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

N-gram Language Models

一、 实验内容和环境描述

1. 实验内容

N-gram Language Models: 20 points

-In this assignment you will explore a simple, typical N-gram language model.

-This model can be trained and tested on sentence-segmented data of a Chinese text corpus. "Word Perplexity" is the most widely-used evaluation metric for language models.

-Additional points: if you can test how does the different "Word Perplexity" of the different "N" grams, you will get additional 10 points

-Additional points: if you can test how does the different "Word Perplexity" of the different smoothing methods, you will get additional 10 points

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 unigram、bigram 和 trigram 模型下，采用 Add-one 及 Good-turing 平滑方法所计算出的测试语料困惑度，与此同时，输出适当提示语句。

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 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，前 90%的语料用于训练语言模型，剩余部分作为测试文本。

基本实现了 unigram、bigram 和 trigram 三种语言模型。由于存在数据稀疏问题（即受采样样本总数的限制，极大似然估计不可靠，或由于未知词和未知词组的存在，导致频率无法计算），对他们分别采用 Add-one 和 Good-turing 平滑方法，并计算出各自的困惑度。

2. 重点说明

2.1 训练词典

在预处理过程中，对训练语料库进行词频统计，并将结果输出至 unigram、bigram 和 trigram 词典。

Unigram 词典格式：

词　　词频

举例如下：

勇敢　18
勇斗　1
勇方　1
勇气　21
勇猛　1
勇辉　1
勉　　1
勉为其难　　2

Bigram 词典格式：

词/已出现词　　词频

举例如下：

社会/近年　1
社会/这个　1
社会/这种　1
社会/进行　1
社会/适应　2
社会/造福　1
社会/遵守　1
社会/配置　1
社会/重大　1

Trigram 词典格式：

词/已出现词/更早出现词　　词频

举例如下：

歌剧/祝国/发展　　1
歌剧/祝国/繁荣　　1
歌剧/提高/还是　　1
歌剧/水平/高　1
歌剧/的/下午　1
歌剧/的/当代　1
歌剧/的/演出　1
歌剧/相信/不　1
歌剧/著名/中外　　1

注：在输出 bigram 和 trigram 统计结果时，插入了句子开始标志“start”及结束标志“$”，并进行相应的词频统计。

2.2 Add-one 平滑方法

本实验对所有模型进行 Add-one 平滑处理。

Unigram 计算公式：

$$p_{Laplace}(w_i) = \frac{c_i + 1}{N + |V|}$$

其中 C_i 为词 W_i 在训练语料中的出现次数，N 为语料的 token 总数，|V| 为 unigram 词表大小。

Bigram 计算公式：

$$p_{Laplace} = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + |V|}$$

其中 $p(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$ 为已知词 W_{i-1} 出现的情况下词 W_i 出现的条件概率，|V| 为 bigram 词表大小。

trigram 计算方法与 bigram 同理。

2.3 Good-turing 平滑方法

本实验对所有模型进行 Good-turing 平滑处理。

Unigram 计算方法：

基本思路为根据仅出现一次的 unigram 个数，来确定那些未出现的 unigram 的概率。

仅出现一次的 unigram 个数为

$$N_1 = \sum_{w:count(w)=1} 1$$

出现 c 次的 unigram 个数为

$$N_c = \sum_{w:count(w)=c} 1$$

出现频次乘折扣可得仅出现一次的 unigram 概率

$$p_0 = \frac{c_0^*}{N} = \frac{N_1}{N_0 N}$$

出现 c 次的 unigram 概率

$$p_c = \frac{c^*}{N} = \frac{(c+1)N_{c+1}}{N_c N}$$

扩展至 N-gram 中词 W 出现概率的计算方法：

若其频次为 0

$$p(W) = \frac{1 - \sum_{r=1}^{\infty} n_r \frac{r^*}{N}}{n_0} \approx \frac{n_1}{n_0 N}$$

否则

$$p(W)=\frac{r^*}{N} \qquad r^*=\frac{(r+1)n_{r+1}}{n_r}$$

其中，n_r 表示出现 r 次的 N-gram 个数。

2.4 未出现词概率计算

在 bigram 和 trigram 模型中，对于未出现词的概率计算，采用回退和预留符号相结合的方法。以 trigram 的回退为例，套用以下公式：

$$p(w_i \mid w_{i-2},w_{i-1})=\begin{cases} p(w_i \mid w_{i-2},w_{i-1}) & if \quad C(w_{i-2},w_{i-1},w_i)>0 \\ \alpha_1 p(w_i \mid w_{i-1}) & if \quad C(w_{i-2},w_{i-1},w_i)=0 \; and \; C(w_{i-1},w_i)>0 \\ \alpha_2 p(w_i) & otherwise \end{cases}$$

在本实验中取 α_1=1，α_2=1。具体步骤如下：

读取 unigram 词典时，加入预留符号，即“<unknown>”，并记其频次为 0。紧接着，对 unigram 采用 Add-one 平滑。

读取 bigram 词典时，加入预留符号，即“<unknown>/已出现词”，并记其频次为 0。紧接着，对 bigram 采用 Add-one 平滑。

读取 trigram 词典时，加入预留符号，即“<unknown>/已出现词/更早出现词”，并记其频次为 0。紧接着，根据不同平滑方法，对 trigram 采用 Add-one 平滑或 Good-turing 平滑。

进行语言模型测试时，若 $C(W_{i-2},W_{i-1},W_i)=0$ 且 $C(W_{i-1},W_i)>0$，使用 trigram 预留符号的概率，否则若 $C(W_{i-1},W_i)=0$ 且 $C(W_i)>0$，回退，使用 bigram 预留符号的概率，若 $C(W_i)$也为 0，则回退至 unigram，使用 unigram 预留符号的概率。

2.5 困惑度

本实验利用以下公式进行困惑度的计算：

$$H(W)=-\frac{1}{N}\log_2 p(w_1...w_N)$$

$$PP(W)=2^{H(W)}$$

为了避免较小概率连续相乘损失精度或直接造成下溢，对每个词的出现概率取对数，再相加，从而得到测试集的整体概率。

四、 文件说明

共有 4 个源码文件

Pre-Processing.py：对语料库进行预处理，划分出训练语料和测试文本，同时生成 unigram、bigram 和 trigram 的词典，并输出至相应文本文件。

unigram(Add-one&Good-turing).py：实现了 unigram 模型，采用 Add-one 和 Good-turing 平滑，并分别输出对测试文本的困惑度。

bigram(Add-one&Good-turing).py：实现了 bigram 模型，采用 Add-one 和 Good-turing 平滑，并分别输出对测试文本的困惑度。

trigram(Add-one&Good-turing).py：实现了 trigram 模型，采用 Add-one 和 Good-turing 平滑，并分别输出对测试文本的困惑度。

共有 6 个文本文件：

Chinese text corpus.txt：教师所提供的语料库。

train.txt：划分后的训练语料。

test.txt：划分后的测试文本。

uniDict.txt：unigram 词典。

biDict.txt：bigram 词典。

triDict.txt：trigram 词典。

五、 实验结果

如下表，表中数据代表困惑度：

	Add-one	Good-turing
Unigram	2393.98	2095.28
Bigram	174.71	57.30
Trigram	33.78	0.44

对于同种平滑方法，从一元语言模型到三元语言模型，困惑度逐渐降低；对于同种语言模型，Good-turing 平滑方法困惑度小于 Add-one 平滑方法。

六、 实验心得

在本次实验中，遇到的主要难点如下：

① 曾进行过 python 语言的自主学习，但未有实战经验。而其在机器学习领域的优越性，使得它成为了此次实验编程语言的最佳选择，这本身就是难点之一；

② 起初对 Good-turing 算法并不是很理解，研究后发现算法并未考虑 Nr=0 的情况，经过了解 Katz 算法，通过将平滑与回退相结合，进行词条概率的计算；

③ 困惑度按照 PPT 上的方法进行计算，即计算一句话中各词语的几何平均概率后取倒数。但由于概率较小，造成了下溢。经过学长建议，采用先取对数再算 2 的幂得到困惑度。

实践出真知，本次实验是对课堂所学知识的补充。从初拿到题目的无从下手，到查阅各种资料逐步分析出算法，加深了对 N-gram 语言模型的理解，收获颇丰。