文本可视化：个性化词云及主题分析

PB22061267 周嘉煊

1. 简介

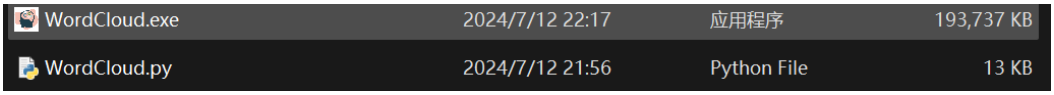
该大作业核心功能为文本可视化。

具体而言主要分为蒙版的生成、个性化词云与主题分析。

除此之外，还支持从网页上抓取中文，进行文本分析，更换词云字体等功能。

小组成员只有笔者一人。

已使用 Pyinstaller 打包成 exe 文件，便于未配置 python 相关环境的用户直接使用。



上图中WordCloud.py 为源代码 WordCloud.exe 为打包好的应用程序，点击即可运行。

2. 所使用的库

使用的外部库较多，主要可分为如下几类：

2.1 爬虫类

```
import requests
from bs4 import BeautifulSoup
import re
import webbrowser
```

Fence 2-1

其中

- requests 用于请求网页连接
- BeautifulSoup 用于解析 html 内容
- re 用于提取所需内容
- webbrowser 用于向用户展现 html 文件

2.2 中文分词

由于中文的词与词之间无空格隔开，所以需要借助外部库分词。

这里使用的著名的 jieba 库。

2.3 UI 界面

这里使用的是 tk 库，虽然 tk 库的外观不如 ttk 库看起来现代化，但兼容效果更好，也更加稳定。

2.4 文件管理

由于需要对文件进行读取、删除、存储等操作，需要用到文件管理相关的库。

```
import sys
import os
from shutil import rmtree
from tkinter import filedialog
```

Fence 2-2

其中，os 库和 sys 库用于获取文件路径，rmtree 用于清空文件夹，firedialog 用于用户选择文件、文件夹。

2.5 主题分析

```
from gensim.models.ldamodel import LdaModel
from gensim.corpora.dictionary import Dictionary
import pyLDAvis.gensim_models
```

Fence 2-3

pyLDAvis 库中有较为完善的用于主题分析的模型框架。

Dictionary 用于将本地的文本文件转换为训练模型的形式。

2.6 图片生成及处理

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from PIL import Image, ImageTk
```

Fence 2-4

其中

- plt 用于展示图片
- WordCloud 用于生成词云图
- PIL 用于对选择的图片进行操作，生成合适的蒙版

3. 实验思路

3.1 生成词云蒙版

蒙版最好是黑白二值图，但实践发现，无法正常读取二值图，故采用数值差异较大的灰度图，阈值由用户预览效果后，自行选择。

先将图片转化为灰度图，然后低于阈值的灰度设置为 0，高于阈值的设置为 1.

核心代码：

```
    def create_image(self):
        img = Image.open(self.filename).convert('L')
        table = []
        for i in range(256):
            if i < self.threshold:
                table.append(0)
            else:
                table.append(255)
```

```
        # 图片二值化
        self.photo2 = img.point(table, 'L')
        self.photo2 = self.photo2.resize((self.base_width,
self.h_size))
        a = ImageTk.PhotoImage(self.photo2)
        self.photo_image.config(image= a)
        self.label_image.image = a
```

Fence 3-1

3.2 生成词云

文本中，有些词语虽然出现很多次，但并不具备太多意义，比如“的”，“了”等助词。所以事先下载好停用词表。

中文文本需要借助jieba库进行分词，英文文本则不需要。

同时可以使用 jieba.analyse 调节关键词的权重（而不仅仅是从频率上判断不同词汇的重要性）。

WordCload还可以选择字体，笔者准备了多种字体（.ttf后缀的文件）.

核心代码：

```
def CreateCloud():
    global Text
    global texts
    name = name_entry.get()
    if name == '':
        name = Text.split('/')[-1].split('.')[0]
    if url == None or url == '':
        with open(Text, 'r', encoding='utf-8') as f:
            texts = f.read()
    # 中文分词
    global mask
    freq = jieba.analyse.extract_tags(texts, topK=200,
withWeight=True)
    freq = {i[0]: i[1] for i in freq}
    Mask = np.array(Image.open(mask)) if mask != None else None
    wc = WordCloud(mask =
Mask,font_path=ziti+fonttype.get()+".ttf", mode=modetype.get(),
background_color=None,stopwords =
stopwords).generate_from_frequencies(freq)
    if var.get() is True:
        wc.to_file(picture+name+".png")
    # 显示词云
```

```
    plt.imshow(wc, interpolation='bilinear')
    plt.axis('off')
    plt.show()
```

Fence 3-2

3.3 中文网页爬取

由于网页的绝大部分代码为英文（我们只希望抓取文本，而非代码），所以要爬取中文网页，这里直接把解析后的网页的中文提取出来再合并即可。

核心代码：

```
def fetch_chinese_text(url):
    response = requests.get(url)
    response.raise_for_status()  # 如果请求失败，抛出HTTPError异常
    soup = BeautifulSoup(response.text, 'html.parser')
    text = soup.get_text()
    chinese_text = re.findall(r'[\u4e00-\u9fa5，。？！""‘’]', text)

    chinese_text_str = ''.join(chinese_text)
    return chinese_text_str
```

Fence 3-3

3.4 主题分析

3.4.1 分词与筛选

- 使用jieba分词库（假设jp是jieba的别名或实例）对文本进行分词。
- 筛选分词结果，只保留词性标签在flags列表中的词，并且排除掉stopwords（停用词列表）中的词。

3.4.2 构建词典和语料库

- 使用gensim的Dictionary类根据筛选后的词列表构建词典。

3.4.3 LDA主题建模

- 使用gensim的LdaModel类对语料库进行LDA主题建模，设置主题数为5，迭代次数为50，并指定随机状态以保证结果的可重复性。

3.4.4 可视化

- 使用pyLDAvis库将LDA模型的结果可视化。

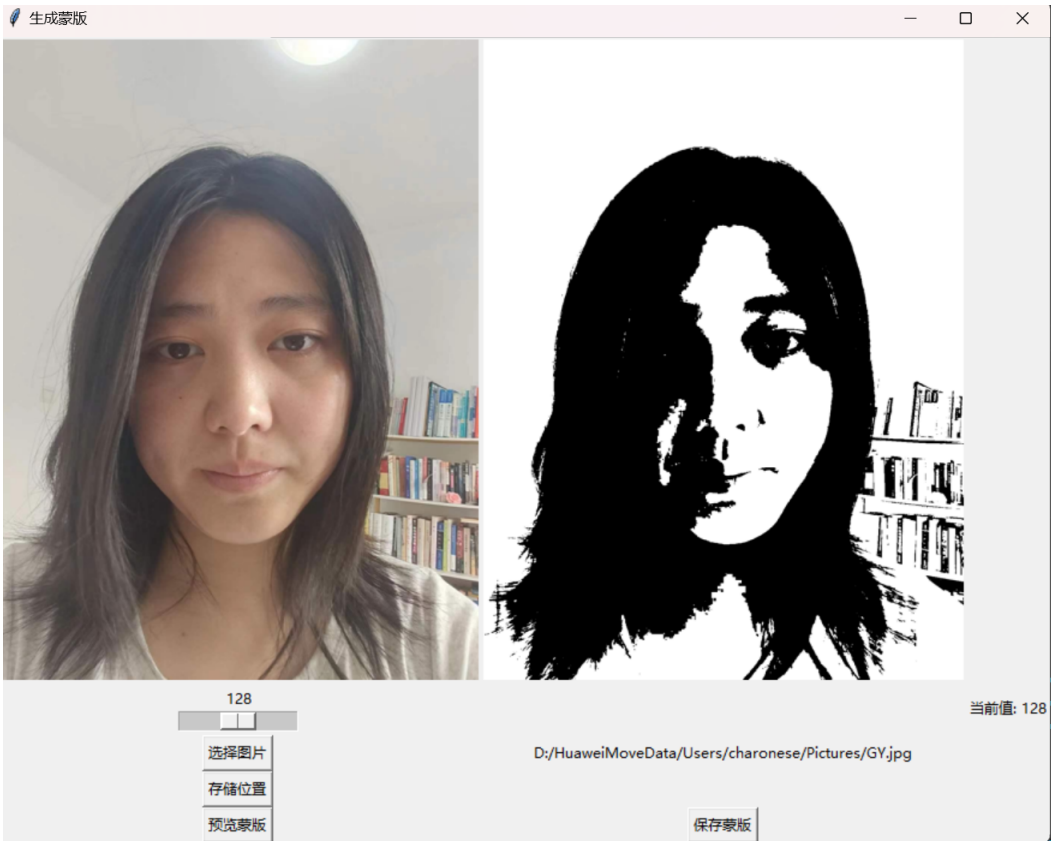
核心代码：

```
def analysis():
    global texts
    flags = ('n', 'nr', 'ns', 'nt', 'eng', 'v', 'd','vn','vd')
    words = [[word.word for word in jp.cut(texts) if word.flag in
flags and word.word not in stopwords]]
    dictionary = Dictionary(words)
    corpus = [dictionary.doc2bow(words[0])]
    lda = LdaModel(corpus=corpus, id2word=dictionary, num_topics=5,
random_state=100, iterations=50)
    plot =pyLDAvis.gensim_models.prepare(lda,corpus,dictionary)
    save_html = road('主题分析/')+Text.split('/')[-1].split('.')
[0]+'.html'
    pyLDAvis.save_html(plot,save_html)
    webbrowser.open(save_html)
```

Fence 3-4

4. 效果展示

4.1 生成蒙版



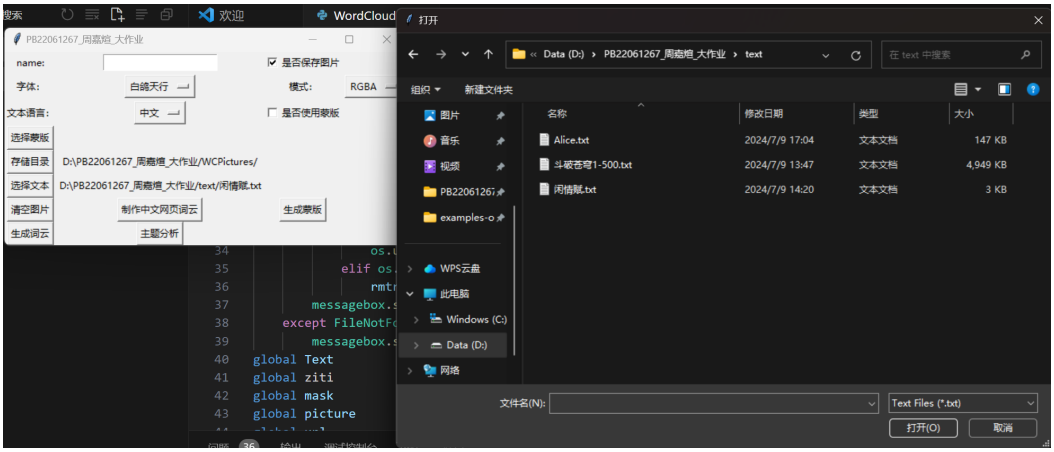
点击生成蒙版，选择本地图片，拖动进度条选择阈值，预览后可存储在选定位置（可更改）。

（上图照片为本人朋友，使用素材前已征得其同意）

4.2 生成词云

4.2.1 中文词云

选择蒙版和文本后，可生成词云。



下图为以女性身体形状的蒙版，斗破苍穹前五百章为文本生成的词云图。

4.2.2 英文词云

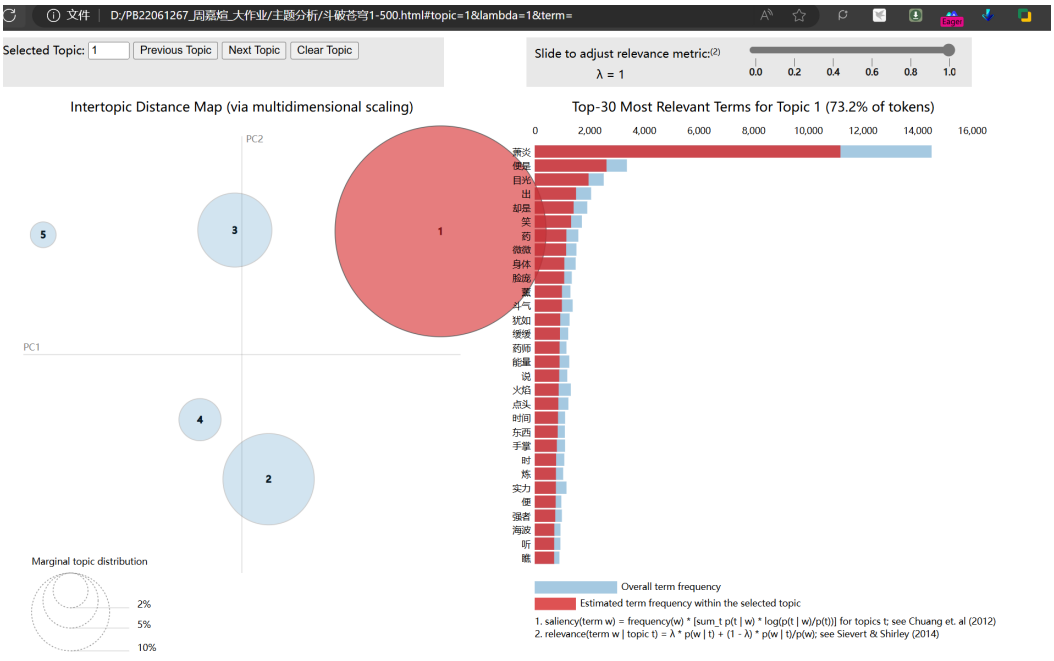
下图为以爱丽丝梦游仙境为为本，本人朋友照片生成的蒙版生成的英文词云.

注意，中文的字体ttf文件同时支持英文，反之则不成立，英文的字体只支持英文（这也是为何英文的ttf字体文件比中文小很多）。

4.3 主题分析

选择文本，点击主题分析即可。

下图为斗破苍穹前500章的主题分析。可以观察到，斗破苍穹的主题较为统一，不愧为网络小说中的经典之作。



4.4 中文网页词云

点击制作中文网页词云，输入网址，再点击生成词云即可。下图为输入中科大的百度百科网址后，生成的词云图。

5. 附录

5.1 完整源码

```
import matplotlib.pyplot as plt
import jieba.posseg as jp
import numpy as np
import jieba.analyse
import sys
import os
import tkinter as tk
from tkinter import messagebox,filedialog
from wordcloud import WordCloud
from PIL import Image, ImageTk
```

```
from shutil import rmtree
import requests
from bs4 import BeautifulSoup
import re
from gensim.models.ldamodel import LdaModel
from gensim.corpora.dictionary import Dictionary
import pyLDAvis.gensim_models
import webbrowser

current_path = os.path.dirname(os.path.realpath(sys.argv[0]))
def road(file) :
    return (current_path + '/'+file)
def warning():
    askback = messagebox.askyesno('温馨提示','亲，确认清空文件夹？（∩
__∩)')
    return askback
def empty_folder(folder_path):
    ask = warning()
    if ask == False:
        return ask
    try:
        for filename in os.listdir(folder_path):
            file_path = os.path.join(folder_path, filename)
            if os.path.isfile(file_path) or
os.path.islink(file_path):
                os.unlink(file_path)
            elif os.path.isdir(file_path):
                rmtree(file_path)
        messagebox.showinfo('succeed',f"成功清空文件夹
{folder_path}!")
    except FileNotFoundError:
        messagebox.showinfo(f"文件夹 {folder_path} 不存在!")
global Text
global ziti
global mask
global picture
global url
global texts
mask = None
Text = road('text/闲情赋.txt')
ziti = road('字体/')
picture = road('WCPictures/')
url = None
```

```
texts = open(Text, 'r', encoding='utf-8').read()
# 创建一个GUI窗口
root = tk.Tk()
root.title('PB22061267_周嘉煊_大作业')

mode_label = tk.Label(root,text="模式：")
modetype = tk.StringVar(root)
modetype.set('RGBA')  # 默认值
mode_options = ['RBGA','RGB']
mode_dropdown = tk.OptionMenu(root,modetype,*mode_options)

language_label = tk.Label(root,text="文本语言：")
languagetype = tk.StringVar(root)
languagetype.set('中文')  # 默认值
language_options = ['中文','English']
font_label = tk.Label(root,text="字体：")
fonttype = tk.StringVar(root)

if(languagetype.get() == '中文'):
    fonttype.set('白鸽天行')  # 默认值
    font_options = ['白鸽天行','瘦金加粗','敦煌飞天']
else:
    fonttype.set('TypeWriter')  # 默认值
    font_options = ['TypeWriter','LOVEQueen','NoteScript']
font_dropdown = tk.OptionMenu(root,fonttype,*font_options)

def on_language_change(a):
    global font_dropdown
    font_dropdown.destroy()
    if(a == '中文'):
        fonttype.set('白鸽天行')  # 默认值
        font_options = ['白鸽天行','瘦金加粗','敦煌飞天']
    else:
        fonttype.set('TypeWriter')  # 默认值
        font_options = ['TypeWriter','LOVEQueen','NoteScript']
    font_dropdown = tk.OptionMenu(root,fonttype,*font_options)
    font_dropdown.grid(row=1,column=1)
language_dropdown =
tk.OptionMenu(root,languagetype,*language_options,command=on_langua
ge_change)
global stopwords
```

```
stopwords = set([line.strip() for line in
open(road('stopWords.txt'),'r',encoding='utf-8').readlines()])

def ChangeMask():
    if Qmask.get() == True:
        global mask
        a = filedialog.askopenfilename(filetypes=[("PNG Files",
"*.png"),("JPG Files", "*.jpg")],initialdir=current_path)
        if a!= '':
            mask = a
    else:
        mask = None
    ChangeMask_Label.config(text=mask)
    return mask
def ChangeText():
    global Text
    a = filedialog.askopenfilename(filetypes=[("Text Files",
"*.txt")],initialdir=current_path)
    if a != '':
        Text = a
    ChangeText_Label.config(text=Text)
    url = None
    global texts
    texts = open(Text, 'r', encoding='utf-8').read()
    return Text
def ChangePicture():
    global picture
    a = filedialog.askdirectory(title = "请选择一个文件
夹",initialdir=current_path)
    if a!= '' :
        picture = a
    ChangePicture_Label.config(text=picture)
    return picture
def CreateMask():
    new_window = tk.Toplevel(root)
    new_window.title("新窗口")
    ImageApp(new_window)

ChangeMask_button = tk.Button(root, text='选择蒙
版',command=ChangeMask)
ChangeMask_Label = tk.Label(root,text=mask)
```

```
CreateMask_button = tk.Button(root, text='生成蒙
版',command=CreateMask)

ChangeText_button =tk.Button(root, text='选择文
本',command=ChangeText)
ChangeText_Label = tk.Label(root,text=Text)

ChangePicture_button = tk.Button(root, text='存储目
录',command=ChangePicture)
ChangePicture_Label = tk.Label(root,text=picture)

var = tk.BooleanVar(root,value=True)
checkbox = tk.Checkbutton(root, text='是否保存图片', variable=var)
Qmask = tk.BooleanVar(root,value=False)
MaskCheckbox = tk.Checkbutton(root, text='是否使用蒙版',
variable=Qmask)

class ImageApp:
    def __init__(self, root):
        self.root = root
        self.root.title("生成蒙版")
        def update_value(event=None):
            """更新标签显示的滑块值"""
            self.threshold = slider.get()
            label.config(text=f"当前值: {self.threshold}")
        # 图片显示区域
        self.base_width = 400
        self.h_size = 300
        self.photo = None
        self.photo2 = None
        self.label_image = tk.Label(root)
        self.label_image.grid(row=0,column=0)
        self.photo_image = tk.Label(root)
        self.photo_image.grid(row=0,column=1)

        self.threshold = 0
        # 初始状态
        self.filename = ""
        self.save_path = None
        # 创建一个滑块（进度条）
        slider = tk.Scale(root, from_=0, to=255,
orient="horizontal", command=update_value)
        slider.grid(row=1,column=0)
```

```
        # 创建一个标签用于显示当前值
        label = tk.Label(root, text="当前值: 0")
        label.grid(row=1,column=2)
        # 按钮
        load_btn = tk.Button(root, text="选择图片",
command=self.load_image)
        load_btn.grid(row=2,column=0)
        self.load_label = tk.Label(root,text=self.filename)
        self.load_label.grid(row=2,column=1)

        save_btn = tk.Button(root, text="存储位置",
command=self.change_save_image)
        save_btn.grid(row=3,column=0)
        self.save_label = tk.Label(root,text=self.save_path)
        self.save_label.grid(row=3,column=1)

        create_btn = tk.Button(root, text="预览蒙版",
command=self.create_image)
        create_btn.grid(row=4,column=0)

        download_btn = tk.Button(root, text="保存蒙版",
command=self.save_image)
        download_btn.grid(row=4,column=1)

    def load_image(self):
        self.filename = filedialog.askopenfilename()
        if self.filename:
            # 使用Pillow打开图片，并转换为Tkinter可接受的格式
            img = Image.open(self.filename)
            self.h_size =
int((float(img.size[1])/float(img.size[0]) *self.base_width ))
            img = img.resize((self.base_width, self.h_size))  # 可
选：调整图片大小
            self.photo = ImageTk.PhotoImage(img)
            # 更新Label以显示新图片
            self.load_label.config(text = self.filename)
            self.label_image.config(image=self.photo)
            self.label_image.image = self.photo
    def create_image(self):
        img = Image.open(self.filename).convert('L')
        table = []
        for i in range(256):
```

```
            if i < self.threshold:
                table.append(0)
            else:
                table.append(255)
        # 图片二值化
        self.photo2 = img.point(table, 'L')
        self.photo2 = self.photo2.resize((self.base_width,
self.h_size))
        a = ImageTk.PhotoImage(self.photo2)
        self.photo_image.config(image= a)
        self.label_image.image = a
    def change_save_image(self):
        a = filedialog.askdirectory(title = "请选择一个文件
夹",initialdir=current_path)
        if a!= '' :
            self.save_path = a
        self.save_label.config(text=self.save_path)
    def save_image(self):

self.photo2.save(self.save_path+'/'+self.filename.split('/')[-1])

def empty_folder():
    ask = warning()
    if ask == False:
        return ask
    folder_path = picture
    try:
        for filename in os.listdir(folder_path):
            file_path = os.path.join(folder_path, filename)
            if os.path.isfile(file_path) or
os.path.islink(file_path):
                os.unlink(file_path)
            elif os.path.isdir(file_path):
                rmtree(file_path)
        messagebox.showinfo('succeed',f"成功清空文件夹
{folder_path}!")
    except FileNotFoundError:
        print(f"文件夹 {folder_path} 不存在!")

del_button = tk.Button(root, text="清空图片",command= empty_folder)

name_label = tk.Label(root,text='name:')
name_entry = tk.Entry(root)
```

```
def analysis():
    global texts
    flags = ('n', 'nr', 'ns', 'nt', 'eng', 'v', 'd','vn','vd')
    words = [[word.word for word in jp.cut(texts) if word.flag in
flags and word.word not in stopwords]]
    dictionary = Dictionary(words)
    corpus = [dictionary.doc2bow(words[0])]
    lda = LdaModel(corpus=corpus, id2word=dictionary, num_topics=5,
random_state=100, iterations=50)
    plot =pyLDAvis.gensim_models.prepare(lda,corpus,dictionary)
    save_html = road('主题分析/')+Text.split('/')[-1].split('.')
[0]+'.html'
    pyLDAvis.save_html(plot,save_html)
    webbrowser.open(save_html)

analysis_button = tk.Button(root,text='主题分析',command = analysis)

def CreateCloud():
    global Text
    global texts
    name = name_entry.get()
    if name == '':
        name = Text.split('/')[-1].split('.')[0]
    # 中文分词
    global mask
    freq = jieba.analyse.extract_tags(texts, topK=200,
withWeight=True)
    freq = {i[0]: i[1] for i in freq if i[0] not in stopwords}

    Mask = np.array(Image.open(mask)) if mask != None else None
    global stopwords
    wc = WordCloud(mask =
Mask,font_path=ziti+fonttype.get()+".ttf", mode=modetype.get(),
background_color=None,stopwords =
stopwords).generate_from_frequencies(freq)
    if var.get() is True:
        wc.to_file(picture+name+".png")
    # 显示词云
    plt.imshow(wc, interpolation='bilinear')
    plt.axis('off')
    plt.show()

def fetch_chinese_text(url):
```

```
    response = requests.get(url)
    response.raise_for_status()  # 如果请求失败，抛出HTTPError异常
    soup = BeautifulSoup(response.text, 'html.parser')
    text = soup.get_text()
    chinese_text = re.findall(r'[\u4e00-\u9fa5，。？！""‘’]', text)

    chinese_text_str = ''.join(chinese_text)
    return chinese_text_str
def create_popup():
    # 创建一个Toplevel窗口作为弹窗
    popup = tk.Toplevel()
    popup.title("抓取网页（中文）")
    # 在弹窗中添加一个输入框
    entry = tk.Entry(popup)
    entry.pack(pady=20)
    # 添加一个按钮，点击时获取输入框的值并关闭弹窗
    def on_submit():
        global url
        global texts
        url = entry.get()
        texts = fetch_chinese_text(url)
        if texts == '':
            messagebox.showinfo('目标网页无法解析出中文\\可能为全英网
页或被加密处理')
        ChangeText_Label.config(text=url)
        CreateCloud()
        popup.destroy()  # 关闭弹窗
    submit_button = tk.Button(popup, text="提交", command=on_submit)

    submit_button.pack(pady=10)

# 创建一个按钮，点击时调用create_popup函数
UrlButton = tk.Button(root, text="制作中文网页词云",
command=create_popup)

CreateCloud_button = tk.Button(root, text="生成词云",
command=CreateCloud)

name_label.grid(row=0, column=0)
name_entry.grid(row=0, column=1)
checkbox.grid(row=0, column=2)

font_label.grid(row=1, column=0)
```

```
font_dropdown.grid(row=1,column=1)

mode_label.grid(row=1, column=2)
mode_dropdown.grid(row=1,column=3)

language_label.grid(row=2, column=0)
language_dropdown.grid(row=2,column=1)
MaskCheckbox.grid(row=2,column=2)

ChangeMask_button.grid(row=3, column=0)
ChangeMask_Label.grid(row=3, column=1)

ChangePicture_button.grid(row=4, column=0)
ChangePicture_Label.grid(row=4, column=1)

ChangeText_button.grid(row=5, column=0)
ChangeText_Label.grid(row=5, column=1)

del_button.grid(row=6,column=0)
UrlButton.grid(row=6, column = 1)
CreateMask_button.grid(row= 6,column=2)

CreateCloud_button.grid(row=7, column=0)
analysis_button.grid(row=7,column=1)
# 运行GUI窗口
root.mainloop()
```

Fence 5-1