Clab-2 Report

ENGN6528

Wangshu Cai

U7546753

20/04/2023

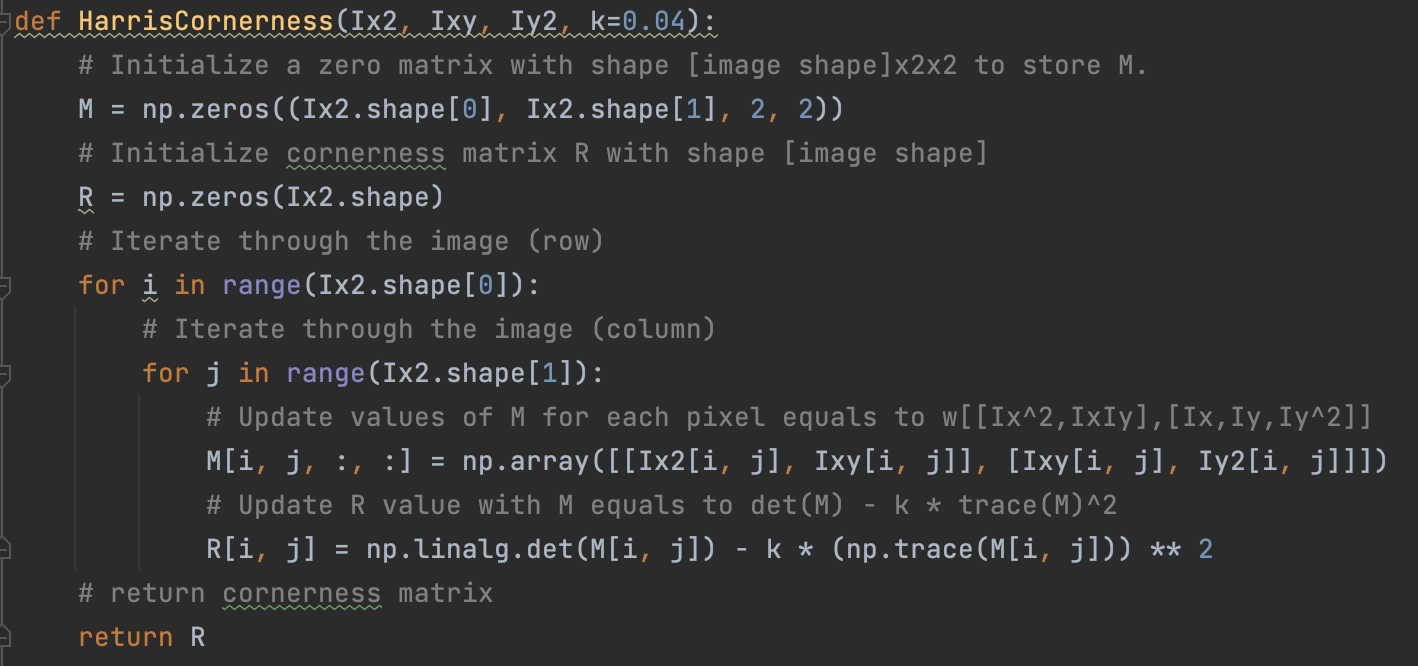
# This is just a template for CLab-2. Please make sure you follow all the requirements in Clab-2 assignment file if it is not mentioned in this template. #

# You may need to refer to the assignment file for detailed requirements

**Task 1: Harris Corner Detector (6 marks)**

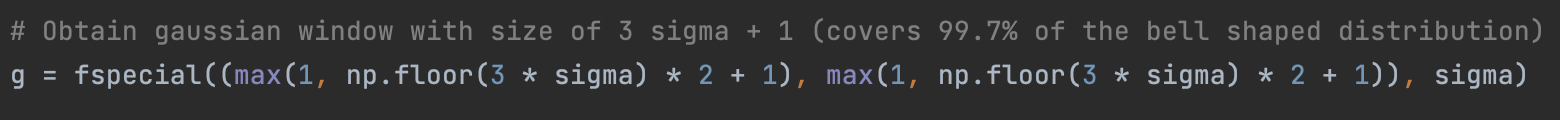
1. **Read and understand the corner detection code ‘harris.m’ in Fig 1.**
2. **Complete the missing parts, rewrite ‘harris.m’ (or harris.py) into a Matlab (or Python) function, and design appropriate function signature (1 mark).**

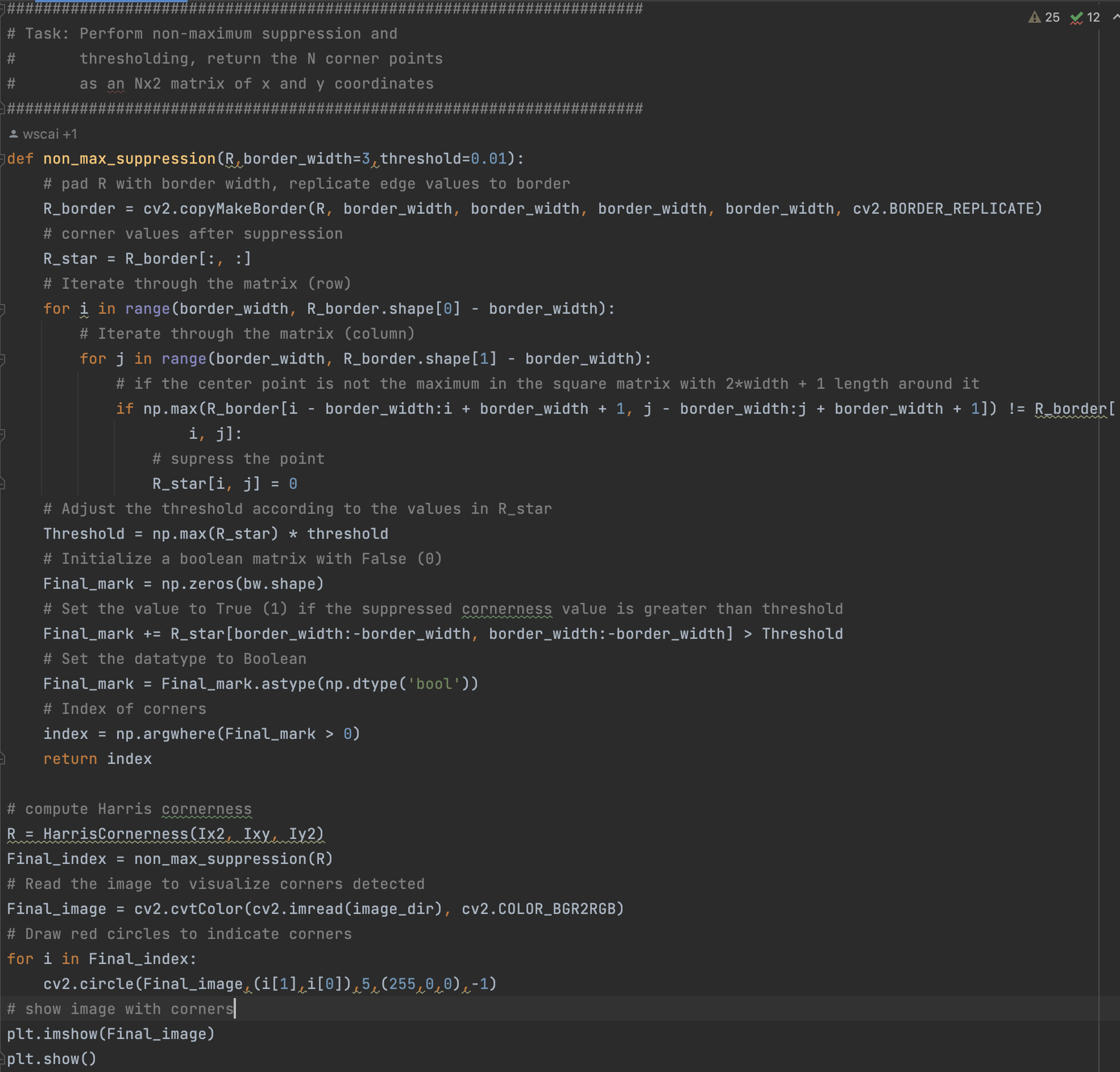
# Complete the Harris corner detection function and show a screenshot of your function here.



|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong/inappropriate codes of Harris corner detector | -1.0 |
| Inappropriate function signature | -0.2 |

1. **Please provide comments on line … and every line of your solution after line … (0.5 mark). Specifically, you need to provide short comments on your code, which should make your code readable.**

****

****

# Provide a copy or screenshot of your codes with clear and short comments here.

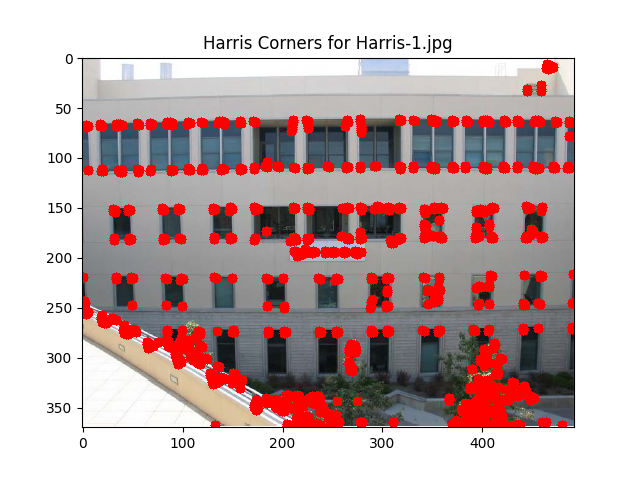
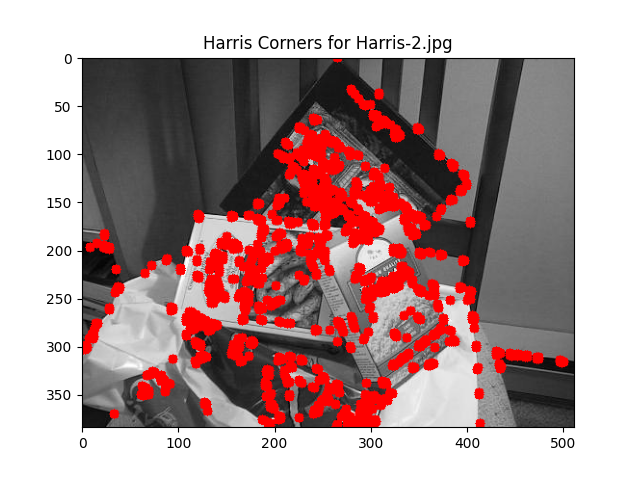
# If you already show your code including comments in Task1.2, then you do not need to show anything here. Just simply say something like “Code and comments are included in Figure 1.”

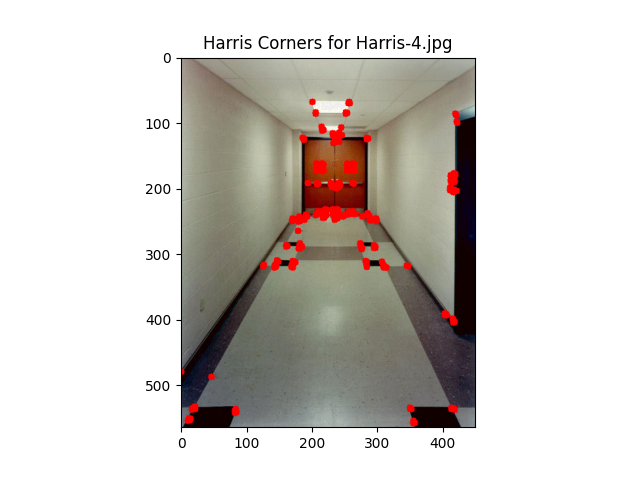
|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong comments on line … | -0.2 |
| Partial wrong comments on your codes | -0.2 |
| Inappropriate comments (e.g. too long) | -0.3 |
| Wrong comments or no comments provided | -0.5 |

1. **Test this function on the provided four test images (Harris-[1,2,3,4].jpg, they can be downloaded from Wattle). Display your results by marking the detected corners on the input images (using circles or crosses, etc) (0.5 mark for each image, (2 marks) in total).**

**Please make sure that your code can be run successfully on a local machine and generate results. If your submitted code cannot replicate your results, you may need to explain and demonstrate the results in person to tutors.**

# Display the four annotated output images with appropriate titles (0.5 mark per image)

**** ****

**** ****

**Example:**

|  |
| --- |
|  |

|  |
| --- |
|  |
|  |



Figure 1: Detected corners of “example.jpg”

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong results | -0.5 for each image |
| No annotation | -0.5 for each image |
| Cannot reproduce your results by your codes | -2.0 |

1. **Compare your results with that from built-in function used to detect corners (0.5 mark), and discuss the factors that affect the performance of Harris corner detection (1 mark) .**

**In your Lab Report, you need to list your complete source code with detailed comments and show corner detection results and their comparisons for each of the test images.**

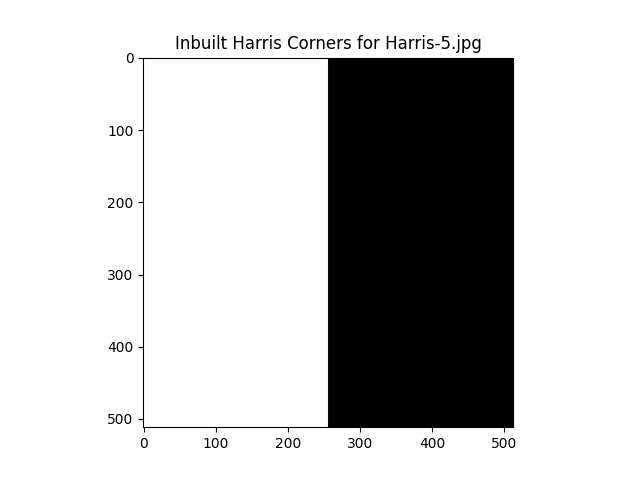
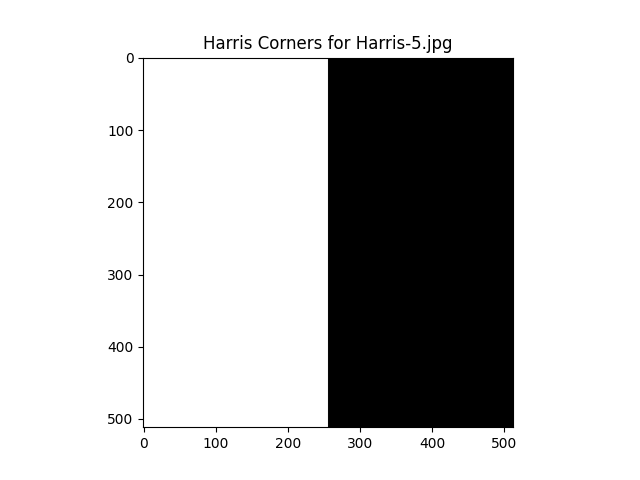
# Display the four annotated images with corners detected using Matlab’s built-in function (0.125 mark per image)

# Discuss at least two factors that affect the performances of Harris corner detection (1.0 mark)

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong images or no annotations | -0.125 for each image |
| Partially wrong discussion | -0.5 |
| Wrong discussion or no discussion provided. | -1.0 |

1. **Test your built function on ‘Harris-5.jpg’ (Harris-5.jpg can be downloaded from wattle.). Analyse the results why we cannot get corners by discussing and visualising your corner response scores of the image. (0.5mark)**

# Display the annotated image with corners detected using your function (0.1 marks)

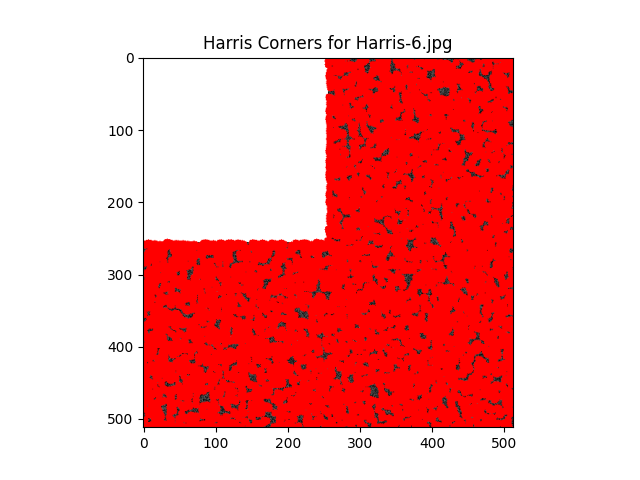
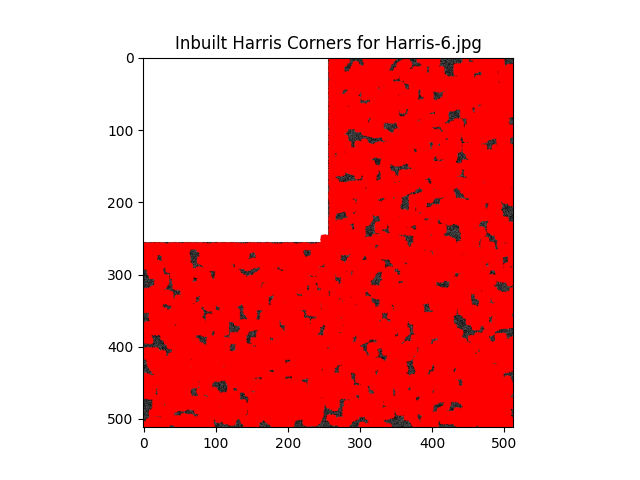


# Analyse the results why we cannot get corners by discussing and visualising your corner response scores of the image (0.4 marks)

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong images or no annotations | -0.1 |
| Partially wrong discussion | -0.2 |
| Discussion not based on the result (ex: corner response scores, etc.) | -0.2 |
| Wrong discussion or no discussion provided. | -0.4 |

1. **Test your built function on ‘Harris-6.jpg’ (Harris-6.jpg can be downloaded from wattle.). Plot your harris corner detector results in your report and propose a solution to obtain the salient corners which is robust to noise. (0.5mark)**

# Display the annotated image with corners detected using your function (0.1 marks)

# Propose a solution to obtain the salient corners which is robust to noise (0.4 marks)

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong images or no annotations | -0.1 |
| Partially wrong solution | -0.2 |
| Incorrect solution or no solution provided. | -0.4 |

**Task 2 - Deep Learning Classification (10 marks)**

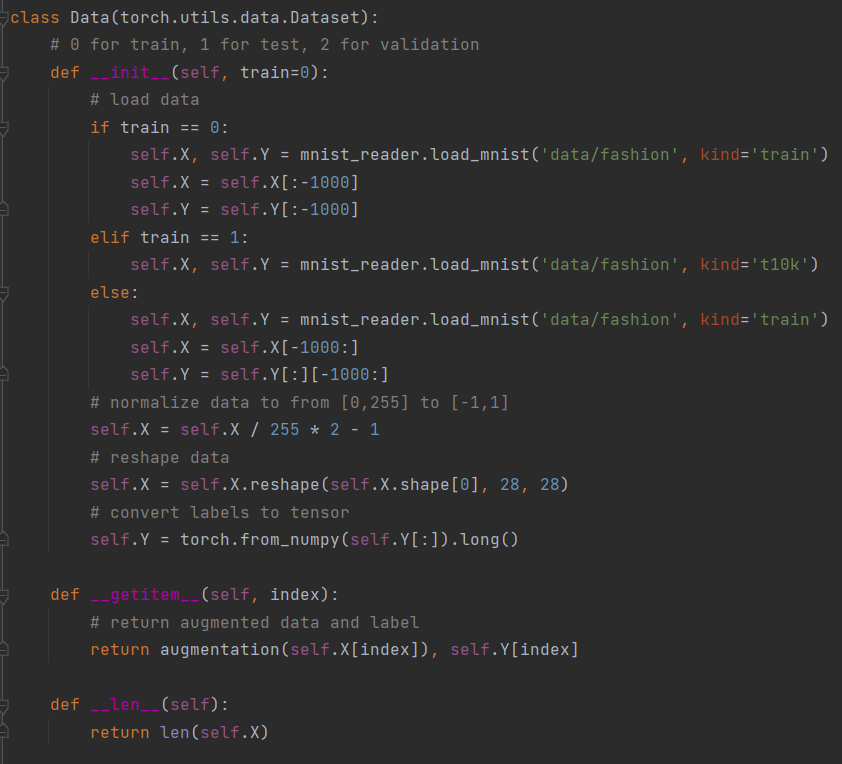
**In this lab, we will train a CNN with the Fashion-MNIST using the PyTorch1 deep learning frame- work. The Fashion-MNIST dataset contains 70000 images: 60000 training images and validation images, 10000 testing images2 for fashion classification task. Note that we have 60000 training images with labels. You are required to split the 60k training data further by yourself to reserve 1000 images for validation (they should be from 10 classes). Images are of size 28×28 Greyscale, for 10 classes:**

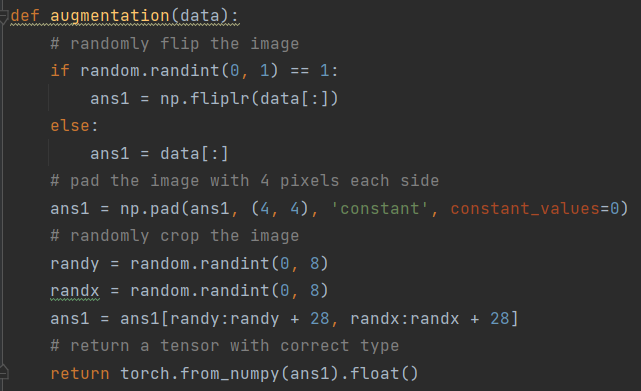
**Table

Description automatically generated**

1. **Download the Fashion-MNIST dataset from the following** [**git link**](https://github.com/zalandoresearch/fashion-mnist/tree/master/data/fashion)
2. **After loading the data using numpy, normalize the data to the range between (-1, 1). Also perform the following data augmentation when training (1 mark):** 
   * **randomly flip the image left and right.**
   * **zero-pad 4 pixels on each side of the input image and randomly crop 28x28 as input to the network.**

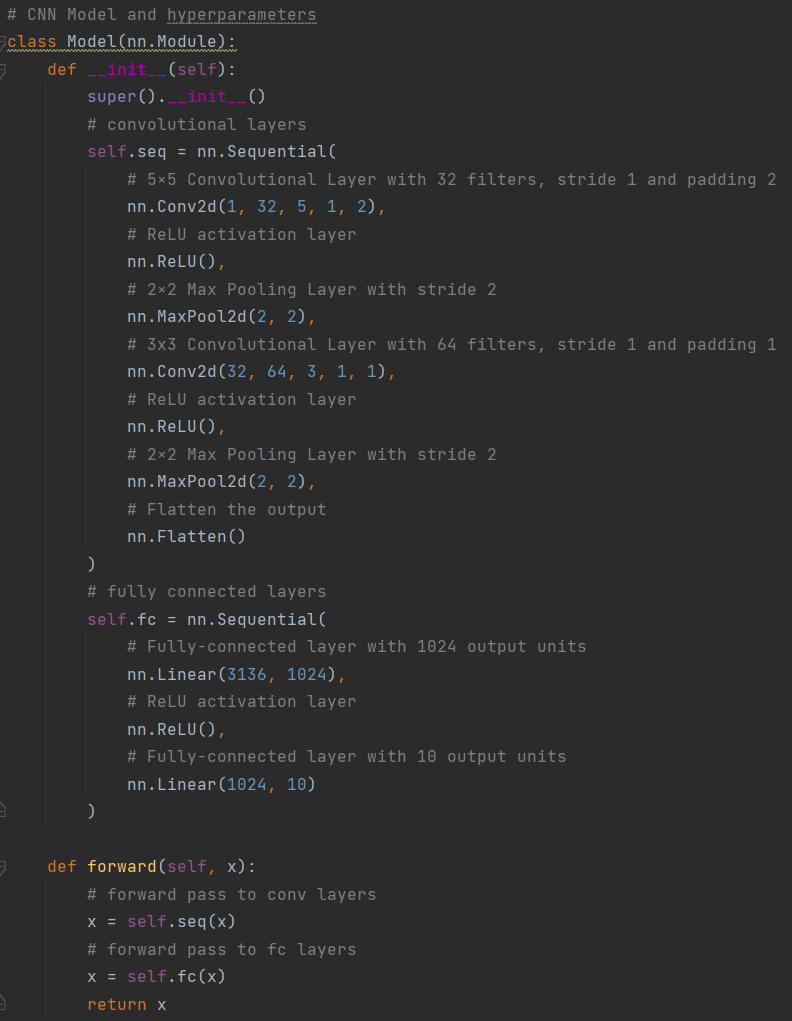
# Display a screenshot of your codes with short commends here #

****

****

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Incorrect image normalization | -0.25 |
| Incorrect image flipping | -0.25 |
| Incorrect image padding | -0.25 |
| Incorrect image cropping | -0.25 |

1. **Build a CNN with the following architecture (1 mark):** 
   * **5×5 Convolutional Layer with 32 filters, stride 1 and padding 2.**
   * **ReLU Activation Layer.**
   * **2×2 Max Pooling Layer with a stride of 2.**
   * **3×3 Convolutional Layer with 64 filters, stride 1 and padding 1.**
   * **ReLU Activation Layer.**
   * **2×2 Max Pooling Layer with a stride of 2.**
   * **Fully-connected layer with 1024 output units.**
   * **ReLU Activation Layer.**
   * **Fully-connected layer with 10 output units.**

****

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Incorrect implementation | -0.1 per component (9 components in total) |
| No comment provided for the code | -0.1 |

1. **Set up cross-entropy loss. (0.5 mark)**

# Show your codes here #



|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong loss function | -0.5 |

1. **Set up Adam optimizer, with 1e-3 learning rate and betas=(0.9, 0.999). (0.5 mark)**

# Show your codes here #



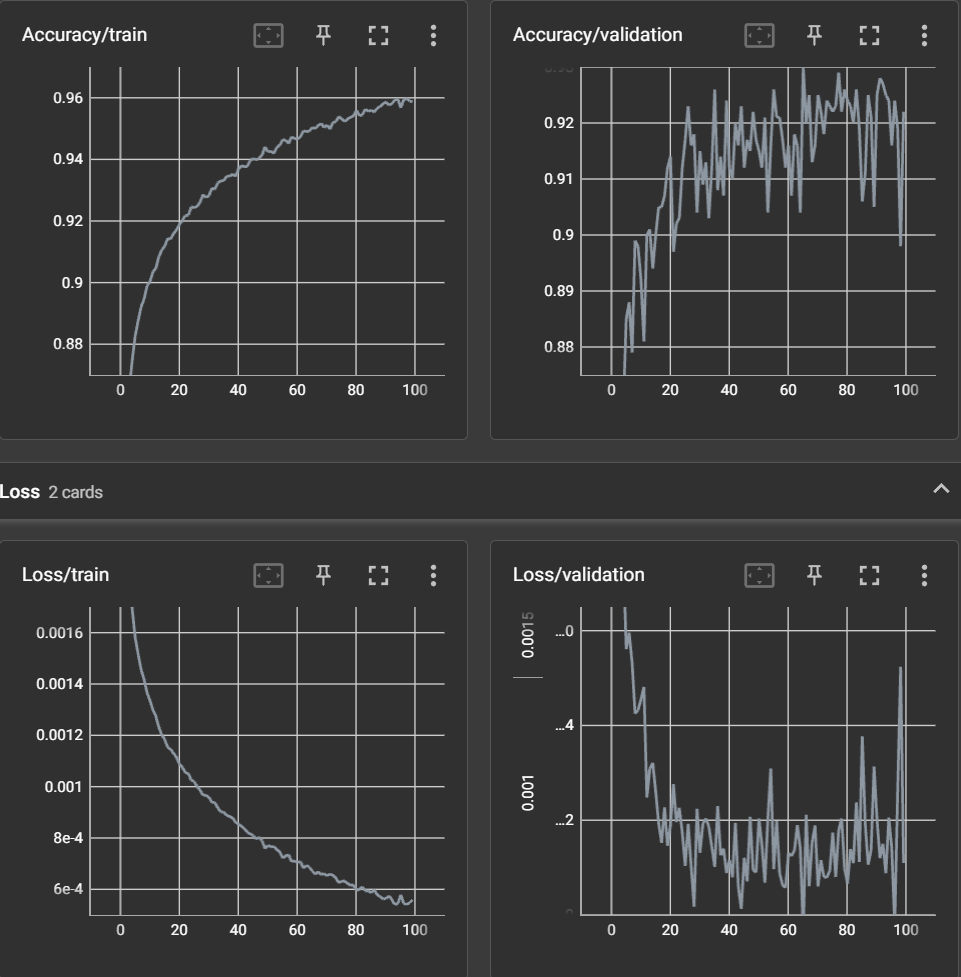


|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong optimizer | -0.5 |
| Correct optimizer but wrong parameters | -0.1 |

1. **Train your model. Draw the following plots:** 
   * **Training loss vs. epochs.**
   * **Training accuracy vs. epochs.**
   * **Validation loss vs. epochs.**
   * **Validation accuracy vs. epochs.**

**You can either use Tensorboard to draw the plots or you can save the data (e.g. in a dictio- nary) then use Matplotlib to plot the curve. (2 marks)**

**#** Display above four plots 0.5 mark each, 2 in total.



Epoch 100, training loss 0.0005587212414938515, training accuracy 0.9586779661016949, validation loss 0.0011096658408641816, validation accuracy 0.922.

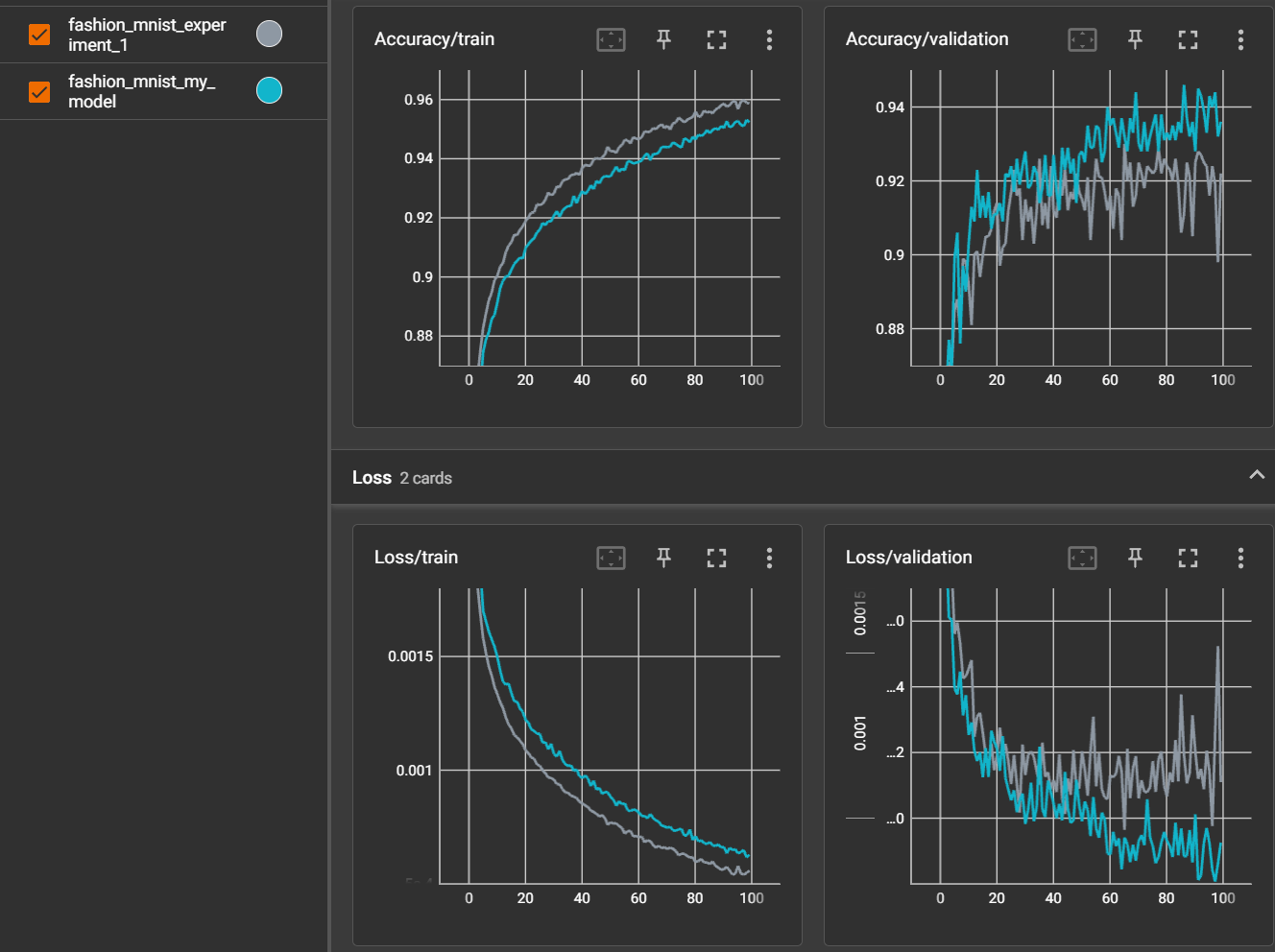
Finished training in 393.17488503456116 seconds, 3.9317488503456115 seconds per epoch

Test loss 0.001374639219045639, test accuracy: 0.9173

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Wrong plot or no plot provided | -0.5 for each plot |

1. **Train a good model. Marks will be awarded for high performance and high efficiency in training time and parameters (there may be a trade-off), good design, and your discussion. You are not allowed to use a pre-trained model, you should train the model yourself. You need to describe what exactly you did to the base model to improve your results in your report, and your motivation for your approach (no more that 1 page of text). Please include plots as above for training and validation loss and accuracy vs. epochs, as well as the final accuracy on the test set. Please submit the code and your trained model for this. Your performance will be verified. Please ensure you follow the test/train splits as provided. You also must show your training and validation accuracy vs. epochs for this model in your report. Note that you may be asked to run training of your model to demonstrate its training and performance. (4 marks)**

# Display the accuracy of your modified model with reasonable interpretations/ motivations and four plots as required (3 mark)



Epoch 100, training loss 0.0006281563676262306, training accuracy 0.9523389830508474, validation loss 0.0009255427569150925, validation accuracy 0.936, LR [0.0004717333229069918].

Finished training in 147.66043949127197 seconds, 1.4766043949127197 seconds per epoch

Test loss 0.0011745612069964408, test accuracy: 0.9294

|  |  |  |
| --- | --- | --- |
|  | Original Model | My Model |
| Training Accuracy | 0.958678 | 0.952339 |
| Training Loss | 0.000559 | 0.000628 |
| Validation Accuracy | 0.922000 | 0.936000 |
| Validation Loss | 0.001110 | 0.000925 |
| Test Accuracy | 0.917300 | 0.929400 |
| Test Loss | 0.001375 | 0.001175 |
| #Trainable Parameters | 3241866 | 1176394 |
| Training time (s) | 393.17 | 147.66 |

According to the performance data obtained from last problem, we can observe great difference between training and testing loss. The difference is caused by overfitting as the parameters fit to the training data instead of the real pattern behind. Also, the accuracy indicates that the model can be further improved to achieve higher according to the benchmark mentioned in task 8.

* + To alleviate overfitting, batch normalization and dropout layers are added in the network while keeping the total number of parameters unchanged.
  + To achieve higher accuracy, the depth of the neural network is increased to make the model more powerful. Also, adapting learning rate scheduler is used to make each backward propagation step smaller as the epoch increase so that the performance could improve.

We can observe that despite the increase of training accuracy, the difference between training and testing loss decreases, which is expected as several methods are used against overfitting as long as the validation loss is decreasing. The final test loss of my model smaller and test accuracy is larger than that of the original one.

Moreover, the time consumption is greatly decreased because:

* + The number of total trainable parameters is decreased as more convolutional layers are used so that the parameter number in the first fully connected layer decreases
  + Augmentation function is rewritten with torchvision, resulting in faster augmentation in GPU instead of CPU.

# Good design/ high performance/ high model efficiency/ good discussion (1 mark)

(Listing your modifications and corresponding motivations by dots are preferred)

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| Insufficient or partially wrong explanation, interpretation about your modifications | -1.0 |
| Wrong explanation, interpretation about your modifications | -2.0 |
| Wrong plot, or no plots, result provided. | -1.0 |
| Cannot reproduce the result of your model. | -4.0 |

1. **The main dataset site on github (https://github.com/zalandoresearch/fashion-mnist) in- cludes a series of results for other network models (under Benchmark). How does your model compare? Explain why the ResNet18 model may produce better results than yours (or the other way around if this is the case).(1 mark).**

# Compare your model with others. Also provide a comparison in words (0.2 marks)

The performance of model with preprocessing is shown below for comparation

|  |  |  |
| --- | --- | --- |
| Model | Test Accuracy | Parameter size (million) |
| WRN40-4 | 0.967 | 8.9 |
| ResNet18 | 0.949 | 11 |
| My Model | 0.929 | 1.1 |
| 2 Conv | 0.922 | 0.1 |

We can observe that the test accuracy of my model is 2% lower than ResNet18. The result is expected. I think the difference is due to:

* + ResNet18 has 10 times more trainable parameters and 14 more convolutional layers than my model. The huge network could learn more features at various layers of abstraction and generalize better because of the additional intermediate features it learnt through training.
  + ResNet18 applies shortcut connections to overcome vanishing gradient problem, making values of derivative to each layer more significant and thus improve the performance overall.

# Discuss why the ResNet18 model may produce better results than yours (or the other way around if this is the case) (0.8 marks)

|  |  |
| --- | --- |
| **Mistake** | **Marks Deducted** |
| No comparison provided | -0.2 |
| Partially wrong discussion | -0.4 |
| No or wrong discussion | -0.8 |