**拼音输入法**

何仲凯 2017011335

1. 算法的基本思路和实现流程

在本次实验中，我实现了基于字的二元和三元模型。

二元模型就是指，当输入一个拼音序列，对每一个字考虑其在前一个字后面出现的概率。对可能出现的汉字序列，我们将这一个汉字序列中这样的所有的概率乘起来，再乘上该序列的句首汉字出现在句首的概率（指句首汉字在句首拼音下的概率），称为拼音序列呈现为该汉字序列的概率，那么我们的目的就是找到概率最大的那个汉字序列。

具体的算法实现参考了Viterbi算法。这是一种动态规划算法，可以用于较快地求解我们所想要的最优路径（最概然汉字序列）。该算法的核心思想是最优路径的子路径一定也是最优路径，否则这条子路径可以被最优路径替换从而变得更优。

类似地，三元模型就是指汉字序列中考虑的不只是前后两个字的关系了，而是每个字出现在前两个字后面的概率，具体实现起来同样是采用Viterbi算法，只不过复杂度要上一个层次，在计算每一个字（节点）的时候从二元的变成了（指一个拼音对应汉字的个数），也就是所有前面两个汉字的组合。显然，从理论上分析来说三元模型的性能会比二元模型好，运用了更多隐藏在拼音序列中的上下文信息。（如果二元就能完美地达成效果，那汉语也不会成为最难学的语言，而是最简单的语言了）

在弄明白算法原理后，就要开始实现了。首先，我们需要一些预处理，在这里，我处理了提供的文本数据，并从中得到以下信息。

1. 每一个拼音分别是每一个汉字的概率。
2. 每一个字后面跟每一个字的概率
3. 每两个字后面跟每一个字的概率
4. 每一个字发每一个音的概率（一元，作用在后面会介绍）

但是样本量虽然很大，但样本终究是有限的，而且涵盖方面也有局限性，都是新闻，所以肯定会有可能的合理的汉字序列没有出现在语料库中，自然不会被统计到，那么当我们希望得到这样的回答时，在我们的程序的认知中，这样的序列出现的概率是0，是不合法的，这就出现了问题，也就是说我们所统计出来的数据是有失偏颇的，离真实情况（虽然不知道定义真实情况）还是或多或少会有差距的，所以需要用到平滑算法。

平滑算法有很多种，最简单的平滑算法Laplace算法就是将每一个可能的出现次数加1。这样的做法可想而知效果非常有限，可以忽略，因为就算加了1那和0也没什么区别。本次实验中我采用了线性插值算法。

对于二元模型

对于三元模型

在实现完基础功能之后，实际测试数据后会发现多音字会给带来很大的误导。 所以我统计了每个字发每个音的概率在计算概率时考虑进去，希望可以规避掉这种情况。

1. 实验效果展示
2. 好的例子
3. 泥石流和森林火灾等灾害也有不同程度发生
4. 一季度低温雨雪冰冻灾害明显偏重
5. 十三届全国人大常委会第五次会议分组审议了全国人大常委会执法检查组关于检察传染病防执法实施情况的报告
6. 报告客观全面反映了贯彻实施传染病防治法的总体情况
7. 能去亚丁湾执行护航任务是一件很荣幸的事情
8. 小时候看到父亲严谨的作风和他帅气的军装
9. 差的例子
10. 清华大学自动画（化）系
11. 张学有（友）
12. 染色体上的遗址（位置）
13. 脚臭（娇羞）
14. 制药（只要）公司管理层认识到客服的重要性
15. 为记分（微积分）

a、b是对于二元模型而言的输出，若使用三元模型，就可以正确输出“清华大学自动化系”和“张学友”了，因为“自动化”和“张学友”都是一个三个字的词，三元以下的模型不会给三个字的词较高的权重，而三元模型则不然，所以这样的结果是预料之中的。

c、d都是典型的多音字带来的干扰，如“遗”念“wei”，“臭”念“xiu”的情况其实是非常少的，我们应该给予这样的情况较低的权重，在加入多音字频率表并使用三元模型时，这样的情况可以得到改善这两句都可以正确输出。

e出现的问题则体现出了三元模型的局限性，对三元模型而言，“制药公司”也是一个可以接受的结果，而且由于在开头，后面的拼音不会影响到前面的输出。

f则说明了语料库对结果的影响，因为选自新闻，所以不会出现很多学术术语，虽然确实出现过“微积分”的组合，但是出现次数实在过少，所以不会被优先考虑。

1. 参数选择对比与性能分析

由于涉及词汇的差异，不同的测试数据偏差有可能很大，我自己找了一些从新闻网上爬下来的文章，并从中选取了一篇表现较好且字数较多的进行测试，对同一个测试数据（4548 chars， 538lines）进行参数调整并进行测试，

1. 对二元模型平滑参数的对比选择

|  |  |  |
| --- | --- | --- |
|  | 字正确率 | 句正确率 |
| 0.1 | 57.4564% | 15.0558% |
| 0.01 | 76.4589% | 33.6431% |
| 0.001 | 81.4464% | 41.4498% |
| 0.0001 | 81.8204% | 41.2639% |

由表中数据可知选择为佳。

1. 对三元模型平滑参数的对比选择

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 字正确率 | 句正确率 |
| 0.1 | 0.1 | 76.8329% | 34.3866% |
| 0.01 | 0.01 | 86.3591% | 51.4870% |
| 0.001 | 0.001 | 87.9052% | 55.7621% |
| 0.0001 | 0.0001 | 88.1546% | 56.8773% |
| 0.00001 | 0.00001 | 87.7805% | 56.5056% |

最终选择参数。

1. 考虑多音字频率后的表现

二元模型（参数为0.001）

|  |  |  |
| --- | --- | --- |
| 是否考虑多音字 | 字正确率 | 句正确率 |
| 否 | 81.4464% | 41.4498% |
| 是 | 82.8678% | 44.4238% |

三元模型（参数为0.0001，0.0001）

|  |  |  |
| --- | --- | --- |
| 是否考虑多音字 | 字正确率 | 句正确率 |
| 否 | 88.1546% | 56.8773% |
| 是 | 88.5536% | 57.9926% |

从中可以看出考虑多音字确实对准确率的提升有所帮助，在测试多组数据时，可以发现效果提升大概在0.5%到2%之间不等。

1. 综合性能比较

选取了十篇数据进行测试，其中编号3是老师提供的以往同学提供的数据，虽然里面的测试数据很多比较奇怪，和语料库的数据区别有点大。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | 1 | 2 | 3 | 4 | 5 |
| 三元多音字 | 80.3279% | 92.9730% | 82.7893% | 94.2414% | 92.2743% |
| 46.0606% | 73.5484% | 42.6938% | 72.3810% | 67.1053% |
| 三元无多音 | 79.1439% | 92.7568% | 82.5061% | 94.2414% | 92.1777% |
| 44.8485% | 72.9032% | 41.6773% | 72.8571% | 67.1053% |
| 二元多音字 | 76.1384% | 90.0541% | 76.7467% | 90.5869% | 90.6808% |
| 39.3939% | 63.2258% | 27.5731% | 63.8095% | 60.0877% |
| 二元无多音 | 74.6812% | 89.1892% | 75.1012% | 90.0332% | 88.0734% |
| 36.9697% | 60.6452% | 25.5400% | 61.9048% | 55.7018% |
| No | 6 | 7 | 8 | 9 | 10 |
| 三元多音字 | 61.7707% | 88.5536% | 87.6984% | 90.2705% | 92.3939% |
| 57.2052% | 57.9926% | 61.9718% | 62.5000% | 66.0959% |
| 三元无多音 | 61.3877% | 88.1546% | 87.4008% | 89.4977% | 92.1938% |
| 56.1135% | 56.8773% | 59.8592% | 61.1702% | 65.4110% |
| 二元多音字 | 58.0761% | 82.8678% | 89.1865% | 85.5638% | 89.4315% |
| 45.4148% | 44.4238% | 63.3803% | 51.0638% | 60.2740% |
| 二元无多音 | 56.1162% | 81.4464% | 87.5000% | 84.2290% | 88.1505% |
| 39.5197% | 41.4498% | 58.4507% | 48.9362% | 55.8219% |

综合来看，可以发现性能由高到低如预期，分别是三元多音字、三元无多音字、二元多音字、二元无多音字，从二元到三元的性能的提升不是多音字频率表就能弥补的。如果实现了四元模型，那么可以合理估计四元无多音字会高于三元有多音字，四元有多音字还会更高。

1. 总结收获和改进方案

总结收获：这次实验非常有趣，算是第一次根据应用数学原理在问题的解决里，如这学期概率论课才学到的贝叶斯公式分析得到的原理实现了一个东西，在调研相关资料的时候还了解到了隐马尔可夫模型。在经过实际测试之后还发现效果不差。问题的定义是关键，也有同学使用神经网络等不同的做法，老师介绍的做法将看似复杂的问题简化为了搜索最优路径的问题，虽然描述还是近似的，但对于实现起来就简便多了，我也从中收获了很多。

改进1:这样定义问题实现出来的模型会优先考虑前面的字，优先由前面的字决定后面的字，实际上后面的字一定程度上也可以对推理出前面的字有一定的帮助，所以也可以反着建模，最后再对正反两个概率分别给一个权重，

改进2:我实现的多音字频率表虽然有一定效果，但还只是单纯地统计了频率，还是一元的，但多音字的发音是和上下文有关的比如“的”大多时候读“de”，但在“的确”、“目的”中的“的”一般是读“di”，只有在这些特殊情况下才会读“di”，所以如果把多音字表也做成多元的，对模型的描述会更加准确，效果应该会更好。

改进3:应该克服语料库的局限性，加入除了新闻更多方面的数据，这样当不管输入什么东西，模型都有几率可以进行正确的反馈。