COMS0031132095 Natural Language Processing - HW 1

李鹏，*10175501102@stu.ecnu.edu.cn*

October 30, 2019

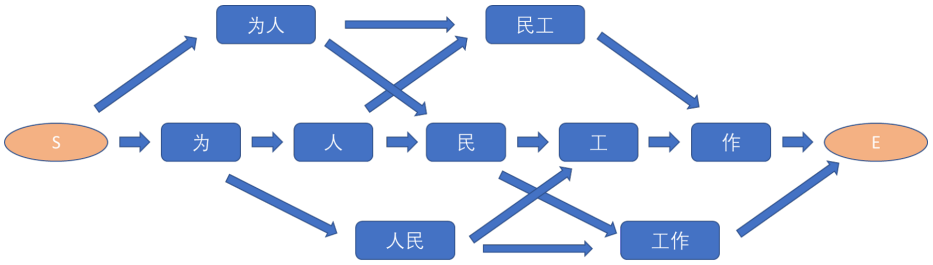
题目一

有如下词典，每个词的费用为 $C = -logP, P$ 为该词在语料库中的使用频率。对“**为人民工作**”进行分词。
(1) 分别给出正向和逆向最大匹配法的分词结果；
(2) 画出词图，用最短路径法分词；
(3) 用最大概率法分词，给出详细计算过程。

词	费用
为人	4.2
人民	2.8
民工	3.2
工作	2.5
为	3.6
人	3.4
民	4.5
工	4.0
作	4.8

答：
(1) 设定最大词长为 5，
正向最大匹配法分词结果：“**为人/民工/作**”
逆向最大匹配法分词结果：“**为/人民/工作**”

(2)
词图如下，用最短路径法分词结果：“**为/人民/工作**”、“**为人/民工/作**”、“**为人/民/工作**”（均含有 3 个词）



(3) 利用动态规划算法进行最大概率法分词 (标 * 的为最优路径)：

序号	候选词	费用	累计费用	最佳左邻
0	为	3.6	3.6*	-1
1	为人	4.2	4.2	-1
2	人	3.4	7.0	0
3	人民	2.8	6.4*	0
4	民	4.5	8.7	1
5	民工	3.2	7.4	1
6	工	4.0	10.4	3
7	工作	2.5	8.9*	3
8	作	4.8	12.2	5

分词结果：**“为/人民/工作”**

题目二

一个分词系统分词得到如下结果：
据/ 中央/ 气象台/ 预报/，未来/三天/，四川/盆/地/等/地/将有/一次/降/雨天/气/过程/。
假设正确分词结果如下：
据/ 中央气象台/ 预报/，未来/三/天/，四川盆地/等/地/将有/一/次/降雨/天气/过程/。
计算此次分词的准确率、召回率及 F-Measure 值（标点符号不计入词数）。

答：
分词结果中正确的分词：“据”，“预报”，“，未来”，“等”，“地”，“将有”，“过程”，共 7 个。
结果中所有的分词数共 17 个。
标准答案中所有分词数共 15 个。

(1) 准确率：

$$Precision(P) = \frac{\text{分词结果中正确分词数}}{\text{结果中所有分词数}} \times 100\% = \frac{7}{17} \times 100\% \approx 41.17\%$$

(2) 召回率：

$$Recall(R) = \frac{\text{分词结果中正确分词数}}{\text{标准答案中所有分词数}} \times 100\% = \frac{7}{15} \approx 46.67\%$$

(3)F-measure 值：

$$F - measure = \frac{2PR}{P+R} = \frac{2 \times \frac{7}{17} \times \frac{7}{15}}{\frac{7}{17} + \frac{7}{15}} = 43.75\%$$

注：在判断分词正确与否的时候，要注意这里“四川盆地”的“地”与“等地”中的“地”是不是要区分考虑。

题目三

用 HMM 和 Viterbi 算法进行词性标注

人民	收入	和	生活	水平	进一步	提高
n	n	c/p/v	n/v	n/a	n/d	n/v

词性转移表

Tag	n	c	p	v	a	d
n	80000	10000	12000	80000	5000	10000
c	30000	1000	2000	20000	10000	5000
p	40000	1000	500	5000	5000	10000
v	50000	5000	5000	20000	4000	10000
a	30000	8000	4000	7000	1000	20000
d	20000	10000	6000	30000	5000	9000

词语频度表

词语	词性	频次	词语	词性	频次
人民	n	5000	水平	n	4000
收入	n	4000	水平	a	1000
和	c	2000	进一步	n	1000
和	p	1000	进一步	d	2000
和	v	200	提高	n	1000
生活	n	5000	提高	v	4000
生活	v	2000			

词性频度表

词性	频次
n	200000
c	100000
p	100000
v	200000
a	100000
d	100000

答：
本部分内容由于笔算过于繁琐，自己动手用 python 实现了这两个算法，过程中遇到并解决了浮点数下溢、笛卡尔积计算以及 Viterbi 算法的实现等困难。代码参见附件，最终结果如下：

人民	收入	和	生活	水平	进一步	提高
n	n	p	v	a	n	n

其中传统的 HMM 算法用时 0.6659998893737793s, 而基于 Viterbi 算法的 HMM 用时 0.08899998664855957s，可见 Viterbi 算法能够显著提升算法效率。