



您的位置: 首页 → 脚本专栏 → python → python pickle使用

请输入关键词



Python 中Pickle库的使用详解

更新时间: 2018年02月24日 11:45:36 作者: default

pickle是python语言的一个标准模块, 安装python后已包含pickle库, 不需要单独再安装。这篇文章主要介绍了Python 中Pickle库的使用详解,需要的朋友可以参考下

在“通过简单示例来理解什么是机器学习”这篇文章里提到了pickle库的使用, 本文来做进一步的阐述。

那么为什么需要序列化和反序列化这一操作呢?

1.便于存储。序列化过程将文本信息转变为二进制数据流。这样就信息就容易存储在硬盘之中, 当需要读取文件的时候, 从硬盘中读取数据, 然后再将其反序列化便可以得到原始的数据。在Python程序运行中得到了一些字符串、列表、字典等数据, 想要长久的保存下来, 方便以后使用, 而不是简单的放入内存中关机断电就丢失数据。python模块大全中的Pickle模块就派上用场了, 它可以将对象转换为一种可以传输或存储的格式。

2.便于传输。当两个进程在进行远程通信时, 彼此可以发送各种类型的数据。无论是何种类型的数据, 都会以二进制序列的形式在网络上传送。发送方需要把这个对象转换为字节序列, 在能在网络上传输; 接收方则需要把字节序列在恢复为对象。

通过简单示例来理解什么是机器学习

pickle是python语言的一个标准模块, 安装python后已包含pickle库, 不需要单独再安装。

pickle模块实现了基本的数据序列化和反序列化。通过pickle模块的序列化操作我们能够将程序中运行的对象信息保存到文件中去, 永久存储; 通过pickle模块的反序列化操作, 我们能够从文件中创建上一次程序保存的对象。

在官方的介绍中, 序列化操作的英文描述有好几个单词, 如“serializing”, “pickling”, “serialization”, “marshalling” 或者“flattening”等, 它们都代表的是序列化的意思。相应的, 反序列化操作的英文单词也有好多个, 如“de-serializing”, “unpickling”, “deserialization”等。为了避免混淆, 一般用“pickling”/“unpickling”, 或者“serialization”/“deserializatio n”。

pickle模块是以二进制的形式序列化后保存到文件中(保存文件的后缀为“.pkl”), 不能直接打开进行预览。而python的另一个序列化标准模块json, 则是human-readable的, 可以直接打开查看(例如在notepad++中查看)。

pickle模块有两类主要的接口, 即序列化和反序列化。

其中序列化操作包括:

```
1 pickle.dump()  
2 Pickler(file, protocol).dump(obj)
```

反序列化操作包括:

```
1 pickle.load()  
2 Unpickler(file).load()
```

2 序列化操作

2.1 序列化方法pickle.dump()

序列化的方法为 `pickle.dump()`, 该方法的相关参数如下:

大家感兴趣的内容

- 1 Python入门教程 超详细
- 2 Pycharm 2020最新永久
- 3 Python 元组(Tuple)操作
- 4 Python 列表(List)操作方
- 5 Python 字典(Dictionary)
- 6 Pycharm 2020年最新激
- 7 python strip()函数 介绍
- 8 pycharm 使用心得 (一)
- 9 python中使用xlrd、xlw
- 10 python 中文乱码问题深

最近更新的内容

python numpy数组中的集
python将秒数转化为时间
python搜索指定目录的方
python scrapy简单模拟登
浅谈python中的getattr函
python开发一个解析prot
Python 实现Mac 屏幕截
Python Pandas实现数据分
在Python中过滤Window
关于windos10环境下编译

常用在线小工具

CSS代码工具
JavaScript代码格式化工具
在线XML格式化/压缩工具
php代码在线格式化美化工
sql代码在线格式化美化工
在线HTML转义/反转义工
在线JSON代码检验/检验/
JavaScript正则在线测试工
在线生成二维码工具(加强
更多在线工具

```
1 | pickle.dump(obj, file, protocol=None, *, fix_imports=True)
```

该方法实现的是将序列化后的对象obj以二进制形式写入文件file中，进行保存。它的功能等同于 `Pickler(file, protocol).dump(obj)`。

关于参数file，有一点需要注意，必须是以二进制的形式进行操作（写入）。

参考前文的案例如下：

```
1 | import picklewith open('svm_model_iris.pkl', 'wb') as f:
2 |     pickle.dump(svm_classifier, f)
```

file为 `'svm_model_iris.pkl'`，并且以二进制的形式（'wb'）写入。

关于参数protocol，一共有5中不同的类型，即（0,1,2,3,4）。（0,1,2）对应的是python早期的版本，（3,4）则是在python3之后的版本。

此外，参数可选 `pickle.HIGHEST_PROTOCOL`和`pickle.DEFAULT_PROTOCOL`。当前，python3.5版本中，`pickle.HIGHEST_PROTOCOL`的值为4，`pickle.DEFAULT_PROTOCOL`的值为3。当protocol参数为负数时，表示选择的参数是`pickle.HIGHEST_PROTOCOL`。

关于参数protocol，官方的详细介绍如下：

There are currently 5 different protocols which can be used for pickling. The higher the protocol used, the more recent the version of Python needed to read the pickle produced.

- Protocol version 0 is the original "human-readable" protocol and is backwards compatible with earlier versions of Python.
- Protocol version 1 is an old binary format which is also compatible with earlier versions of Python.
- Protocol version 2 was introduced in Python 2.3. It provides much more efficient pickling of new-style classes. Refer to PEP 307 for information about improvements brought by protocol 2.
- Protocol version 3 was added in Python 3.0. It has explicit support for bytes objects and cannot be unpickled by Python 2.x. This is the default protocol, and the recommended protocol when compatibility with other Python 3 versions is required.
- Protocol version 4 was added in Python 3.4. It adds support for very large objects, pickling more kinds of objects, and some data format optimizations. Refer to PEP 3154 for information about improvements brought by protocol 4.



php 中文网

2.2 序列化方法pickle.dumps()

`pickle.dumps()`方法的参数如下：

```
1 | pickle.dumps(obj, protocol=None, *, fix_imports=True)
```

`pickle.dumps()` 方法跟 `pickle.dump()` 方法的区别在于，`pickle.dumps()` 方法不需要写入文件中，它是直接返回一个序列化的bytes对象。



Python客栈
Python知识百科



关注公众号 [Python客栈]
每天送红包，免费送纸质书
回复：学习资料 领优质高清教程视频
回复：电子书 领300+Python电子书

2.3 序列化方法Pickler(file, protocol).dump(obj)

`pickle`模块提供了序列化的面向对象的类方法，即 `class pickle.Pickler(file, protocol=None, *, fix_imports=True)`，`Pickler`类有`dump()`方法。

`Pickler(file, protocol).dump(obj)` 实现的功能跟 `pickle.dump()` 是一样的。

关于`Pickler`类的其他method，请参考官方API。

插播一条硬广：技术文章转发太多，本文来自微信公众号：“Python数据之道”（ID：PyDataRoad）。

3 反序列化操作

3.1 反序列化方法pickle.load()

序列化的方法为 `pickle.load()`，该方法的相关参数如下：

```
1 | pickle.load(file, *, fix_imports=True, encoding="ASCII", errors="strict")
```

该方法实现的是将序列化的对象从文件file中读取出来。它的功能等同于 `Unpickler(file).load()`。

关于参数file，有一点需要注意，必须是以二进制的形式进行操作（读取）。

参考前文的案例如下：

```
1 import picklewith open('svm_model_iris.pkl', 'rb') as f:
2     model = pickle.load(f)
```

file为' `svm_model_iris.pkl`'，并且以二进制的形式（'rb'）读取。

读取的时候，参数protocol是自动选择的，load()方法中没有这个参数。

3.2 反序列化方法pickle.loads()

pickle.loads()方法的参数如下：

```
1 pickle.loads(bytes_object, *,fix_imports=True, encoding="ASCII". errors="strict")
```

`pickle.loads()` 方法跟`pickle.load()`方法的区别在于，`pickle.loads()` 方法是直接从bytes对象中读取序列化的信息，而非从文件中读取。

3.3 反序列化方法Unpickler(file).load()

pickle模块提供了反序列化的面向对象的类方法，即 `class pickle.Unpickler(file, *,fix_imports=True, encoding="ASCII". errors="strict")`，Unpickler类有load()方法。

Unpickler(file).load() 实现的功能跟 `pickle.load()` 是一样的。


关于Unpickler类的其他method，请参考官方API。

4 那些类型可以进行序列化和反序列化操作


官方文档是这么介绍的，这里我就不进一步描述了。

The following types can be pickled:

- None, True, and False
- integers, floating point numbers, complex numbers
- strings, bytes, bytearray
- tuples, lists, sets, and dictionaries containing only picklable objects
- functions defined at the top level of a module (using def, not lambda)
- built-in functions defined at the top level of a module
- classes that are defined at the top level of a module
- instances of such classes whose `__dict__` or the result of calling `__getstate__()` is picklable (see section Pickling Class Instances for details).



Python数据之道



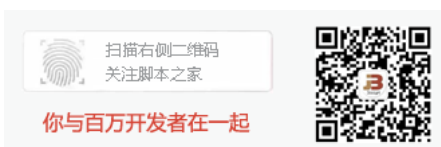
写在后面

pickle模块还是比较实用的，当然，关于pickle模块，其实还有许多的信息可以去了解，想了解更多信息的童鞋，建议可以阅读下python官方的API文档（library文件）。

以上所述是小编给大家介绍的Python 中Pickle库的使用详解，希望对大家有所帮助，如果大家有任何疑问请给我留言，小编会及时回复大家的。在此也非常感谢大家对脚本之家网站的支持！

您可能感兴趣的文章：

[python爬取之json、pickle与shelve库的深入讲解](#)
[Python 解析库json及jsonpath pickle的实现](#)
[Python标准库json模块和pickle模块使用详解](#)
[Python使用Pickle库实现读写序列操作示例](#)
[Python pickle类库介绍（对象序列化和反序列化）](#)
[老生常谈Python中的Pickle库](#)



微信公众号搜索“脚本之家”，选择关注
程序猿的那些事、送书等活动等着你

[python](#) [pickle](#)

相关文章



Python脚本实现Web漏洞扫描工具

这是去年毕设做的一个Web漏洞扫描小工具，主要针对简单的SQL注入漏洞、SQL盲注和XSS漏洞。下文给大家介绍了使用说明和源代码，一起看看吧

2016-10-10



Python 字符串池化的前提

这篇文章主要介绍了Python 字符串池化的前提，文中示例代码非常详细，帮助大家更好的理解和学习，感兴趣的朋友可以了解下

2020-07-07



用Python获取亚马逊商品信息

大家好，本篇文章主要讲的是用Python获取亚马逊商品信息，感兴趣的同学赶快来看一看吧，对你有帮助的话记得收藏一下，方便下次浏览

2022-01-01



浅谈Tensorflow 动态双向RNN的输出问题

今天小编就为大家分享一篇浅谈Tensorflow 动态双向RNN的输出问题，具有很好的参考价值，希望对大家有所帮助。一起跟随小编过来看看吧

2020-01-01



python3 使用traceback定位异常实例

这篇文章主要介绍了python3 使用traceback定位异常实例，具有很好的参考价值，希望对大家有所帮助。一起跟随小编过来看看吧

2020-03-03



在Django中自定义filter并在template中的使用详解

这篇文章主要介绍了在Django中自定义filter并在template中的使用详解，具有很好的参考价值，希望对大家有所帮助。一起跟随小编过来看看吧

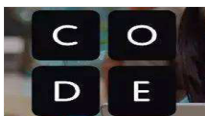
2020-05-05



Python matplotlib修改默认字体的操作

这篇文章主要介绍了Python matplotlib修改默认字体的操作，具有很好的参考价值，希望对大家有所帮助。一起跟随小编过来看看吧

2020-03-03



对tensorflow中cifar-10文档的Read操作详解

今天小编就为大家分享一篇对tensorflow中cifar-10文档的Read操作详解，具有很好的参考价值，希望对大家有所帮助。一起跟随小编过来看看吧

2020-02-02



Django Admin实现三级联动的示例代码(省市区)

多级菜单在很多上面都有应用，这篇文章主要介绍了Django Admin实现三级联动(省市区)，小编觉得挺不错的，现在分享给大家，也给大家做个参考。一起跟随小编过来看看吧

2018-06-06



Python安装spark的详细过程

这篇文章主要介绍了Python安装spark的详细过程，本文通过图文实例代码相结合给大家介绍的非常详细，对大家的学习或工作具有一定的参考借鉴价值,需要的朋友可以参考下

2021-10-10

最新评论

[关于我们](#) - [广告合作](#) - [联系我们](#) - [免责声明](#) - [网站地图](#) - [投诉建议](#) - [在线投稿](#)

©CopyRight 2006-2021 JB51.Net Inc All Rights Reserved. 脚本之家 版权所有