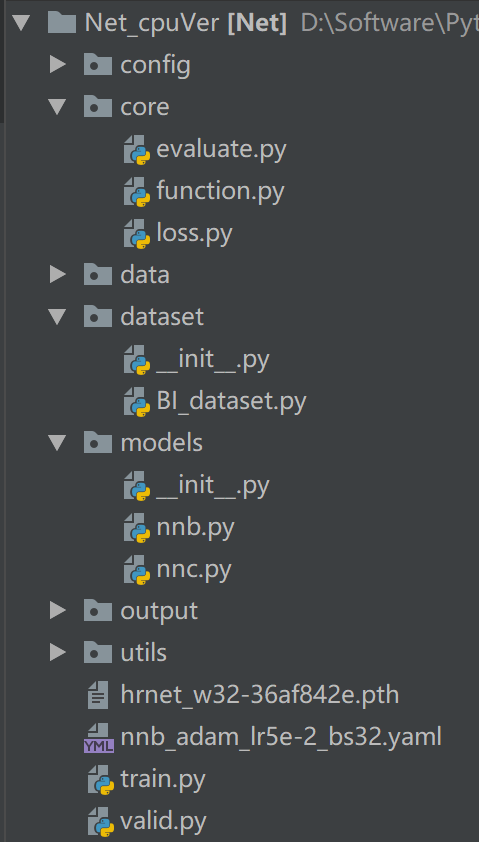
# FaceXRay复现报告

1. 代码结构



1. config

配置相关，加载.yaml文件，且MODEL.EXTRA模块允许新建子类，在.yaml文件中被申明。

2. core

(1) evaluate.py

使用语义分割计算的方式计算准确率指标；

使用了(TP + TN) / (TP + TN + FP + TN)的PA准确率和TP / (TP + FP + FN)的MioU准确率，最终在训练时只选取了PA准确率。

(2) function.py

训练、验证步骤封装。

(3) loss.py

Loss函数及过程封装；

生成facexray的网络nnb使用BCELoss，判断是否伪造的网络nnc使用CrossEntropyLoss。

3. data

存放训练、验证数据集(BI\_dataset)，input是混合人脸图像，label是黑白FaceXRay图像。

4. dataset

读取BI\_dataset的类；

数据目录需要在./data/train下创建：

文件夹data：里面放所有的混合人脸图像，“1.jpg” …；

文件夹label：里面放所有的FaceXRay图像(标签)，“2.png” …(与图像的名字一样)；

train.txt：包含图像的名字，每一行是一个名字。

数据读取采用PIL库，对混合人脸读RGB图像，对FaceXRay读灰度图像，为了与网络output保持相同shape，即[1，256，256]

5. models

(1) nnb.py

生成facexray的网络。

(2) nnc.py

最终进行是否伪造判断的网络。

6. output

存放最后保存的.pth和tensorboard的文件。

7. utils

其他工具。

8. hrnet\_w32.....pth

原hrnet基于imagenet训练得到的模型参数，该实验基于该参数进行训练，前50000次迭代训练时需要固定这部分参数。

9. nnb\_adam......yaml

配置文件，包括hrnet网络模型形状信息、训练、验证信息(数据路径等)、学习率等超参数。

10. train.py

训练代码。

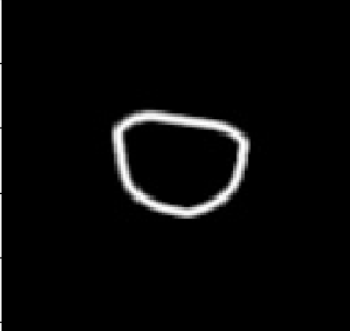
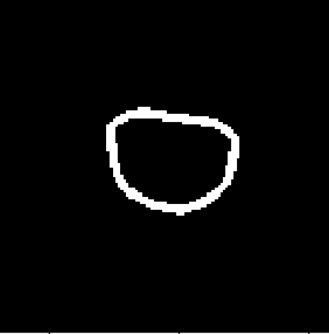
11. valid.py

验证代码，现在只是输出网络输出。

1. 问题

1. 网络nnb的loss计算过程

由于label里的数据应该非0即1，而直接读取的label数据是0-1的float数据，所以设置了阈值0.5，大于0.5的数值为1，其余为0。但是这样导致模糊边缘消失，label自身就变成了带有锯齿状的图像。

图a：直接读取的label 图b：经过阈值处理后的label

2. 关于网络nnb输出的facexray output边缘具有明显的锯齿状

问题1对此有一定的影响。

3. 关于nnc网络

论文里对它的构建是：全局平均池化层 + 全连接层 + softmax，但是根据我已有的浅显的知识，全局平均池化层是在最后输出[batchsize, classnum, w, h]的情况下，对一个channel的w\*h全局取一个平均数，最后构成[batchsize, classnum, 1, 1]，再接入一个softmax，每一个输出结果代表着这classnum个类别的概率，其中全局平均池化层是取代全连接层的作用，且有不需要参数，避免在该层产生过拟合的优点。

因此我觉得它一是感觉层作用有点冗余，二是nnb最后的output的channel是1，但是是否伪造应该有2类，也不是很正确的全局平均池化层的使用？

同时，我感觉这个网络的作用似乎有一点点多余？因为在输出facexray时，里面有无轮廓就相当于默认是否伪造。

综上所述，虽然我构建了网络和loss但是我没有把他加入训练里，并且训练，论文取=100，更多关注nnb网络的学习，所以我觉得这样训练问题应该不大。