实习报告

刘钰纯

1. 引言

我在实习期间 (2018 年 5 月 21 日至 2018 年 8 月 15 日)，主要的工作是路面标志检测模型的实现。路面标志中，包含标识 (左右转箭头，限速标志等) 和标线 (单双线，导流线等)。经过一段时间的熟悉工作环境和搜集相关资料，我复现了 2017 年三星公司计算视觉实验室发表的 VPGNet 模型 [2]。这一模型能够在多种天气条件下，高效同时完成路面标记与标线的检测识别工作。

由于论文使用的数据集和代码文件都没有完整公开，我在实习期间，将现有公开数据集的图像标注进行了比较和人工处理筛选，依照论文描述和 Github 上的部分项目 [1]，补全了预处理和后处理部分代码。现有整个模型的训练使用了 BDD100K 数据库基础上经过校准补标的图像 4687 张，经过预处理过程增广为 9374 张，测试使用 Caltech lane 数据库中 1225 张图像。训练及测试集中包含 12 类禁止线标注。模型使用 caffe 框架，预测单张图像的速度在 $0.1s$ 左右，能够在一些光照缺乏等特殊条件下取得稳定的预测效果。

由于两个月左右的时间有限，没有能够在更大的数量级上对模型进行训练，同时也需要在更加多样的测试集上测试模型对于不同种类的标线分类准确度。在之后如果有更多的标注数据，或者对路面上的标识 (如限速标识等) 提出更高的要求，这个模型可以作为后续工作的基础或参考。为了对实习期间的工作做一个总结，同时以备万一，我对整个预处理和后处理部分的代码补充了注释，并且在本实习报告中进行了总结和说明。

2. 数据集和标注

在实习期间，我所使用和标注过的数据集主要有以下几个：

1．Caltech lane database：1225 张图像，该数据集场景较为单一，标线种类只有 5 种，但图片质量高。在模型完善后作为测试集，测试后处理方法的可行性。我对该数据集中所有图像标注了 VP 坐标，方便进行结果比对。

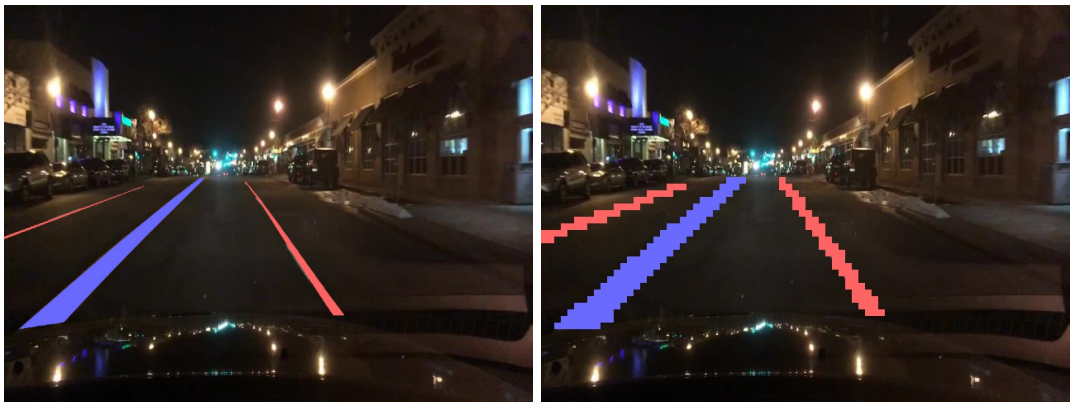
2．Apollo Database: 下载使用了官网提供的部分数据，该数据库需要翻墙下载，且图片质量很高，标注精确，数据量很大。所以现有下载 10248 张图像，每帧图像有两个机位拍摄。使用和训练过程中，发现由于其标注风格与论文中不一致，训练结果并不理想，并且图像中标线种类并不齐全，导流线等线类缺失，所以最后放弃使用做为训练。我对该数据集的单个机位图像进行了人工标注，含有 VP 坐标的有 4544 张，不含有 VP 的有 580 张。

3．BDD100K：下载使用部分数据，场景多样，标注风格统一。经过标注人员校准修改部分标识，得到 4687 张图像。用该数据集作为完整模型的训练数据，进行了图像翻转做数据增广。增广后得到训练集 7453 张，验证集 1921 张。含有 VP 图像 3229 张，缺失 1458 张。

数据标注方面，由于有的时候需要处理的图像较大，经过压缩成固定 640×480 大小的图像时，细长的线条会几乎无法看见，所以在论文中使用了 grid 标注，将图片分割成 8×8 的小格进行标注。

输入数据的格式如图2所示，首先输入标注图像的相对路径，之后输入的数据分别为:

{是否存在 VP(boolean 值) 横轴坐标　纵轴坐标　背景外格子标注的个数　左上点横坐标　左上点纵坐标　右下点横坐标　右下点纵坐标　格子标签　...　}。



(a) 像素级标注　　(b) 格子级标注

图 1: 像素标注有部分线条的标注不明显 (a)，预测难度大。修改为格子标注 (b) 后降低了预测难度。

```
/bdd_image/0c140522-227e6f11.jpg  1 349 218  44  353 233 360 240 9  353 241 360 248 9
```

图 2: 输入数据示例

由于需求方要求中，优先需要处理的是禁止类标线，所以在标注过程中一共标注了斑马线，双黄线等 12 类标注。然而在模型中，斑马线属于道路标识类，标注方式和标线不同，所以在训练过程中并没有对斑马线进行训练。

3. 网络结构

VPGNet 的网络结构分为四个分支，使其能够同时处理物体的检测（包括交通标识和车辆等），道路标线的检测和 VP 的定位。在训练的过程中，我发现四个分支相互影响，如果做标线分类，关闭 Grid Box 分支，会使训练难以收敛。在后处理阶段，道路标线分类只需要用到 Multi-label 分支，而交通标志则需要同时使用到 Grid Box 和 Multi-label 分支。在 Multi-label 分支的输出层中，大大多于我们所需要的分类种类，多余的是冗余层数。如果需要修改输出层数，则需要对整个网络的结构进行修改。修改到合适参数需要大量调试，所以在实习期间我并没有修改整个网络的结构，在分类的时候需要对输出进行筛选，才能得到需要的结果。

4. 训练过程

VPGNet 的模型结构有四个分支任务，在训练的时候，需要进行一系列调整才能得到较好的效果。

根据论文中描述，第一阶段，我首先单独训练了 VPP 模型，当损失函数逐渐收敛的时候再加入其他三个分支。在论文中提到，VPP 模型的训练对于其他分支会有一定影响。VPP 模型训练收敛之后，其他分支的损失函数也会降低 20%。在实际训练中，其他分支损失函数的确略有降低，但是并没有达到 20%。

第二阶段，同时开启四个分支任务。四个任务之间需要平衡，否则如果一个任务的损失函数权重过小，这一任务会变得依赖于别的任务；反之亦然。为了解决这一问题，文中使用公式1来计算整个模型的损失函数。

$$Loss = w_1 L_{reg} + w_2 L_{om} + w_3 L_{ml} + w4 L_{vp} \tag{1}$$

其中 L_{reg} 是 $L1$ loss,L_{ml}, L_{om}, L_{vp} 是 cross entropy loss。在论文中，所有权重开始都设定为 1，可以看到不同分支的损失函数，根据损失函数之间的差值调整权重，使得每个分支均衡。之后在训练过程中，如果出现不同分支的损失函数不均衡，则调整权重。重复这一调整过程，直到收敛。

Table 3. Proposed network structure.

Layer	Conv 1	Conv 2	Conv 3	Conv 4	Conv 5	Conv 6	Conv 7	Conv 8
Kernel size, stride, pad	11, 4, 0	5, 1, 2	3, 1, 1	3, 1, 1	3, 1, 1	6, 1, 3	1, 1, 0	1, 1, 0
Pooling size, stride	3, 2	3, 2			3, 2			
Addition	LRN	LRN				Dropout	Dropout, branched	Branched
Receptive field	11	51	99	131	163	355	355	355

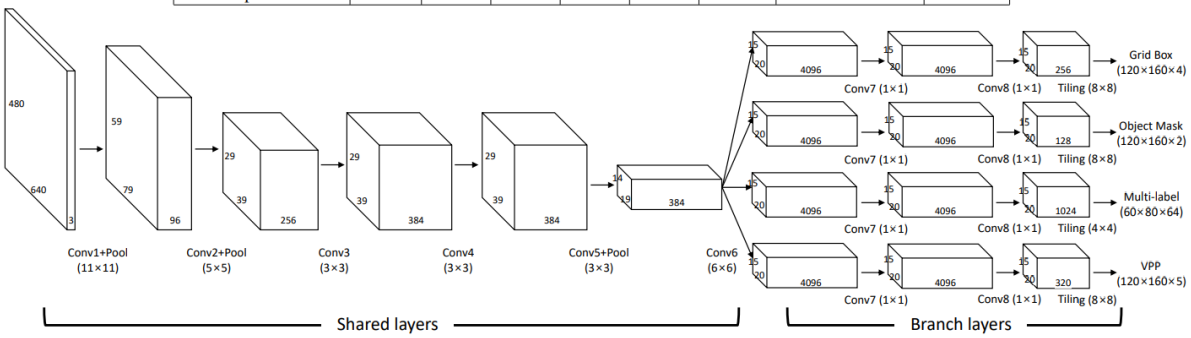


图 3: 网络结构图

在训练过程中，由于训练图像有限，训练时长并没有论文中长，总共训练 20000 轮，我只进行了一次调整，L_{reg} L_{ml}, L_{om}, L_{vp} 的权重比值调整为 $1:5:5:5$。

5. 后处理过程

对于 VPGNet 的复现，其中最有挑战性和和创新性的部分在于后处理。由于作者并没有在 github 上提供完整代码，需要对照论文中提及的方法描述，自行补全完善。该过程涉及许多尝试和实验，在整个实习过程中耗时较久。

灭点（Vanishing Point）． 根据论文中提到的方法，VPGNet 的 VPP(Vanishing Point Prediction) 模型输出了一个 $120 \times 160 \times 5$ 的矩阵。在五层中，四层分别标识该点在四个象限的置信度，而 VP 则处于四个象限的焦点。余下的一层表示该预测图像无 VP 的概率。我们需要首先判断该图是否具有 VP，之后找到一个点，在该点处四个象限的置信度相近。

$$P_{avg} = \frac{1 - \frac{\sum p_0(x,y)}{(m \times n)}}{4} \tag{2}$$

$$loc_{vp} = \arg\min_{(x,y)} \sum_{n=1}^{4} |P_{avg} - p_n(x,y)|^2 \tag{3}$$

其中 P_{avg} 为图片中存在 VP 的概率，当 VP 不存在的平均概率 $\sum p_0(x,y)/(m \times n)$ 值接近于 1 的时候，P_{avg} 的值接近 0; 当平均概率值接近于 0 的时候，即 VP 极有可能存在时，P_{avg} 的值接近 0.25。所以在编程中我设定，当 P_{avg} 小于 0.125 时，判定图片中不存在 VP。

$p_n(x,y)$ 是点 (x,y) 在第 n 层的置信度（$n=0$ 时是指预测 VP 不存在的概率的一层），$m \times n$ 则是输出矩阵的大小，在模型中为 120×160。loc_{vp} 则是 VP 所在位置。

在对这一方法进行实验的初期，使用了 Apollo Database 的图像进行了训练。该数据集中总共含有 10248 张图片，由于每帧含有两个摄像机位的图像，两个机位图像之间差异不大，为了人工标注的方便，我只选取了其中一个机位拍摄图像进行了处理。总共得到 4544 张含有 VP 的图像和 580 张缺失图像。基于这些标注得到的 VP，训练 VPP 模型，测试不同的后处理方式得到结果的优劣。

由于 Apollo 数据库中图片的拍摄角度及拍摄情景较为单一，一开始得到的训练结果并不理想，我改写使用了信息熵的方式，选定信息熵最大的几个点为可能点，求这些可能点的坐标均值得到 VP 位置。这一方法一定程度上改良了预测结果，但是改良程度不大，在一些较为简单的场景下依旧会预测错误。

为了解决这一问题，我在 Apollo 数据集的基础上，请标注人员人工标注了 bdd100k 数据库中的 4687 张图像，其中具含有 VP 的图像有 3229 张，缺失 1458 张。bdd100k 数据库中多样的数据和拍摄情景弥补了之前训练的缺陷，得到的结果在 Caltech lanes 数据集上得到了很好的提升，得到的测试准确率为 0.84，满足了之后做后处理的基本需要。由于信息熵计算时间较长，结果也并没有在更大数据集上有提高，所以在 VPP 模型的后处理上，还是使用了论文中提到的直接计算的方法。

道路标线. 在道路标线方面，主要使用了取点，视角变换和标线回归。后处理分为四个步骤：

首先，我在多标签概率图上取了一些局部最优点作为种子点，这些种子点是我们之后进行标线分割的重要基础。取点过程在论文中并没有详细解释，根据作者在 Github 上的问答记录，我通过尝试复现了取点过程。其中编程使用的一部分超参数是实验得到，可能与论文不符，但在我所使用的测试集上得到了较好结果。根据作者在回答相关问题中的描述，多标签预测模型 (Multi-label Prediction Model) 的输出为 $60 \times 80 \times 64$ 的矩阵，64 层中第一层为每一个格子 ($8 * 8$ grid) 作为背景的概率，其他为各个标签的概率 (有部分冗余)。将除了第一层外的其它层求和，得到如图4所示的有道线的概率图。

(a) 原始图像　　(b) 预测得到概率图

图 4: 给定一个输入图像 (左)，我们根据模型得到 $60 \times 80 \times 64$ 的预测结果。将得到的矩阵除了第一层背景概率之外的 63 层求和，得到输出的概率图像 (右)。

利用概率图，可对每一行截面做使用滤波函数，取到局部最优点。而这些局部最优点经过筛选，可以作为种子节点方便下一步处理。在 Github 上，作者提到他使用了 scipy 包中 fftconvolve 函数作为滤波，得到图5中所示的蓝色曲线，再在该函数曲线上求局部最优点（红色点）。在实验过程中发现，有一部分局部最优点所对应的函数值较低，如果纳入考虑会使得种子点的个数过多，影响后续处理。因此，我对所取到的局部最优点进行了筛选，其所对应的概率值低于 0.5 则不纳入种子点。以图5右图所示，所得到的四个局部最优点中，只取中间两个点为种子点。

在得到种子点后，需要对这些点进行聚类。由于在道路标线随着视觉延伸会最后交汇于灭点，标线之间的距离会不断变小，影响聚类效果，所以通过透视变换来分离 VP 周围的种子点。论文中提到使用了 IPM(Inverse Perspective Mapping) 逆透视变换，而该方法需要设定一些相机参数，由于无法测量，所以我实验了几种方法，共同缺点是：都需要通过设定一些超参数来调整变换效果。而在拍摄图像的设备调整之后，无法自动调整变换角度，所以这仍是需要改进和提高的部分。

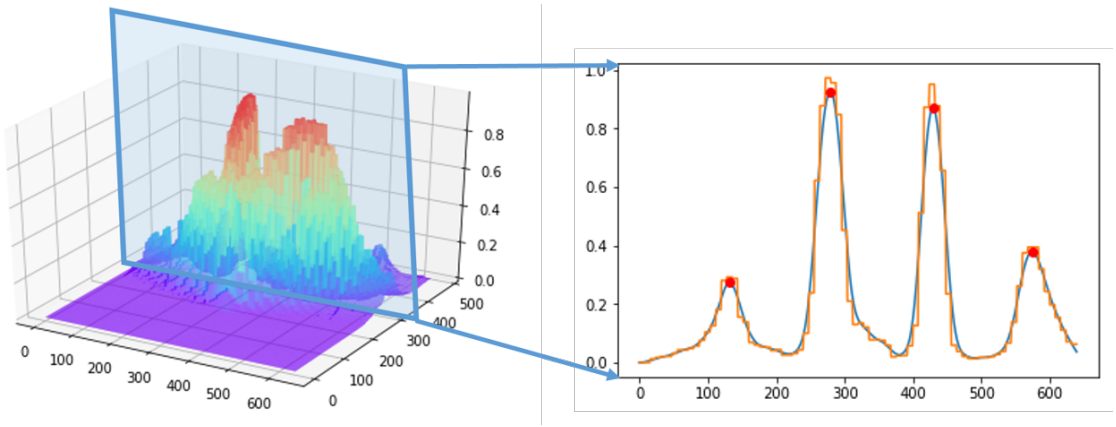


图 5: 根据概率图 (左为概率图的三维表示)，可以取每一行的概率，得到截面 (右)。由截面经过滤波可以得到所有局部最优点，这些局部最优点 (红色点) 可以作为种子点。

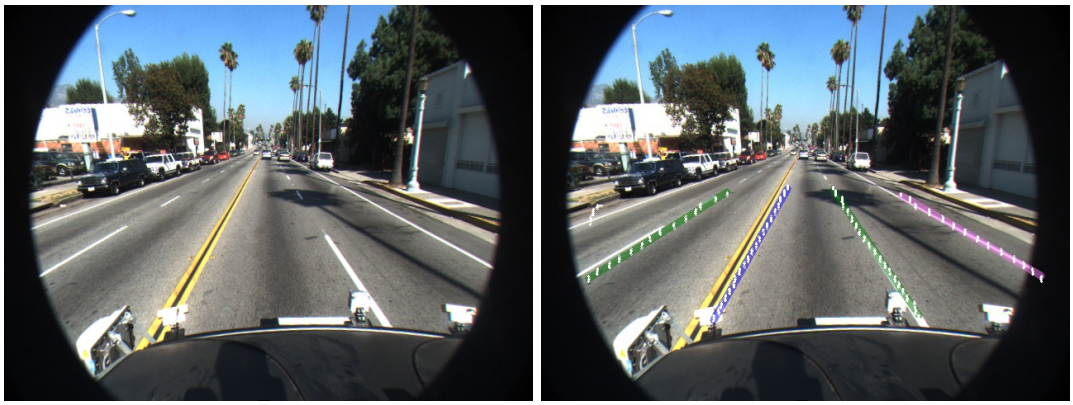
在现有的方法中，我实验得到的效果比较好的方法来自于 JonathanCMitchell[3]在 Github 上写的结合 VP 的逆视角变换方法。该方法主要是利用 VP 坐标与预估的路面宽度，分割得到路面的梯形区域。根据梯形区域的四个定点坐标，与变换后的图像四个顶点位置，得到逆视角变换的变换矩阵。利用变换矩阵，得到每个种子点在变换后图像上的投影。该视角变换方法在测试集上虽然能够达到将 VP 周围点分离的作用，但是只适用于较为粗糙简单的分离任务，仍需要改进和提高，才能适应不同设备，不同拍摄角度的实际环境。

得到各个种子节点在视角变换图上的投影后，可以对这些点进行聚类。这一步的主要作用在于：将距离相近的种子点聚为一类，这样预测标线种类的时候，不会出现一条标线上有多种不同的类别划分。论文中阐述所使用的聚类方法是基于密度的聚类方法，并且对于聚类方法进行了改写来提高聚类速度。而在我实际操作当中，我认为作者描述所使用的改进方法没有降低算法的复杂程度，而且在种子节点的数量有限的情况下，这种改进可能无法有显著效果。所以我在代码实现中还是使用了未改进的聚类方法。在得到聚类成果之后，对每个类里的种子点进行投票，以多数种子点的类别决定整个类的的类别。

最后使用逆视角变换矩阵的转置，将视角变换图上投影的种子点转换回原始图像上坐标，并且根据聚类得到的结果，画出不同类别，画出得到不同种类的标线。整个后处理的速度在每张图 0.1s 左右。时间上比论文中的速度慢，但是已经可以达到使用要求。中间仍有很多可以继续完善的部分。

(a) 划定路面区域　　(b) 原始图像　　(c) 逆视角变换后

图 6: 给定一个输入图像，利用预测得到的 VP 坐标，划分出路面区域，如绿色梯形所示 (a)。利用划分出的路面区域顶点坐标，求得逆视角变换矩阵，将原始图像中种子点的坐标 (b) 投射于变换后的图像上 (c)。



(a) 原始图像　　(b) 预测结果

图 7: 由原始图像 (a) 通过模型预测和后处理得到预测结果 (b)，其中蓝色为单黄线，绿色为虚线，粉红色为实线。线上白色点为后处理时的种子点。

6. 总结

在经过接近两个月的摸索和试验中，我基本完成了整个模型的复现。复现的这个模型的优点在于能够识别道路的标识和标线，之后的可用性高，如果之后需要对于更多的标识进行识别，只要在数据输入层做少量改变。并且，后处理的方式对其他道路标线识别模型也有借鉴意义，如 Deeplab 模型中，由于分割任务要求精度高，经常出现同一条标线中有多个类别，而寻找外部轮廓再投票的方法效率较低，可以考虑使用该后处理方法改进。

该模型也存在很多缺点：首先是训练的过程有技巧，需要提前训练 VP，而且要对于四个分支进行平衡；其次，后处理方式依赖于 VP 的识别准确度，如果 VP 的位置偏差过大，则会严重影响结果；最后，后处理中对于超参数的设定是通过调整实验得到的，可能无法适应拍摄角度充满变化，道路情况复杂的极端情况。

模型代码已上传至 209 服务器，后处理代码部分已有完整注释。文件夹下保存路径为

```
/liuyc
└── /model_report
    ├── vpgnet_report.pdf
    └── vpgnet.tar.gz
        ├── bdd_data/   图片及标注路径
        └── caffe/      模型路径
            ├── models/
            │   └── vpgnet-novp/
            │       ├── train.sh
            │       ├── ipm_img/    预测结果存放路径
            │       ├── post_processing.py 后处理代码
            │       └── snapshots/ 训练得到模型的保存路径
            └── tools/ 输入层代码
```

参考文献

[1] S. Lee. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition(iccv 2017). https://github.com/SeokjuLee/VPGNet. 1

[2] S. Lee, I. S. Kweon, J. Kim, J. S. Yoon, S. Shin, O. Bailo, N. Kim, T.-H. Lee, H. S. Hong, and S.-H. Han. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 1965–1973. IEEE, 2017. 1

[3] J. Mitchell. Advanced-lane-line-detection. https://github.com/JonathanCMitchell/Advanced-Lane-Line-Detection/. 5