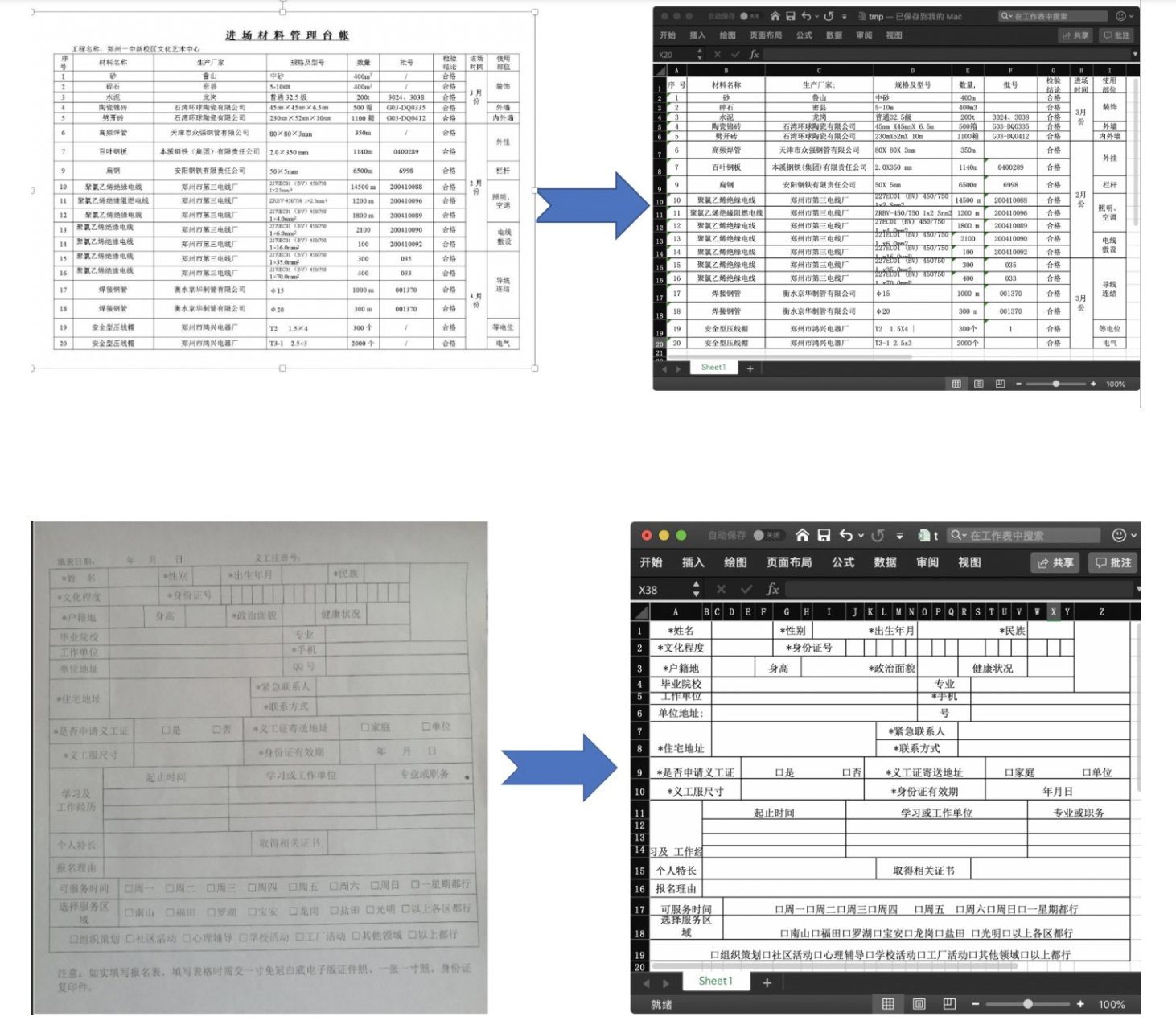
**表格图像识别技术**

表格图像识别有较高的商业价值，一般都在付费的专业OCR软件中才能体验到：比如百度的电子表单识别。问题在于付费的 OCR 软件没有开源，不方便定制和修改。

****

遍阅近几年比较有实操价值的论文，可分为以下三种思路：

1. 利用OCR检测文本，从文本框的空间排布信息推导出有哪些行、有哪些列、哪些单元格需合并，由此生成电子表格；

2. 运用图像形态学变换、纹理提取、边缘检测等手段，提取表格线，再由表格线推导行、列、合并单元格的信息；

3. 神经网络端到端学习，代表工作是TableBank，使用image to text技术，将表格图片转为某种结构化描述语言（比如html定义表格结构的标签）。

经过实验，发现以上三种思路都有不便落地的缺陷：

思路1：极度依赖OCR检测结果和人工设计的规则，对于不同样式的表格，需做针对性开发，推广性差；

思路2：依赖传统图像处理算法，在鲁棒性方面较欠缺，并且对于没有可见线的表格，传统方法很吃力，很难把所有行/列间隙提取出来；

思路3：解决方案没有次第，一旦出现bad case，无法从中间步骤快速干预修复，只能重新调整模型（还不一定能调好），看似省事，实则不适合工程落地。

**解决方案**

**针对已有方案的缺点和优点，我觉得可行性的解决方案流程如下：**

1. 对表格图片应用深度学习进行图像分割，分割的目的是对表格线部分进行标注，分割类别是4类：横向的线，竖向的线，横向的不可见线，竖向的不可见线，类间并不互斥，也就是每个像素可能同时属于多种类别，这是因为线和线之间有交点，交点处的像素是同属多条线的。

2. 对分割图分别做几何分析，即先提取连通区域，再对连通区域拟合折线，再对游离的线段根据距离和倾角进行合并形成框线。由于拍摄角度或者纸张的弯曲，一般原图表格会有一些倾斜，可使用投影变换（perspective transformation）对原图进行校正，使得横框线校至水平，竖框线校至竖直。

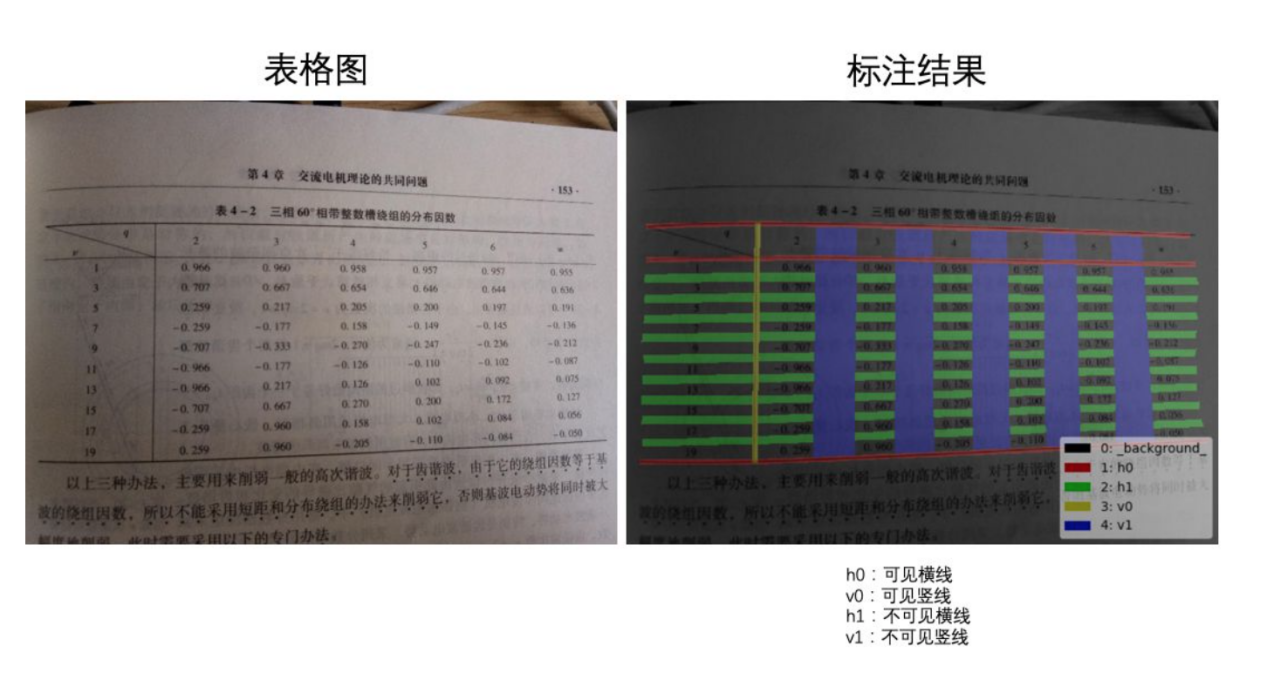
3. 对校正后的图调用OCR，识别其中的文本内容，以及每个字符的坐标。

4. 根据第2步得到的框线，计算出有哪些行，哪些列，其中哪些单元格跨行列合并了。由此得到每个单元格在图中的位置（top\_left, top\_right, bottom\_left, bottom\_right）四点坐标。

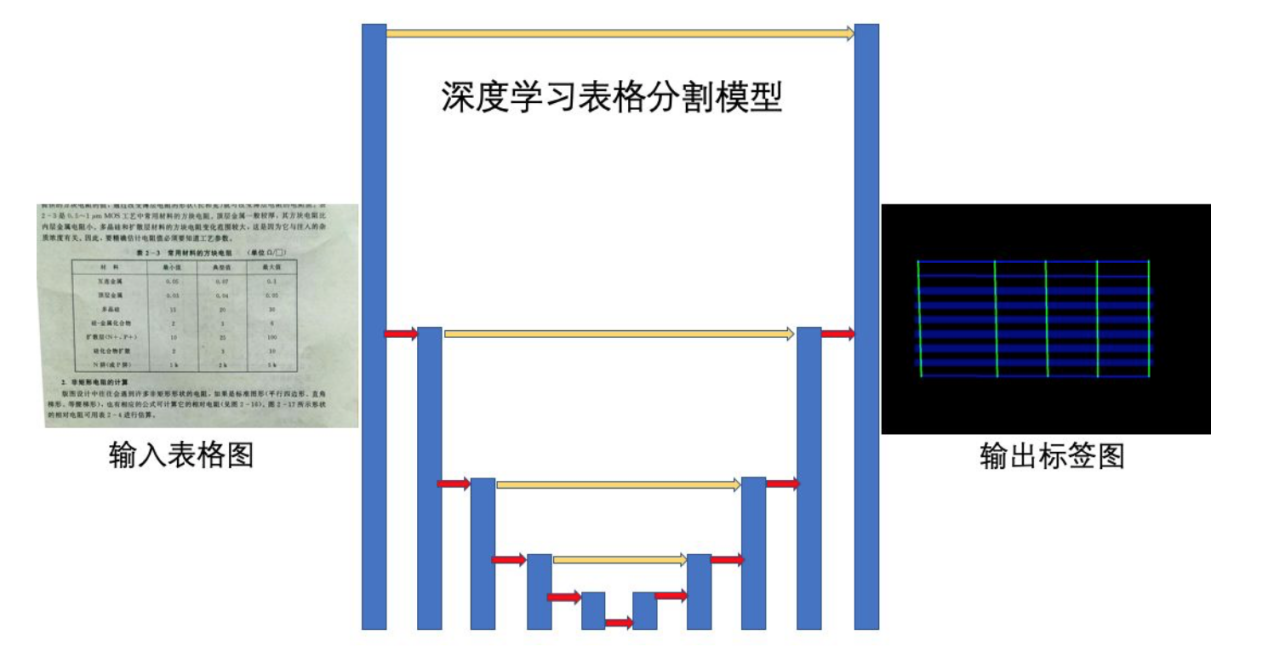
5. 将单元格位置，与字符坐标进行匹配，决定每个字符在哪个单元格中。最后计算每个单元格的字号大小，对齐方式等格式信息。​

**2.1 图像分割模型**

图像分割（segmentation）旨在对图像的每个像素赋予标签。在这里，我们的分割任务有多标签，每个像素可能属于横线、竖线、不可见横线、不可见竖线。



目前较常用的深度学习图片分割模型有DeepLab系列，fcn，Unet，SegNet等。

****

**2.2. 分割结果几何分析**

对分割结果设定阈值0.5进行二值化，转成几张二值化图，分别表示每种线所属的像素。接着对每个二值化图求连通区域。对连通区域进行过滤，长度太小的丢弃。

对剩下的每个有效连通区域，分别拟合折线，即得到大量线段。对线段的角度进行统计，横、竖两种线段与x轴的夹角均值应接近0和90度，若否，则认为识别失败并终止。在横、竖线段中，若有角度偏离均值3个标准差以上的，则过滤掉。

对于剩下的线段，应用DisjointSet算法进行合并，被合并的线段构成一条新的长直线，这些直线代表框线。两线段合并的判定条件是：夹角小于15度，并且一条线段的端点到另一条线段的距离小于一定阈值。

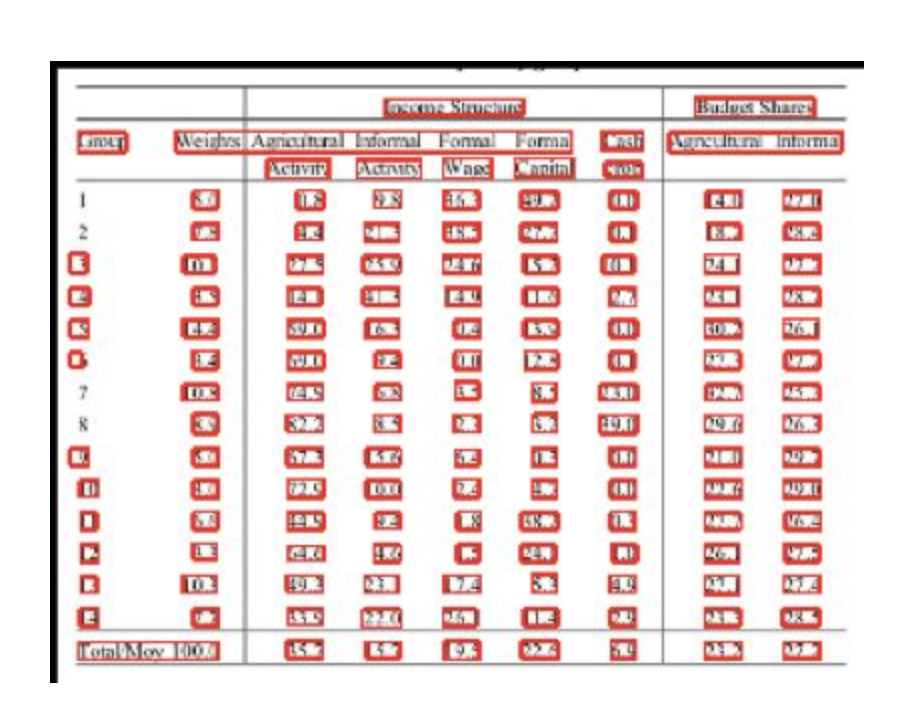
最终得到的若干直线，就是表格的框线。但是手机拍摄的照片一般都有一些倾斜，为便于后续处理和提高OCR结果的质量，我们将对图片进行倾斜校正。

校正方法使用投影变换，也即拟合一个单应矩阵H，使得HX=X'，X的每一列是在每条直线上以固定距离采样的点的齐次坐标，X'的对应列是该点校正后的齐次坐标。横线校正至水平，也即线上所有点的y坐标一致；竖线校正至竖直，也即线上所有点的x坐标一致。最后将求得的投影变换应用到原图中，将图片也校正。

2.3 OCR

将校正后的图片送去OCR，可得到图中每个字符的坐标。OCR 模型可以使用

CNN+RNN+CTC的模型结构。



**2.4. 识别表格结构**

接下来需要识别表格的结构，以跟OCR结果进行匹配。我们对一个完整的表格定义如下：

1）所有单元格，单元格定义为[起始行，结束行，起始列，结束列]

2）每一行的行高（像素）

3）每一列的列宽（像素）

4）每个单元格的字号大小（像素）

5）每个单元格的对齐方式（left\right\center）

6）每个单元格的文字内容

表格的结构是指1），2）和3）。我们提出一套高效的算法从表格线推导出每行（列）的高（宽）和所有单元格的坐标。

由表格框线推导行（列）的高（宽）比较容易，只需对所有的横（竖）线按从上（左）到下（右）排序，相临框线形成一行（列），所以只需计算相临框线的y坐标（x坐标）差即可。

由表格框线推导单元格坐标就不太容易了。因为现实中存在很多单元格合并的情况，一个单元格可能跨了若干行和若干列。

对此可以列举所有的单元格候选，每个单元格表示为（起始行，结束行，起始列，结束列），然后对所有单元格按面积从小到大排序。接着遍历排序好的候选单元格，去判断其上下左右的框线是否都真实存在，若存在，则此单元格就在原图存在。

注意到，每当确立一个单元格存在，所有与其共享起始行和起始列的其他单元格则不可能再存在，因为我们不考虑单元格中套着单元格的情况。所以虽然单元格候选集很大，但我们可以利用这一性质在遍历过程中进行剪枝，所以会很高效。

**2.5 匹配文字内容，确定字号和对齐方式**

2.4定义的表格还有4）5）6）没有识别。经过以上步骤，我们已经得到每个单元格的坐标和每个字符的坐标。接下来就只需进行对号入座就可得到每个单元格中的文本，也即解决了6）。

字号可由OCR文本高度确定，但是由于返回的高度总有一些不一样，实际表格中常常不会有太多字号，经常是同一列的单元格用一样的字号。因此我们对所有得到的文本高度进行聚类，当两行文本高度比例在[0.91, 1.1]之间，就可以认为是同个高度。聚好类后，对类内高度求平均值，以平均值做为此类所有文本的真实高度。

最后将文本高度换算为字号，由此4）也解决了。最后根据文本在单元格中的位置，判断每个单元格的对齐方式，对于对齐方式，也采取类似的聚类方法来去除噪音。由此5）也解决了。

至此，表格的所有单元格，每一行的行高，每一列的列宽，每个单元格的字号大小，每个单元格的对齐方式，每个单元格的文字内容都已经识别出来了。只需将单位换成Excel、WPS或者腾讯文档的标准单位，就可以转成电子表格了！