

原

深度学习（十九）基于空间金字塔池化的卷积神经网络物体检测

2015年12月05日 17:40:41

hjimce

阅读数：16707

更多

版权声明：本文为博主原创文章，欢迎转载，转载请注明原文地址、作者信息。<https://blog.csdn.net/hjimce/article/details/50187655>

基于空间金字塔池化的卷积神经网络物体检测

原文地址：<http://blog.csdn.net/hjimce/article/details/50187655>

作者：hjimce

一、相关理论

本篇博文主要讲解大神何凯明2014年的paper：《Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition》paper主要的创新点在于提出了空间金字塔池化。paper主页：<http://research.microsoft.com/en-us/um/people/kahe/eccv14sppnet/index.html> 这个算法比R-CNN算法的速度快了n多倍。

我们知道在现有的CNN中，对于结构已经确定的网络，需要输入一张固定大小的图片，比如224\*224，32\*32,96\*96等。这样对于我各种大小的图片的时候，需要经过裁剪，或者缩放等一系列操作，这样往往会降低识别检测的精度，于是paper提出了“空间金字塔池化”这个算法的牛逼之处，在于使得我们构建的网络，可以输入任意大小的图片，不需要经过裁剪缩放等操作，只要你喜欢，任意大小的图片仅如此，这个算法用了以后，精度也会有所提高，总之一句话：牛逼哄哄。

空间金字塔池化，又称之为“SPP-Net”，记住这个名字，因为在以后的外文文献中，你会经常遇到，特别是物体检测方面的paper。比如：OverFeat、GoogleNet、R-CNN、AlexNet.....为了方便，学完这篇paper之后，你就需要记住SPP-Net是什么东西了。空间金字塔池化学习、特征表达的相关文献中，看到过几次这个算法。

既然之前的CNN要求输入固定大小的图片，那么我们首先需要知道为什么CNN需要输入固定大小的图片？CNN大体包含3部分，卷积连接。

首先是卷积，卷积操作对图片输入的大小会有要求吗？比如一个5\*5的卷积核，我输入的图片是30\*81的大小，可以得到(26,77)大小的图片。这会影响卷积操作。我输入600\*500，它还是照样可以进行卷积，也就是卷积对图片输入大小没有要求，只要你喜欢，任意大小的图片都可以进行卷积。

池化：池化对图片大小会有要求吗？比如我池化大小为（2，2）我输入一张30\*40的，那么经过池化后可以得到15\*20的图片。输入一张60\*60的图片，经过池化后，我可以得到30\*30大小的图片。因此池化这一步也没对图片大小有要求。只要你喜欢，输入任意大小的图片，都可以池化。

全连接层：既然池化和卷积都对输入图片大小没有要求，那么就只有全连接层对图片结果又要求了。因为全连接层我们的连接权值矩阵经过训练后，就是固定的大小了，比如我们从卷积到全连层，输入和输出的大小，分别是50、30个神经元，那么我们的权值矩阵（50\*30的矩阵了。因此空间金字塔池化，要解决的就是从卷积层到全连接层之间的一个过度。

也就是说在以后的文献中，一般空间金字塔池化层，都是放在卷积层到全连接层之间的一个网络层。

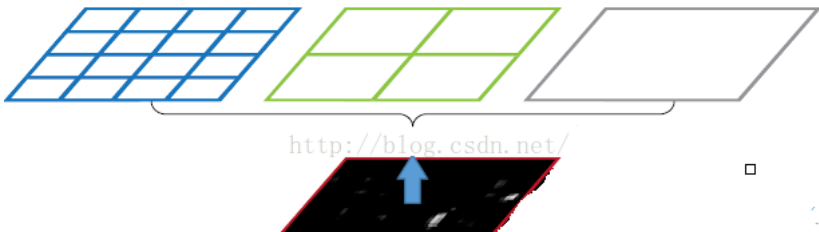
二、算法概述

OK,接着我们即将要讲解什么是空间金字塔池化。我们先从空间金字塔特征提取说起（这边先不考虑“池化”），空间金字塔是很久以前提出的特征提取方法，跟Sift、Hog等特征息息相关。为了简单起见，我们假设一个很简单两层网络：

输入层：一张任意大小的图片,假设其大小为(w,h)。

输出层：21个神经元。

也就是我们输入一张任意大小的特征图的时候，我们希望提取出21个特征。空间金字塔特征提取的过程如下：



## 图片尺度划分

如上图所示，当我们输入一张图片的时候，我们利用不同大小的刻度，对一张图片进行了划分。上面示意图利用了三张不同大小的图片进行了划分，最后总共可以得到 $16+4+1=21$ 个块，我们即将从这21个块中，每个块提取出一个15维特征向量，这样刚好就是我们需要的21维特征向量。

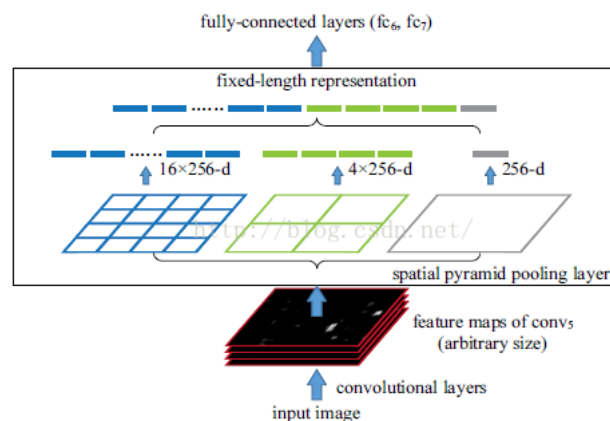
第一张图片，我们把一张完整的图片，分成了16个块，也就是每个块的大小就是 $(w/4, h/4)$ ;

第二张图片，划分了4个块，每个块的大小就是 $(w/2, h/2)$ ;

第三张图片，把一整张图片作为一个块，也就是块的大小为 $(w, h)$

空间金字塔最大池化的过程，其实就是从这21个图片块中，分别计算每个块的最大值，从而得到一个输出特征图。最后把一张任意大小的特征图，通过空间金字塔池化，转换成了一个固定大小的21维特征（当然你可以设计其它维数的输出，增加金字塔的层数，或者改变划分网格的大小）。上面的三种不同划分，每一种刻度我们称之为：金字塔的一层，每一个图片块大小我们称之为：windows size了。如果你希望金字塔的某一层输出 $n \times n$ ，那么你就用windows size大小为： $(w/n, h/n)$ 进行池化了。

当有很多层网络的时候，当网络输入的是一张任意大小的图片，这个时候我们可以一直进行卷积、池化，直到网络的倒数几层的时候，我们即将与全连接层连接的时候，就要使用金字塔池化，使得任意大小的特征图都能够转换成固定大小的特征向量，这就是空间金字塔池化（多尺度特征提取出固定大小的特征向量）。具体的流程图如下：



## 三、算法源码实现

理论学的再多，终究要实践，实践是检验理论的唯一标准，caffe中有关于空间金字塔池化的源码，我这边就直接把它贴出来，以供学友们参考。  
<https://github.com/BVLC/caffe> :

```

1 //1、输入参数pyramid_level: 表示金字塔的第几层。我们将对这一层，进行划分为 $2^n$ 个图片块。金字塔从第0层开始算起，0层就是一整张图片
2 // 第1层就是把图片划分为 $2 \times 2$ 个块，第2层把图片划分为 $4 \times 4$ 个块，以此类推.....，也就是说我们块的大小就是 $[w/(2^n), h/(2^n)]$ 
3 //2、参数bottom_w、bottom_h是我们输入这一层网络的特征图的大小
4 //3、参数spp_param是设置我们要进行池化的方法，比如最大池化、均值池化、概率池化.....
5 LayerParameter SPPLayer<Dtype>::GetPoolingParam(const int pyramid_level,
6           const int bottom_h, const int bottom_w, const SPPParameter spp_param)
7 {
8     LayerParameter pooling_param;
9     int num_bins = pow(2, pyramid_level); // 计算可以划分多少个刻度，最后我们图片块的个数就是num_bins*num_bins
10    // 计算垂直方向上可以划分多少个刻度，不足的用pad补齐。然后我们最后每个图片块的大小就是(kernel_w, kernel_h)
11    int kernel_h = ceil(bottom_h / static_cast<double>(num_bins)); // 向上取整。采用pad补齐，pad的像素都是0
12    int remainder_h = kernel_h * num_bins - bottom_h;
13    int pad_h = (remainder_h + 1) / 2; // 上下两边分摊pad
14    // 计算水平方向的刻度大小，不足的用pad补齐
15    int kernel_w = ceil(bottom_w / static_cast<double>(num_bins));
16    int remainder_w = kernel_w * num_bins - bottom_w;
17    int pad_w = (remainder_w + 1) / 2;
18
19
20    pooling_param.mutable_pooling_param()->set_pad_h(pad_h);
21    pooling_param.mutable_pooling_param()->set_pad_w(pad_w);
22    pooling_param.mutable_pooling_param()->set_kernel_h(kernel_h);
23    pooling_param.mutable_pooling_param()->set_kernel_w(kernel_w);
24    pooling_param.mutable_pooling_param()->set_stride_h(kernel_h);
25    pooling_param.mutable_pooling_param()->set_stride_w(kernel_w);
26
27    switch (spp_param.pool()) {

```

```

28 | case SPPParameter_PoolMethod_MAX:// 窗口最大池化 29 | pooling_param.mutable_pooling_param()->set_pool(
29 | PoolingParameter_PoolMethod_MAX);
30 | break;
31 | case SPPParameter_PoolMethod_AVE:// 平均池化 15
32 | pooling_param.mutable_pooling_param()->set_pool(
33 | PoolingParameter_PoolMethod_AVE);
34 | break; 10
35 | case SPPParameter_PoolMethod_STOCHASTIC:// 随机概率池化
36 | pooling_param.mutable_pooling_param()->set_pool(
37 | PoolingParameter_PoolMethod_STOCHASTIC);
38 | break;
39 | default:
40 | LOG(FATAL) << "Unknown pooling method.";
41 | }
42 |
43 |
44 | return pooling_param;
45 | }
46 |
47 | template <typename Dtype>
48 | // 这个函数是为了获取我们本层网络的输入特征图、输出相关参数，然后设置相关变量，比如输入特征图的图片的大小、个数
49 | void SPPLayer<Dtype>::LayerSetUp(const vector<Blob<Dtype>*>& bottom,
50 | const vector<Blob<Dtype>*>& top) {
51 | SPPParameter spp_param = this->layer_param_.spp_param();
52 |
53 | num_ = bottom[0]->num();//batch size 大小
54 | channels_ = bottom[0]->channels();//特征图个数
55 | bottom_h_ = bottom[0]->height();//特征图宽高
56 | bottom_w_ = bottom[0]->width();
57 | reshaped_first_time_ = false;
58 | CHECK_GT(bottom_h_, 0) << "Input dimensions cannot be zero.";
59 | CHECK_GT(bottom_w_, 0) << "Input dimensions cannot be zero.";
60 |
61 | pyramid_height_ = spp_param.pyramid_height();//金字塔有多少层
62 | split_top_vec_.clear();//清空相关数据
63 | pooling_bottom_vecs_.clear();
64 | pooling_layers_.clear();
65 | pooling_top_vecs_.clear();
66 | pooling_outputs_.clear();
67 | flatten_layers_.clear();
68 | flatten_top_vecs_.clear();
69 | flatten_outputs_.clear();
70 | concat_bottom_vec_.clear();
71 | // 如果金字塔只有一层，那么我们其实是对一整张图片进行pooling，也就是文献所提到的: global pooling
72 | if (pyramid_height_ == 1) {
73 | // pooling layer setup
74 | LayerParameter pooling_param = GetPoolingParam(0, bottom_h_, bottom_w_, spp_param);
75 | pooling_layers_.push_back(shared_ptr<PoolingLayer<Dtype> > (new PoolingLayer<Dtype>(pooling_param)));
76 | pooling_layers_[0]->SetUp(bottom, top);
77 | return;
78 | }
79 | // 这个将用于保存金字塔每一层
80 | for (int i = 0; i < pyramid_height_; i++) {
81 | split_top_vec_.push_back(new Blob<Dtype>());
82 | }
83 |
84 | // split layer setup
85 | LayerParameter split_param;
86 | split_layer_.reset(new SplitLayer<Dtype>(split_param));
87 | split_layer_->SetUp(bottom, split_top_vec_);
88 |
89 | for (int i = 0; i < pyramid_height_; i++) {
90 | // pooling layer input holders setup
91 | pooling_bottom_vecs_.push_back(new vector<Blob<Dtype>*>);
92 | pooling_bottom_vecs_[i]->push_back(split_top_vec_[i]);
93 |
94 |
95 | pooling_outputs_.push_back(new Blob<Dtype>());
96 | pooling_top_vecs_.push_back(new vector<Blob<Dtype>*>);
97 | pooling_top_vecs_[i]->push_back(pooling_outputs_[i]);
98 |

```

```

99 | // 获取金字塔每一层相关参数100|
    | LayerParameter pooling_param = GetPoolingParam(i, bottom_h_, bottom_w_, spp_param);101|
102 | pooling_layers_.push_back(shared_ptr<PoolingLayer<Dtype> > (new PoolingLayer<Dtype>(pooling_param)));
103 | pooling_layers_[i]->SetUp(*pooling_bottom_vecs_[i], *pooling_top_vecs_[i]);
    | 15
104 |
105 | // 每一层金字塔输出向量
106 | flatten_outputs_.push_back(new Blob<Dtype>());
    | 10
107 | flatten_top_vecs_.push_back(new vector<Blob<Dtype>*>);
108 | flatten_top_vecs_[i]->push_back(flatten_outputs_[i]);
109 |
110 | // flatten layer setup
111 | LayerParameter flatten_param;
112 | flatten_layers_.push_back(new FlattenLayer<Dtype>(flatten_param));
113 | flatten_layers_[i]->SetUp(*pooling_top_vecs_[i], *flatten_top_vecs_[i]);
114 |
115 | // concat layer input holders setup
116 | concat_bottom_vec_.push_back(flatten_outputs_[i]);
117 | }
118 |
119 | // 把所有金字塔层的输出，串联成一个特征向量
120 | LayerParameter concat_param;
121 | concat_layer_.reset(new ConcatLayer<Dtype>(concat_param));
122 | concat_layer_->SetUp(concat_bottom_vec_, top);
123 | }

```

函数GetPoolingParam是我们需要细读的函数，里面设置了金字塔每一层窗口大小的计算，其它的函数就不贴了，对caffe底层实现感兴趣可以自己慢慢细读。

#### 四、算法应用之物体检测

在SPP-Net还没出来之前，物体检测效果最牛逼的应该是RCNN算法了，下面跟大家简单讲一下R-CNN的总算法流程，简单回顾一下

- 1、首先通过选择性搜索，对待检测的图片进行搜索出2000个候选窗口。
- 2、把这2k个候选窗口的图片都缩放到227\*227，然后分别输入CNN中，每个候选窗台提取出一个特征向量，也就是说利用CNN进行提量。
- 3、把上面每个候选窗口的对应特征向量，利用SVM算法进行分类识别。

可以看到R-CNN计算量肯定很大，因为2k个候选窗口都要输入到CNN中，分别进行特征提取，计算量肯定不是一般的大。

OK，接着回归正题，如何利用SPP-Net进行物体检测识别？具体算法的大体流程如下：

- 1、首先通过选择性搜索，对待检测的图片进行搜索出2000个候选窗口。这一步和R-CNN一样。
- 2、特征提取阶段。这一步就是和R-CNN最大的区别了，同样是用卷积神经网络进行特征提取，但是SPP-Net用的是金字塔池化。这一操作如下：把整张待检测的图片，输入CNN中，进行一次性特征提取，得到feature maps，然后在feature maps中找到各个候选框的区个候选框采用金字塔空间池化，提取出固定长度的特征向量。而R-CNN输入的是每个候选框，然后在进入CNN，因为SPP-Net只需要图片进行特征提取，速度是大大地快啊。江湖传说可一个提高100倍的速度，因为R-CNN就相当于遍历一个CNN两千次，而SPP-Net5次。
- 3、最后一步也是和R-CNN一样，采用SVM算法进行特征向量分类识别。

算法细节说明：看完上面的步骤二，我们会有一个疑问，那就是如何在feature maps中找到原始图片中候选框的对应区域？因为候选框张原图片进行检测得到的，而feature maps的大小和原始图片的大小是不同的，feature maps是经过原始图片卷积、下采样等一系列操作的。那么我们要如何在feature maps中找到对应的区域呢？这个答案可以在文献中的最后面附录中找到答案：APPENDIX A：Mapping a Window to Feature Maps。这个作者直接给出了一个很方便我们计算的公式：假设(x',y')表示特征图上的坐标点，坐标点(x,y)入图片上的点，那么它们之间有如下转换关系：

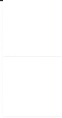
$$(x,y)=(S*x',S*y')$$

其中S的就是CNN中所有的strides的乘积。比如paper所用的ZF-5：

$$S=2*2*2*2=16$$

而对于Overfeat-5/7就是S=12，这个可以看一下下面的表格：

model	conv <sub>1</sub>	conv <sub>2</sub>	conv <sub>3</sub>	conv <sub>4</sub>	conv <sub>5</sub>	conv <sub>6</sub>	conv <sub>7</sub>
ZF-5	96 × 7 <sup>2</sup> , str 2 LRN, pool 3 <sup>2</sup> , str 2 map size 55 × 55	256 × 5 <sup>2</sup> , str 2 LRN, pool 3 <sup>2</sup> , str 2 27 × 27	384 × 3 <sup>2</sup> 13 × 13	384 × 3 <sup>2</sup> 13 × 13	256 × 3 <sup>2</sup> 13 × 15	-	-
Convnet*-5	96 × 11 <sup>2</sup> , str 4 LRN, map size 55 × 55	256 × 5 <sup>2</sup> LRN, pool 3 <sup>2</sup> , str 2 27 × 27	384 × 3 <sup>2</sup> pool 3 <sup>2</sup> , 2 13 × 13	384 × 3 <sup>2</sup> 13 × 13	256 13 × 10	-	-
Overfeat-5/7	96 × 7 <sup>2</sup> , str 2 pool 3 <sup>2</sup> , str 3, LRN map size 36 × 36	256 × 5 <sup>2</sup> pool 2 <sup>2</sup> , str 2 18 × 18	512 × 3 <sup>2</sup> 18 × 18	512 × 3 <sup>2</sup> 18 × 18	512 18 × 15	512 × 3 <sup>2</sup> 18 × 18	512 × 3 <sup>2</sup> 18 × 18



需要注意的是Strides包含了池化、卷积的stride。自己计算一下Overfeat-5/7(前5层)是不是等于12。

反过来，我们希望通过(x,y)坐标求解(x',y')，那么计算公式如下：

$$x' = \lfloor x/S \rfloor + 1$$

因此我们输入原图片检测到的windows，可以得到每个矩形候选框的四个角点，然后我们再根据公式：

Left、Top:

$$x' = \lfloor x/S \rfloor + 1$$

Right、Bottom：

$$x' = \lceil x/S \rceil - 1.$$

参考文献：

- 1、<https://github.com/BVLC/caffe>
- 2、《Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition》
- 3、<http://research.microsoft.com/en-us/um/people/kahe/eccv14sppnet/index.html>
- 4、<http://caffe.berkeleyvision.org/>

\*\*\*\*\*作者：hjimce 时间：2015.12.5 联系QQ：1393852684 地址：<http://blog.csdn.net/hjimce> 原创文章，转载请保  
址、作者等信息\*\*\*\*\*

想对作者说点什么？

我来说一句

- Li\_haiyu**：你好,博主,请教一个问题,文章中图二是怎么得到的？（3个月前 #8楼）
- M\_jiao**：全连接层fc6层的参数怎么设置，原来为4096，现在应该改为？？？（4个月前 #7楼）
- haixwang**：博主，我的理解是：全连接层的输入大小，并不是神经元个数，因为神经元个数是人为设定的。输入的应该是经过卷积层以及池化层处理之后的图像的尺  
维向量）。然后全连接层参数总数为尺寸\*feature map数\*分类数。所以文中"输入和输出的大小，分别是50、30个神经元"可能不妥，不知我说的对不（10个月前 #6楼  
[查看回复\(1\)](#)

查看 10 条热评

- 深度学习笔记（一）空间金字塔池化阅读笔记Spatial Pyramid Pooling in Deep Convolutional Network...

空间金字塔池化 空间金字塔池化层简介： 在对图片进行卷积操作的时候，卷积核的大小是不会发生... 来自：[HAHA的专栏](#)
- 【神经网络与深度学习】【计算机视觉】SPPNet-引入空间金字塔池化改进RCNN

转自：<https://zhuanlan.zhihu.com/p/24774302?refer=xiaoleimlnote> 继续总结一下RCNN系列。上篇RCNN- 将CNN... 来自：[ZhangPY的专栏](#)
- SPPnet——空间金字塔池化

转载：<http://blog.csdn.net/xjz18298268521/article/details/52681966> 论文： 《Spatial Pyramid Pooling in D... 来自：[gn102038的博客](#)
- 空间金字塔池化

本文为转载，原文网址：<http://blog.csdn.net/xzzppp/article/details/51377731> 1、简介 空间金字塔池化，使得任意... 来自：[dfsrewwg的博客](#)



ssp(空间金字塔池化)

前言： 接着上一篇文章提到的RCNN网络物体检测，这个网络成功的引入了CNN卷积网络来进行特征提取，...

来自： [Wl](#) [i6510的博客](#)

15

2557

金字塔池化过程及其优势

金字塔池化过程及其优势 第一次完全自己动手写博文，起初有点不知所措，后来是有种深深的责任感，经过查...

来自： [可为而为之](#)

10

6737

Spatial pyramid pooling (SPP)-net （空间金字塔池化）笔记

1、简介 空间金字塔池化，使得任意大小的特征图都能够转换成固定大小的特征向量，这就是空间金字塔池化的意...

来自： [ZPPP的博客](#)

3272

池化总结（Overlapping Pooling、一般池化、Spatial Pyramid Pooling）

池化方法总结（Pooling）在卷积神经网络中，我们经常会碰到池化操作，而池化层往往在卷积层后面，通过池化来...

来自： [可为而为之](#)

2.5万

深度学习的局部响应归一化LRN(Local Response Normalization)理解

1、其中LRN就是局部响应归一化： 这个技术主要是深度学习训练时的一种提高准确度的技术方法。其中caffe、ten...

来自： [yangdashi888的博客](#)

948

深度学习笔记空间金字塔池化阅读笔记Spatial Pyramid Pooling in Deep Convolutional Networks for Vi...


空间金字塔池化 空间金字塔池化层简介： 在对图片进行卷积操作的时候，卷积核的大小是不会发生...

来自： [oppo62258801的博客](#)

相关热词

深度学习深度学习 深度学习和 深度学习will 深度学习（ in深度学习


博主推荐



大饼博士X

关注


93篇文章



hjimce

关注

197篇文章



卜居

关注

77篇文章

...——OpenCV图像金字塔 - wsp\_1138886114的博客 - CSDN博客...

一、上采样理论 1.1 bilinear 1.2 Deconvolution(反卷积) 1.3 unpooling 二、OpenCV金字塔:高斯金字塔、拉普拉斯金字塔与图片缩放 一、上采样理论 FCN全卷积网络:...

net\_surgery中如何将全连接层转换成卷积层 - zy3381的专栏 - CSDN...

(shape)为1x256x13x13,然后将下一层的全连接层fc6修改为卷积层fc6-conv,...基于空间金字塔池化的卷积神经网络物体检测 Batch Normalization 学习笔记 深度学习...

学习金字塔理论 - CSDN博客

"学习金字塔"理论 美国学者埃德加·戴尔(Edgar Dale)1946年提出了"学习金字塔"(Cone of Learning)的理论。以语言学习为例,在初次学习两个星期后:学习金字塔理论...

卷积神经网络Lenet-5详解 - d5224的专栏 - CSDN博客

卷积神经网络Lenet-5实现 原文地址:http://blog.csdn.net/hjimce/article/details/47323463 作者:hjimce 卷积神经网络算法是n年前就有的算法,只是近年来因为深度...

金字塔实现 - 沐之博 - CSDN博客

金字塔算法一直是面试常见的基础题目,如何实现呢? //半边金塔 for (int i = 1; i <= rows;i++) { for (int j = 1; j<=i;j++) { printf("\*\* ...

matlab中卷积运算conv2的三种形式 - zy3381的专栏 - CSDN博客


matlab中的conv2是用于对二维数据进行卷积运算,有三个参数可供选择,下面是help content of conv2 conv2 Two dimensional convolution. C = conv2(A,...

图像处理之理解卷积 - CSDN博客

图像处理之理解卷积 一:什么是卷积离散卷积的数学公式可以表示为如下形式: f(x) = - 其中C(k)...

SPP空间金字塔池化(Spatial Pyramid Pooling) - jurong...\_CSDN博客

空间金字塔池化 空间金字塔池化层简介： 在对图片进行卷积操作的时候,卷积核的大小是不会发生变化...



hjimce

关注

博客专家

原创

196

粉丝

4790

喜欢

881

评论

660

等级： [博客 7](#) 访问： 169万+

https://blog.csdn.net/hjimce/article/details/50187655

6/8

积分：1万+

排名：1618

勋章：



个人简介

声明:博文的编写，主要参考网上资料，并结合个人见解，仅供学习、交流使用，如有侵权，请联系博主删除，原创文章转载请注明出处。博主qq：1393852684。内推阿里，欢迎大牛砸简历.....

最新文章

caffe to pytorch

screen 使用

tmep

深度学习（七十四）半监督Mean teachers

深度学习(七十三)pytorch学习笔记

博主专栏



深度学习

阅读量：120730468 篇

个人分类

图像处理

19篇

机器学习

20篇

深度学习

89篇

数据挖掘

4篇

基础知识

31篇

展开

归档

2018年10月

1篇

2018年9月

1篇

2018年7月

1篇

2018年6月

1篇

2018年3月

6篇

展开

热门文章

深度学习（二十九）Batch Normalization 学习笔记

阅读量：134708

深度学习（十八）基于R-CNN的物体检测

阅读量：71509

深度学习（四）卷积神经网络入门学习(1)

阅读量：52549

深度学习（十）keras学习笔记

阅读量：47684

深度学习（六）caffe入门学习

阅读量：47671

最新评论

深度学习（四）卷积神经网络入门学习...

15

10

lesonly：请问楼主，c3层特征组合时，仅仅是相加吗？不是相加后再求均值吗？？？

图像处理（十三）保刚性图像变形算法...  
u010760034：博主你好，我最近也在做这个方向，然后在编译这个代码的时候修改了一些错误但还是不能运行，右键选择一个...

深度学习（四）卷积神经网络入门学习...  
wwy\_\_：学习了

机器学习（四）高斯混合模型  
jicong44：写的很好，给楼主点赞。请问WeightGSM函数中第18行，是不是应该是：sigma=1/sum...

机器学习（十四）Libsvm学习笔记  
SJH\_CSDN：您好，可以发一份完整的代码吗？上面缺少#include "Vec.h"文件，邮...

15

10

联系我们



扫码联系客服



下载CSDN APP

 kefu@csdn.net

 400-660-0108

 QQ客服

 客服论坛

关于我们 招聘 广告服务 网站地图

 百度提供站内搜索 京ICP证09002463号

©2018 CSDN版权所有

经营性网站备案信息 网络110报警服务

北京互联网违法和不良信息举报中心

中国互联网举报中心