第二次编程作业：基于最大概率的汉语切分

> 任务：实现基于最大概率的汉语切分算法。
>
> 要求：采用语言模型，语言模型可以是传统n-gram语言模型，也可以是神经语言模型，如果用传统n-gram语言模型，n>1，并至少用Laplace平滑。

Class ChineseSegmenter

1. 说明：采用了传统的Bigram语言模型，并在计算概率P时使用了Laplace平滑
2. 属性
    1. `corpus`：处理后的训练语料。
    2. `word_dic`：根据 `corpus` 生成的Bigram词典。
    3. `bigram`：根据 `word_dic` 计算的参数，用于计算概率P。
3. 方法
    1. 静态方法
        1. `is_chinese`：判断一个unicode是否是汉字。
        2. `is_number`：判断一个unicode是否是数字。
        3. `is_alpha`：判断一个unicode是否是英文字母。
    2. 实例方法
        1. `is_other`：判断一个unicode是否其他字符（汉字、数字、英文字母之外的字符）。
        2. `fit`：根据输入的语料对模型进行初始化/学习/训练。
            1. `init_corpus`：根据输入的语料进行处理。
            2. `init_dic`：根据处理后的语料生成Bigram词典。
        3. **`seg`**：遍历输入的句子，把句子按汉字、数字、字母、标点符号切开，对其中的汉字进行MP切分。
            1. `P`：输入句子，根据Bigram词典计算该种切分结果下的概率，并采用拉普拉斯平滑。
            2. **`best_seg`**：输入句子，使用 `get_all_seg` 获得句子所有可能的切分结果，比较所有的切分结果，取最大概率的分法。
            3. **`get_all_seg`**：输入句子，根据Bigram概率得到句子所有可能的切分结果。
            4. `get_all_words`：输入句子，根据 `corpus` 找到一个句子中所有可能被切分出来的词和其对应的起始位置和结束位置。
4. 主要算法
    1. 方法 `seg`，算法如下：
        1. `for` 依次遍历句子中的每一个字符ch，并记录前一个字符pre
            1. `if` ch是字母、数字、汉字
                1. 将ch累积至对应类型的缓存
                2. `if` pre是字母或数字或汉字，但与ch类型不同
                    1. `if` pre是汉字
                        1. 调用 `best_seg` 对汉字缓存进行汉语切分
                    2. 将pre对应类型的缓存保存至result
            2. `if` ch是其它字符
                1. 将ch保存至result
                2. `if` pre是字母或数字或汉字，但与ch类型不同
                    1. `if` pre是汉字
                        1. 调用 `best_seg` 对汉字缓存进行汉语切分
                    2. 将pre对应类型的缓存保存至result

2. 遍历结束后，把将最后剩余的缓存保存至result（汉字则先进行汉语切分）
3. 返回result

2. 方法 `best_seg` ，算法如下：
    1. 调用 `get_all_seg` 获得所有的切分结果，算法如下：
        1. 先得到句子中所有可能被切分出的词all_words
        2. 使用seg_result保存所有可能的切分结果
        3. `while` 在所有的词all_words中找到当前需要进行继续拼接的词word
            1. `if` word开始于句首且未结束于句尾（还没有组成整个词序）
                1. `for` 在所有的词all_words中找到可以接在该word后面的词word_next
                    1. 拼接出一个新词word_new，并计算概率P_new
                    2. `for` 在所有词all_words中找到和该新词对应位置一致但不一样的词word_old
                        1. `if` word_old的概率P_old小于P_new，则移除word_old，避免后续无用计算
                        2. `else` P_new = P_old，
                    3. `if` 没有比word_new更好的切分word_old，则插入word_new，否则不添加新词
                2. 移除word，避免后续无用计算
                    1. 如果
            2. `elseif` word是一个满足条件的切分结果
                1. 把word插入seg_result
            3. `elseif` word开始于非句首
                1. `break`
        4. 返回seg_result
    2. 比较所有的切分结果，取最大概率的分法，返回best_seg

5. 缺点与改进空间
    1. 只实现了Bigram，没有设计n>2时的算法。
    2. 没有处理OOV，无法正确切分中文人名。
    3. 与PPT中的DP算法不同，没有保存MAP，而只是简单地计算了每种切分的概率并比较，因此在处理大规模文本时性能可能稍差