拼音输入法作业

电 62 班 李明轩 2015010705

一、 算法介绍与编程实现

算法基于二元字模型，基于隐马尔可夫模型进行实现。

$$p(w|o) \propto \prod_{i=1}^{n} p(w_i|w_{i-1})e(o_i|w_i)$$

w代表汉字，o代表拼音，二元字模型只运用了前一个字的信息，故用的是$p(w_i|w_{i-1})$，代表前一个字为w_{i-1}时当前字是w_i的概率。

$p(w_i|w_{i-1})$可由下式得到：

$$p(w_i|w_{i-1}) = \frac{p(w_{i-1}w_i)}{p(w_{i-1})} = \frac{cnt(w_{i-1}w_i)}{cnt(w_{i-1})}$$

为了避免连乘时出现$p(w_i|w_{i-1}) = 0$直接导致$p(w|o) = 0$的现象，故考虑对$p(w_i|w_{i-1})$进行平滑处理，即：

$$p(w_i|w_{i-1}) = \lambda\frac{cnt(w_{i-1}w_i)}{cnt(w_{i-1})} + (1-\lambda)\frac{cnt(w_i)}{cnt(all)}$$

其中λ为超参数。

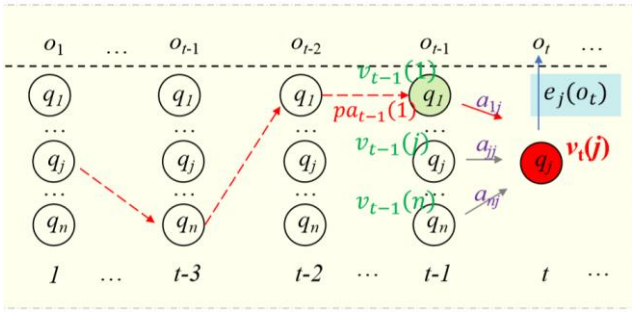
$e(o_i|w_i)$为发射概率，代表字w_i在句中的发音为o_i的概率。由于语料库中没有具体给句子中每个字的发音，因而暂不对多音字进行概率统计，即认为$e(o_i|w_i)$只为 0 或 1。

基于以上算式，依据拼音序列推断可能性最大的的中文字符串，即求解以下问题：

$$argmax_{\{w_i \dots w_n\}} \prod_{i=1}^{n} p(w_i|w_{i-1})e(o_i|w_i)$$

如果遍历所有解空间的话耗时过长，故用 Viterbi 算法，采用动态规划的方式寻找最优解，其时间复杂度为$O(n^2T)$，其中 n 代表隐状态数，即各拼音对应的汉字数量；T 代表拼音序列长度。

算法流程见下页。



Viterbi 算法示意图，图中q_i为隐状态，o_i为观测值[1]

本问题中在线求解算法流程如下：

1. 根据拼音汉字表，找到各拼音可能对应的汉字（即推断出各隐状态$q_t(i)$,其中 t 代表本句话中第 t 个字），由于不考虑多音字的概率分布，故$e(o_t|q_t(i)) = 1$，后续计算中**不再考虑发射概率** e 的影响
2. 对于第一个拼音 t=1,定义概率函数$v_1(i) = \pi_i = \frac{cnt(w_i)}{cnt(all)}$
3. 对于后面的拼音 t=2,...,T, j=1...n，依次进行如下迭代计算：

$$转移概率a_{ij} = \lambda\frac{cnt(w_iw_j)}{cnt(w_i)} + (1-\lambda)\frac{cnt(w_j)}{cnt(all)}$$

$$概率值v_t(j) = v_{t-1}(i) * a_{ij}$$

$$前缀字符\mathrm{pre_t}(j) = \underset{i}{\mathrm{argmax}}[\, v_{t-1}(i) * a_{ij}]$$

4. 最终得到:

$$p^* = \max_j v_T(j)$$

$$q_T^* = arg\max_j v_T(j)$$

5. 根据q_T^*及$pre_t(q_j^*)$的信息依次推出前面的字符q_{j-1}^*，最终得到字符串$\{q_1^*, \dots, q_T^*\}$，结束。

程序见 src 文件夹，其中 **data_prep.py** 将拼音字符表、二元字统计等内容以字典形式存储到 pkl 文件中；而 **hmm.py** 中的 **predict 函数**调用这些数据计算转移概率a_{ij}等参数，在此基础上按照 Viterbi 算法进行中文字符串的预测。

[1] 图片摘自自动化系《模式识别与机器学习》课件

二、 效果展示

用网络学堂发布的“拼音输入法测试样例.txt”进行测试，设置$\lambda = 0.9$，得到字、句准确率如下。

测试字数	3643
测试句数	365
字准确率	0.7977
句准确率	0.3205

效果好的例子：

ren gong zhi neng ji shu fa zhan xun meng
人工智能技术发展迅猛
人工智能技术发展迅猛

jin nian qing kuang bu tai hao
今年情况不太好
今年情况不太好

ji dong che jia shi yuan pei xun shou ce
机动车驾驶员培训手册
机动车驾驶员培训手册

ni zai gan shen me a tuan zhang
你在干什么啊团长
你在干什么啊团长

效果不好的例子：

① **新闻中语料的频次不同于现实中的常用语：**

wo qu gei ni mai yi ge ju zi
我去给你买一个橘子
我去给你买一个巨资

ta yang le yi zhi qing wa dang chong wu
他养了一只青蛙当宠物
他养了一致青瓦当宠物

② **新闻语料库可能未覆盖一些专有名词**

wei ji bai ke shi yi ge wang luo bai ke quan shu xiang mu
维基百科是一个网络百科全书项目
违纪伯克是一个网罗伯克全数项目

gei a yi dao yi bei ka bu qi nuo
给阿姨倒一杯卡布奇诺
给阿姨到一杯咖不奇诺

③ **二元字模型只用到上一个字，存在缺陷，可能需要用到三元/四元模型**

ni de li jie shi dui de
你的理解是对的
你的理解释对的

④ **对多音字的学习有所欠缺**

qing bu yao shu ru qi guai de ju zi
请不要输入奇怪的句子
情不要输入奇怪的车子

综上，可以看出本次使用的模型有两个主要缺陷：

1. 只用上一个字进行推断，没有结合前文的信息，因此需要考虑再用三元字模型进行修正。
2. 对多音字的处理不当，没有对发射概率进行学习，因此会输出一些不常用的多音字（比如“车(ju1)”）。

这些都是值得改进的地方。此外如果再用一些日常对话的语料库，准确率应该也会有所提升。

三、 参数选择

改变$\lambda\frac{cnt(w_iw_j)}{cnt(w_i)}+(1-\lambda)\frac{cnt(w_j)}{cnt(all)}$中的$\lambda$取值，得到准确率如下：

λ	字准确率	句准确率
0.6	0.7904	0.2904
0.7	0.7963	0.3014
0.85	0.8002	0.3178
0.9	0.7977	0.3205
0.95	0.7993	0.3205
1	0.8007	0.3233

可见平滑化处理对于本次学习并没有显著的帮助，而λ过小时准确率反而会下降.

四、 总结收获

总结：

尝试了手写隐马尔科夫模型进行序列预测，如果能够加入发射概率则模型会更完善，但没有多音字的概率数据所以就没考虑发射概率。

本次实验主要有两点收获，一是对于动态规划的高效性有了比较清晰的认识，二是认识到 python 中 dict 结构的强大。

可以改进的地方：

如报告中例子分析部分所述，一是值得考虑加入二元词模型或三元字模型，从而更好地利用前文的内容；二是获取数据得到发射概率，从而减少生僻的多音字带来的不良影响。