北京邮电大学　计算机学院

《自然语言处理》实验报告

姓名　　王睿嘉

学号 2015211906

班级 2015211307

Part-of-speech Tagging

一、 实验内容和环境描述

1. 实验内容

Part-of-speech Tagging: 30 points

-This data set contains one month of Chinese daily which are segmented and POS tagged under Peking Univ. standard.

-Design a sequence learning method to predicate a POS tags for each word in sentences.

-Use 80% data for model training and other 20% for testing (or 5-fold cross validation to test learner's performance. So it could be interesting to separate dataset.)

2. 环境描述

编程语言：python
集成开发环境：PyCharm
解释器版本：2.7.13

二、 输入输出定义

1. 输入定义

用户无需自行输入数据。

本实验以压缩包所含文件“Chinese text corpus.txt”为语料库，通过一定划分产生训练样本和测试文本。

2. 输出定义

控制台分别输出 5 层-交叉检验（5-fold cross validation）方式下，利用隐马尔可夫模型（HMM）和 Viterbi 算法标注测试语料的正确率。

三、 实验设计

1. 整体说明

由于（词条，词频）的统计形式极为符合字典数据结构，且 re 库中的 split 函数便于语料库词条划分，因而采用 python 作为该实验的编程语言。

本实验使用教师所提供的“Chinese text corpus.txt”语料库，并均分为 5 份，采用 5 层-交叉检验的方式，即 4 部分用于训练语言模型，剩下部分作为测试文本。

隐马尔可夫模型可用来解决 3 个问题：

① 估算问题。即给定观测序列和模型，计算产生某个观测序列的概率（观测的似然）；

② 解码问题。即给定观测序列和模型，计算能最好解释观测的状态序列；

③ 参数学习。即给定足够的观测数据，计算相应的模型参数。

而就词性标注而言，主要关注解码问题和参数学习。

对于解码问题，需要利用 Viterbi 算法；而对于参数学习，有两种主要方案：有监督学习和无监督学习。前者基于已知“正确答案”的数据进行估计，即通过统计标注训练集获得相应参数。后者基于无“正确答案”的数据（不完全数据）进行估计，可以通过类似于爬山算法的局部最大化方法求解，即 Baum-Welch 算法或前向-后向（Forward-Backward）算法。本实验利用提供的标注训练集“Chinese text corpus.txt”进行有监督的参数学习。

实验流程如下：

a. 对原始语料进行预处理，删去无效信息并断句。同时，将其均分 5 份，分别用于训练语言模型；将 5 份训练集的标注词性删除并输出，用作测试文本；

b. 采用 5 层-交叉检验方式对隐马尔可夫模型进行训练，并将 Viterbi 算法和基于规则（Ruled-Based）的词性标注方法结合，对测试文本进行词性标注；

c. 将标注结果与相应训练集进行比对，统计正确率并输出。

2. 关键点说明

2.1 断句

原语料库中并非以换行标志句子结束。显然，若以段作为词性标注单位，正确率将受到极大影响。因而在预处理中按照句号、问号和叹号进行换行，即断句。

2.2 参数学习

隐马尔可夫模型需要的参数有三个，即（π，A，B）。其中 π 为初始状态的概率向量，表示各词性作为初始状态的概率；A 为状态转移矩阵，每个元素 a_{ij} 表示从词性 i 到词性 j 的转移概率；B 为发射概率矩阵，每个元素 $b_i(o_t)$ 表示观测的似然，即由词性 i 产生单词 o_t 的概率。

对于参数 π，通过统计训练语料库中各词性作为初始状态的频次，再除以语句总数得到。

对于状态转移矩阵 A，可以按照以下公式计算：

$$\mathrm{P}(W_t|W_{t-1}) = \frac{P(W_{t-1},W_t)}{P(W_{t-1})} \approx \frac{\#(W_{t-1},W_t)}{\#(W_{t-1})}$$

其中，$\mathrm{P}(W_t|W_{t-1})$ 表示已知词性W_{t-1} 出现的情况下，词性W_t 紧接着出现的概率；$\#(W_{t-1},W_t)$ 表示词性W_{t-1} 、W_t 相邻出现的频次，$\#(W_{t-1})$ 表示词性W_{t-1} 出现的频次。

对于发射概率矩阵 B，可以按照以下公式计算：

$$\mathrm{P}(O_t|W_t) = \frac{P(O_t,W_t)}{P(W_t)} \approx \frac{\#(O_t,W_t)}{\#(W_t)}$$

其中，$\mathrm{P}(O_t|W_t)$ 表示已知词性W_t 的情况下，产生词O_t 的概率；$\#(O_t,W_t)$ 表示词性为W_t 的词O_t 出现的频次，$\#(W_t)$ 表示词性W_t 出现的频次。

假设通过训练语料库获得了 N 个词性，M 个词，则 π 是一个长度为 N 的向量，A 是一个 N*N 矩阵，B 是一个 N*M 矩阵。

2.3 Viterbi 算法

本实验利用 Viterbi 算法求解能最好解释句子的词性序列。

Viterbi 算法实际是一种动态规划的方法，因而需解决两个关键问题：初始化和递归方程。其思想为：若已经得知能最好解释前 t 个词的词性序列，在求解第 t+1 个词的词性时，需要从 N 个词性中选择一个，使得这 t+1 个词性序列能最好解释前 t+1 个词。

初始化公式如下：

$$v_1(j) = a_{0j}b_j(o_1), 1 \le j \le N$$

递归方程如下：

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)a_{ij}b_j(o_t), 1 \le j \le N, 1 < t \le T$$

在实际编程中，Viterbi 算法作为一个函数，在标注过程中进行调用，共有 5 个参数：待标注语句、词性集合、初始状态概率向量、状态转移矩阵和发射概率矩阵，并返回一个二元组（概率，结果词性序列）。

2.4 基于规则的标注方法

本实验利用的基于规则的词性标注方法，主要针对标点符号。由于标点符号词性确定，不存在歧义，因而在标注后，进行标点符号的判别，若是，则直接修改为确定值，从而提高标注的正确率。

2.5 未出现词

在词性标注过程中，需要对未出现词进行处理。本实验采取直接赋值、总频次倒数相结合的平滑方法。

2.6 概率计算

本实验语料库有一定的规模，为了避免较小概率连续相乘损失精度或直接造成下溢，对出现概率取对数，再相加，从而得到整体概率。公式如下：

$$\mathrm{T} = -\log_{10}(P_1 P_2 \ldots P_n)$$

2.7 性能评估

本实验采用 5 层-交叉检验的方式进行模型的性能评估：将“Chinese text corpus.txt”语料库均分为 5 份，进行 5 次实验。每次实验都从 5 部分中选择一部分组作为测试文本，剩余 4 部分作为训练语料集。

四、 文件说明

共有两个源码文件：

Pre-Processing.py：对语料库进行预处理，删去无效信息并断句。同时，将其均分 5 份，分别用于训练语言模型；将 5 份训练集的标注词性删除并输出，用作测试文本。

POS Tagging.py：建立隐马尔可夫模型，并进行测试。在标注过程中，会输出相应的初始状态概率向量、状态转移矩阵、发射概率矩阵文件和最终的标注结果文件。

共有 20 个文本文件：

Chinese text corpus.txt：教师所提供的语料库。

partitioned corpus.txt：预处理后的语料库。

first corpus.txt：第一部分语料库。

second corpus.txt：第二部分语料库。

third corpus.txt：第三部分语料库。

fourth corpus.txt：第四部分语料库。

fifth corpus.txt：第五部分语料库。

first test.txt：第一部分测试文本。

second test.txt：第二部分测试文本。

third test.txt：第三部分测试文本。

fourth test.txt：第四部分测试文本。

fifth test.txt：第五部分测试文本。

first result.txt：第一部分标注结果。

second result.txt：第二部分标注结果。

third result.txt：第三部分标注结果。

fourth result.txt：第四部分标注结果。

fifth result.txt：第五部分标注结果。

Pi.txt：初始状态概率向量。

A.txt：状态转移矩阵。

B.txt：发射概率矩阵。

五、 实验结果

如下表：

实验序号	词总数	正确标注个数	正确率
1	262095	216535	82.62%
2	265439	217517	81.95%
3	269591	222698	82.61%
4	274110	217766	79.44%
5	268114	208080	77.61%
平均	267869.8	216519.2	80.83%

六、实验心得

在本次实验中，遇到的主要问题有以下三点：

a) Viterbi 算法的实现。讲义中有算法的伪代码，但落实到 python，仍让我摸不着头脑。经 Baidu，理解并参考部分代码，该问题得到解决；

b) 代码运行效率。此次实验采用五层交叉检验方式，每层检验时间均较长。第一次运行，20 多分钟，仍未显示结果，以为程序进入了死循环。检查代码并无错误，再次运行，40 多分钟后才得结果。曾多次阅读代码，意图找到提高运行效率的方法，无果；

c) 模型性能。本次实验所构建的 HMM 模型性能较差，标注正确率仅在百分之八十左右，百思不得其解。通过 print 打印出标注错误的词与词性，发现绝大多数为 vn，即动名词。尝试强制将未出现词标注为 vn，正确率并未增加，反而减少。又尝试更改平滑方法，仍然增幅甚微。因程序执行时间较长，更改后再次检验，耗时极大，但可悲的是，仍一无所获。

本人对此次实验的结果极不满意，可能对模型的理解存在些许偏差，若有可能，希望得到老师指点。