

**实验报告**

****

**作业题目： 计算机视觉**

**院 系： 计算机学院**

**专 业： 计算机科学与技术**

**姓 名：**

**学　　号：**

**指导教师：**

**2 0 1 8 年 0 1月**

**计算机视觉实验报告**

**项目主题**

自然场景下的文本识别

**课题介绍**

期中实现的文本识别课题，对环境条件限制较为严格，比如图片背景必须很纯净、字体必须为标准字体、字符之间不能相连、文本不能有倾斜等等 …… 于是想在期末项目中克服之前的问题，实现适用场景更普遍的自然场景下的文本识别

本次项目分成了两个板块：文本检测和字符识别

**文本检测**：基于SegLink网络对自然场景图片中的所有文本区域进行检测

**文本识别**：基于LSTM + CTC模型对检测的文本区域进行字符识别

报告中我会先介绍上述两个阶段的具体实现过程，同时对每个阶段每一步的实验结果进行展示，然后阐述为什么选择这两种方法，并将期中与期末的实验进行对比

**实验过程**

**注：本部分基于自己对该方法和代码的理解，绝非堆砌概念与过程。每一个步骤自己都查阅了相关论文与博客，并根据自己的思路进行整理，然后分成了几个模块。如果内容有错误，也希望陈老师和助教能够通过邮件进行指正！！！（做了好久TnT）**

**文本检测**

注：文本检测用到的SegLink方法出自于CVPR2017的一篇论文，看完论文后我从github上下载了该

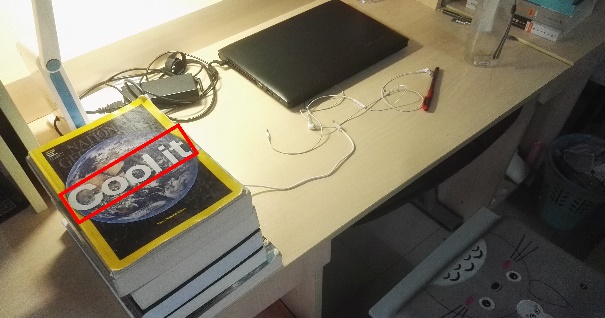
模型的代码，阅读并理解所有代码后，在自己的服务器上将网络跑了一遍，然后将源码稍作修改，

输出了每个小阶段的实验结果，等下会逐一进行展示

**检测过程:** a . 将图片输入SegLink网络，学习后得到seg和link信息：seg框出某个文本区域中的一个或多个字符，link连接相邻的seg

b . 通过link将图中seg进行连接形成最终的文本区域

**结果展示：**

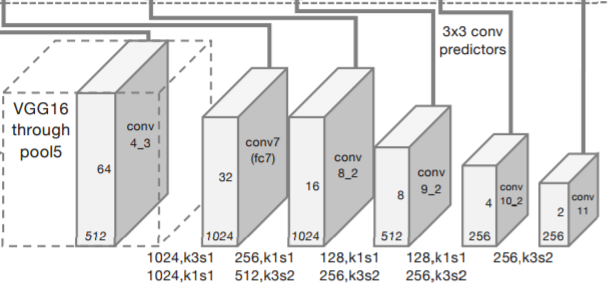


a1. 原图 a2. 识别结果

注：图a1是我寝室拍的自己的书桌，后续阶段我将使用图a1（原图），来演示各个layer的实验结果

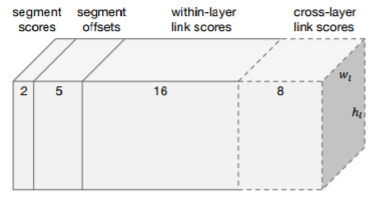
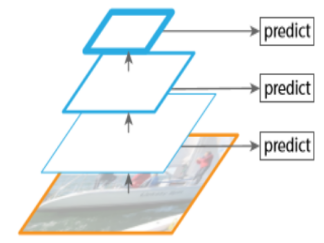
**模型介绍：**

SegLink网络结构如图b1所示，其basenet是一个VGG16网络，随后在其基础上添加了4个卷积层（conv8\_2 – conv11），各个feature map和kernel的参数如图所示



b1.SegLink网络结构

SegLink的特别之处在于：它在conv4\_3，conv7，conv8\_2，conv9\_2，conv10\_2，conv11这6个卷积层中（**layer1-6**），每一层都添加了对各seg和link的预测与评估，用于学习，而非只在最终结果中进行预测与评估

b2.layer1-6中进行预测和评估的输出层

C:\Users\赵世豪\Desktop\QQ截图20180105234510.png C:\Users\赵世豪\Desktop\QQ截图20180105234145.pngC:\Users\赵世豪\Desktop\QQ截图20180105234203.pngC:\Users\赵世豪\Desktop\QQ截图20180105234213.png

b3.实现代码中的layer1-6的预测与评估模块

**Seg模块：**

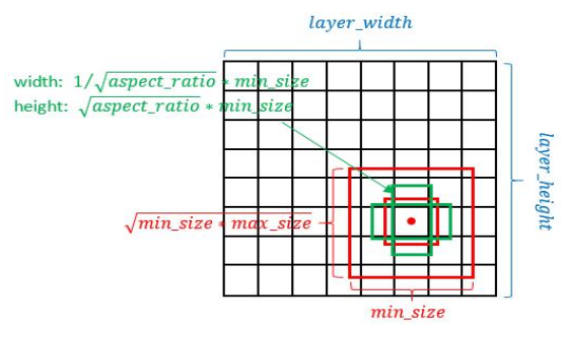
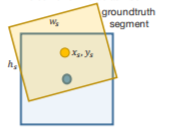
Seg是一个bounding-box类，由C:\Users\赵世豪\Desktop\QQ截图20180105235204.png 唯一确定，各个分别表示box中心坐标（cx, cy），box宽高（w, h），box的旋转角度theta

预测过程分为以下几步：

1. 在layer1-6的每个特征图( feature map )上，以每个点为中心，生成一系列同心的**default-box**（bounding-box类），如图c2所示
2. 对每个default-box，用图b2中的卷积层进行判断,得到每个box在“文本label”上的得分segment score，最高的即为正确的框**right-box**
3. 通过bounding box regression对right-box进行微调，得到最终的预测位置**seg**

C:\Users\赵世豪\Desktop\QQ截图20180106005142.png

c1.预测过程

c2.default-box图解 c3.default-box与text-area

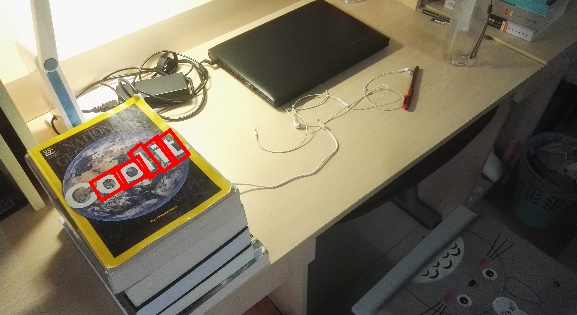
由于特征图feature map随着池化操作pooling大小逐层减小（我取的池化核大小为2 \* 2，因而大约每过一层长宽比例各减少1/2），因而每层feature map中的default-box映射到原图中大小会逐层增大（长宽比例各增大2倍），所以每一层layer将会对不同大小的文本进行预测

训练集中groundTruth的确定：由于所用到的数据集，只在原图中确定了text-area作为groundTruth用于计算loss函数，而本方法在每一层都进行了学习，所以要在原数据集的基础上产生各个预测层的特征图上的text-area

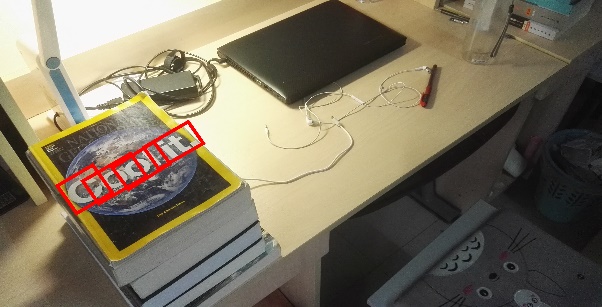
若layer1-6中的default-box满足以下两个条件，则标记该box为真，否则为假：

1. Box坐标中心（x, y）在原图的映射点c，包含在原图的text-area中
2. 对于Box的各边a，text-area各边b，满足max { a/b, b/a } <= 1.5（如图c3）

**每一层的seg预测结果展示：**



result1. layer1-conv4\_3 result2. Layer2-conv7



result3. layer3-conv8\_2 result2. layer4-conv9\_2



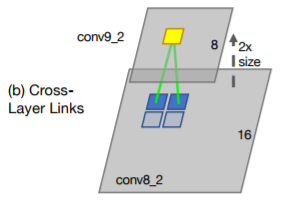
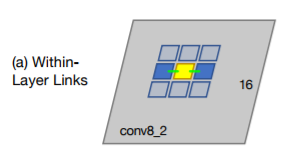
result3. layer3-conv8\_2 result2. layer4-conv9\_2

注：至于各层为什么会出现这个结果，详见上述seg模块

**Link模块：**

Link连接了各相邻的seg，共有两种类型：within-layer-link和cross-layer-link

Within-layer-link：连接同一层layer内相邻的seg，两个被该link连接的seg即为同一文本区域的字符。对于中心点为（x，y）的seg，则任何满足下列式子的seg\_（x\_, y\_）将通过link与seg连接C:\Users\赵世豪\Desktop\QQ截图20180106121443.png



d1.within-layer links d2.cross-layer links

Cross-layer-link：连接相邻层（比如conv7和conv8\_2）之间相邻的seg。两个被该link连接的seg即为同一文本区域的字符。对于在layer\_i中以（x, y）为中心点的seg，则在layer\_i+1中任何满足如下式子的seg\_（x\_, y\_）将通过link与seg连接

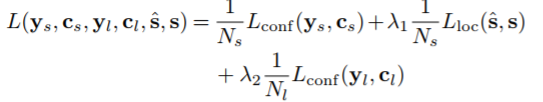
C:\Users\赵世豪\Desktop\QQ截图20180106122029.png

设定cross-layer-link的原因：由于不同layer的feature map大小不同，因而在不同的特征图中default-box的大小也不尽相同。所以，可能同一个文本区域在不同layer中进行了重复检测。为了减少重复检测产生的冗余，就设定了cross-layer-link

训练集中groundTruth的确定：由于原数据集中不存在link及其groundTruth，所以需要自己产生link的正确标记用于学习：对于某个link，若其连接的两个seg都被标记为真，则该link为真，否则为假

**损失函数：**

损失函数表达式如e1所示。函数分为两个板块：



e1.损失函数

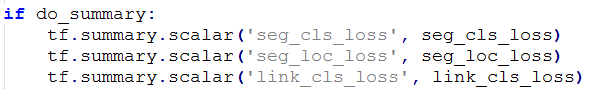
1. C:\Users\赵世豪\Desktop\QQ截图20180106124307.png和C:\Users\赵世豪\Desktop\QQ截图20180106124618.png为seg和link关于各类label得分score的softmax loss

C:\Users\赵世豪\Desktop\QQ截图20180106123857.png

C:\Users\赵世豪\Desktop\QQ截图20180106123948.png

1. C:\Users\赵世豪\Desktop\QQ截图20180106124506.png是预测seg位置的 smooth L1 regression loss

C:\Users\赵世豪\Desktop\QQ截图20180106125147.png



e2.损失函数代码实现部分

注：y表示各类label，c表示得分score，下标s表示seg，下标l表示link

**连接算法：**

经过以上步骤后得到原图中预测的各个seg和link。连接时可以将seg看做点，link看做边，构成一个图，一个连通图可以视为一个检测到的文本区域。所以将同一个连通图内的的各seg进行连接，得到最终结果。具体融合算法如下：

1. 求出连通图内所有seg(x, y, w, h, theta) 的theta平均值theta\_
2. 找到一条直线l : tan(theta\_) \* x + b，满足l到连通图内各seg距离和最短
3. 设直线l两端的seg分别为seg1（x1, y1, w1, h1, theta1），seg2（x2, y2, w2, h2, theta2）。建立一个新的seg（x, y, w, h, theta）作为连接结果，各参数满足：

C:\Users\赵世豪\Desktop\QQ截图20180106132230.png

f1.代码中的参数更新

**训练集&性能度量：**

原论文中用到的数据集为ICDAR2015和MSRA-TD500，github上下载的代码训练的数据集为ICDAR2013和ICDAR2015。我租了个服务器，只跑了ICDAR2015数据集(资金问题，按小时租的)。最好的情况下，精密度precision为72.9，召回率recall为76.8，F值为73.9（三个都赶不上原paper的结果TnT）

**其他结果示例：**



show-good. 最好的一次了(也是寝室拍的) show-bad. 这就不怎么好了(教室拍的)

**文本识别**

注：本部分参考了网上的一些博客，然后自己独立实现了代码

**方法简介：**

在原始图片中提取到文本区域后，接下来便是对文本区域进行文本识别的过程

由于自然场景下字符的字体、大小、字符间的间隔、字符串的长度等诸多因素都是不确定的，而传统方法普适性较低，所以在自然场景下处理具有一定的局限性。于是我觉得可以用神经网络来学习这些不确定的情况，并最终选择了LSTM网络和CTC模型

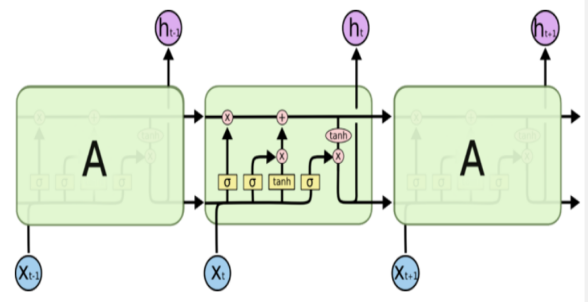
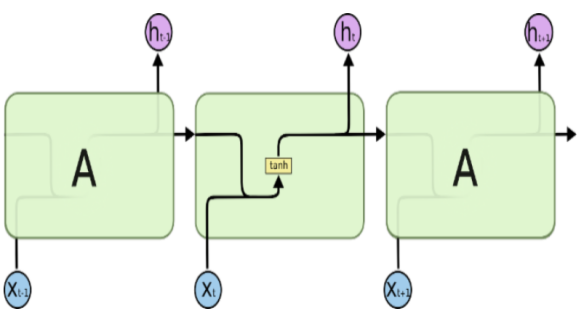
之前用到的方法将 ”文本识别” 分离成 ”文本切割” 和 ”字符识别” 两个阶段，而本次实验中的模型融合了并自动完成了这两个过程：

对于大小为width \* height的文本区域图片：

1. 将其平均切分为m = width列，每一列作为LSTM的输入，最终得到m个输出
2. 通过最小化CTC缺失函数，将m个输出映射为长度为n的最终结果（n即为字符串中字符的个数）

**LSTM网络：**

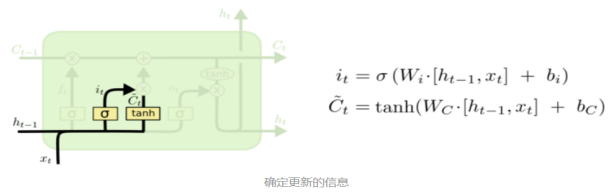
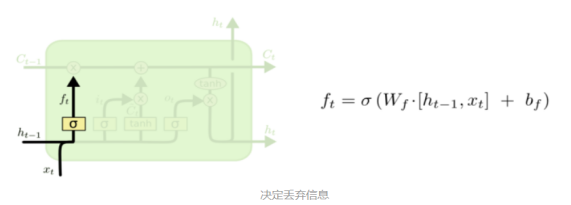
LSTM为一种特殊的RNN，结构如图g2。普通的RNN很难记住很久以前的信息，但LSTM通过一条贯穿整个网络的细胞状态水平线，能够联系较为久远的输入信息



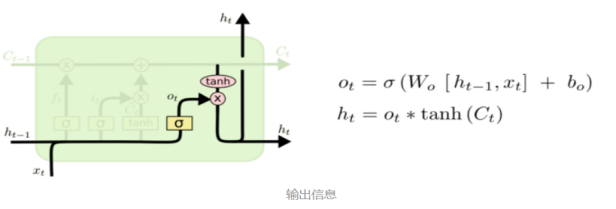
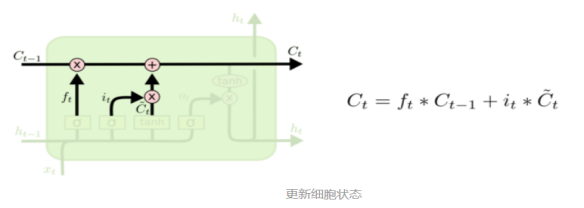
g1.普通RNN结构 g2.LSTM结构

**LSTM结构简介：**对于LSTM的具体原理我只是大致了解，没有去看相应的论文。

但是通过很多博客看了模型中各个模块的公式和意义（如图g3），所以还是懂其大概的



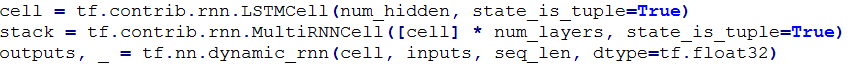
g3-1.确定要丢弃的信息 g3-2.确定要更新的信息



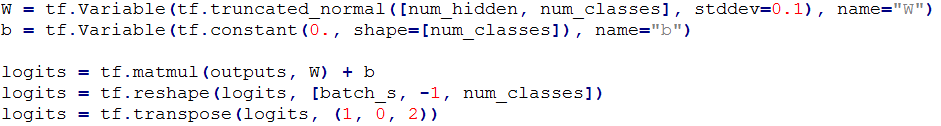
g3-3.更新细胞的状态 g3-4.输出与传递的信息

**本次实验为什么使用LSTM :**

由于CNN输入的是一张静态的图片，而本次实验中，不同文本区域中字符长度不尽相同，即字符长度是随着输入图片的不同而动态的变化，因而静态的CNN无法进行准确的识别。但对于LSTM，由于输入的长度m等于图片的宽度，即每一列像素作为一个输入，假设文本中的某个字符占据了k列，LSTM较于CNN的优势便是可以将这k列信息动态的联系起来，最终确定这k列所属的字符类别



g4-1.代码实现：网络搭建



g4-2.代码实现：归一化

**CTC模型：**

**问题的产生：**由于LSTM的输入为m个像素列，输出却为n个字符（m<<n）。

但LSTM的输出数量和输入一样多，所以只用LSTM会产生比如将字

符ab识别为aaabbbb等类似情况

**CTC解决方法：**CTC在标注符号集中加了一个空白符号 ”-“ 。在LSTM得到的输

出字符中，把空白符和预测出的重复字符消除。比如对于ab预测

出的aaa-bbbb则转换成ab，这样就得到了准确的结果

C:\Users\赵世豪\Desktop\QQ截图20180107130928.png

h1.代码实现：CTC缺失函数

**训练集&性能：**

在网上并没有找到适合的数据集（可能是我太不会找了），于是自己利用第一阶段产生的文本区域，手动标注了200个实例作为数据集（感觉好无助TnT），其中180个作为训练集，剩下的20个做了测试集。结果惨不忍睹，主要问题应该是训练集太少，但是字符之间的组合又太多，180个远远不够！

**OCR课题对比与总结**

**首先想谈一谈我对OCR这类问题的理解：**

OCR即光学字符识别，应用场合大致可以分为以下两种情况：

1. 一种是在某种特定场景下进行字符识别，比如PDF到word格式的转换、较为工整的纸质文字的扫描录入等。这类问题对待识别的图片要求较高，比如背景纯净、文本归正、字符工整等等，因而处理起来较为简便
2. 另一种则是自然场景下的字符识别，比如车牌检测，环境中的文本提取等。这类问题情况更为复杂，处理时需要考虑的因素较多
3. 自然场景下的图片背景可能会很杂乱，可能会影响到对文本区域的判断
4. 文本中字符串各行列的大小、间隔都不固定，会影响到单个字符的提取
5. 单个字符的字体、大小、颜色、字符间的间距也不尽相同

**针对这三个阶段，我谈谈期中期末两种方法的差异：**

期中我完成了情况1（某种特定场景下）的OCR，期末即本次实验我实现了情况2（自然场景下）的OCR。两次实验都是从三个方面着手的：文本提取、字符分割和字符识别。其实这三个阶段也恰恰对应了我上面提到的a. b. c三个方面的困难，每一步实现的时候都感受到了两种OCR之间显著的差异！！！

**文本提取：**1. 期中用了形态学操作——膨胀和腐蚀来提取文本。这种方法有优

点，比如算法简单易懂、计算量较小，很“轻便”；但其缺点也是不可掩盖的，那就是对环境要求高，适用场景少，在某些自然场景中可能会产生误判，比如图片中远处的天上有一群排列飞行的鸽子，那么在经过膨胀和腐蚀后，鸽群也有可能被检测成文本区域

2. 期末用了神经网络SegLink来检测文本区域。这种方法的优点即普

适性好，它是基于大量的数据集学习而得到，涉猎的场景非常广泛，因而在自然场景下的识别率相较于1更高。但神经网络一方面需要喂大量的数据去学习，另一方面计算量巨大、复杂度极高，这一短板也让其在特定场景下的OCR中显得笨重，大材小用了

**字符分割&字符识别：**

1. 期中将两个阶段分开来做：字符分割用到了投影法，字符识别用到了CNN。对于前者——投影法，其局限性是非常大的：由于该方法是通过字符与字符间的空白区域对字符串进行分割的，所以一旦字符间有连接、或者同一个字符里有中断，该算法就会将多个字符识别为一个、或者将一个字符识别为多个，进而失败。这些情况对于某些花体或手写体字符是即为常见的；对于后者CNN而言，由于当时图片中的字符都是等线、宋体等，所以识别算法用SVM就可以达到很好的效果了
2. 期末则将两个阶段进行了融合：采用LSTM+CTC的方法。前文提到过，一方面自然场景中很多字符都很“花哨”，另一方面文本区域中字符个数不是固定的，所以无法用投影法等一众传统方法进行有效的分割。而LSTM和CTC则不受上述限制，在数据集丰富时各种字体的字符识别能够达到高准确率，同时还能够很好的处理变长字符串

从上面的对比可以看出，每一种算法都有它自己的优缺点、并适用于各自特定的

场合。这点和我们的生活也很像，每一个物体都有它存在的道理，每一件事情都有它发生的缘由，每一个人也都有适合的岗位与最终的归宿！

**参考文献**

[1]Baoguang Shi, Xiang Bai, Serge Belongie. Detecting Oriented Text in Natural Images by Linking Segments. CVPR, 2017

[2] <https://www.cnblogs.com/52machinelearning/p/5821591.html>. #Deep Learning回顾#之LeNet、AlexNet、GoogLeNet、VGG、ResNet

[3] <http://blog.csdn.net/marsjhao/article/details/72955935>. 深度学习经典卷积神经网络之VGGNet

[4]W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. SSD: single shot multibox detector. ECCV, 2016

[5] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, and A.W.M. Smeulders. Selective Search for Object Recognition. IJCV, 2012

[6] <https://www.cnblogs.com/zhao441354231/p/5941190.html>. 目标检测--Selective Search for Object Recognition(IJCV, 2013)

[7] <https://www.jianshu.com/p/4fadf629895b>. 端到端的OCR：LSTM＋CTC的实现

[8] <http://blog.csdn.net/prom1201/article/details/52221822>. RNN介绍，较易懂

[9] <https://www.csdn.net/article/2015-06-05/2824880>. 深入浅出LSTM神经网络

[10] <http://blog.csdn.net/xmdxcsj/article/details/51763868>. CTC学习笔记（一） 简介