generate_action_combination.py

Code Structure

def main():

1. `load_charges('charges.json')` ：从'charges.json'文件加载charges数据
2. `extract_actions(charges)`：从charges数据中提取动作(actions)
3. `prepare_phrase_components(charges)`：准备生成短语所需的组件
4. `extract_keywords(actions)`：从动作中提取关键词
5. `analyze_pos(keywords)`：对关键词进行词性标注(POS tagging)
6. `generate_phrases(pos_data, phrase_components)`：基于词性分析结果和短语组件生成结果短语

特殊处理

1. **当结果短语长度为1时：**
   - 检查短语长度是否大于2
   - 进行词性分析验证
   - 移除特定词性的字符
   - 生成所有可能的组合

主要函数

1. **数据加载模块** (`load_charges`)
   - 从JSON文件加载charges数据
2. **动作提取模块** (`extract_actions`)
   - 从charges数据中提取动作短语
   - 处理中文顿号("、")分隔的短语
3. **括号处理模块** (`process_brackets`)
   - 处理中文括号内的可选内容
   - 生成括号内容的不同组合形式
4. **短语分割模块** (`split_crime_phrases`)
   - 将包含顿号的短语分割成多个部分
5. **关键词提取模块** (`extract_keywords`)
   - 使用jieba分词提取关键词
6. **词性分析模块** (`analyze_pos`)
   - 对关键词进行词性标注
   - 按词性分类存储词语
7. **短语生成模块** (`generate_phrases`)

- 基于词性组合规则生成短语
- 双词组合模式(如形容词+名词、动词+名词等)

if **name** == "**main**":特别处理

1. **数据加载：**
   - 从'word_counts_test.csv'文件中读取词语及其计数
   - 跳过表头行
   - 只保留计数小于10000的词语
2. **词性过滤：**
   - 定义无实际意义的词性列表(如代词、助词、数词等)
   - 使用`analyze_pos`函数分析每个词的词性
   - 移除被标记为无意义词性的词语
3. **词语清理：**
   - 移除包含已过滤词语的复合词

`get_word_count_n_new_fact.py`

Code Structure

主要功能

1. **动作数据加载：**
   - 从JSON文件加载预定义的法律动作列表
   - 将所有动作提取到`action_lists`中
2. **文本处理：**
   - `find_all_contexts_fast`函数：
     - 将输入事实按句号分割成句子
     - 找出包含预定义动作的句子
     - 统计每个动作出现的频率
     - 生成新的文本内容（只保留包含动作的句子）

`new_fact.py`

Code Structure

主要功能

1. **数据处理：**
   - 提取每个案例的标准charges信息
   - 使用charges.json中的映射关系转换charges为标签

- 收集所有出现的charges标签并排序
- 为每个charges标签分配新的编号

2. **数据整合：**

- 从'new_fact.json'加载处理后的案件事实
- 将案件事实与对应的charges标签组合
- 检查并输出包含多个charges的案例

3. **结果保存：**

- 将整合后的数据保存为`final_output.json`

总结

这三个Python脚本共同构成了一个完整的法律文本处理流水线，实现了从原始法律文书到结构化训练数据的转换。以下是三个代码文件的逻辑串联和整体流程分析：

1. 整体处理流程

```
原始法律文书 → 动作提取 → 事实文本过滤 → charges标签映射 → 训练
```

2. 各阶段功能分解

第一阶段：动作提取与短语生成

- 从`charges.json`提取法律动作短语
- 使用jieba分词和词性分析生成合理短语组合
- 输出`action_1.json`（动作短语库）

第二阶段：事实文本过滤

- 读取法律案例原始数据(`train.jsonl`)
- 使用生成的动作短语过滤事实描述
- 只保留包含关键动作的句子
- 输出：
  - `word_counts.csv`（动作频率统计）
  - `new_fact.json`

第三阶段：charges标签映射

- 加载原始charges映射(`charges.json`)
- 将案例中的charges标准化为数字标签
- 整合过滤后的事实与charges标签
- 输出`final_output.json`（结构化训练数据）