Day 38

深度學習與電腦視覺

學習馬拉松

cupoy 陪跑專家：杜靖愷

CUPOY

YOLO 演進

重要知識點

- 了解 YOLO 改進的思路。
- YOLO YOLO 的優缺點及其極限。

YOLOv1 的缺點

CUPOY

- 對相互靠很近或者很小的物體檢測效果不好，這是因為一個網格只預測兩個 bboxes，並且只屬於一個類別。
- 同一個類別出現新的，不常見的長寬比時（也就是訓練集裡面沒出現過的 bbox 比例），會檢測不到。
- 損失函數的設計缺陷。
    - 假設定位誤差為 E，這個 E 對大物體來說可能是還可以接受的，但是對小物體來說可能就偏離 groundtruth bbox 很遠了，直覺上來說，小物體對 E 應該要更加敏感，雖然作者試圖在損失函數使用平方根來克服這個缺點，但還是沒有完全解決。

YOLO 改進

YOLO 作者在 2016 年時發表了 YOLO9000，這篇論文的主要貢獻是

- 提出 YOLOv2
  - 在 YOLO 速度的基礎上對準確度做的改進
- 提出 YOLO9000
  - 設計出一種 Joint Training Algorithm，對檢測以及分類任務同時進行訓練
  - 應用至 ImageNet 和 COCO 資料集當中，最終使得網路檢測的物體超過 9000 種

在 2018 年時又在發表了 YOLOv3，設計出更好的檢測網路，在保持其速度的情況下，達到更高的準確度。

接下來會簡要地說明 YOLOv2 以及 YOLOv3 的改進，對 YOLO9000 有興趣的同學可以自己去看論文以及補充的參考資料，如果要完全掌握 YOLO，建議需要去看論文以及程式碼。

YOLOv2 各個改進策略帶來的提升

YOLOv2 具體做了不少改進，根據論文，其在 VOC2007 資料集上的 mAP 由 63.4 提升到 78.6！

	YOLO							YOLOv2
batch norm?		✓	✓	✓	✓	✓	✓	✓
hi-res classifier?			✓	✓	✓	✓	✓	✓
convolutional?				✓	✓	✓	✓	✓
anchor boxes?				✓	✓			
new network?					✓	✓	✓	✓
dimension priors?						✓	✓	✓
location prediction?						✓	✓	✓
passthrough?							✓	✓
multi-scale?								✓
hi-res detector?								✓
VOC2007 mAP	63.4	65.8	69.5	69.2	69.6	74.4	75.4	76.8	**78.6**

YOLOv2 主要改進策略簡述

CUPOY

策略	簡述
Batch Normalization	在每個卷積層後添加 batch normalization，同時丟棄 dropout
High Resolution Classifier	一開始在 ImageNet finetune 時，使用更高 resolution 的圖像作為 input， 同時最後輸出層的 feature map 大小由 7 x 7 改成 13 x 13
Convolutional With Anchor Boxes	為克服 YOLOv1 每個網格只能預測一個物件類別的缺點，借鑒 Faster RCNN 的 anchor boxes 思路，放棄用全連接層預測 bboxes，而是基於 anchor boxes 的 offsets 和 confidence 來獲得 bboxes，並且每個 anchor boxes 都會預測類別 **YOLO 輸出層: S x S x (B x 5 + C)** **YOLOv2 輸出層: S x S x (B x (5 + C))**
Dimension Clusters	在 Faster RCNN 中 anchor box 的長寬值是人工設定的，YOLOv2 使用 K-Means 的方式對資料集的 bboxes 做分群來取得長寬值，作者通過實驗最後決定用 5 個 anchor

這部分其實很難在一天裡面理解完，這裡只做了簡單的介紹，想要更理解細節的可以參考這份筆記

YOLOv2 網路架構 - Darknet-19

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Table 6: Darknet-19.

作者在 YOLOv2 採用了新的網路 backbone 該網路包含 19 個 convolutional layer 和 5 個 max pooling layer，最後用 average pooling layer 代替 YOLOv1 中的 fully connected layer 進行預測

YOLOv3 主要改進思路

YOLOv3 則是在 YOLOv2 的基礎上，改良了網路 backbone 以及 output layer 的結構，使得其準確度有很大的提升，主要的改進思路是

- Multiscale prediction
  - 輸出了 3 個不同 scale 的 feature map (分別是 13 x 13, 26 x 26 以及 52 x 52)，越大的 feature map 就能劃分出越細緻的網格，也就能檢測出越精細的物體
- 加深 backbone
  - 提出新的 backbone Darknet-53，採用簡單的 residual block 作為加深網路的手段
- Loss function
  - 在 YOLO 系列的論文中，只有 YOLOv1 明確提了 loss function 的公式，而其實 YOLOv3 有把原本的 YOLOv2 中的 softmax loss 變成 logistic loss，主要是因為 softmax 意味著每一個 boxes candidate 只對應一個類別，但實際上並不總是這樣，因為有些數據集會有重疊的 label，比如說不同種類的馬。

YOLOv3 網路架構 - Darknet - 53

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

作者在 YOLOv3 採用了新的網路 backbone 的架構，該網路包含 53 個 convolutional layer，相比 YOLOv2，拿掉了 MaxPooling Layer 然後添加了 residual block。

This new network is much more powerful than Darknet-19 but still more efficient than ResNet-101 or ResNet-152. Here are some ImageNet results:

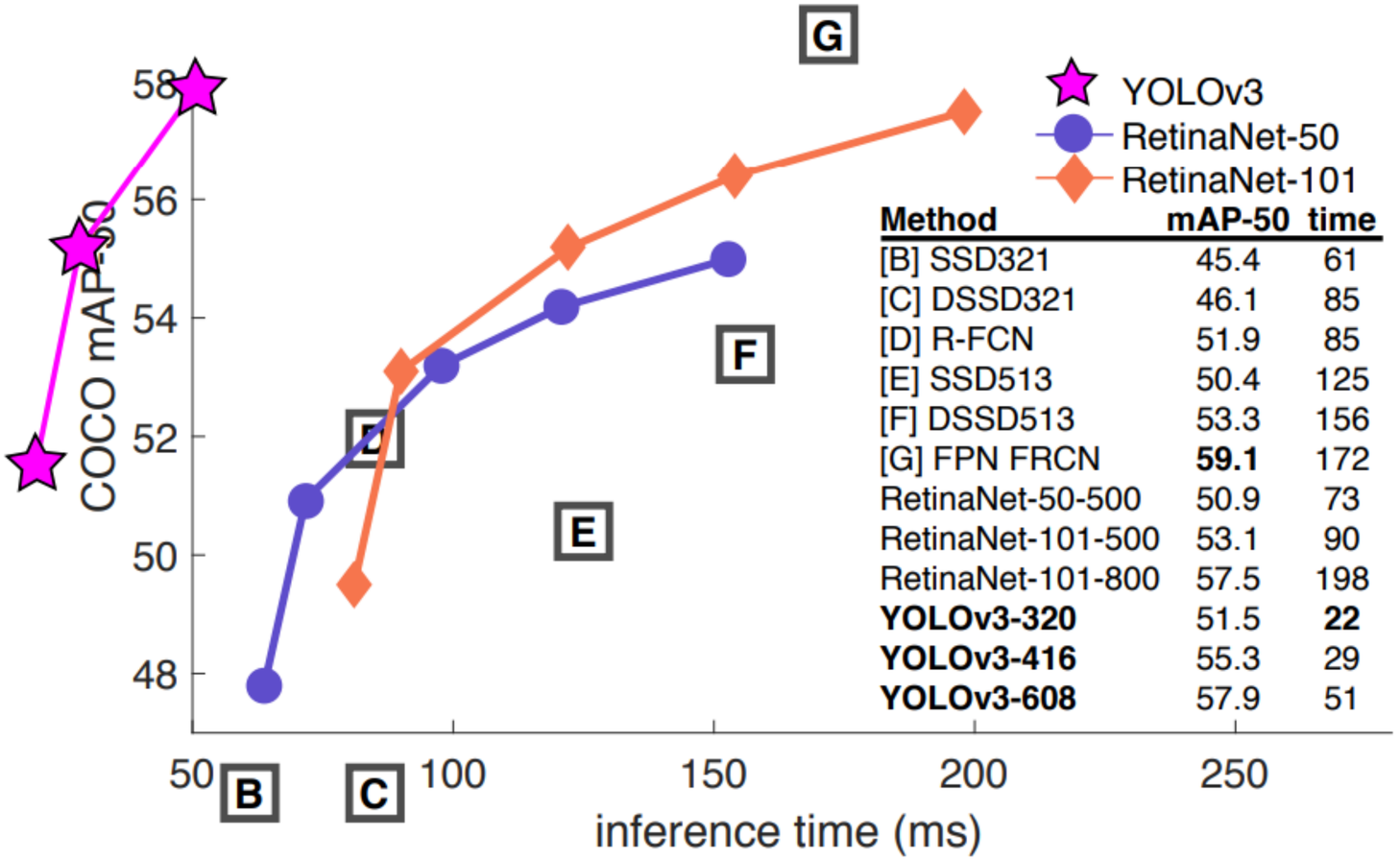
Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [15]	74.1	91.8	7.29	1246	**171**
ResNet-101[5]	77.1	93.7	19.7	1039	53
ResNet-152 [5]	**77.6**	**93.8**	29.4	1090	37
Darknet-53	77.2	**93.8**	18.7	**1457**	78

Table 2. **Comparison of backbones.** Accuracy, billions of operations, billion floating point operations per second, and FPS for various networks.

論文中的圖表顯示出原本的 Darknet-19 相比 ResNet 或者 Darknet-53 還是最快的 backbone，但是 Darknet-53 在維持 real time 的速度下，在 ImageNet 的分類任務上具備和 ResNet 差不多的準確度。

YOLOv3 的提升

由論文中的圖表可以看出來，YOLOv3 並沒有追求更高的速度，而是在保證足夠實時的基礎上追求更好的準確度。



Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

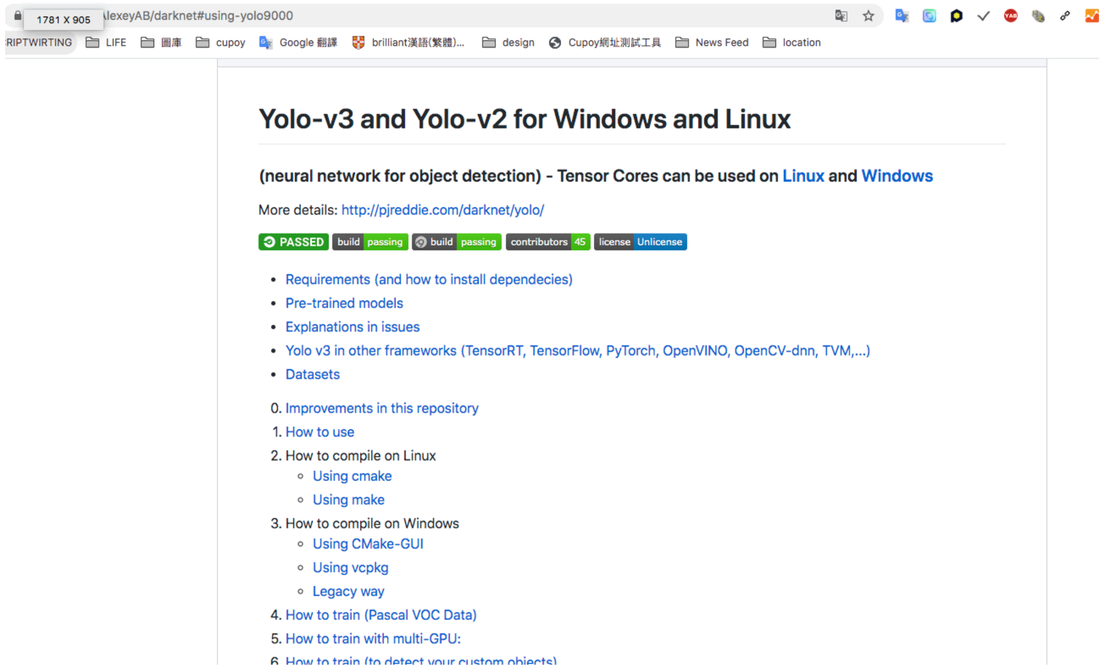
推薦閱讀資料

- YOLO 的發展
- YOLOv2
  - YOLOv2--論文學習筆記（算法詳解）
- YOLOv3
  - 【目標檢測簡史】進擊的YOLOv3，目標檢測網絡的巔峰之作

- YOLO 系列的精華在於它通過把圖片劃分為網格來做檢測，只是不同版本劃分的數量不一樣而已。
  - 也由於這樣的設計，靠一個 loss function 搞定訓練，只需要關注 input 以及 output layer
- 從 YOLOv2 在每一層 convolutional layer 加入 batch normalization 作為正規化、加速收斂和避免 overfitting 的方法
- 在速度和準確度直接是有 tradeoff 的，想要速度快點，可以犧牲準確度，用 tinyyolo 做 backbone
- YOLOv3 在多個 scale 的 feature map 上做檢測，對小目標的效果提升明顯
- 下一天的課程是 YOLOv3 的程式碼實作，你可能可以很快地就用上 YOLOv3，但你不大可能很快就懂 YOLOv3，所以建議沒辦法完全消化的 YOLO 這系列算法細節的同學可以先往實作課程去體會，之後有需要再回過頭來消化

參考資料

- [YOLO9000 - darknet 實現](#)
  - [github 上相關的討論](#)



解題時間 Let’s Crack It

CUPOY

請跳出 PDF 至官網 Sample Code & 作業開始解題