嵌入式智慧影像分析與實境界面
Fall 2021

Instructor：Yen-Lin Chen(陳彥霖), Ph.D.

Professor

Dept. Computer Science and Information Engineering

National Taipei University of Technology

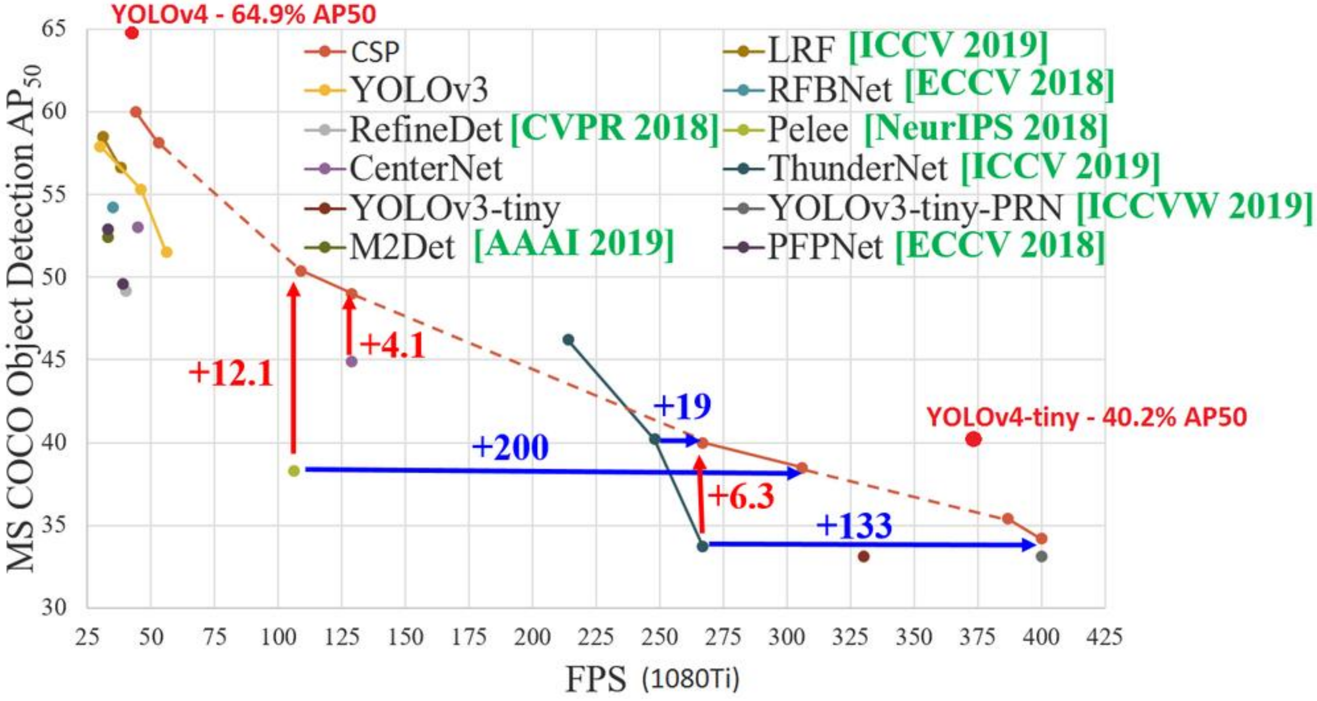
Lecture 9

YOLOv4-tiny介紹

YOLOv4-tiny

You Only Look Once (YOLOv4-tiny)

- 於2020年六月提出。
- YOLOv4著重於準確度，而tiny版則是著重於運算速度，將整體架構的層數大幅減少。
- 為了在Jetson Nano上使用時可以達到高FPS，於本課程專案中我們選用YOLOv4-tiny來執行。



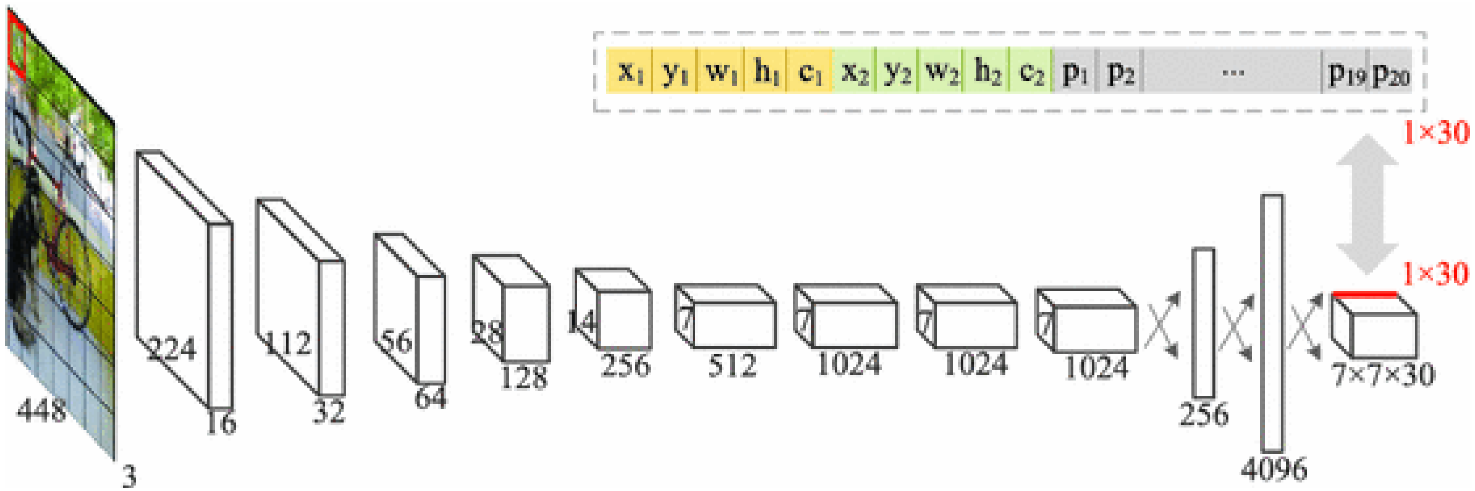
- 在1080Ti上，YOLOv4與YOLOv4-tiny與其他影像辨識方法的精準度(AP)與速度 (FPS)比較

YOLOv4-tiny Introduction

- **主要改良部分:**
  - **Generic Processing Element(PE) model for calculation**
    - Computing engine is flexible enough to handle with both CONV layers and FC layers
    - 64 identical PEs are included, each accomplishing up to 32 MAC operations in a pipeline manner
  - **Hardware Memory Hierarchy for data reuse**
    - Data and parameters are well organized so that they can be reused or shared
  - **Ping-pong buffer for task parallelism**
    - Two input buffers and two output buffers both work in a ping-pong manner
    - Data loading/storing and data computing tasks are conducted simultaneously, which greatly reduces whole processing time
  - **Singular Value Decomposition（SVD) for fully-connected layers**
    - Application of SVD to FC layers reduces 80.6% memory access as well as computing operations

YOLOv4-tiny Architecture

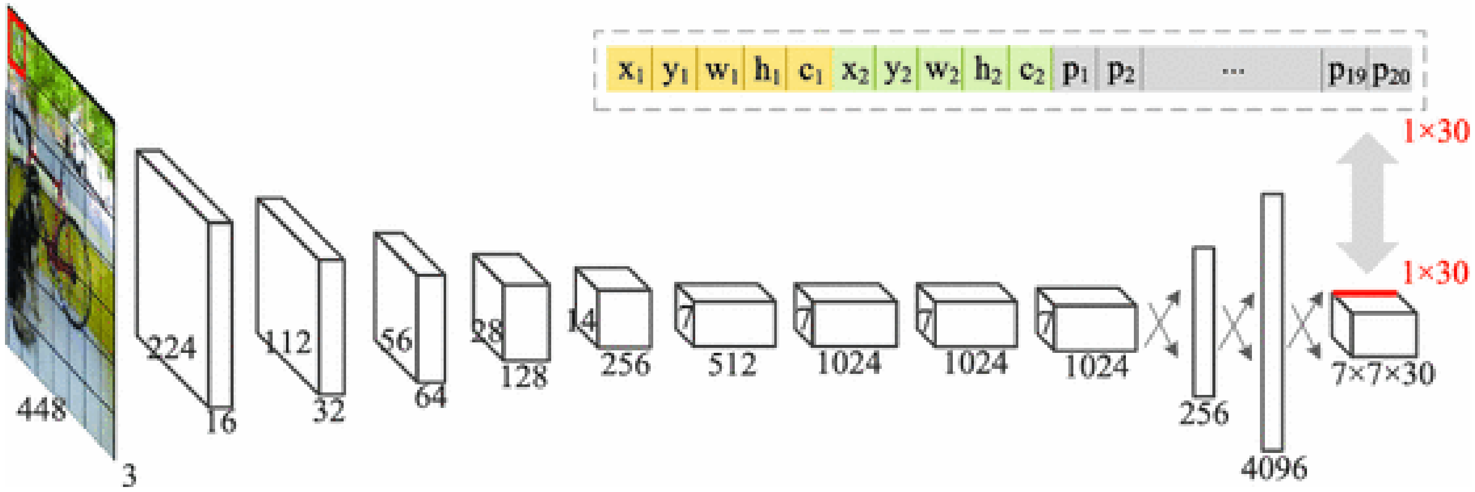
- 9 convolutional layers and 3 fully-connected layers.
- 30 values in dashed box refer to the contents of each output segment:
  - 10 for location information
  - 20 for class probabilities



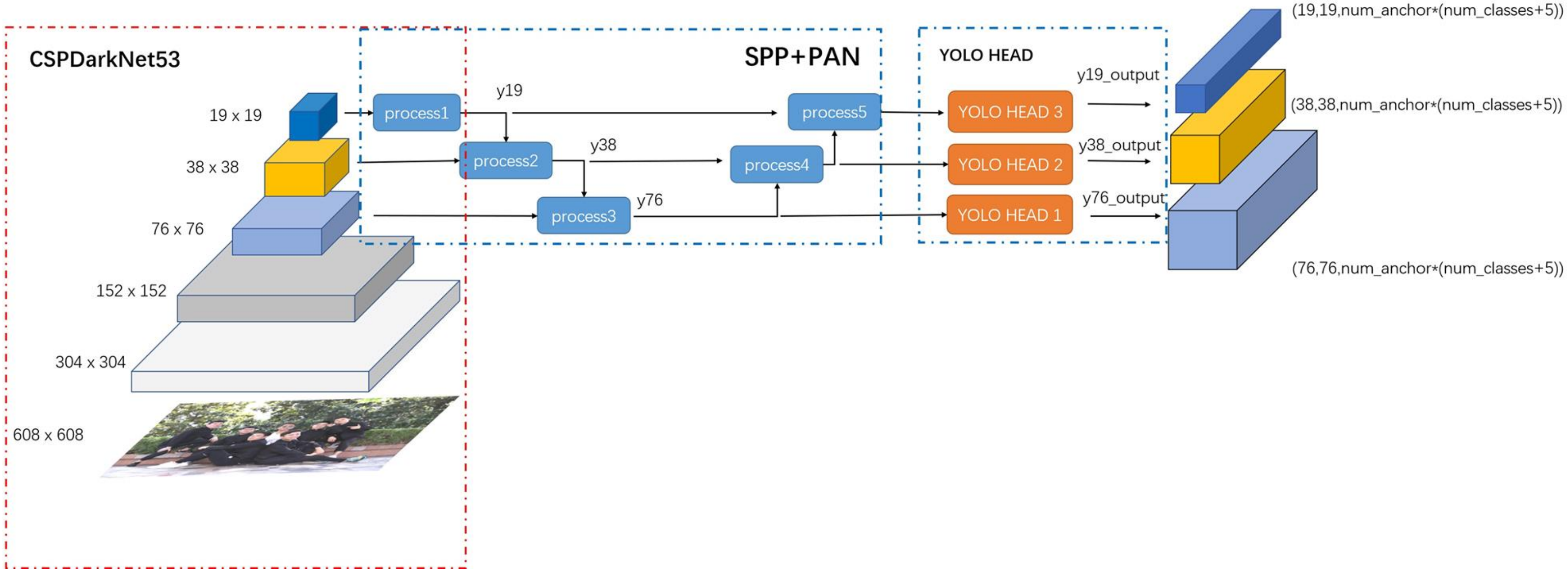
x_1 y_1 w_1 h_1 c_1 x_2 y_2 w_2 h_2 c_2 p_1 p_2 ··· p_{19} p_{20}
1×30
1×30
448
3
224
16
112
32
56
64
28
128
14
256
7
512
7
1024
7
1024
7
1024
256
4096
7×7×30

YOLOv4-tiny Architecture cont.

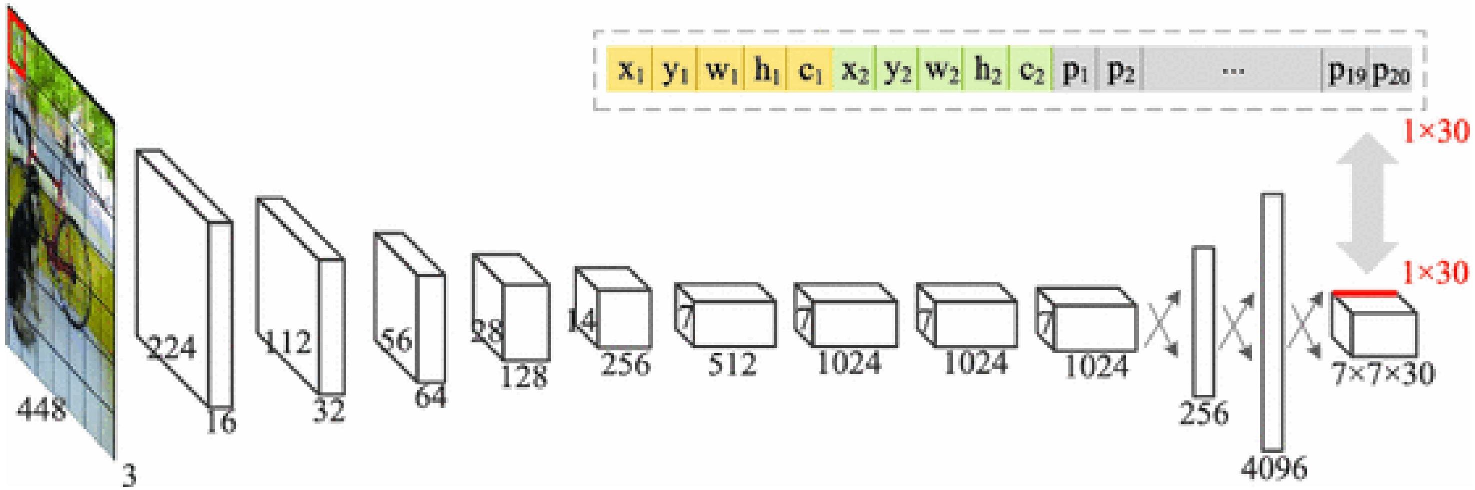
- Bounding box: coordinates (*x*, *y*), width *w*, height *h* and confidence c
  - $confidence = P_r(Object) * IOU$
- Probability of the *i*th class
  - $P_r(Class_i|Object) * P_r(Object) * IOU = P_r(Class_{i)} * IOU$



Yolov4架構 vs Yolov4-tiny架構

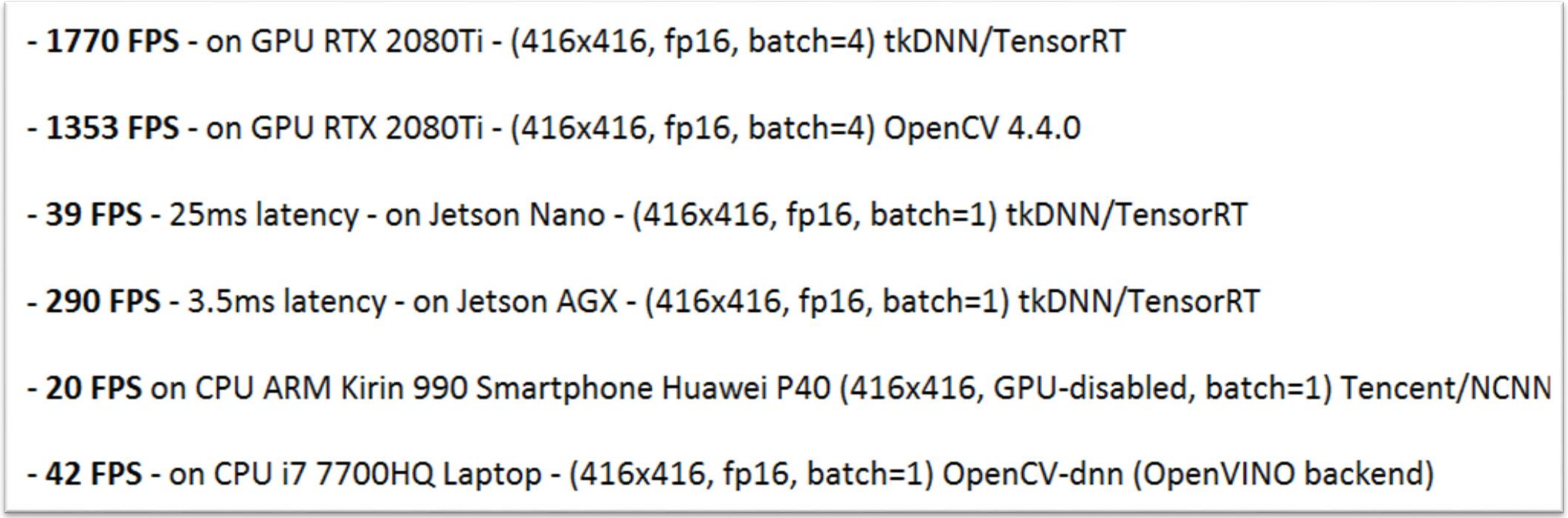


Yolov4架構



Yolov4-tiny
架構

YOLOv4-tiny FPS比較



- **1770 FPS** - on GPU RTX 2080Ti - (416x416, fp16, batch=4) tkDNN/TensorRT

- **1353 FPS** - on GPU RTX 2080Ti - (416x416, fp16, batch=4) OpenCV 4.4.0

- **39 FPS** - 25ms latency - on Jetson Nano - (416x416, fp16, batch=1) tkDNN/TensorRT

- **290 FPS** - 3.5ms latency - on Jetson AGX - (416x416, fp16, batch=1) tkDNN/TensorRT

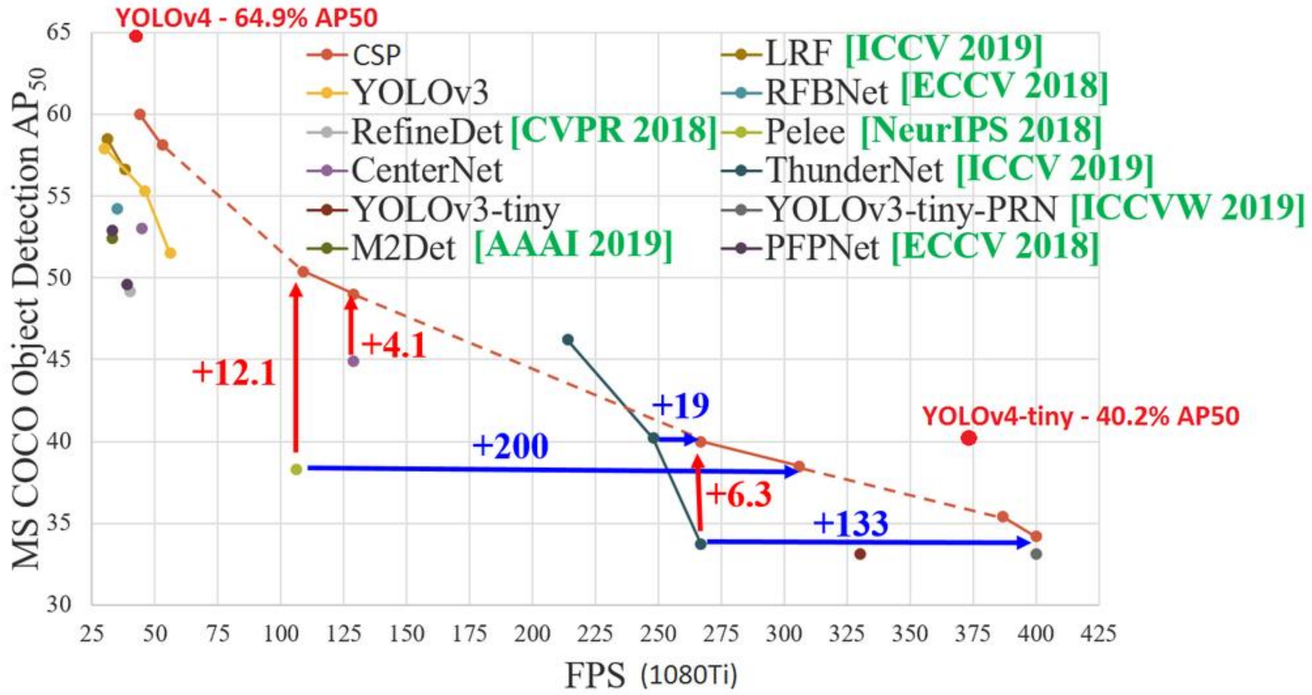
- **20 FPS** on CPU ARM Kirin 990 Smartphone Huawei P40 (416x416, GPU-disabled, batch=1) Tencent/NCNN

- **42 FPS** - on CPU i7 7700HQ Laptop - (416x416, fp16, batch=1) OpenCV-dnn (OpenVINO backend)

- YOLOv4-tiny在不同平台上的FPS比較

YOLOv4 vs YOLOv4-tiny

	YOLOv4	YOLOv4-tiny
Pre-trained convolution layers	137	29
FPS	50	375
AP(Average Precision)	65	40
使用場景	需要高準確率之應用	Real-time object detection



參考資料

參考資料

- Yolo-tiny
  - https://link.springer.com/chapter/10.1007/978-981-10-8108-8_21
  - YOLOv4 vs YOLOv4-tiny. Training custom YOLO detectors for Mask… | by Techzizou | Analytics Vidhya | Medium