成绩:___

机器学习

基于 BP 神经网络算法的性别识别

姓名：赵清风

班号：HC001502

学号：2015300050

内容摘要：人脸识别技术在现实生活中有着广泛的应用，在人脸识别的基础上可以进行很多应用开发，比如性别识别，年龄识别，人脸比对，美图装饰等。本文基于 python 语言内的计算机视觉库 OpenCV 对人脸识别。利用成熟的 BP 神经网络算法对数据库中的照片进行学习，然后实现对视频中的人脸进行性别识别，最后成功再电脑屏幕显示出来。

关键字：人脸识别；OpenCV;BP 神经网络；python

1.背景介绍

在处于高速发展的现代社会中，“快速、便捷、安全”成为当代社会的代名词。随着计算机视觉技术和光电技术的快速发展，人们越来越希望自己生活在一个安全的环境下，尤其是自己的人身安全以及财产安全。为了让生活更加便捷，使人们在繁重的劳动中解救出来，计算机技术及其应用已经成为人们生活中扮演这必不可少的角色。

其中人工智能技术的开发和运用成为国内外学者和研究人员研究的重点和热点。在人工智能领域中，三维人脸识别技术的研究已经成为一个热门研究领域，它可以应用于需要身份认证的各种各样的场所之中，例如在金融领域中的银行交易身份验证，互联网或网银上的交易身份验证：在国家安全防御中出入安全检查，各种监控系统以及门禁系统等，是解绝身份识别的有效手段。它可以有效避免 2 维人脸识别中存在的缺陷和不足，具有更高的精准性和可信度。早在 1964 年，国外人员就开始从事人脸识别技术的研究。人脸识别是生物识别技术的一种形式，它涉及模式识别，计算机视觉，心理学及生理学及认知科学等方面的诸多方式，在计算机的辅助下实现身份识别，是基于人独有的特征进行身份验证的有效手段。也可以说，人脸识别也是指对于给定一幅人脸作为输入，在待识别的数据库中寻找匹配，在数据库中找到与输入人脸一致的人脸图像。

起初人脸识别是通过计算在二维图像的特征点，如双眼间距，五官比例等，其算法的本质是将人脸分成 N 部分，分部分计算的问题，即将需要识别的人脸输入分类器给出类别判定，相似度最高的是同一人的图像。为了提高人脸识别的准确性，研究人员提出三维人脸识别的设想，经过大量的研究表明，三维人脸信息比二维人脸信息更丰富，减少了使用者因表情变化和姿势改变带来的实验误差，大大的提高了人脸识别的准确性。大量的研究人员投入到三维人睑识别的研究中来，使得三维人睑识别技术得到快速的发展。目前不少的三维人脸扫描设备已经投入商用，例如 Head Color 3D、scanner、3D Capture、VIVID910 等。

人脸识别技术在 20 世纪就开始逐渐进入人们的视野。人脸识别作为生物识别技术的一种形式，它涉及模式识别，计算机视觉，心理学及生理学及认知科学等方面的诸多方式，在计算

机的辅助下实现身份识别，是基于人独有的特征进行身份验证的有效手段。人脸识别并不是一个单独的研究领域。起初，人脸识别只是被作为一般性的模式识别问题进行研究，方法也不是仅仅是针对人脸几何特征实现的算法。到 20 世纪 90 年代后，人睑研究开始突飞猛进，不仅建立了数个大型人睑库的建立，而且出现了一些商业化的人睑识别系统。最为该技术作为获取空间数据的有效手段，能够快速的扫描人脸信息进行辨识，具有扫描速度快，精度高的特点。尽管人脸识别技术早在 20 世纪就开始研究，已经发展多年，但还是未能达到人们的预期目标，因此，人们对人脸识别技术的研究仍然在探索之中。

2.OpenCV 的介绍

OpenCV 是一个基于 BSD 许可（开源）发行的跨平台计算机视觉库，可以运行在 Linux、Windows、Android 和 Mac OS 操作系统上。它轻量级而且高效——由一系列 C 函数和少量 C++ 类构成，同时提供了 Python、Ruby、MATLAB 等语言的接口，实现了图像处理和计算机视觉方面的很多通用算法。

OpenCV 用 C++语言编写，它的主要接口也是 C++语言，但是依然保留了大量的 C 语言接口。该库也有大量的 Python、Java and MATLAB/OCTAVE（版本 2.5）的接口。这些语言的 API 接口函数可以通过在线文档获得。如今也提供对于 C#、Ch、Ruby,GO 的支持。

OpenCV 具有模块化结构，这就意味着开发包里面包含多个共享库或者静态库。下面是可使用的模块：

1. 核心功能（Core functionality）：一个紧凑的模块，定义了基本的数据结构，包括密集的多维 Mat 数组和被其他模块使用的基本功能。
2. 图像处理（Image processing）：一个图像处理模块，它包括线性和非线性图像滤波，几何图形转化（重置大小，放射和透视变形，通用基本表格重置映射），色彩空间转换，直方图等。
3. 影像分析（video）：一个影像分析模块，它包括动作判断，背景弱化和目标跟踪算法。
4. 3D 校准（calib3d）：基于多视图的几何算法，平面和立体摄像机校准，对象姿势判断，立体匹配算法，和 3D 元素的重建。
5. 平面特征（features2d）：突出的特征判断，特征描述和对特征描述的对比。
6. 对象侦查（objdetect）：目标和预定义类别实例化的侦查（例如：脸、眼睛、杯子、人、汽车等等）。

7. highgui:一个容易使用的用户功能界面。

8. 视频输入输出（videoio）:一个容易使用的视频采集和视频解码器。

9. GPU：来自不同 OpenCV 模块的 GPU 加速算法。

10. 一些其他的辅助模块，比如 FLANN 和谷歌的测试封装，Python 绑定和其他。

3. 安装 OpenCV

我们利用 python 里面已有的库来安装 OpenCV。Python 是一种计算机程序设计语言。是一种动态的、面向对象的脚本语言，最初被设计用于编写自动化脚本(shell)，随着版本的不断更新和语言新功能的添加，越来越多被用于独立的、大型项目的开发。

我们只需要通过 pip 或者 anaconda 安装 OpenCV 包就可以通过导入包并使用。

4. BP 神经网络算法介绍和实践

BP 神经网络通常是指基于误差反向传播算法(Back Propagation, 简称 BP 算法)的多层前向神经网络 (Multi-laye Feed-forward Neural Networks, MFNN),它最初是由 Paul Werboss 在 1974 年提 出, 但未在传播，直到 20 世纪 80 年代中期 Rumelhart、Hinton 和 Williams, David Parker 和 Yann Le Cun 重新发现了 BP 算法，同时因此算法被包括在《并行分布处理》(Parallel Distributed Processing)，此算法才广为人知。目前 BP 算法已成为应用最广泛的神经网络学习算法，据统计有近 90%的神经网络应用是基于 BP 算法的。BP 网络的神经元采用的传递函数通常是 Sigmoid 型可微函数，所以可以实现输入和输出间的任意非线性映射，这使得它在诸如信号处理、计算机网络、过程控制、语音识别、函数逼近、模式识别及数据压缩等领域均取得了成功的应用。

4.1 BP 神经网络结构

BP 网络的基本结构如图所示，其输入层节点数为n，输入变量为$x_i(p)(i = 1,2,\dots,n)$，隐层节点数为l，输入层节点与隐层节点间的连接权值为$w_{ji}(i = 1,2,\dots,n; k = 1,2\dots,l)$，隐层节点的阈值为$\theta_j(j = 1,2,..,l)$，隐节点输出为$y_j(p)$；输出层节点数为$m$，隐层节点与输出层节点间的连接权值为$v_{kj}(j = 1,2,\dots,l; k = 1,2,\dots,m)$，输出层节点的阈值为$\phi_k(k = 1,2,\dots,m)$，输出层节点输出为$O_k(p)$。每一个神经元用一个节点表示，网络由输入层、隐层和输出层节点组成。隐层可以是如图所示的一层，也可以是多层，前层到后层的节点通过权联结。隐节点一般采用

Sigmoid 型函数，输入和输出节点可以采用 Sigmoid 型函数或者线性函数。由于采用的是 BP 算法，所以常称为 BP 神经网络。

4.2 BP 算法原理

BP 算法由正向传播和反向传播两部分组成。在正向传播过程中，输入信息从输入层经隐层单元处理后，传至输出层，每一层神经元的状态只影响下一层神经元的状态。如果输出层得不到期望的输出，就转为反向传播，即：把误差信号沿连接路径返回，并通过修改各层神经元之间的连接权值，使误差信号最小。

由于多输入-多输出的网络可以转化为多输入-单输出的情况，所以这里采用的是多输入-单输出的神经模型。假设网络由M层，而第M层仅含输出节点，第一层为输入节点。另外，假设有N个标准样本对$undefined$，对网络的第M层的输出为$O(p)$。而且，隐层节点和输出层节点的作用函数为 SIgmoid 行函数，即

$$f(x)=\frac{1}{1+e^{-Qx}} \tag{4.1}$$

它反映了神经元的饱和特性，在 0 和 1 之间取值。上式中，Q为表示神经元非线性的参数，称增益型(Gain)，也称调节参数。Q值越大，S型曲线越陡峭；反之，Q值越小，S型曲线越平坦；一般取$Q=1$。

假设对某一输入$x(p)$，网络输出为$O(p)$，节点i的输出为x_i，节点j的输入为

$$net_j=\sum_i w_{ji}\,x_i \tag{4.2}$$

节点j的输出为:$y_j=f(net_j)$

并将误差函数定义为

$$E=\frac{1}{2N}\sum_{p=l}^{N}(O_p-T_p)^2 \tag{4.3}$$

其中T_p为网络目标输出。

现定义$E=\frac{1}{2}(O(p)-T(p))^2,\delta_j=\frac{\partial E}{\partial net_j}$,且$y_j=f(net_j)$，于是有

$$\frac{\partial E}{\partial W_{ji}}=\frac{\partial E}{\partial net_j}.\frac{\partial net_j}{\partial W_j i}=\frac{\partial E}{\partial net_j}x_i=\delta_j x_i \tag{4.4}$$

下面分两种情况来讨论：

（1） 当j为输出节点时，$y_j = O$

$$\delta_j = \frac{\partial E}{\partial net_j} = \frac{\partial E}{\partial O} \cdot \frac{\partial O}{\partial net_j} = -(T - O) f'(net_j) \qquad (4.5)$$

(2)若j隐层节点时，则有

$$\delta_j = \frac{\partial E}{\partial net_j} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial net_j} = \frac{\partial E}{\partial y_j} \cdot f'(net_j) \qquad (4.6)$$

式中y_j是送到下层$(l+1)$的输入，

计算$\frac{\partial E}{\partial y_j}$要从$(l+1)$层算回来。 在$(l+1)$层第$m$个节点：

$$\frac{\partial E}{\partial y_j} = \sum_m \frac{\partial E}{\partial net_j} \cdot \frac{\partial net_j}{\partial y_j} = \sum_m \frac{\partial E}{\partial net_m} \cdot \frac{\partial}{\partial y_j} \sum_j W_{jm} O_j \qquad (4.7)$$

$$= \sum_m \frac{\partial E}{\partial net_m} \cdot \sum_j W_{jm} = \sum_m \delta_m W_{jm}$$

由以上两式可以得到：

$$\delta_j = f'(net_j) \sum_m \delta_M W_{jm} \qquad (4.8)$$

$$\frac{\partial E}{\partial W_{jm}} = \delta_j y_j$$

又由于

$$f'(net_j) = \frac{dE}{dnet_j} \left[\frac{1}{1e^{-Qnet_j}}\right] = Q(1 + e^{-qnet_j})^{-2} . e^{-Qnet_j} \qquad (4.9)$$

$$= Q . f(net_j) . (1 + f(net_j)) = Q . y_j . (1 + y_j)$$

根据前面的推导过程,具体的 BP 算法步骤可概括如下：

第一步，选取初始权值、阈值。

第二步，重复下述过程直至满足性能要求为止：

1. 对于学习样本$p = 1$到N

1） 计算每层各节点j的y_j，net_j和O的值（正向过程）；

2 ）对各层从M层到第二层，对每层各节点，由式(4.5)和(4.8)反向计算δ_j（反向过

程）；

2. 修正权值

$$W_{ji}((t+1) = W_{ji}(t) - \eta \frac{\partial E}{\partial W_{ji}}, \eta > 0$$

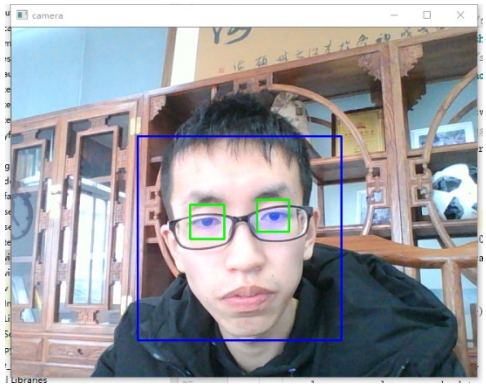
其中$\frac{\partial E}{\partial W_{ji}} = \sum_{p=1}^{N} \frac{\partial E}{\partial W_{ji}}$, t为迭代次数,η为步长。

4.3 BP 算法收敛性

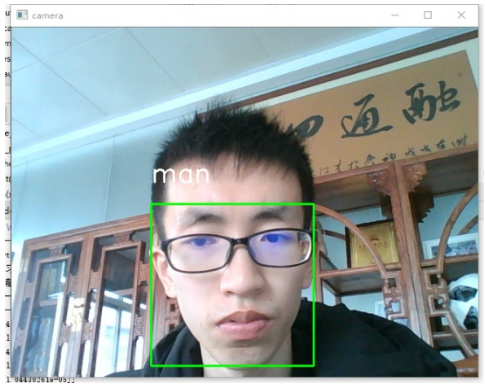
BP 算法是最小均方误差的近似最快下降算法，其性能曲面可能有多个局部极小点而且在参数空间的不同区域曲率也是变化的。依照 BP 网络的性能曲面，设置 BP 算法的初始参数时需注意， 首先不能把初始参数设置为 0。这是由于对性能曲面来说，参数空间的原点趋向鞍点。其次，不能把初始参数设置过大。这是由于在远离优化点的位置，性能曲面将变得十分平坦。典型情况下，可以选择一些小的随机值作为初始权值和偏置值。这样可以在不离开性能曲面平坦区域的同时避开可能的鞍点。也可用 S 型传递函数的形状和输入变量的范围决定权值的大小，然后用偏置值将 S 型函数的形状移到操作区域的中央国）。通过调整参数，BP 算法最终收敛到最优化的解，但收敛的速度很慢。如果提高学习速度，算法将通过初始的平坦曲面而快速收敛，即可以在平坦曲面时增加学习速度，在斜率增加时减少学习。另外，也可采用平滑轨迹，即当算法开始发散时，用平均改变参数的方法过滤轨迹可以提高收敛性。

5. 结果展示

在此，我们是基于 python 语言。学习了 200 张男生照片和 200 张女生照片。运用了三层神经网络，一个输入层，一个隐藏层和一个输出层。下面是最终结果。



图片 1. 视频识别人脸



图片 2. 视频识别性别