拼音输入法 实验报告

计 82 郑凯文 20181011314

2020 年 4 月 12 日

本实验分别使用字的二元模型和三元模型来完成拼音输入法。

1 算法原理

对拼音输入法的概率建模 我们的任务目标是将拼音序列转化为汉字序列。由于每个位置上的拼音给定，拼音到汉字的转化列表也已知，我们需要从每个位置上有限的候选汉字中选出某一个，使得整个序列出现的概率最大。设序列长度为 n，第 i 个位置上的汉字为 w_i，这也就是

$$\max p(w_1 w_2 \dots w_n)$$

字的二元模型 由于考虑句子整体需要在整个状态空间中搜索，需要指数级别的成本，且整体概率无从估计，因此结合语言特点进行一些简化。二元模型假设某个位置上某个汉字出现的概率只依赖于上一个汉字，即

$$p(w_1 \dots w_{r+1} | w_1 \dots w_r) = p(w_r w_{r+1} | w_r)$$

这样我们只需要考虑句子中相邻两个字构成的二元组。

字的三元模型 三元模型在二元模型的基础上，增加模型的复杂度，考虑某个汉字对之前的两个汉字的依赖，即

$$p(w_1 \dots w_{r+1} | w_1 \dots w_r) = p(w_{r-1} w_r w_{r+1} | w_{r-1} w_r)$$

词频统计与概率估计 无论是二元模型还是三元模型，对整个句子概率的估计均按照从前到后逐个汉字生成的方法，对应于概率的不断相乘过程

$$p(w_1 \dots w_{r+1}) = p(w_1 \dots w_r) p(w_1 \dots w_{r+1} | w_1 \dots w_r)$$

在从前向后的概率计算中，需要用到条件概率 $p(w_r w_{r+1}|w_r)$ 或 $p(w_{r-1}w_r w_{r+1}|w_{r-1}w_r)$。我们采取从大量语料中统计的方法，使用频率作为概率的估计，这也是最大似然估计给出的结果。设 $num\_character$ 为语料中参与统计的总字数，$num(w_r)$ 为 w_r 出现的次数，$num(w_r w_{r+1})$ 二元组出现的次数，$num(w_{r-1}w_r w_{r+1})$ 出现的次数，则对条件概率的估计为

$$p(w_r) = \frac{num(w_r)}{num\_character}, p(w_r w_{r+1}|w_r) = \frac{num(w_r w_{r+1})}{num(w_r)}, p(w_{r-1}w_r w_{r+1}|w_{r-1}w_r) = \frac{num(w_{r-1}w_r w_{r+1})}{num(w_{r-1}w_r)}$$

平滑化方法 由于语料有限，不可能覆盖全面，对于某些二元组或三元组很可能出现统计数量为零的情况。为了对语料没有涵盖的词组进行合理“推测”，采用平滑化的方法。对于字的二元模型，将单字概率和二元组的条件概率进行归一化加权作为生成概率

$$p(w_1 \dots w_{r+1}|w_1 \dots w_r) = \lambda p(w_{r+1}) + (1-\lambda)p(w_r w_{r+1}|w_r)$$

其中 λ 为引入的超参数。字的三元模型中则引入超参数 λ_1, λ_2，将单字概率、二元组的条件概率和三元组的条件概率三者进行加权

$$p(w_1 \dots w_{r+1}|w_1 \dots w_r) = (1-\lambda_1-\lambda_2)p(w_{r+1}) + \lambda_1 p(w_r w_{r+1}|w_r) + \lambda_2 p(w_{r-1}w_r w_{r+1}|w_{r-1}w_r)$$

维特比算法 虽然我们已经得到了上述的逐字生成过程，但我们需要选择的是累乘到最后一项时最大概率的序列。若采取暴力枚举，仍需指数级别的复杂度。这里巧妙利用下一个状态只与当前状态有关的特性，应用称为维特比算法的 dp 算法。以字的二元方法为例，用 $f[i][w]$ 表示第 i 个位置处汉字为 w 时的最大概率，用 $last(w)$ 表示达到最大概率时 w 的上一个汉字，用 W_{i-1}, W_i 表示第 $i-1, i$ 个位置的候选汉字，则转移方程为

$$f[i][w] = \max_{w_{i-1} \in W_{i-1}} p(w_{i-1}, w) f[i-1][w_{i-1}]$$

$$last[w] = \arg \max_{w_{i-1} \in W_{i-1}} p(w_{i-1}, w) f[i-1][w_{i-1}]$$

这里 $p(w_{i-1}, w)$ 是序列 $w_1 \dots w_{i-1}$ 到序列 $w_1 \dots w_{i-1} w$ 的转移概率。这样，取 $f[n][w]$ 最大的 w，并根据记录下的 $last()$ 值向前回溯，就能从后向前得到概率最大的序列。

2 文件结构与运行方法

./src/loadtable.py 将拼音到汉字的转换列表”./data/table.txt” 转化为 json 并保存至”./src/table.json”，便于直接加载。

表 1: λ 的选择对逐字准确率的影响

λ	0	0.01	0.02	0.05	0.1	0.2	0.5
逐字准确率	74.08%	74.08%	73.78%	73.47%	73.16%	72.24%	71.94%

表 2: λ_1, λ_2 的选择对逐字准确率的影响

λ_1	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
λ_2	0.99	0.89	0.79	0.69	0.59	0.49	0.39	0.29	0.19	0.09
逐字准确率	74.69%	76.12%	76.43%	76.33%	76.33%	75.71%	76.22%	75.92%	75.82%	75.00%

./src/loadfrequency.py 将读取"./datasets" 文件夹下所有文件，并作为语料进行统计，将结果保存至"./src/frequency.json" 中。注意语料文件需使用 utf-8 无 BOM 格式编码，否则会引起报错。这里已经导出了统计好的文件，可直接使用，且没有附带语料文件。由于没有进行频率较低词的剔除，且使用浮点数保存，"./src/frequency.json" 具有 1.25G 的较大体积。

./src/character2.py 是使用二元模型的转换程序,内部可以切换为调参模式。默认读入"./data/input.txt" 中的拼音，并将转换结果输出至"./data/output.txt" 中。由于频率统计文件较大，程序初始化阶段可能需要几分钟时间，但转换时速度很快。

./src/character3.py 是使用三元模型的转换程序，其他用法同上。

3 参数选择

我在众筹的测例中挑选了 100 条有代表性的例子，拼音文件在"./data/input.txt" 中，正确的输出文件在"./data/output_.txt" 中，以此测例上的逐字准确率为根据进行调参。

3.1 λ 的选择

结果如表 1所示。可见，λ 较小时影响不明显，而随着 λ 增大，准确率有所下降。最终设置 $\lambda = 0.01$。

3.2 λ_1, λ_2 的选择

单字频率的权重保持 0.01，改变 λ_1, λ_2。结果如表 2所示。最终选择 $\lambda_1 = 0.2, \lambda_2 = 0.79$。

可见使用三元模型后，整体的准确率有所上升。

4 效果展示与对比

上述参数下二元模型和三元模型的输出分别位于"./data/output1.txt" 和"./data/output2.txt" 中。

4.1 二元和三元模型表现好的例子

- 两会在北京召开
- 哲学是一切学科的基础
- 我和我的祖国一刻也不能分割
- 金庸的武侠小说非常精彩
- 小朋友们都喜欢去郊游
- 全国人民代表大会在北京人民大会堂隆重召开
- 深度学习技术推动了人工智能的发展
- 为了方便大家测试自己的拼音输入法的性能
- 吹啊吹啊我的骄傲放纵
- 毛泽东思想和中国特色社会主义理论体系概论
- 对亚洲人的种族歧视已经在全球蔓延
- 床前明月光
- 西方国家应对疫情措施不够

4.2 二元和三元模型表现不好的例子

第一行为正确输出，第二行为二元模型或三元模型的输出。

- 今天回家比较**晚**

  今天回家比较**完**

- 北京**是**一个美丽的城市

  北京**市**一个美丽的城市

- **她**是我的母亲

  他是我的母亲

- 有绝对的言论自由**吗**

  有绝对的言论自由**马**

- 编写这种文章的人往往**哗众取宠**

  编写这种文章的人往往**华中区重**

4.3 三元模型相比于二元模型有所改进的例子

第一行为二元模型的输出，第二行为三元模型的输出。

- 一**致**可爱的大黄狗

  一**只**可爱的大黄狗

- 拼音之间用**控各个开**

  拼音之间用**空格隔开**

- 他养了一**致青瓦**当宠物

  他养了一**只青蛙**当宠物

- 玩游戏让我**赶**到快乐

  玩游戏让我**感**到快乐

- 这两个人站在上面已经三天**散射**了

  这两个人站在上面已经三天**三夜**了

- 从现在开始不要太过**预防死**

  从现在开始不要太过**于放肆**

- 你不是我的**新商人**

  你不是我的**心上人**

- 中国和意大**力**都是历史悠久的文明古国

  中国和意大**利**都是历史悠久的文明古国

- 你在故意**无到**别人

  你在故意**误导**别人

- 机器学习**机器**应用

  机器学习**及其**应用

- 每**个**四年一次的奥运会就要召开了

  每**隔**四年一次的奥运会就要召开了

- 科比**以外**去世

  科比**意外**去世

- 清华大学优秀**小游**

  清华大学优秀**校友**

- 我从未见过**犹**如此**后眼**无耻之人

  我从未见过**有**如此**厚颜**无耻之人

4.4 三元模型相比于二元模型有所退步的例子

第一行为二元模型的输出，第二行为三元模型的输出。

- 特朗普希望不久和中国国家主席面对面**会晤**

  特朗普希望不久和中国国家主席面对面**挥舞**

- 你的看法**有**偏见

  你的看法**由**偏见

- 毕竟我是一个不擅长表大的**人**

  毕竟我是一个不擅长表大的**任**

4.5 分析

可以看到，应用了三元模型后，无论是逐字正确率还是整句正确率均有较大改进。可以看到，由于三元模型考虑了更多的上下文，“一只青蛙”、“空格隔开”、“三天三夜”、“过于放肆”、“心上人”、“厚颜无耻”这些较长的短语的正确率有所提升，相比于退步的例子是利大于弊。

总的来说，模型在特定类型的复杂句子上表现较好，如“全国人民代表大会在北京人民大会堂隆重召开”、“毛泽东思想和中国特色社会主义理论体系概论”、“对亚洲人的种族歧视已经在全球蔓延”，由于学习的语料是新闻，因此这些政治话语准确率较高。出乎我的意料的是，“金庸”、“吹啊吹啊我的骄傲放纵”、“床前明月光”这样带有文化背景的例子也能够识别正确，这应该与语料的覆盖有关。

对于不好的例子，如“比较晚”，句尾的语气词“吗”，“哗众取宠”，主要的原因应该是在语料中出现较少，或有“比较完善”这样的常见短语拉高了“比较完”的概率，且由于采取的是字的模型，没有进行分词处理。而“他是我的母亲”这样的错误则存在于语义上，很难做出判别。

5 总结

通过对典型实例的分析，有以下几点结论：

- 三元模型相较于二元模型，准确率有所上升
- 可以尝试增大训练集规模，且除新闻外使用更多类别的语料
- 可以尝试对语料分词，在此基础上使用基于词的模型
- 若希望改善语义上的合理性，需要对句法和词性进行建模和分析