中国农业大学

本科生毕业论文

论文题目 在线社交网络用户行为分析——微博短视频推荐

学生姓名 李观波 学号 1206020120

专　　业 计算机科学与技术 年级 13 级

指导教师 贾璐 职称 副教授

学　　院 信息与电气工程学院

中国农业大学教务处制

2017 年 6 月

摘　要

在互联网内容承载方式不断变化的背景下，短视频因为具有丰富的视听信息、占用时间短的特点，受到了用户的欢迎。在社交网络平台微博上，每日短视频上传量达 32 万，用户想要从数量庞大的短视频中找到自己感兴趣的短视频变得十分困难。目前微博推荐短视频存在的主要问题有：缺少个性化推荐、“热门视频”不热门、分类存在问题，推荐的短视频很大程度上不符合用户的兴趣。因此需要设计一种个性化的短视频推荐算法，筛选出用户真正感兴趣的短视频进行推荐，满足用户的观看需求。

本文基于在线社交网络用户行为分析，分析用户行为数据对短视频进行分类、推荐，为此，首先构建了基于 Scrapy 的短视频数据爬虫工具，获取所需的短视频数据；其次基于余弦相似度构建了短视频相似关系网络；最后使用社区发现算法 Fast Unfolding 对短视频进行分组，基于视频相似度及用户过往行为分析实现个性化的短视频推荐。

关键词：微博，行为分析，短视频，推荐

Abstract

In the background of Internet contents changing, short videos have become one of the most popular ways of entertainment, because of the abundance of information and a relatively time-saving way of entertainment that based on videos. Weibo is one of the most popular social networks in China. It has an average of 32 million short videos uploaded by users every day, and therefore it is difficult for users to find what they are interested in. There are some minor flaws in Weibo's short videos recommendation system: recommendation not personalized, hot videos not actually popular, classification not quite right. It makes the recommendation not match users' interest sometimes. It is necessary to find out a recommendation algorithm to recommend short videos to users.

This paper analyses users' behaviors in the online social network, and further utilizes behavior data to classify and recommend short videos. To do so, first, we use Scrapy to extract needed data. Second, a short videos network is constructed based on similarity. Third, we use Fast Unfolding Algorithm to classify short videos. The similarity among short videos and the analysis of users' past behavior will be used for personalized short videos recommendation.

Keywords: Weibo, behaviors analysis, short videos, recommendation

目　　录

第 1 章 绪论

1.1 引言

近年来，我国的互联网蓬勃发展。截止到 2016 年 12 月，我国网民规模达到了 7.31 亿，其中手机网民规模达到 6.95 亿。随着移动通信基础设施进一步完善，越来越多的网民使用手机上网，网民中使用手机上网人群的占比由 2015 年的 90.1%提升至 95.1%。

2016 年，我国互联网各类社交应用持续稳定发展，互联网平台实现泛社交化。一方面，综合性社交应用引入短视频等服务，带来用户和流量的增长，另一方面，针对不同场景、不同垂直人群、不同信息承载方式的细分社交平台进一步丰富，向创新、小众化方向发展。

微博，作为社交应用之一，得益于名人明星、网红和媒体内容生态的建立与不断强化，以及在短视频上的深入布局，用户使用率持续回升，达到 37.1%，与 2016 年 6 月相比提升 3.1%。微博用户规模从 2015 年的 2.3 亿增至 2016 年的 2.71 亿，全年增长率 17.8%，手机微博用户规模从 2015 年的 1.87 亿增至 2016 年的 2.41 亿，全年增长率 28.9%[1]。

在微博平台上，短视频的播放量峰值达到了 23 亿次，人均播放时长 15.2 分钟, 短视频每天发布量达到 32 万条[2]。

本文选取社交应用微博作为研究对象，通过对微博短视频数据的抓取、分析和处理，分析用户行为，为用户推荐视频。

1.2 研究背景与意义

1.2.1 研究背景

移动互联网的快速发展给我们的生活带来了诸多便利，我们可以通过移动互联网随时随地了解身边所发生事情，网络信息更新的速度越来越快，大部分人的时间越来越碎片化，这种矛盾使我们所消费的网络内容在发生变化[3] 。

互联网内容的承载方式有很多，按照出现的先后顺序大致为：文字、图片、音频、视频等，发展的趋势从低维到高维。不同互联网内容的承载方式本身的形式也在发生着变化，文字从没有字数限制的博客，到限制在 140 字左右的微博；图片由黑白变成彩色，由静态变成动态;音频从调频广播到语音消息;视频从几十分钟的长视频，到现在几分钟甚至几十秒的短视频，同时，电视剧、电影本身虽然是长视频内容，但也出现了微电视剧、微电影的短视频形式。由此可见，用户消费的单个网络内容占用的时间越来越短。

在互联网内容的承载方式中，视频具有较高的信息密度，既能传递文字、图片、音频所有信息，同时还能刺激接收者的视觉和听觉，使传递的信息更容易被理解和接受。其中，短视频由于

在包含丰富视听信息的前提下，又不会占用用户太多时间，深受用户喜欢，用户可以在碎片化时间进行观看。短视频是目前较好的传播形式。

1.2.2 研究意义

在微博平台上，短视频的播放量峰值达到了 23 亿次，人均播放时长 15.2 分钟。

根据微博官方的一项调查结果显示，76.6%的用户喜欢观看并且经常观看短视频，用户对短视频未来的发展持积极态度。83.2%的用户会因为短视频内容有趣而分享，79.2%的用户会因为对短视频内容不感兴趣而不看或者不完整观看[2]，短视频内容的好坏是影响用户是否观看的重要因素。面对每天发布量达 32 万条的短视频，用户想要从中找到自己感兴趣的短视频变得十分困难，视频推荐系统是解决这个问题的有效方法。个性化的视频推荐系统，不仅可以帮助用户找到感兴趣的视频,还可以帮助用户挖掘自己的兴趣。

据调查，微博的视频推荐系统存在以下问题：

（1）缺少个性化推荐

微博向用户推荐的视频为热门视频，用户并不一定对热门视频感兴趣，而一些用户真正感兴趣的但不热门的视频用户却无法观看。

（2）“热门视频”不热门

微博视频首页推荐的“热门视频”播放量虚高，对这些视频进行简要分析，视频页面标注的视频播放量超过千万，但该视频下转发、评论、点赞的用户却很少；

（3）分类存在问题

微博按照视频上传博主的类型对视频进行分类，一个经常上传 A 类型视频的博主，如果突然上传了一个 B 类型的视频，微博会将该视频分在 A 类型中，从而造成视频分类的错误。

因此，通过对微博用户兴趣的分析，构建个性化的视频推荐方法，有一定的研究意义。

1.3 国内外研究现状

协同过滤(Collaborative Filtering, CF)是目前视频推荐中，被普遍接受并广泛采用的机制。协同过滤的总体思想是，将一个用户对于视频的评价与其他用户的评价进行比较，发现与该用户兴趣相似的人，将这些人感兴趣的视频推荐给这个用户。这种推荐方法的关键在于，如何确定用户与用户之间具有兴趣相似，如何提取用户的兴趣 [4]。

目前，国内外研究人员已经提出了一些方法和策略来解决这一问题，例如：采用基于社交网络的视频推荐方法，认为社交网络中的朋友应该具有相似的视频喜好，分析用户的社交网络关系，根据用户的朋友的视频喜好进行推荐[5] ；采用联合模型，基于组内其它用户信息，预测用户的兴趣点[6] ；基于情感分析的视频推荐方法，通过判断用户当前的情绪来推荐合适的视频[7]，通过识别用户的面部表情，分析用户情绪并推荐适合当前情绪的视频[8]；建立用户-视频图表示各个用户的观看信息，通过对图进行遍历，获取邻近节点视频的标签，根据标签进行推荐[9] 。

协同过滤最大的困难来自于视频推荐中两个大数据的问题，即视频数据和观看用户数据。对这两种大数据进行分类、聚类的代价都非常高。因此，避免直接对视频数据和观看用户数据进行处理，采用其它的方式来进行视频推荐，成为本文探索的一个方向。

1.4 研究内容

1.4.1 研究方法概述

目前用户规模排名前三的社交应用分别是微信朋友圈、QQ 空间、微博，三者虽然同属于综合类社交应用，但在交流属性上存在较大差异。微信朋友圈是相对封闭的个人社区，分享的信息偏向朋友之间的互动，微博是基于社交关系来进行信息传播的公开平台，用户关注的内容越来越倾向于基于兴趣的垂直领域，QQ 空间则介于两者之间[1]。

广大的用户群体可以在微博平台上真实、自发地表达或分享自己的情感、观点和态度。短视频的主要传播渠道是社交平台，具有强烈的社交属性，用户在观看视频后有评论的习惯，这种习惯的快速成熟，使短视频下的人均评论量与 2016 年初相比增长 8 倍。视频上传者的粉丝用户人均评论数是其他用户的 80 多倍，大多数粉丝用户还会在视频下回复他人的评论来交流感受[10]。微博平台为分析用户兴趣视频提供了大量真实、可靠的数据。

针对微博公开的特点，本文采用的研究方法是：从微博平台抓取短视频数据，获取用户行为信息；根据用户在短视频下的评论内容，提取短视频的关键词；计算短视频之间的相似度，根据相似度对短视频进行分组；分析用户转发的短视频，将其所属分组下的其他短视频按相似度排序，根据相似度向用户推荐。

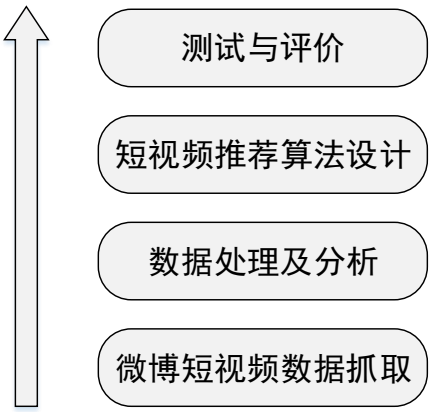

图 1-1 研究方法与思路

本文涉及的范畴包括：

（1）构建微博短视频数据爬虫工具；

（2）抓取微博短视频数据；

（3）对短视频数据进行处理和分析；

（4）设计短视频推荐算法；

（5）短视频算法的测试与评价

1.4.2 需要解决的问题

在明确本文的研究范围和采取的研究方法后，提出下列需要解决的问题：

（1）分析获取短视频数据的请求，建立获取短视频数据的爬虫工具，模拟发送数据请求获取短视频数据；

（2）实现对短视频评论内容的筛选、分词，对短视频进行关键词提取；

（3）计算短视频之间的相似度；

（4）基于相似度对短视频进行分组；

（5）分析用户感兴趣的短视频，向用户推荐短视频。

第 2 章 开发技术简介

2.1 Linux

Linux 操作系统是目前最受关注的操作系统之一，主要原因是它的免费，以及系统的开放性。本文选择 Linux 操作系统有三个原因：

（1）系统稳定、效率高

Linux 是基于 UNIX 的概念而开发出来的操作系统，具有与 UNIX 系统相似的程序接口和操作方式，继承了 UNIX 稳定并且效率高的特点。

（2）多用户、多任务

Linux 系统调度每一个进程平等地访问处理器，所以它能同时执行多个程序，而且各个程序的运行是互相独立的。利用这个特点，在抓取数据的时候，可以同时执行多个爬虫程序，提高爬虫效率。

（3）高安全性

Linux 系统是一个具有先天病毒免疫能力的操作系统，很少受到病毒攻击。利用 Linux 自带防火墙、入侵检测和安全认证等工具，及时修补系统的漏洞，大大提高 Linux 系统的安全性，保证在开发阶段不会受到病毒攻击。

2.2 Python

Python 是一种面向对象的解释型计算机程序设计语言，本文使用 Python 作为开发语言的原因有三个：

（1）Python 编写代码的速度非常的快，代码的可读性高，具备更好的可重用性，方便维护，与现在流行的编程语言 Java、C、C++等相比较，完成同样的功能，Python 编写的代码更短，开发的效率是其它语言的好几倍。

（2）Python 支持多平台开发，用它编写的代码可以不经过任何转换就能在 Linux 与 Windows 系统任何移植，系统提供的一些 GUI 图形化编程、数据库操作、网页网络编程接口都可以直接移植到任何系统中。

（3）Python 有非常丰富的标准库(Standard Library)，标准库的这些模块从字符串到网络脚本编程、游戏开发、科学计算、数据库接口等为开发提供了便利。

本文在数据获取、数据分析、算法实现等研究上均使用 Python 实现。

2.3 Scrapy

Scrapy 是一套基于 Twisted 的异步处理框架，是纯 Python 实现的爬虫框架，用户只需要定制开发几个模块就可以轻松的实现一个爬虫，抓取网页内容或者各种图片。

Scrapy 的大体架构包括：调度(Scheduler)、项目管道(Item Pipeline)、下载器(Downloader)、Spiders 以及引擎(Engine)。

Scrapy 的整个数据处理流程由 Scrapy 引擎进行控制，其主要的运行方式分为如下几个步骤：

（1）Scrapy 引擎从 Spider 获取初始请求 URL；

（2）Scrapy 引擎将 URL 作为请求在调度中进行调度，从调度获取下一个要爬取的 URL；

（3）调度将下一个爬取的 URL 返回给引擎；

（4）Scrapy 引擎将 URL 通过下载中间件发送到下载器；

（5）页面下载完成后，下载器将生成响应，通过下载中间件发送到引擎；

（6）Scrapy 引擎从下载器接收响应，通过 Spider 中间件发送到 Spider 进行处理；

（7）Spider 处理响应，通过 Spider 中间件返回抓取的项目和发送新的请求到引擎；

（8）引擎将抓取的项目发送到项目管道，将已处理的请求发送到调度，并获取下一个 Spider 请求；

（9）重复上述操作，直到调度中没有请求[11]。

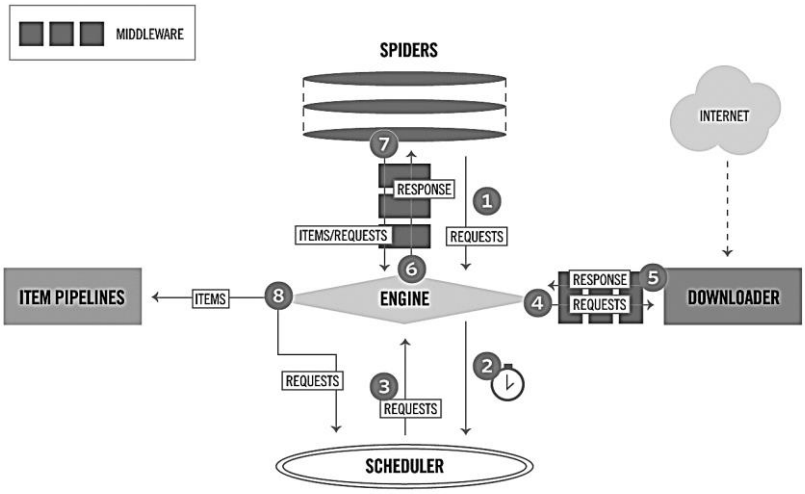

图 2-1 Scrapy 总体框架及数据处理流程

2.4 本章小结

本章通过分析确定了开发环境，在 Linux 操作系统下，使用 Python 作为编程语言，基于 Scrapy 框架构建爬虫系统。

第 3 章 基于 Scrapy 的短视频数据爬虫的设计

3.1 爬虫流程

Scrapy 包含各种中间件接口，可以灵活完成各种需求。基于 Scrapy 的微博短视频数据爬虫的流程分为以下几步：

（1）获取初始 URL，插入 URL 队列

（2）爬取短视频页面，提取所需数据，不同的数据类型处理方式不同

（3）对于短视频数据，将数据存储到数据库

（4）对于视频 URL，判断该 URL 是否已经爬取，如果没有爬取，则将该 URL 插入队列

（5）URL 队列为空，爬虫结束

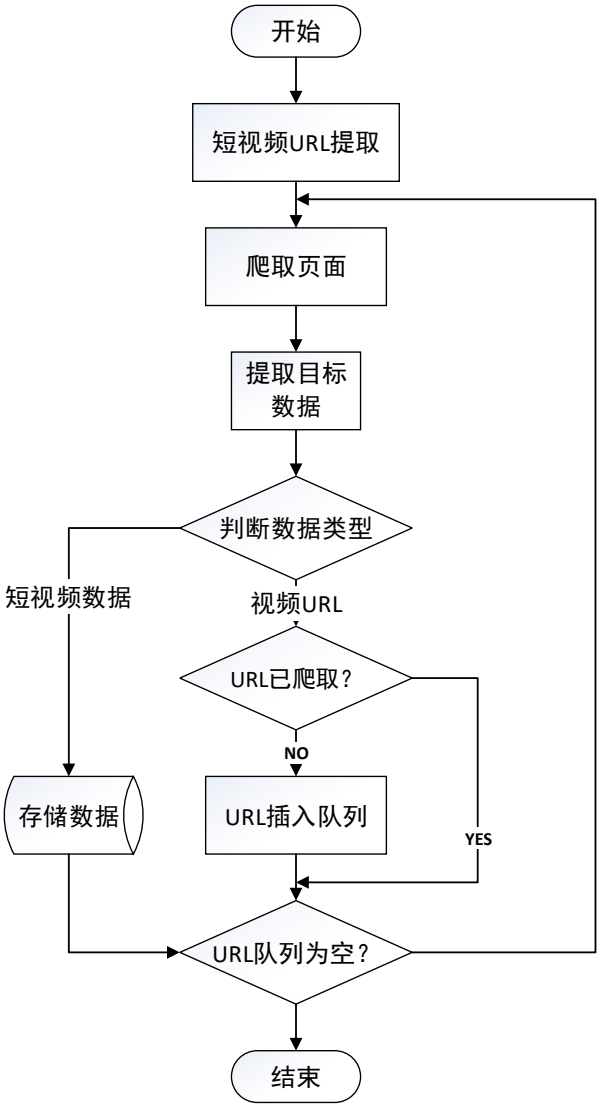

图 3-1 爬虫流程图

3.2 微博短视频网页结构分析

3.2.1 短视频内容

如图 3-2 所示，可供获取的短视频数据有：短视频上传者、短视频描述、播放量、转发数、评论数、点赞数。

图 3-2 短视频内容页面

3.2.2 用户转发

如图 3-3 所示，短视频微博下的用户转发分页显示，可分页获取转发该视频的用户及转发内容。

图 3-3 短视频转发区

3.2.3 用户评论

如图 3-4 所示，评论用户及内容没有分页显示，当阅读到最后一条评论时，需单击“查看更多”来获取更多评论。

图 3-4 短视频评论区

3.2.4 更多短视频 URL

如图 3-5 所示，从短视频页面可提取更多短视频 URL。

图 3-5 更多短视频

3.3 微博短视频爬虫

3.3.1 模拟登陆

微博是一个公开的社交平台，但这种公开是有限制的，如果用户在访客状态（即未登录状态）下访问微博，只能浏览到微博的一部分内容，甚至无法阅读微博下的评论，无法知道有哪些用户转发过该微博，在这种用户状态下对微博短视频数据进行抓取，就失去了意义。因此，需要模拟用户登陆状态进行访问。

HTTP 协议中向服务器请求数据的方法有 GET 和 POST，用户在微博登陆时，页面发送的登陆请求是 POST。要使用发送 POST 请求来模拟登陆状态，需要理解所有请求参数的含义，最困难的是需要清楚微博是如何对登陆密码进行加密的。因此，这种方法可行性较低。

Internet 的各种服务系统常常需要记录访问者的一些信息，用于辨别用户身份、进行 Session 跟踪，服务器会在用户本地终端上存储一些数据，通常都经过加密，这些数据就是 Cookies。Cookies 最典型的应用是判定用户是否已经登录网站，用户在登陆时勾选“自动登陆”或“保持登陆状态”复选框，登陆成功后会在一定时间内保持用户的登陆状态，以便用户再次进入此网站时简化登录手续。通过提取登陆成功后服务器保存在本地的 Cookies，替换数据请求中的 Request Cookies，达到模拟用户登陆状态的目的。

本文使用了 Selenium 实现自动登陆并提取 Cookies 的功能。Selenium 通过浏览器驱动对浏览器进行控制，如图 3-6 所示，根据登陆框的表单标签，选中登录框并输入存储在本地的账号和密码；输入完成后，根据表单标签选中登陆按钮，单击按钮进行登陆；等待页面成功跳转后，完成自动登陆的过程，提取浏览器 Cookies 内容，以 JSON 文件格式保存到本地，供发送数据请求时使用。

图 3-6 Selenium 控制浏览器实现自动登陆

3.3.2 数据定义

Item 是保存爬取到数据的容器，根据获取到的数据对 Item 进行统一建模。在 Item 中定义相应字段的 Field，对应的 items.py 的主要代码为：

```
class WeiboItem(scrapy.Item):
    author = scrapy.Field() # 上传者
    content = scrapy.Field()    # 视频描述
    id = scrapy.Field() # 视频 id
    url = scrapy.Field()    # 视频 url
    comments_num = scrapy.Field()   # 评论数
    forwards_num = scrapy.Field()   # 转发数
    likes_num = scrapy.Field()  # 点赞数
    comments = scrapy.Field()   # 评论
    forwards = scrapy.Field()   # 转发
```

3.3.3 数据获取请求方式

与微博登陆请求不同，获取微博数据的请求方法是 GET。本文通过开发者工具直接从请求 URL 获得所需要的参数，分析其中部分重要参数的含义，替换这些参数的值拼接请求 URL，发送请求 URL 获取微博数据。

短视频属于微博，但有区别于普通微博的特征。普通微博通过 16 位数字组成的 id 标识，短视频在 id 标识的基础上，还通过由 9 位英文字母和数字组成的短链接(short URL)标识，两种标识方式应用于不同的场合：获取短视频基本信息页面时，使用 short URL 进行请求；获取短视频下转发、评论、赞的用户行为数据时，使用 id 进行请求。

针对不同类型的数据，分别说明数据获取的具体请求方式：

（1）基本信息

短视频页面有统一的 URL 前缀(http://weibo.com/tv/v/)，与短视频 short URL 拼接构成完整 URL(http://weibo.com/tv/v/ + short URL)，对 URL 进行访问，获取该视频 HTML 页面数据。

（2）用户转发

短视频转发数据请求的 URL 如下：

http://weibo.com/aj/v6/mblog/info/big?ajwvr=6&id=4094887219662546&max_id=4110452914704513&page=1&__rnd=1497023805769

其中关键参数含义如下：

- id：短视频的标识；
- max_id：该短视频最新一条转发信息的标识；
- page：获取转发页面的页码；

- __rnd：请求时间，使用 UNIX 时间戳表示。

将请求参数替换为%s，得到转发数据请求的 URL 格式：

http://weibo.com/aj/v6/mblog/info/big?ajwvr=6&id=%s&max_id=%s&page=%s&__rnd=%s

（3）用户评论

短视频评论数据的请求比其他数据的获取复杂。评论内容的加载没有进行分页，无法通过页码获取评论内容；更多评论内容需要用户点击当前最后一条评论后的“查看更多”，网页会向服务器发送一次数据请求，获取 15 个评论条目，每个评论条目下包括若干二级评论，使得最终获取的评论数量变化。

获取评论数据的请求 URL 不统一，第一次请求的 URL 区别于其他请求,第一次请求 URL 如下：

http://weibo.com/aj/v6/comment/big?ajwvr=6&id=4094887219662546&filter=all&from=singleWeiBo&__rnd=1497026383906

其中关键参数含义如下：

- id：短视频的标识；
- filter：排序方式，all 为按时间排序，hot 为按热度排序；
- __rnd：请求时间，使用 UNIX 时间戳表示；

只需将 id 参数替换为请求评论数据视频的 id，就能得到第一次评论数据请求的 URL。

其他请求的 URL 如下：

http://weibo.com/aj/v6/comment/big?ajwvr=6&id=4094887219662546&root_comment_max_id=4095167348577093&root_comment_max_id_type=&root_comment_ext_param=&page=2&filter=all&sum_comment_number=15&filter_tips_before=0&from=singleWeiBo&__rnd=1497026992516

其他请求与第一次请求相比，关键参数的数量增加，增加的关键参数含义如下：

- root_comment_max_id：当前最后一条评论 id 的序号，序号的值为 id 减 1；
- page：表示第几次进行数据请求；
- sum_comment_number：已获取评论（包含二级评论）的数目；

将请求参数替换为%s，得到评论数据请求的 URL 格式：

http://weibo.com/aj/v6/comment/big?ajwvr=6&id=%s&root_comment_max_id=%s&root_comment_max_id_type=&root_comment_ext_param=&page=%s&filter=all&sum_comment_number=%s&filter_tips_before=0&from=singleWeiBo&__rnd=%s

评论内容的排序方式有两种，all 按时间排序和 hot 按热度排序。选择 hot 按热度排序时，因为评论 id 和序号排序不一致，获取的 root_comment_max_id 出现错误，最终导致评论数据获取错误。因此，本文选择 all 按时间排序。

3.3.4 编写 Spider 提取数据

WeiboSpider 是用于爬取微博短视频数据的类，选择从 scrapy.spiders.CrawlSpider 类继承，定义以下的属性和方法：

- name 属性：定义 Spider 名字，唯一；
- start_urls 属性：Spider 初始化时，获取初始短视频 URL 作为 start_urls。Spider 启动时，从 start_urls 列表开始爬取，后续的 URL 从初始的页面提取；
- rules 属性：定义 URL 提取规则和回调函数；
- cookies 属性：用于模拟用户登陆状态；
- start_requests 方法：对 start_urls 列表中的 URL 进行访问；
- parse_video 方法：提取短视频基本信息，发送获取用户转发、评论请求；
- parse_forwards 方法：提取用户转发数据，如果仍有转发数据没有获取，发送获取用户转发请求；
- parse_comments 方法：提取用户评论数据，如果仍有评论数据没有获取，发送获取用户评论请求。

Scrapy 提取数据使用 Selector 选择器机制，使用特定的 Xpath 表达式提取 HTML 网页中的数据，短视频基本信息数据可以通过这种方式直接提取，用户转发、评论数据需要经过处理。

Scrapy 向服务器发送获取用户转发、评论数据请求后，服务器返回 JSON 格式数据，返回数据中 data 字段下 html 的内容是请求的结果，这些数据是没有经过解码(decode)的原始数据，对这些进行数据解码，使用 lxml.etree 将数据解析成 HTML，再利用 Xpath 提取 HTML 中所需要的数据。

3.3.5 数据存储方式

爬虫系统爬取的数据多为半结构化或非结构化数据，传统的关系型数据库并不擅长这类数据的存储与处理，而 NoSQL 在数据存取上具备关系型数据库无法比拟的性能优势，因此选择使用 NoSQL 数据库存储爬取到的数据。

MongoDB 是一个基于分布式文件存储的非关系型数据库，是一个介于关系型数据库和非关系型数据库之间的产品，是非关系型数据库中功能最丰富、最像关系型数据库的。MongoDB 具有以下特点，适合存储爬虫抓取的数据：

（1）面向集合存储

MongoDB 将数据分组存储在数据集合(Collection)中，每个集合都有一个唯一的标识名，数据集合类似于关系型数据库中的表(table)。

（2）BSON 格式存储

MongoDB 支持的数据结构非常松散，是类似 JSON 的 BSON 格式，可以存储比较复杂的数据类型。对于存储在 MongoDB 数据库中的数据集合，不需要知道它的结构定义，集合中的文档，

以键-值对的形式存储，键用于唯一标识一个文档，为字符串类型，值可以是各种复杂的类型。

（3）查询语言强大

MongoDB 最大的特点是支持的查询语言非常强大，几乎可以实现关系型数据库单表查询的绝大部分功能，而且还支持对数据完全索引，可以通过 MongoDB 的查询语句，查询包含内部对象在内的全部内容。

（4）实时数据处理

MongoDB 具有实时数据处理的特点，非常适合实时的插入、更新与查询，并具备实时数据存储所需的复制及高度伸缩性。

3.3.6 使用 MongoDB 存储数据

MongoDB 的设置和连接方法如下：

（1）设置 settings.py 文件

设置 MongoDB 的参数：

```
MONGO_URI = '127.0.0.1:27017' # 服务器地址和端口号
MONGO_DATABASE = 'WeiboTV'  # 数据库名字
```

在 ITEM_PIPELINES 中添加数据库设置：

```
ITEM_PIPELINES = {
    'weibo.pipelines.WeiboPipeline': 300,
}
```

（2）设置 pipeline.py

在 pipelines.py 文件中定义 WeiboPipeline 类对 MongoDB 进行操作，根据 Scrapy 的文档必须定义以下方法：

- from_crawler：这是一个类方法(classmethod)，Scrapy 在创建 Spider 时会调用这个类方法，由于 Spider 被初始化(__init__)之后才有 settings 属性，因此__init__方法无法访问可以 settings 中设置的 MongoDB 参数。在调用该方法时，会将这个类对象（不是类的实例对象）隐式地当作第一个参数传入，这里将类对象 WeiboPipeline 作为参数传入；
- open_spider：用于 Spider 连接数据库；
- close_spider：用于关闭 Spider 与数据库的连接；
- process_item：用于 Spider 对数据库进行操作。

3.3.7 反爬虫技术的应对措施

避免微博网站反爬虫技术识别，本爬虫主要采取以下措施：

（1）设置 Cookies。在 3.3.1 中通过 Selenium 模拟用户进行自动登陆并获取 Cookies，将数据请求时的 Request Cookies 替换，使微博网站将本爬虫识别为正常用户访问；

（2）设置 user_agent 池。服务器能识别爬虫程序默认的 user_agent 从而拒绝爬虫程序访问，通过设置 user_agent 池，每次发送请求时随机选择一个 user_agent 替换默认值，防止被服务器识别；

（3）设置 ROBOTSTXT_OBEY。微博网站通过 robots.txt 拒绝爬虫程序的访问，本爬虫将 settings.py 文件中 ROBOTSTXT_OBEY 设置为 False；

（4）开启自动限速(AutoThrottle)。用户可以手动设置 Scrapy 爬取网站的时间间隔，但是，如果设置的间隔过大，爬取数据的效率会降低，如果设置的时间间隔过小，会增加被服务器拒绝访问的几率。AutoThrottle 使用限速算法，根据爬取的网站的负载，动态调节爬取时间间隔。

3.4 本章小结

本章通过对不同数据获取请求 URL 进行分析，得到了请求各数据时的 URL 格式，根据分析结果，基于 Scrapy 框架构建爬虫系统。

第 4 章 短视频数据处理与分析

4.1 数据简析

本文对爬取的微博短视频数据进行如下操作：

（1）去除重复数据

在爬取数据时，多个爬虫程序同时运行并且独立运行，对同一个短视频 URL 会出现重复抓取的可能，但是在抓取的时间点有先后之分，出现重复时，保留最新的数据。

（2）二次抓取

对于用户评论内容少于 50 条的短视频进行筛选，一段时间之后，再次对这些短视频 URL 进行抓取，这次抓取不进行更多视频 URL 提取。如果该视频的用户评论数量仍少于 50 条，会对下一步中短视频关键词的提取有较大影响，因此将该视频数据移除。

使用 Robomongo 对 MongoDB 进行访问，可视化查看爬虫程序抓取结果，如图 4-1 所示，抓取的数据存储在 WeiboTV 数据库下的 WeiboItem 数据集合中，一个微博短视频对应一个数据文件，微博短视频数据以键-值对形式存储，共抓取 1077 个微博短视频。

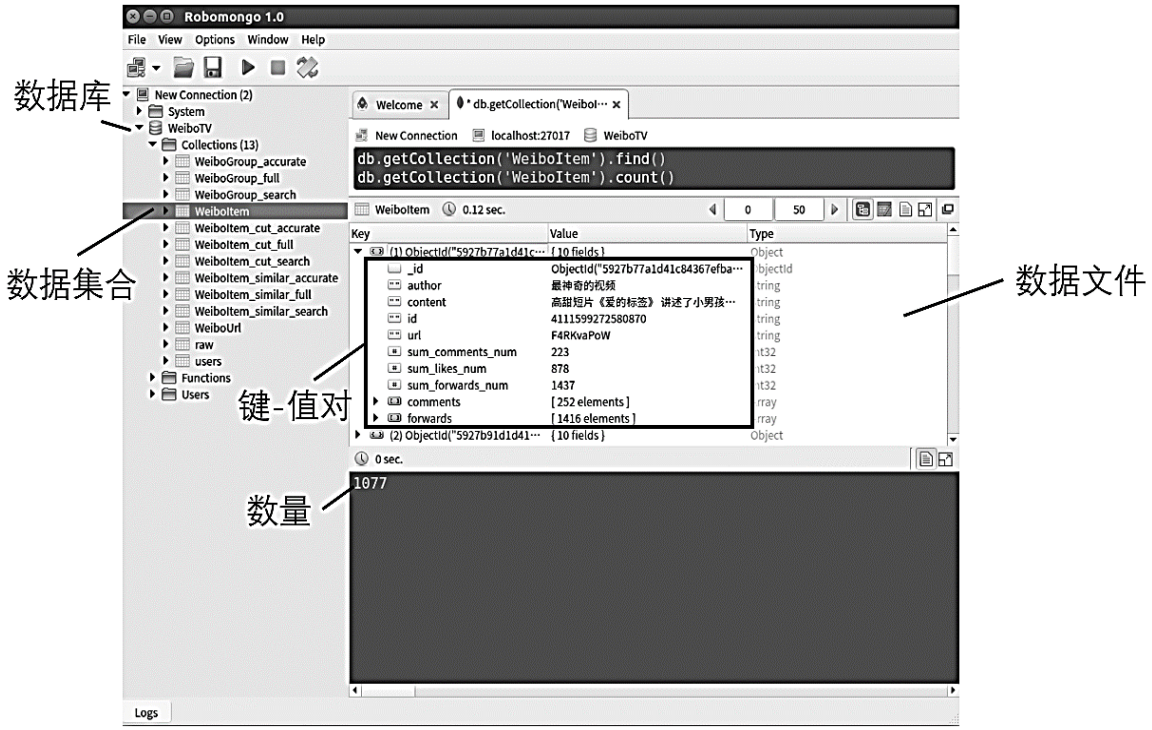

图 4-1 在 Robomongo 展示短视频数据

对爬取的数据进行简要分析，抓取微博短视频总数 1077 个，其中与短视频有交互行为的用户有 1204806 个，用户转发 1628566 次，用户评论 985670 条，平均每个短视频被用户转发 784.1 次，被用户评论 474 次，平均每个用户转发了 1.24 个短视频，评论 0.82 个短视频。

表 4-1 抓取短视频数据数量统计

项目	数量
短视频	1,077
转发数	1,628,566
评论数	985,670
用户数	1,204,806

表 4-2 抓取短视频数据平均值统计

	短视频	用户
转发数	784.10	1.24
评论数	474.56	0.82

4.2 数据处理

由于对短视频的内容进行分类、聚类的代价非常高，应避免直接对短视频处理。考虑到用户在观看短视频后评论成为一种习惯，抓取的短视频下平均每个视频被评论 474.56 次，丰富的用户评论内容提供了一种间接获取短视频内容的方法。

4.2.1 分词算法简介

词是表达语义的最小单位，自然语言处理模型建立在词的基础上。对于西方的拼音语言，词之间有明确的分界符，而对于一些亚洲语言（如汉语、日语、韩语、泰语等），词之间没有明确的分界符，需要现对句子进行分词，才能做进一步的自然语言处理[12] 。

现有的分词算法可分为三大类：基于字符串匹配的分词方法、基于理解的分词方法和基于统计的分词方法。

（1）基于字符串匹配的分词方法，按照一定的策略将待分析的汉字串与一个“充分大的”机器词典中的词条进行匹配，若在词典中找到某个字符串，则匹配成功，识别出一个词。

（2）基于理解的分词方法，通过让计算机模拟人对句子的理解，达到识别词的效果，基本思想是在分词的同时进行句法、语义分析，利用句法信息和语义信息来处理歧义现象。

（3）基于统计的分词方法,在大量已经分词的文本基础上，利用统计机器学习模型学习词语切分的规律，从而实现对未知文本的切分。[13]

随着大规模语料库的建立，统计机器学习方法的研究和发展，基于统计的中文分词方法渐渐成为了主流方法。以统计语言模型为基础的中文分词经过几十年的发展和完善，基本可以看作是

一个已经解决的问题了，不同分词器好坏的差异主要在于统计模型数据的使用和工程实现的精度。

结巴(Jieba)分词是 Python 下中文分词组件，它使用以下算法对中文进行分词：

- 基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图(DAG)；
- 采用了动态规划查找最大概率路径, 找出基于词频的最大切分组合；
- 对于未登录词，采用了基于汉字成词能力的 HMM 模型，使用了 Viterbi 算法。

4.2.2 分词处理

结巴分词支持三种分词模式：精确模式、全模式和搜索引擎模式，其中精确模式试图将句子最精确地切开，适合文本分析，因此使用精确模式对微博短视频下用户评论数据进行分词。

jieba.cut 方法的作用是进行分词，返回的结果是一个可迭代的生成器(generator)，使用以下的代码对用户评论数据进行分词，并且将分词结果以“/”分割存储：

cut_comments = '/'.join(jieba.cut(comments, cut_all=False))

- comments：待分词文本
- cut_all：False 表示使用精确模式

4.2.3 关键词提取

结巴分词提供两种关键词提取的方法：基于 TF-IDF 算法的关键词提取、基于 TextRank 算法的关键词提取。

TF-IDF 算法的主要思想是：如果某个词或短语在一篇文章中出现的频率高，并且在其他文章中很少出现，则认为此词或者短语具有很好的类别区分能力，适合用来分类，但是 IDF 的值需要经过大语料库来统计获取[12]。

TextRank 算法的主要思想是：以固定窗口大小构建词之间的无向带权图，计算图中节点的 PageRank。

TextRank 基于 PageRank 算法，不需要经过语料库来统计获取，因此选用 TextRank 算法对短视频下用户评论进行关键词提取。在提取关键词时，可以通过过滤词性忽略停止词(Stop Word)，去除停止词对结果的影响[14] 。

考虑到实验机器计算能力有限，为了提供下一步相似度计算的效率，对每个视频只抽取不超过 20 个关键词。使用以下的代码对视频关键词进行提取：

tags = jieba.analyse.textrank(cut_a, topK=20, withWeight=True, allowPOS=('ns', 'n', 'vn', 'v'))

- cut_a：已分词文本；
- topK：提取关键词最大个数；
- withWeight：True 表示输出关键词权值；
- allowPOS：词性过滤，只对词性地名(ns)、名词(n)、动名词(vn)、动词(v)进行提取。

将视频关键词提取结果存储，图 4-2 展示了某一个短视频部分关键词及权值

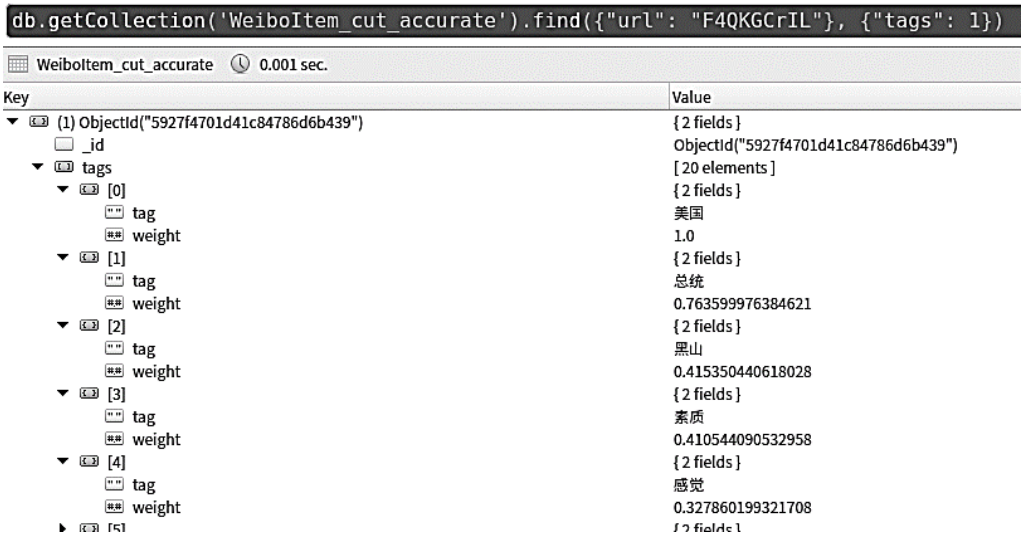

图 4-2 某短视频部分关键词及权值

4.3 本章小结

本章对爬取的微博短视频数据进行简要分析，使用结巴分词对短视频下用户评论内容进行分词，使用 TextRank 算法提取视频的关键词及其权值。

第 5 章 基于相似度的短视频推荐算法

5.1 按相似度分组

5.1.1 相似度计算

上一章使用 TextRank 算法提取了视频的关键词。本文利用视频的关键词，计算视频之间的相似度。

余弦相似度是用向量空间中两个向量夹角的余弦值作为衡量两个个体间差异的大小的度量。向量是多维空间中有方向的线段，如果两个向量的方向一致，即夹角接近零，那么这两个向量就相近。通过使用余弦定理计算向量的夹角，可以确定两个向量方向是否一致[12]。

对任意两个短视频 A、 B，计算他们之间的相似度方法如下：

（1）取出 A、B 的关键词集合 tags_a、tags_b，其中 tags_a 包含 a 个关键词，tags_b 包含 b 个关键词；

（2）将 tags_a 中的 a 个关键词与 tags_b 中的 b 个关键词合并，构成新的关键词集合 tags；

（3）分别统计 tags 中的关键词在 A、B 的用户评论中的相对词频；

（4）根据 A、B 相对词频构造特征向量 α、β，

$$\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_n)$$

$$\beta = (\beta_1, \beta_2, \cdots, \beta_n)$$

（5）计算特征向量 α 和 β 的夹角余弦值，得到 A、B 的相似度：

$$\cos(\theta) = \frac{\alpha \cdot \beta}{\|\alpha\|\|\beta\|}$$

两个短视频之间向量夹角余弦值越大，表示两个短视频越相关；夹角的余弦值越小，表示两个短视频越不相关。

5.1.2 相似度网络构建

上一节根据相似度计算方法计算了短视频之间的相似度值，但不是所有短视频之间相似度计算结果均是有效的，需要通过确定合适的阈值，过滤掉一些无效的弱连接。

根据短视频之间的相似度，以短视频为的节点，如果两个短视频之间的相似度大于所选阈值，则在对应的两个节点间添加一条边，相似度作为边的权值，构造一个无向带权图表示短视频相似关系。

5.1.3 网络社区发现与社区分类

使用 Fast Unfolding 算法对构建的短视频相似度网络进行聚类，以甄别短视频的主要类别。Fast Unfolding 算法[15] 是一个基于模块化值(modularity)优化的启发式算法，其聚类结果能很好的将相似度高的节点聚类到同一个社区，具有很好的聚类质量。

使用 Fast Unfolding 算法对构建的短视频相似度网络进行聚类，如表 5-1 所示，相似度阈值大于 0.5 模块化值变化不大。

表 5-1 相似度网络统计特征

相似度阈值	模块化值	平均度	平均路径长度	平均聚类系数
0.3	0.505	42.642	2.819	0.567
0.4	0.587	22.366	3.704	0.645
0.5	0.627	14.507	4.508	0.688
0.6	0.632	9.66	4.536	0.749
0.7	0.636	7.027	4.329	0.764

选择相似度阈值 0.5 对短视频相似度网络进行聚类，聚类结果如图 5-1 示

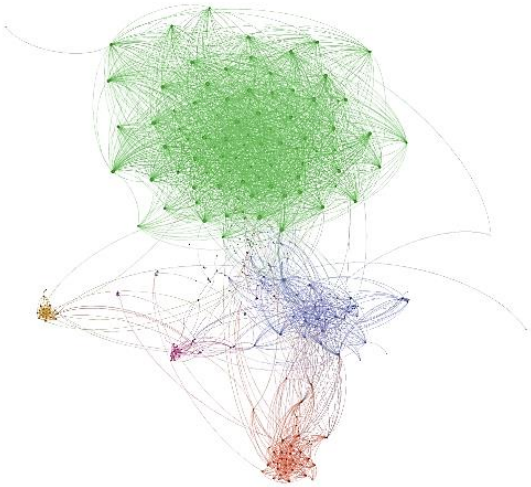

图 5-1 视频相似度网络聚类结果

5.2 按用户兴趣推荐短视频

要提取用户的兴趣，就要先获取用户的短视频观看记录，分析用户对哪些视频感兴趣，但用户的观看记录数据无法获取。

由于微博具有公开的特点，可以获得大量用户与视频的交互行为数据。根据图 5-2 调查结果显示[2]，83.2%的用户因为视频有趣分享视频。在某种程度上可以认为，一个用户转发了某个短视频，说明这个用户认为该短视频有趣。本文通过分析用户转发过的短视频，根据短视频相似度网

络聚类结果，提取用户兴趣视频分类。

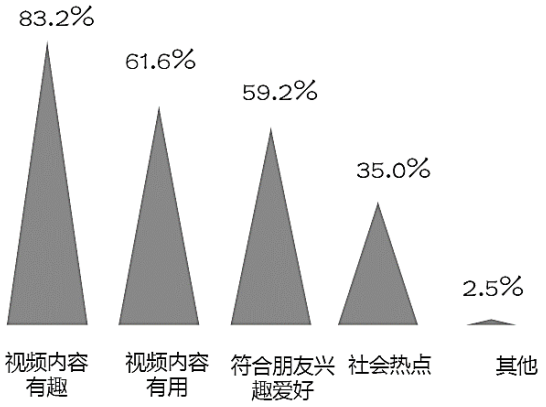

图 5-2 用户分享视频原因调查结果

根据用户兴趣视频分类，对分类下的短视频按相似度进行排序，将相似度高的短视频推荐给用户，达到推荐的目的。

5.3 推荐算法测试与评价

5.3.1 测试用户选取

微博短视频爬虫共抓取 1077 个短视频数据，与短视频进行交互的用户有 1204806 人，用户转发 1628566 次，人均转发 1.24 次。如图 5-3，根据用户转发视频个数进行统计，随着转发视频个数的增加，用户数量明显下降。本文选择转发短视频数量大于 9 个的用户，作为测试用户进行推荐算法测试。

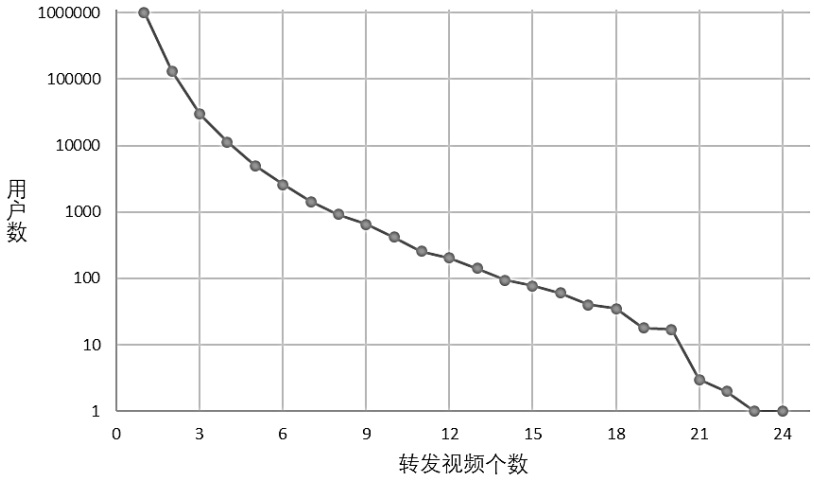

图 5-3 用户转发视频个数统计图

5.3.2 测试方法

由于抓取的数据有限，按照下列步骤对推荐算法进行测试：

（1）从测试用户转发过的短视频中，随机抽取一个短视频；

（2）根据短视频相似度分组结果，找到该短视频所属分类；

（3）对该分类下的短视频按相似度进行排序；

（4）根据排序结果，选择不超过 20 个相似度较高的短视频，作为向用户推荐的短视频；

（5）验证推荐的短视频是否存在于用户转发过的视频中。

5.3.3 评价指标

准确率 P(Precision)和召回率 R(Recall)是广泛用于信息检索和统计学分类领域的两个度量值，用来评价结果的质量，其中准确率是检索出相关文档数与检索出的文档总数的比率，召回率是指检索出的相关文档数和文档库中所有的相关文档数的比率。

在短视频推荐算法进行评价中，准确率 P 和召回率 R 的定义如下

$$\text{准确率 P} = \frac{\text{推荐视频中在用户转发视频中个数}}{\text{推荐视频个数}}$$

$$\text{召回率 R} = \frac{\text{推荐视频中在用户转发视频中个数}}{\text{用户转发视频个数}}$$

当准确率 P 和召回率 R 指标出现矛盾的情况时，需要综合考虑他们，最常见的方法就是 F-Measure，F-Measure 是准确率 P 和召回率 R 的加权调和平均

$$F = \frac{(a^2+1)P \cdot R}{a^2(P+R)}$$

当参数 α=1 时，就是最常见的 F1-Measure

$$F1 = \frac{2PR}{P+R}$$

可知 F1 综合了准确率 P 和召回率 R 的结果，当 F1 较高时，说明推荐方法比较有效[16]。

5.3.4 测试结果

在测试过程中，每次测试随机抽取 30 个用户进行推荐测试，总共进行了 1196 次测试，将推荐算法与热门推荐、随机推荐进行对比，结果如表 5-2，可以看出，推荐算法在准确率 P、召回率 R、F1-Measure 的结果上都比热门推荐、随机推荐好。

表 5-2 对比测试结果

	准确率 P	召回率 R	F1-Measure
推荐算法	0.096	0.066	0.043
热门推荐	0.032	0.052	0.039
随机推荐	0.011	0.017	0.013

5.4 本章小结

本章实现了基于相似度的短视频推荐算法的设计，并根据抓取的短视频数据，对短视频推荐算法进行了测试与评价。

第 6 章 总结与展望

6.1 总结

本文以社交网络平台微博为研究对象，通过对短视频数据的分析，设计了基于相似度的短视频推荐算法，并对算法进行了对比测试。

第 1 章对课题开展的背景、研究方法和需要解决的问题进行了描述。

第 2 章对所用的开发技术进行简单介绍。

第 3 章为所需的短视频数据获取工作，基于微博平台，通过分析微博短视频网页结构，构建基于 Scrapy 的微博短视频数据爬虫，对微博短视频数据进行抓取。

第 4 章对抓取的微博短视频数据进行初步的分析和处理，使用结巴分词组件对短视频进行关键词提取。

第 5 章设计了基于相似度的短视频推荐算法，通过构建短视频相似度网络对短视频进行分类，分析用户行为提取用户兴趣进行短视频推荐，并对推荐算法进行了对比测试与评价。

6.2 展望

基于相似度的短视频推荐算法在测试结果上优于其他推荐方法，这为微博用户兴趣视频推荐提供一种可行的方法，在后续工作中可对算法进行如下改良：

（1）在短视频数据获取阶段，抓取更多的短视频数据用于推荐；

（2）基于社交网络中的好友具有相似兴趣的观点，获取用户的社交网络关系，结合用户和他的好友的兴趣进行推荐；

（3）用调整余弦相似度算法(Adjusted Cosine Similarity)计算短视频相似度。余弦相似度对数值的不敏感可能会导致结果的误差，需要修正这种不合理性，使结果更加符合现实。

参考文献

[1] 第 39 次中国互联网络发展状况统计报告[R]. 中国互联网络信息中心(CNNIC), 2017.

[2] 新浪微博数据中心. 2016 微博短视频行业报告[EB/OL]. http://data.weibo.com/report/reportDetail?id=342, 2016-12-19/2017-06-11.

[3] 王蕾. 互联网信息消费碎片化特征及文化意义[D]. 山东师范大学, 2014.

[4] 李鹏, 于晓洋, 孙渤禹. 基于用户群组行为分析的视频推荐方法研究[J]. 电子与信息学报, 2014, 36(6):1485-1491.

[5] Dias A S, Roesler V, Wives L K. POI Enhanced Video Recommender System Using Collaboration and Social Networks[C]//International Conference on Web Information Systems and Technologies. 2012.

[6] Su C R, Li Y W, Zhang R Z, et al. An Adaptive Video Program Recommender Based on Group User Profiles[M]//Advances in Intelligent Systems and Applications - Volume 2. Springer Berlin Heidelberg, 2013. Niu Jian-wei, Zhao Xiao-ke, Zhu Li-ke, et al.. Affivir: an affect-based internet video recommendation system[J]. Neurocomputing, 2013, 120.

[7] Zhao S, Yao H, Sun X. Video classification and recommendation based on affective analysis of viewers[J]. Neurocomputing, 2013, 119(16):101-110.

[8] Öztürk G, Cicekli N K. A Hybrid Video Recommendation System Using a Graph-Based Algorithm[C]//International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems Conference on Modern Approaches in Applied Intelligence. Springer-Verlag, 2011:406-415.

[9] 2016 互联网文化娱乐产业洞察[EB/OL]. http://cf.dtcj.com/2016 互联网文化娱乐产业洞察_1480756778405_5r3vjqx5sr0c3ow29.pdf,2016-09-10/2017-06-1.

[10] Scrapy developers. Architecture overview[Z]. https://doc.scrapy.org/en/latest/topics/architecture.html.

[11] 吴军. 数学之美[J]. 2012.

[12] 龙树全, 赵正文, 唐华. 中文分词算法概述[J]. 电脑知识与技术, 2009, 5(4):2605-2607.

[13] Mihalcea R, Tarau P. TextRank: Bringing Order into Texts[J]. Unt Scholarly Works, 2004:404-411.

[14] Meo P D, Ferrara E, Fiumara G, et al. Fast unfolding of communities in large networks[J]. Journal of Statistical Mechanics Theory & Experiment, 2008, 2008(10):155-168.

[15] 王春才, 邢晖, 李英韬. 推荐系统评测方法和指标分析[J]. 信息技术与标准化, 2015(7).

致 谢

谨向所有给予我指导和帮助的老师、同学、朋友致以深深的谢意。

2017 年 6 月

李观波