A L G O R I T H M

P R A C T I C E

深度学习算法与实践

Steven Tang

渡一教育
Duyi Education

文件和数据格式化

os库的使用

为什么使用os库

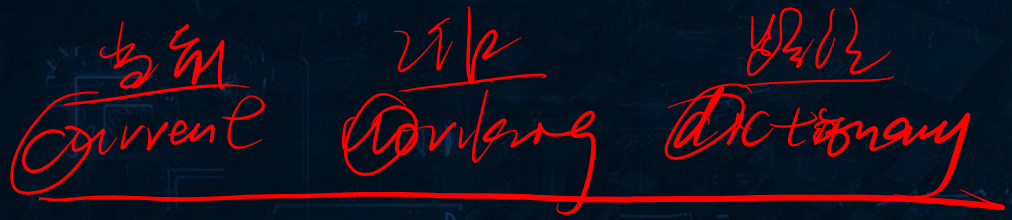
- 主要是涉及文件和路径的查找，os库提供了相对强大的方法。

```
import os

print("Hello world!")

os.system("pause")
```

C++

os模块主要方法

- os.getcwd()
- os.listdir()




```
In [1]: import os

In [2]: os.getcwd()
Out[2]: 'C:\\Users\\Dr. Tang.DESKTOP-DQDB847\\Desktop\\渡一\
\codes\\codes5'

In [3]: os.listdir
Out[3]: <function nt.listdir(path=None)>

In [4]: os.listdir()
Out[4]: ['.spyproject', 'reviews.txt', 'tanshishe.py',
'wordcount.py']
```

os模块主要方法

- os.makedirs()、os.mkdir()
- os.path.join()
- os.system()

```
In [5]: os.makedirs("test")

In [6]: os.makedirs("test/test")

In [7]: os.path.join(os.getcwd(),'test')
Out[7]: 'C:\\Users\\Dr. Tang.DESKTOP-DQDB847\\Desktop\\渡一\
\codes\\codes5\\test'
In [9]: os.path.abspath("test")
Out[9]: 'C:\\Users\\Dr. Tang.DESKTOP-DQDB847\\Desktop\\渡一\
\codes\\codes5\\test'
```

文件的使用

open()函数

使用相对路径打开

```
In [7]: textfile=open('firsttxt.txt')

In [8]: textfile.read()
Out[8]: 'Happy families are all alike, every unhappy
families is unhappy in its own way.'

In [9]: textfile.close()
```

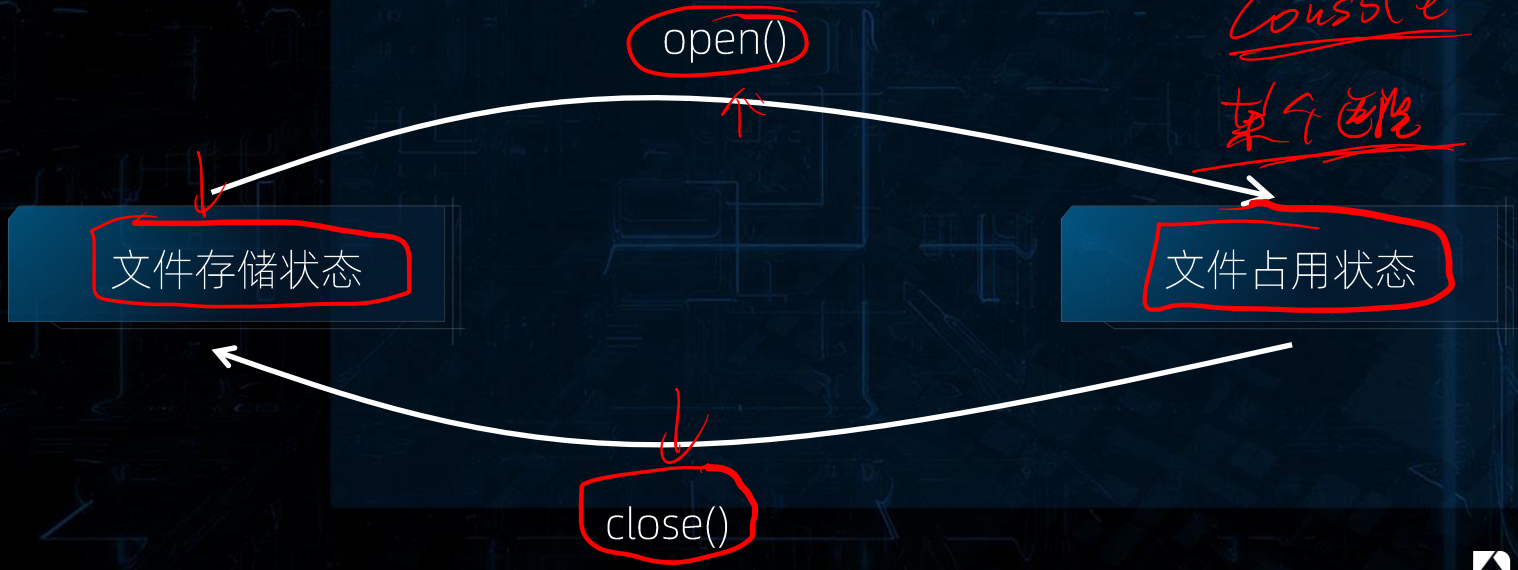
```
In [6]: textfile=open('Path/firsttxt.txt')

In [7]: textfile.read()
Out[7]: 'test'

In [8]: textfile.close()
```

绝对路径和相对路径

```
In [12]: textFile = open(r"C:\Users\Dr. Tang.DESKTOP-
DQDB847\Desktop\渡一\codes\codes6\firsttxt.txt","r")

In [13]: textFile.read()
Out[13]: 'Happy families are all alike; every unhappy family is
unhappy in its own way.'

In [14]: textFile = open("C:\\Users\\Dr. Tang.DESKTOP-DQDB847\
\Desktop\\渡一\\codes\\codes6\\firsttxt.txt","r")

In [15]: textFile.read()
Out[15]: 'Happy families are all alike; every unhappy family is
unhappy in its own way.'
```

文件的打开关闭

Python对文本文件采用统一的操作步骤，即“打开-操作-关闭”



文件的打开关闭

Python通过解释器内置的open()函数打开一个文件，并实现该文件与一个TextIOWrapper的关联，open()函数格式如下：

<变量名> = open(<文件名>, <打开模式>)

open()函数有两个参数：文件名和打开模式。文件名可以是文件的实际名字，也可以是包含完整路径的名字

```
In [21]: textFile
Out[21]: <_io.TextIOWrapper name='C:\\Users\\Dr. Tang.DESKTOP-
DQDB847\\Desktop\\渡一\\codes\\codes6\\firsttxt.txt' mode='r'
encoding='cp936'>
```

文件的打开关闭

open()函数提供7种基本的打开模式

打开模式	含义
'r'	只读模式，如果文件不存在，返回异常FileNotFoundError，默认值
'w'	覆盖写模式，文件不存在则创建，存在则完全覆盖源文件
'x'	创建写模式，文件不存在则创建，存在则返回异常FileExistsError
'a'	追加写模式，文件不存在则创建，存在则在原文件最后追加内容
'b'	二进制文件模式
't'	文本文件模式，默认值
'+'	与r/w/x/a一同使用，在原功能基础上增加同时读写功能

文件的打开关闭

基本的打开模式之间的区别

打开模式	含义
'r+'	r+读写模式，但不能创建文件
'w+'	新建读写，会将文件内容清零
'a+'	追加读写模式

with 用法

```
In [17]: with open("firsttxt.txt","r") as f:
    ...:     print(f.read())
    ...:
Happy families are all alike; every unhappy family is unhappy in
its own way.

In [18]: f
Out[18]: <_io.TextIOWrapper name='firsttxt.txt' mode='r'
encoding='cp936'>
```

文件的读写

常用的文件内容读取方法和文件的光标操作

方法	含义
<file>.readall()	读入整个文件内容，返回一个字符串或字节流*
<file>.read(size=-1)	从文件中读入整个文件内容，如果给出参数，读入前size长度的字符串或字节流
<file>.readline(size = -1)	从文件中读入一行内容，如果给出参数，读入该行前size长度的字符串或字节流
<file>.readlines(hint=-1)	从文件中读入所有行，以每行为元素形成一个列表，如果给出参数，读入hint行
<file>.seek(offset[,whence])	改变当前文件操作指针的位置，whence的值： 0：文件开头； 1: 当前位置； 2: 文件结尾

文件的读取

文件读取是一个单向流程。

```
In [21]: fo=open('lines_eng.txt','r')

In [22]: fo.readline()
Out[22]: 'this is the first line;\n'

In [23]: fo.readline()
Out[23]: 'this is the second line;\n'

In [24]: fo.readline()
Out[24]: 'this is the third line;'

In [25]: fo.readline()
Out[25]: ''

In [26]: fo.readline()
Out[26]: ''
```

文件的读取

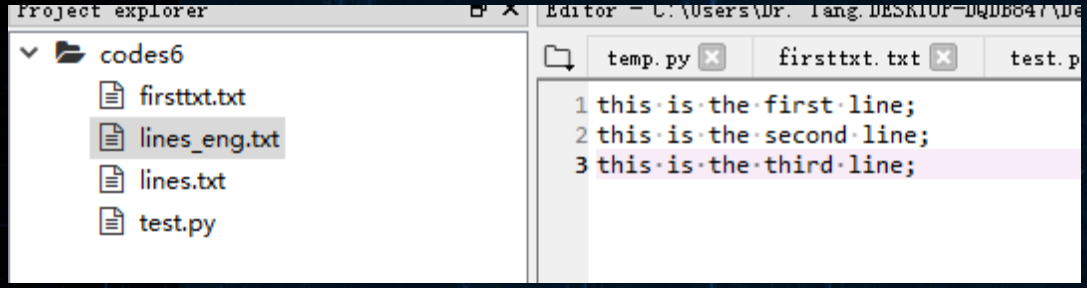
文件读取是一个单向流程。

```
In [13]: fo=open('lines_eng.txt','r')

In [14]: lines=fo.readlines()

In [15]: lines
Out[15]:
['this is the first line;\n',
 'this is the second line;\n',
 'this is the third line;']

In [16]: lines=fo.readlines()

In [17]: lines
Out[17]: []
```

```
In [18]: fo.seek(0)
Out[18]: 0

In [19]: lines=fo.readlines()

In [20]: lines
Out[20]:
['this is the first line;\n',
 'this is the second line;\n',
 'this is the third line;']
```

文件的读写

文本文件逐行打印

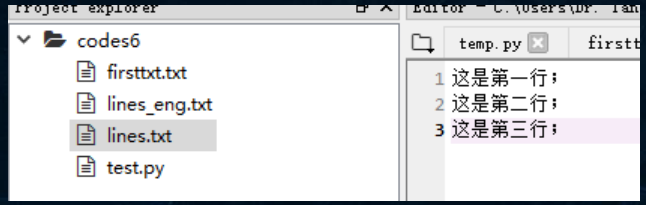



```
In [5]: fo=open('lines_eng.txt','r')

In [6]: for line in fo.readlines():
   ...:     print(line)
   ...:
this is the first line;

this is the second line;

this is the third line;
```

```
In [9]: fo=open('lines_eng.txt','r')

In [10]: for line in fo:
    ...:
    ...:     print(line)
    ...:
this is the first line;

this is the second line;

this is the third line;
```

文件的读写

中文文本文件逐行打印




```
In [1]: fo=open('lines.txt','r')

In [2]: for line in fo.readlines():
   ...:     print(line)
   ...:
Traceback (most recent call last):

  File "<ipython-input-2-e8d9cbf29128>", line 1, in <module>
    for line in fo.readlines():

UnicodeDecodeError: 'gbk' codec can't decode byte 0xac in
position 8: illegal multibyte sequence
```

```
In [7]: fo=open('lines.txt','r',encoding='utf-8')

In [8]: for line in fo.readlines():
   ...:
   ...:     print(line)
   ...:
这是第一行；

这是第二行；

这是第三行；
```

文件的写入

Python提供2个与文件内容写入有关的方法，如表所示。

方法	含义
<file>.write(s)	向文件写入一个字符串或字节流
<file>.writelines(lines)	将一个元素为字符串的列表写入文件

文件的写入

```
fo = open('2.txt', "w+",encoding='utf-8')
ls = ["唐诗", "宋词", "元曲"]
fo.writelines(ls)
fo.close()
```

```
for line in fo:
    print(line)
fo = open('3.txt', "w+")
ls = ["first line\nsecond line\n", "third line\n", "fourth line\n"]
fo.writelines(ls)
fo.close()
```

jieba库的使用

jieba库的概述

jieba是Python中一个重要的第三方中文分词函数库

```
pip install jieba -i
https://pypi.douban.com/simple/
```

jieba库是第三方库，不是安装包自带，需要通过pip指令安装

```
In [15]: import jieba

In [16]: jieba.lcut('今年将会是载入史册的一年')
Out[16]: ['今年', '将会', '是', '载入史册', '的', '一年']
```

jieba库的解析

函数	描述
jieba.cut(s)	精确模式，返回一个可迭代的数据类型
jieba.cut(s, cut_all=True)	全模式，输出文本s中所有可能单词
jieba.lcut(s)	精确模式，返回一个列表类型，建议使用
jieba.lcut(s, cut_all=True)	全模式，返回一个列表类型，建议使用

```
In [18]: jieba.lcut('今年将会是载入史册的一年',cut_all=True)
Out[18]: ['今年', '将', '会', '是', '载入', '载入史册', '史
册', '的', '一年']
```

```
In [17]: jieba.cut('今年将会是载入史册的一年')
Out[17]: <generator object Tokenizer.cut at
0x0000025E0886ECF0>
```

```
In [14]: for word in jieba.cut('今年将会是载入史册的一年'):
    ...:     print(word)
    ...:
今年
将会
是
载入史册
的
一年
```

使用结巴库进行中文的词频统计

```
import jieba
txt = open("shuihu.txt", "r", encoding='utf-8').read()
biaodian=', 。“”; : —’‘# $ % & ' （） * + ， - / '
for b in biaodian:
    if b in txt:
        txt=txt.replace(b,'')
words  = jieba.lcut(txt)
counts = {}

for word in words:
        if len(word) == 1:
            continue
        else:
            counts[word] = counts.get(word,0) + 1

items = list(counts.items())
items.sort(key=lambda x:x[1], reverse=True)
for i in range(20):
    word, count = items[i]
    print ("{0:<10}{1:>5}".format(word, count))
```

PIL库的使用

PIL库概述

PIL（Python Image Library）库是Python语言的第三方库，需要通过pip工具安装。

```
:\>pip install pillow -i
https://pypi.douban.com/simple/
```

PIL库概述

PIL库可以完成图像归档和图像处理两方面功能需求：

- 图像归档：对图像进行批处理、生成图像预览、图像格式转换等；
- 图像处理：图像基本处理、像素处理、颜色处理等。

PIL库Image类解析

在PIL中，任何一个图像文件都可以用Image对象表示Image类的图像读取和创建方法。图像模式有RGB、CMYK、P、L等等

方法	描述
Image.open(filename)	根据参数加载图像文件
Image.new(mode, size, color)	根据给定参数创建一个新的图像

```
In [34]: from PIL import Image

In [35]: img = Image.new("RGB",(32,32))

In [36]: img
Out[36]:
```

```
In [39]: img = Image.new("RGB",(32,32),'#00FF00')

In [40]: img
Out[40]:
```

PIL库Image类解析

要加载一个图像文件，最简单的形式如下：

Image.open

```
In [3]: from PIL import Image

In [4]: image=Image.open('Images/000001.jpg')

In [5]: image.show()

In [6]:
```

PIL库Image类解析

Image类有3个处理图片的常用属性

属性	描述
Image.format	标识图像格式或来源，如果图像不是从文件读取，值是None
Image.mode	图像的色彩模式，"L"灰度图像、"RGB"真彩色图像、"CMYK"出版图像
Image.size	图像宽度和高度，单位是像素（px），返回值是二元元组（tuple）

PIL库Image类解析

GIF文件图像提取。

对一个GIF格式动态文件，提取其中各帧图像，并保存为文件。

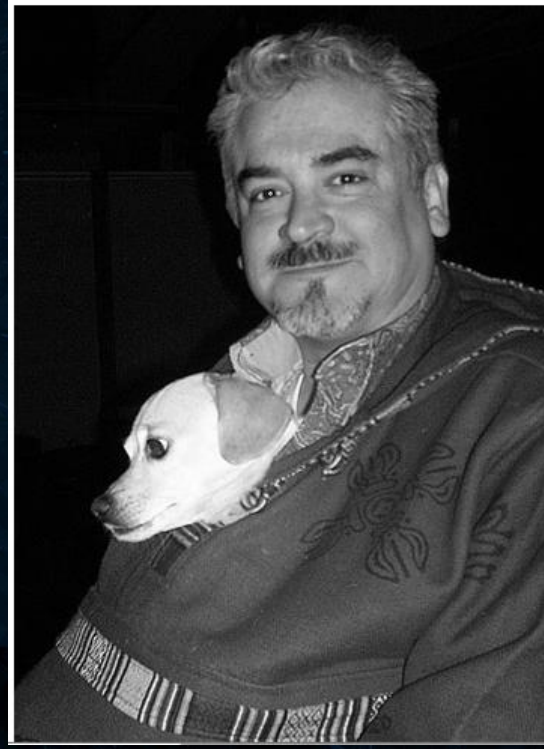
```
from PIL import Image
im = Image.open('tanshishe.gif')      # 读入一个GIF文件
try:
    im.save('GIFimages/picframe{:02d}.png'.format(im.tell()))
    while True:
        im.seek(im.tell()+1)
        im.save('GIFimages/picframe{:02d}.png'.format(im.tell()
except:
    print("处理结束")
```

PIL库Image类解析

Image类的图像转换和保存方法如表所示。

方法	描述
Image.save(filename, format)	将图像保存为filename文件名，format是图片格式
Image.convert(mode)	使用不同的参数，转换图像为新的模式
Image.thumbnail(size)	创建图像的缩略图，size是缩略图尺寸的二元元组

```
In [11]: image.convert(mode='L')
Out[11]:
```



PIL库Image类解析

生成“000001.jpg"图像的缩略图，其中（176，250）是缩略图的尺寸。

```
In [17]: image=Image.open('Images/000001.jpg')

In [18]: image.size
Out[18]: (353, 500)

In [19]: image.thumbnail((353//2,500//2))

In [20]: image
Out[20]:
```

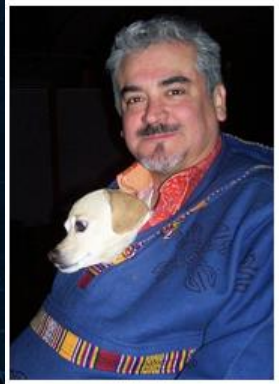


image.size

176 . 250

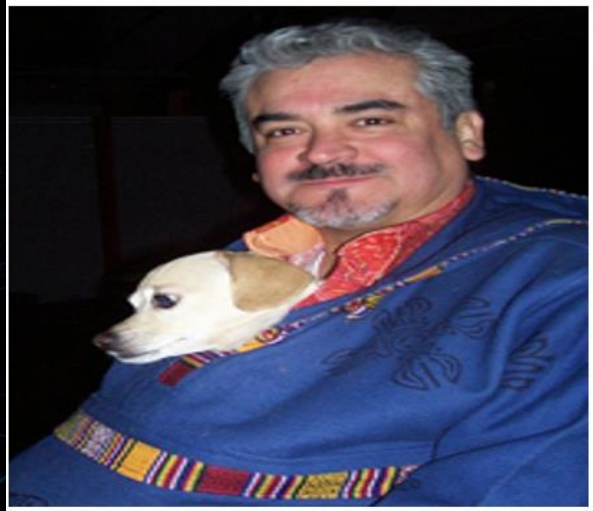
PIL库Image类解析

Image类可以缩放和旋转图像，注意resize与thumbnail的区别其中，rotate()方法以逆时针旋转的角度值作为参数来旋转图像。

方法	描述
Image.resize(size)	按size大小调整图像，生成副本
Image.rotate(angle)	按angle角度旋转图像，生成副本

PIL库旋转和缩放

```
In [26]: image_new=image.resize((440,380))

In [27]: image_new
Out[27]:
```



```
In [28]: image_new=image.rotate(30)

In [29]: image_new
Out[29]:
```

PIL库Image类解析

Image类能够对每个像素点或者一幅RGB图像的每个通道单独进行操作,split()方法能够将RGB图像各颜色通道提取出来，merge()方法能够将各独立通道再合成一幅新的图像。

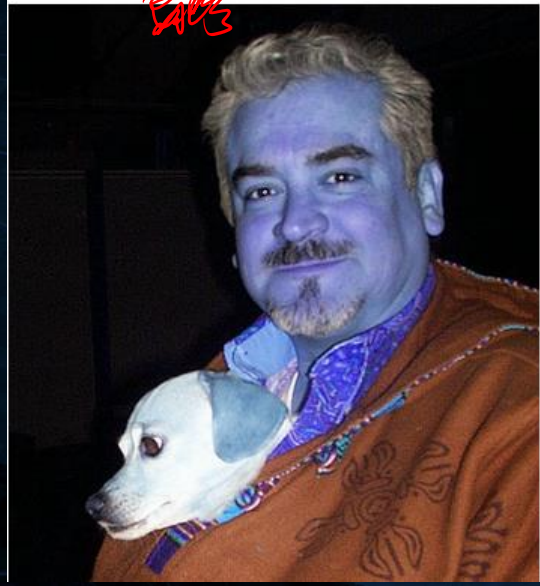
方法	描述
Image.point(func)	根据函数func功能对每个元素进行运算，返回图像副本
Image.split()	提取RGB图像的每个颜色通道，返回图像副本
Image.merge(mode，bands)	合并通道 ，采用mode色彩，bands是新色的色彩通道
Image.blend(im1,im2,alpha)	将两幅图片im1和im2按照如下公式插值后生成新的图像： im1 * (1.0-alpha) + im2 * alpha

PIL库Image类解析

图像的颜色交换。

交换图像中的颜色。可以通过分离RGB图片的三个颜色通道实现颜色交换

```
In [35]: from PIL import Image
    ...: rbg_image = Image.open('images/000001.jpg')
    ...: r, g, b = rbg_image.split()
    ...: bgr_image = Image.merge("RGB", (b, g, r))

In [36]: bgr_image
Out[36]:
```



PIL库Image类解析

操作图像的每个像素点需要通过函数实现，采用lambda函数和point()方法搭配使用，例子如下

```
In [47]: image = Image.open('Images/000001.jpg')

In [48]: newg=g.point(lambda i:i*0.9)

In [49]: newb=b.point(lambda i:i<150)

In [50]: new_image=Image.merge(image.mode,(r,newg,newb))

In [51]: new_image
Out[51]:
```

图像的过滤和增强

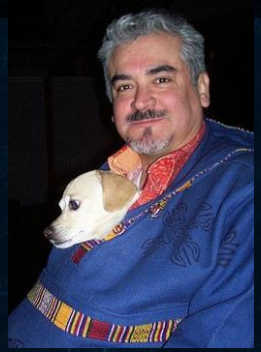
PIL库的ImageFilter类和ImageEnhance类提供了过滤图像和增强图像的方法，共10种

方法表示	描述
ImageFilter.BLUR	图像的模糊效果
ImageFilter.CONTOUR	图像的轮廓效果
ImageFilter.DETAIL	图像的细节效果
ImageFilter.EDGE_ENHANCE	图像的边界加强效果
ImageFilter.EMBOSS	图像的浮雕效果
ImageFilter.FIND_EDGES	图像的边界效果
ImageFilter.SMOOTH	图像的平滑效果
ImageFilter.SMOOTH_MORE	图像的阈值平滑效果
ImageFilter.SHARPEN	图像的锐化效果

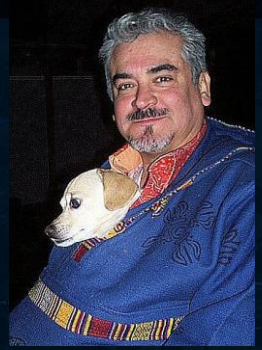
BLUR

CONTOUR

DETAIL



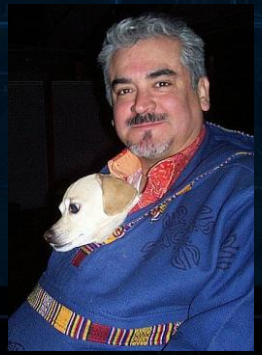
EDGE_ENHANCE



EMBOSS

FIND_EDGES

SMOOTH



SHARPEN

图像的过滤和增强

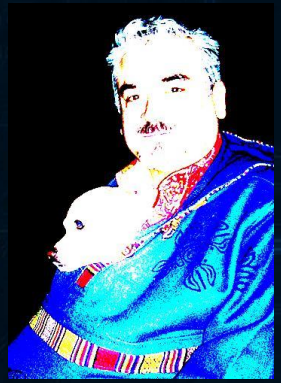
ImageEnhance类提供了更高级的图像增强需求，它提供调整色彩度、亮度、对比度、锐化等功能。

方法	描述
ImageEnhance.enhance(factor)	对选择属性的数值增强factor倍
ImageEnhance.Color(im)	调整图像的颜色平衡
ImageEnhance.Contrast(im)	调整图像的对比度
ImageEnhance.Brightness(im)	调整图像的亮度
ImageEnhance.Sharpness(im)	调整图像的锐度

图像的过滤和增强

Color

Contrast



Brightness

Sharpness