软件详细设计书

天猫网店营业执照识别系统

黄晓佳、贾晓玉、郑传奇

2018/6/29

目录

1 绪论

1.1 背景

互联网的高速发展使得电子商务得到了普及，从而推动了网店的大量出现与发展，电子商务成为现代商务中不可缺少的组成部分。此外信息技术与经济社会的交汇融合引发了数据迅猛增长，数据信息已成为国家基础性战略资源。

天猫网店上传的工商营业执照为不限格式的图片。在此背景下，根据国家工商总局《网络交易管理办法》对网店营业执照信息进行公示的要求，如何对天猫网店经营者在天猫店铺上以图片形式进行公示的营业执照信息进行结构化处理，提取出图片中的价值信息如企业注册号与企业名称形成结构化文档对于网上店铺的综合管理与监督具有重要意义。

光学字符识别(OCR,Optical Character Recognition)是指对文本资料进行扫描，然后对图像文件进行分析处理，获取文字及版面信息的过程。图片文字识别的研究属于 OCR 技术的一种，传统的处理方法为单个切分字符，人工提取特征，进行模板匹配。而近年来深度学习作为机器学习研究中一个新的领域发展迅猛，对各行各业都产生了深远影响，其动机在于建立、模拟人脑进行分析学习的神经网络，它模仿人脑的机制来解释数据，例如图像，声音和文本。深度学习通过建立类似于人脑的分层模型结构，对输入数据逐级提取从底层到高层的特征，从而能很好地建立从底层信号到高层语义的映射关系。

近年来，谷歌、微软、百度等拥有大数据的高科技公司相继投入大量资源进行深度学习技术研发，在语音、图像、自然语言、在线广告等领域取得显著进展。从对实际应用的贡献来说，深度学习可能是机器学习领域最近这十年来最成功的研究方向。深度学习模型不仅大幅提高了图像识别的精度，同时也避免了需要消耗大量的时间进行人工特征提取的工作，使得在线运算效率大大提升。

如果需要统计某家企业在天猫平台的网店经营情况，就需要先把天猫平台的网店营业执照信息采集下来，对图片中文字信息进行识别和结构化处理。利用深度学习技术实现图片文字识别意义重大，因此本团队依据赛题要求设计并实现了一种基于深度学习的天猫网店工商营业执照图片信息采集与自动识别系统。

1.2 项目内容

本项目的主要目标是根据赛题要求，设计一个天猫网店营业执照图片的采集与文字识别，结构化处理一体化的系统。

项目的核心算法为深度学习算法 LSTM，在开源开源图像识别引擎 Tesseract 的基础上，通过针对本系统需识别的天猫工商图片的特征进行字符库训练，并修改 Tesseract4.0 所采用的 LSTM 深度学习算法进行 ReLU 优化，提高了对中文的训练准确度。

本系统包括对本地中存储图片的文件夹路径自动读取并按顺序识别图片，应对特殊需要，本系统也可提供天猫网店营业执照自动化爬取。系统可以匹配各类常见图片类型。识别并提取出图片中的企业注册号、企业名称数据项，并保存进 Excel 中。

用户可以对每次的识别任务进行后续查看与处理或直接交付。图片识别采用深度学习技术，在进行识别时，用户可随时查看当前识别进展。一次识别任务结束后，对于难以识别的图片为保证系统健壮性会单独列出让使用者自行识别。

1.3 项目特色

1.3.1 基于用户辅助模式的系统流程设计

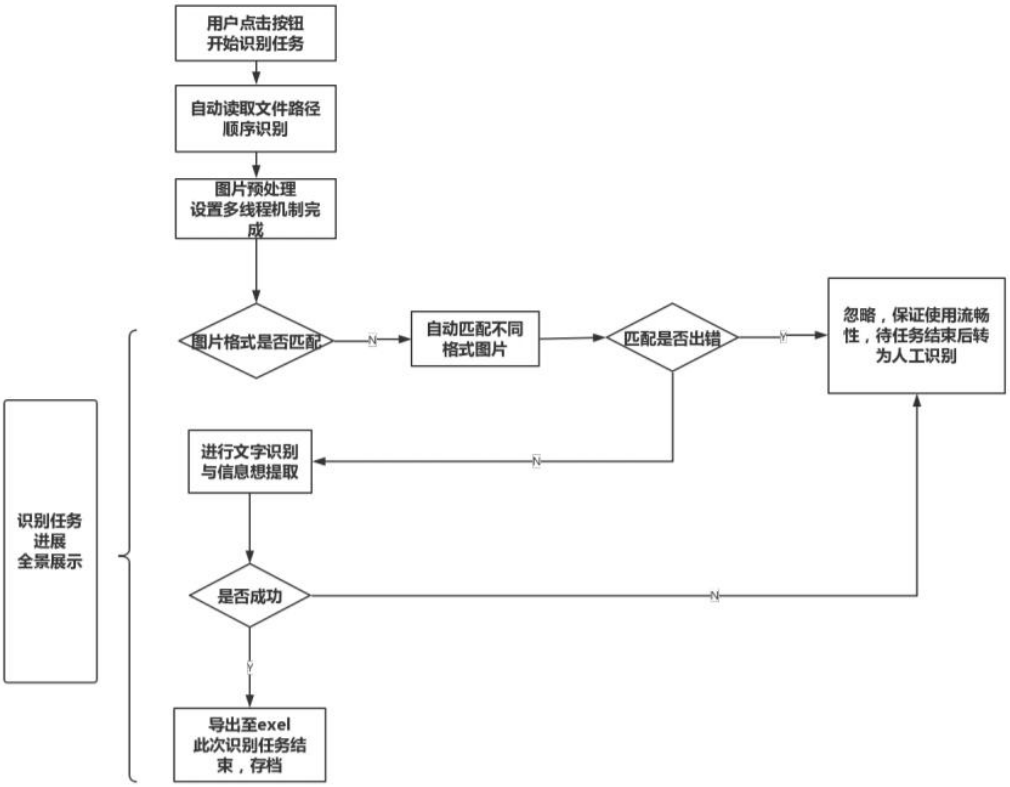


图 1-1 系统流程设计图

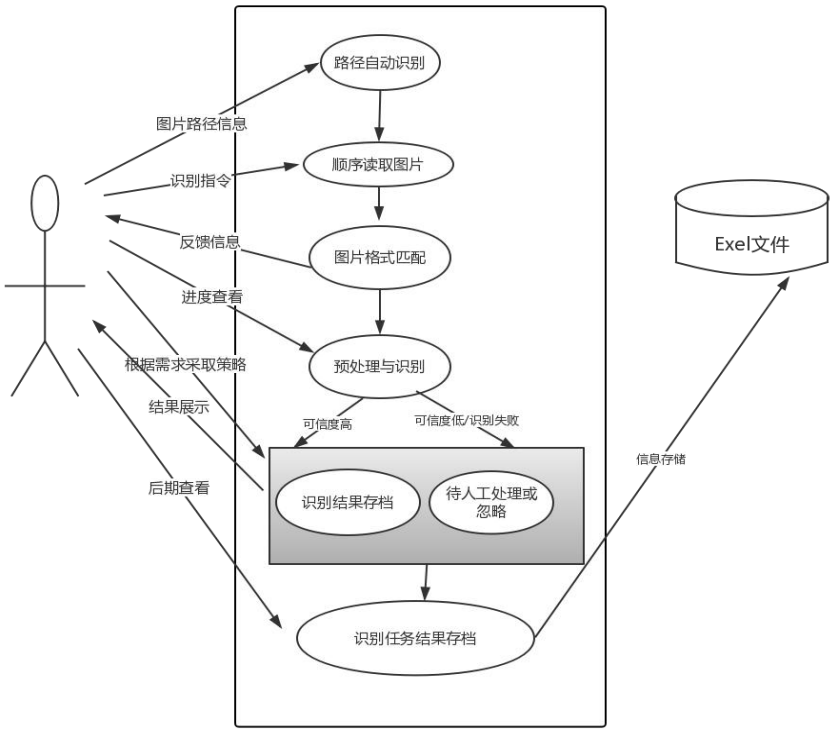


图 1-2 系统用例图

整个识别系统的体系结构设计基于用户辅助模式，软件对图片的识别过程用户可实时获悉当前进度状况，并可对识别任务加以干预，如暂时中止，重新恢复等。新建识别任务根据需求设置可信度阈值，当图片识别的文字低于该可信度时，用户界面动态显示，识别任务过程中系统为保证流畅性自动忽略，识别任务结束后，界面展示此次识别任务的详细信息，用户可根据自身需求选择对未成功识别与识别可信度低的图片的处理方式，放弃识别或人工进行校验。此外根据系统用户的不同准确度要求，用户可自定义置信度阈值，从而既保证了识别出的信息项的准确性有兼顾了不同需求用户的使用体验。

当用户完成一次识别任务后，系统会自动将其归档在该用户的识别任务记录文件中，包括识别任务的 Excel 结果和识别完成度的详细参数统计，包括识别总图片数，系统难以识别的图片数目、识别失败的原因、是否已人工校验等。方便用户对数据进行二次利用。

1.3.2 基于 CNN 神经网络设计的图片数据集获取模块

本系统采用深度学习机制进行 OCR 识别训练，由于深度学习是基于大量带标注的训练数据集组成，因此数据集的获取也是一个重要的模块。我们通过研究发现天猫商铺的验证码长度固定为 4 位，又大写字母和小写字母以及 0--9 数字组成。而天猫网为防止商铺数据被爬取，对字符有进行变形和粘连，这导致采用传统的基于字符切割的模板匹配算法难以发挥作用。

基于以上特征分析，本项目采用基于深度学习机制的 cnn 神经网络设计算法，神经网络的设计如下：输入层为 100*30 的像素矩阵，对该输入值经过两层卷积与两层池化处理后，生成 1024 个神经元，经过全连接层，采用激活函数将其归类。在实际训练中，迭代训练约 8000 轮后，准确率达到 92%。

1.3.3 基于 LSTM 深度学习网络的 Relu 优化文字识别算法

本系统采用了基于 LSTM 深度学习网络的 Relu 优化文字识别算法用于训练与文字识别。前期直接采用 Tesseract4.0 识别图片，准确率不能达到要求，因此需要一定的训练来提高准确率。但是采用原有的激活函数进行训练，需要庞大的训

练样本以及漫长的训练时间，为了解决这个问题，我们采取了基于 LSTM 深度学习网络的 Relu 优化文字识别算法，即使用 TLU 作为训练时的激活函数，TLU 具有函数 ReLU 和 LReLU 的优点 ，同时在变量取值大于 0 时导数为常数，因此在饱和区内的梯度永远不会为 0，能够有效缓解梯度消失问题。TLU 能显著地加快深度神经网络的训练速度并有效地降低训练误差。

1.3.4基于 Java 多线程机制的多周期运行模式

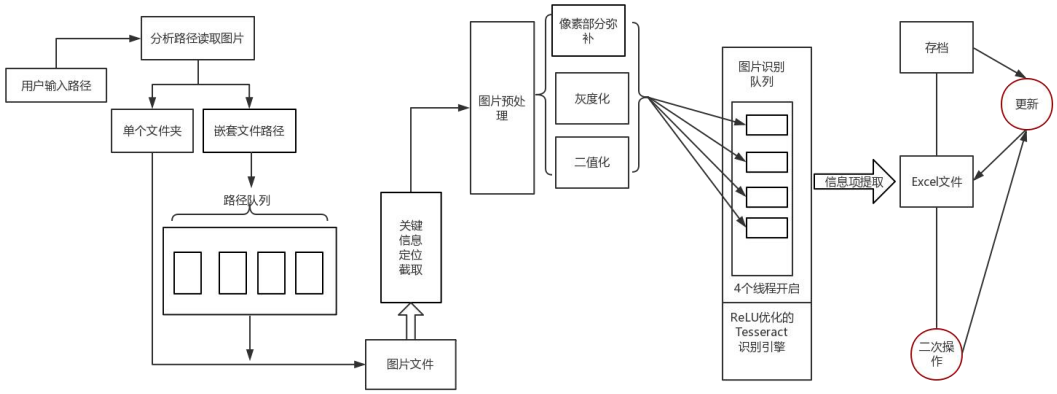


图 1-3 双周期运行模式

本项目整个系统架构分为：图片文件读取运行、定位截取与预处理周期、图片识别与信息项提取，Excel 归档周期双周期。整个软件的各个周期的工作流程高度并行化，系统的运行机制合理有序，为识别任务的高效稳定完成提供可靠的保障。

1.3.5 基于识别任务的系统存档与管理机制

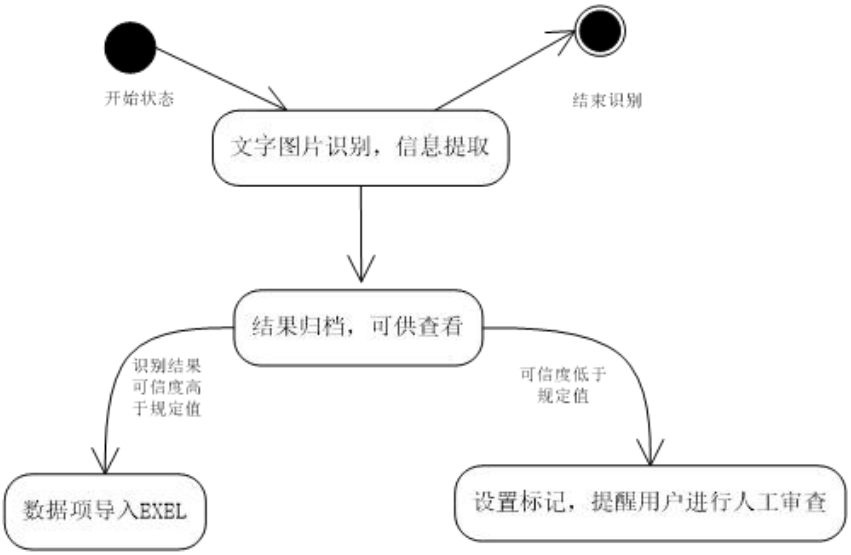


图 1-4 识别任务存档与管理状态机

本系统提供图片识别的结果归档与纠错，设置阈值，对于最终识别的字符中可信度较低的置以标记提示用户可进行人工审查以确保最终结果的可靠性。对于识别结果满足可信度的数据项自动保存入 Excel 表中进行存放。达到效率与准确的协调。系统将用户的历史识别任务的完成情况与 Excel 文件存储在本地文件，可随时查看并进行二次操作，保存更新，对不在需要的文件与任务完成记录可删除。

2 系统介绍

2.1 系统用户特点

本项目适用于任何有对天猫网店营业执照信息的企业注册号与企业名称数据项有分析需求的用户。

2.2 功能性需求

2.2.1 图片路径读取自动化

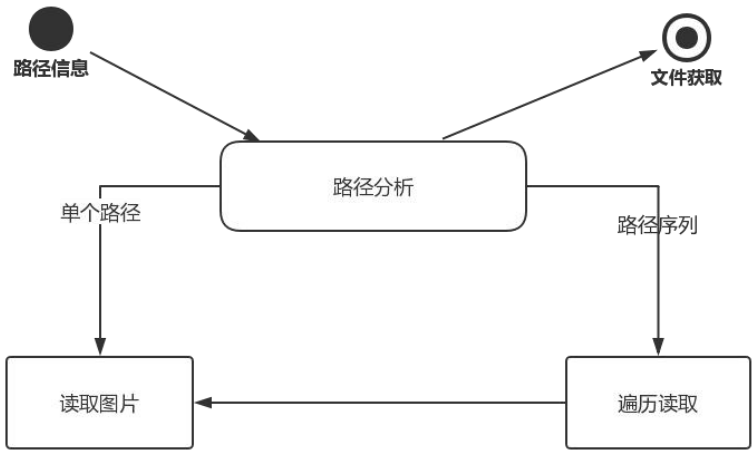


图 2-1 路径读取状态机

系统可以自动识别图片存放的文件夹路径，并从文件夹路径下顺序取出图片进行识别。

当系统获取到用户输入的文件路径时进行路径分析，如果是单个路径，直接获取该路径下的图片文件，如果为路径序列，则使用遍历算法依次获取路径下的图片文件。

2.2.2 图片顺序识别

系统从文件夹中顺序识别图片文件。

2.2.3 多种格式图片匹配

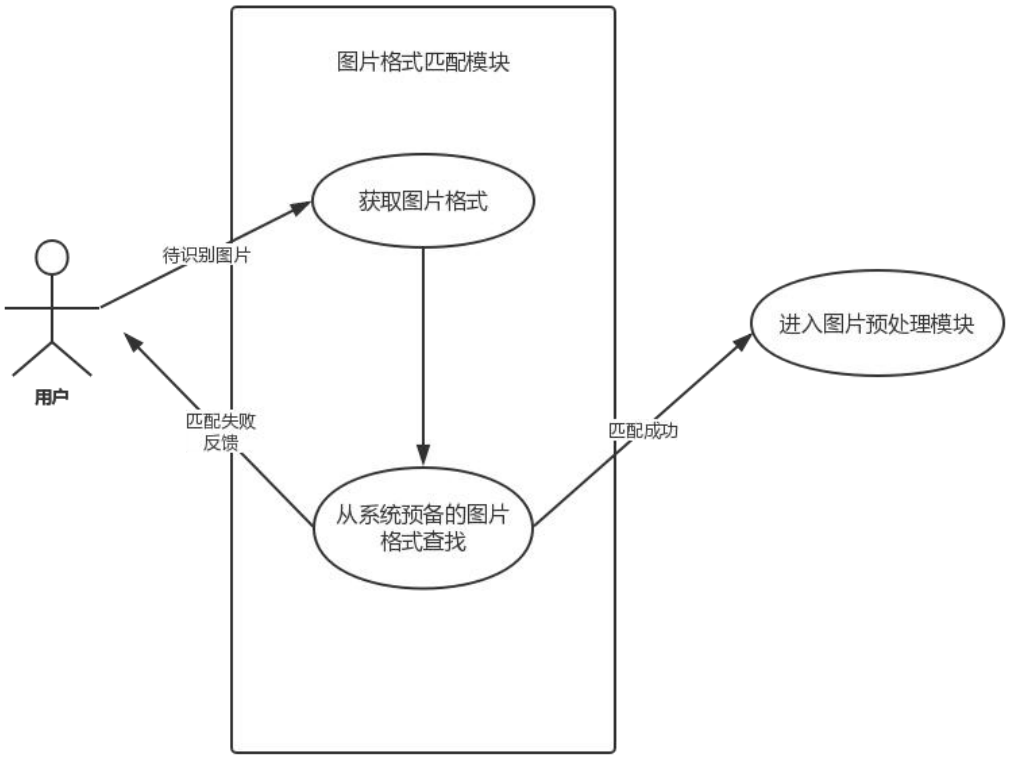


图 2-2 图片格式匹配用例图

天猫平台公示的图片内容没有固定格式，需要程序能匹配不同格式的图片内容提取信息。

2.2.4 提取企业注册号、企业名称数据项

提取出图片中的企业注册号、企业名称数据项。

2.2.5 识别准确率不低于 95%

本系统需要保证图片文字识别的正确率超过 95%。

2.2.6 识别结果导入 Excel 交付

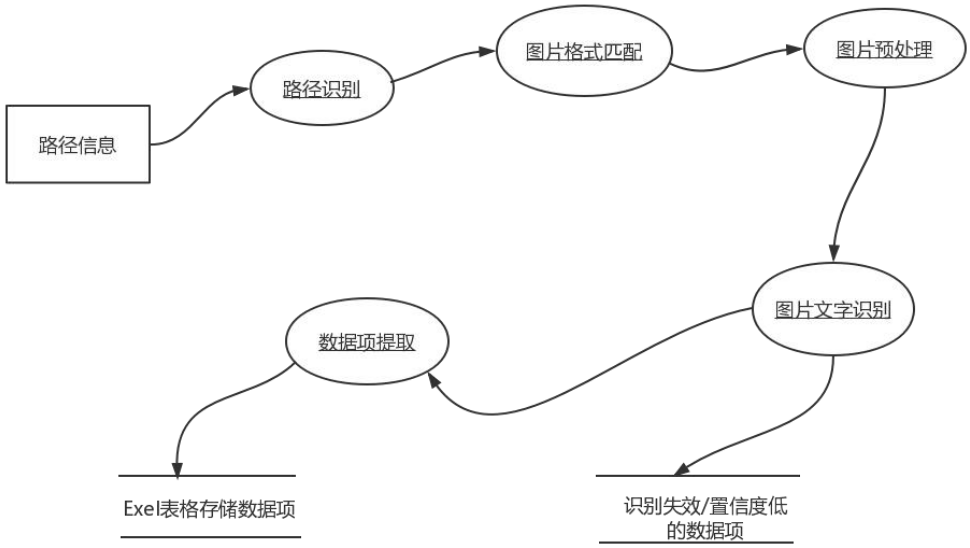


图 2-3 识别系统数据流图

最终的识别结果按规定格式以一份汇总的 Excel 交付。整个系统的输入信息为文件的路径信息，经过路径识别与文件提取，图片格式匹配，图片预处理与文字识别，根据是别的置信度分别处理，置信度高，可信任的数据之际交付到 Exel 表格中，置信度低或由于图片格式错误，图片内容不清晰等院系识别失败的图 p 片数据暂时存档，并根据用户选的的处理策略进行放弃处理或人工校验识别。

2.3 非功能性需求

2.3.1 识别速度

为保证程序运行效率，提升使用体验，图片识别速度保持在 60 秒识别 50 张图片。

2.3.2 程序容错性

考虑到待识别的图片格式不固定，图片板式与可识别性有区分，程序设置容错机制，对很难识别的图片自动放弃，保证程序运行流畅性。

2.4 运行环境规定

2.4.1 设备

市场上常规可见 pc 机

2.4.2 运行软件

开发环境操作系统：Windows

开发语言：C++、JAVA、Python

3 系统设计

3.1 总体架构

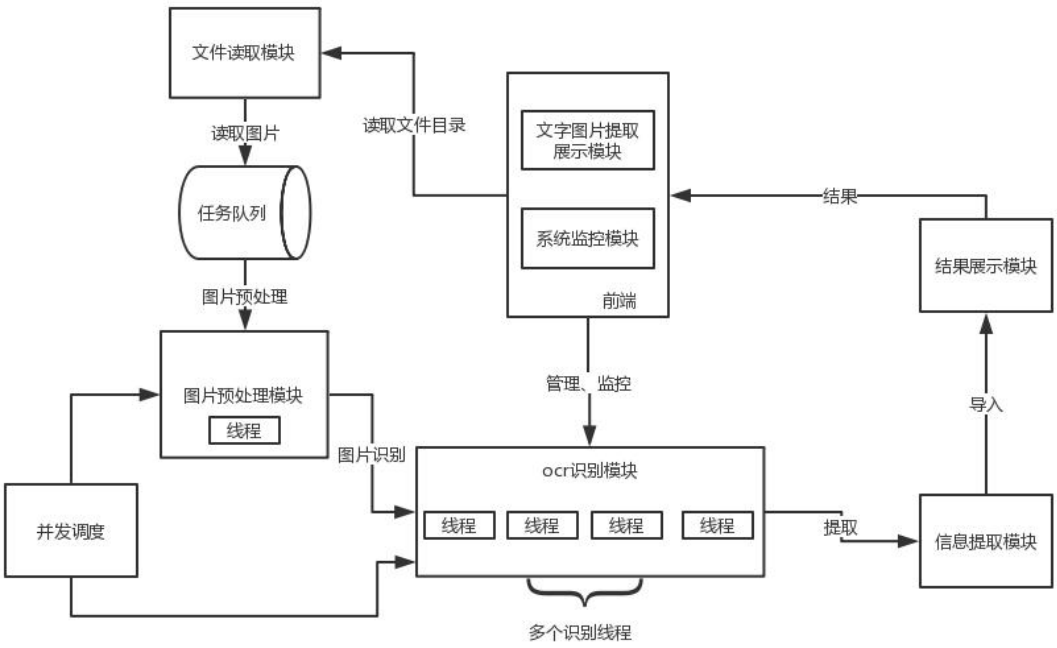


图 3-1 系统架构图

本系统是基于 LSTM 深度学习的 Tesseract 提供的 OCR 识别算法实现的网店工商信息图片文字提取。主要分为文件读取模块、图片预处理模块、识别模块、信息提取模块、结果展示模块、图片文字提取展示模块、系统监控模块等部件。其中，OCR 识别模块和信息提取模块是整个系统的核心，主要负责对目标图片中文字信息的识别以及需求信息的提取；文件读取模块负责根据指定的文件路径，将该文件目录中的图片导入到识别队列中；图片预处理模块负责完成对识别区域进行切割，缩小识别范围，并将图片处理为易于识别的图片；结果展示模块负责将识别出的结果以 Excel 的形式展示给用户；此外，本系统还提供了文字图片提取展示模块以及系统监控模块作为可视化图形界面接口，方便用户设置文件目录，了解识别进度以及查看识别结果。

3.2 模块划分

3.2.1 文件读取模块

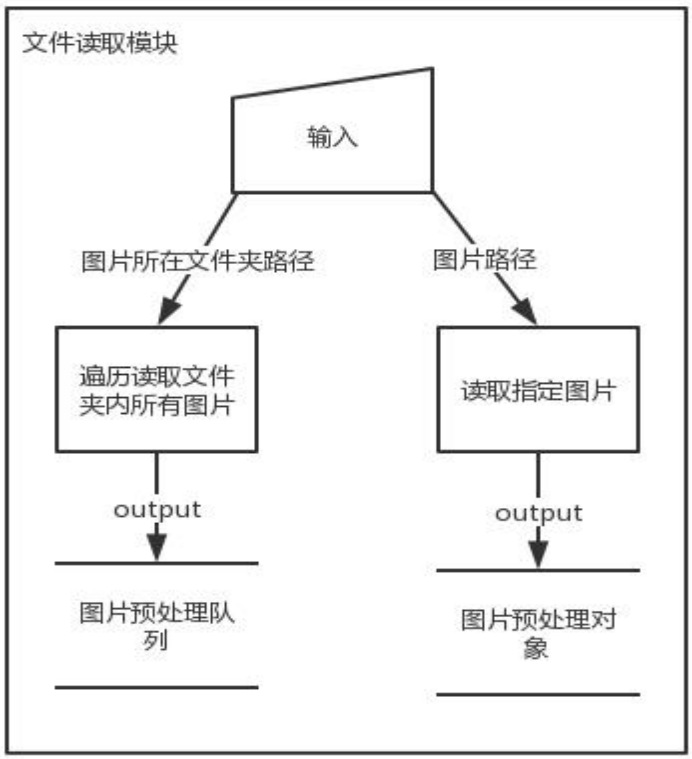


图 3-2 文件读取模块

文件读取模块根据用户所提供的文件目录，将该文件目录下所要识别的图片放入识别队列。用户所提供的信息可以为单张图片路径，也可为文件目录，当提供为文件目录时，系统将遍历识别整个目录下的图片。输入为图片目录，输出为待处理的图片。

3.2.2 图片预处理模块

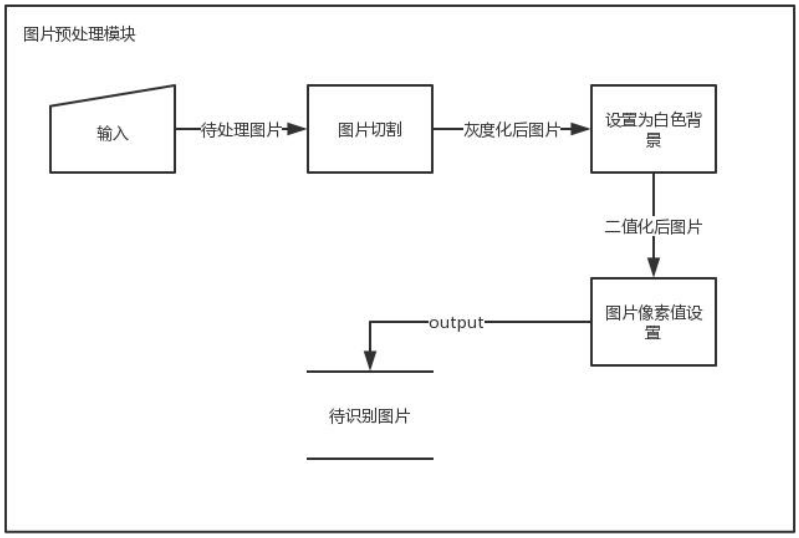


图 3-3 图片预处理模块

图片预处理模块包括图片切割，设置白色背景，图片设置像素三大部分。

图片切割部分使得图片仅留下文字区域的前两行（提取信息部分），后期不许再做多余的识别。

设置白色背景部分使得原有图片的透明背景变为白色，有利于 Tesseract 进行识别；

设置像素部分弥补图片预处理中像素的丢失，使得图片清晰度更高，易于识别；

输入为待处理的图片或图片队列，输出为处理过待识别的图片或图片队列。

3.2.3 图片识别模块

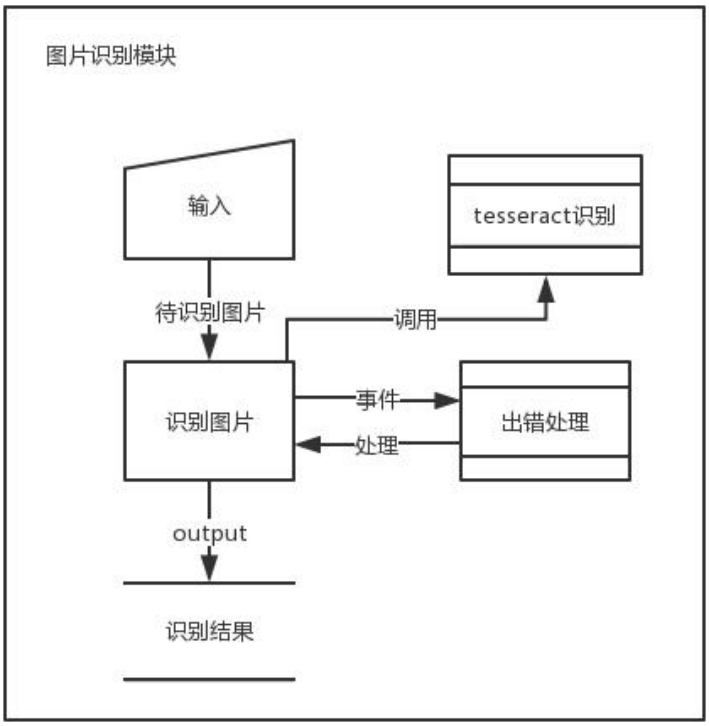


图 3-4 图片识别模块

图片识别模块包括识别图片，Tesseract 识别，出错处理三个部分。

识别图片部分负责调用 Tesseract 对处理后的图片进行识别，并将识别后生成的文本信息保存至后台，以做后期的提取；

Tesseract 识别部分负责识别图片信息的工作；

出错处理部分针对无法识别的图片做特殊标注，并加入到出错图片队列，以增强系统识别效率与系统的稳定性。

输入为待识别的图片，输出为识别的结果。

3.2.4 信息提取模块

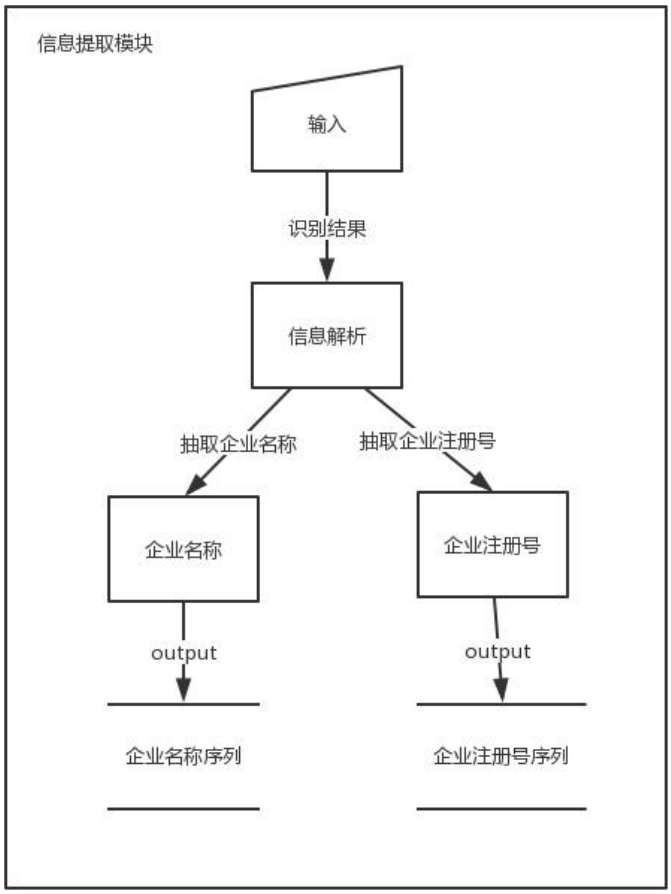


图 3-5 信息处理模块

信息提取模块负责将识别的结果进行解析，分为企业名称，企业注册号分别存储。

输入为初期识别结果，输出为一一对应的企业名称与企业注册号。

3.2.5 结果展示模块

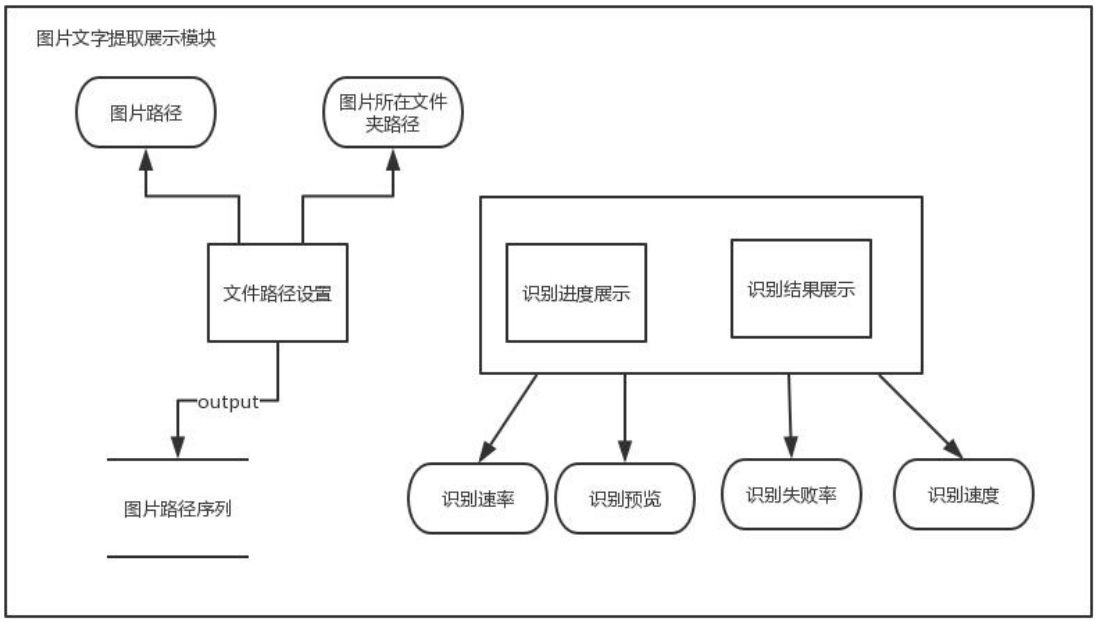


图 3-6 图片文字提取展示模块

图片文字提取展示模块负责向用户提供文字图片提取展示模块以及系统监控模块作为可视化图形界面接口，包括识别文件目录设置，图片识别进度，图片识别结果展示三大部分。

文件目录设置部分能设置识别图片所在的目录；

图片识别进度部分负责在识别过程中展示识别的进度，识别的速率，识别的准确率，以图表数据的形式呈现；

图片结果展示部分负责展示图片识别的结果，识别的失败率，识别速度，以图表数据的形式呈现。

3.2.6 存档管理模块

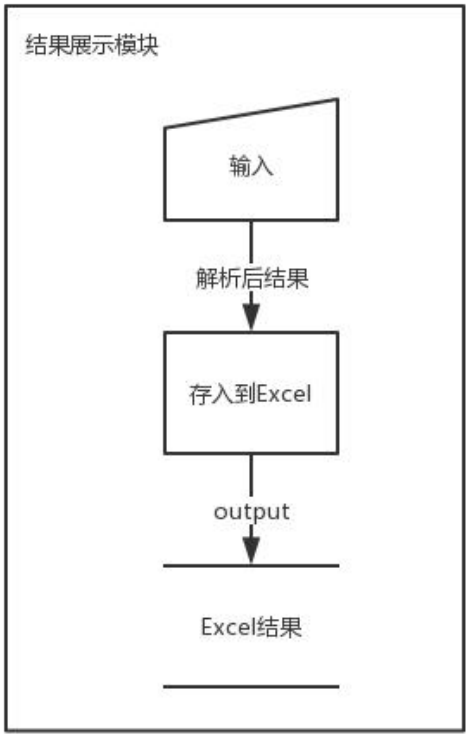


图 3-7 结果展示模块

结果展示模块负责将提取后的企业名称与企业注册号导入到用户指定目录中的 Excel 表格中，表格企业名称与企业名称一一对应。

3.3 关键技术描述

3.3.1 基于 Java 类库的图片处理技术

（一）图片有效区域定位与切割

根据题目要求，最终提取结果只需企业名称和企业注册号两项，而这两项在大部分图片中都处于前两行的位置，而后面的信息不在需求范围内，因此本系统采用 Java 自带的图片切割技术对图片切割，只保留需求的部分，去除不相关的部分，这种做法，可以大大减少识别的字数以及提高图片在处理中预处理的速度。

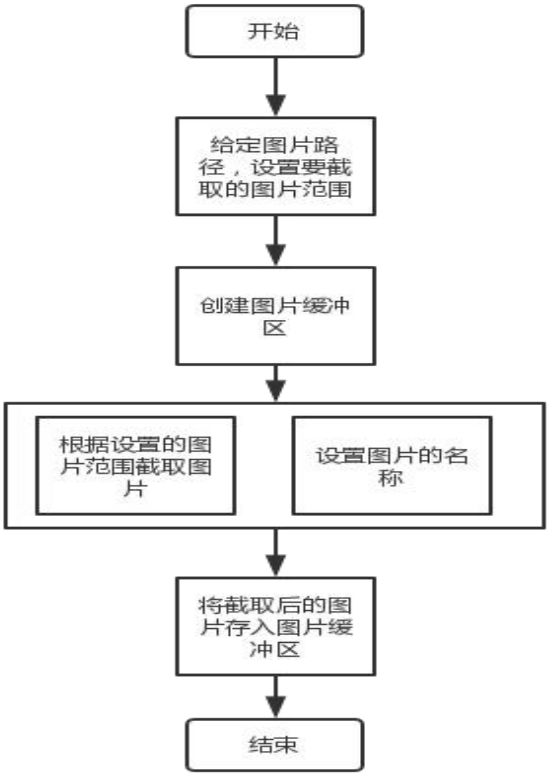


图 3-8 Java 类库实现图片切割

首先，给定待处理图片的路径，设置要截取的范围（企业名称所在行与企业注册号所在行）；

其次，创建图片缓冲区以存储结果；

然后，根据设置的图片范围调用 getSubimage 方法截取图片，并将源图片名

称赋给结果图片；

最后，将截取后的图片存入图片缓冲区，等待处理。

基于 Java 类库截取图片不依赖于其他第三方插件，只需引入 Java 自带的图形编辑的包便可实现预期效果，有利于提高图片处理的效率。

（二）图片预处理

为了构建适于 Tesseract 识别的图片环境，需要将图片设置为白色背景，并提高图片的像素，以提高图片的清晰度，使得图片更易于识别。

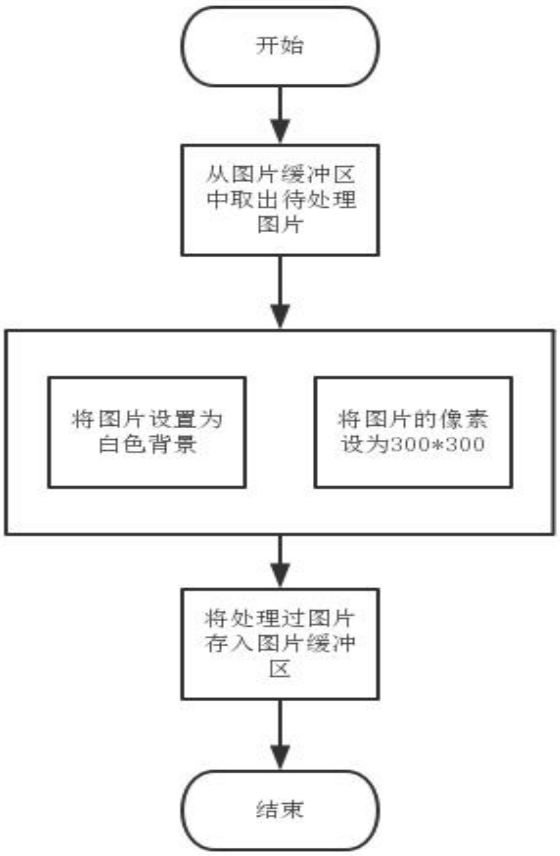


图 3-9 Java 类库实现图片处理

首先，从图片缓冲区中去除切割好的图片；

其次，将图片设置为白色背景即消除透明度，并且将图片的像素设置为 300*300，提高图片的清晰度；

最后，将处理好的图片存入图片缓冲区，等待识别。

经过处理后的图片更易于被 Tesseract 识别，大大地提高了 Tesseract 识别的

准确率。

3.3.2 基于 TensorFlow 框架的 CNN 神经网络设计

（一）数据集获取

首先使用 selenium 模拟浏览器，获取天猫登录的 cookie 值，由于天猫爬取工商图片需要识别验证码，因此本团队利用从天猫网站上爬取的 5000 张验证码图片进行人工标注后作为训练数据集。采用 FensorFlow 框架搭建 CNN 神经网络进行训练，算法思路为：把验证码 4 位字符的识别问题作为一个多标签学习的问题，4 个数字组成的验证码就相当于有 4 个标签的图片识别问题，且标签有序。经过约 8000 步迭代训练后，最终获得的模型针对天猫网站的验证码识别率可达到 92%。然后爬取 5000 张工商营业执照图片。

（二）卷积神经网络 CNN 识别验证码

关于验证码识别问题，传统的机器学习方法，对于多位字符验证码都是采用的 化整为零 的方法：先分割成最小单位，再分别识别，然后再统一。 卷积神经网络方法，则直接采用端到端不分割 的方法：输入整张图片，输出整个图片的标记结果，具有更强的通用性。端到端 的识别方法显然更具备优势，而且考虑到天猫网站的字符型验证码为了防止被识别，多位字符已经完全融合粘贴在一起了且有变形干扰，利用传统的技术基本很难实现分割。

Cnn 神经网络原理图如下：

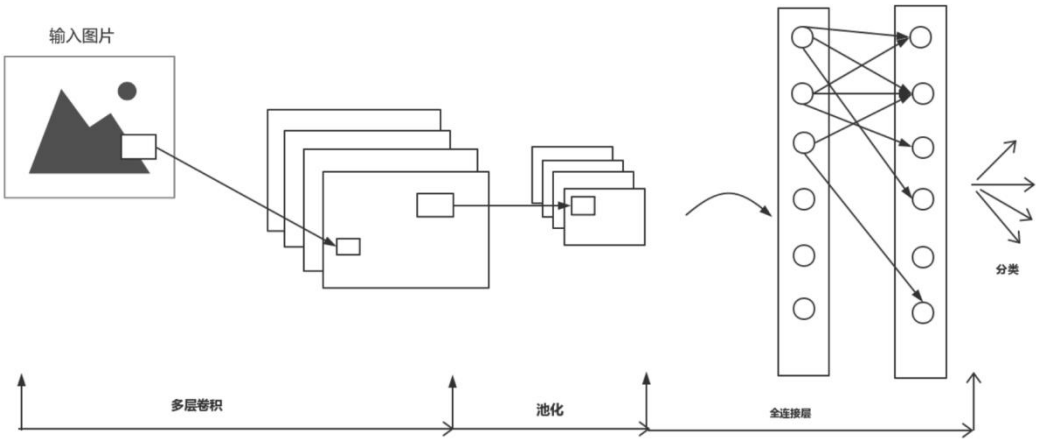


图 3-10 卷积神经网络原理图

卷积神经网络用于图像识别具有巨大优势：首先，由于该神经网络共享卷积核，对高维数据处理无压力。而且无需手动选取特征，训练好权重，特征分类效果好。cnn 神经网络主要分为以下几层：

数据输入层 => 卷积计算层 => 池化层 => 全连接层

分类函数选择 sigmoid loss，考虑到 CNN 的输出的维度是 固定的 4 维，其实这些维度并不都是完全独立分布的，相当于先用 sigmoid 进行了一次归一化，然后再将各个维度的值向目标值进行 回归，最后 loss 越小，两个向量的对应的值也越接近。

在模型训练过程中，本网络可看作是训练一个可进行多维分类的网络的常用方法是使用多项式逻辑回归,又被叫做 softmax 回归。Softmax 回归在网络的输出层上附加了一个 softmax nonlinearity，并且计算归一化的预测值和 label 的 1-hot encoding 的交叉熵。在正则化过程中，我们会对所有学习变量应用权重衰减损失。模型的目标函数是求交叉熵损失和所有权重衰减项的和，loss()函数的返回值就是这个值

根据天猫网站上爬取的验证码图片像素大小为 100*30，设置两层卷积，每次卷积后进行池化，最后全连接层输出 1024 个神经元。激活函数采用 softmax，迭代训练约 8000 轮后，准确率达到 92%。

3.3.3 基于 ReLU 优化算法的 Tesseract 训练引擎优化

（一）基于 LSTM 深度学习的 OCR 识别技术

OCR 文字识别是现在普遍使用的一种将图片识别转换成可编辑的 WORD 文档的技术，OCR 技术是光学字符识别的缩写，是通过扫描等光学输入方式将各种票据、报刊、书籍、文稿及其他印刷的文字转化为图像信息，再利用文字识别技术将图像信息转化为可以使用的计算机输入技术。它是人工智能技术之一，它让计算机和人一样，可以看图识字。它是一种快捷、省力、高效的文字输入方法。

因此在本系统中采用了 OCR 技术实现文字的识别。同时为了提高识别的准确度，针对系统识别汉字的要求，系统采取了基于 LSTM 的开源 OCR 训练引擎 Tesseract 并对其的 ReLU 函数进行优化以提高对中文字的识别率与速度。

本系统改良的 Long Short Term 网络是一种循环神经网络的特殊的类型，可以学习长期依赖信息。循环神经网络 RNN 是包含循环的网络，允许信息的持久化。RNN 可以被看做是同一神经网络的多次复制，每个神经网络模块会把消息传递给下一个，其关键点之一是可用来连接先前的信息到当前的任务上，当相关的信息和预测的词位置之间的间隔比较小时，RNN 可以很好发挥的发挥作用。但遇到更加复杂的场景，但当这个间隔不断增大时，RNN 会丧失学习到连接如此远的信息的能力。

针对这个问题，LSTM 通过刻意的设计来避免长期依赖问题。其创新点在于记住长期的信息在实践中是 LSTM 的默认行为，而非需要付出很大代价才能获得的能力。所有 RNN 都具有一种重复神经网络模块的链式的形式。在标准的 RNN 中，这个重复的模块只有一个非常简单的结构，例如一个 tanh 层。LSTM 同样是这样的结构，但是重复的模块拥有一个不同的结构。

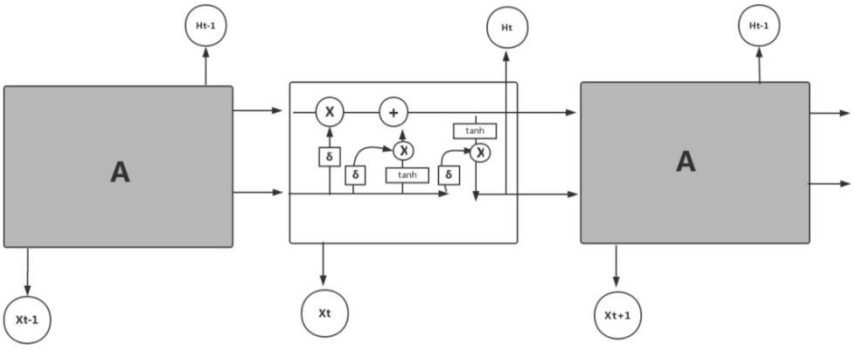


图 3-11　LSTM 神经网络信息传递图

LSTM 的关键就是细胞状态，水平线在图上方贯穿运行。细胞状态类似于传送带。直接在整个链上运行，只有一些少量的线性交互。信息在上面流传保持不变会很容易。

LSTM 有通过精心设计的称作为“门”的结构来去除或者增加信息到细胞状态的能力。门是一种让信息选择式通过的方法。他们包含一个 sigmoid 神经网络层和一个 pointwise 乘法操作。

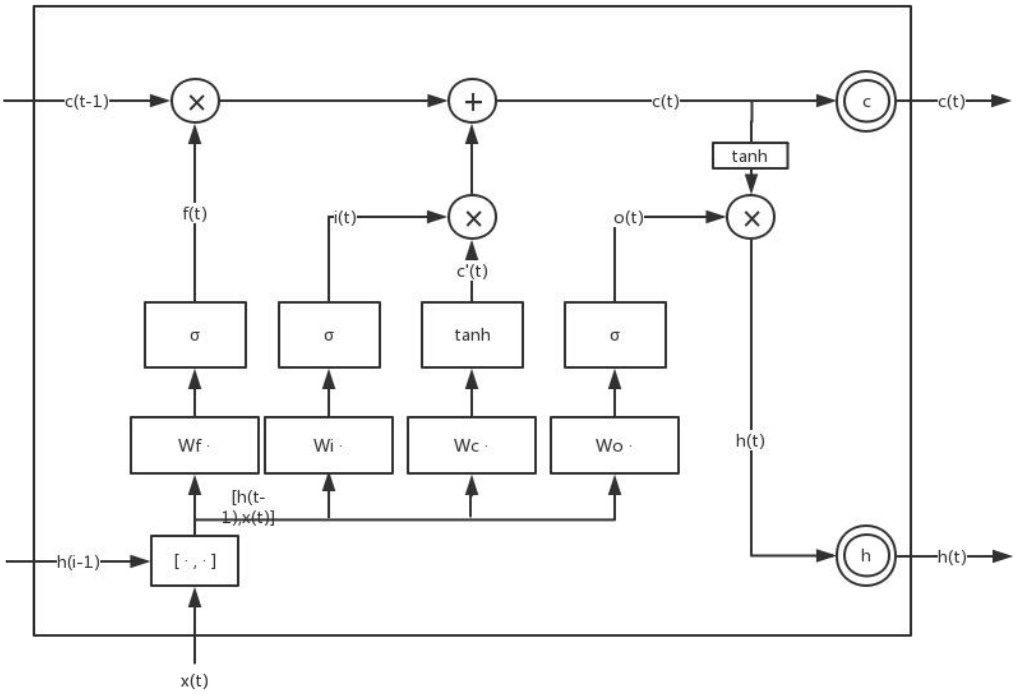


图 3-12 LSTM 原理图

长短时记忆网络(Long Short Term Memory Network, LSTM)，是一种改进之后的循环神经网络，可以解决 RNN 无法处理长距离的依赖的问题，在处理图片文字识别上也有很大的优势，可以有效地加快训练速度，提高识别准确率。

LSTM 用两个门来控制单元状态 c 的内容，一个是遗忘门（forget gate），它决定了上一时刻的单元状态 c(t-1)有多少保留到当前时刻；另一个是输入门（input gate），它决定了当前时刻网络的输入 x(t)有多少保存到单元状态 c(t)。LSTM 用输出门（output gate）来控制单元状态 c(t)有多少输出到 LSTM 的当前输出值 h(t)。

（二）Tesseract 图片识别训练引擎优化

（1）基于深度学习的训练引擎 Tesseract

Tesseract 是开源的 OCR 引擎。Tesseract 最初设计用于英文识别，经过改进引擎和训练系统，它能够处理其它语言和 UTF-8 字符。Tesseract 需要知道相同字符的不同形状，也就是不同字体。最多允许的字体数量在 intproto.h 中通过 MAX_NUM_CONFIGS 定义，目前支持 64 种。为了训练一种新语言当识别库，需要在 tessdata 子文件夹中创建一些数据文件，然后用 combine_tessdata 将它们合并为一个文件。

（2）ReLU 优化算法

本系统采用基于深度学习的图片识别技术，明确的目标是从数据变量中解离出关键因子。图片原始数据中通常缠绕着高度密集的特征，这些特征向量是相互关联的，一个小小的关键因子可能牵扰着一堆特征，有点像蝴蝶效应，牵一发而动全身。基于数学原理的传统机器学习手段在解离这些关联特征方面具有致命弱点。然而，如果能够解开特征间缠绕的复杂关系，转换为稀疏特征，那么特征就有了鲁棒性（去掉了无关的噪声）。稠密缠绕分布着的特征是信息最富集的特征，从潜在性角度，往往比局部少数点携带的特征成倍的有效。

而稀疏特征，正是从稠密缠绕区解离出来的，潜在价值巨大。不同的输入可能包含着大小不同关键特征，使用大小可变的数据结构去做容器，则更加灵活。

假如神经元激活具有稀疏性，那么不同激活路径上：不同数量（选择性不激活）、不同功能（分布式激活），两种可优化的结构生成的激活路径，可以更好地从有效的数据的维度上，学习到相对稀疏的特征，起到自动化解离效果。

本系统选择使用 ReLu 激活函数，使得网络可以自行引入稀疏性。这一做法，等效于无监督学习的预训练。缩小了非监督学习和监督学习之间的代沟，可以更快的学习特征。

ReLU 函数能够有效缓解梯度消失问题，其以监督的方式训练深度神经网络，无需依赖无监督的逐层预训练，显著提升了深度神经网络的性能。激活函数是

GRU 等深度神经网络结构的核心所在，目前常见的激活函数包括 sigmoid 系的 sigmoid 和 tanh 函数 ，ReLU 系的 ReLUl4。ReLU 函数 能够有效缓解梯度消失问题 ，其以监督的方式训练深度神经网络，无需依赖无监督的逐层预训练，显著提升了深度神经网络的性能。

通过查阅论文，找到对 ReLU 函数进行了改进的实现，将 ReLU 函数 x < 0 的部分使用 tanh 函数代替，构造出了一个新的激活函数 TLU，函数定义如下：

$$f(x)=\begin{cases}x(x>=0)\\ \alpha\tanh(x)(x>0)\end{cases}$$

函数图像如下：

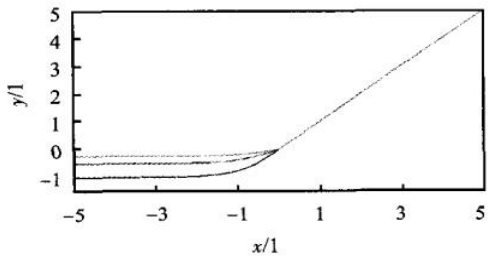


图 3-13　TLU 函数图像

TLU 在右侧的线性部分具有函数 ReLU 和 LReLU 的优点 ，在 I>0 时导数为常数，因此在饱和区内的梯度永远不会为 0，能够有效缓解梯度消失问题。TLU 能显著地加快深度神经 网络的训练速度并有效地降低训练误差。

（3）Tesseract 训练引擎优化

Tesseract 是基于分割的 OCR 的经典系统，在这种模式下，对于印刷体和手写字符串分割有很多方法。除了传统的模板匹配法，最近流行的深度学习也非常适合用于这图片文字识别这一领域。本组成员查阅文献，提出对 tesseract 采用的 LSTM 神经网络进行优化设计。

系统优化思想如下：

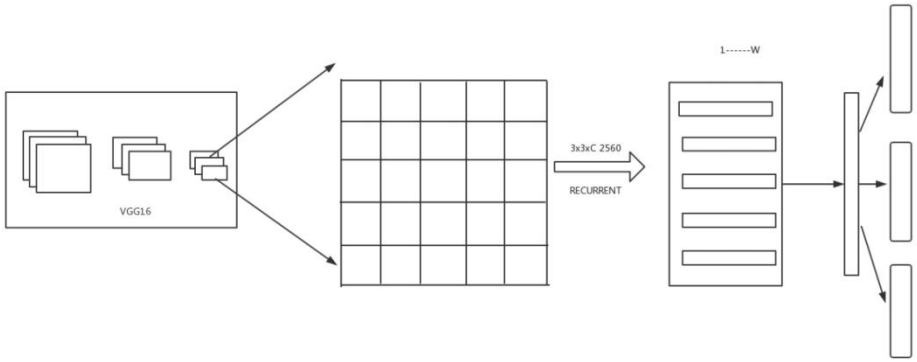


图 3-14 算法优化原理

第一，用 VGG16 的前 5 个 Conv stage（到 conv5）得到 feature map(W*H*C)

第二，在 Conv5 的 feature map 的每个位置上取 3*3*C 的窗口的特征，这些特征将用于预测该位置 k 个 anchor（anchor 的定义和 Faster RCNN 类似）对应的类别信息，位置信息。

第三，将每一行的所有窗口对应的 3*3*C 的特征（W*3*3*C）输入到 RNN（BLSTM）中，得到 W*256 的输出

第四，将 RNN 的 W*256 输入到 512 维的 fc 层

第五，fc 层特征输入到三个分类或者回归层中。第二个 2k scores 表示的是 k 个 anchor 的类别信息（是字符或不是字符）。第一个 2k vertical coordinate 和第三个 k side-refinement 是用来回归 k 个 anchor 的位置信息。2k vertical coordinate 表示的是 bounding box 的高度和中心的 y 轴坐标（可以决定上下边界），k 个 side-refinement 表示的 bounding box 的水平平移量。这边注意，只用了 3 个参数表示回归的 bounding box，这里默认每个 anchor 的 width 是 16，且不再变化（VGG16 的 conv5 的 stride 是 16）。回归出来的 box 如 Fig.1 中那些红色的细长矩形，它们的宽度是一定的。

3.3.4 基于 Java 多线程机制的多周期运行模式

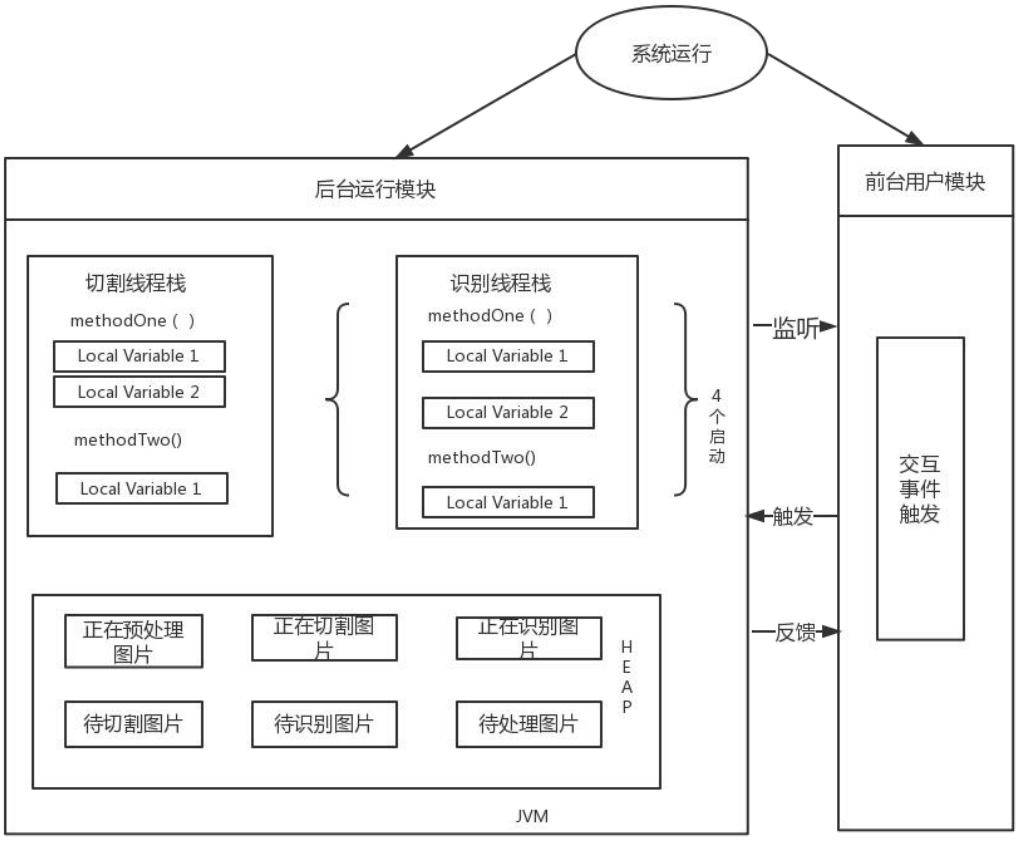


图 3-15 基于多线程机制的多周期运行模式

本系统采用前后端分周期独立运行模式，其中后台 Java 程序设置监听模块负责应对前台界面的按钮响应，保证程序体验的流畅性。而程序后端同时开启 5 个线程：图片切割与预处理模块单独启动一个线程，图片识别模块模块启动 4 个线程，在对共享资源即经过预处理的图片，由于线程的先后没有固定顺序，由操作系统实时分配，当两个线程竞争同一资源时，如果对资源的访问顺序敏感，就称存在竞态条件，导致竞态条件发生的代码区称作临界区。

本系统使用多线程技术，拥有多个线程并行执行。一个线程的执行可以被认为是一个 CPU 在执行该程序。当一个程序运行在多线程下，就好像有多个 CPU 在同时执行该程序。

多线程比多任务更加有挑战。多线程是在同一个程序内部并行执行，因此会对相同的内存空间进行并发读写操作。这可能是在单线程程序中从来不会遇到的问题。其中的一些错误也未必会在单 CPU 机器上出现，因为两个线程从来不会得到真正的并行执行。然而，更现代的计算机伴随着多核 CPU 的出现，也就意

味着不同的线程能被不同的 CPU 核得到真正意义的并行执行。

为保证线程安全，引入 Java 同步块（synchronized block）来标记方法或者代码块是同步的。Java 中的同步块用 synchronized 标记。同步块在 Java 中是同步在某个对象上。所有同步在一个对象上的同步块在同时只能被一个线程进入并执行操作。所有其他等待进入该同步块的线程将被阻塞，直到执行该同步块中的线程退出。

4 详细设计

4.1 文件读取模块

4.1.1 功能说明

文件读取模块根据用户所提供的文件目录，将该文件目录下所要识别的图片放入识别队列。用户所提供的信息可以为单张图片路径，也可为文件目录，当提供为文件目录时，系统将遍历识别整个目录下的图片。当提供的图片为单张图片时，直接识别该图片，当提供文件所在目录时，则遍历整个文件夹，得到要识别的图片序列。

4.1.2 处理流程

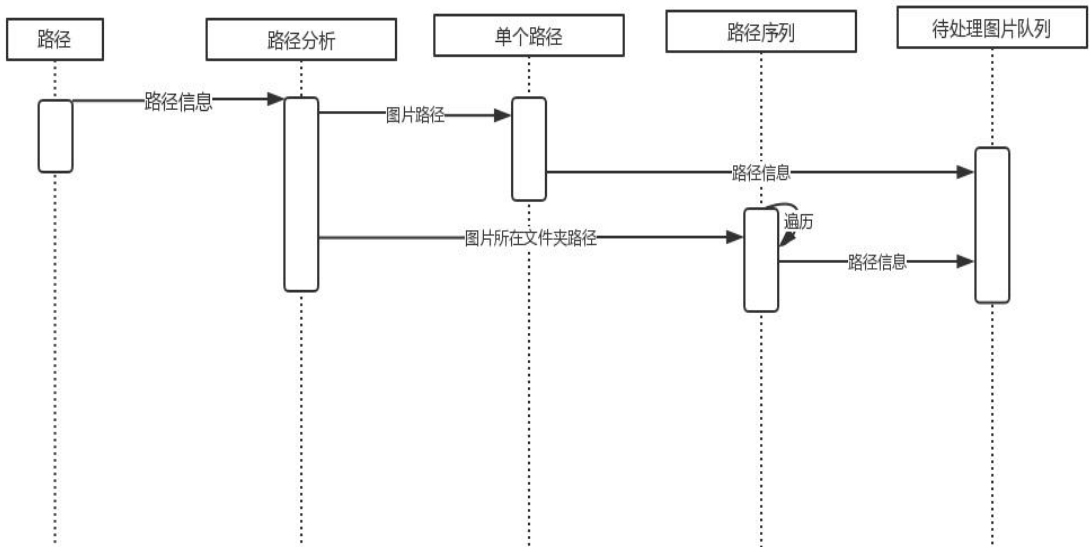


图 4-1 文件读取模块序列图

用户提供路径信息，后台解析路径，若是单个图片路径，直接上传到待处理图片队列等待处理；若是文件夹路径，则遍历该文件夹，将在遍历中得到的路径信息送至待处理图片队列。

4.1.3 关键实现技术描述

系统对用户导入的路径信息进行检测，当路径为单个图片文件夹时，系统自动读取文件夹内的图片并顺序识别，若用户导入的路径信息为多个嵌套的文件夹，则采用遍历算法，逐步便利读取文件路径内的图片。

4.2 图片预处理模块

4.2.1 功能说明

图片预处理模块包括图片切割，设置白色背景，图片设置像素三大部分。

图片切割部分使得图片仅留下文字区域的前两行（提取信息部分），后期不许再做多余的识别。

设置白色背景部分使得原有图片的透明背景变为白色，有利于 tesseract 进行识别；

设置像素部分弥补图片预处理中像素的丢失，使得图片清晰度更高，易于识别；

4.2.2 处理流程

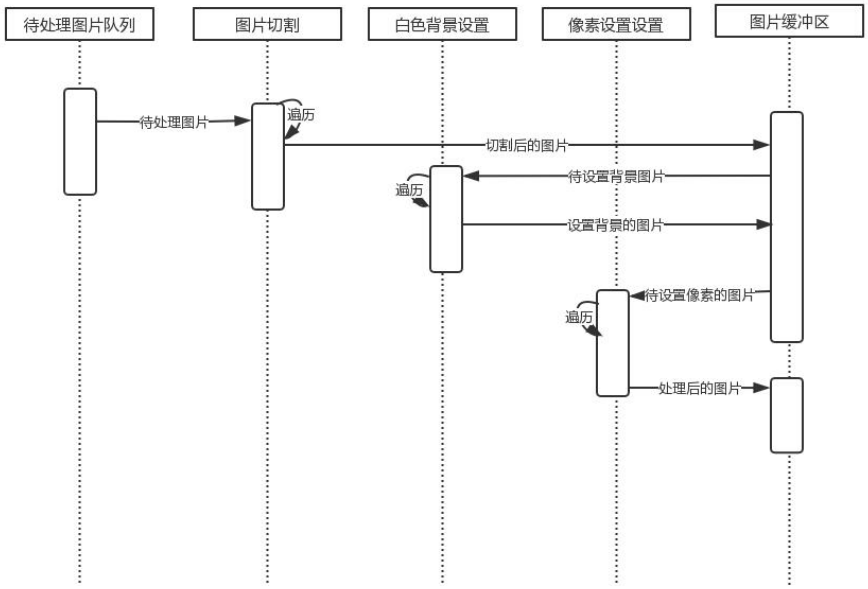


图 4-2 图片预处理模块序列图

图片预处理模块首先从待处理图片队列中取出图片信息，遍历进行图片切割，将切割好的图片送入图片缓冲区，然后从图片缓冲区取出待设置背景的图片遍历设置白色背景，处理后再次送入图片缓冲区等待设置像素，最后，从图片缓冲区中取出图片设置像素，图片预处理完毕，送入图片缓冲区等待识别。

4.2.3 关键实现技术描述

（1）图片切割：采用了 java 自带 awt 中的 BufferedImage 类与 imageio 中的 ImageIO 类，其中 BufferedImage 用于设置处理图片时所需的图片缓冲区，并且对图片进行范围划分与切割，ImageIO 负责图片的读写；

（2）白色背景设置：采用了 java 自带 awt 中 Color 类设置图片为白色背景，消除透明；

（3）像素设置：采用了 com.sun.image.codec.jpeg 中的 JPEGImageEncoder 类、JPEGCodec 类、JPEGEncodeParam 类，其中 JPEGImageEncoder 类负责创建 jpegEncoder 实例，JPEGCodec 类负责创建 JPEG 图片解析器，JPEGEncodeParam 负责将图片转化成 JPEG 图片，并设置像素。

4.3 图片识别模块

4.3.1 功能说明

图片识别模块包括识别图片，tesseract 识别，出错处理三个部分。

识别图片部分负责调用 tesseract 对处理后的图片进行识别，并将识别后生成的文本信息保存至后台，以做后期的提取；

tesseract 识别部分负责识别图片信息的工作；

出错处理部分针对无法识别的图片做特殊标注，并加入到出错图片队列，以增强系统识别效率与系统的稳定性。

4.3.2 处理流程

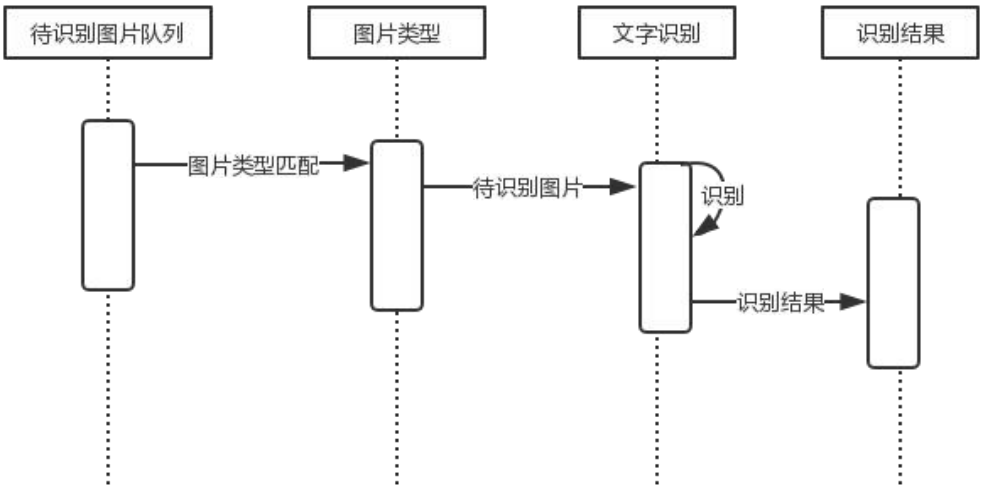


图 4-3 图片识别模块

从待处理图片队列中取出图片首先进行类型匹配，找到合适的识别入口，然后调用 tesseract 识别程序对图片进行识别，抽取识别结果至结果区。

4.3.3 关键实现技术描述

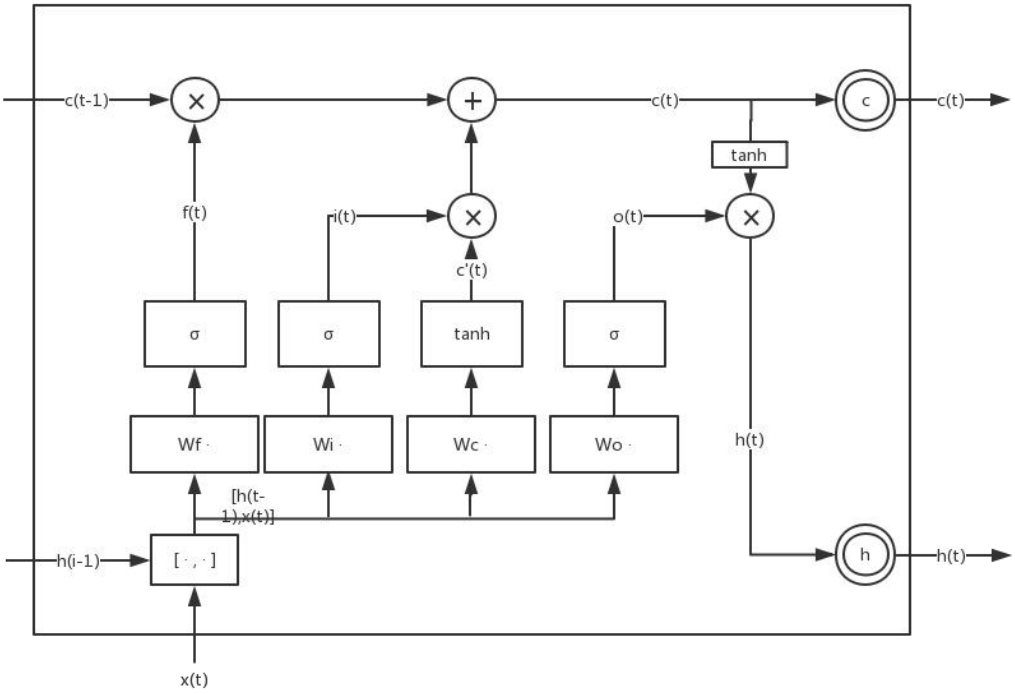


图 4-4 LSTM 神经网络原理

长短时记忆网络(Long Short Term Memory Network, LSTM)，是一种改进之后的循环神经网络，可以解决 RNN 无法处理长距离的依赖的问题，在处理图片文字识别上也有很大的优势，可以有效地加快训练速度，提高识别准确率。

LSTM 用两个门来控制单元状态 c 的内容，一个是遗忘门（forget gate），它决定了上一时刻的单元状态 c(t-1)有多少保留到当前时刻；另一个是输入门（input gate),它决定了当前时刻网络的输入 x(t)有多少保存到单元状态 c(t)。LSTM 用输出门（output gate）来控制单元状态 c(t)有多少输出到 LSTM 的当前输出值 h(t)。

其中，遗忘门的计算为：

$f(t)=\sigma(Wf \cdot [h(t-1),x(t)]+b(f))$

Wf 是遗忘门的权重矩阵，[h(t-1),x(t)]表示把两个向量连接成一个更长的向量，b(f)是遗忘门的偏置项，σ 是 sigmod 函数（激活函数）；

输入门的计算为:

$i(t)=\sigma(Wi \cdot [h(t-1),x(t)]+b(i))$

Wi 是输入门的权重矩阵，b(i)是输入门的偏置项；

用于描述当前输入的单元状态 c’(t)，它是根据上一次的输出和本次的输入来计算：

$$c'(t)=\tanh(Wc \cdot [h(t-1),x(t)]+b(c))$$

tanh()为激活函数，b(c)为此次的偏置项；

计算当前时刻的单元状态 c(t)，它是由上一次的单元状态 c(t-1)按元素乘以遗忘门 f(t)，再用当前输入的单元状态 c’(t)按元素乘以输入门 i(t)，再将两个积加和产生：

$$c(t)=f(t)\bigcirc c(t-1)+i(t)\bigcirc c'(t)$$

符号○表示按元素乘；

把 LSTM 关于当前的记忆 c’(t)和长期的记忆 c(t-1)组合在一起，形成了新的单元状态 c(t)。由于遗忘门 f(t)的控制，它可以保存很久很久之前的信息，由于输入门 o(t)的控制，它又可以避免当前无关紧要的内容进入记忆。

输出门的计算：

$$o(t)=\sigma(Wo \cdot [h(t-1),x(t)]+b(o))$$

b(o)为输出门的偏置项；

最终结果：

$$h(t)=o(t)\bigcirc\tanh(c(t))$$

由此实现 LSTM 前向计算。

(2) ReLU 优化算法

对于线性函数而言，ReLU 的表达能力更强，尤其体现在深度网络中；而对于非线性函数而言，ReLU 由于非负区间的梯度为常数，因此不存在梯度消失问题，使得模型的收敛速度维持在一个稳定状态。

ReLU 函数能够有效缓解梯度消失问题，其以监督的方式训练深度神经网络，无需依赖无监督的逐层预训练，显著提升了深度神经网络的性能。激活函数是 GRU 等深度神经网络结构的核心所在，目前常见的激活函数包括 sigmoid 系的 sigmoid 和 tan 函数,ReLU 系的 ReLU14。ReLU 函数能够有效缓解梯度消失问题 ，其以监督的方式训练深度神经网络，无需依赖无监督的逐层预训练，显著提升了深度神经网络的性能。

通过查阅论文，找到对 ReLU 函数进行了改进的实现，将 ReLU 函数 $x < 0$ 的

部分使用 tanh 函数代替，构造出了一个新的激活函数 TLU，函数定义如下：

$$f(x)=\begin{cases}x(x>=0)\\ \alpha\tanh(x)(x>0)\end{cases}$$

函数图像如下：

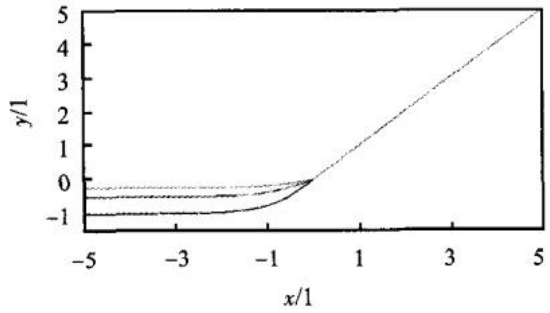


图 4-5　TLU 函数图像

TLU 在右侧的线性部分具有函数 ReLU 和 LReLU 的优点 ，在 I>0 时导数为常数，因此在饱和区内的梯度永远不会为 0，能够有效缓解梯度消失问题。TLU 能显著地加快深度神经网络的训练速度并有效地降低训练误差。

4.4 信息提取模块

4.4.1 功能说明

信息提取模块将从文字识别模块中得到的信息根据需求进行提取，得到有效信息，提交到结果展示模块。

4.4.2 处理流程

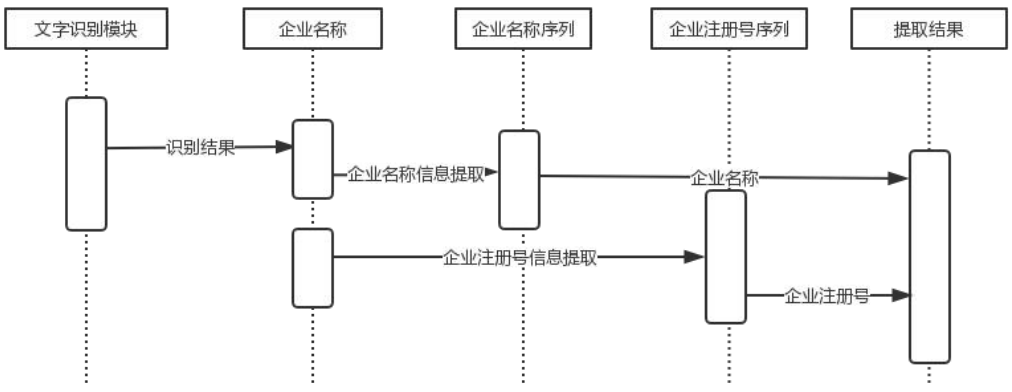


图 4-6 信息提取模块序列图

信息提取模块从文字识别模块得到识别结果根据企业名称汉字部分，提取企业名称信息字段至企业名称序列；根据企业注册号汉字部分，提取企业注册号信息至企业注册号序列，最后将两个序列并入至识别结果，等待展示。

4.4.3 关键实现技术描述

信息提取模块重点用到了 java 中字符串类型的 split 操作，利用提供的参数实现字符串的分割；本次需求为企业注册号和企业名称，利用 split 方法，实现企业注册号与企业名称的分割，并实现分别存储，以供后续的成果展示所需。

4.5 进度控制

4.5.1 功能说明

进度控制模块可以实现根据用户在主界面的开始、暂停、继续的操作，控制程序的执行进度。

4.5.2 处理流程

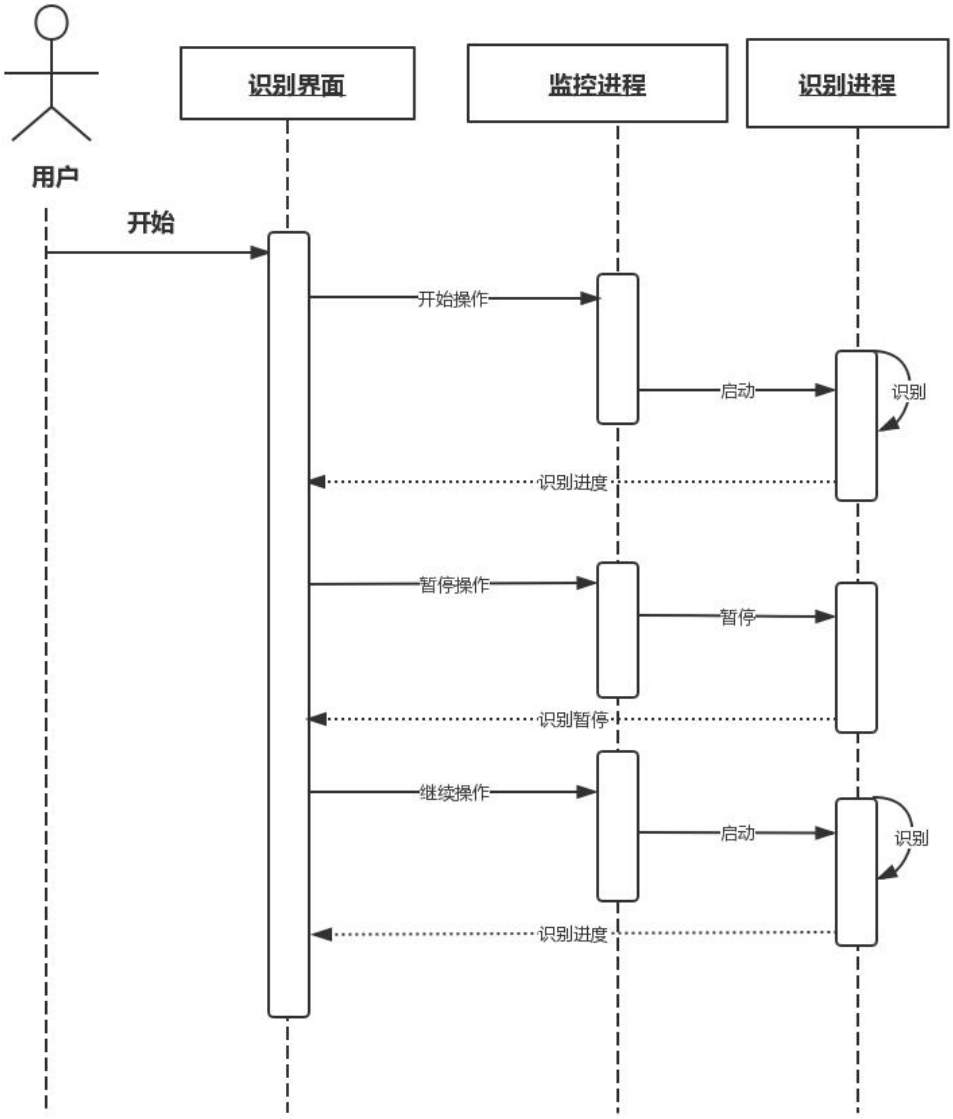


图 4-7 进度控制模块序列图

用户在识别界面选择开始识别，监控进程捕获到操作，启动识别进程，开始识别，识别进程将识别进度反馈到识别界面；若用户选择了暂停，监控进程暂停识别进程，识别进程停止识别，并反馈暂停信息；若用户选择了继续，监控程序将再次启动识别进程，识别进程将继续识别，并将识别进度反馈到用户界面。

4.5.3 关键技术描述

为了实现用户对图片识别进程的控制，在 java 中设置了一个控制变量，控制程序的执行。

为达到目的，在 java 主程序中设置一个进度监控进程，监控用户在主界面的操作，并根据用户当前的操作，调整控制变量：当用户选择开始识别图片的操作时，该监控进程将控制变量设为开启状态，通知图片切割进程与图片识别进程开始对目标图片进行识别；若在识别过程中，用户选择了暂停操作，监控程序将控制变量设置为关闭状态，此时图片切割进程与识别进程处于暂停状态；若用户之后选择了开始操作，监控程序在得到此操作指令后，将控制变量重新设置为开始状态，图片切割进程与识别进程将从上次暂停的地方继续执行。

该监控进程在识别任务创建时创建，在识别任务结束时结束。

4.6 结果展示模块

4.6.1 功能说明

结果展示模块分为图片识别结果展示，置信度标识，识别所用时间统计三大部分。

图片识别结果展示负责一次识别任务完成后提取出的企业注册号和企业名称两项数据。

置信度标识通过在对图片文字进行识别时归类的置信度来个性化标识识别完成后展示的数据项，其中置信度高于百分之 90 的系统认定为可信赖，可直接加以保存的数据；若置信度低于百分之 30，系统将其判定为识别失效并加以标识。在这之间的置信度，使用者可以根据自己对天猫图片识别的任务的需求自定义准确度要求，系统会根据用户定义的置信度标准在识别结果中加以标识。确保针对不同需求层次用户的使用便捷性与个性化定制。

识别结果统计模块包括一次识别任务完成的时间，识别总字数，识别图片张数，基于容错机制考虑忽略的图片数量等。方便用户在识别图片数目较大时，对识别任务整体完成情况有大体了解。

4.6.2 处理流程

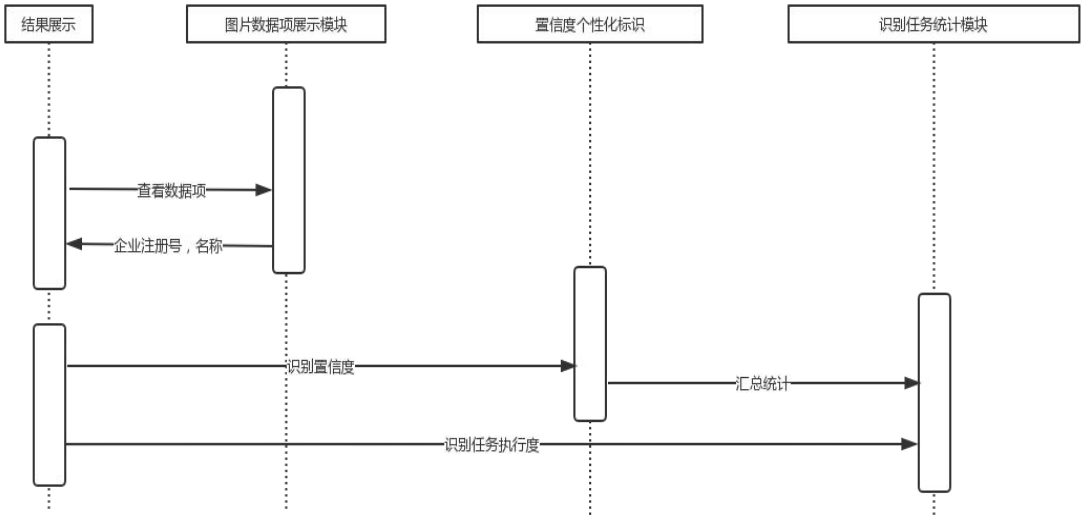


图 4-8 结果展示模块序列图

用户从结果展示首页查看本次识别任务提取出的数据项，即企业名称和企业注册号。

对识别任务的总体执行情况生成报告，使用户对本次识别数据的可信度有大致了解。

对识别过程中难以识别的图片和文字，基于容错机制加以忽略，在任务完成详情中展示；对每个成功识别的文字记录其归类置信度。并根据置信度高低对文字进行颜色标识。

用户根据自身需求选择人工操作方案，设置置信度可信与失败阈值。

对每次识别任务生成任务完成情况统计数据，和 Exel 文件一同归档存储，便于查看。

4.6.3 关键实现技术描述

（1）在识别文字的过程中，对神经网络最后的归类置信度进行保存。

（2）系统在完成识别任务的过程中，统计相关任务完成情况参数并统计，包括识别任务总图片数目，识别任务运行时间，识别失败的图片数，识别置信度低于可靠阈值的字数与解决方式等。通过这些字段的数据生成统计报表。从而量

化识别任务的具体完成情况，方便后续管理与二次利用。

4.7 存档管理模块

4.7.1 功能说明

本系统在用户每次的识别任务结束后对其进行存档管理，方便用户随时查看。

用户在程序主界面可以新建自己的识别任务，也可在个人页面点击查看由自己创建的历史识别任务的列表。

点击进入历史识别任务的详情页，可查看识别任务的具体完成情况，包括识别的图片数目，总处理时间，提取出的数据项详细信息，成功识别的图片数目，识别失败的文字数目及其所在图片等，识别任务存档为本地文件，用户可打开保存的 Exel 文件进行二次操作。

4.7.2 处理流程

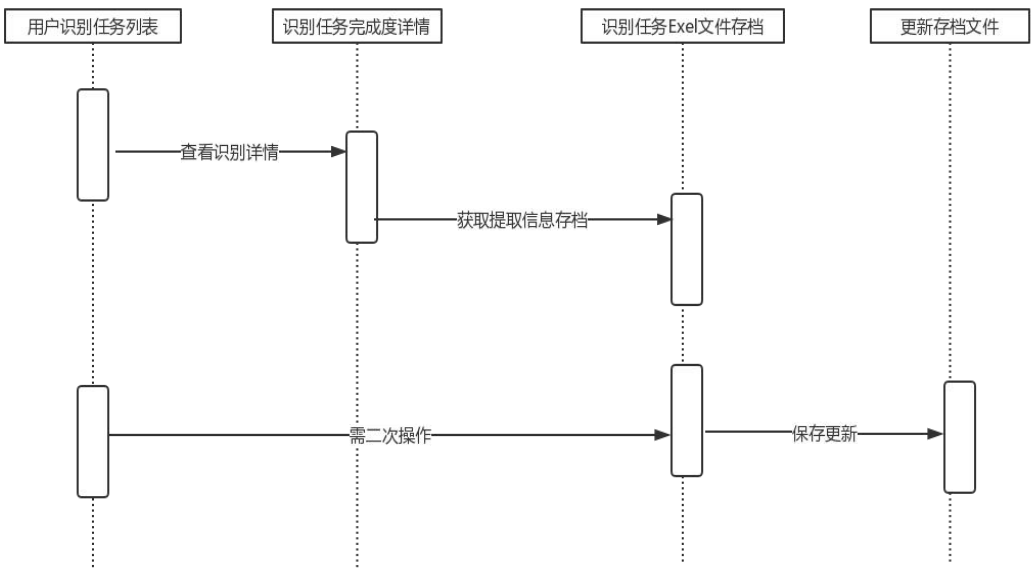


图 4-9 存档管理模块序列图

用户在识别任务列表中课点击查看识别任务完成度详情。

用户可对存档文件进行实时更新。

4.7.3 关键实现技术描述

（1）对每个用户创建的识别任务，包括识别的最终 Exel 文档，识别质量的统计报表等信息均可存储在本地文件中，方便用户后续查看与二次操作。

（2）对于保存的存档文件用户可根据需要进行动态更新，实时保存。

5 系统测试

5.1 测试环境

测试环境：

Win10 操作系统， 安装 Microsoft Excel 组件，java jdk 以及 tesseract 4.0

测试内容包括个单元单独测试部分和集成测试部分，并对系统的识别结果与 Excel 保存给出评估。

配置	性能参数
处理器 CPU	双核
主存	8GB
操作系统	Windows

5.2 主要功能测试

5.2.1 单元测试

5.2.1.1 文件读取单元

功能	输入	预期输出	实际输出	平均耗时
单个图片路径分析	单个图片路径	单个图片信息	单个图片信息	0.83s
文件夹内图片路径分析	图片所在文件夹地址（50 张）	整个文件夹下图片信息（50 张图片信息）	整个文件夹下图片信息（50 张图片信息）	2s

5.2.1.2 图片预处理单元

功能	输入	预期输出	实际输出	平均耗时
切割图片	原图	切割后的图片（50 张）	切割后的图片（50 张）	0.4s
设置图片为白色背景	切割后图（50 张）	白色背景的图片(50 张）	白色背景的图片（50 张）	0.4s
设置图片像素	设置为白色背景的图片（50 张）	设置了像素的图片（50 张）	设置了像素的图片（50 张）	0.6s

5.2.1.3 图片识别单元

功能	输入	预期输出	实际输出	平均耗时
识别图片（50 张）	预处理过的图片（50 张）	识别信息（50 条）	识别信息（企业注册号，序列号）（50 条）	30s

5.2.1.4 进度控制模块

功能	输入	预期输出	实际输出	平均耗时
识别任务暂停	对正在识别的任务点击暂停按钮(10 次）	识别任务暂停	识别任务暂停	0.05s
识别任务恢复	对已暂停的识别任务点击开启按钮（10 次）	识别任务继续	识别任务继续	0.04s

5.2.1.5 信息提取模块

功能	输入	预期输出	实际输出	平均耗时
提取企业注册号	文字识别完成的执照图片（100 张）	企业注册号 50 项	企业注册号数据 50 项	3.1s
提取企业名称	文字识别完成的执照图片（100 张）	企业名称 50 项	企业名称 50 项	2.7s

5.2.1.6 结果展示模块

功能	预期输出	实际输出	平均耗时

详情页动态展示识别完成后 Excel 交付的内容，点击即可编辑	在详情页展示可动态实时编辑的表格信息	在详情页展示可动态实时编辑的表格信息	1.9s
用户自定义置信度阈值，系统自动识别结果分为置信度高与低两部分暂存	详情页分为两个动态表格展示，有不同颜色标识	详情页分为两个动态表格展示，有不同颜色标识	0.5s
对识别失败的图片显示失败原因	100 张均正确显示识别失败原因	93 张判断出识别失败原因	1.4s
识别置信度低的图片与识别失败图片可点击查看原图	100 张均可点击查看原图	100 张均可点击查看原图	1s
对人眼难以识别的图片有舍弃机制	可点击舍弃	可点击舍弃	0.8s
用户校正完成后系统将结果合并为一张 Excel 保存	动态合并	动态合并	2s
可在详情页点击查看保存的 Excel 文件	100 份 Excel 文件保存	100 份 Excel 文件保存	7.1s

5.2.1.7 存档管理单元

功能	输入	预期输出	实际输出	平均耗时
将提取的信息归档到 Excel 中（50 条）	识别出的原始信息（50 条）	带有有效信息的 Excel 表格（50 条）	带有有效信息的 Excel 表格(50 条）	3s

5.2.2 集成测试

5.2.2.1 图片文件读取运行、定位截取与预处理周期

功能	输入	预期输出	实际输出
读取文件信息	文件路径	文件信息	一致
文件截取	原图信息	截取后的图片	一致
文件预处理	截取后的图片	预处理后文件	一致

5.2.2.2 图片识别与信息项提取，Excel 归档周期

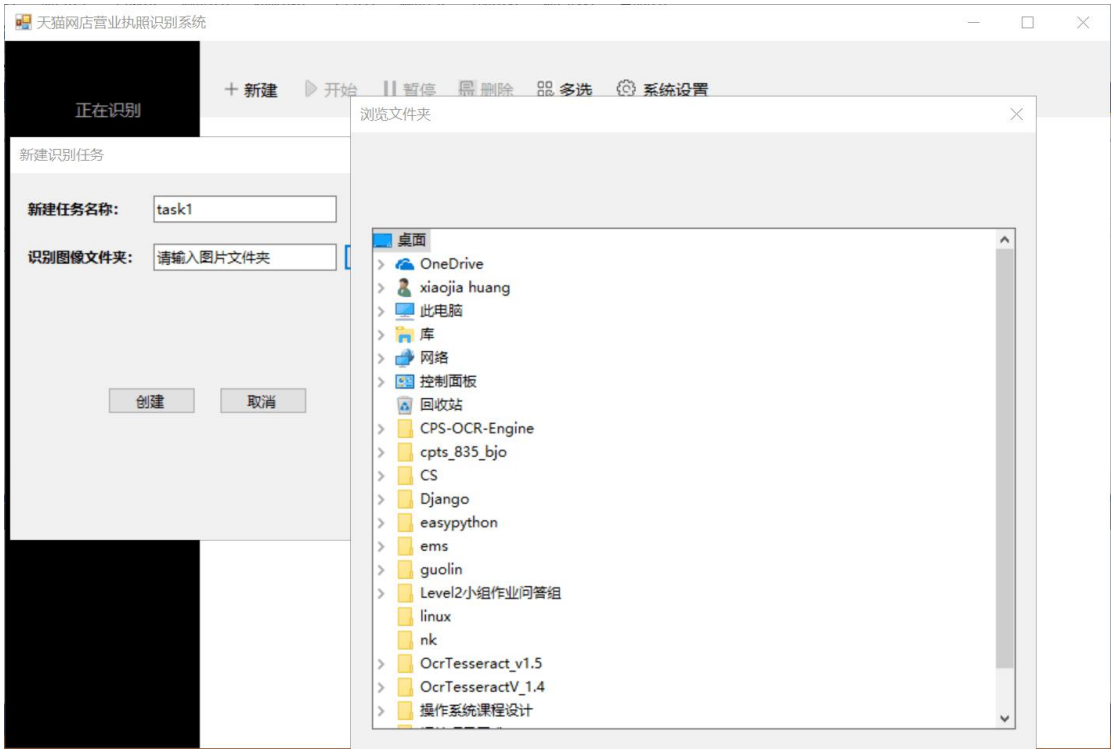
功能	输入	预期输出	实际输出
图片文字识别	1000 张经过预处理的天猫网站营业执照图片测试集	测试集的文字识别正确率超过 95%	一致
信息项提取	100 份经过正确识别的文字信息	提取出 100 份企业名称与企业注册号信息项	一致
Excel 归档	测试 10 次，每次 100 份提取完全的信息项数据	10 次成功归档在 Excel 中，保存在本地文件	一致

5.2.2.3 天猫网店营业执照识别系统整体运行测试

功能	输入	预期输出	实际输出
读取文件信息	文件路径	文件信息	一致
文件截取	原图信息	截取后的图片	一致
文件预处理	截取后的图片	预处理后文件	一致
图片文字识别	1000 张经过预处理的天猫网站营业执照图片测试集	测试集的文字识别正确率超过 95%	一致
信息项提取	100 份经过正确识别的文字信息	提取出 100 份企业名称与企业注册号信息项	一致
Excel 归档	测试 10 次，每次 100 份提取完全的信息项数据	10 次成功归档在 Excel 中，保存在本地文件	一致

5.2.3 确认测试

（1）新建任务，读取文件目录

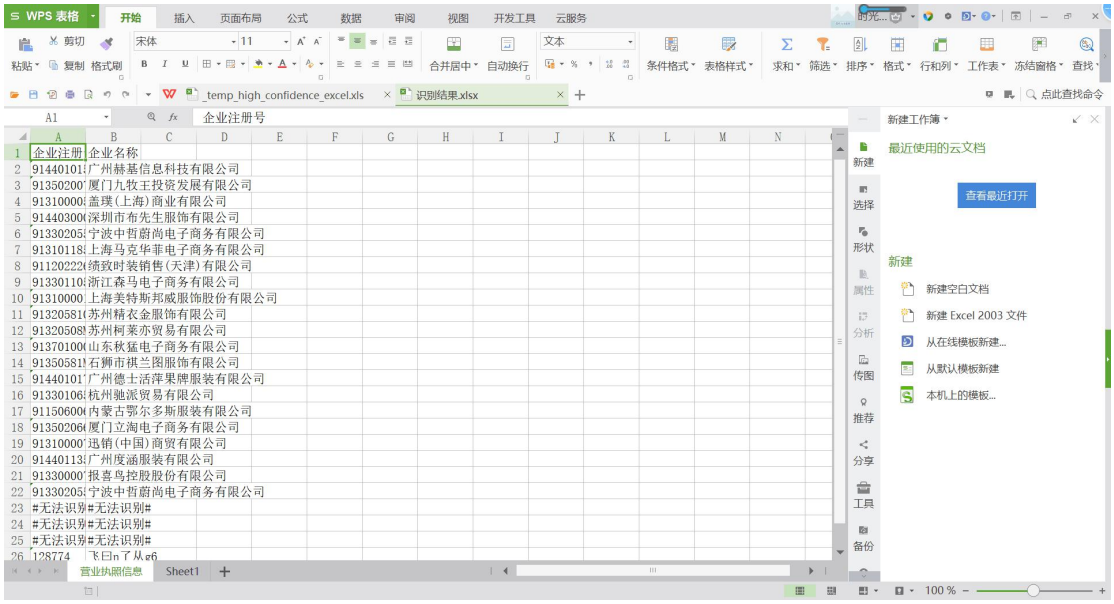


（2）识别过程

（3）识别结果预览（可信赖项）

（4）识别结果（不可信赖项）

（5）Excel 展示结果



6 总结

本系统完成了预定的全部功能性需求和非功能性需求，开发了一个完善的天猫网店工商营业执照图片识别系统。具体完成情况如下表：

表 6-1 完成情况表

需求	完成情况
图片路径自动读取，顺序识别图片	路径分析与自动文件读取算法
多种格式图片的匹配	系统支持 jpg，jpeg.png，tif，bmp 等多种格式
提取企业注册号与名称	正则匹配，字符串提取
图片识别准确率不低于 95%	ReLU 优化的 LSTM 神经网络训练
识别结果导入 Excel 保存	Office Excel 插件应用
识别速度约 50 张/min	多线程机制
程序容错机制	识别失败在运行过程中忽略，识别结束后反馈给用户，保证流畅体验

在系统的设计与实现中，我们充分尊重用户不同层次的需求，为用户提供定

制化图片文字识别服务。系统的创新在于通过采用 Relu 优化的思想改进 Tessract 训练引擎的 LSTM 神经网络，从而达到 95%准确率功能需求。简约的应用界面与操作步骤，用户辅助模式的系统流程设计，协助用户使用本系统获得符合自身准确度要求的天猫网店工商营业执照的企业名称与注册号的数据项。多线程机制的使用确保的系统的高效流畅。

系统的下一步研究工作，将在现有系统的基础上探寻更有效的，训练时间更短的神经网络模型，从而进一步提供系统的识别准确度。辅助用户花费更少人工校验的时间获得更高可信赖度的数据项。积极探索和调整系统的流程设计，提升用户体验，为广大用户提供更具个性化，更加智能化的天猫网店工商图片营业执照图片识别系统。