影像處理、電腦視覺及深度學習概論
(Introduction to Image Processing, Computer Vision and Deep Learning)

Homework 1

TA:

彥博: nckubot65904@gmail.com

Office Hour: 17:00~19:00, Mon.

10:00~12:00, Fri.

At CSIE 9F Robotics Lab.

Notice (1/2)

- Copying homework is strictly prohibited!! Penalty: Grade will be zero for both persons!!
- If the code can't run, you can come to our Lab within one week and show that your programming can work. Otherwise, you will get zero!!
- Due date =>23:59:59, 2021/11/28 (Sun.)

  No delay. If you submit homework after deadline, you will get 0.
- Upload to => 140.116.154.1 -> Upload/Homework/Hw1
  - User ID: opencvdl2021 Password: opencvdl2021
- Format
  - Filename: Hw1_StudentID_Name_Version.rar
    - Ex: Hw1_F71234567_林小明_V1.rar
    - If you want to update your file, you should update your version to be V2, ex: Hw1_F71234567_林小明_V2.rar
  - Content: project folder*(including the pictures)

    *note: remove your "Debug" folder to reduce file size

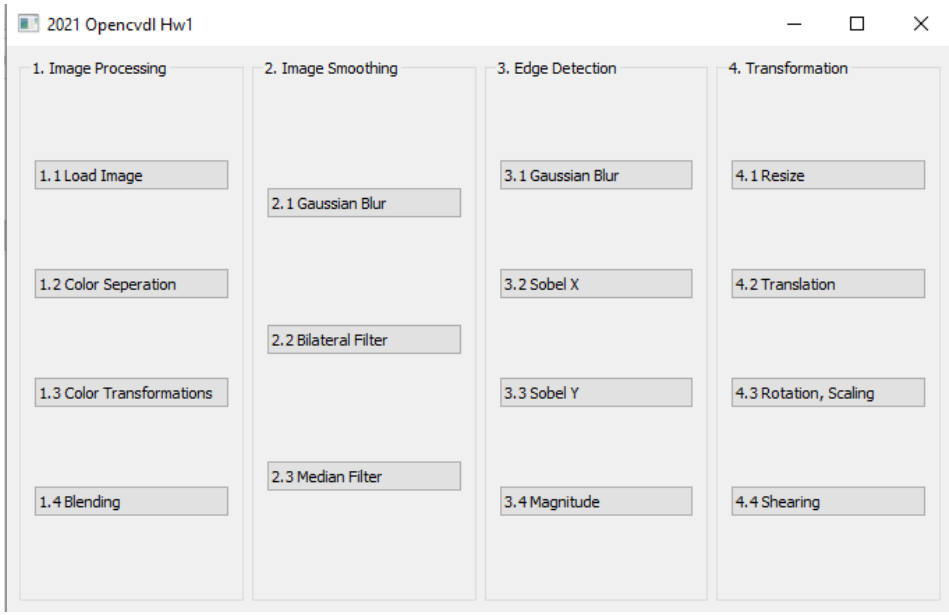
Notice (2/2)

- Python (recommended)
  - Python 3.7 (https://www.python.org/downloads/)
  - opencv-contrib-python (3.4.2.17)
  - Matplotlib 3.1.1
  - UI framework: pyqt5 (5.15.1)

- C++ (check MFC guide in ftp)
  - OpenCV 3.3.1 (https://opencv.org/release.html)
  - Visual Studio 2015 (download from http://www.cc.ncku.edu.tw/download/)
  - UI framework: MFC

Assignment scoring (Total: 100%)

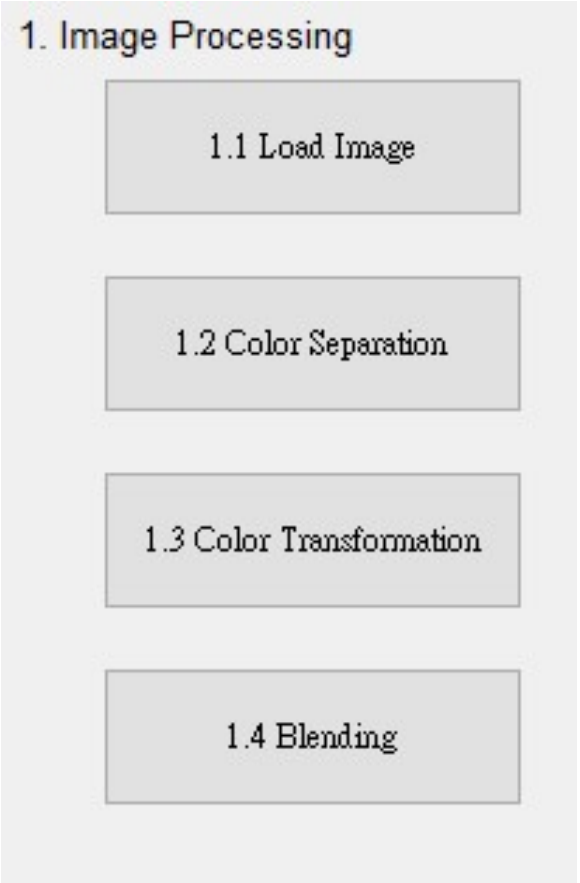
1. (20%) Image Processing (出題: Tina)
   - 1.1 (5%) Load Image File
   - 1.2 (5%) Color Separation
   - 1.3 (5%) Color Transformation
   - 1.4 (5%) Blending
2. (20%) Image Smoothing (出題: Willy)
   - 2.1 (5%) Gaussian blur
   - 2.2 (5%) Bilateral filter
   - 2.3 (10%) Median filter
3. (20%) Edge Detection (出題: Lydia)
   - 3.1 (5%) Gaussian Blur
   - 3.2 (5%) Sobel X
   - 3.3 (5%) Sobel Y
   - 3.4 (5%) Magnitude
4. (20%) Transforms: (出題: Ray)
   - 4.1 (5%) Resize
   - 4.2 (5%) Translation
   - 4.3 (5%) Rotation, Scaling
   - 4.4 (5%) Shearing
5. (20%) Training Cifar-10 Classifier Using VGG16 (出題: Tommy)



1. Image Processing (20%)

(出題: Tina)

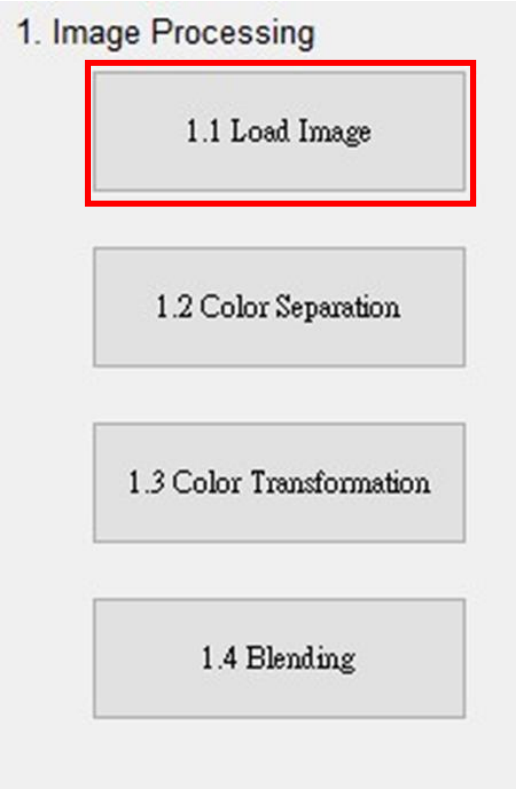
- 1.1 Load Image File (5%)
- 1.2 Color Separation (5%)
- 1.3 Color Transformation (5%)
- 1.4 Blending(5%)



1.1 Load Image File (5%)

(出題: Tina)

- Given: “Sun.jpg” image
- Q: 1) Open a new window to show the image (Sun.jpg)
  2) Show the height and width of the image in **console mode**
- Hint :
  - Textbook Chapter 2, p. 22~23
  - cv2.imread(), cv2.imshow()



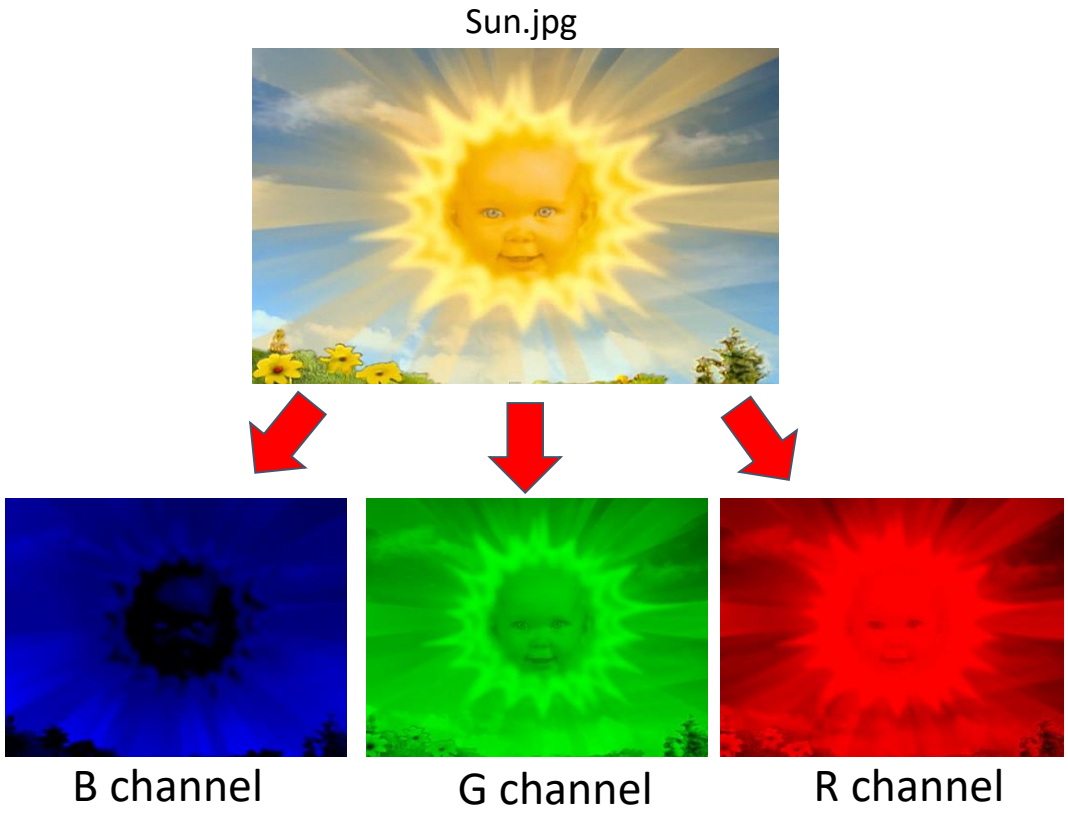

```
Height :  387
Width :  620
```

1.2 Color Separation (5%)

(出題：Tina)

- Given: a color image, “Sun.jpg”
- Q: 1) Extract 3 channels of the image BGR to 3 separated channels and show the result images.

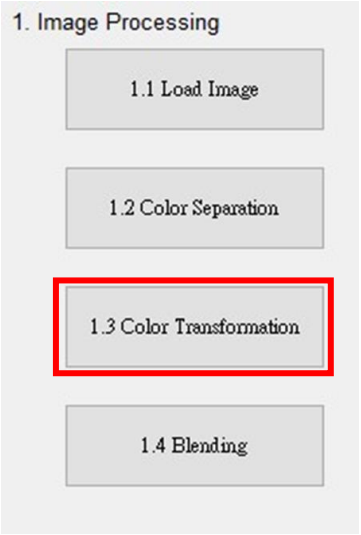
- Hint:
  - Textbook Chapter 3, p.31 ~ p.49
  - cv2.split(), cv2.merge()

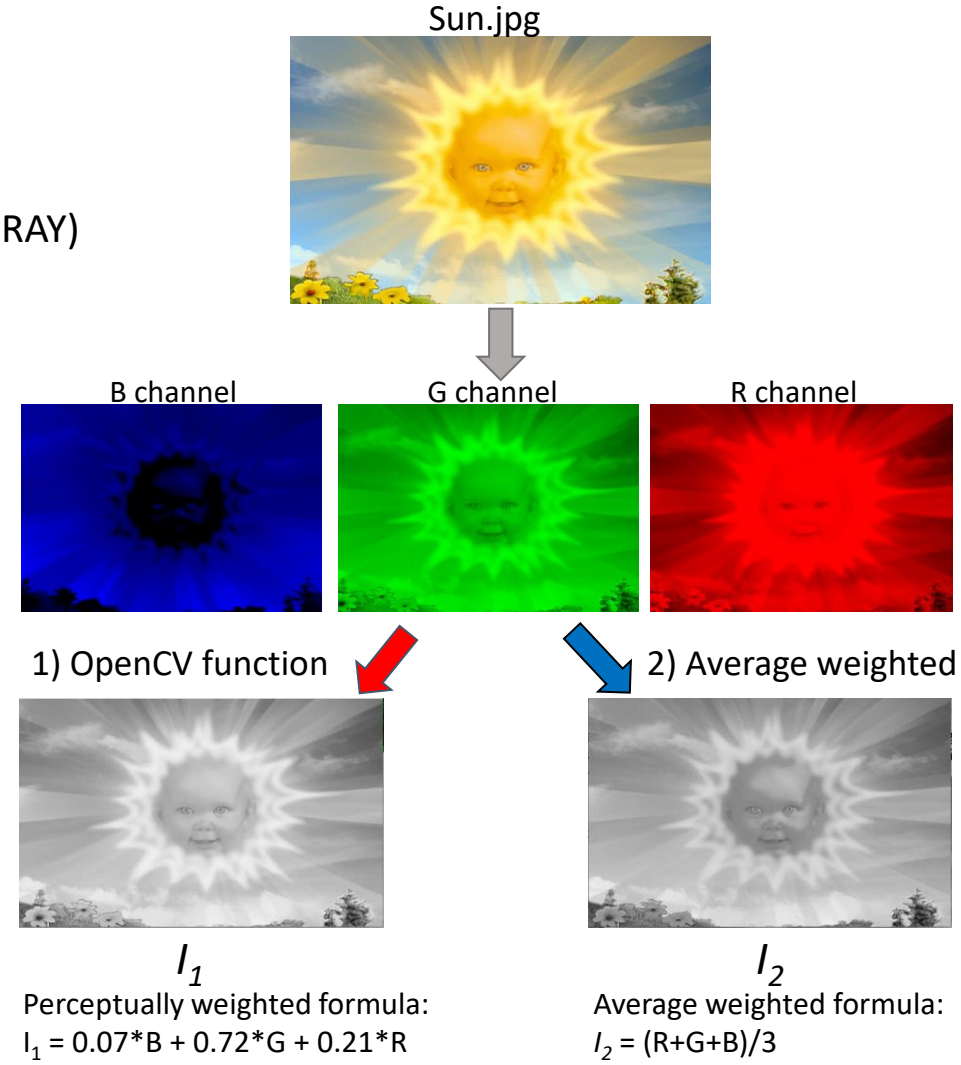


1.3 Color Transformation (5%)

(出題：Tina)

- Given: 1 color image: “Sun.jpg”
- Q: 1) Transform “Sun.jpg” into grayscale image I_1 by calling OpenCV function directly.
  2) Merge BGR separated channel images from problem 1.2 into grayscale image I_2 by $I_2 = (R+G+B)/3$.
  3) Show the above 2 results.
- Hint:
  - Textbook Chapter 3, p.56 ~ p.59
  - cv2.cvtColor(…, cv2.COLOR_BGR2GRAY)

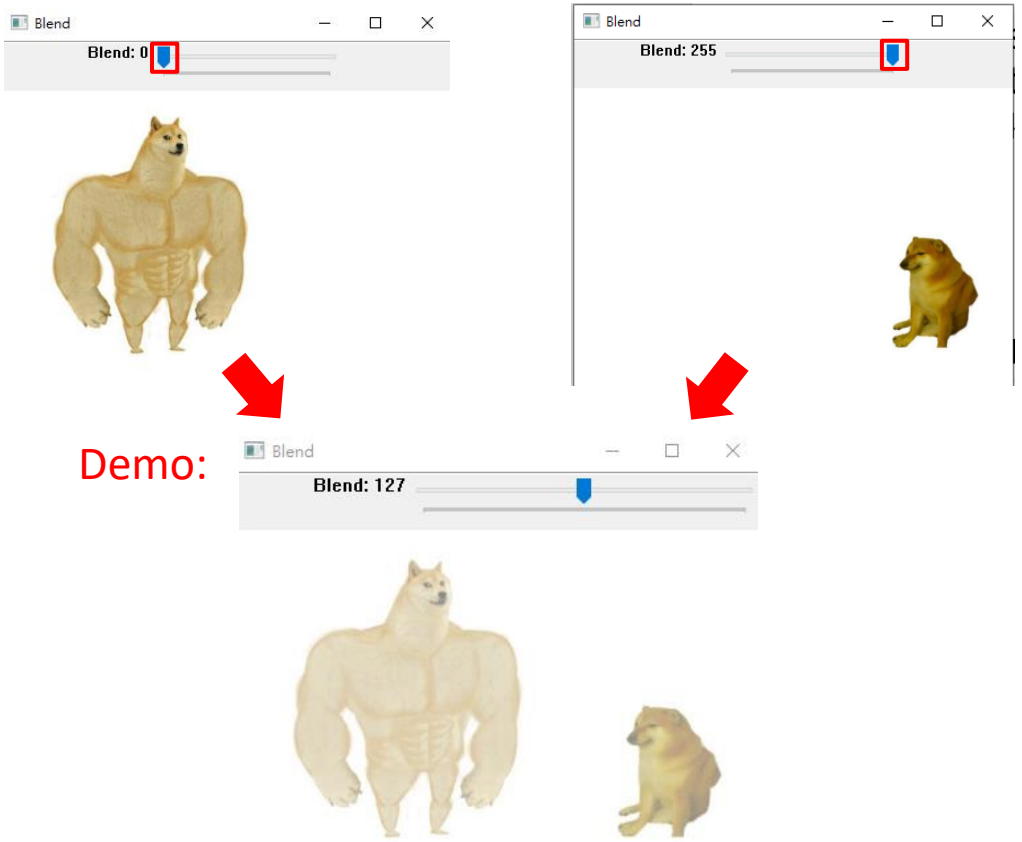




1.4 Blending (5%)

(出題：Tina)

- Given: 2 images, “Dog_Strong.jpg” and “Dog_Weak.jpg”
- Q: 1) Combine two images (Dog_Strong.jpg and Dog_Weak.jpg).
  2) Use `Trackbar` to change the weights and show the result in the new window.
- Hint:
  - Textbook Chapter 3, p. 50 ~ 52
  - cv2.addWeighted(), cv2.createTrackbar()



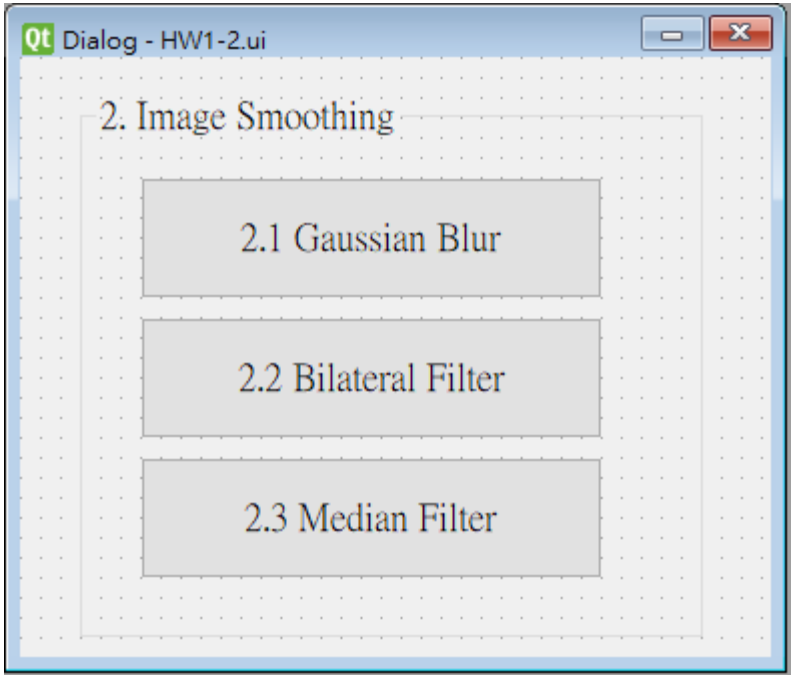
2. Image Smoothing (20%)

(出題: Willy)

2.1 Gaussian blur (5%)

2.2 Bilateral filter (5%)

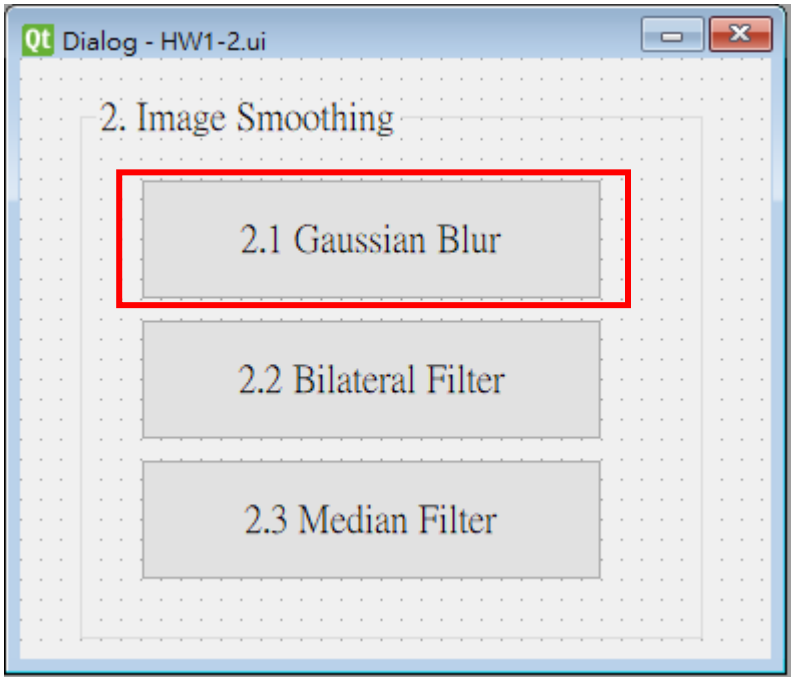
2.3 Median filter (10%)

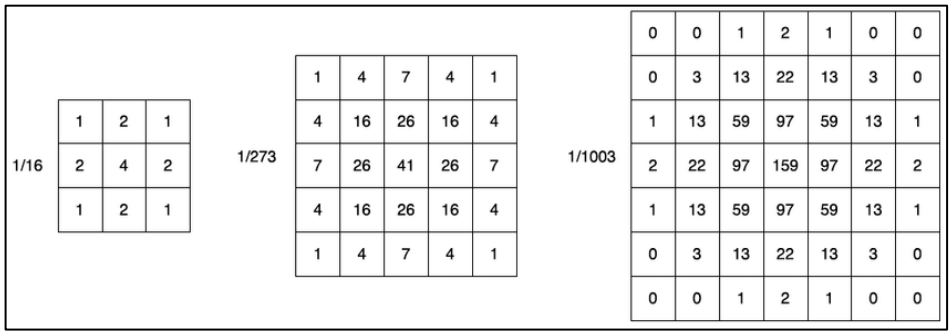


2.1 Gaussian Blur

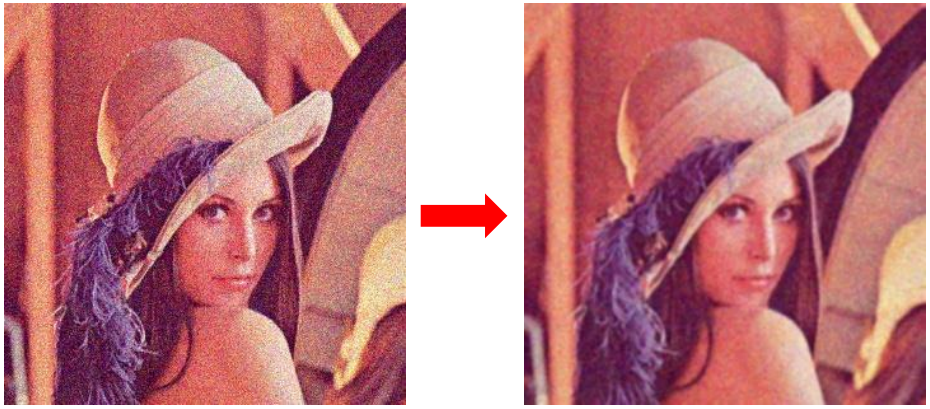
(出題：Willy)

- Given: a color image, “Lenna_whiteNoise.jpg”
- Q: 1) Apply 5x5 Gaussian filter to “Lenna_whiteNoise.jpg”
- Hint: Textbook Chapter 5, p.109 ~ p.115





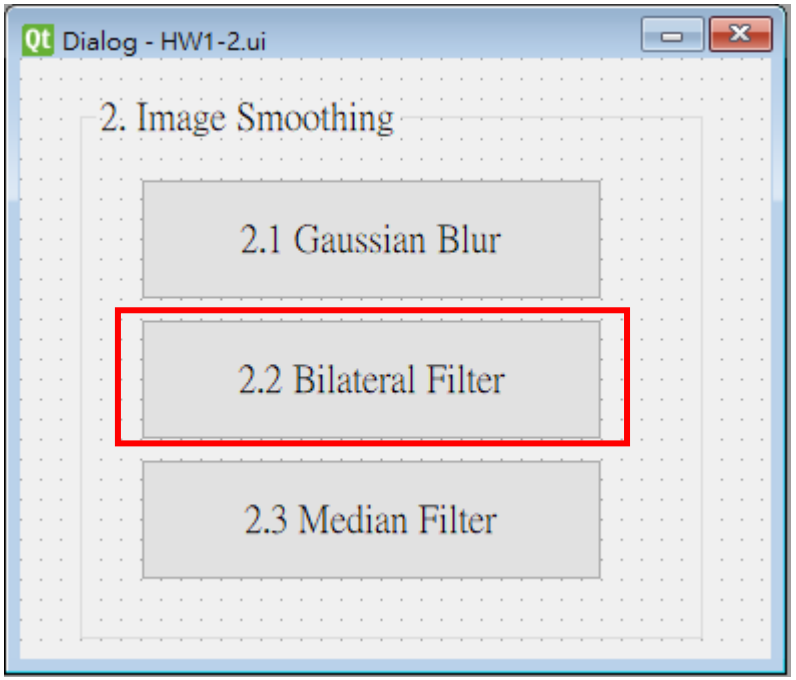
Lenna_whiteNoise.jpg

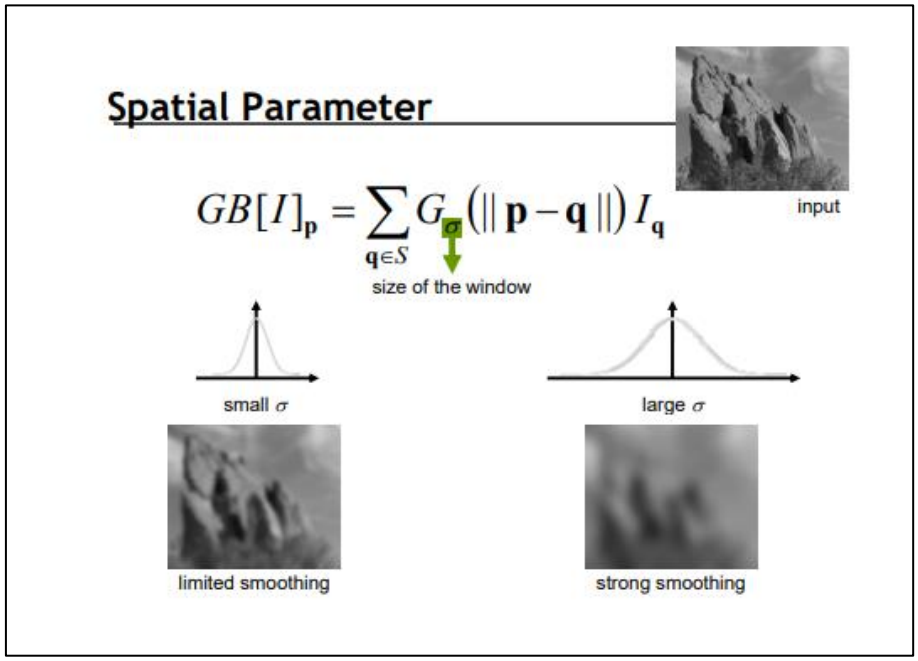


2.2 Bilateral Filter

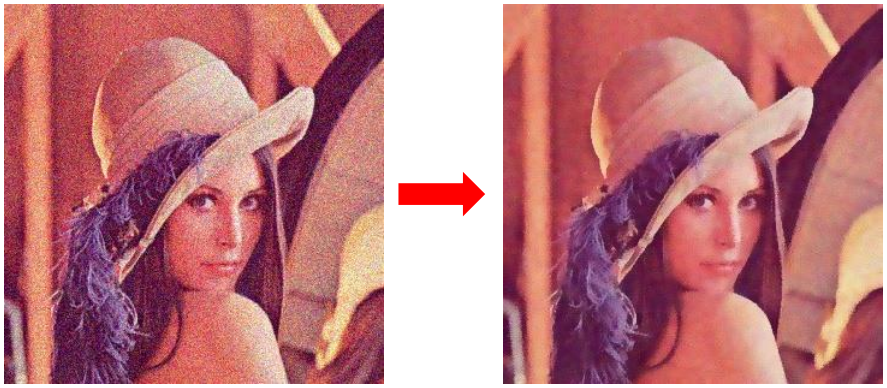
(出題：Willy)

- Given: a color image, “Lenna_whiteNoise.jpg”
- Q: 1) Apply 9x9 Bilateral filter with 90 sigmaColor and 90 sigmaSpace to “Lenna_whiteNoise.jpg”
- Hint: Textbook Chapter 5, p.109 ~ p.115





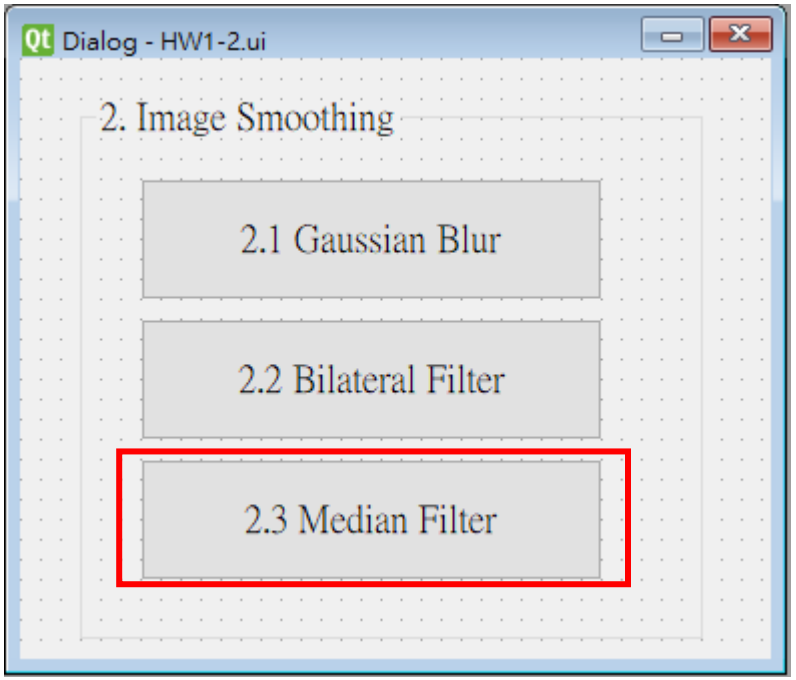
Lenna_whiteNoise.jpg



2.3 Median Filter

(出題：Willy)

- Given: a color image, “Lenna_pepperSalt.jpg”
- Q:
- 1) Apply 3x3 median filter to “Lenna_ pepperSalt.jpg”
- 2) Apply 5x5 median filter to “Lenna_ pepperSalt.jpg”
- Hint: Textbook Chapter 5, p.109 ~ p.115

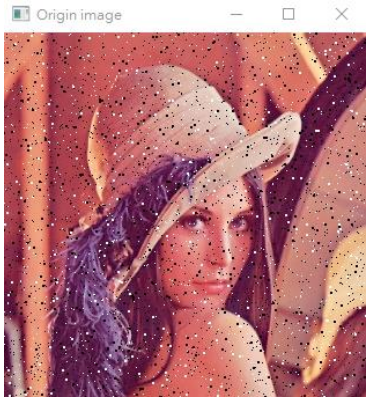


Median filter example

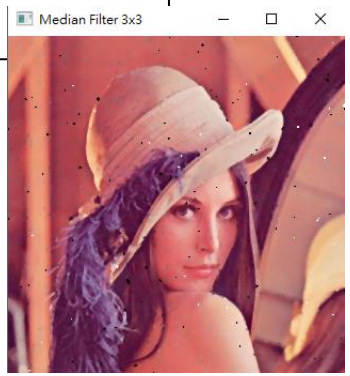
223	186	114
204	161	106
219	194	138

106 114 138 161 186 194 204 219 223

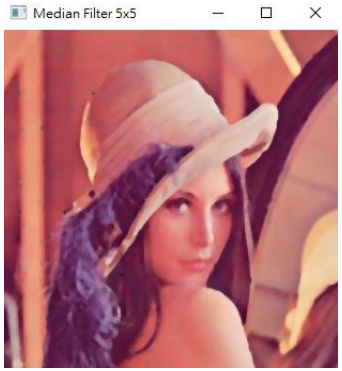
Lenna_pepperSalt.jpg



3x3



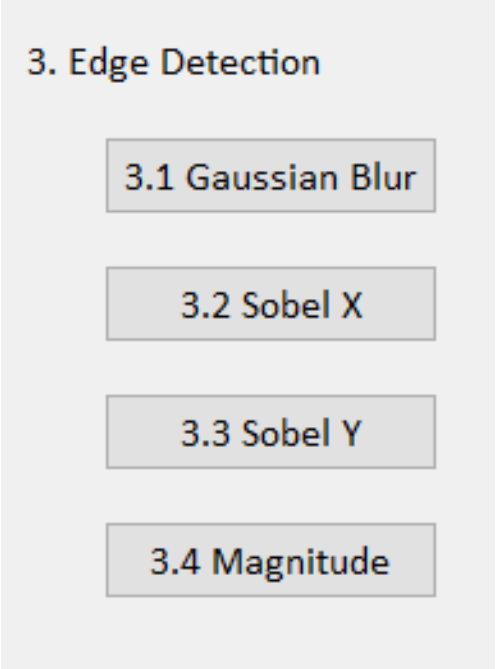
5x5



3. Edge Detection (20%)

(出題: Lydia)

- 3.1 Gaussian Blur (5%)
- 3.2 Sobel X (5%)
- 3.3 Sobel Y (5%)
- 3.4 Magnitude (5%)



3.1 Gaussian Blur (5%)

(出題：Lydia)

- Given: a RGB image, “House.jpg”
- Q: 1) **Gaussian Blur**: Convert the RGB image into a grayscale image, then smooth it by your own 3x3 Gaussian smoothing filter (Can not use OpenCV Function). Please show the result.
- Hint: Textbook Chapter 5, p.109 ~ p.114

  How to generate Gaussian Filter:

  ① Let $G_{init}(x,y) = \begin{bmatrix} (-1,-1) & (0,-1) & (1,-1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,1) & (0,1) & (1,1) \end{bmatrix}$

  ② Calculate $G(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$

  ③ Normalize $G(x,y), G_{norm}(x,y) = \begin{bmatrix} 0.045 & 0.122 & 0.045 \\ 0.122 & 0.332 & 0.122 \\ 0.045 & 0.122 & 0.045 \end{bmatrix}$

House.jpg

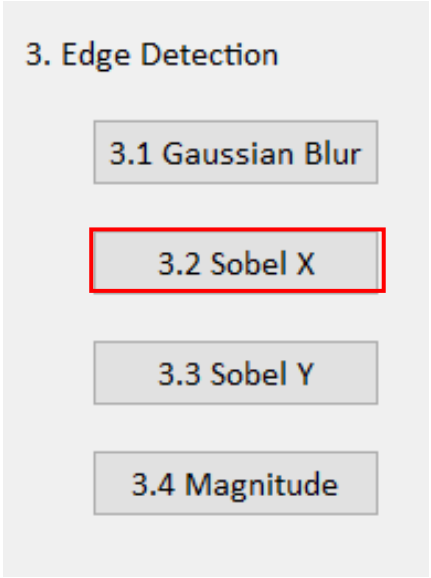
Grayscale

Gaussian Blur

3.2 Sobel X (5%)

(出題：Lydia)

- Given: the result of 3.1) Gaussian Blur
- Q: 2) **Sobel X**: Use Sobel edge detection to detect vertical edge by your own 3x3 Sobel X operator (Can not use OpenCV Function). Please show the result.
- Hint: Textbook Chapter 6, p.148 ~ 149



Gaussian Blur

-1	0	1
-2	0	2
-1	0	1

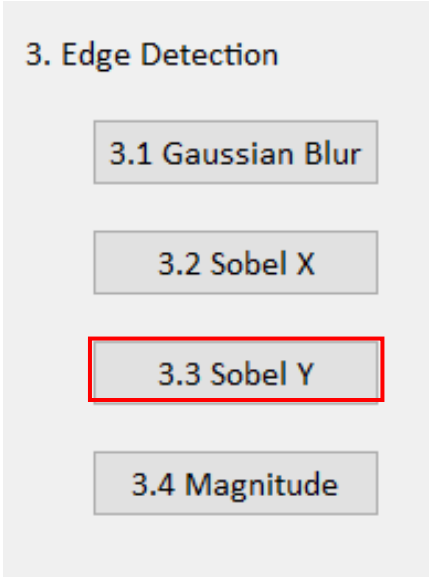
Sobel X Filter

Sobel X

3.3 Sobel Y (5%)

(出題：Lydia)

- Given: the result of 3.1) Gaussian Blur
- Q: 3) **Sobel Y**: Use Sobel edge detection to detect horizontal edge by your own 3x3 Sobel Y operator (Can not use OpenCV Function). Please show the result.
- Hint: Textbook Chapter 6, p.148 ~ 149



Gaussian Blur

1	2	1
0	0	0
-1	-2	-1

Sobel Y Filter

Sobel Y

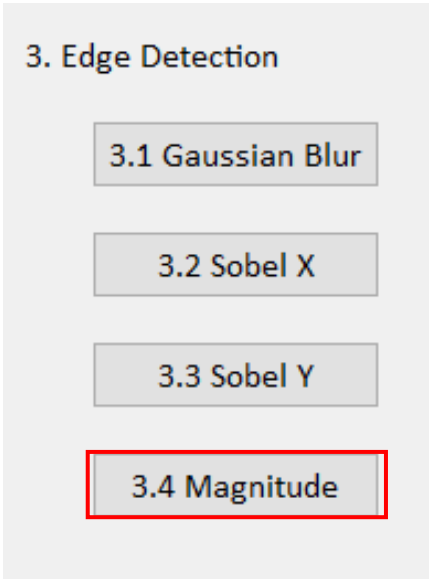
3.4 Magnitude (5%)

(出題：Lydia)

- Given: the result of 3.2) Sobel X and 3.3) Sobel Y
- Q: 4) **Magnitude**: Use the results of 3.2) Sobel X and 3.3) Sobel Y to calculate the magnitude. Please show the result.
- Hint: Textbook Chapter 6, p.148 ~ 149

$$\text{Magnitude} = \sqrt{\|Sobel_X{}^2 + Sobel_Y{}^2\|}$$

Normalize the result to 0~255.



Sobel X

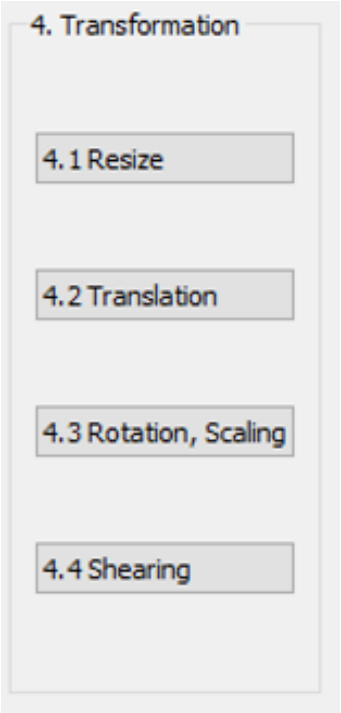
Sobel Y

Magnitude

4. Transforms (20%)

(出題: Ray)

- 4.1 Resize (5%)
- 4.2 Translation(5%)
- 4.3 Rotation, Scaling (5%)
- 4.4 Shearing (5%)



4. Transforms: Resize, Translation, Rotation, Scaling, Shearing(20%)

(出題: Ray)

- Given: "*SQUARE-01.png*"
- Q:Please resize, translate , rotate, scale and shearing the Red SQUARE (as image below) with following parameters (should be entered in the GUI)
  1) Resize = (256,256)
  2) Translation with:
     - ■Xnew = Xold + 0 pixels = 128 + 0 = 128
     - ■Ynew = Yold + 60 pixels = 128 + 60 = 188

     Point C (128, 128) is center of resized image

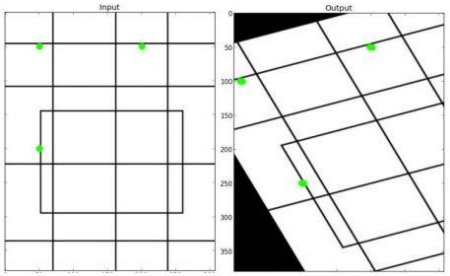
     Point C'(128, 188) is new center of image
  3) Angle = 10º (counter-clockwise)

     Scale = 0.5 , window size (400,300)
  4) Shearing with:

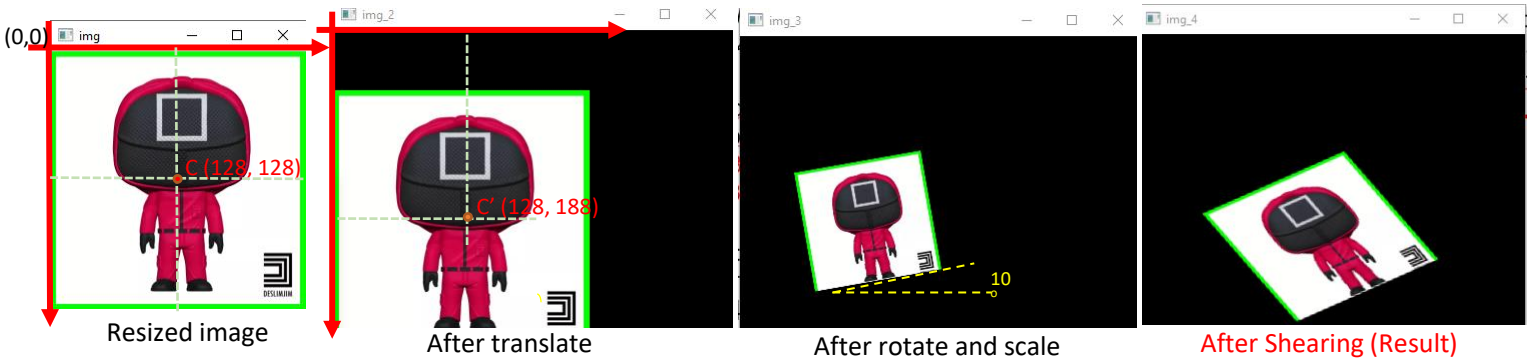
     Old location: ([[50,50],[200,50],[50,200]])

     New location: ([[10,100],[200,50],[100,250]])
- Hint: Textbook Chapter 12, (p.407 ~ 412)

  python: `cv.warpAffine`



- EX:



5. Training Cifar10 Classifier Using VGG16 (20%) (出題: Tommy)

5.1 Show Training Images (4%)

5.2 Show Hyperparameters (4%)

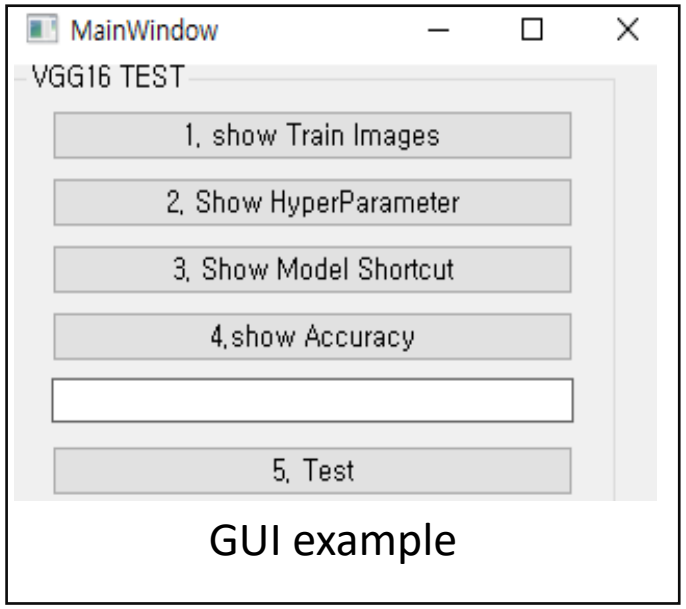
5.3 Show Model Structure (4%)

5.4 Show Accuracy and Loss (4%)

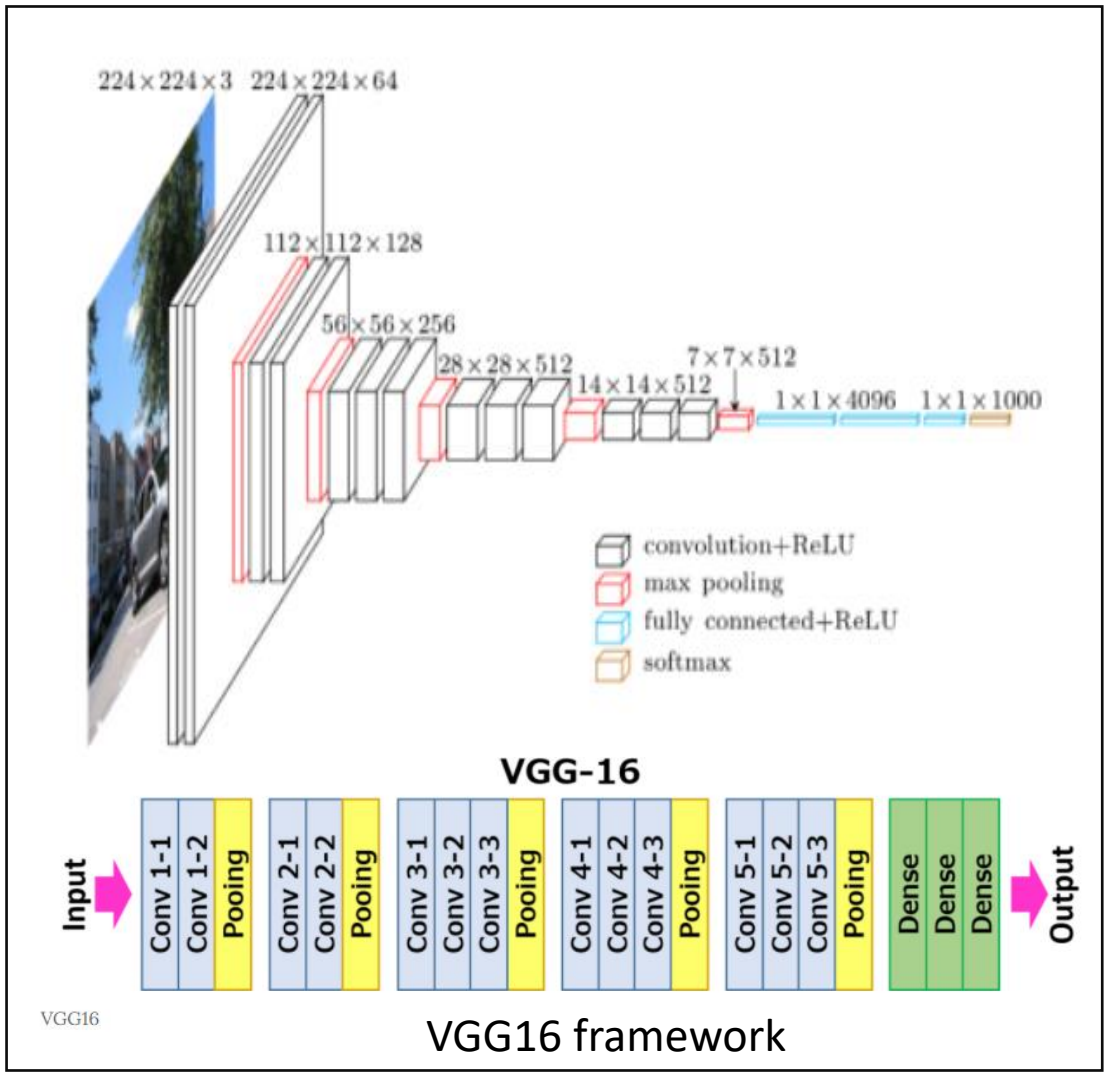
5.5 Test (4%)

5.0 Training Cifar10 Classifier Using VGG16（出題：Tommy)

1. Learn how to construct VGG16 and train it on Cifar10.
2. Environment Requirement
   1) Python
   2) Tensorflow / PyTorch (Can choose the one)
   3) opencv-contrib-python
   4) Matplotlib



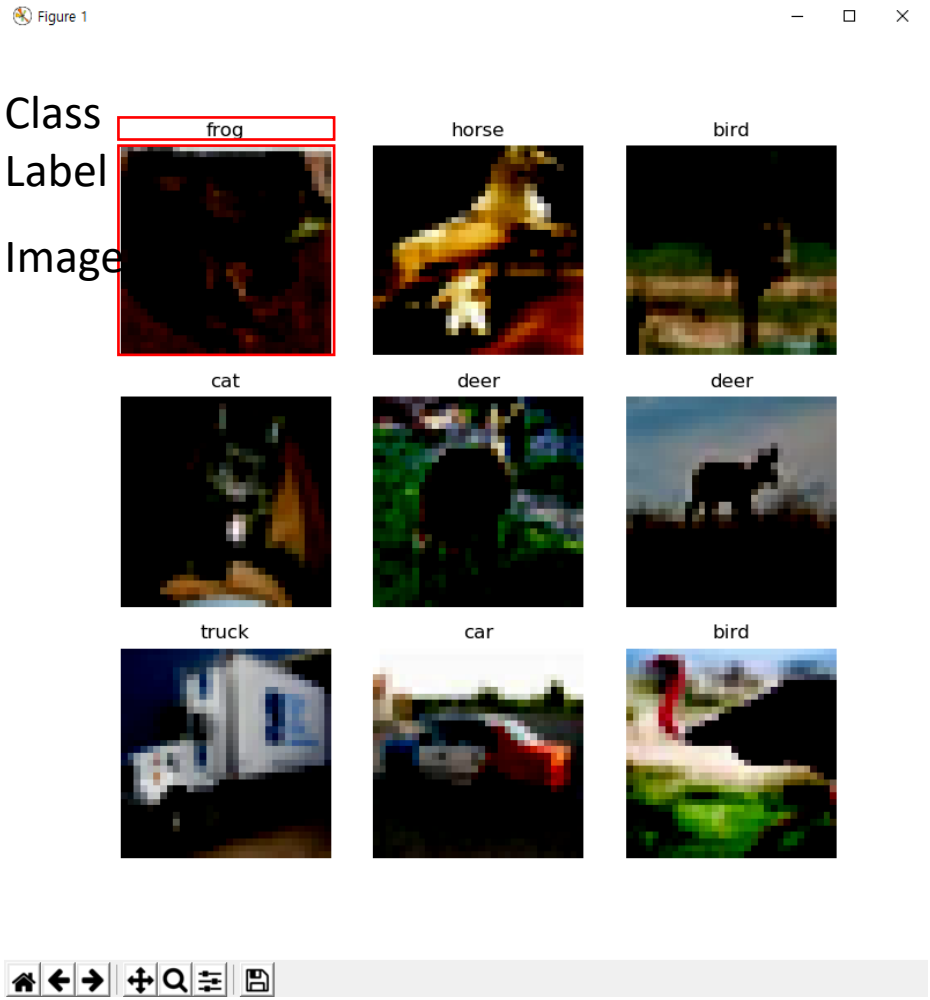
GUI example



VGG16 framework

3. Reference
   1) https://paperswithcode.com/method/vgg (VGG16, Paper and Source Code)
   2) https://www.cs.toronto.edu/~kriz/cifar.html (Cifar10 Dataset)

5.1 Load Cifar10 **training dataset**, and then **show 9 Images(Pop-up)** and **Labels** respectively (4%)



0 airplane
1 automobile
2 bird
3 cat
4 deer
5 dog
6 frog
7 horse
8 ship
9 truck

10 Class of CIFAR 10 Dataset

◆ Hint

Use Matplotlib 4 function

1. figure()
2. title()
3. Axis()
4. imshow()

- Can refer by tutorial at the Matplotlib library official web-site

https://matplotlib.org/stable/tutorials/index.html

5.2 Print out training **hyperparameters on the terminal**

(batch size, learning rate, optimizer). (4%)

```
hyperparameters:
batch size: 32
learning rate: 0.001
optimizer: SGD
```

◆ Hint

Use the python print function

Print('Batch Size {} ', format(Batchsize variable name))

5.3 Construct and show your model structure **by print out on the terminal** (You can use available architecture provided by ML framework to build your model) (4%)

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 32, 32]	896
BatchNorm2d-2	[-1, 32, 32, 32]	64
ReLU-3	[-1, 32, 32, 32]	0
Conv2d-4	[-1, 32, 32, 32]	9,248
BatchNorm2d-5	[-1, 32, 32, 32]	64
ReLU-6	[-1, 32, 32, 32]	0
MaxPool2d-7	[-1, 32, 16, 16]	0
Conv2d-8	[-1, 64, 16, 16]	18,496
BatchNorm2d-9	[-1, 64, 16, 16]	128
ReLU-10	[-1, 64, 16, 16]	0
Conv2d-11	[-1, 64, 16, 16]	36,928
BatchNorm2d-12	[-1, 64, 16, 16]	128
ReLU-13	[-1, 64, 16, 16]	0
Conv2d-14	[-1, 128, 16, 16]	73,856
BatchNorm2d-15	[-1, 128, 16, 16]	256
ReLU-16	[-1, 128, 16, 16]	0
Conv2d-17	[-1, 128, 16, 16]	147,584
BatchNorm2d-18	[-1, 128, 16, 16]	256
ReLU-19	[-1, 128, 16, 16]	0
Conv2d-20	[-1, 128, 16, 16]	147,584
BatchNorm2d-21	[-1, 128, 16, 16]	256
ReLU-22	[-1, 128, 16, 16]	0
MaxPool2d-23	[-1, 128, 8, 8]	0
Conv2d-24	[-1, 256, 8, 8]	295,168
BatchNorm2d-25	[-1, 256, 8, 8]	512
ReLU-26	[-1, 256, 8, 8]	0
Conv2d-27	[-1, 256, 8, 8]	590,080
BatchNorm2d-28	[-1, 256, 8, 8]	512
ReLU-29	[-1, 256, 8, 8]	0
Conv2d-30	[-1, 256, 8, 8]	590,080
BatchNorm2d-31	[-1, 256, 8, 8]	512
ReLU-32	[-1, 256, 8, 8]	0
Conv2d-33	[-1, 512, 8, 8]	1,180,160
BatchNorm2d-34	[-1, 512, 8, 8]	1,024
ReLU-35	[-1, 512, 8, 8]	0

Layers List of Model

After processing each layer Change of input data type

Number of trainable parameters

◆ Hint

Pytorch API
Use the two option

1) Summary function

from torchsummary import summary
-> import the package
summary(Model name, (Input Channel, Input Width, Input Height))
-> run the function and print on the terminal

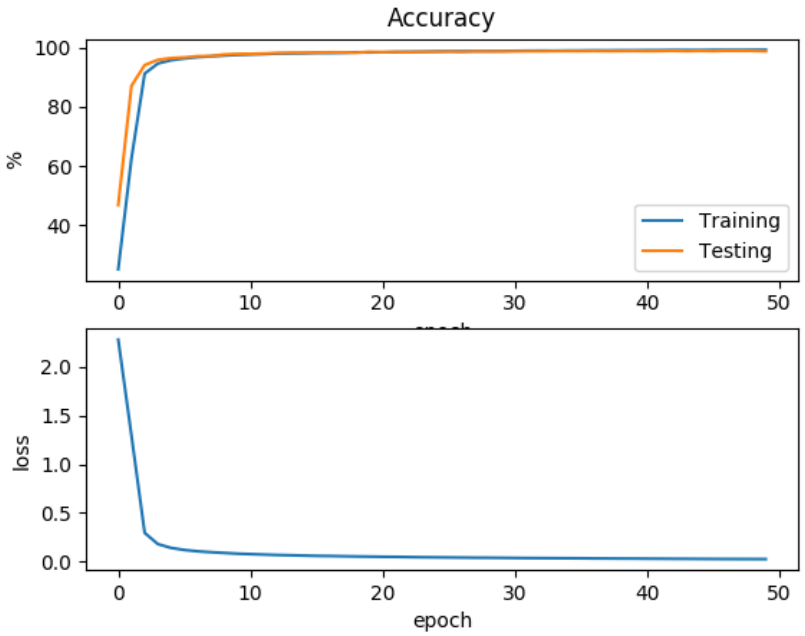
2) Print function

From torchvision import models
-> import the package
Model = VGG16()
-> Make the mode data
Print(Model)

- Can refer this web-site

https://pypi.org/project/pytorch-model-summary/

5.4 **Training** your model at least **20** epochs **by your own computer**, then **save your model** and **take a screenshot of your training loss and accuracy**. **No saved images no points** (4%)



◆ Hint

There are two option. (record accuracy/loss per epoch)

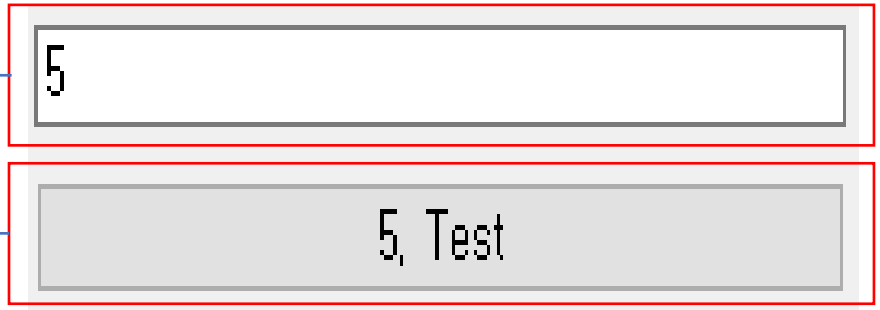
1) Just use the normal method(Above image used this way)

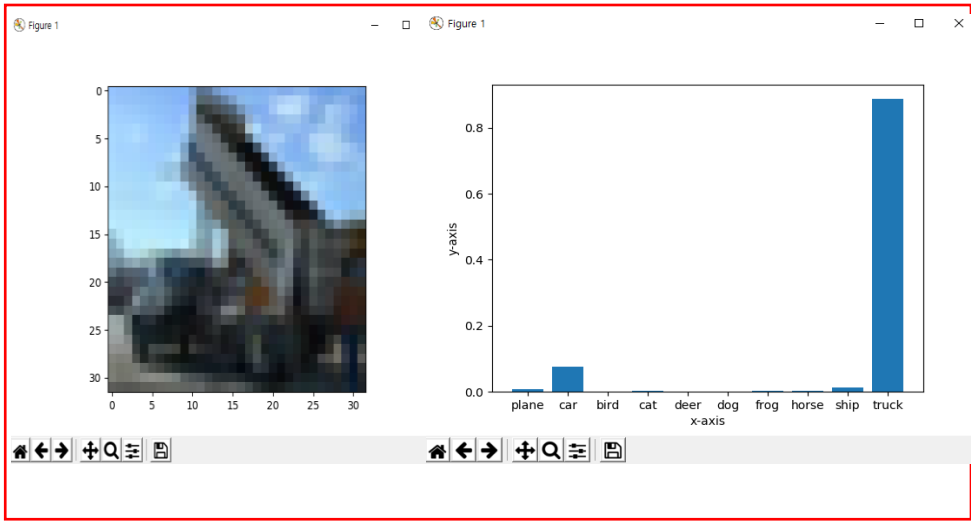
https://www.pyimagesearch.com/2021/07/19/pytorch-training-your-first-convolutional-neural-network-cnn/

2) Use tensorboard in Tensorflow API or tensorboard in pytorch

https://pytorch.org/tutorials/intermediate/tensorboard_tutorial.html

5.5 **Load your model** trained at 5.4, let us **choose one image from test images**, **inference the image**, show the result image and class(use the pop-up) (4%)





1. Choose the any data
2. Run Inference
3. Show the result

◆ Hint

Can refer two web-site

1. https://towardsdatascience.com/understanding-pytorch-with-an-example-a-step-by-step-tutorial-81fc5f8c4e8e#5017
2. https://pytorch.org/tutorials/