网络资源是无限的

个人资料

fengbingchun

访问： 2252590次
积分： 25003
等级： BLOG 7
排名： 第202名

原创： 341篇 转载： 144篇
译文： 0篇 评论： 1434条

文章分类

Android (9)
ActiveX (18)
Bar Code (16)
Caffe (20)
C# (5)
CImg (4)
Contour Detection (9)
CxImage (6)
Code::Blocks (3)
Cloud Computing (1)
C/C++ (82)
CUDA (10)
CMake (3)
Design Patterns (25)
Database/Dataset (4)
Deep Learning (9)
Eclipse (3)
Emgu CV (1)
Eigen (1)
FFmpeg (1)
Feature Extraction (1)
FreeType (1)
Face (8)
GPU (3)
Git (3)
GCC (1)
GDAL (5)

深度学习开源库tiny-dnn的使用(MNIST)

2016-12-04 13:37 467人阅读 评论(0) 收藏 举报

本文章已收录于： 深度学习知识库

分类： Caffe (19) Deep Learning (8) Neural Network (12)

tiny-dnn是一个基于DNN的深度学习开源库，它的License是BSD 3-Clause。之前名字是tiny-cnn是基于CNN的，tiny-dnn与tiny-cnn相关又增加了些新层。此开源库很活跃，几乎每天都有新的提交，因此下面详细介绍下tiny-dnn在windows7 64bit vs2013的编译及使用。

1. 从https://github.com/tiny-dnn/tiny-dnn 下载源码：

$ git clone https://github.com/tiny-dnn/tiny-dnn.git 版本号为6281c1b，更新日期2016.12.03

2. 源文件中已经包含了vs2013工程，vc/vc12/tiny-dnn.sln，默认是win32的，这里新建一个x64的控制台工程tiny-dnn；

3. 仿照源工程，将相应.h文件加入到新控制台工程中，新加一个test_tiny-dnn.cpp文件；

4. 仿照examples/mnist中test.cpp和train.cpp文件中的代码添加测试代码；

[cpp]

```cpp
#include "funset.hpp"
#include <string>
#include <algorithm>
#include "tiny_dnn/tiny_dnn.h"

static void construct_net(tiny_dnn::network<tiny_dnn::sequential>& nn)
{
    // connection table [Y.Lecun, 1998 Table.1]
#define O true
#define X false
    static const bool tbl[] = {
        O, X, X, X, O, O, O, X, X, O, O, O, O, X, O, O,
        O, O, X, X, X, O, O, O, X, X, O, O, O, O, X, O,
        O, O, O, X, X, X, O, O, O, X, X, O, X, O, O, O,
        X, O, O, O, X, X, O, O, O, O, X, X, O, X, O, O,
        X, X, O, O, O, X, X, O, O, O, O, X, O, O, X, O,
        X, X, X, O, O, O, X, X, O, O, O, O, X, O, O, O
    };
#undef O
#undef X

    // by default will use backend_t::tiny_dnn unless you compiled
    // with -DUSE_AVX=ON and your device supports AVX intrinsics
    tiny_dnn::core::backend_t backend_type = tiny_dnn::core::default_engine();

    // construct nets: C: convolution; S: sub-sampling; F: fully connected
    nn << tiny_dnn::convolutional_layer<tiny_dnn::activation::tan_h>(32, 32, 5, 1, 6,  //
        tiny_dnn::padding::valid, true, 1, 1, backend_type)
```

HTML (3)
Image Recognition (8)
Image Processing (18)
Image Registration (13)
ImageMagick (3)
Java (5)
Linux (20)
Log (2)
Makefile (2)
Mathematical Knowledge (6)
Multi-thread (4)
Matlab (33)
MFC (8)
MinGW (3)
Mac (1)
Neural Network (13)
OCR (9)
Office (2)
OpenCL (2)
OpenSSL (7)
OpenCV (86)
OpenGL (2)
OpenGL ES (3)
OpenMP (3)
Photoshop (1)
Python (4)
Qt (1)
SIMD (14)
Software Development (4)
System architecture (2)
Skia (1)
SVN (1)
Software Testing (4)
Shell (2)
Socket (3)
Target Detection (2)
Target Tracking (2)
VC6 (6)
VS2008 (16)
VS2010 (4)
VS2013 (3)
vigra (2)
VLC (5)
VLFeat (1)
wxWidgets (1)
Watermark (4)
Windows7 (6)
Windows Core Programming (9)
XML (2)

Free Codes

pudn
freecode
Peter's Functions
CodeProject
SourceCodeOnline
Computer Vision Source Code
Codesoso
Digital Watermarking
SourceForge
HackChina
oschina

```
        << tiny_dnn::average_pooling_layer<tiny_dnn::activation::tan_h>(28, 28, 6, 2)   /
        << tiny_dnn::convolutional_layer<tiny_dnn::activation::tan_h>(14, 14, 5, 6, 16, /
        connection_table(tbl, 6, 16),
        tiny_dnn::padding::valid, true, 1, 1, backend_type)
        << tiny_dnn::average_pooling_layer<tiny_dnn::activation::tan_h>(10, 10, 16, 2)  /
        << tiny_dnn::convolutional_layer<tiny_dnn::activation::tan_h>(5, 5, 5, 16, 120, /
        tiny_dnn::padding::valid, true, 1, 1, backend_type)
        << tiny_dnn::fully_connected_layer<tiny_dnn::activation::tan_h>(120, 10,        /
        true, backend_type);
}

static void train_lenet(const std::string& data_dir_path)
{
    // specify loss-function and learning strategy
    tiny_dnn::network<tiny_dnn::sequential> nn;
    tiny_dnn::adagrad optimizer;

    construct_net(nn);

    std::cout << "load models..." << std::endl;

    // load MNIST dataset
    std::vector<tiny_dnn::label_t> train_labels, test_labels;
    std::vector<tiny_dnn::vec_t> train_images, test_images;

    tiny_dnn::parse_mnist_labels(data_dir_path + "/train-labels.idx1-ubyte", &train_labels);
    tiny_dnn::parse_mnist_images(data_dir_path + "/train-images.idx3-ubyte", &train_images, -1.0, 1.0, 2, 2);
    tiny_dnn::parse_mnist_labels(data_dir_path + "/t10k-labels.idx1-ubyte", &test_labels);
    tiny_dnn::parse_mnist_images(data_dir_path + "/t10k-images.idx3-ubyte", &test_images, -1.0, 1.0, 2, 2);

    std::cout << "start training" << std::endl;

    tiny_dnn::progress_display disp(static_cast<unsigned long>(train_images.size()));
    tiny_dnn::timer t;
    int minibatch_size = 10;
    int num_epochs = 30;

    optimizer.alpha *= static_cast<tiny_dnn::float_t>(std::sqrt(minibatch_size));

    // create callback
    auto on_enumerate_epoch = [&](){
        std::cout << t.elapsed() << "s elapsed." << std::endl;
        tiny_dnn::result res = nn.test(test_images, test_labels);
        std::cout << res.num_success << "/" << res.num_total << std::endl;

        disp.restart(static_cast<unsigned long>(train_images.size()));
        t.restart();
    };

    auto on_enumerate_minibatch = [&](){
        disp += minibatch_size;
    };

    // training
    nn.train<tiny_dnn::mse>(optimizer, train_images, train_labels, minibatch_size, num_epochs, on_enumerate_minibatc

    std::cout << "end training." << std::endl;

    // test and show results
    nn.test(test_images, test_labels).print_detail(std::cout);

    // save network model & trained weights
    nn.save(data_dir_path + "/LeNet-model");
}

// rescale output to 0-100
template <typename Activation>
static double rescale(double x)
{
    Activation a;
    return 100.0 * (x - a.scale().first) / (a.scale().second - a.scale().first);
}

static void convert_image(const std::string& imagefilename, double minv, double maxv, int
```

关闭

libsvm
joys99
CodeForge
cvchina
tesseract-ocr
sift
TiRG
imgSeek
OpenSURF

Friendly Link

OpenCL
Python
poesia-filter
TortoiseSVN
imgSeek
Notepad
Beyond Compare
CMake
VIGRA
CodeGuru
vchome
aforgenet
CVLAB
Doxygen
Coursera
OpenMP

Technical Forum

Matlab China
OpenCV China
The CImg Library
Open Computer Vision Library
CxImage
ImageMagick
ImageMagick China
OpenCV_China
Subversion China

```
{
    tiny_dnn::image<> img(imagefilename, tiny_dnn::image_type::grayscale);
    tiny_dnn::image<> resized = resize_image(img, w, h);

    // mnist dataset is "white on black", so negate required
    std::transform(resized.begin(), resized.end(), std::back_inserter(data),
        [=](uint8_t c) { return (255 - c) * (maxv - minv) / 255.0 + minv; });
}

int test_dnn_mnist_train()
{
    std::string data_dir_path = "E:/GitCode/NN_Test/data";
    train_lenet(data_dir_path);

    return 0;
}

int test_dnn_mnist_predict()
{
    std::string model { "E:/GitCode/NN_Test/data/LeNet-model" };
    std::string image_path { "E:/GitCode/NN_Test/data/images/"};
    int target[10] { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 };

    tiny_dnn::network<tiny_dnn::sequential> nn;
    nn.load(model);

    for (int i = 0; i < 10; i++) {
        std::string str = std::to_string(i);
        str += ".png";
        str = image_path + str;

        // convert imagefile to vec_t
        tiny_dnn::vec_t data;
        convert_image(str, -1.0, 1.0, 32, 32, data);

        // recognize
        auto res = nn.predict(data);
        std::vector<std::pair<double, int> > scores;

        // sort & print top-3
        for (int j = 0; j < 10; j++)
            scores.emplace_back(rescale<tiny_dnn::tan_h>(res[j]), j);

        std::sort(scores.begin(), scores.end(), std::greater<std::pair<double, int>>());

        for (int j = 0; j < 3; j++)
            fprintf(stdout, "%d: %f;  ", scores[j].second, scores[j].first);
        fprintf(stderr, "\n");

        // save outputs of each layer
        for (size_t j = 0; j < nn.depth(); j++) {
            auto out_img = nn[j]->output_to_image();
            auto filename = image_path + std::to_string(i) + "_layer_" + std::to_string(j
            out_img.save(filename);
        }

        // save filter shape of first convolutional layer
        auto weight = nn.at<tiny_dnn::convolutional_layer<tiny_dnn::tan_h>>
(0).weight_to_image();
        auto filename = image_path + std::to_string(i) + "_weights.png";
        weight.save(filename);

        fprintf(stdout, "the actual digit is: %d, correct digit is: %d \n\n", scores[0].s
    }

    return 0;
}
```

关闭

5. 运行程序，train时，运行结果如下图所示，准确率达到99%以上：

Technical Blog

邹宇华
深之JohnChen
HUNNISH
周伟明
superdont
carson2005
OpenHero
Netman(Linux)
wqvbjhc
yang_xian521
gnuhpc
gnuhpc
千里8848
CVART
tornadomeet
gotosuc
onezeros
hellogv
abcjennifer
crzy_sparrow

评论排行

Windows7 32位机上，O	(120)
tiny-cnn开源库的使用(MI	(93)
Ubuntu 14.04 64位机上才	(89)
tesseract-ocr3.02字符识	(63)
Windows7上使用VS201	(47)
tesseract-ocr	(42)
图像配准算法	(41)
Windows 7 64位机上Ope	(36)
OpenCV中resize函数五利	(34)
小波矩特征提取matlab代	(30)

最新评论

Tesseract-OCR 3.04在Windows
fengbingchun: @ilikede:没有密码，那个commit只是提示是从哪个commit fork过来的，无需管那个

Tesseract-OCR 3.04在Windows
ilikede: 问一下，你第一句中的commit的那个密码，怎么用啊

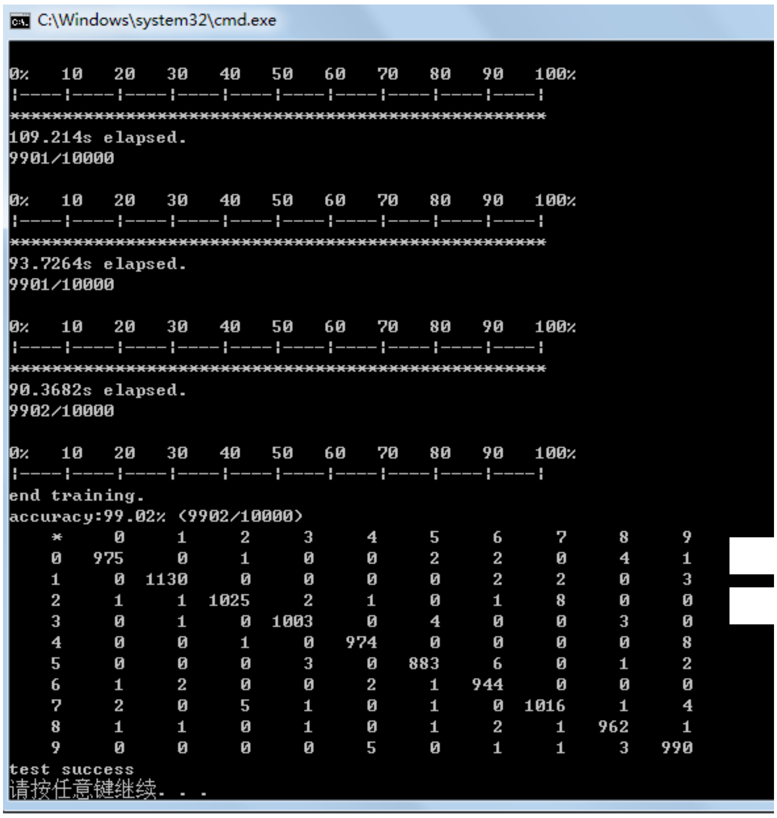
卷积神经网络(CNN)的简单实现(
fengbingchun: @hugl950123:是需要opencv的支持，你在本地opencv的环境配好了吗，配好了就应该没...

卷积神经网络(CNN)的简单实现(
hugl950123: @fengbingchun:博主请问一下，test_CNN_predict()函数是不是需要open...

卷积神经网络(CNN)的简单实现(
hugl950123: @fengbingchun:博主请问一下，test_CNN_predict()函数是不是需要open...

卷积神经网络(CNN)的简单实现(
hugl950123: @fengbingchun:谢谢，能够成功运行了现在

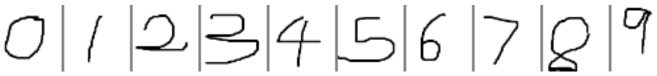
卷积神经网络(CNN)的简单实现(
fengbingchun: @hugl950123:NN中一共有四个工程，它们之间没有任何关系，都是独立的，如果要运行这篇文章的...



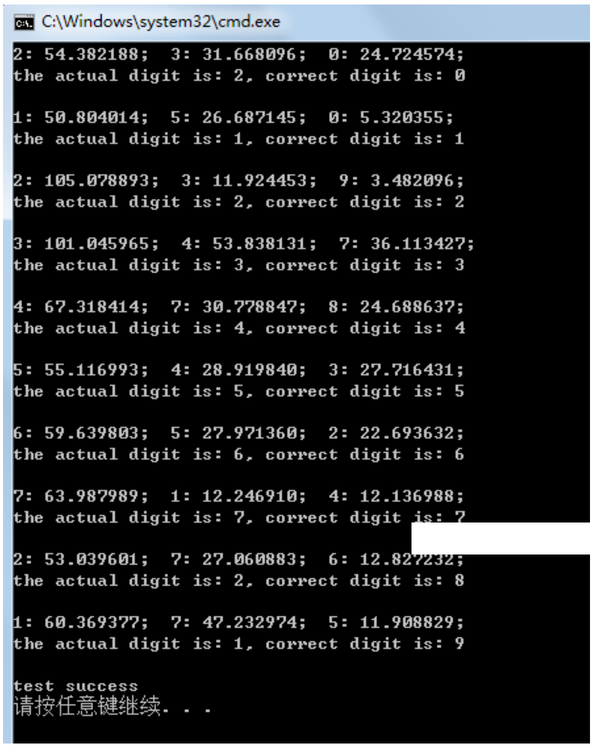
C:\Windows\system32\cmd.exe

```
0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
***************************************************
109.214s elapsed.
9901/10000

0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
***************************************************
93.7264s elapsed.
9901/10000

0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
***************************************************
90.3682s elapsed.
9902/10000

0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
end training.
accuracy:99.02% (9902/10000)
     *     0     1     2     3     4     5     6     7     8     9
     0   975     0     1     0     0     2     2     0     4     1
     1     0  1130     0     0     0     0     2     2     0     3
     2     1     1  1025     2     1     0     1     8     0     0
     3     0     1     0  1003     0     4     0     0     3     0
     4     0     0     1     0   974     0     0     0     0     8
     5     0     0     0     3     0   883     6     0     1     2
     6     1     2     0     0     2     1   944     0     0     0
     7     2     0     5     1     0     1     0  1016     1     4
     8     1     1     0     1     0     1     2     1   962     1
     9     0     0     0     0     5     0     1     1     3   990
test success
请按任意键继续. . .
```

6. 对生成的model进行测试，通过画图工具，每个数字生成一张图像，共10幅，如下图：



7. 通过导入train时生成的model，对这10张图像进行识别，识别结果如下图，其中0,8,9被误识别为2,2,1.



C:\Windows\system32\cmd.exe

```
2: 54.382188;  3: 31.668096;  0: 24.724574;
the actual digit is: 2, correct digit is: 0

1: 50.804014;  5: 26.687145;  0: 5.320355;
the actual digit is: 1, correct digit is: 1

2: 105.078893;  3: 11.924453;  9: 3.482096;
the actual digit is: 2, correct digit is: 2

3: 101.045965;  4: 53.838131;  7: 36.113427;
the actual digit is: 3, correct digit is: 3

4: 67.318414;  7: 30.778847;  8: 24.688637;
the actual digit is: 4, correct digit is: 4

5: 55.116993;  4: 28.919840;  3: 27.716431;
the actual digit is: 5, correct digit is: 5

6: 59.639803;  5: 27.971360;  2: 22.693632;
the actual digit is: 6, correct digit is: 6

7: 63.987989;  1: 12.246910;  4: 12.136988;
the actual digit is: 7, correct digit is: 7

2: 53.039601;  7: 27.060883;  6: 12.827232;
the actual digit is: 2, correct digit is: 8

1: 60.369377;  7: 47.232974;  5: 11.908829;
the actual digit is: 1, correct digit is: 9

test success
请按任意键继续. . .
```

关闭

GitHub：https://github.com/fengbingchun/NN_Test

卷积神经网络(CNN)的简单实现(
hugl950123: @fengbingchun:下的是新的，我在CNN.cpp文件中每个函数都设置了断点，还是没有变化=...

卷积神经网络(CNN)的简单实现(
fengbingchun: @hugl950123:你用的是GitHub上最新的吗？既然能编译过，在Debug下设断点，应该很快...

卷积神经网络(CNN)的简单实现(
hugl950123: 博主，请问我按照您的代码成功编译后执行结果窗口一闪而过，并且里面什么内容也没有，应该如何解决，能不能...

阅读排行

C#中OpenFileDialog的使	(47141)
tesseract-ocr3.02字符识	(34575)
举例说明使用MATLAB C	(25987)
OpenCV中resize函数五种	(24317)
利用cvMinAreaRect2求耳	(24277)
Windows 7 64位机上搭建	(22586)
opencv 检测直线、线段、	(20776)
OpenCV运动检测跟踪(bl	(20475)
图像配准算法	(19237)
有效的rtsp流媒体测试地:	(19143)

文章存档

2017年01月	(18)
2016年12月	(11)
2016年11月	(8)
2016年10月	(7)
2016年09月	(16)

展开

顶 0　　踩 0

上一篇　卷积神经网络(CNN)代码实现(MNIST)解析

下一篇　Dlib库中实现正脸人脸检测的测试代码

我的同类文章

Caffe（19）　Deep Learning（8）　Neural Network（12）

Caffe中Layer注册机制	2017-01-10	阅读 87	windows7下解决caffe check…	2017-01-09	阅读 121
cifar数据集介绍及到图像转…	2016-12-10	阅读 245	卷积神经网络(CNN)代码实…	2016-12-03	阅读 2188
一步一步指引你在Windows…	2016-03-26	阅读 1381	Windows7 64bit VS2013 Ca…	2016-03-26	阅读 1550
卷积神经网络(CNN)的简单…	2016-03-06	阅读 7518	tiny-cnn执行过程分析(MNIST)	2016-01-31	[illegible]
tiny-cnn开源库的使用(MNIST)	2016-01-24	阅读 9464	卷积神经网络(CNN)基础介绍	2016-01-16	阅读 11361

更多文章

猜你在找

VC++Windows多线程实战图片编辑器	京东DNN Lab首席科学家用深度学习搞定80的客服工作
VC++DLL动态链接库编程	京东DNN实验室大数据深度学习与计算平台的实践
C++ 单元测试（GoogleTest）	京东DNN Lab首席科学家用深度学习搞定80的客服工作
VC++游戏开发基础系列从入门到精通	京东DNN Lab首席科学家用深度学习搞定80的客服工作
Windows Server 2012 DHCP Server 管理	深度学习DNNRNNCNN介绍

查看评论

暂无评论

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
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

关闭

公司简介 | 招贤纳士 | 广告服务 | 联系方式 | 版权声明 | 法律顾问 | 问题报告 | 合作伙伴 | 论坛反馈

关闭