8.8、人脸识别

8.8.1、实验目标

本节课主要学习人脸识别功能，比人脸检测功能多了识别效果。

本次实验的参考代码路径为：K210_Broad\05-AI\face_recog.py

8.8.2、实验前准备

请先将模型文件导入内存卡上，再将内存卡插入到K210模块的内存卡插槽上。具体操作步骤请参考：

附录：导入模型文件到内存卡

8.8.3、实验过程

模块的出厂固件已经集成AI视觉算法模块，如果下载过其他固件，请烧录回出厂固件再进行实验。

1. 导入相关库，并初始化摄像头和LCD显示屏。

```
from maix import GPIO, utils
from fpioa_manager import fm
from board import board_info

lcd.init()
sensor.reset()
sensor.set_pixformat(sensor.RGB565)
sensor.set_framesize(sensor.QVGA)
sensor.skip_frames(time = 100)
clock = time.clock()
```

2. 新建人脸特性图像大小为64*64。

```
feature_img = image.Image(size=(64,64), copy_to_fb=False)
feature_img.pix_to_ai()

FACE_PIC_SIZE = 64
dst_point =[(int(38.2946 * FACE_PIC_SIZE / 112), int(51.6963 * FACE_PIC_SIZE /
112)),
            (int(73.5318 * FACE_PIC_SIZE / 112), int(51.5014 * FACE_PIC_SIZE /
112)),
            (int(56.0252 * FACE_PIC_SIZE / 112), int(71.7366 * FACE_PIC_SIZE /
112)),
            (int(41.5493 * FACE_PIC_SIZE / 112), int(92.3655 * FACE_PIC_SIZE /
112)),
            (int(70.7299 * FACE_PIC_SIZE / 112), int(92.2041 * FACE_PIC_SIZE /
112)) ]
```

3. 初始化人脸检测模型相关的参数，模型文件路径
   为：/sd/KPU/yolo_face_detect/face_detect_320x240.kmodel，并使用yolo2来计算是否符合模型要求。

```
anchor = (0.1075, 0.126875, 0.126875, 0.175, 0.1465625, 0.2246875, 0.1953125,
0.25375, 0.2440625, 0.351875, 0.341875, 0.4721875, 0.5078125, 0.6696875,
0.8984375, 1.099687, 2.129062, 2.425937)
kpu = KPU()
kpu.load_kmodel("/sd/KPU/yolo_face_detect/face_detect_320x240.kmodel")
kpu.init_yolo2(anchor, anchor_num=9, img_w=320, img_h=240, net_w=320 , net_h=240
,layer_w=10 ,layer_h=8, threshold=0.7, nms_value=0.2, classes=1)
```

4. 初始化ld5模型相关参数，模型的地址为：/sd/KPU/face_recognization/ld5.kmodel。

```
ld5_kpu = KPU()
print("ready load model")
ld5_kpu.load_kmodel("/sd/KPU/face_recognization/ld5.kmodel")
```

5. 初始化特征模型相关参数，模型的地址
   为：/sd/KPU/face_recognization/feature_extraction.kmodel。

```
fea_kpu = KPU()
print("ready load model")
fea_kpu.load_kmodel("/sd/KPU/face_recognization/feature_extraction.kmodel")
```

6. 新建按键功能，上升沿触发，主要功能是记录要识别的人脸。

```
start_processing = False
BOUNCE_PROTECTION = 50

fm.register(board_info.BOOT_KEY, fm.fpioa.GPIOHS0)
key_gpio = GPIO(GPIO.GPIOHS0, GPIO.IN)
def set_key_state(*_):
    global start_processing
    start_processing = True
    time.sleep_ms(BOUNCE_PROTECTION)
key_gpio.irq(set_key_state, GPIO.IRQ_RISING, GPIO.WAKEUP_NOT_SUPPORT)
```

7. 新建人脸识别变量，其中，变量`record_ftrs`：表示识别人脸特征的数组；变量`THRESHOLD`：表示人脸识别的阈值，超过此值则认为是识别到的人脸；变量`recog_flag`：表示检测到的人脸是否已识别的状态，已识别为True，未识别为Flase。

```
record_ftrs = []
THRESHOLD = 80.5
recog_flag = False
```

8. 提取检测到的人脸的信息。

```
def extend_box(x, y, w, h, scale):
    x1_t = x - scale*w
    x2_t = x + w + scale*w
    y1_t = y - scale*h
    y2_t = y + h + scale*h
    x1 = int(x1_t) if x1_t>1 else 1
    x2 = int(x2_t) if x2_t<320 else 319
    y1 = int(y1_t) if y1_t>1 else 1
    y2 = int(y2_t) if y2_t<240 else 239
    cut_img_w = x2-x1+1
    cut_img_h = y2-y1+1
    return x1, y1, cut_img_w, cut_img_h
```

9. 新建while循环，主要功能是先检测出人脸，当识别到BOOT按键按下时，记录人脸信息。当检测到的人脸信息大于识别阈值，就表示是已识别的人脸，用绿色方框，并显示识别分数，否则就表示未识别的人脸，用白色方框。

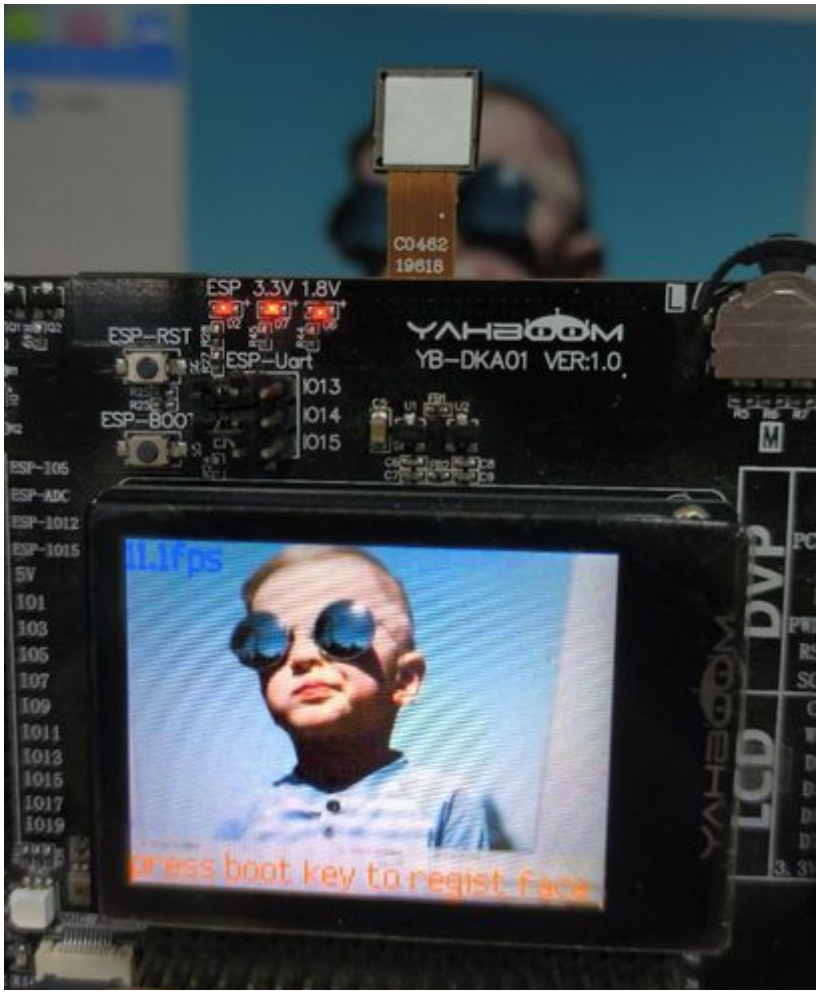
```
while True:
    gc.collect()
    # print("mem free:",gc.mem_free())
    # print("heap free:",utils.heap_free())
    clock.tick()
    img = sensor.snapshot()
    kpu.run_with_output(img)
    dect = kpu.regionlayer_yolo2()
    fps = clock.fps()
    if len(dect) > 0:
        for l in dect :
            x1, y1, cut_img_w, cut_img_h= extend_box(l[0], l[1], l[2], l[3],
scale=0)
            face_cut = img.cut(x1, y1, cut_img_w, cut_img_h)
            face_cut_128 = face_cut.resize(128, 128)
            face_cut_128.pix_to_ai()
            out = ld5_kpu.run_with_output(face_cut_128, getlist=True)
            face_key_point = []
            for j in range(5):
                x = int(KPU.sigmoid(out[2 * j])*cut_img_w + x1)
                y = int(KPU.sigmoid(out[2 * j + 1])*cut_img_h + y1)
                face_key_point.append((x,y))
            T = image.get_affine_transform(face_key_point, dst_point)
            image.warp_affine_ai(img, feature_img, T)
            feature = fea_kpu.run_with_output(feature_img, get_feature = True)
            del face_key_point
            scores = []
            for j in range(len(record_ftrs)):
```

```
                score = kpu.feature_compare(record_ftrs[j], feature)
                scores.append(score)
            if len(scores):
                max_score = max(scores)
                index = scores.index(max_score)
                if max_score > THRESHOLD:
                    img.draw_string(0, 195, "persion:%d,score:%2.1f" %(index,
max_score), color=(0, 255, 0), scale=2)
                    recog_flag = True
                else:
                    img.draw_string(0, 195, "unregistered,score:%2.1f" %
(max_score), color=(255, 0, 0), scale=2)
            del scores
            if start_processing:
                record_ftrs.append(feature)
                print("record_ftrs:%d" % len(record_ftrs))
                start_processing = False

            if recog_flag:
                img.draw_rectangle(l[0],l[1],l[2],l[3], color=(0, 255, 0))
                recog_flag = False
            else:
                img.draw_rectangle(l[0],l[1],l[2],l[3], color=(255, 255, 255))
            del (face_cut_128)
            del (face_cut)

    img.draw_string(0, 0, "%2.1ffps" %(fps), color=(0, 60, 255), scale=2.0)
    img.draw_string(0, 215, "press boot key to regist face", color=(255, 100, 0),
scale=2.0)
    lcd.display(img)
```

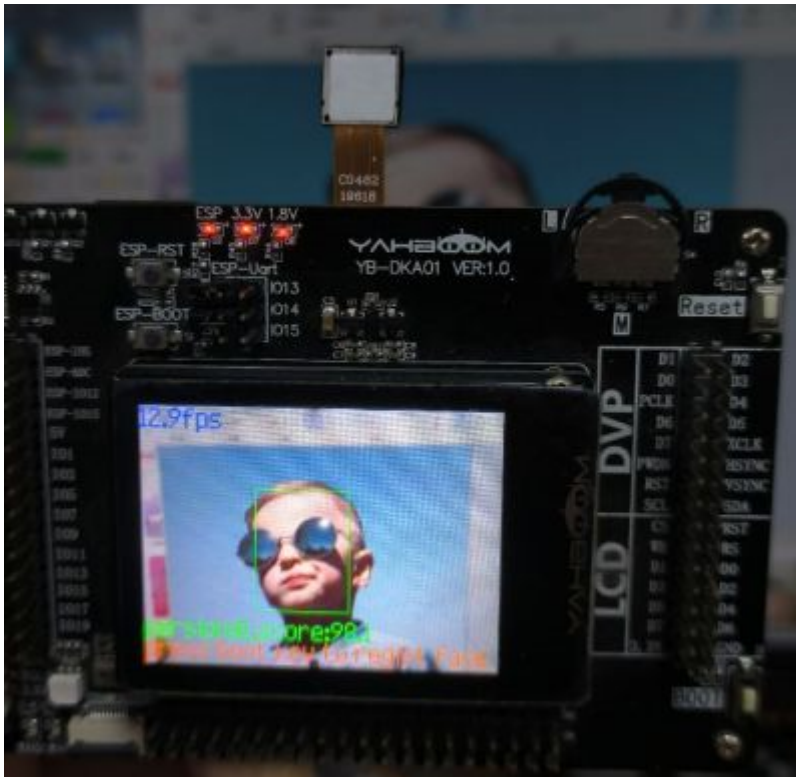
8.8.4、实验效果

由于需要用到BOOT按键，不要在CanMV IDE里直接运行，CanMV IDE目前无法检测到BOOT按键，请将代码作为main.py下载到K210开发板上运行。

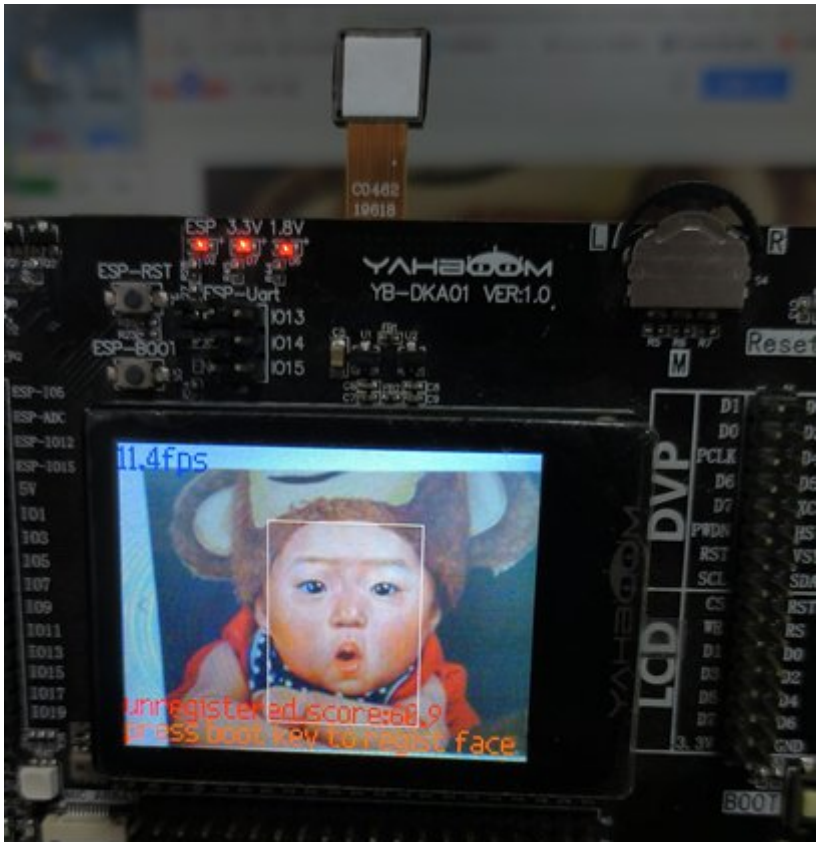
将摄像头对着人脸，如果是未识别的人脸，会显示白色的方框。



此时按一下右上角的BOOT按键，记录当前人脸信息，白色边框变为绿色边框，并显示识别到的分数。



识别到未按下boot按键注册进人脸库的结果如下：



8.8.5、实验总结

人脸识别需要用的内存卡加载模型文件，所以需要提前将模型文件导入内存卡，再将内存卡插入K210开发板的内存卡卡槽里，如果无法读取到内存卡里的模型文件，则会报错。

由于人脸识别需要用到BOOT按键，所以不要在CanMV IDE运行人脸识别代码，请将代码作为main.py下载到K210芯片上，然后按复位键 开始运行。

识别人脸是按照顺序排序的，第一个识别的人为person:0，第二个识别的人为person:1，依此类推。