1. 1,在网上找图片（训练样本） (文件夹：字体100)
2. 从图片库里对每种字体进行分割(build\_hanzi.m)
   1. 2.1 图片灰度化
   2. 2.2 二值化(黑白图)
   3. 2.3 通过二值图找到文字所在（只要全0的行就不是文字所在）
   4. 2.4 对每行文字进行分割
   5. 2.5 对分割后的每行文字进行分割，从第一列像素开始，如果找到有1的列开始进行分割，记做n，采用文字高度比例对第n+h\*r列进行标记并判断，其中h为文字高度，r为比例，如果该列含有非0元素 则往后移动一格，直到全部为非0元素时停止分割。 从当前分割停止列的后一列开始 进行重复运算
   6. 2.6 对所有分割下来的文字先统一大小（30\*30）并以cell形式储存
   7. 同时记录每个文字图片对应的文字字符 保存为str
   8. 2.7 对每种字体所有保存的的文字以struct形式保存
3. 神经网络训练
   1. 读取训练样本
   2. 对每个样本向量化并归一化
   3. 创建目标矩阵，大小为n\*m， 其中n为每个字体的种类，m为每种字体训练的字数，矩阵每列除了目标字符的元素为1 其余全0
   4. 训练神经网络，nnstart生成的神经网络代码，采用 Scaled conjugate gradient backpropagation（共轭梯度反向传递），神经网络隐藏层为10层
   5. 获取训练好的神经网络
4. Hog特征匹配
   1. 读取训练样本
   2. 对每个样本进行二值化
   3. 提取hog特征 储存为矩阵形式
   4. 创建目标向量
   5. 用SVM（one\_vs\_one）进行训练
   6. 获取训练好的分类器
5. 字符识别
   1. 对新输入的图像进行灰度化 然后二值化
   2. 先对二值化后的图像进行卷积（功能相当于图像膨胀）
   3. 然后找到联通点
   4. 对所有联通区域进行分割
   5. 分割之后的文字图像统一大小（30\*30）
   6. 对文字图像向量化并归一化
   7. 用训练好的模型（神经网络或svm分类器）进行识别
   8. 保存结果