Face recognition with opencv

周强

2017 年 7 月 21 日

目录

1 Introduction

OpenCV（开源计算机视觉）是 1999 年由英特尔开发的流行的计算机视觉库。跨平台库将重点放在实时图像处理上，并且包括无专利的最新计算机视觉算法实现。2008 年，Willow Garage 接手了支持，OpenCV 2.3.1 现在配备了 C，C ++，Python 和 Android 的编程接口。OpenCV 根据 BSD 许可证发布，因此被用于学术项目和商业产品。

OpenCV 2.4 现在提供了面向人脸识别的全新 FaceRecognizer 类，因此您可以立即开始实验人脸识别。这份文件是我所期待的指导，当我自己进行人脸识别。它向您展示如何使用 OpenCV 中的 FaceRecognizer 执行人脸识别（具有完整的源代码清单），并为您介绍后面的算法。我还将展示如何创建可以在许多出版物中找到的可视化，因为很多人都要求。

目前可用的算法有：

- 特征脸（请参阅 createEigenFaceRecognizer）
- Fisherfaces（请参阅 createFisherFaceRecognizer）
- 本地二进制模式直方图（请参阅 createLBPHFaceRecognizer）

您不需要从此页面复制和粘贴源代码示例，因为它们在本文档附带的 src 文件夹中可用。如果您已打开 OpenCV 打开样品，那么您已经编译好的机会很好！虽然对于非常高级的用户来说，这可能是有趣的，但我决定将实施细节放在外面，因为我们担心会混淆新用户。

本文档中的所有代码均以 BSD 许可证发布，因此请随时将其用于您的项目。

2 人脸识别

人脸识别对人类而言是一项容易的任务。实验表明，即使一到三天的婴儿也能够区分已知的面孔。那么电脑有多难？事实证明，到目前为止，我们对于人类如何进行识别了解很少。内部特征（眼睛，鼻子，嘴巴）或外部特征（头部形状，发际线）是否用于成功的脸部识别？我们如何分析图像，大脑如何编码？David Hubel 和 Torsten Wiesel 表明，我们的大脑具有响应于场景的特定局部特征（如线条，边缘，角度或运动）的特殊神经细胞。由于我们看不到这个世界是分散的，我们的视觉皮层必须以不同的方式将不同的信息来源组合成有用的模式。自动人脸识别是关于从图像中提取有意义的特征，将其置于有用的表示中并对其进行某种分类。

基于面部几何特征的人脸识别可能是最直观的面部识别方法。[1] 中描述了第一个自动人脸识别系统之一：标记点（眼睛，耳朵，鼻子，... 的位置）用于构建特征向量（点之间的距离，它们之间的角度，... ）。通过计算待判断图像的特征向量与参考图像之间的欧氏距离来进行识别。这种方法对于照明的性质而言是可靠的，但是具有巨大的缺点：即使使用现有技术的算法，标记点的精确配准也是复杂的。

[2] 中描述的特征脸（Eigenfaces）方法采用了整体的人脸识别方法：面部图像是高维图像空间的一个点，找到了其较低维度的表示方式，使得分类变得容易。使用主成分分析找到低维子空间，主成分分析识别具有最大方差的轴。虽然从重建的角度来看，这种转型是最佳的，但并没有考虑任何类标签信息。想象一下，如果是外部来源导致了较大的方差，比如光照变化。具有最大方差的轴根本不一定包含任何判别信息，因此不可能进行分类。因此，采用线性判别分析的类特定投影开始应用于人脸识别 [3]。基本思想是最小化类内的方差，同时最大化类之间的方差。

最近出现了用于局部特征提取的各种方法。为了避免输入数据的高维度，仅描绘图像的局部区域，所提取的特征（有希望地）对于部分遮挡，照明和小样本大小更有效。用于局部特征提取的算法是 Gabor 小波 [4]，离散余弦变换 [5] 和局部二进制模式 [6]。这仍然是一个开放的研究问题：当应用局部特征提取时，保留空间信息的最佳方式是什么？因为空间信息是潜在有用的信息。

3 人脸图像数据集

让我们先来一些数据进行实验。我不想在这里做一个玩具的例子。我们正在做人脸识别，所以你需要一些脸部图像！您可以创建自己的数据集，或者从一个可用的人脸数据集开始，http://face-rec.org/databases/ 提供最新的概述。三个有趣的数据库（部分描述来自 http://face-rec.org）是：

- AT & T Facedatabase AT & T Facedatabase 有时也被称为 ORL 数据库面，包含 40 个不同主题中的每一个的十个不同的图像。对于某些科目，图像在不同时间拍摄，改变照明，面部表情（开放/闭合的眼睛，微笑/不微笑）和面部细节（眼镜/无眼镜）。所有的图像都是针对黑暗均匀的背景拍摄，受试者处于直立的正面位置（对于某些侧面运动具有容忍性）。
- Yale Facedatabase A ， 也称为 Yalefaces。AT & T Facedatabase 适用于初始测试，但它是一个相当简单的数据库。Eigenfaces 方法已经具有 97%的识别率，所以你不会看到其他算法有很大的改进。Yale Facedatabase A（也称为 Yalefaces）是初步实验的更合适的数据集，因为识别问题更难。数据库由 15 人（14 男，1 女）组成，每个人拥有 11 个灰度图像，尺寸为 320×243 像素。光照条件（中心光，左光，右光），面部表情（快乐，正常，悲伤，困倦，惊讶，眨眼）和眼镜（眼镜，无眼镜）都有变化。

  原始图像不被裁剪和对齐。请查看附录中的 Python 脚本，这是为您做的。
- Extended Yale Facedatabase B 扩展的耶鲁大学数据库 B 包含 38 个不同人的裁剪版本的 2414 图像。该数据库的重点放在提取对照明强大的特征，图像几乎没有情感/遮挡的变化。我个人认为，这个数据集对于我在本文档中执行的实验来说太大了。您最好使用 AT & T Facedatabase 进行初步测试。在 [3] 中使用了耶鲁 Facedatabase B 的第一个版本，以了解特征面和 Fisherfaces 方法在重度照明变化下的性能。[7] 使用相同的设置来拍摄 16128 张 28 人的图像。扩展耶鲁 Facedatabase B 是两个数据库的合并，这两个数据库现在称为扩展 Yalefacedatabase B.

3.1 准备数据

一旦我们获得了一些数据，我们将需要在我们的程序中读取。在演示应用程序中，我决定从非常简单的 CSV 文件读取图像。为什么？因为它是最简单的平台无关的方法，我可以想到。但是，如果你知道一个更简单的解决方案，请给我看看。基本上 CSV 文件都需要包含一个文件名，后跟一个; 后面是标签（作为整数），组成一行如下：

```
/path/to/image.ext; 0
```

让我们剖析一下。/path/to/image.ext 是图像的路径，如果您在 Windows 中，可能是这样的：C:/faces/person0/image0.jpg。然后有分隔符; 最后我们将标签 0 分配给图像。将标签视为该图像所属的主题（该人物），所以相同的主题（人物）应该具有相同的标签。

从 AT & T Facedatabase 下载 AT & T Facedatabase，并从 at.txt 下载相应的 CSV 文件，该文件看起来像这样：

```
./at/s1/1.pgm;  0
./at/s1/2.pgm;  0
...
./at/s2/1.pgm;  1
./at/s2/2.pgm;  1
...
./at/s40/1.pgm;  39
./at/s40/2.pgm;  39
```

想象一下，我将文件解压缩到 D:/data/at，并将 CSV 文件下载到 D:/data/at.txt。那么你只需要用 D:/data/来搜索和替换./。您可以在选择的编辑器中执行此操作，每个足够高级的编辑器都可以执行此操作。一旦拥有有效的文件名和标签的 CSV 文件，您可以通过将路径传递到 CSV 文件作为参数来运行任何演示实例：

```
facerec_demo.exe  D:/data/at.txt
```

有关创建 CSV 文件的详细信息，请参阅创建 CSV 文件。

4 Eigenfaces

我们给出的图像表示的问题是它的高维度。二维 $p \times q$ 灰度图像跨越 $m = pq$ 维向量空间，因此具有 100×100 像素的图像已经存在于 10,000 维图像空间中。问题是：所有维度对我们来说同样有用吗？我们只能利用数据的差异性来做出决定，因此我们正在寻找的是能表示大部分信息的成分。主成分分析（PCA）由 Karl Pearson（1901）和 Harold Hotelling（1933）独立提出，将一组可能相关的变量转化为较小的一组不相关变量。这个想法是，高维数据集通常由相关变量描述，因此只有少数有意义的维度占据了大部分信息。PCA 方法在数据中找到方差最大的方向，称为主成分。

4.1 特征脸方法的算法描述

令 $X = \{x_1, x_2, \cdots, x_n\}$ be a random vector with observations $x_i \in R^d$.

1. 计算均值 μ

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

2. 计算协方差矩阵 S

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$$

3. 计算协方差矩阵 S 的特征值 λ_i 和特征矢量 v_i

$$Sv_i = \lambda_i v_i, \;\; i = 1, 2, \cdots, n$$

4. 将特征向量按其特征值进行递减排序。k 个主成分是与 k 个最大特征值对应的特征向量。观测矢量 x 的 k 个主成分由下式给出：

$$y = W^T(x - \mu)$$

其中，$W = (v_1, v_2, \cdots, v_k)$。

Eigenfaces 方法然后通过以下方式进行人脸识别：

- 将所有训练样本投影到 PCA 子空间。
- 将查询图像投影到 PCA 子空间中。
- 找到投影的训练图像和投影的查询图像之间的最近邻居。

还有一个问题需要解决。想象一下，我们给了 400 张 100×100 像素的图像。主成分分析求解协方差矩阵 $S = XX^T$，其中 size（X）= 10000×400。你会得到一个 10000×10000 的矩阵，大概是 0.8GB。解决这个问题是不可行的，所以我们需要应用一个技巧。从您的线性代数课程可知，M> N 的 M×N 矩阵只能具有 N-1 个非零特征值。因此，可以取而代之计算大小为 N×N 的矩阵 $S' = X^T X$ 的特征值分解：

$$X^T X v_i = \lambda_i v_i$$

并用数据矩阵的左乘法得到原始协方差矩阵 $S = XX^T$ 的特征向量：

$$XX^T\ Xv_i) = \lambda_i (Xv_i)$$

得到的特征向量是正交的，得到正交特征向量，它们需要被归一化为单位长度。我不想把它变成一个出版物，所以请研究 [8] 推导和证明方程式。

4.2 Eigenfaces in OpenCV

完整的程序参见当前文件夹下的：facerec_eigenfaces.cpp

我使用了 jet colormap，所以你可以看到灰度值在特定的特征脸内是如何分布的。您可以看到，特征脸不仅可以对面部特征进行编码，还可以对图像中的照明进行编码（参见特征面 4 中的左侧光线，特征曲线# 5 中的右侧光）：

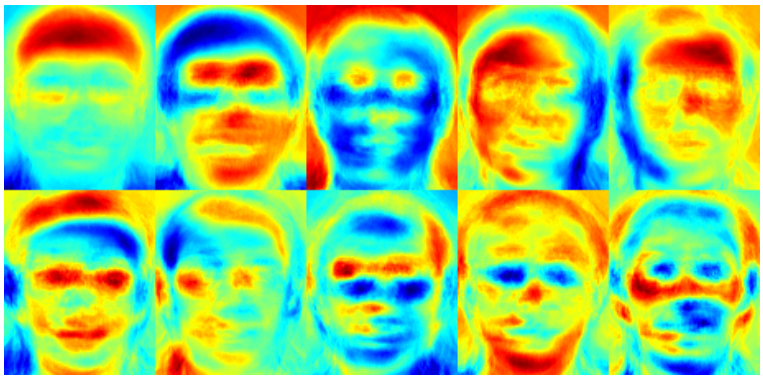

图 1: eigenfaces

我们已经看到，我们可以从其低维近似重建一个面部图像。所以让我们来看一下好的重建需要多少个特征脸。下图中的子图分别是用 10,30，...，310 个特征脸重建的效果：

```
1         // Display or save the image reconstruction at some predefined steps:
2         for(int num_components = 10; num_components < 300; num_components+=15) {
```

```
3                 // slice the eigenvectors from the model
4                 Mat evs = Mat(W, Range::all(), Range(0, num_components));
5                 Mat projection = LDA::subspaceProject(evs, mean, images[0].reshape(1,1));
6                 Mat reconstruction = LDA::subspaceReconstruct(evs, mean, projection);
7                  // Normalize the result:
8                 reconstruction = norm_0_255(reconstruction.reshape(1, images[0].rows));
9                  // Display or save:
10                if(argc == 2) {
11                                imshow(format("eigenface_reconstruction_%d", num_components), reconst
12                } else {
13                                imwrite(format("%s/eigenface_reconstruction_%d.png", output_folder.c_
14                }
15        }
```

10 个特征向量显然不足以进行良好的图像重建，50 个特征向量可能已经足以编码重要的面部特征。您将获得 AT & T Facedatabase 的大约 300 个特征向量的良好重建。有一个经验法则，你应该选择多少特征面才能成功的面部识别，但它在很大程度上取决于输入数据。[9] 是开始研究的完美点：

图 2: eigenfaces_reconstruction

5 Fisherfaces

主成分分析（PCA）是 Eigenfaces 方法的核心，它找到最大化数据总方差的特征的线性组合。虽然这显然是一种表达数据的强大方法，但它并不考虑任何类，所以当抛出成分时，很多判别信息可能会丢失。想象一下，您的数据的变化是由外部来源产生的，让它成为光。由 PCA 识别的成分根本不一定包含任何判别信息，因此投影

后的样本被涂抹在一起,并且不可能进行分类(参见 http://www.bytefish.de/wiki/pca_lda_with_gnu_octave 为例)。

线性判别分析执行类别特定的维数降低，并由伟大的统计学家 R. A. Fisher 爵士发明。他在 1936 年的论文中成功地将其用于分类花。在分类问题中使用多重测量 [10]。为了找到在类之间最好分离的特征的组合，线性判别分析最大化了类之间的类之间的散射，而不是最大化整体散射。这个想法很简单：相同的类应该集中在一起，而不同的类在低维表示中尽可能远离彼此。这也被 Belhumeur，Hespanha 和 Kriegman 认可，所以他们在 [3] 中应用了判别分析来进行人脸识别。

5.1 Fisherfaces 方法的算法描述

令 X 是一个随机向量，其中样本来自 C 个类：

$$X = \{X_1, X_2, \cdots, X_c\}$$

$$X_i = \{x_1, x_2, \cdots, x_n\}$$

类间（between-class）分散矩阵和类内（within-class）分散矩阵计算如下：

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{1}$$

$$S_W = \sum_{i=1}^{c} \sum_{x_j \in X_i} (x_j - \mu_i)(x_j - \mu_i)^T \tag{2}$$

其中，μ 是所有样本的均值：

$$\mu = \frac{1}{N} \sum_{i=1}^{N} x_i$$

μ_i 是类 $i \in \{1, 2, \cdots, c\}$ 的均值：

$$\mu_i = \frac{1}{|X_i|} \sum_{x_j \in X_i} x_j$$

Fisher 的经典算法现在寻找一个投影 W，最大化了分类可分性标准：

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|} \tag{3}$$

根据文献 [3]，通过求解一般特征值问题给出了该优化问题的解决方案：

$$S_B v_i = \lambda_i S_W v_i \tag{4}$$

$$S_W^{-1} S_B v_i = \lambda_i v_i \tag{5}$$

有一个问题需要解决：S_W 的秩最多 $(N - c)$，其中 N 个样本和 c 个类。在模式识别问题中，样本数 N 几乎总是比输入数据的维数（像素数）小，所以散射矩阵 SW 变得奇异（参见 [**?**]）。在 [3] 中，这通过对数据执行主成分分析并将样本投影到 $(N - c)$ 维空间来解决。然后对减少的数据进行线性判别分析，因为此时 S_W 不再是奇异。

优化问题可以重写为：

$$W_{pca} = \arg\max_{W} |W^T S_T W| \tag{6}$$

$$W_{fld} = \arg\max_{W} \frac{|W^T W_{pca}^T S_B W_{pca} W|}{|W^T W_{pca}^T S_W W_{pca} W|} \tag{7}$$

然后，将样本投影到（c-1）维空间中的变换矩阵 W 由下式给出：

$$W = W_{fld}^T W_{pca}^T \tag{8}$$

5.2 Fisherfaces in OpenCV

完整的程序参阅当前目录下的facerec_fisherfaces.cpp。

For this example I am going to use the Yale Facedatabase A, just because the plots are nicer. Each Fisherface has the same length as an original image, thus it can be displayed as an image. The demo shows (or saves) the first, at most 16 Fisherfaces:

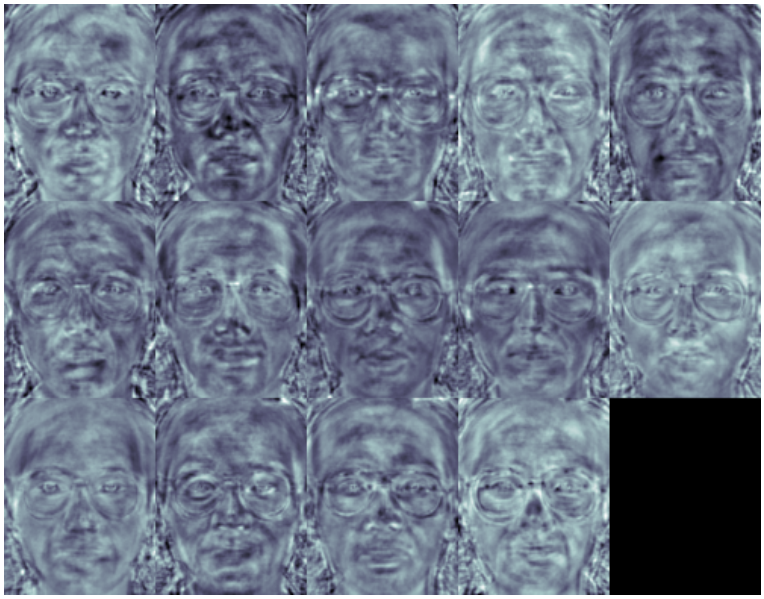

图 3: fisherfaces

Fisherfaces 方法学习一个特定于类的转换矩阵，使得它们不像 Eigenfaces 方法那样明显地捕捉照明。判别分析反而发现面部特征来区分人。值得一提的是，Fisherfaces 的表现也很大程度上取决于输入数据。实际上说：如果你只学习光照良好图片的 Fisherfaces，并尝试识别不良照明场景中的脸部，那么方法很可能会发现错误的成分（只是因为这些特征可能不会在不良照明图像上占优势）。这是有点合理的，因为该方法没有机会学习照明。

Fisherfaces 可以像 Eigenfaces 一样重建投影图像。但是，由于我们只确定了区分不同类的特征，所以您不能期望原始图像的良好重建。对于 Fisherfaces 方法，我们将将样本图像投影到每个 Fisherfaces 上。所以你会有一个很好的可视化，每个 Fisherfaces 描述的特征。

差异对于人眼可能是微不足道的，但您应该能够看到一些差异：

图 4: fisherface_reconstruction_opencv

6 局部二进制模式直方图：Local Binary Patterns Histograms

特征脸和 Fisherfaces 采取了一种整体的方法来进行人脸识别。您将数据视为高维图像空间中的某个矢量。我们都知道高维度是坏的，所以确定了一个较低维的子空间，其中（可能）有用的信息被保留。特征脸方法最大化总方差，如果方差是由外部源产生的，则可能导致问题，因为所有类别上具有最大方差的分量不一定对分类有用（参见 http://www.bytefish.de /维基/ pca_lda_with_gnu_octave）。因此，为了保留一些判别信息，我们应用了线性判别分析，并按照 Fisherfaces 方法的描述进行了优化。Fisherfaces 方法运行良好，至少对于我们在模型中假设的约束场景。

现在现实生活并不完美。您根本不能保证您的图像中完美的光线设置或则每个人都有 10 种不同的图像。那么如果每个人只有一个图像呢？我们对子空间的协方差估计可能是非常错误的，所以识别也是如此。记住，Eigenfaces 方法在 AT & T Facedatabase 上有 96%的识别率？我们实际需要多少幅图像来获得有用的估计值？以下是 AT & T Facedatabase 的特征脸和 Fisherfaces 方法的等级 1 识别率，这是一个相当容易的图像数据库：

所以为了获得良好的识别率，您将需要每个人至少 8（+ - 1）个图像，而 Fisherfaces 方法在这里并不真正有帮助。上述实验是使用 facerec 框架执行的十倍交叉验证结果：(https：//github.com/bytefish/facerec)。这不是出版物，所以我不会用深入的数学分析来回答这些数字。

所以一些研究集中在从图像中提取局部特征。这个想法不是把整个图像看成一个高维向量，而只是描述一个对象的局部特征。您以这种方式提取的特征将隐含地具有低维度。一个好主意！但是，您将很快观察到我们给出的图像表示，不仅会受到照明变化的影响。想像像图像中的缩放，平移或旋转这样的东西 - 你的局部描述必须至少对这些东西有一些鲁棒性。就像 SIFT 一样，“局部二进制模式”方法的根源在于 2D 纹理分析。局部二进制模式的基本思想是通过将每个像素与其邻域进行比较来总结图像中的局部结构。以像素为中心，并对其邻居进行阈值。如果中心像素的强度大于等于其邻居，则表示为 1，如果不是则为 0。你会得到每个像素的二进制数，就像：

1. 所以如果有 8 个周围像素，你会得到 2^8 可能的组合，称为局部二进制模式，有时也称为 LBP 编码。文

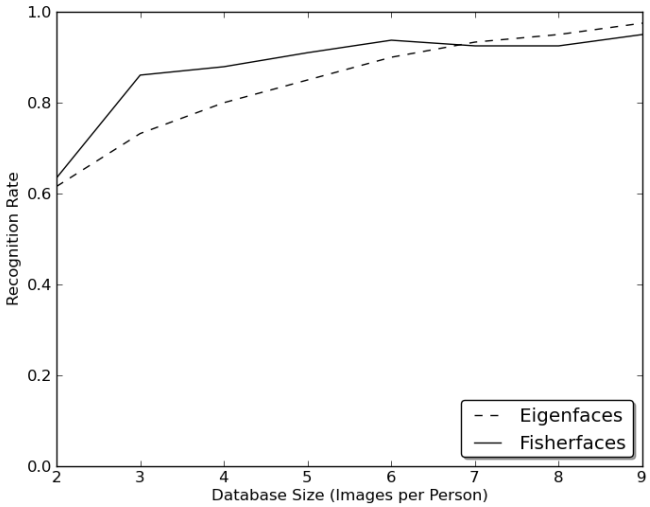

图 5: at_database_small_sample_size

献中描述的第一个 LBP 算子实际上使用了一个固定的 3 x 3 邻域，就像这样：

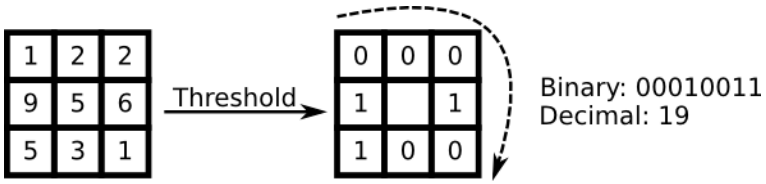

图 6: LBP

6.1 LBPH 方法的算法描述

对 LBP 算子的更正式地描述如下

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^p s(i_p - i_c) \tag{9}$$

其中，i_c 是中心像素 (x_c, y_c) 的强度，i_p 是相邻像素的强度。s 是如下定义的符号函数：

$$s(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{else} \end{cases} \tag{10}$$

此描述使您能够捕获图像中非常细粒度的细节。事实上，作者能够与纹理分类的最新技术结果进行竞争。算法发布后不久，有人指出，一个固定的邻域（neighborhood）不能对不同尺度的细节进行编码。所以算子被扩展到在 [3] 中使用一个可变邻域。这个想法是将圆周上的邻居数量与可变半径对齐，这样可以捕获以下邻域：

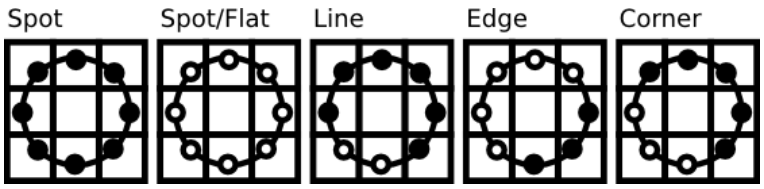

图 7: patterns

For a given Point (x_c, y_c), the position of the neighbor (x_p, y_p), $p \in P$ can be calculated by:

$$x_p = x_c + R\cos(\frac{2\pi p}{P}) \tag{11}$$

$$y_p = y_c - R\sin(\frac{2\pi p}{P}) \tag{12}$$

Where R is the radius of the circle and P is the number of sample points.

该算子是原始 LBP 代码的扩展，因此有时称为扩展 LBP（也称为圆形 LBP）。如果圆上的点坐标与图像坐标不对应，则内插点。计算机科学有一堆聪明的插值方案，OpenCV 实现一个双线性插值：

$$f(x,y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix} \tag{13}$$

根据定义，LBP 算子对单调灰度变换是鲁棒的。我们可以通过查看人工修改的图像的 LBP 图像来轻松地验证这一点（所以你看到一个 LBP 图像是什么样的！）：

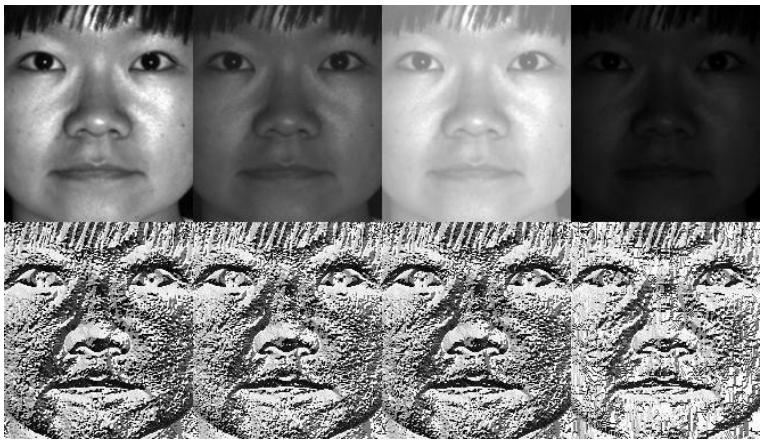

图 8: LBP_yale

那么剩下的就是如何将空间信息整合到人脸识别模型中。Ahonen 等人提出的表述 [6] 是将 LBP 图像划分为 m 个局部区域，并从每个区域提取直方图。然后通过连接局部直方图（不合并它们）获得空间增强的特征向量。这些直方图称为局部二进制模式直方图。

6.2 Local Binary Patterns Histograms in OpenCV

完整程序参阅当前文件夹下的 src/facerec_lbph.cpp

7 结论

您已经学会了如何在实际应用中使用新的 FaceRecognizer。阅读文档后，您也知道算法如何工作，所以现在是时候尝试可用的算法了。使用它们，改进它们，让 OpenCV 社区参与！

8 Credits

如果没有使用 AT & T 人脸数据库和耶鲁 Facedatabase A / B 的脸部图像的类型许可，本文档将无法实现。

8.1 AT&T 人脸数据库

重要提示：使用这些图像时，请注明“剑桥 AT&T 实验室”。

人脸数据库，之前是 ORL 人脸数据库，面包含一组 1992 年 4 月至 1994 年 4 月期间拍摄的面部图像。该数据库用于剑桥大学工程系进行的与语音，视觉和机器人组合作进行的人脸识别项目。

该数据库包含 40 个不同的人脸，每个人脸有十张不同的图像。对于某些人脸，图像在不同时间拍摄，改变照明，面部表情（开放/闭合的眼睛，微笑/不微笑）和面部细节（眼镜/无眼镜）。所有的图像都是针对黑暗均匀的背景拍摄，受试者处于直立的正面位置（对于某些侧面运动具有容忍性）。

这些文件采用 PGM 格式。每个图像的大小为 92x112 像素，每像素 256 个灰度级。图像组织在 40 个目录（每个主题一个），其名称为 s_X，其中 X 表示主题编号（介于 1 和 40 之间）。在这些目录的每个目录中，该目标人脸有十个不同的图像，它们具有 Y.pgm 格式的名称，其中 Y 是该主题的图像编号（1 到 10 之间）。

数据库的副本可以从以下网址获取：http://www.cl.cam.ac.uk/research/dtg/attarchive/pub/data/att_faces.zip。

8.2 耶鲁大学 Facedatabase A

耶鲁脸数据库 A（尺寸 6.4MB）包含 15 个人的 GIF 格式的 165 灰度图像。每个科目有 11 张图像，每个不同的面部表情或配置，每个不同的面部表情或配置：中心灯，眼镜，快乐，左光，无眼镜，正常，右光，悲伤，困倦，惊讶和眨眼。（来源：http://cvc.yale.edu/projects/yalefaces/yalefaces.html）

8.3 耶鲁 Facedatabase B

扩展的耶鲁脸数据库 B 包含 28 个人类受试者在 9 个姿势和 64 个照明条件下的 16128 张图像。该数据库的数据格式与耶鲁脸数据库 B 相同。有关数据格式的更多详细信息，请参考耶鲁脸数据库 B 的主页（或本页的一个副本）。

您可以自由使用扩展的耶鲁脸数据库 B 进行研究。使用这个数据库的所有出版物应该承认使用“the Exteded Yale Face Database B”，并且引用 Athinodoros Georghiades，Peter Belhumeur 和 David Kriegman 的文章 [?]。

扩展的耶鲁脸数据库 B 使用的所有测试图像数据手动对齐，裁剪，然后重新调整为 168x192 张图像。如果您使用裁剪的图像发布实验结果，请参考 PAMI2005 文章。(资料来源：http://vision.ucsd.edu/ leekc/ExtYale-Database/ExtYaleB.html)

9 附录

9.1 Creating the CSV File

您肯定不想手动创建 CSV 文件。我已经准备好一些 Python 脚本src/create_csv.py，会自动创建一个 CSV 文件。

9.2 Aligning Face Images

您的图像数据的准确对齐在像情绪检测这样的任务中尤为重要。相信我，你肯定不想手工做。所以我为你准备了一个很小的 Python 脚本。该代码真的很容易使用。要缩放，旋转和裁剪脸部图片，您只需要调用 CropFace（image，eye_left，eye_right，offset_pct，dest_sz）的，其中：

- eye_left 是左眼的位置
- eye_right 是右眼的位置
- offset_pct 是要保持在眼睛旁边的图像的百分比（水平，垂直方向）
- dest_sz 是输出图像的大小

如果您的图像使用相同的 offset_pct 和 dest_sz，则它们都在眼睛对齐。

想象一下，我们得到了阿诺德 · 施瓦辛格（Arnold Schwarzenegger）的这张照片。眼睛的（x，y）位置对于左眼约为（252,364），右眼为（420,366）。现在，您只需要定义水平偏移，垂直偏移，以及缩放，旋转和裁剪之后面部图像应具有的尺寸。

这里有些例子：

图 9: 0.1, 0.1, (200, 200)

图 10: 0.2, 0.2, (200, 200)

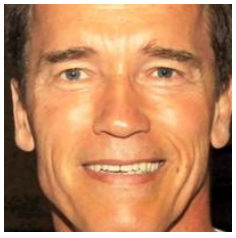

图 11: 0.3, 0.3, (200, 200)

图 12: 0.2, 0.2, (70, 70)

参考文献

[1] Takeo Kanade. Picture processing system by computer complex and recognition of human faces, November 1973.

[2] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, January 1991.

[3] Peter N. Belhumeur, João P. Hespanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, 1997.

[4] Laurenz Wiskott, Jean-Marc Fellous, Norbert Krüger, and Christopher von der Malsburg. Face recognition by elastic bunch graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):775–779, July 1997.

[5] Kieron Messer, Josef Kittler, James Short, G. Heusch, Fabien Cardinaux, Sebastien Marcel, Yann Rodriguez, Shiguang Shan, Y. Su, Wen Gao, and X. Chen. *Performance Characterisation of Face Recognition Algorithms and Their Sensitivity to Severe Illumination Changes*, pages 1–11. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[6] Timo Ahonen, Abdenour Hadid, and Matti Pietikäinen. *Face Recognition with Local Binary Patterns*, pages 469–481. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[7] Kuang-Chih Lee, Jeffrey Ho, and David J. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(5):684–698, May 2005.

[8] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2Nd Edition)*. Wiley-Interscience, 2000.

[9] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, December 2003.

[10] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.