

多模态神经影像数据与特征存储方法的设计

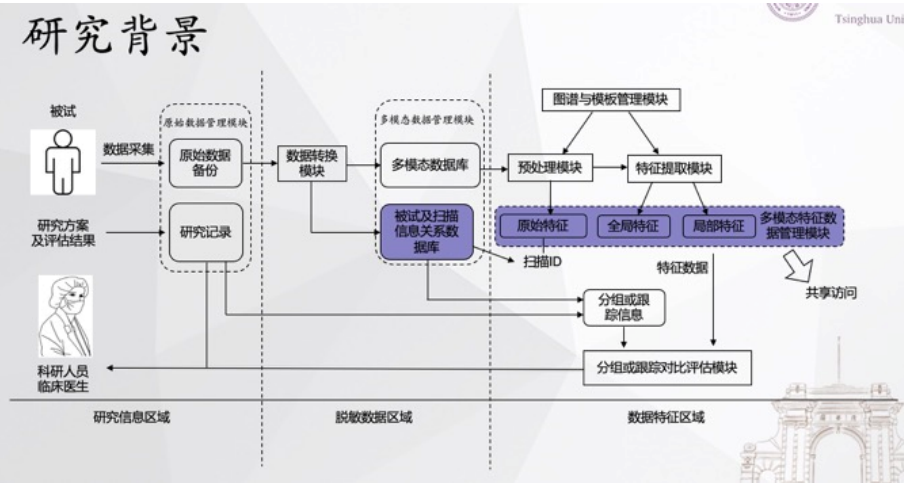
无81 王力 2018011037

【摘要】：当今时代，数据量存储的规模越来越大，并且海量的数据背后带来的信息是很多互联网公司的生产力；换言之，这是一个大数据的时代，一切跟数据有关的存储、分析以及深度学习都显得至关重要。而脑科学的发展也来到与数据科学相结合的时代，扫描得到的脑图谱数据，分析提取脑电的特征数据，然后将这些多模态的神经影像数据存储，并且能够方便的查询和使用，既能够服务于医护工作者和患者，又能够为后续的脑科学的数据分析提供一个方便管理的模式，是我们此次SRT项目的目标。结题报告大致分为如下几个部分：项目介绍、项目开发流程、项目总结。

一，项目介绍

脑科学和类脑人工智能研究，都需要探秘人脑神经系统的结构、状态、工作原理和模式。尤其是在中枢神经系统受损状态下的精准诊疗和康复，更期待对人脑神经功能可塑性机理和模式的揭秘。在这些探秘过程中产生了大量的多模态神经影像数据。同时，神经科学研究正逐步地走向大数据时代，研究者提取的特征呈现多样化的趋势。如何高效地存储、访问这些数据 and 特征成为研究者需要认真考虑的问题。

神经影像数据的采集方式主要为磁共振成像(Magnetic Resonance Imaging)和脑电(electroencephalogram, EEG)。由它们产生出不同模态，反映不同组织结构以及功能信息的数据被称为多模态神经影像数据。对这些数据进行特征提取及融合，会生成大量的特征数据。目前开发设计一种高效地存储、访问这些数据和特征的方法。



二，项目开发过程

2.1 项目使用工具 MongoDB 、Python

MongoDB是一个使用C++开发的高性能、开源、无模式的文档型数据库，是当前NoSQL数据库中使用最广泛的一种数据库。在许多场景下用于替代传统的关系型数据库或键值对存储方式，旨在为WEB应用提供可扩展的高性能数据存储解决方案。在具体使用过程中，MongoDB将数据存储成一个JSON文档，对应的分级结构有数据库(database)、表(collection),可以很便利地进行CURD操作。并且每个文档没有固定的键值对完整性约束，所以提供了灵活的可扩展性，非常适合用于存储多模态神经影响数据。

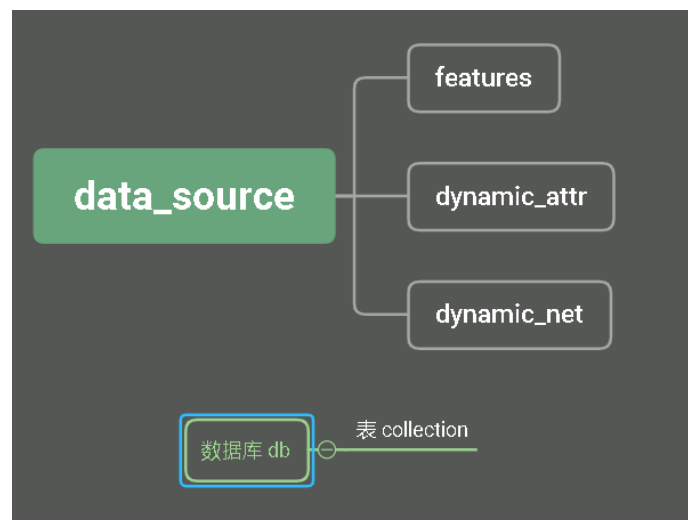
Python编程语言便于学习、开发，并且和MongoDB有完整的API接口，以及将脑电数据与matlab交互都非常方便。本项目开发Mongo部分主要使用了pymongo包，相关的API接口介绍均可以查阅官方文档获得。

2.2 MongoDB数据库模式设计

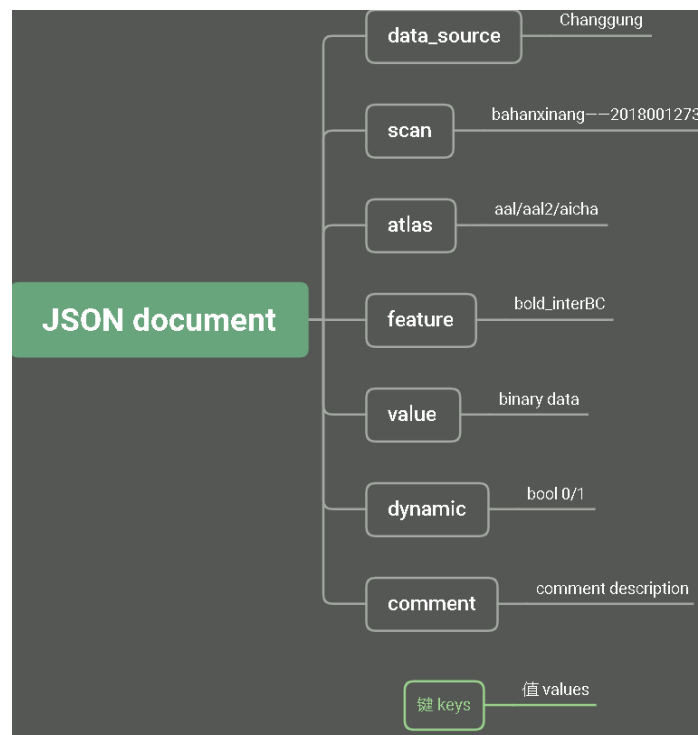
本项目使用到多模态神经影响数据可以分为静态属性数据、静态网络数据、动态属性数据和动态网络数据。对应的具体文件结构是：扫描文件夹(scan)->图谱文件夹(atlas)->属性文件夹/文件(feature)

2.2.1 第一种存储设计模式

第一种设计模式：如下图所示，数据源名称做数据库名称，features、dynamic_attr、dynamic_net是每个数据源对应数据库下的三个表，其中features负责存储所有的静态属性和静态网络数据，dynamic_attr存储所有的动态属性数据，dynamic_net存储所有的动态网络数据。



每个表中存储的文档结构如下图所示，每个文档会有许多的键值对组成，动态数据的键值对还会扩展几项。



该存储模式的优缺点：扁平化设计，设计的原理比较简单，每个文档可以充分地存储信息。但是由于每个文档都会重复性地存储几个键值对信息，所以造成了占用额外的资源空间，并且由于每个数据库下只分了三个表，所以每个表对应的数据量非常大，每次在加载数据时都会占用非常多的内存，在做测试时，由于内存开销过大，查询时间缓慢，查询过程受到严重影响。

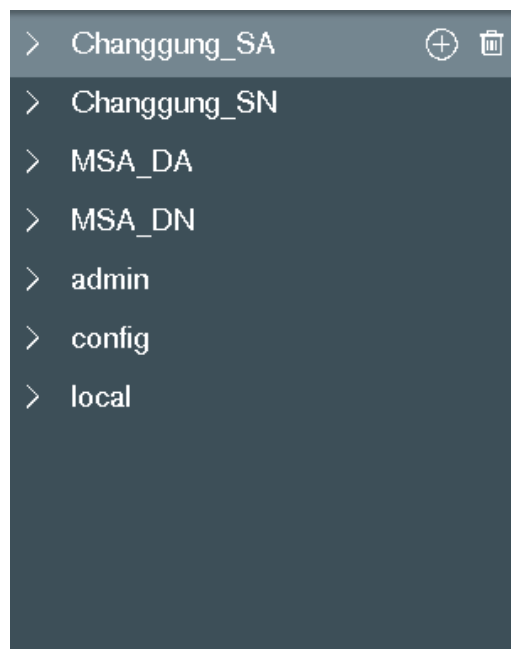
考虑到具体的内存占用以及避免重复性存储的问题，我们重新设计了数据库的存储模式。

2.2.2 第二种存储设计模式

为了避免重复性的存储，定义四种数据的属性为

```
{
  "static attr": 'SA',
  "static net": 'SN',
  "dynamic attr": 'DA',
  "dynamic net": 'DN',
}
```

数据库名采用数据源和数据属性拼接的方式来命名，表名采用图谱名和特征名拼接的方式来命名。



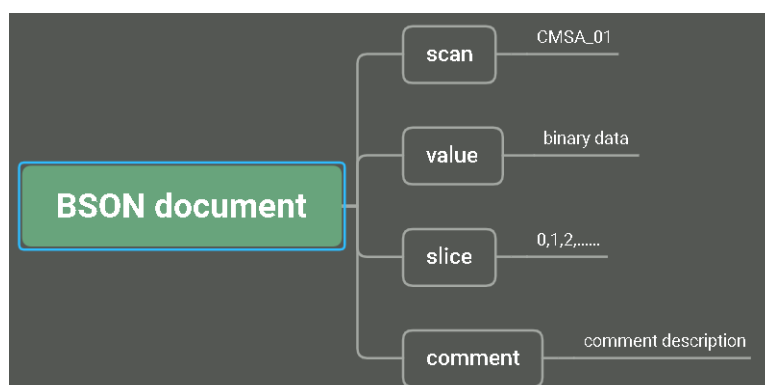
aal:BOLD.BC.inter

aal:BOLD.CCFS.inter

aal:BOLD.LE.inter

aal:BOLD.WD.inter


JSON 文档的键值对就得以减少，基本不会存放可能重复的数据。如下图所示，只有扫描键值对、值数据键值对、分片键值以及评价对应的键值对。每一个键值对存储的数据，基本不会在其他的文档中找到重复的内容。所以这种设计模式减少了无用的空间开销，每个文档的大小变小，并且文档的数目相较于模式I没有增加。



第二种设计模式在理论上相较第一种模式性能更优，具体的测试数据在之后给出。

2.3 代码文件清单

开发的MongoDB部分代码文件均已经上传至GitHub项目mongo分支，以下为代码文件的截图及简介

 handsomewl	update	b6237ce 26 days ago	🕒 175 commits
config	Add readme and adjust file structures	8 months ago	
.gitignore	Merge branch 'mongo' of https://github.com/geyunxiang/MMDPDatabases...	3 months ago	
EEG_conf.json	update EEG JSON	3 months ago	
MongoDB.py	update	26 days ago	
Mongotest.py	update mongo test script	26 days ago	
README.md	Add readme and adjust file structures	8 months ago	
gridfs_test.py	提交gridfs	7 months ago	
migrate.py	Add migrate.py	2 months ago	
migrate_test.py	new model	2 months ago	
mmdpdb.py	update	26 days ago	
mmdpdb_test.py	Signed-off-by: yaojh18 <yaojh18@mails.tsinghua.edu.cn>	4 months ago	
mongo_test.py	new model	2 months ago	
mongo_test_parallel.py	Add mongo parallel test	4 months ago	
mongodb_database.py	new model	2 months ago	
redis_cache_test.py	Signed-off-by: yaojh18 <yaojh18@mails.tsinghua.edu.cn>	4 months ago	
redis_database.py	Merge branch 'redis' into mongo	4 months ago	
redis_test.py	Signed-off-by: yaojh18 <yaojh18@mails.tsinghua.edu.cn>	4 months ago	
test_redis.py	Add readme and adjust file structures	8 months ago	
testmat.py	just update	3 months ago	

文件名	简介
EEG_conf.json	EEG数据库配置文件
MongoDB.py	模式II对应的类设计
Mongotest.py	模式II对应的测试
gridfs_test.py	gridfs使用测试
migrate.py	数据库迁移文件
mongodb_database.py	模式I对应的类设计
mongo_test.py	模式I对应的测试
testmat.py	处理mat文件的测试

2.4 数据库实现总览

最后我们选择模式II的设计来生成数据库。

Database Name ^	Storage Size	Collections	Indexes	
Changgung_SA	8.2MB	20	40	
Changgung_SN	788.0MB	5	10	
MSA_DA	71.4MB	32	64	
MSA_DN	599.9MB	2	3	

模式II下生成的四个数据库总览

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes ^	Total Index Size
aal-BOLD.BC.inter	472	1.1 KB	536.3 KB	2	44.0 KB
aal-BOLD.CCFS.inter	472	1.1 KB	536.3 KB	2	44.0 KB
aal-BOLD.LE.inter	472	1.1 KB	536.3 KB	2	44.0 KB
aal-BOLD.WD.inter	472	1.1 KB	536.3 KB	2	44.0 KB

Changgung_SA数据库中的部分表

<pre>_id:ObjectId("5f9e592643ea878ae8fad127") scan:"baihanxiang_20190211" : Binary('gANjbnVtcHkuY29yZS5tdWx0aWwFycmF5C19yZWVnbmN0cnVjdApwAGNudW1weQpuZGFycmF5CnEB5wCFcQJDAWJxA4dxBFJxB5') comment:Object</pre>	
<pre>_id:ObjectId("5f9e592643ea878ae8fad128") scan:"baihanxiang_20190307" : Binary('gANjbnVtcHkuY29yZS5tdWx0aWwFycmF5C19yZWVnbmN0cnVjdApwAGNudW1weQpuZGFycmF5CnEB5wCFcQJDAWJxA4dxBFJxB5') comment:Object</pre>	
<pre>_id:ObjectId("5f9e592643ea878ae8fad129") scan:"caipinrong_20180412" : Binary('gANjbnVtcHkuY29yZS5tdWx0aWwFycmF5C19yZWVnbmN0cnVjdApwAGNudW1weQpuZGFycmF5CnEB5wCFcQJDAWJxA4dxBFJxB5') comment:Object</pre>	
<pre>_id:ObjectId("5f9e592643ea878ae8fad12a") scan:"caochangsheng_20161027" : Binary('gANjbnVtcHkuY29yZS5tdWx0aWwFycmF5C19yZWVnbmN0cnVjdApwAGNudW1weQpuZGFycmF5CnEB5wCFcQJDAWJxA4dxBFJxB5') comment:Object</pre>	

aal-BOLD.BC.inter表中部分JSON文档

2.5 具体测试以及性能对比

由于本项目的多模态神经影像数据库是读多于写的数据库，所以查询时间是一项很重要的衡量数据库性能的指标。loader是基于文件系统的查询方法，MongoDB数据库查询是在mongo数据库中查询，比较两种方式查询相同文件的所用时间，进而比较MongoDB相较之前存储方式的性能提升。

静态Attr数据

TEST1	TEST2	TEST3	TEST4	COUNT
-------	-------	-------	-------	-------

	TEST1	TEST2	TEST3	TEST4	COUNT
Loader	12.43s	12.23s	10.70s	9.87s	9133
Mongo	23.68s	18.98s	21.02s	20.88s	9133

静态Net数据

	TEST1	TEST2	TEST3	TEST4	COUNT
Loader	153.22s	144.33s	142.45s	143.42s	2287
Mongo	14.08s	8.27s	8.10s	7.79s	2287

动态Attr数据

	TEST1	TEST2	TEST3	TEST4	COUNT
Loader	9.07s	2.18s	2.18s	2.18s	44
Mongo	10.88s	5.11s	5.12s	5.02s	44

动态Net数据

	TEST1	TEST2	TEST3	TEST4	COUNT
Loader	122.13s	81.19s	81.21s	81.47s	44
Mongo	29.08s	29.01s	28.00s	29.14s	44

在属性数据方面，loader查询表现比mongo要好一些，但是在查询网络数据时，mongo查询的性能相较loader会有较大的提升。loader相当于是直接进行数据的文件系统定位，再进行相应的按照文件位置进行的查找，时间复杂度方面相当于是 $O(n)$ ，但是在实际使用的过程中，数据存储在文件系统，首先是路径也许未知，然后是数据没有压缩，占用较多的空间；MongoDB数据库的方案会将数据压缩成二进制之后存入，是在数据库系统中进行查询，未加索引时，查询的时间复杂度是 $O(n^2)$ ，在合适的field加上索引之后，查询时间复杂度可以接近 $O(\lg n)$ 。

三，总结

在经过为期大致一年的预调研、设计模型、编程、调试等步骤，基本开发出功能较为完善的基于MongoDB存储多模态神经影像数据库，可以正常实现CURD操作，并且具有良好的可扩展性。衷心感谢窦维蓓老师、葛云祥学长的指导，以及和我共同开发的搭档姚季涵同学，大家都提出了很多宝贵意见和建议，使我们的项目得以一步步的开发下去，并且最终实现这一性能较好、扩展性强的数据库。

四，参考文献

参考文献基本为互联网的文档资源

[1]MongoDB教程|菜鸟教程.<https://www.runoob.com/mongodb/mongodb-tutorial.html>

[2]The MongoDB 4.4 Manual.<https://docs.mongodb.com/manual/>

[3]The MongoDB 4.4 Manual:The `mongo` Shell .<https://docs.mongodb.com/manual/mongo/>

[4]The MongoDB 4.4 Manual:Indexes.<https://docs.mongodb.com/manual/indexes/>

[5]The MongoDB 4.4 Manual:Storage<https://docs.mongodb.com/manual/storage/>

[6]MongoDB documentation Guides.<https://docs.mongodb.com/guides/>

[7]Tutorial — PyMongo 3.9.0 documentation - MongoDB API.<https://api.mongodb.com/python/current/tutorial.html>

[8]常见问题：MongoDB诊断.<https://mongodb.net.cn/manual/faq/diagnostics/>

[9]MongoDB 无法启动，WiredTiger 如何恢复数据（二）？ | MongoDB中文社区.<https://mongoing.com/archives/5371>

[10]MongoDB 存储引擎 WiredTiger 原理解析 .<https://mongoing.com/archives/5367>

[11]MongoDB索引原理 .<https://mongoing.com/archives/2789>

[12]论MongoDB索引选择的重要性.<https://mongoing.com/archives/4913>

[13]MongoDB 事务，复制和分片的关系.<https://mongoing.com/archives/38461>

[14]常见问题：并发_MongoDB中文网.<https://mongodb.net.cn/manual/faq/concurrency/#what-isolation-guarantees-does-mongodb-provide>