Recommendit

A Recommendation Service for Reddit Communities

Suphanut Jamonnak, Jonathan Kilgallin, Chien-Chung Chan, En Cheng
Department of Computer Science
The University of Akron
Akron, OH

Abstract—Reddit.com is a popular website consisting of many online forums, or "subreddits", each centered on a particular topic. Many of these subreddits represent, prima facie, closely related topics. However, two communities that might appear to be similar may in fact have little overlap in their user-base or content, while apparently dissimilar subreddits may actually represent topics that appeal to such similar audiences that their users do substantially overlap. At present, there is no generally effective method of identifying these relations. We propose and implement an automated method of identifying such relations using association rule mining, and show that the results from this system can be used to create a web services that a user can query to find subreddits associated with a subreddit representing a given topic.

Keywords—Data mining; Association rule mining; Reddit.com; Internet systems

I. INTRODUCTION

Reddit is a social network and content aggregation site launched in 2005. Users create posts containing text, photos, or links, to a particular "subreddit" - a forum on the site centered around a single topic. As of April 2015, Reddit had over eight thousand subreddits, ranging across a wide variety of popular topics such as "jokes", "art", "world news", "gaming", and "recipes", as well as more esoteric subjects like "haskell", "doctorwho", and "cyber laws". Users may view posts recently submitted to a subreddit, and can also contribute to a crowd-based ranking system by voting posts up or down to indicate approval or disapproval. A user may also comment on a post and perform, on the comments, the same actions that can be done with the post itself. The interactive nature of the site can best be illustrated by visiting the site directly - for example, at "www.reddit.com/r/python" for the forum about the Python programming language, though note that some actions reqire account creation and login.

The Reddit homepage consists of a feed of popular and recent posts across a set of different subreddits. The nature of the set of included subreddits depends on whether a user is logged in or not. A user browsing anonymously (i.e. without logging in) will see posts from among a select group of "default" subreddits, which are hand-picked by the site administrators according to the subreddit's perceived quality and applicability to a broad audience. If a user is logged in, however, the front page will show primarily content from subreddits that the user has elected to subscribe to. A user can subscribe or unsubscribe from any subreddit from either the subreddit itself or from a list of subreddits available through a personal settings menu. A particularly popular and recent post from a subreddit the user has not subscribed to may also occasionally appear on their front page, but generally, the relevance, quality, and quantity of posts that appear in this feed depend on the user's selection of an appropriate set of subreddits. As the feed of posts on the front page is so integral to the site, and as the site entertains hundreds of millions of visitors per month, the task of selecting a set of subreddits is important for users' engagement on the site.

II. RELATED WORK

The process of discovering subreddits requires a user to actively search and browse through a list of subreddits, or else to come across links to new subreddits more organically through other posts on the site or elsewhere. Such links may regularly be found within comments to a post in a related subreddit. For example, a user from the United Kingdom requesting help from the "/r/legaladvice" forum may likely receive a comment directing him to "/r/legaladviceuk". A subreddit's main page may list a limited set of ostensibly related subreddits, as determined by the moderators of that subreddit. For example, toward the bottom of the sidebar on the r/python page, a list is included as shown in Figure 1 below. With only this system, a user could go months or years on the site remaining unaware of the existence of a subreddit pertinent to the user. Furthermore, in any case, all of these discovery methods rely on input from other users, which may at any time be incomplete or out of date.

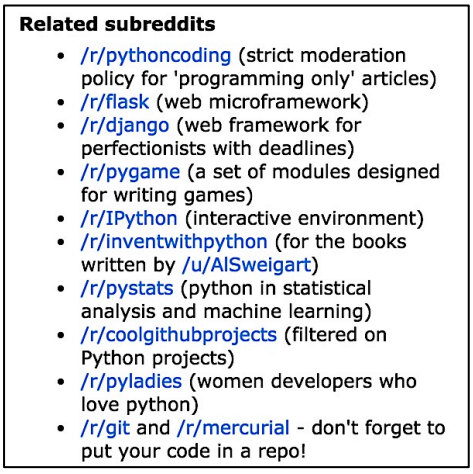

Fig. 1. Subreddits related to /r/python

Some third-party sites aim to facilitate this discovery process. For example, the site “subreddits.org/” and other third-party sites group subreddits by category; again, the site development team among other shortcomings largely builds these by hand. At least a few projects do aim to programmatically build recommendation engines though. In one case, Sundaresan et al build weighted graphs to identify communities on the site [1]. A github project - also called “Recommendit” by “ben444422”, with no updates in 2 years, appears to have begun on a similar path to the one we intend, but did not seem to get very far [2].

III. METHODOLOGY

To facilitate the process of discovering subreddits pertinent to a user, we propose an association-rule based recommendation engine, associating two subreddits when a large number of individual users post to both. The degree to which the user-base overlaps provides raw insight into the degree to which the topic interests overlap themselves.

The general process as a data mining solution consists of identifying “transactions”, each containing multiple “items”, and selecting those items that appear together in sufficiently high proportion. In our case, an “item” is a subreddit and a transaction is a set of subreddits that a given user frequently posts to. By mining reddit for many users’ post history, we identify and associate subreddits that have several joint users by using the “a priori” association rule-mining algorithm, with appropriate support and confidence thresholds (as discussed in section V). From the rules produced, we develop a web application that allows a user to input a subreddit of interest, and returns related subreddits, which may be of interest to the user as well.

To implement this approach to identifying similar subreddits, we perform four steps, further illustrated in Fig. 2:

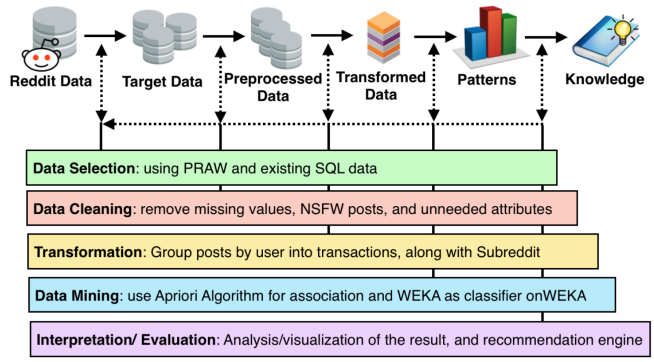

Fig. 2. Recommendit Methodology

- For each of the top 2500 subreddits by subscriber count, as obtained from redditlist.com [3], we scrape Reddit to obtain the set of users that have made a popular post to the subreddit this year. This is done with a Python program using Reddit’s API [4] and is stored in a CSV file for each subreddit. This dataset is augmented with the top 1000 posts of all time as of August 2014, downloaded from [5]. See "Data Collection" below for more information on this part.
- We then strip the irrelevant attributes and unusable records from the data collected in step 1, and group records by author in order to form a list of transactions as described in our formulation above. This is done by loading the CSV files into SQLite and using SQL queries to transform the data.
- We next pass the list of transactions to an implementation of the a priori algorithm written in Python [6], in order to find the list of association rules.
- Using the association rules discovered, we host a simple webapp, also written in Python and using the CherryPy library [7], that allows a user to query for the set of subreddits associated with a given input subreddit.

This process, taken together, allows users to find subreddits of interest. Due to the growing and shifting nature of Reddit communities, steps 1 to 3 should be repeated periodically - perhaps on the order of once a month.

IV. DATA COLLECTION AND PROCESSING

A. *Data acquisition*

The data we require consists of the author and subreddit for as many posts as we can collect, store, and process. There are many ways to obtain such data - from Reddit itself we can scrape the HTML site or use their provided API. In this case, we have used a Python library called “Python Reddit API Wrapper”, or PRAW [8]. PRAW is a python package that allows for simple access to Reddit’s API. It aims to be easy to use for us and is also designed to respect all of Reddit’s API rules. Another approach, which we also make use of, is to rely on data previously collected by others and hosted elsewhere. Fortunately, an existing github project provides an older, but still usable, copy of exactly the data we are interested in - post information from the top 2500 subreddits by subscriber count [5].

We created a crawling application implemented in python using PRAW called “subrtime.py”. Subrtime.py will take as input a subreddit’name, along with start and end timestamps. Using this, it will crawl all of the subreddit’s posts in the range of the given timestamps, and return the data and attributes shown in Table 1 below. After collecting posts from the top 2500 subreddits, we are left with 2.5 million posts.

B. *Data preprocessing*

In this step, we clean the data to preserve only the posts we are able to use. Specifically, in order to create the recommendation system, Recommendit requires the author name in order to compute a transaction set for the a priori algorithm. Thus, we have to remove posts with “none” or “NULL” values for the author. Moreover, we are also removing posts marked “Not Safe For Work”, or NSFW. These are posts with adult or explicit material, which in general the site allows but we wish to disregard. We have removed the missing-author and NSFW posts by writing another python script "transubreddit.py" to filter the data in the SQL library. This decreased the number of records from 2.5 million to 2.4 million, or about 4% of our original data set.

C. Data transformation

In order to apply the a priori algorithm, these records need to be grouped by author into sets of subreddits the author posts to. This is accomplished with SQLite and SQL queries to group the records. The nature of the transformation is illustrated in the next section below.

V. A PRIORI APPLICATION

Discovering interesting relations between variables in large databases is a popular and well-researched problem in data mining. These techniques are very popular in several fields such as marketing strategy, data analytics, machine learning and so on. For instance, in marketing strategy, association rule mining may be used for discovering patterns of purchase behavior in large-scale transaction data recorded by Point-Of-Sale (POS) systems in supermarket.

For example, {onions, potatoes}⇒burger found in sales data of a supermarket can indicate that, if a customer buys onions and potatoes together, they may be also likely to buy a burger. Such information can be used and adapted for several marketing activities such as promotional pricing, product placements, and so on.

A priori is a classic algorithm for solving and finding association rules, introduced by Agrawal and Srikant in 1994 [9]. This technique is popular for this type of data mining problem, and although it is not the only algorithm for solving this problem, it is sufficient for our case. We address only the necessary concepts for our application here, but further information on the algorithm can be found in Agrawal [9] or in many texts or other resources on data mining.

In Agrawal [9], an association rule is defined as follows:

- Let $I = \{I_1, I_2, \ldots, I_m\}$ be a set of m binary attributes called items
- Let $T = \{T_1, T_2, \ldots, T_n\}$ be a set of transactions called the database.

Each transaction in T has a unique transaction ID and contains a subset of the items in I. A rule is defined as an implication of the form:

$$(X \Rightarrow Y) \text{ Where } (X, Y) \subseteq I \text{ and } (X \cap Y) = \emptyset.$$

The sets of items (for short itemsets) X and Y are called antecedent (left-hand-side or LHS) and consequent (right-hand-side or RHS) of the rule respectively. To illustrate this concept, we use this concept with a small hypothetical example from reddit. The set of subreddit items is I = {python, compsci, django, java} and a small database is coded with 0 as absence and 1 is presence. From the table below:

Transaction ID	*Python*	*Compsci*	*Java*	*Django*
Author1	0	1	0	1
Author2	1	1	0	1
Author3	1	1	0	1
Author4	0	0	1	1
Author5	1	1	0	1
Author6	1	1	1	0

TABLE I. ITEMSET DATABASE AND STRUCTURE

We can predict that a user interested in *Python* and *Compsci* is also interested in *Django* as well. In larger cases of association rule mining, an item set may need to appear together many times before it can be considered significant. Our dataset contains millions of transactions, and so constraints on the associations made are needed. The best-known constraints are minimum thresholds on support and confidence, as defined below:

- *Support* - The support supp(X) of an item-set X is defined as the proportion of transactions in the data set which contain the item-set.

supp(X) = number of transactions which contain the item-set X / total number of transactions

In the example item-set database above, the item-set {*python, compsci, django*} has a support of 3 /6 = 0.50 since it occurs in 60% of all transactions.

- *Confidence* - The confidence of a rule is defined:

$$\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y) / \text{supp}(X)$$

For the rule {python, compsci} => {django} we have the following confidence: supp({python, compsci, django}) / supp({python, compsci}) = 0.50 / 0.66 = 0.75. This means that for 75% of the transactions containing python and compsci the rule is correct.

A priori uses a tree structure and breadth-first search to count candidate itemsets efficiently. By searching all of the items in the transaction database. It generates candidate itemsets of length K from itemsets of length K-1. Then it prunes the candidates, which have an infrequent sub pattern. This makes it effective for exploring transactions with a large number of items and transactions. In our project, we make use of an existing implementation of the a priori algorithm in Python, which takes the transaction list along with support and confidence thresholds, and outputs the set of subreddits associated with each subreddit in the dataset [10].

VI. RECOMMENDATION SERVICE

The output of the a priori algorithm now gives us the data we need to host *Recommendit* as a Python-based web application. A simple online version is, at the time of this writing, hosted at “kilgallin.com”. This version allows a user to enter the name of a subreddit in a webform, and returns a formatted list of subreddits with links directly to the subreddit so that a user can quickly examine the forum and determine if it is actually of interest. It is suitable to be used as a web service or integrated into a richer application. It is important to note, however, that the underlying data will quickly become stale as the reddit communities evolve, and repeated mining would be necessary to maintain such a service.

VII. Results

With the web application hosted, we can now easily examine the results of our mining process and evaluate their usefulness to the Reddit community at large. Looking at the subreddits related to "mycology", for example, we see the following list:

- whatsthisplant
- gardening
- botanicalporn
- entomology
- biology
- spiders
- botany
- permaculture

These results certainly capture multiple types of association with a mycology forum. Some entries, such as "/r/whatsthisplant", "/r/entomology", and "/r/spiders", have in common the feature of containing many posts containing photos of their respective category of natural plants and animals. Some, such as "/r/gardening" and "/r/permaculture", may appeal more to enthusiasts of growing mushrooms for culinary use. Finally, "/r/biology" and "/r/botany" both capture broader categories that subsume the field of mycology. Compare this to the "related subreddits" list actually appearing on the mycology subreddit:

- /r/Slimemolds
- /r/seedstock
- /r/Mycoporn
- /r/MushroomPorn
- /r/Gardening
- /r/PhysicGarden
- /r/InvasiveSpecies
- /r/MushroomGrowers

This list consists of subreddits more directly addressing the study of fungi, with only "gardening" appearing on both. One may question why these would not show up in our web service. The reason for this is that many of these are smaller subreddits, which do not make the top-2500 cutoff, which is certainly a limitation of our results, as discussed below. Conversely, though, the related subreddits listed omit several subreddits of potential interest to a mycology enthusiast, such as botany. These lists thus serve to complement each other effectively.

VIII. Future Work

While our process appears to be sound for the data we collect, there are some limitations that could be overcome in future work. First, by only looking at the 2500 largest subreddits, we miss out on the ability to suggest new or small subreddits. Since these are the subreddits that are typically the hardest to discover on one's own, this is a potentially valuable application of our program - perhaps even one of the most valuable from a discovery viewpoint, and one that could help smaller communities to grow effectively. However, since the 2500th subreddit has only 0.01% the user count of the top few subreddits, the next 6000 subreddits will, cumulatively, only correspond to the number of users in the top 4-6 subreddits. The overhead to scrape and store the data for these subreddits is thus not justified for the scope of this project. Second, our web app architecture as written is limited in its ability to serve a large amount of traffic. Since Reddit has several million unique daily visitors and hundreds of millions of unique monthly visitors, were this application to gain traction among the userbase, this would be a significant problem that would require, at the least, additional servers and a load-balancing solution. Again, though, this concern is beyond the scope of our project as it stands.

IX. Conclusion

We have shown that association rule mining can be a useful tool for identifying relations between subreddits, and that this can be used to facilitate the process of discovering additional Reddit communities of potential interest to a user. While there is ample room for future development of this technique, the results we obtain are already indicative of a promising automated method for such associations.

References

[1] Sundaresan et al. Subreddit Recommendations with in Reddit Communities, Stanford University. http://web.stanford.edu/class/cs224w/projects/cs224w-16-final.pdf

[2] "ben444422". github.com. https://github.com/ben444422/Recommendit/

[3] Reddit list. redditlist.com

[4] Reddit API documentation. https://www.reddit.com/dev/api.

[5] "Umbrae". *Reddit List of Top 2.5 Million Posts.* https://github.com/umbrae/reddit-top-2.5-million

[6] "asaini". github.com. https://github.com/asaini/Apriori.

[7] CherryPy. http://cherrypy.org/

[8] PRAW. praw.org

[9] Rakesh Agrawal and Ramakrishnan Srikant. *Fast algorithms for mining association rules in large databases*. Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994.

[10] Apriori Algorithm. http://en.wikipedia.org/wiki/Apriori_algorithm