CS-233: Introduction to Machine Learning Course Project Milestone 2 Report

Introduction

In this second report, we implemented and evaluated two deep learning methods, Multilayer Perceptron (MLP) and Convolutional Neural Network (CNN), on the DermaMNIST dataset. This time the database consists of dermoscopic images of skin lesions across seven different diagnostic categories. For this milestone we will compare the performance of these architectures in terms of accuracy and F1-Score.

Method

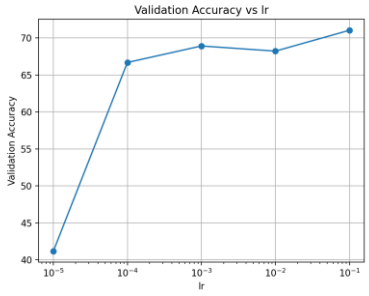
First, let's talk about data preparation. We used the provided data split with 7007 training images and 2005 test images. A validation set was also created by splitting 20% of the training data. All images were normalized based on the training set mean and standard deviation. For the MLP, images were flattened into vectors while the CNN received 3-channel images with spatial dimensions intact.

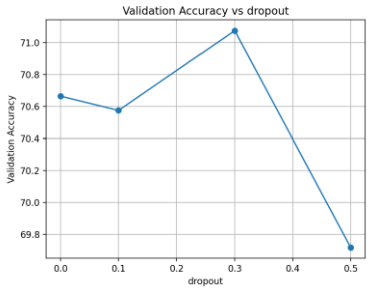
Secondly, we had to choose a certain architecture for each model. For MLP we chose three hidden layers (512, 256, 128 units) with ReLU activations and dropout layers applied after each hidden layer and for CNN we have three convolutional layers (channels 16, 32, 64) each followed by max-pooling and dropout and two fully connected layers at the end. This is the architecture and activation function that we started with. We then did our hyperparameter tuning to find the optimal values and afterwards played around with different architectures to see if we could optimize the model even more.

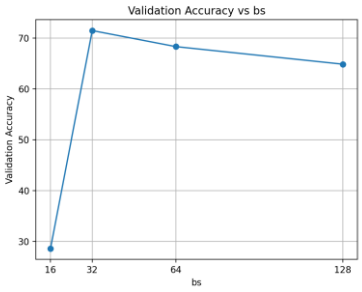
Lastly, for hyperparameter tuning, we decided to run k-fold cross validation on the training data just like in the first report, with k = 5. We ran it to find the optimal learning rate, dropout and batch size, on predefined ranges: learning rates from 1e-5 to 1e-1, dropout from 0 to 0.5 and batch sizes from 16 to 128. We also heavily regretted doing this since it took a long time to run every single time.

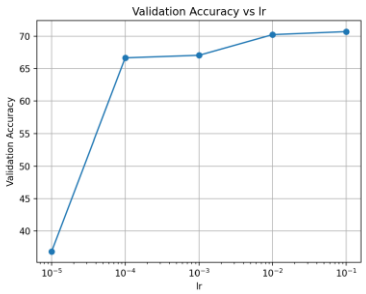
Experiment

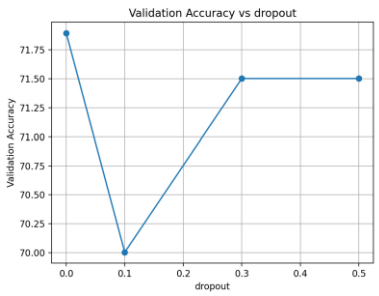
Right under this you can see the computed graphs that showcase what 5-fold cross-validation found as the optimal hyperparameter values. The first row of graphs are the results we had for MLP and the second for CNN. We didn't expect 1e-1 to be the optimal learning rate for both models as it is extremely fast and we thought it would be too likely to overfit the model. We also didn't expect the best dropout rate to be zero for CNN. Since the model is overfitted we felt like dropout would help but it seems like it did not particularly help. And finally for the batch size, we weren't too sure what to expect but for MLP it determined that the best batch size would be 32 and 128 for CNN.

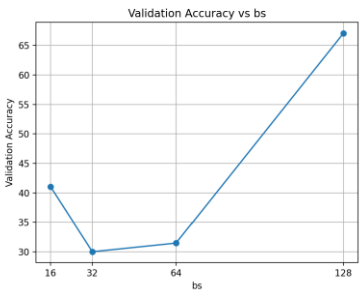












Now that we found the hyperparameters that should be optimal, we tried different architectures for the network to see how the accuracy and F1 values would fluctuate. We put all of them in this table

Architecture	**Hidden Layers**	**Accuracy**	**Macro-F1**	**Architecture**	**Filters**	**Accuracy**	**Macro-F1**
MLP_Small	256–128–64	73.466%	0.4491	CNN_Small	8–16–32	74.251%	0.5061
MLP_Baseline	512–256–128	74.465%	0.4502	CNN_Baseline	16–32–64	74.465%	0.4716
MLP_Wide	1024–512–256	73.110%	0.4444	CNN_Wide	32–64–128	75.963%	0.5196
MLP_Deep	512–512–256–128	75.535%	0.4832	CNN_Deep	16–32–64–128	75.036%	0.5075

It seems like the difference in the architecture don't change the accuracies by a huge margin, but there are still a few changes that help us determine what dimensions are optimal, here it is the dimensions of MLP_Deep and CNN_wide.

Results

Now that we have played around with the hyperparameters and the architecture of the models to find the best accuracy and F1 value on the training data, let's test it out properly. We will also test how long it takes to train both models.

Model	**Architecture (Dims)**	**Dropout**	**Learning Rate**	**Batch Size**	**Train Acc**	**Train F1**	**Test Acc**	**Test F1**	**Train Time (s)**	**Inf Time (s)**
MLP	512–512–256–128	0.30	1e-1	32	95.861%	0.935612	72.718%	0.505187	**277.56**	**0.25**
CNN	32–64–128	0.00	1e-1	128	100.000%	1.000000	74.514%	0.528290	**453.95**	**0.52**

Conclusion

The result table is strongly indicative that both models are overfit. We can see that the training accuracy is a lot more important than the testing accuracy and the same thing can be noticed from the F1 score. It is also noticeable that both models have their perks. The MLP is computed a lot faster (half the inference time) than the CNN but the later is a tiny bit more accurate. We found that a few things were surprising such as 1e-1 being the best performing learning rate and 0 dropout having the best results for CNN. We think that those are both red flags that prove the models can still be greatly improved. Otherwise, this has been a great learning experience, and it was really fun to implement the models of the course on different data sets in MS1 and MS2.