Lab3

中興　黃聖祐　L122502

April 10, 2024

1 Overview

Lab3 使用 UNet 和 ResNet34 結合 UNet 實作影像分割，分割的資料集爲 Oxford-iiit Pet Dataset，圖片以貓狗等寵物爲主的照片，目的在於學習語意分割和 pixel 級的分類。我們需要去訓練如何分割出寵物所在的區塊，像是

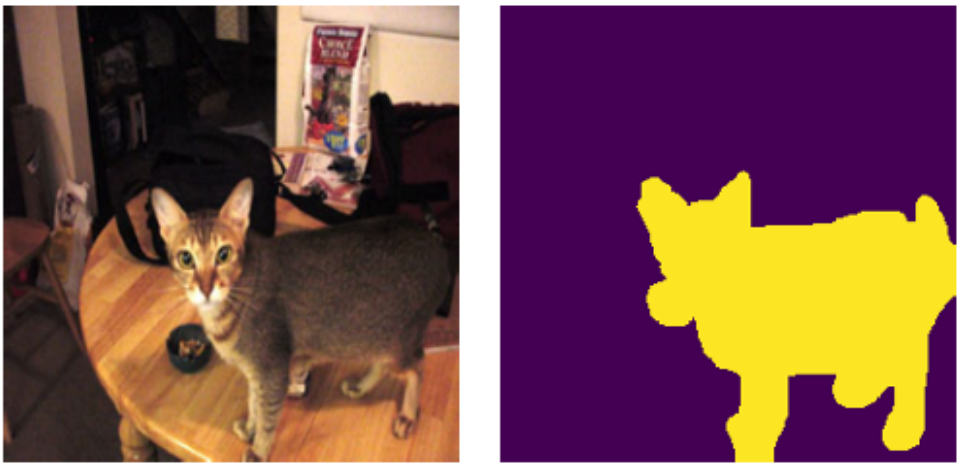


Image　　GT mask

2 Implementation Details

2.1 Details of my training, evaluating, inferencing code

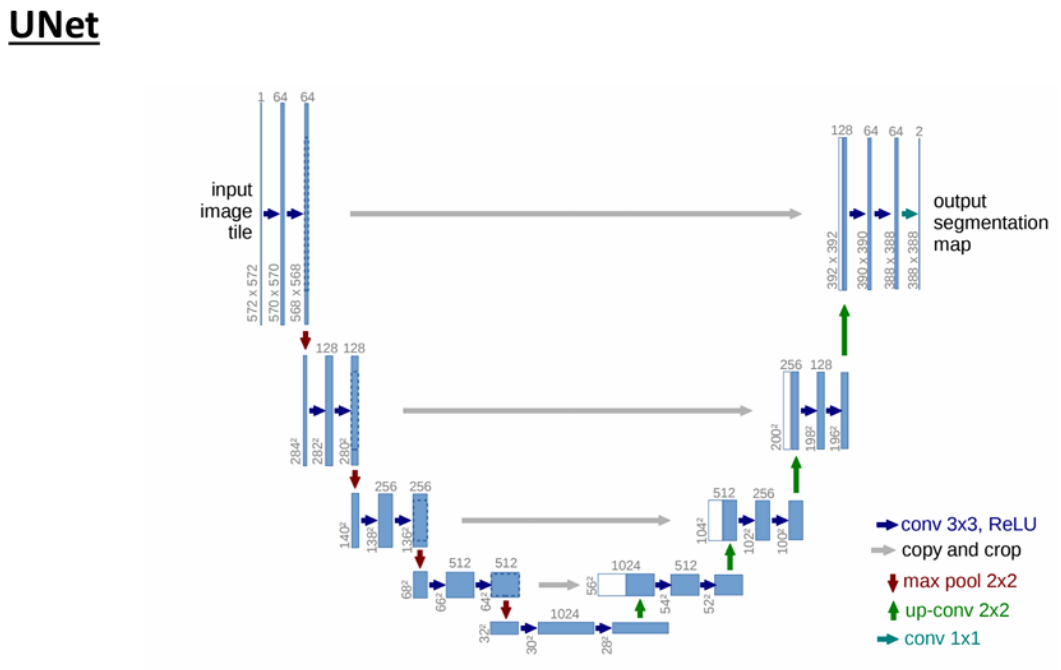
在 train.py 中，我先做前處理，將 dataset 載入並用 OxfordPetDataset() 做成可訓練 dataset。經由 data augmentation 和轉成 dataloader，再分別用兩種模型進行訓練。在訓練過程中，我使用了 Adam 作爲優化器，並加入了 SWA scheduler，在訓練後期使用，對多個不同的階段的權重進行平均，以獲得更穩定和準確的模型。這樣做的好處是可以抵抗訓練過程中的過擬合現象，從而提高模型的泛化能力。訓練完成後，儲存最後五個 epoch 的權重，並存取所有 epoch 的 accuracy。

在 inference.py 中，同樣先將測試資料集轉成 dataloader，再分別丟入兩個模型去做測試，並利用 utils 中 plot_compare_graph() 自定意函數隨機抽取數張圖片做比對，其結果類似於:

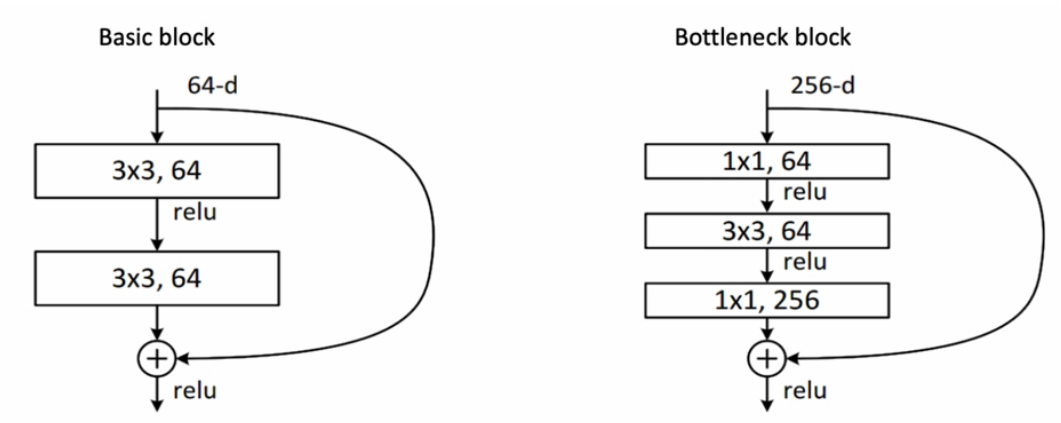
最後將所有測試資料透過兩種 model 所建構而成的 mask 儲存起來。

2.2 Details of my model (UNet and ResNet34UNet)

UNet: 因爲 UNet 每個 block 都是由兩個 convolution, batch normalize, relu，所以我將它們做成一個 block function，再由 UNet 架構去層層建構 decoder 以及 encoder。在 forward 中，我用 list 去記錄每個 block 的結果，用於在 decoder 時去與 decoder 的 input 做 concatenate。encoder 和 decoder 分別各五層，其中包含四層的 maxpooling 做降維，以及四層 convolution transpose 做升維，並在最後銜接一個卷基層。

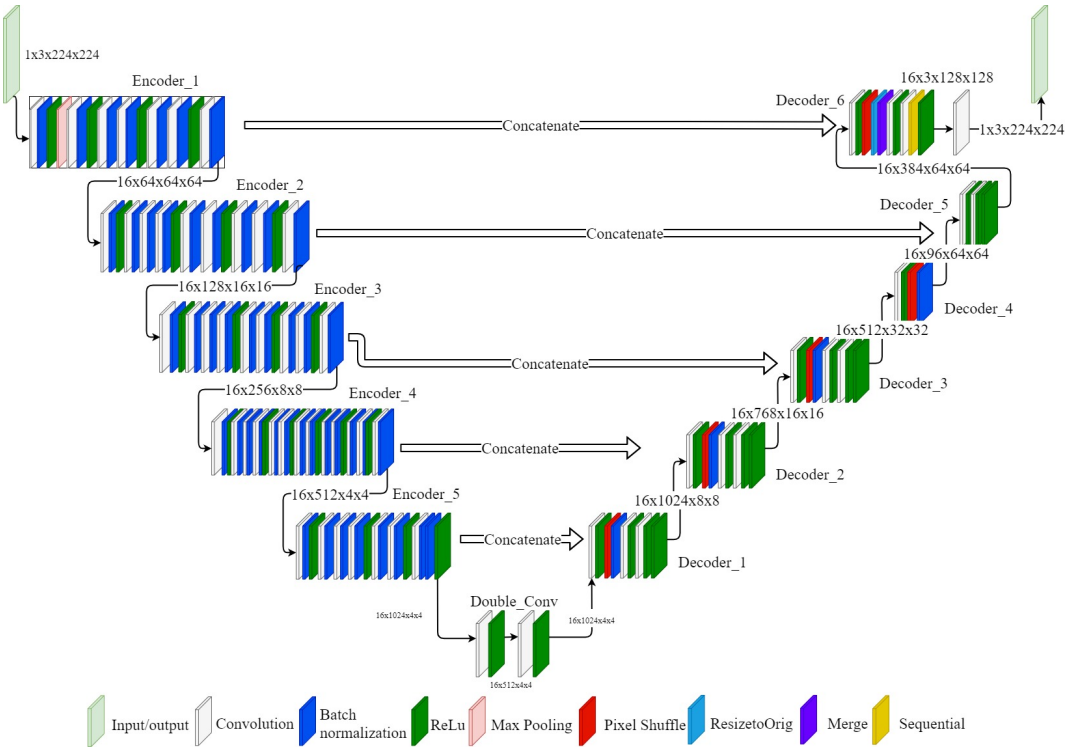


ResNet34+UNet: 由 ResNet 建構 encoder，並維持 UNet 作爲 decoder。我先建構 ResNet34，再將它使用於 UNet 的 forward 中。與 lab2 實做的 ResNet50 相比，ResNet34 最大的不同在於使用了 basic block 而不是 bottleneck block:



因此我先將 basic block 做成一個函數，在用迴圈的方式建構 ResNet34。其中我也加入了一個 list 用來記錄每個 basic block 的 output 做爲在 decoder 需要被

concatenate 的對象。在 encoder 和 decoder 之間，我加入了 bridge block，其中包含了卷積層、BN、ReLU、maxpooling 各一層，好讓 size 能夠對接上，並保留其特徵重要性。而 decoder 的部分如同前面 UNet 實作。



2.3 Anything more I want to mention

UNet 實做起來比 ResNet34 結合 UNet 好做，因爲 UNet decoder 和 encoder 構造差不多，但 ResNet 和 UNet 不太相同，尤其是 image size，爲了 concate 他們，需要事先計算確認兩邊的大小相同。我在這兩個架構中最後都沒有加入 sigmoid function，而是將它加在訓練或是測試中。

3 Data Preprocessing

3.1 How I preprocessed my data

一開始，我先確認資料集是否存在，若否，則下載並解壓縮。再來將資料集轉換成 OxfordPetDataset，將訓練資料和驗證資料作分割。這兩個步驟都是利用助教提供的程式碼實作。在 transforms 中，我將 image 和 mask resize 成 (256, 256)，並將他們轉成 tensor，以及對 image 做 normalization。

3.2 What makes my method unique

我做 normalization 所用的 mean 和 std 並不是常看到 ImageNet 的 mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]，而是在 utils 中自訂義的函數 mean_std()，並用它計算所有 train data 的平均和標準差去做 normalization。透過加入了 normalization，訓練出來的 test dice score 上升了約 0.03。

3.3 Anything more I want to mention

我有嘗試過使用 RandomRotation, RandomHorizonFlip, ColorJitter, RandomCrop... 等方法做 data augmentation，但訓練出來的結果十分的差，因此我沒有做任何資料增強。

4 Analyze on the experiment results

4.1 What did I explore during the training process

在訓練過程中，兩個模型的訓練時間差不多，且剛開始的 dice score 已經達到 0.7 以上，在 5 個 epochs 內就到 0.8，然而之後提升的速度變慢許多，尤其是 evaluation 開始震盪。我嘗試加入其他的 scheduler，但效果反而更差。藉由實驗，learning rate 設 0.0001 會比 0.001 來的更穩定，且並不需要更多的 epoch 進行收斂。

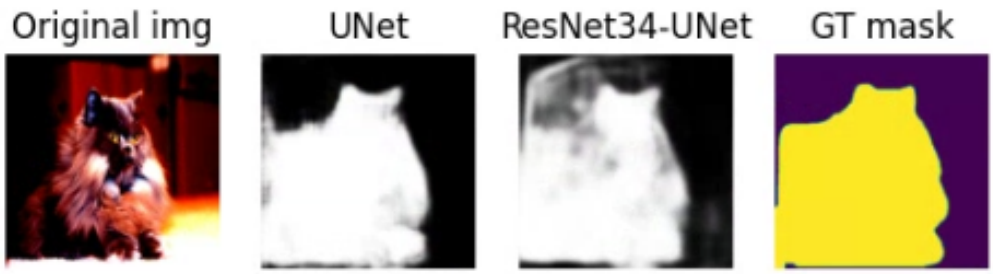
4.2 Found any characteristics of the data

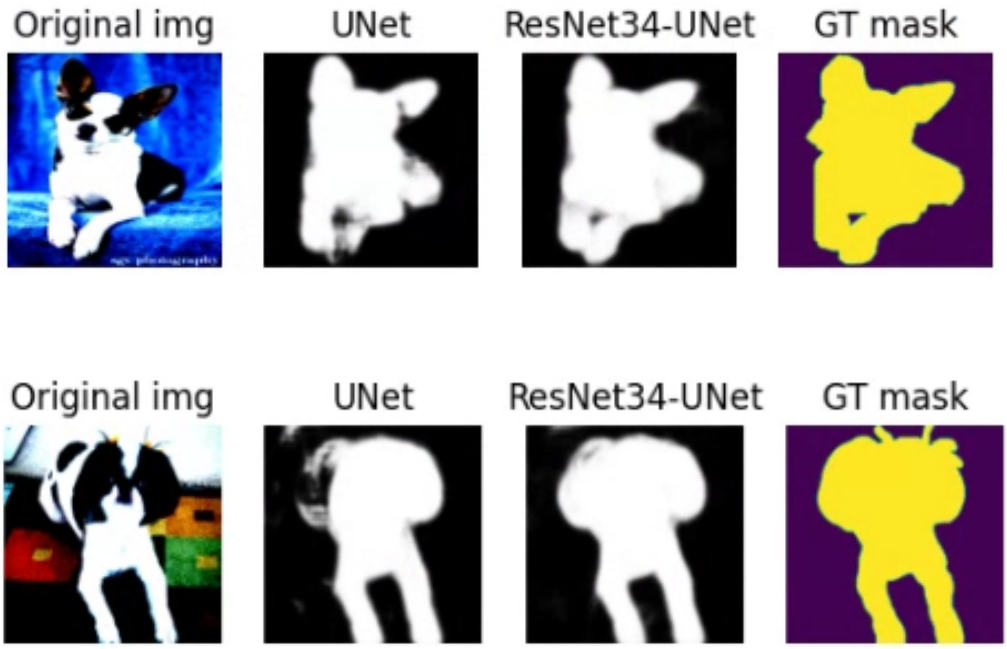
我發現此資料集的一些特色:

- 類別不平衡: 在 OxfordPetDataset 中，貓和狗的數量是不平衡的，這可能導致模型在狗的圖片上的表現比貓更好。
- 圖像大小和解析度: OxfordPetDataset 中的圖片大小和解析度有所不同，這可能會對模型的訓練和性能產生影響。
- 圖片中貓狗的部分所佔的比例多，不會導致背景過多而訓練成果不佳。
- 測試資料比訓練資料多

4.3 Anything more I want to mention

我在訓練過程中遇到一個問題是隨著 epoch 增加，gpu memory 使用量越來越多，導致我不管 batch size 設 16, 8, 4, 2，都在 epoch 10 左右就 out of memory。因此我只用 batch size = 1 去訓練。尚不清楚是參數設定的問題，還是其他問題導致，但 batch size 設 1 可能會讓訓練模型的標準差變大。以下是我的三張測試結果:





5 Execution command

5.1 The command and parameters for the training process

”python train.py –data_path ../dataset –epochs 30 –batch_size 1 –learning-rate 0.0001”
上面的各個參數前的 - 都爲兩個。

5.2 The command and parameters for the inference process

”python inference.py –batch_size 1 –data_path ../dataset –model ../saved_models”
上面的各個參數前的 - 都爲兩個。

6 Discussion

6.1 What architecture may bring better results

在我的實驗和測試中，ResNet34 + UNet 模型在 lab3 中些微的好一點，我認爲這跟 ResNet 強大的擷取特徵能力有關，它可以比 UNet encoder 有著更好的效果。以下是我最好的測試結果:

```
UNet dice score: 0.8874 | ResNet34 UNet dice scroe: 0.8899
```

雖然最好的結果相差不多，但在訓練過程中，ResNet34 + UNet 的架構一直都比 UNet 還來得好。

6.2 What are the potential research topics in this task

針對 lab3 的主題可能的潛在延伸研究，我分成以下幾點:

1. 泛化訓練：探索如何通過跨領域訓練技術來改進模型的泛化能力，使其在不同類別的寵物圖像上都能取得良好的效果。

2. 模型改進與優化：尋找適合的 encoder 或 decoder 與 UNet 做結合，以提高準確率、加速收斂速度、降低模型參數等。

3. 標籤效率：研究如何利用半監督學習或主動學習等技術，以更少的標註成本來訓練高性能的語意分割模型。

4. 眞實場景應用：將語意分割應用到眞實場景中，例如智能寵物監控、智能寵物玩具等，並進行實際效果評估和應用驗證。

5. 寵物姿勢和行爲分析：基於語意分割的結果，進一步研究寵物的姿勢和行爲分析，例如動作識別、姿勢估計等。

6.3 Anything more I want to mention

我在做 SWA 時有再用 swa_utils.AveragedModel 去實作 swa 的 model，並且也有存取此 model 的權重，但在測試時，若將此 model 加上.eval()，則 UNet 和 ResNet34 + UNet 的 dice score 分別爲 0.43 和 0，而沒加上.eval() 卻可以都達到 0.89。我在網路上找到可能的原因有幾項:

1. 測試時將 AveragedModel 設置爲評估模式可能會導致過度平滑的結果，因爲 AveragedModel 試圖將所有模型的預測結果統一化，這可能會降低模型的準確率。

2. 可能與 Batch Normalizaiton 和 Dropout 放置的位置有關。

3. batch size 太小，數據標準差大，所以使用 model.eval() 時，準確率會下降。

造成此問題的原因我尚不清楚，有待之後測試。
在 inference.py 中，我有加入儲存兩個模型所生成的 masks 的程式，其函數我寫在 utils 中，但存完我就將它 comment 掉。
我在 train.py 中存權重的 path 是我在實驗中設置的，如果有需要更改再請助教留意 168, 169 行，這部分不影響助教再次執行我的程式。
我將 train 和 valid 的 dice score 畫出，發現訓練過程 validation 震盪十分嚴重，這將會是我接下來去優化的目標之一。

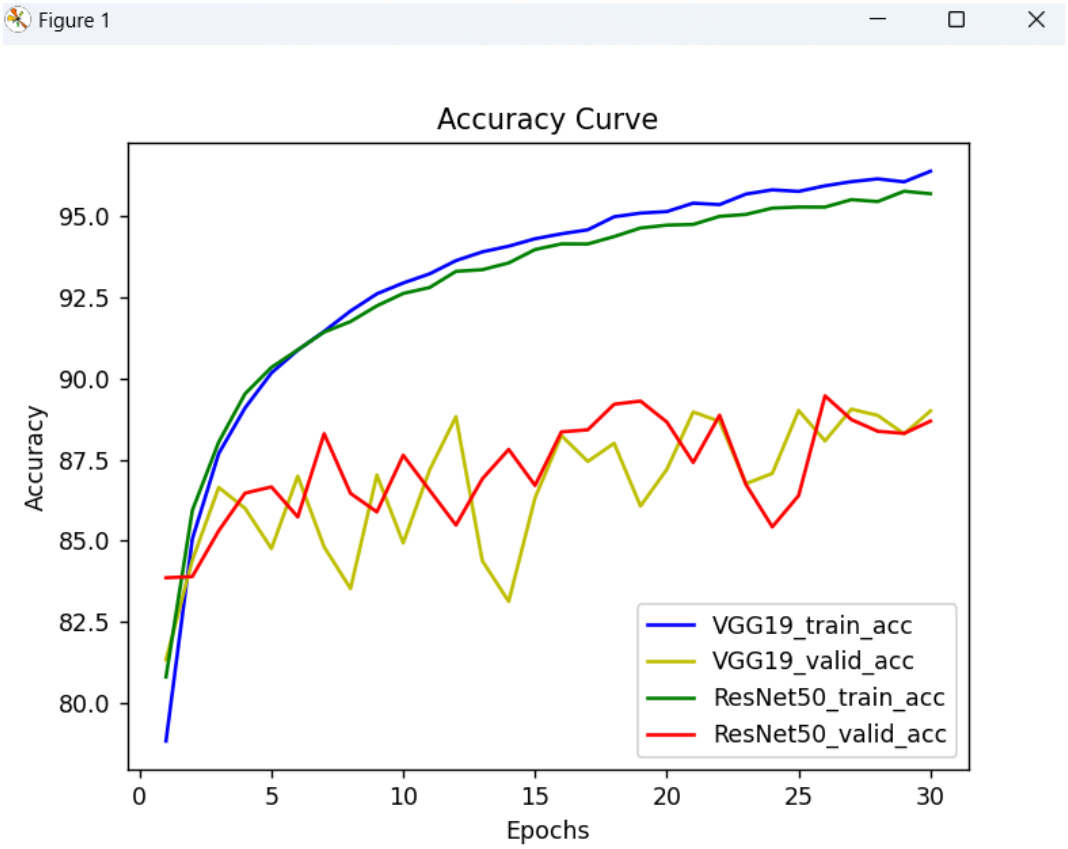


Figure 1
Accuracy Curve
Accuracy
Epochs
95.0
92.5
90.0
87.5
85.0
82.5
80.0
0
5
10
15
20
25
30
VGG19_train_acc
VGG19_valid_acc
ResNet50_train_acc
ResNet50_valid_acc