同济大学
TONGJI UNIVERSITY

硕士学位论文

(专业学位)

一种基于 P2P 网络的信息自主流动机制的研究

姓　　名：　先毅昆

学　　号：　1236191

所在院系：　软件学院

职业类型：

专业领域：　软件工程

指导教师：　张晨曦

副指导教师：奚自立

二〇一五年三月

同濟大學
TONGJI UNIVERSITY

A dissertation submitted to

Tongji University in comformity with the requirements for

the degree of Master of Computer Science

Study of the Mechanism of Information Spontaneous Propagation on P2P Network

Candidate:	Yikun Xian
Student Number:	1236191
School/Department:	School of Software Engineering
Discipline:	
Major:	Software Enginnering
Supervisor:	Chenxi Zhang

March, 2015

一种基于P2P网络的信息自主流动机制的研究

先毅昆 同济大学

学位论文版权使用授权书

本人完全了解同济大学关于收集、保存、使用学位论文的规定，同意如下各项内容：按照学校要求提交学位论文的印刷本和电子版本；学校有权保存学位论文的印刷本和电子版，并采用影印、缩印、扫描、数字化或其它手段保存论文；学校有权提供目录检索以及提供本学位论文全文或者部分的阅览服务；学校有权按有关规定向国家有关部门或者机构送交论文的复印件和电子版；在不以赢利为目的的前提下,学校可以适当复制论文的部分或全部内容用于学术活动。

学位论文作者签名：

年　　月　　日

同济大学学位论文原创性声明

本人郑重声明: 所呈交的学位论文, 是本人在导师指导下, 进行研究工作所取得的成果。除文中已经注明引用的内容外, 本学位论文的研究成果不包含任何他人创作的、已公开发表或者没有公开发表的作品的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体, 均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

学位论文作者签名：

年　　月　　日

摘　　要

应概括地反映出本论文的主要内容, 包括工作目的、研究方法、研究成果和结论, 要突出本论文的创造性成果。摘要力求语言精炼准确, 硕士学位论文建议 1000 字以内, 博士学位论文建议 3000 字以内。摘要中不要出现图片、图表、表格或其他插图材料。

学位论文原则上应用汉语撰写, 对于用汉语授课并享受中国政府奖学金的博士硕士留学研究生, 学位论文如用英语 (德语、法语) 撰写, 硕士学位论文不少于 3000 汉字摘要, 博士学位论文不少于 5000 汉字摘要; 对于其他情况 (含用英语授课) 的博士硕士留学研究生, 学位论文如用英语 (德语、法语) 撰写, 可不要求撰写汉语摘要, 但必须有英语摘要。

关键词是为了便于文献索引和检索工作, 从论文中选取出来用以表示全文主题内容信息的单词或术语, 摘要内容后另起一行标明, 一般 3 5 个, 之间用“,”分开。

关键词：关键字，摘要

Abstract

First Paragraph

SecondParagraph

Tongji University, located in Shanghai, has more than 50,000 students and 8,000 staff members (as of 1 September 2007). It offers degree programs at both undergraduate and postgraduate levels. Established in 1907 by the German government together with German physicians in Shanghai, Tongji is one of the oldest and most prestigious universities in China. Among its various departments it is especially highly ranked in engineering, among which its architecture, urban planning, and civil engineering departments have consistently ranked first in China for decades, and its automotive engineering, oceanography, environmental science, software engineering, German language departments are also ones of the best domestically.

Kew Words: English Keywords, Abstract

Contents

第 1 章 绪论

1.1 选题背景

互联网技术的蓬勃发展在很大程度上给人们的日常生活带来了越来越多的新鲜与便利。人们在逐渐适应网络信息这样的平台的同时，也更倾向于甚至更依赖于通过网络平台来完成生活中的各种事务。可以说，网络对于人们的重要性几乎已经等同于空气、水和食物。在现今的网络框架下，按照网络平台的功能来划分，门户网站（新浪、搜狐等）是新闻实事评论发布的主要渠道，社交网站（微博、人人等）是人们分享个人想法、心情，进行虚拟社交的首选，博客系统（新浪博客、百度空间等）是人们发表、传达思想和经验的主要平台。除此之外，一些新新涌现的以图片和视频等多媒体资源为载体的分享平台（优酷土豆、POCO等）也正在极大地丰富与改变人们的娱乐生活。另外，随着国内物流行业的兴起和发展，各大电子商务平台（淘宝、京东等）也使得人们足不出户便可享受到温馨且迅捷的购物体验。按照网络平台资源的载体来划分，总共分成文本、图片、音乐和视频几大类。其中文本无疑是整个互联网资源的主体。无论是传统的新闻、博客、评论、说明，还是新新发展的弹幕视频网站（Acfun[1]、Bilibili[2]等），都是由大数据量的文本信息组成。按照网络平台的用户参与角度来划分，可以将用户角色分为两种：信息发布者和信息获取者。以程序员为例，当他需要学习一项新技术时，往往会通过搜索引擎去寻找一些与该技术相关的教程与经验文章，从而使自己尽快掌握该项技术。在对该技术熟练掌握并积累一定的实践和经验后，往往又会通过写技术博客的方式记录下他的学习心得和学习中所碰到的困难以及相对应的解决方法供他人参考。同时，该用户还悉数拥有其他多个兴趣爱好，比如他喜欢与网上的其他用户分享旅游日记和自己的摄影作品等等。

虽然这些网站系统在功能上、信息载体上或是用户参与方式上都截然不同，但是宏观来看他们都存在一个共同的问题：信息孤岛现象。如图 1.1所示，在这些网站发展得越来越多，越来越复杂的同时，各个网站之间信息不流通的问题也日渐明显。出于安全性、管理和竞争策略等方面的考虑，每个网站独立发展，其资源和用户信息并不实现跨平台共享，从而使得每个网站成为一座座信息孤岛，信息的流通，传播以及发展得到了阻碍。

举例来说，当用户作为信息获取者时，虽然每个网站可以为本平台上的所有用户提供很好的用户体验：通过时下流行的数据挖掘等技术发现用户的兴趣，为其推荐有潜在需求的信息，但是不同网站之间的用户兴趣不能共享，这将直接导致兴趣推荐的不准确甚至误推，影响客户体验。当用户作为信息发布者时，其所需要的操作会更加繁琐复杂。用户往往需要打开多个网站重复几乎相同的操作

[1]http://www.acfun.tv/
[2]http://www.bilibili.com/

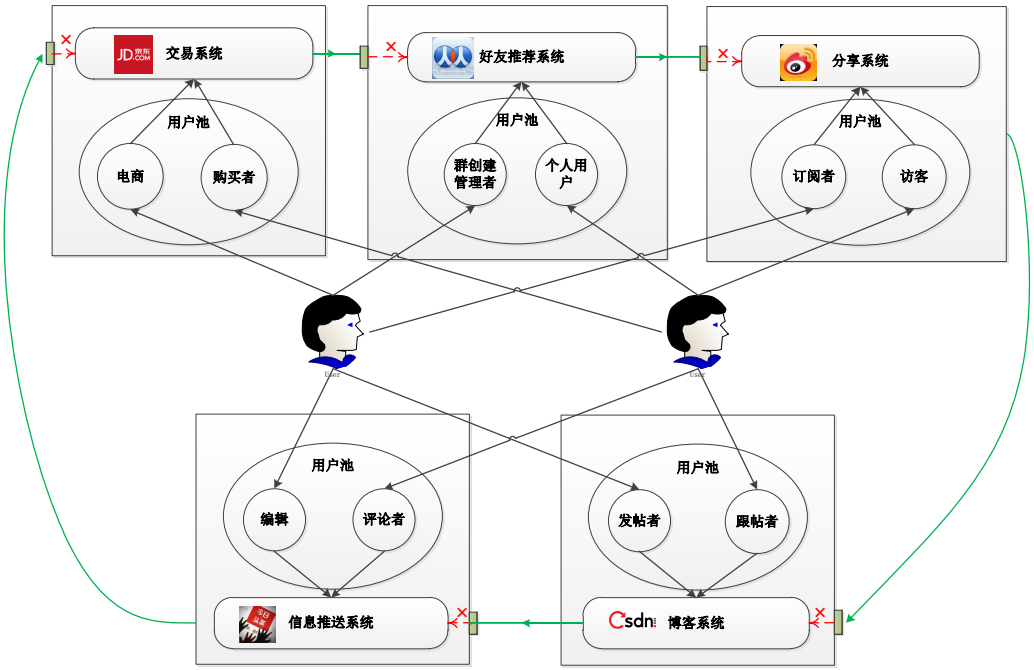


图 1.1: 互联网平台的信息孤岛现象

来发布同样的内容。最明显的证据就是同时使用微信、微博和人人的用户需要在三个平台上重复三次操作完成发布。当这些网站越来越多的时候，问题也随之而来，即用户可能需要打开很多个独立的网站来完成一系列类似的操作。比如，先在新闻网站上浏览最新发生的时事，接着在摄影网站上发布新照片，然后在社交网站上浏览好友更新的状态，最后在电商网站上购买一些商品等等。换句话说，用户是单向地去寻找或发布信息，整个流程始终由用户作为发起者向下进行，这其中无疑包含了一些不必要的重复劳动。从某种程度上来说，即使存在某些用户只在社交网站上浏览信息，很少接触其它网站，也会有一些重复操作的问题。因为该用户的好友很有可能是分散地活跃于各个不同的社交网站平台 (微博、人人网等)，并且每个平台发布的信息也有所不同，所以该用户仍然需要逐个登录各个网站后才能浏览到他朋友们的信息和状态。退一步说，即便现在已经有些软件把所有社交网站整合成一个统一接口，让用户只需一次登录就能同时访问多个平台，用户仍然会遇到信息冗余的问题，比如重复的新闻、不感兴趣的推荐等等。

为了解决上述问题，现在设想有这样一个智能系统：每当用户打开系统时，系统会自动推送今天的时事新闻、其好友最近更新的状态、感兴趣或者正在促销的商品，以及一些根据用户偏好过滤的信息。此外，他也可以在系统上发布自己的信息给他的好友，甚至给那些对他信息感兴趣的陌生人。虽然，实现这样一个系统的工作量和难度是巨大的，但仔细观察后可以发现这样一个重要的规律：

即用户希望所有的信息能够在整个互联网上智能地自主流动，在用户单方面寻找信息的同时，让信息也能自主地流向符合特定需求的用户。

从抽象层面来看，要实现这样一套信息自主流动的机制，传统的集中式计算模式已经不再适用。如图 1.2所示，这里每个用户均看作一个独立的节点，所有的节点整合在一起就形成一个巨大的 P2P 网络。其中，每个节点既充当服务器用于分发信息，也充当客户端用于接收信息，并且由某个节点发出的信息在其它节点间传播的时候会自主流动，寻找潜在的、匹配的节点。简单来说，要使信息能够自主流动，就要攻克这几方面的难点：

- 资源信息描述与匹配：互联网资源普遍呈现出半结构化或非结构化的特征。以普通的新闻或者博客为例，除了作者、标题、时间等结构化信息，正文纯文本都是典型的非结构数据。因此，如何从这些非结构化的文本信息中提取出有用的特征、并用这些特征来衡量这些文本之间的相关性是本文重要研究内容。

- 用户兴趣挖掘与关联：用户的兴趣可以从其发布和查看的信息中反映出来，基于上述对资源信息的描述模型，可以训练出某一时刻或者某一时段内的用户兴趣。但是，由于用户兴趣可能会随着时间的推移而不断变化，并且在同一时刻可能同时存在长期兴趣和短期兴趣。因此，这部分需要基于资源模型来解决用户兴趣随时间变化的问题。

- P2P 网络路由与搜索：从整个网络结构来看，在初始状态下各个用户节点之间的关系仅仅为物理意义上的距离关系，换句话说，节点与节点之间的边的权重不能代表用户与用户之间兴趣的相似度，从而也就导致每个节点上的信息不能自由传播。因此，这部分需要借助用户兴趣模型，通过 P2P 网络的路由搜索算法来寻找潜在的相似节点，最后构成某一时间段内的稳定结构。

1.2 研究意义与应用价值

对于上述提出的信息孤岛现象是主要由目前网站之间相互独立而导致的，这些中心化结构的独立网站架构存在以下几个问题：

- 从资源传播的角度来看，每个网站的数据都仅仅在一个局部范围内可用，但是用户却要在多个这样的局部范围内同时使用多个网站。虽然对于每个网站可以维护相对较好的资源整合和管理，但是对于属于用户的资源的传播带来了很大的阻碍。

- 从计算资源的角度来看，每个独立的传统 B/S 架构的网站通常由背后的一套计算群组来支撑，当网络流量十分巨大的时候，服务器的硬件和软件性

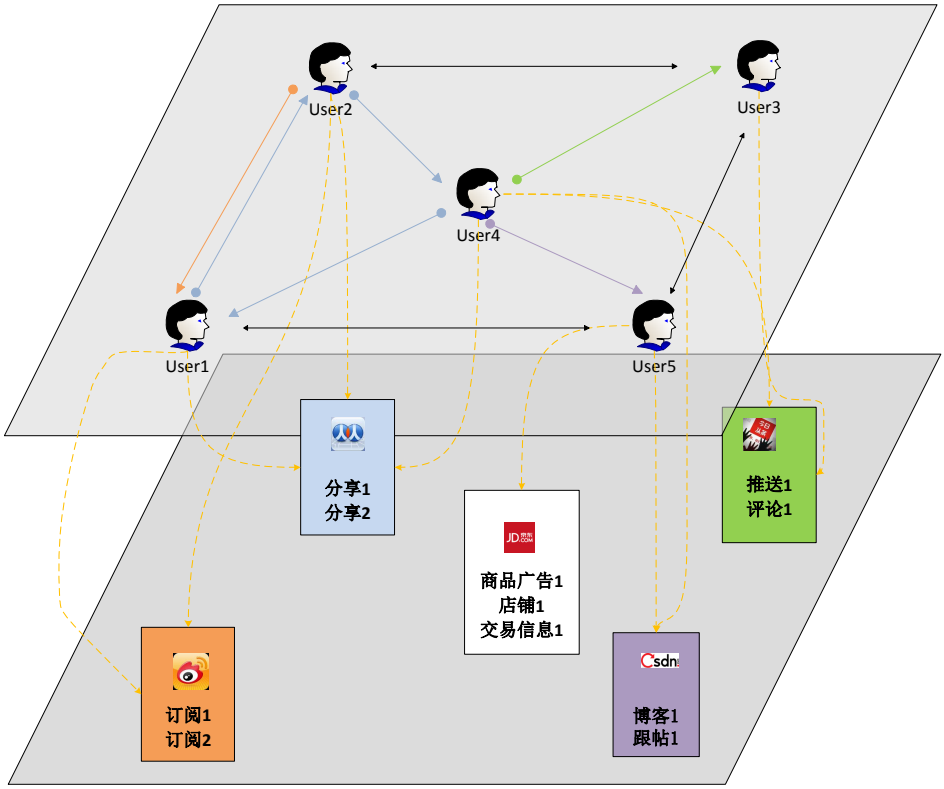


图 1.2: 一种基于 P2P 网络的智能系统解决方案

能决定了系统平台可承受的最大程度。而在用户这边的客户端的 I/O 消耗相对小很多，但是大量客户端的请求对于服务器是一个不小的考验，从而导致了服务器成为整个系统的计算瓶颈。

- 从隐私安全的角度来看，目前网站的数据均由平台自己的数据库进行维护，对于数据泄露成为重大隐患之一，即使网站的安全性是非出色，用户也会担心其数据是否会被第三方利用于商业用途。

- 从拓扑架构的角度来看，用户与网站系统组成了中心化的拓扑结构，该结构不利之处在于当网站服务器宕机或者网络中断的时候，所有用户均无法访问该系统，这就使得网站系统的备份机制与主从切换机制十分完善，而对于大部分初创公司的流量有限和资金不足等特征是一个很大的考验。

与中心化结构网站架构不同，在基于 P2P 网络架构中，每个节点都可以既可以充当服务器为其它节点提供资源，也可以作为客户端向其它节点获取资源。综合来说，P2P 网络的应用具有以下几大优点 [1]：

- 非中心化（Decentralization）：在 P2P 网络中，节点之间的通信无需经过中心服务器，因此大大降低了服务器因为计算资源问题而导致的性能瓶颈。同

时，网络中的信息资源完全存储在各个节点中，信息的交互完全在节点之间进行。

- 可扩展性（Scalability）：由于 P2P 网络中的每个节点可以看做服务提供者，所以每当有新的节点加入时，相当于扩充了整个网络的服务能力和资源，相反用户节点也可以随时退出，十分的灵活。以网络下载资源为例，普通 C/S 模式的计算下载通道的传输能力基本依赖于服务器的带宽，当用户一旦很多的时候，下载速度往往会受到很大的影响，但是在 P2P 网络中却恰恰相反，更多的节点请求意味着有更多的资源提供者，从而下载速度也更快。
- 健壮性（Robustness）：与中心化架构模式不同，P2P 网络不会因为某一个节点退出而在很大程度上影响网络的正常运行，相反只有很小的一部分影响。结构良好的 P2P 网络一旦在某几个节点发生异常的时候会自动调整拓扑结构，从而保证整个网络的连通性。
- 安全性（Security）：传统集中式的计算模式在通信上往往需要经过中心服务器，也就是说用户的数据信息会全部经过这些服务器，因此只要在这些为数不多的服务器上动些手脚，用户隐私就会被泄露。但是，在 P2P 网络上则不同，信息传输不会经过某些特定的节点，并且还可以与特定有需求的目标节点建立连接，从而在很大程度上提升了匿名通信的可靠性以及灵活性。
- 负载均衡（Load-Balance）：根据上文所述，P2P 网络下的用户节点既能充当信息发布者的角色，也可以作为信息接受者的角色，所以在计算资源和存储资源的分布十分均匀。同时，对于信息接受者来说，其可以向计算资源空闲的节点请求资源，而不用排队等待那些繁忙的节点直到任务完成。

基于上述有点，本文首先建立一套用于描述互联网信息和用户行为偏好的模型，旨在将这些具体的信息和用户行为偏好上升到抽象层面，剥离出一个统一的输入输出接口。对于那些专注于复杂的数据分析和建模工作的模块，该接口起到了整合的作用。而且，在此之后，也可以展开专门基于该描述模型的信息与用户匹配算法的研究。其次，根据上述信息描述模型与用户行为偏好模型的特点，从拓扑结构，通信机制等方面入手，建立一套合适的分布式架构 (如 P2P 的计算模式)，并利用上述的匹配算法，研究由各个节点组成的覆盖网络间的信息自主流动的机制。最后，将上述两项内容结合在一起，实现一套完整的信息自主流动的原型系统。

1.3 国内外研究现状

从目前的科学研究和商业应用方面来看，还没有关于一套基于 P2P 网络的信息自主流动机制的相关工作。因此，这部分将从文本信息挖掘、用户兴趣关联以

及 P2P 网络搜索等三方面来具体展开阐明已有的研究工作。

1.3.1 文本挖掘相关工作

在文本信息挖掘方面，由于当今的硬件和软件技术在各方面都有大幅度提高，产生了大量的不同类型的数据资源 [2]，特别是纯文本的数据。这些文本大多来自于社交网络、新闻博客等网站，内容形式对于人们来说也是十分的丰富。但是这对机器来说却没有那么容易识别，因此需要设计一系列的算法来从这些文本数据中找出一些模式和规律，从而让机器更好地利用这些数据为人们创造更大的价值。结构化数据（Structured）通常可以存储在各类数据库中，但是纯文本数据却因为它的半结构化（Semi-structured）或者非结构化（Unstructured）特性从而只能有通过搜索引擎等技术才能进行索引和查找 [3]。搜索引擎是信息检索（IR）中的一种方式，其作用是方便用户直接通过输入关键词找到内容相关的文档集合，其目标是如何通过有效和高效的方法是检索的结果更加精确，涉及的研究领域包括文本聚类、文本分类、文本概括、推荐系统等 [4--6]。但是，在本文中，除了这些信息检索基本的需求，更重要的是从文本中挖掘出重要的特征和模式，将原本非结构化的数据量化成半结构化数据，并且这种量化过程更多地从用户兴趣这点切入的。针对这个要点，文本挖掘技术一般可以分为以下几大类：

1.3.1.1 文本抽取

文本抽取（Text Extraction）主要是从半结构化或结构化的纯文本里找出结构化信息，涉及了自然语言处理（NLP），信息检索和 Web 挖掘（Web Mining）等领域的技术。该研究最基本的两大工作是：a）命名实体检测（NER），包括找出文本中的人物、地点、组织等等；和 b）关系抽取（Relation Extraction），主要包括名字之间的地理位置关系、人物关系、动作执行关系等等。

其中，对于命名实体检测问题，主要包括基于规则的方法 [7, 8] 和基于统计学习的方法，后者有三个十分著名和常用的方法：隐马尔可夫模型（HMM）[9]，这是一个简单且有效的生成模型（Generative Model），最大熵马尔可夫模型（MEMM）[10]，这是一个判别模型（Discrimitive Model），适合于训练集十分充足的情况，因为它能给出一个更小的预测误差，以及条件随机场（CRF）[11]，该模型是一个基于无向图的判别模型，因此适用于当前状态均受前后影响的情况。

对于关系抽取问题，主要包括基于分类的方法（Classification-based）[12--15] 和基于核的方法（Kernal-based）[16, 17]。其中，分类方法在应用之前需要对文本进行特征提取（Feature Extraction），比较有用的特征包括实体、词汇、语法和背景知识等。而核方法最常见的场景是先将文本用向量空间模型（VSM），再进行下一步计算，主要分为基于序列的核方法、基于树的和方法和混合核方法，在

机器学习中，核函数（Kernal Function）是两组向量空间的内积，也可以看做观测值之间的一种相似度衡量方法，从而观测值不需要显示地映射成向量空间。

总结来说，当前命名实体检测最有效的方法都是基于概率图模型而提出的，典型代表是 MEMM 和 CRF，而关系抽取的方法则是很大程度上基于选取的特征或者定义的核函数，然后再利用分类算法进行解决。可以发现，上述这些主流方法都是有监督方法，但是，随着互联网产生数据的规模不断增大，半监督和无监督方法会更加实用。

1.3.1.2 文本摘要

文本摘要（Text Summarization）用于从纯文本中识别出重要的内容，包括重要的句子、关键词和主题等。因此根据不同的表示方式，基本方法可以分为主题表示法、标识符表示法、句子表示法。

其中，主题表示法需要首先找出一种主题描述的方式，一般是用主题词，这种词的特性是排除文档中出现频率高以及极少出现的词，然后利用对数似然比检验（LR Test）来识别那些能够很具有代表性的单词 [18]。关于单词出现频率的计算方式，最经典的就是 TF-IDF[19]，这种方法一方面计算了单词出现的频率，另一方面也衡量了单词的重要性。再进一步，文献 [20] 提出的潜语义分析（LSA）方法则用一种隐含的方式来表示文本中的语义信息，其十分适用于对新闻文本的总结，而且不需要使用第三方的词典数据就能实现。在 LSA 的基础上，更多的基于贝叶斯理论的主题模型被陆续提出，这些模型在主题表示上十分成熟，而且很多已经在工业界广泛使用 [21--24]。

对于句子粒度的摘要方法，通常是先将相似的句子进行聚类 [25, 26]，其中相似度的计算方式包括简单的余弦相似度（Cosine Similarity）等。处于同一类中的句子被认为是某种主题，而有很多句子的类则说明包含了重要的主题，然后从每个这样的类中选取一个句子作为这种主题的代表，组合起来就是对文章的摘要。同时，对于选出来的句子可以用 KL 散度（KL Divergence）来进行评分，从而来判断选出来的摘要的好坏。

1.3.1.3 文本降维

文本降维（Dimensionality Reduction）用于解决文本向量的维度过大的问题，这里主要的方法包括诸如 LSI（Latent Semantic Indexing）、PLSI（Probabilistic Latent Semantic Indexing）和 LDA（Latent Dirichlet Allocation）[27, 28] 等主题模型，这些模型共同的假设前提是把文档看做词袋模型（Bag-of-words），即不关心单词的先后顺序，从而可以更有效更快速地来处理文本。

对于主题模型来说，在整个文档集合 D 中，共有 M 篇文档，词汇表共有 W

个不同的词汇，并且假设总共有 K 个主题，其中文档和词汇都是已知的，主题为待求的问题。因为已经存在 $D*W$ 维的文档 -词汇矩阵 X，通过主题模型以后本质是将该矩阵分解成两个矩阵：$D*K$ 维的文档 -主题矩阵 A，和 $K*W$ 维的主题 -词汇矩阵 B，即 $X = A*B$。在矩阵 X 中，每一行表示一篇文章 d_m，每一列表示一个词汇 v_i，每个值 $x_{m,i}$ 表示该文档具有该词汇的比重，这个比重可以是通过正规化后的词频，也可以是 TF-IDF 的值。在矩阵 A 中，每一行表示文档 d_m，每一列表示主题 z_k，每个值 $a_{m,k}$ 表示该文档具有该主题特征的权重，所以文档可以表示为一个关于主题的 K 维向量，从而将文档看成一个服从多项分布的随机变量 $\vec{\theta_m}$。在矩阵 C 中，每一行表示主题 z_k，每一列表示词汇 v_i，每个值 $b_{k,i}$ 表示该主题中该词汇占有的比重，由此可见，这里的每个主题表示为关于词汇的一个 W 维向量，从而主题可以看做一个服从多项分布的随机变量 $\vec{\phi_k}$。关于如何求出参数 $\vec{\theta_m}$ 和 $\vec{\phi_k}$ 将在后文结合本文提出的模型具体详解。

但是诸如 LDA 这类的主题模型还有很多局限性，第一，LDA 推理的方法大致分为两种 Collapsed Gibbs sampling[28] 和 Variational Approximation[27]，但是两者在整个迭代过程中十分耗时，当数据量很大的时候，特别是词汇表的大小 W 达到百万级的时候，$\vec{\phi_k}$ 的维度十分高，导致计算很慢。因此，基于这个问题，目前已经提出了并行的亦或是分布式的 LDA 算法 [29, 30]，从而应对工业界大规模数据的问题。第二，当文档集随时间变化时，LDA 本身并不能很好适应这个动态的过程，一种方法是引入时间变量重构概率图模型。文献 [20] 通过扩展了时间维度和 LSI 的张量分析从而实现了对历史文档的更新。文献 [31, 32] 则在 PLSI 的基础上加上时间维度提出两个全新的主题模型，分别在视频中活动抽取和音乐抽取方面有所作为。David Blei 作为 LDA 的提出者，后来根据时间动态变化这个需求又提出了基于时间演化的主题模型 [33]，很好地解决了这方面的问题。第三，有些文档间会存在关联，比如学术论文间可以以作者和引用上建立连接，在诸多改进方案之中，RTM 模型（Relational Topic Model）[34] 在 LDA 的基础上联合模拟文档和链接的生成过程，从而很好地解决了文档网络的问题。

总结来说，基于概率图的主题模型是一种很好的文本降维方式，一方面它以统计学中的降维方法为基础，使得可以推理或近似得出隐藏在文档背后的主题是什么，另一方面，它也给出了对主题的一种新的定义，并且基于这种定义使得模型能够很容易地根据不同类型文档的需求来改进。但是一个很重要的问题是如何能够将这些主题模型应用到实际工业界中，因为在性能、效率方面还是有不尽如人意的地方。

1.3.2 用户兴趣挖掘相关工作

用户兴趣偏好主要体现在其对文本资源的喜好程度，换句话说，对用户兴趣挖掘的过程类似于对文本挖掘的过程，但是在技术上有所不同。对于用户兴趣的分析主要包括三个方面：用户兴趣的描述方式、用户兴趣挖掘和用户兴趣的匹配。

1.3.2.1 用户兴趣描述

传统的用户兴趣表示方法包括 VSM[35]、兴趣树等层次结构 [36] 以及基于本体构建的描述方式。其中，VSM 是最基本的描述方式，即将用户的兴趣表示成一个向量，每个值表示用户对该兴趣的喜好程度。对于层次结构的兴趣，主要考虑了兴趣之间的相关性问题，以及兴趣概念的大小范围之分。而最近本体工程的发展使得很多研究都偏向于将用户兴趣基于本体进行构建。User Profile（UP）是一种表示用户信息的概念模型，由用户的背景知识生成而得，文献 [37, 38] 中提出 UP 作为描述用户偏好和行为的信息。比如在用户读文章的时候，就可以用这个概念模型来判断该文档是否符合该用户的口味。为了模拟用户的这种概念模型, 可以使用本体（Ontology），即所谓的 Ontological User Profiles[39, 40]，或者 Personalized Ontology[41]，在文献 [42] 中提出了一种改进后的 Ontological User Profile。UP 可以分为三大类：a）interviewing 模式是最好的 UP，因为它们通过人工的方式来获得的，比如通过问卷调查来分析用户所在的分组，一个典型的例子是 TREC Filtering Track 训练数据集，它就是纯手工采集的数据：受访者在读完某篇调查材料后会给出一个支持或反对的表态，由于只有用户最知道自己的兴趣所在，所以这些答案能十分精确地反映用户背景知识。b）Semi-interviewing 则使用半自动技术来获得用户的信息，该技术往往是提供用户一系列分类列表，并要求用户选择自己喜欢的分类。c）Noninterviewing 则完全不用让用户参与进来，而是观察用户的活动、行为以及背景知识。一个典型的模型是 OBIWAN，其通过用户浏览的历史记录来获得用户偏好。如图**??**所示, 这是一种基于若干种上述方法采集数据后建立的 UP 模型 [43]。

1.3.2.2 用户兴趣挖掘

用户兴趣挖掘技术与文本挖掘类似，但是目的是不一样的。在情感分析（Sentimental Analysis）方面，可以使用常见的有监督分类算法对文本进行分类，比如朴素贝叶斯分类器（Naive Bayesian）和支持向量机（SVM），对用户正负情感进行分类 [44]，当用户对某个文本资源产生负情感时，很有可能说明他对主题的资源没有兴趣，反之亦然。但是，在分类之前的特征选取有所不同，因为文档

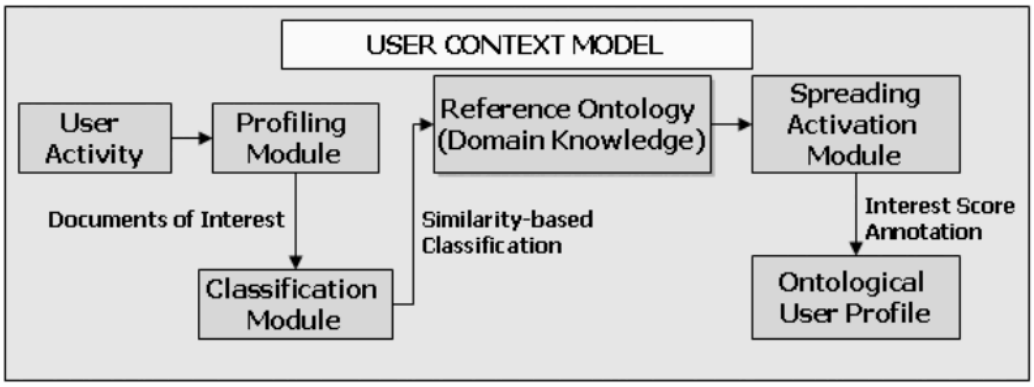


图 1.3: 一种基于 Ontological User Profile 的环境模型

中表示情感的词的权重和否定词必须更大，而为了找出那些是具有主观色彩的词汇，又需要借助已经准备好的情感关键词字典等数据集。同理，对于无监督聚类算法也是适用的，并且反而更加接近真实状态 [45, 46]。如果将分析的粒度从文档缩小到句子，也就是主观性分类问题（Subjectivity Classification），一些已有的工作 [47--50] 在这方面已经表现得十分出色，但是这些工作有一个假设前提，即假每个用户只会对一个句子产生一种情感。另外，单纯地从比较两个情感词不一定能够找出他们之间的关系，这时候依然需要借助于情感字典 [51]，即事先定义好近义词、反义词、上义词和下义词等等，这样就可以匹配两个含义类似但拼写或者写法完全不同的两个词。

1.3.2.3 用户兴趣匹配

用户兴趣的匹配主要体现在为兴趣类似的用户推荐资源，在各大电子商务平台中都存在着这样的推荐系统，其背后的原理就是基本的协同过滤算法（Collaborative Filtering）[52, 53]。CF 是在信息过滤和信息系统中正迅速成为一项很受欢迎的技术。与传统的基于内容过滤直接分析内容进行推荐不同，协同过滤分析用户兴趣，在用户群中找到指定用户的相似（兴趣）用户，综合这些相似用户对某一信息的评价，形成系统对该指定用户对此信息的喜好程度预测。典型的协同过滤推荐算法是基于用户的协同过滤推荐算法，其基本原理是利用历史评分数据形成用户邻居，根据评分相似的最近邻居的评分数据向目标用户产生推荐。基于用户的协同过滤推荐是基于这样一个假设：如果用户对一些项目的评分比较相似，则他们对其他项目的评分也比较相似。推荐过程可分为三步：a）首先获得用户对项目的评分，建立用户一项目评分矩阵；b）计算目标用户与其他各个用户的相似度，得到目标用户的最近邻居集合；c）根据其最近邻居对项目的评分信息预测目标用户对未评分项目的评分，取出预测评分最高的前 n 项

CF categories	Representative techniques	Main advantages	Main shortcomings
Memory-based CF	*Neighbor-based CF (item-based/user-based CF algorithms with Pearson/vector cosine correlation) *Item-based/user-based top-*N* recommendations	*easy implementation *new data can be added easily and incrementally *need not consider the content of the items being recommended *scale well with co-rated items	*are dependent on human ratings *performance decrease when data are sparse *cannot recommend for new users and items *have limited scalability for large datasets
Model-based CF	*Bayesian belief nets CF *clustering CF *MDP-based CF *latent semantic CF *sparse factor analysis *CF using dimensionality reduction techniques, for example, *SVD*, *PCA*	*better address the sparsity, scalability and other problems *improve prediction performance *give an intuitive rationale for recommendations	*expensive model-building *have trade-off between prediction performance and scalability *lose useful information for dimensionality reduction techniques
Hybrid recommenders	*content-based CF recommender, for example, *Fab* *content-boosted CF *hybrid CF combining memory-based and model-based *CF* algorithms, for example, Personality Diagnosis	*overcome limitations of CF and content-based or other recommenders *improve prediction performance *overcome CF problems such as sparsity and gray sheep	*have increased complexity and expense for implementation *need external information that usually not available

图 1.4: 协同过滤推荐算法的综述

推荐给用户，完成推荐。如图 1.4所示，CF 主要分为三大类，即基于记忆的 CF、基于模型的 CF 和混合 CF。其中，基于记忆的 CF 通常使用用户对资源的评价信息来计算用户之间的兴趣相似度，再根据相似度来给出推荐和预测。该方法的不足之处是由于用户评价过的资源数量相对于资源总数是一个很小的值，导致了数据十分稀疏，从而推荐不准确甚至无法推荐。基于模型的 CF 则直接使用评价的数据来训练模型，并用过模型来做预测，这些模型包括上面提到的主题模型、聚类算法等，但是这些模型本身的缺陷会在 CF 预测的过程中带来潜在影响。混合 Cf 则结合上述两种方法，在一定程度上可以弥补上述方法各自的缺陷。

1.3.3 P2P 网络路由搜索相关工作

P2P 网络主要以其非中心化特点为主，因此主要研究问题集中在网络中的搜索技术，但是搜索的方法在很大程度上基于整个网络的拓扑结构，因此下面从拓扑结构和搜索技术两部分来阐述现有工作。

1.3.3.1 P2P 网络的拓扑结构

拓扑结构表示 P2P 网络中各个节点之间在物理或者逻辑上相互连接的关系，主要有四种类型：中心化拓扑、全分布式非结构化拓扑、全分布式结构化拓扑、半分布式拓扑。这四类拓扑结构的优缺点比较如图 1.1所示。

其中，中心化拓扑结构最为简单，典型的例子是 Napster[3]，该结构与传统的中心化服务器架构类似，普通的节点需要从中央索引服务器中获取其他节点的

[3]http://www.napster.com/

表 1.1: P2P 网络四种拓扑结构的优缺点对比（A 最好，D 最差）

拓扑	中心化	全分布式非结构化	全分布式结构化	半分布式
可扩展性	D	D	B	C
可靠性	D	B	B	C
可维护性	A	A	B	C
搜索性能	A	C	B	B
复杂查询	支持	支持	不支持	支持

信息，从而才能进行路由，该结构的好处是十分便于节点搜索复杂的内容，并且速度十分快。但是，该结构的缺陷也十分明显，一旦中央索引服务器宕机时，整个网络就会处于瘫痪状态，因此该拓扑不适用于大型的网络系统。全分布式非结构化拓扑适用于覆盖网络（Overlay Network），该结构一般基于随机图来构建，节点的度数服从幂定律（Power-Law）[54]，因为节点之间没有严格定义的关系，所以该拓扑具有很好的灵活性，动态变化的结构在容错计算上也有很好表现，经典的案例是 Gnutella[55]。但是，由于该拓扑结构是完全随机的，因此具有一定的不可预见性，在查找和搜索方面也相对复杂很多。全分布式结构化拓扑则是一种综合了上述两种拓扑的特性，使用了分布式散列表（DHT）来组织网络中的节点关系，典型的例子包括 Tapestry[56]、Pastry[57] 和 Chord[58] 等等。DHT 的作用是将网络中节点的名字映射到散列表中，这样可以更加快速地进行查找，而且同时也支持动态添加和删除节点。半分布式拓扑（Hybrid Structure）则是一种介于中心化结构和完全分布式结构之间的一种拓扑，在网络中存在一些超级节点（Super Node/Hub），这些节点与中心化拓扑中的中央索引服务器作用类似，提供周围节点的信息。当有源节点提出请求时，这些超级节点会互相通信，找出目标节点的位置等相关信息，并返回给源节点以供其进行连接，一个著名的例子就是 KaZaa[59]。

1.3.3.2 P2P 网络的搜索技术

P2P 网络中由于每个节点仅仅存储有限的信息，包括节点本身的网络配置、索引服务器的信息以及周围小部分节点的信息等，因此需要通过搜索算法来找到要与之建立连接的目标节点。在结构化拓扑的 P2P 网络中，搜索算法常常是基于 DHT 查找就能完成。在非结构化拓扑的 P2P 网络中，搜索算法都是基于小世界理论 [60]，常见的搜索算法有以下几个 [61]：

- Flood Fill：这个算法最早用在 Gnutella 协议中，与传统的图算法 BFS 类似，即先搜索当前节点的所有邻居节点，然后对每个邻居节点执行相同的操作。

该方法实现起来十分简单，但是会产生大量的网络流量并造成网络拥堵。

- Modified-BFS：该方法在 Flood Fill 的基础上加上一个概率条件，即以一定的概率选取部分邻居节点，其他不变。
- Iterative Deepening：该方法是在 Flood Fill 的基础上增加了一个阈值 TTL（Time to Live），从而控制了搜索半径。
- Random Walk：这个算法基于图的 DFS 搜索，首先随机选取源节点的一个邻居节点，并对该邻居节点进行同样操作，直至完成一条完整路径的索索，然后从源节点开始重新执行这些操作 [62]。
- Agent-based：移动 Agent 是一个在网络中交换和传递资源的程序，也就是所有的搜索的任务都有网络中的 agent 来完成，agent 与 agent 之间也会交换信息，从而实现搜索 [63]。
- Query Routing：这个方法是直接对每个节点的资源建立索引，同时也会记录周围节点的索引信息，所以当查询请求来到时，不用再次转发给其他节点进行搜索，而是直接从索引表中定位到目标资源的位置信息，并直接与目标节点建立连接。

总结来说，这些方法在一定程度上已经十分适应很多分布式的场景，但是随着移动设备的普及，每个移动设备可能也是一个潜在的节点，但是这些设备的加入或退出更大地加剧了网络波动（Churn），从而导致整个网络拓扑结构的不稳定，并且如何保证在这种大波动的情况下进行精确的查询和搜索是未来 P2P 网络中搜索技术的一大挑战。

1.4 本文内容安排

本文的主要内容安排如下。首先本章对该研究的背景，主要研究内容和研究意义与研究现状进行了一个总体的介绍。第二章将对基于 P2P 网络的信息自主流动机制提出一个完整的技术方案，并且结合研究现状和存在的问题对设计需求和挑战进行详细描述，从而给出关键的技术点。第三章将对解决方案中的第一部分即文本资源的描述与匹配问题进行深入分析，通过结合现有成熟的模型，提出一整套全面的理论模型。第四章将结合第三章的研究基础对用户兴趣进行建模，结合现实中兴趣的动态变化特征给出相应的解决方案。第五章将基于第四章的研究基础提出一种基于 P2P 网络的路由与搜索算法，从而实现动态构建兴趣覆盖网络。第六章将根据第三、四、五章的理论进行模拟实验和真实实验，验证上面提出的新的方法的有效性。第七章则基于前文的研究成果实现了一套简

单的智能系统原型，包括功能设计和系统架构等方面。最后第八章对本研究做了一个总结，并对研究进一步的工作方向进行了讨论。

第 2 章 基于 P2P 网络的信息自主流动机制的方案概述

本章将阐述基于 P2P 网络的信息自主流动机制的基本方案。在整套机制中，需要涉及三大研究模块：在文本信息分析与建模的研究中，主要针对互联网中纯文本的信息的非结构化问题，通过数据挖掘和机器学习等方法从中抽取出一部分能够充分表示原文本的特征，并且提出一套适用于各种类型文本的描述方式；在用户兴趣表示与匹配的研究中，主要结合文本信息与用户反馈提出一种具有代表性的兴趣构建方法，从而可以进行用户与用户、用户与资源之间的兴趣相似度计算。在 P2P 网络路由与发现的研究中，主要利用上述提出的针对资源和用户兴趣的模型，以现在网络为底层架构，构建一层用户兴趣覆盖网络，并提出了网络构建、动态更新与搜索发现等基本方法。最后，基于上述三个模块，本文还设计并实现了一个简易实现的原型系统，主要包括架构设计和功能设计，以此来说明信息自主流动机制的可行性和实际应用价值。

2.1 文本信息分析与建模

按照互联网资源的类型可以分为：文本、图片、音乐、视频等。虽然如今涌现了越来越多的诸如音频和视频的流媒体资源，但是互联网上的资源大多任然以文本形式存在，或者可以用文本来代替原有的非文本资源。不仅限于传统的新闻、博客是文本信息，连图片也可以用很成熟的技术来转换成文本，比如百度识图[4]就可以自动识别出图片中的物体，即使是视频资源也可以用元信息、标题、评论和弹幕等文本信息进行概括。因此下文主要针对互联网文本信息进行深入研究，并假设其他类型的资源均可以通过已有成熟的方法转换成文本信息。这一小节将根据互联网上四种不同类型的文本，分别提出文本挖掘与建模的方法。

普通的新闻、博客等文章都归类为长文本信息，这类资源一般都是纯文本数据，因此具有两个明显的特征：稀疏性和高维度。举个例子来说：假设给定一由 $|\mathcal{V}|$ 个不同词汇组成的字典 $\mathcal{V}$，在由 M 篇文章组成的语料库中，每篇文章 d 的用词都属于词典 $\mathcal{V}$，且每篇文章的单词数量不少于 $n(0 < n \ll W)$。如果直接简单地将文档表示成关于词汇的向量，向量中的每个值表示该词汇在文中的词频，如果该词汇没有在文中出现，则向量中对应的值为 0。那么有两点是显而易见的：a）文档向量的维度为 $|\mathcal{V}|$，在正常情况下，$|\mathcal{V}|$ 可以达到百万数量级（10^6）；b）文档向量中最多只有 n 个值大于 0，一般情况下，n 只有几百数量级（10^2）。这就是纯文本的高维度和稀疏性带来的问题。为了解决这个问题，本文采用基于概率图的主题模型来进行分析，这里先形式化地定义几个重要的基本概念。

对于包含 M 篇文档 d 的语料库 $\mathcal{D}$，有 $\mathcal{D} = \{d_1, d_2, ..., d_M\}$。词汇表 $\mathcal{V}$ 中包含

[4]http://stu.baidu.com/

了 W 个不同的词汇 v。对于文档 d_i，其中每个单词 w 都取自于 $\mathcal{V}$，并且单词可以重复，即 $w_i = w_j = v\ (i \neq j, v \in \mathcal{V})$。那么，对于长度为 N 的文档 d 表示为关于单词的向量，$d = \vec{w} = (w_1, w_2, ..., w_N)$。

为了简化模型，一般认为文档单词之间是没有先后关系的，换句话说，每个单词相互独立，一个单词出现的概率不会收到另一个单词的影响。因此可以计算文档 d 的生成概率 $p(d)$：

$$p(d) = p(\vec{w}) = p(w_1, w_2, ..., w_N) = p(w_1)p(w_2)...p(w_N) \tag{2.1}$$

而对于语料库 $\mathcal{D}$ 的生成概率 $p(\mathcal{D})$ 为：

$$p(\mathcal{D}) = p(d_1)p(d_2)...p(d_M) = p(\vec{w_1}, \vec{w_2}, ..., \vec{w_M}) \tag{2.2}$$

如果用词频来描述文档的话，$d_m = \vec{n_m} = (n_{m,1}, n_{m,2}, ..., n_{m,W})$，其中 $n_{m,i}\ (1 \leq i \leq W)$ 表示文档 d_m 中出现词汇 v_i 的频数。于是，整个语料库 $\mathcal{D}$ 可以表示为一个 $M * W$ 维的矩阵 X，即：

$$X = \begin{pmatrix} n_{1,1} & n_{1,2} & \cdots & n_{1,W} \\ n_{2,1} & n_{2,2} & \cdots & n_{2,W} \\ \vdots & \vdots & \ddots & \vdots \\ n_{M,1} & n_{M,2} & \cdots & n_{M,W} \end{pmatrix}$$

其中，$n_{m,v}\ (1 \leq m \leq M, 1 \leq v \leq W)$ 表示文档 d 中拥有词汇 v 的数量。

假设词汇 v_i 出现的先验概率为 p_i，词汇表 $\mathcal{V}$ 中所有词汇组成的先验概率为 $\vec{p} = (p_1, p_2, ..., p_W)$。又文档 $d = \vec{n} = (n_1, n_2, ..., n_W)$，那么 d 生成的概率为：

$$p(d) = p(v_1)^{n_1} p(v_2)^{n_2} ... p(v_W)^{n_W} = \prod_{i=1}^{W} p_i^{n_i} \tag{2.3}$$

从整个语料库来看，假设每个词汇 v_i 出现的次数为 n_i，那么语料库 $\mathcal{D}$ 又可以表示为 $\vec{n} = (n_1, n_2, ..., n_W)$，在语料库生成过程中，可以把 $\vec{n}$ 看做一个是服从多项分布的随机变量，而 $(n_1, n_2, ..., n_W)$ 是一组具体的观测值，得到如下：

$$p(\vec{n}) = mult(\vec{n}|\vec{p}, N) = \binom{N}{\vec{n}} \prod_{i=1}^{W} p_i^{n_i} \tag{2.4}$$

现在的问题是如何利用语料库 $\mathcal{D}$ 已知的数据来估计未知的参数 $\vec{p}$。$\vec{p}$ 的实际意义是词汇表 $\mathcal{V}$ 中每个词汇各自出现的频率。

一种简单的方法是利用最大似然估计（MLE）来估计参数 p_i 的值得到：

$$\widehat{p_i} = \frac{n_i}{N} \tag{2.5}$$

其中 N 为语料库的单词总数。这个方法的假设前提是 $\vec{p}$ 的值本身是一个实数常量，换句话说，任何其他语料库 $\mathcal{D}'$ 的生成过程也是依赖于同一个词汇表来生成的。

但是，这个假设对于互联网上的文本是不正确的，有以下三点主要原因：

- 多类型文档：互联网上的文档类型包括了新闻、技术博客、心情随笔等等，这些文档的用词肯定是很不一样的。比如，对于新闻类型的文档，很少能使用主观色彩的词汇，因此如果将所有新闻文档组成一个词汇表的话，主观性形容词的先验概率是十分低的。但是，这与心情随笔类型的文档恰恰相反，因为此类文档大多是抒发作者内心的心情的，如果将此类型文档组成一个词汇表的话，主观性形容词的先验概率十分高。
- 多元化主题：即使是同一类型的文档也有各种不同主题，比如新闻中就包含体育、娱乐、经济、社会、科技等各种主题。对于不同的主题，生成文档的词汇表应该也是不同的。如果将上述 $\vec{p}$ 中的值认为是常数的话，相当于所有文档都是从同一个主题中生成的，这显然违背互联网资源的特征。
- 个性化写作：即使上述两点都保持一致，每个作者写作风格和兴趣爱好同样会影响文档生成的过程。比如说，将每个作者写的所有技术博客各自组成一个词汇表，由于用词习惯等因素，那么这些词汇表中的词汇出现概率肯定也千差万别。

总而言之，一篇文档的生成过程受到包括文档类型的制约、主题风格的用词和用户个性化的定制等几方面的影响。因此，基于这些原因，对互联网文本的表示不能简单地用一个词频向量 $d = \vec{n} = (n_1, n_2, ..., n_W)$ 来表示。

自然而然就考虑到了两个重要的因素。第一，存在不止一个的词汇表，那么 $\vec{p}$ 就应该被认为是一个随机变量，所以语料库生成的概率应该是受词汇表影响的一个条件概率 $p(\mathcal{D}|\vec{p})$。为了得到随机变量 $\vec{p}$ 的分布和参数，可以根据贝叶斯定理得到：

$$p(\vec{p}|\mathcal{D}) = \frac{p(\vec{p})p(\mathcal{D}|\vec{p})}{p(\mathcal{D})} \tag{2.6}$$

其中，$p(\mathcal{D}|\vec{p})$ 就是之前的 $\vec{n} = (n_1, n_2, ..., n_W)$，即语料库中每个词汇出现的频数，且和公式 2.4一样，是一个服从多项分布的随机变量。为了计算方便，这里使得 $p(\vec{p}|\vec{\alpha})$ 成为多项分布的共轭先验，也就是 Dirichlet 分布，得到：

$$p(\vec{p}|\vec{\alpha}) = Dir(\vec{p}|\vec{a}) = \frac{1}{\Delta(\vec{\alpha})}\prod_{i=1}^{W} p_i^{\alpha_i - 1}, \ \ \vec{\alpha} = (\alpha_1, ..., \alpha_W) \tag{2.7}$$

式中的 $\Delta(\vec{\alpha})$ 是事先给定的归一化因子：

$$\Delta(\vec{\alpha}) = \int \prod_{i=1}^{W} p_i^{\alpha_i - 1} d\vec{p}$$

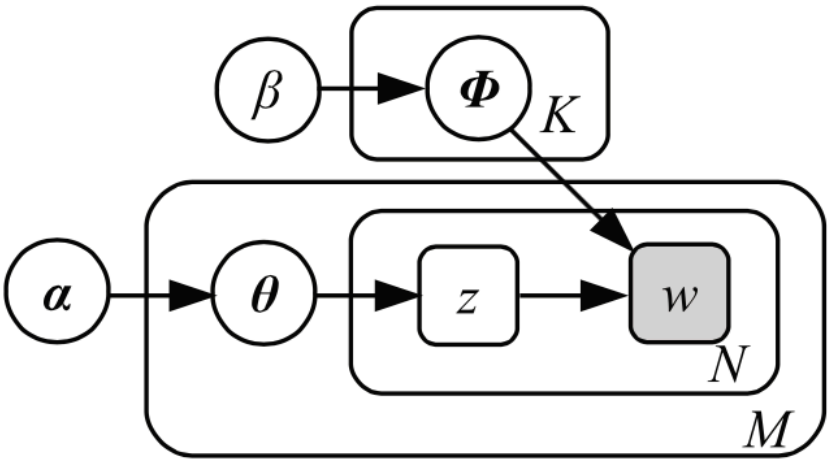


图 2.1: LDA 主题模型的概率图表示方式

至此，对于给定 $\vec{\alpha}$，语料库 D 的生成概率为：

$$p(\mathcal{D}|\vec{\alpha}) = \int p(\mathcal{D}|\vec{p})p(\vec{p}|\vec{\alpha})d\vec{p} \tag{2.8}$$

$$= \int (\prod_{i=1}^{W} p_i^{n_i} \frac{1}{\Delta(\vec{\alpha})})(\prod_{i=1}^{W} p_i^{\alpha_i - 1})d\vec{p} \tag{2.9}$$

$$= \frac{\Delta(\vec{n}+\vec{\alpha})}{\Delta(\vec{\alpha})} \tag{2.10}$$

第二，除了词汇表是随机取一个的之外，还可以为每个文档加上主题，这里的主题可以近似看做不同的词汇表，或者更加确切应该是词汇表中词汇的分布。对于互联网文本来说，每篇文档的主题应该不只一个，而且多个主题之间的比重也是不一样的。因此，文档的生成过程应该是先给定一个主题的分布，表示每个主题在文档中占有的比重，然后再根据这个主题的分布随机选取一个主题（也就是词汇表），最后根据这个词汇表生成一个单词。在生成第二个单词的时候，再根据主题分布选择一个主题，并用相应的词汇表生成一个单词。以此类推直至完成一篇文章的生成过程。这个生成过程与 LDA 背后的思想类似，因此就直接引用 LDA 模型的算法进行求解。

如图 2.1所示，对于 M 篇文档、K 个主题的语料库，文档 -主题的分布为 $\vec{\theta_m}$ $(1 \leq m \leq M)$，主题 -词汇的分布为 $\vec{\phi_k}$ $(1 \leq k \leq K)$。这两个随机变量与上文的词汇表类似，且都服从 Dirichlet 分布，即：

$$p(\vec{\theta}|\vec{\alpha}) = Dir(\vec{\theta}|\vec{\alpha}) = \frac{1}{\Delta(\vec{\alpha})} \prod_{i=1}^{K} \theta_i^{\alpha_i - 1} \tag{2.11}$$

$$p(\vec{\phi}|\vec{\beta}) = Dir(\vec{\phi}|\vec{\beta}) = \frac{1}{\Delta(\vec{\beta})} \prod_{i=1}^{W} \theta_i^{\beta_i - 1} \tag{2.12}$$

在文档 d 中生成第 n 个单词的时候，首先要从文档 -主题分布 $\vec{\theta}$ 中抽样得到

的主题编号为 $z_{d,n}$，然后选取编号为 $z_{d,n}$ 的主题 -词汇分布 $\vec{\phi}$，并抽样得到的一个单词 $w_{d,n}$。由上面的理论可知，从 Dirichlet 分布抽样可以得到的观测值服从多项分布，记 $\vec{x_d} = (x_{d,1}, ..., x_{d,K})$，$x_{d,k}$ $(1 \leq k \leq K)$ 表示在文档 d 中，抽样得到的编号为 k 的主题数量；同理，$\vec{y_d} = (y_{d,1}, ..., y_{d,W})$，$y_{d,v}$ $(1 \leq v \leq W)$ 表示在文档 d 中，抽样得到的词汇 v 的数量。$\vec{y_d}$ 和 $\vec{y_d}$ 均为多项分布，即：

$$p(\vec{x_d}|\vec{\theta_d}) = mult(\vec{x_d}|\vec{\theta_d}) = \binom{N}{\vec{x_d}} \prod_{i=1}^{K} \theta_i^{x_{d,i}} \tag{2.13}$$

$$p(\vec{y_d}|\vec{x_d}, \vec{\phi_d}) = mult(\vec{y_d}|\vec{x_d}, \vec{\phi_d}) = \binom{N}{\vec{y_d}} \prod_{i=1}^{W} \phi_i^{y_{d,i}} \tag{2.14}$$

有了以上基础之后，最终可以得到文档 $d_m = \vec{w_m}$ 的生成概率为 [28]：

$$p(d_m|\vec{\alpha}, \vec{\beta}) = \iint p(\vec{\theta_m}|\vec{\alpha}) p(\Phi|\vec{\beta}) \cdot \prod_{n=1}^{N_m} p(w_{m,n}|\vec{\theta_m}, \Phi) d\Phi d\vec{\theta_m} \tag{2.15}$$

所以整个语料库生成的概率为：

$$p(\mathcal{D}|\vec{\alpha}, \vec{\beta}) = \prod_{m=1}^{M} p(d_m|\vec{\alpha}, \vec{\beta}) \tag{2.16}$$

现在的问题转换成估计参数 $\Theta = \{\vec{\theta_1}, ..., \vec{\theta_M}\}$ 和 $\Phi = \{\vec{\phi_1}, ..., \vec{\phi_K}\}$。考虑到该问题涉及过多纯统计学知识，下文将直接使用成熟的 collapsed Gibbs Sampling 算法 [64] 进行求解，具体公式将不再列出。

2.2 用户兴趣表示与匹配

随着互联网上新型的网站越来越多，用户也会被各种各样的信息资源所吸引，而这些资源的主题可以在一定程度上反映出用户的兴趣所在。因此，将利用之前提出的文本信息分析与匹配方法先挖掘出隐藏在文本背后的主题，然后，通过发现并结合互联网用户兴趣的特点，提出一种可以全面且有效的表示方法和其对应的匹配机制。

首先，目前最常用的表示用户兴趣的方式是 VSM，假设一共存在 K 种兴趣，并且兴趣之间没有关联性，而用户的兴趣则表示为一个 K 维向量 $\vec{in} = (w_1, w_2, ..., w_K)$，$w_i \in [0, 1](1 \leq i \leq K)$，$w_i$ 表示用户对第 i 个兴趣的喜好程度。用 VSM 来表示用户兴趣的好处是在计算兴趣相似度的时候速度十分快，一般使用余弦相似度就能计算得出两个向量之间的距离。当余弦值越小就表示用户之间拥有更为相似的兴趣爱好。虽然这个方法在大部分情况下都十分快捷有效，但是，考虑到互联网用户的特点，上述方法存在下面两个问题，因此不能用于完全表示用户兴趣。

- 兴趣关联性：由于用户的兴趣对应于文本资源中的主题，而主题之间是由相关性的，这种相关性体现在两个方面。一是主题之间的同义或反义，比如，CBA 和 NBA 是两个主题，但是它们之间的共同点是都是篮球比赛，区别在于举办地点不同、规则不同等等。二是主题之间有层级关系，比如篮球和 NBA 是两个主题，篮球是一个更加广阔的概念，包括了篮球比赛、篮球运动员、实体篮球等，而 NBA 则是一个相对狭窄的概念，所以 NBA 这个主题是属于篮球的。

- 兴趣动态性：用户的兴趣一般可以分为长期兴趣和短期兴趣。长期兴趣相对稳定，变化较少，构建模型时比较容易。但是短期兴趣是一个难点，因为它会随着时间的变化而变化，而且每次持续的时间也都未知，可以近似把兴趣与时间的关系看成一个随机过程。

为了解决上述两个问题，第三章提出了一套完整的用户兴趣构建方式，包括一种叫做动态兴趣树的兴趣表示方式和一种层级结构的兴趣相似度衡量算法。动态兴趣树的表示方式解决了兴趣随着时间变化的问题，树中的每个节点表示一种兴趣，当用户兴趣发生变化时，这些节点存储的兴趣会通过旋转方式分辨出长期兴趣和短期兴趣。而层级结构的兴趣相似度算法是在主题模型的基础上，增加了层次结构，使得主题之间具有相关性。为了在第四章中更加简单和清晰地阐述这两种方法的构建和维护，这里先对用户兴趣模型的基本概念进行说明。

假设总共有 K 种兴趣点构成了兴趣全集 $\mathcal{I}$，即 $\mathcal{I} = \{in_1, in_2, ..., in_K\}$，并且事先定义大小为 $|V|$ 的词汇表 V。对于兴趣点 in_k $(1 \leq k \leq K)$，是一个关于词汇的向量 $in_k = (w_1, w_2, ..., w_V)$，其中，$w_i$ $(1 \leq i \leq |V|)$ 表示词汇 v_i 在当前兴趣点 in_k 的重要程度。那么，兴趣点之间的相似度计算可以利用余弦相似度来计算两个向量的夹角。

普通的静态兴趣树是一种将兴趣点进行层次分类的表示方式，这种数据结构能够很好地表示出兴趣点之间的关联性。如图 2.2所示，这是一棵四层结构的兴趣树，树上的每个节点表示一种兴趣点，距离根节点越近的节点表示范围越广泛。换句话说，在兴趣树上，父节点表达的兴趣点一般包含了其孩子节点所表达的内容，同时，处于同一层且拥有相同父节点的孩子节点一般具有部分交集的内容。于是，可以对兴趣全集 $\mathcal{I}$ 中所有的兴趣点来构建一个完整的兴趣树。具体的构建方法可以分为人工构建法、半自动构件法和自动构建法。人工构建法可以借助词典来进行，因为在词典中定义了部分词汇的上义词、下义词、同义词，如果兴趣点 in 的名称正好在这部分词汇内，那么可以直接建立关系，然后再将剩余节点根据人为的经验来判断具体的位置。可以看到这种方法的精确性应该是最高的，因为它十分符合人正常的思维习惯，但是问题也十分明显，即当兴趣全集的数量十分巨大的时候，人工就十分费时费力了。因此，与人工构建法相对的

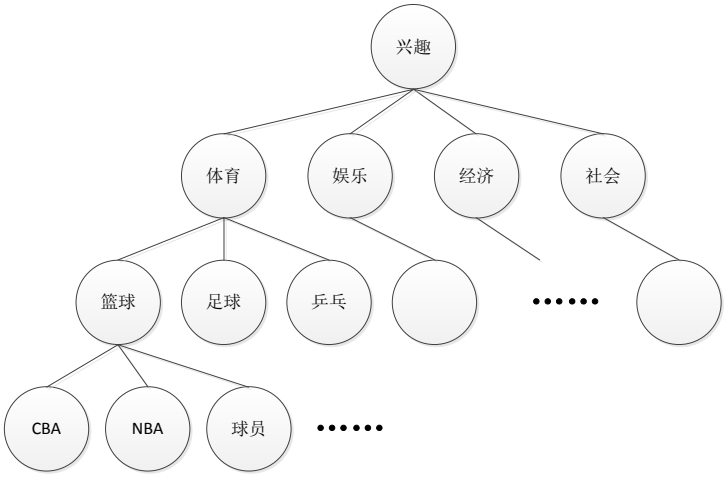


图 2.2: 静态兴趣分类树实例

是自动构建法，这种方法基于兴趣点相似度的计算方式，最简单的方法是计算余弦相似度。给定两个兴趣点 in_i 和 in_j，余弦相似度 $sim(in_i, in_j)$ 为：

$$sim(in_i, in_j) = cos(in_i, in_j) = \frac{\sum_{k=1}^{|V|} w_{i,k} w_{j,k}}{\sqrt{\sum_{k=1}^{|V|} w_{i,k}^2 \cdot \sum_{k=1}^{|V|} w_{j,k}^2}} \tag{2.17}$$

然后，当构建第二层时（因为第一层的根节点不是兴趣点），对于给定的分类数量 N，对所有兴趣点进行聚类，可以用常见的 K-means 算法进行计算。对于每一个类中，对每个兴趣点进行两两的相似度计算，从而得到一个兴趣点之间的相关性矩阵 A：

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,j} & \cdots \\ \vdots & \ddots & \vdots & \\ a_{i,1} & \cdots & a_{i,j} & \cdots \\ \vdots & & \vdots & \ddots \end{pmatrix}$$

其中，$a_{i,j} = sim(in_i, in_j)$。接着，对矩阵 A 中的每个行向量计算均值 $mean(\vec{a_i})$ 和方差 $var(\vec{a_i})$，选择均值较大方差较小的行向量，其对应的兴趣点作为第二层的一个节点。依次类推完成其他节点的构建。可以发现，自动构建法适用于大规模兴趣点的场景，但是精确度比人工构建法会差一点，因此，一般使用半自动构建法，即结合人工反馈的方式和自动分类的方式进行构建，具体方法将在第四章中详细说明。同时，对于每个用户 u 来说，其各自还需要维护一棵兴趣子树，从而可以进行用户之间两两的兴趣匹配。

此外，这种静态的分类兴趣树仅仅描述了兴趣点之间的关联性，但是用户的兴趣还分为长期兴趣和短期兴趣，对于动态变化的短期兴趣来说，这种静态的结构不太适合，因此提出了一种基于二叉伸展树结构的动态兴趣树，r 其基本原理是每当用户有新的兴趣加入时，将交换已有树中的节点位置，离根节点越近

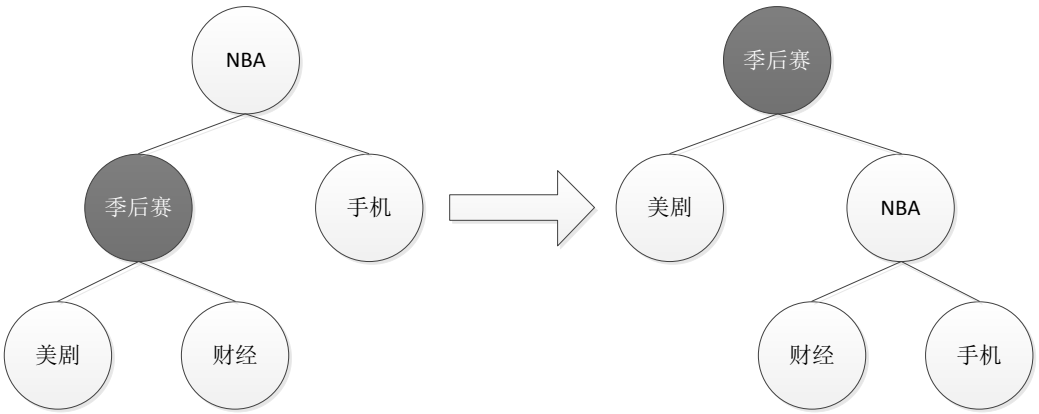


图 2.3: 动态兴趣伸展树实例

的节点说明是用户最新的兴趣，相反离得越远的节点说明用户只是在很久以前曾经有过的兴趣。如图 2.3所示，当系统发现最近用户对季后赛有了浓厚的兴趣时，伸展树的结构就会发生变化，季后赛节点从左边旋转到根节点，表示用户当前最感兴趣的一项。

二叉伸展树是一种自适应的二叉搜索树，其遵循的特点是近期访问过的节点将会在不久之后又被访问到。对于诸如插入节点、查找节点、和删除节点等基本操作都在 $O(log\ n)$ 的时间内。对于一系列非随机的操作，伸展树的性能比其他搜索树表现得更好。这一点尤为重要，因为在互联网中，用户的兴趣变更十分频繁，这些变更的操作包括了加入新的兴趣、重新对已有兴趣有了热情等等，之所以选择二叉伸展树就是因为其可以表现出这个特征。另外，伸展树的性能在一定程度上取决于它的自优化过程，频繁操作的节点比不操作的节点更容易接近根节点。伸展树的高度在最坏情况达到了 $O(n)$，但平均情况稳定在 $O(log\ n)$，最坏情况下伸展树的高度成为了最主要的弊端。

但是，在兴趣伸展树中不能体现出兴趣点之间的联系，因此需要将静态和动态的兴趣树结合起来判断用户的完整兴趣，具体方法将在第四章中详细说明。

2.3 P2P 网络构建与发现

基于上述两部分的工作，接下来要在已有的网络架构上建立一层 P2P 网络来实现信息自主流动的机制。在构建网络之前，必须确保理解以下几个关于信息流动的问题。

- 信息流动的目的：互联网上的信息内容众多，但是用户真正需要的信息不多，因此信息流动的本质是为了让用户获得更多有用的信息，过滤掉相对没用的信息。这个过程与信息推荐有所类似，但是也有所不同，相同点是用

户都是被动地接受处理过后的信息，但不同点是，现在推荐系统都是基于集中式的架构，而此处信息流动是一个更加广域的概念，其基础是建立在 P2P 的网络中。

- 信息流动的依据：在文本中，用户有用的信息是指用户真正感兴趣的内容，用户兴趣模型的作用就是先挖掘出用户的潜在兴趣，然后匹配信息的主题与用户兴趣之间的关联度，并进行推荐。换句话说，信息流动是由用户兴趣所驱动的，因此只要衡量目标用户的兴趣与信息发布者的资源是否相似即可。

因此，要建立的 P2P 网络是建立在用户兴趣的基础之上的，或者更确切的说法是，将在已有的网络架构上添加一层基于 P2P 架构的用户兴趣覆盖网络，从而使得信息能够在这一层网络中进行自动传播。为了说明兴趣覆盖网络的基本概念，本节用暂时使用 VSM 模型来表示用户兴趣。

首先，在基于 P2P 架构的兴趣覆盖网络上，先有一层用户好友关系层。其中，每个用户表示一个节点，用户集合 U 可以表示为：

$$U = \{u_1, u_2, ..., u_{|U|}\}$$

$|U|$ 表示用户集合的大小。

在目前传统的网络中，用户之间可以相互加为好友，把这种好友的关系定义为 follower 和 followee 的单向关注的关系：

$$\forall u_i, u_j \in U, \exists e_{i,j} : \ u_i \rightarrow u_j$$

如果用户 u_i 关注了用户 u_j，u_i 和 u_j 之间则存在一条边 $e_{i,j}$。那么所有的关注关系集合 E_f 可以表示为：

$$E_f = \{e_{i,j} | u_i \rightarrow u_j, 1 \leq i, j \leq n\}$$

于是用户集合和关系集合构成一张有向图 $G_f = \{U, E_f\}$。

接下来，要在用户好友关系层上加上兴趣覆盖网络。假设兴趣全集定义与上面相同，即表示为：

$$\mathcal{I} = \{in_1, in_2, ..., in_{|\mathcal{I}|}\}$$

用户 u 的兴趣可以用向量简单地表示为：

$$d_u = (w_{u,1}, w_{u,2}, ..., w_{u,|\mathcal{I}|})$$

其中 $w_{u,i} \in [0, 1] \ (1 \leq i \leq |\mathcal{I}|)$ 表示用户 u 对兴趣点 in_i 实际感兴趣的程度。对于给定的阈值 ϵ，当且仅当 $w_i > \epsilon$ 时，称用户 u 拥有兴趣点 in_i，否则用户 u 不拥有兴趣点 in_i。

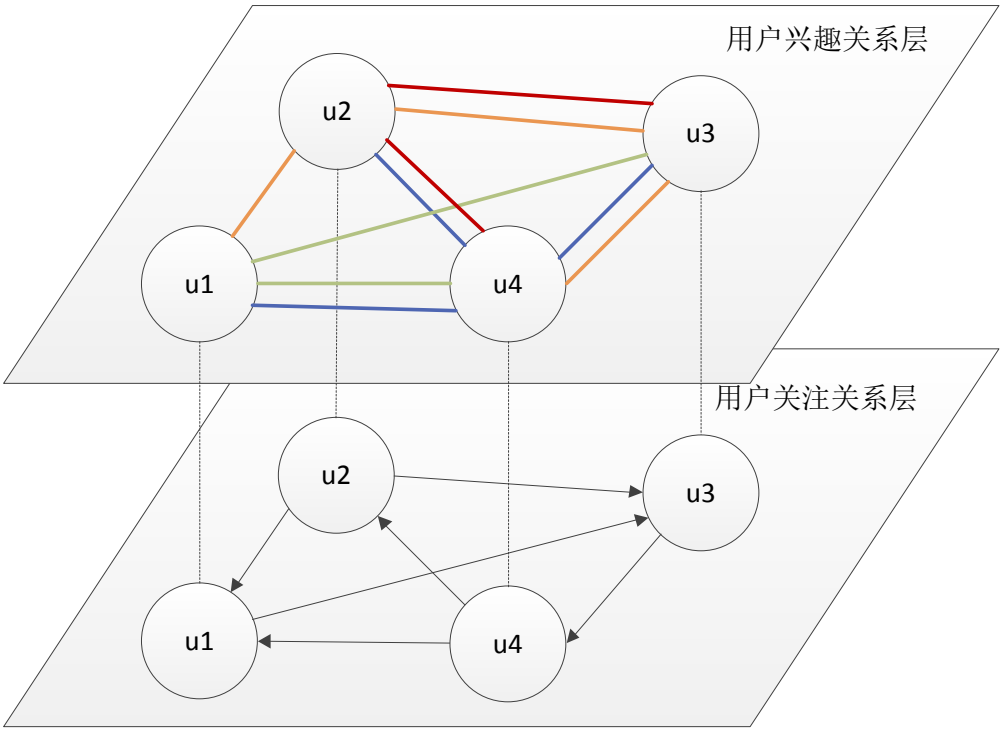


图 2.4: 基于向量表示的兴趣覆盖网络实例

对于两个用户 u_i 和 u_j，他们的兴趣向量分别是 d_{u_i} 和 d_{u_j}，如果存在 k $(1 \leq k \leq |\mathcal{I}|)$，满足条件：

$$w_{u_i,k} > \epsilon \text{ 并且 } w_{u_j,k} > \epsilon$$

那么称用户 u_i 和用户 u_j 分享共同的兴趣点 in_k。于是在兴趣覆盖网络中，这两个节点之间存在一条边 $v_{i,j}^k : u_i \overset{k}{\leftrightarrow} u_j$。兴趣关系集 V_i 则表示为：

$$V_i = \{v_{i,j}^k | u_i \overset{k}{\leftrightarrow} u_j, 1 \leq i, j \leq n\}$$

用户集合与兴趣关系集合组成了兴趣覆盖网络 $G_i = \{U, V_i\}$。

如图 2.4所示，在一个简化的四节点的网络结构中，可以看到在用户关注关系层上，u_1, u_2, u_3, u_4 四个用户之间存在着单向的关注关系，但是在这一层上不能直观地看到用户之间存在哪种相关性，这导致了在路由搜索的时候性能下降（在搜索过程中再进行匹配会很慢）。所以，在用户关注关系层上添加了一层用户兴趣关系层，在这一层中，四个用户相对位置并没变化，但是他们之间多出了几条彩色的边，即共同兴趣关系边，这里不同的颜色表示不同的兴趣点。两个用户之间可以存在一条或多条兴趣边，因为他们之间可能存在的共同兴趣不止一个。当然，在这一层中，节点之间存在多条边的关系不能在后续的搜索过程中进行路由，甚至会造成多重环的问题。所以，在计算过程中，可以将两节点间多条边转

换成一条权重为向量的边，具体计算方式会在第五章中详细阐述。

2.4 小结

在这一章中，简单阐述了基于 P2P 网络的信息自主流动机制的方案，在这一机制中主要涉及三大模块：文本信息的分析与建模、用户兴趣的表示与匹配和 P2P 网络的构建与发现。其中，在文本信息分析与建模的问题中，分析了直接用 VSM 模型来描述互联网资源的弊端，并且根据问题的根结所在，提出了用主题来表示资源是一个更好的方法。一方面对文本的高维度进行了降维，另一方面也表达出了文本隐藏在背后的含义。然后，通过模拟文本的生成过程，从而解释了对文本主题求解的基本方案。在用户兴趣表示与匹配的问题中，首先基于兴趣之间的相关性和兴趣动态变化两大特点，分别提出了两种解决方案：静态兴趣分类树和动态兴趣伸展树。在 P2P 网络的构建与发现问题中，引入了兴趣覆盖网络这一概念，并且用简单的基于 VSM 的兴趣模型来说明兴趣覆盖网络的直观意义和表示方法。

第 3 章　互联网文本信息分析与建模的研究

本章将从互联网文本信息的特征入手，详细分析与研究各种特征下的文本建模方法。根据第二章的分析，工作主要集中在对文本的主题进行建模与分析，根据不同文本的特征，包括长文本、短文本、文本间相关性以及动态文本等方面，提出有针对性的主题分析方法。利用主题模型对文本进行建模的好处有两个，一是可以对高维度的词汇向量进行降维，也就是用低维度的主题向量来表示文档；二是完全遵循用户生成文档的习惯，一般会先选取几个文档的主题，然后根据主题来生成单词，而主题模型则是很好地模拟了这一过程。

3.1　长文本信息的分析与建模方法

所谓的长文本信息一般是指新闻、博客等篇幅相对较长的文本，而评论、微博、弹幕等文本则属于短文本，关于短文本的分析将在下一节进行讨论。在分析之前，首先给出长文本分析问题的形式化描述：

定义 3.1 (长文本主题识别问题) *给定大小为 K 的主题集合 $\Phi = \{\vec{\varphi_1}, \vec{\varphi_2}, ..., \vec{\varphi_K}\}$，以及文档 $d = \vec{w} = (w_1, w_2, ..., w_{|d|})$，求出文档 d 的主题向量 $\vec{\tau} = (\tau_{k_1}, \tau_{k_2}, ..., \tau_{k_l})$，使得在 l 最小的情况下满足 $\epsilon < (\sum_{i=1}^{l} \tau_{k_i}) \leq 1$ $(\tau_{k_i} \in [0,1], k_i \in [1,K])$。*

在长文本分析中，需要解决的问题是如何识别出给定的文档主题，而主题集合可以通过已有的主题模型对语料库进行训练来得到，因此在问题定义中，主题向量 $\vec{\varphi}$ 被认为是已知条件。这里问题的难点在于如何将已知的主题赋给新来的文档，找出文档中权重排名前 l 的主题，并且使权重之和大于一个给定的阈值 ϵ，ϵ 的含义是选出的文档能在多大的程度上表示原文。其中 τ_{k_i} 表示主题编号为 k_i 在这篇文档中的权重值，i 表示这个权重值从高到低排在第 i 位。

3.1.1　基于相似度的长文本主题识别算法

这个算法的思路是将主题集合中的所有主题分布与输入文档计算余弦相似度，从而得到一个相似度向量，然后对这个向量进行归一化处理，然后从大到小对向量中的每个值进行排序，取出前 l 个总和达到 ϵ 的主题，从而得到主题向量。

从图 3.1中可以发现，主题集合中的 K 个主题分布 $\vec{\phi_1}, \vec{\phi_2}, ..., \vec{\phi_K}$ 在经过 LDA 推理后，其每个值得到的都是关于词汇表 V 中词汇的权重，从而组成一个 $|V|$ 维的向量，这与文档向量 $\vec{w}$ 的维数是一样的，当然这里的 $\vec{w} = (n_1, n_2, ..., n_{|V|})$，即对词汇表中每个词汇的频数（因为这里只有一篇文档，没有办法将文档向量表示成 TF-IDF 值）。然后通过计算主题分布于文档向量间的相似度 $d_k = sim(\vec{\phi_k}, \vec{w})$，得到一个新的 K 维向量，记为 $\vec{s} = (d_1, d_2, ..., d_K)$，但是这个向量中每个值之和

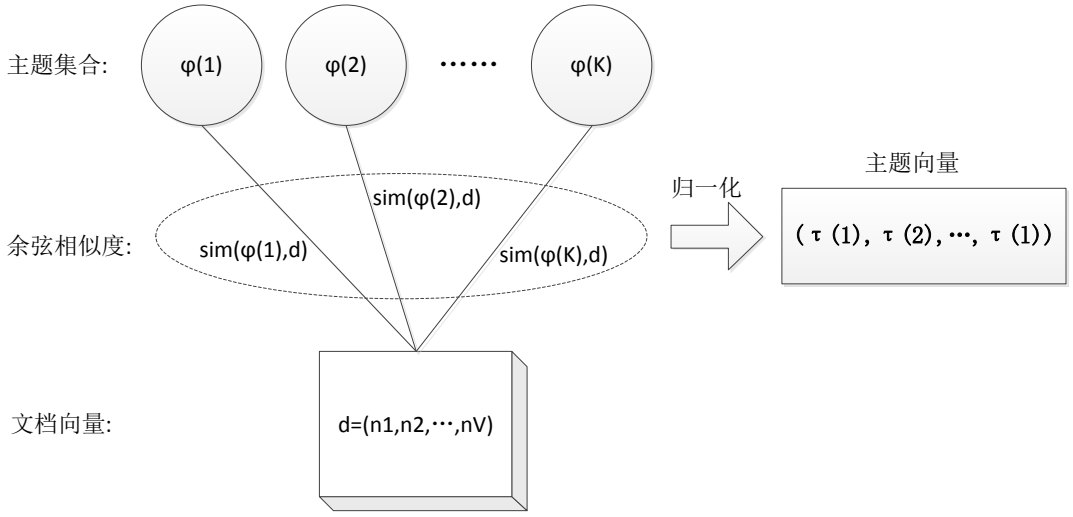


图 3.1: 基于相似度的长文本主题识别算法过程示例

并不为 1，因此要进行一次归一化操作从而得到一个修正后的向量 $\vec{s'}$：

$$\vec{s'} = (d'_1, d'_2, ..., d'_K),\ \ (d'_k = \frac{d_k}{\sum_{i=1}^{K} d_i}, k \in [1, K]) \tag{3.1}$$

其中 d'_k 表示文档 d 拥有第 k 个主题的权重。然后，对 $\vec{s'}$ 中的每个值进行从大到小的排序，排序后得到向量 $\vec{s''} = (d_{k_1}, d_{k_2}, ..., d_{k_K})$，其中 d_{k_i} 表示文档中编号为 k_i 的主题占有的权重，i 表示该主题所占的权重在文档 d 中的重要性排在第 i 位。最后，对于给定的 ϵ，在 $\vec{s''}$ 中选择前 l 个主题且满足 $\sum_{i=1}^{l} d_{k_i} > \epsilon$。这时文档 d 的主题向量 $\vec{\tau}$ 为：

$$\vec{\tau} = (d_{k_1}, d_{k_2}, ..., d_{k_l})\ \ (l \in [1, K]) \tag{3.2}$$

这个算法的时间复杂度是 $O(K \cdot |V|)$，因为在计算主题分布和文档的相似度时，消耗的时间是 $O(K \cdot |V|)$，在之后对向量权重进行排序的时候，平均时间复杂度达到 $O(K \cdot log\ K)$，很显然 $log\ K \ll |V|$，所以总共的时间复杂度是 $O(K \cdot |V|)$。

3.1.2 基于标签的主题分析技术

对于上述提出的算法，可以基本实现对新文本进行主题标注的功能，但是上述算法存在一个很重要的问题，准确地说应该是主体模型 LDA 导致的一个问题。因为 LDA 是无监督学习，与 K-mean 等聚类算法相似，虽然它们可以计算出最后 K 个主题分布，但是对于每个主题分布的实际含义还需要人工标注，通过观察主题分布中的主题词来判断用什么词可以概括相应的主题。如表格 3.1所示，第一行中的主题名称都是有人工标注的，这无疑造成了很大的麻烦，原因如下。

- 自动化：在信息自主流动的机制中，由于主题数量巨大，主题也会不断地更

表 3.1: 主题模型训练后的主题示例

“艺术”	“财务”	“孩子”	“教育”
新的	百万	小伙伴	学校
电影	税收	女性	学生
展览	项目	人们	大学
音乐	上亿	多年	教育
影片	政府	家庭	教师
播放	每年	工作	高等
乐器	开销	家长	公开的
最好的	全新的	说	多媒体
演员	省市	回家	基础的
首次	计划	福利	试卷
戛纳	资金	爸爸	丰富的
剧院	扶植	比例	电子化
电影院	企业	关心	准时
爱	繁荣	生活	兴趣
票房	单位	愉快的	山区

新，因此用户不适合每次都使用手动标注，更好的办法是将更多的任务交给每个节点的智能系统进行处理。

- 歧义性：在表格中可以看到，虽然第一列标注为艺术，但是这一列的主题词也可以换成一个更加小的概念，比如电影艺术等等。同样第二列主题标注为财务，但是同样也可以认为是预算、经济等相同意义的词。

对于如今的互联网文本来说，文本资源其实不仅仅包含纯文本，而且还有一些元数据，其中最常见的新闻和博客资源中还包含了标签信息。因此，现在的问题是如何利用这些文本的标签的信息来重新训练主题，使得训练出来的主题都自动分配到一些标签。这里有两个问题，一个是如何通过语料库训练出带有标签的主题，另一个是如何给文档赋上多个带有标签的主题。对于第一个问题，一个比较成熟的方法是利用 Labeled-LDA 模型，或者 L-LDA[65] 重新训练这些主题。这里先对 L-LDA 模型进行简单的说明，进而对比该模型生成的主题与传统 LDA 生成的主题的不同之处，以及如何在下文中的进行修改使用。

L-LDA 同样是一个概率图模型，其模拟了一个带有标签的文档集合的生成过程，换句话说，其充分利用了互联网上文档自带的标签数据。与 LDA 相同的地方是在生成过程中，每个单词依然是从一个选定的主题中生成的，但不同的是，

该模型是一种有监督学习算法，通过分析标签与主题之间的对应关系来判断文档对应的标签是什么。接着前文对 LDA 的形式化定义，现在假设共有 K 个不同的标签对应 K 个不同的主题，完整的生成过程如下：首先从给定的参数 $\vec{\beta}$ 的 Dirichlet 分布中抽样得到 K 个主题分布 $\Phi = \{\vec{\phi_1}, \vec{\phi_2}, ..., \vec{\phi_K}\}$，这一步与 LDA 中一样。与 LDA 直接从给定参数 $\vec{\alpha}$ 的 Dirichlet 分布中取样得到的文档 -主题分布 $\Theta = \{\vec{\theta_1}, \vec{\theta_2}, ..., \vec{\theta_W}\}$ 不同的是，在 L-LDA 中先要限制 Θ 使其能与标签一一对应。于是，在这个过程中先生成文档 d 的标签 $\vec{\Lambda_d}$，使其从伯努利分布中进行抽样，即

$$\vec{\Lambda_{d,k}} \in \{0, 1\} \sim Bernoulli(\cdot|\Gamma_k)$$

其中，Γ_k 是标签的先验概率。接着，定义文档的标签向量为 $\vec{\lambda_d} = \{k|\Lambda_{d,k} = 1\}$，从而可以为每个文档 d 定义一个 $M_d \times K$ 维的标签映射矩阵 L_d，其中 $M_d = |\vec{\lambda_d}|$ 对于矩阵中的每一行 $i \in \{1, ..., M_d\}$ 和每一列 $j \in \{1, ..., K\}$ 有：

$$L_{d,i,j} = \begin{cases} 1 & \text{if } \lambda_{d,i} = j \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

换句话说，如果矩阵中的第 i 行在第 j 列的值是 1 的话只有当第 i 个文档标签 $\lambda_{d,i}$ 等于编号为 j 的主题，否则就是 0，这样就限制了主题和标签一一对应的关系。同时，这个矩阵 L_d 可以用来对 Dirichlet 分布中的超参数 $\vec{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_K)^T$ 进行降维，得到文档 d 自己的超参数 $\vec{\alpha}_d$：

$$\vec{\alpha}_d = L_d \times \vec{\alpha} = (\alpha_{\lambda_{d,1}}, ..., \alpha_{\lambda_{d,M_d}})^T \tag{3.4}$$

很明显可以看到，这个向量的维度原本是受主题数量决定的，现在转而由标签数量决定。在完成上面这个额外的过程后，原本的文档 -主题分布 $\vec{\theta}_d$ 可以从改进后的超参数 $\vec{\alpha}_d$ 来取样。最后根据 LDA 中的生成单词方法继续执行。关于这个模型的参数训练方法不再详细赘述，推理过程将直接参照文献 [65] 中的方法。

如图 3.2所示，左半部分为 LDA 模型的层次结构，右半部分为 L-LDA 模型的层次结构。通过对比可以发现，两种模型的目标都是为了推理计算出主题的隐藏层，但是 L-LDA 比 LDA 多出了一层标签（中间三角形）。而这一层的目的也十分地明确，一方面是为了找到与主题一一对应的关系，弥补原本 LDA 无监督学习导致的主题名称不明确的问题；另一方面是在训练过程中减少了工作量，如上述公式中 $\vec{\alpha}$ 由原本的 K 维降到了向量 $\vec{\alpha}_d$ 的 M_d 维，因此添加标签这层导致了一个一举两得的结果。

3.1.3 基于标签的长文本主题识别算法

现在利用 L-LDA 训练出的带有标签的主题对长文本主题识别问题的算法进行修正：首先保持估计出来的主题 -单词分布 Φ 保持不变，对新文档的每个单词

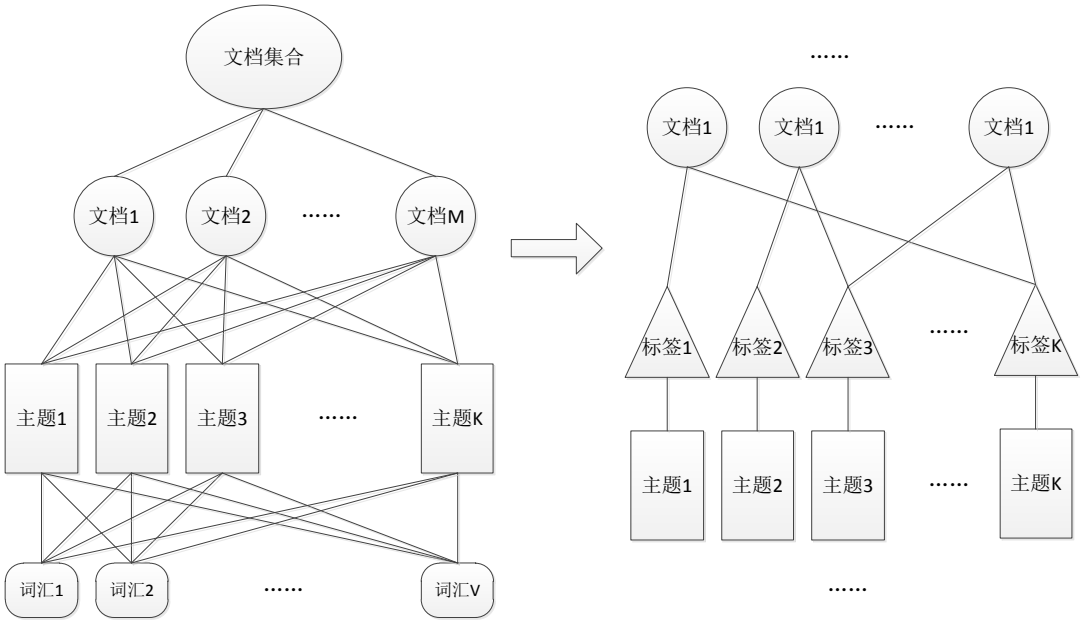


图 3.2: LDA 模型与 L-LDA 模型之间的区别

随机一个主题编号，然后通过 Gibbs Sampling 采样出每个单词新的主题编号，反复迭代直至 Gibbs Sampling 收敛，最后统计该文档中主题的分布 $\vec{\theta}_d$，选取前 l 个主题使得它们的权重达到 ϵ，这样就初步推理出了文档的主题集合的一个解。随后，再按照之前基于相似度的主题识别算法中同样的步骤计算出该文档的主题集合的第二个解。通过选取两个解集合中的交集主题，并找到这些主题对应的标签，从而得到文档的标签集合。之所以计算两次主题向量并再取交集的原因是：

- 文档标签数量很少：因为 L-LDA 模型适用于互联网上带有标签的文档，而通过统计和观察，这些文档的平均标签数相对于主题的数量是十分少的，只有 2-3 个。但是，在第一个主题识别算法中，主题选取的数量依赖于给定的阈值 ϵ，当 ϵ 达到 95% 的时候，根据普通 LDA 对文本数据集的实验结果来看，每个文档会 7-8 个主题，因此是不符合实际情况的。通过去交集的办法可以过滤掉一些可能是噪声的主题。

- 主题数量的不确定性：因为在两个模型中，主题数量 K 都是人工确定给定，因此 K 的大小会间接决定了文档最后的标签数量。而 L-LDA 是根据已有文档的标签集合进行有监督的学习，因此在控制每个文档的标签数量上更加出色。

3.2 短文本信息的分析与建模方法

互联网中的短文本一般包括各种资源的评论、微博的内容，甚至是资源的元数据，针对这些数据的分析目前已经有比较的研究，在此就不再罗列。本节要研

究的短文本信息是一种新型出现的视频弹幕，考虑到这种弹幕数据独有的特性，因此可以用来对视频精彩片段进行提取和索引。相比于利用图像技术进行视频精彩镜头提取的工作，利用弹幕短文本来做的优势是：

1. 图像本身很难被计算机充分理解，因为视频中讲述的故事很难用像素点来表示，计算机不像人类一样可以通过图像描述的方式来理解这些隐藏的主题。

2. 当用户对视频进行查询时，一般是通过输入文本信息的，因为这是一种简单而有效的交互方式，所以这不可避免需要进行文本和图像之间的转换。虽然现在已经有技术来实现这一功能，但是相比文本的匹配过程，这种技术显然是更加耗时耗资源的。

基于上述理由，一种更好的方式是利用弹幕这种带有时间戳的短文本对视频的精彩镜头进行描述。这节的研究成果所发表的论文已经被 ACM MobiHoc 2015 - HotPOST '15 录用，其中 MobiHoc 是 CCF 推荐的 B 类会议，HotPOST 是其下的一个 workshop。在这一节中，将简单说明该论文的内容。

3.2.1 视频精彩镜头的抽取问题

为了简化问题，假设弹幕的文本内容可以充分用一个关键词来表示，关键词来自一个充满情感词的词典，这样做的理由是：

- 直观上来看，每个用户在看完一个精彩镜头之后都会触发最多一种情感，所以弹幕的内容应该是单一的。比如看完一个恐怖镜头之后，人可能会感到恐惧。

- 统计上来说，弹幕是典型的短文本，在本文的数据集中平均只有 6.28 个中文字符。

首先来形式化定义一下待解决的问题。一个时间长度为 T_v 秒的视频 v 可以被分割成一个连续小段的序列，$v = f_{v,1}, f_{v,2}, ..., f_{v,N_v}$，其中小段 $f_{v,i}\ (1 \leq i \leq N_v)$ 的时间长度是固定的 T_f，比如是 256 毫秒。显然，视频 v 中小段的数量 $N_v = T_v/T_f$，其中最后一个小段可能比 T_f 短。类似的，精彩镜头 s_v 也是视频 v 中的一个连续子序列，记作 $s_v = \{f_{v,i}, f_{v,i+1}, ..., f_{v,j}\}\ (1 \leq i \leq j \leq N_v)$。基于之前的假设，本文的的弹幕表示成三元组 $c_{v,i} = (w, t_0, T_c)$，其中 w_k 是来自词典 $\mathcal{W}$ 的情感词，t_0 是弹幕在视频 v 中的开始时间戳，T_c 是弹幕出现在屏幕上的持续时间，每个视频的弹幕集合通常包括了很多不同的关键词和开始时间戳。对于视频 v 中的弹幕集合记为 $\vec{c_v} = \{c_{v,1}, c_{v,2}, ..., c_{v,M_v}\}$。所有符号见表 3.2所示，精彩镜头、视频小段和弹幕的关系如图 3.3所示。

表 3.2: 本模型中用到的符号

SYMBOL	DESCRIPTION
v	time-sync commented video
T_v	video length (time duration)
N_v	total number of frames in video v
$f_{v,i}$	i-th frame in video v
T_f	frame length (time span)
s_v	highlight shot in video v
$c_{v,i}$	i-th TSC in video v
w	sentimental keyword of TSC
t_0	start time stamp of TSC
T_c	TSC length (time span)
M_v	total number of TSCs in video v
e_v	global emotional feature of video v
$\mathcal{V}$	video corpora of size $\|\mathcal{V}\|$
$\mathcal{C}$	TSC corpora of size $\|\mathcal{V}\|$
$\mathcal{W}$	sentimental keywords dictionary of size $\|\mathcal{W}\|$

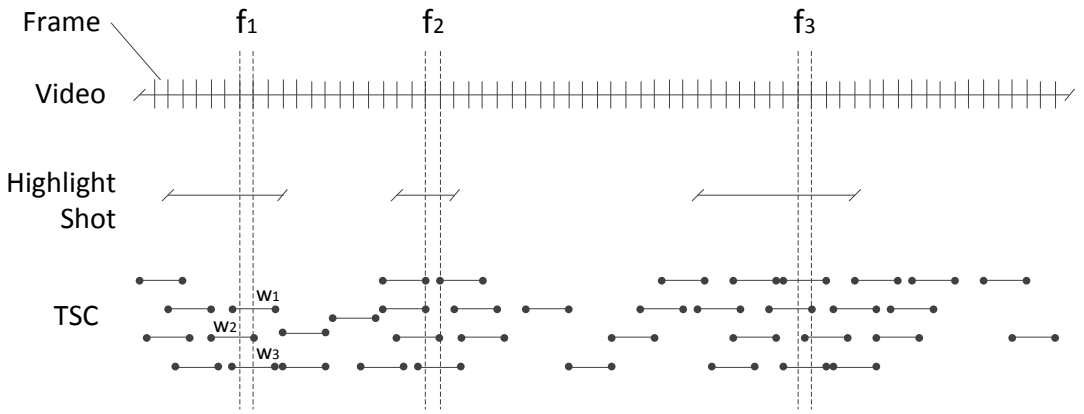


图 3.3: 视频、小段、镜头和弹幕之间的关系

下文将主要关注视频的精彩镜头抽取和查询的过程，问题定义如下：

定义 3.2 (视频精彩镜头的抽取问题) *给定视频 v 和其弹幕集合 $\vec{c_v}$，求出可能的精彩镜头集合 $\vec{s_v} = \{s_{v,1}, s_{v,2}, ...\}$，且任何两个 $s_{v,i}$ 和 $s_{v,j}$ 是不相交的。*

定义 3.3 (视频精彩镜头的查询问题) *给定查询的精彩镜头 s_q、视频 v 和对应的弹幕 $\vec{c_v}$，求出视频 v 中的具有相似情感的精彩镜头。*

为了解决上述两个问题，我们需要做的是：

1. 如何利用弹幕来描述小段、镜头和视频；

2. 如何计算小段与小段、镜头与镜头之间的相似度；

3. 如何自动发现精彩镜头的边界；

3.2.2 弹幕短文本的特征

对弹幕这种短文本的特征进行研究有助于计算视频小段间的相似度。因为弹幕之前被定义成一个三元组 $c = (w, t_0, T_c)$，假设关键词 w 已经从现有成熟的技术从原始评论文本转换来的。经验上，用户只有当他被当前或者临近的前几个镜头所感染，这说明了弹幕只是在时间段 $[t_0, t_0 + T_c]$ 的开头更加有效。所以，这里提出了一种弹幕强度在时间段 $[t_0, t_0 + T_c]$ 上的衰退函数 g_c：

$$g_c(t) = \begin{cases} T_c^{t_0 - t}, & \text{if } t_0 \le t \le t_0 + T_c \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

可以看到，弹幕在 t_0 的时候强度最大，在之后的 T_c 秒内，强度逐渐趋向于 0，这解释了弹幕在不同时间戳的时候是如何反应用户情感的。

3.2.3 视频小段的表示和相似度

如图 3.3表示，每个小段都或多或少包含了一些弹幕，所以最简单的方式来描述小段是用一个关于关键词词频的向量 $f = (n_1, n_2, ..., n_{|\mathcal{W}|})$，其中 $n_i (1 \le i \le |\mathcal{W}|)$ 是词典中第 i 个关键词在小段中出现的数量。举例来说，在小段 f_1 的区间上，有三个关键词 $w1$、$w2$ 和 $w3$，因此 $f_1 = (1, 1, 1, 0, 0, ...)$。Jaccard 相似度可以用于计算小段之间的相似程度，但是当两个小段中的关键词都是同义词时，就不适用了。根据弹幕强度衰退函数来看，关键词的强度是一个更好的选择，另一方面，弹幕在每个视频小段内的数量都远小于 M_v，因此会导致小段的歧义性。

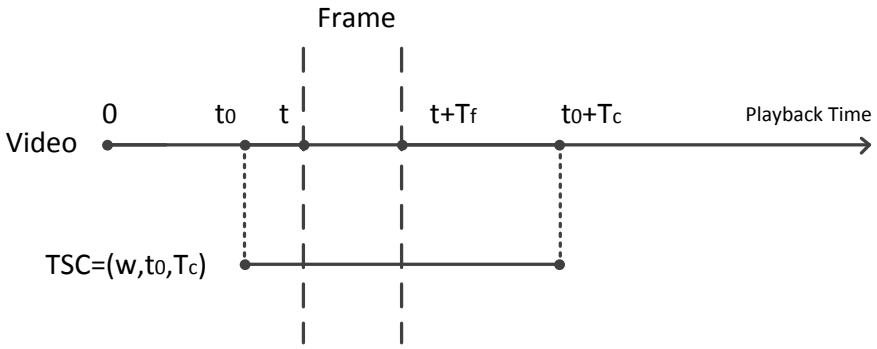


图 3.4: 对于关键词强度向量的每一维度，找出所有的弹幕使得其时间区间与小段的时间区间相交

所以，在时刻 t，小段 $f_v = (g_1, g_2, ..., g_{|\mathcal{W}|})$ 中的关键词强度向量可以计算为：

$$g_i = \sum_{c \in \vec{c_f}} g_c(t) \tag{3.6}$$

其中 $\vec{c_f}$ 是视频 v 的弹幕的集合:

$$\vec{c_f} = \{c | w = w_i, t - T_c < t_0 < t + T_f\}$$

图 3.4解释了弹幕关键词关于小段的表示方式。

但是，利用视频小段的关键词强度向量来计算 Jaccard 相似度仍然有不足，因为每个小段的弹幕数量是十分有限的。为了解决这个问题，引入一个新的变量 e_v，称为视频 v 的全局情感特征。在实际的例子中，全局情感特征与用户在看完一个精彩镜头之后抒发的情感是一致的。同时，精彩镜头中的局部情感特征在一定程度上组成了全局情感特征，而且精彩镜头和视频都是由小段组成的。鉴于此原因，特征相似度的计算在小段和全局情感特征间进行，而不是小段间的相似度。

为了获得视频 v 的全局情感特征，这里使用了传统的 LDA 主题模型。原本文本中的文档 -主题 -词汇三层关系转换成了视频 -特征 -弹幕的关系，其中弹幕中的关键词与单词是类似的。另一方面，这里的视频仅仅考虑文本信息，而排出了图像信息，所以它可以表示为一个带有权重的全局情感向量。类似于文档的主题，全局情感特征是一个隐藏变量，并定义为关于关键词的分布，$e_v = (n_1, n_2, ..., n_{|\mathcal{W}|})$，为了简化问题，假设总共只有 K 个不同的特征。

在应用 LDA 之前，弹幕生成的过程与文档的生成过程有两个小区别：

- 精彩镜头的影响：精彩镜头是视频中额外的特殊元素，而在文件中没有完全相同的概念。正如前文所说，弹幕的主要是受精彩镜头影响，更具体地来说，是局部情感特征。为了简化问题，这里特意把弹幕的生成，通过大体上，由视频，即全局情感特征的影响。在实验中，我们表明，这种近似计算还执行比其他传统的方法要好得多。

- 弹幕密度影响：弹幕的密度反映了弹幕视频的特征，因为一些现有的工作使用频率提取镜头。不同于文档，其中的每一个单词是依次产生并均匀地分布的，而弹幕生成的不确定性取决于用户的感觉是否将被某个镜头所触发。例如，弹幕更可能在精彩镜头的区间内生成。

现在，可以开始衡量小段和全局情感特征之间的相似性。鉴于视频集合 $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$，而弹幕集合 $\mathcal{C} = \{\vec{c_{v_1}}\}$，其中的关键字来自大小为 $|\mathcal{W}|$ 的字典，可以得到大小为 $|\mathcal{V}|$*$|\mathcal{W}|$ 的视频 -弹幕矩阵，矩阵元素的值是关键字的数量。为了分解的视频 -弹幕矩阵成两个矩阵，大小为 $|\mathcal{V}|$*K 的视频 -特征矩阵和大小为 K*$|\mathcal{W}|$ 的特征 -弹幕矩阵。在这种情况下应用的时 LDA，而不是 SVD，因为 LDA 在主题分析上更好，尽管其速度推理过程比 SVD 要慢一点。在此之后，我们可以得到小段 f_v 的特征相似度 d_f，通过 f_v 的关键字强度向量和全局情感特征之间 e_v：

$$d_{f_v} = cos(f_v, e_v) = \frac{\sum_{i=1}^{|\mathcal{W}|} g_i \times n_i}{\sqrt{\sum (g_i)^2} \times \sqrt{\sum (n_i)^2}} \tag{3.7}$$

3.2.4 精彩镜头边界检测

之后，根据给定的全局情感特征，得到特征相似度 d_f，就可以通过算法 1检测出精彩镜头的范围。该算法表明，在每次迭代期间，小段的最大的特征相似度在全局情感特征中是第一次被选出的，并且从中间向两侧依次比较。此外，还需要设置两个阈值。ϵ 控制了下边界的最大相似特征 $d_{f_{v,k}}$，这是因为如果 $d_{fv,k}$ 过小，可能没有具有这种情感的精彩镜头存在，所以应该忽略这个特征。δ 控制了潜在的精彩镜头的最左边和最右边小段的边界。更好的方法是根据不同的视频，设置动态的 δ，因为阈值会随着弹幕的数量和视频的长度变化。

3.2.5 精彩镜头相似度计算

一旦精彩镜头从视频提取出来之后，就可以计算出该精彩镜头的局部特征向量 $\vec{e_s} = (w_1, w_2, ..., w_K)$，其中每个 w_i 是精彩镜头 s 的每一个特征维度的权重。首先，计算出精彩镜头的关键词词频 $n_s = (n_1, n_2, ..., n_{|\mathcal{W}|})$。然后，对每一个视频的全局情感向量的特征 $e_{v,k} = (n_1, n_2, ..., n_{|\mathcal{W}|})$ 和权重 $w_{v,k}$，计算余弦相似度：

$$w_k = w_{v,k} \times cos(n_s, e_{v,i}), (1 \leq k \leq K) \tag{3.8}$$

然后，对给定的两个精彩镜头，各自计算它们的局部特征向量 $\vec{e_{s_1}}$ 和 $\vec{e_{s_2}}$ 之后，就可以衡量它们之间在语义上的相似度了。

算法 1 精彩镜头的边界检测算法

foreach *frame* $f_{v,i} \in v$ **do**
  calculate keyword strength vector $f_{v,i} = (g_1, g_2, ..., g_{|\mathcal{W}|})$;
end
foreach *global emotional feature* $e_{v,j} \in e_v$ **do**
  foreach *each frame* $f_{v,i} \in v$ **do**
    calculate feature similarity $d_{f_{v,i}} = cos(f_{v,i}, e_{v,j})$;
  end
  find $max(d_{f_{v,k}})$ indexed by k;
  if $d_{f_{v,k}} < \epsilon$ **then** continue;
  set $n_l := k - 1$;
  while $n_l \geq 1$ *and* $|d_{f_{v,n_l}} - d_{f_{v,n_l+1}}| < \delta$ **do** $n_l := n_l - 1$;
  set $n_r := k + 1$;
  while $n_r \leq N_v$ *and* $|d_{f_{v,n_r}} - d_{f_{v,n_r-1}}| < \delta$ **do** $n_r := n_r + 1$;
  $\{f_{v,n_l}, ..., f_{v,n_r}\}$ is the highlight shot with feature $e_{v,j}$;
end

3.3 文本网的分析与建模方法

在上面几节中讨论的文本都是相互独立的，即没有考虑文本之间的相关性，因此，在这一节中，将主要讨论文本间通过链接相连的情况。这种类型的文本在互联网上尤为明显，因为用户在网上写文章的时候可以加入超链接，相反当用户在互联网浏览文章的时候，比如是在学习一项新的编程技能的时候，常常会从技术博客开始着手，由于这类博客中的链接非常多，因此如何从这一组文本网络中分析出主题分布是一个值得研究的问题。问题的形式化定义如下：

3.3.1 有向文本网的主题识别问题描述

定义 3.4 (有向文本网的主题识别问题) *给定文本集合* $D = \{d_1, d_2, ..., d_{|D|}\}$，*有向文本边集合* $E = \{e_{i,j} : d_i \rightarrow d_j | d_i, d_j \in D\}$，*以及带有标签的主题分布* $\Phi = \{\vec{\phi}_1, ..., \vec{\phi}_K\}$，*求出文本网* $G = D, E$ *的主题向量* $\vec{\tau} = (\tau_{k_1}, ..., \tau_{k_l})$。

在这个问题中，不能像对待相互独立的长文本那样对每个文档依次计算主题向量，再通过平均等方式把所有文档的主题向量整合起来；相反，也不能把所有文档整合看成一个词袋模型，然后计算出一个主题向量。这其中的主要原因是文档间存在一定的相关性，而相关性是由文档间的链接所决定的，但是在之前的主

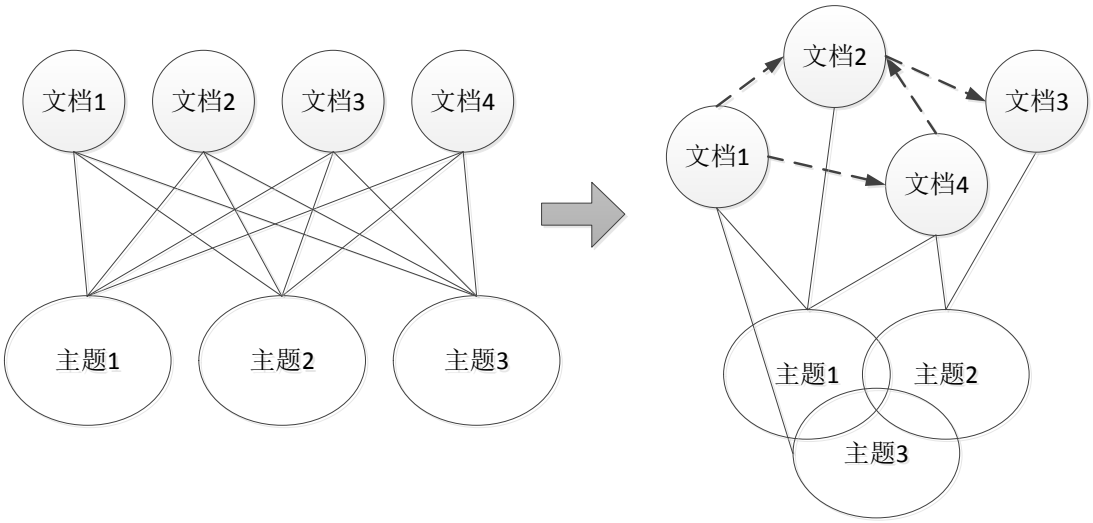


图 3.5: 文本网络的主题关系示例

题分析时并没有把链接作为一个变量加入模型进行训练，因而直接使用上文讨论的模型会有失偏差。如图 3.5所示，文本网的主题直观上与普通文本的主题是有一定的区别，在右边半部分中，文本之间由虚线相连表示文本间存在链接，而这直接导致了主题间也存在一部分交集。

一个可行的解决方案是利用 RTM[34] 模型重新对文本以及链接进行训练，得到新的主题分布。但是，这个方法存在两个问题，一是该模型得到的主题没有标签，二是很难结合已经通过 L-LDA 训练出的主题对标签进行预测。因此，下文提出了另一种折衷的方法，一方面直接利用已有的带有标签的主题进行预测，另一方面则是考虑到了文本间由链接引起的相关性问题。

3.3.2 基于贝叶斯网络的有向文本网主题识别算法

为了解决上述问题，这里提出了一种用于识别有向文本网主题的方法，具体流程将结合图 3.5中右半部分的实例进行说明。

1. 假设 $\vec{\tau}_G = p(d_1, d_2, d_3, d_4)$ 为图中文本网络 G 的主题分布，根据 Chain Rule，可以将该联合分布进行分解，得到连乘的形式：

$$\vec{\tau}_G = p(d_1) \cdot p(d_2|d_1, d_4) \cdot p(d_3|d_2) \cdot p(d_4|d_1) \tag{3.9}$$

根据贝叶斯定理，上式中的后三项变为：

$$\begin{aligned} p(d_2|d_1, d_4) &= \frac{p(d_2, d_1, d_4)}{p(d_1, d_4)} \\ p(d_3|d_2) &= \frac{p(d_3, d_2)}{p(d_2)} \\ p(d_4|d_1) &= \frac{p(d_4, d_1)}{p(d_1)} \end{aligned} \tag{3.10}$$

2. 对于一般的 $\vec{\tau}_{d_i|D} = \frac{p(D')}{p(D)}$ $(D' = \{d_i \cup D\})$，运算过程定义如下：

$$\begin{cases} \dfrac{p(D')}{p(D)} = (\dfrac{x_1}{\Delta}, ..., \dfrac{x_K}{\Delta}), \ (\Delta = \sum_{i=1}^{K} x_i) \\ x_i = \dfrac{1}{\Delta} \cdot cos(\vec{\phi}_i, (q_1, ..., q_V)^T), \ (i \in [1, K]) \\ q_j = \dfrac{count(v_j \in D')}{count(v_j \in D)}, \ (j \in [1, V]) \end{cases} \tag{3.11}$$

3. 根据第二步中的公式计算出公式 3.9中的四项的向量，然后对四个向量进行标量乘法，并且进行归一化，从而得到最终的文本网络的主题向量 $\vec{\tau}_G$。

下面对上述算法进行简单的说明。式中 $p(d_1)$ 表示文档 d_1 的 K 维主题向量 $\vec{\tau}_{d_1}$，向量中的每个值表示该主题在文档 d_1 中占有的权重。$p(d_2, d_1, d_4)$ 表示三个文档重新组成的文档 $D = \{d_2, d_1, d_4\}$，在不考虑文档之间的链接关系时，对应的 K 为主题向量 $\vec{\tau}_D$。而 $p(d_2|d_1, d_4)$ 表示在受文档 d_1 和 d_4 影响的情况下，文档 d_2 的主题向量 $\vec{\tau}_{d_2|d_1,d_4}$。为了计算这个向量的值，首先要构造表示这种链接关系的文本向量 $(q_1, ..., q_V)^T$，其第 i 值表示词汇 v_i 同时出现在集合 $\{d_2, d_1, d_4\}$ 中的数量与出现在集合 $\{d_1, d_4\}$ 中的数量的比值，然后用这个文本向量与每个给定的主题分布进行相似度的计算，从而最后得到结果。

3.4 文本流的分析与建模方法

在各种大规模应用随时间产生了大量的文本数据被连续产生，例如社会网络中海量数据流。通常大量文本流是由用户非常大规模的相互作用而产生，或者通过专门的组织来创造特定种类的内容。后一类的例子是新闻通讯社创作了大量的文本流。这样的文本流提供从效率的角度来看带来了前所未有的挑战。在这一节中，将对普通的文本流进行讨论，并从中挖掘和建立主题。

定义 3.5 (文本流的主题识别问题) *给定 M 个文本序列 $D = (d_1, t_1), ..., (d_M, t_M)$ $(t_1 < ... < t_M)$，以及大小为 K 主题分布 Φ，求文档序列的主题序列 $\{\vec{\tau}_1, ..., \vec{\tau}_T\}$ $(1 \leq T \leq M)$。*

如图 3.6所示，类比本章第二节对视频中精彩镜头的判断方法，这里的文本流于弹幕在特征上十分相似。显而易见的是，在给定时间段内，相关主题的特点会影响数据流的一些具体特征，这样的特征可以从一个聚类方法来找出。在文本的情况下的一个关键问题是确定时间文本流进化模式。进化分析在文本流的话题早期的调查可能会在 [66] 中找到。这样进化模式可以在各种各样的应用中，比如总结在新闻文章的事件和揭示研究趋势在科学文献中非常有用的。例如，一个事件可具有生命周期中的基本主题图案，例如一个特定的事件的开始，持续时

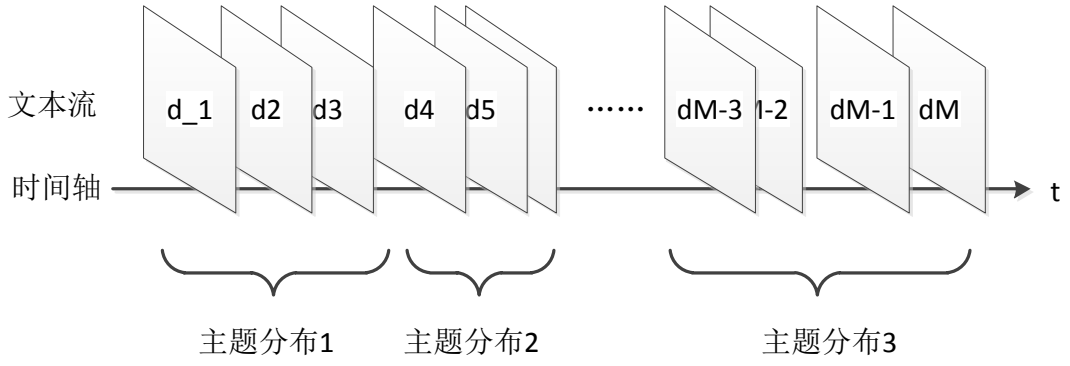


图 3.6: 文本流中的主题变化实例

间和结束。同样，在研究文献中特定主题的演变可能有生命周期中的不同主题是如何影响的。此问题主要包含三个主要部分：发现主题、建立主题演变、研究主题的生命周期。这几个问题与弹幕中的精彩镜头发现是极其相似的，因此完全可以使用之前提出的方法进行计算，主要分为三个步骤：

- 文本流分割：第一步先将文字流转换为多个具有固定的或可变的时间跨度的重叠子集，这一过程类比于对弹幕视频进行精彩镜头提取的工作。可以先将整个文本流按照时间小段进行切割，并通过相似度匹配算出大致的区间范围。另一种方法是直接对文本流中的所有文本进行聚类，从而可以实现同样的效果。

- 主题匹配：对每个子集进行主题识别，使用上述提供的带有标签的主题分布即可。这一过程类比于在弹幕视频中选出的精彩镜头进行评价的工作，对子集的主题识别方法与对精彩镜头的算法一致。

- 合并主题：最后，所有演变转移从这些主题的模式来确定，可以通过使用 KL 散度来进行评价，即计算两个主题之间的演变距离。在该相似度高于一个给定阈值的情况下，过渡被认为是有效的。

3.5 小结

本章的主要工作是对互联网中的文本信息进行分析和建模，考虑到与用户兴趣最直接相关的文本信息就是主题，因此分别对普通的长文本信息、短文本信息、文本网信息以及文本流等四种不同类型文本进行主题分析。

对于诸如博客和新闻等长文本信息时，考虑到它们大多情况下是相互独立的，因此可以借助 L-LDA 模型对长文本语料库训练出带有标签的主题。对于诸如评论和弹幕等短文本信息时，文本提出了一种基于弹幕的视频精彩镜头提取

方法，这样做的优势是可以利用文本信息直接对视频片段进行索引，而且计算速度远远快于传统的图像识别技术。对于诸如含有大量链接的网页等文本网信息时，本文提出了一种基于贝叶斯网络的算法用于识别有向文本网中包含的主题。最后，对于诸如故事类型的文本流信息时，可以直接借鉴对于弹幕视频的挖掘方法，进而得出在一定时间段内主题的演化过程。

第 4 章 用户兴趣表示与匹配的研究

本章将从用户兴趣的特征作为切入点，对每种特征进行深入分析，提出一套完整的兴趣表示方法。根据第二章中的讨论，兴趣具有最明显的两个特征是关联性和动态性。前者说明的是兴趣概念的关系如同名词之间的关系一样具有同义词、上义词和下义词等，文中提出了一种静态兴趣分类树的模型来充分表示这个特征。后者说明的是用户兴趣会随时间变化而变换，并且分为长期兴趣和短期兴趣，所以文中提出了一种动态兴趣伸展树的模型来解决这一变化的过程。此外，为了使系统能够自动地学习用户的兴趣爱好，以便将来可以为用户推荐兴趣（主题）相似的资源，文中提出了一种学习和推荐的方法来完成这个工作。

4.1 基于文本主题的兴趣点描述方法

在正式讨论兴趣树的细节之前，先对文本中涉及兴趣相关的概念进行说明和形式化定义。

给定大小为 V 的词汇表 $\mathcal{V} = \{v_1, ..., v_V\}$，对于大小为 K 的兴趣全集 $\mathcal{I} = \{\vec{in}_1, ..., \vec{in}_K\}$，其中 $\vec{in}_i$ $(i \in [1, K])$ 为第 i 个兴趣点。在本文中，兴趣点的名称 $name(\vec{in})$ 是一个字符串，兴趣点的值 $value(\vec{in})$ 看做是从兴趣变量 $\vec{x}$ 抽样出的一个观测值，兴趣变量 $\vec{x}$ 服从 Dirichlet 分布，即：

$$p(\vec{x}|\vec{\alpha}) = Dir(\vec{x}|\vec{\alpha}) = \frac{\prod_{i=1}^{V}\Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{V}\alpha_i)} \cdot \sum_{i=1}^{V} x_i^{\alpha_i - 1}, \ \ (\vec{\alpha} = (\alpha_1, ..., \alpha_V))$$

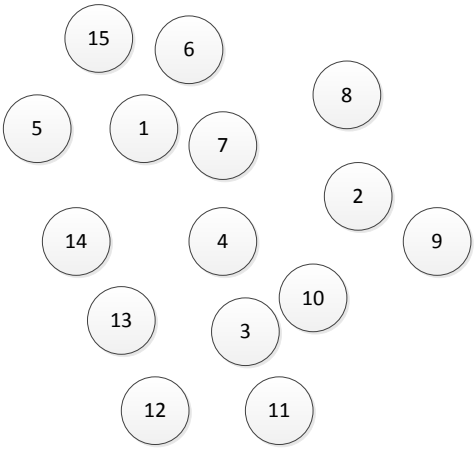
可以看出，兴趣点 $\vec{in}$ 是一个 V 维的向量，向量中每个值表示当前词汇对兴趣的权重。因此，对于两个兴趣点 $\vec{in}_i$ 和 $\vec{in}_j$\$，它们之间的相似度 $d_{i,j}$ 可以用余弦相似度来衡量：

$$d_{i,j} = cos(\vec{in}_i, \vec{in}_j) \tag{4.1}$$

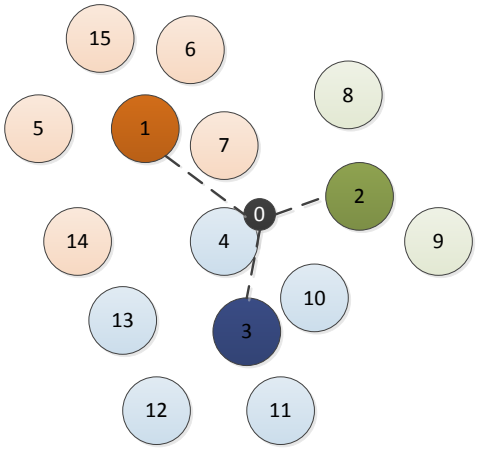
要说明的是，这里的兴趣点的表示方式本质上是继承自文本中的主题，在下文没有特别说明的情况下，可以认为兴趣点等价于主题。同时，关于如何估计兴趣变量 $\vec{x}$ 参数的方法就等同于如何在给定的语料库中训练出主题分布 Φ。所以，在这一章中，可以默认提取兴趣点的方式都已经存在，重点是如何利用这些兴趣点构建用户的兴趣树。

4.2 静态兴趣分类树的表示方法

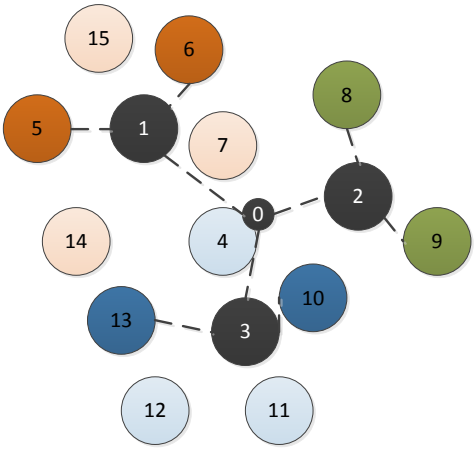
这一节主要针对兴趣间的关联性问题，对整个兴趣集建立一个描述模型，从而能够展现出兴趣间的关系。从兴趣点的名称来看，这些名称一般为名词，自然而然地想到了兴趣点之间应该是一种层次关系，于是本节提出了用静态兴趣分类树的方式进行描述整个兴趣集。



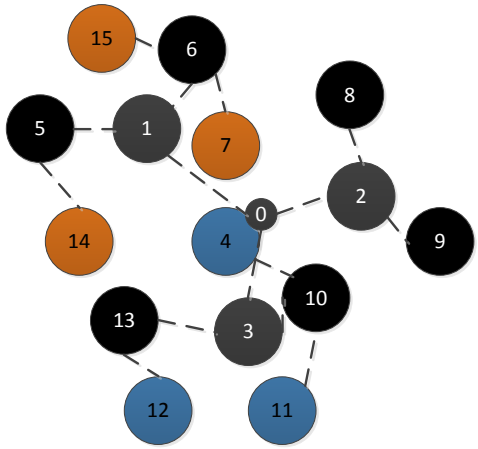
(A) 初始状态



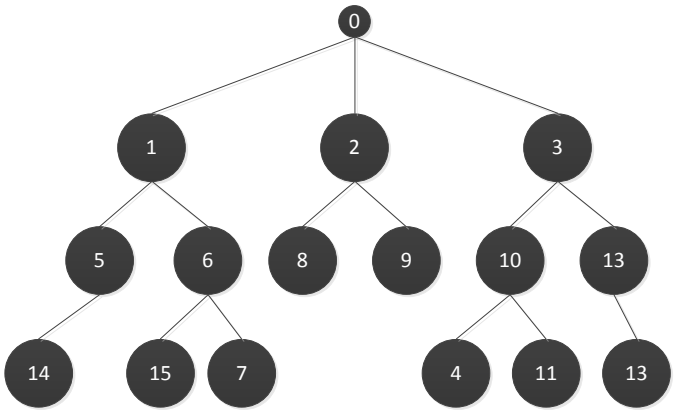
(B) 第一次聚类，K1=3



(C) 第二次聚类，K2=2



(D) 第三次聚类，K3=2



(E) 完成构建

图 4.1: 基于兴趣余弦相似度的静态兴趣分类树的自动构建示例

4.2.1 静态兴趣分类树的构建

在第二章中提出了几种构建静态兴趣分类树的方法，其中，人工构建法最为精确，但是对于大量的兴趣点时就会很耗时间和精力。相对的是自动构建法，其能够快速自动地对兴趣集合 $\mathcal{I}$ 构建兴趣分类树，但是精确度是一个问题，下面来详细讨论一下基于公式 4.1计算出来的兴趣相似度为何不能体现出兴趣的层次结构。

首先回顾一下自动构建兴趣分类树的方法，参照图 4.1中的实例进行详解其每一步的过程：

1. 对于初始化状态的兴趣点集合，它们之间可以通过公式 4.1计算相似度。如图中（A）所示，初始状态一共有 15 个不同的兴趣点，编号从 1 至 15。

2. 因为根节点并不表示任何兴趣，所以虚构一个根节点 0，所以第一步是构建树中的第二层。如图中（B）所示，先手动给定要聚成 3 类，那么根据聚类算法（这里假设是 K-means）可以分成三个不同颜色的类，其中节点 1、2、3 分别是三个类的中心，中心的点在兴趣集合中表示该兴趣点涉及的概念和范围更加广泛，因此更适合做该类中其他节点的父节点。最后这三个中心节点与根节点相连，形成树中的第一层。

3. 第二步是构建树中的第三层。如图中（C）所示，这是给定要在三个类中各自再聚成 2 类，经过聚类算法实施后，执行和第一步一样的操作。

4. 第三步是构建树中的第四层。如图中（D）所示，这次依然给定要聚成 2 类，这时因为各类中的元素已经小于或等于 2 个，所以在聚类算法结束的时候正好把所有节点都可以连起来。

5. 第四步对图重画，得到如图（E）中的一个兴趣分类树。由于这个树的结构需要变化的情况十分少，即插入新的兴趣点或删除旧的兴趣点的情况很少发生（至少相对于用户自己的兴趣变化来说），因此称之为静态兴趣分类树。

从上面的过程可以看出，兴趣分类树的构建基础依赖于兴趣点的余弦相似度，树中每一层节点的确定完全依赖于聚类过程中的中心节点，而在树中父节点与孩子节点的实际意义是位于父节点的兴趣点在概念和内容上比处于孩子节点的兴趣点更加宽泛。换句话说，在这个构建的过程中，“宽泛概念”的表现是树中的层次结构，其量化的方法是对“中心节点”的求解，而中心节点的计算又依赖于余弦相似度，因此这里的假设前提是由普通的余弦相似度得到的节点间距离能够反映出层次结构。

但是经过仔细思考之后发现，上述假设存在一定的瑕疵。举个反例来说，如图 4.2所示的是一种人工构建的可能的静态兴趣分类树。在树中的最底层分别有

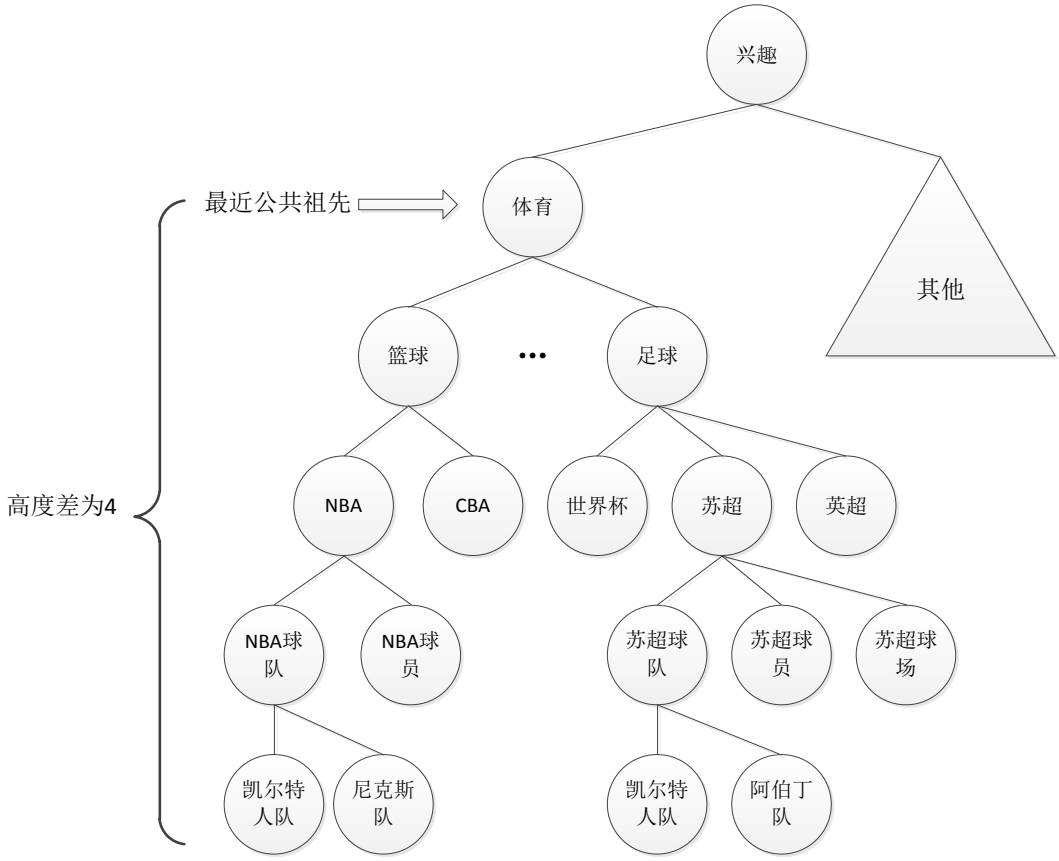


图 4.2: 人工构建的一个静态兴趣分类树的实例

两个兴趣点都叫做“凯尔特人队”，但是它们一个父节点是“NBA 球队”，另一个父节点是“苏超球队”，而且它们的公共父节点是“体育”，距离它们的高度分别都是 4，也就是说这两个兴趣点其实差别比较大，于是，按照上述理论得出之间的余弦相似度应该比较小才对。但是事实上，这两个兴趣点的相似度会十分地接近。如表格 4.1所示，两个“凯尔特人队”的兴趣点在主题词上的相似性很大，比如“得分”、“防守”、“冠军”等词的权重都非常高。如果仅仅考虑用余弦相似度来计算两者的层次关系时，两者在层次结构上会非常地接近，因此与人工判别的实际情况矛盾。

从而可以看出，仅仅使用余弦相似度来判断兴趣点所在树中的层是存在不足的。所以在条件允许的情况下，使用人工构建的办法来建立一棵全局的静态兴趣分类树是一个更好的办法，主要是保证了层级之间兴趣点的准确性。

4.2.2 基于静态兴趣分类树修正的兴趣点匹配

假设根据一定的办法人工或者半自动地构建出一个全局的静态兴趣分类树后，接下来可以利用这棵树对兴趣点匹配的机制进行修正。在上一节中，已经讨论过直接通过余弦相似度计算出来的兴趣点相似度不能很好反映层级的关系，

表 4.1: “凯尔特人队”兴趣点的向量实例

“凯尔特人队（NBA）”	权值	“凯尔特人队（苏超）”	权重
得分	0.131	联赛	0.201
比赛	0.112	财务	0.098
进攻	0.084	冠军	0.082
防守	0.048	得分	0.065
季后赛	0.030	抽签	0.021
首轮	0.019	防守	0.011
击败	0.014	击败	0.007
篮球	0.009	赞助商	0.007
伤病	0.003	点球	0.005
冠军	0.001	赛事	0.002

因此，可以利用已经存在的静态兴趣分类树对兴趣点的相似度在层级关系上进行修正，从而是其更加符合实际情况。

给定两个兴趣点 $\vec{in}_i$ 和 $\vec{in}_j$，可以在静态兴趣分类树中找到它们所在的层数和位置，记 h_i 为兴趣点 $\vec{in}_i$ 在树中的高度，H 为树的高度。通过计算还能够得到两个节点的最近公共祖先（LCA）节点 $\vec{in}_p = lca(\vec{in}_i, \vec{in}_j)$。于是，定义修正过后的两个兴趣点相似度 $d'_{i,j}$ 如下：

$$d'_{i,j} = \varepsilon^{\Delta h-1} \cdot cos(\vec{in}_i, \vec{in}_j), \ \ (0 < \varepsilon < 1) \tag{4.2}$$

其中，

$$\Delta h = \frac{1}{H} \cdot max\{h_p - h_i, h_p - h_j\}$$

这里，$\varepsilon^{\Delta h-1}$ 看做是一个松弛因子，当兴趣点距离最近公共祖先节点越远时，松弛因子越小，从而相当于对原本的余弦相似度进行缩小，使其更加不相似；反之，当兴趣点距离最近公共祖先节点越近，松弛因子越大，从而余弦相似度表达得越准确。同时，当两个兴趣点具有公共父节点时，松弛因子等于 1，即保持原来的余弦相似度。

4.3 动态兴趣伸展树的表示方法

静态兴趣分类树的作用是给全局的用户兴趣建立一个能够表示兴趣层次关系的结构，以此来更好地计算具有层次关系的用户兴趣相似度。在这一节中将考虑用户兴趣的另一个特征：动态性。参考第三章中针对文本流的主题识别问题，用户往往会在 $[t_1, t_2]$ 时间段内产生某一个兴趣点，然后在 $[t_2, t_3]$ 时间段内产生另

一个兴趣点，现状假设已经对文本流分析出一个长度为 L 的主题序列，也即对应于长度为 L 的用户兴趣序列，记作 $seq = (\vec{in}_{k_1}, ..., \vec{in}_{k_L}), (k_i \in \{1...K\})$，所以，问题就转换为如何能够动态地存储和表示这种兴趣序列。

4.3.1 动态兴趣伸展树的构建

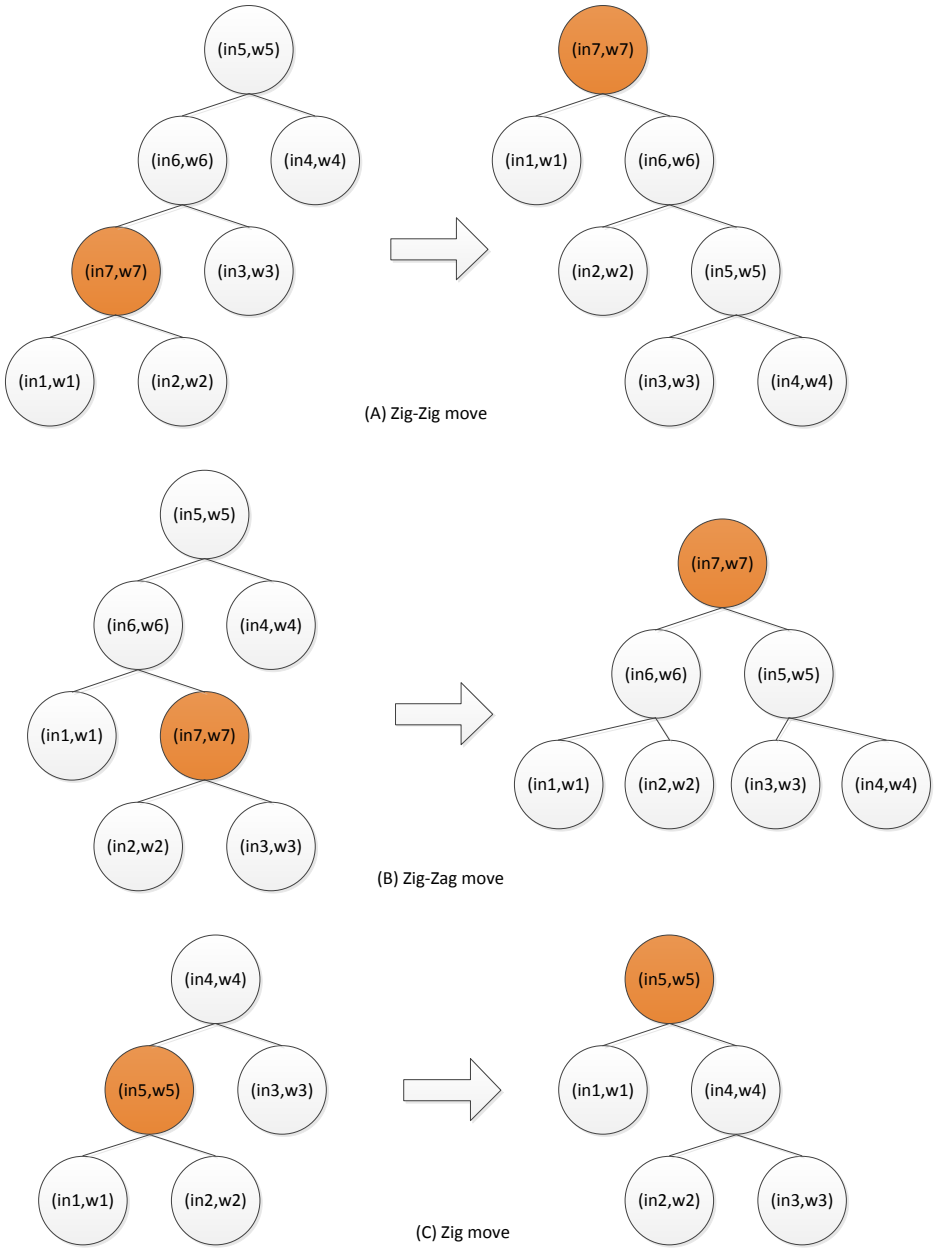


图 4.3: 动态兴趣伸展树在不同情况下的节点查询实例

在伸展树中，最近访问的节点总是会通过旋转进行上浮，而不经常访问的节点则会下沉。伸展树是一种平衡的二叉搜索树，在结构上，它是相同于普通二叉搜索树，唯一不同的在于算法寻找，插入和删除条目。所有伸展树操作为 $O(logn)$ 的平均时间，其中 n 为在树中的条目的数量。任何单一的操作在最坏的

情况下达到 $\Theta(n)$ 时间。但任何对伸展树的连续操作，且树最初是空的并且从来没有超过 n 个项目的条件下，最坏情况下需要 $O(Klogn)$ 的时间。伸展树的目的是为了快速访问到那些最近刚被访问的节点，所以他们真正擅长的应用场景是大多数的查找操作的目标都是小条目。因此，这个数据结构非常适合描述用户兴趣随时间的变化。

在动态兴趣伸展树中，节点 $node_i$ 表示第 i 个兴趣点 $\vec{in}_i$，节点的键表示用户对兴趣 $\vec{in}_i$ 的喜好程度 $key(node_i) = w_i \ \ (w_i \in [0,1])$ 该值用于判断节点之间的大小关系，节点的值表示兴趣 $\vec{in}_i$ 的兴趣分布 $val(node_i) = (v_1, ..., v_V)$，对应文本中的一个主题分布。当用户对某一类主题的文本发生兴趣时，等价地认为在动态兴趣伸展树中查找了对应节点。在查找过程中，树中被查找的节点会根据一系列的旋转直至最后作为根节点，如图 4.3所示为被查询节点旋转的情况。

相比较而言，在普通的二叉搜索树中执行的单个旋转，一些节点在树中移动高，一些较低。在左旋转时，父节点向下移动，其右子节点上移一个层。在双旋转时，是由两个单转组成，并且一个节点向上移动两个层，而所有其他的向上或向下移动通过至多一个层。从刚刚访问的目标节点开始向上移动直至根节点，可以在每一步做单一旋转，从而把该节点一路升至根。这种方法将实现使目标节点到根的目标，但是，事实证明，分摊在很多访问树的性能可能不太好。相反，伸展的关键思想是在每一步中将目标节点向上移动两层。考虑从根下降到刚访问节点的路径，每一次向左下移动时，称之为 zig，而每次向右下移动，则称之为 zag。向左下移动两步称为 zig-zig，右下两步称为 zag-zag，先左再右称为 zig-zag，先右后左称为 zag-zig。这四种情况在移动两步下来的路径中存在唯一的可能。但是如果路径的长度为奇数，最后将多一个步骤，即 zig 或者 zag 中的一步。

当用户产生新的兴趣时，则需要对动态兴趣伸展树进行插入节点的操作，如图 4.4所示。图（A）中表示用户在某一时刻下动态兴趣伸展树的初始状态，每个节点中 (in, w) 表示 (兴趣编号, 权重)，这里省略了兴趣分布。在图（B）中，现在要插入一个新的兴趣 8，其权重为 4，于是先从根节点开始搜索，找到改新增节点在二叉搜索树中的位置，接下来的任务是把该节点向上移动到根节点处。图（C）中展示了在经过一次 zig-zag 旋转之后，兴趣 8 节点所在的位置。最后，再经过一次 zig 旋转，将兴趣 8 节点上移至了根节点出，最后的动态兴趣伸展树的结构如图（D）所示。

关于在动态兴趣伸展树上的其他操作，比如删除节点，合并两个兴趣伸展树等等，将不再详细说明，具体的方法与上述思想类似。

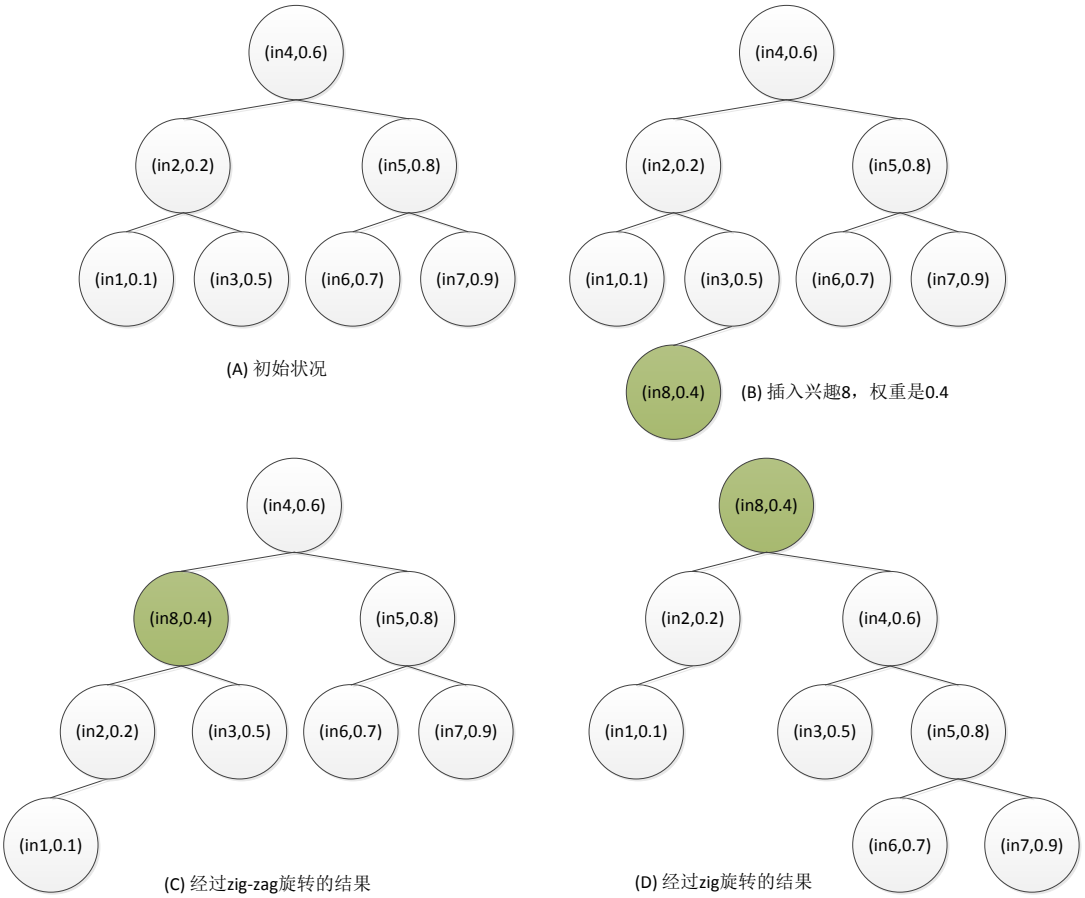


图 4.4: 动态兴趣伸展树添加兴趣点的实例

4.3.2 基于动态兴趣伸展树修正的兴趣点匹配

动态兴趣伸展树其实表示的是用户在某一时段内兴趣的变化状态，而普通的通过兴趣向量计算出的两个用户之间的兴趣相似度则体现不出这一点，因此可以利用动态兴趣伸展树对其进行一定程度上的修正。

对于用户兴趣向量 $D = \{(i_1, w_1), ..., (i_K, w_K)\}$，其中 $i_1, ..., i_K$ 表示 K 个兴趣的编号，$w_1, ..., w_K$ 表示用户对 K 个兴趣的喜好程度，简单来说就是权重，其值的范围是 $[0, 1]$。兴趣向量的求解方式可以是通过对用户涉及的所有历史文档进行主题分析，不同类型的文档主题识别方式已经在上一章详细讨论。可以看出，这种兴趣向量是一种比较全局的描述，因此无法体现出用户近期的兴趣程度。而利用动态兴趣伸展树的结构可以弥补这个兴趣向量在动态性上的不足。于是，定义修正后的兴趣 i_k 的权重 w'_k 为：

$$w'_k = \varepsilon^{h-1} \cdot w_k, \ \ (0 < \varepsilon < 1) \tag{4.3}$$

其中，h 表示兴趣 i_k 对应节点在动态兴趣伸展树中的深度，ε^{h-1} 是一个松弛因子。

当兴趣节点距离根节点越远时，松弛因子越小，说明这个兴趣节点很少被访问，即用户已经很久没有对这个兴趣有喜好了；相反，当兴趣节点距离根节点越近时，松弛因子越大，说明这个兴趣节点最近刚刚被访问过，也即用户最近看过具有对应主题的文章了；当兴趣节点是根节点时，松弛因子是 1，即兴趣权重没有变化。

同时，当原有的 w_k 很大时，该兴趣是一个长期兴趣，反之是一个短期兴趣，长期兴趣即使乘以一个较小的松弛因子时，其权重相对于其他短期兴趣的权重来说还是很大的，所以该方法同时保证了用户的长期兴趣和短期兴趣。

4.4 用户兴趣描述模型

在这一节中，将基于上一章文本主题分析的方法以及这一章用户兴趣分析的方法，提出一套完整的用户兴趣描述与匹配的方法，即用户兴趣描述模型。这个方法主要解决三个问题：如何表示用户兴趣并且如何构建用户兴趣模型、如何计算两个用户之间的公共兴趣、如何匹配用户与资源之间的相似度。

4.4.1 构建准备

在正式提出用户兴趣描述模型之前，先要说明一下准备工作和构建基础。

给定全局语料库 $\mathcal{D}$，根据上一章中长文本分析方法和文本网分析方法，对 $\mathcal{D}$ 进行主题分析，训练得到 K 个全局主题 $\Phi = \{\phi^{(1)}, .., \phi^{(K)}\}$。每个全局主题是一个二元组，即 $\phi^{(i)} = \langle tag^{(i)}, \vec{\varphi}^{(i)} \rangle$，其中 $tag^{(i)}$ 表示 $\phi^{(i)}$ 的名称，$\vec{\varphi}^{(i)}$ 表示 $\phi^{(i)}$ 的主题向量。$\vec{\varphi}^{(i)}$ 中第 i 维的值表示全局词汇表 $\mathcal{V}$ 中第 i 个词汇对全局主题 $\phi^{(i)}$ 的贡献度。假定全局语料库 $\mathcal{D}$ 包含的内容十分丰富，从而保证 Φ 在一定时间段内是一组常量。

同时，根据人工或者半自动的办法，对 K 个全局主题 Φ 构建一棵静态兴趣分类树（Static Interest Category Tree，下面简称 SICT）。类似的，假定全局语料库 $\mathcal{D}$ 十分稳定，从而保证 SICT 在一定时间段内是静态的。

这里认为资源中的“主题”等价于用户中的“兴趣点”，因此兴趣点全集 $\mathcal{I} = \{I^{(1)}, .., I^{(K)}\}$，其中兴趣点 $I^{(i)} = \phi^{(i)} = \langle tag^{(1)}, \vec{\varphi}^{(i)} \rangle$。

4.4.2 用户兴趣的表示方法

这里提出一种全新的用户兴趣表示模型：用户兴趣集，定义如下。

对于用户 u，其大小为 L 的用户兴趣集 $I_u = \{in^{(k_1)}, ..., in^{(k_L)}\}$，$I_u$ 中的每个元素 $in_u^{(k_i)}$ 称为用户兴趣元，它是一个二元组，记为 $in_u^{(k_i)} = \langle I^{(k_i)}, \omega_u^{(k_i)} \rangle$，其中，$I^{(k_i)}$ 是编号为 k_i 的兴趣点，$\omega_u^{(k_i)}$ 表示用户 u 对该兴趣点的偏好程度且满足 $0 < \omega_u^{(k_i)} \leq 1$，越接近 0 则说明用户对该兴趣点越没有喜好，反之亦然。

本节所有的符号如表 4.2所示：

表 4.2: 本节中用到的符号

符号	含义
$\mathcal{D}$	全局语料库
$\mathcal{V}$	全局词汇表
V	全局词汇表的大小
Φ	全局主题集合
K	全局主题集合（兴趣点全集）的大小
$\phi^{(i)}$	编号为 i 的全局主题
$tag^{(i)}$	编号为 i 的全局主题（兴趣点）的名称
$\vec{\varphi}^{(i)}$	编号为 i 的全局主题（兴趣点）的主题向量（V 维）
d	文档
$\vec{\vartheta}_d$	文档 d 的主题成分向量（K 维）
$w_d^{(i)}$	编号为 i 的全局主题在文档 d 中所占的权重
$\mathcal{I}$	兴趣点全集
$I^{(i)}$	编号为 i 的兴趣点
I_u	用户 u 的兴趣集
$in_u^{(i)}$	包含编号为 i 兴趣点的用户兴趣元
$\omega_u^{(i)}$	用户 u 对编号为 i 兴趣点的喜好程度
t	时间戳

4.4.3 用户兴趣集的构建

定义 4.1 (用户兴趣集初始化问题) *给定长度为 M 的文本序列 $\{\langle d_1, t_1\rangle, ..., \langle d_M, t_M\rangle\}$ $(t_1 < ... < t_M)$，求用户 u 在 t_M 时刻的兴趣集 $I_{u|t=t_M}$。*

用户兴趣集的初始化方法如下：

1. 初始化用户兴趣集为空集，即 $I_u = \emptyset$。初始化动态兴趣伸展树（Dynamic Interest Splay Tree，下文简称 DIST）为空树。
2. 取出文本序列中第一个元素 $\langle d_1, t_1\rangle$，即 t_1 时刻的文档 d_1。根据第三章中的方法，对文档 d_1 进行主题分析，从而得到该文档的主题成分向量 $\vec{\vartheta}_d = (w_d^{(1)}, ..., w_d^{(K)})$。从 $\vec{\vartheta}_d$ 取出最大的权重 $w_d^{(k_{max})}$，对应编号为 k_{max} 的全局主题，是该主题作为文档 d 的主要主题。
3. 判断中 I_u 中是否包含编号为 k_{max} 的用户兴趣元，如果存在，则执行第 4 步，否则执行第 5 步。

4. 更新 I_u 中用户兴趣元 $in_u^{k_{max}} = \langle I^{(k_{max})}, \omega_u^{(k_{max})} \rangle$，使得 $\omega_u^{(k_{max})} += w_d^{(k_{max})}$ 并归一化。同时，删除 DIST 中的对应的节点，并重新插入更新后的节点。跳转第 6 步。

5. 构建新节点，使该节点的 key 为 k_{max}，value 是 $w_d^{(k_{max})}$。根据上一节的方法对 DIST 插入该节点。对 I_u 中插入用户兴趣元 $in_u^{k_{max}} = \langle I^{(k_{max})}, w_d^{(k_{max})} \rangle$。

6. 对文档序列中的剩下所有文档依次重复上述 2-5 步，直至完成。

当上述步骤完成后，会生成两个结构：用户兴趣集 I_u 和 DIST。下文将说明如何使用这两个结构。

4.4.4 用户公共兴趣的计算方法

定义 4.2 (用户公共兴趣问题) *给定用户 x 的兴趣集 I_x 和 DIST，以及用户 y 的兴趣集 I_y 和 DIST。求用户 x 和 y 的公共兴趣点集合。*

该问题的求解过程如下：

构建用户兴趣相关矩阵 $\Sigma = \sigma_{i,j}$ $(i = \{1..|I_x|\}, j = \{1..|I_y|\})$，其中矩阵的元素 $\sigma_{i,j}$ 计算如下：

$$\begin{cases} \sigma_{i,j} = z \cdot cos(z_x, z_y) \\ z = \varepsilon^{\Delta h - 1} \\ z_x = \varepsilon^{h_x - 1} \cdot \vec{\varphi}^{(i)} \\ z_y = \varepsilon^{h_y - 1} \cdot \vec{\varphi}^{(j)} \end{cases} \tag{4.4}$$

其中，矩阵的行表示用户 x 兴趣集中的元素，大小为 $|I_x|$，列表示用户 y 兴趣集中的元素，大小为 $|I_y|$。因此，在第 i 行第 i 列中，取出用户 x 兴趣集的第 i 个兴趣元 $in_x^{(k_i)}$，包括全局主题的主题向量 $\vec{\varphi}^{k_i}$、该用户 x 对该兴趣点的喜好程度 $\omega_x^{(k_i)}$ 等。对于用户 y 也是如此。

因此，$\sigma_{i,j}$ 表达的含义是用户 x 的第 k_i 个兴趣和用户 y 的第 k_j 个兴趣间的相似度。考虑到兴趣的层次关联性和动态性，依然引入本章开始提出的松弛因子 ε 作为修正。从上面的公式中可以看出，Δh 是两个兴趣在 SICT 中的深度差，h_x 为第 k_i 个兴趣在用户 x 的 DIST 中的深度，h_y 同理。

当计算完之后，可以取出矩阵中尽可能大的 $\sigma_{i,j}$，其对应着两个用户各自的兴趣点，将这些兴趣点组成的集合既是公共兴趣点集合。

4.4.5 用户与资源之间匹配方法

定义 4.3 (用户与资源间匹配问题) *给定文本 d 和用户 u 的兴趣集 I_u，求 d 与 I_u 之间的相似度。*

首先通过主题分析求出文本 d 的 K 维主题成分向量 $\vec{\vartheta}_d$，构建用户的兴趣向量 $\vec{a}$，兴趣向量中的每一维 a_i $(i \in [1, K])$ 计算方式为：

$$a_i = \begin{cases} \varepsilon^{h-1} \cdot \omega_u^{(i)} & \text{if } in_u^{(i)} \in I_u \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

从上式看出，依然加入了松弛因子，h 为该兴趣节点在 DIST 中的深度。

最后，将 $\vec{\vartheta}_d$ 与 $\vec{a}$ 求余弦相似度即可。

4.5 小结

在这一章中，提出了两种兴趣树的结构用来充分体现用户兴趣的特征。静态兴趣分类树反映的是用户兴趣之间的层次关联性。在计算两个兴趣点的相似度时，如果仅仅对兴趣对应的主题分布求余弦相似度的话并不能体现出兴趣点之间的层次性，因此需要利用静态兴趣分类树对这个相似度进行修正。类似的，动态兴趣伸展树反映的是用户兴趣随时间变化的动态性。在计算用户的兴趣向量时，由于每个兴趣点对应的权值反映的是全局用户兴趣的特征，因此不能反映最新实时的用户兴趣，所以需要利用动态兴趣伸展树对这个权重进行修正。最后，基于文本主题分析方法和用户兴趣分析方法，提出了完整的一套用户兴趣表示与匹配的方式。

第 5 章 基于 P2P 网络的信息自主流动机制的研究

本章将基于上一章提出的用户兴趣模型给出建立用户兴趣覆盖网络的基本思路和方法。首先，从宏观上给出整个网络的组织架构和拓扑结构，由于是非中心化的 P2P，因此网络中的每个节点既要充当服务端也可以作为客户端，如何能够快速发现兴趣相同的节点在信息传播中是尤为重要的。然后，从微观上来看，每个节点存储的信息是决定整个网络流动是否顺畅的基础，因此本章将充分利用用户兴趣集的特性来构建节点存储结构，并阐述如何利用该结构快速找到潜在的目标节点。

5.1 用户兴趣覆盖网络的构建

在第二章中提到了用户之间遵循一种关注 -被关注的关系，这在社交网络中是十分常见的，因此这里直接将这种关系引入，默认用户节点直接会实现存储一些时常关注的邻居节点。整个网络的架构如图 5.1所示，图（A）是节点初步连接后的结果。可以看到，在用户关注关系层上，用户 u_1 和用户 u_3 并没有直接的关注或被关注的关系，u_3 必须先后间接通过 u_4 和 u_3 才能连接到。但是在上面的用户兴趣关系层中，用户 u_1 和 u_3 之间直接建立了关系，依据是他们拥有共同的兴趣（图中绿色边表示）。此时，用户 u_1 充当的是服务器，他将自己的资源 1 和资源 2 分别传播给了另外三个用户，并且转播的依据是存在共同兴趣。当经过一系列运作以后，用户兴趣关系层中的拓扑结构会发生变化，如图（B）所示。可以看到，用户 u_4 和 u_1、u_2 之间发现了新的兴趣（图中蓝色边表示）。此时，用户 u_4 作为信息发布者将资源 3 传播给这两个节点，因此 u_1 此时充当了资源接受者，但同时 u_1 也在继续发布资源 1，它充当了两个角色。这就是基于 P2P 网络的一个实例，在构建这样一个覆盖网络归结起来要解决这样两个问题：

- 节点的数据存储结构：因为整个网络是一个非结构化的拓扑，也就是说没有中心服务器来直接获得其他节点的信息。所以，每个节点必须自己存储其他部分节点的信息。
- 节点间兴趣导向边的建立：节点与节点之间的关系在用户兴趣关系层上表现为两个用户具有共同的兴趣。以上一章的用户兴趣模型来看，也就等价于两者的兴趣集中存在公共兴趣点集合。

5.1.1 节点的数据存储结构

在基于 P2P 的兴趣覆盖网络中，几乎所有节点都会既要充当信息发布者，又要充当信息接受者，既同时作为服务器和客户端。考虑到这种网络的特性，需要

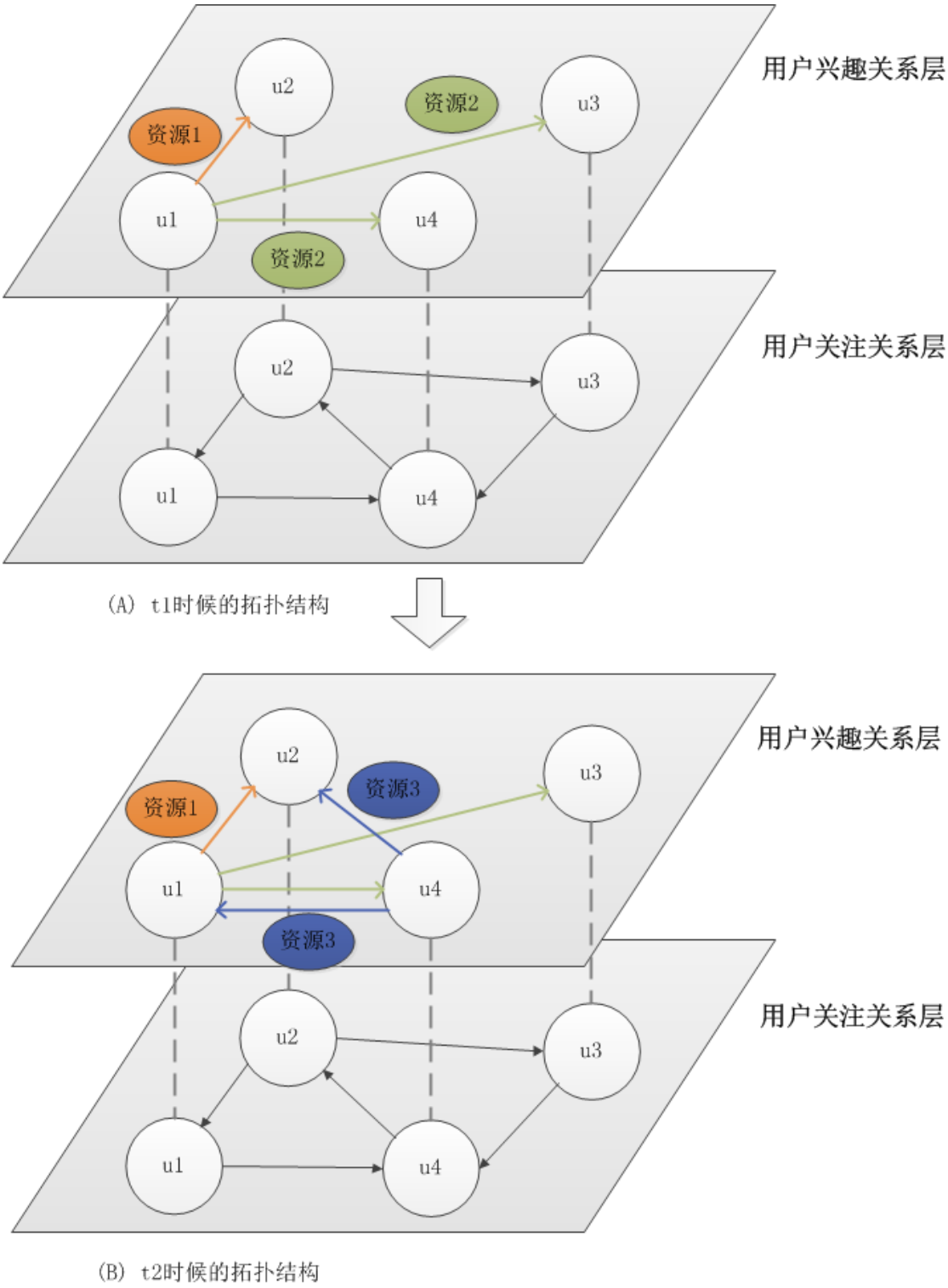


图 5.1: 基于 P2P 的兴趣覆盖网络示意图

给节点加入以下一些松耦合的规则才能形成。虽然最终得到的拓扑结构具有一定属性，但数据项的设计不能像结构化拓扑中的那样基于任何拓扑相关的知识。在之后的搜索过程中，先要找到一个数据项，查询该节点其邻居，最典型的查询方法是泛洪。该查询协议是对一定范围内的所有的邻居进行搜索，因此需要在物理上存储周围这些节点的信息。这样的设计是非常有弹性的，特别是对于那些进

入和离开该系统的节点。

因此，每个节点的需要保存的数据结构如图 5.2所示:

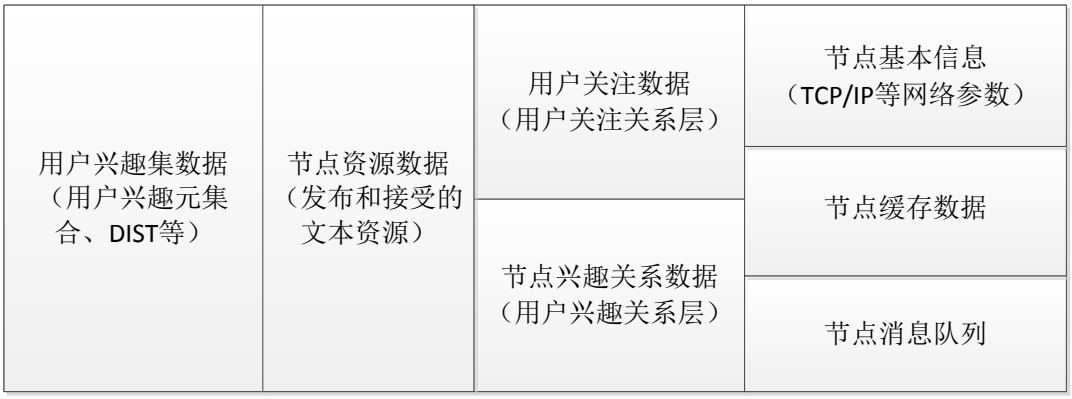


图 5.2: 兴趣覆盖网络中节点的数据结构

1. 节点基本信息：节点在 TCP/IP 上的基本信息，包括 IP 地址、节点 ID 等，这些信息用来方便其他节点进行定位。换句话说，当节点之间要直接通信时，可以根据这些数据直接连接，这里的通信仍然遵循物理网络中的协议。

2. 用户关注数据：这里存储的是用户关注关系层中的数据，即用户关注的其他用户节点列表，列表中的信息包括上述基本信息，主要是在网络初始化的时候，节点用来寻找其他潜在的兴趣相同的节点。

3. 用户兴趣集数据：这里的用户兴趣集和上文中提出的概念是一致的，具体参见第四章相关的讨论。

4. 节点兴趣关系数据：这里存储的是用户兴趣关系层中的数据，包括具有共同兴趣的用户节点信息，共同兴趣集的数据等。

5. 节点资源数据：当用户生成新的文本资源时，资源副本都会在本地进行保存。同理，用户接受的文本资源也会进行存储。此外，该部分数据还包括了用户对收到资源的反馈、关联的发布者信息、其他节点收到资源后反馈的信息等等。

6. 节点消息队列：缓存中需要发送和接受的消息，包括节点的基本信息，兴趣集等。

7. 节点缓存数据：在网络通信中临时保存的数据，这些数据主要用于方便其他节点更加快速地找到资源，以及方便自己找到其他兴趣类似的节点。

5.1.2 兴趣覆盖网络的初始化过程

这部分的工作是为了在用户兴趣关系层中建立节点之间的边，这里的边称为兴趣导向边。换句话，一旦两个节点之间有边相连时，说明两个用户之间存在较为明显的共同兴趣，关于如何根据用户兴趣集求出共同兴趣点集合的算法参见第四章。如图 5.3所示，下面对兴趣覆盖网络的初化化过程进行说明：

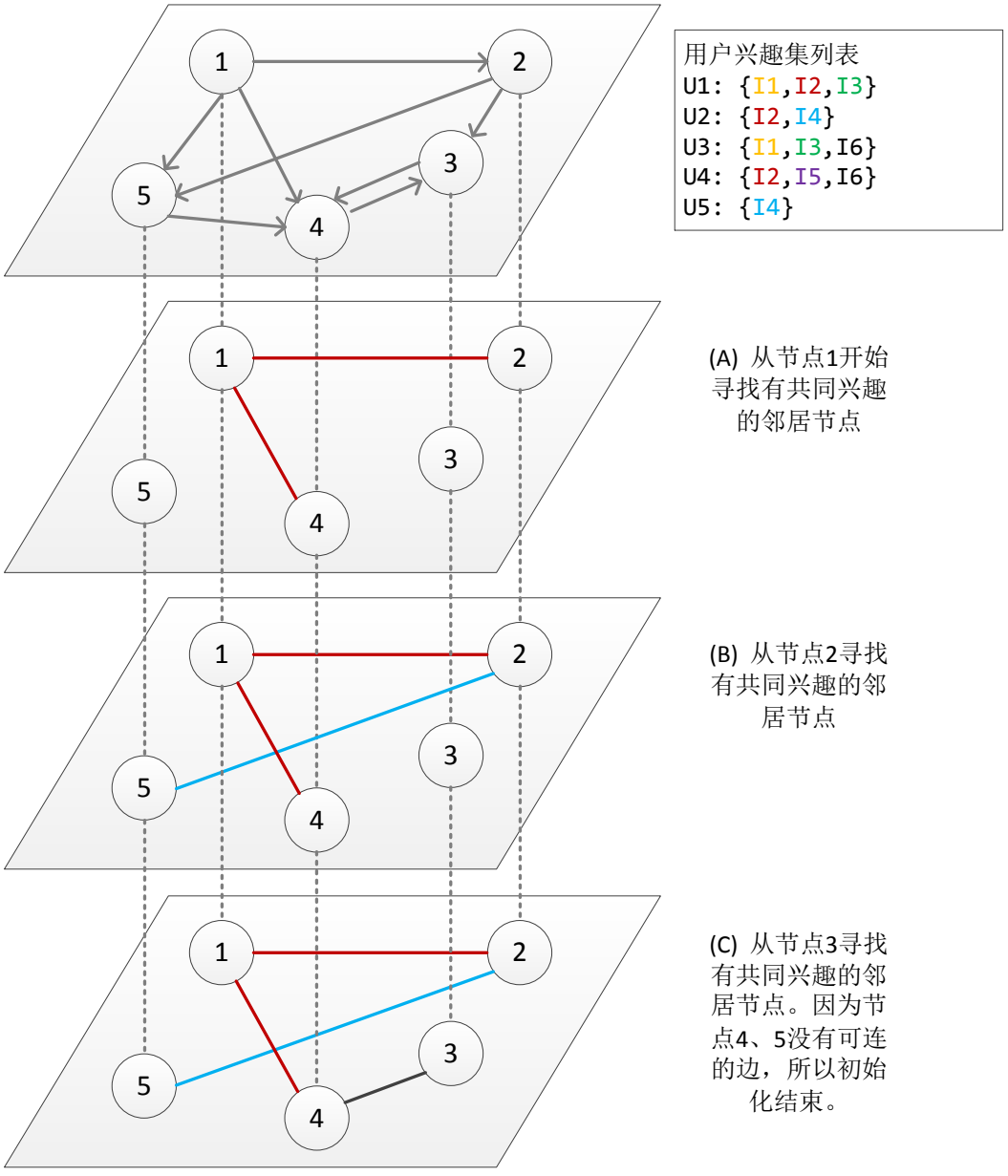


图 5.3: 兴趣覆盖网络的初始化实例

在整个初始化过程中，我们使用 Iterative Deepening 的方式进行路由，为了简化问题，这里设置 TTL=1，即在网络中只执行一跳。

1. 如图（A）所示，先从节点 1 开始寻找具有相同兴趣的邻居节点。从用户关

注关系层中可以发现，节点 1 关注了节点 2、4、5，因此依次访问这三个节点进行判断。这里共同兴趣的判断采用了简化的模式，即认为两个兴趣集的交集元素是共同元素（实际判断方法还要引入 DIST 和 SICT 进行相似度的计算）。经过计算后发现，节点 1 和节点 2 有共同兴趣 I_2，与节点 4 也有共同兴趣 I_2，所以在用户兴趣关系层中这这三个节点间分别建立兴趣导向边（图中红色的边）。

2. 如图（B）所示，节点 2 继续寻找具有相同兴趣的邻居节点。从用户关注关系层中发现，节点 2 关注了节点 3、5，因此依次访问这两个节点来判断是否有共同兴趣。经过计算后发现，只有节点 2 和节点 5 之间存在共同兴趣 I_4，因此建立一条兴趣导向边（图中蓝色的边）。

3. 如图（C）所示，节点 3 继续寻找具有相同兴趣的邻居节点。从用户关注关系层中发现，节点 3 关注了节点 4，经过计算后判断出它们之间有共同兴趣 I_6，从而建立一条兴趣导向边（图中黑色的边）。对节点 4 和节点 5 执行相同的操作，发现它们与其他的节点之间都没有可以新增的兴趣导向边，所以初始化结束。

从上述例子中可以总结出以下几点。第一，虽然在用户关注关系层中边的数量比较多，但是在最后的用户兴趣关系层中边的数量却要少很多。导致这个现象的原因可能是因为这些节点间共同的兴趣集十分少，用户之间之所以关注是因为社交网络中好友的关系。第二，在用户兴趣关系层中，每个节点拥有的边的数量也减少很多，这样的好处是用户只会接受到兴趣相似的用户发来的资源，从而可以过滤掉没用的信息。第三，这里设置的 TTL 为 1，因此在用户兴趣关系层中的边数比较少，但是如果将 TTL 设置更大一下，本来没有相互关注的节点也会直接建立兴趣上的连接。

5.2 兴趣覆盖网络拓扑结构的更新

根据第四章中关于用户兴趣讨论的内容可知，动态性是用户兴趣的主要特征之一，而且在实际应用中十分重要。因此当用户兴趣发生变化时，用户兴趣关系层中的拓扑结构也会发生相应的变化，如图 5.4所示为用户产生新兴趣时结构变化的过程。

1. 如图（A）所示，由于节点 1 新增了一个兴趣 I_4，所以从节点 1 开始路由查找潜在的享有相同兴趣的节点。这里设置 TTL=2。在第一跳中，依然根据用户关注关系层中的结构，节点 1 先后与节点 2、4、5 进行交互，发现与节点 2 和 5 都新增了共同兴趣 I_4，所以在用户兴趣关系层中，节点 1 分别与节点 2、5 建立连接（图中蓝色边）。

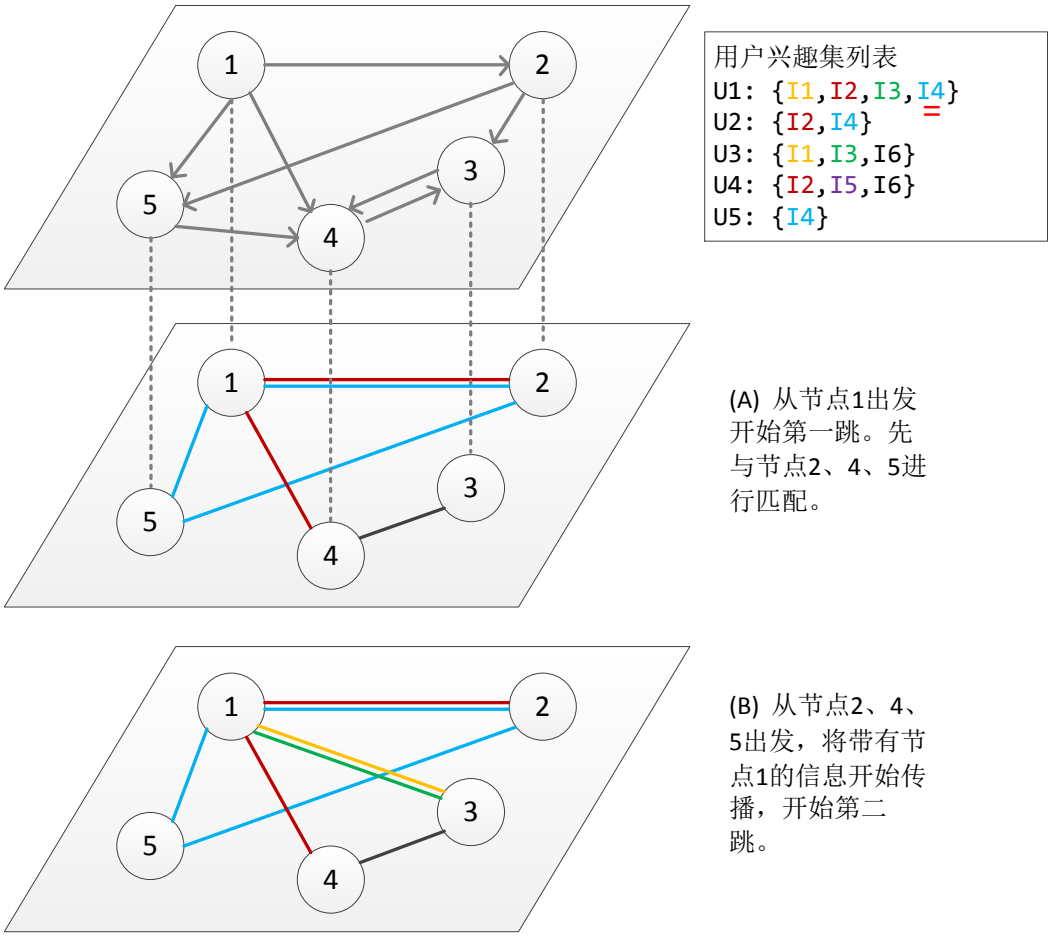


图 5.4: 当用户兴趣变化时，兴趣覆盖网络拓扑调整的实例

2. 如图（B）所示，节点 1 将自己的信息传播给节点 2、4、5 之后，依次从这三个节点开始第二跳的查找。由于在用户关注关系层中，节点 2 关注了节点 3，因此节点 2 将节点 1 的信息传播给了节点 3，判断之后发现节点 1 与 3 之间存在共同兴趣 I_1 和 I_3，于是在用户兴趣关系层中建立两条兴趣导向边（图中绿色和黄色的边）。在依次完成其他二跳之后，发现拓扑稳定，完成更新。

5.3 兴趣覆盖网络的资源传播

在兴趣覆盖网络上传播资源时，在很大程度上依赖于用户兴趣关系层中的拓扑结构。当源节点与目标节点存在一条兴趣导向边的时候，两个节点之间可以直接建立连接进行通信。当源节点的用户生成一篇新的文章时，如果该文章的主题属于兴趣导向边中的共同兴趣集，那么该文章就会自动流向目标节点的用户，这是一个基于兴趣的资源推荐情况。

当目标节点收到该资源时，该节点上的用户会人工判断是否要详细查看该资

源，相当于源节点可以收到一个对方用户的反馈信息。如果目标节点的反馈是正评价时，即对方用户喜欢这个资源，那么源节点上的数据结构保持不变。但是，当反馈是负评价时，源节点会进行如下操作：在数据结构中找到与目标节点相关的节点兴趣关系数据，对与之相连的兴趣导向边上，将公共兴趣的相似度降低一点，两者的相似度低于一个给定的阈值时，该边会自动取消。

5.4 小结

在这一章中，详细讨论了基于 P2P 网络的信息自主流动机制，并且这一部分的工作在很大程度上依赖于上一章的用户兴趣模型。首先，在用户兴趣覆盖网络的构建上，分别对节点上要存储的数据结构进行了分析，然后基于该数据结构，提出了兴趣覆盖网络的初始化过程，以及当节点上兴趣变化后拓扑更新的方法。最后简单分析了在兴趣覆盖网络上资源传播的机制，以及基于用户反馈的拓扑调整。

第 6 章　实验分析与结果分析

// todo

6.1　文本主题分析的实验与评价

// todo

6.2　基于兴趣树的用户兴趣描述模型的实验与评价

// todo

6.3　兴趣覆盖网络的实验与评价

// todo

6.4　小结

// todo

第 7 章　原型系统设计与实现

本章将根据前文对基于 P2P 网络的信息自主流动机制的讨论，设计并实现一套较为完善的原型系统。这里系统是指运行在每个节点上既能充当服务器发送信息、也能充当客户端接受信息，我们称之为节点系统。节点系统中作为服务器角色的模块成为节点服务端程序，另外作为客户端角色的模块成为节点客户端程序。本章将分别从节点系统需求分析、架构、数据库、功能模块等方面入手，详细讨论各个实现环节的细节，最后将简单展示节点客户端程序的一些功能。

7.1　节点系统的总体需求设计

在本节中，将分析节点系统中各大服务模块的功能，包括节点客户端程序的资源管理功能，以及节点服务器程序中包含的四大功能模块。

7.1.1　节点客户端程序的需求设计

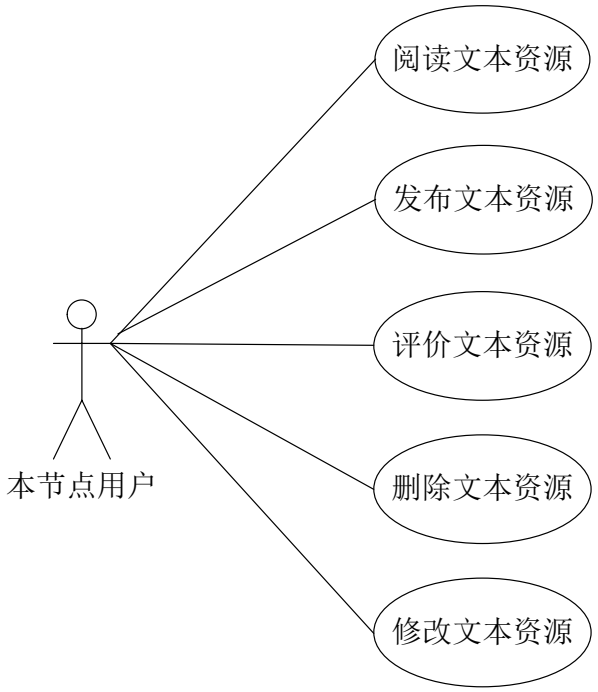


图 7.1: 节点客户端程序的用例图

该模块的功能比较简单，主要针对的角色是本节点用户，具体的操作包括对文本资源的阅读、删除和修改，以及发布新的文本资源，或对文本资源进行评价和反馈。如图**??**所示，展示了节点客户端程序的用例图。

7.1.2 服务端资源管理的需求设计

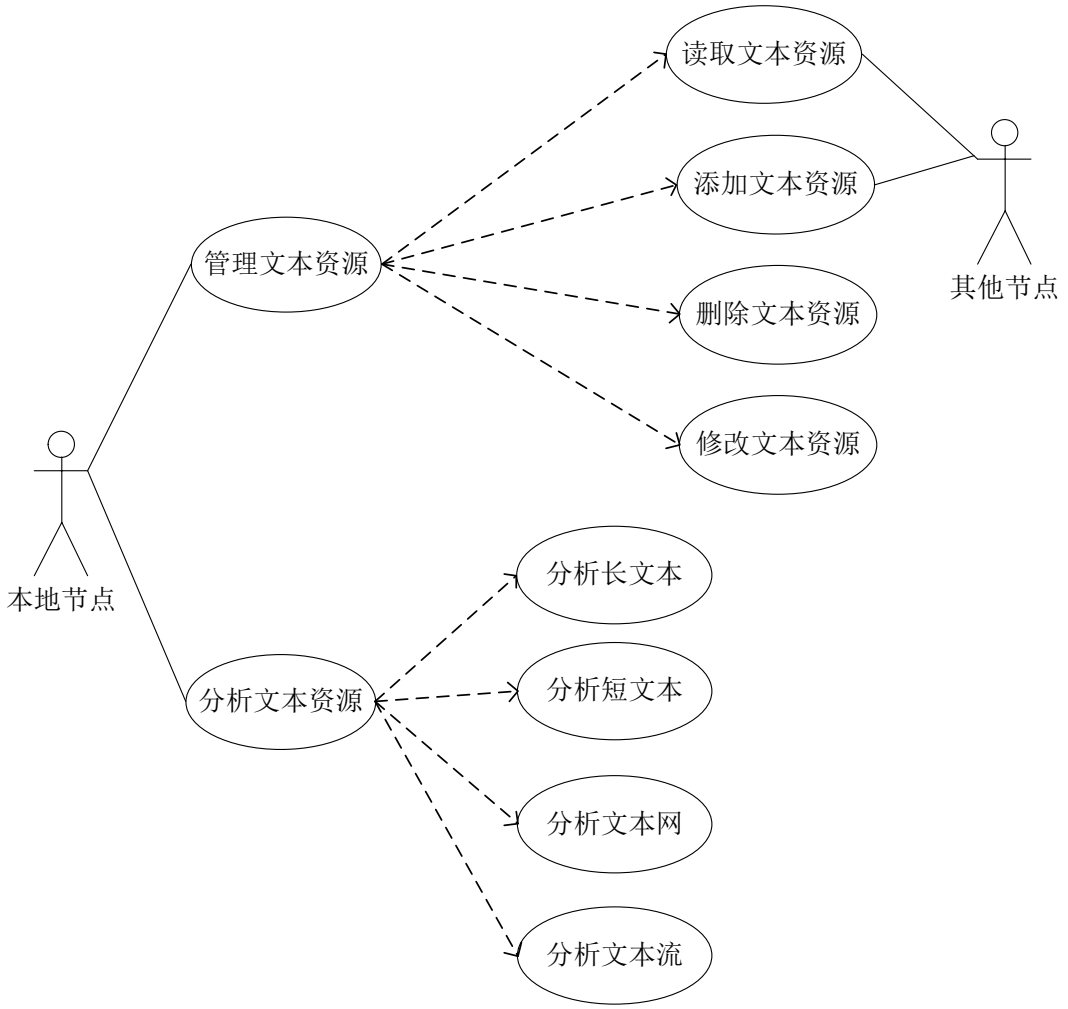


图 7.2: 服务端资源管理的用例图

该模块的功能主要是对文本资源的进行管理和分析，本地节点拥有所有的功能权限。在管理文本资源模块中，可以对文本资源进行读取、添加、删除和修改的操作。而其他节点只能对文本资源进行读取和添加。在分析文本资源模块中，本地节点可以对四种类型的文本进行分析，包括长文本资源、短文本资源、文本网资源和文本流资源。这四种文本的分析主要是从这些类型的文本中识别出主题，从而在以后的用户兴趣分析中起到作用。如图**??**所示，展示了节点服务端程序中资源管理模块的用例图。

7.1.3 服务端节点管理的需求设计

该模块的功能主要是对本地节点和其他节点进行管理和搜索。在管理本地节点模块中，本地节点可以对本地节点的信息进行读取、添加、删除和修改，而其

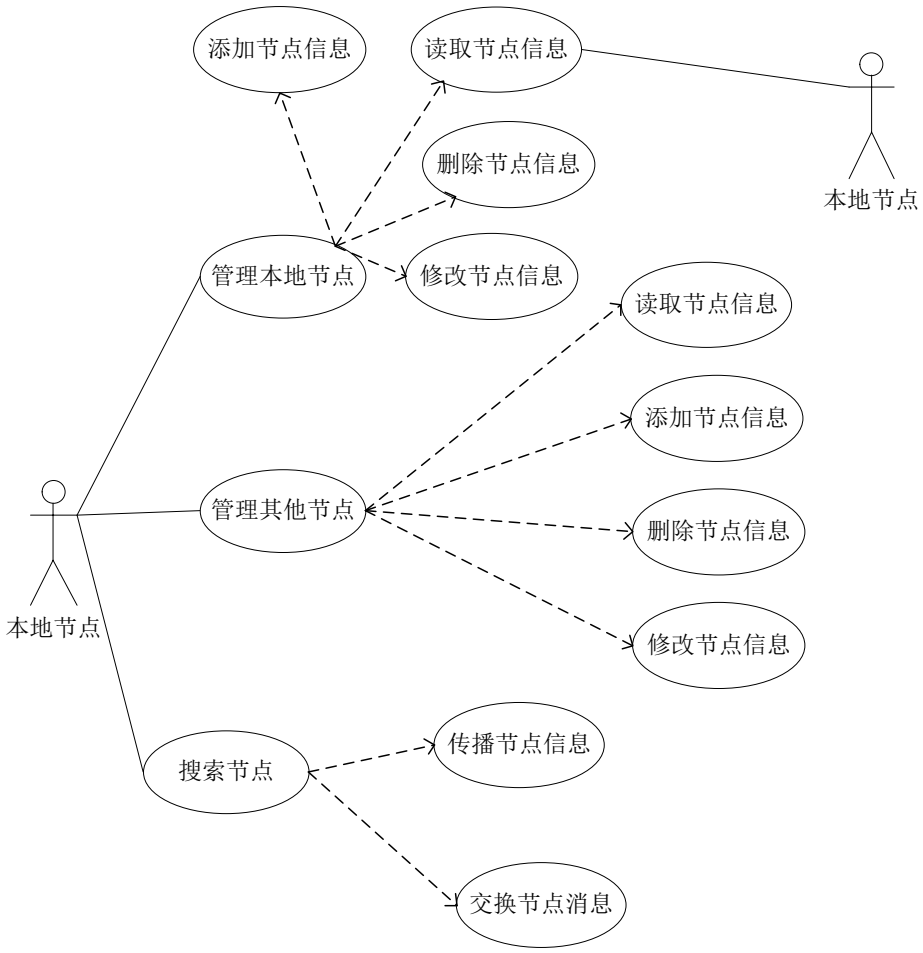


图 7.3: 服务端节点管理的用例图

他节点则可以进行读取的操作。在管理其他节点模块里，只有本地节点可以对其他节点的信息进行读取、添加、删除和修改。在搜索节点模块里，本地节点可以与其他节点进行传播和交易信息。如图**??**所示，展示了节点服务端程序中节点管理模块的用例图。

7.1.4 服务端兴趣集管理的需求设计

该模块的功能主要对本地和其他的兴趣进行管理，以及对兴趣集进行分析。在管理本地兴趣集的模块里，本地节点可以对本地兴趣集进行添加、删除、更新和读取，而其他节点可以读取本地兴趣集。在管理其他兴趣集的模块中，本地节点可以对其他兴趣集进行读取和更新，其他节点则可以直接进行更新。在分析兴趣集的模块里，本地节点可以对本地存储的动态兴趣伸展树进行读取、建立、更新和删除的操作，同时也能计算给定两个兴趣集的公共兴趣。关于兴趣集的运算详情，请参见第四章的讨论。如图**??**所示，展示了节点服务端程序中兴趣集管理模块的用例图。

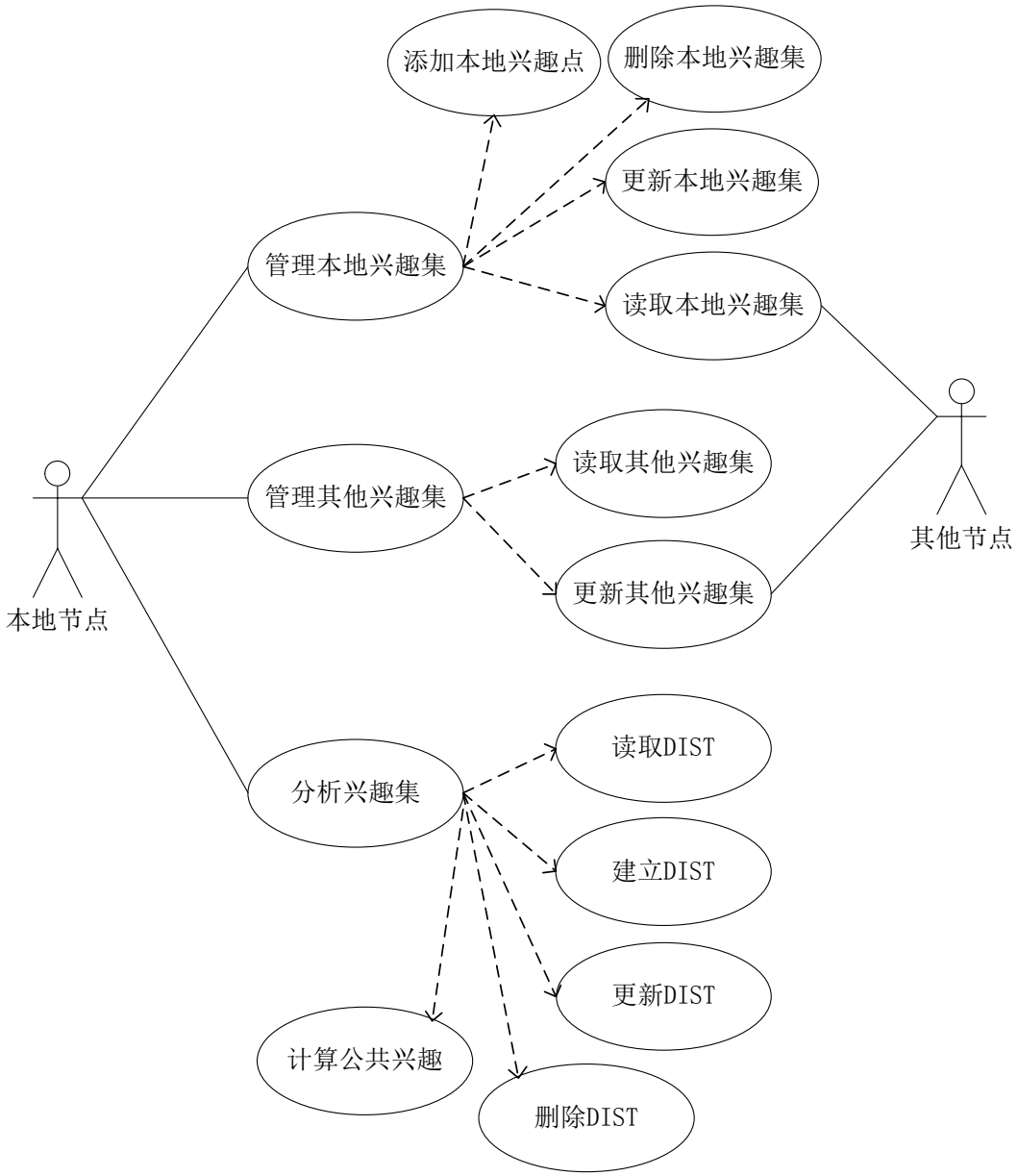


图 7.4: 服务端兴趣集管理的用例图

7.1.5 服务端缓存服务的需求设计

该模块主要提供缓存管理的功能，其中，本地节点可以对缓存中的数据进行读取、更新、删除和添加的操作，而其他节点则只能读取缓存的数据。如图**??**所示，展示了节点服务端程序中缓存管理模块的用例图。

要注意的是，这里的缓存是为了让其他节点执行更加快速地访问，从而可以有效地减少网络带宽的占用，减少用户等待的延迟以及减少其实服务器的负载。但是缓存的大小不是无限的，特别是在用户节点上运行的机器，因此缓存的内容必须要有取舍。这里默认使用的是基于 LRU（Least Recently Used）的缓存替换

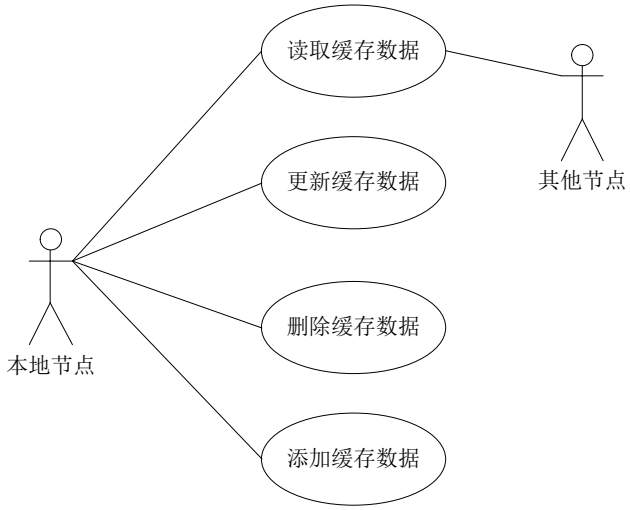


图 7.5: 服务端缓存管理的用例图

策略，即节点会每次替换距离上一次被访问时间最长的内容。

7.2 主要开发技术

节点系统主要用 Java 编程语言实现，辅以 Python 作为数据分析之用，模块间与系统间的消息传递格式为 JSON，具体技术如表 7.1。鉴于本系统只是一个雏形，因此使用了大量轻量级框架进行敏捷开发，在并发性等性能方面的问题暂时先不纳入考虑。

表 7.1: 原型系统开发主要技术列表

操作系统：	OS X 10.10.3
系统后台开发语言：	JDK 1.8
系统后台开发框架：	Spring、JFinal
后台数据库：	sqlite 3
系统前端开发语言：	Html、CSS、Javascript
系统前端开发框架：	Bootstrap、JQuery
数据引擎开发语言：	Python 2.7、R
数据引擎计算框架：	numpy、scipy、sklearn、gensim、NLTK

7.3 节点系统的架构设计

首先，节点系统的最大特点是既要充当服务器向外部的节点进行交互信息，也要作为客户端为本地用户提供一个可视化的浏览平台。为了满足这样的需求，节点系统的架构设计如图 7.6所示。

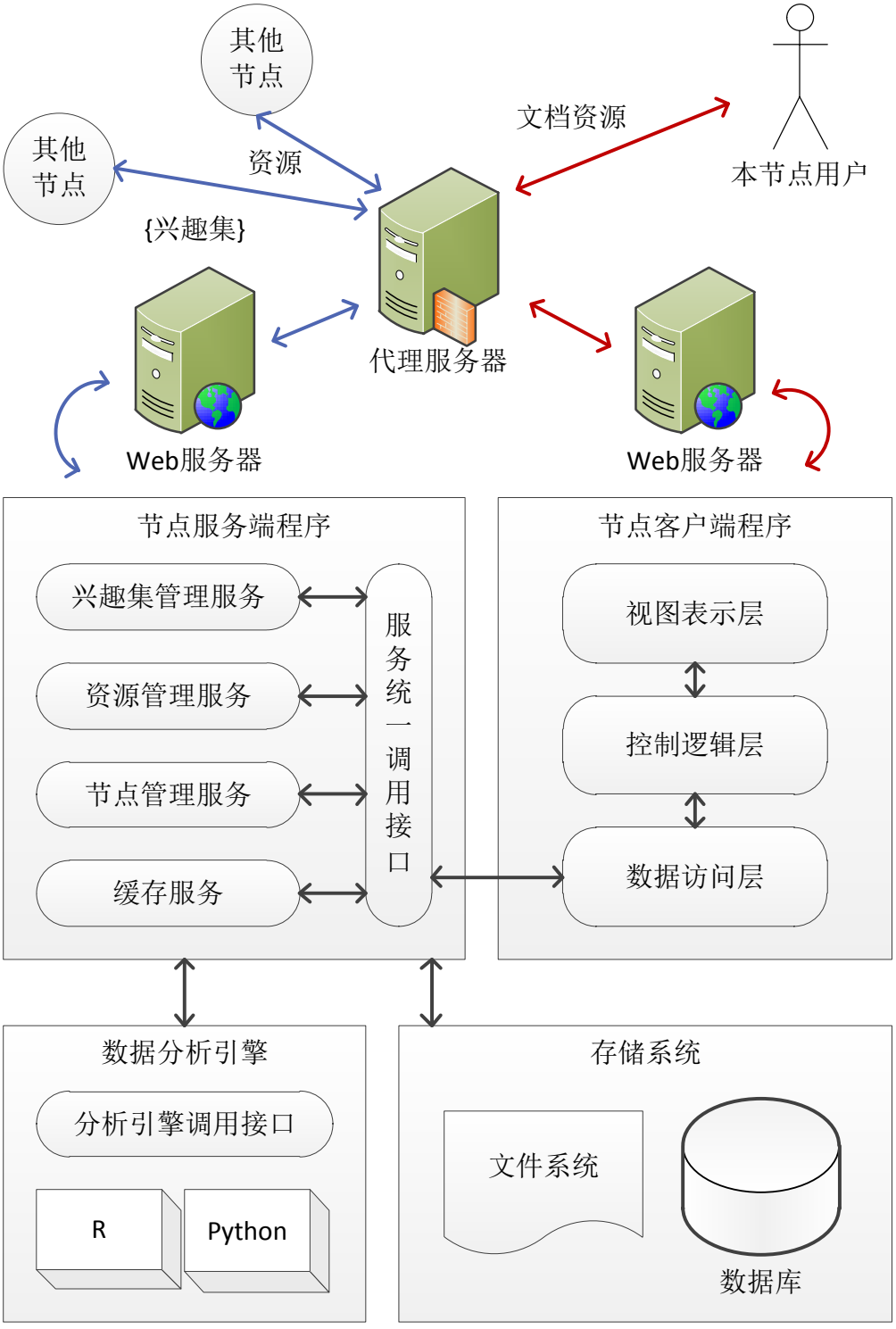


图 7.6: 节点系统的架构示意图

从图中可以看到，节点系统处理的请求用户可以分为两大类：本节点的用户和其他节点的程序。其中，本节点的用户主要是对本地资源进行操作，包括发表新的文章、查看推荐的文章等。而其他节点的程序主要是与本节点交换兴趣信息

和网络信息等。考虑到两种请求的独立性，我们将节点系统分为节点服务端程序和节点客户端程序两大模块，并分别运行于两个独立的 web 服务器中，并由代理服务器统一接受、处理并转发双方的请求。这样做的好处是模块之间保持相对地独立，作用分工明确，耦合性也比较小。

对于节点客户端程序，本质是一个运行在 web 服务器上的网站系统，由于其功能相对简单，主要是对资源数据的读写，因此在架构设计方面采用了如下两点方案：

- MVC 三层架构：在传统的 J2EE 架构中，控制层和业务逻辑层往往是分开的，但是文本为了快速实现原型系统，对节点客户端程序采用了更加简单的 MVC 架构，即将业务逻辑与页面控制合并在控制层实现。由于其功能在逻辑处理上非常简单，所以这样的实现方式并不会给系统带来多大的负担。

- 数据访问限制：节点客户端程序不能直接访问数据库或文件系统，而是通过节点服务端程序提供的统一接口进行操作。换句话说，整个系统对底层存储资源的调用全部集中在节点服务端程序中，这样对读写操作的隔离在设计上保证了数据的安全性，同时避免了因为多进程对数据读写导致的一致性问题。

对于节点服务端程序，其功能和模块相对复杂很多，内部主要分为四大功能模块，并且给外部暴露了一个统一的 API，一方面给外来节点请求做响应，另一方面给内部节点客户端程序调用资源相关操作。在四个功能模块中，资源管理服务模块负责用户文本资源相关操作的处理，其只需对存储系统简单的读写操作。节点管理服务模块负责管理本节点的相关信息，以及在用户关注关系层和用户兴趣关系层中的节点信息，这部分对 I/O 的要求比较高，所以需要结合缓存服务模块一起是用。兴趣集管理服务则管理了用户兴趣模型，以及对文本进行主题分析，从而分析用户兴趣，维护了 DIST、SICT 等数据结构，该模块主要对文件系统有较多的读写操作，而且需要调用后台的数据分析引擎进行模型的训练和使用。缓存服务负责对节点内常用的信息进行缓存，从而增加外部节点请求的效率。

对于分析引擎来说，其目的是为了从用户的行为和文本中挖掘出用户的兴趣，并给兴趣集管理模块提供支持。该分析引擎提供了一个统一调用的接口，用于封装后台异构的数据分析模块，比如 R 和 Python，当然还可以集成更多其他的分析引擎。

对于存储系统来说，这里用到了关系型数据库和文件系统。数据库主要存储结构化的信息，从而方便快速索引，而文件系统则存储文本资源、缓存等。

7.4 节点系统的存储机制

本系统共有三种存储的方式，即数据库、文件系统和内存，下面将具体讨论三种类型的存储介质所保存的数据。

7.4.1 节点系统的数据库

数据库中主要存储的是结构化数据，从而便于快速查找。下面的这些数据是存在数据库中。

1. nodes 表

   该表用于存储 P2P 网络上除本节点之外的其他节点的基本信息，表的结构如下所示。

属性	类型	说明
id	integer	PK. 节点编号
ipv4	varchar(32)	节点的 IPv4 地址
ipv6	varchar(128)	节点的 IPv6 地址
port	integer	节点可供访问的端口
mac	varchar(128)	节点的物理 MAC 地址
bandwidth	double	节点所处位置的平均带宽
longtitude	varchar(32)	物理位置的经度
latitude	varchar(32)	物理位置的纬度

2. interest_sets 表

   该表用于存储在用户兴趣关系层中节点的兴趣集或者兴趣导向边的兴趣集，表中的每一条记录表示一个用户兴趣元。表的结构如下所示。

属性	类型	说明
id	integer	PK.
node_id	integer	FK. 节点编号
type	varchar(32)	兴趣元的类型：节点或兴趣导向边
interest_id	integer	全局兴趣点的编号
weight	double	该兴趣点的权重

3. resources 表

   该表用于存储用户发表的或者接受的文本资源，表的结构如下所示。

属性	类型	说明
id	integer	PK. 文本资源编号
node_id	integer	FK. 发布资源的节点编号
topics	text	文本资源的主题向量
created	datetime	资源创建时间
modified	datetime	资源修改时间

4. reviews 表

该表用于存储用户对文本资源的反馈，表的结构如下所示：

属性	类型	说明
id	integer	PK. 资源反馈的编号
resource_id	integer	FK. 资源的编号
node_id	integer	FK. 用户反馈所在的节点编号
duration	double	查看资源的时间
comment	varchar(1024)	评论内容
created	datetime	反馈创建的时间

7.4.2 节点系统的文件系统

在文件系统中主要存储一些在关系型数据库中无法存储的非结构化的数据，主要包括以下几种类型：

- 文本资源：由于纯文本的资源文件大小浮动很大，是典型的非结构化数据，如果在数据库中作为 text 类型进行存储往往会影响到查询性能。而且对于互联网资源中的文本来说，往往使用 html 语言来描述的，还会包括图片、超链接、字体格式等内容。对于这种情况，本系统将所有的资源均存储在文件系统中。
- 全局主题：在第三章和第四章对文章的主题进行了定义，考虑到主题是一个关于词汇的向量，维度十分巨大，因此在数据库中很难用这么多行列来存储。因此这里的做法是将主题向量表示成 JSON 的格式，从而方便各种编程语言调用。
- 兴趣树：兴趣树分为 DIST 和 SICT。其中，DIST 的特征是动态变化十分频繁，但是结构不是很大，所以这里只是在文件系统上存储一个序列化的数

据结构文件，当系统运行时，会自动将该数据结构反序列化载入内存，从而加快处理速度，当系统要关闭时，会向文件系统重新更兴趣树文件。SICT 的特征则相反，其一般是静态保持不变的，但结构十分庞大，因此系统将这个结构依然存储为文件，但是载入内存是选择局部载入，这样节省了内存的开销。

- 词汇表：词汇表相当于电子词典，有两种存储方式。第一，可以建立带有索引的词汇表文件，这样管理的代价比较大，但是处理速度很快；第二，直接调用公开的 API，英文词库如 WordNet[5]等，但是在网络开销上会有一定的影响。

7.4.3 节点系统的内存

将数据存储在内存中是为了提高系统的处理速度，除了上述提到的需要序列化的数据结构之外，缓存和路由表是内存中最重要的内容。与传统的 RAM 一样，这里的缓存数据会在系统断电后直接消失，因此，为了解决这个问题，我们采用了定期备份重要数据至文件系统的策略。所谓重要的信息，一般是指访问次数相对较多的其他节点列表、请求次数较多的文本资源等等。

7.5 节点系统的关键功能实现

//TODO

7.6 小结

本章主要说明了基于 P2P 网络的信息自主流动机制的原型系统。首先分析了不同角色之间拥有的功能有哪些，并用用例图详细分析了各个模块的功能需求。然后，为了使得系统能够同时充当服务器和客户端，文中提出了一种全新的系统架构，将节点服务端程序和节点客户端程序分为两个 web 应用，并使得两者分别向外提供不同的调用接口，供本地用户或者外来节点使用。此外，本章还说明了系统在存储方面的实现细节，采用了三种存储介质：数据库、文件系统和内存，分别为了适应不同类型的数据和资源。

[5]http://wordnet.princeton.edu/

第 8 章　结论与展望

// todo

8.1　总结

// todo

8.2　展望

// todo

致谢

逾尺的札记和研究纪录凝聚成这么薄薄的一本, 高兴和欣慰之余, 不禁感慨系之。记得鲁迅在一篇文章里写道:“人类的奋战前行的历史, 正如煤的形成, 当时用大量的木材, 结果却只是一小块”。倘若这一小块有点意义的话, 则是我读书生活的最好纪念, 也令我对于即将迈入的新生活更加充满信心。回想读书生活, 已经整整二十个年头, 到同济求学将近五年, 攻读博士学位也已三年了。进入同济大学以来, 深深醉心于一流学府的大家风范。名师巨擘, 各具特点; 中西融合, 文质相顾。处如此佳境以陶铸自我, 实乃人生幸事。

2015 年 3 月

参考文献

[1] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, ``A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys & Tutorials, IEEE*, vol. 7, no. 2, pp. 72--93, 2005.

[2] J. Han, M. Kamber, and J. Pei, *Data mining, southeast asia edition: Concepts and techniques*. Morgan kaufmann, 2006.

[3] W. B. Croft, D. Metzler, and T. Strohman, *Search engines: Information retrieval in practice*. Addison-Wesley Reading, 2010.

[4] D. A. Grossman, *Information retrieval: Algorithms and heuristics*, vol. 15. Springer Science & Business Media, 2004.

[5] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*, vol. 1. Cambridge university press Cambridge, 2008.

[6] G. Salton and M. J. McGill, ``Introduction to modern information retrieval," 1983.

[7] D. E. Appelt, J. R. Hobbs, J. Bear, D. Israel, and M. Tyson, ``Fastus: A finite-state processor for information extraction from real-world text," in *IJCAI*, vol. 93, pp. 1172--1178, 1993.

[8] R. Mooney, ``Relational learning of pattern-match rules for information extraction," in *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pp. 328--334, 1999.

[9] R. Dugad and U. B. Desai, ``A tutorial on hidden markov models," *Signal Processing and Artificial Neural Networks Laboratory Department of Electrical Engineering Indian Institute of Technology*, 1996.

[10] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra, ``A maximum entropy approach to natural language processing," *Computational linguistics*, vol. 22, no. 1, pp. 39--71, 1996.

[11] J. Lafferty, A. McCallum, and F. C. Pereira, ``Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[12] N. Kambhatla, ``Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations," in *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, p. 22, Association for Computational Linguistics, 2004.

[13] Z. GuoDong, S. Jian, Z. Jie, and Z. Min, ``Exploring various knowledge in relation extraction," in *Proceedings of the 43rd annual meeting on association for computa-*

tional linguistics, pp. 427--434, Association for Computational Linguistics, 2005.

[14] J. Jiang and C. Zhai, ``A systematic exploration of the feature space for relation extraction.,'' in *HLT-NAACL*, pp. 113--120, 2007.

[15] Y. S. Chan and D. Roth, ``Exploiting background knowledge for relation extraction,'' in *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 152--160, Association for Computational Linguistics, 2010.

[16] R. C. Bunescu and R. J. Mooney, ``A shortest path dependency kernel for relation extraction,'' in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 724--731, Association for Computational Linguistics, 2005.

[17] S. Zhao and R. Grishman, ``Extracting relations with integrated information using kernel methods,'' in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pp. 419--426, Association for Computational Linguistics, 2005.

[18] T. Dunning, ``Accurate methods for the statistics of surprise and coincidence,'' *Computational linguistics*, vol. 19, no. 1, pp. 61--74, 1993.

[19] S. Gupta, A. Nenkova, and D. Jurafsky, ``Measuring importance and query relevance in topic-focused multi-document summarization,'' in *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pp. 193--196, Association for Computational Linguistics, 2007.

[20] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, ``Indexing by latent semantic analysis,'' *JAsIs*, vol. 41, no. 6, pp. 391--407, 1990.

[21] H. Daumé III and D. Marcu, ``Bayesian query-focused summarization,'' in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pp. 305--312, Association for Computational Linguistics, 2006.

[22] A. Haghighi and L. Vanderwende, ``Exploring content models for multi-document summarization,'' in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 362--370, Association for Computational Linguistics, 2009.

[23] D. Wang, S. Zhu, T. Li, and Y. Gong, ``Multi-document summarization using sentence-based topic models,'' in *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pp. 297--300, Association for Computational Linguistics, 2009.

[24] A. Celikyilmaz and D. Hakkani-Tur, ``A hybrid hierarchical model for multi-

document summarization," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 815--824, Association for Computational Linguistics, 2010.

[25] K. McKeown, J. Klavans, V. Hatzivassiloglou, R. Barzilay, and E. Eskin, ``Towards multidocument summarization by reformulation: Progress and prospects," in *AAAI/IAAI*, pp. 453--460, 1999.

[26] V. Hatzivassiloglou, J. L. Klavans, M. L. Holcombe, R. Barzilay, M.-Y. Kan, and K. McKeown, ``Simfinder: A flexible clustering tool for summarization," 2001.

[27] D. M. Blei, A. Y. Ng, and M. I. Jordan, ``Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993--1022, 2003.

[28] G. Heinrich, ``Parameter estimation for text analysis," tech. rep., Technical report, 2005.

[29] A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh, ``On smoothing and inference for topic models," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 27--34, AUAI Press, 2009.

[30] A. Smola and S. Narayanamurthy, ``An architecture for parallel topic models," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 703--710, 2010.

[31] J. Varadarajan, R. Emonet, and J.-M. Odobez, ``Probabilistic latent sequential motifs: Discovering temporal activity patterns in video scenes," tech. rep., Idiap, 2010.

[32] L. L. Mølgaard, J. Larsen, and C. Goutte, ``Temporal analysis of text data using latent variable models," in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing, 2009.*, 2009.

[33] D. M. Blei and J. D. Lafferty, ``Dynamic topic models," in *Proceedings of the 23rd international conference on Machine learning*, pp. 113--120, ACM, 2006.

[34] J. Chang and D. M. Blei, ``Relational topic models for document networks," in *International Conference on Artificial Intelligence and Statistics*, pp. 81--88, 2009.

[35] S. Gong, ``The personalized information retrieval model based on user interest," *Physics Procedia*, vol. 24, pp. 817--821, 2012.

[36] H. R. Kim and P. K. Chan, ``Learning implicit user interest hierarchy for context in personalization," in *Proceedings of the 8th international conference on Intelligent user interfaces*, pp. 101--108, ACM, 2003.

[37] Y. Li and N. Zhong, ``Mining ontology for automatically acquiring web user information needs," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 18, no. 4, pp. 554--568, 2006.

[38] X. Tao, Y. Li, and N. Zhong, ``A personalized ontology model for web information gathering," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 23, no. 4, pp. 496--511, 2011.

[39] A. Pretschner and S. Gauch, ``Ontology based personalized search," in *Tools with Artificial Intelligence, 1999. Proceedings. 11th IEEE International Conference on*, pp. 391--398, IEEE, 1999.

[40] A. Sieg, B. Mobasher, and R. Burke, ``Web search personalization with ontological user profiles," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 525--534, ACM, 2007.

[41] X. Tao, Y. Li, N. Zhong, and R. Nayak, ``Ontology mining for personalized web information gathering," in *Web Intelligence, IEEE/WIC/ACM International Conference on*, pp. 351--358, IEEE, 2007.

[42] J. Trajkova and S. Gauch, ``Improving ontology-based user profiles.," in *RIAO*, vol. 2004, pp. 380--390, 2004.

[43] A. Sieg, B. Mobasher, and R. D. Burke, ``Learning ontology-based user profiles: A semantic approach to personalized web search.," *IEEE Intelligent Informatics Bulletin*, vol. 8, no. 1, pp. 7--18, 2007.

[44] B. Pang and L. Lee, ``Opinion mining and sentiment analysis," *Foundations and trends in information retrieval*, vol. 2, no. 1-2, pp. 1--135, 2008.

[45] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, ``Lexicon-based methods for sentiment analysis," *Computational linguistics*, vol. 37, no. 2, pp. 267--307, 2011.

[46] M. Taboada, C. Anthony, J. Brooke, K. Dilkina, M. A. Gillies, J. Grieve, M. Hay, P. Larrivee-Woods, G. Lyons, P. McFetridge, *et al.*, ``Thumbs up or thumbs down?," 2007.

[47] E. Riloff, S. Patwardhan, and J. Wiebe, ``Feature subsumption for opinion analysis," in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp. 440--448, Association for Computational Linguistics, 2006.

[48] E. Riloff and J. Wiebe, ``Learning extraction patterns for subjective expressions," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pp. 105--112, Association for Computational Linguistics, 2003.

[49] J. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin, ``Learning subjective language," *Computational linguistics*, vol. 30, no. 3, pp. 277--308, 2004.

[50] T. Wilson, J. Wiebe, and R. Hwa, ``Recognizing strong and weak opinion clauses," *Computational Intelligence*, vol. 22, no. 2, pp. 73--99, 2006.

[51] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller, ``Introduction to wordnet: An on-line lexical database*," *International journal of lexicography*, vol. 3, no. 4, pp. 235--244, 1990.

[52] X. Su and T. M. Khoshgoftaar, ``A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, p. 4, 2009.

[53] Y. Shi, M. Larson, and A. Hanjalic, ``Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Computing Surveys (CSUR)*, vol. 47, no. 1, p. 3, 2014.

[54] M. Ripeanu, I. Foster, and A. Iamnitchi, ``Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *arXiv preprint cs/0209028*, 2002.

[55] M. Ripeanu, ``Peer-to-peer architecture case study: Gnutella network," in *Peer-to-Peer Computing, 2001. Proceedings. First International Conference on*, pp. 99--100, IEEE, 2001.

[56] B. Y. Zhao, J. Kubiatowicz, A. D. Joseph, *et al.*, ``Tapestry: An infrastructure for fault-tolerant wide-area location and routing," 2001.

[57] A. Rowstron and P. Druschel, ``Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *Middleware 2001*, pp. 329--350, Springer, 2001.

[58] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, ``Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149--160, 2001.

[59] N. S. Good and A. Krekelberg, ``Usability and privacy: a study of kazaa p2p file-sharing," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 137--144, ACM, 2003.

[60] D. J. Watts and S. H. Strogatz, ``Collective dynamics of ‘small-world’ networks," *nature*, vol. 393, no. 6684, pp. 440--442, 1998.

[61] D. Tsoumakos and N. Roussopoulos, ``Analysis and comparison of p2p search methods," in *Proceedings of the 1st international conference on Scalable information systems*, p. 25, ACM, 2006.

[62] N. Bisnik and A. Abouzeid, ``Modeling and analysis of random walk search algorithms in p2p networks," in *Hot topics in peer-to-peer systems, 2005. HOT-P2P*

2005. Second International Workshop on, pp. 95--103, IEEE, 2005.

[63] O. Babaoglu, H. Meling, and A. Montresor, ``Anthill: A framework for the development of agent-based peer-to-peer systems,'' in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, pp. 15--22, IEEE, 2002.

[64] T. L. Griffiths and M. Steyvers, ``Finding scientific topics,'' *Proceedings of the National Academy of Sciences*, vol. 101, no. suppl 1, pp. 5228--5235, 2004.

[65] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, ``Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora,'' in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pp. 248--256, Association for Computational Linguistics, 2009.

[66] A. Kontostathis, L. M. Galitsky, W. M. Pottenger, S. Roy, and D. J. Phelps, ``A survey of emerging trend detection in textual data mining,'' in *Survey of Text Mining*, pp. 185--224, Springer, 2004.

[67] G. Liu, H. Shen, and L. Ward, ``An efficient and trustworthy p2p and social network integrated file sharing system,'' *Computers, IEEE Transactions on*, vol. 64, no. 1, pp. 54--70, 2015.

[68] M. Yang and Y. Yang, ``An efficient hybrid peer-to-peer system for distributed data sharing,'' *Computers, IEEE Transactions on*, vol. 59, no. 9, pp. 1158--1171, 2010.

个人简历、在读期间发表的学术论文与研究成果

个人简历：

先毅昆，男，1989 年 11 月出生。

2012 年 6 月毕业于同济大学软件学院，软件工程专业，获学士学位。

2012 年 9 月入同济大学软件学院，软件工程专业，攻读硕士学位。

已发表论文：

[1] Yikun Xian, Jie Huang, Yefim Shuf, Gene Fuh, and Zhen Gao. An Approach for In-Database Scoring of R Models on DB2 for z/OS. Rough Sets and Knowledge Technology. Springer International Publishing, 2014. 376-385.

[2] Yikun Xian, Jiangfeng Li, Chenxi Zhang, Zhenyu Liao. Video Highlight Shot Extraction with Time-Sync Comment. ACM MobiHoc 2015 - HotPOST '15.

待发表论文：

发明专利：

[1] 张晨曦, 先毅昆, 李江峰. 一种用户兴趣获取与传播的方法和装置: 中国, 201410494809.4[P]. 2015.01.07.