**SSSBI数据集说明**

**一、数据集介绍**

SSSBI数据集，是2024年SRP项目《基于拼接的人脸伪造检测数据集合成研究》的成果，是在孙璐老师的带领下，由vggface2\_crop\_arcfacealign\_224数据集，结合SBI和SimSwap两种方法生成的合成数据集，通过仿真拼接伪影和拼接区域内外的统计伪影（SBI）和引入生成器带来的特定噪声模式（SimSwap），用于Deepfake检测器训练，期望提高检测器的泛化能力。

**1.1 vggface2\_crop\_arcfacealign\_224数据集**

vggface2\_crop\_arcfacealign\_224数据集是一个10.8G大小的图片数据集，包含8631个子文件夹，每个文件夹都有几十张224\*224大小的人脸图片。

完整的数据集下载链接如下：

[[Baidu Driver]](<https://pan.baidu.com/s/1OiwLJHVBSYB4AY2vEcfN0A>)

[Password: lrod]

**1.2 SSSBI数据集**

SSSBI数据集，全称SimSwap evolved SelfBlendedImages（韦业取的名w），如上所述，是一个24.8G的合成数据集，同样包含8631个子文件夹，每个子文件夹里是4个二级子文件夹，每个二级文件夹中是672\*224大小的若干图片，由人脸图片和2个掩膜组成，数据集结构如下：

SSSBI

├─ n000002

│ ├─ 0

│ │ ├─ 0002\_01.jpg

│ │ ├─ 0003\_01.jpg

│ ├─ 1

│ ├─ 2

│ ├─ 3

├─ n000003

├─ ...

完整的数据集下载链接如下：

链接: https://pan.baidu.com/s/1qhz5DWNWlndZ08ENH2SJ8w?pwd=1145

提取码: 1145

**1.2.1数据集目的**

Deepfake伪造过程通常包括使用生成模型（GAN、扩散模型等）生成伪造内容（脸部），以及使用计算机图像处理技术将生成图像融合到原图像中，前者可能引入生成器带来的特定噪声模式，而后者可能带来拼接伪影和拼接区域内外的统计伪影，于是针对这些问题，我们对vggface数据集中的每张图片，用SBI仿真拼接伪影和统计伪影，用SimSwap引入生成器伪影，生成4种图片和2种掩膜，希望检测器能关注到两种伪影，提高泛化能力。

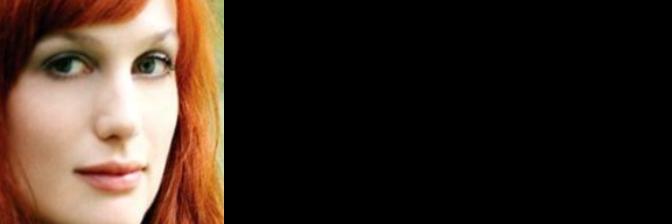
**1.2.2掩膜**

边界掩膜：使用拼接方法（SBI）时，标注了拼接边界，值为[0,1]

操纵掩膜：使用生成方法（SimSwap）时，标注了GAN生成部分，值为{0,1}

**1.2.3生成图片**

IS（0）：真实图片，但是经过随机的图像增强操作（空处理、随机JPEG压缩、颜色扰动、对比度调节、高斯滤波）。他的边界掩膜和操纵掩膜全为0



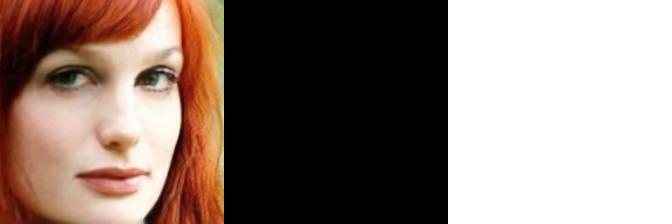
ISB（1）：SBI生成的虚假图片，他的边界掩膜在拼接边界有值[0,1]，操纵掩膜全为0



ISBR（2）：IR经过SBI生成的虚假图片，他的边界掩膜在拼接边界有值[0,1]，操纵掩膜在脸部分有值{0,1}



IR（3）：SimSwap生成的虚假图片，他的边界掩膜全为0，操纵掩膜全为1



**二、运行环境**

合成程序的运行环境基于python3.9，需要numpy,pytorch,sklearn,dlib,albumentations,imutils,pandas,matplotlib,imgaug等库，具体环境配置过程参照环境配置文件夹里的说明，dlib安装起来可能有点麻烦，给了一个轮子文件，未开启GPU加速，但用于数据集生成也够了。

无论是否支持GPU都可运行合成程序，请根据自己的情况安装对应的pytorch版本，合成前10个文件夹，使用GPU大约要71.08秒，用CPU大约要587.39秒

**三、合成程序**

**3.1 运行程序**

整个工程包括SBI和SimSwap的工程文件和Generate\_SSSBI.py，合成请运行Generate\_SSSBI.py

SBI和SimSwap均使用了2024/8/3之前的最新版本，里面有些修改，比如加了arcface模型的文件，cpu运行的修改，还有一些库版本更新带来的改动。

参数设置如下：



Image\_size：控制图片大小

Input\_path：是vggface数据集的路径，也可以换其他的图片，比如ff++的，但是要注意数据集结构得和vggface一样，或者改一下程序里读取图片的部分。

Output\_path：合成文件保存的路径

Already\_folder\_num：已处理子文件的总数，本来是因为vggface的子文件太多，八千多个，万一中间出问题运行中断了，可以从断的地方继续生成

Folder\_num：待处理的子文件夹总数，这是不包含已处理子文件总数的，比如alreay\_folder\_num设为10，folder\_num设为100，那么程序会合成10-110这100个子文件夹的图片，如果觉得这个逻辑反人类，也可以在154行155行那边修改，应该换一下顺序就好了。

**3.2程序逻辑**

合成程序首先是导入了SBI、SimSwap还有本身用到的一些库，并导入了涉及到的SBI和SimSwap的模型和函数

然后遍历vggface数据集的每张图片，分别生成4种图片并保存：

1.IS的生成：对图片随机选择空处理、随机JPEG压缩、颜色扰动、对比度调节、高斯滤波中的一种进行处理

2.ISB与ISBR的生成：原本应该生成ISB时取原始图片的landmark与bbox，生成ISBR时取IR的landmark与bbox，但简单一点就都取原始图片的landmark与bbox了；重写了SBI的self\_blending函数，原来的函数接收原始图片，在函数内部区分source图片与target图片，改为接收参数时即区分source图片和target图片

3.ISB与ISBR的边界掩膜：因为SBI程序中的mask\_f是乘以随机拼接率的，为了成功计算出边界，改写了self\_blending函数和blend文件中dynamic\_blend函数，返回选取的拼接率，然后用mask\_f除以选取的拼接率，得到内部全白的mask，使用face X-ray的公式mask⊙（1-mask）⊙4计算得到边界。

4.ISBR的操纵掩膜：将mask\_f非0的地方置1得到。

5.数据集平衡：因为程序将检测landmark和生成SBI的过程合并，图片不一定能检测到landmark，因此会有少量原始图片得不到ISB或ISBR，这导致ISB和ISBR数量小于等于IS和IR的数量，为了让四种图片数量一致，对于无法生成ISB或ISBR的图片，也不保存他的IS和IR了。

6.SBI的两次随机裁剪：因为SBI有两次随机裁剪，因此生成过程中没用到SBI的IS和IR会小一点，要么都去掉随机裁剪，要么都用随机裁剪，想着随机裁剪也是数据增强的一种就没去，但也不知道如何固定一张图片的随机裁剪，干脆IS和IR都进行两次随机裁剪了。

7.图片格式：因为SBI和SimSwap处理图片用到了好几种不同的库，他们的图片格式并不一致，程序中转换图片格式的逻辑可能有些繁琐，存在优化空间。

如果程序和SSSBI数据集的下载有啥问题可以加qq联系我，3520713257（孟文瑜）-2025.2.8

上传至github，对于SimSwap和SBI我均只上传了修改的部分（我记得住的），希望不会违反许可证（第一次用github上传东西，关于许可这方面还完全搞不清，就用了目前感觉没问题的方式上传，等以后了解加深了我再优化一下）。不过感觉这样很难运行的起来，所以把我这边的完整工程打包也放度盘上了，链接如下：

链接: https://pan.baidu.com/s/1upF7XRk6Oduj1johlCmofw?pwd=1145

提取码: 1145

-2025.3.24