Reports of NLP Assignment 1 (22-23 Autumn)

Due to tasks given on page 87 of slides for chapter 2, this report is divided to 3 parts, including spiders for both English and Chinese texts, data preprocessing and computing Shannon entropy, and experiment results analysis.

1. Spiders for both English and Chinese texts

Chinese and English texts were separately obtained from websites 'https://www.chinanews.com.cn/scroll-news/news1.html' and 'https://m.wuxiaworld.co/' using package 'Xpath' and 'requests'. The former website contains instant news in China, while the latter one contains popular novels in English.

1.1 Spider for Chinese texts

Code was run as follows:

```
import requests
from bs4 import BeautifulSoup
import datetime
from multiprocessing import Pool

# 用request和BeautifulSoup处理网页
def requestOver(url):
    response = requests.get(url)
    response.encoding = 'utf-8'
    if("gb2312" in response.text):
        response.encoding = 'gb2312'
    soup = BeautifulSoup(response.text, 'html.parser')
    return soup

# 从网页下载标题和内容到txt文档
def download(title, url):
    soup = requestOver(url)
    tag = soup.find('div', class_="left_zw")
    if(tag == None):
        return 0
    title = title.replace(':', '')
    title = title.replace('"', '')
    title = title.replace('|', '')
    title = title.replace('/', '')
    title = title.replace('\\', '')
    title = title.replace('*', '')
    title = title.replace('<', '')
    title = title.replace('>', '')
    title = title.replace('?', '')
    content = ""
    for p in tag.findAll('p'):
        if (p.string != None):
```

```
            content = content + p.string
    filename = r'/Users/renee/Downloads/datachinanewswenhua.txt'
    with open(filename, 'a', encoding='utf-8', errors='ignore') as file_object:
        file_object.write('            ')
        file_object.write(title)
        file_object.write(tag.get_text())
    print('正在爬取新闻:' + title + " " + url)

# 爬虫具体执行过程
def crawlAll(url):
    soup = requestOver(url)
    for s in soup.findAll("div", class_="content_list"):
        for tag in s.findAll("li"):
            sp = tag.findAll("a")
            if("文化" in str(sp)):#更换新闻标签
                title = list(sp)[1].string
                urlAll = "http://www.chinanews.com" + str(list(sp)[1])[9:str(list(sp)
[1]).find("shtml")+5]
                try:
                    download(title, urlAll)
                except Exception:
                    print("新闻爬取失败")


if __name__ == '__main__':
    pool = Pool(4)
    collection = set()
    url1 = "http://www.chinanews.com/scroll-news/"
    date = "2022/0926"
    url2 = "/news.shtml"
    p1 = []
    # 3650：十年的新闻数据
    for i in range(365*14):
        date1 = datetime.datetime.strptime(date, "%Y/%m%d")
        date2 = datetime.timedelta(days=-1)
        date = (date1 + date2).strftime("%Y/%m%d")
        target_url = url1 + date + url2
        p1.append(target_url)
        print(target_url)
    pool.map(crawlAll, p1)
    pool.close()
    pool.join()
```

Results were stored in 'datachinanewswenhua.txt'.

1.2 Spider for English texts

Code was run as follows:

```
# usr/bin/python
# -*- coding: utf-8 -*-
import re
from urllib import request
from lxml import etree
import sys
import os


title='Hellbound-With-You'#修改小说名称
homepage_url = 'http://m.wuxiaworld.co/'+title+'/'
headers = {'User-Agent':'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/60.0.3112.78 Safari/537.36'}

def get_all_url():
    contentpage_list = []
    # 验证list是否为空
    if len(contentpage_list) != 0:
        raise ValueError("该列表不为空")
    request_url = homepage_url
    request1 = request.Request(request_url, headers=headers)
    response = request.urlopen(request1)
    homepage = response.read()
    homepage_tree = etree.HTML(homepage)
    urls = homepage_tree.xpath('//ul[@class="chapter-list clearfix"]/a/@href')
    '''if len(index_node) == 1:
        index_node = index_node[0]
    else:
        raise ValueError("有不止一个目录节点")
    print index_node'''
    # print etree.tostring(index_node) # 打印当前节点中的内容
    #urls = index_node.xpath('.//p/a[starts-with(@title,"ST Book")]')       # 第九章开始
没有title了，爬不全
    #print [etree.tostring(each) for each in urls]
    print (len(urls))
    #for each in urls:
     #   url = each.xpath('./@href')
      #  if len(url) == 1:
       #     contentpage_list.append(url[0])
    contentpage_list =list(urls)
    return contentpage_list


def get_content(contentpage_url, txt):
    request_url = contentpage_url
```

```
    request2 = request.Request(request_url, headers=headers)
    response =request.urlopen(request2)
    contentpage = response.read()
    contentpage_tree = etree.HTML(contentpage)
    content_root_node = contentpage_tree.xpath('//*[@class="chapter-entity"]')[0]
    #print len(content_root_node)
    content = content_root_node.xpath(u'./text()')


    print(content)
    fp = open(txt + '.txt', 'a',encoding='utf-8')
    for each in content:

        fp.write('%s' % each + '\n')
    fp.close()

    #print etree.tostring(content_root_node[0])



if __name__ == '__main__':
    contentpage_list = get_all_url()
    #print contentpage_list[0]
    #contentpage_list = ['']
    #contentpage_list[0] = 'http://www.wuxiaworld.com/st-index/st-book-1-chapter-1/'
    for url in contentpage_list:
        each_url='http://m.wuxiaworld.co/'+url

        get_content(each_url, title)
```

Results were stored in title+ '.txt'.

2. Data preprocessing and computing Shannon entropy

2.1 Introduction of Shannon Entropy

Shannon entropy was introduced by in his 1948 paper "A Mathematical Theory of Communication", which is defined as an index indicating uncertainty of information. It can be described as

$$H(x) = \sum_{x \in X} p(x) log_2 p(x) \tag{1}$$

Where x is a discrete random variable with its probility distribution as

$$p(x) = P(X = x) \tag{2}$$

2.2 Spliting TXT Files in Size

Size of texts increase 2M every time to observe change of Shannon entropy. So firstly, text data is supposed to be divided into several small files with size 2M.

Code was run as follows:

```
def split():
    # 读取源文件，文件名最好加上绝对路径
    title='Super-Gene'
    with open('/Users/renee/Desktop/'+title+'.txt', 'r') as f:
        # 把数据写入列表
        wordlist = f.read()
        # 算出总字数
        length = len(wordlist)
    # 设置每个拆分文件的字数
    unit = int(1024*1024*2)
    # 计算新文件的个数，如果总行数整除新文件行数，就取这个商的值，如果不整除，取商加1的值
    file_amount = length // unit + 1 if length % unit > 0 else length // unit
    # 遍历所有新文件
    for num in range(file_amount):
        # 计算新文件中第一行在源文件中对应的行号
        start = num * unit
        # 计算新文件中最后一行在源文件中对应的行号
        end = length if length < (num + 1) * unit else (num + 1) * unit
        # 写入新文件，文件名最好加上绝对路径
        with open(title+str(num + 1) + '.txt', 'w+') as f:
            # 遍历新文件的所有行
            for i in range(start, end):
                # 把列表中的数据写入新文件
                f.write(wordlist[i])

if __name__ == '__main__':
    split()
```

2.3 Data preprocessing and computing Shannon entropy for Chinese texts

Data preprocessing is essential because noise of other letters will impact results of Shannon entropy. Special characters such as '#$%&' and etc. are supposed to be filtered and deleted before computing Shannon entropy.

Code was run as follows:

```
import jieba
import math
import time
```

```
import re

class TraversalFun():

    # 1 初始化
    def __init__(self, rootDir):
        self.rootDir = rootDir

    def TraversalDir(self):
        return TraversalFun.getCorpus(self, self.rootDir)

    def getCorpus(self, rootDir):
        corpus = []
        r1 = u'[a-zA-Z0-9’!"#$%&\'()*+,-./:： ;<=>?@，。?★、…【】《》？“”‘’！[\\]^_`{|}~]+'
 # 用户也可以在此进行自定义过滤字符
        count=0
        with open(rootDir, "r", encoding='utf-8') as file:
            filecontext = file.read();
            filecontext = re.sub(r1, '', filecontext)
            filecontext = filecontext.replace("\n", '')
            filecontext = filecontext.replace(" ", '')

            #seg_list = jieba.cut(filecontext, cut_all=True)
            #corpus += seg_list
            count += len(filecontext)
            corpus.append(filecontext)
        return corpus,count

def cal_unigram(corpus,count):
    before = time.time()
    split_words = []
    words_len = 0
    line_count = 0
    words_tf={}
    for line in corpus:
        for x in jieba.cut(line):
            split_words.append(x)
            words_len += 1
        get_tf(words_tf, split_words)
        split_words = []
        line_count += 1
    count_0=[]
    words_len_0=[]
    entropy_0=[]
    count_0.append(count)
    words_len_0.append(words_len)

    print("语料库字数:", count)
    print("分词个数:", words_len)
```

```
    print("平均词长:", round(count / words_len, 5))
    entropy = []
    for uni_word in words_tf.items():
        entropy.append(-(uni_word[1] / words_len) * math.log(uni_word[1] / words_len,
2))
    en=round(sum(entropy), 5)
    entropy_0.append(en)
    print("基于词的一元模型的中文信息熵为:",en , "比特/词")
    after = time.time()
    print("运行时间:", round(after - before, 5), "s")
    return entropy_0

# 词频统计，方便计算信息熵
def get_tf(tf_dic, words):

    for i in range(len(words)-1):
        tf_dic[words[i]] = tf_dic.get(words[i], 0) + 1



if __name__ == '__main__':

    #tra ='' #TraversalFun("/Users/renee/Downloads/datachinanewswenhua.txt")

    for i in range(1,50):
        print(i)
        path='/Users/renee/Documents/'+str(i)+'.txt'
        with open(path) as data:
            datacontext=data.read()
        data.close()
        with open("/Users/renee/Downloads/d1.txt",'a') as data_1:
            data_1.write(datacontext)

        add_tra = TraversalFun("/Users/renee/Downloads/d1.txt" ) # 需要进行分割的文件，请修
改文件名
        corpus,count=add_tra.TraversalDir()
        cal_unigram(corpus, count)
```

Then we use a diagram to describe relationship between file size and Shannon entropy.

Code was run as follows:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(20,102,40)
plt.title('Change of Shannon Entropy for Chinese texts')
plt.xlabel('File Size(M)')
plt.ylabel('Shannon Entropy(Bits/letter)')
plt.plot(x, entropy_0[9::],color='purple')
```

The result is shown as follows:

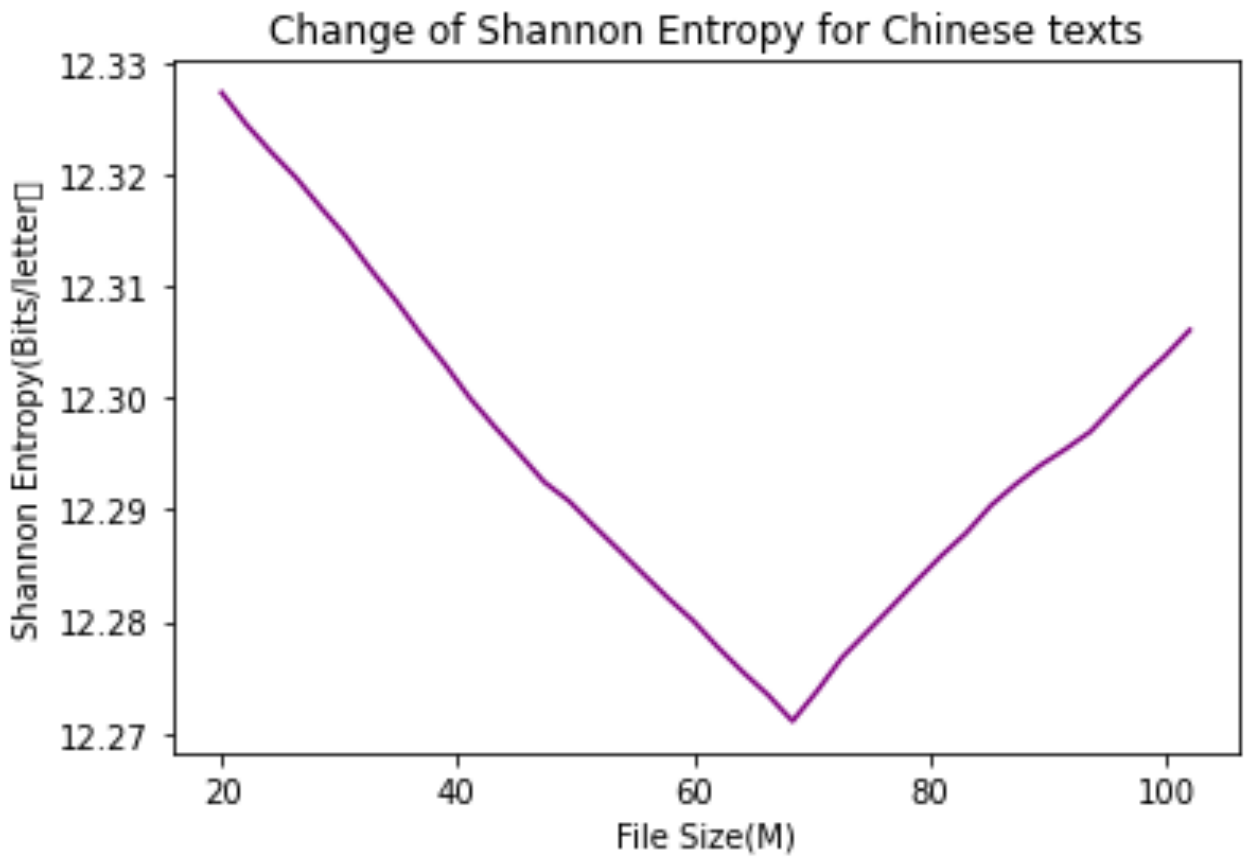

Figure 1

2.4 Data preprocessing and computing Shannon entropy for English texts

Data preprocessing for English texts is similar with Chinese. It's easier to compute Shannon entropy for English texts because dictionary of English letters is fixed.

Code was run as follows:

```
import re
import math
from prettytable import PrettyTable
import numpy
```

```
#import random
#import time

entropy_0 = []

for i in range(1, 46):
    print(i)
    path = '/Users/renee/Documents/Eng/' + str(i) + '.txt'
    with open(path) as data:
        datacontext = data.read()
    data.close()
    with open("/Users/renee/Documents/Eng/d2.txt", 'a') as data_1:
        data_1.write(datacontext)
        data_1.close()
    with open("/Users/renee/Documents/Eng/d2.txt", 'r', encoding='UTF-8') as f:
        content = f.read()

    # print('预处理前的字符串：')
    #print(content)
    # Str =content#引入数据
    content = content.replace("Please go to", '')
    content = content.replace("install our App to read the latest chapters for free",
'')
    content = content.replace("Previous Chapter", '')
    content = content.replace("Next Chapter", '')
    content = content.replace("E.3.3.", '')
    content = content.replace("Chapter", '')
    a = re.findall(r'[^\*"/:?|,!-.%‘—;()’“”‘’。°【0123—456789】<>\[\]]', content, re.S)
 #
    a1 = "".join(a)  # 去掉特殊字符
    b = a1.lower()  # 大写变小写
    c = ' '.join(b.split())  # 去连续的空格
    Str1 = c
    # print('预处理后的字符串：')
    # print(Str1)
    sum_1 = 0
    Hx = 0
    k = 0
    j = 0
    s = 0
    Hx = []
    for a1 in Str1:
        if a1 in 'abcdefghijklmnopqrstuvwxyz ':
            sum_1 += 1
    print(f'"字符总数为":{sum_1}个')
    # strfloat=numpy.empty(27,dtype=float)
    # strArr=numpy.empty(27,dtype=str)
    resoult = {} # 定义一个空字典
    for i in 'abcdefghijklmnopqrstuvwxyz ':  # 遍历输入的字符串，以键值对的方式存储在字典中
```

```
        resoult[i] = Str1.count(i)
        # print(type(Str1.count))
    for key in resoult:  # 遍历字典，格式化输出结果
        Hx.append((resoult[key] / sum_1) * math.log((sum_1 / resoult[key]), 2))  # 信
    print(Hx)
    Hx_1 =round(sum(Hx),5)
    print(Hx_1)
    entropy_0.append(Hx_1)
```

Then we use a scatter diagram to describe relationship between file size and Shannon entropy.

Code was run as follows:

```
import matplotlib.pyplot as plt
import numpy as np

x=np.linspace(2,92,45)
plt.title('Change of Shannon Entropy for English texts')
plt.xlabel('File Size(M)')
plt.ylabel('Shannon Entropy(Bits/letter) ')
plt.plot(x, entropy_0,color='purple')
```

The result is shown as follows:

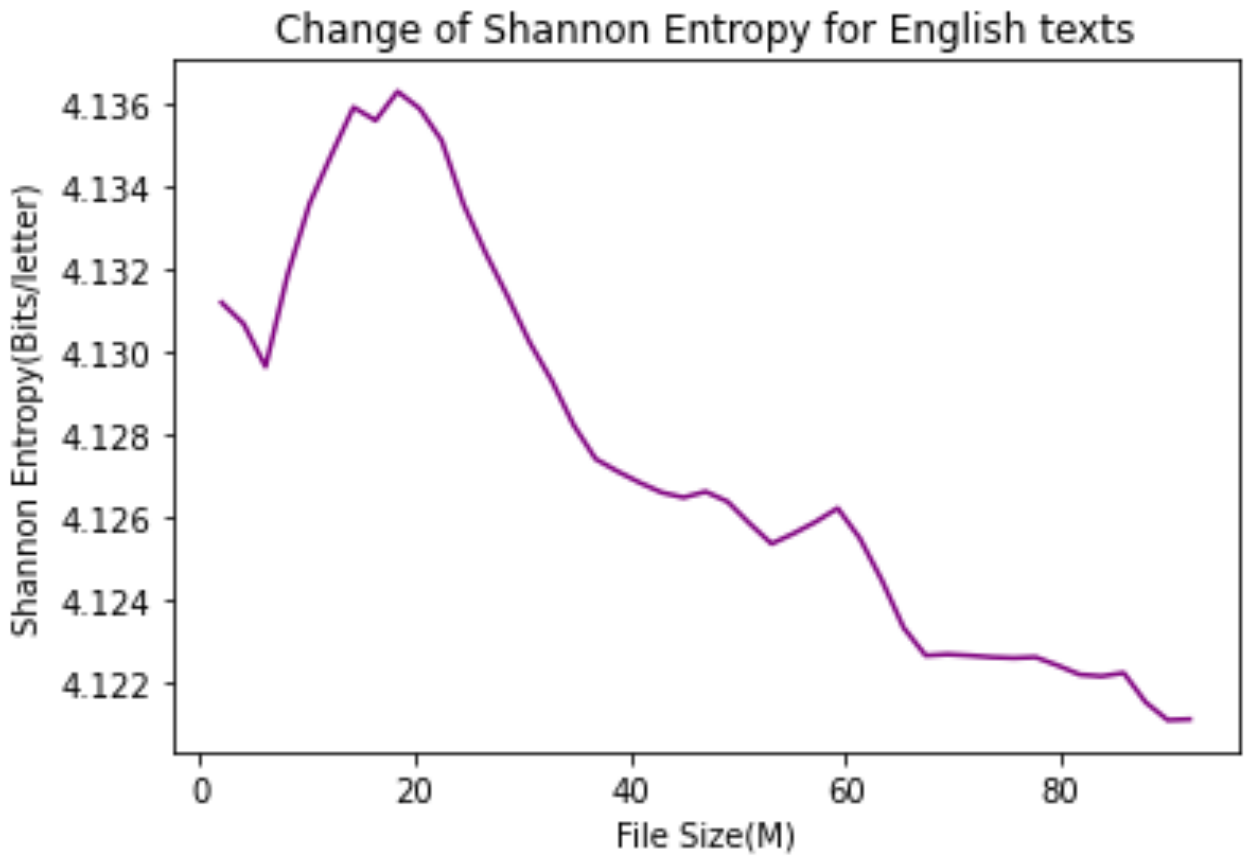

Figure 2

3.Experiment results analysis

Sizes for both Chinese and English texts are around 100M. For Chinese texts, Shannon entropy first decreases to 12.27111 bits/letter at size 68M and then increases, while it generally increases to 4.13629 bits/letter at size 18M and then drop sharply with file size increasing.

The reason why entropy for Chinese texts is much bigger than prediction (given on page 15 of slides for chapter 2) perhaps is that Chinese texts are collected from news as written language with a large quantity of proper nouns.

End

Deng Ruxin

2022.09.30