

CSDN

博客 (//blog.csdn.net/defnet01mr) 学院 (//edu.csdn.net?ref=toolbar)

下载 (//download.csdn.net?ref=toolbar)    GitChat (//gitbook.cn/?ref=csdn)

更多 ▾

Q



(//write.blog.csdn.net/post) (//write.blog.csdn.net/post) (//write.blog.csdn.net/post)

## 基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）——【三】使用Caffe的MemoryData层与VGG网络模型提取Mat的特征

原创 2016年09月07日 10:19:59

标签：opencv (http://so.csdn.net/so/search/s.do?q=opencv&t=blog) /

计算机视觉 (http://so.csdn.net/so/search/s.do?q=计算机视觉&t=blog) /

深度学习 (http://so.csdn.net/so/search/s.do?q=深度学习&t=blog) /

机器学习 (http://so.csdn.net/so/search/s.do?q=机器学习&t=blog) /

caffe (http://so.csdn.net/so/search/s.do?q=caffe&t=blog)

12465

### 前言

基于深度学习的人脸识别系统，一共用到了5个开源库：OpenCV（计算机视觉库）、Caffe（深度学习库）、Dlib（机器学习库）、libfacedetection（人脸检测库）、cudnn（gpu加速库）。

用到了一个开源的深度学习模型：VGG model。

最终的效果是很赞的，识别一张人脸的速度是0.039秒，而且最重要的是：精度高啊！！！

CPU：intel i5-4590

GPU：GTX 980

系统：Win 10

OpenCV版本：3.1（这个无所谓）

Caffe版本：Microsoft caffe（微软编译的Caffe，安装方便，在这里安利一波）

Dlib版本：19.0（也无所谓

CUDA版本：7.5

cudnn版本：4

libfacedetection：6月份之后的（这个有所谓，6月后出了64位版本的）

这个系列纯C++构成，有问题的各位朋同学可以直接在博客下留言，我们互相交流学习。

本篇是该系列的第三篇博客，介绍如何使用VGG网络模型与Caffe的MemoryData层去提取一个OpenCV矩阵类型Mat的特征。

田牧



开发一个app多少钱



原创	粉丝	喜欢	未开通
28	238	0	(https://github.com/mr_curry)

#### 他的最新文章

更多文章 (http://blog.csdn.net/mr\_curry)

利用Matlab自带的深度学习工具进行车辆区域检测与车型识别【Github更新！！】（三）(http://blog.csdn.net/Mr\_Curry/article/details/68921178)

如何快糙好猛的使用libfacedetection库【最新版】(http://blog.csdn.net/Mr\_Curry/article/details/65945071)

Matlab使用鼠标标注图像位置并返回坐标（标注图像ROI）(http://blog.csdn.net/Mr\_Curry/article/details/54783041)

#### 相关推荐

OpenCV+深度学习预训练模型，简单搞定图像识别 | 教程 (http://blog.csdn.net/tansuo17/article/details/77833932)

Deep Learning（深度学习）之（一）特征以及训练方法 (http://blog.csdn.net/lwq1026/article/details/70208711)

一文看懂迁移学习：怎样用预训练模型搞



开发一个app多少钱



内容举报  
返回顶部



## 心得

VGG网络模型是牛津大学视觉几何组提出的一种深度模型，在LFW数据库上取得了97%的准确率。VGG网络由5个卷积层，两层fc图像特征，一层fc分类特征组成，具体我们可以去读它的prototxt文件。这里是模型与配置文件的下载地址。

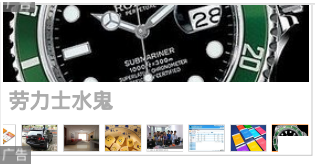
[http://www.robots.ox.ac.uk/~vgg/software/vgg\\_face/](http://www.robots.ox.ac.uk/~vgg/software/vgg_face/) ([http://www.robots.ox.ac.uk/~vgg/software/vgg\\_face/](http://www.robots.ox.ac.uk/~vgg/software/vgg_face/))  
话题回到Caffe。在Caffe中提取图片的特征是很容易的，其提供了extract\_feature.exe让我们来提取，提取格式为lmdb与leveldb。关于这个的做法，可以看看我的这篇博客：  
[http://blog.csdn.net/mr\\_curry/article/details/52097529](http://blog.csdn.net/mr_curry/article/details/52097529) ([http://blog.csdn.net/mr\\_curry/article/details/52097529](http://blog.csdn.net/mr_curry/article/details/52097529))  
显然，我们在程序中肯定是希望能够灵活利用的，使用这种方法不太可行。Caffe的Data层提供了type：MemoryData，我们可以使用它来进行Mat类型特征的提取。

**注：你需要先按照本系列第一篇博客的方法去配置好Caffe的属性表。**  
[http://blog.csdn.net/mr\\_curry/article/details/52443126](http://blog.csdn.net/mr_curry/article/details/52443126) ([http://blog.csdn.net/mr\\_curry/article/details/52443126](http://blog.csdn.net/mr_curry/article/details/52443126))

## 实现

首先我们打开VGG\_FACE\_deploy.prototxt，观察VGG的网络结构。

```
20 }
21
22 layers {
23   bottom: "data"
24   top: "conv1_1"
25   name: "conv1_1"
26   type: CONVOLUTION
27   convolution_param {
28     num_output: 64
29     pad: 1
30     kernel_size: 3
31   }
32 }
33 layers {
34   bottom: "conv1_1"
35   top: "conv1_1"
36   name: "relu1_1"
37   type: RELU
38 }
39 layers {
40   bottom: "conv1_1"
41   top: "conv1_2"
42   name: "conv1_2"
43   type: CONVOLUTION
44   convolution_param {
45     num_output: 64
46     pad: 1
47     kernel_size: 3
48   }
49 }
50 layers {
```



### 博主专栏

深度学习的程序应用  
——Caffe带给我们的可 ...  
15453  
(<http://blog.csdn.net/column/details/13863.html>)

### 在线课程

[http://www.baidu.com/cb.php?c=lgF\\_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H](http://www.baidu.com/cb.php?c=lgF_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H)

[http://www.baidu.com/cb.php?c=lgF\\_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H](http://www.baidu.com/cb.php?c=lgF_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H)

[http://www.baidu.com/cb.php?c=lgF\\_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H](http://www.baidu.com/cb.php?c=lgF_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H)

[http://www.baidu.com/cb.php?c=lgF\\_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H](http://www.baidu.com/cb.php?c=lgF_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H)

[http://www.baidu.com/cb.php?c=lgF\\_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H](http://www.baidu.com/cb.php?c=lgF_pyfqHmknjsnjD0IZ0qnK9uYzP1mznWR10Aw-5y9YIZ0IQzq-0Zr8mLP0B48uqE1AqspvETZ-70TAq15H)



基于深度学习的人脸识别系统系列(Caffe+OpenCV+Dlib) ——【三】使用Caffe的MemoryData层与VGG网络模型提取Mat的特征 - CSDN博客

基于深度学习的人脸识别系统系列(Caffe+OpenCV+Dlib) ——【三】使用Caffe的MemoryData层与VGG网络模型提取Mat的特征 - CSDN博客

```
51     bottom: "conv1_2"
52     top: "conv1_2"
53     name: "relu_2"
54     type: RELU
55 }
56 layers {
57     bottom: "conv1_2"
58     top: "pool1"
59     name: "pool1"
60     type: POOLING
61     pooling_param {
62         pool: MAX
63         kernel_size: 2
64         stride: 2
65     }
66 }
67 layers {
68     bottom: "pool1"
69     top: "conv2_1"
70     name: "conv2_1"
71     type: CONVOLUTION
72     convolution_param {
73         kernel_size: 3
74         stride: 1
75         padding: 1
76     }
77 }
```

有意思的是，MemoryData层需要图像均值，但是官方网站上并没有给出mean文件。我们可以通过这种方式进行输入：

1	mean_value:129.1863
2	mean_value:104.7624
3	mean_value:93.5940

我们还需要修改它的数据层：（你可以用下面这部分的代码去替换下载下来的prototxt文件的数据层）

```
1 layer {
2   name: "data"
3   type: "MemoryData"
4   top: "data"
5   top: "label"
6   transform_param {
7     mirror: false
8     crop_size: 224
9     mean_value:129.1863
10    mean_value:104.7624
11    mean_value:93.5940
12  }
13  memory_data_param {
14    batch_size: 1
15    channels:3
16    height:224
17    width:224
18  }
19 }
```

为了不破坏原来的文件，把它另存为vgg\_extract\_feature\_memorydata.prototxt。

🔊 vgg\_extract\_feature\_memorydata.pro... 2016/8/8 15:30    PROTOTXT 文件    5 KiB

好的，然后我们开始编写。添加好这个属性表：

使用Opencv的dnn模块进行深度学习人脸  
识别（速度较慢）(http://blog.csdn.net/mr\_curry/article/details/52183263)  
📄 9674

⚠️

内容举报

⬆️

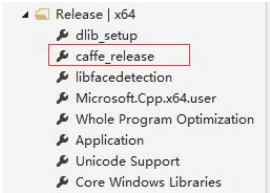
返回顶部



⚠️

内容举报

⬆️  
返回顶部



然后，新建caffe\_net\_memorylayer.h、ExtractFeature\_.h、ExtractFeature\_.cpp开始编写。  
**caffe\_net\_memorylayer.h：**

```
1  #include "caffe/layers/input_layer.hpp"
2  #include "caffe/layers/inner_product_layer.hpp"
3  #include "caffe/layers/dropout_layer.hpp"
4  #include "caffe/layers/conv_layer.hpp"
5  #include "caffe/layers/relu_layer.hpp"
6  #include <iostream>
7  #include "caffe/caffe.hpp"
8  #include <opencv.hpp>
9  #include <caffe/layers/memory_data_layer.hpp>
10 #include "caffe/layers/pooling_layer.hpp"
11 #include "caffe/layers/lrn_layer.hpp"
12 #include "caffe/layers/softmax_layer.hpp"
13 // must predefined
14 caffe::MemoryDataLayer<float> *memory_layer;
15 caffe::Net<float>* net;
```

**ExtractFeature\_.h**

```
1  #include <opencv.hpp>
2  using namespace cv;
3  using namespace std;
4
5  std::vector<float> ExtractFeature(Mat FaceROI);//给一个图片 返回一个vector<float>容器
6  void Caffe_Prefine();
```

**ExtractFeature\_.cpp：**

```
1  #include <ExtractFeature_h>
2  #include <caffe_net_memorylayer.h>
3  namespace caffe
4  {
5  ...
```



⚠️  
内容举报

⬆️  
返回顶部





7



```

5  extern INSTANTIATE_CLASS(InputLayer);
6  extern INSTANTIATE_CLASS(InnerProductLayer);
7  extern INSTANTIATE_CLASS(DropoutLayer);
8  extern INSTANTIATE_CLASS(ConvolutionLayer);
9  REGISTER_LAYER_CLASS(Convolution);
10 extern INSTANTIATE_CLASS(ReLULayer);
11 REGISTER_LAYER_CLASS(ReLU);
12 extern INSTANTIATE_CLASS(PoolingLayer);
13 REGISTER_LAYER_CLASS(Pooling);
14 extern INSTANTIATE_CLASS(LRNLayer);
15 REGISTER_LAYER_CLASS(LRN);
16 extern INSTANTIATE_CLASS(SoftmaxLayer);
17 REGISTER_LAYER_CLASS(Softmax);
18 extern INSTANTIATE_CLASS(MemoryDataLayer);
19 }
20 template <typename Dtype>
21 caffe::Net<Dtype>* Net_Init_Load(std::string param_file, std::string pretrained_param_file, caffe::Phase phase)
22 {
23     caffe::Net<Dtype>* net(new caffe::Net<Dtype>("vgg_extract_feature_memorydata.prototxt", caffe::TEST));
24     net->CopyTrainedLayersFrom("VGG_FACE.caffemodel");
25     return net;
26 }
27
28 void Caffe_Predefine()//when our code begining run must add it
29 {
30     caffe::Caffe::set_mode(caffe::Caffe::GPU);
31     net = Net_Init_Load<float>("vgg_extract_feature_memorydata.prototxt", "VGG_FACE.caffemodel", caffe::TEST);
32     memory_layer = (caffe::MemoryDataLayer<float> *)net->layers()[0].get();
33 }
34
35 std::vector<float> ExtractFeature(Mat FaceROI)
36 {
37     caffe::Caffe::set_mode(caffe::Caffe::GPU);
38     std::vector<Mat> test;
39     std::vector<int> testLabel;
40     std::vector<float> test_vector;
41     test.push_back(FaceROI);
42
43     testLabel.push_back(0);
44     memory_layer->AddMatVector(test, testLabel);// memory_layer and net , must be define be a global variable.
45     test.clear(); testLabel.clear();
46     std::vector<caffe::Blob<float>*> input_vec;
47     net->Forward(input_vec);
48     boost::shared_ptr<caffe::Blob<float>> fc8 = net->blob_by_name("fc8");
49     int test_num = 0;
50     while (test_num < 2622)
51     {
52         test_vector.push_back(fc8->data_at(0, test_num++, 1, 1));
53     }
54     return test_vector;
55 }

```

=====注意上面这个地方可以这么改：=====

( 直接可以知道这个向量的首地址、尾地址，我们直接用其来定义vector )



内容举报



返回顶部



7





```
1 float* begin = nullptr;
2 float* end = nullptr;
3 begin = fc8->mutable_cpu_data();
4 end = begin + fc8->channels();
5 CHECK(begin != nullptr);
6 CHECK(end != nullptr);
7 std::vector<float> FaceVector{ begin,end };
8 return std::move(FaceVector);
```

请特别注意这个地方：

内容举报

返回顶部

7

```
1 namespace caffe
2 {
3     extern INSTANTIATE_CLASS(InputLayer);
4     extern INSTANTIATE_CLASS(InnerProductLayer);
5     extern INSTANTIATE_CLASS(DropoutLayer);
6     extern INSTANTIATE_CLASS(ConvolutionLayer);
7     REGISTER_LAYER_CLASS(Convolution);
8     extern INSTANTIATE_CLASS(ReLULayer);
9     REGISTER_LAYER_CLASS(ReLU);
10    extern INSTANTIATE_CLASS(PoolingLayer);
11    REGISTER_LAYER_CLASS(Pooling);
12    extern INSTANTIATE_CLASS(LRNLayer);
13    REGISTER_LAYER_CLASS(LRN);
14    extern INSTANTIATE_CLASS(SoftmaxLayer);
15    REGISTER_LAYER_CLASS(Softmax);
16    extern INSTANTIATE_CLASS(MemoryDataLayer);
17 }
```

为什么要加这些？因为在提取过程中发现，如果不加，会导致有一些层没有注册的情况。我在Github的Microsoft/Caffe上帮一外国哥们解决了这个问题。我把问题展现一下：



```
layer {
  name: "fc8"
  type: "InnerProduct"
  bottom: "fc7"
  top: "fc8"
  inner_product_param {
    num_output: 2622
  }
}

layer {
  name: "prob"
  type: "Softmax"
  bottom: "fc8"
  top: "prob"
}

[0907 09:52:43.977771 4452 layer_factory.hpp:77] Creating layer data
[0907 09:52:43.978771 4452 layer_factory.hpp:81] Check failed: registry.count(type) == 1 (0 vs. 1) Unknown layer type:
MemoryData (known types: )
*** Check failure stack trace: ***
微软拼音 半;
```

如果我们加了上述代码，就相当于注册了这些层，自然就不会有这样的问題。

**在提取过程中，我提取的是fc8层的特征，2622维。当然，最后一层都已经是分类特征了，最好还是提取fc7层的4096维特征。**

在这个地方：

```
1 void Caffe_Predefine()//when our code begining run must add it
2 {
3     caffe::Caffe::set_mode(caffe::Caffe::GPU);
4     net = Net_Init_Load<float>("vgg_extract_feature_memorydata.prototxt", "VGG_FACE.caffemodel", caffe::TEST);
5     memory_layer = (caffe::MemoryDataLayer<float> *)net->layers()[0].get();
6 }
```

是一个初始化的函数，用于将VGG网络模型与提取特征的配置文件进行传入，所以很明显地，在提取特征之前，需要先：

```
1 Caffe_Predefine();
```

进行了这个之后，这些全局量我们就能一直用了。

我们可以试试提取特征的这个接口。新建一个main.cpp，调用之：

```
1 #include <ExtractFeature_h>
2 int main()
3 {
4     Caffe_Predefine();
5     Mat lena = imread("lena.jpg");
6     if (!lena.empty())
7     {
8         ExtractFeature(lena);
9     }
10
11 }
```

因为我们得到的是一个vector< float>类型，所以我们可以把它逐一输出出来看看。当然，在ExtractFeature()的函数中你就可以这么做了。我们还是在main()函数里这么做。

来看看：



⚠

内容举报

⬆

返回顶部

⚠

内容举报

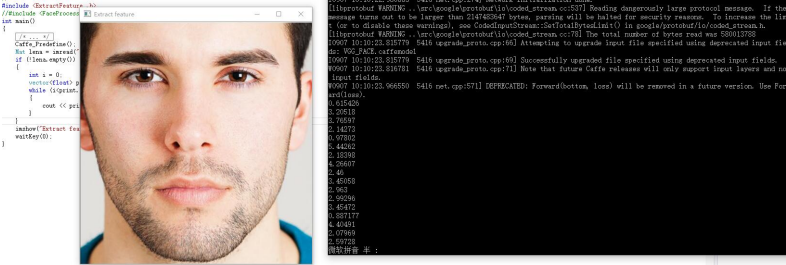
⬆

返回顶部



```
1 #include <ExtractFeature.h>
2 int main()
3 {
4     Caffe_Predefine();
5     Mat lena = imread("lena.jpg");
6     if (!lena.empty())
7     {
8         int i = 0;
9         vector<float> print=ExtractFeature(lena);
10        while (i<print.size())
11        {
12            cout << print[i++] << endl;
13        }
14    }
15    imshow("Extract feature",lena);
16    waitKey(0);
17 }
```

那么对于这张图片，提取出的特征，就是很多的这些数字：



提取一张224\*224图片特征的时间为：0.019s。我们可以看到，GPU加速的效果是非常明显的。而且我这块显卡也就是GTX980。不知道泰坦X的提取速度如何（泪）。

附：net结构（prototxt），注意layer和layers的区别：

```
1 name: "VGG_FACE_16_layer"
2 layer {
3     name: "data"
4     type: "MemoryData"
5     top: "data"
6     top: "label"
7     transform_param {
8         mirror: false
9         crop_size: 224
```







7



```
10   mean_value:129.1863
11   mean_value:104.7624
12   mean_value:93.5940
13 }
14 memory_data_param {
15   batch_size: 1
16   channels:3
17   height:224
18   width:224
19 }
20 }
21 layer {
22   bottom: "data"
23   top: "conv1_1"
24   name: "conv1_1"
25   type: "Convolution"
26   convolution_param {
27     num_output: 64
28     pad: 1
29     kernel_size: 3
30   }
31 }
32 layer {
33   bottom: "conv1_1"
34   top: "conv1_1"
35   name: "relu1_1"
36   type: "ReLU"
37 }
38 layer {
39   bottom: "conv1_1"
40   top: "conv1_2"
41   name: "conv1_2"
42   type: "Convolution"
43   convolution_param {
44     num_output: 64
45     pad: 1
46     kernel_size: 3
47   }
48 }
49 layer {
50   bottom: "conv1_2"
51   top: "conv1_2"
52   name: "relu1_2"
53   type: "ReLU"
54 }
55 layer {
56   bottom: "conv1_2"
57   top: "pool1"
58   name: "pool1"
59   type: "Pooling"
60   pooling_param {
61     pool: MAX
62     kernel_size: 2
```



内容举报

返回顶部



```
63     stride: 2
64   }
65 }
66 layer {
67   bottom: "pool1"
68   top: "conv2_1"
69   name: "conv2_1"
70   type: "Convolution"
71   convolution_param {
72     num_output: 128
73     pad: 1
74     kernel_size: 3
75   }
76 }
77 layer {
78   bottom: "conv2_1"
79   top: "conv2_1"
80   name: "relu2_1"
81   type: "ReLU"
82 }
83
84 layer {
85   bottom: "conv2_1"
86   top: "conv2_2"
87   name: "conv2_2"
88   type: "Convolution"
89   convolution_param {
90     num_output: 128
91     pad: 1
92     kernel_size: 3
93   }
94 }
95 layer {
96   bottom: "conv2_2"
97   top: "conv2_2"
98   name: "relu2_2"
99   type: "ReLU"
100 }
101 layer {
102   bottom: "conv2_2"
103   top: "pool2"
104   name: "pool2"
105   type: "Pooling"
106   pooling_param {
107     pool: MAX
108     kernel_size: 2
109     stride: 2
110   }
111 }
112 layer {
113   bottom: "pool2"
114   top: "conv3_1"
115   name: "conv3_1"
116   type: "Convolution"
```



7



内容举报



返回顶部



```
115 type: "Convolution"
116 convolution_param {
117   num_output: 256
118   pad: 1
119   kernel_size: 3
120 }
121 }
122 layer {
123   bottom: "conv3_1"
124   top: "conv3_1"
125   name: "relu3_1"
126   type: "ReLU"
127 }
128 layer {
129   bottom: "conv3_1"
130   top: "conv3_2"
131   name: "conv3_2"
132   type: "Convolution"
133   convolution_param {
134     num_output: 256
135     pad: 1
136     kernel_size: 3
137   }
138 }
139 layer {
140   bottom: "conv3_2"
141   top: "conv3_2"
142   name: "relu3_2"
143   type: "ReLU"
144 }
145 layer {
146   bottom: "conv3_2"
147   top: "conv3_3"
148   name: "conv3_3"
149   type: "Convolution"
150   convolution_param {
151     num_output: 256
152     pad: 1
153     kernel_size: 3
154   }
155 }
156 layer {
157   bottom: "conv3_3"
158   top: "conv3_3"
159   name: "relu3_3"
160   type: "ReLU"
161 }
162 layer {
163   bottom: "conv3_3"
164   top: "pool3"
165   name: "pool3"
166   type: "Pooling"
```



 内容举报

 返回顶部

 内容举报

 返回顶部



  
7












```
167 pooling_param {
168   pool: MAX
169   kernel_size: 2
170   stride: 2
171 }
172 }
173 layer {
174   bottom: "pool3"
175   top: "conv4_1"
176   name: "conv4_1"
177   type: "Convolution"
178   convolution_param {
179     num_output: 512
180     pad: 1
181     kernel_size: 3
182   }
183 }
184 layer {
185   bottom: "conv4_1"
186   top: "conv4_1"
187   name: "relu4_1"
188   type: "ReLU"
189 }
190 layer {
191   bottom: "conv4_1"
192   top: "conv4_2"
193   name: "conv4_2"
194   type: "Convolution"
195   convolution_param {
196     num_output: 512
197     pad: 1
198     kernel_size: 3
199   }
200 }
201 layer {
202   bottom: "conv4_2"
203   top: "conv4_2"
204   name: "relu4_2"
205   type: "ReLU"
206 }
207 layer {
208   bottom: "conv4_2"
209   top: "conv4_3"
210   name: "conv4_3"
211   type: "Convolution"
212   convolution_param {
213     num_output: 512
214     pad: 1
215     kernel_size: 3
216   }
217 }
218 layer {
219   bottom: "conv4_3"
```

  
7







  
内容举报

  
返回顶部





7



```
220 top: "conv4_3"
221 name: "relu4_3"
222 type: "ReLU"
223 }
224 layer {
225 bottom: "conv4_3"
226 top: "pool4"
227 name: "pool4"
228 type: "Pooling"
229 pooling_param {
230 pool: MAX
231 kernel_size: 2
232 stride: 2
233 }
234 }
235 layer {
236 bottom: "pool4"
237 top: "conv5_1"
238 name: "conv5_1"
239 type: "Convolution"
240 convolution_param {
241 num_output: 512
242 pad: 1
243 kernel_size: 3
244 }
245 }
246 layer {
247 bottom: "conv5_1"
248 top: "conv5_1"
249 name: "relu5_1"
250 type: "ReLU"
251 }
252 layer {
253 bottom: "conv5_1"
254 top: "conv5_2"
255 name: "conv5_2"
256 type: "Convolution"
257 convolution_param {
258 num_output: 512
259 pad: 1
260 kernel_size: 3
261 }
262 }
263 layer {
264 bottom: "conv5_2"
265 top: "conv5_2"
266 name: "relu5_2"
267 type: "ReLU"
268 }
269 layer {
270 bottom: "conv5_2"
271 top: "conv5_3"
272 name: "conv5_3"
```

内容举报

返回顶部



```
272 name: "conv5_3"
273 type: "Convolution"
274 convolution_param {
275   num_output: 512
276   pad: 1
277   kernel_size: 3
278 }
279 }
280 layer {
281   bottom: "conv5_3"
282   top: "conv5_3"
283   name: "relu5_3"
284   type: "ReLU"
285 }
286 layer {
287   bottom: "conv5_3"
288   top: "pool5"
289   name: "pool5"
290   type: "Pooling"
291   pooling_param {
292     pool: MAX
293     kernel_size: 2
294     stride: 2
295   }
296 }
297 layer {
298   bottom: "pool5"
299   top: "fc6"
300   name: "fc6"
301   type: "InnerProduct"
302   inner_product_param {
303     num_output: 4096
304   }
305 }
306 layer {
307   bottom: "fc6"
308   top: "fc6"
309   name: "relu6"
310   type: "ReLU"
311 }
312 layer {
313   bottom: "fc6"
314   top: "fc6"
315   name: "drop6"
316   type: "Dropout"
317   dropout_param {
318     dropout_ratio: 0.5
319   }
320 }
321 layer {
322   bottom: "fc6"
323   top: "fc7"
324   name: "fc7"
```



7



内容举报



返回顶部



内容举报



返回顶部



```
325 type: "InnerProduct"
326 inner_product_param {
327   num_output: 4096
328 }
329 }
330 layer {
331   bottom: "fc7"
332   top: "fc7"
333   name: "relu7"
334   type: "ReLU"
335 }
336 layer {
337   bottom: "fc7"
338   top: "fc7"
339   name: "drop7"
340   type: "Dropout"
341   dropout_param {
342     dropout_ratio: 0.5
343   }
344 }
345 layer {
346   bottom: "fc7"
347   top: "fc8"
348   name: "fc8"
349   type: "InnerProduct"
350   inner_product_param {
351     num_output: 2622
352   }
353 }
354 layer {
355   bottom: "fc8"
356   top: "prob"
357   name: "prob"
358   type: "Softmax"
359 }
```

基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）  
——【三】使用Caffe的MemoryData层与VGG网络模型提取Mat的特征 完结，如果在代码过程中出现了任何问题，直接在博客下留言即可，共同交流学习。

版权声明：本文为博主原创文章，未经博主允许不得转载。

 发表你的评论

(http://my.csdn.net/waivin\_35068028)



退出打印

 内容举报

 返回顶部





<http://my.csdn.net/aa6121223>

aa6121223 (/aa6121223)

前天 23:01

31楼

(/aa6121223)

您好，请问出现\*\*\*check failure stack trace\*\*\*是怎么回事呢？在网上看到是路径设置问题，但是感觉工程里好像并没有涉及到啊？

回复

Morey\_Liu (/Morey\_Liu)

2017-09-11 11:36

30楼

(/Morey\_Liu)

楼主好！感谢提供这么详细的解决方案。我想咨询下：我用caffe提取一张图片的特征，在仅CPU模式下，平均耗时1.2s，和你的相差太大，虽然你有GPU加速，但总感觉我的时间不太正常，我试了下主要是net->Forward(input\_vec);耗时（基本都是1s），不知道问题出在哪，还望各位提供建议O(n\_n)O谢谢

回复

XJRemotion (/XJRemotion)

2017-08-19 23:50

29楼

(/XJRemotion)

看了楼主的代码，也参考了 caffe 源文件的代码，谢谢博主的博客指导，已完成2 inputs的任务，就是还有个问题：  
在出现了“MemoryData does not transform array data on Reset()”，我知道这个只是一个 Warning，但是接下来并不能显示 prob，只有再加上main.cpp上加上  
int i = 0;  
vector<float> print=ExtractFeature(lena);  
while (i<print.size())  
{  
cout << print[i++] << endl;  
}  
这样才能显示 prob，这是为啥？请问有遇到这个情况吗？

回复

查看 55 条热评

相关文章推荐

OpenCV+深度学习预训练模型，简单搞定图像识别 | 教程 ([http://blog.csdn.net/tansuo17/arti...](http://blog.csdn.net/tansuo17/article/details/51700418))

转载： <https://mp.weixin.qq.com/s/J6eo4MRQY7JLo7P-b3nvJg> 李林 编译自 pyimagesearch 作者 Adrian Rosebrock ...

tansuo17 (<http://blog.csdn.net/tansuo17>) 2017年09月04日 10:17 833

Deep Learning（深度学习）之（一）特征以及训练方法 ([http://blog.csdn.net/lwq1026/articl...](http://blog.csdn.net/lwq1026/article/details/51700418))

转自： [http://blog.csdn.net/boon\\_228/article/details/51700418](http://blog.csdn.net/boon_228/article/details/51700418) Deep Learning（深度学习）之（一）特征以及训练方法 ...

lwq1026 (<http://blog.csdn.net/lwq1026>) 2017年04月17日 10:55 667



一个普通程序员的内心独白....躺枪！躺枪！

我，一个普普通通程序员，没有过人的天赋，没有超乎寻常的好运，该如何逆袭走上人生巅峰？

广告

([http://www.baidu.com/cb.php?c=igF\\_pyfqnHmknjDLnjT0iZ0qnfK9ujYzP1nsrjD10Aw-5Hc3rHnYnHb0TAq15HfLPWRznpjBT1Y3rHfdnvnvPH0YPWF1mH9h0AwY5HDdnHc3rjbdnjf0lgF\\_5y9YIZ0IQzq-uZR8mLPbUB48ugfEXyN9T-KzUvdEIA-EUBqbugw9pysEn1qdlAdxTvqdThP-5yF\\_UvTkn0KzujYk0AFV5H00TZcqN0KdpYfqHRLPjnvnfKEpyfqHc4rj6kP0KWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6KWTnqnWbLnHb](http://www.baidu.com/cb.php?c=igF_pyfqnHmknjDLnjT0iZ0qnfK9ujYzP1nsrjD10Aw-5Hc3rHnYnHb0TAq15HfLPWRznpjBT1Y3rHfdnvnvPH0YPWF1mH9h0AwY5HDdnHc3rjbdnjf0lgF_5y9YIZ0IQzq-uZR8mLPbUB48ugfEXyN9T-KzUvdEIA-EUBqbugw9pysEn1qdlAdxTvqdThP-5yF_UvTkn0KzujYk0AFV5H00TZcqN0KdpYfqHRLPjnvnfKEpyfqHc4rj6kP0KWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6KWTnqnWbLnHb))




内容举报

返回顶部




一文看懂迁移学习：怎样用预训练模型搞定深度学习？(http://blog.csdn.net/hust\_hudi/articl...

跟传统的监督式机器学习算法相比，深度神经网络目前最大的劣势是什么？贵。尤其是当我们在尝试处理现实生活中诸如图像识别、声音辨识等实际问题的时候。一旦你的模型中包含一些隐藏层时，增添多一层...

 hust\_hudi (http://blog.csdn.net/hust\_hudi) 2017年07月08日 16:58 424


用Caffe提取深度特征 (http://blog.csdn.net/greenapple\_shan/article/details/50856499)

用Caffe提取深度特征 发表于 2015-05-28 | 1条评论 最近做对比实验，要比较非深度的方法加上deep feature之后的效果。于是就用Caffe提了一把特征，...

 greenapple\_shan (http://blog.csdn.net/greenapple\_shan) 2016年03月11日 15:37 8016

基于VGG-Face的人脸识别测试 (http://blog.csdn.net/u013078356/article/details/60955197)

VGG Face Descriptor 是牛津大学VGG小组的工作，现在已经开源训练好的网络结构和模型参数，本文将基于此模型在caffe上使用自己的人脸数据微调，并进行特征提取与精确度验证。数据传...

 u013078356 (http://blog.csdn.net/u013078356) 2017年03月09日 10:02 2044



程序员跨越式成长指南

完成第一次跨越，你会成为具有一技之长的开发者，月薪可能翻上几番；完成第二次跨越，你将成为拥有局部优势或行业优势的专业人士，获得个人内在价值的有效提升和外在收入的大幅跃迁.....

(http://www.baidu.com/cb.php?c=Igf\_pyfqnHmknjfrjD0lZ0qnfK9ujYzP1f4PjnY0Aw-5Hc4nj6vPjm0TAq15Hf4rjn1n1b0T1YzmmvLuARduH03ujKWmWT30AwY5HDdnHc3rjbdnjf0IgF\_5y9YlZ0lQzqMpgwBL5HDknWFBmhkEusKzujYk0AFV5H00TZcqn0KdpyfqnHRLPjnvnfKEpyfqnHnsnj0YnsKWpyfqP1civrHnz0AqLUWYs0ZK...


【Caffe实践】基于Caffe的人脸识别实现 (http://blog.csdn.net/chenriwei2/article/details/495...

导言深度学习深似海、尤其是在图像人脸识别领域，最近几年的顶会和顶刊常常会出现没有太多的理论创新的文章，但是效果摆在那边。DeepID是深度学习方法进行人脸识别中的一个简单，却高效的一个网络模型，其结构...

 chenriwei2 (http://blog.csdn.net/chenriwei2) 2015年11月01日 17:02 58603


基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）——【一】如何在Visual Studio中...

前言基于深度学习的人脸识别系统，一共用到了5个开源库：OpenCV（计算机视觉库）、Caffe（深度学习库）、Dlib（机器学习库）、libfacedetection（人脸检测库）、cudnn（gpu...

 Mr\_Curry (http://blog.csdn.net/Mr\_Curry) 2016年09月05日 20:35 12902

Dlib+OpenCV深度学习人脸识别 (http://blog.csdn.net/jcx0315/article/details/73449315)

Dlib+OpenCV深度学习人脸识别 前言 人脸识别在LWF(Labeled Faces in the Wild)数据集上人脸识别率现在已经99.7%以上，这个识别率确实非常高了，但是真实的环境...

 jcx0315 (http://blog.csdn.net/jcx0315) 2017年06月19日 01:10 3271

 内容举报

 返回顶部



...iWc4uHf3uAckPHRkPWN9PhcsmW9huWqdIAdxTvqdThP-




7



基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）——【二】人脸检测与预处理接口...

前言基于深度学习的人脸识别系统，一共用到了5个开源库：OpenCV（计算机视觉库）、Caffe（深度学习库）、Dlib（机器学习库）、libfacedetection（人脸检测库）、cudnn（gpu...

 Mr\_Curry (http://blog.csdn.net/Mr\_Curry) 2016年09月06日 20:53 13222



**Delphi7高级应用开发随书源码 (http://download.csdn.net/download/chenx...**


<http://download.csdn.net/download/chenx...>

2003年04月30日 00:00 676KB

下载


【深度学习】基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）(http://blog.csdn.n...

基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）——【二】人脸检测与预处理接口的设计 前言 基于深度学习的人脸识别系统，一共用到了5个开源库：OpenC...

 Taily\_Duan (http://blog.csdn.net/Taily\_Duan) 2016年12月28日 16:41 2365


基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）——【四】使用CUBLAS加速计算...

前言 本篇是该系列的第四篇博客，介绍如何使用CUBLAS加速进行两个向量间余弦距离的计算。##思路 我们先来温习一下两个向量之间余弦距离的数学公式，大家自己可以回忆一下：x,y均为同维度的向量...

 Mr\_Curry (http://blog.csdn.net/Mr\_Curry) 2016年09月09日 21:25 6097


VGG-Face：Deep Face Recognition 笔记 (http://blog.csdn.net/kunyXu/article/details/543...

vgg-face face-recognition

 kunyXu (http://blog.csdn.net/kunyXu) 2017年01月12日 12:39 2058

deep learning---利用caffe在vgg-face上finetuing自己的人脸数据 (http://blog.csdn.net/hlx37...

Abstract：本文将讲解如何利用自己的人脸数据在vgg-face上finetuing，主要包括数据的生成和文件的设置，以及最后的运行。1.代码和文件准备 代码caffe：http://caffe...

 hlx371240 (http://blog.csdn.net/hlx371240) 2016年05月12日 22:17 21981

深度学习与人脸识别系列（6）\_\_利用训练好的vgg模型进行人脸识别(利用摄像头) (http://blog....

作者：wjmishuai 出处：http://blog.csdn.net/wjmishuai/article/details/50854178 声明:版权所有,转载请注明出处 一:人脸识别系统...

 wjmishuai (http://blog.csdn.net/wjmishuai) 2016年03月11日 10:50 8697

Torch 7 利用已有VGG模型提取图片特征 (http://blog.csdn.net/DreamD1987/article/details/5...

在看torch的东西，感觉在深度学习的运用上，相对于Caffe来说更灵活，不过发现没有利用已有caffe模型提取图片特在的代码，在网上看见了一个，利用了GPU来处理批量图片的特征提取。写的比较规范，看...


 内容举报

 返回顶部





 内容举报

 返回顶部

 DreamD1987 (<http://blog.csdn.net/DreamD1987>) 2016年09月01日 11:43  3440



### 基于深度学习的人脸识别系统系列（Caffe+OpenCV+Dlib）——【六】设计人脸识别的识别类 ...

现在我们希望能够有一个识别的接口来实现输入一张图片，便可以分辨出他是哪个人。我们需要提前：1、定义一个人脸空间；2、将一些人脸的图片放到这个人脸空间中；3、将n个人脸图片提取特征为n个向量，并且...

 Mr\_Curry ([http://blog.csdn.net/Mr\\_Curry](http://blog.csdn.net/Mr_Curry)) 2016年10月01日 21:19  7265



### opencv+CNN实现人脸识别 (<http://blog.csdn.net/xs1997/article/details/68489868>)

在知乎上看到一个有趣的专栏，讲的是国外（日本？）一个牛人用OpenCV+CNN实现了一个人脸识别工具，觉得挺好玩的，所以fork下来自己也研究了一下，在这里做一个总结：项目描述 总的来说...

 xs1997 (<http://blog.csdn.net/xs1997>) 2017年03月30日 16:50  4220



### RCNN--对象检测的又一伟大跨越 (<http://blog.csdn.net/relar/article/details/52973889>)

最近在实验室和师兄师姐在做有关RCNN的研究，发现这里面坑很深呀，在网上找了一个大牛的博客，准备下来继续OPENC V同时，再来追一个RCNN的学习笔记的博文，博文地址如下：<http://blog.cs...>

 relar (<http://blog.csdn.net/relar>) 2016年10月30日 20:17  1133

### opencv 车道线检测（三） ([http://blog.csdn.net/Fate\\_fjh/article/details/61210633](http://blog.csdn.net/Fate_fjh/article/details/61210633))

车道线检测程序 特点：1.尽可能多的找出当前帧的车道线 2.车道线颜色不明显，破损，不清晰 3.车道切换 4.去除路面标志干扰 5.加入车辆检测增加鲁棒性测试视频：[这里写链接内容...](#)

 Fate\_fjh ([http://blog.csdn.net/Fate\\_fjh](http://blog.csdn.net/Fate_fjh)) 2017年03月10日 22:33  1279



 内容举报

 返回顶部

