项目总结报告

一、 回顾项目阶段步骤：

回顾整个项目经历了一以下几个步骤，这也是以后类似项目要所经历的步骤：

1、 选择检测模型

2、 下载合适的检测模型 model zoo，并导入测试

3、 选择分类模型

4、 下载合适的分类模型 model zoo

5、 用项目数据在 finetune 的模型上训练网络，并用网络汽车照片验证分类网络效果

6、 用检测网络预测出的框分割输入图片，输出多张只包含汽车的小图片

7、 对分割出来汽车小图片进行预处理，输出等大的、方便分类网络使用的一批数据

8、 将预处理好的数据输入分类网络，预测出每张汽车图片的车型

9、 将输入图片、框、车型文字合成结果图片

10、 将预测过程封装成方法，用 flask 方式发布成 web 服务，输入和输出都采用 web 形式

11、 对整个系统进行优化

12、 整理代码，编写并归集文档，录制项目结果视频

二、 项目心得经验：

很多具体细节的经验和心得，在一至四周的项目过程记录中都有详细的记录，这里就不再重复，以下只是抽取出几个印象特别深和比较耗时间的点再总结一下

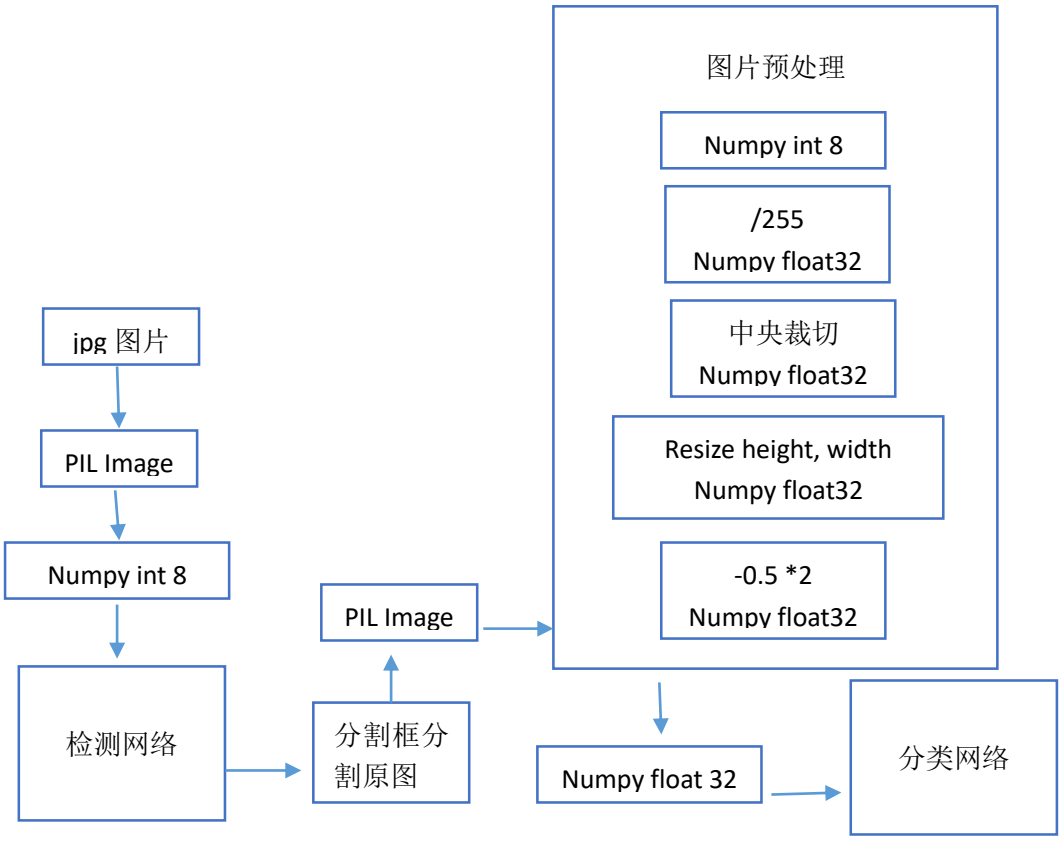
1、 拿到一个 AI 项目，需要了解项目的需求，根据已有的数据确定模型方案，本次项目就是受到数据所限，选择了检测，分类串行结构。因此，数据很大程度决定了项目的选型，这个是和传统项目不太一样的地方。

2、 用 model zoo finetune 训练网络的时候，注意 checkpoint 的哪些部分是需要

checkpoint_exclude_scopes 排除，哪些部分是需要 trainable_scopes 指定训练，或者全部节点训练。这个项目就需要排除一些冲突的节点，并且进行全节点训练的。

3、 训练网络的时候，学习率的调整很重要，注意观察 loss 的变化情况，调整学习率。

4、 Tensorflow 的机制特殊，基于图的计算，数据不能任意传输，feed 进入模型的只能是 numpy，要用 Tensor 的数据也要用 eval()进行转换，但是转换的速度非常慢，因此，在数据流动的过程的设计，需要充分考虑数据格式。比如这次项目各个阶段数据的流动格式如下：

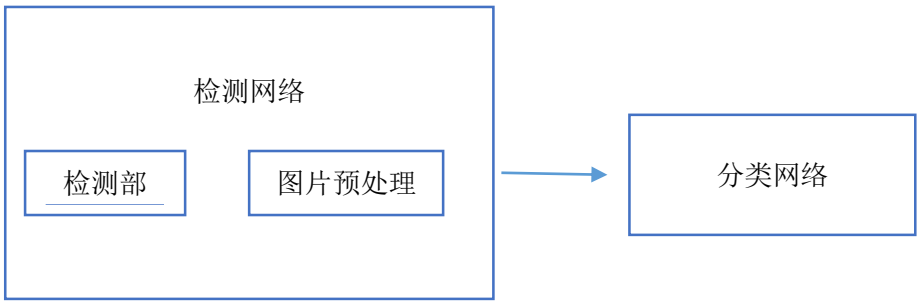


4、flask 发布 tensorflow 模型的时候，是可以将加载 Graph 和 Session 部分放在 request 之外，这样就不需要没次验证图片的时候都要重新加载一次，只需要跑验证过程，节省很多时间，但是有一点要注意的时，在启动 web 服务前，需要先调用一次预测，Graph 和 Session 才算是真正加载，这个我还是搞不明白为什么。

三、项目扩展展望

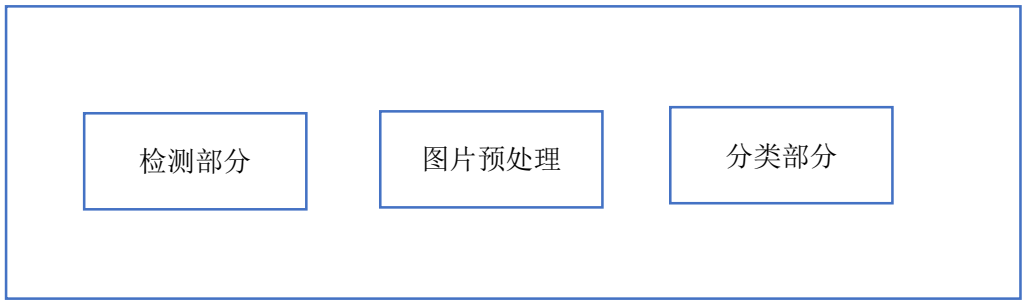
1、在项目完成后，可以再考虑用其他的项目结构来实现，以下考虑了两种结构

1)之前已经尝试过把图片预处理把入到分类网络的模型之中，由于输入的图片大小不一致，无法 feed 传入，因此把预处理抽取出分类网络之外，放置在检测网络与分类网络之间。同样的，也可以考虑把出片预处理部分连接到检测网络之中，如下图：



这样可以吧检测部分的代码和图片预处理的代码合在一起，并用 export_inference_graph.py 导出，冻结

2）按照这个思路，是否可以把 3 部分合在一起，尝试以下结构：



把检测部分、图片预处理、分类部分 3 部分代码合在一起，并用 export_inference_graph.py 导出，冻结。

以上两种方式，是尽量把数据流在 tensorflow 图中完成，提高运行效率。但是估计这样做会遇到很多问题需要克服。

2、既然是毕业项目，可以考虑租用一个云服务器，将这个 web 项目部署到云端，用微信小程序作为前端，通过微信社交工具可以给更多人展示自己的成果。

3、基于这个项目的经验，尝试做成 App 单机版本的，通过 tesorflow lite 等方式改写，不用通过把图片传到云端预测，而在手机移动设备本地完成整个车型的预测，但是由于手机的计算性能很低，需要使用 ssd、mobilenet 等运行速度更快的网络重新训练。

4、再扩展这个项目，最好的产品形态是，通过手机摄像头实时的拍摄现实视频画面，动态的在视频画面中完成车辆的检测、定位、和车型分类，并在换面中叠加各种 2D 的，3D 的信息，这个除了需要非常高效的算法和系统结构之外，还需借助很多其他领域的技术，例如 AR 技术，AR 也是我现在工作的领域，希望未来真的能把 AI 结合到 AR 和 VR 领域中来，这也是我的发展方向。