What libraries can load image in Python and what are their difference?

Summarization & Comparison of .imread()

Kevin Luk Follow
Mar 30 · 3 min read ★

When we face computer vision project, first of all we need to load the images before any preprocessing.

There are various libraries out there to perform `imread()` . Here I want to consolidate the popular libraries for loading image and their difference. This article will go through:

1. Libraries for loading image
2. Colour channel
3. Efficiency
4. **Cheatsheet!**

Library for loading image

There are four libraries that are usually used for loading images.

- Matplotlib— `plt.imread()`
- OpenCV— `cv2.imread()`
- Pillow— `Image.open()`
- scikit-image— `io.imread()`


```
import matplotlib.pyplot as plt

img = plt.imread(img_dir)
```

. . .


```
import cv2

img = cv2.imread(img_dir)
```

. . .


```
from PIL import Image
```

```
img = Image.open(img_dir)
```

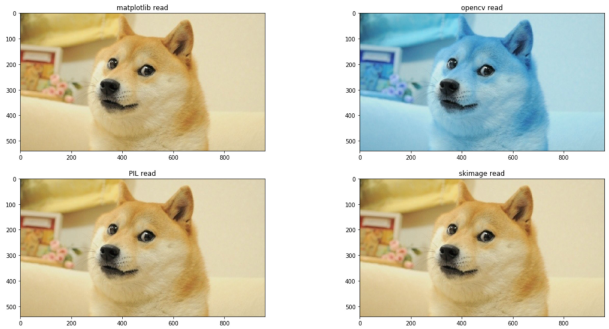
. . .


```
from skimage import io

img = io.imread(img_dir)
```

Colour channel

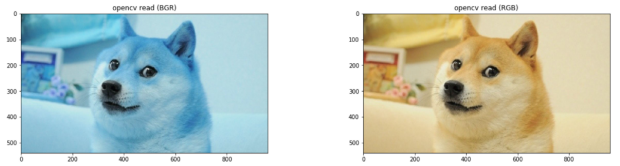
After loading the image, usually `plt.imshow(img)` will be used to plot the images. Let's plot some doge!



You may spot that the OpenCV image above looks odd. It is because matplotlib, PIL and skimage represent image in **RGB** (Red, Green, Blue) order, while **OpenCV is in reverse order**! (**BGR**—Blue, Green, Red)

Easy Fix

Just convert the image from BGR to RGB using `cv2.cvtColor(img, cv2.COLOR_BGR2RGB)` before plotting using `plt.imshow()` .



From blue to yellow doge

```
new_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

plt.imshow(new_img)
```

Efficiency

So, you may ask which one is the most efficient library in loading the image. Here a function is defined to track the time:

```
import time
def test_read_image(imgfile, func):
    t0 = time.time()
    img = func(imgfile)
    return img, time.time() - t0
```

The result is as follow:

```
+------------+--------------+---------+
|  Library   |   Function   |  Time   |
+------------+--------------+---------+
| matplotlib | plt.imread() | 0.02254 |
| OpenCV     | cv2.imread() | 0.01096 |
| Pillow     | Image.Open() | 0.00166 |
| Skimage    | io.imread()  | 0.01463 |
+------------+--------------+---------+
```

Pillow— `Image.Open()` seems to be the most efficient based on the result. For further study, we may go back to the source code to find out

more about the difference!

Cheatsheet

I have combined the above information into a Jupyter Notebook. Feel free to download the cheatsheet and happy coding!

Understanding image read libraries

In [1]:

```
import numpy as np

# image read libraries
from PIL import Image
import cv2
import matplotlib.pyplot as plt
import skimage
from skimage import io

img_dir = 'doge.jpg'
```

Source:

https://blog.csdn.net/renelian1572/article/details/78761278

https://github.com/ZhangXinNan/LearnPractice/blob/master/cv/opencv/test_cvlib.py