用Python 寫了一個圖像文字識別OCR 工具

Python數據科學 2021-12-27 13:35

以下文章來源於快學Python ， 作者蝦米小餛飩

引言

最近在技術交流群裡聊到一個關於圖像文字識別的需求，在工作、生活中常常會用到，比如**票據、漫畫、掃描件、照片的文本提取**。

博主基於PyQt + labelme + PaddleOCR 寫了一個桌面端的OCR工具，用於快速實現圖片中**文本區域自動檢測+文本自動識別**。

識別效果如下圖所示：

▲OCR工具識別效果

所有框選區域為OCR算法自動檢測，右側列表有每個框對應的文字內容；點擊右側“識別結果”中的文本記錄，然後點擊“複製到剪貼板”即可複制該文本內容。

功能列表

- 文本區域檢測+文字識別
- 文本區域可視化
- 文字內容列表
- 圖像、文件夾加載
- 圖像滾輪縮放查看
- 繪製區域、編輯區域
- 複製所選文本識別結果

OCR部分

圖像文字檢測+文字識別算法，主要藉助 `paddleocr` 實現。

创建或者选择一个虚拟环境，安装需要用到的第三方库。

```
conda create -n ocr
conda activate ocr
```

安装框架

如果你没有NVIDIA GPU，或GPU不支持CUDA，可以安装CPU版本：

```
# CPU版本
pip install paddlepaddle==2.1.0 -i https://mirror.baidu.com/pypi/simple
```

如果你的GPU安装过CUDA9或CUDA10，cuDNN 7.6+，可以选择下面这个GPU版本：

```
# GPU版本
python3 -m pip install paddlepaddle-gpu==2.1.0 -i https://mirror.baidu.com/pypi/simple
```

安装 PaddleOCR

安装paddleocr：

```
pip install "paddleocr>=2.0.1" # 推荐使用2.0.1+版本
```

版面分析，需要安装 Layout-Parser：

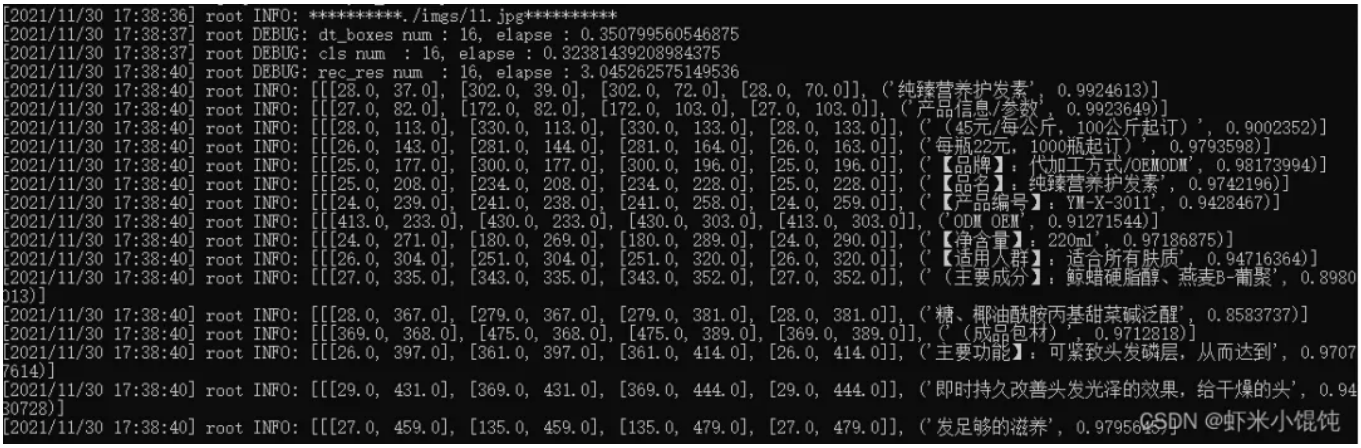
```
pip3 install -U https://paddleocr.bj.bcebos.com/whl/layoutparser-0.0.0-py3-none-any.whl
```

测试安装是否成功

安装完成后，测试一张图片 `--image_dir ./imgs/11.jpg` ，采用中英文检测+方向分类器+识别全流程：

```
paddleocr --image_dir ./imgs/11.jpg --use_angle_cls true --use_gpu false
```

输出一个list：



在python中调用

```
from paddleocr import PaddleOCR, draw_ocr

# Paddleocr目前支持的多语言语种可以通过修改lang参数进行切换
# 例如`ch`, `en`, `fr`, `german`, `korean`, `japan`
ocr = PaddleOCR(use_angle_cls=True, lang="ch")  # need to run only once to download and load m
img_path = './imgs/11.jpg'
result = ocr.ocr(img_path, cls=True)
for line in result:
    print(line)
```

输出结果是一个list，每个item包含了文本框，文字和识别置信度：

> [[[24.0, 36.0], [304.0, 34.0], [304.0, 72.0], [24.0, 74.0]], ['纯臻营养护发素', 0.964739]] [[[24.0, 80.0], [172.0, 80.0], [172.0, 104.0], [24.0, 104.0]], ['产品信息/参数', 0.98069626]] [[[24.0, 109.0], [333.0, 109.0], [333.0, 136.0], [24.0, 136.0]], ['（45元/每公斤，100公斤起订）', 0.9676722]] .....

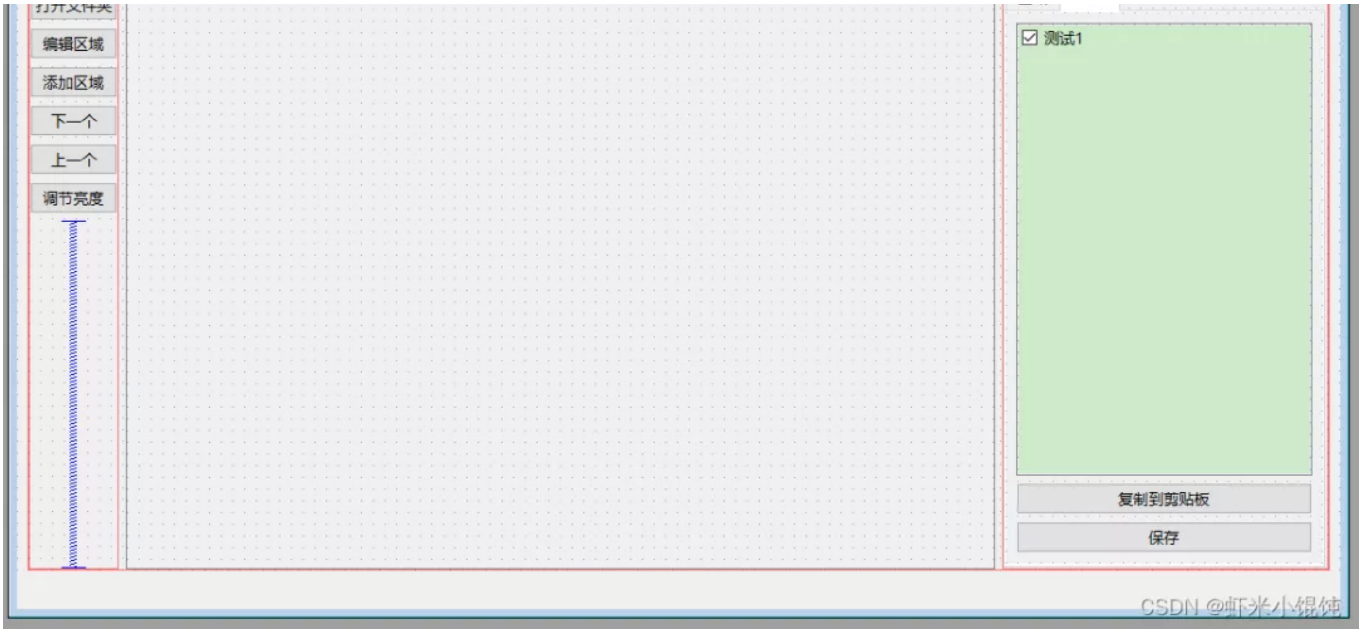
界面部分

界面部分基于pyqt5实现，其中pyqt GUI程序开发入门和环境配置，详见一篇博客(具体见文末)。

主要步骤：

界面布局设计

在QtDesigner中拖拽控件，完成程序界面布局，并保存`*.ui`文件。



利用 pyuic 自动生成界面代码

在 pycharm 的项目文件结构中找到 `*.ui` 文件，右键——External Tools——pyuic，会在ui文件同级目录下自动生成界面 ui 的 Python 代码。

```
from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(1139, 669)
        MainWindow.setStyleSheet("font: 10pt \"Microsoft YaHei UI\";")
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.verticalLayout_3 = QtWidgets.QVBoxLayout(self.centralwidget)
        self.verticalLayout_3.setObjectName("verticalLayout_3")
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")
        self.label_2 = QtWidgets.QLabel(self.centralwidget)
        self.label_2.setObjectName("label_2")
        self.horizontalLayout_3.addWidget(self.label_2)
        self.checkBox_ocr = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_ocr.setObjectName("checkBox_ocr")
        self.horizontalLayout_3.addWidget(self.checkBox_ocr)
        self.checkBox_det = QtWidgets.QCheckBox(self.centralwidget)
        self.checkBox_det.setObjectName("checkBox_det")
        self.horizontalLayout_3.addWidget(self.checkBox_det)
```

```
self.checkBox_recog = QtWidgets.QCheckBox(self.centralwidget)
```


编写界面业务类

业务类 MainWindow 实现程序逻辑和算法功能，与前面第2步生成的ui实现解耦，避免每次修改ui文件会影响业务代码。ui界面上的控件可以通过 `self._ui.xxxObjectName` 访问。

```
class MainWindow(QMainWindow):
 FIT_WINDOW, FIT_WIDTH, MANUAL_ZOOM = 0, 1, 2

 def __init__(self):
  super().__init__()  # 调用父类构造函数，创建QWidget窗体
  self._ui = Ui_MainWindow()  # 创建ui对象
  self._ui.setupUi(self)  # 构造ui
  self.setWindowTitle(__appname__)

  # 加载默认配置
  config = get_config()
  self._config = config

  # 单选按钮组
        self.checkBtnGroup = QButtonGroup(self)
        self.checkBtnGroup.addButton(self._ui.checkBox_ocr)
        self.checkBtnGroup.addButton(self._ui.checkBox_det)
        self.checkBtnGroup.addButton(self._ui.checkBox_recog)
        self.checkBtnGroup.addButton(self._ui.checkBox_layoutparser)
        self.checkBtnGroup.setExclusive(True)
```

实现界面业务逻辑

对主界面上的按钮、列表、绘图控件进行**信号槽连接**。自定义的槽函数不用专门声明，如果是自定义的信号，需要在类__init__()前加上 `yourSignal= pyqtSignal(args)` 。

这里以按钮响应函数、列表响应函数为例。按钮点击的信号是 `clicked` ，listWidget列表切换选择的信号是 `itemSelectionChanged` 。

```
# 按钮响应函数
self._ui.btnOpenImg.clicked.connect(self.openFile)
self._ui.btnOpenDir.clicked.connect(self.openDirDialog)
self._ui.btnNext.clicked.connect(self.openNextImg)
```

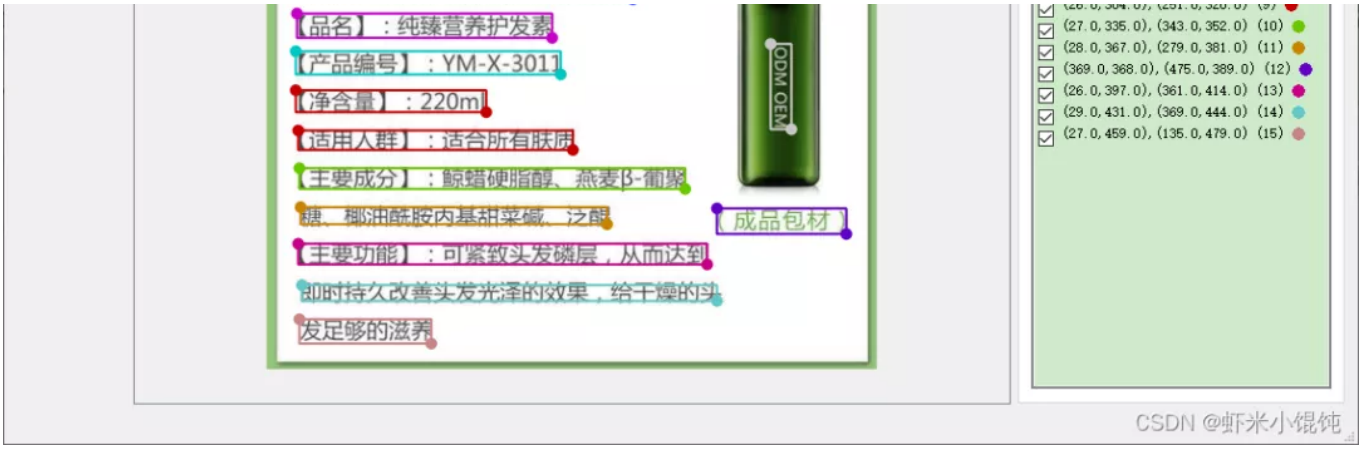
```
self._ui.btnPrev.clicked.connect(self.openPrevImg)
self._ui.btnStartProcess.clicked.connect(self.startProcess)
self._ui.btnCopyAll.clicked.connect(self.copyToClipboard)
self._ui.btnSaveAll.clicked.connect(self.saveToFile)
self._ui.listWidgetResults.itemSelectionChanged.connect(self.onItemResultClicked)
```

5. 运行看看效果

运行 `python main.py` 即可启动GUI程序。

打开图片→选择语言模型ch（中文）→选择文本检测+识别→点击开始，检测完的文本区域会自动画框，并在右侧识别结果——文本Tab页的列表中显示。

所有检测出文本的区域列表，在识别结果——区域Tab页：



软件代码

由于时间有限，软件细节功能还需进一步完善。代码已开源到 gitee 上，欢迎感兴趣的朋友提pull request，共同修改完善。

代码开源地址：*https://gitee.com/signal926/ocr-gui-demo*

参考链接

画框、区域列表：*https://github.com/wkentaro/labelme*

icons：*https://github.com/google/material-design-icons*

https://gitee.com/paddlepaddle/PaddleOCR/blob/release/2.3/doc/doc_ch/quickstart.md

https://blog.csdn.net/Bit_Coders/article/details/119304488

推荐阅读

1. pandas100个骚操作
2. 机器学习原创系列
3. 数据科学干货下载

最后给大家**分享《10本数据挖掘电子书》**，包括数据分析、统计学、数据挖掘、机器学习。

现在免费分享出来，有需要的读者可以下载学习，在下面的公众号「**数据挖掘工程师**」里回复关键字：**数据挖掘**，就行。

数据挖掘工程师

数万名数据挖掘爱好者的聚集地，致力于前沿数据技术研究。公众号以数据为核心，分...

17篇原创内容

公众号

喜欢此内容的人还喜欢

能进这个Java组织的都是大神，现在只有三个中国人

码农小胖哥

人体肤色检测：100 行 Python 实现

新机器视觉

拆解内存系统：在Python 虚擬機中引入複製內存管理算法| 極客時間

AI前線