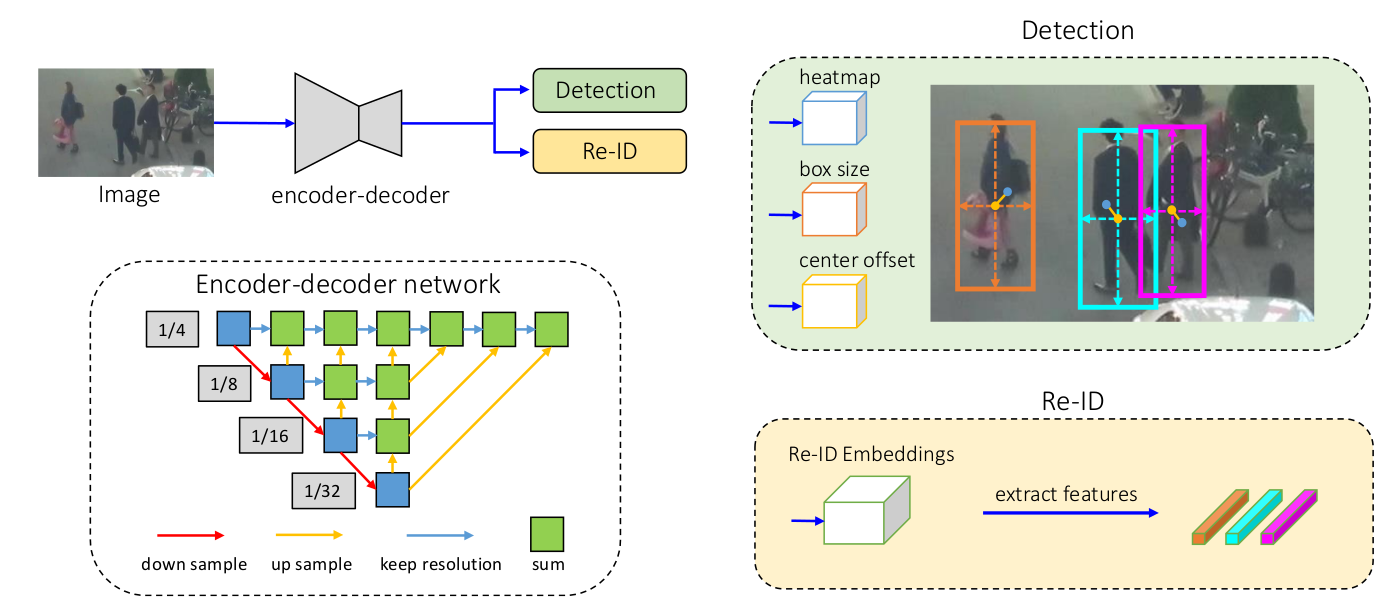
学习笔记

**FairMOT：**

****

reID：

re-identification (re-ID)

affinity among different objects should be smaller than that between same objects

摘要：

Multi-object tracking 一般先obj detection 再生成reID，所以过去的模型对于reID部分不fair

借助anchor-free object detection architecture CenterNet，再加上reID模块完成工作

两阶段问题：

* anchor不适宜reID：competition、ambiguity
* two tasks需要不同的层次的features
* reID的demension更大

主要贡献：

* 分析了之前两阶段模型的问题
* 提出了基于centernet的fairMOT

数据集：

~~2DMOT15 ，MOT16, MOT17，MOT20用作test~~

ETH、CityPerson用于detection（只有bbox信息）

Calech、~~MOT17~~、CUHK-SYSU、PRW可以train两个分支

CrowdHuman用于selfsupervised training

用COCO pretrain

模型详细信息：

backbone：resnet34

detection branch：centernet，生成heatmaps、center、bbox

heatmaps head算一个focal loss，center和bbox size算一个l1 loss

reID branch：通过一个conv算出全部的reID，只使用center的feature，来自detection分支

使用分类任务的数据集训练，将reID映射到class type，相同class的数据被认为是同一个物体

training：三个loss jointly training，使用一个uncertainty loss组合detection和identity

todo：

查看一下tracking数据集

数据集整理

* CUHK-SYSU在等回复  
  <http://www.ee.cuhk.edu.hk/~xgwang/PS/dataset.html>  
  <http://www.ee.cuhk.edu.hk/~xgwang/PS/paper.pdf>  
  <https://github.com/ShuangLI59/person_search>  
  <https://www.payititi.com/opendatasets/show/39/>有下载链接
* Caltech在等回复  
  <https://zhuanlan.zhihu.com/p/360827742> 有下载链接  
  <http://www.vision.caltech.edu/datasets/>

挺好的数据集，图片很多，分辨率很高

有很多人已经把他转换成了pascaloc格式，而googleapi可以直接读pascal到tfrecord

* PRW，发邮件问了

<https://blog.csdn.net/songwsx/article/details/102536374>

* Duke MTMC 全部是人  
  <https://blog.csdn.net/qq_39220334/article/details/121483223> 有下载链接

就是train / test / query 三个list，图片尺寸很小，并且是crop出来的，感觉不太能用  
有人商用，但是有些隐私问题，并且license是写着不能呢个商用的

* DanceReID 跳舞的  
  https://github.com/azuxmioy/DanceReID

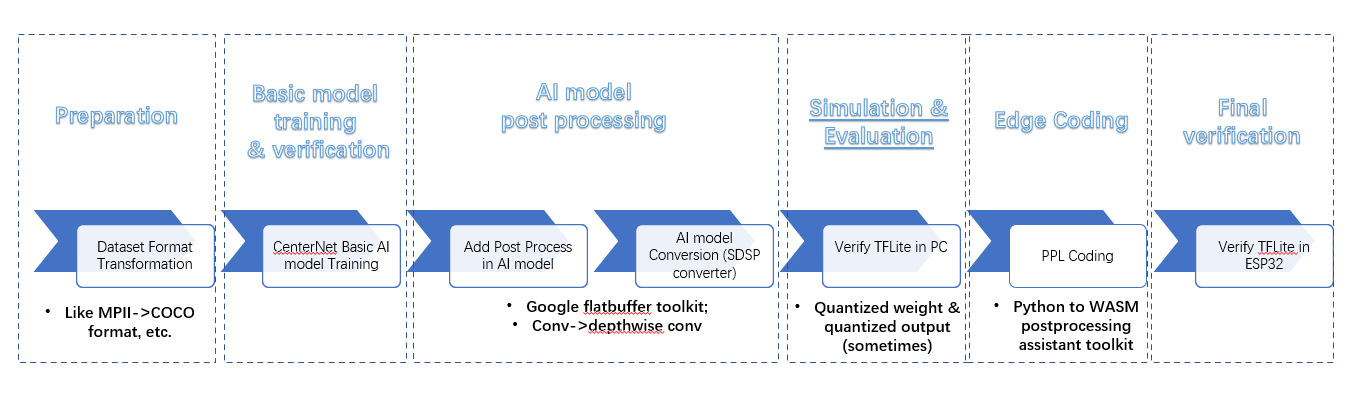
<https://zhuanlan.zhihu.com/p/153103174>

<https://blog.csdn.net/felaim/article/details/108383576>

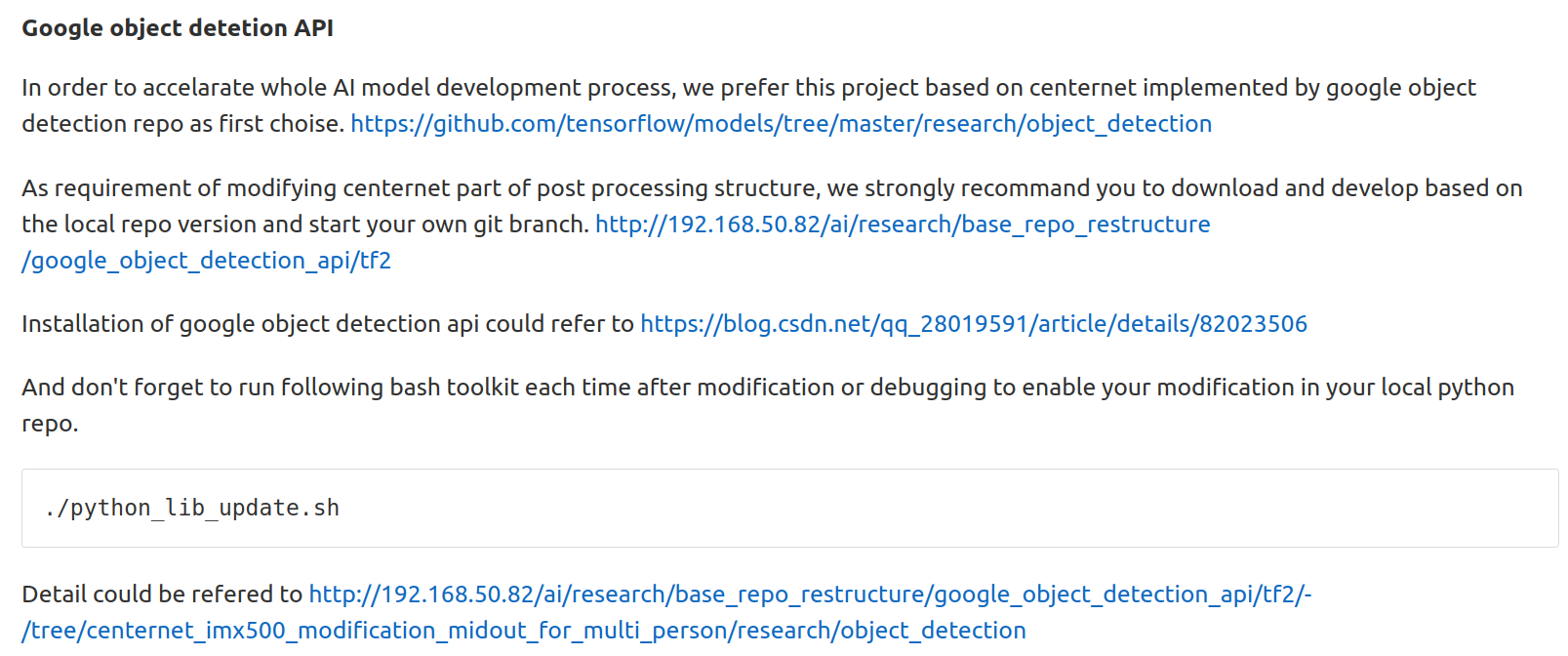
<https://github.com/Zhongdao/Towards-Realtime-MOT/blob/master/DATASET_ZOO.md>

这里有所有数据集的下载链接

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset name** | **Nums of frames** | **Release time** | **Bbox + reID** | **commercial** |
| CUHK-SYSU | 18184 imgs  8,432 ids | 2017 | Y | Asked |
| Caltech Pedestrain | 250,000 imgs  2300 ids | 2009 | Y | Asked |
| PRW | / | 2017 | Y | Asked |
| Duke MTMC | 36,411 imgs  1812 ids | 2017 | Y | Y |

****

**Google API**

* clone 到本地拉个新的分支
* center\_net\_meta\_arch记录了主要结构
  + class TrackParams 记录了一些param元组
  + class CenterNetMetaArch 初始化一个centerne

有tracking相关的layers和loss计算，也有后处理

* doc文件夹里的configure\_pipeline说config文件实际上对应的是protos文件
* 数据集的labelmap实际上是class id和class name的对应
* inputs里的train\_input函数负责处理input数据
* example decoder里对于trackid的处理  
   'image/object/track/label'
* centernet\_proto文件看一下如何写config文件

docker

docker run -it --name mwt\_dataset --ipc=host -e DISPLAY=$DISPLAY -v ~/Desktop/dataset:/workspace/dataset -p 8023:22 --device=/dev/video0:/dev/video0 --privileged=true --gpus all jym\_tf /bin/bash

docker run -it --name mwt\_workspace --ipc=host -e DISPLAY=unix$DISPLAY -v ~/Desktop:/workspace -p 8765:22 --device=/dev/video0:/dev/video0 -v /tmp/.X11-unix:/tmp/.X11-unix --privileged=true --gpus all jym\_tf /bin/bash

sudo docker run -it --name pc\_simulation --ipc=host -e DISPLAY=unix$DISPLAY -v ~/桌面/pc\_simulation:/workspace -p 7654:22 --device=/dev/video0:/dev/video0 -v /tmp/.X11-unix:/tmp/.X11-unix --privileged=true 192.168.50.82:5000/tensorflow\_xnnpack\_tflite\_x86\_accelerate:0.3 /bin/bash

docker权限解决：

<https://blog.csdn.net/weixin_43321041/article/details/120399194>

Caltech Pedestrain

1. 使用scipy.io的loadmat完成数据读取，读出来是一个dict，里面有  
   dict\_keys(['\_\_header\_\_', '\_\_version\_\_', '\_\_globals\_\_', 'A', 'vers'])等keys
2. 数据都存在’A’里，numpy的格式  
   vbb['A'][0][0][0] [0][0]是有多少张图片  
   vbb['A'][0][0][1] [0]里就顺序存储着每张图片的信息  
   vbb['A'][0][0][2] [0]是47，人员数量  
   vbb['A'][0][0][3] [0]是47个1  
   vbb['A'][0][0][4] [0]是47张图片但是person还是people  
   vbb['A'][0][0][5/6] [0]是47个数  
   vbb['A'][0][0][7] [0]是47个0  
   vbb['A'][0][0][8] [0]是一个0  
   还有9和10
3. 每张图片的信息包括  
   dtype=[('id', 'O'), ('pos', 'O'), ('occl', 'O'), ('lock', 'O'), ('posv', 'O')] pos为bbox

tfrecord转换

1. 所有文件格式转换成的tfrecord层次都是一样的
2. 需要修改dataset文件结构，以及重写一个读xml函数

将google-api 打包

* 首先需要使用一个protoc库将所有proto文件弄成.py，接下来在object-detection/packages/tf2目录下运行python setup.py build 和 python setup.py install，完成包的安装
* setup.py 里声明了所需要的依赖库，以及所打包的结构信息和版本信息
* 可以在api的文件里进行修改，之后直接运行lib\_update.sh即可

tensorboard

* 需要重启一下ssh服务，之后通过vscode直接连接即可

NETRON

* NETRON是一个用js写的可以可视化网络结构的工具，他支持很多模型格式，比如TF，TFLite，和pytorch script等

TFLite转换

* 跑export\_tflite\_graph\_tf2.py，生成pb文件
* 输入的image需要自己做preprocess，比如normalize，看object\_detection/models的preprocess函数
* 跑pb2tflite.py完成quantization和到tflite的转化

Estimated count of arithmetic ops: 44.361 G ops, equivalently 2.180 G MACs

* 写一个后处理，里面包含图片生成，加入reID，验证转换的正确性

X

* 本地已经默认开启了X服务
* 在docker上安装了xclock程序  
  apt install x11-apps

PIPELINE

1. 数据集准备，需要满足规定格式

不满足需要自己写脚本转化

1. 调用google api的脚本完成到tfrecord的转化
2. 根据需求修改google api
3. 运行lib\_update完成打包
4. 运行train.sh训练
5. 运行eval.sh测试
6. 运行tf2pb转换模型成.pb
7. 运行pb2tflite完成quant和tflite转换
8. 运行tf\_lite.py进行simulation

bbox的后处理

1. eval分析
   1. 输出的是scale/normalize过后的坐标
   2. eval在eval.util里处理，**有一个\_scale\_box\_to\_absolute来恢复坐标，在core里有一个box\_list\_ops有处理函数。**
   3. **在visualization\_utils里有个**draw\_side\_by\_side\_evaluation\_image  
      分析发现输出没有id，gt有id
2. model\_lib\_v2里有eager\_eval\_loop，
   1. 调用了compute\_eval\_dict
      1. 里面调用了detection\_model.postprocess，出来的prediction\_dict有id
   2. 调用了prepare\_eval\_dict
      1. 输入的detections里有reid
      2. 在这调用了result\_dict\_for\_batched\_example，没有对id进行处理
   3. 调用了draw\_side\_by\_side\_evaluation\_image
      1. 调用了draw\_bounding\_boxes\_on\_image\_tensors，没有传入id信息，可以对id进行处理
3. 我的改动
   1. result\_dict\_for\_batched\_example加上了对于id的处理
   2. 把ids传入了visualize函数里，predict和gt都显示正常
   3. draw\_side\_by\_side\_evaluation\_image可以调整一张图画多少个box，一共画多少张图
   4. 只有一个bbox被画出来的原因是后处理部分\_center\_params.max\_box\_predictions是1，导致就predict出来了一个bbox，后来发现改一下config就成了
   5. 在eager\_eval\_loop里保存为图片

**一些解决ID问题的可能方案**

1. **改总loss的计算方式**

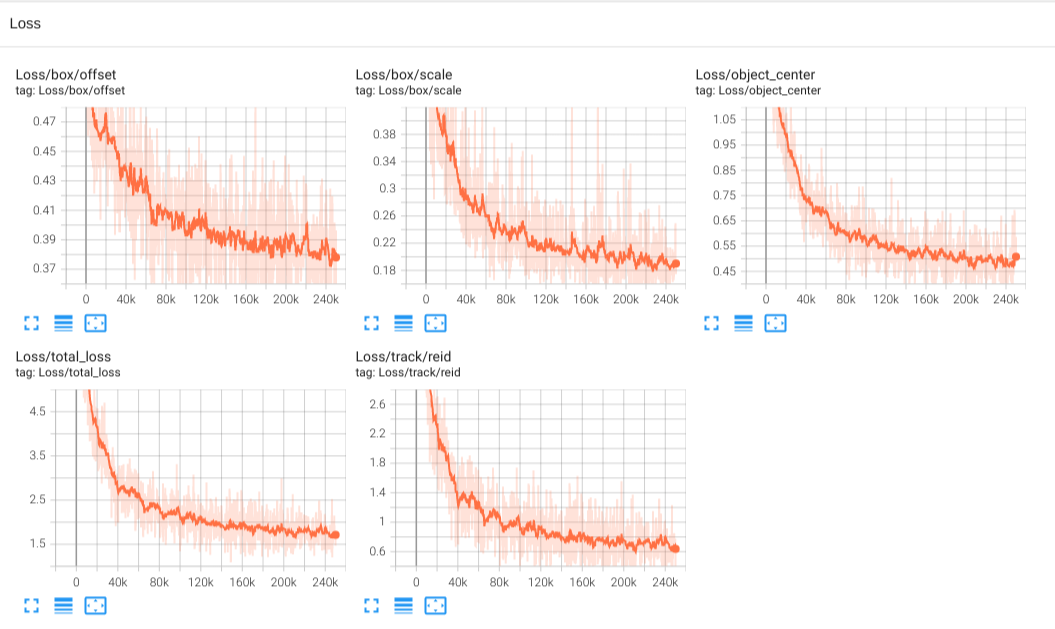
self.s\_det = tf.Variable(-1.85 \* tf.ones(1))

self.s\_id = tf.Variable(-1.05 \* tf.ones(1))

**没有啥用**

1. **换dataset**
2. **换一个loss计算方式**

**成功了**



分析reID相关的后处理

1. compute\_track\_embedding\_loss 记录了计算loss的过程
   1. 从 object\_reid\_predictions [8,96,96,128]
   2. 先通过cn\_assigner.get\_batch\_predictions\_from\_indices变成embedding[8,100,128]
   3. 再通过self.track\_reid\_classification\_net变成reid\_classification[8,100,2300]
   4. 之后直接与targets算loss  
      数据：ids[8,100],boxes[8,100,4],转化后的targets[8,100,2300]
2. 目前的后处理是
   1. 从 object\_reid\_predictions [8,96,96,128]
   2. 通过predicted\_embeddings\_at\_object\_centers[8,10,128]
   3. 在经过self.track\_reid\_classification\_net[8,10,2300]
   4. 在经过一个argmax[8,10,1]
3. 我的做法：只需要把track\_reid\_classification\_net加进来，再argmax一下

网络出来的是pred是logits，然后targets是onehot形式的label

忽然发现print出来的argmax后的不全是34，成功了

1. 先做量化意识训练，在export那步骤的时候，做切割，就是传heatmap，接收heatmap
2. 量化意识训练

<http://192.168.50.82/ai/demos/type3/simple_scenarios/tracking/multiobjtracking/-/blob/main/configs/centernet_mobilenetv2_mpii_coco_single_person_export_pb_384_upsample_new_incl_obd_remove_flipaug_qat_test.config>

1. 量化意识训练 byobu
   1. Graph\_rewriter
   2. <https://blog.csdn.net/lifengcai_/article/details/81706171>
   3. 只需要改一下config文件即可，相当于在设定步数之后，复刻一个新的int8的模型参与训练，可以提高量化后的模型准确率
2. 改bbox每个人1个
   1. Eval部分num\_visualizations10和max\_num\_boxes\_to\_visualize20
   2. Train部分max\_box\_predictions1和max\_number\_of\_boxes1
   3. 在train和eval代码中先不改了，不好操作
   4. 之后可以在后处理部分解决，由于在输出heatmap之前加上了一个maxpooling等操作，可以使得heatmap峰值比较离散

<http://192.168.50.82/ai/research/base_repo_restructure/google_object_detection_api/tf2/-/blob/centernet_imx500_modification_midout_for_multi_person/research/object_detection/meta_architectures/center_net_postm_meta_arch.py>

1. Tflite切分
   1. 首先把transposeConv变成upsampling，只保留到24\*24  
      /workspace/google\_api/tf2/research/object\_detection/models/center\_net\_mobilenet\_v2\_feature\_extractor.py, ok（又改回96了）
   2. 后续的预测头维度大概都得改

\_construct\_prediction\_heads，是自动的. ok

* 1. 把两个卷积+upsampling加到reID的分支（又给注释掉了）

\_construct\_prediction\_heads, make\_prediction\_net, ok

* 1. 将reID enbedding size 变为36（又改成4了）, ok
  2. 抄一个pooling heatmap的函数，有一些定义的变量

top\_k\_feature\_map\_simplification, ok

* 1. 写一个新的postprocess函数, ok
  2. Id部分的后处理恢复到原来，ok
  3. Core/standard\_fileds需要更新, ok
  4. 发现gt的size也需要改，eric到时候应该能支援一下

/workspace/google\_api/tf2/research/object\_detection/models/center\_net\_mobilenet\_v2\_feature\_extractor.py, （又改回来了）ok

* 1. 调整2pb的转换接口, ok
  2. 训练
  3. 导出tflite

3.258 G ops, equivalently 1.629 G MACs

1. QAT和checkpoint

# fine\_tune\_checkpoint\_version: V2

# fine\_tune\_checkpoint\_type: "detection"

# fine\_tune\_checkpoint: "/workspace/Project/CTQ/pose\_detection\_centernet/train\_output/train\_centernet\_pose\_detection\_single\_mp\_debug\_3600\_steps\_384\_upsample\_24\_output\_new\_flip\_incl\_obd\_remove\_flipaug/ckpt-301"

# from\_detection\_checkpoint: true

#graph\_rewriter {

# quantization {

# delay: 18000

# weight\_bits: 8

# activation\_bits: 8

# }

#}

Steps

问题：

* 1. Delay步数
  2. Learning rate
  3. Full还是detection part

1. BBOX从heatmap提取可以参考下

<http://192.168.50.82/ai/toolkits/imx500_postdebug_python/post_processing_assistant/-/blob/main/ref/poseDetection/TFLite_verification_centenet_m_pose_detection_multiperson_quantization_debug_topk_wbox_video.py>

* 1. 处理输入，获取输出

发现heatmap可视化有问题，全是0和-50，debug所做工作

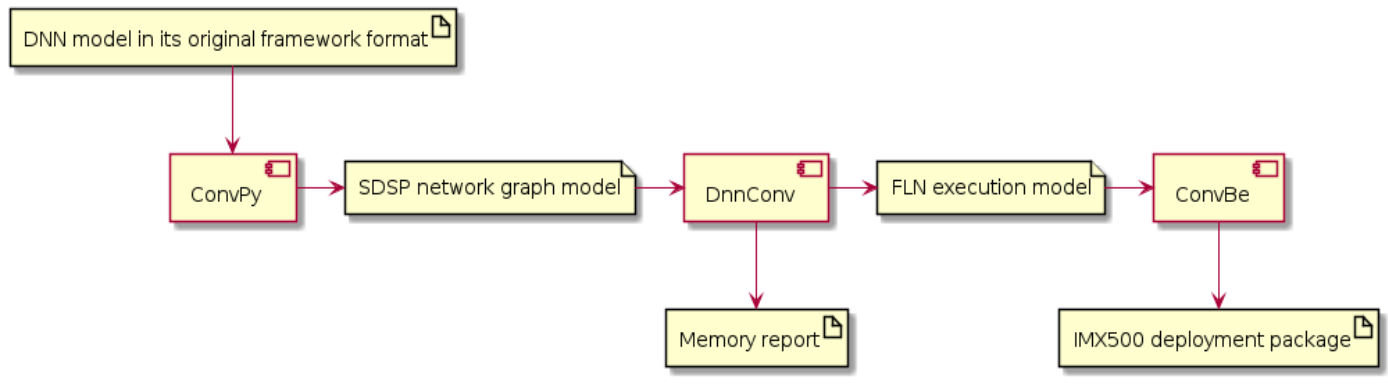
1. SAVE\_HEATMAP调成1
2. 查maxpooling相关，没有问题，希望变成-50和最高点的原值
3. HEATMAP\_SIMPLIFIED\_DEBUG:打开
4. NORMALIZATION\_INPUT应该关闭
5. 调了一下heatmap窗口大小和保留两位小数
6. 查maxpooling前的heatmap，可以从tflite中通过location找到，发现不同图片的heatmap都长得一样
7. 把pb2tf.py里面图片的正则化给去掉之后好了
   1. get\_object\_center\_coordinates\_scores负责从heatmap中找到topk的indices
   2. get\_boxes\_coordinates负责给出topk的bbox坐标
      1. 把heatmap调试结果保存
      2. 把heatmap调试接口关闭
      3. 恢复坐标，写draw函数
   3. 自己写一个decode id的函数
      1. 余弦相似度都是0.9几，不太好用
      2. <https://www.jb51.net/article/164673.htm> 各种各样的距离
   4. 把信息画在原图上，打印
8. Id检测的不准确
   1. 首先可以把center上下的点的embedding都算进去，符合人的特点
   2. 将bbox wh算进去，当成距离的一部分
   3. 目前改了size=8，调了阈值，会更新embedding
9. SDSP转换
   1. <http://192.168.50.82/ai/demos/type3/simple_scenarios/tracking/multiobjtracking> SDSP converter那段
   2. 还有这个 flatbuffer 关于tflite的操作python 脚本，你结合你的模型要稍微改下  
      <http://192.168.50.82/ai/demos/type3/simple_scenarios/tracking/multiobjtracking/-/blob/main/src/flatbuffer_process/pose_detection/flatbuffer_postprocess.py>
   3. Eric说是SDSP转换会遇到一些bug，这时候需要使用flatbuffer对tflite进行一些修改
   4. Docker  
      127.0.0.1:5000/type3\_nuttx\_sdsp\_docker\_slim:0.01

*docker run -it --name mwt\_sdsp\_converter --ipc=host -e DISPLAY=unix$DISPLAY -v ~/Desktop:/mwt\_workspace -p 9876:22 --device=/dev/video0:/dev/video0 -v /tmp/.X11-unix:/tmp/.X11-unix --privileged=true --gpus all 127.0.0.1:5000/type3\_nuttx\_sdsp\_docker\_slim:0.01 /bin/bash*

unset http\_proxy; unset https\_proxy;之后就可以正常apt了

这个docker里已经有convpy了，好像没有DnnConv和ConvBE，手动安转一下

* 1. SDSP官方文档学习
     1. SDSP的目的是要把tflite类型的DNN网络转换成二进制文件
     2. ConvPy转换成另外的的graph格式，DnnConv转换成节点model（同时可以进行一下模型大小的评估），ConvBE转换成二进制文件



* + 1. Convpy -i-o输入输出路径，-n制定一个json作为网络结构的额外说明，应该就是用flatbuffer生成的，也可能不是
    2. DnnConv -n-o 输入输出路径 –mem\_size输出内存报告。这一步会有网络layer的替换，目的是运算效率更高。同时需要convpy输出的网络和一个xml的网络定义文件。Source ~/.profile安装
    3. ConvBE -f-o输入输出路径
    4. 后面的步骤需要切换到SDSP-Packager的位置进行，需要用到eric的密码ABCDabcd12

目前有个unsigned的bug，可以通过改 /workspace/MultiObjTracking/SDSP/sdsp/dnnconv\_output/convpy\_output/dnnParams.xml 来解决

1. Flatbuffer代码
   1. 首先是改名字：输入、输出、squeeze(unpack)、deprecated\_builtin\_code；  
      其次是改了sigmoid的值；最后是删掉一些行（unpack的一些参数）；  
      这里需要配好环境后，看一下json，再去理解
   2. Flatbuffer：<https://blog.csdn.net/ZM_Yang/article/details/108408062>   
      flatbuffer是一种序列化文件存储的格式，.tflite的存储实际上就是使用的flatbuffer格式。Netron就是进行的解析然后可视化。利用flatc把tflite解析成json，然后直接替换
   3. Schema.fbs
      1. Wget源有问题，直接手动粘贴的  
         <https://raw.githubusercontent.com/tensorflow/tensorflow/r1.13/tensorflow/lite/schema/schema.fbs>
      2. 介绍文档  
         <https://blog.csdn.net/Pengcode/article/details/121776674>
      3. 这里网上找的有点老，出现了bug；后来eric给了一个更新的，解决了尺寸不匹配的问题
   4. 输出的json
      1. 实际上并不符合json的语法，只是结构类似
      2. <https://blog.csdn.net/Pengcode/article/details/121776674>
      3. <https://zhuanlan.zhihu.com/p/391770192?utm_id=0>
      4. Eric又给了两个文档  
         <https://blog.csdn.net/SilentOB/article/details/86479442>   
         <https://blog.csdn.net/u011279649/article/details/83186550>
      5. 结构
         1. Subgraphs
            1. Tensors：buffer作为序号。理论上要是删掉，所有序号都得变
            2. Inputs：tensor序号
            3. Outputs：tensor序号
            4. Operator就是一堆算子，由tensor组成，理论上序号应该是opcode\_index，但是删完后没有改，所以应该是顺序匹配
2. 效果进一步提升
   1. 动态更新字典的tensor，有一定的权重（效果不行，取消）
   2. 每张图的前四个tensor做一个归一化（好像是负效果..，取消）
   3. 同一个id只显示一个框
   4. Id map做一个卷积smooth