# 我们是CYY\_XAUT团队，我们的指导老师是朱虹老师。来自西安理工大学，队长账号13689246834。我们的方法是首先采用faster进行车牌检测，其次对检测到的车牌放入VGG16根据畸变程度进行粗分类，然后在利用crnn+attention进行车牌识别。老师我们昨天上交的程序有误，麻烦您重新接收一下我们的程序。

# 程序说明文档

本文的主要算法流程如下所示，首先对输入图像进行车牌检测，其次对检测到的车牌进行分类处理，最后进行车牌识别。

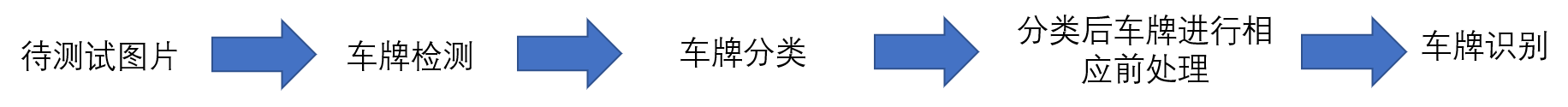


图1 本文算法流程图

## 检测过程：

1. **网络结构**

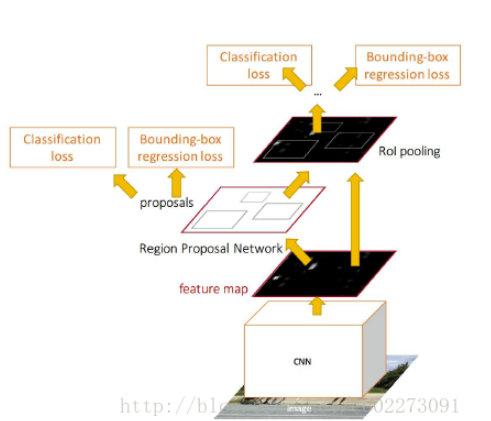


图2 faster网络结构图

本题目采用faster-rcnn网络进行车牌检测，faster-rcnn网络结构如图2所示。在本题目中前端卷积网络采用ResNet101网络进行特征提取，faster-rcnn网络使用RPN网络生成Proposals（建议窗口），这里每张图片生成300个Proposals，通过ROI pooling层使每个ROI生成固定尺寸的feature map，之后利用Softmax Loss(用于探测分类概率) 和Smooth L1 Loss(用于探测边框回归)对分类概率和边框回归联合训练。本题目在对车牌进行检测的同时进行车牌颜色分类。因此在打标数据集时会标注每张图像的车牌颜色类别和车牌位置，生成xml标签文件。

1. **训练过程**
2. 网络训练过程中，在Resnet101预训练模型的基础上进行车牌检测模型的训练，对已经训练好的模型用车牌数据进行微调；
3. 训练集包含3300张图像，学习率为0.001，batchsize为1，进行了70000次迭代训练；
4. 在训练和测试中，根据车牌目标的特点，采用了（8，16，32）三种尺度的anchor box，并将长宽比设为（2.0，2.8，3.0）以更好适应车牌目标。

## 识别过程：

1. **网络结构**

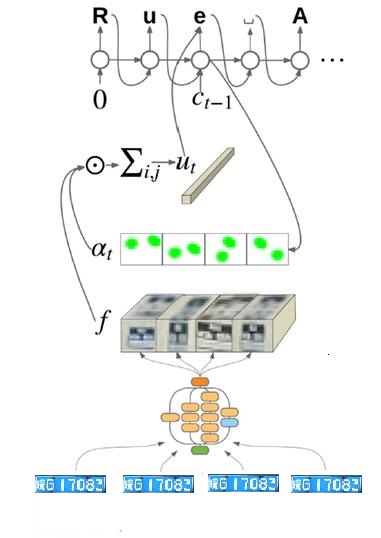


图3 识别网络结构

CNN: 7层卷积层，4层下采样，最后一层卷积层经过Relu激活函数输出；CNN的主要功能是为了完成输入图片特征的提取，即提起车牌的特征；

BiLSTM：拥有256维的隐层单元的双层LSTM,；BiLSTM主要用来对CNN提取到的车牌特征进行编码，即将特征映射到256维；

GRU：拥有256维的隐层单元的双层LSTM，对BiLSTM的输出进行解码；

Attention：在产生当前输出同时，产生一个“注意力范围”表示接下来输出的时候要重点关注输入序列中的哪些部分，然后根据关注的区域来产生下一个输出，如此往复。这样，解码不再依赖一个中间向量，而是由注意力模型对所有编码特征进行加权调整后得到的特征向量。

1. **训练过程**
2. 由于训练样本较少，我们采用一定方法，生成了一些车牌，生成车牌如下：

图4 生成的车牌图像

等等；

由上图可以看出，生成的样本真实车牌还是有一定差距的，因此我们只用生成的车牌对模型进行预训练，预训练时学习率0.001；训练100个epoch，每迭代5000个batch学习率减小一个数量级。

1. 模型训练

将所有真实的车牌放入网络继续训练，80%训练，20%验证；

预训练时学习率0.001，训练100个epoch，每迭代5000个batch学习率减小一个数量级；

Batchsize：300；

输入：120\*32；

输出：车牌上的文字，长度任意。

1. 样本的处理
2. 样本增强，随机旋转，加噪声、模糊、对比度增强等；
3. 对双层车牌，先对其进行裁剪，将其拼成单层，如

**3、测试过程**

读入待测试图片，先进入检测网络进行车牌检测，其次将检测车牌进入识别网络，识别网络输出车牌识别结果。

## **车牌分类**

**利用VGG16,对检测到的车牌进行简单粗分类，以便于下一步的精确识别。**

**对于发生旋转，模糊等比较严重情况下的车牌，如果直接进行识别，准确率很低，因此。本文训练了一个分类为10类的分类。**

**具体做法为：检测待测试图片中包含车牌图像的一个正方形区域，然后根据车牌在此区域内的特征，对图片进行简单分类。**

## 分类后车牌的处理

由于卷积神经网络的学习能力是有限的，而对性能检测库而言，包含各种各样类型的车牌，如果全部放入网络进行训练，那么网络就会很难收敛，甚至不收敛。因此除了和功能检测库一样，对其进行首先进行车牌检测，其次，在识别之前，对各个性能检测到的车牌进行了前处理，最后对处理后的车牌进行识别。我们对于性能库中样本的处理方法如下：

1. **竖直透视角**

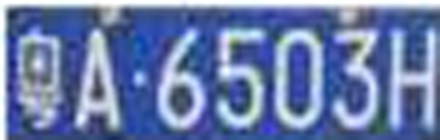
检测到的图像 检测图中的红色点 进行仿射变换后结果

1. **水平透视角**

检测到的图像 检测图中的红色点对应到右边的绿色点 进行仿射变换后结果

1. **低分辨率**



检测到的图像 三次插值后结果

1. **亮度不均匀、平均亮度**

对检测到的图片进行均值方差统一处理

检测到的图片 处理后图片 

检测到的图片 处理后图片

1. **散焦模糊、运动模糊**

检测到的图片 恢复的图片

**利用全卷积网络，对模糊进行恢复。**

**8、水平旋转**

 检测旋转角度，仿射变换恢复 

**9、竖直错切**



检测到的图像 检测图中的红色点 进行仿射变换后结果

## 程序主要文件简介

**lib文件夹**：用于存放一些python接口文件，datasets主要用于数据库的读取；layer\_utils为与anchor proposal相关的接口文件；model主要为网络的config配置文件；nets为基类网络的接口，如resnet、vgg等；nms为nms的c和cuda的相关加速代码；roi\_data\_layer为ROI层接口文件；utils为一些辅助工具接口文件，如计时、可视化等。

**output文件夹**：存放训练好的faster模型；

**img\_test**文件夹：存放测试图片；

**detect.py**:主要用来车牌检测

**VGG16.PY**;主要用于车牌分类

**config.py**:主要保存了识别模型的参数，比如模型的输入图像大小为120\*32；

**model.py**:主要存储了识别模型的结构；

**test.py**：恢复低分辨率，投视角等性能库；

**main.py:**调用检测分类、识别模型，完成整个车牌识别。

程序运行指南

依赖的库：tensorflow1.6.0；python3.5.4;opencv3.4.1;numpy1.13.1;xlsxwriter(用来写入xlsx文件)；pillow5.0.0用来图像增强；

运行步骤：将要测试的样本置于./img\_test/ 文件夹下；

运行main函数，最后会生成一个名字为test.xlsx的文件。即为最终车牌结果。

**最后，我们的测试程序是端到端的，只需将含有车牌的图像输入，即可经过检测，识别网络得到车牌识别结果。**