互联网+实验项目开题

* 1. 小组成员：

俞佩成（算法研究）、

马帅奇（Android开发）、

耿睿（UI设计）、

杨江锐、杨学昊（前端交互设计）、

危欢（后台+测试等）

项目名称：社区化的网络音乐播放器（互联网+音乐）

* 1. 研究背景
     1. 国内的情况

2013年网易正式发布网易云音乐，到2017年已经具备音乐播放，推荐歌单，听歌识曲以及社交等主要功能，是互联网+音乐的完美体现。分别来说，过去在网络上可以通过在线播放器听音乐，或者通过资源下载使用本地播放器播放。网络与音乐隔离较大，并且mp3/mp4保有着主要市场，互联网模式落后，不利于网络+音乐的结合。但是现在，互联网已经大大发展，大多数传统的音乐播放器，如酷狗播放器、千千静听、酷米等等都要么经过转型，面向用户提供跨平台的在线音乐播放支持和广泛的音乐仓库，或者慢慢消亡，消失在众人的记忆里。

比较不同的产品有助于我们发现业界标杆的优势，网易云音乐通过旗下互联网产品积累了大量的活跃用户；抓住音乐这个大众文化刚需，使受众更为广泛；大胆建立自己的云音乐库方便用户使用互联网下载音乐；并提供了关键的个性化歌单，大大增强用户粘性；在使用的过程中，提供用户之间相互交流的平台，让听歌不再孤单；此后的听歌识曲更是让云音乐如虎添翼。

* + 1. 国外的情况

对于国外，使用Play Store下载的music player不计其数，其中最优秀的是Audiomack(17), Samsung Music(22)和Google Play Music(34)。此3者的着重点稍有不同，S支持多种的音乐格式来提高自己的竞争力，还有控制音乐播放速度的能力，S和G均利用广告（可付费清除）增加变现途径，A包含心情歌单这项特殊功能等。而因为版权和听众的地域不同，但有的缺少歌词，有的用户引导不佳，或没有高清音质，或不能下载设置为铃声。

* + 1. 小结

我们认为国内目前这个环境造就了网易云音乐，虽然版权问题突出，但是为广大用户创造了良好体验。为什么国外的产品没有听歌识曲，我想应该是因为搜到的歌曲有版权导致App也无法下载，容易给用户造成不良的印象。而针对听歌识曲的应用，国内还有一款Music Radar。我们借鉴一个相对成功的音乐播放器，主要研究服务器高速解析客户端数据和兴趣挖掘算法的细节，学习如何发挥移动端轻量高效的优势。我们的目标大致列出如下：

* 用户注册、登录功能
* 好友系统
* 在线播放歌曲
* 推荐感兴趣的歌曲
* 歌词自动获取和解析
* 音乐列表（歌单）
* 在线点赞和评论系统
* 上传自己的音乐片段（高手在民间）
  1. 研究现状

针对以上需求，我们针对兴趣推荐算法和前后台交互的技术的调研如下。

* + 1. 一种基于模式的兴趣挖掘算法-李俊芳-2014年3月28日科技通报

总体来说，用户感兴趣的因素可以从应用内的搜索、会话的保持时长、访问的本体中具有的相似语义，或者留下的评论等因素来获得。为了能挖掘用户的兴趣点，算法的系统应该包括以下能力：从数据源发现兴趣、兴趣模型的表示、挖掘兴趣的方法。

本篇文献在前篇集中讨论了6种现成的兴趣挖掘方案的主要特性和缺陷（概略见下），从缺少表示用户兴趣的方法切入提出了自己的方案。

|  |
| --- |
| * 基于用户反馈标注的概率，解决兴趣漂移问题。这种方法能描述用户的兴趣并且可以更新，但是缺乏兴趣中间的语义关联。 * 使用SVM从用户的查询中识别其意图的方案，由于SVM从本体中分类出了摘要，通过传播激活过程推导出用户的兴趣，从而能够准确的描述兴趣的方向。 * 基于ODP标注提交搜索返回前10个查询结果、session中点击日志、查询结果点击记录等构建用户兴趣模型，但是结果仅限于查询过的session资源。 |

表格1.1 - 3种兴趣挖掘方案

方案从用户的行为记录中发现兴趣，不断从数据中更新用户的兴趣点，从而实现按兴趣推荐的功能。由于兴趣表示是兴趣模型的基础所在，因此作者提出了12项定义：

1. 兴趣粒子IP：等同于标签（概念集合记为C）;
2. 兴趣停留时间t：用户在某Ci映射的所有对象上停留的时长之和；
3. 兴趣持续时间T：Ci的两次访问时间Tnow & Tprevious差的绝对值，可以评判兴趣漂移；

|  |  |
| --- | --- |
|  | (公式1) |

1. 兴趣得分公式S(C)：多项式因子是访问的对象数V(C)、收藏数F(C)、和T(C)，并用权重表现不同因子的重要性；

|  |  |
| --- | --- |
|  | (公式2) |

1. 强兴趣QI：

|  |  |
| --- | --- |
|  | (公式3) |

1. 弱兴趣WI：

|  |  |
| --- | --- |
|  | (公式4) |

1. 无效兴趣NI：

|  |  |
| --- | --- |
|  | (公式5) |

1. 在e中，若有且仅有1个S(C)大于，则称为唯一兴趣OI，多个的集合称为多重兴趣MI，用以标定兴趣漂移；
2. 兴趣漂移DI：在相邻两段等长的时间里，若存在C从WI转变成QI或相反则称为兴趣漂移；
3. 长期兴趣LI：在相邻两段等长的时间里，存在S(C)始终大于，则称为长期兴趣；
4. 短期兴趣SI：在相邻两段等长的时间里，存在S(C)或S(C)，则称为短期兴趣。

将兴趣分类，调整权重和阈值的值就能成功把兴趣的表示完成了，此时再按照上述的步骤，从不同的因子获取兴趣粒子，计算它们的S(C)就可以分类成不同的兴趣模式P (stands for pattern)。

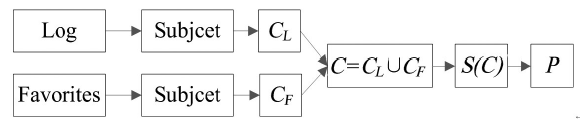


图1.1 基于模式的兴趣挖掘算法

这样应用可根据用户的兴趣模式，对长期兴趣和强兴趣对象进行推荐。

* + 1. 数据库访问对象模式+DBHelper抽象数据库访问技术

我们应该建立起实体类来承载歌曲的名称、时长、专辑、歌手等信息，本地存储将使用SQLite做数据库，建立查询表来存储对应的信息。因为字段较多，最好将反复CRUD的查询语句用DBHelper封装起来，DAO将业务接口暴露给应用。数据访问对象模式（Data Access Object Pattern）正是用于把低级的数据访问API或操作从高级的业务服务中分离出来的。

DAO模式需要以下参与者：

* 数据访问对象接口（Data Access Object Interface）-该接口定义了在一个模型对象上要执行的标准操作；
* 数据访问对象实体类（Data Access Object concrete class）-该类实现了上述的接口。该类负责从数据源获取数据，数据源可以是数据库，也可以是 xml，或者是其他的存储机制；
* 模型对象/数值对象（Model Object/Value Object）-该对象是简单的 POJO，包含了 get/set 方法来存储通过使用 DAO 类检索到的数据。

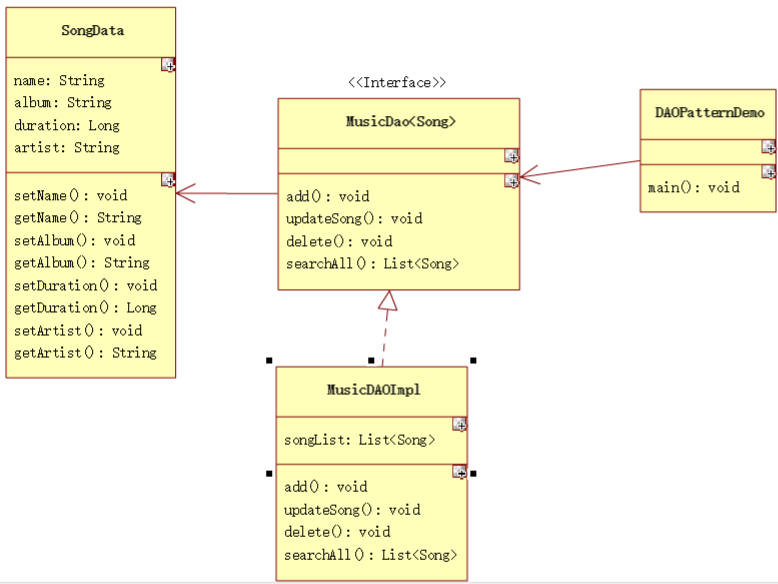


图1.2 DAO模式的参与者

当中的List就是从DB中获取的数据，遍历Cursor得到，从而受到类的维护，提高了重复访问的性能。我们需继承SQLiteOpenHelper，复写onCreate(db: SQLiteDatabase)和onUpdate(db: SQLiteDatabse, oldVersion: Int, newVersion: Int)。这样我们就可以在MusicDAOImpl中使用此DBHelper类的getReadableDatabse() / getWritableDatabase()进行安全的数据库查询，查询使用db的rawQuery()执行即可。

对于客户端来说，数据库操作就抽象为方法的调用和传递参数了。

代码1

|  |
| --- |
| public class DBHpler extends SQLiteOpenHelper {   public DBHpler(Context context) {  super(context, DBData.*DATABASE\_NAME*, null, DBData.*VERSION*);  }   @Override  public void onCreate(SQLiteDatabase db) {  //DO EXECUTION  }   @Override  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  // DELETE THE OLD TABLES OR YOU CAN UPDATE THEM  } } |

* 1. 困难工程问题的提出

首先简介一下移动应用的开发架构与网页应用的差别。相比后者，它的模式更像C/S。Activity负责与用户的交互，显得像前端；由于Service组件可以后台执行需要长时间保持的任务（如数据解析，状态监测），或从数据库提供数据，Activity与Service相互通信，使得Service有点像后端。

但这里的Service并不是后端，一不需要框架二不考虑负载，app可以只用Service执行简单的录音工作。所以传统意义上的后端仍然体现在移动互联网应用的远程服务器上。

总览我们的项目，难点有二：

* App：开发的工作量会比较大，难度中等但需要相当的API和开发模式的知识；
* 算法：难点在算法的参数标定和代码实现上。

UI层次的杂项问题如下表所示。

|  |
| --- |
| * 歌词方案设计与实现 * 歌词显示原理 * 创建实体类LrcRow * ILrcBuilder接口和DefaultLrcBuilder歌词解析构造器 * ILrcViewListener接口（在ILrcView中使用） * 同步显示歌词功能 * 同步显示歌词功能 * 拖动歌词的功能 * 缩放歌词的功能 * UI适配机型 * 如何绘制歌词 |

* + 1. 研究工作的整体安排

首先需要的是软件框架的设计，包括各页面UI、跳转逻辑、数据库设计等。从各大音乐软件中借鉴优点，在使用后去掉不适用的地方，来完善我们的软件预估需要二周完成。其次是api的使用。由于没有服务器，选择使用各大音乐网站的api。从其他使用者的使用情况来看，我们可以选择出最适合此软件的一个接口。完成api这部分工作需三周时间。再是对于推荐音乐的算法研究。这部分是我们比较陌生的一块。通过收集文献，系统分析，定性的分析后，选择一种或综合多种算法来实现我们的需求。预估需要三至四周。

* + 1. 技术路线研究方法

技术路线:

绪论

提出课题

以人为本搜索

研究现状

以物为本建立关系

用户潜在因子矩阵

音乐潜在因子矩阵

研究分析

偏好空间

建立评价规则

收集用户偏好

方案设计

寻找相近用户

寻找符合歌曲

推荐歌曲

结论与改进

总结

研究方法：

（1）文献搜集法。搜集互联网+、各大网站api、推荐系统算法、需求预测等相关期刊专著以及网络文献，对资料进行归纳、整理、分类、分析，得出相关的研究成果，为本论文的研究奠定理论基础。

（2）系统分析法。在整个项目研究过程中，始终贯彻以理论知识为基础，以社会实践为根本，进行市场调研分析，以此对本论文进行布局、设计，保证本论文体系结构的逻辑性、科学性、可行性。

（3）定性与定量分析法。本文采取的定性方法是分析本软件在各个层面的缺陷与优势，并找出现阶段发展存在的主要问题；定量方法是运用需求预测模型预测每个阶段所需要耗费的时间，运用成本比较分析法设计运作模式等。

* + 1. 细化主要研究内容

分析“商品推荐”系统的算法的四种类型：

1. 基于物品自身相似度：通过TAG分类进行推荐。
2. 基于用户协同过滤：采集用户喜爱的音乐，若两者偏好相似，推荐另一人未听过的歌曲。
3. 基于物品协同过滤：相似的用户喜欢A和B，将A和B的相似度记录并绑定。
4. 混合算法：融合多种推荐系统。

潜在因子算法：每个用户（user）都有自己的偏好，比如A喜欢带有清新的、吉他伴奏的、王菲等元素（factor），如果一首歌（item）带有这些元素，那么就将这首歌推荐给该用户，也就是用元素去连接用户和音乐。每个人对不同的元素偏好不同，而每首歌包含的元素也不一样。我们通过用户的行为(如收藏，跳过等)，在经过分析后能获得实际评分矩阵。实际评分矩阵可以较为准确的评价歌曲对于用户的合适度，当较为相似的用户出现时，便将这种优质歌曲优先推荐。

如用户潜在因子矩阵(数值越大越喜欢)：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 流行 | 古典 | 重金属 | 轻音乐 |
| 张三 | 0.5 | 0.4 | 0.1 | 0.6 |
| 李四 | 0.4 | 0.3 | 0.7 | 0.1 |
| 王五 | 0.7 | 0.2 | 0.2 | 0.4 |

如音乐潜在因子矩阵(数值越大占比越重)：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 流行 | 古典 | 重金属 | 轻音乐 |
| 音乐A | 0.2 | 0.3 | 0.9 | 0.1 |
| 音乐B | 0.4 | 0.4 | 0.1 | 0.3 |
| 音乐C | 0.5 | 0.2 | 0.3 | 0 |

由此两表可以得出某人对某音乐的偏好：

如张三对音乐A的偏好：0.5\*0.2+0.4\*0.3+0.1\*0.9+0.6\*0.1=0.37

|  |  |  |  |
| --- | --- | --- | --- |
|  | 音乐A | 音乐B | 音乐C |
| 张三 | 0.37 | 0.55 | 0.6 |
| 李四 | 0.81 | 0.4 | 0.51 |
| 王五 | 0.42 | 0.5 | 0.45 |

由此推荐张三音乐C，李四音乐A，王五音乐B。

这些因子的得出来源于用户的行为与歌曲的tag。

* 1. 实施方案可行性讨论

从经济可行性方面来看，本软件几乎不需要成本。使用的开发平台与软件皆是开源免费。利益方面由于非商业用，不多考虑关于利益方面的问题，主要偏重实用。

从技术可行性方面来看，在完成一个基础的框架方面并无大问题，使用java与sql即可完成。在api和推荐算法方面，就目前我们通过文献所了解来看，要完成也是可行的。

从法律可行性方面来看，本软件为独立开发，在技术上没有使用任何现有的软件与方法。所以在法律方面不会存在侵犯专利权、侵犯版权等问题，完全按照合同规定的责任履行。