作業：提取像素鴨子

從無人機中取出的養鴨場圖像，使用貝氏分類器從圖像中提取鴨體的像素

一、進行步驟

1、

手動收集鴨體樣本像素。從養鴨場圖像收集包含鴨體的正樣本，非鴨子的負樣本(圖樣內無任何鴨體，例如水塘、石子地、青草、馬路….等)。再依據程式中需要樣本大小切割成2x2、4x4、8x8…等大小圖片。

2、

使用鴨體圖片的三維（[紅色，綠色，藍色]）特徵向量像素和非鴨體像素，建立兩個高斯概率似然模型，模型參數（平均向量μ和協方差矩陣Σ）通過從最大似然估計導出的公式估計。

假設若特徵向量是獨立的，可以把鴨體圖片的三維看成是三個獨立的高斯分佈，𝑃（𝐱|ω1）=P(x|B) \* P(x|G) \* P(x|R)

假設每一個類別的資料均是由 d 維的高斯機率密度函數（Gaussian probability density function）所產生：：

gi(x, , ) = (2)-d/2\*det()-0.5\*exp[-(x-)T-1(x-)/2]

其中  是此高斯機率密度函數的平均向量（Mean vector）， 則是其共變異矩陣（Covariance matrix），我們可以根據 MLE，產生最佳的平均向量  和共變異矩陣 。

3、

導出兩個高斯分佈模型後，𝑃（ω0|𝐱）和𝑃（ω1|𝐱），可以依貝葉斯決策規則來對每個像素進行分類。假設𝑃（ω0）=𝑃（ω1）。因此只需要比較貝氏決策規則時的似然值。𝑃（𝐱|ω1）> 𝑃（𝐱|ω0）保留鴨體的像素，若𝑃（𝐱|ω1）<𝑃（𝐱|ω0）則將該像素替換成黑色像素，以此循環比對整張圖片。

4、

依據步驟3輸出新的圖像保持每個鴨體像素並替換所有非鴨體像素為黑色像素。

二、執行的結果

|  |  |  |  |
| --- | --- | --- | --- |
| 樣本大小(pix) | 2X2 | 4X4 | 8X8 |
| 花費時間(s) | 15546s | 7494s | 4039s |
| 正樣本數 | 2386 | 494 | 142 |
| 負樣本數 | 6320 | 980 | 432 |
| 正樣本識別率 | 0.9793 | 0.9935 | 0.9791 |
| 負樣本識別率 | 0.9937 | 0.9931 | 1 |
|  |  |  |  |
| 結果 |  |  |  |

測試環境：(桌機)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OS | CPU | Memory | Python | Anaconda3 | IDE |
| Win7 X64 | Phenom II X3 B73 | 16G | 3.61 | 4.4.0 | PyCharm |

三、討論

1、

以資料結構來看養鴨場為N\*M大小的矩陣，以2個 For循環來計算，其時間複雜度為O(n2)。其程式結構較易懂

for j in range(yp\_min, yp\_max, h):  
 for i in range(0, xp\_max, w):

若改成以While迴圈，While 本身為循序方式，其時間複雜度為O(n)，但是While 內部還是O(n)，相乘後複雜度還是為O(n2)。

while y + h < yp\_max:  
 ……   
 x = x + w  
 if x + w >= xp\_max:  
 x = 0  
 y = y + h

若改成以執行緒方式，生產消費者模式去執行，將圖片分成多個區域，分別去判斷是否為鴨體像素，且多個執行緒將非鴨體的座標存入一個共用Queue中，讓另一個執行緒替換Queue中非鴨體座標修改成具有黑色的像素。

其結果仍不理想，其時間複雜度還是O(n2)，只是拆成數個去執行罷了。

r1 = threading.Thread(target=read\_full\_duck, args=(img, 0, 4000, xp\_length, w, h, p\_mu\_hat, p\_sigma\_hat, n\_mu\_hat, n\_sigma\_hat))  
r2 = threading.Thread(target=read\_full\_duck, args=(img, 4000, 8000, xp\_length, w, h, p\_mu\_hat, p\_sigma\_hat, n\_mu\_hat, n\_sigma\_hat))  
r3 = threading.Thread(target=read\_full\_duck, args=(img, 8000, yp\_length, xp\_length, w, h, p\_mu\_hat, p\_sigma\_hat, n\_mu\_hat, n\_sigma\_hat))  
r7 = threading.Thread(target=drow\_full\_duck, args=(img1, w, h))

2、

依照執行的結果來看，發現取樣像素越大其執行時間越短，若以此來看，若取樣像素太小，其優點較能保持鴨體形狀，其缺點雜訊干擾大容易誤判及執行時間過長。取樣像素太大，其缺點鴨體形狀不易保持，其優點是執行時間較短及雜訊干擾小。

四、總結

這次作業檢討有下列缺失：

1. 執行時間過久且程式碼不夠精緻

在寫程式方面是先求有再求好，先把程式架構規劃出來，再慢慢一步一步將缺失補上，等程式確定能運行再去修正或替換某些功能，另外Python是初學所以很多功能不熟悉，再加上平時對於Numpy、Scipy、sklearn...沒有用過，以至於程式寫法以土法煉鋼完成，程式沒最佳化，所以執行時間過久。

若教授在出題時也能推薦Python套件中可用的Function，那在程式寫作方面應該可以減輕不少負擔。

1. 對於作業要求不夠瞭解

剛開始看到題目會緊張，如同期中考一樣，看到題目好像似乎懂了會寫，而實際上真的要去寫去做完全沒思路沒方向不知如何解決問題，有問題也不知道該找誰去問去討論，目前看來只能多寫多做一些題目才能改善，除了基本理論要有外，來這裡是要學習一個方法論，才能將理論的東西給實做出來。