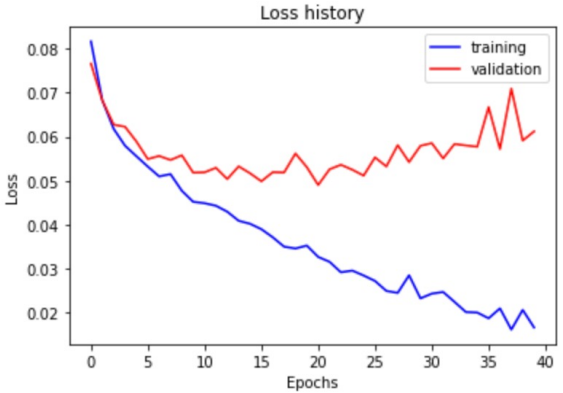
CS 4476/6476 Project 4

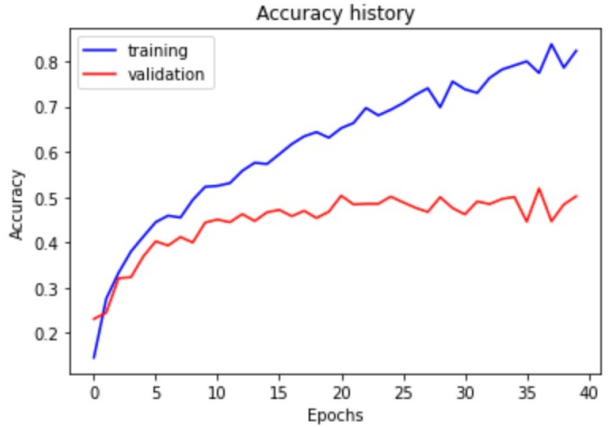
[Zixin Yin]
[zyin81@gatech.edu]
[zyin81]
[903718320]

Part 1: SimpleNet

[Insert loss plot for SimpleNet here]



[Insert accuracy plot for SimpleNet here]



Final training accuracy:0.8231155778894472

Final validation accuracy:0.502

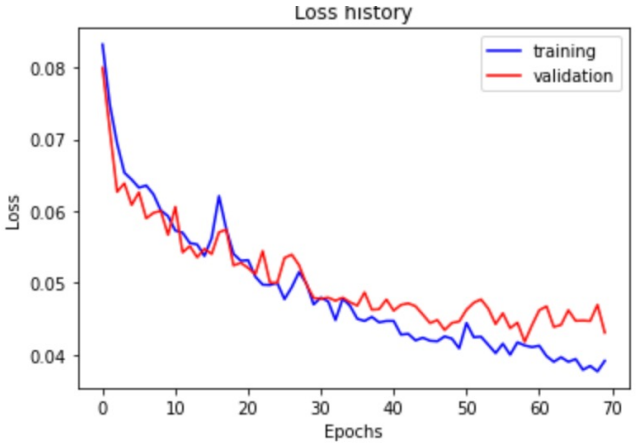
Part 2: SimpleNetFinal

Add each of the following (keeping the changes as you move to the next row):

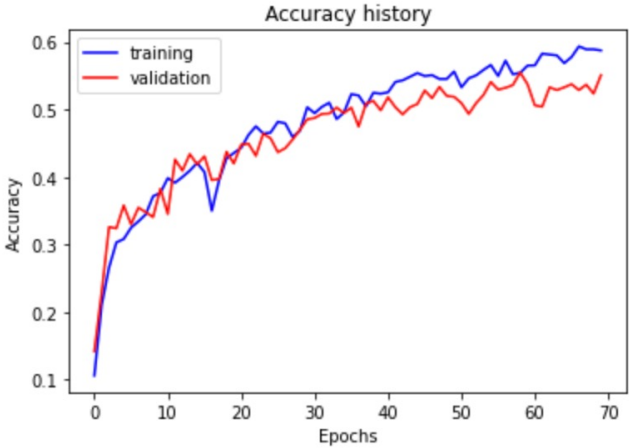
	Training accuracy	Validation accuracy
SimpleNet	0.8231	0.502
+ Jittering	0.7856	0.511
+ Zero-centering & variance-normalization	0.5889	0.523
+ Dropout regularization	0.5799	0.5287
+ Making network "deep"	0.5685	0.5327
+ Batch normalization	0.5889	0.5367

Part 2: SimpleNetFinal

[Insert loss plot for SimpleNetFinal here]



[Insert accuracy plot for SimpleNetFinal here]



Final training accuracy: 0.5872696817420435

Final validation accuracy: 0.5506666666666666

Part 2: SimpleNetFinal

[Name 10 different possible transformations for data augmentation.]

1. Resize
2. ColorJitter
3. randomHorizontalFlip
4. Normalize
5. centerCrop
6. randomRotation
7. randomVerticalFlip
8. GaussianBlur
9. randomAdjustSharpness
10. randomAutocontrast

[What is the desired variance after each layer? Why would that be helpful?]

The desired variance should be 1 and stay the same through each layer.
This will help each layer to learn at the same pace.

Part 2: SimpleNetFinal

[What distribution is dropout usually sampled from?]

Bernoulli distribution

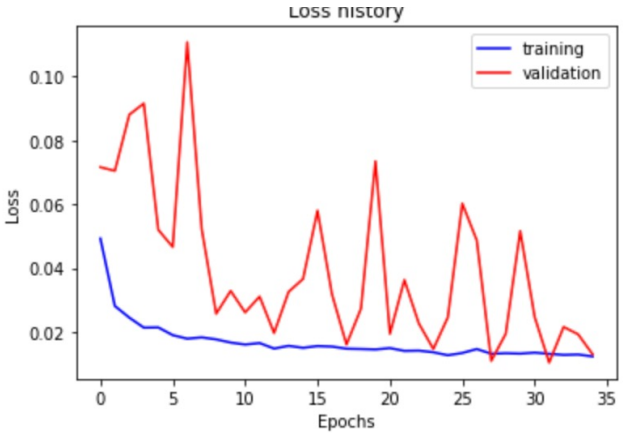
[How many parameters does your base SimpleNet model have? How many parameters does your SimpleNetFinal model have?]
Base model:
56895
SimpleNetFinal:
31210

[What is the effect of batch norm after a conv layer with a bias?]

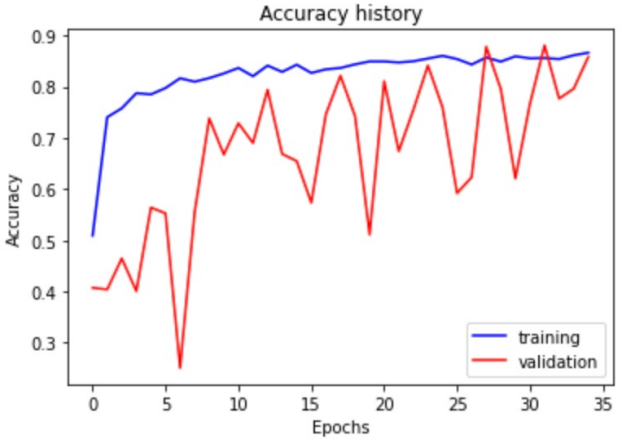
Cancel out the bias because BatchNorm shifts the activation by their mean values, hence, any constant will be canceled out.

Part 3: ResNet

[Insert loss plot here]



[Insert accuracy plot here]

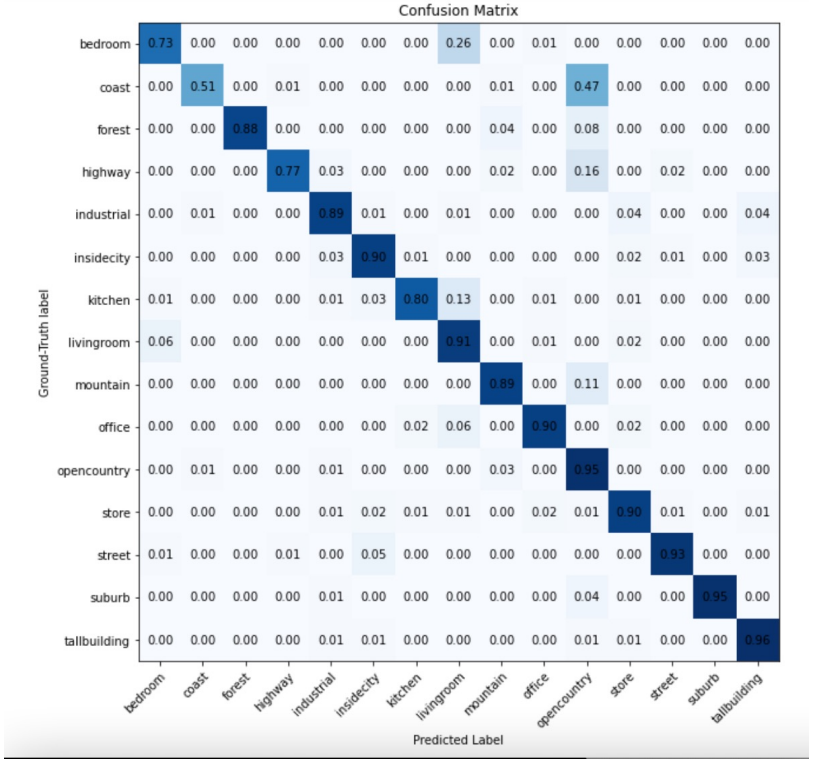


Final training accuracy: 0.8663316582914573

Final validation accuracy: 0.858

Part 3: ResNet

[Insert visualization of confusion matrix obtained from your final ResNet model.]



Part 3: ResNet

[Insert visualizations of 3 misclassified images from the most misclassified class according to your confusion matrix. Explain why this may have occurred.]



Reasoning: sea scene can be misinterpreted as grass or land because both of them have either very little change(Sand and stable sea surface) or some small textures(sea waves and grass)

Part 3: ResNet

[What does fine-tuning a network mean?]

It means we take the weights of a trained neural network and use it as initialization for a new model being trained on data from the same domain. So that we can speed up the training and overcome small dataset size.

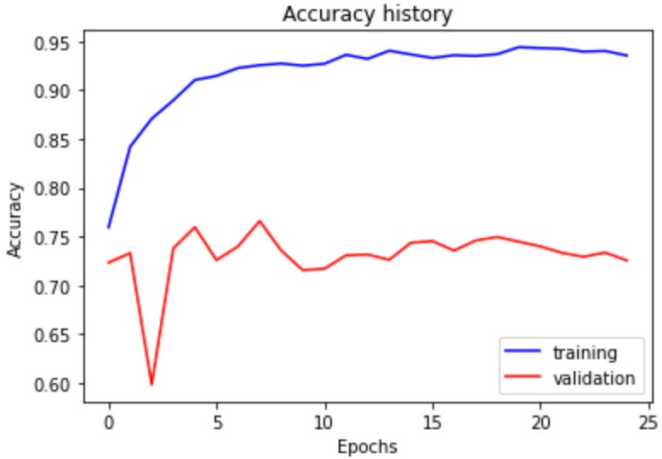
[Why do we want to "freeze" the conv layers and some of the linear layers from a pre-trained ResNet? Why can we do this?]

Freeze the layer means we cannot modify its weight later on, and it helps with saving computational time.

Part 4: Multi-label Scene Attributes

[Insert loss plot here]

[Insert accuracy plot here]



Final training accuracy: 0.9353798375537503

Final validation accuracy: 0.7257142857142859

Part 4: Multi-label Scene Attributes

[Insert visualization of accuracy table obtained from your final MultilabelResNet model.]

Part 4: Multi-label Scene Attributes

[List 3 changes that you made in the network compared to the one in part 3.]

1. Added sigmoid layer
2. Used BCELoss instead of CrossEntropyLoss
3. Changed out_features to 7

[Is the loss function of the ResNet model from part 3 appropriate for this problem? Why or why not?]

No.
Because we can use **sigmoid** to process the single output while using BCELoss as the loss function.
We cannot use sigmoid for 2 output features while using CrossEntropyLoss as the loss function.

Part 4: Multi-label Scene Attributes

[Explain a problem that one needs to be wary of with multilabel classification. HINT: consider the purpose of visualizing your results with the accuracy table. You might want to do some data exploration here.]

The data imbalance problem.
In multi-label classification, one target variable has multiple dimensions. In some cases, the numbers of instances of some dimensions are significantly larger than the instances of other dimensions. In this case, our model will tend to predict the dimensions with more instances as the results.