

## 任务 A

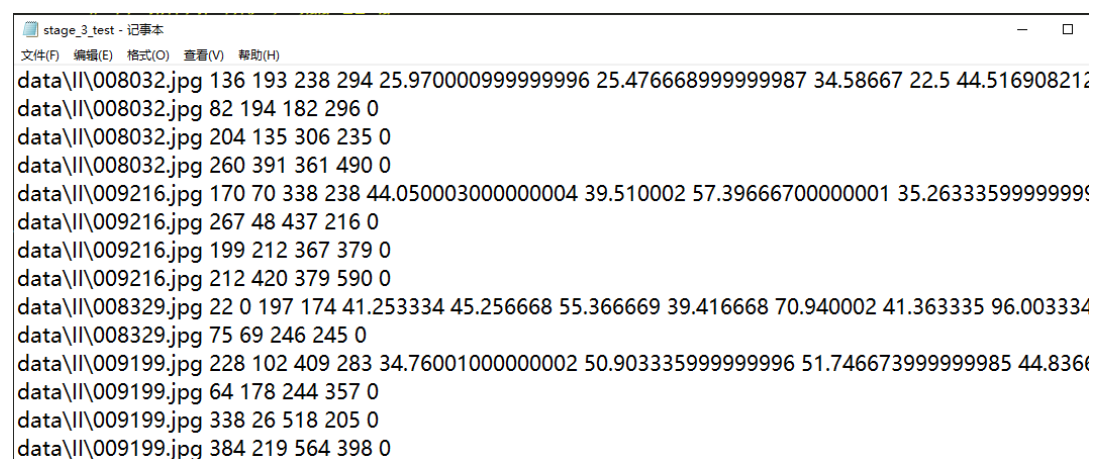
我们组用了 stage3 默认分支网络。由于使用的交叉熵 LOSS 已经包含了 log softmax, 所以在分类网络的末端我们组没有添加 softmax。

## 任务 B&C

我们组重新生成了数据 stage\_3\_train.txt 和 stage\_3\_test.txt。

→ 每个正样本按照 1:3 的比例生成 3 张负样本的图片

→ 如果正样本中人脸过大, 导致无法按照  $iou < 0.3$  生成足够 3 份负样本的话, 则减少生成负样本的数量或者放弃生成负样本



```
stage_3_test - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
data\II\008032.jpg 136 193 238 294 25.970000999999996 25.476668999999998 34.58667 22.5 44.516908212
data\II\008032.jpg 82 194 182 296 0
data\II\008032.jpg 204 135 306 235 0
data\II\008032.jpg 260 391 361 490 0
data\II\009216.jpg 170 70 338 238 44.050003000000004 39.510002 57.396667000000001 35.263335999999996
data\II\009216.jpg 267 48 437 216 0
data\II\009216.jpg 199 212 367 379 0
data\II\009216.jpg 212 420 379 590 0
data\II\008329.jpg 22 0 197 174 41.253334 45.256668 55.366669 39.416668 70.940002 41.363335 96.003334
data\II\008329.jpg 75 69 246 245 0
data\II\009199.jpg 228 102 409 283 34.760010000000002 50.903335999999996 51.746673999999985 44.83667
data\II\009199.jpg 64 178 244 357 0
data\II\009199.jpg 338 26 518 205 0
data\II\009199.jpg 384 219 564 398 0
```

## 模型参数

**optimizer** : ADAM    Learning rate: 0.01    beta:0.9, 0.99

**Batch\_Size** : 64

**criterion1** : Cross Entropy    --> CLS loss

**criterion2** : Smooth L1    --> Pts loss

**LOSS 权重** :  $\text{loss} = 2 * \text{loss\_cls} + 1 * \text{loss\_pts}$

**BN** : 分别在 conv1\_1,conv2\_1,conv2\_2,conv3\_1,conv3\_2 之后加了 BN 层

**Dropout** : 在 flatten 后面加了 dropout 层, 随机激活 50%的神经元

## 任务 D

**a** 在一开始我们没有加入 weighted cross entropy 的参数。通过训练观察到，由于负样本数比较多，在训练初期，模型对负样本(0)的识别率较高，而对正样本(1)的识别率比较低，但是通过训练之后，模型对正负样本都能进行很好的识别了。

### 训练初期

```
Train Epoch:3[5120/7250(70%)]    loss:1.911907
Valid: loss: 6.595177
Train_CLS_accuracy:90.8000% (6583 / 7250)
Train_CLS_one_accuracy:80.8560% (1757 / 2173)
Train_CLS_zero_accuracy:95.0561% (4826 / 5077)
Valid_CLS_accuracy:77.3087% (1465 / 1895)
Valid_CLS_one_accuracy:32.7206% (178 / 544)
Valid_CLS_zero_accuracy:95.2628% (1287 / 1351)
=====
```

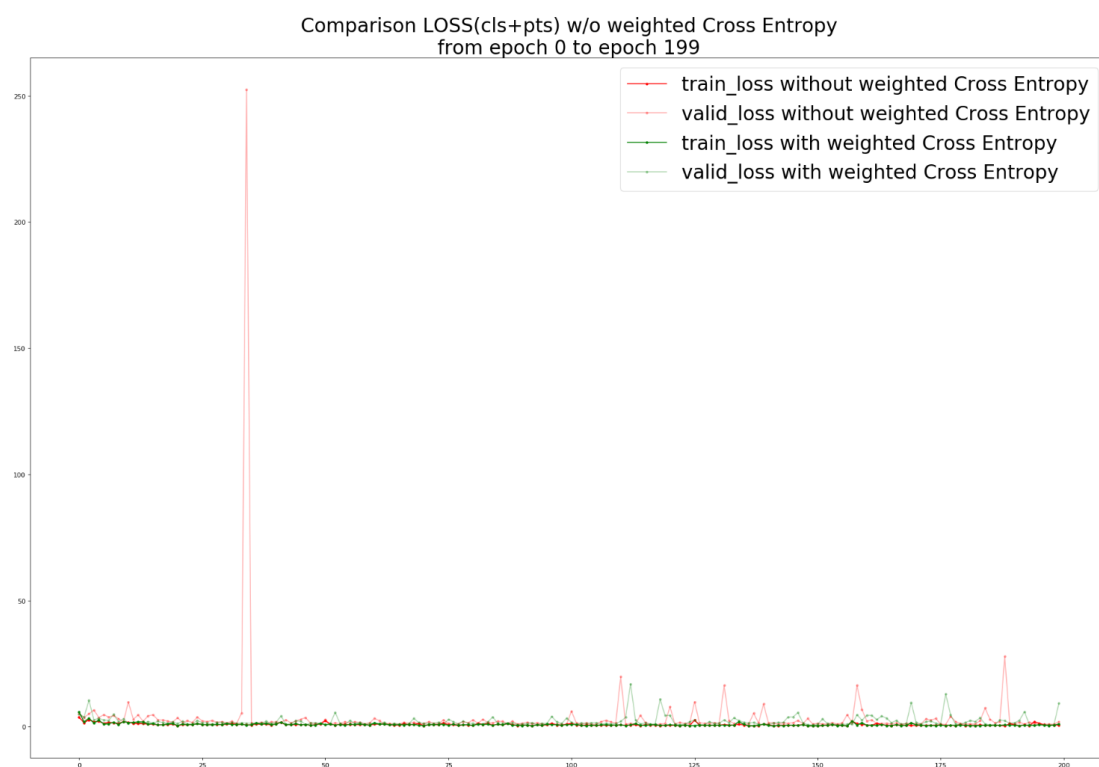
### 训练之后

```
Train Epoch:99[0/7250(0%)]    loss:0.490468
Train Epoch:99[5120/7250(70%)]    loss:0.999248
Valid: loss: 0.801685
Train_CLS_accuracy:99.6414% (7224 / 7250)
Train_CLS_one_accuracy:99.3097% (2158 / 2173)
Train_CLS_zero_accuracy:99.7833% (5066 / 5077)
Valid_CLS_accuracy:98.9446% (1875 / 1895)
Valid_CLS_one_accuracy:97.4265% (530 / 544)
Valid_CLS_zero_accuracy:99.5559% (1345 / 1351)
```

然而通过观察 valid loss 的值，我们发现，对于识别正样本"1" 的 valid loss 会每隔一定数量的 epoch 时突然升高"跳动"，我们认为这可能是因为没设置权重的原因，导致样本数量较少的正样本没有被模型很好的识别，于是我们在模型里按照正负样本的比例加入了权重，然后重新进行了训练。

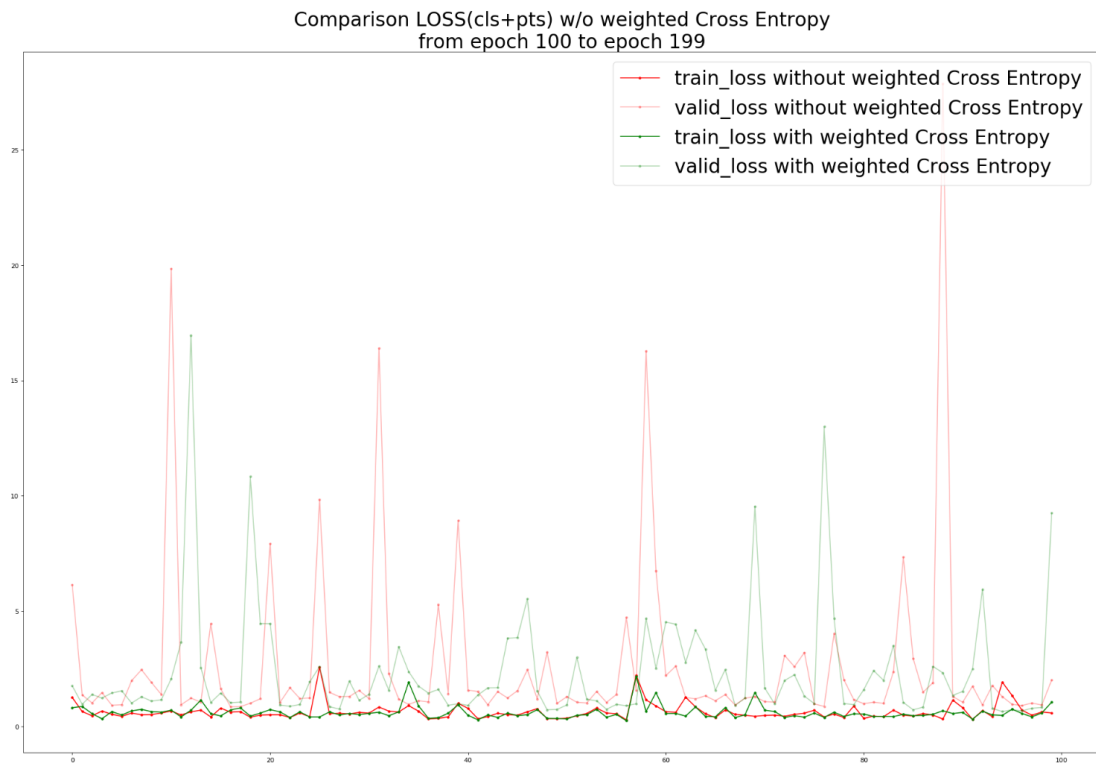
```
# parameter of weighted cross entropy
weights = [5077.0/7250.0, 2173.0/7250.0]
class_weights = torch.FloatTensor(weights).cuda()
criterion1 = nn.CrossEntropyLoss(weight=class_weights) # binary classification num(0), num(1)
```

下面是我们的 2 次训练



→ 没有加入交叉熵权重的模型，在进行验证模型时，valid\_loss 会出现强烈的波动

→ 100 个 epoch 训练之后，此模型的 valid\_loss 还是会有明显波动



→ 而加了权重之后, `valid_loss` 的值相对于之前稳定了很多

→ 这代表正样本的识别率提高了

→ **正负样本比例 1:3 是合适的训练比例, 我们的目的是为了试验 `valid_loss` 稳定性和交叉熵权重的相关性**

**b** 对于 loss 权重我们组侧重于分类 loss 的训练

$$\text{loss} = 2 * \text{loss\_cls} + 1 * \text{loss\_pts}$$

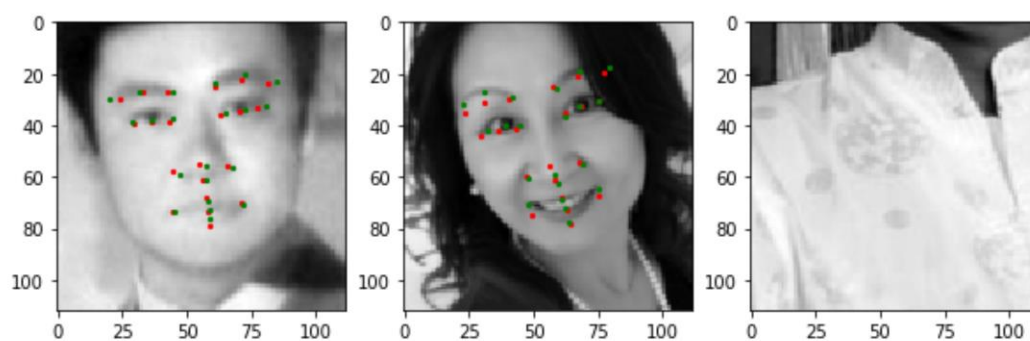
## 任务 E

我们组分别对总体样本, TP 样本(gt=1,pred=1) 和 TN 样本(gt=0,pred=0) 进行了精确度的计算

## 模型 Test 结果

用了训练了 100 轮的模型来进行 test (任务 D 中训练之后的模型)

```
==> Testing  
0.8016849358876547  
Valid_CLS_accuracy:98.9446%  
Valid_CLS_one_accuracy:97.4265%  
Valid_CLS_zero_accuracy:99.5559%
```



总体 loss 为 0.8, 分类和关键点检测都基本达到了我们的要求