# **实验报告**

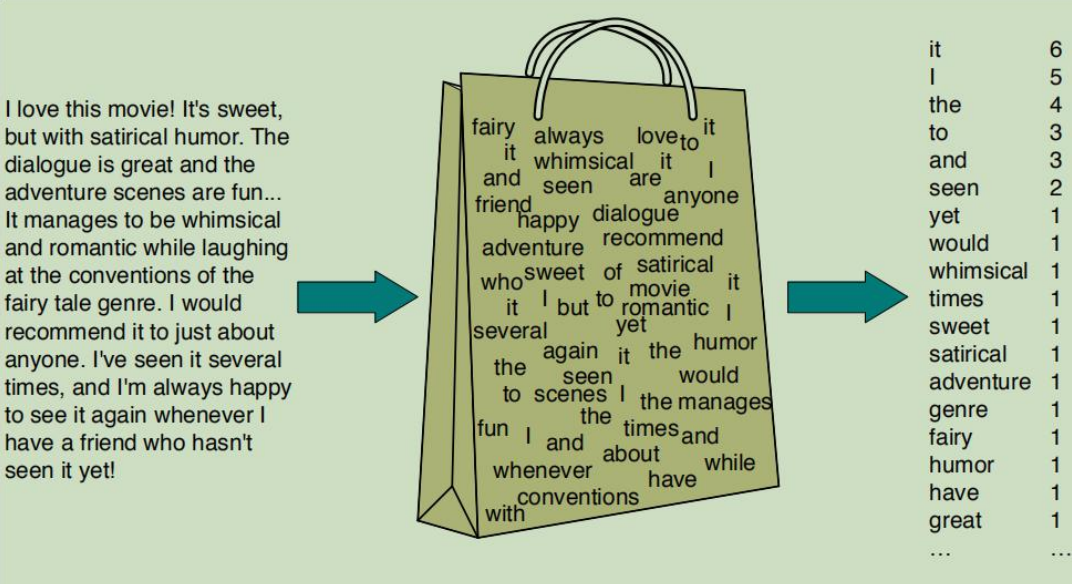
1. **实验内容**

本次实验使用词袋（bag of words）技术，利用词袋模型进行编程并计算了不少于10组句子对的相似度，同时设计了图形界面，可以在界面输入句子对，然后点击按钮便可计算句子对的相似度。

1. **使用技术描述**

**2.1基本介绍**

词袋Bag-of-words（BOW）模型是n-gram语法模型的特例1元模型，是自然语言处理和信息检索领域一种常用的文档表示方法，词袋模型忽略了文档中文本的语法、语序和语意等要素，将文本看作是一个无序的词汇的集合，每个单词的出现都是独立的。词袋模型较为简单方便易用，在垃圾邮件过滤等领域有着很好的应用。



**2.2词袋模型实现方式**

**2.2.1 0/1向量One-Hot Encoding**

只要在词典中出现的词，无论出现多少次，在BOW向量中都只算成1次，未出现的词算0次，然后根据词典构造只有0/1的向量。

这种方式只表现了出现了的词和未出现的词的差异，对于出现了的词之间的差异并没有很好地体现。

**2.2.2 词频向量TF**

如果是词典中出现的词，这个词语在文本中出现的频次等数值算成多少次，没有出现的词算成0次，然后根据词典构造词袋模型向量。

这种方法比起上种方法，还考虑了词语在文本出现的频次，一定程度上体现了文本的主题和语义。

**2.2.3 词频-逆文档频率向量TF-IDF**

如果是词典中出现的词，那么可以给每个词语分配一个权重，以此降低常用词的影响，给较多出现的词一个较小的权重，给较少出现的词一个较高的权重，这个词语在文本中出现的频次乘以词语的逆文档频率就算成多少次，没有出现的词算成0次，然后根据词典构造词袋模型向量。

这种方法比起上种方法，还考虑了词语在文本出现的逆文档频率，一定程度上体现了文本的独特性和语义。

**2.3 例子推导**

比如句子对：

I like basketball, I like football.

I like running, I love exercising.

**2.3.1首先提取两个句子中出现过的单词组成词袋**

{'exercising', 'I', 'football', 'like', 'love', 'running', 'basketball'}

**2.3.2然后根据词袋构建一个词典**

{'exercising': 0, 'I': 1, 'football': 2, 'like': 3, 'love': 4, 'running': 5, 'basketball': 6}

每个出现过的单词对应一个数值

**2.3.3接着根据词典中的词语在每个句子中的出现频率，获取TF向量**

在第一个句子I like basketball, I like football.中，

'exercising': 0, 出现0次

'I': 1,出现2次

'football': 2,出现1次

'like': 3, 出现2次

'love': 4, 出现0次

'running': 5, 出现0次

'basketball': 6出现1次

形成的TF向量为：[(0, 0), (1, 2), (2, 1), (3, 2), (4, 0), (5, 0), (6, 1)]->[0, 2, 1, 2, 0, 0, 1]

在第二个句子I like basketball, I like football.中，

'exercising': 0, 出现1次

'I': 1,出现2次

'football': 2,出现0次

'like': 3, 出现1次

'love': 4, 出现1次

'running': 5, 出现1次

'basketball': 6出现0次

形成的TF向量为：[(0, 1), (1, 2), (2, 0), (3, 1), (4, 1), (5, 1), (6, 0)]->[1, 2, 0, 1, 1, 1, 0]

**2.3.4最后计算两个向量的余弦相似度，以得到句子对相似度**

句子1的词袋模型TF向量v1：[0, 2, 1, 2, 0, 0, 1]

句子2的词袋模型TF向量v2：[1, 2, 0, 1, 1, 1, 0]

v1、v2的数量积v1·v2=0\*1+2\*2+1\*0+2\*1+0\*1+0\*1+1\*0=6

v1的长度|v1| = =

v2的长度|v2| = =

余弦相似度= = 0.67082039

句子对相似度为0.67082039

**2.4 不足之处**

（1）、由于词袋模型会给每一个在词典中出现的词都计数，所以在较长的文本中，有可能导致词汇表和词袋向量的维度不断增大，有可能向量中存在大量的0，成为稀疏向量，计算时会耗费大量的资源和时间。

（2）、词袋模型不考虑语序，往往会导致对句子的语义信息判断不准确，导致计算句子对的相似度时产生一些误差。

**3.核心算法代码分析**

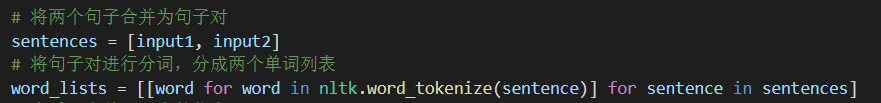
为了方便理解，使用句子对：

I like basketball, I like football.

I like running, I love exercising.

作为例子帮助解释算法代码，解释算法每一步对句子对的操作

**3.1合并两个句子，提取其中单词组成两个单词列表**



对于两个句子：

I like basketball, I like football.

I like running, I love exercising.

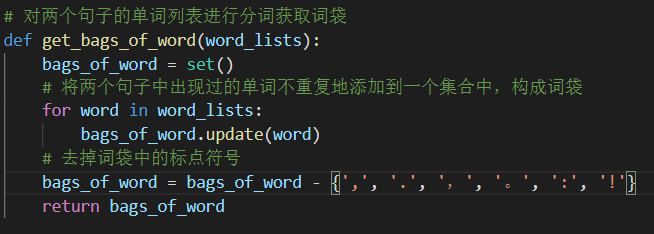
合并后句子对为：

['I like basketball, I like football.', 'I like running, I love exercising.']

对句子进行分词，分成两个单词列表：

[['I', 'like', 'basketball', ',', 'I', 'like', 'football', '.'], ['I', 'like', 'running', ',', 'I', 'love', 'exercising', '.']]

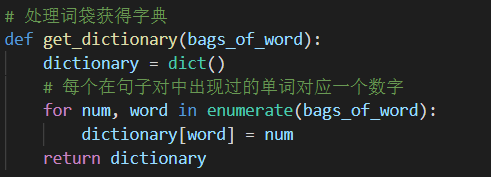
**3.2对两个句子的单词列表进行分词获取词袋**



对于之前获得的单词列表，提取其中出现过的单词，构建词袋，词袋中每个单词只能出现一次，且要去掉标点符号以增加准确性：

{'love', 'like', 'I', 'running', 'football', 'exercising', 'basketball'}

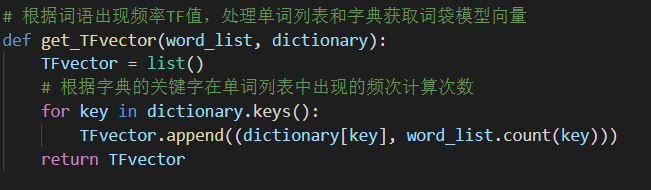
**3.3处理词袋获得字典**



对于词袋{'love', 'like', 'I', 'running', 'football', 'exercising', 'basketball'}，获取字典：

{'love': 0, 'like': 1, 'I': 2, 'running': 3, 'football': 4, 'exercising': 5, 'basketball': 6}

**3.4根据词语出现频率TF值，处理单词列表和字典获取词袋模型向量**

****

对于字典：

{'love': 0, 'like': 1, 'I': 2, 'running': 3, 'football': 4, 'exercising': 5, 'basketball': 6}

对于句子1处理到的单词列表：

['I', 'like', 'basketball', ',', 'I', 'like', 'football', '.']

获取到的词袋模型TF向量为：

[(0, 0), (1, 2), (2, 2), (3, 0), (4, 1), (5, 0), (6, 1)]->[0, 2, 2, 0, 1, 0, 1]

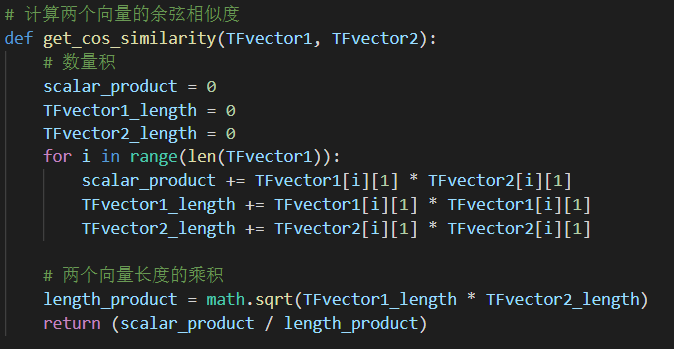
对于句子2处理到的单词列表：

['I', 'like', 'running', ',', 'I', 'love', 'exercising', '.']

获取到的词袋模型TF向量为：

[(0, 1), (1, 1), (2, 2), (3, 1), (4, 0), (5, 1), (6, 0)]->[1, 1, 2, 1, 0, 1, 0]

**3.5计算两个向量的余弦相似度**

****

句子1的词袋模型TF向量v1：[0, 2, 2, 0, 1, 0, 1]

句子2的词袋模型TF向量v2：[1, 1, 2, 1, 0, 1, 0]

v1、v2的数量积v1·v2=0\*1+2\*1+2\*2+0\*1+1\*0+0\*1+1\*0=6

v1的长度|v1| = =

v2的长度|v2| = =

余弦相似度= = 0.67082039

句子对相似度为0.67082039

**4.实验结果分析**

在控制台打印算法对句子对操作的各个步骤，实现结果分析

**4.1测试句子对1**

**4.1.1句子对**

I like basketball, I like football.

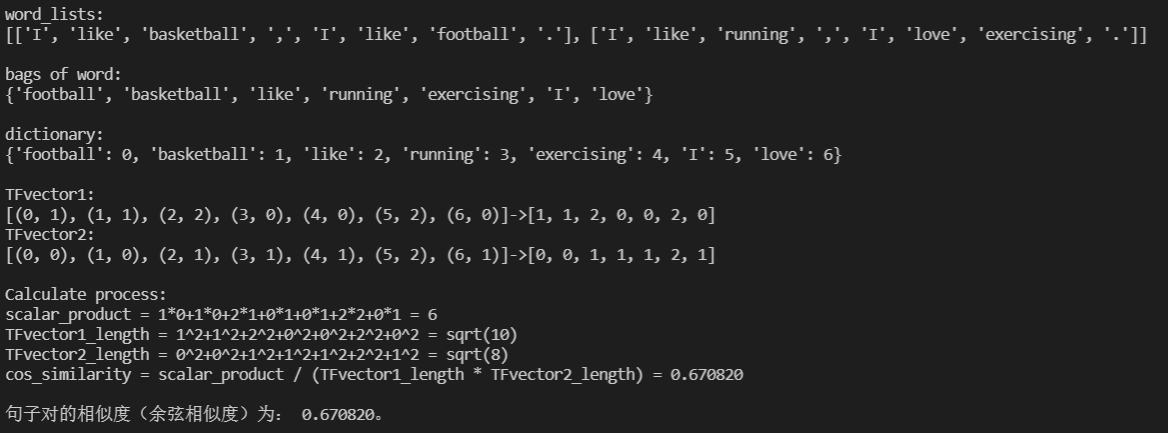
I like running, I love exercising.

**4.1.2执行截图**



**4.1.3分析**

在控制台打印算法各步骤：



两个句子的含义分别是我喜欢篮球，我喜欢足球、我喜欢跑步，我爱锻炼，语义上说都表达了自己所喜欢的东西，但是两个句子表达的喜欢的东西并没有什么关联，所以算法给出的相似度是0.670820左右，是一个一般强的相似度。

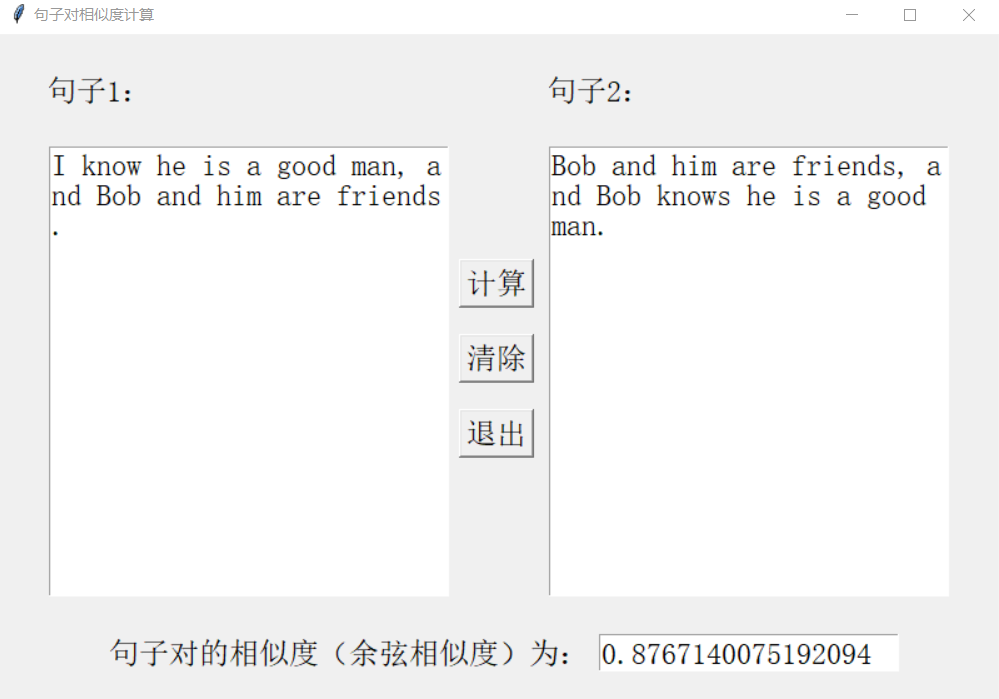
**4.2测试句子对2**

**4.2.1句子对**

I know he is a good man, and Bob and him are friends.

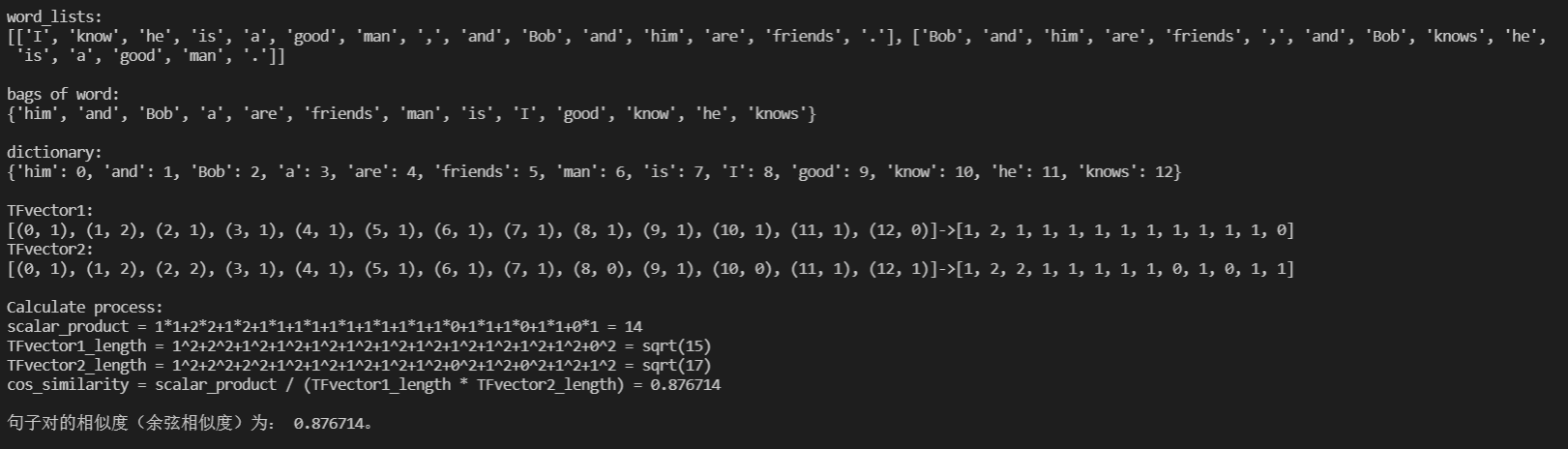
Bob and him are friends, and Bob knows he is a good man.

**4.2.2执行截图**



**4.2.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我知道他是个好人，Bob和他是朋友、Bob和他是朋友，Bob知道他是个好人，语义上说都表达了他是个好人，Bob和他是朋友，两个句子较为相似，所以算法给出的相似度是0.876714左右，是一个非常强的相似度。

**4.3测试句子对3**

**4.3.1句子对**

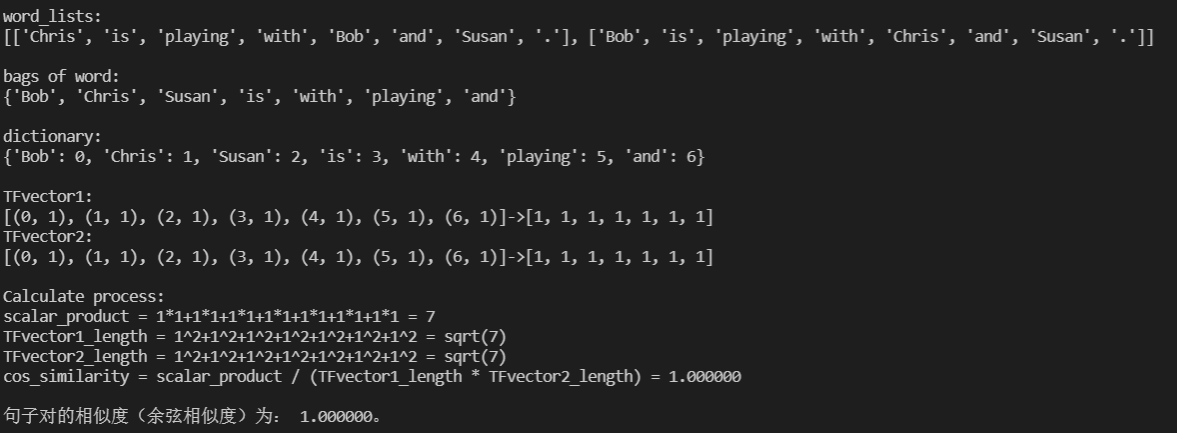
Chris is playing with Bob and Susan.

Bob is playing with Chris and Susan.

**4.3.2执行截图**



**4.3.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是Chris正在和Bob和Susan一起玩、Bob正在跟Chris和Susan一起玩，语义上说意思都是Chris、Bob和Susan三个人正在一起玩，两个句子基本是同一个意思，算法给出的相似度是1.0，相似度基本一模一样。

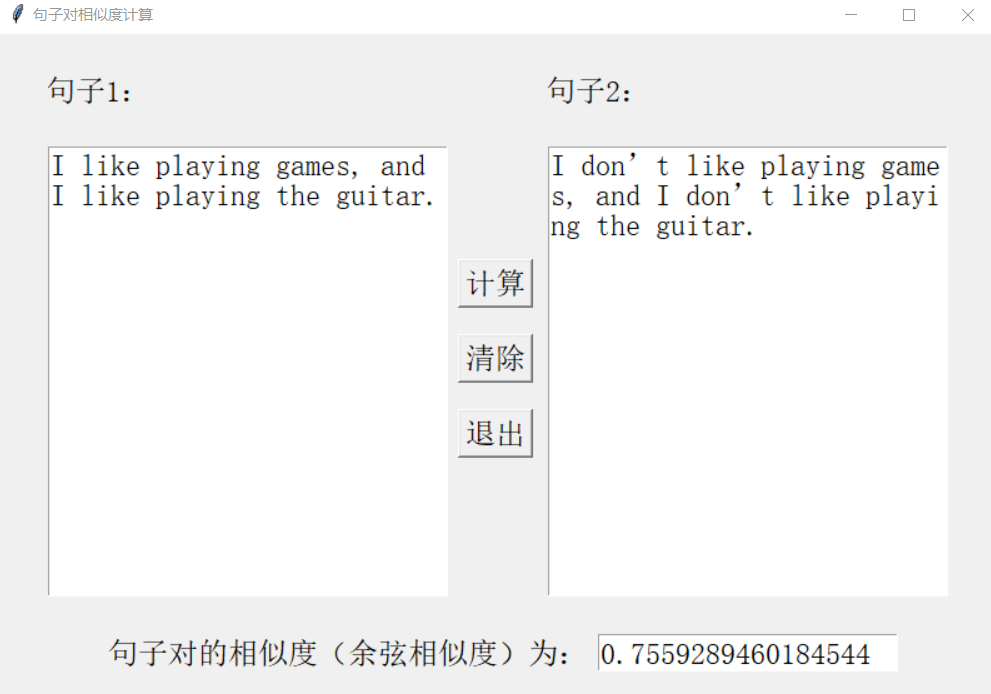
**4.4测试句子对4**

**4.4.1句子对**

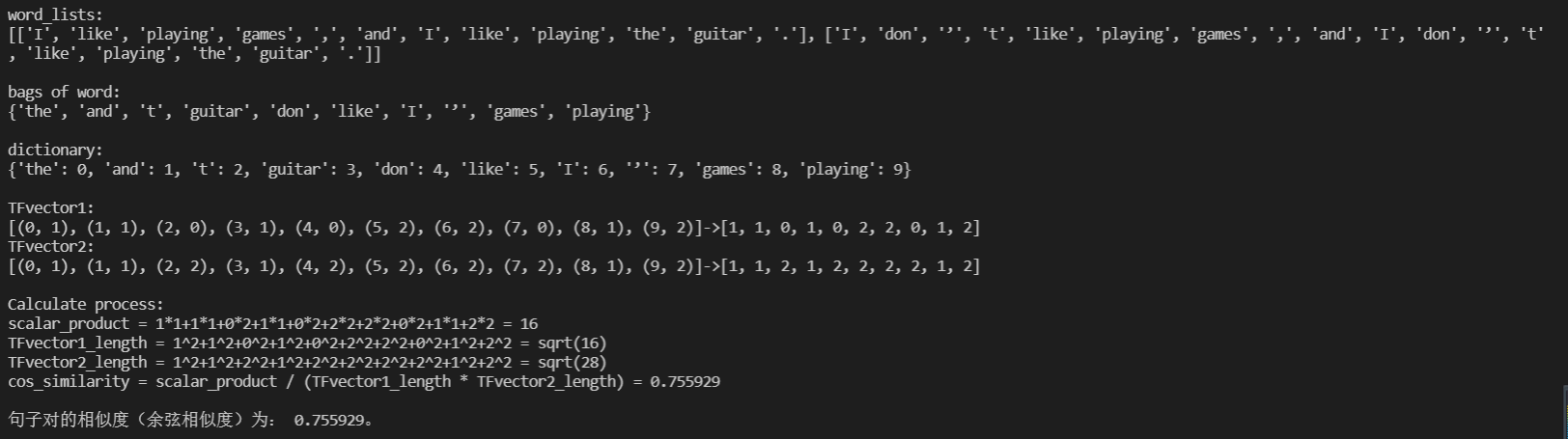
I like playing games, and I like playing the guitar.

I don’t like playing games, and I don’t like playing the guitar.

**4.4.2执行截图**



**4.4.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我喜欢玩游戏，也喜欢玩吉他、我不喜欢玩游戏，也不喜欢玩吉他，语义上说意思完全相反，但是都表达了自己对玩游戏和玩吉他两种事情的态度，用到的词语和语法结构也基本一样，所以算法给出了0.755929的相似度，是一个较高的相似度，说明算法并不依赖语义判断两个句子的相似度，否则这个例子的两个语句的相似度应该非常低。

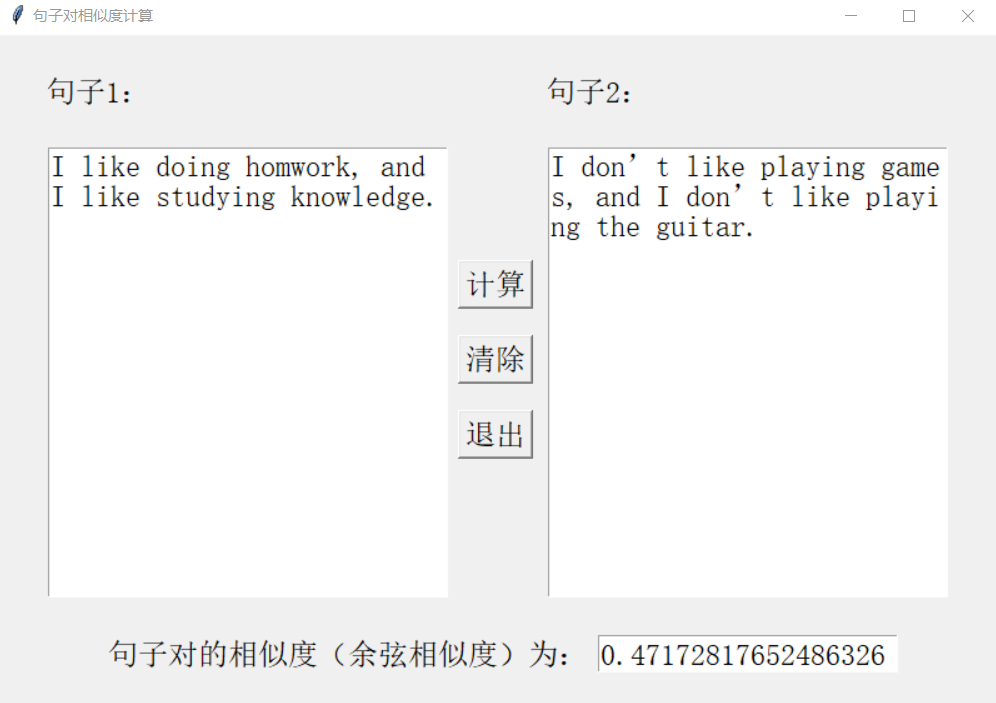
**4.5测试句子对5**

**4.5.1句子对**

I like doing homework, and I like studying knowledge.

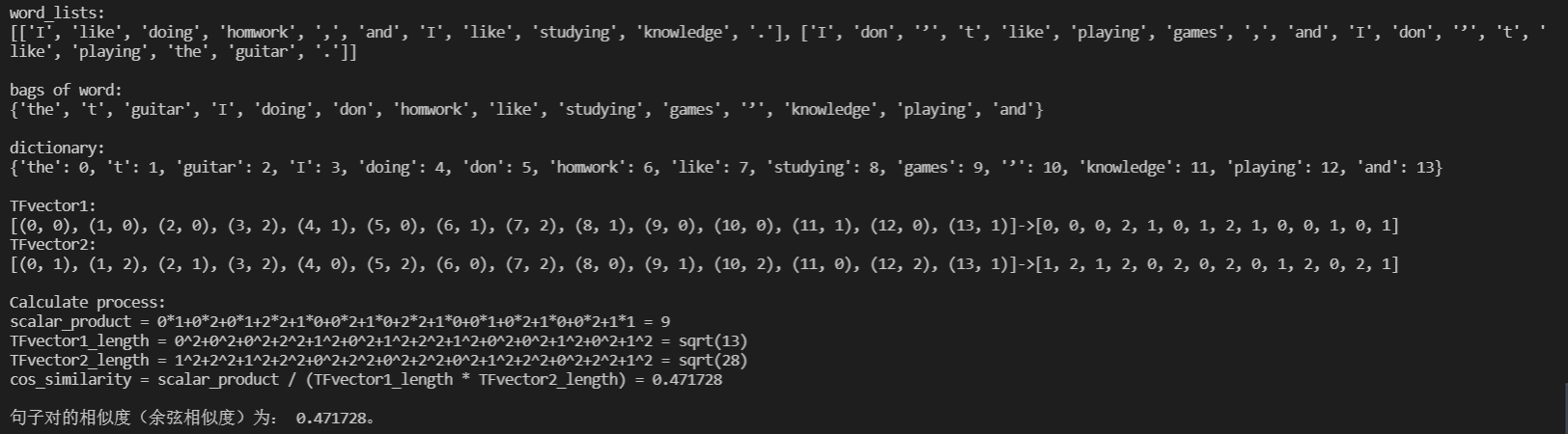
I don’t like playing games, and I don’t like playing the guitar.

**4.5.2执行截图**



**4.5.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我喜欢做作业，也喜欢学知识、我不喜欢玩游戏，也不喜欢玩吉他，语义上说意思基本没有关系，和之前一个例子不同的是，两个句子所描述的事物也基本没有关系，只是都表达了自己对某些事物的态度，只有表达时的语法结构差不多，所以算法给出了0.471728的相似度，是一个比较低的相似度，比之前一个描述了相同事物的例子也要低。

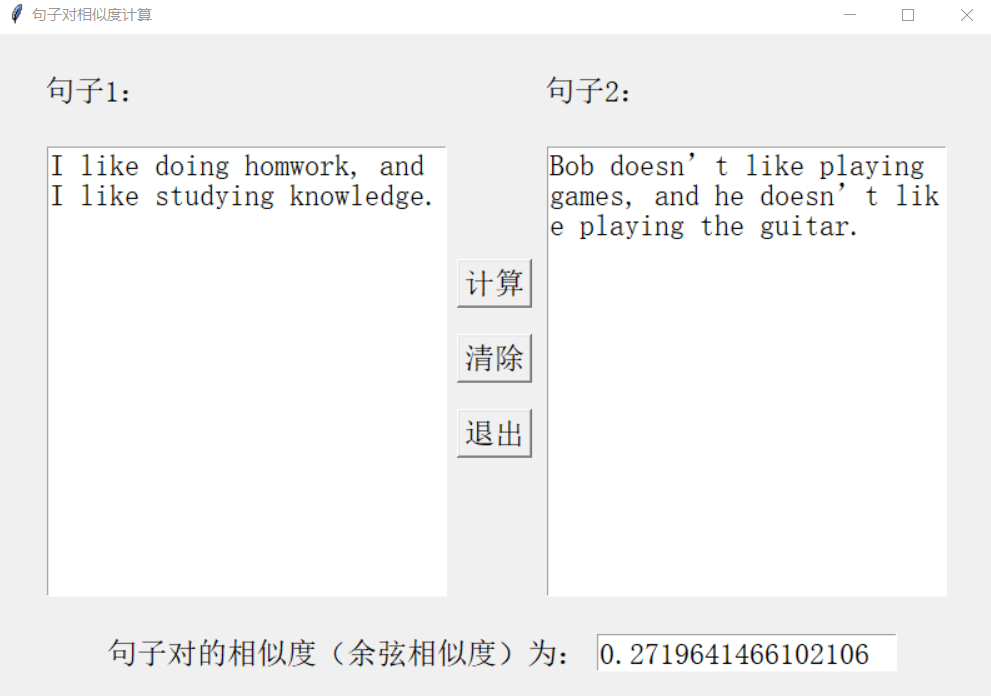
**4.6测试句子对6**

**4.6.1句子对**

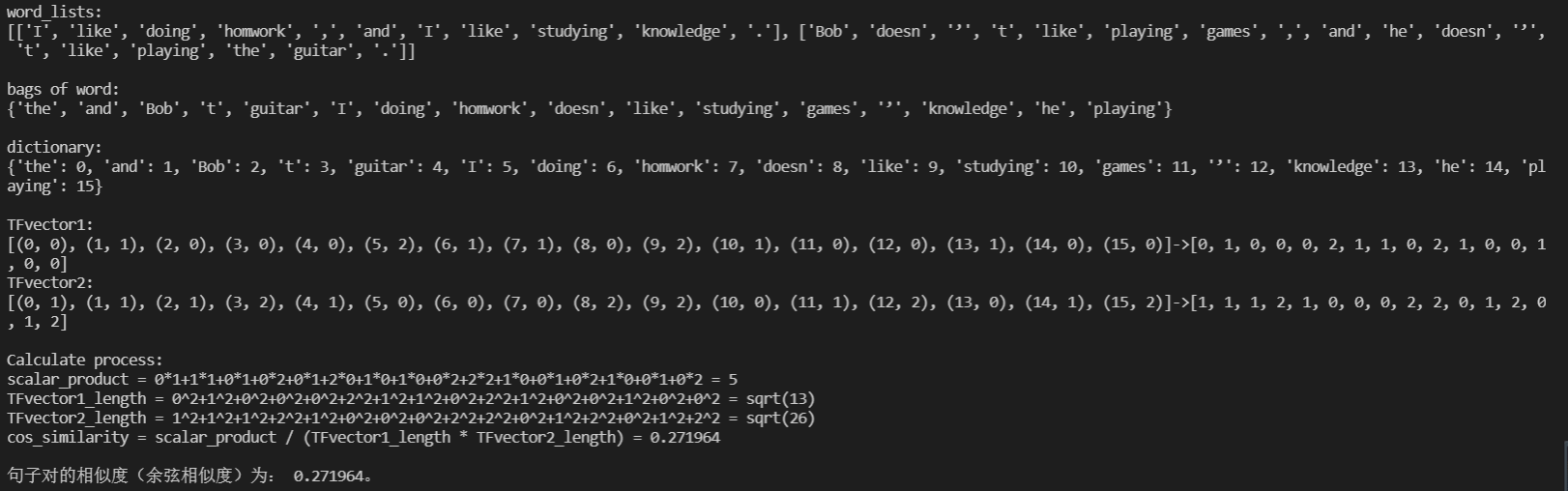
I like doing homework, and I like studying knowledge.

Bob doesn’t like playing games, and he doesn’t like playing the guitar.

**4.6.2执行截图**



**4.6.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我喜欢做作业，也喜欢学知识、Bob不喜欢玩游戏，也不喜欢玩吉他，语义上说意思基本没有关系，两个句子所描述的事物也基本没有关系，和之前一个例子不同的是，表达态度的人也基本没有关系，只是都表达了对某些事物的态度，只有表达时的语法结构差不多，所以算法给出了0.271964的相似度，是一个非常低的相似度，比之前一个表达主体相同的例子也要低。

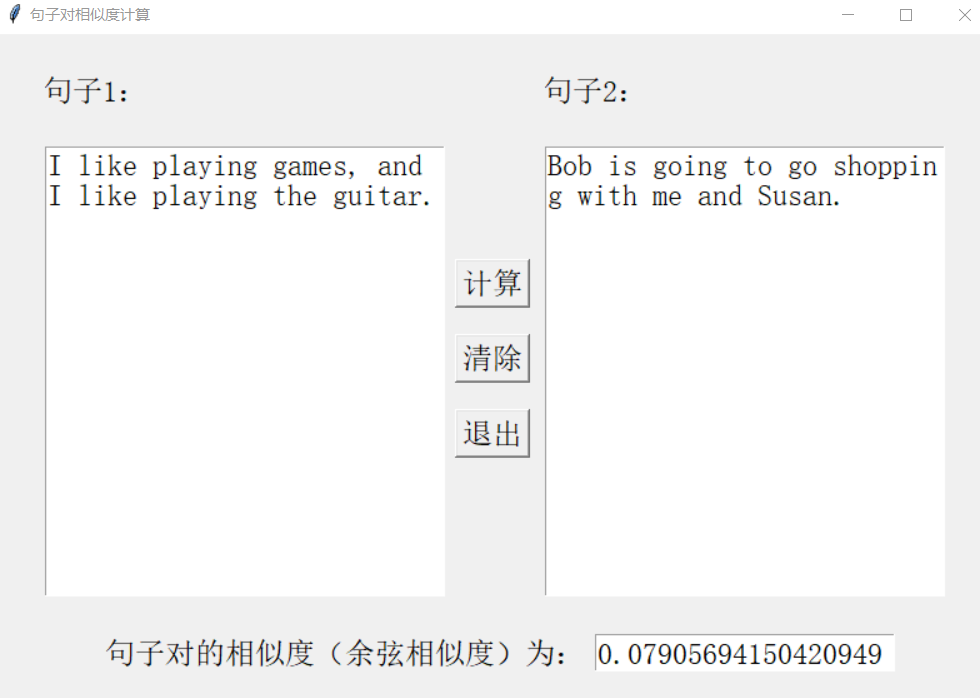
**4.7测试句子对7**

**4.7.1句子对**

I like playing games, and I like playing the guitar.

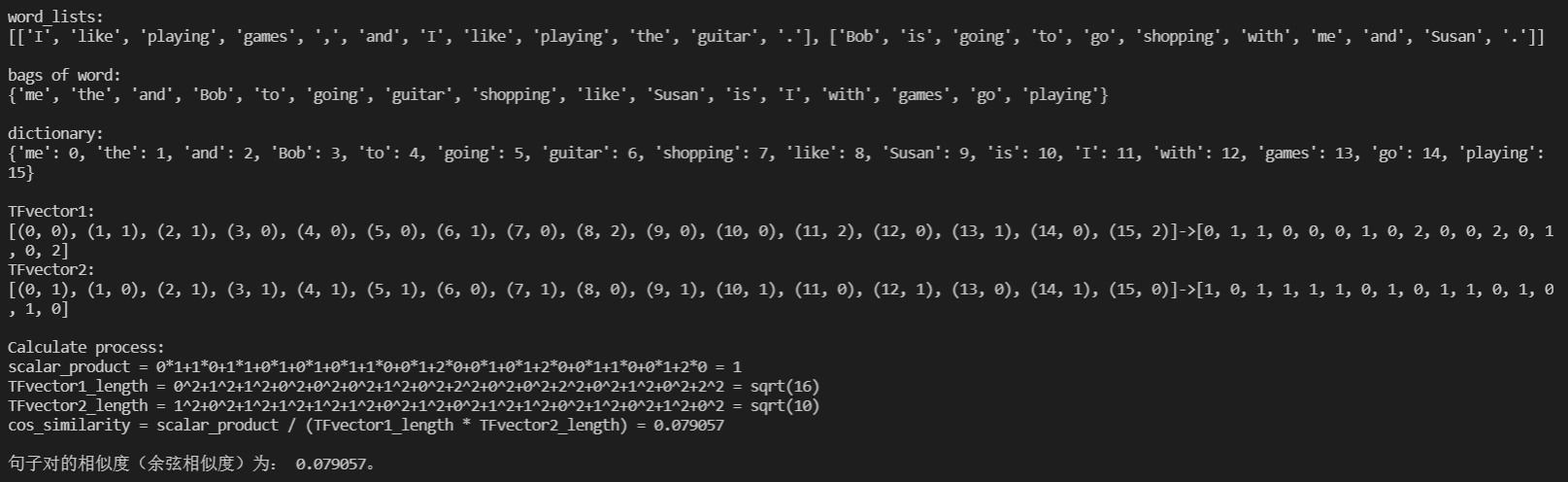
Bob is going to go shopping with me and Susan.

**4.7.2执行截图**



**4.7.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我喜欢玩游戏，也喜欢玩吉他、Bob将要和我和Susan去购物，无论是从语义、时态、表达主体、语法结构来说都几乎没有任何关系，是两个完全不一样的句子，所以算法给出了0.079057的相似度，说明几乎没有相似度。

**4.8测试句子对8**

**4.8.1句子对**

I often played football with him in the past, having some fun in company.

He used to play soccer with me from time to time, enjoying our day together.

**4.8.2执行截图**



**4.8.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我以前时常和他一起去踢足球，享受乐趣、他以前时常和我去踢足球，享受时光，两个句子的语义，时态，语法结构都基本相同，两个句子基本相似，但是算法只给出了0.057354的非常低相似度，说明词袋模型不能识别具有同一个意思的不同的词语，对语义识别存在困难，有一定缺陷。

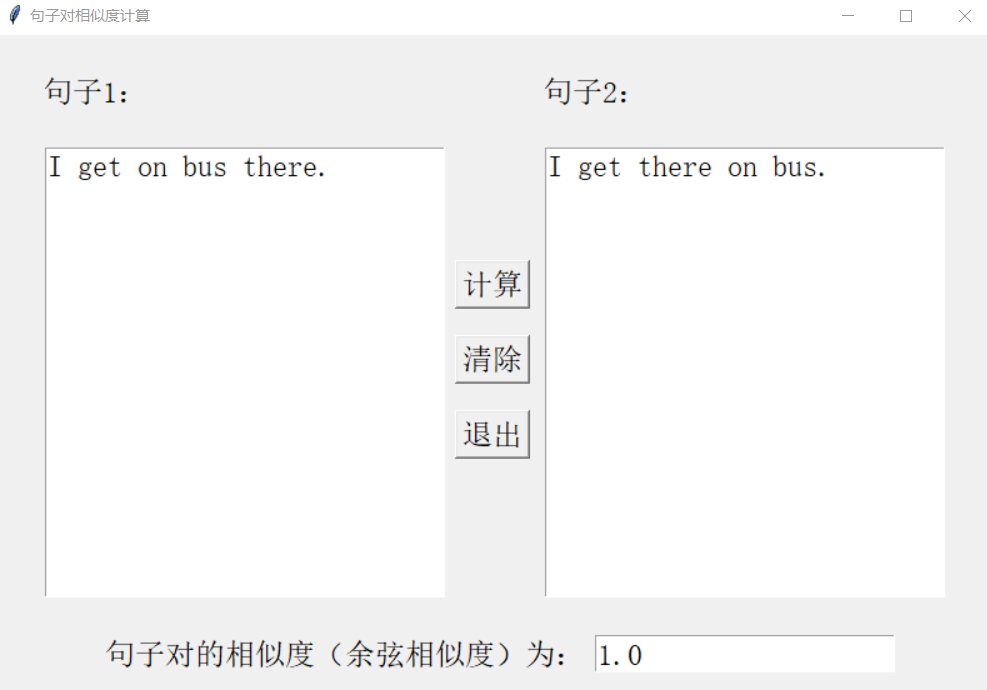
**4.9测试句子对9**

**4.9.1句子对**

I get on bus there.

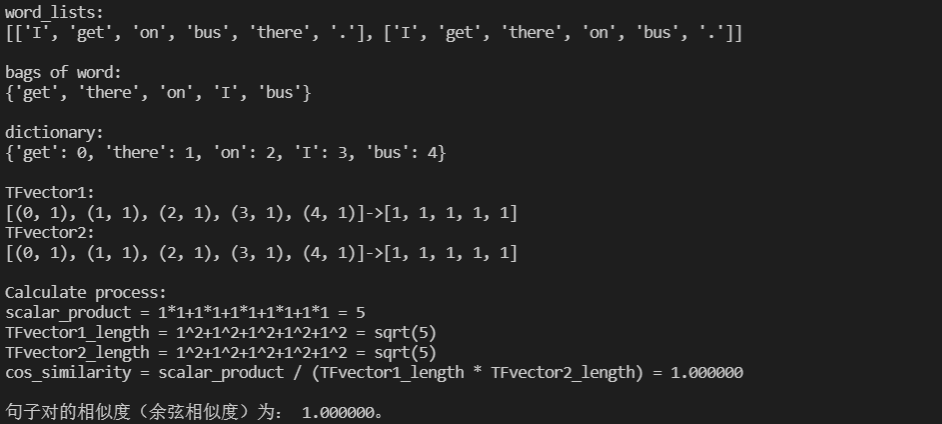
I get there on bus.

**4.9.2执行截图**



**4.9.3分析**

在控制台打印算法各步骤：

两个句子的含义分别是我搭上了那里的巴士、我坐巴士到那，两个句子的语义完全不同，相似性不高，但是算法给出了1的相似度，说明词袋模型不能识别语序，只统计出现过的单词，两个句子出现的单词完全一样，所以算法给出了1的相似度，认为几乎完全一样，不能识别语序是词袋模型的一个缺陷。

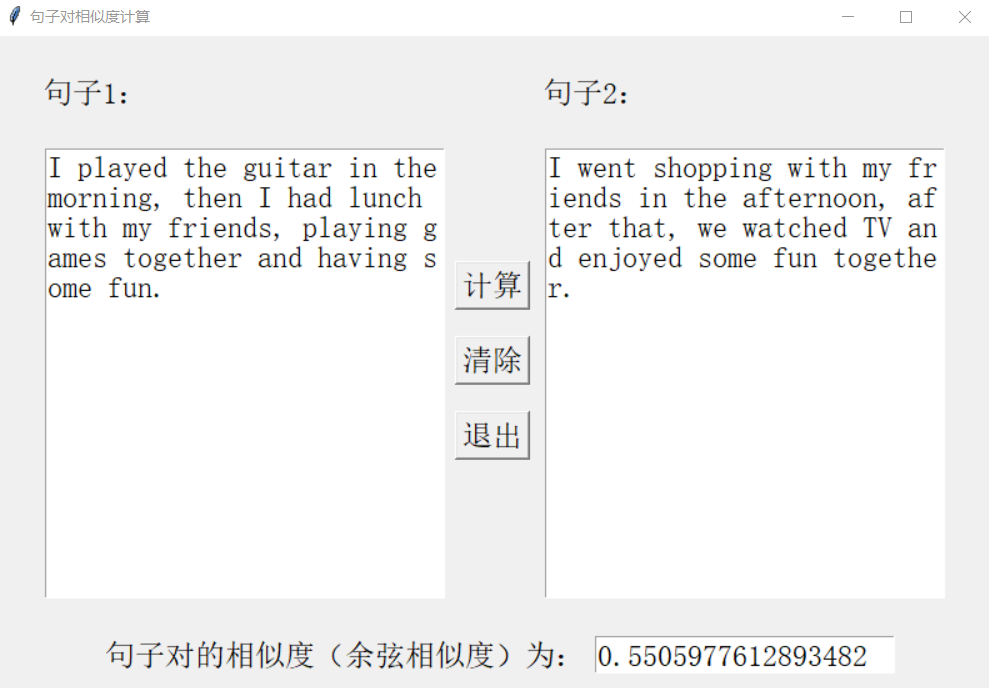
**4.10测试句子对10**

**4.10.1句子对**

I played the guitar in the morning, then I had lunch with my friends, playing games together and having some fun.

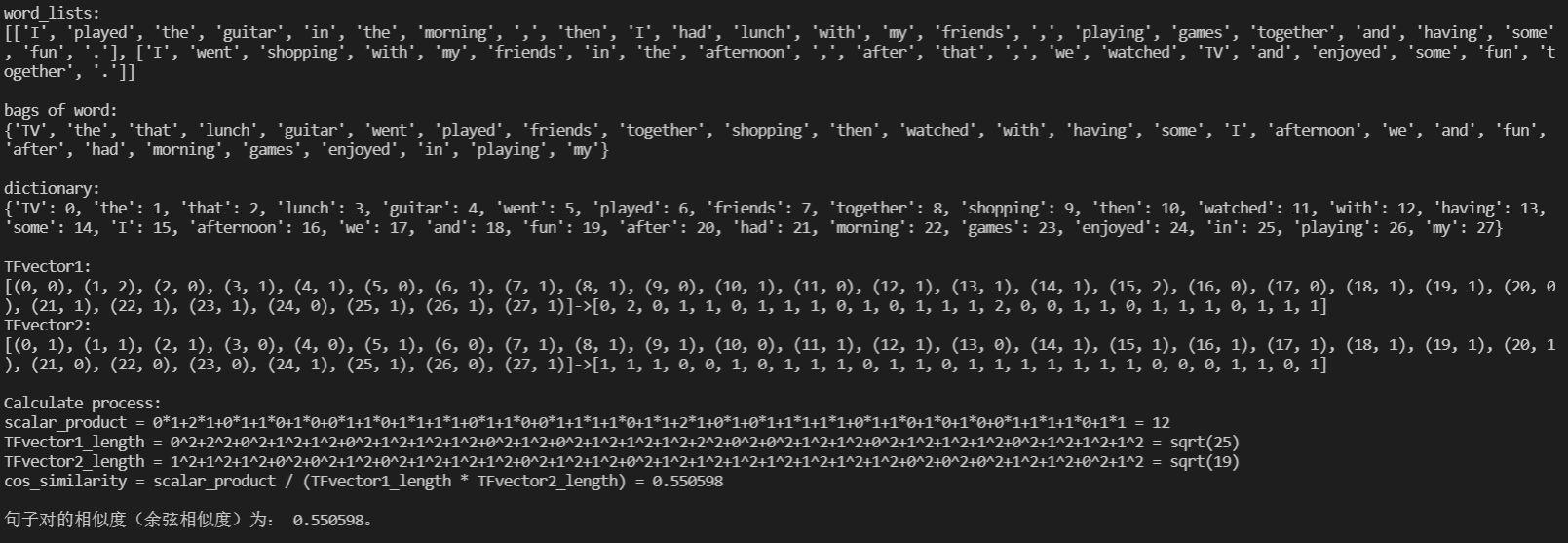
I went shopping with my friends in the afternoon, after that, we watched TV and enjoyed some fun together.

**4.10.2执行截图**



**4.10.3分析**

在控制台打印算法各步骤：

随机长句测试，算法认为这两个句子的相似度为0.550598

**4.11测试句子对11**

**4.11.1句子对**

Do

空

**4.11.2执行截图**



**4.11.3分析**

当两个输入框中只有一个输入框为空时，默认相似度为0

**4.12测试句子对12**

**4.12.1句子对**

空

空

**4.12.2执行截图**



**4.12.3分析**

当两个输入框中都为空时，默认相似度为1