中图分类号: 学校代码: 10055
UDC: 密级: 公开

南開大學

硕 士 学 位 论 文

基于互联网日志的用户行为分析算法和用户模型的研究

Behavior analysis and user modeling based on Internet access logs

论文作者 孙卓豪 指导教师 章辉

申请学位 学士 培养单位

学科专业 电子信息科学与技术 研究方向

答辩委员会主席 评 阅 人

南开大学研究生院

二〇一六年五月

南开大学学位论文使用授权书

根据《南开大学关于研究生学位论文收藏和利用管理办法》，我校的博士、硕士学位获得者均须向南开大学提交本人的学位论文纸质本及相应电子版。

本人完全了解南开大学有关研究生学位论文收藏和利用的管理规定。南开大学拥有在《著作权法》规定范围内的学位论文使用权，即：(1) 学位获得者必须按规定提交学位论文(包括纸质印刷本及电子版)，学校可以采用影印、缩印或其他复制手段保存研究生学位论文，并编入《南开大学博硕士学位论文全文数据库》；(2) 为教学和科研目的，学校可以将公开的学位论文作为资料在图书馆等场所提供校内师生阅读，在校园网上提供论文目录检索、文摘以及论文全文浏览、下载等免费信息服务；(3) 根据教育部有关规定，南开大学向教育部指定单位提交公开的学位论文；(4) 学位论文作者授权学校向中国科技信息研究所和中国学术期刊 (光盘) 电子出版社提交规定范围的学位论文及其电子版并收入相应学位论文数据库，通过其相关网站对外进行信息服务。同时本人保留在其他媒体发表论文的权利。

非公开学位论文，保密期限内不向外提交和提供服务，解密后提交和服务同公开论文。

论文电子版提交至校图书馆网站：http://202.113.20.161:8001/index.htm。

本人承诺：本人的学位论文是在南开大学学习期间创作完成的作品，并已通过论文答辩；提交的学位论文电子版与纸质本论文的内容一致，如因不同造成不良后果由本人自负。

本人同意遵守上述规定。本授权书签署一式两份，由研究生院和图书馆留存。

作者暨授权人签字：_____

20　　年　　月　　日

南开大学研究生学位论文作者信息

论 文 题 目	基于互联网日志的用户行为分析算法和用户模型的研究				
姓　　名	孙卓豪	学号	1210403	答辩日期	
论 文 类 别	博士 □　学历硕士 □　硕士专业学位 □　高校教师 □　同等学力硕士 □				
院 / 系 / 所			专　　业		
联 系 电 话			**Email**		
通讯地址 (邮编)：					
备注：					

注:本授权书适用我校授予的所有博士、硕士的学位论文。由作者填写 (一式两份) 签字后交校图书馆，非公开学位论文须附《南开大学研究生申请非公开学位论文审批表》。

南开大学学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师指导下进行研究工作所取得的研究成果。除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人创作的、已公开发表或者没有公开发表的作品的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

学位论文作者签名：_____ 年 月 日

非公开学位论文标注说明

根据南开大学有关规定，非公开学位论文须经指导教师同意、作者本人申请和相关部门批准方能标注。未经批准的均为公开学位论文，公开学位论文本说明为空白。

论文题目			
申请密级	□ 限制 (≤2 年) □ 秘密 (≤10 年) □ 机密 (≤20 年)		
保密期限	20 年 月 日至 20 年 月 日		
审批表编号		**批准日期**	20 年 月 日

南开大学学位办公室盖章 (有效)

限制 □ 2 年（最长 2 年，可少于 2 年）

秘密 □10 年（最长 5 年，可少于 5 年）

机密 □20 年（最长10年，可少于10年）

中文摘要

随着互联网的高速发展，用户留下越来越多的网络行为信息。基于这些用户信息，分析用户行为和研究用户模型，研发适合海量数据的自然语义分析、文本挖掘、位置分析等算法对预测用户行为及商品推荐至关重要。本文针对大数据精准营销的难点，开展数据挖掘技术、模型与算法、场景化实践的实证研究，构建面向新媒体业务的场景化智能营销平台，并针对新媒体公司自有海量数据的特征与需求进行系统优化，提高资源利用率及推荐效果，从而全面提升公司的用户运营和内容运营能力。

关键词： 海量数据处理；自然语义分析；数据挖掘

Abstract

Today Internet users, typically 4G mobile users, leave billions tons of data on Servers. Base on such big data, it's urgent to develop algorithm about natural semantic analysis, text mining to predict users' behavior and recommand related content. This paper focus on big data analysis, develop some data mining algorithm and build a bunch of system to pop up Internet company's profit.

Key Words: Big data; natural semantic analysis; data mining

目录

第一章　背景及研究现状

第一节　背景

随着互联网的普及，人们每天花费越来越多时间通过网络工作，娱乐。目前，得益于芯片技术极速发展，比尔盖茨的目标，“地球上每个人都有个人电脑”，已基本实现，其中有 40% 的电脑连接着互联网；全球大概 30 亿人持有计算能力相当于 1980 年房间般大小的超级计算机的智能手机，这些移动手机用户使用各地运营商大力推广的 4G 上网服务，实现随时随地上网的梦想。除了电脑和手机，人们还将可穿戴设备及各类物联网接入互联网中。因此，互联网各网络节点每天都能采集到大量用户的上网信息。基于这些用户信息，分析用户行为和研究用户模型，我们能挖掘出用户的兴趣爱好，由此推荐相关的内容，从而提高互联网的交互能力以及网络运营商内容运营能力。

在多篇论文中，研究人员已经对用户行为进行了细致的研究。比如使用 K-means 距离算法 [1] 把行为类似的用户分为几组；又如研究上下文推断用户连续性爱好 [2]; 也有使用支持向量机给用户类型分类的 [3]；还有研究用户偏好随群体交流改变的 [3, 4]。根据协同过滤算法 [5]，用户行为的建模可以描述为一个协同过滤中的 User-Item 矩阵。为了得到这个矩阵，本文要实现一个协同过滤的推荐系统。

目前，大量互联网公司采用推荐系统给用户推荐相关的内容。全球最大网上购物平台 Amazon 最早将推荐系统引入网络销售领域 [6]。之后，推荐系统被应用于各种类型的互联网服务中。比如影视服务方面，视频网站 Netflix 将推荐系统与 A/B testing 结合起来 [7]。Google 旗下的 YouTube 给用户推荐视频 [8]。

目前推荐系统的主要使用协同过滤算法。有两种等效的协同过滤算法，一种是基于用户的协同过滤，另一种是基于物品的协同过滤。基于用户的协同过滤算法通过对相似用户的评分求和来预测用户对目标物品的评分 [5]。基于用户协同过滤算法是该类算法的最简形式 [9]，它能推广到其他复杂模型，比如引入细颗粒度的邻近权重因子 [10]，迭代寻找邻近方法 [11]，或者基于用户的个人资料计算用户相似度 [12]。

本文所介绍的系统从互联网提供商（ISP）的流量分析入手。由于大多数用户通过 ISP 上网，ISP 的节点路由或者交换机能采集到用户访问各种类型的网站的流量。这与之前提到的公司不同，他们只关注与公司业务相关的流量。从 ISP 的角度看，我们可以全天候地提供全方位的推荐。

由 Google 带领的全球互联网进入 HTTPS 时代，许多互联网服务使用 HTTPS，即将网页等网络信息进行 TSL 或者 SSL 加密传输。这就意味着许多网络数据虽然流经 ISP 的路由或者交换机，但是 ISP 无法查看数据包中的内容。但是目前作为领导者的 Google 承认自己有 25% 的数据传输没有加密。并且，日前以色列大学的研究团队最近发表一篇论文显示他们使用统计计算方法分析 https 加密流量，能够以 96% 的准确性确定用户的操作系统、浏览器以及他们正在浏览的网页。这就表明有大部分的网络数据是可以读取的。

本文就从学术的角度，探讨推荐系统实现的可行性及其中的关键技术。

第二节　本文结构

本文主要目标是研究用户模型，得出代表用户模型的 User-Item 矩阵，这些都基于本文提出的基于大数据分析的推荐系统。其中第一章主要介绍了相关背景及研究现状。第二章主要介绍相关算法及简单叙述 Hadoop 分布式数据处理的概念。第三章主要介绍用户上网数据采集，网站语义分析，数据处理及存储，基于 Hadoop 的推荐算法等模块的工作细节。第四章主要介绍整体系统工作流程，以及实证分析。第五章总结以上内容。

第二章　相关算法及 Hadoop 分布式数据处理

本文提出的推荐系统基于语义分析、协同过滤算法、Hadoop 和 MapReduce 等几个基本算法及基本概念。以下简单介绍这几个算法及概念。

第一节　语义分析

自然语言分析包含以下主要步骤：

1. 句尾检测

   这一步主要是把一段文字分成一个有意义句子的集合 [13]。因为句子一般都含有思想的逻辑单元，所以句子的语法是可以预测的。而切词工作是建立在单个句子上的，所以分析文章一般从句尾检测开始。

2. 切词

   这一步主要在单个句子上进行操作，将句子分为单个词。切词工作对于不同的文字有不同的切法，比如英文切词工作主要按空格切分，但还要注意不能混淆点号的意义；而中文没有空格，切词时要按照语料库是否存在该词来切分。

3. 词义类型标记

   这一步主要是在切好的词语中标记词语的类型，比如是动词还是名词或者是长名词中的一部分。对于社交网络中的语言，比如推特或者微博，语言比较口语化，会出现大量动词名词混用情况 [14–16]。通过标记，机器甚至能指出名词或者是长名词是属于什么类型，比如是人，地点，或者组织。

4. 分块

   这一步主要是在有逻辑意义的整句和复合标记的基础上分析词语。根据预先定义好的语法，将标记了类型的词语分成块。

5. 提取主要内容

   这一步主要是进一步为分好的块标记名称或者类型，这与第三步词义类型标记不同，因为这一步以块为整体标记名称或者类型。

其中每一步都很关键，因为一旦其中一步出错，误差会逐步传播，使得语

义分析出严重错误。下面我们介绍计算自然语言分析准确度的方法。在语义分析完成后，统计以下数量：

TP 被正确识别为实体的词语

FP 本来不应该被正确识别为实体的但确实被正确识别为实体的词语

TN 本来不应该被正确识别为实体的，同时没有被正确识别为实体的词语

FN 本来应该被正确识别为实体的，但没有被正确识别为实体的词语

此时再定义精确度与召回率如下：

精确度 $precision = \frac{TP}{TP+FP}$

召回率 $recall = \frac{TP}{TP+FN}$

则可以用下式来描述语义分析模型的准确度为

$$F = 2 * \frac{precision * recall}{precision + recall}$$

第二节 协同过滤算法

协同过滤算法基于三个假设：人们有共同的兴趣爱好；他们的兴趣爱好是稳定的；可以根据兴趣爱好的历史数据预测用户的选择。协同过滤算法主要比较用户之间的行为，找到最近的相邻用户，根据最近的相邻用户的行为预测用户的兴趣爱好。

对于 m 个用户的用户集 U，和 n 个物品的物品集 I，用户 $u_i \in U$ 与物品 $i_j \in I$ 的交互信息记录为一个数值 $r_{ij} \in R$。若用户 i 对物品 j 无交互行为，则 r_{ij} =?。最基本的协同过滤算法所研究的问题为：基于 R，给每位用户推荐一个不包含该用户已经评价过物品的列表，其中的物品按照符合用户兴趣程度递减排序。

协同过滤算法能分为主要两个步骤：第一步是计算用户 x 与用户 y 之间的相似程度：

$$S_{x,y} = \frac{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r}_x)(r_{y,i} - \bar{r}_y)}{\sqrt{\sum_{i \in I_{xy}} (r_{x,i} - \bar{r}_x)^2} \sqrt{\sum_{i \in I_{xy}} (r_{y,i} - \bar{r}_y)^2}}$$

其中 x,i 为用户 x 对物品 i 的评分，$\bar{r}_x$ 是用户 x 的平均评分，I_{xy} 为与用户 x 和用户 y 曾发生交互行为的物品集。

第二步计算用户 u 对物品 i 评分的预测值 $r_{u,i}$：

$$r_{u,i} = \bar{r}_u + \frac{\sum_{u' \in U^k} (r_{u',i} - \bar{r}_{u'}) S_{u,u'}}{\sum_{u' \in U^k} S_{u,u'}}$$

其中 U^k 为与用户 u 距离最近的 n 个用户。最后，只需从物品集中筛选出预测分数前几的物品，推荐给用户 u。

第三节 Hadoop 和 MapReduce

Google 于 03 至 06 年公布了三篇论文，描述了 GFS、BigTable、MapReduce 三种技术以解决这些问题 [17–19]。由于 Google 并没有公布算法细节，因此由雅虎牵头，在 06 年左右建立了开源项目 Hadoop，目的是根据 Google 的三篇论文，实现一个大规模的管理计算系统。

Hadoop 的基础组件是 Hadoop 分布式文件系统 (HDFS)。HDFS 的机制是将大量数据分布到计算机集群上，数据一次写入，但可以多次读取用于分析。Hadoop 的主要执行框架即 MapReduce。对于保存在分布式文件系统的数据，MapReduce 在每个数据节点上进行本地运算。

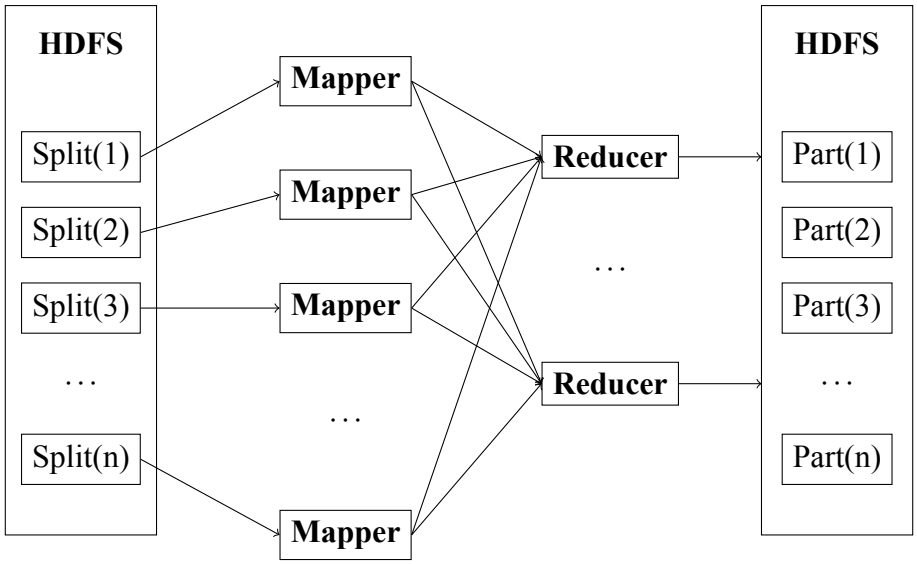


图 2.1 Hadoop 的 MapReduce 工作流程图

MapReduce 框架包含两个阶段，map 阶段和 reduce 阶段。输入数据和计算后的输出数据都是一个 (key,value) 集合。用 $< k, v >$ 来表示这些键值对。key 主要用在 reduce 阶段，用来决定哪个 value 结合在一起。用两个函数来表示这两个过程：

$$\textbf{Mapper} \quad : \quad < k1, v1 > \rightarrow list < k2, v2 > \tag{2.1}$$

$$\textbf{Reduce} \quad : \quad < k2, list < v2 > > \rightarrow list < k3, v3 > \tag{2.2}$$

计算过程从 map 阶段开始，其中并行执行 map 函数同时把保存在分布式文件系

统的输入数据切分。进行每次切分就是指定一个 map 任务。每个 map 函数的输出键值对基于中间生成的键进行哈希分区。然后排序每个分区接着按照键的排序进行合并。有同样的键的分区被分配到单独的 reduce 任务中，接着 reduce 函数输出最终结果。

在 Hadoop 的 MapReduce 工作流程中，一个工作记录服务器作为主节点把数据分成几部分作为 map 阶段的输入。同时任务记录服务器作为数据节点保存 map 函数中间生成的结果到 HDFS 中。Hadoop 会基于地点计划分配 MapReduce 运算，同时提高集群整体的输入输出效率来减少运算的成本。

第三章　用户上网数据采集与处理

本章主要介绍用户上网如何留下信息，如何将上网信息转变为推荐系统所需要的训练集。其中包括了以下内容：网络流量信息采集、打开 URL 后发生了什么、如何给原数据库中没有的 url 分类、如何格式化、存储

第一节　网络信息结构及内容

互联网用户通过各类软件发送、接受信息。对于不同的软件，有不同的信息发送、接受格式, 称作协议。以大多数网络浏览器使用的基本 HTTP 协议来说，其请求的头部有以下常用的信息

表 3.1　HTTP 协议请求头部常用的信息

头部名称	描述
Accept	Content-Types that are acceptable for the response
Content-Type	The MIME type of the body of the request (used with POST and PUT requests)
Date	The date and time that the message was sent
Host	The domain name of the server (for virtual hosting), and the TCP port number on which the server is listening. The port number may be omitted if the port is the standard port for the service requested
Referer	This is the address of the previous web page from which a link to the currently requested page was followed
User-Agent	The user agent string of the user agent

根据以上 HTTP 请求头部我们可以获得用户请求网站 URL，请求时间，使用浏览器类型信息。当然对于纪录了所有网络流量包的情况，我们甚至可以解压信息包，获得网页信息。对于使用其他协议的软件来说，只要按照该协议的约定，也能正确解压信息包，获取其中内容。

第二节 网络流量信息采集与处理

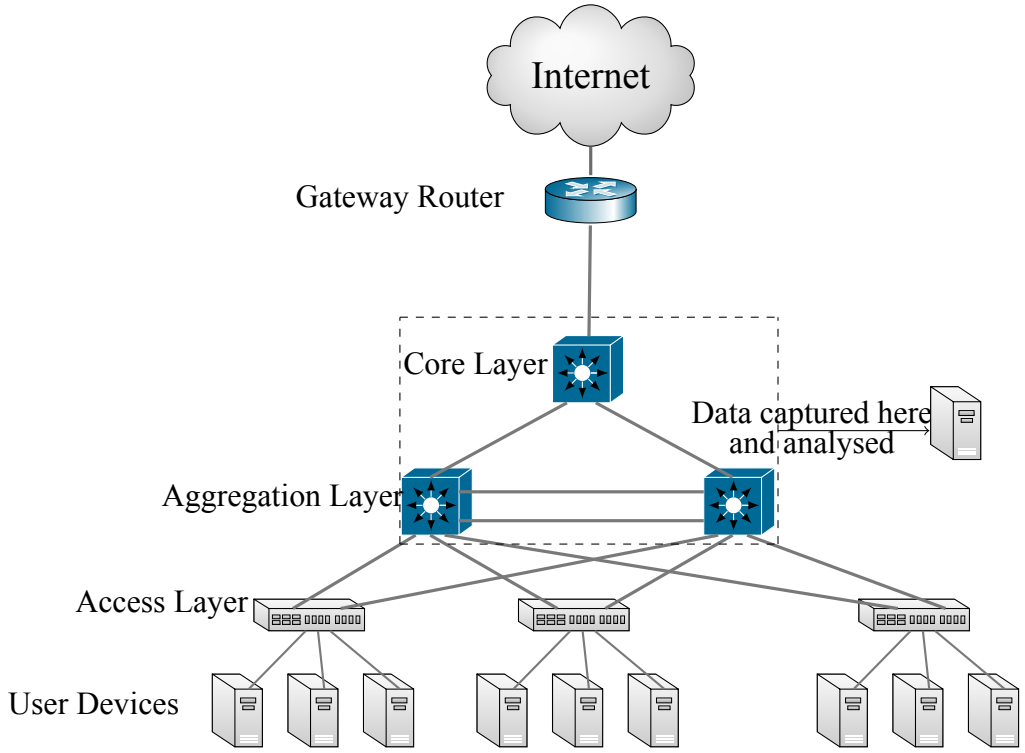


图 3.1 用户数据采集及处理

目前，国内外比较流行使用的网络流量信息采集方式有 Cisco 公司开发的 NetFlow 工具。支持 NetFlow 的路由器和交换机能够收集所有接口上 IP 交换数据，并且能把数据导出到 NetFlow 信息分析服务器上。国内研究人员也设计了类似的分布式网络流量分析系统 [20–22]。

本文根据 NetFlow 采集模式设计采集及处理模块。对于每个地区的路由器或者交换机，都连接到信息分析服务器。信息分析服务器将过滤 Content-Type 不是 HTML 类型的文件，比如 js 和 css 文件，然后记录如下信息。

表 3.2 信息分析服务器记录信息格式

User ID	Request URL	Time	Location	Content	Device

对于以上信息，本模块主要分析 Request URL 与 HTML 文件。由于 URL 包含网站名称，以及访问路径，故 URL 包含部分关键词。可将 URL 作为一个语句进行语义分析。另外，由于 HTML 文件结构可知，其中的内容镶嵌在各种 HTML 标签中。故分析 HTML 文件首先要去除各种标签。通过实现监督学习算法 [23] 去除标签，将剩下的文字内容输入语义分析模块。

按照第二章第一节描述的语义分析原理，本模块首先分析得出所有关键词。从关键词中取出所有名词，并按照出现次数进行排序。将高频名词按照以下格式输出保存。

表 3.3 信息分析服务器输出信息格式

User ID	Request URL	Time	Location	Item	Device

第三节 基于大数据处理的推荐算法

根据 Hadoop 与 MapReduce 的原理及特点，我们可以将协同过滤算法计算过程分解到 Map 阶段与 Reduce 阶段中。

首先要进行简单的计数工作，即计算每个 User-Item 对的数目。将该数目代入逻辑函数 $S(t)=1/(1+e^{-t})$ 进行简单的归一化。得到 User-Item-Value 格式的一组数据，形成下一步所需的 User-Item 矩阵，记为 $R=\{<i,j,r_{i,j}>\}$，保存于 Hadoop 的 HDFS 中。以下假设 n 个用户，m 个物品，

3.3.1 计算每个用户的平均评分

每个用户 i 的平均评分可以如下计算：

$$\bar{r}_i=\frac{\sum_{j=1}^{n} r_{i,j}}{\sum_{k=1}^{l} 1} \qquad \forall i\in[1,n]$$

其中 l 表示该用户评价了几个物品。

生成的平均评分数据集记为 $U_{all}=\{<i,\bar{r}_i>\},\forall i\in[1,n]$。具体的 Map 与 Reduce 算法如下：

Map-I: 将 i 相同的 $<i,j,r_{i,j}>$ 映射到同一个机器中，简记为 $<j,r_{i,j}>$。对于 k 个机器，本步骤的复杂度为 $o(\frac{nm}{k})$。

Reduce-I: 取出 $<i,j,r_{i,j},\bar{r}_i>$，保存到 HDFS 中，此时将 R 指向 $\{<i,j,r_{i,j},\bar{r}_i>\}$。另外保存 $U_{all}=\{<i,\bar{r}_i>\},\forall i\in[1,n]$ 到 HDFS 中。本步骤的复杂度同样为 $o(\frac{nm}{k})$。

3.3.2 计算用户间相似度

计算用户间相似度矩阵 $S=S_{i,j}|\forall i,j\in[1,n]$。定义 $U_i=\{<i,j,r_{i,j},\bar{r}_i>|\forall j\in[1,m],<i,j,r_{i,j},\bar{r}_i>\in R\}$ 为用户 i 评价过的物品集。定义 $N_{k,l}=U_k\cap U_l$，为用户 k 与用户 l 共同评价过的物品集。为了计算可行性，$N_{k,l}$ 将分到同一个节点上。具

体的 Map 与 Reduce 算法如下：

Map-II: 将 $\{U_i|i\in[1,n]\}$ 映射为 $\{N_i,j|\forall i,j\in[1,n]\}$。如果有 k 个机器，本步骤的复杂度为 $o(\frac{m^2n}{k})$。

Reduce-II: 取出 $\{S_{i,j}|\forall i,j\in[1,n]\}$，保存到 HDFS 中。本步骤的复杂度为 $o(\frac{m^2n}{k})$。此步骤可以看出可以看出，对于物品数远大于用户数的情况，本算法更适合于用户数远大于物品数的情况。

3.3.3 计算预测矩阵

将矩阵 R 变形为矩阵 I，其中矩阵 I 的数据按物品的序号排序：

$$I=\begin{pmatrix} I_1 \\ I_2 \\ \vdots \\ I_m \end{pmatrix}=\{I_j|\forall j\in[1,m]\} \tag{3.1}$$

其中

$$I_j=\begin{pmatrix} <1,j,r_{1,j},\bar{r}_1> \\ <2,j,r_{2,j},\bar{r}_2> \\ \vdots \\ <n,j,r_{n,j},\bar{r}_n> \end{pmatrix}=\{<i,j,r_{i,j},\bar{r}_i>|\forall i\in[1,n],<i,j,r_{i,j},\bar{r}_i>\in R\} \tag{3.2}$$

即 I_j 表示与物品 j 相关的所有评分数据集。定义相似度矩阵 S 为

$$S=\begin{pmatrix} S_{1,1} & \dots & S_{1,n} \\ \vdots & \vdots & \vdots \\ S_{n,1} & \dots & S_{n,n} \end{pmatrix}=\begin{pmatrix} S_1 \\ \vdots \\ S_n \end{pmatrix} \tag{3.3}$$

其中 $S_i=\{S_{i,j}|\forall j\in[1,n]\}$ 表示用户 i 与其它用户间的相似度。具体的 Map 与 Reduce 算法如下：

Map-III: 将 j 相同的 $<i,j,r_{i,j}>$ 映射到同一个机器中，简记为 $<j,r_{i,j}>$，同时将 S_i 映射到同一个机器中。对于 k 个机器，本步骤的复杂度为 $o(\frac{nm}{k})$。

Reduce-III: 取出 $B=<i,j,r_{i,j},\bar{r}_i,S_i>$，保存到 HDFS 中。本步骤的复杂度为 $o(\frac{m^2n}{k})$。

这一步是为计算预测矩阵做准备。按照第二章的公式计算用户 i 对物品 j 的评分 $p_{i,j}$。预测矩阵 P 定义为：

$$P = \begin{pmatrix} \{p_{1,1}, p_{1,3}, p_{1,5}\} \\ \vdots \\ \{p_{i,1}, p_{i,3}, p_{i,5}\} \\ \vdots \\ \{p_{n,1}, p_{n,3}, p_{n,5}\} \end{pmatrix} = \begin{pmatrix} P_1 \\ \vdots \\ P_i \\ \vdots \\ P_n \end{pmatrix} \tag{3.4}$$

其中 $P_i = \{p_{i,j} | \forall j \in [1,m], < i, j, r_{i,j}, \bar{r}_i > \notin R\}, \forall i \in [1,n]$ 表示用户 i 对未评价物品集的所有预测分数。具体的 Map 与 Reduce 算法如下：

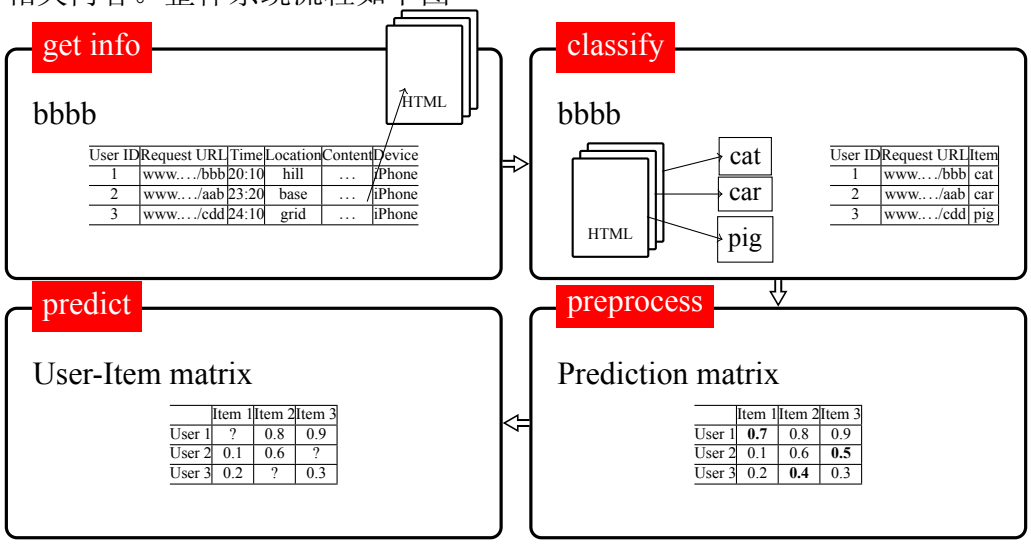
Map-IV: 将 i 相同的 $B =< i, j, r_{i,j}, \bar{r}_i, S_i >$ 映射到同一个机器中，简记为 $< j, r_{i,j}, \bar{r}_i, S_i >$，同时将 Map-I 生成的 U_{all} 中的 U_i 映射到同一个机器中。对于 k 个机器，本步骤的复杂度为 $o(\frac{nm}{k})$。将

Reduce-IV: 取出 $P =< i, j, p_{i,j}, >$，保存到 HDFS 中。本步骤的复杂度为 $o(\frac{m^2 n}{k})$。至此所有用户对未评价物品集的分数 $p_{i,j}$ 都已经计算出来。即构建了完整的 User-Item 矩阵，也即用户兴趣模型。此时对于每个用户，对所有物品按评价分数大小排序，排名前 k 的物品组成一个向量。下一步可以根据这个向量推荐用户内容。

第四章 系统工作流程与实证分析

第一节 系统整合

基于前几章的描述，我们可以综合构建一个数据挖掘系统，发现用户兴趣，推荐相关内容。整体系统流程如下图



窗口化时序数据训练，或者按照新闻衰减度给旧数据安排小的权值

第二节 真实数据测试

抱歉暂时没申请到机器以及数据

第五章　总结

本文通过描述几个算法，最后构建了一个系统。

参考文献

[1] 杨清龙. 基于网络日志的互联网用户行为分析 [D], **2013**.

[2] 史艳翠. 基于通信数据的上下文移动用户偏好动态获取方法研究 [D], **2013**.

[3] 程辉. 网络用户偏好分析及话题趋势预测方法研究 [D], **2013**.

[4] 张欢. 网络用户偏好分析方法的研究 [D], **2014**.

[5] P. Resnick *et al.* "*GroupLens: an open architecture for collaborative filtering of netnews*" [J]. In: *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, **1994**: 175 ∼ 186.

[6] G. Linden, B. Smith and J. York. "*Amazon.com Recommendations: Item-to-Item Collaborative Filtering*" [J]. *IEEE Internet Computing*, **2010**, *7*(1): 76 ∼ 80.

[7] C. A. Gomez-Uribe and N. Hunt. "*The Netflix Recommender System: Algorithms, Business Value, and Innovation*" [J]. *Acm Transactions on Management Information Systems*, **2015**, *6*(4): 29 ∼ 47.

[8] J. Davidson *et al.* "*The YouTube video recommendation system*" [C]. In: *Fourth Acm Conference on Recommender Systems*, **2010**: 293 ∼ 296.

[9] G. Adomavicius and A. Tuzhilin. "*Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions*". *Knowledge and Data Engineering, IEEE Transactions on*, **2005**, *17*(6): 734 ∼ 749.

[10] J. L. Herlocker, J. A. Konstan and J. Riedl. "*Explaining collaborative filtering recommendations*" [C]. In: *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, **2000**: 241 ∼ 250.

[11] J. Zhang and P. Pu. "*A recursive prediction algorithm for collaborative filtering recommender systems*" [C]. In: *Acm Conference on Recommender Systems*, **2007**: 57 ∼ 64.

[12] Y. Shi, M. Larson and A. Hanjalic. "*Exploiting user similarity based on rated-item pools for improved user-based collaborative filtering*" [C]. In: *Proceedings of the third ACM conference on Recommender systems*, **2009**: 125 ∼ 132.

[13] T. Kiss and J. Strunk. "*Unsupervised multilingual sentence boundary detection*" [J]. *Computational Linguistics*, **2006**, *32*(4): 485 ∼ 525.

[14] K. Gimpel *et al.* "*Part-of-speech tagging for Twitter: annotation, features, and experiments*" [C]. In: *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*, **2011**: 42 ∼ 47.

[15] B. O. Owoputi *et al.* "*Improved part-of-speech tagging for online conversational text with word clusters*" [J]. *Proceedings of Naacl*, **2015**.

[16] L. Derczynski *et al.* "*Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data*" [C]. In: *Recent Advances in Natural Language Processing - RANLP*, **2013**.

[17] J. Dean and S. Ghemawat. "*MapReduce: Simplified Data Processing on Large Clusters.*" [J]. *In Proceedings of Operating Systems Design and Implementation (OSDI*, **2004**, *51*(1): 107 ∼ 113.

[18] S. Ghemawat, H. Gobioff and S. T. Leung. "*The Google file system*" [C]. In: *Acm Sigops Operating Systems Review*, **2003**: 29 ∼ 43.

[19] F. Chang *et al.* "*Bigtable: A Distributed Storage System for Structured Data*" [J]. *Acm Transactions on Computer Systems*, **2008**, *26*(2): 205 ∼ 218.

[20] 乔媛媛. 基于 *Hadoop* 的网络流量分析系统的研究与应用 [D], **2014**.

[21] 延皓. 基于流量监测的网络用户行为分析 [D], **2011**.

[22] 董超. 基于网络流量监测的移动互联网特征研究 [D], **2013**.

[23] C. Kohlschütter, P. Fankhauser and W. Nejdl. "*Boilerplate Detection Using Shallow Text Features*" [C]. In: *International Conference on Web Search and Web Data Mining*, **2010**: 441 ∼ 450.

致谢

感谢您使用本模板。