上海大学

SHANGHAIUNIVERSITY

毕业设计（论文）

UNDERGRADUATEPROJECT(THESIS)

题目:XXXXX

学院	计算机工程与科学学院
专业	计算机科学与技术
学号	**XXX**
学生姓名	**XXX**
指导教师	宋　波
起讫日期	**2016.09.05 – 2016.12.16**

目录

XXX 中文论文题目 XXX

摘要

XXX。

关键词：XXXX，XXX，XXX，XXX

XXXEnglish Title XXX

ABSTRACT

In recent years, with the rapid development of Micro-blog, the need that users gain the access to information is also a linear growth momentum. The amount of Sina Micro-blog registered users has reached 503 million by 2012. However, the daily flood of Micro-blogs has a serious impact on the quality of information users receive. Thus, how to find content that they are interested in quickly and accurately? Or can we push the information according to the user's interest actively? That is what this paper concerns.

User model is a formal description of users' interests. To establish a precise user model for Micro-blog users, in order to recommend the information they concerned about and contents they are interested in, is the development trend of personalized recommendation. The achievements of this paper can be used for e-commerce, public opinion monitoring, advertising and other areas.

In this paper, Micro-blog user model construction method based on the ontology technology is discussed. First of all, according to each user's micro-blog content analysis, extract the keywords which represent the content information of each Micro-blog; then create user's eigenvectors in order to calculate the user's interest degree; finally match these keywords with the ontology library to create Micro-blog user interest model. The achievements of this paper will lay the foundation of personalized service based on Micro-blogs.

Keywords: Ontology, Micro-blog, User model, Interest Degree

第1章 绪论

本章主要描述了微博用户模型构建的背景、意义，分析了相关课题国内外的研究现状，进而提出了本文所要研究的内容及目标。

§1.1 微博用户模型构建的背景及意义

微博的数量和质量千变万化，各种海量、实时的数据信息已严重影响着用户接收信息的质量，进而影响着用户的生活质量。传统的人找信息和人找服务的模式已越来越难以满足用户的需求。如何为微博用户建立一个精准的用户模型，以便于之后为其推荐所关注的信息和感兴趣的内容，是各大移动电子商务网站进行个性化推荐的发展趋势。本课题研究在微博上基于本体的用户模型构建方法具有现实意义。

§1.2 用户模型研究现状及存在的问题

近年来国内外学者对用户模型的研究做了大量的工作，而基于微博的用户模型构建也开始逐步成熟起来，下面具体阐述研究现状以及存在的问题。

§1.2.1 研究现状

近年来，用户建模技术作为个性化服务中的基础，愈来愈受到重视，并逐渐地从个性化服务中独立出来，形成了专门的研究方向。研究人员逐渐意识到个性化服务质量的好坏不仅取决推荐技术或者检索技术，而且还取决于准确的用户模型。

国内的研究人员对用户模型构建和更新也展开了研究，例如大连理工大学林鸿飞和杨元生[1]根据用户提供的各类示例文档，通过考察特征、段落和类别的表达能力构建用户模型。南京大学多媒体技术研究所开发的个性化搜索引擎DOLTRI-Agent[2]采用一些相互关联的关键词组成用户模型，对每个关键词设置权值来表示用户对该关键词的感兴趣程度。国防大学的应晓敏[3]提出构建细粒度的基于关键词的用户建模方法，以更好体现出用户间的兴趣差异。国防科技大学的徐振宁[4]和李勇[5]构建了一个包括个性化领域本体的用户模型，跟踪记录用户在Web 上的浏览和检索过程，从大量数据中统计、分析和计算出用户的个性化信息需求。

在国外，Fragoudis 和 Likothanassis[6]对几个典型的个性化服务系统 LIRA[7]，Letizoa 等采用的用户建模方法进行了综述和分析，指出用户建模在个性化服务系统中的重要地位。Pazzani 和 Binsusu[8]通过用户对浏览页面的标注获取用户感兴

趣与不感兴趣的页面作为训练样本，而后计算单字的期望信息增益，选择期望信息增益大的 128 个单字构成用户模型。Chan[9]通过观察用户对页面中超链接的选择获取用户感兴趣与不感兴趣的页面作为训练样本，而后计算单字的期望互信息，选择期望互信息大的 250 个单字构成用户模型。Schwab[10]等通过观察用户对页面的选择获取用户感兴趣的页面作为训练样本，而后以出现在感兴趣页面中指定位置的单字构成用户模型。Adomavicious 和 Tuzhilin[11]采用数据挖掘方法对用户个体的访问记录进行挖掘，挖掘出来的关联规则以及用户登记的个人信息构成用户模型。

在微博的用户模型研究方面，近年来国内学者做了大量研究工作。例如广东社会主义学院的余伟[12]设计了一个基于本体的微博用户行为分析模型构架。北京邮电大学的赵岩露[13]等提出了基于微博用户兴趣模型的发现算法。而国外对 Twitter 的用户模型研究也有很多。

§1.2.2 微博存在的问题

虽然，近年来微博得到了空前的发展。相比于传统博客，微博传播模式更加便捷，更新的频率更高。作为新兴的媒体，目前仍存在很多问题。

（1）很多用户感兴趣的有用信息，往往被迅速湮没。

（2）微博信息过于简单，微博之间的联系松散、逻辑关系复杂，容易引起误解。

（3）微博监管困难，对于敏感信息传播的预测和监控缺乏有效手段。

§1.2.3 微博用户模型构建的难点

虽然用户建模技术已较为成熟，但针对微博这一特殊的平台，仍然存在了以下一些技术难点：

（1）对微博信息收集时，如何能够获取到大量有效的数据。

（2）微博信息短小精悍，对用户模型构建的准确性影响较大。

§1.3 本文研究内容及目标

本文针对微博中存在的问题，试图对用户的微博内容进行分析，提取用户的兴趣，并建立微博用户模型，为微博信息推荐、舆情监控、微博营销等提供技术支持。

§1.3.1 研究内容

本文研究基于本体的微博用户模型构建方法，具体研究内容有一下几个方面。

（1）领域本体构建；

（2）用户微博收集；

（3）微博内容分析；

（4）兴趣主题提取；

（5）用户模型构建。

§1.3.2 研究目标

针对本文的研究内容，制定了以下几项指标：

（1）自动对搜集到的所有用户（实验 10 个以上）的所有微博（30 条以上）进行分词；

（2）自动统计每个用户的关键词词频；

（3）合理计算每个用户模型中的兴趣度；

（4）根据已有的本体库建立用户模型；

（5）开发系统原型，验证提出的方法。

§1.4 本文组织结构

整篇论文分为五章。

第一章介绍了研究背景、研究意义，分析了用户模型研究现状以及存在的问题和难点，并提出了本文的研究内容以及研究目标。

第二章主要介绍了本体的基本概念，并提出了基于维基百科的本体库构建方法。

第三章首先介绍了用户模型的基本概念及其表示方法，其次着重介绍了微博用户模型的构建方法，主要分为特征词提取、兴趣度计算和用户模型的生成。

第四章主要描述了微博用户模型构建系统设计与实现，展示了系统整体设计、数据库设计和各功能模块设计的内容。

第五章对全文进行了总结，归纳了本文的主要工作与创新点，并指出了需要进一步研究的问题。

第2章 基于维基百科的本体构建方法

本章具体描述了基于维基百科的本体构建方法：介绍了本体的基本概念，并引出本文所使用的基于维基百科的本体构建方法。

§2.1 本体概述

本节介绍了本体的基本概念以及目前研究学者常用的四种构建方法。

§2.1.1 本体的基本概念

在计算机领域，1991 年开始，研究者们对本体做了多次说明，它表示的含义也更加清晰明确，现在人们一般认为本体论是对概念化对象的明确表示和描述[14]。随着研究者们对本体研究的不断完善，本体的定义有很多种，以下是几种比较有代表性的定义。

1991 年，Neches[15]等给出了构成相关领域词汇的基本术语、关系，以及这些词汇外延的规则。1996 年，Swartout[16]提出本体是一个知识库结构中术语集合，该结构中的术语是按照继承关系组织起来的，强调了本体中术语(Terms)的重要性。

1993 年，Gruber[17]提出本体是概念模型明确的规范说明。1997 年，Borst[18]提出本体是共享概念模型的形式化规范说明。1998 年，Studer[19]等对上述两个定义进行了深入研究，认为本体是“共享概念模型的明确的形式化规范说明”，它有以下四个方面的含义：

（1）本体是一个概念模型(Conceptualization)，它是指通过抽象客观世界中一些现象的相关概念得到的模型。它表现的含义独立于具体的环境状态；

（2）本体的明确性(Explicit)，本体包含的概念和概念之间的约束都应该有明确的定义；

（3）本体是形式化的(Formal)，意思是本体应该是计算机可处理的；

（4）本体是可共享的(Sharable)，本体中概念、关系、属性的描述是基于标准的、规范的、能被共享的。

本体的定义多种多样，其核心都一样，把本体作为一种描述资源的手段，为不同的主体进行知识交流提供语义基础。在计算机领域中，对本体的研究主要是如何实现这种统一标准，以及本体的构建方法、本体描述语言、本体的管理和本体的应用等内容。

根据本体的语义特性，本体可以应用于不同的领域，致力于提高服务的联想能力和准确性。

总而言之，尽管本体的定义方式多种多样，但本体所包含的基本要素：概念、

概念之间的关系等。

§2.1.2 本体构建方法

目前，本体构建成功的案例很多，根据不同的领域，构建的方法也不一样，现在还没有构建本体的标准。许多研究人员根据经验总结出来了一些方法，1995年，Gruber 提出构建本体的五条规则如下：

（1）明确性和客观性：本体应该是背景独立的、客观的，能反映社会真实情况，满足可计算性，具有明确的、客观的形式化语义；

（2）完整性：给出的定义应该是完整的，能表达特定属于的含义；

（3）一致性：只是推理产生的结论与属于本身的含义不产生矛盾；

（4）可扩展性：在扩展本体功能的时候，可以自由添加新的术语而对已有本体的结构和内容不做修改；

（5）最少约束：在满足可能的知识共享需求的基础上本体的约定应该最小。它可以通过只定义通讯所需的词汇或者定义约束最弱的公理来保证。

目前比较普遍的构造特定领域的本体，一般都需要相关领域专家的参与。以下列举了一些在项目实践过程中形成的方法：

IDEF-5 方法：IDEF(ICAM Definition Languages)方法是上世纪七十年代由美国空军发明的。在 1981 年针对集成计算机辅助制造(Integrated Computer Aided Manufacturing，简称 ICAM)项目中用于描述企业内部运作的建模方法。最初该方法只是应用于制造业，经过改造后，适用于软件开发。目前已经形成了一系列方法。包括 IDEF1X 和 IDEF0 到 IDEF14 共 16 套方法，每一套方法都通过建模程序获取某特定类型信息，其中 IDEF-5 是本体描述获取语言。

骨架法(Skeletal Methodology)：该方法是由爱丁堡大学人工智能应用研究所开发企业建模过程中总结出来的。

评价法：该方法由多伦多大学企业集成实验室，在开发虚拟企业本体工程项目时总结出来的。通过建立制定知识的逻辑模型，用一阶逻辑构造形式化的模型，包括企业设计本体、工程本体、计划本体和服务本体。

七步法：该方法是斯坦福大学医学院提出的基于 Protégé本体构建工具的一种领域本体构建方法。一共包括七个步骤：

（1）确定只是本体的专业领域和范畴；

（2）考察复用现有只是本体的可能性；

（3）列出本题中的重要术语；

（4）定义类和类的层次体系；

（5）定义类的属性；

（6）定义类的分面(Facets)；

（7）创建本体实例。

§2.2 基于维基百科的本体构建

以上传统的构建本体的方法仍然存在许多弊端，下面就针对这些不足进行分析阐述，并提出本文所使用的基于维基百科的本体构建方法的优势所在。

§2.2.1 维基百科简介

维基百科(Wikipedia)是一个自由、免费、内容开放的网络百科全书，参与者来自世界各地。这个站点使用 Wiki，这意味着任何人都可以编辑维基百科中的任何文章及条目。

维基百科是一个基于 Wiki 技术的全球性多语言百科全书协作计划，同时也是一部用不同语言写成的网络百科全书，其目标及宗旨是为全人类提供自由的百科全书——用他们所选择的语言来书写而成的，是一个动态的、可自由访问和编辑的全球知识体。

维基百科自 2001 年 1 月 15 日正式成立，由维基媒体基金会负责维持，其大部分页面都可以由任何人使用浏览器进行阅览和修改。因为维基用户的广泛参与共建、共享，维基百科也被称为创新 2.0 时代的百科全书、人民的百科全书。这本全球各国人民参与编写，自由、开放的在线百科全书也是知识社会条件下用户参与、大众创新、开放创新、协同创新的生动诠释。英语维基百科的普及也促成了其它计划，例如维基新闻、维基教科书等计划的产生，虽然也造成对这些所有人都可以编辑的内容准确性的争议，但如果所列出的来源可以被审察及确认，则其内容也会受到一定的肯定。维基百科中的所有文本以及大多数的图像和其他内容都是在 GNU 自由文档许可证下发布的，以确保内容的自由度及开放度。所有人在这里所写的文章都将遵循 copyleft 协议，所有内容都可以自由的分发和复制。

截至 2013 年 1 月，维基百科条目数第一的英文维基百科已有 415 万个条目，而全球所有 282 种语言的独立运作版本共突破 2100 万个条目，总登记用户也超越 3200 万人，而总编辑次数更是超越 12 亿次。大部分页面都可以由任何人使用浏览器进行阅览和修改，英文维基百科的普及也促成了其它计划。

Wiki 一词来源于夏威夷语的“wee kee wee kee”，原本是“快点快点”的意思。在这里“WikiWiki”指一种超文本系统。这种超文本系统支持面向社群的协作式写作，同时也包括一组支持这种写作的辅助工具。

§2.2.2 传统方式构建本体存在的问题

本体的构建是一个系统性工程，由于没有统一的构建原则、方法，到目前为止，本体工程仍处于相对不成熟的阶段，整个建设过程在以下几个方面还存在很多问题[20]。

（1）构建方法

7 种构建方法都是从具体领域本体的开发中总结出来的，应用领域很有限，大多数方法的细节比较粗，相关技术比较少，没有一种方法完全按照生命周期法进行开发。另外，由于没有统一的构建原则作为指导，整个本体的构建过程难以进行规范的管理。

（2）可扩展性

随着领域的不断发展、变化，必然会有更多的领域相关概念和关系引入到核心本体中，需要通过知识的进一步获取、概念的进一步扩充或更改等方式，不断改进和扩展领域本体。但是目前本体的维护和扩展问题还没有得到很好的研究和支持。

（3）共享和重用

领域本体构建的目的是为不同系统提供彼此交流的语义基础。目前，为减少构建本体的工作量，多数的研究均利用现有术语丰富的叙词表作为构建工作的起点。而叙词表到本体的转换还没有一个统一的标准，在转换的过程中各项目使用的描述语言以及描述广度和深度不尽相同，给今后本体之间语义互操作和重用造成了困难。

§2.2.3 基于维基百科的本体构建

维基百科全书于 2001 年 1 月投入运行，到 2013 年 1 月，英文版的维基百科全书就包含了 415 万多个条目，全部条目超过了 2100 万条，具有内容相互独立的 282 种语言版本。维基百科全书具有与大英百科全书类似的写作风格，相近的准确性，但内容的丰富性已经超过了大英百科全书。其内容不仅包括大量的插图，还包含了大量的多媒体内容，时效性更是印刷版的大英百科全书无法比拟的。与传统百科全书相比，维基百科特点突出：

（1）首先，维基百科始终将自己定位为包含人类所有知识领域的百科全书，而不仅仅是一本词典、在线论坛或其他。

（2）其次，计划也是一个 Wiki，允许大众广泛参与。维基百科是第一个使用 Wiki 系统进行百科全书编撰工作的协作计划。

（3）最后，维基百科是一部内容开放的百科全书，其内容允许任何第三方

不受限制地复制、修改及再发布，广泛的参与性，使得维基百科的权威性、中立客观性得到了保证。它方便不同行业的人士寻找知识，而使用者也可以不断增加自己的知识从而充实自己。当然，因为一些特殊的原因，目前中文版和英文版和百科全书也有部分敏感词汇被禁用。

目前，维基百科定期免费提供各个语言版本的所有数据，放到网络上供人下载。其中，数据包中包括有 page.sql 文件和 categorylinks.sql 文件，前者记录了所有页面的基本信息，如：页面标题、命名空间、页面长度等，后者记录了各个页面标记的类目信息。[21][22]虽然维基百科网站也提供了树状形式的类目网络供人浏览，但是这个网页的数据存在如下缺陷：

（1）由人工维护，所以数据更新比较缓慢；

（2）迫于部分类目深度过深，所以这个网页把这些类目删除掉，如“生物分类树”；

（3）由于采用简单的深度遍历算法将图输出为树状结构，所以部分类目深度失衡，例如“自然科学”类目就没有子类目。

基于以上原因，我们编程实现了类目网络的自动生成。基于维基百科中文本体的构建主要包括以下几个步骤：

（1）中文繁简转换

目前中文存在两种书写系统——繁体中文与简体中文。一般来自台湾、香港、澳门的使用者使用正体中文（繁体中文），来自中国大陆、新加坡、马来西亚则使用简体中文。[23]作为一个全球华人共同创作的平台，中文维基百科发布的数据中，既有繁体形式的，也有简体形式的，甚至很多在同一篇文章中繁简夹杂。这给我们的抽取工作带来很大的不便：一方面，文本繁简混杂的问题使得我们不能用现有的基于单一文字模式的中文信息处理工具直接分析文本；另一方面，繁简夹杂必然使得我们的研究成果不能很好地得到利用。所以，我们在利用维基百科所提供的简繁对应词表基础上，借鉴 MediaWiki 1.4 的繁简转换功能的“用字模式”，实现了繁简转换功能。

（2）类目网络清理

在维基页面分类系统中，存在着若干为了方便管理而添加的元类目，例如：“维基百科站务”。[24]因为这些类目所含语义信息较少，所以我们必须清理这些类目。我们剔除所有包含以下关键字的类目：维基，列表，模板，维基人，专题，分类，条目，小作品。另外，从数据库自动生成的类目网络中存在一些孤立点，我们将此类类目也全部清除。在清理之前，中文维基类目之间的直接连接数为 21776 个，清理后的直接连接关系总数 14009 个。

（3）识别父子关系

首先界定两个概念：下位词与“Is-a”关系（父子关系）[25][26]。语言学家 Fromkin 和 Rodman[27]认为，下位词是一个一般化词语具体化之后的相关词语集合。例如，深红色，朱红，绯红色都是红色的下位词，而红色就是它们的上位词。同时，红色又是颜色的下位词。因此，下位关系也就是一般化术语（如多边形）和它的具体化实例（如三角形）之间的关系。在计算机科学中，常常将此关系称为 Is-a 关系[28]。例如，用“红色 is a 颜色”来描述红色和颜色之间的下位关系。在知识表示和面向对象编程与设计中，在 A is a B 中，Is-a 表示类 A 是类 B 的子类，即 B 是 A 的父类。换言之，"A is a B"通常意味着概念 A 是概念 B 的具体化，概念 B 是概念 A 的一般化。举例而言，“水果”是“苹果”、“桔子”、“芒果”等概念的一般化。我们可以说，“苹果 is a 水果”。下文对这些类目之间的 Is-a 和 Not Is-a 关系进行自动识别。

经过以上步骤可以建立一棵基于维基百科的概念知识树，这就是我们维基百科本体知识树的原型。

§2.3 本章小结

本体库的构建是微博用户模型构建的基础，基于所构建完成的类目网络结构，可以生成所有类目的树形结构本体，进而为解决知识的共享和重用问题提供了新思路，也成为当前信息科学领域的研究热点之一。由于传统本体构建方法存在诸多问题，本文中利用维基百科这一新的互动开放的信息交流平台，构建出中文本体库，为后续进行微博用户模型的构建打下了基础。但本体库的完备性将对用户模型构建的准确性影响较大，因此对本体的研究工作仍任重而道远。

第3章 基于本体的微博用户模型构建方法

本章是全文的重点章节，全面阐述了本文的工作内容。首先简单介绍了用户模型的概念及其表示方法，随后将本文的构建过程完整的进行了描述，并将部分系统功能运行的结果作了展示。

§3.1 用户模型概述

在基于内容的个性化推荐中，文本内容是主要的研究对象。自然语言所表示的文本内容需要转化为机器能够识别和可计算的模型才能进行进一步的研究。用户模型并不仅仅是对用户兴趣的准确描述，由于可计算性是它对用户模型的基本要求，也就是说，用户模型不是对用户个体的一般性简单描述，而是一种面向算法的，具有特定数据结构，形式化的用户兴趣描述，它是实现个性化服务的基础和核心。

根据 Gerhard Fischer[29]的论述，可以认为广义上的用户模型有如下三种：

（1）用户头脑中的概念模型。这是用户头脑中关于计算机系统及其所应具有的功能的模型，表示了用户对计算机系统的理解和期望，该模型随着用户使用系统经验的增加而不断完善；

（2）设计者的用户模型。设计者头脑中关于用户的模型，是设计者对用户特征的描述，被设计者用来作为系统设计的基础；

（3）计算机系统的用户模型。它是由设计者在设计阶段依据设计者的用户模型用计算机软件构造的，在系统的运行过程中实现的。

由此可以看出，用户建模是这样一个过程：设计者根据用户概念模型调整设计者用户模型，将设计者用户模型用软件的方法转换为计算机系统的用户模型，用户建模的最终目标是计算机系统模型。狭义上的用户模型是指软件系统的用户模型，我们更关心的也是计算机所拥有的关于用户特征的模型。

本文的用户模型是一种对于用户兴趣内容特征的描述和表达，可以收集并提取用户的兴趣偏好，并与本体库进行匹配，进而更好地理解用户的需求和任务，实现个性化的推荐服务。

§3.2 用户模型表示方法

用户模型表示是用户建模的基础，决定了用户模型反映用户信息的能力和可计算能力。本节将讨论用户模型表示的有关方法。

§3.2.1 常用用户模型表示方法

用户模型表示方法有很多种，目前使用较为广泛的有关键词表示法、主题表示法和向量空间模型表示法等。

关键词表示法是以用户感兴趣的一组关键词来表示用户模型。例如{篮球，火箭，后卫}。关键词可以由用户自己设定，也可以通过用户的行为、他们在网上留下的信息及其它特征来获取。

主题表示法是关键词表示法的一种改进，这种表示法是以用户感兴趣的信息的主题来表示用户模型的。例如用户对体育、文化、科技感兴趣，则该用户的用户模型表示为{体育，文化，科技}。

向量空间模型表示法则是利用特征词和相应的权值向量来表示用户模型。其基本思想是根据用户感兴趣的文档中各个关键词的出现频率建立特征词及其相应的权值向量来表示用户模型。

虽然存在多种不同的方法来表示用户模型，但常用的用户模型表示方法普遍存在以下两大问题：

（1）缺乏统一标准。仅能被特定的系统所运用，不能在不同系统间实现共享。

（2）缺乏领域知识支持。大部分的表示方式只是对用户兴趣的简单罗列，没有考虑用户兴趣间的关联关系，难以表达用户兴趣的语义内容。

§3.2.2 本体用户模型表示方法

基于本体的用户模型表示方法使用规范的结构模式描述用户兴趣，把用户兴趣与领域知识的语义概念层次相结合，具有强大的语义表达能力、兼容性和可扩展性，可以很好地解决常用用户模型存在的问题。

本文中使用的本体用户模型是一棵带权的结点树，如**错误!未找到引用源。**所示，图中两结点之间存在父子关系，比如 Sports 是 Football 的父节点。节点旁边的数字则是每个兴趣节点的兴趣度大小，也就是用户对这个兴趣的感兴趣程度。

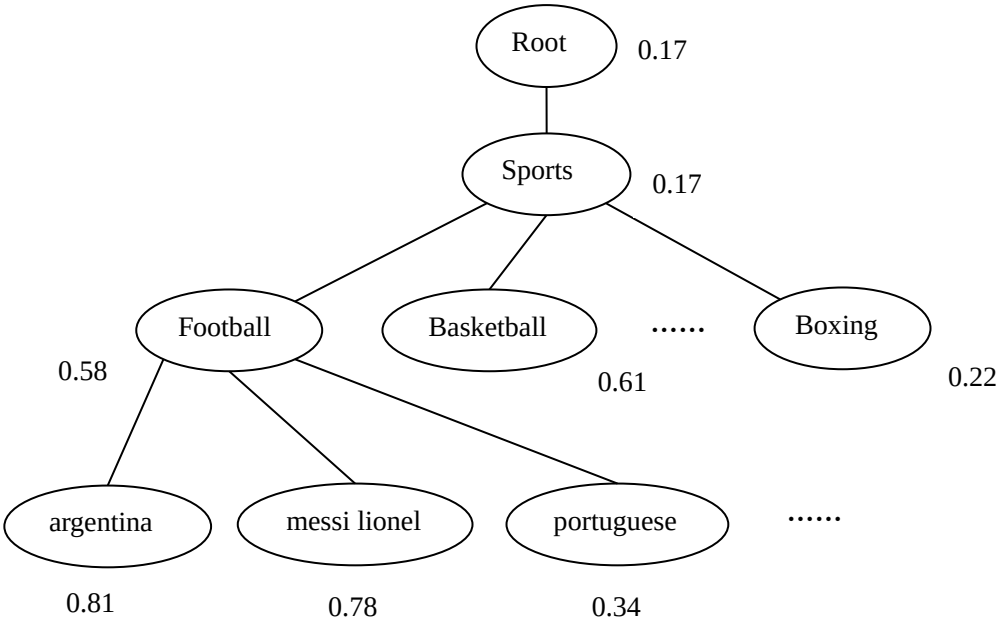

图 3-1 本体用户模型兴趣树

基于本体的用户模型表示方法主要有以下几个优势：

（1）充分描述用户兴趣的语义。

用户兴趣中的一个词条往往包含丰富的语义。该方法能描述该词条对应的用户兴趣的层次概念，结合领域本体，使兴趣表示带着丰富的语义。

（2）兼容性和可扩展性。

该方法具有很好的兼容性和可扩展性。根据该方法建立的模型可以方便地移植到其它系统中，也可随着应用的发展，对模型进行扩展。

（3）自适应性。

随着现实世界知识体系的变化以及用户兴趣的变化，该方法能够自动适应这些变化，准确表达用户的当前兴趣。

可以说，基于本体的用户模型表示方法是用户模型表示方法的发展方向，因此本文将实现基于本体的用户模型构建。

§3.3 微博用户模型的构建方法

微博用户模型的构建主要包括特征词提取、兴趣度计算以及用户模型的生成。以下将分为三个小节来详细描述构建方法。

§3.3.1 特征词提取

首先，我们将获取微博用户信息，理想的用户建模方法应该无需用户主动提供任何信息。自动用户建模就是根据用户在微博平台上注册留下的信息及用户发

布在微博平台上的微博信息来构建用户模型。我们可以通过微博平台提供的 API，获取到用户的昵称、密码、编号、姓名、关注数、粉丝数、发表微博的数量等个人基本信息以及每个用户在平台上发布过的所有微博信息内容来构建用户模型。

在中文信息处理领域，对中文自动分词的研究已较为成熟，经典的分词方法主要有最大向前匹配法、逐词遍历匹配法、最小向前匹配法等。本文未对分词方法进行深入研究，而是选择直接使用开源的中国科学院计算技术研究所研制的汉语词法分析系统 ICTCLAS (Institute of Computing Technology, Chinese Lexical Analysis System)[31]，该分词系统分词准确率达到 97%以上。本文实现的本体构建系统中使用 ICTCLAS 汉语分词系统官网提供的 32 位 Windows 操作系统下的 JAVA 版本的中文分词和词性标注。

计算机不具有人类的智能，不能像人类一样阅读微博后根据自身的知识和理解能力对微博内容产生理解。因此，在进行微博语义扩展之前首先要将微博转换成易被计算机理解和识别的结构形式。微博的表示要求能够准确有效的表达微博内容，并且还要易于计算机处理。

目前，典型的文本表示方法主要有：布尔模型（Boolean Model）、向量空间模型（VSM）、语言模型（Language Model）、潜在语义索引（LSI, Latent Semantic Indexing）[32]和概率检索模型（Probability Model）[33][34]。这些模型从不同的角度出发，使用不同的方法标注特征词权重和相似度计算等问题。

向量空间模型由 Salton 等人于 20 世纪 70 年代提出，并成功地应用于著名的 SMART 文本检索系统。VSM 被广泛应用于文本分类、文本聚类、信息检索等领域。近年来，在文本挖掘领域向量空间模型已经成为最常用的文本表示方法。VSM 是基于这样一个关键假设下提出的，即文档中各词条出现的先后顺序是无关紧要的，每个特征词对应特征空间的一维，他们每一维对于判定文档所属的类别所起的作用是相互独立的。因此，可以把一篇文档看成是一系列无序词条的集合，从文档中选取出 n 个特征词来表示文本就是形成一个 n 维向量空间。例如一篇文档中选取三个特征词 t1、t2、t3，那么这篇文档就表示为 $d=(t_1,t_2,t_3)$。但是对于整个文档来说，每个特征词对文本的重要程度不同，因此，需要对每个特征词赋予一定的权重。一篇具有 n 个特征词的文档利用 VSM 表示方法就可以表示为公式（1）。

$$d_i=((t_{i1},w_{i1}),(t_{i2},w_{i2}),...,(t_{ij},w_{ij}),...(t_{in},w_{in})) \quad (1)$$

其中，di 是第 i 篇文档，tij 表示第 i 篇文档的第 j 个关键词，wij 是第 i 篇文档的第 j 个关键词权重。关于权重的计算方法有很多种，将在 3.3.2 兴趣度计算步

骤中再作介绍。本文中微博文本表示采用 VSM 方法。

§3.3.2 兴趣度计算

经过文本分词处理后，需要抽取一定数量的特征词作为向量的各维表示文本。然而，文本中每个特征词对文本主题内容的贡献度不一样，即每个特征词的权重不同，如何准确有效地计算特征词权重成为重要的研究点。在研究最初特征词的权重只有 0 或者 1，如果该特征词在文本中出现过它的权重就设为 1，否则设为 0。这种方法完全没有体现出在文本中出现的特征词之间对文本主题内容贡献度的差异性，所以这种权重计算方法慢慢被更精确的基于词频统计的方法替代。

常用的权重计算方法有布尔函数、特征词频平方根、WIDF 函数及 TF-IDF 法等。目前使用最为广泛也是本文中所用到的方法就是 TF-IDF 法，计算方法如公式（2）所示。

错误!不能通过编辑域代码创建对象。 （2）

其中，**错误!不能通过编辑域代码创建对象。**表示在文本**错误!不能通过编辑域代码创建对象。**中第 j 个特征词**错误!不能通过编辑域代码创建对象。**出现的次数，N 表示文本集中所有文本数，n 表示文本集中含有特征词**错误!不能通过编辑域代码创建对象。**的文本数，**错误!不能通过编辑域代码创建对象。**表示文本**错误!不能通过编辑域代码创建对象。**中第 j 个特征词的权重。根式**错误!不能通过编辑域代码创建对象。**是归一化因子。

将每个特征词在某个用户发表的所有微博中出现的权重相加，就得到了该用户对于该特征词的兴趣度值。

§3.3.3 用户模型的生成

本文中利用基于维基百科的中文本体自动向上扩展建立一个树状层次结构，其中最主要的关系为上下位关系，下一层的同义词集是其父节点的下位关系，反之亦然。上下层的关系也是包含与被包含的关系，下一层的节点包含于其父节点。我们手动建立了一个基于维基百科分类下的分类特征词本体库，共输入有 4757 条记录，分为财经、IT、健康、体育、旅游、教育、招聘、文化、军事九个大类。

当之前分析出的用户兴趣主题与我们所建立的本体库中的某个词匹配时，说明用户也对该兴趣主题的直接父节点以及祖先节点感兴趣，只不过对它们的感兴趣程度有所差别。基于这个思想，用户模型构建中会自动查找并更新兴趣主题的父亲节点，直至根节点。

通过以上步骤就可以构建出微博用户模型了，包括用户兴趣树和兴趣度两大模块。在用户模型中，用户的兴趣表示为一棵用户兴趣的本体子树，如**错误!未找**

到引用源。所示。这棵本体子树包含的信息主要有：

（1）用户的兴趣主题

（2）用户不同的兴趣主题对应的兴趣度

（3）兴趣主题之间的层次结构关系

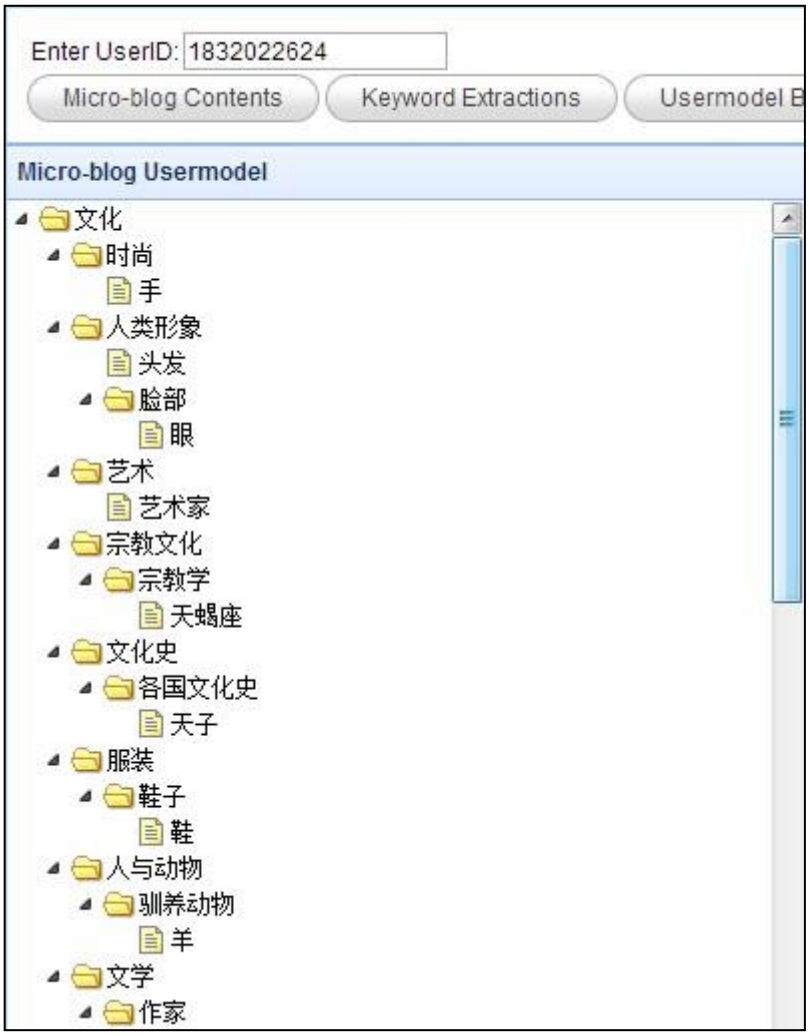

图 3-2 用户模型中一个用户的兴趣树

某个用户对于其所有微博中的每个特征词的兴趣度值已经过计算得到，在用户模型构建中，也将这部分内容在界面上显示出来，如图-3-3 所示。

User Interest Degree

uid	category_id	category_name	content weight
1832022624	166	时尚	0.45437430300924403
1832022624	290	童话	0.726487821524757
1832022624	297	学生	0.175502983341437
1832022624	689	头发	0.45437430300924403
1832022624	1026	艺术家	0.124876359684708
1832022624	1866	国宝	1.77058152685343
1832022624	3803	天蝎座	0.124876359684708
1832022624	3806	天子	0.401828430842653
1832022624	4025	小学	0.175502983341437
1832022624	4042	鞋	0.45437430300924403
1832022624	4051	心地	0.310565847008961
1832022624	4101	星座	0.124876359684708
1832022624	4144	学历	0.15885706886455397
1832022624	4150	学识	0.310565847008961
1832022624	4207	哑巴	3.5411630537068697
1832022624	4217	研究生	0.15885706886455397
1832022624	4226	眼	0.401828430842653
1832022624	4257	羊	0.124876359684708
1832022624	4276	药	0.401828430842653
1832022624	4517	月	0.15885706886455397
1832022624	4704	智商	0.401828430842653
1832022624	4719	中学	0.175502983341437

图-3-3 用户模型中一个用户对每个微博特征词的兴趣度

§3.4 本章小结

本章主要介绍了用户模型的基本概念以及常用用户模型和本体用户模型的表示方法。另外，详细地描述了微博用户模型构建方法的三大步骤，也是本文的最主要内容和贡献。

常用的三种用户模型表示方法缺乏语义和统一的标准。而基于本体的用户模型表示方法不仅能充分描述用户兴趣的语义，还具有兼容性和可扩展性，很好地解决了常用用户模型表示方法存在的问题，是用户模型表示方法的发展方向，因此本文将实现基于本体的用户模型构建。

本文采用基于词频统计的分词方法进行微博文本内容的特征词提取；基于TF-IDF 方法计算出微博特征词的权重，并相加得到其兴趣度；最后生成用户模型，显示出用户兴趣树以及用户对每个特征词的兴趣度。

第4章 微博用户模型构建系统设计与实现

本章主要介绍微博用户模型构建系统的设计与实现，包括系统整体设计、数据库设计以及系统内各功能模块的设计与界面展示。

§4.1 系统整体设计

§4.1.1 开发环境简介

本文中设计实现的用户模型评价系统使用的开发工具是 Java 开发平台 Eclipse 和关系型数据库 MySQL。

Eclipse 是一个开放源代码的软件开发项目，专注于为高度集成的工具开发提供一个全功能的、具有商业品质的工业平台。

基于 Java 的开发平台还有很多，比如 JBuilder 在以前比较流行，但它是收费的。它的优点在于可以拖拉窗体，比较适合桌面软件的开发。而 Eclipse 的优点在于它是开源的软件，不收费，并且拥有大量丰富的插件，现在大多数企业都用它来开发，是目前最流行的开发工具之一，所以本文选择 Eclipse 来进行微博用户模型的构建与展示。

MySQL 是一个小型关系型数据库管理系统。与其他的大型数据库例如 Oracle、DB2、SQL Server 等相比，MySQL 自有它的不足之处，如规模小、功能有限等，但是这丝毫也没有减少它受欢迎的程度。MySQL 灵活、小巧、便捷，对于一般的个人使用者和中小型企业来说，MySQL 提供的功能已经绰绰有余，而且 MySQL 也是开源软件。所以 MySQL 非常适合用于轻量级应用的开发，这也是本文选择 MySQL 作为数据库的原因。

§4.1.2 系统整体架构

系统整体将分为五个步骤来构建：

（1）分类本体库构建：基于维基百科的分类手动建立一个本体库。

（2）微博特征词提取：收集某一用户所发表的所有微博内容，并对每条微博进行分词操作，提取出特征词，计算得到其权重。

（3）主题兴趣度计算：将每个特征词在所有微博中出现的权重相加，得到这个用户对每个微博特征词的兴趣度。

（4）用户模型生成：将特征词与分类本体库进行匹配，得到这个用户的所有兴趣并形成一棵兴趣树。

（5）系统设计与实现：在系统界面上显示用户的兴趣树以及该用户对每个

微博特征词的兴趣度，构建完成一个微博用户的兴趣模型。

系统设计流程图如**错误!未找到引用源。**所示。

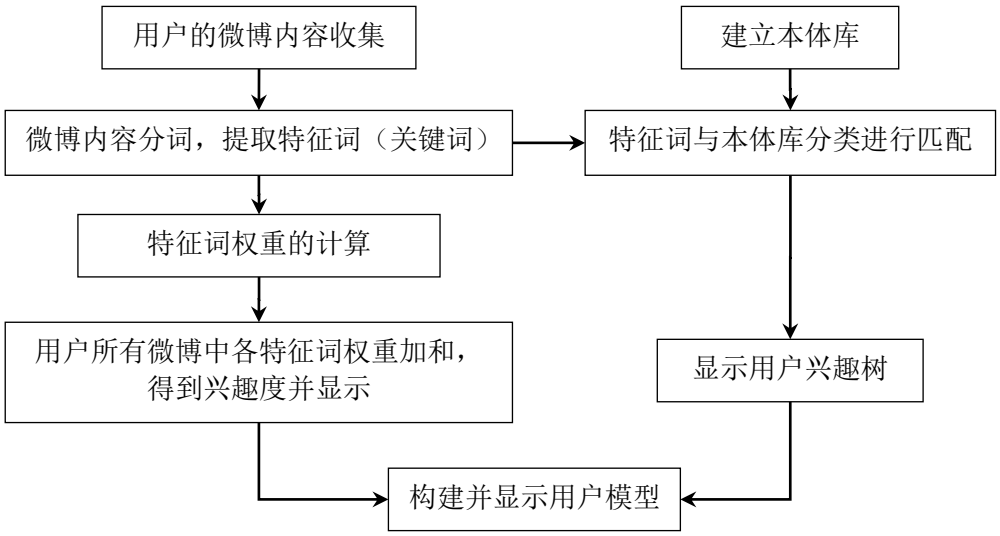

图 4-1 系统流程图

§4.2 数据库设计

数据库表的设计是依据本文第三章所描述的用户模型构建的三大步骤，创建了如下四张表格。

用户发表的所有微博的内容 s_weibo_content（如**错误!未找到引用源。**所示），包括的字段信息有用户编号 uid，该用户发表的微博编号 weibo_id，微博内容 content 等。

id	uid	weibo_id	content	comments_c...	reposts_c...
20	283358587	3527140630864894	转发微博:#抢占动漫微吧#我们在这	0	0
21	283358587	3527118824156292	转发微博:点开收取祝福：圣诞快乐	0	0
22	283358587	3526356388593505	//@早安丶晚安丶心语:[偷笑]你中了	0	0
23	283358587	3526352064182445	转发微博:各种罕见颜色的法拉利E	0	0
24	283358587	3526330564274498	//@早安丶晚安丶心语:正能量！:转	0	0
25	283358587	3526328748218136	@浮梦-YFP @芒果VS饺子 @烟妖精	4	0
26	283358587	3526126171600894	//@早安丶晚安丶心语:【史上最便	0	0
27	283358587	3526125450300986	//@早安丶晚安丶心语:人的一生，	0	0
28	283358587	3526124900603356	//@-iam黄馨玄:帅飞了//@Baidu俞	0	0
29	283358587	3526124812639831	转发微博:史上最全长尾夹妙用~~[	0	0
30	283358587	3523441132983997	@浮梦-YFP 嘿咻嘿咻，我想你了。	0	0
31	283358587	3522056429252280	//@两只老虎吃了尼姑欠:怎么那么	0	2
32	283358587	3522044660100002	//@浮梦-YFP: 你是我心底最珍贵的	0	0
33	283358587	3514735813719050	生日快乐哦[蛋糕][蛋糕][蛋糕]@Wal	0	1
34	283358587	3514454795178661	//@_李若琳: //@-iam黄馨玄: //@-g	0	0
35	283358587	3508569880218806	@胡陈纯422 生日快乐[蛋糕]	0	0
36	283358587	3507133737972993	405的孩子们，天冷了，要加衣服哦	6	0
37	283358587	3505437397751933	今天是个值得纪念的日子。祝愿我	10	0
38	283358587	3502961357982286	害怕参加校运会，害怕那种声音，	10	0
39	283358587	3501750412009534	亲爱的培培，生日快乐哦[蛋糕][蛋糕	0	1
136	1092338624	3532174323747664	[哀]法海你不懂爱...不懂爱.懂爱.爱	0	0
137	1092338624	3526766624981178	：我刚更新了#微博客户端冬季版#	0	0
138	1092338624	3520710590503921	[哈哈]:某校的雕塑，远看像是鲁迅	0	12
139	1092338624	3520710380763507	我一直有一个问题，明明我只关注	0	0
140	1092338624	3520696535281929	我已经抢"鲜"升级#新版微博#，想	0	2
141	1092338624	3491522491881399	我刚更新了#微博客户端秋季版#，	0	0
142	1092338624	3491347472797847	拿块豆腐撞了得了!糟透了。。。h	0	0
143	1092338624	3491343496316034	[赞]:《跳出棋盘的棋子》	0	0
145	1092338624	3491340006811781	@微博小秘书 麻烦你帮我清理掉被	3	0
146	1092338624	3491260117926094	好烦啊，好烦啊，这个工作从年初	2	0

图 4-2 微博内容数据表

对每条微博进行分词后所提取得到的特征词 s_feaword（如图 4-3 所示），包括的字段信息有用户发表的微博编号 weibo_id，从该微博提取到的特征词 feaword，以及该词所占的权重 weight。

Id	weibo_id	feaword	weight
68565	3527140630864894	祸害	0.496148603878051
68566	3527140630864894	节操	0.473698841561808
68567	3527140630864894	姐	0.360158635332544
68568	3527140630864894	此申请	0.569370719516428
68569	3527140630864894	机会	0.243864976641369
68570	3527140630864894	手机	0.25758797651822
68571	3527118824156292	图	1.5283258290017
68572	3526356388593505	心	0.196782645031837
68573	3526356388593505	作者	0.445976937934054
68574	3526356388593505	图	0.254720971500283
68575	3526356388593505	枪	0.399172503004772
68576	3526356388593505	事	0.25264286474035
68577	3526356388593505	80	0.385119022054766
68578	3526352064182445	颜色	2.49977036854864
68579	3526330564274498	漫画	0.561858074196097
68580	3526330564274498	能量	1.31017214987725
68581	3526330564274498	心	0.295173967547755
68582	3526328748218136	芒果	0.484056826019949
68583	3526328748218136	童话	0.445976937934054
68584	3526328748218136	妖精	0.553219050681752
68585	3526328748218136	烟	0.416628394758107
68586	3526328748218136	饺子	0.540022176340481
68587	3526328748218136	梦	0.320810437210716
68588	3526126171600894	心	0.147586983773877
68589	3526126171600894	非常低	0.502285538553316
68590	3526126171600894	投影仪	0.929313578190637
68591	3526126171600894	成本	0.383005224873401
68592	3526126171600894	牛	0.241490559718973
68593	3526126171600894	手机	0.38638196477733
68594	3526125450300986	家庭	0.279237147789625

图 4-3 特征词数据表

手动创建的基于维基百科的分类本体库 s_category（如图 4-4 所示），包括的字段信息有分类序号 ids，分类名称 name，每个分类所对应的父节点序号 parent_id。

ids	name	parent_id	relation
1	财经	0	2
2	IT	0	2
3	健康	0	2
4	体育	0	2
5	旅游	0	2
6	教育	0	2
7	招聘	0	2
8	文化	0	2
9	军事	0	2
10	金融业	1	1
11	银行	1	1
12	金融学	1	1
13	融资	1	1
14	经济	1	2
15	财政	1	2
16	投资	1	2
17	服务	1	3
18	国际金融保税仓库	1	1
19	地下金融	1	1
20	地下钱庄	1	1
21	计算机科学	2	2
22	电脑科学	2	2
23	计算	2	3
24	计算机编程	2	1
25	软件	2	2
26	人工智能	2	2
27	信息论	2	2
28	数据结构	2	2
29	最优化	2	2
30	算法	2	2

图 4-4 分类本体库数据表

表 4.1 分类本体库数据表

序号	姓名		父节点	关系
1	陈寅恪		0	2
2	鲁迅		1	2
3	章太炎		2	3

最终构建完成的用户模型 s_user_profile（如图 4-5 所示），包括的字段信息有用户编号 uid，特征词的分类序号 category_id（即对应表 s_category 中的分类序号 ids），该特征词的内容权重 contentweight 等。

Id	uid	category_id	contentweight	semanticweight
2772	1832022624	166	0.454374303009244	6.54720624390244e-010
2773	1832022624	290	0.726487821524757	1.43165576533333e-010
2774	1832022624	297	0.175502983341437	1.38547332129032e-011
2775	1832022624	689	0.454374303009244	3.27360312195122e-011
2776	1832022624	1026	0.124876359684708	9.92000946045825e-011
2777	1832022624	1866	1.77058152685343	6.39132038095238e-011
2778	1832022624	3803	0.124876359684708	1.00726249906191e-011
2779	1832022624	3806	0.401828430842653	5.81973888346884e-012
2780	1832022624	4025	0.175502983341437	1.84729776172043e-012
2781	1832022624	4042	0.454374303009244	2.00680651154098e-011
2782	1832022624	4051	0.310565847008961	1.12222180602007e-011
2783	1832022624	4101	0.124876359684708	1.27100121212121e-011
2784	1832022624	4144	0.158857068864554	2.88640275268817e-012
2785	1832022624	4150	0.310565847008961	2.88640275268817e-012
2786	1832022624	4207	3.54116305370687	1.29678964251208e-010
2787	1832022624	4217	0.158857068864554	5.36870912e-011
2788	1832022624	4226	0.401828430842653	5.644143313709e-012
2789	1832022624	4257	0.124876359684708	4.36480416260163e-012
2790	1832022624	4276	0.401828430842653	3.13959597660819e-011
2791	1832022624	4517	0.158857068864554	7.26286406926407e-012
2792	1832022624	4704	0.401828430842653	5.65127275789474e-011
2793	1832022624	4719	0.175502983341437	4.19840400391007e-012
2794	1832022624	4838	0.124876359684708	2.38609294222222e-010
2795	1832022624	4862	0.310565847008961	3.07134389016018e-011
2796	1832022624	5772	2.16702595970746	2.88640275268817e-012
2797	1832022624	5804	0.158857068864554	2.88640275268817e-012
2798	1832022624	6012	0.371725431276255	1.17889967501098e-011
2799	1832022624	6058	0.124876359684708	1.36400130081301e-011
2800	1832022624	6238	0.124876359684708	2.18240208130081e-011
2801	1832022624	6763	0.785175492248319	4.97102696296297e-011

图 4-5 用户模型数据表

§4.3 功能模块设计

系统界面的功能模块主要分为本体的构建与显示、本体的构建与显示和用户模型构建与显示三个部分。

§4.3.1 本体的构建与显示

基于维基百科的分类手动建立了 4757 条分类记录，初步形成了一个可作为研究基础的分类本体库，如**错误!未找到引用源。**所示。

图 4-6 基于维基百科的分类本体库

§4.3.2 用户基本数据显示

在系统界面上输入一个用户的 ID 号，就可以展示出该用户所发表的所有微博内容（如**错误!未找到引用源。**所示）以及从这些微博中提取得到的特征词和相应的权重（如**错误!未找到引用源。**所示）。

图 4-7 用户发表的所有微博内容

图 4-8 用户所有微博中提取的特征词及权重

§4.3.3 用户模型构建与显示

构建完成的用户模型显示分为两部分，左边为所查询用户的兴趣树，右边为对应计算得到的用户对其所有特征词的感兴趣程度，如**错误!未找到引用源。**所示。

图 4-9 用户模型的构建与显示

§4.4 本章小结

本章主要介绍了微博用户模型构建系统的整体设计及各功能模块的展示，简单介绍了开发环境及系统的架构，并展示了部分数据表的内容及演示界面的布局、数据等。

第5章 总结与展望

本章对全文的主要工作和创新点作了总结，并提出需要进一步研究和改进之处。

§5.1 本文总结

本文针对微博中存在的问题，对用户的微博内容自动进行分析，从而提取用户的兴趣，并建立微博用户模型，为微博信息推荐、舆情监控、微博营销等提供技术支持。

§5.1.1 本文的主要工作

本文主要研究的是基于本体的微博用户模型构建方法，主要工作内容有以下几个方面：

（1）分类本体库构建：基于维基百科的分类手动建立了 4757 条分类记录，初步形成了一个可作为研究基础的分类本体库。

（2）微博特征词提取：收集某一用户所发表的所有微博内容，并对每条微博进行分词操作，提取出特征词，计算得到其权重。

（3）主题兴趣度计算：将每个特征词在所有微博中出现的权重相加，即得到了这个用户对每个微博特征词的兴趣度。

（4）用户模型生成：将特征词与之前建立的分类本体库进行匹配，得到这个用户的所有兴趣并形成一棵兴趣树。

（5）系统设计与实现：在系统界面上显示用户的兴趣树以及该用户对每个微博特征词的兴趣度，即构建完成了一个微博用户的兴趣模型。

§5.1.2 本文的主要创新点

本文基于本体的微博用户模型构建方法创新点主要有以下两项：

（1）实现了利用微博信息分析用户兴趣方法。

（2）实现了利用本体构建用户模型的方法。

§5.2 展望

虽然本文实现了基于本体的微博用户模型构建，但仍存在不少需要改进的地方：

（1）本体的完备性和微博的短文本对用户模型的准确性影响较大。由于手动建立的本体库有限，仅能作为初步研究工作的基础，但为了下一步研究的准确性更上一层楼，本体库的数据需要大量的增加，维护数据库的成本将会变得非常

高。而短文本又是微博的一大特点，在 140 字以内的短文本中提取有效的特征词变得更加不易。

（2）本文对特征词所做的匹配仅限于其内容，但是中文中写法相同却有着截然不同含义的词汇有很多，因而会对用户模型的准确性带来考验。而若是加上语义的因素，难度又将增加不少。

致谢

XXX。

参考文献

[1] 林鸿飞,杨元生.用户兴趣模型的表示和更新机制.计算机研究与发展，200239(7):838-842.

[2] 胡学联,潘金贵,李俊,张灵玲. 一个个性化的信息搜集 Agent 的设计与实现.软件报,2001,12(7):1074-1079.

[3] 应晓敏,面向 Internet 个性化服务的用户建模技术研究. 2003,国防科技大学:长沙

[4] 徐振宁,张维明,陈文伟. 基于 Ontology 的智能信息检索. 计算机科学,20.28(6):21-26.

[5] 李勇.智能信息检索中基于本体的个性化用户建模技术及应用. 2002,国防科技学:长沙.

[6] FragoudisD.User Modeling in Information Discovery:An overview.InProceedings of Advanced Course on Artificial Intelligence,1999(ACAI99), July,1999, Greece.

[7] Balabanovie M, Shoharn Y.Learning Information Retrieval Agents:Experiments with Auto-mated Web BrowsingIn:Proceedings of the AAAISpring Symposium Series on Information Gathering from Heterogeneous, Distributed Environments, March, 1995:13-18.

[8] Lieberman H.Letizia. An Agent that Assists Web Browsing.In:Proceedings ofthe International Joint Conference on Artificial Intelligence, Montreal, August, 1995:924-929.

[9] Chan P.K.A Non-Invasive Learning Approach to Building Web User Profiles in KDD-99Workshop on Web Usage Analysis and User Profiling, 1999, New York:ACMpress.

[10] Schwab L,Kobsa A,and Koychev I. Learning about User from Observation[J].in AAAISpring Symposium on Adaptive User Interface, 2000, Standord,California:AAAI Press.

[11] Adomavicius G and Tuzhilin A.Using Data Mining Methods to Build CustomerProfiles.IEEE Computer.Feb 2001:74-82.

[12] 余伟. 基于本体的微博客用户行为模型研究[J]. 广东技术师范学院学报, 2010, 31(006): 27-30.

[13] 赵岩露, 王晶, and 沈奇威. "基于特征分析的微博用户兴趣发现算法." 电信工程技术与标准化 25.11: 79-83.

[14] Uschold M. and Gruninger M., Ontologies: Principle, Methods and Application. Knowledge Engineering Review, 1996, 11(2): p.93-155.

[15] Neches R, et al. Enabling technology for knowledge sharing.AI Magazine, 1991, 12(3): p.36-56.

[16] Swartout W. Ontologies, 1999. Intelligent Systems and their Applications, IEEE. Issue:1, p.18-19.

[17] Gruber T.R.,A Translation Approach to Portable Ontology Specifications .knowledge acquisition, 1993, 5(2): p.199-221.

[18] Borst W.N. Construction of Engineering Ontologies for Knowledge Sharing and Reuse. 1997, University of Twente: Enschede.

[19] Studer R, Benjamins VR, and Fensel D. Knowledge Engineering: Principles

and Methods. Data and Knowledge Engineering, 1998, 25(1-2): p.161-197.
[20] 朱晓冰，寇雅楠基于维基技术的本体构建方法探讨图书馆学研究 2009.1: p.55-58.
[21] 于江生，俞士汝.中文概念词典的结构.中文信息学报.2002 年第 4 期: p.64-68.
[22] 由丽凭，杨翠.汉语框架语义知识库概述电脑开发与应用.2007 年 06 期: p.72-74.
[23] 张晶，姚建民，赵铁军，李生.基于 WordNet 和 HowNet 建设双语语义词典.高技术通讯.2001.12(2-3): p.43-46.
[24] 侯汉清，薛春香.用于中文信息自动分类的《中图法》知识库的构建.中国图书馆学报.2005 年第 5 期: p.67-70.
[25] 罗志成，马费成，吴晓东，宋倩倩.从维基分类系统构建中文语义词典研究.信息系统学报.2008 年 02 期: p.35-39.
[26] 杨玲贤.基于 ontology 的教学资源知识库构建.计算机与现代化.2009 年第 11 期: p.27-32.
[27] Fromkin V, Rodman R. Introduction to Language. London: Holt, Rinehart and Winston, Inc., 1988, 12(1): p.10–16.
[28] Brachman R, What IS-A is and isn't: An analysis of taxonomic links in semantic networks [J]. IEEE Computer, 1983, 16(10): p.30–36.
[29] Fischer G. User modeling in human–computer interaction[J]. User modeling and user-adapted interaction, 2001, 11(1-2): 65-86.
[30] 王金花. 一种利用本体关联度改进的 TF-IDF 特征词提取方法[D]. 河北大学, 2011.
[31] 中国科学院计算技术研究所[EB/0L]. 中文自然语言处理开放平台. http://www.nlp.org.en/project/ project.php?proj_id=6, 2005-2-2.
[32] Scott Deerwester, Susan T Dumais, Furnas W George, et al. Indexing by latent semantic analysis[J]. Journal of the American Society for Information Science, Vol 41, 1990.
[33] 王娟琴. 三种检索模型的比较分析研究—布尔模型、向量模型、概率模型[J]. 情报科学, 1998, 16(3): 225-231.
[34] 景玉峰等. 概率检索模型.现代图书情报技术, 1987(1): 29-31.
[35] 冀胜利, 李波. 基于 SVM 的中文文本分类算法[J]. 重庆工学院学报(自然科学), 2008, Vol.22 No.7: 84-87.
[36] 崔争艳. 中文短文本分类的相关技术研究[D].
[37] 埃克尔著, 陈昊鹏译. Java 编程思想[M]. 第 4 版. 北京: 机械工业出版社, 2007.
[38] 李刚. 疯狂 Java 讲义[M]. 第 2 版. 北京: 电子工业出版社, 2008.
[39] 李刚. 轻量级 Java EE 企业应用实战:Struts2＋Spring3＋Hibernate 整合开发[M]. 第 3 版. 北京: 电子工业出版社, 2011.
[40] Dou Shen, Jian-Tao Sun, Qiang Yang, et al. Latent Friend Mining from Blog Data[C]. Sixth International Conference on Data Mining, 2006: 552-561.
[41] Katarzyna Musiał, Przemysław Kazienko, and Tomasz Kajdanowicz. Social Recommendations within the Multimedia Sharing Systems[C]. First World Summit on the Knowledge Society, 2008: 364-372.

附录：部分源程序清单

```
//1. 分词、特征词提取
publicvoid Feaword()throws Exception
    {
        List<S_user> userlist =findusers();
        Map<String,Integer> globalmap= newHashMap<String,Integer>();//存放
全部微博特征词
        int weibo_count=0;
        for(int u=0;u<userlist.size();u++)
        {
            String uid=userlist.get(u).getUid();
            List<S_weibo_content> weibolist=findByUid(uid);
            if(weibolist==null||weibolist.size()==0)
            {
            }
            else
            {
                weibo_count=weibo_count+weibolist.size();
                for(int w=0;w<weibolist.size();w++)
                {
                    S_weibo_content sweibo=weibolist.get(w);
                    //单条微博开始
                    Map<String,Integer> currentmap = new
HashMap<String,Integer>();//存放单条微博特征词
                    SplitWord splitWord = SplitWord.getInstance();
                    splitWord.init();
                    int count=0;
                    String str=sweibo.getContent();
                    str=str.replaceAll(" ", "");
                    String result=null;
                    result=splitWord.split(str);
                    //System.out.println(result);
                    String[] rr=result.split(" ");
                    for(int i=0;i<rr.length;i++)
                    {
                        String[] rr1=rr[i].split("/");
                        if(rr1[1].equals("n"))
                        {
                            String noun=rr1[0];
                            count++;
                            if(currentmap.get(noun)==null)
                            {
                                currentmap.put(noun, new Integer(1));
                            }
                            else
                            {
                                Integer number = currentmap.get(noun);
                                number++;
                                currentmap.put(noun, newInteger(number));
                            }
                        }
```

```
                    }
                    Iterator iter = currentmap.entrySet().iterator();
                    while (iter.hasNext())
                    {
                        Map.Entry<String,Integer> entry =
(Map.Entry<String,Integer>) iter.next();
                        String key=entry.getKey();
                        if(globalmap.get(key)==null)
                        {
                            globalmap.put(key, new Integer(1));
                        }
                        else
                        {
                            Integer number=globalmap.get(key);
                            number++;
                            globalmap.put(key, new Integer(number));
                        }
                    }
                }//单条结束
            }
            System.out.println("用户"+uid+"的微博分词结束");
        }
        System.out.println("总微博数 weibo_count="+weibo_count);


//2. 特征词权重计算
for(int u=0;u<userlist.size();u++)
        {
            String uid=userlist.get(u).getUid();
            List<S_weibo_content> weibolist=findByUid(uid);
            if(weibolist==null||weibolist.size()==0)
            {
            }
            else
            {
                for(int w=0;w<weibolist.size();w++)
                {
                    S_weibo_content sweibo=weibolist.get(w);
                    Map<String,Integer> currentmap = new
HashMap<String,Integer>();
                    SplitWord splitWord = SplitWord.getInstance();
                    splitWord.init();
                    int count=0;
                    String str=sweibo.getContent();
                    str=str.replaceAll(" ", "");
                    String result=null;
                    result=splitWord.split(str);
                    System.out.println(result);
                    String[] rr=result.split(" ");
                    for(int i=0;i<rr.length;i++)
                    {
                        String[] rr1=rr[i].split("/");
                        if(rr1[1].equals("n"))
                        {
```

```
                        String noun=rr1[0];
                        count++;
                        if(currentmap.get(noun)==null)
                        {
                            currentmap.put(noun, new Integer(1));
                        }
                        else
                        {
                            Integer number = currentmap.get(noun);
                            number++;
                            currentmap.put(noun, new Integer(number));
                        }
                    }
                }
                Iterator iter2 = currentmap.entrySet().iterator();
                while (iter2.hasNext())
                {
                    Map.Entry<String,Integer> entry2 =
(Map.Entry<String,Integer>) iter2.next();
                    String key=entry2.getKey();
                    double
tf=(double)currentmap.get(key)/(double)count;
                    System.out.println("count="+count);
                    System.out.println("tf="+tf);
                    double
idf=(double)weibo_count/(double)globalmap.get(key);
                    idf=Math.log10(idf);
System.out.println("globalmap.get(key)="+globalmap.get(key));
                    System.out.println("idf="+idf);
                    double weight=tf*idf;
                    S_feaword ff=new S_feaword();
                    ff.setSweibo(sweibo);
                    ff.setFeaword(key);
                    ff.setWeight(weight);
                    saveFeaword(ff);
                }
            }
        }
    }
```

//3.用户模型兴趣树的显示

```
<ul class="easyui-tree">
    <s:iterator value="#request['menus']" id="c2">
    <s:if test="#c2.parent_id == 0">
    <li data-options="state:'closed'">
    <span><s:property value="#c2.name" /></span>
    <ul>
        <s:iterator value="#request['menus']" id="c3">
        <s:if test="#c3.parent_id == #c2.ids">
        <li>
        <span><s:property value="#c3.name" /></span>
        <ul>
            <s:iterator value="#request['menus']" id="c4">
```

```
            <s:if test="#c4.parent_id == #c3.ids">
            <li>
            <span><s:property value="#c4.name" /></span>
            <ul>
                <s:iterator value="#request['menus']" id="c5">
                <s:if test="#c5.parent_id == #c4.ids">
                <li>
                <s:property value="#c5.name" />
                </li>
                </s:if>
                </s:iterator>
            </ul>
            </li>
            </s:if>
            </s:iterator>
        </ul>
        </li>
        </s:if>
        </s:iterator>
    </ul>
    </li>
    </s:if>
    </s:iterator>
</ul>


//4. 用户兴趣度计算
public void contentinterest() throws Exception
    {
        List<S_user> userlist = findByuid("283358587");
        //读取所有用户
        if(userlist==null||userlist.size()==0)
        {
        }
        else
        {
            //读取所有本体类别
            List<S_category>
categorylist=ms_categoryDao.getS_categoryname();
            if(categorylist==null||categorylist.size()==0)
            {
            }
            else
            {
                for(int u=0;u<userlist.size();u++)
                {
                    //读取某一个用户的所有微博
                    String uid=userlist.get(u).getUid();
                    List<S_weibo_content> weibolist=findByUid(uid);
                    if(weibolist==null||weibolist.size()==0)
                    {
                    }
                    else
                    {
```

```
                        for(int c=0;c<categorylist.size();c++)
                        {
                        //判断任意一条微博在每一个分类上的权重，如有则类别兴趣累
计
                            Double wc=(double) 0;
                            Double ws=(double) 0;
    System.out.println(weibolist.get(w).getWeibo_id());
    feaword=findBywid(s_weibo_content.getWeibo_id());//找到微博 w 的所有特征
词记录
                                List<S_feaword>
feaword=findBywfid(categorylist.get(c).getName(),uid);
                                if(feaword==null||feaword.size()==0)
                                {
                                }
                                else
                                {
                                    for(int f=0;f<feaword.size();f++)
                                    {
    wc=wc+feaword.get(f).getWeight();
    ws=ws+categorylist.get(c).getScd();
                                    }
                                }
                            if(wc>0)
                            {
                                S_s_user_profile      user_profile=      new
S_s_user_profile();
                            user_profile.setUser(userlist.get(u));
    user_profile.setCategory(categorylist.get(c));
                                user_profile.setContentweight(wc);
                                user_profile.setSemanticweight(ws);
                                try
                                {
                                s_UserProfileDao.save(user_profile);
                                }catch(Exception e)
                                {
                                    e.printStackTrace();
                                }
                            }
                        }
                    }
                }
            }
        }
        System.out.println("用户完");
    }
```

//5. 用户兴趣度显示

```
<table id="tt"class="easyui-datagrid"style="width:auto;">
    <thead>
    <tr>
    <thdata-options="field:'uid',width:150"align="center">uid</th>
    <thdata-options="field:'s_category.ids',width:100"align="center">cate
```

```
gory_id</th>
    <thdata-options="field:'s_category.name',width:150"align="center">cat
egory_name</th>
    <thdata-options="field:'contentweight',width:200"align="center">conte
nt weight</th>
    </tr>
    </thead>
    <s:iteratorvalue="#request['plansearch']"id="c2">
    <tr>
    <td><s:propertyvalue="#c2.uid"/></td>
    <td><s:propertyvalue="#c2.category_id"/></td>
    <td><s:propertyvalue="#c2.category_name"/></td>
    <td><s:propertyvalue="#c2.contentweight"/></td>
    </tr>
    </s:iterator>
</table>
```