

Homework #2

Problem 1: Semantic Segmentation (100/110)

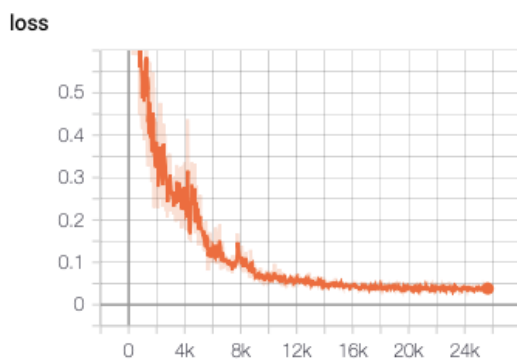
1. Baseline model

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

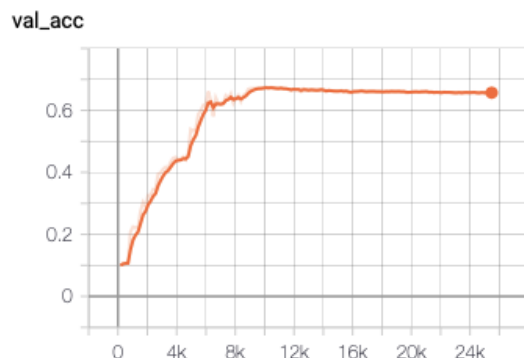
因為 segmentation mask 其實也可看作一張影像，所以當 data 做 transform 的時候，target mask 也要跟著一起變 (跟 classification 的 target class 不一樣)，而 torchvision transforms 提供的方法是 random 的，無法確保 data 跟 target mask 做一樣的 transform，所以必須要自己 implement transform，此處參考了[1]，分別使用了 random crop, random horizontal flipping, random vertical flipping，最後再使用 torchvision 提供的 normalize function，對 image 做 normalize (mask 不用做 normalize)。

- B. Show the following two figures:

- 甲、Training loss versus number of training iterations (Y coordinate: training loss. X coordinate: number of iterations.) (5%)
- 乙、IoU score on validation set versus number of training iterations (Y coordinate: IoU score on validation set. X coordinate: number of epochs.) (5%)

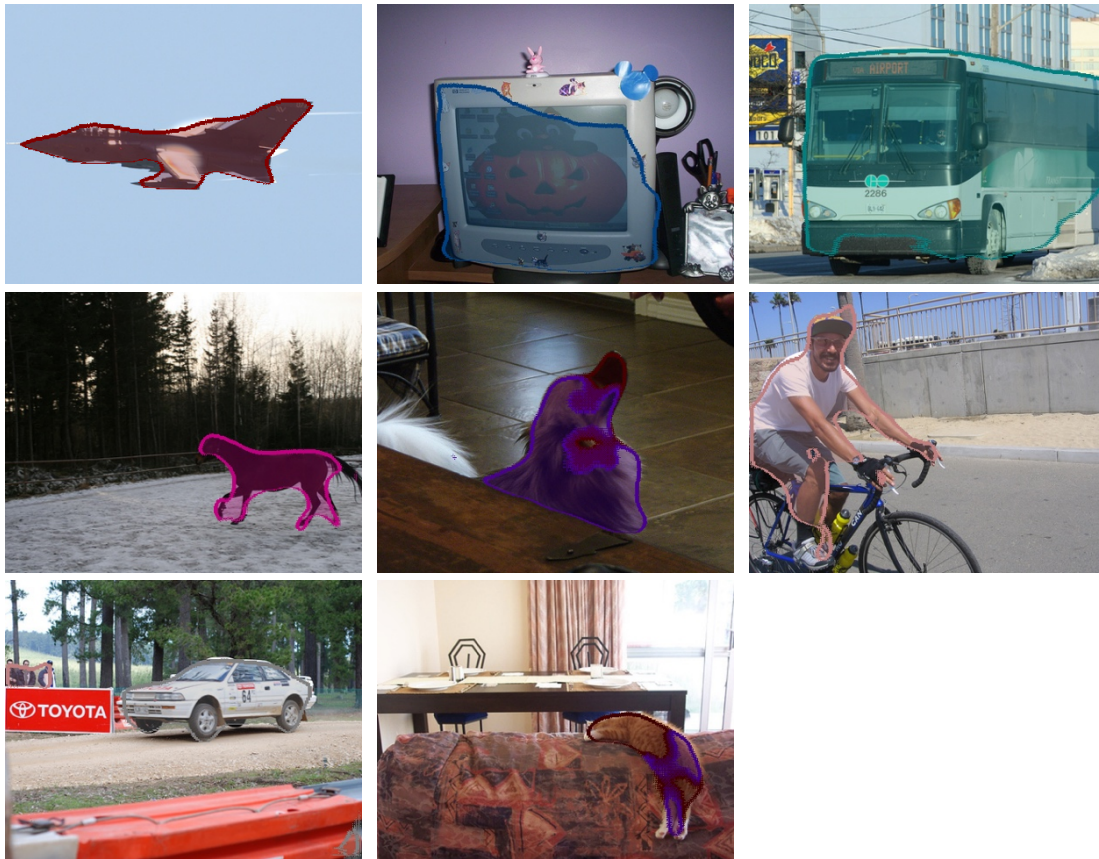


Training loss



IoU score on validation set

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



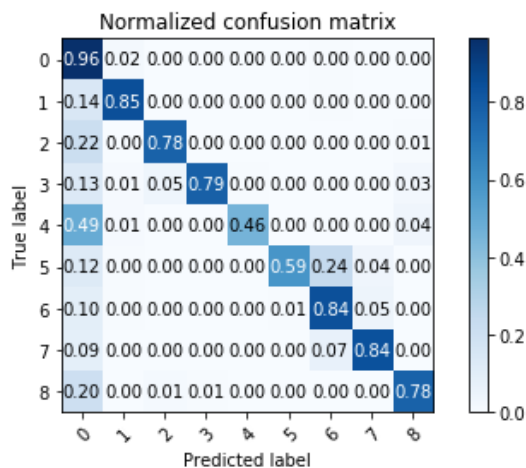
- D. Report mIoU score and per-class IoU score of the baseline model. Which class has the highest IoU score? Which class has the lowest IoU score? Please also hypothesize the reason why. (10%)

```
# preds: 500; pred.shape: (500, 352, 448)
# labels: 500; labels.shape: (500, 352, 448)
class #0 : 0.90832
class #1 : 0.76032
class #2 : 0.65438
class #3 : 0.71115
class #4 : 0.38774
class #5 : 0.53485
class #6 : 0.64630
class #7 : 0.76651
class #8 : 0.70657

mean_iou: 0.675128
```

```
class 0 appears 5460 times
class 1 appears 2670 times
class 2 appears 514 times
class 3 appears 333 times
class 4 appears 412 times
class 5 appears 294 times
class 6 appears 805 times
class 7 appears 681 times
class 8 appears 851 times
```

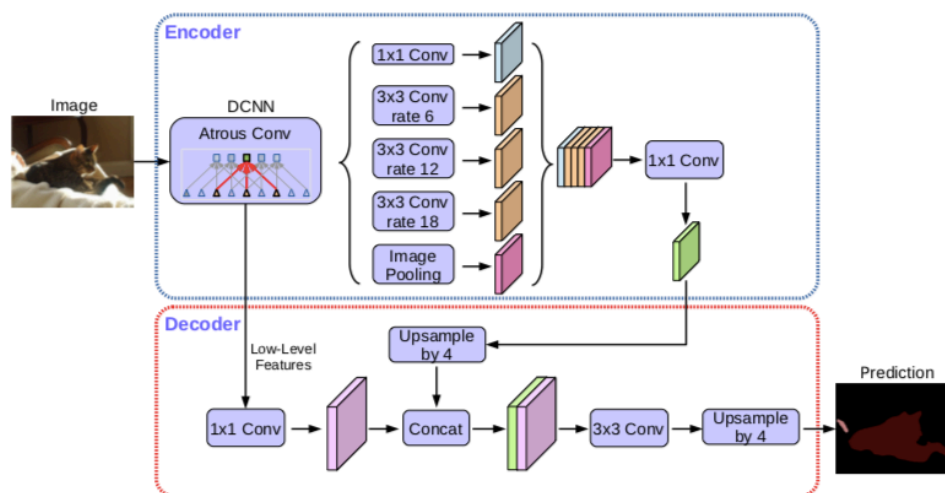
分類最好的是 class 1，最差的是 class 4。統計每個 class 在 train set 出現的次數後可以發現，class 1 出現最多次，因此 model 在這個 class 表現較好。而 class 4 雖然不是出現次數最少的，但是從下圖的 confusion matrix 可以觀察到，model 容易將其誤認為背景，有可能是因為 TV/ Monitor 裡面的內容都很不一樣，model 無法為其找到一個規律，所以表現比較差。



2. Improved model

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

我所使用的 improved model 是 DeepLab v3 +，參考[2]。架構圖如[3]上所提供的：



B. Discuss the reason why the improved model performs better than the baseline one. (15%)




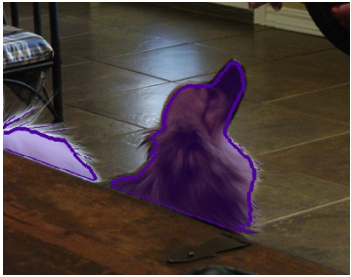

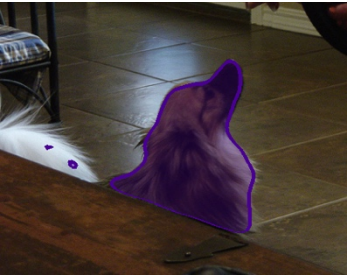


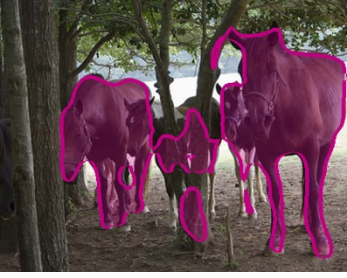



DeepLab v3+ 比原 baseline model 好的原因有以下幾點：

- 1) 選用的 backbone 為 resnet 101，比 vgg16 抽取 features 的能力要強
- 2) 使用 Atrous Conv，可以增加 receptive field。且藉由調整 dilation rate 可以調整 receptive field，進而獲取 multi scale context information
- 3) 使用 encoder-decoder 的架構，可以獲得比較精準的 mask 邊界

- C. 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%)

```
# preds: 500; pred.shape: (500, 352, 448)
# labels: 500; labels.shape: (500, 352, 448)
class #0 : 0.93109
class #1 : 0.80976
class #2 : 0.82419
class #3 : 0.83799
class #4 : 0.57999
class #5 : 0.76909
class #6 : 0.84872
class #7 : 0.87821
class #8 : 0.81360

mean_iou: 0.810292
```

Ground Truth	Baseline	Improved
		
		
		
		



Problem 2: Image Filtering (10/110)

1. 證明部分請參照最後面
2. (3%) Implement a discrete 2D Gaussian filter using a 3x3 kernel with $\sigma \approx \frac{1}{2\ln 2}$.

Use the provided lena.png as input, and plot the output image in your report.

Briefly describe the effect of the filter.

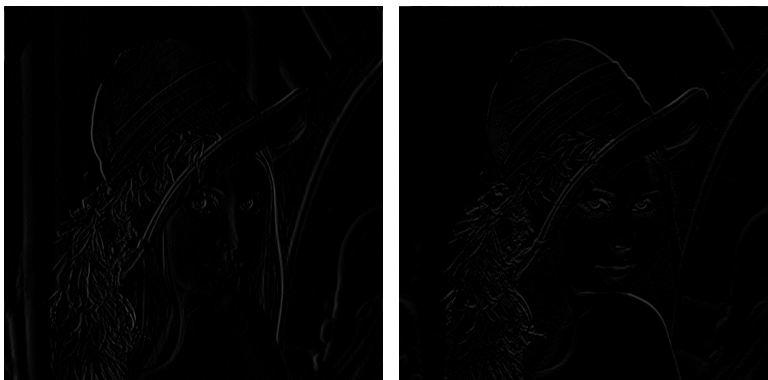


Lena 原圖

經過 Gaussian Filter

觀察原圖與結果圖可以發現，比較細緻的地方，例如帽子上的毛，被模糊了。而 Gaussian filter 可以減少圖像的雜訊，或是平滑細節的部分。

3. 計算部分請參照最後面。plot the resulting images I_x and I_y using the provided lena.png as input.



I_x

I_y

4. 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.

由下圖可以觀察到，雖然不是很明顯，但是經過 Gaussian filter 的影像，在算 gradient magnitude 求出的 edge 在帽子的毛那邊比較少。



Lena Im

Gaussian-filtered Im

Reference:

- [1] <https://discuss.pytorch.org/t/torchvision-transforms-how-to-perform-identical-transform-on-both-image-and-target/10606/6>
- [2] <https://github.com/jfzhang95/pytorch-deeplab-xception>
- [3] <https://arxiv.org/pdf/1802.02611.pdf>
- [4] https://cecas.clemson.edu/~stb/ece847/internal/cvbook/ch03_filtering.pdf

1. Show that convolving with a 2D Gaussian filter is equivalent to sequentially convolving with a 1D Gaussian filter in both vertical and horizontal direction.

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

$$\text{1D kernel} = G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

對於一個 2D 訊號 $f(x, y)$

$$\begin{aligned} f(x, y) * G(x, y) &= \sum_i \sum_j f(x-i, y-j) \exp\left\{\frac{-(i^2+j^2)}{2\sigma^2}\right\} \\ &= \sum_i \left[\sum_j f(x-i, y-j) \exp\left\{\frac{-j^2}{2\sigma^2}\right\} \right] \exp\left\{\frac{-i^2}{2\sigma^2}\right\} \\ &= [f(x, y) * G(y)] * G^T(x) \end{aligned}$$

例如:

$$f * \left(\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \right) = \left(f * \frac{1}{4} [1 \ 2 \ 1] \right) * \left(\frac{1}{4} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \right)$$

$$\text{因為: } \frac{1}{4} [1 \ 2 \ 1] * \frac{1}{4} \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

3. 已知 $I_x(x, y) = \frac{\partial I}{\partial x} \approx \frac{1}{2} [I(x+1, y) - I(x-1, y)]$

$$I_y(x, y) = \frac{\partial I}{\partial y} \approx \frac{1}{2} [I(x, y+1) - I(x, y-1)]$$

$$I_x = I * k_x$$

$$I_y = I * k_y$$

求 1D convolution kernel $k_x \in \mathbb{R}^{1 \times 3}$, $k_y \in \mathbb{R}^{3 \times 1}$ 使得

$$\text{令 } k_x = [k_{x1}, k_{x2}, k_{x3}]$$

$$\begin{aligned} I * k_x &= I(x-1, y) \cdot k_{x3} + I(x, y) \cdot k_{x2} + I(x+1, y) \cdot k_{x1} \\ &= \frac{1}{2} I(x+1, y) - \frac{1}{2} I(x-1, y) \end{aligned}$$

$$\Rightarrow k_{x3} = -\frac{1}{2}, k_{x2} = 0, k_{x1} = \frac{1}{2} \Rightarrow k_x = \left[\frac{1}{2}, 0, -\frac{1}{2} \right]$$

	$x, y-1$	
$x-1, y$	x, y	$x+1, y$
	$x, y+1$	

同理, 令 $k_y = [k_{y1}, k_{y2}, k_{y3}]^T$

$$I * k_y = I(x, y-1) \cdot k_{y3} + I(x, y) \cdot k_{y2} + I(x, y+1) \cdot k_{y1}$$

$$= \frac{1}{2} I(x, y+1) - \frac{1}{2} I(x, y-1)$$

$$\Rightarrow k_{y3} = -\frac{1}{2}, k_{y2} = 0, k_{y1} = \frac{1}{2} \Rightarrow k_y = \begin{bmatrix} \frac{1}{2} \\ 0 \\ -\frac{1}{2} \end{bmatrix}$$

$$\Rightarrow k_x = [\frac{1}{2}, 0, -\frac{1}{2}], k_y = \begin{bmatrix} \frac{1}{2} \\ 0 \\ -\frac{1}{2} \end{bmatrix} \quad \#$$