Deep Learning
Keshigeyan 1002327
Coding Homework #5 Report

Overview

Architecture used: ResNet18

Batch size = 64

Infrastructure: I used my own GPU (GTX 1050) to save AWS credits for projects.

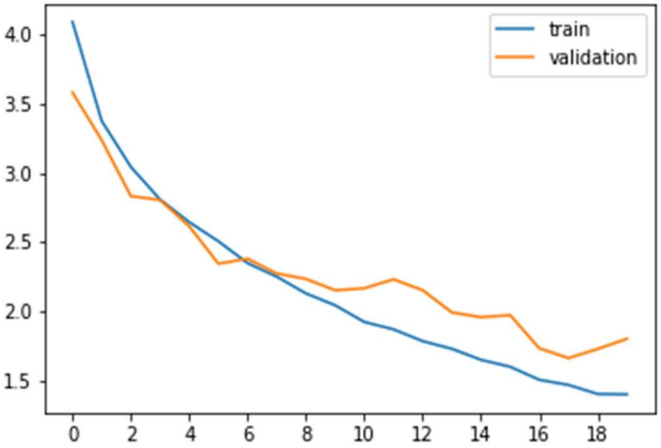
Notes: All networks were trained for 20 epochs with batch size = 64, learning rate = 1e-4 in order to keep the control variables constant.

Metrics used: Classification Accuracy = num of correctly classified images/ total number of images

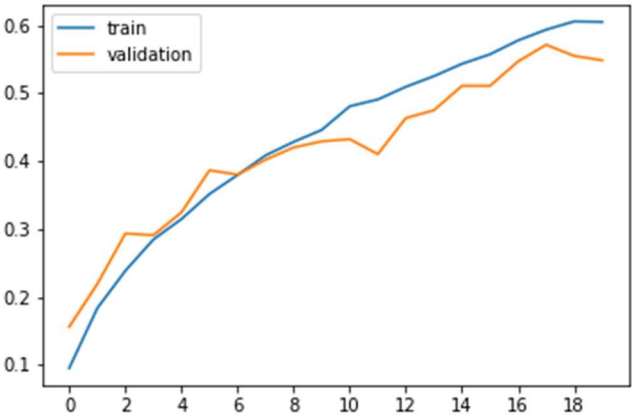
Task 1

Training the network without loading weights and training all layers

Training Loss and Validation Loss



Training Accuracy and Validation Accuracy



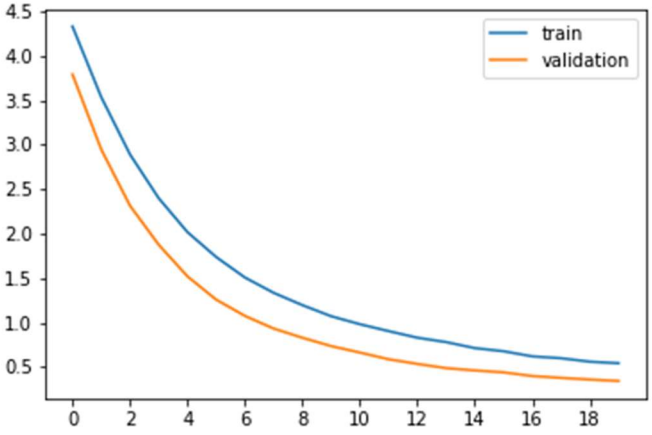
Test time loss: 1.7524

Test time accuracy: 709/1278 = 0.555

Task 2

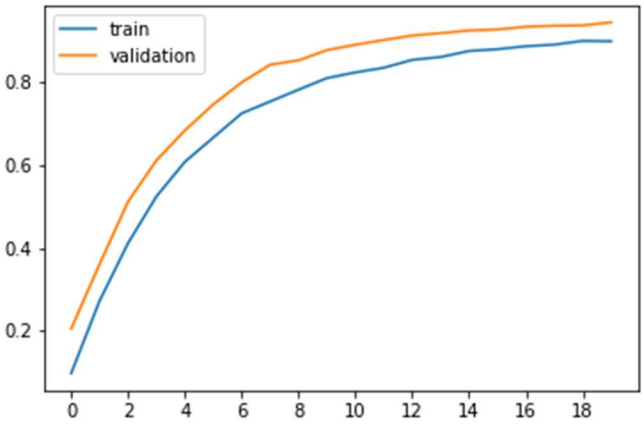
Training the network with loading weights before training and training all layers

Training Loss and Validation Loss



Deep Learning
Keshigeyan 1002327
Coding Homework #5 Report

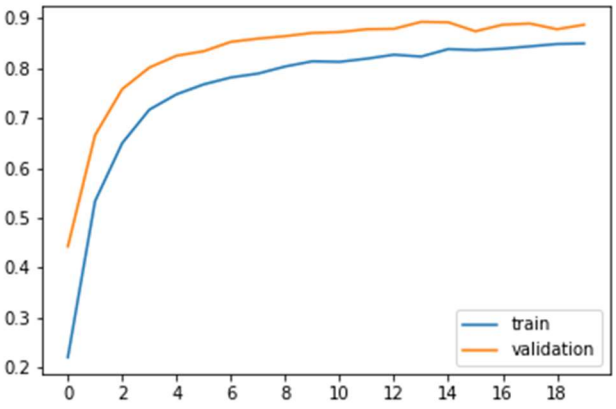
Training Accuracy and Validation Accuracy



Test time loss: 0.3631

Test time accuracy: 1185/1278 = 0.927

Training Accuracy and Validation Accuracy



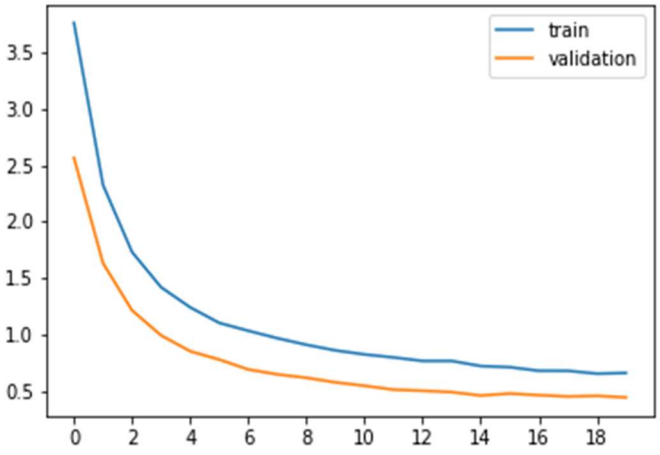
Test time loss: 0.5393

Test time accuracy: 1111/1278 = 0.869

Task 3

Training the network with loading weights before training and only training the last 2 trainable layers

Training Loss and Validation Loss



Observations

Validation accuracy

Validation accuracy trend is quite unstable in task 1 since we are learning from scratch. (Like a toddler trying to walk).

For tasks 2 and 3, the validation accuracy trend is very stable and increasing with epochs since we use transfer learning.

Test time Accuracies

The test time accuracy for task 1 is very low since we are learning from scratch (Also we run only for 20 epochs).

Test time accuracies for task 2 is the higher than task 3 since we train all the layers in task 2 whereas in task 3, we strictly retain all the low-level features and allow the network to learn only high-level features. (Do note that task 2 and 3 use pretrained weights on ImageNet)