caffe神经网络框架的辅助工具（将图片转换为leveldb格式）

分类： 深度学习（deep learning） 工具源码 2014-06-26 20:05 6148人阅读 评论(37) 收藏 举报

caffe 深度学习 神经网络 计算机视觉

caffe中负责整个网络输入的datalayer是从leveldb里读取数据的，是一个google实现的非常高效的kv数据库。

因此我们训练网络必须先把数据转成leveldb的格式。

这里我实现的是把一个文件夹的所有图片转成leveldb的格式。

工具使用命令格式：convert_imagedata src_dir dst_dir attach_dir channel width height

样例：./convert_imagedata.bin /home/linger/imdata/collar_train/ /home/linger/linger/testfile/crop_train_db/ /home/linger/linger/testfile/crop_train_attachment/ 3 50 50

源代码：

```
#include <google/protobuf/text_format.h>
#include <glog/logging.h>
#include <leveldb/db.h>

#include <stdint.h>
#include <fstream>  // NOLINT(readability/streams)
#include <string>
#include <set>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <dirent.h>
#include <sys/stat.h>
#include <unistd.h>
#include <sys/types.h>
#include "caffe/proto/caffe.pb.h"
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/highgui/highgui_c.h>
#include <opencv2/imgproc/imgproc.hpp>

using std::string;
using namespace std;



set<string> all_class_name;
map<string,int> class2id;
```

```
/**
 * path:目录
 * files：用于保存文件名的vector
 * r：是否需要遍历子目录
 * return:文件名，不包含路径
 */
void list_dir(const char *path,vector<string>& files,bool r = false)
{
        DIR *pDir;
        struct dirent *ent;
        char childpath[512];
        pDir = opendir(path);
        memset(childpath, 0, sizeof(childpath));
        while ((ent = readdir(pDir)) != NULL)
        {
                if (ent->d_type & DT_DIR)
                {

                        if (strcmp(ent->d_name, ".") == 0 || strcmp(ent->d_name, "..") == 0)
                        {
                                continue;
                        }
                        if(r) //如果需要遍历子目录
                        {
                                sprintf(childpath, "%s/%s", path, ent->d_name);
                                list_dir(childpath,files);
                        }
                }
                else
                {
                        files.push_back(ent->d_name);
                }
        }
        sort(files.begin(),files.end());//排序

}

string get_classname(string path)
{
        int index = path.find_last_of('_');
        return path.substr(0, index);
}



int get_labelid(string fileName)
{
        string class_name_tmp = get_classname(fileName);
        all_class_name.insert(class_name_tmp);
        map<string,int>::iterator name_iter_tmp = class2id.find(class_name_tmp);
        if (name_iter_tmp == class2id.end())
        {
                int id = class2id.size();
                class2id.insert(name_iter_tmp, std::make_pair(class_name_tmp, id));
                return id;
        }
        else
        {
                return name_iter_tmp->second;
        }
}

void loadimg(string path,char* buffer)
{
        cv::Mat img = cv::imread(path, CV_LOAD_IMAGE_COLOR);
        string val;
        int rows = img.rows;
        int cols = img.cols;
        int pos=0;
        for (int c = 0; c < 3; c++)
        {
                for (int row = 0; row < rows; row++)
                {
                        for (int col = 0; col < cols; col++)
                        {
                                buffer[pos++]=img.at<cv::Vec3b>(row,col)[c];
                        }
                }
        }

}
void convert(string imgdir,string outputdb,string attachdir,int channel,int width,int height)
{
        leveldb::DB* db;
        leveldb::Options options;
        options.create_if_missing = true;
        options.error_if_exists = true;
        caffe::Datum datum;
        datum.set_channels(channel);
        datum.set_height(height);
        datum.set_width(width);
        int image_size = channel*width*height;
        char buffer[image_size];

        string value;
        CHECK(leveldb::DB::Open(options, outputdb, &db).ok());
        vector<string> filenames;
        list_dir(imgdir.c_str(),filenames);
        string img_log = attachdir+"image_filename";
        ofstream writefile(img_log.c_str());
        for(int i=0;i<filenames.size();i++)
        {
                string path= imgdir;
                path.append(filenames[i]);//算出绝对路径

                loadimg(path,buffer);

                int labelid = get_labelid(filenames[i]);
                datum.add_label(labelid);
                datum.set_data(buffer,image_size);
                datum.SerializeToString(&value);
                snprintf(buffer, image_size, "%05d", i);
                printf("\nclassid:%d classname:%s
abspath:%s",labelid,get_classname(filenames[i]).c_str(),path.c_str());
                db->Put(leveldb::WriteOptions(),string(buffer),value);
                //printf("%d %s\n",i,fileNames[i].c_str());

                assert(writefile.is_open());
                writefile<<i<<" "<<filenames[i]<<"\n";

        }
        delete db;
        writefile.close();

        img_log = attachdir+"image_classname";
        writefile.open(img_log.c_str());
        set<string>::iterator iter = all_class_name.begin();
        while(iter != all_class_name.end())
        {
                assert(writefile.is_open());
                writefile<<(*iter)<<"\n";
                //printf("%s\n",(*iter).c_str());
                iter++;
        }
        writefile.close();

}

int main(int argc, char** argv)
{
        if (argc < 6)
        {
            LOG(ERROR) << "convert_imagedata src_dir dst_dir attach_dir channel width height";
            return 0;
        }
//./convert_imagedata.bin  /home/linger/imdata/collarTest/ /home/linger/linger/testfile/dbtest/
/home/linger/linger/testfile/test_attachment/ 3 250 250
        //   ./convert_imagedata.bin /home/linger/imdata/collar_train/
/home/linger/linger/testfile/crop_train_db/ /home/linger/linger/testfile/crop_train_attachment/ 3 50
50
        google::InitGoogleLogging(argv[0]);
        string src_dir = argv[1];
        string src_dst = argv[2];
        string attach_dir = argv[3];
        int channel = atoi(argv[4]);
        int width = atoi(argv[5]);
        int height = atoi(argv[6]);

        //for test
        /*
        src_dir = "/home/linger/imdata/collarTest/";
        src_dst = "/home/linger/linger/testfile/dbtest/";
        attach_dir = "/home/linger/linger/testfile/";
        channel = 3;
        width = 250;
        height = 250;
         */

        convert(src_dir,src_dst,attach_dir,channel,width,height);




}
```

上一篇 广告贴
下一篇 致家驹--祝你愉快