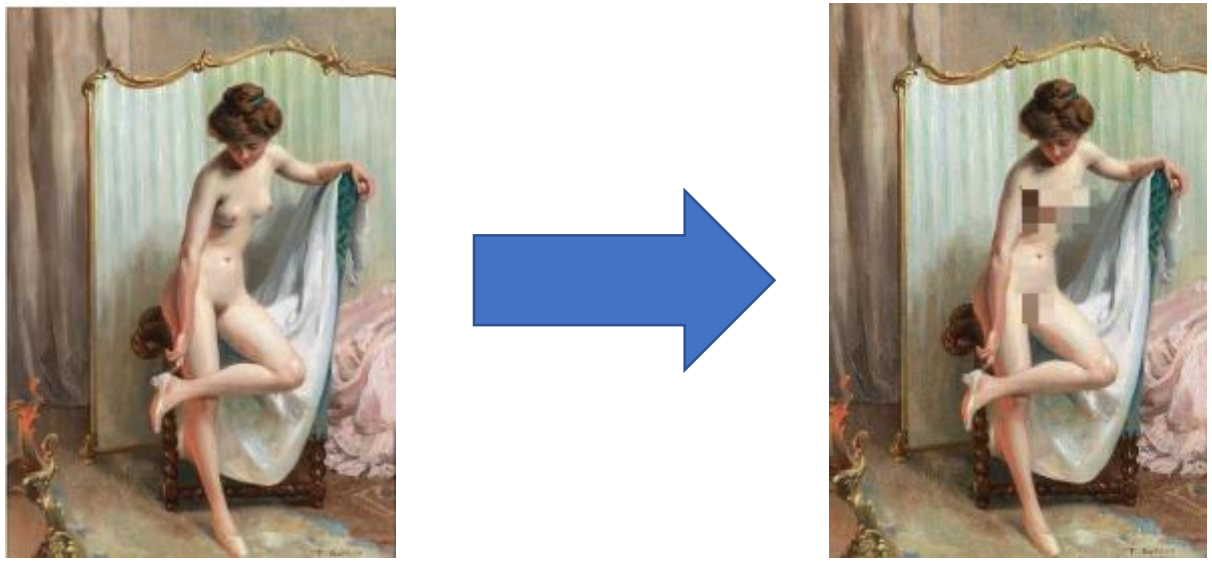
人工智慧概論
Team24 期末project

江皓霖 何承原 周庭右 葉晟育

Introduction

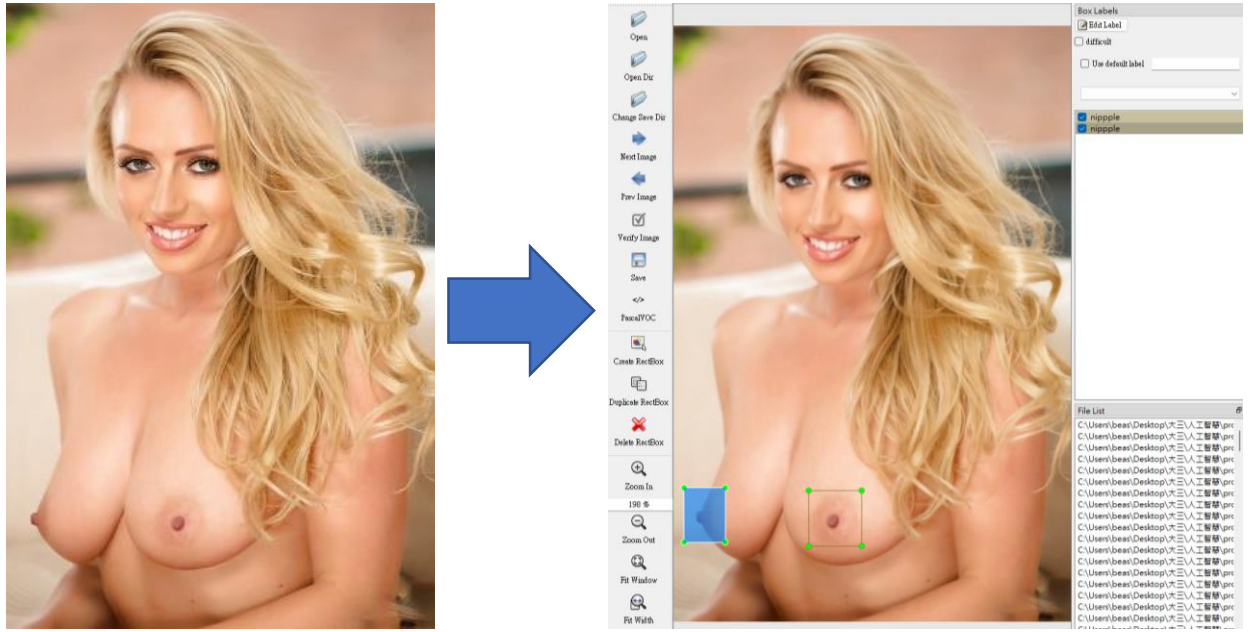
- 在網路上有許多「少兒不宜」的圖片，而馬賽克能夠將這些內容模糊化，讓這些圖片能 夠在合法的狀態下在網路上傳播，但若只用人工的方式打馬賽克，是件很沒有效率的事，所以我們的專題便是自動偵測出圖片中十八禁的地方，再自動打上馬賽克，讓打 馬這件是自動化。未來可能可以將它改寫成背景程式放在電腦上，讓兒童玩電腦時不 會輕易地看到腥羶色的內容。



Dataset

- 首先我們將奶頭圖片用 LabelImg 將奶頭標框起來，並且輸出其 xml 檔案

- 總共準備了1000 多份的 train data，一份 train data 包含原圖以及旗標框過後的 xml 檔



- 我們最後使用華爾街之狼的片段當作馬賽克的目標

Literature-Review

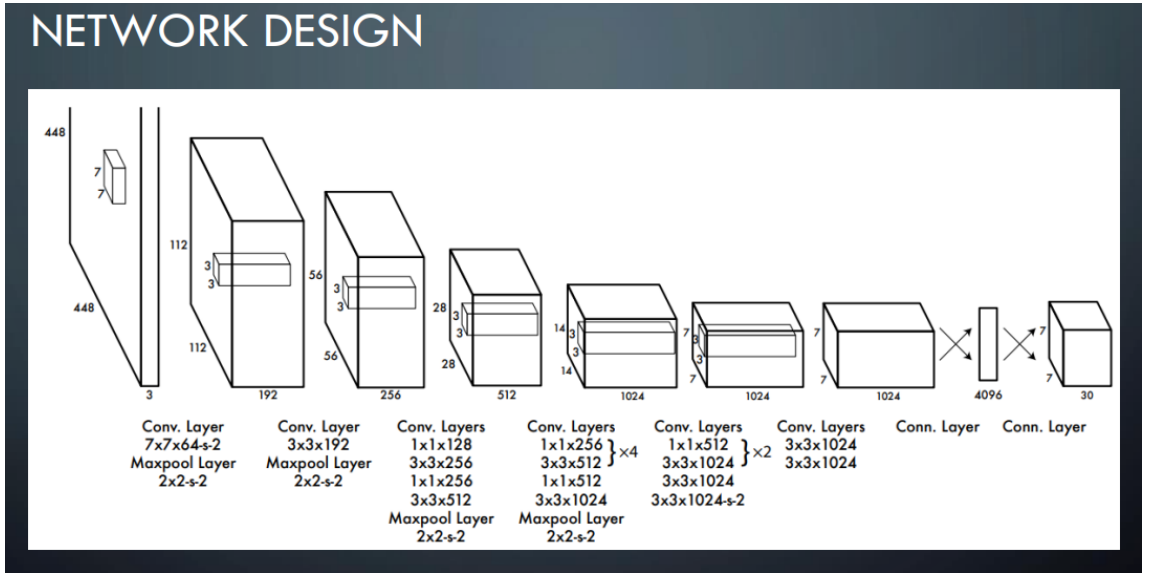
- **You Only Look Once: Unified, Real-Time Object Detection**
  https://arxiv.org/abs/1506.02640

- According to the paper:
  - You only look “once”: an one stage object detection model
  - Based on CNN
  - Real time: yolov5s can detect 140fps
  - Reason globally: see entire image, unlikely to mistake background patches in an image for objects
  - Generalizable representations

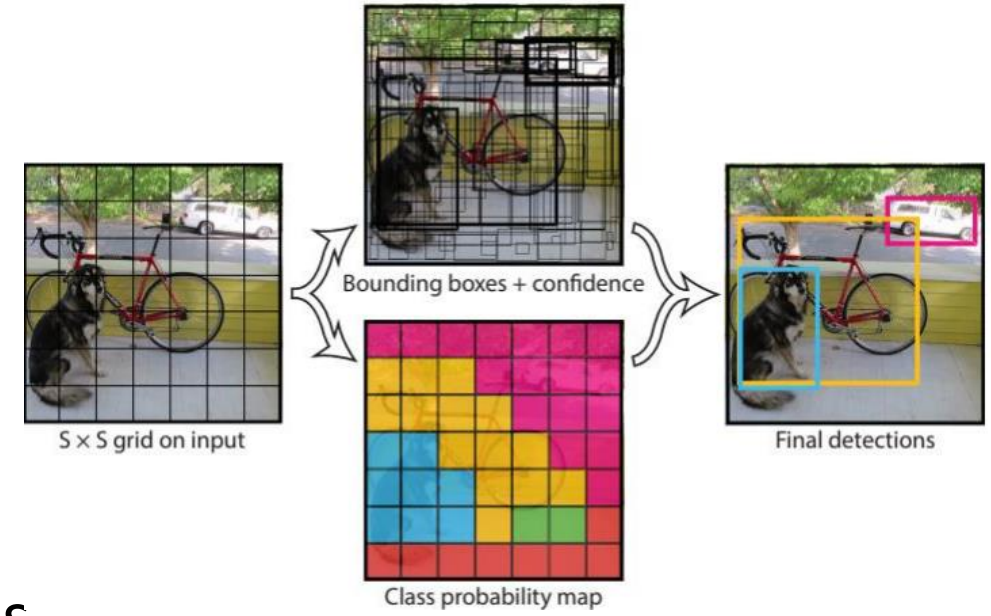
Baseline & Main Approach

- 我們使用 yolov5m 當作是 Baseline 的 model ，並且把 yolov5n 跟 yolov5s 作為 Main Approach 的 model，用於偵測圖片中的目標(奶頭)，而最後對圖片進行馬賽克的方式是相同的

- Yolov5 的 model 一開始由 CONVOLUTIONAL LAYER 用來 feature extraction
- 其中的 1＊1 CONVOLUTIONAL LAYER 是用來 Dimension reduction
- 最後接上 FULLY CONNECTED LAYER 用來 Predict the output probabilities and coordinates



Baseline & Main Approach



- Yolov5 input & output

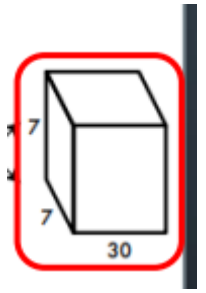
input : image

output :

- Divide the input image into $S * S$ grid cells
    - If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.
    - Each grid cell predict B bounding boxes and confidence score of these boxes.
    - Each grid cell also predicts C conditional class probabilities

Baseline & Main Approach

- OUTPUT OF YOLO FOR EACH GRID
  - Predict the bounding box
    - The center of the bounding box: x, y
    - The width w and height h are predicted relative to the whole image.
    - Confidence: $P\ Object * IOU_{pred}^{truth}$
    - IOU_{pred}^{truth} : intersection over union (IOU) between the predicted box and the ground truth.
  - Predict C conditional probability
    - $P\ (Classi \mid Object)$: conditioned on the grid cell containing an object
    - Only predict one set of class probabilities per grid cell



$S * S * (B * 5 + C)$
S = 7
2 bounding boxes
20 classes

- Output of YOLO
  - At test time we multiply the conditional class probabilities and the individual box confidence predictions
    - $P\ (Classi|Object) * P\ Object * IOU_{pred}^{truth} = P\ Classi * IOU_{pred}^{truth}$
  - YOLO can be a regression problem with $S * S * (B * 5 + C)$ output

Baseline & Main Approach

- Ground truth: $w = 2, h = 2$, predict $w = 1, h = 1$
  - If we use $(w_i - \widehat{w_i})^2 + (h_i - \widehat{h_i})^2$
    - $(2-1)^2 + (2-1)^2 = 2$
  - Use $\left(\sqrt{w_i} - \sqrt{\widehat{w_i}}\right)^2 + \left(\sqrt{h_i} - \sqrt{\widehat{h_i}}\right)^2$
    - $(1.4-1)^2 + (1.4-1)^2 = 0.32$
- Ground truth: $w = 17, h = 17$, predict $w = 16, h = 16$
  - If we use $(w_i - \widehat{w_i})^2 + (h_i - \widehat{h_i})^2$
    - $(17-16)^2 + (17-16)^2 = 2$
  - Use $\left(\sqrt{w_i} - \sqrt{\widehat{w_i}}\right)^2 + \left(\sqrt{h_i} - \sqrt{\widehat{h_i}}\right)^2$
    - $(4.1-4)^2 + (4.1-4)^2 = 0.02$

- Loss function
  - 1_{ij}^{obj} = 1 if object appears in cell i and bounding box j is responsible for this object(i.e. has the highest IOU of any predictor in that grid cell)
  - 1_{ij}^{obj} = 0 otherwise
  - $\lambda coord$ = 5
  - 1_{ij}^{noobj}: opposite to 1_{ij}^{obj}
  - $\lambda noobj$ = 0.5
  - 1_{i}^{obj} = 1 if object appears in cell i
  - 1_{i}^{obj} = 0 otherwise

Center
Width and height

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

Calculate the confidence score

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

Calculate the loss of classification

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

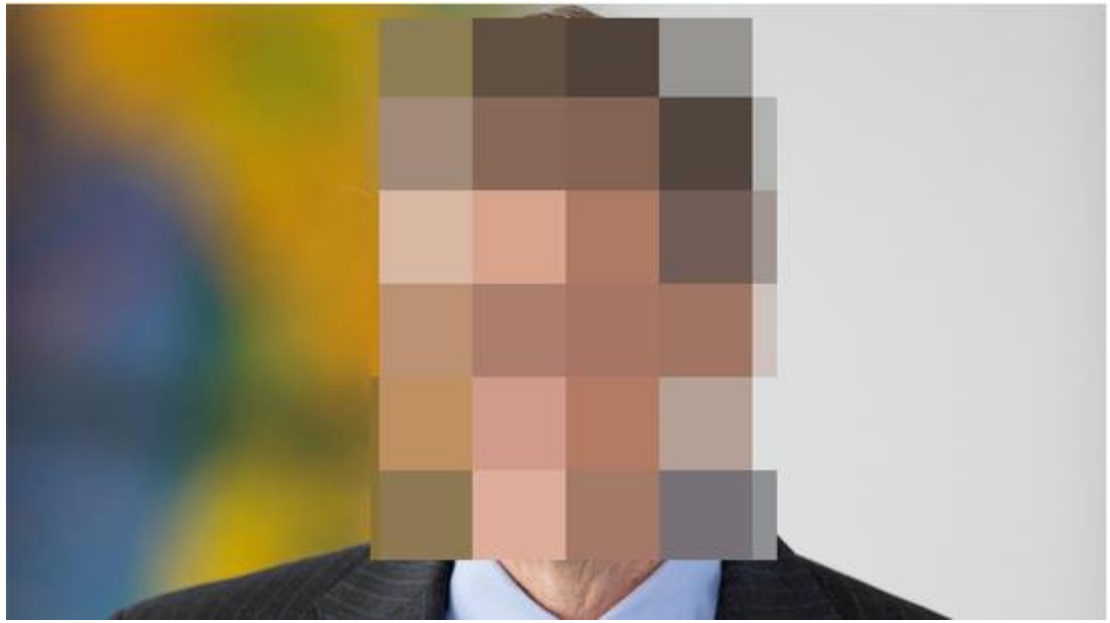
Baseline & Main Approach

- 馬賽克

  像素大小能自由調整

  每個格子以中央點的像素展開

  邊界以左上角像素展開

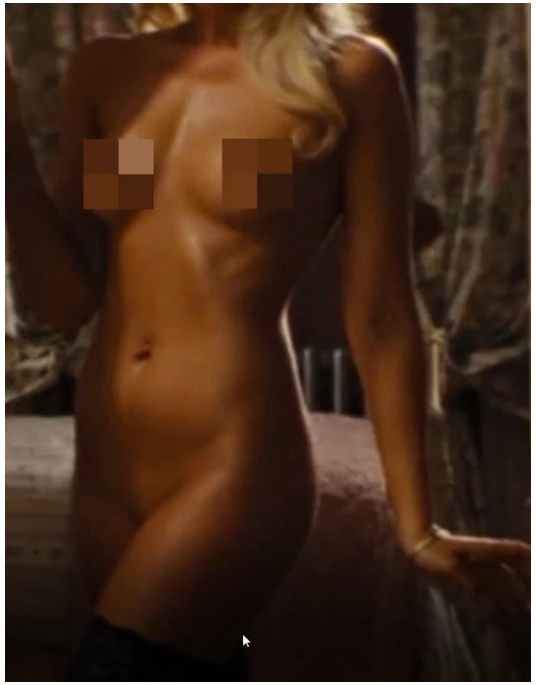


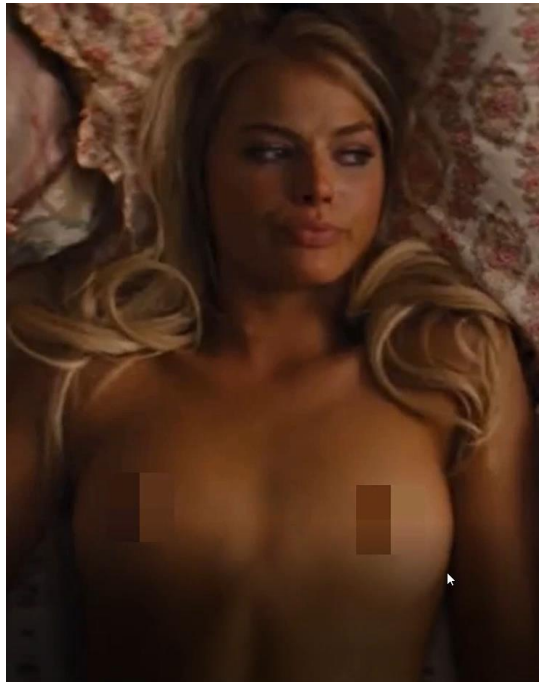
Evaluation Metric

Using 500 image as test data

```
100 epochs completed in 0.275 hours.
Optimizer stripped from runs/train/exp/weights/last.pt, 14.3MB
Optimizer stripped from runs/train/exp/weights/best.pt, 14.3MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model Summary: 213 layers, 7012822 parameters, 0 gradients
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 4/4 [00:04<00:00,  1.08s/it]
                 all        504        931      0.914       0.87        0.9      0.404
Results saved to runs/train/exp
```





從這兩個結果可以看出，我們可以成功的將影片當中所出現的奶頭成功打馬賽克，且一開始訓練的model 的準確率也高達90%以上

Results & Analysis

```
100 epochs completed in 0.470 hours.
Optimizer stripped from runs/train/exp/weights/last.pt, 42.1MB
Optimizer stripped from runs/train/exp/weights/best.pt, 42.1MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model Summary: 290 layers, 20852934 parameters, 0 gradients
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 4/4 [00:05<00:00,  1.31s/it]
                 all        504        931      0.899      0.864      0.886      0.401
Results saved to runs/train/exp
```

yolov5m

```
100 epochs completed in 0.242 hours.
Optimizer stripped from runs/train/exp/weights/last.pt, 3.8MB
Optimizer stripped from runs/train/exp/weights/best.pt, 3.8MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model Summary: 213 layers, 1760518 parameters, 0 gradients
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 4/4 [00:03<00:00,  1.01it/s]
                 all        504        931      0.913      0.846      0.886      0.404
Results saved to runs/train/exp
```

yolov5n

```
100 epochs completed in 0.275 hours.
Optimizer stripped from runs/train/exp/weights/last.pt, 14.3MB
Optimizer stripped from runs/train/exp/weights/best.pt, 14.3MB

Validating runs/train/exp/weights/best.pt...
Fusing layers...
Model Summary: 213 layers, 7012822 parameters, 0 gradients
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 4/4 [00:04<00:00,  1.08s/it]
                 all        504        931      0.914       0.87        0.9      0.404
Results saved to runs/train/exp
```

yolov5s

Results & Analysis

	P	R	mAP.5:.95	mAP@.5	Completed time (hours)
Yolov5m	0.899	0.864	0.401	0.886	0.47
Yolov5n	0.913	0.846	0.404	0.886	0.242
Yolov5s	0.914	0.87	0.404	0.9	0.275

mAP = mean Average Precision
每個分類的 AP 值的平均數 (越高越好)

- P : 準確率 = 正確數 / 預測總數 (預測的東西正確了多少百分比)
- R : 召回率 = 正確數 / 真實A類總數
- mAP@.5 : 當 IoU 為0.5時的mAP
- mAP.5:.95 : 當 IoU 為 range (0.5:0.95:0.05)時的 mAP 的平均數

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5

Yolov5 github上各版本數據

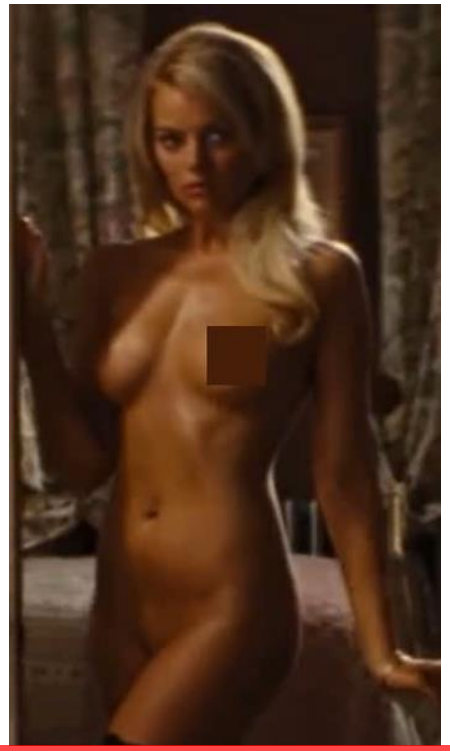
從這兩個圖表中，我們觀察到我們得到的結果與官網的圖表略有差異 :

1. mAP @.5
   (ours) s > m=n
   (github) m > s >n
2. mAP .5:.95
   (ours) s = n >m
   (github) m > s > n

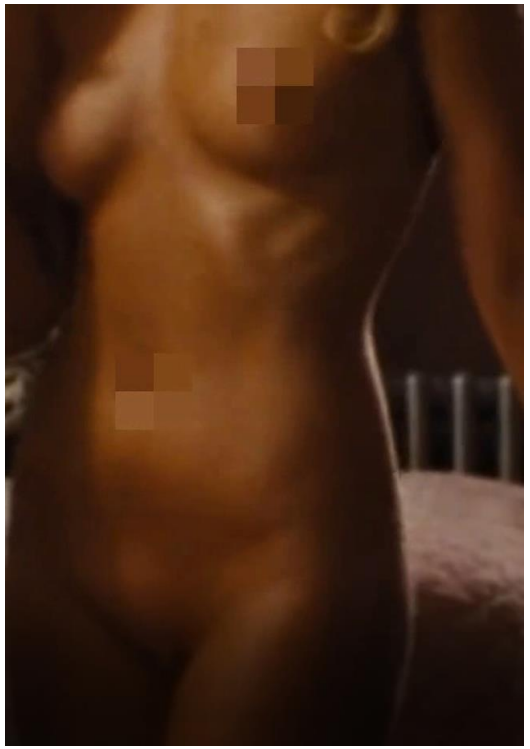
除了有可能是因為我們只要偵測單一物件讓 mAP 都比官網的數值高以外，對於這種比較簡單的偵測參數太多反而會使偵測效果變差

Error Analysis

- 從影片的一些片段我們可以觀察到，當目標物 (奶頭) 被陰影遮擋住的時候，或是整個影片亮度較暗的時候，會偵測失敗
- 而有些時候則會將肚臍這類圓形又有皮膚色的物體偵測為奶頭



當陰影遮住目標物的時候 model 會偵測不出來



Model 將肚臍誤認為是奶頭

Future Work

- 在最後輸出馬賽克影片的時候，我們發現在燈光較暗情況下，yolo 會較難去偵測目標 (奶頭)

- 所以之後除了丟更多圖片較暗的奶頭進去訓練以外，我認為可以先把影片的亮度增強在去進行偵測，再進行完馬賽克的動作後，再把影片調整回原來亮度，這樣應該有助於增強model去進行偵測。

Code

- gitHub 連結
  https://github.com/chen-yu-ye/project
- Requirements
  https://github.com/ultralytics/yolov5/blob/master/requirements.txt

detect_R18.ipynb

R18_detect.ipynb 用來訓練、偵測yolov5 的model

```
!python train.py --img 448 --batch 64 --epochs 100 --data R18.yaml  --weights yolov5s.pt --cache --exist-ok
```

```
!python detect.py —weights /content/yolov5/runs/train/exp/weights/best.pt —img 448 —conf 0.5 —save-txt —exist-ok —source
```

打馬.py

用來將以偵測完的影片打馬賽克，並以mp4檔輸出

Contribution of each member

- 葉晟育 : pixelating pictures ,testing yolov5m and collecting train & test data

- 江皓霖 : preprocessing collecting train data, making power point, recording demo video

- 何承原 : Writing the code to train yolo, pixelating pictures and collecting training data.

- 周庭右 : testing yolov5m and collecting, train & test data

References

- **You Only Look Once: Unified, Real-Time Object Detection**
  https://arxiv.org/abs/1506.02640