IN4310 mandatory 1

Erlend Kristensen

16.03.2022

1 Task 1

For how to compile the code, please see the readme.txt file. The model was initialized using pytorch resnet18 model with pretrained weights. I used 20 epochs and a learning rate of 0.001 with the SGD optimizer. These hyperparameters were found after doing a few tests, with different learning rates and optimizers like ADAM.

This is the resulting accuracy after training on the entire dataset:

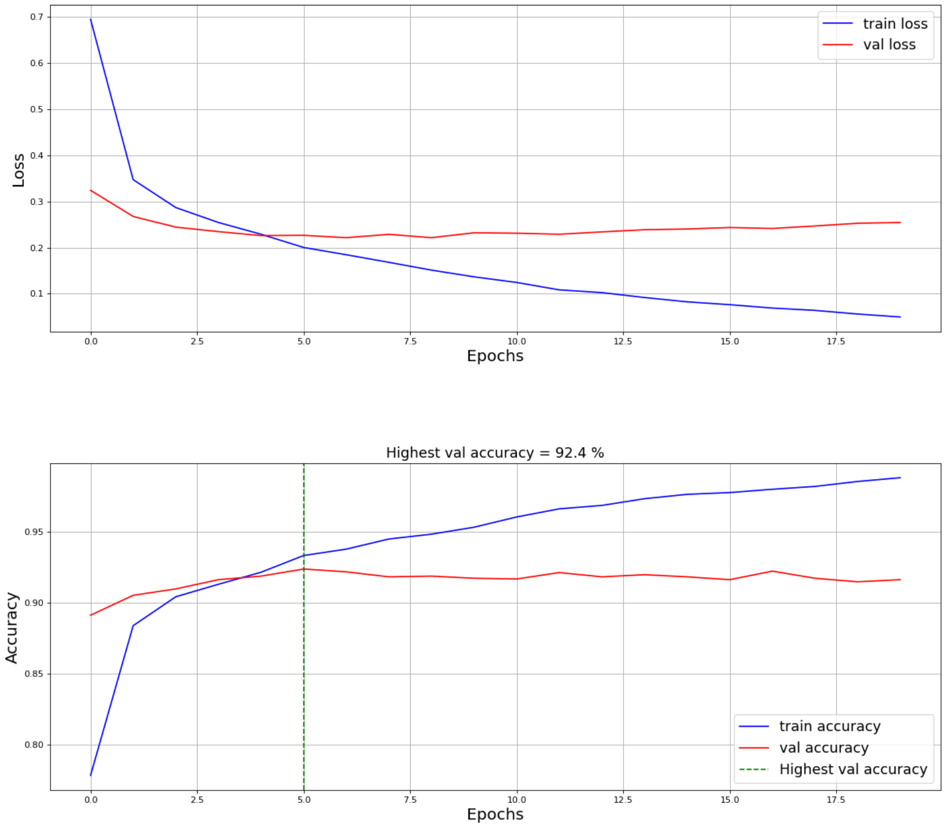

Figure 1: Accuracy and loss plots for training and validation set.

These plots show how the training accuracy go to over 95%, and the validation accuracy caps at around 92%, which is very good.

Here are the following class wise mean accuracies: For training data:

Accuracy of buildings : 99.2%

Accuracy of forest : 99.5%

Accuracy of glacier : 97.4%

Accuracy of mountain : 98.9%

Accuracy of sea : 98.7%

Accuracy of street : 99.4%

For validation data:

Accuracy of buildings : 94.2%

Accuracy of forest : 99.4%

Accuracy of glacier : 86.5%

Accuracy of mountain : 87.6%

Accuracy of sea : 96.3%

Accuracy of street : 91.4%

These results are very promising. It seems for training data that all of the images are easily classified, while for validation data, glacier and mountain are harder to separate.

We will then take a look at the top 10 and bottom 10 sorted images, to see how our model does:

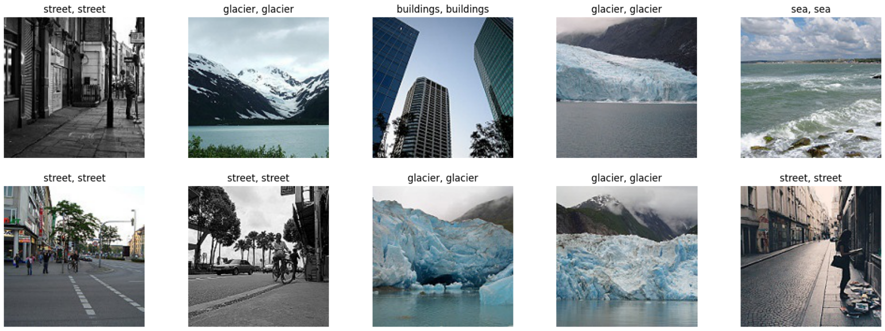

Figure 2: Top 10 images that our model was most certain about.

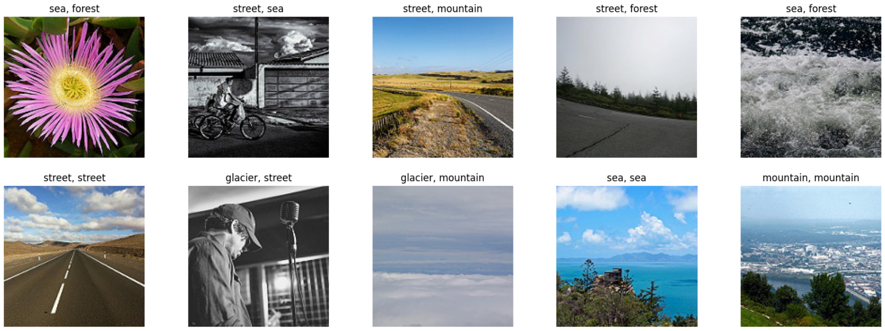

Figure 3: Bottom 10 images that our model was least certain about.

(Note, i tried to make it for each class as can be seen with some example code i added in mandatory1_ task1.py, but it did not work, so i added top 10 and bottom 10 for entire model.).

On the title of each image, the label on the left in the correct label and the label on the right is the predicted one.

Top 10 is not the most interesting to look at. It is clear what each image is, which is why it is in top 10. It is interesting however, how most of the images are street and glacier, even though some of the images that the model struggled with was in fact glaciers (as seen with the validation accuracy).

In bottom 10 however, we see for example the top left image being classified as forest, while the true classification says sea. Seeing as it is an image of a flower, it is clear that the sea label was wrong, and our model managed to predict correctly. The other images show how the model can be uncertain, for example with top middle being street, but the model classifies as mountain since most of the image is in fact covered in mountains.

Now that our model is trained, we do an extra test on a test set, and get the following results:

Accuracy of buildings : 94.8%

Accuracy of forest : 99.4%

Accuracy of glacier : 85.0%

Accuracy of mountain : 90.1%

Accuracy of sea : 95.7%

Accuracy of street : 93.1%

Accuracy of all combined : 93.0%

Average precision of buildings : 89.3%

Average precision of forest : 98.4%

Average precision of glacier : 77.1%

Average precision of mountain : 79.8%

Average precision of sea : 94.5%

Average precision of street : 88.7%

An overall accuracy of 93% and a relatively high precision rating on all the different classes is very good. The test data shows us again how the model seems to have trouble classifying glacier and mountain correctly, but seeing as the two are similar, it could be that the model struggles differentiating between the two.

2 Task 2

The modules I chose was relu, maxpool, avgpool, along with layers 1 and 4 in the model. The percentage of non-positive values were as follows:
relu: 0.281
maxpool: 0.231
layer1: 0.237
layer4: 0.526
avgpool: 0.025

For the Cifar-10 dataset i got: relu: 0.267
maxpool: 0.253 layer1: 0.260
layer4: 0.558
avgpool: 0.036

And for imagenet dataset i got:
relu : 0.560
maxpool: 0.360
layer1 : 0.457
layer4 : 0.745
avgpool: 0.374

3 Task 3

For task 3 i chose to plot the 64 largest eigenvalues, seeing as some of the modules i chose only had 64 channels. Here are the resulting eigenvalues for the mandatory dataset, cifar-10 and imagenet:

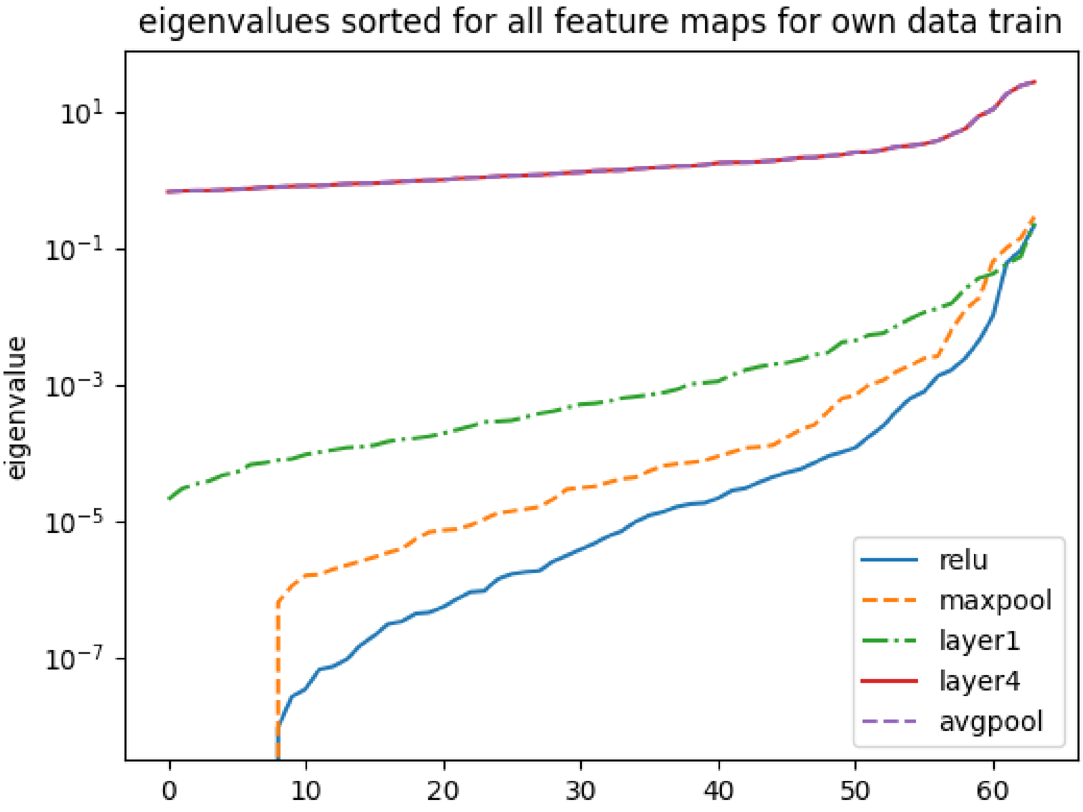

Figure 4: 64 Largest eigenvalues found using the modules specified above and on the mandatory dataset using the train data. Images upscaled to 224×224.

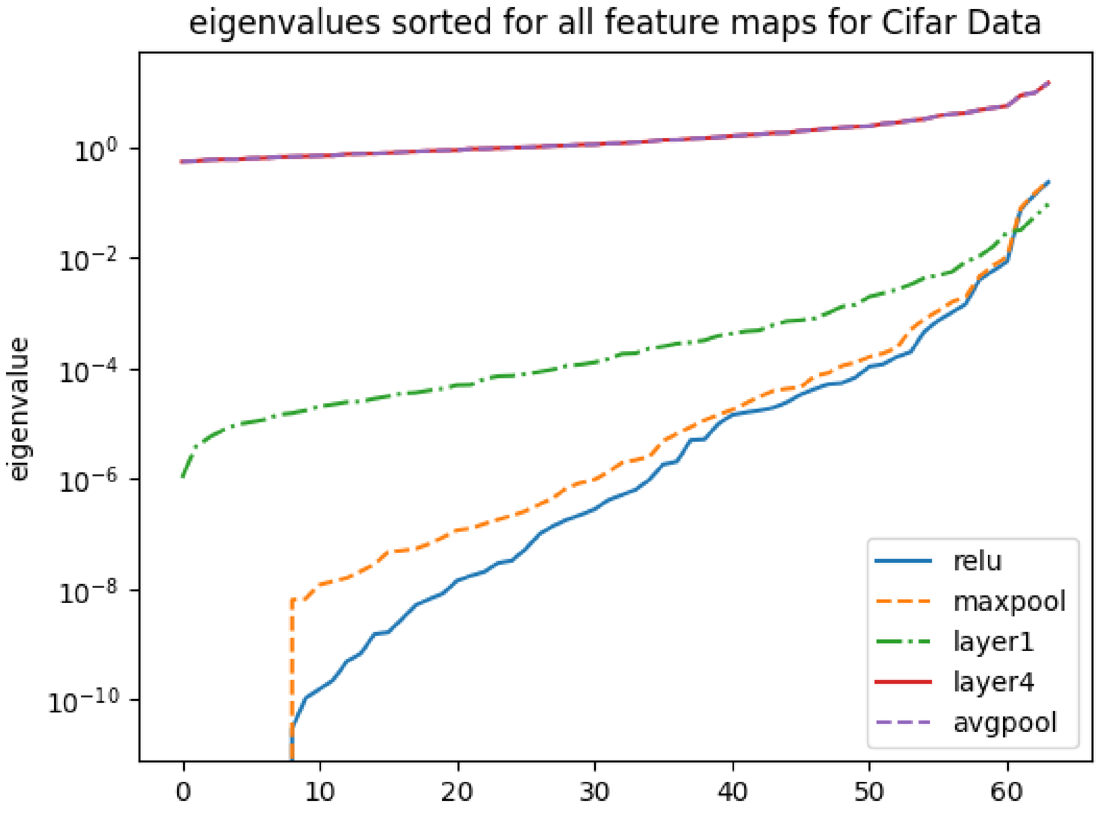

Figure 5: 64 Largest eigenvalues found using the modules specified above and on the cifar-10 dataset. Images upscaled to 224 × 224.

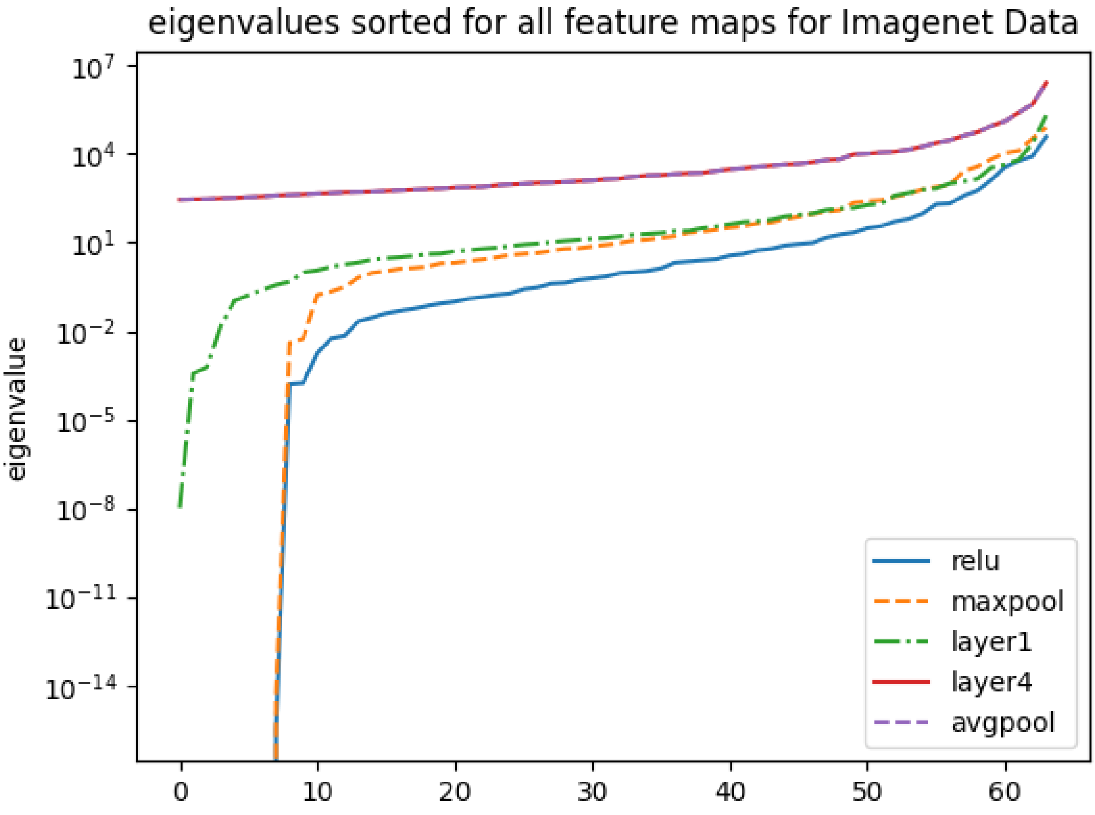

Figure 6: 64 Largest eigenvalues found using the modules specified above and on the imagenet. Images upscaled to 224 × 224.

The figures show that the eigenvalues on the Imagenet vary a lot more than the mandatory data set and Cifar-10 (which varies the least). The first layer, relu, is always the layer containing the smallest eigenvalues, whule avgpool and layer4, being the last layers, contain the largest eigenvalues and also the least amount of variety among the eigenvalues.