

基于 Wikipedia 的语义相关度计算

刘 军, 姚天昉

(上海交通大学计算机科学与工程系, 上海 200240)

摘 要: 在意见挖掘中, 为实现特殊领域知识的语义相关度计算, 提出基于 Wikipedia 的语义相关度计算方法。在构建 Wikipedia 类别树的基础上, 通过 Wikipedia 类别向量表示 Wikipedia 中的词汇, 形成一部包含各种领域知识的 Wikipedia 词典, 利用该词典计算语义相关度。实验结果表明, 该方法的斯皮尔曼等级相关系数可达到 0.77。

关键词: 语义相关度; 领域知识; Wikipedia 类别树; 意见挖掘

Semantic Relevancy Computing Based on Wikipedia

LIU Jun, YAO Tian-fang

(Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai 200240, China)

【Abstract】 In order to compute semantic relevancy for the specific domain knowledge in opinion mining, this paper proposes a semantic relevancy computing method based on Wikipedia. On the basis of constructing a category tree from Wikipedia, it represents the vast words in Wikipedia by using the category and the result in a Wikipedia dictionary which contains rich domain-specific knowledge, and then computes semantic relevancy by using the dictionary. Experimental results show Spearman rank correlation coefficient of this method can reach 0.77.

【Key words】 semantic relevancy; domain knowledge; Wikipedia category tree; opinion mining

1 概述

在意见挖掘中, 需要处理 BBS、Blog 等社区上的网络语言^[1]。网络语言经常出现专有名词和专业术语等特殊领域相关的词语, 处理它们需要更全面的语义知识。如苹果的 ipod 很棒。在不考虑上下文的情况下, 苹果可指水果, 也可指公司。通过比较(苹果公司-ipod)和(苹果水果-ipod)的语义相关度的大小可消去歧义, 这对评价对象 ipod 的语义理解是很有帮助的。由于传统语义相关度^[2-4]研究没有使用世界知识, 因此很难正确计算领域相关词的相关度。本文在 Gabrilovich 方法^[5-6]的基础上, 利用 Wikipedia 来解决该问题。

2 基于 Wikipedia 的语义相关度计算

2.1 Wikipedia 词典的构造方法

Wikipedia 中的每一个概念对应于 Wikipedia 网站上的一个页面, 包括标题、类别、正文等。即:

$$Wikipedia = \{page^+\}$$

$$page = \{title, category^+, text, url^+, \dots\}$$

类别是 Wikipedia 对概念进行分类。对于每个类别, 使用所有属于它的概念来表示, 而概念则用解释它的正文表示。即:

$$category = \{page^+\} = \{text^+\}$$

设 Wikipedia 类别的数量为 M , 词的数量(词典规模)为 N 。采用向量空间模型对类别进行表示, 则任意一个类别 $category_i$ 的词向量为:

$$category_i = [(word_1, weightW_{i1}), (word_2, weightW_{i2}), \dots, (word_N, weightW_{iN})] \quad (1)$$

对上述结果进行倒排索引, 则词典中任意一个词 $word_i$ 的类别向量为:

$$word_i = [(category_1, weightC_{i1}), (category_2, weightC_{i2}), \dots, (category_M, weightC_{iM})] \quad (2)$$

综上, Wikipedia 词典构造完毕。

2.2 语义相关度的计算过程

任意 2 个文本 $text_1$ 和 $text_2$, 相关度的具体计算如下:

步骤 1 将文本表示为词向量

设 $text_1$ 包含 Wikipedia 词典中词的数量为 K , $text_2$ 包含 Wikipedia 词典中词的数量为 L 。即

$$BOW(text_1) = \{word_{11}, word_{12}, \dots, word_{1K}\}$$

$$BOW(text_2) = \{word_{21}, word_{22}, \dots, word_{2L}\}$$

根据 $tf \times idf$ 计算特征权重, 得到 $text_1$ 和 $text_2$ 的词向量:

$$text_1 = [(word_{11}, weightW_{11}), (word_{12}, weightW_{12}), \dots, (word_{1K}, weightW_{1K})] \quad (3)$$

$$text_2 = [(word_{21}, weightW_{21}), (word_{22}, weightW_{22}), \dots, (word_{2L}, weightW_{2L})] \quad (4)$$

步骤 2 将词向量转化为类别向量

对 $\forall word_{li} \in BOW(text_i)$, 查找其在 Wikipedia 词典中的类别向量, 即由式(2)可得:

$$word_{li} = [(category_{i1}, weightC_{li1}), (category_{i2}, weightC_{li2}), \dots, (category_{iM}, weightC_{liM})] \quad (5)$$

类别 $category_j$ 在 $text_i$ 中的累加权重的计算为:

$$weightC_{ij} = \sum_{l=1}^K weightC_{lij} \times weightW_{li} \quad (6)$$

通过式(3)、式(5)、式(6)计算 $text_1$ 的类别向量为:

$$text_1' = [(category_{11}, weightC_{11}), (category_{12}, weightC_{12}), \dots, (category_{1M}, weightC_{1M})] \quad (7)$$

基金项目: 国家自然科学基金资助项目(60773087)

作者简介: 刘 军(1981—), 男, 硕士研究生, 主研方向: 自然语言处理, 意见挖掘; 姚天昉, 副教授、博士

收稿日期: 2010-04-09 **E-mail:** steven.jun.liu@sjtu.edu.cn

其中, $weightC_{11} \leq weightC_{12} \leq \dots \leq weightC_{1M}$ 。

同理可得:

$$text'_2 = [(category_{21}, weightC_{21}), (category_{22}, weightC_{22}), \dots, (category_{2M}, weightC_{2M})] \quad (8)$$

其中, $weightC_{21} \leq weightC_{22} \leq \dots \leq weightC_{2M}$ 。

步骤 3 根据两者的类别向量计算相关度

(1)采用向量夹角余弦公式

$$Relevancy^A(text'_1, text'_2) = \frac{text'_1 \times text'_2}{|text'_1| \times |text'_2|} \quad (9)$$

(2)采用距离度量

因为类别具有层次关系, 所以可采用距离度量的方式计算:

$$Relevancy^B(text'_1, text'_2) = \frac{\sum_{i=1}^M (weightC_{1i} + weightC_{2i})}{2} \cdot \frac{dist(category_{1i}, category_{2i})}{dist(category_{1i}, category_{2i}) \times \beta + \alpha} \quad (10)$$

其中, $dist$ 为 2 个类别之间的距离; 参数 $\alpha > 0$, α 一般取距离的平均值; 参数 $\beta \geq 1$, β 为对距离的惩罚系数。

3 实验

3.1 前期处理

从中文 Wikipedia 下载 2008 年 6 月 25 日的 XML DUMP 文件, 解压后得到一个 897 MB 的 XML 文件。经过繁体转简体和 WikiPrep 工具处理后, 得到一个 572 MB 的 XML 文件, 包含 221 208 个页面。因为有些页面是噪声页面或过于细化, 所以去除 Wikipedia 用于消歧义的页面和字数小于 100、链入数或链出数少于 5 的页面。最后得到一个 367 MB 的 XML 文件, 包含 88 503 个页面。

3.2 Wikipedia 类别树

每个 Wikipedia 类别都有一个类别页面, 该页面的 title 由“Category:”和“类别名”组成。若某类别有父类别, 则在该类别页面的正文会以“[[Category: 父类别名]]”标明。首先抽取出 Wikipedia 中所有类别, 并找到每个类别的直接孩子, 以“一个类别及其直接孩子为一行”的格式输出到一个文件中, 总共抽取出 15 056 个类别。

在这 15 056 个类别中包含一些 Wikipedia 编辑用的类别, 这些类别不能用于词典构造。首先删除类别名中包含“专题|用户页|帮助|模板|存档|首页|用户框|消歧义|页面|维护|条目|维基|Wiki”的类别。然后按类别的父子关系建立类别森林, 结果发现以“页面分类”为根的类别树的结点数占据所有结点的 98% 以上。从中可看出, Wikipedia 的类别具有良好的层次性, 其他类别树的出现有可能是 Wikipedia 编辑的疏忽。本文手动将结点数超过 100 的类别树并入以“页面分类”为根的类别树。如以“音乐”为根的类别树不在以“页面分类”为根的类别树中, 则手动将“音乐”添加为“页面分类”的子类来完成归并。对于结点数小于 100 的类别树, 则舍弃。

经过以上处理, 最后得到由 13 417 个类别组成的类别树。该树的层次为 21 (根节点为第 1 层), 2 个节点间的最大距离为 35, 第 10 层的结点到其他结点的平均距离为 12。

3.3 Wikipedia 词典的构造

Wikipedia 的页面对专有名词、专业术语等概念进行解释, 许多页面的 title 本身就是专有名词、专业术语。为了有效地提取出这些词, 设定如下规则: 当页面的类别属于人物或地名, 则直接将该 title 提取出来。否则, 若 title 长度小于 5, 且在 Wikipedia 正文中出现的次数大于 30, 则将该 title

提取出来。通过以上处理, 从中提取出 28 196 个普通分词词典中不存在的词。

本文对 Wikipedia 正文进行分词后, 去除 $DF < 3$ 的词、全数字的词和停用词, 最后得到 295 848 个词作为 Wikipedia 词典的容量。

4 实验结果

为了验证本文方法, 同时实现了 Gabrilovich 方法^[4]。Gabrilovich 方法使用 Wikipedia 概念向量表示词, 设 Wikipedia 中的概念数 (即页面数, 一个概念对应一个页面) 为 V , $V \gg M$ (M 为类别数)。与 2.1 节类似, 利用向量空间模型, Gabrilovich 首先得到概念的词向量为式 (11), 然后倒排索引, 得到词的概念向量为:

$$page_i = [(word_1, weightW_{i1}), (word_2, weightW_{i2}), \dots, (word_N, weightW_{iN})] \quad (11)$$

$$word_i = [(page_1, weightP_{i1}), (page_2, weightP_{i2}), \dots, (page_V, weightP_{iV})] \quad (12)$$

因为 Wikipedia 概念不存在层次关系, 利用 Gabrilovich 方法计算语义相关度只能采用向量夹角余弦公式 $Relevancy^A$ 。本文分别采用向量夹角余弦公式和距离度量式 $Relevancy^B$ 计算语义相关度。表 1 是 Gabrilovich 方法与本文方法计算出来的一些示例词之间的相关度。从结果可以看出, Gabrilovich 方法计算出的 (微软-谷歌)、(手机-MMC) 和 (拨打-挂断) 等词对的相关度偏小, 因为这 3 个词对实际上都很相关: 微软和谷歌同为 IT 界的知名企业、MMC 为手机内存卡的一种常见类型、拨打和挂断为对应的 2 个动作。而本文方法的 3 种计算结果都较为合理, 尤其是利用了类别的层次关系的 $Relevancy^B$ 的结果更理想。这主要是因为 Wikipedia 概念很细化, 2 个词的概念向量的交集通常很小。遇到这种情况时, Gabrilovich 方法很难利用概念之间的关系去增补这 2 个词原有的相关性, 导致许多词之间的相关度偏小。而由于 Wikipedia 类别具有良好的定义, 是具有层次关系的知识库, 因此更适合用于语义相关度的计算。

表 1 3 种方法计算示例词的相关度比较

| 序号 | 文本对 | Gabrilovich 方法 | $Relevancy^A$ 方法 | $Relevancy^B$ 方法 | |
|----|------------|-------------------|---------------------|----------------------|----------------------|
| | | | | $\alpha=12, \beta=1$ | $\alpha=12, \beta=2$ |
| 1 | Windows-微软 | 0.530 | 0.881 | 0.634 | 0.518 |
| 2 | 微软-谷歌 | 0.005 | 0.183 | 0.440 | 0.340 |
| 3 | Windows-谷歌 | 0.015 | 0.192 | 0.388 | 0.293 |
| 4 | 苹果-ipod | 0.186 | 0.488 | 0.367 | 0.294 |
| 5 | 苹果公司-ipod | 0.359 | 0.670 | 0.539 | 0.437 |
| 6 | 苹果水果-ipod | 0.143 | 0.308 | 0.304 | 0.227 |
| 7 | 手机-电话 | 0.275 | 0.462 | 0.545 | 0.414 |
| 8 | 手机-CDMA | 0.104 | 0.398 | 0.577 | 0.456 |
| 9 | 手机-MMC | 0.049 | 0.123 | 0.472 | 0.373 |
| 10 | 拨打-挂断 | 0.036 | 0.101 | 0.555 | 0.416 |

本文在 WordSimilarity-353^[7]数据集上做了进一步实验。WordSimilarity-353 数据集包括 353 个词对, 每个词对由 13 名~16 名人员对它们的相关度打分, 得分的平均值作为每个词对的相关度。将这些词对翻译成中文, 如 (Harvard-Yale) 翻译成 (哈佛-耶鲁)、(psychology-Freud) 翻译成 (心理学-弗洛伊德)。WordSimilarity-353 中相关度的值范围为 0~10, 本文将程序产生的相关度 0~1 乘以 10 转换为 0~10 之间的数。使用斯皮尔曼 (Spearman) 等级相关系数比较各种方法产生的相关度与 WordSimilarity-353 的相关度差异。从表 2 可看出, 本文方法产生的相关度与人工判断的相关度更接近。

(下转第 46 页)

| 视图类型标记 | 节点 | Agent | 软件测试信息序列 |
|--------|-------------|-------|----------------------------|
| P1 | 1、4、5、6、7、8 | Ap1 | STS p1, STS p4,..., STS p8 |
| P2 | 2、9 | Ap2 | STS p2 |
| P3 | 10 | Ap3 | STS p10 |

表 3 测试数据生成方法

| STS | 元素 | 规则集合 | 标志集合 | Agent 集 | 软件测试数据集 |
|-------|-------------|------|------|---------|----------|
| | Operation | L1 | T1 | M1 | D11~D110 |
| STS1, | Object | L2 | T2 | M2 | D21~D210 |
| STS2, | Var | L3 | T3 | M3 | D31~D310 |
| STS3, | VarType | L4 | T4 | M4 | D41~D410 |
| ..., | VarRange | L5 | T5 | M5 | D51~D510 |
| STS10 | Start | L6 | T6 | M6 | D61~D610 |
| | Final | L7 | T7 | M7 | D71~D710 |
| | Restriction | L8 | T8 | M8 | D81~D810 |

```
graph TD
    MS((Method Selector)) --> RB1[(规则库1)]
    MS -.-> M2((Manager2))
    M2 -.-> NA((New Agent))
    Admin[Admin] -- "(1)编写新的 Agent" --> NA
    Admin -- "(2)添加Agent 规则" --> RB1
```

| 元素 | 规则集合 | 标志集合 |
|--------------|----------|----------|
| Operation | L1: Lnew | T1: Tnew |
| Object | L2: Lnew | T2: Tnew |
| Var | L3: Lnew | T3: Tnew |
| VarType | L4: Lnew | T4: Tnew |
| VarRange | L5: Lnew | T5: Tnew |
| Start | L6: Lnew | T6: Tnew |
| Final | L7: Lnew | T7: Tnew |
| Rrestriction | L8: Lnew | T8: Tnew |

本文提出一种基于 Agent 的软件测试数据生成框架,通过原型的开发,验证了该框架的可行性。该框架有以下 2 个优点: (1)测试数据生成方法扩展性好。通过在测试数据生成方法 Agent 组中增加新的 Agent 就可以增加新的测试数据生成方法,且具有一定的智能性。(2)可基于多种 UML 视图生成测试数据。不同的视图遍历 Agent 提取出的测试信息序列保证了可对基于多种 UML 视图生成测试数据。

- [1] Wang Linzhang, Yuan Jiesong, Yu Xiaofeng, et al. Generating Test Cases from UML Activity Diagram Based on Gray-box Method[C]//Proc. of Software Engineering Conference. Edinburgh, Scotland, UK: [s. n.], 2004: 284-290.
- [2] Lugato D, Maraun F, Traon L, et al. Automated Functional Test Case Synthesis from THALES Industrial Requirements[C]//Proc. of Real-time and Embedded Technology and Applications Symposium. Toronto, Canada: IEEE Computer Society, 2004: 104-111.
- [3] Nebut C, Fleurey F, le Traon Y, et al. Automatic Test Generation: A Use Case Driven Approach[J]. IEEE Transactions on Software Engineering, 2006, 32(3): 140-155.
- [4] Bai Xiaoying, Dai Guilan, Xu Dezheng, et al. A Multi-Agent Based Framework for Collaborative Testing on Web Services[C]//Proc. of SEUS-WCCIA'06. Los Alamitos, CA, USA: IEEE Computer Society, 2006: 72-78.
- [5] 毛新军. 面向主体的软件开发[M]. 北京: 清华大学出版社, 2005.

| 方法 | Spearman 等级相关系数 |
|--|-----------------|
| Gabrilovich 方法 | 0.64 |
| Relevancy ^A 方法 | 0.69 |
| Relevancy ^B 方法 $\alpha=12, \beta=1$ | 0.77 |
| Relevancy ^B 方法 $\alpha=12, \beta=2$ | 0.75 |

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved. <http://www.cnki.net>