人工智能导论 第一次作业 拼音输入法 实验报告

骆科运 致理书院-信计21 2022012405

一、实验环境介绍：

实验在windows11系统下进行，采用编程语言为python 3.13.2。

二、语料库与数据预处理方法介绍：

采用的语料库为新浪新闻2016年新闻语料库（sina_news_gbk）。

数据预处理方法：

1. 首先从拼音汉字表中提取汉字与拼音的多对多映射；
2. 提取每一篇新闻的html部分（即新闻正文），然后根据处理好的拼音汉字表将所有新闻正文划分为若干个文本段。（文本段长度在50000左右）

   每个文本段由若干句子（句子内部无任何符号）组成，句子之间用一个问号"?"分隔，无其他任何符号。

   然后使用**pypinyin**库对一个文本段注音（使用问号分割以及长度控制都是便于pypinyin处理），得到文本段的拼音。

   再从文本段中识别出每个短句，找到与其对应的拼音序列，然后扫描它并处理出**单字出现频率、两字出现频率、三字出现频率、四字出现频率 和 单字作为句首句尾出现频率，这些内容以json格式存储。**

三、基于字的二元模型的拼音输入法的介绍：

a. 原理与算法：

1. 由于汉字存在多音字的情形，因此我们将一个汉字和其一个发音组成一个唯一的pair，比如（“国”，“guo”），这样来处理多音字就不会有任何错误；
2. 对于一个给定的拼音序列O，我们实际上是需要找到一个pair序列S，使得P(S|O)最大。

   比如对于拼音序列 qing hua da xue，我们需要找到 ('清', ‘qing’) （‘华’，‘hua’） （‘大’，‘da’）（‘学’，‘xue’）。

   公式推导如下：

$$P(S|O)=\frac{P(S,O)}{P(O)}=\frac{P(O|S)*P(S)}{P(O)}$$

   注意到，由于S是pair序列，而每个pair都包含汉字和拼音，因此事实上有

$$P(O|S)=\begin{cases}1\text{，}S\text{中每个}pair\text{的拼音与}O\text{中对应位置拼音一致}\\0\text{，不一致}\end{cases}$$

   而我们进行搜索（动态规划）时，可以只去搜索满足拼音一致的pair序列，因此所有的P(O|S)值都**严格**为1，而P(O)与S无关，对于固定的O为常数，因此P(S|O)事实上可以由P(S)表示。

   对于一个pair序列S，

$$S=s_1s_2\dots s_n$$
$$P(S)=P(s_1)P(s_2|s_1)P(s_3|s_1s_2)\dots P(s_n|s_1\dots s_{n-1})$$

   在基于字的二元模型中，我们仅考虑一个字和它前一个字的耦合，忽略一个字与它前面一个字以前的所有字的关系，即

$$P(s_n|s_1s_2\dots s_{n-1})\approx P(s_n|s_{n-1})$$

   此时

$$P(S)=P(s_1)\prod_{i=2}^{n}P(s_i|s_{i-1})$$

其中$P(s_1)=\frac{cnt(s_1)}{total_{cnt}}$ ， $cnt(s_1)$是$pair\ s_1$在语料库中出现的次数，$total_{cnt}$是语料库中所有$pair$出现的总次数

$$P(s_i|s_{i-1})=\frac{cnt(s_{i-1}s_i)}{cnt(s_{i-1})}$$ ， $cnt(s_{i-1}s_i)$是$s_{i-1}s_i$这两个$pair$在语料库中连续出现的次数

由于$cnt(s_{i-1}s_i)$值可能为0，因此实际计算时将$\lambda P(s_i)+(1-\lambda)\frac{cnt(s_{i-1}s_i)}{cnt(s_{i-1})}$作为真实的$P(s_i|s_{i-1})$使用(平滑)

做变换：

$$-log(P(S))=-log(P(s_1))-\sum_{i=2}^{n}log(P(s_i|s_{i-1}))$$ ，将做变换过后的值记为F

3. 基于上述公式推导，我们可以确定以下算法流程：

    1. 首先对语料库进行预处理**（“训练”）**，处理好所有的$cnt(s_i)$与$cnt(s_{i-1}s_i)$;

    2. 对于第i个拼音，假设s_i是一个满足条件的pair，那么由二元模型的性质，计算所有以s_i结尾的pair序列的概率时，与前i-2个拼音对应什么pair都没有任何关系，而仅仅与第i-1个pair是什么有关系，这样的性质导出了如下的**动态规划**算法：

        记录所有以s_i pair结尾的pair序列中F值最小的pair序列是什么，以及其对应的F值是多少，然后按如下动规公式递推：

        $F(s_i)$表示所有以s_i结尾的序列中最小的F值

        $$F(s_i)=\begin{cases}min_{s_{i-1}}\{\ F'(s_{i-1})-log(P(s_i|s_{i-1}))\ \}\ ,\ i\geq 2\\-log(P(s_1))\ ,\ i=1\end{cases}$$

        在递推过程中记录取到最小F值的路径（即从头开始的pair序列）即可。

b. 实验效果：

选择新浪新闻作为语料库，训练时间为$1.5h$左右，主要耗费在调用**pypinyin**注音上。

在给定测试样例上**句准确率**为42.5%，**字准确率**为85.3%。生成所有给定测试样例总时间为3s。

```
D:\Work\新建文件夹\Tsinghua\人工智能导论\lab1\code1>python test.py
sentence ok: 213, cnt: 501, acc: 0.42515
word ok: 4469, cnt: 5235, acc: 0.85368
```

c. 例子分析：

好的例子：人工智能技术发展迅猛；

这个句子中成分都可划分为二字词语，“人工“，”智能“，”技术“，”发展“，”迅猛“，而且这些二字词语与同拼音的其他二字词语相比较，其出现频率呈现断崖式的高，因此最终能够非常”稳妥“地锁定正确答案，其F值除以字数为3.92，属于较确定的猜测。

坏的例子：地道战是一种以地道为策略应用的陆军步兵战术；

运行结果：地道站是一种异地到位策锃应用的陆军步兵战术；

这个句子结果不好，其F值除以字数达到了7.19，属于较犹豫的猜测。分析其原因主要在于句子不能完全划分为二字词语，存在一些较复杂的语法，比如”是一种“、“...的...”等。同时”以地道为“又存在一个较明显的错误划分“异地”“到位”，因此对猜测产生了一定混淆。

四、思考题：

a. gbk和utf-8的区别：

gbk是一种编码范围包括ASCII码以及简体中文的编码方式，ASCII码占1字节，中文字符占2字节；

utf-8是一种编码范围覆盖全球所有语言的字符的编码方式，采用可变长度编码（1字节到4字节），ASCII码占1字节，中文字符占3字节；

总的来说，utf-8相比gbk更加通用、国际化，往往用于处理跨平台、数据库、网页等场景，而GBK仅仅适用于中文场景。但在处理中文字符时gbk编码比utf-8占用更小的空间。

b. 假设读音数量为常数m，则大小为V的字库（且读音均匀分布）中平均一个读音有$\frac{V}{m}$个对应的字。

那么在使用**viterbi**算法进行动态规划时，句子长度为n，则需要进行$O(n)$层转移，每层转移需要计算$O(\frac{V}{m})$个F值，而每计算一个F值需要考虑$O(\frac{V}{m})$种可能的转移（上一个字是什么），因此总共时间复杂度为$O(\frac{nV^2}{m^2})$。如果考虑m为常数，则时间复杂度为$O(nV^2)$。

在进行动态规划时可以使用**滚动数组**的空间优化技巧，这样空间复杂度为$O(\frac{V}{m})$，考虑m为常数，则为$O(V)$。但训练得到的数据集也是空间复杂度的一个重要部分，理论上每两个字都可能成为一对连续出现的两字，存储它们出现次数的空间复杂度为$O(V^2)$，因此总空间复杂度应为$O(V^2)$。

c. 仅从输入输出角度来看，代码A采用的方式是读入一次后立刻进行输出而不是进行存储，代码B采用的方式是读入数据后先将其存入list当中，在所有数据都读入完成过后再依次进行输出。在实际应用中，代码B的方法不能实时输出，同时还会占用更多空间，不是一个好的选择。

五、实现的其他模型或算法介绍：

相比于基于字的二元模型，在实验二中最后采用了基于字的四元模型，并且加入了对于**句首**的判断。（还有一定剪枝策略）

基本思路：

1. 采用基于字的四元模型：

   将原本的

   $$P(s_n|s_1s_2\dots s_{n-1}) \approx P(s_n|s_{n-1})$$

   更改为

   $$P(s_n|s_1s_2\dots s_{n-1}) \approx P(s_n|s_{n-3}s_{n-2}s_{n-1})$$

   以增加字与字之间的耦合关系，去模拟“词”的存在。

   更改过后依然采用动态规划的方法去计算，转移方程为

   $$F(s_{i-2}s_{i-1}s_i)表示所有以s_{i-2}s_{i-1}s_i结尾的序列中最小的F值$$
   $$F(s_{i-2}s_{i-1}s_i) = min_{s_{i-3}s_{i-2}s_{i-1}}\{\ F'(s_{i-3}s_{i-2}s_{i-1}) - log(P(s_i|s_{i-3}s_{i-2}s_{i-1}))\ \}$$

   对于i不足4的情况特殊处理即可。

   同时依然使用**平滑**方法去处理四元字组出现次数为0的情况。

   $$P(s_n|s_{n-3}s_{n-2}s_{n-1}) = \lambda\frac{cnt(s_{n-3}s_{n-2}s_{n-1}s_n)}{cnt(s_{n-3}s_{n-2}s_{n-1})} + (1-\lambda)P(s_n|s_{n-2}s_{n-1})$$

2. 增加对于句首的判断：

   $$P(s_1|h) = \lambda\frac{cnt(hs_1)}{cnt(h)} + (1-\lambda)\frac{cnt(s_1)}{total_{cnt}}$$

   用此概率去代替原本作为句子开头的pair的概率$P(s_1)$，可以利用字在语料库中的位置信息，增加句首单字猜测的准确率。

3. 剪枝策略：

   使用最终预测答案的F值除以句子长度（字数），可以得到一个平均F值，一般在2~7之间。因此可以将动态规划过程中F值已经超过了10 * 句子长度的状态全部剪去，可以一定程度上不影响求解效果地加快求解速度。

例子分析：

选取例子”创意来自中国的传家宝“，对于二元模型而言，其很难通过两个字的耦合关系准确预测出”传家宝“三字，因此其只能预测出

”创意来自中国的传家暴“，因为”传家“和”家暴“两词出现频率都较高。但对于四元模型而言，”传家宝“一词出现了190次，因此其能够正确预测。同样的例子还有”第一批九零后已经三十岁了“，二元模型的预测结果是”第一啤酒令后已经三十岁了“，因为其没有”九零后“的概念。

但四元模型也有表现较差的例子，比如”武汉零新增病例给世界以希望“，四元模型最后几字的预测是”世界一系网“，反而二元模型预测出了”世界一希望“。经过分析，这是因为四元模型采用了剪枝策略，由于单字“系”比“希”出现频率高很多，在临界条件处“希”刚好被剪枝剪掉了。其余情况四元模型均有较好的表现。而四元模型的坏处在于其搜索速度极其缓慢，平均1min才能跑出一个句子的推测结果。

效果对比：

（三个模型训练耗时相同，均为1.5h左右）

模型	字准确率	句准确率	生成所有测例耗时
二元	85.3%	42.5%	3s
三元（句首优化）	90.0%	59.5%	4min40s
四元（句首优化）	89.9%	59.9%	3h

可以发现提升到四元过后字准确率甚至还稍有下降，句准确率稍有提升，但生成耗时大大增加。

这应该是因为四元的剪枝策略使得四元的效果没有完全发挥出来，因此一些表现还不如三元。

六、调参分析

由于使用基于字的四元模型生成速度过慢，不能自由灵活地切换参数并尝试，因此调参分析过程针对基于字的三元模型进行。

主要调整参数为平滑参数λ（三元模型中有两个需要调整的参数）：

当$P(s_3|s_1s_2):P(s_3|s_2)=7:3$ $P(s_3|s_2):P(s_3)=9:1$时，三元模型句准确率为59.5%；

当$P(s_3|s_1s_2):P(s_3|s_2)=99:1$ $P(s_3|s_2):P(s_3)=99:1$时，三元模型句准确率为57.5%；

当$P(s_3|s_1s_2):P(s_3|s_2)=1:1$ $P(s_3|s_2):P(s_3)=1:1$时，三元模型句准确率为52.9%；

这样的结果是因为三种参数对应了三种“推理策略”。

第一种属于兼顾型，考虑到三元词组对推理的重要性比二元词组大，二元又比单字大，并且单字所占权重较小。

第二种属于非常相信多元词组型，推理过程中会优先考虑多字之间的耦合关系，坏处是在耦合关系弱的句子中表现较差，比如”...是...“。

第三种属于非常不相信多元词组型，推理过程中认为多字之间的耦合关系与单字是同等重要的，这与常识是相悖的，因此表现也最差。

七、其他评价指标

由于使用基于字的四元模型生成速度过慢，不能自由灵活地对输出分析，因此其他评价指标分析过程针对基于字的三元模型进行。

可以使用最终答案的F值除以句子字数得到**平均F值**，以此来衡量程序对此次猜测的“把握程度”，比如：

“江苏省委省政府决定成立调查组”句子的平均F值为2.699，因为其划分为二元三元词非常清晰，预测也非常确定；

“十七张牌你能秒我”的预测结果是“时期长排你能描我”，其预测非常差，平均F值达到了8.452，这是因为这个句子与语料库风格不匹配，难以用词频统计来预测。

使用平均F值，还可以帮助四元模型的剪枝策略，对于动态规划过程中平均F值非常大（>10）的句子，可以直接将其剪去以缩小后续搜索范围。

八、实验感受

完成实验的总时间大概在15h左右，代码工作量在500行左右，大部分时间在debug以及等待程序运行并实验。

我认为这次实验是一个不错的让同学们接触并处理**较大规模数据**的机会，经过此次实验过后**收获较大**。

在做实验之前，我一直认为拼音输入法是一个需要非常大规模数据来完成的任务，流行的输入法一定要么有巨量的数据库、要么有精妙的算法。但实际上上手完成之后才发现只是普通的数据统计 + 动态规划即可完成。

事实上实验需要处理的语料库达到大约7亿个字，确实属于较大规模数据，因此与平时小规模的普通算法或实验不同，本次实验中每一步处理和尝试都需要花费较多时间成本，这样的体验丰富了我的经验，也让我学习到了需要大量时间成本的实验中处理和测试数据和算法的方法：先在小规模数据上测试和debug，再在大规模数据上进行完整实验。

除了收获较大以外，实验二的内容鼓励同学们积极探索新的模型和方法，同时又不设统一的总榜，遏止了恶意卷榜的风气，我非常赞同这样的实验形式。