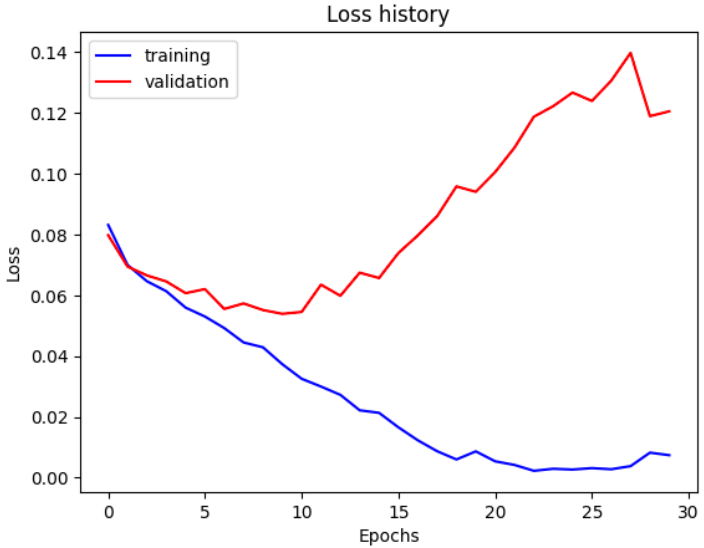
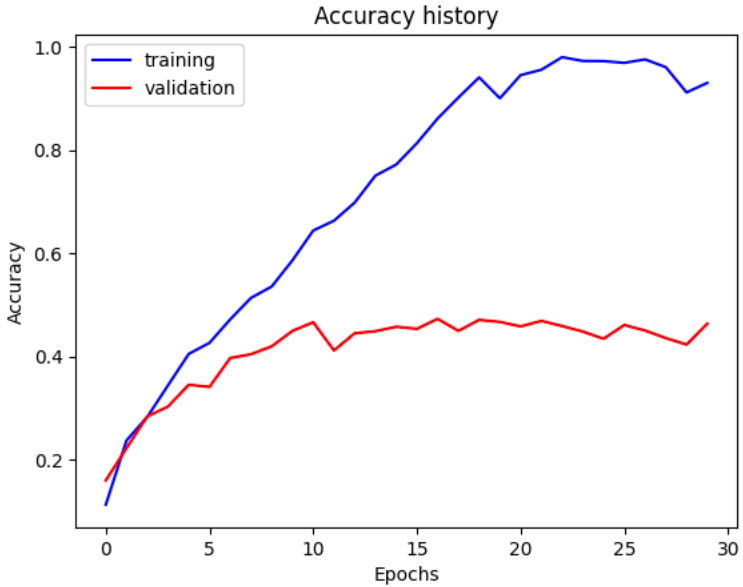
CS 4476 Project 4

Vidit Pokharna
vidit@gatech.edu
vpokharna3
903772087

Part 1: SimpleNet





Final training accuracy: 0.930

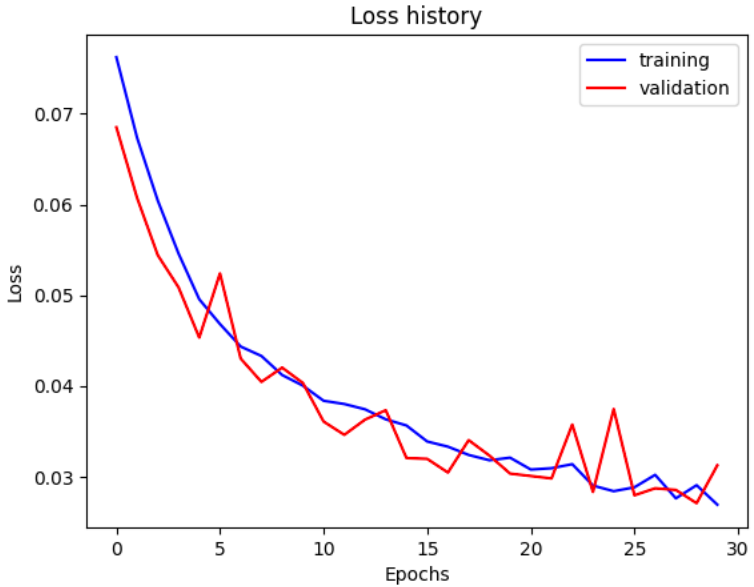
Final validation accuracy: 0.463

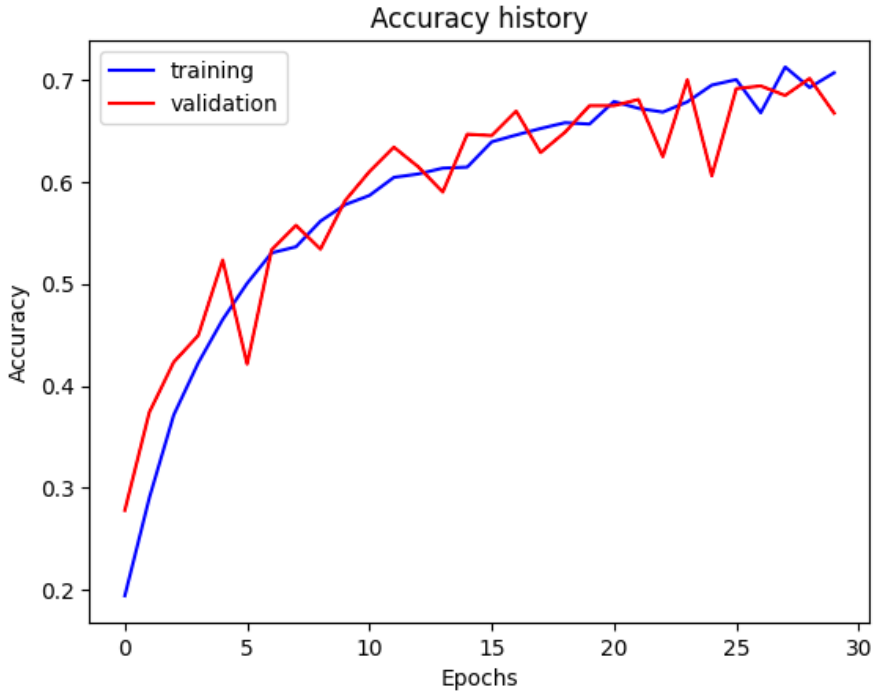
Part 2: SimpleNetFinal

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

	Training accuracy	Validation accuracy
SimpleNet	0.930	0.463
+ Jittering	0.540	0.521
+ Zero-centering & variance-normalization	0.542	0.548
+ Dropout regularization	0.537	0.593
+ Making network "deep"	0.686	0.640
+ Batch normalization	0.707	0.667

Part 2: SimpleNetFinal





Final training accuracy: 0.707

Final validation accuracy: 0.667

Part 2: SimpleNetFinal

Transformations for data augmentation:

1. Random Rotation
2. Random Crop
3. Horizontal Flip
4. Vertical Flip
5. Color Jitter
6. Random Resizing and Cropping
7. Random Perspective Transformation
8. Random Affine Transformation
9. Gaussian Blur
10. Adding Noise

The desired variance after each layer typically refers to keeping the activations at each layer of the network to have a unit variance, which prevents the gradients from vanishing or exploding, especially in deep networks. This is often achieved using batch normalization layers after each convolutional or linear layer. Keeping variance in check helps in maintaining a stable and faster convergence during training. It ensures that the distribution of the activations remains more consistent across different layers, allowing higher learning rates and reducing the sensitivity to the initial weights.

Part 2: SimpleNetFinal

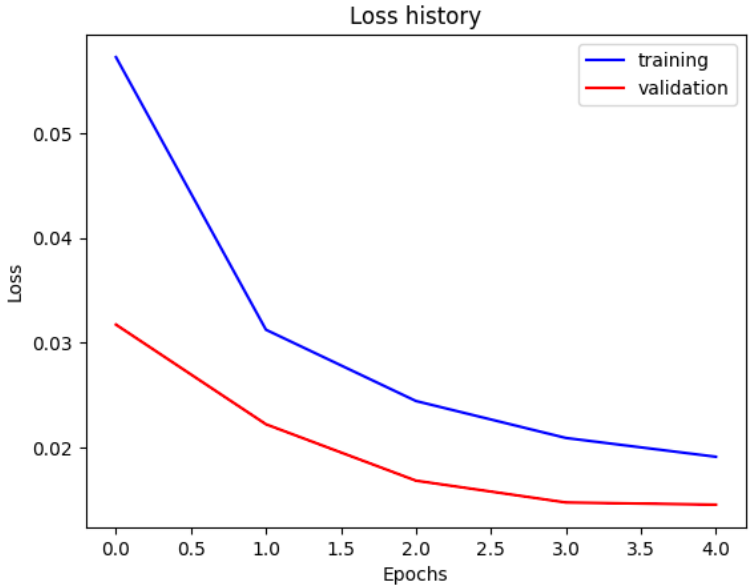
Dropout is usually sampled from a Bernoulli distribution. During training, for each layer that dropout is applied to, each neuron (or node) is kept active with a probability p (commonly set to 0.5) and is set to zero with probability 1 - p.

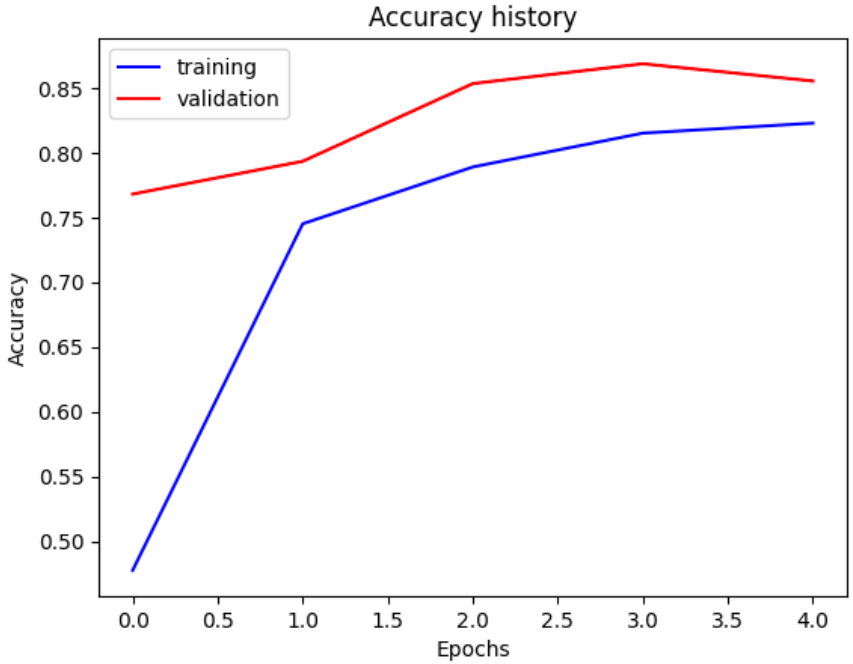
Parameters for SimpleNet: 307395
Parameters for SimpleNetFinal: 2527439

The effect of batch normalization after a convolutional layer with bias compared to not using batch normalization (batch norm) can be quite significant:

- With Batch Norm: It normalizes the output of the convolutional layer by subtracting the batch mean and dividing by the batch standard deviation, which helps in stabilizing the learning process. The bias becomes redundant because the mean subtraction cancels it out, hence the bias term can be omitted without loss of functionality.
- Without Batch Norm: The convolutional layer outputs are solely scaled and biased by the layer's filters and bias terms. Without batch normalization, the distributions of the outputs could vary greatly during training, which could lead to the vanishing or exploding gradient problems.

Part 3: ResNet

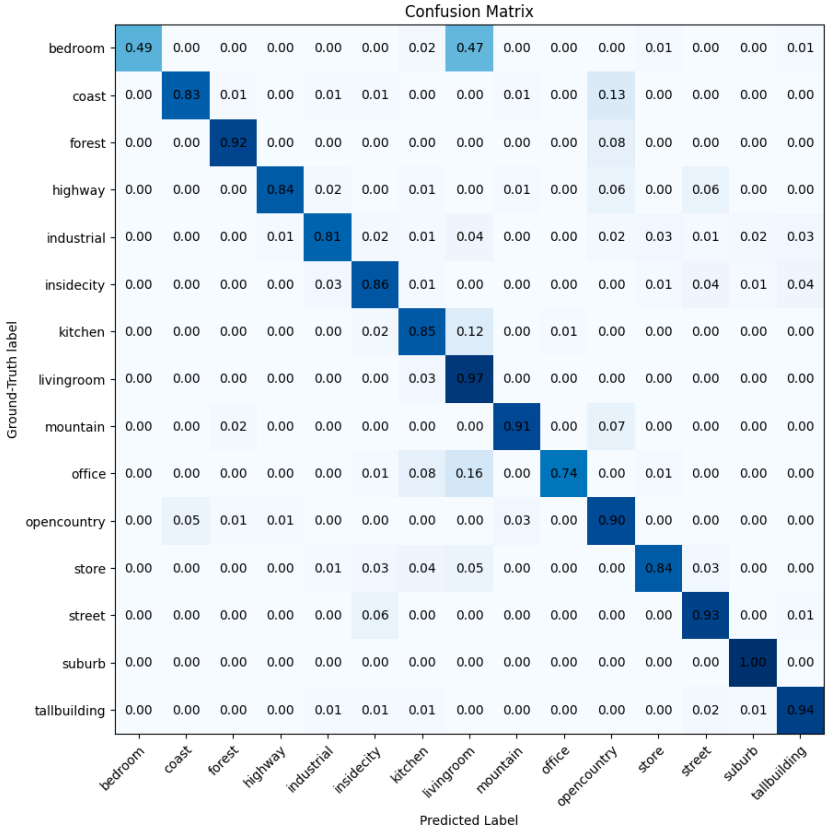




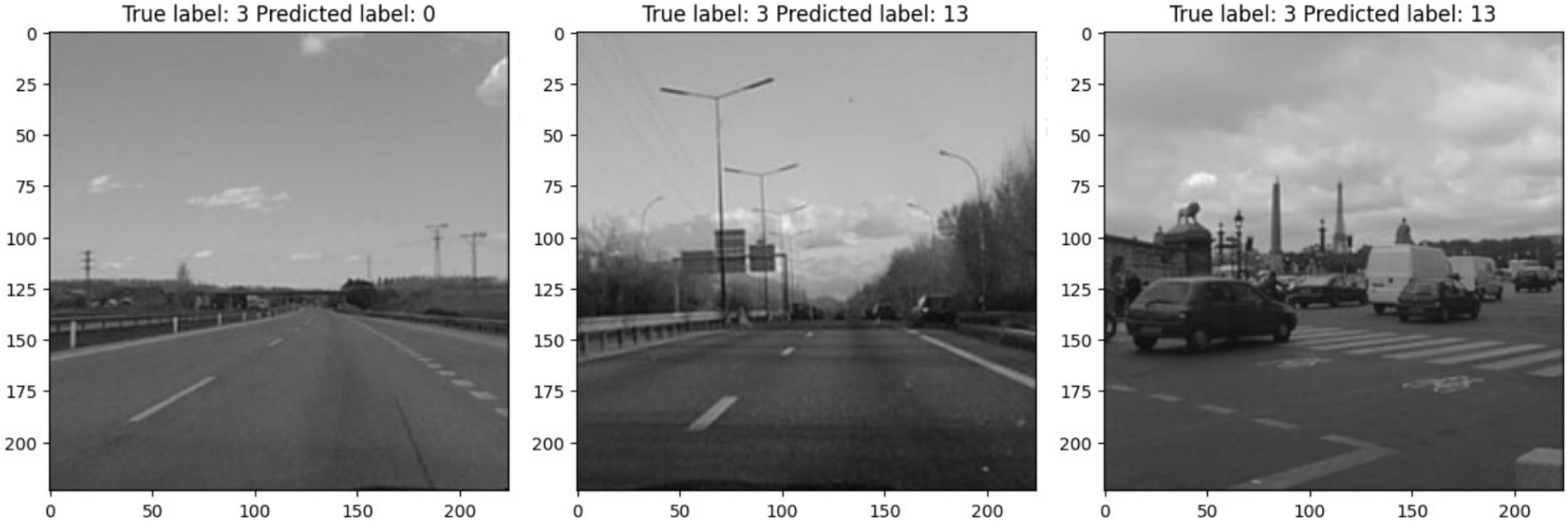
Final training accuracy: 0.823

Final validation accuracy: 0.855

Part 3: ResNet



Part 3: ResNet



This could have happened due to multiple reasons. The true label "3" could correspond to a class of images with features like roads, vehicles, or open spaces which are also present in the class corresponding to the predicted label "13". If these features are dominant, the model may struggle to differentiate between these classes. Additionally, if there weren't enough training examples for the model to learn the nuances between the two classes effectively, it may not have developed a detailed enough understanding to distinguish them. The model also might be overfitting on the training data, capturing noise that does not generalize to the test set. Conversely, underfitting would mean the model has not learned enough detail about the data.

Part 3: ResNet

Fine-tuning a network means taking a pre-trained model (a model trained on a large dataset, typically on a general task like image recognition) and continuing the training on a smaller, specific dataset to tailor the model for a particular task.

We "freeze" the convolutional layers and some of the linear layers in a pre-trained ResNet to leverage the already learned features, which are useful across different visual tasks, and to prevent overfitting by only training the final layers that are crucial for the specific task at hand. This approach is efficient as it requires less computational resources and time compared to training a model from scratch.

Part 4: Logging to Weights and Biases

https://wandb.ai/viditdpokharna/dlrecog/runs/vsm6vn3c/workspace

Extra credit

https://wandb.ai/viditdpokharna/huggingface_scene_recognition/runs/pd032ofo/workspace