SLT Term Project Report

Car Detection Using Cascaded AdaBoost

u9562171, 雷禹恆

1. 專題介紹：

本次期末專題是我大學部實作專題的內容，指導教授是許秋婷老師。計畫目的是視覺輔助，希望透過電腦視覺及機器學習的技術，協助校園內的視障師生，使他們在日常活動及行走時，更能辨別及應付道路上的突發狀況。目前專注在車輛偵測，我特別強調是「來車」的偵測，因爲開走的車和橫向經過的車對於行人威脅不大，不太符合視覺輔助的目的。

而爲了簡化問題，我們將限制在國立清華大學的室外校園，並將天候條件限制在「白天、好天氣」的理想情況。計畫最終目標是利用筆記型電腦配上網路攝影機，以模擬出可攜式的視覺輔助設備。

2. 相關研究：

在人形、人臉偵測的領域，近年來有不少文獻被提出，同樣的道理套用在車輛偵測上也是可行的。Viola & Jones在CVPR'01曾經提出過一個物件偵測的訓練架構，把AdaBoost的強分類器串成一個cascade的decision tree；而在feature部分則是使用intensity-based的rectangle (Haar-like) features。優點是偵測快速，缺點是rectangle features本身不夠好。

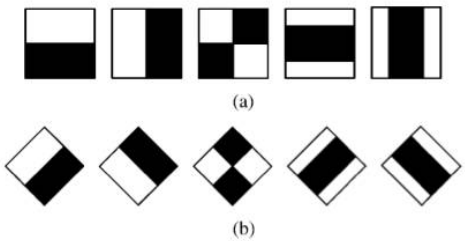
另一方面，Dalal & Triggs在CVPR'05曾經提出了gradient-based的HOG (Histogram of Oriented Gradients) features，是將Lowe (IJCV'04) 提出的SIFT文獻後半段獨立出來使用；而training部分則是直接使用現成的SVM。優點是HOG準確率高，缺點是HOG features的維度太高，動輒高達數千維，如果是要在影像裡跑sliding window找尋物件的話，速度會太慢，在影片方面的應用會更不理想。

本專題實作的主要參考文獻是國內陳昱廷和陳祝嵩(Chen & Chen)在2008年所發表的"*Fast human detection using a novel boosted cascading structure with meta stages*"，可以視為一個結合兩者優點的方法。Feature部分intensity-based和gradient-based的都使用來加強準確率，而training部分使用cascaded real AdaBoost來達到快速偵測。

3. Feature採用：

對於影像中的任一矩形區塊(block)，文獻提出的Feature pool包含三種類型的影像特徵：1. Rectangle features (**REC**)、2. Edge Orientation Histogram (**EOH**)、3. Edge Density (**ED**)，分述如下：

- Rectangle features (**REC**)：

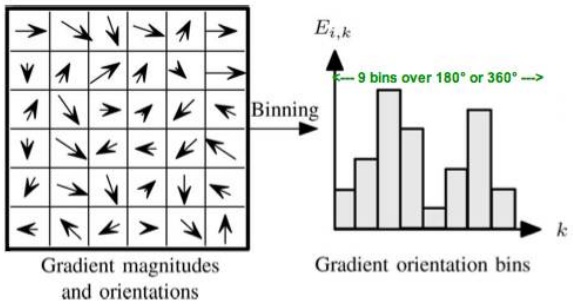


如上圖所示，REC feature計算的單純是pixel intensity相減，例如白色區域減黑色區域，寫成數學型式可表示為式(1)：

$$F_{i,r}^{REC} = White\ pixel\ sum - Black\ pixel\ sum \tag{1}$$

下標i代表block B_i，下標r = 1 ~ 10代表上圖REC feature的子類型。(a)部分的五種和Viola & Jones所使用的差不多，而(b)部分的五種是Lienhart & Maydt在2002年提出的45°傾斜版本，有助於偵測的準確率。計算pixel sum可使用integral image的技巧來加速，針對(a)的一般矩形和(b)的傾斜矩形都有其計算法。

- Edge Orientation Histogram (**EOH**)：



對於block B_i，我們可算出各點的水平方向(G_x)和垂直方向(G_y)的gradient值，再透過式(2)、式(3)的關係式得到如上圖左的gradient量值(m)和方向角(θ)：

$$m(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \tag{2}$$

$$\theta(x,y) = \tan^{-1}\frac{G_y(x,y)}{G_x(x,y)} \tag{3}$$

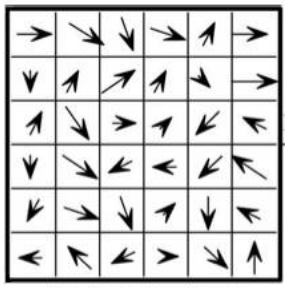
文獻使用的是unsigned gradient，把第三、第四象限的方向角(180°~360°)直接視為第一、第二象限的對應角度(0°~180°)，並且將0°~180°的角度範圍分成K個bin，例如K = 9，每個bin 20度，統計出histogram如上圖右。

EOH所衡量的是在B_i 中每個gradient orientation bin值($E_{i,k}$)，佔所有bin值總和的比例，如式(4)所示：

$$F_{i,k}^{EOH} = \frac{E_{i,k}}{\sum_{j=1}^{K} E_{i,j} + \epsilon} \tag{4}$$

下標k = 1 ~ K代表histogram的第k個bin，ε代表防止除數為0所加上的小數字。如同integral image，EOH features也可用integral histogram的技巧來加速計算。

- Edge Density (**ED**)：



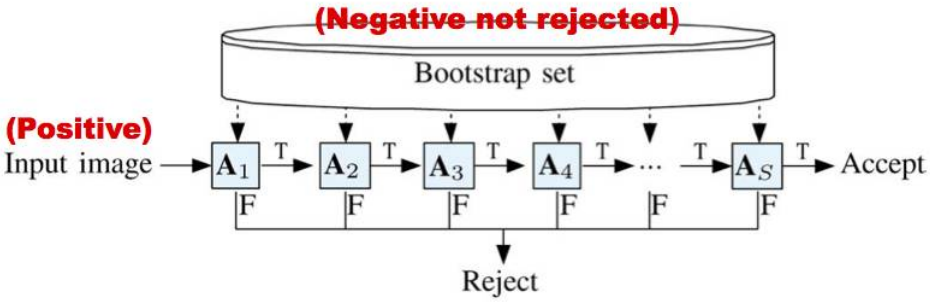
Gradient magnitudes

ED feature只純粹在乎gradient的量值。對於block B_i，ED的衡量方法是B_i 範圍內的平均gradient量值，如式(5)：

$$F_i^{ED} = \frac{\sum_{(x,y)} m(x,y)}{area(B_i)} \tag{5}$$

總計對於一個block，在上述情形中可取出10個REC，9個EOH，和1個ED共20個features。對於整張影像，我們嘗試所有可能的block大小和block位置，每次都去擷取該block的20個特徵值。以文獻為例，一張影像共有6948 x 20 = 138,960個特徵。維度非常高，但重點在於接下來的訓練過程「一次只使用一個feature」，也就是AdaBoost弱分類器是作1-D classification。我的實作方式是用最簡單的分別對兩群數值分布「取平均」，再將決策門檻(decision boundary)訂為pos_mean和neg_mean的中間值。

4. Training架構：



以上為訓練和偵測的架構圖，是一個cascade的decision tree。在sliding window尋找車輛的應用中，大部分的windows都是negative，因此我們的目標是在這些時候能快速reject掉negative windows，也就是在cascade的前面幾個階段就能判斷出是negative，真正的少數positive資料則必須通過所有的判斷點才能被accept。以下分作訓練和偵測兩部分來說明：

- 訓練(training)：

上圖中的每一個方框A_i 都代表一個AdaBoost的強分類器，左方代表positive資料，上方的bootstrap set代表negative資料。此種訓練架構需要非常大量的negative資料，假設positive資料量N_1，negative資料量N_2，則N_2必須是N_1的十多倍至數十倍，例如$N_1 = 500$，$N_2 = 10{,}000$。

每一個AdaBoost階段都會訓練出一個AdaBoost強分類器。在每一個階段，positive

的N_1筆資料都會使用到，negative部分則只隨機選出N_1筆「尚未被淘汰掉(not rejected)」的資料來訓練，來確保正、負的資料量一致。強分類器訓練完後，能成功被正確分類的true negative將從bootstrap中被「淘汰(reject)掉」，之後的AdaBoost階段將不再被列為候選。這樣做的道理，是因為能在前面的訓練階段就被分類正確的negative資料，代表「已經被學會了」，是相對簡單的例子。後面的訓練階段會針對還學不會的相對「困難」例子來加強。

訓練過程會一直持續到N_2筆negative資料幾乎被reject完，以致於不到N_1筆可被選出為止。或是因整體的false positive rate達到所設定的目標(F_target)而結束。

- 偵測(detection)：

如同本節一開始的敘述，測試資料從A_1開始經過一連串AdaBoost強分類器的判斷。只要其中一個A_i判斷為negative，程式就判斷該張影像為negative並結束（較快）；只有在所有的A_i都判斷為positive時，程式才將該張影像判斷為positive（較慢）。

5. Training演算法：

- 參數：

```
f_target = 1e-20;   // 整體 false positive rate 的目標值。
d_minA = 99.9%;     // 每階段最小可接受的 detection rate，差 0.1%是預防 outliers。
f_maxA = 50.0%;     // 每階段最大可接受的 false positive rate。
```

- 演算法：

```
while (F_current > F_target) do {
  (1) Learn AdaBoost stage A(i,j) {
    Randomly select N1 non-rejected negatives from bootstrap set;
    Set f_local = 1.0;
    while (f_local > f_maxA) do {
      [1] Add the weak classifier to A(i,j) that minimizes error;
      [2] If necessary, modify threshold of A(i,j) to fulfill d_minA;
      [3] Update f_local of A(i,j);
    }
    F_current = F_current * f_local;
    Reject the negatives that were correctly classified;
  }
  (2) Learn Meta stage M(i) ... skipped
}
```

- 說明：

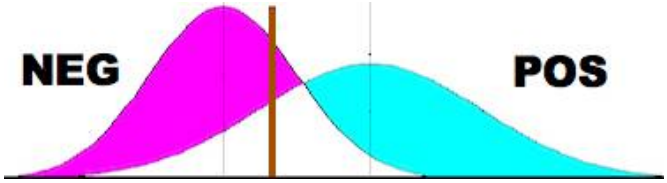
f_local代表在該AdaBoost階段的局部false positive rate，必須小於等於f_maxA才會結束此階段。

添加弱分類器的部分，我的實作和課堂上傳統的discrete AdaBoost一樣，有所謂的資料權重，和對資料加權的“weighted error rate”。每次添加時選出表現最佳的(block, feature)加入強分類器，接著依照傳統方法算出該弱分類器的權重，並更新資料權重。最後的強分類器判斷結果可視為各個弱分類器的「加權投票」如式(6)：

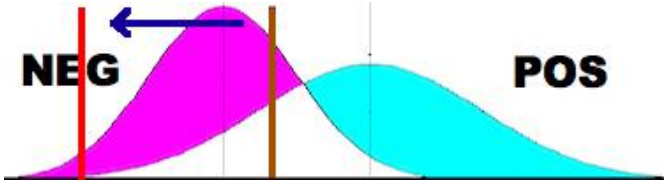
$$H(x)=sign\left(\sum_{t=1}^{T}\alpha_t h_t(x)-threshold\right) \tag{6}$$

threshold代表一個調整結果的門檻值，每個強分類器都有一個threshold。

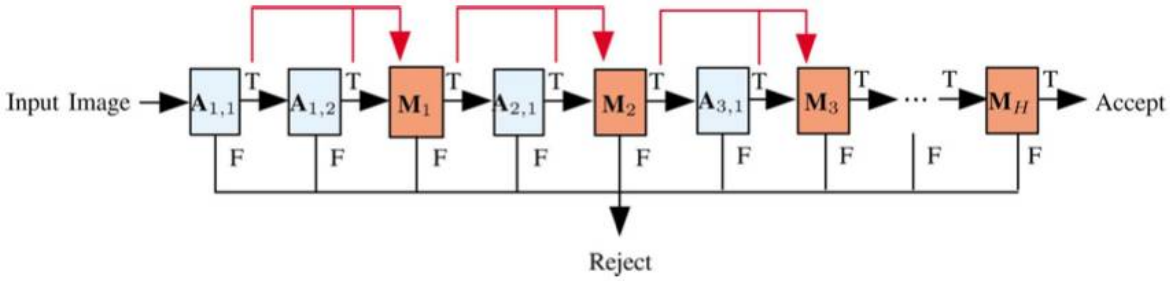
調整threshold使滿足d_minA的部分，若去掉式(6)的threshold，資料分布情形可能如下圖：



棕線代表的數值是0，此時positive資料無法被分到99.9%正確。於是我們調整強分類器的判斷結果，將決策界線從0移至紅線，使得positive資料可以99.9%被分對（滿足d_minA），如下圖。這個調整的量(負值)就是該強分類器的threshold，偵測的時候該強分類器也是依照threshold調整判斷結果。



Learn Meta stages的部分被我省略掉了，因為細節複雜。其實文獻中所訓練的不僅有AdaBoost階段，也有在中間安插「Meta階段」。而文獻採用的AdaBoost不是傳統的discrete AdaBoost，而是real AdaBoost，並根據此算出強分類器的confidence value。Meta階段就是根據前面數個(例如2個)階段的confidence values（每筆資料有2個confidence values，組成一個input vector），然後用現成的SVM訓練出一個2-D分類器，這就是一個Meta階段。相關的概念圖如下：



6. 實驗結果與討論：

本專題的程式碼等檔案資料放在Google Code的專案托管下，網址是：
http://code.google.com/p/candy2009

首頁包含了最新進度的demo影片。未來有新的訊息或文件也會公布在網站上，最後會放上編譯好的跨平台可執行檔(Mac, Windows, Linux)。

開發環境使用C++語言，配合OpenCV 2.0 library, C interface。作業系統是Mac OS X 10.6，IDE是Apple Xcode。機器是Apple MacBook with Intel Core 2 Duo CPU, 3GB RAM。

Data set：為了針對國立清華大學的校園，所有的車輛、非車輛照片皆由自己拍攝。為了符合視覺輔助的目標，車輛照片一律為「正面」或是「有斜角的正面」。目前共收集107張positive，199張negative資料，並用DV拍下數十分鐘的校園行走影片。Detection window size定為128 x 128，因此positive影像皆手動切為128 x 128，並用自己的程式把199張negative影像切成9355張128 x 128的小影像。

先前使用的是無任何library的C語言，僅能處理BMP圖檔，但offline訓練已經寫完。為了相容更多影像、影片格式，並實現出有GUI的偵測系統，甚至使用網路攝影機作為輸入，必須將舊的程式碼重新「翻譯」成符合OpenCV使用方法。幾天前已大致上完成了程式碼搬遷，並寫出了很基本的單一影像，無sliding window的cascaded AdaBoost classification。程式執行介面如下圖：

對於判斷為positive的影像，在命令列上會顯示出一個參考的分數（暫定為所有H(x)的和），判斷為negative的影像，會顯示出在第幾個階段被reject的。目前進度的程式demo影片見youtube網址：

http://www.youtube.com/watch?v=BqW82ovKoDc

因為時間不足，目前的訓練部分省略了不少細節，不少是重要的細節，所以表現差強人意。以下是幾組結果：

Model 檔案	測試資料	false positive rate	false negative rate
model_1000.txt	直接用訓練資料	7.00% (7 / 100)	39.3% (42 / 107)
model_3000.txt	直接用訓練資料	8.00% (8 / 100)	35.5% (38 / 107)
model_3000.txt	INRIA data set	4.00% (6 / 150)	無 positive 資料

model_1000.txt 由 107 筆 positive 對上 1000 筆 negative 資料訓練，共包含 14 個 AdaBoost階段；model_3000.txt是由107筆positive對上3000筆negative資料訓練，。因

為資料量明顯不足，前兩次實驗直接把訓練用的positive，negative資料拿來測試，發現reject negative的表現不錯，但accept positive的表現很差。第三次實驗用了INRIA data set裡的negative資料，結果表現反而更好，代表系統對於reject negative有一定的能力。

速度部分目前為止看起來不錯。對10筆資料的feature extraction計時如下：

```
pos_107/train001.JPG ... Running Time: 0.002313 seconds.
pos_107/train002.JPG ... Running Time: 0.002889 seconds.
pos_107/train003.JPG ... Running Time: 0.002306 seconds.
pos_107/train004.JPG ... Running Time: 0.002405 seconds.
pos_107/train005.JPG ... Running Time: 0.002435 seconds.
pos_107/train006.JPG ... Running Time: 0.002431 seconds.
pos_107/train007.JPG ... Running Time: 0.002363 seconds.
pos_107/train008.JPG ... Running Time: 0.002568 seconds.
pos_107/train009.JPG ... Running Time: 0.002442 seconds.
pos_107/train010.JPG ... Running Time: 0.002501 seconds.
```

而detection所花的時間幾乎不比extraction多多少，可以在0.003秒內完成。

目前其實有很多造成實驗結果不佳的明顯原因：

1. 資料量過少。107張positive資料對於cascaded AdaBoost的訓練明顯不足，但由於堅持使用校園拍的照片，和堅持正面或斜正面的車輛，未來仍不考慮使用網路上現成的data set作為訓練資料。以大學部專題的水準，預計收集到300張positive的訓練資料。negative因為可以用程式去大量切，所以問題不大。

2. Feature過少。由於時間因素，在改寫成OpenCV版本時只取出了5個標準矩形的rectangle features。我們知道gradient-based features是以準確率見長，如果能補上EOH和ED，相信準確率會有不少的提升。另外45°傾斜的rectangle features對於結果也有幫助，不過OpenCV內建函式算的tiltedSum和我認知的有差異，所以要自己寫，並且留意對速度的影響，來決定要不要採用。

   省略很多文獻細節。目前仍是「系統能運作、功能完整、速度快」優先於「準確率高」，所以像EOH、ED、45°-tiled REC features，integral histogram，real AdaBoost，confidence values，Meta Stages等細節都還沒實作出來，甚至有的理論還沒弄懂。這些對於準確率或多或少都有影響，也是未來加強的方向。

7. 未來目標：

後續的改進大致上會照著以下順序進行：

1. 增加資料量至300張positive訓練資料。2. 補上EOH、ED，以追上舊程式碼的水準。3. 再補上45°-tiled REC，如此可預期一定水準的準確率，若對速度影響太多則可能放棄。4. integral histogram加速。5. 加上Sliding window的功能，測試影像將不限於128 x 128。6. 加上video mode的功能，讓使用環境更貼近真實的校園行走。7. 加上camera mode的功能，達到計畫最終的demo形式，不過攝影機本身的品質是一個重要因素。8. 補上real AdaBoost，會讓訓練強分類器的過程合理許多。9. 補上confidence values的計算。10. 使用OpenCV內建的SVM訓練出Meta Stages，並補進訓練演算法。

基本上[1~4]屬於performance的基本改進，[5~7]屬於系統功能改進，[8~10]屬於performance的進階改進，比較可有可無。