

硕 士 学 位 论 文

中文词汇语义相似度计算研究

Research on Chinese Word Semantic Similarity Computation

作 者 姓 名: 裴家欢

学 科、 专 业: 计算机应用技术

学 号: 21409138

指 导 教 师: 黄德根 教授

完 成 日 期: 2017 年 5 月 9 日

大连理工大学

Dalian University of Technology

大连理工大学学位论文独创性声明

作者郑重声明：所呈交的学位论文，是本人在导师的指导下进行研究工作所取得的成果。尽我所知，除文中已经注明引用内容和致谢的地方外，本论文不包含其他个人或集体已经发表的研究成果，也不包含其他已申请学位或其他用途使用过的成果。与我一同工作的同志对本研究所做的贡献均已在论文中做了明确的说明并表示了谢意。

若有不实之处，本人愿意承担相关法律责任。

学位论文题目：_____

作 者 签 名 : _____ 日期 : _____ 年 ____ 月 ____ 日

摘要

词汇语义相似度是对两个词语对象所蕴含意义的相近程度的度量。词汇语义相似度计算是一项基础而核心的工作，可以将“词汇相似”这个抽象关系通过特定的计算方法映射成计算机可以处理的数值，从而将自然语言处理问题转化为机器学习问题，其性能的好坏将直接影响到自然语言处理与信息检索的各项任务。近年来，基于词向量的词汇语义相似度计算方法及其改进方法已成为该领域内的前沿、热点研究课题。

本文研究中文词汇语义相似度计算，重点研究如何改进基于词向量的方法，并根据是否融入语义约束分为两部分研究：

(1) 无语义约束的词向量模型

本文分别使用机器翻译和 LSTMs 网络改进标准 Skip-gram 模型：首先，分别利用标准 Skip-gram 模型根据不同的训练语料得到标准词向量，通过实验分析了语料规模和语料质量对词向量模型的影响。其次，尝试用机器翻译构建中、英文词向量的关系，并用公开大规模英文词向量选择性地替换中文向量，改进原有中文词向量计算性能。最后，将词汇相似度计算问题转化为词汇关系预测问题，并通过 LSTMs 网络学习词对共现的句子序列，从而建模词汇关系。

(2) 融入语义约束的词向量模型

本文提出了一种改进 Counter-fitting 模型将语义约束融入已有词向量模型：首先，利用网络爬虫技术扩展词汇上下文。分别抓取词汇出现以及词对共现的句群作为词对“上下文”，并抓取同义词对、反义词对扩充现有人工语义词典。其次，计算词汇语义相似度。分别利用语义词典、检索结果、预训练好的词向量计算语义相似度。最后，利用改进 Counter-fitting 方法优化预训练好的词向量，具体的做法是，通过语义约束、向量空间存留构造多项式目标函数，然后用梯度下降算法对目标函数求解，其中，语义约束不仅包括同义约束、反义约束，还包含了相似约束。

实验结果表明，基于语义词典的方法在登录词覆盖率较高的情况下，有着先天优势，而当出现大量未登录词时，基于词向量方法和基于 Web 检索的方法更具有实用性，此外，融入语义约束的词向量模型的实验结果达到目前 PKU-500 数据集上的最佳水平，斯皮尔曼相关系数为 0.552，其性能明显好于语义词典模型、Web 检索模型和基本词向量模型。

关键词：语义相似度计算；词向量；LSTMs；语义约束

Research on Chinese Word Semantic Similarity Computation

Abstract

Word semantic similarity is a measure of the degree of similarity in terms of the meanings of two words. Word semantic similarity computation is a fundamental and core task, and it can be used to map the abstract relationship of “similarity between words” into real value, therefore, a natural language processing problem can be transformed into a machine learning problem. Its performance will directly affect various tasks in the natural language processing and information retrieval field. In recent years, computing word semantic similarity using embedding-based methods and its improved methods has become the frontier and hot research topic in this field.

In this thesis, we study on the “Chinese Word Semantic Similarity Computation”, and mainly focus on how to improve the word similarity computation based on word embeddings and divide the research into two parts:

(1) Embedding-based Word Similarity Computation without Semantic Constraints

We use machine translation techniques and LSTMs network to improve a standard Skip-gram model respectively: firstly, the standard Skip-gram model is used to obtain a basic word embedding according to different training corpus. The influence of the size and quality of corpus on word embedding model is analyzed experimentally. Secondly, we try to construct the relationship between Chinese and English words by machine translation, to be more specific, we use large-scale English word embedding to alternatively replace the Chinese one to get a better performance. Finally, the problem of word similarity computation is transformed into word relationship prediction problem, and the word relationship is constructed by learning the coherent sentence through LSTMs network.

(2) Embedding-based Word Similarity Computation with Semantic Constraints

In this paper, we propose an improved Counter-fitting model to incorporate semantic constraints into a pre-train word embedding to compute word semantic similarity: firstly, we use web crawler to expand the context of the words. Specifically, we capture the sentences that a word occurs or word pairs co-occur as the “context”, and we get some synonyms and antonyms to expand the existing manual semantic lexicons. Secondly, we compute the word semantic similarity using semantic lexicons, retrieval results and pre-trained word vectors. Finally, the improved counter-fitting model is used to optimize the pre-trained word vectors. The concrete approach is to construct the polynomial objective function by semantic constraints and topological space reservation, and then use the gradient descent algorithm to solve the

objective function. The semantic constraints include not only synonym constraints and antonym constraints, but also the similarity constraints.

The experimental results show that the method based on semantic lexicons has the inherent advantages in the case of high coverage of known words. While methods based on word embedding and retrieval are more practical when there are a large number of unknown words. In addition, the counter-fitting method that incorporates semantic constraints into word embedding get the state-of-the-art performance on PKU-500 dataset with a Spearman's rank correlation coefficient of 0.552, which outperforms the performance of semantic lexicon-based model, retrieval-based model and embedding-based model.

Key Words: Semantic Similarity Computation; Word Embedding; LSTMs; Semantic Constraints

目 录

摘要.....	I
Abstract.....	II
1 绪论.....	1
1.1 研究背景与意义.....	1
1.2 研究现状与分析.....	2
1.2.1 基于语义词典资源的方法.....	2
1.2.2 基于语料库的传统统计方法.....	3
1.2.3 基于语料库的词向量方法.....	3
1.2.4 基于组合多策略的方法.....	4
1.2.5 研究的重点和难点分析.....	5
1.3 本文的主要工作.....	6
1.3.1 本文的研究内容.....	6
1.3.2 本文的创新之处.....	6
1.4 本文的组织结构.....	7
2 理论基础.....	8
2.1 词汇语义相似度.....	8
2.1.1 基本概念.....	8
2.1.2 任务描述.....	8
2.2 语义词典资源.....	9
2.3 词表示及空间向量距离.....	12
2.3.1 词表示方法.....	12
2.3.2 Word2vec 模型.....	13
2.3.3 空间向量相似度.....	16
2.4 语言模型与人工神经网络.....	17
2.4.1 统计语言模型.....	17
2.4.2 神经语言模型.....	18
2.4.3 循环神经网络.....	19
3 基于词向量的词汇语义相似度计算.....	23
3.1 标准 Skip-gram 词向量模型.....	23
3.2 基于机器翻译的改进方法.....	24

3.3 基于 LSTMs 的改进方法	25
3.4 实验与分析.....	27
3.4.1 实验语料.....	27
3.4.2 评价方法.....	28
3.4.3 结果与分析.....	28
4 融入语义约束的词向量模型.....	31
4.1 模型构建.....	31
4.2 具体实现.....	33
4.2.1 预处理.....	34
4.2.2 计算词汇相似度.....	34
4.2.3 最优化词向量.....	37
4.3 实验与分析.....	39
4.3.1 改进模型参数调整实验.....	39
4.3.2 改进模型的稳定性分析.....	42
4.3.3 改进模型的有效性分析.....	42
4.3.4 不同模型上的对比实验.....	45
4.3.5 其它语料上的对比实验.....	47
结 论.....	48
参 考 文 献.....	50
攻读硕士学位期间发表学术论文情况.....	54
致 谢.....	55
大连理工大学学位论文版权使用授权书.....	56

1 绪论

1.1 研究背景与意义

随着互联网时代的到来，海量信息不断生成，信息过载问题给人们带来的影响日益显著。如何处理如此规模巨大的互联网信息资源，对数据挖掘和文本理解提出了新要求。其中，如何自动理解自然语言中所隐含的语义，从而迅速而准确地获取有效信息，是自然语言处理（Natural Language Processing, NLP）与信息检索（Information Retrieval, IR）领域亟待解决的基础而核心的问题之一。

词汇是自然语言中可以表示完整语义的基本单元，是理解一切自然语言的根本。语义是指自然语言所表达的含义，能够代表其在现实世界中所对应事物的概念。而如何量化词汇语义相似性是人工智能科学利用计算机模仿人类对相似性认知过程的核心步骤。为此，研究者提出词汇语义相似度作为一种对两个词汇对象所表达含义相近程度的度量。

词汇语义相似度计算是一项基础性工作，其研究的是基于特定度量标准、通过量化方式来评价词汇之间的语义距离或者相近程度，并将“词汇相似”这个抽象关系通过特定计算方法映射成计算机可以处理的数值。这样便可以将自然语言处理问题转化为机器学习问题，并利用各种机器学习算法自动完成 NLP 和 IR 领域的诸多任务需求。

词汇的语义相似度计算研究的应用场景非常广泛，而且其性能的好坏将直接影响到各项任务，例如：

（1）词义消歧

词义消歧主要任务是对词汇的多个义项进行区分，其中，词汇的相似程度是理解多义项之间语义关系的重要手段。Sinha 和 Mihalcea^[1]结合了多种语义相似性度量方法和图中心算法，提出了一个无监督词义消歧模型，实验结果表明词义消歧的性能明显提升。于东和荀恩东^[2]利用词向量的语义相似度对所用特征词进行扩展，并利用两步聚类算法实现了对缩略术语的消歧。

（2）关系抽取

现有研究表明，丰富的语义信息能够有效地改善实体关系抽取效果。Pirrò 等^[3]利用基于词语语义相似度的词汇匹配器，用于发现实体之间的语义关系，从而构建实体之间的映射。Meilicke 等^[4]利用词语语义相似度计算方法能有效的修复本体的映射结果，从而避免因语义异构问题而引发映射错误。

(3) 查询扩展与推荐

查询扩展与推荐是词汇语义相似度计算另一个重要的应用场景。Qumsiyeh 和 Ng^[5]利用查询日志中的关键字相似性计算、出现频率、查询的修改模式，提出了 WebQS 模型解决查询推荐问题。Pal 等^[6]综合考虑术语之间的语义相似性及术语罕见度，使用 WordNet 中对查询语句进行扩展。

(4) 文本相似度计算

词汇是组成文本的最小基本单元，因此，文本级别的相似度计算自然也离不开词汇级别的语义相似度计算。Mihalcea 等^[7]提出了一个基于语料库和知识库的词汇相似度来衡量两个短文本（句子或者段落）的相似度。Islam 和 Inkpen^[8]使用语料库统计方法计算词的语义相似度，并利用字符串匹配算法得出篇章级别文本的相似度。

1.2 研究现状与分析

迄今为止，国内外学者相继提出许多词汇相似度计算的研究方法，本文综合其计算对象及使用技术，大致将现有方法分为四类：基于语义词典资源的方法，基于语料库的传统统计方法，基于语料库的词向量方法和基于组合多策略的方法。

1.2.1 基于语义词典资源的方法

语义词典资源是指通过人工构建或者自动构建并人工校正后得到的描述词汇之间语义关系的结构化知识库。基于语义词典资源的方法，主要关注语义词典结构特点及词汇之间语义关系，计算词对中词汇之间的语义距离，从而定义相似度计算公式，其中，常见语义关系包括上下位关系、同义关系、反义关系等^[9]。根据使用语义词典资源的不同，常见方法主要可分为：基于同义词林的方法^[10-11]，基于知网（HowNet）的方法^[12-13]，基于 WordNet 的方法^[14-16]，基于百度百科或维基百科的方法^[17-19]，基于语义框架 FrameNet 的方法^[20]等。此外，还有也有不少学者开始提出多种语义词典资源融合的方法^[21-23]。

这些基于语义词典资源的词汇相似度计算方法，虽然可解释性强、计算算法复杂度低，但是存在一个致命的缺点，即只有当词对中的两个成员全部出现在语义词典中才能够给出有效的相似度评价。而事实上，这种假设并不合理，因为当用户提交的查询中包含网络流行新词时，例如“葛优瘫”，往往很难在上述提到的语义词典中找到对应的词项。这极大限制了词汇相似度的应用范围，更加无法适用于时效性较强的 IR 领域任务。因此，如何根据大规模语料库计算词汇相似度吸引了更多学者的目光。

1.2.2 基于语料库的传统统计方法

语料库是人们为了特定应用目标而专门收集和整理的大量文档集合^[24]，广义而言，语料库可以是静态、离线的文档数据集，也可以是动态、在线的 Web 数据集。基于大规模离线语料库的传统研究中，本文将主流研究方法^[25]归纳为：基于相关熵、互信息等的信息论模型，基于词语共现等的概率模型，基于词语属性的特征空间模型及基于 LSA、LDA 等的语义空间模型等。

离线语料库难以实时更新，仍然无法解决未登录词相似度计算问题，在线 Web 语料库的出现为基于语料库的相似度计算研究提供了新思路。许多学者通过搜索引擎获取词语出现的文档，并根据词语出现、共现及上下文语义片段信息来定义相似度，或者将搜索引擎返回文档结果数等统计数据作为词汇相似度计算的重要依据，具有代表性的研究包括：Turney 提出的 PMI-IR 算法^[26]、Higgins 提出的 LC-IR 算法^[27]和 Bollegala 等提出的 Web-PMI 算法^[28]，Rudi 提出的 NGD 算法^[29]等。基于在线 Web 语料库的算法虽然解决了语料库过度依赖问题，但是存在数据稀疏、数据噪声以及计算失效等问题。

1.2.3 基于语料库的词向量方法

词向量（Word Vector），又称为词嵌入（Word Embedding），是基于上下文与目标词之间关系建模得到的一种分布式表示，是一种含有语义信息的实数型数值向量^[30]。基于语料库的词向量方法，往往将词汇相似度计算的重心放在词向量模型上，即如何将字符串型的文字一一映射到实数域上的语义空间中，然后利用空间距离公式定义词汇语义相似度。

较为知名的词向量模型主要包括：Collobert 等^[31]提出了一个由查表层、卷积层、hard tanh 层、max 层组成的神经网络框架来解决多种 NLP 任务，并将相关任务的标注语料应用到有监督的联合训练过程中得到词向量。Socher 等^[32]利用递归自编码器来解决情感分类问题，并联合训练出词向量。Bengio 等^[33]借助语言模型的思想，利用回归神经网络对目标词与上下文之间关系建模，使语义上相近的词在空间中的分布也变得接近。Mikolov 等^[34]公布了 Word2vec 工具，提供了 Skip-gram 和 CBOW 两种词向量训练模型，构建某个词与其上下文词的条件概率模型。Pennington 等^[35]提出了 GloVe 模型，该模型能够结合全局的矩阵分解和局部的上下文窗口，利用统计信息来进行词向量的训练。

然而，受限于分布假设^[36]，即相似的词语出现在相似的上下文中，标准词向量模型一般具有三个先天性不足。第一，它们几乎不能区分语义相似与概念相关的不同^[37]，例如，对于词对（残疾/disability，死亡/death，score=2.8）而言，概念相关但语义上并不

相似的两个词“残疾”和“死亡”可能被误认为相似度较高，因为两者可以出现在相似或者完全相同的语境中，例如，“一种疾病导致他的残疾/死亡”。第二，标准词向量模型无法捕获近义词与反义词的区别^[38]，例如，对于词对（积极/positive，消极/negative，score=4.1），“积极”和“消极”这对语义相反的两个词语利用 Mikolov 等^[34]提出的 Skip-gram 模型训练得到的词向量之间的余弦相似度却高达 0.76。第三，依赖于上下文的标准词向量模型难以区分一词多义的情况^[39]。例如，词对（包袱/baggage; jokes in the crosstalk, 段子/joke, score=2.6），如果“包袱”被指定为最直接的含义“包裹”，那么“包袱”和“段子”两者之间相似度应该较小，反之，如果“包袱”特指相声中的“笑话”，则这两个词之间的相似度应该较大，然而对于标准词向量模型而言，只得到字符“包袱”对应的数值向量，对于两种含义并无区分。因此，如何在词向量模型中合理地融入语义知识成为近年来学者研究的重点和难点。

1.2.4 基于组合多策略的方法

近来，基于多策略融合的方法主要集中在如何利用语义约束改进词向量方法，其中，语义约束主要包括兼容对约束（如，同义词），对比对约束（如，反义词）以及概念层次结构中的词汇关系约束（如，上位-下位关系，部分-整体关系，蕴含关系）等。

一些研究使用了基本数学运算或简单规则，直接对基于语义词典资源的模型与词向量模型的相似度计算的结果实行合并策略，这种方法虽然简单有效，但是无法保证合并结果达到全局组合最优。

一部分学者关注如何利用语义知识重新训练一个新的词向量模型。Yu 和 Dredze^[40]从 Paraphrase Database 和 Wordnet 中抽取同义词作为语义知识，通过有监督学习改进词向量模型。Ono 等^[38]利用语义词典中的同义、反义信息以及大规模未标注文本数据中的分布信息，训练可以捕获反义词对的词向量。Liu 等^[41]提出了一个通用框架，将语义知识表示为三类排序不等式作为优化 Skip-gram 模型的约束条件，并利用随机梯度下降算法找到最优解。Nguyen 等^[42]通过一个表示同义词和反义词之间的重叠差异的函数对特征加权，从而将词汇对比约束加入原始词向量模型。

另一部分学者则关注如何改进一个已经训练好的词向量实例，将原始词向量模型的改进过程视为有约束的最优化问题，避免重新训练的巨大代价。Rothe 和 Schütze^[43]提出了一个 AutoExtend 系统，利用词语（Word）、同义词（Synset）和词素（Lexeme）在语义词典中的关系作为约束条件，将三者学到同一空间中，而不需要额外的训练语料。Faruqui 等^[44]提出了一个 Retrofitting 模型，其核心思想是对一个已有词向量模型以基于

关系知识为约束条件进行最优化后处理。随后，在此工作基础上，Mrkšić 等^[45]又提出了轻量级的 Counter-fitting 模型。

不难看出，近年来，词汇语义相似度计算研究越来越关注基于多策略融合方法，尤其是如何利用语义知识改进已有词向量模型，更是该研究领域内的前沿、热点研究课题。

1.2.5 研究的重点和难点分析

本文综合其计算对象及使用技术大致将现有研究方法分为四类，下面将分别对这四类方法目前存在的问题进行对比和总结。

(1) 基于同义词林、WordNet、HowNet 等语义词典的方法虽然简单有效，但存在一些缺点：①人工构建的语义词典虽然含有丰富的语义信息，但是构建词典的过程一般都比较耗时费力，因此，很难保证人工词典可以实时更新，而互联网时代信息急速增长，无法避免人们对新实体、新词等的探索和理解，因此，这种方法最大的缺陷就是存在未登录词（即，词典中没有出现的词）计算失效的可能。②现有方法中大多依赖于“上下位”层次关系，而对其他语义关系的研究较少，而且研究的重点主要是名词，对于形容词、动词以及一些虚词的研究较少。③当两个词语具有不同的词性或者多个词性时，计算相似度往往失效。

(2) 基于语料库的传统统计方法中主要包含：基于离线语料库的方法以及基于 Web 在线语料库的方法。对于这两类基于语料库的方法目前存在的问题在于：①基于离线语料库的方法效果对于静态语料库依赖性较强，难以解决数据噪音问题和未登录词（即，语料库中没有出现的词）失效问题。②基于 Web 在线语料库的方法虽然可以解决未登录词的相似度计算问题，但检索返回文档中的噪声和冗余问题一直没有很理想的解决方案。③此外，只考虑统计信息而忽略了语义关系是导致语料库统计方法性能提升的主要瓶颈之一。

(3) 基于语料库的词向量方法近年来备受关注，以 Word2vec 为代表的工作被 NLP 领域的多项任务广泛应用，其最大的意义在于解决了如何将自然语言转换为计算机可以计算的数字向量问题。然而，大部分工作侧重于基于分布假设对语料库建立概率统计模型，往往忽略人工知识，基本词向量方法目前存在的问题主要包含：①难以区分语义相似与概念相关。②无法捕获近义词与反义词的不同。③无法解决一词多义的问题。不难看出，该类方法的难点在于如何打破分布式假设的限制，从而得到能够克服上述问题的改进模型。

(4) 早期的基于组合策略的方法，往往是对在上述三类方法中的同类别方法进行融合，例如，WordNet 和 HowNet 两种语义词典方法的融合，这类方法很难充分避免一

类方法存在的通用问题，也无法充分利用每类方法的优势。随着词向量方法的出现与发展，如何将不同类别方法进行多策略融合，尤其是如何在词向量方法中融入人工语义知识，成为最近两年研究者热衷探索的问题，也是今后几年内词向量方法改进的重点和难点之一。

1.3 本文的主要工作

1.3.1 本文的研究内容

本文研究中文词汇语义相似度计算，重点研究了如何改进基于词向量的词汇语义相似度计算方法，并根据是否融入语义知识分为两部分研究：

(1) 无语义约束的词向量模型

本文分别使用机器翻译和 LSTMs 网络改进标准 Skip-gram 模型：首先，分别利用标准 Skip-gram 模型根据不同的训练语料得到标准词向量，通过实验分析了语料规模和语料质量对词向量模型的影响。其次，尝试用机器翻译构建中、英文词向量的关系，并用公开大规模英文词向量选择性地替换中文向量，改进原有中文词向量计算性能。最后，将词汇相似度计算问题转化为词汇关系预测问题，并通过 LSTMs 网络学习词对共现的句子序列，从而建模词汇关系。

(2) 融入语义约束的词向量模型

本文提出了一种改进 Counter-fitting 模型将语义约束融入已有词向量模型：首先，利用网络爬虫技术扩展词汇上下文。分别抓取词汇出现以及词对共现的句群作为词对“上下文”，并抓取同义词对、反义词对扩充现有人工语义词典。其次，计算词汇语义相似度。分别利用语义词典、检索结果、预训练好的词向量计算语义相似度。最后，利用改进 Counter-fitting 方法优化预训练好的词向量，具体的做法是，通过语义约束、向量空间存留构造多项式目标函数，然后用梯度下降算法对目标函数求解，其中，语义约束不仅包括同义约束、反义约束，还包含了相似约束。

1.3.2 本文的创新之处

针对以上研究内容，本文的创新之处如下：

(1) 无语义约束的词向量模型研究中，分别利用机器翻译替换中文向量和 LSTMs 学习词对共现句子提升词向量模型的性能。实验结果证明，机器翻译和 LSTMs 相对于标准词向量模型效果略有提高。

(2) 在融入语义约束的词向量模型研究中，在 Mrkšić 等人^[45]提出的 Counter-fitting 的模型基础上，不仅考虑同义约束、反义约束以及向量空间存留，还引入了相似约束。这样做的优点是，不仅局限于利用有限的强语义关系提升词向量，还能将基于多种语义资源得到的弱语义关系融入到词向量的改进过程中。实验结果证明，融入语义约束可有效地改进标准词向量方法的性能。

1.4 本文的组织结构

根据本文的主要工作，组织各章节内容如下：

第 1 章：绪论部分。首先，介绍了词汇语义相似度计算研究的产生背景与研究意义。其次，分别介绍了基于语义词典资源的方法、基于语料库的传统统计方法、基于语料库的词向量方法和基于组合多策略的方法四类主流方法的研究现状。然后，分别从研究内容和创新之处两方面总结了本文的主要工作。最后，简单介绍了本文的组织结构。

第 2 章：简单介绍了本文研究的理论基础与模型。首先，简要阐述了词汇语义相似度的基本概念、本文进行的词汇语义相似度计算研究的具体任务。其次，介绍了词表示方法、Word2vec 模型及空间向量距离的评价方法。最后，介绍了语言模型与人工神经网络。

第 3 章：详细介绍了基于词向量的语义相似度计算研究。首先，简单介绍了本文如何利用标准 Skip-gram 模型根据不同的训练语料得到词向量。其次，具体介绍本文用机器翻译构建中、英文词向量的关系，并用公开大规模英文词向量选择性地替换中文向量，改进原有中文词向量计算性能。然后，将词汇相似度计算问题转化为词汇关系预测问题，并通过 LSTMs 网络学习词对共现的句子序列建模词汇关系。最后，对第 3 章方法进行了实验和分析。

第 4 章：详细介绍了融入语义约束的词向量模型。首先，介绍了基于语义约束的词向量模型的构建。然后，详细介绍了本文实现该模型的具体方法，主要包括预处理、相似度计算和最优化原始词向量三部分。最后，对本文提出的改进 Counter-fitting 模型进行了实验和分析。

最后，结论。总结本文工作、找出本文不足并提出了相应的预期改进方案。

2 理论基础

2.1 词汇语义相似度

2.1.1 基本概念

在语言学研究中，语义（Semantics）是指词汇、句子、篇章等自然语言的意义，能够代表其本身在现实世界中所对应事物的概念（Concept）。在心理学研究中，相似性（Similarity）被认为是人类对两个对象之间关系进行定性比较所产生的认知感受，是人类思想和语言中最基本的元素之一^[46]。不难看出，语义相似性（Semantic Similarity）是指人类对自然语言对象之间关系进行定性比较所产生的心理感知。

语义相关性（Semantic Relatedness）常常与语义相似性的概念混淆。语义相似是一种特殊的语义相关，但是，事实上“语义相似”不能够完全等同于“语义相关”。语义相关性涉及两个对象 X 和 Y 之间的任何能构建两者联系的关系，而语义相似性仅仅关注一种“is-a”关系，也就是说，只有当“X is a Y”或者“Y is a X”成立的可能性很大时，可以认为 X 和 Y 之间语义相似度很高，但是当两个对象语义相关性较强时，并不能够绝对说明它们之间的语义相似度一定较高。例如，我们可以认为“汽车”与“公交车”语义相似度高，但是却不能认为“汽车”和“道路”之间非常相似，因为它们显然对应于现实世界中的不同类别的语义概念。

语义相似度是人们提出的一种对语义相似性进行量化的数值指标，Harispe 等^[47]认为语义相似度可以通过两个对象之间在语义内容层面上的距离表示，并且可以通过比较支持其含义或描述其性质的信息获得数值表达，从而定量地估计语言单元、概念、实例之间的语义关系的强度。

词汇（Word）是能够表达完整语义的基本单元，是理解一切自然语言的根本。而如何量化词汇语义相似性是人工智能科学利用计算机模仿人类认知相似性过程的核心步骤。为此，研究者定义了词汇语义相似度（Word Semantic Similarity）作为对两个词汇对象所表达含义相近程度的度量。

2.1.2 任务描述

词汇是能够表达完整语义的最小基本单元，本文研究的词汇语义相似度是指词汇在其所表达真实含义层面的相似程度。

本文限定该任务的讨论对象是词汇语义相似度，该任务的具体要求是：给定任意一对词汇，要求设计一种计算方法给出它们语义相似程度的实数值评分（1~10 分），如果

两个词汇的意义或者语义内容之间的距离越小，则它们在语义上就越相似、评分就越高，相反则分数越低。

本文的目标是提出一种算法或者模型能够自动地标出相似度得分，该得分与人工标注结果越一致越好，而最终结果的评价指标一般是机器自动计算得分序列与人工标注结果序列之间的一致性评价指数——斯皮尔曼相关系数和皮尔逊相关系数。即，给定数据集 $Data = \{(w_1^1, w_1^2, s_1), (w_2^1, w_2^2, s_2), \dots, (w_n^1, w_n^2, s_n)\}$ ，通过某种词汇语义相似度计算算法 sim 得到 $s'_i = sim(w_i^1, w_i^2), i \in 1, 2, \dots, n$ ，令 $S = (s_1, s_2, \dots, s_n)$ 和 $S' = (s'_1, s'_2, \dots, s'_n)$ ，则 S 和 S' 之间的序列相关系数越大越好。

2.2 语义词典资源

语义词典资源是组织词汇之间语义关系的规范结构化知识库，一般需要通过人工或者半人工方式构建而成。基于语义词典资源的方法，重点考虑语义词典结构特点以及两个词之间的关系来计算语义距离，并据此定义与之相应的计算相似度的公式。在本节中，本文将介绍同义词林、HowNet 和 WordNet 词典三种语义词典资源，以及利用它们计算词汇语义相似度的基本原理。

(1) 同义词林

《同义词林》是一本由梅家驹等整理的囊括了丰富的同义词和相关词的语义词典，在此基础上，哈尔滨工业大学 IR 实验室进行了改进与完善，剔除了原有词典中的 1 万多个罕见词及非常用词后完成了《同义词林扩展版》，收录词语共 77,343 条。目前基于同义词林的研究中，大多数是依赖于《同义词林扩展版》这一语义词典完成，本文所指的亦是如此。

同义词林利用 5 层树状层次结构将收录词汇组织起来，从上至下主要包括 12 个大类、97 个中类及 1400 个小类，小类之下是词群（段落）和原子词群（行），语义刻画逐层细致，最终，在同一行的词语互为同义词或者相关性极强的词语。与之相对应地，同义词林设计了表 2.1 所示的 5 级 8 位的编码模式。

表 2.1 同义词林编码模式

Tab. 2.1 The Coding Modes of Tongyici Cilin

编码位	1	2	3 4	5	6 7	8
符号	B	n	2 3	A	0 3	=\#\@\n
性质	大类	中类	小类	词群	原子词群	关系位
级别	第 1 级	第 2 级	第 3 级	第 4 级	第 5 级	

例如，词对（紫禁城/the Forbidden City，故宫/the Imperial Palace，score=10）中的两个词汇可以分别被编码在“Bn23A03# 正殿 … 金銮殿 紫禁城”和“Bn23A02# 行宫 东宫 … 故宫”两项中，它们在树状层次结构中的关系如图 2.1 所示。

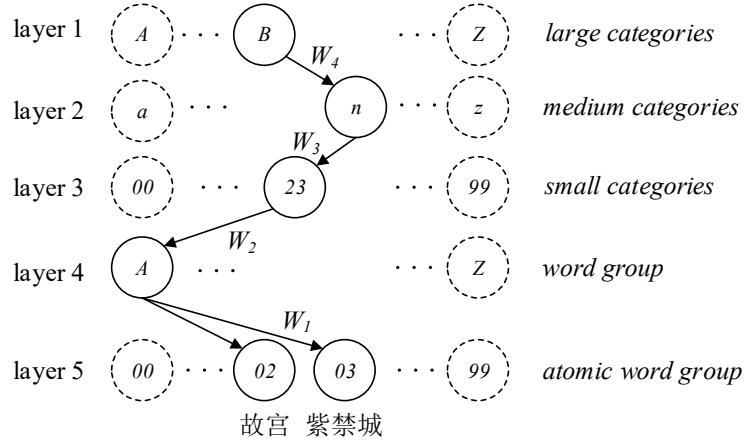


图 2.1 “紫禁城”和“故宫”在同义词林中关系

Fig. 2.1 The relationship between “the Forbidden City” and “the Imperial Palace” in Tongyici Cilin

基于同义词林计算语义相似度的主要想法是，根据层级结构和编码特点，综合考虑词语的相似性与相关性，设计以分支节点总数 n 和两个分支之间间隔距离 k 等结构特点为主要考虑因素的计算公式。

(2) 知网

知网（HowNet）是最早由董振东先生构建的一个常识知识库，它将中英文词汇所表示的概念作为描述对象，用于揭示概念之间的关系及其属性之间的关系。“概念（Concept）”和“义原（Sememe）”是 HowNet 中的两个重要定义：“概念”是对词汇语义的一种描述，也可以理解为对词汇含义的一种解释，一个词可以表达多个概念。“义原”是可以表达“概念”的独立、最小的语义单元，不可分割且无歧义。

HowNet 共包含 1500 个义原，大致上可以分为 3 组、10 个大类（如表 2.2 所示），并由同义关系、反义关系、上-下位关系等 8 种语义关系连接构成语义网。理论上，所有词汇均可用义原来表达，语义相似度较高的词对，它们对应概念的义原也会相同或者相近，这是利用《知网》进行语义相似度计算的前提假设。

基于 HowNet 的语义相似度计算方法的基本思路是：一个复杂的整体可以被分解成简单的部分，而通过计算部分之间的相似度得到整体的相似度，其中，每个部分的相似度则又需要根据虚词、实词和义原关系类型的不同制定不同的评价公式。

表 2.2 HowNet 中的 3 组（10 大类）义原

Tab. 2.2 Three Groups (10 classes) of Sememe in HowNet

组名	义原大类	作用
基本义原	Event 事件, Entity 实体, Attribute 属性值, aValue 属性值, Quantity 数量, qValue 数量值, SecondaryFeature 次要特征	用来描述单个概念的语义特征
语法义原	Syntax 语法	用于描述词语的语法特征
关系义原	EventRole 动态角色, ventFeatures 动态属性	用于描述概念和概念之间的关系

(3) 中文 WordNet

WordNet 是一种基于认知语言学、面向语义的英语词典，是根据词汇意义及词汇之间关系构建的一个语义网。它既定义了词汇的概念，又揭示了词汇所指概念之间的关系，在语义理解方面应用较为广泛，而且越来越多的人意识到大规模语义库对自然语言理解的重要性，因此，出现了开放多语言 WordNet (Open Multilingual Wordnet)，并且它们都被映射到普林斯顿英语 WordNet，即不同语言的 WordNet 之间只有语言不同而基本概念、网络结构等完全一致。特别地，Wang 和 Bond^[15]还构建了一个汉语开放词网 (Chinese Open Wordnet, COW)，这是一个大型、结构丰富、面向语义的中文词语网络，它重点关注包括上-下位，同义-反义，整体-部分，蕴含等语义关系。

WordNet 中很重要的一个概念是同义词集 (Synset)，每个 Synset 是一组具有同义关系的词汇的集合，其中包含一组无歧义的词元 (Lemma)。一个 Synset 对应于一个“概念”，这里的“概念”可以是“抽象概念”，例如，实体 (Entity)、状态 (State)、事件 (Event) 等，也可以是“具体概念”，例如，“掀背车”等非常具体的词。这些 Synsets 是 WordNet 的基本构成单位，并由复杂的词汇语义关系 (上位词和下位词) 相互连接形成复杂的概念分层网络。如图 2.2 所示，以词对 (借口，理由，score=7.3) 找到了 3 组 Synsets 在 WordNet 中的层次网络片段。

WordNet 可以用于计算语义相似度的基本思路是：给定一个 Synset 时可以遍历整个网络找到与之语义相关的一个 Synset 或者多个 Synsets。每个 Synset 都有一个或者多个上位词路径连接到一个根上位词，连接到同一个根的两个 Synsets 可能有一些共同的上位词，如果两个 Synsets 共用一个比较具体的上位词，那么它们在语义上一定具有较为密切的联系^[48]。

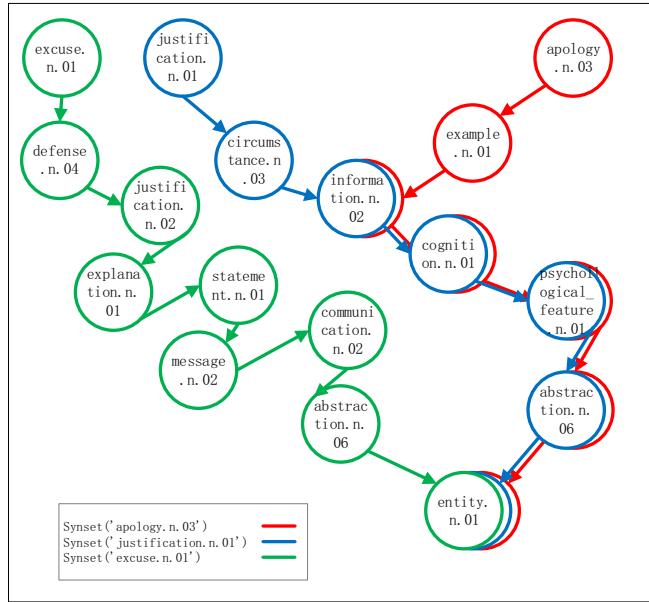


图 2.2 一个例子: Synsets 在 WordNet 中的关系

Fig. 2.2 An example: the relationship between Synsets in WordNet

2.3 词表示及空间向量距离

将“词汇语义相似”这个抽象关系通过特定计算方法映射成计算机可以处理的实数值是词汇语义相似度计算研究的关键问题，其中，涉及到两方面主要问题：如何合理地表示词汇以及如何衡量语义距离从而估计相似度。因此，本节将分别介绍词表示方法、典型基于分布式表示的 Word2vec 模型及空间向量距离的常用衡量方式。

2.3.1 词表示方法

(1) One-hot 表示

词汇作为文本的基本单位，将其从符号表示映射到数字表示，是利用机器学习方法解决自然语言处理问题的首要关键步骤。传统方法中，经常采用独热(One-hot)的编码方式将一个词语表示为一个长度为词表长的稀疏向量，并在单词对应编号的维度上置为1，其余维度都是0。例如，假设词汇“积极”和“消极”在词表中的索引分别为1和2，那么他们对应的向量表示分别为。

“积极”的独热表示 $[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, \dots]$

“消极”的独热表示 $[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, \dots]$

这种表示方式虽然在很长一段时间广泛应用于自然语言处理任务，但是在将字符转为数字的过程中丢弃了大量的语义信息，词汇之间彼此独立，即使是语义上是同义词或

者反义词也无法从它们对应的 One-hot 向量中发现任何关系。此外，One-hot 向量的维数随着词表的增大而变大，当向量维数过大时，运算的时间和空间复杂度会明显升高。因此，人们开始研究是否有其他更加合理的将文本符号数字化的方法，分布式表示方法应运而生。

(2) 分布式表示

Harris 最早于 1954 年提出了分布式假说，认为上下文相似度的词其语义也相似^[49]，随后，Firth 进一步阐明，词汇的语义由其所处的上下文决定^[50]。这为分布式表示的出现奠定了坚实的理论基础，随后便出现了词向量空间模型等分布式表示模型。

目前，分布式表示的主流方法可根据建模方式的不同分为：基于矩阵的分布式表示，基于聚类的分布式表示和基于神经网络的分布式表示^[51]。基于矩阵的分布式表示的方法核心任务是构建一个“词-上下文”矩阵，定义矩阵中的元素，然后通过矩阵分解等技术得到词表示，具有代表性的方法包括 LSA 模型，GloVe 模型等。基于聚类的分布式表示方法的主要技术是通过聚类构建“词-上下文”之间的联系，具有代表性的工作包括布朗聚类等。基于神经网络获得词表示的核心操作是通过神经网络对“词-上下文词”的关系进行建模，从语料库中训练得到每个词对应的语义向量，由于该类方法灵活易用，可建模复杂上下文关系，因此，近年来受到学术界的广泛关注，而当人们再次提起分布式表示时，大多数都是特指词向量（或者词嵌入）这一类特别的表示方法。

分布式词向量引入最大的意义在于，能够把高维度、稀疏的向量映射成低维度、稠密的向量表示，从而避免了 One-hot 表示方法忽略大量语义信息的问题。为符号文本提供了一种合理的数字表示方式，即将每一个单词映射成一个固定长度的实值向量表示，为利用向量空间距离衡量单词之间的相似性提供了有力的保证，也为机器学习算法在 NLP 任务中的更好应用提供了坚实的基础。

2.3.2 Word2vec 模型

Word2vec 作为 Google 公布的一款词向量训练工具，能够迅速而有效地把文本字符转化成分布式表示的实值向量。该工具简单高效，近年来备受关注，许多学者对此模型的扩展、改进和应用进行了大量研究。而 Word2vec 中用到的模型主要包含两种，即 CBOW (Continuous Bag-of-Words) 模型和 Skip-gram 模型，它们的模型架构如图 2.3 所示。接下来，本文将结合两种模型的架构图分别对它们的原理进行简单介绍。

(1) CBOW 模型

CBOW 模型的核心问题是在给定当前词 w_t 的上下文信息 $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ 的前提下，如何预测当前词，即构造语言模型求解条件概率 $p(w | context(w))$ 的问题。

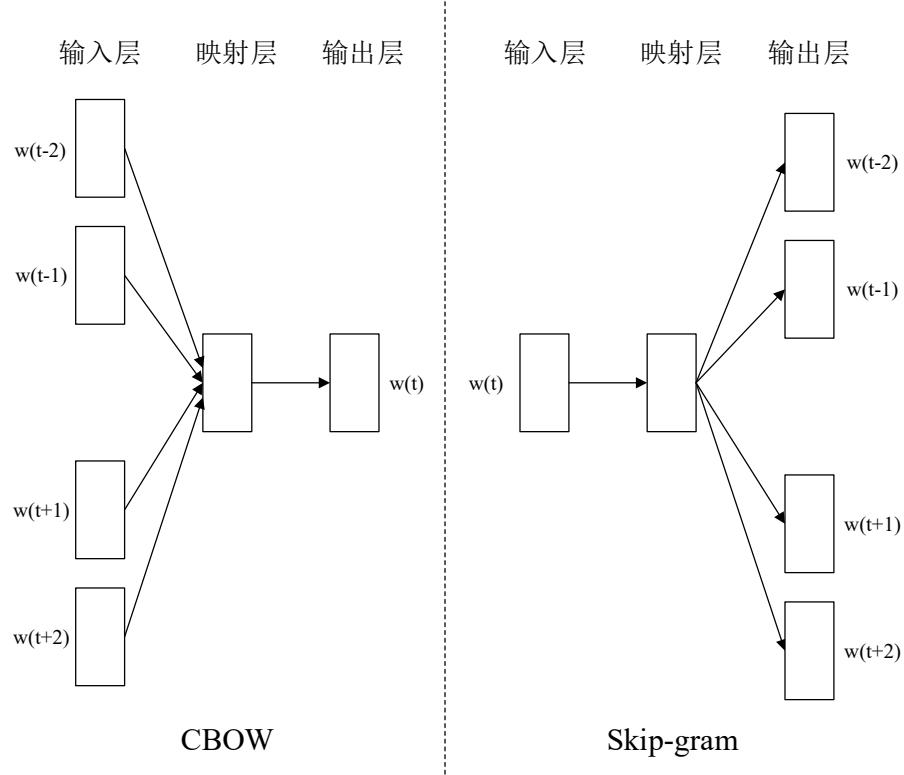


图 2.3 CBOW 模型和 Skip-gram 模型架构图
Fig. 2.3 CBOW and Skip-gram model architectures

CBOW 模型架构主要含有输入层、映射层、输出层，不同的只是去掉了 NNLM 模型中的隐藏层，并将最后的 Softmax 层替换成 Huffman 树的结构，从而简化了 NNLM 中计算复杂的环节。

其具体过程是：输入层接收词语 w 对应的上下文 $context(w)$ ，然后取 c 窗口内的词语构造 $2c$ 个词语对应的词向量 $v(context(w)_1), v(context(w)_2), \dots, v(context(w)_n)$ ，并指定词向量的长度为 m 。映射层将所有词向量按照公式进行累加得到 x_w ，并将其作为投影层的输出，送入输出层。

$$x_w = \sum_{i=1}^{2c} v(context(w)_i) \quad (2.1)$$

最后在输出层，通过统计词表 V 及语料 C 中的词频信息构造 Huffman 树结构，其中每个叶子节点对应 V 中的词语，从根节点到叶子节点的过程中，每次分支都相当于利用 Logistic 回归处理一个二分类问题，为 0 和 1 标签预测出概率值。当进行第 i 层分类时，节点 p_i^w 对应的参数是 θ_i^w ，那么，当前节点分到正类的概率是：

$$\sigma(x_w^T \theta_i^w) = \frac{1}{1 + e^{-x_w^T \theta_i^w}} \quad (2.2)$$

对于 V 中的任意一个词 w 都会有通过 k 个非叶子节点指向根节点 r 的唯一路径 l ，那么就需要 k 次二分类，而此时的目标为求解公式 2.3 的极大似然估计来保证 k 次分类效果最好。

$$p(w | context(w)) = \prod_{j=1}^k p(d_j^w | x_w, \theta_{j-1}^w) \quad (2.3)$$

其中， θ_{j-1}^w 是 $j-1$ 的分类过程优化参数， x_w 表示输入向量， d_j^w 是分类结果。

(2) Skip-gram 模型

Skip-gram 与 CBOW 模型恰好相反，它的核心思想为：给定当前词 w_t 已知的情况下，估计 w_t 的上下文信息 $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ ，即构造一种语言模型并求解条件概率 $p(context(w) | w)$ 。

其具体过程是：输入层先将输入的词语 w 转化为词向量形式 $v(w) \in \mathbb{R}^m$ ，架构图 2.3 中的投影层只是为了和 CBOW 模型做对比，而实际上并不存在。输出层构造 Huffman 树过程与 CBOW 模型一致，而唯一不同的是，对于 Skip-gram 模型，需要优化一个新的目标函数，如公式 2.4 所示：

$$p(context(w) | w) = \prod_{j=1}^k p(d_j^u | v(w), \theta_{j-1}^w) \quad (2.4)$$

其中， θ_{j-1}^w 是 $j-1$ 的分类过程优化参数， $v(w)$ 表示输入向量， d_j^u 是分类结果， u 代表当前词 w 上下文中的某一个词。

综上所述，Word2vec 利用三层人工神经网络构造“词-上下文”关系模型，简化了 Bengio 神经语言模型，算法快速有效，因而，这是目前最流行、应用最为广泛的词汇向量化工具之一。

2.3.3 空间向量相似度

在对词汇进行向量化表示之后，如何衡量词汇语义相似度的问题将被转化为如何计算语义空间向量相似度的问题。不同的空间距离定义方法对于相似度算法的结果有时影响较大，因此，有必要根据实际情况选择一种合适的语义距离度量方法。本文将现有空间向量相似度评价方法大致划分为四大类：一、直接定义相似度，例如，余弦相似度、Jaccard 相似系数等。二、利用空间向量距离定义相似度，例如，马氏距离，闵可夫斯基距离，汉明距离等。三、利用相关系数定义相似度，例如，皮尔逊相关系数，斯皮尔曼相关系数等。四、利用信息熵定义相似度，例如，互信息等。如图 2.4 所示，本文对上述提到的几种计算空间向量距离的方法进行了总结与归纳。

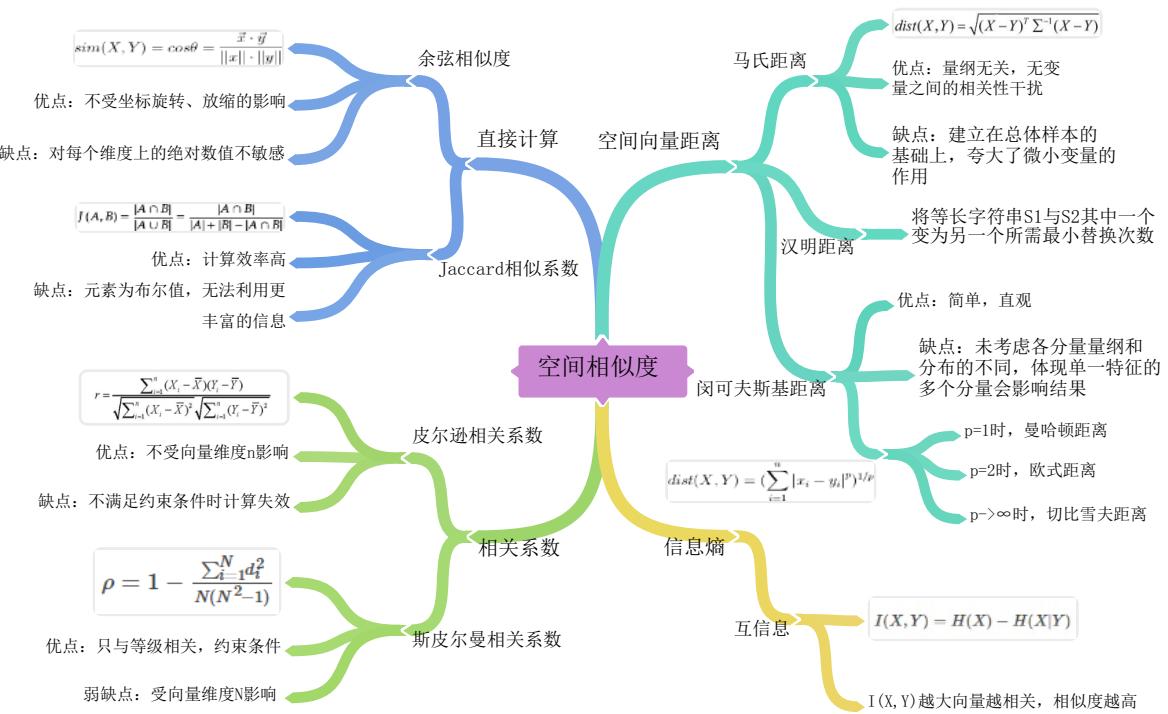


图 2.4 空间向量相似度计算方法总结
Fig. 2.4 An overview of Similarity of Space Vectors

词向量是一种含有上下文信息的 n 维连续型实值向量，其中的每个维度都可以被认为代表着某种特定的抽象特征， n 维抽象特征共同表达了对应词汇的上下文语义信息。但是，这种抽象特征与人工特征不同的是，我们无法知道每种特征具体是什么，也无法确定每种抽象特征的量纲，所以，词向量每个维度上的绝对数值并没有特别的含义，而

所有词向量在语义向量空间中的相对位置分布才能够确定其真实语义。为此，从上述提到的评价方法中，本文选择使用余弦相似度作为语义空间向量相似度衡量方法，这也是被学术界广泛认可的评价词向量相似度的主流方法之一。

2.4 语言模型与人工神经网络

2.4.1 统计语言模型

统计语言模型在 NLP 和 IR 领域一直占据着重要位置，在语音识别、机器翻译等众多子领域应用广泛，而最具有代表性的就是 N-gram 模型。如果将任意一个句子 S 看作是 w_1, w_2, \dots, w_i 这样一串具有特定顺序的词语序列，那么，N-gram 语言模型将对该序列作为一个句子出现的概率 $P(S)$ 建模，从而求出句子 S 在文本中出现的可能性。根据条件概率的公式，我们将求解句子 S 出现的概率问题转化为求解其构成词出现的条件概率的乘积问题，即为：

$$P(S) = P(w_1)P(w_2 | w_1)P(w_3 | w_1 w_2) \dots P(w_i | w_1 w_2 \dots w_{i-1}) \quad (2.5)$$

其中， $P(w_1)$ 是句首词 w_1 出现的概率， $P(w_i | w_{i-1} \dots w_1)$ 是在已知前 $i-1$ 个词的前提下，第 i 个词出现的概率。假设 V 表示词语集合的大小，那么前 $i-1$ 个词会有 V^{i-1} 种组合，模型含有 V^i 个自由参数，不难看出，随着 i 值增大，自由参数的数量成指数增长，能否从语料中正确地估计出这些参数将决定着语言模型的可用性。因此，引用等价类的方法减少参数，即：如果两个历史最近的 $N-1$ 个词相同，那么把这两个历史映射到同一个等价类当中，这种方法就称为 N 元语法（N-gram）。虽然在理论上，等价类的个数 n 越大则对下一个词出现的约束信息越多，具有越大的辨别力，但在实际应用中， N 的取值过大会导致在训练语料库中出现的次数过少，统计信息可靠性降低。据经验表明，在实际应用中 $N=1, 2, 3$ 的情况最多，即分别为我们常说的 Uni-gram，Bi-gram 和 Tri-gram 模型。

从本质上看，N-gram 语言模型是将自然语言视作一种正则文法语言，对有限自动状态进行建模的。这种模型简单直接，能够捕捉自然语言中的局部性约束性质，但是其表达能力非常有限，且无法较好地处理真实自然语言中的长程依赖问题。此外，由于数据稀疏问题必须要配合相应的数据平滑算法使用，平滑算法的选择也是其中的重点和难点问题之一。

2.4.2 神经语言模型

传统 N-gram 模型有两个缺点：一方面，由于训练语料限制，无法设置较大的 N 值，另一方面，没有考虑到相似的语法结构。为了解决这两个问题，提高语言模型的表达性能，Bengio 等^[33]于 2003 年提出一种新的语言模型——神经网络语言模型（Neural Network Language Model, NNLM）。

首先，他假设语料库中词语只与前 k 个词相关，然后，为了计算条件概率 $p(w_n | w_{n-k+1}^{n-1})$ ，他提出了的一种如图 2.5 所示的神经网络模型。这里为了方便后续说明，我们将词汇 w_n 的上下文表示为 $context(w_n) = w_{n-k+1}^{n-1}$ ，那么便可以将给定词语 w 在当前语境中出现的概率为 $p(w | context(w))$ 。与传统 N-gram 不同的是，NNLM 模型并不直接统计词频，而是将 $p(w | context(w))$ 这个条件概率表示为一个关于 θ 的函数 $F(w, context(w), \theta)$ ，这样问题就由求解条件概率问题转化为参数最优化问题，即，求解 F 函数的最优参数 θ^* 来生成语言模型。当求解得到 θ^* ，对于词表 V 中的任意词语 w ，都可以通过 $F(w, context(w), \theta^*)$ 得到对应的条件概率。NNLM 正是用来构造这样的 F 函数的，而最优参数 θ^* 求解即可通过某一训练语料 C 训练 NNLM 得到。具体的求解算法采用的是极大似然估计来训练模型的参数，即：

$$\max_{\theta} \sum_{w \in C} \log(F(w, context(w), \theta)) + R(\theta) \quad (2.6)$$

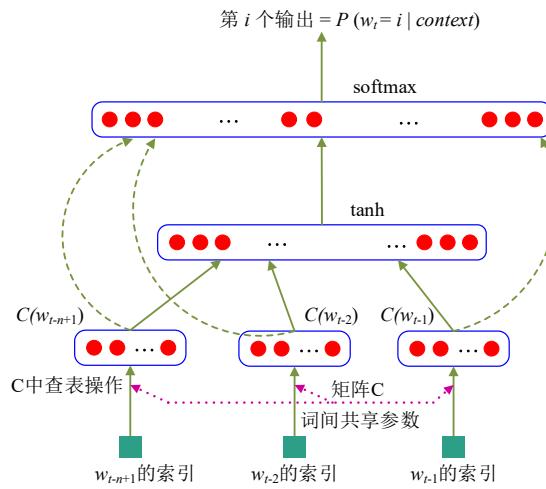


图 2.5 神经语言模型示意图

Fig. 2.5 Neural Network Language Model

2.4.3 循环神经网络

(1) RNNs 模型概述

在传统人工神经网络模型中，一个模型一般是由输入层-隐藏层-输出层组成，相邻层间任意两个神经细胞之间存在连接，而对于同一层中的神经元之间是无连接的，但是这种结构并不适合于序列标记问题，例如，在句子模型中，每个单词并不是独立存在的，预测某一个单词一般需要依赖于上下文的单词，即词序列中某一神经元的输出与同层的前后神经元存在关系，网络需要对前面信息“记忆”并将其应用到当前预测中。循环神经网络（Recurrent Neural Networks, RNNs）是可以解决这个问题的深度神经网络，除了层间连接之外，隐层中神经元之间存在连接，上一位置神经元的输入会影响对当前位置神经元的预测。基本的 RNNs 网络结构如图 2.6 所示，下面本文将以 Elman-type RNNs 为例，简述 RNNs 学习算法和求解过程。

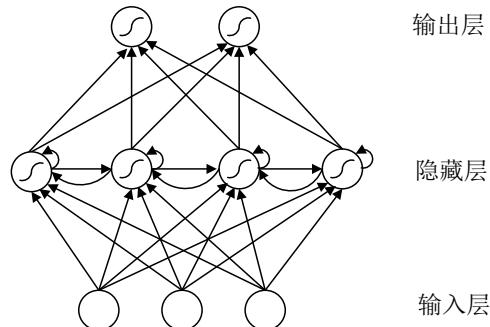


图 2.6 RNNs 模型示意图

Fig. 2.6 Recurrent Neural Networks Model

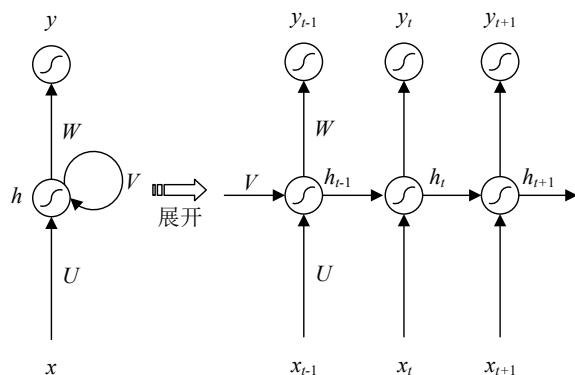


图 2.7 Elman RNNs 展开示意图

Fig. 2.7 An unrolled Elman-type RNNs

(2) RNNs 的学习算法和求解过程

RNNs 神经网络与 BP 神经网络在结构相似，区别在于 RNNs 网络的隐藏层神经元之间有权值连接，在学习算法和求解过程也类似于 BP 神经网络分为前向传播和反向传播两个阶段。

① 前向传播

不同于 2 层前馈网络，隐藏层中的神经元 S_t 由非递归的输入层映射变换和时序递归的上一个神经元 S_{t-1} 的历史输出共同决定。尽管 RNNs 与多层感知机看似差别不大，但是对于序列学习而言，这种改变影响深远，多层感知机只能实现输入到输出的映射，而 RNNs 原则上可以将前面输入的所有历史映射到输出中，使得整个网络具有“记忆”功能^[52]。

图 2.7 为简单、前向 Elman-type RNNs 的展开示意图，该网络可划分为输入层、隐藏层、输出层三层结构组成。令 t 代表时间， x_t 表示第 t 个时刻输入层接收的经 contextwin 操作取得上下文的词向量， $y_t = P(x_t | x_t, y_{t-1})$ 表示当前词训练后得到的输出，其维度与标注类别总数相等， U 是输入到隐藏层的权值矩阵， V 是隐藏层的权值矩阵， W 是隐藏层到输出层的权值矩阵。

隐藏层第 t 个时刻神经元接收来自输入层的输入值 x_t 以及来自隐藏层上一时刻输出值 h_{t-1} ，并将两者加权相加后经过非线性函数 f 得到当前的输出值 h_t ，即为：

$$h_t = f(Ux_t + Vh_{t-1}) \quad (2.7)$$

输出层第 t 个时刻接收同一时刻隐藏层输出 h_t ，并加权后经过非线性函数 g 得到输出预测值 y_t ，即为：

$$y_t = g(Wh_t) \quad (2.8)$$

对上述过程而言，涉及到了两个非线性函数 f 及 g ，它们分别为 Sigmoid 激活函数和 Softmax 函数，具体可表示为：

$$f(z) = \frac{1}{1 + e^{-z}} \quad (2.9)$$

$$g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}} \quad (2.10)$$

② 反向传播

误差反向传播算法是利用最优化方法解决人工神经网络训练的主流方法之一。在 RNNs 网络的反向传播算法中，比较著名的有 RTRL (Real Time Recurrent Learning) 算

法和 BPTT (Back Propagation Through Time) 算法, 由于 BPTT 算法在计算时间上较有优势, 这里主要介绍 BPTT 算法。BPTT 算法与 BP 算法的反向传播机制类似, 重复应用链式准则反向更新网络中的权值, 将最后时刻的残差传递回来, 与 BP 反向传播机制略有不同的是, 损失函数不仅要考虑输出层的影响, 还要考虑隐藏层中下一个神经元的影响, 也可以简单地理解为在原有 BP 算法基础上加入了对时间维度的考虑。为了应用链式准则实施梯度下降算法, 常常将 RNNs 中的环型结构展开 (见图 2.8)。

首先, 定义误差的信息熵函数作为损失函数, 即为,

$$E(y, \hat{y}) = \sum_t E_t(y_t, \hat{y}_t) = -\sum_t y_t \log(\hat{y}_t) \quad (2.11)$$

其中, y_t 是第 t 个时刻正确的输出, \hat{y}_t 是第 t 个时刻模型预测的输出, 如果将整个序列 (可能是一个句子或者一篇文章) 作为一个实例, 那么整体误差会在每个单词的位置上不断累加。图 2.8 为 Elman RNNs 误差反向传播机制示意图, 误差沿着虚箭头方向传播。

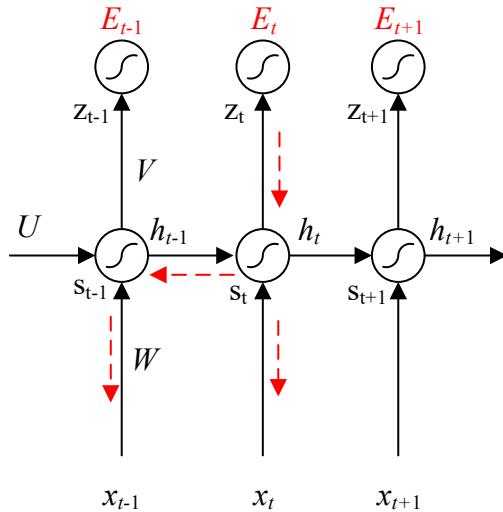


图 2.8 Elman RNNs 误差反向传播
Fig. 2.8 Error Back Propagation in Elman-type RNNs

我们的目标是分别求得损失函数对权值矩阵的偏导数 $\frac{\partial E}{\partial V}$, $\frac{\partial E}{\partial U}$ 和 $\frac{\partial E}{\partial W}$, 从而利用梯度下降算法学习出 RNN 模型中的最优权值矩阵 V , U 和 W 。

其次, 对于输出层而言, $\frac{\partial E}{\partial V}$ 是对该层每个时刻损失函数的梯度累加得到的, 即,

$$\frac{\partial E}{\partial V} = \sum_t \frac{\partial E_t}{\partial V} = \sum_t \frac{\partial E_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial V} = \sum_t \frac{\partial E_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial z_t} \frac{\partial z_t}{\partial V} = \sum_t (\hat{y}_t - y_t) \otimes h_t \quad (2.12)$$

其中， $\hat{y}_t = \text{softmax}(z_t)$ 是模型预测的结果输出， $z_t = Vh_t$ 表示隐藏层的输出经线性变换得到的输出层的输入， \otimes 表示计算两个向量外积运算， $\frac{\partial E_t}{\partial V}$ 为当前层、第 t 个时刻的误差，且不难看出，第 t 个时刻的误差只依赖于该时刻的预测值、实际值之间误差向量与当前时刻隐藏层神经元输出向量的张量积。

令第 t 个时刻隐藏层的输入 $s_t = Uh_{t-1} + Wx_t$ ， $h_t = \tanh(s_t)$ 是隐藏层的输出， $\delta_t = \frac{\partial E_t}{\partial s_t}$ ，

那么，隐藏层和输入层累积误差分别为，

$$\frac{\partial E}{\partial U} = \sum_t \sum_{k=0}^t \delta_k \otimes h_{k-1} \quad (2.13)$$

$$\frac{\partial E}{\partial W} = \sum_t \sum_{k=0}^t \delta_k \otimes x_k \quad (2.14)$$

最后，利用梯度下降更新各个参数：

$$V \leftarrow V - \lambda \sum_t (\hat{y}_t - y_t) \otimes h_t \quad (2.15)$$

$$U \leftarrow U - \lambda \sum_t \sum_{k=0}^t \delta_k \otimes h_{k-1} \quad (2.16)$$

$$W \leftarrow W - \lambda \sum_t \sum_{k=0}^t \delta_k \otimes x_k \quad (2.17)$$

(3) RNNs 扩展和改进模型

这些年，众多相关学者使用更多复杂的 RNNs 结构来提升现有 vanilla RNN 模型的性能。目前常见的一些 RNNs 模型包括 Simple RNNs, Bidirectional RNNs, Deep Bidirectional RNNs, Gated Recurrent Unit RNNs, LSTM Networks 等。其中，LSTMs 目前非常流行的改进版本之一，它相比于标准 RNNs 引入了“门”机制，从而使得信息在神经元链上选择性地通过。大量研究已表明，该网络结构在对长序列依赖问题中非常有效，且性能优于原始 RNNs 模型。

3 基于词向量的词汇语义相似度计算

3.1 标准 Skip-gram 词向量模型

Mikolov 等于 2013 年首次提出了 Skip-gram 模型从大规模语料库中学习连续词向量。该模型的核心思想是：建立一个输入层-映射层-输出层的三层神经语言模型，利用每个当前词语 w_t 作为投影层线性 Logistic 分类器的输入，来预测 w_t 前后一定范围内的词语 $w_{t-T}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+T}$ 。

令 $W^{(1)}$ 表示词向量矩阵，词表中的每个词语都可以通过查找该矩阵来映射成连续的数值向量， $W^{(2)}$ 表示窗口内周围词构成的矩阵。给定训练数据 w_1, w_2, \dots, w_n 序列，那么这个模型实际上目标是最大化函数：

$$Q = \frac{1}{N} \sum_{n=1}^N \sum_{-c \leq j \leq c, j \neq 0} \log p(w_{n+j} | w_n) \quad (3.1)$$

其中， N 代表词数， c 是上下文窗口大小， w_n 代表中心词， w_{n+j} 是中心词的相邻词，条件概率 $p(w_{n+j} | w_n)$ 可以表示为：

$$p(w_{n+j} | w_n) = \frac{\exp(w_{n+j}^{(2)} \cdot w_n^{(1)})}{\sum_{k=1}^V \exp(w_k^{(2)} \cdot w_n^{(1)})} \quad (3.2)$$

其中， $w_n^{(1)}$ 和 $w_k^{(2)}$ 表示矩阵 $W^{(1)}$ 和 $W^{(2)}$ 中的行向量。

因此，只要找到大规模的上下文语料，就可以根据上述模型训练出一个分布式表示的词向量空间，然后，利用语义空间向量的距离来计算得到最终的词汇语义相似度评分。

具体的实施方法是：

(1) 获取大量上下文原始语料。一方面，本文通过直接下载的方式得到部分语料，包括 199M 的数据堂新闻语料，1.1G 的维基百科语料和 261M 的微博语料。另一方面，为了避免评价词中出现在上述语料中未登录的情况，本文还利用词对构造 Query，检索“百度新闻”和“写搜”造句网爬取了部分互联网语料。

(2) 进行原始语料预处理。为了清洗原始数据获得更利于语义抽取的训练语料，本文对原始语料进行了预处理工作，其中，主要过程包括：过滤 XML 标签、去除冗余句、过滤数字和特殊字符、过滤标点符号、去除停用词、分词、分句。

(3) Skip-gram 模型训练词向量。我们利用 Word2vec 工具对预处理后的语料进行训练，其中，训练参数设置 $sg=1$ 且 $hs=1$ 选择层级 softmax 实现的 Skip-gram 模型，词向量维度 $size=300$ ，训练窗口大小 $window=10$ ，采样阈值 $sample=1e-4$ ， $min_count=5$ 。

(4) 语义空间向量的相似度计算。依次查表找到数据集中的词汇对应的词向量，计算向量余弦相似度，并按照线性放缩将相似度调整至[1,10]范围内，便可得到最终的相似度评分。

3.2 基于机器翻译的改进方法

利用语义空间中的词向量计算词汇相似度时，最主要的工作在于如何合理地进行词表示，从而得到质量较高的词向量模型。在 3.1 节中，本文利用了标准的 Skip-gram 模型训练了中文词向量，然而，中文语料库训练词向量时会遇到分词错误的问题，从而可能影响语料库的训练效果，此外，基于大规模中文语料库训练好的公开词向量也非常少见。相比之下，英文具有天然单词分隔符，不存在分词错误这类问题，而且研究开始较早，有大规模公开词向量，例如，利用 Word2vec 预训练的公开词向量模型 GoogleNews-vectors-negative300.bin，这是在 Google News dataset 语料上训练得到的具有 300 万个单词和短语的大规模词向量模型。

因此，基于上述事实，我们提出这样一个问题：能否利用在海量数据上训练的英文公开词向量模型来提升中文词汇相似度计算的性能呢？对此，我们做出假设：当两个词汇从中文到英文翻译完全正确的时候，使用大规模语料训练好的英文词向量会比中文词向量语义损失更小、语义相似度计算会更加准确。

在此假设之下，我们需要解决的问题是如何评价机器翻译是否完全翻译正确。具体来说，中文词汇翻译到英文后，一方面，要求英文译文的拼写不存在错误，另一方面，要求翻译译文词汇的语义不改变。众所周知，现有机器翻译系统在句子级别及句子级别以上语言单元的翻译还无法达到可以代替人工翻译的水平，然而，词和简单短语级别的翻译性能一般在可接受范围内。因此，本文设置两条较为严格的约束来保证翻译质量：

- (1) 英文译文经拼写检查后无拼写错误。
- (2) 一个中文词汇只翻译成一个英文单词。

具体实现的步骤是：

- (1) 利用每个词构造成查询 Query，调用 Google Translation API，将每组中文词汇翻译成与之分别相对应的英文词汇。
- (2) 检查并标记两个译文长度均为 1 个单词的词对。

(3) 利用单词拼写检错程序检测步骤(2)中标记的词对，将存在拼写错误的词对剔除。本文使用了 PyEnchant 工具包提供的算法进行单词拼写检测。

(4) 将剩下的词对取出英文词向量模型中的向量进行相似度计算，其他词对仍然使用中文词向量计算。

上述的方法，虽然比较简单并没有改变原始中文词向量模型的向量空间，只是利用英文词向量计算得到的数值替换掉了一些利用中文词向量计算得到的数值，但经过实验验证，可以有效提升性能，更多跨语言的工作还需进一步完善。

3.3 基于 LSTMs 的改进方法

3.1 节中，本文讨论了如何利用词汇出现的大量真实语料训练 Skip-gram 模型，得到一个可以表示语义的分布式词向量空间。在标准 Skip-gram 模型训练得到词向量基础上，本节将进一步讨论如何利用 LSTMs 模型构建一个神经语言模型，对相似度评分进行预测。

我们的基本假设是：除了词语的上下文环境可以反映其语义外，对于词汇相似度评价任务而言，词对中的两个词共现的上下文环境也能够加强它们之间的语义关系。因此，本文想要构造这样一种 LSTMs 模型学习句子序列的过程：训练过程中，输入是词对共现的句子序列和将相似度评分取整后的整数标签 $i = 1, \dots, 10$ ，并据此训练得到 LSTMs 模型；测试过程中，输入是词对共现的句子序列和已经训练好的 LSTMs 模型，输出是最终预测词对的语义相似度评分。

图 3.1 是 LSTMs 模型学习词对共现句子的示意图。假设要学习词对 (w_1, w_2) 所在的句子 S ，那么需要分别考虑如何构造输入向量与如何对模型中的每个权值进行更新。

首先，我们构造一个 $4n$ ($n=150$) 维度的含有距离信息的词向量：其中前 $2n$ 维度为 Skip-gram 预训练好的词向量，接下来 n 个维度用当前词距 w_1 的距离填充，最后 n 维用当前词距 w_2 的距离填充，如果当前词在目标词右侧则距离数值为正值，如果当前词在目标词左侧则距离数值为负值，如果目标词为当前词本身则距离数值为 0。

下面本文将用公式进一步说明记忆神经元如何在每个时刻进行参数更新，令 x_t 是第 t 个时刻记忆神经元的输入， $W_i, W_f, W_c, W_o, U_i, U_f, U_c, U_o, V_o$ 分别是权重矩阵， b_i, b_f, b_c, b_o 分别是偏置向量。

首先，计算第 t 个时刻的输入门激活值 i_t 和记忆神经元状态的候选值 \tilde{C}_t ：

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (3.3)$$

$$\tilde{C}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3.4)$$

其次，计算第 t 个时刻的忘记门的激活值 f_t ：

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (3.5)$$

给定输入门激活值 i_t ，忘记门激活值 f_t 以及记忆神经元状态的候选值 \tilde{C}_t ，则可以计算出第 t 个时刻新的记忆神经元状态为：

$$C_t = i_t * \tilde{C}_t + f_t * C_{t-1} \quad (3.6)$$

然后，利用新记忆神经细胞状态，可以计算出它们的输出门激活值及最终输出，分别用公式表示为：

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3.7)$$

$$h_t = o_t * \tanh(C_t) \quad (3.8)$$

句子序列 x_0, x_1, \dots, x_n 输入 LSTMs 层之后会产生一个新的序列 h_0, h_1, \dots, h_n ，随后，该序列经过 Mean Pooling 层在所有时间步长上进行平均，再进入 Logistic 回归层输出每个类别标签是 i 的概率为 p_i ，最终，可以得到词对之间相似度评分为：

$$\text{auto_score} = \sum_{i=1}^{10} i * p_i \quad (3.9)$$

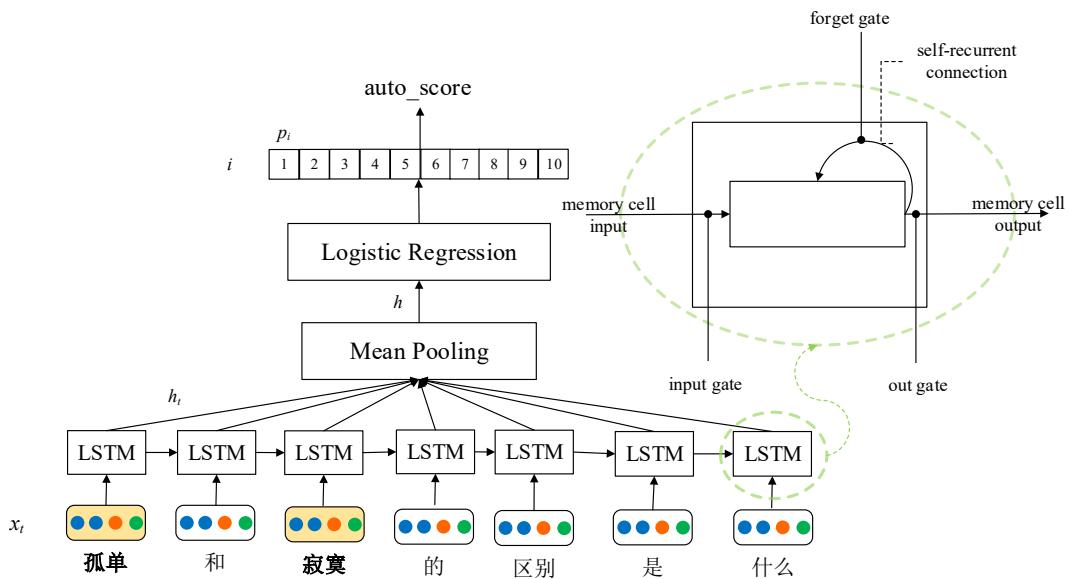


图 3.1 LSTM 学习词对共现句子

Fig. 3.1 Learning co-occurrence sentence using LSTMs

具体实现的步骤是：

(1) 爬取词对共现的文本片段。本文以词语 w_1 和 w_2 构造查询 Query 向百度 API 发送查询请求，并在检索结果页集合中爬取同时含有词语 w_1 和 w_2 网页快照片段。

(2) 预处理得到词共现的句子。为了清洗原始数据获得更利于语义抽取的训练语料，本文对原始网页快照片段进行了预处理工作，其中，主要过程包括：过滤 XML 标签、去除冗余句和长度小于 5 个字的句子、过滤数字和特殊字符、过滤标点符号、去除停用词、分词、分句。然后，将 w_1 和 w_2 共现的片段按照一句一行的形式保存到文件中，最后，得到不重复的句子共 2,954 句。

(3) 模型训练和测试。按照图 3.1 的结构搭建 LSTMs 网络框架，并将词对及步骤(2)中得到的句子构成的语料随机抽样分为 5 份，其中 1 份作为测试语料，另 4 份合并作为训练语料，构成 5 个实验组，即进行 5 倍交叉实验。

3.4 实验与分析

3.4.1 实验语料

本文实验语料为 NLPCC-ICCPOL 会议于 2016 年提供的中文词汇语义相似度公共评测语料 PKU-500^[53]。评测官方提供 40 条词对作为样例数据 (PKU-SAMPLE-40)，又在 10,000 条词对数据挑选 500 条词对进行了人工标注，得到共享任务的测试数据集 (PKU-500)。该语料区分语义相似与语义关联的不同，是目前较新的中文词汇语义相似度语料。实验语料的具体实例如表 3.1 所示。

表 3.1 实验语料实例
Tab. 3.1 Examples of dataset

词对 (w_1, w_2)	相似度
(没戏, 没辙)	4.9
(只管, 尽管)	4
(GDP, 生产力)	6.5

评测官方在筛选词汇时综合考虑了多方面因素，包括：

(1) 领域。该数据集主要是从新闻文章和微博文本中获取，从而能够保证这些词既可以应用在比较正式的书面语文档，也可以出现在一部分表达比较随意的互联网短文本中。

(2) 词频。该数据集首先在候选语料库上进行词频统计，然后从高频、中频、低频词中分别选择了约 30%，50%，20% 的词语。

(3) 词性。既包含名词、动词、形容词这类实词，也包含副词、连词等功能性词。

(4) 长度。选择词汇包括单字词、双字词、三字词以及少量的四字成语。

(5) 语义。该数据集挑选了一些语义模糊不清的多义词。

此外，词对由计算语言学的专家精心挑选，并由 20 名语言学学生人工标注，评分从 1 至 10 分，最终结果为 20 人标注结果的平均值。

3.4.2 评价方法

本文的实验效果将由斯皮尔曼 (Spearman, 以下简称 ρ) 和皮尔逊 (Pearson, 以下简称 r) 等级相关系数评价。这两种评价指标被广泛用于自动预测结果序列和人工标注结果序列的一致性检验。其中， ρ 值和 r 值可分别定义为：

$$\rho = 1 - \frac{6 \sum_{i=1}^n (R_{X_i} - R_{Y_i})^2}{n(n^2 - 1)} \quad (3.10)$$

$$r = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^n (X_i - \bar{X})^2} \sqrt{\sum_{i=1}^n (Y_i - \bar{Y})^2}} \quad (3.11)$$

其中， X_i 和 Y_i 为自动评分和人工评分的原始数据， \bar{X} 和 \bar{Y} 是两组数据序列中的平均数， R_{X_i} 和 R_{Y_i} 是标准差等级变量， n 是序列中样本的数量。

在本文实验中，我们将斯皮尔曼 ρ 视为主要评价指标，皮尔逊 r 作为次要评价指标。

3.4.3 结果与分析

本章重点研究如何训练和使用词向量来得到语义相似度，其中包括三部分：标准 Skip-gram 方法，基于机器翻译的改进方法以及 LSTMs 学习词对共现句子的改进方法。

首先，我们使用标准 Skip-gram 模型训练了 5 类不同的语料库，其中包括直接下载获得的 199M 的数据堂新闻语料 (Datatang)，1.1G 的维基百科语料 (Wiki)，261M 的微博语料 (Weibo)，也包括将词对作为 Query 检索得到的“百度新闻”语料 (News) 和“写搜”造句网爬取的语料 (Xieso)。将这 5 类语料组合成 11 组不同规模、不同来源的语料库，利用 Skip-gram 模型训练词向量，计算测试语料中词的向量余弦相似度，进行线性放缩至 [1,10]，并利用斯皮尔曼 ρ 和皮尔逊 r 评价自动计算得分与人工标注得分结果的一致性，结果如表 3.2 所示。

表 3.2 Skip-gram 模型在不同语料库上的结果

Tab. 3.2 Results of Skip-gram Models Based on Different Corpora

No.	语料库组合	ρ	r
1	Xieso (62M)	0.205	0.203
2	Wiki (1.1G)	0.211	0.213
3	Weibo (261M)	0.227	0.077
4	Datatang (199M)	0.267	0.272
5	News (381M)	0.311	0.305
6	News+Xieso	0.311	0.311
7	News+Xieso+Weibo	0.180	0.128
8	News+Xieso+Wiki	0.178	0.197
9	News+Xieso+Datatang	0.174	0.190
10	News+Xieso+DataTang+Wiki	0.214	0.239
11	News+Xieso+DataTang+Wiki+Weibo	0.214	0.134

分析表 3.2 不难看出，语料库的规模和质量对词向量模型影响显著。对比 No.1,5,6 组实验结果， $Xieso (0.205 \rho / 0.203 r) < News (0.311 \rho / 0.305 r) < News+Xieso (0.311 \rho / 0.311 r)$ ，我们可以发现越大规模的语料可能含有越丰富的上下文信息来提升词向量语义表达能力，即增加语料的规模可以提升词向量模型的性能。然而，这并不代表说规模越大性能一定提升，例如，随着语料资源规模的增加，No.7-11 组实验结果较 No.6 组实验 ($0.311 \rho / 0.311 r$) 全部明显降低，下降最明显的一组是 No.9，降低了 0.137ρ 。对此，我们推断是因为语料库的质量影响了词向量的性能，具体来说，Datatang 和 Wiki 语料中可能存在不符合语法规范的文本片段，或者存在其他噪声问题。最后，我们选择第 6 组实验作为这 11 组组合方案中的最佳方案，这也是 3.2 节和 3.3 节中改进方法的基线。

表 3.3 机器翻译的改进结果

Tab. 3.3 Result of Improvement by Translation

No.	方法	ρ	r
1	Baseline	0.311	0.311
2	Baseline + Translation	0.397	0.443

表 3.3 展示的是标准 Skip-gram 以及使用机器翻译技术改进已有中文词向量的结果。具体做法是将英文译文经拼写检查后无拼写错误且一个中文词汇只翻译成一个英文单词的词对，根据其英文译文查找公开英文词向量模型，将英文词向量语义相似度结果替换中文结果。实验结果表明，改进结果达到 $0.397 \rho / 0.443 r$ ，比基线方法提高 $0.086 \rho / 0.132 r (27.65\% / 42.44\%)$ ，即基于机器翻译的改进方法可以有效提升原有词向量模型的性能。

表 3.4 LSTMs 的改进结果
Tab. 3.7 Result of Improvement by LSTMs

No.	方法	ρ	r
1	Baseline	0.311	0.311
2	Baseline + LSTMs	0.351	0.364

如表 3.4 所示，是利用 LSTMs 改进词向量模型的结果。具体操作中，本文以 w_1 和 w_2 构造查询 Query “ $w_1 w_2$ ”，调用百度 API 爬取网页快照片段，并执行去噪、分句、分词等预处理工作，最后将 w_1 和 w_2 共现的片段按照一句一行的形式保存到文件中，使用 LSTMs 模型学习词对共现的句子，不难看出，改进方法达到 $0.351 \rho / 0.364 r$ ，比基线方法提高 $0.04 \rho / 0.053 r$ ，即 LSTMs 模型学习词对共现的句子对词向量模型有一定提升作用。

4 融入语义约束的词向量模型

4.1 模型构建

受 Mrkšić 等人^[45]工作的启发，本文引入一项剩余相似约束指数（Residual Similarity Index, RSI）改进基本 Counter-fitting 模型，将基于词典和检索的相似度作为语义知识融入词向量模型而得到改进 Counter-fitting 模型，即为本文提出的融入语义约束的词向量模型。

由于词向量的训练是基于上下文语境来进行的，所以该模型往往关注于语料的统计特征而很难学习到特定语言学上的约束关系，例如同义或者反义关系。而 Counter-fitting 模型的主要想法是根据同义和反义约束来对训练得到的原始词向量进行微调、优化，使其能够在词汇相似度计算任务中表现出更好的性能。

具体来说，该模型已知一个原始词向量空间 $V = \{v_1, v_2, \dots, v_n\}$ ，其中，每一个词向量对应词汇表中的一个词， N 则代表着词汇表所包含的所有词汇的数量，找出一组语义约束条件 C ，使得关于新词向量空间 $V' = \{v'_1, v'_2, \dots, v'_n\}$ 和语义约束条件 C 的目标函数经数次迭代达到最值，在此过程中，便可以得到一个融入了语义约束的新词向量空间 V' 。

在原始 Counter-fitting 模型中，分别从反义词词典和同义词词典中抽取反义约束集合 A 和同义约束集合 S 。这两种语义约束集合中的元素均为词汇索引构成的元组，例如， A 中的元素 (i, j) 代表的意义是词汇表中第 i 个词和第 j 个词互为反义词。然而，两种语义约束集合 A 、 S 的规模和质量都只取决于对应的词典资源，这限制了两种约束对词向量模型的进一步提升，因此，我们添加一项剩余相似约束 R ，来表示除了同义约束这种强约束之外、其它能够表达两个词汇相似度较高的弱约束，特别地，当语义词典方法以及基于检索结果方法计算所得的词汇相似度的最大值高于某个特定阈值时，我们便将其对应的语言约束抽取出来得到剩余相似度约束对 (i, j) ，其代表的意义则是词汇表中的第 i 个词和第 j 个词相似关系较显著。

不难理解，模型的目标函数需要满足：对于包含在同义约束集合 S 和相似约束集合 R 中的词对，词对中的词所对应的词向量之间的距离应当尽可能小，而对于包含在反义约束集合 A 中的词对，词对中的词所对应的词向量之间的距离应当尽可能大。此外，由于原始词向量空间中的拓扑结构反映了词汇之间的基本关系，为了最大程度上保留原始词向量的语义信息，最优化后的得到的新词向量集合应当尽可能地接近原始的词向量集合。

因此，我们定义 $d(v_i, v_j) = 1 - \cos(v_i, v_j)$ 作为词向量之间的真距离，用于对原始词向量集合进行 Counter-fitting 优化的目标函数将包括如下几项：

(1) 反义排斥 (Antonym Repel, AR)

$$AR(V') = \sum_{(u,w) \in A} \tau(\delta - d(v'_u, v'_w)) \quad (4.1)$$

这一项的作用是增大反义词的词向量之间的距离。其中， $\tau(x) \triangleq \max(0, x)$ ， $\delta \in [0, 1]$ 表示的是反义词对的词向量之间的理想化距离。

(2) 同义词吸引 (Synonym Attract, SA)

$$SA(V') = \sum_{(u,w) \in S} \tau(d(v'_u, v'_w) - \gamma) \quad (4.2)$$

和 AR 项类似，这一项的作用是减小同义词的词向量之间的距离， $\gamma \in [0, 1]$ 则表示同义词对的词向量之间的理想化距离。

(3) 剩余相似指数 (Residual Similarity Index, RSI)

$$RSI(V') = \sum_{(u,w) \in R} \tau(d(v'_u, v'_w) - \theta) \quad (4.3)$$

和 SA 项类似，这一项的作用是减小相似关系显著的词汇之间的向量距离， $\theta \in [0, 1]$ 表示相似度高的词向量之间的理想化距离的下界。R 中的元素按照如下规则进行抽取： $\max\{sim_1, sim_2, \dots, sim_N\} \geq \sigma$ ，其中， sim_i 表示使用第 i 种相似度计算方法得到的相似度值， $\sigma = f(1 - \theta)$ 则表示抽取的阈值，线性函数 f 是将理想化相似度值映射到 sim_i 评分标准的函数。

(4) 向量空间存留 (Vector Space Preservation, VSP)

$$VSP(V') = \sum_{i=1}^N \sum_{j \in N(i)} \tau(d(v_u, v_w) - d(v'_u, v'_w)) \quad (4.4)$$

这一项的作用是保证优化后的词向量集合和原始词向量集合之间的相似性。为了保证计算过程的速度，对于词向量集合中的每一个词，在实际操作中并不是计算它和所有其他词汇之间的距离，而是仅计算在词向量空间中、该词与它周围特定半径 ρ 覆盖的词汇之间的距离。

综合上述四项，可以定义 Counter-fitting 算法需最小化的目标函数为：

$$C(V, V') = k_1 AR(V') + k_2 SA(V') + k_3 RSI(V') + k_4 VSP(V, V') \quad (4.5)$$

其中，超参数 $k_1, k_2, k_3, k_4 \geq 0$ 代表的是 AR、SA、RSI 和 VSP 四个项的权重，且令 $k_4 = 1 - k_1 - k_2 - k_3$ 。我们使用了随机梯度下降算法（Stochastic Gradient Descent, SGD）来对该目标函数进行最优化计算，在迭代一定次数后达到收敛。

4.2 具体实现

图 4.1 是融入语义约束的词向量模型——改进 Counter-fitting 的具体实现流程图，主要包含了三个阶段：

(1) 预处理。该阶段主要利用网络爬虫，分别抓取词对出现以及词对共现的上下文片段，并且对特定网站抓取一些同义词对、反义词对扩充现有语义词典。

(2) 计算词汇语义相似度。分别使用同义词林、WordNet 和知网（HowNet）、Web 检索结果以及词向量计算相似度，供后续筛选相似约束集合 R 使用。

(3) 最优化词向量。首先，抽取语义约束对集合：从反义词典中抽取反义约束对 A，从同义词典中抽取同义约束对集合 S，根据第(2)阶段得到的多种相似度筛选相似约束 R，其次，利用语义约束 A、S、R 以及词向量模型构造改进 Counter-fitting 模型的目标函数，最后，使用随机梯度下降算法最优化原始词向量空间。

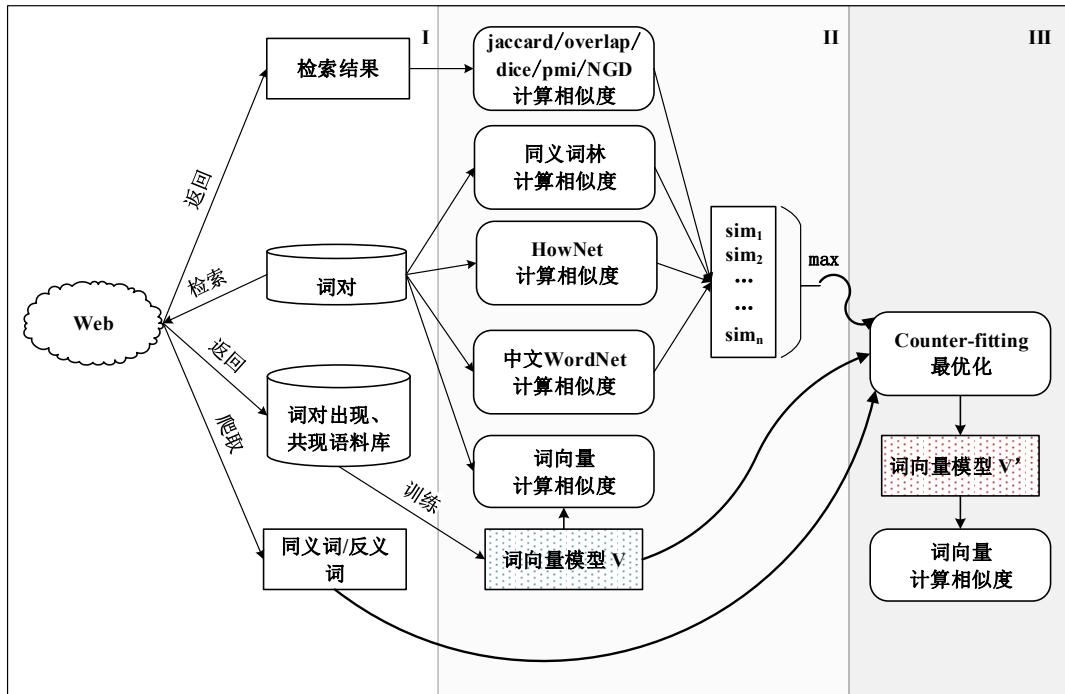


图 4.1 改进 Counter-fitting 模型具体实现流程图

Fig. 4.1 The Flowchart of Improved Counter-fitting Implementation

4.2.1 预处理

预处理的工作主要分为如下三部分：

(1) 获取检索结果。对于测试语料中的每一个词对 (w_1, w_2) ，分别将 “ w_1 ”，“ w_2 ” 及 “ $w_1 w_2$ ” 作为检索语句，向 “百度搜索” 发起请求，并分别记录检索返回词出现、共现的结果文档数量。供后续基于 Web 检索结果计算相似度公式使用。

(2) 爬取词对出现、共现语料库。为了避免测试词对在直接下载的语料库中是未登录词的情况，我们对测试语料中的每一个词对 (w_1, w_2) ，分别将 “ w_1 ”，“ w_2 ” 及 “ $w_1 w_2$ ” 作为检索语句，检索 “百度新闻” 网站和 “写搜” 造句网，得到词对出现和共现的语料库。然后，对原始语料进行过滤 XML 标签、去除冗余句、过滤数字和特殊字符、过滤标点符号、去除停用词、分词等预处理工作。后续供词向量训练使用。

(3) 扩充已有的同义词和反义词词典。本文使用的同义反义约束一部分是从同义词林中抽取得到，另一部分是利用网络爬虫从网站上获取，具体来说，从 “在线同义词查询” 和 “在线反义词查询” 网站爬取了 8,411 条同义词和 7,411 对反义词。最终将两部分合并、去除冗余。供后续改进 Counter-fitting 模型优化过程使用。

4.2.2 计算词汇相似度

在完成了预处理工作之后，本文将具体地介绍如何得到构造剩余相似指数这一项所需的局部词汇相似度计算结果，主要包括如下几部分：

(1) 利用同义词林计算相似度

基于同义词林的语义相似度计算方法中，较有代表性的是田久乐和赵蔚的工作^[11]，该算法利用同义词林的结构和编码特点，并综合考虑词汇的相似性和相关性，提出了以分支节点总数 n 和两个分支之间间隔距离 k 为主要考虑因素的词汇语义相似度计算方法。朱新华等^[23]认为仅考虑 n 和 k 则可能出现语义距离较近、却因为分支间隔较远而算得相似度较小的情况，因此，提出以词汇之间距离 d 为主要考虑因素、以 n 和 k 为调节参数的改进公式。经实验证，改进方法在本文语料上的性能好于原始方法，因而，本文将使用改进公式计算词汇在同义词林中的语义相似度。

具体计算的方法为：

给定两个词汇 w_a 和 w_b ，令 $C_a = \{c_a^1, c_a^2, \dots, c_a^A\}$ 和 $C_b = \{c_b^1, c_b^2, \dots, c_b^B\}$ 分别表示 w_a 和 w_b 的所有义项的集合， $W = \{W_1, W_2, W_3, W_4\}$ 表示同义词林中相邻两层之间的权重，并要求其满足 $0.5 \leq W_1 \leq W_2 \leq W_3 \leq W_4 \leq 5$ 且 $W_1 + W_2 + W_3 + W_4 \leq 10$ ，则 w_a 和 w_b 之间的语义相似度可以认为是两者所有义项组合之间相似度的最大值，即：

$$SIM_{cillin}(w_a, w_b) = \max_{1 \leq i \leq A; 1 \leq j \leq B} \{sim(c_a^i, c_b^j)\} \quad (4.6)$$

其中, $sim(c_a^i, c_b^j)$ 表示义项 c_a^i 和 c_b^j 之间的相似度, 如果义项对应的前 7 位编码相同且第 8 为 “=” 时, 则义项相似度为 1.0; 如果义项对应的前 7 位编码相同且第 8 为 “#” 时, 则义项相似度为 0.5; 否则, 需要根据两个词语在语义层次树中的距离函数 $dis(c_a^i, c_b^j)$ 计算可得, 即为:

$$sim(c_a^i, c_b^j) = \begin{cases} 1.0, & code(c_a^i) = code(c_b^j) \& endwith "=" \\ 0.5, & code(c_a^i) = code(c_b^j) \& endwith "#" \\ (1.05 - 0.05 * dis(c_a^i, c_b^j))^{\sqrt{\exp\{-k/2n\}}}, & otherwise \end{cases} \quad (4.7)$$

其中, n 是两个词的最邻近公共父结点的密度, k 是两个分支之间的间距, 对于距离函数 $dis(c_a^i, c_b^j)$, 如果两个词不属于同一个大类, 那么距离赋值为 18, 否则距离可以定义为词对连接路径各边权重之和, 令 $l=2,3,4,5$ 代表词项在语义层次树中的层号, 则距离函数可定义为:

$$dis(c_a^i, c_b^j) = \begin{cases} 18, & c_a^i \text{ 和 } c_b^j \text{ 不在同一个大类中} \\ 2 * \sum_l W_{i-l}, & otherwise \end{cases} \quad (4.8)$$

据此, 我们可以得到一个可以衡量两个词汇 w_a 和 w_b 之间语义相似度的实数 $SIM_{cillin}(w_a, w_b) \in [0,1]$, 并通过线性放缩将该数值扩大到 1 到 10 之间作为最终的相似度评分。

(2) 利用 HowNet 计算相似度

根据刘群和李素建的工作^[12], 基于 HowNet 的语义相似度计算方法的基本假设是: 复杂的整体可以被分解成简单的部分, 并且根据部分之间的相似度算得整体的相似度。具体做法可简单概括为: 首先, 将 HowNet 中的词语分为虚词和实词两类。然后, 对于虚词只需计算其对应句法义原或者关系义原相似度, 而对于实词则基于基本假设将实词的整体相似度计算转化为第一基本义原相似度、其它基本义原相似度、关系义原相似度、关系符号相似度四部分局部语义相似度, 最后, 对局部语义相似度加权求和得到整体语义相似度。

具体的计算方法为:

给定两个词汇 w_a 和 w_b , 令 $C_a = \{c_a^1, c_a^2, \dots, c_a^A\}$ 和 $C_b = \{c_b^1, c_b^2, \dots, c_b^B\}$ 分别表示 w_a 和 w_b 的所有概念的集合。与公式 2.1 类似, w_a 和 w_b 之间的语义相似度可以认为是两者所有概念组合之间相似度的最大值, 而 c_a^i 和 c_b^j 两个概念之间的相似度可以表示为:

$$\text{sim}(c_a^i, c_b^j) = \sum_{i=1}^4 \beta_i \prod_{j=1}^i \text{sim}_j(c_a^i, c_b^j) \quad (4.9)$$

其中, $\text{sim}_1(c_a^i, c_b^j), \dots, \text{sim}_4(c_a^i, c_b^j)$ 分别指第一基本义原相似度、其它基本义原相似度、关系义原相似度、关系符号相似度四个局部语义相似度, β_i 代表第 i 个部分的重要程度, 需要满足 $\beta_1 + \beta_2 + \beta_3 + \beta_4 = 1$ 且 $\beta_1 \geq \beta_2 \geq \beta_3 \geq \beta_4$ 。

据此, 我们可以得到一个可以衡量两个词汇 w_a 和 w_b 之间语义相似度的实数 $SIM_{HowNet}(w_a, w_b) \in [0, 1]$, 并通过线性放缩将该数值扩大到 1 到 10 之间作为最终的相似度评分。

(3) 利用 WordNet 计算相似度

本文使用 WordNet 来估计语义相似度具体做法为: 给定两个词汇 w_a 和 w_b , 首先, 在 COW 网络中遍历分别找到 w_a 所有的所有 m 个 Synset 的集合 $Synsets(w_a)$ 以及 w_b 所有的所有 n 个 Synset 的集合 $Synsets(w_b)$, 然后, 遍历 $Synsets(w_a)$ 和 $Synsets(w_b)$ 的所有组合对, 并计算每组的相似度 $\text{sim}_{synset}(i, j)$ 。最后, w_a 和 w_b 的整体相似度为所有组相似度值的最大值, 即

$$SIM_{WordNet} = \max_{1 \leq i \leq m; 1 \leq j \leq n} \{\text{sim}_{synset}(i, j)\} \quad (4.10)$$

其中, $\text{sim}_{synset}(i, j)$ 可以是任意计算两个 Synsets 之间相似度的算法, 往往可用 Synsets 在上-下位词层次结构概念图中相互关联的最短路径来衡量, 本文使用了 Python NLTK 模块中提供的 *path_similarity*, *wup_similarity*, *lch_similarity* 三种方法, 并根据实验结果最终选择了 *path_similarity* 方法, 即基于上位词层次结构概念中相互关联的最短路径的计算相似度。

(4) 利用 Web 检索结果计算相似度

在线 Web 语料库数据量大、实时性好, 这使得许多研究开始关注如何根据搜索引擎获取词语出现、共现及上下文语义片段信息, 或者根据搜索引擎返回文档结果数等统计数据, 从而定义词汇相似度, 这为解决离线语料库难以实时更新、未登录词无法计算的问题提供了新思路。

本文抽取了 web-jaccard、web-overlap、web-dice、web-pmi、NGD 五种基于检索的相似度^[28,29], 具体做法可形式化表示为: 令符号 P 和 Q 代表某个词对中的两个词汇, $H(\cdot)$ 代表以 P 或者 Q 为 Query 查询语句搜索得到的结果条目数, c 为常数阈值(本文中 $c=5$), $P \cap Q$ 表示两者连接的查询语句, N 则表示检索返回的结果条目总数(本文中 $N=10^{16}$)。为了降低算法噪声, 当 $H(P \cap Q) < c$ 时, 认为计算算法失效, 直接赋值为 0, $H(P \cap Q) \geq c$ 时, 则可根据 $H(P)$ 、 $H(Q)$ 和 $H(P \cap Q)$ 进行相似度计算, 即有公式 4.11-4.15:

$$\text{web-jaccard}(P, Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \frac{H(P \cap Q)}{H(P) + H(Q) - H(P \cap Q)}, & H(P \cap Q) \geq c \end{cases} \quad (4.11)$$

$$\text{web-overlap}(P, Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \frac{H(P \cap Q)}{\min\{H(P), H(Q)\}}, & H(P \cap Q) \geq c \end{cases} \quad (4.12)$$

$$\text{web-dice}(P, Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \frac{2 * H(P \cap Q)}{H(P) + H(Q)}, & H(P \cap Q) \geq c \end{cases} \quad (4.13)$$

$$\text{web-pmi}(P, Q) = \begin{cases} 0, & H(P \cap Q) < c \\ \log_2 \frac{H(P \cap Q) / N}{H(P) / N * H(Q) / N}, & H(P \cap Q) \geq c \end{cases} \quad (4.14)$$

$$\text{NGD}(P, Q) = \frac{\max\{\log H(P), \log H(Q)\} - \log_2 H(P \cap Q)}{\log_2 N - \min\{\log H(P), \log_2 H(Q)\}} \quad (4.15)$$

据此, 我们可以得到五种基于 Web 检索的相似度, 从而根据检索结果数来衡量两个词汇 w_a 和 w_b 之间语义相似度, 并通过线性放缩将该数值扩大到 1 到 10 之间作为最终的相似度评分。

4.2.3 最优化词向量

首先, 用 Skip-gram 模型训练一个词向量模型, 也就是我们优化的原始语义空间向量 V 。其次, 从反义词典中抽取反义约束对集合 A, 从同义词典中抽取同义约束对集合 S, 根据得到的多种相似度中的最大值筛选出相似约束集合 R。然后, 根据语义约束 A、S、R 以及词向量模型构造改进 Counter-fitting 模型的目标函数, 最后, 利用随机梯度下降算法原有对词向量模型进行最优化, 在迭代一定次数后达到收敛。

令 V 代表原有词向量空间, V' 代表最优化后的词向量空间, $iter$ 代表当前迭代的次数, max_iter_num 代表设定的最大迭代次, (v, w) 代表词汇表中词 v 和 w 构成的词对, $distance(\bar{v}, \bar{w})$ 代表词向量 \bar{v} 和 \bar{w} 的余弦距离, $gradient$ 表示当前词对 (v, w) 更新梯度向量, $gradient_updates$ 是一个记录梯度向量更新的字典, $update_count$ 则是一个负责记录 $gradient_updates$ 中对应词向量更新次数的字典, 则具体算法的实现如图 4.2 所示。

输入: 原始词向量空间 V , 反义约束集 A , 同义约束集 S , 相似约束集合 R

输出: 优化后新词向量空间 V'

```

L1: iter = 0 and V' = V //初始化迭代次数, 并将原始向量空间赋给新空间为初值
L2: while iter <= max_iter_num do
L3:   for each (v, w) in A do //遍历反义约束, 计算 AR 项
L4:     if distance(V'[v], V'[w]) < delta then //词向量间距离小于 delta 时
L5:       gradient = k1(V'[v] * (V'[v] · V'[w]) - V'[w]); //计算梯度值
L6:       gradient_updates[v] += gradient; update_count[v]++;
L7:   end if
L8: end for
L9: for each (v, w) in S do //遍历同义约束, 计算 SA 项
L10:   if distance(V'[v], V'[w]) > gamma then //词向量间距离大于 gamma 时
L11:     gradient = k2(V'[v] * (V'[v] · V'[w]) - V'[w]); //计算梯度值
L12:     gradient_updates[v] -= gradient; update_count[v]++;
L13:   end if
L14: end for
L15: for each (v, w) in R do //遍历相似约束, 计算 RSI 项
L16:   if distance(V'[v], V'[w]) > theta then //词向量间距离大于 theta 时
L17:     gradient = k3(V'[v] * (V'[v] · V'[w]) - V'[w]); //计算梯度值
L18:     gradient_updates[v] -= gradient; update_count[v]++;
L19:   end if
L20: end for
L21: for each (v, w) in all_word_pairs do //遍历每对词, 计算 VSP 项
L22:   if distance(V[v], V[w]) < distance(V'[v], V'[w]) then
L23:     gradient = k4(V[v] * (V[v] · V'[w]) - V'[w]); //计算梯度值
L24:     gradient_updates[v] += gradient; update_count[v]++;
L25:   end if
L26: end for
L27: for each word in gradient_updates do //更新梯度
L28:   V'[v] += gradient_updates[v] / update_count[v];

```

```

L29:    end for
L30:    iter++;
L31: end while

```

图 4.2 改进 Counter-fitting 算法实现

Fig. 4.2 Improved Counter-fitting Algorithm Implementation

4.3 实验与分析

这部分将主要对第 4 章的实验结果进行展示和分析，进行实验的数据与第 3 章的一致，且评价方法仍然是斯皮尔曼指数 ρ 和皮尔逊指数 r 。

4.3.1 改进模型参数调整实验

本文在 Mrkšić 等人^[45]提出的经验值基础上进行了参数调整实验。反义词虽然相似性较低但是相关性较高，因此，设置反义词向量之间的理想化距离为中间值 $\delta = 0.5$ ；同义词之间的语义相似度非常高，因此，设置同义词向量之间的理想化距离 $\gamma = 0$ ；然后对相似距离下界 θ 、参数 k_1, k_2, k_3, k_4 、半径 ρ 分别进行参数估计。由于半径 ρ 对于计算速度影响较大，因此，先根据经验值设定一个较小的值，最后再对 ρ 进行估计。

参数估计使用的 dev 开发集语料是从 PKU-500 中随机抽取的 200 条词对以及 PKU-40 共同组成的 240 条词对，同时，将 PKU-500 中没有抽到的 300 条词对作为 test 测试语料。按照此种方法抽取了 5 组实验语料进行参数估计和性能评估。性能评价指标则使用 Spearman 相关系数。参数的默认初始值分别为 $\theta = 0$ ， $k_1 = k_2 = k_3 = 0.1$ ， $\rho = 0.2$ 。

(1) 相似距离下界 θ

相似距离下界 $\theta \in [0,1]$ 表示两个词语可以被判定为相似关系显著的最大距离，或者可以理解为两个词之间的相似度大于等于 $1 - \theta$ 时可以被认为相似性较高。本文先假定在其他参数保持初始值的情况下，以步长 0.01 对 $\theta \in [0,0.4]$ 的可行解进行搜索。

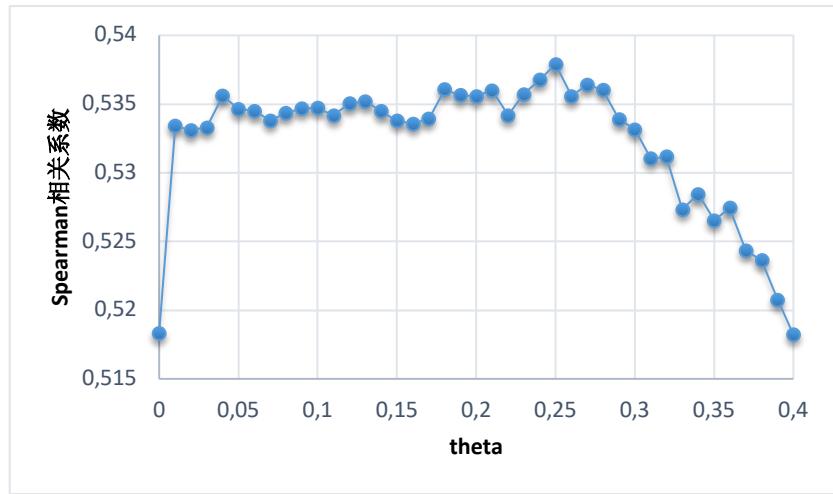
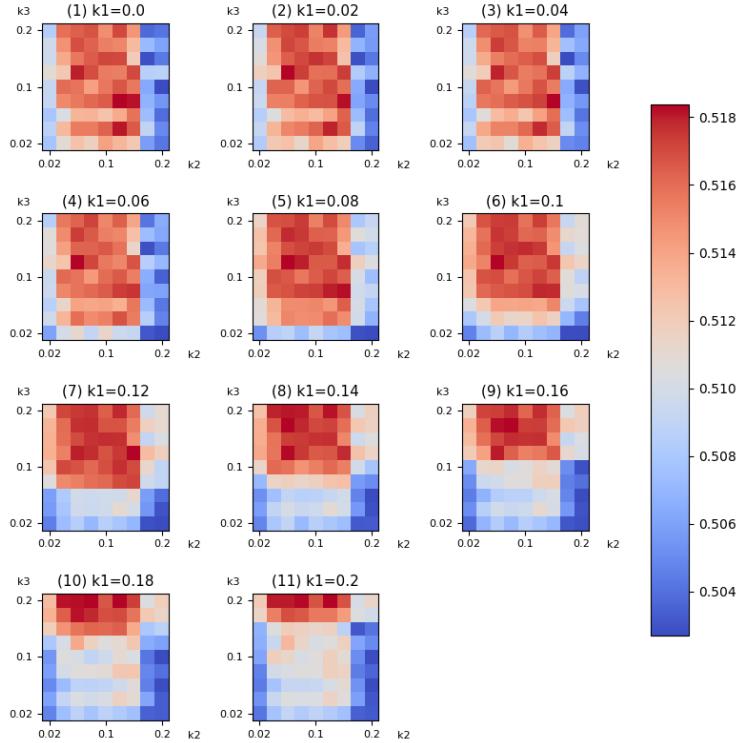
图 4.3 Spearman 相关系数在不同 θ 值时的变化曲线Fig. 4.3 Spearman's Rank Correlation Coefficient with Different θ Value

图 4.3 显示了不同 θ 值所得到的 Spearman 相关系数，其中的每个数据点是根据 5 组实验的平均值计算得出。从图 4.3 中可以看出，在 $\theta = 0.25$ 时，Spearman 指数取得了最高值，因此将 θ 设置为 0.25。

(2) 参数 k_1, k_2, k_3, k_4

多项式系数 k_1, k_2, k_3 的经验值为 0.1，本文将对其他参数保持不变的情况下，分别在 $[0, 0.2]$ 范围内以步长 0.02 对 k_1, k_2, k_3 的可行解进行搜索，最后利用 $k_4 = 1 - k_1 - k_2 - k_3$ 求解 k_4 。图 4.4 是在 k_1 在 11 组不同取值下， k_2, k_3 对 Spearman 相关系数的影响，其中的每个数据点是根据 5 组实验的平均值计算得出，暖色越深，性能越好。从图 4.4 可以看出 $k_2 \in [0.08, 0.16]$ 和 $k_3 \in [0.08, 0.2]$ 时性能较好且变化不大， $k_1 \in [0.0, 0.1]$ 时结果受 k_2, k_3 变化的影响较小，性能比较稳定。根据实验数据，本文得到 $k_1 = 0.02, k_2 = 0.08, k_3 = 0.14$ 时，Spearman 相关系数取得了最高值。分析参数比例，可以看出取得最值时同义约束和反义约束不如相似约束对的总体重要度高，这可能因为受到语义词典规模的限制，评价集中同义约束、反义约束对较少造成的，也直接说明了引入相似约束的必要性。

图 4.4 k_1, k_2, k_3 不同值时的 Spearman 相关系数Fig. 4.4 Spearman's Rank Correlation Coefficient with Different k_1, k_2, k_3 (4) 半径 ρ

在实际计算 VSP 项时, 为了保证计算过程的速度, 仅计算在词向量空间中当前词与其周围特定半径 ρ 覆盖的词汇之间的距离。本文以步长 0.1 对 $\rho \in [0,1]$ 上的可行解进行搜索, 结果如图 4.5 所示, 不难看出, $\rho = 0.3$ 时性能达到最佳, 且 $\rho > 0.3$ 后不同 ρ 对结果影响很小, 又考虑 ρ 在实际操作中对计算时间影响较大, 因此, 设定 $\rho = 0.3$ 。

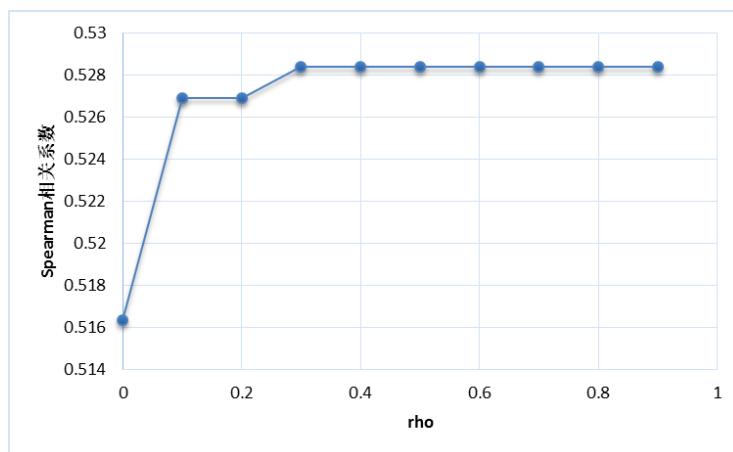


图 4.5 Spearman 相关系数在不同半径 ρ 值时的变化曲线Fig. 4.5 Spearman's rank correlation coefficient with different radius ρ value

4.3.2 改进模型的稳定性分析

在 4.3.1 节中，我们随机抽取 5 组实验组，并利用其中的开发集进行了参数估计。在此基础上，为了证明本文提出的改进模型对原有词向量模型的提升作用的稳定性，本文利用调节好的参数，再分别对这 5 个实验组中的测试集进行测试，结果如表 4.1 所示。

表 4.1 五组随机采样实验结果

Tab. 4.1 Result of five groups of Five random sampling experiments

Group	Spearman ρ	Pearson r
1	0.551	0.525
2	0.527	0.511
3	0.528	0.491
4	0.526	0.467
5	0.506	0.453
AVG	0.528	0.489

从表 4.1 可以看出，5 组随机采样实验的结果均明显高于最优化前词向量的结果 ($0.311 \rho / 0.311 r$)，由此可见，在词向量中融入语义约束的方法可以有效提升词汇语义相似度计算结果，且其提升作用具有普适性和稳定性。

4.3.3 改进模型的有效性分析

为了证明我们提出的融入语义的词向量模型——改进 Counter-fitting 模型的有效性，我们进行了 3 组对比实验，结果如表 4.2 所示，No.1 代表优化前的标准 Skip-gram 模型的实验结果，No.2 代表仅仅融入了同义、反义两种语义约束的基本 Counter-fitting 模型的实验结果，No.3 代表改进 Counter-fitting 模型的实验结果，其中，不仅融入了同义、反义约束，而且融入了基于词典和检索结果的相似语义约束。

表 4.2 基线方法、基本及改进 counter-fitting 实验结果对比

Tab. 4.2 Result of improvement by basic counter-fitting and improved counter-fitting

No.	方法	ρ	r
1	Baseline: Skip-gram	0.311	0.311
2	Baseline	0.520	0.496

	+Basic Counter-fitting	Baseline	0.552	0.513
3				
	+Improved Counter-fitting			

由表 4.2 可知, No. 2 组基本 Counter-fitting 模型的结果分别为 $0.520 \rho / 0.496 r$, 比最优化之前的标准 Skip-gram 模型的结果分别提高了 0.209ρ (67.20%) 和 $0.185 r$ (59.49%), No. 3 组改进 Counter-fitting 模型的结果达到 $0.552 \rho / 0.513 r$, 比基本 Counter-fitting 模型的结果分别提高了 0.032ρ (6.15%) 和 $0.017 r$ (3.43%)。不难看出, 融入同义、反义语义约束对提升已有词向量模型效果非常明显, 而且融入相似语义约束能够有效改进基本 Counter-fitting 模型。

如图 4.6 显示了 20 次迭代内的基本 Counter-fitting 模型和改进 Counter-fitting 模型的学习曲线。从中我们不难看出, 两种模型均在 20 次迭代内达到收敛状态, 这说明两种优化方法在有限次迭代内可以找到可行解, 而且, 达到收敛状态后, 改进模型结果显著好于基本模型结果。其次, 在第 7 次迭代之后, 两种方法性能接近其本身最优性能, 说明短周期迭代可达到其对应的最大值附近, 因此, 两种模型在运算时间方面也具有一定优势。此外, 在 10~20 次迭代过程中, 基本模型性能出现了轻微的下降, 相比之下, 改进模型显示出了更加稳定的性能。

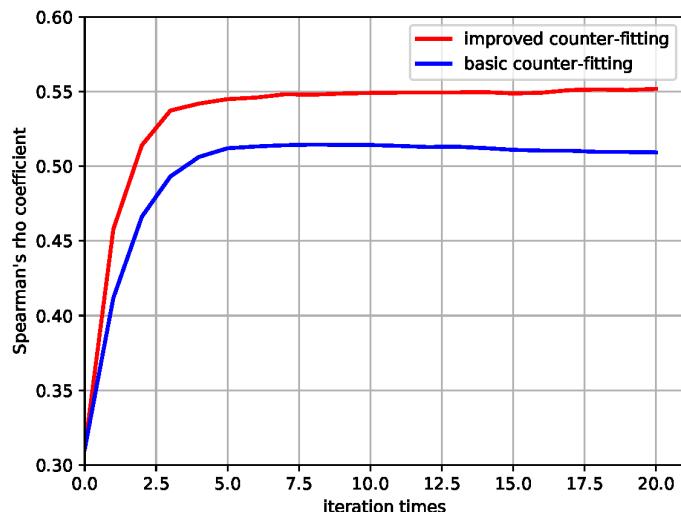


图 4.6 基本 Counter-fitting 和改进 Counter-fitting 的学习曲线

Fig. 4.6 The Learning Curves of the Basic Counter-fitting and Improved Counter-fitting Method

为了进一步分析实验结果，我们选出了 11 对例子（如表 4.3 所示）可以明显看出 Counter-fitting 方法的改进，并将其对应词向量经过主成分分析（PCA）算法降维后可视化展示于图 4.7 中。

表 4.3 改进最显著的 11 对词

Tab. 4.3 The Top 11 Word Pairs Change Better after Counter-fitting

No.	词对	人工评分	改进前	改进后
1	睡/寐	8.4	5.375	10.00
2	羸/胜	9.8	6.616	9.888
3	行/可以	9.2	6.121	7.616
4	恶劣/坏	8.4	5.318	9.844
5	服气/口服心服	8.7	5.629	10.00
6	滚蛋/滚开	9.4	6.364	10.00
7	拖后腿/拉后腿	9.5	6.451	10.00
8	书籍/图书馆	3.4	7.589	6.582
9	利润/成本	3.2	7.694	6.923
10	互联网/因特网	9.5	6.895	9.999
11	计算机/电脑	9.9	7.397	10.00

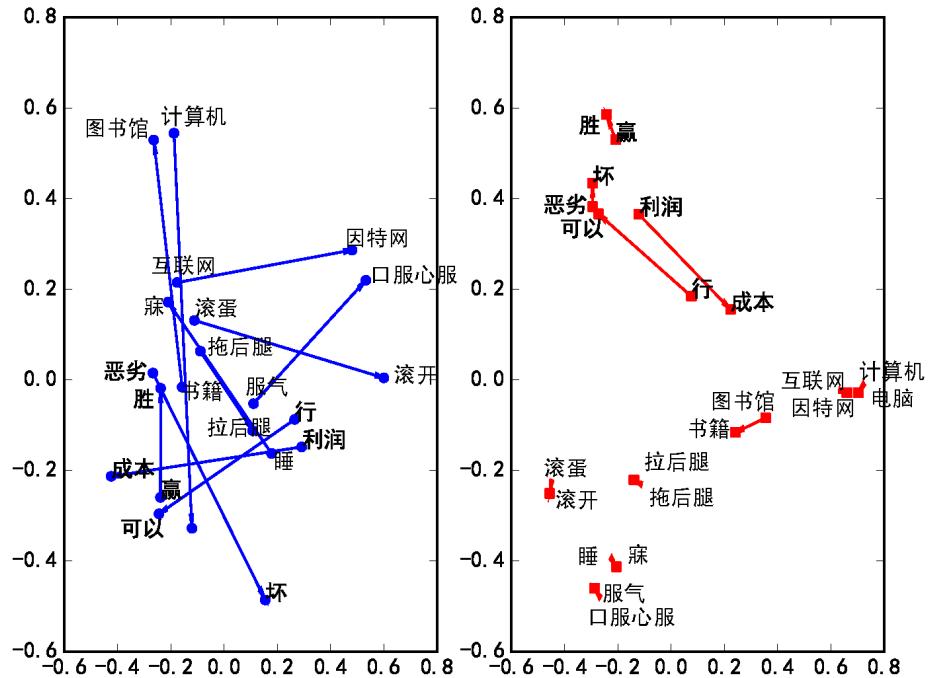


图 4.7 Counter-fitting 前后词对分布情况
Fig. 4.7 Distribution of Word Pairs before and after Counter-fitting

为了进一步分析改进 Counter-fitting 方法有效的原因，我们将从解决基本词向量模型存在的问题的角度进行讨论：

(1) “缺乏”训练语料。互联网虽然提供了一个很好的平台供人们访问大量的语料，但是，这些语料大部分覆盖标准、规范的书面语，却仍然很难完全覆盖其他语言现象。例如，单字词（No.1-4），成语（No.5）和口语化的词汇（No.6-7），这类词汇的训练文本实际上是比较稀少的。

(2) 基于语料库的词向量的“先天缺点”。这种基于分布假设的词向量很难区分“语义相似”和“概念关联”，以 No.8-9 为例，这两组词对虽然在语义上并不相似，但由于高频共现，彼此之间有很强的联系，容易过高评分。

(3) 对于高分评判太过“谨慎”现象。根据经验，词向量模型很少给两个词很高的相似分数，分数 0.6~0.7（最大值为 1）已经是相对较高的分数了，例如，(computer, laptop) 这对在语义上已经是非常接近的词，根据 GoogleNews-vectors-negative300.bin 查找其对应词向量并计算其余弦相似度仅仅为 0.66，而人类打分时就可能评分较高。

4.3.4 不同模型上的对比实验

本章分别利用了不同模型计算相似度，其中主要包括四类：基于语义词典的模型、基于 Web 检索的模型、词向量模型、融入语义约束的词向量模型。本节将分别通过实验来分析、比较同类别内的方法以及不同类别间的方法。

表 4.4 分别从总体评价、登录词评价、覆盖率和词典规模四个方面分析本文实现的三种基于词典方法。其中，登录词评价是指只评价词对中两个词语均在词典中出现的部分语料，覆盖率是两个成员词语均在词典中出现的词对占语料中总词对数的比率。

表 4.4 不同基于词典的方法对比
Tab. 4.4 Comparison among Different Lexicon-based Methods

No.	方法	总体评价 ρ / r	登录词评价 ρ / r	覆盖率 %	词典规模 个词
1	同义词林	0.422/0.472	0.520/0.550	90.8	77,343
2	HowNet	0.258/0.336	0.483/0.490	76.4	11,000
3	CWordNet	-0.053/-0.001	0.520/0.606	31.2	61,533

从总体评价上看，当词对中任意一个词不在词典中，则认为词典方法计算失效，相似度评分赋值为最低分（1分），此时，基于同义词林的方法性能（ $0.422 \rho / 0.472 r$ ）明显好于 HowNet 方法（ $0.258 \rho / 0.336 r$ ）和中文 WordNet 方法（ $-0.053 \rho / -0.001 r$ ）。分析其原因可能是，同义词林中词汇覆盖率（90.8%）以及词典规模（77,343个词）明显高于 HowNet 和中文 WordNet。而中文 WordNet 在总体评价上出现了负相关，可以说明大量未登录词的存在极大地影响了语义词典方法的性能。

从登录词评价上看，当词对中任意一个词都在语义词典中可以查找到时，中文 WordNet 性能（ $0.520 \rho / 0.606 r$ ）明显好于同义词林方法（ $0.520 \rho / 0.550 r$ ）和 HowNet 方法（ $0.483 \rho / 0.490 r$ ），由此不难推测，WordNet 虽然在实验语料上覆盖率较低（31.2%），但是含有更加丰富的语义信息，有利于捕获更多语义关系从而定义更加合理的语义相似度计算方法。

表 4.5 是五种基于 Web 检索结果计算语义相似度的实验结果。从该表中不难看出，web-pmi 的性能（ $0.275 \rho / 0.311 r$ ）远远好于同表中其他的方法，其次是 web-overlap（ $0.200 \rho / 0.136 r$ ），web-dice（ $0.138 \rho / 0.130 r$ ）和 web-jaccard 性能较差（ $0.138 \rho / 0.130 r$ ），NGD 的结果甚至出现了负值（ $-0.131 \rho / -0.130 r$ ）。结合表 3.3 总体来看，Web 检索结果模型中的 web-pmi 方法性能不如语义词典模型中的同义词林方法性能，但是好于 HowNet 和中文 WordNet，这说明登录词覆盖率高时词典方法具有优势，反之基于 Web 检索结果的方法更有优势，因此，当存在大量未登录词时，web-pmi 是可以考虑的重要因素之一。

表 4.5 不同 Web 检索相似度方法对比

Tab. 4.5 Comparison among Different Retrieval-based Methods

No.	方法	ρ	r
1	web-jaccard	0.138	0.141
2	web-overlap	0.200	0.136
3	web-dice	0.138	0.130
4	web-pmi	0.275	0.311
5	NGD	-0.131	-0.130

为了进一步探讨不同类型方法的性能，本文从表 4.6 中分别列出了四类模型在本文中表现最佳的方法，分别为：语义词典模型中性能表现最好的同义词林方法，Web 检索

结果模型中表现最好的 web-pmi 方法，标准词向量模型中训练效果最好的词向量方法，融入语义约束的词向量模型中的改进 Counter-fitting 方法。

表 4.6 不同类型模型实验结果对比

Tab. 4.6 Comparison among Different Types of Models

No.	方法	ρ	r
1	同义词林	0.422	0.472
2	词向量	0.311	0.311
3	web-pmi	0.275	0.311
4	改进 Counter-fitting	0.552	0.513

从 No.1-3 组实验结果我们可以看出，基于词典的方法领先于基于 Web 检索结果的方法和基于词向量的方法，分析其中的原因可能为语义词典由人工构建，在登录词覆盖率较高的情况下，有着先天优势。此外，对比 No.4 组实验和 No.1-3 组实验的结果，我们可以认为，融入语义约束的词向量模型的性能往往好于语义词典模型、传统统计模型和词向量模型，且提升效果比较明显，分析其中原因可能是该模型可以整合语义词典、统计规律及词向量等多元化资源，具有优势互补的特性。No.4 组结果达到 0.552ρ ，比 No.1-3 组结果分别提升了 0.130ρ ， 0.241ρ 和 0.277ρ ，直接证明了本文方法的有效性。

4.3.5 其它语料上的对比实验

现有英文词汇相似度计算的公共评测语料较多，较为知名的主要包括：RG-65，MC-30，WordSim-353，Huang 提出的包含语境的数据集等^[53]。然而，中文词汇相似度评测语料较为稀缺，除了目前较新的 PKU-500 之外，也有部分研究者将英语语料翻译成中文语料进行测试，例如，朱新华等^[23]根据词性、词义将 MC-30 翻译成对应的中文对，并直接用英文数据的人工判定值作为最终评分值，得到了中文 MC-30 数据集。评分范围为 0~1 分，评分越高，相似度越高。

为了研究本文方法在不同语料上的性能，文本将在中文 MC-30 上进行实验，并与在此数据集上进行实验的 5 种方法进行比较。由于本文主要使用 PKU-500 评分标准不同于 MC-30，因此，选择只与相对排序数有关的 Spearman 相关系数 ρ 来评价本文提出的方法与其它方法在该数据集上的性能，实验结果如表 4.7 所示。

表 4.7 中文 MC-30 语料上不同模型实验结果对比

Tab. 4.7 Comparison among Different Models on Chinese MC-30

No.	方法	ρ
1	同义词林（田久乐等 ^[11] ）	0.520
2	HowNet（刘群 ^[12] ）	0.682
3	词向量	0.697
4	web-pmi	0.371
5	同义词林+ HowNet（朱新华等 ^[23] ）	0.847
6	改进 Counter-fitting	0.731

分析表 4.7 中 No.1-4,6 组结果我们可以得到与表格 4.6 类似的规律，即融入语义约束的词向量模型的性能往往好于语义词典模型、传统统计模型和词向量模型，且提升效果比较明显，No.6 相比于 No.1-4 结果分别提升了 0.211ρ , 0.049ρ , 0.034ρ , 0.360ρ 。No.5 结果优于 No.6 结果，分析原因可能是因为该方法结合了两种词典进行计算，能够充分利用语义两种词典在高登录率的小数据集上的天然优势，因此，不能简单地认为这种语义词典结合的方法优于改进 Counter-fitting 方法。

结 论

随着互联网时代的到来，海量信息不断生成，信息过载问题给人们带来的影响日益显著。面对规模如此庞大的互联网信息资源，如何自动理解自然语言中所隐含的语义，从而迅速而准确地获取有效信息，是亟待解决的基础而核心的问题之一。

词汇是能够表达完整语义的基本单元，是理解一切自然语言的根本。词汇语义相似度是对两个词语对象中蕴含的意义相近程度的度量。词汇语义相似度计算是一项基础而核心的工作，可以将“词汇相似”这个抽象关系通过特定的计算方法映射成计算机可以处理的数值，从而将自然语言处理问题转化为机器学习问题，其性能的好坏将直接影响到自然语言处理与信息检索的各项任务。近年来，基于词向量的词汇语义相似度计算方法及其改进方法已成为该领域内的前沿、热点研究课题。

本文研究中文词汇语义相似度计算，重点研究了如何改进基于词向量的词汇语义相似度计算方法，并根据是否融入语义知识分为两部分研究：

(1) 无语义约束的词向量模型

分别利用机器翻译选择性替换中文向量和 LSTMs 学习词对共现句子提升词向量模型的性能。实验结果证明，基于机器翻译和 LSTMs 的改进模型相对于标准词向量模型效果略有提高。

(2) 融入语义约束的词向量模型

在基本 Counter-fitting 的模型基础上，不仅考虑同义关系约束、反义关系约束和向量空间存留，还引入了相似关系约束。这样不仅局限于利用有限的强语义关系提升词向量，还能将基于多种语义资源得到的弱相似关系融入到词向量的改进过程中。

实验结果表明：①改进 Counter-fitting 模型性能优于标准 Skip-gram 词向量模型及基本 Counter-fitting 模型，由此可见，融入语义约束可有效提升词向量模型的性能。②基于语义词典的方法在登录词覆盖率较高的情况下，有着先天优势，而当出现大量未登录词时，基于词向量方法和基于检索结果的方法更具有实用性。

目前的研究中仍然存在一些不足以及预期改进方案为：

(1) 在相似词和相关词的区分上仍然改进不明显。例如，在使用 LSTMs 学习词对共现的句子时，会认为当两个词共现在同一个句子中时它们之间的关系会更加“紧密”。然而，反义词出现在并列结构时的情况应该加以区分。下面是一个相似性较低的词对存在于并列结构句子的例子。

词对：（亏损，盈利，score= 3.9）

句子：在过去的十年中，日航在巨额【亏损】和小额【盈利】中摇摇摆摆。

改进预想：可以发现并列结构中的对比关系，例如“巨额”和“小额”，通过规则的方式对这类情况的相似度进行修正。

(2) 现有改进 Counter-fitting 模型仍然是处于分阶段优化过程，抽取语义相似约束时用到的其他相似度计算模型计算结果值，其中含有一些参数是各个模型独立训练得到的，有时不一定得到全局最优结果。改进预想是将分阶段优化过程改为联合优化过程，在同一个学习过程中得到最佳模型。

参 考 文 献

- [1] Sinha R, Mihalcea R. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity [C]. Proceedings of the 1st IEEE International Conference on Semantic Computing, Irvine, 2007: 363–369.
- [2] 于东, 荀恩东. 基于 Word Embedding 语义相似度的字母缩略术语消歧[J]. 中文信息学报, 2014, 28(5): 51–59.
- [3] Pirrò G, Ruffolo M, Talia D. SECCO: on building semantic links in Peer-to-Peer networks[J]. Journal on Data Semantics, 2009, 12: 1–36.
- [4] Meilicke C, Stuckenschmidt H, Tamlin A. Repairing ontology mappings[C]. Proceedings of the 22nd AAAI Conference on Artificial Intelligence, Vancouver, 2007: 1408–1413.
- [5] Qumsiyeh R, Ng Y K. Assisting web search using query suggestion based on word similarity measure and query modification patterns[J]. World Wide Web, 2014, 17(5): 1141–1160.
- [6] Pal D, Mitra M, Datta K. Improving query expansion using WordNet[J]. Journal of the Association for Information Science and Technology, 2014, 65(12): 2469–2478.
- [7] Mihalcea R, Corley C, Strapparava C. Corpus-based and knowledge-based measures of text semantic similarity[C]. Proceedings of the 21st National Conference on Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference, Boston, 2006: 775–780.
- [8] Islam A, Inkpen D. Semantic text similarity using corpus-based word similarity and string similarity[J]. ACM Transactions on Knowledge Discovery from Data, 2008, 2(2): 10:1–10:25.
- [9] 刘萍, 陈烨. 词汇相似度研究进展综述[J]. 现代图书情报技术, 2012, 28(7): 82–89.
- [10] 赵军, 胡栓柱, 樊兴华. 一种新的词语相似度计算方法[J]. 重庆邮电大学学报(自然科 学版), 2009, 21(4): 528–532.
- [11] 田久乐, 赵蔚. 基于同义词词林的词语相似度计算方法[J]. 吉林大学学报: 信息科学版, 2010, 28(6): 602–608.
- [12] 刘群, 李素建. 基于《知网》的词汇语义相似度计算[J]. 中文计算语言学, 2002, 7(2): 59–76.
- [13] 张亮, 尹存燕, 陈家骏. 基于语义树的中文词语相似度计算与分析[J]. 中文信息学报, 2010, 24(6): 23–31.
- [14] 吴思颖, 吴扬扬. 基于中文 WordNet 的中英文词语相似度计算[J]. 郑州大学学报: 理学版, 2010, 42(2): 66–69.

- [15] Wang S, Bond F. Building the Chinese Open Wordnet (cow): Starting from Core Synsets[C]. Proceedings of the 11th Workshop on Asian Language Resources, Nagoya, 2013: 10–18.
- [16] Ahsaee M G, Naghibzadeh M, Naeini S E Y. Semantic similarity assessment of words using weighted WordNet[J]. International Journal of Machine Learning and Cybernetics, 2014, 5(3): 479–490.
- [17] Radinsky K, Agichtein E, Gabrilovich E, et al. A word at a time: computing word relatedness using temporal semantic analysis[C]. Proceedings of the 20th International Conference on World Wide Web, Hyderabad, 2011: 337–346.
- [18] 盛志超. 基于维基百科的语义比较[D]: 硕士学位论文. 上海: 复旦大学, 2011.
- [19] 詹志建, 梁丽娜, 杨小平. 基于百度百科的词语相似度计算[J]. 计算机科学, 2013, 40(6): 199–202.
- [20] 李茹, 王智强, 李双红, 等. 基于框架语义分析的汉语句子相似度计算[J]. 计算机研究与发展, 2013, 50(8): 1728–1736.
- [21] Liang C, Shao Y, Zhao J. Construction of a Chinese Semantic Dictionary by Integrating Two Heterogeneous Dictionaries: TongYiCi Cilin and HowNet[C]. Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies, Atlanta, 2013: 203–207.
- [22] Zhang P, Zhang Z, Zhang W. An approach of semantic similarity by combining HowNet and Cilin[C]. Proceedings of GreenCom/iThings/CPScom, Beijing, 2013: 1638–1643.
- [23] 朱新华, 马润聪, 孙柳, 等. 基于知网与词林的词语语义相似度计算[J]. 中文信息学报, 2016, 30(4): 29–36.
- [24] KARIN A, BENGT A. 语料库语言学的进展[M]. 北京: 时间图书出版公司, 2009.
- [25] 李慧. 词语相似度算法研究综述[J]. 现代情报, 2015, 35(4): 172–177.
- [26] Turney P D. Mining the web for synonyms: PMI–IR versus LSA on TOEFL[C]. Proceedings of the 12th European Conference on Machine Learning, Freiburg, 2001: 491–502.
- [27] Higgins D. Which statistics reflect semantics? Rethinking synonymy and word similarity[J]. Linguistic Evidence: Empirical, Theoretical and Computational Perspectives, Studies in Generative Grammar, 2005, 85: 265–284.
- [28] Bollegala D, Matsuo Y, Ishizuka M. Measuring semantic similarity between words using web search engines[C]. Proceedings of the 16th International Conference on World Wide Web, Banff, 2007: 757–766.
- [29] Cilibrasi R L, Vitanyi P M B. The google similarity distance[J]. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(3): 370–383.

- [30] Turian J, Ratinov L, Bengio Y. Word representations: a simple and general method for semi-supervised learning[C]. Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, 2010: 384–394.
- [31] Collobert R, Weston J, Bottou L, et al. Natural language processing (almost) from scratch[J]. Journal of Machine Learning Research, 2011, 12: 2493–2537.
- [32] Socher R, Pennington J, Huang E H, et al. Semi-supervised recursive autoencoders for predicting sentiment distributions[C]. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Edinburgh, 2011: 151–161.
- [33] Bengio Y, Ducharme R, Vincent P, et al. A neural probabilistic language model[J]. Journal of Machine Learning Research, 2003, 3: 1137–1155.
- [34] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.
- [35] Pennington J, Socher R, Manning C D. Glove: Global Vectors for Word Representation [C]. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Doha, 2014: 1532–1543.
- [36] Turney P D, Pantel P. From frequency to meaning: Vector space models of semantics[J]. Journal of Artificial Intelligence Research, 2010, 37: 141–188.
- [37] Hill F, Reichart R, Korhonen A. Simlex-999: Evaluating semantic models with (genuine) similarity estimation[J]. Computational Linguistics, 2016, 41(4): 665–695.
- [38] Ono M, Miwa M, Sasaki Y. Word Embedding-based Antonym Detection using Thesauri and Distributional Information[C]. The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, 2015: 984–989.
- [39] Iacobacci I, Pilehvar M T, Navigli R. SensEmbed: Learning Sense Embeddings for Word and Relational Similarity[C]. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, 2015: 95–105.
- [40] Yu M, Dredze M. Improving Lexical Embeddings with Semantic Knowledge[C]. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, 2014: 545–550.
- [41] Liu Q, Jiang H, Wei S, et al. Learning Semantic Word Embeddings based on Ordinal Knowledge Constraints[C]. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint

- Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Beijing, 2015: 1501–1511.
- [42] Nguyen K A, im Walde S S, Vu N T. Integrating Distributional Lexical Contrast into Word Embeddings for Antonym–Synonym Distinction[C]. The 54th Annual Meeting of the Association for Computational Linguistics, Berlin, 2016: 454–459.
- [43] Rothe S, Schütze H. AutoExtend: Extending Word Embeddings to Embeddings for Synsets and Lexemes[C]. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Beijing, 2015: 1793–1803.
- [44] Faruqui M, Dodge J, Jauhar S K, et al. Retrofitting Word Vectors to Semantic Lexicons[C]. The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, 2015: 1606–1615.
- [45] Mrkšić N, Séaghdha D O, Thomson B, et al. Counter-fitting word vectors to linguistic constraints[J]. arXiv preprint arXiv:1603.00892, 2016.
- [46] Quine W V. Ontological relativity[J]. Journal of Philosophy, 1968, 65(7): 185–212.
- [47] Harispe S, Ranwez S, Janaqi S, et al. Semantic similarity from natural language and ontology analysis[J]. Synthesis Lectures on Human Language Technologies, 2015, 8(1):1–254.
- [48] 伯德, 克莱恩, 洛佩尔. Python 自然语言处理[M]. 北京: 人民邮电出版社, 2014.
- [49] Harris Z S. Distributional structure[J]. Word, 1954, 10(2–3): 146–162.
- [50] Firth J R. A synopsis of linguistic theory, 1930–1955[J]. Studies in linguistic analysis, 1957.
- [51] 来斯惟. 基于神经网络的词和文档语义向量表示方法研究[D]: 博士学位论文. 北京: 中国科学院大学, 2016.
- [52] Graves, Alan, Abdel-rahman Mohamed, Geoffrey Hinton. Speech recognition with deep recurrent neural networks [C]. IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, 2013: 6645–6649.
- [53] Wu Y, Li W. NLPCC–ICCPOL 2016 shared task 3: Chinese word similarity measurement[C]. International Conference on Computer Processing of Oriental Languages, Kunming, 2016: 828–839

攻读硕士学位期间发表学术论文情况

- 1 Combining Word Embedding and Semantic Lexicon for Chinese Word Similarity Computation. **Pei, J.**, Zhang, C., Huang, D., & Ma, J. NLPCC-ICCPOL, 2016, pp. 766-777. 主办单位: 中国计算机学会. EI 检索会议, 本文 EI 检索号: 20165303205378.
(本硕士学位论文第三、四章)
- 2 DUT-NLP-CH @ NTCIR-12 Temporalia Temporal Intent Disambiguation Subtask. **Pei J.**, Huang D., Ma J., et al. Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies, 2016, pp.253-257.

本文获得国家自然科学基金资助

- 1 国家自然科学基金 (61672127) “基于深度学习的句子相似度计算研究”

致 谢

首先，非常感谢我的导师黄德根教授，在学术上，他是我的良师，在我的选题、研究方向上悉心指导，在平时科研工作中对我充分信任，并鼓励和支持我参与领域内的学术会议增长见识、多多锻炼；在生活中，他是既是长辈又是诤友，情绪低落时总能给我支持和鼓励。感谢马建军教授，她对科研的执着、对生活的热爱、对他人的热情让我受益匪浅。同时感谢李丽双和周惠巍老师，她们在科研上的严谨、专注和热忱都非常值得我学习，是我的学术生涯中的好榜样。

其次，感谢实验室所有的兄弟姐妹们，在科研上，共同营造了一个安静融洽的学习氛围，在生活中，给我最珍贵的陪伴、关照与鼓励。尤其感谢蒋振超、李丹、张聪、梁晨等同学，他们对我的科研工作给予了很大的帮助。

感谢我的舍友对我的包容与关怀，和她们一起度过的时光虽然短暂，但是非常幸福难忘。感谢我的同学对我工作上的支持和生活上的关照，和大家一起成长的青春弥足珍贵。

感谢我的父母、亲人一直以来给我生活上的关怀和包容、学业上鼓励和支持，他们是我不断努力奋斗的不竭动力。感谢军晓哥多年陪伴，在我遇到困难和坎坷时，总能给我开导和帮助。

最后，衷心感谢三年时光中认识的所有人，缘分使得我们相遇，我会把记忆穿成思念的线，永记你们带给我的温暖。

大连理工大学学位论文版权使用授权书

本人完全了解学校有关学位论文知识产权的规定，在校攻读学位期间论文工作的知识产权属于大连理工大学，允许论文被查阅和借阅。学校有权保留论文并向国家有关部门或机构送交论文的复印件和电子版，可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印、或扫描等复制手段保存和汇编本学位论文。

学位论文题目：_____

作者签名：_____ 日期：____年____月____日

导师签名：_____ 日期：____年____月____日