拼音输入法

陈新 2018013443

1 算法简介

最终在学习过程中采用了一阶隐马尔可夫二元字模型，在翻译过程中采用了 viterbi 算法。

学习

对于每个拼音，认为若某个汉字有这个读音则隐马尔可夫模型中的 B 矩阵相应值为 1，否则为 0。分析语料库中每一个二元组在语料库中出现的次数并计算$P(w_i|w_{i-1}) = \frac{P(w_i w_{i-1})}{P(w_{i-1})}$。

引入参数 λ，由于我计算最大概率$P = P_{首位}(w_1)\prod_{i=2}^{n}[(1-\lambda)P(w_i|w_{i-1}) + \lambda P(w_i)]$，所以转移矩阵中直接记录$(1-\lambda)P(w_i|w_{i-1}) + \lambda P(w_i)$。

翻译

加载学习好的信息，认为每个汉字只决定于上一个汉字的状态，通过 viterbi 计算 P 的最小值，并迭代找到该最小值的汉字序列。

具体实现过程

根据汉字表编号汉字，统计训练语料中的二元词组并计数，采用 laplace 平滑直接将$(1-\lambda)P(w_i|w_{i-1}) + \lambda P(w_i)$记录到 n*n 的矩阵。

翻译时用 viterbi 算法。

运行程序时无需外部库，采用 python3 以上

在 src 目录下命令　python pinyin.py ../data/input.txt ../data/output.txt

2 效果

正确长句：

人工智能技术发展迅猛
互联网还能够成为图书传播的平台
中国共产党员的初心和使命是为中国人民谋幸福为中华民族谋复兴

青山绿水就是金山银山
经济建设和文化建设突出了十八大精神的重要性
中国贫困地区实现网络服务全覆盖
走出清华大学特色的世界一流大学道路
夺取新时代中国特色社会主义伟大胜利
德国总理默克尔日前发表演说
消除恐惧的最好办法就是面对恐惧

可以看出表现较好的主要是与给我们的新闻语料相关的官方语句，翻译官方固定用语表现较为优秀。其余的一些无异议的句子也表现不错。

错误案例：

(1) 给阿姨倒一杯卡布奇诺
（错）给阿姨**到**一杯**咖**不奇诺
(2) 请不要输入奇怪的句子
（错）请不要输入奇怪的**车**子
(3) 你的理解是对的
（错）你的理解**释**对的
(4) 锐化空间滤波器
（错）**瑞华**空间**铝箔漆**

可以看出，以（2）为代表的错误句子主要是多音字的问题。

以（3）为代表的错误句子，原因在于我选用了二元模型，而不是三元、四元。在二元模型下，“解释”显然概率高于“解是”。

以（1）（4）为代表的错误在于训练语料的问题。由于给的训练语料主要集中在新闻等正是用语上，专业术语、生活用语很少，导致输入法在这些方面表现很差。

3 laplace 参数与其他改进尝试

对于一个总字数 3356，总行数 334 的偏向日常用语的测试集而言

训练数据大小（MB）	是否加 jieba 和 pypinyin	λ	正确字数	字正确率	正确行数	行正确率
948	否	0	2661	0.792908224	99	0.296407186
948	否	0.01	2666	0.794398093	100	0.299401198
948	否	0.02	2684	0.799761621	106	0.317365269
948	否	0.03	2663	0.793504172	100	0.299401198
948	否	0.04	2661	0.792908224	98	0.293413174
948	否	0.05	2658	0.792014303	98	0.293413174

117	是	0	2603	0.775625745	94	0.281437126
117	否	0	2581	0.769070322	90	0.269461078
993	否	0.02	2678	0.797973778	98	0.293413174

实验 1-6 比较了 laplace 参数的影响，7、8 实验比较了 jieba 和 pypinyin 库的影响，实验 9 添加了大约 45MB 的小说语料

（1）修改 laplace

laplace 平滑$(1-\lambda)P(w_i|w_{i-1})+\lambda P(w_i)$

从实验 1-6 可以发现，大约 0.02 左右是效果最佳的，故选取 0.02 作为最终参数。

（2）pypinyin 和 jiba 库

由于我对多音字概率的处理如下：如 zhuai 拼音有两个字“转”“拽”，所以 viterbi 算法运行时就考虑这两个字，附加概率乘上$P(转|zhuan)=1$（即不变）。之后我又修改了一个版本，利用 pypinyin 统计一音多字的概率，并将附加概率算成$P(转|zhuan)=\frac{P(转)}{P(zhuan)}$。

同时希望利用 jieba 库在现行计算的基础上加强词组的联系。

结果发现运行速度过慢，不得已只训练 117MB 的语料。对比不加这两个库且也只训练 117MB 语料的原版本而言，正确率有了很大提升：字正确率由 76.9%提升到 77.6%，句子正确率由 26.9%提升到 28.1%。但是由于一开始错误的操作，错误计算了 117MB 不加这两个库的训练方式的正确率，误以为这两个库对正确率没有提升，最后写报告时才核实了正确率，发现提升很大。但时间已经不够，还是提交了原版，不得不说十分可惜。

（3）语料影响

我认为语料与测试集的匹配程度对结果的影响十分巨大，但没有找到日常聊天的语料，因此采取小说代替，在 948MB 的语料上添加 45MB 小说语料进行训练。

但是奇怪的是，字正确率有所提升，但句正确率却有所下降，至今尚未弄清原因。

4 总结收获与改进方案

总结收获：

我觉得之后的作业必须留足时间先做，以免出现没有时间训练或者调整的情况，上述的 jieba 库和 pypinyin 库的应用就是一个很大的遗憾。

Viterbi 算法的应用加深了我对动态规划的理解。

Pypinyin 的应用加深了我对隐马尔可夫模型中 B 矩阵的理解，而非最先版本我简单的套公式。

改进方案：

修改数据结构，写一下三元模型。因为现在使用的是二位 list 存储，若直接改成三元会暴空间。之前尝试了改成字典，但是没有对词组统计进行剪枝，导致查找过程极慢，最终舍弃了三元模型。如果有时间，我想尝试一下三元模型。

希望将 Jieba 库和 pypinyin 库应用的那个版本跑完训练过程。

其次，语料库也应该进一步扩大，新闻的占比目前过大，不利于各种语言风格的广泛应用。

最后，可能从句尾反向构建句子也是一种思路，可以与正向构建相互参考。