第 4 讲　图像的增广

有一组作为数据集的图片，数量较少，需要对其进行增广扩充训练数据。比如图像剪裁、图像的翻转、图像直方图均衡化、增加亮度、增加高斯噪声等。

4.1 图像颜色变换

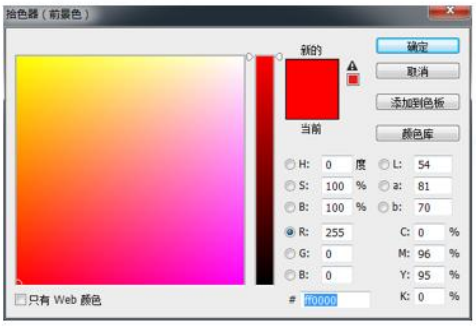
4.1.1 图像色彩空间模型

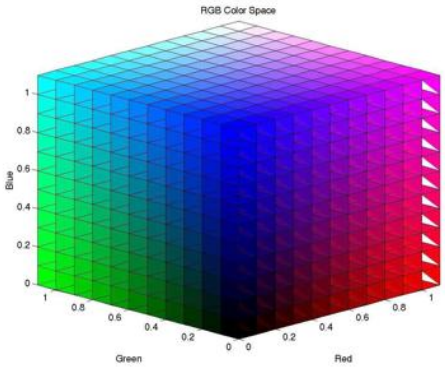
（1）RGB 空间模型

任意色彩都是用 R、G、B 三色不同分量的相加混合而成：F=r[R]+r[G]+r[B]。RGB 色彩空间还可以用一个三维的立方体来描述。

当三基色分量都为 0(最弱)时混合为黑色光；

当三基色都为 k(最大，值由存储空间决定)时混合为白色光。





Opencv 读入的图像文件，B 通道、G 通道、R 通道

```
import cv2
img = cv2.imread("opencv.jpg")      #bgr
B=img[:,:,0]
G=img[:,:,1]
R=img[:,:,2]
cv2.imshow('opencv',img)
cv2.imshow('B',B)
cv2.imshow('G',G)
cv2.imshow('R',R)
img[:,:,0]=0
cv2.imshow('opencvB0',img)
img[:,:,1]=0
cv2.imshow('opencvB0G0',img)
cv2.waitKey()
cv2.destroyAllWindows()
```

（2）灰度色彩空间

256 个灰度级别，图像由 RGB 钻护卫灰度色彩时，GRAY=0.299R+0.587G+0.114B

（3）HSV 颜色空间模型

色相、饱和度、明度。

色相就是平时说的颜色，如红色、黄色

饱和度是指色彩的纯度，其值越高色彩越纯，越低则逐渐变灰，取值范围 1%-100%。

4.1.2 颜色空间变换

（1）颜色空间的转换 BGR->RGB

cv2.cvtColor(p1,p2) 是颜色空间转换函数，p1 是需要转换的图片，p2 是转换成何种格式。

cv2.COLOR_BGR2RGB 将 BGR 格式转换成 RGB 格式

cv2.COLOR_BGR2GRAY 将 BGR 格式转换成灰度图片

```
import cv2
import matplotlib.pyplot as plt
img_BGR = cv2.imread("opencv.jpg")
img_RGB=cv2.cvtColor(img_BGR,cv2.COLOR_BGR2RGB)
img_GRAY=cv2.cvtColor(img_BGR,cv2.COLOR_BGR2GRAY)
plt.subplot(121),plt.imshow(img_RGB),plt.title('RGB')
plt.subplot(122),plt.imshow(img_GRAY),plt.title('GRAY')
plt.show()
```

4.2 图像直方图处理

4.2.1 直方图的绘制

1.灰度直方图

直方图就是对图像中每个像素值个数统计。如灰度图像中灰度值为 0 的像素点多少个，100 的像素点有多少个等等。

直方图的 x 轴是灰度值（0-255），y 轴是图片中具有同一个灰度值的像素点的个数。通过直方图对图像的对比度、亮度和灰度分布有一个直观的认识。

```
cv2.calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate ]])
```

- imaes:输入的图像
- channels:选择图像的通道
- mask:掩膜，是一个大小和 image 一样的 np 数组，其中把需要处理的部分指定为 1，不需要处理的部分指定为 0，一般设置为 None，表示处理整幅图像
- histSize:使用多少个 bin(柱子)，一般为 256
- ranges:像素值的范围，一般为[0,255]表示 0~255

后面两个参数基本不用管，注意，除了 mask，其他四个参数都要带[]号。

np.histogram(img.ravel(),bins,ranges)

- img.ravel():图像的像素矩阵
- bins：统计的区间个数
- ranges：要计算的像素值范围，[0,255]

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
img = cv2.imread('./testImages/opencv.jpg')
imgGray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
hist1=cv2.calcHist([imgGray],[0],None,[256],[0,256])
hist2,bins=np.histogram(imgGray.ravel(),256,[0,256])
plt.subplot(121),plt.plot(hist1),plt.title('hist1')
plt.subplot(122),plt.plot(hist2),plt.title('hist2')
plt.show()
```

2.彩色直方图

彩色图像三个通道，分别进行绘制，每个通道上像素的分布，得到原图中哪种颜色份量比较多。

```
import cv2
import matplotlib.pyplot as plt
plt.figure(figsize=(10,10))
img = cv2.imread('./testImages/cat.jpg')
hist0=cv2.calcHist([img],[0],None,[256],[0,256])
plt.plot(hist0,color='b')
hist1=cv2.calcHist([img],[1],None,[256],[0,256])
plt.plot(hist1,color='g')
hist2=cv2.calcHist([img],[2],None,[256],[0,256])
plt.plot(hist2,color='r')
plt.show()
```

4.2.2 直方图均衡化

直方图均衡化是改变图像的像素灰度分布，从而达到对图像进行增强的目的。

1.灰度图像均衡化

cv2.equalizeHist(img)

将要均衡化的原图像**【要求是灰度图像】**作为参数传入，则返回值即为均衡化后的图像。

```
plt.figure(figsize=(10,10))
img = cv2.imread('./data/night.jpg')
imgGray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
dst=cv2.equalizeHist(imgGray)
plt.subplot(121),plt.imshow(img),plt.title('orginal')
plt.subplot(122),plt.imshow(imgGray),plt.title('equalizeHist')
plt.show()
```

请绘制直方图的变化

```
hist1=cv2.calcHist([img],[0],None,[256],[0,256])
hist2,bins=no.histogram(imgGray.ravel(),256,[0,256])
plt.subplot(121),plt.imshow(hist1),plt.title('hist1')
plt.subplot(122),plt.imshow(hist2),plt.title('hist2')
plt.show()
```

2.彩色图像均衡化

需要先用 **split（）将彩色图像的 3 个通道拆分，然后分别进行均衡化。**

```
plt.figure(figsize=(10,10))
img = cv2.imread('./data/night.jpg')
(b,g,r)=cv2.split(img)
bcolor=cv2.equalizeHist(b)
gcolor=cv2.equalizeHist(g)
rcolor=cv2.equalizeHist(r)
result=cv2.merge((bcolor,gcolor,rcolor))
ImgRGB=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
ResutlRGB=cv2.cvtColor(result,cv2.COLOR_BGR2RGB)
plt.subplot(121),plt.imshow(imgRGB),plt.title('ORGINAL')
plt.subplot(122),plt.imshow(ResutlRGB),plt.title('EQUALIZEHist')
plt.show()
```

4.3 Opencv 图像的几何变换

4.3.1 图像缩小或放大

改变图像的大小，使用 cv2.resize()函数，图像的大小可以手动指定，也可以使用缩放比例，其函数形式如下所示：

```
cv2.resize(src,dsize[,fx[,fy[,interpolation]]])
```

src：原图像
dsize：输出图像大小（width，height）
fx：水平轴上的缩放比例 1.2
fy：垂直轴上的缩放比例 1.4
interpolation：插值方法　20*20　40*40

插值方法	含义
INTER_NEAREST	最近邻插值
INTER_LINEAR	双线性插值（默认设置）
INTER_AREA	使用像素区域关系进行重采样
INTER_CUBIC	4x4 像素邻域的双立方插值
INTER_LANCZOS4	8x8 像素邻域的 Lanczos 插值

#将图像放大或者缩小

```
import cv2
import matplotlib.pyplot as plt
img = cv2.imread('./data/opencv.jpg')
res1 = cv2.resize(img, (100, 100))#变成 100*100 像素图像
res2 = cv2.resize(img, None, fx=2, fy=2, interpolation=cv2.INTER_LI
NEAR)#放大两倍
img0=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
img1=cv2.cvtColor(res1,cv2.COLOR_BGR2RGB)
img2=cv2.cvtColor(res2,cv2.COLOR_BGR2RGB)
plt.subplot(131),plt.imshow(img0),plt.title('img0')
plt.subplot(132),plt.imshow(img1),plt.title('img1')
plt.subplot(133),plt.imshow(img2),plt.title('img2')
plt.show()
```

将当前工作目录下的./data 文件夹下所有的图片进行裁剪，统一尺寸为 200*200 像素，并将裁剪好的图片放到当前工作路径的./resizeData 目录下，图像名称为 resize001.jpg，resize002.jpg...。

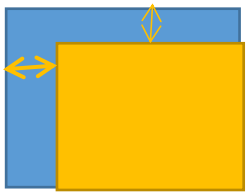
4.3.2 图像的仿射变换

通过仿射变换函数 cv2.warpAffine()可实现图像的旋转、平移、缩放，变换后的平行线依旧平行。

1.图像平移

将图像沿着 x、y 轴移动指定的像素，需要构建平移矩阵： t_x 为 x 的偏移量， t_y 是 y 轴的偏移量，单位为像素。

$$M = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix}$$



cv2.warpAffine(src,M,dsize）#将图像平移到指定的位置

src：图像矩阵

M：图像的变换矩阵

dsize：输出后的图像大小

例如：对图像进行平移，例如读取一幅图像，将图像分别沿 x 轴移动 100 个像素，沿 y 轴移动 50 个像素。代码如下：

```
import  cv2
import matplotlib.pyplot as plt
import numpy as np
img = cv2.imread('./data/opencv.jpg')
rows,cols,channel = img.shape
print(img.shape)
M = np.float32([[1,0,100],[0,1,50]])
dst = cv2.warpAffine(img,M,(cols,rows))
img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
dst=cv2.cvtColor(dst,cv2.COLOR_BGR2RGB)
plt.subplot(121),plt.imshow(img),plt.title('img')
plt.subplot(122),plt.imshow(dst),plt.title('dst')
plt.show()
```

2.图像旋转

图像旋转是以图像的中心为原点，旋转一定的角度，将图像上的所有像素都旋转一个相同的角度。旋转后图像的大小一般会改变，因为要把转出显示区域的图像截去，或者扩大图像范围来显示所有的图像。

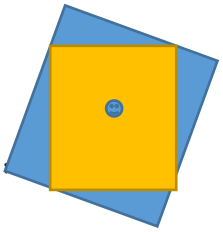
opencv 提供了一个获取变换矩阵的函数 cv2.getRotationMatrix2D,获取后再通过 cv2.warpAffine 进行变换。

cv2.getRotationMatrix2D（center，angle，scale）#对图像进行旋转操作

center：图像旋转的中心点。

angle：旋转的角度，+逆时针方向，-顺时针。

scale：图像缩放比例。



例如，将图像按照逆时针方向旋转 30 度,并缩小到 80%。实例代码如下：

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
img = cv2.imread('./data/opencv.jpg')
h,w,c = img.shape
M = cv2.getRotationMatrix2D((w / 2, h / 2), 30, 0.8)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.figure(figsize=(10,10))
img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
dst=cv2.cvtColor(dst,cv2.COLOR_BGR2RGB)
plt.subplot(121),plt.imshow(img),plt.title('img')
plt.subplot(122),plt.imshow(dst),plt.title('dst')
plt.show()
```

在当前工作路径下的./data 目录下有若干张热狗的图片，因为图片较少，需要进行增广。编写 python 代码，10 度内的随机旋转（random.randint(-10,10)），可以将图像数目增加一倍。旋转后的图像保存到./rotated 下，图像命名为“rotated_”，重命名为 rotated_001.jpg,rotated_002.jpg。

3.扭曲变换

扭曲变换是两种简单变换的叠加，线性变换和平移变换。

通过函数 cv2.getAffineTransform()获取变换矩阵，再通过 cv2.warpAffine()函数进行变换。

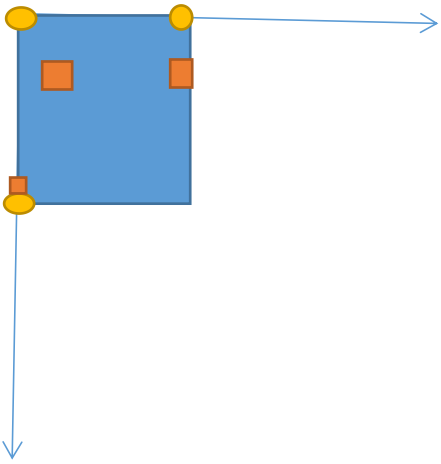
cv2.getAffineTransform(src，dst)

src：源图像的三个坐标点

dst：仿射后图像的三个坐标点

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
```

```
img = cv2.imread('./data/opencv.jpg')
h,w,c = img.shape
m1=np.float32([[0,0],[256,0],[0,256]])
m2=np.float32([[50,100],[256,50],[0,256]])
M = cv2.getAffineTransform(m1,m2)
dst = cv2.warpAffine(img,M,(cols,rows))
plt.figure(figsize=(10,10))
img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
dst=cv2.cvtColor(dst,cv2.COLOR_BGR2RGB)
plt.subplot(121),plt.imshow(img),plt.title('img')
plt.subplot(122),plt.imshow(dst),plt.title('dst')
plt.show()
```



4.图像翻转

对图像进行水平或者垂直方向上的翻转，可以使用如下函数。

```
cv2.flip(src,flipCode[,dst])#对图像进行翻转操作
```

src:原始图像

flipcode: 1 水平翻转　　0 垂直翻转　　-1 水平垂直翻转

对一张图进行各个方向的翻转，实例代码如下：

```
img = cv2.imread('./data/opencv.jpg')
h = cv2.flip(img, 1)
v = cv2.flip(img, 0)
hv = cv2.flip(img, -1)
plt.figure(figsize=(10,10))
h=cv2.cvtColor(h,cv2.COLOR_BGR2RGB)
v=cv2.cvtColor(v,cv2.COLOR_BGR2RGB)
hv=cv2.cvtColor(hv,cv2.COLOR_BGR2RGB)
plt.subplot(221),plt.imshow(img),plt.title('img')
plt.subplot(222),plt.imshow(h),plt.title('h')
plt.subplot(223),plt.imshow(v),plt.title('v')
plt.subplot(224),plt.imshow(hv),plt.title('hv')
plt.show()
```

在当前工作路径下的./data 目录下有若干张狗的图片，因为图片较少，需要进行翻转。编写 python 代码，水平、垂直以及两个方向的随机翻转（random.randint(-1,1)），将图像增加一倍。翻转后的图像保存到./flipped 下，图像命名为“filpped_”，重命名为filpped_001.jpg,filpped_002.jpg。

4.4 图像的算数运算

4.4.1 加法运算

cv2.add 实现两幅图像进行相加运算。要求两幅图像必须大小一致、类型相同。opencv 对超过 255 的数值按照 255 处理，Numpy 则会取模处理。

add(src1, src2, dst=None, mask=None, dtype=None)

- src1, src2：需要相加的两副大小和通道数相等的图像或一副图像和一个标量（标量即单一的数值）
- dst：可选参数，输出结果保存的变量，默认值为 None
- mask：图像掩膜，可选参数，为 8 位单通道的灰度图像，用于指定要更改的输出图像数组的元素
- dtype：可选参数，输出图像数组的深度，即图像单个像素值的位数（如 RGB 用三个字节表示，则为 24 位）。

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
img = cv2.imread('./data/dog5.jpg')
M1=np.ones(img.shape, dtype='unit8')*100  #生成与图像一样大小的全 100 的矩阵
imgAdded = cv2.add(img,M1)
plt.figure(figsize=(10,10))
img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
imgAdded=cv2.cvtColor(imgAdded ,cv2.COLOR_BGR2RGB)
plt.subplot(121),plt.imshow(img),plt.title('img')
plt.subplot(122),plt.imshow(imgAdded ),plt.title('imgAdded ')
plt.show()
```

4.4.2 减法运算

cv2.subtract 实现两幅图像进行相减运算。要求两幅图像必须大小一致、类型相同。

cv2.subtract(src1, src2, dst=None, mask=None, dtype=None)

- src1, src2：需要相加的两副大小和通道数相等的图像或一副图像和一个标量（标量即单一的数值）
- dst：可选参数，输出结果保存的变量，默认值为 None
- mask：图像掩膜，可选参数，为 8 位单通道的灰度图像，用于指定要更改的输出图像数组的元素
- dtype：可选参数，输出图像数组的深度，即图像单个像素值的位数（如 RGB 用三个字节表示，则为 24 位）。

```
import cv2
```

```
import matplotlib.pyplot as plt
import numpy as np
img = cv2.imread('./data/dog5.jpg')
M1=np.ones(img.shape, dtype='unit8')*50  #生成与图像一样大小的全 100 的矩阵
imgSubtracted = cv2.subtract(img,M1)
plt.figure(figsize=(10,10))
img=cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
imgSubtracted=cv2.cvtColor(imgSubtracted ,cv2.COLOR_BGR2RGB)
plt.subplot(121),plt.imshow(img),plt.title('img')
plt.subplot(122),plt.imshow(imgSubtracted),plt.title('imgSubtracted ')
plt.show()
```

4.4.3 图像混合

图像混合函数 cv2.addWeighted 也是图像相加的操作。不过两幅图像权重不一样。

```
cv2.addWeighted(src1, alpha, src2, beta, gamma, dst=None, dtype=None)
```

- src1, src2：需要融合相加的两副大小和通道数相等的图像
- alpha：src1 的权重
- beta：src2 的权重
- gamma：gamma 修正系数，不需要修正设置为 0
- dst：可选参数，输出结果保存的变量，默认值为 None
- dtype：可选参数，输出图像数组的深度，即图像单个像素值的位数（如 RGB 用三个字节表示，则为 24 位），选默认值 None 表示与源图像保持一致。
- 返回值：融合相加的结果图像

```
import cv2
import matplotlib.pyplot as plt
import numpy as np
img1 = cv2.imread('./data/opencv.jpg')
img1=cv2.resize(img1,(200,200))
img2 = cv2.imread('./data/mushroom.jpg')
img2=cv2.resize(img2,(200,200))
dst=cv2.addWeighted(img1,0.5,img2,0.5,0)

img1=cv2.cvtColor(img1,cv2.COLOR_BGR2RGB)
img2=cv2.cvtColor(img2 ,cv2.COLOR_BGR2RGB)
dst=cv2.cvtColor(dst,cv2.COLOR_BGR2RGB)
plt.subplot(131),plt.imshow(img1),plt.title('img')
```

```
plt.subplot(132),plt.imshow(img2),plt.title('img2')
plt.subplot(133),plt.imshow(dst),plt.title('dst')
plt.show()
```

在当前工作路径下的./data 目录下有若干张狗的图片，通过增加图像的亮度来增广图像。请为图像随机增加亮度（将原图转换成灰度图像，随机增加亮度后，转换成彩色图像，与源图像相加）./bright 下，图像命名为“bright_”,重命名为 bright_001.jpg,bright_002.jpg。