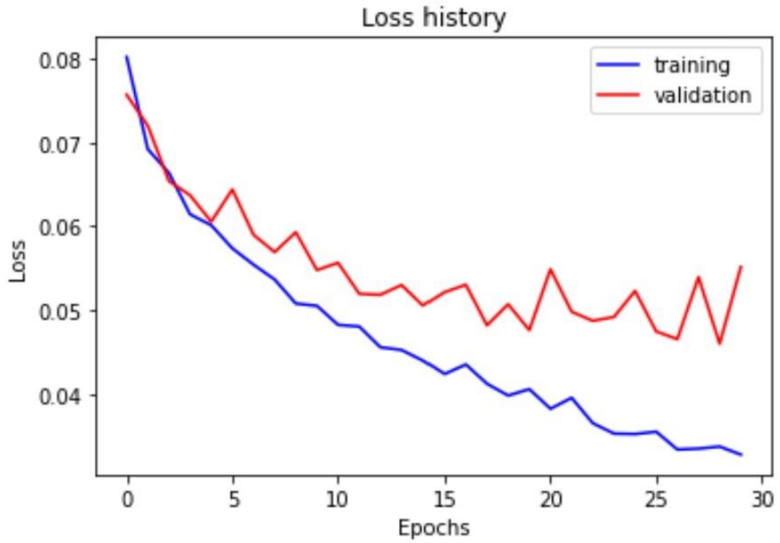
CS 4476/6476 Project 4

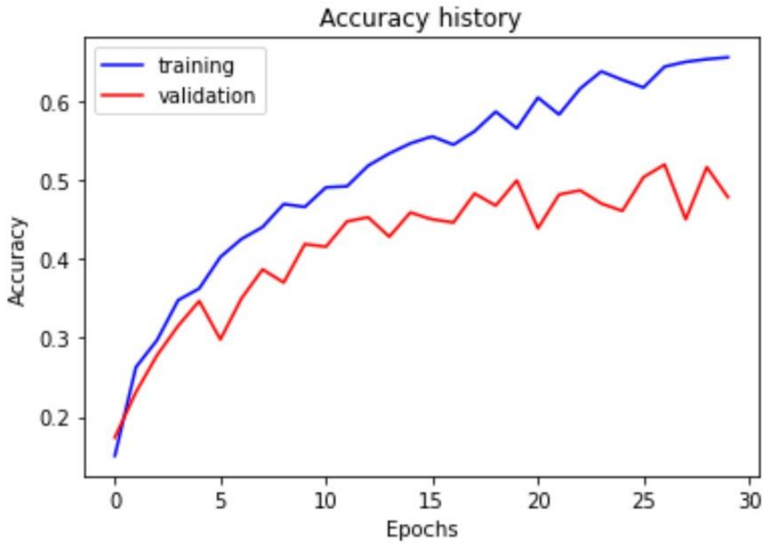
[Manan Patel]
[mpatel608@gatech.edu]
[mpatel608]
[903748003]

Part 1: SimpleNet

[Insert loss plot for SimpleNet here]



[Insert accuracy plot for SimpleNet here]



Final training accuracy: `0.6549413735343383`

Final validation accuracy: `0.478`

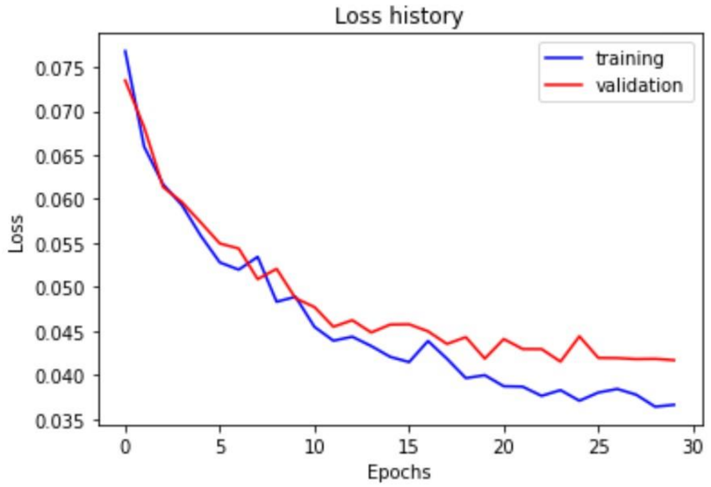
Part 2: SimpleNetFinal

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

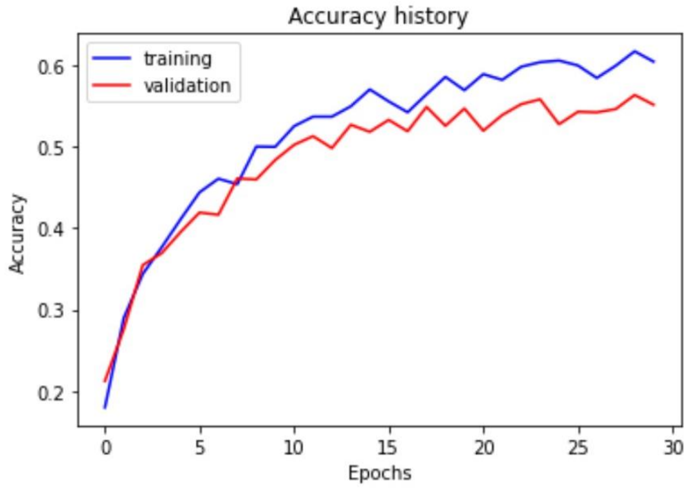
	Training accuracy	Validation accuracy
SimpleNet	0.65	0.47
+ Jittering	0.668	0.492
+ Zero-centering & variance-normalization	0.826	0.5253
+ Dropout regularization	0.6864	0.524
+ Making network "deep"	0.656	0.53066
+ Batch normalization	0.649	0.55

Part 2: SimpleNetFinal

[Insert loss plot for SimpleNetFinal here]



[Insert accuracy plot for SimpleNetFinal here]



Final training accuracy: `0.6050251256281407`

Final validation accuracy: `0.552`

Part 2: SimpleNetFinal

[Name 10 different possible transformations for data augmentation.]

- Horizontal image flip
- Brightness altering
- Cropping
- Rotating
- Saturation altering
- Scaling
- Contrast altering
- Translating the image
- Affine transforms
- Hue altering

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

- The variance in general should be low and decreasing with each layer.
- Reason being, if the variance is high, the network may tend to overfit and it won’t be general enough to classify test images which are significantly different than the train images.
- With low variance, our ‘feature detector’ will be able to generalize the features and that will be useful to boost test accuracy.

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?]

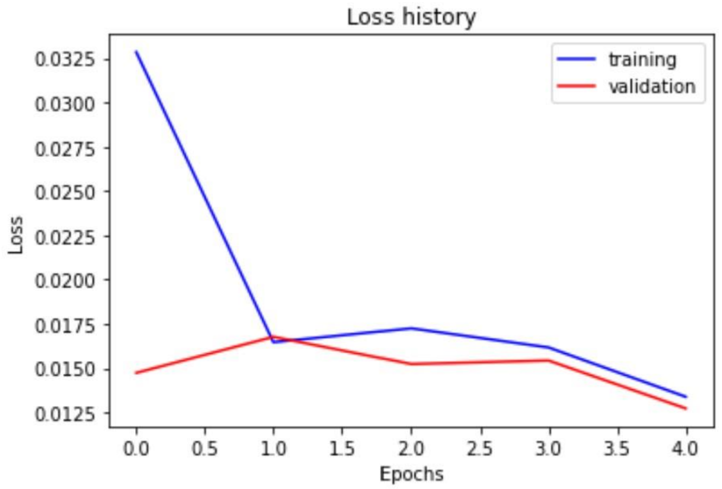
- SimpleNet: 56895
- SimpleFinalNet: 57765

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

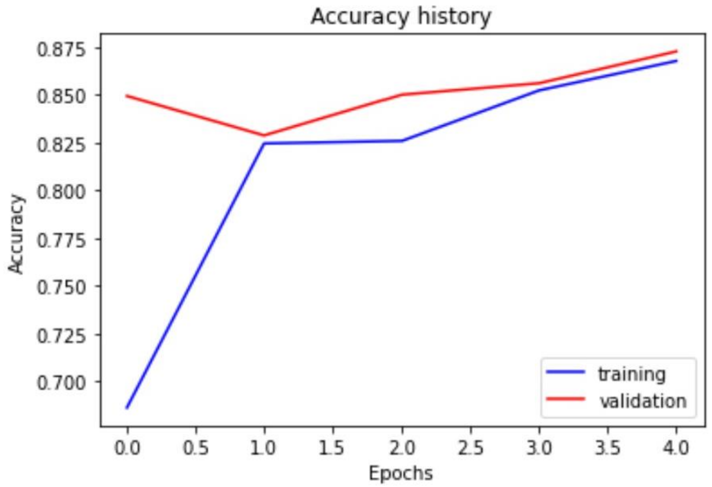
- Batch norm normalizes the activation output from the previous layer to a standard normal distribution.
- In relation to the conv layer with bias, it reduces the variance between all the output channels thus allowing for more stable and faster learning.

Part 3: ResNet

[Insert loss plot here]



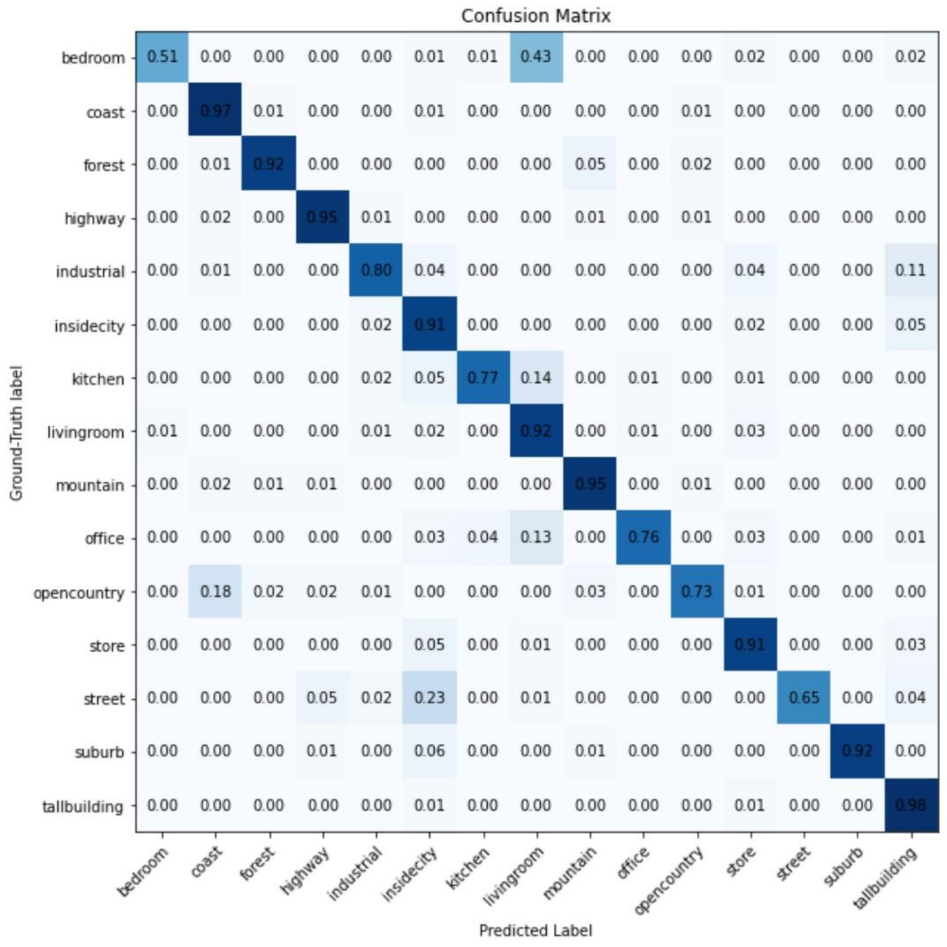
[Insert accuracy plot here]



Final training accuracy: `0.8676716917922948`

Final validation accuracy: `0.872666666666`

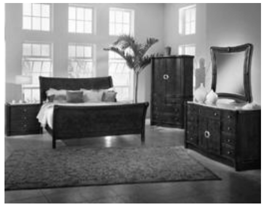
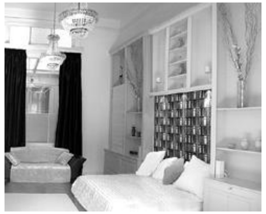
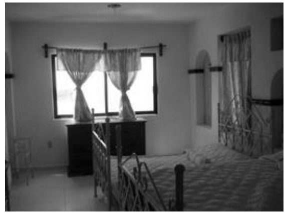
Part 3: ResNet



	bedroom	coast	forest	highway	industrial	insidecity	kitchen	livingroom	mountain	office	opencountry	store	street	suburb	tallbuilding
bedroom	0.51	0.00	0.00	0.00	0.00	0.01	0.01	0.43	0.00	0.00	0.00	0.02	0.00	0.00	0.02
coast	0.00	0.97	0.01	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00
forest	0.00	0.01	0.92	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.02	0.00	0.00	0.00	0.00
highway	0.00	0.02	0.00	0.95	0.01	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.00
industrial	0.00	0.01	0.00	0.00	0.80	0.04	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.00	0.11
insidecity	0.00	0.00	0.00	0.00	0.02	0.91	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.05
kitchen	0.00	0.00	0.00	0.00	0.02	0.05	0.77	0.14	0.00	0.01	0.00	0.01	0.00	0.00	0.00
livingroom	0.01	0.00	0.00	0.00	0.01	0.02	0.00	0.92	0.00	0.01	0.00	0.03	0.00	0.00	0.00
mountain	0.00	0.02	0.01	0.01	0.00	0.00	0.00	0.00	0.95	0.00	0.01	0.00	0.00	0.00	0.00
office	0.00	0.00	0.00	0.00	0.00	0.03	0.04	0.13	0.00	0.76	0.00	0.03	0.00	0.00	0.01
opencountry	0.00	0.18	0.02	0.02	0.01	0.00	0.00	0.00	0.03	0.00	0.73	0.01	0.00	0.00	0.00
store	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.01	0.00	0.00	0.00	0.91	0.00	0.00	0.03
street	0.00	0.00	0.00	0.05	0.02	0.23	0.00	0.01	0.00	0.00	0.00	0.00	0.65	0.00	0.04
suburb	0.00	0.00	0.00	0.01	0.00	0.06	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.92	0.00
tallbuilding	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.98

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.]

1. For the first image, the angle is such that a majority of the area of the bed is hidded. Since, the most significant feature of a bed is its long spanning topside, it gets missclassifed.
2. Here, the bed looks more like a sofa than a bed. Also, after the image being converted to grayscale, a majority of the image is white and it makes it difficult to make a distinction between the attribute and the background.
3. Here, a combination of the issues mentioned in 1 and 2 is faced by the classifer. A part of the bed is hidden, moreover, due to low lighting it makes it difficult to separate out the bed from rest of the background.

Part 3: ResNet

[What does fine-tuning a network mean?]
- Fine-tuning means first updating a pre-trained network to output classes which are desried for our application.

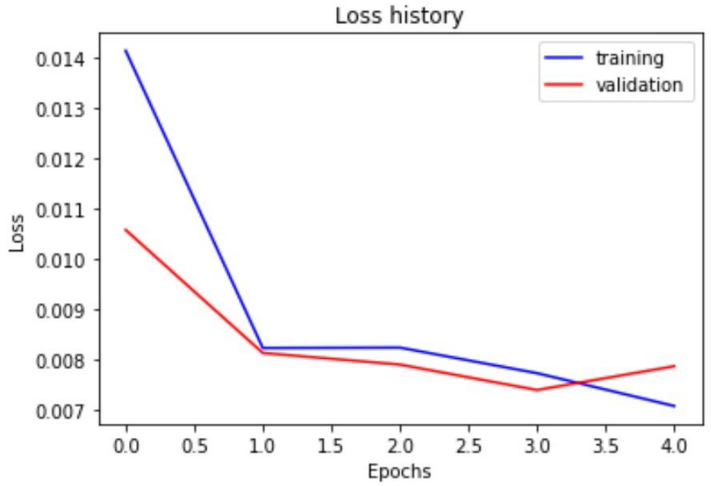
- Then, freezing a majority of layers which already have the required set of weights and then training the network for a small number of layers which were added for our application.
- With, this, we can use the knowledge of the pre-trained network and train a small amount of new weights, instead of training the whole network froms scratch.

[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?]

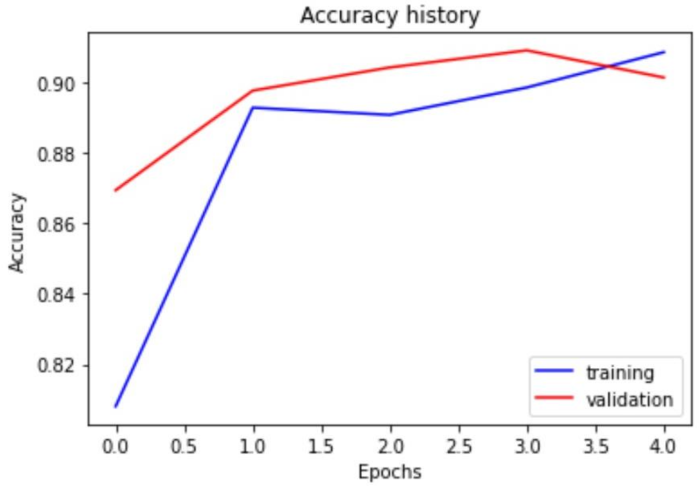
- This is known as transfer learning
- This is to avoid repeating the training process from scratch, given that the application that we desire is similar to the existing pre-trained network.
- This saves us from a lot of computational time especially when one does not have access to GPUs.

Part 4: Multi-label Scene Attributes

[Insert loss plot here]



[Insert accuracy plot here]

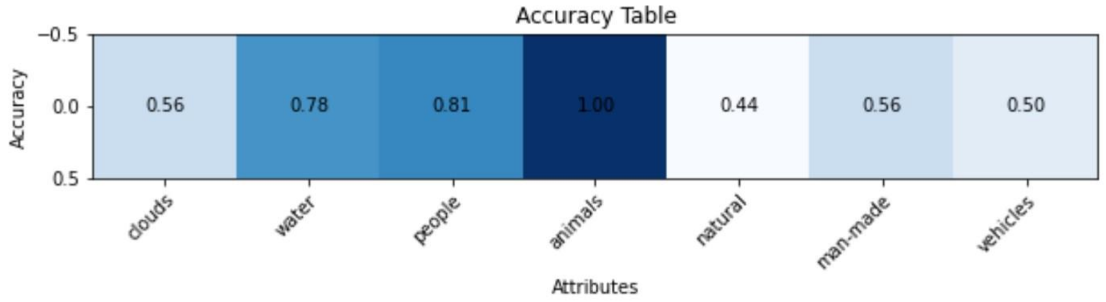


Final training accuracy: `0.9086`

Final validation accuracy: `0.9014`

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.]

- Added a sigmoid activation function to output probabilities for each attribute
- Chagned the loss from cross entropy to Binary cross entropy since we are doing multi-class output label identification
- Changed the output channels from 15 to 7

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

- No it is not valid since cross entropy loss is weighs the probability output of all variables to all the probability value of the other variables.
- As a result, the loss is not independent while classifying attributes.
- We want the loss to propogate for each attribute independently

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.]

- High dimensionality is one of the most challenging problems to solve in case of multilabel classification.
- For many lables, the total number of training instances are less compared to the total number of instances in the whole dataset. Thus, there is an imbalance in class distribution while training.
- A very big amount of data can fix the problem, but then again, it would not be readily available.
- Also corelation between labels is a problem. The label instances are not no longer mutually exclusive, and thus, the network might tend to learn an incorrect label which is highly co-related to the correct one.