DLCV HW2

R08922050 Chih-Chun YANG

November 2019

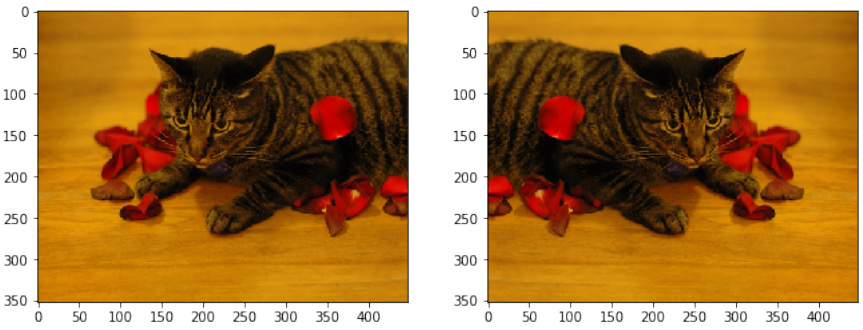
1 Baseline Model

1.1 Describe how you pre-process the data.(5%)(Any data augmentation technique used? Do you normalize the data?)

- Image Pre-processing:
  - Scale image to 0-1 by the following formula:

    $$ScaledImage = \frac{image - min(image)}{max(image) - min(image)}$$

  - Normalize image with $mean = [0.485, 0.456, 0.406]$, $std = [0.229, 0.224, 0.225]$
- Data Augmentation:
  - I adopt horizon flip as the data augmentation technique to increase amount of data. Since if we rotate the image 90, 180, 270 degree, it makes normal people uneasy to identify the image accurately, I did not adopt the rotation method to do data augmentation.



(a) Origin Image (b) Flipped Image

Figure 1: Data Augmentation

1.2 Show the following two figures:

1.2.1 Training loss versus number of training iterations (Y coordinate: training loss. X coordinate: number of iterations.) (5%)

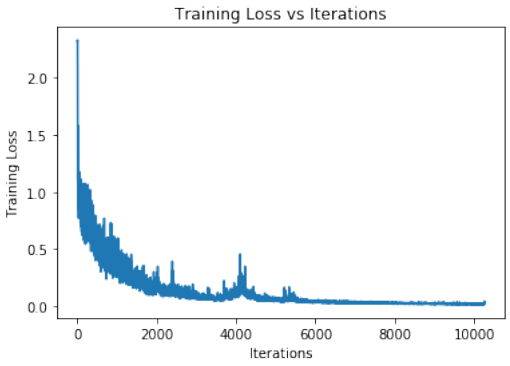


Figure 2: Training Loss

1.2.2 IoU score on validation set versus number of training iterations (Y coordinate: IoU score on validation set. X coordinate: number of epochs.) (5%)

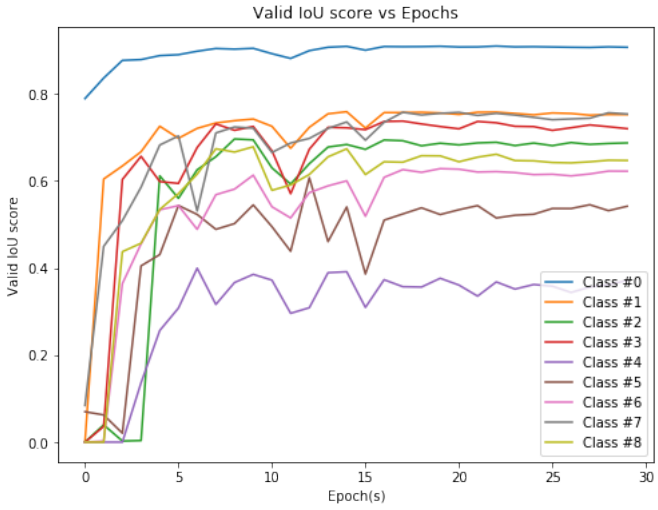


Figure 3: Valid IoU of each class

1.3 Visualize at least one semantic segmentation result for each class. (5%)

- Visualization:

(a) Class #0 (b) Class #1 (c) Class #2

(d) Class #3 (e) Class #4 (f) Class #5

(g) Class #6 (h) Class #7 (i) Class #8

Figure 4: Image of Each Class

1.4 Report mIoU score and per-class IoU score of the baseline model. Which class has the highest IoU score? Which class has the lowest IoU score? Please also hypothesize the reason why. (10%)

- **Validation mIoU: 0.668175**
- Validation Class IoU:
  - class #0: 0.90874
  - class #1: 0.75574
  - class #2: 0.68618
  - class #3: 0.72449
  - class #4: 0.37620
  - class #5: 0.52245
  - class #6: 0.62784
  - class #7: 0.75484
  - class #8: 0.65711

- **Class #0 get the highest IoU score** while **class4 get the lowest IoU score**. My hypothesis is as following:
  - **Class #0 is background**. The reason that the class #0 has the highest IoU score is probably that background usually occupies a lot of space in each image. So the model can easily learn from data that which part is the background.
  - **Class #4 is tv/monitor**. One of the reasons that class #4 has the lowest IoU score is probably that the amount of class #4 data is low. There are only 412 images containing tv/monitor. Another reason is that there is usually image in the tv/monitor, which may lead the model harder to distinguish whether it is tv/monitor or the image inside.

    * class #0: 5460
    * class #1: 2670
    * class #2: 514
    * class #3: 333
    * class #4: 412
    * class #5: 294
    * class #6: 805
    * class #7: 681
    * class #8: 851

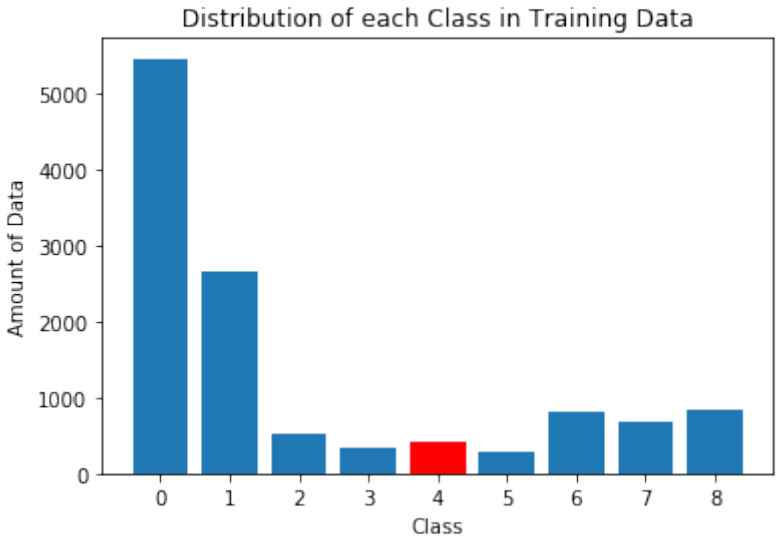
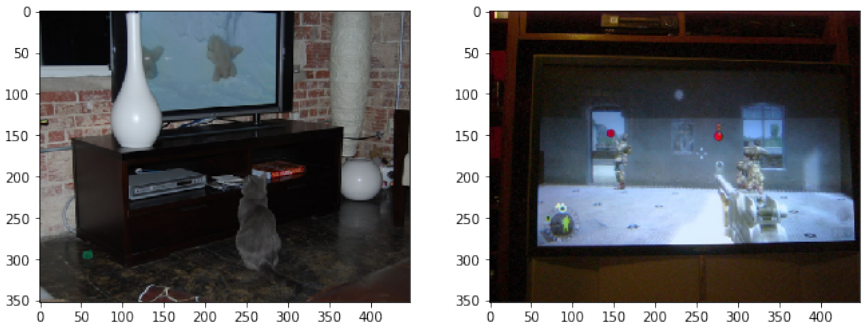


Figure 5: Data Distribution



(a) Example 1 (b) Example 2

Figure 6: There are other kinds of image in the TV/Monitors

2 Improved model (If the mIoU of your improved model is worse than that of the baseline model, you will get at most 20 points in this part.)

2.1 Draw the model architecture of your improved model. (5%)

- I use **ResNet34** instead of ResNet18 in the improved model and Conv2dTranspose as up-sampling layer.

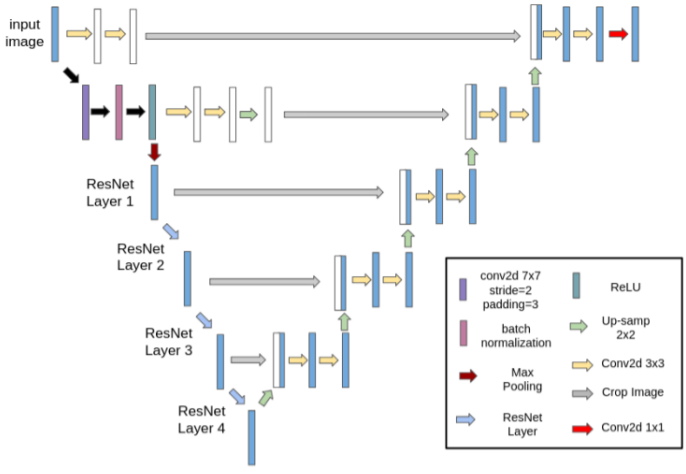


Figure 7: Model Structure (inspired by U-Net)[1] [2]

2.2 Discuss the reason why the improved model performs better than the baseline one. You may conduct some experiments and show some evidences to support your discussion. (15%)

- The feature extraction method of baseline model and improved model are almost the same (ResNet18, ResNet34) and the up-sampling process are almost the same (Improved model has not only Conv2dTranspose but also Conv2d layer). But in the improved model, during each process of up-sampling, the model combine the information of the output specific layer of ResNet and the current image, which provide the model more information about the contour of input image.

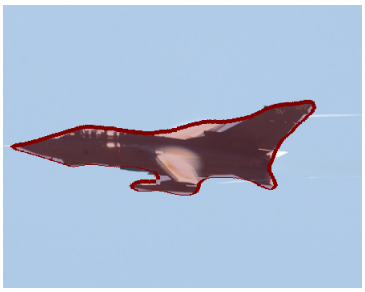
- To prove that model benefits from combining the information of the output specific layer of ResNet and the current image, **I change the feature extractor of improved model to ResNet18 (the same as baseline model)** and train the model on the same dataset.

Class	Baseline	Improved
#0	0.90874	**0.92169**
#1	0.75574	**0.79583**
#2	0.68618	**0.72803**
#3	0.72449	**0.75767**
#4	0.37620	**0.40609**
#5	0.52245	**0.66072**
#6	0.62784	**0.70601**
#7	0.75484	**0.78577**
#8	0.65711	**0.74360**
Validation mIoU	0.668175	**0.722824**

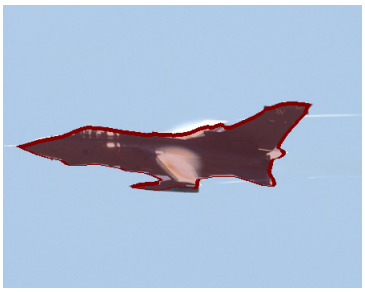
2.3 To prove that your improved model is better than the baseline one, report the mIoU score of your improved model. Please also show some semantic segmentation results of your improved model and the baseline model. (10%)

- **Validation mIoU: 0.780503**
  - class #0: 0.92899
  - class #1: 0.81564
  - class #2: 0.75926
  - class #3: 0.82283
  - class #4: 0.48201
  - class #5: 0.80188
  - class #6: 0.82054
  - class #7: 0.82746
  - class #8: 0.76591

– Visualization:



(a) Class #0 Baseline



(b) Class #0 Improved

(c) Class #1 Baseline

(d) Class #1 Improved

(e) Class #2 Baseline

(f) Class #2 Improved

(g) Class #3 Baseline

(h) Class #3 Improved

Figure 8: Comparison of Baseline & Improved Model

(a) Class #4 Baseline

(b) Class #4 Improved

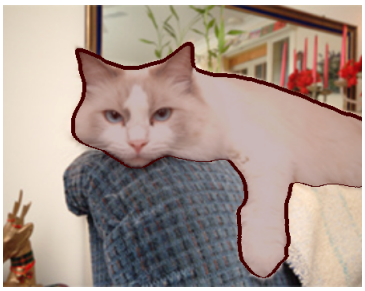
(c) Class #5 Baseline

(d) Class #5 Improved

(e) Class #6 Baseline

(f) Class #6 Improved

(g) Class #7 Baseline



(h) Class #7 Improved

Figure 9: Comparison of Baseline & Improved Model (Conti.)

(a) Class #8 Baseline

(b) Class #8 Improved

Figure 10: Comparison of Baseline & Improved Model (Conti.)

3 Problem 2

3.1 (1%) Given a variance σ^2, the convolution of a 2D Gaussian kernel can be reduced to two sequential convolutions of a 1D Gaussian kernel. Show that convolving with a 2D Gaussian filter is equivalent to sequentially convolving with a 1D Gaussian filter in both vertical and horizontal directions.

- $G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$
- $G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} = G(x)G(y)$
- $\sum_{x=0}^{I_x} \sum_{y=0}^{I_y} G(x,y) = \sum_{x=0}^{I_x} \sum_{y=0}^{I_y} G(x)G(y) = \sum_{x=0}^{I_x} G(x) \sum_{y=0}^{I_y} G(y)$
- Thus we can do 1D Gaussian filter in both vertical and horizontal directions and multiply them to get the 2d Gaussian filter.

3.2 (3%) Implement a discrete 2D Gaussian filter using a 3 x 3 kernel with 1 / 2 ln 2 . Use the provided lena.png as input, and plot the output image in your report. Briefly describe the effect of the filter.

- There seems no significant difference between the original image and the image applied Gaussian filter.
- The image applied Gaussian filter is a little bit blurry and smooth than the original image.
- I apply symmetric padding in numpy to the image when applying gaussian filter.

(a) Before (b) After

Figure 11: Lena Before and After Gaussian Filter

3.3 (4%) Consider the image I(x, y) as a function I : $R^2 -> R$. When detecting edges in an image, it is often important to extract information from the derivatives of pixel values. Write down your answers of k_x and k_y. Also, plot the resulting images I_x and I_y using the provided lena.png as input.

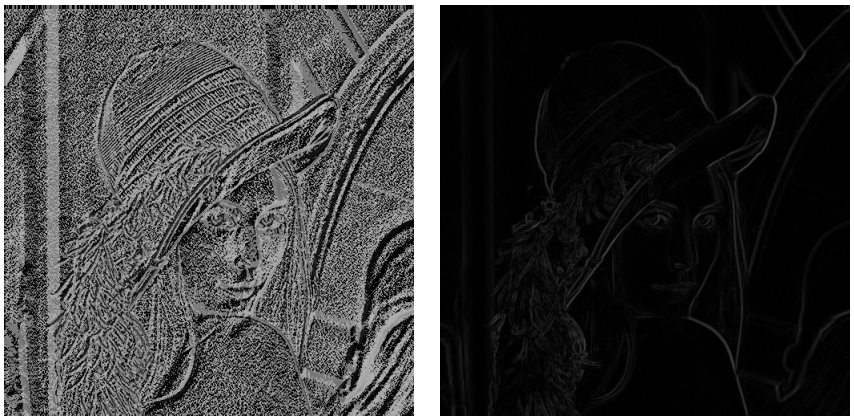
- $k_x = [-0.5, 0, 0.5]$, $k_y = [-0.5, 0, 0.5]^T$
- I use symmetric padding in numpy as padding method.

(a) I_x (b) I_y

Figure 12: I_x and I_y

3.4 (2%) Use both the provided lena.png and the Gaussian-filtered image you obtained in 2. as input images. Plot the two output gradient magnitude images in your report. Briefly explain the differences in the results.

- Visualization:



(a) Origin Image (b) Gaussian-filtered Image

Figure 13: Gradient Magnitude Images

- The Gradient Magnitude of Gaussian-filtered image is more clear than the original image because the Gaussian-filter will remove the noise in the image so there will be less noise during computing gradient.

3.5 Collaborators

- 林子淵 R07921100
- 黃孟霖 R07922170
- 張緣彩 R07922141

References

[1] *U-Net: Semantic Segmentation with PyTorch.* https://github.com/milesial/Pytorch-UNet.

[2] Thomas Brox Olaf Ronneberger, Philipp Fischer. *U-Net: Convolutional Networks for Biomedical Image Segmentation.* MICCAI, 2015.