工科海工專論
期末報告

FACE ANTI-SPOOFING DETECTION
真假人臉識別

第二組 游振彤 劉正德 葉富銘
日期　2022/6/13

目錄

CONTENTS

Part 01

論文模型原理簡介

針對選擇之論文做講解

Searching Central Difference Convolutional Networks for Face Anti-Spoofing

Zitong Yu, Chenxu Zhao, Zezheng Wang, Yunxiao Qin,
Zhuo Su, Xiaobai Li, Feng Zhou, Guoying Zhao

- 使用CDC特徵讀取
- 可抓出如lattice artifacts的細微特徵
  (相片/影片的微小pixel)

- 使用深度圖depth map作為輸出

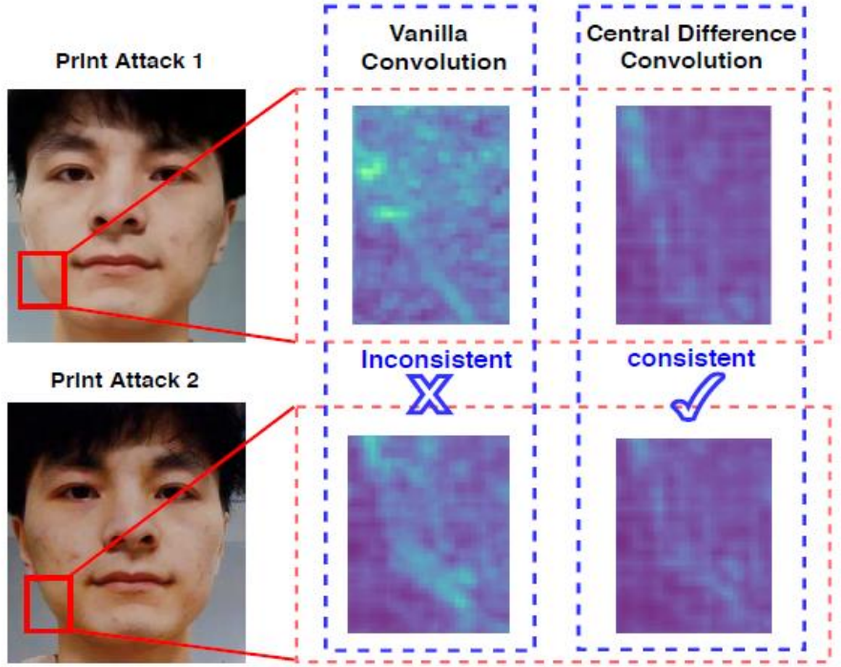


Figure 1. Feature response of vanilla convolution (VanillaConv) and central difference convolution (CDC) for spoofing faces in shifted domains (illumination & input camera). VanillaConv fails to capture the consistent spoofing pattern while CDC is able to extract the invariant detailed spoofing features, e.g., **lattice artifacts**.

CDC - Central Difference Convolution

此論文中底層的網路層

Kernel (R)

- 3×3 kernel
- $R = \{(-1, -1),(-1, 0), \cdots ,(0, 1),(1, 1)\}$.

R	**R**	**R**
R	**P_0**	**R**
R	**R**	**R**

Vanilla Convolution

- 中心與四周「梯度差」

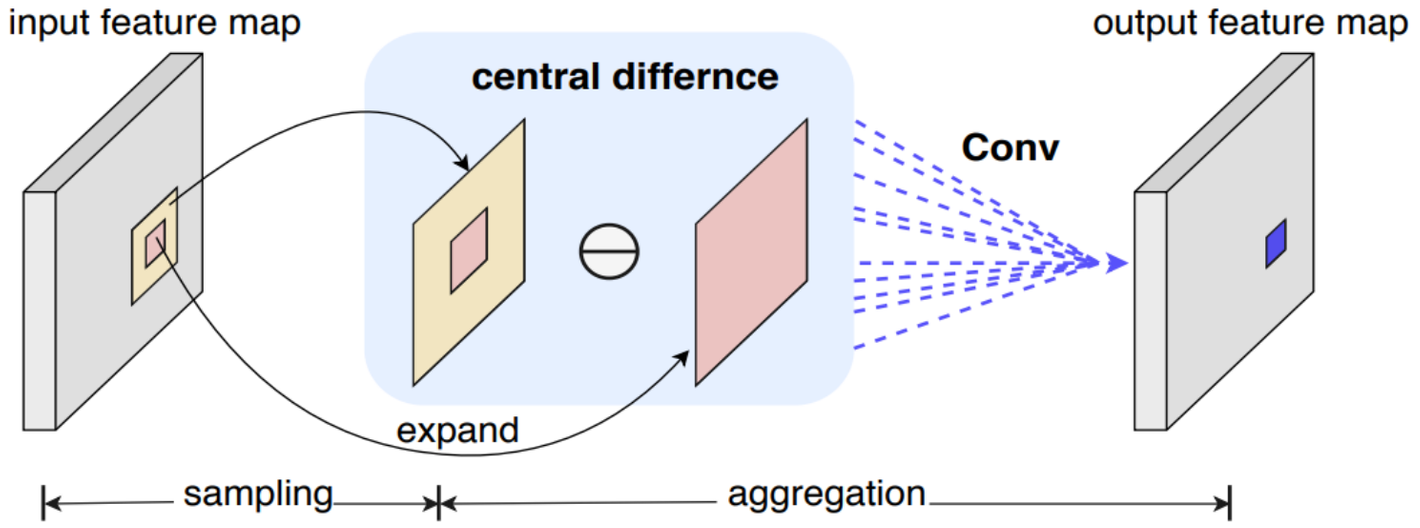
$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n)$$

Central Difference Convolution

- Inspired by “local binary pattern”(LBP)
- 忽略中心pixel

$$y(p_0) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot (x(p_0 + p_n) - x(p_0))$$

Central Difference Convolution



$$y(p_0) = \theta \cdot \underbrace{\sum_{p_n \in \mathcal{R}} w(p_n) \cdot (x(p_0 + p_n) - x(p_0))}_{\text{central difference convolution}} + (1-\theta) \cdot \underbrace{\sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n)}_{\text{vanilla convolution}},$$

=> 最後將兩者以theta比例組合

Calculations

$$y(p_0) = \underbrace{\sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_0 + p_n)}_{\text{vanilla convolution}} + \underbrace{\theta \cdot (-x(p_0) \cdot \sum_{p_n \in \mathcal{R}} w(p_n))}_{\text{central difference term}}. \tag{4}$$

- Centeral-difference (second order, with 9 parameters and a const theta for 3x3 kernel) 2D Convolution

- ## | a1 a2 a3 | | w1 w2 w3 |
- ## | a4 a5 a6 | * | w4 w5 w6 |
- ## | a7 a8 a9 | | w7 w8 w9 |
- ## --> output = \sum_{i=1}^{9}(ai * wi) - \sum_{i=1}^{9}wi * a5 --> Conv2d (k=3) - Conv2d (k=1)

- ## --> output =
- ## | a1 a2 a3 | | w1 w2 w3 |
- ## | a4 a5 a6 | * | w4 w5 w6 | - | a | * | w_sum | (kernel_size=1x1, padding=0)
- ## | a7 a8 a9 | | w7 w8 w9 |

Code

```
out_normal = self.conv(x) # vanilla convolution

if math.fabs(self.theta - 0.0) < 1e-8:
    return out_normal
else:
    #pdb.set_trace()
    [C_out,C_in, kernel_size,kernel_size] = self.conv.weight.shape
    kernel_diff = self.conv.weight.sum(2).sum(2)
    kernel_diff = kernel_diff[:, :, None, None]
    out_diff = F.conv2d(input=x, weight=kernel_diff,
            bias=self.conv.bias, stride=self.conv.stride,
                        padding=0, groups=self.conv.groups)
    # Central Difference

    return out_normal - self.theta * out_diff
```

CDCN - Central Difference Convolution Networks

For Depth-supervised

DepthNet vs CDCN

- 基本上就是用CDC去替代DepthNet中的conv layer

Level	Output	DepthNet [36]	CDCN ($\theta = 0.7$)
	256×256	3×3 conv, 64	3×3 **CDC**, 64
Low	128×128	$\begin{bmatrix} 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ conv}, 196 \\ 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ max pool} \end{bmatrix}$	$\begin{bmatrix} 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \textbf{ CDC}, 196 \\ 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \text{ max pool} \end{bmatrix}$
Mid	64×64	$\begin{bmatrix} 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ conv}, 196 \\ 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ max pool} \end{bmatrix}$	$\begin{bmatrix} 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \textbf{ CDC}, 196 \\ 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \text{ max pool} \end{bmatrix}$
High	32×32	$\begin{bmatrix} 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ conv}, 196 \\ 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ max pool} \end{bmatrix}$	$\begin{bmatrix} 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \textbf{ CDC}, 196 \\ 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \text{ max pool} \end{bmatrix}$
	32×32	[concat (Low, Mid, High), 384]	
	32×32	$\begin{bmatrix} 3 \times 3 \text{ conv}, 128 \\ 3 \times 3 \text{ conv}, 64 \\ 3 \times 3 \text{ conv}, 1 \end{bmatrix}$	$\begin{bmatrix} 3 \times 3 \textbf{ CDC}, 128 \\ 3 \times 3 \textbf{ CDC}, 64 \\ 3 \times 3 \textbf{ CDC}, 1 \end{bmatrix}$
# params		2.25×10^6	2.25×10^6

CDCN++

extended version of CDCN, consisting of the searched backbone network and

Multiscale Attention Fusion Module (MAFM)

Multiscale Attention Fusion Module (MAFM)

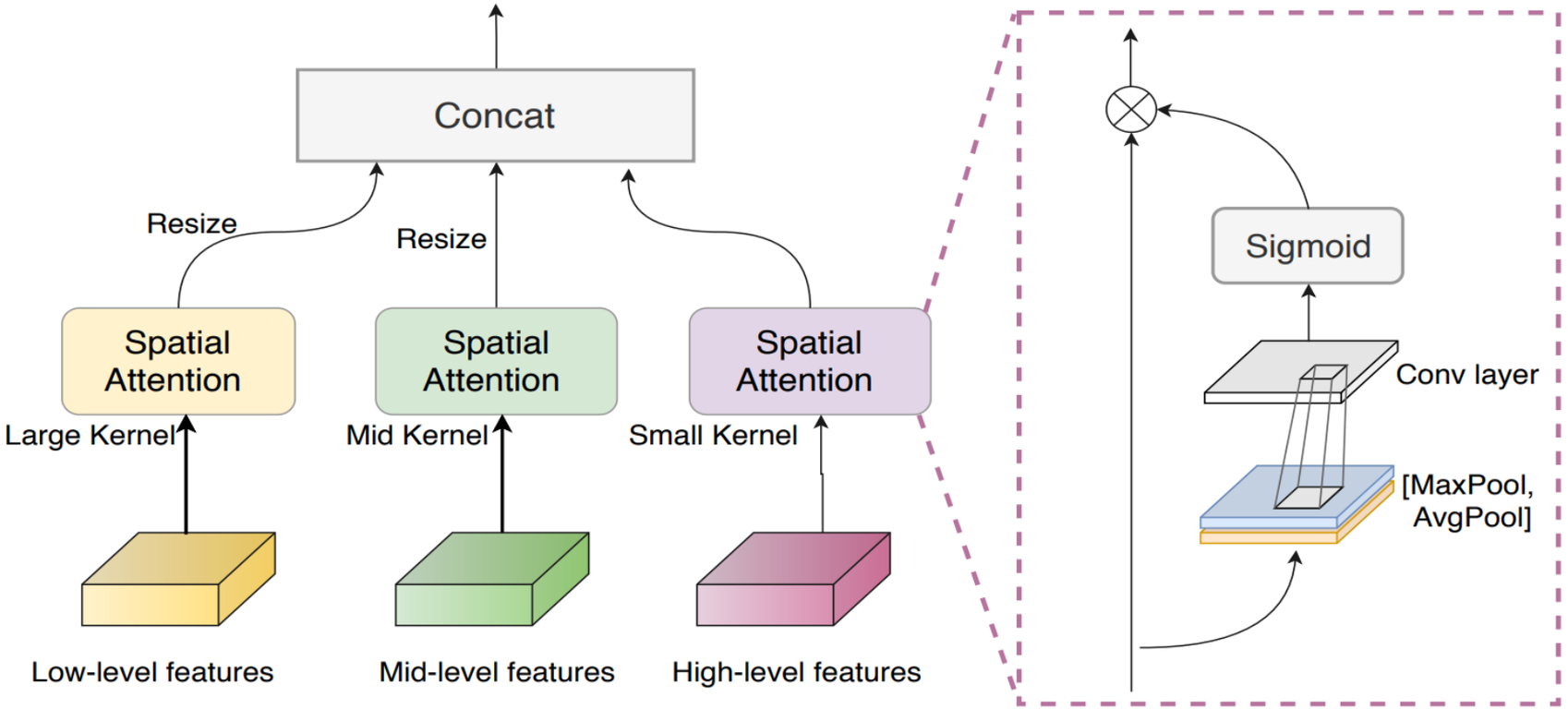
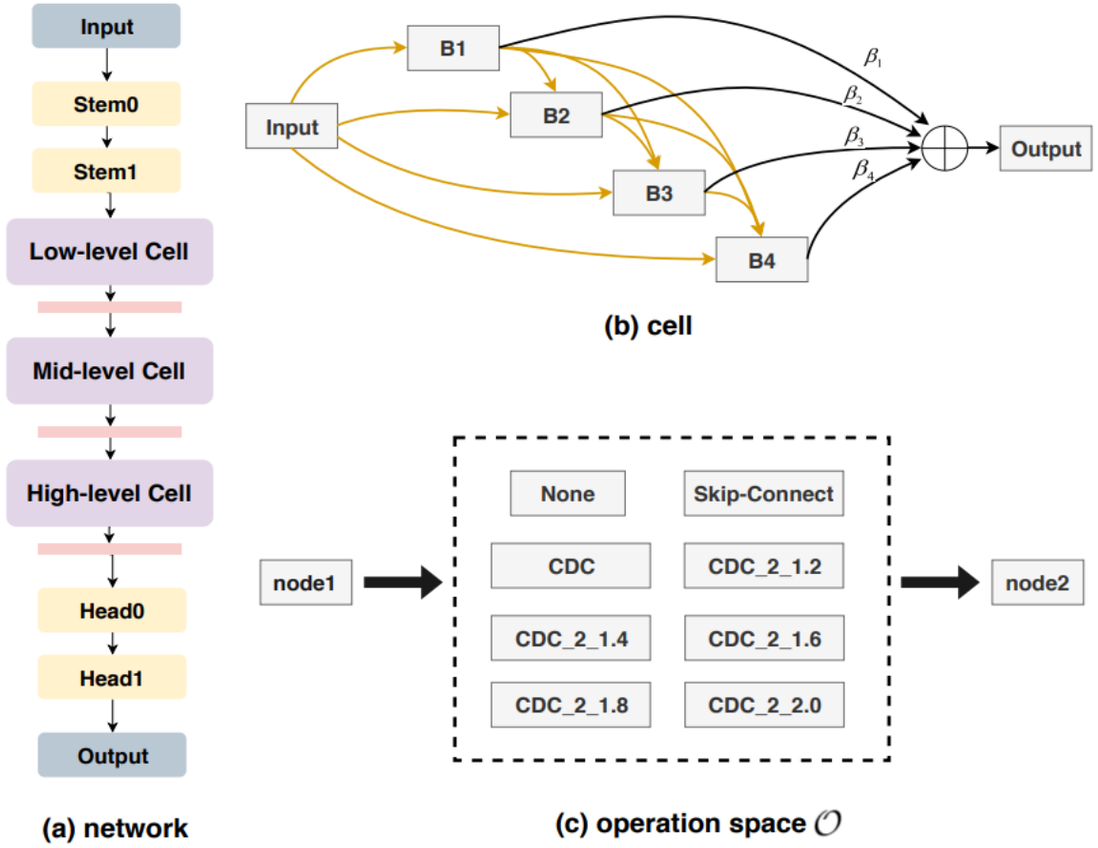


Figure 4. Multiscale Attention Fusion Module.

Total Architecture of CDCN++ (cells)



Total Architecture of CDCN++

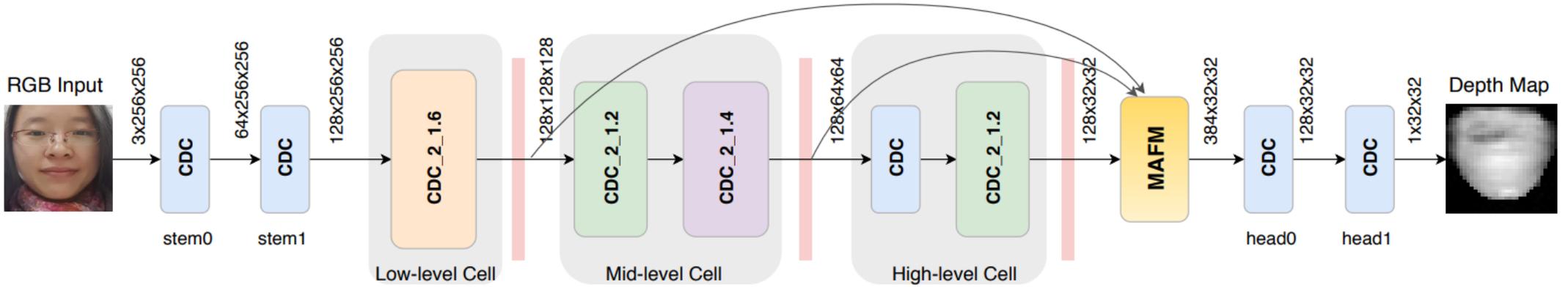


Figure 5. The architecture of CDCN++. It consists of the searched CDC backbone and MAFM. Each cell is followed by a max pool layer.

Part 02

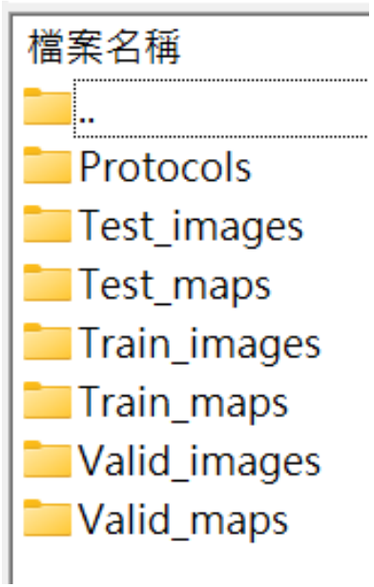
資料預處理

臉部擷取與深度圖生成

OULU & SiW

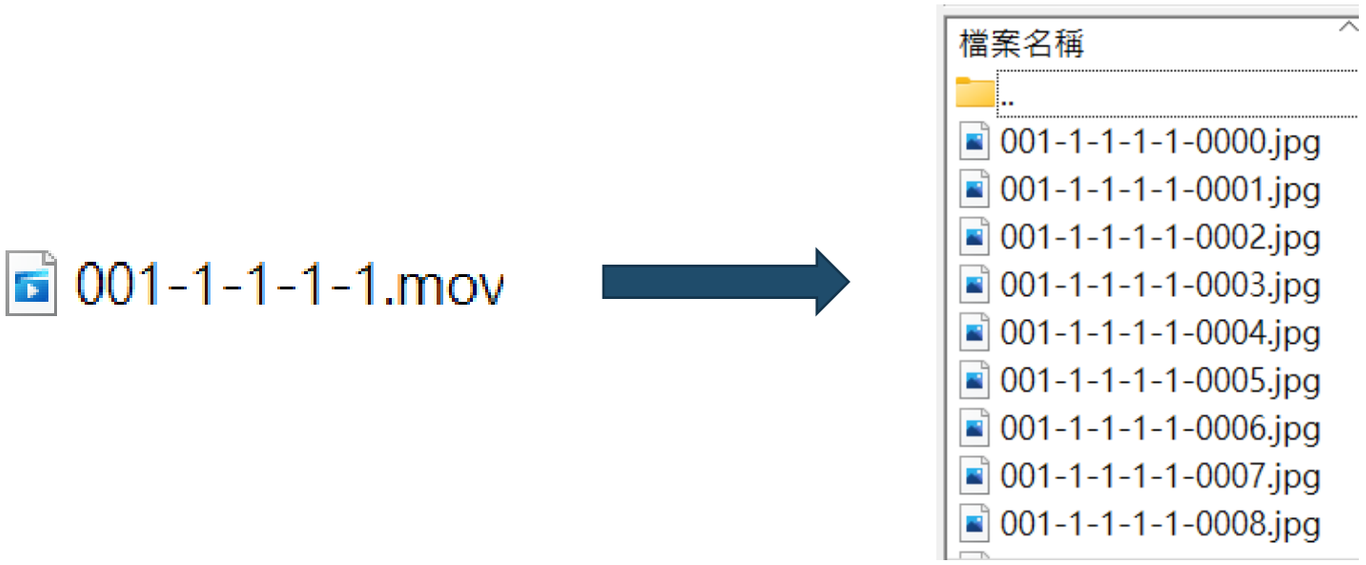
- 主要分為 Live datasets 與 Spoof datasets
- Spoof datasets 是使用印出的照片或重播的影片生成

- 處理上會分成Test/ Train/ Valid 並包含輸入images與輸出maps
- 預計處理成：



OpenCV – 擷取單幀圖像

- 擷取單幀圖像
- 官網：https://opencv.org/
- 將單個影片輸出成一個含有其所有單幀圖像的資料夾



MTCNN – 擷取人臉

- 資料集給予的 bounding box 準確率低落
- 使用於資料預處理中擷取人臉的步驟
- https://github.com/ipazc/mtcnn

Joint 3D Face Reconstruction and Dense Alignment with Position Map Regression Network

Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, Xi Zhou

- 本篇論文純使用其PRNet中偵測深度圖 (Depth map) 的功能
- 產生出dataset中各張真實照片對應的輸出 (32*32的深度圖)
- 訓練時Spoof image的灰度則會設為0 (全黑)

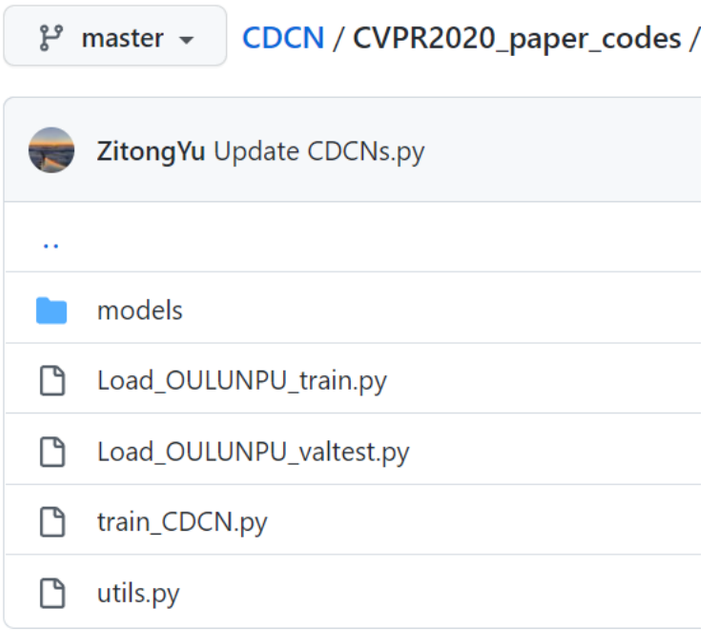
Part 03 實際訓練

train_CDCN.py

CDCN模型

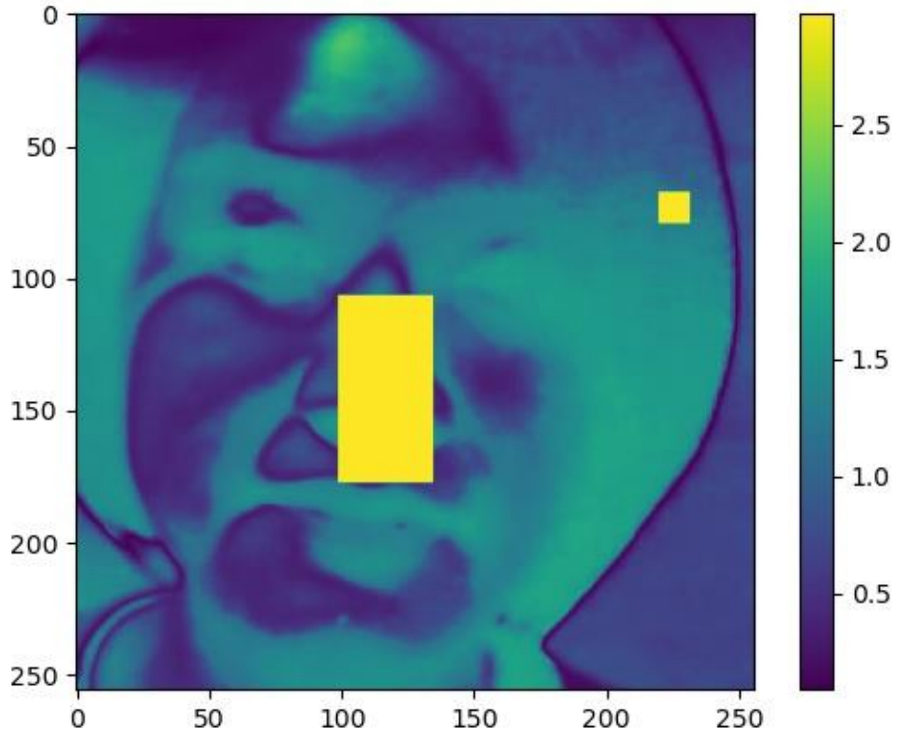
- GitHub - ZitongYu/CDCN: Central Difference Convolutional Networks (CVPR’ 20)

- Load_(dataset)NPU_train/valtest.py 為讀取資料集的程式碼
- train_CDCN.py 為訓練與測試的程式碼
- Models內有CDCNs.py則是用以model的內容



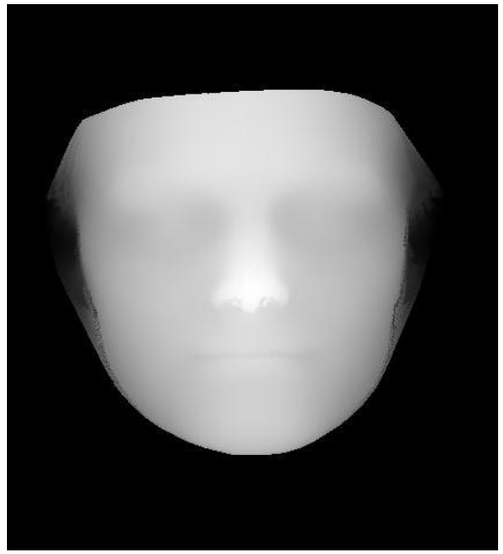
Augmentations – 加強資料集

- RandomErasing
- RandomHorizontalFlip
- ToTensor
- Cutout
- Normaliztion



訓練 Loss 定義

- 將模型參數所生成之灰度圖與正確灰度圖比較
- 繼而得出深度的分數差距
- 使用兩種loss進行訓練
  - Absolute_Depth_loss
  - Contrastive_Depth_loss



最終分數(dataset performance)

混淆矩陣 Confusion Matrix

	實際yes	實際no
預測yes	TP (True Positive)	FP Type I Error (False Positive)
預測no	FN Type II Error (False Negative)	TN (True Negative)

Yes : 真臉　No : 假臉

最終分數(dataset performance)

計算方式如下

- Attack Presentation Classification Error Rate(APCER):

  APCER = FN / (TP + FN)

- Bona Fide Presentation Classification Error Rate(BPCER):

  BPCER = FP / (FP + TN)

- Average Classification Error Rate (ACER):

  ACER = (APCER + BPCER) / 2

(原github程式碼已有計算方式)

訓練成果 – OULU datasets

	APCER(%)	BPCER(%)	ACER(%)
Protocol I	0.00	7.50	3.75
Protocol II	0.28	1.11	0.69
Protocol III	0.00	3.33	1.67
Protocol IV	5.00	15.00	10.00

訓練成果 – SiW datasets

	APCER(%)	BPCER(%)	ACER(%)
Protocol I	0.00	1.00	0.50
Protocol II	0.00	0.00	0.00
Protocol III	0.00	0.00	0.00

訓練成果 – Cross datasets

	APCER(%)	BPCER(%)	ACER(%)
SiW Protocol II on OULU Protocol III	0.00	10.00	5.00
OULU Protocol III on SiW Protocol II	2.81	3.67	3.24

⇒ SiW 的真臉辨識率低、泛用性可能較差(也有可能是protocol選擇的問題)

⇒ OULU較為穩定

Part 04

樹梅派實作

尝試實作樹梅派

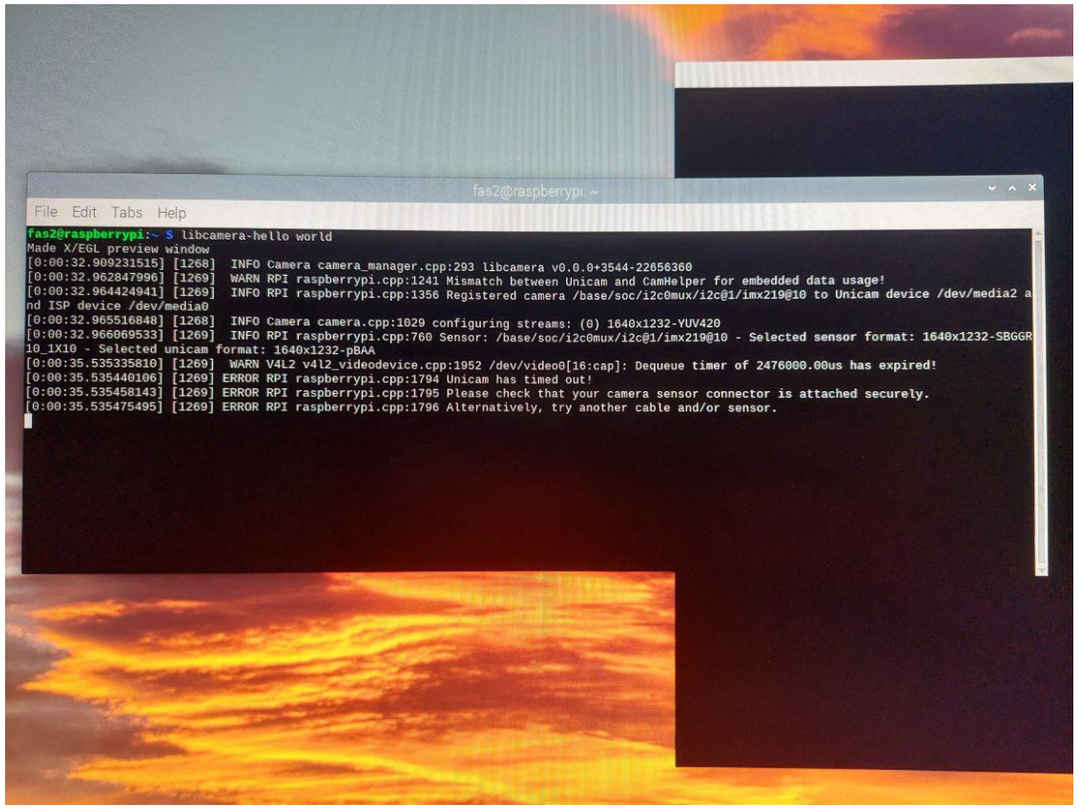
樹梅派實作

- 使用全部資料（OULU+SiW）進行模型訓練
- APCER/ BPCER/ ACER 皆趨近於零（但不是像SiW P2, P3全部為0)

```
), Val:  val_threshold= 0.5164, val_ACC= 0.9950, val_ACER= 0.0063
, Test:  ACC= 0.9987, APCER= 0.0000, BPCER= 0.0033, ACER= 0.0017
```

樹梅派實作

- 昨天在實作時發現攝影機與樹梅派無法連線



- 詢問第一組的執行狀況
- 發現會由於模型過於龐大導致**樹梅派需要經過長時間的運算才能得出結果**

Part 05 結論心得與分工

結論

- 模型表現得好，但由於模型過於龐大導致**樹梅派成效不佳**
- 使用於有一定規格的硬體上應該能得到很好的效果，如手機，筆電等等

- 其他討論:
- 實作的可行解法

  詢問第一組實做的情形，發現他們會遇到結果不斷跳動的狀況，我們自己組別在討論後認為有可能是影像輸入畫質差異或是人在移動中產生影像不完全的問題，但不確定在縮小 / 放大的過程中是否會影響CDC對於圖片的判斷

- 提高準確率方法

  可以採用幾幀內正確率達一定數才能通過(但同時會降低確認的速度)

心得

在此次的專論中我們遇到許多方面的難題，從最初的一頭霧水，到中期的各種問題浮現，比如截圖太慢，還要去找其他方法或是利用cyphon加速的方案，都十足的讓我們體驗到實際在執行一個專題企畫會遇到的各種問題。此外後期還遇到各種原因導致我們的進度延宕，導致老師和助教又要多花時間聽取我們的報告，這部份真的非常抱歉。

雖然在過程中遇到許多令我們燒破腦袋的問題，但最終產生出的報告還是讓我們有種看著孩子成長的心情，雖然最後的樹梅派沒有成功產出，但也很開心今年有多元的專論主題，讓我們能體驗到這種過去應該很難體驗到的題目，最後再次謝謝老師以及助教提供的幫助！

期末專題分工

游振彤 B08505018

資料深度圖生成

模型訓練 樹梅派實做(Fail)

葉富銘 B08505045

SiW 資料前期處理 協助模型訓練

劉正德 B08505022

OULU 資料前期處理 協助模型訓練

資料夾路徑

SiW:
/home/fas2/Train/SiW_Train
OULU:
/home/fas2/Train/OULU_Train
Cross:
/home/fas2/Train/Cross_Test

THANK YOU

感謝聆聽！