**基于深度学习的自然语言处理**

**第1次实验报告**

10182100359 郑佳辰

**一、实验名称**

使用word2vec和Fasttext训练中文词向量

**二、实验日期**

2020.09.14 – 2020.10.09

**三、实验目的**

（1）熟悉这两种训练词向量方法的内部原理

（2）使用这两种方法训练词向量，并在测评任务上测试词向量的精度

**四、实验数据**

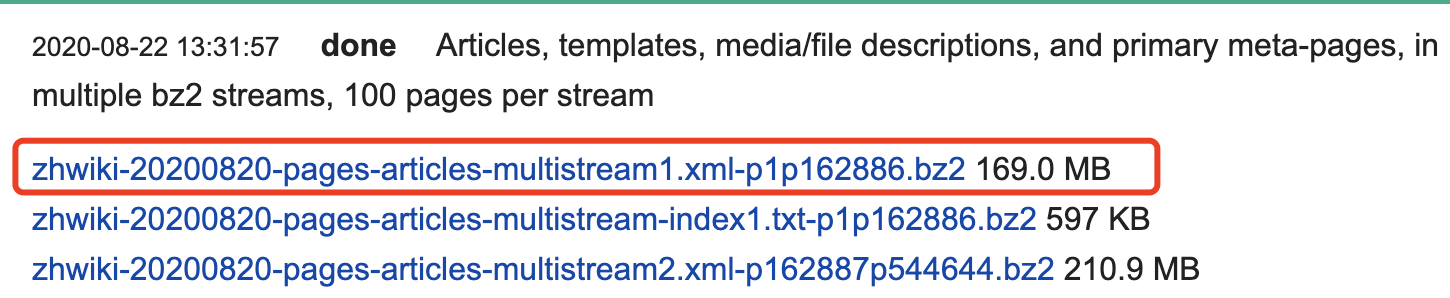
维基百科上的中文数据

**五、实验步骤及结果**

**5.1 word2vec**

（1）使用维基百科下载中文语料库

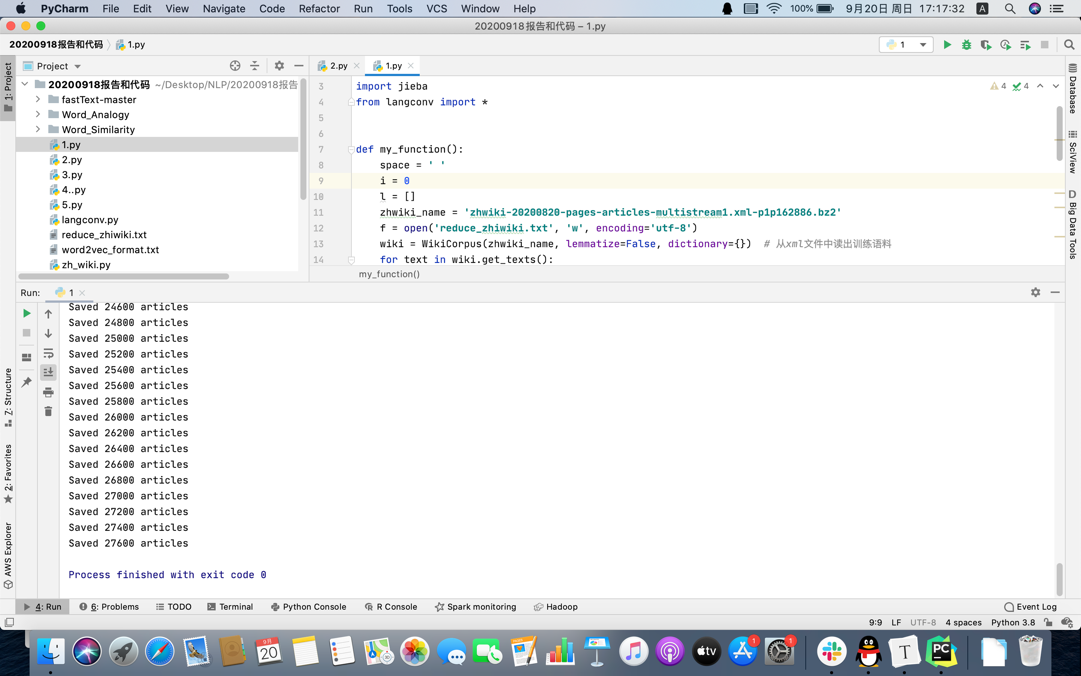
下载位置：<https://dumps.wikimedia.org/zhwiki/20200820/>



从给出的下载地址下载文件zhwiki-20200820-pages-articles-multistream1.xml-p1p162886.bz2作为我们的语料。

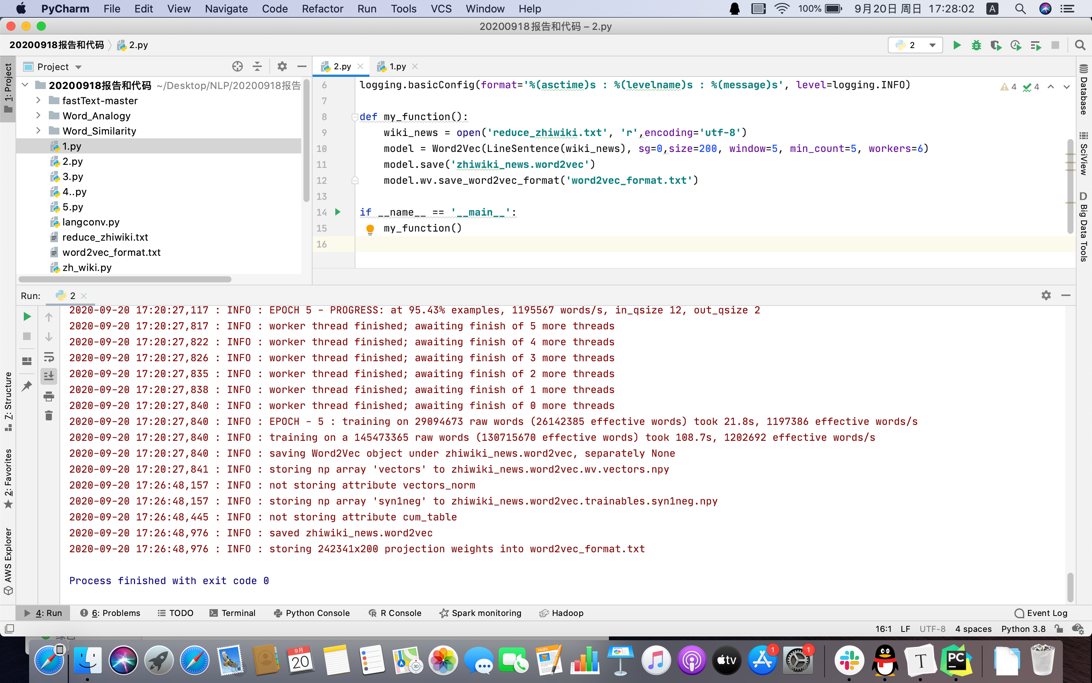
（2）用python wikipedia extractor抽取维基百科的内容（运行1.py）

先使用pip install gensim安装必要的gensim库，然后运行1.py。在1.py中，先打开下载好的新闻压缩包，使用WikiCorpus从文件中读取出训练语料。之后对于每一句话，使用zh\_wiki.py中的字典将繁体字转换为简体字，再利用jieba库将中文句子分为一个个单词，最后存入输出文件reduce\_zhwiki.txt中。本程序运行共将新闻分成中文单词27600余个，运行时截图如下图所示。



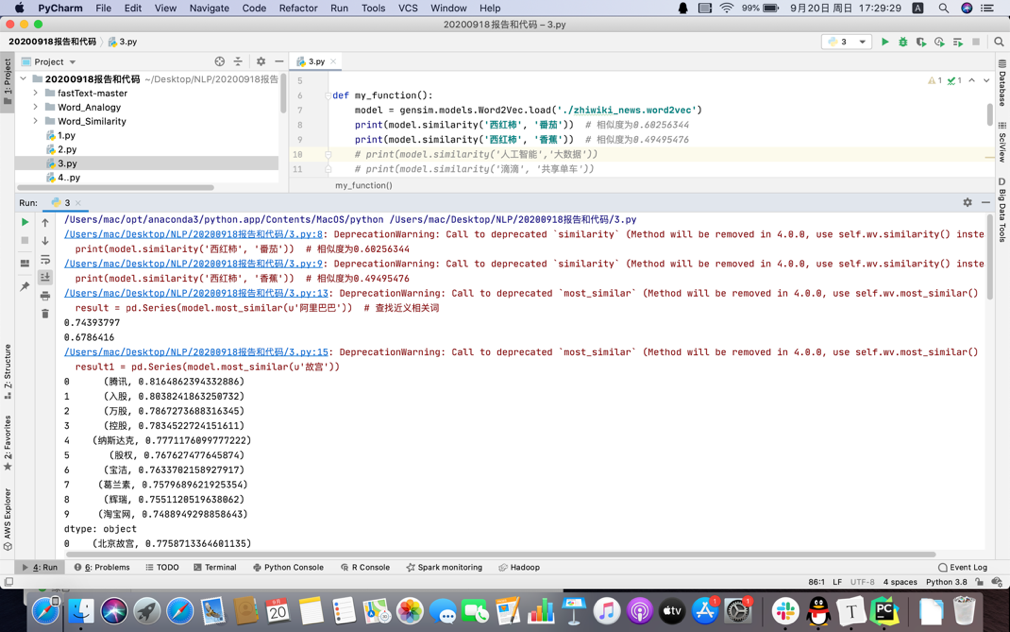
（3）安装包、调试代码跑通整个模型使用Gensim进行词向量训练（运行2.py）

使用gensim.models里面的Word2Vec模型进行词向量训练。程序2.py先打开之前分好的中文单词，然后选择一定的参数利用Word2Vec对词向量进行训练。并将模型和词向量保存并输出。运行结果截图如下。

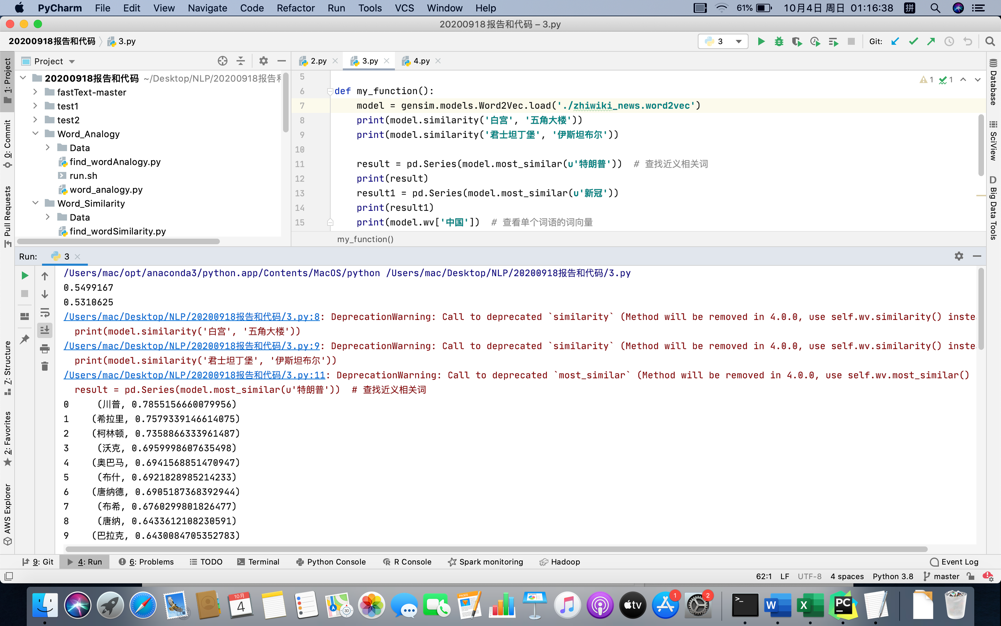


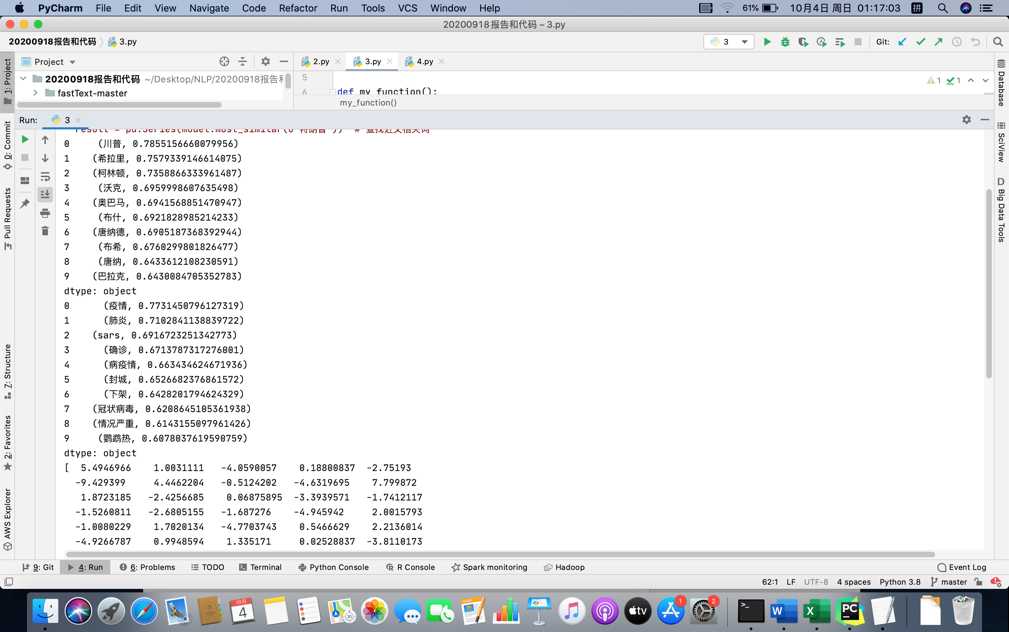
（4）测试同义词，找几个单词，看下效果（运行3.py）

使用一些词对词向量进行测试，首先测试词语的相似度，使用西红柿和番茄，西红柿和香蕉进行测试，结果如图所示。然后找出阿里巴巴和故宫的近义相关词。并且打印中国的词向量。运行结果如图所示。



修改参数使之达到最高分数之后，更改词语相似度测试单词为白宫和五角大楼与伊斯坦布尔与君士坦丁堡两组。然后找出特朗普和新冠肺炎的相似词，最后打印中国的词向量。输出结果如图。



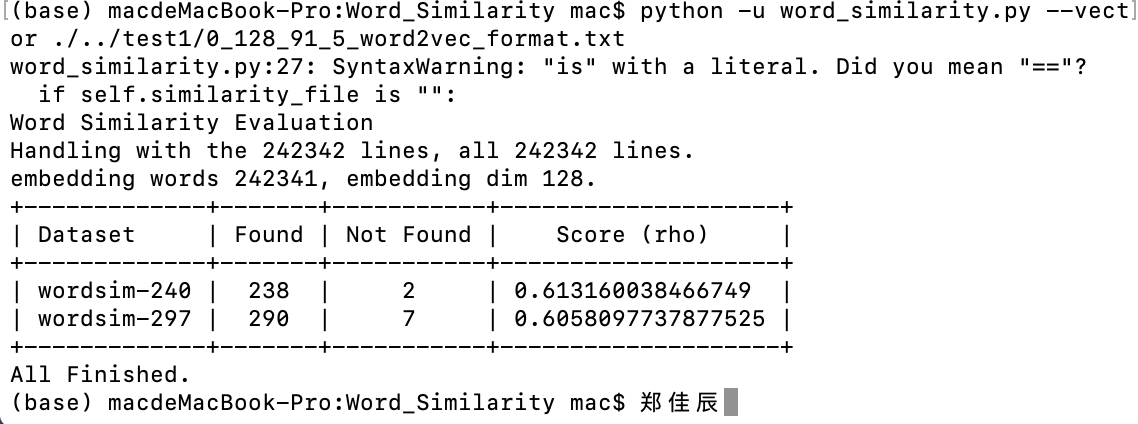


（5）使用训练好的词向量测试词相似度任务的精度

在Word \_Similarity文件夹下运行 ：

python -u word\_similarity.py --vector word2vec\_format.txt

使用word\_similarity.py测试词相似度时输入上述指令，可以同时测试wordsim-240.txt和wordsim-297.txt。该程序在读取词相似度文件之后，对于每一行先判断这两个词是否在词向量中出现。如果出现则计算这两个词对应的词向量的余弦相似度，并与给出的标准余弦相似度进行比较，计算得分。将参数优化后得出的得分截图如下。

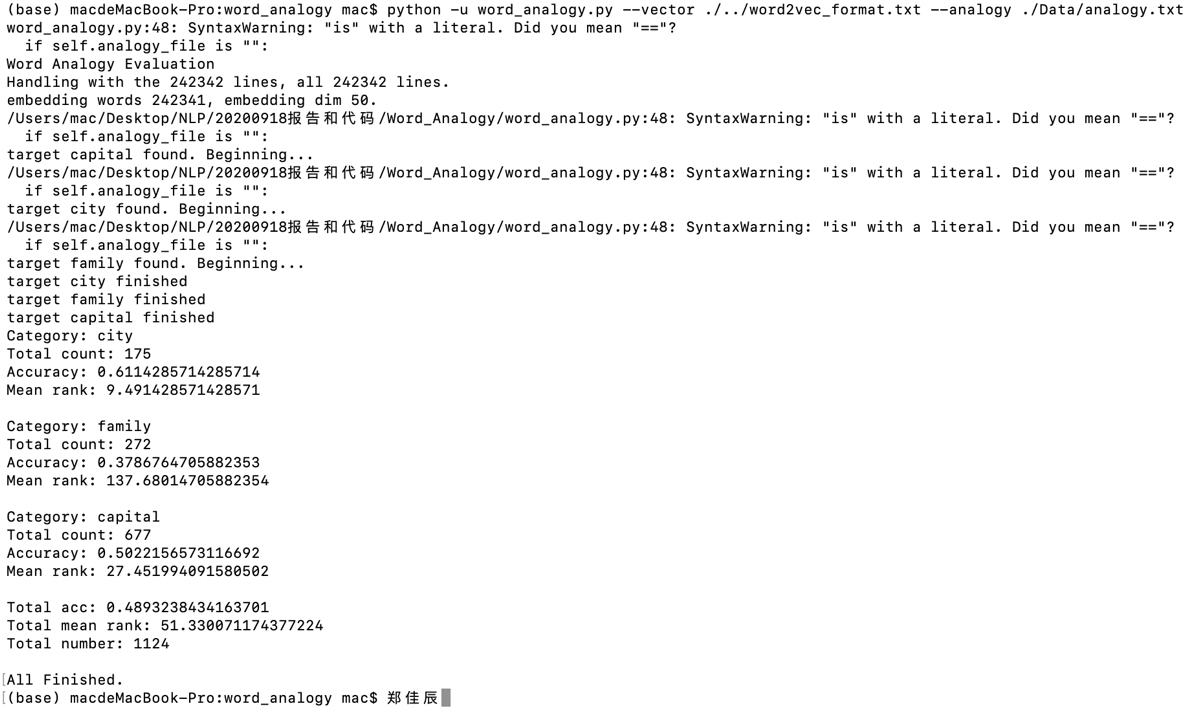


（6）使用训练好的词向量测试词类比任务的精度

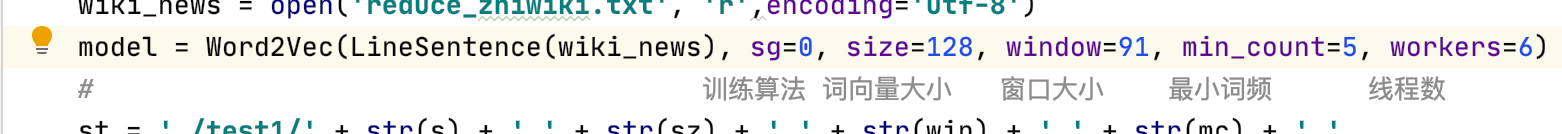
在Word\_Analogy文件夹下运行 ：

python -u word\_analogy.py --vector word2vec\_format.txt -–analogy analogy.txt

使用word\_analogy.py测试词类比，测试分为capital，city，family三类，使用多线程的方法进行测试。具体方法为：如果前两个词的词向量之差，和后两个词的词向量之差相同，则可以认为这两组关系相似。该程序通过计算给定词向量的差值来测试词向量，测试结果如下图所示。



（7）调试模型中的参数重新训练词向量，提高两个任务的精度



参数sg用于设置训练算法，默认为0，对应CBOW算法。如果设置为1则会采用skip-gram算法。参数size用于设置词向量的维数。参数window是窗口大小，表示当前词与预测词在一个句子中的最大距离。参数min\_count将对字典进行截断，出现频数低于这个数的词将不会出现在字典里。参数workers用于控制训练的并行数。优化后的选择CBOW算法，词向量为128，滑动窗口大小为91，最小词频为5，工作线程数保持6不变。

此外，其他的一些参数包括初始的学习率alpha，学习率最小值min\_alpha，控制是否使用层级softmax技巧的参数hs，设置负采样及负样本个数的参数negative，设置词向量构建时RAM大小的参数max\_vocab\_size等。

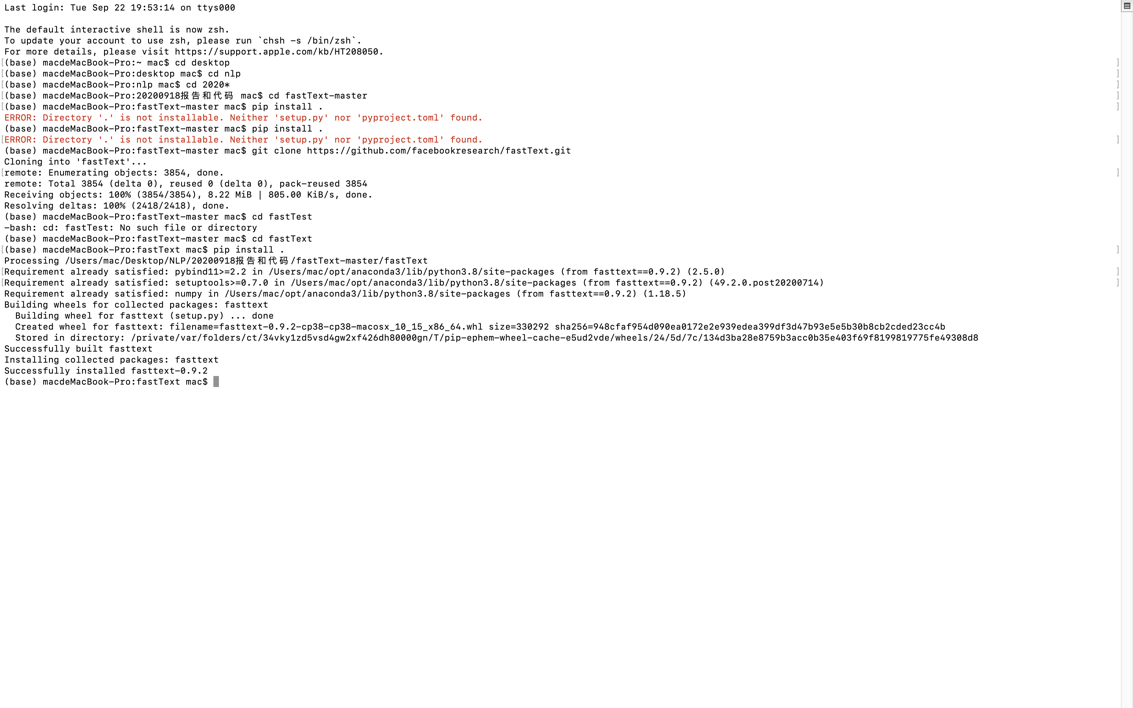
在调参过程中取不同参数时，两个相似度测试的得分如下图（部分）。根据测试过程及结果结果来看，skip-gram算法比CBOW算法要快，但是CBOW算法的精度会高一些，这和网上查到的结果是一致的。一般来说，在一定范围内，词向量大小和窗口大小越大，测试的准确度也就越高。但是如果这两个参数设置的过大，也会出现过拟合的现象。对于词向量大小和窗口大小来说，随着参数值的增加，评分的增速逐渐放缓，并在某个数值过后基本保持不变。这时继续增加参数值，会大大增加所需的计算资源，但却只能得到很小的评分提升。这种情况下继续增加参数大小其实是不合算的。

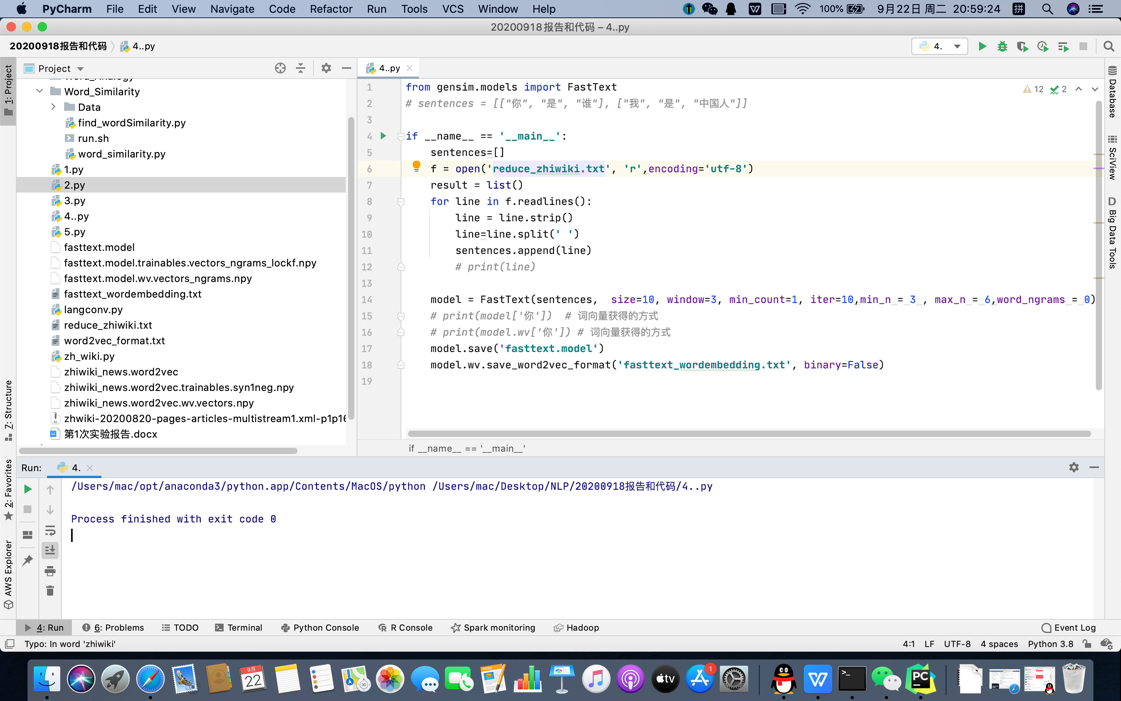


**5.2 Fasttext**

（1）安装包、调试代码跑通整个模型使用Gensim进行词向量训练（运行4.py）

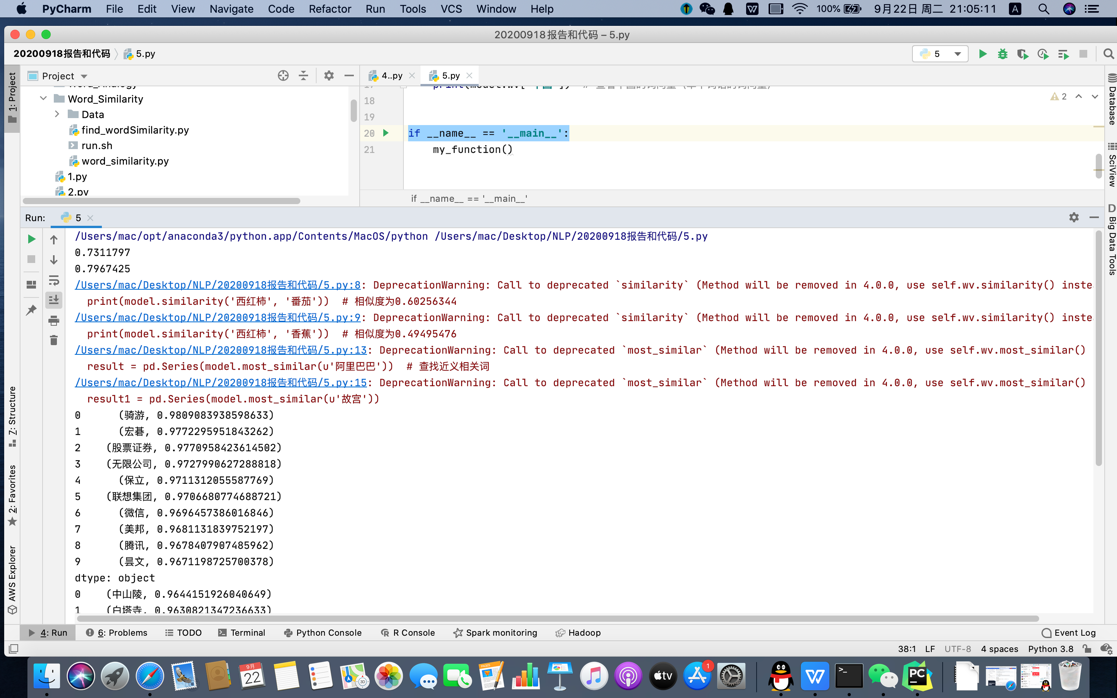
首先安装fastTest库，然后利用1.py中已经提取出来的单词reduce\_zhwiki.txt进行训练。先将文件头尾的空格移除，然后根据空格分出词语。之后设定合适的参数并利用Fasttest训练词向量模型。最后将训练出的词向量模型保存。安装库和运行时截图如图所示。



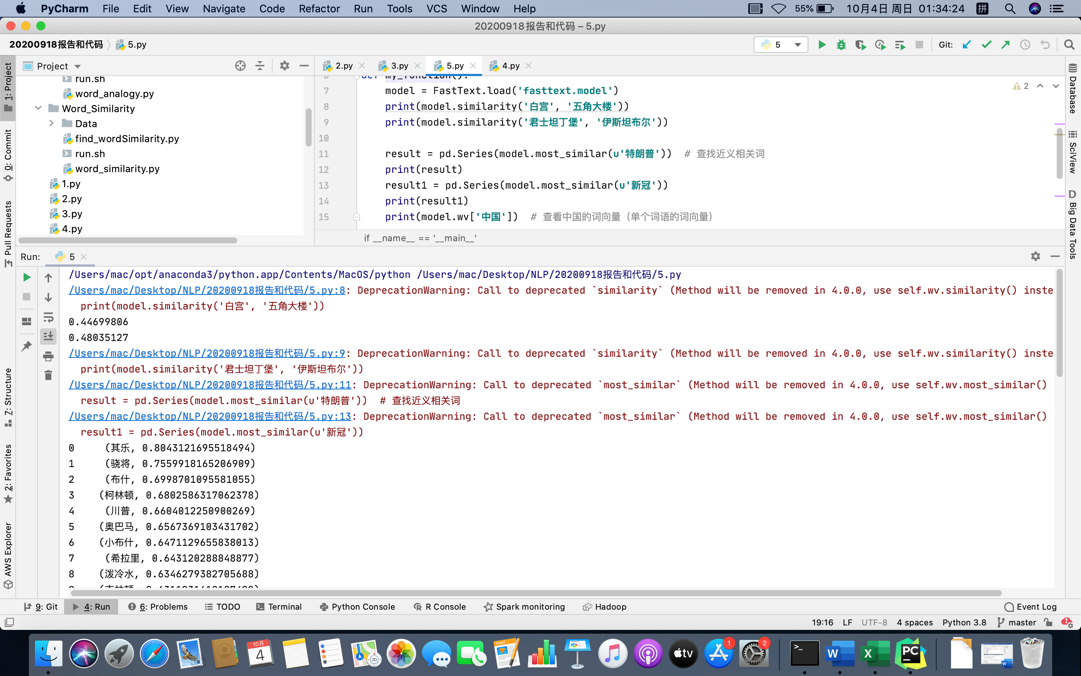


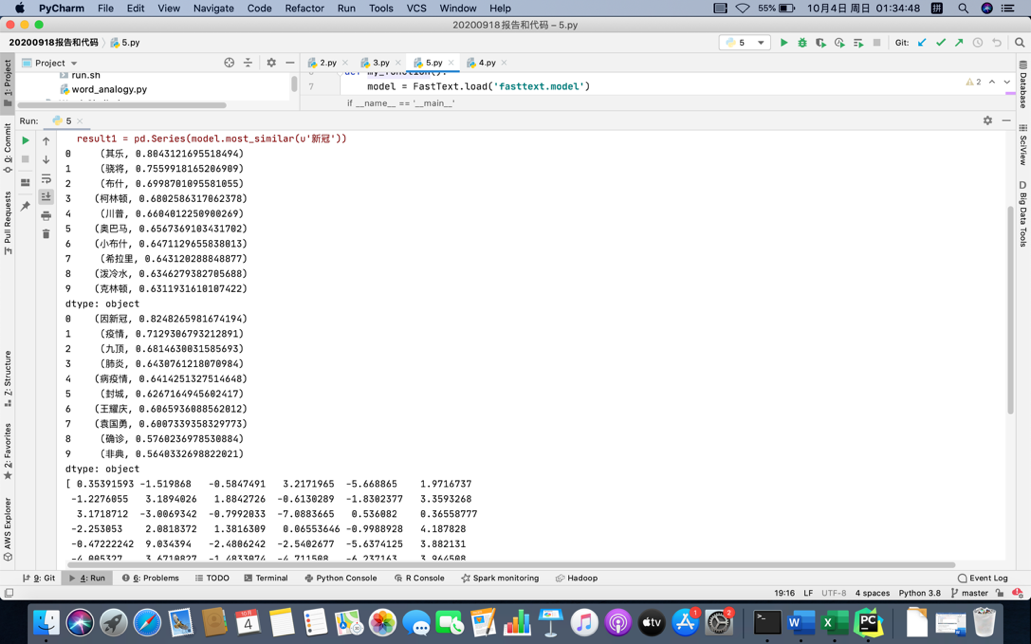
（2）测试同义词，找几个单词，看下效果（运行5.py）

运行5.py使用一些词对词向量进行测试，首先测试词语的相似度，使用西红柿和番茄，西红柿和香蕉进行测试，结果如图所示。然后找出阿里巴巴和故宫的近义相关词。并且打印中国的词向量。运行结果如图所示。



调整参数使得分达到最高之后，按照3.py中更换的词进行测试。结果如图。



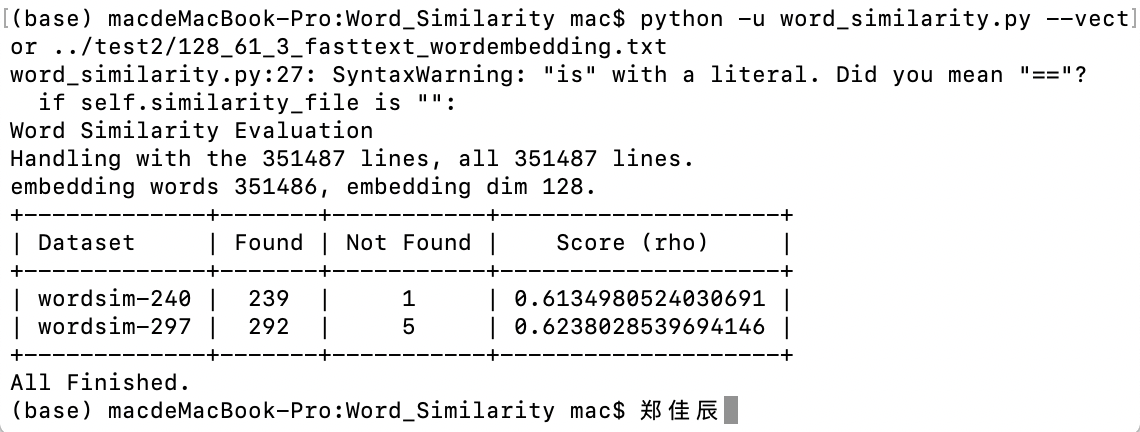


（3）使用训练好的词向量测试词相似度任务的精度

在Word \_Similarity文件夹下运行 ：

python -u word\_similarity.py --vector fasttext\_wordembedding.txt

同时测试wordsim-240.txt和wordsim-297.txt。原理和上一问相同。将参数优化后得出的截图如下。



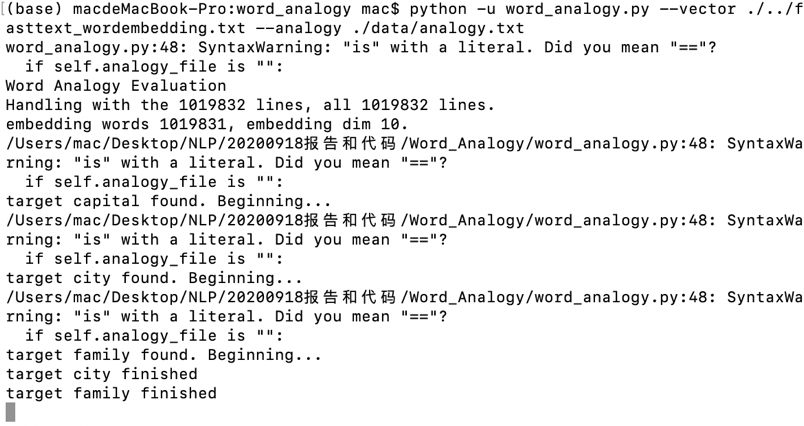
（4）使用训练好的词向量测试词类比任务的精度

在Word\_Analogy文件夹下运行 ：

python -u word\_analogy.py --vector fasttext\_wordembedding.txt -–analogy analogy.txt

（--vector 向量文件位置）

使用word\_analogy.py测试词类比。测试方法与上一问中相同。由于词类比测试运行过程中的截图如下图所示。



（5）调试模型中的参数重新训练词向量，提高两个任务的精度

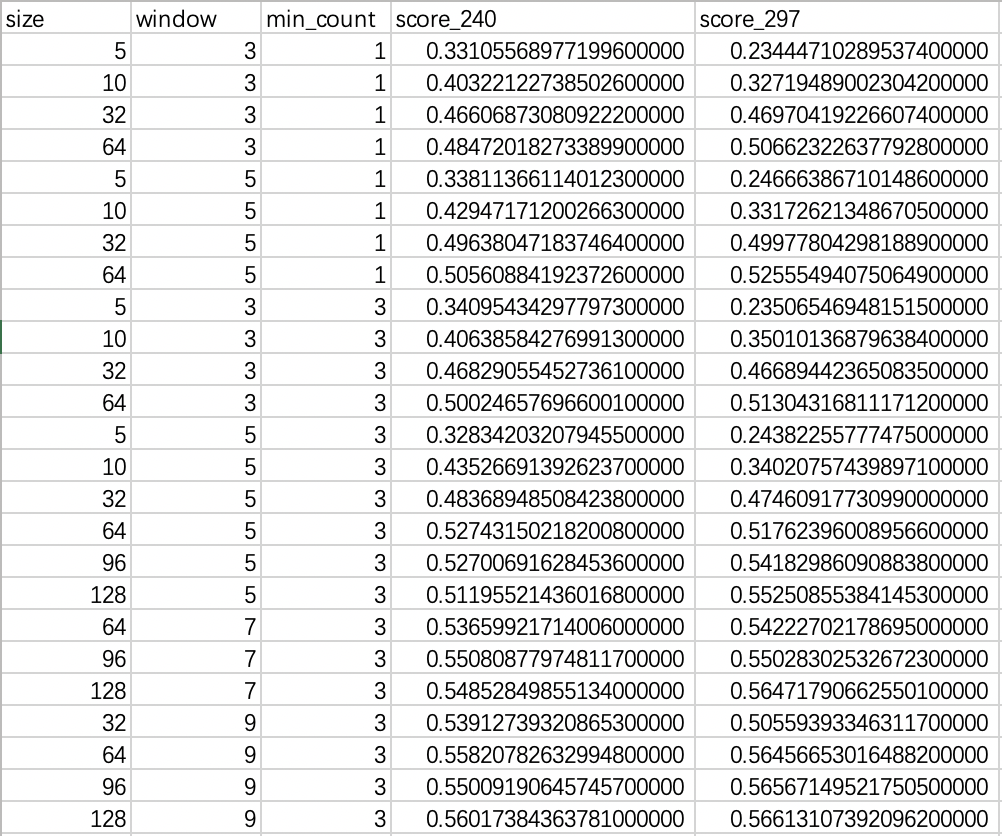


参数size用于设置词向量的维数。参数window是窗口大小，表示当前词与预测词在一个句子中的最大距离。参数min\_count将对字典进行截断，出现频数低于这个数的词将不会出现在字典里。参数min\_n和max\_n分别是模型的最小和最大的字符长度。参数word\_ngrams用于设置ngrams的相关信息。优化后的选择的词向量大小为128，滑动窗口大小为61，最小词频为3，其他参数保持不变。

此外，其他的一些参数包括更新次数的参数epoch，使用的损失函数loss，线程数量thread，初始的学习率lr，学习率更新速率lr\_update\_rate等。

在调参过程中取不同参数时，两个相似度测试的得分如下图（部分）。它和上一节中word2vec的规律大致相同。在一定范围内，词向量大小和窗口大小越大，测试的准确度越高。如果这两个参数设置的过大，会出现过拟合的现象。事实上，为了极大化最终的测试得分，我的参数确实出现了过拟合现象。表现在于虽然得分非常高，但在上一问的词语相似度的测验上结果有些不理想。

在word2vec和fasttext这两种方法中，一般情况下，fasttext准确程度较高。在fasttext中，可以在窗口大小和词向量大小两个参数值较小时仍达到较好的得分。从运行时间上来看，fasttest在参数值设置较大时运行速度会非常慢，变得名不副实。但word2vec在这种情况下时间增加不大。



**六、实验感想**

本次实验我学会了使用word2vec和Fasttext来训练词向量，明白了CBOW算法和skip-gram算法的原理和它们的差异。CBOW算法使用滑动窗口中周围的词预测并优化中间的词，而skip-gram算法是使用中间的词来预测并优化周围的词。学会了利用词向量计算两个单词的相似度和进行词类比。并且练习了如何优化参数使评分最大化。这些知识给我之后的自然语言处理的学习打下了良好的基础。