《自然语言处理》

-词向量实验

华为技术有限公司

目录

1 实验总览 2

1.1 实验背景 2

1.2 实验目的 2

1.3 实验清单 2

2 Word2Vec 词向量训练实验 3

2.1 实验简介 3

2.2 实验环境 3

2.3 实验步骤 3

2.3.1 实验准备 3

2.3.2 实验过程 8

2.4 实验小结 11

1 实验总览

1.1 实验背景

词向量（Word embedding），又叫 Word 嵌入，是自然语言处理（NLP）中的一组语言建模和特征学习技术的统称，其中来自词汇表的单词或短语被映射到实数的向量。 从概念上讲，它涉及从每个单词一维的空间到具有更低维度的连续向量空间的数学嵌入。

当用作底层输入表示时，单词和短语嵌入已经被证明可以提高 NLP 任务的性能，例如语法分析和情感分析。

1.2 实验目的

本章实验主要介绍如何在 ModelArts 平台上完成词向量的训练，并应用于语义相似词的搜索、扩展。通过实验掌握词向量训练方法。

1.3 实验清单

实验	简述	难度	软件环境	开发环境
Word2vec词向量训练实验	基于Python和gensim框架实现Word2Vec在Wikipedia语料集上面的应用	初级	Python3.7、Gensim	ModelArts Ascend Notebook环境

2 Word2Vec 词向量训练实验

2.1 实验简介

Word2vec 是 Google 在 2013 年开源的一款用于词向量计算的工具，一经发布就引起了工业界和学术界的关注。首先，Word2vec 可以在百万数量级的词典和上亿的数据集上进行高效地训练；其次，该工具训练得到的词向量（word embedding），可以很好地度量词与词之间的相似性。Word2vec 不是一种深度学习算法，其后面只是一个浅层神经网络，包含两种模型：CBOW 模型和 Skip-gram 模型。

本章实验主要是基于 Python 和 gensim 框架实现 Word2vec 在 Wikipedia 语料集上面的应用，并且获取词的词向量以及寻找相近词。

2.2 实验环境

ModelArts Ascend Notebook 环境，环境设置参考《华为云 ModelArts Ascend 环境配置》

2.3 实验步骤

2.3.1 实验准备

步骤 1　OBS 创建项目文件夹

使用 OBS Brower+登录 OBS

图2-1 OBS Brower+登录

进入之前创建的用于挂载 notebook 的 obs 目录，此处我们是建了一个“nlp”的目录用于挂载昇腾环境下的 notebook：

图2-2 进入课程主目录

创建“word_embedding”文件夹

图2-3 创建项目文件夹

创建完如下所示：

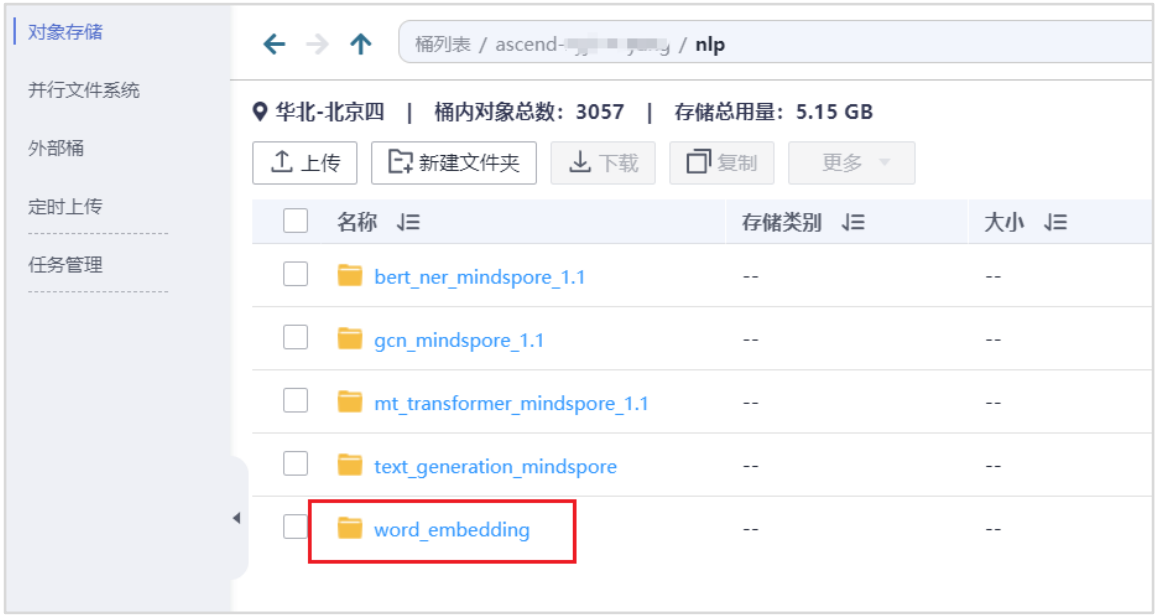

图2-4 项目文件夹创建成功

步骤 2　上传实验数据

进入刚创建的“word_embedding”文件夹，上传数据至该目录下。

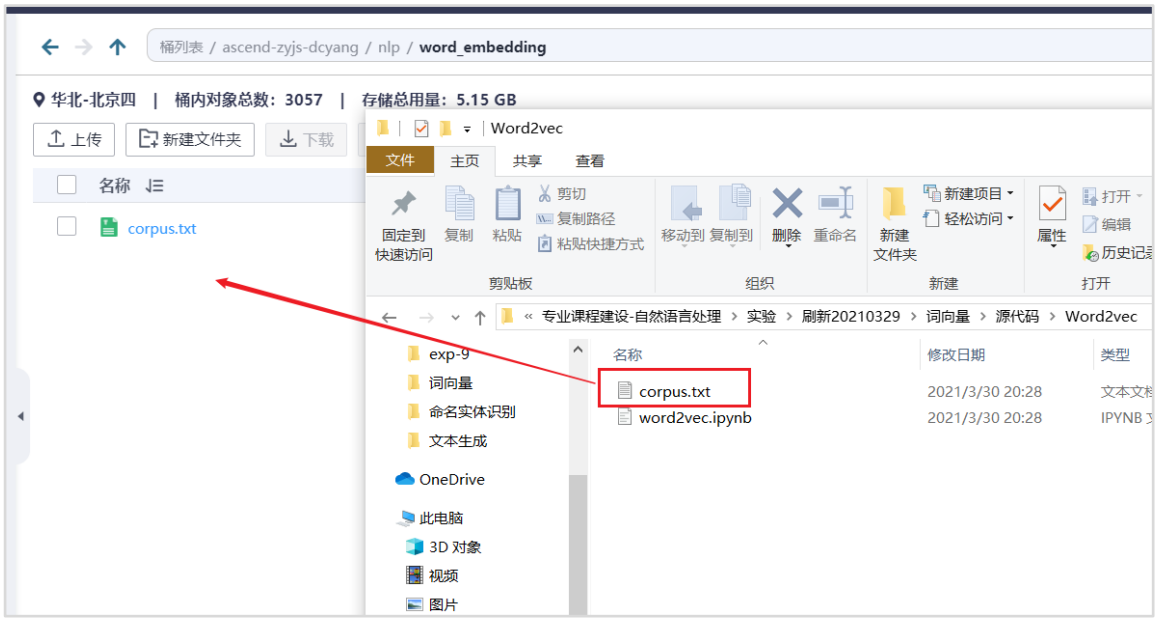

图2-5 上传实验数据

步骤 3　打开 JupyterLab

登录华为云，启动之前创建的 ModelArts Ascend notebook 环境

图2-6 启动 notebook

打开 JupyterLab，并进入之前创建的“word_embedding”文件夹

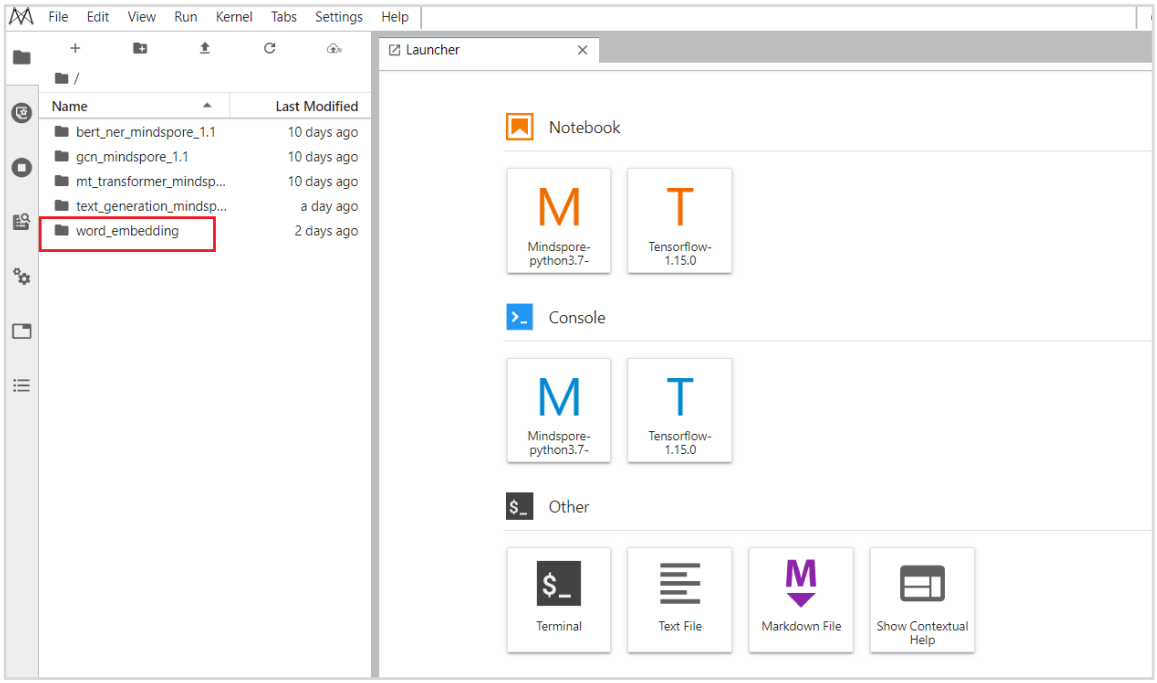

图2-7 进入项目文件夹

新建 notebook 文件：

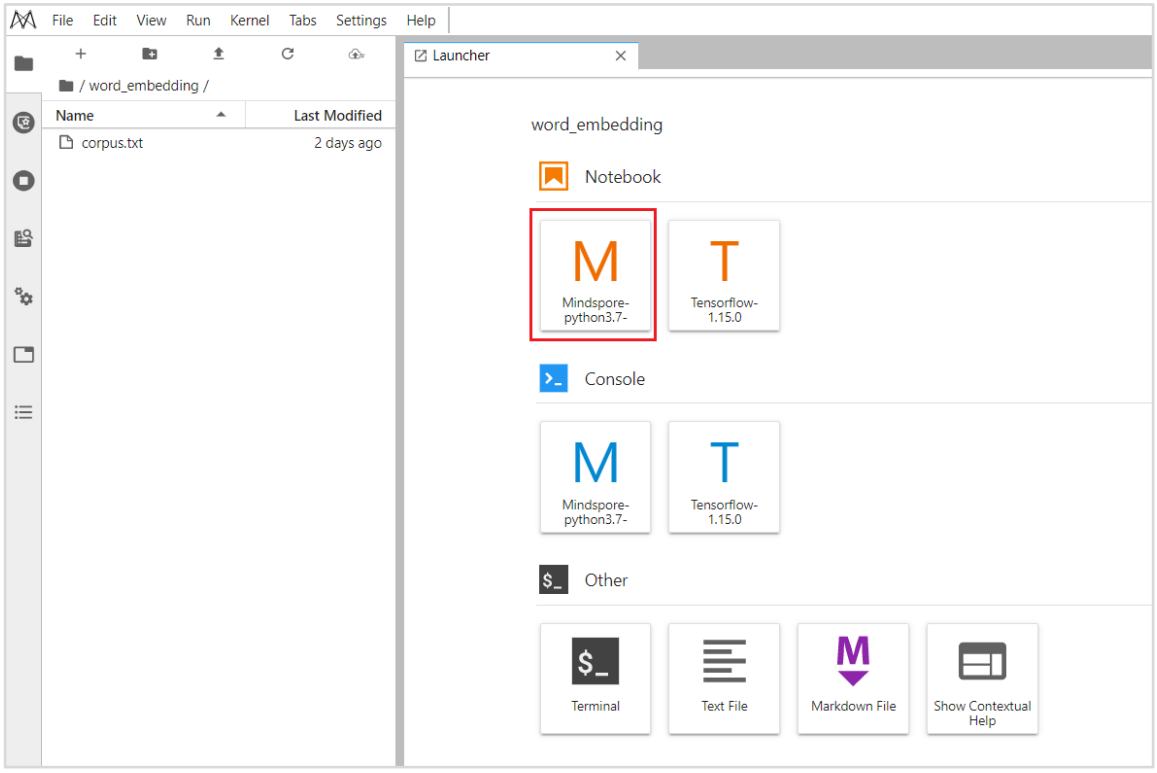

图2-8 新建 notebook 文件

重命名为“word2vec.ipynb”并打开，右侧显示代码编辑区

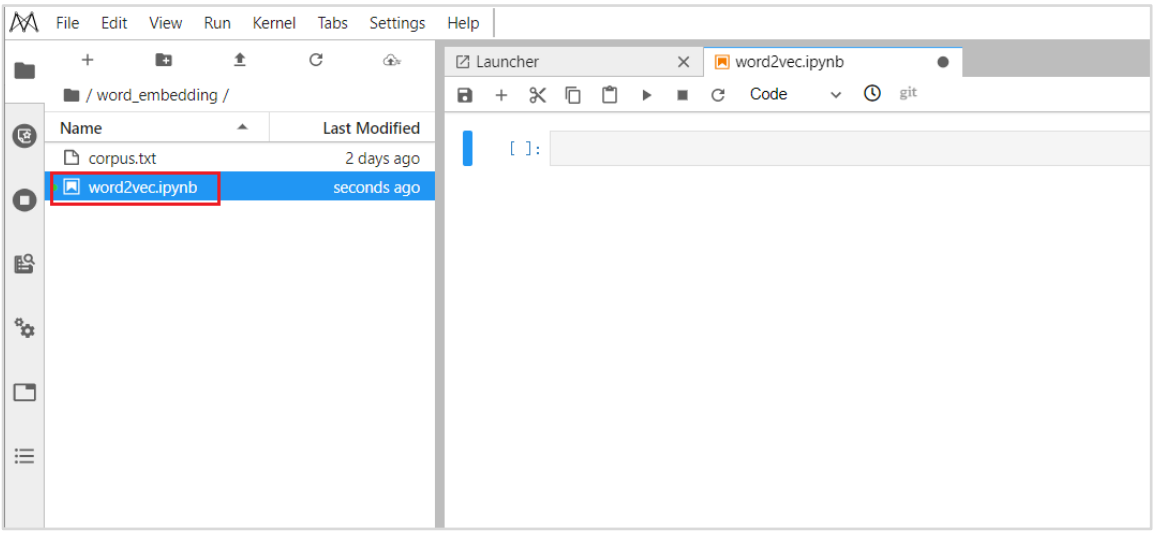

图2-9 重命名并打开 notebook 文件

2.3.2 实验过程

步骤 1　安装 gensim 库

输入：

```
! pip install gensim
```

输出：

```
! pip install gensim

Looking in indexes: http://repo.myhuaweicloud.com/repository/pypi/simple
Collecting gensim
  Downloading http://repo.myhuaweicloud.com/repository/pypi/packages/cc/ff/f809deb11f066dfe658fc9756ac7d04d1f8954d691f31ddd40d40db59b85/gensim-4.0.0.tar.gz (23.1MB)
    100% |████████████████████████████████| 23.1MB 73.2MB/s ta 0:00:01
Requirement already satisfied: numpy>=1.11.3 in /home/ma-user/miniconda3/envs/Mindspore-python3.7-aarch64/lib/python3.7/site-packages (from gensim) (1.17.5)
Requirement already satisfied: scipy>=0.18.1 in /home/ma-user/miniconda3/envs/Mindspore-python3.7-aarch64/lib/python3.7/site-packages (from gensim) (1.5.4)
Collecting smart_open>=1.8.1 (from gensim)
  Downloading http://repo.myhuaweicloud.com/repository/pypi/packages/c3/ff/f8ac652bc62fc78a35b0201d922e43da63e3469a6eab1d71776af6dd7844/smart_open-4.2.0.tar.gz (119kB)
    100% |████████████████████████████████| 122kB 62.3MB/s ta 0:00:01
Building wheels for collected packages: gensim, smart-open
  Running setup.py bdist_wheel for gensim ... done
  Stored in directory: /home/ma-user/.cache/pip/wheels/8c/5d/8e/094b2a0e3039fca03c5d7cffc57faae5d3954ba5c9d930c621
  Running setup.py bdist_wheel for smart-open ... done
  Stored in directory: /home/ma-user/.cache/pip/wheels/b6/d3/fa/ca97de62b50f6e9d99f29d0599ddaf5696f5331e3aac70a457
Successfully built gensim smart-open
schedule-search 0.0.1 requires absl-py, which is not installed.
schedule-search 0.0.1 requires tensorflow>=1.10.0, which is not installed.
moxing-mindspore 1.17.3 requires enum34>=1.1.6, which is not installed.
modelarts 1.1.6 has requirement pyyaml==3.12, but you'll have pyyaml 5.3.1 which is incompatible.
modelarts 1.1.6 has requirement urllib3==1.21.1, but you'll have urllib3 1.25.11 which is incompatible.
Installing collected packages: smart-open, gensim
Successfully installed gensim-4.0.0 smart-open-4.2.0
You are using pip version 10.0.1, however version 21.0.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
```

步骤 2　同步数据和源码至本地容器

因为 notebook 是挂载在 obs 上，运行的容器实例不能直接读取操作 obs 上的文件，需下载至容器本地环境中，请在 src_url 字段将 obs 桶名称替换成自己创建的 obs 桶名称。

输入：

```
import moxing as mox
mox.file.copy_parallel(src_url="s3://替换成自己的路径/nlp/word_embedding/corpus.txt",
dst_url='corpus.txt')
```

步骤 3　导入依赖库

输入：

```
from multiprocessing import cpu_count
from gensim.models import Word2Vec
from gensim.models import KeyedVectors
```

步骤 4　定义输入、输出文件路径

输入：

```
corpus_file = "corpus.txt"
out_embedding_file = "embedding.txt"
```

步骤 5　查看函数文档

输入：

```
?Word2Vec
```

输出：

```
?Word2Vec
Init signature:
Word2Vec(
    sentences=None,
    corpus_file=None,
    vector_size=100,
    alpha=0.025,
    window=5,
    min_count=5,
    max_vocab_size=None,
    sample=0.001,
    seed=1,
    workers=3,
    min_alpha=0.0001,
    sg=0,
    hs=0,
    negative=5,
    ns_exponent=0.75,
    cbow_mean=1,
    hashfxn=<built-in function hash>,
    epochs=5,
    null_word=0,
    trim_rule=None,
    sorted_vocab=1,
    batch_words=10000,
    compute_loss=False,
    callbacks=(),
    comment=None,
    max_final_vocab=None,
)
Docstring:
Serialize/deserialize objects from disk, by equipping them with the `save()` / `load()` methods.
```

步骤 6　词向量训练并保存

输入：

```
model = Word2Vec(corpus_file=corpus_file, vector_size=100, window=5, min_count=5, workers=cpu_count(), sg=1)
model.wv.save_word2vec_format(out_embedding_file, binary=False)
```

输出：

```
INFO:gensim.models.word2vec:worker thread finished; awaiting finish of 1 more threads
INFO:gensim.models.word2vec:worker thread finished; awaiting finish of 0 more threads
INFO:gensim.models.word2vec:EPOCH - 5 : training on 9202724 raw words (7445726 effective words) took 64.6s, 115334 effective words/s
INFO:gensim.utils:Word2Vec lifecycle event {'msg': 'training on 46013620 raw words (37225405 effective words) took 336.9s, 110479 effective words/s', 'datetime': '2021-03-30T02:02:38.009792', 'gensim': '4.0.0', 'python': '3.7.6 | packaged by conda-forge | (default, Jun  1 2020, 18:15:32) \n[GCC 7.5.0]', 'platform': 'Linux-4.19.36-vhulk1907.1.0.h619.eulerosv2r8.aarch64-aarch64-with-centos-2.0-SP8', 'event': 'train'}
INFO:gensim.utils:Word2Vec lifecycle event {'params': 'Word2Vec(vocab=99249, vector_size=100, alpha=0.025)', 'datetime': '2021-03-30T02:02:38.011460', 'gensim': '4.0.0', 'python': '3.7.6 | packaged by conda-forge | (default, Jun  1 2020, 18:15:32) \n[GCC 7.5.0]', 'platform': 'Linux-4.19.36-vhulk1907.1.0.h619.eulerosv2r8.aarch64-aarch64-with-centos-2.0-SP8', 'event': 'created'}
INFO:gensim.models.keyedvectors:storing 99249x100 projection weights into embedding.txt
```

步骤 7　加载离线词向量

输入：

```
word2vec_model = KeyedVectors.load_word2vec_format("embedding.txt")
```

获取单个词的词向量

输入：

```
word2vec_model['中国']
```

输出：

```
word2vec_model['中国']

array([-0.11945462,  0.80811197, -0.2192669 , -0.35121506, -0.2546237 ,
       -0.00980086,  0.7964671 ,  0.36271024,  0.31742495, -0.3251099 ,
       -0.13367757, -0.512894  ,  0.18774055,  0.28745985, -0.08601924,
        0.0150877 ,  0.73333174, -0.6884307 ,  0.09896889, -0.4288761 ,
        0.7132578 ,  0.396008  ,  0.4390735 , -0.09943724,  0.20115437,
        0.07450946,  0.01946275,  0.5919555 ,  0.11550876, -0.2978358 ,
       -0.18850829, -0.279784  , -0.56403273, -0.6189067 , -0.2669414 ,
       -0.1924906 ,  0.56643754,  0.05672867,  0.20835687,  0.53800774,
        0.05601315,  0.01130022, -0.13399325,  0.01399448,  0.44789463,
       -0.16624065,  0.23680332,  0.48087487,  0.2066727 ,  0.30111927,
        0.23712948, -0.01020607,  0.09069001, -0.35201445,  0.30945036,
       -0.29315245,  0.0818078 , -0.15353864, -0.21203907,  0.33678278,
       -0.26471385, -0.40392622,  0.12019241,  0.37057823,  0.10111269,
        0.29821014,  0.1682322 ,  0.3837336 , -0.48044497,  0.00659311,
       -0.12619178,  0.08527908, -0.10650562,  0.01504959, -0.17726953,
       -0.42102095,  0.4874898 ,  0.4325759 ,  0.54365766, -0.02363636,
       -0.3604603 ,  0.21117018, -0.56193644, -0.2412196 ,  0.33035398,
       -0.09440871,  0.8251365 ,  0.16836917, -0.2605197 ,  0.40633464,
        0.7236247 , -0.36978406,  0.37283292,  0.40109146, -0.09860536,
       -0.48287332,  0.05875314,  0.15065585, -0.8095557 ,  0.62114567],
      dtype=float32)
```

步骤 8　相似度测试

输入：

```
testwords = ['金融', '喜欢', "中国", "北京"]
for word in testwords:
    res = word2vec_model.most_similar(word)
    print (word)
    print (res)
```

输出：

```
金融
[('金融服务', 0.7745651006698608), ('证券期货', 0.772100031375885), ('信贷', 0.7595240473747253), ('投资银行', 0.7579774260520935),
('国际金融', 0.7491804361343384), ('银行学', 0.7486739158630371), ('金融保险', 0.7417786717414856), ('保险业', 0.7401484251022339),
('期货', 0.7384737133979797), ('金融市场', 0.738309383392334)]
喜欢
[('讨厌', 0.7161690592765808), ('吃喝', 0.6951901912689209), ('很会', 0.6930608153343201), ('喝酒', 0.6888067722320557), ('吓人',
0.6877806186676025), ('偏爱', 0.686855137348175), ('迟钝', 0.6796375513076782), ('卖弄', 0.6752473711967468), ('没什么', 0.6695799
231529236), ('鞋子', 0.6682382822036743)]
中国
[('大陆', 0.712977409362793), ('顾诚', 0.6032476425170898), ('李泽厚', 0.6030479669570923), ('榜上有名', 0.6003459692001343), ('世
界史', 0.5973314046859741), ('中国地图出版社', 0.5888921618461609), ('内地', 0.5883994102478027), ('少帅', 0.5853191018104553),
('戏曲史', 0.5793296098709106), ('瑰宝', 0.5780259370803833)]
北京
[('上海', 0.7301627993583679), ('天津', 0.7032418847084045), ('杭州', 0.6845793128013611), ('北京市', 0.660686194896698), ('沈阳',
0.6576042175292969), ('南京', 0.6466558575630188), ('上海市', 0.6343791484832764), ('武汉', 0.6275659203529358), ('西安', 0.618435
263633728), ('首都国际机场', 0.6135655641555786)]
```

步骤 9　词向量文件回传 obs

输入：

```
mox.file.copy_parallel(src_url="embedding.txt", dst_url='s3://替换成自己的路径
/nlp/word_embedding/embedding.txt')
```

2.4 实验小结

本实验介绍了在 ModelArts 平台上基于 gensim 库实现词向量的训练并进行相似词的测试评估，通过实验使学员了解词向量的训练方法，以及 gensim 库的基本使用。