关键词：微博；推荐系统；社会标签；Android

**引言**

**课题背景**

**新型信息平台—— twitter、微博**

     2006年，twitter诞生于美国旧金山，它允许用户将自己的最新动态、想法、分享内容写在最多140字的微型博文中，通往网页、短信、移动客户端等方式发布出去。款基于web2.0技术的新型信息互动平台，有着信息灵活短小即时性强的特点，受到了广大网民的追捧。从普通人到明星政要，都感受并运用着twitter带来的便捷快速信息传递的威力，它甚至成为美国总统大选的助选工具。热点突发事件来袭经常也伴随着twitter的信息暴涨，同时这一平台上活跃的信息传播也推动着现实世界的发展。

     作为twitter的后继模仿者，新浪微博推出于2009年8月,它虽然继承了twitter的140字规则和大体框架，但是，在细节甚至基本理念上，还是有着诸多不同的：首先，twitter的转发不附带任何信息，而微博转发鼓励用户在原微博基础上进行评论或延伸；其次，微博中的转发和评论均带有原信息以及上下文，便于用户参与讨论，而twitter使用户更专注于一条信息链；再者，twitter不像微博一样，直接程序图片、视频等富媒体信息

     不难看出，微博表现出更强的社交属性，而twitter更多专注于内容本身，一个在“丰富”的道路上越走越远，另一个执着于简洁的体验。虽然新浪微博走着与twitter相背离的道路，有时被诟病为不敬业的抄袭者，但通过对社交性和国内用户使用习惯的准确把握，到如今已是国内用户数最大的微博产品，截止2012年底，其注册人数已超过五亿，每日微博产生量超过一亿条。

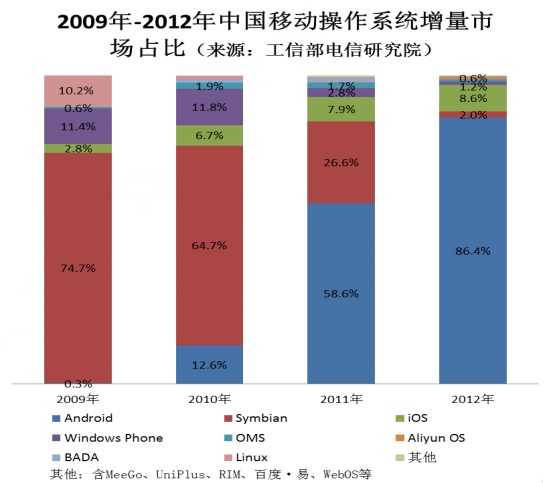
**移动互联网与Android的兴起**

2011年可以说是移动互联网元年，智能手机开始逐步占领手机市场，各种社交网络和新型信息平台的流行也加剧了人们对智能手机的依赖。手机厂商们纷纷在新推出的机型中集成各种SNS服务客户端，优秀的第三方开发者也陆续出现，与官方和大厂商们的移动应用争抢市场份额。

     2013年初，一家名为BusinessInsider的科技网站的市场研究团队BI Intelligence发布了名为“移动未来”的系列市场数据统计图表。如今，移动互联网占全球网络流量的比例已达到15%，随着智能手机和平板设备出货量的持续增加，这一比例还在继续上升中。

<图：移动互联网占全球网络流量的15%>

     从国内来看，工信部2013年最新数据显示，今年一季度，移动互联网接入流量达到2.7亿G，增幅达56.5%，已经连续3个月增长率超过50%。从工信部电信研究院发布的移动互联网白皮书中，我们可以看出，Android系统在中国的移动互联网中的占有率连续四年快速增加，截止2012年底，其份额已经达到惊人的86.4%。（注，<http://www.catr.cn/kxyj/qwfb/bps/201303/P020130301397809834073.pdf> ）



     同时在2012年5月，新浪在第四届GMIC全球移动互联网大会上分享的运营数据表示，活跃用户中有60%通过移动终端登录，其中来自Android系统的流量占据了超过四分之一的比例。（注[http://mobi.adsage.cn/news/news/338](http://mobi.adsage.cn/news/news/338%C2%A0&#65289)）

由以上信息我们可以看出，移动互联网是信息是信息时代的未来，而Android还将在其中承担重要的角色。

**推荐系统研究与应用的发展**

     twitter和新浪微博这类新型信息平台崛起，和移动互联网流量持续上升的事实从侧面反映出了一个趋势，随着互联网和信息技术的发展，人们逐渐步入了是信息过载（Information Overload）的时代，信息过载指的是在信息爆炸的环境下信息的利用率反而大大降低的状况。在这样一个时代，无论是信息消费者还是信息生产者都遇到了很大的困扰：对于信息消费者来说，在海量信息中找到自己感兴趣的信息是一件非常困难的事情，如同沙里淘金；对于信息生产者，如何让自己生产的信息脱颖而出，受到广大用户的关注，也同样是一件无比困难的事情。

     面对这一困境，许多科学家和工程师们提出了大量出色的解决方案，其中代表性的解决方案有分类目录和搜索引擎。然而这两种方案都需要用户对自己的需求有较为清晰的认识，而当用户无法准确提供描述自己需求的关键词时，这两种方案就作用有限了。

     于是另一种技术孕育而生，推荐系统。推荐系统也是一种帮助用户快速发现有用信息的工具，不同的是，它不需要用户给出明确的需求，而是通过分析用户的信息历史行为，主动给用户推荐可能满足其兴趣和需求的的信息。推荐和个性化技术，作为解决信息爆炸问题的一个方法，取得了不错的效果。在互联网行业内的各个公司的产品中应用也越来越多，这反过来也促进了大家对推荐的热情 。抛开需求的角度来说，推荐是系统和人的互动，推荐首先需要更好地理解人，理解用户。就百度前技术委员会主席廖若雪看来，这是一个互联网更加智能化的发展方向，系统的智慧会越来越高级，这本身是非常有吸引力的方向。

     我们可以看到，推荐系统已经无所不在，尤其是随着大数据时代的到来，系统掌握了越来越多、越来越精细的用户数据，通过语义分析、协同过滤、聚类等算法让系统能够读懂用户，发掘他们的潜在需求，例如在亚马逊购物时的推荐购买、豆瓣猜推荐的书籍和电影、百度知道的问题推荐等，这些都极大地改善了用户在网站上的使用体验。

**研究的意义与目标**

将目光从信息时代的大局转移锁定到身边的具体实例——新浪微博上，我们不难观察到，它集中体现着信息时代的繁荣与嘈杂，有许多有待解决的问题。

根据新浪微博的自我定义，它是一个基于用户关系的信息分享、传播以及获取信息的平台。TODO其核心价值之一便在于海量的信息。（注<http://hr.weibo.com/jobs/about.php> ）然而作为一个社交网站，twitter和微博都是以“人”而非“信息”为基本聚合单位的：用户注册一个账号，加入微博的网络成为一个节点产生信息，同时关注其他账号，收听其发布的信息。这样的设计强调了人与人之间的联系，体现其社交纽带的意义。

一家来自圣安东尼奥的市场研究公司 Pear Analytics 在2009年8月连续两周分析了每天11点到17点2000条英文twitter消息，对这些消息进行归类，得出了以下6种大致分类：(注<http://zh.wikipedia.org/zh/Twitter>  )

无目的的闲谈 – 40%，对话 – 38%，传递价值 – 9%，自我推广 – 6%，垃圾信息 – 4%，新闻 – 4%

社交网络研究者 Danah Boyd 随后对这一研究结果展开了讨论。他认为“无目的的闲谈”更应该是一种人类维系社交关系的行为，人们会想知道自己身边的人在做什么。然而在这个庞大的关注网络中，信息并不只是向着有效受众传递，人们也并不是总是希望接受到这样的信息。

而且，虽然在设计上每个用户都有平等的发布信息的权利，但伴随着平台中名人效应的影响和营销账号的盛行，微博已渐渐变成了小部分创作者发声的平台，大多数用户只是这些内容的消费者。

在这样以内容消费为重点的的使用情景下，以人为聚合单位的设计，就显现出其不便之处了：某些运营账号发布的微博主题或许较为明确，而大多数用户产生的信息内容繁杂，有效信息和垃圾信息难以区分，使接受方获取有效信息的成本加大。许多时候，一个普通用户在这样的社交-信息平台上耗费大量的精力在琐碎无关的内容上，而真正感兴趣的信息则被淹没其中。

新浪微博推出了一些功能来应对上述问题：

     1）“微话题”：通过将发表的微博加上标签，来对同一主题的内容进行分类聚合，然而这一机制并不能很好的解决上述问题：首先，小而杂；其次，话题标签需要主动设置，不符合大多数用户的使用习惯；更重要的是，关注话题的相关微博并不直接显示在用户的时间线上，需要通过主动搜索或是话题页面推荐，。

     2）关注分组：将关注的用户分成几个组别，可分别查看不同分组的微博。这可能是一种比微话题更直接有效信息整理的方式。但分组的有效性依赖于用户自己的持续维护，增加了一定的操作成本，如果用户没有将关注对象放在合适的分组，分组的作用就大大减弱了；同时这样的分组依然针对于”人“而不是信息，在维护社交关系上分组是实用的，而对于内容来说，在同一个分组中一样会出现很多的信息杂讯。

     于是第三方的开发者也提供了他们自己的解决方案：各式各样的过滤插件，如“微博控“等。这类插件的共同特征是根据一定的关键词、特征设定白名单黑名单，过滤掉用户不希望看到的微博。这是一个容易实现且较有效率的解决方案，可以清理掉大部分明显的垃圾信息。新浪微博官方只对付费会员用户提供这一功能。然而这一方案也有它的局限，识别精度提升困难，若为了提升识别精度而严格设置过滤条件，则误判率显著上升。同时它们仅专注于信息的过滤，并没有尝试进一步优化用户的微博信息流，即进行信息的推荐。

//TODO

     因此，本项目从信息接收者的角度出发，分析新浪微博使用体验上缺陷，并参考了现有的解决方案，设计微博分类推荐算法，并实现一个Android微博过滤推荐应用。采用关键词特征过滤方式结合统计学的方法，尝试优化用户接收到的信息流，提升用户体验。

**文章结构**

本章节回顾了互联网时代新型信息平台的发展和移动互联网的兴起，认识了信息过载的现状及解决方案——推荐系统。同时，从能反映了问题的具体实例新浪微博入手，引出了本研究课题的合理用户需求与希望尝试的解决方案。本文的其余章节分为如下几个部分：

     第二章，理论概念与技术基础，简单介绍了项目设计到的新浪微博开放平台、推荐系统算法、社会化标签以及用于实现的Android平台。

     第三章，需求分析与系统架构，对第一章提出的需求进行分析，提出系统的功能需求以及系统架构。

     第四章，系统核心功能模块实现，阐述项目中对微博信息获取、推荐模块、用户反馈模块等具体功能的实现要点。

     第五章，总结与展望。回顾相关研究,总结研究中的发现和成果,并对下一步研究和项目开发进行展望。

**课题相关技术及原理**

**个性化推荐系统**

在上文中已经讨论过，个性化推荐系统是为解决信息过载问题发展出来的工具，其本质在于代替用户评价尚未接收到的信息，减少用户选择、处理信息的成本。

     一个完整的推荐系统通常包含一下三个模块：

     行为记录模块，用于收集用户信息以及用户行为；

     模型分析模块，利用行为记录模块收集的信息构建分析用户需求与喜好；

     推荐算法模块，利用用户的需求和喜好模型筛选出较为符合用户需求的信息或产品。

     其中，推荐算法是个性化推荐系统的核心。

对推荐算法的讨论和研究始于，时至今日已有不少成熟的理论和实践。以下是两种已被广泛使用的推荐算法。

**协同过滤算法**（collaborative filtering）是推荐系统中非常热门的一个实现技术，它分析用户的兴趣模型，然后从用户群中寻找兴趣模型相似用户，从相似用户那里找到指定用户可能感兴趣的内容或物品。它的优点在于可以对音乐、电影等难以进行结构化表示和自动内容分析的的对象进行推荐。该算法通常需要建立一个用户-对象评分矩阵，当该矩阵非常稀疏时，算法很难找到相似用户或对象，使协同过滤无法很好的发挥作用。

**基于内容的推荐**（content-based recommendation）最初是协同过滤技术的延伸与发展而来，它并不需要过多依赖用户对对象的评价，而是根据用户已经选择对象的内容信息，计算与其他对象之间的相似度，进而推荐相似度大于一定阈值的对象。由于这种算法不受评价矩阵数据稀疏性的影响，能够对新产品、对象进行评价和推荐。这一技术经常被运用于基于文本的推荐。

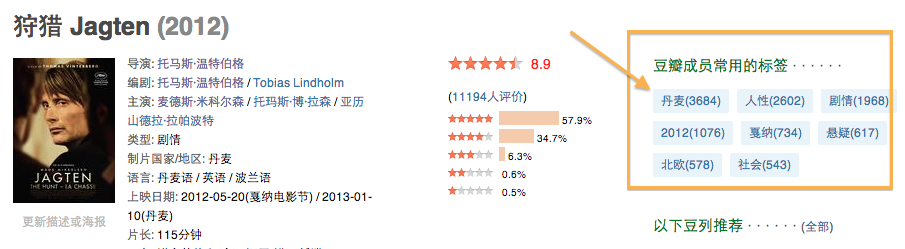
除此之外还有许多新兴推荐算法正在研究完善和或已投入使用，如各种混合推荐算法、基于规则关联的推荐和基于网络结构的推荐算法等等。完成一个推荐系统需要根据应用场景和需求，结合实际选择适当的推荐算法。

//Move to implementation segment?

    当考虑将协同过滤算法运用在微博过滤推荐的实现中，我们会发现不少问题。首先，微博中并不像亚马逊商品、豆瓣图书或电影有打分评价功能，缺少能直接获取用户对一条微博反馈的渠道。其次，虽然用户对一条微博的转发或是收藏可以看做是对其的正反馈，但新浪微博开放平台API不允许请求未授权用户的收藏，转发信息，于是难以计算用户的相似度。所以在构建用户对，协同过滤算法应用起来较为困难。从需求上来说，本项目希望建立一个针对微博内容的过滤推荐系统，以文本信息的评价处理为主，因而，基于内容的推荐是较为合适的选择。

**社会化标签**

     社会化标签通常是一组短小简洁的描述，由用户自行标注在文档、影片、音乐等等各种资源甚至用户资料中，反映了用户对特定对象的认知和评价。标签的内容、个数和一致性没有限制，是一种新型的索引方式。作为由用户产生的元数据，社会化标签有效地反映用户的兴趣、需求及其变化，因此越来越多地受到推荐系统研究者的关注。



**//TODO**

**新浪微博开放平台**

**简介**

     新浪微博开放平台是一个基于新浪微博客系统的开放的信息订阅、分享与交流平台。它得力于微博强大的传播能力、海量用户，为各种第三方开发者提供了广阔的展现平台与发展空间，促进有着高度社交特性的应用的构建。这一平台上目前已经有超过20万款应用，其中约2万款为受到用户的青睐的活跃应用，应用的数量还在持续迅猛增长中。

     目前在新浪微博经过多次的安全性提升之后,现在对微博的用户数据进行爬虫式的挖掘已经变得相当困难,如果需要对用户数据进行分析挖掘就必须要使用新浪开放平台的 V2 接口。对于本项目来说，使用这一平台最大便利之处在于，该平台开放提供了获取用户信息、用户关系以及微博列表等关键信息的API，大大减少了数据获取的成本。通过接入新浪微博开放平台，开发者便可以利用微博生态圈各种资源，同时参与它的变化与发展。

**应用创建管理与权限获取**

     微博账号登陆http://open.weibo.com后，进入应用开发版块可创建应用，可选择移动应用、站内应用、网页应用、浏览器插件等多种类型。为了实现在Andorid平台上的微博推荐应用，需要创建一个移动应用用于访问平台资源。

     创建应用完成后，可以在“我的应用 – 应用信息”中查所创建应用的AppKey及AppSecret，这些信息是调用新浪微博开放平台各API的身份标志。

     在“我的应用 – 数据统计”页面中，开发者可以查看应用的相关统计数据，包括应用统计、接口统计、用户特征统计等信息。

     当应用开发完成后，开发者需在“我的应用”页面提交审核。只有应用通过审核后，才能在微博中正确显示来源显示等，同时提高应用的API接口访问限制。

**基于API的数据获取**

     在开发者成功申请到AppKey和AppSecret之后，就可以调用新浪微博开放平台上的一系列接口了。其中包括微博接口、评论接口、用户接口、关系接口等，涵盖了新浪微博的大部分常用功能。

     新浪微博开放平台使用的是标准的REST API，开发者只需要通过 HTTP Get/Post 请求就可以方便地使用所有的接口功能，均返回json对象作为结果，与语言无关。以获取当前登录用户及其所关注用户的最新微博为例：

请求header为 HTTP GET https://api.weibo.com/2/statuses/friends\_timeline.json

接口参数作为http请求payload

|  |  |  |  |
| --- | --- | --- | --- |
| **参数** | **必选** | **类型及范围** | **说明** |
| source | false | string | 采用OAuth授权方式不需要此参数，其他授权方式为必填参数，数值为应用的AppKey。 |
| access\_token | false | string | 采用OAuth授权方式为必填参数，其他授权方式不需要此参数，OAuth授权后获得。 |
| since\_id | false | int64 | 若指定此参数，则返回ID比since\_id大的微博（即比since\_id时间晚的微博），默认为0。 |
| max\_id | false | int64 | 若指定此参数，则返回ID小于或等于max\_id的微博，默认为0。 |
| count | false | int | 单页返回的记录条数，最大不超过100，默认为20。 |
| page | false | int | 返回结果的页码，默认为1。 |
| base\_app | false | int | 是否只获取当前应用的数据。0为否（所有数据），1为是（仅当前应用），默认为0。 |
| feature | false | int | 过滤类型ID，0：全部、1：原创、2：图片、3：视频、4：音乐，默认为0。 |
| trim\_user | false | int | 返回值中user字段开关，0：返回完整user字段、1：user字段仅返回user\_id，默认为0。 |

返回值示例

{

     "statuses": [

          {

               "created\_at": "Wed May 15 21:03:34 +0800 2013",

               "id": 3578326646227319,

               "mid": "3578326646227319",

               "idstr": "3578326646227319",

               "text": "[pdf]《现代派小说技巧初探》 作者：敏感词 下载地址：http://t.cn/zTeDl8U",

               "source": "<a href=\"[http://weibo.com/\](http://weibo.com/%5C)" rel=\"nofollow\">新浪微博</a>",

               "favorited": false,

               "truncated": false,

               "in\_reply\_to\_status\_id": "",

               "in\_reply\_to\_user\_id": "",

               "in\_reply\_to\_screen\_name": "",

               "pic\_urls": [],

               "geo": null,

               "user": {

                    "id": 3362683514,

                    "idstr": "3362683514",

                    "screen\_name": "没我找不到的电子书",

                    "name": "没我找不到的电子书",

                    "province": "50",

                    "city": "6",

                    "location": "重庆 沙坪坝区",

                    "description": "没我找不到的电子书，请需要寻书的朋友在关注我之后，发书名和需要的格式，并@，我将会为亲肝脑涂地。",

                    "url": "",

                    "profile\_image\_url": "<http://tp3.sinaimg.cn/3362683514/50/40023087591/1>",

                    "profile\_url": "qiushu1",

                    "domain": "qiushu1",

                    "weihao": "",

                    "gender": "m",

                    "followers\_count": 9289,

                    "friends\_count": 13,

                    "statuses\_count": 857,

                    "favourites\_count": 0,

                    "created\_at": "Wed Apr 24 13:00:11 +0800 2013",

                    "following": true,

                    "allow\_all\_act\_msg": false,

                    "geo\_enabled": true,

                    "verified": false,

                    "verified\_type": -1,

                    "remark": "",

                    "allow\_all\_comment": true,

                    "avatar\_large": "<http://tp3.sinaimg.cn/3362683514/180/40023087591/1>",

                    "verified\_reason": "",

                    "follow\_me": false,

                    "online\_status": 1,

                    "bi\_followers\_count": 1,

                    "lang": "zh-cn",

                    "star": 0,

                    "mbtype": 11,

                    "mbrank": 1,

                    "block\_word": 0

               },

               "retweeted\_status": {

                ...

               },

               "reposts\_count": 0,

               "comments\_count": 0,

               "attitudes\_count": 0,

               "mlevel": 0,

               "visible": {

                    "type": 0,

                    "list\_id": 0

               }

          },

...

],

"previous\_cursor": 0, // 暂时不支持

"next\_cursor": 11488013766, // 暂时不支持

"total\_number": 81655

同时，平台还提供了多种语言和平台的官方、非官方SDK，极大地方便了应用开发者。本项目采用官方提供的Android SDK访问各种微博API。

新浪微博根据开发者的需求不同，开放了不同的API访问次数权限。例如对于开发阶段的，未通过审核的测试应用，针对一个服务器IP的请求次数限制为每小时1000次；针对一个用户在使用应用的请求次数限制为每小时150次；针对不同类型的接口还有更为具体的限制。（注<http://open.weibo.com/wiki/Rate-limiting> ）申请更高级接口需要并通过审核，并且达到一定的用户人数。

**OAuth2.0认证技术**

     OAuth（开放授权）是一个开放标准，允许用户使用一个Token，而不是用户名和密码来让第三方应用在特定时段内访问该用户在某一网站上存储的特定的私密资源（如照片，视频，联系人列表），而无需将用户名和密码提供给第三方应用。

     OAuth2.0是OAuth协议的第二代，但与OAuth1.0不兼容。它简化了OAuth的认证流程，减轻了开发者的工作量；同时提供多种不同的认证流程以满足网页应用、桌面应用、移动应用和其他智能设备不同认证需求；Oauth2.0删除了繁琐的加密算法，利用了https传输对认证的安全性进行了保证。OAuth2.0 很可能成为下一代的“用户验证和授权”标准，该协议已经被众多国内外知名网站如facebook，twitter和百度、腾讯等的开放平台采用作为标准用户认证机制。

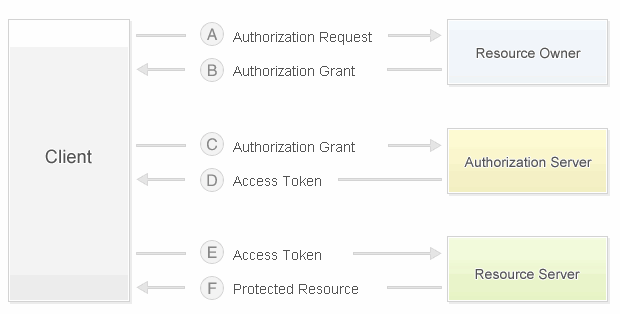
     对新浪微博开放平台上大部分API的访问，如获取、发表微博、添加关注等都需要用户身份信息，目前该平台也采用OAuth2.0技术技术进行应用的用户授权认证。对于站外网页应用或客户端应用，出于安全性考虑，需要在“我的应用 – 应用信息 – 高级信息”中填写redirect\_url（授权回调页），才能使用OAuth2.0。用户同意授权后，会重定向到回调面，在该页面上，可获取AccessToken，作为调用API的必要参数。

     新浪微博开发平台中的OAuth2.0验证流程如下：

     首先，第三方应用向用户申请授权，用户完成授权后返回AuthorizationCode给应用。

     然后，应用将AuthorizationCode发给验证服务器，换回AccessToken.

     最后，将AccessToken加入API请求参数，从服务器获取受保护的资源。



     AccessToken有一定的生命周期，用户授权时，oauth2/access\_token接口返回的expires\_in值就是其生命周期。如果用户的access\_token失效，则调用需要用户认证的API时会返回错误信息，需要引导用户重新授权。

**Android系统与应用**

**//todo**

Android是Google推出的一个基于Linux的操作系统，同时也可以理解为以Android系统为核心的平台，具有高度的开放性。Android平台提供为第三方开发者提供了一个宽松、自由的环境，从而能产生出更多高质量的应用。

**Android系统结构**

     Android系统分为四个层次，由上层到底层分别是分别是应用程序层、应用程序框架层、系统运行时和核心类库层、Linux 核心层。

     应用程序层是直接面向用户的层次，包含在设备上运行的所有系统应用和第三方应用。

     框架层由多个系统服务组成，应用层中的运用的各种操作，如窗口构造，用户交互事件处理，获取设备信息等，都是在各种系统服务的支持下实现的。

     Android运行时由java核心类库和Dalvik虚拟机组成,核心类库包括了Android框架层和应用层所需的各种java基础库；而Dalvik为是Android定制的java虚拟机，负责动态解释执行程序、分配内存空间还有管理对象的生命周期。

     Linxu核心层包含Android系统底层的Linux核心，以及各种硬件驱动。

**Android组件模型**

     Android应用采用基于组件的应用设计模式：应用由众多组件搭建而成，并通过配置文件设置和描述组件功能。Android的四大基本组件构成了各式各样的应用：

     界面组件Activity：构造与用户进行交互的界面的组件。Android中的每个任务（应用）都会维护一个Activity栈，栈中的元素是Activity对象的实例，前台任务的栈顶元素既是当前与用户交互的Activity。同时，Android的每个应用进程都有一个Application Context,可以通过其在相同任务的不同Activity之间传递小数据量的共享数据。

     服务组件Service：在后台帮助应用与系统中的其他组件或服务沟通的组件，扮演着功能调度者和提供者的角色。

     触发器组件Broadcast Receiver：用于监听系统或系统中其他应用发出的事件，做出适当响应的组件。在Android系统中，当特定事件发生时，由组件管理服务根据配置信息通知对应的Broadcast Receiver组件，并构造执行组件的进程。这样便无需在后台保持一个应用程序进程用于事件监听。

     数据源组件Content Provider: Android中每个应用的数据都是私有的，其他应用没有访问权限。而Content Provider组件则实现了数据访问接口，使应用程序可以将指定数据集提供给其他应用程序使用。对Content Provider的访问是通过URI进行定位的。

     以上除了Contend Provider的三个组件，进行连接通信时都依赖于Android中的Intent机制。该机制引入了组件管理服务作为连接组件的管理调度者，它从调用组件中接受Intent对象，然后将其与其他组件的Intent Filter对象进行比较，从中选出符合调用组件需求的实现组件，最后构造并调用该组件，从而完成了组件间的协作。

**应用数据存储**

     数据和程序是构成应用的两个核心要素，数据存储是应用开发中最重要的主题之一，尤其对于一个致力于数据优化处理的应用。Android的数据存储是基于Linux文件系统构建的，充分利用了Linux的账号权限系统来限制对数据的访问，实现了一套安全与灵活并重的数据存储解决方案。

     在Android中，所有第三方应用存放在/data/app目录下，而它们对应的数据，则存放与/data/data目录下与安装包同名的应用数据目录中。除了文件存储外，Android还提供了Shared Preference、数据库存储已经云存储等多种数据存储解决方案。在这里简单介绍项目中使用到的Shared Preference和数据存储两种方式。

     Shared Preference以键值对的形式存储一些基本的类型数据，以XML的格式存放在应用数据目录的shared\_prefs子目录下，由此可知Shared Preference无法存储二进制数据。为了提升它的读写效率，所有关于Shared Preference的磁盘操作，包括加载和写入，会在独立的线程上进行，这样避免了阻塞主线程。而且它的所有读写操作都是线程安全的，可以在不同的线程上进行操作。因此，除了通常的设置信息外，本项目中将访问API需要的AccessToken也存放于SharedPreference中，以提高程序的效率。

     Android中的数据库实现，依赖于如今十分流行的开源嵌入式数据库SQLite，应用的数据库文件位于应用数据目录下的database子目录中。从代码结构上看，Android的数据库实现分为两个层次，底层通过C++调用Sqlite接口来执行AQL语句，并通过JNI暴露java的访问接口；在框架层中，Android提供了相关java类库，用于数据库操作，其中数据的读取是由数据窗口机制控制的。

在Android SDK中还提供了Sqlite3工具，在adb shell模式下，可使用它来对数据库进行各种调试。

**需求分析与系统架构**

**功能需求**

本项目期望实现一个在Android平台上的简易版个性化微博推荐客户端，完成一个结合社会化标签的基于内容的推荐系统，需实现以下几个基本功能。

获取并展示用户的新浪微博信息流：一个微博客户端最基本的功能

根据用户信息对微博数据流进行推荐过滤：本项目研究的核心功能，利用推荐系统解决信息过载问题，优化用户的使用体验

根据关键字屏蔽相应微博：屏蔽不需要的信息，快速有效地清理微博信息流。

用户自定义喜好标签：通过社会化标签解决推荐系统常见的冷启动问题，同时使用户自行对喜好模型进行调整。

喜欢、隐藏微博：在发方便用户选择需要的信息的同时，持续为推荐系统提供反馈，逐渐优化用户的兴趣模型

**系统架构**

系统分为以下功能模块：

API调用模块：发送和处理包括登录授权在内的新浪微博API请求。

推荐模块：推荐系统的核心算法模块，根据用户喜好需求模型对API返回的微博进行过滤或排序。

个性化设置模块：推荐系统的模型分析和行为记录模块，设置个人喜厌标签、收藏或隐藏列表中的微博，并记录用户喜好信息。

数据存储模块：将用户喜好信息的存储。

界面显示模块：与用户交互的界面。

**系统预览**

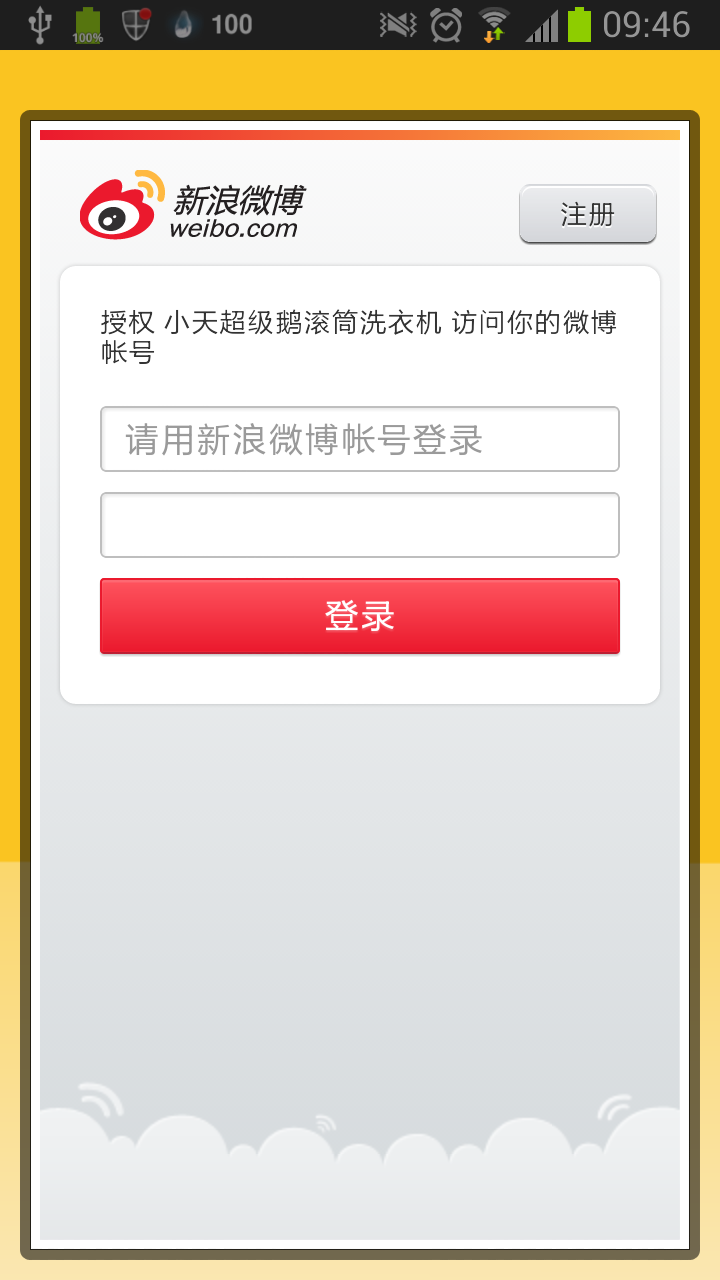
LaunchActivity

     应用启动界面，若用户未授权应用或授权过期，则弹出授权提示。确定后打开WebView开始OAuth2.0授权流程，授权完成后应用便可调用新浪微博API的各种接口开始工作。

     若用户第一次授权完成，则进入TagActivity进行标签的初始化设置。

     若用户已完成过初始化设置且授权未过期，则转跳到HomeActivity。





HomeActivity

应用的主要展示界面，分成“关注微博”和“猜你喜欢”两个模块，可点击顶部功能条或左右滑动切换模块。

两个模块结构相同，内容上一个是过滤后的用户关注的微博信息，另一个是系统推荐的公共微博，在列表顶部向下拖动可更新最新微博，在底部点击“更多”可加载之前的微博。

在每条微博的左边有两个功能按钮，将红心点亮则表示对这条微博表示喜好；而点叉则将这条微博从列表中删去。同时被操作的按钮对应的微博信息将会被记录下来，用于完善用户喜好模型。

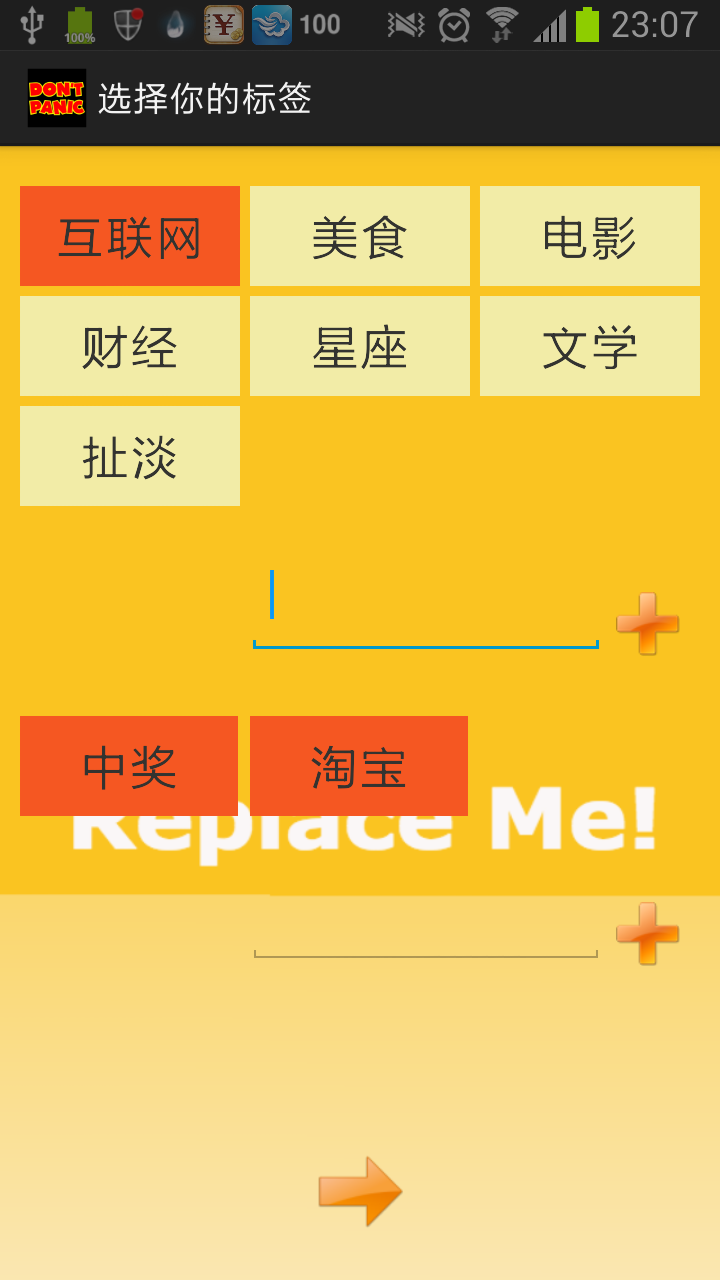


TagActivity

     从HomeActivity点击菜单键选择“标签”选项，可进入TagActivity。页面的上下两部分分别展示了用户的喜、恶标签。砖红色的标签为以选中的标签，即系统当前采用的设置，而米白色的为未选择标签，点击标签即可方便地进行选择和或反选择。用户可以使用两个输入框和按钮可分别向两个种类添加标签，也可通过点击选中或去除标签。点击下方的确认箭头则当前的标签信息被储存下来。

     对于喜爱关键字的标签，当用户授权应用后，应用会从API取得该用户的个人标签、收藏列表、以及收藏标签添加到应用的用户标签中；若从用户信息中获得的关键词过少，则系统补充一系列的标签供用户进行初始化选择。同时，用户加心微博的关键字也会被加入到标签中。带有标签中关键词的在用户微博时间线的出现概率加大。

     对于下半部分厌恶标签，带有标签关键词的微博将会被过滤掉，不出现在用户的时间线上。用户隐藏的微博的关键字也会加入到列表中作为建议并不默认选中。



**系统核心功能模块实现**

**API调用模块**

     在系统中所有的API调用都采用相似的实现机制，以请求关注用户的最新微博为例：

public class OAuth2 {

     private static StatusesAPI statusesAPI;

     ...

     public void init(

     ...

     statusesAPI=new StatusesAPI(AccessTokenKeeper.readAccessToken(context));

     }

     ...

    public void requestFriendStatus(long sinceId,long maxId,RequestListener listener){

              statusesAPI.friendsTimeline(sinceId, maxId, 15, 1, false, FEATURE.ALL, false, listener);

    }

}

     所有API调用的方法都封装于OAuth2类中。当用户对应用进行授权之后(同样是一个API请求)，应用可以获得一个OAuth2.0的AccessToken,调用大部分微博API需要这个的AccessToken，于是将这一token存放在应用的shared preference中。当需要调用API接口时，则利用AccessTokenKeeper.readAccessToken(context)方法获取Token。

     对于每一个API调用，都要对其注册一个RequestListener，通常由一个Activity实现RequestListener的接口，当API请求返回时调用listener的onComplete方法，在其中对返回值进行解析和完成后续操作。请求微博返回后，则会启动Service对微博进行推荐过滤。

     以下为本项目中应用到的新浪微博API

     oauth2/authorize 请求用户授权Token

     oauth2/access\_token 更新过期的AccessToken

statuses/friends\_timeline 获取当前登录用户及其所关注用户的最新微博

     statuses/public\_timeline 获取公共微博

     statuses/user\_timeline 获取某一用户发布的微博

     tags 获取指定用户的标签列表

     favorites 获取当前用户的收藏列表

     favorites/tags 获取当前用户收藏的标签列表

**推荐模块**

     为了避免阻塞主线程导致用户界面锁死，程序中注册了一个IntentService：TextAnalysisService进行推荐模块的工作。TextAnalysisService接到启动Intent之后，调用WeiboAnalyzer类中的静态方法对微博进行分词和过滤推荐操作。

     在获取到微博原始数据之后，呈现给用户之前，微博信息流将经过推荐模块筛选排序。因为微博内容长短常常存在显著区别，为了减小工作量，本项目中认为10字以下微博可供分析的信息量过小，不对这类微博进行分析操作。

**微博数据预处理与分词**

     为了能帮助计算机理解文本的内容主题，需要对其进行分词处理。文本分词是文本挖掘的基础，因为大多数自然语言处理模型，都是建立在词的基础上的，它为之后的文本索引、关键词提取等操纵提供了必要的条件。

     具体操作时需注意到，一条微博文本的内容经常包含着一些先与主题无关的信息，如被转发、提到用户的名称，短链接地址等。首先需要通过正则表达式清理文本除去些信息，以免对分词以及关键词提取造成干扰。

     文本清理完成后，便开始对一条微博进行分词。本项目中使用IKAnalyzer作为分词工具，IKAnalyzer是一个开源的java中文分词工具包，对开源搜索引擎工具lucene提高了良好的支持。IKAnalyzer采用其特有的“正向迭代最细粒度切分算法“，支持细粒度和智能分词两种切分模式。

例：“对张三说得确实在理”进行两种模式比较分词

   智能分词结果: 张三 | 说的 | 确实 | 在理

   最细粒度分词结果: 张三 | 三 | 说的 | 的确 | 的 | 确实 | 实在 | 在理

     因为对微博主题的分析对分词粒度不存在特别的需求，且最细粒度分词模式结果且给出了多层次的分词结果，对后续操作中计算文本频率可能存在一定影响，所以在本项目中使用的是智能分词模式。

     通常在文本处理中会遇到许多停止词(Stop Words)，指语言中使用出现过于频繁的单词，它们在文本中出现频率很高，但对于文本主题的判断意义不大。所以进行分词的时候往往需要忽略掉这些词语。同时在很多情况下，分词器无法识别一个新出现或词语时，则会将其拆分成更小的单位。所以在项目中为IKAnalyzer配置了一个停用词词典和一个扩展词语词典，以优化其分词结果。

**文本分类与文本相似度**

     最初的考虑是通过构建朴素贝叶斯分类器将微博分成几个大类，通过用户选择的标签推荐其中某几类的内容。但首先这样的方式实现成本较为高昂，需要分别大量抓取常见类别的微博用于训练分类器；其次微博明显不同于清晰地分门别类的新闻网站，微博上的信息多而杂乱，对微博进行人工分类本身并没有合适的依据，则所训练出来的分类器不具有代表性和实用价值。

     于是借鉴垃圾邮件分类算法的思路，不将微博文本按主题分类，而是按照用户喜好分为正负反馈两个集合，进行推荐和隐藏。这样训练集合的获取和分类就变得相当简单了。这却又带来了另一个问题，用户正负反馈的微博数据集的手机依赖于用户对微博做出回应，可能其中的样本过小而未能达到训练出稳定可靠分类器的要求。

     最后，我再次思考调整策略，认识到对文本进行推荐不一定要对其进行分类。根据基于内容的推荐算法理念，以与用户认可内容的相似度作为推荐标准的进行推荐的情况下，只需要计算待选微博与用户正反馈微博集合内元素的相似度即可。在这种情况下，即使正负反馈微博集合内的元素数量很少，只要不为空，便可针对其推荐或屏蔽。

     计算文本相似度时，采用的是余弦相似性定理（cosine similiarity）

     一般计算步骤如下：

　     首先提取出两条微博的关键词；

         将两条微博的关键词合并为一个集合，生成它们各自对于集合中的词语的词频向量；

         计算两个向量夹角的余弦大小，值越大就表示两条微博越相似。

**微博关键词提取**

     在对文本信息获取分析的过程中，确定特征关键词最常用方法就是TF-IDF方法。还有其他一些解决方案如Bayes分类、聚类分析、决策树、人工神经网络等也在不同领域中发挥着关键的作用。

      TF-IDF（term frequency–inverse document frequency）是一种常用于信息检索的加权技术。它是一种统计方法，用以评估一个字词在一个文档之中的的重要程度。字词的在特定文档中重要性随着它在文件中出现的次数成正比增加，但同时会随着语料库中文档包含它的比例成反比下降，原理基于这样一个假定，如果一个词语只在较少的文本中出现，则其对文档主题的区分能力则较大。

<http://www.ruanyifeng.com/blog/2013/03/tf-idf.html>

     在用TF-IDF提取微博关键字时对于逆文档频率IDF的计算需要语料库的支持，若直接把语料库存放于Android应用数据中，则占用空间过大且操作效率低下；若编写辅助程序创建语料库中所有词的集合表，表中包含词以及词在语料库条目中的出现频率，再将该表导入应用的数据库中，计算IDF时依然需要大量的数据库访问，依然会引起程序性能问题。

     重新审视一下微博文本的特性，和IF-IDF算法的原理，我得出了一个解决方案。对于最多只有140个汉字的微博来说，分词后产生的词语数量较为有限，且大多数时候除了停止词之外的词语的词频（TF）往往仅为1。若实现一个辅助程序计算出整个语料库中的词频表，将频率最高的一部分词语全部作为停止词，则剩余的词汇就有了较高的强的区分作用。这时可将分词的结果直接作为微博的关键词。

在这样的关键词提取策略下，计算两条文本的相似度过程如下：

例：

分词     句A：英/马拉松/仅/1人/完成/比赛/第二名/带着/5000人/跑错路

            句B:   第二名/带着/5000人/跑错路/英国/马拉松/仅/1人/完赛/成冠军

关键词集合：英,马拉松,仅,1人,完成,比赛,完赛,第二名,带着,5000人,跑错路,成冠军

生成词频向量：句A:[1,1,1,1,1,1,0,1,1,1,1,0]

                        句B:[1,1,1,1,0,0,1,1,1,1,1,1]

计算余弦值：8/10=4/5

     从上述计算过程不难看出，计算出的余弦相似度越小，则两条微博在用词上相差越大；而相似度大于一定阈值的时候，两条微博可认为两条微博相似度较高，或谈论相同话题的可能性较高，可进行推荐。然而若阈值设置得过大，则可能像例子中的情况一样，推荐出的微博与用户接收认可过的内容重复度过高，属于无效信息。

     因此，得出以下模型：

     N为未处理的微博信息流集合。

     M为用户喜好模型中的微博集合。

     对于Ni计算它与Mj的相似度Sim(i,j)

     若

     //TODO

     到此为止，我们得到了初始微博信息流与用户喜好模型

**个性化设置模块**

     推荐模块中用于计算相似度的微博样本是由个性化设置模块获取和记录的。

喜欢及隐藏微博

在主页面上每条显示的微博都带有喜欢和隐藏两个控件，操纵任意一控件，则将这条微博提取关键词

**标签设定**

     为了更好的掌握用户信息。

**总结与展望**

**相关研究**

目前关于微博的数据挖掘和推荐研究渐渐兴起，各种试图改进微博使用体验的第三方客户端也开始成为各种应用市场的热门，但其中将微博内容推荐投入应用的成果还较少。一个名叫优微的新浪微博客户端和本研究的方向相似，运用人工智能技术进行微博的过滤和推荐。2012年1月，其开发团队获得了200万的天使投资，期待有这样的产品推出，可供学习和研究。

     虽然在应用层面的研究较少，但本课题研究涉及到的核心问题如自然语言处理、文本聚类等国内外均有许多出色的研究团队和实践成果，如复旦自然语言处理组的开源中文处理工具包等，本人在课题的进行过程中也从他们的研究成功中受益颇多

**成果总结**

在理论上，本课题仔细研究了数据推荐系统领域的现状，并研究分析了推荐系统应用在微博这一特定领域的可行性和具体变通方案。

     在实践上，完成了一个微博推荐应用的Android demo版本，实现了结合社会化标签的基于内容的微博推荐，验证了课题研究理论的可行性。

     本课题的进行过程中，对许多针对同一问题的不同解决方案进行了学习和比较，分析每种方案的优缺点,并进一步对于其适合场景进行估计，从而选择对课题项目适合的实践方案。从零开始接触并了解自然语言处理和推荐系统的研究领域。同时，在demo的开发过程中也进一步深化了在Android平台的了解。

**进阶展望**

本课题的研究成果以及完成的应用demo还相当稚嫩，有许多可以进行后续完善的地方，我会就一下几个方面进行改进：

     微博的关键词提取算法过于简单，由于微博篇幅短小，常用的TF-IDF算法不能很好进行关键词提取，于是实践中仅进行了分词和除去停止词的处理，考虑可以参考Textrank算法进行改进。

     在文本相似度的处理上，目前采用的是计算关键词特征向量用余弦值，但这只代表其用词的相似性，希望考虑其他方式。

     在处理用户反馈行为时加入时间维度，并进行系统推荐时效性的研究。

     优化应用demo在用户交互上的体验，同时将其功能扩展为一个较为全面的微博客户端，完成一个完整的产品。

**参考文献**

项亮, 陈义, & 王益. (2012). 推荐系统实践.

范怀宇. (2012). Android开发精要

王晟, 王子琪, & 张铭. (2012). 个性化微博推荐算法. *计算机科学与探索*, *6*(10), 895-902.

周鑫, 彭斯俊, & 罗熹. 基于新浪微博开放平台的用户数据挖掘.

刘建国, 周涛, & 汪秉宏. (2009). 个性化推荐系统的研究进展. 自然科学进展, 19(1), 1-15.

李静梅, 孙丽华, 张巧荣, 等. 一种文本处理中的朴素贝叶斯分类器[J]. 哈尔滨工程大学学报, 2003, 24(1).

陈渊, 林磊, 孙承杰, & 刘秉权. (2011). 一种面向微博用户的标签推荐方法. *智能计算机与应用*, *1*(3), 21-26.

赵晨琳,马春娥. (2011). 探索推荐引擎内部的秘密，第1部分:推荐引擎初探

曹海. (2006). 基于文本内容分析的过滤技术研究 [J]. 四川大学学报: 自然科学版, 43(6), 1248-1252.

Zhang, Z. K., Liu, C., Zhang, Y. C., & Zhou, T. (2010). Solving the cold-start problem in recommender systems with social tags. EPL (Europhysics Letters), 92(2), 28002.

JA Konstan and J Riedl. (2012). "Deconstructing Recommender Systems: Recommended for you," IEEE Spectrum

**致谢**

感谢李江峰老师的指导

感谢身边同学朋友们的支持与陪伴

感谢老大在项目最忙的时候让我请假一个月把精力投入到毕设上

感谢在软院度过的四年时光，培养了我独立自学的能力。

感谢互联网上乐于分享的人们