一种基于词义向量模型的词语语义相似度算法

李小涛 [1] 游树娟 [1] 陈 维 [1]

摘 要 针对基于词向量的词语语义相似度计算方法在多义词、非邻域词和同义词三类情况计算准确性差的问题, 提出了一种基于词义向量模型的词语语义相似度算法. 与现有词向量模型不同, 在词义向量模型中多义词按不同词义被分成多个单义词, 每个向量分别与词语的一个词义唯一对应. 我们首先借助同义词词林中先验的词义分类信息, 对语料库中不同上下文的多义词进行词义消歧; 然后基于词义消歧后的文本训练词义向量模型, 实现了现有词向量模型无法完成的精确词义表达; 最后对两个比较词进行词义分解和同义词扩展, 并基于词义向量模型和同义词词林综合计算词语之间的语义相似度. 实验结果表明本文算法能够显著提升以上三类情况的语义相似度计算精度.

关键词 词语语义相似度, Word2vec, 同义词词林, 词义消歧, 词义向量

引用格式 李小涛, 游树娟, 陈维. 一种基于词义向量模型的词语语义相似度算法. 自动化学报, 2020, **46**(8): 1654−1669

DOI 10.16383/j.aas.c180312

开放科学 (资源服务) 标识码 (OSID):

An Algorithm of Semantic Similarity Between Words Based on Word Single-meaning Embedding Model

LI Xiao-Tao[1] YOU Shu-Juan[1] CHEN Wai[1]

Abstract We propose a novel algorithm of semantic similarity between words, based on our word single-meaning embedding model, to address the issue of existing word-embedding-based approaches that have low computation accuracy in polysemous words, nonadjacent words and synonyms. Differently from the existing word embedding models, each polysemous word is decomposed into a series of monosemous words in our model, and there is a one-to-one correspondence between a word meaning and a vector. First of all, the word sense disambiguation (WSD) of polysemous words in different contexts of the corpus is achieved with the help of the prior classification information contained in Tongyici Cilin. Then, the word single-meaning embeddings are learned from the processed corpus and realize the precise expression for each word meaning, and as far as we know, no existing word embedding model could complete this task. At last, two test words are decomposed into marked monosemous words according to the number of meaning and expanded with synonyms, and then semantic relatedness between words is computed based on the word single-meaning embedding model and Tongyici Cilin. The experimental results showed our method can significantly improve the computation accuracy of polysemous words, nonadjacent words and synonyms.

Key words Semantic similarity, Word2vec, Tongyici Cilin, word sense disambiguation (WSD), word single-meaning embedding model

Citation Li Xiao-Tao, You Shu-Juan, Chen Wai. An algorithm of semantic similarity between words based on word single-meaning embedding model. *Acta Automatica Sinica*, 2020, **46**(8): 1654−1669

词语的语义相似度作为自然语言处理领域的重要研究方向, 已经广泛应用于词义消歧、知识管理中信息抽取、语义标注以及本体学习与合并、Web 服务发现等相关领域[1]. 词语语义相似度计算的准确性直接影响以上领域相关算法的性能.

目前, 词语语义相似度的计算方法大部分基于本体和语义词典, 利用词语节点间的位置关系来衡量词语间的语义相似程度, 但存在着词汇量不足、扩展性差和准确性不高的问题. 基于词向量的语义相似度算法通过从包含海量词汇的语料库中训练词向量模型, 利用词语对应的向量之间的距离来计算词语之间的语义相似度. 这种方法能够有效解决现有的基于本体和基于语义词典的方法存在的缺陷, 但是随之也引入了一些新的问题. 首先, 许多词语能够表达多个词义, 如 “仪表” 既可以表示人的外表, 也可以表示测量仪器, 这类词语称为多义词. 现有的词向量模型对于每个词语使用单一的词向量表示, 多义词也不例外. 每个多义词的词向量实际上是多个词义的一个折中, 这在一定程度上弱化了每个单独的词义, 利用词向量的距离计算的多义词之

收稿日期 2018-05-16 录用日期 2018-08-23
Manuscript received May 16, 2018; accepted August 23, 2018
本文责任编委 张民
Recommended by Associate Editor ZHANG Min
1. 中国移动研究院 北京 100053
1. China Mobile Research Institute, Beijing 100053

间的语义相似度并不准确. 其次, 在词向量模型训练过程中, 一个词的词向量只受训练文本中以该词为中心的固定窗口内的上下文词语的影响, 造成一个词与窗口外的词语相似度较低. 因此一些词义上相似度较高的词对由于不经常在同一窗口内出现, 使得基于词向量计算的语义相似度和词语之间真实的语义相似度之间存在着误差, 这种情况我们称之为非邻域词. 例如 "旅行" 和 "宾馆" 两个词语, 通过 Word2vec 算法[2−3] 在搜狗新闻语料库[1] 训练的连续词带模型 (Continuous bag-of-words model, CBOW) 词向量计算的语义相似度 (范围 0∼1) 仅为 0.003, 几乎不相似, 显然和人的主观判断不一致. 另外, 词向量既无法像本体那样通过等价关系来表示同义词关系, 也无法通过语义词典中同义词分组的方式来表达同义词, 单纯利用词向量的距离不能准确计算同义词的相似度. 例如 "西红柿" 和 "番茄" 两个词属于同义词, 理论上语义相似度应为 1. 在词向量空间中, 两个词语分别对应空间中的两个点, 而且 "西红柿" 在词向量空间中和 "黄瓜" 的距离相对 "番茄" 更近, 造成 "西红柿 – 黄瓜" 的语义相似度大于 "西红柿 – 番茄" 的语义相似度. 显然, 由于词向量无法表达同义词关系, 导致了上述语义相似度的计算结果缺乏准确性.

针对上述问题, 本文提出了一种基于词义向量模型的词语语义相似度算法 (An algorithm of semantic similarity between words based on word single-meaning embedding model, WSME). 本文的词义向量模型和现有的词向量模型最大的区别是: 词义向量模型中的每个向量对应的词语只表达唯一的词义, 多义词按不同词义被分成多个词, 并利用同义词词林 (Tongyici Cilin, TC) 的词义编码作为前缀进行标识, 每个词义的词分别对应唯一的向量. 基于词义向量模型, 可以计算多义词不同词义之间的相似度, 避免了词义的混淆. 例如, 对于多义词 "仪表", 在词义向量模型中分为 "Dc04A01 = 仪表" 和 "Bo18A01 = 仪表" 两个词, 分别由唯一的词义向量表示, 词语前的标识为同义词词林中的词义编码. "Dc04A01 = 仪表" 表示人的外表, "Bo18A01 = 仪表" 表示测量仪器. 当计算 "仪表" 和其他词语之间的相似度时, 就可以分别利用 "仪表" 的两个词义向量进行更为精确的计算. 此外根据词义向量模型中词语的标识, 可以进一步利用同义词词林中先验的词义分类信息和同义词分组信息, 对不同词义的词语进行同义词扩展, 借助同义词信息校正非邻域词的相似度计算误差, 以及直接判断两个词语是否为同义词关系, 从而进一步提升非邻域词和同义词的语义相似度计算精度. 例如, 非邻域词 "旅行" 和 "宾馆", 在词义向量模型中分别对应 "Hf04A01 = 旅行" 和 "Dm04A12 = 宾馆" 两个词语, 根据词语的标识, 可以从同义词词林中获取两个词语的同义词集合. "旅行" 的同义词包含 "行旅" 和 "远足", "宾馆" 的同义词包含 "旅馆"、"旅店" 和 "旅社" 等词. 虽然 "旅行" 和 "宾馆" 之间的语义相似度较小, 但可以通过 "旅行" 和 "旅店" 以及 "旅行" 与 "旅馆" 等词语之间的语义相似度对其进行修正, 弥补语义关系的缺失. 同义词 "西红柿" 和 "番茄" 在词义向量模型中对应 "Bh06A32 = 番茄" 和 "Bh06A32 = 西红柿" 两个词, 它们的词义标识使用相同的同义词词林编码, 据此可以判断两个词语为同义词, 语义相似度为 1, 有效弥补了词向量模型无法表示词语之间同义词关系的缺陷. 经过上述处理, 多义词、非邻域词和同义词的语义相似度计算准确性能够得到有效提升.

本文组织结构如下: 第 1 节介绍了词语语义相似度算法的相关工作; 第 2 节描述了本文提出的词义向量模型的构建过程; 第 3 节阐述了基于词义向量的词语相似度算法; 第 4 节描述了对比实验并对实验结果进行了分析; 第 5 节对本文工作进行总结, 并展望未来的工作.

1 相关工作

目前, 词语的语义相似度计算方法可归纳为三类: 基于本体的语义相似度计算方法、基于语义词典的语义相似度计算方法和基于词向量的语义相似度计算方法.

1.1 基于本体的语义相似度计算方法

基于本体的词语语义相似度计算方法利用了本体的结构特征, 每个词语分别作为本体模型中的一个节点, 根据词语在本体中的位置和节点间的路径长度衡量相似度的大小. 常用的基于本体模型的词语语义相似度计算方法包括:

1) 基于语义距离的方法: 通过定义两个词语在本体模型中的语义距离来衡量词语的语义相似度. 语义相似度与语义距离成反比, 两个词语的语义距离越大, 相似度越小. 语义距离可以通过节点之间的最短路径长度、深度或者是两者的综合来衡量[4].

2) 基于信息量的方法: 信息量表示一个节点包含信息的多少. 如果两个词语共享的信息越多, 它们之间的语义相似度也就越大. 信息量有多种表示方法, Meng 等[5] 利用概念和子概念在本体中的深度计算信息量, Seddiqui 等[6] 利用与概念相关的属性关系数量计算信息量, Sánchez 等[7] 同时使用概念的叶子节点和祖先节点数量来计算信息量.

3) 基于特征的方法: 词语之间的语义相似度也

[1] SogouCS: http://www.sogou.com/labs/resource/cs.php

可以借鉴人工智能领域的物体识别方法, 通过定义词语的特征, 利用特征的接近程度衡量相似度的大小. Sánchez 等[8] 利用概念节点到根节点经过的路径作为特征, 两个特征中包含的公共节点数量越多, 概念的语义相似度就越大. Zadeh 等[9] 将本体概念之间的连接关系以及两个概念与中间概念之间的连接关系作为特征, 在此基础上采用模糊集合理论 (Fuzzy set theory) 计算两个概念之间的语义相似度.

4) 混合型方法: 同时考虑了词语在本体中的距离、信息量以及特征等因素, 选择其中的若干方法进行组合来评判词语的相似度. 李文清等[1] 通过对基于信息量计算的相似度和最短距离计算的相似度进行线性加权求和作为概念之间的相似度. Li 等[10] 综合利用本体概念深度、局部密度、分类特征和复合成分特征等计算词语的语义相似度.

基于本体的方法充分利用了本体内部结构信息, 但该方法严重依赖于本体模型的质量, 目前本体仍以领域专家手动创建为主, 缺乏一定客观性. 此外本体大都针对特定领域, 对于领域外的词语缺少相应的描述, 因此存在词汇量不足的问题.

1.2 基于语义词典的语义相似度计算方法

基于语义词典的语义相似度计算方法利用语义词典中的分层结构以及同义词和近义词信息来计算词语之间的语义相似度. 常用的英文语义词典包括普林斯顿大学的 WordNet[2]、加州伯克利大学的 FrameNet[3] 和维基百科的姐妹工程 Wiktionary[4] 等, 常用的中文语义词典包括 "知网" (HowNet)[5]、同义词词林和台湾大学的中文 WordNet[6] 等. 语义词典具有和本体类似的结构, 因此当不使用本体概念之间的属性关系时, 一些本体模型的语义相似度计算方法也可以应用到语义词典中. Meng 等[5] 利用概念节点在 WordNet 中的信息量计算词语相似度. Gao 等[11] 提出了一种基于信息量和最短路径的非线性组合的相似度计算方法. 此外, 一些方法从语义词典自身的特点出发来计算词语之间的相似度. 田久乐等[12] 利用了哈工大《同义词词林扩展版》, 首先获取每个词的不同词义所对应的同义词词林词义编码, 然后根据编码之间的语义距离计算不同词义的相似度, 最后取最大值作为词语的相似度.

基于语义词典的方法在词汇量上优于本体模型, 收录了常见领域的大部分词语. 但语义词典在组织形式上, 只存在上下位关系, 词语之间的关系没有本体丰富. 另外本体模型和语义词典两种方法均未能充分利用两个词语在文本中的同现频率信息及所处上下文的语义相似信息, 造成很多词语相似度的计算缺乏精度. 例如田久乐等[12] 的方法, 对于 "学生" 和 "学校" 两个词语, 由于分属同义词词林中的两个大类, 按照文中方法计算的语义相似度计算结果为 0, 显然与人的主观判断不符.

1.3 基于词向量的语义相似度计算方法

1.3.1 词向量模型

随着机器学习和神经网络技术的不断发展, 基于神经网络语言模型自动学习词向量的方法相继被提出. 词向量是词语的特征表示, 词语之间的语义相似度可以由向量之间的余弦距离计算. 词向量的表示方法主要分为独热表示 (One-hot representation) 和分布式表示 (Distributed representation) 两种. One-hot 用一个很长的向量来表示一个词, 向量的长度为词典的大小, 向量的分量只有一个 1, 其他全为 0, 1 的位置对应该词在词典中的位置. 但这种词向量表示容易受维数灾难的困扰, 并且不能很好地刻画词语之间的相似度. 分布式表达方式通过训练神经网络模型将每一个词映射成一个固定长度的 n 维向量, 将所有向量放在一起形成一个词向量空间, 而每个向量则为该空间中的一个点, 因此可以根据词之间的距离来判断它们之间的相似度. 典型的分布式的词向量模型有 Word2vec[2−3]、GloVe[13] 和 CVG[14] 等, 这些词向量表示模型具有良好的语义特性, 是表示词语特征的常用方式, 但向量中每一维的信息不如 One-hot 明确, 更强调整体上的语义信息.

1.3.2 字符信息加强的词向量模型

目前基于词向量模型的研究中, 研究人员相继提出了一些改进算法来进一步丰富词向量表达的语义信息, 以提高基于词向量的语义相似度计算精度. Chen 等[15] 针对中文词向量的表达, 提出了一种字符信息加强的词向量模型 (Character enhanced word embedding model, CWE). CWE 提出了字符向量的概念, 每个中文字符和中文词语一样被映射为一个固定长度的向量, 字符向量的维度和词向量的维度相等. CWE 模型在训练前, 将语料库中每个中文词语分解成多个中文字符, 然后同时学习字符向量和词向量, 最终每个词语的词向量由训练的词向量加上字符向量得到. 以 "仪表" 一词为例, "仪表" 最终的词向量是 "仪" 和 "表" 的字符向量以及训练得到的 "仪表" 的词向量的叠加. 该方法能够借助字符向量信息增强低频词的词向量表达, 使得训练的词向量具有更丰富的语义信息. 但是目前常用的汉字数量只有三四千个, 远没有词语数量丰富.

[2] http://wordnet.princeton.edu/
[3] http://framenet.icsi.berkeley.edu/fndrupal/
[4] http://www.wiktionary.org/
[5] http://www.keenage.com/
[6] http://lope.linguistics.ntu.edu.tw/cwn/

而且字符的歧义性相对词语更为严重, 经过字符信息增强的词向量并不能准确表达每个词语不同的词义. 和 CWE 类似, Bojanowski 等[16−17] 提出一种将 N 元模型 (N-gram) 与 Word2vec 的跳字模型 (Continuous skip-gram model, Skip-gram) 结合的词向量表示方法 FastText. FastText 将每个词语分解为多个 N-gram 字符串, 并同时训练 N-gram 向量和词向量, 之后将词向量和出现在该词中的所有 N-gram 对应的向量求和作为最终的词向量. 以 2-gram 的 FastText 模型为例, “book” 的词向量是 2-gram 集合 {“bo”, “oo”, “ok”} 中所有 2-gram 对应的向量和 Skip-gram 模型中 “book” 的词向量的叠加. 对于中文而言, 1-gram 的 FastText 模型就转换为了 CWE 模型. 相对 CWE, FastText 支持中英文在内的多种语言, 具有更好的通用性, 但仍然无法表达词语的不同词义.

1.3.3 义原信息加强的词向量模型

除了字符信息加强的方式外, Niu 等[18] 将义原信息 (Sememes) 考虑到词向量的学习任务中, 提出了一种义原编码词语表示学习模型 (Sememe-encoded word representation learning model, SE-WRL), 认为一个词的词义可以由 HowNet 中多个义原信息的组合来表达. 在 HowNet 中一个词语包含多个词义, 每个词义又由多个义原组成, 义原信息是词义的最小语义单位. SE-WRL 按照义原的使用策略分为单义原聚集模型 (Simple sememe aggregation model, SSA)、基于目标词的义原注意力模型 (Sememe attention over target model, SAC) 和基于上下文的义原注意力模型 (Sememe attention over context model, SAT) 三种模型. SSA 使用所有义原向量的均值来表示一个词向量; SAC 根据中心词来对每个上下文词语做消歧, 使用注意力机制的方法来计算这个词语的各个词义的权重, 使用词义向量的加权平均值来表示上下文词向量; SAT 和 SAC 模型正好是互相对称, 使用上下文词语预测中心词的含义, 对中心词消歧, 从而选择出符合情境的词义信息, 并使用词义向量的加权平均值来表示中心词的向量. SE-WRL 通过将义原信息融入到词向量, 提升了词向量的表示能力, 能够更好地发现近义词间的语义相似性. 但是 SE-WRL 的三个子模型仍采用了一词一向量的表达方式, 没有实现对词向量按词义划分, 依旧存在词义混淆的问题. 而且由于模型训练过程需要不断地更新义原向量、词向量以及进行词义消歧处理, 算法耗时非常严重.

1.3.4 多元词向量模型

基于词向量的方法根据语料库中每个词语的上下文信息来训练词向量模型, 从而得到了整个语料库所有词的词向量. 这里上下文信息指的是词语在语料库中每个出现位置与其左右相邻的词语集合. 当语料库规模较大时, 包含的词汇量可以超过语义词典的词汇数量. 词向量包含了词语的语义信息, 对于经常在同一上下文中出现的词语之间的语义相似度有较好的计算精度. 但是在上述词向量模型中, 训练的词向量均为词语多个词义的混合. 因此当涉及到多义词之间的语义相似度计算时, 基于词向量的方法均无法提供令人满意的结果. 对此, 一些研究人员针对多义词的影响, 提出了多元词向量模型 (Multi-prototype word embedding model), 对于每个词语训练多个向量. 取两个词语对应的多个向量之间余弦距离的最大值或均值作为词语的语义相似度. Huang 等[19] 利用球面 k 均值聚类算法对语料词典中的每个词语的上下文进行聚类, 为每个词语分配 k 个不同词义, 对语料库中词语使用类似 “bank_1” 和 “bank_2” 方式进行标注, 经过训练得到多元的词向量模型. 该方法尽管按词义对词向量进行了区分, 但是对所有词语分配了相同的词义数量, 忽视了词语词义数量不同的事实, 因此这种多元词向量模型并不能准确表示词语的词义. Guo 等[20] 基于双语对应数据来判断词语的词义, 进而利用递归神经网络训练多元词向量模型. 以中英双语数据为例, 对于一个中文词 “制服”, 从对应的英文数据集中获取其翻译的英文词如 “subdue”、“uniform”、“vestment” 等词语, 然后以这些英文词语的词向量作为特征通过相似传播聚类 (Affinity propagation, AP) 方法进行聚类, 得到词语的类别数量和聚类信息. 该方法尽管为不同词语分配了不等的词义数量, 但在确定词义数量时, 使用了英文词语对应的词向量作为特征, 而这些英文词很多本身也具有多个词义, 使得 AP 聚类算法的准确度受到负面影响, 造成多义词词义数量确定的误差, 相应也会降低基于多元词向量模型的词语间相似度计算精度.

1.3.5 本文算法的创新

我们认为基于词向量的词语语义相似度算法精度的提升需要以词向量对于词语词义的准确表达为前提. 仍然以多义词 “仪表” 为例, 当其上下文为 “误差分析是提高仪表精确度的前提” 时, “仪表” 表示的是测量仪器; 当其上下文为 “优雅的仪表能够增加人的自信心” 时, “仪表” 表示人的外表. “仪表” 的两个词义必须通过不同的向量才能够表达, 对此本文提出了词义向量模型, 利用每个向量表达词语的一个词义, 从而可以计算词语不同词义之间的语义相似度, 这是现有的词向量模型无法完成的. 本文的词义向量模型本质上也是一种多元词向量模型,

但是和 Huang 等[19] 的模型以及 Guo 等[20] 的模型存在如下区别: 首先本文词义向量模型在确定多义词的词义数量时利用了同义词词林先验的词义分类信息, 因此相比聚类算法能够更准确地获得多义词的词义数量; 其次词义向量模型对于多义词的不同词义采用同义词词林编码进行明确的标识, 可以利用每个词义编码下的同义词信息进一步提升词语语义相似度的计算精度, 而不是简单将多元词向量之间的相似度取均值或取最大值作为两个词语之间的相似度. SE-WRL[18] 算法虽然也采用了基于知识库 (HowNet) 的词义区分策略, 但是其目的是将义原信息嵌入到词向量中, 使得词向量包含更多的语义信息, 仍然是一种一对一的词向量模型. 本文基于词义向量的算法则是利用知识库的词义分类信息为多义词的每个词义训练词义向量模型, 在计算词语相似度时, 根据词义标识综合利用了词义向量和各词义编码下的同义词信息进行计算精度的提升, 因此从向量表示方式和相似度计算方法上和 SE-WRL 算法都有所区别.

2 词义向量模型的构建

由于现有词向量模型的每个多义词只用一个词向量来表达, 没有对多义词的不同词义进行区分, 词向量表达的信息并不能和词义一一对应, 使得基于词向量计算的词语语义相似度的准确性不高. 针对此缺陷, 本文提出了词义向量模型, 模型中每个多义词按照词义数量被分为多个单义词语, 每个词义的词语分别对应唯一的向量.

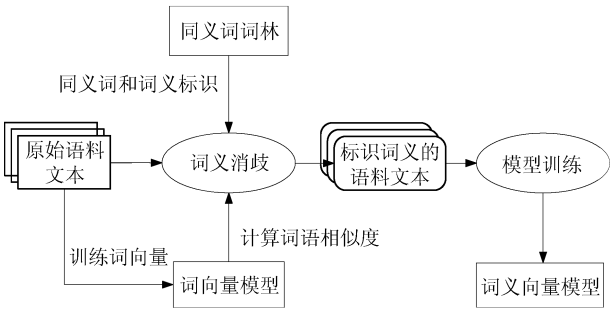


图 1 词义向量模型的构建流程

Fig. 1 The build process of word single-meaning embeddings

词义向量模型的构建过程如图 1 所示, 包含词义消歧和模型训练两个部分. 首先根据词语的上下文信息和同义词词林对语料库中的词语进行词义消歧, 每个词语根据当前上下文采用不同的词义编码进行标识, 实现词义的明确; 然后基于标识词义后的语料库利用词向量算法训练词义向量模型. 在词义消歧过程, 同义词词林提供了先验的词义分类信息, 帮助确定每个多义词的词义数量和对应的词义标识. 从原始语料文本中训练的词向量模型, 用于衡量当前词语的上下文和同义词词林中各词义编码下的词语之间的语义距离, 确定当前词语所表达的词义. 词义向量模型的训练过程其实就是以标识词义的语料文本作为训练集训练词向量的过程, 该过程可以直接使用现有的比较成熟的词向量算法 (如 Word2vec) 来完成. 训练得到的模型中每个词均是单义词, 原多义词的每个词义均通过唯一的向量表示, 所以这个模型是一种词义向量模型.

2.1 Word2vec 词向量模型与同义词词林介绍

2.1.1 Word2vec 词向量模型

Word2vec 是 Google 于 2013 年推出的一个用于训练词向量的开源算法, 核心思想是基于神经网络概率语言模型, 根据词语在语料库中的上下文信息, 为每个词语训练一个相同维数的实向量. Word2vec 包含两种训练模型: CBOW 模型和 Skip-gram 模型, 均包含输入层、投影层和输出层. 不同之处在于, CBOW 模型是通过上下文词语来预测当前词, Skip-gram 模型则是通过当前词来预测其上下文词语. 这里以 CBOW 为例, 介绍 Word2vec 的原理. 通常, 给定一个词序 $C = \{w_1, w_2, \cdots, w_m\}$, CBOW 模型的目标函数如下:

$$L(C) = \frac{1}{m}\sum_{i=t}^{m-t} \log P(w_i|w_{i-t}, \cdots, w_{i+t}) \tag{1}$$

其中, w_i 为某个中心词, t 为中心词 w_i 左右窗口的大小, $P(w_i|w_{i-t}, \cdots, w_{i+t})$ 表示已知 w_i 的上下文中心词为 w_i 的概率, 通过 softmax 回归函数计算:

$$P(w_i|w_{i-t}, \cdots, w_{i+t}) = \frac{\exp(\boldsymbol{w}_0^{\mathrm{T}}\boldsymbol{w}_i)}{\sum\limits_{w\in W}\exp(\boldsymbol{w}_0^{\mathrm{T}}\boldsymbol{w})} \tag{2}$$

其中 W 是词典库, $\boldsymbol{w}_i$ 是中心词 w_i 的词向量表示, $\boldsymbol{w}_0$ 是 w_i 的上下文词语的词向量的均值, 公式为:

$$\boldsymbol{w}_0 = \frac{1}{2t}\sum_{j=i-t,\cdots,i+t,j\neq i} \boldsymbol{w}_j \tag{3}$$

Word2vec 采用随机梯度上升法将式 (1) 中的目标函数最大化, 经过整个语料库词语的训练, 最终得到词典库中每个词对应的词向量.

2.1.2 同义词词林

同义词词林是一个包含词语的所有词义解释, 以及不同词义下的同义词和近义词的语义词典, 使用统一的 8 位编码格式来标识不同的词义 (表 1), 从上至下共有 5 级结构. 其中第 1 级大类用大写英文字母表示; 第 2 级中类用小写英文字母表示; 第 3 级

小类用二位十进制整数表示; 第 4 级词群用大写英文字母表示; 第 5 级原子词群用二位十进制整数表示. 第 8 位的标记有 3 种, “ = ” 表示同一编码下的词语为同义词, “#” 表示相同编码下的词语为词义不相等但属于同一类的近义词; “@” 表示编码下的词语具有独占的词义, 没有同义词和近义词.

表 1　同义词词林的编码格式

Table 1　The coding format of the Tongyici Cilin

位数	1	2	3	4	5	6	7	8
符号	D	a	1	5	B	0	2	= \ # \ @
性质	大类	中类	小类		词群	原子词群		
层级	第 1 层	第 2 层	第 3 层		第 4 层	第 5 层		

2.2　基于词向量和同义词词林的词义消歧

目前主流的词义消歧方法分为三类: 有监督的词义消歧方法、无监督的词义消歧方法和基于知识库 (语义词典) 的词义消歧方法. 有监督的词义消歧方法基于对不同上下文词语的词义进行了人工标注的文本训练集, 利用机器学习的方法 (决策树、支持向量机 (Support rector machine, SVM)、人工神经网络等) 训练分类器, 通过分类器识别新文本中歧义词的词义. 该方法具有较高的准确率, 但识别精度受限于标注语料库的规模和质量, 难以应用于大规模词义消歧任务. 无监督的词义消歧方法主要利用聚类算法对歧义词出现的所有上下文词语集合进行聚类, 如 Huang 等[19] 的模型和 Guo 等[20] 的模型均采用了此类方法进行消歧. 该类方法无法对词语的词义进行明确标注, 只能标识词义归属的类别, 而且受聚类算法精度的影响, 无法准确确定多义词的词义数量. 基于知识库的方法根据歧义词所处的上下文, 利用同义词词林、HowNet 和 WordNet 等由语言学家编纂的知识库中先验的词义解释或词义分类信息来判断当前歧义词最可能的词义. SE-WRL[18] 算法中 SAC 和 SAT 模型训练时对于中心词或上下文词语的词义消歧采用了基于 HowNet 知识库和词向量的方法, 属于此类方法. 在以上三类方法中, 基于知识库的方法是目前唯一能真正用于大规模词义消歧任务的方法[21].

本文词义向量模型的训练需要基于吉字节 (Gigabyte, GB) 级的大规模的语料库 (如搜狗新闻语料库), 语料库中包含多达几十万条的文本及更大规模的语句. 尽管有监督的词义消歧方法具有更高的词义消歧精度, 但由于尚未有如此大规模词义标注过的中文语料库, 无法训练适合的词义消歧模型, 因此不能完成本文词义向量构建过程中的词义消歧任务. 而无监督的词义消歧方法由于存在词语类别划分误差, 并且每个词语均需进行聚类操作, 对于大规模语料耗时严重. 综上分析, 本文选择基于知识库的词义消歧方法对原始语料文本进行词义消歧处理. 知识库选择同义词词林, 利用同义词词林编码对多义词进行词义标识.

2.2.1　函数定义

在介绍词义消歧算法之前, 首先定义和本文算法相关的函数.

函数 1. 获取词 w 在同义词词林中的词义编码集合:

$$C = findCode(w) = \{c_1, c_2, \cdots, c_n\} \tag{4}$$

n 为词 w 在同义词词林中的词义个数, 即在同义词词林中的编码数量, 表示为 $n = |findCode(w)|$.

函数 2. 获取词 w 在同义词词林编码 c 下的所有同义词:

$$W = findSyn(w, c) = \{w_1, w_2, \cdots, w_m\} \tag{5}$$

通过同义词词林编码第 8 位符号, 可判断同一编码下的词语是否为同义词. 因此当 c 的末位为 “=” 时, 返回编码 c 下除 w 以外的词语集合; 当 c 的末位为 “#” 或 “@” 时, 编码下不包含同义词, 只返回词语本身. m 为词林编码 c 中与 w 为同义关系的词语数量, 表示为 $m = |findSyn(w, c)|$.

函数 3. 编码和词语组合函数:

$$group(c, w) = cw \tag{6}$$

$group(c, w)$ 函数实现了同义词词林编码和词语的字符串连接, 组合后的词只表达编码为 c 的词义, 实现了原多义词 w 在不同词义下的标识. 如词语为 “仪表”, 编码为 “Dc04A01 = ”, 组合后为 “Dc04A01 = 仪表”, 表达 “人的外表” 的词义.

2.2.2　词义消歧原理

词义消歧的实质就是对文本中的每个多义词的词义进行明确, 包括确定一个词语的词义数量以及识别在不同上下文下表达的真正词义两个步骤. 由于同义词词林已经包含了每个词语对应的词义数量和各词义的编码, 因此通过同义词词林可以准确地完成词义消歧的第一步, 在此基础上利用同义词词林编码作为前缀可以对多义词的各词义进行明确区分. 为了确定不同上下文下每个词语表达的词义, 本文首先对词语当前上下文词语的词向量进行加权组合得到上下文向量; 然后根据词语的每个词义编码获取同义词词林中的同义词集合, 对同义词的词向量加权叠加作为该词义的编码向量; 最后比较上下文向量与每个编码向量的余弦距离, 其中余弦距离最大的编码向量对应的词义编码就是当前多义词的词义. 词义消歧算法的实现步骤如下:

算法 1. 取词 w 的上下文词语集合.

对语料库文本采用滑动窗口的方式获取中心词 w 的上下文词语集合, 表示为 $Context(w) = \{\tilde{w}_1, \tilde{w}_2, \cdots, \tilde{w}_k\}$, k 为上下文窗口大小.

算法 2. 计算 $Context(w)$ 中每个词的权重.

本文认为上下文词语对于中心词的影响程度与该词和中心词的距离以及上下文词语的词义数量有关. 上下文词距离中心词越近, 影响程度越大; 上下文词语的词义数量越少, 说明该上下文表达词义更明确, 影响程度越大. 上下文中的词 $\tilde{w}_i$ 与中心词 w 的距离表示为 L_i, 为 $\tilde{w}_i$ 与 w 之间相隔词语的个数, 包括 $\tilde{w}_i$ 词语本身. 上下文词语集合中每个词 $\tilde{w}_i$ 所对应的权重 v_i 为 $\tilde{w}_i$ 与 w 之间距离 L_i 与 $\tilde{w}_i$ 词义数量乘积的倒数.

$$v_i = \frac{1}{L_i \times |findCode(\tilde{w}_i)|} \tag{7}$$

邻域词集合 $Context(w)$ 所对应的权重集合表示为 $V = \{v_1, v_2, \cdots, v_k\}$.

算法 3. 计算上下文向量.

中心词 w 在不同上下文中表达的词义不同, 但 w 的词义与 $Context(w)$ 整体表达的语义是一致的, 因此需要先确定 $Context(w)$ 表达的语义, 才能进一步判断 w 在出现位置的特定词义. 这里通过将 $Context(w)$ 中词语的词向量加权求和作为上下文的语义表达. 以 $ContextVector(w)$ 表示上下文词语集合 $Context(w)$ 中词语的词向量集合, 即 $ContextVector(w) = \{\tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2, \cdots, \tilde{\boldsymbol{w}}_k\}$, 其中 $\tilde{\boldsymbol{w}}_i \in \mathbf{R}^m$ 为词语 $\tilde{w}_i$ 对应的词向量, m 为词向量的维度. $ContextVector(w)$ 与权重集合 V 进行加权求和得到上下文向量 $\boldsymbol{Context}_w$, 计算公式为:

$$\boldsymbol{Context}_w = \sum_{i=1}^{k} v_i \times \tilde{\boldsymbol{w}}_i \tag{8}$$

算法 4. 根据同义词词林的词义分类, 计算中心词 w 的编码向量集合 E.

同义词词林中已经创建了词语的词义分类, 每个编码代表一个特定的词义, 利用该先验信息, 可以确定一个词语具有几种词义. 本文用属于同一编码下的所有词语的词向量的加权和来表示该编码所对应的语义, 即编码向量. 在计算编码向量时, 我们认为一个词语的词义数量越少, 表达的词义越纯正, 该词语对于当前编码词义的贡献越多, 权值相应也越大. 因此, 同一编码下每个词语的权值为该词语包含的词义数量的倒数. 中心词 w 在同义词词林的编码集合为:

$$C = \{c | c \in findCode(w)\} \tag{9}$$

$m_w = |C|$ 表示词 w 的词义个数, 编码 c_j 所对应的编码向量 $\boldsymbol{e}_j$ 为:

$$\boldsymbol{e}_j = \sum_{i=1}^{n_j} \boldsymbol{w}_j^i \times \frac{1}{|findCode(w_j^i)|} \tag{10}$$

其中 $w_j^i \in findSyn(w, c_j)$, 表示 w 在同义词词林编码 c_j 下的同义词, $n_j = |findSyn(w, c_j)|$ 为同义词的个数, $\boldsymbol{w}_j^i$ 为 w_j^i 对应的词向量. w 的编码向量集合为 $E = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_{m_w}\}$.

算法 5. 确定词语 w 在当前上下文的词义编码.

在得到 w 的上下文词向量和所有编码向量集合后, 根据 Chen 等[22] 提出的多义词上下文向量与其真实词义向量相似度更高的思想, 通过计算上下文向量 $\boldsymbol{Context}_w$ 与各编码向量的余弦相似度, 最大相似度对应的编码向量 $\boldsymbol{e}$ 就是当前 w 表达的词义, 通过 $\boldsymbol{e}$ 进而得到词义编码 c.

$$\boldsymbol{e} = \underset{\forall \boldsymbol{e}_j \in E, 1 \le j \le m_w}{\arg\max} \frac{\boldsymbol{e}_j^{\mathrm{T}} \boldsymbol{Context}_w}{|\boldsymbol{e}_j| \times |\boldsymbol{Context}_w|} \tag{11}$$

算法 6. 对词语 w 进行词义标识. w 在 $Context(w)$ 的词义被确定后, 将词义编码 c 与 w 的组合, 记为 $group(c, w)$, 编码 c 作为 w 的前缀, 使得 w 在不同上下文能够表示明确的词义.

2.2.3 词义消歧结果

经过上述词义消歧过程, 不同上下文中的多义词根据其确定的词义被替换成由同义词词林编码标识的词语. 通过同义词词林判断为具有单一词义的词语在消歧后的文本中同样添加了词林编码作为前缀, 方便后续语义相似度计算过程使用该编码下的同义词集合. 另外, 语料库中还存在一些没有被同义词词林包含的词语, 无法通过同义词词林进行词义数量判断和标识, 这些词语本文默认为单义词, 并且不进行词义的标识. 以 “仪表” 这个多义词为例, 在同义词词林中, 具有 “Dc04A01 = ” 和 “Bo18A01 = ” 两个词义编码, 分别表示 “人的外表” 和 “测量仪器”. 经过词义消歧处理, 语料库中的 “仪表” 根据上下文被替换为 “Dc04A01 = 仪表” 或 “Bo18A01 = 仪表”. “温度计” 这个词在同义词词林中包含唯一的词义编码 “Bo18A05 =” , 在消歧后的语料被替换为 “Bo18A05 = 温度计”. “控制系统” 这个词没有包含在同义词词林中, 对于这种情况的词语, 本文设定为单义词, 在语料库中保持原型.

2.3 词义向量模型训练

2.3.1 模型训练方法

本文词义模型的训练直接采用了 Word2vec 词向量算法的经典三层神经网络, 架构如图 2 所示. Word2vc 算法的原理已在第 3.1.1 节介绍, 输入层

为上下文中各词的词向量, 投影层将输入层的词向量累加求和, 输出层为中心词的词向量. Word2vc 将窗口视作训练单位, 通过滑动窗口的方式产生训练集. 在每个训练过程, 窗口内词语都要进行一次参数更新. 为了加快训练过程, 可以采用随机负采样算法 (Negative sampling) 减少参数更新的数量. 也可以用霍夫曼树来代替从隐藏层到输出层的映射, 并利用层次 softmax 回归的方法训练树中的节点参数和词向量信息.

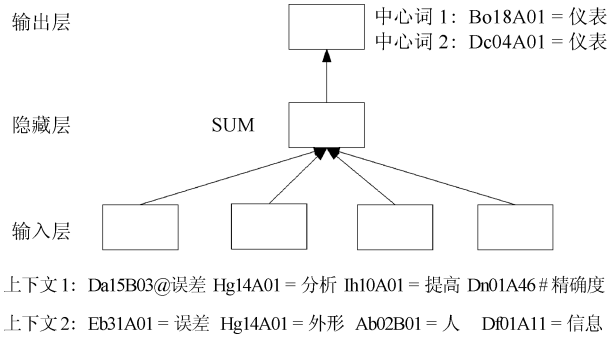


图 2　训练词义向量的神经网络结构

Fig. 2　The architecture of neural network to learn word single-meaning embeddings

与训练原始词向量不同, 经过词义消歧后的训练语料中每个多义词被替换为多个不同编码标识的单义词, 因此每个多义词会对应多个向量的训练过程. 例如图 2 中训练原始词向量时, 上下文 1 和上下文 2 中的词语均用来作为神经网络模型的输入来更新 “仪表” 一词的向量表达. 而词义向量的训练过程, 由于上下文 1 和上下文 2 对应于 “仪表” 的不同词义, 因此上下文 1 中的词语只作为 “Bo18A01 = 仪表” 训练的输入, 而上下文 2 中的词语则作为 “Dc04A01 = 仪表” 训练的输入, 两者在训练过程中是不重叠的. 同样 “仪表” 的两个词义向量训练过程中也分别迭代更新上下文 1 和上下文 2 中的词义向量.

经过上述训练过程, 得到了与词语的词义一一对应的词义向量模型. 需要指出的是, 词义向量的构建流程采用一种松耦合的方式与词向量模型训练方法结合, 因此同样可以采用 Word2vec 以外的其他词向量技术基于消歧的文本训练词义向量, 但并不推荐选择 CWE、FastText 这一类需要进行词形分解的算法. 因为词义消歧之后的语料文本中很多词语带有词义前缀, 这些前缀必须和同义词词林结合才有明确的含义, 单纯将其分割为 N-gram 不会提升词向量的表达能力.

2.3.2　词义向量对词义表达的提升

词义向量模型实现了对词语词义的表达, 避免了原始词向量中多义词词义的混淆. 例如在词义向量模型中, “Dc04A01 = 仪表” 和 “Bo18A01 = 仪表” 作为 “仪表” 一词的两个词义分别对应一个词义向量. “Bo18A05 = 温度计” 作为 “温度计” 唯一的词义使用唯一的词义向量表示. “控制系统” 一词不在同义词词林中, 以原型形式对应唯一的向量. 我们通过在模型中查找与一个词的最近邻词集来进一步对比词向量模型和词义向量模型的区别. 表 2 为基于搜狗新闻语料库训练的 CBOW 词向量模型中与 “仪表” 最接近的 10 个词语, 可以看出模型中 “仪表” 的词向量基本表现的是 “测量仪器” 的词义, 而在一定程度上缺失了 “人的外表” 的词义. 这是由于在语料中, 表达 “测量仪器” 的上下文文本相对 “人的外表” 含义的上下文文本更多, 因而向量空间中, “仪表” 更接近表达 “测量仪器” 词义的词语. 表 3 为基于相同的语料库训练的词义向量模型中与 “仪表” 一词的两个词义最相似的 10 个词, 和表 2 相比可以看出词义向

表 2　CBOW 词向量模型中与 “仪表” 最相似的 10 个词

Table 2　Top 10 most similar words to the polyseme in the CBOW word embedding model

仪表	相似度
压力表	0.671
控制系统	0.666
电子设备	0.655
控制技术	0.650
电子式	0.647
液压	0.641
主动式	0.639
飞控	0.638
机械式	0.637
仪表板	0.635

表 3　词义向量模型中与 “仪表” 两个词义最相似的 10 个词

Table 3　Top 10 most similar words to the different meanings of the polyseme in WSME

Dc04A01 = 仪表	相似度	Bo18A01 = 仪表	相似度
风流倜傥	0.700	控制系统	0.697
De04B02 = 才情	0.679	电子设备	0.684
Ee31A01 = 儒雅	0.669	电子系统	0.674
貌美	0.667	Bo18A16 # 压力表	0.662
De04A04 = 才思	0.663	转速表	0.653
Ee31A01 = 雍容	0.662	Bo25B01 = 方向盘	0.652
Ee10B01 = 旷达	0.659	Bo18A17 # 高度计	0.652
Eb30B01 = 其貌不扬	0.659	Fa05B03 = 液压	0.650
Dk02B02 # 才学	0.653	Dc01C16 # 机械式	0.644
De04A02 = 天资	0.647	仪表板	0.643

量实现了很好的词义聚类. 在和 “Dc04A01 = 仪表” 相似的词语中, 包含了同义词词林没有收录的词 “风流倜傥”, 只有一个词义的单义词 “De04B02 = 才情”, 以及表达 “豪放” 词义的 “Ee10B01 = 旷达”. 同样, 和 “Bo18A01 = 仪表” 最相似的词语中也包含了 “控制系统” 这种不包含在词林中的词和 “Bo18A16 # 压力表” 这种单义词. 由此看出, 词义向量模型能够实现多义词词义的准确表达, 对于多义词的不同词义之间的相似度, 多义词的词义与单义词之间的相似度, 以及多义词词义与不涵盖在同义词词林中的词语的相似度都能准确地度量. 在此基础上, 利用同一编码下的同义词集可实现对多义词、非邻域词和同义词的相似度计算质量的进一步提升.

3 基于词义向量模型的语义相似度算法

3.1 算法动机

针对基于词向量的词语相似度算法在多义词、非邻域词和同义词三类情况下缺乏足够计算精度的缺陷, 本文从两个方向对此进行改进. 第一是利用本文提出的词义向量模型实现对词义的精确表达, 使得向量中包含的语义信息更为明确, 不再是多个词义的混合. 第二是提出了综合考虑词义向量和同义词集合的语义相似度算法. 目前在现有多元词向量模型[19−20] 中, 计算词语之间的相似度通常使用均值法 (AvgSim) 和最大值法 (MaxSim), 计算公式如下:

$$\text{AvgSim}\,(w_a, w_b) = \frac{1}{n_a \times n_b}\sum_{i=1}^{n_a}\sum_{j=1}^{n_b} sim(w_a^i, w_b^j) \tag{12}$$

$$\text{MaxSim}(w_a, w_b) = \max_{1\le i\le n_a, 1\le j\le n_b} sim(w_a^i, w_b^j) \tag{13}$$

其中 n_a 和 n_b 分别为词语 w_a 和 w_b 的词义数量, $sim(,)$ 为余弦相似度. AvgSim 方法取各词义之间的相似度均值作为词语之间的相似度, MaxSim 取词义之间相似度的最大值作为词语之间的相似度. Guo 等[20] 对两种方法在相同测试集下进行了对比, MaxSim 方法的计算精度 (55.4 %) 显著优于 AvgSim 方法 (49.3 %). AvgSim 方法实质上是对词语词义的一种折中, 和多元词向量模型的设计初衷相违背. 因此对于使用多元词向量模型计算词语的相似度, 最大值的方法更为适合.

现有的多元词向量模型尽管能够对词语的词义进行区分, 在一定程度上提升了多义词相似度的计算质量, 但对于非邻域词和同义词的相似度仍然不具备很好的计算精度. 这是因为非邻域词相似度的计算精度主要受训练集中词语的词频和词语之间的搭配习惯影响. 而且一些词对同时具备多义词、非邻域词和同义词的多个特征, 仅仅区分词义对于词语相似度的提升比较有限. 词义向量作为一种多元词向量模型, 在相似度的计算上也遵循 MaxSim 的基本思想. 但和现有的多元向量模型不同, 本文词义向量模型通过同义词词林编码标识了词语的不同词义, 根据标识信息在可以进一步获取词语每个词义的同义词信息对原 MaxSim 方法进行扩展, 从而形成扩展的 MaxSim 方法 (Extended MaxSim, ExMaxSim). 首先根据词林同一编码下的同义词信息, 我们可以准确判断两个词语是否为同义词关系, 这是通过向量之间距离无法完成的. 其次, 通过同义词词集合可以有效减少非邻域词造成的影响. 理论上如果 w_c 和 w_b 是同义词关系, w_a 和 w_b 的相似度应等价于 w_a 和 w_c 的相似度. 如果把将每个词义下的同义词也引入到词语相似度计算中, 可以有效补充非邻域词词义向量的语义信息. 同时为了避免矫枉过正, 在计算扩展的词义的相似度时, 至少包含 w_a 或 w_b 的一个词义向量. 最终词义相似度集合包含三个部分: 1) w_a 的所有词义和 w_b 的所有词义之间的相似度; 2) w_a 的所有词义和 w_b 的所有词义下的同义词之间的相似度; 3) w_b的所有词义和 w_a 的所有词义下的同义词之间的相似度. 其中 1) 即 MaxSim 方法中的词义相似度集合, 2) 和 3) 为通过同义词信息扩展的词义相似度集合. 在此基础上对 1) 、2) 和 3) 中的词义相似度取最大值作为 w_a 和 w_b 的相似度.

基于 ExMaxSim 的思想, 词语相似度的计算过程如图 3 所示. 首先基于同义词词林的词义标识信息对两个待比较的词语进行词义分解, 将其转化为带标识的单义词集. 经过词义分解之后, 可以从词义向量模型中获取每个词语的一个词义对应的词义向量, 以及计算词义之间的语义相似度. 例如 “仪表” 可分解为 “Dc04A01 = 仪表” 和 “Bo18A01 = 仪表”, 分别表示 “人的外表” 和 “测量仪器”. 然后利用同义词词林对分解后的词进行同义词扩展, 得到每个词义下的同义词集合. 例如, “Dc04A01 = 仪表” 的同义词集合包含 “Dc04A01 = 仪容”、“Dc04A01 = 仪态” 和 “Dc04A01 = 仪观” 等词; “Bo18A01 = 仪表” 的同义词集合包含 “Bo18A01 = 仪器”. 对于不包含在同义词词林中的词, 经过词义分解和同义词扩展后仍然是原词. 以上文提到的非邻域词对 “旅行” 和 “宾馆” 为例, 两个词均为单义词, 在词义向量模型中为 “Hf04A01 = 旅行” 和 “Dm04A12 = 宾馆”. 经过同义词扩展之后, 可以按照计

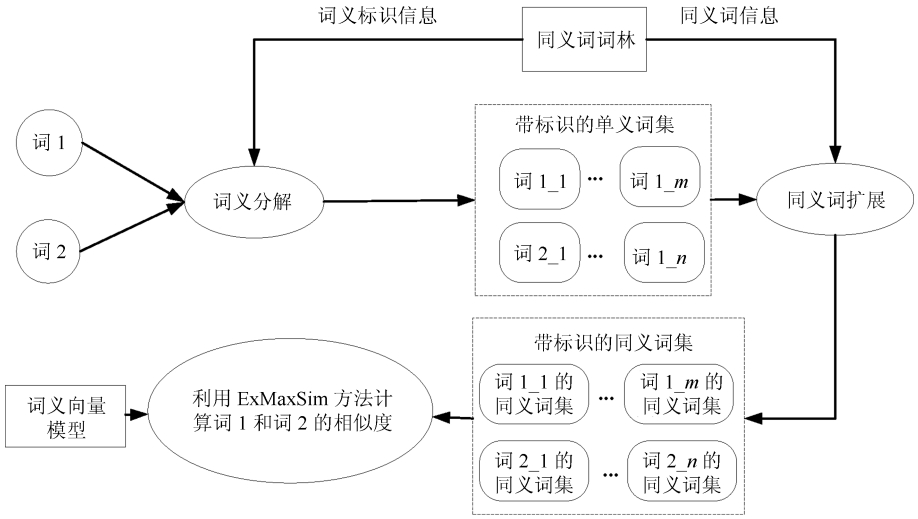


图 3　基于词义向量的词语语义相似度计算过程

Fig. 3　The computing process of similarity between words based on word single-meaning embedding model

算 ExMaxSim 方法三个部分的词义相似度集合: 1) 部分为 {sim(“Hf04A01 = 旅行”, “Dm04A12 = 宾馆”)}, 2) 部分为 {sim(“Hf04A01 = 旅行”, “Dm04A12 = 旅馆”), sim(“Hf04A01 = 旅行”, “Dm04A12 = 旅社”), sim(“Hf04A01 = 旅行”, “Dm04A12 旅店”)}, 3) 部分为 {sim(“Hf04A01 = 行旅”, “Dm04A12 = 宾馆”), sim(“Hf04A01 = 远足”, “Dm04A12 = 宾馆”)}. 通过取 1)、2) 和 3) 中的最大值作为 “旅行” 和 “宾馆” 的相似度, 有效弥补了非邻域词缺失的语义.

3.2　词语语义相似度计算过程

对于任意两个词 w_a 与 w_b, 计算语义相似度的具体步骤如下.

步骤 1. 获取词语 w_a 与 w_b 的同义词词林编码, 得到带标识的单义词集.

词 w_a 与 w_b 的同义词词林词义编码集合 C_a 与 C_b 可通过式 (4) 获取, 分别为:

$$C_a = findCode(w_a) = \{c_a^1, \cdots, c_a^{n_a}\} \tag{14}$$

$$C_b = findCode(w_b) = \{c_b^1, \cdots, c_b^{n_b}\} \tag{15}$$

其中 n_a 和 n_b 分别为词语 w_a 与 w_b 的词义编码个数. 将词语 w_a 和 w_b 与其对应的编码集合 C_a 与 C_b 中的编码按式 (6) 进行组合, 形成编码和词语的单义词集合 $groupA$ 与 $groupB$, 分别表示为:

$$groupA = \{group(c_a^1, w_a), \cdots, group(c_a^{n_a}, w_a)\} \tag{16}$$

$$groupB = \{group(c_b^1, w_b), \cdots, group(c_b^{n_b}, w_b)\} \tag{17}$$

步骤 2. 获取词语 w_a 与 w_b 的在同义词词林中不同词义的同义词集合.

词语 w_a 在编码 $c_a^i, \forall 1 \le i \le n_a$ 下的同义词集合和 w_b 在编码为 $c_b^j, \forall 1 \le j \le n_b$ 下的同义词集合可通过式 (5) 获得, 分别为:

$$Syn_a^i = findSyn(w_a,\ c_a^i) \tag{18}$$

$$Syn_b^j = findSyn(w_b,\ c_b^j) \tag{19}$$

步骤 3. 按照 ExMaxSim 方法计算 w_a 与 w_b 的相似度.

本文 ExMaxSim 相似度计算方法是取 1) w_a 的所有词义和 w_b 的所有词义之间的相似度, 2) w_a 的所有词义和 w_b 的所有词义下的同义词之间的相似度, 3) w_b 的所有词义和 w_a 的所有词义下的同义词之间的相似度三个部分的最大值.

集合 1) 可以通过计算 $groupA$ 和 $groupB$ 各元素之间的相似度得到. 对于 w_b 的同义词集合 Syn_b^j 中的任意一个词语 w', 与对应的编码 c_b^j 进行组合, 得到标识后的单义词语 $group(c_b^j, w')$. 将集合 $groupA$ 中的每个元素与每个 $group(c_b^j, w')$ 计算相似度可得到集合 2). 同理, 对于 w_a 的同义词集合 Syn_a^i 中的任意一个词语 w', 与对应的编码 c_a^i 进行组合, 得到标识后的单义词语, 记为 $group(c_a^i, w')$. 将集合 $groupB$ 中的每个元素与每个标识后的单义词计算相似度可得到集合 3). 最后, 我们取三者最大值作为 w_a 与 w_b 的相似度, 具体公式如下:

$$sim_1 = \max_{1 \le i \le n_a, 1 \le j \le n_b} sim(group(c_a^i, w_a), group(c_b^j, w_b)) \tag{20}$$

$$sim_2 = \max_{1\le i\le n_a, 1\le j\le n_b, w'\in Syn_b^j} sim(group(c_a^i, w_a), group(c_b^j, w')) \quad (21)$$

$$sim_3 = \max_{1\le i\le n_a, 1\le j\le n_b, w'\in Syn_a^i} sim(group(c_a^i, w'), group(c_b^j, w_b)) \quad (22)$$

$$sim\,(w_a, w_b) = \max\{sim_1 \cup sim_2 \cup sim_3\} \quad (23)$$

其中 $sim(,)$ 函数用来计算两个标识后的词语对应的词义向量之间的余弦距离.

3.3 算法合理性分析

算法计算的词语之间的语义相似度和人主观判断的语义相似度之间的一致程度是衡量语义相似度算法优劣的标准. 对于多义词、非邻域词和同义词三类情况下语义相似度的计算, 本文提出的算法充分借鉴了人判断词语语义相似度时的思考方式. 对于多义词, 人通常会先分析这个词有哪些不同的词义, 然后判断哪些词义之间的语义相似度最大. 例如在判断 "仪表" 和 "温度计" 之间的语义相似度时, 此时会认为 "仪表" 表示的是测量仪器; 而在判断 "仪表" 和 "相貌" 之间的语义相似度时, 则认为 "仪表" 表示的是人的外表. 因此对于每个多义词按词义进行分解, 是人判断语义相似度时的一种本能. 以往词向量模型只实现了每个词语的向量化表达, 并没有考虑词语具有多个义项, 导致基于词向量的多义词语义相似度计算精度较差. 为了解决这个问题, 本文通过对语料库进行词义消歧, 经过训练得到了词义向量模型, 模型中每个多义词的词义均对应一个向量. 在计算语义相似度时, 就可以先进行词义的分解, 然后获取每个词义的词向量, 从而计算不同词义之间的语义相似度.

在语料库文本中, 一些不经常同时出现在一个上下文窗口内的词对, 经过词向量训练之后, 语义相似度通常很小. 这些词对分为两种情况, 一种是本身词义之间存在较弱的语义相似度, 另一种是非邻域词情况, 通过词向量的距离无法准确体现词语之间真正的语义相似程度. 对此, 我们通过对原词的各词义进行了同义词扩展, 计算扩展的同义词和另外一个词的不同词义之间的语义相似度, 以此来校正非邻域词之间的语义相似度计算误差. 因此当两个词为非邻域词时, 仍能够通过各自的同义词信息来弥补两个词语之间语义相似度的缺失. 而对于本身词义之间相似性较弱的词语, 同义词扩展并不能对语义相似度产生明显影响. 对于 w_a 和 w_b 本身属于同义词的情况, 无法直接用词向量来体现. 通过同义词词林的同义词分组信息, 如果 w_a 和 w_b 同时出现在某一词义编码下, 并且编码尾号为 " = ", 则认为两者为同义词, 此时语义相似度计算的结果为 1, 与同义词关系相符合. 实际上, 一些词对通常具有多义词、非邻域词和同义词中的多个特征, 因此我们取三类情况下语义相似度的最大值作为两个词语之间的语义相似度.

综上分析, 本文提出的基于词义向量的词语语义相似度算法的设计思想和人判断语义相似度的方式是一致的, 充分利用词义向量和同义词信息, 对多义词、非邻域词和同义词三类情况的词语语义相似度计算质量进行了有效提升, 具备充分的合理性. 另外本文语义相似度算法在计算过程中, 每个词语不同词义之间的相似度均采用同一词义向量模型中对应词义向量之间的余弦距离来表示, 具有相同的取值范围, 因此在融合中无需再进行额外的归一化处理.

4 实验结果及分析

为了验证本文算法的有效性, 分三个方面设计实验. 1) 首先进行了词义消歧的对比实验并验证了词义消歧对于后续词语相似度算法精度的影响. 2) 然后将本文算法与其他相似度算方法在不同测试集下对比了斯皮尔曼 (Spearman) 系数, 评估语义相似度算法与人的主观评分之间的一致性程度. Spearman 系数是一种评价词语相相似度算法准确度的有效方式, 许多词向量模型[13−20] 均利用测试集下算法的计算结果与人的主观评分之间的 Spearman 系数来评价算法的优劣. 其值越大, 算法的正确性越好. 为方便比较, Spearman 系数全都被转换到 0 %～100 %的范围. 3) 最后选取了多组典型测试词分别在在多义词、非邻域词和同义词三种情况下比较了算法相似度计算的结果.

4.1 词义消歧实验结果分析

本实验采用词义消歧领域比较权威的 Senseval-3 数据集[7] 进行结果验证. 该测试集包含 20 个歧义词和 379 个语句实例. 词义消歧的评测指标包括两种: 微平均准确率 (Micro-average) P_{mir} 和宏平均准确率 (Macro-average) P_{mar}:

$$P_{mir} = \frac{\sum_{i=1}^{N} m_i}{\sum_{i=1}^{N} n_i} \quad (24)$$

$$P_{mar} = \frac{\sum_{i=1}^{N} p_i}{N}, p_i = \frac{m_i}{n_i} \quad (25)$$

[7] Senseval3: http://www.senseval.org/senseval3

其中 N 为测试歧义词的总数, m_i 为测试集中第 i 个词的词义被正确标识的语句个数, n_i 为该词测试语句的总数.

在本文词义向量构建过程中, 需要对整个文本语料库进行词义消歧处理, 而且为了支撑本文的相似度算法, 也需要明确标识词语的词义. 基于以上原因, 本文提出了基于同义词词林和词向量的词义消歧算法. 唐共波等[23] 提出了一种基于 HowNet 义原词向量的无监督词义消歧算法, 并应用于 SE-WRL 算法的 SAC 模型和 SAT 模型训练过程, 和本文词义消歧方法均属于基于知识库的词义消歧方法. 因此将本文词义消歧方法和基于 HowNet 义原向量的词义消歧算法进行对比会更有说服力. 在基于 HowNet 义原向量算法中, 词 w 在不同上下文中表达的词义通过如下词义消歧公式获得:

$$\boldsymbol{s} = \underset{\forall \boldsymbol{s}_j^w \in S, 1 \le j \le n}{\arg\max} \frac{\exp(\boldsymbol{w}_c'^{\mathrm{T}} \boldsymbol{s}_j^w)}{\sum_{k=1}^{n} \exp(\boldsymbol{w}_c'^{\mathrm{T}} \boldsymbol{s}_k^w)} \tag{26}$$

其中 S 为词 w 的所有词义向量集合, $\boldsymbol{s}_j^w$ 为第 j 个词义的向量, 表示为该词义包含的所有义原向量的均值. $\boldsymbol{w}_c'$ 为上下文词义向量, 是窗口内上下文词语的词向量的均值:

$$\boldsymbol{w}_c' = \frac{1}{2K'} \sum_{k=i-K'}^{k=i+K'} \boldsymbol{w}_k, k \neq i \tag{27}$$

唐共波的方法[23] 和本文方法用于预测中心词词义的上下文窗口均设置为 6, 即根据左右各 3 个词语. 需要强调的是测试集中的一些词语的词义没有包含在 HowNet 和同义词词林中, 如 “包” 和 “钱” 表示姓氏的义项, 在这些测试实例上将不能得到正确的词义预测结果, 最终会降低词义消歧算法精度, 但是对于本实验两种算法的对比并不造成影响. 表 4 为 Senseval-3 测试集上两种算法的词义消歧精度的对比, 可以看出本文算法表现出更好的词义消歧精度. 分析原因, 首先在上下文词义向量的计算上, 本文方法不是对上下文词语向量的简单叠加, 而是考虑了不同上下文词的重要性, 认为重要性与该词语和中心词的距离以及该词语的词义数量有关. 一个上下文词语离中心词越近, 其重要性越大; 上下文词语的词义数量越少, 其表达的词义越纯粹, 对上下文整体表达的语义影响越重要. 通过这种方式, 本文词义消歧方法能够得到更为准确的上下文词义向量. 另外在词语的不同词义的向量表达上, 本文采用了对相同词义的同义词进行加权叠加方式, 同义词的词义数量越少, 权值越大. 而唐共波等[23] 的算法对于词语的每个词义向量通过为其关联的所有义原的向量之和表示, 并没有考虑词义和义原之间属性关系的不同, 而且许多义原同时出现在词语不同词义中, 使得词义的区分不如本文方法明确. 综合以上原因, 本文方法在词义消歧任务中能够更好地辨别不同上下文中词语的词义.

词义消歧作为词义向量模型构建的重要步骤, 对于词义向量的表达也会产生影响. 我们设定了不同的上下文窗口, 分别进行词义消歧和词义向量模型的训练, 统计各窗口大小下词义消歧精度和语义相似度计算的精度, 来分析两者间的关系. 实验选择 wordsim-240 测试集[8] 来计算 WSME 算法的 Spearman 系数作为相似度算法的精度. 图 4 为词义消歧精度 (微平均和宏平均) 和语义相似度算法精度与上下文窗口大小之间的关系. 从图中可以看出, 语义相似度精度和消歧算法精度的变化趋势基本一致, 当窗口大小为 6 和 12 时, 词义消歧算法和基于词义向量的语义相似度均取得了最高的计算精度. 以上结果证明词义消的精度能够影响后续基于词义向量的语义相似度算法的精度. 考虑到窗口越大, 词义消歧所需时间越长, 综合准确度与实时性, 大小为 6 的窗口是最适合的.

表 4　词义消歧精度对比

Table 4　Evaluation results of WSD

比较算法	P_{mir} (%)	P_{mar} (%)
基于 HowNet 义原向量的方法[23]	36.35	40.19
本文算法	**42.74**	**44.08**

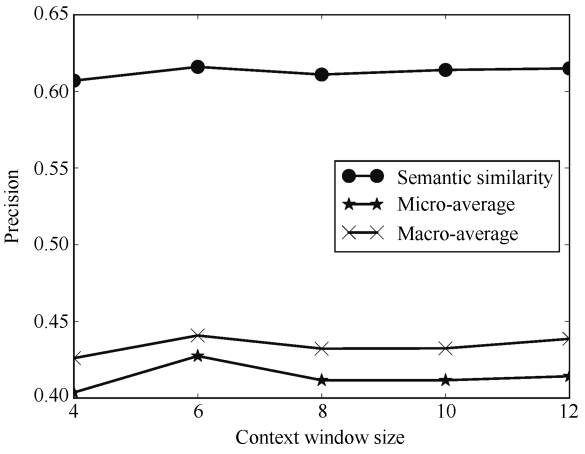


图 4　词义消歧精度和语义相似度精度与上下文窗口的关系

Fig. 4　The precisions of WSD and semantic similarity at different context window sizes

4.2　Spearman 系数比较

为了更全面地评估本文提出的相似度方法, 本实验分为三个子部分, 分别与同义词词林和词向量

[8] https://github.com/Leonard-Xu/CWE

简单叠加的方法、基于词向量模型的方法以及基于多元词向量的方法进行了 Spearman 系数的对比.

4.2.1 和同义词词林与词向量简单叠加方法对比

本节实验对同义词词林和词向量简单叠加的语义相似度方法 (Embedding Tongyici Cilin, Emb + TC)、基于词向量 (CBOW 模型) 余弦距离的语义相似度方法以及本文基于词义向量的方法 WSME 进行了对比. CBOW 模型和本文词义向量模型均基于搜狗新闻语料库训练, 测试集选择 wordsim-240 和 wordsim-297 数据集[15], 分别包含 240 个词对和 297 个词对. 测试集包含了多名用户对多组词对语义相似度的评分, 最终每组词的评分为多个人工评分的平均值. Emb + TC 的计算公式如下:

$$sim(w_1, w_2) = sim\_Emb(w_1, w_2) \times k + sim\_TC(w_1, w_2) \times (1 - k) \quad (28)$$

其中 $sim\_Emb(w_1, w_2)$ 为 CBOW 模型中 w_1 和 w_2 对应的的词向量的余弦距离, $sim\_TC(w_1, w_2)$ 为按照田久乐等[12] 提出的相似度方法计算的 w_1 和 w_2 在同义词词林中的相似度. 权值 k 分别取 0.3、0.5 和 0.7.

通过对比表 5 中两组测试集下的实验结果, 可以看出直接将同义词词林和词向量模型简单叠加的方法并不能比单纯基于词向量的方法有提升. Emb + TC 的计算精度随着 k 的增加而增加, 说明其精度主要依赖于词向量的距离. WSME 算法虽然也结合了同义词词林和词向量模型, 但却实现了更好的计算精度. 这是因为 WSME 没有直接利用同义词词林计算词语的相似度信息, 而是采用了同义词词林中的词义分类和同义词分组的信息, 而后者由于经过语言学家多年的编纂校对已经具有很高的准确度. 利用词义分类信息与词向量算法构建了词义向量模型, 得到对词语词义的准确表达. 同时利用同义词信息进一步提升了非邻域词和同义词的计算精度, 实现了比基于原始词向量模型以及简单叠加同义词词林和词向量更好的相似度计算结果.

表 5 WSME 与 Emb + TC 方法的对比

Table 5 The comparison result of WSME and Emb + TC

比较算法	wordsim-240	wordsim-297
CBOW	51.47	62.72
Emb + TC ($k = 0.3$)	30.87	45.69
Emb + TC ($k = 0.5$)	33.04	47.57
Emb + TC ($k = 0.7$)	39.18	53.34
WSME	**61.45**	**64.09**

4.2.2 和基于词向量模型的方法比较

本实验测试集仍选择 wordsim-240 数据集和 wordsim-297 数据集, 训练数据为搜狗新闻语料库. 搜狗新闻语料库包含 1 143 530 条新闻, 在训练词向量前进行了预处理, 去除了词语个数小于 50 的文本, 最终语料库包含 916 247 条新闻文本. 选择了 Word2vec 的两个子模型 CBOW 和 Skip-gram、字符信息增强的 FastText 模型[16−17]、义原信息增强的 SE-WRL 算法[18] 的两个子模型 (SSA 和 SAT) 和 WSME 算法对比算法计算的相似度与测试集中人工标注结果之间的 Spearman 系数, 整体地评估算法的准确性.

根据 SE-WRL 算法论文中的实验结果, SSA 和 SAT 在 wordsim-240 和 wordsim-297 上的表现优于 SAC, 因此本文算法选择 SE-WRL 中这两个性能更好的模型进行对比. 另外 WSME 算法根据构建词义向量过程中所选的词向量算法不同, 分为基于 CBOW 算法构建的词义向量模型的相似度算法 (WSME + CBOW) 和基于 Skip-gram 算法构建的词义向量模型的相似度算法 (WSME + Skip-gram). 在本实验中, 所有词向量模型和本文的词义向量模型的训练过程采用相同的配置参数和运行环境. 本文算法和各基于词向量模型的算法的 Spearman 系数如表 6 所示.

表 6 与词向量模型的 Spearman 系数对比

Table 6 The Spearman correlation result of models

比较算法	wordsim-240	wordsim-297
CBOW	54.07	62.72
Skip-gram	57.94	61.82
FastText[16−17]	59.05	63.97
SSA[18]	61.69	56.86
SAT[18]	58.42	61.80
WSME + CBOW	61.45	**64.09**
WSME + Skip-gram	**63.54**	62.64

通过对比表 6 的实验结果, WSME 算法在两个测试集下均取得了最高的 Spearman 系数, 表现出稳定的计算精度. 无论基于 CBOW 模型构建的词义向量还是基于 Skip-gram 模型构建的词义向量模型, 均实现了相对原词向量模型精度的提升. 另外词向量增强的两个方法 FastText 和 SE-WRL 方法相对基于原始词向量方法也有一定程度的提升. FastText 算法通过 N-gram 向量叠加的方式丰富了每个词语的语义信息, 相对 CBOW 模型对词义的表达更为准确, 但对于语义相似度准确性的提升效果没有 WSME 明显. 这是因为 FastText 的词向量模型对于多义词仍然用单一向量表示, 同样无

法准确表达多义词的不同词义. SE-WRL 在训练的词向量模型中嵌入了 HowNet 义原信息, 同样扩充了词向量表达的信息量. 从实验结果上看, SSA 在 word-sim240 测试集相对原始 Skip-gram 模型有一定的提升, 但在 wordsim-297 下却出现了下降, 这说明 SSA 方法的稳定性较低. SAT 在训练过程根据上下文对中心词的词义进行了词义消歧, 明确了中心词表达的词义, 从结果上看, SAT 的稳定性相对 SSA 更好. WSME 利用了同义词词林中的词义分类信息, 通过对语料文本进行词义消歧, 训练得到词义向量模型, 从而能够准确计算多义词每个词义之间的语义相似度. 从第 4.1 节实验的对比结果来看, 本文词义向量构建过程中使用的词义消歧算法相对 SAT 训练过程词义消歧算法具有更高的准确度, 这也是本文算法精度更高的一个原因. 另外, 在计算相似度时, 本文不仅考虑了每个多义词的不同词义, 而且利用了同义词信息, 结合扩展的同义词信息可以有效弥补在非邻域词和同义词情况下只依靠词义向量计算相似度的不足, 实现了准确度的显著提升. 以上实验结果也充分验证了 WSME 算法能够实现比现有基于词向量的算法更高的相似度计算精度.

4.2.3 和基于多元词向量模型的方法比较

本文词义向量模型能够准确表达词语的每个词义, 可以根据词多义词语所处的上下文或词对之间的关系计算词语间的相似度. 为了更好地评价本文方法对于多义词词语相似度的计算质量, 在本节实验与 Guo 等[20] 提出的多元词向量模型进行了对比, 该模型使用了 AvgSim 方法和 MaxSim 方法计算相似度, 我们选择其中表现更好的 MaxSim 方法的结果进行对比. 训练数据集和测试集分别选择该论文提供的中英双语对应数据集和 wordsim-401 测试集[9]. wordsim-401 中每个词对至少包含了一个多义词, 而且除了包含 401 个多义词词对之间的相似度评分之外, 还提供词对之间的关系, 如同义词、上下位关系词、主题相关及不相关等关系, 根据此关系可以进一步限定多义词的候选词义集合.

表 7 为两种方法的 Spearman 系数对比结果, 可以看出本文词义向量模型相对 Guo 的多元词向量模型在 wordsim-401 测试集表现出了更好的结果. 这是因为: 首先本文词义向量模型利用了同义词词林先验的词义分类信息相对直接通过聚类的方式除能够获得更准确的词义分类, 避免了聚类算法的误差, 在此基础上实现了多义词词义更准确的表达; 其次, WMSE 模型对词语的词义进行明确标识, 根据上下文或词对的关系可以进一步判断当前多义词表达的具体词义, 进而可以利用同义词信息进行相似度的计算. 很多多义词对之间同样具有非邻域词的性质, 而 Guo 及其他现有多元词向量模型均没有考虑此问题, 本文算法则利用同义词实现了对非邻域词相似度信息的有效补充. 基于以上原因, 本文基于词义向量的词语相似度方法对于多义词具有好的计算精度.

[9] ir.hit.edu.cn/ jguo

表 7 wordsim-401 数据集上的 Spearman 系数对比

Table 7 The Spearman correlation evaluated on wordsim-401

比较算法	polysemous-wordsim-401
Guo 等[20]	55.4
WSME	**56.9**

4.3 多义词、同义词和非邻域词的语义相似度计算结果比较

除了比较不同算法的 Spearman 系数, 本文还从 wordsim-240 和 wordsim-297 中选取了若干组典型的多义词、非邻域词、和同义词对基于 CBOW 词向量模型的算法、基于 FastText 词向量模型的算法和 WSME 算法进行比较, 模型均基于搜狗新闻语料库训练.

表 8 为多义词语义相似度的对比结果, 可以看出 WSME 算法计算的语义相似度和人的评分更接近, 而 CBOW 算法和 FastText 算法的计算结果与人的主观评分差距非常大. 这是因为每个多义词有多个词义, 如 “钱” 可以表示货币, 也可以表示重量单位. 单靠一个词向量难以准确表达每个多义词的语义特征, 本来词向量在向量空间中有多个不同的表示点, 现在却只用一个点来表达折中的词义. 因此单一向量表达多义词的词向量模型计算的相似度与人的评分有很大的误差. FastText 算法虽然利用 N-gram 信息使得词向量的表达更准确, 但训练的词向量与词语仍然是一对一的关系, 并没有根据不同上下文信息对词语的词义进行区分, 因此对多义词相似度计算精度的提升并不如 WSME 明显. WSME 算法通过对语料库进行词义消歧, 实现了语料文本中每个多义词的词义标识, 经过训练得到了表达单一词义的词义向量模型. 在语义相似度计算过程中, 对每个多义词按词义进行分解, 从词义向量模型中得到了对应分解后的单一词义词语的词义向量, 从而可以更为准确地计算两个词语不同词义间的相似度, 这和人判断两个多义词相似度的过程相一致, 实验结果也充分说明本文算法计算的多义词的语义相似度更准确.

词向量模型的训练充分利用了词语的上下文信息, 但对于非邻域词情况, 即使其自身词义接近, 但由词向量计算出的相似度值很低. 本实验选取了 3

组非邻域词对, 为了排除多义词因素对结果的影响, 测试词均为单一词义的词语, 其中 “基础设施” 没有被同义词词林收录, 其余的词只出现在同义词词林中的一个编码之下. 表 9 为三种算法计算的 3 组非邻域词之间的语义相似度结果, 以 “旅行” 和 “宾馆” 词对为例, 人的评分为 0.800, 直接使用 CBOW 词向量计算的结果为 0.003, 有明显的差距. FastText 算法结果为 0.096, 相对 Word2vec 更准确. WSME 计算的结果为 0.226, 相对前两种算法更加接近人的评分. 这是因为 WSME 除了考虑了词本身的语义信息, 还通过同义词词林对各词义的词进行同义词扩展, 只要两个词在语义上是相似的, 不会因为不常在同一上下文窗口内出现而造成语义相似度计算的误差.

表 8 多义词语义相似度计算结果

Table 8 The semantic-similarity result of polysemous words

词 1	词 2	人的评分	CBOW	FastText	**WSME**
自然	人	0.661	0.104	0.168	**0.443**
书	图书馆	0.772	0.253	0.409	**0.425**
钱	金融	0.775	0.080	0.022	**0.362**

表 9 非邻域词语义相似度计算结果

Table 9 The semantic-similarity result of nonadjacent words

词 1	词 2	人的评分	CBOW	FastText	**WSME**
旅行	宾馆	0.800	0.003	0.096	**0.226**
医生	责任	0.882	0.057	0.142	**0.160**
医院	基础设施	0.528	0.036	0.053	**0.129**

同义词的语义相似度计算结果如表 10 所示. 表中的三对比较词不在 wordsim-240 和 wordsim-297 测试集内, 因此没有给出人的评分数据. 但可以判定 “番茄” 和 “西红柿” 表示同一种事物, 理论上相似度为 1. 但 CBOW 算法计算的 “西红柿 – 番茄” 的相似度 (0.473) 却低于 “西红柿 – 黄瓜” 的相似度 (0.508), 这明显不符合人的主观认知. 而且, 由于 “番茄” 和 “西红柿” 为同义词, “西红柿 – 黄瓜” 的相似度和 “番茄 – 黄瓜” 的相似度应该相等, 但 CBOW 和 FastText 算法中 “番茄 – 黄瓜” 的语义相似度明显低于 “西红柿 – 黄瓜” 的语义相似度. 这是因为 CBOW 和 FastText 算法均没有单独考虑同义词情况, 仅依靠词向量的距离无法识别同义词关系, 计算的语义相似度会存在明显误差. WSME 算法利用了同义词词林先验的同义词分类信息, 能够准确地识别出待比较词对是否表达相同的词义, 如 “番茄 – 西红柿” 两个测试词均出现在最后一位为 “= ” 的同一编码下 (Bh06A32 =), 按照式 (23) 计算的结果为 1, 和理论值相等. 而且 “西红柿 – 黄瓜” 和 “番茄 – 黄瓜” 的语义相似度相等 (0.568), 小于 “西红柿 – 番茄”, 计算结果更为合理.

表 10 同义词相似度计算结果

Table 10 The semantic-similarity result of synonyms

词 1	词 2	CBOW	FastText	**WSME**
西红柿	番茄	0.473	0.697	**1.000**
西红柿	黄瓜	0.508	0.682	**0.568**
番茄	黄瓜	0.436	0.530	**0.568**

5 结论

本文提出了一种基于词义向量模型的词语语义相似度算法 (WSME), 实现了对多义词、同义词和非邻域词相似度计算精度的提升. 通过词义消歧算法明确了语料库中的每个多义词表达的词义, 并基于消歧后的语料利用 Word2vec 算法训练词义向量模型. 在计算词语的语义相似度时, 综合利用了词义向量信息和同义词词信息, 通过扩展的最大值融合方式得到了对词语相似度更准确的衡量. 为了验证本文方法的有效性, 分别进行了词义消歧精度、相似度计算精度和以及同义词、多义词和非邻域词三种情况下相似度计算结果的对比实验, 实验结果表明: 1) 本文提出的基于同义词词林和词向量的词义消歧方法相对其他基于知识库的词义消歧方法具有更好的准确度; 2) 相对基于同义词词林和词向量简单叠加的相似度计算方法算法、基于词向量的相似度算法以及基于多元词向量的相似度算法, 本文基于词义向量方法均取得了更好的计算精度, 而且具有较高的稳定性; 3) 本文方法对于多义词、同义词和非邻域词能够取得更接近人主观判断的相似度评价, 具有更好的合理性.

我们下一步的工作方向是利用 WordNet 英文语义词典对英文语料库进行语义消歧, 通过 WordNet 编码对语料库词语进行词义标识, 训练英文词义向量模型, 基于此模型提升英文词语相似度算法的计算精度.

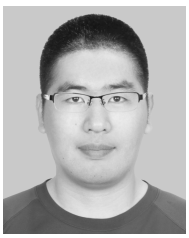
References

1 Li Wen-Qing, Sun Xin, Zhang Chang-You, Feng Ye. A semantic similarity measure between ontological concepts. *Acta Automatica Sinica*, 2012, **38**(2): 229−235
(李文清, 孙新, 张常有, 冯烨. 一种本体概念的语义相似度计算方法. 自动化学报, 2012, **38**(2): 229−235)

2 Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space [Online], available: https://arxiv.org/abs/1301.3781, September 7, 2013

3 Mikolov T, Sutskever I, Chen K, Corrado G, Dean J. Distributed representations of words and phrases and their compositionality. In: Proceedings of the 26th International Conference on Neural Information Processing Sys-

tems. Lake Tahoe, Nevada, USA: Curran Associates Inc., 2013. 3111−3119
4 Banu A, Fatima S S, Khan K U R. A new ontology-based semantic similarity measure for concept′s subsumed by multiple super concepts. *International Journal of Web Applications*, 2014, **6**(1): 14−22
5 Meng L L, Gu J Z, Zhou Z L. A new model of information content based on concept′s topology for measuring semantic similarity in WordNet. *International Journal of Grid and Distributed Computing*, 2012, **5**(3): 81−94
6 Seddiqui M H, Aono M. Metric of intrinsic information content for measuring semantic similarity in an ontology. In: Proceedings of the 7th Asia-Pacific Conference on Conceptual Modelling. Brisbane, Australia: Australian Computer Society, Inc., 2010. 89−96
7 Sánchez D, Batet M, Isern D. Ontology-based information content computation. *Knowledge-Based Systems*, 2011, **24**(2): 297−303
8 Sánchez D, Batet M, Isern D, Valls A. Ontology-based semantic similarity: A new feature-based approach. *Expert Systems with Applications*, 2012, **39**(8): 7718−7728
9 Zadeh P D H, Reformat M Z. Feature-based similarity assessment in ontology using fuzzy set theory. In: Proceedings of the 2012 IEEE International Conference on Fuzzy Systems. Brisbane, Australia: IEEE, 2012. 1−7
10 Li M, Lang B, Wang J M. Compound concept semantic similarity calculation based on ontology and concept constitution features. In: Proceedings of the 27th International Conference on Tools with Artificial Intelligence (ICTAI). Vietri sul Mare, Italy: IEEE, 2015. 226−233
11 Gao J B, Zhang B W, Chen X H. A WordNet-based semantic similarity measurement combining edge-counting and information content theory. *Engineering Applications of Artificial Intelligence*, 2015, **39**: 80−88
12 Tian Jiu-Le, Zhao Wei. Words similarity algorithm based on Tongyici Cilin in semantic web adaptive learning system. *Journal of Jilin University (Information Science Edition)*, 2010, **28**(6): 602−608
(田久乐, 赵蔚. 基于同义词词林的词语相似度计算方法. 吉林大学学报 (信息科学版), 2010, **28**(6): 602−608)
13 Pennington J, Socher R, Mannin C D. GloVe: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014. 1532−1542
14 Socher R, Bauer J, Manning C D, Ng A Y. Parsing with compositional vector grammars. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. Sofia, Bulgaria: Association for Computational Linguistics, 2013. 455−465
15 Chen X X, Xu L, Liu Z Y, Sun M S, Luan H B. Joint learning of character and word embeddings. In: Proceedings of the 24th International Conference on Artificial Intelligence. Buenos Aires, Argentina: AIAA, 2015. 1236−1242
16 Bojanowski P, Grave E, Joulin A, Mikolov T. Enriching word vectors with subword information [Online], available: https://arxiv.org/abs/1607.04606, June 19, 2017
17 Joulin A, Grave E, Bojanowski P, Mikolov T. Bag of tricks for efficient text classification [Online], available: https://arxiv.org/abs/1607.01759, August 9, 2016
18 Niu Y L, Xie R B, Liu Z Y, Sun M S. Improved word representation learning with sememes. In: Proceeding of the 55th Annual Meeting of the Association for Computational Linguistics. Vancouver, Canada: Association for Computational Linguistics, 2017. 2049−2058
19 Huang E H, Socher R, Manning C D, Ng A Y. Improving word representations via global context and multiple word prototypes. In: Proceeding of the 50th Annual Meeting of the Association for Computational Linguistics. Jeju Island, Korea: Association for Computational Linguistics, 2012. 873−882
20 Guo J, Che W X, Wang H F, Liu T. Learning sense-specific word embeddings by exploiting bilingual resources. In: Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers. Dublin, Ireland: COLING, 2014. 497−507
21 Lu Wen-Peng, Huang He-Yan, Wu Hao. Word sense disambiguation with graph model based on domain knowledge. *Acta Automatica Sinica*, 2014, **40**(12): 2836−2850
(鹿文鹏, 黄河燕, 吴昊. 基于领域知识的图模型词义消歧方法. 自动化学报, 2014, **40**(12): 2836−2850)
22 Chen X X, Liu Z Y, Sun M S. A unified model for word sense representation and disambiguation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014. 1025−1035
23 Tang Gong-Bo, Yu Dong, Xun En-Dong. An unsupervised word sense disambiguation method based on sememe vector in HowNet. *Journal of Chinese Information Processing*, 2015, **29**(6): 23−29
(唐共波, 于东, 荀恩东. 基于知网义原词向量表示的无监督词义消歧方法. 中文信息学报, 2015, **29**(6): 23−29)



李小涛 中国移动研究院工程师. 2016 年获得北京航空航天大学博士学位. 主要研究方向为物联网语义, 知识图谱. 本文通信作者.
E-mail: lixiaotao@chinamobile.com
(**LI Xiao-Tao** Engineer at China Mobile Research Institute. He received his Ph. D. degree from Beihang University in 2016. His research interest covers IoT semantics and knowledge graph. Corresponding author of this paper.)

游树娟 中国移动研究院助理工程师. 2016 年获得中国海洋大学硕士学位. 主要研究方向为知识图谱, 语义相似度计算.
E-mail: youshujuan@chinamobile.com
(**YOU Shu-Juan** Assistant engineer at the China Mobile Research Institute. She received her master degree from Ocean University of China in 2016. Her research interest covers knowledge graph and semantic similarity computation.)

陈 维 中国移动研究院首席科学家, 主持物联网领域的创新研发工作. 主要研究方向为机器智能和边缘计算.
E-mail: wai.w.chen@gmail.com
(**CHEN Wai** Chief scientist at the China Mobile Research Institute, where he directs R&D of Internet of Things (IoT). His research interest covers machine intelligence and edge computing.)