目录

[1. 功能模块 3](#_Toc480899928)

[1.1. 功能模块架构 3](#_Toc480899929)

[1.2. 功能模块设计 3](#_Toc480899930)

[1.2.1.用户端 3](#_Toc480899931)

[1.2.2.管理员端 4](#_Toc480899932)

[2. 爬虫技术 5](#_Toc480899933)

[2.1. 爬虫介绍 5](#_Toc480899934)

[2.2. 页面解析 5](#_Toc480899935)

[2.2.1.使用语言 5](#_Toc480899936)

[2.2.2.数据来源分析 5](#_Toc480899937)

[2.2.3.页面解析工具 5](#_Toc480899938)

[2.3. 爬虫框架 5](#_Toc480899939)

[3. 算法技术 6](#_Toc480899940)

[3.1. 分词算法 6](#_Toc480899941)

[3.1.1. 分词器的选择 6](#_Toc480899942)

[3.1.2. 分词过程 6](#_Toc480899943)

[3.1.3. 分词过程算法 6](#_Toc480899944)

[3.1.4. 分词器附带功能 7](#_Toc480899945)

[3.2. 关键词提取算法 8](#_Toc480899946)

[3.3. 垃圾评论过滤算法 8](#_Toc480899947)

[3.3.1. 垃圾评论检测算法 8](#_Toc480899948)

[3.3.2. 垃圾评论检测框架设计 9](#_Toc480899949)

[3.3.3.垃圾评论数据集 9](#_Toc480899950)

[3.3.4. 9](#_Toc480899951)

[3.4. 评论情感分析算法 9](#_Toc480899952)

[4. 关键词提取技术 9](#_Toc480899953)

[4.1.1. TF-IDF算法使用 9](#_Toc480899954)

[4.1.2. 针对不同部门的关键词分类 9](#_Toc480899955)

[4.1.3. 针对不同部门的关键词提取 10](#_Toc480899956)

[4.1.4. 自定义分词的添加 10](#_Toc480899957)

[4.1.5. 基于用户需求的TF-IDF算法改进 10](#_Toc480899958)

[5. 软件技术 11](#_Toc480899959)

[5.1. Laravel框架 11](#_Toc480899960)

[5.1.1. 模型—视图—控制器 11](#_Toc480899961)

[5.1.2. 响应流程 11](#_Toc480899962)

[5.2. Nginx服务器 11](#_Toc480899963)

[5.3. 基于Ajax技术的Web服务架构 11](#_Toc480899964)

[6. 前端交互技术 12](#_Toc480899965)

[6.1. 框架选择 12](#_Toc480899966)

[6.2. 产品颜色设计 13](#_Toc480899967)

[6.3. 响应式布局 14](#_Toc480899968)

[7. 数据可视化技术 15](#_Toc480899969)

[7.1. 基于几何的技术 15](#_Toc480899970)

[7.2. 面向像素技术 16](#_Toc480899971)

[7.3. 基于图标的技术 16](#_Toc480899972)

[7.4. Echarts技术 17](#_Toc480899973)

[7.5. 独立坐标系 17](#_Toc480899974)

[7.6. 多维数据支持和丰富视觉编码 18](#_Toc480899975)

[8. 数据库技术 19](#_Toc480899976)

[8.1. 可信任的数据库 19](#_Toc480899977)

[8.2. 数据库表关系 20](#_Toc480899978)

[8.3. 数据库表详细设计 20](#_Toc480899979)

[9. 系统实现 22](#_Toc480899980)

[9.1. 系统用户端实现 22](#_Toc480899981)

[9.2. 系统管理员端实现 22](#_Toc480899982)

# 功能模块

## 功能模块架构



## 功能模块设计

### 1.2.1.用户端

**口碑详情模块**

本模块主要展示用户所关注APP的总体情况，在该模块中，用户查看评论数、口碑指数、下载量等信息在不同时间段内的变化趋势

**热词展示模块**

该模块中，在对用户的评论进行分词的基础上，将分词归类，分成活动类、功能类、问题类这三个类别，呈现给用户的是每个类别下Top10的词及其词频；且考虑到不同的部门关注点不一样，为不同部门定制个性化的词库，使其可以更加精确地获取到有用的信息。

**口碑比较模块**

在该模块中，根据用户的需求将不同的APP进行比较，将不同APP的相关信息以图表的形式展示给用户，可以更加清晰地进行变化趋势的比较。

**版本比较模块**

将同款APP不同版本间的相关信息放在一起进行比较，以图表的形式呈现给用户。用户还可以将图表下载下来。

**分词展示模块**

以三维词云图的方式展现一段时间内热词的分布情况、展现每一个分词词频数随时间的变化趋势、展现一天中不同词的词频比较。

**区域展示模块**

在该模块中，用户可以查看自己所关注的APP在不同国家不同地区的详细信息，也可以通过颜色的对比清楚地比较APP在不同地区的使用差异。

**选择查询模块**

在该模块中，用户可以根据所需按时间段、按星级、按APP名称、按相关关键词、按平台名称对APP评论进行查询。

**评论展示模块**

将评论内容、评论星级、评论时间等评论详情以表格的形式呈现出来，且采用分页技术，加快查询次数，增加用户体验。

**APP管理模块**

在该模块中，用户可以选择自己想要关注的APP，以便下次打开页面时可以直接看到关注APP的详情；用户可以将APP进行分类，更好地与其他同类APP进行比较。

**个人信息管理模块**

在该模块中用户可以选择自己的部门、可以对自己的相关信息进行更改、可以处理相关邮件。

**分词管理模块**

分词管理模块将为用户提供个性化服务为出发点，在管理员为其设定的特定词库基础上，人为地增加想关注的分词，在下次登陆时可以优先分析展示。

### 1.2.2.管理员端

* **列表展示模块**

展示从数据库中获得的用户信息，包括用户名、账号、分组等重要信息，管理员可为不同的用户分组，进而与相关的权限进行绑定。

* **权限管理模块**

在该模块中，管理员可以为不同的组分配可以查看的APP的相关信息，进而为不同的组提供个性化的服务。

* **本地导入模块**

通过点击相应按钮，选择本地已经整理好的Excel内的评论内容，导入到数据库，并可以通过评论展示模块显示出来。

* **自动导入模块**

管理员通过配置获取地址、用户名和密码，即可自动从网络平台获取数据，并自动对获取评论进行分词并分类，分类结果保存在数据库中。

* **评论展示模块**

该模块是导入成功与否的验证环节，管理员无需登录数据库查看是否导入成功，在后台管理员界面就可以查看评论的详细信息。

* **APP详情模块**

在该模块中，呈现的是APP在各大应用商店各个时间段的总体情况。

* **分词模块**

在该模块中，用户可以为不同的APP分配不同的分词词库，并可随时根据所需增加分词，为不同的分词规定优先级。

* **分类管理模块**

该模块是分词管理的基础，将不同类型的APP进行分类管理，也可以通过点击相关按钮进行增加或删除APP。

# 爬虫技术

## 爬虫介绍

由于期望获得来自各大平台的APP评论以及其他相关数据，为了减少管理者的工作，通过爬虫技术自动获取评论数据并结构化地存入数据库中。

已覆盖的平台有APP Store、豌豆荚、百度手机助手、vivo应用商店、应用宝、小米应用商店、360手机助手、oppo应用商店、魅族应用商店。

针对于某一APP，管理者只需配置其发布的平台，就可实现将该APP位与该平台的数据进行自动导入。

## 页面解析

### 2.2.1.使用语言

我们选择Python作为爬虫语言。该语言简洁方便、对于文本及字符串的处理具有优势、同时具有多种成熟爬虫框架可供选择。

### 2.2.2.数据来源分析

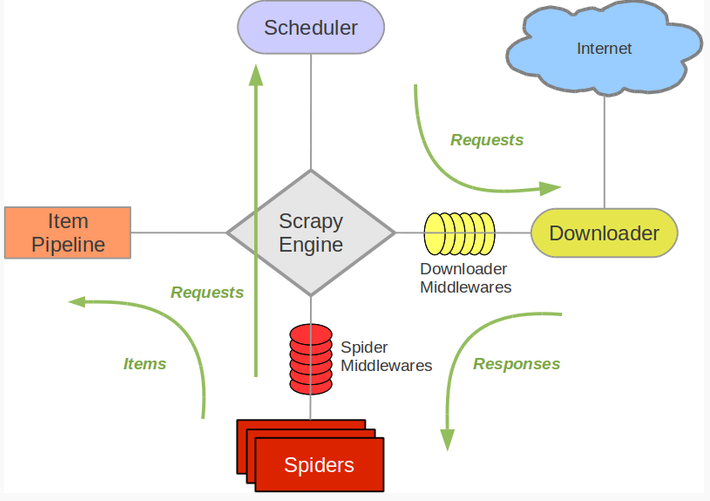
评论的数据来源主要可分为官方API、静态页面中的数据、应用AJAX技术实现异步加载的动态数据。

### 2.2.3.页面解析工具

应用Python中的 Beautiful Soup 库从HTML或XML文件中提取结构性目标数据。该工具主要功能为将HTML的标签文件解析成树形结构，然后方便地获取到指定标签的对应属性。因此对于静态页面，只要标定评论数据位于页面中的位置属性，就能方便地解析获取评论文本。

## 爬虫框架

采取Python中最为流行的爬虫框架Scrapy。该框架中的组件包括引擎（Scrapy Engine）、调度器（Scheduler）、下载器（Downloader）、爬虫器（Spiders）、数据管道（Item Pipeline）。这些组件的互相协调可实现并行爬取数据，提高爬取速度；通过分布式爬虫避免应用平台的反爬虫机制，降低本地IP被封的几率；通过PhantomJS抓取AJAX异步加载的请求地址，实现AJAX异步加载的评论数据的获取。



# 算法技术

## 分词算法

### 分词器的选择

本项目Viiision—APP评论数据分析系统采用结巴(Jie Ba)分词技术，结巴分词是国内程序员用Python开发的一个中文分词模块，它的特点有：

1. 支持三种分词模式：

* 精确模式，试图将句子最精确地切开，适合文本分析；
* 全模式，把句子中所有的可以成词的词语都扫描出来，速度非常快，但是不能解决歧义；
* 搜索引擎模式，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。

1. 支持繁体分词
2. 支持自定义词典

### 3.1.2.分词过程

1. 加载字典，生成Trie树；
2. 给定待分词的句子，使用正则获取连续的中文字符和英文字符，切分成短语列表，对每个短语使用DAG(查字典)和动态规划，得到最大概率路径，对DAG中那些没有在字典中查到的字，组合成一个新的片段短语，使用HMM模型进行分词，也就是识别新词，即识别字典外的新词；
3. 使用python的yield 语法生成一个词语生成器，逐词语返回。

### 3.1.3.分词过程算法

* **基于Trie树结构实现词典扫描并生成有向无环图 (DAG)**

结巴分词自带了一个叫做dict.txt的词典，里面有2万多条词，包含了词条出现的次数和词性。Trie树结构实现的词图扫描，就是把这2万多条词语，放到一个Trie树中，而Trie树是有名的前缀树，也就是说一个词语的前面几个字一样，就表示他们具有相同的前缀，就可以使用Trie树来存储，具有查找速度快的优势。

DAG有向无环图，就是后一句的生成句子中汉字所有可能成词情况所构成的有向无环图，即就是给定一个待分词的句子，对这个句子进行生成有向无环图。切分步骤如下所示：

1. 根据dict.txt生成Trie树
2. 对待分词句子，根据dict.txt生成的Trie树，生成DAG，实际上通俗的说，就是对待分词句子，根据给定的词典进行查词典操作，生成几种可能的句子切分。

例如:{0:[1，2，3]} 这样一个简单的DAG，就是表示0位置开始，在1，2，3位置都是词，就是说0~1，0~2，0~3这三个起始位置之间的字符，在dict.txt中是词语。

* **采用动态规划查找最大概率路径并找出最大切分组合**

字典在生成Trie树的同时，也把每个词的出现次数转换为了频率。对于频率和概率，按照定义，频率其实也是一个0~1之间的小数，是事件出现的次数/实验中的总次数，因此在试验次数足够大的情况下，频率约等于概率，或者说频率的极限就是概率。

动态规划中，先查找待分词句子中已经切分好的词语，对该词语查找该词语出现的频率(次数/总数)，如果没有该词(既然是基于词典查找，应该是有的)，就把词典中出现频率最小的那个词语的频率作为该词的频率，也就是说P(某词语)=FREQ.get(‘某词语’，min\_freq)，然后根据动态规划查找最大概率路径的方法，对句子从右往左反向计算最大概率，因为汉语句子的重心经常落在后面，就是落在右边，因为通常情况下形容词太多，后面的才是主干。因此，从右往左计算，正确率要高于从左往右计算，这个类似于逆向最大匹配)，P(NodeN)=1.0，P(NodeN-1)=P(NodeN)\*Max(P(倒数第一个词))…依次类推，最后得到最大概率路径，得到最大概率的切分组合。

* **使用HMM模型和Viterbi算法对未登陆词进行识别**

给定一个待分词的句子，**利用HMM模型可以实现为其分词。**中文词汇按照BEMS四个状态来标记。对HMM(BEMS)四种状态的模型来说，就是为了找到一个最佳的BEMS序列，这个就需要使用Viterbi算法来得到这个最佳的隐藏状态序列。

通过训练得到的概率表和Viterbi算法，就可以得到一个概率最大的BEMS序列，按照B打头，E结尾的方式，对待分词的句子重新组合，就得到了分词结果. 比如对待分词的句子 ‘全世界都在学中国话’ 得到一个BEMS序列 [S，B，E，S，S，S，B，E，S]，通过把连续的BE凑合到一起得到一个词，单独的S放单，就得到一个分词结果了: 上面的BE位置和句子中单个汉字的位置一一对应，得到全/S 世界/BE 都/S 在/S 学/S 中国/BE 话/S 从而将句子切分为词语。

### 3.1.4.分词器附带功能

* **词性标注**

对分词后的词语应用N-gram语言模型，比较该词在上下文中为某些词性的概率，从而提取最大概率者作为该词词性。该步骤对于后续关键词抽取和停用词过滤的效果也具有至关重要的影响。

* **自定义词典载入**

可以指定自己自定义的词典，以便包含jieba词库里没有的词。虽然jieba有新词识别能力，但是自行添加新词可以保证更高的正确率。用户可行编添加新词到自定义词典中，调整频率，从而满足用户希望获取近期流行新词的诉求。

## 关键词提取算法

### TF-IDF算法介绍

我们选取TF-IDF算法作为关键词选取的主要算法，该算法在搜索引擎等实际应用中广泛使用，主要用以评估一个字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。

### TF-IDF算法使用

该算法以TF和IDF的乘积作为特征空间坐标系的取值测度，并用它完成对权值TF的调整，调整权值的目的在于突出重要单词，抑制次要单词，通过该方式，我们即可获取用户希望获得的关键词。TF—IDF计算公式如下所示：

在一个文本中出现很多次的单词，在另一个同类文本中出现次数也会很多，反之亦然。所以如果特征空间坐标系取TF词频作为测度，就可以体现同类文本的特点，所以引入了词频TF的概念。TF（Term Frequency），意为词频，用于计算该次描述文档内容的能力，词频计算公式如下所示：

Viiision—APP评论数据分析系统将一段时间内的所有评论内容放在一个文件中，因评论内容涵盖各个应用商店，可保证评论内容足够多，进而可类比于公式中的文章。

TF—IDF算法认为一个单词出现的文本频率越小，它区别不同类别的能力就越大，所以引入了逆文本频度IDF的概念。IDF（Inverse Document Frequency），意为“逆文档频率”，用于计算该词区分文档的能力。就是在词频的基础上，要对每个词分配一个"重要性"权重。最常见的词（如："的"、"是"、"在"）给予最小的权重，较常见的词（如："内容"）给予较小的权重，较少见的词（如："补光"、"妆容"）给予较大的权重。当一个词在这篇文档中出现的频率越高，同时在其他文档中出现的次数越少，则表明该词对于表示这篇文档的区分能力越强，所以其权重值就应该越大。IDF计算公式如下所示：

### TF-IDF算法优缺点分析

TF-IDF算法的优点是：简单快速，易于理解与操作，结果比较符合实际情况。

TF-IDF算法的缺点是：

1. 单纯以"词频"和“逆文档频率”衡量一个词的重要性，对于本项目Viiision—APP评论数据分析系统来说不够全面，会出现一些重要词因出现次数并不多而被忽略的情形。
2. 引入IDF调整权值的目的在于突出重要单词，抑制次要单词。实际上，IDF是一种试图抑制噪音的加权，并且单纯地认为文本频数小的单词就越重要，文本频数大的单词就越无用，显然这并不是完全正确的。
3. 在TF-IDF算法中并没有体现出单词的位置信息，特征词在不同的标记符中对文章内容的反映程度不同，其权重的计算方法也应不同，出现位置靠前的词与出现位置靠后的词，都被视为重要性相同，这是不完全正确的。

### TF-IDF算法改进措施

针对某些比较重要的词但是出现频率比较小的词，直接应用上述算法可能会被直接过滤，造成重要信息的遗漏。为了避免该问题的出现，我们选取比较重要的信息，将其列入专用词库中，别人为赋予一点的权重，这样保证其不会被过滤，但又保留了该词区分其他评论的特性。

## 垃圾评论过滤算法

### 垃圾评论分类

针对当前不同类型的垃圾评论，通常将垃圾评论分为内容型垃圾评论和欺诈型垃圾评论，依据本项目的实际倾向，本团队只考虑内容型垃圾评论。内容型垃圾评论通常指在评论内容中发布广告、评论内容与主题相关性低、在评论内容中发布WEB SPAM以及黄赌毒信息的评论。

内容型垃圾评论的检测主要考虑评论的内容特征和评论质量特征，将经过分析的特征训练分类模型作为依据，进行内容型垃圾评论的检测。其中评论内容的质量特征包括如下几个方面：1.评论内容与主题的相关性；2.评论内容中是否包含广告；3.评论内容的结构完整性和可读性。综合国内外研究现状，本项目组对内容型垃圾评论的特征进行归纳整合，常用特征集合如下表所示：

|  |  |
| --- | --- |
| 特征类别 | 特征类别描述 |
| 评论内容中情感词使用情况 | 评论内容中正、反向情感词个数比例 |
| 评论对象等关键词在评论内容中的体现 | 统计关键词在评论内容中出现的次数等信息 |
| 评论内容特征 | 评论中名词、形容词数量、副词的长度、数量和比特等特征 |
| 评论时间、评论时间的聚集程度 | 用户评价的整体表现形式，如评论的时间间隔均值和方差等特征 |
| 词频统计信息 | 评论中重要信息，如关键词的词频等特征 |

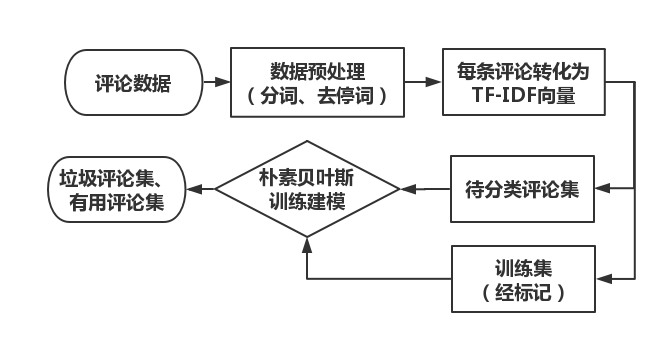
|  |
| --- |
| 筛选垃圾评论示例 |
| 手机簧网: KV444.com 美女脱衣，少妇自模出水。人与兽，教室诱惑，快来爽吧 |
| 本人🔞全露☝➕微❤a①⑤⑦⓪③⑨⑦⓪②①来👀打✈️爽坏你哦👠 |
| ?苹.果?用．户?.?⑧⭕块.钱-！ ☘ |
| →→→还下这个干嘛新版快播都出来了点赞呦吓载地址- ΚⅤ３２.соМ |
| 外贸高仿⥁原单A货⥁原单奢侈品 <http://t.cn/RMKH8Z0> 博柏利ᛪDiorᛪRolexᛪPradaᛪRolexᛪ路易威登ᛪ巴利ᛪ迪奥ᛪ |
| tyuijghjh(4537766676648)fctdftgtg. |
| 信用卡花呗套现秒到，VX加mushitaoxian |
| ☝招，APP试用员?完全免费?熟练后每天几分钟?轻松月入几百，做的好更多?请看昵称++? 点赞有福利 |

### 基于朴素贝叶斯的垃圾评论分类算法

通过使用有监督学习的朴素贝叶斯二分类器可区分有用评论与垃圾评论，进而过滤垃圾评论。该方法是在分布独立这个假设成立的情况下实现，而在该文本数据中，分布独立的假设基本成立。该方法过程简单速度快，分类效果好。

### 垃圾评论检测框架设计

内容型垃圾评论的检测中，首先通过停用词库、情感词库和后缀词库对数据集进行预处理；然后将每一句评论使用TF-IDF算法进行特征的选择和提取，并转化为可计算的TF-IDF向量。将训练集使用朴素贝叶斯分类器进行训练建模。对于待检测的评论，将其置于模型中计算出分类概率，若为垃圾评论的概率大于1/2，标记为垃圾评论。



# 关键词提取技术

### 针对不同部门的关键词分类

由于不同的部门对于评论信息有不同方面的需求，因此在在针对不同部门用户进行关键词展示需使用不同的侧重词集，这些文字集大部分可通过网络获取。这里我们针对「摄影工具」类对不同的部门整理关键词，具体关键词如下表所示：

|  |  |  |
| --- | --- | --- |
| 部门 | 类别 | 具体关键词 |
| 营销策划部门 | 活动类关键词 | 购买商品打折优惠抢购余额购物车交易理财钱包免费会员会员卡奖品骗子支付红包奖励福利消费付费价格特价业务抽奖活动贵族赠送中秋春节五一代言玩家模式玩法券券代言女神榜首 |
| 开发部 | 功能类关键词 | 美颜软件下载推荐功能自拍建议发现使用评论微信分享关注滤镜使用设计电话推荐评价表情美化自拍界面保存视频使用效果神器大头贴化妆朋友圈游戏整容图片皮肤相片一键磨皮动画补光妆容装饰素材照相机风景手绘漫画网红脸型水印静音 |
| 产品部 | 问题类关键词 | 下载更新手机广告垃圾骗子问题评价内存版本黑屏无聊死机运行封号注册错误登录乱反馈商家客服像素习惯时间画面团队内容太假权限差评信息总会速度不卡卡真卡 |
| 全部部门 | 情感程度关键词 | 好真好很好挺好有点超棒挺棒棒极了真棒不错呀不错不错呀给力好玩 |

### 针对不同部门的关键词提取

对于不同部门的用户，加载各自所属的关键词备选表，使用TF-IDF算法提取排名前8位词，并按照权重进行排序。

这里我们将「美颜相机」于近一个月在各平台上的全部评论数据进行针对不同部门的关键词提取，详细结果如下表：

|  |  |
| --- | --- |
| 关键词 | 权重 |
| 美颜 | 0.279267 |
| 软件 | 0.122494 |
| 下载 | 0.083518 |
| 自拍 | 0.052284 |
| 效果 | 0.050256 |
| 推荐 | 0.035647 |
| 神器 | 0.029505 |
| 大头贴 | 0.027193 |

功能类关键词筛选

|  |  |
| --- | --- |
| 关键词 | 权重 |
| 垃圾 | 0.173095 |
| 手机 | 0.114964 |
| 广告 | 0.070160 |
| 像素 | 0.040791 |
| 版本 | 0.035307 |
| 内存 | 0.026239 |
| 习惯 | 0.025426 |
| 问题 | 0.021667 |

问题类关键词筛选

### 自定义分词的添加

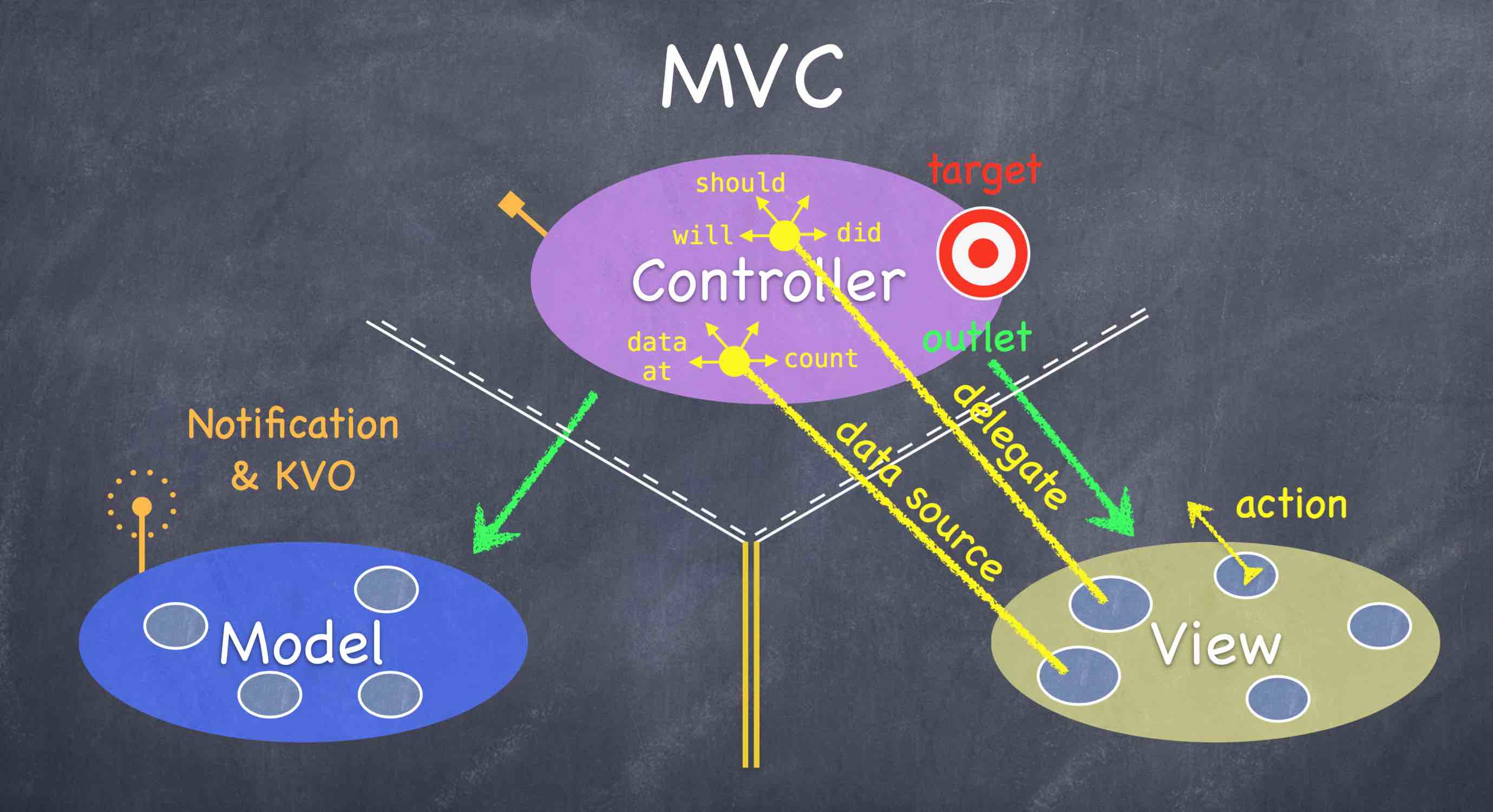
由于默认提供的关键词库存在一定的局限性，同时每一阶段会出现网络新词以及新的需求，因此按照原有词库进行关键词提取并不能满足用户需求。因此对用户提供了自定义添加分词的入口，用户可安装自己的需求添加新词至词库中。经重新关键词提取后，将优先展示。

# 软件技术

## Laravel框架

### 模型—视图—控制器

Laravel是Model-View-Controller(MVC)架构模式，它使用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。



### 响应流程

当访问一个laravel应用程序时，浏览器发送一个请求，由Web服务器接收并传递到laravel的路由引擎。该laravel路由器接收到请求后，根据配置重定向到相应的控制器类的方法。

然后由控制器类接管。在某些情况下，控制器将立即渲染一个视图，这是一个模板，将被转换成HTML并且发送回浏览器。更普遍的动态网站，控制器与模型进行交互，与数据库进行通信。调用模型后，控制器呈现最终视图(HTML、CSS和图像)并返回完整的web页面到用户的浏览器。

Laravel提倡模型、视图和控制器，应保持完全独立的存储单独的文件在不同的目录，这就是laravel的目录结构发挥作用的地方。

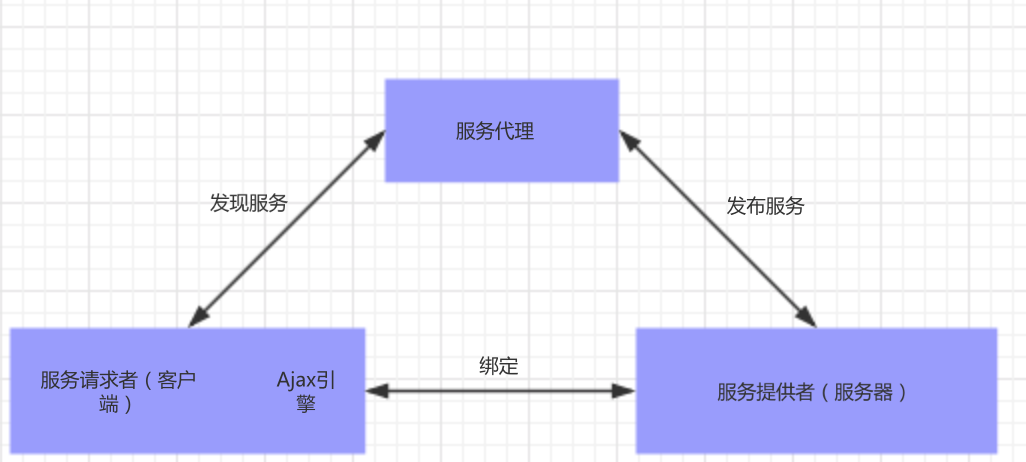
## Nginx服务器

Nginx服务器是一款轻量级的Web服务器和电子邮件代理服务器，并在一个BSD-like协议下发行其稳定性好、功能集丰富、系统资源低、占有内存少，并发能力强，且其并发能力在同类型的网页服务器中表现突出。

## 基于Ajax技术的Web服务架构

在传统的Web服务模式下，用户和服务器之间是一种同步关系，服务器在处理请求的时候，用户多数时间只能等待，限制了交互性，用户体验较差。基于Ajax技术的Web服务架构为浏览器提供了与服务器端异步通信的能力，可以实现页面的局部刷新而不是加载整个页面，减少了用户等待的时间，更好的满足了用户需求，使得Web应用程序更加人性化。

Ajax即“Asynchronous JavaScript and XML”，是一种创建交互式网页应用的网页开发技术。Ajax技术实现过程是，Web页面中的JavaScript脚本使用XM-LHTTP Request对象与服务器异步通信，服务器接收请求后返回业务数据；数据通过脚本程序处理后，经过数据可视化技术更新显示在Web页面中。这种异步数据读取方法使Ajax可以自主的发起Web请求，与远端服务器完成必要的数据交互，在构建Web页面时，无需中断交互流程即可重新加载和动态更新，既减轻了服务器负载又加快了响应速度，缩短了用户等待的时间。



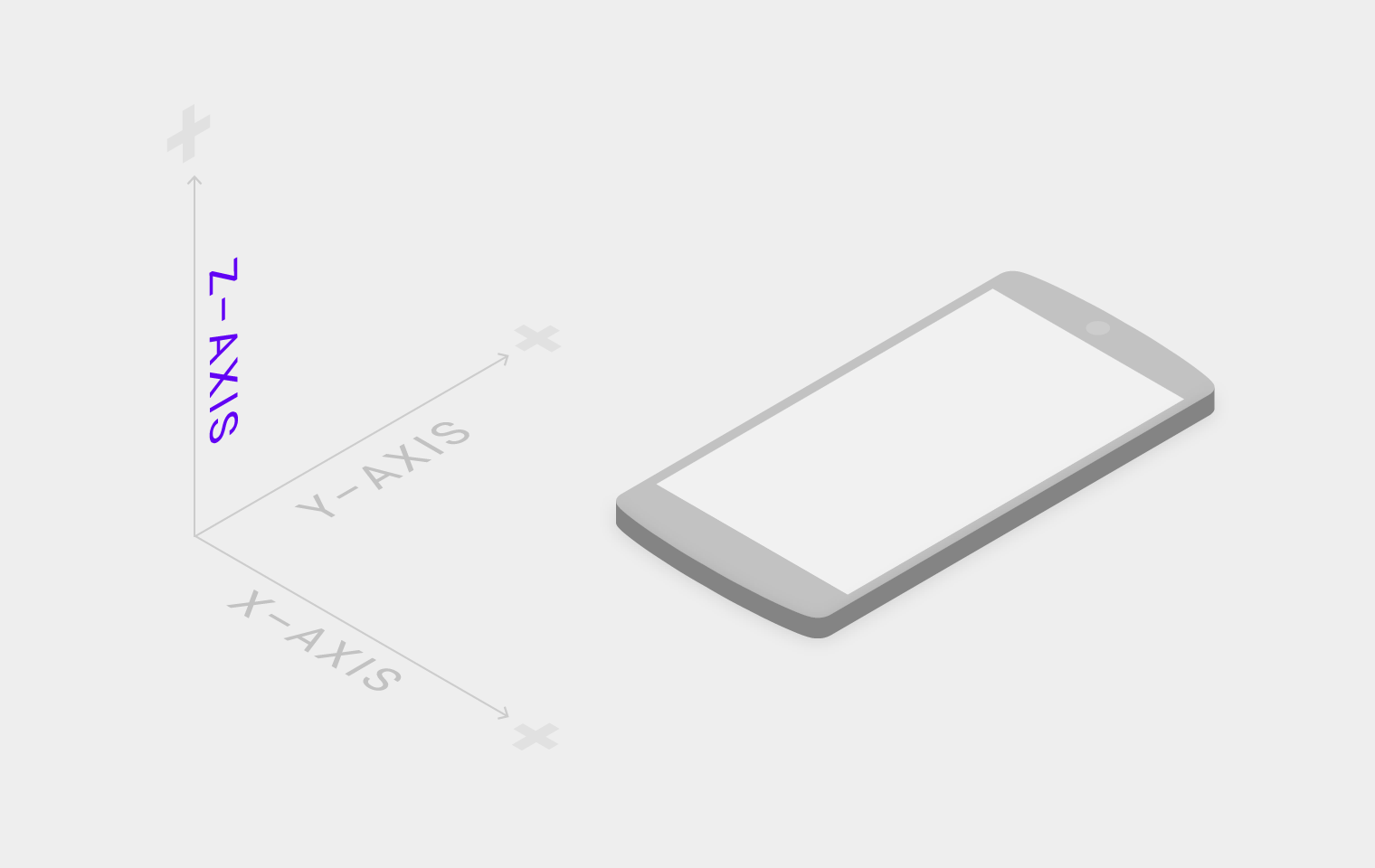
图基于Ajax技术的Web服务架构

本项目应用此技术可以在不刷新全部页面的情况下加载app的数据，用户可以方便快捷的查看到app在不同时间段内的各种情况，以及不同筛选条件下的评论情况，提高了数据加载的速度，增加了用户体验。

# 前端交互技术

## 框架选择

Viiision—APP评论数据分析系统前端设计框架采用的是Google推出的一款重视跨平台体验的设计语言——Material design。



Material design期望的是把物理世界的体验带进屏幕，去掉现实中的杂质和随机性，保留其最原始纯净的形态、空间关系、变化与过渡。Viiision所做的也是最直观的数据体现，浸入式的数据分析氛围，配合Material design构建的虚拟世界的灵活特性，还原最贴近真实的体验，达到简洁与直观的效果。

## 产品颜色设计

* **原色**

Viiision—APP评论数据分析系统应用程序屏幕和组件中最常显示的颜色为一种优雅的紫色，它是我们的原色。

同时，为了创建元素之间的对比度，我们使用较浅或较深色调的原色，增添明度和饱和度的变化，较浅和较暗色调之间的对比有助于显示表面之间的划分，例如状态栏和工具栏之间的划分。

* **次要颜色**

辅助颜色用于重写UI的部分选择。它是原色紫色的互补色，一种荧光绿。它与周围的元素形成对比，突出强调。

****

## ****响应式布局****

Material design要求这些布局能够适应任何屏幕尺寸。为了简化适用于各种尺寸的分屏模式的应用程序，我们最先设计最小的尺寸。利用栅格化样式，确保布局之间的一致性。

* **布局摘要及详细视图内容**

600dp以下的布局可以用单级内容层次结构填充屏幕（摘要或详细内容，但不同时使用）。

超过600dp的布局可以在屏幕上放置两个层次的内容层次结构（摘要和详细内容）

* **最大屏幕宽度**

1600dp宽的布局可能会使布局增长，直到达到最大宽度。此时，栅格会执行以下操作：

1. 成为中心与边缘增加一致
2. 保持左对齐，右边距增长
3. 继续增长，同时显示其他内容

****

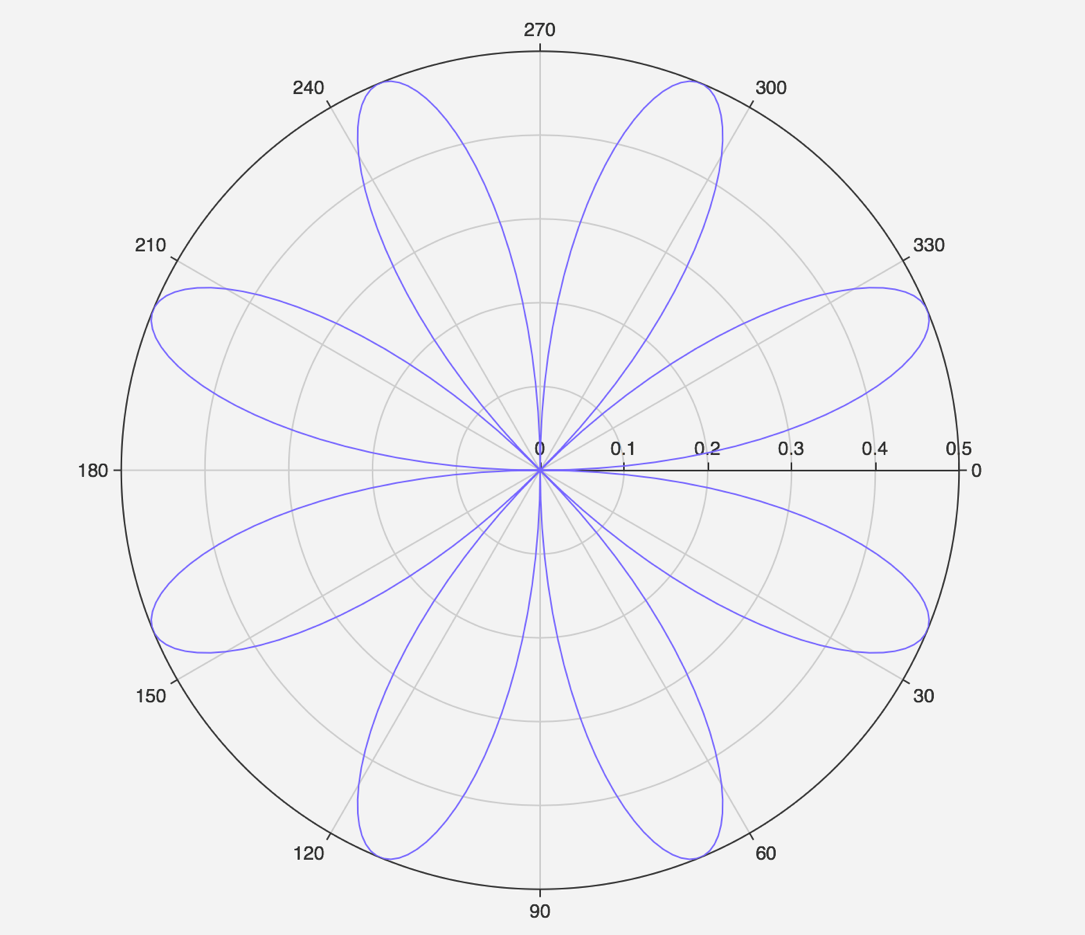
# 数据可视化技术

数据可视化是利用计算机图形学的图像处理技术，将数据转换成图形或图像在屏幕上显示出来，并进行交互处理的理论、方法和技术。

## 基于几何的技术

基于几何的可视化技术包括Scatter plots、Landscapes、Projection Pursuit、Parallel Coordinates等等，是以几何画法或几何投影的方式来表示数据库中的数据。平行坐标法是最早提出的以二维形式表示n维数据的可视化技术。它的基本思想是将n维数据属性空间通过n条等距离的平行轴映射到二维平面上，每一条轴线代表一个属性维，轴线上的取值范围从对应属性的最小值到最大值的均匀分布。这样，每一个数据项都可以根据其属性值用一条折线段在n条平行轴上表示出来。

利用这个技术加上d3.js我们可以设计出符合要求的x，y轴，图表可以跨坐标系存在，例如折、柱、散点等图可以放在直角坐标系上，也可以放在极坐标系上，甚至可以放在地理坐标系中。以及合适的数据范围和比例尺。



极坐标双数值设计实例

## 面向像素技术

面向像素技术的基本思想是将每一个数据点的数据值对应于一个带颜色的屏幕像素，对于不同的数据属性以不同的窗口分别表示（图2）。面向像素技术的特点在于能在屏幕中尽可能多的显示出相关的数据项，对于高分辨率的显示起来说，可实现多达数量级的数据。

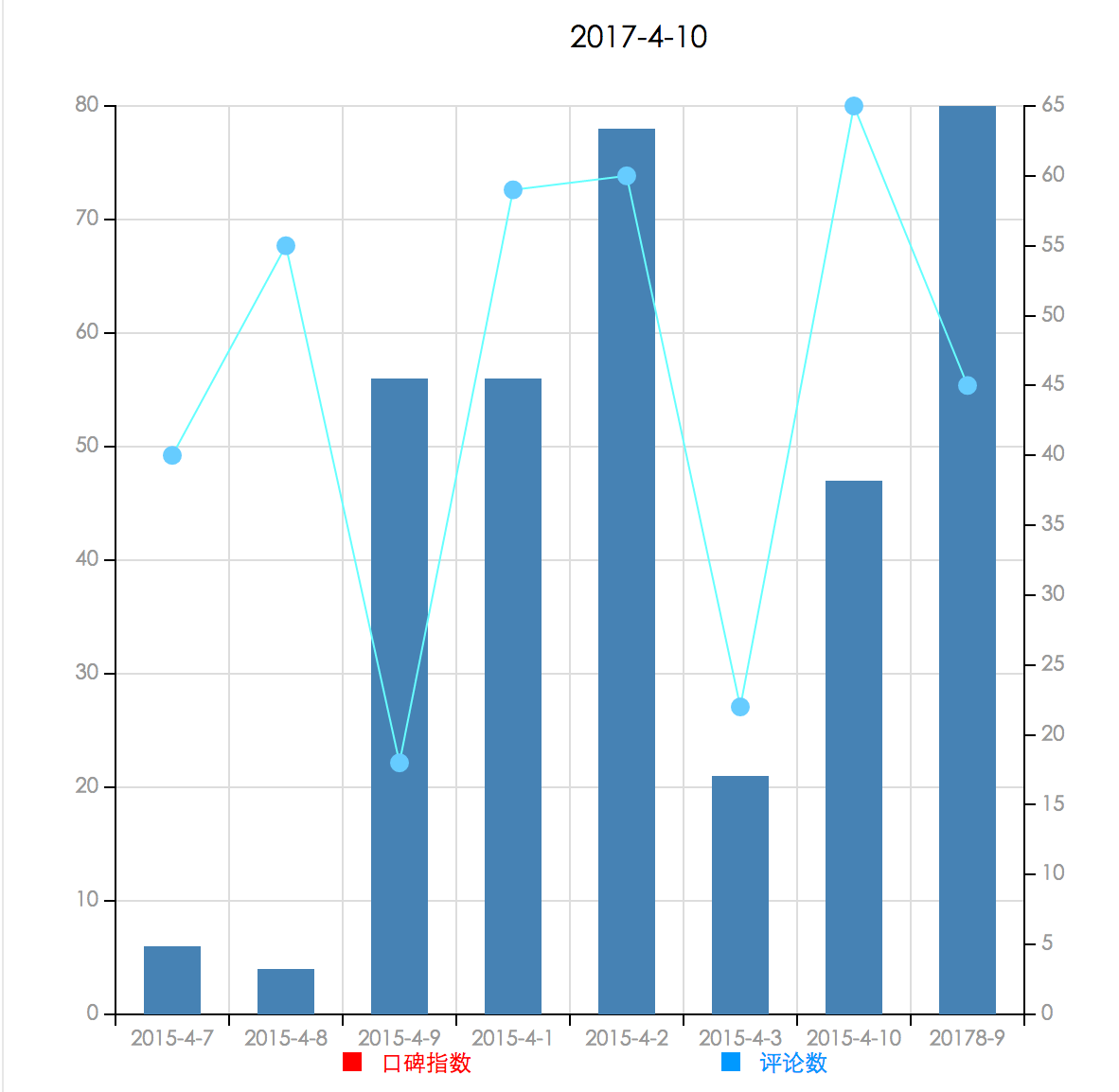
本项目收集的评论数据以及下载量等数据少则几万条，多则几十万，利用此技术可以将数量级很大的数据清晰的显示在图表中，尽可能的保持数据的完整性。

## 基于图标的技术

基于图标技术的基本思想是用一个简单图标的各个部分来表示n维数据属性。基于图标的可视化技术包括Chernoff-face、Shape Coding、Stick Figures等，这种技术适用于某些维值在二维平面上具有良好展开属性的数据集

枝形图方法是其中的基本方法之一。枝形图方法首先选取多维属性中的两种属性作为基本的X-Y平面轴，在此平面上利用小树枝的长度或角度的不同表示出其他属性值的变化。

本项目利用多维属性在一个图表中表示出一个或多个app的多个属性，可以使用户在短时间内有效快速的获得app的不同方面的信息。例如下图所示的两个数据点，它们对左边的二维属性含有相同的数据值，而右边的二维属性的数据值则不相同。

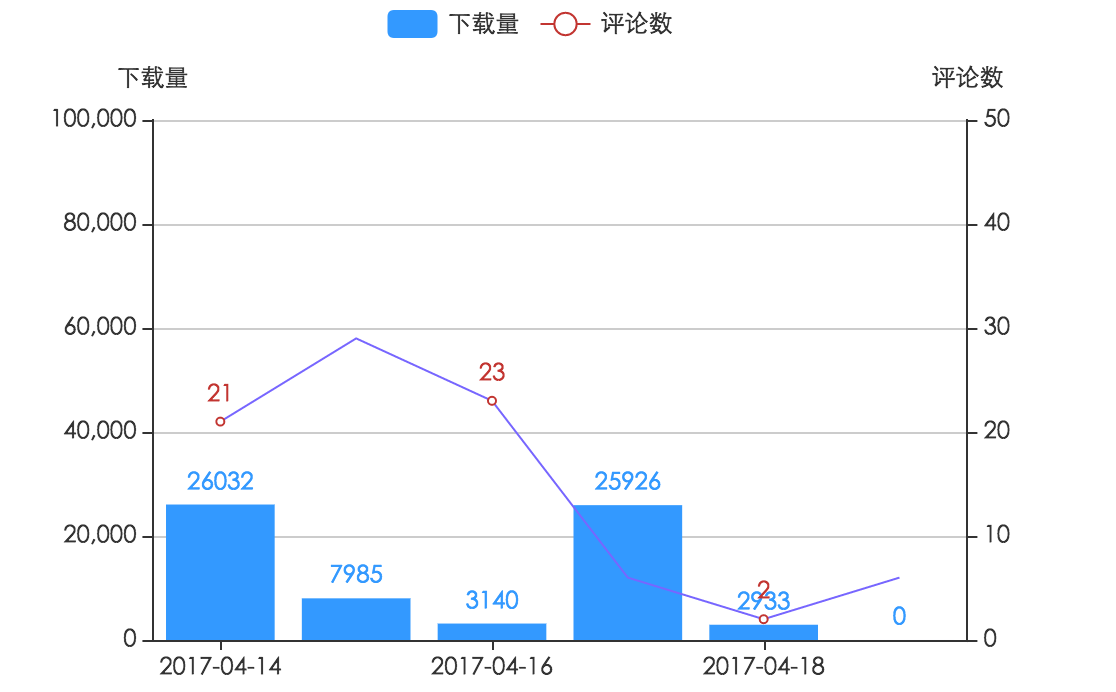


## Echarts技术

ECharts，一个纯Javascript的图表库，可以流畅的运行在 PC 和移动设备上，底层依赖轻量级的 Canvas 类库ZRender，提供直观，生动，可交互，可高度个性化定制的数据可视化图表，更是加入了更多丰富的交互功能以及更多的可视化效果，并且对移动端做了深度的优化。

## 独立坐标系

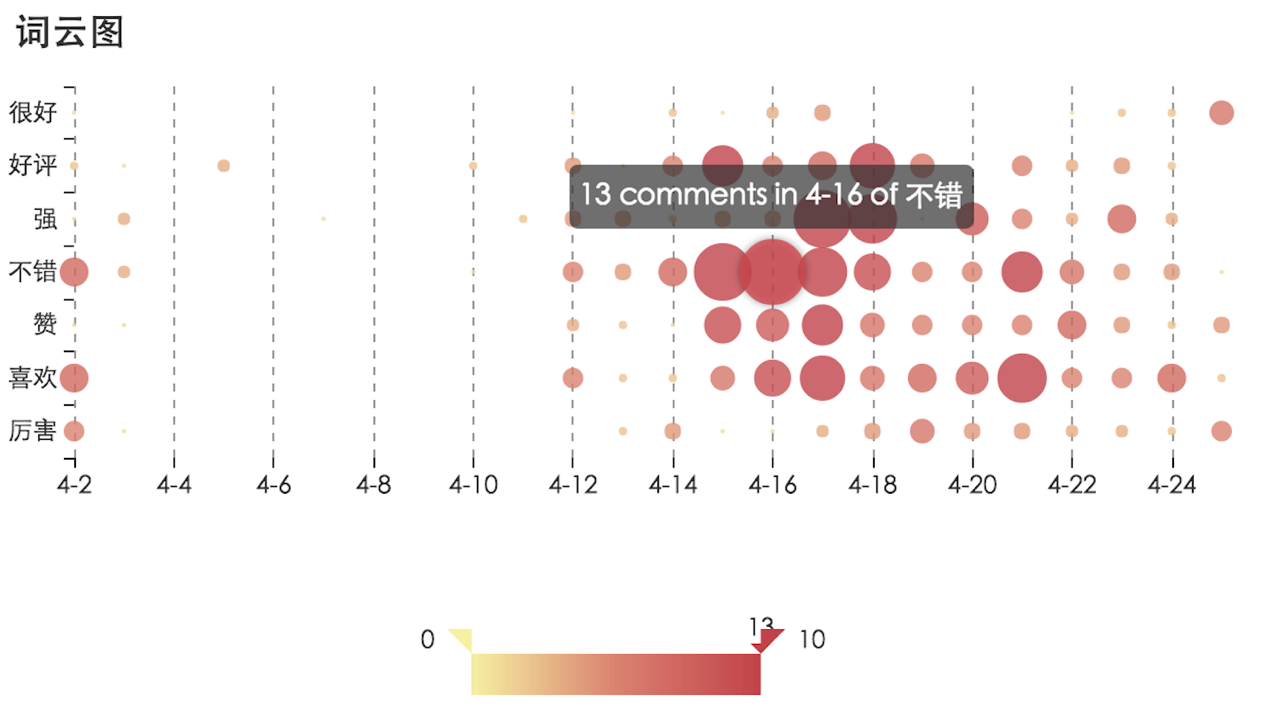
支持直角坐标系（catesian，同 grid）、极坐标系（polar）、地理坐标系（geo）。图表可以跨坐标系存在，例如折、柱、散点等图可以放在直角坐标系上，也可以放在极坐标系上，甚至可以放在地理坐标系中。



## 多维数据支持和丰富视觉编码

除了加入了平行坐标等常见的多维数据可视化工具外，对于传统的散点图等，传入的数据也可以是多个维度的。配合视觉映射组件 Visual Map提供的丰富的视觉编码，能够将不同维度的数据映射到颜色、大小、透明度、明暗度等不同的视觉通道。

我们可以在图表中加入视觉组件增加用户体验，当鼠标移动到相应位置时，会提示具体的数据，以及数据的权重大小会根据颜色的深浅更加直观的视觉输出。图表的控件可以将图表中的数据通过表格一键呈现，还可以将图表一键导出为图片，便于后续的整理以及邮件等的发送。



图多维数据及视觉特效

# 数据库技术

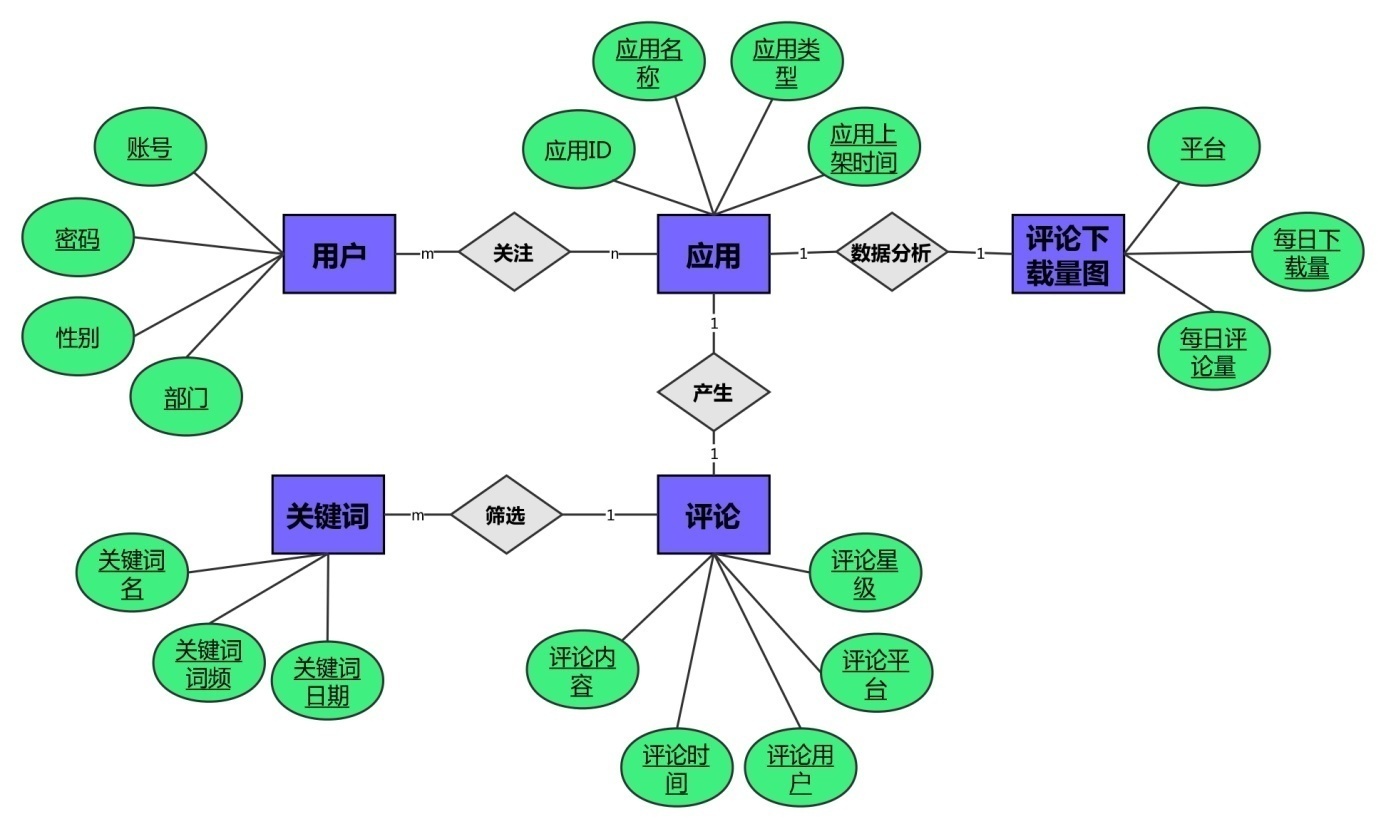
## 可信任的数据库

MySQL是一个关系型数据库管理系统，由瑞典MySQL AB 公司开发，目前属于 Oracle 旗下产品。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL是最好的 RDBMS (Relational Database Management System，关系数据库管理系统) 应用软件，也是最适配于PHP框架Laravel的数据库。

MySQL具有如下特性：

1. 使用C和C++编写，并使用了多种编译器进行测试，保证源代码的可移植性。
2. 支持AIX、BSDi、FreeBSD、HP-UX、Linux、Mac OS、Novell NetWare、NetBSD、OpenBSD、OS/2 Wrap、Solaris、Windows等多种操作系统。
3. 为多种编程语言提供了API。这些编程语言包括C、C++、C#、VB.NET、Delphi、Eiffel、Java、Perl、PHP、Python、Ruby和Tcl等。
4. 支持多线程，充分利用CPU资源，支持多用户。
5. 既能够作为一个单独的应用程序在客户端服务器网络环境中运行，也能够作为一个程序库而嵌入到其他的软件中。
6. 提供多语言支持，常见的编码如中文的GB 2312、BIG5，日文的Shift JIS等都可以用作数据表名和数据列名。
7. 提供TCP/IP、ODBC和JDBC等多种数据库连接途径。
8. 提供用于管理、检查、优化数据库操作的管理工具。
9. 可以处理拥有上千万条记录的大型数据库。

## 数据库表关系



## 数据库表详细设计

|  |  |  |
| --- | --- | --- |
| app |  |  |
| 字段名 | 数据类型 | 备注 |
| app\_id (PK) | int | 应用ID |
| app\_name | varchar | 应用名称 |
| app\_category | varchar | 应用类型 |
| app\_date | date | 应用上架日期 |

|  |  |  |
| --- | --- | --- |
| app\_info |  |  |
| 字段名 | 数据类型 | 备注 |
| app\_info\_id(PK) | int | 信息记录ID |
| app\_id (FK) | int | 应用ID |
| app\_platform | varchar | 应用所在平台 |
| app\_download\_count | int | 应用下载量 |
| app\_comment\_count | int | 应用评论量 |
| app\_date | date | 数据对应日期 |

|  |  |  |
| --- | --- | --- |
| app\_comment |  |  |
| 字段名 | 数据类型 | 备注 |
| comment\_id(PK) | int | 评论ID |
| app\_id(FK) | int | 评论来源应用的ID |
| name | varchar | 评论用户名 |
| content | varchar | 评论内容 |
| score | int | 用户打分 |
| date | date | 评论日期 |
| platform | varchar | 评论来源平台 |

|  |  |  |
| --- | --- | --- |
| keywords |  |  |
| 字段名 | 数据类型 | 备注 |
| keyword\_id(PK) | int | 分词ID |
| app\_id (FK) | int | 应用ID |
| keyword\_name | varchar | 关键词名 |
| weight | double | 关键词权重 |
| keyword\_date | date | 关键词所属日期 |

|  |  |  |
| --- | --- | --- |
| user |  |  |
| 字段名 | 数据类型 | 备注 |
| user\_id(PK) | int | 用户ID |
| password | varchar | 用户密码 |
| sex | char | 用户性别 |
| class | varchar | 用户权限 |

|  |  |  |
| --- | --- | --- |
| user\_app |  |  |
| 字段名 | 数据类型 | 备注 |
| user\_app\_id(PK) | int | 用户评论关系ID |
| user\_id(FK) | int | 用户ID |
| app\_id(FK) | int | 应用ID |

# 系统实现

## 系统用户端实现

## 系统管理员端实现