有线表格合成算法

有线表格合成算法主要分为:

- 表格框架的生成
- 键值对数据的生成
- 表格图片合成器
- 后处理器

表格框架的生成原理

见 uniform.py
在采集的表格的基础上总结出了多个类型的子表格:
如:

- single_line 单行文字
- multiline_text 多行文字，如条款描述
- cross_rows_cross_cols 跨行跨列
- complex 复杂多级表头
- multiple_key_value_pairs 多个键值对
- multiple_rows_multiple_columns 正规的多行多列
- single_key_multiple_values 单键多选
- single_key_value 单键单值

将上述子类表格随机生成纵向连接（vstack），构成覆盖所有情况的表格框架

表格框架生成参数配置，为每一个类型创建一个yaml配置文件
base 中定义了资源文件与输出文件路径
filter 中定义了过滤器相关参数，实际使用了最小行数过滤器
formtype 中是针对这个表格类型使用的子表格随机参数和概率，调整参数以得到不同复杂度的表格框架

```yaml
base:
  label_dir: static/label/baoxian
  bg_dir: static/paper/baoxian
  output_dir: output//baoxian
  type: baoxian
  save_mid: false
```

```yaml
filters:
  disable: true
  keys_length: 6
  keys_length_fixed: true
  max_num_of_rows: 15
  max_table_width: 80
  min_num_of_rows: 10
  min_table_width: 10
  vstack_split_ratio: 1
  min_width_of_cell: 4
  empty_cell_ratio: 0.05

form_type:
  cross_rows_cross_cols:
    probability: 0.5
    max_num: 2
    max_num_of_rows: 2
    max_num_of_cols: 3
    max_num_of_blocks: 3
  complex:
    left_to_right_ratio: 0.5
    max_depth_of_header: 3
    max_num: 1
    max_num_of_cols: 2
    max_num_of_rows: 3
    min_depth_of_header: 2
    min_num_of_cols: 2
    min_num_of_rows: 2
    fixed_width_ratio: 1
    probability: 0.8
  multiline_text:
    has_title_raito: 0.5
    max_num: 1
    max_num_of_rows: 8
    min_num_of_rows: 3
    probability: 0.5
  multiple_key_value_pairs:
    max_num: 3
    max_num_of_pairs: 3
    min_num_of_pairs: 2
    probability: 0.8
  multiple_rows_multiple_columns:
    header_left_ratio: 0.5
    header_top_ratio: 0.5
```

```
    hrules_all_ratio: 1
    max_num: 2
    max_num_of_cols: 6
    max_num_of_rows: 5
    min_num_of_cols: 4
    min_num_of_rows: 3
    probability: 0.8
    vrules_all_ratio: 1
  single_key_multiple_values:
    max_num: 3
    max_options: 4
    min_num: 1
    min_options: 2
    probability: 0.8
  single_key_value:
    max_num: 3
    min_num: 1
    probability: 0.8
  single_line:
    max_num: 3
    min_num: 1
    probability: 0.8
```

键值对数据的生成

见 fakekeys.py
生成：在人工标注的数据上清洗，用模糊算法区分键和值和长文本，也可不做区分，构成键值生成器
使用：在生成上述表格框架的时候调用键值生成器，得到同用表格的文本表示。

无线表格银行流水合成算法

银行流水合成算法主要由四大块组成：

1. 银行数据生成器
2. 表格生成器
3. 表格图片合成器
4. 后处理器

银行数据生成器

1. 提供银行表单所有的字段，根据字段随机生成对应的值；
2. 使用 dataframe 保存流水明细
3. 分为必要字段和非必要字段，非必要字段可随机丢弃；
4. 为每个字段随机选择一个别名，同一字段不同银行选用的别名不同，如'账号': ['卡号', '主卡卡号', '账号/卡号', '借记卡卡号']
5. 搜集每家银行logo图片，用于标题或水印
6. 输出：

```
{'卡号': self.cardnumber,
'户名': self.name,
'银行': self.bank,
'起始日期': str(self.start),
'截止日期': self.df['工作日期'][nums - 1],
'操作地区': self.df['地区号'][0],
'操作网点': random.randint(1000, 1100),
'操作柜员': random.randint(10000, 20000),
'流水明细': self.df,
}
```

其中流水明细：

```
        self.df = pd.DataFrame({
            '工作日期': pd.date_range(self.start, periods=nums,
freq='M').strftime(
                '%Y-%m-%d'),
            '账号': self.cardnumber,
            '应用号': 1,
            '序号': range(1, nums + 1),
            '币种': 'RMB',
            '钞汇': '钞',
            '交易代码': np.random.randint(0, 3, nums),
            '注释': np.random.choice(['工资', 'ATM转账', 'ATM取款', '消费',
'现存'], nums),
            '借贷': '借',
            '发生额': [_('float_number', start=-1000, end=1000, precision=2)
for i
                     in range(nums)],
            '余额': 0,
            '存期': 0,
            '约转期': '不转存',
```

```
            '通知种类': 0,
            '利息': 0,
            '利息税': 0,
            '起息日': self.start,
            '止息日': '2099-12-31',
            '地区号': _('zip_code'),
            '网点号': np.random.randint(1000, 1100, nums),
            '操作员': np.random.randint(10000, 20000, nums),
            '界面': np.random.choice(['ATM交易', '网上银行', '批量业务', 'POS
交易'], nums)
        })
```

表格生成器

将数据生成器的输出字典生成纯文本表示的表格

1. 参数有一个最大列数，当明细条目超过最大列数，将多出的字段换行处理。
2. 若有换行，则在这一行后添加一行标记行 每个单元格内形如<r1c2> 标记所在的行列
3. 输出：文本

吴江农村商业银行账户历史明细清单

卡号:6530118533948481　户名:吴丽娟　起始日期:2021-04-02　截止日期:2023-01-31

操作地区:212321　操作网点:1023　操作柜员:13605

工作日期 <r0c0> 通知种类 <r0c11>	账号 <r0c1> 利息 <r0c12>	应用号 <r0c2> 利息税 <r0c13>	序号 <r0c3> 起息日 <r0c14>	交易代码 <r0c4> 地区号 <r0c15>	交易描述 <r0c5> 网点 <r0c16>	借贷 <r0c6> 授权柜员号 <r0c17>	发生额 <r0c7> 界面 <r0c18>	活存账户余额 <r0c8>	存期 <r0c9>	约转期 <r0c10>
2021-04-30 <r1c0> 0 <r1c11>	6530118533948481 <r1c1> 0 <r1c12>	1 <r1c2> 0 <r1c13>	1 <r1c3> 2021-04-02 <r1c14>	1 <r1c4> 212321 <r1c15>	现存 <r1c5> 1041 <r1c16>	借 <r1c6> 16193 <r1c17>	125.51 <r1c7> 批量业务 <r1c18>	29432.49 <r1c8>	0 <r1c9>	不转存 <r1c10>
2021-05-31 <r2c0> 0 <r2c11>	6530118533948481 <r2c1> 0 <r2c12>	1 <r2c2> 0 <r2c13>	2 <r2c3> 2021-04-02 <r2c14>	1 <r2c4> 212321 <r2c15>	ATM取款 <r2c5> 1079 <r2c16>	借 <r2c6> 11017 <r2c17>	832.45 <r2c7> POS交易 <r2c18>	28600.04 <r2c8>	0 <r2c9>	不转存 <r2c10>
2021-06-30 <r3c0> 0 <r3c11>	6530118533948481 <r3c1> 0 <r3c12>	1 <r3c2> 0 <r3c13>	3 <r3c3> 2021-04-02 <r3c14>	0 <r3c4> 212321 <r3c15>	消费 <r3c5> 1040 <r3c16>	借 <r3c6> 13688 <r3c17>	375.23 <r3c7> 网上银行 <r3c18>	28224.81 <r3c8>	0 <r3c9>	不转存 <r3c10>

文本表格合成图片

将表格生成器的输出的纯文本表格解析成图片

1. 逐行解析文本表格
2. 第一行为标题，用更大的字体居中写出，附带添加logo的效果
3. === 解析成需要化横线
4. 按 || 分成多个单元格的文本，写上文本
5. 函数签名如下，可控的位置，字体和字号，背景图片或背景色，如果是背景图片还要提供标注框 bg_box，行高或行间距，银行logo路径，是否背景水印，是否画虚线。

```
def banktable2image(table,
                    xy=None,
                    font_size=20,
                    bgcolor='white',
                    background=None,
                    bg_box=None,
                    font_path="./static/fonts/simfang.ttf",
                    line_pad=0,
                    line_height=None,
                    logo_path=None,
                    watermark=True,
                    dot_line=False):
```

后处理器配置

通用后处理器配置文件 config/post_processor_config.yaml 是后处理器的随机参数范围配置与使用概率配置
有线表格的后处理器配置放在其配置文件的 post_processor 的条目下，方便修改使用
在这里可以配置各种后处理器效果的随机参数范围与该效果的出现概率 ratio，如果不需要某种效果，可将其设置为 0;
该配置文件内的顺序与最后的效果有关，依次是：盖章>污染>折痕>噪声>扭曲>旋转>透视>背景，不要改变盖章和背景的位置。

```
  random_seal:
    seal_dir: static/seal
    ratio: 0.4
  random_pollution:
    dirty_dir: static/dirty
    ratio: 0
  random_fold:
    min_range: 0.25
    max_range: 0.75
    ratio: 0.5
  random_noise:
    max_prob: 0.01
    ratio: 0.5
  random_distortion:
    max_peak: 0.4
    max_period: 10
    ratio: 0.3
```

```
  random_rotate:
    min_angle: -0.5
    max_angle: 0.5
    ratio: 0.3
  random_perspective:
    min_offset: 0.01
    max_offset: 0.05
    ratio: 0.3
  random_background:
    bg_dir: static/background
    min_offset: 20
    max_offset: 50
    ratio: 0.3
```

项目结构

- config 保存配置文件
- static 保存项目用到的静态资源、背景、字体、标注文件、印章文件等
- data_generator 数据生成器包
  - data_generator.py 银行数据生成
  - fakekeys.py 读取标注文件生成通用表格可用的键值
  - uniform.py 通用表格生成器算法
- post_processor 后处理器包
  - noisemaker.py 噪声
  - seal.py 印章 包括 gen_seal 和 add_seal
  - rotation.py 旋转
  - perspective.py 透视
  - distortion.py 扭曲
  - label.py 写标注文件,显示标注
  - watermark.py 水印
  - random.py 包含以上所有的后处理器的随机参数版本
- utils 没地方放的工具
- awesometable.py 定义了 AwesomeTable (核心)
- factory.py 多线程表格工厂

运行方法

1. 安装所需依赖 见 requirements.txt

2. `python factory.py [type] [batch]` type 为类型：无线银行流水用 bank；其余有线与其配置文件同名如：baoxian yinghang huoyun yiliao zhizao；batch 为具体数量：如
   `python factory.py bank 10000` 生成10000张无线银行流水。