自然语言处理课程报告-中文文本纠错系统

北京理工大学 计算机学院 1120161874-谭超

一．问题：

怎样找出中文文本中的错别字、同音字以及实现中文文本中的错别字、同音字的纠正。

二．已有方法以及相关信息:

n-gram 模型：

对于一个句子，计算它出现的概率公式为

$$P(w_1, w_2, \cdots, w_m) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)\cdots P(w_m|w_1, \cdots, w_{m-1})$$

而 n-gram 模型是引入马尔可夫假设后的结果，即认为当前词仅与前 n 个词有关，例如：

bigram 为：

$$P(w_1, w_2, \cdots, w_m) = \prod_{i=1}^{m} P(w_i|w_{i-1})$$

trigram 为：

$$P(w_1, w_2, \cdots, w_m) = \prod_{i=1}^{m} P(w_i|w_{i-2}w_{i-1})$$

利用 n-gram 模型，当 n 越大时，模型越复杂，所以一般采用 bigram 或 trigram。

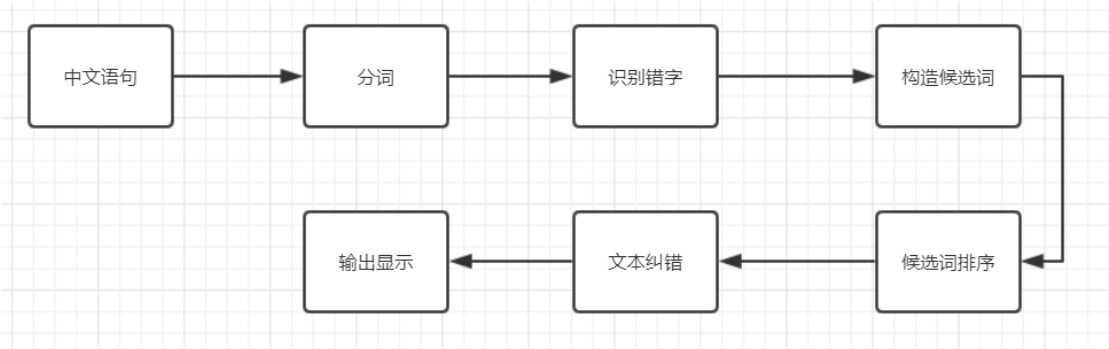
2.1 候选生成：对于纠错系统一般分为全部替换和单字替换，全部替换是假设每个字都是错的，对于每一个汉字都通过混淆集构造候选，然后利用语言模型进行打分，取得分最高且大于阈值的替换原来的字，这种方法能找出大部分可能，但效率太低。单字替换是认为分词之后的单字都是错的，对单字通过混淆集构造候选，然后通过语言模型打分，取得分最高且大于阈值的候选替换原来的字，单字替换较为简单，召回率较高，但同时会造成很多正确的单字也被修改。无论是全部替换还是单子替换都有不可避免的缺点，为了提高效率和正确率，可以利用未登录词识别找到无意义的单音节词素进行替换，这样就减少了将正确的单字修改的概论。

2.2 候选排序：利用语言模型对候选进行打分，得分最高且大于阈值的则作为纠正后的结果。

三．个人想法：

整个系统后端主要包含两个子任务：错误识别和错误纠正。

整个流程图如下：



3.1 错误识别：在识别错字时，首先利用 kenlm 训练出 2-gram 和 3-gram 语言模型，通过语言模型打分找到待纠错位置，在找待纠错位置时，由于不同句子中的得分差异可能比较大，若直接设置固定阈值则效果较差，所以我们设置相对的阈值，即每个字与其他字得分的偏离程度 。然后将待纠错位置与上下文结合到词典中进行查找，如果所有组合都找不到则视为错字，也就是说分词后为单字的就视为错字。

3.2 错误纠正：错误纠正时，利用混淆集构造候选词，通过语言模型对候选词进行打分排序。通过音近字和形近字从混淆集中找出错字的候选字，找回的候选字和上下文进行组合然后到词典中查找以过滤出有效的候选词。在有效的候选词中通过语言模型得分最高且大于阈值的则替换原来的错字。

3.2.1 混淆集：混淆集也就是音近与形近字典，由容易混淆的字符和词组成的数据集，由于当前使用拼音输入法比较普及，所以混淆集中音近词占比较大，形近词占比稍小一些。在本系统中，我们使用的是 SIGHAN-7 提供的混淆集。

四．实验结果及分析：

根据阈值的不同，我们会得到不同的实验结果:

阈值	Precision	Recall	F1
15	0.396	0.561	0.464
2.5	0.169	0.691	0.271

- 由上述结果可以看出，整体的准确率比较低，这与单字作为待替换位置有很大的关系。实验结果与阈值的选择有很大的关系，因为阈值的大小决定了哪些字可以被识别为错字，阈值越大，已识别的错字中确实为错字的概率越大，即准确率越大，检测出的错字占总错字的比例越小，即召回率越小，反之，阈值越小，准确率越小，召回率越大。

参考文献：

https://cloud.tencent.com/developer/article/1156792
https://blog.csdn.net/hqc888688/article/details/74858126