文章编号：2096-1472(2018)-10-11-03
DOI:10.19644/j.cnki.issn2096-1472.2018.10.003

基于公共词集对长篇小说相似度的研究

郭 涛，霸元婕，李绍昂
(吉林大学计算机科学与技术系，吉林 长春 130012)

摘 要：传统的文本相似度计算基于向量空间模型(VSM)，文本映射成独立的、互不关联的词构成的向量。由于长篇小说具有比普通文本更为复杂的构成元素，以及更加紧密的上下文联系，传统算法忽略词项的上下文联系，并且产生高维向量，因此算法的效率和精度不理想。为此，本文基于公共词集对长篇小说进行相似度计算，并对公共词集进行上下文约束检查，得到关联比较紧密的词集，作为小说的主要特征。实验结果表明，对于某些小说类型，效果有很大的提升。

关键词：公共词集；小说相似度；上下文约束
中图分类号：TP391.1 **文献标识码**：A

Similarity of Long Novels Based on Common Word Sets

GUO Tao,BA Yuanjie,LI Shaoang
(*School of Computer Science,Jilin University,Changchun* 130012,*China*)

Abstract: Traditional text similarity computation is based on Vector Space Model (VSM),where the text is mapped into independent and unrelated words.Because novels have more complex elements and much closer context than ordinary texts,the traditional algorithm ignores the context of the words and produces the high dimensional vector,so that the efficiency and accuracy of the algorithm are not ideal.For this reason,this paper calculates the similarity of the novels based on the common word set,and carries out the context constraint check on the common word set to achieve a more closely related word set as the main feature of the novel.The experimental results show that for some types of novels,the effect is greatly improved.

Keywords:common word set;novel similarity;context constraint

1 引言(Introduction)

随着互联网技术的发展，网络上的文本数据呈现爆炸式增长，文本处理算法的相关研究也随之发展起来。其中，文本相似度计算成为热点研究方向，其目的在于建立一个合理的衡量模型，对文本间的相似程度进行量化。小说作为一种文学作品，与普通文本有较大区别，小说的构成要素要比普通文本复杂很多，比如时间、地点、人物、社会、环境等等，并且小说的上下段落、上下情节之间联系十分紧密。所以，必须要从新的角度建立小说相似度的衡量模型。

目前经典的文本相似度计算算法大部分基于向量空间模型(VSM)[1]。向量空间模型将文本视作由独立的、互不关联的词构成的一个向量，并且把词语在文中出现的频数作为文本的主要特征。通过将文本映射成一个向量模型，文本相似度计算也就转换成向量之间的相似度计算。小说作为一种特殊的文本类型，词语之间的关联比普通的文本更加紧密，如果依然将小说表示成向量空间模型，将失去很多重要的特征信息，尤其是词条间的上下文信息，词语之间的关联隐含着情节信息，对文义的理解起着至关重要的作用[2]。不仅如此，对于长篇小说而言，向量空间模型将产生一个维数十分巨大的向量，严重影响算法的效率，问题将变得不可行。

本文主要介绍了一种基于公共词集对长篇小说相似度研究的算法[3]。对小说进行预处理后，建立Map映射结构，在构建公共词集的过程中，加入上下文约束，最终得到满足上下文约束的若干词集簇，并以此作为衡量相似度的依据，建立相似度衡量算法，并通过实验验证算法可行。

2 相关工作(Related work)

2.1 向量空间模型

文本的内容特征常常用它所含有的基本语言单位，如

字、词或者短语等来表示，这些基本的语言单位被统称为文本的项[4]。向量空间模型(Vector Space Model,VSM)将文本D转化为由词项w构成的m维向量V^m，即：

$$V^m = [w_1, w_2, w_3, \dots, w_m]^T$$

文本中的每个项相互独立，可以通过计算向量之间的距离来衡量文本之间的相似度。每个词项往往都赋予一个权重(Term Weight)，表示该词项在文本中的重要程度。*TF*−*IDF*(Term Frequency−Inverse Document Frequency)是使用最广泛的一种权重计算方法，公式如下：

$$TF(w_i) = \frac{f_{w_i}}{\sum_{1\le j\le m} f_{w_j}} \tag{1}$$

$$IDF(w_i) = \log\frac{D_s}{D_{w_i}} \tag{2}$$

其中，f_{w_i}表示词项w_i的出现频数，D_s表示文档集中文本数量，D_{w_i}表示词项w_i在文档集中包含该词项的文本数量。

在文本中的出现频率反映该词项的重要程度，词项在多个文本中的出现情况反映了词项的文义甄别能力，TF−IDF综合考虑了以上两点，每一个词项的权重由TF权值和IDF权值两个部分组成。通过计算向量之间的余弦角，可以得到两个文本向量之间的相似程度，定义如下：

$$\alpha = [w_{\alpha_1}, w_{\alpha_2}, w_{\alpha_3}, \dots, w_{\alpha_m}] \tag{3}$$

$$\beta = [w_{\beta_1}, w_{\beta_2}, w_{\beta_3}, \dots, w_{\beta_m}] \tag{4}$$

$$\cos(\alpha,\beta) = \frac{\sum_{i=1}^{m} w_{\alpha_i} w_{\beta_i}}{\sqrt{\sum w_{\alpha_j}}\sqrt{\sum w_{\beta_j}}}$$

2.2 公共词集

从小说的词法方面研究其文本特征，如果不考虑词项之间的先后顺序，可以比较小说词域之间的相交程度来衡量相似度。将小说的词集提取出来，两篇小说的公共词集可以反映小说在用词造句方面的相似性[5]。相对于两篇小说的平均文本长度而言，如果公共词集包含的词项数越多，小说的相似程度越高，两篇小说的用词方式更为接近；反之，若公共词集包含的词项数越少，相似程度越低。在对小说进行文本预处理操作后，分别统计词项的频数和位置信息，可以得到小说N_1和N_2的公共词集CWS，公共词集中的元素由词项w_i和词项在小说中的频数f_{w_i}构成。可以用采取如下计算公式计算相似度：

$$\mathrm{SimCom}(N_1, N_2) = \frac{L(cws)}{L(N_1)+L(N_2)} \tag{5}$$

其中，$L(N_1)$表示小说N_1在文本预处理且过滤低频词之后的词项数；$L(N_2)$同上；$L(cws)$表示小说N_1和N_2的公共词集词项频数之和。然而在文本中，词序在文意表述中起到至关重要的作用，如果忽略词项之间的先后顺序，将文本看作由一定频数的词项构成词集，将造成很大的信息量损失。通过加入关于词项之间的先后顺序和前后距离的约束条件，可以有效地解决这个问题。

2.3 上下文约束

上下文约束条件能够有效地筛选小说中出语义联系比较紧密的词项集合[6]。

定义1：词序列切分Slice。将小说文本记为长度为l的词序列N，即：

$$\boldsymbol{N} = [w_1, w_2, w_3, \dots, w_l]^T$$

词集M：

$$\boldsymbol{M} = \{w_i | w_i \in N, 1 \le i \le l\}$$

则N中词集M的词序列切分定义为：

$$\boldsymbol{Slice}(\boldsymbol{M},\boldsymbol{N}) = \{(p,q) | M \subseteq w_{(p,q)}, 0 \le p < q \le l, M \not\subset w_{(p,q-1)}, M \not\subset w_{(p+1,q)}\} \tag{6}$$

其中，p、q代表词项在词序列N中的位置信息，(p,q)代表分片的位置信息。由于词集中的词条在小说中可能多次出现，所以$Slice(M,N)$可能包含多个分片。

定义2：上下文约束(Context_constraint)。一个词集M满足上下文约束定义为：选定一个词窗$Word_window$，词集M满足上下文约束，当且仅当在小说N中，对于M的词序列切分$Slice(M,N)$，至少存在一组$(p,q) \in Slice(M,N)$满足$|q-p| \leqslant Word_window$

形式化表示如下：

$$ContextCon(M) = 1 \Leftrightarrow \exists (p,q) \in Slice(M,N), |\mathrm{q}-\mathrm{p}| \le Word\_window \tag{7}$$

从公共词集中挖掘满足上下文约束的子集，可以得到一个元素数为h的集合簇Cs：

$$\boldsymbol{Cs} = \{Cs_i | 1 \le i \le h, |Cs_i| \ge 2, ContextCon(Cs_i) \to \mathrm{true}\} \tag{8}$$

即对于集合簇Cs每一个元素，至少包含两个词项，构成一个满足上下文约束的词集。引入上下文约束条件，可以筛选出满足一定距离约束的词集，通过调节词窗大小可以灵活地调整约束强度大小。提取出一篇小说中上下文约束的词集构成的集合，可以有效地从词域角度表示出小说的文本特征，进而通过比较两篇小说的集合簇Cs的相似程度来衡量小说的相似度[7]。

3 公共词集上下文约束算法(Context constraint algorithm)

3.1 上下文约束

上下文约束其实是一种距离约束，限制了词集的覆盖范围，一旦设定词窗大小，利用上下文约束，可以从公共词集中形成多个词项子集，每个子集中的词项都落在词窗范围大小的片段上。不满足上下文约束的词集，词项之间词序位置跨度大，分散稀疏，关联性小，如果作为小说的特征之一，将造成文义上的偏差，降低相似度计算的精准性；反之，满足上下文约束的词集语义联系紧密，往往可以构成独立的语义段落，更有可能体现小说的文义特征[8]。考虑下面两个的片段：

Segment 1:*With the rapid development of Internet reporting,data mining is being widely applied in*

many areas,which brings more power for economic development...

Segment 2:By collecting data and information from multi-dimensional tools,workers can locate the location accurately and start mining gold mines...

虽然两个片段都包含了*data*和*mining*两个词项，但是很明显两个片段文义相差非常大，如果不考虑上下文约束，只考虑公共词项，则两个片段将可能获得较高的相似度。如果考虑上下文约束，两个词项在*Segment 1*的距离很小，而在*Segment 2*的距离相对较大，可以在一定程度上区分两个片段。文本中距离较近的词项往往可以构成一个固定含义或者特殊含义短语或者词组，这样的短语和词组往往是围绕文本的中心内容，对表征文本的内容特征起着关键的作用。如果两篇文本的共有短语较多，那么文义往往是相近的；反之，如果两篇文本几乎没有共同的短语，那么相似程度可能是很低的。

本文基于这个观点，在挖掘出小说之间的公共词集时，在词集上施以上下文约束，得到满足约束的词集簇，综合考虑公共词集和满足约束的词集簇，给出相似度计算公式。

3.2 Map索引结构

在挖掘小说的公共词集和进行上下文约束过程中，要频繁地访问词项的内容，以及在文中的位置信息。对于长篇的小说而言，频繁的检索将产生较高的复杂度，为了记录并快速访问词条在小说中的位置信息，本算法采用一种Map索引结构，结构如图1所示。

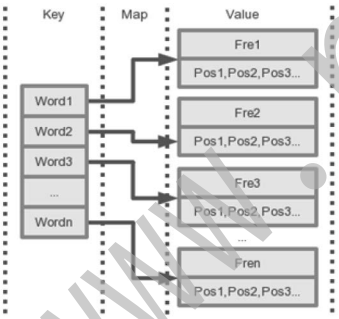

图1 Map索引结构

Fig.1 Map index structure

其中，*Map*结构中*Key*表示词项本身，*Value*表示词项频数和词条在小说中的位置信息，构成*Map*映射关系。由于词条可能在小说中多次出现，其位置信息可能有多个，如果保留全部的位置信息，会大大增加存储空间，并且对算法产生复杂度为O(n)的困扰，可以设置适当的数量限制，避免保留过多的位置信息。

3.3 相似度计算

公共词集可以反映小说内容上的相似程度，如果公共词集越大，说明小说的内容越相近。同时，满足上下文约束的词集语义联系更强，可以更加准确地传达小说的文义特征。综合考虑小说之间的公共词集和满足上下文约束的子集簇对小说文义的影响，给出相似度计算公式：

$$Sim(N_1,N_2)=\varepsilon\cdot SimCom(N_1,N_2)+(1-\varepsilon)\cdot SimContx(\mathrm{cws}) \quad (9)$$

$$SimContx(cws)=\frac{\sum_{i=1}^{k}Cs_i}{L(cws)} \quad (10)$$

其中，$Sim(N_1,N_2)$表示两篇小说N_1和N_2的相似度值，$SimCom(N_1,N_2)$表示由两篇小说的公共词集计算而得的相似度，$SimContx(cws)$表示从公共词集CWS中挖掘出来的满足上下文约束词集簇Cs得出来的相似度，ε作为权重因子，综合考$SimCom(N_1,N_2)$，$SimContx(cws)$二者对计算总体相似度的贡献。

4 实验(Experiment)

实验选取了54篇长篇英文小说，分成六种类型，分别是*Adventure*、*Children*、*Mystery*、*Romance*、*Science Fiction*、*Social*。

从每种类型的小说中选取一篇主题最明确的小说，作为比较相似度的基准。当对一篇小说进行归类时，与所有基准计算相似度，将小说归类成与之相似度最大的基准对应的类型。通过计算在归类过程中的正确率和召回率两个指标，衡量每种方法归类的效果，实验比较了没有加入上下文约束(方法1)和加入上下文约束(方法2)两种方法的分类效果，详见表1和表2所示。

表1 数据集

Tab.1 Data set

类型	数量
Adventure	8
Children	15
Mystery	9
Romance	6
Science Fiction	9
Social	7

表2 实验结果对比

Tab.2 Comparison of experimental results

类型	正确归入	实际归入	总数	准确率	召回率
Adventure	5	7	8	0.71	0.63
	2	5	8	0.40	0.25
Children	11	14	15	0.79	0.73
	6	18	15	0.33	0.40
Mystery	5	7	9	0.71	0.56
	3	8	9	0.38	0.33
Romance	4	4	6	1	0.67
	2	3	6	0.67	0.33
Science Fiction	5	8	9	0.63	0.56
	1	7	9	0.14	0.11
Social	5	14	7	0.36	0.71
	6	13	7	0.46	0.86

(下转第10页)