**NLP第一次作业**

姓名：贾旺旺 学号：20204251

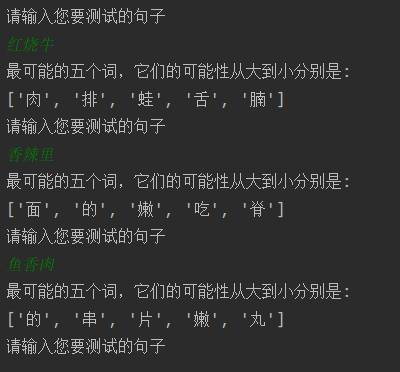
1. 代码详解

（1）构建数据库部分，并将最终数据库保存为一个npy文件

1. #构建数据库
2. doubleWords = {}
3. **def** train02(filename):
4. doubleWords\_pre={}
5. f=open(filename,encoding='utf-8')
6. #一共有多少个句子
7. num=0
8. **for** line **in** f.readlines():
9. # 匹配所有中文形成一个list包,按符号分割出所有的字符
10. line1 = re.findall('[\u4e00-\u9fa5]+', line)
11. **for** words **in** line1:
12. #存前一个字符
13. words1='B'+words+'E'
14. pre=' '
15. #计算一个字的频数和两个字连续出现的频数
16. **for** word **in** words1:
17. **if** pre != ' ':
18. **if** pre **in** doubleWords\_pre:
19. **if** word **in** doubleWords\_pre[pre]:
20. doubleWords\_pre[pre][word]+=1
21. **else**:
22. doubleWords\_pre[pre][word]=1
23. **else**:
24. doubleWords\_pre[pre]={}
25. pre=word
26. num+=1
27. f.close()
28. # 保存初始概率文件
29. np.save("data/doubleWords\_pre", doubleWords\_pre)

（2）采用二元文法+加1法平滑处理方法

1. #采用平滑处理方法+二元文法
2. **def** fun1(worda ,wordb):
3. **if** worda **in** doubleWords:
4. **if** wordb **in** doubleWords[worda]:
5. sum=0
6. **else**:
7. doubleWords[worda][wordb]=0
8. **else**:
9. doubleWords[worda]={}
10. doubleWords[worda][wordb]=0
11. fenzi=doubleWords[worda][wordb]+1
12. fenmu=0
13. **for** key **in** doubleWords[worda]:
14. fenmu=fenmu+doubleWords[worda][key]+1
15. doubleWords[worda][wordb]+=1
16. **return** (fenzi/fenmu)
18. #采用二元文法计算概率,需要自己加'B'和'E':
19. **def** fun(word):
20. sum=1.0
21. **for** i **in** range(len(word)-1) :
22. sum=sum\*fun1(word[i],word[i+1])
23. **return** sum
24. 预测
25. #预测结果
26. **def** predict(word):
27. preRes={}
28. str=word[len(word)-1]
29. s = sum(doubleWords[str].values())
30. **for** key **in** doubleWords[str]:
31. **if** key=='E':**continue**
32. str1=str+key
33. preRes[key]=fun(str1)
34. **return** preRes
36. #将预测结果排序，选取前五个
37. **def** Paixu(word):
38. Result=[]
39. preRes=predict(word)
40. preRes = sorted(preRes.items(), key=**lambda** x: x[1], reverse=True)
41. **for** i **in** range(5):
42. Result.append(preRes[i][0])
43. **return** Result
44. 主函数
45. #二元文法中后一个字出现的概率仅仅取决于前一个字
46. **if** \_\_name\_\_ == '\_\_main\_\_':
47. #train02('data/sentiment\_analysis\_trainingset.txt')
48. doubleWords=np.load('data/doubleWords\_pre.npy',allow\_pickle='TRUE').item()
49. **while**(True):
50. word = input("请输入您要测试的句子\n")
51. **if** word=='q':
52. **print**("结束")
53. result = Paixu(word)
54. **print**("最可能的五个词，它们的可能性从大到小分别是:")
55. **print**(result)
56. 代码运行（选取的数据集为美团评论）：



个人观点，二元文法存在较大的局限性，它只能用来基于前一个词来预测，效果有些不佳。

在本个模型中，我觉得最有意义的事把数据集存成一个文件，这在接下来的每一次训练中，我们都不用重复构建数据集。