无约束人脸图像识别系统

2023.1.13

一、项目内容

本项目为无约束人脸图像识别系统。需要实现的任务为给定两张人脸图像，判断是否为同一人。训练集为 5354 个不同个体，每个个体均有一张或以上的人脸图像，共 12448 张人脸图像。而测试集则为 600 对人脸组合，用于判断是否为同一人。

二、实现思路、方案、对比

项目难点一方面在于数据量大，另一方面则在于对于无约束人脸图像识别系统，目标人脸在图像中央，而背景较复杂，人脸未经对齐。因此项目主要内容由两部分组成，一个是实现人脸检测与对齐（Face Detection & Face Alignment），对人脸实现检测后进行裁剪与转正，第二部分则是基于检测并裁剪后的图像对人脸进行特征提取与表示，从而区分不同的人脸。

在人脸检测与对齐方面，主要思路参考了课程 PPT 上所给的利用 Dlib 库对人脸进行检测得到检测框与 68 个关键点，得到检测框之后裁剪出人脸，并以检测得到的两眼关键点的中点为中心，对裁剪后的图像进行旋转从而转正，旋转角度也是通过两眼连线与水平竖直方向所成角度计算而成。而整个实现过程中需注意：1.检测出的人脸可能不止一个，需要排除边缘的人脸，在本项目中，本人采取将 x<0，y<0 的人脸去除并取之后的第一个人脸框（从而得到位于中间的人脸）。

而基于检测并裁剪后的图像对人脸进行特征提取与表示，区分人脸异同则有很多实现方法，主要思路有以下两种。

一是基于数据集中所含有的人脸进行人脸识别，之后对测试集中每对人脸分别进行识别，再对比标签，此思路有明显的弊端，即该任务为开集人脸比对，因此会出现数据集中未出现的人脸，而此情况下进行识别得到的标签并不能表征此人。更何况，仅是进行如此大数据量的人脸识别，准确率与置信度本身已是一个很大问题。当然，此思路的优点是容易实现，可以传统方法如 LBP，SVM 等方法，在训练集上直接找到合适的参数从而进行识别，再对测试集的人脸进行比对。

二是跳出人脸识别的观点，直接从比对的角度出发，即将任务建模为度量学习，对于两张图片，我们寻找一个距离（在本项目中实际表现为一个特征），来使得同类的特征更加相似，异类的特征更加相互远离。从而寻找到一个对人脸图像进行描述的模型。之后再基于这个模型处理后的特征进行分类器的训练。参考 Facenet 论文，此问题实际上可以通过三元组作为损失函数来训练，从而获得一个 128 维向量作为每一张图片的特征，之后再对特征进行分类器。这正是本人在该项目中实现后效果最好的方法。

三、实现结果与代码附件解释

1. **人脸检测与对齐。**

   此部分的内容较为简单，便不赘述，可参考第二点中描述。而此部分的代码实现则在 utils.py 文件中，实现为 face_alignment 函数，输入一个图片（项目中为[0,255]的 250*250*3 的图像），经检测与裁剪之后输出相同大小的 RGB 图像或灰度图像。值得一提的是，该过程处理较为费时，因此为了后续训练方便，在 utils.py 的注释中本人还利用该函数对数据集进行了完整的处理，得到 gray_set 和 rgb_set 两个文件夹，包含所有数据集的处理结果。（因图片过多未附在附件中）

2. **人脸特征提取与表示。**

A. 尝试传统方法。

a.利用 opencv 中自带的 LBPH 人脸识别函数，本人尝试了传统的 LBP 方法，将数据集切分为 2000，4000，6000，8000，10000，12000 张图像，分别测试 LBP 算法，发现 LBPH 随着数据集增大时效果急剧下降，具体结果见表 1，同时附件中 LBP.ipynb 含有完整处理过程。另外，分别用得到的 LBP 模型，在测试集的 6000 对人脸中进行预测，对比每对中两张图是否属于同一标签，得到结果于 LBPH_result 文件夹中。

Total images	Error	Acc	Not_confident(confidence>80)
2000	0	1.0	0
4000	1870	0.5325	2
6000	4000	0.3333	2
8000	5993	0.2509	2
10000	8001	0.1999	3
12000	10000	0.1667	6
12448	11999	0.03607	23

表 1 LBP **在训练集上表现**

b. 进行项目中，还使用了 SVM 进行直接训练，发现会因为数据量过大而导致训练十分缓慢的问题，利用 PCA 进行降维可以解决这个问题，但最终训练效果不理想，便没有保留在测试集上的结果。可以通过附件中的 SVM 相关文件查看代码。

B. 基于 Facenet 的神经网络处理方法

本人认为该方法主要实现的难点在于数据集的生成。因此在 Dataset.py 中保留了两个以 datasets.ImageFolder 为基类的数据集类 TriletFaceDataset，和 Pairs_Dataset。前者的编写参考了 github 上的项目，于参考文献中提及。

而该方法的过程则分为两步，完整的训练过程均展示在 facenet.ipynb 中，结果则在 result_facenet.txt 中。

第一步是以 facenet 文章中提到的模型训练了一个 net_front，net_front 将对齐并裁剪完的人脸处理成 128 维向量，作为特征表示。而训练时损失函数采用三元组损失函数，数据则是随机产生了 20000 个三元组作为训练数据，在 margin=1.0, p=2 的情况下训练了 5 个 epoch，生成了 margin1_epoch5.pkl 文件，再在此基础上在 margin=0.5, p=2 情况下又训练 5 个 epoch，生成了 margin0.5_epoch5.pkl 文件，训练器为 Adam。

第二步则是对基于 net_front 产生的特征训练了一个 net_output，其将两个图片的特征作为输入，输出一个二维向量，选取最大值在维度 0 处或在 1 处作为两张图片是否为同一人的表示。数据是基于原数据集产生的 pairs 数据，训练集中有 10000 对相同人的样本，和 10000 对不同人的样本，值得一提的是此处的数据产生要考虑分布问题，因随机产生的数据在训练时很容易便达到 acc=1.0，因随机选出的数据基本都是两个不同人的照片，同时测试集中也随机选了 500 对相同人的样本和 500 对不同人的样本。训练过程中 net_front 保持参数不变，只训练 net_output。注意，此处产生了两次数据集分别训练了 5 个 epoch（因人脸组合过多，仅用同一次产生的数据集容易导致数据分布不均匀产生误差）。这样粗略训练 net_front 和 net_output 之后便可在训练集上判断人脸异同达到 85%左右的正确率。之后对测试集生成的结果在 result_facenet.txt 中。

四、反思与总结

1. 经此次大作业，我认为体会最深的一点就是，大数据时代，一方面模式识别与机器学习具有很强的重要性，可以帮助我们处理数据，另一方面虽然算法上有很多选择，但实际落实到自己用代码处理数据时也有很多处理方式可以优化，比如本项目中本人先基于原数据集产生新数据集，可以省去后续处理的时间。

2. 另一方面，实际上本项目由于时间关系，facenet 的复现部分并未完全实现，原文中 margin 采取的是 0.2，而训练所用时长也远超本人在完成大作业时长，本人只是想探索利用度量学习基于三元组来处理此任务效果确实比较好这件事。
3. 同时，尽管未完整复现 facenet，也发现了很多问题，一方面是可以采用数据增强去增加数据量。原因是本人在对结果进行检查时，发现模型判断受光照等因素影响较大，如同一人的彩色图与黑白图便会被判断为两个人（见图 1）。当然这是极端情况，通过此方法训练出来的模型准确率还是较高，但考虑到实际情况，可以通过数据增强去改变数据的明暗，色调等因素，从而增加数据量，一定程度上提高模型的泛化性与准确率。

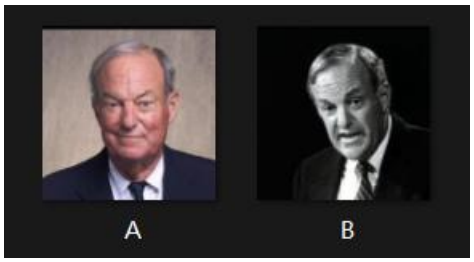

图 1 受光照等因素影响示例

五、参考文献

1. Dlib 实现人脸对齐
   https://blog.csdn.net/liuxiao214/article/details/83411820
2. LBPH 实现人脸识别
   https://blog.csdn.net/cyj972628089/article/details/122055527
3. SVM+PCA 实现人脸识别
   https://blog.csdn.net/qq_42902997/article/details/109186079
4. 三元组
   https://skyfaker.cc/2019/08/22/san-yuan-zu-sun-shi-mo-xing/
5. 三元组的 Dataset 编写时参考
   https://github.com/liorshk/facenet_pytorch/blob/master/TripletFaceDataset.py
6. Facenet
   https://arxiv.org/abs/1503.03832
   https://www.jianshu.com/p/4236679aa3ac