###### [net]層

**參數：batch, subdivisions，max\_batches**  
每batch個樣本代表完成一次權重參數更新。max\_batches是最大權重更新次數，完成max\_batches次權重更新後停止訓練。

[net]

batch=64 每batch個樣本更新反向傳遞更新參數

subdivisions=8 如果記憶體不夠大，將batch分割為subdivisions個子batch，前傳

**max\_batches = 500200，所以要達到訓練迭代到第500200個batch時纔會訓練自動結束**

學習速率：學習速率第一次可以設置一個大一點的學習速率加快收斂，也可以採用動態變化學習速率的方法，比如每一輪都乘上一個衰減係數或者是根據損失的變化動態調整速率值。

1.Batch\_Size(批尺寸)

該引數主要用於批梯度下降演算法(Batch Gradient Descent)中，批梯度下降演算法是每次迭代都遍歷批中的所有樣本，由批中的樣本共同決定最優的方向，Batch\_Size 正是批中的樣本數量。 。

在合理範圍內增大Batch\_Size,可以

（1）提高記憶體利用率，進而提高大矩陣乘法的並行效率；

（2）跑完一次epoch（全資料集）所需的迭代次數減少，對於相同資料量的處理速度進一步加快；

（3）在一定範圍內，一般來說Batch\_Size越大，其確定的下降方向越準，引起的訓練震盪越小。

盲目增大Batch\_Size的壞處：

（1）超出記憶體容量；

（2）跑完一次epoch(全資料集)所需的迭代次數減小，要想達到相同的精度，所需要的epoch數量越多，對引數的修正更加緩慢；

（3）Batch\_Size 增大到一定程度，其確定的下降方向已經基本不再變化。

2.subdivisions

如果記憶體不夠大，將batch分割為subdivisions個子batch,每個子batch的大小為batch/subdivisions；

darknet程式碼中，是將batch/subdivisions命名為batch；

3.衝量-momentum

梯度下降法中一種常用的加速技術，對於一般的SGD，其表示式為

沿著負梯度方向下降，而帶momentum項的SGD則寫成

（1）batch\_size：批大小。在深度學習中，一般採用SGD訓練，即每次訓練在訓練集中取batch\_size個樣本訓練；

（2）iteration：1個iteration等於使用batchsize個樣本訓練一次；

（3）epoch：1個epoch等於使用訓練集中的全部樣本訓練一次；

訓練log中各引數的意義

Region Avg IOU：平均的IOU，代表預測的bounding box和ground truth的交集與並集之比，期望該值趨近於1。

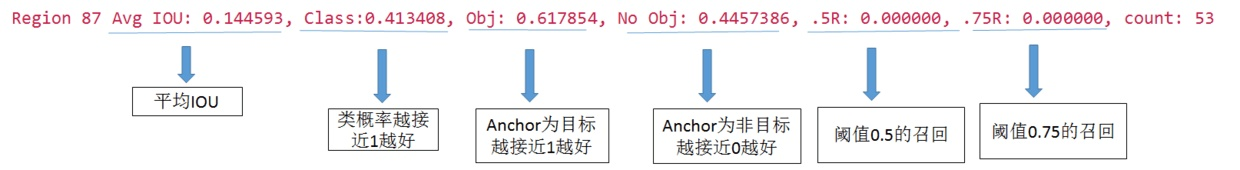
Class:是標註物體的概率，期望該值趨近於1.

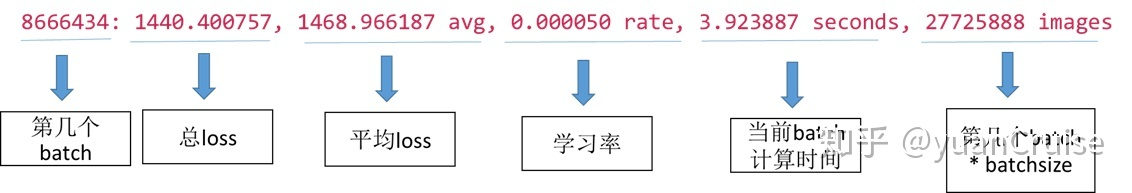
Obj：期望該值趨近於1.

No Obj:期望該值越來越小但不為零.

Avg Recall：期望該值趨近1

avg：平均損失，期望該值趨近於0

1. dropout：數據第一次跑模型的時候可以不加dropout，後期調優時加上dropout用於防止過擬合。特別是數據量相對較小的時候。
2. 訓練輪數：模型收斂即可停止迭代，一般可以採用驗證集作爲停止迭代的條件
3. 若batch=32,subdivions=2，一次iteration有32張圖，但是分成2部分來輸出。那麼輸出log就會分成兩部分，第一部分用到16張圖，第二部分用到另外16張圖。但是這裏我自己的代碼是設置的：batch=8,subdivions=5。  
   
4. 對比上面的截圖。補充：上面截圖中count值的意思是：the amount of positives (objects to be detected) present in the current subdivision of images。當前subdivision圖片中正樣本的圖片的數量。在我的實際輸出中，平均IOU值是-nan。類概率值也是-nan。Obj值也是-nan。閾值0.5的召回也是-nan，閾值0.75的召回也是-nan。



82 卷積層爲最大的預測尺度, 使用較大的 mask, 但是可以預測出較小的物體;  
94 卷積層爲中間的預測尺度, 使用中等的 mask;  
106卷積層爲最小的預測尺度, 使用較小的 mask, 可以預測出較大的物體.

Region 82 Avg IOU: 0.761178, Class: 0.998803, Obj: 0.870125, No Obj: 0.003177, .5R: 1.000000, .75R: 0.500000, count: 2

Region Avg IOU:0.761178： 表示在當前 subdivision 內的圖片的平均 IOU, 代表預測的矩形框和真實目標的交集與並集之比, 這裏是 7６.１%, 這個模型此時已經達到了很高的訓練精度;

Class: 0.998803: 標註物體分類的正確率, 期望該值趨近於1;

Obj: 0.870125： 越接近 1 越好;

No Obj: 0.003177： 期望該值越來越小, 但不爲零;

.5R: 1.000000： 是在 recall/count 中定義的, 是當前模型在所有 subdivision 圖片中檢測出的正樣本與實際的正樣本的比值。在本例中, 全部的正樣本被正確的檢測到。

count: ２： 所有當前 subdivision 圖片（本例中一共 ２張）中包含正樣本的圖片的數量。 在輸出 log 中的其他行中。（在 subdivision 中含有不包含在檢測對象 classes 中的圖片）

若steps=1000，scale=.1，表示迭代到1000次時，學習率衰減10倍。若調整max\_batches的值，那麼需要同時調整steps值，scale值隨便調整或不調整。

**4. 提升檢測效果**

* random=1可以設置適應多解析度
* 提升解析度：416--> 608等必須是32倍數
* 重新計算你的資料集的anchor:(注意設置的時候計算問題)

darknet.exe detector calc\_anchors data/obj.data -num\_of\_clusters 9 -width 416 -height 416

* 檢查資料集通過https://github.com/AlexeyAB/Yolo\_mark
* 資料集最好每個類有2000張圖片，至少需要反覆運算2000\*類的個數
* 資料集最好有沒有標注的物件，即負樣本，對應空的txt檔，最好有多少樣本就設計多少負樣本。
* 對於一張圖有很多個樣本的情況，使用max=200屬性(yolo層或者region層)
* 為了加速訓練，可以做fine-tuning而不是從頭開始訓練，設置stopbackward=1在網路的結束部分（以####作為分割）
* 在訓練完以後，進行目標檢測的時候，可以提高網路的解析度，以便剛好檢測小目標。
  + 不需要重新訓練，需要使用原先低解析度的權重，測用更高解析度。
  + 為了得到更高的檢測效果，可以提升解析度至608\*608甚至832\*832

**5. 總結**

為了小目標：

* 提升解析度
* 在測試時候提升解析度
* 資料集添加跟正樣本數量一樣多的負樣本
* 資料集每個類至少2000張，訓練反覆運算次數2000\*classes個數
* 設置自己資料集的anchor

yolov3:filters=(classes + 5)x3

random=1

egion xx: cfg檔中yolo-layer的索引，82是大物體層，94是中物體層，106小物體層；

Avg IOU: 當前疊代中，預測的box與標註的box的平均交並比，越大越好，期望數值為1；

Class: 標註物體的分類準確率，越大越好，期望數值為1；

obj: 越大越好，期望數值為1；

No obj: 越小越好；

.5R: 以IOU=0.5為閾值時候的recall; recall = 檢出的正樣本/實際的正樣本

0.75R: 以IOU=0.75為閾值時候的recall;

count: 正樣本數目。

* 設置cfg檔中max\_batches參數 (一般可以設置為classes\*2000 但是不要低於 4000), 比如 如果你有三個類，那麼設置max\_batches=6000。
* 設置steps參數，一般為80%和90%的max\_batches。比如 steps=4800,5400
* 設置網路輸入長寬必須能夠整除32，比如 width=416 height=416

loss曲線和mAP曲線，mAP每4個epoch對驗證集進行一次測試

1. 訓練之後：

* 通過在.cfg檔中設置（height=608 and width=608）或者（height=832 and width=832）或者任何32的倍數，這會提升準確率並使得對小目標的檢測更加容易。
  + 沒有必要重新訓練模型，直接使用用416x416解析度訓練出來的.weights模型檔。
  + 但是要獲得更高的準確率，你應該使用608x608或者832x832來訓練，注意如果Out of memory發生了，你應該在.cfg檔中增加subdivisions=16，32，64。

Cfg參數

[net]                        ★ [xxx]開始的行表示網絡的一層，其后的內容為該層的參數配置，[net]為特殊的層，配置整個網絡

# Testing                    ★ #號開頭的行為注釋行，在解析cfg的文件時會忽略該行

# batch=1

# subdivisions=1

# Training

batch=64                     ★ 這兒batch與機器學習中的batch有少許差別，僅表示網絡積累多少個樣本后進行一次BP

subdivisions=16              ★ 這個參數表示將一個batch的圖片分sub次完成網絡的前向傳播

                             ★★ 敲黑板：在Darknet中，batch和sub是結合使用的，例如這兒的batch=64，sub=16表示訓練的過

                             程中將一次性加載64張圖片進內存，然后分16次完成前向傳播，意思是每次4張，前向傳播的循環過程中

                             累加loss求平均，待64張圖片都完成前向傳播后，再一次性后傳更新參數

                             ★★★ 調參經驗：sub一般設置16，不能太大或太小，且為8的倍數，其實也沒啥硬性規定，看着舒服就好

                             batch的值可以根據顯存占用情況動態調整，一次性加減sub大小即可，通常情況下batch越大越好，還需

                             注意一點，在測試的時候batch和sub都設置為1，避免發生神秘錯誤！

width=608                    ★ 網絡輸入的寬width

height=608                   ★ 網絡輸入的高height

channels=3                   ★ 網絡輸入的通道數channels

                             ★★★ width和height一定要為32的倍數，否則不能加載網絡

                             ★ 提示：width也可以設置為不等於height，通常情況下，width和height的值越大，對於小目標的識別

                             效果越好，但受到了顯存的限制，讀者可以自行嘗試不同組合

momentum=0.9                 ★ 動量 DeepLearning1中最優化方法中的動量參數，這個值影響着梯度下降到最優值得速度

decay=0.0005                 ★ 權重衰減正則項，防止過擬合

angle=0                      ★ 數據增強參數，通過旋轉角度來生成更多訓練樣本

saturation = 1.5             ★ 數據增強參數，通過調整飽和度來生成更多訓練樣本

exposure = 1.5               ★ 數據增強參數，通過調整曝光量來生成更多訓練樣本

hue=.1                       ★ 數據增強參數，通過調整色調來生成更多訓練樣本

learning\_rate=0.001          ★ 學習率決定着權值更新的速度，設置得太大會使結果超過最優值，太小會使下降速度過慢。

                             如果僅靠人為干預調整參數，需要不斷修改學習率。剛開始訓練時可以將學習率設置的高一點，

                             而一定輪數之后，將其減小在訓練過程中，一般根據訓練輪數設置動態變化的學習率。

                             剛開始訓練時：學習率以 0.01 ~ 0.001 為宜。一定輪數過后：逐漸減緩。

                             接近訓練結束：學習速率的衰減應該在100倍以上。

                             學習率的調整參考

                             ★★★ 學習率調整一定不要太死，實際訓練過程中根據loss的變化和其他指標動態調整，手動ctrl+c結

                             束此次訓練后，修改學習率，再加載剛才保存的模型繼續訓練即可完成手動調參，調整的依據是根據訓練

                             日志來，如果loss波動太大，說明學習率過大，適當減小，變為1/5，1/10均可，如果loss幾乎不變，

                             可能網絡已經收斂或者陷入了局部極小，此時可以適當增大學習率，注意每次調整學習率后一定要訓練久

                             一點，充分觀察，調參是個細活，慢慢琢磨

                             ★★ 一點小說明：實際學習率與GPU的個數有關，例如你的學習率設置為0.001，如果你有4塊GPU，那

                             真實學習率為0.001/4

burn\_in=1000                 ★ 在迭代次數小於burn\_in時，其學習率的更新有一種方式，大於burn\_in時，才采用policy的更新方式

max\_batches = 500200         ★ 訓練次數達到max\_batches后停止學習，一次為跑完一個batch

policy=steps                 ★ 學習率調整的策略：constant, steps, exp, poly, step, sig, RANDOM，constant等方式

                             參考https://nanfei.ink/2018/01/23/YOLOv2%E8%B0%83%E5%8F%82%E6%80%BB%E7%BB%93/#more

steps=400000,450000

scales=.1,.1                 ★ steps和scale是設置學習率的變化，比如迭代到400000次時，學習率衰減十倍，45000次迭代時，學

                             習率又會在前一個學習率的基礎上衰減十倍

[convolutional]              ★ 一層卷積層的配置說明

batch\_normalize=1            ★ 是否進行BN處理，什么是BN此處不贅述，1為是，0為不是

filters=32                   ★ 卷積核個數，也是輸出通道數

size=3                       ★ 卷積核尺寸

stride=1                     ★ 卷積步長

pad=1                        ★ 卷積時是否進行0 padding,padding的個數與卷積核尺寸有關，為size/2向下取整，如3/2=1

activation=leaky             ★ 網絡層激活函數

                             ★★ 卷積核尺寸3\*3配合padding且步長為1時，不改變feature map的大小

# Downsample

[convolutional]              ★ 下采樣層的配置說明

batch\_normalize=1

filters=64

size=3

stride=2

pad=1

activation=leaky             ★★ 卷積核尺寸為3\*3，配合padding且步長為2時，feature map變為原來的一半大小

[shortcut]                   ★ shotcut層配置說明

from=-3                      ★ 與前面的多少次進行融合，-3表示前面第三層

activation=linear            ★ 層次激活函數

    ......

    ......

[convolutional]              ★ YOLO層前面一層卷積層配置說明

size=1

stride=1

pad=1

filters=255                  ★ filters=num(預測框個數)\*(classes+5)，5的意義是4個坐標加一個置信率，論文中的tx,ty,tw,th,

                             c，classes為類別數，COCO為80,num表示YOLO中每個cell預測的框的個數，YOLOV3中為3

                             ★★★ 自己使用時，此處的值一定要根據自己的數據集進行更改，例如你識別4個類，則：

                             filters=3\*(4+5)=27,三個fileters都需要修改，切記

activation=linear

[yolo]                       ★ YOLO層配置說明

mask = 0,1,2                 ★  使用anchor的索引，0，1，2表示使用下面定義的anchors中的前三個anchor

anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,  156,198,  373,326

classes=80                   ★ 類別數目

num=9                        ★ 每個grid cell總共預測幾個box,和anchors的數量一致。當想要使用更多anchors時需要調大num

jitter=.3                    ★ 數據增強手段，此處jitter為隨機調整寬高比的范圍，該參數不好理解，在我的源代碼注釋中有詳細說明

ignore\_thresh = .7

truth\_thresh = 1             ★ 參與計算的IOU閾值大小.當預測的檢測框與ground true的IOU大於ignore\_thresh的時候，參與

                             loss的計算，否則，檢測框的不參與損失計算。

                             ★ 理解：目的是控制參與loss計算的檢測框的規模，當ignore\_thresh過於大，接近於1的時候，那么參與

                             檢測框回歸loss的個數就會比較少，同時也容易造成過擬合；而如果ignore\_thresh設置的過於小，那么

                             參與計算的會數量規模就會很大。同時也容易在進行檢測框回歸的時候造成欠擬合。

                             ★ 參數設置：一般選取0.5-0.7之間的一個值，之前的計算基礎都是小尺度（13\*13）用的是0.7，

                             （26\*26）用的是0.5。這次先將0.5更改為0.7。

random=1                     ★ 為1打開隨機多尺度訓練，為0則關閉