通过中文语料库来验证 Zipf's Law

Abstract:

本文利用已有的中文语料库，先对中文进行提取，再根据哑巴分词，统计汉字词语的频率并将其进行排序，绘制词语的排名与频率的关系图，使用对数坐标轴，当图中的数据点大致落在一条直线上，且符合对数关系，即可验证 Zipf's Law。

Introduction

Zipf's Law 是指为：在自然语言的语料库里，一个单词出现的频率与它在频率表里的排名成反比。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。即在给定语料中，对于任意一个单词，其频率（Frequency）与频率排序（Rank）乘积大致是一个常数，即：$Rank * Frequency \approx Constant$

Methodology

M1 ： 验证步骤

利用中文语料库验证 Zipf's Law，根据以下步骤：

1、将语料录入。

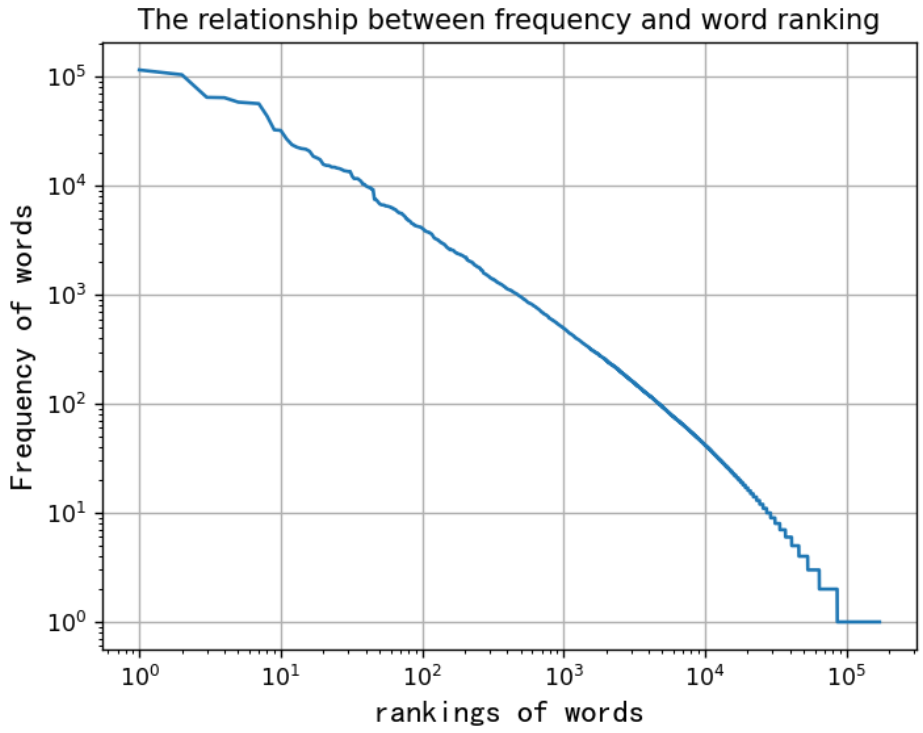
2、对语料库中的词语进行分词，并统计每个词语的出现频率。

3、将词语按照频率从高到低进行排序。

4、绘制词语的排名与频率的关系图，使用对数坐标轴，即横坐标为词语的排名（取对数），纵坐标为词语的频率（取对数）。

5、如果图中的数据点大致落在一条直线上，且符合对数关系，那么就验证了 Zipf's Law。

Experimental Studies



如图所示，频率和词语的排名对应的数值点大致落在一条直线上，即可验证 Zipf's Law。

计算中文的平均信息熵。

Abstract:

根据文献中提出的多元模型，利用该模型来计算中文(分别以词和字为单位)的平均信息熵，其中语料库为所提供的中文语料库。

Introduction

信息熵的概念最早由香农（1916-2001）于 1948 年借鉴热力学中的“热熵”的概念提出，旨在表示信息的不确定性。熵值越大，则信息的不确定程度越大。其数学公式可以表示为：

$$H(x) = \sum_{x \in X} P(x) log(\frac{1}{P(x)}) = -\sum_{x \in X} P(x) log(P(x))$$

但由于事物的实际概率往往是未知的需要估计，而所估计的概率与真实概率又存在差值，因此引入交叉熵的概念，交叉熵（Cross Entropy）是衡量两个概率分布之间差异的一种方法。公式可以表示为：

$$H(P, Q) = -E_{X \sim P}[logQ(x)] = -\sum_{i} P(x_i) logQ(x_i)$$

对于两个概率分布 P 和 Q，其中 P 为真实分布，Q 为预测分布。

本论文利用多元模型预测概率分布，进而求得信息熵。

Methodology

M1 ： 信息熵

信息熵是信息论中的一个重要概念，用于衡量一个随机变量的不确定性或者信息量的大小。信息熵最初由香农在他的《通信的数学理论》中提出，被广泛应用于通信、数据压缩、密码学、统计学等领域。

在信息论中，一个离散型随机变量的信息熵表示为：

$$H(x) = \sum_{x \in X} P(x) log(\frac{1}{P(x)}) = -\sum_{x \in X} P(x) log(P(x))$$

信息熵的单位通常是比特（bit），表示信息的量。信息熵越高，表示随机变量的不确定性越大，需要更多的信息来描述。

M2 ： 统计语言模型

假定 S 表示某一个有意义的句子，由一连串特定顺序排列的词 $w_1, w_2, \cdots, w_n$ 组成， n 为句子的长度。现在想知道 S 在文本中出现的可能性，即 $P(S)$ 。此时需要有个模型来估算，不妨把 $P(S)$ 展开表示为 $P(S) = P(w_1, w_2, \cdots, w_n)$ 。利用条件概率的公式， S 这个序列出现的概率等于每一个词出现的条件概率相乘，于是 $P(w_1, w_2, \cdots, w_n)$ 可展开为：

$$P(w_1, w_2, \cdots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)\cdots P(w_n|w_1, w_2, \cdots, w_{n-1})$$

其中 $P(w_1)$ 表示第一个词 w_1 出现的概率； $P(w_2|w_1)$ 是在已知第一个词为 w_1 的前提下，第二个词 w_2 出现的概率，后面以此类推。

显然，当句子长度过长时， $P(w_n|w_1, w_2, \cdots, w_{n-1})$ 的可能性太多，无法估算。假设该句子具有马尔科夫性，即任意一个词 w_i 出现的概率只同它前面的词 w_{i-1} 有关， S 的概率变为：

$$P(S) = P(w_1)P(w_2|w_1)P(w_3|w_2)\cdots P(w_n|w_{n-1})$$

其对应的统计语言模型为二元模型。也可以假设一个词由前面 $N-1$ 个词决定，即 N 元模型。当 $N = 1$ 时，每个词出现的概率与其他词无关，为一元模型， S 对应的概率变为：

$$P(S) = P(w_1)P(w_2)\cdots P(w_i)\cdots P(w_n)$$

M3 ： 信息熵计算

如果统计量足够大，字、词、二元词组或三元词组出现的概率大致等于其出现的频率。由此可得，字和词的信息熵计算公式为：

$$H(X) = -\sum_{x \in X} P(x)\log P(x)$$

其中， $P(x)$ 可近似等于每个字或词在语料库中出现的频率。

二元模型的信息熵计算公式为：

$$H(X|Y) = -\sum_{x \in X, y \in Y} P(x, y)\log P(x|y)$$

其中，联合概率 $P(x, y)$ 可近似等于每个二元词组在语料库中出现的频率，条件概率 $P(x|y)$ 可近似等于每个二元词组在语料库中出现的频数与以该二元词组的第一个词为词首的二元词组的频数的比值。

三元模型的信息熵计算公式为：

$$H(X|Y, Z) = -\sum_{x \in X, y \in Y, z \in Z} P(x, y, z)\log P(x|y, z)$$

其中，联合概率 $P(x, y, z)$ 可近似等于每个三元词组在语料库中出现的频率，条件概率 $P(x|y, z)$ 可近似等于每个三元词组在语料库中出现的频数与以该三元词组的前两个词为词首的三元词组的频数的比值。

M4 ： 实验步骤

1、 将文件夹里的语料枚举输入

2、 分别读取 txt 文件，并进行预处理，删除文章内的所有非中文字符，以及和小说内容无关的片段，得到字符串形式的语料库。

3、 按照”分词“和”分字符“两种不同模式生成词频字典，在”分词“模式下，用 jieba 库中的 cut 函数对原始语料库进行处理，在”分字符“模式下，使原始语料库

4、 根据一元、二元、三元的相关公式获取得到词频表

5、 根据词频表的概率，利用相关公式，求解各个文件的以字和词两种形式的信息熵

白马啸西风,碧血剑，飞狐外传，连城诀,鹿鼎记，鹿鼎记,射雕英雄传,神雕侠侣,书剑恩仇录，天龙八部,侠客行,笑傲江湖,雪山飞狐,倚天屠龙记,鸳鸯刀,越女剑

Experimental Studies

通过 python 的计算，获得了各个 txt 文件以字和词，在一元、二元、三元模型下的平均信息熵

```
文件名： D:\(1)\三十三剑客图.txt
unigram熵（字）： 9.668181
bigram熵（字）： 4.835396
trigram熵（字）： 0.982960
unigram熵（词）： 11.681753
bigram熵（词）： 2.940446
trigram熵（词）： 0.271587
```

```
文件名： D:\(1)\书剑恩仇录.txt
unigram熵（字）： 9.465975
bigram熵（字）： 5.780698
trigram熵（字）： 2.391201
unigram熵（词）： 11.712306
bigram熵（词）： 5.031480
trigram熵（词）： 1.039232
```

```
文件名： D:\(1)\侠客行.txt
unigram熵（字）： 9.152494
bigram熵（字）： 5.591191
trigram熵（字）： 2.369172
unigram熵（词）： 11.187687
bigram熵（词）： 4.957815
trigram熵（词）： 1.127015
```

```
文件名： D:\(1)\倚天屠龙记.txt
unigram熵（字）： 9.393248
bigram熵（字）： 6.021115
trigram熵（字）： 2.805284
unigram熵（词）： 11.758690
bigram熵（词）： 5.519220
trigram熵（词）： 1.328786
```

```
文件名: D:\(1)\天龙八部.txt
unigram熵（字）: 9.404126
bigram熵（字）: 6.124947
trigram熵（字）: 2.949546
unigram熵（词）: 11.734144
bigram熵（词）: 5.678140
trigram熵（词）: 1.481071
```

```
文件名: D:\(1)\射雕英雄传.txt
unigram熵（字）: 9.438576
bigram熵（字）: 6.062098
trigram熵（字）: 2.760550
unigram熵（词）: 11.827972
bigram熵（词）: 5.470345
trigram熵（词）: 1.267269
```

```
文件名: D:\(1)\白马啸西风.txt
unigram熵（字）: 8.910503
bigram熵（字）: 4.582740
trigram熵（字）: 1.620533
unigram熵（词）: 10.238114
bigram熵（词）: 3.993585
trigram熵（词）: 0.741710
```

```
文件名: D:\(1)\碧血剑.txt
unigram熵（字）: 9.445713
bigram熵（字）: 5.869314
trigram熵（字）: 2.353483
unigram熵（词）: 11.736674
bigram熵（词）: 5.002440
trigram熵（词）: 0.993712
```

```
文件名: D:\(1)\神雕侠侣.txt
unigram熵（字）: 9.372591
bigram熵（字）: 6.063014
trigram熵（字）: 2.846327
unigram熵（词）: 11.731231
bigram熵（词）: 5.532182
trigram熵（词）: 1.350327
```

```
文件名: D:\(1)\笑傲江湖.txt
unigram熵（字）: 9.206285
bigram熵（字）: 5.897625
trigram熵（字）: 2.870949
unigram熵（词）: 11.396663
bigram熵（词）: 5.621101
trigram熵（词）: 1.527772
```

```
文件名: D:\(1)\越女剑.txt
unigram熵（字）: 8.823919
bigram熵（字）: 3.642355
trigram熵（字）: 0.910546
unigram熵（词）: 10.071071
bigram熵（词）: 2.529464
trigram熵（词）: 0.327490
```

```
文件名: D:\(1)\连城诀.txt
unigram熵（字）: 9.170998
bigram熵（字）: 5.398803
trigram熵（字）: 2.157033
unigram熵（词）: 11.040701
bigram熵（词）: 4.693123
trigram熵（词）: 0.955601
```

```
文件名： D:\(1)\雪山飞狐.txt
unigram熵（字）： 9.201373
bigram熵（字）： 5.163626
trigram熵（字）： 1.758497
unigram熵（词）： 11.107393
bigram熵（词）： 4.113304
trigram熵（词）： 0.693900
```

```
文件名： D:\(1)\飞狐外传.txt
unigram熵（字）： 9.307863
bigram熵（字）： 5.753991
trigram熵（字）： 2.407004
unigram熵（词）： 11.526291
bigram熵（词）： 4.996462
trigram熵（词）： 1.053255
```

```
文件名： D:\(1)\鸳鸯刀.txt
unigram熵（字）： 9.033372
bigram熵（字）： 4.215351
trigram熵（字）： 1.117573
unigram熵（词）： 10.476217
bigram熵（词）： 3.085965
trigram熵（词）： 0.432043
```

```
文件名： D:\(1)\鸳鸯刀.txt
unigram熵（字）： 9.033372
bigram熵（字）： 4.215351
trigram熵（字）： 1.117573
unigram熵（词）： 10.476217
bigram熵（词）： 3.085965
trigram熵（词）： 0.432043
```

```
文件名： D:\(1)\鹿鼎记.txt
unigram熵（字）： 9.281200
bigram熵（字）： 5.993113
trigram熵（字）： 2.953184
unigram熵（词）： 11.445723
bigram熵（词）： 5.764458
trigram熵（词）： 1.618579
```

```
所有文件：
unigram熵（字）： 9.527281
bigram熵（字）： 6.726566
trigram熵（字）： 3.951511
unigram熵（词-）： 12.178831
bigram熵（词）： 6.950565
trigram熵（词）： 2.299706
```