

# 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： 219485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--data\_layer.cpp

分类： 深度学习 (deep learning)

2014-05-31 21:24

5375人阅读

评论(5) 收藏 举报

神经网络 caffe 深度学习 deep learning

dataLayer作为整个网络

数据从leveldb中取。leveldb数据是从图片转换过来的。

网络建立的时候，

datalayer主要是负责设置一些参数，比如batchsize，channels，height，width等。

这次会通过读leveldb一个数据块来获取这些信息。

然后启动一个线程来预先从leveldb拉取一批数据，这些数据是图像数据和图像标签。

正向传播的时候，

datalayer就把预先拉取好数据拷贝到指定的cpu或者gpu的内存。

然后启动新线程再预先拉取数据，这些数据留到下一次正向传播使用。

```
[cpp]
01. // Copyright 2013 Yangqing Jia
02.
03. #include <stdint.h>
04. #include <leveldb/db.h>
05. #include <pthread.h>
06.
07. #include <string>
08. #include <vector>
09.
10. #include "caffe/layer.hpp"
11. #include "caffe/util/io.hpp"
12. #include "caffe/vision_layers.hpp"
13.
14. using std::string;
15.
16. namespace caffe {
17.
18. template <typename Dtype>
19. void* DataLayerPrefetch(void* layer_pointer) {
20.     CHECK(layer_pointer);
21.     DataLayer<Dtype>* layer = reinterpret_cast<DataLayer<Dtype>*>(layer_pointer);
22.     CHECK(layer);
23.     Datum datum;
24.     CHECK(layer->prefetch_data_);
25.     Dtype* top_data = layer->prefetch_data_->mutable_cpu_data();//数据
26.     Dtype* top_label = layer->prefetch_label_->mutable_cpu_data();//标签
27.     const Dtype scale = layer->layer_param_.scale();
28.     const int batchsize = layer->layer_param_.batchsize();
29.     const int cropsizesize = layer->layer_param_.cropsizesize();
```

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: caffe特征可视化的代  
fqss0436: 博主，您好，感谢您  
分享代码。在调试您的代码时，  
程序中中断于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，能说一下测试单  
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  
wwdzhxbnjwcnmd: 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 (4386)  
deep learning实践经验总结 (4225)

推荐文章

```
30.     const bool mirror = layer->layer_param_.mirror();
31.
32.     if (mirror && cropsize == 0) { //当前实现需要同时设置mirror和cropsize
33.         LOG(FATAL) << "Current implementation requires mirror and cropsize to be "
34.             << "set at the same time.";
35.     }
36.     // datum scales
37.     const int channels = layer->datum_channels_;
38.     const int height = layer->datum_height_;
39.     const int width = layer->datum_width_;
40.     const int size = layer->datum_size_;
41.     const Dtype* mean = layer->data_mean_.cpu_data();
42.     for (int itemid = 0; itemid < batchsize; ++itemid) { //每一批数据的数量是batchsize，一个循环拉取一
张？
43.         // get a blob
44.         CHECK(layer->iter_);
45.         CHECK(layer->iter_->Valid());
46.         datum.ParseFromString(layer->iter_->value().ToString()); //利用迭代器拉取下一批数据
47.         const string& data = datum.data();
48.         if (cropsize) { //如果需要裁剪
49.             CHECK(data.size()) << "Image cropping only support uint8 data";
50.             int h_off, w_off;
51.             // We only do random crop when we do training.
52.             // 只是在训练阶段做随机裁剪
53.             if (Caffe::phase() == Caffe::TRAIN) {
54.                 // NOLINT_NEXT_LINE(runtime/threadsafe_fn)
55.                 h_off = rand() % (height - cropsize);
56.                 // NOLINT_NEXT_LINE(runtime/threadsafe_fn)
57.                 w_off = rand() % (width - cropsize);
58.             } else { //测试阶段固定裁剪
59.                 h_off = (height - cropsize) / 2;
60.                 w_off = (width - cropsize) / 2;
61.             }
62.             // NOLINT_NEXT_LINE(runtime/threadsafe_fn)
63.             //怎么感觉下面两种情况的代码是一样的？
64.             if (mirror && rand() % 2) {
65.                 // Copy mirrored version
66.                 for (int c = 0; c < channels; ++c) {
67.                     for (int h = 0; h < cropsize; ++h) {
68.                         for (int w = 0; w < cropsize; ++w) {
69.                             top_data[(itemid * channels + c) * cropsize + h) * cropsize
70.                                 + cropsize - 1 - w] =
71.                                 (static_cast<Dtype>)(
72.                                     (uint8_t)data[(c * height + h + h_off) * width
73.                                         + w + w_off])
74.                                     - mean[(c * height + h + h_off) * width + w + w_off])
75.                                     * scale;
76.                         }
77.                     }
78.                 }
79.             } else {
80.                 // Normal copy
81.                 for (int c = 0; c < channels; ++c) {
82.                     for (int h = 0; h < cropsize; ++h) {
83.                         for (int w = 0; w < cropsize; ++w) {
84.                             top_data[(itemid * channels + c) * cropsize + h) * cropsize + w]
85.                                 = (static_cast<Dtype>)(
86.                                     (uint8_t)data[(c * height + h + h_off) * width
87.                                         + w + w_off])
88.                                     - mean[(c * height + h + h_off) * width + w + w_off])
89.                                     * scale;
90.                         }
91.                     }
92.                 }
93.             }
94.         } else { //如果不需要裁剪
95.             // we will prefer to use data() first, and then try float_data()
96.             //我们优先考虑data(), 然后float_data()
97.             if (data.size()) {
98.                 for (int j = 0; j < size; ++j) {
99.                     top_data[itemid * size + j] =
100.                         (static_cast<Dtype>)((uint8_t)data[j]) - mean[j] * scale;
101.                 }
102.             } else {
103.                 for (int j = 0; j < size; ++j) {
104.                     top_data[itemid * size + j] =
105.                         (datum.float_data(j) - mean[j]) * scale;
106.                 }
107.             }
108.         }
109.     }
110. }
```

```

108.     }
109.
110.     top_label[itemid] = datum.label();
111.     // go to the next iter
112.     layer->iter_->Next();
113.     if (!layer->iter_->Valid()) {
114.         // We have reached the end. Restart from the first.
115.         DLOG(INFO) << "Restarting data prefetching from start.";
116.         layer->iter_->SeekToFirst();
117.     }
118. }
119.
120. return reinterpret_cast<void*>(NULL);
121. }
122.
123. template <typename Dtype>
124. DataLayer<Dtype>::~DataLayer<Dtype>() {
125.     // Finally, join the thread
126.     CHECK(!pthread_join(thread_, NULL)) << "Pthread joining failed.";
127. }
128.
129. template <typename Dtype>
130. void DataLayer<Dtype>::SetUp(const vector<Blob<Dtype>*>& bottom,
131.     vector<Blob<Dtype>*>* top) {
132.     CHECK_EQ(bottom.size(), 0) << "Data Layer takes no input blobs.";
133.     CHECK_EQ(top->size(), 2) << "Data Layer takes two blobs as output.";
134.     // Initialize the leveldb
135.     leveldb::DB* db_temp;
136.     leveldb::Options options;
137.     options.create_if_missing = false;
138.     options.max_open_files = 100;
139.     LOG(INFO) << "Opening leveldb " << this->layer_param_.source();
140.     leveldb::Status status = leveldb::DB::Open(
141.         options, this->layer_param_.source(), &db_temp);
142.     CHECK(status.ok()) << "Failed to open leveldb "
143.         << this->layer_param_.source() << std::endl << status.ToString();
144.     db_.reset(db_temp);
145.     iter_.reset(db_->NewIterator(leveldb::ReadOptions())); //通过迭代器来操纵leveldb
146.     iter_->SeekToFirst();
147.     // Check if we would need to randomly skip a few data points
148.     // 是否要随机跳过一些数据
149.     if (this->layer_param_.rand_skip()) {
150.         // NOLINT_NEXT_LINE(runtime/threadsafe_fn)
151.         unsigned int skip = rand() % this->layer_param_.rand_skip();
152.         LOG(INFO) << "Skipping first " << skip << " data points.";
153.         while (skip-- > 0) { //循环次数
154.             iter_->Next();
155.             if (!iter_->Valid()) {
156.                 iter_->SeekToFirst();
157.             }
158.         }
159.     }
160.     // Read a data point, and use it to initialize the top blob.
161.     // 读取一个数据点, 用来初始化topblob. 所谓初始化, 只要是指reshape.
162.     // 可以观察到下面iter_调用调用next. 所以这次读取只是用来读取出来channels等参数的, 不作处理.
163.     Datum datum;
164.     datum.ParseFromString(iter_->value().ToString()); //利用迭代器读取第一个数据点
165.     // image图像数据
166.     int cropsizesize = this->layer_param_.cropsizesize(); //裁剪大小
167.     if (cropsizesize > 0) { //需要裁剪
168.         (*top)[0]->Reshape(
169.             this->layer_param_.batchsize(), datum.channels(), cropsizesize, cropsizesize);
170.         prefetch_data_.reset(new Blob<Dtype>(
171.             this->layer_param_.batchsize(), datum.channels(), cropsizesize, cropsizesize));
172.     } else { //不需要裁剪
173.         (*top)[0]->Reshape(
174.             this->layer_param_.batchsize(), datum.channels(), datum.height(),
175.             datum.width());
176.         prefetch_data_.reset(new Blob<Dtype>(
177.             this->layer_param_.batchsize(), datum.channels(), datum.height(),
178.             datum.width()));
179.     }
180.     LOG(INFO) << "output data size: " << (*top)[0]->num() << ", "
181.         << (*top)[0]->channels() << ", " << (*top)[0]->height() << ", "
182.         << (*top)[0]->width();
183.     // label标签数据
184.     (*top)[1]->Reshape(this->layer_param_.batchsize(), 1, 1, 1);
185.     prefetch_label_.reset(
186.         new Blob<Dtype>(this->layer_param_.batchsize(), 1, 1, 1));

```

```

187. // datum size
188. datum_channels_ = datum.channels();
189. datum_height_ = datum.height();
190. datum_width_ = datum.width();
191. datum_size_ = datum.channels() * datum.height() * datum.width();
192. CHECK_GT(datum_height_, cropsiz);
193. CHECK_GT(datum_width_, cropsiz);
194. // check if we want to have mean是否要减去均值
195. if (this->layer_param_.has_meanfile()) {
196.   BlobProto blob_proto;
197.   LOG(INFO) << "Loading mean file from" << this->layer_param_.meanfile();
198.   ReadProtoFromBinaryFile(this->layer_param_.meanfile().c_str(), &blob_proto);
199.   data_mean_.FromProto(blob_proto);
200.   CHECK_EQ(data_mean_.num(), 1);
201.   CHECK_EQ(data_mean_.channels(), datum_channels_);
202.   CHECK_EQ(data_mean_.height(), datum_height_);
203.   CHECK_EQ(data_mean_.width(), datum_width_);
204. } else {
205.   // Simply initialize an all-empty mean.
206.   data_mean_.Reshape(1, datum_channels_, datum_height_, datum_width_);
207. }
208. // Now, start the prefetch thread. Before calling prefetch, we make two
209. // cpu_data calls so that the prefetch thread does not accidentally make
210. // simultaneous cudaMalloc calls when the main thread is running. In some
211. // GPUs this seems to cause failures if we do not so.
212. prefetch_data_->mutable_cpu_data();
213. prefetch_label_->mutable_cpu_data();
214. data_mean_.cpu_data();
215. DLOG(INFO) << "Initializing prefetch";
216. CHECK(!pthread_create(&thread_, NULL, DataLayerPrefetch<Dtype>,
217.   reinterpret_cast<void*>(this))) << "Pthread execution failed.";
218. DLOG(INFO) << "Prefetch initialized.";
219. }
220.
221. template <typename Dtype>
222. void DataLayer<Dtype>::Forward_cpu(const vector<Blob<Dtype>*>& bottom,
223.   vector<Blob<Dtype>*>* top) {
224.   // First, join the thread 等待线程结束
225.   CHECK(!pthread_join(thread_, NULL)) << "Pthread joining failed.";
226.   // Copy the data拷贝数据到top, 即该层的输出
227.   memcpy((*top)[0]->mutable_cpu_data(), prefetch_data_->cpu_data(),
228.     sizeof(Dtype) * prefetch_data_->count());
229.   memcpy((*top)[1]->mutable_cpu_data(), prefetch_label_->cpu_data(),
230.     sizeof(Dtype) * prefetch_label_->count());
231.   // Start a new prefetch thread启动新线程拉取下一批数据
232.   CHECK(!pthread_create(&thread_, NULL, DataLayerPrefetch<Dtype>,
233.     reinterpret_cast<void*>(this))) << "Pthread execution failed.";
234. }
235.
236. // The backward operations are dummy - they do not carry any computation.
237. template <typename Dtype>
238. Dtype DataLayer<Dtype>::Backward_cpu(const vector<Blob<Dtype>*>& top,
239.   const bool propagate_down, vector<Blob<Dtype>*>* bottom) {
240.   return Dtype(0.);
241. }
242.
243. INSTANTIATE_CLASS(DataLayer);
244.
245. } // namespace caffe

```

本文作者: linger

本文链接: <http://blog.csdn.net/lingerlanlan/article/details/27348265>

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 全局内存性能测试

下一篇 使用squid架设自己的代理服务器

顶 0

踩 0

主题推荐

源码 网络 线程 图片 内存 标签 数据 博客 class gpu pre

猜你在找

- 韦东山嵌入式Linux第一期视频

HTML 5移动开发从入门到精通

C语言及程序设计初步

Android入门实战教程

零基础学HTML 5实战开发(第一季)
- caffe源码解析 netcpp

caffe源码解析 blobcpp

caffe源码修改抽取任意一张图片的特征

caffe源码修改抽取任意一张图片的特征

蔡军生先生第二人生的源码分析四十六获取纹理图片的

准备好了么？跳吧！

更多职位尽在 CSDN JOB

需求分析	我要跳槽	数据分析工程师	我要跳槽
河北家乐园信息技术服务有限公司	4-7K/月	腾讯科技（深圳）有限公司	20-40K/月
高级商业数据分析师	我要跳槽	数据分析师---SQL	我要跳槽
上海点我吧信息技术有限公司	20-40K/月	欧唯特信息服务有限公司	6-9K/月

查看评论

4楼 沧海1梦 2015-07-08 10:08发表



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，还没找到修改输入的地方

3楼 进击的城管 2015-01-29 11:10发表



ou no``caffe更新了 我的心好痛。。。代码变了

2楼 houqiqi 2014-11-09 23:55发表



博主你好，我想问下你是用什么来查看caffe的源代码的，我用gedit感觉效率太低了，查看一个调用函数都要自己去猜这个函数可能在什么位置，大概在哪个文件下~~~能将您的配置说一下吗

Re: lingerlanlan 2014-11-13 00:34发表



回复houqiqi: 我在Ubuntu装了cuda之后，会有一个叫insight的ide，其实是Eclipse的扩展版本。

1楼 barbara2008 2014-07-15 10:24发表



top\_data[((itemid \* channels + c) \* cropsiz + h) \* cropsiz + cropsiz - 1 - w]和84行的index不一样

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

