

基于 Haar 特征、肤色特征和 BP 神经网络的人脸识别算法

陆世豪（919101960122 机械工程学院）

摘要：为了实现对图片中人脸的检测（对出现人脸的区域划定矩形框），本文设计了一种使用 Haar 特征、肤色特征和 BP 神经网络的方法，通过两种特征各自构建和训练 BP 神经网络，两个神经网络直接级联获得人脸图像的分类器，在图片中使用滑动窗口和训练好的分类器进行检测，从而画出人脸的所在范围。经过测试，这种算法有较高的准确率，并且相对普通的基于 Haar 特征的分类器有较快的检测速度。

关键词：人脸识别，机器学习，Haar 特征，肤色特征，BP 神经网络

1. 引言

人脸识别是指在输入图像中确定所有人脸的位置、大小、位姿的过程。人脸检测作为人脸信息处理中的一项关键技术，近年来成为模式识别与计算机视觉领域内一项受到普遍重视、研究十分活跃的课题。人脸模式具有复杂而细致的变化，因此一般需要采用多种模式特征综合的方法才能加以检测。其中，Haar 特征就是用于物体识别的一种数字图像特征其具有检测速度快，计算过程简单等特点，但由于图像在计算 Haar 特征时会丢失颜色形状等有助于分类的信息，所以其不易对人脸和非人脸特征做出更加有效的区分，为了解决这个问题，可以在 Haar 特征的基础上加上对颜色特征的区分。既能够在大型图片中提高检测速度又能进一步提高检测准确率。而 BP 神经网络每次根据训练得到的结果与预想结果进行误差分析，进而修改权值和阈值，一步一步得到能输出和预想结果一致的模型。将两种特征和 BP 神经网络相结合，针对两种特征均采用 BP 神经网络进行监督学习，最后两个神经网络直接级联构成分类器对图像进行检测。^[1]

2. 相关研究

早期的人脸识别算法使用了模板匹配技术，即用一个人脸模板图像与被检测图像中的各个位置进行匹配，确定这个位置处是否有人脸。本文使用的 BP 神经网络，Haar 特征都是这一技术的分支。2000 年左右又出现了基于 PAC 学习理论的人脸检测算法，检测速度较之前的方法有 2 个数量级的提高。^[2]进入深度学习时代后，卷积神经网络在图像分类问题上取得成功之后很快被用于人脸识别问题，在精度上大幅度超越之前的框架，当一些图像由于太大而计算量无法实现时，可以使用卷积神经网络直接进行识别。虽然这些算法发展和研究很快，但最初的模板匹配技术依然应用于很多场景，其优点在于简单易懂，代码易于维护，这也是本文选择基于此技术开发算法的原因。

3. 实现算法

3.1. Haar 特征

Haar 特征是一种反映图像的灰度变化的，像素分模块求差值的一种特征。它对一些简单的图形结构如边缘线段等比较敏感，人脸上的一些特征符合这些图形特征的简单组合，如眼睛的颜色深于周围的皮肤，他们之间有明显的分界线，嘴巴和鼻子的阴影可以用一个黑色矩形来简单描述等。^[3]为了对人脸的各种特点进行描述，本文使用了以下四种特征模板。



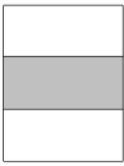

名称（模板号）	特征模板	
边缘特征 (1)(2)		
线性特征 (3)(4)		

图 1 特征模板

边缘特征（1）（2）为长和宽为 2 和 1 的矩形框，其中长和宽可以按任意整数比例放大，边缘特征（3）（4）为长和宽为 3 和 1 的矩形框，其长度比例也可以按整数比放大，需要注意的是，不同位置 and 不同大小的同种特征模板应该看做不同的特征分别计算特征值。

定义每种特征模板的特征值为白色区域的灰度像素值减去黑色区域的灰度像素值，即：

$$\sum i(x,y)_{white} - \sum i(x,y)_{black}$$

其中 $i(x,y)$ 是该区域内像素的灰度值。为了保证每种图片都能合理检测，在计算特征值之前还要对图像的灰度值进行归一化操作。

3.2. 积分图

本文选择的检测窗口大小为20X20，除去宽度为一个像素的四种特征外，还有 1816 个特征值，而检测窗口在图片上按一定步长滑动，如果直接按照原图片的像素值进行计算，产生的计算量非常巨大，所以引入了积分图的概念。通过先计算图片的积分图的值，再计算每种特征的特征值就只需要固定的时间，积分图的引入可以大大提高算法效率。

对于图像内的一点，定义该点的积分图为

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x',y')$$

其中 $i(x',y')$ 是该点在原图上的灰度值，积分图的含义是计算该点与原点围成矩形框内所有灰度

值的总和。

使用积分图可以方便的计算特征模板的特征值，如计算下图 D 区域的像素值总和，可以通过其四个顶点的积分图值来计算。

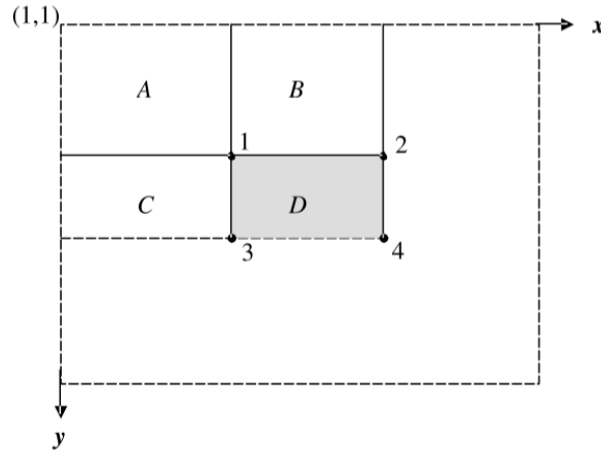


图 2 计算像素值的和

计算公式为 $ii_4 + ii_1 - ii_2 - ii_3$ ，因为 ii_4 ， ii_3 ， ii_1 ， ii_2 分别代表 $A+B+C+D$ ， $A+C$ ， A 和 $A+B$ 的像素值的和。解出即可得到任意区域的像素和值公式。要进一步计算出模板的特征值，只需要分别计算模板所在的各矩形区域的像素和值然后按需要进行加减即可。

由此可见，计算特征值只是常数次的加减运算，且图片大小确定后，所需要计算的特征值数量和计算量也随之确定，积分图的引入大大提高了检测速率。

3.3. 肤色特征

人脸和周围环境的明显不同一点就是肤色的不同，在图像中表现为 RGB 通道的值和周围环境有差异，可以以此为特征，构建一个监督学习的分类器，在用 Haar 特征进行检测之前，可以先对检测窗口执行颜色的检测，这样能在检测初期大量过滤掉非人脸的窗口提升检测效率。

对于一张 25×25 的待测图片，每一个像素点有 3 个 RGB 通道值，可以将全部 1875 个通道值作为特征放入神经网络中进行检测。

3.4. BP 神经网络

由感知机发展而来 BP 神经网络是一种按照误差逆向传播算法训练的多层前馈神经网络，是应用最广泛的神经网络模型之一。BP 网络内含多个感知器，由输入层、隐藏层、输出层组成。给定训练集 $D = \{(x_1, y_1)(x_2, y_2)(x_3, y_3) \dots (x_n, y_n)\}$ 其中 $x_n \in R^d$ 是一个 d 维向量 $y_n \in R^l$ 是一个 1 维向量^[4]，针对算法， d 为两种特征值的数量，分别为 1816 和 1875；1 为输出的结果，0（代表非人脸）和 1（代表人脸）是一个一维向量。BP 神经网络的模型如下图所示：

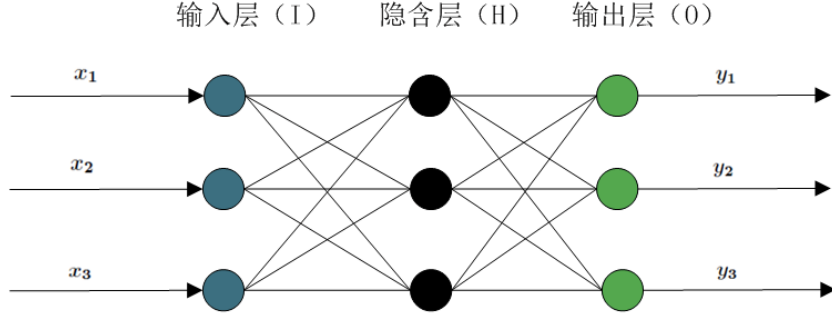


图 3 BP 神经网络模型

由感知机发展而隐含层的每个节点对所有的输入都赋予一个权值 w_i ，输入给该节点的数据即为每个输入的加权和，即

$$Net_{in} = \sum x_i w_i$$

每个神经元还有一个激活函数。最常用的激活函数是 Sigmoid 函数，它可将正无穷到负无穷之间的值转换到 0-1 的范围内。

Sigmoid 函数通常可以写成

$$f(\alpha) = \frac{1}{1 + e^{-\alpha}}$$

采用了激活函数的神经元的值可以写成

$$Net_{in} = \frac{1}{1 + e^{-\sum w_i x_i}}$$

输出层的输出可以写成

$$Out = f\left(\sum_{j=1}^H w_{jk} f\left(\sum_{i=1}^d w_{ij} x_i\right)\right)$$

其中 H 是隐含层节点的数量，f 是 Sigmoid 激活函数 w_{ij} 是输入层到隐含层的权值， w_{jk} 是隐含层到输出层的权值。

BP 神经网络具体的训练过程是：

- (1) 确定网络的结构，并初始化所有权值
- (2) 从训练集中得到一个训练样本 $x = [x_1, x_2, \dots, x_n]^T$ ，记其期望输出为 $D = [d_1, d_2, \dots, d_n]^T$
- (3) 使用公式 $Out = f\left(\sum_{j=1}^H w_{jk} f\left(\sum_{i=1}^d w_{ij} x_i\right)\right)$ 计算实际的输出。
- (4) 调整权值，使用公式

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$$

其中 $\Delta w_{ij}(t)$ 为权值修正项

$$\Delta w_{ij}(t) = \eta \delta_j x_i$$

η 为学习率，需要在训练之前给定， x_i 为这一次输入的输入值。

对输出层

$$\delta_j = y_j(1 - y_j)(d_j - y_j)$$

y_j 为当前的输出值

对中间层

$$\delta_j = x_j(1 - x_j) \sum \delta_k w_{jk}(t)$$

(5) 重复进行前四步，知道误差率小于设定值，或达到了训练次数上限。

BP 神经网络的关键在于选择正确的学习率和节点数，学习率一般要根据数据多次尝试后取最佳值，节点数可以由以下经验公式取得

$$\begin{aligned} l &< n - 1 \\ l &< \sqrt{m + n} + a \\ l &< \log_2(n) \end{aligned}$$

n 为输入层节点数； l 为隐藏层节点数； m 为输出层节点数； a 为 $0 \sim 10$ 之间的常数。

4. 实验设置

4.1. 数据集与图像预处理

本文使用了 MIT 的人脸数据库（在附件中以及文后附录链接），其中包含 20×20 灰度人脸 2706 张，非人脸 20×20 灰度图 4381 张，彩色人脸图 13233 张，彩色非人脸图 7267 张，其中彩色人脸图仅中间部分含有人脸，读取彩色人脸后需要对图像进行裁剪和重新调整大小，由于彩色人脸主要用于训练基于肤色特征的网络，所以调整大小后并不会明显改变，裁剪的 MATLAB 代码如下：

```
I=imread(name);%按图像名读取图像
I=imcrop(I,[85 85 84 84]);%裁剪图像中心人脸部分
I=imresize(I,[25 25]);%重新调整大小为 25*25
```

4.2. 基于肤色特征的 BP 神经网络

计划使用 8000 张彩色人脸图和 8000 张彩色非人脸图对神经网络进行训练。

首先依次读取人脸图和非人脸图并进行预处理，在循环中代码如下：

```
a=num2str(a);%a 为图像名称编号
name=['oface (' ,a, ').jpg'];%获得图像名称
I=imread(name);%读取图像
I=imcrop(I,[85 85 84 84]);%裁剪图像
I=imresize(I,[25 25]);%重新调整大小
```

然后对读取到的每张图像，得到它们的 RGB 的通道值，按顺序赋值给矩阵 e

```
[A,B,C]=size(I);
for a=1:A
    for b=1:B
        for c=1:C
```

```

%e1 用于保存人脸, e2 用于非人脸后面的 e 用于保存人脸和非人脸特征值的集合
e1(25*(a-1)+3*(b-1)+c,i-m1+1)=I(a,b,c);

end

end

end

```

获得 RGB 通道值后, 使用 MATLAB 构建 BP 神经网络, 设置好网络的参数开始对网络进行训练:

%网络输入的训练集, 所有像素点的 RGB 通道值

```
input_train = e(1:(m2-m1+1+n2-n1+1),:);
```

%输出的训练集, 对图片进行判断

```
output_train = [linspace(1,1,m2-m1+1) linspace(-1,-1,n2-n1+1)];
```

%建立网络模型, 传递函数使用 logsig(即 sigmoid 函数), 隐含层根据经验公式为 10 个节点

%采用梯度下降法训练

```
net=newff(input_train,output_train,10,{'logsig','logsig'},'trainlm');
```

%网络参数配置 (训练次数, 学习速率, 训练目标最小误差等)

```
net.trainParam.show=1; %每训练一次显示一次结果
```

```
net.trainParam.epochs=1000; % 训练次数为 1000 次
```

```
net.trainParam.lr=0.01; % 学习率, 这里设置为 0.01
```

```
net.trainParam.goal=0.00001; % 训练目标最小误差, 这里设置为 0.00001
```

```
net.trainParam.max_fail=100; %训练错误自检 100 次 (即误差不再下降)
```

```
net=train(net,input_train,output_train); %开始训练
```

该网络训练用时 5 小时左右, 训练完成后获得肤色特征的识别网络 net, 将其重命名为 rgbnet 后保存在文件 rgbnet.mat 中。

4. 3. 基于 Haar 特征的 BP 神经网络

计划使用 2000 张灰度人脸图和 2000 张灰度非人脸图作为训练集。



图 4 训练使用的人脸和非人脸图

首先是要对每张图片的 Haar 特征值进行计算, 每张 20*20 的图片有 1816 个特征值, 计算特征值的 MATLAB 代码如下:

```
k=1;
```

%所有边缘特征 (1) 的特征值

```
w=2;h=4;e=1;
```

%判断矩形框是否大于图片大小

while(2^e<20)

for i=1:20

for j=1:20

%判断特征矩形框是否还在图片内

if((i+w<=20)&&(j+h<=20))

white=SI(j+h/2,i+w)+SI(j,i)-SI(j,i+w)-SI(j+h/2,i);%白色区域像素和值

black=SI(j+h,i+w)+SI(j+h/2,i)-SI(j+h/2,i+w)-SI(j+h,i); %黑色区域像素和值

eigenvalue(1,k)=white-black;%计算特征值并存入矩阵

k=k+1;%k 用于对每个特征值编号，计算完一个特征值后 k 自加 1

end

end

end

e=e+1;w=w*2;h=h*2;%相同大小特征值计算完成后，计算下一大小矩形框的特征值

end

对于其他特征，按照相似的原则和代码即可获得对应的特征值，将计算单张图片特征值的代码写为函数 *GetEigenvalue*，按图片数量批量获得特征值的函数写为 *BatchGetEigenvalue*，该函数返回所有输入图片的特征值。

获得特征值后对 Haar 特征构建 BP 神经网络并进行训练，这部分代码在函数 *TrainNet* 中，网络的操作和部分参数如下：

input_train = e';%输入获得的特征值作为训练集

output_train = [linspace(1,1,m2-m1+1) linspace(-1,-1,n2-n1+1)];%图片的分类作为训练集的输出

%建立网络模型，传递函数使用 logsig(即 sigmoid 函数)，隐含层根据经验公式为 10 个节点

%采用梯度下降法训练

net=newff(input_train,output_train,10,{'logsig','logsig'},'trainlm');

net.trainParam.show=1; %每训练一次显示一次结果

net.trainParam.epochs=1000; % 训练次数为 1000 次

net.trainParam.lr=0.01; % 学习率，这里设置为 0.01

net.trainParam.goal=0.00001; % 训练目标最小误差，这里设置为 0.00001

net.trainParam.max_fail=50; %训练错误自检 50 次（即误差不再下降）

net=train(net,input_train,output_train); %开始训练

训练完成后函数返回一个基于 Haar 特征额 BP 神经网络，将其保存在 facenet.mat 文件中。至此实验的设置工作全部完成，下面对实验结果即该算法的效果进行研究。

5. 实验结果

5.1. 测试结果

使用 *faces_test* 文件夹内的图片检验该算法是否能正确识别人脸，具体检测方法是使用固定大小的滑动窗口，设置一定的步长，对图片逐像素进行检测，若出现人脸则返回当前的像素坐标值和窗口大小，最后合并重叠的窗口，并在原图片中用矩形框画出人脸范围。此部分代码在 *Recognize.m* 中图片的加载设置如下：

```
name='faces_test(1).jpg';%待测图片名
load('rgbnet.mat');%加载神经网络
load('facenet.mat');
I=imread(name);%读取图片
[H,W,~]=size(I);%获得图片的大小
SIZE=[H W];
length=260;%设置滑动窗口大小
```

使用基于肤色特征的 BP 网络进行检测如下：

```
WI=imcrop(I,[j i length length]);%裁剪滑动窗口，ij 为当前滑动窗口的左上角坐标值
if isempty(WI)%判断该窗口是否为空
    continue;
end
WI=imresize(WI,[25 25]);%重新调整窗口大小，使其能够匹配神经网络的检测大小
[A,B,C]=size(WI);
e=[];
for a=1:A
    for b=1:B
        for c=1:C
            e(25*(a-1)+3*(b-1)+c,1)=WI(a,b,c);%计算 RGB 通道值
        end
    end
end
anl=sim(rgbnet,e);%计算识别结果
```

使用基于 Haar 特征的 BP 网络进行检测如下：

```
if(abs(anl-1)<0.1)%当肤色特征符合条件时才能进行这一步检测
    %窗口大小重新调整为计算特征值的大小，重新调整大小并不会明显改变人脸特征，却可以有效减少计算量
```

```
    WI=imresize(WI,[20 20]);
    %转化为灰度图并归一化
```



```

WI=rgb2gray(WI);
WI=mat2gray(WI);
%计算积分图
WI=GetSigma(WI);
%计算特征值
eigenvalue=GetEigenvalue(WI)';
%识别, 并判断识别结果, 通过识别的窗口坐标将会保存到s 矩阵内
an=sim(net,eigenvalue);
if(an>0)
    s=[s;j,i,length,an];
end

```

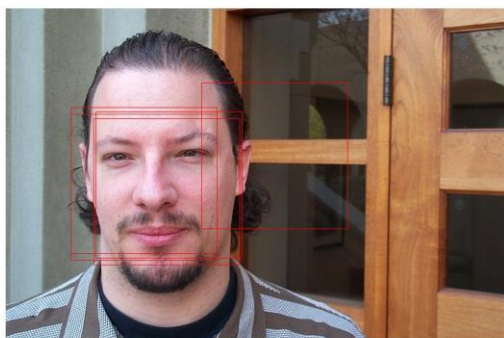
最后合并 s 矩阵内重叠的矩形框, 在原图上画出剩余的矩形框即为识别结果, 一些测试集 (MIT 数据集) 内的识别结果如下 (其余测试结果在压缩包内):



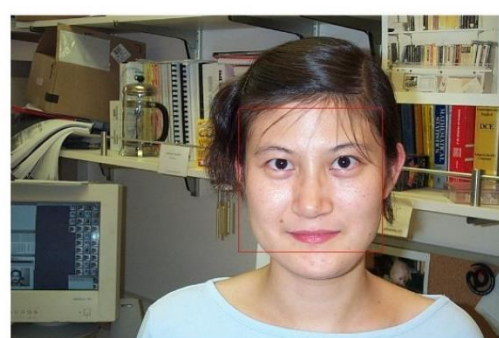
错检: 0 漏检: 0 正确: 1



错检: 0 漏检: 0 正确: 1



错检: 1 漏检: 0 正确: 1



错检: 0 漏检: 0 正确: 1

图 5 部分测试结果

5. 2. 结论

使用这种算法可以对图片中的人脸进行有效的识别, 并且检测速度较快, 该算法稍加改进进一步提高速度后还能有广阔的应用空间。

6. 附录与参考文献

6.1. 核心代码

```

1.RGB 通道特征提取
%读取图片并调整大小
I=imread(name);
I=imcrop(I,[85 85 84 84]);
I=imresize(I,[25 25]);
%按 RGB 的顺序依次读取通道值存入 e1 矩
阵内
[A,B,C]=size(I);
for a=1:A
for b=1:B
for c=1:C
e1(25*(a-1)+3*(b-1)+c,i-m1+1)=I(a,b,c);
end
end
end

2.基于肤色特征 BP 神经网络的构建
%设置训练集的输入输出,是人脸输出 1,非
人脸输出-1
input_train = e(1:(m2-m1+1+n2-n1+1),:);
output_train = [linspace(1,1,m2-m1+1)
linspace(-1,-1,n2-n1+1)];
%神经网络参数设置,激活函数 sigmoid,采
用梯度下降法训练
net=newff(input_train,output_train,10,{'logsig
','logsig'},'trainlm');
%其他训练参数设置训练 1000 次,每次显示
结果,学习率 0.01,误差 0.00001,错误自检 100
次
net.trainParam.show=1;
net.trainParam.epochs=1000;
net.trainParam.lr=0.01;
net.trainParam.goal=0.00001;
net.trainParam.max_fail=100;

% 开始训练
net=train(net,input_train,output_train);

3.计算积分图
%获得图片大小
[X,Y]=size(I);
%初始化积分图矩阵
SI=zeros(X,Y);
for i=1:X
for j=1:Y
for x=1:i
for y=1:j
%按像素依次计算积分图
SI(i,j)=SI(i,j)+I(x,y);
end
end
end
end

4.Haar 特征提取,以边缘特征(1)为例
%设置矩形框初始大小
w=2;h=4;e=1;
%终止条件为矩形框大于图像大小
while(2^e<20)
%矩形框依次遍历图片
for i=1:20
for j=1:20
%判断矩形框是否超出图片范围
if((i+w<=20)&&(j+h<=20))
%用积分图计算各区域像素和值
white=SI(j+h/2,i+w)+SI(j,i)-SI(j,i+w)-
SI(j+h/2,i);
black=SI(j+h,i+w)+SI(j+h/2,i)-SI(j+h/2,i+w)-
SI(j+h,i);
%计算特征值并保存

```

```

eigenvalue(1,k)=white-black;
k=k+1;
end
end
end
% 下一次遍历开始前将矩形框放大
e=e+1;w=w*2;h=h*2;
end

```

5. 基于 Haar 特征的 BP 神经网络构建

%设置训练集的输入输出, 是人脸输出 1, 非人脸输出-1

```

input_train = e';
output_train = [linspace(1,1,m2-m1+1)
linspace(-1,-1,n2-n1+1)];

```

%设置训练集的输入输出, 是人脸输出 1, 非人脸输出-1

```

net=newff(input_train,output_train,10,{'logsig', 'logsig'}, 'trainlm');

```

%其他训练参数设置训练 1000 次, 每次显示结果, 学习率 0.01, 误差 0.00001, 错误自检 50 次

```

net.trainParam.show=1;
net.trainParam.epochs=1000;
net.trainParam.lr=0.01;
net.trainParam.goal=0.0001;
net.trainParam.max_fail=50;
% 开始训练
net=train(net,input_train,output_train);

```

6. 使用滑动窗口识别人脸

%读取图片并获得图片大小

```

I=imread(name);
[H,W,~]=size(I);
SIZE=[H W];

```

%设置窗口初始大小, 初始化矩阵 s,s 用于保存检测到人脸的窗口数据

```

length=260;
s=[0,0,0,0];
%终止条件为窗口大于图片大小

```

```

while(length<=0.5*min(min(SIZE)))
%步长为 4 个像素, 移动滑动窗口进行检测
for i=1:4:H
for j=1:4:W
%判断窗口是否超过图片大小
if((i+length<H)&&(j+length<W))
%按肤色特征网络的要求裁剪图片并调整大

```

小

```

WI=imcrop(I,[j i length length]);
if isempty(WI)
continue;
end
WI=imresize(WI,[25 25]);
%计算肤色特征值
[A,B,C]=size(WI);
e=[];
for a=1:A
for b=1:B
for c=1:C
e(25*(a-1)+3*(b-1)+c,1)=WI(a,b,c);
end
end
end
%使用训练好的网络判断结果
an1=sim(rgbnet,e);
%判断肤色特征是否符合要求, 符合要求计

```

算 Haar 特征

```

if(abs(an1-1)<0.1)
%重新调整大小为 Haar 特征的要求
WI=imresize(WI,[20 20]);
%转化为灰度图并归一化
WI=rgb2gray(WI);
WI=mat2gray(WI);
%计算积分图
WI=GetSigma(WI);
%计算特征值
eigenvalue=GetEigenvalue(WI)';
%带入 Haar 特征的网络并判断是否符合要求
an=sim(net,eigenvalue);

```

求

<i>if(an>0)</i>	<i>end</i>
<i>%符合要求的窗口保存进 s 矩阵</i>	<i>end</i>
<i>s=[s;j,i,length,an];</i>	<i>end</i>
<i>end</i>	<i>%窗口大小提高 1.5 倍进行下一轮遍历检测</i>
<i>end</i>	<i>length=round(length*1.5);</i>
<i>else</i>	<i>end</i>
<i>continue;</i>	

6. 2. 参考文献

- [1]李武军, 王崇骏, 张炜, 陈世福. 人脸识别研究综述[J]. 模式识别与人工智能, 2006, 19(01):58-66.
- [2]梁路宏 , 艾海舟 , 徐光祐 , 张钊. 人脸检测研究综述[J]. 计算机学报, 2002(05):449-458.
- [3]赵楠 . 基于 AdaBoost 算法的人脸检测[D]. 2005(06)
- [4]fanxin_i. 详解 BP 神经网络[EB/OL]. (2018-05-06). blog.csdn.net/fanxin_i/article/details/80212906