深度学习与自然语言处理第四次作业

（基于 Word2Vec 的词向量聚类问题）

姓名：刘千歌

学号：BY2139121

一、 实验目标

利用给定语料库（或者自选语料库），利用神经语言模型（如：Word2Vec， GloVe 等模型）来训练词向量，通过对词向量的聚类或者其他方法来验证词向量的有效性。

二、背景介绍

2.1 词向量简介

自然语言技术通常需要将语言数学化，而向量是人类把自然界的东西抽象出来交给机器处理的典型形式。词向量把一个词表示成一个向量。 我们都知道词在送到神经网络训练之前需要将其编码成数值变量，常见的编码方式有两种：One-Hot Representation 和 Distributed Representation。

2.2 Word2Vec 模型介绍

传统的自然语言处理(NLP)方式往往将单词看作为离散的符号(discrete symbols)，就像一个词典一样，一个词对应一个编号———比如独热（One-Hot）编码形式，通过把符号就转换为数值进行运算。然而，这样的处理方式对于系统处理不同的词语没有提供有用的信息。词映射(word embedding)实现了将一个不可量化的单词映射到一个实数向量。Word embedding 能够表示出文档中单词的语义和与其他单词的相似性等关系。它已经被广泛应用在了推荐系统和文本分类中。

2.3 词的独热表示-One-hot 与分布式表示

最简单的也最容易想到的词表示方法是 One-hot Representation，这种方法把每个词表示为一个很长的向量。这个向量的维度是词表大小，其中绝大多数元素为 0，只有一个维度的值为 1，这个向量就代表了当前的词。例如：

“火锅”表示为 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 …]

“烧烤”表示为 [0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 …]

每个词都是茫茫 0 海中的一个 1。这种 One-hot Representation 如果采用稀疏方式存储，会是非常的简洁：也就是给每个词分配一个数字 ID。比如刚才的例子中，可爱

记为 3，面包记为 8（假设从 0 开始记）。这样的表示带来两点问题：

- 向量的维度会随着句子的词的数量类型增大而增大，容易导致维度灾难；
- 任意两个词之间都是孤立的，无法表示语义层面上词汇之间的相关信息。

针对上述问题，Harris 在 1954 年提出的“分布假说”为这一设想提供了理论基础：上下文相似的词，其语义也相似。Firth 在 1957 年对分布假说进行了进一步阐述和明确：词的语义由其上下文决定。Word Embedding 正是这样的分布式表示模型，而 Word2Vec 则是其中的一个典型，Word2Vec 包含两种模型，即 CBOW 模型和 Skip-gram 模型。以 CBOW 模型为例，如果有一个句子“the cat sits one the mat”，在训练的时候，将“the cat sits one the”作为输入，预测出最后一个词是“mat”。

三、实验流程

3.1 语料处理

实验的首要步骤已是老生常谈，首先读取语料库重的文本，去掉 txt 文件中一些无意义的广告和标点符号等内容，具体代码如下所示：

```
    def content_deal(content):  # 语料预处理，进行断句，去除一些广告和无意义内容
    ad = ['本书来自 www.cr173.com 免费 txt 小说下载站\n 更多更新免费电子书请关注
www.cr173.com', '----〖新语丝电子文库(www.xys.org)〗', '新语丝电子文库',
          '\u3000', '\n', '。', '? ', '! ', ', ', '; ', ': ', '、', '《', '》',
'"', '″', '`', '′', ' [', '] ', '....', '......',
          '『', '』', ' (', ') ', '…', '「', '」', '\ue41b', '<', '>', '+', '\x1a',
'\ue42b']
    for a in ad:
        content = content.replace(a, '')
    return content
```

进一步，利用 jieba 工具对语料分词。

```
def read_novel(path_in, path_out):  # 读取语料内容
    content = []
    names = os.listdir(path_in)
    for name in names:
        novel_name = path_in + '/' + name
        fenci_name = path_out + '/' + name
        for line in open(novel_name, 'r'):
            line.strip('\n')
            line = re.sub("[A-Za-z0-9\: \·\—\, \。\"\”\\n \《\》\! \? \、\...]",
"", line)
            line = content_deal(line)
            con = jieba.cut(line, cut_all=False) # 结巴分词
            content.append(" ".join(con))
        with open(fenci_name, "w", encoding='utf-8') as f:
            f.writelines(content)
    return names
```

3.2 模型训练

gensim 是一个 Python 的自然语言处理库，能够将文档根据 TF-IDF，LDA，LSI 等模型转换成向量模式。在本次实验中，我将利用 gensim 实现 word2vec，实现将单词转换为词向量的过程。引入的工具库语调用代码如下所示：

```
import multiprocessing
from gensim.models import Word2Vec
from gensim.models.word2vec import LineSentence

model = Word2Vec(sentences=LineSentence(name), hs=1, vector_size=200,
window=5, min_count=10, sg=0 ,epochs=200)
```

- sentence 是一个语料文本的列表，LineSentence 函数读入文件路径，将文本处理成“一行一文本”的格式。
- hs:参数若为 1，则该模型的训练采用 hierarchica softmax，如果设置为 0（default）则使用 negative sampling 的方式。
- vector_size 为词向量的维度，这里设置为 400；
- window 是一个句子中当前单词和被预测单词的最大距离。滑动窗口的大小；
- min_count 参数可以调整文本处理时可忽略的词频树。在不同大小的语料集中，我们对于基准词频需求也是不一样的，譬如在较大的语料集中，我们希望忽略那些只出现一两次的单词，这里设置为 5。
- sg 参数取值为{0, 1}，其决定了模型的训练算法：1: skip-gram; 0: CBOW.
- epochs:调用 Word2Vec(sentences, epoches=1)会调用句子迭代器运行两次(一般来说，会运行 iter+1 次，默认情况下 iter=5）。第一次运行收集单词和它们的出现频率，从而构造一个内部字典树；第二次运行负责训练神经模型。这里设置为 200。

四、实验结果与分析

本次实验选用《倚天屠龙记》、《天龙八部》、《射雕英雄传》、《神雕侠侣》、《鹿鼎记》作为样本，对其中的人物姓名与武功/道具关键词进行聚类分析。具体的方法即使用 gensim 工具中自带的 model.wv.similar_by_word 找出与给定关键词向量最相近的词集合。实验结果表明，关联度高的词语与小说的实际情节与背景相吻合。

4.1 实验样例 1：射雕英雄传

人名关联度聚类分析，郭靖和黄蓉关联性最大，其他出现的人名也和主人公在故事情节中紧密相关。然而，由于预处理时没有除去代词，使分析结果出现了没有意义的代词。

人物名称（对比实验：黄蓉）	关联度
郭靖	0.5589439272880554
她	0.44932934641838074
洪七公	0.4401666224002838
欧阳锋	0.42560017108917236
欧阳克	0.4125075936317444
陆冠英	0.37949222326278687
他	0.37014296650886536
完颜洪烈	0.3696635663509369
完颜康	0.3375110924243927
韩小莹	0.33431166410446167

武功关联度分析，关联词汇大多是武功名称。

武功名称（对比实验：降龙十八掌）	关联度
掌法	0.3234564960002899
殊	0.25227251648902893
所传	0.24950313568115234
阴毒	0.24859504401683807
落英	0.24640336632728577
逍遥游	0.37949222326278687
诀窍	0.24446988105773926
一阳指	0.23377646505832672
十五	0.2331300526857376
使	0.23217162489891052

4.2 实验样例 2：鹿鼎记

人名关联度分析，韦小宝和康熙关联最大。同样由于预处理的失误出现了“我”。

人物名称（对比实验：韦小宝）	关联度
康熙	0.6614575982093811
海老公	0.5741968750953674
了韦小宝	0.5640528798103333
郑克爽	0.558604896068573
太后	0.5575572848320007
双儿	0.5459750294685364
茅十八	0.4994068145751953
皇上	0.49672624468803406
小桂子	0.4869828522205353
我	0.48102328181266785

九阴真经的关联词出现了很多人名，即和《九阴真经》有关系的关键角色。

道具名称（对比实验：九阴真经）	关联度
真经	0.6614575982093811
经文	0.5741968750953674
上卷	0.5640528798103333
周伯通	0.558604896068573
欧阳锋	0.5575572848320007
经书	0.5459750294685364
洪七公	0.4994068145751953
法门	0.49672624468803406
郭靖	0.4869828522205353
黄药师	0.48102328181266785

4.3 实验样例 3:神雕侠侣

原理与上述实验保持一致。

人物名称（对比实验：小龙女）	关联度
杨过	0.7213680744171143
李莫愁	0.6130053997039795
陆无双	0.5894923210144043
郭靖	0.5886887907981873
黄蓉	0.5644753575325012
法王	0.5406153798103333
她	0.5286277532577515
赵志敬	0.526600182056427
周伯通	0.5286277532577515
绿萼	0.5230267643928528

武功名称（对比实验：一阳指）	关联度
弹指	0.6614575982093811
杨家枪	0.2907264530658722
打狗棒法	0.2797088027000427
真经	0.272583544254303
降龙十八掌	0.25541597604751587
而言	0.25286877155303955
一灯大师	0.25102031230926514
树林	0.24870480597019196
金轮法王	0.24725763499736786
藏僧	0.24485351145267487

4.4 天龙八部

对人名进行聚类与相似度的分析，从结果可以看，萧峰即是乔峰，段誉和虚竹为他的两

个兄弟，其他人物即和主人公发生事件与情感纠葛的关键人物。

人物名称（对比实验：乔峰）	关联度
萧峰	0.7213680744171143
段誉	0.6130053997039795
游坦之	0.5894923210144043
虚竹	0.5886887907981873
全冠清	0.5644753575325012
王语嫣	0.5406153798103333
段正淳	0.5286277532577515
慕容复	0.526600182056427
木婉清	0.5286277532577515
包不同	0.5230267643928528

人物名称（对比实验：降龙十八掌）	关联度
打狗棒法	0.39572709798812866
九阴真经	0.3353138267993927
掌	0.32246583700180054
南山	0.3158573508262634
蛤蟆功	0.30072158575057983
掌法	0.29732754826545715
空明拳	0.29624149203300476
招数	0.2768039107322693
一招	0.27557870745658875
老毒物	0.2714084982872009

4.4 倚天屠龙记

人物名称（对比实验：赵敏）	关联度
张无忌	0.6435161828994751
周芷若	0.6142603754997253
谢逊	0.5267996788024902
黄蓉	0.4897893965244293
张翠山	0.4804368019104004
鹿杖客	0.47520989179611206
胡青牛	0.47445693612098694
金花婆婆	0.46533268690109253
韦一笑	0.46436747908592224
殷离	0.46068739891052246

武功名称（对比实验：亢龙有悔）	关联度
落英	0.28696587681770325
经文	0.2826395630836487
竹棒	0.2751367688179016
火焰刀	0.2745816707611084
兵	0.2737678587436676
左掌	0.2669064998626709
商阳剑	0.2663888931274414
取水	0.2660776972770691
硬生生	0.2585292160511017
哪知	0.2503337264060974

本次实验利用 Word2Vec 模型进行了 Word Embedding 模型的训练，进一步根据训练得到的模型对五本小说的一些人名和武功绝学/道具进行了聚类，最终聚类得到的结果基本与小说内容相符。另外需要反思的是，在进行聚类分析的时候频频出现“她”、“他”等代词，这些词是没有意义的，以后在进行相关的分析任务时会将其去除，使训练效果更好。