第二十六届“冯如杯”学生学术科技作品竞赛项目论文

基于 GitHub 平台的数据挖掘与分析

——开发者兴趣圈挖掘

二〇一七年二月

摘要

近年来，GitHub 作为最火热的开源代码库及分布式版本控制系统，吸引了成千上万的开发者加入。每位开发者可以关注其他开发者，可以查看一些自己感兴趣的项目。本项目的目的在于给每位开发者找到与自己比较相似的其他 GitHub 开发者，根据共同关注的开发者、共同 fork 并做出贡献的开源项目深度挖掘分析开发者之间的关系链。通过爬虫技术挖掘 GitHub 开发者所有相关的开源数据，综合运用 Sql 数据库技术、代号编码思想、字典树哈希表数据结构等，实现开发者兴趣圈挖掘的任务。研究得出，开发者之间存在比较稠密的关系网。

关键字：GitHub 兴趣圈，开发者大数据挖掘，字典树

Abstract

In recent years, GitHub as the hottest open source code library and distributed version control system, attracting thousands of developers to join in. Each developer can follow other developers and view some open interesting projects. The purpose of our project is to find the similar developers for every GitHub's developer, according to the number of common following developers and common forking open projects deeply digging and analysing the relation-chain between all developers. Through the crawler technology to dig all relevant developer's open source data. Making use of the Sql database' technology, code coding ideas, trie and hash table data structures to achieve the task of mining the interest circle of developers. The study concludes that there is a dense net of relationships among all the GitHub developers.

Keywords：Interest circle of GitHub，Developers' big data mining，Trie

目录

第一章 项目简介

1.1 项目背景

有句话说：“程序员的指尖有改变世界的力量”。近年来，越来越多的人参与到软件硬件开发的潮流中。GitHub 作为最火热的开源代码库及分布式版本控制系统，吸引了成千上万的开发者加入。据不完全统计，目前 GitHub 拥有的开发者已经超过 1000 万，托管的项目已有千万个，每天都有大批次的新开发者加入 GitHub 大家庭，通过下图 1 的 GitHub 开发者社区用户数量和托管项目的增长趋势就可知道其火热趋势。在 GitHub 上，每一位开发者的信息都是开源的，他的注册名，他托管的所有项目其他开发者都可以看到。

在 GitHub 这个开源的世界里，每位开发者都可以 follow 他自己敬佩喜欢的其他开发者，实时关注他们最新的开发动态；也可以在 GitHub 海量的开源项目中，找到自己感兴趣的，fork 到自己的代码仓库里，进行代码学习、开发和维护。

经常使用 QQ 或者新浪微博的人都知道，QQ 有一个叫做可能认识的人推荐(很大程度上基于两个人之间共同好友的数量)；而新浪微博，则是根据你所关注的其他用户，推荐他们共同关注的人，这些功能让很多人受惠，找到了他们极有可能感兴趣或者相似的用户。但是目前对于 GitHub 的用户来说，这种推荐技术目前还没有。我们项目正是基于此而展开的，依据 follow 开发者以及 fork 开源项目的共同相似性，对每一位开发者，去寻找所有与之存在相关联的开发者群。

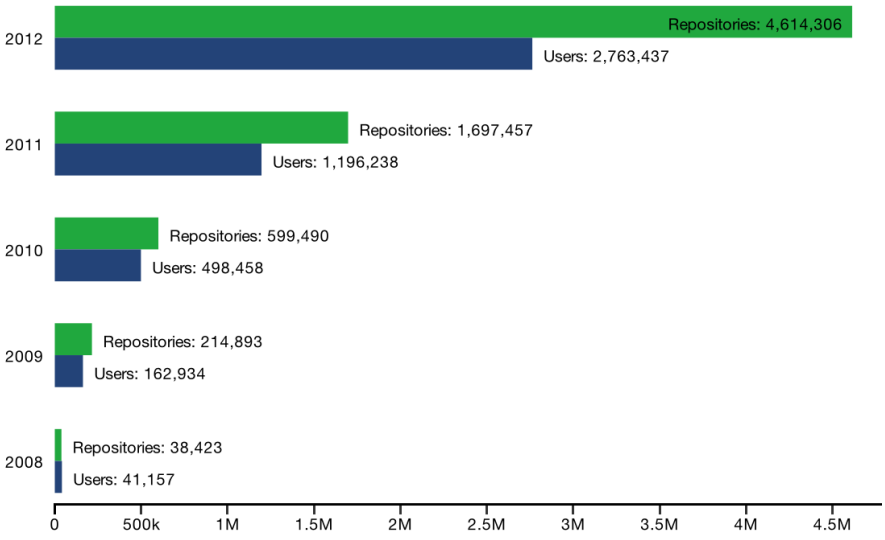

图 1 GitHub 开发者和项目增长情况

1.2 项目制作的目的与意义

我们基于 GitHub 数据开源的特征，挖掘所有用户所 Follow(关注)的开发者以及其参与过的所有的开源项目(Fork 的项目)。对于挖掘到的庞大的数据进行深度分析，找出所有与给定的开发者具有关系的其他开发者，按照相似度高低排序，做相似性推荐，提高 GitHub 开发者之间交流程度，方便开发者找到与自己志趣相投的人，增强 GitHub 灵活性，提高 GitHub 的社交性，更好的促进用户在 GitHub 上的开发工作，为 GitHub 程序员社区增添活力。

第二章 数据挖掘与存储

2.1 Python 网页爬虫

项目实现基于的数据全部来源于开源平台 GitHub，用户数据的开源性使得我们可以很方便地通过网页爬虫技术抓取到有效数据。Python 作为面向对象的解释型语言，具有小巧灵便易实现的特性，所以我们采用了 Python 作为我们挖掘原始数据的编程语言。

项目采用的是开源的 Python 网页数据爬虫库 urllib 和 urllib2，利用成熟的爬虫框架 Scrapy，根据 GitHub 每位开发者 Profile 网页界面 Html 标签内容，寻找到有关 Followers、Following 的全部信息，并依据开发者代码仓库中项目来源辨识哪些项目是从其他开发者代码仓库 Fork 来的。

由于 GitHub 界面数据的复杂性，挖掘得到的用户数据包含了注册邮箱、开发者注册日期、开发者所在地、昵称等多种暂时不需要的信息，为了避免一开始写入大量杂乱数据到 MySQL 数据库中，我们选择 CSV 文件格式存储得到的原始未去噪的数据。由于 Python 内置了处理 CSV 文件的模块，对我们的数据存储带来极大的方便。

最后挖掘得到的用户原始数据达 10GB，Follow 关系集有一千多万记录，Fork 项目集达到四千五百万记录。基于挖掘结果可知，项目面对的数据规模是特别大的。开发者的数量之多决定了分析这些数据的意义很大。

预估计，每位开发者都具有很多与自身有共同 Follow 或者共同 Fork 项目的其他开发者。

2.2 数据去噪存入数据库

第一，针对用户基本信息，在分析过程中，我们实际只需要知道 GitHub 用户名即可，例如 jmettraux、tosch、kennethkalmer 等。所以针对用户名数据信息我们只需建立一个两字段的数据表即可。由于用户名的长度与格式的多重性，以及考虑到后期采用字典树数据结构的深度优先遍历复杂度，每一个用户名都对应一个正整数代码。并创建一个该表的副本，将用户名作为数据库的主键，为了加快数据库的访问速度。运用 MySQL 自身内置的主键排序、建立 BTree 索引的功能，将两个表的数据进行排序。

第二，针对 Follow 的关系的数据，数据库中每行记录的格式设为 A-B 类型，A 代表一位开发者，B 代表另一位开发者，A、B 均为 Integer 类型，具体代表的开发者姓名对应于用户数据表中同整数值的名字。对此，我们同样建立两个数据表，一个数据表记录存储为 A following B，另一个记录存储为 A 被 B Follow，每个数据表均以 A 为主键。

第三，针对 Fork 项目的数据，同样采取针对 Follow 数据的存储方法录入有效数据进 MySQL 数据库。

第三章 算法与数据结构设计

3.1 JDBC 连接 MySQL 数据库

JDBC(Java Data Base Connectivity)，是一种用于执行 SQL 语句的 Java API，连接效率高、查询数据库得到结果集的速度快。由于是直接调用 SQL 语句在后台执行，性能极佳，所以我们连接数据库采用 JDBC[1]。

3.2 算法与 Trie 树设计

由于使用的是 MySQL 数据库，且 SQL 已经根据数据表的主键以及设置的排序规则将所有记录有序化，以 BTree 作为索引，以 BinarySearch(二分查找)作为高效查找算法，因而在数据查询上使用数据库会非常的高效。正因为为了极大程度的让数据库采用时间复杂度最小的折半查找算法，在设计数据库的时候，我们都会复制一份表的备份，将主键换为另外一个字段重新排序，在双向查询数据

时不会因为所选择字段没有排序而造成数据库不采用折半查找算法。

分析一个开发者与其他开发者共同 Follow 了哪些人与分析一个开发者同其他开发者共同 Fork 了哪些开源项目的解决方案实际上是一样的，设计的算法与数据结构能够解决 Follow 关系点的分析就同样可以解决 Fork 关系点的分析。

以解决共同 Follow 关系问题为例，首先得到用户给定的一位开发者名，根据开发者名搜索数据库，得到开发者对应的整数值编号。依据得到的整数值编号，搜索为 Follow 关系数据建的数据表，得到该开发者所有 Follow 的开发者编号集。遍历该编号集，每次对应一个整数编号，找到其所有 Follow 他的其他开发者编号集，将该编号集插入到 Trie 树中[2]，如果有新增结点，设置统计量为 1；如果是插入原有结点，统计量加一即可。之所以利用字典树，是因为开发者编号长度最多不超过八位，在做查询的时候复杂度极低，而且也避免了完全二叉树、Hash 表的不稳定性，兼顾了稳定与速度。字典树完全建立后，利用 DFS(深度优先遍历)算法将字典树上所有的有效结点取出放入一个动态数组中。根据实际情况，与给定开发者只具有一个共同 Follow 的其他开发者占总结果集的大多数，为了避免排序的大量消耗，首先采用二路划分算法，将只具有一个共同 Follow 关系的数组元素全部移至数组前端，再使用快速排序算法对后端数据进行排序，最终得到排序结果，将其写入数据库或者 Excel。

3.3 算法性能

经过对多个开发者相关性数据测试，大部分开发者的与自身有共同 Follow 的其他开发者数目在一千以下。依据我们的测试结果，平均运行时间在 500ms 左右，如下图 3 对开发者 tosch 的数据分析结果。对于个别具有一万以上级别数据相关用户的情况，平均运行时间在 3s 左右，如下图 2 对开发者 kennethkalmer 的数据分析结果。

由此可以看出项目核心算法与 Trie 树数据结构的构建是非常成功的，能够在短时间内获得最终结果。而对于 Fork 数据集的测试，同样的处理模式，平均运行时间在 1s 左右，关系集相对庞大一点。除对于极少部分超规模数据分析得出最终结果稍微有一点慢之外，总体而言项目的算法设计还是较成功的，对于移植到其他平台数据分析上或仍基于 GitHub 平台其他的数据分析具有很大潜力。

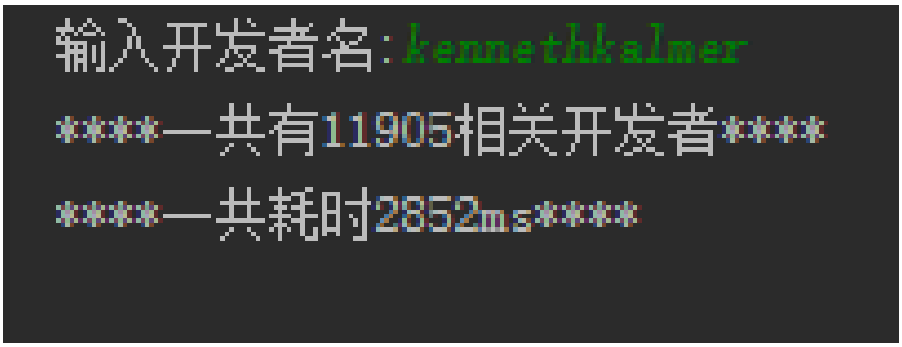

图 2　极端数据测试

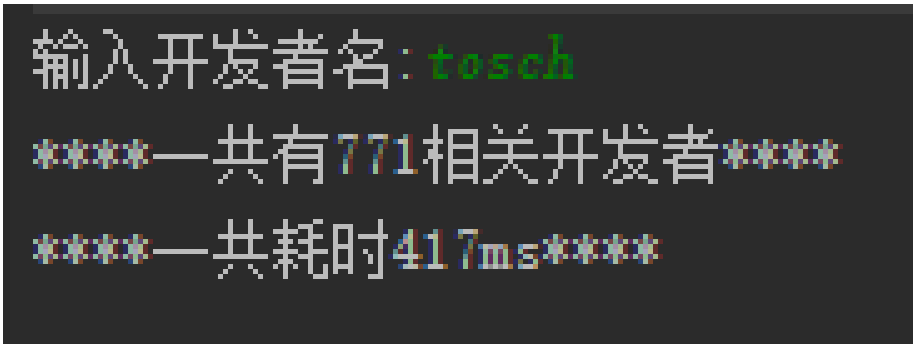

图 3　正常数据测试

第四章 网站设计与简易交互

4.1 JSP 架构设计网站

对于每位开发者，数据分析后的结果写入微软开发的 Excel 表格中。根据对大量开发者相似性挖掘可知，基本上排序前 1000 的开发者与给定开发者之间的相似性最为紧密，1000 以后的开发者与给定开发者之间的相似程度都是比较低的，而且考虑到用户浏览疲劳，给出过多的相关开发者用户不一定选择去关注，因而我们对结果集择优录入，只保存排序前 1000 的开发者。

交互式设计是本项目的重点，项目的宗旨是为了服务 GitHub 上的开发者，所以我们采用网站的形式用户提供交互式体验。搭建网站有很多的技术，比如 ASP、ASP.NET、JSP 等。考虑到对 Excel 表格最佳支持，我们选择 JSP 作为网站架构设计的框架，以 Tomcat 作为服务器，利用 Java 开源 API Jxl 实现对 Excel 表格数据的读取。

经过测试，数据显示到网站前端的效率非常高，几乎可以在网站打开的瞬间将数据读取出来并前端回显。

基于开源开放的特性，使用本网站不需要注册，只需在前端界面的文本输入框内输入所需查询分析的开发者名即可，如下图 4 所示。

后端 Java 读取数据库，验证所查询的开发者是否存在，若存在，Java 读取该开发者对应的 Excel 表格，将表格内容显示到网页上，并提供开发者下载该 Excel 文件的窗口，如下图 5 所示。

当然，如果开发者希望知道自己与其他开发者所共同 Follow 的人或者共同 Fork 的项目具体是什么，只需点击链接，便可获得最详细的结果。

目前由于项目搭建的网站只是用于测试，所以网站的语言架构采用的是中文，还未发布英文版。由于 GitHub 的开发者遍布全球各地，因而最终提交服务器的会是英文版本。

图 4　网站开始界面

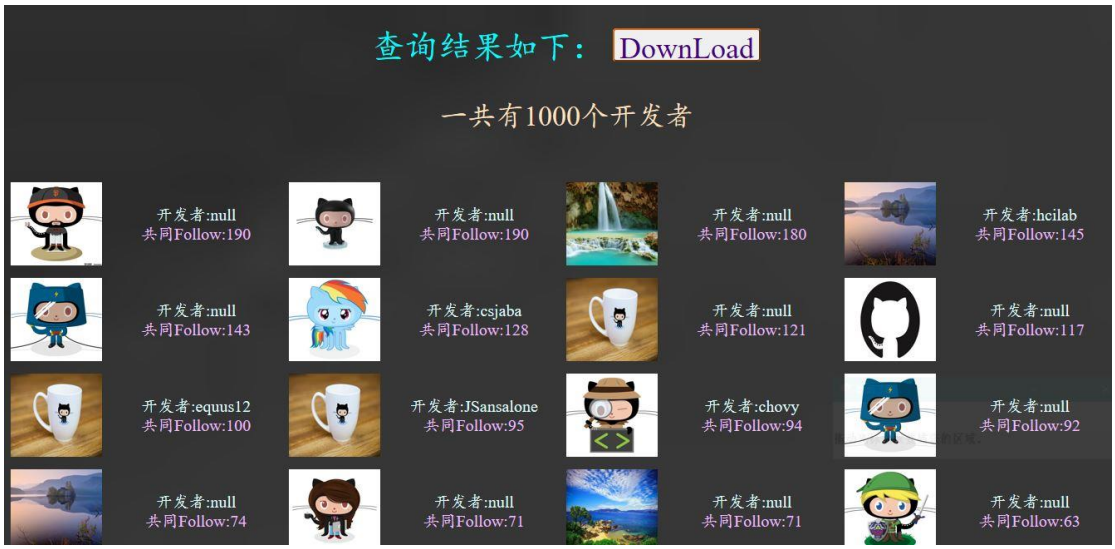

图 5　用户查询结果界面

第五章 后期规划

5.1 拓展目标

目前项目的相似性分析基于的数据是共同 Follow 和共同 Fork 的项目，而对于 GitHub 来说，还有更多的数据等待挖掘分析。每位开发者都有使用不同编程语言的习惯，每位开发者对项目贡献提交的时间点和次数不同，每位开发者对其他项目给予点赞数目也不相同，这些都是可以作为兴趣圈相似性挖掘的数据依赖。后期项目将进一步挖掘这方面的数据，通过聚类分析从不同的角度发现 GitHub 开发者之间的相似度。

结论

在大数据时代，通过数据挖掘与聚类分析，可以得到很多实际有效的价值信息。通过对开源平台 GitHub 上用户数据的挖掘与分析，我们可以发现全世界的开发者之间存在很多的相似性。提供这种数据分析服务，将使得原本社交活力不足的平台获得源源不断的社交动力。数据无时不在，对其挖掘分析将创造无穷的价值。

参考文献

[1]明日科技. Java Web 从入门到精通[M]. 北京: 清华大学出版社, 2012.9

[2]张启飞,吴杰义等. 利用频率特征的 Trie 树索引快速构造算法[A]. 北京邮电大学学报, 2013, (2)