httpsblog.csdn.netbuchidanhuangarticledetails89716252utm\_medium=distribute.pc\_relevant.none-task-blog-BlogCommendFromBaidu-2.control&depth\_1-utm\_source=distribute.pc\_relevant.none-task-blog-BlogCommendFromBaidu-2.control

**理解h5文件并使用python对h5格式文件进行读写操作**



[buchidanhuanger](https://blog.csdn.net/buchidanhuang) 2019-05-01 14:59:31  22205  收藏 101

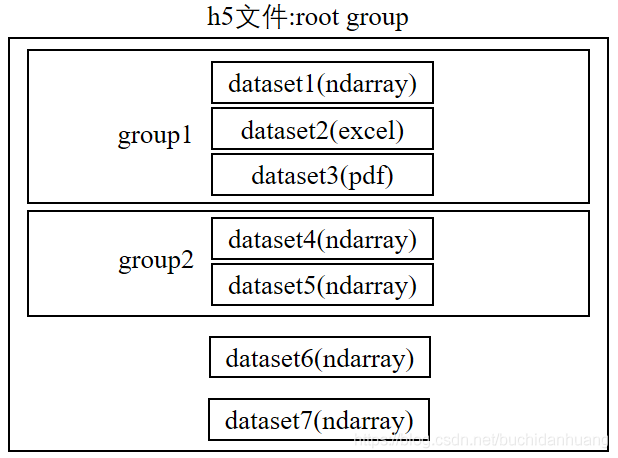
分类专栏： [编程相关](https://blog.csdn.net/buchidanhuang/category_9217263.html) 文章标签： [h5格式文件](https://so.csdn.net/so/search/s.do?q=h5%E6%A0%BC%E5%BC%8F%E6%96%87%E4%BB%B6&t=blog&o=vip&s=&l=&f=&viparticle=) [打开h5文件](https://www.csdn.net/tags/Mtjagg4sOTQ1MjEtYmxvZwO0O0OO0O0O.html) [生成h5文件](https://so.csdn.net/so/search/s.do?q=%E7%94%9F%E6%88%90h5%E6%96%87%E4%BB%B6&t=blog&o=vip&s=&l=&f=&viparticle=) [h5py](https://www.csdn.net/tags/Mtjakg2sMjg4OTYtYmxvZwO0O0OO0O0O.html) [python](https://www.csdn.net/tags/MtjaQg4sNDk0LWJsb2cO0O0O.html)

版权

**（一）HDF与h5**

  HDF(Hierarchical Data Format层次数据格式)是一种设计用于存储和组织大量数据的文件格式，最开始由美国国家超算中心研发，后来由一个非盈利组织HDF Group支持。HDF支持多种商业及非商业的软件平台，包括MATLAB、Java、Python、R和Julia等等，现在也提供了Spark。其版本包括了HDF4和现在大量用的HDF5。h5是HDF5文件格式的后缀。h5文件对于存储大量数据而言拥有极大的优势，这里安利大家多使用h5文件来存储数据，既高逼格又高效率。

**（二）h5文件数据组织方式:像Linux文件系统一样组织数据**

   h5文件中有两个核心的概念：组“group”和数据集“dataset”。 一个h5文件就是 “dataset” 和 “group” 二合一的容器。  
   dataset ：简单来讲类似数组组织形式的数据集合，像 numpy 数组一样工作，一个dataset即一个numpy.ndarray。具体的dataset可以是图像、表格，甚至是pdf文件和excel。  
   group：包含了其它 dataset(数组) 和 其它 group ，像字典一样工作。  
   一个h5文件被像linux文件系统一样被组织起来：dataset是文件，group是文件夹，它下面可以包含多个文件夹(group)和多个文件(dataset)。  
形象来看h5数据组织方式大概像酱婶儿的，诺！跟文件系统一样，大概知道它为啥叫层次数据格式了吧！  


**（三）使用python对h5文件进行操作**

python对h5文件的操作依赖于h5py包  
通过举个栗子来介绍h5py包是如何读写h5文件的  
读h5文件：

# Reading h5 file

import h5py

with h5py.File('cat\_dog.h5',"r") as f:

for key in f.keys():

#print(f[key], key, f[key].name, f[key].value) # 因为这里有group对象它是没有value属性的,故会异常。另外字符串读出来是字节流，需要解码成字符串。

print(f[key], key, f[key].name) # f[key] means a dataset or a group object. f[key].value visits dataset' value,except group object.

"""

结果：

<HDF5 group "/dogs" (1 members)> dogs /dogs

<HDF5 dataset "list\_classes": shape (2,), type "|S7"> list\_classes /list\_classes

<HDF5 dataset "train\_set\_x": shape (209, 64, 64, 3), type "|u1"> train\_set\_x /train\_set\_x

<HDF5 dataset "train\_set\_y": shape (209,), type "<i8"> train\_set\_y /train\_set\_y

代码解析：

文件对象f它表示h5文件的根目录(root group),前面说了group是按字典的方式工作的,通过f.keys()来找到根目录下的所有dataset和group的key，然后通过key

来访问各个dataset或group对象。

结果解析：

1.我们可以发现这个h5文件下有1个叫dogs的文件夹(group)和3个文件(dataset)它们分别叫list\_classes,train\_set\_x,train\_set\_y它们的shape都可知。

dogs group下有一个成员但我们不知道它是group还是dataset。

2.我们可以发现key和name的区别:

上层group对象是通过key来访问下层dataset或group的而不是通过name来访问的；

因为name属性它是dataset或group的绝对路径并非是真正的"name"，key才是真正的"name"。

name绝对路径：比如下文中访问name得到：/dogs/husky，它表示根目录下有dogs这个挂载点，dogs下又挂载了husky。

"""

dogs\_group = f["dogs"]

for key in dogs\_group.keys():

print(dogs\_group[key], dogs\_group[key].name)

"""

结果：

<HDF5 dataset "husky": shape (64, 64, 3), type "<f8"> /dogs/husky

可见dogs文件夹下有个key为husky的文件dataset

"""

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13
* 14
* 15
* 16
* 17
* 18
* 19
* 20
* 21
* 22
* 23
* 24
* 25
* 26
* 27
* 28
* 29
* 30
* 31
* 32
* 33
* 34

from h5py import Dataset, Group, File

with File('cat\_dog.h5','r') as f:

for k in f.keys():

if isinstance(f[k], Dataset):

print(f[k].value)

else:

print(f[k].name)

* 1
* 2
* 3
* 4
* 5
* 6
* 7

写h5文件：

# Writing h5

import h5py

import numpy as np

# mode可以是"w",为防止打开一个已存在的h5文件而清除其数据,故使用"a"模式

with h5py.File("animals.h5", 'a') as f:

f.create\_dataset('animals\_included',data=np.array(["dogs".encode(),"cats".encode()])) # 根目录下创建一个总览介绍动物种类的dataset,字符串应当字节化

dogs\_group = f.create\_group("dogs") # 在根目录下创建gruop文件夹:dogs

f.create\_dataset('cats',data = np.array(np.random.randn(5,64,64,3))) # 根目录下有一个含5张猫图片的dataset文件

dogs\_group.create\_dataset("husky",data=np.random.randn(64,64,3)) # 在dogs文件夹下分别创建两个dataset,一张哈士奇图片和一张柴犬的图片

dogs\_group.create\_dataset("shiba",data=np.random.randn(64,64,3))

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11

我们来检查一下这个animals.h5文件

with h5py.File('animals.h5','r') as f:

for fkey in f.keys():

print(f[fkey], fkey)

print("======= 优雅的分割线 =========")

'''

结果：

<HDF5 dataset "animals\_included": shape (2,), type "|S4"> animals\_included

<HDF5 dataset "cats": shape (5, 64, 64, 3), type "<f8"> cats

<HDF5 group "/dogs" (2 members)> dogs

'''

dogs\_group = f["dogs"] # 从上面的结果可以发现根目录/下有个dogs的group,所以我们来研究一下它

for dkey in dogs\_group.keys():

print(dkey, dogs\_group[dkey], dogs\_group[dkey].name, dogs\_group[dkey].value)

'''

husky <HDF5 dataset "husky": shape (64, 64, 3), type "<f8"> /dogs/husky [[[ 6.22221467e-01 2.29412386e-01 1.70099600e-01]

[-9.53310941e-01 -1.65325168e+00 6.50092663e-02]

[-2.33444396e-01 5.32328485e-01 -1.23046495e+00]

...

[-8.27186186e-04 -9.54570238e-01 1.20224835e+00]

[-3.03556381e-01 5.30470941e-01 -1.49928878e-01]

[ 5.24641964e-01 -1.55304472e+00 1.30016600e+00]]

...

'''