

linger(心怀梦想，活在当下)

机器学习，深度学习，数据挖掘，推荐系统，分布式算法

☰ 目录视图

☰ 摘要视图

RSS 订阅

个人资料



lingerlanlan

访问：219483次

积分：3991

等级：BLOG 5

排名：第3738名

原创：156篇

转载：0篇

译文：2篇

评论：260条

文章搜索

文章分类

深度学习（deep learning） (28)

机器学习 (16)

cuda(GPU programming) (20)

文本挖掘 (5)

c/c++ (15)

dota外挂 (5)

hack programming (6)

web2.0 (5)

工具源码 (7)

语言学习 (22)

设计模式 (2)

读书笔记 (1)

翻译 (1)

足球大数据 (2)

大杂烩 (18)

Hadoop (12)

Spark (2)

sklearn (1)

文章存档

2015年08月 (1)

caffe源码分析--SyncedMemory类代码研究

分类：深度学习（deep learning）2014-04-24 19:422347人阅读评论(1) 收藏 举报

神经网络深度学习机器学习cudacaffe

数据成员：

void*cpu_ptr_;

数据

指针

void*gpu_ptr_;

数据在gpu的指针

size_tsize_;

数据的大小

SyncedHeadhead_;

表示数据的状态，有四种状态，分别是未初始化，数据在cpu中，数据在gpu中，数据在cpu和gpu中都有

enumSyncedHead { UNINITIALIZED, HEAD_AT_CPU, HEAD_AT_GPU, SYNCED};

构造函数

SyncedMemory()

:cpu_ptr_(NULL), gpu_ptr_(NULL), size_(0), head_(UNINITIALIZED) {}

简单的初始化

explicitSyncedMemory(size_tsize)

:cpu_ptr_(NULL), gpu_ptr_(NULL), size_(size), head_(UNINITIALIZED) {}

只是把size（大小）设置了，并未申请内存

析构函数

SyncedMemory::~SyncedMemory() {

//如果cpu有数据，则释放

if(cpu_ptr_){

CaffeFreeHost(cpu_ptr_);

}

http://blog.csdn.net/lingerlanlan/article/details/24379607

1/6

2015年07月 (3)

2015年06月 (3)

2015年05月 (3)

2015年04月 (8)

展开

最新评论

总结一下用caffe跑图片数据的研
liangzhituzi: @zzq1989_:可能是那
两个文件路径的问题，可以看
看train_prototxt里面的路径

deep learning实践经验总结
查志强: 问下，怎样判断“错误”的
标签？

神经网络的CNN框架 VS caffe
fqss0436: 博主，您好，谢谢您
分享代码。在调试您的代码时，
程序中断于175行
caffe_test_net.For...

我所写的CNN框架 VS caffe
gzp95: 楼主，求问一下您写的代
码的速度和caffe的速度有多大的
差距。因为最近在实现word2vec
的cud...

总结一下用caffe跑图片数据的研
依然_范佩西11: 训练完的模型，
如是调用呢，能说下能么测试单
张图像或者批量图像的流程么

Dota全图那些事儿
女主、女主:。。。单机理论效
果，实际不好用啊。。。。支
持一下~不错的

caffe源码修改：抽取任意一张图
wwdzhbknjwcnmd: 想请教一下博
主，caffe网络中batch_size和
crop_size这两个参数的含义是什
么？哪一...

caffe源码分析--data_layer.cpp
沧海1梦: 请问caffe中如何修改输
入和裁剪尺寸，因为我的图像大
小是48的，想通过修改alexnet来
训练，还...

caffe卷积神经网络框架安装
yang123jx: 我也遇到
relu_layer.cu:29 check failed
error == cudaSuc...

caffe卷积神经网络框架安装
yang123jx: 我也遇到
relu_layer.cu:29 check failed
error == cudaSuc...

阅读排行

总结一下用caffe跑图片数据 (7192)

word2vector学习笔记 (· (6942)

caffe神经网络框架的辅助 (6147)

caffe源码修改：抽取任意 (5905)

caffe卷积神经网络框架安装 (5550)

caffe源码分析--data_layer (5374)

神经网络：caffe特征可视化 (4679)

word2vec源码解析之word (4510)

caffe源码分析--Blob类代码 (4386)

deep learning实践经验总结 (4225)

推荐文章

//如果gpu有数据，则释放

```
if(gpu_ptr_){  
  
    CUDA_CHECK(cudaFree(gpu_ptr_));  
  
}  
  
}
```

函数voidto_cpu()

功能：把数据放到cpu上

1数据未初始化，则在cpu申请内存。此时状态为HEAD_AT_CPU

2数据本来在gpu，则从gpu拷贝内存到cpu。此时状态为SYNCED

3数据本来在cpu，不做处理

4数据在cpu和gpu都有，不做处理

```
inlinevoidSyncedMemory::to_cpu(){  
  
    switch(head_){  
  
        caseUNINITIALIZED:  
  
            CaffeMallocHost(&cpu_ptr_,size_);  
  
            memset(cpu_ptr_,0,size_);  
  
            head_=HEAD_AT_CPU;  
  
            break;  
  
        caseHEAD_AT_GPU:  
  
            if(cpu_ptr_==NULL){  
  
                CaffeMallocHost(&cpu_ptr_,size_);  
  
            }  
  
            CUDA_CHECK(cudaMemcpy(cpu_ptr_,gpu_ptr_,size_,cudaMemcpyDeviceToHost));  
  
            head_=SYNCED;  
  
            break;  
  
        caseHEAD_AT_CPU:  
  
        caseSYNCED:  
  
            break;  
  
    }  
  
}
```

函数voidto_gpu();

功能：把数据放到gpu上

1数据未初始化, 在gpu申请内存。此时状态为HEAD_AT_GPU

2数据在cpu, 从cpu拷贝到gpu。此时状态为SYNCED

3数据在gpu, 不做操作。

4数据在cpu和gpu都有, 不做操作。

```
inline void SyncedMemory::to_gpu() {  
    switch(head_) {  
        case UNINITIALIZED:  
            CUDA_CHECK(cudaMalloc(&gpu_ptr_, size_));  
            CUDA_CHECK(cudaMemset(gpu_ptr_, 0, size_));  
            head_ = HEAD_AT_GPU;  
            break;  
        case HEAD_AT_CPU:  
            if(gpu_ptr_ == NULL) {  
                CUDA_CHECK(cudaMalloc(&gpu_ptr_, size_));  
            }  
            CUDA_CHECK(cudaMemcpy(gpu_ptr_, cpu_ptr_, size_, cudaMemcpyHostToDevice));  
            head_ = SYNCED;  
            break;  
        case HEAD_AT_GPU:  
        case SYNCED:  
            break;  
    }  
}
```

函数 `const void* cpu_data()`;

功能: 返回数据在cpu的指针

```
const void* SyncedMemory::cpu_data() {  
    to_cpu();  
    return (const void*)cpu_ptr_;  
}
```

函数`const void*gpu_data()`;

功能: 返回数据在gpu的指针

```
const void*SyncedMemory::gpu_data() {  
    to_gpu();  
    return(const void*)gpu_ptr_;  
}
```

函数`void*mutable_cpu_data()`;

功能: 返回数据在cpu的指针, 并改变数据的状态为`HEAD_AT_CPU`

```
void*SyncedMemory::mutable_cpu_data() {  
    to_cpu();  
    head_=HEAD_AT_CPU;  
    returncpu_ptr_;  
}
```

函数`void*mutable_gpu_data()`;

功能: 返回数据在cpu的指针, 并改变数据的状态为`HEAD_AT_GPU`

```
void*SyncedMemory::mutable_gpu_data() {  
    to_gpu();  
    head_=HEAD_AT_GPU;  
    returngpu_ptr_;  
}
```

函数`SyncedHeadhead()` {returnhead_;}

功能: 返回数据的状态

函数`size_tsize()` {returnsize_;}

功能: 返回数据的大小

`DISABLE_COPY_AND_ASSIGN(SyncedMemory);`

一个宏, 把该类的拷贝函数和等号操作符给禁止掉

其实就是

```
private:\n\nSyncedMemory(constSyncedMemory&);\n\nSyncedMemory&operator=(constSyncedMemory&)
```

如果你想让你的类不能使用copy构造函数和赋值操作符，只要将该类的copy构造函数和赋值操作符函数定义为private即可，并且只是声明，不用实现.

版权声明：本文为博主原创文章，未经博主允许不得转载。

上一篇

caffe源码分析--math_functions.cu代码研究

下一篇

给cuda核函数传递二维数组的一种方法

主题推荐

函数 cuda c语言 color 代码 源码 class 构造函数 内存

猜你在找

- Spark 1.x大数据平台
- 3D游戏开发基础
- 3D游戏引擎之GPU渲染（DX篇）
- Android入门实战教程
- C语言及程序设计提高
- Spring源码学习-容器初始化之
- 从汇编代码学习C++语言1类对象构造函数与析构函数
- java语言基础之4种代码块以及构造函数比较大集合-
- C++类对象的复制-拷贝构造函数The c++class
- 《ASCE1885的源码分析》のWM_代码转字符串表示的函数

准备好了么？跳吧！

更多职位尽在 CSDN JOB

数据分析工程师	我要跳槽	高级商业数据分析师	我要跳槽
腾讯科技（深圳）有限公司	20~40K/月	上海点我吧信息技术有限公司	20~40K/月
数据分析师---SQL	我要跳槽	数据挖掘 / 数据分析工程师	我要跳槽
欧唯特信息服务有限公司	6~9K/月	上海智子信息科技有限公司	8~16K/月

查看评论

您还没有登录,请[登录](#)或[注册](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery
BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity
Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack
FTC coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide Maemo
Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP HBase Pure Solr
Angular Cloud Foundry Redis Scala Django Bootstrap

