整个比赛做下来，经历了以下几个主要阶段：

1. **题目分析和模型选择**

不难分析题目是一个目标检测问题，当时较为主流的目标检测算法分为one-stage的yolo系列算法，ssd算法和two-stage的rcnn系列算法。其主要的性能差别是，one-stage系列的算法速度更快，但准确率相对偏低；而rcnn系列算法发展到faster-rcnn准确率相对较高，但速度相对较慢。具体到本次比赛的试题，不需要较高的帧率，而对准确率的要求更高，所以选取的faster-rcnn作为参赛模型。

1. **论文研读/源码分析**

论文见附件，源码见git地址，过程中遇到的问题见问题记录exl

论文研读和源码分析最大的收获是把作者的论文和代码实现对应起来，并且学习了解到caffe框架的模型是基于一份.proto的文件

1. **算法复现keras/tensorflow版本**

作者源码基于caffe框架，安装较为复杂，特别是在windows下，且我们打算使用流行度更广的tensorflow框架来进行比赛，所以复现论文没有考虑使用caffe

使用keras是因为它是一个高度封装的深度学习框架，后端基于tensorflow。吴恩达的课程也有接触过，keras的搭建模型非常的迅速，代码简单，依赖的库少，非常适合于理解模型和快速上手。从git上下载稍微修改本地环境就可以跑起来，效果也不错（但没有做量化）

使用tensorflow框架，git上down一个star数目较多的版本作为我们的base版本，这是因为faster-rcnn算是一个挺复杂的模型，重复造轮子在代码结构和细节地方会有较大风险，不适合时间较为紧张的竞赛。从base版本中主要学习到以类的形式组织起整个模型的训练和推理代码，层次较为清晰，也很适用于直接拿到别的项目作为base代码框架；学习到tensorflow模型/训练可视化 和 断点后接续训练 的方法

在该base版本上跑通voc\_pascal 2007，得到71%的mAp

1. **基于竞赛数据集的方案研讨**

当前数据集，是网络图片的文字检测。通过观察数据集，发现数据集的主要特点是：1）标注的框不是矩形，而是通过4个（x，y）标识的不规则的四边形，特别是存在一些斜的类平行四边形图案或非水平的类矩形图案；2）标注框的大小和宽高比例较多，原文使用的anchor size和ratio适用性不足；3）对于位置回归的偏移计算，因为变成了不规则四边形原公式不再适用。

当时讨论的方案主要有3种：

1. 把所有的不规则四边形处理为矩形，再根据人为观察数据集采样的方式，决定出几个size和ratio；并适用透射变换的公式，把4个位置回归的参数变成9个透射变换参数，再设计位置回归公式和loss。此方法的优点是对模型的改动最小，可以较快改出新模型，缺点也很明显，粗暴的把检测框处理成平行四边形会损失斜体字精度
2. Rotate rpn的思想。参考论文...为anchor加入旋转属性，即与x正方向的夹角来表示anchor。相应的在计算iou，nms和roi-pooling的时候都需要考虑角度信息。该方法引入了旋转信息很好的解决了非水平gtbox的问题，但是缺点是模型因为加入了角度信息变得非常复杂，iou和nms和roi-pooling都需要重写，因为不是基于矩形的操作很别扭，而且参数空间变得比较大，每个feature点上的anchor由原来的9个变成了54个（加入了6个角度信息）
3. 后期进一步研读该领域的论文，发现现在主流的文字检测方法，主流的做法是水平矩形的使用faster-rcnn可以较好解决，但是对于非水平的检测框，使用的更多的还是基于语义分割的办法，见论文...加入一个mask层要做到像素级分割。该方法如果基于mask-rcnn需要增加的模型成本不大，但是对算力要求过高

最终由于时间和精力问题，B方法模型改动太过复杂，C方法太晚才看到，所以主要使用的是A方法，但也有改进，见下文

1. **基于tensorflow的代码实现**

基于上述A方案给出了代码实现，但没有使用9个透射参数的位置回归，依然使用原算法的4个位置回归参数。与原算法相比，初版主要的改动有：

1. 使用resize and maxpooling 代替原算法的roi-pooling层，这是因为tensorflow里面没有这个层的api实现，而且有文献表示使用resize and maxpooling的方法不会造成太大影响
2. 送给roi-pooling层的roi的128 batch只取自当前img，不从两张图来取
3. 为了能够检测小尺寸物体，移除了原作中过滤最小anchor尺寸为16的操作
4. 初版的anchor size不变，仍为8，16，32；anchor ratio选用了 1：5 ， 1：2 ，1：1， 2：1
5. 对应4）的修改，rpn的分类分支channel由原来的2\*9（anchor个数）=18变为2\*3\*4=24，位置回归分支channel由原来的4\*9=36变为4\*12=48，最后一层分类器和回归器也做了相应修改

调参过程：

1. 对训练集所有anchor进行了聚类，得到新的anchor ratio分别为：0.15，0.31，0.53，0.84，1.1，2；anchor size分别为：4，8，16，32，每个feature对应的anchor个数为4\*6=24个
2. 数据增强，在原只做水平flip的基础上再做垂直flip，使得总数据量翻为原来的4倍
3. 对应2）增加训练的轮次由70000增加到200000，并且学习率递减开始的轮次从50000调整到120000
4. 针对推理的效果图，发现检测出来有较多的重复框，即nms没有抑制住，修改推理过程的nms阈值由0.3减小到0.15
5. **下一步改进方向**
6. 增加验证集。虽然这方面的知识有学过，但是论文和库上的代码都是只使用了训练集和测试集，其中训练集是数据集中的原训练集+原验证集。在比赛时沿用了这种数据集分类方法。但是到提交的时候才发现，官方只允许做3次提交，所以在本地没有办法测试自己当前模型的性能，导致非常尴尬，只能人为的去观察做出来的图像来分析下一步改进点。没有分验证集是很大的一个失误和改进点
7. 加入基于语义分割的方法来改进模型，来覆盖非水平矩形的gtbox检测率，并且也是进一步学习语义分割算法的机会
8. 加入FPN的模型结构，来提高对小尺寸物体的检测率。这是因为近期读到综述文章，2017年检测领域成绩较好的单位的模型，全部使用了FPN的结构，该结构不仅仅使用了feature抽取网络的最后一层输出，还融合了较低维度的feature，使得较低维度的feature也能够有用武之地