成 绩 _____

北京航空航天大學
BEIHANG UNIVERSITY

# 深度学习与自然语言处理

# 第四次作业

| | |
|---|---|
| 院（系）名称 | 自动化科学与电气工程学院 |
| 专 业 名 称 | 电子信息 |
| 学 生 学 号 | ZY2103202 |
| 学 生 姓 名 | 黄君辉 |
| 指 导 教 师 | 秦曾昌 |

2022 年 5 月

# 四　　词向量聚类

## 一、问题描述

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、方法介绍

### 2.1 词向量

在自然语言处理任务中，首先需要考虑词如何在计算机中表示。通常，有两种表示方式：one-hot representation（离散表示）和 distribution representation（分布式表示）。

传统的基于规则或基于统计的自然语义处理方法将单词看作一个原子符号，这种方法被称作 one-hot representation。one-hot representation 把每个词表示为一个长向量。这个向量的维度是词表大小，向量中只有一个维度的值为 1，其余维度为 0，这个维度就代表了当前的词。例如：苹果: [0,0,0,1,0,0,0 ... ... ]。one-hot representation 相当于给每个词分配了一个 id，这就导致这种表示方式不能展示词与词之间的关系。另外，one-hot representation 将会导致特征空间非常大，但也带来一个好处，就是在高维空间中，很多应用任务线性可分。

分布式的方式通常称为 distribution representation，是将词转化为一种分布式的、连续的、定长的稠密向量，其优点是可以表示词与词之间的距离关系，每一维度都有其特定的含义。

两者的区别是用 one-hot 特征时，可以对特征向量进行删减，而分布式的则不可以。

生成词向量的方法有很多，这些方法都依照一个思想：任一词的含义可以用它的周边词来表示。生成词向量的方式可分为：基于统计的方法和基于语言模型(language model)的方法。基于统计的方法包括共现矩阵、奇异值分解等。

共现矩阵就是统计一个窗口内 word 共现次数，以 word 周边的共现词的次数做为当前 word 的 vector。该矩阵一定程度上缓解了 one-hot 向量相似度为 0 问题，但并没有解

决数据的稀疏性和高维性问题。奇异值分解则是针对共现矩阵存在的问题，提出了对原始词向量进行降维，从而得到一个稠密的连续词向量。利用 SVD 的方法，最终可以得到一个正交矩阵，进行归一化后即为词向量。该方法的优点是可以一定程度上反映语义相近的词，以及 word 间的线性关系；但由于很多词没有出线，导致矩阵及其稀疏，需要对词频做额外处理才能达到好的结果，并且其矩阵也是非常大，维度高。

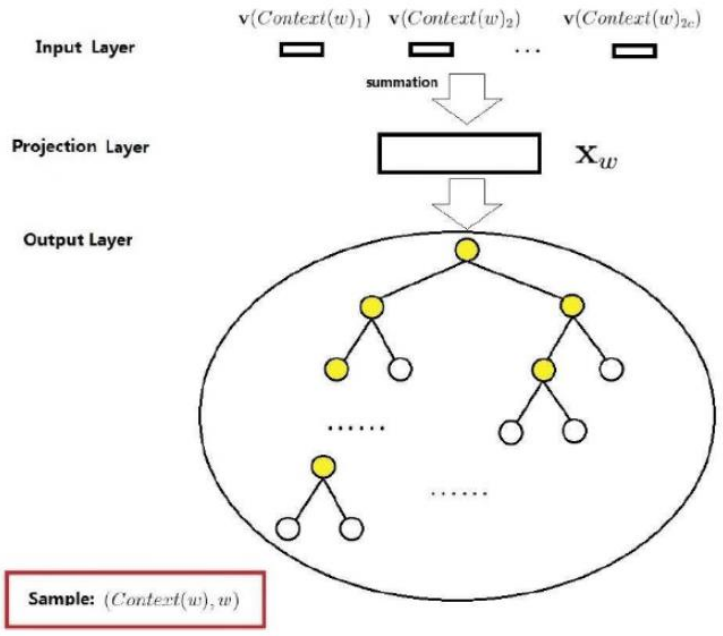
## 2.2 语言模型

语言模型生成词向量是通过训练神经网络语言模型 NNLM（neural network language model），词向量做为语言模型的附带产出。NNLM 背后的基本思想是对出现在上下文环境里的词进行预测，这种对上下文环境的预测本质上也是一种对共现统计特征的学习。

较著名的采用神经网络语言模型生成词向量的方法有：Word2Vec、LBL、NNLM、C&W、GloVe 等。本次作业采用的是常见的 Word2Vec 方法。

早期用于生成词向量的神经网络模型主要是 DNN，一般是采用三层神经网络结构，分别为输入层、隐藏层以及输出层。Word2Vect 方法就是在此基础上利用 Hierarchical softmax 和负采样进行改进，得到的方法主要分别为以下两种：CBOW 模型（连续词袋模型）和 Skip-gram 模型（跳字模型）。本次作业采用的是前者，因此接下来将介绍如何采用 CBOW 语言模型生成词向量。

下图是 Word2vec 基于 Hierarcical Softmax 优化的模型，训练模式选用 CBOW 模型：



该网络结构包含了三层，输入层，投影层（即原来的隐藏层）和输出层，假设存在样本$(Context(w), w)$,$Context(w)$是由 word 前后各 c 个词构成作输入样本$train_X$，word 作输出值$train_y$。

输入层：包含$Context(w)$中 2c 个词向量$v(Context(w)_1), \ldots, v(Context(w)_2c)$组成，词向量长度相同。

投影层：将输入层 2c 个词向量累加后求平均作为$X_w$。

输出层：输出层是对应一棵霍夫曼树，其中叶子节点就是对应词汇表中的词，非叶子节点即（黄色节点）等价于原来 DNN 模型中隐藏层到输出层的参数$W'$，用$\theta_i$表示该节点的权重，是一个向量，根节点是投影层的输出$X_w$。

Word2vec 相比较于 DNN 训练词向量，其网络结构有两点很大的不同：

舍去了隐藏层，在 CBOW 模型从输入层到隐藏层的计算改为直接从输入层将几个词的词向量求和平均作为输出;舍去了隐藏层到输出层的全连接结构，换成了霍夫曼树来代替隐藏层到输出层的映射。

第一个改进在于去除了隐藏层，Word2vec 训练词向量的网络结构严格上来说不算是神经网络的结构，因为其整个网络结构是线性的，没有激活函数并且取消了隐藏层。这么做对于训练词向量反而是极好的，我们在神经网络中使用激活函数，是因为我们处理的问题很多不是线性相关的，输入的样本之间一般也不是线性相关的。但处理词的问题时，我们知道一个词与其上下文是相关的，也就是说输入的上下文几个词也应该是线性相关的。取消了隐藏层没有了激活函数也就意味着承认了输入的几个上下文词的关系也是呈线性相关的。

第二个改变是为了针对降低原来 DNN 的 softmax 的计算量，我们把 softmax 计算改成了沿着一棵霍夫曼树找叶子节点的计算，这里霍夫曼树德非叶子节点相当与 DNN 中隐藏层到输出层的权重，在霍夫曼树中不需要计算所有的非叶子结点，只需要计算找寻某个叶子结点时经过的路径上存在的节点，极大的减少了计算量。

# 三、实验分析

## 3.1 实验设计

根据题目要求，本次实验主要分为以下几个步骤：

（1）对需要分析的金庸小说进行预处理，利用 jieba 库进行分词，去除其中多余的

符号和广告之类的无关内容，之后重新构成新的语料库。

（2）本次作业使用 python 中的自然语言处理库 Gensim 提供的接口来训练 Word2Vec 模型，采用的是 CBOW 模式，调用的函数如下：

```
model = Word2Vec(sentences=sentences, sg=0, vector_size=200, window=5, min_count=5, hs=1, epochs=100, workers=4)
```

Word2Vec 函数包括多个参数，下面对主要的参数进行简单介绍：

- sentences (iterable of iterables, optional)：供训练的句子，可以使用简单的列表，本次作业使用的语料库就是第一步中 jieba 分词得到的句子。
- vector_size(int, optional)：word 向量的维度。
- window (int, optional)：一个句子中当前单词和被预测单词的最大距离。
- min_count (int, optional)：忽略词频小于此值的单词。
- workers (int, optional)：训练模型时使用的线程数。
- sg ({0, 1}, optional)：模型的训练算法: 1: skip-gram; 0: CBOW.
- hs ({0, 1}, optional)：1:采用 hierarchical softmax 训练模型; 0:使用负采样。
- cbow_mean ({0, 1}, optional)：0: 使用上下文单词向量的总和; 1: 使用均值，适用于使用 CBOW。
- alpha (float, optional)：初始学习率。
- min_alpha (float, optional)：随着训练的进行，学习率线性下降到 min_alpha。
- seed (int, optional)：随机数发生器种子。
- epochs (int, optional)：迭代次数。
- batch_words (int, optional)：每一个 batch 传递给线程单词的数量。

（3）在模型训练完毕后，Gensim 库同样给出了一些接口函数，包括 most_similar()，similarity()等，它们可以给出训练之后的模型中，与某个给定输入词关联度最高的词或者是给定的某两个词之间的关联性，借由这些接口函数，可以验证词向量的有效性。

## 3.2 实验结果及分析

具体代码见 Jinhan-Lin/DL-NLP4 (github.com)

本次实验主要以个人较为熟悉的小说《天龙八部》为例，来分析 Word2Vec 模型得到词向量的准确性。

1.首先以“段誉”为例，查看相关词及其关联度

| “段誉”关联词 | 关联度 |
| --- | --- |
| 王语嫣 | 0.43105021 |
| 木婉清 | 0.38355538 |
| 王夫人 | 0.36098498 |
| 鸠摩智 | 0.35646450 |
| 萧峰 | 0.34551024 |
| 段正淳 | 0.33868089 |

可以看出，与段誉关联度最高的词就是他一直苦苦追求的“神仙姐姐”王语嫣；木婉清和段誉的感情也是一波三折，但是最后还是有情人终成眷属，段誉当上皇帝以后，迎娶木婉清为妻；王夫人是段誉父亲的情人，也是王语嫣的母亲；鸠摩智和段誉算是亦敌亦友，两人之间也存在着羁绊；萧峰作为段誉的大哥，给了段誉很大帮助；段誉的养父是段正淳，由段正淳抚养长大。

但是模型每次运行得到的结果还是会存在差异，例如某次运行的结果中就出现了“钟灵”“虚竹”，他们也是与段誉有着较为紧密的联系，甚至比上面的六个词中的一些更高，所以说模型的准确度和重复度还有待提高。

2.由训练好的 Word2Vec 模型也可以直接查看两个词向量之间的关联性，例如“乔峰”和“段正淳”之间可以看作是仇人，他们两个词的关联度为：0.36028725，与“段誉”和“王夫人”之间的关联度差不多。

考虑到 Word2Vec 模型的训练数据仅为《天龙八部》这一本小说，所以对不同词语之间关联度的计算并不完全准确；随着模型训练数据的增加，词向量之间关联度的计算应该会更加准确。

3.“乔峰”这种人名是专有名词，对神经网络模型来说比较容易辨认，它的关联词也全都为人名；例如其他“少林”“大理”等专有名词也可以取得类似的效果。接下来以动词“骑马”为例，检验 Word2Vec 模型对动词词向量的准确性。

| “骑马”关联词 | 关联度 |
| --- | --- |
| 乘马 | 0.28222001 |
| 马蹄声 | 0.26004645 |

| | |
|---|---|
| 两响 | 0.25313204 |
| 蚂蚁 | 0.25026404 |
| 七袋 | 0.24323409 |
| 女扮男装 | 0.24051413 |

可以看出，前两个关联词都很正常，骑马又可以称作乘马，骑马通常伴随着马蹄声；但后四个关联词则与骑马并无太大关系，并且他们的关联度与前两个词语的关联度相差并不多，明显存在问题。

除动词之外，由于 Word2Vec 模型词和向量是一对一的关系，所以多义词的问题它无法解决。以多义词“攻击”（既是名词也是动词）和“江湖”（含义较为丰富）为例，两个词语的关联词如下。

| “攻击”关联词 | 关联度 |
|---|---|
| 抵御 | 0.31741735 |
| 马蹄声 | 0.26004645 |
| 两响 | 0.25313204 |
| 蚂蚁 | 0.25026404 |
| 七袋 | 0.24323409 |
| 女扮男装 | 0.24051413 |

对“攻击”这种具有不同词性的词来说，模型取得的效果还算不错：抵御与攻击是反义词，攻击一般都要瞄准要害，出剑伴随着寒光。虽然基本上都是偏动词词性的，但是“攻击”这个词本身也算偏动词词性，所以结果也还算合理。

| “江湖”关联词 | 关联度 |
|---|---|
| 战场 | 0.33998131 |
| 世界 | 0.33093938 |
| 马鞍 | 0.32542836 |
| 中原 | 0.32082286 |
| 大厅 | 0.29602655 |
| 石壁 | 0.29186839 |

对“江湖”这种含义比较丰富的词来说，甚至每个人的理解都会有很大区别，模型

给出的结果里面，“战场”“世界”，都算是与江湖相关的词，前者认为江湖大多伴随着战斗，后者认为江湖与世界一样，都是一种对于某种事物的集合。或许每个人心中都有不同的江湖，而这也正是金庸小说的魅力所在。

4. Gensim 提供的练 Word2Vec 函数中包括多个参数，不同参数导致训练结果也存在一定差异，例如初始参数下“段誉”的关联词如 1 中表格所示，但是更改相关参数：

```
model = Word2Vec(sentences, sg=0, vector_size=200, min_count=10, hs=1,  workers=4)
```

得到的结果如下：

| “段誉”关联词 | 关联度 |
|---|---|
| 王语嫣 | 0.70167595 |
| 慕容复 | 0.66917926 |
| 木婉清 | 0.61417192 |
| 钟灵 | 0.61041945 |
| 段正淳 | 0.60222852 |
| 乌老大 | 0.59277266 |

可以看出，词语之间的关联度有很明显的提升，而且个人觉得这一参数下得到的结果相对于前一参数更为准确。在改变迭代次数 epochs 后（采用默认的 epochs=20），模型训练需要的时间也大幅减少。

## 3.3 总结

Word2vec 是 Word Embedding 的方法之一，将不可计算、非结构化的词转化为可计算、结构化的词向量；这一步解决的是“将现实问题转化为数学问题”，是人工智能非常关键的一步。CBOW 和 Skip-gram 是 Word2Cec 的两种训练模式。为了提高速度，Word2Vec 经常采用 Hierarchical Softmax 和 Negative Sample 2 种加速方式。

由于 Word2Vec 会考虑上下文，跟之前的 Embedding 方法相比，效果要更好；比之前的 Embedding 方法维度更少，所以速度更快；并且 Word2Vec 模型通用性很强，可以用在各种 NLP 任务中。但缺点是 Word2Vec 模型对多义词的问题很难解决；并且 Word2vec 是一种静态的方式，虽然通用性强，但是无法针对特定任务做动态优化

# 四、收获、体会及建议

通过此次作业，我对词向量有了更加直观的理解，也对构建词向量的相关模型有了

更深刻的认识，学习了利用 Word2Vec 模型构建词向量的相关方法，对自然语言处理的相关内容有了更深入的认识。