The overall objective of this assignment is:
(a) getting a grasp of each hyperparameter's magnitude.
(b) devising a new neural network structure.

Neural network

Neural network feeds on 32×32 randomly cropped, flipped, and normalized images with different batch sizes for training. CNN, CNN2, and FCNN were used in this assignment. Each structure is shown as follows:

CNN

1. 3×3 2d convolution layer with padding (to preserve dimension) from 3 channels of RGB to 32 channels.
2. ReLU filter
3. 2×2 Max pool layer reducing the size to 16×16.
4. 3×3 2d convolution layer with padding (to preserve dimension) from 32 channels to 64 channels.
5. same as 2
6. same as 3. The size reduces to 8×8.
7. flattening layer which linearizes the tensor.
8. linear transformation layer from $64 \times 8 \times 8$ to 128.
9. linear transformation layer from 128 to 10, which is the number of labels in the CIFAR-10 dataset.

CNN2

As with CNN, 2d convolution layer channels have shifted to 16 channels instead of 32. This is because the overall size of tensors can grow more naturally exponentially.

FCNN

Unlike the previous two, this network flattens the images first, then applies 5 layers of linear transformation, ReLU, and dropout layer, reducing the features from $32 \times 32 \times 3 \rightarrow 2048 \rightarrow 1024 \rightarrow 512 \rightarrow 256 \rightarrow 10$. The dropout layer would remove unnecessary connections and will prevent overfitting from all neurons connecting.

Experiment 1

The first experiment tests those hyperparameters on CNN:
Batch size: 16, 32, 64, 128
Number of epochs: 5, 10, 20, 40
Learning rate: 0.0001, 0.001, 0.01, 0.1

(The Experiment data is on hypertuning-analysis.ipynb file)

The model code is written on cifar_training.py, which is executed with different hyperparameter by run_experiment.sh. The results are stored in a log file in the log folder, which is then parsed to create one pd.DataFrame for statistics. The statistics can be found on hypertuning-analysis.ipynb file.

Batch size: 16 showed the highest and most consistent accuracy.
Number of epochs: The performance peaked at 20 epochs and would reach a plateau.
Learning rate: 0.0001 and 0.001 showed best results. A learning rate of 0.001 dominated the top 4 in the entire experiment. Meanwhile, most 0.1 and 0.01 groups showed no improvements over their epochs.

The first experiment came with a few follow-up questions, which became the foundation of the next experiment.

(a) What if we tried smaller batch sizes?
(b) Can we find a middle ground between 0.0001 and 0.001? What if we tried smaller learning rates?
(c) Can other neural networks perform better?

Experiment 2

The second experiment tests those hyperparameters with the same number of 20 epochs:
Batch size: 8, 16, 32, 64
Learning rate: 0.00003, 0.0001, 0.0003, 0.001, 0.003
Model: CNN, CNN2, FCNN

The log files for this experiment are stored in log2 folder.

Batch size: 8 showed the highest and most consistent accuracy.
Learning rate: 0.0003 showed the highest average accuracy with a low standard deviation. However, the 0.001 learning rate group had the highest accuracy in this experiment, scoring 8 of the top 10 accuracies.
Model: FCNN showed 20%p less accuracy than CNN and CNN2. CNN showed approximately 0.8% higher accuracy than CNN2 on average with more consistent results.

Conclusion

- FCNN showed less accuracy than both CNN models. Since images are 2d, we might deduce that 2d convolution layers played a major role in representing image structures.
- A smaller learning rate is more optimal, offering more accurate optimization in gradient descent.
- Smaller batch sizes also resulted in more accurate predictions.