拼音输入法实验报告

石大川 <sdc17@mails.tsinghua.edu.cn>

2020 年 4 月 11 日

目录

1 介绍

在本次实验中，本文先后实现了基于二元字模型，三元字模型和二元词模型的拼音输入法。其中三元字模型在群里发的一份去年的测试集上达到了相对最高的单字准确率 81.33% 和句子准确率 30.48%。此外本文另外引入了句首和句尾两个虚拟字/词，并通过实验给其赋予了相对于其他位置的二元/三元关系更大的权重。

运行环境和各个模型的使用方法汇总在 README.md，在里面复制粘贴即可。

2 实现

2.1 二元字模型

用法：python predict.py -i=./data/input.txt -o=./data/output.txt –model_type=2c

2.1.1 朴素形式

每一个字出现的概率仅仅和其前一个字相关：

$$P(S) = \prod_{i=1}^{n} P(w_i|w_{i-1}) \tag{1}$$

其中每一项概率由带平滑的频数之比给出：

$$\lambda \frac{\#\{w_{i-1}w_i\}}{\#\{w_{i-1}\}} + (1-\lambda)P(w_i) \tag{2}$$

最后用 viterbi 算法求解式 (1) 的最大值并 backtrace 找出相应的路径：

$$P(W_{i,j}) = \max_k (P(W_{i-1,k}) * P(W_{i,j}|W_{i-1,k})) \tag{3}$$

朴素的二元字模型在前述的数据集上单字准确率为 0.7319，句子准确率为 0.1637。

2.1.2 引入句首句尾

在朴素二元字模型的基础上对输入的每一个句子引入一个虚拟的句首和句尾，其中句首与句子第一个字之间构成二元字关系，句子的最后一个字和句尾之间也构成二元字关系。

具体来说，首先在统计二元字频，即生成 2gram 时对于每个连续的合法中文字符串增加一个‘s 字首’和‘字尾 t’的键值。在解码时将‘s’视为每个句子的第一个字，将‘t’视为句子的最后一个字，将 viterbi 算法同样地应用在这两个节点之上。

引入句首句尾后，在前述的数据集上单字准确率为 0.7331，句子准确率为 0.1662。

2.1.3 句首句尾权重

在计算句首和句尾的二元字关系的概率时，赋予其一个大于 1 的权重，具体来说就是当处理‘s’和‘t’节点时在式 (2) 中引入权重参数 st：

$$st \times \lambda \frac{\#\{w_i w_{i-1}\}}{\#\{w_{i-1}\}} + (1-\lambda)P(w_i) \tag{4}$$

参数 st 和准确率的关系测试如下。

参考Figure 1可知，取 $st = 25$ 时可得单字准确率最大值 0.7396 和句子准确率最大值 0.1700。

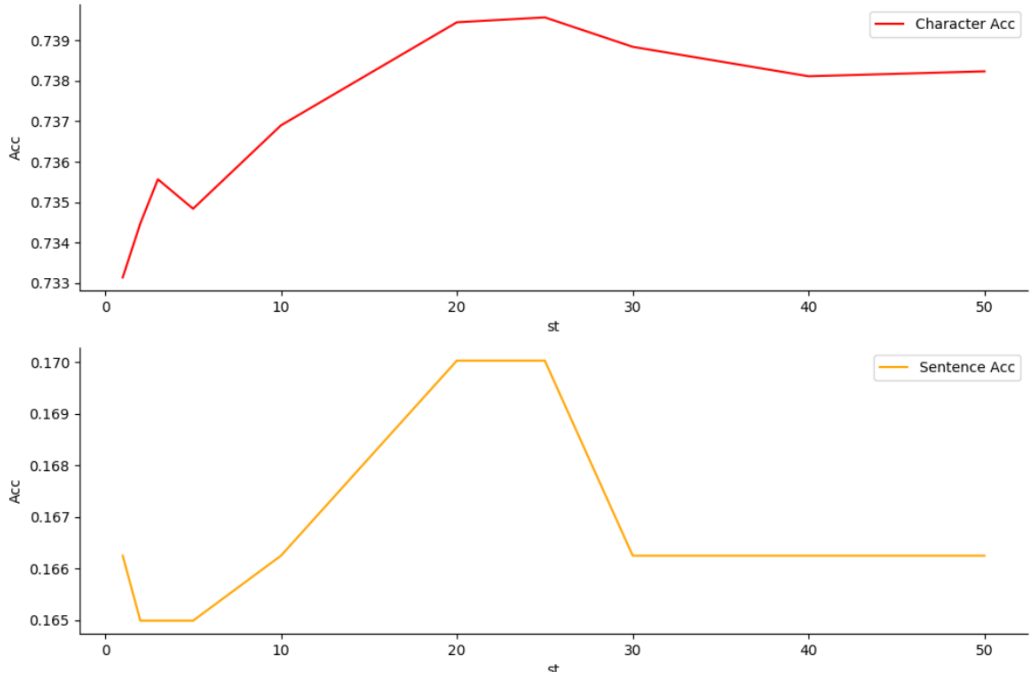

Figure 1: 参数 st 和准确率之间关系

2.2 三元字模型

20% 模型用法：python predict.py -i=./data/input.txt -o=./data/output.txt –model_type=3c –full_model=False

全模型用法(默认):python predict.py -i=./data/input.txt -o=./data/output.txt –model_type=3c –full_model=True

2.2.1 朴素形式 + 句首句尾

三元字模型中每一个字出现的概率仅仅和其前两个字相关，式 (1) 和式 (2) 分别变为：

$$P(S) = \prod_{i=1}^{n} P(w_i|w_{i-1}, w_{i-2}) \tag{5}$$

$$\lambda \frac{\#\{w_{i-2}w_{i-1}w_i\}}{\#\{w_{i-2}w_{i-1}\}} + (1-\lambda)P(w_i) \tag{6}$$

解码同样使用 viterbi 算法，不再赘述。朴素形式的三元字模型加上句首句尾节点的引入后，在前述的数据集上单字准确率为 0.7837，句子准确率为 0.2569，可见相比二元字模型有了大幅提升。

2.2.2 朴素形式 + 句首句尾权重

同二元字模型进一步引入句首句尾权重：

参考Figure 2可知，取 $st = 40$ 时取得单字准确率最大值 0.7853 和句子准确率最大值 0.2582。

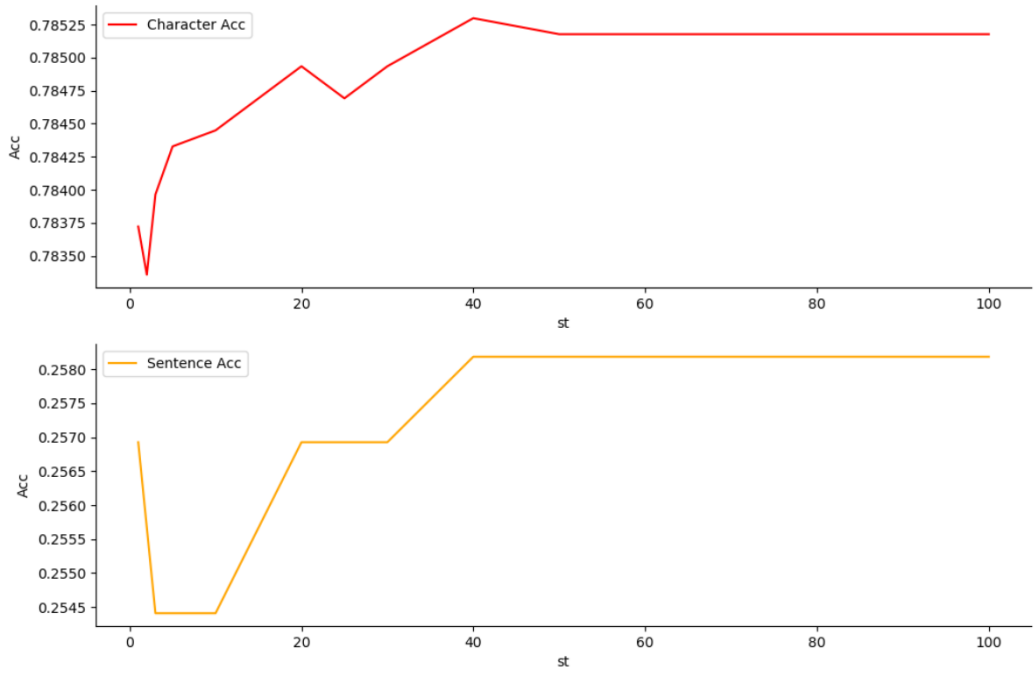

Figure 2: 参数 st 和准确率之间关系

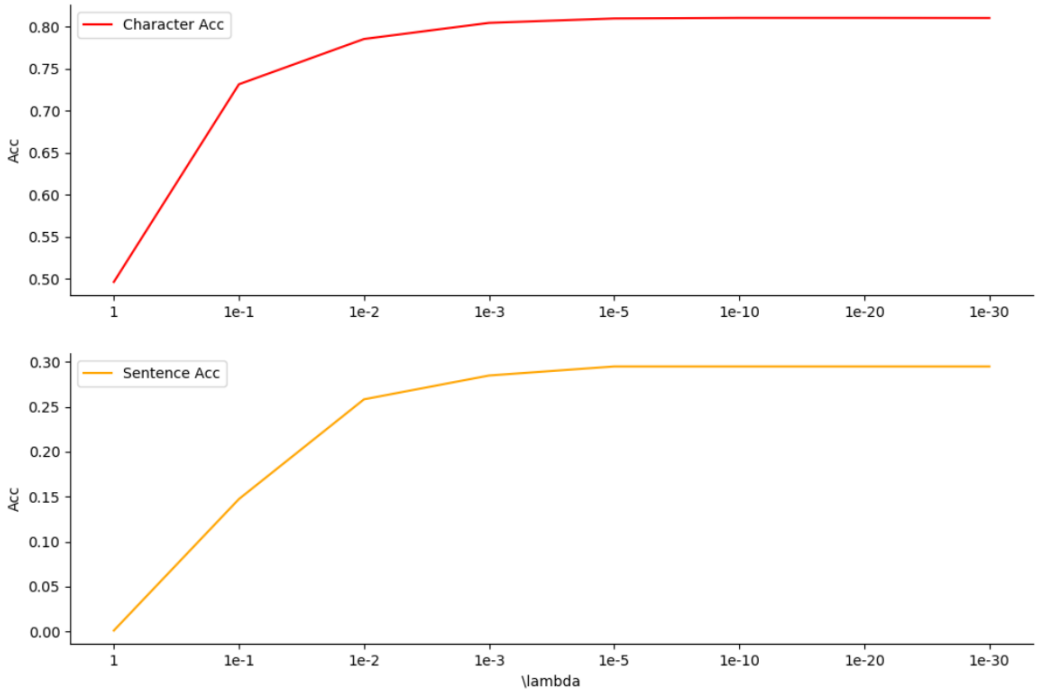

Figure 3: 平滑系数 λ 和准确率之间关系

2.2.3 平滑系数 λ

调整平滑系数 λ 的值做对比实验：

参考Figure 3可知，取 $\lambda <= 1e-10$ 时，准确率几乎不再变化。取 $\lambda = 1e-10$ 可得单字准确率最大值 0.8104 和句子准确率最大值 0.2947。相比与上一个小节中的三元字模型，准确率得到了明显的提升。

2.2.4 字频截断

字的三元关系比较大，因此上述实验中使用的 3gram 都按其出现的频数截取了前 20%，这将在一定程度上减小模型的加载时间成本和存储 gram 的空间成本，但是也会牺牲一定的准确率。

完整 3gram 按照 pkl 格式序列化后大小为 $571M$，截取前 20% 后为 $114M$，准确率对比见表 1

Model	20%	100%
Char Acc	0.8104	0.8133
Word Acc	0.2947	0.3048
Size	571M	114M

表 1: 模型规模对准确率的影响

2.3 二元词模型

用法：20% 模型用法：python predict.py -i=./data/input.txt -o=./data/output.txt –model_type=2w –full_model=False

全模型用法（默认）：python predict.py -i=./data/input.txt -o=./data/output.txt –model_type=2w –full_model=True

2.3.1 朴素形式 + 句首句尾

使用 jieba 的精确模式进行分词，然后将二元字模型中字的概念替换成这里的词即可。

具体来说用分词后的结果分别统计出一元词模型 1word.pkl 和二元词模型 2word.pkl。解码时采用的算法仍然是 viterbi 算法，和二元字模型解码不同的是词的长度不再固定是 1，因此解码时每轮迭代的拼音音节跨度取决于词的长度，因为中文中二元词比较多，所以我固定了每轮迭代跨度为 2 个音节。

句首句尾部分基本同之前的模型，不再赘述。

朴素的二元词模型在前述的数据集上单字准确率为 0.5827，句子准确率为 0.0718。

2.3.2 词频截断

截断概念同三元字模型，截断的比例为前 25%。

完整 2word 按照 pkl 格式序列化后大小为 $572M$，截取前 25% 后为 $140M$，准确率对比为表 2

Model	20%	100%
Char Acc	0.5827	0.5898
Word Acc	0.0718	0.0756
Size	571M	140M

表 2: 模型规模对准确率的影响

3 效果展示

用全三元词模型的结果作为展示。

效果较好的例子见表 3, 可见对于一些较长较复杂的句子，输入法也能取得较高的准确率。

输入	输出
wo geng ai ni xian zai bei shou cui can de rong yan	我更爱你现在备受摧残的容颜
bei jing ao yun hui kai mu shi fei chang jing cai	北京奥运会开幕式非常精彩
quan guo ren min dai biao da hui zai bei jing ren min da hui tang long zhong zhao kai	全国人民代表大会在北京人民大会堂隆重召开
ju you liang hao de gou tong neng li he jiao liu neng li	具有良好的沟通能力和交流能力
dang ran wo men de jue ding quan ye shi hen zhong yao de	当然我们的决定权也是很重要的
wo ye mei you shen me bie de dong xi song gei ni men	我也没有什么别的东西送给你们
ye wei le guang da he fa yong hu de zheng dang quan yi	也为了广大合法用户的正当权益
Shen du shen jing wang luo dui ji suan zi yuan de xiao hao hen da	深度神经网络对计算资源的消耗很大
Gou li guo jia sheng si yi	苟利国家生死以
Qi yin huo fu bi qu zhi	岂因祸福避趋之

表 3: 效果较好的例子

效果较差的例子见表 4，可见对于大部分效果较差的例子，通常是一句话中个别词出现错误，稍加思考可以知道其原意。但是也有少数句子如最后一行所示，解码出来的结果是完全不知所云的。

输入	输出（括号内为纠正）
Qing wa shi cong ke dou bian lai de	青蛙是从科都（蝌蚪）变来的
Zhe shi shi jie shang zui hao de bian cheng yu yan	这是世界上最好的变成（编程）语言
Er ji de jia ge shi yin xiang de san bei	耳机的价格是印象（音响）的三倍
Ni men cheng li ren zhen hui wan	你们成立认（城里人）真会玩
ting han si zhong sheng qing ye fo	挺韩四种生亲谒佛（听寒寺钟声请野佛）

表 4: 效果较差的例子

4 总结

总的来说三个模型从准确率角度来看，相对成功的是三元字模型。对于二元词模型其准确率不高的原因主要是固定了迭代的跨度为 2 个音节。对于包含三音节的词例如“输入法真难写啊”就会被截断成“输入 | 法真 | 难写 | 啊”进行解码，显然“法真”这里就很难解码出真确的结果。

5 改进方向

基于字的模型　还可以尝试四元字模型。

基于词的模型　如总结一节中所言可以将迭代的跨度改为动态数目的音节，让其适配三音节词，四音节词等多音节的词。

多音字的处理　可以进一步在模型中加入对多音字的特殊处理。

超参数　句首句尾的权重是从实验经验中得到的超参数，可能出现过拟合。可以考虑将其改为自适应参数或者增大训练语料的规模。