ML2019SPRING HW5

B06902074 資工二 柯宏穎

1. 試說明 **hw5_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。

我的$best$一樣是使用$FGSM$去做攻擊，使用的$proxy\ model$為$ResNet50$。與一般$FGSM$的差別是，我嘗試去分配每張圖片的變化量，因為$L-inf$的計算方式為取所有照片的平均值，原本的$FGSM$是設定一個$epsilon$，每張照片就以這個為bound，儘可能地作出最大的變化。但實際上，有些照片可能只要有些為的改變，就可以達到有效地攻擊，有些卻要加上超出規定的$L-inf$許多的雜訊才能攻擊。我做了簡單的實驗，每次的$epsilon$為1，若攻擊失敗就再增加。我們可以發現，多數的圖片只要一次的$FGSM$就能成功擊破，那我們就可以讓比較難攻擊的照片，做多一點的改變，已達到100的成功率。

2. 請列出 **hw5_fgsm.sh** 和 **hw5_best.sh** 的結果 (使用的 **proxy model**、**success rate**、**L-inf. norm)**。

	model	success rate	L-inf
fgsm	ResNet50(pytorch)	0.925	5.000
best	ResNet50(pytorch)	1.000	2.695
fgsm	VGG19(keras)	0.445	4.975
deepfool	ResNet50(keras)	0.330	5.520

$Resnet50(pytorch)$的$\mu=[0.485,\ 0.456,\ 0.406]$, $\sigma=[0.229,\ 0.224,\ 0.225]$ ($torchvision.transforms.Normalize(\mu,\sigma)$)

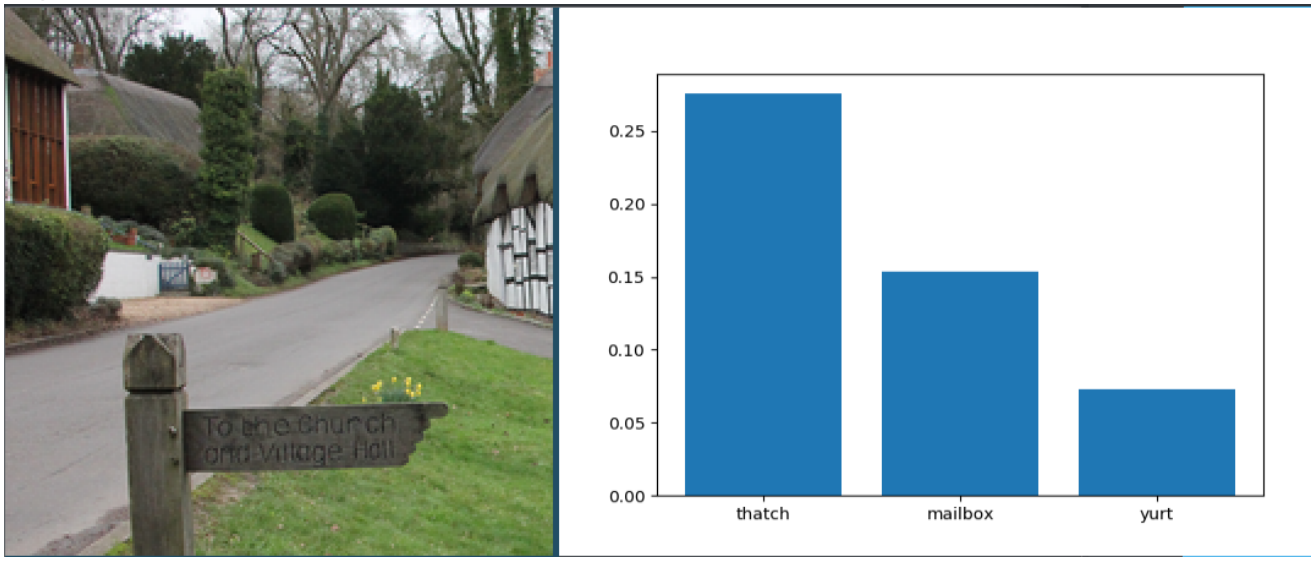
$Resnet50(keras)$與$VGG19(keras)$的$\mu=[103.939,\ 116.779,\ 123.68]$ ($keras.applications.vgg19/resnet50.preprocess\_input()$)

3. 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。

我最初使用的是$keras$的原生模型，分別嘗試過了$VGG16$, $VGG19$, $ResNet50$ 與 $DenseNet121$，進行$one\ punch$攻擊的結果，$L-inf$ 大約5時，準確率大概落在0.45左右，一直都不是太理想的數值。之後也去實作了$deepfooling$，嘗試找到最好改變的label並做最大的改變，可惜成效依然不章。且自行用該模型$re-predict$的結果，與上傳結果相距甚遠，若是使用相同的模型，理論上計算出來的值應要差不多。　後來才知道，原來不同的套件，雖然model名字相同，依然有所差別。我覺得最大的可能原因就是在predict前的$preprocesseing$，預處理過後的圖不同，即便判斷的方法一樣，也會得出不一樣的結果，model的weight也會有所差別。後來使用手把手的範例下手，發現$pytorch$的$ResNet50$ 有非常好的performing，線下計算結果也與線上judge一模一樣，單單使用一次性的$FGSM$，在$L-inf$為5時，就有九成的成功率，效果十分顯著。

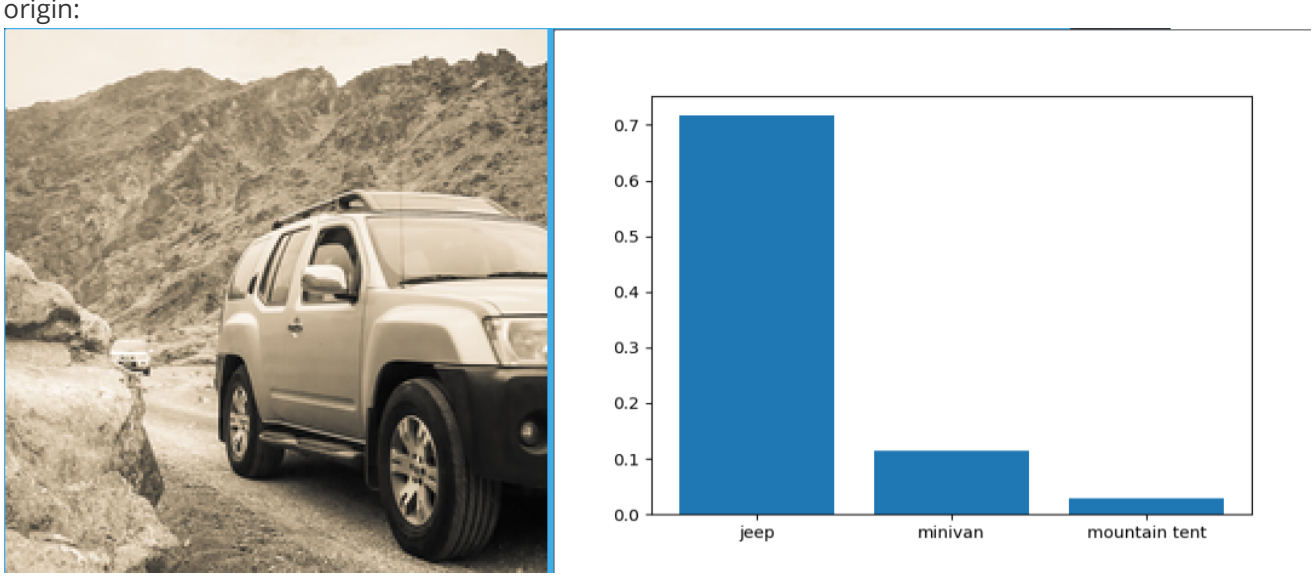
4. 請以 **hw5_best.sh** 的方法，**visualize** 任意三張圖片攻擊前後的機率圖。

origin:

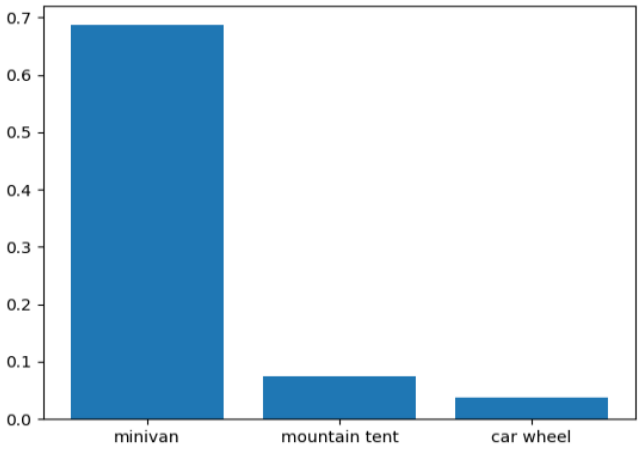


attacked:

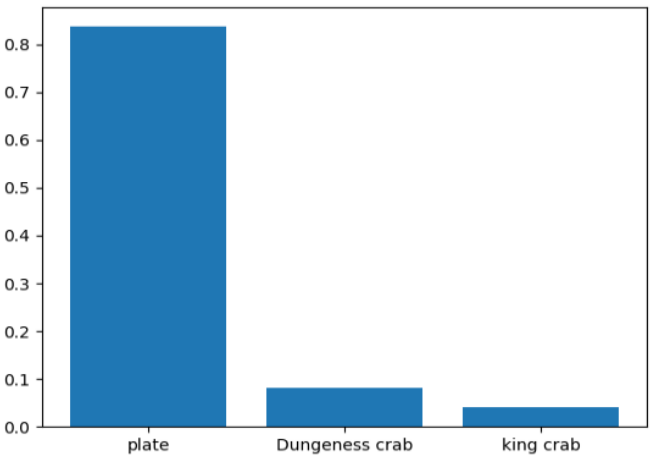
origin:



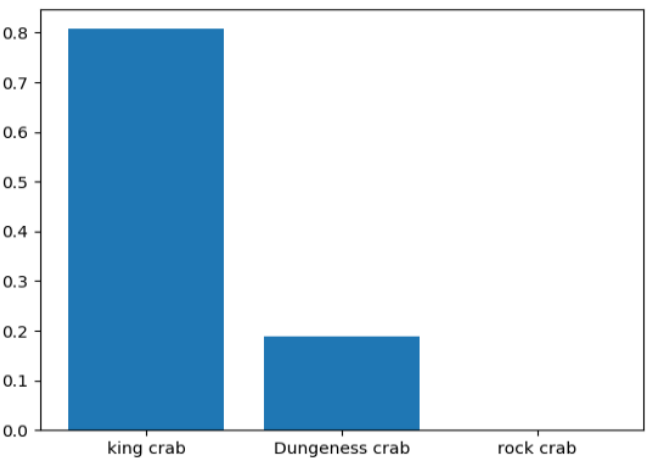
attacked:



origin:



attacked:



5. 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。

我使用$opencv$這個套件來進行$smoothing$，成功率從1.000下降到0.320。我們做的攻擊，主要就是去強化某些特徵，讓$model$預測失準。而$smoothing$就是去平均化整張圖，減緩我們對特定像素的改變。　不過做完$smoothing$後，整張圖看起來會模糊化，$L-inf$也會上升許多。因為他會將每個像素的權重比例降低，以降低每個像素的差距。當我們$smooth$太多，每個像素都很接近時，看起來就會是一團白白糊糊的東西，此時攻擊效果可能也很好，但我們就可以非常明顯地看出不是原本的圖片了，不會達到我們所預期的效果。