逛周技术报告

一.客户端产品框架

0. 文件树图

```
├──guangzhou
|    ├──model（数据模型类文件夹）
|    ├──net（网络请求类文件夹）
|    └──ui（和UI相关的Activity，Fragment等类文件夹）
|        └──product
|    └──db（数据库存储相关）
├──tool
|    └──divider
└──widget
```

1. 总体框架：

a. **MVC**: 将应用程序划分为三种组件，模型 - 视图 - 控制器（MVC）设计定义它们之间的相互作用。[2]

模型（Model） 用于封装与应用程序的业务逻辑相关的数据以及对数据的处理方法。“ Model ”有对数据直接访问的权力，例如对数据库的访问。“Model”不依赖“View”和“Controller”，也就是说， Model 不关心它会被如何显示或是如何被操作。但是 Model 中数据的变化一般会通过一种刷新机制被公布。为了实现这种机制，那些用于监视此 Model 的 View 必须事先在此 Model 上注册，从而，View 可以了解在数据 Model 上发生的改变。（比如：观察者模式（软件设计模式））

视图（View）能够实现数据有目的的显示（理论上，这不是必需的）。在 View 中一般没有程序上的逻辑。为了实现 View 上的刷新功能，View 需要访问它监视的数据模型（Model），因此应该事先在被它监视的数据那里注册。

控制器（Controller）起到不同层面间的组织作用，用于控制应用程序的流程。它处理事件并作出响应。“事件”包括用户的行为和数据 Model 上的改变。

b. **异步框架：**响应式开发-RxJava

响应式编程是一种面向数据流和变化传播的编程范式。这意味着可以在编程语言中很方便地表达静态或动态的数据流，而相关的计算模型会自动将变化的值通过数据流进行传播。响应式编程是一种基于异步数据流概念的编程模式。响应式实际上是观察者模式加上事件源的完成通知能力、错误传播能力和监听者同事件源通信的能力。响应式流是一种规范，ReactiveX是一种常用的跨平台实现。

RxJava是 ReactiveX 在JVM上的一个实现,它为观察者模式提供了一种通用的实现，并且提供了丰富的操作符来处理数据流。它同时支持不同线程之间切换，使得它经常用来实现Android中的异步调用。

c. **消息框架：**

(1) Looper - MessageQueue - Handler

通过一个消息队列和一个无限循环取出和分发消息来处理来自这个线程或其他线程的消息。

(2) EventBus

EventBus是一款针对Android优化的发布/订阅事件总线。简化了应用程序内各组件间、组件与后台线程间的通信。优点是开销小，代码更优雅，以及将发送者和接收者解耦。

2. 网络层

(1) 网络请求框架：Retrofit + OkHttp

Retrofit是一个RESTful的HTTP网络请求框架，基于OkHttp并遵守RESTful API的设计风格，通过注解配置网络请求参数，支持同步和异步的网络请求，支持多种数据的解析和Json、Xml等序列化格式，同时支持RxJava。

(2) GzService：封装网络有关对象，给其他层提供统一的接口。

(3) Api：封装和服务器通信的API

3. UI层

依赖注入框架： Android Annotations

(1) 依赖注入(Dependency injection):支持view, extras, system service, resource等等

(2) 简单的线程模型(Simplified threading model):进行方法注解以让该方法在UI线程或后台线程进行执行

(3) 事件绑定(Event binding):进行方法注解以让方法执行view的时间而不用再添加一些监听

(4) REST client:创建一个接口,AndroidAnnotations用来实现

(5) 编译检测:提供的多种注解,用于检测代码编译时可能存在的异常,并给开发者相关提示,提高代码质量

图片加载框架：Glide

Glide是一个Android的图片加载和缓存库，它主要专注于大量图片的流畅加载，Glide几乎可以胜任任何你需要使用到图片从网络拉取，压缩，显示的场景。

4. Model层

业务逻辑

POJO

5. 数据存储层

ORM框架 DbFlow

dbflow是一款android高性的ORM数据库.可以使用在进行项目中有关数据库的操作。dbflow是Android SQLite ORM 的一个工具库。综合了 Active Android, Schematic, Ollie,Sprinkles 等库的优点；能生成 ContentProvider

优点①：扩展性

ORM所需的数据类只需要实现Model接口即可，而不需要必须继承一个类，同时为了方便，我们还是推荐继承BaseModel，这是Model接口的一个标准实现。这样你既可以通过继承一个来自其他包的非Model类来生成你的数据库表，也可以通过继承一个Model类并通过添加@Column注解的属性向表中自由添加列。这一切都是为了方便你的使用。

优点②：速度

DBFlow基于AnnotationProcessing(注解处理器），通过编译期代码生成，运行时对性能是零损耗的。通过模板来为你维护生成的代码。通过缓存和尽可能地重用对象，我们得到了比原生SQLite更快的速度。同时我们还支持懒加载（lazy-loading），比如对于@ForeignKey和@OneToMany，这使得我们有着更高效得查询效率。

二. 服务器端产品框架：

0. 文件树图：

```
├──bin
├──models
├──node_modules
├──public
|   ├──images
|   ├──javascripts
|   └──stylesheets
├──routes
└──views
```

1. 框架优势阐述：

①Node.js

这次服务端我们使用node.js来实现，nodejs的优点是事件驱动，采用异步回调的方式来处理时间，所以nodejs在面对大量的请求时能够保持较好的稳定性。其次，此次服务端扮演的角色主要是根据前端请求返回json数据以图片，即主要的都是IO密集型的请求，对CPU的压力并不大，所以使用异步回掉可以很好地发挥作用。

②Express

Nodejs中我们使用的框架为Express。Express 是一种保持最低程度规模的灵活 Node.js Web 应用程序框架，为 Web 和移动应用程序提供一组强大的功能。这个框架的优点在于资源丰富，实现网络框架较为简单，能够通过简单的路由将请求分类。

这个框架将我们的路由分为了User，租人Post两个主路由。然后再user中有一个路由，可以找到这个用户所拥有的Post及Order。

Android前端通过Get/Post请求访问上传数据，服务端返回json字段，以此完成客户服务端的通信。

③MongoDB

在数据库方面，我们使用了较为流行的mongoDB，这是一种非关系型数据库，在面对大量的网络访问请求时查找效率会比关系型数据库更快，以保证服务器的稳定性。

个性化推荐算法：

基于聚类的推荐算法

在这次的例子中，商品就是我们的Post（在我租界面的表项），而品味则需要根据每

个Post所带有的特征来决定。这些特征包括：饮食，景点，技能等等。

举几个例子，如果我们的app中有一个条目是“带你吃杭州美食，游雷峰塔”。那么这个条目在“导游“添加时，导游就会给这个条目带上两个tag：饮食与景点。我们的算法就会给饮食和景点这两个特征打上一个较高的数值；另外由于这个导游并没有给游客提供技能方面的体验，就会给技能项打上一个较低的数值。所以，我们应该可以得到如下的矩阵：

	饮食	游景点	技能
杭州美食雷峰塔	0.8	0.7	0.0
山东日照打海米	0.2	0.2	0.9
中大一日游	0.1	0.9	0.0

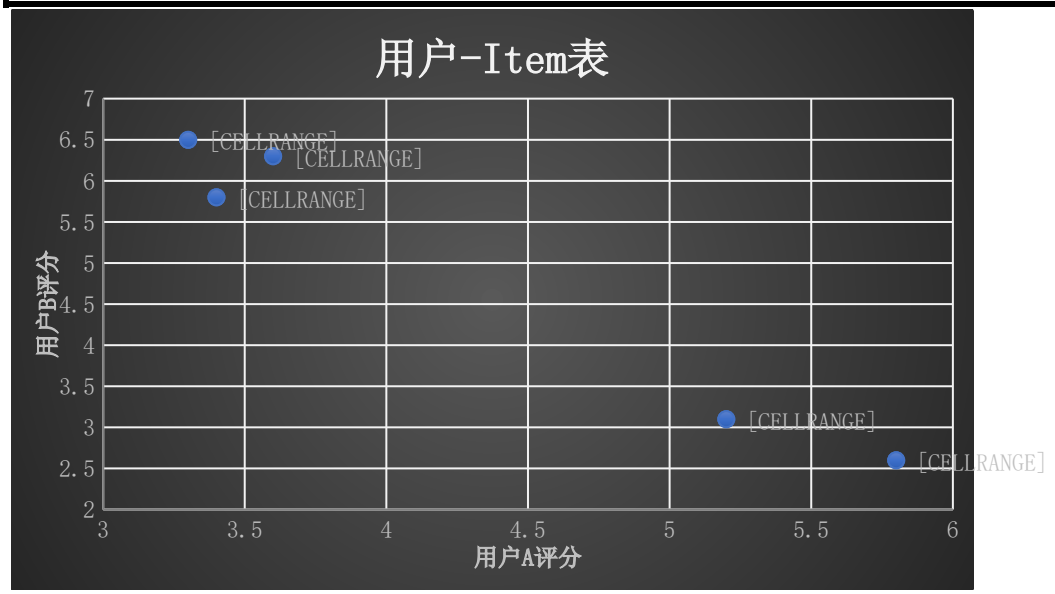
当我们积累了足够多的Post与特征之后，我们就可以通过聚类的方法来将类似的Post聚集在一起。然后在推荐时，如果某一个用户选择了某一个Post，那算法就会给他推荐同处一个聚类中的其他Post。

基于物品的协同过滤推荐算法

基于物品的推荐算法通过计算不同用户对不同物品的评分获得物品间的关系，基于物品间的关系对用户进行相似物品的推荐。这里的评分代表用户对商品的态度和偏好。简单来说就是如果用户A同时购买了商品1和商品2，那么说明商品1和商品2的相关度较高。当用户B也购买了商品1时，可以推断他也有购买商品2的需求。

1. 寻找相似物品

	用户A	用户B
杭州美食雷峰塔（商品1）	3.3	6.5
中大一日游（商品2）	5.8	2.6
重庆美食（商品3）	3.6	6.3
珠海渔女（商品4）	3.4	5.8
秦皇岛一日游（商品5）	5.2	3.1



欧几里德距离评价

在基于物品的协同过滤算法中，我们依然可以使用欧几里德距离评价来计算不同商品间的距离和关系。在上图中我们可以很明显的区分到：杭州、重庆与珠海距离较近且关系密切；而秦皇岛和中大距离较近。

欧几里德距离评价

	系数	倒数
商品1&2	4.63	0.18
商品1&3	0.36	0.73
商品1&4	0.71	0.59
商品1&5	3.89	0.20
商品2&3	4.30	0.19
商品2&4	4.00	0.20
商品2&5	0.78	0.56
商品3&4	0.54	0.65
商品3&5	3.58	0.22
商品4&5	3.24	0.24

倒数越大，关系越近。