**Word2vec训练中文词向量并生成《斗破苍穹》风格的自然语言**

**陈春祺**

学号:5140379022

Email:simoncqchen@outlook.com

**摘 要：**本论文基于谷歌提出的word2vec模型，提出了通过word2vec模型训练中文词向量的方法，实现了word2vec的skip-gram模型，以小说《斗破苍穹》作为语料库训练了中文词向量；此外，本论文还实现了基于LSTM结构的语言模型，在训练过程中生成字向量，并生成与语料库具有相似语言风格的自然语言。

**关键词:** word2vec，词向量，LSTM，自然语言生成

# 引言

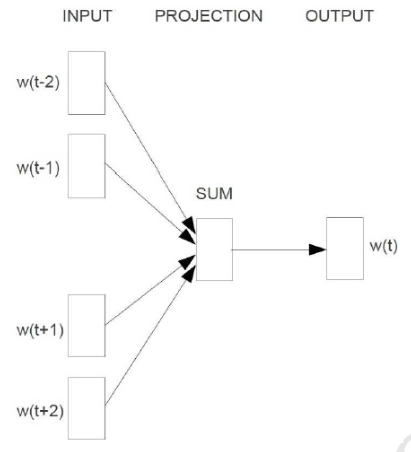
在自然语言理解问题中，为了应用神经网络模型，需要将自然语言转化为向量输入。如何将自然语言转化为向量，使得向量能准确表达自然语言的表层和深层含义，并反映出自然语言之间的依赖关系，是基于深度学习、神经网络等方法的自然语言理解问题的基础。2013年，Tomas Mikolov等人在论文中提出了word2vec模型，可以将自然语言的组成单元——词语——转化为向量表示，通过考察语料库之中上下文的出现规律，训练模型得到每个词的向量表示[1]。Word2vec是目前应用比较广泛的获得词向量的方式，如果语料库较为全面，就可以得到相对准确的向量表示，为后续问题的研究提供了良好的基础。原始word2vec模型针对英文词语进行训练，本文由于实验室项目需要，实现了基于word2vec的中文词向量训练。

此外，自然语言生成问题也是当前备受关注的一个领域，可以应用于生成数据集、对话机器人、辅助写作、生成报告等领域。本文对自然语言生成问题进行了尝试，实现了基于RNN变体LSTM[2]的语言模型，选取小说《斗破苍穹》为原始数据集，以字符为单位对语言模型进行了训练，并最终生成了一系列自然语言文本，经过实验可以看出我们的模型比较准确的把握了小说中字符的含义和独特的语言风格，达到了我们的预期。

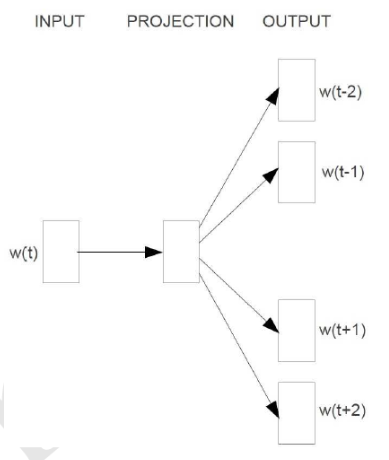
# Word2vec中文词向量模型

## Word2vec模型介绍

word2vec对词向量的训练有两种方式，一种是CBOW模型，即通过上下文来预测中心词（图1）；另一种skip-Gram模型，即通过中心词来预测上下文（图2）[1]。其中CBOW对小型数据比较适合，而Skip-gram模型在大型的训练语料中表现更好。



**图1. CBOW模型**



**图2. Skip-gram模型**

CBOW模型包含三层：输入层、投影层和输出层。CBOW模型以上下文预测中心词，输入输出样本为，由w前后各c个词组成。模型的三层分别为：

输入层，包含中2c个词的词向量、、……、；

投影层，将输入层的2c个向量求和累加，如公式(1)所示；

输出层，对投影层的结果进行权重相乘、添加偏置值的计算，输出结果，作为对于中心词向量的预测结果。

类似的，skip-gram也包含三层，与CBOW不同的地方在于，由于skip-gram是从中心词预测上下文词语，输入层只有一个词向量，所以skip-gram的结构中的投影层是一个恒等投影，不对输入层的数据进行变换。之所以保留这样一个投影层是为了方便与CBOW模型进行对比。

## 中文词向量的训练

本文实现了word2vec中的Skip-gram模型。此模型对于中文词向量的训练步骤如下：

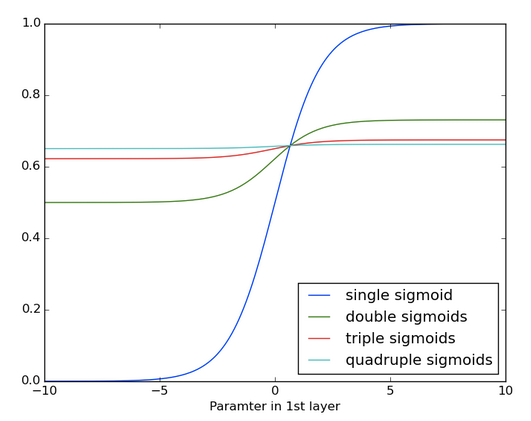
1. 使用Jieba分词，将原始语料库进行分词，按照词语出现的顺序将所有词语记录入一个列表。
2. 统计词频、词典等信息，将生僻词替换为UNKNOWN，将之前记录所有词语的列表转换为词语在词典中的序列的列表。
3. 构建skip-gram模型所需的训练数据。本文实现的模型为skip-gram，即根据中心词预测上下文，所以将记录所有词语序列的列表作为输入，而输入词语在原文中的上下文词语作为Label。具体来说，如果原文中存在“abcde”这句话，a、b、c、d、e分别为词语，上下文指中心词前后各一个词，那么根据这句话生成的输入数据与Label对如下：（b，a）、（b，c）、（c，b）、（c，d）、（d，c）、（d，e）。所有的训练数据生成完成后，再切分为minibatch。
4. 定义skip-gram的损失函数，在这里使用了tensorflow的nce\_loss函数，应用了负采样机制，提高训练速度并改善了最终训练结果中词向量的质量。使用梯度下降法来更新权值。
5. 对所有的输入数据进行训练，损失收敛之后记录当前的词向量作为结果。

# 基于RNN的语言模型生成自然语言

## RNN与LSTM

由于自然语言的特殊结构，在研究中很适合使用递归神经网络来建立语言模型。传统的语言模型包括n-gram语言模型和神经概率语言模型，和这两种语言模型相比，基于RNN的语言模型可以充分利用之前出现的所有词语的信息，并且处理的成本不会随着依赖上下文的增加而指数级增长。

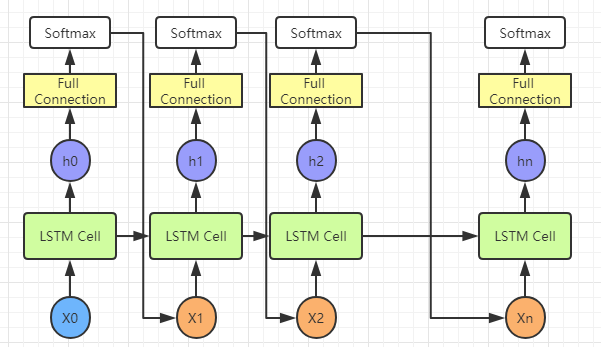
然而，在上世纪九十年代早期，梯度消失问题成为影响循环网络表现的重大障碍。正如直线表示x如何随着y的变化而改变，梯度表示所有权重随误差变化而发生的改变。如果梯度未知，则无法朝减少误差的方向调整权重，网络就会停止学习。原因之一是， 神经网络中流动的信息会经过许多级的乘法运算。由于深度神经网络的层和时间步通过乘法彼此联系，导数有可能消失或膨胀。梯度膨胀时，这些权重的梯度增大至饱和，亦即它们的重要性被设得过高。但梯度膨胀的问题相对比较容易解决，因为可以将其截断或挤压。而消失的梯度则有可能变得过小，以至于计算机无法处理，网络无法学习。这个问题更难解决。反复应用sigmoid函数的结果如图3所示。数据曲线越来越平缓，直至在较长的距离上无法检测到斜度。梯度在经过许多个层后消失的情况与此相似。



**图3. 多次应用Sigmoid后函数的曲线**

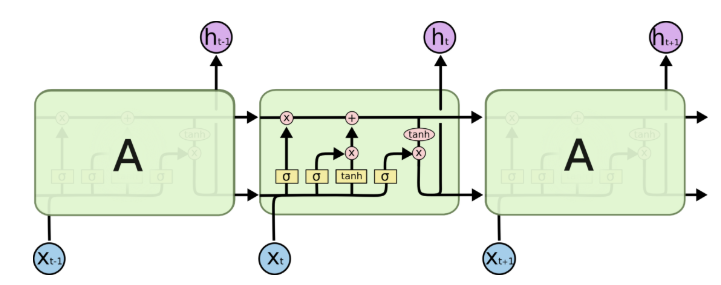
九十年代中期，德国学者Sepp Hochreiter和Juergen Schmidhuber提出了循环网络的一种变体，带有所谓长短期记忆单元，或称LSTM，可以解决梯度消失的问题[2]。LSTM可保留误差，用于沿时间和层进行反向传递。LSTM将误差保持在更为恒定的水平，让循环网络能够进行许多个时间步的学习（超过1000个时间步），从而打开了建立远距离因果联系的通道。LSTM将信息存放在循环网络正常信息流之外的门控单元中。这些单元可以存储、写入或读取信息，就像计算机内存中的数据一样。单元通过门的开关判定存储哪些信息，以及何时允许读取、写入或清除信息。但与计算机中的数字式存储器不同的是，这些门是模拟的，包含输出范围全部在0~1之间的sigmoid函数的逐元素相乘操作。相比数字式存储，模拟值的优点是可微分，因此适合反向传播。这些门依据接收到的信号而开关，而且与神经网络的节点类似，它们会用自有的权重集对信息进行筛选，根据其强度和导入内容决定是否允许信息通过。这些权重就像调制输入和隐藏状态的权重一样，会通过循环网络的学习过程进行调整。也就是说，记忆单元会通过猜测、误差反向传播、用梯度下降调整权重的迭代过程学习何时允许数据进入、离开或被删除。在控制信息是否通过的过程中，LSTM很好的控制了梯度的变化，减轻了梯度消失的问题。基于这样的情况，本课题的语言模型将基于RNN的变体LSTM来建立。

## 模型结构



**图4. 本文建立的语言模型**

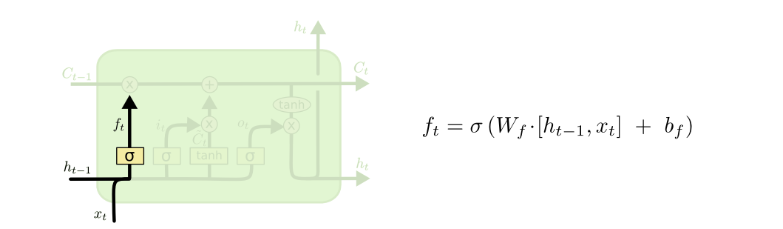
图4为本课题所建立的语言模型，模型初始的输入为x0，表示最初提供的字向量，文本的生成会从这个初始字符开始，经过LSTM Cell的计算得到当前的隐藏状态h0，将h0作为下一时刻的一部分输入；对h0进行一次全连接层的计算，经过Softmax层预测出下一时刻可能出现的字符，即x1，作为下一时刻的另一部分输入。循环这个过程，直到生成足够多的字符。



**图5. LSTM Cell结构**

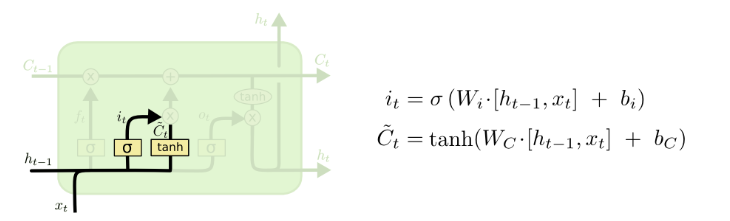
本文所应用的LSTM Cell结构如图5所示，当前LSTM的输入为、，LSTM Cell的计算过程如下：

1. 输入向量通过遗忘门，得到，计算公式如图6所示。



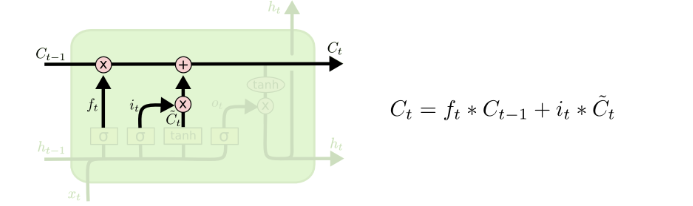
**图6. LSTM Cell遗忘门**

1. 这一步决定什么样的信息要被保留在当前细胞状态中，包括两个部分，一个输入门，进行sigmoid函数操作，决定将要被更新的值；一个tanh层，创建一个新的候选向量。计算公式如图7所示。



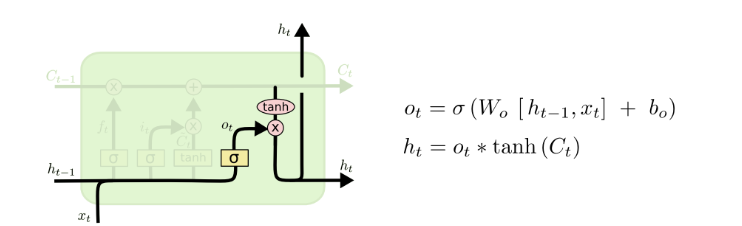
**图7. LSTM Cell输入门与tanh层**

1. 根据上一步生成的信息，更新上一时刻的细胞状态，公式如图8所示。



**图8. LSTM Cell更新细胞状态**

1. 决定需要输出的值，根据当前细胞状态和输入向量，通过sigmoid和tanh层计算得到隐藏状态，公式如图9所示。



**图9. LSTM Cell计算隐藏状态**

LSTM计算得到隐藏状态，对隐藏状态进行一次全连接层和Softmax层的计算（公式(2)），得到下一时刻输出字符的概率。

在训练过程中发现，如果直接选择概率最高的字符进行输出，生成过程会陷入循环，即不断生成之前出现过的句子，所以这里模型会选择概率大小前五的字符，将这五个字符的概率归一化，再根据它们归一化之后的概率随机选择其中一个字符作为下一时刻生成的字符，这个字符的子向量会和一起作为下一时刻的输入，重复进行计算，直到生成足够的字符，最后就是模型根据初始字符生成的自然语言文本。

# Word2vec中文词向量训练

## 数据集

本模型所使用的自然语言语料库是小说《斗破苍穹》， 去除空格之后总词数3427356个，总字符数5260233；实验中使用的停用词表大小为1893。

## Word2vec词向量实验

本节描述了word2vec进行中文词向量训练的实验过程，实验中，训练参数的设置如表1所示。

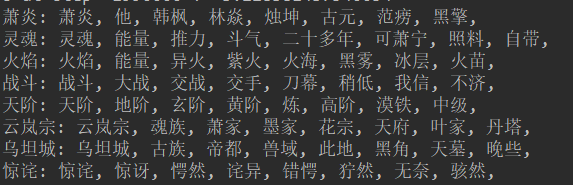
**表1. Word2vec中文词向量训练参数设置**

|  |  |
| --- | --- |
| 参数名 | 值 |
| 词典大小 | 50000 |
| batch size | 128 |
| 词向量维度 | 128 |
| 上下文窗口大小 | 1 |
| 学习率 | 1.0 |
| 负采样个数 | 64 |
| 训练轮数 | 2000000 |

训练过程中损失随训练轮数的变化如图10所示。

**图10. 训练过程中损失随训练轮式的变化**

训练完后，选取了几个实例词语和词典中与它们最接近的词，如图11所示。可以看出，得到的结果是比较符合自然规律的，对于一些通用性的词语理解的很好，能比较准确的把握它们的语义，对于语料库中特有的词语，也能找到语料库中在语义上与其相似的词语。目前所使用的语料库还比较小，是一本单一的小说，如果增加语料库的大小，应该可以得到更好的结果。



**图11. 词向量和同义词的示例**

# RNN语言模型的训练和实验

## 数据集

本模型所使用的自然语言语料库是小说《斗破苍穹》， 去除空格之后总词数3427356个，总字符数5260233。

## 训练参数设置

本节描述了RNN语言模型进行自然语言文本生成的训练过程，训练参数的设置如表2所示。

**表2. RNN语言模型训练参数设置**

|  |  |
| --- | --- |
| 参数名 | 值 |
| 全连接层隐藏层节点数 | 512 |
| rnn层数 | 2 |
| 词向量维度 | 50 |
| dropout概率 | 0.5 |
| 字典大小 | 4060 |
| batch size | 64 |
| 学习率 | 0.001 |
| 学习率衰减率 | 0.98 |
| 学习率衰减轮数 | 50 |
| 最大梯度 | 5 |
| 训练轮数 | 24 |

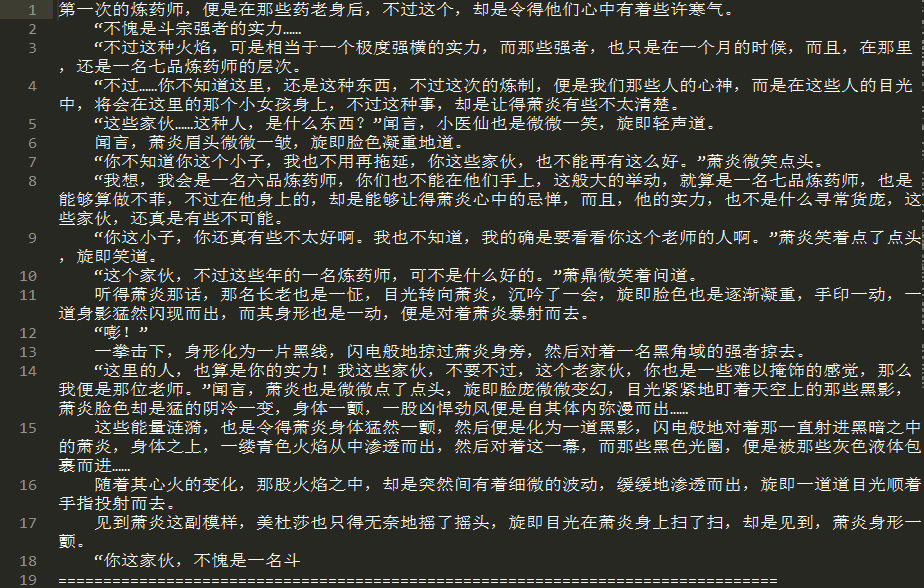
## 实验结果评估

语言模型常用的评估标准为Perplexity（PPL）[3]，它主要是根据每个词来估计一句话出现的概率，PPL公式如公式（3）所示。

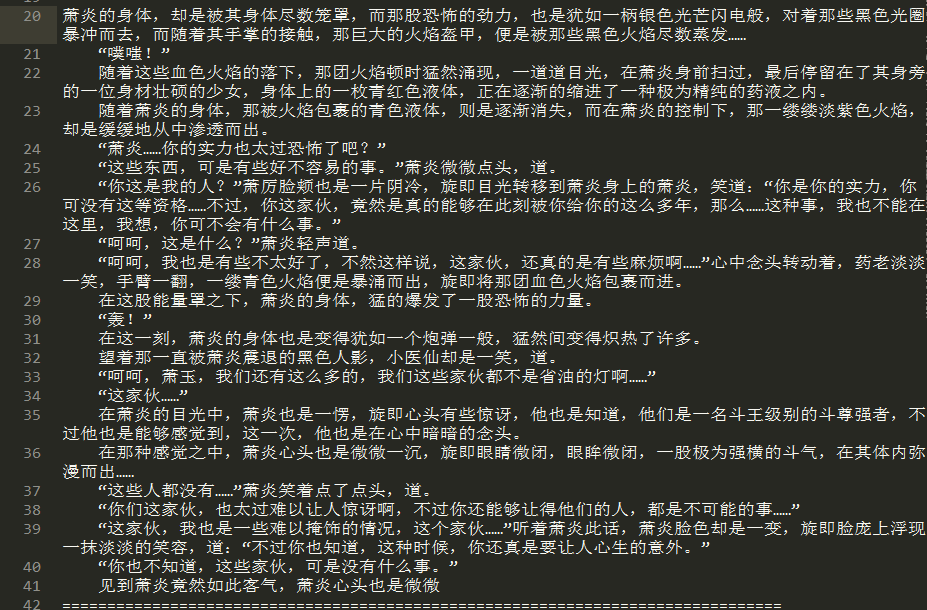
其中，S代表句子，n为句子长度，越小，代表句子出现的期望越高，在训练过程中，生成句子的PPL随训练轮数的变化如图12所示。

**图12. PPL随训练轮数的变化**

可以看出，随着训练轮数的增加，生成语句的PPL逐步减小，并趋于收敛。模型训练完成后，对于模型的生成能力进行了实验，给定几个初始字符，由模型生成的文本如图13、图14所示。



**图13. 训练完成后模型生成的文本（1）**



**图14. 训练完成后模型生成的文本（2）**

可以看出，最终训练完成后，模型已经比较准确的理解了字符的含义，也很好的把握了原小说的语言风格；模型根据提供的初始字符所生成的几段文本，在遣词造句等方面都比较通顺，行文风格也和原小说十分相似。但从整体来看，整段文字的整体含义还是比较混乱，如果要进一步提升生成文本的质量，可能需要从这个方面入手。

# 总结与展望

本文实现了word2vec的中文词向量模型，并以小说《斗破苍穹》作为语料库进行了训练，达到了比较准确的结果，词向量之间的关系很符合自然语言之间的规律，比较精确的理解了词语的意思。此模型的语料库可以被替换成其余的通用语料，如维基百科、StackOverflow等大型语料，相比于本文所使用的单一的小说，应该可以得到更为广泛和更通用的结果。

此外，本文还实现了基于RNN变体LSTM的语言模型，对自然语言文本生成问题进行了尝试，同样使用《斗破苍穹》作为原始数据集；模型经过训练之后，所生成的文本行文流畅，且与原始数据集风格相似，可以看出模型比较准确的把握了原始数据集字符的含义和组合规律，达到了我们预期的成果。

在接下来的工作中，为了把中文词向量模型投入实际应用，应该会使用软件工程、计算机相关领域的语料来进行词向量的训练，同时基于Hierarchical Softmax和Negative Sampling两种机制，实现word2vec中的skip-gram模型和CBOW模型，使我们的模型趋于完善，具有更强的实用性。

对于本文实现的语言模型，接下来可能会尝试增加迭代轮数，加入Attention[4]和强化学习等机制来提升文本生成能力，使生成的文本在整体上更有意义。

References (参考文献)

1. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient estimation of word representations in vector space[J]. ICLR Workshop, 2013.
2. Hochreiter, S, and J. Schmidhuber. Long short-term memory[J]. Neural Computation 9.8(1997):1735-1780.
3. Brown, Peter F., Stephen A. DellaPietra, Vincent J. DellaPietra, Jennifer C. Lai, and Robert L. Mercer. 1992. An estimate of an upper bound for the entropy of English[J]. Computational Linguistics, 18(1):31–40, March.
4. V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. Recurrent models of visual attention. In Proceedings of NIPS, pages 2204–2212, 2014.