ABBYY® Real-Time Recognition SDK®
Free OCR SDK for iOS and Android
LEARN MORE

How to check whether a jpeg image is color or gray scale using only Python stdlib

Ask Question

I have to write a test case in python to check whether a jpg image is in color or grayscale. Can anyone please let me know if there is any way to do it with out installing extra libraries like opencv?

python image-processing python-2.6

edited May 14 '14 at 17:38
smci
14k 6 67 102

asked May 14 '14 at 17:06
kadina
1,327 1 11 28

Questions: a) What libraries are not considered extra libraries? NumPy/Scipy? b) Do you want to simply detect 2 vs 3 channels and use this as your grayscale criteria or will you have 3 channel images that are actually grayscale in appearance? – YXD May 14 '14 at

Home

Users

Jobs

Teams
Q&A for work

Learn More

some 3 channel images that are actually grayscale in appearance. – kadina May 14 '14 at 17:15

1 Do you have *any* way of opening an image as pixels? If not this is going to be a hard problem. – Mark Ransom May 14 '14 at 17:18

@Mark Ransom: you mean you can't just trust the JPEG header, offset 6: number of components (1 = grayscale, 3 = RGB) ? – smci May 14 '14 at 17:40

@smci I guess grayscale JPEGs are so rare that I didn't remember it was possible. There will also be cases where a grayscale image is saved with 3 components. – Mark Ransom May 14 '14 at 19:03

6 Answers

Expanding @gat answer:

```
import Image
def is_grey_scale(img_path="lena.jpg"):
    im = Image.open(img_path).convert('RGB')
    w,h = im.size
    for i in range(w):
        for j in range(h):
            r,g,b = im.getpixel((i,j))
            if r != g != b: return False
    return True
```

Basically check every pixel to check if it is grayscale (R == G == B)

edited May 15 '14 at 16:38

answered May 14 '14 at 17:33

have black or white border, you'd expect faster termination by sampling random i,j-points from `im` and test them? Or use modulo arithmetic to traverse the image. If sampling(-without-replacement) say 100 random i,j-points isn't conclusive, then just scan it linearly. Or maybe vary the row order with modulo arithmetic. You could wrap all this in a custom iterator `iter_pixels(im)` . – smci May 14 '14 at 17:44

Sorry. The code is failing when I tried to run the script and it is giving the error @ r,g,b = im.getpixel((i,j)) TypeError: 'int' object is not iterable – kadina May 14 '14 at 21:28

1 Need to add rgb_im = im.convert('RGB') – kadina May 14 '14 at 22:08

1 @kadina if it opens and isn't RGB then you already have your answer - I believe the only other possibility is grayscale. At least for a JPEG. – Mark Ransom May 15 '14 at 21:45

Can be done as follow:

```
from scipy.misc import imread, imsave, imresize
image = imread(f_name)
if(len(image.shape)<3):
      print 'gray'
elif len(image.shape)==3:
      print 'Color(RGB)'
else:
      print 'others'
```

Effective answer. Whether an image is RGB or gray can be determined by its size. – Ahmet Tavli Aug 16 at 12:29

A performance-enhance for fast results: since many images have black or white border, you'd expect faster termination by sampling a few random i,j-points from im and test them? Or use modulo arithmetic to traverse the image rows. First we sample(-without-replacement) say 100 random i,j-points; in the unlikely event that isn't conclusive, then we scan it linearly.

Using a custom iterator iterpixels(im). I don't have PIL installed so I can't test this, here's the outline:

```
import Image

def isColor(r,g,b): # use tuple-unpacking to unpack pixel -> r,g,
    return (r != g != b)

class Image_(Image):
    def __init__(pathname):
        self.im = Image.open(pathname)
        self.w, self.h = self.im.size
    def iterpixels(nrand=100, randseed=None):
        if randseed:
            random.seed(randseed) # For deterministic behavior i
        # First, generate a few random pixels from entire image
        for randpix in random.choice(im, n_rand)
            yield randpix
        # Now traverse entire image (yes we will unwantedly revi
once)
        #for pixel in im.getpixel(...): # you could traverse row
(say) (im.height * 2./3) -1
        #    yield pixel

    def is_grey_scale(img_path="lena.jpg"):
        im = Image_.(img_path)
```

RGB). If it's 1=grayscale, you know the answer already without needing to inspect individual pixels.)

edited May 17 '14 at 18:50

answered May 14 '14 at 18:12

smci

14k 6 67 102

For faster processing, it is better to avoid loops on every pixel, using ImageChops, (but also to be sure that the image is truly grayscale, we need to compare colors on every pixel and cannot just use the sum):

```
from PIL import Image,ImageChops

def is_greyscale(im):
    """
    Check if image is monochrome (1 channel or 3 identical chann
    """
    if im.mode not in ("L", "RGB"):
        raise ValueError("Unsuported image mode")

    if im.mode == "RGB":
        rgb = im.split()
        if ImageChops.difference(rgb[0],rgb[1]).getextrema()[1]!
            return False
        if ImageChops.difference(rgb[0],rgb[2]).getextrema()[1]!
            return False
    return True
```

edited Dec 9 '15 at 9:59

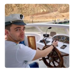
Why wouldn't we use ImageStat module?

```
from PIL import Image, ImageStat

def is_grayscale(path="image.jpg")

    im = Image.open(path).convert("RGB")
    stat = ImageStat.Stat(im)

    if sum(stat.sum)/3 == stat.sum[0]:
        return True
    else:
        return False
```

stat.sum gives us a sum of all pixels in list view = [R, G, B] for example [568283302.0, 565746890.0, 559724236.0]. For grayscale image all elements of list are equal.

edited Sep 24 '14 at 17:59

answered Sep 24 '14 at 17:47


GriMel
742 1 8 26

4 an image composed of an equal number of pure red, pure green and pure blue pixels would wrongly identify as greyscale – scruss Jan 12 '17 at 18:14

```
import Image
im = Image.open("lena.jpg")
```

EDIT As pointed out by Mark and JRicardo000, you may iterate over each pixel. You could also make use of the im.split() function here.

edited May 14 '14 at 17:35

answered May 14 '14 at 17:15

gat
1,880 13 20

3 The mode is always going to be RGB from a JPEG. You need to actually examine the pixels. – Mark Ransom May 14 '14 at 17:18

Yes, you are right. Let me fix that in a moment. – gat May 14 '14 at 17:19