學號：B05902010　系級：資工四　姓名：張頌平

ML HW6 Report

(2%) 試說明 hw6_best.sh 攻擊的方法，包括使用的 proxy model、方法、參數等。此方法和 FGSM 的差異為何？如何影響你的結果？請完整討論。(依內容完整度給分)

Ans:
在我的 hw6_best.sh 當中，我使用的攻擊方法為I-FGSM(iterative Fast Gradient Sign Method)。它的架構跟一般的FGSM類似，都是會將epsilon乘以loss對data的gradient後對每一個dimension取sign，加上原圖後得到攻擊完後的照片，只是I-FGSM不像一般的FGSM只跑一次就結束，I-FGSM會多攻擊幾次model；這邊實作的方法是，在第一次攻擊以後的每次攻擊，都要確保攻擊後圖片和原圖不會相差超過epsilon，如果超過，代表那個維度進行了超過一次同方向的攻擊，因此就讓那個維度的值不變，若是沒有超過，則維持攻擊後的結果。多次的攻擊有助於讓照片多次地往不同loss的gradient的反方向移動，這樣更有機會使得照片在多維空間中精準地往不同的方向找尋最有效的攻擊結果。

而我在自己本機測試時，在epsilon=0.02時是可以達到準確率1.0且L-inf=0.925，但有可能因為儲存image的函式會讓照片有些許誤差，因此實際實做時，我將攻擊完且儲存過後的照片再進行一次攻擊，若是辨認不出照片分類，則直接不攻擊，若是仍然分類正確，則再攻擊後進行儲存，發現會有3張圖片是原本攻擊完顯示辨認不出分類，但儲存過後又可以分對了，(不知道有沒有除了save image更好的儲存方法QQ)，因此最後的L-inf為0.94，而再多一次攻擊後，全數照片都無法被model分出來原本的分類了。

最後reproduce的參數和實際準確率如下：
proxy model: DenseNet-121(因為透過實驗發現可能為judge的black box model，下一題會更進一步說明)，
iteration times: 20
epsilon: 0.017

Acc & L-inf：
FGSM: 0.915, 5.550
I-FGSM: 1.000, 0.94

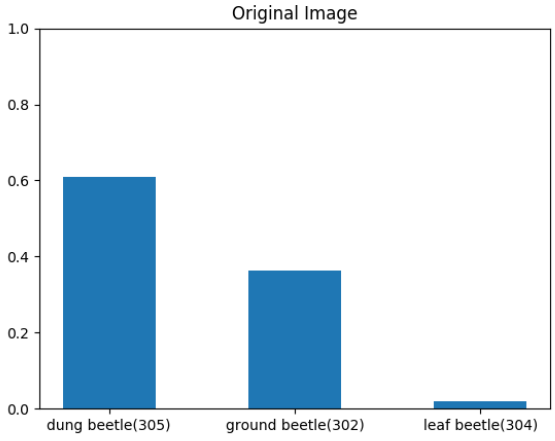
(1%) 請嘗試不同的 proxy model，依照你的實作的結果來看，背後的 black box 最有可能為哪一個模型？請說明你的觀察和理由。

Ans:
在作業的slide當中，有提到Black box可能的模型種類，為pytorch當中的6種模型，因此在其他參數皆一樣的情況下(epsilon=0.1)，我使用一般的FGSM(Fast Gradient Sign Method)分別對每個model都進行攻擊，並將其攻擊過後的照片傳至JudgeBoi，其結果(success rate)如下：

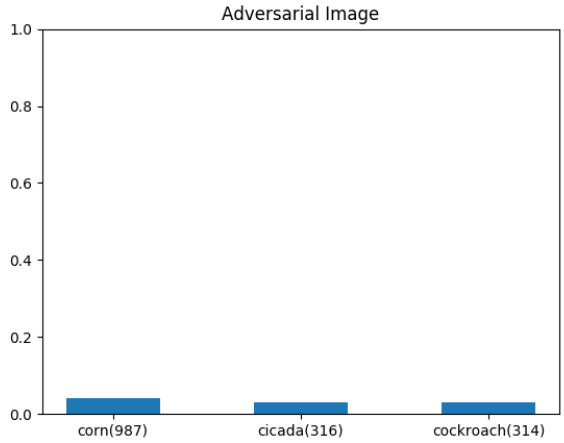
VGG-16: 0.315
VGG-19: 0.275
ResNet-50: 0.420
ResNet-101: 0.360
DenseNet-121: 0.915
DenseNet-169: 0.420

從以上結果我們可以得知，攻擊表現最好的應該就是我們這次Judgeboi系統所使用的black box model - ***DenseNet-121***

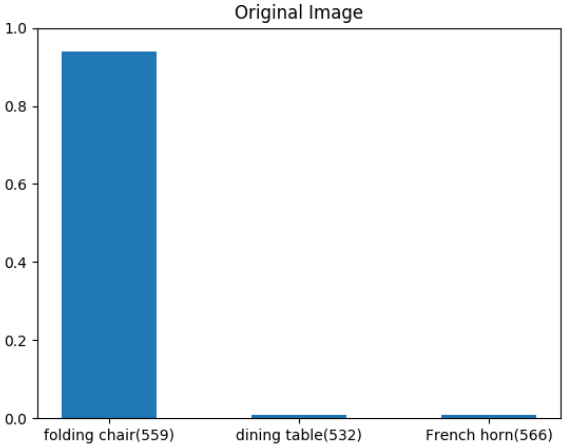
(1%) 請以 hw6_best.sh 的方法，visualize 任意三張圖片攻擊前後的機率圖(分別取前三高的機率)。



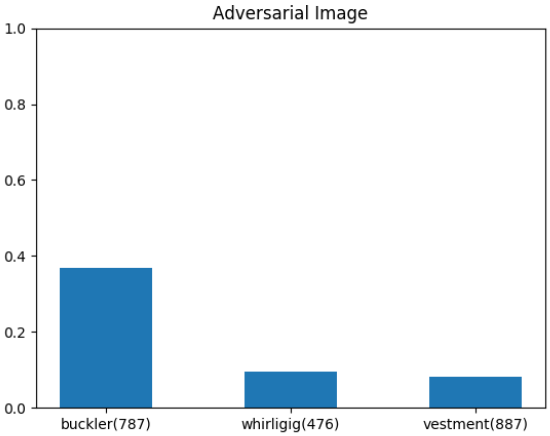
dung beetle 60.88%



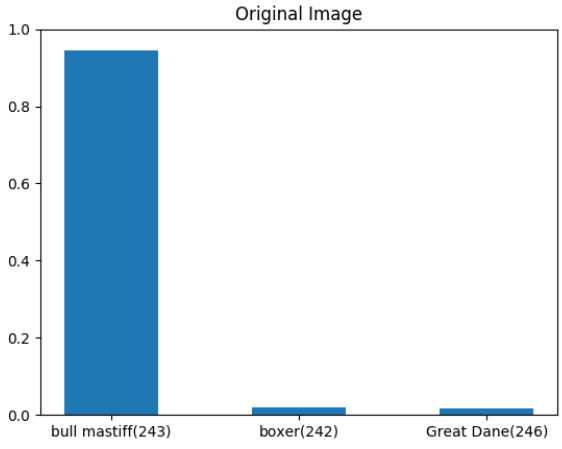
Corn 4.22%



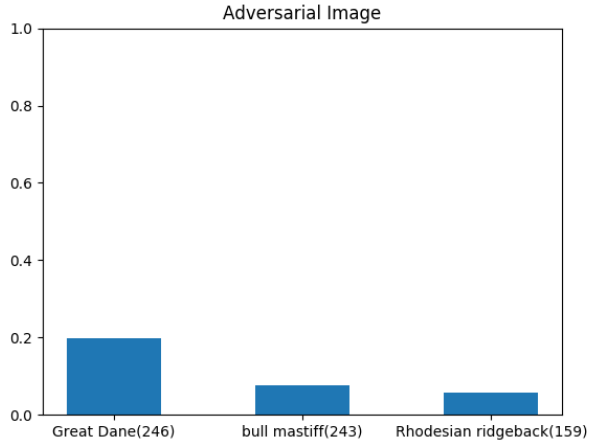
Folding chair 93.92%



Buckler 36.72%



Bull Mastiff 94.59%



Great Dane 19.72%

原圖：



(2%) 請將你產生出來的 adversarial img，以任一種 smoothing 的方式實作被動防禦 (passive defense)，觀察是否有效降低模型的誤判的比例。請說明你的方法，附上你防禦前後的 success rate，並簡要說明你的觀察。另外也請討論此防禦對原始圖片會有什麼影響。

Ans:

我使用的方法是老師上課影片所提到的 “Randomization at Inference Phase”，在原本被攻擊的照片當中，隨機地resized照片的大小後並且對其進行小幅度的縮放和裁減，並且加上一些雜訊，這樣做的目的是因為原本被攻擊的圖片其實和原圖會非常接近(因為不想被發現是攻擊過後的圖片)，因此圖片只能在某一個方向會是能成功攻擊的，而當我們做了一些隨機的擾動後，原本會被攻擊的那個方向因著隨機地縮放照片後，就有可能失效了 ，圖片也因此就能預測成功。實際實作是利用torchvision裡面的RandomResizedCrop，並且在最外圍加上padding，讓外圍的pixel等於0(需要注意的是擾動並不能夠太大，不然會使得圖片和原本相差很多而預測錯誤)，即可達成上面所說的效果。

實際參數如下：
crop scale: (0.8, 1.0)
aspect ratio: (0.8, 1.2)
padding: 3層

原本經由上面hw6_best.sh所攻擊的照片的準確率為0，在進行passive defense後，有105張照片重新會預測正確，準確率為0.525(attack success rate為0.475)，在實驗中也有嘗試將padding的層數由2~10進行調整，發現在padding數過多的情況，會因為影響的部分資訊喪失而使得準確率下降，詳細準確率如下：

padding=2: 0.52

padding=3: 0.525
padding=5: 0.5
padding=10: 0.485

以下為兩組經過passive defense後和原本被攻擊的圖的比較，可以發現經過防禦後的照片因為有做resized跟crop的動作，因此整體圖變得更大，或是更為集中(也有可能使得資訊被切掉，像是下圖中鳥的頭)，再加上最外層有部分的padding(邊框的灰色部分)，因此削弱了原本攻擊的地方，使得照片有可能重新預測正確。