

Assignment 3

Abdelsalam ElTamawy
900170376

December 9, 2020

1 Data processing

No major data processing was used beyond resizing the images so they all are the same size. Other forms of data processing were also attempted such as adjusting saturation and adding sharpness but none proved to have a meaningful improvement on the results. As for what the size was set to upon resizing the images, 32×32 proved to give a nice medium between performance, memory consumption and accuracy.

2 Hyper parameters

As with the last assignment, All hyper parameters started with their “rule of thumb” and then dialed in to their final values.

As for the network architecture of the model, the fully connected section remained unchanged from the first assignment at 2 layers with 1200 nodes each since it proved to perform so well the last time.

After trying a handful of configuration for the convolutional layers. I ended up settling on having alternating convolving and max pooling layers. The choosing the configuration of the convolving layers centered around the philosophy of making the volume produced at each layer be smaller along the cross section and deeper, providing room to draw features from nicely.

Step sizes were kept rather small since the data already was at a small 32×32 . Max pooling layers were kept at smaller size as well, also so we don't have too few features to draw from at the fully connected layers.

After some experimentation, the convolving layers ended up having the following configuration:

1. 3×3 convolution kernel, padding 2, 4 filters, step size of 1.
2. 2×2 max pooling, step size of 2.
3. 500 node fully connected layer.
4. 500 node fully connected layer.

5. 5 node classification layer.

Now that the most critical hyper parameter is settled, we can now tune the other hyper parameters. Similar to last time, I defined a set of limits at which each parameter would give NaN results and then logarithmically randomly generate values for each hyper parameter on this scale.

Then to dial it in further, we randomly generate all parameters according to this smaller region recommended to us by the previous set of graphs.

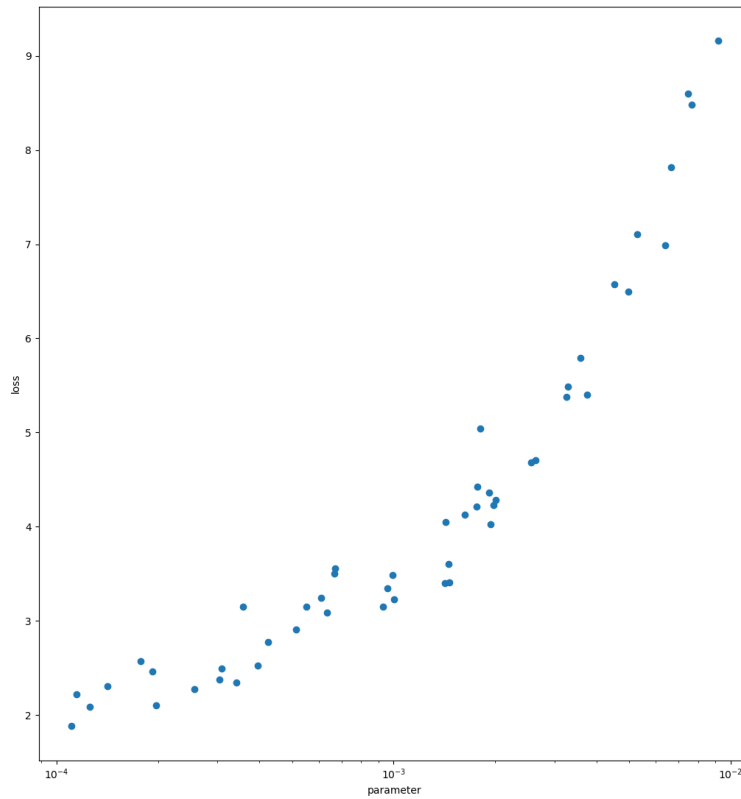


Figure 1: Tuning for regularization

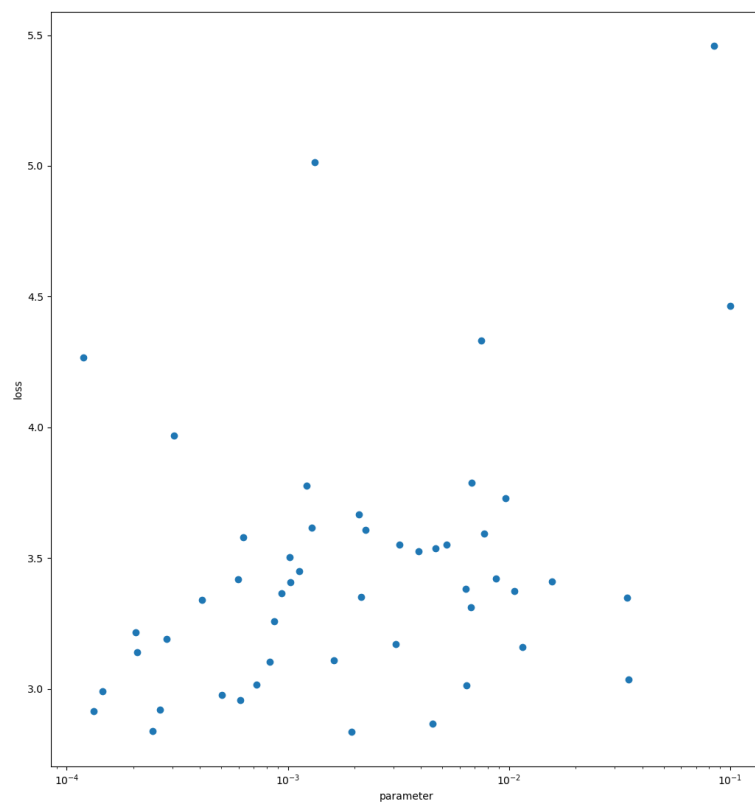


Figure 2: Tuning for leaky RELU factor

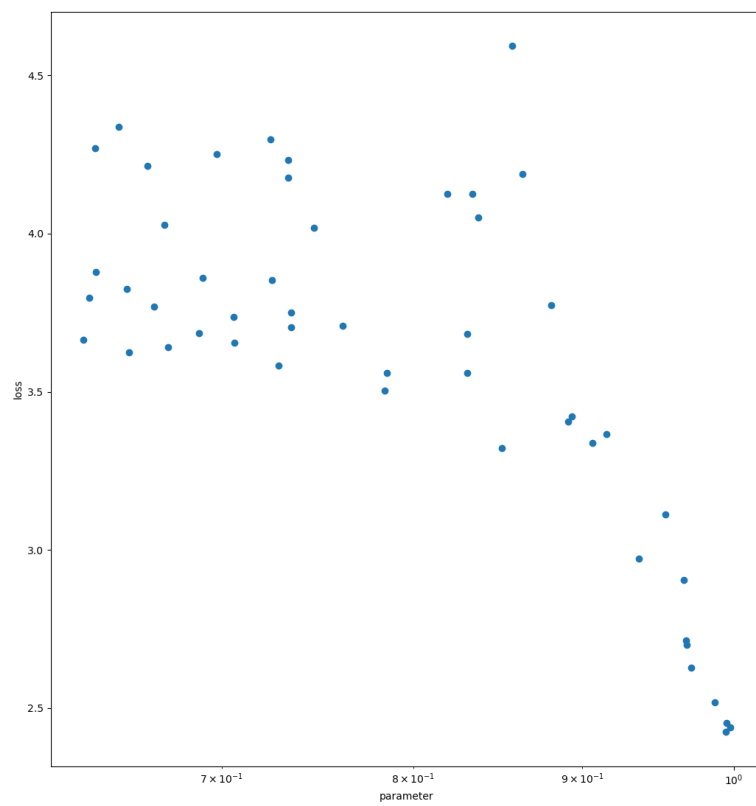


Figure 3: Tuning for momentum

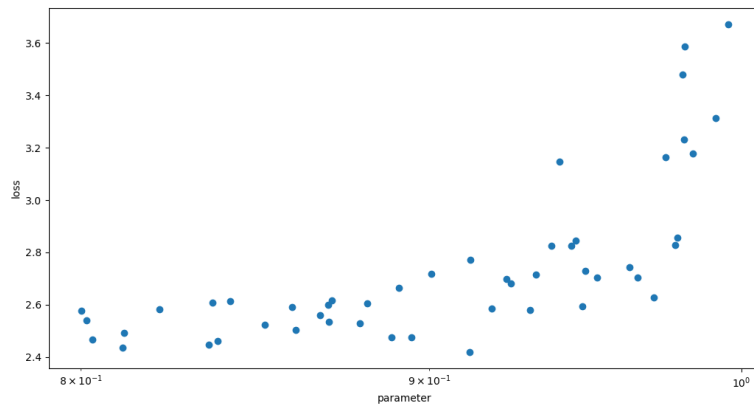


Figure 4: Tuning for accumulator

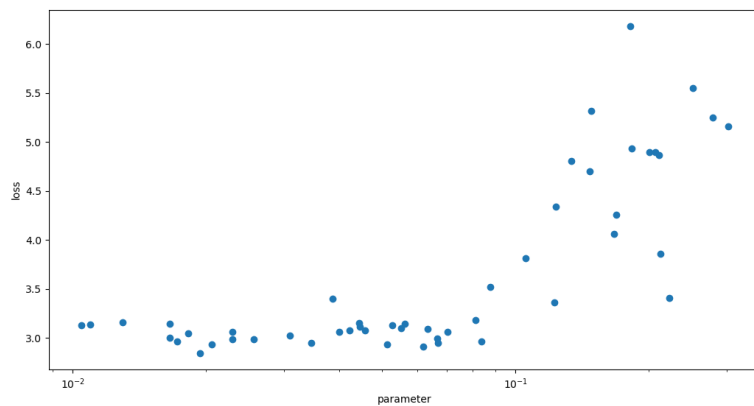
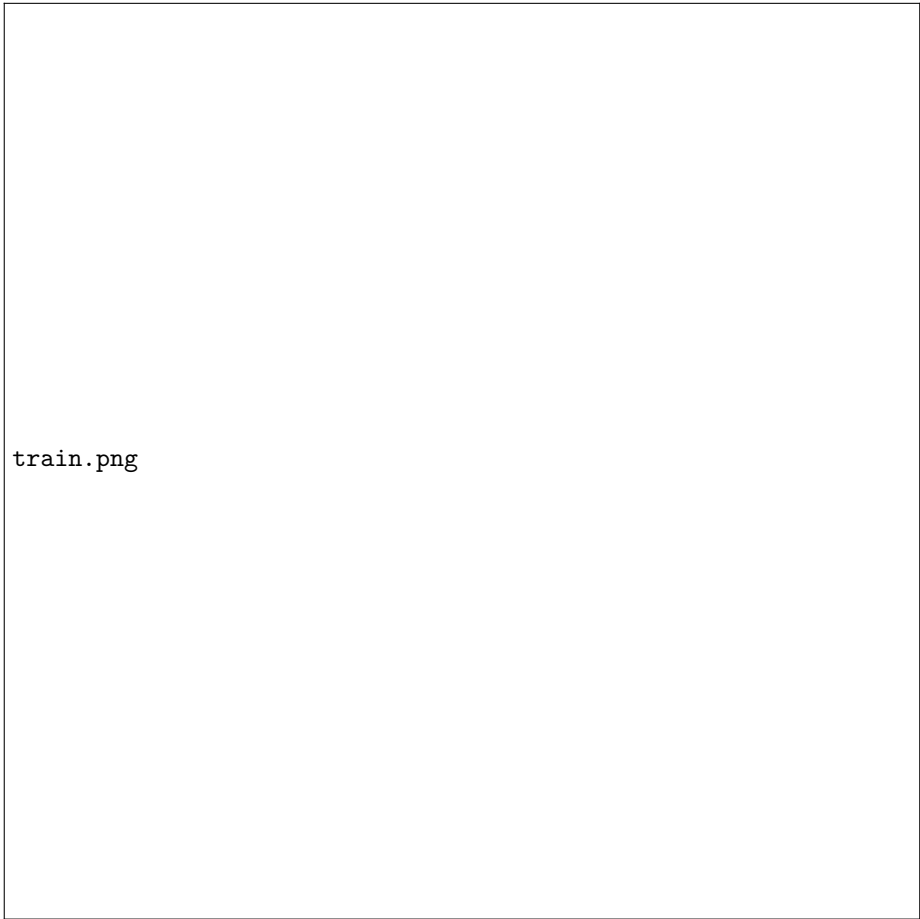


Figure 5: Tuning for drop rate

3 Training and losses

Seed used is 42.



train.png

4 Comparison

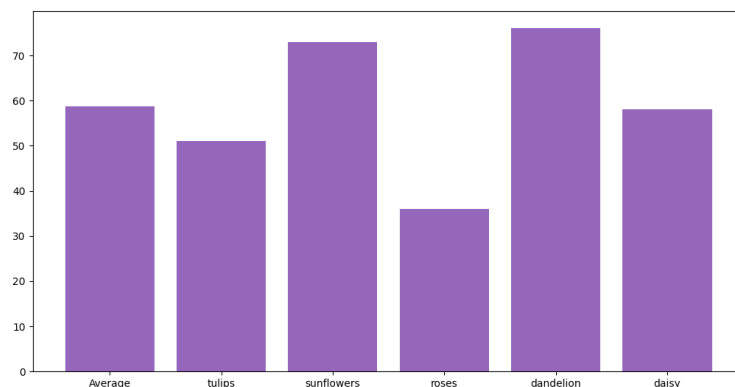
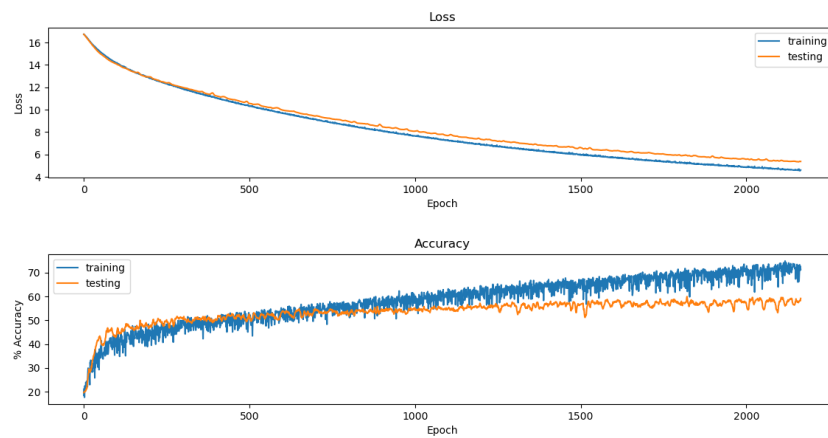


Figure 6: Per class results of training

The jump from KNN to NN was for the most part a direct almost constant increase across the board for the accuracy of all flowers. CNN on the other hand, seems to sort of “normalize” the results more, as in there is less of a direct constant relation from the results from KNN and NN to CNN, the lower scoring flowers had a marked increase. This suggests there being more “reasoning” happening behind CNN as it really is with it’s convolutions.

5 ACCR

A final ACCR of 60.2% after almost 3000 epochs.



Stopped at that point since the accuracy seems to level out and due to time constants.