**第 一 次 实 验 报 告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 课程名称 | 内容安全实验 | | | | |
| 学生姓名 |  | 学号 |  | 指导老师 | 熊翘楚 |
| 专业 | 网络空间安全 | 班级 |  | 实验时间 | 2022.03.28 |

**一、实验内容**

1、基于word2vec模型的文本分类任务：文本素材可使用爬虫爬取或使用网上现成语料库。

注：实验报告中应写出所使用的算法基本原理

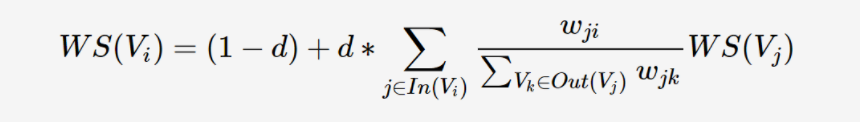
2、调用百度AI开放平台的文本内容审核API，完成自定义数据集的内容审核。

**二、实验原理**

基于word2vec模型的文本分类任务的主要思路是先将所有文本进行分词处理，然后使用TextRank算法对其中的动名词进行排序，获取词表后将其输入至word2vec模型中训练，从而得到词表中每个词相应的特征向量。然后按照词频对每个类别中的词语进行排序。在每个类别中取k个高频词（k属于超参数）作为该类别的关键词，将这些词的向量取平均即为该类别的特征向量。当模型预测一篇文本的所属类别时，会按照同样的方法取该文本中的k个高频词，取平均作为该文本的特征向量，再分别计算该特征向量与每个类别的特征向量的余弦相似度，相似度最高的类别即为该文本的预测所属类别。

在实验中使用到了TextRank算法和word2vec模型，这两个算法或模型在本实验中起到了关键作用。

TextRank算法的思想是通过词之间的相邻关系构建网络，然后用类似PageRank算法的公式迭代计算每个节点的权重值，最后按照权重值排序得到关键词。PageRank依据超链接计算转移概率，即网络结构中的边，而TextRank按照任意两个句子之间的相似度计算转移概率。PageRank计算公式如下图所示。



其中PR(Vi)表示节点Vi的权重值，In(Vi)表示节点Vi的前驱节点集合，Out(Vj)表示节点Vj的后继节点集合。d是阻尼系数，用于平滑。Wji表示两个节点之间的边的权重，也可以说是重要程度。

TextRank算法的步骤可以概括为以下六个步骤：首先，将给定文本分成多个句子；其次，对每个句子进行分词处理，同时进行词性标注，去掉停用词，保留需要的词性所对应的词；然后开始构建网络，两个节点之间存在边仅当这两个节点对应的词语在长度为K的窗口中同时出现；按照上述公式迭代计算每个节点的权重值；按照权重值对节点进行排序，从而得到最重要的k个关键词；如果其中存在关键词在文本中相邻，则将这些关键词组合起来形成一个关键词。

word2vec模型是词嵌入方法的一种，用于生成词向量，即将自然语言转换为数学形式的向量。word2vec模型的网络结构比较简单，包括输入层、隐藏层和输出层。模型框架主要包括CBOW和Skip-gram模型，前者是已知词的上下文预测当前词语，后者是已知当前词语预测该词语的上下文。

在使用一个训练好的word2vec模型时，我们可以把输入词语以独热码形式输入进模型中，这个词语的独热码在模型中能够激活隐藏层到输出层的权重，从而组成一个新的向量，这个向量能够唯一表示这个词语，因此可以认为该向量是这个词语的特征向量。独热码的维度一般而言是相当大的，而按照隐藏层的权重生成的特征向量的维度与隐藏层的节点数量一致，因此word2vec也可以看作是对词向量进行了降维操作。

从word2vec模型的输入和输出的角度来看，这个模型的作用是根据上下文预测词语，然而我们在实际应用中往往将word2vec模型用于词嵌入，只用到了word2vec模型的隐藏层权重，也就是说在使用word2vec模型进行词嵌入时，我们并不关心模型的准确率等性能如何，只关心模型中的隐藏层的权重，这是word2vec模型词嵌入的关键。

**三、实验步骤**

**基于word2vec模型的文本分类任务**

**1.安装并导入工具包**

安装并导入本实验使用到的工具包，主要有gensim包，jieba包，numpy包和re包。

使用如下命令可以安装gensim包，将命令中的gensim替换为其他包名可以安装相应的包。在python代码中使用import语句导入工具包。

pip install genism

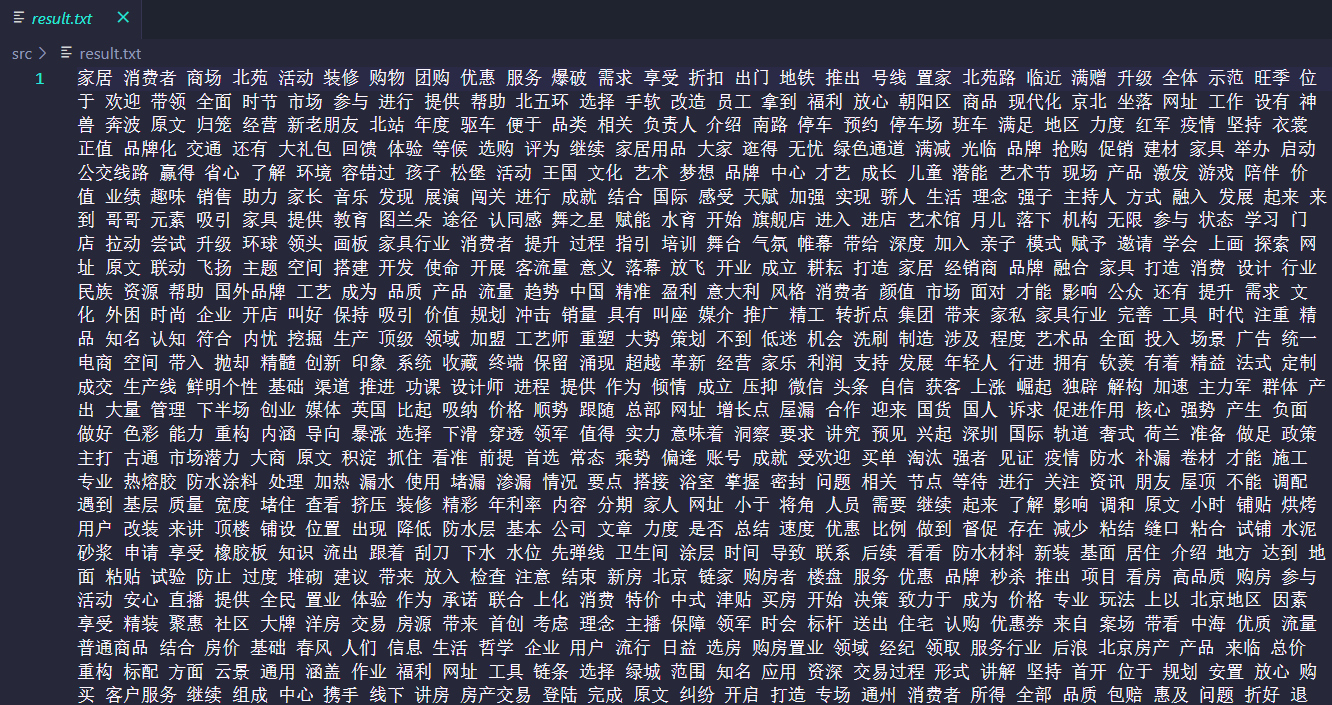
**2.获取词表**

首先需要读取数据集，本实验使用的数据存放在data目录下，且data目录下是各个文本分类的目录，在这些目录中存放着许多文本文档，每个文本文档存放着一篇中文文本，比如一篇财经类文本的相对路径是data/财经/1.txt。

根据上述数据集的结构，可以使用如下代码读取所有中文文本，并且每读取一篇就对该文本进行如下操作：去除文本中的空格、缩进和换行，再使用jieba.analyse.textrank方法获取该文本中权重最大的关键词，默认获取20个关键词，关键词词性为地名、名词、动名词、动词。将这些关键词全部整合起来，从而获取了一个word2vec的词表。最后将词表写入至result.txt文件中，使用空格分隔不同关键词，便于后续取用词表。

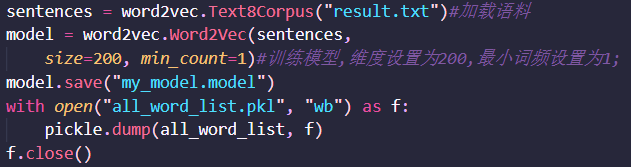


result.txt文件中的内容如图所示，保存了数据集中所有文本的关键词。



**3.训练模型**

先使用word2vec.Text8Corpus方法加载预料，再调用word2vec.Word2Vec方法训练模型，其中词向量维度设为200，最小词频设置为1。将训练后的模型保存至文件my\_model.model中，并将关键词列表序列化保存至文件all\_word\_list.pkl中。



**4.统计词频**

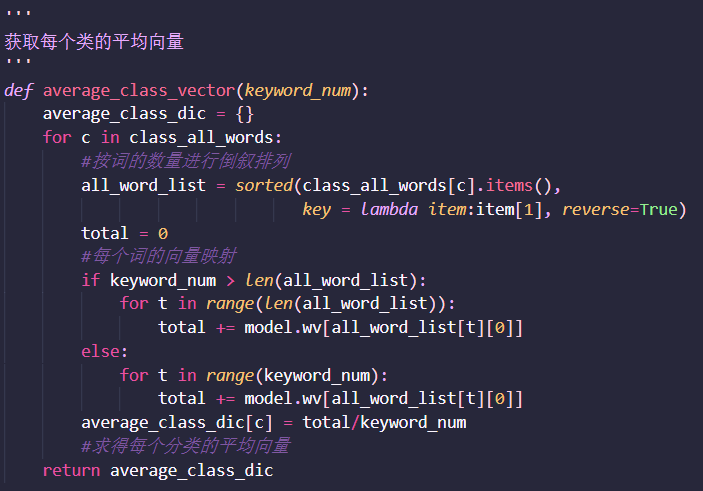
在实验代码中将统计词频的工作交由函数train\_f完成，该函数取每个类别的70%文本作为训练集，剩余30%的文本用于后续的测试集。对这70%的文本执行与上述第二部相似的操作，读取每篇文本并去除空格、缩进和换行符，再调用jieba.analyse.textrank方法获取文本中权重值最高的20个名词和动词。最后记录这些关键词的词频，并将数据集中所有文本的关键词的词频存入class\_all\_words字典中，每个键值对的键是词语，值是该词的词频。成功统计词频后，将class\_all\_words字典序列化保存至all\_words\_practice.pkl文件中，以便后续直接取用。该函数的返回值是class\_all\_words字典。



**5.获取特征向量**

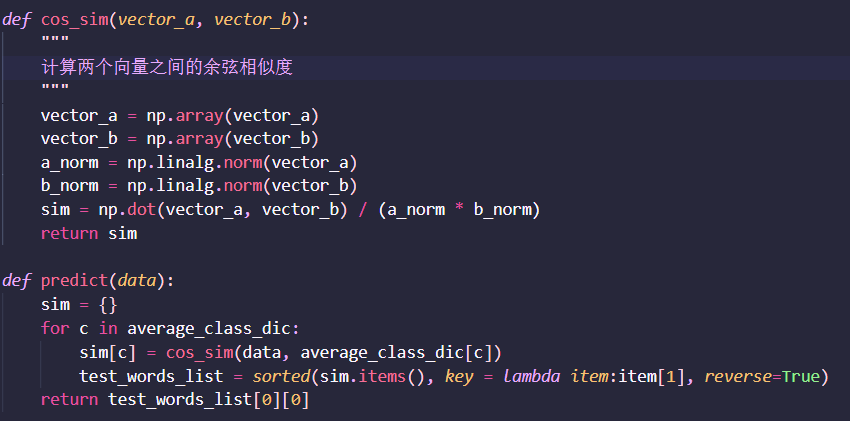
该步骤的整体思路是对于每个类别，取其中前k个高频词作为关键词，通过训练好的word2vec模型获取这些关键词的向量，再对这些向量求平均即为对应类别的特征向量。

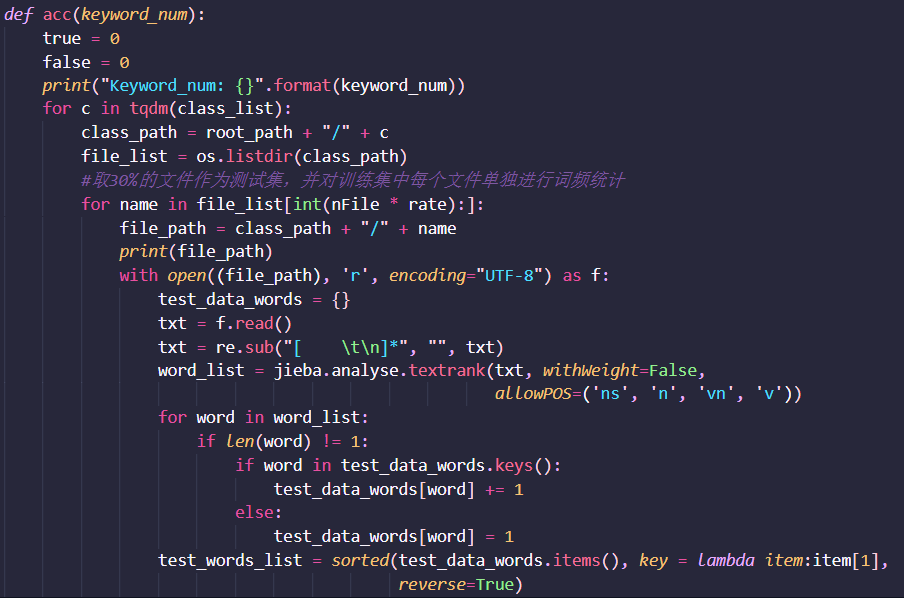
在实验代码中，首先将每个类别的词频字典按照词频进行倒序排列，然后根据参数keyword\_num分为两种情况。参数keyword\_num指的是在每个类别中获取的高频词数量，如果这个值大于该类别的所有关键词数量，那么就对该类别的所有关键词的向量求平均；如果这个值不大于该类别的所有关键词数量，那么就对该类别的前keyword\_num个关键词的向量求平均。最后将每个类别的特征向量以类别为键，特征向量为值的键值对格式保存至average\_class\_dic字典中。

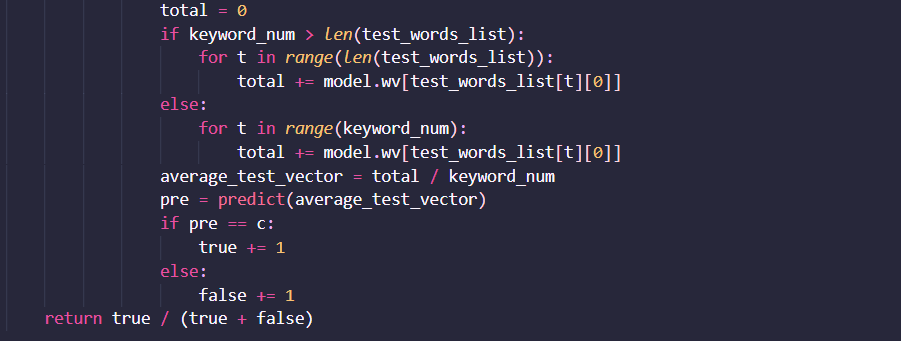


**6.准确率计算**

该步骤首先取数据集中剩余30%的文本作为测试集，对于测试集的每篇文本都需要单独统计词频，再取前keyword\_num个高频词的向量并求平均，关于keyword\_num取值不同从而可能产生的两种情况的处理方式与上述第五步相同。计算得到该文本的特征向量后，将该特征向量与上述第五步中计算得到的每个类别的特征向量计算余弦相似度，从而预测该文本的类别是相似度最高所对应的那个类别。然后将预测的类别与该文本的真实类别进行比较，从而求得最终测试集的准确率。







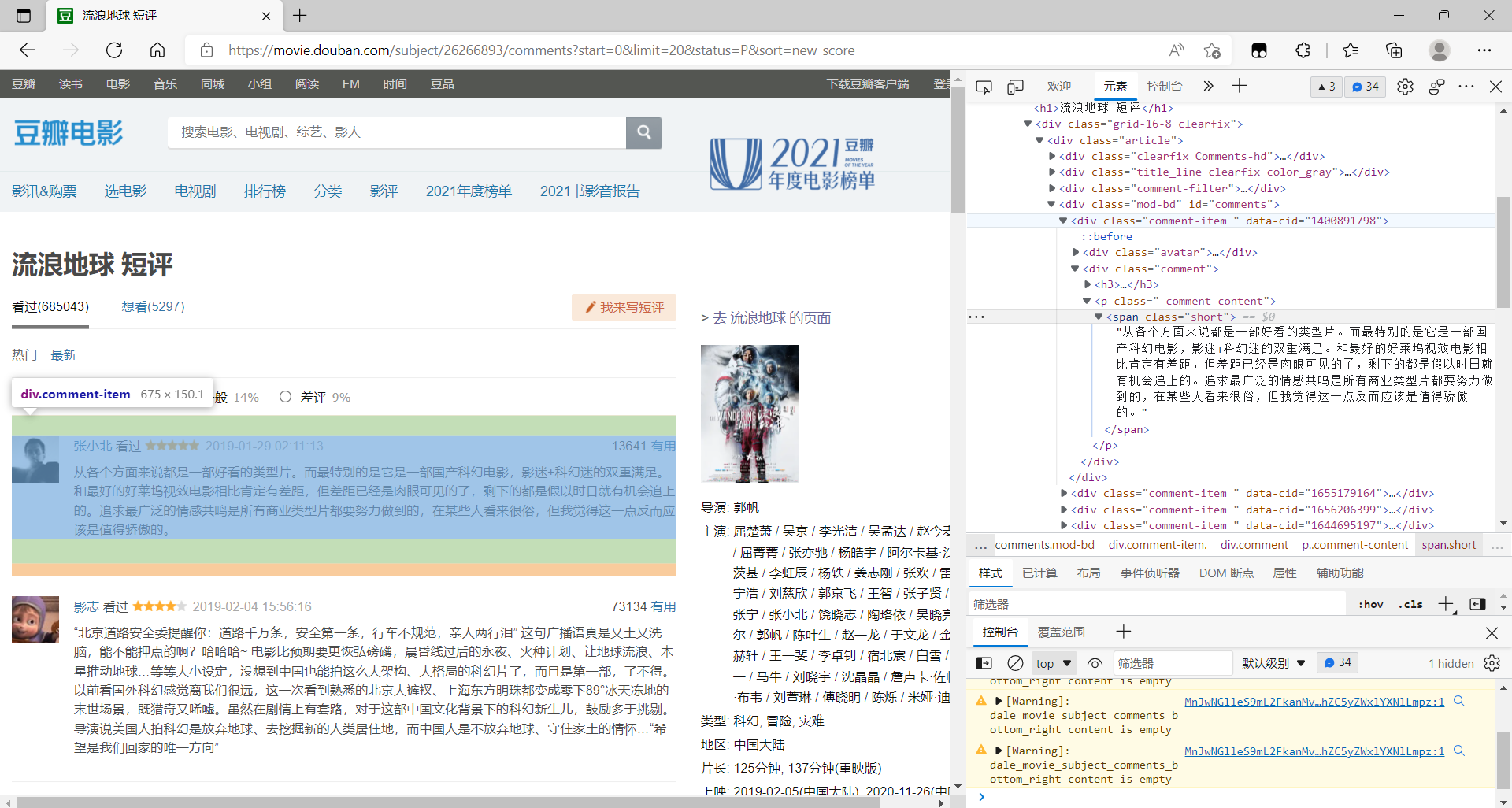
在以下代码中比较选取的关键词个数，即keyword\_num的值，与准确率的关系，并使用matplotlib.pyplot库绘制关键词个数-准确率折线图，从而将两者之间的关系可视化。



**百度文本内容审核API调用**

**1.制作自定义数据集**

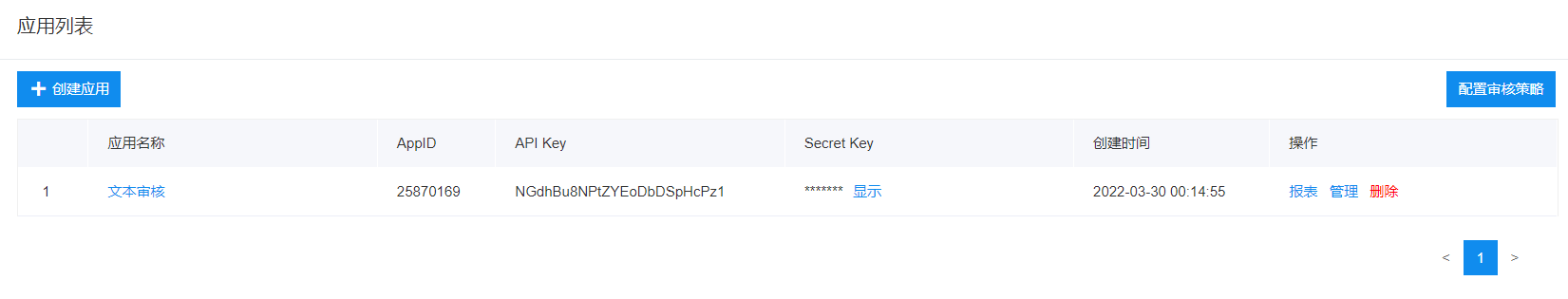
使用python代码爬取豆瓣电影评论，首先观察豆瓣电影评论的URL格式，以这个URL为例：<https://movie.douban.com/subject/26266893/comments?start=0&limit=20&status=P&sort=new_score，其中需要关注的是26266893>是该电影在豆瓣上的ID，get参数start的值表示该URL所包含的评论的起始序号，limit的值表示该URL所包含的评论的数量。然后根据url对应的html页面的特点，寻找html代码中包含文本的标签，从而为爬虫做好准备。



将该页面中的所有电影评论文本提取出来后，修改URL中start的值，继续爬取电影评论，直到爬取的数据量满足需求为止。爬虫代码会以截图形式附在实验报告的最后部分。

**2.调用文本内容审核API接口，完成文本内容审核任务**

首先需要使用百度账号在百度智能云平台上进行个人认证，完成认证之后申请个人免费额度，本实验需要申请“内容审核平台-文本”这个API的个人免费额度。申请成功后需要创建应用，应用创建成功后在应用列表中能够看到使用该API所需要的API Key以及Secret Key。



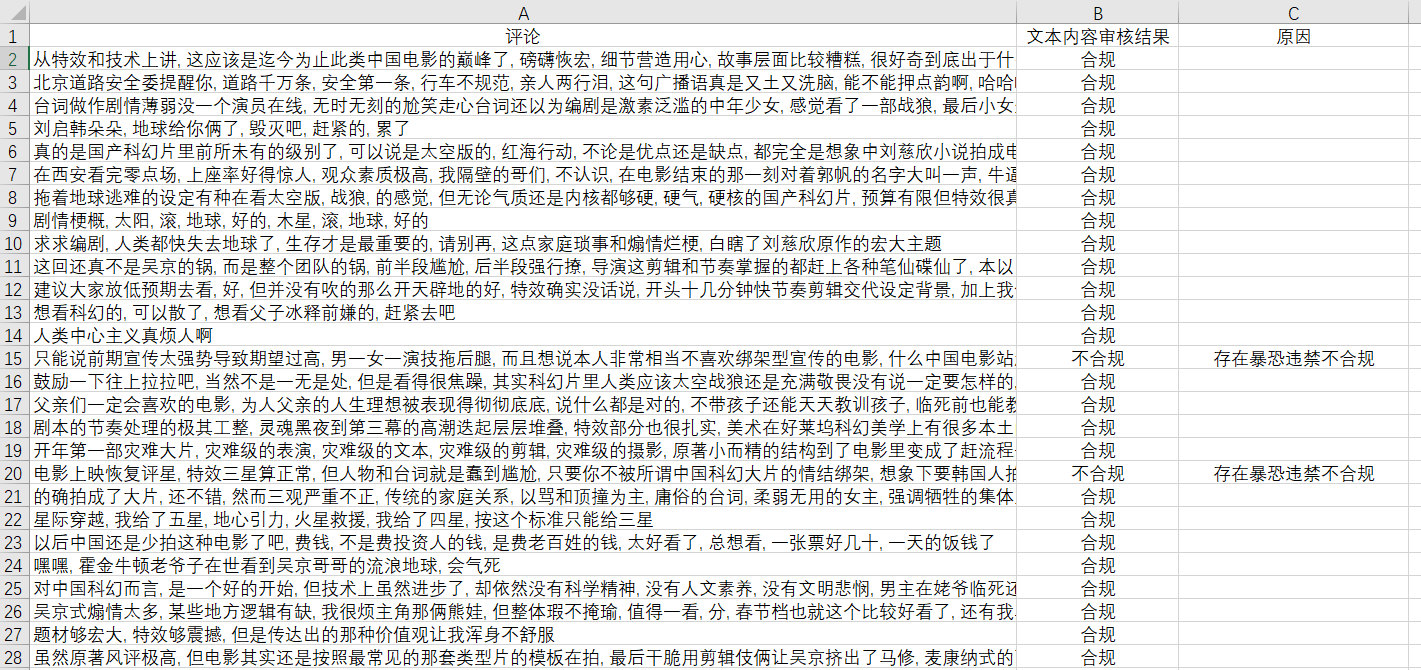
调用API使用的代码可以参考<https://github.com/Baidu-AIP/QuickStart/blob/master/CONTENT_CENSOR/main.py>中的代码，本实验只需要用到其中的文本内容审核部分。以测试文本为例，百度文本内容审核API调用的结果如下图所示。



现在将爬虫得到的数据集与百度文本内容审核API结合起来，将对每条电影评论调用一次该API，从而得到该评论是否合规的结果，如果不合规，API会返回文本不合规的原因。本实验所有步骤的代码均会以截图形式附在实验报告的最后部分。

**3.输出审核结果**

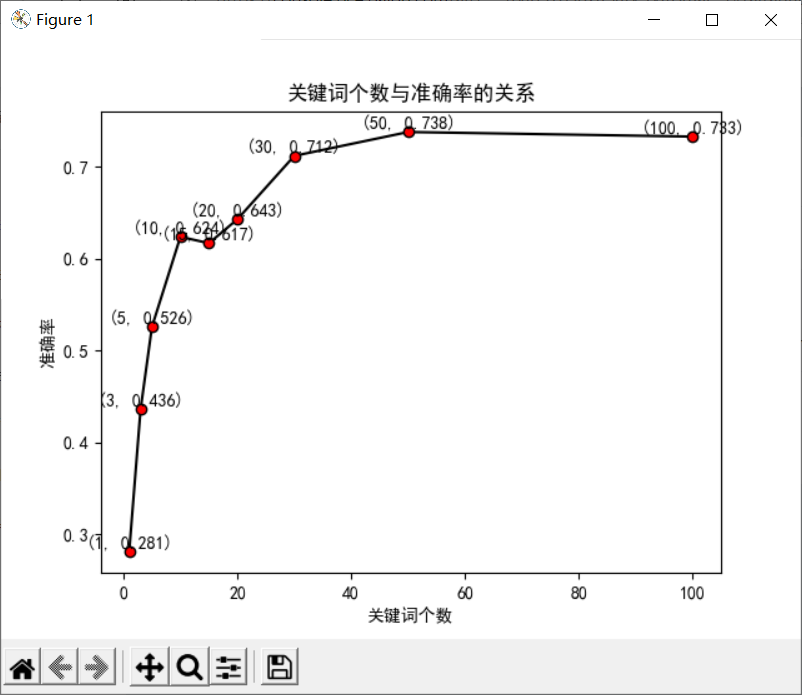
本实验中我将审核结果输出至csv文件中，以豆瓣电影评论为数据集，对于电影《流浪地球》的部分评论的文本内容审核结果如下图所示。



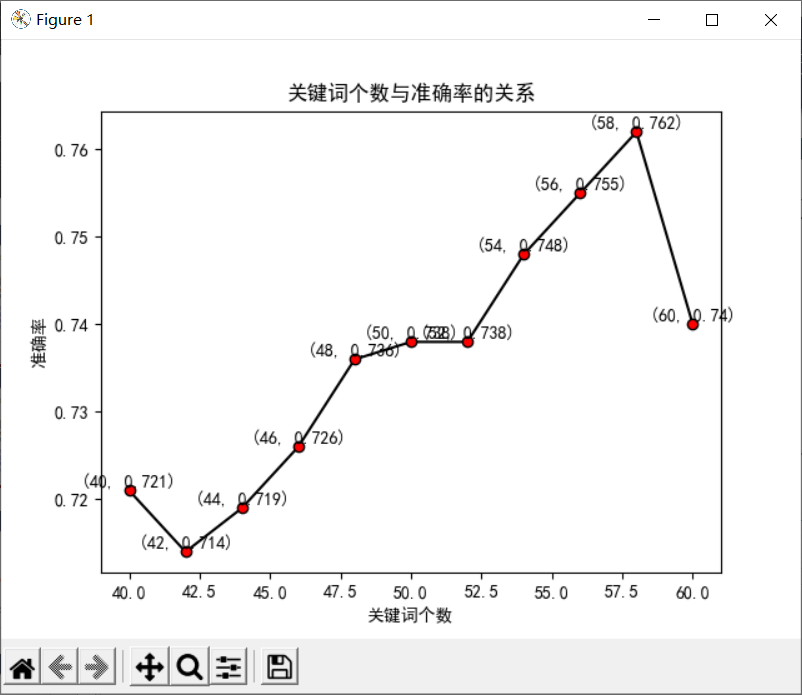
**四、实验结果**

**基于word2vec模型的文本分类任务**

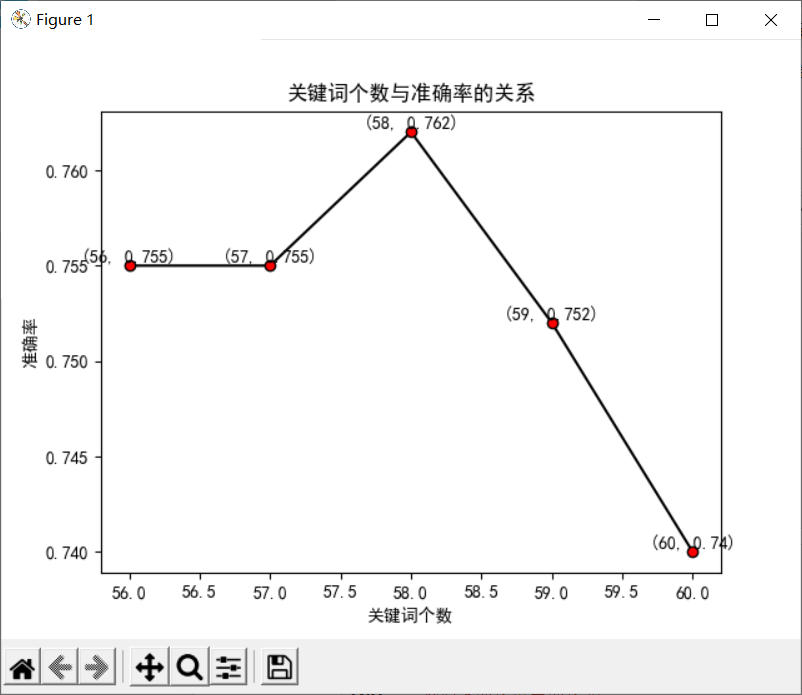
实验得到的关键词个数-准确率折线图如下图所示，其中纵轴是分类七个文档的准确率，横轴是关键词个数。可以发现随着关键词个数的增多，获得特征越多，模型预测的准确率也随之增大。当关键词个数增加到一定数量后，再增大关键词个数会形成冗余的特征，从而使得模型预测的准确率有所下降。



从上图中可以发现准确率大致在关键词个数为[40,60]这个区间中，因此修改代码中的列表keyword\_num\_list的值，分别取取keyword\_num为40、42、……、58、60这些数值，固定随机数，再次计算这些数值对应的准确率，同时生成相应的关键词个数-准确率折线图。从下图中可以看出当关键词个数的值是58时，准确率最大，值为76.2%。

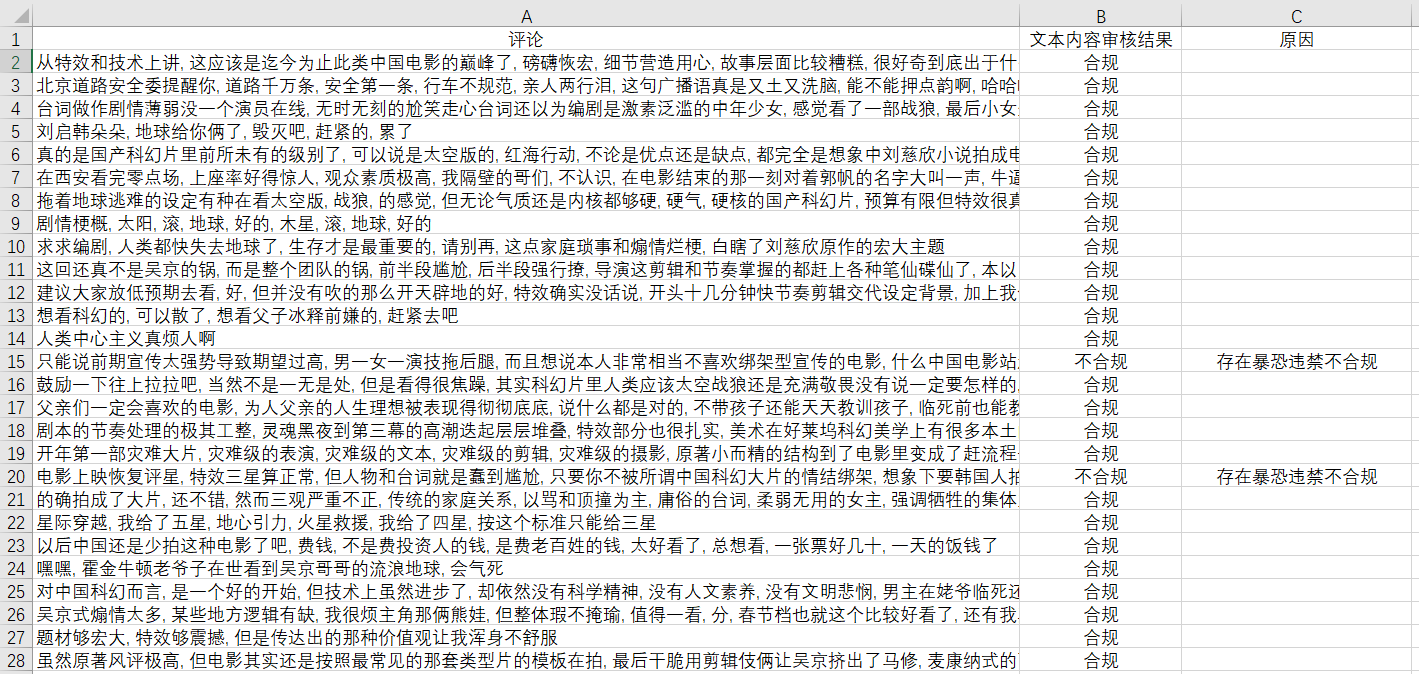


进一步缩小关键词个数的值之间的间隔，分别取keyword\_num为56、57、58、59、60这些数值，固定随机数，再查看相应的关键词个数-准确率折线图。如下图所示，可以得出结论该模型在其他参数不变的情况下，keyword\_num取值为58时，在给定测试集上的准确率最高，准确率的值为76.2%。



**百度文本内容审核API调用**

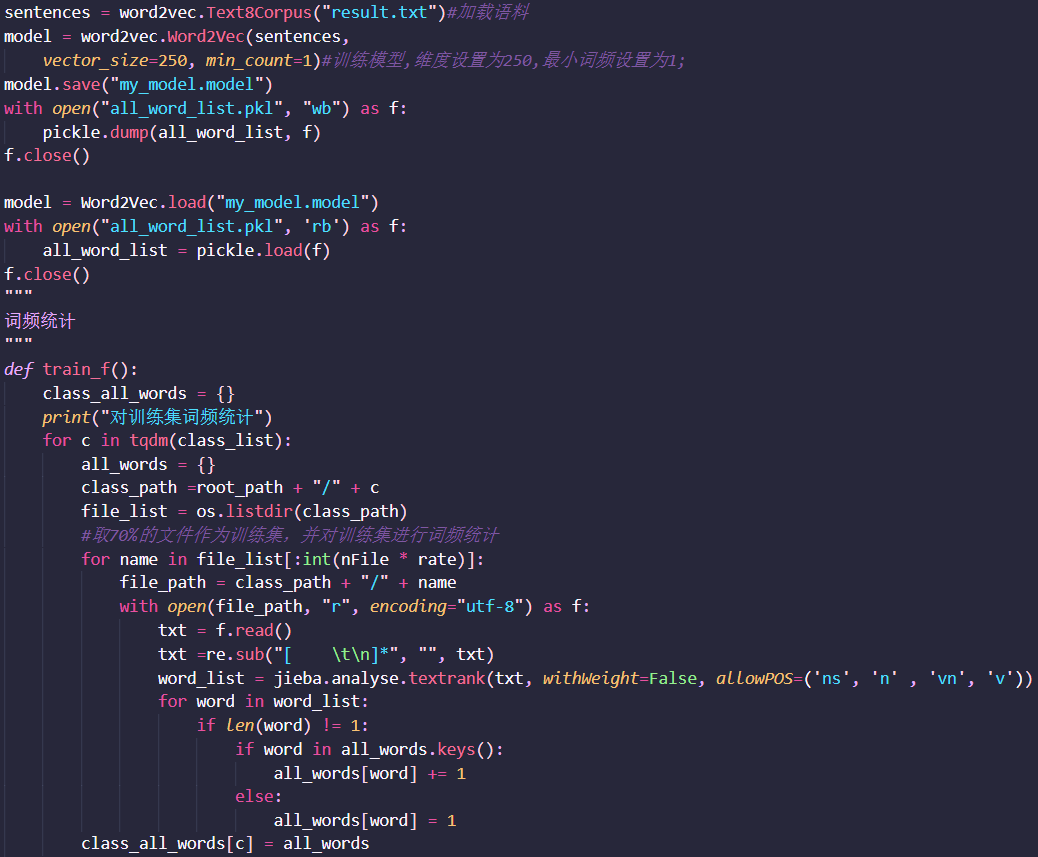
本实验结果以csv文件呈现，csv文件中包含了爬取的豆瓣电影评论、百度文本内容审核API的审核结果、相应原因。实验结果如图所示。

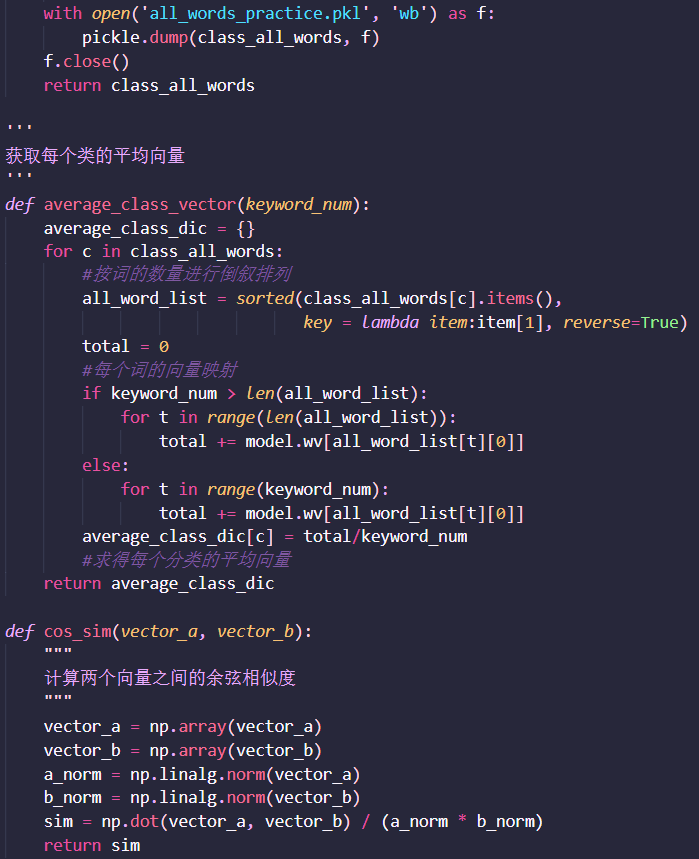


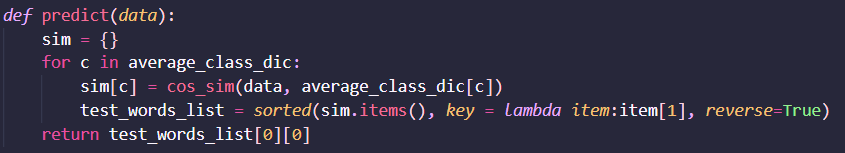
**基于word2vec模型的文本分类任务**

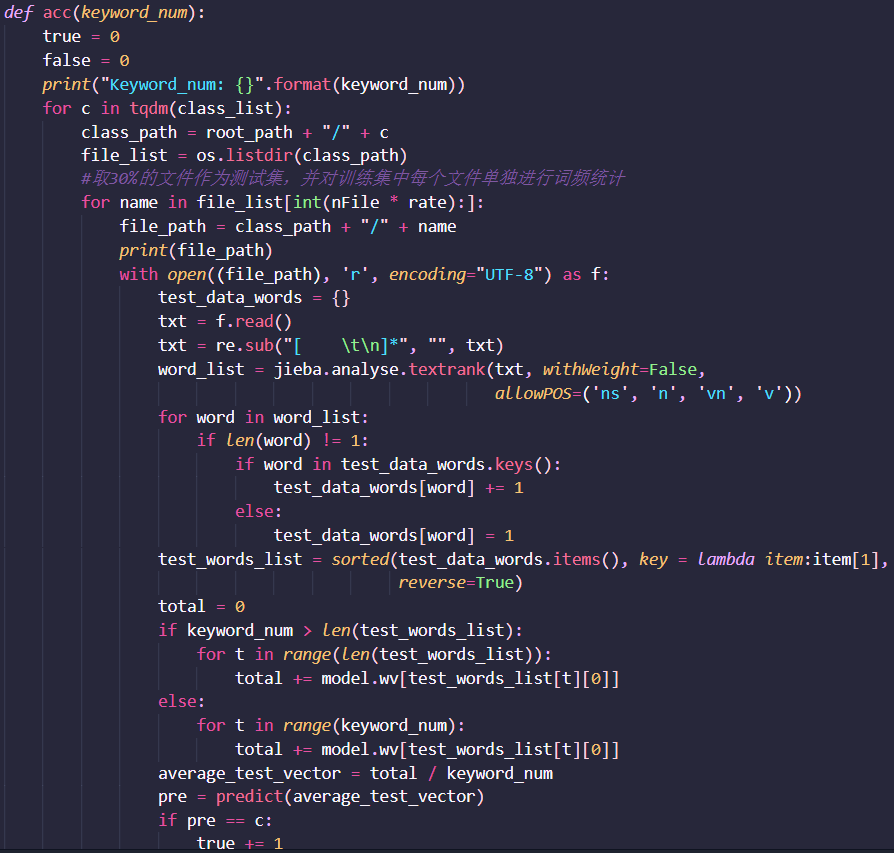
本实验使用到的实验代码如下图所示。













**百度文本内容审核API调用**

本实验使用到的实验代码如下图所示。

