Report for part A

I started with implementing the architecture as-is, almost exactly the same as described in the assignment without any modification. Only additions were

- Early stopping that stops training if there is no improvement in the validation accuracy for 3 epochs.
- Model checkpoint that saves the best model so far—the one with the best validation accuracy—and loads it at the end of training.
- Also, the optimizer used was rmsprop.

This architecture was giving predictions with around 63% accuracy.

The next changes I made were

- Optimizer used: adam
- Initializing the layers with *he_normal* distribution.

This architecture was producing 67.9% accuracy.

Report for part B

Architectures tried out

- DenseNet
    - With 5 layers per densely connected block, and growth rate 12. Accuracy was around 62%
    - With 4 layers per densely connected block, and growth rate 8. Accuracy was around 64.55%
- ResNet v1, with depth 14. Accuracy was around 55.9%
- Same architecture as of part A but with added dropouts after each layer // block of layers. This was the best performing one, accuracy was around 69.32%

So the final architecture used was —

1. Convolution with 64 features, 3x3 kernel, ReLU activation, *he_normal* weights initialization.
2. Max pooling with pool size 2x2.
3. 25 percent dropout.
4. Convolution with 128 features, 3x3 kernel, ReLU activation, *he_normal* weights initialization.
5. Max pooling with pool size 2x2.
6. 25 percent dropout.
7. Flatten.
8. Dense layer with 512 nodes, ReLU activation, *he_normal* weights initialization.

9. 25 percent dropout.
10. Dense layer with 256 nodes, ReLU activation, *he_normal* weights initialization.
11. 25 percent dropout.
12. Final dense layer with 10 nodes, softmax activation, *he_normal* weights initialization. Also doing batch normalization before applying activation.

The optimizer used was adam.