HW2P2 Writeup

Model Architectures

The architectures that I tried out for this assignments are as below:

1. **MobileNetV2**: I tried different combinations of model depth (number of layers) and breadth (number of filters in each layer) while preserving the MobileNetV2 base architecture of using inverted residual bottlenecks. The best stats for MobileNetV2 architecture were Validation Accuracy=69.2%, Validation Loss=1.4 and Test Accuracy=60.6%.
2. **ResNet50**: Not many changes were to this architecture except reducing the stride to account for the smaller input image size for this assignment. The best stats for ResNet50 architecture were Validation Accuracy=65.4%, Validation Loss=1.6 and Test Accuracy=56.2%.

In general I found that MobileNetV2 is much faster then ResNet50 and is a more suitable architecture for this assignment. ResNet50 takes too much time to converge, and might end up giving better results if trained for a long duration. I accounted the slower convergence to the fact that the network will take time to learn that a lot of the filters are useless. I had also tried DenseNet but it was not showing a considerable improvement above MobileNetV2 or ResNet50, so I stopped the training early and discarded that architecture.

Loss Functions

I tried using cross entropy as well as cross entropy + center loss. I found that using center loss slows down the convergence and was not giving as good results as with only cross entropy.

Learning Rate and Regularization

Scheduling the learning rate was quite tricky. I tried exponentialLR, stepLR, and stepLRonPlateau from PyTorch. In my experience using exponentialLR helps. But most important step was the manually check how the loss was improving and then re-run the training from last checkpoint with LR reduced by a factor of 0.1. For example, with MobileNetV2, I found the for first three epochs with a LR = 1e-2, the accuracy would increase as 20% —> 30% —> 35%. At this point it would fluctuate around 35-40% range for several next epoch. But decaying the LR by 0.1 factor after third epoch and restarting the training from previous checkpoint would give a direct boost from 35% —> 60% accuracy.

I used weight decay and dropout. Although neither weight decay nor dropout seemed to improve or aggravate the results much.

Model Ensemble

Model ensemble is by far the most useful technique I found to improve my results. I used the “Top 3 Highest Validation Accuracy” and “Top 2 Lowest Validation Loss” for 4 different architectures, that is, three MobileNetV2 and one ResNet50. So total 20 models were used for final prediction in classification as well as verification. I found model ensemble to improve my results by 7-8% on validation set. It also helped reduce the gap between validation accuracy and test accuracy slightly (by 1%).