自然语言处理实验报告

Natural Language Processing Experimental report

基于 N-Gram 的中文分词算法及文本概率分析

——Python 实现

姓　名：南佳霖

班　级：19 计算机一班

学　号：19011402

指导教师：孙媛

学　院：信息工程学院

2021 年 11 月 11 日

摘 要

在统计语言模型中，自然语言看作一个随机过程，其中每一个统计基元如字、词、句子、段落都可看作是有一定概率分布的随机变量。根据贝叶斯公式可计算文本的概率：$\mathrm{P(s)} = \mathrm{P}(w_1) \times \mathrm{P}(w_2|w_1) \times \ldots \times \mathrm{P}(w_m|w_1 \ldots w_{m-1}) = \prod_{i=1}^{m} w_i|w_1 \ldots w_{i-1}$；并以此为基础构建 n 元文法模型，当 n=2 时被称为二元文法。可应用于音字转换、汉语分词、文档分类等多种场景。本次实验针对中文分词消歧问题与文本概率计算进行了代码实现。

关键词： 自然语言处理；n 元文法；二元文法；中文分词；

Abstract

Natural language is regarded as a random process in the view of statistics, for each statistical substrate such as word, sentence or paragraph can be a random variable with a certain probability distribution. The probability of sentences can be calculated with the Bayesian formula,$\mathrm{P(s)} = \mathrm{P}(w_1) \times \mathrm{P}(w_2|w_1) \times \mathrm{P}(w_3|w_1w_2w_3) \ldots \times \mathrm{P}(w_m|w_1 \ldots w_{m-1}) = \prod_{i=1}^{m} w_i|w_1 \ldots w_{i-1}$ which builds N-gram model, on which it is called Bi-gram when n=2. It can be applied to many scenes such as phonetic word conversion, Chinese word segmentation, document classification and so on. This lab is to code to save the problem of Chinese participle disambiguation and text probability calculation in python.

Key Words: NLP; N-gram; Bi-gram; Chinese word segmentation;

目录

引言

0.1 研究背景

随着互联网技术的快速发展，谷歌、必应、百度等通用搜索引擎成为互联网世界的重要组成部分，其核心技术是基于自然语言处理的全文检索技术。在中文的句子中，词与词之间不使用分隔符或空格，这使得计算机对于词的准确识别变得特别困难，想建立关键词的索引必须先使用分词技术进行句子中词语的切分。

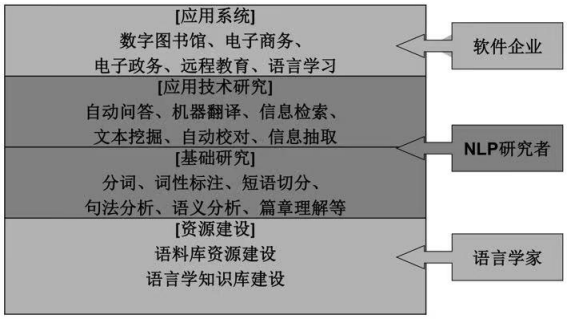


图 0-1 NLP 的不同层次

所以，中文分词是语言信息处理的基础性工作，是语言机器翻译、智能检索、自然语言理解与处理等智能信息应用的前提。分词系统的好坏将直接影响以此为基础的智能处理系统性能的优劣。

分词算法主要有以下三类：

A. 基于规则词典的中文分词法，如：正向最大匹配算法 FMM、逆向最大匹配算法 BMM

B. 基于统计的中文分词法，如：n 元文法模型、隐马尔可夫模型、最大熵模型等。

C. 基于理解的中文分词法。

本次实验基于 n 元文法模型(N-gram)对文本分词，并结合第二次实验结果（FMM、BMM），对比两类分词算法的性能。

0.2 实验目标

A. 根据实验二中正向、逆向最大匹配结果，对于歧义问题采用 2 元文法进行消歧。针对消歧后的文本，再次计算 P、R、F 值。

B. 利用 2 元文法计算测试语料中每段话的概率

0.3 作业目录

NLP_Lab3_19011402_南佳霖.pdf	-	实验报告
bi_gram.py	-	概率分析
HMM.py	-	分词
Evaluate.py	-	评价函数
jiebaCut.py	-	结巴分词
test.txt	-	测试语料

一、 实验原理

1.1 N-gram

n 元文法可以预测一个单词序列出现的概率。n 元文法假设一个单词出现的概率分布只与这个单词前面的 n－1 个单词有关，与更早出现的单词无关。

N-gram 模型的实质是 N-1 阶的隐马尔科夫模型。

$$\mathrm{P}(w_i|h_i)$$
$$= \mathrm{P}(w_i|w_{i-n+1}, w_{i-n+2} \cdots, w_{i-1},)$$

N-gram 模型有两个主要问题：

- 由于设备的存储空间有限，限制了 n 值。N 值过小则不能准确区分长词。
- 自然语言遵循 Zipf 定律，面临数据稀疏问题，一

些小概率语言现象可能算法崩溃

- 注：Zipf 定律指，语料库里，一个单词出现的频率与它在频率表里的排名成反比。所以，频率最高的单词出现的频率大约是出现频率第二位的单词的 2 倍，而出现频率第二位的单词则是出现频率第四位的单词的 2 倍。

1.2 基于词网格的 Bi-gram 分词

Bi-gram 模型就是一阶马尔可夫模型，在计算路径评价值时，要假设当前词的概率只受前一个词影响。即$P(S)=\prod_{i=1}^{n}P(w_i|w_{i-1})$，若每个概率都很小，亦可对结果取对数，以便于计算机运算。

为了保证条件概率在 i=1 时有意义，同时为了保证句子内所有字符串的概率和为 1，可以在句子首尾两端增加两个标志<BOS>和<EOS>。

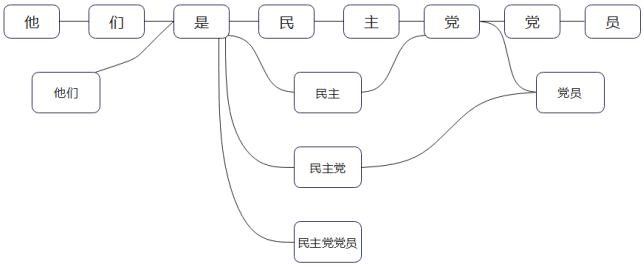


图 1-1 词网格举例

如有以上词网，存在以下 14 种候选分词路径，需依次计算概率得出 argmaxP(S)，确定最合理分词路径。

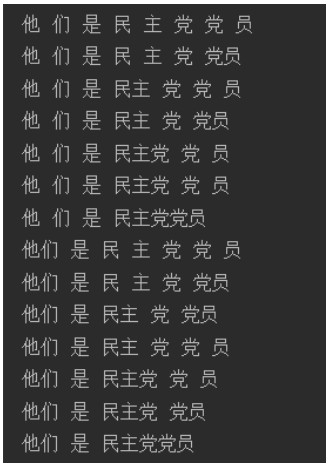


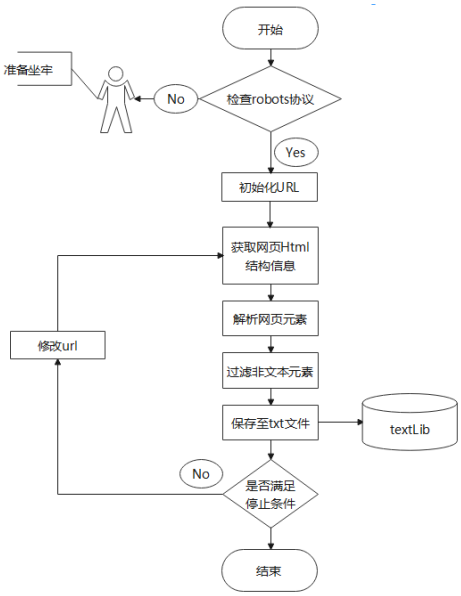
图 1-2 分词路径

二、 建立语料库

获取中文语料数据、建立语料库是进行分词的基础和前提。而通过手动录入或复制粘贴的方法效率极低且质量低下。网络爬虫技术可以在互联网中自动爬取语料数据，基于此技术可以迅速高效地建立语料库。

此实验中，采用 txt 文件的形式存储生语料

2.1 爬虫工作流程



2-1 爬虫工作流程

2.2 爬取对象

此次实验选用中文网站 AMiner 为爬取对象以建立中文语料库。相对于“AI 报告”、“会议论文”等板块,“趋势分析”板块中文章内容英文及术语简写较少，更适合建立中文语料库，所以选择爬取“趋势分析”板块内文章。

图 2-1 AMiner 网站

Aminer 是一个科技情报大数据挖掘与服务系统平台，由清华大学计算机科学与技术系教授唐杰率领团队建立，平台提供信息工程和 AI 领域的最新资讯。

2.3 文本预处理

图 2-3 预处理流程

- **文本合并**
  a) 引入 glob 库，对语料库\内文件进行遍历
  b) 新建 data 存放所有数据
  c) 使用 readlines( )读取所有行
  d) 将 data 写入文件 all.txt
- **去除标点**

由于爬取的预料并不是纯中文，其中夹杂着大量的标点符号西洋字符，在实现字符匹配算法时要考虑如何处理这些意外符号。

第一种策略是：将符号也作为字进行切分。然而，对于符号的切分并不能体现匹配算法的准确的，在进行评价时，由于大量符号的“正确“切分，会导致算法评价高于实际的情况。

第二种策略是：采用算法去除标点符号，过滤出纯文本或仅带有少量标点的文本。本次实验采用此策略。去除标点过程如下：

a) 输入需要处理的字符串
b) 将想去除的标点建为一个列表
c) 遍历字符串，若字符不在符号列表中，就将其添加到新列表中
d) 遍历完成后，将新列表保存至 txt 文件

图 2-4 去除标点后的语料库 all_train_cleaned.txt

- **断句换行**

  断句换行后便于代码编写，逐行处理。

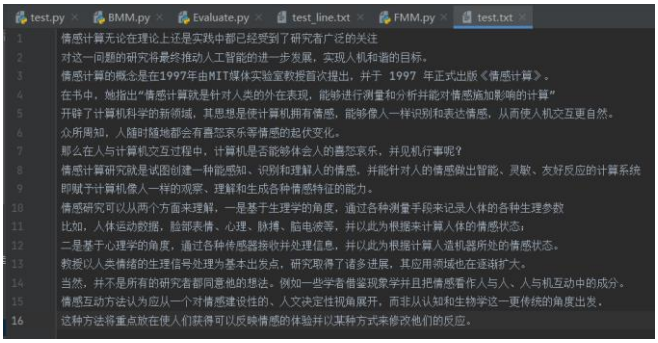


图 2-5 断句换行后的文本

三、 算法实现

3.1 概率测试 bi_gram.py

- 利用正则表达式，匹配子串，添加守首尾标签

```
# 添加起始符BOS和终止符EOS
s_modify1 = re.sub(r"[%s]+" % punctuation, "EOSBOS", s)
s_modify2 = "BOS" + s_modify1 + "EOS"
```

- 遍历语料字符串，统计词频，如果测试的第一个词和语料的第一个词相等则比较第二个词

```
count_list=[0]*(len(test_list)-1)
#遍历测试的字符串
for i in range(0, len(test_list)-1):
    for j in range(0,len(ori_list)-2):
        if test_list[i]==ori_list[j]:
            if test_list[i+1]==ori_list[j+1]:
                count_list[i]+=1
return count_list
```

- 计算概率，采取加一平滑策略

```
def Probability(test_list,count_list,ori_dict):
    flag=0
    #概率值为p
    p=1
    for i in range(len(test_list) - 1):
        p *= ((float(count_list[flag] + 1)) / (float(ori_dict[test_list[i]]) + 1))
        flag += 1
    return p
```

- 存入词频表，输出结果并取对数保存结果

```
count_list = CompareList(ori_list, test_list)
p=Probability(test_list, count_list, ori_dict)
#p=round(p,4)
print(p)
p_list.append(math.log(p))
```

- 结果输出

```
bi_gram
7.195020094397983e-41
7.896210699341181e-19
7.92949583191893e-28
3.8187153713480 46e-30
1.986983204989639e-27
6.618003548105908e-23
4.905824299880359e-34
8.108085206275259e-28
7.227095392732918e-20
[-39.06572353381909, -50.73797972400269, -67.577
 .68273378079798, -62.40179314751926, -67.737638
```

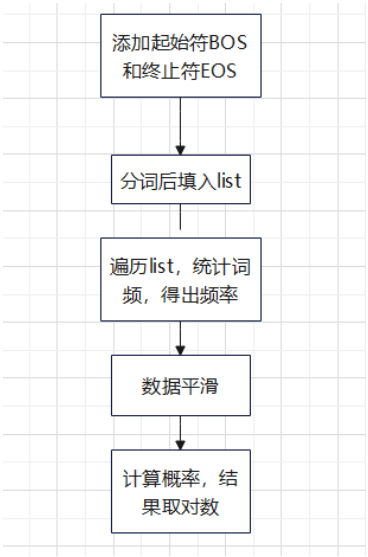


图 3-1 流程图

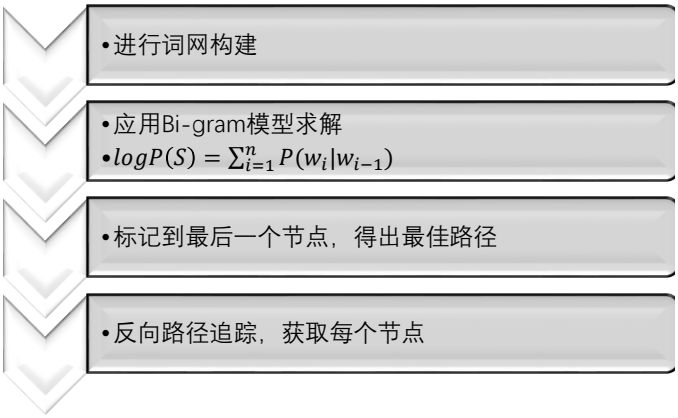
3.2 分词消歧 HMM.py

训练语料：人民日报 1998.txt

测试语料：《情感计算》test.txt

输出结果：text_out_2gram.txt

➢ 维特比算法



➢ 加一平滑方法

```
self.B_dic = {k: {k1: (v1 + 1) / Count_dic[k] for k1, v1 in v.items()}
              for k, v in self.B_dic.items()}
```

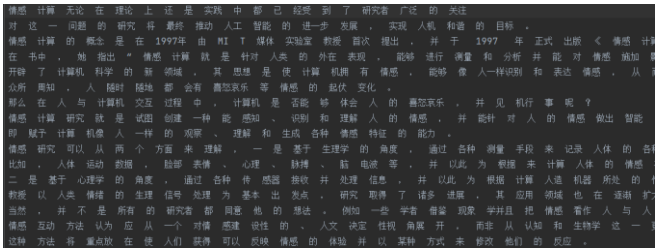


图 3-2 分词结果

四、 实验结果

针对《情感计算》文本，三种分词方法的性能指标如下：

```
生成结果词的个数： 427
jieba分词文本词的个数： 382
正确词的个数： 332
正确率： 0.7775175644028103
召回率： 0.8691099476439791
f值： 0.8207663782447465
```

图 5-1 正向最大匹配算法

```
逆向最大匹配算法BMM:
生成结果词的个数： 427
jieba分词文本词的个数： 382
正确词的个数： 329
正确率： 0.7704918032786885
召回率： 0.8612565445026178
f值： 0.8133498145859085
```

图 5-2 逆向最大匹配算法

```
生成结果词的个数： 419
jieba分词文本词的个数： 382
正确词的个数： 307
正确率： 0.7326968973747017
召回率： 0.8036649214659686
f值： 0.7665418227215981
```

图 5-3N 元文法

➢ 相比于 uni-gram 模型，Bi-gram 模型描述了更丰富的语言信息，具有更好的性能；但数据稀疏问题相对 uni-gram 模型更为严重，需要进行数据平滑。

五、 总结与分析

➢ 分词

实验中发现，对于小规模语料的分词，F_FMM > F_BMM > F_N-gram，使用了 N 元文法反而效果更差。我不理解。

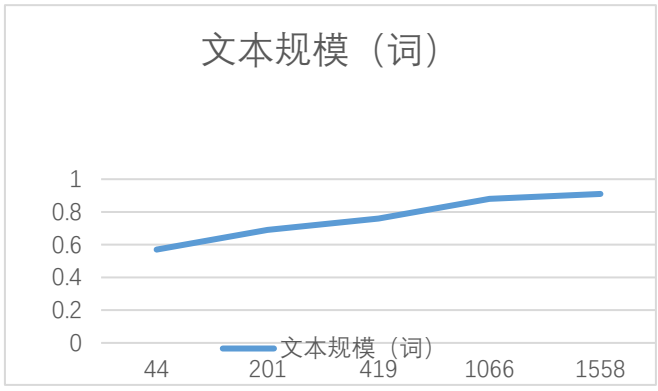


图 5-1 结果分析

由于测试样本仅有 419 词，于是更换不同规模的测试样本，发现 N-gram 的小鬼随文本规模增大而提高。在语料规模超过 1500 词左右时，效果已经超过 FMM 与 BMM 算法。

➢ 概率测试

“你算他的概率，这东西也没啥用啊，这没有意义”

——宗成庆《从 N 元文法到神经语言模型》

通过这次实验，我的心态再一次被击垮。一个人做 lab 除了孤独弱小无助，更多的是听到其他组在讨论分工和报告排版的时候，我陷入深深的内卷焦虑。

但是，一个人做实验确实能学到不少东西，对 n-gram 模型的理解非常透彻，提高了我的抗压能力和问题分析能力，感谢党和人民的考验，我会继续努力

六、 参考文献

[1] 信息检索中的中文分词问题研究[J]. 情报杂志，2008(7)
[2] 姜维. 文本分析与文本挖掘[M]. 科学出版社，2018：10-20
[3] 自然语言处理之维特比算法实现中文分词[EB/OL].