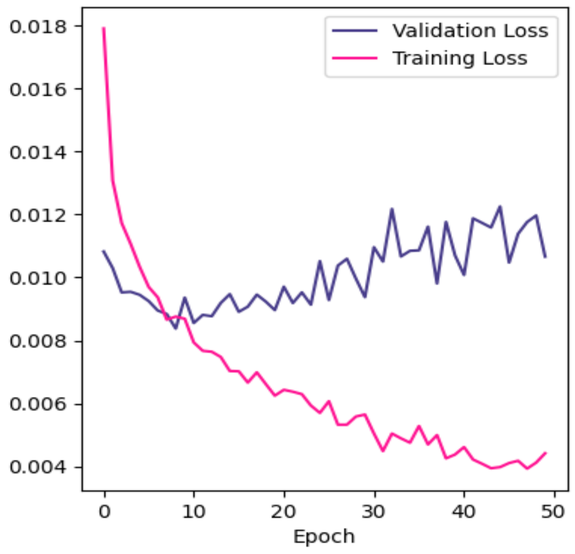




```
Best training loss: 0.019579
Best training accuracy: 0.653714
Best validation accuracy: 0.672000
Best validation loss: 0.017394
Test accuracy: 0.571000

Accuracy of class Grass : 68.421%
Accuracy of class Ocean : 71.875%
Accuracy of class Redcarpet : 87.500%
Accuracy of class  Road : 75.000%
Accuracy of class Wheatfield : 38.095%
```

Batch_size = 64 (NN_64 files)





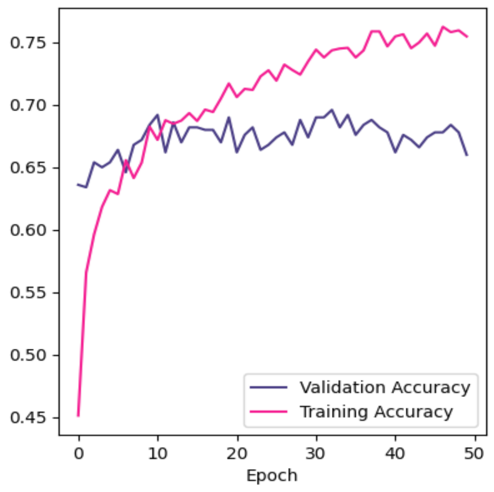
```
Best training loss: 0.008757
Best training accuracy: 0.672571
Best validation accuracy: 0.696000
Best validation loss: 0.008376
Test accuracy: 0.570000

Accuracy of class Grass : 50.000%
Accuracy of class Ocean : 72.222%
Accuracy of class Redcarpet : 98.400%
Accuracy of class  Road : 75.000%
Accuracy of class Wheatfield : 55.556%
```

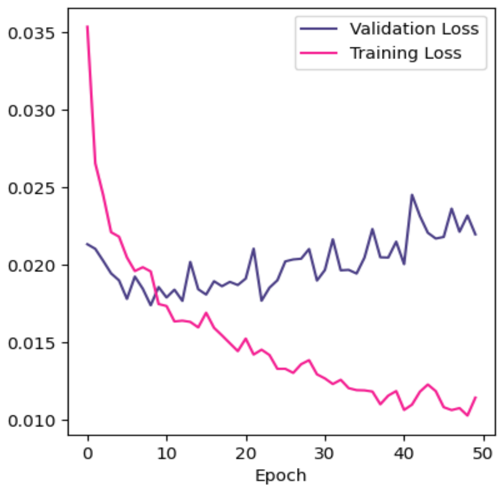
It's interesting that the accuracy for the image class Redcarpet is >95%. Given that there are some images in Redcarpet that are not so obvious, the network might be overfitted. Also, the network does not perform very well with distinguishing between Grass and Wheatfield. Frankly, they were hard to distinguish with bare eyes (my own). And the accuracy distribution is somewhat similar to the results from SVM (Grass vs Wheatfield is difficult).

Fully connected CNN

- Network : (Conv + Maxpool) * 3 -> linear on dense layer * 2
- Batch norm 2d following all of Conv layers
- Activation: ReLU / dropout
- nn.CrossEntropyLoss for loss function
- Maxpool has filter size 2*2, convolution has kernel size 3*3. I haven't tested out different values given the constraints.

Batch_size = 32 (CNN_32 files)



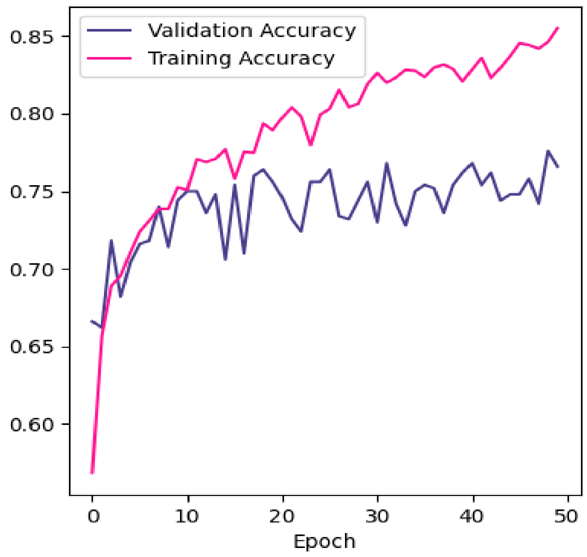


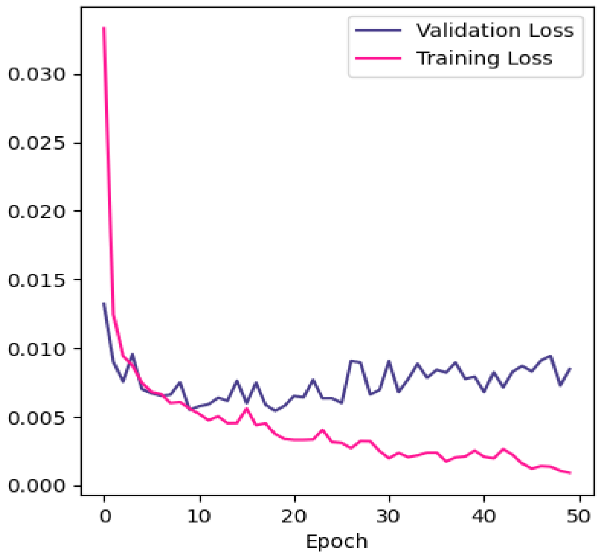
```
Best training loss: 0.019579
Best training accuracy: 0.653714
Best validation accuracy: 0.672000
Best validation loss: 0.017394
Test accuracy: 0.571000

Accuracy of class Grass : 68.421%
Accuracy of class Ocean : 71.875%
Accuracy of class Redcarpet : 87.500%
Accuracy of class  Road : 75.000%
Accuracy of class Wheatfield : 38.095%
```

```
batch size: 32

 Epoch      Training accuracy        Training loss      Validation accuracy        Validation loss
(01/50)          0.4511                 0.0354                 0.6360                  0.0213
(02/50)          0.5657                 0.0266                 0.6340                  0.0210
(03/50)          0.5963                 0.0245                 0.6540                  0.0203
(04/50)          0.6183                 0.0221                 0.6500                  0.0195
(05/50)          0.6317                 0.0218                 0.6540                  0.0190
(06/50)          0.6286                 0.0205                 0.6640                  0.0178
(07/50)          0.6557                 0.0196                 0.6460                  0.0192
(08/50)          0.6414                 0.0198                 0.6680                  0.0185
(09/50)          0.6537                 0.0196                 0.6720                  0.0174
(10/50)          0.6826                 0.0175                 0.6840                  0.0186
(11/50)          0.6720                 0.0173                 0.6920                  0.0179
(12/50)          0.6877                 0.0163                 0.6620                  0.0184
(13/50)          0.6849                 0.0164                 0.6860                  0.0177
(14/50)          0.6874                 0.0163                 0.6700                  0.0202
(15/50)          0.6934                 0.0160                 0.6820                  0.0184
(16/50)          0.6871                 0.0169                 0.6820                  0.0181
(17/50)          0.6963                 0.0159                 0.6800                  0.0189
(18/50)          0.6943                 0.0155                 0.6800                  0.0186
(19/50)          0.7051                 0.0149                 0.6700                  0.0189
(20/50)          0.7171                 0.0144                 0.6900                  0.0187
(21/50)          0.7063                 0.0152                 0.6620                  0.0191
(22/50)          0.7129                 0.0142                 0.6760                  0.0210
(23/50)          0.7120                 0.0145                 0.6820                  0.0177
(24/50)          0.7229                 0.0142                 0.6640                  0.0185
(25/50)          0.7277                 0.0133                 0.6680                  0.0190
(26/50)          0.7194                 0.0133                 0.6740                  0.0202
(27/50)          0.7323                 0.0130                 0.6780                  0.0204
(28/50)          0.7280                 0.0136                 0.6680                  0.0204
(29/50)          0.7243                 0.0139                 0.6880                  0.0210
(30/50)          0.7349                 0.0129                 0.6740                  0.0190
(31/50)          0.7443                 0.0127                 0.6900                  0.0197
(32/50)          0.7380                 0.0123                 0.6900                  0.0217
(33/50)          0.7437                 0.0126                 0.6960                  0.0197
(34/50)          0.7451                 0.0120                 0.6820                  0.0197
(35/50)          0.7457                 0.0119                 0.6920                  0.0194
(36/50)          0.7380                 0.0119                 0.6760                  0.0205
(37/50)          0.7437                 0.0118                 0.6840                  0.0223
(38/50)          0.7589                 0.0110                 0.6880                  0.0205
(39/50)          0.7589                 0.0116                 0.6820                  0.0205
(40/50)          0.7469                 0.0119                 0.6780                  0.0215
(41/50)          0.7549                 0.0106                 0.6620                  0.0200
(42/50)          0.7566                 0.0110                 0.6760                  0.0245
(43/50)          0.7454                 0.0118                 0.6720                  0.0232
(44/50)          0.7500                 0.0123                 0.6660                  0.0221
(45/50)          0.7571                 0.0119                 0.6740                  0.0217
(46/50)          0.7474                 0.0108                 0.6780                  0.0218
(47/50)          0.7626                 0.0106                 0.6780                  0.0236
(48/50)          0.7583                 0.0108                 0.6840                  0.0221
(49/50)          0.7597                 0.0103                 0.6780                  0.0232
(50/50)          0.7549                 0.0114                 0.6600                  0.0220
```

validation loss increases over epochs

Batch_size = 64 (CNN_64 files)





```
Best training loss: 0.005633
Best training accuracy: 0.750571
Best validation accuracy: 0.750000
Best validation loss: 0.005495
Test accuracy: 0.620000

Accuracy of class Grass : 81.818%
Accuracy of class Ocean : 75.000%
Accuracy of class Redcarpet : 100.000%
Accuracy of class  Road : 92.857%
Accuracy of class Wheatfield : 60.000%
```

From the results, batch size of 32 seems optimal. First, batch size of 64 seems to overfit to the data. Compared to 32, when it's 64, the validation accuracy fluctuates more up and down. I'm assuming that is because the network is somehow more sensitive to the noise in the data set. However, Batch of 32 still does not do very well in classifying the wheatfield, and it's the same for batch of 64: wheatfield is the hardest one of all to classify.

Based on the steepness of the validation loss and slow decrease in the training loss, I'd like to say that batch of 64 is actually doing really well if it wasn't the case that it probably overfitted. To get better performance on test data, I'd try different kernel sizes, dropout percentages, different shapes of networks such that the dimension of it can protect against overfitting and such. Also, different activation functions and smaller epochs could be tried to reduce overfitting.